

FAST PARALLEL VOLUME VISUALIZATION ON CUDA TECHNOLOGY

ADESHINA ADEKUNLE MICHEAL

A thesis submitted in
fulfilment of the requirement for the award of the
Doctor of Philosophy



PTTA UTHM
PERPUSTAKAAN TUNKU TUN AMINAH

Faculty of Computer Science and Information Technology
Universiti Tun Hussein Onn Malaysia

November 2013

UNIVERSITI TUN HUSSEIN ONN MALAYSIA

STATUS CONFIRMATION FOR DOCTORAL THESIS

FAST PARALLEL VOLUME VISUALIZATION ON CUDA TECHNOLOGY

ACADEMIC SESSION: 2012/2013

I, **ADESHINA ADEKUNLE MICHEAL**, agree to allow this Doctoral Thesis to be kept at the Library under the following terms:

1. This Doctoral Thesis is the property of Universiti Tun Hussein Onn Malaysia.
2. The library has the right to make copies for educational purposes only.
3. The library is allowed to make copies of this report for educational exchange between higher educational institutions.
4. ****Please Mark (✓)**

CONFIDENTIAL

(Contains information of high security or of great importance to Malaysia as STIPULATED under the OFFICIAL SECRET ACT 1972)

RESTRICTED

(Contains restricted information as determined by the organization/institution where research was conducted)

FREE ACCESS

Approved by,

(WRITER'S SIGNATURE)

(SUPERVISOR'S SIGNATURE)

Permanent address

Supervisor's Name

3B, Akorede Street,
Oke Shaje, Abeokuta,
Ogun State, Nigeria

Assoc. Prof. Dr. Rathiah Hashim

Date: _____

Date: _____

NOTE:

****** If this Doctoral Thesis is classified as CONFIDENTIAL or RESTRICTED, please attach the letter from the relevant authority/organization stating the reasons and duration for such classifications.

I hereby declare that the work in this project report is my own except quotations
and summaries which have been duly acknowledged

Student :
ADESHINA ADEKUNLE MICHEAL

Date :

Supervisor :
ASSOC. PROF. DR. RATHIAH HASHIM



PTTAUTHM
PERPUSTAKAAN TUNKU TUN AMINAH

To my beloved family



PTTA UTHM
PERPUSTAKAAN TUNKU TUN AMINAH

ACKNOWLEDGEMENT

I thank Allaah (S.W.T) for making my thoughts and my dreams come to reality. I appreciate my supervisor, Assoc. Prof. Dr. Rathiah Hashim for accepting me as her Ph.D. student and standing firm beside me throughout the duration of my study. I am indeed grateful for your support. Dr. Noor Elaiza Abdul Khalid, my co-supervisor from the Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, I appreciate your support and those valuable advices in the course of the study. Many thanks to the Department of Surgery, University of North Carolina, Chapel Hill, United States, for making all the brain datasets available for this research.

My wife, Ruqqayah Lola Adeshina has always been with me even when I was almost in confusion with situations. My children, Firdaous Ifemadewa Adeshina, Mutmahina Ifedolapo Adeshina and Tasliymah Ifedunmade Adeshina, I am indeed grateful to you all for your support and endurance especially during my implementation stage that I usually feel like seeing nobody around me.

The entire academic and non-academic staff of the Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia, you have contributed in one way or the other towards my success, you are a wonderful team. The community of postgraduate students, thank you for accommodating me in your midst.



ABSTRACT

In the medical diagnosis and treatment planning, radiologists and surgeons rely heavily on the slices produced by medical imaging scanners. Unfortunately, most of these scanners can only produce two dimensional images because the machines that can produce three dimensional are very expensive. The two dimensional images from these devices are difficult to interpret because they only show cross-sectional views of the human structure. Consequently, such circumstances require highly qualified doctors to use their expertise in the interpretation of the possible location, size or shape of the abnormalities especially for large datasets of enormous amount of slices. Previously, the concept of reconstructing two dimensional images to three dimensional was introduced. However, such reconstruction model requires high performance computation, may either be time-consuming or costly. Furthermore, detecting the internal features of human anatomical structure, such as the imaging of the blood vessels, is still an open topic in the computer-aided diagnosis of disorders and pathologies. This study proposed, designed and implemented a visualization framework named *SurLens* with high performance computing using Compute Unified Device Architecture (CUDA), augmenting the widely proven ray casting technique in terms of superior qualities of images but with slow speed. Considering the rapid development of technology in the medical community, our framework is implemented on Microsoft .NET environment for easy interoperability with other emerging revolutionary tools. The Visualization System was evaluated with brain datasets from the department of Surgery, University of North Carolina, United States, containing 109 datasets of MRA, T1-FLASH, T2-Weighted, DTI and T1-MPRAGE. Significantly, at a reasonably cheaper cost, *SurLens* Visualization System achieves immediate reconstruction and obvious mappings of the internal features of the human brain, reliable enough for instantaneously locate possible blockages in the brain blood vessels without any prior segmentation of the datasets.



ABSTRAK

Dalam diagnosis perubatan dan perancangan rawatan, pakar radiologi dan pakar bedah bergantung pada hirisan yang dihasilkan oleh pengimbas pengimejan perubatan. Malangnya, kini kebanyakan pengimbas hanya boleh menghasilkan imej dua dimensi. Mesin yang dapat menghasilkan imej tiga dimensi adalah terlalu mahal. Imej dua dimensi yang terhasil ini adalah sukar untuk ditafsir kerana mereka hanya menunjukkan pandangan keratan rentas struktur manusia. Oleh itu, keadaan seperti ini memerlukan doktor pakar untuk menggunakan pengalaman mereka dalam tafsiran lokasi, saiz atau bentuk keabnormalan terutama sekali untuk set data yang besar. Sebelum ini, konsep membina semula imej dua dimensi ke tiga dimensi diperkenalkan. Walau bagaimanapun, model penyusunan semula itu memerlukan pengiraan berprestasi tinggi, sama ada memakan masa atau kos yang tinggi. Tambahan pula, mengesan ciri-ciri dalaman struktur anatomi manusia, seperti pengimejan saluran darah merupakan topik yang masih hangat dalam diagnosis berbantu komputer dan pathologi. Kajian ini mencadangkan, mereka bentuk dan melaksanakan rangka kerja visualisasi dinamakan *SurLens* dengan high performance computing menggunakan Compute Unified Device Architecture (CUDA) menggunakan platform Microsoft.NET. Sistem Visualisasi ini telah divalidasi dengan menggunakan dataset daripada jabatan Pembedahan, Universiti North Carolina, Amerika Syarikat, yang mengandungi 109 dataset dari jenis MRA, T1-FLASH, T2 Berwajaran, DTI dan T1-MPRAGE. Pada kos yang rendah, *SurLens* Sistem Visualisasi mencapai pembinaan semula serta-merta dan pemetaan jelas ciri-ciri dalaman otak manusia dengan kebolehpercayaan yang tinggi untuk menentukan lokasi kemungkinan berlaku sumbatan pada saluran darah otak tanpa perlu disegmentasi terlebih dahulu.



LIST OF PUBLICATIONS

Conferences

1. A.M. Adeshina, R. Hashim, N.E.A. Khalid and Siti Z.Z. Abidin. 2011. Medical Volume Visualization: A Decade of Review, 2nd World Conference on Information Technology, Queen Elizabeth Hotel, Antalya, Turkey: Near East University & Bahcesehir University, 23-27 November 2011.
2. A.M. Adeshina, R. Hashim, N.E.A. Khalid and Siti Z.Z. Abidin. 2011. Infrared-Modified V-Gear Talk-Cam Tracer for Image Processing, 2nd World Conference on Information Technology, Queen Elizabeth Hotel, Antalya, Turkey: Near East University & Bahcesehir University, 23-27 November 2011.
3. A.M. Adeshina, R. Hashim, N.E.A. Khalid and Siti Z.Z. Abidin. 2011. Medical Image Modalities: A Conceptual Review for Volume Visualization, 2nd World Conference on Information Technology, Queen Elizabeth Hotel, Antalya, Turkey: Near East University & Bahcesehir University, 23-27 November 2011.

Journals

4. A.M. Adeshina, R. Hashim, N.E.A. Khalid and Siti Z.Z. Abidin. 2011. Hardware-Accelerated Raycasting: Towards an Effective Brain MRI Visualization. *Journal of Computing* Vol. 3, Issue 10, pages 36-42, October 2011. (ISSN 2151-9617).
5. A.M. Adeshina, R. Hashim, N.E.A. Khalid and Siti Z.Z. Abidin. 2012. Medical Volume Visualization: A Decade of Review. *Global Journal on Technology (Formerly AWERProcedia Information Technology and Computer Science Journal)*. Vol. 1, pages 152-157, May 2012. Academic World Education & Research Centre. AWER Digital Library. (ISSN 2147-5369).

6. A.M. Adeshina, R. Hashim, N.E.A. Khalid and Siti Z.Z. Abidin. 2012. Infrared-Modified V-Gear Talk-Cam Tracer for Image Processing. Global Journal on Technology (Formerly AWERProcedia Information Technology and Computer Science Journal). Vol. 1, pages 175-180, May 2012. Academic World Education & Research Center. AWER Digital Library. (ISSN 2147-5369).
7. A.M. Adeshina, R. Hashim, N.E.A. Khalid and Siti Z.Z. Abidin. 2012. Medical Image Modalities: A Conceptual Review for Volume Visualization. Global Journal on Technology (Formerly AWERProcedia Information Technology and Computer Science Journal). Vol. 1, pages 115-121, May 2012. Academic World Education & Research Center. AWER Digital Library. (ISSN 2147-5369).
8. A.M. Adeshina, R. Hashim, N.E.A. Khalid and S.Z.Z. Abidin. 2012. Locating Abnormalities in Brain Blood Vessels using Parallel Computing Architecture. Interdisciplinary Sciences: Computational Life Sciences. Vol. 4, Issue 3, pages 161 – 172, September 2012. Journal of Systems Biology and Bioinformatics, Springer Publications. (Print - ISSN: 1913-2751, Electronic - ISSN: 1867-1462).
PMID: [23292689]
9. A.M. Adeshina, R. Hashim, N.E.A. Khalid and S.Z.Z. Abidin. 2013. Multimodal 3-D Reconstruction of Human Anatomical Structures using SurLens Visualization System. Journal of Interdisciplinary Sciences: Computational Life Sciences. Vol. 5, Issue 1, pages 23-36, March 2013. Journal of Systems Biology and Bioinformatics, Springer Publications. (Print - ISSN: 1913-2751, Electronic - ISSN: 1867-1462).
PMID: [23605637]



CONTENTS

TITLE	i
DECLARATION	ii
DEDICATION	iii
ACKNOWLEDGMENT	iv
ABSTRACT	v
LIST OF PUBLICATIONS	vii
CONTENTS	ix
LIST OF TABLES	xiv
LIST OF FIGURES	xv
LIST OF SYMBOLS AND ABBREVIATIONS	xix
LIST OF APPENDICES	xxiv

CHAPTER 1 INTRODUCTION

1.1 Background Study	1
1.2 Brain Anatomy and Abnormalities	2
1.3 Motivation	4
1.4 Research Questions	7
1.5 Research Objectives	7
1.6 Scope of the Research	8
1.7 Organization of the Thesis	8

CHAPTER 2 VISUALIZATION OF VOLUMETRIC DATASETS

2.1	Introduction	10
2.2	Volume Visualization	12
2.3	Volumetric Image Datasets	13
2.4	Medical Imaging Modalities	16
2.4.1	Computed Tomography	17
2.4.2	Magnetic Resonance Imaging	18
2.4.3	Clinical Applications / Relevancies	20
2.5	Dataset Pre-Processing Techniques	21
2.5.1	Filtering, Enhancement, Detection& Extraction	21
2.5.2	Volume Segmentation	23
2.5.3	Data Reduction	25
2.6	Medical Volume Visualization	26
2.6.1	Volumetric Image Visualization	27
2.6.2	Multiplanar Reformation (MPR)	28
2.6.3	Surface Rendering Technique	29
2.6.4	Direct Rendering Technique	30
2.6.5	3-D Reconstruction	31
2.7	CUDA Technology	32
2.8	Software Components	36
2.8.1	Graphics Execution	37
2.8.2	Volume Rendering	39
	2.8.2.1 Classification	40
	2.8.2.2 Rendering	42
2.9	Frameworks in Medical Volume Visualization	43
2.9.1	Probe-Volume: An Exploratory Volume Visualization Framework	45
2.9.2	VDVR: Verifiable Visualization of Projection-Based Data	46



2.9.3	GPU Accelerated Generation of Digitally Reconstructed Radiographs for 2-D / 3-D Image Registration	48
2.9.4	Volume Visualization with Grid-Independent Adaptive Monte Carlo Sampling	48
2.9.5	Framework for Volume Segmentation, Visualization using Augmented Reality	49
2.9.6	Illustrative Volume Visualization using GPU-Based Particle Systems	50
2.9.7	Volumetric Ambient Occlusion for Real-Time Rendering and Games	51
2.9.8	An Improved Volume Rendering Algorithm Based on Voxel Segmentation	52
2.9.9	ParaView Visualization Framework	53
2.9.10	VolView Framework	54
2.10	Advantages & Disadvantages of Previous Frameworks	55
2.11	Summary	59

CHAPTER 3 METHODOLOGY: THE DEVELOPMENT OF *SurLens*

3.1	Introduction	62
3.2	SurLens Architecture	63
3.3	The Framework of SurLens	65
3.3.1	Datasets Pre-Processing	68
3.3.1.1	Application of Projective Plane Theorem	68
3.3.1.2	Coordinates Systems	74
3.3.1.3	Intensity Matching	81
3.3.2	Accelerating Hardware	83
3.3.2.1	Data Structuring & Fragmentation	85
3.3.3	Graphic Execution Phase	89

3.3.4	Volume Rendering Phase	90
3.3.4.1	Camera Model	91
3.3.4.2	Volume Classification	93
3.3.4.3	Shading and Gradient Computation	95
3.3.4.4	Interpolation / Re-sampling	97
3.3.4.5	Compositing	98
3.4	Summary	102

CHAPTER 4 IMPLEMENTATION AND TEST DATASETS

4.1	Introduction	104
4.2	Conversion Scheme	108
4.3	Feature & Edge Detection Scheme	112
4.4	Automatic Feature & Mapping Technique	115
4.5	SurLens Robust Algorithms' for Mass data	119
4.6	Summary	120

CHAPTER 5 RESULTS AND DISCUSSION

5.1	Introduction	122
5.2	Results of New Feature & Edge Detection Scheme	123
5.2.1	Experimentation with MRA Datasets	124
5.2.2	Experimentation with DTI Datasets	126
5.2.3	Experimentation with T1-FLASH Datasets	128
5.2.4	Experimentation with T1-MPRAGE Datasets	129
5.3	Results of New Feature Mapping Techniques	130
5.3.1	Experimentation with MRA Datasets	131
5.3.2	Experimentation with DTI Datasets	142
5.3.3	Experimentation with T1-FLASH Datasets	145
5.3.4	Experimentation with T1-MPRAGE Datasets	147
5.4	SurLens Comparison with other Visualization Systems	149

5.4.1	ParaView & VolView Visualization Systems	149
5.5	Results of Robust Algorithms for Mass data	154
5.5.1	Speed Evaluation with MRA Datasets	155
5.5.2	Speed Evaluation with DTI Datasets	159
5.5.3	Speed Evaluation with T1-FLASH Dataset	160
5.5.4	Speed Evaluation with T1-MPRAGE Dataset	161
5.6	Summary	162

CHAPTER 6 CONCLUSION

6.1.	Introduction	164
6.2.	Conclusion	165
6.3.	Contributions	169
6.4.	Future Works	170

REFERENCES	171
-------------------	-----

APPENDIX A	195
-------------------	-----

APPENDIX B	214
-------------------	-----

VITA	
-------------	--



LIST OF TABLES

2.1	Comparison of CPU and GPU	32
3.1	Homogeneous Coordinates Representation of Points and Lines	73
3.2	<i>SurLens</i> Access Design to Intensity	82
4.1	Experimental Testbeds	107
5.1	MRA Speed Evaluation for Datasets of Patient_001 to Patient_009	155
5.2	MRA Speed Evaluation for Datasets of Patient_010 to Patient_015	156
5.3	MRA Speed Evaluation for Datasets of Patient_016 to Patient_020	157
5.4	Randomly Selected Patients MRA Datasets for Speed Evaluation	158
5.5	Speed Evaluation for Selected Patients DTI Datasets	159
5.6	Speed Evaluation for Selected Patients T1-FLASH Datasets	160
5.7	Speed Evaluation for Selected Patients T1-MPRAGE Datasets	161



PTTAUTHM

PERPUSTAKAAN TUNKU TUN AMINAH

LIST OF FIGURES

1.1	Conventional pathologists' slides viewing microscope	5
1.2	Clinical support with visualization	5
2.1	Volumetric Data in Cartesian Grid	15
2.2	Data Structure Grids	16
2.3	An example of a CT slice, a head scan	18
2.4	An example of an MRI slice, brain's Scan	20
2.5	VDVR Pipeline	47
2.6	The GPGPU Paradigm	50
3.1	<i>SurLens</i> Architecture	64
3.2	<i>SurLens</i> Framework Overview	66
3.3	<i>SurLens</i> Framework Phases	67
3.4	Parallel Lines in Projective Plane	69
3.5	Point Estimation of 2-D Slices	70
3.6	Translation	75
3.7	Scaling about the Origin	77
3.8	Rotation about x-axis	77
3.9	Rotation about y-axis	78
3.10	Rotation about z-axis	79
3.11	Algorithm1: <i>SurLens</i> Algorithm for Dataset Pre-Processing	81
3.12	Algorithm 2: <i>SurLens</i> Algorithm for Accelerating Hardware	82
3.13	<i>SurLens</i> -CUDA Architecture	84
3.14	<i>SurLens</i> Data Fragmentation Procedures	87
3.15	<i>SurLens</i> Memory System Architecture	88
3.16	<i>SurLens</i> Graphic Execution Phase (Phase 3)	89
3.17	Algorithm 3: <i>SurLens</i> Algorithm for Graphic Execution	89
3.18	<i>SurLens</i> Volume Rendering Phase (Phase 4)	91
3.19	Image Point Processing Approach	93

3.20	Ambient Lighting	95
3.21	Diffuse Lighting	96
3.22	Specular Lighting	96
3.23	Absorption and Emission along Light Rays	100
3.24	Algorithm 4: <i>SurLens</i> Algorithm for Volume Rendering	102
4.1	<i>SurLens</i> Conversion Scheme	108
4.2	<i>SurLens</i> Conversion Scheme: Design Pipeline for Data Array Structure	109
4.3	<i>SurLens</i> Conversion Scheme: Design Pipeline for ImageData	110
4.4	<i>SurLens</i> Conversion Scheme: Design Pipeline of PointSet	111
4.5	Design of Contour Class for <i>SurLens</i> Feature & Edge Detection Scheme	112
4.6	Design of Anaglyph and Buffering Class for <i>SurLens</i> Feature & Edge Detection Scheme	113
4.7	Overview of <i>SurLens</i> Automatic Feature Mapping Techniques	116
4.8-a	<i>SurLens</i> Automatic Feature Mapping Techniques	116
4.8-b	<i>SurLens</i> Automatic Feature Mapping Techniques	117
5.1-a	Results of <i>SurLens</i> Feature & Edge Detection with MRA Datasets	124
5.1-b	Results of <i>SurLens</i> Feature & Edge Detection with MRA Datasets	125
5.2-a	Results of <i>SurLens</i> Feature & Edge Detection with DTI Datasets	126
5.2-b	Results of <i>SurLens</i> Feature & Edge Detection with DTI Datasets	127
5.3-a	Results of <i>SurLens</i> Feature & Edge Detection with T1-FLASH Datasets	128
5.3-b	Results of <i>SurLens</i> Feature & Edge Detection with T1-FLASH Datasets	129
5.4-a	Results of <i>SurLens</i> Feature & Edge Detection with T1-MPRAGE Datasets	130
5.4-b	Results of <i>SurLens</i> Feature & Edge Detection with T1-MPRAGE Datasets	130
5.5	MRA Evaluation Datasets of Patient_001 and Patient_002	132
5.6.	MRA Evaluation Datasets of Patient_003 and Patient_004	132
5.7.	MRA Evaluation Datasets of Patient_005	133
5.8.	MRA Evaluation Datasets of Patient_006	133
5.9.	MRA Evaluation Datasets of Patient_007 and Patient_009	134



5.38.	Comparison-Patient_099-Stripped_FLASH Datasets	153
5.39.	Divert Comparison-Patient_01_CT_Abdominal Pelvic Datasets	154



PTTA UTHM
PERPUSTAKAAN TUNKU TUN AMINAH

LIST OF SYMBOLS AND ABBREVIATIONS

λ -	Scale Factor
$f : \mathbf{E}^3 \rightarrow \mathfrak{R};$ -	Scalar Function
$f^n : \mathbf{E}^3 \rightarrow \mathfrak{R}^n$ -	n-Dimensional Vector Function
$f^{n^k} : \mathbf{E}^3 \rightarrow \mathfrak{R}^{n^k},$ -	A k-Ranked Tensor Function
π -	Plane
π' -	New plane
(S_{k-1}, S_k) -	Optical Depth
τ -	Extinction Coefficient
$[X, Y, Z]^T$ -	Vector Notation
2-D -	Two Dimensional
3-D -	Three Dimensional
ANN -	Artificial Neural Networks
ANSI -	American National Standard Institute
AOH -	Application Oriented Hypothesis
API -	Application Programming Interface
AR -	Augmented Reality
B -	Strength of the Magnetic (field in tesla)
BCC -	Body Centered Cubic
C# -	C-Sharp
CAD -	Computer-Aided Design
Cg -	C for Graphics
CIL -	Common Intermediate Language
Cos_α -	Product of the Vector of Light Source (a negative value)
CPU -	Central Processing Unit
CRT -	Cathode Ray Tube
CT -	Computed Tomography

CTF -	Contrast Transfer Function
CUDA -	Compute Unified Device Architecture
da -	normal
DDR -	Double Data Rates
DLL -	Dynamic-Link Library
DoD -	Department of Defense
DTI -	Diffusion Tensor Imaging
DVR -	Direct Volume Rendering
$d\Omega$ -	Solid Angle
FCM -	Fuzzy C-means
FDA -	Federal Drug Administration
FDA -	Food and Drug Administration
FEM -	Finite Element Methods
FFTW -	Fastest Fourier Transform in the West
FM -	Frequency Modulation
fMRI -	Functional Magnetic Resonance Imaging
FPGA -	Field Programmable Gate Array
GB -	Gigabyte
GLSL -	Graphic Language Shading Language
GPGPU -	General-Purpose Graphics Processing Unit
GPU -	Graphic Processing Unit
h -	Planck's Constant
H -	Homography Matrix
h -	Planck Constants
HIPAA -	Health Insurance Portability & Accountability Act
HLSL -	High Level Shader Language
HPC -	High Performance Computing
HSV -	Human Visual System
HU -	Hounsfield Units
$h\nu$ -	Energy Carried by Each Photon



PTTA UITHM
PERPUSTAKAAN TUN AMINAH

I -	Intensity
IV -	Integral Videography
k -	Boltzmann constant
KB -	Kilobyte
kNN -	k-Nearest Neighbour Rule
K_s -	Reflection Constant
L_c -	Light Intensity Curve
LM -	Linear Memory
LMIP -	Local Maximum Intensity Projection
LoD -	Level-of-Details
LUT -	Look Up Table
MDFT -	Multidimensional Discrete Fourier Transform
MIP -	Maximum-Intensity Projection
MPR -	Multiplanar Reformation
MPU -	Multi-Level Partition of Unity
MRA -	Magnetic Resonance Angiography
MRF -	Markov Random Field
MRI -	Magnetic Resonance Images
MRT -	Magnetic Resonance Tomography
MS -	Multiple Sclerosis
μ -	Emission Coefficient
n -	Direction
\check{N} -	Number of Photons
NL-means -	Non-Local Means Modes
NMRI -	Nuclear Magnetic Resonance Imaging
nvcc -	Nvidia CUDA Compiler
NVIDIA -	American Global Technology Company, California
O_c -	Colour Curve of Object
OpenGL -	Open Graphic Language
PCA -	Principal Component Analysis



PTTA UTHM
PERPUSTAKAAN TUN AMINAH

PET -	Positron Emission Tomography
Pixel -	Picture Element
PVE -	Partial Volume Estimation
$R(x,n,v)$ -	Radiance
RAM -	Random-Access Memory
R_c -	Resulting Intensity Curve
Regs -	Register
RF -	Radio-Frequency
RGBA -	Red Green Blue Alpha
s -	Distance
SIMD -	Single Instruction Multiple Data
SIMT -	Single Instruction, Multiple Thread
SMs -	Streaming multiprocessor
SPECT -	Single-Photon Emission Computed Tomography
SPs -	Streaming Processors
SVM -	Support Vector Machine
SVR -	Singular Value Decomposition
s_x, s_y, s_z -	Scale Factors along x, y, z axes
T -	Temperature measured in Kelvin
T1-FLASH -	T1-Fast Low Angle Shot Magnetic Resonance
T1-MPRAGE -	T1-Magnetization Prepared Rapid Gradient Echo
TA -	Tensor Approximation
Tcl -	Tool Command Language
T_s -	Transformation Matrix for Scaling
U_i -	Points on Plane
U_i' -	New sets of points on Plane
V -	Vanishing Point
$V(x,y,z)$ -	Discrete regular volume buffer
$V(x,y,z,d)$ -	Three dimensional data
VDVR -	Verifiable Direct Volume Rendering



VTK -	Visualization Toolkit
W_a -	Weight of Ambient
W_d -	Weight of Diffuse
WHO -	World Health Organization
W_s -	Weight of Specular
X, Y, Z -	Coordinates Notation
XML -	Extensible Markup Language
Z -	Ground Plane
α_k -	Opacity
γ -	Gyromagnetic Ratio of the nucleus in rad/T/s
δE -	Radiant Energy interval $d\nu$ around dt
θ_k -	Transparency of the Material
ν -	Frequency through a in
χ -	Absorption Coefficient
$\psi(x,n,\nu)$ -	Photon Number Density



LIST OF APPENDICES

APPENDIX	TITLE	PAGE
A	Appended Evaluation Results	195
B	<i>SurLens</i> ' Full Screen View	214
C	Vita	



PTTA UTHM
PERPUSTAKAAN TUNKU TUN AMINAH

CHAPTER 1

INTRODUCTION

1.1 Background Study

Throughout the history of humankind, visual imagery is seen as an appropriate way to communicate both abstract and concrete ideas to realization. Visualization is a way of making a form of mental vision, image or picture of something that is not visible, present to the sight or an abstraction, visible to the mind (The Oxford English Dictionary, 1989). Visual images are created through visualization, serving as models through which future things emerge. Some of the ancient uses of visualization are the European cave painting, the introduction of geometry by the ancient Greek and the description of locations in form of map.

Visualization spans through a wide spectrum of knowledge domain, however, it can be broadly categorized into scientific and information. Scientific visualization focuses on physical data such as meteorology, human body and earth while information visualization focuses on abstract, non-physical data such as financial data, bibliographic sources and statistical data (Teyseyre & Campo, 2009).

Volume Visualization is a domain within scientific visualization concerned with the representation, manipulation, modeling and rendering of volumetric datasets. Such volumetric datasets are represented as a 3-D discrete regular grid of volume elements (called voxels), stored in a discrete regular volume buffer $V(x,y,z)$ (Kaufman, 1991). Medicine, Engineering, Geology and Pharmacology are among those fields that are massively benefiting from volumetric datasets. With the evolution of modern

technology, volume visualization has been extensively pushed into many applications, especially with arose consequence production of enormous data from medical community. Such creation of great amount of data has created more challenges and difficulties for the extraction of valuable information, analysis and its explanation in an intuitive way. Undoubtedly, CPU, as a functional processing device has high clock speed, facilitating its competencies for general-purpose tasks, but CPU has no parallel processing capabilities (Qin et al., 2012). Consequently, parallel computing is an alternative promising platform to accelerate visualization of medical volumetric datasets.

1.2 Brain Anatomy and Abnormalities

The relevance of brain in human being cannot be over-emphasized. Whereas, brain does not only exist in human being, it exists as well in other mammals. However, human brain is about three times larger, with around one hundred billion neurons (Kasthuri & Lichtman, 2010). Human brain is the center of nervous system controlling all activities of the human body, from self-control, reasoning, planning to vision, with all features greatly pronounced, enlarged and developed. Skull houses many brain slices. To perceive the complexity of the brain, each of the slices that made up the skull exists in certain measured thickness (ranges from 1 - 5mm) with each slice having distances in-between (ranges from 1 - 5mm) relative to image acquisition device employed.

Blood vessels, blood flows and the fluids surrounding the brain may contain different types of abnormalities. *Vascular abnormalities* may occur in the brain whenever abnormalities involve *arteries* or *veins*. In certain cases, there could be *blockages* in one of the blood vessels in the brain, depriving the brain of its functional flow of blood and oxygen. Vascular abnormalities are deadly medical cases that usually lead to *stroke*. Among other life-threatening abnormal conditions in the brain are *brain lesions*, as a result of abnormal tissue area in the brain, *brain tumor*, *hypertension*, *diabetes*, *walderstrom's macroglobulinemia* and *penetrating brain injury*.

Brain tumor is an abnormal growth of tissue in the brain. It may originate within the brain itself (primary tumor) or from other part of the body and travels to the brain

(secondary or metastatic tumor). While there are about two hundred and twenty (220) types of brain tumor classifications, brain tumor ranges from least aggressive, the benigns, which are non-cancerous, to the most aggressive, the malignants that are cancerous. However, most medical institutions use World Health Organization (WHO) standards for their classification. *Glioblastomas* is a malignant tumor that originates from the brain (primary tumor). Patients with *Glioblastomas* live an average of 12 - 14 months, although the medical communities hope for its medical long-term transfer into a more chronic disease for increase in life span of patients to 10-15 years. However, there is unlikely development of such cure within short expected time frame (Bredel, 2009). In the diagnosing procedures of most of these brain abnormalities, medical community has benefited immensely from the image modalities techniques such as MRI and CT.

Magnetic Resonance Images (MRI) sprung up in few decades ago and its significance is clearly noticed specifically in its ability to assign distinguishing intensity values to different levels of tissue densities. MRI is a non-invasive medical diagnostic technique for imaging human interior anatomical structures. MRI machine signal scans points-by-points into the patient's brain anatomical structures, creating a map, which it captures in *binary codes* (1, 0) and stored as 2-D datasets using mathematical function called *Fourier Transform*. MRI technique utilizes strong magnets and pulses of radio waves to manipulate the natural magnetic properties in the human body. Considering the fact that MRI does not use X-ray techniques unlike CT, there are no known biological risks involved when a patient is exposed to MRI scan. Moreover, it produces better images of organs, soft tissues and the interior structure of bones than those of other brain scanning technologies such as Computed Tomography (CT) and Positron Emission Tomography (PET).

The conventional MRI techniques include *axial*, *coronal* or *sagittal* orientation of *T1-weighted*, *T2-weighted* and *T*2-weighted*. However, a number of specialized MR imagery is available for special purposes. *Magnetic Resonance Angiography (MRA)* is primarily designed for imaging *blood vessels* of the brain, to generate images of the *arteries* for *stenosis* (abnormal narrowing), *occlusion* or *aneurysms*. *Diffusion Tensor Imaging (DTI)* is for determination of *magnitude and direction* of water; based on the principle of diffusion, the movement of water molecules from the region of higher



concentration to the region of lower concentration. The *T1-Fast Low Angle Shot Magnetic Resonance (T1-FLASH)* for *glioma tumor* and *lesions*, the *T1-Magnetization Prepared Rapid Gradient Echo (T1-MPRAGE)* which is for detecting *metastatic brain tumors*, are other specialized MR techniques employed in medical diagnosis.

1.3 Motivation

The physical world around us is in three-dimensional (3-D); yet traditional cameras and imaging sensors are only able to acquire and show two-dimensional (2-D) images that lack the depth information (Geng, 2011). The 2-D cross-sectional images produced from imaging techniques such as CT and MRI scanners are generally difficult to analyze. With this, the practice of surgical pathology involving the use of microscope to view tissue mounted on glass slides still persist significantly over many decades. In such usual cases of handling huge information embedded in each pixel of 2-D images, analyzing to deduce the position relationship between focus of infection and three dimensional geometry, estimating size and shape of focus of infection (Wu et al., 2010) usually require mental visualization of medical professionals based on their experience and expertise. This procedure is tedious, time-consuming and prone to error. Figure 1.1 and Figure 1.2 illustrate the conventional pathologist's procedure of analyzing scanned images of patients and the visualization-assisted procedures respectively.

Feature detection and local mapping of internal features are still open topics in the computer-aided diagnosis of biological disorders and pathologies. Although volume rendering has recorded good ability in depicting internal data features, however, locating object boundaries and revealing internal data features of interest are still challenging task due to the usual occlusion of features of interest by other volume structures (Kirmizibayrak et al., 2011; Gabor, Tornai & Cserey, 2010). Implicit visibility of tiny features, through allocation of transparency based on scalar values and assigning of transparency based on localized gradient magnitude for region of interest, are challenging issues. The difficulties of setting proper mapping functions to convert

original volumetric data to renderable color and opacity values limit the application of volume rendering (Guo, Mao & Yuan, 2011).



Figure 1.1: Conventional pathologists' slides viewing microscope (Jeong et al., 2010)



Figure 1.2: Clinical support with visualization

Medical visualization systems are developed to transform large and complex stacks of datasets into effective visual presentations for immediate medical diagnosis and therapy procedures. Volume rendering can be implemented to produce quality images, however, this technique still has a major outstanding drawback of “timely” generation of

such images (Yun & Xing, 2010). These medical scanners usually produce hundreds of 2-D slices requiring intense algorithm optimization. A computer-assisted brain diagnosis system that could effectively serve its purpose must be able to achieve not only fast generation of 3-D model of datasets but also the entire streams of datasets' processing within an interactive speed. The promise of computer-based surgical planning is to provide better surgical results with fewer procedures, decreased time in the operating room, lower risk to the patients (increased precision of technique, decreased infection risk), and lower resulting cost (Kumar & Rakesh, 2011).

Most of the previously developed medical visualization systems have shortcomings in:

1. reconstructing 2-D sequence of human organ, soft tissue and lesions sectional images to 3-D model, (Geng, 2011; Wu et al., 2010).
2. detecting, mapping and isolating abnormalities / tumor for surgery and/or disease diagnosis procedure, (Kirmizibayarak et al., 2011; Gabor, et al., 2010; Guo et al., 2011).
3. handling mass data within a considerable interactive speed, extensive application interoperability and at a low resulting cost (Yun & Xing, 2010; Kumar & Rakesh, 2011).

Thus, the main concentration of this study is to reconstruct sequence of 2-D imagery into 3-D model capable of clearly detecting, mapping and isolating abnormalities / tumor in MR imagery within a considerable interactive speed, extensive application interoperability and at a low resulting cost for optimum use in medical diagnosis and therapy treatment.

1.4 Research Questions

This study aims to solve the following research questions:

1. How to reconstruct 2-D sequence of brain MRI into 3-D model?
2. How to detect, map and isolate brain abnormalities / tumor for surgery and/or disease diagnosis procedures?
3. How to handle mass volume of volumetric brain MRI datasets within a considerable interactive speed, extensive application interoperability and at a low resulting cost?

1.5 Research Objectives

The objectives for this research are as follows:

1. To propose new approaches for brain volume visualization by introducing
 - a framework for reconstructing sequence of 2-D cross-sectional images to 3-D model,
 - a feature and edge detection scheme that can allocate transparency based on scalar values and assign transparency based on localized gradient magnitude for edge detection of region of interest in the data volume,
 - a technique with automatic local feature mapping scheme that can isolate abnormalities / tumor and reveal internal features of brain blood vessels,
 - algorithms within the framework that are robust enough to handle mass volumetric data within a considerable interactive speed, extensive application interoperability and at a lower resulting cost.
2. To design and implement a visualization system (*SurLens*) based on the proposed approaches.
3. To compare the volume visualization results of *SurLens* with existing approaches.



1.6 Scope of the Research

This research focuses on the design of *SurLens* framework and the implementation of *SurLens* volume visualization system. The study is limited to reconstructing and locating abnormalities / tumor in Magnetic Resonance (MR) Imagery of the brain in 3-D model. Focus is on obtaining quality 3-D images, sufficient enough to reveal detail internal information of datasets using the MRI datasets from the department of Surgery, University of North Carolina, Chapel Hill, United States. The study concentrates on Magnetic Resonance Angiography (MRA) datasets, however, Diffusion Tensor Imaging (DTI), T1-Fast Low Angle Shot Magnetic Resonance (T1-FLASH) and T1-Magnetization Prepared Rapid Gradient Echo (T1-MPRAGE) would also be considered. The development of the proposed visualization system would be within C# programming language environment, built on top of visualization toolkit (VTK) libraries and on parallel computing platform, Compute Unified Device Architecture (CUDA).

1.7 Organization of the thesis

To disseminate the findings of this research, concise investigation is presented into the field of visualization, human brain anatomy and its associated abnormalities.

In order to properly draw attention of the readers to some of the fundamentals of this research, Chapter 2 commences with general introduction of volume visualization and volumetric image datasets. Datasets pre-processing techniques and medical volume visualization as a whole is reviewed. Parallel processing procedures, specifically CUDA technology and previously proposed software components in this domain are extensively reviewed and presented. Strengths and weaknesses of previously proposed frameworks, schemes, algorithms and techniques are presented. The chapter describes and compares ten (10) recently proposed volume visualization frameworks in their entirety, in justification of the newly proposed framework, schemes, algorithms and techniques in this study.

Chapter 3 discusses the procedural research methodology, numerical computations and data structures used in the development of *SurLens* visualization system. The framework, schemes, algorithms, techniques and data collection for the development of *SurLens* visualization system are presented. Chapter 4 focuses on the designs and implementation of the proposed *SurLens* Visualization system for volumetric brain MRI datasets.

Evaluation results and discussion in comparison with two (2) notable, previously developed visualization systems are presented in Chapter 5 while Chapter 6 concludes the research, summarizes the major contributions of the study and presents the future works.



PTTA UTHM
PERPUSTAKAAN TUNKU TUN AMINAH

CHAPTER 2

VISUALIZATION OF VOLUMETRIC DATASETS

2.1 Introduction

Visualization is a phenomenon existing in our day-to-day life. Over a thousand years ago, visualization has been used in the data plots, maps and scientific drawings. As far back as 1137 A.D, visualization was used to draw the map of China and the very famous map of Napoleon's invasion of Russia in 1812 (Owen, 1993). Visualization could be defined as a tool or method for interpreting image data, fed into a computer and for generating images from complex multi-dimensional data sets (McCormick, DeFanti & Brown, 1987). Informally, visualization engages the human vision and the processing power of human mind in the transformation of data or information into visual images referred to as pictures.

Visualization has received many descriptive terminologies over the years. Scientific visualization was first fundamentally used in 1987 (Rosenblum et al., 1994) and its seen as a representation of numerical data in a way that extrapolates meaningful information to understand or analyze interesting feature the data might hold. Data visualization is a more general term that implies treatment of data sources beyond the sciences and engineering. This encompasses marketing, business and financial data. More often, the term information visualization is becoming more pronounced. It is used to describe visualization of abstract information such as hypertext documents on the



World Wide Web, directory or file structures on a computer or abstract data structures (InfoVis, 1995).

Visualization is also seen as a method of extracting meaningful information from complex dataset through the use of interactive graphics and imaging (Kaufman, Cohen & Yagel, 1993), hence, computer graphics and image processing (or imaging) are tools for visualization. Computer graphics is the creation of images using computer, which encompasses 2-D paint techniques, drawing or rendering techniques. The output of computer graphics is an image. With image processing, we can define techniques to transform (rotate, scale, shear), extract, analyze and enhance images. Visualization focuses on exploring, transforming, viewing data as image in order to gain understanding and insight into the data. Computer graphics is used as a tool to produce the output for visualization. However, this study specifically tread the path of volume visualization, which is typically identified with the rendering, modeling, manipulation and representation of datasets (Kaufman, 1991; Kaufman, 1996).

Volume visualization is an important diagnostic tool in modern medicine. With computer imaging techniques such as Computed Tomography (CT) and Magnetic Resonance Imaging (MRI), internal information of a living patient is captured. The information is captured in form of slice-planes or cross-sectional images of patient which could be compared to the conventional photographic X-ray. A slice consists of a series of number values representing the attenuation of X-rays (in case of CT) or the relaxation of nuclear spin magnetization (MRI) (Krestel, 1990). However, with applied and sophisticated mathematical techniques, the slice-planes could be reconstructed and gathered into a volume of data.

Generally, with the slices, the series of number values are arranged in either a matrix pattern or regular array. However, with huge amount of information data in the slice, it is not possible to understand the data in its raw form, even with a trained eye. This is where the gray scale value comes in. Computer only understands 0's and 1's, whereas, human being cannot firmly relates the codes to meaningful information, possible solution is to represent the number values in 2-D cross-section that could be more useful with human vision system. Hence, such representation requires understanding the way medical imaging device scans.

This chapter reviews visualization of volumetric datasets and presents earlier frameworks from which medical volume visualization can be facilitated. After outlining broad collection of volumetric data acquisition methodologies, varying volume rendering techniques are described. The chapter extrapolates image reconstruction approaches, direct volume visualization techniques; possible optimization procedures specifically parallel processing approaches and their outstanding issues, which crystalize the research direction for this study. The weaknesses and strengths of each of the previous techniques are discussed. The strengths and weaknesses of ten (10) recently proposed volume visualization frameworks in entirety, including their proposed schemes, algorithms and techniques, are presented and compared in justification of the proposed framework, schemes, algorithms and technique in this study.

2.2 Volume Visualization

Volume visualization is a sub-field of scientific visualization that extracts meaningful information from volumetric data using interactive graphics and imaging, and it is concerned with volume data representation, modeling, manipulation, and rendering (Suter et al., 2011). Volume visualization is an important tool for visualizing and analyzing data sets with its extensive application into such areas as biomedicine, computational fluid, finite element models, computational chemistry and geophysics. Magnetic Resonance (MR) Imagery and Computed Tomography (CT) are both imaging techniques benefiting optimally from volume visualization. Such numerical simulations and sampling devices create images of the human body for clinical diagnosis while volume visualization presents such datasets for viewing and clinical analyzing of the anatomical structures. Over recent years, volume visualization is continually evolving as visualization approaches, especially with the advent of faster processing devices. One of the challenges depriving the usage of volume visualization is the memory system to support volume processing (Suter et al., 2011; Ma, Murphy & O'Mathura, 2012).

Two-dimensional (2-D) data is represented as X and Y axes. These are mere flat structures in horizontal and vertical axes. Any image we have in this form, if turned to

its other sides, will become a line. Hence, a 2-D structure has corners or vertices and sides in two planes and cannot provide detail information embedded in image data. However, the 2-D representation can be re-represented in three-dimension (3-D), using mathematical models which has X and Y planes (just as 2-D image) but a Z-axis inclusive, this gives the image more features such as rotation. This third axis added faces to the 3-D structures, making the data available for real world simulation of the imaged object.

Since, there is a one-to-one correspondence between the pixel value in the image scan and a specific tissue of a patient, the numbers could be assigned a specific gray scale value. Displaying the data on the computer screen at this stage will emerge the structures in the patient's data. The emerged structures are as a result of the interaction of the human visual system with data spatial organization and the chosen gray-scale values. With this approach, its being possible to translate what computer represents as series of numbers into the corresponding cross-section of human body; the skin, the bone and the tissue. A more useful result could be made available for diagnosis by extending the 2-D into 3-D technique. In this case, the image slices are gathered as a volume of data. With 3-D technique, we can reveal the entire anatomical structure of a living patient without the intervention of surgery.

With the inability of the medical image scanners to present human anatomical structures in 3-D format, reconstruction procedure is the alternative. Reconstruction is a reverse engineering technique of 2-D MR imagery to 3-D. This is achieved in the medical diagnosis and disease management using the combination of computer graphics and image processing tools, the resulting 3-D data could serve as information for *opinion making and intervention planning* on a living patient without any prior mandatory surgical operation.

2.3 Volumetric Image Datasets

The first step towards volume visualization is the acquisition of volumetric data. Typical set of data samples is represented as $V(x,y,z,d)$, in the case of a three- dimensional data,

with d representing the data property at a location determined by x, y, z . To describe value at any d continuous location, *zero-order (the Nearest Neighbor)*, *first-order (trilinear also called piecewise function)* and *higher-order interpolation* are possible options. The region of constant value that surrounds each sample in *zero-order interpolation* is known as a *volume cell* (commonly interchangeably referred to as *voxel (volume element)* or *grid location* or *sample points*) with each voxel being a rectangular cuboid having six faces, twelve edges and eight corners (Kaufman, 1996). Dataset is a collection of volume elements. However, there is variation in the spatial and intensity resolution of images produce by different medical imaging devices. This section presents some of the commonly used tools for volume data acquisition.

It is important to discuss the *topology* or *geometry* in which *volumetric data* must be. Data samples may exist as *scalar data*, holding such values as temperature, pressure and density, or exist as *vector* (e.g. velocity) or *tensor* (e.g. Finite Element Methods (FEM) modeling). Typically, a volume dataset V is a set of element (Winter, 2002) defined as:

$$V = \{ \nu_i(x, y, z) \mid i = 1, 2, \dots, n \}$$

(x, y, z) is a point in 3-D space, \mathbb{E}^3

$\nu_i(x, y, z)$ could be scalar, vector or tensor, which is defined as follows:

- a scalar function $f : \mathbb{E}^3 \rightarrow \mathfrak{R}$;
- an n-dimensional vector function, $f^n : \mathbb{E}^3 \rightarrow \mathfrak{R}^n$, or
- a k-ranked tensor function, $f^{n^k} : \mathbb{E}^3 \rightarrow \mathfrak{R}^{n^k}$,

Scalar and *vector* functions are representation of special cases of tensor functions with ranks 0 and 1 respectively. Usually in volume visualization, a total function is given by a physical or simulated object and then sampled at discrete points which is stored as discrete set of elements resulting in the formation of the defined dataset V . Figure 2.1 is the representation of volumetric data in Cartesian grid.

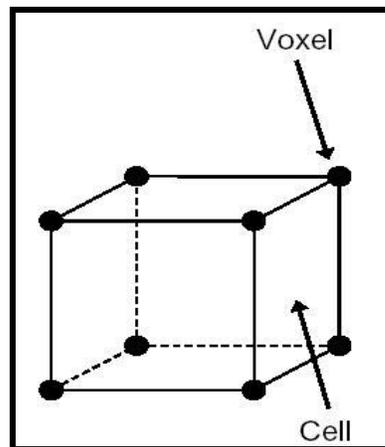


Figure 2.1: Volumetric Data in Cartesian Grid

Speray & Kennon (1990) categorized volume *dataset V* into *structured* and *unstructured* based on the topology of the *dataset*. In line with such categorization, the topology of structured data is well defined in each of its three orthogonal planes. This category includes cartesian, rectilinear and curvilinear grids. Unstructured grids are complex and difficult to use because their structures are not implicitly defined by data arrangements. However, *rectilinear grids* can be defined in *computational space* and classified as being *regular* or *irregular* in structure. If the spacing between samples along each axis is constant along the three orthogonal axes (x, y, z), which is mostly the case, the *dataset V* is called *isotropic*. In certain cases, there might be separation along each axis in the dataset sample but different between the axes, the *dataset V* is referred to as *anisotropic*. Hence, if V is defined on a *regular grid*, a *3-D array* (commonly referred to as *volume buffer*, *3-D raster* or *cubic frame buffer*) is used to store the values and V is referred to as *array of values $V(x, y, z)$* defined only at grid locations. Figure 2.2 shows the different data structure grids.

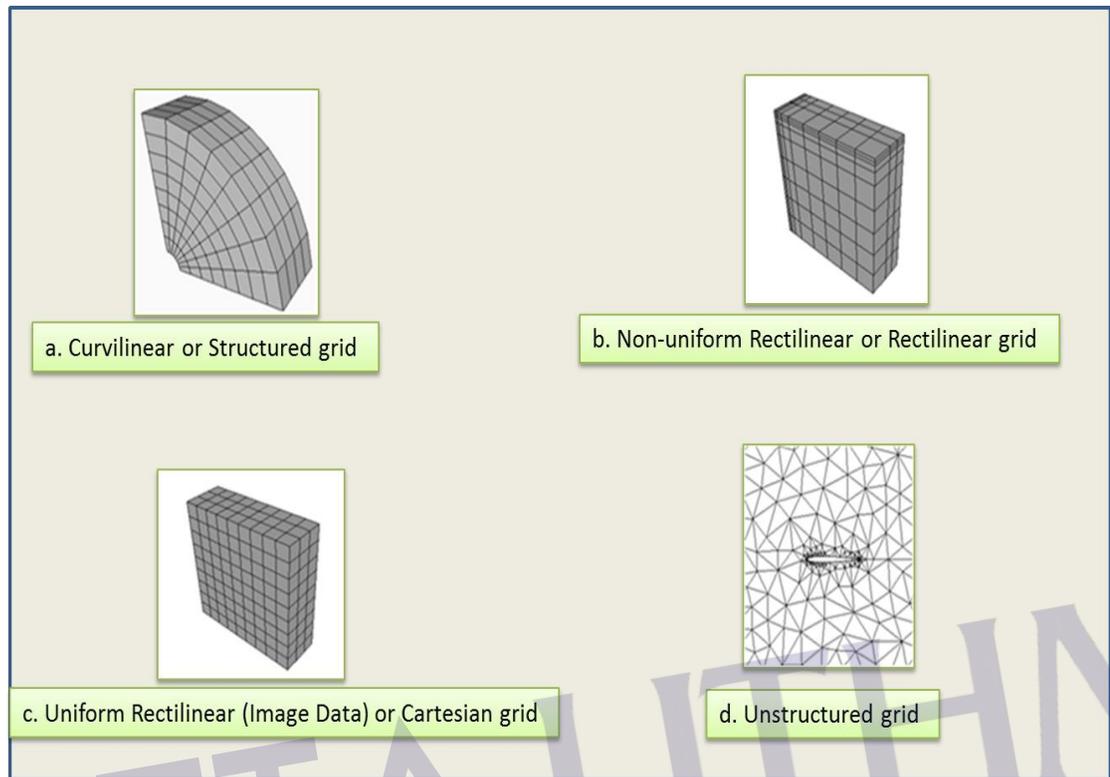


Figure 2.2: Data Structure Grids

2.4 Medical Imaging Modalities

Volume visualization became feasible with the revolution in image acquisition for extensive medical diagnosis and pre-treatment planning. The medical science that uses electromagnetic radiation, ultrasonography or radioactivity for evaluation of body tissues in case of injury or disease is referred to as *diagnoses medical imaging*. However, electromagnetic radiation can either be ionizing or non-ionizing. This section gives a brief overview and concepts of some of medical imaging modalities.

X-ray is the oldest imaging technique widely used throughout the world. It is an ionizing radiation technique discovered by the German physicist in 1895 by Wilhelm Conrad Röntgen (Yang, Guang-Zhong & Firmin, 2000). The discovery of Röntgen in that century drives the use of electromagnetic radiation in the form of ionizing radiation (gamma and X-rays) in an unprecedented speed for diagnostic radiology. The basic

principle for using X-ray involves passing of beam of X-rays, produced by an X-ray tube to selected parts of the body. There was an attempt to reconstruct images from projections as at 1940, this was even planned before the advent of modern computer technology. Gabriel Frank achieved this with the plan of describing the basic idea of modern tomography including such concepts as sonograms and optical back projection (Hsieh, 2002). About 16 years later, Allah M. Cormack furthered the research objectives with some experimental works based on reconstructive tomography.

$$\text{CT Number} = (\mu - \mu_{\text{water}} / \mu_{\text{water}} - \mu_{\text{air}}) \times 1000 \quad (1)$$

In 1967, the first CT scanner was developed by Godfrey N. Hounsfield in England at the Central Research Laboratory of EMI, Ltd (Hounsfield, 1973). Hounsfield investigation on pattern recognition techniques shows that if X-ray is passed through a body from different directions, this would result in its' internal body reconstruction. In his trials in 1969, test objects were scanned with isotope source that required a scan time of 9 days per image (Kalender, 2006).

Research usage of any of the image modalities depends on the intended image area to extract. Some could successfully extract certain information called “*Morphological Information*” while others are very useful in extracting “*physiological or functional information*”. *X-ray*, *CT* and *MRI* are typical examples of former while *PET* and *SPECT* are examples of the later. However, such specific features and functionalities justify their usage in medical community. Section 2.4.3 explains specific clinical relevancies of these image modalities.

2.4.1 Computed Tomography

Computed tomography (CT) is a widely adopted imaging modality with many clinical applications from *diagnosis* to *procedure planning* (Merck, 2009). Computed Tomography is a technique of X-ray photography in which a single plane of a patient is scanned from various angles in order to provide a cross-sectional image of the internal structure of that plane (Hsieh, 2002). Conventional radiography uses the relative

distribution of X-ray intensities for its measurement. It involves sending of uniform intensity X-ray through a patient from an X-ray source of intensity I_0 and corresponding exiting of the X-ray with intensity $I(x, y)$ from the other side, which then interact with a radiography film sheet. The different paths through the material will alternate the X-rays by varying amounts, based only on the mass attenuation coefficient (μ), since the *distance* (d) is the same on all point of the radiography film (Shabaneh et al., 2004). CT uses attenuation as the judgments of its measurements as the X-ray is scanned through the patients.

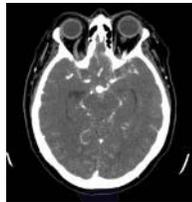


Figure 2.3: An example of a CT slice, a head scan (Lundström, 2007)

The patient is scanned using an X-ray source from one side of the plane and the detector placed on the opposite side is used to measure the attenuated X-ray, which is recorded by computer. After the first scan through the plane, the X-ray source and the detector rotate with a particular predefined amount for another translational scan. Hence, an X-ray technique involves passing electromagnetic radiation through the body. This is usually presented as CT Number, expressed in “*Hounsfield Units*” or “HU” named after Godfrey Hounsfield. A *positive CT* indicates a tissue is more attenuating than water while a *negative CT* denotes a tissue with lower density than water.

2.4.2 Magnetic Resonance Imaging

Magnetic Resonance Imaging (MRI) technique has been one of the primary tools employed in medical diagnosis since the first publication of human body image in 1977 (Damadian, Goldsmith & Minkoff, 1977). MRI imaging technique is completely

different from that of Computed Tomography as it uses energy sources as its imaging procedure rather than ionizing radiation technique of X-ray. In the early years of existence of MRI, it was referred to as *Nuclear Magnetic Resonance Imaging (NMRI)* since it was developed from knowledge gained in the study of nuclear magnetic resonance (Amruta, Gole & Karunakar, 2010). The term NMRI is sometimes still in use when discussing non-medical devices of the same NMRI principle. However, in medical imaging, magnetic resonance tomography (MRT) may sometimes be interchangeably used for MRI. The procedure requires the usage of a strong magnetic field for spin alignment of hydrogen nuclei (photons) in the body.

The spin synchronizes as the radio-frequency (RF) pulse matches the nuclear resonance frequency of the photons. As the pulse is removed, different relaxation times are measured, that is, the times for the spins to go out-of-sync (Lundström, 2007). The density and chemical surroundings of the hydrogen atoms determine the measured value. Whilst some vectors will form alignment towards the direction of the main magnetic field, a slight majority will align themselves in the slightly lower energy state associated with the direction of the main magnetic field (Geoffrey et al., 2008). MRI creates its images as a result of the difference between two populations of vectors leading to the equilibrium net magnetic vectors. We could therefore say that, with MRI, a body is prepared for radio signal transmission on the FM bandwidth. The relative distribution of the vectors aligned within or against the main magnetic field is described by *Boltzmann distribution* as in equation (2).

The value of k is the *Boltzmann constant*, T is the *temperature* measured in *kelvin*, h is the *Planck constants*, γ is the *gyromagnetic ratio* of the nucleus in *rad/T/s* and B is the *strength of the magnetic field in tesla*, \hbar is a constant approximately equal to 3.14159. The number of spins in the lower energy level and the number of spins in the upper energy level are denoted by n_{\uparrow} and n_{\downarrow} respectively.

$$n_{\uparrow} / n_{\downarrow} = \exp(-\Delta E / kT), \text{ with } \Delta E = \hbar\gamma B / 2 \hbar \quad (2)$$

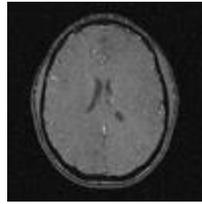


Figure 2.4: An example of an MRI slice, brain's Scan

2.4.3 Clinical Applications / Relevancies

MRI is the only chemically sensitive in-vivo imaging technique with high-resolution soft tissue contrast that allows physicians to peer deep inside the human body, producing clinically relevant images of soft tissue lesions and functional parameters of the body organs, without the use of invasive procedures or ionizing radiation such as X-rays (Cosmus & Parizh, 2011). However, with the knowledge gained in the course of this study, some of the clinical applications of CT and MRI, as being proven by researchers, solemnly depend on the required medical examination on the patient and in certain cases, the image modalities are seen to be complementary to each other in the diagnosis procedures.

With CT scan, herniated disc, spinal stenosis, fractures in the spine can be detected. It has also proven very useful in cartilage invasion and anatomy of the surrounding tissues. MRI has ability to demonstrate and characterize soft tissues hence useful in heart, muscles, brain, spinal cord, some head and neck tumors. Consequently, CT and MRI are mostly used image modality. In order to benefit optimally from CT and MRI, their combinatory techniques were introduced to create more impact features in medical imaging such as PET / CT and PET / MRI. Meanwhile, Magnetic Resonance Imaging (MRI) is the most recently applied technique, most commonly used in radiology to visualize the structure and functions of the body for many reasons among which is, it provides detailed images of the body in any plane with higher discrimination (Sun, Bhanu & Bhanu, 2009).

2.5 Dataset Pre-Processing Techniques

Pre-processing stage in volume visualization is to enhance the visual appearances of the images and the manipulation of the datasets' structures, to convert them from their acquired representation to spatial representation required and appropriate for visualization. However, a lot of caution needs to be exercised with image enhancements' procedures as poorly embarked approach may introduce image artefacts or even lead to loss of information in the datasets.

Segmentation, a key step and a large research area in visualization, is usually performed at the pre-processing step of volume visualization. As a matter of fact, different organs or tissues of an acquired volumetric data might have the same density or intensity hence segmentation stage and not only classification becomes essential. The fundamental principle guiding volume visualization is based on the fact that empowering the user to see a certain structure, using only classification is not always possible (Meißner et al., 2000). Though acquisition methods usually demand different level or extent of required segmentation but most methods require semi-automatic approach which invariably increases the overall processing time of datasets in volume visualization. Studies have shown that segmentation of brain MR is a compulsory, difficult and time consuming stage for volume visualization because of variable imaging parameters, overlapping intensities, noise, partial voluming, gradients, motion, echoes, blurred edges, normal anatomical variations and susceptibility artefacts (Lladó et al., 2012; Sha & Sutton, 2001).

This section reviews previous datasets pre-processing techniques and highlights the significant contribution of *SurLens* Dataset Pre-processing approach.

2.5.1 Filtering, Enhancement, Detection & Extraction

One of the key processes in the pre-processing is the removal of noise from MRI data. Some of the techniques used for MRI de-noising include non-linear filtering methods (Muhamed et al., 2011; Gupta, Anand & Tyagi, 2012) spectral subtraction (Liu et al.,

2012), wavelet-based thresholding (Agrawal & Sahu, 2012), anisotropic non-linear diffusion filtering (Zhang & Ma, 2010; Perona & Malik, 1990), Markov Random Field (MRF) models (An & An, 1984), wavelet models (Nowak, 1990), non-local means modes (NL-means) (Buades, Coll & Morel, 2005), and analytical correction schemes (Sijbers, 1998). Despite the fact that there are quite a substantial number of state-of-the-art methods for de-noising, accurate removal of noise from MRI is still a challenge; as all these methods are almost the same in terms of computation cost, de-noising, quality of de-noising and boundary preserving, which has retained MRI de-noising as an open issue that needs better improved methods (Bandhyopadhyay & Paul, 2012). Hence, de-noising methods at this current state of research are not reliable enough to fully support pre-processing stage of volume visualization. The main challenge in de-noising MRI is to preserve the edges and the details, at the same time to reduce noise in uniform regions (Diaz et al., 2011).

Edge detection or extraction is an important step in MRI data pre-processing. There are three steps in edge detection process (Senthilkumarn & Rajesh, 2008), the image filtering, the image enhancement, and the image detection. Image filtering is required in pre-processing because the target MRI images might have been corrupted through a number of circumstances like impulse noise, Gaussian noise, being common situations. More filtering procedures to reduce noise may results in loss of the strength of the edges (Senthilkumarn & Rajesh, 2009). Image enhancement emphasizes pixels where there is a significance change in local intensity values and is usually performed by computing gradient magnitude (Wen, Zhang & Jiang, 2008) while image detection usually based on thresholding criterion (Paulinas & Usinskas, 2007).

Quite a number of operators are usually used for image filtering, enhancement, and detection such as Sobel, Prewitt, Roberts, Laplacian of Gaussian, Zero-cross and cunny (usually refers to as Gaussian) operators. Among these set of techniques, Sobel operators' produces best sharpness and clear edges (Ponraj et al., 2011). Though Sobel operator has been proven to produce superior qualities compared to other techniques, it has also been confirmed inaccurate and sensitive to noise (VenuGopal & Naik, 2011). Therefore, since image filtering, enhancement, detection and extraction technique play a key role in the development of a reliable medical visualization framework, a better and a

contributing approach must be considered during the pre-processing stage of a volume visualization framework in order to improve accuracy and noise sensitivity interference.

As one of our contributions to this field, we have therefore designed and implemented a new algorithm for image filtering, enhancements, detection and extraction, actualized at the graphic execution phase of our framework, which is the main entry point of datasets into volume visualization. This is an improved and better approach tackling accuracies of image filtering, enhancements, detection and extraction by enabling the datasets to be processed at the main entry point of volume visualization in order to avoid any unwanted noise sensitivity. We do not observe any shortcoming of this design hence it is noted as an improvement over all the previously pre-processing approaches.

2.5.2 Volume Segmentation

Brain MRI segmentation has been attracting attention for a while considering its significance in the medical image analysis and diagnosis. As each of the points in the image scan corresponds to a particular point in the human body structure, during segmentation process, each point in the scanned image and its correspondence to the tissue or organ is identified. A number of segmentation algorithms have been proposed in the past. Clustering-based (Kannan & Pandiyarajan, 2009), region-growing (Welinski & Fabijanska, 2011; Deng et al., 2010), active contour-based (Tanoori et al., 2011), watershed-based (Freitas et al., 2011) and morphological-based segmentation (Li et al., 2011) have been previously applied to brain MRI volume segmentation. Sethian (1999), Ben-Zadok, Riklin-Raviv & Kiryati (2009) and Cremers et al., (2007) have made appreciable contribution in the boundary-based segmentation procedures.

One of the notable studies in this regard is that of Bezdek, Hall & Clarke (1993). Bezdek et al. (1993) made a thorough review on MRI segmentation using pattern recognition techniques. The study categorized brain MRI segmentation algorithms into supervised methods and unsupervised strategies. Supervised segmentation strategy is based on some prior information or knowledge to perform segmentation while

unsupervised strategy performs brain MRI segmentation with no prior knowledge or information. The supervised methods are listed to include Bayes classifiers with labeled maximum likelihood estimators, the k-nearest neighbour rule (kNN) and artificial neural networks (ANN) while the unsupervised methods include Bayes classifiers with unlabelled maximum likelihood estimators or the fuzzy C-means (FCM) algorithms.

Though segmentation is usually performed at the pre-processing stage of volume visualization, being a key and a large research area, some studies separated the usual pre-processing stages distinctly from segmentation. Clarke et al. (1995) reviewed both pre-processing and segmentation methods of soft brain tissue. In the same vein, Styner et al. (2008) reviewed semi-automated and automated multiple sclerosis (MS) lesion segmentation approaches, analyzing MS lesions, pre-processing steps and segmentation approaches. More recently, Lladó et al. (2012) presented a review of brain MRI with the goal of helping diagnosis and follow-up of multiple sclerosis lesions in brain MRI.

In order to enhance the visual appearance of the brain MRI images, any possible artefacts will need to be removed. Removal of the contained artefacts could be done at this stage, done partly or delayed until the final entry point of the dataset into volume visualization phase, this depends on the design of the volume visualization framework. Whichever of the approach being adopted in the framework design, there must be adequate provision set aside in case of unexpected introduction of certain level of artefacts during the pre-processing phase.

Skull stripping is another important pre-processing step since fat, skull, skin and other non-brain tissues may cause mis-classifications in some approaches due to the intensity similarities with brain structures (Detta & Narayana, 2011). Some of the components of the brain require a particular MRI technique for their diagnosis, hence, without thorough skull stripping it might be difficult to have the intended structures' of study visible with volume visualization algorithms.

In cases where studies need to be carried out on more than one components structure of the brain e.g. tissue and fat, alignment of the soft brain images would be required. Aligning all the images from different modalities or MR images is known as registration (Zitova & Flusser, 2003). The precise steps involve include feature



PTTA
PERPUSTAKAAN TUN AMINAH

matching, transform model estimation and image resampling with transformation (Vivekanandan & Raj, 2011).

This study has contributed to the pre-processing stage of volume visualization in terms of segmentation of datasets. Though manual segmentation of brain tissues is reliable than automatic approach but certainly difficult, time-consuming and it is highly dependent on large intra and inter-observer variability that leads to degradation of credibility in the segmentation analysis (Kasiri et al., 2010). Hence a reliable computer-aided brain analysis development using segmentation as a prerequisite usually pay the cost of inaccuracy and untimeliness which are important requirements needed to support medical diagnosis. Within the context of volume visualization, it is an established fact that quantitative and qualitative studies of anatomical brain tissues and structures that have distinctive structural or functional properties usually relies on accurate segmentation of brain (Pham, Xu & Prince, 2000; Sonka & Fitzpatrick, 2000). However, this thesis developed a computer-aided diagnosis system that do not use segmentation as a pre-requisite and yet able to obtain a reliable, accurate and robust volume visualization of brain MRI.

2.5.3 Data Reduction

Data reduction is of great importance to save storage space at all the stages of the processing and rendering pipelines, as well as to reduce time and cost of transmission between layers of the memory hierarchy (Suter et al., 2011). Data reduction are usually carried out using compact representations with pre-defined and data-specific bases, amounting into the two (2) broadly categorized approaches for data representations. Pre-defined bases include Fourier transform (Chiueh, 1997) and wavelet transform (Yeo & Liu, 1995). Methods using pre-defined bases are often computational cheaper than data-specific bases approaches. Such additional computational cost leveled on data-specific bases is due to the requirement of more pre-computing time required to compute bases. However, methods using data-specific bases approach have more potential in removing more redundancy from datasets when closely compared with



pre-defined basis methods. Singular value decomposition, SVD, (Kumar, Nasser & Sarker, 2011), Principal Component Analysis, PCA, (Chen & Yang, 2011), Tensor Approximation, TA, (Suter, Zollikofer & Pajarola, 2010; Tsai & Shih, 2006; Wu et al., 2008) are the commonly used methods of data-specific bases approach.

This study has contributed to the data reduction in volume visualization by embedding image data functions in the *SurLens* data pre-processing algorithm, which make use of data array objects for its data representation. Our data array utilized contiguous data arrays as the basis of the structures. With contiguous data array, we record better data memory allocation and size since contiguous arrays can be created, deleted and likewise transversed than linked lists or array pointers in data structures. Hence, in terms of redundancy removal and computation cost, our data reduction approach is better compared to both the pre-defined bases and data-specific bases approaches.

2.6 Medical Volume Visualization

Volume visualization is a sub-field of scientific visualization that extracts meaningful information from volumetric data using *interactive graphics* and *imaging*. It is concerned with volume data representation, modeling, manipulation, and rendering (Kaufman, 1991; Kaufman, 1996; Chen, Kaufman & Yegel, 2000; Mueller, Chen & Kaufman, 2001; Kaufman & Mueller, 2005). Volume data are 3-D (possibly time-varying) entities that may have information inside them, might not consist of tangible surfaces and edges, or might be too voluminous to be represented geometrically (Kaufman & Mueller, 2005). Hospitals usually rely on radiology departments as their main source of data when it comes to the decision on the treatment of most diseases, hence clinical discipline of radiology is seen to be providing the greater percentage of all diagnoses for patients' treatments.

Volumetric data can be obtained using sampling, simulation, or modeling techniques. 2-D slice is obtained through different image modalities such as Magnetic Resonance Imaging (MRI), Computed Tomography (CT), functional MRI (fMRI), or

Positron Emission Tomography (PET). In the medical sciences, this is visualized for either diagnostic purpose, planning of treatment or surgery exercises.

2.6.1 Volumetric Image Visualization

Visualization of 3-and 4-dimensional volumetric medical data has been increasingly attracting attention recently because of their huge applications in many areas of life. This section reviews some of the related works in this domain.

Inaccuracy in volume of pathology zone in 3-D images is usually due to lack of precision in the images, this is as a result of manual extraction of the images. Krechetova, Glaz & Platkajis (2008) proposed an approach for volume estimation and automatic region calculation. With this procedure, segmentation is done automatically and loss of information in the data is prevented.

Integral Videography (IV) (Herlambang et al., 2008) and stereoscopic image (Ferre et al., 2007) are both significance stages in 3-D visualization. Herlambang et al. (2008) present a real-time autostereoscopic visualization system using the principle of Integral Videography (IV). The study developed maximum-intensity projection (MIP) and composite volume ray casting method for IV volume rendering algorithms that enables real-time visualization of 3-D and 4-D medical images. The system was used to visualize 4-D MR image that was generated from registration of 3-D MR image and 4-D ultrasound images. One of the greatest difficulties in the use of stereo vision systems is image disparity, which affects the correct image fusion process of the brain (Ferre et al., 2007). The experiments of Ferre et al. (2007) obtained the limits for a proper image fusion and assessed the use of stereoscopic image in executing tele-operated robot guidance tasks. The research records a guaranteed depth perception for the performance of stereoscopic image.

Graphic Processing Units is of great importance to the speed of volume rendering algorithm. Researchers applied this to medical volume visualization (Manke & Wönsche, 2008 ; Zhang, Eagleson & Peters, 2007). Manke & Wönsche (2008) framework for interactive exploration of complex data was GPU-based. There-in the

framework, a controller object is responsible for the management of the program execution from initialization, rendering to termination stage. CT data set of the visible male and a PET data set of a monkey were used for the experiments. Zhang et al. (2007) segment-based, post colour-attenuated classification algorithm was implemented and tested with GPU. This algorithm was developed to solve the existing difficulties in achieving an optimized balance between real-time artefact-free medical image volume rendering and interactive data classification. The system was tested and confirmed faster than the non-accelerated pre-integrated classification approach in about 100 times when implemented within the GPU-based volume ray casting system (Zhang et al., 2007).

In an attempt to reduce datasets processing time, registration and segmentation methods, being integral part of medical volume visualization, was merged into surface model proposed by Aliroteh & McInerney (2007). The research combines a general deformable subdivision surface model with a novel sketch-line user initialization process. The efficiency of the proposed method was demonstrated in segmentation of objects from several 3-D data sets to ascertain its accuracy.

Medical image visualization has recorded three basic principles of rendering algorithm. These are the multiplanar reformation (MPR), the surface rendering technique and the direct rendering technique (Zhang, Eagleson & Peters, 2007). Meanwhile, we can likewise say we have direct rendering and indirect rendering. While direct rendering includes direct surface rendering and direct volume rendering, surface rendering is usually refers to as indirect rendering. In reality of it all, there is always a price to pay for using any of the techniques.

2.6.2 Multiplanar Reformation (MPR)

Multiplanar Reformation (MPR) is an image processing technique, which extracts two-dimensional (2-D) slices from a 3-D volume using arbitrarily positioned orthogonal or oblique planes (Baek et al., 2001). This is regarded as a 2-D method but it is quite better in terms of ease of use, high speed and in fact, no information loss with MPR (Sha & Sutton, 2001). MPR is very good for visualizing blood vessels and curved MPR is

greatly useful for bypass grafts and tortuous coronary arteries (Roos, 2007; Lv, Gao & Zou, 2008). Curved MPR was combined with DVR to enhance the visualization of tubular structures such as trachea and colon (Williams et al., 2008). A major disadvantage of MPR includes the fact that it is a 2-D display technique hence it is orientation dependent. In fact, if MPR is used to display coronary arteries, it may introduce false-positive or false negative stenoses (as a major 2-D display technique) (Rodallec et al., 2008). In the actual sense of it, research confirmed the development of algorithms for automatically defining vessel centerlines, but the curved MPR is still operator dependent, this limits its diagnostic accuracy (Joemai et al., 2008).

2.6.3 Surface Rendering Technique

In principle, surface rendering first constructs a 3-D data field called *geometric primitives*, which might be by setting thresholds or using object labels (triangles, curved or flat surfaces) to define a range of voxel intensities to be viewed. The generated geometric primitives can be rendered efficiently using graphics hardware. This method is also referred to as an *indirect drawing method* (Hong & Shuhuil, 2010), its able to produce clearer *isosurface*, use the existing accelerated graphics hardware to realize the faster rendering, and can also realize the rendered surfaces dynamically.

Matching Cube, Dividing Cube, are examples of Surface Rendering. Surface rendering techniques loses one dimension in its consideration of 2-D in a 3-D space (Ponraj et al., 2011), does not fully reflect the internal information in the data field but needs to classify the data on the body that determines whether the voxel intersects the isosurface. Therefore, the classification errors often occurs while it is applied to deal with complex boundaries of human organizations and results in an incorrect equivalent surfaces or voids in the equivalent surface. Matching cubes and dividing cubes have been widely applied in commercial rendering techniques.

2.6.4 Direct Rendering Technique

Direct Volume Rendering (DVR) is a visualization technique that aims to convey an entire 3-D data in a 2-D image by assigning *semi-transparent colors* to the data samples (Lundström, 2007). DVR directly obtains a 2-D image of the original object from the 3-D volumetric data without generating any intermediates (Kaufman & Mueller, 2005). When compared with surface rendering, the main advantage of DVR is that interior information is retained, and so provides more information about the spatial relationships of different structures (Wu, 2006). DVR generally has high sensitivity and specificity for diagnosis (Drebin, Carpenter & Hanrahan, 1998) but according to Zhang et al. (2010), its computationally intensive, so interactive performance is not always optimum hence it may be difficult to interpret the “cloudy” interiors.

Direct Volume Rendering techniques can be categorized into three (3) based on the processing order of the data. *Object-Order*, *Image-Order* and *Domain-based direct volume rendering*. The Object-order methods uses forward mapping to obtain 2-D image of the original object from 3-D volumetric data, *Image-order* does this through *backward mapping* while the *domain-based* first maps the *spatial volumetric data* to another *corresponding domain* (including frequency domain) and then directly generates a projection from that domain.

Volumetric data can be visualized with any of the fundamental algorithms, the surface rendering and the direct volume rendering. Surface rendering generates geometric primitives, though efficient but it only considers a surface 2-D in 3-D space. While direct volume rendering overcomes the shortcomings of surface rendering, there is still an outstanding *cost* of increased in *algorithm complexity* and higher rendering time. Volume rendering can obtain 2-D images from 3-D image in three (3) ways. Out of the three (3) types highlighted, image-order is considered most flexible and hence most widely used. Through compositing, the colour and opacity contributions from re-sampled locations are summed up into a final pixel colour (Porter & Duff, 1984) for display while transfer function is responsible for classification in DVR to prevent any image obscuring another.

2.6.5 3-D Reconstruction

In medical diagnosing, observing and analyzing of a group of 2-D images, CT and MRI are the conventional way. However, MRI provides much greater contrast between the different soft tissues of the body than computed tomography (CT), making it especially useful in neurological, musculoskeletal, cardiovascular and oncological (cancer) imaging, thus, it is commonly used in radiology to visualize detailed internal structure and limited function of the body (Amruta et al., 2010). Whereas, especially with the usual huge information embedded in each pixel of 2-D images, to accurately identify disease with the 2-D slices, medical professionals need to mentally visualize based on their experience and expertise (Wang et al., 2011). This is prone to error, affecting the accuracy of diagnosis and disease treatment, threatening patients' life. To migrate the whole diagnosis procedure from the traditional use of considerable amount of imagination and experience to the modern technological-assisted diagnosis, the 2-D images need to be reconstructed to 3-D model, however, the current duration of the image reconstruction approach and its non-affordable cost, are currently limiting the clinical applicability of the overall imaging methods (Birk et al., 2011; Zhu et al., 2011).

The 3-D reconstruction technology is a morphological research method transforming 2-D image slices of human body into the corresponding 3-D model. Such reconstruction process is within *computer-aided design* (CAD) and it is referred to as *Reverse Engineering*; it is used to reconstruct the product model from the points cloud obtained from the original data of medical images (Song et al., 2009) to a real world scenario. Very recently, genetic algorithm was introduced to optimize the 2-D reconstruction (Siddique & Zakaria, 2010). Meanwhile, based on the report of the experiment and authors' recommendation, there were other parameters that still affected the 3-D model such as light, materials, translation, rotation and scaling, hence the approach still requires application of better image processing techniques to extract primitives and depth information from the 2-D image.

Matching cube (Lorensen & Cline, 1987) is a famous algorithm previously proposed for 3-D reconstruction. Matching cube algorithm is a surface rendering

technique; it has high computation, very costly and thus restricted in application. Recently, Archirapatkave et al. (2011) improved matching cube usage with GPGPU for a more feasible and cost effective 3-D reconstruction. He (2009) and Gong & Zhao (2010) also implemented matching cube algorithm for medical images reconstruction. However, as far as surface rendering technique is concerned, it could not still fully satisfy the need of 3-D reconstruction based on most of the limitations expounded by literatures as being reviewed in section 2.6.3. This study leverages the computational capabilities of *Compute Unified Device Architecture (CUDA)* in the development of *SurLens Memory System Architecture*; hence, section 2.7 focuses on related works in volume visualization computational hardware methods.

Table 2.1: Comparison of CPU and GPU (Ghorpade et al., 2012)

CPU	GPU
Really fast caches (great for data reuse)	Lots of mathematical units
Fine branching granularity	Fast access to onboard memory
Lots of different processes / threads	Run a program on each fragment / vertex
High performance on a single thread of execution	High throughput on parallel tasks
Great for task parallelism	Great for data parallelism
Optimized for high performance on sequential codes (caches and branch prediction)	Optimized for higher arithmetic intensity for parallel nature (floating point operations)

2.7 CUDA Technology

Compute Unified Device Architecture (CUDA) is a Graphic Processing Unit (GPU) released by NVIDIA to support high performance computing (Qureshi, Lee & Lee, 2012). Although CPU can rapidly tackle many general-purpose tasks owing to its high

clock speed, operation re-scheduling ability and large cache memory, it is less efficient in massively data-parallel applications (Qin et al., 2012). One of the greatest advancement to combat this issue introduced by Graphic Processing Units (GPU) into computing environment is in the optimization performances. While CPU is optimized for high performance on sequence codes, GPU is of higher optimization for higher arithmetic intensity for parallel nature. Table 2.4 shows detail comparison between CPU and GPU as being justified by Ghorpade et al., (2012). Evolution of Graphic Processing Units (GPUs) in the recent years has moved the computing technology to a greater effort. GPU's emergence offers solutions to the major challenges in medical imaging which are majorly constituted by increasingly large volume of medical datasets, which cannot be accommodated by the slow CPU processing and limiting hardware support.

Among the existing parallel computing platforms on GPU, NVIDIA's CUDA (Kirk & Hwu, 2010; Lei et al., 2012; Nvidia CUDA, 2009) provides an intuitive and scalable programming model based on an extended C programming language. NVIDIA CUDA technology was developed to contain hundreds of streaming processors (SPs) grouped into several streaming multiprocessor (SMs).

Computational challenges of medical visualization algorithms have been an open issue to address in the recent years. While software-based, hardware-based and parallel acceleration algorithms are the three major acceleration techniques that could be applied to the improvement of medical volume visualization processings, the latter two can improve rendering speed more effectively (Ling & Zhi-Yu, 2011). Software-based acceleration technique is in-built on the software such as flexibility, accuracy and efficient memory utilization.

Hardware-based and parallel acceleration are alternatives to the optimization of volume visualization. General-Purpose computing on Graphics Processing Units (GPGPU) serves this purpose in the field of computational sciences. The term hardware-accelerated refers to the use of GPU in handling computational applications which are usually being traditionally handled by Central Processing Units (CPU). GPU usually process data within its architecture in parallel hence such parallelism capability can be leveraged in achieving data processing optimization. Efficient utilization of GPU parallel processing architecture will offer both the hardware-based and parallel

acceleration to volume visualization. GPU has been leveraged as hardware-based acceleration technique in volume visualization of polyhedral grids data structure (Muigg et al., 2011), acceleration of texture mapping using polygon meshes Wakid, Kirmizibayarak & Hahn (2011) and in the entropy formula earlier proposed by Cao, Wu & Wang, (2011) to automatically calculate the importance of sub-block volume which can be used in the feature extraction and analysis of massive datasets. Similarly, Guo, Xiao & Yuan (2012) and Guo, Xiao & Yuan (2011) developed multivariate transfer functions that rely on GPU processing capabilities. The evaluation of the developed system shows its performance for datasets of different sizes and varieties.

The concept of data parallel computational methods has been around for many decades, dating back to military cluster computing models of the 1970's and the idea of thread or instruction level parallelism has been a central focus of the computing industry for the past 20 years (Moulik & Boonn, 2011). In the early computing era, the Application Programming Interface (API) was the order of the day, meanwhile with APIs, the vertex and fragment segments have different instructions hence their programs are separated. However, as features of these segments become more fully functioning and well developed, the instructions of both segments apparently converge. In an attempt to unify hardware programmable units, *Compute Unified Device Architecture (CUDA)*, a parallel and unified programmable architecture units was released by NVIDIA with American National Standard Institute (ANSI) for C language (Nickolls et al., 2008). Unlike the Cg (C for Graphics), HLSL (High Level Shader Language), GLSL (OpenGL Shading Language) which are inherently shading languages, in which computation must be expressed only in graphic terms, CUDA is a unified architecture. The release of CUDA (Compute Unified Device Architecture) by NVIDIA in 2006 has eliminated the need of using the graphics APIs for computing applications, pushing GPU computing to more extensive use (Fang, Varbanescu & Sips, 2011).

In a unified architecture, programmable units share the same programmable hardware. The routine works of vertex, fragment and geometric are time-shared for parallel execution. CUDA offers a unified hardware and software solution for parallel computing on NVIDIA GPUs, supporting the standard C programming language



together with high performance computing numerical libraries (Xiao, Chen & Zhang, 2009; Lindholm et al., 2008). CUDA architecture contains Streaming Multiprocessor (SM) for command execution in *Single Instruction, Multiple Thread (SIMT)* rather than the traditional *Single Instruction Multiple Data (SIMD)* model (Oiso et al., 2011). These application-processing accelerators have been driven into a number of research areas among which are level-of-details management (Tornai & Cserey, 2010), image reconstruction using Monte Carlo simulation (Praßni et al., 2010), texture mapping with bilinear warping (Vawter & Roman, 2001), level-sets (Tan, Yang & Sun, 2011) and volume segmentation (McManus & Kinsman, 2002).

Qu, Luo & Tan (2011) is one of the studies that utilized CUDA technology in volume visualization to accelerate two computation-density algorithms, the machine learning and volume rendering. The study implemented both Artificial Neural Network (ANN) training and ANN-based classification using CUDA acceleration. Chiw et al. (2012) present the designed and the implementation of Diderot, a parallel domain-specific language for biomedical image analysis and visualization. Diderot was developed to supports a high-level model of computation based on continuous tensor fields that rely on NVIDIA CUDA architecture. Diderot is a domain specific programming language with both a high-level concise notation for image analysis, visualization algorithms and high sequential and parallel performance.

Although Shi et al. (2009) solved curved-ray, prestack Kirchhoff time migration 16.3× faster than an optimized CPU version and Wang, Zhang & Yao (2009) achieve an impressive 25–40× speedup for a Fourier migration. Xie et al. (2011) based the development of their CUDA multi-resolution volume rendering algorithm on the fact that hierarchical volume rendering algorithms (Weiler et al., 2000) can accelerate the speed of rendering. The study was able to reduce the run-time performance of volume rendering and insure interpolation consistency between levels. Similarly, Hu & Hou (2011) accelerated the implementation of Non Local means (NL-means) algorithm dedicated to 3D ultrasound image with CUDA architecture which optimized parameters for searching and matching window.

Petrescu et al. (2011) present a memory implementation of the Marching Cubes algorithm using NVIDIA's CUDA. Although the proposed implementation of the



Marching Cubes algorithm has proven to offer excellent memory usage and to solve the problem of memory limitedness in an optimal manner but marching cubes algorithm is not able to represent the entire volumetric data (Ma et al., 2012) and even with high computation. Relatively, CUDA-based method to accelerate Normalization Mutual Information (NMI) registration algorithm in multimodality medical image registration was proposed by Jinzhu et al. (2011). The study shows the significance of CUDA acceleration in the experiment is not only ensuring the provision of correct registration of images but also an efficiency improvement as well.

Fundamentally, volume visualization is usually implemented as CPU-based and GPU-based. CPU made best use of sequential computing in the CPU while GPU set larger-scale scientific computing in the GPU for time optimization (Chen et al., 2011). Complex architecture of the GPUs is suitable for vector and matrix algebra operations, which leads to a wide use of GPUs in the area of scientific computing with applications in information retrieval, data mining, image processing and data compression (Krömer, Platoš & Snásel, 2011). Hence, this thesis designed and implemented new algorithm using CUDA C-programming environment that seamlessly interoperate with our new dataset pre-processing algorithms, providing hardware-acceleration to the entire datasets processing within *SurLens* framework. The compute-intensive task in *SurLens* memory system architecture is offloaded to its integrated accelerating hardware component, CUDA, for supplying computational units with high bandwidth voxel values.

2.8 Software Components

Software component is a core and integral entity of all volume visualization systems.

The software components' developed for this study composed of software application that analyze, filter and map data into geometric form, *the Graphics Execution Phase* and the key software that transforms all the brain MRI datasets into usable medical diagnostic, three-dimensional (3-D) outputs, *the Volume rendering Phase*. In order to

bring this thesis up to date, this section reviews some of the methods previously used in the software components of volume visualization.

2.8.1 Graphics Execution

Graphic execution constitutes a key pipeline in visualization. The datasets need to be represented by objects (data objects) while the objects subsequently represent processes within the visualization pipeline. The medical datasets must be represented and transformed into data objects which internally represents data, provides graphic methods of accessing the data, create methods for data manipulation and attach necessary properties into the data to facilitate rendering activities.

Some of the processes in the graphic execution and data pre-processing phase of volume visualization are complementary towards the preparation of datasets for rendering. Quality of the images in the datasets must have been greatly improved and feature extraction must be dully reliable in order to have a worthwhile visualization output. Graphic execution of volumetric data include thorough analysis, partial or full filtering and mapping of datasets depending on the type of datasets in use. Processes such as data cleansing, data normalization, data transformation, data formatting are all necessary procedures in order to have images presented in an appropriate format for rendering techniques. Bremer et al. (2011) proposed a framework to improve data analysis and visualization capabilities for high-resolution simulations. The framework uses pre-computed topological information to enable interactive exploration of large-scale datasets.

It is paramount to have a smoothened reconstruction results from visualization. Hossain, Alim, & Möller (2011) implemented multidimensional discrete Fourier transform (MDFT) filter in the pre-processing step. However, the MDFT on the BCC (Body Centered Cubic) lattice is non-separable but based on the experiment of Alim & Möller (2009), it can still be efficiently evaluated. The study also utilized the filter during the isosurface extraction stage.

Pu et al. (2009) used simple thresholding operations in the identifying point of interest in medical datasets. In the same vein is the study of Van Rikxoort et al. (2009) that utilized pattern recognition procedure, which was based on a training dataset and a supervised filter for identification of interests' areas in medical datasets. However, such detection procedure was categorized as a second-order information, similar to the contribution of Ross et al. (2010) which was applied to group points in plates.

Gu et al. (2012) developed analytical plane fitting algorithm for automatic detection of features in medical datasets. This proposed algorithm was specifically evaluated with pulmonary fissures depicted in CT examination. Zhao & McGinnity (2011) likewise achieved extraction of features using Rutovitz crossing number. The authors enhance the features in the dataset through series of procedures using Gaussian Kernel function for noise removal and morphological operations.

Peng et al. (2011) presents optimal colour mapping strategy based on energy minimization in order to reveal the features and changes of the datasets clearly. The study uses bilateral filter in the smoothing of the data and global mapping based on energy minimization to obtain colour mapping for adjacent time steps. The study was evaluated with several datasets and the results of the experiment records high efficiency and feature reservation but more effort is required to be able to balance the features at different time steps. Relatively, Yang, Sechopoulos & Fei (2011) applied multiscale bilateral filter to the processing of medical images. The authors utilized morphological methods and modified fuzzy C-means to skin mask and classify the datasets respectively. The evaluation of the study demonstrated the accuracy and robustness of the method for breast CT.

In analyzing and structuring of datasets, Jovanovic & Lorentz (2012) proposed a lossy compression method for volumetric images. This study developed the algorithm using data dependent triangulations of the third dimension. It is an extension of Silva's refinement algorithm (Da Silva & Scharcanski, 2005), which was previously proposed for mammographic medical images to the third dimension.

Maximum Intensity Projection is usually being used in medical volume visualization because of its simplicity. Zhou et al. (2011) proposed a technique, shape-



enhanced maximum intensity projection, for image enhancement in the dataset samples. Apparently, MIP is easy to use but it has limitations of loss of spatial context and shape information. Although the authors have made substantial contribution with the study but the use of global threshold in the representative gradient search is a great shortcoming of the proposed technique.

Some previous studies handled most of the major processes such as data cleansing, data filtering, feature enhancement, feature mapping and feature extraction at the pre-processing stage. However, based on the literatures reviewed in section 2.5, there could still be some elements of noise that might have possibly affected the data features, data structuring etc. during the pre-processing stage. In lieu of these, a distinct separation of pre-processing and the graphic execution phase will not only increase the level of accuracy of the data processing but will also increase the quality of the rendering output.

This study has contributed into the image-filtering, enhancement, image detection and extraction using contour filter and poly data mapper within the graphic execution phase, which is the main entry point of the datasets into the visualization phase. Our new algorithm utilized antialiasing and polygon smoothing to produce smoother edge look. With this design, we will be able to ensure thorough, final data analysis and filtering off, of unwanted materials, which might possibly affect quality of the visualization output. Features in the volumes will become more conspicuous hence, increases features' identification accuracy in as much as the image enters the visualization phase with optimum feature enhancement. This design strategy also enables our framework to handle any possible artefacts that might have unconsciously been introduced at the pre-processing phase without causing any delay to the entire visualization processing time.

2.8.2 Volume Rendering

Volume rendering is an important aspect of image processing and it has been increasingly attracting attentions in a decade or so because of its significance importance

in medical domain. Whenever *volume rendering* is mentioned in medical visualization, *direct volume rendering* is usually referred. Section 2.6.4 enumerates the superiority of direct rendering techniques over the traditional surface rendering, among which is its strength to directly generate 3-D volumetric data without generation of intermediates samples (Kaufman & Mueller, 2005) and provision of more information about spatial relationship of data structures (Wu, 2006). However, volume rendering has high overhead computational cost as a result of its complex algorithm implementation procedures (Zhang et al., 2010). The main distinguishing phases in the direct volume rendering are the *classification* and the *rendering* stages. There are several barriers to adoption of volume visualization in the clinic, including the *quality of visualization* and *overall performance* of rendering engines on commodity hardware (Wong, Wong & Tang, 2009). This section crystallizes some of the very recent materials published in this domain.

2.8.2.1 Classification

Wong et al. (2009) design transfer functions based on morphing factor function. The work inputs the start and the end transfer function from user for automatic generation of the intermediate transfer function.

Recently, Chu, Chen & Yong (2010) design an effective transfer function for direct volume rendering. Design of transfer function has a greater impact in direct volume rendering processes towards assigning color and opacity to each sample based on a measured property in the data. The work proposed a transfer function-design method based on feature variation curve. With such good design transfer function coupled with volume rendering development on GPU, the research records a very desirable speed even shows that segmentation exercises might not be compulsory for volume rendering processes.

Towards the classification of data for an effective direct volume rendering, Correa & Ma (2009) propose ambient occlusion. Since transfer function does the classification in DVR, the proposed approach uses occlusion spectrum, the distribution

of ambient occlusion of a given intensity value in a 3-D volumetric data, which provides a better two-dimensional transfer functions for complex data classification. Xujia et al. (2009) contributed to this with the research on multi-dimensional transfer function based on boundaries and present the scalar-gradient magnitude histogram.

A technique for computation of illumination using local approximation of ambient occlusion was proposed by Hernell, Ljung & Ynnerman (2010). This approach avoids fully shadow region. Outcome of the work proves that a combination of a multi-resolution framework and adaptive rendering with restriction of ambient occlusion to local neighborhood produces an appreciable interactive rendering speed. In the same vein, Lindholm et al. (2010) enhance DVR transfer function design with spatial localization based on user specific material dependencies. This work has contributed to encoding of knowledge of spatial relations and multi-dimensional data in TF domain.

As a contribution towards the finding of transfer function in Direct Volume Rendering, Liu, Wünsche & Ropinski (2010) propose a spreadsheet-like constructive visual component-based interface which automatically analyses histograms using the Douglas – Peucker algorithm.

Othman, Abdulahi & Ahmad Rusli (2010) implemented Support Vector Machine (SVM) with Field Programmable Gate Array (FPGA) that offers custom computing for classification of Brain MRI, which is an important phase in many of the direct volume rendering techniques. Distinguishing among patients with normal and abnormal brain MRI is the main yardstick to identify patients with brain abnormalities / tumor.

One of the earlier proposed algorithms for brain tissue classification for MRI is Partial Volume Estimation (PVE). This was earlier being implemented on Field Programmable Gate Array (FPGA). Koo, Evans & Gross (2009) improved on this algorithm and extended it to include probability density estimation. Human Brain MRI was used for the analysis and evaluation. Considerably better performances were achieved with the FPGA-based probability density estimation researched.



2.8.2.2 Rendering

Shihao, Guiqing & Chongyang (2009) propose a direct volume rendering with 3-D texture using hardware-assisted texture mapping. As the outstanding challenges in medical domain is how to render a 3-D image fast, the implementation uses trilinear interpolation to accelerate rendering speed. Some samples of engine CT scan, a human MRI head data set, beetle CT scan and hand CT scan data set were exploited.

Visualization Tool Kit, VTK, (Schroeder, Martin & Lorensen, 2002) from Kitware Inc. in the United States has been a great research development tool for thousands of researchers around the world. VTK enables programmatic abstraction of the access to arbitrary dimensions in the image data residing on the hard drive unlike other visualization tools (Walter, 2010). Ling et al. (2009) employ the use of VTK for context-preserving volume rendering. The idea was to contribute to the improvement of Maximum Intensity Projection (MIP) technique. Local Maximum Intensity Projection (LMIP) is an extended version of MIP introduced to overcome the shortcoming of MIP, which is its inability to adequately depict the spatial relationships of overlapping tissues. The work presents a better and improved Local Maximum Intensity Projection that computes threshold and shading for LMIP.

With the faster and more stable Graphics Processing Units (GPUs) from Intel, Shihao, Guiqing & Chongyang (2009) implement a GPU-based volume ray casting for re-sampling and representation of 3-D texture onto a sampling surface. Fragment shaders were performs with the ray-casting algorithm. Visual human male CT, human head MRI and PET chest data scan were used for the experiment. The research proved an interactive speed with the algorithm for high image data.

Ray casting technique has been noted to produce high-quality images in direct volume rendering. Kim & Jaja (2009) implement cell processor architecture for broadband engine with regular datasets for direct volume rendering. Similarly, Cox et al. (2009) propose parallel cell architecture broadband engine processor for speeding up the ray casting of irregular datasets. The work still requires optimization of the approach.

Direct Volume Rendering involves sampling and resampling of data. The resampling stage is done with the use of filters, typically trilinear, in order to ensure efficient quality image. In some cases, quadratic or cubic filters are used for higher image quality but they are expensive to evaluate even with GPU acceleration. Also, Csébfalvi & Domonkos (2009) propose a frequency-domain upsampling on an optimal Body-Centered Cubic (BCC) lattice, which was demonstrated to have similar quality as cubic filters.

To have image rendered in an interactive speed has become the greatest concern of researchers. Bentoumi, Gautron & Bouatouch (2010) implement a shear-Warp algorithm on GPU. They also studied the performance in terms of image quality and rendering speed with the ray marching algorithm implemented on GPU. In order to have acceptable quality of rendered image, add in-between slices computed by interpolation was introduced to have more number of slices. However, Mensmann, Ropinski & Hinrichs (2010) present GPU-based ray casting technique using the new programming interfaces for stream processing. Meanwhile, NVIDIA, the American global technology company, has introduced Compute Unified Device Architecture (CUDA) for GPUs as a parallel computing architecture and programming model, this creates another possible research avenue for volume visualization.

In the design of transfer functions for DVR classifications, colour and opacities are used to represent field values. This has introduced a limitation in using direct volume rendering mainly for higher dimensional data. In contribution to this, Manke & Wünsche (2009) introduce texture-enhanced DVR for visualization of supplementary data like material properties and additional data fields.

2.9 Frameworks in Volume Visualization

Inability of medical imaging devices to acquire and present the human anatomical structures in 3-D has opened up a number of research challenges in volume visualization. The base of the challenges created is in the reconstruction of 2-D sequence of the human organs, the soft tissue and the lesions sectional images into 3-D model.

The situation is highly challenging when dealing with the reconstruction of lesions sectional images and tiny features such as brain blood vessels (Subhranil-Koley & Majumder, 2011). However, usefulness of the reconstructed 3-D structures for medical diagnosis is firmly attached to the strength of the volume visualization framework in producing quality images capable of detecting, mapping and isolating abnormalities / tumor for surgery and/or disease diagnosis procedure.

Moreover, apart from producing accurate results, volume visualization framework should be able to handle mass data within a considerable interactive speed, extensive application interoperability and at a low resulting cost. Hence, with this, computer-based surgical planning will be tending towards its objective of providing surgical results with fewer procedures, decreasing time in operating room and lowering the risk of patient by increasing level of accuracy of diagnosis at a lower resulting cost. It was based on these highlighted requirements that an Application Oriented Hypothesis (AOH) was formulated in the medical community.

A group of twenty-four (24) medical doctors from all over Europe, of whom 50% have more than 5 years of professional experience, formulated Application Oriented Hypotheses (AOHs) presented by Kainz et al. (2011). These AOHs were considered during the development of *SurLens* visualization system. The Application Oriented Hypotheses (AOHs) are as follows:

1. AOH1: Medical 3-D image synthesis algorithms must speed up the information finding process to be accepted by medical doctors. For standard diagnostic procedures, 3-D representations do not provide additional information to radiologists, but they are useful to illustrate pathological findings to other medical specialists, who use that information for opinion making and intervention planning.
2. AOH2: If a 3-D image synthesis algorithm is comprehensible and if it is related to a familiar physical principle, 3-D image synthesis is accepted as diagnostic valuable tool and integrated into the clinical workflow.
3. AOH3: 3-D image synthesis gets crucial if the data input dimensionality exceeds normal human experience.



4. AOH4: State-of-the-art 3-D image synthesis algorithms are either not able to provide the necessary image quality or the necessary rendering speed, or they are restricted by the amount of input data. This prevents a common use of these techniques in the clinical practice and for clinical Augmented Reality (AR) applications or for applications where the rendering result is used as intermediate result and where the overall result must be available within reasonable time.

In the same vein, modeling physical principles and the application of geometric theorems are among the greatest challenges in achieving such set standard for medical visualization systems. Among a number of models proposed in the past for modeling medical volumetric datasets, emission-absorption model proposed by Saballa (1988) attracted utmost attention (Hege, Höllerer & Stalling, 1996). With Sabella's model, wave polarization of light, scattering, diffraction and light interferences could be ignored. Similarly, when we consider imaging processing of a camera, fundamental Euclidean theorem (Wang, Zheng & Feng, 2005) is not sufficient simply because in Euclidean, the sides of the objects have lengths, intersecting lines usually determine angles between them and two (2) lines are said to be parallel if they lie in the same plane and never meet. Meanwhile lengths and angles are no longer preserved when dealing with imaging processing and parallel lines may intersect. Based on this explanation, Lai & Yilmaz (2008) applied projective geometry in the image processing as an alternative to the fundamental Euclidean theorem.

Some of the recent previously proposed frameworks in volume visualization are reviewed in this section. Their advantages and disadvantages are highlighted for justification of our new framework.

2.9.1 Probe-Volume: An Exploratory Volume Visualization Framework

The probe-volume framework proposed by Ng et al. (2010) was based on the principle of direct volume rendering. The study focuses on producing quality 3-D images of good accuracy in terms of revealing the internal structures of datasets and avoiding occlusion.

The study utilizes a moveable and sizeable probing box in the exploration of particular volumetric space of the data enclosed by the box. The authors refer to the probe-volume size in the study as the width and height of the probing box. The probing box can be resized and moved to the desirable position to inspect volume (Ng et al., 2010). Following the fundamental principle of direct volume rendering, the volumetric space of the data enclosed by the box is then rendered with the user specified inner transfer functions using ray casting (Hadwiger, 2006).

Probe-Volume framework uses transfer functions in revealing the internal structures of the datasets. It generates texture during transfer function which is passed to a shader. This framework uses Cg from NVIDIA with OpenGL for its hardware-acceleration technique. A design of bound checking was in the framework to process either inner or outer image plane of the dataset.

The use of movable and sizeable box in the framework was designed for user to obtain unconstructed view on the inspected features of interest (Ng et al., 2010). Experimentation with the framework proved its performances with head, foot and vertebrae CT datasets.

2.9.2 VDVR: Verifiable Visualization of Projection-Based Data

The development of Verifiable Visualization of Projection-Based Data was proposed by Zeng, Wu & Mueller (2010) with the plan of addressing the need for a loss bounded transformation and compression in volume visualization. The study was based on the underlying fact that the scanning potential of medical imaging datasets acquires redundant data. The study evaluated the framework with CT datasets samples. The method was able to certify a CT reconstructed volume for use with a given interpolation filter, explicitly specifying the maximum error that might occur in the rendering.

Fastest Fourier Transform in the West (FFTW) library (Frigo & Johnson, 2005) was employed in performing of the Fourier and inverse Fourier transform. The framework utilized NVIDIA Cg as its GPU components. Provision was made for the storage and the speed of the framework by using block-marching approach which keeps



the size of the active memory reasonable. The base volume and index volume was stored into 3D textures. The rendering phase of the framework uses trilinear filter. During ray tracing, each sample position is interpolated and if the current region indicates a finer subdivision then the index points to the corresponding children in the octree (Zheng et al., 2010).

The Verifiable Direct Volume Rendering (VDVR) verifiable pipeline takes X-ray projections as input and an error threshold. The error bound is first estimated by determining the upsampling rate of the filtered projections. It was after this, the authors ran frequency domain upsampling based on the determined upsampling rate and ramp-filter the upsampled projections. Then, the reconstruction of CT data was performed. At the refinement location within the framework, error checking on the dataset is done by verifying mixed-resolution octree. The affected neighbor checking is done at the feeding location by adding continuous boundary. Figure 2.5 shows the VDVR pipeline.

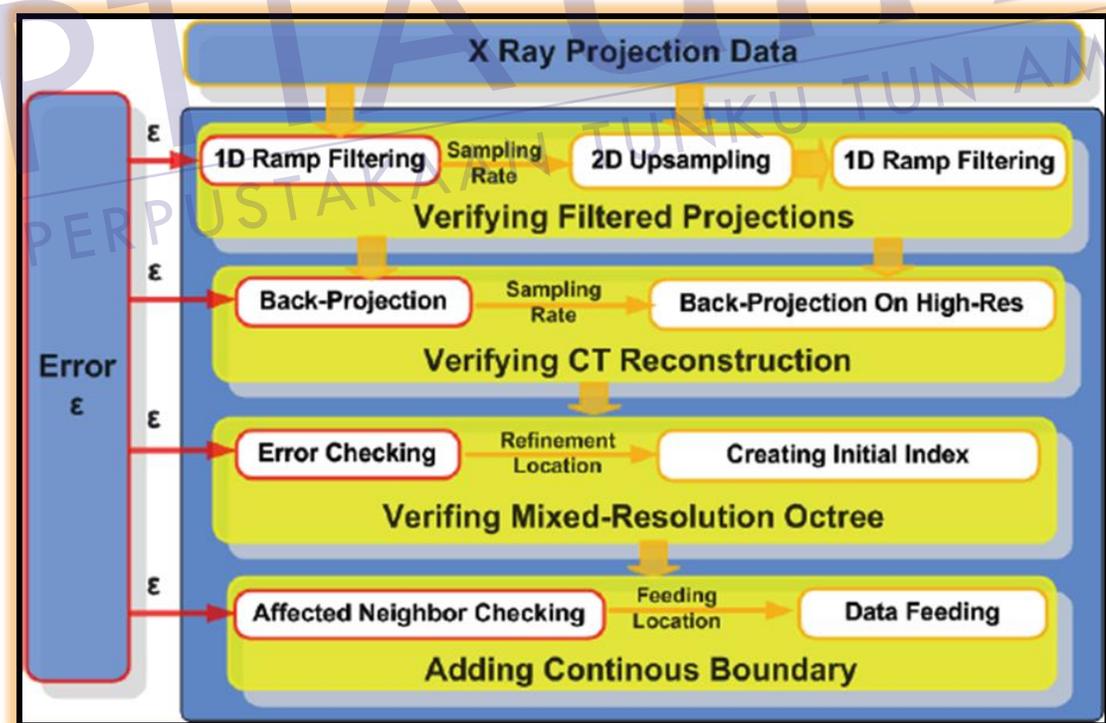


Figure 2.5: VDVR pipeline (Zheng et al., 2010)

2.9.3 GPU Accelerated Generation of Digitally Reconstructed Radiographs for 2-D / 3-D Image Registration

Dorgham, Laycock & Fisher (2012) focus on approximate methods, which reduce the computational overhead while still maintaining a quality that is sufficient for 2-D/3-D registration with accuracies that are clinically acceptable in radiation oncology (Martin et al., 2005). The implementation of the framework of Dorgham et al. (2012) was presented in two (2) steps. The study utilized the algorithm proposed by Williams et al. (2005) for their robust ray-box intersection as their first step. This step was used for the determination of rays that intersect the CT volume. The authors considered time-in and time-out computations firstly for rays used for the center pixel in the DDR for efficiency and later utilized the same parameter values for other rays.

Dorgham et al. (2012) second step of the implementation made use of a fixed internal sampling algorithm which is based on a point-based rendering algorithm for DDRs which was developed by (Shen & Luo, 2008). This approach is in line with the conventional method of ray casting. The performance of this framework was evaluated in terms of rendering time and image quality using CT volume dataset of human pelvis. The authors likewise compared the performances of the algorithm on both CPU and GPU in order to evaluate the rendering algorithms implemented in C++. CPU and GPU in CUDA was used for the evaluation.

The framework accelerates the speed of image registration processes through the use of GPU by enhancing the rendering speed. CUDA was used for the framework which improves the speed for the rendering processes without any deterrent to the image output.

2.9.4 Volume Visualization with Grid-Independent Adaptive Monte Carlo Sampling

Nakajima et al. (2009) proposed a method for sampling regular and irregular-grid volume data for visualization. The volume rendering that uses Monte Carlo method.

A Monte Carlo volume graphics (Csébfalvi & Domonkos, 2009; Sakamoto & Koyamada, 2005) were used in order to increase dataset processing augmenting the slow processing limitations of ray casting.

Metropolis method (Metropolis, 1953) was used to generate sample points using random number. To generate an image, the generated sample points are projected to 2D screen coordinates by representing $P(x)$ to be the scalar at point x in the image plane. Sampling is executed in each volume partition, the Multi-Level Partition of Unity (MPU). The authors introduced the space partitioning of the volumic MPU to improve the Monte Carlo volume rendering method employed in the study. This actually enables adaptive tuning of sampling parameters in each MPU cell. The utilized MPU was an extension of the original proposed Multi-Level Partition of Unity (MPU) (Ohtake, 2003) which aims at constructing an implicit surface from 3-D scattered points.

2.9.5 Framework for Volume Segmentation, Visualization using Augmented Reality

Previous studies in volume visualization have proved the importance of thorough volume segmentation processes in achieving reliable volume visualization. Volume segmentation is a challenging problem and it includes many individually difficult problems such as the scanning, filtering for noise removal and enhancement, and edge detection of datasets after which visualization and analysis follows (Tawara & Ono, 2010). Tawara & Ono (2010) proposed two-handed direct manipulation system to achieve complex volume segmentation of medical dataset using augmented reality and the resulting segmented data was rendered by direct rendering technique using a programmable GPU. The framework handles filtering, amplification and binarization of the dataset by the magnitude of gradients of the data at the pre-processing stage. This also includes locating a seed point and growing a region.

User makes use of the virtual needle. A seed point is placed at the tip of the virtual needle indicated by the controller in the region of extraction. Users can freely expand regions to extract desired shapes (Tawara & Ono, 2010). The study made use of

head MRI and the algorithms within the framework were made simply basic without considering smoothness and possible leakages during operations.

2.9.6 Illustrative Volume Visualization using GPU-Based Particle Systems

Pelt, Vilanova & Watering (2010) proposed an interactive GPU-based illustrative volume rendering framework, the *VolFliesGPU*. The authors adopted the illustrative concept of the software-rendered *VolumeFlies* framework presented by Busking, Vilanova & Wijk (2007). The framework adopts particle systems to produce user-configurable stylized renderings from the volume data.

Initialization of the particle system for feature location was the first step in the framework. The feature is taken in form of isosurface at the user-selected isovalue, which is followed by initialization of volume data at the user-defined grid. The entire processing procedures are based on geometry shader model. Active vertex or geometry shader threads serves as computation kernels which are triggered by rendering the vertices of the proxy geometry. Figure 2.6 below shows the GPGPU paradigm implemented.

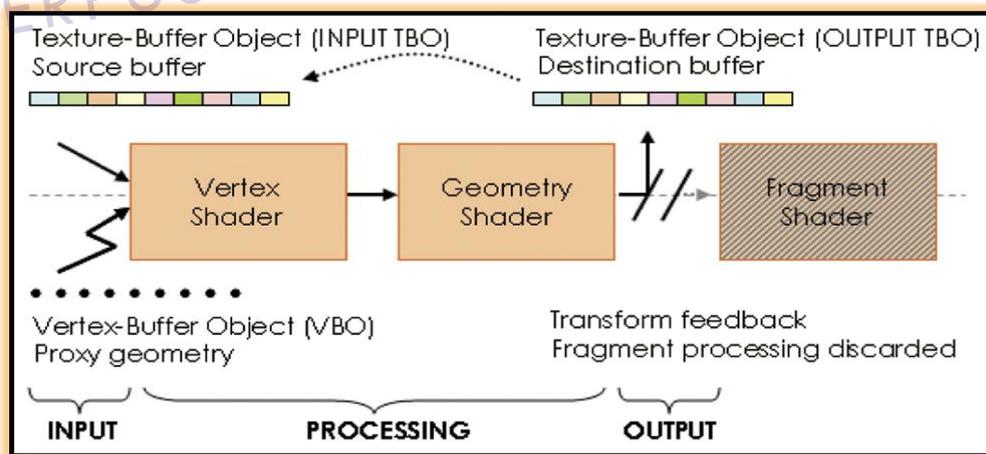


Figure 2.6: The GPGPU paradigm (Pelt et al., 2010)

The entire framework is divided into four (4) phases. Volume data from a database is passed into initialization phase with which iso-values are generated for feature location using iso-surfaces. The particles are manipulated through re-distribution using repulsive radius at the particle manipulation compartment of the framework. Filtering of the particles using hidden-surface removal via splat radius is aligned with the particle set(s) before the illustrative visualization. The entire processes are interfaced with GPU and shader model in GPGPU. Computed Tomography (CT) datasets was used in the evaluation of the framework.

2.9.7 Volumetric Ambient Occlusion for Real-Time Rendering and Games

Szirmay-Kalos et al. (2010) proposed a framework based on ambient occlusion. Ambient occlusion involves computing the illumination of ambient lighting from sky or large-area light sources. The authors are of the opinion that most of the illumination comes from such sources hence all must be computed to achieve high-quality rendering unlike the classical ambient occlusion model that considers only near occlusions. The study focuses on high frame rates on GPUs, waive pre-processing of datasets for processing time gain hence applies the general dynamic models but can only achieve smooth shading with only a few data samples.

The framework utilized DirectX HLSL (High-Level Shader Language) environment evaluated on NVIDIA GPU. The computation of volumetric ambient occlusion is achieved by approximating the volume tangent sphere's occluded part. Considering the disk base area of each sample point of pipe's cross section inside the sphere's occluded part, the pipe's volume base area is calculated which is obtained by multiplying the base area with the length of the line segment in the tangent sphere's occluded part. The length value is obtained by considering when the line-surface intersection is in the sphere, when the surface is behind and in front of the sphere.

The framework used the same quasirandom numbers for noise reduction with interleaved sampling. The argument of the authors was that they would have implemented Monte Carlo quadrature directly in the framework but because such

approach usually generates an error in each pixel, which is dependent of the particular samples used, however, they utilized the same quasirandom numbers in every pixel which would make the error corrected and replace dot noise with stripes.

2.9.8 An Improved Volume Rendering Algorithm Based on Voxel Segmentation

Zhang et al. (2012) proposed volume-rendering framework based on segmentation in the three-dimensional reconstruction. The framework includes segmentation of medical dataset, the normal vector of volume element and the entire rendering phase.

The volume visualization framework was designed and developed for visualization of CT datasets. Direct Volume Rendering is a computational intensive task that may be performed in several ways, however, scholars have proposed it can be mostly performed based on discrete methods to achieve (Zhang et al., 2012; Max, 1995). The improved volume rendering based on data segmentation proposed by Zhang et al. (2012) firstly reduces the voxel with CT data segmentation for accelerating the rendering speed, calculate the normal vector of the voxel after segmentation and then calculate the lighting effects before composing the images. The image composite stage leads to the generation of final image in the framework.

After the data segmentation, the framework picks up the boundary of various objects. This was done without considering the contribution of the internal voxel in the same object of the final image effect. However, the authors' opinion was that, in as much as the volume-rendering algorithm does not require the accurate data segmentation, the framework can use threshold for volume data segmentation. The threshold is needed for such cases where gray value of the image concentrated in two regions.

The study attempted to provide a solution to the major drawback in ray casting algorithm, which is the difficult task of choosing accurate and effective way to calculate the vectors and processes of the ray effect to emergent sense of space. The framework uses trilinear interpolation associated with the direction needed to obtain the intersection and its normal vector of the sight direction and iso-surface, which is with the assumption

that the line of sight through the volume element could get an impact for a pixel with the calculation of the ray effect at the intersection.

The image synthesis employed in the framework followed the back-forward approach. The voxel on the boundary is projected slice-by-slice and line-by-line on the projection plane for any sight direction. The framework employed the combination of phong, diffuse and ambient model in the image synthesis.

2.9.9 ParaView Visualization Framework

ParaView is an application framework developed by Kitware Inc., Los Alamos and Sandia National Laboratories in the United States. The research which is funded by Trilabs and Army Research Laboratories started in 1999 (Ahrens, Geveci & Law, 2005) and continues until date.

ParaView was built on top of Visualization Tool Kit libraries (VTK). VTK was based on the visualization pipeline approach, a key metaphor in many development systems and on which most of today great visualization libraries and application are based on because it tolerates abstraction and likewise users can combine algorithms in powerful ways (Moreland et al., 2011). ParaView uses VTK as data processing and rendering engine and avoids the use of centralized execution to obtain a scalable solution (Sanftmann, Cipriani & Weiskopf, 2011). The framework consists of Tcl /TK and C++. It uses TK as the widget sets and C++ objects created for encapsulation of widgets (Law, Henderson & Ahrens, 2001).

The level-of-details (LoD) technique was implemented in ParaView to increase frame rates for interactive rendering. Although with the parallel compositing module in this application, large datasets size can be handled but with slow rendering rates (Law, Henderson & Ahrens, 2001). Cost of communicating the frame buffers to the first mode for display is also added onto the entire application processing computational cost. However, the implementation of LoD in the framework increases its level of rendering interactivity.

ParaView follows the visualization pipeline of Visualization Tool Kit with the main execution paradigm being the concept of a downstream flow of data. Its data streaming within the visualization pipeline is facilitated following piece by piece, releasing / re-using memory after each subset or accumulating sub-object representations for the final image. Data coming into the framework can be portioned with a quadtree (or octree) subdivision by pieces in the geometric description or reading all the data into the memory.

One of the key stages in ParaView framework is data filtering which typically involves generating, extracting and deriving features from the data. This stage verified data for geometrical and topological coherency of data structures. Rendering of the processed data is the last phase of the framework. In the case of parallel computation, the source base will distribute data pieces to multiple execution engines, separable into data input and output, processing and rendering stages of the framework. ParaView framework requires OpenGL library for its image rendering.

2.9.10 VolView Framework

VolView is a visualization framework that allows the user to visualize volumetric data (Martin et al., 2005; Cheung & Krishnan, 2012). It has facility for both surface and volume rendering. It is owned and developed by Kitware Inc., United States as their premiere 3-D application developed on top of Visualization Toolkit (VTK). The development of VolView began in 1999 with the target of developing a high performance volume rendering framework.

Segmentation is a core pre-processing stage in VolView framework. It mandatorily requires thorough segmentation of the datasets in order to have a reasonable visualization results. The framework is designed to support a wide variety of data types from single component short to RGBA volumes with transfer function editor for material classification. These functionalities complement VolView framework in ensuring efficient data segmentation prior to data visualization.

VolView uses Maximum Intensity Projection (MIP), composite for recording image calculations and orthogonal data structuring for storing and manipulation of datasets. However, as earlier being justified in section 2.8.1, although MIP is easy to use and might not necessarily impose computational overheads in its implementation, but it has the greatest limitation of loss of spatial context and shape information. Apparently, spatial context and shape information play key roles in the representation and presentation of features and edges of datasets in 3-D model, without which accurate detection, mapping and isolating of abnormal features in the datasets will not be feasible to achieve.

2.10 The Advantages & Disadvantages of Previous Volume Visualization Frameworks

Probe-Volume framework proposed by Ng et al. (2010) attempted addressing the challenges of generating quality images and achieving a remarkable rendering. The authors' approach likewise dealt with dataset occlusion issues which is a threat to quality images in 3-D volume visualization. However, there was no consideration for data reduction and data filtering after the pre-processing stage. This may possibly introduces overhead in the framework.

Probe - Volume framework could only be applicable to head CT, it cannot reveal the internal structures of some structures such as vessels in the brain and abnormalities. Moreover, although the framework uses hardware-accelerated approach in gaining better rendering processing time, but unfortunately the authors were only able to use Cg shading language from NVIDIA with OpenGL for the graphic hardware –accelerated ray casting. Cg and OpenGL shading languages are inherent shading languages, they only accept computation in graphics terms. This limits the Probe-Volume framework capabilities. The incurred limitation of Probe Volume framework in terms of the hardware-accelerated procedure makes the framework high computational.

The VDVR framework is a robust framework that has contributed immensely to the efficient visualization of datasets, however, it is only able to fully address

transformations and compressions i.e. data reduction in volume visualization. VDVR is not a complete framework that can be adopted independently to volume visualization. Moreover, the framework still inherits most of the notable outstanding challenges of previous framework in terms of hardware-acceleration approach, use of Cg shading languages, which is an inherent shading language. The framework is only applicable to CT datasets and not MRI.

GPU accelerated generation of digitally reconstructed radiographs for 2-D/ 3-D image registration proposed by Dorgham et al. (2012) would have been a breakthrough in volume visualization because it focuses on approximate methods which reduce the computational overhead while not neglecting the acceptable clinical image quality of datasets but the authors did not make any consideration for management of shared device cache. This limited the performance of the application. Nearest neighbor interpolation was used which limited the quality of output image.

In the GPU accelerated generation of digitally reconstructed radiographs for 2-D/ 3-D image registration, CUDA is used. This is an advantage over Probe - Volume framework (Ng et al., 2010) and VDVR framework (Zheng et al., 2010) because CUDA supports general-purpose programming (Kirk & Hwu, 2010) unlike Cg or GLSL, which is mainly useful for graphics applications. However, GPU accelerated generation of digitally reconstructed radiographs for 2-D/ 3-D image registration is not efficient for visualization of tiny features like brain blood vessels in which data input dimensionality exceeds normal human experience. Moreover, the off-the-Shelf GPUs used for this framework do not have and not yet in consideration for Food and Drug Administration (FDA) approval. A more suitable platform for clinical evaluation would be the NVIDIA Quadro or Tesla GPU.

The GPU accelerated generation of digitally reconstructed radiographs for 2-D/ 3-D image registration framework was implemented in C++. C++ has limited interoperability compared to .NET platform which can seamlessly be attached to all presently existing and possible future medical revolutionary tools.

The volume visualization with grid independent adaptive Monte Carlo Sampling framework proposed by Nakajima et al. (2009) maintained quality 3-D images but could only achieve better rendering time with non-real world medical datasets due to the



overwhelming size of the most real world medical datasets. However, the framework for Volume Segmentation and Visualization using augmented reality proposed by Tawara & Ono (2010) was able to produce good processing time of datasets but does not take into account smoothness of extracting shape and edge detection after which visualization and analysis follows (Tawara & Ono, 2010).

Illustrative Volume Visualization using GPU-Based Particle System was based on shader model paradigm falling onto the limitations of inherent shading language in the same vein as the frameworks proposed by Ng et al. (2010) and Zheng et al. (2010). In addition, the framework cannot render large datasets because it has memory limitations. The amount of particle has a great influence in the performance of the framework. Higher amount of particle affects the final quality of image achieved. However, *VolFliesGPU* employed in the framework added more value to the volume visualization output as the framework has no dependence on the resolution of the render window.

The framework, illustrative Volume Visualization using GPU-Based Particle System, was implemented in C++. Having API limitations, similar to the previous frameworks implementations' issues, such as that proposed by Dorgham et al. (2012) and OpenGL 3-D graphics, making it only useful in graphics applications. The framework has no provision for feature detection and mapping of tiny features such as brain blood vessels just as all the framework reviewed so far; Ng et al. (2010), Zheng et al. (2010), Dorgham et al. (2012), Nakajima et al. (2009) and Tawara & Ono (2010). The image quality of illustrative volume visualization using GPU-based particle system was poor compared to those frameworks that used volume-rendering technique.

The Volumetric Ambient Occlusion for Real-Time Rendering and Games proposed by Szirmay-Kalos et al. (2010) focuses on high frame rates on GPUs, waive pre-processing of data for processing time gain. The framework records good volume visualization performances in term of processing time but can only support fewer samples. Even for the fewer samples, it only achieves smooth shading with only few data samples.

Among other limitations of the framework is the outstanding issue of most of the volume visualization frameworks, the use of shading language. The framework used



DirectX HLSL (High-Level Shader Language) in NVIDIA environment, which streamlined the framework to use only with graphics applications. Moreover, the framework left many issues un-addressed in volume visualization such as data reduction, detection and extraction, due to its target on achieving processing time gain of datasets.

Recently, the development of an improved volume rendering algorithm based on voxel segmentation made immense contribution with the implementation of volume rendering technique in the target of producing quality image for medical diagnosis and disease therapy. The framework attempted to provide solution to the major drawback of the difficult tasks of choosing accurate and effective way to calculate the vectors and processes of the ray effect to emergent sense of space in ray casting algorithm. With prior segmentation procedures, the framework was able to produce quality image of CT datasets of the head. However, it cannot be applied to revealing internal features of the brain.

In terms of processing time, only software acceleration technique is applied. This does not produce remarkable processing time. With the computational intensive nature of volume rendering and the usual added time from segmentation procedures within the framework, hardware-based and parallel acceleration optimization procedures are needed.

GPU accelerated generation of digitally reconstructed radiographs for 2-D/ 3-D image registration proposed by Dorgham et al. (2012), ParaView Visualization framework (Moreland et al., 2012; Sanftmann et al., 2011; Law et al., 2001) and Volview Visualization framework consider parallelism in particular as a way of increasing processing speed, however, there should be a better consideration and concentration on the entire visualization process not just on the rendering speed. The framework architecture of ParaView and VolView framework follow the visualization pipeline paradigm, which facilitates piece-by-piece, releasing and re-using memory after each subset of accumulating sub-object representations for the final image. This is a great advantage over all previously reviewed frameworks. However, data reduction is lacking in ParaView and Volview frameworks. The data reduction is crucial in order to adapt the framework to handling mass data within a considerable interactive speed, extensive application interoperability and at a low resulting cost. Moreover, ParaView



PT TAAUTHM
PERPUSTAKAAN TUNJUKKAN

and VolView frameworks are tied up to mandatory segmentation procedures virtually like all existing volume visualization frameworks, this increases the entire processing time. Reconstructing 2-D sequence of human brain MRI, particularly the soft internal tissues and lesions sectional images, to 3-D model within a considerable time is crucial to volume visualization framework.

According to the investigation report of Mensmann et al. (2010) which addressed volume visualization using ray casting on GPU, only small gains in performance are possible unless factors such as the shared memory model are featured into the design. All those previously reviewed frameworks do not manage a share device cache. Generally, the authors' opinion was that ray casting has no reflection and that the nearest model, nearest neighbor interpolation enhance the model's performances, however based on Mensmann et al. (2008) proof, such opinion has a lot of limitations as samples are only accessed once. Timely detection, mapping, isolating of abnormalities of the internal soft tissues of the brain MRI that could fully support surgical planning and therapy procedures require better attention. Although the parallel compositing module of ParaView Visualization framework is designed to handle any size of dataset, it still has overhead issue of slow rendering rates (Law et al., 2001).

2.11 Summary

Volume visualization is a sub-field of scientific visualization that explore, sample, classify and represent volume data for better medical examination. Medical image devices such as Computed Tomography (CT) and Magnetic Resonance Imaging (MRI) are commonly used in medical imaging techniques. Whenever any of these imaging devices is used to scan human anatomical structure, the resulting data is a *volume data* containing *anatomical information* of the scanned human body. Although these data lack optical information, the color and opacity, but they have associated *scalar value* (called *density or intensity*), which quantifies the reaction of the corresponding medium to the scanning device. The slices are in 2-D and are often arranged in regular grid of volume elements called *voxel*. The *voxel* constitute the 3-D equivalents of pixels from the 2-D



PTTA
PERPUSTAKAAN TUN AMINAH

images of the scan device. However, depending on the type of device used, the scalar values are respective correspondence of densities of various tissues relative to the scanning device. Bone, muscles, and vessels, all have different associated scalar values that could be used in their accurate mappings and separations using computing and numerical operations. Producing 2-D images from scan devices in 3-D model involves subjecting the datasets into various computational stages. Loading the data into supporting and reliable memory architecture, reconstruction of the volume from the data sets, sampling using mathematical models, segmentation for determination of type of tissue present for each pixel or voxel in the 2-D and various other techniques and enhancements, are the typical procedures.

Medical diagnosis using medical data produced by image modalities requires studying of the 2-D images by medical professionals followed by analysis based on their expert experience, which takes time and prone to error especially when some features are occluding the region of interest. With the attached difficulties to getting the images available in the better medical diagnosis format using computing architecture, medical volume visualization is still limited in use. The conceptual review of the above literatures crystalizes direction for this research.

The most recently and widely applied medical imaging techniques for visualization of structure and a function of body is Magnetic Resonance Imaging (MRI). Apart from the fact that it is suitable for imaging of the brain and other soft tissue, it is more flexible, provides a better spatial resolution with higher discrimination and effective in providing detailed images of the body in any plane.

Direct Volume Rendering (DVR) method is quite efficient in 3-D Visualization. It is an approach that directly obtains *3-D volumetric* data from *2-D images* without generating any *intermediates*. Ray casting follows the approach of direct volume rendering. Though there are various ray-casting techniques, generally, literatures say ray casting technique is *slow* mainly because of its mandatory processing requirement of huge amount of rays in the procedure. Each of such processing involves complex calculation, resulting in total long *time computation*. However, ray casting speed can be improved through *software optimization*, *hardware* and/or *parallel acceleration* approaches.



Feature detection, automatic local feature mappings are significantly important to have a closer inspection of the features in brain MRI during medical diagnosis. With this volume visualization capability, it becomes easier for medical practitioners to clearly see the internal features of the datasets, especially in the case of tiny structures like vessels without consideration for surgical operation before proceeding with the diagnosis and treatment of the patient concerned.

In the same vein, enabling general acceptability and usefulness of volume visualization in medical community, such a computational system's architecture must be robust enough to handle mass data, within considerable interactive speed and with extensive application interoperability for present and future revolutionary tools' compatibilities. Moreover, according to the reviewed literatures, most of the hospitals cannot still afford the high cost of 3-D medical image technology. A 3-D medical imaging system at a lower resulting cost will contribute immensely towards the use of volume visualization in medical communities.

With respect to the highlighted limitations of use of volume visualization at the level of this survey, there is no 3-D medical image system yet that fully closes the highlighted shortcomings of volume visualization, particularly for brain MRI. The next chapter discusses the proposed research methodology, the framework, the numerical computations, the algorithms and the data structures for this research.



CHAPTER 3

METHODOLOGY: THE DEVELOPMENT OF *SurLens*

The visualization system developed for this study is named *SurLens* (short for Surgical Lens). This chapter discusses *SurLens*' framework, numerical computations, algorithms, dataset collection and data structures used in the development of *SurLens* visualization system. In justification of our proposed framework, schemes, algorithms and techniques, strengths and the weaknesses of the previously proposed frameworks, including their proposed schemes, algorithms and techniques are compared and presented in the previous chapter. Each of the components of *SurLens* framework has been presented in conferences and published in journals as being documented in page (vii) of this thesis.

3.1 Introduction

The framework, numerical computations, algorithms, dataset collection and data structures used in the development of *SurLens* visualization system are discussed in this section. With the broad review of related work in chapter 2, it is obvious that many visualization algorithms have been proposed since the advent of the first scanning device. However, as being documented in the previous chapter, when working in the clinics or in close collaboration with medical doctors, one might have the impression that none of these algorithms ever found their way into daily use; diagnostics and treatment planning are done in 2-D slices of the scanner's raw data by radiologists and surgeons. Hence, *SurLens* Visualization System was developed based on the Application

Oriented Hypotheses (AOHs) formulated by a group of twenty-four (24) medical doctors from all over Europe, of whom 50% have more than 5 years of professional experience. The AOHs are documented in chapter 2 and were considered during the development of *SurLens* visualization system.

3.2 *SurLens* Architecture

The architectural design of *SurLens* Visualization framework is built with C# programming language in .NET platform for optimum benefit from object-oriented paradigm. .NET is a parent platform compiling CIL (Common Intermediate Language) through which applications can connect and interoperate. With the fast growing and integration of various revolutionary tools into medical therapy procedures, interoperability of application is an issue to reckon with.

SurLens Visualization framework consists of four (4) major phases, the *SurLens Data Pre-processing*, *SurLens Accelerating Hardware*, *SurLens Graphic Execution* and *SurLens Volume Rendering* phases. Within each of these phases are respective algorithms integrating the phases in pipelined fashions. CUDA is interfaced with *SurLens Accelerating Hardware* phase of *SurLens* Visualization framework for hardware acceleration. The design and the implementation of *SurLens* Visualization framework is built on Visualization ToolKit (VTK) libraries, serving as its rendering engine for efficient pragmatic abstraction needed for optimum algorithm performances. Figure 3.1 shows *SurLens* architecture.

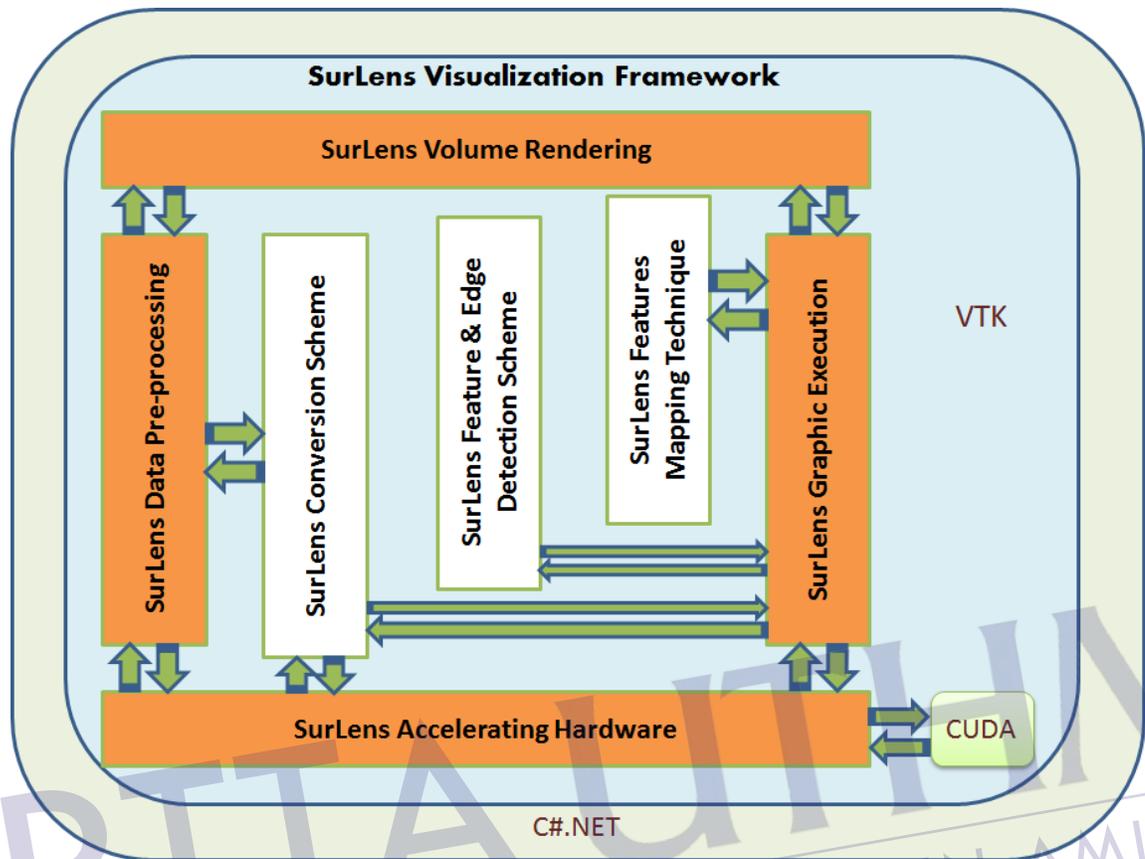


Figure 3.1: *SurLens* Architecture

In the same vein, two (2) major schemes and a feature mapping technique are designed and integrated into the architecture. The *SurLens Conversion Scheme*, and the *SurLens Feature and Edge Detection Scheme*. These schemes facilitate thorough transformation of the datasets and seamless detection of their edges and features within the framework. The designed and implemented feature mapping technique supports *SurLens* framework with necessary *graphic* and *writer* mapping functions in order to map and isolate abnormalities / tumor in the datasets.

3.3 The Framework of *SurLens*

SurLens framework is illustrated in Figure 3.2. The methodology used in the development of *SurLens* visualization system is shown in Figure 3.3. According to Figure 3.3 the major phases in the development of *SurLens* are the *Dataset Pre-Processing*, *Accelerating Architecture*, *Graphic Execution* and *Volume Rendering*. The *Dataset Pre-Processing* and *Accelerating Hardware* constitute the *Memory System Architecture* of the framework and is typically concerned with transforming the medical datasets from the usual raw format to ready-to-use and easily manipulated format for *SurLens* optimization. An efficient memory system has a great impact on the performance of volume visualization; *Memory System Architecture* of *SurLens* framework handles such task. Figure 3.15 shows the condensed *SurLens Memory System Architecture*. In the design of memory system architecture, it is understood that transistor does the amplification, switching of electronic signals and power. However, GPUs usually have many more transistors than CPUs, hence, the compute-intensive task in the volume rendering could be offloaded to the dedicated hardware. With this architectural arrangement, the throughput of our memory system is independent of various viewing positions. *SurLens* data pre-processing was carefully designed to read dataset sample using image reader and then writing to XML (Extensible Markup Language) image data. *SurLens* utilizes Compute Unified Device Architecture (CUDA) to facilitate the computational units with voxel values.

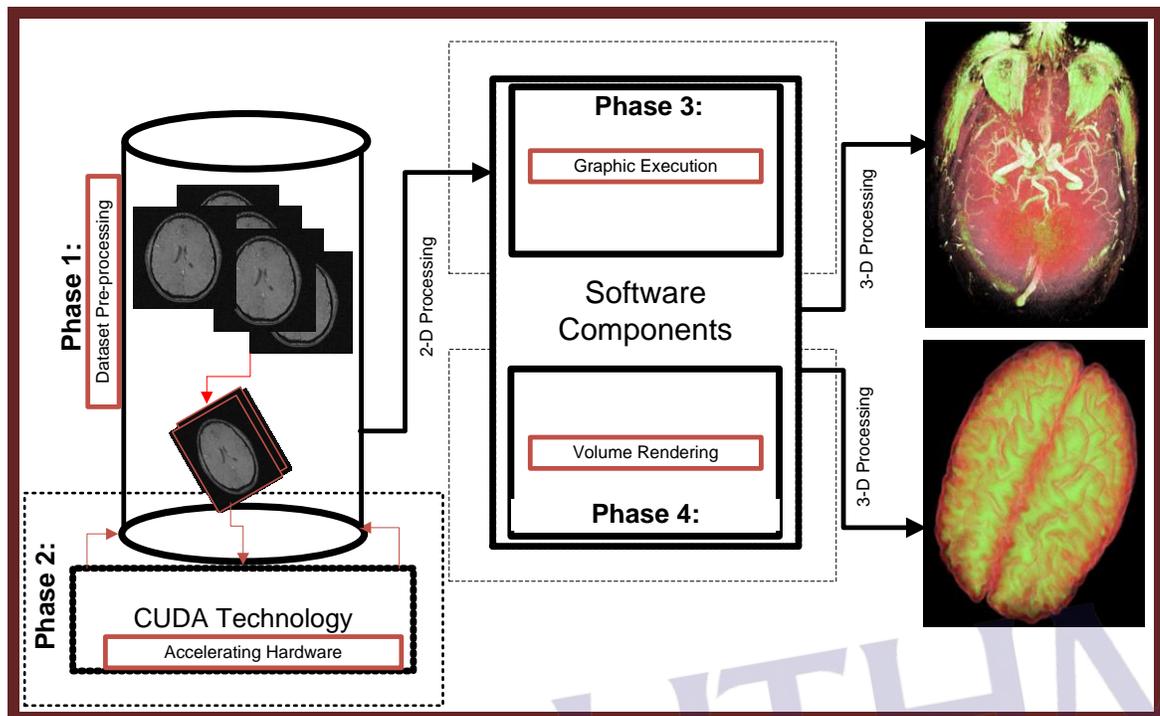


Figure 3.2: *SurLens* Framework Overview

SurLens graphic execution represents Phase 3 of the framework. This phase consists of data objects that represent data, objects that process, transform, filter and map data objects from one form into another. This phase also includes the control objects that handle the execution, controlling the entire execution of data and processing of objects in the phase. The phase receives initial flow of data from the *SurLens* Memory System Architecture, which is subjects to further routine check and modification (if needed) such as conversion, reduction and merging in the phase before delivering the data to Mapper, the geometric representation for an *actor*. *Actor* represents the object rendered in the scene, with both its properties and position in the world coordinates. More than one actor may refer to the same mapper, a data source and/or filter can drive more than one filter and/or mapper. Mappers serve as a transition between the focus data and the geometric data resulting in the actor, the input file for the *SurLens* volume rendering phase.

Volume rendering is the concluding phase of *SurLens* development. The output

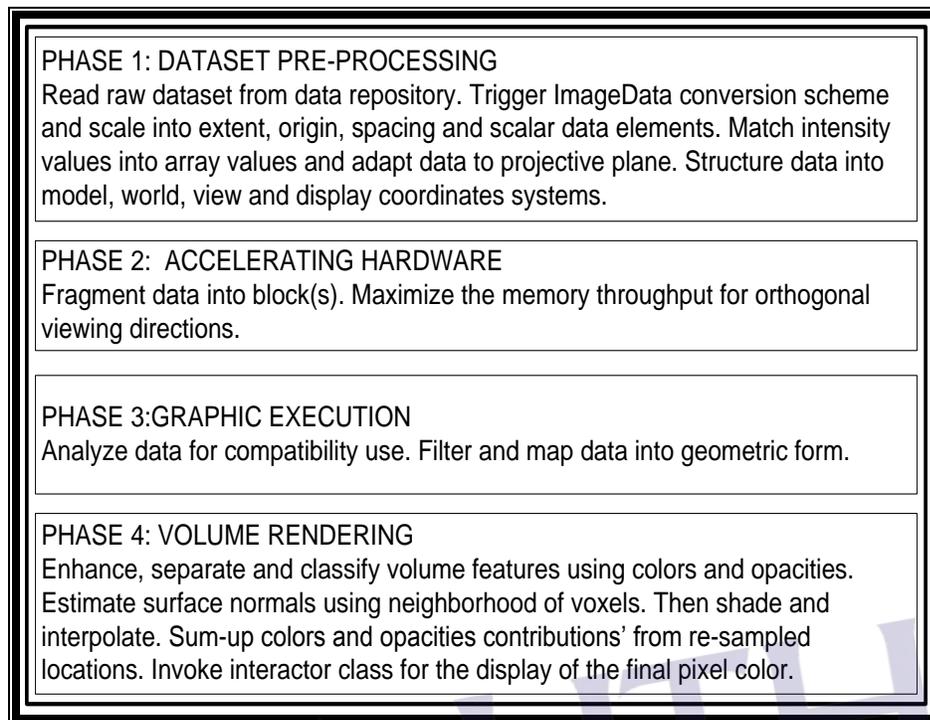


Figure 3.3: *SurLens* Framework Phases

file from *SurLens* phase 3 serves as input for this phase. Typical volume undergoes classification procedure during which colour, scalar opacity, feature enhancement and separation is introduced. The enhancement and feature separation procedures are to facilitate clear identification of any tumor or abnormalities in the volume. The intensity values of the features are categorized, and abnormal or tumor pixels intensities, which normally ranges from 100 to 190, are placed on the 0 to 255 *SurLens* pixel scale. *SurLens* uses colour and opacity transfer function to differentiate different points or cells during rendering.

SurLens is designed to choose *Flat*, *Phong*, *Gourand shading* or their combinations for better image shadings. Such selection preference is useful in cases where the background illumination of an object, which is the ambient functionality, the bright and shiny reflections, the specular and the non-shiny illumination and shadows of diffuse effect are needed in the rendering of the object. Interpolation is very important for filtering or smoothing of the resulting images. *SurLens* utilizes *trilinear interpolation*

simply because *trilinear interpolation* usually produces smoother images with less artefacts, though *trilinear interpolation* may possibly be a bit longer in implementation compared to the *nearest neighbour* but this is augmented with *SurLens* CUDA memory architecture. Some of the factors to be considered during compositing are the absorption, emission and scattering based on the principle of transportation of light. However, for the feasibility of volume rendering in the visualization domain, it is practically impossible to model scattering of light, *SurLens* development follows Sabella (1988) emission-absorption model, usually referred to as density-emitter model, which ignored scattering of light. The detail of the components of *SurLens* framework is discussed in the following sections.

3.3.1 Dataset Pre-Processing

Datasets acquired from medical imaging devices are usually in different structures being relative to the device employed, this has been extensively discussed in Chapter 2. Even with similar acquisition device, anatomical structures of the human body differ in their relaxation times. The different relaxation times of the structures resulted in distinct MR signals during MR examination, which can be used in the reconstruction of the 2-D MR slices into 3-D showing different contrast for different tissues. However, understanding of data and its adaptation for visualization is a key step to developing *SurLens* Visualization framework. A number of established mathematical principles are applied to *SurLens* dataset pre-processing.

3.3.1.1 Application of Projective Plane Theorem

Projective plane is seen as a geometric structure with extended concept of a plane. However, in ordinary Euclidean plane, unless line cross each other and intersect, parallel lines do not intersect. Meanwhile a projective plane with any two lines intersect in one and only one point called *vanishing point*, Figure 3.4 illustrates vanishing point in the case of train rail. Vanishing point is a point where parallel lines that are not parallel to

the image plane appear to converge, this could be better interpreted with projective plane.

In the development of *SurLens*, using homogenous coordinates principle, a point x,y of 2-D slice in the Euclidean plane is represented in the projective plane (3-D) by adding a third coordinate 1 at the end $(x,y,1)$. This is based on the fundamental Euclidean theorem which states that a point in an n -dimensional Euclidean space is represented as a point in an $(n+1)$ -dimensional projective space. However, overall scaling is not important. This work uses X, Y, Z as coordinates notation and $[X, Y, Z]^T$ as a vector notation which could be interchangeably used in further discussion.



Figure 3.4: Parallel Lines in Projective Plane

The MRI slices is abstractly represented as stack of images as in Figure 3.5. It is assumed there are points U_i with $i=1, 2, 3, 4$ arranged parallel in line with *plane* π . Since there exist such many slices, we assume the slices are moved up a distance Z as shown in Figure 3.5. With such moved distance of the slices, there will be a formation of new sets of points $U_{i'}$ with $i' = 1, 2, 3, 4$ leaning on a new *plane* π' . The first issue to address is estimation of the new points $U_{i'}$ automatically which can be done by estimating directly from the first *plane* π .

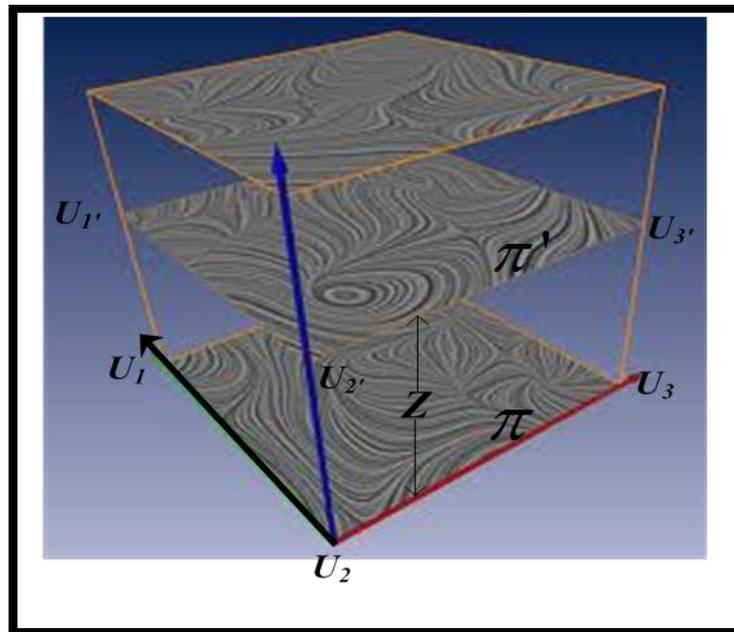


Figure 3.5: Point Estimation of 2-D Slices

At this point, it can be assumed that U_1 and U_3 are known, hence, U_2 and U_4 can be estimated and computed by applying intrinsic properties of the vanishing points. Figure 3.5 shows the point estimation of 2-D slices. The vanishing point of the parallel lines leaning on *plane* π could be computed as follows:

$$V = (U_1 * U_2) * (U_3 * U_4)$$

However, based on projective geometry, which describes the physical characteristics of the virtual camera and the relationships between the images, the projection of a point X_w in the object space to a point U_i in the image space using projective camera is expressed in terms of a direct linear mapping in homogeneous coordinates as:

$$\lambda U_i = PX_w = [p_1 \ p_2 \ p_3 \ p_4] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (1)$$

where λ is the scale factor due to projective equivalency of $(k_x; k_y; k) = (x;y;1)$, P is a 3×4 camera projection matrix and p_i is the i^{th} column of P .

As earlier discussed, with homogenous coordinates representation, value 1 in the last row of the vector denotes that the defined point lean on the image plane. However, if the point in the object space lean on ground plane $Z=0$, hence the linear mapping will change to equation (2).

$$sU = HX'_w = [p_1 \ p_2 \ p_4] \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \quad (2)$$

H is the homography matrix mapping points lying on a plane in the object space across different images, s introduced scaling factor in the mapping equation stems from setting Z to 0 .

In order to establish relationship between U_i and U_i' , we can re-state equation (1) above as:

$$\lambda_i U_i = [p_1 \ p_2 \ p_4] \begin{bmatrix} X_i \\ Y_i \\ 1 \end{bmatrix} + p_3 Z \quad (3)$$

where p_3 corresponds to the vanishing point in the direction of Z axis or the normal of the ground plane.

The main target is to project the lines and points that made up the 2-D slices in 3-D. The Euclidean formula for s line is $ax+ay+c=0$, this is regarded as non-zero

scaling factor since the equation is unaffected by scaling, we can however arrive at the following:

$qX + rY + sZ = 0$ where q, r, s are the homogeneous coordinates of points (x, y) in the line.

$$t^T p = p^T t = 0$$

$t = [q, r, s]^T$ representing the line

$p = [X, Y, Z]^T$ representing the point

Substituting V_z for p_3 in equation (3) and combining the result with equation (2), we have:

$$\lambda_i U_i = s_i U_i + V_z Z \quad (4)$$

λ_i and s_i are the unknowns from this equation, though they were both defined earlier in equation(1) and equation(2). We can estimate the respective values of λ_i and s_i by using equation (5)

$$\begin{bmatrix} \lambda_i \\ s_i \end{bmatrix} = (A_i^T A_i)^{-1} A_i^T b_i \quad (5)$$

$$A_i = [U_i | -U_i]$$

$$b_i = V_z Z \text{ and } V_z = p_3$$

Since s_i is estimated, we can continue setting different values for Z in order to estimate any other image point along the lines. Hence, we can equally estimate U_2 and U_4 as follows:

$$U_2 = (U_2 \times V_z) \times (U_1 \times V) \quad (6)$$

$$U_4 = (U_4 \times V_z) \times (U_3 \times V) \quad (7)$$

The applied homogenous coordinates to this study for points and lines estimation is summarized as in Table 3.1.

Table 3.1: Homogenous Coordinates Representation of Points and Lines

	Points	Lines
Point	$P = (X, Y, Z)$	$t = (q, r, s)$
Incidence	$P^T t = 0$	$P^T t = 0$
Ideal points	$X, Y, 0$	$(0, 0, s)$
Joining of points (2-points)	$T = p_1 * p_2$	$P = t_1 * t_2$
Collinearity	$ P_1 p_2 p_3 = 0$	$ t_1 t_2 t_3 = 0$

The representation of points and lines using homogenous coordinates made it feasible to estimate points that lie on the same line and likewise lines that are concurrent i.e. intersect at the same point. If points p_1, p_2 and p_3 lie on the same line, then the line joining the points p_1 and p_2 is estimated as $p_1 \times p_2$ while the third point p_3 lies on $p_3^T (p_1 \times p_2) = 0$, provided the determinant of $[p_1 p_2 p_3] = 0$. In the same vein, the lines t_1, t_2, t_3 are estimated by assigning $t_1 \times t_2$ to be the intersection of line t_1 and t_2 , while $t_3^T (t_1 \times t_2) = 0$ is for the intersection of the third point provided the determinant of $[t_1 t_2 t_3] = 0$. Homogenous coordinates representation of points and lines are shown in Table 3.1.

By example, finding the intersection of two lines at point $t_1 = (q_1, r_1, s_1)$ and $t_2 = (q_2, r_2, s_2)$ can be obtained using the elementary algebra, cross product. Hence the intersection at a point $P = (r_1 s_2 - r_2 s_1, q_2 s_1 - q_1 s_2, q_1 r_2 - q_2 r_1)$. However, if two lines are in parallel such as $-q_1 / r_1 = -q_2 / r_2$, the point of intersection is $(r_1 s_2 - r_2 s_1, q_2 s_1 - q_1 s_2, 0)$ being the ideal point for slope $-q_1 / r_2$.

3.3.1.2 Coordinates Systems

SurLens Memory System Architecture is typically concerned with making the data available for its optimal architectural use. It is at this stage that series of the original data slices are stacked, shaped and positioned for flow. Properties such as *rotation*, *scaling* and *translation* are likewise necessary in the data for better *value distribution*. Coordinates system is greatly useful for the success of data preparation hence *SurLens* is developed to use the *model*, *world*, *view* and *display coordinates systems*.

The *model coordinate system* is typically a local *cartesian coordinate system*. As the name *model* implies, the coordinates system in which model is defined. This type of coordinate system is locally defined by the modeler. We can refer to this as an *inherent coordinate system* based on the decision of the person that generates it. The units used in its definition may be *meters*, *inches* or *feet* and its axis might be arbitrary, these are based on discretion of the modeler.

The *world coordinates system* is the *3-D space* where *actors* are positioned. Unlike model coordinates which is typical local Cartesian Coordinates System, world coordinates is the only standard coordinate system where all actors *locally defined coordinates systems* are converted to. The world coordinates system is the coordinate system that all the actors are *scaled*, *rotated* and *translated into*. Moreover, the position and orientation of cameras and light are specified in the world coordinates system.

The *view coordinates system* is directly referenced to the *camera*; it represents what is *visible* to the *camera*. It consists of *x*, *y*, *z values*. The *x* and *y* specify location of the *image plane* and it ranges from *-1,1* while *z* is *depth coordinate* that represents the distance or ranges from the camera. In order to convert from the *world coordinates* to *view coordinates*, a four by four (*4 x 4*) *coordinate transformation matrix* is applied, introducing the perspective effects of a camera.

The usual way to represent element in three dimension is through *cartesian vector* *x,y,z*. However, in order to project 2-D image to 3-D plane, vanishing point must be included in the projection, hence homogenous coordinate system is needed. Unlike *cartesian vector* with three (3) elements *x*, *y*, *z*, homogenous coordinate has four



PTTA UTM
PUSAT TEKNOLOGI DAN TATA KELOLA

elements vector represented as X, Y, Z, W , as earlier explained in the previous section.

The conversion from Cartesian coordinates to homogeneous coordinates is as follows:

$$x = \frac{X}{W} \quad (8)$$

$$y = \frac{Y}{W} \quad (9)$$

$$z = \frac{Z}{W} \quad (10)$$

Four by four (4×4) matrix is used for the performance of translation, scaling and rotation through repeated multiplication of matrix. We can create a transformation of matrix that translates a point x, y, z in Cartesian space by vector t_x, t_y, t_z as in equation (11). Figure 3.6 illustrates translation.

$$T_T = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (11)$$

where T_T is the matrix for translation

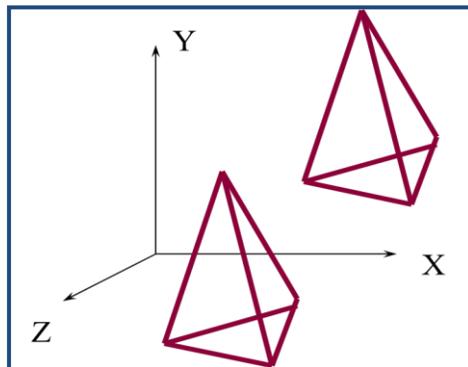


Figure 3.6: Translation

The created translated matrix needs to be post-multiplied with homogenous coordinate X, Y, Z, W . Meanwhile, we have to construct the homogeneous coordinate from the Cartesian coordinate before such multiplication, hence if we set $W=1$ representing *finite point*, X, Y, Z will yield $X, Y, Z, 1$. In the same vein, we pre-multiply the current position by the transformation matrix T_T in order to determine the translated point X', Y', Z' for yielding the translated coordinate. Hence, we have equation (12).

$$\begin{bmatrix} x' \\ y' \\ z' \\ w' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad (12)$$

Using the general pattern of conversion back to Cartesian coordinates as in equation (8, 9 and 10), we have:

$$x' = x + t_x \quad (13)$$

$$y' = y + t_y \quad (14)$$

$$z' = z + t_z \quad (15)$$

Equation 13, 14 and 15 are the procedure to translate an object. Similar procedure can be employed for scaling or rotating of an object. Using the transformation matrix as equation (16) where T_s is the transformation matrix for scaling, s_x, s_y, s_z representing the scale factors along x, y, z axes respectively. Figure 3.7 illustrates scaling about the origin.

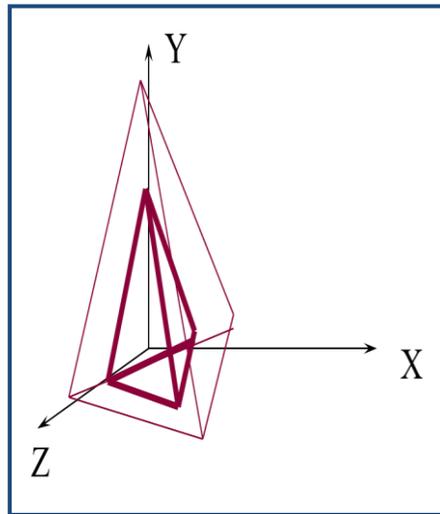
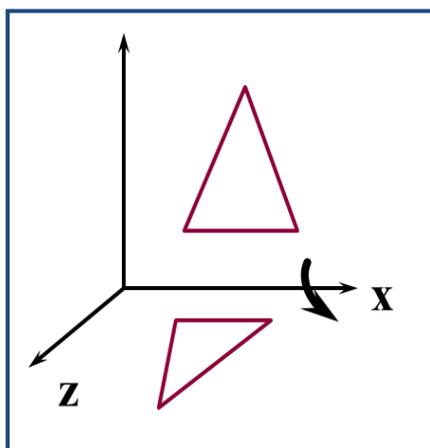


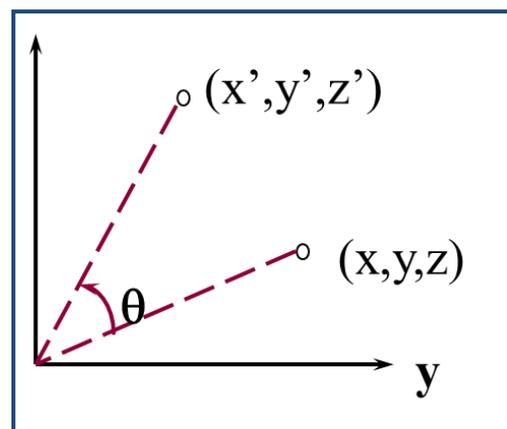
Figure 3.7: Scaling about the Origin

$$T_s = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (16)$$

In the same vein, we can do rotation around x , y and z axes by angle θ as illustrated in Figure 3.8, Figure 3.9 and Figure 3.10 to produce T_{R_x} , T_{R_y} and T_{R_z} respectively.



(a)



(b)

Figure 3.8: Rotation about x-axis

$$R_x(\theta): \begin{aligned} y' &= y \cos \theta - z \sin \theta \\ z' &= y \sin \theta + z \cos \theta \\ x' &= x \end{aligned}$$

$$T_{R_x} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta & 0 \\ 0 & \sin \theta & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (17)$$

Illustration of rotation about y-axis is given in Figure 3.9.

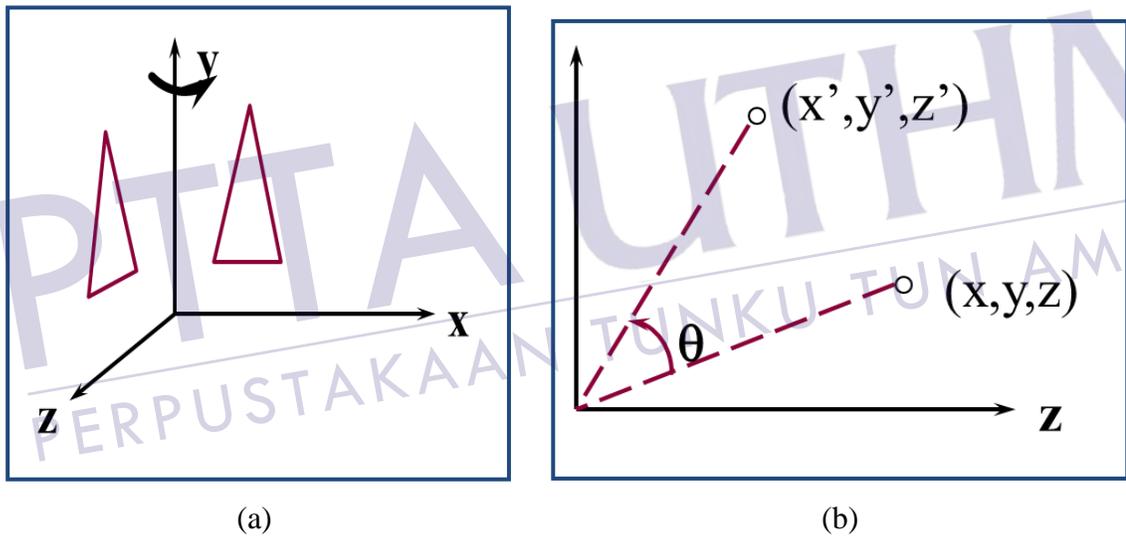


Figure 3.9: Rotation about y-axis

$$R_y(\theta): \begin{aligned} z' &= z \cos \theta - x \sin \theta \\ x' &= z \sin \theta + x \cos \theta \\ y' &= y \end{aligned}$$

$$T_{R_y} = \begin{bmatrix} \cos \theta & 0 & \sin \theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \theta & 0 & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (18)$$

Similarly, Figure 3.10 illustrates rotation about z-axis producing T_{R_z}

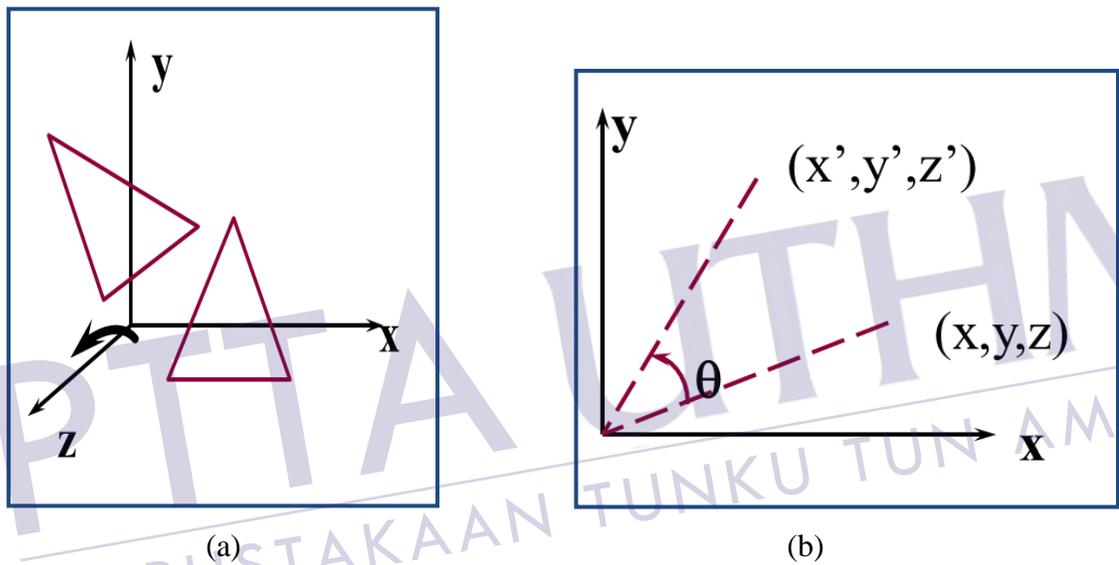


Figure 3.10: Rotation about z- axis

$$\begin{aligned} x &= x' \cos \theta - y' \sin \theta \\ R_z(\theta): \quad y &= x' \sin \theta + y' \cos \theta \\ z &= z' \end{aligned}$$

$$T_{R_z} = \begin{bmatrix} \cos \theta & -\sin \theta & 0 & 0 \\ \sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (19)$$

However, during the rotation of the object, we might need to transform the object from one coordinate axes to another, from $x - y - z$ to $x' - y' - z'$. In order to do this, we need to first derive a transformation matrix by assuming the following:

1. the unit x' axis makes the angle $\theta_{x'x}, \theta_{x'y}, \theta_{x'z}$ around $x - y - z$ axis
2. the unit y' axis makes the angle $\theta_{y'x}, \theta_{y'y}, \theta_{y'z}$ around $x - y - z$ axis
3. the unit z' axis makes the angle $\theta_{z'x}, \theta_{z'y}, \theta_{z'z}$ around $x - y - z$ axis

Where $(\theta_{x'x}, \theta_{x'y}, \theta_{x'z}), (\theta_{y'x}, \theta_{y'y}, \theta_{y'z})$ and $(\theta_{z'x}, \theta_{z'y}, \theta_{z'z})$ are the directional cosines

Hence, placing the directional cosines along the rows of the transformation matrix will produce equation (20) which is referred to as the resulting rotation matrix T_R .

$$T_R = \begin{bmatrix} \cos \theta_{x'x} & \cos \theta_{x'y} & \cos \theta_{x'z} & 0 \\ \cos \theta_{y'x} & \cos \theta_{y'y} & \cos \theta_{y'z} & 0 \\ \cos \theta_{z'x} & \cos \theta_{z'y} & \cos \theta_{z'z} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (20)$$

To rotate around the *center of the object*, which is usually more convenient, we must first *translate* from the *center of the object* to the *origin*, then we apply *rotations* followed by *translating the object back to its center*. However, in order to achieve the *translation, rotation* and *scaling* of the object using the transformation matrix, the order of the multiplication is important.

In *display coordinate system*, the coordinates are actual x, y pixel locations on the image plane. Though display coordinate uses the same basis as view coordinates except it does not use $-1, 1$ range. The *view coordinates* determine *window size* and *view point*. *Display coordinates* determine how the negative one to one $(-1, 1)$ of *view coordinates* is *mapped* into *pixel locations of display*. With view port, it is possible to divide the port which ranges from $0, 1$ for x and y axes and depth value representation with z axis. This is particularly useful in cases where one need to render two different scenes but display them in the same window.

3.3.1.3 Intensity Matching

It is paramount for *SurLens* to recognize data in their specific discrete nature, topological dimensions and adaptation of such data to its specific usage based on its

1. Begin
2. Read R_p
3. Run file Check // checking for rectilinear grid data formatting structure
4. If *foundStack*
5. Check format of Stack
6. Goto Step 12
7. Else if *found Slice*
8. Search and Combine other slice patterns
9. Trigger *ImageData*
10. Scale into *Extent* and *Origin*
11. Scale into *Spacing* and *Scalar* data types
12. Match *Intensity Values* into *Array Values*
13. Structure to *projective planetary*
14. Adapt into *Model, World, View* and *Display* coordinates
15. Repeat Step 2 until the end of R_p
16. Store output in J
17. End

Figure 3.11: Algorithm 1: *SurLens* Algorithm for Dataset Pre-Processing

architectural design. Those are essential procedures to map data into graphics primitives, which will ensure fast interactive display of data and thorough external data conversion into internal visualization data structures. *SurLens* was developed to combine sets of 2-D slices for visualization procedures and create access to intensity values using scalar array based on Table 3.2 values.

Table 3.2: *SurLens* Access Design to Intensity

Intensities Values	Scalar Values
Point 0	0, 0, 0
Point 1	1, 0, 0
Point 10	0, 1, 0
Point 100	0, 0, 1

Figure 3.11 and 3.12 present both the algorithm for *SurLens Pre-Processing* and for *Accelerating Hardware* respectively.

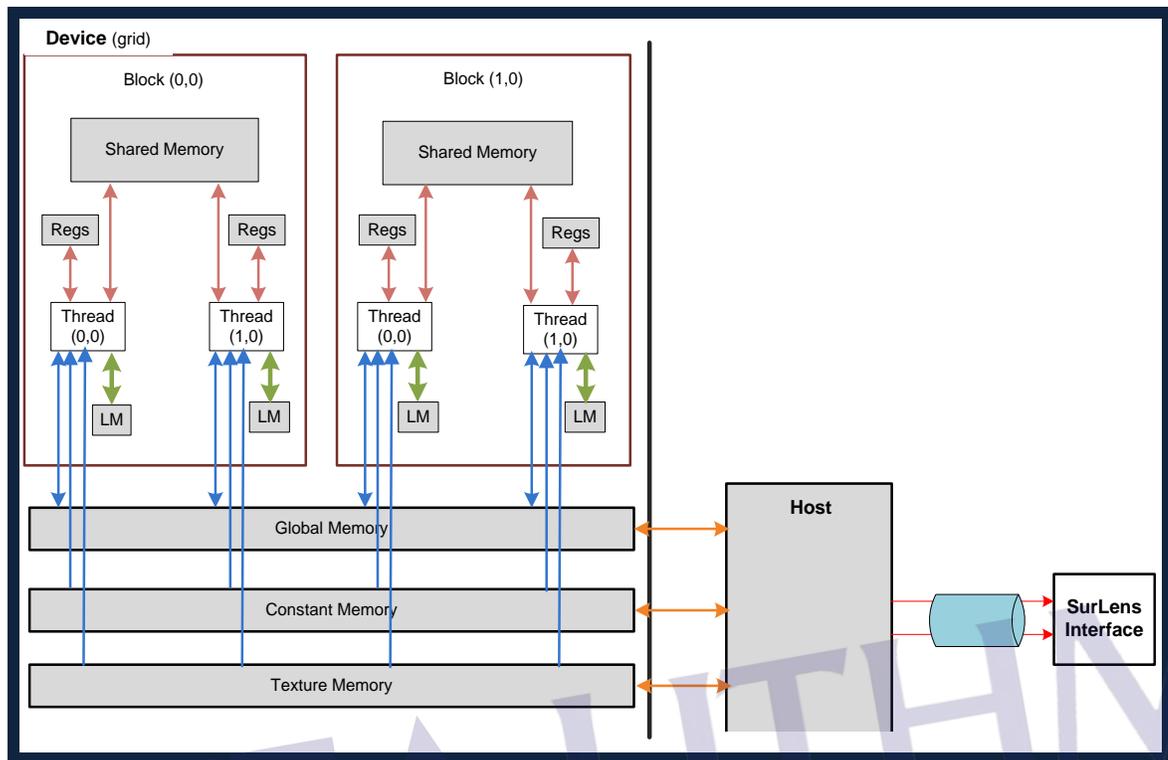
1. Begin
2. Read Output of J
3. Call nvcc
4. Create pointers to host and device arrays
5. Set bands for each array
6. Allocate array to each CUDA host
7. Release J to nvcc
8. Accept threads from nvcc
9. Copy all blocks to shared memory // for efficient computational calculations
10. If block \leq 1kb
11. Forward blocks to Q
12. Repeat Step 1 until end of J
13. Else fragment blocks into orthogonal slices
14. Push output file to Q
15. Repeat Step 2 until end of J
16. End

Figure 3.12: Algorithm 2: *SurLens* Algorithm for Accelerating Hardware

3.3.2 Accelerating Hardware

SurLens leverages the parallel architectural capabilities of *CUDA* technology in its data throughput. With *CUDA* architectural arrangements, the vertex and fragment segments share the same programmable hardware unlike the olden design of the Cg (C for Graphics), HLSL (High Level Shader Language) and GLSL (OpenGL Shading Language). The CPU and the main memory in *CUDA* are both referred to as *the host* while the GPU component is called *the device*. However, programmers have the choice of shifting all the computational processing tasks to only the device. The GPU is considered a *co-processor* that can execute multiple threads in parallel. However, with *CUDA*, certain portion of the implementation codes can be exclusively run on *device*. Outputs from *SurLens* dataset pre-processing phase are channeled to the integrated *CUDA* accelerating hardware for further processing. At this phase, the output datasets are fragmented into blocks which in certain cases, depending on the size of the data, might have been sufficiently processed enough for *SurLens* graphic execution phase, else, the resulting blocks would have to be re-fragmented into orthogonal slices for better data throughput in the system. Figure 3.15 shows *SurLens* Memory System Architecture.

In *CUDA* architecture, the device has a set of multi-core processors each of which is referred to as *Multiprocessor*. However, with different access time, all the multiprocessors can access the three common memories, *the device memory*, with their cores working in a *Single Instruction, Multiple Data (SIMD)* fashion. The *CUDA* program is in the host consisting of one sequential thread running on the host CPU alongside with several parallel kernels executed on the parallel processing device, the GPU. Each of the kernels runs a set of parallel threads, the execution of a scalar sequential program. The program generates a grid of thread blocks from the program. The 3-D data are projected using the earlier introduced concept of homography. *SurLens-CUDA* architecture is illustrated in Figure 3.13.



NOTE: Regs = Register, LM = Linear Memory

Figure 3.13: *SurLens*-CUDA Architecture

The fact that the *SurLens* hardware components, CUDA, is in *C programming* model and the development of the entire *SurLens* software is in *C#* makes the interfacing of *SurLens* hardware and software component a critical task during this study. Using Microsoft Visual Studio 2008, CUDA C programming language was interfaced with VC++, which is in turn interfaced with C# .NET platform for *SurLens* interoperability. With the *System.Runtime.InteropServices* namespace, *SurLens* can seamlessly interoperate by invoking platform services to unleash the hardware-acceleration capability of CUDA. Through *System.Runtime.InteropServices* namespace, the *DllImportAttribute*, which is required to define platform invoke methods for accessing unmanaged APIs, and the *MarshalAsAttribute*, for specifying how data is to be marshaled between managed and unmanaged memory are utilized. This works with *XMLImageDataReader* class within *SurLens* Memory System Architecture. *XMLImageDataReader* is likewise utilized in the preprocessing phase for data

conversion to the *SurLens* standard usable *ImageData* format (.vti). The information sets up by the reader is initially saved in *SetupOutputInformation*, and later copied to *outInfo*. The predefined procedure is specifically tailored to port.

In leveraging CUDA accelerating capability for *SurLens* Visualization, the first step was the configuration and enabling of *nvcc* in the GPU-enabled experimental testbeds used. *CUDA Software Development Kit* and *CUDA ToolKit version 4.1.28, Window 64-bits* were used for the integration. During such integration, two methods of CUDA integration were considered for the implementation using Visual Studio project.

1. *Method 1*: Specific usage of “*template*” project from CUDA Software Development Kit. This requires setting a custom build step for each .cu file and calling *nvcc* from the command line.
2. *Method 2*: Using other projects from CUDA Software Development Kit. This requires importing the *Cuda.rules* files into the project after which the project’s .cu files will be built.

Having a greater control over *nvcc* and the environment in general is more useful to this development, which is the gained advantage for using *method 1* though with the cost of typical settings for each .cu file. With *method 2*, no such specific settings and likewise no control over the *nvcc*. Hence, *method 1* is applied for the *SurLens-CUDA* integration. CUDAVS2008 class was created for interaction with localized resources associated.

3.3.2.1 Data Structuring & Fragmentation

One of the greatest challenges in medical visualization is the issue of data structuring and fragmentation. This is as a result of the usual bulkiness of medical datasets, which apparently determines the processing speed of the volume visualization frameworks. Data structuring and fragmentation are carefully handled within the *SurLens* memory

system architecture. Data fragmentation procedures in *SurLens* is illustrated in Figure 3.14.

SurLens uses uniform rectilinear grid referred to as *ImageData* as its basic class of storing images. Figure 2.2 shows the structure of uniform rectilinear grid. The term uniform rectilinear grid is used to describe cells of the same type. The dimensionality of the dataset is used in determining such grid type, which can be *0-Dimension* for vertex, *1-Dimension* for line, *2-Dimension* in case of pixel representation and *3-Dimension* for voxel. *SurLens* pre-processing stage define four (4) key elements for its data storage:

1. Extent: This is the size of the image to be stored. This element is used to define the minimum and the maximum indices in each direction,
2. Origin: The ready-to-use positioning of the image in 3-D space of a point defines with indices (0,0,0),
3. Spacing: This is the dimension of the voxel. This is the distance between each point, which varies with different acquisition devices,
4. Scalar Type: This is the image type such as float or short.

A typical example for data structuring elements in *SurLens* could be seen as a circumstance whereby an image data with extents (0,19), (0,29), (0,39), with origin (0.0, 0.0, 0.0), spacing (1.0, 1.0, 1.0) of scalar type unsigned short. This implies that the image has 20 points in the x-direction, 30 points in the y-direction and 40 points in the z-direction, making up a total number of points $20 * 30 * 40$.

However, Magnetic Resonance Imaging (MRI) data is scalar data. The topology and points coordinates of such data are defined implicitly. However, the defining of such mesh for a ready-to-use image data in *SurLens* follows previously highlighted four (4) key elements. The coordination of each point is defined as follows:

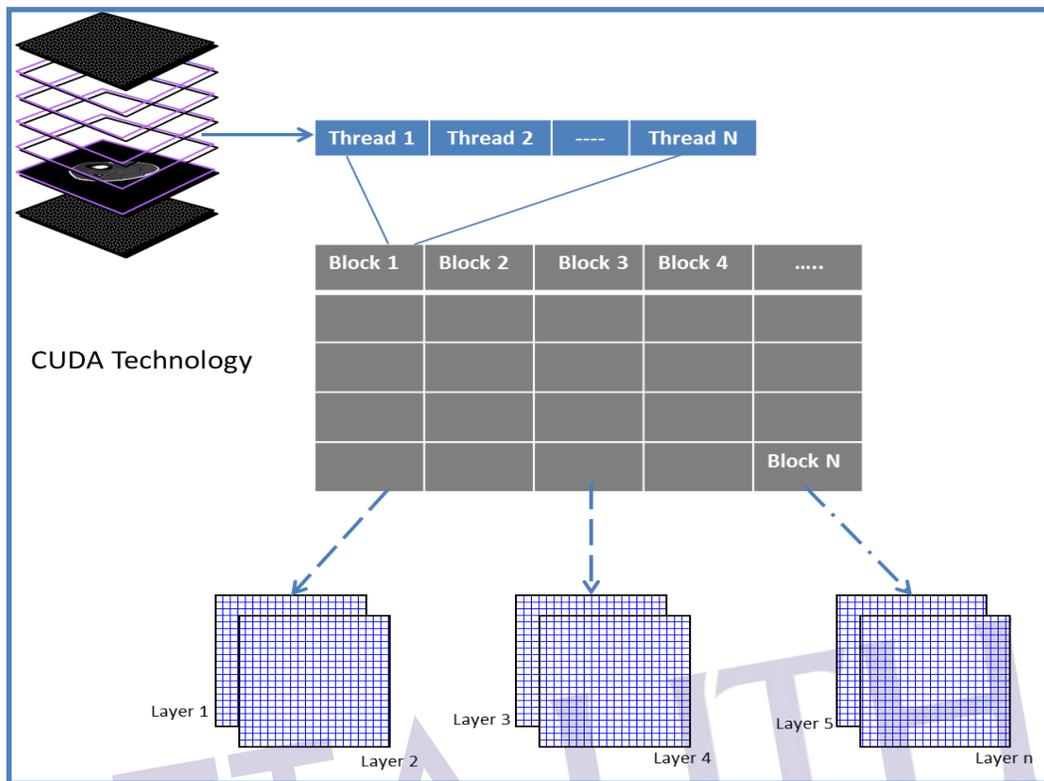


Figure 3.14: *SurLens* Data Fragmentation Procedures

Coordinate = origin + index * spacing (coordinate, origin, index and spacing are vectors of length 3)

However, *SurLens* uses flat index for storing of the dataset, hence, it converts i, j, k index to its corresponding flat structure index as follows:

$$\text{idx_flat} = k * (\text{npts_x} * \text{npts_y}) + j * \text{nptr_x} + i$$

The coordinates of each point is represented as i, j, k . Meanwhile, with the three (3) arrays in the three directions x, y, z of lengths $\text{npts_x}, \text{npts_y}$ and npts_z , the i, j, k index of the dataset can be converted to the flat index through the above equation.

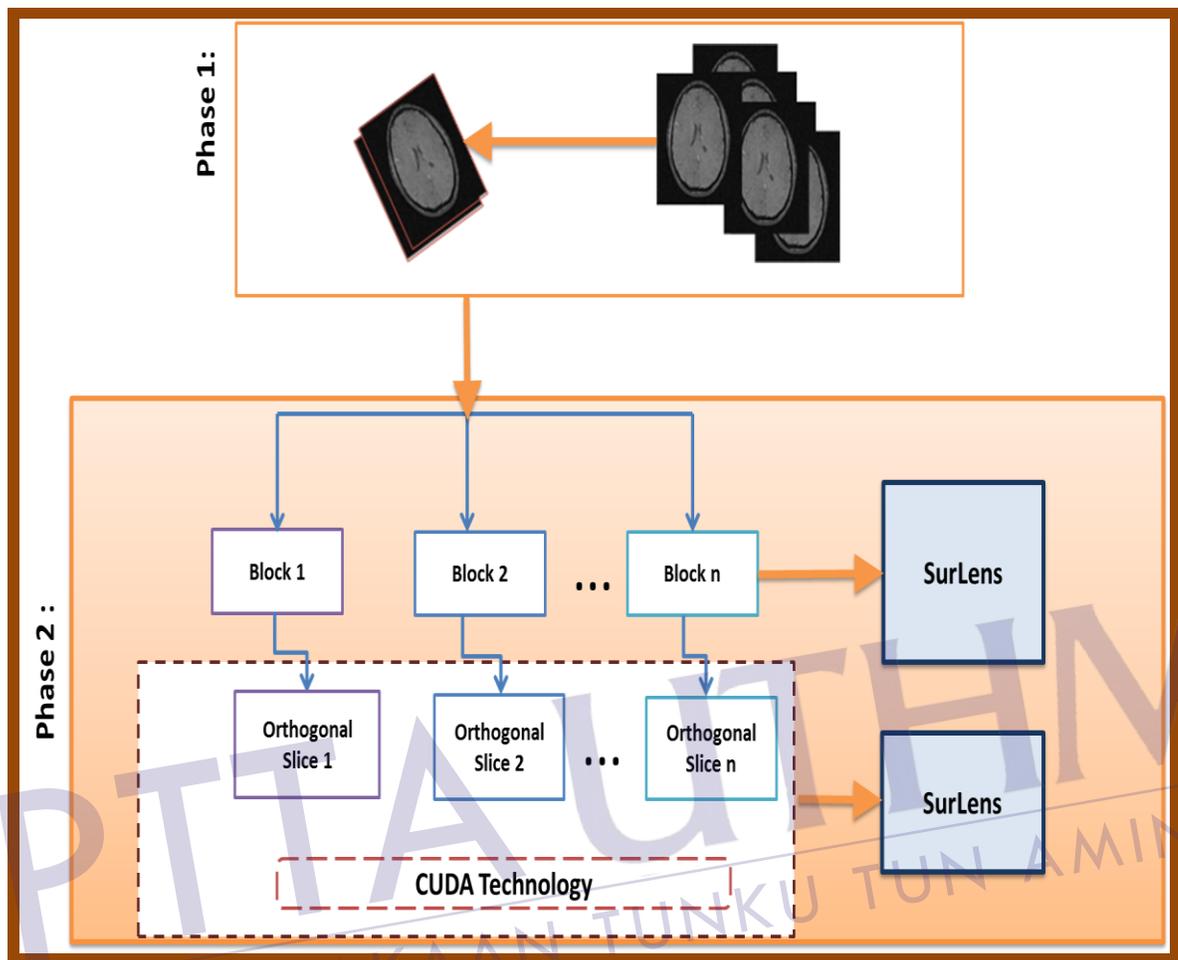


Figure 3.15: *SurLens* Memory System Architecture

As the focus of this study is to obtain quality 3-D, sufficient enough to reveal internal information of datasets within considerable interactive speed, the study considers *ImageData* for *SurLens* data structuring. The Image Data structuring is considered because it is regular in nature, it requires less storage than other datasets type hence *SurLens* algorithm could possibly be optimized to take advantage of such feature.

3.3.3 Graphic Execution Phase

Software components developed for this study consists of Graphic Execution and Volume Rendering. *SurLens* graphic execution phase is to transform data into *graphic primitives* in form of *geometric file output*. The first stage in this phase is to prepare data for visualization, the *data analysis stage*. This is predominantly attributed to computer interaction, user intervention is limited. Necessary *filters* for smoothing of the data are applied at this stage. Values in the data are properly structured to prevent errors in the data processing. The prepared data are filtered. However, *SurLens* is designed to be able to select certain needed data portion for visualization, in certain cases, which it does with the filtering stage. At the filtering stage, if the data coming from *SurLens Memory Architecture* requires more attention, the filter(s) will modify the data further, carry out *conversion, reduction and merging* hence a more *focus data* will be available for mapping. The *focus data* are mapped to *geometric data*. Features like *points, lines, colour, position and size* attributes are made conspicuous for data expressiveness. *Mappers* serve as input in data preparation for ready-to-use *file output, the actors*. Figure 3.16 and Figure 3.17 present *SurLens Graphic Execution Phase* and its proposed algorithms respectively.

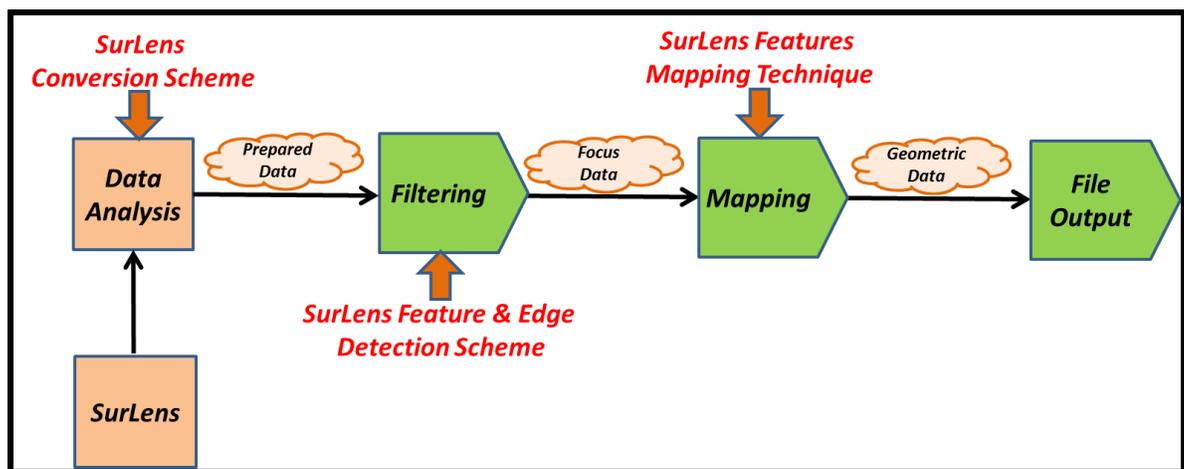


Figure 3.16: *SurLens* Graphic Execution Phase (Phase 3)

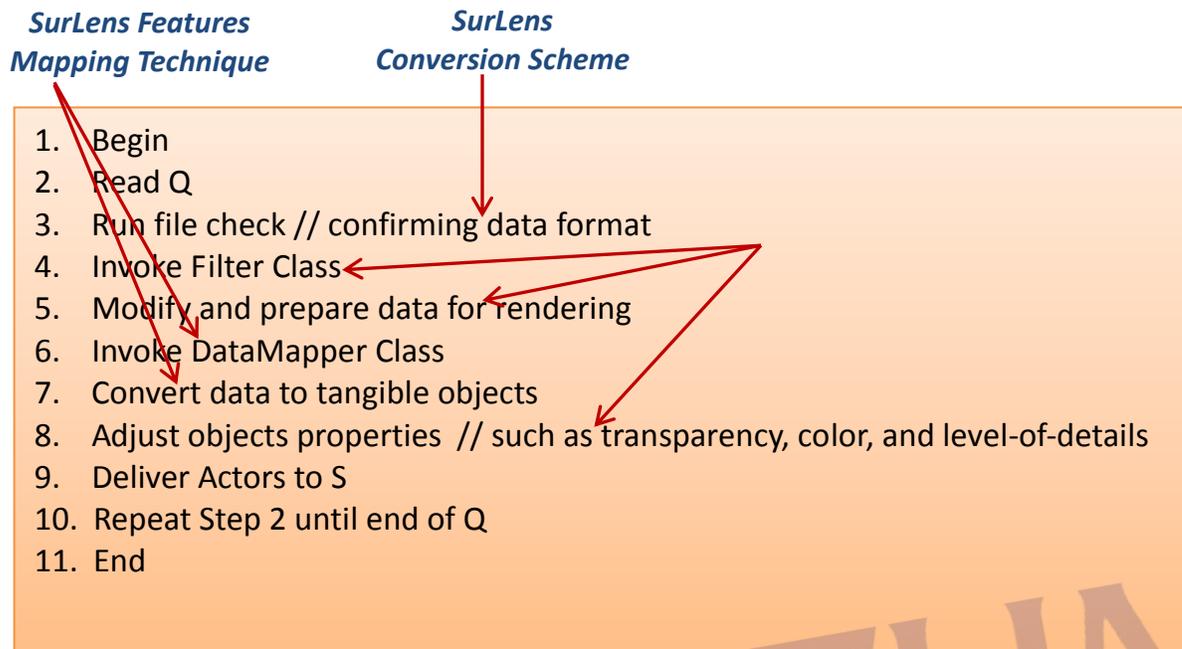


Figure 3.17: Algorithm 3: *SurLens* Algorithm for Graphic Execution

3.3.4 Volume Rendering Phase

Volume Rendering phase is the second phase of the *SurLens* software component denoting the concluding phase, producing the final output in the *SurLens* visualization framework. *SurLens* implemented ray casting technique with C# using Visualization ToolKit (VTK) and Microsoft visual Studio 2008.

The image viewed by the *vtk Camera* needs to be recorded with n pointing towards the camera, the developed algorithms compute $R(x.n.v)$ for all the focussed points. Photon number density and radiance are considered for such computation. Figure 3.18 shows *SurLens Volume Rendering Phase*.

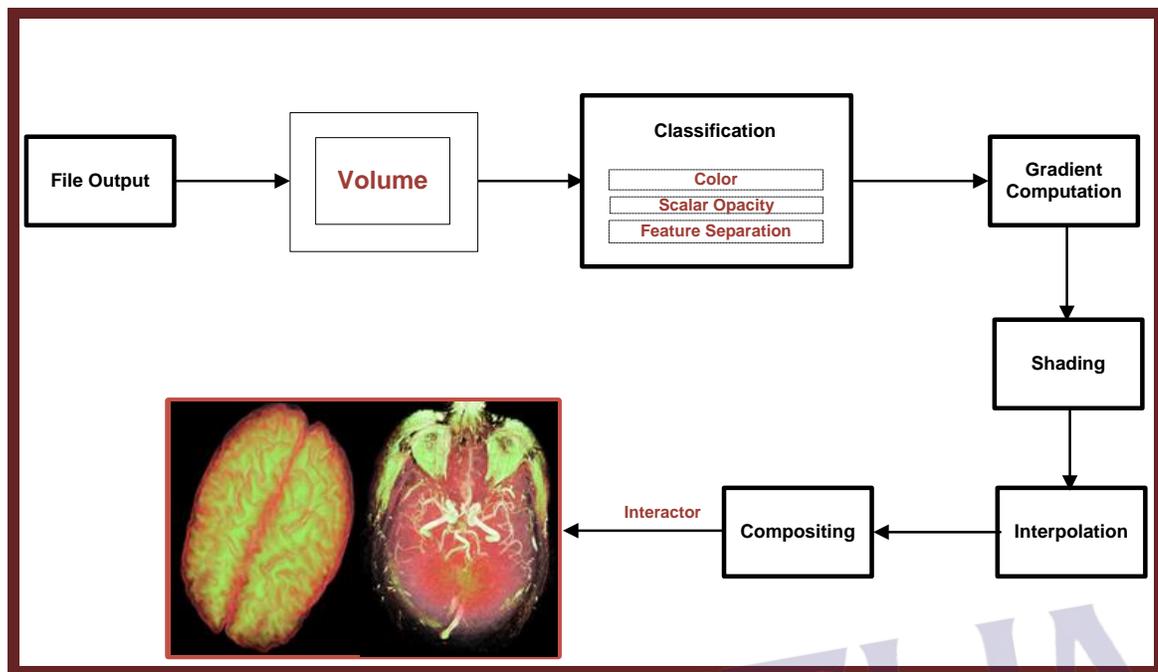


Figure 3.18: *SurLens* Volume Rendering Phase (Phase 4)

3.3.4.1 Camera Model

The virtual viewing and displaying of 2-D images in volume rendering requires modelling and simulating based on some particular established laws. The principle of the famous OpenCv's face detector that uses Viola & Jones approach for detecting objects in images were initially considered for this task. OpenCv's face detector consists of four combined concepts; the haar features (simple rectangular features), an integral image for rapid feature detection, the AdaBoost machine-learning method and a cascaded classifier. Though this approach has proven very successful in facial detection, it's not resourceful in volume rendering. Hence, *SurLens* camera modelling approach is based on the description of physical laws of optics. Simulating such process in the real world scenario requires calculating the radiation field. However, the wave character of light, possible light polarization states, diffraction and interference are neglected while trying to model transport theory of light hence the concentration is only on geometrical optics lights.

Defining radiant energy using radiance $R(x.n.v)$ where x is the radiant field at any point, n is the direction of the radiant energy and around v . Hence the radiant energy δE travelling in time dt within a specified frequency interval dv around v through a solid angle $d\Omega$ in direction n is given by equation (21); where ϑ is the angle between n and the normal on da .

$$\delta E = R(x.n.v) \cos \vartheta da d\Omega dv dt \quad (21)$$

In defining radiant energy using photon number density $\psi(x.n.v)$, we can represent the number of photon \check{N} per unit volume in equation (22), with x representing the position of the photons per unit volume, having frequency interval dv around v which travels in a direction n into an element of solid angle $d\Omega$.

$$\check{N} = \psi(x.n.v) d\Omega dv \quad (22)$$

However, we can also calculate the number of photons \check{N} passing through a surface da with time dt and travelling velocity c .

$$\check{N} = \psi(\cos \vartheta da)(cdt)(d\Omega dv) \quad (23)$$

Considering the energy being carried by each photon as hv where h is the planck's constant, the radiant energy using photon number density could be seen as equation (24).

$$\delta E = chv\psi(x.n.v)(\cos \vartheta da d\Omega dv dt) \quad (24)$$

Equating (21) and (24)

$$R(x.n.v) \cos \vartheta da d\Omega dv dt = chv\psi(x.n.v)(\cos \vartheta da d\Omega dv dt)$$

The resulting equation (25) proves similarity between *radiance* and *photon number density*.

$$R(x.n.v) = chv\psi(x.n.v) \quad (25)$$

Hence computing $R(x.n.v)$ for all the focussed points will record all the image viewed by the camera in the algorithm.

3.3.4.2 Volume Classification

Classification is the labelling of the data regions with *colour* and *opacity*. Such assignment of colour and transparency to volume regions is handled by transfer function in volume rendering. Specifically, *SurLens* technique involves mapping of scalar values through a lookup table to obtain a colour and in order to modify the appearance of the *points* or *cells*, the colour is applied during rendering.

The developed algorithms uses *RGBA 3-D* vector dataset, with *R*, *G*, *B* representing the colour spaces designed for *CRT* devices while “*A*” represents the *opacity*. When the opacity value is turned to value *0*, it means *total transparency* while value *1* represents *total opaque*. The mapping of data to different opacity values aids in the classification of the area of interest in the visualization of data and also enabling the users to have the opportunity of seeing the “*interior of the data*” which varies in the data densities and colours.

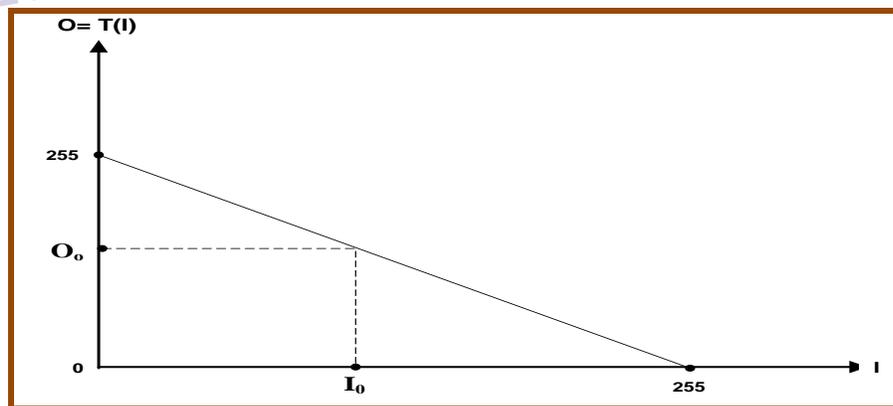


Figure 3.19: Image Point Processing Approach

SurLens applies *point processing* image enhancement techniques, which is based on the intensity of individual pixels in the image. Hence, based on equation (26), intensity transfer function could be represented through *255 Output Pixel* and *255 Input Pixel* as in Figure 3.19.

$$O = T(I) \quad (26)$$

Where *O* represents the *Output Pixel*, *T* is the *transform* and *I* is the *Input Pixel*.

Feature enhancement is extremely important in order to distinguish normal tissues distinctly from abnormal tissues especially when intensities of abnormal tissues match with the intensities of normal ones. Despite the fact that brain tumor might sometimes be large, space occupying, it could still exist in the same intensity as the normal tissues making it difficult to distinguish.

Transfer function was utilized in mapping data value to “renderable quantities” as the output value. The two (2) main transfer functions designed for this study are the *Opacity Transfer Function* and the *Colour Transfer Function*. The *Opacity Transfer Function* maps intensities of volume elements (voxels) in the data sample to the corresponding opacity value based on the *SurLens* intensity scale and selectively make some voxels transparent enough to be seen through the assigned opacity value in order to show the interior of the data sample. Meanwhile, *Colour Transfer Function* uses colouration for its classification procedures. It maps intensities of voxels to corresponding colour values using lookup table and likewise do selective painting of voxels with different colours such that voxels of different intensity values are presented with appropriate corresponding colour variances. However, in order to have better clarities of the output images, contrast-enhancement transfer function referred to as the *Contrast Transfer Function (CTF)*, is applied in the *SurLens* 3-D reconstruction.

3.3.4.3 Shading and Gradient Computation

The three (3) parameters modeled for illumination are *the ambient coefficient*, *the diffuse coefficient* and *the specular coefficient*. The ambient is the background illumination; the specular is the bright and shiny reflections while the diffuse is the non-shiny, illumination and shadows. Ambient lighting is represented as equation (27).

$$R_c = L_c O_c \quad (27)$$

Where R_c is the resulting intensity curve, L_c is the light intensity curve and O_c is the colour curve of object.

Ambient light has no direction, is independent of light position, orientation of the object and observer's position. With this in mind, ambient is simply seen as the approximate contributions of light to the scene which is irrespective of the location of object and light. Figure 3.20 illustrates this.

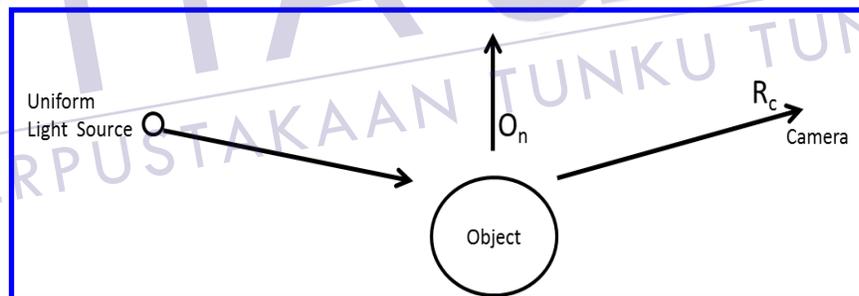


Figure 3.20: Ambient Lighting

Diffuse lighting, which has no dependence on camera angle, is represented as equation (28) and illustrated in Figure 3.21. In order to determine diffuse's contribution to the surface, surface normal and the direction of the incoming rays are important.

$$R_c = L_c O_c \text{Cos}(\Theta) \quad (28)$$

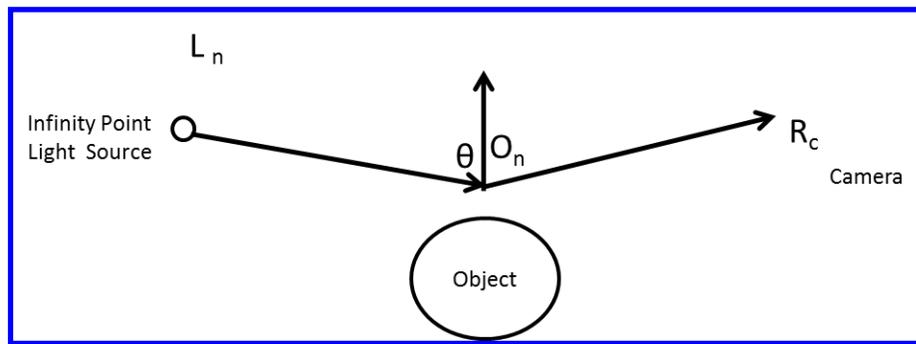


Figure 3.21: Diffuse Lighting

Where L_c is the light colour, O_c is the object colour and $\text{Cos}\theta$ is the product of the vector of light source (a negative value) and the vector of surface normal value to the object.

Specular lighting, which has no dependence on object colour, is represented as equation (29). Figure 3.22 illustrates specular lighting which could assist in calculating specular coefficient.

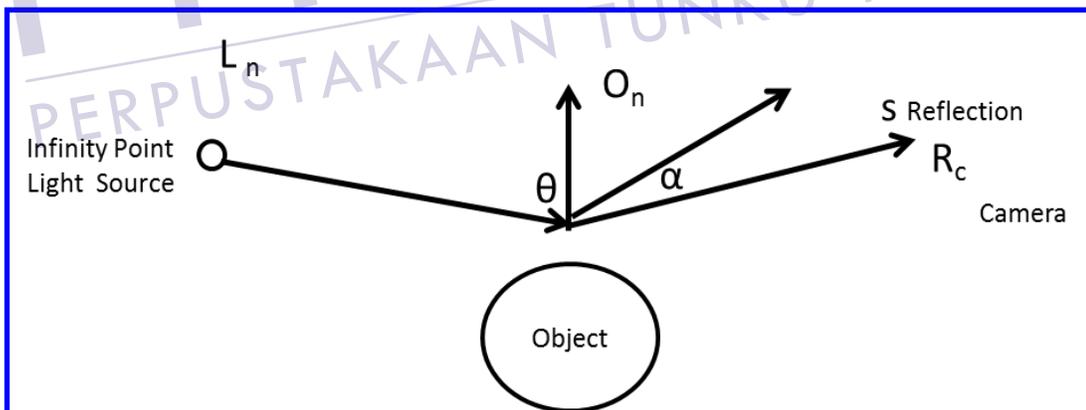


Figure 3.22: Specular Lighting

$$R_c = L_c K_s \text{Cos}(\alpha)^n \quad (29)$$

L_c represents the light colour, K_s is the reflection constant, R_c is *Colour curve*, $Cosa$ is the product of the vector of light source (a negative value) and the vector of surface normal value to the object and n is the specular power since specular light could have different n values. Hence, we can combine the three (3) lighting models to give the equation in equation (30).

$$R_c = W_a(\text{ambient}) + W_d(\text{diffuse}) + W_s(\text{Specular}) \quad (30)$$

where W_a , W_d and W_s are the relative weights of *ambient*, *diffuse* and *specular* respectively.

In line with the focus of *SurLens Visualization System*, to achieve quality image output, *SurLens* framework is configured to choose either *Flat*, *Gourand*, *Phong shading* or their combination for better shading of images with respect to the level of pixels in the datasets. Flat shading is the earliest shading method which requires shading the polygons in the data samples with single colour. However, because sometimes resulting interpolation colour could be needed during shading to have a better image colouration, Gouraud shading was introduced. With Gouraud shading, polygons are shaded by interpolating colour that are computed at the vertices of the image. Unfortunately, Gouraud shading usually produces “*specular highlights*”, a bright spot of light that appears on shining objects when illuminated. Phong shading produces better shading results compared to Gouraud shading by fixing the issue of specular highlights. However, despite the shortcomings in Flat and Gouraud shading, using all in combination will contribute to obtaining better-shaded image.

3.3.4.4 Interpolation / Resampling

Interpolation is very important for filtering or smoothing of the resulting images. *Weights W* are applied to the set of samples. The interpolation constitutes a linear interpolation of data values at each points of the cell. This is illustrated with the interior cell location d in equation (31).

$$d = \sum_{i=1}^{n-1} W_i * d_i \quad (31)$$

Where d is the value at interior cell location, (a,b,c) , d_i is the data volume at the i^{th} cell point and W_i is a weight at the i^{th} cell point. However, the interpolation weights are functions of the parametric coordinates $W_i(a,b,c)$. Meanwhile, we want $d = d_i$ whenever there is a coincidence between the interior point P_i and a cell point P , we can then define weight W_i , as equation (32). The interpolation functions are of a characteristic shape. They reach their maximum value $W_i = 1$ at cell point P_i and are zero at all other points.

$$W_i = 1, W_j = 0 \text{ when } P=P_i \text{ and } i \neq j \quad (32)$$

In addition, we do not want the interpolated value d to be smaller than the minimum d_i and also not larger than the maximum d_i , we therefore define weight to satisfy equation (33).

$$\sum W_i = 1, 0 \leq W_i \leq 1 \quad (33)$$

SurLens uses *trilinear interpolation* approach. This was considered simply because trilinear interpolation usually produces smoother images with less artifacts, though trilinear interpolation may possibly be a bit longer in implementation compared to nearest neighbour but we augmented this with our CUDA memory architecture.

3.3.4.5 Compositing

The compositing system is responsible for summing up colour and opacity contributions from re-sampled locations along a ray into a final pixel colour for display (Porter & Duff, 1984). Some of the factors to be considered during compositing are the *absorption*, *emission* and *scattering* based on the principle of transportation of light. However, for the feasibility of volume rendering in the visualization domain, many authors ignored modelling of the scattering of light. One of the highly referenced authors is Sabella. Sabella (1988) introduced the *emission-absorption model* or *density-emitter model*, which ignored scattering of light. Hence, during the compositing, while solving

the emission-absorption, this study assumes absorption and emission occur with no scattering.

In the implementation of the compositing function for this research, we assume a volume of density $\rho(x,y,z)$ and Ray R coming from the back of the volume $S=0$ and to the eye of the viewer at S_{eye} . The density is express as equation (34).

$$P(x(S), y(S), z(S)) = \rho(S) \quad (34)$$

where S denotes distance

$e^{-\tau(S_o, S_{eye})}$ is the absorption

On a normal circumstance, we have density distributed inside the volume and this attenuated the light allowing only some in reaching the viewer at S_{eye} . If we assume to have ray path at points $S_o, S_1, S_2, \dots, S_n$, we equally assume there are emissions and absorptions but no scattering at each of the points, S_o to S_n , as illustrated in Figure 3.23.

At point S_o , we have initial intensity at I_0 , $I = Intensity$, $S = distance$. If this occurs without absorption, we have all the light reaching point S_{eye} . Meanwhile, with absorption along S_o to S_{eye} we have equation (35).

$$I(s) = I(S_o) e^{-\tau(S_o, S_{eye})} \quad (35)$$

At another point S_1 , we have similar occurrence of emission and absorption and then we can have the intensity measured at the eye position as equation (36).

$$I(S_{eye}) = I(S_o) e^{-\tau(S_o, S_{eye})} + I(S_1) e^{-\tau(S_1, S_{eye})} + I(S_2) e^{-\tau(S_2, S_{eye})} \quad (36)$$

If we consider a volume of two points S_1 and S_2 emitting light rays towards the viewer at S_{eye} , hence the integral of the light rays for the two points can be illustrated in Figure 3.23. This is also in line with the explanation of Hege, Höllerer & Stalling (1996) as being documented in the previous chapter. However, at every point along the ray path, we assume to have absorption and emission, hence, the total intensity that reaches the

eye is calculated by the sum of all the intensities attenuated in all the points as equation (37).

$$I(S) = \int_0^S I(\check{S}) e^{-\tau(\check{S}, S)} d\check{S} \quad (37)$$

Where S is the distance at eye (S_{eye}), \check{S} is the point in the ray path (i.e. S_0 to S_n) and τ is the extinction coefficient. Equation (37) does the compositing in ray casting technique.

Therefore, by ignoring the scattering based on the Sabella model, it means there could not be any mixing between frequencies hence there won't be ν . With these in mind, emission coefficient η and absorption coefficient χ would consist of only source terms, which could be represented as w and k respectively. However, based on Fig. 3.23 we have assumed rays path $S_0, S_1, S_2, \dots, S_n$ hence we can have equation (37) changed to equation (38).

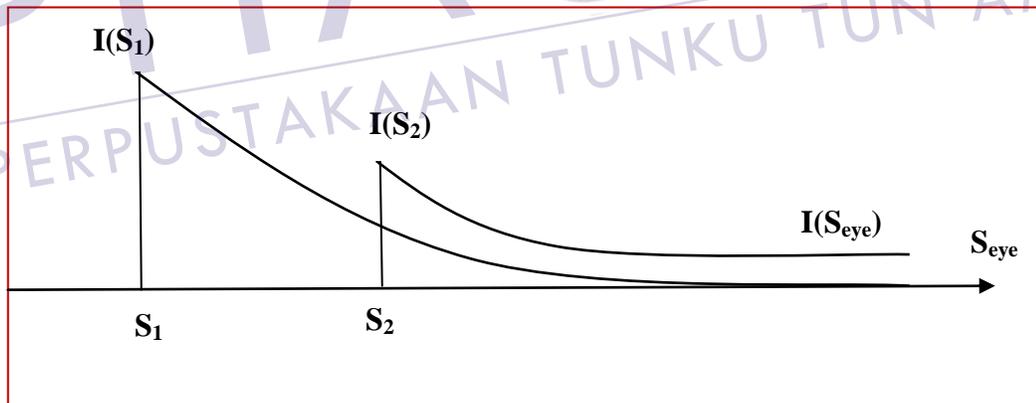


Figure 3.23: Absorption and Emission along Light Rays'

$$I(S_{eye}) = I(S_0) e^{-\tau(S_0, S_{eye})} + \int_{S_0}^{S_{eye}} w(s') e^{-\tau(S_0', S_{eye})} dS'_{eye} \quad (38)$$

We can also have equation 39.

$$\tau(S_1, S_2) = \int_{S_1}^{S_2} k(s) ds \quad (39)$$

If we apply discrete approximation of the ray density distribution, the specific intensity at point S_k is related to the specific intensity at point S_{k-1} , this leads us to equation (40).

$$I(S_k) = I(S_{k-1}) e^{-\tau(S_{k-1}, S_k)} + \int_{S_{k-1}}^{S_k} w(s) e^{-\tau(S, S_k)} dS \quad (40)$$

Hence we could represent θ_k , the transparency of the material between S_{k-1} to S_k in equation (41).

$$e^{-\tau(S_{k-1}, S_k)} = \theta_k \quad (41)$$

$\tau(S_{k-1}, S_k)$ is the optical depth and is represented as equation (42).

$$\tau(S_{k-1}, S_k) = \int_{S_1}^{S_2} \chi(S_k) dS \quad (42)$$

The following applies in the above expressions:

1. At infinitely large optical depth, $\tau(S_{k-1}, S_k)$, the transparency, $\theta_k = 0$.
2. At vanishing optical depth $\tau(S_{k-1}, S_k)$, the transparency, $\theta_k = 1$.
3. Opacity can be represented as $a_k = 1 - \theta_k$

Algorithms for *SurLens Volume Rendering*, the concluding phase in *SurLens Visualization framework* is presented in Figure 3.24.

1. Begin
2. Load and Display Actors from S
3. Create Class Interactor
4. Control mouse functions
5. Display rendered window with 3-D Volume
6. Create needed pipeline for objects
7. Read volume into the application
8. Go through the visualization pipeline
9. Set Color Curve
10. Set Opacity Curve
11. Set Gradient Curve and feature enhancement
12. Go through the created graphics pipeline
13. Display slices in render window
14. Fetch the name of the volume to load
15. Create needed pipeline objects
16. Read the image
17. Go through created pipeline
18. Visualization pipeline
19. Graphics pipeline
20. Update global variable
21. Adjust the camera and control the slice with trackball movement
22. Attach trackball movement with camera resetting
23. Clean global variable
24. Repeat Step 2 until end of S
25. End

Figure 3.24: Algorithm 4: *SurLens* Algorithm for Volume Rendering

3.4 Summary

SurLens Visualization System principally consists of *four (4) major phases*, the *Dataset Pre-Processing*, the *Accelerating Hardware*, the *Graphic Execution* and the *Volume Rendering Phases*. The *Dataset Pre-Processing* and the *Accelerating Hardware* make up *SurLens Memory System Architecture*. Preparation of datasets for rendering commenced with reading of the dataset from the *system repository* and corresponding conversion based on *SurLens* pre-defined *data storage format*. *Intensities*, *values matching* and adaptation of the data into *model*, *world*, *view* and *display coordinates*

systems are all introduced at the *SurLens Memory System Architecture*. *SurLens* leverages *CUDA parallel processing* capabilities in the *fragmentation* of the data into *blocks* in order to maximize the memory system architecture' throughput for *orthogonal viewing directions*.

SurLens Graphic Execution phase focuses on checking the forwarded data from the *Memory System Architecture* and making them available in *geometric format* for further processing within the framework. Volume *filtering operations* employed are dedicated to *data processing, image enhancements, feature detection* and *enhancement of detail internal structures* of the volume.

SurLens Volume Rendering phase predominantly focuses on rendering of the volume. Colours and transparencies are assigned to volume regions in order to aid in the classification of the area of interest in the visualization of data. Provisions are made during the classification for such cases in which abnormal tissues might have the same intensity values as normal tissues. Moreover, *SurLens* classification procedures also enable users to have the opportunity of seeing the “*interior of the data*” with varying data densities or intensities and colour. The *background illumination*, both *the bright and shiny reflections, non-shiny illumination* and *shadows* are defined in the *Shading and gradient computation* for further volume ray characterization. However, *SurLens* can utilize *Flat, Phong, Gourand* shading techniques or the combinations of the two or all of the techniques.

SurLens development follows *trilinear interpolation*. The desired qualities of *smoother images with less artefacts* of *trilinear interpolation* is prioritized and considered more preferable to its usual longer processing time, *SurLens Memory Architecture* augmented this with its CUDA hardware. Sabella density-emitter model is applied in the *SurLens compositing* procedures for summing up contributions of various *colours and opacities* from *re-sampled locations* along a ray into a *final pixel colour* for visual display. The next chapter presents implementation of *SurLens Visualization System*.



CHAPTER 4

IMPLEMENTATION AND TEST DATASETS

SurLens Visualization System is developed using *C-sharp (C#)* programming language. The development and testing was achieved with *Window 7 Operating System* integrated with *NVIDIA Quadro 600 CUDA 2GB* memory capacity as the main experimental testbed. Two (2) other experimental testbeds were set-up as control with *Window 7 Home Premium* integrated with *NVIDIA GeForce GT 520 M CUDA 1GB* and *Window XP Professional, CPU T4200, 2.0 GHz, 3.00 GB RAM*. Previous chapter presented the methodology used in the development of *SurLens* Visualization System. This chapter discusses *SurLens* implementation and the evaluation test datasets.

4.1 Introduction

Implementations of the developed algorithms are carried out in the system implementation along with the *SurLens'* experimental testbeds presented in Figure 3.3. Such implementation approach makes provision for studying the performance of *SurLens* Visualization framework in different implementation environments. According to *SurLens'* framework, as in Figure 3.2, there are four main processing steps involved in its implementation; the *dataset pre-processing*, the *accelerating hardware*, the *graphic execution* and the *volume rendering*.

The *SurLens* Data-preprocessing algorithm reads raw datasets from the repository and run file checking on them to confirm the dataset are in rectilinear grid formatting

structure. Medical datasets are usually in singly 2-D slices. Meanwhile, in certain cases, patient datasets might be saved in stack at the point of acquisition. However, we design our algorithm to check if the datasets are in stack or singly. If found in stack, the format of the stack is confirmed and the necessary matching of intensity values into corresponding array values are done. Projective planetary, adaptation of the datasets into model, world, view and display coordinates are triggered and executed by the algorithm. The final processed datasets are stored in the data outlet of *SurLens* Data Preprocessing Algorithm for further processing within the framework. This sequence of operation is repeated until all the datasets in the repository are processed. Meanwhile in cases where the datasets are singly, the algorithm search and combine all the slice patterns and then trigger the *ImageData* class which facilitates scaling of the datasets into extent, origin, spacing and scalar data types. At this point, the processed datasets are in stack hence all the processing procedures for the stack datasets can be executed as previously being discussed.

SurLens Accelerating Hardware Algorithm reads processed data deposited by the *SurLens* Data-preprocessing algorithm. The algorithm is interfaced with CUDA, hence it calls nvidia cuda compiler (nvcc) and then launch the creation of pointers to the host and the device arrays. Specified bands are set for each of the device arrays and the algorithm allocate array to each CUDA host before releasing the data in its inlets (the outlets of *SurLens* Data-preprocessing algorithm) to nvcc. All the threads raised within the algorithm are accepted through nvcc and all blocks are copied to the shared memory system. Bands are set for the size of processed data that are passed to the outlet of this algorithm. The algorithm can only permit a maximum size 1 kilobyte (1KB) of processed data to exit to the next phase of algorithm for processing. Processed datasets greater than 1KB will be automatically fragmented into orthogonal slices before being pushed forward for further processing within the framework. Bands check measure is to avoid computational overhead in the framework.

Processed data from *SurLens* Accelerating Hardware Algorithm is further processed at the *SurLens* Graphic Execution Algorithm. *SurLens* Feature & Edge Detection Scheme and the *SurLens* Mapping Technique are attached to the *SurLens* Graphic Execution Algorithm. The *SurLens* Graphic Execution Algorithm at this phase



firstly confirms the data formatting of the in-coming data and ensure it's in accordance with the pre-defined structure in the previous algorithms, the *SurLens* Accelerating Algorithm. Filter class is invoked, which may have one or more input (likewise one or more output), depending on the volume of data required to process. The filter class is invoked at this phase for thorough cleansing and final modification of the data prior to its entering into the rendering phase. In the same vein, this algorithm ensures efficient conversion of all data into tangible objects, the graphics primitives. Data mapper class interfaces geometric structures and data attributes to the appropriate graphic library within the framework in order to have fast, interactive display of data thereby ensuring flexibility by preventing overheads due to complex data conversions.

The responsibilities of the entire rendering processes and user define interactive functions are attached to the *SurLens* volume rendering algorithm. The algorithm load and display brain MRI volume and enable seamless communication between the mouse and the application. It displays the render window with the 3-D volume and creates all required pipeline objects for the rendering. The algorithm ensures all the graphic and visualization pipelines scheduled in the framework are routinely executed as any delay or jump stage may lead to poor output image. Functions for colour, opacity and gradient curves are executed. Camera control, trackball movement, joystick usage are all activated by the algorithm to facilitate efficient application interactivity. At every execution of routines, global variables are periodically updated and cleaned-up to mark the completion of a task.

SurLens Visualization System is tested and evaluated using Magnetic Resonance (MR) Imagery from Computer-Assisted Surgery and Imaging Laboratory, University of North Carolina, United States. These datasets consist of the conventional MRI, *the T2-weighted* and the advanced MR techniques, *the Magnetic Resonance Angiography (MRA), Diffusion Tensor Imaging (DTI), T1-Fast Low Angle Shot Magnetic Resonance (T1-FLASH), the T1-Magnetization Prepared Rapid Gradient Echo (T1-MPRAGE)* which are specialized MR imaging techniques. About *Four Hundred and Forty (540)* datasets of patients were used for this study, with each dataset containing a minimum of 160 image slices. They were divided into five age groups; each age group (*18-29, 30-39, 40-49, 50-59 60-74* years) comprised 20 subjects equally divided by gender



(Bullitt et al., 2010). Some of the datasets are of patients with abnormalities such as *hypertension, diabetes, walderstrom's macroglobulinemia, strokes and penetrating brain injury* while some are for normal volunteer patients.

Table 4.1: Experimental Testbeds

Testbed I	
CPU	Pentium (R) Dual-Core, CPU T4200, 2.0 GHz, 3.00 GB RAM
Operating System	Window XP Professional, 32-bit Operating System
Testbed II	
GPU	NVIDIA GeForce GT 520 M CUDA - 1GB
Operating System	Window 7 Home Premium, 64-bit Operating System
Testbed III	
GPU	NVIDIA Quadro 600 CUDA – 2GB
Operating System	Window 7 Professional, 64-bit Operating System

The datasets were approved by the Institutional Review Board of the University of North Carolina at Chapel Hill, was in compliant with the United States' Health Insurance Portability and Accountability Act (HIPAA). All were tested for experimental purposes and the results were published in National Institute of Health (NIH) Journal. Most of the images were in *MetalImage* format prior to *SurLens Pre-processing stage* during which they were converted into *SurLens ImageData (.vti) format*. We based our evaluation on the adjunct information from the department of surgery, University of North Carolina, United States.

4.2 Conversion Scheme

The design of *SurLens conversion scheme* is to have a robust, extensive and reusable data structuring and data reduction procedures within the framework. Major data objects employed in the design of *SurLens* conversion scheme are the *DataArrays*, *ImageData* and *PointSet* classes.

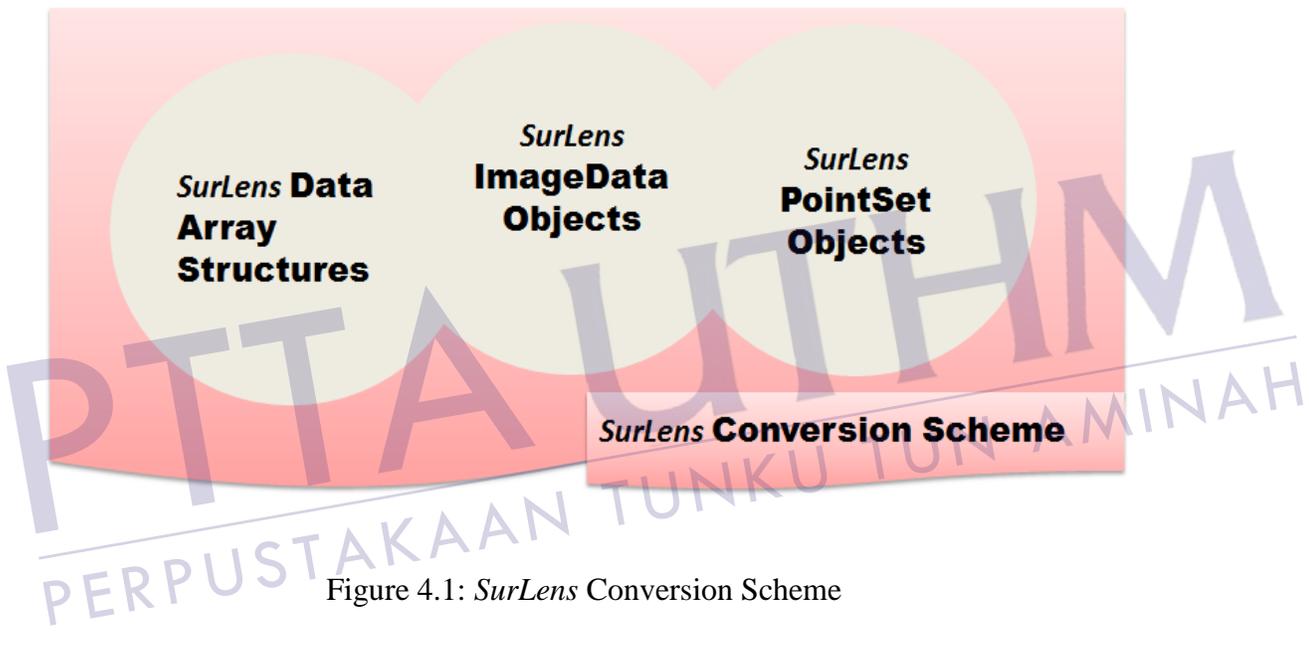


Figure 4.1: *SurLens* Conversion Scheme

The conversion scheme handles structuring of received 2-D images, their filtering and immediate reduction for the framework processing. Figure 4.1 illustrates *SurLens Conversion Scheme*. With the parallel architectural design of *SurLens* framework, the conversion scheme is attached to the *SurLens Data Pre-Processing*, *SurLens Accelerating Hardware* and *SurLens Graphic Execution Algorithms* developed for the framework.

Data arrays are data objects that represent contiguous arrays of data such as *char*, *int*, *float*. Data array is used in the design in order to represent attribute and field data such as scalar, vectors, normal, texture and tensor coordinates. With these data arrays, designed internal memory of *SurLens* conversion scheme is managed through dynamic

allocation and representation of arrays. Figure 4.2 illustrates the design pipeline for Data Array Structures in *SurLens* conversion scheme.

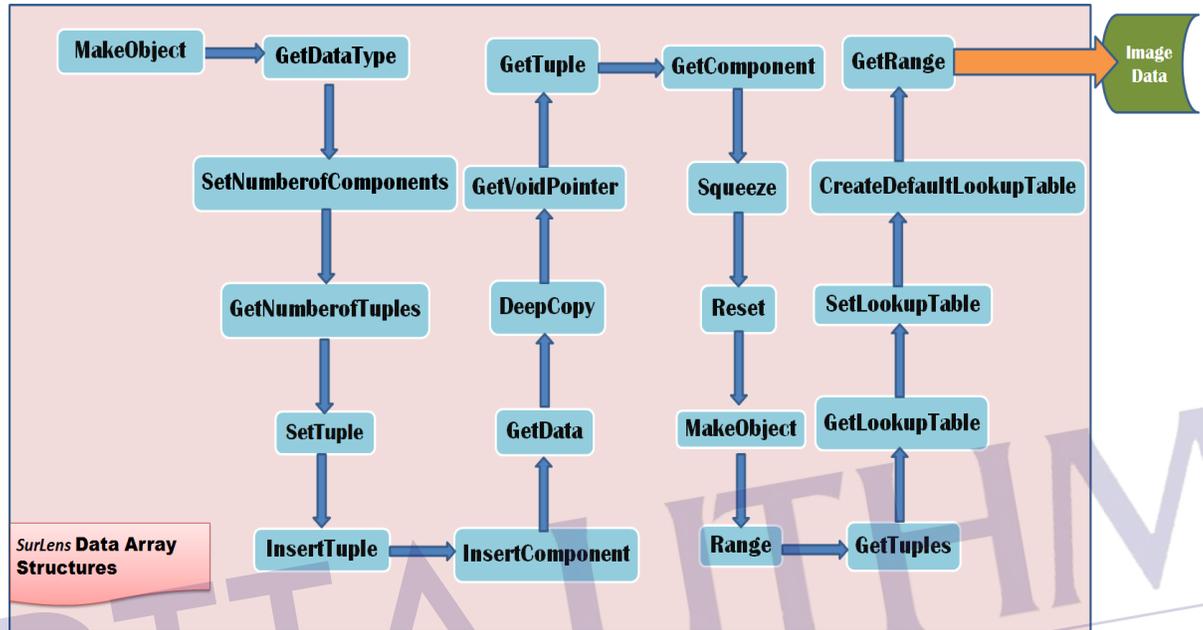


Figure 4.2: *SurLens* Conversion Scheme: Design Pipeline for Data Array Structures

The creation of virtual constructor is the first step to be achieved for design of data arrays in *SurLens* Conversion Scheme. This virtual constructor is an instance creation of the data array. Once the instance of the same data type is created, the native type of the data must be returned as an integer token using the function *GetDataType*. Number of components per tuple is specified and obtained using *SetNumberOfComponents(numComp)* and *GetNumberOfComponents()* functions respectively. However, in order to allocate storage, *SetNumberOfTuples(number)* must be invoked which is tied to the successful execution of previous *SetNumberOfComponents()*.

Processing of tuple in the data array requires getting the number of tuples in the array. This is achieved using *GetNumberOfTuples()*. Similarly, returning a pointer to an array, *GetTuple(i)* and filling-in the previously allocation of tuple *i* using

InsertTuple(i, tuple). Using *GetData(tupleMin, tupleMax, CompMin, CompMax, data)*, a rectangular array of data is extracted, perform deep copying of data and return void * pointer to the data.

In order to ensure robustness and misuse of memory, *Squeeze()* method is invoked to claim any unused memory and *Reset()* to check excessive allocation or de-allocation of memory. The entire component is ranged into Minimum (Min) and Maximum (Max) *ith* component before the configuration of *LookUpTable* for the scheme. The *LookUpTable* is used for mapping values to colours. The *Range* is obtained before further conversion processing in the *ImageData* class.

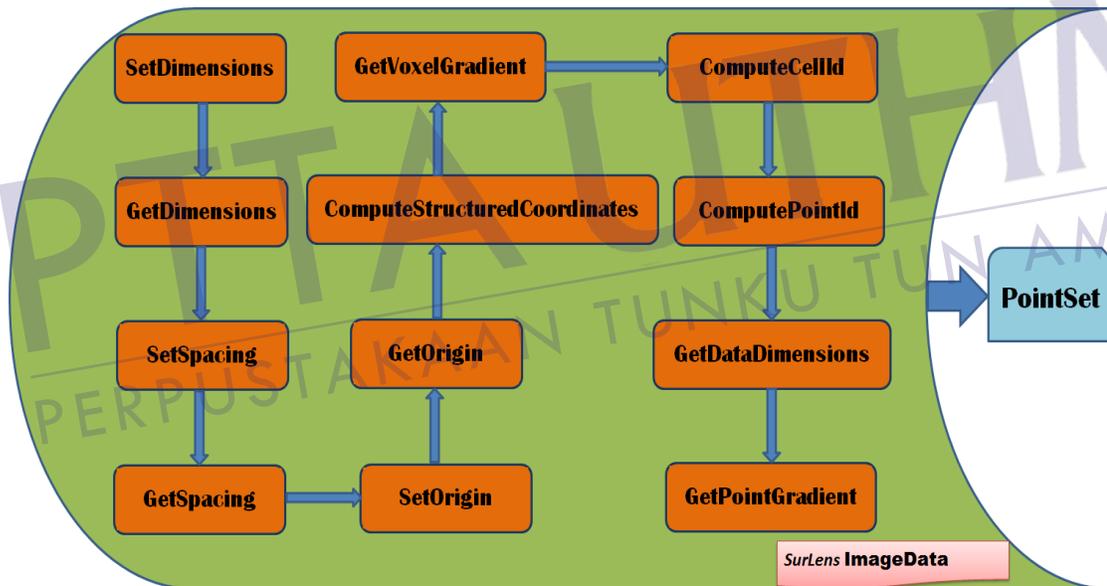


Figure 4.3: *SurLens* Conversion Scheme - Design Pipeline for ImageData

SurLens ImageData class represents a concrete dataset structural element, aligning image into the three (3) coordinates. The geometry and topology of the datasets of 1-D and 2-D can be represented in 3-D depending on the class invocation. Figure 4.3 shows the design pipeline for *SurLens ImageData*.

The points of the dataset in the scheme is structured into three (3) coordinates using $SetDimensions(i,j,k)$ and simultaneously returning i,j,k dataset dimension of array size 3. All the 2-D image slices entering the framework have spacing in-between. The scheme sets spacing using $SetSpacing(Sx,Sy,Sz)$, returning the pointer to an array with $GetSpacing()$.

Origin of the structured points in the dataset are set using $SetOrigin(x,y,z)$ and likewise returned as pointer to the array with $GetOrigin()$. The spacing settings and origin structure coordinates was computed for the dataset in order to obtain the voxel gradients with specification in i,j,k coordinates. The computational unit completes computation with allocated *Cell Ids* and *Point Ids* needed by *PointSet Class* for data dimension and point gradient.

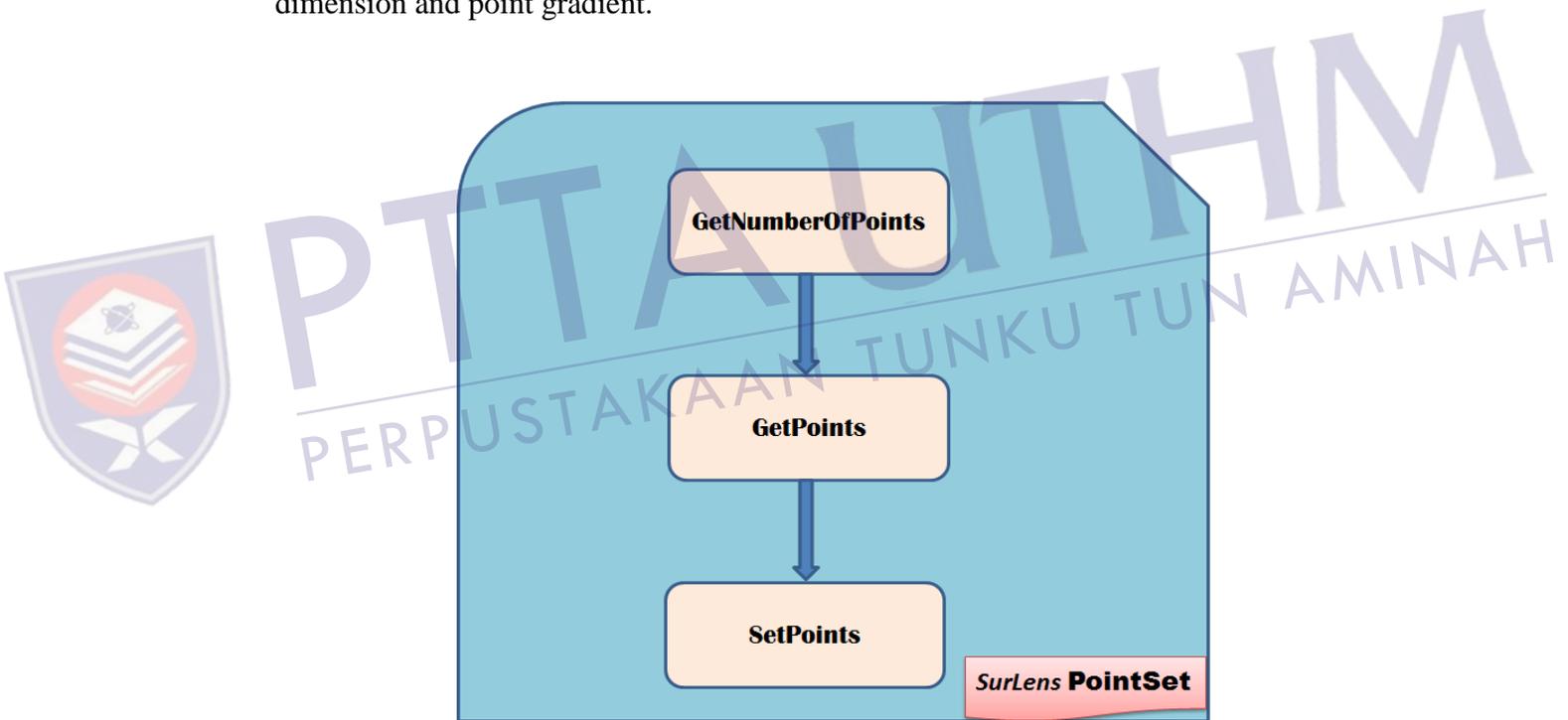


Figure 4.4: *SurLens* Conversion Scheme - Design Pipeline for PointSet

The primary function of *SurLens PointSet* Class is to explicitly specify and represent points of the datasets. However, the design, though dependent on the other classes, it is likewise critical in order to avoid overloading the entire framework. The *PointSet class*

search, obtain and return the number of points in the datasets. Figure 4.4 shows the *SurLens PointSet* Class.

4.3 Feature & Edge Detection Scheme

Detecting the features and edges in volume datasets is a general challenge in volume visualization. Meanwhile, detecting brain features for medical studies is more challenging for the fact that brain contains some tiny features such as brain blood vessels that cannot be singly identified with normal human imagination. Such features require good detection scheme that can make them available for diagnosis. *Contour*, *Anagyph*, *zBuffer*, *Colour Buffer* and *Stereobuffer* classes are employed in the design of *SurLens Feature & Edge Detection Scheme*. Figure 4.5 shows design of *Contour* Class.

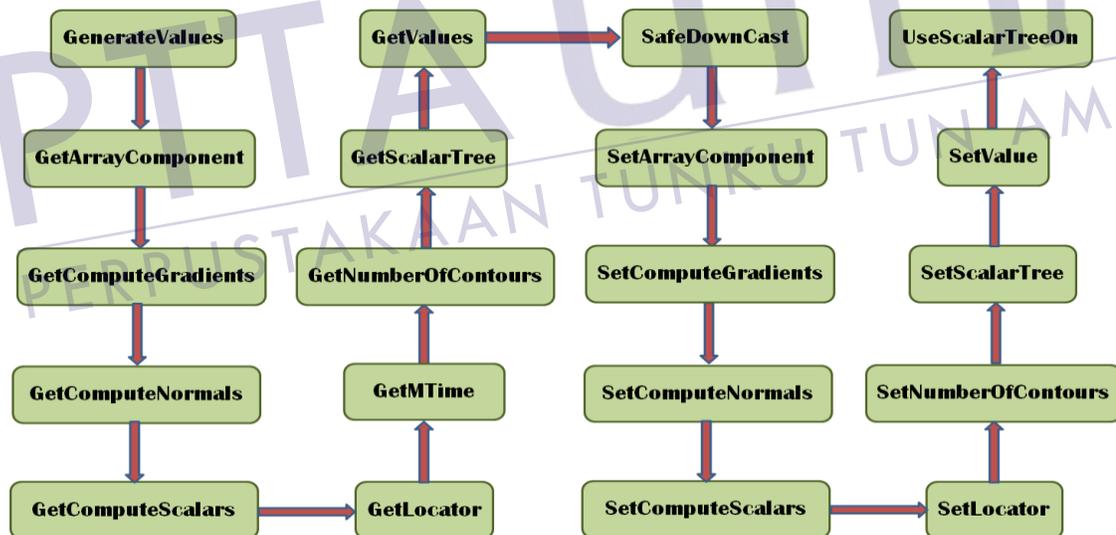


Figure 4.5: Design of Contour Class for *SurLens* Feature & Edge Detection Scheme

SurLens Feature & Edge Detection Scheme firstly generates values for contouring. This consists of equally spaced speed values representing the points in each datasets. The array component to contour is important, the default contour set component is 0.

Setting the array component to contouring facilitates getting of computation of gradient, computation of normals and computation of scalars enabled. However, all the points in the datasets are merged using *GetLocator()*. Each of the time used for the execution is recorded by the scheme for evaluation. Alongside with the listing of contour values with *GetNumberOfContours()* method, the scheme is enabled for scalar tree in order to accelerate the detection of contours likewise a pointer to an array for contour values with *GetValues()* method. A system pulse for reset in the scheme is executed with *SafeDownCast()* method.

At this stage, all the *Gets* methods executed, *the Array Component, Computation Gradients, Computation Normals, Computation Scalars, Locator settings, settings of number of contours, Scalar trees and values* are configured. The entire contouring is activated with *UseScalarTreeOn()*. However, the scalar tree must be turned off for the design of anaglyph and buffering class for the scheme as it can be seen in the Figure 4.6.

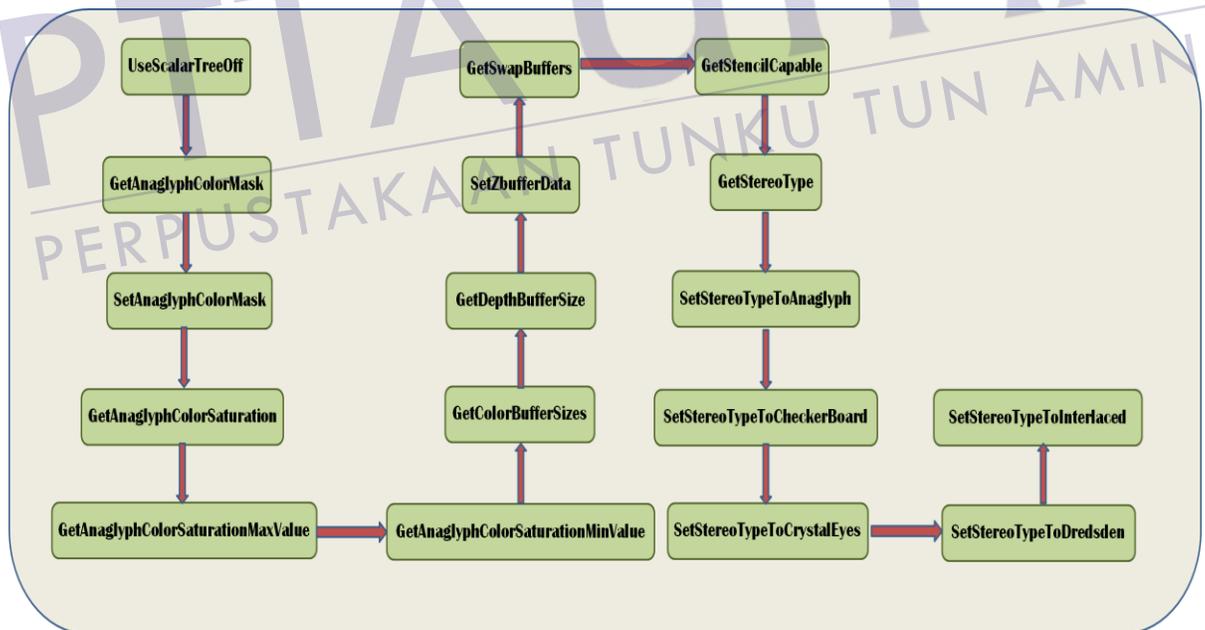


Figure 4.6: Design of Anaglyph & Buffering Class for *SurLens* Feature & Edge Detection Scheme

Undersampling data usually results in aliasing artifacts. However, the process of reducing such aliasing artifacts from data samples is known as antialiasing. The feature and edge detection scheme utilized antialiasing. Antialiasing is important for producing smoother edges. The frame number settings for doing antialiasing is set to 0 for *default*, with up to 6X. Anaglyph colour mask values is for stereoscopic 3-D effect. The colour channels of the original stereo images are used to produce the final anaglyph image. However, such colour channels are controlled by the numbers, the *bits mask*. The bits arrangement are labelled and identified by the application in the left and the right view. The bits are *R,G,B* in value 4, 2 and 1 respectively. *GetAnaglyphColorMask()* and *SetAnaglyphColorMask()* is employed for enabling and activating the Anaglyph mode.

It is important to determine colour saturation factor for each of the datasets in the scheme. The strength of the colour saturation factor is ranged from 0.0 to 1.0 values. 0.0 represents a situation where no object's colour is maintained while 1.0 represents maintaining of the whole colour of the original object. Anaglyph Colour Saturation is activated and the minimum and maximum values are obtained using Anaglyph, *GetAnaglyphColorSaturationMinValue()* and *GetAnaglyphColorSaturationMaxValue()*.

In order to hold data temporarily within the scheme, there are three (3) buffers created. The *colour buffer*, the *depth buffer* and the *stencil buffer*. Colour buffer is the buffer for pixel, the depth buffer is memory containing the depth of element in which its values are stored in the z-buffer. The stencil buffer is designated as per pixel for integer values. The scheme obtains sizes for color and depth buffers with *GetColorBufferSizes()* and *GetDepthBufferSize()* respectively and likewise activate *SetZBufferData()* method. However, buffer can be swapped by activating *GetSwapBuffers()* method. Stencil buffer is needed for stereo rendering. Activating stereo rendering facilitates simultaneous rendering of images and provision for a wide selective modes of the following stereo rendering features. *GetStencilCapable()* and *GetStereoType()* methods are enabled for stereo rendering. The following are some of the facilities offered by stereo rendering:

1. CrystalEyes mode
2. RedBlue mode

3. Anaglyph mode
4. Interlaced stereo mode
5. Dresden mode

The *CrystalEyes* mode uses frame-sequential while the *RedBlue* is a stereo mode used with red-blue glasses. As previously explained, Anaglyph mode uses *AnaglyphColorMask* and the anaglyph colour saturation factor for maintaining original image colour. With interlaced stereo mode, there is an introduction of horizontal lines available in an alternate form from left and right views, these alternate positions are typically referred to as *StereoLeft* and *StereoRight* for left and right views respectively. The last option that can be used with *SurLens* is *dresden* mode which is stereoscopic interleaving. *SetStereoTypeToAnaglyph* (), *SetStereoTypeToCheckerBoard*(), *SetStereoTypeToCrystalEyes*(), *SetStereoTypeToDresden*() and *SetStereoTypeToInterlaced*() methods are configured in the scheme for stereo rendering and its supporting features.

4.4 Automatic Feature Mapping Technique

There is usually a trade-off between quality images and rendering or processing speed of existing frameworks in volume visualization. The clearer the 3-D outputs produced by a volume visualization framework, the better the level of quality of images it could produce. In the same vein, for an independent framework, the rendering speed is the entire processing time it takes a framework to process the datasets into 3-D model.

SurLens Automatic Feature Mapping technique is designed to achieve a very remarkable quality 3-D output that can reveal tiny features such as brain blood vessels. The designed automatic feature technique is attached to the “*Mapping*” component in *SurLens Graphic Execution Phase*. Two different mapper components are designed for the technique; the *graphic mapper* and the *writer mapper*. *SurLens automatic feature mapping technique* is composed of feature & edge extraction, and the feature & image enhancement components as the *graphic mapper*. This components interfaces the

geometric structures and data attributes to the graphic library of the framework. The *writer mapper* stands out as an independent component that is responsible for writing the datasets to the disk for framework utilization. Figure 4.7 illustrates overview of *SurLens Automatic Feature Mapping Technique*.

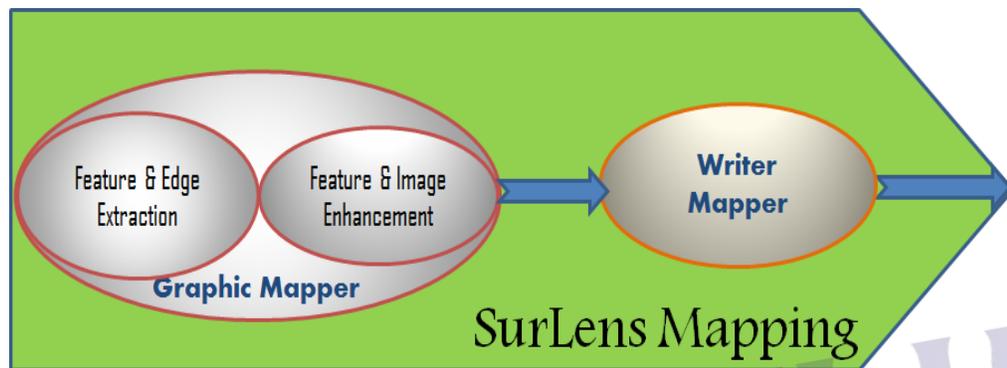


Figure 4.7: Overview of *SurLens* Automatic Feature Mapping Technique

Cell and *Polydata* classes are designed for the development of the graphic mapper component of *SurLens Mapping technique*. A set of classes are designed for smoothing

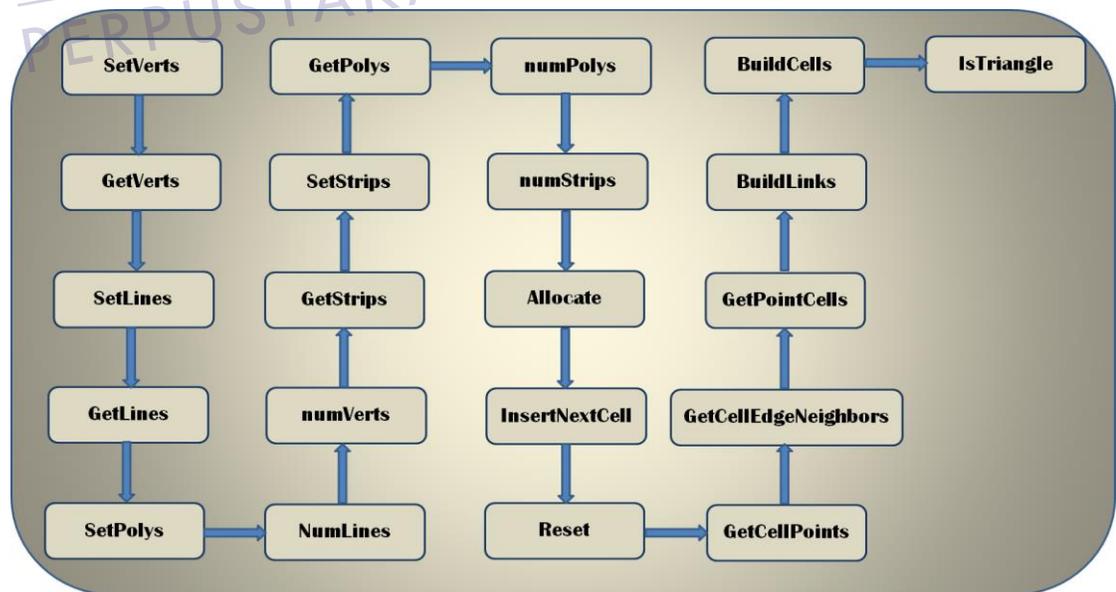


Figure 4.8-a: *SurLens* Automatic Feature Mapping Technique

class employed specifically in the image and feature enhancement. Figure 4.8-a and Figure 4.8-b show *SurLens Automatic Feature Mapping Technique*.

Unlike points, which are defined by cell geometry, vertices are used to represent cell. *SetVerts(verts)* is employed to specify the list of vertices and the specified list of the vertices is obtained with *Verts = GetVerts()* method. Features specifying the list of lines, polygons and strips are enabled using *SetLines(lines)*, *SetPolys(Polys)*, *SetStrips(Strips)* respectively. Similarly, all the specified lists are obtained with Get functions of *GetLines()*, *GetPolys()*, *GetStrips()* respectively. The number of vertices is returned with *numVerts = GetNumberOfVerts()*, the number of lines with *numLines = GetNumberOfLines*, the number of polygons with *numPolys = GetNumberOfPolygons* and the number of triangle strips using *numStrips = GetNumberOfStrips*.

Memory storage is approximate to the number of cells to be inserted and the size of the internal structure is allocated with *Allocate(numcells, extend)* method prior to the utilization of *InsertNextCell()* method which is needed to insert a cell with corresponding return of the cell *id*. With the *SurLens* mapping design, complete execution of a pipeline task requires restoring the class for further processing. *Reset()* method is used to restore the class after each successful execution. Building of cells and

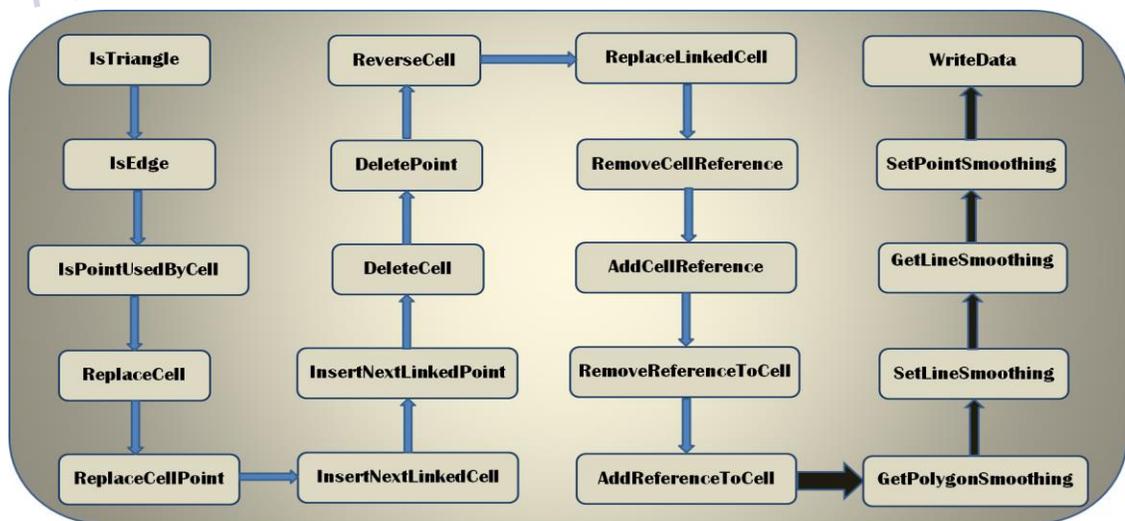


Figure 4.8-b: *SurLens* Automatic Feature Mapping Technique

links of data structures within framework is achieved using *BuildCells()* and *BuildLinks()* respectively. These facilitates the returning of the number of cells using the points identified as *ncells* in the method *GetPointCells(ptId, ncells, cells)*. The cell of the neighboring edge of the data is obtained by invoking *GetCellEdgeNeighbors(CellId, p1 p2, CellIds)* followed by the method to return the number of points in the cell.

A flag method, *IsTriangle()* is invoked for identifying the structure of triangle in the data. The method returns either *1*, if triangle meshes is present or *0* if triangle meshes is absent in the data structure. *IsEdge()* and *IsPointUsedbyCell()* methods are also flags that returned as *0* or *1* for indicating two points as edge and a given point used by a particular *CellId* respectively. Cells can be replaced with new cell *id* by *ReplaceCell(CellId, npts, pts)* where; *CellId* is the *Id* to replace, *npts* is the number of point and *pts* is the specific point to replace its *CellId*. In the same vein, *ReplaceCellPoint(CellId, OldptId, newPtId)* will redefine a new cell point while *ReverseCell()* can reverse the order of the cell connectivity. Point and Cell can be deleted with *DeletePoint(prId)* and *DeleteCell(CellId)* respectively. Likewise, next linked cell or next linked point can be inserted or replaced.

Identifying the cells and points is significant to data mapping techniques. Therefore cell reference, reference to cell or cell to reference can be removed or added. *RemoveCellReference(cellId)*, *AddCellReference(cellId)*, *RemoveReferenceToCell(ptId, cellId)*, *AddReferenceToCell(ptId, cellId)* methods can be executed. The feature and the image enhancement component require smoothing functions in order to contribute to the generation of quality 3-D model. Polygon, lines and point smoothing methods are employed. *GetPolygonSmoothing()* and *GetLineSmoothing()* activate all these features while *SetPointSmoothing()* and *SetLineSmoothing()* invoke the procedures prior to writing the data into the disk or input / output device by the *Writer Mapper*.



4.5 *SurLens* Robust Algorithms' for Mass data

Diagnosing the brain of a patient usually requires analyzing hundreds of MRI slices. According to literature reviewed for this study, one of the greatest challenges confronting the adoption of previously proposed volume visualization frameworks in medical diagnosis and disease therapy is their inability to handle mass datasets. Apart from such huge number of datasets produced by medical imaging devices, their sizes, as a result concentration of picture elements and their overwhelming points are obstacles in achieving robustness in volume visualization framework. This study designed and implemented four (4) pipelined executable algorithms to ensure robustness of *SurLens* framework in handling such enormous medical datasets. The pipelined algorithms designed for this study are discussed in detail in the previous chapter.

The algorithms implemented for *SurLens* framework are developed using object oriented paradigm. Data and operations are fully encapsulated in objects, giving room for effective interaction of objects through “*message passing*” procedures. Programmatic abstraction was efficiently utilized in the design for optimum algorithm performances. These enables prompt processing of data or information and exceptional conditions are greatly handled without causing any delay to the entire processing time. In the same vein, though the algorithms are phased into four (4), the *SurLens Data Pre-Processing Algorithm*, *SurLens Accelerating Hardware Algorithm*, *SurLens Graphic Execution Algorithm* and the *SurLens Volume Rendering Algorithm*, objects are reusable throughout the entire four (4) algorithms and each of the stages in the framework are pipelined and integrated. The phasing of the algorithms is for modularity such that the relationships among components are logically managed to prevent overhead in the framework.

The general procedure in volume visualization is to interface framework with external data source, map the external data into internal form, process the mapped data and generates the 3-D model as output image via computer device. However, data representation in the framework apparently determines the overall performances of the entire framework. The algorithms are designed to support compact and mappable data

structures. Since medical datasets are identified to be large, the algorithms are designed to accommodate compact storage of data with minimum memory capacity for computational accessibility.

User errors, exhaustions and internal errors are adequately catered for in the design of the algorithms. With the design of *SurLens* framework in handling huge size of datasets, possible execution errors, which might due to invalid input from users, are handled. The algorithms are robust enough to structure, process any medical image datasets, and reject any invalid inputs through system exceptions commands. Exhaustions are challenge in shared resources system. Despite the fact that the four (4) algorithms are pipelined, the design ensures efficient resources sharing with methods integrated into the application such as *CommonAppDataPath { get; }* which ensures the application data threads a correct path with a registered key using *CommonAppDataRegistry { get; }*. These prevent any conflicts of data sharing among the algorithms, which may threaten the performances of the entire framework. Meanwhile, each of the phased algorithms confirms the status of the input data in order to tackle any possible internal error in the framework.

4.6 Summary

The major phases in the implementation of *SurLens Visualization System* are the *Dataset Pre-processing*, the *Accelerating Hardware*, the *Graphic Execution* and the *Volume Rendering*. However, all the algorithms within the phases are all integrated and designed to process data in parallel fashion. The first phase, the *Dataset Pre-processing* and the second phase, the *Accelerating Hardware* stages are tagged *SurLens Memory System Architecture*. At this architectural unit, processing of the datasets and possible optimization methods are implemented. *SurLens Conversion Scheme* is attached to *SurLens Memory System Architecture* supporting the architecture in its efficient processing of data. The *Graphic Execution phase*, denoting the third phase of the framework, has two (2) schemes attached to it, the *SurLens Conversion Scheme* and the *SurLens Feature and Edge Detection Scheme*. *SurLens Features Mapping Technique* is



PTTA UTHM
PERPUSTAKAAN TUNKU TUN AMINAH

likewise attached to the *Graphic Execution phase*. The last phase in the framework is the *Volume Rendering phase* which handles major rendering activities in the framework.

All the implementations of *SurLens Visualization System* were achieved entirely in C# except accelerating optimization algorithms, *CUDA*, which was achieved in C programming language, integrated into visual .NET platform. However, all the compilations are within Visualization ToolKit Environment to ensure efficient pragmatic abstraction and generic multi-dimensional data structure approach. The developments involve interfacing of accepted inputs data to display format. There are mappings of input data into internally confined form, processing of the data and subsequent generation of the images for usage. Representations of data are important and have overall effect on the performance of our system.

The mappings of data are majorly in two folds. In order to ensure a fast and interactive display of data, there must be thorough mapping of data into graphics primitive. The second type of mapping involves adequate conversion of input data into usable internal visualization data structures. The interfacing files within the system are the *readers* and the *writers*. The *readers* are the source objects and *writers* are the mappers. Meanwhile, medical data are usually bulky; efficient storage scheme is a key to achieving software flexibility hence *SurLens* implementation was interfaced with *CUDA* as its accelerating hardware.

The Feature and edge detection of datasets are implemented through *SurLens Volume Rendering Phase*. Feature extraction is applied in the highlighting of the data. The algorithms can allocate transparency based on scalar values and assign the transparency based on localized gradient magnitude for edge detection within volume. The algorithms do not require prior segmentation stage. *SurLens* utilizes RGBA for its gradual transition, for exhibiting visualization with respect to data relative density. The implemented system is able to correctly map scalar values to opacity, clearly isolate distinctly and visualize any noticeable abnormalities within the volume. The algorithms leverage accelerating capabilities of *CUDA* in handling mass data within a considerable interactive speed.

The next chapter presents the results and discussion of *SurLens* experimentation with patients' datasets.

CHAPTER 5

RESULTS AND DISCUSSION

This chapter presents and discusses the results of *SurLens* experimentation with medical datasets. Implementation of *SurLens Visualization framework* is discussed in Chapter 4. The accuracy of the framework in detecting, mapping of the internal features and revealing the abnormalities in the brain MRI datasets with quality 3-D image, within a considerable interactive speed is highlighted. The results of the study and its comparison with two (2) notable, previously developed systems, the *ParaView* and the *VolView* are presented. The two (2) visualization systems were chosen for the comparison because the frameworks are the best among all reviewed and compared in Chapter 2 of this thesis, moreover, they are likewise built on top of the Visualization toolkit (VTK) libraries.

5.1. INTRODUCTION

The focus of *SurLens Visualization framework* is to obtain quality 3-D images, sufficient enough to reveal detail internal information of the datasets within considerable interactive speed, having extensive application interoperability with other revolutionary tools and providing an immediate visualization at a resulting lower cost for accurate clinical diagnosis and decision making. Hence, the experimentations with *SurLens* focus on evaluating its strength to producing quality 3-D image and its performances in terms of rendering speed. However, it is difficult to evaluate image quality; often researchers

simply put images of competing algorithms side by side, appointing the human visual system (HSV) to be the judge (Meißner et al., 2000). This is for the confirmed fact that human visual system (HSV) is not sensitive to some errors such as stochastic noise and other regular pattern errors. Therefore, two (2) notable visualization frameworks were chosen as bench of comparison of this study, the *VolView Visualization framework* and *ParaView Visualization framework*. These two frameworks are resourceful frameworks, likewise developed on top of the visualization toolkit (VTK) libraries similar to the development of *SurLens Visualization framework*. In the same vein, *SurLens* framework was evaluated for processing speed with its experimental testbeds.

Series of experiments were performed with *SurLens* using Table 4.1, the experimental testbeds. The evaluation was achieved with *MRA*, *DTI*, *T1-FLASH*, *T2-Weighted*, *T1-MPRAGE*. A number of the volunteers' datasets are with abnormalities while some are for normal patients. In order to evaluate the impact of the designed and implemented robust algorithms for this study, average speed performances with each of the datasets were recorded. Selected samples of the results are reported in this chapter while others are attached as appendix.

5.2. Results of New Feature & Edge Detection Scheme

One of the objectives of this study is to introduce a new feature and detection scheme that can allocate transparency based on scalar values and assign transparency based on localized gradient magnitude for edge detection of region of interest in data volume. The newly proposed, designed and implemented feature detection scheme was evaluated with sets of patients' volunteers' datasets from the department of Surgery, University of North Carolina, Unites States, in order to show the contribution of our new feature and detection scheme to the knowledge domain. Experimentations were carried out with *MRA* datasets, *DTI* datasets, *T1-FLASH* and *T1-MPRAGE*. The results obtained from the study at this stage of the experiment are presented in this section.



PTTA UTHM
PERPUSTAKAAN TUNKU TUN AMINAH

5.2.1 Experimentation with MRA Datasets

Magnetic Resonance Angiography (MRA) is special MRI technique to diagnose brain for tumor / abnormalities. MRA is primarily designed for imaging *blood vessels* of the brain, to generate images of the *arteries* for *stenosis (abnormal narrowing)*, *occlusion or*

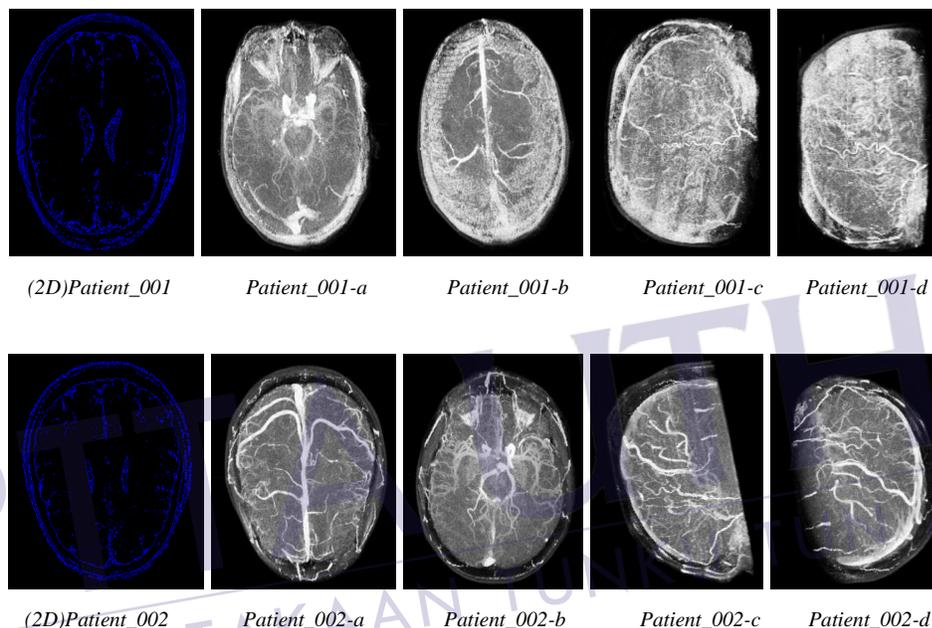


Figure 5.1-a: Results of *SurLens* Feature & Edge Detection with MRA Datasets

aneurysms. The human brain contains many features of higher data input dimensionality that exceeds normal human experience. Some of the features are so tiny and interlaced hence might be difficult to be revealed with volume visualization framework. In Figure 5.1-a, Patient_001 and Patient_002 are in 2-D structure while Patient_001-a to Patient_001-d and Patient_002-a to Patient_002-d are in 3-D structure where we can easily identify the internal features of the brain blood vessels showing the experimental results of the new feature and edge detection scheme with MRA datasets.

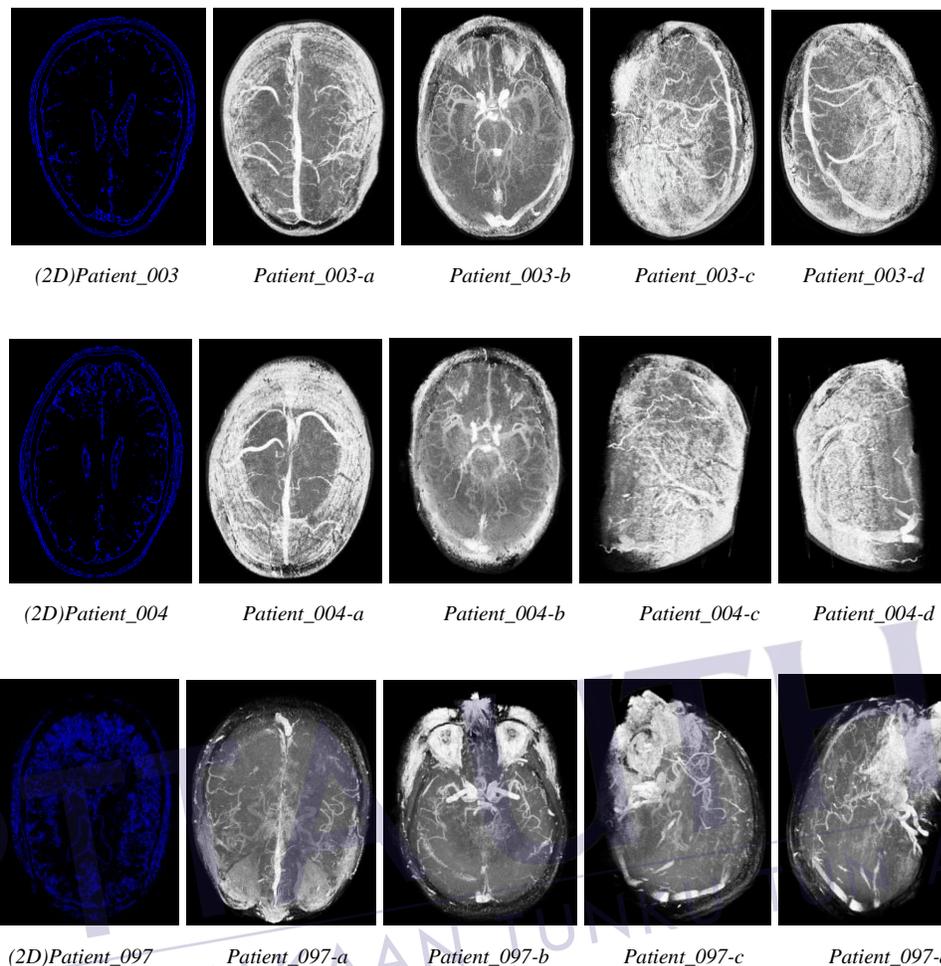


Figure 5.1-b: Results of *SurLens* Feature & Edge Detection with MRA Datasets

SurLens Feature and Edge Detection scheme is able to reveal clearly the features and edges of each of the MRA datasets from 2-D structures reconstructed to 3-D model. The detection of the 2-D structures and the four (4) spatial orientations of the MRA datasets are presented in Figure 5.1 above. In Figure 5.1-b, Patient_004 and Patient_097 are in 2-D structure while Patient_004-a to Patient_004-d and Patient_097-a to Patient_097-d are in 3-D structure where we can easily identify the internal features of the brain blood vessels. The results show the detection of the features in the brain structures. It clearly identifies edges of the datasets. Though the blood vessels structures are tiny, the scheme is able to show all the vessels with active detection. Some features represented as points and lines are revealed. The scheme is able to achieve one of the outstanding challenges

in volume visualization. Therefore, subsection 2 of the first objective and the second objective of this study, have been achieved with Magnetic Resonance Angiography (MRA) datasets.

5.2.2 Experimentation with DTI Datasets

Diffusion Tensor Imaging (DTI) is one of the advanced and special magnetic resonance imaging employed in brain diagnosis and disease treatment. DTI is used in determining

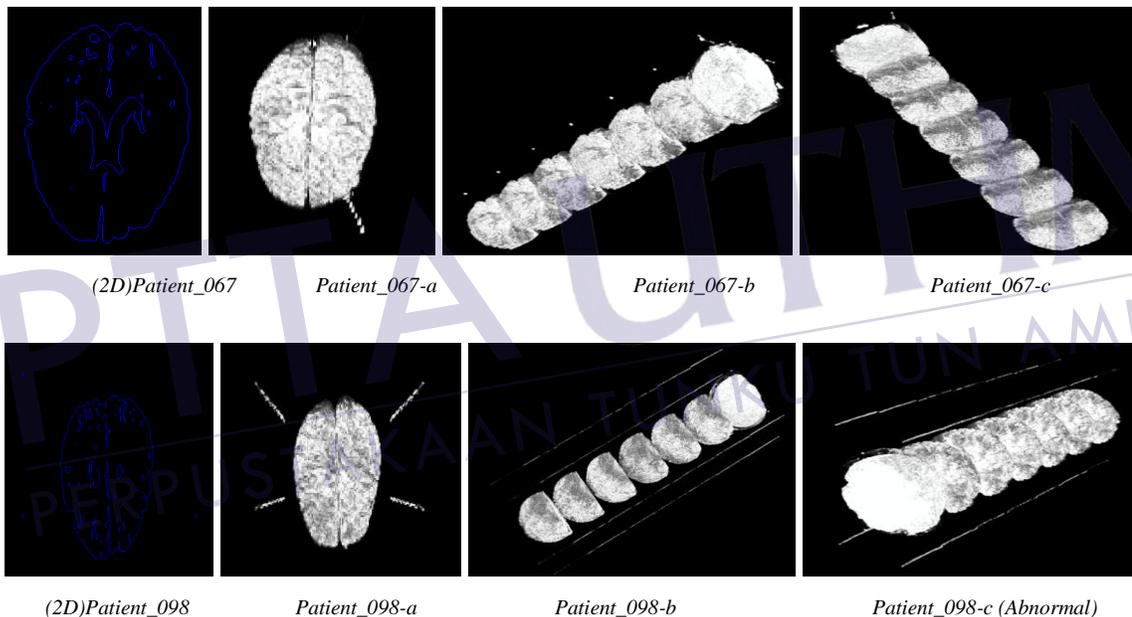


Figure 5.2-a: Results of *SurLens* Feature & Edge Detection with DTI Datasets

the *magnitude and direction* of water in the brain. In cases where patients is suspected to have internal fibrous structure in his / her brain regions, such as neural axons of white matter, DTI is used to study the movement of water in the direction aligned with the internal structure. Figure 5.2-a and Figure 5.2-b show the experimental results of the new feature and edge detection scheme with DTI datasets. In Figure 5.2-a, Patient_067 and Patient_098 are in 2-D structure while Patient_067-a to Patient_067-c and

Patient_098-a to Patient_098-d are in 3-D structure where we can easily identify the internal features and edges of the DTI datasets of Patient_067 and Patient_098.

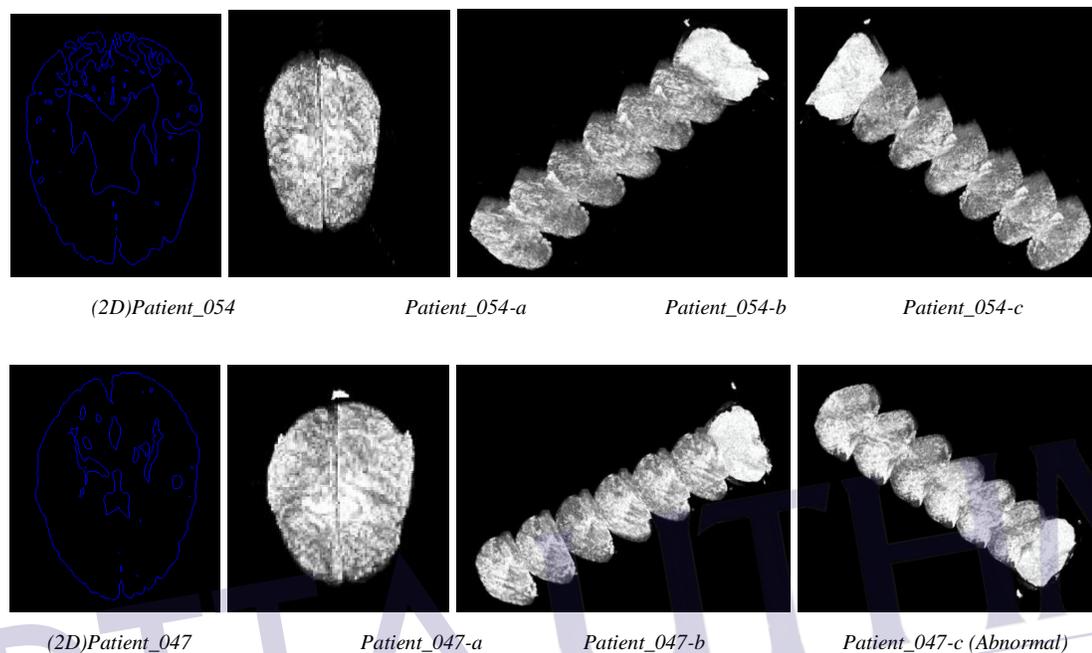


Figure 5.2-b: Results of *SurLens* Feature & Edge Detection with DTI Datasets

In Figure 5.2-b, Patient_054 and Patient_047 are in 2-D structure while Patient_054-a to Patient_054-c and Patient_047-a to Patient_047-c are in 3-D structure where we can easily identify the internal features and edges of the DTI datasets of Patient_054 and Patient 047. The detection results identify the direction and the magnitude of water within the datasets, which can be further revealed with mapping techniques. Among the datasets are those of abnormal patients, however, the algorithms show as much as every faintly represented structures in the datasets. Some of the tiny structures detected in the datasets, in the case of Patient_047, are responsible for its classification as abnormal based on the adjunct information that accompanied the datasets. However, finding the distinct characteristic of such abnormalities is beyond the scope of this study. The results show that the subsection 2 of the first objective and the second objective of this study have been achieved with Diffusion Tensor Imaging (DTI) datasets.

5.2.3 Experimentation with T1-FLASH Datasets

The *T1-Fast Low Angle Shot Magnetic Resonance (T1-FLASH)* is a special Magnetic Resonance Imaging employed in the diagnosis of *glioma tumor* and *lesions*. There is usually a trade-off between quality images and acquisition time in some of the available imaging techniques. *T1-FLASH* is one of the imaging techniques that sacrifices its quality image output for rapid acquisition. Volume visualization scheme needs to be sensitive enough in order to detect features and the edges of image with T1-FLASH. Figure 5.3 shows the experimental results of the feature and edge detection scheme with *T1-FLASH* datasets.

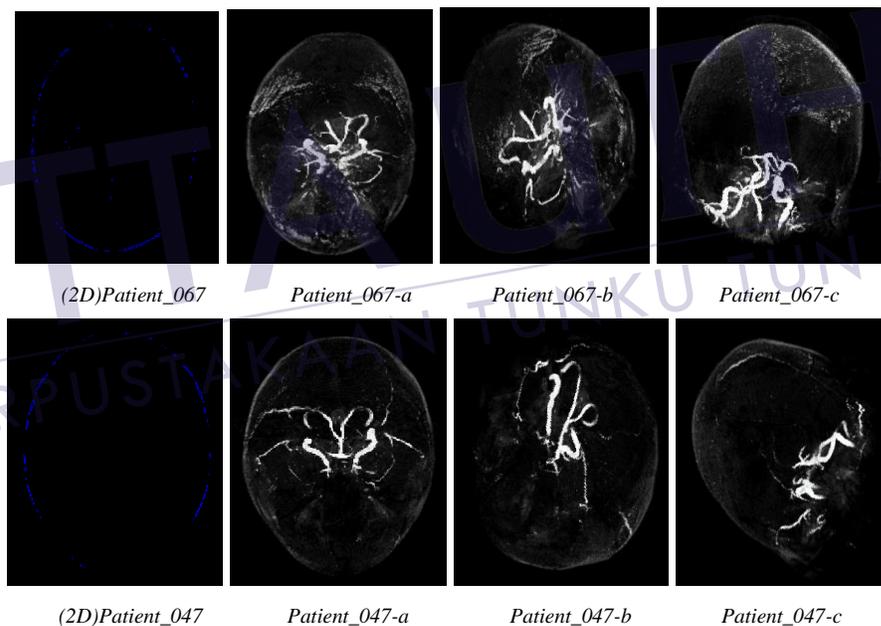


Figure 5.3a: Results of *SurLens* Feature & Edge Detection with T1-FLASH Datasets

In Figure 5.3-a, Patient_067 and Patient_047 are in 2-D structure while Patient_067-a to Patient_067-c and Patient_047-a to Patient_047-c are in 3-D structure where we can easily identify the internal features and edges of the T1-FLASH datasets of Patient_067 and Patient_047.

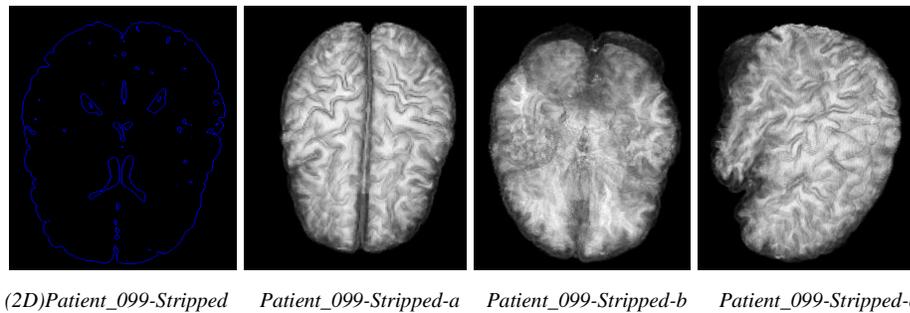


Figure 5.3b: Results of *SurLens* Feature & Edge Detection with T1-FLASH Datasets

Similarly, Patient_099-stripped in Figure 5.3-b is in 2-D structure while Patient_099-stripped-a to Patient_099-c are in 3-D structure where we can easily identify the internal features and edges of the T1-FLASH datasets of Patient_099-stripped.

Despite the poor quality of images produced by *T1-FLASH* imaging device, *SurLens Feature and Edge Detection Scheme* is able to reveal the structures for possible medical diagnosis. One of the objectives of this study, the subsection 2 of the first objective and the second objective of this study have been achieved with *T1-Fast Low Angle Shot Magnetic Resonance (T1-FLASH)* datasets.

5.2.4 Experimentation with T1-MPRAGE Datasets

T1-Magnetization Prepared Rapid Gradient Echo (T1-MPRAGE) is in the category of rapid acquisition imaging technique similar to *T1-FLASH*. *T1-MPRAGE* is usually employed for detecting *metastatic brain tumors*. Metastatic brain tumors are tumors that originate from other part of the body and travels to the brain. One of the motives of using *T1-MPRAGE* is to produce images of the patients' brain as soon as possible, which is at the expense of quality image output of the device. Figure 5.4 shows the experimental results of the feature and edge detection scheme with *T1-MPRAGE* datasets.

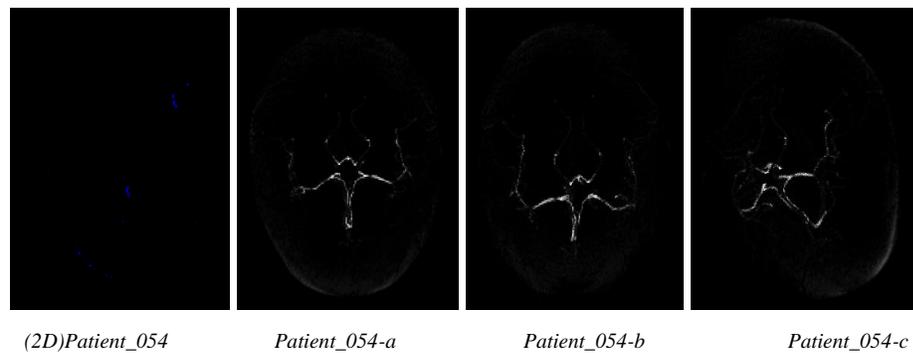


Figure 5.4a: Results of *SurLens* Feature & Edge Detection with T1-MPRAGE Datasets

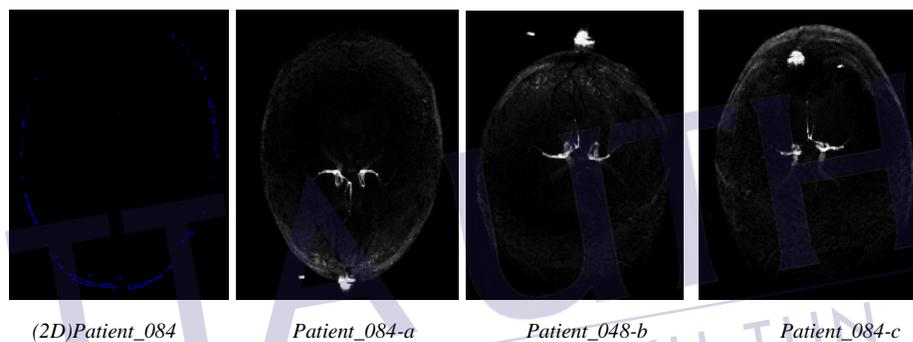


Figure 5.4b: Results of *SurLens* Feature & Edge Detection with T1-MPRAGE Datasets

In Figure 5.4, Patient_054 and Patient_084 are in 2-D structure while Patient_054-a to Patient_054-c and Patient_084-a to Patient_084-c are in 3-D structure where we can easily identify the internal features and edges of the T1-MPRAGE datasets of Patient_054 and Patient 084. The above results confirmed the achievement of the subsection 2 of the first objective and the second objective of this study with T1-*Magnetization Prepared Rapid Gradient Echo (T1-MPRAGE)* datasets.

5.3. Results of New Feature Mapping Techniques

The detection of features and edges of brain MRI datasets achieved by *SurLens* scheme, as being documented in the previous chapter, would be more contributing if the features

are automatically mapped with distinct isolation of possible abnormalities / tumor in the datasets. Such results will go a long way in given more clues to medical professional concerned in the surgery and disease diagnosis procedure. However, one of the objectives of this study is to introduce a new technique with automatic local feature mapping scheme that can isolate abnormalities / tumor and reveal internal features of brain blood vessels. The newly feature mapping technique is evaluated with *MRA*, *DTI*, *T1-FLASH* and *T1-MPRAGE*. The results of the study and its comparison with the *ParaView* and the *VolView* techniques are presented in the subsequent sections.

5.3.1 Experimentation with MRA Datasets

Magnetic Resonance Angiography (*MRA*) is a specialized magnetic resonance technique for imaging the human brain blood vessels, blood flows and the fluids surrounding the brain. Abnormalities associated with blood brain vessels are called *Vascular Abnormalities*. Vascular abnormalities may be as a result of inflicted abnormalities in the brain arteries or veins, the brain may be deprived of functional blood or oxygen flow, which can lead to many life-threatening cases.

About one hundred and nine (109) *MRA* datasets of volunteer patients' were used for this study. The images were divided into five age groups; each age group (18-29, 30-39, 40-49, 50-59 and 60-74 years) comprised 20 subjects equally divided by gender (Bullitt et al., 2010). While majority of the subjects were healthy volunteers, some were with certain cases of abnormalities such as hypertension, diabetes, walderstrom's macroglobulinemia, stroke and penetrating brain injury. The adjunct information from the department of surgery, University of North Carolina guided the author in the evaluation of this study. In Figure 5.5, Patient_001, and Patient_002 are in 2-D structure while Patient_001-a to Patient_001-d and Patient_002-a to Patient_002-d are in 3-D structure where we can easily identify the internal features, edges, automatic mapping and isolation of abnormalities / tumor in the brain blood vessels.

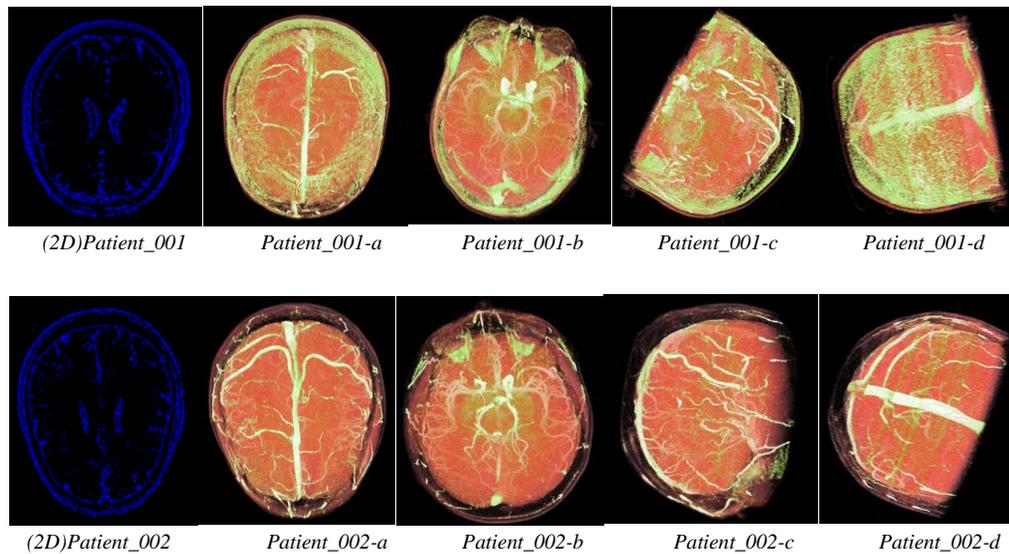


Figure 5.5: MRA Evaluation Datasets of Patient_001 and Patient_002

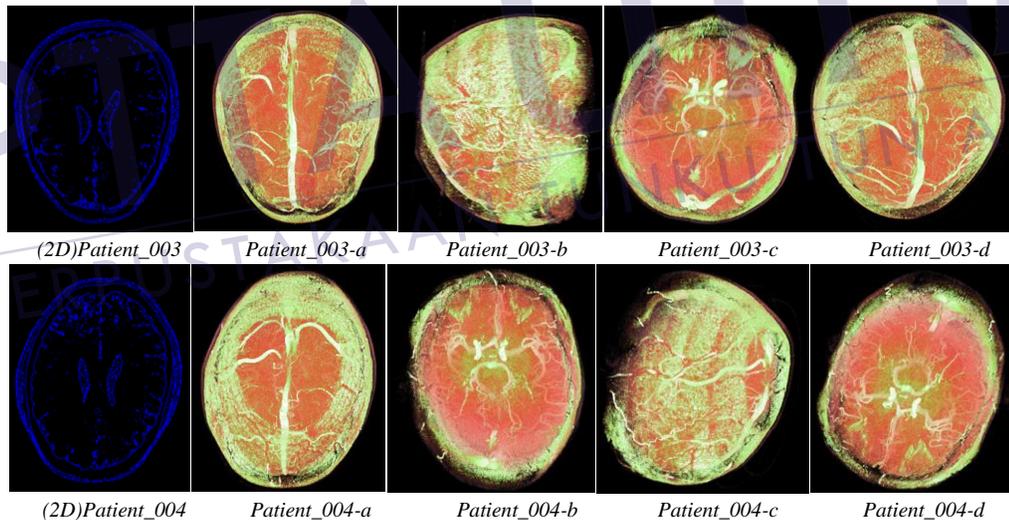


Figure 5.6: MRA Evaluation Datasets of Patients 003 and Patient_004

MRA dataset is used to image brain blood vessels to generate images of the arteries in order to evaluate them for such cases as abnormal narrowing, *stenosis* and vessel wall dilations, *occlusion or aneurysms*. Patients' datasets in 3-D model were taken with *SurLens* in four (4) spatial orientation positions. In Figure 5.6, Patient_003 and Patient_004 are in 2-D structure while Patient_003-a to Patient_003-d and Patient_004-a

to Patient_004-d are in 3-D structure where we can easily identify the internal features, edges, automatic mapping and isolation of abnormalities / tumor in the brain blood vessels.

SurLens Feature Mapping technique is able to reveal detail features of the datasets. The new feature mapping technique is able to automatically map and isolate abnormalities / tumor based on the local features and distinctly reveal internal features of brain blood vessels.

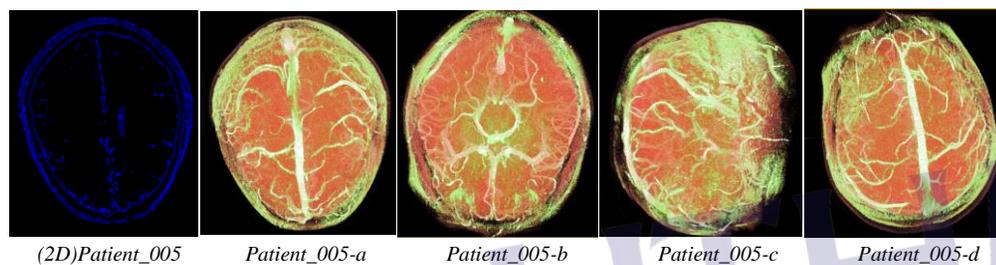


Figure 5.7: MRA Evaluation Datasets of Patient_005

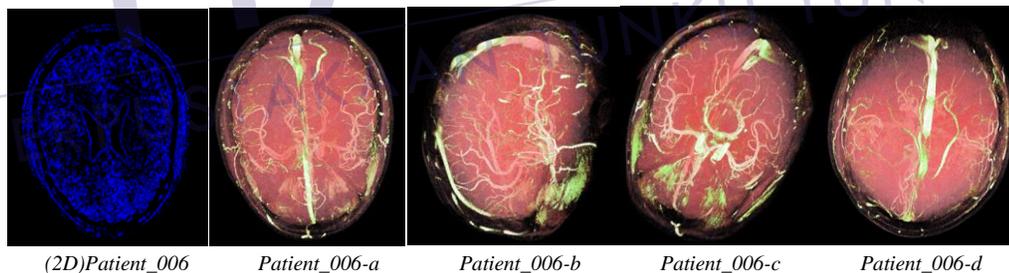


Figure 5.8: MRA Evaluation Datasets of Patient_006

In Figure 5.7 and Figure 5.8, Patient_005 and Patient_006 are in 2-D structure while Patient_005-a to Patient_005-d and Patient_006-a to Patient_006-d are in 3-D structure where we can easily identify the internal features, edges, automatic mapping and isolation of abnormalities / tumor in the brain blood vessels. Looking through the images, the brain blood veins are clearly shown and other parts of the structures are in varying coloration, which is as a result of the presence of different chemical fluids or

substances surrounding the patient's brain. Age and sex are some of the reasons for such varying coloration.

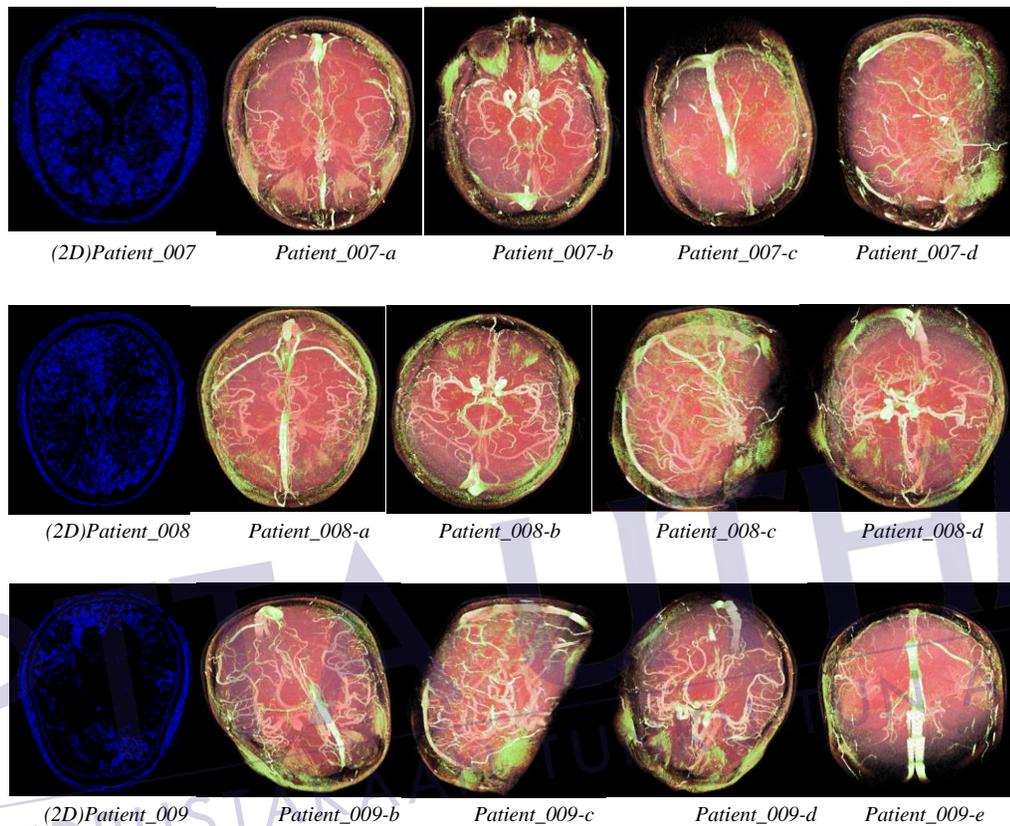


Figure 5.9: MRA Evaluation Datasets of Patient_007 to Patient_009

In Figure 5.9 above, Patient_007, Patient_008 and Patient_009 are in 2-D structure while Patient_007-a to Patient_007-d, Patient_008-a to Patient_008-d and Patient_009-b to Patient_009-d are in 3-D structure where we can easily identify the internal features, edges, automatic mapping and isolation of abnormalities / tumor in the brain blood vessels.

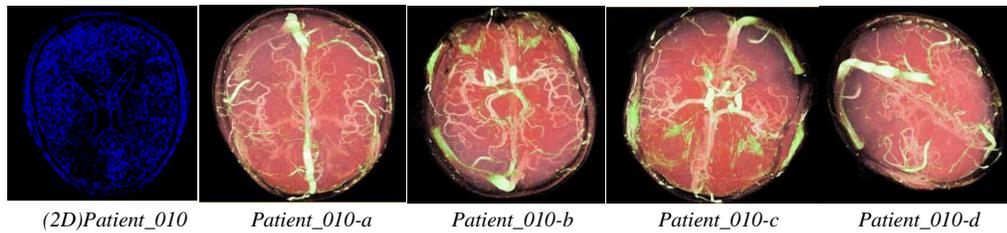


Figure 5.10: MRA Evaluation Datasets of Patient_010

Presences of some elements of abnormalities in the structure are equally represented with the greenish coloration. Although based on the adjunct information that accompanied the datasets, some of the dataset are not necessarily classified abnormal as a results of such greenish pigments. Patient_010 in Figure 5.10 is in 2-D structure while Patient_010-a to Patient_010-d are in 3-D structure where we can easily identify the internal features, edges, automatic mapping and isolation of abnormalities / tumor in the brain blood vessels.

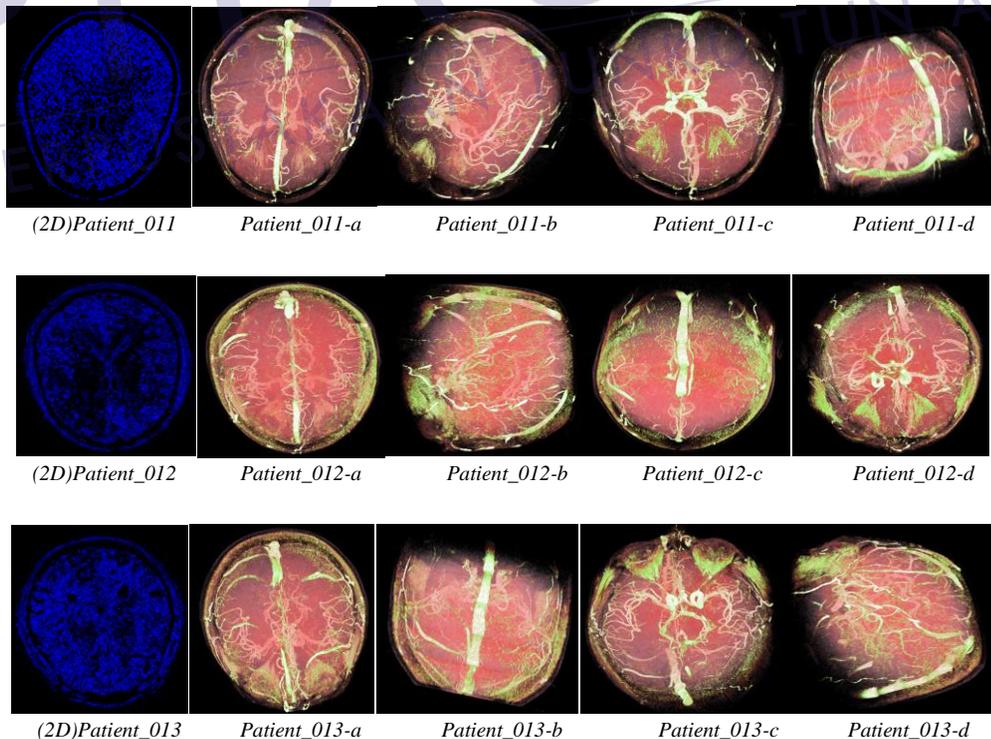


Figure 5.11: MRA Evaluation Datasets of Patient_011 to Patient_013

In Figure 5.11, Patient_011, Patient_012 and Patient_013 are in 2-D structure while Patient_011-a to Patient_011-d, Patient_012-a to Patient_012-d and Patient_013-a to Patient_013-d are in 3-D structure where we can easily identify the internal features, edges, automatic mapping and isolation of abnormalities / tumor in the brain blood vessels.

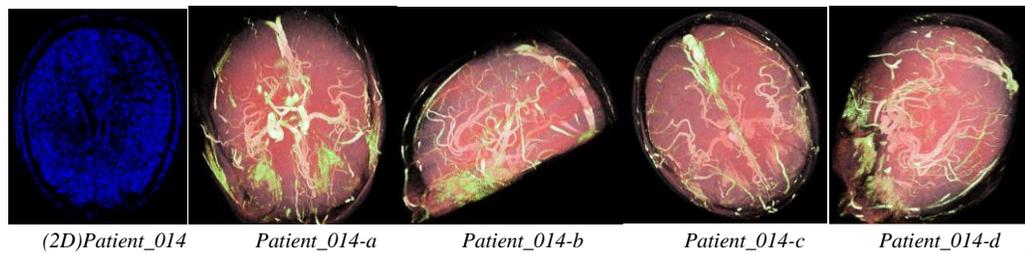


Figure 5.12a: MRA Evaluation Datasets of Patient_014 to Patient_016

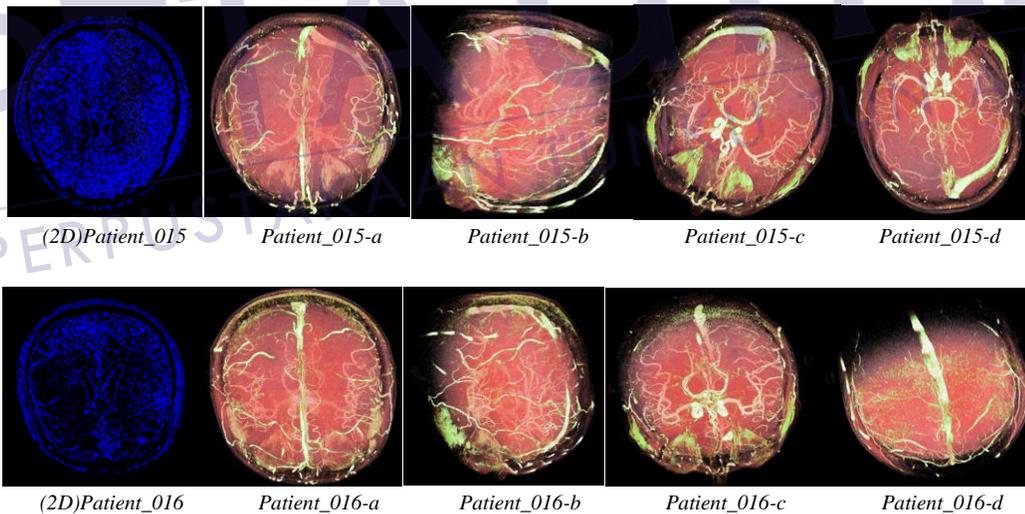


Figure 5.12b: MRA Evaluation Datasets of Patient_014 to Patient_016

In Figure 5.12, Patient_014, Patient_015 and Patient_016 are in 2-D structure while Patient_014-a to Patient_014-d, Patient_015-a to Patient_015-d, and Patient_016-a to Patient_016-d are in 3-D structure where we can easily identify the internal features, edges, automatic mapping and isolation of abnormalities / tumor in the

brain blood vessels. Studies with the datasets show variance in the structure and the features of human brain relative to the sex and age. Two patients of the same sex might not necessarily have exactly the same structure of the brain even when both are classified normal.

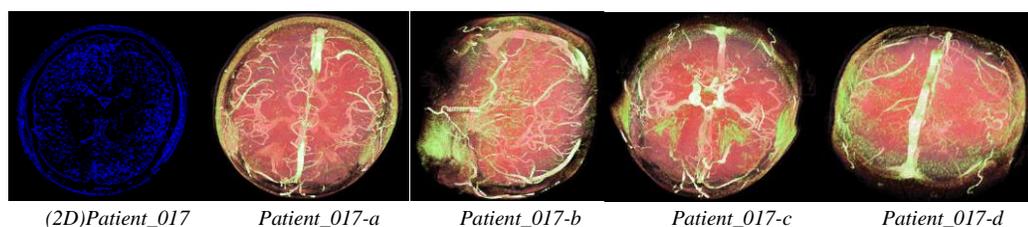


Figure 5.13: MRA Evaluation Datasets of Patient_017

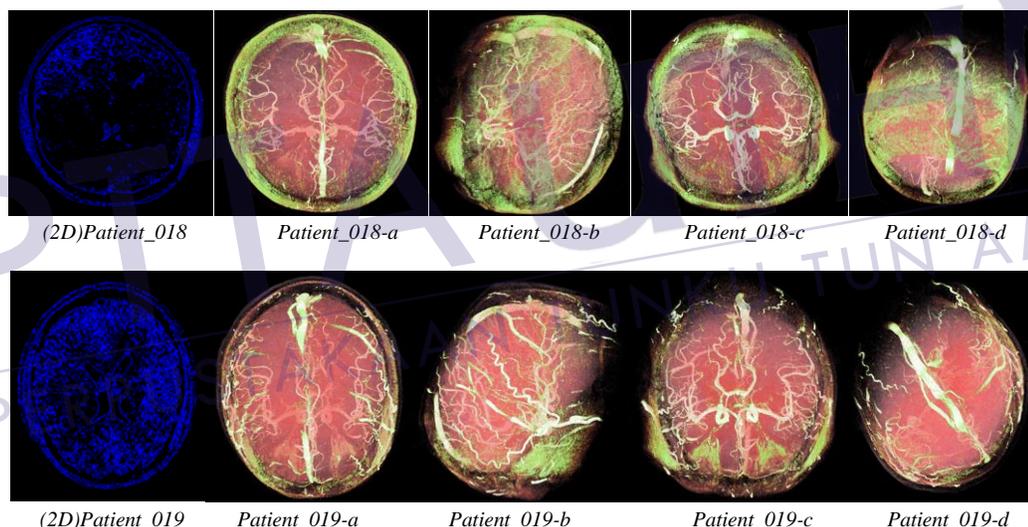


Figure 5.14: MRA Evaluation Datasets of Patient_018 and Patient_019

SurLens Visualization System utilizes *RGBA* technique for its gradual transition; for exhibiting visualization with respect to data relative densities / intensities. In Figure 5.13 and Figure 5.14, Patient_017, Patient_018 and Patient_019 are in 2-D structure while Patient_017-a to Patient_017-d, Patient_018-a to Patient_018-d and Patient_019-a to Patient_019-d are in 3-D structure where we can easily identify the internal features, edges, automatic mapping and isolation of abnormalities / tumor in the brain blood vessels.

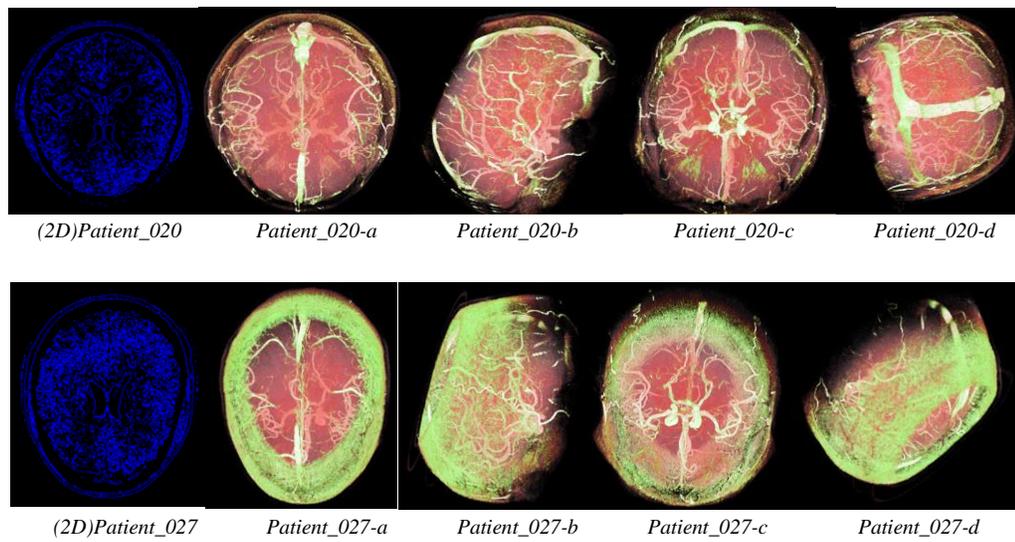


Figure 5.15: MRA Evaluation Datasets of Patient_020 and Patient_027

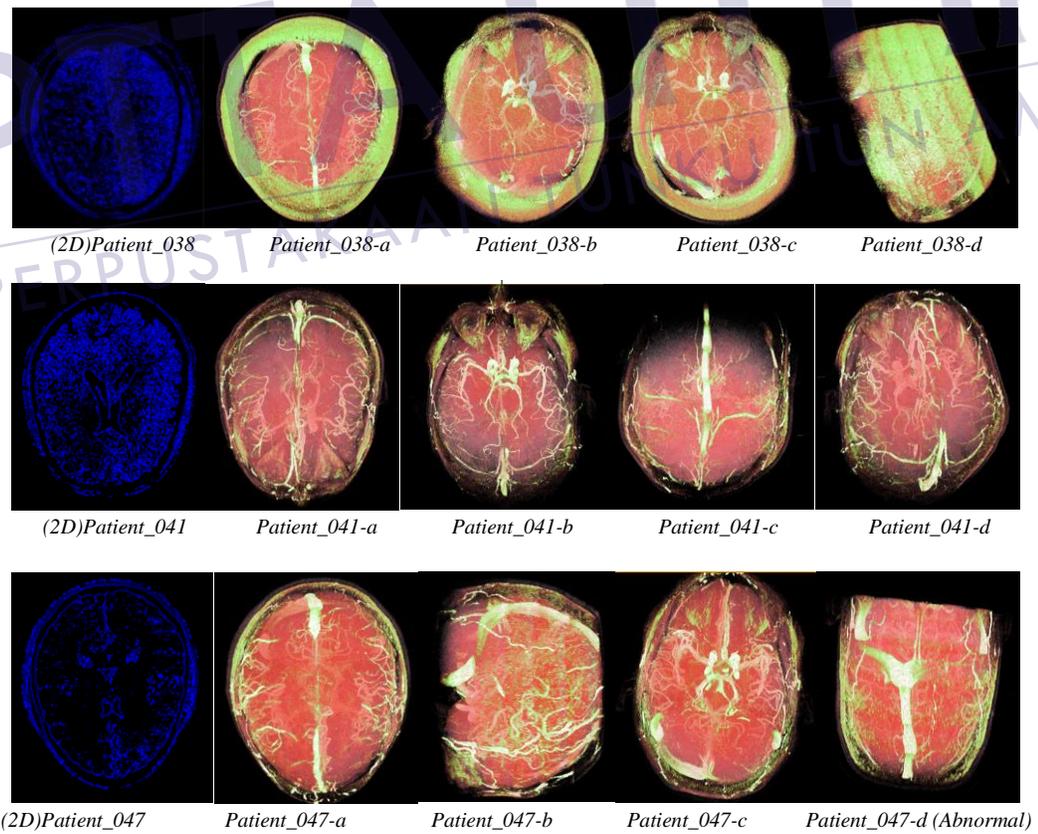


Figure 5.16: Randomly Selected Patient MRA Evaluation Datasets

SurLens is able to correctly map scalar values to opacity and able to clearly isolate distinctly those features within the volume. Despite the fact that all the datasets are un-segmented, the feature extractions and mappings are adequately handled by the technique. In Figure 5.17 and Figure 5.18, Patient_048, Patient_052 and Patient_054 are in 2-D while Patient_048-a to Patient_048-d, Patient_052-a to Patient_052-d and Patient_054-a to Patient_054-d are in 3-D structure where we can easily identify the internal features, edges, automatic mapping and isolation of abnormalities / tumor in the brain blood vessels of the datasets of Patient_048, Patient_052 and Patient_054.

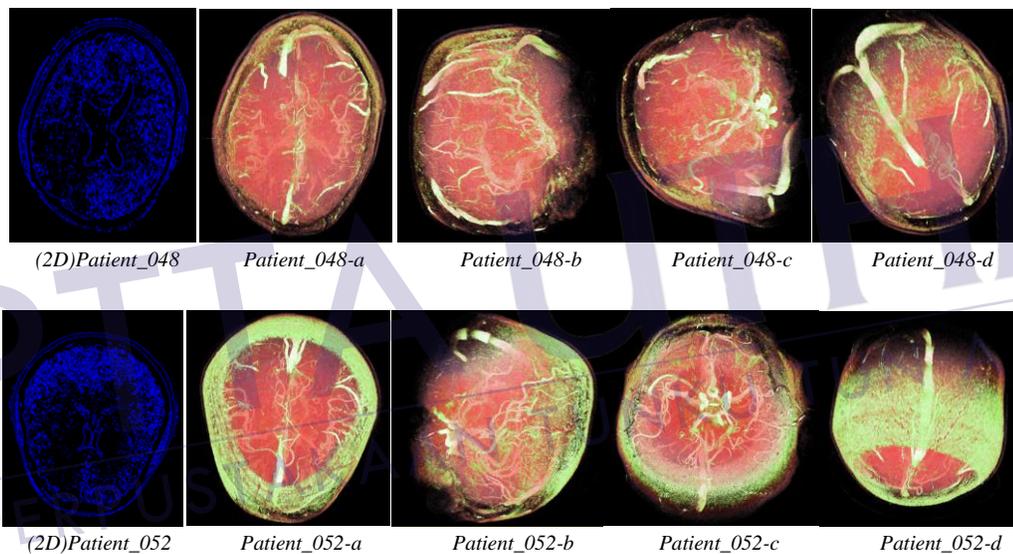


Figure 5.17: MRA Evaluation Datasets of Patient_048

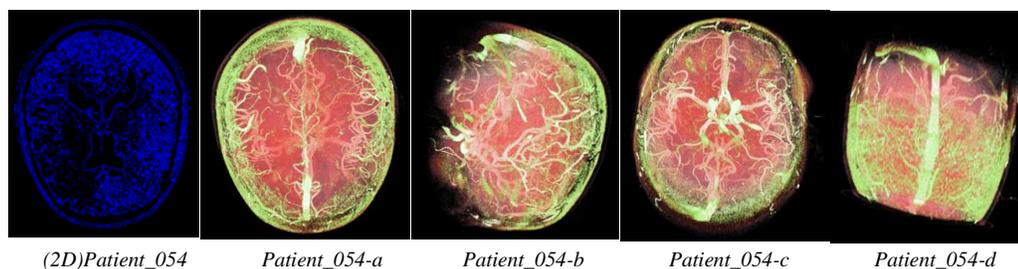


Figure 5.18: MRA Evaluation Datasets of Patient_052 and Patient_054

Segmentation is a large research field handling the aggregation of data regions before data labeling. Generally, datasets need to be segmented thoroughly in order to achieve good 3-D images. *SurLens* is able to bypass the time-consuming segmentation processes and still produce quality 3-D model of the evaluated datasets.

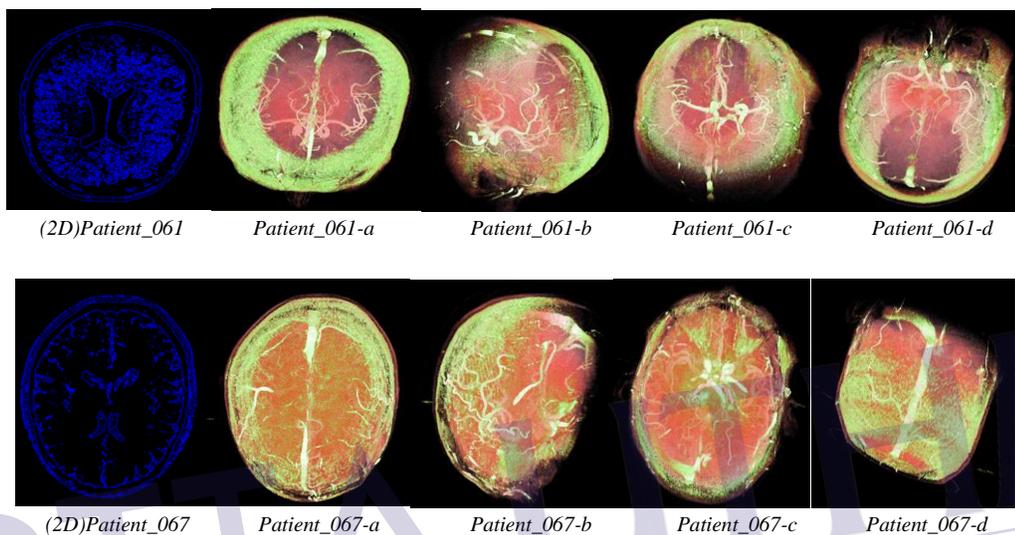


Figure 5.19: MRA Evaluation Datasets of Patient_061 and Patient_067

In Figure 5.19 and Figure 5.20, Patient_061 and Patient_067 are in 2-D while Patient_061-a to Patient_061-d and Patient_067-a to Patient_067-d in 3-D structure where we can easily identify the internal features, edges, automatic mapping and isolation of vascular abnormalities and tumor in the brain blood vessels of the datasets of Patient_061 and Patient_067.

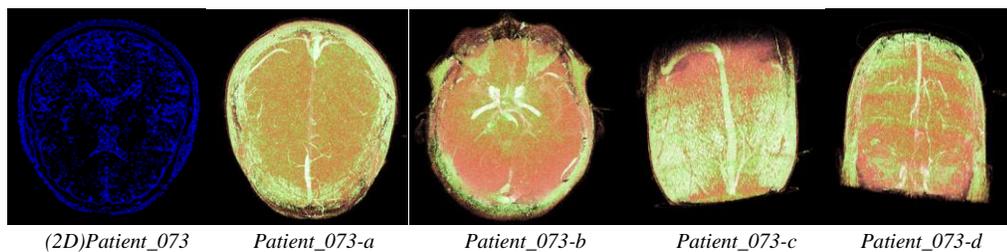


Figure 5.20-a: MRA Evaluation Datasets of Patient_073

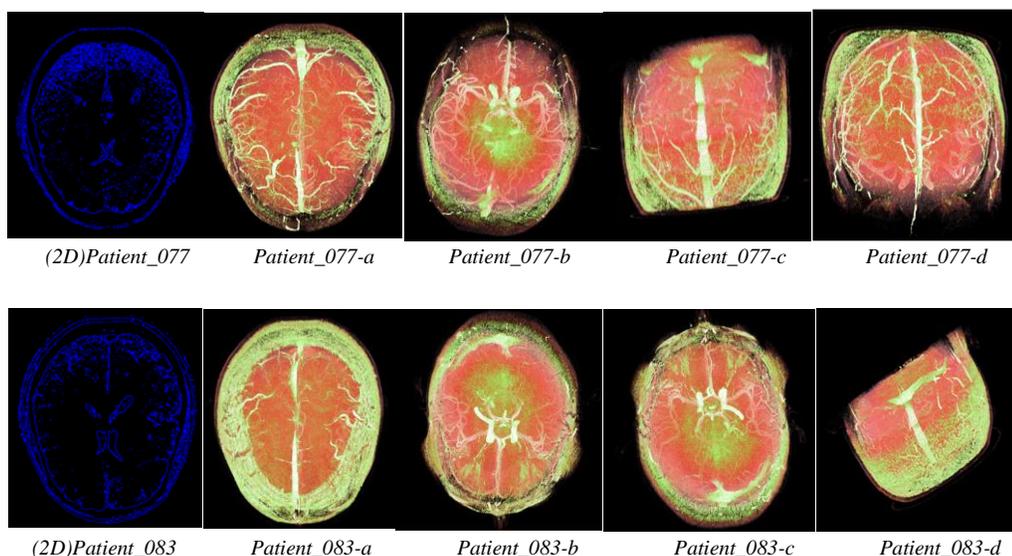


Figure 5.20-b: MRA Evaluation Datasets of Patient_077 and Patient_083

Localization's strength of *SurLens* in revealing and isolating abnormalities could be seen in some of the abnormal datasets such as in Figure 5.20-a and Figure 5.20-b. The Patient_073, Patient_077 and Patient_083 are in 2-D while Patient_073-a to Patient_073-d, Patient_077-a to Patient_077-d and Patient_083-a to Patient_083-d are in 3-D structure where we can easily identify the internal features, edges, automatic mapping and isolation of vascular abnormalities and tumor in the brain blood vessels of the datasets of Patient_073, Patient_077 and Patient_083. Such abnormalities are represented with varying scalar values through *SurLens* feature mapping schemes within a considerable interactive speed, as being documented in Table 5.1 to Table 5.4.

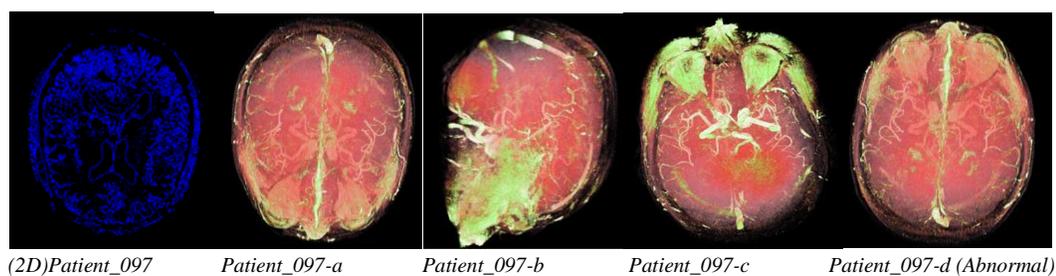


Figure 5.21: MRA Evaluation Datasets of Patient_097

Figure 5.21 and Figure 5.22, are the documentation of 2-D structure of Patient_097 and Patient_098 while Patient_097-a to Patient_097-d and Patient_098-a to Patient_098-d are in 3-D structure where we can easily identify the internal features, edges, automatic mapping and isolation of vascular abnormalities and tumor in the brain blood vessels of the datasets of Patient_097 and Patient_098.

According to the adjunct information that accompanied the datasets, Patient_097 is a patient with old tumor. The tumor is more conspicuous with Figure 5.21 (patient_097-c). *SurLens Visualization System* is able to clearly reveal and isolate the tumor and its deformation to the patient's brain in form of vessels blockages.

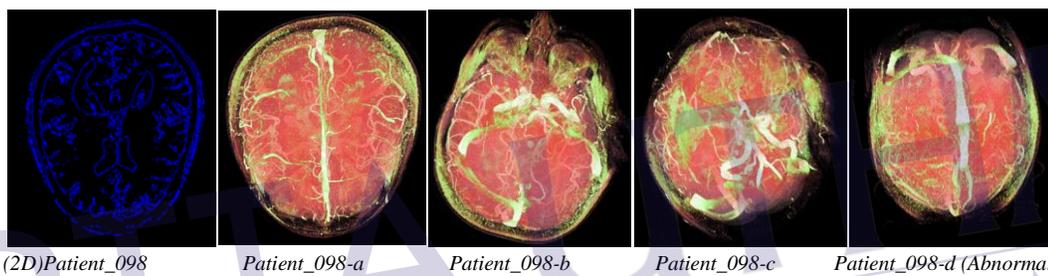


Figure 5.22: MRA Evaluation Datasets of Patient_098

In the same vein, Patient_098 was medically categorized abnormal. The positioning of the blood vessels in the 3-D model are quite different from other categorized normal patients even with patient of the same sex and relative age group. Medical professionals concerned would be able to read more meanings to these.

The subsections 3 of the first objective (to propose new approaches for brain volume visualization by introducing a new technique with automatic local feature mapping scheme that can isolate abnormalities / tumor and reveal internal features of brain blood vessels) and the second objective of this study, to implement the proposed approach, have been achieved with Magnetic Resonance Angiography (MRA) datasets.

5.3.2 Experimentation with DTI Datasets

Diffusion Tensor Imaging (DTI) is an MRI technology that enables imaging of the magnitude and directions of water diffusion with the human brain. Nerve fibres connecting the area of the brain may be affected by stroke, DTI can be used to estimate damages of the affected area by measuring its distance from other regions of the brain. With such measurements, appropriate stroke medications can be applied by medical professionals concerned. *SurLens* was evaluated with DTI datasets as in Figure 5.23 - Figure 5.26.

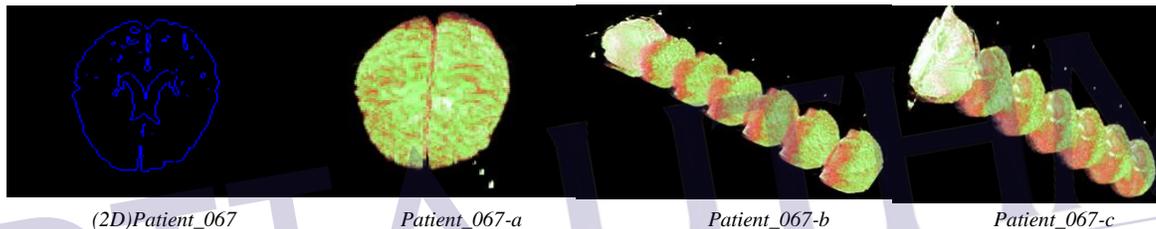


Figure 5.23: DTI-Patient_067-Normal, Female, 57yrs

In Figure 5.23, Patient_067 is in 2-D while Patient_067-a to Patient_067-c are in 3-D structure where we can easily identify the internal features, edges, automatic mapping and isolation of abnormalities in the DTI datasets of Patient_067.

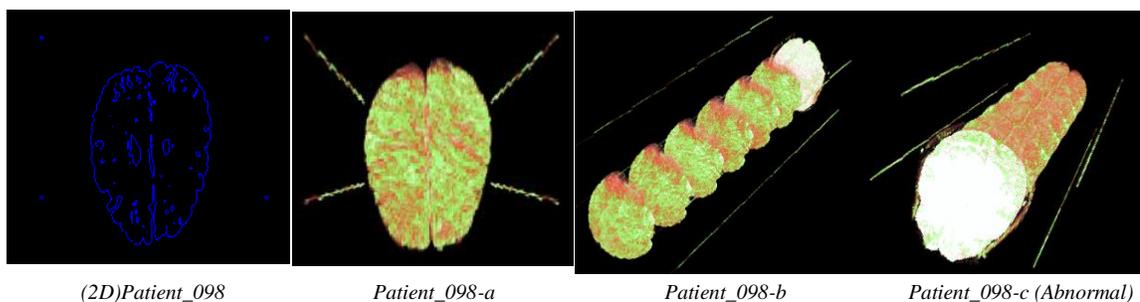


Figure 5.24: DTI-Patient_098-Abnormal, Female, 54yrs

In Figure 5.24 and Figure 5.25, Patient_098 and Patient_054 are in 2-D while Patient_098-a to Patient_098-c, and Patient_054-a to Patient_054-c are in 3-D structure where we can easily identify the internal features, edges, automatic mapping and isolation of abnormalities in the DTI datasets of Patient_098 and Patient_054.

It is greatly difficult to use 2-D images for medical analysis and decision procedures especially with some technologies like DTI, T1-FLASH and T1-MPRAGE that are mainly used for rapid acquisition in detriment to quality images. The 3-D results of this study have really proved the significance of 3-D structures in this regard. With the 2-D images, we are only able to see some features as dot(.), however, with the corresponding 3-D model of the same 2-D orientation, the features are better enhanced.

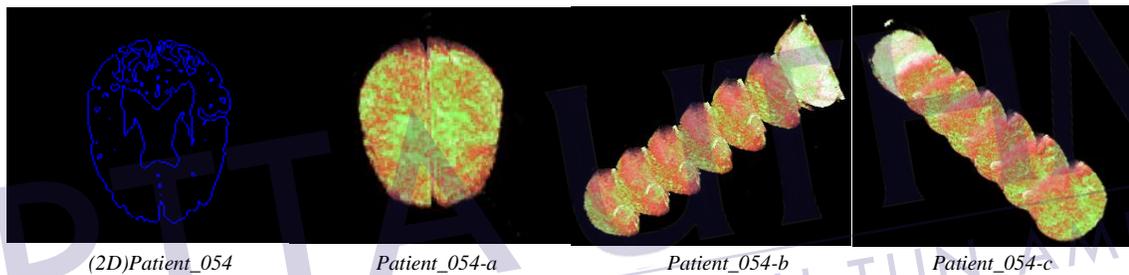


Figure 5.25: DTI-Patient_054-Normal, Female, 34yrs

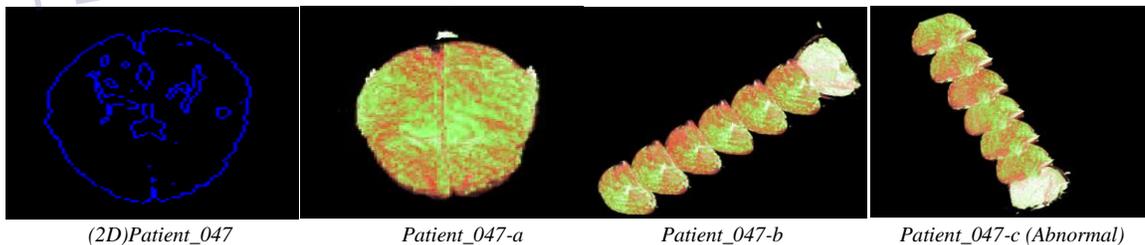


Figure 5.26: DTI-Patient_047-Abnormal, Female, 31yrs

In Figure 5.26, Patient_047 is in 2-D while Patient_047-a to Patient_047-c are in 3-D structure where we can easily identify the internal features, edges, automatic mapping and isolation of abnormalities in the DTI datasets of Patient_047.

Therefore, subsections 3 of the first objective and the second objective of this study have been achieved with Diffusion Tensor Imaging (DTI) datasets.

5.3.3 Experimentation with T1-FLASH Datasets

Fast Low Angle Shot (FLASH) is an MRI technology for achieving rapid imaging of the brain though with compromise on the image quality. Despite the fact that the raw 2-D images from the T1-FLASH images was of very low quality, *SurLens* was able to significantly reconstruct the images into 3-D model. Some of the evaluation results of the experimentation of *SurLens* Visualization System with the T1-FLASH datasets are presented in Figure 5.27 - Figure 5.29. In Figure 5.27, Patient_067 is in 2-D structure while Patient_067-a, Patient_067-b and Patient_067-c are in 3-D structure where we can easily identify the internal features, edges, automatic mapping and isolation of abnormalities / tumor in the T1-FLASH datasets of Patient_067.

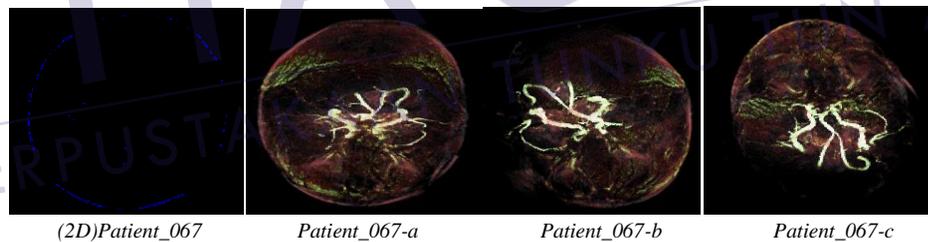


Figure 5.27: T1-FLASH-Patient_067-Normal, Female, 57yrs

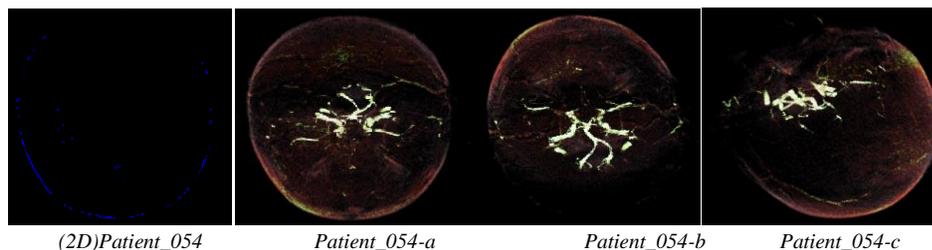


Figure 5.28: T1-FLASH-Patient_054-Normal, Female, 34yrs

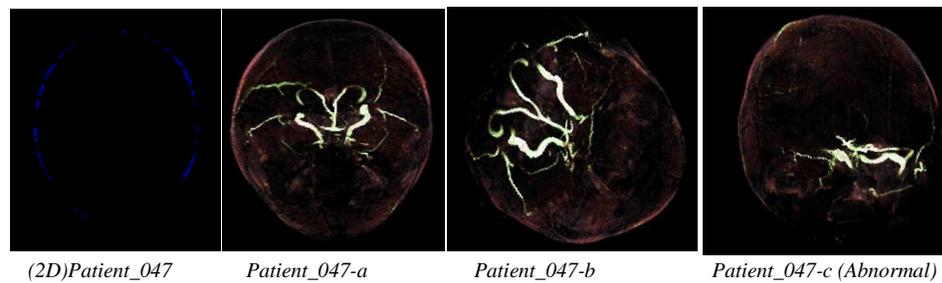


Figure 5.29: T1-FLASH-Patient_047-Abnormal, Female, 31 yrs

In Figure 5.28 and Figure 5.29, Patient_054 and Patient_047 are in 2-D while Patient_054-a to Patient_054-c and Patient_047-a to Patient_047-c are in 3-D structure where we can easily identify the internal features, edges, automatic mapping and isolation of abnormalities in the T1-FLASH datasets of Patient_054 and Patient_047.

The *stripped FLASH* imaging of stripped cerebrum was reconstructed using *SurLens* Visualization System. In cases of abnormal patients, abnormalities / tumor around the datasets can easily be isolated and revealed for necessary medical procedures. The datasets in Figure 5.30 is a *stripped FLASH* of a normal patient.

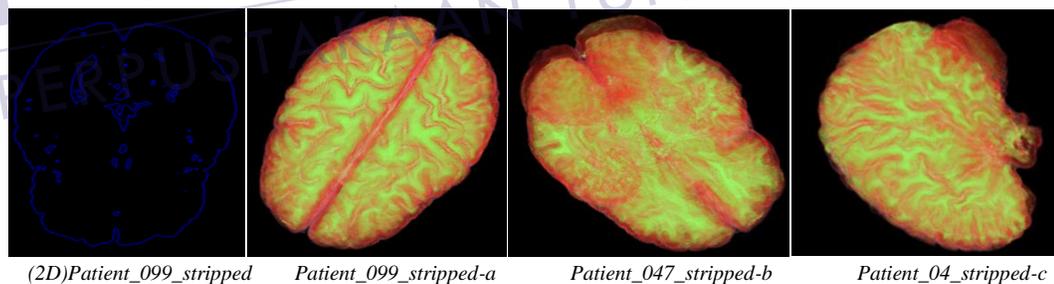


Figure 5.30: Patient_099-Stripped-FLASH (Normal Patient)

The Patient_099_stripped dataset is in 2-D in Figure 5.30 while Patient_099_stripped-a to Patient_099_stripped-c are in 3-D structure where we can easily identify the internal features, edges, automatic mapping and isolation of abnormalities in the stripped FLASH datasets of Patient_099-stripped.

The above results show that the subsections 3 of the first objective as well as the second objective of this study have been achieved with *T1-Fast Low Angle Shot (T1-FLASH)* datasets.

5.3.4 Experimentation with T1-MPRAGE Datasets

Magnetization Prepared Rapid Gradient Echo (MPRAGE) is an MRI technology mainly designed to detect metastatic brain tumors. Metastatic brain tumors are malignant tumor that are within certain part of the body and travels to the brain. Such cancerous tumors could be primarily located in areas such as the kidney, the lungs, the breast and later spread to the brain. *SurLens* experimentations were extended to involve its evaluation with a number of *T1-MPRAGE* datasets of patients. Some of the evaluation results of *SurLens* with *T1-MPRAGE* datasets are presented in Figure 5.31 - Figure 5.33.

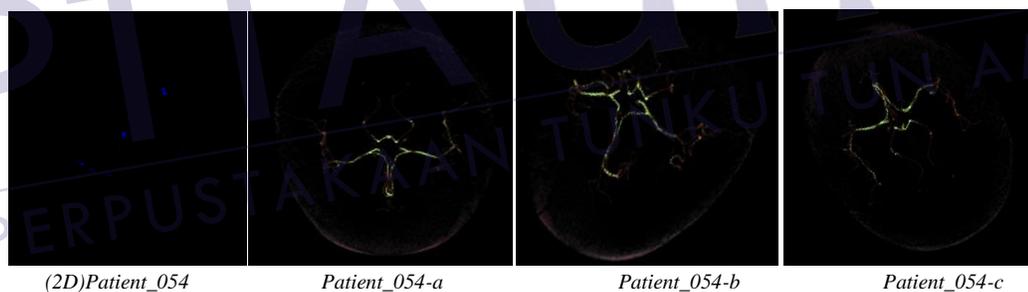


Figure 5.31: T1-MPRAGE-Patient_054, Normal, Female, 34yrs

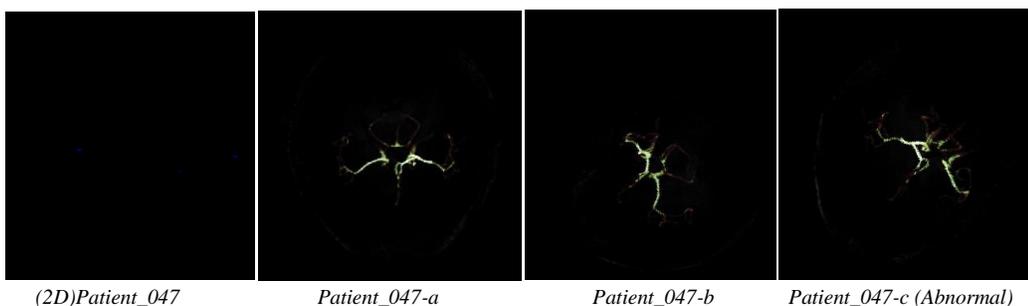


Figure 5.32: T1-MPRAGE-Patient_047, Abnormal, Female, 34yrs

T1-MPRAGE is usually taken when a rapid acquisition is required and it is at the expense of quality images. The quality of the acquired 2-D images are so poor to the extent that it's almost invisible to the normal human vision; medical practitioners concerned might need some reading aids to be able to use the 2-Ds. However, with *SurLens Visualization System*, the 2-Ds were interpreted into 3-D model as presented in Figure 5.31 – Figure 5.33. The 3-D results obtained from *SurLens Visualization System* are presented in the later sections of this chapter for comparison with other two (2) notable visualization systems. In Figure 5.31 and Figure 5.32, Patient_054 and Patient_047 are in 2-D while Patient_054-a to Patient_054-c and Patient_047-a to Patient_047-c are in 3-D structure where we can easily identify the internal features, edges, automatic mapping and isolation of abnormalities in the T1-MPRAGE datasets of Patient_054 and Patient_047.

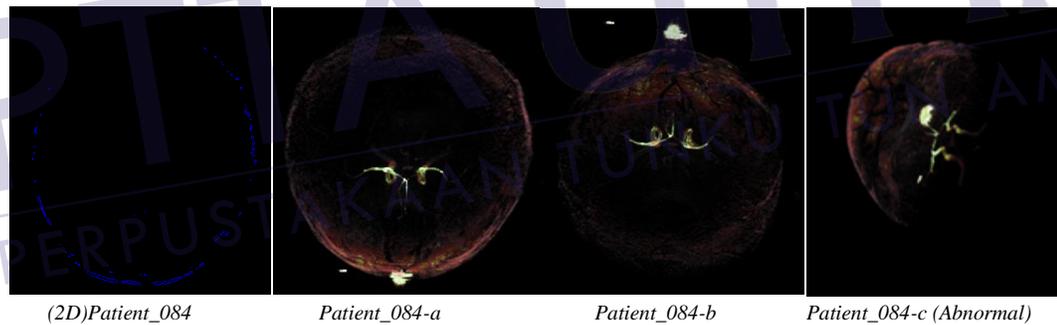


Figure 5.33: T1-MPRAGE-Patient_084, Abnormal, Female, 67yrs

Patient_084 is in 2-D in Figure 5.33 while Patient _084-a to Patient_084-c are in 3-D structure where we can easily identify the internal features, edges, automatic mapping and isolation of abnormalities in the T1-MPRAGE datasets of Patient_084.

In the same vein, successful mapping of the presented *T1-MPRAGE* datasets show that the objectives of this study, the subsection 2 of the first objective and the second objective of this study have been achieved with *T1- Magnetization Prepared Rapid Gradient Echo (T1-MPRAGE)* datasets.

5.4 *SurLens* Comparison with Other Visualization Systems

The focus of this study is on image quality and performances in terms of rendering speed. However, it is difficult to evaluate image quality; often researchers simply put images of competing algorithms side by side, appointing the human visual system (HSV) to be the judge (Meißner et al., 2000). The HSV has been confirmed to be less sensitive to some errors (stochastic noise) and more others (regular patterns), even sometimes images with larger numerical errors are judged as worse by a human observer than images with lower numerical errors (Teo & Heeger, 1994). Hence, two notable visualization systems were chosen as bench of comparison, the *VolView Visualization System* and *ParaView Visualization System*. These two visualization systems were chosen because they are well known systems, likewise developed on top of the visualization toolkit (VTK) libraries similar to the development of *SurLens Visualization System*. The following sections discuss *ParaView* and *VolView* Visualization Frameworks.

5.4.1 *ParaView* & *VolView* Visualization System

Visualization Toolkit (VTK), developed by Kitware Inc., United States, consists of a set of libraries supporting image processing, Computer Graphics and Visualization while the *ParaView* and *VolView* are the applications developed for visualizing 3-D datasets, built on top of the VTK libraries.

The development of an application for visualizing 3-D datasets named “*ParaView*” began in the year 2000 (Moreland, 2008) as a collaborative effort between Kitware Inc., and the three national laboratories in the United States, the Los Alamos National Laboratories, the Sandia National Laboratories and the Livermore National Laboratories. The first outcome of *ParaView* research developments was released in 2002 (Kitware News, October 10, 2002). Since then, the research developments have been extended to include the United States Army Research Laboratory, CSimSoft company and various other academic and government institutions in the United States.



PTTA UTHM
PERPUSTAKAAN TUN AMINAH

ParaView (Martin et al., 2010) is a very powerful visualization tool used in the Department of Defense (DoD) high performance computing (HPC) community. The development of ParaView still continues till today with major funding being provided by Sandia National Laboratories, the US Department of Energy through Los Alamos National Laboratories and the US National Science Foundation.

VolView is principally owned and developed by Kitware Inc., as their premiere 3-D application built on top of VTK. Its development was initiated before the year 1999 with the motive of developing an intuitive and interactive system for volume visualization. According to Rick Avila, Senior Director of Health Care Solutions at Kitware “*The 3-D imaging and segmentation capabilities of VolView are among the most advanced in the world*”. Although Rick Avila believes, *VolView* has great research and real-world medical value but being not approved by the Federal Drug Administration (FDA) restricted it from use in the medical community. In order to improve *VolView* capabilities and to meet an acceptable standard for use in clinical settings, Kitware recently offers researchers a free global access to the application (Benzinga News, November 29, 2010).

In accordance with the objectives of *SurLens*, focusing on image quality and performances (in terms of rendering speed), the results of this study were compared with *VolView* and *ParaView*. Our comparison was based on the open (demo) versions of *VolView* and *ParaView*. The three systems (*SurLens*, *VolView* and *ParaView*) were compared using their default parameter /configuration settings. The comparison were carried out using *MRA*, *DTI*, *Stripped FLASH* and *T1-MPRAGE* datasets. The results of the comparison have proven *SurLens* the best Visualization System in terms of image quality. A divert comparison test of *SurLens*, *VolView* and *ParaView* with CT Abdominal Pelvic datasets has likewise proven *SurLens* to be a future promising multimodal visualization system.



PTIT UNIVERSITAS
PERPUSTAKAAN TUNJUNG AMINAH

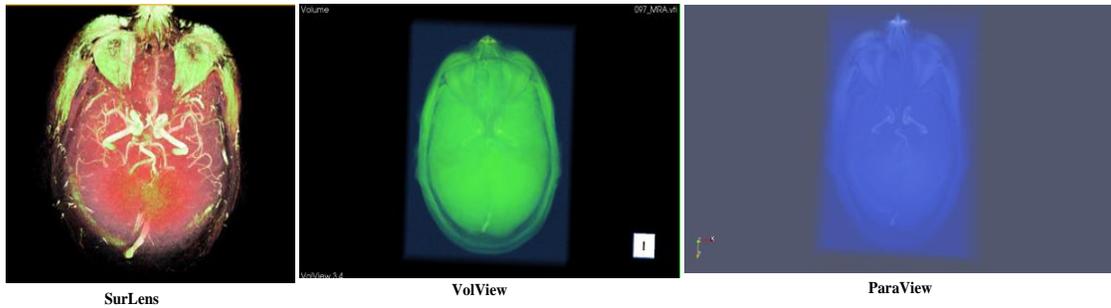


Figure 5.34: Comparison –Patient_097-MRA Datasets

Figure 5.34 and Figure 5.35 show the image quality comparison among *SurLens*, *VolView* and *ParaView* with MRA datasets. With *SurLens*, the vessels in the dataset can be seen clearly; there exist clear isolation of the tumor and feature separations within the datasets unlike with *VolView* and *ParaView* 3-D model results, which could only be faintly seen. Infact, *VolView* and *ParaView* cannot be used without prior thorough segmentation of datasets, while *SurLens* achieved all the 3-D models without any prior segmentation stage.

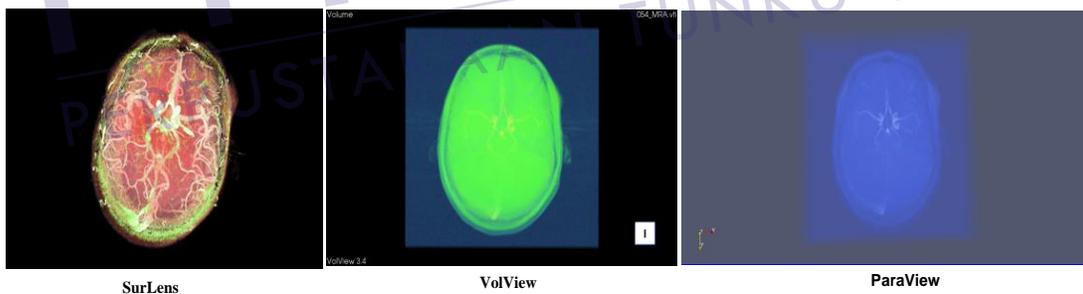


Figure 5.35: Comparison – Patient_054-MRA Datasets

Visualization of DTI datasets also require good feature detection algorithm to see the tiny features in the datasets. Diffusion Tensor Imaging datasets are used in studying the directionality and magnitude of water diffusion. *SurLens* is far better in features detection, contributing to the production of quality images compared to *VolView* and *ParaView*. The result of the comparison is shown in Figure 5.36.

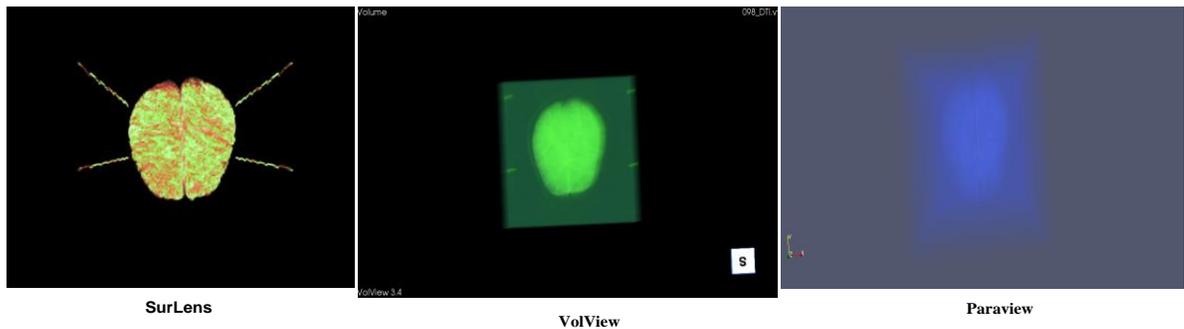


Figure 5.36: Comparison - Patient_098-DTI Datasets

T1-MPRAGE is a technique that focuses on non-conspicuous features in the datasets. Those features require very good algorithms to reveal. *SurLens* had best results in comparison with *ParaView* and *VolView* as shown in Figure 5.37.

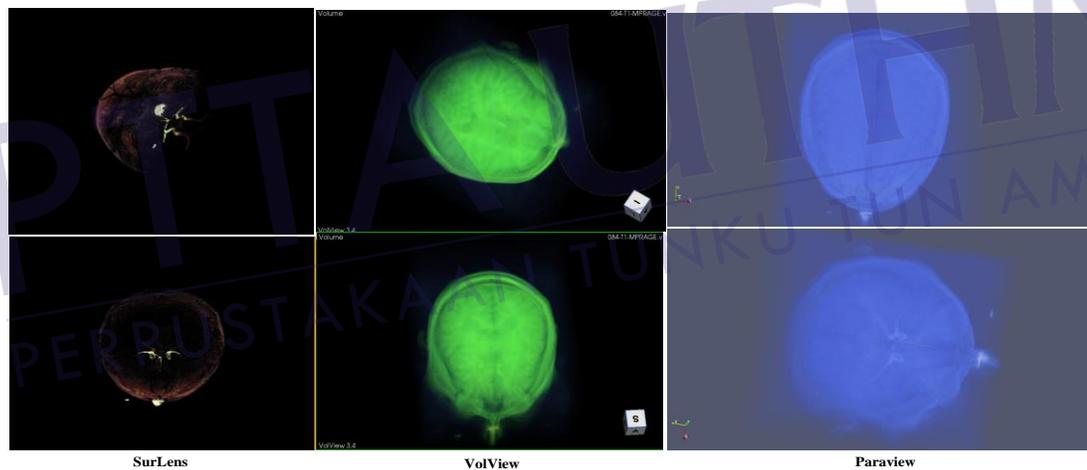


Figure 5.37: Comparison - Patient_084-T1-MPRAGE Datasets

In some circumstances, there could be a need to identify any abnormalities in the cerebrum. The cerebrum is the portion of the brain that performs the motor functions, the sensory functions and other mental activities. The cerebrum is in four lobes (the frontal, temporal, parietal and occipital lobes), each being separated by deep grooves referred to as fissures. With a clear 3-D view of the 2-D datasets, abnormalities in any of the four lobes could be identified. *SurLens* comparison experiments' with *Volview* and *ParaView* is presented in Figure 5.38.

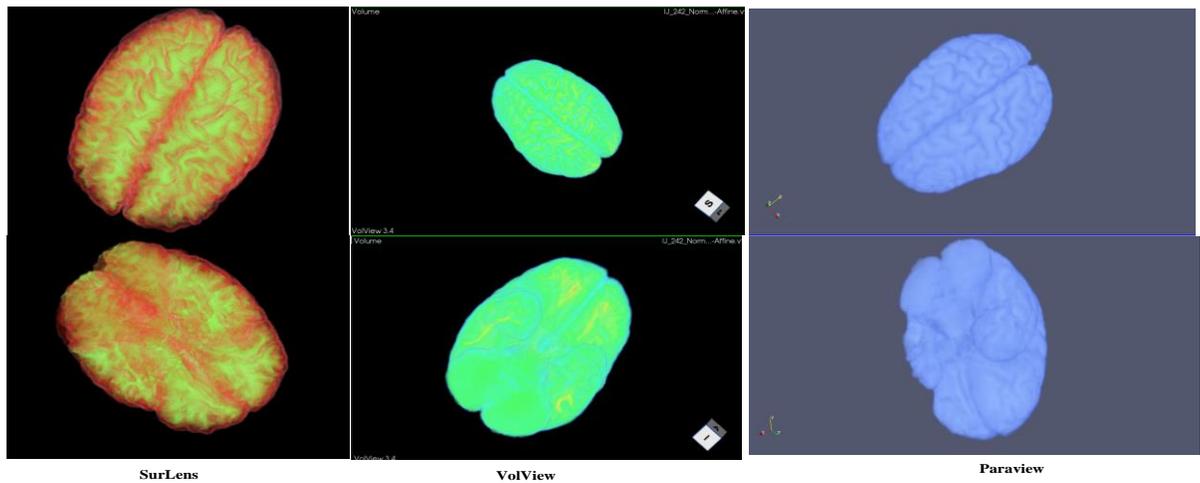


Figure 5.38: Comparison - Patient_099-Stripped_FLASH Datasets

We engaged *SurLens* in a divert comparison with un-segmented Computed Tomography (CT) datasets from Uppsala Multidisciplinary Centre for Advanced Computational Science. Usually 2-D datasets must be properly segmented before visualizing into 3-D structure. This is the case with *ParaView* and *VolView Visualization System* which have been identified by the leading imaging company, Kitware, of having the best approach for volume visualization. Our divert experiments showed that *SurLens* can handle such un-segmented CT data clearly unlike *ParaView* and *Volview*. These results of the comparison are presented in Figure 5.39.

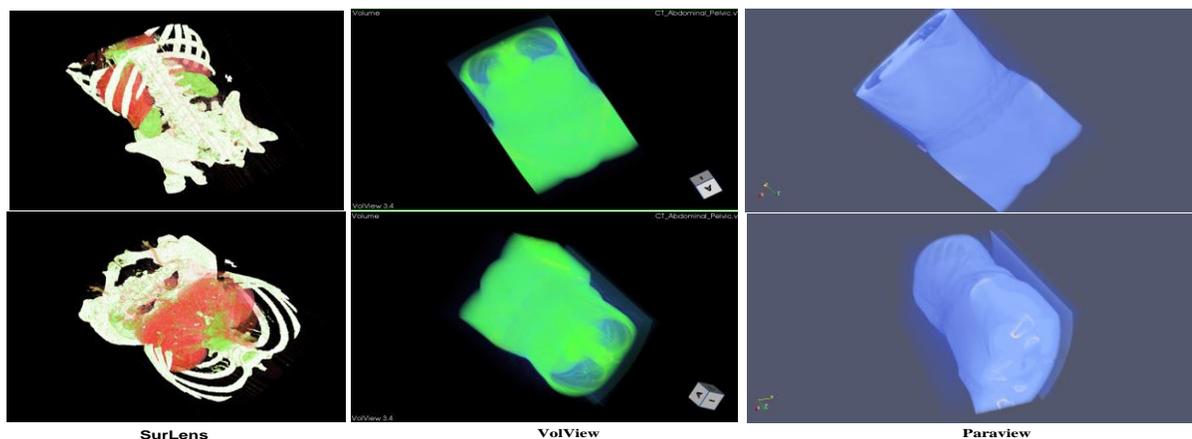


Figure 5.39: Divert Comparison - Patient_01_CT_Abdominal Pelvic Datasets

One of the objectives of this study is to compare the volume visualization results of our newly proposed, designed and implemented visualization framework named *SurLens* with the results of existing approaches. Therefore, based on the above comparison results presented, objective 3 of this study has been achieved.

5.5 Results of Robust Algorithm for Mass Data

Robustness is the ability of algorithm to handle errors during execution. In proving robustness of a volume visualization approach, varying sizes of volume datasets are usually employed in experimental testing and the performances of the algorithms or approach in terms of processing speed are recorded (Shen & Johnson, 1994). Hence, one of the objectives of this study is to design and implement new algorithms within the framework that are robust enough to handle mass data within a considerable interactive speed, extensive application interoperability and at a lower resulting cost. This objective can be assessed using speed and processing performances of our newly designed and implemented algorithms with different types of data sizes. Therefore, we evaluate the performances of our algorithm using different sizes of MRA, DTI, T1-MPRAGE and T1-FLASH datasets and the results are documented in the following sections.

5.5.1 Speed Evaluation with MRA Datasets

The speed performances of *SurLens* were evaluated with MRA datasets. The experiments were carried out using the three (3) study testbeds. With the evaluation

Table 5.1: MRA Speed Evaluation for Datasets of Patient_001 to Patient_009

<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 -1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
001_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.8000001	09.70s	05.10s	02.95s
002_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.8000003	09.71s	05.11s	02.96s
003_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.8	09.71s	05.11s	02.96s
004_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.8	09.71s	05.11s	02.96s
005_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.8	09.71s	05.11s	02.96s
006_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800004	09.72s	05.13s	02.97s
007_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799998	09.71s	05.11s	02.96s
008_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799998	09.71s	05.11s	02.96s
009_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799998	09.71s	05.11s	02.96s

results, it was observed that some abnormalities might increase the size of the brain structures while some might reduce or drastically reduce sizes of some features, hence reducing the entire size of the brain. The analogue for this is in the case of swollen wounds or drained wounds. Such effects of abnormalities in datasets would require

identifying the nature of the abnormalities involved and subsequently measuring its level, these might require some clinical investigations with certain standard medical procedures that are beyond the scope of this study. Table 5.1 to Table 5.4 shows the speed performance evaluation of *SurLens* with variable sizes of MRA datasets.

Table 5.2: MRA Speed Evaluation for Datasets of Patient_010 to Patient_015

<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - GB)</i>
<i>010_MRA</i>	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799995	09.70s	05.10s	02.95s
<i>011_MRA</i>	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799999	09.71s	05.11s	02.96s
<i>012_MRA</i>	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.8	09.71s	05.11s	02.96s
<i>013_MRA</i>	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799997	09.70s	05.11s	02.96s
<i>014_MRA</i>	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.8	09.70s	05.10s	02.95s
<i>015_MRA</i>	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799999	09.71s	05.11s	02.96s

According to the study conducted by Bullitt et al. (2010), aging of the adult human brain is known to be associated with a variety of anatomical tissue changes.

Aging may induce progressive shrinkage of grey matter volume (Courchesne et al., 2000), it may affects and may lead to loss of myelinated axons (Marner et al., 2003). Meanwhile in certain cases, an adult brain may exhibits characteristics similar to a much younger individual (Creasey, 2003).

Table 5.3: MRA Speed Evaluation for Datasets of Patient_016 to Patient_020

<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 -1 GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
<i>016_MRA</i>	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.8000001	09.70s	05.10s	02.95s
<i>017_MRA</i>	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799998	09.70s	05.11s	02.96s
<i>018_MRA</i>	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799997	09.71s	05.11s	02.96s
<i>019_MRA</i>	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800002	09.70s	05.10s	02.95s
<i>020_MRA</i>	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800004	09.73s	05.13s	02.97s

One of the factors influencing the processing time of the datasets is Voxel Spacing. If the voxels are spacious, algorithms might do more work in bringing them together before actualizing the rendering. However, in certain cases, other factors such as level of pixels, accumulation of points might possibly overtake the voxel spacing factor, hence influencing the processing time of the datasets.

In the same vein, presence and/or nature of abnormalities in the datasets is a factor to take into consideration while looking into the variations in the rendering processing time of medical datasets.

As *SurLens* was designed to isolate the abnormalities in the dataset based on different intensity or density values, this might likewise take considerable time when considering the dataset of a normal patient and abnormal patient. In the case of a normal datasets, the algorithms will save more time in mapping of varying values for the

Table 5.4: Randomly Selected Patients' MRA Datasets for Speed Evaluation

<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 -1 GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
027_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799998	09.71s	05.11s	02.96s
038_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799995	09.70s	05.10s	02.95s
041_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800002	09.70s	05.10s	02.95s
048_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799998	09.71s	05.11s	02.95s
052_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.8	09.71s	05.11s	02.96s
061_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.8	09.71s	05.11s	02.96s
073_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799999	09.71s	05.11s	02.96
077_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800002	09.70s	05.10s	02.95s
083_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800002	09.70s	05.10s	02.95s
097_MRA	176	352 x 448 x 176 voxels	0.513393 x 0.513393 x 0.800003	10.43s	05.18s	03.10s

isolation of possible abnormal features in the dataset, unlike the case of datasets with abnormalities. However, such factors as intense of the abnormalities, diverse positioning of abnormalities, the nature of the abnormalities, are also factors that might reduce or increase the rendering processing time of the datasets.

However, this study only focuses on achieving immediate reconstruction and obvious mappings of the internal features of human brain, we only compare variations in

rendering-processing time with the three (3) testbeds. Other factors which might either increase or decrease the rendering processing time such as level of pixels in each of the datasets, accumulation of points, nature and intense of abnormalities are beyond the scope of this study and they are therefore not considered in the evaluation.

Therefore, the results proved the robustness of the designed and implemented mass data algorithms with varying data sizes of Magnetic Resonance Imaging (MRA) datasets.

5.5.2 Speed Evaluation with DTI Datasets

SurLens was evaluated for speed performances with *Diffusion Tensor Imaging (DTI)*. Most of the DTI datasets used for the evaluation contains 322 images each, making up

Table 5.5: Speed Evaluation for Selected Patients' DTI Datasets

Patient's Dataset	No of Images	Voxel Dimension	Voxel Spacing	CPU - 3GB	GPU (GT520 -1GB)	GPU (Quadro 600 - 2GB)
067_DTI	322	128x128x322 voxels	2x2x2	04.06s	02.26s	01.43s
098_DTI	322	128x128x322 voxels	2x2x2	04.06s	02.14s	01.46s
054_DTI	322	128x128x322 voxels	2x2x2	04.06s	02.16s	01.56s
047_DTI	322	128x128x322 voxels	2x2x2	04.06s	02.02s	01.42s

128 x 128 x 322 voxels within 2 x 2 x 2 spacing. We evaluated the speed performances of *SurLens* with around 109 of such DTI datasets and some are presented for explanatory purposes in this chapter while others are attached as appendix. Despite the fact that most of the datasets in Table 5.5 are of the same number of images, voxel

dimensions and spacing, they show variance in processing time with CPU and other GPUs used in the experiment. However, the main experimental testbed, GPU Quadro, produced better results though with variations in different datasets as a results of such factors as number of pixels, accumulation of points, nature of abnormalities and intense of abnormalities in the datasets.

The results show the designed and implemented algorithms are robust enough to handle mass datasets of Diffusion Tensor Imaging (DTI) within an interactive speed.

5.5.3 Speed Evaluation with T1-FLASH Datasets

The three (3) experimental testbeds are likewise utilized in the evaluation of *SurLens* speed performances with *T1-FLASH* datasets. As earlier discussed, though some of the datasets consist of the same number of images, we still observed variations in the datasets' processing time with the best results from our Quadro testbed. Some of the results are presented in Table 5.6 while others are attached as appendix.

Table 5.6: Speed Evaluation for Selected Patients' T1-FLASH Datasets

Patient's Dataset	No of Images	Voxel Dimension	Voxel Spacing	CPU -3GB	GPU (GT52 0 -1G)	GPU (Quadro 600 - 2 GB)
067_T1-FLASH	176	176x256x176voxels	1x1x1	04.50s	02.39s	01.62s
054_T1-FLASH	176	176x256x176voxels	1x1x0.99 9996	04.84s	02.41s	01.83s
099_Stripped_FLASH	176	176x256x176voxels	1x1x1	04.57s	02.07s	01.61s
047_T1-FLASH	176	176x256x176voxels	1x1x0.99 9999	04.61s	02.37s	01.79s

Therefore, speed performance evaluation of the designed and implemented algorithms shows their robustness with *T1-FLASH* datasets.

5.5.4 Speed Evaluation with T1-MPRAGE Datasets

Speed performances of *SurLens* Visualization System with *T1-MPRAGE* were obtained using *SurLens* experimental testbeds. A number of the results are presented in Table 5.7 while others are attached as appendix.

Table 5.7: Speed Evaluation for Selected Patients' T1-MPRAGE Datasets

Patient's Dataset	No of Images	Voxel Dimension	Voxel Spacing	CPU -3GB	GPU (GT520 -1G)	GPU (Quadro 600 - 2GB)
054_T1-MPRAGE	128	208x256x128 voxels	1x1x1	05.16s	02.61s	01.73s
047_T1-MPRAGE	128	208x256x128 voxels	1x1x0.999999	04.70s	02.33s	01.73s
084_T1-MPRAGE	160	208x256x160 voxels	1x1x1	05.19s	02.36s	01.74s

T1-MPRAGE datasets are usually light weighted in comparison with most of the imaging techniques. Nevertheless, the speed evaluation of the newly designed and implemented algorithms for this study with T1-MPRAGE confirms the algorithms are robust enough to handle mass data of T1-MPRAGE.

Subsection four (4) of the objective one (1) and the objective two (2) of this study is to propose, design and implement new algorithms within the framework that are robust enough to handle mass data within a considerable interactive speed, extensive

application interoperability and at a lower resulting cost. The above presented results show that, those objectives have been achieved. In the same vein, two (2) independent experiments were setup sideways with Quadro GPU (the main experimental testbed), using CPU and 1GB of GPU as indicated in the results above. That was to evaluate the results of the proposed, designed and implemented new robust algorithms for mass data. The comparison results presented for the performance comparison results' of the framework in terms of rendering speed is therefore a part of the accomplishment of the objective three (3) highlighted for this study. However, the framework comparison results in terms of image quality as being presented in the previous sections of this chapter affirms the accomplishment of the other half of the objective three (3) highlighted for this study.

5.6 Summary

This study has concentrated on producing quality images at a cheaper cost for immediate medical diagnosis. The implemented system named “*SurLens*” (short for Surgical Lens) has its core strength in its ability to clearly reveal the detailed internal features of human brain blood vessels, which could be very resourceful in assisting the medical practitioners concerned for immediate location of blockages within the human brain blood vessels. *SurLens Visualization System* has been tested and found resourceful in the reconstruction of other specialized MR techniques; the *DTI*, the *T1-FLASH*, the *T2-Weighted*, and the *TI-MPRAGE* of human brain.

SurLens applies feature extraction in the highlighting of data; it allocates transparency based on scalar values and assigns the transparency based on localized gradient magnitude for edge detection within volume. It is robust enough to handle mass data within a considerable interactive speed. Experimentations with *SurLens* recorded rendering time of less than 2 seconds for most of the datasets containing up to 322 images and not more than 3 seconds with some datasets, which could possibly be as a results of variations in the quantity of pixels, accumulation of pixels to certain region,

voxel spacing, the presence and nature of abnormalities in the datasets. Clear investigation of such factors is beyond the scope of this study.

SurLens Visualization System is able to depict gradual transition within dataset samples, exhibiting visualization with respect to data relative *density or intensity*. *SurLens* can isolate tumor and other abnormalities in the experimented datasets using *opacity* and *colour transfer functions*. *SurLens* can show *directionality* as well as *magnitude of water diffusion* using *DTI* dataset samples. It can correctly map scalar values to opacity and able to clearly isolate distinctly those features within the volume. With all the experimentations on *SurLens*, segmentation might not be necessarily required before using *SurLens* for visualization.

With respect to the objectives of *SurLens Visualization System*, it has been dully compared with the two (2) notable visualization systems that were also built on top of visualization toolkit libraries, the *ParaView* and the *VolView*. *SurLens* produced the best quality of images among the three systems.

Our divert experiments and comparison with un-segmented abdominal pelvic CT datasets have proven *SurLens* a future 3-D Multimodal Visualization System. Without going through the usual time-consuming segmentation processes, *SurLens* can clearly reveal the internal features of the un-segmented datasets and isolate any abnormality / tumor therein.

SurLens was developed on .NET framework for good interoperability with other emerging revolutionary technology; this offers *SurLens* a promising and forthcoming future in the world of Computer-Assisted Diagnosis Therapy.



PTTA ALITHAM
PERPUSTAKAAN TUNJUN AMINAH

CHAPTER 6

CONCLUSION

This chapter concludes the entire study. Major contributions of the study are summarized in this chapter. Identified future tasks to extend this study are likewise presented.

6.1 INTRODUCTION

The human brain, all the tissues and the organs of the human anatomical structures are in 3-D but imaging devices can only produce them in 2-D slices. The medical professionals concerned have to work with the 2-D slices that lack depth information. Diagnosis procedures with the scanned images have to be achieved manually, which is time consuming, tedious and prone to errors. These slow and highly inaccurate procedures endanger lives of patients especially when dealing with the usual numerous 2-D image slices produced by medical imaging modalities.

The rapid development in information technology has immensely contributed to the use of medical visualization, volume graphics and GPU computing as modern approaches for visualizing medical volumetric data. Rapid evolvement of such domains has created good environment for contribution into Computational Sciences and Computational Life Sciences in medicine. However, the existence of standard surface modeling has only assisted in defining the opaque surface of objects; this hinders us

from seeing the inner part of the object. With volume visualization, we can make the boundaries of the object transparent and hence the inner part would become visible.

Some of the features in the brain are quite tiny, such as brain blood vessels, hence a resourceful framework, integrated with active schemes, techniques and algorithms is required to successfully detect, map and isolate features. Such algorithms supporting the framework must be robust enough to handle mass data without any compromise on the accuracy of the results in terms of the produced quality 3-D outputs. Depending on the level of abnormalities in the brain datasets, intensity of normal and abnormal features in patients' datasets may have close intensity proximity. Only active framework can distinctly detects, maps and isolates such brain abnormalities.

Medical diagnosis and disease therapy procedures need to be achieved on time. A computer-aided brain medical diagnosis system that could effectively serve its purpose must be able to achieve not only fast generation of 3-D model of datasets but also achieve the entire streams of datasets' processing within an interactive speed, robust enough to handle mass datasets. This study is a contribution to Computational Life Sciences, an interdisciplinary research domain within Computational Science & Engineering.

6.2 CONCLUSION

Human brain is a complex system that controls all the activities of the human anatomical and physiological system. It contains billions of nerves and connections, each being responsible for handling one task or another within a fraction of a second. However, a number of abnormalities could infect the brain, which usually requires clinical diagnoses and therapy procedures. The brain is protected by skull, which in turns made up of many slices. In order to diagnose a patient with any of the brain abnormalities, suspected areas or portions of the infection are to be studied by the radiologists and surgeons. This involves acquiring the medical scans of the brain.

Magnetic Resonance Imaging (MRI) is one of the special imaging techniques that can penetrate the bones of the skull and image the brain. Hence, MRI is the best image



PTTAAUTHM
PERPUSTAKAAN TUNKU TUN AMNAH

modalities for imaging the brain and other soft tissues in the human anatomical structures. This study employs the specialized *Magnetic Resonance Angiography (MRA)*, the *Diffusion Tensor Imaging (DTI)*, *T1-Fast Low Angle Shot Magnetic Resonance (T1-FLASH)* and *T1-Magnetization Prepared Rapid Gradient Echo (T1-MPRAGE)*. The *MRA* is for imaging blood vessels of the brain, to generate images of the arteries for stenosis (abnormal narrowing), occlusion or aneurysms while the *DTI* is for determining the magnitude and the direction of water and oxygen in the brain. Glioma tumors and lesions can be diagnosed using *T1-FLASH* procedures and the *T1-MPRAGE* is for detecting metastatic brain tumors.

Meanwhile, all the human anatomical structures are in 3-D, inclusive the brain and the slices therein, but the traditional cameras and the medical imaging sensors can only acquire the 3-D of the human anatomical structures in 2-D. With such sequence of 2-D slices, clinicians need to study in slice-by-slice, which in most cases runs into hundreds. Diagnosis requires surgeons and radiologists to visualize and reconstruct the slices abstractly in their minds, based on their expertise and experience, in order to be able to imagine the possible nature, the size and probably the position of the suspected abnormality. This is tedious, it is time-consuming and infact it is endangering patients concerned. In lieu of these, a Visualization System named *SurLens* (short for Surgical Lens) is developed during this study. The author concentrated on the reconstruction of sequence of 2-D imagery into 3-D model that is capable of clearly detecting, mapping and isolating abnormalities / tumor in MR imagery within a considerable interactive speed, extensive application interoperability and at a low resulting cost for optimum use in medical diagnosis and therapy treatment.

SurLens Visualization System was evaluated with Five Hundred and Forty (540) Patients' datasets with each dataset containing a minimum of 160 image slices amounting to One Hundred and Seventy-Three Thousand, Eight Hundred and Eighty (173,880) images in total. The datasets were obtained from the Computer-Assisted Surgery and Imaging Laboratories, University of North Carolina, United States. The results, in relative to the objectives of *SurLens Visualization System*, were compared with the two (2) notable Visualization Systems, the *ParaView* and the *VolView*, which are likewise built upon the Visualization Tool Kit. *SurLens Visualization System*



produces the best image quality, functionally revealing the internal features of brain datasets than *ParaView* and *VolView*. The study achieved the quality 3-D models with *SurLens Visualization System* without prior segmentation stages. This cannot be achieved with *ParaView* and *VolView Visualization Systems*.

In order to evaluate the framework speed performances, the author implemented *SurLens Visualization System* on three (3) experimental testbeds. Quadro GPU is the main *SurLens* testbed while others are for comparison in terms of rendering speed. The study achieved the best interactive speed with *SurLens* Quadro GPU testbed. The experiments recorded overall rendering time of less than 2 seconds for most of the datasets containing up to 322 images and not more than 3 seconds with some datasets, which might be as a results of variations in the quantity of pixels, accumulation of pixels to certain region, voxel spacing, the presence and nature of abnormalities in the datasets.

SurLens Visualization System was taken through a divert experiment and comparison with *ParaView* and *VolView Visualization Systems* using un-segmented abdominal pelvic CT datasets obtained from Uppsala Multidisciplinary Centre for Advanced Computational Science, Sweden. The results have proven *SurLens* to be a future, promising 3-D Multimodal Visualization System. Without going through the usual time-consuming segmentation processes of datasets, *SurLens* clearly reveals the internal features of the un-segmented datasets and isolate any abnormality / tumor therein, which is not possible with *ParaView* and *VolView Visualization Systems*.

The development of *SurLens Visualization System* was validated with the standards set aside for the development of clinical applications. This standard was presented as Application Oriented Hypothesis (AOH), approved by 24 independent medical doctors (the radiologists, the surgeons, the internists, the radiology physicists and the general practitioners), from three (3) different European hospitals and recently presented by Kainz et al. (2011). The detail of the AOH was presented in Chapter 2. The application development standard addresses four (4) main paradigms. Firstly, medical 3-D image synthesis algorithm must speed up the information finding processes with the datasets. Secondly, it must be comprehensible for maintenance purposes and be related to familiar physical principle. Thirdly, the 3-D image synthesis must be able to synthesize datasets that exceed normal human dimensional analysis experience. Finally,



it must be able to provide necessary image quality and the overall results within a reasonably rendering time, which is the sacrificing issue with the present state-of-art 3-D image synthesis algorithms. In accordance with the clinical applications' development standard, *SurLens Visualization system* has been tested with raw medical datasets and identified to be:

1. a medical 3-D image synthesis system that speeds up the information finding processes in medical datasets;
2. a fully oriented programmatic abstraction system, that shields the complexity of its algorithms, hence creating more comprehensibility for maintenance. *SurLens* compositing procedure is based on the familiar physical principle, the ray casting technique.
3. a 3-D synthesis system that is capable of synthesizing datasets containing hundreds of images and complex tiny features which are far beyond the normal human mental analysis.
4. a functional 3-D synthesis system that is able to achieve both the quality images and the processing of overall results within a reasonably rendering time. *SurLens* was able to achieve overall rendering time of less than 2 seconds for most of the datasets containing up to 322 images and not more than 3 seconds with some datasets of special features. It had the best image qualities in comparison with two (2) notable Visualization Systems, the *ParaView* and the *VolView*.

Therefore, *SurLens Visualization System* is in conformity with the standard of clinical application acceptance requirements for medical 3-D image synthesis algorithms. The summary of the major contributions of this thesis is presented in the next section.

6.3 CONTRIBUTIONS

This study explored solving these four (4) main issues. Firstly, a framework for reconstructing sequence of 2-D cross-sectional images produced by Magnetic Resonance (MR) Imagery scanners, to 3-D model. Secondly, the integration of a better efficient feature detection scheme that can allocate transparency based on scalar values and assign transparency based on localized gradient magnitude for edge detection. Thirdly, an automatic technique with local feature mapping scheme that can isolate abnormalities / tumor and reveal internal features of brain blood vessels. Finally, introduction of algorithms within the framework that are robust enough to handle mass data, evaluated through speed performances of varying datasets as being explained in Chapter 5 section 5.5, within a considerable interactive speed, having extensive application interoperability for revolutionary tools and at a cheaper computational cost. The ideas and findings of this study have been published in conferences, and in both Computer Science / Information Technology and Computational Life Sciences journals. The major contributions of this study are as follows:

1. A new framework was introduced for reconstructing sequence of 2-D cross-sectional images to 3-D model.
2. A feature and edge detection scheme was introduced which can allocate transparency based on scalar values and assign transparency based on localized gradient magnitude for edge detection of region of interest in data volume.
3. A novel automatic technique was introduced with local feature mapping scheme that can isolate abnormalities / tumor and reveal internal features of brain blood vessels.
4. Four (4) independent and parallel sets of algorithms (the *SurLens Data Pre-Processing Algorithm*, the *SurLens Accelerating Hardware Algorithm*, the *SurLens Graphic Execution Algorithm* and the *SurLens Volume Rendering Algorithm*), that are robust enough to handle mass data within a considerable



interactive speed, extensive application interoperability and at a lower resulting cost are developed.

SurLens, a volume visualization system is developed using C# programming language built on top of visualization toolkit (VTK) libraries and on parallel computing platform, Compute Unified Device Architecture (CUDA).

6.4 FUTURE WORKS

In the course of this study, a number of future works was identified. The level of pixels, the nature and intense of abnormalities are among the factors influencing the processing time of datasets. Future work will focus on acquiring extensive medical knowledge related to the nature, the types, the distinct characteristics and the behaviours of some of the common brain abnormalities. The understanding of which would be translated into series of subroutines to be appended to *SurLens* Visualization System. Similarly, some intelligent components might be integrated to facilitate automatic medical diagnosis procedures. These will not only adapt *SurLens* Visualization System to function as an independent automatic abnormalities-specific computer-aided medical diagnosis system but will also enable measuring of the levels and the extents of the detected abnormalities.

Tensor Visualization is an emerging area in volume visualization. Tensors are multidimensional array typically in second-order. Future work will align *SurLens* algorithms to more oriented tensor reconstruction algorithms, in an attempt to fully adapt *SurLens* Visualization System to a functional 3-D multimodal abnormalities / tumor clinical analysis and classification system.



REFERENCES

- Agrawal, S. & Sahu, R. (2012). Wavelet Based MRI Image Denoising Using Thresholding Techniques. *International Journal of Science, Engineering and Technology Research (IJSETR)*. Vol. 1, Issue 3, September.
- Ahrens, J., Geveci, B., & Law, C. (2005). *The Visualization Handbook, ParaView: An End-User Tool for Large Data Visualization*. Burlington, MA: Elsevier, 717.
- Alim, U.R. & Möller, T. (2009). A Fast Fourier Transform with Rectangular Output on the BCC and FCC Lattices. *Proc. Eighth Int'l Conf. Sampling Theory and Applications (SampTA)*.
- Aliroteh, M. & McInerney, T. (2007). SketchSurfaces: Sketch Line Initialized Deformable Surfaces for Efficient and Controllable Interactive 3D Medical Image Segmentation, *Third International Symposium on Visual Computing (ISVC)*, LNCS 4841, Lack Tahoe, Nevada/California, November 26-28, pp. 542-553.
- Amruta, A., Gole, A. & Karunakar, Y. (2010). A Systematic Algorithm for 3-D Reconstruction of MRI based Brain Tumors using Morphological Operations and Bicubic Interpolation. *2nd International Conference on Computer Technology and Development (KCTD)*.
- An, S. & An, D. (1984). Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Trans Pattern Anal Mach Intell* 6: 721-741.
- Archirapatkave, V., Sumilo, H., See, S.C.W. & Achalakul, T. (2011). GPGPU Acceleration Algorithm for Medical Image Reconstruction: *Ninth IEEE International Symposium on Parallel and Distributed Processing with Applications IEEE*.



PTTA (Perpustakaan Tunjukkan Ammah)

- Baek, S.Y., Sheafor, D.H., Keogan, M.T., DeLong, D.M. & Nelson, R.C. (2001). Two-dimensional multiplanar and three-dimensional volume-rendered vascular CT in pancreatic carcinoma: interobserver agreement and comparison with standard helical techniques. *Am J Roentgenol* 176(6):1467–1473.
- Bandhyopadhyay, S.K. & Paul, T.U. (2012). Segmentation of Brain MRI Image A Review. *International Journal of Advanced Research in Computer Science and Software Engineering*. Volume 2, Issue 3, March 2012 ISSN: 2277 128X.
- Bentoumi, H., Gautron, P. & Bouatouch, K. (2010). GPU-Based Volume Rendering for Medical Imagery. *International Journal of Electrical, Computer, and Systems Engineering* 4:1.
- Ben-Zadok, N., Riklin-Raviv, T. & Kiryati, N. (2009). Interactive level set segmentation for image-guided therapy. In *IEEE Int. Symp. On Biomedical Imaging*, pages 1079–1082.
- Benzinga News. November 29, 2010. Kitware Offers Free Global Access to VolView at RSNA.
- Bezdek, J.C., Hall, L.O. & Clarke, L.P. (1993). Review of MR Image Segmentation Techniques using Pattern Recognition, *Med. Phys.* 20 (4) 1033–1048.
- Birk, M., Guth, A., Zapf, M., Balzer, M., Rüter, N., Hübner, M. & Becker, J. (2011). Acceleration of Image Reconstruction in 3D Ultrasound Computer Tomography: An Evaluation of CPU, GPU AND FPGA Computing. *IEEE Conference on Design and Architectures for Signal and Image Processing (DASIP)*. On page(s): 1 – 8, E-ISBN : 978-1-4577-0619-6, Print ISBN: 978-1-4577-0620-2.
- Bredel, M. (2009). Gene Connections Key to Brain Tumor. *Journal of the American Medical Association*. The U.S. National Cancer Institute, Cancer Newsletter. July, 20.
- Bremer, P.T., Weber, G.H., Tierny, J., Pascucci, V., Day, M.S. & Bell, J.B. (2011). Interactive Exploration and Analysis of Large-Scale Simulations Using



- Topology-Based Data Segmentation. *IEEE Transactions on Visualization And Computer Graphics*, Vol. 17, No. 9, pp. 1307- 1324.
- Buades, A., Coll, B. & Morel, J. (2005). A Non-Local Algorithm for Image Denoising. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp 60–65.
- Bullitt, E., Zeng, D., Mortamet, B., Ghosh, A., Aylward, S.R., Lin, W., Marks, B.L. & Smith, K. (2010). The Effects of Healthy Aging on Intracerebral Blood Vessels Visualized by Magnetic Resonance Angiography: *Neurobiol Aging* 31(2): 290–300.
- Busking, S. Vilanova, A. & Wijk, J.V. (2007). Particle-Based Non-Photorealistic Volume Visualization,” *Visual Computer*, vol. 24, No. 5, pp. 335-346, May 2007.
- Cao, Y., Wu, G. & Wang, H. (2011). A Smart Compression Scheme for GPU-Accelerated Volume Rendering of Time-Varying Data. *IEEE International Conference on Virtual Reality and Visualization*, Page(s): 205 – 210.
- Carlos, D.C. & Ma, kwan-Liu (2009). The occlusion spectrum for volume classification and visualization, *IEEE Transactions on Visualization and Computer Graphics*, Vol. 15, No. 6.
- Chen, C. & Yang, J. (2011). Essence of Two-dimensional Principal Component Analysis and Its Generalization: Multi-dimensional PCA. *Second International Conference on Innovations in Bio-inspired Computing and Applications*.IEEE Computer Society.
- Chen, M., Kaufman, A. & Yagel, R. (2000). *Volume Graphics*, Springer (Eds.). London.
- Chen, Ming-Da., Hsieh, Tung-Ju. & Chang, Yang-Lang. (2011). Volume Data Numerical Integration and Differentiation Using CUDA. *IEEE 17th International Conference on Parallel and Distributed Systems*.
- Cheung, M.R. & Krishnan, K. (2012). Using Manual Prostate Contours to Enhance Deformable Registration of Endorectal MRI. *Computer Methods and Programs in Biomedicine* 108, 330-337.



PTTA UTHM
PERPUSTAKAAN TUNKUTUNJAMINAH

- Chiueh, T.-C., Yang, C.-K., He, T., Pfister, H. & Kaufman, A.E. (1997). Integrated Volume Compression and Visualization. In Proc. IEEE Visualization, Pages 329–336. Computer Society Press.
- Chiw, C., Kindlmann, G., Reppy, J., Samuels, L. & Seltzer, N. (2012). Diderot: A Parallel DSL for Image Analysis and Visualization. Proceedings of the 33rd ACM SIGPLAN conference on Programming Language Design and Implementation. ACM New York, NY, USA, pp 111-120.
- Chu, H., Chen, L. & Yong, J. (2010). Feature variation curve guided transfer function design for 3D medical image visualization, 3rd International Conference on Biomedical Engineering and Informatics.
- Clarke, L.P., Velthuizen, R.P., Camacho, M.A., Heine, J.J., Vaidyanathan, M., Hall, L.O., Thatcher, R.W. & Silbiger, M.L. (1995). MRI Segmentation: Methods and Applications, *Magn. Reson. Imag.* 13 (3) 343–368.
- Cosmus, C. C & Parizh, M. (2011). Advances in Whole-body MRI Magnets. *IEEE transactions on applied semiconductor*, Vol. 21, No. 3.
- Courchesne, E., Chisum, H.J., Townsend J, Cowles, A., Covington, J., Egaas, B., Harwood, M., Hinds, S. & Gary, A. (2000). Normal Brain Development and Aging: Quantitative Analysis at in vivo MR Imaging in Healthy Volunteers. *Journal of Radiology*. Press GA. 216:672–682. [PubMed: 10966694].
- Cox, G., Maximo, A., Bentes, C. & Farias, R. (2009). Irregular grid Raycasting implementation on the cell broadband engine, 21st International Symposium on Computer Architecture and High Performance Computing.
- Creasey, H. (2003). Rapoport SI. The aging human brain. *Annals of Neurology*. 17:2–10. [PubMed:3885841].
- Cremers, D., Fluck, O., Rousson, M. et al. (2007). A probabilistic level set formulation for interactive organ segmentation. *Medical Imaging 2007: Image Processing*, 6512(1):120–129.



PTTA UTM
 PERPUSTAKAAN TUNJUKU AMINAH

- Csébfalvi, B. & Domonkos, B. (2009). Frequency-Domain Upsampling on a Body-Centered Cubic Lattice for Efficient and High-Quality Volume Rendering. Conference on Vision Modeling and Visualization – VMV, pp. 225-232.
- Csébfalvi, B. & Szirmay-Kalos, L. (2003). Monte Carlo Volume Rendering. Proc. of IEEE Visualization, pp.449- 456, 2003.
- Da Silva, L.S., & Scharcanski, J. (2005). A lossless Compression Approach for Mammographic Digital Images Based on the Delaunay Triangulation. IEEE International Conference on Image Processing, ICIP. pp.11 - 758-61.
- Damadian, R., Goldsmith, M. & Minkoff, L. (1977). NMR in cancer: XVI. Fonar image of the live human body”, *Physiological Chemistry and Physics*, Vol. 9, pp. 97-100.
- Datta, S. & Narayana, P.A. (2011). Automated Brain Extraction from T2- Weighted Magnetic Resonance Images, *J. Magn.Reson.* 33 (4) 822– 829.
- Deng, W., Xiao, W., Deng, E. & Liu, J. (2010). MRI Brain Tumor Segmentation With Region Growing Method Based On The Gradients and Variances Along And Inside Of The Boundary Curve. 3rd International Conference on Biomedical Engineering and Informatics (BMEI).
- Diaz, I., Boulanger, P., Greiner, R. & Murtha, A. (2011). A Critical Review of the Effect of De-noising Algorithms on MRI Brain Tumor Segmentation. 33rd Annual International Conference of the IEEE EMBS, Boston, Massachusetts, USA.
- Dorgham, O.M., Laycock, S.D. & Fisher, M.H. (2012). GPU Accelerated Generation of Digitally Reconstructed Radiographs for 2-D/3-D Image Registration. *IEEE Transactions on Biomedical Engineering*, Vol. 59, No. 9. Pp. 2594 – 2603.
- Drebin, R., Carpenter, L., Hanrahan, P. (1988). Volume rendering, *Proceedings SIGGRAPH88*, pp 65–74.
- Fang, J., Varbanescu, A.L. & Sips, H. (2011). A Comprehensive Performance Comparison of CUDA and OpenCL. *IEEE International Conference on Parallel Processing*.



- Ferre, M., Cobos, S., Aracil, R. & Sánchez Urán, M.A. (2007). 3D Image Visualization and Its Performance in Teleoperation, HCI. International Conference, Peking, China. Virtual Reality, Vol.14, LNCS 4563, R. Shumaker (Hrg.); Springer, Volume 14, LNCS 4563, pp 669-707.
- Freitas, P., Rittner, L., Appenzeller, S. & Lotufo, R. (2011). Watershed-based Segmentation of the Midsagittal Section of the Corpus Callosum in Diffusion MRI. 24th Conference on Graphics, Patterns and Images 2011 24th SIBGRAPI Conference on Graphics, Patterns and Images. Pg 274-280.
- Frigo, M. & Johnson, S. (2005). The Design and Implementation of FFTW3. Proc. of the IEEE, 93(2): 216-231.
- Gabor J. Tornai, G.J. & Cserey, G. (2010). 2D and 3D Level-Set Algorithms on CPU. 12th International Workshop on Cellular Nanoscale Network and their Applicatins (CNNA).
- Geng, J. (2011). Structured-light 3D surface imaging: a tutorial. Advances in Optics and Photonics 3, 128-160.
- Geoffrey S.P., Elizabeth, Charles-Edwards & Christopher, P. (2008). Applications of Computed Tomography, Magnetic Resonance Imaging and Magnetic Resonance Spectroscopy for Planning External Beam Radiotherapy, Current Medical Imaging Reviews, 4, 236-249.
- Ghorpade, J., Parande, J., Kulkarni, M. & Bawaskar, A. (2012). Gpgpu Processing in Cuda Architecture. Advanced Computing: An International Journal (ACIJ), Vol.3, No.1.
- Gong, F. & Zhao, X. (2010). Three-Dimensional Reconstruction of Medical Image Based on Improved Marching Cubes Algorithm. International Conference on Machine Vision and Human-machine Interface.
- GPU Computing and the CUDA architecture (2009). NVIDIA CUDA Architecture Introduction & Overview, Version 1.1.
- Gu, S., Wilson, D., Wang, Z., Bigbee, W.L., Siegfried, J., Gur, D. & Pu, J. (2012). Identification of Pulmonary Fissures using A Piecewise Plane Fitting



- Algorithm. *Computerized Medical Imaging and Graphics*. *Computerized Medical Imaging and Graphics* 36 560– 571.
- Guo, H., Mao, N. & Yuan, X. (2011). WYSIWYG (What You See is What You Get) Volume Visualization. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 17, NO. 12, on page(s): 2106 – 2114, ISSN : 1077-2626.
- Guo, H., Xiao, H. & Yuan, X. (2012). Scalable Multivariate Volume Visualization and Analysis Based on Dimension Projection and Parallel Coordinates. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 18, No. 9, pp 119-120.
- Guo, H., Xiao, H. & Yuan, X. (2011). Multi-Dimensional Transfer Function Design Based on Flexible Dimension Projection Embedded in Parallel Coordinates. *Proc. IEEE Pacific Visualization Symp.*, pp. 19-26.
- Gupta, D., Anand, R.S., & Tyagi, B. (2012). Enhancement of Medical Ultrasound Images using Non-Linear Filtering Based on Rational-Dilation Wavelet Transform. *Proceedings of the World Congress on Engineering and Computer Science (WCECS)*, Vol. I October 24-26. San Francisco, USA
- Hadwiger, M., Kniss, J., Rezk-Salama, C., Weiskopf, D. & Engel, K. (2006). *Real-time volume graphics*, A K Peters Publications.
- He, X. (2009). Reconstruction of 3d microstructure of the rock sample Basing on the CT images. *Proceedings of the International Conference on Wavelet Analysis and Pattern Recognition*, Baoding, 12-15 July.
- Hege, H.C., Höllerer, T. & Stalling, D. (1996). *Volume Rendering - Mathematical Models and Algorithmic Aspects*. W. Nagel (Hrsg.) *Partielle Differentialgleichungen, Numerik und Anwendungen*. Konferenzen des Forschungszentrums Jülich GmbH, S. 227-255.
- Herlambang, N., Liao, H., Matsumiya, K., Masamune, K. & Takeyoshi, D. (2008). Real Time Autostereoscopic Visualization of Registration Generated 4D MR Image of Beating Heart, *Medical Imaging and Augmented Reality (MIAR)*, 4th International Workshop Tokyo, Japan, August 1-2, pp 349-358.



- Hernell, F., Ljung, F. & Ynnerman, A. (2010). Local ambient occlusion in direct volume rendering, *IEEE Transactions on Visualization and Computer Graphics*, Vol. 16, No. 4.
- Hong, L. & Shuhuil, M. (2010). High Precision Hybrid Technique of Surface and Volume Rendering. *Second International Conference on Computational Intelligence and Natural Computing (CINC)*.
- Hossain, Z., Alim, U.R. & Möller, T. (2011). Toward High-Quality Gradient Estimation on Regular Lattices. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 17, No. 4, pp. 426 – 439.
- Hounsfield G.N. (1973). Computerized transverse axial scanning tomography, Description of system *Br. J. Radiol.*, 46 1016.
- Hsieh, J. (2002). *Computed Tomography Principles, Design, Artifacts, and recent Advances*, Spie Press.
- Hu, S. & Hou, W. (2011). Denosing 3D Ultrasound images by Non-local Means Accelerated by GPU. *IEEE International Conference on Intelligent Computation and Bio-Medical Instrumentation*, pp. 43-45.
- Jeong, Won-Ki., Schneider, J., Turney, S.G., Faulkner-Jones, B.E., Meyer, D., Westermann, R., Reid, R.C., Lichtman, J. & Pfister, H. (2010). Interactive Histology of Large-Scale Biomedical Image Stacks. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 16, no. 6, November/December.
- Jinzhu, Y., Fangfang, H., Chaolu, F., Dazhe, Z. & Yanfei, W. (2011). An Accelerative Method for Multimodality Medical Image Registration Based on CUDA. *4th International Congress on Image and Signal Processing (CISP)*, pp. 1817 – 1821.
- Joemai, R.M.S., Geleijns, J., Veldkamp, W.J.H., De Roos, A. & Kroft, L.J.M. (2008). Automated cardiac phase selection with 64-MDCT coronary angiography, *AJR Am J Roentgenol* 191:1690–1697.
- Jovanovic, R. & Lorentz, R.A. (2012). A Combined Image Approach to Compression of Volumetric Data using Delaunay Tetrahedralization. *Conference on Image Processing (IPR), IET*. Pp. 1-6.



- Kainz, B., Portugaller, R.H, Seider, D., Moche, M., Stiegler, P. & Schmalstieg, D. (2011). Volume visualization in the clinical practice. *Augmented Environments for Computer Assisted Interventions (AE-CAI'11)*.
- Kalender, W.A. (2006). Review: X-ray Computed Tomography, *Institute of Physics Publishing Phys. Med. Biol.* 51, R29–R43.
- Kannan, S.R. & Pandiyarajan, R. (2009). Effective fuzzy c-mean Clustering technique for segmentation of T1-T2 brain MRI. *IEEE International Conference on Advances in Recent Technologies in Communication and Computing*. Pg. 537-539.
- Kasiri, K., Dehghani, M.J., Kazemi, K., Helfroush, M.S. & Kafshgari, S. (2010). Comparison Evaluation of Three Brain MRI Segmentation Methods in Software Tools. *IEEE Proceedings of the 17th Iranian Conference of Biomedical Engineering (ICBME)*.
- Kasthuri, N. & Lichtman, J.W. (2010). Neurocartography, *Neuropsychopharmacology*, 35, 342–343; doi:10.1038/npp.2009.138.
- Kaufman, A. (1991). *Volume Visualization (Tutorial)*. IEEE Computer Society Press, Los Alamitos, California.
- Kaufman, A. & Mueller, K. (2005). Overview of Volume Rendering, *The Visualization Handbook*, eds. C. Johnson and C. Hansen, Academic Press.
- Kaufman, A.E. (1996). Volume Visualization. *ACM Computing Survey*, 28(1): 165-167.
- Kaufman, A.E. (2000). *Volume visualization: Principles and advances*. International Spring School on Visualization, Bonn.
- Kim, J. & Jaja, J. (2009). Streaming Model Based Volume Ray Casting Implementation for Cell Broadband Engine, *Scientific Programming*, Vol. 17, no. 1-2, pp. 173-184.
- Kirk, D. & Hwu, W.-M. (2010). *Programming Massively Parallel Processors: A Hands-on Approach*. Morgan Kaufmann, 2010.
- Kirk, D.B. & Hwu, W.M.W. (2010). *Programming Massively Parallel Processors: A Hands on Approach*. New York: Elsevier, 2010.



- Kirmizibayrak, C., Radeva, N., Mike Wakid, M., Philbeck, J., John Sibert, J. & Hahn, J. (2011). Evaluation of Gesture Based Interfaces for Medical Volume Visualization Tasks. Proceedings of the 10th International Conference on Virtual Reality Continuum and Its Applications in Industry Pg. 69-74. ACM New York, NY, USA.
- Kitware News. October 10, 2002. Kitware Announces ParaView 0.6.
- Koo, J.J., Evans, A.C. & Gross, W.J. (2009). 3-D Brain MRI tissue classification on FPGAs, IEEE Transactions on image processing, vol.18, No.12.
- Krechetova, K., Glaz, A. & Platkajis, A. (2008). 3D Medical Image Visualization and Volume Estimation of Pathology Zones, NBC – 14th Nordic-Baltic Conference on Biomedical Engineering and Medical Physics, Latvia (IFMBE Proceedings) Vol. 20, pp 532-535.
- Krestel, E. (1990). Imaging Systems for Medical Diagnosis. Siemens, Aktienges, Munich. Krömer, P., Platoš, J. & Snásel, V. (2011). Differential Evolution for the Linear Ordering Problem Implemented on CUDA. IEEE Publications.
- Kumar, N., Nasser, M., Sarker, S.C. (2011). A New Singular Value Decomposition Based Robust Graphical Clustering Technique and Its Application in Climatic Data. Journal of Geography and Geology Vol. 3, No. 1.
- Kumar, T.S. & Rakesh, P.B. (2011). 3D Reconstruction of Facial Structures from 2D images for cosmetic surgery. International Conference on Recent Trends in Information Technology, ICRTIT MIT, Anna University, Chennai.
- Lai, Po-Lun & Yilmaz, A. (2008). Projective reconstruction of building shape from silhouette Images acquired from uncalibrated cameras. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. Vol. XXXVII. Part B3b. Beijing.



PTTA UTHM
PERPUSTAKAAN TUNKU TUNJUNGAN AMINAH

- Law, C.C., Henderson, A. & Ahrens, J. (2001). An Application Architecture for Large Data Visualization: A Case Study. *IEEE Symposium on Parallel and Large-Data Visualization and Graphics*. Pp. 125 – 159.
- Lei, G., Dou, Y., Wan, W., Xia, F., Li, R., Ma, M. & Zou, D. (2012). CPU-GPU hybrid accelerating the Zuker algorithm for RNA Secondary Structure Prediction Applications. *BMC Genomics*, 13 (Suppl 1):S14.
- Li, M., Zheng, X., Wan, X., Luo, H., Zhang, S. & Tan, L. (2011). Segmentation of brain tissue based on connected component labeling and mathematic morphology. *4th International Conference on Biomedical Engineering and Informatics (BMEI)*. pg 482-485.
- Lindholm, E., Nickolls, J., Oberman, S. & Montrym, J. (2008). NVIDIA Tesla: A Unified Graphics and Computing Architecture. *IEEE Micro*, 28(2):39–55.
- Lindholm, S., Ljung, P., Lundstrom, C., Persson, A. & Ynnerman, A. (2010). Spatial conditioning of transfer functions using local material distributions, *IEEE Transactions on Visualization and Computer Graphics*, Vol. 16, No. 6.
- Ling, F., Yang, L. & Wang, Zhong-Ke (2009). Improvement on Direct Volume Rendering, *Image and Signal Processing, CISP*.
- Ling, T. & Zhi-Yu, Q. (2011). An Improved Fast Ray Casting Volume Rendering Algorithm of Medical Image. *IEEE 4th International Conference on Biomedical Engineering and Informatics (BMEI)*.
- Liu, B., Wünsche, B. & Ropinski, T. (2010). Visualization by example – A constructive visual component-based interface for direct volume rendering, *Computer Graphics Theory and Applications*, 254-259.
- Liu, H., Yu, X., Wan, W. & Swaminathan, R. (2012). An Improved Spectral Subtraction Method. *International Conference on Audio, Language and Image Processing (ICALIP)*. 16-18 July, Page(s): 790 – 793.
- Lladó, X., Oliver, A., Cabezas, M., Freixenet, J., Vilanova, J.C., Quiles, A., Valls, L., Ramió-Torrentà, L. & Rovira, A. (2012). Segmentation of multiple sclerosis lesions in brain MRI: A Review of Automated Approaches. *Information Sciences* 186 (2012) 164–185.



- Lorensen, W.E. & Cline, H.E. (1987). Marching cubes: A High Resolution 3-D Surface Construction Algorithm. *Computer Graphics*. Volume 21, Number 4.
- Lundström, C. (2007). Efficient Medical Volume Visualization, Linköping Studies in Science and Technology Dissertations, No. 1125.
- Lv, X., Gao, X. & Zou, H. (2008). Interactive curved planar reformation based on snake model. *Comput Med Imaging Graph* 32(8):662–669.
- Ma, J., Murphy, D. & O’Mathuna, C. (2012). Visualizing Uncertainty in Multi-Resolution Volumetric Data Using Marching Cubes. *Proceedings of the International Working Conference on Advanced Visual Interfaces*, ACM New York, NY, USA, pp. 489-496.
- Manke, F. & Wönsche, B. (2008). A Direct Volume Rendering Framework for the Interactive Exploration of Higher-Order and Multifield Data. *GRAPP – International Conference on Computer Graphics Theory and Applications*.
- Manke, F. & Wünschev, B. (2009). Texture-Enhanced Direct Volume Rendering. *GRAPP- International Conference on Computer Graphics Theory and Applications*.
- Marner, L., Nyengaard, J. R., Tang, Y. & Pakkenberg, B. (2003). Marked loss of Myelinated Nerve Fibers in the Human Brain with Age. *Journal of Comparative Neurology*. 462:144–152. [PubMed: 12794739].
- Martin, J.P., Vickery, R.J., Ziegeler, S. & Angelini, R. (2010). SSH Enabled ParaView. *DoD High Performance Computing Modernization Program Users Group Conference*. IEEE.
- Martin, K., Ibáněz, L., Avila, L., Barré, S. & Kaspersen, J.H. (2005). Integrating Segmentation Methods from the Insight Toolkit into A Visualization Application. *Medical Image Analysis* 9, 579–593.
- Max, N. (1995). Optical Models for Direct Volume Rendering. *IEEE Transactions on Visualization and Computer Graphics*, Vol.1, 99-108.



- McCormick, B.H., DeFanti, T.A. & Brown, M.D. (1987). Visualization in Scientific Computing, Computer Graphics Vol. 21, No 6, November, 1987.
- McManus, J.P. & Kinsman, C. (2002). *C# Developer's Guide to ASP .NET, XML, and ADO .NET*: Addison-Wesley, New York.
- Meißner, M., Pfister, H., Westermann, R. & Wittenbrink, C.M. (2000). Volume Visualization and Volume Rendering Techniques. The Euro Graphics Association.
- Mensmann, J., Ropinski, T. & Hinrichs, K. (2010). An advanced volume raycasting technique using GPU stream processing. Computer Graphics Theory and Applications, page 190-198.
- Mensmann, J., Ropinski, T. & Hinrichs, K. (2010). An Advanced Volume Raycasting Technique using GPU Stream Processing. International Conference of Computer Graphics Theory Appl., 2010, pp. 190–198.
- Merck, D. (2009). Model Guided Rendering for Medical Images, University of North Carolina at Chapel Hill.
- Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A. & Teller, E. (1953). Equation of State Calculations by Fast Computing Machines. Journal of Chem. Physics, 21: 1087– 1092.
- Meißner, M., Huang, J., Bartz, D., Mueller, K. & Crawfis, R. (2000). A Practical Evaluation of Popular Volume Rendering Algorithms: IEEE/ACM. Symposium on Volume Visualization, Salt Lake City, Utah.
- Mohamed, M.A., Abdul-Fattah, A.F.A, Asem, A.S., & El-Bashbishy, A.S. (2011). Medical image filtering, fusion and classification Techniques. Egyptian Journal of Bronchology. Vol. 5, No 2.
- Moreland, K. (2008). The ParaView Tutorial. Sadia National Laboratories, United States.



PTTA UTHM
 PERPUSTAKAAN TIINKU TUN AMINAH

- Moreland, K., Ayachit, U., Geveci, B. & Ma, K. -L. (2011). Dax Toolkit: A Proposed Framework for Data Analysis and Visualization at Extreme Scale. IEEE Symposium on Large Data Analysis and Visualization (LDAV). pp.: 97 – 104.
- Moulik, S. & Boonn, W. (2011). The Role of GPU Computing in Medical Image Analysis and Visualization. Medical Imaging, 2011. Advanced PACS- based Imaging informatics and Therapeutic Applications. Proc. of SPIE Vol. 7967, 79670L.
- Mueller, K., Chen, M. & Kaufman, A. (2001). Volume Graphics', (Eds.) Springer. London.
- Muigg, P., Hadwiger, M., Doleisch, H., & Gröller, E. (2011). Interactive Volume Visualization of General Polyhedral Grids. IEEE Transactions on Visualization and Computer Graphics, Vol. 17, no.12.
- Nakajima, H., Hasegawa, K., Nakata, S. & Tanaka, S. (2009). Volume Visualization with Grid-Independent Adaptive Monte Carlo Sampling. Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing. IHH-MSP. Pp. 1301-1304.
- Ng, K. -W., Wong, H.-C., Wong, U.-H. & Pang, W.-M. (2010). Probe-Volume: An Exploratory Volume Visualization Framework. 23rd International Congress on Image and Signal Processing (CISP). Vol. 5, pp. 2392 – 2395.
- Nickolls, J., Buck, I., Garland, M. & Skadron, K. (2008). Scalable Parallel Programming with CUDA. ACM Queue, 6(2):40–53.
- Nowak, R.D. (1999). Wavelet-Based Rician Noise Removal for Magnetic Resonance Imaging. IEEE Trans Image Process 8(10):1408–1419.
- Ohtake, Y., Belyaev, A., Alexa, M., Turk, G. & Seidel, H.-P. (2003). Multi-Level Partition of Unity Implicits, ACM Transactions on Graphics, Vol. 22, No.2 (2003) pp. 463-470.
- Oiso, M., Yasuda, T., Ohkura, K. & Matumura, Y. (2011). Accelerating Steady-State Genetic Algorithms based on CUDA Architecture. Congress on Evolutionary Computation (CEC), IEEE. Pp. 687 – 692, New Orleans, LA.



- Othman, M.F., Abdulahi, N. & Ahmad Rusli, N.A. (2010). An Overview of MRI Brain Classification Using FPGA Implementation, IEEE Symposium on Industrial Electronics and Applications.
- Owen, S. (1993). Visualization Education in the USA. *Journal of Computers and Education*. Vol. 8, pp. 339-345, 1993.
- Paulinas, M. & Usinskas, A. (2007). A survey of genetic Algorithms applications for image enhancement and segmentation”, *Information Technology Control*, Vol. 36, No. 3, pp. 278-284.
- Pelt, R.V., Vilanova, A. & Wetering, H.V.D. (2010). Illustrative Volume Visualization Using GPU-Based Particle Systems. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 16, Issue: 4, pp. 571-582.
- Peng, Y., Dong, J., Chen, L., Chu, H. & Yong, J. (2011). An Optimal Color Mapping Strategy based on Energy Minimization for Time-varying Data. 212th International Conference on Computer-Aided Design and Computer Graphics (CAD/Graphics), pp. 411-417.
- Perona, P. & Malik, J. (1990). Scale-space and edge detection using anisotropic diffusion, “*IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 12, No. 7, pp629 – 639.
- Petrescu, L., Morar, A., Moldoveanu, F. & Asavei, V. (2011). Real Time Reconstruction of Volumes from Very Large Datasets Using CUDA. 15th International Conference on System Theory, Control, and Computing (ICSTCC), pp. 1-5.
- Pham, D., Xu, C. & Prince, J. (2000). Current methods in medical image segmentation. *Annual Review of Biomedical Engineering*, vol. 2, pp. 315–337.
- Ponraj, D.N., Jenifer, M.E., Poongodi, P. & Manoharan, J.S. (2011). A Survey On the Preprocessing Techniques of Mammogram for the Detection of Breast Cancer. *Journal of Emerging Trends in Computing and Information Sciences*. Vol. 2, No. 12, ISSN 2079-8407.



PTTA AUTHM
PERPUSTAKAAN TUNJUNG AMINAH

- Porter, T. & Duff, T. (1984). Compositing Digital Images: *Computer Graphics*, vol. 18, no. 3.
- Praßni, Jörg-Stefan., Ropinski, T. & Hinrichs, K. (2010). Uncertainty-Aware Guided Volume Segmentation. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 16, no. 6.
- Pu, J., Leader, J.K., Zheng, B., Knollmann, F., Fuhrman, C., Scieurba, F.C., Gur, D. (2009). A Computational Geometry Approach to Automated Pulmonary Fssure Segmentation in CT Examinations. *IEEE Transaction Medical Imaging* 28(5): pp. 710–9.
- Qin, A.K., Raimondo, F., Fobes, F. & Ong, Y.S. (2012). An Improved CUDA-Based Implementation of Differential Evolution on GPU. *ACM Genetic and Evolutionary Computation Conference. GECCO*, Philadelphia, USA.
- Qu, D., Luo, Y. & Tan, W. (2011). An Improved Painting-Based Transfer Function Design Approach with CUDA-Acceleration. *IEEE International Conference on Computer Science and Automation Engineering (CSAE)*. Pp 372-377.
- Qureshi, M.N.I., Lee, J.-E., & Lee, S.W. (2012). Robust Classification Techniques for Connection Pattern Analysis with Adaptive Decision Boundaries Using CUDA. *IEEE International Conference on Cloud Computing and Social Networking (ICCCSN)*.
- Rodallec, M.H., Marteau, V., Gerber, S., Desmottes, L. & Zins, M. (2008). Craniocervical arterial dissection: spectrum of imaging findings and differential diagnosis, *Radiographics* 28:1711– 1728.
- Roos, J.E., Fleischmann, D., Koechl, A., Rakshe T., Straka M., Napoli, A., Kanitsar, A., Sramek, M., & Groeller, E. (2007). Multipath curved planar reformation of the peripheral arterial tree in CT angiography, *Radiology* 244:281–290.
- Rosenblum, L., Earnshaw, R.A., Encarnacao, J., Hagen, H. A., Kaufman, S.K., Nelson, G., Post, F. & Thalmann, D. (1994). *Scientific Visualization: Advances and Challenges*, Academic Press.



PT TAJEM
PERPUSTAKAAN TUNJUK TUNJANG
AMINAH

- Ross, J.C., Estepar, R., Kindlman, G., Dfaz, A., Westin, C.F., Silverman, E.K. Washko, G.R. (2010). Automatic Lung Lobe Segmentation using Particles, Thin Plate Splines and Maximum A Posteriori Estimation. *Med Image Comput-Assist Intervent* 2010:163–71.
- Sabella, P. (1988). A Rendering Algorithm for Visualizing 3D Scalar Fields: *Computer Graphics*. Volume 22, Number 4.
- Sakamoto, N. & Koyamada, K. (2005). Particle Generation from User- Specified Transfer Function for Point-Based Volume Rendering. *IEEE Visualization Proceedings Compendium*: 125-126.
- Sanftmann, H., Cipriani, N. & Weiskopf, D. (2011). Distributed Context-Aware Visualization. *IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*. Pp. 251 – 256.
- Schroeder, W., Martin, K. & Lorensen, B. (2002). *The visualization Toolkit, An object-oriented approach to 3D graphics*, 3rd Edition, Pearson Education, Inc.
- Senthikumar, N. & Rajesh, R. (2008). A Study of Split and Merge for Region Based Image Segmentation. *Proceedings of UGC sponsored National conference network security (NCNS-08)*, 2008, pp.57-61.
- Senthilkumarn, N. & Rajesh, R. (2009). Edge Detection Techniques for Image Segmentation - A Survey of Soft Computing Approaches. *IJRTE*, Vol 1, No2, 250-254.
- Sethian, J. A. (1999). *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry*. Fluid Mechanics. Computer Vision, and Materials. Cambridge University Press, 2 Edition.
- Sha, D.D. & Sutton, J.P. (2001). Towards automated enhancement, segmentation and classification of digital brain images using networks of networks, *Inf. Sci.* 138 (1–4) (2001) 45–77.



- Shabaneh, N., Amirpour, S., Stafford, S. & Shirazi, N. (2004). Radiation Health Risks and Benefits. *Computed Tomography*.
- Shen, A. & Luo, L. (2008). Point-Based Digitally Reconstructed Radiograph. *Proc. Int. Conf. Pattern Recog.*, 2008, pp. 1–4.
- Shen, H-W., & Johnson, C. (1994). Differential Volume Rendering: A Fast Volume Visualization Technique for Flow Animation. *Proceedings of the Visualization '94 Conference*, October 1994, pp. 180-187.
- Shi, X., Li, C., Wang, X. & Li, K. (2009). A Practical Approach of Curved Ray Prestack Kirchhoff Time Migration on GPGPU. *Advanced Parallel Processing Technologies 8th International Symposium*, pp. 165–176.
- Shihao, C., Guiqing, H. & Chongyang, H. (2009). Rapid Texture-Based Volume Rendering, *International Conference on Environmental Science and Information Application Technology*.
- Shihao, C., Guiqing, H. & Chongyang, H. (2009). Interactive GPU-based volume rendering for medical image, *Biomedical Engineering and Informatics, BMEI*.
- Siddique, M.T. & Zakaria, M.N. (2010). 3D Reconstruction of geometry from 2D image using Genetic Algorithm. *IEEE International Symposium in Information Technology (ITSim)*, Volume : 1, on page(s): 1 – 5, ISSN : 2155- 897, Print ISBN: 978-1-4244-6715-0.
- Sijbers, J., Dekker, A.J.D., Audekerke, J.V., Verhoye, M. & Dyck, D.V. (1998) Estimation of the noise in magnitude MR images. *Magnetic Resonance Imaging* 16(1):87–90.
- Song, J., Liu, Y., Gewalt, S.L., Cofer, G. & Johnson, G.A. (2009). Least-Square NUFFT Methods Applied to 2-D and 3-D Radially Encoded MR Image Reconstruction. *IEEE Transactions On Biomedical Engineering*, Vol. 56, No. 4, April 2009.
- Sonka, M. & Fitzpatrick, J.M. (2000). *Handbook of Medical Imaging*. SPIE.



PTTA UTHM
 PERPUSTAKAAN TUNJANGAN AKADEMIK

- Speray, D. & Kennon, S. (1990). Volume probes: interactive data exploration on arbitrary grids. *SIGGRAPH Comput. Graph*, 24(5): 5-12.
- Styner, M., Lee, J., Chin, B., Chin, M., Commowick, O., Tran, H., Jewells, V. & Warfield, S. (2008). 3D Segmentation in the Clinic: A Grand Challenge II: MS Lesion Segmentation. *Grand Challenge Work.: Mult.Scler.Lesion Segm. Challenge*, pp. 1–8.
- Subhranil Koley, S. & Majumder, A. (2011). Brain MRI Segmentation for Tumor Detection using Cohesion based Self Merging Algorithm. *IEEE 3rd International Conference on Communication Software and Networks (ICCSN)*. Page(s): 781 – 785.
- Sun, Y., Bhanu, B. & Bhanu, S. (2009). Symmetry Integrated Injury Detection for Brain MRI, *Image Processing (ICIP), 16th IEEE International*, Pages 661 – 664.
- Suter, S.K., Zollikofer, C.P. & Pajarola, R. (2010). Application of Tensor Approximation to Multiscale Volume Feature Representations. *Proc. Vision, Modeling and Visualization*, pages 203–210.
- Suter, S.K., Guiti'an, J.A.I., Marton, F., Agus, M., Elsener, A., Zollikofer, C.P.E., Gopi, M., Gobbetti, E. & Pajarola, R. (2011). Interactive Multiscale Tensor Reconstruction for Multiresolution Volume Visualization. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 17, No. 12.
- Szirmay-Kalos, L., Umenhoffer, T., Tóth, B. & Szécsi, L. (2010). Volumetric Ambient Occlusion for Real-Time Rendering and Games. *IEEE Computer Graphics and Applications*, Volume: 30, Issue: 1, Pp. 70-79.
- Tan, S., Yang, J. & Sun, W. (2011). Internet-Based Platform for Power System Simulating and Planning: *Second International Conference on Mechanic Automation and Control Engineering (MACE)*, IEEE.
- Tanoori, B., Azimifar, Z., Shakibafar, A. & Katebi, S. (2011). *Computers in Biology and Medicine* 41 (2011) 619–632.



- Tawara, T. & Ono, K. (2010). A Framework for Volume Segmentation and Visualization Using Augmented Reality. IEEE Symposium on 3D User Interfaces (3DUI).
- Teo, P.C. & Heeger, D.J. (1994). Perceptual image distortion: Proc. 1st International Conference on Image Processing, pp. 982-986.
- Teyseyre, A.R. & Campo, M.R. (2009). An Overview of 3D Software Visualization. IEEE Transactions on Visualization and Computer Graphics, Vol. 15, No. 1.
- The first Information Visualization Symposium (InfoVis) (1995). IEEE Computer Society Press, Los Alamitos, CA.
- The Oxford English Dictionary (1989). Oxford Advanced Learner's Dictionary of Current English, Oxford University Press, Second Edition.
- Tornai, G.J. & Cserey, G. (2010). 2D and 3D Level-Set Algorithms on GPU: 2010 12th International Workshop on Cellular Nanoscale Networks and their Applications (CNNA), IEEE.
- Tsai, Y.-T. & Shih, Z.-C. (2006). All-Frequency Precomputed Radiance Transfer using Spherical Radial Basis Functions and Clustered Tensor Approximation. ACM Transactions on Graphics, 25(3):967-976.
- Van, R.E.M, De, H.B., Van, D.V.S., Prokop, M. & Van, G.B. (2009). Automatic Segmentation of Pulmonary Segments from Volumetric Chest CT Scans. IEEE Trans Med Imaging 2009;28(4):621-30.
- Vawter, C. & Roman, E. (2001). J2EE vs. Microsoft.NET A comparison of building XML-based web services: Prepared for Sun Microsystems, Inc.
- VenuGopal, T. & Naik, P.P.S. (2011). Image Segmentation and Comparative Analysis of Edge Detection Algorithms. Int. Journal of Electrical, Electronics & Computing Technology, Vol.1 (3).ISSN 2229-3027.
- Vivekanandan, D. & Raj, S.R. (2011). A Feature Extraction Model for Assessing the Growth of Lung Cancer in Computer Aided Diagnosis IEEE-International Conference on Recent Trends in Information Technology, MIT, Anna University, Chennai.



PTTA UTM
PERPUSTAKAAN TUNKU TUN AMINAH

- Wakid, M., Kirmizibayrak, C. & Hahn, J.K. (2011). Texture Mapping Volumes using GPU-Based Polygon-Assisted Raycasting. IEEE 16th International Conference on Computer Games. Page(s): 162 – 166.
- Walter, T., Shattuck, D.W., Baldock, R., Bastin, M.E., Carpenter, A.E., Duce, S., Ellenberg, J., Fraser, A., Hamilton, N., Pieper, S., Ragan, M.A., Schneider, J.E., Tomancak, P. & Hériché, Jean-Karim (2010). Visualization of image data from cells to organisms” S26, Vol.7 No.3s, Nature Methods Supplement.
- Wang, L., Zhang, Y. & Feng, J. (2005). On the Euclidean Distance of Images. IEEE Transactions On Pattern Analysis and Machine Intelligence, Vol. 27, No. 8.
- Wang, R., Wan, W., Ma, X., Wang, Y. & Zhou, X. (2011). Accelerated Algorithm for 3D Intestine Volume Reconstruction Base on VTK. IEEE International Conference on Audio Language and Image Processing (ICALIP), pp: 448 – 452, Print ISBN: 978-1-4244-5856-1.
- Wang, S.Q., Zhang, J.H. & Yao, Z.X. (2009). Accelerating 3D Fourier Migration on Graphics Processing Units. SEG Expanded Abstracts, pp. 3020–3024.
- Węgliński, T. & Fabijańska, A. (2011). Brain Tumor Segmentation from MRI Data Sets using Region Growing Approach. MEMSTECH11, 11-14 May 2011, Polyana-Svalyava (Zakarpattya), Ukraine.
- Weiler, M., Westermann, R., Hansen, C., Zimmerman, K. & Ertl, T. (2000). Level-of-Detail Volume Rendering via 3d Textures. Proceedings of Symposium on Volume Visualization, pages 7–13. ACM, SIGGRAPH.
- Wen, X.B., Zhang, H. & Jiang, Z.T. (2008). Multiscale Unsupervised Segmentation of SAR Imagery using the Genetic Algorithms. Sensors, Vol.8, pp.1704-1711.
- Williams, A., Barrus, S., Morley, R.K. & Shirley, P. (2005). An Efficient and Robust Ray-Box Intersection Algorithm. Journal of Graph., GPU Game Tools, Vol. 10, no. 1, pp. 49–54, 2005.



- Williams, D., Grimm, S., Coto, E., Roudsari, A. & Hatzakis, H. (2008). Volumetric curved planar reformation for virtual endoscopy. *IEEE Trans Vis Comput Graph* 14(1):109–119.
- Winter, A.S. (2002). *Field-based Modelling and Rendering*. University of Wales, Swansea. PhD thesis.
- Wong, H.C., Wong, U.H. & Tang, Z.S. (2009). Direct volume rendering by transfer function morphine. *The 7th International Conference on Information Communications and Signal Processing (ICICS)*, Beijing, China, pp. 1-4.
- Wu, D., Tian, H., Hao, G., Du, Z. & Sun, L. (2010). Design and Realization of an Interactive Medical Images Three Dimension Visualization System. *IEEE 3rd International Conference on Biomedical Engineering and Informatics (BMEI)*.
- Wu, Q., Xia, T., Chen, C., Lin, H.-Y.S., Wang, H. & Yu, Y. (2008). Hierarchical Tensor Approximation of Multidimensional Visual Data. *IEEE Transactions on Visualization and Computer Graphics*, 14(1):186–199.
- Wu, Y. (2006). *Effective, Intuitive and Intelligent Volume Visualization*, Department of Computer Science, Hong Kong University of Science and Technology.
- Xiao, Y., Chen, Z. & Zhang, L. (2009). Accelerated CT Reconstruction Using GPU SIMD Parallel Computing with Bilinear Warping Method: *The 1st International Conference on Information Science and Engineering (ICISE)*, IEEE.
- Xie, K., Yu, W., Yu, H., Wu, P., Li, T. & Peng, M. (2011). GPU-based Multi-Resolution Volume Rendering for Large Seismic Data. *IEEE International Conference on Intelligence Science and Information Engineering*, pp. 245 - 248.
- Xujia, Q., Sida, Z., Xinhong, C. & Jun, H. (2009). Research and implementation of multi-dimensional transfer function based on boundaries, *Biomedical Engineering and Informatics*.



PTTA AUTHM
PERPUSTAKAAN TUN KU TUN AMINAH

- Yang, Guang-Zhong & Firmin, D.N. (2000). The birth of the first CT scanner. *Engineering in Medicine and Biology Magazine*, IEEE. Volume: 19 , Issue:1 pp: 120 – 125, ISSN : 0739-5175.
- Yang, X., Sechopoulos, I. & Fei, B. (2011). Automatic Tissue Classification for High-resolution Breast CT Images Based on Bilateral Filtering. *Proceedings Vol. 7962. Medical Imaging 2011: Image Processing*, ISBN: 9780819485045.
- Yeo, B.-L., & Liu, B. (1995). Volume rendering of DCT-based compressed 3D Scalar Data. *IEEE Transactions on Visualization and Computer Graphics*, 1(1):29–43.
- Yun, Y. & Xing, Z. (2010). An improved Method for volume rendering. 2nd International Symposium on Information Engineering and Electronic Commerce (IEEC), Issue Date : 23-25 July 2010, On page(s): 1 – 3.
- Zhang, F. & Ma, L. (2010). MRI Denoising Using the Anisotropic Coupled Diffusion Equations. *IEEE 3rd International Conference on Biomedical Engineering and Informatics (BMEI 2010)*.
- Zhang, Z.-M., Lu, W., Shi, Y.-Z., Yang, T.-L. & Liang, S.-L. (2012). An Improved Volume Rendering Algorithm Based on Voxel Segmentation. *IEEE International Conference on Computer Science and Automation Engineering (CSAE)*, Vol.1, pp. 372 – 375.
- Zhang, Q., Eagleson, R. & Peters, T.M. (2007). Rapid Voxel Classification Methodology for Interactive 3D Medical Images Visualization. *The 10th International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI)*, Brisbane, Australia, pp 86-93.
- Zhang, Q., Eagleson, R. & Peters, T.M. (2010). A Technical Overview with a Focus on Medical Applications, *Journal of Digital Imaging*.
- Zhao, F. & McGinnity, T.M. (2011). A Low-cost Real-time Three-dimensional Confocal Fluorescence Endomicroscopy Imaging System. *IEEE International Conference on Healthcare Informatics, Imaging and Systems Biology*, pp. 126 – 133.



- Zheng, Z., Xu, W. & Mueller, K. (2010). VDVR: Verifiable Visualization of Projection-Based Data. *IEEE Transactions on Visualization And Computer Graphics*, Vol. 16, NO. 6, pp. 1515 – 1524.
- Zhou, H., Tao, Y., Lin, H., Dong, F. & Clapworthy, G. (2011). Shape-Enhanced Maximum Intensity Projection. *Springer-Verlag Vis Comput* 27: 677–686.
- Zhu, Y., Ma, X., Zhou, X., Sun, Y., Yang, W., Zhang, S. & Wang, W. (2011). Research of Medical Image Reconstruction System based-on MAC OS. *IEEE Conference on Smart and Sustainable City (ICSSC 2011)*. IET International. pp: 1 – 4, Print ISBN: 978-1-84919-326-9.
- Zitova, B. & Flusser, J. (2003). Image Registration Methods: A Survey. *Image and Vision Computing* 21 (2003), 977–1000.



PTTA UTHM
PERPUSTAKAAN TUNKU TUN AMINAH

1.0 Appended MRA Datasets Speed Evaluation Results

<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
021_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800003	09.75s	05.15s	02.99s
022_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799995	09.70s	05.10s	02.95s
023_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799998	09.71s	05.11s	02.96s
024_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799998	09.71s	05.11s	02.96s
025_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800001	09.70s	05.10s	02.95s
026_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800002	09.70s	05.10s	02.95s
028_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.8	09.71s	05.11s	02.96s
029_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800001	09.70s	05.10s	02.94s
030_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800004	09.72s	05.12s	02.96s
031_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.8	09.71s	05.11s	02.96s
032_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800003	09.72s	05.13s	02.97s
033_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799998	09.71s	05.11s	02.96s
034_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.8	09.71s	05.11s	02.96s

<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
035_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800003	09.71s	05.11s	02.96s
036_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799996	09.71s	05.11s	02.96s
037_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799997	09.71s	05.11s	02.96s
039_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799998	09.71s	05.11s	02.96s
040_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800003	09.70s	05.11s	02.96s
042_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800003	09.70s	05.10s	02.95s
043_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800001	09.70s	05.10s	02.95s
044_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.8	09.71s	05.11s	02.96s
045_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800002	09.70s	05.10s	02.95s
046_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800003	09.70s	05.10s	02.95s
047_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800004	09.64s	05.18s	03.03s
049_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800003	09.70s	05.10s	02.95s
050_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799998	09.71s	05.11s	02.95s
051_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.8	09.71s	05.11s	02.96s
053_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799998	09.70s	05.11s	02.95s
054_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799999	09.71s	05.11s	03.05s



<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
055_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799997	09.71s	05.11s	02.95s
056_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800003	09.70s	05.10s	02.95s
057_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800002	09.70s	05.10s	02.95s
058_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800002	09.70s	05.10s	02.95s
059_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799997	09.70s	05.11s	02.96s
060_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.8	09.71s	05.11s	02.96s
062_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800003	09.70s	05.10s	02.95s
063_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799994	09.70s	05.10s	02.95s
064_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800002	09.70s	05.10s	02.95s
065_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800002	09.70s	05.10s	02.95s
066_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799997	09.71s	05.11s	02.96s
067_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800002	09.67s	05.18s	02.99s
068_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799998	09.71s	05.11s	02.96s
069_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800004	09.72s	05.13s	02.97s
070_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.8	09.71s	05.11s	02.96s
071_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800002	09.70s	05.10s	02.95s



<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
072_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799992	09.70s	05.10s	02.95
074_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.8	09.71s	05.11s	02.96
075_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800003	09.70s	05.10s	02.95s
076_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799998	09.71s	05.11s	02.96s
078_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800002	09.70s	05.10s	02.95s
082_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.8	09.70s	05.11s	02.95s
084_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799997	09.71s	05.11s	02.95s
085_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800001	09.70s	05.10s	02.95s
086_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799998	07.71s	05.11s	02.95s
087_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800003	09.70s	05.10s	02.95s
088_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800001	09.70s	05.10s	02.95s
089_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.8	09.71s	05.11s	02.95s
090_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800002	09.70s	05.10s	02.96s
091_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799995	09.71s	05.11s	02.96s
092_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800004	09.72s	05.13s	02.98s
093_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800001	09.70s	05.10s	02.95s



PERPUSTAKAAN TUNJUKKAN AMINAH

<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
094_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.800004	09.73s	05.13s	02.98s
095_MRA	128	448 x 448 x 128 voxels	0.513393 x 0.513393 x 0.799996	09.70s	05.10s	02.95s
096_MRA	176	352 x 448 x 176 voxels	0.513393 x 0.513393 x 0.800002	10.42s	05.17s	03.09s
098_MRA	176	352 x 448 x 176 voxels	0.513393 x 0.513393 x 0.8	09.67s	05.51s	03.30s
099_MRA	176	352 x 448 x 176 voxels	0.513393 x 0.513393 x 0.800002	10.42s	05.17s	03.09s
100_MRA	176	352 x 448 x 176 voxels	0.513393 x 0.513393 x 0.799999	10.41s	05.16s	03.08s
101_MRA	176	352 x 448 x 176 voxels	0.513393 x 0.513393 x 0.800004	10.41s	05.16s	03.08s
102_MRA	176	352 x 448 x 176 voxels	0.513393 x 0.513393 x 0.799998	10.40s	05.16s	03.09s
103_MRA	160	352 x 448 x 160 voxels	0.513393 x 0.513393 x 0.800004	09.16s	05.13s	02.70s
104_MRA	176	352 x 448 x 176 voxels	0.513393 x 0.513393 x 0.800001	10.76s	05.51s	03.30s
105_MRA	176	352 x 448 x 176 voxels	0.513393 x 0.513393 x 0.799998	10.79s	05.56s	03.30s
106_MRA	176	352 x 448 x 176 voxels	0.513393 x 0.513393 x 0.799998	10.79s	05.56s	03.30s
107_MRA	143	352 x 448 x 143 voxels	0.513393 x 0.513393 x 0.800002	09.13s	05.10s	02.43s
108_MRA	176	352 x 448 x 176 voxels	0.513393 x 0.513393 x 0.799993	10.40s	05.15s	03.06s
109_MRA	176	352 x 448 x 176 voxels	0.513393 x 0.513393 x 0.800003	10.42s	05.16s	03.06s



2.0 Appended DTI Datasets Speed Evaluation Results

<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
001_DTI	294	128 x 128 x 294 voxels	2 x 2 x 4	03.10s	01.30s	00.49s
002_DTI	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
003_DTI	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
004_DTI	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
005_DTI	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
006_DTI	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
007_DTI	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
008_DTI	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
010_DTI	322	128 x 128 x 322 voxels	2 x 2 x 1.99999	04.09s	02.29s	01.46s
011_DTI	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
012_DTI	322	128 x 128 x 322 voxels	2 x 2 x 2	04.00s	02.16s	01.43s
013_DTI	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
014_DTI	322	128 x 128 x 322 voxels	2 x 2 x 1.99999	04.09s	02.29s	01.46s
015_DTI	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
016_DTI	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
017_DTI	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
018_DTI	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
019_DTI	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
020_DTI	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
021_DTI	322	128 x 128 x 322 voxels	2 x 2 x 1.99999	04.09s	02.29s	01.46s

<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
<i>022_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>023_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.02s	02.24s	01.40s
<i>024_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>025_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>026_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>027_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>028_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>029_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>030_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>031_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>032_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>033_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>034_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>035_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>036_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>037_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>038_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>039_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>040_DTI</i>	-	-	-	-	-	-
<i>041_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>042_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>043_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>044_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>045_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>046_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s



<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
<i>048_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>049_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>050_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>051_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>052_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>053_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>055_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>056_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>057_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>058_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>059_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>060_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>061_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>062_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>063_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>064_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>065_DTI</i>	-	-	-	-	-	-
<i>066_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>069_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>070_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>071_DTI</i>	-	-	-	-	-	-
<i>072_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>073_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>074_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>075_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 1.99999	04.09s	02.30s	01.46s



PERPUSTAKAAN TUNKU ABULRAHMAN

<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
<i>076_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>077_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>078_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>079_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>080_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>081_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>082_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>083_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>084_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>085_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>086_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>087_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>088_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>089_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>090_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>091_DTI</i>	-	-	-	-	-	-
<i>092_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>093_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>094_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>095_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>096_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>097_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>099_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2.00001	04.10s	02.30s	01.47
<i>100_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>101_DTI</i>	-	-	-	-	-	-



PERPUSTAKAAN TUNJUNG AMINAH

<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
<i>102_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>103_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>104_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>105_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>106_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>107_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>108_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s
<i>109_DTI</i>	322	128 x 128 x 322 voxels	2 x 2 x 2	04.06s	02.26s	01.43s

3.0 Appended T1-FLASH Datasets Speed Evaluation Results

<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
<i>001_T1-FLASH</i>	256	176 x 256 x 256 voxels	1 x 1 x 1	04.90s	02.79s	01.93s
<i>002_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
<i>003_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
<i>004_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
<i>005_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
<i>006_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 0.999996	04.84s	02.41s	01.83s
<i>007_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s

<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
008_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.57s	02.07s	01.61s
009_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.57s	02.07s	01.61s
010_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 0.999994	05.04s	2.61s	01.84s
011_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.57s	02.07s	01.61s
012_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
013_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.57s	02.07s	01.61s
014_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.57s	02.07s	01.61s
015_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 0.999996	04.84s	02.41s	01.83s
016_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
017_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 0.999994	05.04s	2.60s	01.84s
018_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
019_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 0.999994	05.04s	2.61s	01.84s
020_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 0.999996	04.84s	02.41s	01.83s
021_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1.00001	04.50s	02.39s	01.62s
022_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 0.999999	04.61s	02.37s	01.79s
023_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 0.999998	04.61s	02.37s	01.79s
024_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
025_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 0.999998	04.61s	02.37s	01.79s
026_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
027_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.57s	02.07s	01.61s
028_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s



<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
<i>029_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 0.999999	04.61s	02.37s	01.79s
<i>030_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
<i>031_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 1	04.57s	02.07s	01.61s
<i>032_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 0.999998	04.61s	02.37s	01.79s
<i>033_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 0.999998	04.61s	02.37s	01.79s
<i>034_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 0.999996	04.84s	02.41s	01.83s
<i>035_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 0.999994	05.04s	2.61s	01.84s
<i>036_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
<i>037_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 1	04.57s	02.07s	01.61s
<i>038_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 0.999992	05.04s	2.61s	01.84s
<i>039_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 0.999996	04.84s	02.41s	01.83s
<i>040_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 0.999995	04.84s	02.41s	01.83s
<i>041_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 0.999999	04.61s	02.37s	01.79s
<i>042_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 1	04.57s	02.07s	01.61s
<i>043_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 0.999999	04.61s	02.37s	01.79s
<i>044_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
<i>045_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
<i>046_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
<i>048_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
<i>049_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
<i>050_T1-FLASH</i>	176	176 x 256 x 176 voxels	1 x 1 x 1	04.57s	02.07s	01.61s



<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
051_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 0.999995	04.84s	02.41s	01.83s
052_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
053_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
055_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
056_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
057_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.57s	02.07s	01.61s
058_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 0.999995	04.84s	02.41s	01.83s
059_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 0.999995	04.84s	02.41s	01.83s
060_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
061_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 0.999998	04.61s	02.37s	01.79s
062_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.57s	02.07s	01.61s
063_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 0.999996	04.84s	02.41s	01.83s
064_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 0.999999	04.61s	02.37s	01.79s
065_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
066_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 0.999995	04.84s	02.41s	01.83s
068_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1.00001	04.50s	02.39s	01.62s
069_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 0.999994	05.04s	2.61s	01.84s
070_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 0.999995	04.84s	02.41s	01.83s



PT TAUFIH
PERPUSTAKAAN TUNJUN AMINAH

<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
071_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
072_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1.00001	04.50s	02.39s	01.62s
073_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
074_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
075_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
076_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
077_T1-FLASH	160	176 x 256 x 160 voxels	1 x 1 x 0.999994	05.04s	2.61s	01.84s
078_T1-FLASH	176	176 x 256 x 176 voxels	1 x 1 x 1	04.50s	02.39s	01.62s
079_T1-FLASH	160	176 x 256 x 160 voxels	1 x 1 x 1	04.71s	02.28s	01.51s
080_T1-FLASH	160	176 x 256 x 160 voxels	1 x 1 x 1	04.71s	02.28s	01.51s
081_T1-FLASH	160	176 x 256 x 160 voxels	1 x 1 x 1	04.71s	02.28s	01.51s
082_T1-FLASH	160	176 x 256 x 160 voxels	1 x 1 x 1	04.71s	02.28s	01.51s
083_T1-FLASH	160	176 x 256 x 160 voxels	1 x 1 x 1	04.71s	02.28s	01.51s
084_T1-FLASH	160	176 x 256 x 160 voxels	1 x 1 x 1	04.71s	02.28s	01.51s
085_T1-FLASH	160	176 x 256 x 160 voxels	1 x 1 x 1	04.71s	02.28s	01.51s
086_T1-FLASH	160	176 x 256 x 160 voxels	1 x 1 x 1	04.71s	02.28s	01.51s
087_T1-FLASH	160	176 x 256 x 160 voxels	1 x 1 x 0.999999	04.70s	02.47s	01.69s
088_T1-FLASH	160	176 x 256 x 160 voxels	1 x 1 x 1	04.71s	02.28s	01.51s
089_T1-FLASH	160	176 x 256 x 160 voxels	1 x 1 x 0.999996	04.74s	02.50s	01.73s
090_T1-FLASH	160	176 x 256 x 160 voxels	1 x 1 x 0.999998	04.70s	02.47s	01.69s
091_T1-FLASH	160	176 x 256 x 160 voxels	1 x 1 x 0.999997	04.74s	02.50s	01.73s



<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
<i>092_T1-FLASH</i>	160	176 x 256 x 160 voxels	1 x 1 x 0.999998	04.70s	02.46s	01.69s
<i>093_T1-FLASH</i>	160	176 x 256 x 160 voxels	1 x 1 x 1	04.71s	02.28s	01.51s
<i>094_T1-FLASH</i>	144	176 x 256 x 144 voxels	1 x 1 x 0.999997	04.51s	02.27s	01.49s

4.0 Appended T1-MPRAGE Datasets Speed Evaluation Results

<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
<i>028_T1-MPRAGE</i>	128	208 x 256 x 128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>029_T1-MPRAGE</i>	128	208 x 256 x 128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>030_T1-MPRAGE</i>	128	208 x 256 x 128 voxels	1 x 1 x 0.999994	05.25s	02.39s	01.75s
<i>031_T1-MPRAGE</i>	128	208 x 256 x 128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>032_T1-MPRAGE</i>	128	208 x 256 x 128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>033_T1-MPRAGE</i>	128	208 x 256 x 128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>034_T1-MPRAGE</i>	128	208 x 256 x 128 voxels	1 x 1 x 0.999998	04.70s	02.61s	01.73s
<i>035_T1-MPRAGE</i>	128	208 x 256 x 128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>036_T1-MPRAGE</i>	128	208 x 256 x 128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s

<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
<i>037_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>038_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>039_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>040_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>041_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 0.999996	05.25s	02.39s	01.75s
<i>042_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 0.999999	04.70s	02.61s	01.73s
<i>043_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>044_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 0.999999	04.70s	02.61s	01.73s
<i>045_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 0.999999	04.70s	02.61s	01.73s
<i>046_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>048_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 0.999994	05.25s	02.39s	01.75s
<i>049_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 0.999994	05.19s	02.36s	01.74s
<i>050_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>051_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>052_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 0.999999	04.70s	02.61s	01.73s
<i>053_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 0.999997	04.70s	02.61s	01.73s
<i>055_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>056_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>057_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>058_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1.00001	04.70s	02.61s	01.73s



<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
<i>059_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 0.999999	04.70s	02.61s	01.73s
<i>060_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>061_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>062_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>063_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1.00001	04.70s	02.61s	01.73s
<i>064_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 0.999997	04.70s	02.61s	01.73s
<i>065_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>066_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>067_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 0.999997	04.70s	02.61s	01.73s
<i>068_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 0.999998	04.70s	02.61s	01.73s
<i>069_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>070_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>071_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>072_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>073_T1-MPRAGE</i>	128	208 x 256 x128 voxels	1 x 1 x 1	05.16s	02.61s	01.73s
<i>074_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 1	05.19s	02.36s	01.74s
<i>075_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 1	05.19s	02.36s	01.74s
<i>076_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 1	05.19s	02.36s	01.74s
<i>077_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 0.999994	05.25s	02.39s	01.75s
<i>078_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 1	05.19s	02.36s	01.74s
<i>079_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 1	05.19s	02.36s	01.74s
<i>080_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 1	05.19s	02.36s	01.74s



<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
<i>081_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 0.999993	05.25s	02.39s	01.75s
<i>082_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 1	05.19s	02.36s	01.74s
<i>083_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 1.00001	05.19s	02.36s	01.74s
<i>085_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 1	05.19s	02.36s	01.74s
<i>086_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 1	05.19s	02.36s	01.74s
<i>087_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 0.999999	05.19s	02.36s	01.74s
<i>088_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 1	05.19s	02.36s	01.74s
<i>089_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 0.999996	05.25s	02.39s	01.75s
<i>090_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 0.999998	05.25s	02.39s	01.75s
<i>091_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 0.999998	05.25s	02.39s	01.75s
<i>092_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 0.999998	05.25s	02.39s	01.75s
<i>093_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 0.999998	05.25s	02.39s	01.75s
<i>094_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 0.999997	05.25s	02.39s	01.75s
<i>095_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 1.00001	05.19s	02.36s	01.74s
<i>096_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 0.999998	05.19s	02.36s	01.74s
<i>097_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 0.999998	05.19s	02.36s	01.74s
<i>098_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 0.999998	05.19s	02.36s	01.74s
<i>099_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 0.999997	05.19s	02.36s	01.74s



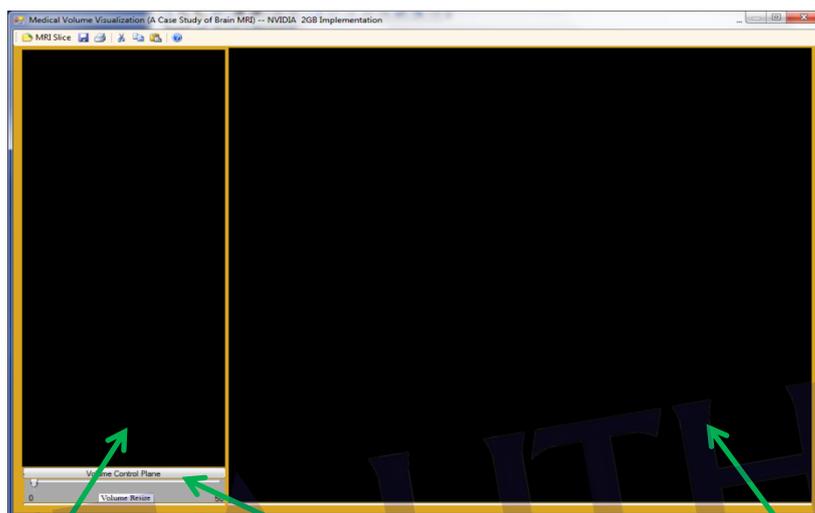
PT TAUHIM
PERPUSTAKAAN TUNKU TUN AMINAH

<i>Patient's Dataset</i>	<i>No of Images</i>	<i>Voxel Dimensions</i>	<i>Voxel Spacing</i>	<i>CPU (3GB)</i>	<i>GPU (GT520 - 1GB)</i>	<i>GPU (Quadro 600 - 2GB)</i>
<i>100_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 0.999998	05.19s	02.36s	01.74s
<i>101_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 0.999995	05.25s	02.39s	01.75s
<i>102_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 0.999999	05.19s	02.36s	01.74s
<i>103_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 1	05.19s	02.36s	01.74s
<i>104_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 0.999999	05.19s	02.36s	01.74s
<i>105_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 0.999999	05.19s	02.36s	01.74s
<i>106_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 1.00001	05.19s	02.36s	01.74s
<i>107_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 1	05.19s	02.36s	01.74s
<i>108_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 0.999998	05.25s	02.39s	01.75s
<i>109_T1-MPRAGE</i>	160	208 x 256 x160 voxels	1 x 1 x 0.999994	05.25s	02.39s	01.75s



PT TUN AMINAH
PERPUSTAKAAN TUN AMINAH

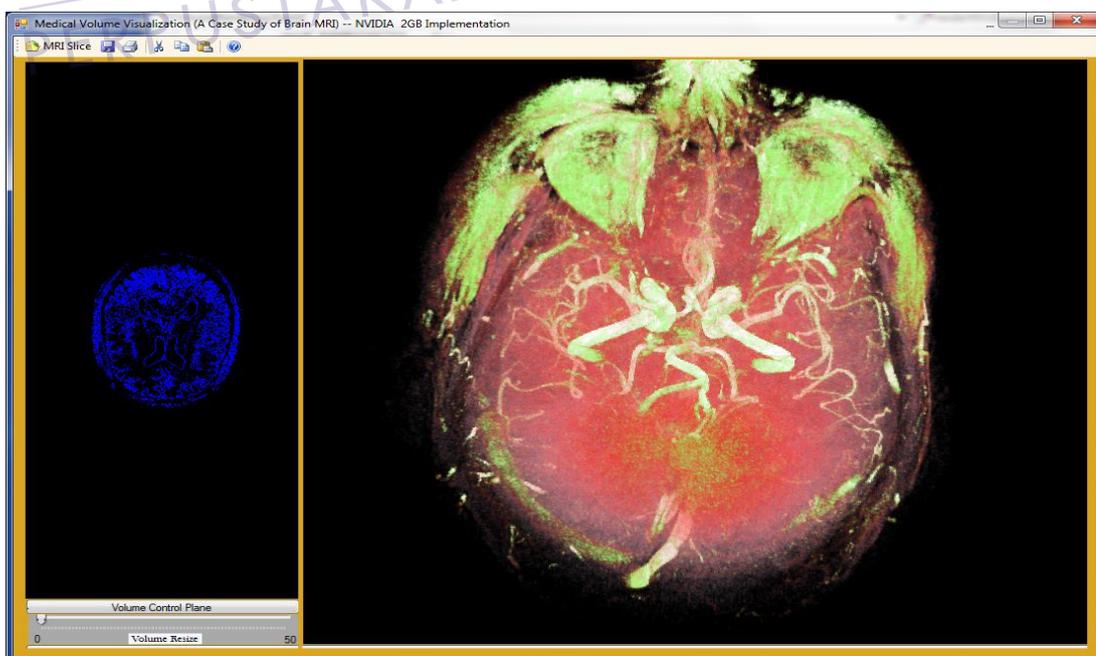
SurLens' Full Screen View



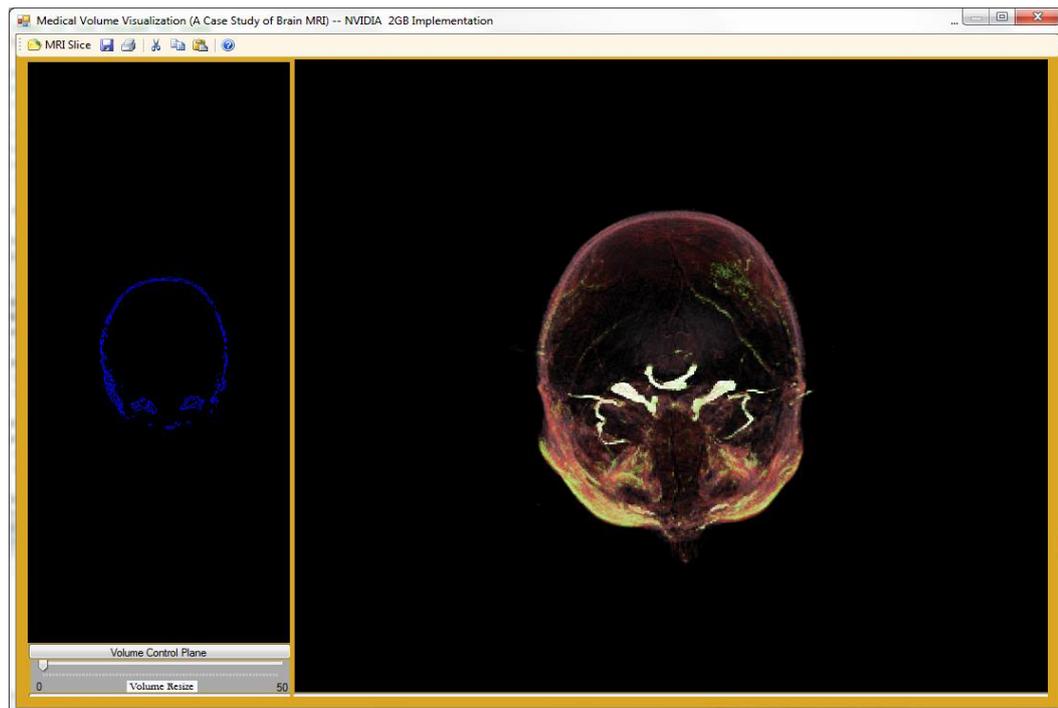
2-D Image Plane

Volume Slicer

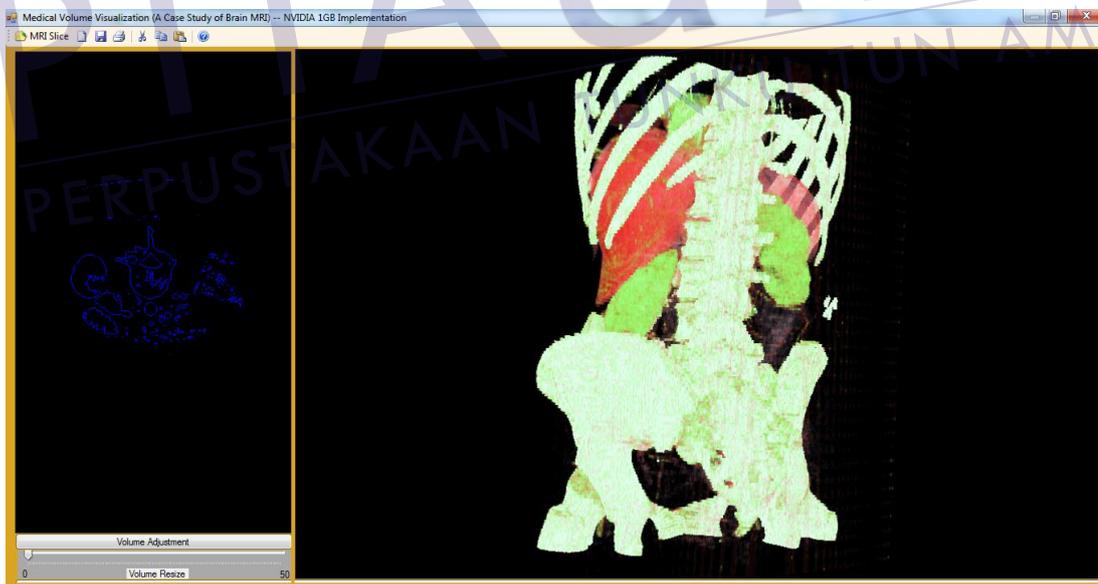
3-D Rendered Image



MRA Dataset of a Male, age 68 – Abnormal Patient with Old Brain Tumor



T1-FLASH Dataset of a Female, age 65 – Abnormal Patient



SurLens with raw CT Dataset

VITA

The author was born in April 11, 1983, in Abeokuta, Ogun State, Nigeria. He attended Baptist Boys' High School, Abeokuta, Ogun State, Nigeria for his secondary school education. He gained his Bachelor and Master's degree from University of Ilorin, Nigeria and Multimedia University, Malaysia in the year 2004 and 2009 respectively. He was a Computer Technologist at the College of Information and Communication Technology, Crescent University, Abeokuta, Nigeria from 2006 to 2007 before being promoted to academic staff in August 2009 after the conferment of his Master's Degree. During his appointment as an academic staff, the author served as both technical and functional International Information Technology consultant to organizations / firms such as UniBank Ghana, through Jethro Systems Limited, London, and Small & Micro-Enterprises Programme (SMEP), Nairobi, Kenya, through Straj Solutions Inc., Ontario, Canada. He resigned as Lecturer II in 2011 to pursue his Ph.D. programme. Mr. Adeshina has published papers in Computer Science and Information Technology Conferences and Journals. He has recently published in Journal of Computational Life Sciences, Springer, and his contribution was included in Pubmed / Medline, the database of the United States National Library of Medicine.

