



# **Post-Production of Holographic 3D Image**

By

**Obaidullah Abdul Fatah**

A thesis submitted for the degree of

Doctor of Philosophy

in

Electronic & Computer Engineering

College of Engineering, Design and Physical Sciences

Brunel University

January 2015

## Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revision, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public; therefore I authorise Brunel University to make available electronically to individual or institution for the purpose of scholarly research.

Signature:.....

Obaidullah Abdul Fatah

Date:.....

I further authorise Brunel University to reproduce this thesis by photocopying or by any other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Signature:.....

Obaidullah Abdul Fatah

Date:.....

**Copyright 2015 Obaidullah Abdul Fatah**

**All Right Reserved**

## **ABSTRACT**

Holographic 3D imaging also known as “Integral imaging” was first proposed by Lippmann in 1908. It facilitates a promising technique for creating full colour spatial image that exists in space. It promotes a single lens aperture for recording spatial images of a real scene, thus it offers omnidirectional motion parallax and true 3D depth, which is the fundamental feature for digital refocusing.

While stereoscopic and multiview 3D imaging systems simulate human eye technique, holographic 3D imaging system mimics fly’s eye technique, in which viewpoints are orthographic projection. This system enables true 3D representation of a real scene in space, thus it offers richer spatial cues compared to stereoscopic 3D and multiview 3D systems.

Focus has been the greatest challenge since the beginning of photography. It is becoming even more critical in film production where focus pullers are finding it difficult to get the right focus with camera resolution becoming increasingly higher. Holographic 3D imaging enables the user to carry out re/focusing in post-production. There have been three main types of digital refocusing methods namely Shift and Integration, full resolution, and full resolution with blind. However, these methods suffer from artifacts and unsatisfactory resolution in the final resulting image. For instance the artifacts are in the form of blocky and blurry pictures, due to unmatched boundaries. An upsampling method is proposed that improves the resolution of the resulting image of shift and integration approach. Sub-pixel adjustment of elemental images including “upsampling technique” with smart filters are proposed to reduce the artifacts, introduced by full resolution with blind method as well as to improve both image quality and resolution of the final rendered image.

A novel 3D object extraction method is proposed that takes advantage of disparity, which is also applied to generate stereoscopic 3D images from holographic 3D image. Cross correlation matching algorithm is used to obtain the disparity map from the disparity information and the desirable object is then extracted. In

addition, 3D image conversion algorithm is proposed for the generation of stereoscopic and multiview 3D images from both unidirectional and omnidirectional holographic 3D images, which facilitates 3D content reformation.

## **ACKNOWLEDGEMENTS**

I would like to take this opportunity to thank my supervisor Professor Abdul Sadka, as this achievement would not be possible without his efforts and support. His knowledge, discipline, availability, patience, constructive criticism of my work and support are unique. He has been very supportive in my research work at Brunel. He was always there for discussion when you needed the most.

I would also like to express my sincere gratitude to my previous supervisor, Dr Amar Aggoun, for his support. He is well respected in the area of Holographic 3D. His supervision and visionary advice helped a lot toward completing this research work.

I would like to thank all CMCR members and 3DVIVANT project team members of Brunel University. It has been great working with them all. Thanks are also due to Rafiq Swash for being always available to discuss my research and to support me when I was down.

I would also like to thank everyone who was around me during my PhD research and supported me when I needed them the most.

Last but not the least, I am grateful to the support given by my family members throughout my PhD studies. They deserve special thanks, as I would not have been able to face the challenges of this research work. I appreciate all their supports and prayers.

**TABLE OF CONTENT**

ABSTRACT .....i

ACKNOWLEDGEMENTS ..... iii

TABLE OF CONTENT ..... iv

TABLE OF FIGURE ..... viii

1 Chapter One..... 1

    1.1 The Research Area..... 2

    1.2 Aim and Objectives..... 3

    1.3 The Original Contributions..... 5

        1.3.1 Digital Refocusing in Holographic 3D images..... 5

        1.3.2 3D Image Generation from Holographic 3D Image..... 5

        1.3.3 Object extraction based on Depth map..... 6

    1.4 Journal and Conference Papers ..... 7

        1.4.1 Journal Papers ..... 7

        1.4.2 Conference Papers ..... 7

    1.5 The Thesis Outlines.....10

    1.6 References.....11

2. Chapter Two.....12

    2.1 3D Imaging Technologies.....12

    2.2 3D Imaging Technologies.....13

    2.3 Stereoscopic 3D .....14

        2.3.1 Anaglyph .....14

        2.3.2 Polarisation.....15

        2.3.3 Time division.....16

    2.4 Autostereoscopic 3D .....17

        2.4.1 Holography .....17

        2.4.2 Volumetric displays .....19

2.4.3	Multi-view displays.....	20
2.4.4	Integral Imaging (II).....	23
2.5	Plenoptic Camera .....	28
2.6	Digital Refocusing .....	33
2.7	Conclusion.....	35
2.8	Reference.....	36
3.	Chapter Three.....	44
3.1	Operational characteristics of the holographic 3D imaging system .....	45
3.2	Flowchart of the proposed method.....	46
3.3	Viewpoint Construction in Holographic 3D Imaging System.....	48
3.3.1	Lens error correction before viewpoint image extraction.....	52
3.4	Digital Refocusing .....	54
3.5	Depth Analysis.....	61
3.6	All-in-focus image .....	62
3.7	Experimental Results.....	64
3.7.1	Subjective Quality Comparison.....	65
3.8	Analysis and Discussion .....	67
3.9	Conclusion.....	70
3.10	References .....	73
4.	Chapter Four .....	75
4.1	Full Resolution Basic Rendering .....	76
4.1.1	Full Resolution Rendering Algorithm: .....	78
4.2	Up-sampling, Shift and Integration with Full Resolution Method .....	78
4.3	Experimental Results and Observations.....	83
4.4	Sub-pixel Adjustment Technique (SPA).....	86
4.5	Experimental Results and Observations.....	89
4.5.1	Comparison of Subjective Image Quality .....	89



4.5.2	Analysis and Discussion.....	93
4.6	Conclusion.....	97
4.7	References.....	98
5.	Chapter Five .....	99
5.1	Flow chart of The Proposed Method .....	100
5.2	Image Correction on Pre-processing .....	101
5.3	Image Smoothing on Post-Processing .....	106
5.3.1	The Background.....	107
5.3.2	L0 Gradient Minimisation .....	108
5.3.3	The Smoothing Technique .....	109
5.4	Experimental Results and Observations.....	110
5.4.1	Subjective quality comparison.....	112
5.5	More Analysis of Resulting images.....	116
5.6	Subjective Assessment of Image Quality.....	121
5.6.1	Review of Subjective Quality Assessment Methods.....	121
5.6.2	Description of Assessment Methods.....	122
5.6.3	Presentation of the test material.....	123
5.6.4	Equipment used and viewing conditions.....	123
5.6.5	Databases of Subjective results & test materials .....	124
5.6.6	Subjective Participants.....	128
5.6.7	Subjective Protocol.....	128
5.6.8	Subjective Grading.....	129
5.6.9	Mean opinion scores .....	130
5.6.10	Results and Analysis.....	131
5.7	Conclusion.....	141
5.8	References.....	143
6.	Chapter Six.....	147

6.1	Camera .....	148
6.1.1	2D Display .....	150
6.1.2	Experimental Results and Observations .....	152
6.2	3D Stereoscopic Display .....	153
6.2.1	Experimental Results and Observations .....	156
6.2.2	Analysis and Discussion.....	158
6.2.3	Experimental Results and Observations .....	159
6.3	Autostereoscopic 3D Display.....	162
6.3.1	Experimental Results and Observations .....	166
6.4	Object Extraction Based On Depth map .....	169
6.5	Proposed holographic 3D object segmentation.....	170
6.6	Experimental Results and Observations.....	172
6.7	Conclusions.....	178
6.8	References.....	179
7.	Chapter Seven.....	180
7.1	Conclusion.....	181
7.2	Further work.....	185
7.3	Appendix A.....	174
7.4	Sample Questionnaire .....	174

**TABLE OF FIGURES**

Fig 2. 1 : Anaglyph from [65].....15

Fig 2. 2 : Polarisation from [66]. .....16

Fig 2. 3 : Time Division from [67].....17

Fig 2. 4 : Holography from [71].....18

Fig 2. 5 : Volumetric from [68] .....20

Fig 2. 6 : Parallax barrier from [69].....21

Fig 2. 7 : Lenticular sheet from [69] .....22

Fig 2. 8 : Principle of Integral Photography [72] .....23

Fig 2. 9 : Principle of two recording [28]. (a) a second stage recording of integral photograph. (b) Replay and viewing of orthoscopic image scene.....24

Fig 2. 10 : The advance integral imaging system from [67] .....25

Fig 2. 11 : Diagrammatic representation of the lens array from [42] .....26

Fig 2. 12 : Traditional Plenoptic camera from [46] .....28

Fig 2. 13 : Behaviour of traditional plenoptic camera where the spatial resolution in  $\partial$  direction is limited by the number of microlenses however, directional information  $\Omega$  is determined by the number of pixels under each element image.....29

Fig 2. 14 : Focused Plenoptic 2.0 microlens imaging in Keplerian mode. The main lens image plane is in front of the microlenses .....30

Fig 2. 15 : Behaviour of Focused Plenoptic camera.....31

Fig 3. 1 : Holographic 3D imaging principle: (a) recording and (b) display process  
46

Fig 3. 2 : Flowchart of the proposed method .....47

Fig 3. 3 : Holographic 3D Images with its constructed viewpoint image .....48

Fig 3. 4 : Holographic 3D capturing systematic.....49

Fig 3. 5 : illustration of (a) UH3DI viewpoint image extraction, (b) OH3DI viewpoint image extraction.....50

Fig 3. 6 : Illustration of the viewpoint (VP), elemental image (EI) and sub-image (SI).....51

Fig 3. 7 : Illustration of barrel distortion effect.....52

Fig 3. 8 : (a) VI (25, 25) without barrel distortion correction. (b) The same VI with barrel distortion correction .....53

Fig 3. 9 : Representation of lens types [17] .....53

Fig 3. 10 : Illustration of Photoshop lens correction tool.....54

Fig 3. 11 : Focused Plenoptic 2.0, Microlens imaging in Galilean mode. The main lens image plane is virtually created behind the image sensor for the microlenses to capture. ....55

Fig 3. 12 : Focused Plenoptic 2.0: Microlens image is in Galilean mode. Full resolution rendering applied on the above images: left image extracts foreground but leaves the background with artifacts as the size of sub-images is too small for the background, where .....56

Fig 3. 13 : (a) Illustration of shift and integration of VPs to generate one high resolution image with a final image size of  $(n \times N) \times (m \times N)$ . (b) Present the steps for the proposed refocused image. ....57

Fig 3. 14 : Schematic illustration of resolution increase .....58

Fig 3. 15 : Ray tracing in holographic 3D imaging system .....60

Fig 3. 16 : Depth Analysis .....61

Fig 3. 17 : All-in-focus image rendering algorithm creates a final image using high resolution images .....63

Fig 3. 18 : Illustrates one of the experiment setup scenes.....64

Fig 3. 19 : Native shift and integration refocusing is illustrated: (a) shows the magnified part of final refocusing image where the focus is at the object. (b) is focused at the background: Notice both (a) and (b) images are in poor quality, containing blocking artifacts with significant noise that is seen more pixelated with naked eyes.....65

Fig 3. 20 : Up-sampling, shift and integration refocusing using 7 by 7 VPs: (a) shows the magnified part of Target 1 that is the ARRI Media test chart. (b)Target 4 is focused at the toy. The final image is at resolution of 1344 x 903 pixels.....66

Fig 3. 21 : Up-sampling, shift and integration refocusing using 7 by 7 VPs:(a) shows the magnified part of Target 1 that is the ARRI Media test chart. (b)Target 4 the blur looks natural and no artifacts.....66

Fig 3. 22 : Rendered VP of UH3DI.....68

Fig 3. 23 : 7 by 7 VP are used in refocusing process to extract different depth plane in (a). In (b) where focus is on the background with virtual depth  $z = -0.518\text{mm}$  and (c) focuses at foreground with virtual depth  $z = -4.147\text{mm}$ .....69

Fig 3. 24 : In (a) all-in-focus image is extracted using different planes with window size 20 by 20 pixels.(b) is the virtual depth map from the microlenses to the main lens with the given virtual distant the real depth can be calculated in relation to the main. ....70

Fig 3. 25 : In (a) 7 by 7 VPs to focused at the background. (b) Focused at the distant 100mm. (c) using 5 by 5 VPs focused at background. (d) Focused at 1 meter from the microlens array.....71

Fig 3. 26 : Result of All-in-focus with disparity .....72

Fig 4. 1 : The full resolution rendering algorithm resulting image at resolution of  $n^*SI$  by  $m^*SI$  77

Fig 4. 2 : The full resolution rendering algorithm is shown with two different views by shifting the axis of  $j$  by 1 which extracts a full resolution view 2 with the same size of  $SI$ . ....80

Fig 4. 3 : Graphically illustration of 2 by 2 full resolution rendering with up-sampling of 2 in  $x$  and  $y$  direction.....82

Fig 4. 4 : A small zoomed portion of OH3DI and red squares shows the view pickup position under each EI.....84

Fig 4. 5 : Signal 2D view at 1344 x 903 resolutions using 7 by 7  $SI$  size under each EI having the test chart 'in-focused'.....84

Fig 4. 6 : Graphical illustration of SPA .....88

Fig 4. 7 : Up-sampling, shift and integration refocusing using 7 by 7 VPs: (a) shows the magnified part of ARRI Media test chart. (b) Is the foreground looks, as it should be. ....90

Fig 4. 8 : The SPA on high resolution rendering with up-sampling, shift and integration (a). (b) Magnified region of the test chart 'in-focus' and (c) illustrate the foreground object.  $SI = 7, \delta = 6$ , number high resolution views used 10. ....90

Fig 4. 9 : The SPA on high resolution rendering with up-sampling, shift and integration (a). (b) Magnified region of the test chart 'in-focus' and (c) illustrate the foreground object.  $SI = 2, \delta = 2$ , number high resolution views used 7.....91

Fig 4. 10 : The SPA on high resolution rendering with up-sampling, shift and integration (a). (b) Illustrate the foreground object.  $SI = 2, \delta = 7$ , number high resolution views used 7. ....91

Fig 4. 11 : Rendered image using up-sampling, shift and integration with high resolution views, focused at background with  $SI = 7$  by  $7$ , Size of EI =  $74$  by  $74$ . Red square is the result of up-sampling, shift and integration and the yellow square is rendered high resolution with Blending [2].....92

Fig 4. 12 : (a) Rendered focused at seagull with  $SI = 9$  by  $9$ , Size of EI =  $74$  by  $74$ . (b) Red square is the result of up-sampling, shift and integration. (c) Yellow square result is rendered with high resolution with Blending [2].....92

Fig 4. 13 : (a) Rendered image using SPA. (b) Magnify red square region of the rendered image (a). ....93

Fig 4. 14 : (a) Rendered image without delta. (b) Magnify red square region of the rendered image (a) containing artifacts. ....93

Fig 4. 16 : demonstrate refocusing with constant SI size in SPA by changing the value of delta in the process.....96

Fig 4. 15 : illustrates in changing the depth plane from background in (a) to foreground in (f). ....96

Fig 5. 1 : Illustrate the three steps pre-process, high resolution refocusing and post-processing respectively in achieving the final enhanced image. 100

Fig 5. 2 : (a) Gaussian or normal distribution and (b) Gaussian distribution truncated at point  $\pm 3 \sigma$ ..... 101

Fig 5. 3 : Presentation of 1D Signal. (a) original signal, (b) smooth signal, (c) is the difference between (a) and (b), (d) is (a) minus (b) plus (a) and (e) is the complete filtering operator. .... 103

Fig 5. 4 : 2D Gaussian or normal distribution..... 104

Fig 5. 5 : Illustrates the results of pre-processing with different values sigma. (a) section of portion of the original HI. (b) low-pass image, (c), (d), (e) and (f) are the result of  $s''_{i,j}$  with different sigma value and with 10 by 10 kernel. .... 106

Fig 5. 6 : Result obtained for both details and sharp edges of an image. (a) Bilateral filtering. (b) Anisotropic diffusion used in the LCIS system. (c) Weighted least square optimization. (d) Total variation smoothing. (e) Smoothing via L0 gradient minimisation..... 108

Fig 5. 7 : Visual representation created by Farbman. (a) Color visualized Noisy input image (b) Result of Subr et al [11]. (c) Result of Bilateral filtering (BLF) with ( $\sigma_s=12$ ,  $\sigma_r=0.45$ ). (d) Weighted least square optimization ( $\alpha=1.8$ ,  $\lambda=0.35$ ). (e) Smoothing via L0 gradient minimisation ( $\lambda=0.01$ ). ..... 110

Fig 5. 8(a) : Illustration of image focused on the background. (SI = 2,  $\delta = 6$  and 7 by 7 different views). ..... 112

Fig 5. 9 : (a) Shows the image plane focused on the foreground object with set parameters (SI = 2,  $\delta = 1$  and 7 by 7 different views). (b) is the result from (a) using image smoothing via via L0gradient minimisation ( $\lambda=0.007$ )..... 113

Fig 5. 10 : Illustrates results of smoothing via L0gradient minimisation with different lambda starting from 0.005, 0.007, 0.009, 0.01, 0.02, 0.03, 0.04 and 0.09 respectively and other parameter are set the same throughout in this figure where the focus is set to the background with set parameter, SI = 2,  $\delta = 6$  and 7 by 7 different views..... 114

Fig 5. 11 : Illustrates results of smoothing via L\_0 gradient minimisation with different lambda starting from 0.005, 0.007, 0.009, 0.01, 0.02, 0.03, 0.04 and 0.09 respectively and other parameter are set the same throughout in this figure where the focus is set to the foreground object with set parameter, SI = 2,  $\delta = 1$  and 7 by 7 different views..... 115

Fig 5. 12 : Results of smoothing via L\_0 gradient minimisation with different lambda with 0.007, 0.01, 0.02, 0.03, 0.04 and 0.09 respectively and other parameter are set the same throughout in this figure. Parameter, SI = 4,  $\delta = 4$ , microlens pitch = 90 $\mu$ m, focal length = 1mm, EI size 29 by 29 pixel resolution. .... 117

Fig 5. 13 Our final refocused image results with different lambda values. .  $\lambda =$  (0.007, 0.01, 0.02, 0.03, 0.04 and 0.09) respectively, SI = 4,  $\delta = 2$ , 7 by 7 different views, microlens pitch = 90 $\mu$ m, focal length = 1mm..... 118

Fig 5. 14 : (a) Post- processing result with  $\lambda$  set to 0.09 focuses on the test chart. (b) shows the close-ups results in (a) that is illustrate with two square color blue and red..... 119

Fig 5. 15 : (a) Post- processing result with  $\lambda$  set to 0.007 focuses on the test chart. (b) close-ups results in (a) that is illustrate with two square color blue and red..... 120

Fig 5. 16 : Structure of presentation for subjective test material ..... 123

Fig 5. 17 : Illustration of dataset used in testing, (a) OHI Image A with micro image size 29x 29 pixels, (b) OHI image B with micro image size 29x 29 pixels, (c) OHI image C with 74x 74 pixels and OHI image D with 29x 29 pixels with different lighting condition than image A. .... 124

Fig 5. 18 : Participants distribution in term of age and gender using bar chart ..... 132

Fig 5. 19 : Gender representation in pie chart..... 132

Fig 5. 20 : Representation of image quality in term of gender in line graph..... 133

Fig 5. 21 : Knowledge of Holoscopic 3D in bar chart ..... 133

Fig 5. 22 : MOS values for image quality using Algorithm 1 represented in bar chart..... 134

Fig 5. 23 : MOS values for image quality using Algorithm 2 represented in bar chart..... 136

Fig 5. 24 :MOS values for image quality using Algorithm 3 represented in bar chart..... 137

Fig 5. 25: MOS value for image quality on all the test images using Algorithm 4 ..... 138

Fig 5. 26 : MOSA values for image quality..... 140

Fig 6. 1 : (a) First version of holoscopic adapter for Alexa camera. (b) A holoscopic 3D camera prototype based on Canon 5D MKII [7]..... 149

Fig 6. 2 : Sample of acquired holoscopic 3D sequence from Alexa camera at resolution of 2880 x 1620 pixels. .... 150

Fig 6. 3 : Alexa XT camera with microlens built close to the image sensor and capture frames at resolution of 3424 x 2202 pixels [7]. .... 150

Fig 6. 4 : The workflow block diagram. (a) Scene capture with (b) holoscopic camera. (c) Frames are rendered in post-production and playing on 2D display (d). .... 151

Fig 6. 5 : (a) A frame sample of a sequence, (b) the magnified version..... 153

Fig 6. 6 : (a) 2D result of rendered frame no.14 and (b) rendered frame no.18. .... 153

Fig 6. 7 : Illustration of baseline distance in single Element Image (EI). .... 155



Fig 6. 8 : Rendered sequences - (a) is the first rendered frame no. 3 of the sequence and (b) is the second view. (c) is the first view and (d) is the second view of the rendered frame no. 18<sup>th</sup> of the sequence. .... 157

Fig 6. 9 : Stereo 3D resulting images in anaglyph view. (a) Result of Fig 6.8(a)-(b) and (b) is result of Fig 6.8(c)-(d)..... 157

Fig 6. 10 : Stereo 3D image pair rendered from a holographic 3D image ..... 158

Fig 6. 11 : (a) shows the captured image from holographic camera at resolution of 5466 x 3588. (b) Shows a small portion of (a) high lighting it with blue square around it..... 159

Fig 6. 12 : The result of stereo 3D image from holographic content. (a) Crop area of the first view. (b) Shows both stereo views and (c) is the anaglyph result of the both view in (b). .... 160

Fig 6. 13 : The rendered stereo 3D images from holographic 3D images with (a)-(b) focusing at background and (e)-(d) at foreground object. (c)-(d) are the results of both in anaglyph. .... 161

Fig 6. 14 : Display size 47inches multiview display from Alioscopy ..... 162

Fig 6. 15 : The workflow graphically. (a) Scene capture with (b) holographic camera. (c) Frames are rendered in post-production and playing on 2D display (d)..... 163

Fig 6. 16 : shows single EI with position of 6 views ..... 165

Fig 6. 17 : Illustration of 8 extracted views from single holographic frame no 3. .... 167

Fig 6. 18 : Rendered multiview 3D content displayed on Alioscopy multiview screen..... 168

Fig 6. 19a : Results are demonstrated on the multiview display when focusing on the background ..... 169

Fig 6. 20 :An Illustration of block matching methods..... 171

Fig 6. 21 : Demonstration of extraction of object from V<sub>1</sub> based on disparity information..... 172

Fig 6. 22 : (a) show stereo views, (b) is disparity result with w=15, R= 35. (c) illustrates object O overlaid with disparity d of (b) and (d) is the final result of object O..... 174

Fig 6. 23a: (a)and(b) is stereo views and (c) is the disparity result of the stereo view..... 175

Fig 6. 24 : The result of object O is within disparity ranging from  $E_{min} = 43$  and  $E_{max} = 36$  shown in (d). ..... 176

## **List of Acronyms**

DMU	-	De Montfort University
3D	-	Three Dimensional
2D	-	Two Dimensional
II	-	Integral Imaging
HOE	-	Holographic Optical Element
CCD	-	Charge-Coupled Device
LCD	-	liquid Crystal Display
OII	-	Omnidirectional Integral Images
OHI	-	Omnidirectional Holographic Images
OH3DI	-	Omnidirectional Holographic Three Dimensional Image
H3DI	-	Holographic Three Dimensional Image
UII	-	Unidirectional Integral Image
UHI	-	Unidirectional Holographic Image
VPIs	-	Viewpoint Images
SVPI	-	Single Viewpoint image
SVPE	-	Single Viewpoint Extraction
VP	-	Viewpoint
VPs	-	Viewpoints
SI	-	Sub-Image
SIs	-	Sub-images
EI	-	Element Image

SPA	-	Sub-Pixel Adjustment
HI	-	Holoscopic Image
FG	-	Foreground
BG	-	Background
HRI	-	High Resolution Image
MOS	-	Mean Opinion Scores
MOSA	-	Mean Opinion Scores Algorithm
SSD	-	Sum of Square Differences
SD	-	Standard Definition
SS	-	Single Stimulus
DS	-	Double Stimulus

# **1 Chapter One**

## **Introduction**

This chapter presents the PhD research aim and objectives, including a brief background of the research. In addition, it discusses the original contributions of the research and the outline of the thesis.

## **1.1 The Research Area**

Content creators are constantly pursuing and searching for enterprise based and more innovative methods of improving and delivering more sensational ways of enhanced media content, one of which has been High Definition Television (HDTV). The greatest innovation in film-making is 3D production that enhances the viewing sensation to realise the perception of depth. Since then, 3D imaging systems have been on a constant pursuit both in scientific community and entertainment industry [1]. One of the significant focuses in the area of image processing is 3D imaging in the present day. Based on the current trend, 3D technology has the potential to establish the future mass-market in the area of entertainment, medical, military and design.[2]

In the past, researchers have attempted different approaches to achieve the perception of depth. One of earlier 3D methods is stereoscopic 3D that is still widely used in today's 3D systems. It works on the principle of projecting both right and left images down to viewer's left and right eye respectively to realize the 3D effect. This technique requires special glasses to correctly channel the left and right images to the corresponding eye of the viewer. This makes the brain to fuse two different perceptive images to create a 3D sensation. However, this strains the viewer's eyes, which results in headaches after long periods of exposure [5]. Also, stereoscopic 3D productions with dual cameras are a ponderous and expensive method due to huge amount of work to be done in the post-production and this makes the technology unsuitable for mass-market. Many different companies in the past have attempted to address the complexity of 3D production without modifying the basic notion of dual capturing device [6].

The growth in 3D TV has not been as momentous as HDTV. This being partly due to the fact that physical 3D glasses are still needed to perceive 3D cues, which is not comfortable for home users in particular. Recently, Liquid crystal display manufactures are searching for new ways of displaying 3D without the need of wearing 3D glasses where technologies like multiview, holographic and holographic (Integral image) displays have emerged and that is considered as a major tipping point as it makes the 3D technology suitable to domestic arenas.

Multiview 3D technology utilizes stereoscopic 3D technique that projects more than two perspective views using parallax barrier or lenticular technology. However viewers still suffer from motion sickness, eye fatigue and unnatural image quality [5]. Holographic and holoscopic 3D, on the other hand, offers true 3D imaging systems [2].

Holographic imaging offers full colour high-resolution images with an ultimate 3D viewing sensation that overcomes the limitation of stereoscopic 3D imaging. Now the question is “why the implementation of holographic 3D TV remains of great interest to the research community?”. Unfortunately, holographic images require a coherent light source and a confined dark room whilst recording, which makes it impractical for live capturing whereas Holoscopic 3D imaging is capable of recording true 3D information by its unique optical component. In addition, holoscopic 3D imaging does not require a coherent light source, thus making it more practical for live capturing and displaying. Furthermore it offers a free viewing sensation to more than one person simultaneously independent of viewers’ position without causing eye fatigue. Recent developments in the field of holoscopic 3D imaging have gained huge interest and acquired good candidacy for being the future generation 3D imaging technique [2].

## **1.2 Aim and Objectives**

The aim of this research is to enhance post-production techniques of holoscopic 3D imaging systems, which delivers a richer representation of the scene compared to stereoscopic 3D and 2D production. Post-production techniques are highly beneficial for various applications in 2D and 3D video production. Today, inventive decisions are taken partly during shooting and partly in post-production. In many cases, problems that may arise during the shooting could be fixed in post-production. However, content captured with a holoscopic 3D camera provides a new possibility to re-assign many critical decisions during the production stage to the post-production stage.

Focus is a prominent example, since this has been the greatest challenge from the birth of photography. It is becoming even more critical in film production where focus pullers are finding it difficult to get the right focus, with camera

resolution becoming increasingly higher. An exacerbating factor is that mis-focusing is generally hard to correct in the post-production stage and expensively time consuming. Content with mis-focus is generally useless and requires to be recaptured again with the correct focus for it to be of any use. This is a quite time consuming and expensive process. However, content captured with holographic 3D camera that contains a richer representation of the scene allows the focus to change after image was captured.

The other challenge in stereoscopic 3D production is the cameras interocular distance as the interocular distance defines the 3D depth in the production. Smaller interocular distance leads to less roundness of the scene, whereas larger interocular distance leads to more roundness. The filmmaking director might want to make decision not on the set but at a later stage in the production process. The reason might be to maintain consistency from shot to shot, which in many cases is not being captured in the same order as they appear in final movie. Setting the interocular distance on a set requires extra personnel and time.

This research finds ways of resolving longstanding problems of mis-focusing and also the interocular distance in today's film production. Holographic 3D imaging can compensate for the mis-focusing problem with one of its refocusing applications. This gives the opportunity to change the focus plane within a desirable range at the post-production stage and therefore this offers the ability to change or make creative decisions after capturing, which is a key benefit for such setups. In addition, holographic 3D imaging offers the interocular distances adjustment in the post-production. The interocular distance obtained from holographic 3D images resolves the problems with the near objects; whereas a different approach such as the wider view angle microlens with different camera calibration approach might be needed to deal with the far objects.



### **1.3 The Original Contributions**

This PhD research incorporates the following contributions.

#### **1.3.1 Digital Refocusing in Holographic 3D images**

A sub-pixel adjustment algorithm with a smart filter is proposed to reduce blocky artifacts and to improve the image resolution.

Recently, researchers have addressed various ways of performing digital refocusing after capturing photography with holographic 3D imaging system. But still the resulting image suffers from blocky artifacts and resolution. Two novel methods were proposed to address the issue. The first approach is based on upsampling of orthographic viewpoint images with shift and integrating method. The upsampling defines the resolution of the output 2D image. On the other hand, the level of the shifts defines the focus plane on the final image.

The second approach is based on extracting sub-images under element images to create a higher resolution of viewpoint images and refocusing is achieved through the choice of sub-images size under the each microlens, as addressed in [4]. However, this approach suffers from various artifacts for which, a new algorithm is proposed to reduce the artifact “blocky noise”. The proposed approach is based on a sub-pixel adjustment and smart filtering technique to blur out the out of focus regions in the output image that contains artifacts and also to decrease blur on the focus regions.

#### **1.3.2 3D Image Generation from Holographic 3D Image**

This section presents a 3D image conversion algorithm, which generates a stereoscopic 3D image pairs as well as multiview 3D images from a single holographic 3D image.

In the past, researchers have proposed stereoscopic 3D capturing with a single aperture holographic 3D lens. However, the results are observed with poor resolution as well as aliasing effect. The proposed method compensates both the above issues through using a sub-pixel adjustment algorithm with a smart filter. Moreover, the problem of ocular distance has been addressed and an

alternative method have been proposed to improve the results with improved optical parameters and better pixel resolution on motion picture camera.

Furthermore, Multiview 3D creation is rather a complex process and is becoming gradually more expensive as it requires multiple camera configurations and synchronisation [3]. However, holographic 3D imaging system is used to capture 3D content, which can be parsed to multiview 3D format and this simplifies the whole multiview 3D creation. A number of views are extracted based on the multiview 3D display requirements. The views are rendered using the sub-pixel adjustment algorithm with a smart filter, which is proposed in this thesis.

Recently researchers successfully demonstrated the process of 3D capturing for holographic 3D display with the help of image processing, though it has so far been attempted on still images. This was the first attempt to shoot moving images to exhibit the whole holographic 3D capturing process as well as parsing the content to various visualization systems such as HD 2D, S3D and multiview 3D. This illustrates that holographic 3D camera could be the ideal solution to the existing 3D content creation limitation as its content can be parsed computationally to different visualization systems. This approach provides a simple and efficient way of capturing real 3D content [1].

### **1.3.3 Object extraction based on Depth map**

An object extraction algorithm is proposed that segments tangible objects based on disparity map, which is created from the holographic 3D image.

Image segmentation and object extraction has been a challenging topic for many years and researchers proposed numerous ways of performing image segmentation and object extraction. A very simple way of object extraction is proposed that is based on disparity map. This method works by extracting stereo images from holographic 3D image, where cross correlation matching is used to obtain the disparity map from those stereo images. Finally, this disparity map is used to extract a desirable object from the image. Experimental results are shown on the final extracted object with minor errors.

## **1.4 Journal and Conference Papers**

### **1.4.1 Journal Papers**

- i. **O. Abdul Fatah**, A. Aggoun, M.R. Swash, B. Li, J. C. Jacome, E. Tseklevs “Refocusing and all-in-focus image based on single aperture 3D holographic imaging camera”, Special Issue on Future of Broadcast Television: Systems, Services and Technologies, IEEE Transactions on Broadcasting. 2013 (Submitted).
- ii. M.R. Swash, A. Aggoun, **O. Abdul Fatah**, B. Li, J. C. Jacome, E. Tseklevs “Dynamic Hyperlinker for H3D Video Search and Retrieval”. Special Issue on Future of Broadcast Television: Systems, Services and Technologies, IEEE Transactions on Broadcasting 2013 (Submitted).
- iii. M.R. Swash, A. Aggoun, **O. Abdul Fatah**, B. Li, J. C. Jacome, E. Tseklevs “Holographic 3D Image Re-formatting for Autostereoscopic 3D Displays”, Special Issue on Future of Broadcast Television: Systems, Services and Technologies, IEEE Transactions on Broadcasting. 2013 (Submitted).
- iv. B. Li, M.R. Swash, A. Aggoun, **O. Abdul Fatah**, J. C. Jacome, E. Tseklevs “Nonlinear Distortion Correction for a Single Aperture 3D Integral Imaging Camera ” Optic Express (Submitted).

### **1.4.2 Conference Papers**

- i. **O. Abdul Fatah**, A. Aggoun, M.R. Swash, B. Li, J. C. Jacome, E. Tseklevs, “Holographic 3D Image Re-formatting for stereoscopic 3D Displays”, 3DTV-CON: Vision beyond Depth AECC, Aberdeen, Scotland, 7-9th October 2013
- ii. M.R. Swash, A. Aggoun, **O. Abdul Fatah**, B. Li, J. C. Jacome, E. Alazawi, E. Tseklevs “Pre-Processing of Holographic 3D Image For Autostereoscopic 3D Display”, 5th International Conference on 3D Imaging (IC3D). 2013
- iii. **O. Abdul Fatah**, P. Lanigan, A. Aggoun, M.R. Swash, E. Alazawi, B. Li, J. C. Jacome, D. Chen, E. Tseklevs, “Three-Dimensional Integral Image reconstruction based on viewpoint interpolation”, IEEE International Symposium on Broadband Multimedia Systems and Broadcasting, London, UK, June 2013.

- iv. M.R. Swash, A. Aggoun, **O. Abdul Fatah**, E. Alazawi, E. Tseklevs, "Distributed Pixel Mapping for Refining Dark Areas in Parallax Barriers Based Holographic 3D Displays", 5th International Conference on 3D Imaging (IC3D). 2013
- v. **O. Abdul Fatah**, A. Aggoun, M. Nawaz, J.Cosmas, E.Tseklevs, M. R. Swash, E. Alazawi, "Depth mapping of Integral images using a hybrid disparity analysis algorithm", IEEE International Symposium On Broadband Multimedia Systems and Broadcasting 2012, Seoul, Korea 27th Jun 2012
- vi. E. Alazawi, A. Aggoun, M. R. Swash, **O. Abdul Fatah**, M. Abbod and J. Fernandez "Scene Depth Extraction From Holographic Imaging Technology", 3DTV-CON: Vision beyond Depth AECC, UK, 2013
- vii. M.R. Swash, A. Aggoun, **O. Abdul Fatah**, B. Li, J. C. Jacome, E. Alazawi, E. Tseklevs, "Moiré-Free Full Parallax Holographic 3D Display based on Cross-Lenticular", 3DTV-CON: Vision beyond Depth AECC, Aberdeen, Scotland, 7-9th October 2013
- viii. M.R. Swash, A. Aggoun, **O. Abdul Fatah**, B. Li, J. C. Jacome, E. Tseklevs, "Holographic 3D Image Rendering for Autostereoscopic Multiview 3D Display", IEEE International Symposium on Broadband Multimedia Systems and Broadcasting, London, UK, June 2013.
- ix. M.R. Swash, A. Aggoun, **O. Abdul Fatah**, B. Li, J. C. Jacome, E. Tseklevs, "Dynamic Hyperlinker for 3D Search and Retrieval", IEEE International Symposium on Broadband Multimedia Systems and Broadcasting, London, UK, June 2013.
- x. M.R. Swash, A. Aggoun, **O. Abdul Fatah**, B. Li, J. C. Jacome, E. Tseklevs, "Omnidirectional Holographic 3D Content Generation using Dual Orthographic Projection", IEEE International Symposium on Broadband Multimedia Systems and Broadcasting, London, UK, June 2013.
- xi. A. Mehanna, A. Aggoun, **O. Abdul Fatah**, M. R. Swash, E. Tseklevs, "Adaptive 3D-DCT based compression algorithms for Integral Images", IEEE International Symposium on Broadband Multimedia Systems and Broadcasting, London, UK, June 2013

- xii. E. Alazawi, A. Aggoun, **O. Abdul Fatah**, M.R. Swash, “Adaptive Depth Map Estimation from 3D Integral Images”, IEEE International Symposium on Broadband Multimedia Systems and Broadcasting, London, UK, June 2013. “Awarded – Best student paper”

## **1.5 The Thesis Outline**

**Chapter 1:** Introduces the research subject and provides an overview of holographic 3D imaging. It also presents the research contributions.

**Chapter 2:** Presents exploitations of 3D imaging and display technologies, which also include digital refocusing.

**Chapter 3:** Presents an improved method of refocusing and all-in-focused based on viewpoint orthographic images.

**Chapter 4:** Presents a high resolution refocusing based on sub-images with sub-pixel adjustment using upsampling with shifting and integration technique.

**Chapter 5:** Presents a smart filter, which reduces noises and artifacts in the resulting final refocused image without affecting the image details.

**Chapter 6:** Presents the integration of different 3D display technologies with holographic 3D contents. In addition, object extraction based on disparity map generated from holographic 3D image data is presented.

**Chapter 7:** Concludes the research undertaken and the accomplishments made within this thesis. Further potential development of the research is also discussed.

## **1.6 References**

- [1] Wu, C., Aggoun, A., McCormick, M. and Kung, S. (2005). Depth measurement from integral images through viewpoint image extraction and a modified multibaseline disparity analysis algorithm. *Journal of Electronic Imaging*, vol. 14, no. 2, pp. 023018-023018-9
- [2] Aggoun, A., Tseklevs, E., Zarpalas, D., Daras, P., Dimou, A., Soares, L. and Nunes, P. (2013). Immersive 3D holographic video system. *MultiMedia, IEEE*, vol. 20, no. 1, pp. 28-37.
- [3] Fatah, O. A., Aggoun, A., Nawaz, M., Cosmas, J., Tseklevs, E., Swash, M. R., & Alazawi, E. (2012). Depth mapping of integral images using a hybrid disparity analysis algorithm. *Broadband Multimedia Systems and Broadcasting (BMSB) IEEE*, pp. 1.
- [4] Lumsdaine, A., & Georgiev, T. (2008). Full resolution light field rendering. technical report, Indiana University and Adobe Systems, Tech. Rep.
- [5] Aggoun, A. (2010). 3D holographic imaging technology for real-time volume processing and display. *High-Quality Visual Experience Springer*, pp. 411-428.
- [6] De Abreu Pereira, N., Duffy, A., Pidancet, O., Biperis, I., Aggoun, A., Tseklevs, E., ... & Weitnauer, M. User Acceptance Validation Plan, pp. 16-19.

## **2. Chapter Two**

### **Exploitation of 3D Imaging Technology**

This chapter presents an in-depth literature review of 3D imaging systems for the past decade. State-of-the-art 3D imaging systems are discussed with 3D image processing techniques which have been proposed in this research to deal with the simplest form of presenting the true 3D including their major drawbacks.

#### **2.1 3D Imaging Technologies**

The concept of 3D has been around since early 11th century by Arabian mathematician and philosopher Al-hazen, who reported in his published optic book [1][2]. However, at that time the use was not appropriate to the world of arts until 400 years later, when the Italian architect Fillippo Brunelleschi



discovered and formulated a set of drawings to prove the principle of perspective. In 1692 the French painter Bois-Clair noticed in real life, where each eye view the scene from slightly different perspective due to the distance between the eye pupils [3][4]. Therefore, he carefully combined two different perspective paintings, interlaced in thin vertical stripes. This is to allow left and right eye perceive with two different perspectives of the scene simultaneously by carefully positioning of the vertical stripes. Later, Sir Charles Wheatstone was the first to formulate the binocular vision and demonstrate the stereoscope technology to the Royal Society in 1838 [5]. Since then, more advanced methods of 3D imaging have manifested themselves such as integral imaging (holographic 3D imaging) and holography. Recently, the two have gained greater attention from the scientific communities and industries. Despite many efforts made, work still remains in perfecting the technology as a package to fully launch it on to home users. The need for 3D technology has made significant benefits to the numerous application areas, such as scientific visualization, 3D analysis, medical imaging, telepresence, gaming as well as photography, movies and television. The purpose of all these applications is to offer rich and immersive user experience. As there are several types of 3D imaging systems i.e. capturing, processing and displaying it will be discussed in more depth in section 2.2.

## **2.2 3D Imaging Technologies**

Over the past years up until now, successful 3D imaging technologies have been adopted to bring new sensation to the viewers by fulfilling real-world depth cues as possible. This is still an ongoing research that gains greater interest in opening other application possibilities to bring 3DTV, e.g. computer games, virtual reality, immersive environment and depth measurement [6]. They are mainly categorised as stereoscopic and autostereoscopic which are described in more detail in section 2.3. Stereoscopic 3D technology has been used in many entertainments and commercial industries and it has offered the simplest method for representing the 3D depths with 2D display system for many years, where different means exist in demonstrating the stereoscopic technology [15].

## **2.3 Stereoscopic 3D**

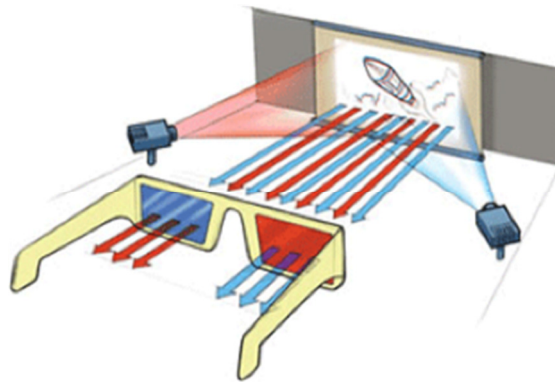
Most researchers had concentrated on stereoscopic method for years; but up until now it has had limited acceptance as it required the viewer to wear the special headgear equipment or stereoscopic glasses for the perception of depth. The stereoscopic glasses are used to channel a different perceptive image from each eye. There are different methods in separating the two perceptive images to each eye [7][8][9][10].

- *Anaglyph*
- *Polarisation*
- *Time division.*

### **2.3.1 Anaglyph**

Anaglyph method uses colour coded image projection to separate the views with colour glasses from left eye and right eye. The viewer only observes the opposite colour of the images projected to the viewer glasses on its left and right eye simultaneously as shown in Fig 2.1. That means blue filter views only red and red filter views only blue. The anaglyph glasses separate the left and right views to give the perception of depth to the viewer [11]. This system is the earliest and most recognized, while it is cheap and easy to produce. Nevertheless, this system lacks in preserving the natural colour of the image and various degrees of cross-talk or ghosting effect. That is when part of the left view leaks over into the right in the process, which leads to producing eyestrain. Another drawback of the system is retinal rivalry, which is when the brightness of left and right views is not the same, making the 3D unpleasant to watch [70].

Some alternative methods have been proposed to solve the anaglyph's challenges, one of which was the polarisation 3D that find its way to commercial success. This method overcomes the colour problems that anaglyph faced [16].



**Fig 2.1 : Anaglyph [65].**

### **2.3.2 Polarisation**

The most modern version of stereoscopic 3D imaging system in the entertainment industry is the perpendicular polarisation technique, where it projects the two separate views simultaneously and then polarised glasses are used to filter out the views to perceive the correct views [12]. Both linear and circular polarisation work in the same principle, however the only benefit of circular polarisation over linear is to allow the viewer to tilt their head without disturbing the effect of 3D perception [13]. One of the advantages of polarised stereoscopic 3D systems over anaglyph [13] is that it offers full-colour image, which is more comfortable to watch. But its disadvantage is that it simultaneously displays both left and right views at the same time, avoiding content delivery at its full resolution. This is due to confusing both left and right views along horizontal direction, which reduces the resolution by half [70]. An alternative method such as Active Shutter was introduced to address the resolution problem in 3D polarisation.

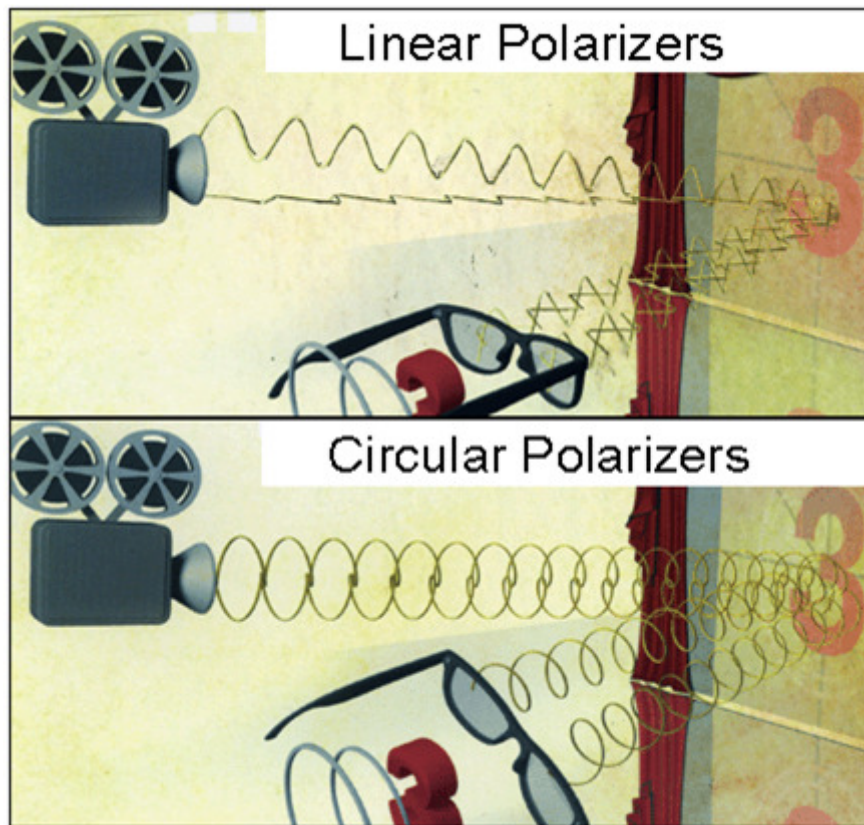


Fig 2. 2 : Polarisation [66].

### 2.3.3 Time division

Time division technique, also known as ‘active shutter’ in commercial industry, works in such a way that left and right views are displayed in sequence one after the other in a very high frame rate [14]. The active shutter glasses are required to synchronise with the display, in such a way that one of the eyes sees nothing while the other eye sees the correct image and few microseconds later the situation is reversed as shown in Fig 2. 3.

It is important to note that each method possesses its drawbacks. For example anaglyph system lacks in preserving the natural colour of the image and various degrees of cross-talk. That is when part of the left view leaks over into the right in the process, which leads to producing eyestrain. Polarisation and time division methods suffer from similar cross-talk that results in visual eye strain.

In addition, all stereoscopic systems fail to provide motion parallax, as only two perceptive views of the scene are captured [70]. The glasses are also required in perceiving the 3D depth, which makes them uncomfortable to wear for prolonged period of time [16].



Fig 2. 3 : Time Division [67].

## 2.4 Autostereoscopic 3D

Autostereoscopic 3D display systems pursue natural viewing as they do not require any special headgear to observe the 3D depth, making this approach more comfortable and practical from viewers' point of view [17]. There are various autostereoscopic 3D imaging principles, namely Holography [18][6], Volumetric display [19], multiview [20] and integral imaging, also known as holographic 3D imaging [21].

### 2.4.1 Holography

Holography is the most well-known technique for creating complete 3D parallax in all directions using light wave interference pattern recorded on photographic film, which later can replay in 3D image with the correct light. The first holography principle was recognised in 1948 and 1949 by Denis Gabor [22]. Fig 2.4 shows the simplest form of creating holographic images using the original 3D model. One way of recording hologram image is to split the laser beam into two. One beam aims directly to the film while passing through the

lens and spreading the beam on the film plate and the other reflects off the object onto a film, causing a beam to reflect light off the object with slight delay on the film. The two beams interference pattern recorded on the film creates holographic image when it is replayed. A model appears on its original position during recording in playback and virtually identical to the real model. The viewer can move around the foreground of the model and see its behind.

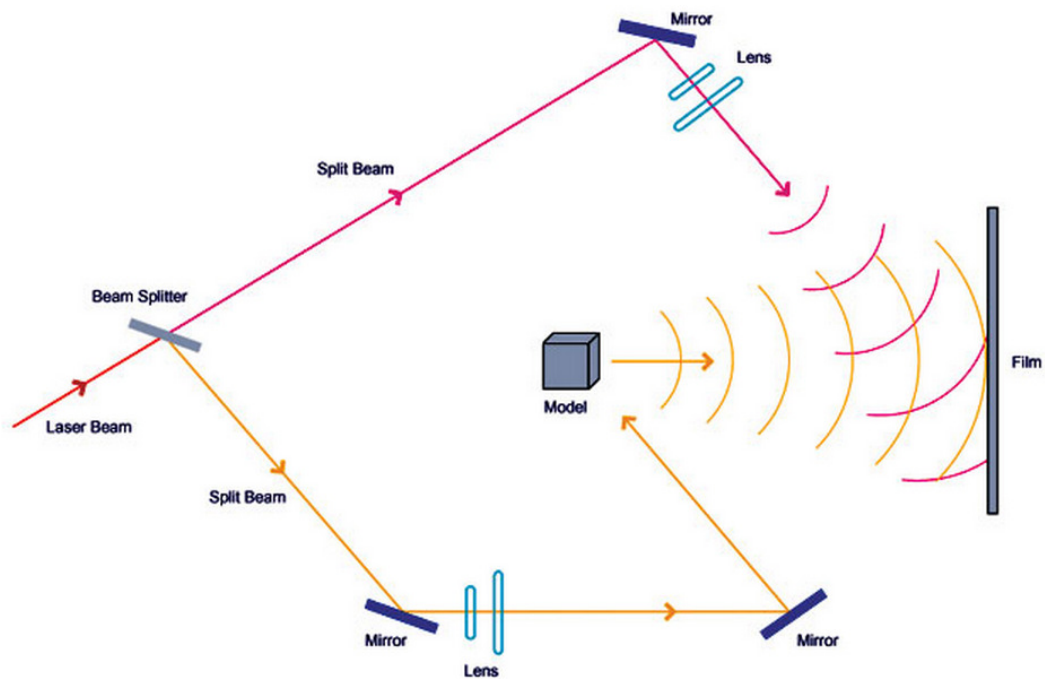


Fig 2.4 : Holography [71].

Holography has a unique characteristic in identifying both the phase and the amplitude of the light waves from a 3D object during recording. It does this by recording the reflected beam. The reflected beam provides the amplitude and the time delay while the reference beam provides the phase. Many variations of holography have been proposed and demonstrated to show the limitation and capability of the technology [8][23][24][25]. One of the best known developments of holography can be found on credit cards and merchandise, where they are used as a security feature [26]. However, the practicality of this technology reduces due to its requirement in making the hologram, as it

requires a coherent light source, dark room conditions and high mechanical stability. Considering the requirement in producing 3D motion picture, the use of Holography is impractical at this stage.

Nevertheless, with recent development on the area of holographic display, a company called “Holografika” launched the HoloVizio display not purely relying on the holographic system itself, but rather based on the holographic geometrical principles with special focus on reconstructing the key elements of spatial vision [36]. This Display is driven with 12 PCs whereas the newer version only requires 4 highly modified PCs. However, the quality of the real image being displayed is poor due to the display rendering process that causes colour deformation of the final image. Also this display is mainly used in scientific lab at this stage for research purposes.

#### **2.4.2 Volumetric displays**

Volumetric displays generate 3D images through emission, scattering or relaying of illumination from well-defined region of (x, y, z) space, where they are mostly seen as hovering inside a rotating projection screen. The first volumetric display was proposed in 1912 and until now the technology is still under development for it to reach the general population [27][28]. Different means exist that can be utilized in presenting the volume display [29], one of which is placing multiple 2D display on top of each other and displaying slices of the scene on them to give the 3D effect [30] as shown in Fig 2. 5a. Another method is the swept-screen volumetric display system in which the 3D image is generated on circular screen by emission of rapid movements. Therefore, circular display offers large field view that is viewable in 360 degrees around the display simultaneously by almost unlimited number of viewers as shown in Fig 2. 5b. Despite many advantages, the volumetric technique is difficult to design given its complexities that limit its use in many applications of 3D display area.

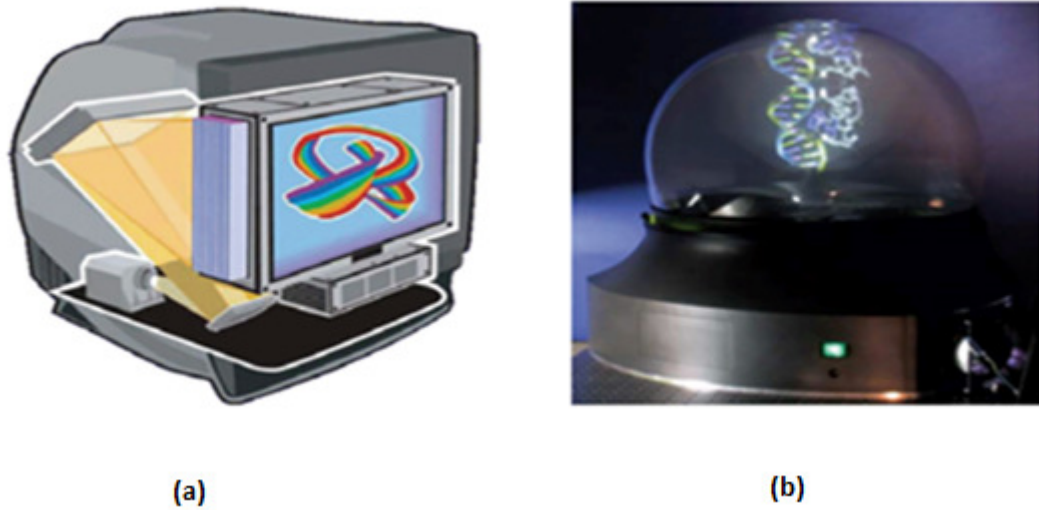


Fig 2. 5 : Volumetric [68].

### 2.4.3 Multi-view 3D displays

The first attempt was made by Auguste Berthier [31] to build multiview 3D display “Autostereoscopic” using parallax barrier technology and later Frederic E. Ives developed the functional mechanism to prove the concept of parallax barrier in 1901. The parallax barrier is placed in front of the display that separates left and right images. This allows channelling both images correctly to the viewer’s left and right eyes. This is achieved by strips of stereo pair images placed in grid screen so that the left strip images can only be seen by left eye and vice-versa as shown in Fig 2. 6. The 3D effect can be perceived if the viewer is within minimum and maximum viewing range that is defined for the display. This was seen as a disadvantage and therefore, complex modification was done to accurately project the appropriate images to the eyes by knowing the position of the viewer’s head. A head tracking system was used to make this possible [32] but it had its own limitations for multi-user situation. The head-tracking 3D display does not support more than one viewer at one time as the display can only be adjusted to one viewer’s eyes.



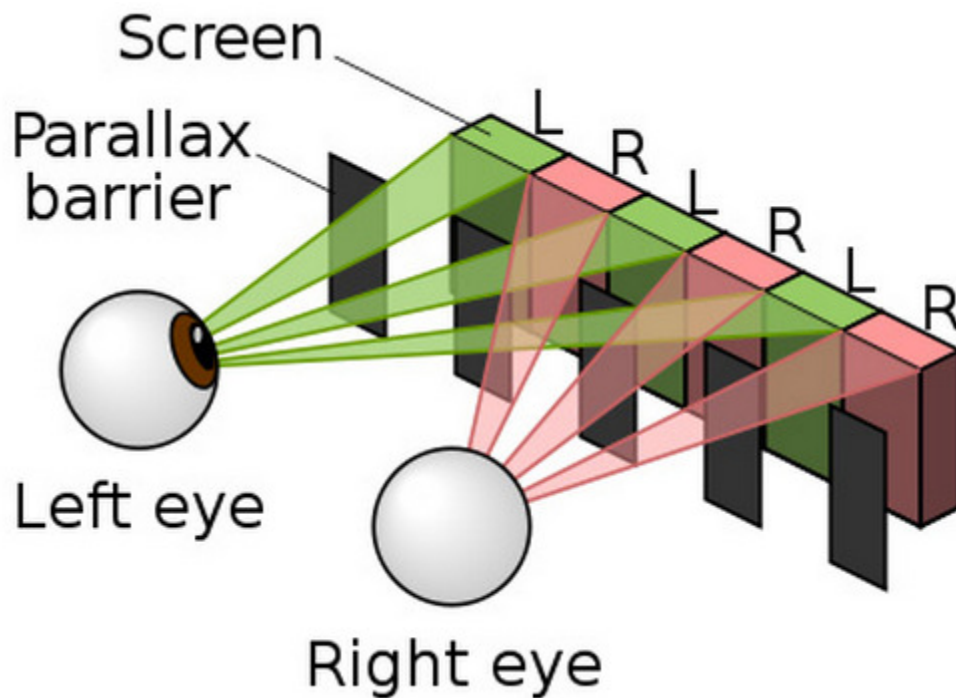


Fig 2. 6 : Parallax barrier [69].

Extensive research continued in improving the parallax barrier's method by introducing the use of lenticular screen to separate the left eye and right eye images to the correct eyes. [28] The simplest form of screen that works on the parallax barrier is the one where lenticular sheet is accurately placed in front of the screen and the stereo pair of view strips is precisely located behind the lenticular sheet as shown in Fig 2. 7. The lenticular sheet works based on the refraction of light whereas the parallax barriers is based on occlusion of light; therefore, the brightness is reduced by half to the viewer perception. Two advantages of lenticular screen over the parallax barrier are the achievement of brighter image quality and lower manufacturing cost. In addition, it allows the observer to position itself to the ideally spot for the 3D effect. The idea of "look-around" capability emerged from this concept by increasing the number of views to achieve multiple viewing zones [33][34]. This technique is widely known as "multi-view" that gives the ability to the viewers to move their head side to side to experience various direction of the scene in 3D.

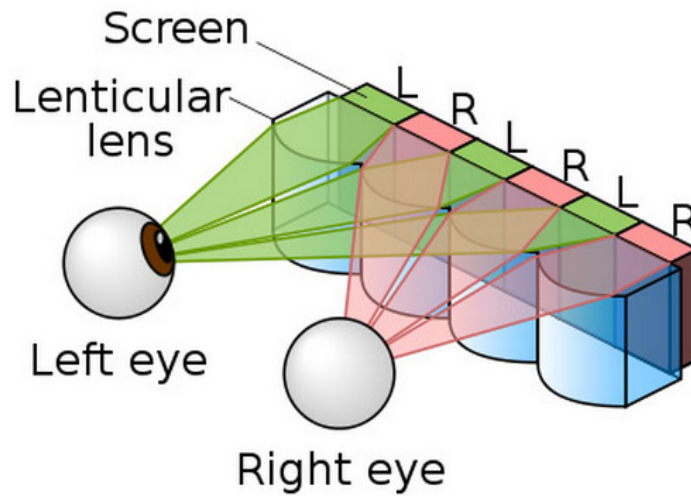


Fig 2. 7 : Lenticular sheet [69].

Multi-view is considered to be the technique with the greatest prospects [35]. This is because Multi-view technique allows capturing based on multiple 2D cameras, hence the migration from 2D techniques is easier in comparison with holographic and volumetric techniques. In recent years, numerous 3D display systems have concentrated on using multiple cameras for recording the scene from different perspectives. This is to allow various degrees of parallax when played back on multi-view displays. The degree of 3D parallax depends on the number of viewpoints. The drawback of such technology lies in its real time capturing that requires complicated multiple camera configurations [38], which is not feasible and becomes an expensive process. It is also essential to introduce clever techniques to improve time-consuming algorithms to render out viewpoints. Multiview 3D imaging mimics human eye technique, thus it relies upon the brain to fuse two disparate images to create the 3D perception. This can cause eyestrain and headache, because the viewers are focusing at the screen plane while simultaneously converging their eyes to a location in space, making an unnatural viewing experience [37].

As a result, it is a great motivation for researchers to continue pursuing an alternative solution to compensate for the complexity of 3D systems such as expensive multiple cameras configuration and multi-step post-production

processes. The result is a promising true 3D imaging system, called Integral Imaging, which uses holography characteristics for reconstructing true 3D scene in space and at the same time, it is applicable to day light environment [21].

#### 2.4.4 Integral Imaging (II)

Integral Imaging also known as holoscopic 3D imaging offers the simplest technique that is capable of recording and replaying the true spatial optical model of the 3D scene in form of a planar intensity distribution by using the optical components [6]. Despite the fact that Integral Imaging is the closest technology to the holographic technique, it records the 3D information in 2D form and displays it in full 3D with optical component, without the need of coherent light source and confine dark fine, making this more practical approach to live capture and display [37].

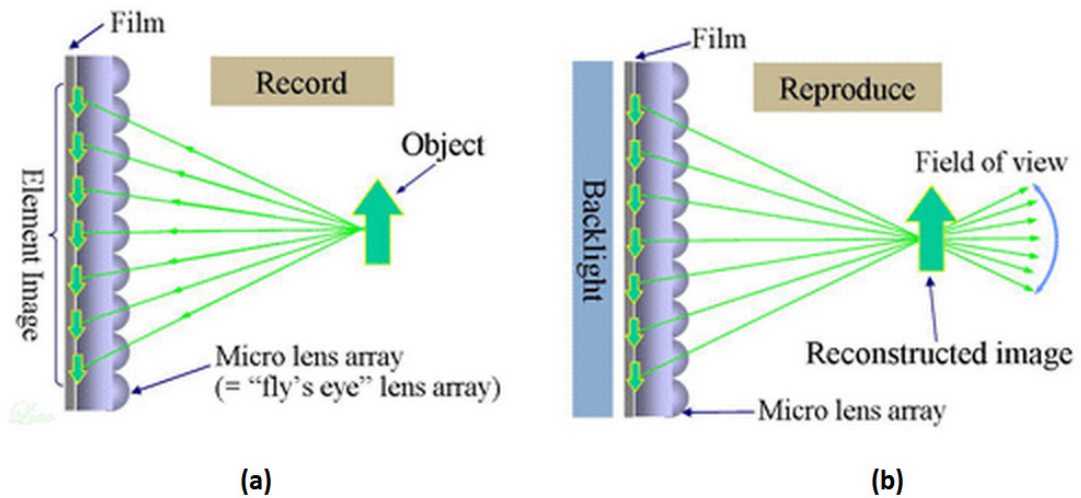


Fig 2. 8 : Principle of Integral Photography [72].

The person who pioneered the integral photography was G. Lippmann in 1908. It was another quest in portraying the 3D scene as accurately as possible [39]. He used a set of small microlens arrays closely packed together and placed them between the object and photographic film, where each microlens views the scene at slightly different angle to its neighbouring one. Single recording captures a number of small perspective 2D images, which is called elemental

images [7][40] as shown in Fig 2. 8a. Once the scene is recorded on the film, a full natural colour with continuous parallax can be replayed by placing an appropriate microlens on the image surface as shown in Fig 2. 8b.

In the traditional setup, the reconstruction of the 3D scene is replayed with pseudoscopic image (spatially inverted). Many researchers have proposed numerous methods of correcting the depth. One optical method was proposed by H. E. Ive in 1931, to address pseudoscopic problem by introducing another optical subsystem during recording, which can be replayed with correct depth as shown in Fig 2. 9. A second-stage recording enabled each microlens image to rotate 180 degrees with respect to its optical axis. This optical approach possessed its own drawbacks by creating a significant amount of noise associated with second-stage recording process [41].

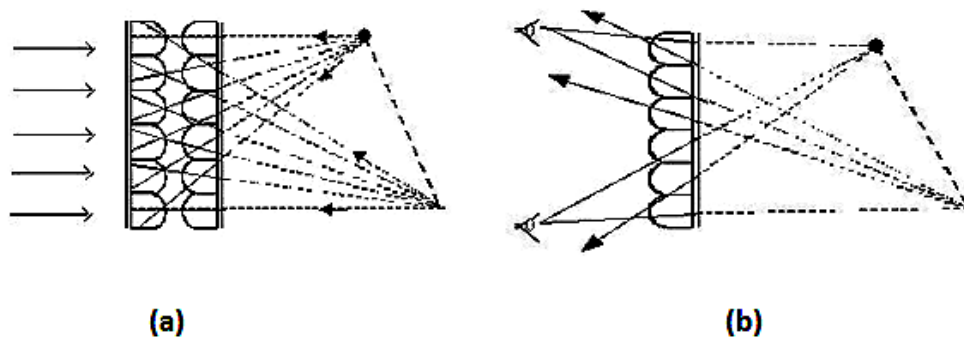


Fig 2. 9 : Principle of two recording [28] (a) a second stage recording of integral photograph. (b) Replay and viewing of orthoscopic image scene.

Therefore, a two-tier network was proposed by Davies and McCormick in De Montfort University (DMU) to address image degradation. The two-tier network worked as an optical “transmission inversion screen” allowing a direct capturing of correct 3D image spatial for orthoscopic replay. The modification of optical element, as shown in Fig 2. 10, illustrates the two-tier network consisting of two pairs of microlens array placed back to back to generate spatial inversion [37].

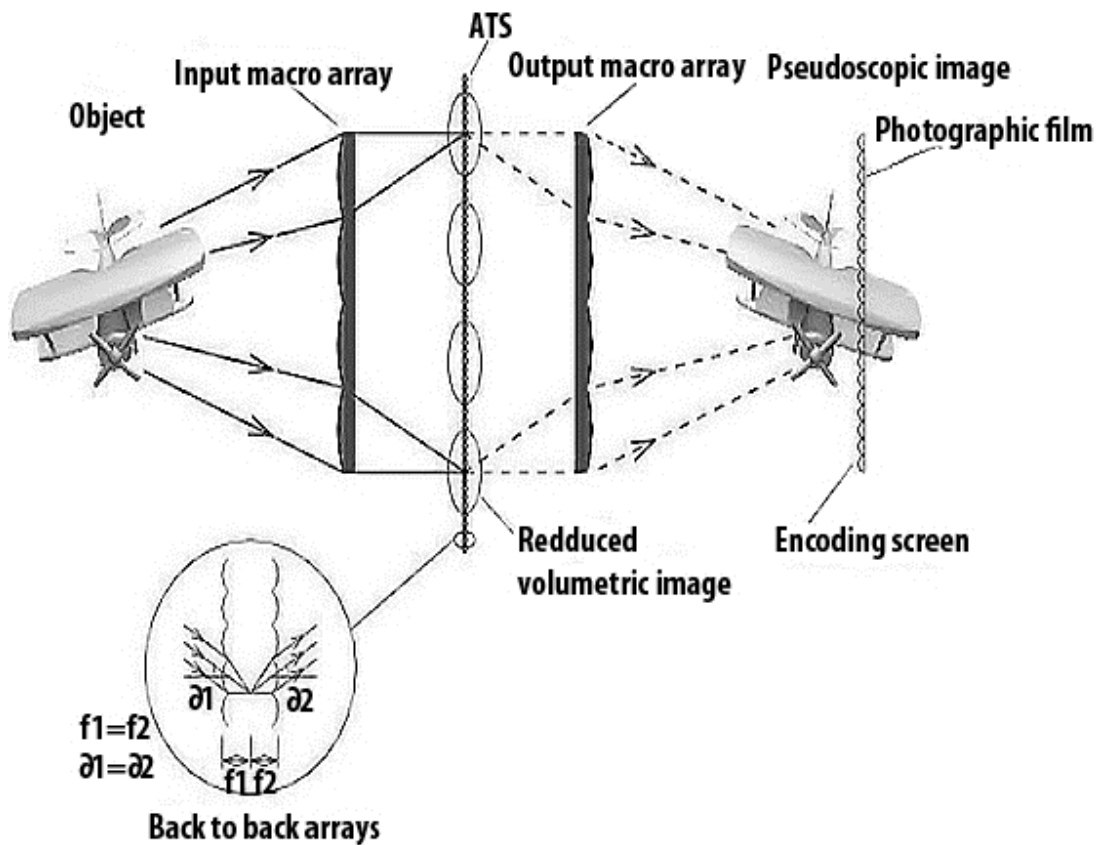


Fig 2. 10 : The advance integral imaging system from [67].

This optical arrangement transfers volumetric image signal in space to such a form that each object points in real space are recorded to the same object position in the image sensor. This allows image signal to transfer with no inversion in the recording as well as preserving the same spatial co-ordinates. There are two types of microlens arrays e.g. Omnidirectional and Unidirectional, which are capable of recording and replaying the 3D images, as shown in Fig 2. 10. The omnidirectional hologoscopic 3D imaging requires square based spherical microlens structure in recording, which offers parallax from all direction. The unidirectional hologoscopic 3D imaging uses 1D cylindrical microlens arrays in capturing, thus contains only horizontal direction parallax [42]. This makes the two-tier network approach capable of capturing the true 3D on the image sensor. However, this form of recording requires an ultra-high resolution imaging sensor together with special optical element, which makes it

impractical in real world capturing. Also, the reconstruction of the image will appear distorted due to the non-constant lateral magnification of the converging lens [37]

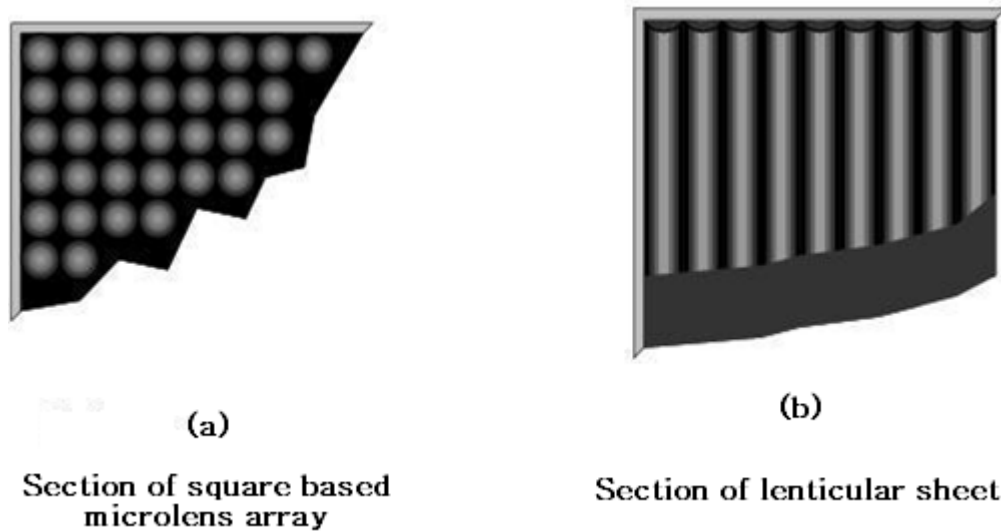


Fig 2. 11 : Diagrammatic representation of the lens array from [42].

Few digital methods were proposed by different researchers to accommodate the pseudoscopic to orthoscopic conversion. The work by Okano *et al.* [61][62] inverts each micro-image during the capturing process, which reduces the parallax angle to an acceptable point in the scene. Another digital contribution was made by Martinez-Corral *et al.* [63], where they proposed a way of fixing the aliasing problem in the pixel mapping. It is vital to have the number of pixels per lenslet as a multiple of the number of lenslets, which makes the number of pixels per micro-image very large (order of 100s) and rendering is impractical for many 3D display applications. [37]

In recent years the focus has been on improving many data processing issues that required special solution to holographic 3D imaging. Additionally, there exists the challenge created by the constrained resolution of Charge-Coupled Device (CCD) and Liquid Crystal Display (LCD) that limited scene depth and narrow viewing angle [36]. Based on these problems, comprehensive studies were undertaken to revolutionise the optimal cylinder and intense theory to

resolve them. Many techniques were proposed to mitigate such obstacles; such as time-multiplex [74], spatial-multiplex [37], and super-resolution [75] to improve the resolution of view point images. Further enhancement has been achieved in the area of scene depth and viewing angle to optimise a wider viewing angle with appropriate size to accommodate high resolution holographic 3D image; examples are Multi-layered display device [76], Holographic Optical Element (HOE) lens array [36], curved screen [77], and lens switching [17].

Later in 2005, new handheld 3D camera design, known as the plenoptic camera, capable of recording holographic 3D image was proposed by Ng as well as Fife and Lumsdaine[45]. Ng was the first to insert a microlens array into a conventional handheld camera by simplifying the form of 7D directional information to 4D. This represents both 2D position and 2D directional information. This idea, which had been initially introduced by Adelson and Wang, later was improved by Ng making it more practical [43]. It was mainly used for digital refocusing on photography after capturing, as opposed to the holographic 3D imaging technique that was capable of live capturing and displaying the true 3D.

## 2.5 Plenoptic Camera

It is worthwhile to point out the difference between traditional plenoptic 1.0 [45] and focused plenoptic 2.0 [46] cameras before continuing with the camera optical components. The traditional plenoptic camera renders the image in a significantly low resolution due to its design as shown in Fig 2. 12. [47] The focal length 'L' of the main lens to the microlens in comparison to the focal length 'F' of microlens to the sensor is considerably large. Therefore, each microlens receives parallel light ray of all the possible directional rays with low spatial resolution. The integration of all the pixels under each microlens will represent one pixel of the overall final images at particular depth plane.

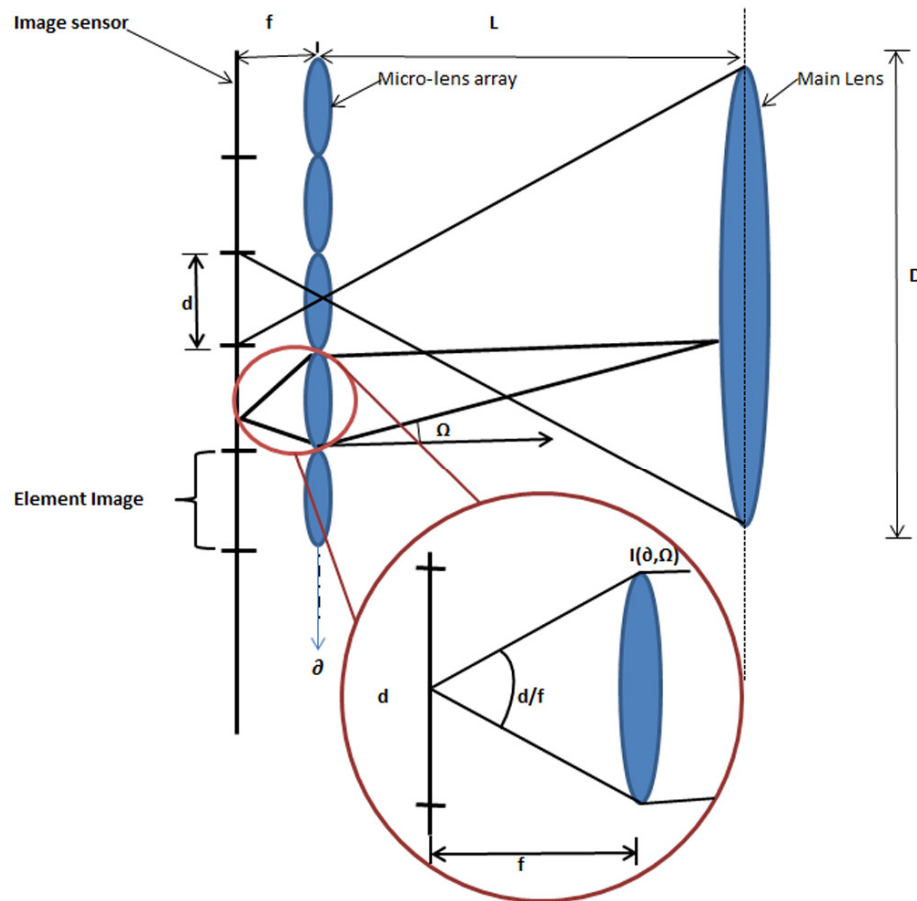
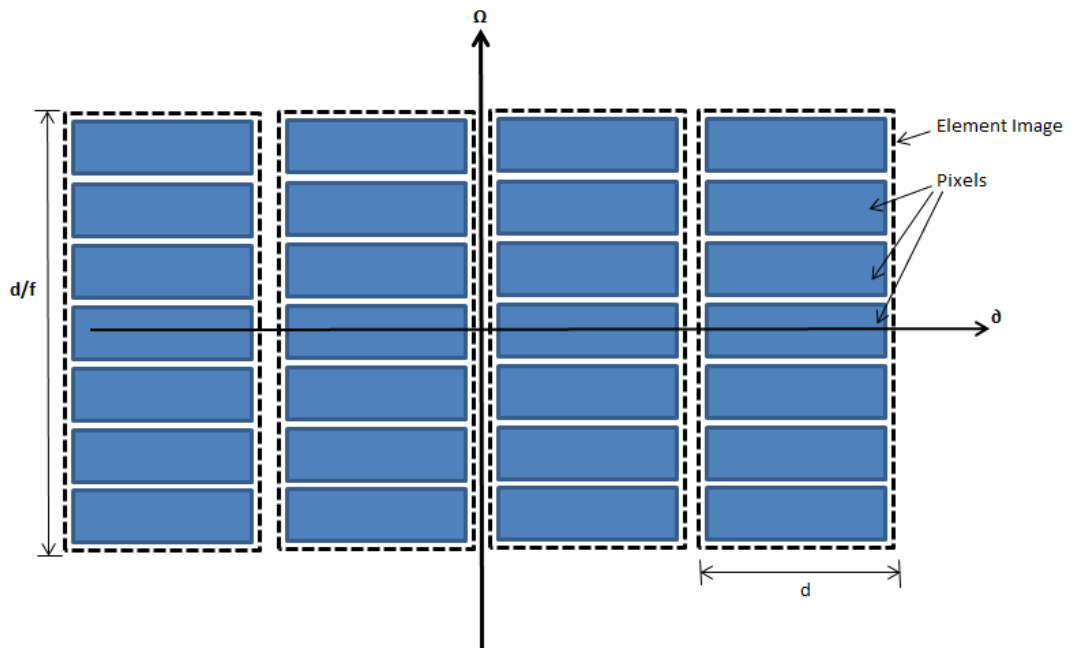


Fig 2. 12 : Traditional Plenoptic camera from [46]

For simplicity purposes, the four-dimensional light rays of Omnidirectional Integral Images  $OII(j,i,n,m)$  are reduced to two-dimensional  $\theta - \Omega$  planes to



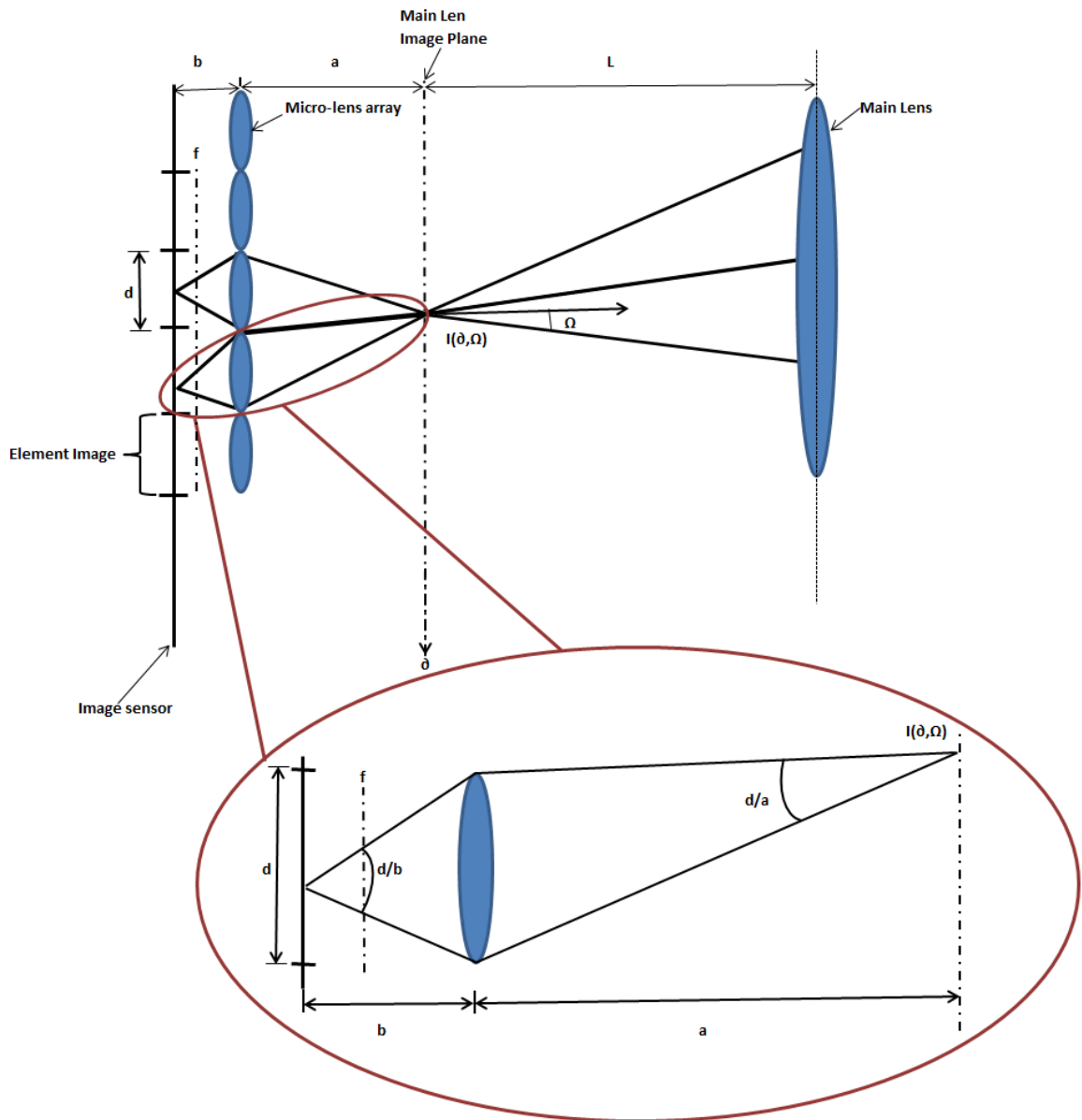
illustrate the behaviour of plenoptic cameras. Fig 2. 13 shows the light field  $I(r(\vartheta, \Omega))$  in plenoptic 1.0, which signifies both position  $\vartheta$  and directional  $\Omega$  of light ray in space with given  $r$  radiance [48]. Therefore, each 3D point in space is sampled by one element image, as a result of which the final rendered image is equal to the number of microlens included in the capturing stages [47].



**Fig 2. 13 : Behavior of traditional plenoptic camera where the number of microlenses limits the spatial resolution in  $\vartheta$  direction however, directional information  $\Omega$  is determined by the number of pixels under each element image**

On the other hand, focused plenoptic (also known as plenoptic 2.0) works in such a way that each microlens is considered as a single camera capturing a small portion of what the main lens sees [48]. It's designed in a way that the microlens focal plane is set to the image plane of the main lens  $L$  leading to higher spatial resolution with lower directional information [47]. As shown in Fig 2. 14, the microlens array moves away from the image sensor at distance  $b$ , where main lens image plane  $L$  forms image in the form of the microlens array at distance  $a$ . Each microlens fulfils the lens equation,  $1/a+1/b=1/f$ , where  $a, b$ , and  $f$  are the distance from the microlens to the main lens image plane, the

distance from the microlens to the sensor, and the focal length of the microlens, respectively.



**Fig 2. 14 : Focused Plenoptic 2.0 microlens imaging in Keplerian mode. The main lens image plane is in front of the microlenses**

Therefore, each microlens acts as a relay imaging system of the main lens image plane  $L$  [48]. This outputs an image, where objects that are placed in different distance in real world, will have different scaling under each microlens to the image sensor. In other words, objects located at various distances from each other are captured by the main camera lens as they appear but the microlens

array sensor is what gives them different scaling relative to their distance and angle.

Hence, focused plenoptic offers higher spatial resolution compared with traditional plenoptic. Fig 2. 15 shows an imitation of focused plenoptic's behaviour to illustrate the relation between spatial resolution and directional resolution. The spatial resolution is directly related to the directional resolution, where the high spatial resolution accommodates areas of directional information; and total resolution of the end image is equal to  $b/a$  times the camera sensor size. Therefore, a trade-off exists between the directional resolutions and spatial resolution focusing on plenoptic design. This camera mode setup, known as Keplerian, places the main lens image plane in front of the microlenses to capture image plane at distance  $L$  [48].

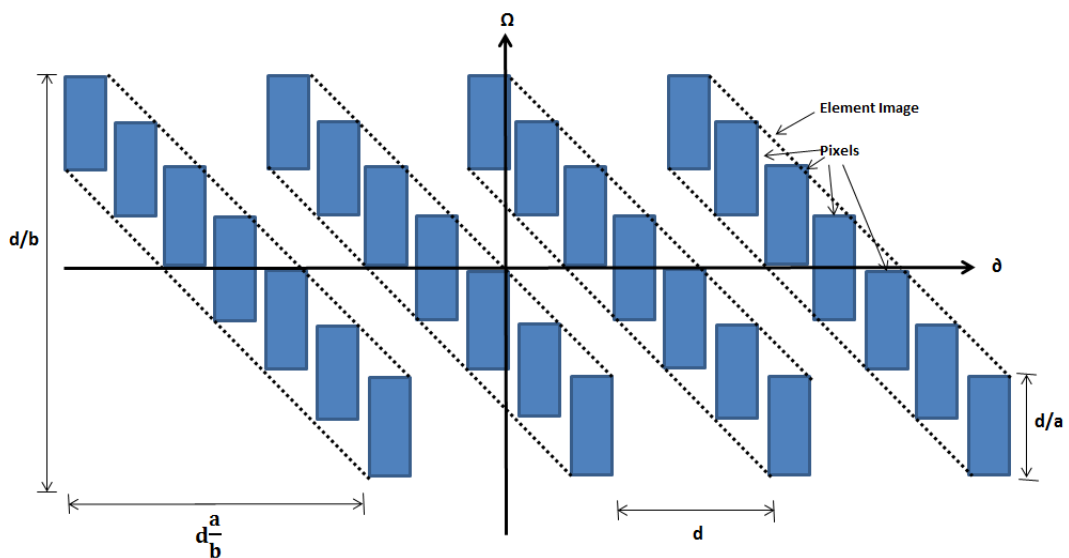


Fig 2. 15 : Behavior of Focused Plenoptic camera

In holographic 3D imaging, focused plenoptic camera design is used in capturing true 3D video images, where it is aimed to achieve not only for digital refocusing after capturing but for whole different purposes. The focused plenoptic camera was designed mainly for the purpose of digital refocusing on photography; whereas holographic 3D takes one step further, by offering realistic 3D viewing experience in a cost effective manner. Therefore, with its

unique optical element no camera calibration is necessary in capturing the 3D information as opposed to 3D stereoscopic production. Also in holographic 3D compactness capturing a true 3D information gains a lot of attraction for which novel depth extraction is proposed to accommodate the conventional stereo matching techniques [38].

## **2.6 Digital Refocusing**

In recent years, researchers have been pursuing a new way to create images through the process of reconstruction for which, the knowledge of digital refocusing after capturing has come to the attention. The first digital refocusing image was generated in 1995 from two views each focused at different depth [49]. This method is called “depth from defocus” in computer vision, where the depth of an object is estimated based on the blur matrix from the two views focused at different depth. Unfortunately, this process returns high artifacts in the final image when changing the virtual plane closer to image plane [49][50][51]. Therefore, the method was rather unsuccessful until some other systems like depth estimation for videos on real-time emerged depth from defocusing algorithm [52][53].

The first demonstration of digital refocusing on light field (also called as holographic 3D imaging) was reported in 2000 by Isaksen, McMillan and Gortler, after which other similar attempts were carried out with different title names [54][55][56][57]. However, these methods faced two drawbacks. First, capturing light field data required moving camera with lengthy scanning or an array of cameras packed together, which is impractical for normal shooting as it's carried with regular handheld cameras. Second, the final refocused images were observed with high aliasing on defocused regions, due to a mismatch in rendering process [45]. Both drawbacks were addressed by Ng[45], as he was the first to introduce a more practical way of capturing light field data with conventional hand-held camera. In addition, aliasing on the final images was resolved with a new optical design in which microlens array was introduced inside the conventional camera with microlens closely packed together in order for all rays to pass through single aperture lens to the image sensor [45].

In 2005, Ng introduced a new plenoptic camera with a new processing technique followed by the work by Fife [58] and Lumsdaine [46][47]. Plenoptic cameras were mainly used for refocusing in photography and the

images rendered by Ng were in low resolution [47]. Since Ng used the angular ray information that refers to the viewpoint image in refocusing, the spatial resolution is defined by the number of microlenses along y and x directions [59]. Next, a full resolution method [47] was introduced to compensate for the poor resolution in Ng's method. The final images of the full resolution method were considerably better compared to the Ng's approach. This facilitated a completely new configuration to the traditional plenoptic camera, where the focus of main camera lens is well in front of the microlenses and focuses the microlenses on the image that is formed inside the camera [48] so that each microlens captures a focused perspective image from a specific position. This would have had a full resolution rendering, which has been applied to acquire a higher resolution image. The full resolution method works by selecting pitch size under each element image to create focused image, but this technique returns artifacts making it unnatural on the final image. Therefore, introducing the depth information to the full resolution method serves the purpose of sustaining a natural looking photographic image. This will remedy the blocky noise artifact problems in the final refocused images. Unfortunately this process is time consuming, as it requires matching the position of each individual element image from its four neighbouring element images.

Later, some research work in the field was presented in [60] to minimise artifacts in the final image by new optical camera design to minimise magnification of scene depth where objects at different depth almost have constant magnification throughout. However, this design reduces the resolution on the number of refocusing planes. In 2012, new blending algorithm was introduced to overcome the arising artifacts down to their minimum, which was caused by patch processing [48]. Yet, this full resolution with blending observed with the aliasing effect of the final rendered images.

The theme of this research is to find alternative way to solve the resolution problem and improving the visual quality by using the orthographic projections (viewpoint) as well as sub-images approach. For more clarity, we will apply the

same algorithm on two different types of light field (holographic 3D) images namely omnidirectional integral image (*OII*) and unidirectional integral image (*UII*) to check the strength of the aforesaid technique.

## **2.7 Conclusion**

This chapter explored state-of-the-art 3D imaging technologies, among which the simplest form of representing a true 3D image is Integral Imaging. Integral Imaging offers true 3D capturing and replay that opens up a new direction and possibilities for digital imaging such as changing the focus plane after the capture as well as more advanced techniques for measuring the scene depth. Furthermore, Integral Image is a promising technique to deliver rich viewing sensation to the eyes without glasses and fatigue effects. Also digital refocusing techniques were touched on here while emphasising their major drawbacks and ways to avoid them.

## **2.8 Reference**

- [1] Al-Haytham, I., Alhazen, Al, S., & Abdelhamid, I. (1989). *The Optics of Ibn Al-Haytham: Books I-III: on Direct Vision*. Warburg Institute, University of London.
- [2] Alhazen, I. (1989). *Book of optics. The optics of Ibn-Haytham*, pp. 149-170.
- [3] Zilly, F., Kluger, J. and Kauff, P. (2011). Production rules for stereo acquisition. *Proceedings of the IEEE*, vol. 99, no. 4, pp. 590-606.
- [4] Roberts, D. E. (2003). *History of lenticular and related autostereoscopic methods*. Leap Technologies. Hillsboro, pp. 1-16.
- [5] Wheatstone, C. (1838). Contributions to the physiology of vision.--Part the first. On some remarkable, and hitherto unobserved, phenomena of binocular vision. *Philosophical transactions of the Royal Society of London*, vol. 128, pp. 371-394.
- [6] Zarpalas, D., Fotiadou, E., Biperis, I. and Daras, P. (2012). Anchoring Graph Cuts Towards Accurate Depth Estimation in Integral Images. *Journal of Display Technology*, vol. 8, no. 7, pp. 405-417.
- [7] Okoshi, T., & Oshima, K. (1976). Three-dimensional imaging from a unidirectional hologram: wide-viewing-zone projection type. *Applied optics*, vol. 15, no. 4, pp. 1023-1029.
- [8] Benton, S. A. (1975). Holographic displays- A review. *Optical Engineering*, vol. 14, no. 5, pp. 402-407.
- [9] Kratomi, S. (1973). Stereoscopic apparatus having liquid crystal filter viewer. U.S. Patent No. 3,737,567. Washington, DC: U.S. Patent and Trademark Office.
- [10] McAllister, D.F. (1994). *Stereo computer graphics and other true 3D technologies*. Princeton University Press, pp. 247 - 262. <http://www.bcin.ca/Interface/openbcin.cgi?submit=submit&Chinkey=203191>



- [11] Dubois, E. (2001). A projection method to generate anaglyph stereo images. *Acoustics, Speech, and Signal Processing, 2001. Proceedings. (ICASSP'01). 2001 IEEE International Conference on IEEE*, pp. 1661.
- [12] Toperverg, B., Nikonov, O., Lauter-Pasyuk, V. and Lauter, H. (2001). Towards 3D polarisation analysis in neutron reflectometry. *Physica B: Condensed Matter*, vol. 297, no. 1, pp. 169-174.
- [13] Boher, P., Leroux, T., Bignon, T. and Collomb-Patton, V. (2010). Multispectral polarisation viewing angle analysis of circular polarised stereoscopic 3D displays. *Proc. SPIE*, pp. 75240R.
- [14] Divelbiss, A., Swift, D., & Tserkovnyuk, W. (2004). 3D stereoscopic shutter glass system. U.S. Patent Application 10/764,277.
- [15] Surman, P. (2013). *Stereoscopic and Autostereoscopic Displays. 3D-TV System with Depth-Image-Based Rendering* Springer, pp. 375-411.
- [16] Lambooij, M., IJsselsteijn, W. and Heynderickx, I. (2011). Visual discomfort of 3D TV: Assessment methods and modeling. *Displays*, vol. 32, no. 4, pp. 209-218.
- [17] Zinger, S., Do, L., & de With, P. H. N. (2010). Free-viewpoint depth image based rendering. *Journal of visual communication and image representation*, vol. 21, no. 5, pp.533-541.
- [18] Tanjung, R.B.A., Xu, X., Liang, X., Solanki, S., Pan, Y., Farbiz, F., Xu, B. and Chong, T. (2010). Digital holographic three-dimensional display of 50-Mpixel holograms using a two-axis scanning mirror device. *Optical Engineering*, vol. 49, no. 2, pp. 025801-025801-9.
- [19] Blundell, B., Schwarz, A. and Horrell, D. (1993). Volumetric three-dimensional display systems: their past, present and future. *Engineering Science and Education Journal*, vol. 2, no. 5, pp. 196-200.
- [20] Zhang, Y., Ji, Q. and Zhang, W. (2010). Multi-view autostereoscopic 3D display. *Optics Photonics and Energy Engineering (OPEE), 2010 International Conference IEEE*, pp. 58.

- [21] Zarpalas, D., Biperis, I., Fotiadou, E., Lyka, E., Daras, P. and Strintzis, M.G. (2011). Depth estimation in integral images by anchoring optimization techniques. *Multimedia and Expo (ICME), 2011 IEEE International Conference on IEEE*, pp. 1.
- [22] Hill, P. (2007). Dennis Gabor-Contributions to Communication Theory & Signal Processing. *The International Conference on Computer as a Tool IEEE*, pp. 2632.
- [23] Saxby, G. (2010) *Practical holography*, CRC Press, pp. 3-15.
- [24] Casper, J. and Feller, S. (1987). *The Complete Hologram Book*, pp 23-30.
- [25] Hariharan, P. (2002). *Basics of holography*, Cambridge University Press, pp. 1- 33.
- [26] Trolinger, J. D., & Weber, D. C. (1999). Holographic labeling and reading machine for authentication and security applications. U.S. Patent No. 5,920,058. Washington, DC: U.S. Patent and Trademark Office.
- [27] Lewis, J.D., Verber, C.M. and McGhee, R.B. (1971). A true three-dimensional display. *Electron Devices, IEEE Transactions on*, vol. 18, no. 9, pp. 724-732.
- [28] Wu, C. (2003). Depth measurement in integral images, pp. 8-22.
- [29] Neil, A. (2005). Autostereoscopic 3D displays. *Computer*, vol. 8, pp. 32-36.
- [30] Olsson, R. (2008). *Synthesis, Coding, and Evaluation of 3D Images Based on Integral Imaging*, pp. 11-18.
- [31] Funk, W. (2012). History of autostereoscopic cinema. In *IS&T/SPIE Electronic Imaging* (pp. 82880R-82880R). International Society for Optics and Photonics.
- [32] Tetsutani, N., Omura, K., & Kishino, F. (1994). Wide-screen autostereoscopic display system employing head-position tracking. *Optical Engineering*, vol. 33, no. 11, pp. 3690-3697.
- [33] Dodgson, N. A. (1997). *Autostereo displays: 3D without glasses*. EID: *Electronic Information Displays*.

- [34] Harman, P. V. (1996). Autostereoscopic display system. In *Electronic Imaging: Science & Technology*, pp. 56-64. International Society for Optics and Photonics.
- [35] Son, J. Y., Javidi, B., & Kwack, K. D. (2006). Methods for displaying three-dimensional images. *Proceedings of the IEEE*, vol. 94, no. 3, pp. 502-523.
- [36] Kovács, P. T., & Balogh, T. (2010). 3D Visual Experience. In *High-Quality Visual Experience*, pp. 391-410, Springer Berlin Heidelberg.
- [37] Aggoun, A. (2010). 3D holoscopic imaging technology for real-time volume processing and display. In *High-Quality Visual Experience*, pp. 411-428, Springer Berlin Heidelberg.
- [38] Fatah, O. A., Aggoun, A., Nawaz, M., Cosmas, J., Tseklevs, E., Swash, M. R., & Alazawi, E. (2012). Depth mapping of integral images using a hybrid disparity analysis algorithm, pp. 1-4.
- [39] Lippmann, G. (1908). Epreuves reversibles donnant la sensation du relief. *J. Phys. Theor. Appl.*, vol. 7, no. 1, pp. 821-825.
- [40] Arimoto, H., & Javidi, B. (2001). Integral three-dimensional imaging with digital reconstruction. *Optics letters*, vol. 26, no. 3, pp. 157-159.
- [41] Jang, J. S., & Javidi, B. (2004). Three-dimensional projection integral imaging using micro-convex-mirror arrays. *Optics express*, vol. 12, no. 6, pp. 1077-1083.
- [42] Forman, M. C., Aggoun, A., & McCormick, M. (1997). A novel coding scheme for full parallax 3D-TV pictures. In *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97, 1997 IEEE International Conference, Vol. 4*, pp. 2945-2947.
- [43] Zhou, Z. L., Yuan, Y., & Xiangli, B. (2009). Computer simulation for digital refocusing imager based on light field photography. *International Symposium of Society for Optics and Photonics*, pp. 73842W-73842W.
- [44] Adelson, E. H., & Wang, J. Y. A. (1992). Single lens stereo with a plenoptic camera. *IEEE transactions on pattern analysis and machine intelligence*, vol. 14, no. 2, pp. 99-106.

- [45] Ng, R., Levoy, M., Brédif, M., Duval, G., Horowitz, M., & Hanrahan, P. (2005). Light field photography with a hand-held plenoptic camera. Computer Science Technical Report CSTR, vol. 2, no. 11.
- [46] Lumsdaine, A., & Georgiev, T. (2009). The focused plenoptic camera. In Computational Photography (ICCP), 2009 IEEE International Conference, pp. 1-8.
- [47] Lumsdaine, A., & Georgiev, T. (2008). Full resolution light field rendering. Indiana University and Adobe Systems, Tech. Rep, pp. 1-12.
- [48] Georgiev, T., & Lumsdaine, A. (2010). Focused plenoptic camera and rendering. Journal of Electronic Imaging, vol. 19, no. 2, pp. 021106-021106.
- [49] Subbarao, M., Wei, T. C., & Surya, G. (1995). Focused image recovery from two defocused images recorded with different camera settings. Image Processing, IEEE Transactions, vol. 4, no. 12, pp. 1613-1628.
- [50] Krotkov, E. (1988). Focusing. International Journal of Computer Vision, vol. 1, no. 3, pp. 223-237.
- [51] Pentland, A. P. (1987). A new sense for depth of field. Pattern Analysis and Machine Intelligence, IEEE Transactions, vol. 9, no. 4, pp. 523-531.
- [52] Nayar, S. K., Watanabe, M., & Noguchi, M. (1996). Real-time focus range sensor. Pattern Analysis and Machine Intelligence, IEEE Transactions, vol. 18, no. 12, pp. 1186-1198.
- [53] Pentland, A., Scherrock, S., Darrell, T., & Girod, B. (1994). Simple range cameras based on focal error. JOSA A, vol. 11, no. 11, pp. 2925-2934.
- [54] Yang, J. C., Everett, M., Buehler, C., & McMillan, L. (2002). A real-time distributed light field camera. In Proceedings of the 13th Eurographics workshop on Rendering, pp. 77-86.
- [55] Levoy, M., Chen, B., Vaish, V., Horowitz, M., McDowall, I., & Bolas, M. (2004). Synthetic aperture confocal imaging. ACM Transactions on Graphics (TOG), vol. 23, no. 3, pp. 825-834.

- [56] Vaish, V., Wilburn, B., Joshi, N., & Levoy, M. (2004). Using plane+ parallax for calibrating dense camera arrays. *Computer Vision and Pattern Recognition. IEEE Computer Society Conference*, vol. 1, pp. 1-2.
- [57] Wilburn, B., Joshi, N., Vaish, V., Talvala, E. V., Antunez, Antunez, E., E., Barth, Adams, A., Horowitz, M., & Levoy, M. (2005). High performance imaging using large camera arrays. *ACM Transactions on Graphics (TOG)*, vol. 24, no. 3, pp. 765-776.
- [58] Fife, K., El Gamal, A., & Wong, H. S. (2008). A Multi-Aperture Image Sensor With 0.7  $\mu\text{m}$  Pixels in 0.11  $\mu\text{m}$  CMOS Technology. *IEEE Journal of Solid-State Circuits*, vol. 43, no. 12, pp. 2990-3005.
- [59] Georgiev, T. G., & Lumsdaine, A. (2009). Resolution in plenoptic cameras. *Computational Optical Sensing and Imaging of Optical Society of America*, pp. CTuB3.
- [60] Georgiev, T., & Lumsdaine, A. (2010). Reducing plenoptic camera artifacts. *Computer Graphics Forum of Blackwell Publishing Ltd*, vol. 29, no. 6, pp. 1955-1968.
- [61] Okano, F., Arai, J., Mitani, K. O. H. J. I., & Okui, M. (2006). Real-time integral imaging based on extremely high resolution video system. *Proceedings of the IEEE*, vol. 94, no. 3, pp. 490-501.
- [62] Arai, J., Okano, F., Hoshino, H., & Yuyama, I. (1998). Gradient-index lens-array method based on real-time integral photography for three-dimensional images. *Applied optics*, vol. 37, no. 11, pp. 2034-2045.
- [63] Martinez-Corral, M., Javidi, B., Martínez-Cuenca, R., & Saavedra, G. (2005). Formation of real, orthoscopic integral images by smart pixel mapping. *Optics Express*, vol.13, no. 23, pp.9175-9180.
- [64] Kishk, S., & Javidi, B. (2003). Improved resolution 3D object sensing and recognition using time multiplexed computational integral imaging. *Optics express*, vol. 11, no. 26, pp. 3528-3541.
- [65] Suhas. (2010). How 3D works? Available: <http://suhastech.com/how-3d-stereoscopic-display-glasses-works/>. Last accessed 10th Jun 2010.

- [66] Computer Desktop Encyclopedia. (2010). 3D glasses. Available: <http://www.answers.com/topic/3d-glasses>. Last accessed 15th Aug 2012.
- [67] Joe Pedoto. (2010). Television Begins Push into the 3rd Dimension - New York Times. Available: <http://www.nytimes.com/imagepages/2010/01/06/business/media/06teleGrfx.html>. Last accessed 20th July 2011.
- [68] Michal Husakm. (1999). Using advanced methods of computer graphics for crystallographic needs. Available: <http://www.iucr.org/resources/commissions/crystallographic-computing/newsletters/1/using-advanced-methods-of-computer-graphics>. Last accessed 20th July 2013.
- [69] Cmglee. (2011). Parallax barrier vs lenticular screen. Available: [http://en.wikipedia.org/wiki/File:Parallax\\_barrier\\_vs\\_lenticular\\_screen.svg](http://en.wikipedia.org/wiki/File:Parallax_barrier_vs_lenticular_screen.svg). Last accessed 25th Aug 2012.
- [70] Olsson, R. (2008) Synthesis, Coding, and Evaluation of 3D Images Based on Integral Imaging,
- [71] National Postal Museum. (2008). Holography: Into the Future. Available: <http://www.postalmuseum.si.edu/stampstakeflight/holography.html>. Last accessed 12th May 2010.
- [72] Hongen Liao. (2001). A novel medical autostereoscopic image: Integral Videography. Available: <http://www.atre.t.u-tokyo.ac.jp/en/content/view/59/86/>. Last accessed 15th May 2012.
- [73] Wu, C., Aggoun, A., Kung, S. Y., & McCormick, M. (2005). Depth measurement from integral images through viewpoint image extraction and a modified multibaseline disparity analysis algorithm. *Journal of Electronic Imaging*, vol. 14, no. 2, pp. 023018-023018-9.
- [74] Jang, J. S., & Javidi, B. (2004). Time-multiplexed integral imaging for 3D sensing and display. *Optics and Photonics News*, vol. 15, no. 4, pp. 36-43.

- [75] Takeda, H., Milanfar, P., Protter, M., & Elad, M. (2009). Super-resolution without explicit subpixel motion estimation. *Image Processing, IEEE Transactions*, vol. 18, no. 9, pp. 1958-1975.
- [76] Fujita, H., Togashi, S., & Tsuzuki, A. (1984). Multi-layer display device with nonactive display element groups. U.S. Patent No. 4,443,062. Washington, DC: U.S. Patent and Trademark Office.
- [77] Kim, Y., Park, J. H., Min, S. W., Jung, S., Choi, H., & Lee, B. (2005). Wide-viewing-angle integral three-dimensional imaging system by curving a screen and a lens array. *Applied optics*, vol. 44, no. 4, pp. 546-552

## **3. Chapter Three**

### **Digital Refocusing based on VI**

This chapter explores digital refocusing techniques in holographic 3D (Integral) imaging that includes state-of-the-art digital refocusing in comparison with the proposed method. The main focus is on an alternative solution to the refocused image resolution as well as to improve the visual quality. The proposed algorithm is applied to both types of holographic 3D images namely omnidirectional and unidirectional to check the robustness of the aforementioned technique. Omnidirectional holographic 3D content contains both horizontal and vertical 3D information. This allows the algorithm to take advantage of both horizontal and vertical directional 3D information in generating the refocusing image planes. Also, the final rendered image will increase resolution along both horizontal and vertical directions.



The proposed method works by extracting the viewpoint images as illustrated in Fig 3.5. The viewpoint image is a low-resolution orthographic projection type of rays from a particular direction. To generate high-resolution images at a particular plane one requires a new interpolation technique, which involves up-sampling, shift and integration of viewpoints. In up-sampling stage, the viewpoints are up-sampled using bi-cubic interpolation before shift and integration of viewpoints. This enhances both visual quality and resolution of the final image, which is an improvement against what was mentioned earlier in the literature review, as to the state-of-the-art refocusing algorithms suffered from poor resolution and artifacts in the final image. In addition, to generate the all-in-focused image, different depth planes are obtained. The Michelson contrast algorithm [15] is applied on an individual plane of the selected window size. The highest contrast will return a window size where the plane is focused and the position of the shift is recorded as a disparity value. The disparity value can be used later to generate the depth map of the scene to benefit coding, transmission, interactive 3D display, as well as interactive video games.

### **3.1 Operational characteristics of the holographic 3D imaging system**

Holographic 3D imaging system involves two processes, namely recording and replaying as shown in Fig 3. 1a. In the recording process, an object is imaged through an array of lenses, where each microlens captures a perspective 2D elemental image of the object from a specific angle. The final captured image contains the intensity and directional information of the corresponding 3D scene in 2D form. The key aspect of holographic 3D imaging principle is that its viewpoint images are orthographic i.e. sets of parallel rays are considered to be projected at various angles from the object, forming viewpoint images (VPs). The replay phase works in the reverse manner of the pickup; the elemental images are projected through the microlens arrays to optically reconstruct the 3D object at the same depth as the original object location, as shown in Fig 3. 1b.

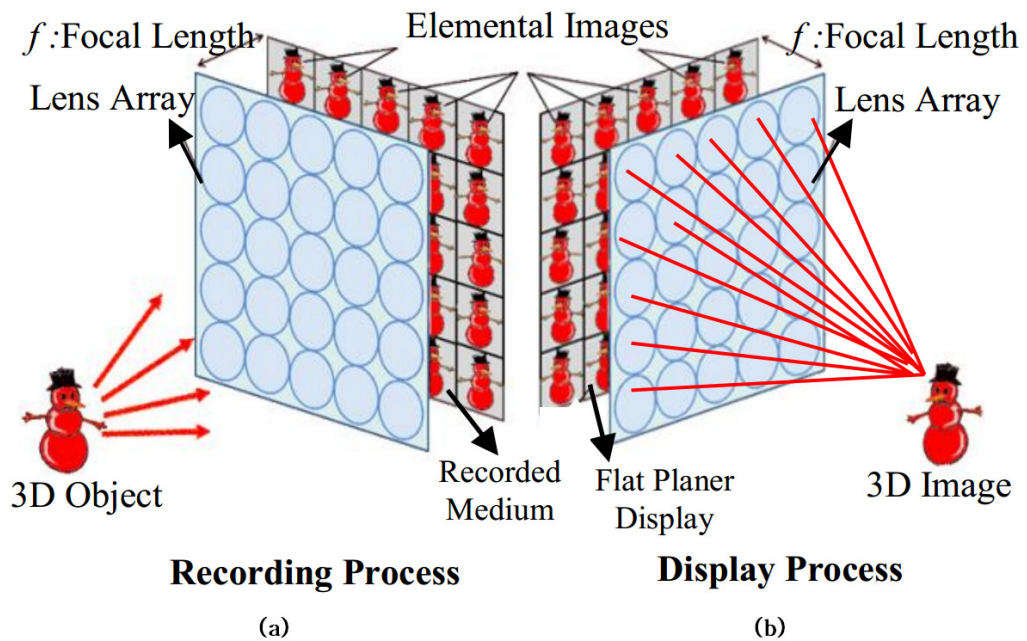


Fig 3. 1 : Holographic 3D imaging principle: (a) recording and (b) display process

### 3.2 Flowchart of the proposed method

Fig 3. 2 presents the flowchart of the proposed method that shows the steps and processes of how to acquire the all-in-focused image, disparity map, and refocusing image planes. The processes of up-sampling, shift, and integration of viewpoints enable us to focus at particular depth of plane with a given shift value after capturing. Therefore, at each shift's value, the point is focused at a particular depth of plane. Thus it allows us to change the depth of field computationally at any desired plane. Furthermore, in obtaining the all-in-focus image and depth information, the Michelson contrast estimation [16] is applied on all depth planes. Finally, the depth information of the objects is extracted by examining the point in space. At the focused point, the Michelson contrast estimation reaches its highest and blur is reduced to its lowest values [15]; whereas if the contrasts decreases and blur increases, the depth plane would be moving away from the object point. Therefore, the highest contrast with the lowest blur will return the object's original position. This means that highest contrast window from different depth planes will return the all-in-focus image.

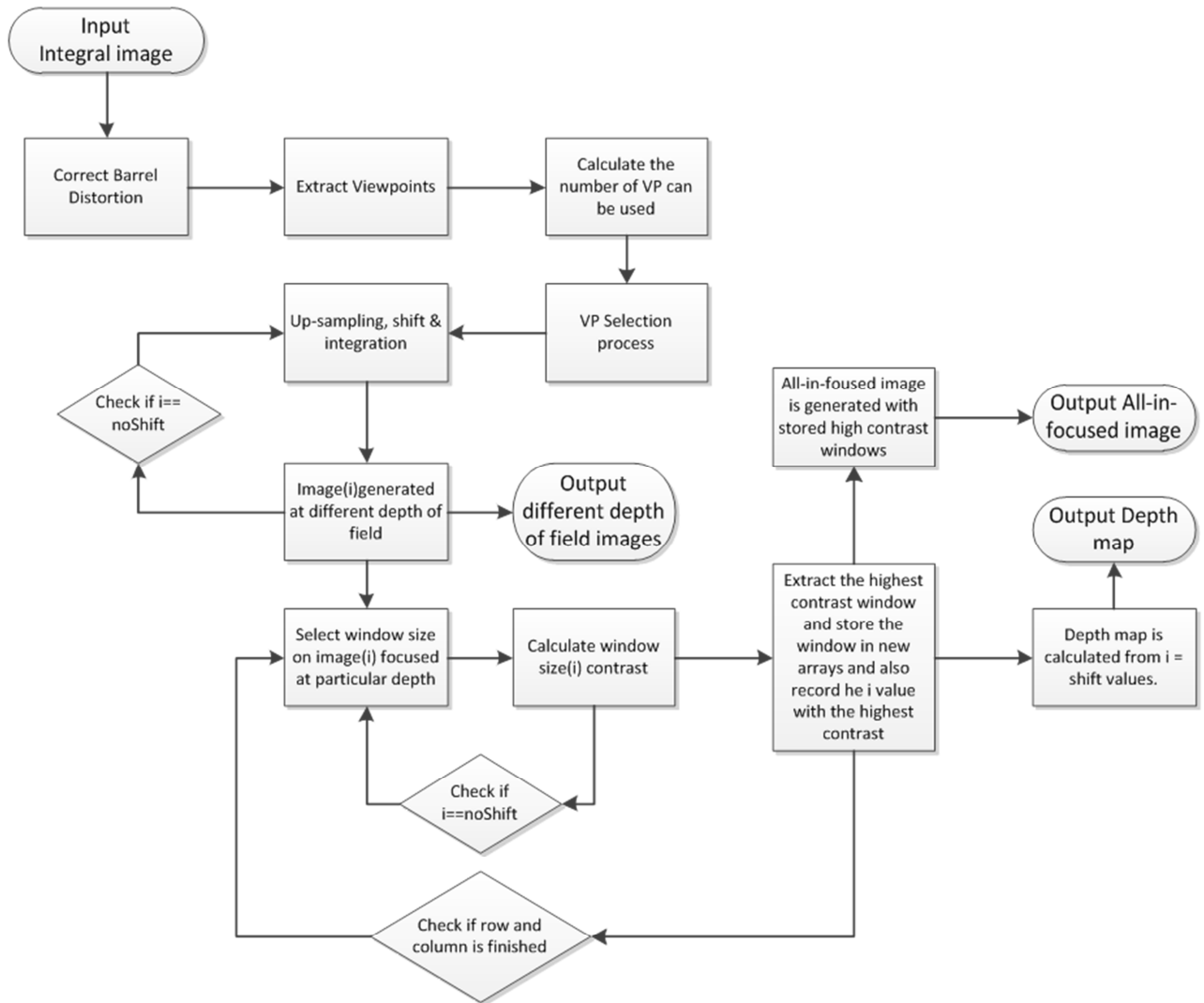


Fig 3. 2 : Flowchart of the proposed method

### 3.3 Viewpoint Construction in Holographic 3D Imaging System

The holographic 3D camera is a single aperture camera that is capable of capturing both intensity and directional information in 2D, at fixed-time. Both UH3I and OH3I are shown in Fig 3. 3.

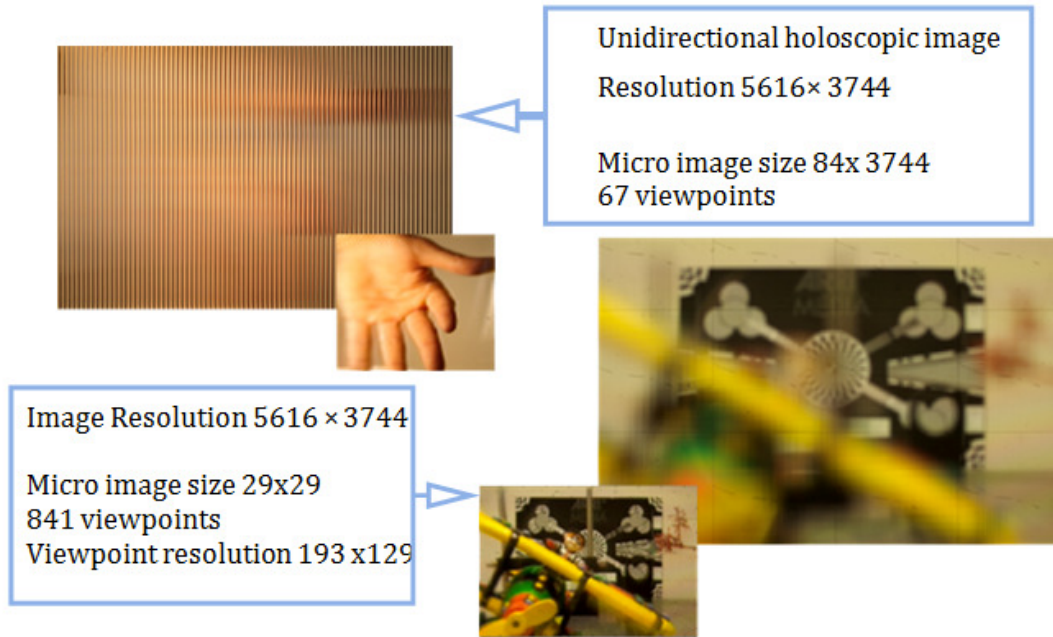
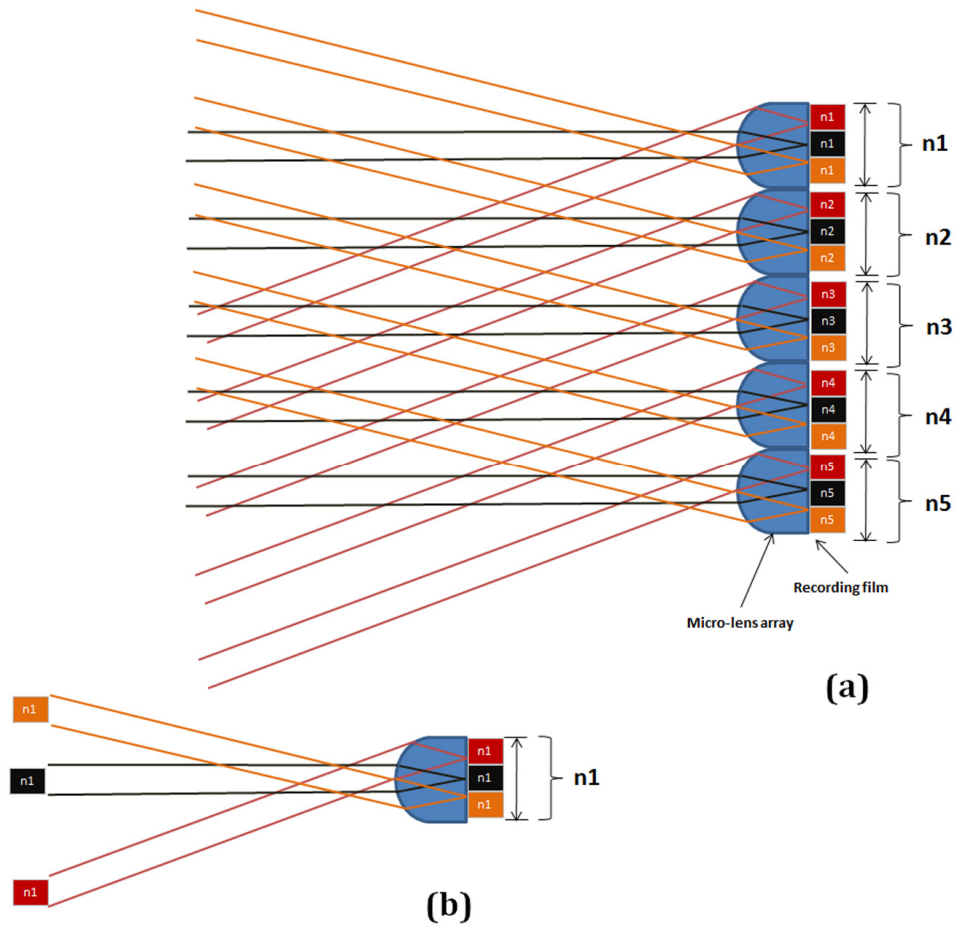


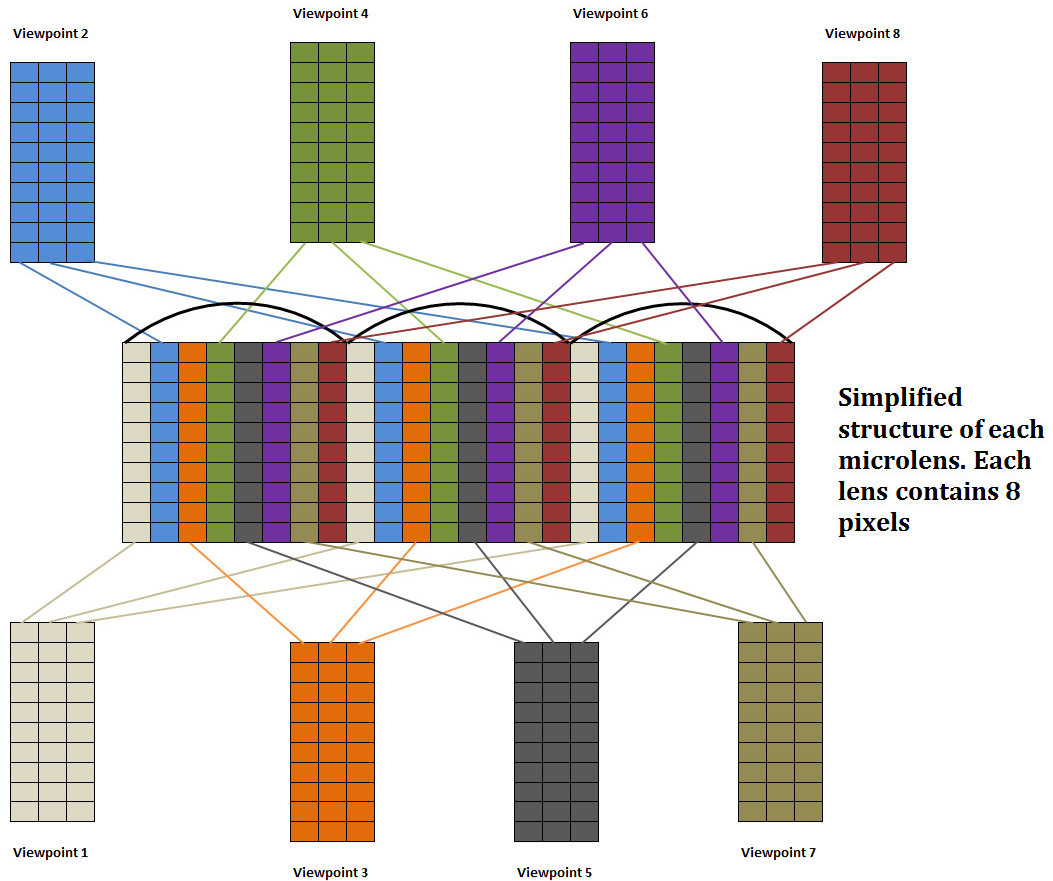
Fig 3. 3 : Holographic 3D Images with its constructed viewpoint image

The real world light rays are captured via a microlens array as shown in Fig 3. 4(a). The rays marked n1, n2, n3, n4, and n5 represent different perspective views of the same scene. Since a microlens array is involved in the recording stage, the local pixel position under each microlens contains directional view of the scene as shown in Fig 3. 4(b).

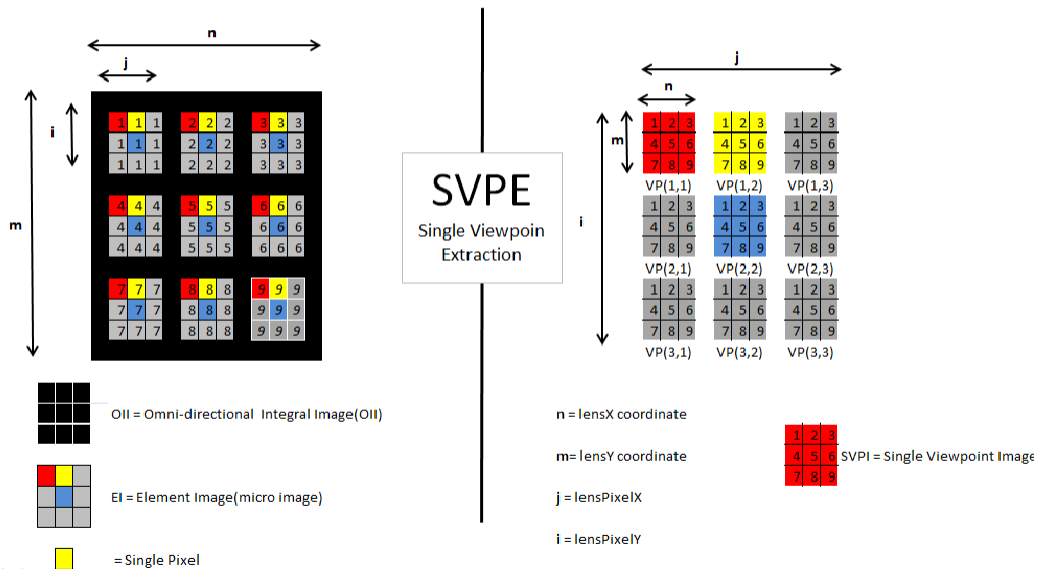


**Fig 3. 4 : Hologscopic 3D capturing systematic**

A single orthographic (VPI) is reconstructed by sampling of all pixels in the same location under different microlenses; this creates a perspective image that portrays one directional view of the scene. The construction of viewpoint images in both UH3DI and OH3I are graphically illustrated in Fig 3. 5. It is also mathematically expressed in eq 3.1.



(a): For illustration purpose, suppose there are 8 pixels under each microlens. Extract one pixel from the same position under different microlenses. placed in orderly fashion it will form one viewpoint image



(b): For illustration purpose, suppose 3x3 pixels under each microlens. Extracting one pixel from the same position under different microlenses and placing them in an orderly fashion to for one viewpoint image.

**Fig 3. 5 : illustration of (a) UH3DI viewpoint image extraction, (b) OH3DI viewpoint image extraction.**

$$O(i, j) = I(i + nk, j + mp) \quad \text{eq (3.1)}$$

The above equation describes the viewpoint sampling, the  $n$  and  $m$  are the pixel co-ordinates under micro-lens of  $j$  and  $i$ , where  $j = 1$  to  $k$ ,  $i = 1$  to  $p$ ,  $n = 1$  to  $N$  and  $m = 1$  to  $M$  are the horizontal and vertical positions of an Omnidirectional Image (OI)'s pixel respectively as shown in Fig 3. 5b. It is important to mention that each individual viewpoint can also be defined as  $VP_{j, i}(n, m) = OI(j, i, n, m)$ ,

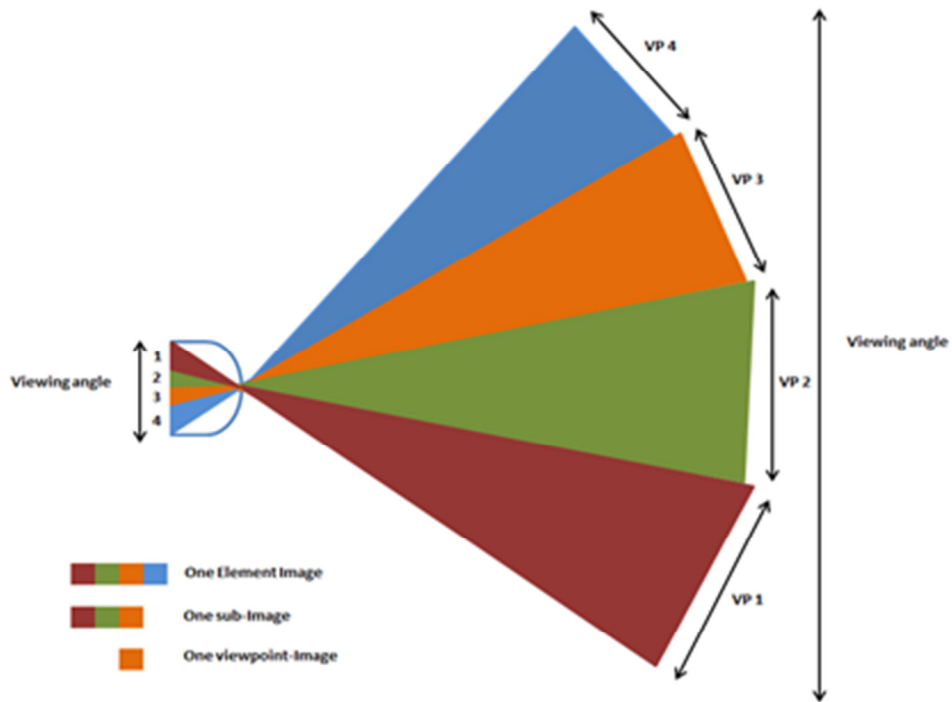


Fig 3. 6 : Illustration of the viewpoint (VP), elemental image (EI) and sub-image (SI).

(For illustration purposes, suppose there are 4 pixels under each microlens. One pixel under each microlens is defined as one viewpoint. Whereas a group of pixels under the same microlens are defined as a sub-image and the whole image under the microlens is known as elemental image.

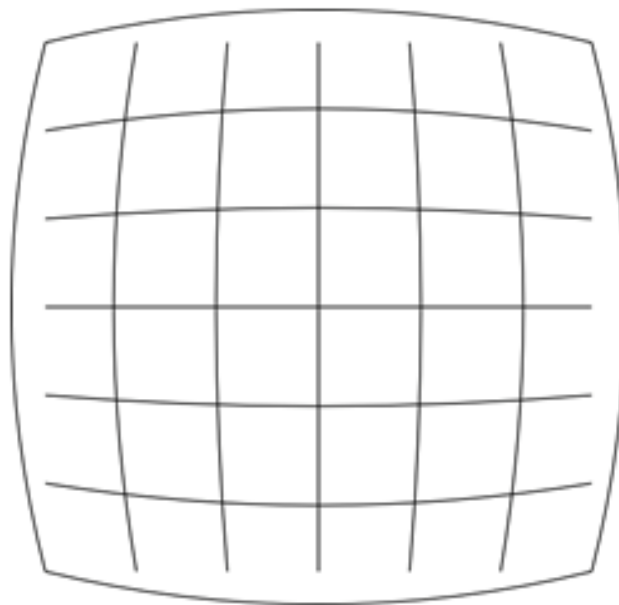
where  $n, m$  are the co-ordinates of parallel light rays that is different from perspective image property. Therefore, the final output  $O(i, j)$ 's image resolution is equal to  $(n \times m)$  pixels.

It should be noted that viewpoint images are different from elemental images (EI) and sub-images (SI). EI is defined using light field terminology  $EIn, m(i, j) = OI(j, i, n, m)$ . An elemental image is the recording image under the recording microlens where the resolution is defined as  $j \times i$  pixels. On the other hand, the sub-image is defined as a group of adjacent pixels under the

same microlens that are responsible for large spatial angle; but it is not the total number of pixels under the same micro image like E1, as shown in Fig 3. 6.

### **3.3.1 Lens error correction before viewpoint image extraction**

The UH3DI and OH3DI data are acquired by placing the microlens array in front of the camera sensor, enabling each microlens to capture the 3D scene from different directions. The most common distortion caused by the lens is barrel distortion, which is the result of fitting the image in a smaller space. The squeezing of image varies radially due to the design of the lenses—making it more visually prominent at the corner and sides of the image, as can be seen in Fig 3. 7. This can be neglected in most of the image applications where the visual barrel effect cannot be noticed by the human eye. However, it does effect in viewpoint image extraction process which requires extracting excel pixel of the same location from different elemental images. Thus, the image distortion needs to be corrected before proceeding to viewpoint extraction. Fig 3. 8a shows a viewpoint image extracted without correcting its distortion. Notice that the final viewpoint image looks unnatural due to being unable to extract the same positioned pixel under different elemental images, leaving out a portion of the scene and part of the object.



**Fig 3. 7 : Illustration of barrel distortion effect**



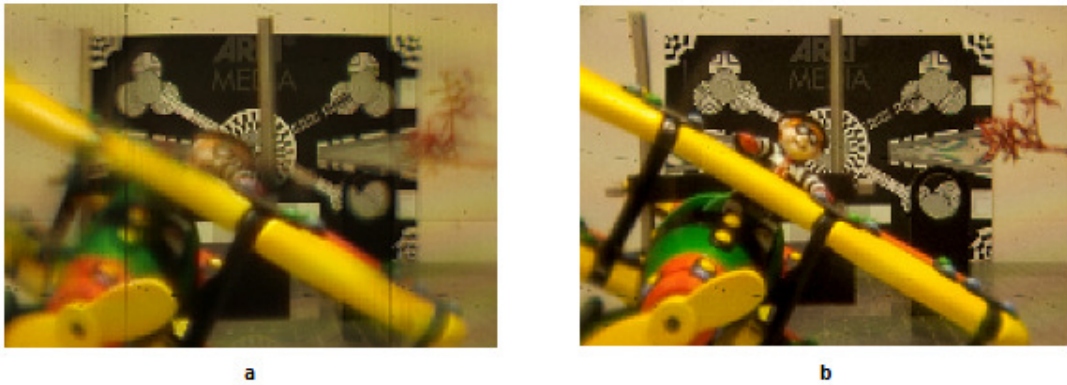


Fig 3. 8 : (a) VI (25, 25) without barrel distortion correction. (b) The same VI with barrel distortion correction

At this point the lens correction is used to reduce such barrel effect, as much as possible. Some lenses exhibit much less distortion than others. The quality of a lens and its type usually determine how much distortion occurs [7]. Lenses with a single focal length, also called prime lenses, tend to produce less distortion because there are fewer elements and a reduced need for optical compromise. It can also be optimized for its particular focal length. While the zoom lenses involve many elements and some compromise, wide-angle involves more compromising. However, all lenses produce distortions. While in the cheaper type of lenses they may be quite prominent and robust, they can hardly be seen in the very best of lenses (see Fig 3. 9 for lens types). During capturing, prime type lens were used together with micro-lens in 5D canon camera.

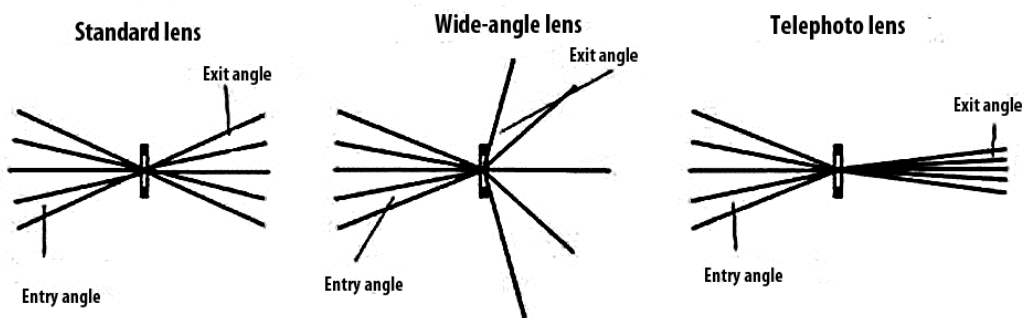


Fig 3. 9 : Representation of lens types [17]

As mentioned above, the captured holographic 3D images have barrel distortion for which it is necessary to use Photoshop tool to reduce the barrel distortion on both UH3DI and OH3DI. In this case, raw data of the acquired image is used, because data is in its richest form without any compression or interpolation of

any type, before correcting the barrel distortion. Otherwise the distortion may be harder to correct, because the intensity of a pixel value spreads across its neighboring pixels, resulting in a decrease in the richness of pixel intensity. The raw data enables greater flexibility in adjusting data parameters; for example brightness, contrast and colour as shown in Fig3. 10.

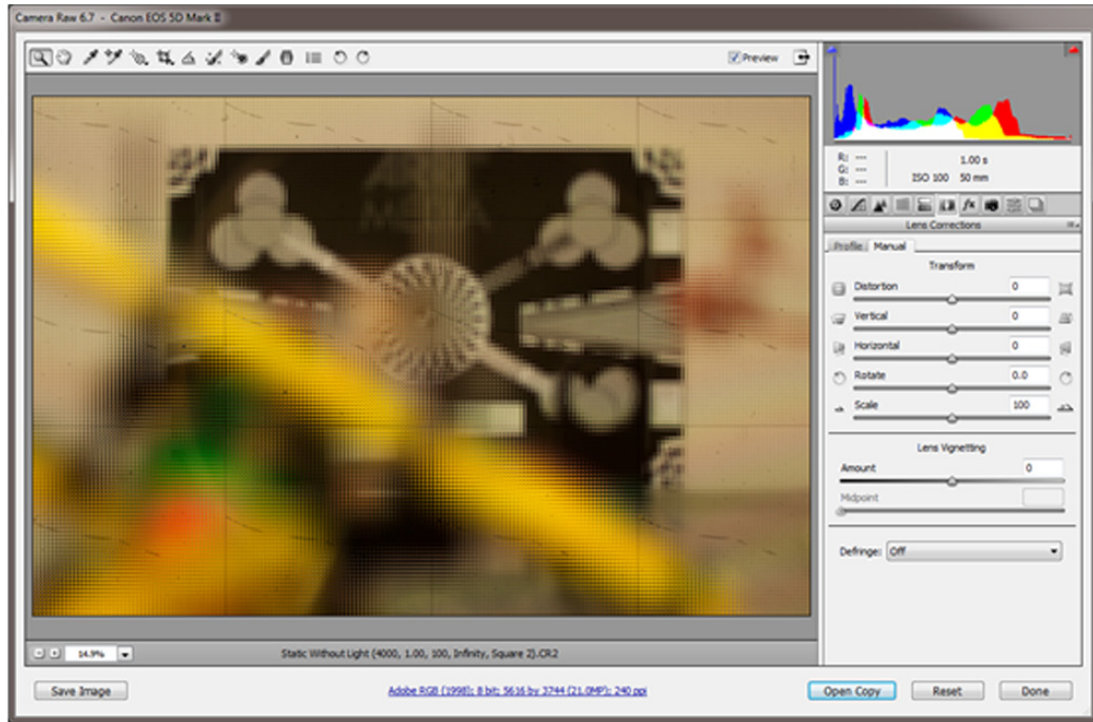
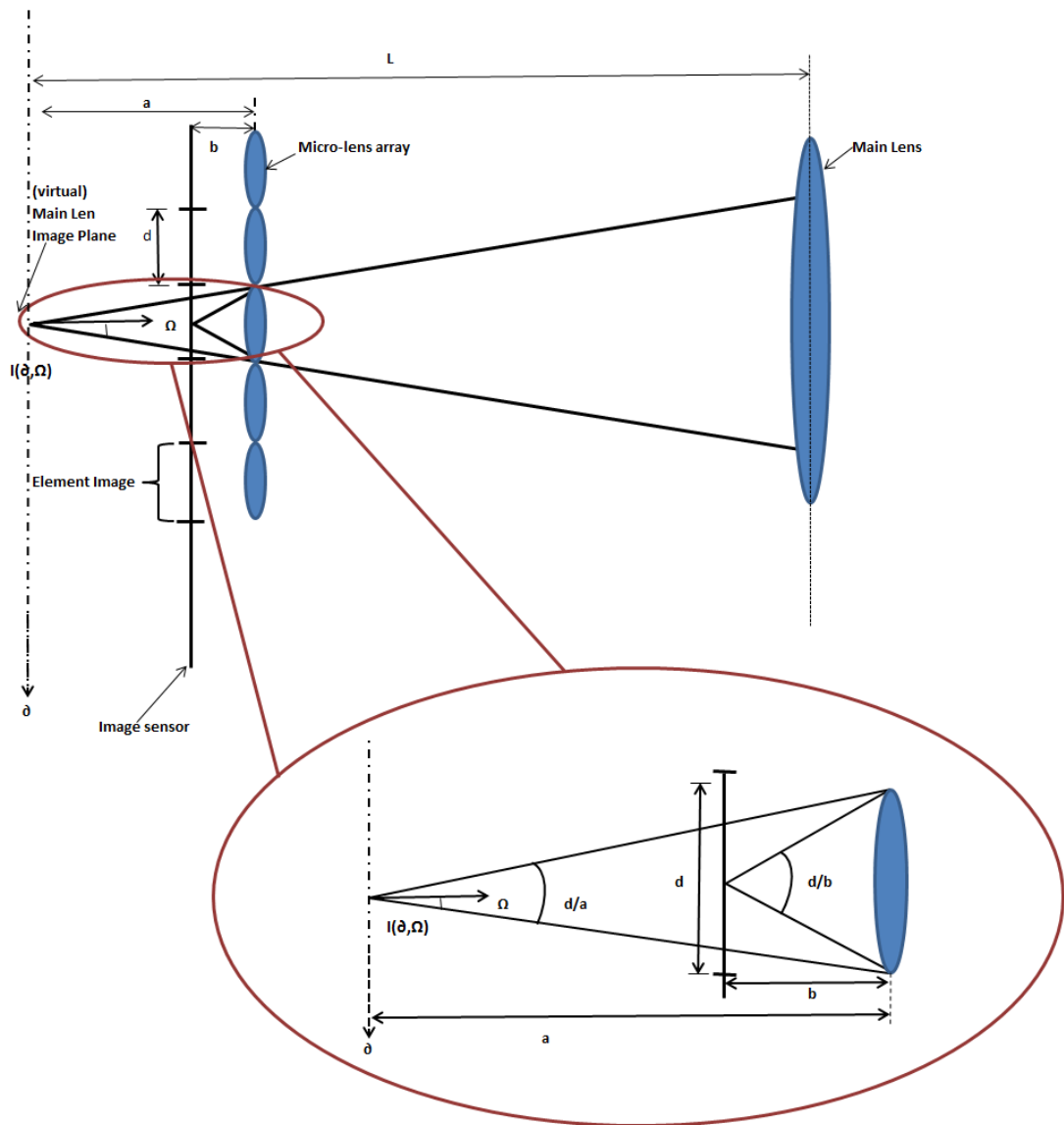


Fig 3. 10 : Illustration of Photoshop lens correction tool

### 3.4 Digital Refocusing

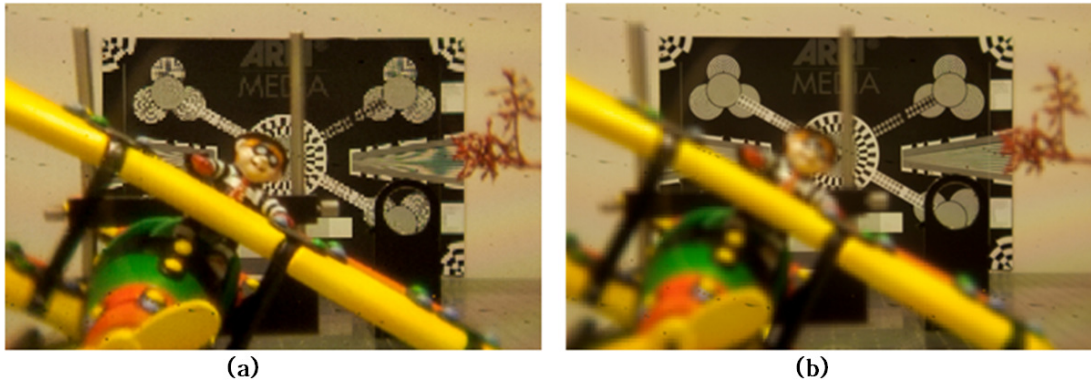
Digital refocusing in holographic 3D imaging means changing the focus plane after the picture is acquired. This is achieved by converging all the light rays to a desired virtual depth plane. The viewpoint rendering pixel manipulation method is used to refocus at different depth planes. In this chapter, the OH3DI are captured in Galilean mode. The main lens is focused behind the image sensor creating a virtual parallax image and each microlens satisfies the equation  $-1/a+1/b=1/f$ , as the distance remains within the same distance from the microlens, but in the opposite direction. Therefore,  $1/a$  in Keplerian mode changes to  $-1/a$  in this setup [13] as shown in Fig 3. 11.



**Fig 3. 11 : Focused Plenoptic 2.0, Microlens imaging in Galilean mode. The main lens image plane is virtually created behind the image sensor for the microlenses to capture.**

Full resolution rendering process results introduces unnatural artifact in areas that are not ‘in focus’ [10]. At this stage, the full resolution rendering is only used to determine the distant object in the OH3DI.

The full resolution rendering method works by selecting a number of pixels (sub-image) from every EI under the same position. Therefore, the size of sub-image determines the depth plane of the real world “in focus”, as can be seen in Fig 3. 11.



**Fig 3. 12 : Focused Plenoptic 2.0: Microlens image is in Galilean mode. Full resolution rendering applied on the above images: left image extracts foreground but leaves the background with artifacts as the size of sub-images is too small for the background, where**

To overcome the artifact problem, up-sampling, shift, and integration of viewpoint (VP) are performed to refocus at a different depth of plane. This helps to sustain the natural look of the photograph without affecting the image quality and resolution too much. Therefore, VPs are up-sampled by a factor  $N$  in horizontal and vertical directions before shift and integration process is applied.  $N$  is ascertained by knowing the farthest object from the camera. This is when the main lens image plane is in Galilean mode—where the farthest object from the camera appears to be the closest in the virtual image plane and fewer microlenses capture that point. Thus, knowing the farthest object will allow extracting the whole object points that appear in fewer microlenses. That means the other objects in the close distant contain all the possible information that can be refocused. The farthest object is discovered using the full resolution rendering, which brings only one of the image planes “in focus”. Depth plane is determined by the size of the sub-image under every EI and combining them together. Therefore, size of the sub-image gives the value of  $N$ , where every VP is up-sampled by  $N$  times. Also,  $N$  number of VP images is used in the refocusing stage to get different depth planes in focus. Up-sampled VPs are stacked adjacently in horizontal and top-to-bottom in vertical directions to form a 4D stack of images  $VP_{j,i,n,m}$ . Fig 3. 13 (b) shows the steps in achieving the refocused images using VPs pixel manipulation method.

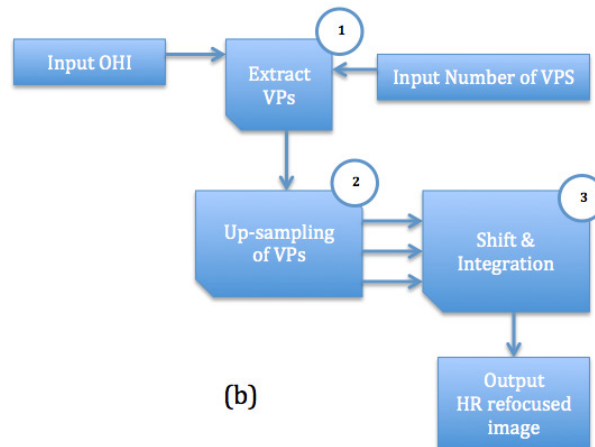
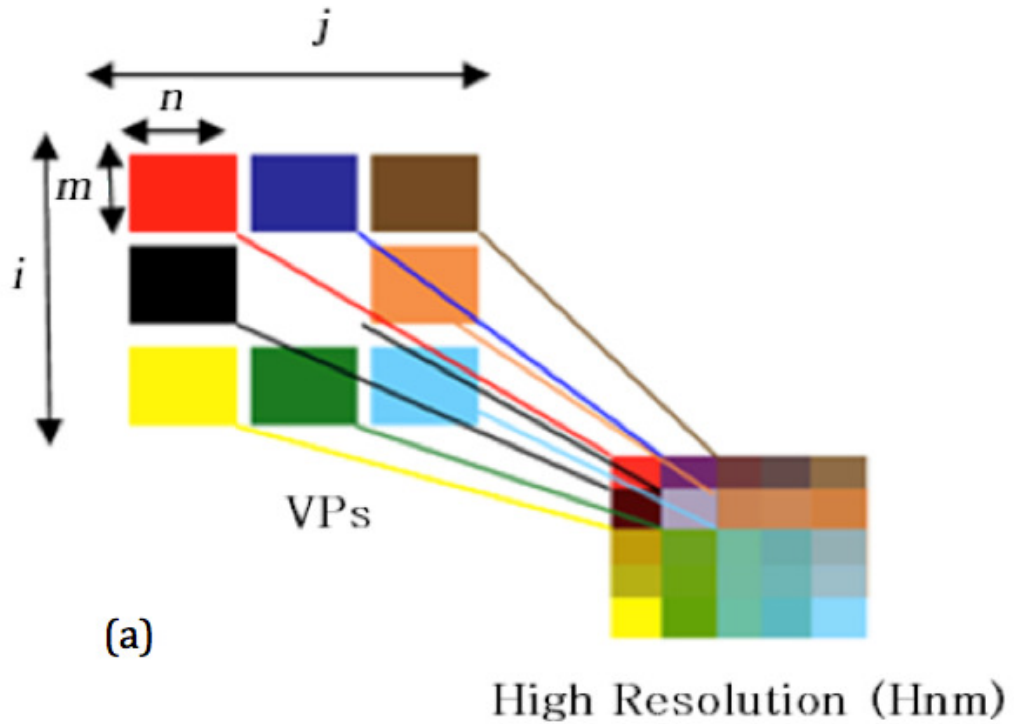


Fig 3. 13 : (a) Illustration of shift and integration of VPs to generate one high resolution image with a final image size of  $(n \times N) \times (m \times N)$ . (b) Present the steps for the proposed refocused image.

The Refocusing operation can be expressed algebraically for OH3DI in a concise form as shown in eq 3.2, where  $H_{nm}$  is the result of the integrated up-sampled VPs with coordinates  $n$  and  $m$ ;  $j$  and  $i$  are the indexed number of VPs ranging from 1 to  $N$ . Other parameters include the shift parameter  $\Delta$ , whose sign modifies the index  $n, m$ .  $i$  and  $j$  are the number of horizontal and vertical resolution elemental images. Each viewpoint is equal to the number of lenses

multiplied by the up-sampling factor. The amount of relative shift in the images, obtained by integration of viewpoints, determines the depth at which the sharp image is formed. This process is graphically illustrated in Fig 3. 13 and eq (3.2).

$$H_{nm} = \sum_{i=1}^N \sum_{j=1}^N VP_{S_{n \pm \Delta(1 \mp i)}, i, m \pm \Delta(1 \mp j), j} \quad \text{eq (3.2)}$$

The resolution enhancement in the refocusing process is explained schematically in a 1D example with two viewpoints represented by vectors; viewpoints integrating their pixel values with shown pixel coordinate within the circles in Fig 3. 14. When shifted by 1, whole pixel = 2 sub pixels—there is no resolution enhancement. This produces the same resolution image as integrating un-shifted viewpoints. Red arrows represent up-sampled sub pixels with the same values and coordinates as their blue counterparts. With half a pixel shift = 1 sub pixel, twice as many integration points are introduced. This is depicted by blue and red rays, integrating their pixel values in the ellipses, resulting in an enhanced resolution image, at a slightly different depth in the z direction.

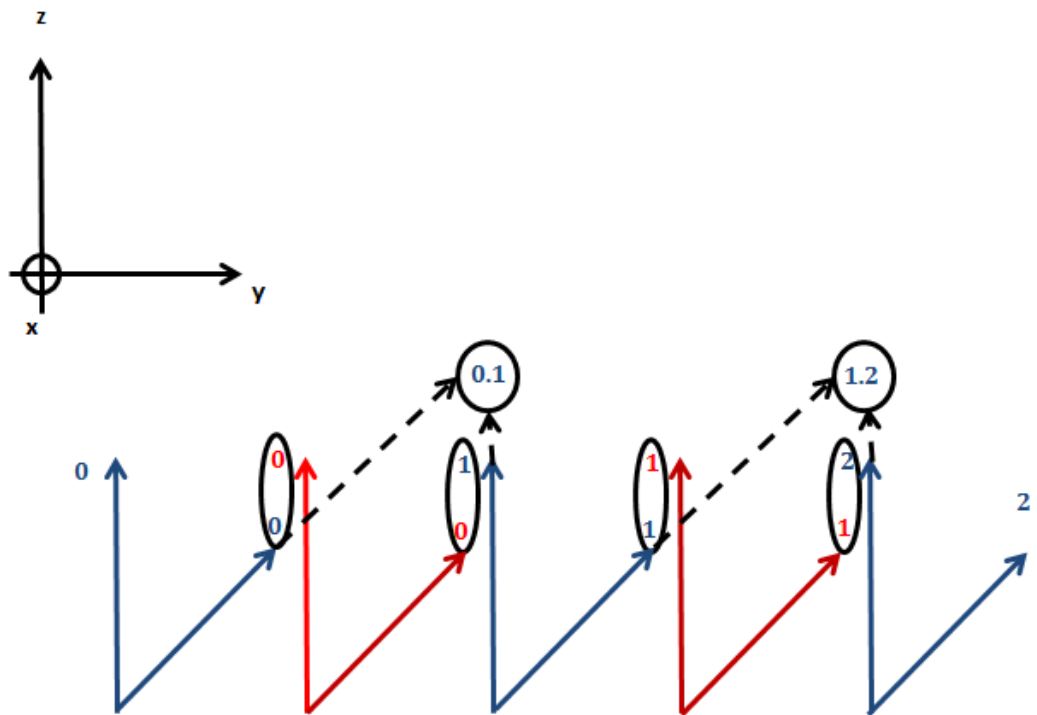


Fig 3. 14 : Schematic illustration of resolution increase

Integration of  $N$  VPs at different shift values results from a different depth plane in focus, as it can be illustrated in Fig 3. 15. To focus at depth plane  $z_5$ , the pixel one under  $n_1$  EI is intersected with pixel five under  $n_2$  EI and shift value equals to 4. The resolution enhancement is also demonstrated when focusing at the depth plane  $z_1$  in Fig 3.15 with up-sampling, shift, and integration process. The depth plane  $z_1$  can be seen from different EI by setting the shift value to 1; therefore, pixels under EI  $n \cdot \text{shift}$  will pick up the position point  $z_1$  from different EI. With up-sampling, shifting, and integration of one pixel shift will focus at depth plane  $z_1$ . Fig 3. 15b shows an enhanced resolution by representing one of the points in depth location  $z_1$  with up-sampling that gained 3 more pixels in comparison to standard shift and integration process as shown in Fig 3. 15c.

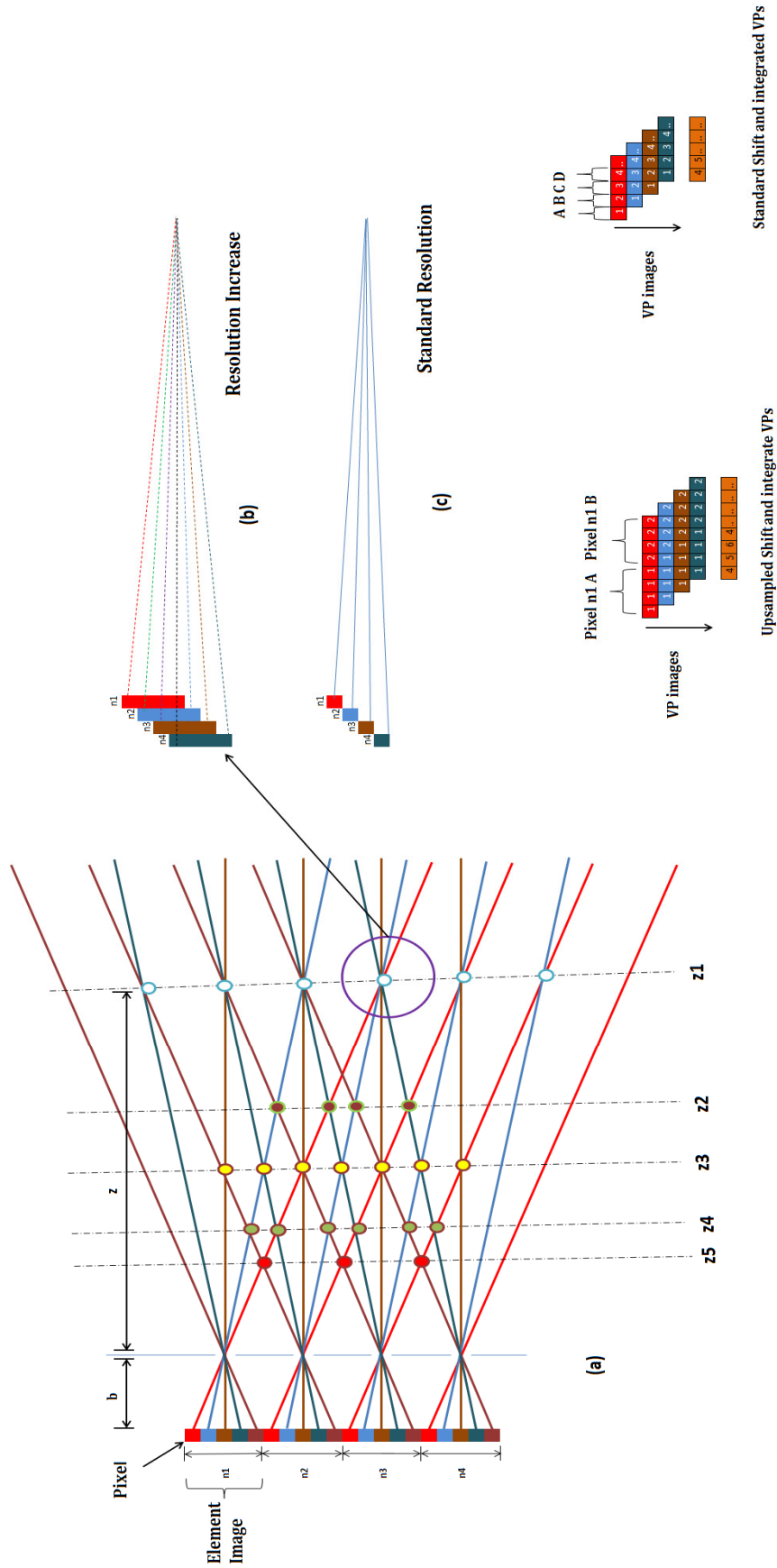


Fig 3. 15 : Ray tracing in hologscopic 3D imaging system



### 3.5 Depth Analysis

The analysis of depth planes with shift and integration process, using simple triangle geometry, is used to calculate the object distance from microlens to the main lens. The real depth can be calculated in relation to the real world distance, given the virtual distance from microlens to main lens.

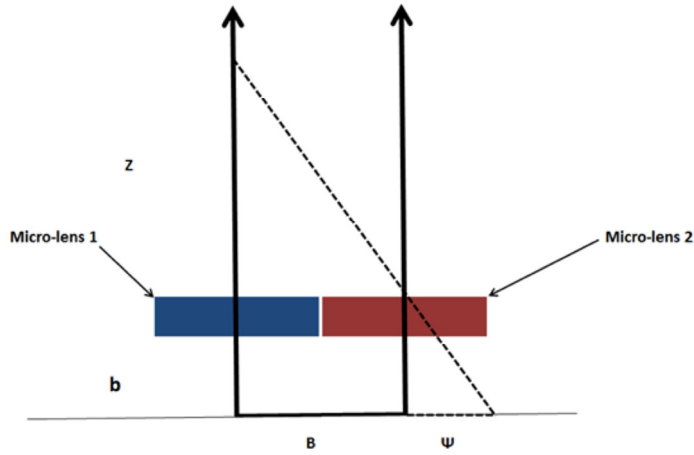


Fig 3. 16 : Depth Analysis

$$\frac{Z}{B} = \frac{b}{\Psi} \quad \text{eq (3.3)}$$

$$Z = \frac{SbB}{\Psi N} \quad \text{eq (3.4)}$$

Let  $S = 1$

$$Z = \frac{bn}{N} \quad \text{eq (3.5)}$$

The depth  $z$  inside the  $L$  shown in Fig 3. 11 is the point at which intersection occurs for  $N$  VPs forming an image plane at particular  $z$  distance with given shift value.  $n$  is the number of pixels under each lens,  $b$ = distance from microlens to image sensor,  $B$  = pitch,  $\Psi$  is width of a pixel =  $B/n$ , and  $S$ =shift. From simple triangle geometry, the distance  $Z$  can be calculated with equation 3.5. But as mentioned above, the camera mode is set to Galilean mode, which the equation  $z=(-bn)/N$  changes, because the image plane from main lens to microlens are set

behind the image sensor creating a virtual image plane. The distance  $z$  will always be equal to a native value since the object is in the virtual image plane. Hence, the closest object to the camera looks the farthest and vice-versa.

### 3.6 All-in-focus image

All-in-focus image is extracted by looking at all depth planes and returning areas, which have high contrast and low blur. The choice of one shift value returns one depth plane 'in focus' with integration of VPs as mentioned above. Thus, a different shift value would correspond to a different depth plane. In other words, the refocusing is accomplished through the choice of shift value with the integration of VPs. The final all-in-focused image process is given by the following equations.

$$AF = H \{F\} (n+K, m+K) \quad \text{eq (3.6)}$$

Where,

$$F = \arg\{\max\{W(S)\}\} \quad \text{eq (3.7)}$$

Where,

$$W(S) = \frac{H\{S\}(n, m)}{H(S) \epsilon K} \left[ \frac{\max_{(n,m)} - \min_{(n,m)}}{\max_{(n,m)} + \min_{(n,m)}} \right] \quad \text{eq (3.8)}$$

$$\left[ \frac{I_{max} - I_{min}}{I_{max} + I_{min}} \right] \quad \text{eq (3.9)}$$

The  $H(S)_{n,m}$  is the result of high resolution image, where the depth plane is dependent on the shift value ( $S$ ), and the number of shift returns the same number of high resolution image, i.e.  $S = 1, 2$ . All depth planes are stored in  $H(S)_{n,m}$  and each  $H(S)$  image is focused at particular depth within the scene. Therefore, to extract the all-in-focus image, the contrast values are calculated for each  $H(S)$  with the given local window. The contrast values are calculated with window block size ( $K$ ) in  $W(S)$  as defined in eq 3.8. The Michelson contrast [12] is used to calculate the contrast value for each local window in  $W(S)$ . The Michelson contrast is defined in eq 3.9 and used in eq 3.8 to define contrast of the local window within the image. The  $\max_{(n,m)}$  and  $\min_{(n,m)}$  represent the highest and lowest luminance in the local window respectively. The highest

contrast score window within  $W(S)$  is selected and stored in  $F$ —indicating the depth plane where the objects are ‘in-focus’. Furthermore, the final image is rendered in  $AF$  with window size ( $F$ ) of  $H \{F\}n,m$  at higher contrast with lower blur to extract all-in-focus image as shown in eq 3.6 and Fig 3. 17.

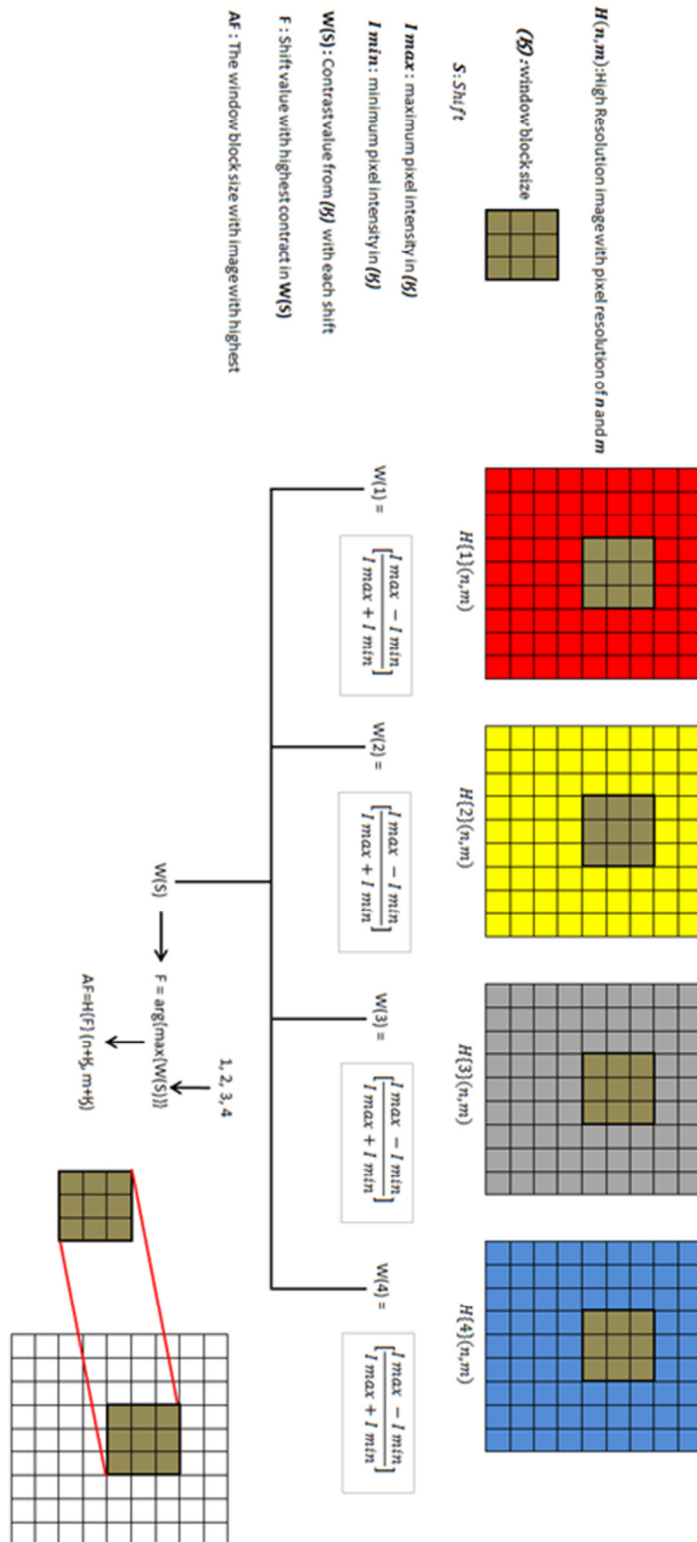


Fig 3. 17 : All-in-focus image rendering algorithm creates a final image using high resolution images

### 3.7 Experimental Results

The experiment demonstrates the success in acquiring refocusing, all-in-focus image, and depth map results. One of the setup scenes used in this experiment is illustrated in Fig 3. 18, where the objects are placed in a precisely measured distance from the camera. Each object is named 'Target' with the recorded distance from the camera's microlens, where Target 1, Target 2, Target 3, and Target 4 are located at  $z_1=3190\text{mm}$ ,  $z_2=2000\text{mm}$ ,  $z_3=1000\text{mm}$ , and  $z_4=700\text{mm}$  respectively as show in Fig 3. 18. In the recording process, 3D objects are captured in 2D format by microlens array placed in front of the camera sensor, enabling each microlens to capture the objects from a particular direction. Therefore, the outcome H3D image holds both direction and position of the scene. 5D canon camera was used with 50mm main lens and 21-megapixel image resolutions. The main lens is attached with a mountable extension tube on the camera to provide a flexible way of adjusting the distance between the main lens image plane to the microlenses and microlenses to the image sensor. The microlens focal length is 0.025mm and pitch size is 0.9mm. Furthermore, the main lens aperture is modified from circle to square, to achieve a more effective way of using sensor space, as the microlenses are in square.

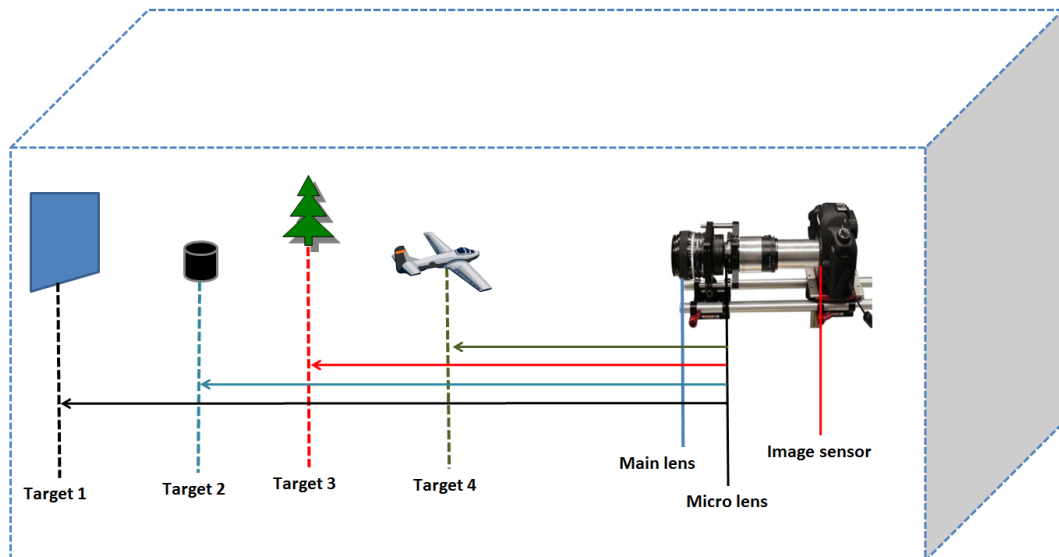
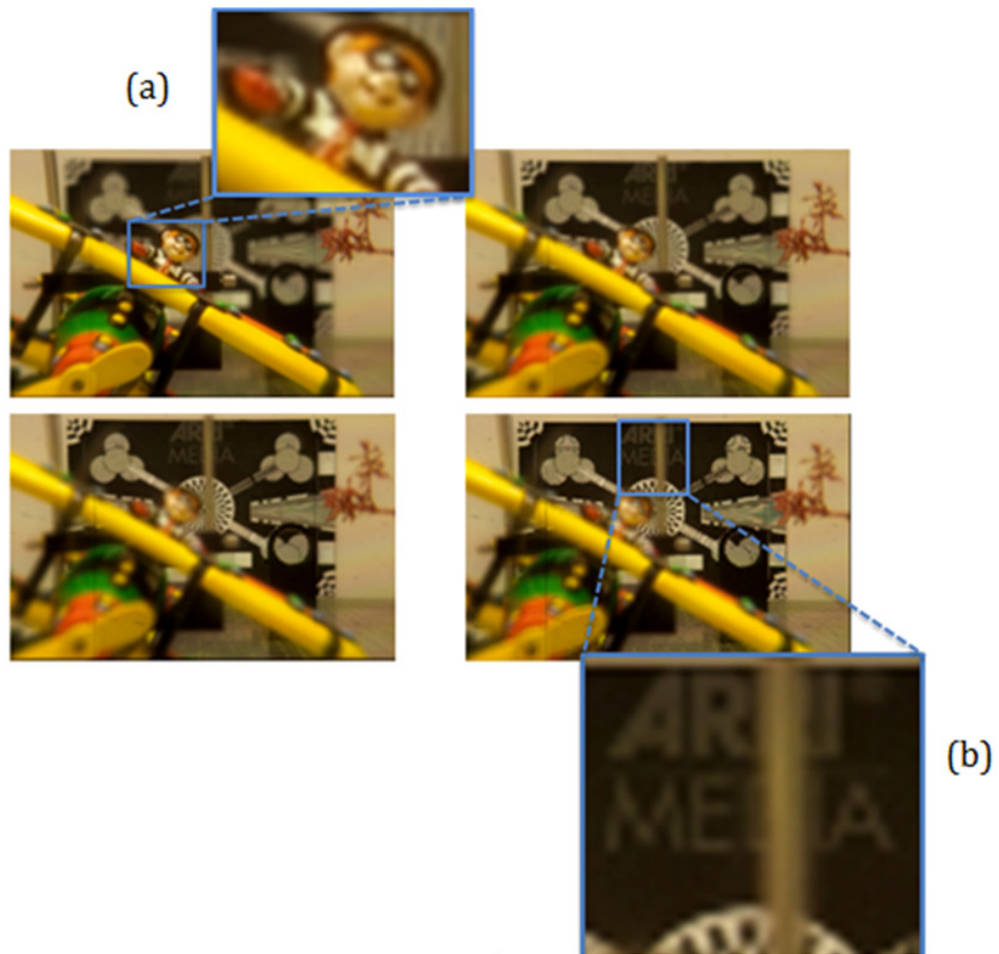


Fig 3. 18 : Illustrates one of the experiment setup scenes.

### 3.7.1 Subjective Quality Comparison



**Fig 3. 19 :** Native shift and integration refocusing is illustrated: (a) shows the magnified part of final refocusing image where the focus is at the object. (b) is focused at the background: Notice both (a) and (b) images are in poor quality, containing blocking artifacts with significant noise that is seen more pixelated with naked eyes.

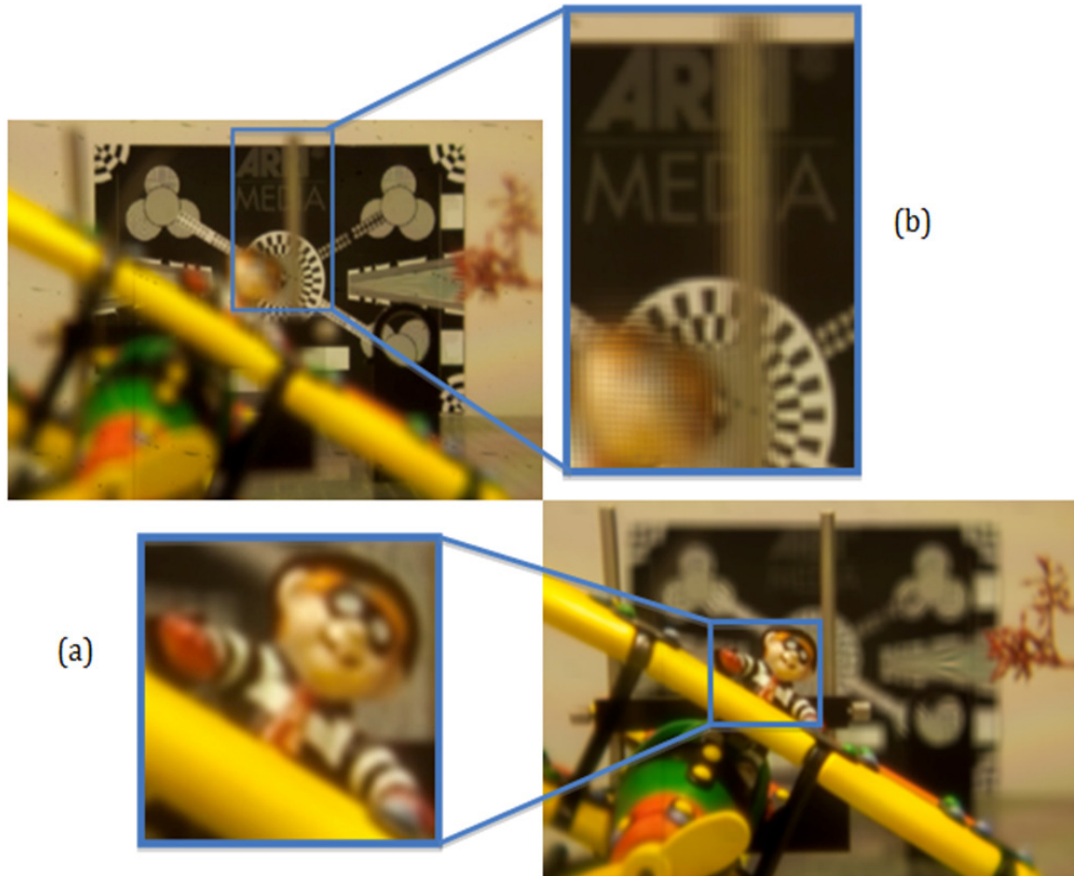


Fig 3. 20 : Up-sampling, shift and integration refocusing using 7 by 7 VPs: (a) shows the magnified part of Target 1 that is the ARRI Media test chart. (b)Target 4 is focused at the toy. The final image is at resolution of 1344 x 903 pixels.

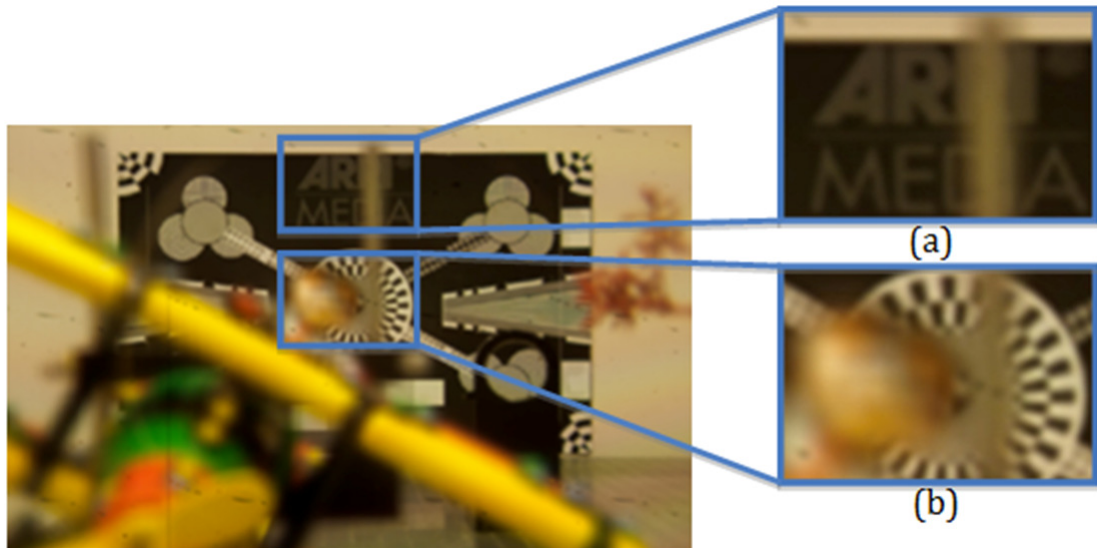


Fig 3. 21 : Up-sampling, shift and integration refocusing using 7 by 7 VPs:(a) shows the magnified part of Target 1 that is the ARRI Media test chart. (b)Target 4 the blur looks natural and no artifacts.

### **3.8 Analysis and Discussion**

The OH3DI is acquired at resolution of 5616 X 3744 pixels that contains 193 X 129 elemental images, which have resolution of 29 X 29 pixels. VP resolution is determined by the number of microlens contained in the recording, which is the same as the number of microlenses i.e. 193 X 129 pixels. The result of applying up-sampling, shift, and integration is to enhance the resolution of the final image in comparison to the traditional refocusing method using VPs manipulation method. Both the methods are compared with each other, as it is clear that the traditional refocusing outputs the same resolution as its VP resolution. The up-sampling, shift, and integration, on the other hand, outputs an image equal to the VP resolution multiplied by the up-sampling factor. Resolution result on normal shift and integration is clearly shown in Fig 3. 19. The final refocusing images in normal shift and integration suffer from poor resolution, as its resolution equals 193 X 129 pixels.

However, applying the up-sampling, shift, and integration algorithm on the same OH3DI, results in a significant increase in resolution and quality of the final refocused images, as shown in Fig 3. 20. An Arri Media test chart is used to determine the effect in comparing both results. The native and proposed method used 7 by 7 VPs in acquiring different depth planes, where the value 7 is obtained by using the high-resolution rendering. This is to locate the distant object plane in the scene that uses patch size. The size of the patch initializes the number of VPs ( $N$ ), which is required in the refocusing. Note that in Fig 3. 18(b) the "ARRI MEDIA" is successfully reconstructed without having the effect of blackening artifacts and noise, leading to an increased resolution and quality of the final image.

Also note that in Fig 3. 20b artifacts arise on the final image. This is due to the focus being at a greater distance from the optical image plane. In other words, the artifacts are more visible in the close up object when the focus is on the far away distance. Therefore, enhancement was made to cure the artifacts by having a smoother transition of VP's pixels integration to gain a natural photographic looking image. The VPs are interpolated, up-sampled before shift and integration of VPs. The VPs are up-sampled by  $N$  times using quadratic

interpolation. As a result the final refocused image compensates for the resolution factor, as well as visual quality by sustaining the natural photographic look (see Fig 3. 21).

Also this method is applied to UII with pitch 1.65mm,  $f$  2mm and 67pixels per lens at VP resolution of 84 x 3744 pixels shown in Fig 3. 22. 5 VP images are used in up-sampling, shift, and integration process to improve the visual resolution quality in the refocus image. Now, it is possible to obtain a high resolution refocused image from the VP approach, compared to native VP refocusing, and by forming additional integration points at user defined depth using up-sampling, shift, and integration algorithm.

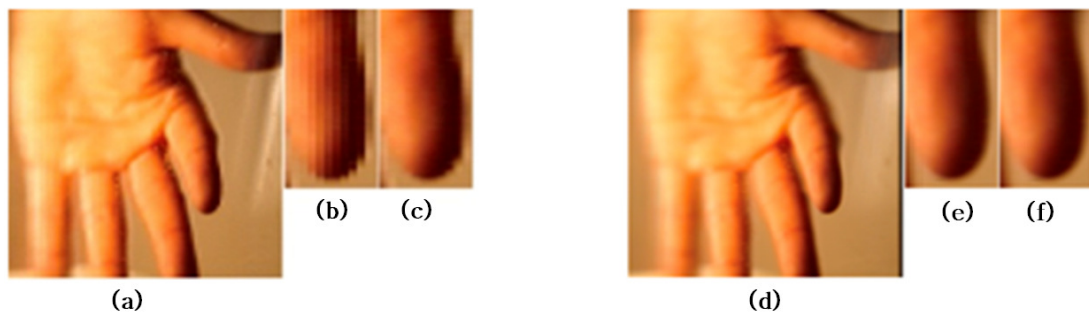


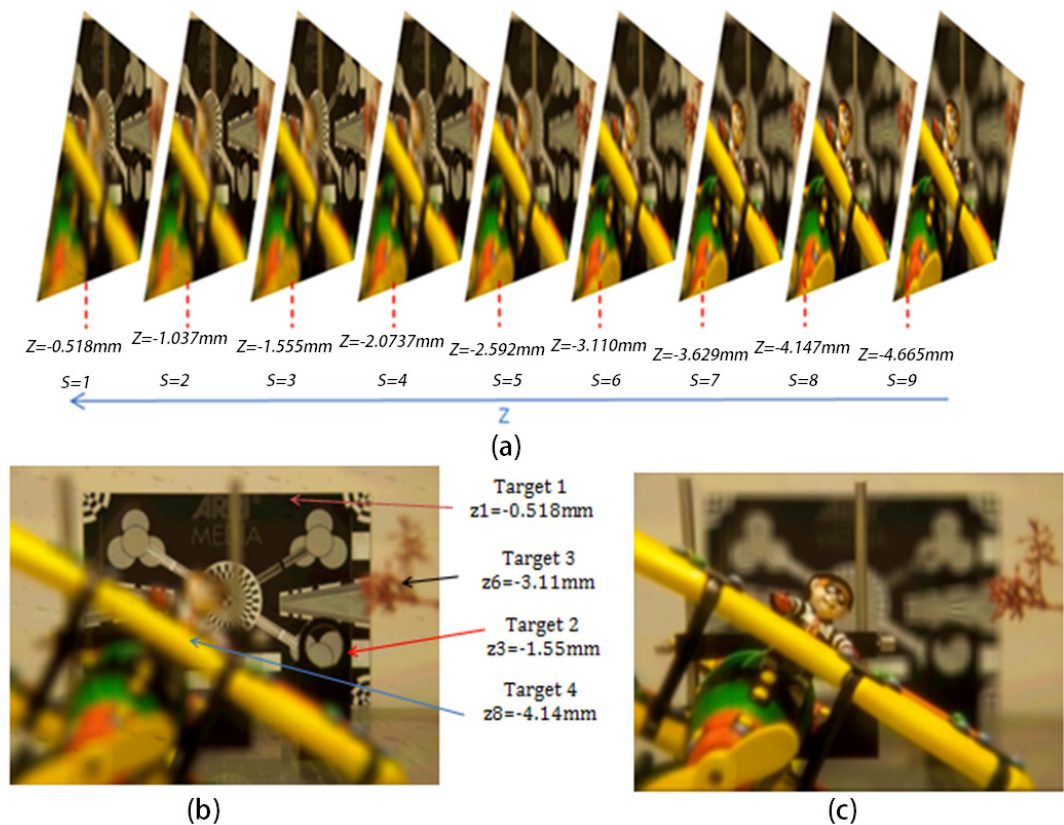
Fig 3. 22 : Rendered VP of UH3DI

Fig 3. 22a shows processed VP of UH3DI upsampled by 5 along x axis. Fig 3. 22b and (c) show magnified region of (a) without and with interpolation, respectively. Fig 3. 22d is result of shift and integral of 5 VPs, while (e) is the magnified region of (d) without using bi-cubic interpolation during VP up-sampling and (f) is that with bi-cubic interpolation. Fig 3. 22arepresents 420 x 3744 pixels, while (b) and (c) are original image sizes, showing detail of forefinger without and with interpolation, respectively. After shift and integration algorithm is used to increase the resolution by selecting four neighboring VPs, the result is shown in (d) at resolution 420 x 3744. (e) and (f) represent the improved versions of the forefinger image using the algorithm without and with interpolation, respectively.

The obtained high-resolution images are focused at particular depth, therefore Michelson contrast algorithm is used in all the depth plane images to return all-



in-focus image, as shown in Fig 3. 24. The depth plane is dependent on the choice of shift value. In the experiments the shift values are selected from 1 to 9, as the result of the different depth planes are extracted (see Fig 3. 23a). At each shift, a different plane is 'in focused' and virtual depth  $z$  is also calculated at each shift with a given eq 3.5, where  $N = 7 * \text{pixel size}$  (0.0031mm),  $f = 0.025\text{mm}$ ,  $b = 0.127\text{mm}$  and  $n$  is the total number of viewpoints  $29 \times 29$  that equals to 0.09mm. The  $N$  is the acceptable number of VP that can be used in the refocusing process, which is equal to  $N= 0.0217\text{mm}$ .



**Fig 3. 23 :** 7 by 7 VP are used in refocusing process to extract different depth plane in (a). In (b) where focus is on the background with virtual depth  $z=-0.518\text{mm}$  and (c) focuses at foreground with virtual depth  $z=-4.147\text{mm}$ .

In addition, all-in-focus image is generated by using Michelson contrast [14] algorithm on each depth plane with a window size of  $20 \times 20$ . The highest contrast values with the lowest blur in the same window locations from the other nine planes are extracted to identify windows where the object is in focus, at a given shift plane. The highest contrast window's shift with the lowest blur are recorded, which can later be used in calculating the virtual depth, as shown in Fig 3. 24. Depth  $z$  is calculated by shift values  $S$  using eq 3.5. The size of the

window in calculating the contrast has great impact on the final all-in-focus image as well as the virtual depth information. Fig 3. 26 shows the result using different window sizes and it is noticed that a large window size generates better results in comparison with small window sizes.

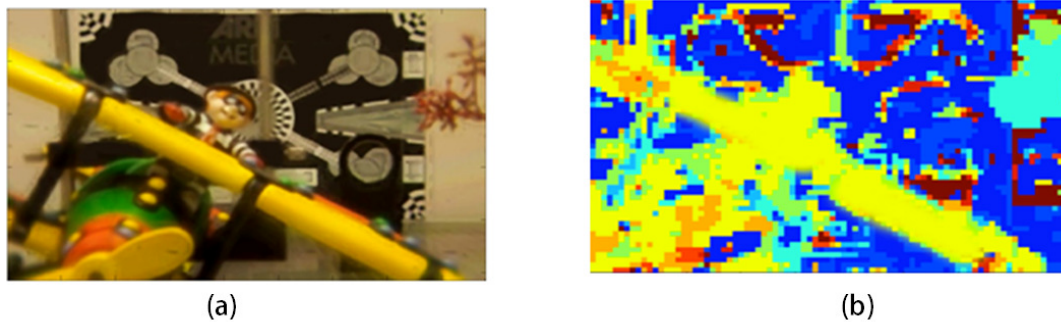
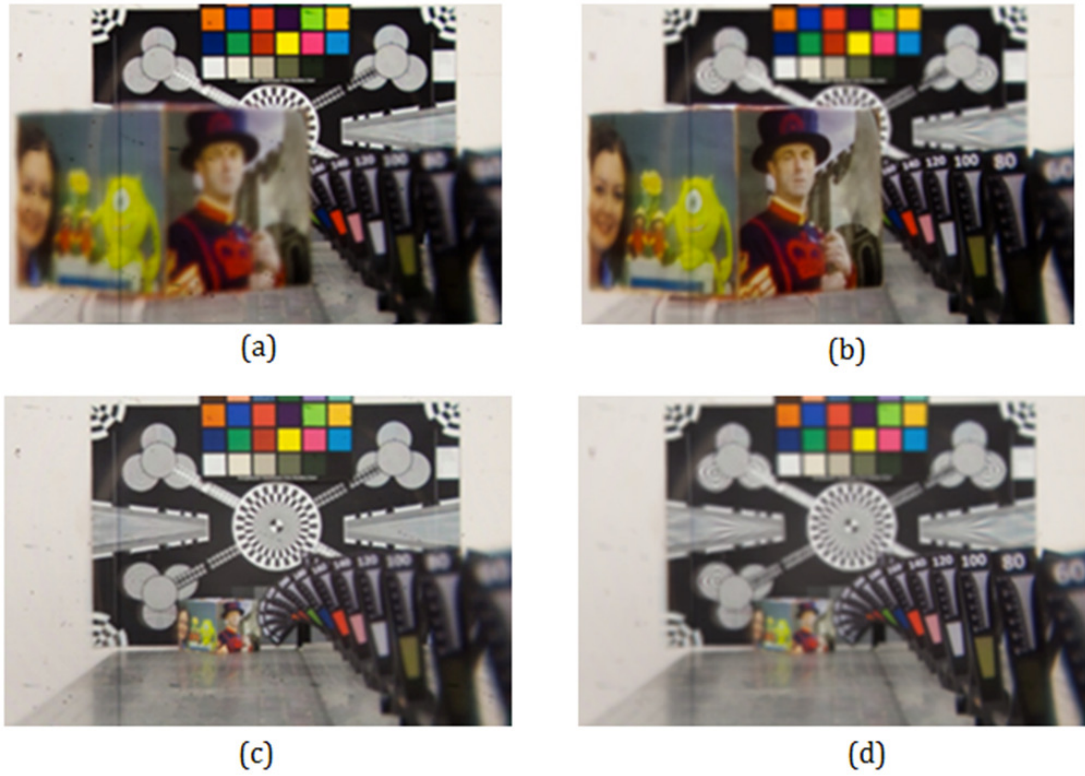


Fig 3. 24 : In (a) all-in-focus image is extracted using different planes with window size 20 by 20 pixels. (b) is the virtual depth map from the microlenses to the main lens with the given virtual distant the real depth can be calculated in relation to the main.

### 3.9 Conclusion

In this Chapter, a novel approach was introduced that effectively refocuses low-resolution orthographic images to form a high-resolution image. A new pixel interpolation approach was introduced to improve the visual quality of the final image. As a result, the final image looks more like a natural photography image without artifacts. A set of different depth planes were extracted using the above refocusing algorithm, where Michelson contrast algorithm was used on all the depth plane images to estimate the contrast of the refocusing points. The extraction of the all-in-focus image was experimentally demonstrated. Depth information of the 3D object was also extracted from the focused points. Computational experiments were carried out to prove the enhancement on the resolution of the final image using the VP method and also to improve visual quality using new interpolation approach with refocusing. The experiments were performed on both, UH3DI and OH3DI, resulting in a successful outcome of an improved final image. The new all-in-focus, with virtual depth information algorithm, was also successful in extracting the all-in-focus image with exceptional depth information. The effect of window sizes was also addressed in generating both the all-in-focus image and virtual depth

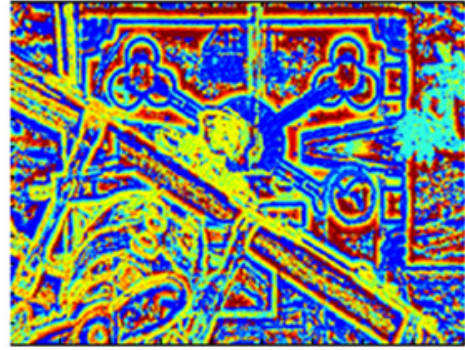
information. Finally, the possibility of the proposed contrast based location coordinated extraction method was confirmed.



**Fig 3. 25 :** In (a) 7 by 7 VPs to focused at the background. (b) Focused at the distant 100mm. (c) using 5 by 5 VPs focused at background. (d) Focused at 1 meter from the microlens array.

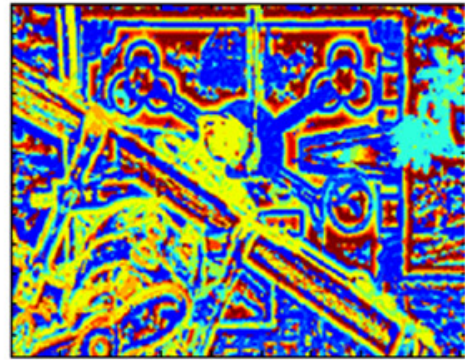


All-in-focus

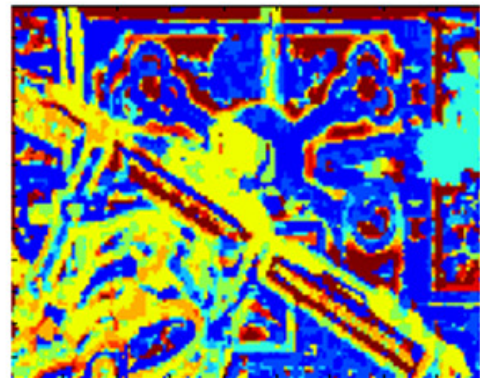


Virtual Depth

Window size 3: note the all-in-focus image contains blur regions on areas where it should have been 'in focuses. The blur regions were extracted with highest contrast as a it also effected the of virtual depth information.



Widow size 5: the blurring region is reduced in compare to window size 3 was the contract is more effective with bigger window size.



Window size 11: An acceptable window size needs to be selected for more promising results with particular scene.

Fig 3. 26 : Result of All-in-focus with disparity

### **3.10 References**

- [1] K. Fife, A. El Gamal, and H. Wong, "A Multi-Aperture Image Sensor With 0.7 Pixels in 0.11 CMOS Technology," *Solid-State Circuits, IEEE*, vol. 43, no. 12, pp. 2990–3005, 2008
- [2] T. Georgiev and A. Lumsdaine, "Reducing Plenoptic Camera Artifacts," *Comput. Graph. Forum*, vol. 29, no. 6, pp. 1955–1968, Sep. 2010.
- [3] T. Georgiev and A. Lumsdaine, "Resolution in plenoptic cameras," *Opt. Soc. Am.*, pp. 1955–1968, 2009.
- [4] T. Georgiev, "Superresolution with the focused plenoptic camera," *IS&T/SPIE Electron. Imaging*, p. 78730X–78730X, 2011.
- [5] T. Georgiev and A. Lumsdaine, "Focused plenoptic camera and rendering," *J. ...*, vol. 12, no. 2, pp. 1–28, 2010.
- [6] A. Lumsdaine and T. Georgiev, "The focused plenoptic camera," *Comput. Photogr.*, pp. 1–8, 2009.
- [7] P. van Walree, "Distortion," *Photographic optics*, 2012. [Online]. Available: <http://toothwalker.org/optics/distortion.html>. [Accessed: 24-Jun-2013].
- [8] A. Lumsdaine and T. Georgiev, "Full resolution lightfield rendering," *Indiana Univ. Adobe Syst. Tech. Rep*, no. January, pp. 1–12, 2008.
- [9] R. Ng, "Digital light field photography," 2006.
- [10] R. Ng, M. Levoy, M. Brédif, and G. Duval, "Light field photography with a hand-held plenoptic camera," *Comput. Sci. ...*, vol. 2, no. 11, pp. 1–11, 2005.
- [11] S. Wanner, J. Fehr, and B. Jähne, "Generating EPI representations of 4D light fields with a single lens focused plenoptic camera," *Adv. Vis. Comput.*, vol. 2011, no. 1, pp. 90–101, 2011.
- [12] D. E. Mitchell, R. D. Freeman, and G. Westheimer, "Effect of orientation on the modulation sensitivity for interference fringes on the retina," *JOSA*, vol. 57, no. 2, pp. 246–249, 1967.
- [13] T. Georgiev and A. Lumsdaine, "Depth of field in plenoptic cameras," *Proc. Eurographics*, vol. 2009, no. 1, 2009.
- [14] S. A. Klein, T. Carney, L. Barghout-Stein, and C. W. Tyler, "Seven models of masking," in *Electronic Imaging'97*, 1997, pp. 13–24.

- [15] V. Raja and M. Murthy, "A Study on Subjective Evaluation of Printed Launch Vehicle Lift-Off View Colour Photographs and Pattern Quantification of Perceived Image Quality Attributes," *ijesrt.com*, vol. 3, no. 3, pp. 1098–1105, 2014.
- [16] E. Peli, "Contrast in complex images.," *J. Opt. Soc. Am. A.*, vol. 7, no. 10, pp. 2032–40, Oct. 1990.
- [17] M. Laikin, "Wide angle lens systems," in *1980 International Lens Design Conference*, 1980, pp. 530–533.

## **4. Chapter Four**

### **Digital Refocusing Based on SI**

This chapter presents a novel way of applying the up-sampling, shift, and integration with full resolution rendering to reduce artifacts and to improve the image quality and resolution. It works by extracting a patch (sub-image) under each microlens and it improves resolution of the view by total microlens times the patch size [1]; This can be compared to what was presented in chapter 3 where only one pixel was extracted under each microlens, forming the final viewpoint at total resolution of total microlenses along vertical and horizontal directions. The shift and integration of views happens by sub-pixels allowing an accurate intersection of different rays coming from individual views. The views were extracted from a holographic 3D image by taking a group of pixels under each microlens as mentioned above, which increases the resolution of individual views. Also, the sub-pixel adjustment improves the visual quality and resolution in comparison with the method discussed in chapter 3 as well as

other methods mentioned in the literature. It is important to explore the effect of basic full resolution rendering [1] as well as full resolution rendering with blending [2]. It is also reported in [3] that artifacts on the final image were resolved using the depth-based rendering. However, the depth-based rendering is not used here, as it requires depth map, which is rather a time-consuming process.

#### 4.1 Full Resolution Basic Rendering

The basic full resolution rendering method takes the full use of the resolution [1]. However, it introduces artifacts on the final image. In chapter 3, the viewpoint images (VPs) were defined as  $VP_{i,j(n,m)}$ , where  $n, m$  are the number of Element Images (EI) along vertical and horizontal directions with  $i, j$  being the coordinates of EI. It is worth mentioning that  $i, j$  and  $n, m$  are the recorded position and direction, respectively. The resolution of the resulting image is higher compared to the viewpoint image; this is because the sampling of full resolution approach uses sub-images (SIs) instead of VP.

The VP images are obtained by extracting one pixel from every EI that makes the resolution equal to the number of EI along vertical and horizontal directions. On the other hand, full resolution rendering is achieved by extracting sub-image with  $SI \times SI$  pixels. This makes the overall resolution to be  $SI$  times the viewpoint resolution. For example, let the  $SI = 2$ , by enabling to extract two pixels from each EI making the resolution twice as higher as the VP resolution. The size of  $SI$  is dependent on the final depth plane; different size of  $SI$  will return a different depth plane 'in focus' [3]. The EI is defined as  $EI_{n,m(j,i)}$  to render full resolution image. We present the algorithm in Fig 4. 1.



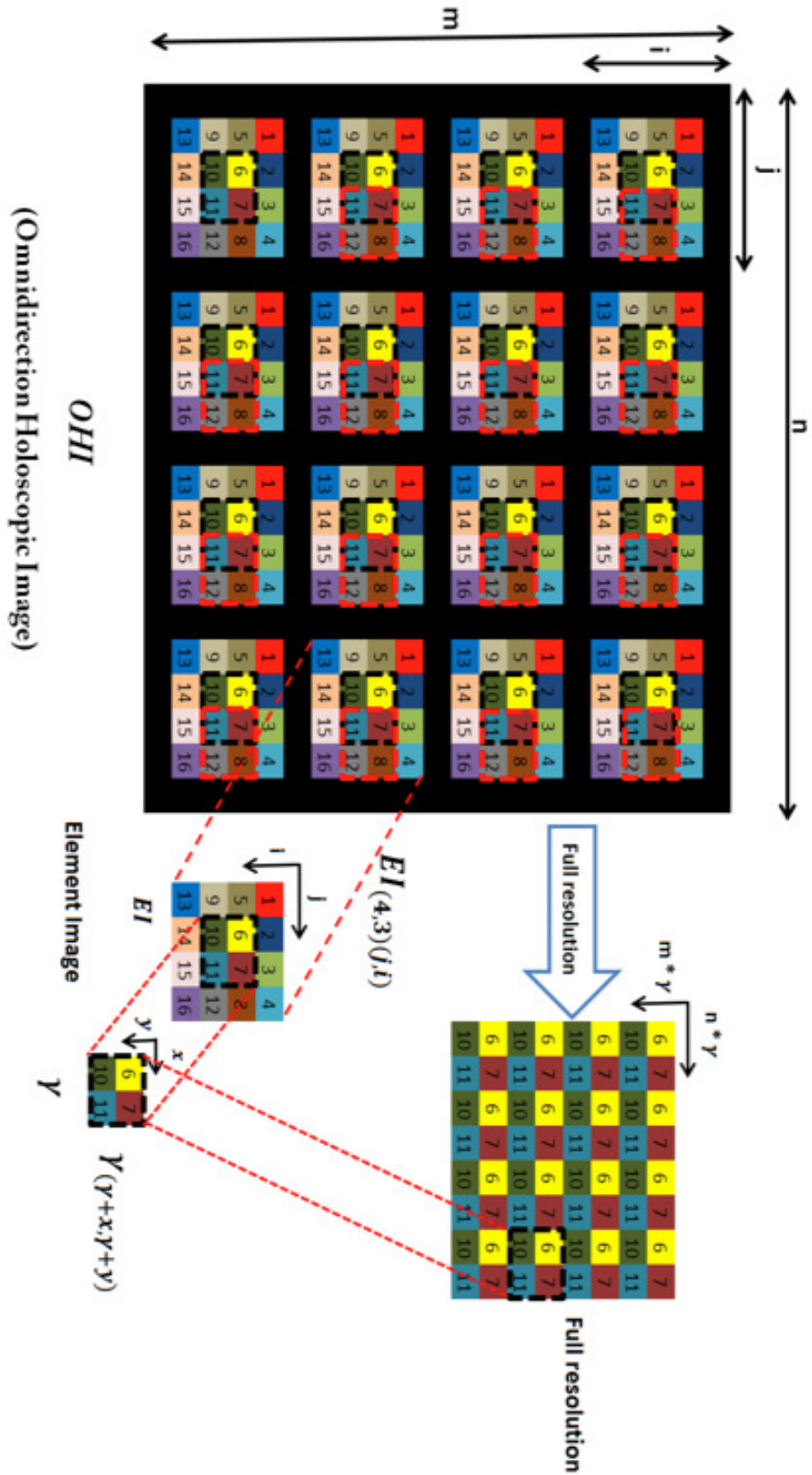


Fig 4. 1 : The full resolution rendering algorithm resulting image at resolution of  $n \cdot SI$  by  $m \cdot SI$

#### 4.1.1 Full Resolution Rendering Algorithm:

```

n=1: EI : length(OHI,2)
m=1: EI : length(OHI,1)
output_Full_resolution=(1:n*  $\gamma$ ,1:m*  $\gamma$ )
n $\gamma$  = 1:  $\gamma$  : n*  $\gamma$ 
m $\gamma$  = 1:  $\gamma$  : m*  $\gamma$ 
    for Row= 1: length(n $\gamma$ )
        InRow(n $\gamma$ (Row): n $\gamma$ (Row+1)-1)=(n(Row):n(Row)+  $\gamma$ )
    end
    for Col=1:length(m $\gamma$ )
        InCol(n $\gamma$ (Col): m $\gamma$ (Col+1)-1)=(m(Col):m(Col)+  $\gamma$ )
    End
Output_Full_resolution(1:length(inRow), 1:length(inCol))=OHI(inRow,inCol)

```

The above pseudo code illustrates how the full resolution rendering works, as graphically shown in Fig 4. 1. The Row and Col variables are row and column of the view size, where the InRow and InCol are row and column of the index locations of view from OHI, which is extracted. The final image in the basic full resolution rendering returns a strong artifact. This is because the size of SI depends on the scene depth. In other words, different depth plane within the scene requires different size of SI. Unfortunately, the basic full resolution rendering uses fixed size SIs; as a result, artifacts become more apparent on the edges of the EI where the size of SI doesn't match the depth plane of the scene. Refocusing is achieved by specifying the size of the SI, but the artifacts remain strong in un-matching region of the scene depth plane. Within this process two artifacts arised - as explained earlier in Fig 3. 12. First is due to the size of SI being too big for a particular object within the scene. As a result, repetitive portion of the same objects are extracted resulting in 'repetitive' artifacts. The second is due to the size SI being too small for that same object within the scene, resulting in 'pixelated' artifacts. Therefore, an up-sampling, shift, and integration is introduced in this method to effectively converge the angular light rays to a particular depth plane making the refocusing look more natural and remove the articles on the final image.

#### 4.2 Up-sampling, Shift and Integration with Full Resolution Method

The up-sampling, shift, and integration were discussed earlier in chapter 3. In this chapter, however, it has been fused with full resolution method, which

reduces the artifacts in the final refocused images. Different prospective views of full resolution are extracted, shifted and then integrated by converging the same light rays across different SIs to give natural blur on the artifacts region. This is because the same points in different SIs are correctly intersected which return 'in focus' points at that particular image (depth) plane; whereas, the mismatch intersection of a point will return a blur region on the image.

Different views are extracted by shifting the axis of  $j$  and  $i$  under EI with same size SI throughout other EI, shown graphically in Fig 4. 2. Those different views are up-sampled by the number of rendered images that are used in shift and integration processing to remove the artifacts in the out-of-focused regions. The out-of-focus regions in this method appear to be pixelated or repetitive. The up-sampling is performed before shift and integration; as a result, it enables sub pixel integration of the same spatial point across different views to enhance the resolution of the refocused image by facilitating more pixels to represent the same point.

It is important to point out the characteristics of full resolution rendering where the refocusing is achieved through the size of SI, as the size determines the image plane in-focus, whereas the out-of-focus regions emerge with artifacts. Therefore, up-sampling, shift, and integration of views compensate the artifacts infected regions by appearing blurred, as the in-focus region will remain in-focus. This is due to different views that have the same disparity value, where the image plane is in-focus and it remains in-focus by integrating the same spatial point with its different views. In this method, the value of shift is not dependent on the image plane, but the image plane is dependent on the size of SI, as mentioned above.

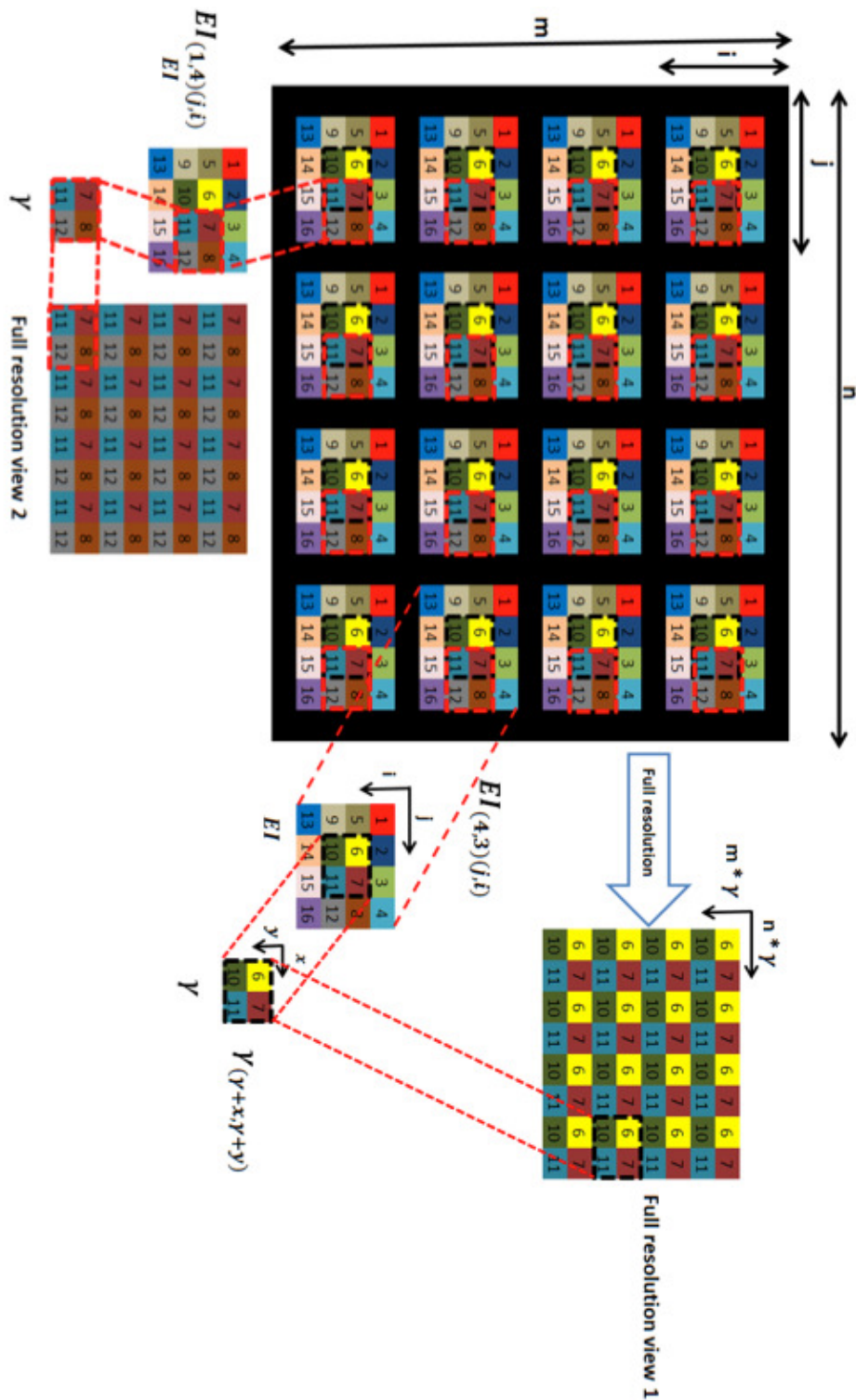


Fig 4. 2 : The full resolution rendering algorithm is shown with two different views by shifting the axis of  $j$  by 1 which extracts a full resolution view 2 with the same size of SI.

The views are stacked adjacently in horizontal and top-to-bottom in vertical directions from one another, forming 4D stack of images  $S_{(xy)(kp)}$  where  $(x, y)$  are the coordinates of the image views with  $(k, p)$  being the pixel coordinate of each view as shown in Fig 4. 3. Furthermore, the selected numbers of views are up-sampled to the number of views used, i.e. if two by two views are selected in shift and integration process, then each view is up-sampled by two in horizontal and vertical directions. Finally, the shift and integration of views are performed on the up-sampled view, which is described in eq 4.1.

$$f_{kp} = \sum_{sx=1}^{NoViews} \sum_{sy=1}^{NoViews} S_{((x+NoViews)-sx, (y+NoViews)-sy), (k+sx, p+sy)} \quad \text{eq (4.1)}$$

where  $f_{kp}$  is the result of the integration of different views along vertical and horizontal direction and  $NoViews$  is the number of views used;  $k$  and  $p$  are the coordinates of the pixels inside each view, which are up-sampled by the  $NoViews$  to obtain the final refocused image with minimum artifacts. The final image resolution is equal to  $SI$  times by the up-sampling, where the up-sampled pixels are replaced with new pixels after shift and integration of different views.

**Example:** If  $SI = 3$ ,  $n=193$ ,  $m=129$ ,  $k = (SI*n)$ ,  $p=(SI*m)$ ,  $y=4$ ,  $x=4$ , and  $NoViews = 3$ , where the views resolution equals to  $(k * NoView)$  by  $(p * NoView)$ . Note that at this point the view's pixels are up-sampled by 3 times. These pixels are later replaced with new values in the shift and integration process, creating the final image with higher resolution than the views. This means that by representing one pixel with 3 x 3 pixels, it gives the final image more richer looking.

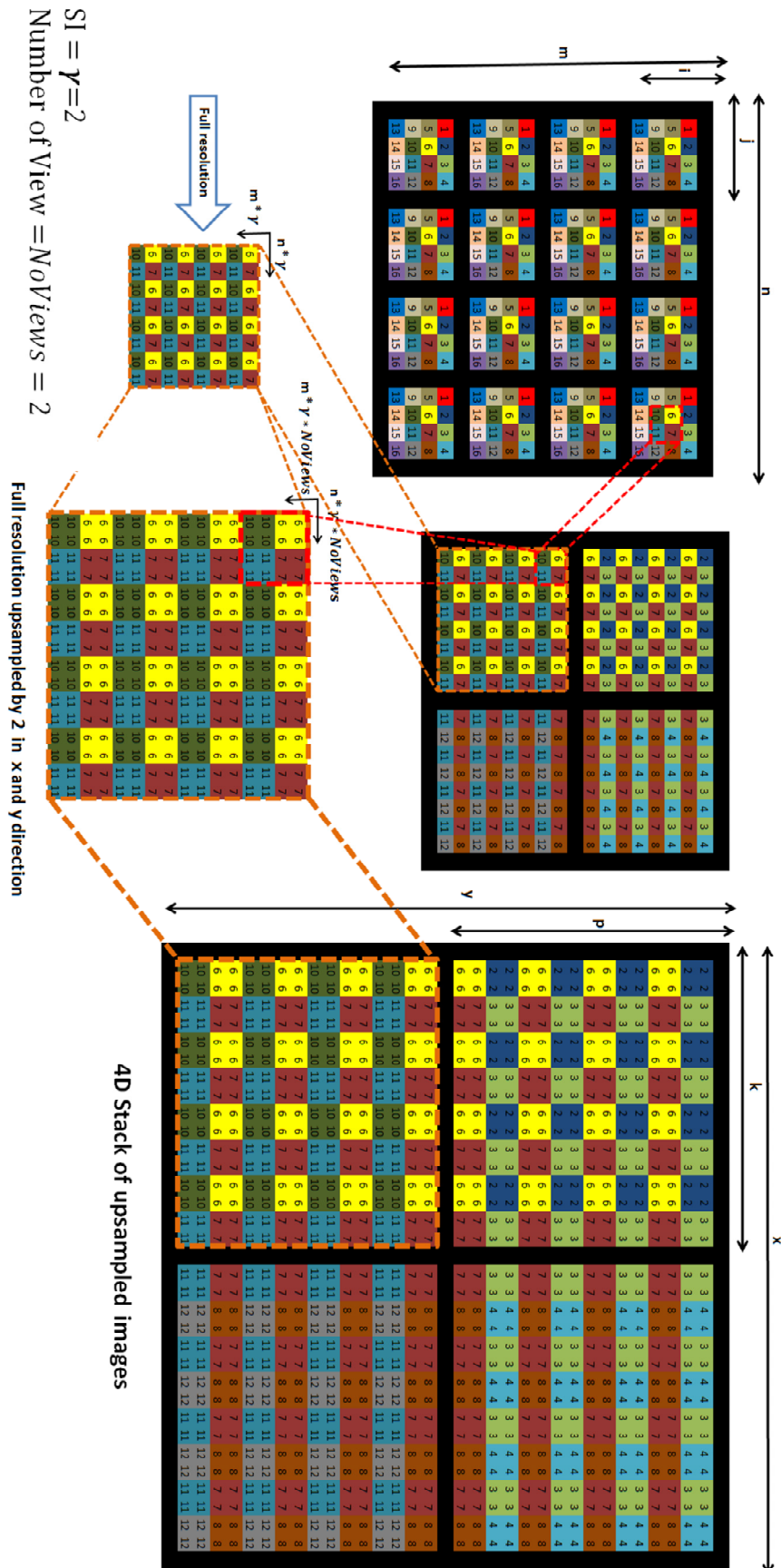


Fig 4. 3 : Graphically illustration of 2 by 2 full resolution rendering with up-sampling of 2 in x and y direction.

### **4.3 Experimental Results and Observations**

To test the proposed method in comparison with the VP pixel manipulation technique, a scene that has four objects placed at different distances. The scene setup was described in detail earlier in chapter 3. The given test chart is placed in the background to check the quality and resolution of the final refocused images. This also gives a clearer and easier way of comparing the quality of the image with other methods that are clearly taken into consideration in this section. The optical system used to capture the images is explained in details in [3] with microlens pitch size ( $B$ ), focal length ( $F$ ), and pixel pitch ( $\Psi$ ) being  $0.09mm$ ,  $1.0mm$  and  $0.0031mm$ , respectively.

Multiple 2D views of the scene are extracted using high resolution rendering method to examine quality of the views. Each view's quality deteriorates as we move closer to the edge of microlens; this is due to the poor quality of microlenses itself. Considering the poor views in the process will affect the quality and colour of the final image. Thus, a view pickup position is developed to visualise and make it more convenient to navigate through the views, avoiding the corners of the EI in extracting the multiple views of the scene, as shown in Fig 4. 4.

Before multiple views are extracted, it is important to determine the size of SI. This is because the size defines the image planes [2]. The size of the SI is set to 7 by 7 under every EI, which gets the background (test chart) 'in-focus', to affectively compare the quality of that image plane with others. Single 2D high-resolution view at  $1344 \times 903$  pixels is extracted as shown in Fig 4. 5a.

After multiple views are extracted with 7 by 7 SI, up-sampling, shift, and integration of views are performed to defocus the artifact infected regions, where the adjustment of blurring is directly related to the amount of views that are used in shift and integration. Also up-sampling is taken into consideration before the shift and integration process takes place, which largely enhances the quality and resolution of the final image—by generating extra pixels in representing the same point with multiple pixels from different views.

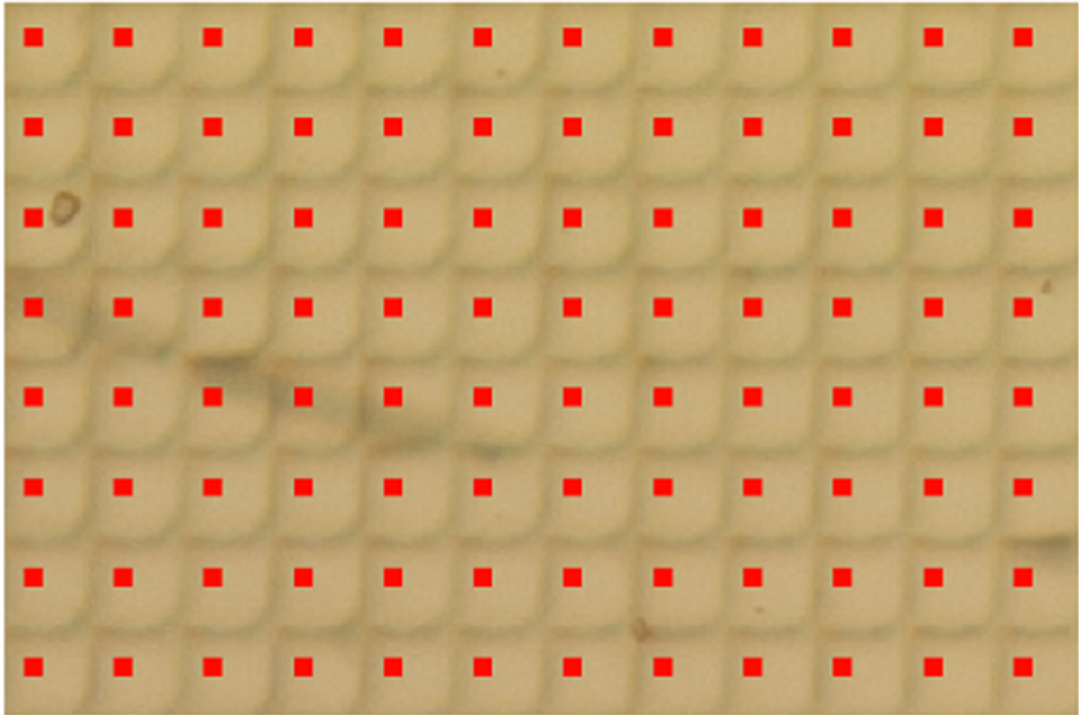


Fig 4. 4 : A small zoomed portion of OH3DI and red squares shows the view pickup position under each EI



Fig 4. 5a : Signal 2D view at 1344 x 903 resolutions using 7 by 7 SI size under each EI having the test chart 'in-focused'.

The shift and integration on high-resolution views shows an increase in resolution, (see Fig 4. 6) for the same OH3DI that was used in chapter 3. VP pixel manipulation technique in chapter 3 and high-resolution up-sampling, Chapter 4 – Digital Refocusing Based on SI



shift, and integration of views are compared, as both are focused at the test chart to see the visual quality. Fig 4. 6(b) shows a magnified region of the test chart to demonstrate the resolution enrichment by the clearly visible circular lines. The obtained results using the VP pixel manipulation technique are shown in Fig 4. 7. A magnified section of the background shows that the method fails to rebuild the circles properly as compared to the results shown in Fig 4. 6(b). This is due to the fact that using VP images are low resolution, to start with in this algorithm.

#### 4.4 Sub-pixel Adjustment Technique (SPA)

High resolution with up-sampling, shift, and integration technique still produces artifacts. For example, putting the focus at distant object results in visible artifacts in the close objects, as seen in Fig 4. 6(c). On the other hand, the VP approach doesn't suffer from such artifacts but suffers from poor resolution quality, as shown in Fig 4. 7(c). The arising artifacts are due to a constant size of SIs throughout rendering. As real scenes are not at constant depths, mismatch exists between different SI sizes. To eradicate this artifacts problem when taking such cases, a modification to the eq 4.1 is needed by introducing a parameter delta ( $\delta$ ). This delta in eq 4.2 acts as a SPA to blur out the artifacts regions in the final image. The SPA is able to intersect light rays coming from sub pixels, which allows more control over the integration of pixels with other views' pixels.

eq (4.2)

$$f_{kp} = \sum_{sx=1}^{NoViews} \sum_{sy=1}^{NoViews} S_{((x+NoViews)-sx, (y+NoViews)-sy), (k+(\delta*(sx-1)), p(\delta*(sy-1)))}$$

The value of  $\delta$  is defined by taking the camera mode into consideration as there are two modes Keplerian and Galilean [4]. Both modes are explained in chapter 2 and have been considered in the process. In Keplerian mode SPA is set to positive as the main image is formed in front of the microlenses. This will reduce the artifacts to its minimum by setting  $\delta$  to positive value by blurring the artifacts regions in the final image. The amount of blurring depends on the SPA that is related to the value of the artifacts in the image to which the value of  $\delta$  is assigned. If the effect of the artifacts is very strong on the image, then  $\delta$  is set to higher value to minimise the artifacts. Whereas, in Galilean mode, the focus of the main lens is behind the microlenses that create a virtual image; this sets the value of  $\delta$  to negative to compensate the artifacts region down to its minimum. This process is illustrated graphically in Fig 4. 5b with two high resolution views. Note that the delta is set to -1 and each pixel is up-sampled by 3x3 in

high resolution views before the shift and integration process. Therefore, SPA is considered based on how strong the artifact is, whereas in this case assuming the artifacts is not as strong by setting delta to -1, it changes to 2. This is because views are up-sampled by 3 in x and y directions and integrating the pixel by sub-pixel of 0.66666; therefore, it is set to -1 in shift and integration process where the delta value changes by the number of views that is up-sampled in the process, namely 2.

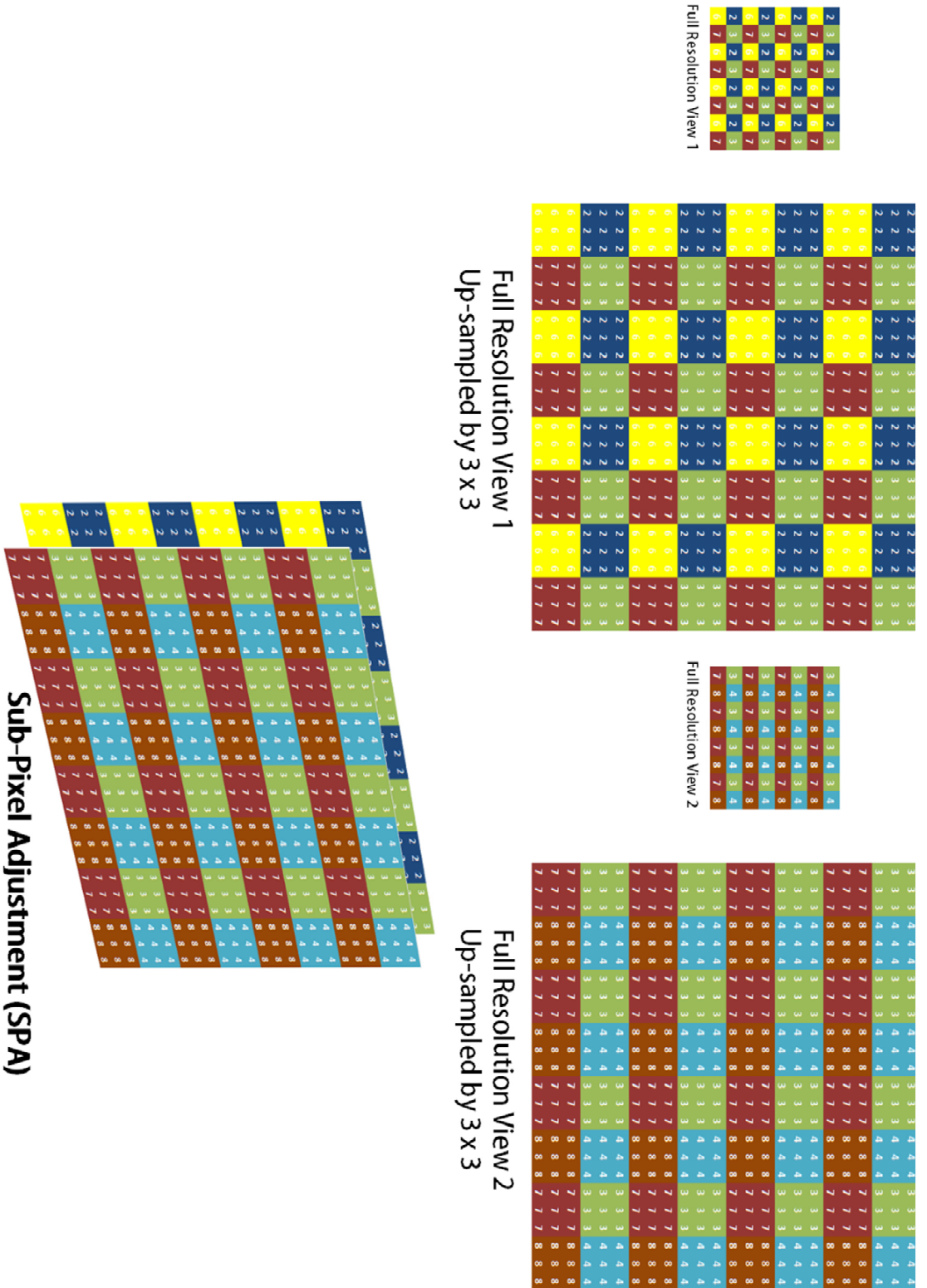


Fig 4. 5b : Graphical illustration of SPA

## 4.5 Experimental Results and Observations

The SPA is tested on the same set of images to compare the effect with the other methods. The results of this technique are presented against those of the others to show the improvement of the artifacts region in the final image with proper distribution of colours. This is achieved with proper selection of SI size in the process and also the value of delta to reduce the artifacts down to its minimum. Where the object is too close to the camera and the focus is on distant objects, it has to be taken into consideration, as mentioned above.

### 4.5.1 Comparison of Subjective Image Quality

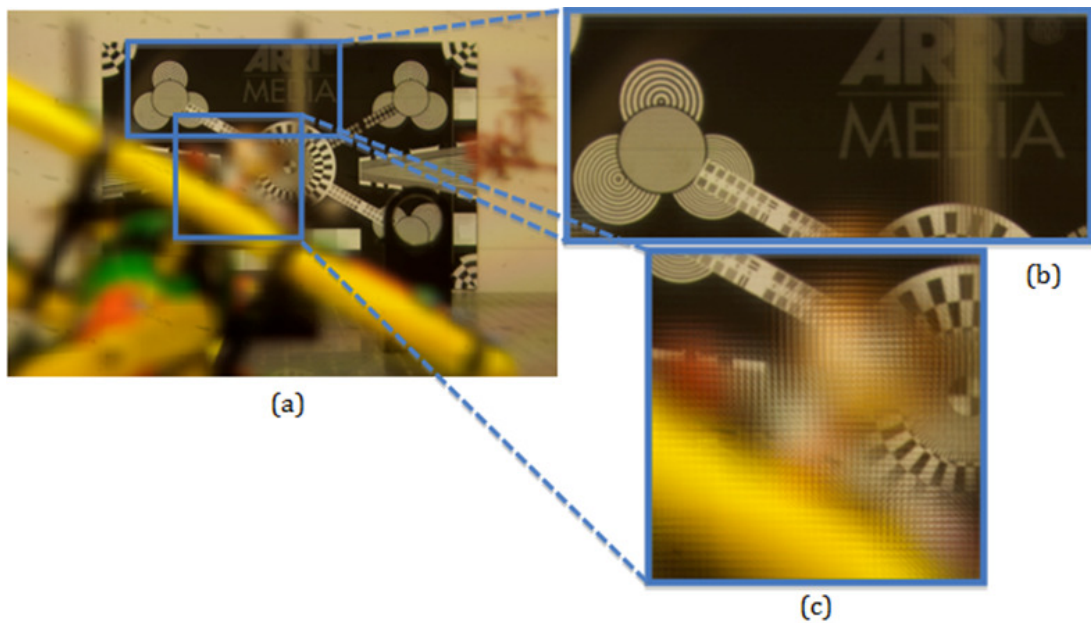


Fig 4. 6 : (a) The high resolution rendering with up-sampling, shift, and integration. (b) magnified region of the test chart 'in-focus' and (c) illustrate the foreground object.

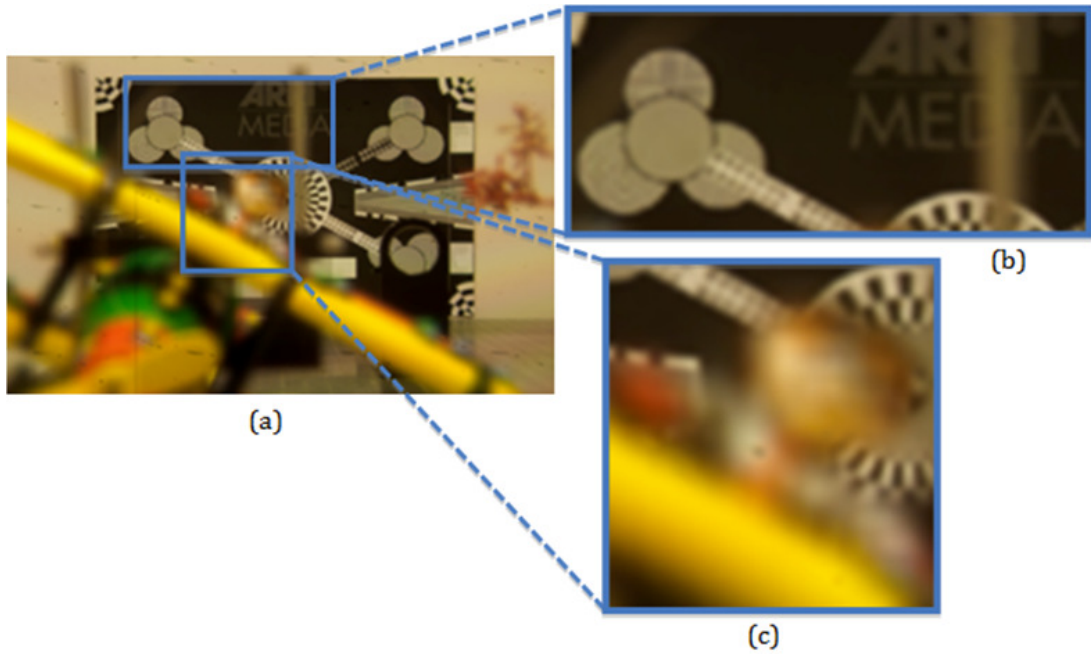


Fig 4. 6 : Up-sampling, shift and integration refocusing using 7 by 7 VPs: (a) shows the magnified part of ARRI Media test chart. (b) Is the foreground looks, as it should be.

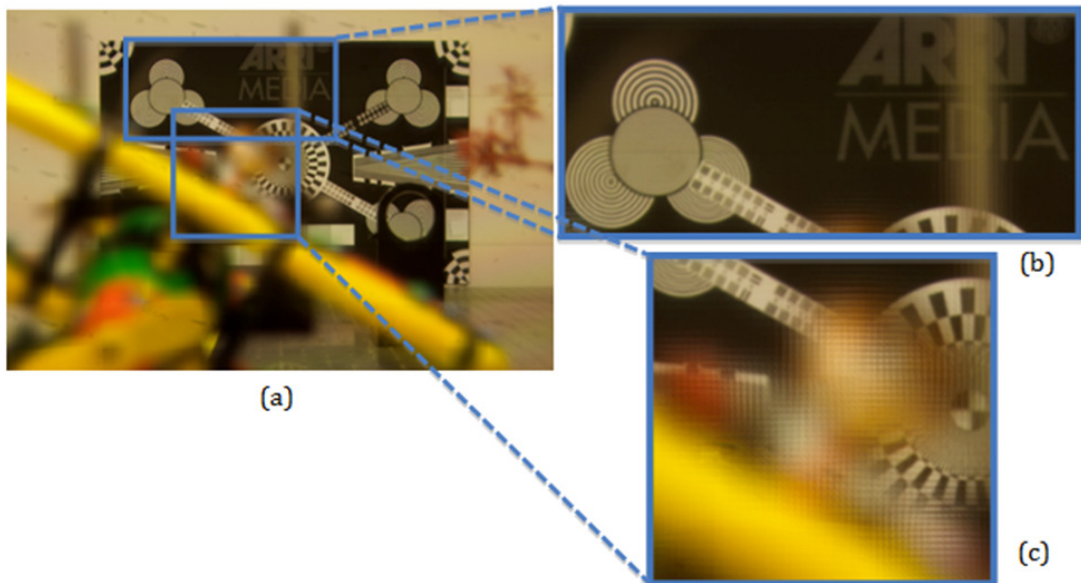


Fig 4. 7 : The SPA on high resolution rendering with up-sampling, shift and integration (a). (b) Magnified region of the test chart 'in-focus' and (c) illustrate the foreground object.  $SI = 7$ ,  $\delta = 6$ , number high resolution views used 10.

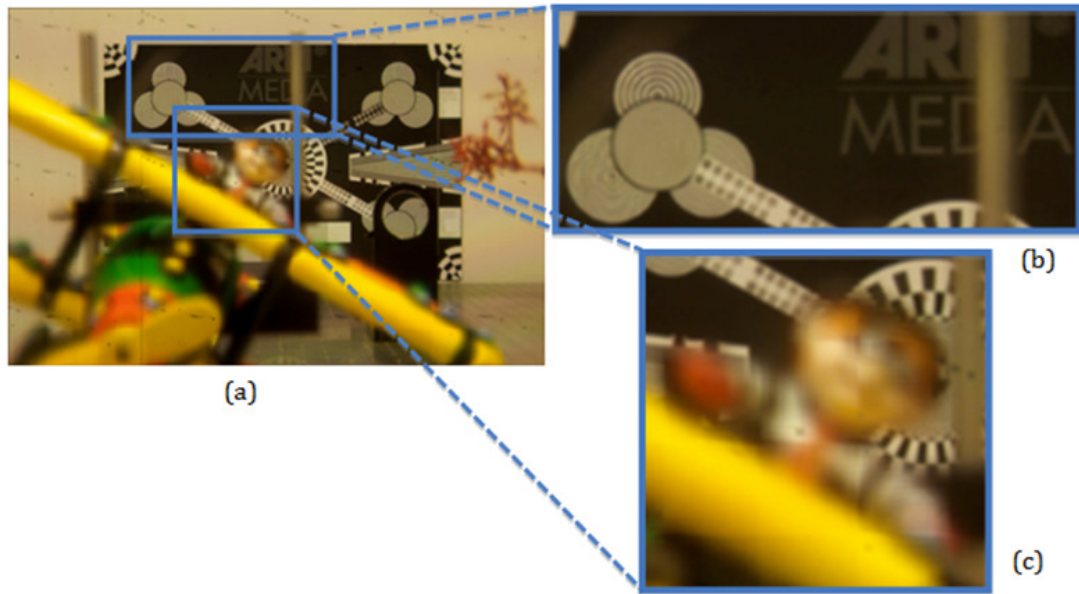


Fig 4. 8 : The SPA on high resolution rendering with up-sampling, shift and integration (a). (b) Magnified region of the test chart 'in-focus' and (c) illustrate the foreground object.  $SI = 2$ ,  $\delta = 2$ , number high resolution views used 7.

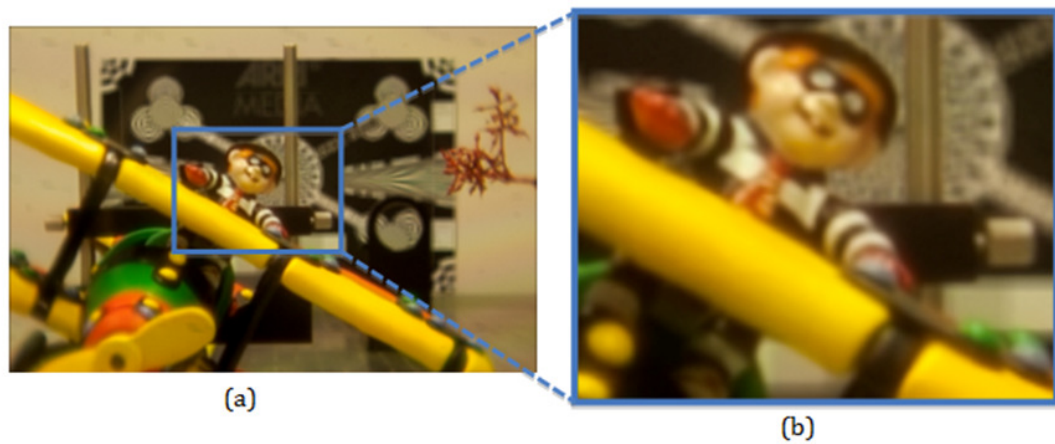


Fig 4. 9 : The SPA on high resolution rendering with up-sampling, shift and integration (a). (b) Illustrate the foreground object.  $SI = 2$ ,  $\delta = 7$ , number high resolution views used 7.

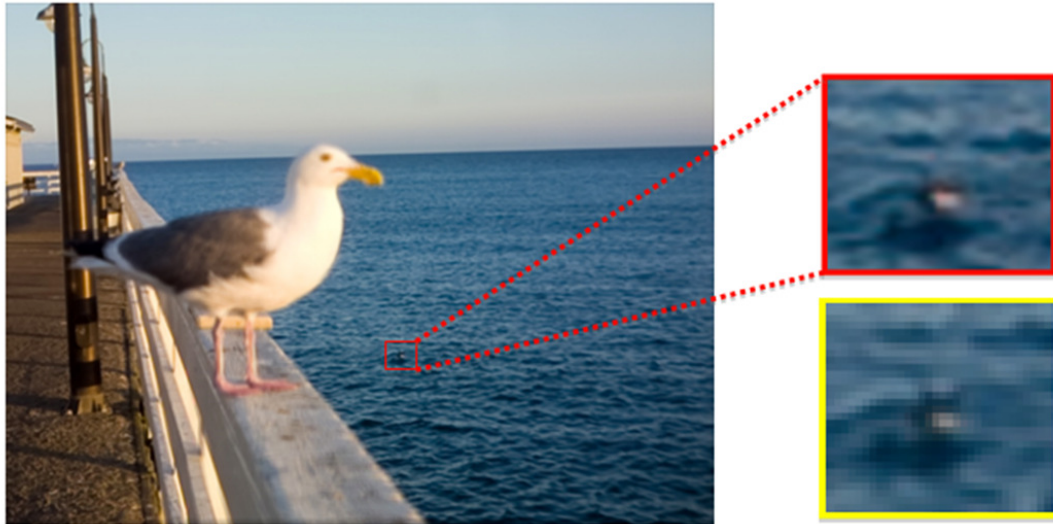


Fig 4. 10 : Rendered image using up-sampling, shift and integration with high resolution views, focused at background with SI = 7 by 7, Size of EI = 74 by 74. Red square is the result of up-sampling, shift and integration and the yellow square is rendered high resolution with Blending [2].

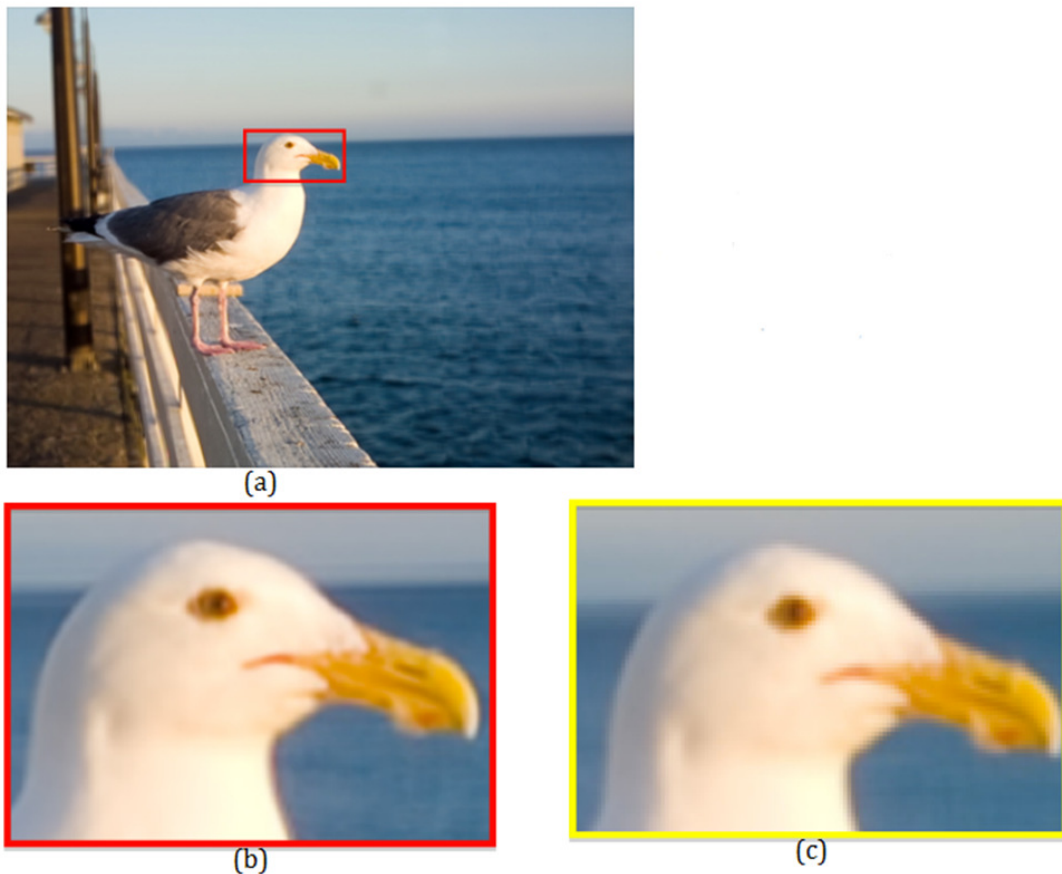


Fig 4. 11 : (a) Rendered focused at seagull with SI = 9 by 9, Size of EI = 74 by 74. (b) Red square is the result of up-sampling, shift and integration. (c) Yellow square result is rendered with high resolution with Blending [2].



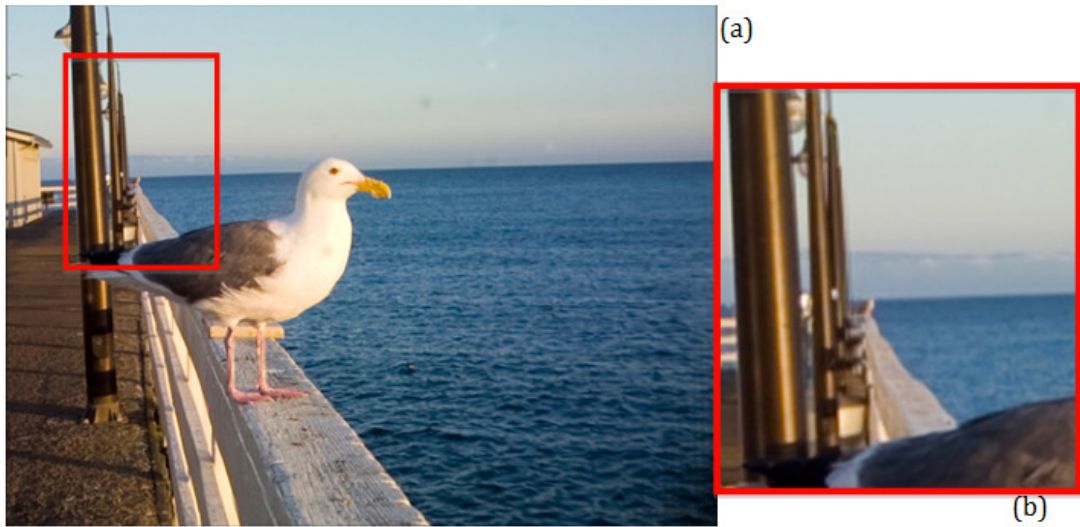


Fig 4. 12 : (a) Rendered image using SPA. (b) Magnify red square region of the rendered image (a).

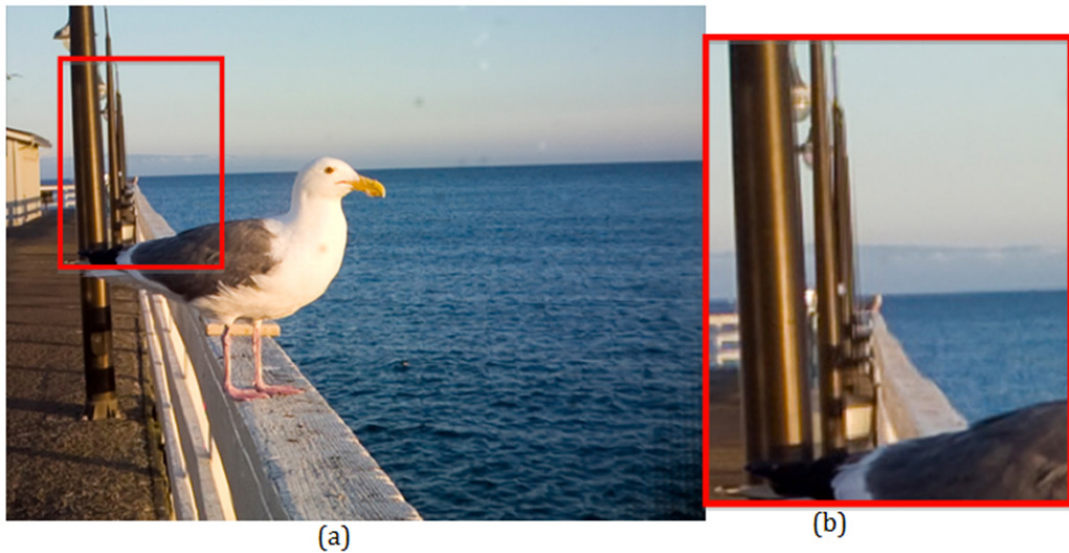


Fig 4. 13 : (a) Rendered image without delta. (b) Magnify red square region of the rendered image (a) containing artifacts.

#### 4.5.2 Analysis and Discussion

Fig 4. 8 shows an improvement on the image with proper distribution of colour and blur, with reduced artifacts on infected regions. This also blurs other regions on the final image, though not noticeable compared to the artifacts on infected region that is blurred out. Note that blurring on the final image with SPA has not affected the lines on the test chart, but it has reduced the artifacts to give a natural photographic look.

As the artifacts remain challenging, if needed to be completely removed from the image, the size of SI has to be reduced to give more natural look to the final image. This will affect the resolution of the image. The artifacts will be completely reduced from the final image, as shown in Fig 4. 9. This demonstrates that there is a trade-off between resolution and removing the artifacts.

Moreover, the Todor's image from adobe called 'Seagull' is rendered with up-sampling, shift, and integration to see the quality and resolution of the final image when they are refocused and also rendered with SPA, where the results are compared. It's worth mentioning that Todor's plenoptic camera design is modified to minimise the artifacts in rendering processing [3]. This is due to the scene having different magnification at variance depth that causes final rendered image with artifacts. The modified design reduces the magnification by comprising scene across the entire microlenses.

Results show an improvement in the proposed approach compared to the high resolution rendering with blending. In such a modified version, the final results are rendered in high-resolution artifacts free, whereas the SPA in this design can benefit from rendering the all-in-focused image as it can be seen in Fig 4. 12. This is because different scene depths are imaged at different magnifications to the camera sensor, as in the modified version the magnification does not differ much in different scene depths [3]. SPA makes a great impact in rendering all-in-focused images as well as in refocusing. Fig 4. 12 shows an example of 8 by 8 SI with  $\delta=1$  using SPA to generate an all-in-focused image. The choice of SI size (SS) is determined by SI background (BG) size plus SI foreground (FG) size divided by 2 (see eq 4.3), and delta ( $\delta$ ) is set to positive value, as camera is in Keplerian mode. The value of delta is set to 9 as defined in eq 4.4. To get the background 'in-focus' the size SI of a BG is set to 7 and that of the FG to 9 in getting the foreground 'in-focused', as can be seen in Fig 4. 11 and Fig 4. 12, respectively. The number of views used in this process is equal to the same number as the size of SI that is calculated from the eq 4.3. This equation is used to work out what size of SI is needed for extraction of

views. Then the outcome of this equation will be added in eq(4.4) to work out the  $\delta$  value.

$$SS = (BG + FG)/2 \quad \text{eq (4.3)}$$

$$\delta = ((FG - BG)/2) + SS$$

$$\delta = FG \quad \text{eq (4.4)}$$

The same parameters are set in comparing the outcome results without  $\delta$  in SPA. This causes the final image with artifacts that can be seen with the naked eyes as demonstrated in Fig 4. 14. Therefore, SPA with delta in up-sampling, shift, and integration process is clearly justified by the improvements in the final images through visibly removing the artifacts in the modified optical design by compressing the scene depth. It also works better without compressing the scene depth as demonstrated above with minimum artifacts in the refocused images.



(a) SI = 3,  $\delta = 4$ , NoViews=12 and focused at 300cm to 290cm



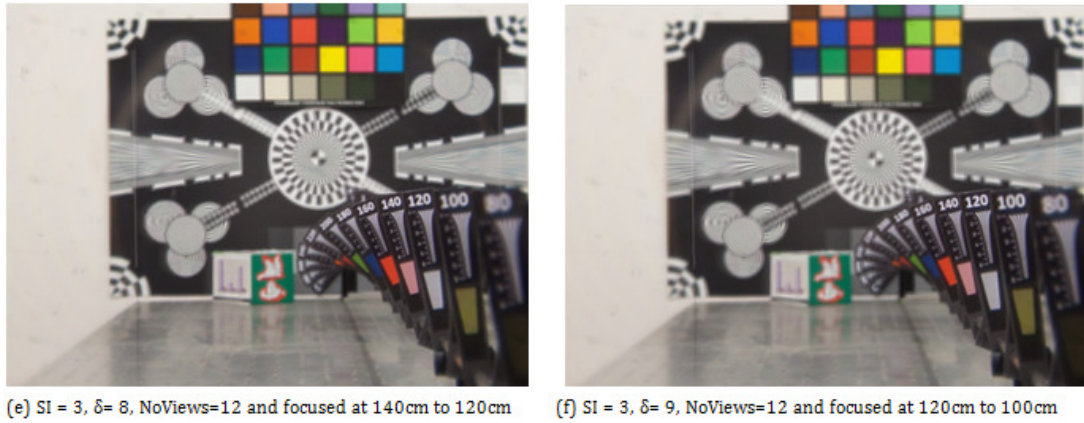
(b) SI = 3,  $\delta = 5$ , NoViews=12 and focused at 240cm to 200cm



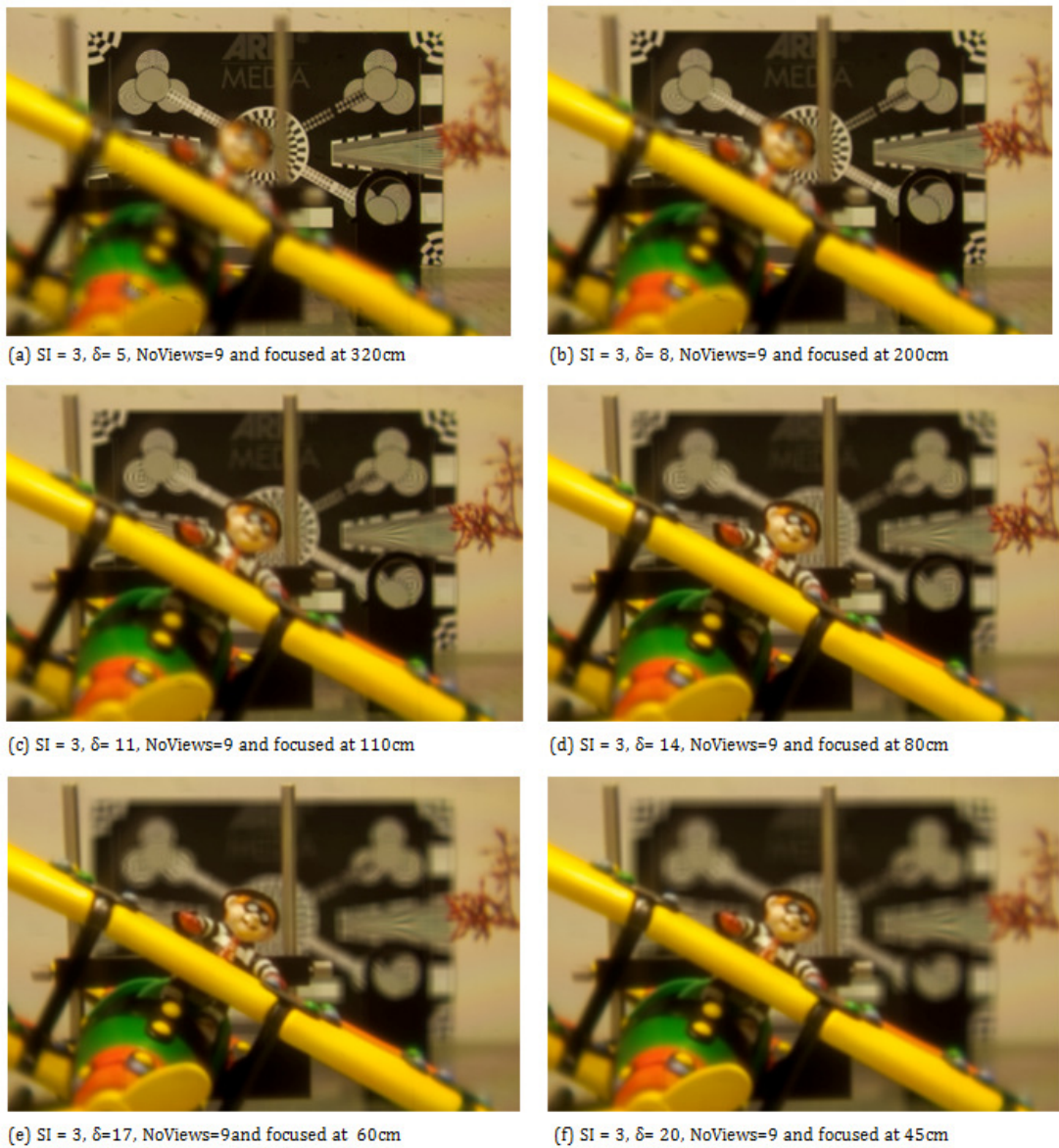
(c) SI = 3,  $\delta = 6$ , NoViews=12 and focused at 200cm to 160cm



(d) SI = 3,  $\delta = 7$ , NoViews=12 and focused at 160cm to 140cm



**Fig 4. 15 :** demonstrate refocusing with constant SI size in SPA by changing the value of delta in the process.



**Fig 4. 14 :** illustrates in changing the depth plane from background in (a) to foreground in (f).

## **4.6 Conclusion**

In this chapter, a novel method of refocusing on high resolution views using the up-sampling, shift, and integration, as well as SPA were explored and discussed. Section 4.2 concentrated on applying the up-sampling, shift and integration on high resolution views by comparing the results with the existing methods. Experiments proved the effectiveness of the methods in using the same data throughout when comparing the results with the existing methods. Sets of other images were also used to check the quality of the final image. The images proved that the proposed method can achieve higher resolution with acceptable quality compared to the current methods.

To further improve the performance of the method in removing the artifacts down to its minimum, the SPA in section 4.4 was presented by introducing the delta in the equation to work affectively. This is carried out by using SPA in blurring out the artifact regions of the final image, given the value of delta. Experiments have shown a clear improvement in the final refocused image, achieved by the SPA method on several captured OHI. The seagull image from 'Todor' is used in this experiment where the optical elements of the camera are modified to reduce the artifacts in the final image by rendering with blending. Using the Seagull image with SPA returns an all-in-focus image with promising results, whereas the same parameters used with up-sampling, shift, and integration result in the final image being infected with artifact. The results obtained from both methods have improved but still returns unwanted noise in the final image.

## **4.7 References**

- [1] A. Lumsdaine and T. Georgiev, "Full resolution lightfield rendering," Indiana Univ. Adobe Syst. Tech. Rep, no. January, pp. 1–12, 2008.
- [2] T. Georgiev and A. Lumsdaine, "Focused plenoptic camera and rendering," *J. ...*, vol. 12, no. 2, pp. 1–28, 2010.
- [3] T. Georgiev and A. Lumsdaine, "Reducing Plenoptic Camera Artifacts," *Comput. Graph. Forum*, vol. 29, no. 6, pp. 1955–1968, Sep. 2010.
- [4] T. Georgiev and A. Lumsdaine, "Depth of field in plenoptic cameras," *Proc. Eurographics*, vol. 2009, no. 1, 2009.

## **5. Chapter Five**

### **Smart Filters**

This chapter presents a new method that further enhances the image quality generated using up-sampling, shift, and integration process described in chapter 4. This process is known as high-resolution or super-resolution image reconstruction, which is achieved by combining a set of low-resolution noisy blurred images to generate a higher resolution image [1]. The process was first introduced in [2] where the idea was carried out on data obtained through traditional light field camera [3]. Later in [4] new optical design was introduced to reduce the artifacts and also to increase the resolution. The new focused plenoptic design in [4] reduces the artifacts but also decreases the refocusing plane, as this setup is limited to the number of refocusing planes.

The images in focused plenoptic camera are rendered with blending, where the final image results contain artifacts. Therefore, up-sampling, shift and

integration with sub-pixel adjustment introduced in chapter 4 reduce the artifacts on both focused plenoptic and modified focused plenoptic [4]. Based on the results in chapter 4, noises on the final refocused image is observed, where the image features are blurred out with noise on the ‘in-focus’ regions during the process. Therefore, a method referred to as ‘smart filters’, is introduced to enhance the quality of the final refocused images in the process. Removing noise from the final refocused image is challenging since only ‘in-focus’ part of the final refocused image needs filtering to obtain an enhanced artifacts free image.

### 5.1 Flow chart of The Proposed Method

Fig 5. 1 illustrates the steps of smart filter in generating the final enhanced refocused image.

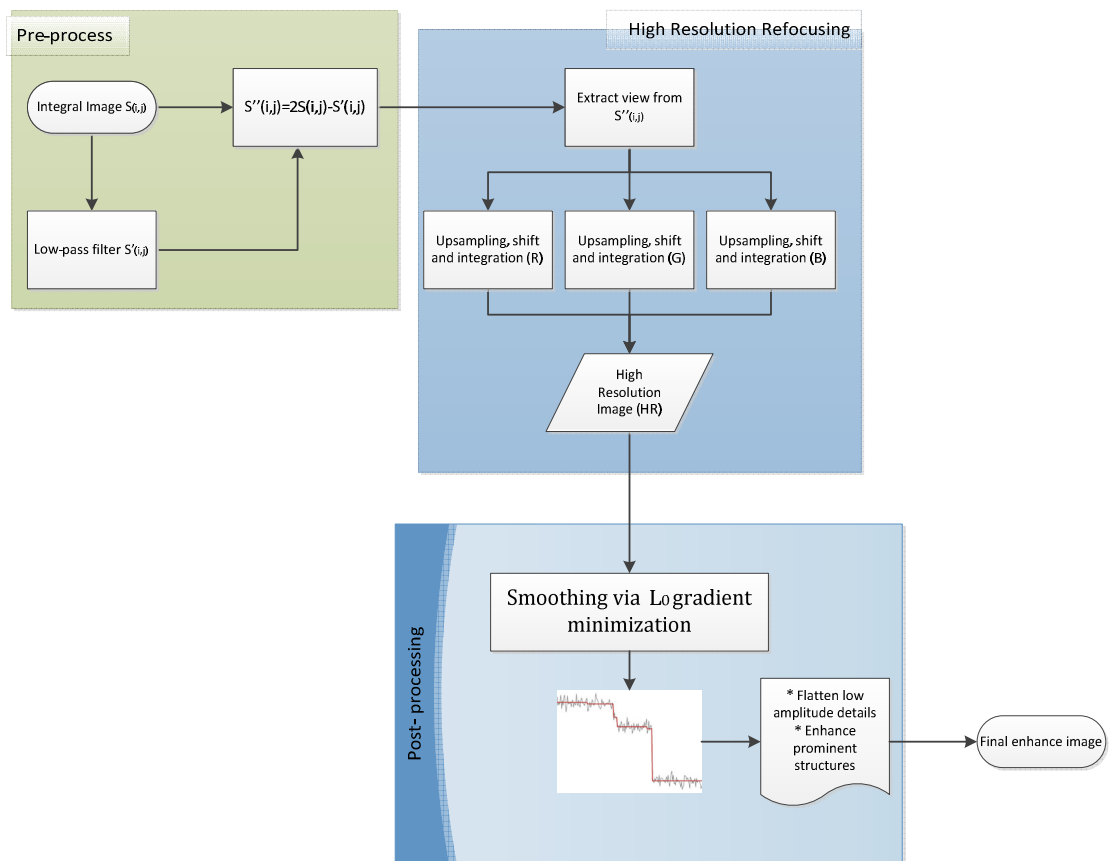


Fig 5. 1 : Illustrate the three steps pre-process, high resolution refocusing and post-processing respectively in achieving the final enhanced image.

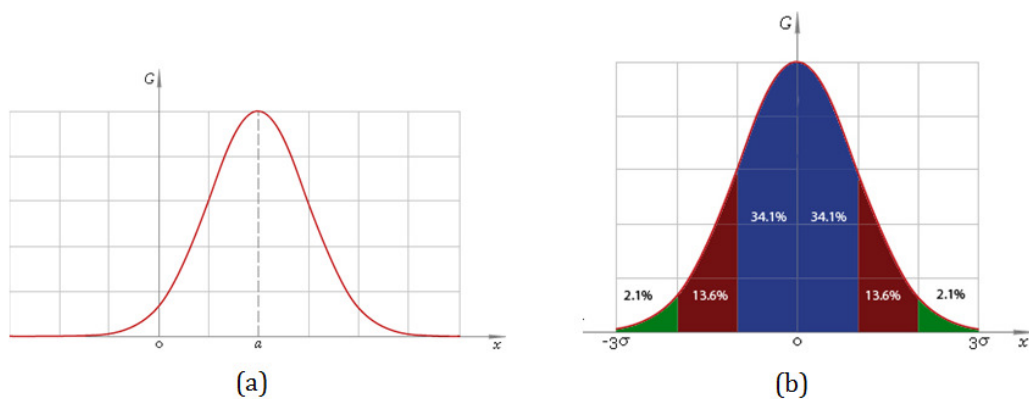


## 5.2 Image Correction on Pre-processing

Before proceeding with high resolution refocusing, the noises of the acquired holographic 3D images (H3DI) are removed that affect the final results. As mentioned in chapter 3, the microlens array of low quality are placed in front of camera, where it will refract light by introducing noise on the captured image. It is vital to recover every detail of the image while removing the unnecessary noise that is obtained during the capture stages. In order to decrease the noise on the holographic 3D image, an efficient image processing technique is introduced.

The image is first processed using low-pass Gaussian filtering by suppressing noise and small fluctuations. In the frequency domain, this process refers to the suppression of high frequencies [5]. The most general function formula in 1D is given in eq 5.1

$$G(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-a)^2}{2\sigma^2}} \quad \text{eq (5.1)}$$



**Fig 5. 2 : (a) Gaussian or normal distribution and (b) Gaussian distribution truncated at point  $\pm 3\sigma$**   
 Where  $\sigma$  is the standard deviation of the distribution and  $a$  is a statistical expectation responsible for distribution shifting along  $x$  axis to be zero:  $a = 0$ ; and work with simplified form in eq 5.2. The distribution is assumed to have a mean of 0. Graphical illustration of Gaussian distribution is shown in Fig 5. 2. The standard deviation of the Gaussian function plays an important role in its behavior.

$$G(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}} \quad \text{eq (5.2)}$$

Note that real axis  $x \in (-\infty, \infty)$  spread endlessly to the left and right. Therefore, using the rule of  $3\sigma$  Gaussian distribution is utilized, which is  $x \in [-3\sigma, 3\sigma]$  as shown in Fig 5. 2b. The values set between  $\pm \sigma$  account for 68% of the set, while two standard deviations from the mean (blue and brown) account for 95% and three standard deviation (blue, brown and green) account for 99.7%. If input signal  $S = \{s_i\}$  is applied for every signal element  $s_i$ , a new modified value  $s'_i$  will be calculated. In other words, “for every element put in window so that this element is in the centre of the window, multiply every element in the window by corresponding weight and sum up all those products, the result of which is the new filtered value.” The smoothing is achieved without suppressing high frequencies to enhance the signal. The method is described below.

$$s''_i = 2s_i - s'_i \quad \text{eq (5.3)}$$

The above formula suppresses the low frequencies and also amplifying the high frequencies, as it is simple and effective in removing the noise in the signal. This is shown in Fig 5. 3, where  $s''_i$  is the modified value from the original signal element of  $2s_i$  subtracting it from the smooth signal element of  $s'_i$ . The above eq (5.3) acts as unsharpened filter that derives its name from the fact it enhances edges via a procedure which subtracts an unsharpened, or smoothed version of signal from the original signal. This method is commonly used in the photographic and printing industries for crispening edges [27]. To better understand how the eq (5.3) is derived, below shows the working.

$$s_{ig} = s_i - s'_i \quad \text{eq (5.4)}$$

Where  $s'_i$  is the smooth signal of  $s_i$ , the original signal.

When subtracting away the low-passed signal from original signal that will produce only a high-passed signal of  $s_{ig}$ , Fig 5. 3(c) is generated. This high-

passed signal can be used to achieve enhanced signal combined with original image. Adding back the original signal to  $s_{ig}$  will sharpen the signal as defined in eq (5.5). The characteristic of this signal response is shown in Fig 5. 3(d).

$$s''_i = s_{ig} + s_i \tag{5.5}$$

The eq (5.5)'s complete form is shown below in eq (5.6).

$$s''_i = (s_i - s'_i) + s_i \tag{5.6}$$

Where this is further simplified and expressed in eq (5.3) as shown in Fig 5. 3(e).

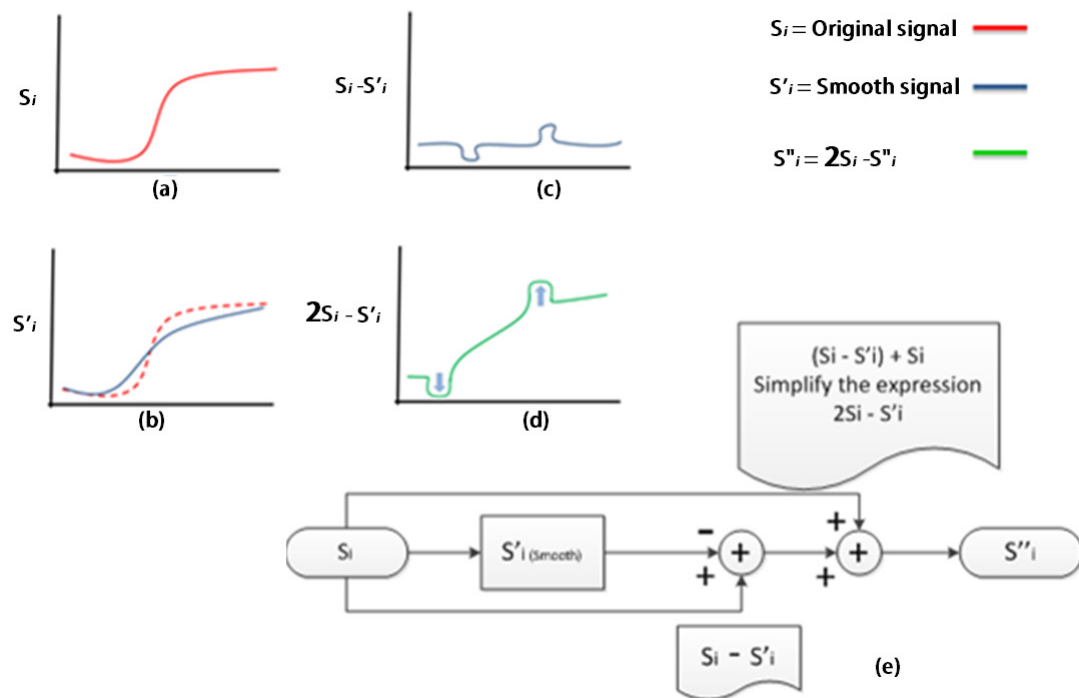


Fig 5. 3 : Presentation of 1D Signal. (a) Original signal, (b) smooth signal, (c) is the difference between (a) and (b), (d) is (a) minus (b) plus (a) and (e) is the complete filtering operator.

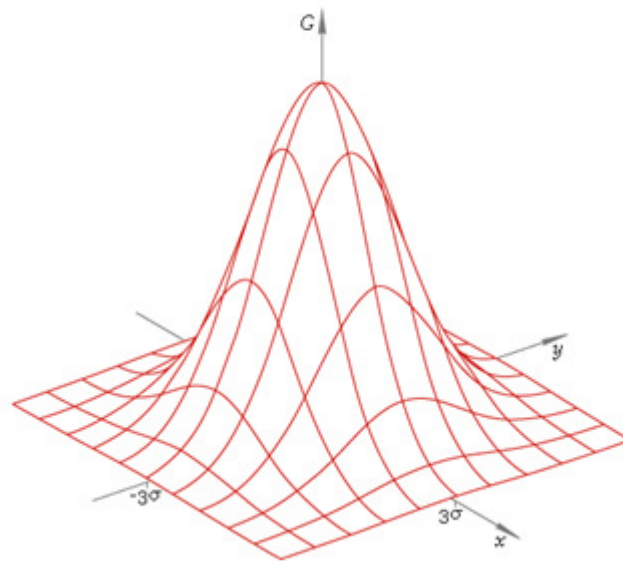
The red line in Fig 5. 3 signifies the original signal before smoothing by the Gaussian filter. Smoothing will suppress the high frequencies and amplifies low frequencies element whereas  $s''_i$  suppresses the low frequencies and amplifies the high frequencies element depending on the smooth signal (see Fig 5. 3). While working with images, it's important to use the two-dimensional Gaussian function which is expressed in eq 5.9. This is simply the product of two 1D Gaussian functions eq (5.7) and eq (5.8), which means that 2D distribution is

split into a pair of 1D in horizontal and vertical directions as shown in Fig 5. 4 for 2D distribution.

$$G(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}} \quad \text{eq (5.7)}$$

$$G(y) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{y^2}{2\sigma^2}} \quad \text{eq (5.8)}$$

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{y^2}{2\sigma^2}} = G(x)G(y) \quad \text{eq (5.9)}$$



**Fig 5. 4 : 2D Gaussian or normal distribution**

The 2D Gaussian filter is applied on the original image as a point-spread function and this is by convolution. Since the images are stored as a collection of discrete pixels, that need to produce a discrete approximation to the Gaussian function before convolution is applied. After a suitable kernel has been calculated, the Gaussian smoothing can be performed using standard convolution methods. The convolution is applied efficiently since the equation for 2D Gaussian shown above (see eq (5.8)) is separable into x and y components. The 2D convolution is performed by first convolving 1D Gaussian along x direction and then along y direction with another 1D Gaussian. The Gaussian smoothing function is to blur an image, which is similar to the mean

filter. However, in this particular one, the degree of smoothing is determined by standard deviation of the Gaussian. The smoothing for each pixel is weighted average that is more towards the value of the central pixels. This provides gentler smoothing and preserves edges better than a similarly sized mean filter does. Finally the eq 5.3 is applied where the original image of  $s_{i,j}$  is multiplied by 2 and is subtracted from the filter image of  $s'_{i,j}$ , which will have an inverse smoothing effect along x and y directions, without affecting the details of the image. Fig 5. 5 shows the obtained results by performing the pre-processing stage. In doing so it will not act as a high-pass filter by amplifying both high and low frequencies, but it will reduce the noise on the image without affecting the details. In other words, when multiplying the original image by 2, this will increase the effect by a factor of 2. Whereas on the Gaussian filtered image, the smoothing process reduces noise while affecting the details, as shown in Fig 5. 5(b). In order to apply negative smoothing effect on the image with the same amount that is applied on the original image effectively without distressing the details, taking the  $2s_{i,j}-s'_{i,j}$  approach will sustain a detailed image of the HI, as every detail in the HI is vital for high resolution refocusing process.

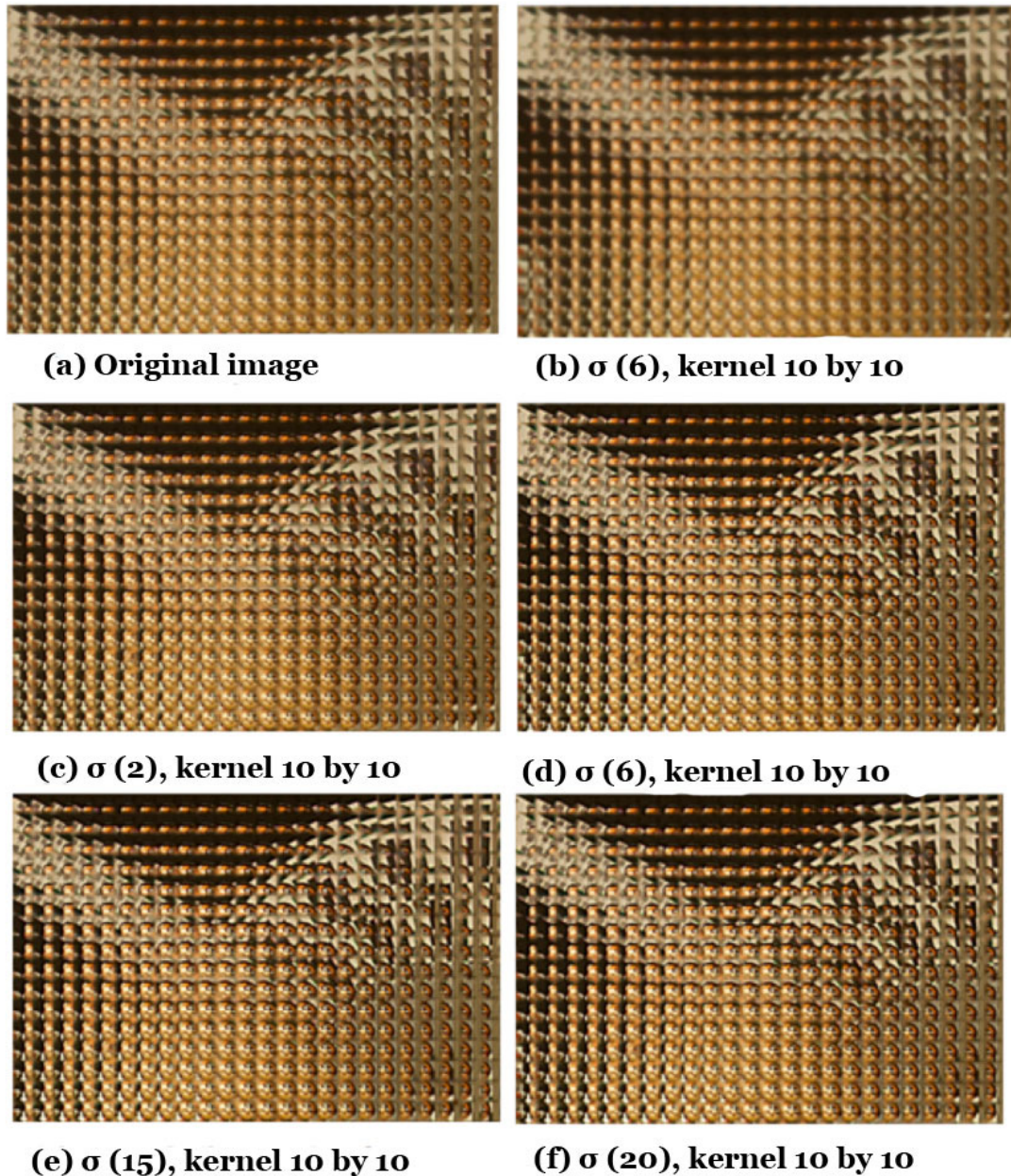


Fig 5. 5 : Illustrates the results of pre-processing with different values sigma. (a) section of portion of the original HI. (b) low-pass image, (c), (d), (e) and (f) are the result of  $s''_{i,j}$  with different sigma value and with 10 by 10 kernel.

### 5.3 Image Smoothing on Post-Processing

Post-processing is an important stage in this section, by effectively removing part of the noises such as slight blur to make the result more photographic without any artifacts on the final image. In this post-processing smoothing technique via  $L_0$ , gradient minimisation [6] approach is adopted, which is particularly effective for sharpening major edges by increasing the steepness of transition while eradicating manageable degrees of low-amplitude structures.

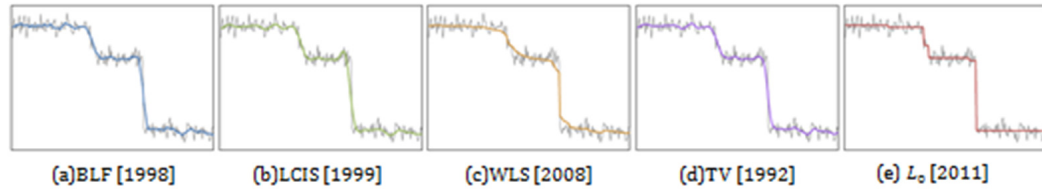
This edge-preserving smoothing approach depends on global important edge instant of local feature [7][8][9][10][11][12], which aims to globally maintain and possibly enhance the most prominent set of edges by increasing steepness of transition while not affecting the overall acutance [6].

The  $L_0$  gradient minimisation produces smoothing results based on new metric to discreetly constrain the number of none-zero gradients. In Fig 5. 6, a 1D scanline of a natural image is presented, after restricting the number of none-zero gradients while smoothing is performed in a global manner. Lower amplitude structure is removed during the process, though it does not reduce blur salient edges even with a close approximation.

### **5.3.1 The Background**

The study in [6] presents most of the edge-preserving smoothing operation that can be achieved using local filtering. The most commonly used filter is bilateral filter for its simplicity in removing noise-like structures effectively. Tomasi and Mansuchi first adopted this method in 1998 [7] and later a fast version of this method was presented in [8][13], producing similar results with less computation time. Shortly after 2006 a real-time bilateral smoothing technique was proposed by Chen and Paris in [14]. Other similar bilateral filters emerged in later years [15][16][17][18].

The bilateral filter provides a trade-off between details flattening and sharp edge preservation when it is compared with weighted least square (WLS) method [10]. Another filter was reported in [19][20] for suppressing noise while preventing its effect on strong edges. At the same time in 1992 total variation (TV) method was used in [6] where the process influences the contrast with large gradient magnitudes during smoothing. The method called edge-preserving multiscale image decomposition based on local extreme was reported in 2009 [11]. The method is originated from Hilbert-Huang transfer (HHT) and is used to remove small noises.



**Fig 5. 6 : Result obtained for both details and sharp edges of an image. (a) Bilateral filtering. (b) Anisotropic diffusion used in the LCIS system. (c) Weighted least squares optimization. (d) Total variation smoothing. (e) Smoothing via  $L_0$  gradient minimisation.**

The  $L_0$  gradient minimisation smoothing works by accurately selecting the boundaries using graph-cut techniques based on the work reported in [21][22][23] and segmentation process reported in [24][25]. An approach to deal with textural replacing a geodesic image and video editing was proposed in [26] and techniques for diffusion maps for edge-aware image editing was reported in [17]. The aim is to effectively remove the noise and globally preserve and enhance salient edges.

### 5.3.2 $L_0$ Gradient Minimisation

This method confines the discrete number of intensity change among neighboring pixels that links mathematically to the  $L_0$  norm for formation sparsity pursuit. It leads to new discrete metric, as reported in [6] involving global optimisation that enables diversified edge manipulation, making edges easier to detect and more visually distinct. This smoothing method globally locates the most prominent set of edges while effectively removing part of noises, unimportant details, and even of slight blurriness, making the resulting image visually high quality without affecting the overall acutance around the edges.

In 2D image representation, the gradient  $\partial_x S_p, \partial_y S_p$  for each pixel  $p$  is calculated as a color difference between neighboring pixels along  $x$  and  $y$  directions. The auxiliary variables  $h_p$  and  $v_p$  are introduced that correspond to  $\partial_x S_p$  and  $\partial_y S_p$ , respectively.  $\beta$  is an automatically adapting parameter to control the similarity between variable  $(h, v)$  and their corresponding gradients. In [6], the equation was diagonalised to derivative operator after Fast Fourier Transform (FFT) is applied to speed up the process (see eq (10)).



$$S = \mathfrak{f}^{-1} \left( \frac{\mathfrak{f}(I) + \beta(\mathfrak{f}(\partial_x) * \mathfrak{f}(h) + \mathfrak{f}(\partial_y) * \mathfrak{f}(v))}{\mathfrak{f}(1) + \beta(\mathfrak{f}(\partial_x) * \mathfrak{f}(\partial_x) + \mathfrak{f}(\partial_y) * \mathfrak{f}(\partial_y))} \right) \quad \text{eq (10)}$$

Where  $\mathfrak{f}$  is the FFT operator and  $\mathfrak{f}()$ \* denotes the complex conjugate.  $\mathfrak{f}(1)$  is the Fourier Transform of the delta function. The plus, multiplication, and division are all component-wise operators. Computing in the Fourier domain is much faster due to the simple component-wise division [28]. Also defined in eq (11) is its minimum energy  $E_p^*$  condition, where for each pixel  $p$  the minimum energy  $E_p^*$  is computed.  $\lambda$  is a smoothing parameter that controls the degree of smoothing. A large  $\lambda$  makes the result have very few edges.

$$(h_p, v_p) = \begin{cases} (0, 0) & (\partial_x S_p)^2 + (\partial_y S_p)^2 \leq \lambda/\beta \\ (\partial_x S_p, \partial_y S_p) & \text{Otherwise} \end{cases} \quad \text{eq (11)}$$

**The minimisation algorithm is sketched below:**

**Input:** image  $I$ , smoothing weight  $\lambda$ , parameter  $\beta_0, \beta_{max}$  and rate  $\kappa$

**Initialization:**  $S \leftarrow I, \beta \leftarrow \beta_0, i \leftarrow 0$

**Repeat**

With  $S^{(i)}$ , solve for  $h_p^{(i)}$  and  $v_p^{(i)}$  with eq (11)

With  $h^{(i)}$  and  $v^{(i)}$ , solve for  $S^{(i+1)}$  eq (10)

$\beta \leftarrow \kappa\beta, i++$

**Until**  $\beta \geq \beta_{max}$

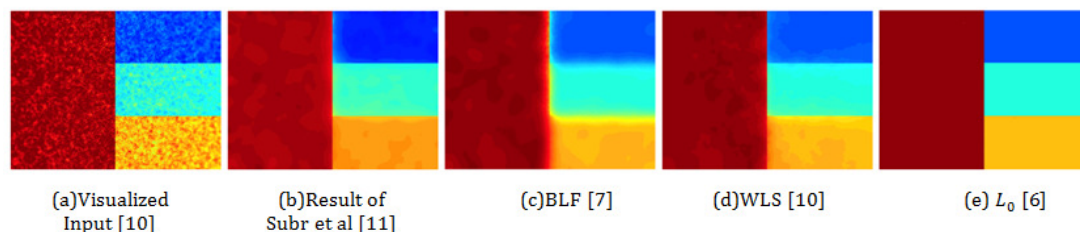
**Output:** result image  $S$

The parameter  $\beta$  is automatically adapted in iteration starting from a small value  $\beta_0$ ; it is multiplied by  $\kappa$  each time. This scheme is effective to speed up the convergence [6].

### 5.3.3 The Smoothing Technique

After successfully rendering the refocusing images from recorded HI as described in chapter 4, blurry noise is still noticeable to the naked eyes. Therefore, a post-processing technique is considered to take care of the unnecessary noises that are observed on the image. Image smoothing via  $L_0$  gradient minimisation method is taken into consideration. This is chosen due to its strong outcomes as reported in [6] when using the 2D example of Farbman et al in 2008, to evaluate and compare the smoothing performance. Fig 5. 7(a)

shows the results with the same noisy image created by Farbman et. al (2008) to illustrate the performance of smoothing. Study in [6] reported that image smoothing via  $L_0$  gradient minimisation generated a cleaner result (see Fig 5. 7(e)) among other approaches, as shown in Fig 5. 7(b)-(d).



**Fig 5. 7 : Visual representation created by Farbman. (a) Color visualized Noisy input image (b) Result of Subr et al [11]. (c) Result of Bilateral filtering (BLF) with ( $\sigma_s=12$ ,  $\sigma_r=0.45$ ). (d) Weighted least square optimization ( $\alpha=1.8$ ,  $\lambda=0.35$ ). (e) Smoothing via  $L_0$  gradient minimisation ( $\lambda=0.01$ ).**

This smoothing is based on a sparsity measure of global preservation of edges, even in very narrow object boundaries. Two features of this smoothing operator that are ideal for our post-processing stage are that, (i) it flattens the insignificant details by removing small non-zero gradients and (ii) it enhances prominent edges because large gradients receive the same penalty as the small ones. As shown in Fig 5. 7(e), acceptable removing of the noise and preserving the edges result in rich textual image, whereas Fig 5. 7 (b) to (d) fail in doing so.

#### 5.4 Experimental Results and Observations

In this process, the blurry noise affect is removed successfully in refocused image that was seen in chapter 4. Fig 5. 8(a) shows the outcome of refocused image portrayed against Fig 5. 8(b), which is the result of performing the image smoothing via  $L_0$  gradient minimisation to enhance the quality through successfully removing the blurring noise on the focused region as well as smoothing the defocused region, giving more natural blurring effect. Also the results obtained look far better in comparison to (a), reason being that this smoothing automatically determines a different size of Gaussian kernels for each pixel to optimum smoothing. Therefore, an enhanced edge can visually be observed on the final image in the Fig 5. 8(b).

Fig 5. 9(a) shows the image plane focused on foreground object, portrayed against Fig 5. 8(b), which is focused on the background, from the same HI in Fig

5. 8(c) performing the refocusing algorithm in chapter 4. The Sub-image (SI) is set to 2; shift ( $\delta$ ) to 6 and 7 by 7 different views are used in Fig 5. 8(a) whereas in Fig 5. 9(a), SI= 2,  $\delta=1$  and the same 7 by 7 different views are used to change the focus plane to foreground object.

### 5.4.1 Subjective quality comparison



Fig 5. 8(a) : Illustration of image focused on the background. (SI = 2,  $\delta = 6$  and 7 by 7 different views).

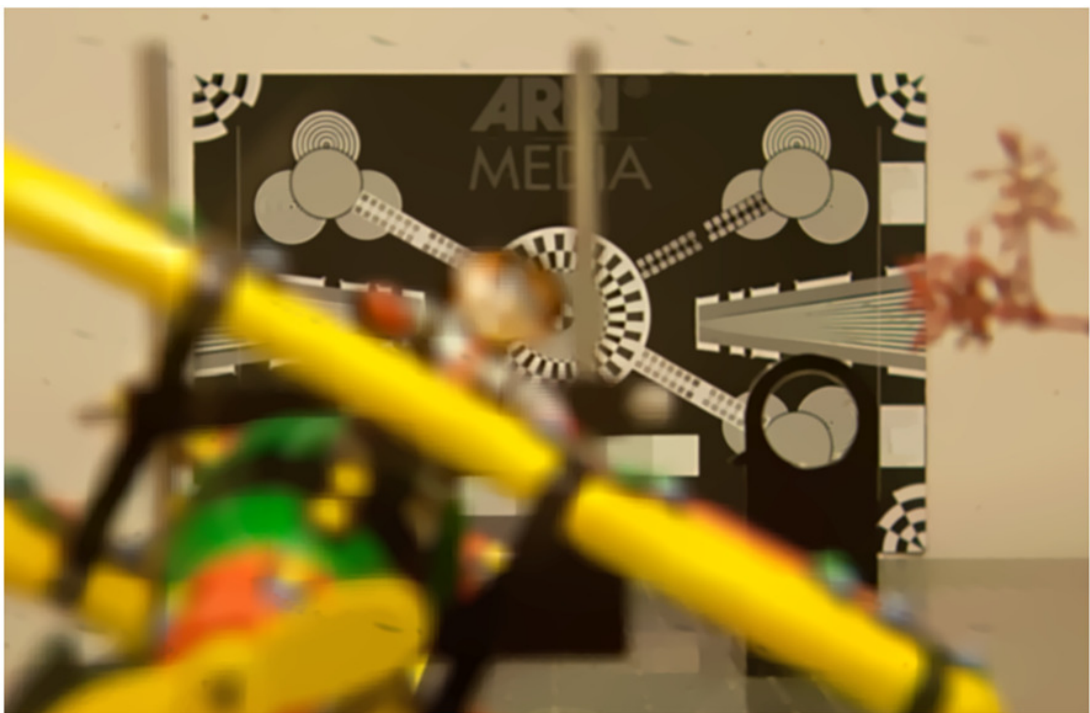


Fig 5. 8(b): Demonstrates the result of Fig 5. 8(a) after performing smoothing via  $L_0$  gradient minimisation ( $\lambda=0.007$ ).



Fig 5. 8(c) : Demonstrates HI with microlens pitch = 90µm, focal length = 1mm, EI size 29 by 29 pixels resolution.

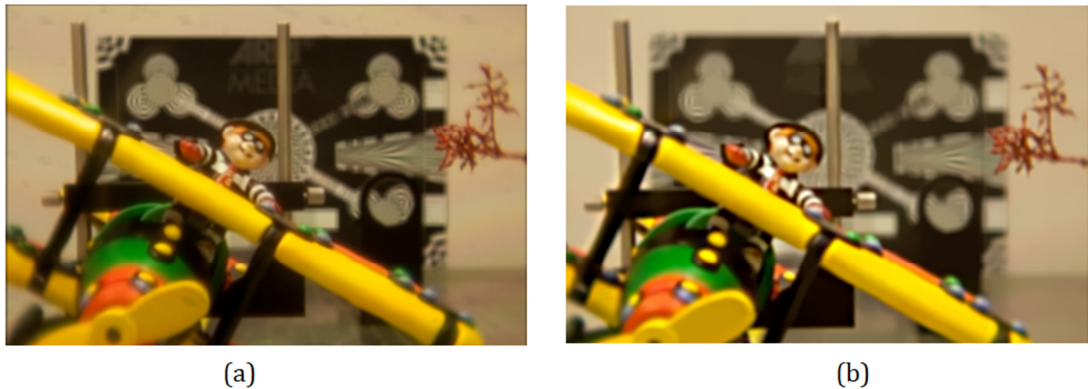
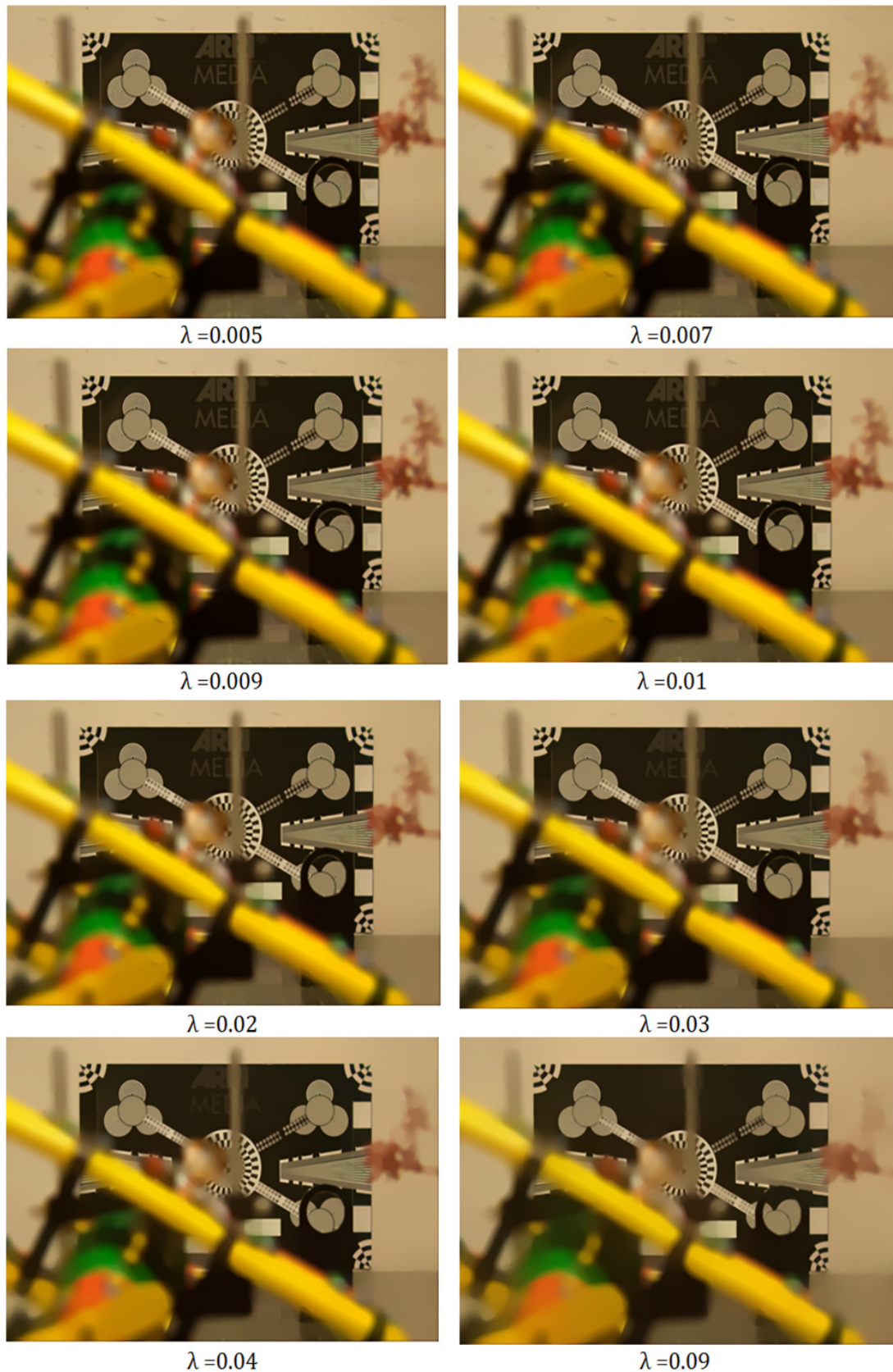


Fig 5. 9 : (a) Shows the image plane focused on the foreground object with set parameters ( $SI = 2$ ,  $\delta = 1$  and 7 by 7 different views). (b) is the result from (a) using image smoothing via  $L_0$  gradient minimisation ( $\lambda=0.007$ ).

As seen in the figures, increasing the lambda ( $\lambda$ ), which is the smoothing parameter controlling the degree of smoothing, affects the edge strength. Thus the parameter  $\lambda$  is set manually depending on the input image to achieve the best possible outcome. Just to note that when smoothing an image in some filters it causes blurring to the edges. However,  $L_0$  gradient minimisation sharpens the edges while flattening the noise to an appropriate level for each pixel. The obtained results demonstrated the effectiveness of this smoothing approach.



**Fig 5. 10 :** Illustrates results of smoothing via  $L_0$  gradient minimisation with different lambda starting from 0.005, 0.007, 0.009, 0.01, 0.02, 0.03, 0.04 and 0.09 respectively and other parameter are set the same throughout in this figure where the focus is set to the background with set parameter,  $SI = 2$ ,  $\delta = 6$  and 7 by 7 different views.



**Fig 5. 11 : Illustrates results of smoothing via L<sub>0</sub> gradient minimisation with different lambda starting from 0.005, 0.007, 0.009, 0.01, 0.02, 0.03, 0.04 and 0.09 respectively and other parameter are set the same throughout in this figure where the focus is set to the foreground object with set parameter, SI = 2,  $\delta = 1$  and 7 by 7 different views.**

## **5.5 More Analysis of Resulting images**

To clarify the difference in setting the lambda, different images are used in order to find the optimum value in successfully suppressing the noise to its minimum by keeping the natural photographic look. In general photos normally consist of very small amount of noise. But with clever image processing, noise can be removed while maintaining a fine edge remains still difficult. In our case, however, the refocused image contains noise as well as blur due to two factors. (1) The microlens array is of a low quality and also it has scratch marks which introduces noise in the final image as it abstracts light flow to right direction(s), (2) the blurring affect in the refocused image is obtained with our refocusing algorithm as explained in chapter 4 with recursive or iterative shift and integration of different views causing the blurring effect on the overall image. Therefore gradient minimisation smoothing is considered to compensate both suppressing noise without deteriorating edges and removing the overall blur on the image down to its minimum.

Detailed observation of our experiment reveals the best  $\lambda$  weight, namely  $0.005 < \lambda < 0.02$ , in achieving the optimum result. This can be perceived from our experiment by generating high quality result in suppressing noise and preserving the edges. On the other hand, increasing the  $\lambda$  greater than 0.02 will diminish insignificant details of the image, which looks more non-photorealistic as this can be seen in the results of our experiment.





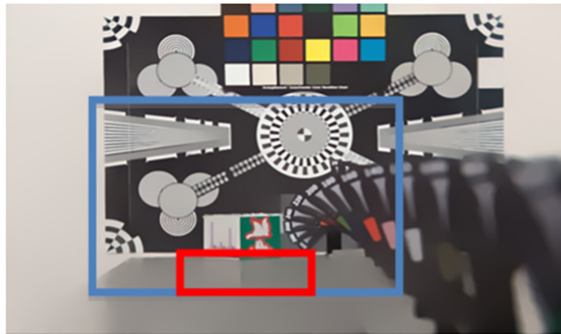
**Fig 5. 12 : Results of smoothing via L<sub>0</sub> gradient minimisation with different lambda with 0.007, 0.01, 0.02, 0.03, 0.04 and 0.09 respectively and other parameter are set the same throughout in this figure. Parameter, SI = 4,  $\delta = 4$ , microlens pitch = 90 $\mu$ m, focal length = 1mm, EI size 29 by 29 pixel resolution.**



**Fig 5. 13** Our final refocused image results with different lambda values. .  $\lambda = (0.007, 0.01, 0.02, 0.03, 0.04 \text{ and } 0.09)$  respectively,  $SI = 4$ ,  $\delta = 2$ ,  $7$  by  $7$  different views, microlens pitch =  $90\mu\text{m}$ , focal length =  $1\text{mm}$ .

Our smart filter framework produces an effective result in removing blur on the focused plane and suppressing noise down to its minimum acceptable level with  $\lambda$  set to  $0.005 < \lambda < 0.02$  as mentioned above. Fig 5. 14 shows the shadow of the cube that is faded away in red square due to low contrast that is indistinguishable around boundaries with large  $\lambda$ . This resulted in significant details loss by global optimisation and maintaining the main structure, while slightly sharpening the overall image. The same resulting image goes under

post-processing with  $\lambda$  set to 0.007, which does not affect the major details of the image, yet it mainly removes unwanted small enough structures that are considered as noise, leaving the final image in much higher quality with more fine edges (see Fig 5. 15). Notice that the faded region in Fig 5. 14 remains visible in Fig 5. 15. Further, other major details look more photographic in Fig 5.15 compared to those in Fig 5. 14.

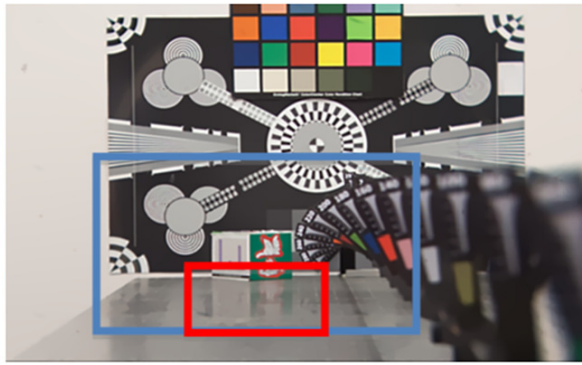


(a)



(b)

Fig 5. 14 : (a) Post- processing result with  $\lambda$  set to 0.09 focuses on the test chart. (b) shows the close-ups results in (a) that is illustrate with two square color blue and red.



(a)



(b)

Fig 5. 15 : (a) Post- processing result with  $\lambda$  set to 0.007 focuses on the test chart. (b) close-ups results in (a) that is illustrate with two square color blue and red.

## **5.6 Subjective Assessment of Image Quality**

Image quality is measured in two ways; subjective and objective methods [29]. Subjective image quality assessment is geared directly toward properties of the human visual system where the evaluation of quality is obtained by mean opinion score (MOS) [30]. Objective image quality assessment is designed mathematical to predict the quality of an image accurately and automatically. The ideal Objective image quality assessment is to mimic the quality predictions of an average human observer based on the availability of reference image that is measured to be undistorted and have perfect quality [31]. Since there is no reference image, as the images are reconstructed from Holographic 3D image, subjective image quality assessment is considered. This is to obtain a better understanding of how the spatial resolution affects the perceived quality in different refocusing algorithms. The main goal of many subjective image quality assessments is based on objective judgment and rational comparison of an external image with the image imprinted or remembered more or less distinctly by the subject [32]. Since human observers are the ultimate users in most of the multimedia applications, the most reliable and accurate methodology of assessing the quality of image is through the subjective evaluation [29]. This section investigates subjective assessment for four different algorithms of refocusing on four images. Subjective tests are performed to verify the quality image for each refocusing algorithm including the proposed refocusing algorithm.

### **5.6.1 Review of Subjective Quality Assessment Methods**

The most reliable method for assessing the quality of image is through the human visual system. In this method, human performs the task of assessing visual quality and understanding the quality perception [33]. In subjective testing a group of observers are asked to give their opinion on each of the image quality. In order to perform subjective image quality testing, numerous international standard frameworks have been proposed to provide reliable results [32]. Below are some of the international image quality testing standards:

- ITU-R BT.500-11 [34] proposes the standard of different subjective quality assessment of television content. This standard quality assessment method contains information about the observer’s viewing condition, instructing how to conduct the subjective experiment, test materials and presentation style of subjective results.
- ITU-T P.910 [35] proposed the standard for video quality assessment with transmission rate below 1.5 Mbits/sec.
- ITU-R BT.814-1 [36] proposed to set the brightness and contrast of the testing equipment.
- ITU-R BT.1129-2 [37] is for assessing the quality of the standard definition (SD) in video sequence.

Below is a list of standardized subjective IQA methods:

1. Single stimulus categorical rating
2. Double stimulus categorical rating
3. Ordering by force-choice pair-wise comparison
4. Pair-wise similarity judgments

### 5.6.2 Description of Assessment Methods

We applied subjective image quality assessment methodology to obtain the participant’s opinions using single stimulus methods (SS). In this method, test images are displayed on a monitor for a fixed period of time, and then observers are asked to rate their quality on a scale of one to five : excellent, good, fair, poor or bad. All rendered refocused images are randomly displayed on the monitor to avoid quantisation artifacts, (see Table 5. 1 for grading). In this assessment, no comparison with an impaired reference is made during the presentation and only single test image is displayed each time. In Fig 5. 16 a typical structure representation of the method is shown.

**Table 5. 1 : Five grade quality scale**

1	2	3	4	5
Bad	Poor	Fair	Good	Excellent

Assessors have been given a survey form, which includes the scale very clearly, and has numbered boxes or some other means to record the grading.

### 5.6.3 Presentation of the test material

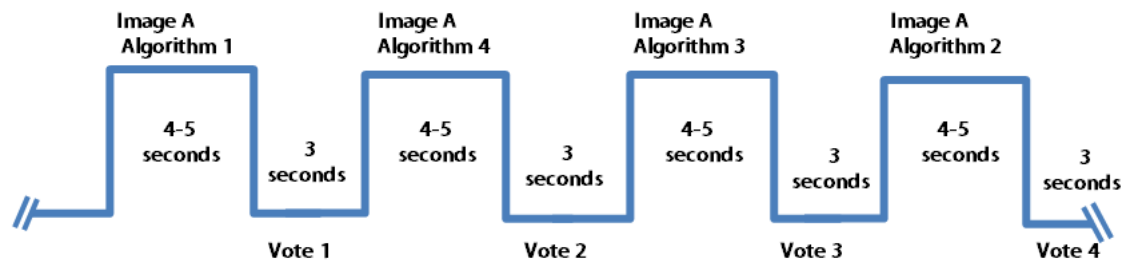


Fig 5. 16 : Structure of presentation for subjective test material

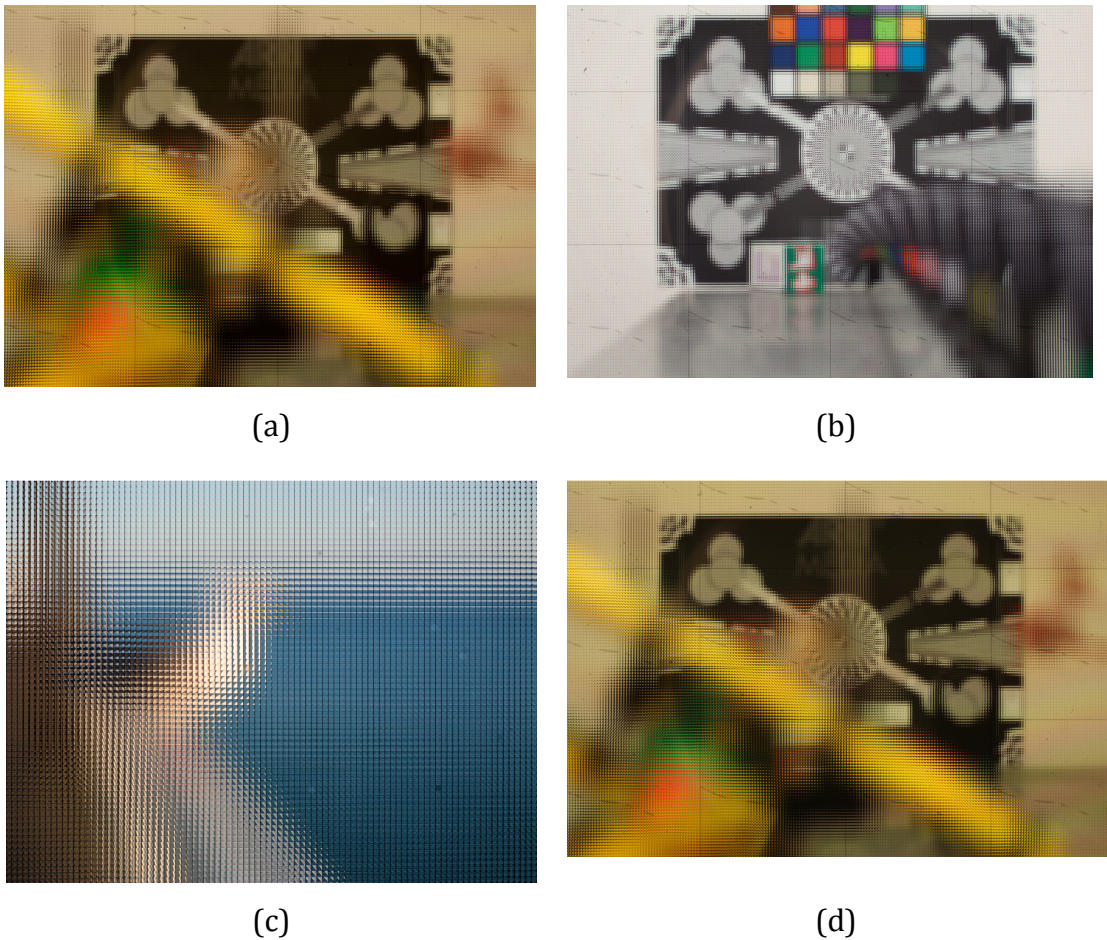
At the beginning of each session, it is important to give the observer an explanation of the whole process of evaluation. That includes type of assessment, the grading scale, the image sequence and timing (test image, voting period). The range and the type of images are shown to the observer for better understanding of how it is done. The observer should be asked to base their judgment on the overall impression given by each image. The observer is asked to continuously look at the display monitor when the image is displayed for 4-5 seconds, then voting should be permitted only during 3 seconds after image is disappeared from the display monitor.

### 5.6.4 Equipment used and viewing conditions

In the experiment, we displayed the images on a 24-inch display monitor with screen resolution of 1920x 1200 pixels, at frequency of 60 Hz x 60 Hz. The display monitor was at a viewing distance of 1m. Each subject had to judge all the images from the same distance. All participants are asked to sit at 1m distances away from the display monitor during the experiment. All the images are displayed in a controlled environment with a constant balance of light in the lab.

### 5.6.5 Databases of Subjective results & test materials

The dataset that we used for this experiment were those that have been used in previous chapters in this thesis and also the *seagull* that made public by researcher Todor Georgiev at tgeorgiev.net. These images were rendered using four algorithms, 1) Native VP refocusing [40], 2) Full resolution refocusing [38], 3) Full resolution with blend [4][39] and 4) SPA with smart filters (proposed method). These images are used for testing in order to show that the proposed method renders images in higher quality across other mentioned algorithms.



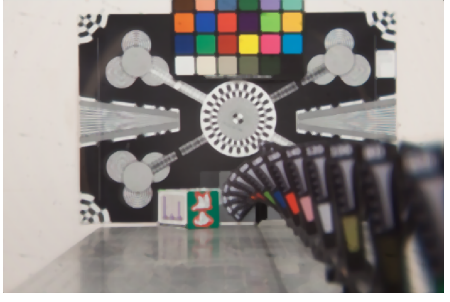
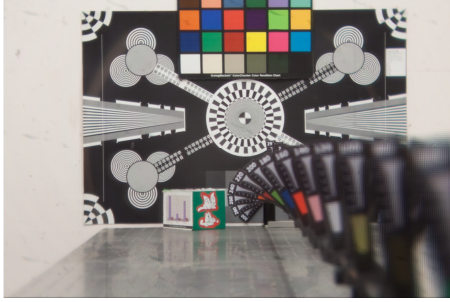
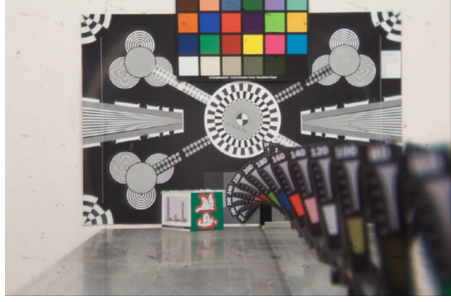
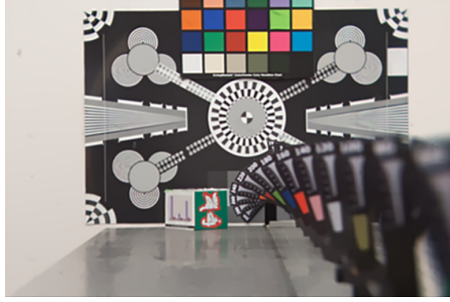


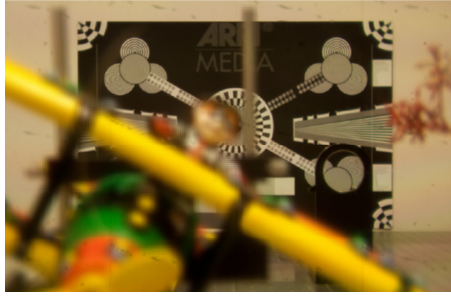
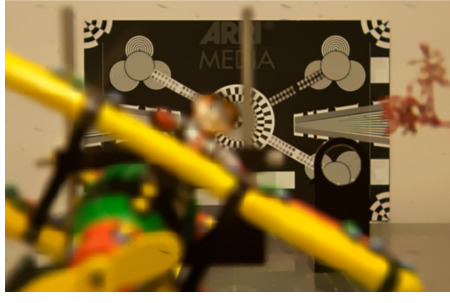
**Fig 5. 17 : Illustration of dataset used in testing, (a) OHI Image A with micro image size 29x 29 pixels, (b) OHI image B with micro image size 29x 29 pixels, (c) OHI image C with 74x 74 pixels and OHI image D with 29x 29 pixels with different lighting condition than image A.**



**Table 5. 2 : Presentation of algorithms**

Count	Name	Description
1	Algorithm 1	Native VP refocusing
2	Algorithm 2	Full resolution refocusing
3	Algorithm 3	Full resolution with blend
4	Algorithm 4	SPA with smart filters (proposed method)

Table 5.3 : Presentation of rendered image for subjective testing

Name	Algorithm 1	Algorithm 2	Algorithm 3	Algorithm 4
A				
B				



### 5.6.6 Subjective Participants

In order to measure human perception on the quality of rendered images using different algorithms, a total of 35 candidates with 19 males and 16 females were volunteered to participate in the CMCR lab experiment at Brunel University. Their age range varies from 18 to 55 with good colour vision. The observers were mostly research students including few undergraduate students with a relevant technical background. All experiments were conducted based on the ITU-R requirement of subjective quality image assessment, based on which the participants are not experienced in or exposed to image quality assessments, nor are they aware of the purpose of the experiment.

Table 5. 4 : Distribution of participants in the subjective studies

Group	Age Range	Female	Male	Sub Total
CMCR Lab, Brunel	18-to-25	6	11	17
	26-to-35	6	6	12
	36-to-45	4	1	5
	46-to-55	0	1	1
	56-to-65	0	0	0
	<b>Total Participants</b>		<b>16</b>	<b>19</b>

### 5.6.7 Subjective Protocol

At the beginning of the experiments, each participant received a brief explanation and performed trial tests. This experiment includes four images that were rendered with different refocusing algorithms in the viewing trial and rating it. But these trial test images were not included in the actual experimental data analysis, as they were only for the purpose of familiarity.

The participants were required to take the actual experiment after completing the trial experiment session. The participants were also asked to accurately grade each image as possible in their judgment, but within a given time period during the voting. After completing the experiment, the participants were required not to change their recorded scores prior to the submission of their scoring sheet.

### 5.6.8 Subjective Grading

For each experiment, the qualities of images were rated by participants on the scale of:

- Excellent, Good, Fair, Poor, Bad
- Very important, Important, Less important, Poor, Not important

Participants were required to continuously assess all of the test images and grade the quality of the image accurately according to the provided scale. Table 5.5 shows the observed results. All the scales were based on ITU-R five point quality scales [37]. Subjective opinion scores obtained from the above studies were averaged across all the subjects to act as an indicator of the perceived image quality. This experiment does not have a reference image to compare it with, but the participants are asked to judge the quality of the image based on the contrast, lighting, artifacts, blurring and colour balance as a whole for each image.

Table 5. 5 : Data view of the Mean Opinion Scores

	A	B	C	D	E	F
1 <b>Users ID</b>		<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>
2 <b>Gender</b>		F	M	F	M	M
3 <b>Age Range</b>		18-to-25	18-to-25	26-to-35	36-to-45	18-to-25
4 <b>Knowledge of Image quality</b>		Good	Excellent	Good	Excellent	Fair
5 <b>Knowledge of Holoscopic 3D</b>		Good	Excellent	Good	Fair	Fair
6 <b>How important is image quality?</b>		Very Important	Very Important	Very Important	Very Important	Important
7						
8 <b>Algorithm 1 image A</b>		Poor Quality	Poor Quality	Poor Quality	Poor Quality	Good Quality
9 <b>Algorithm 2 image A</b>		Poor Quality	Poor Quality	Poor Quality	Poor Quality	Good Quality
10 <b>Algorithm 3 image A</b>		Very Good Q	Good Quality	Good Quality	Very Good Q	Excellent Qua
11 <b>Algorithm 4 image A</b>		Very Good Q	Very Good Q	Excellent Qua	Good Quality	Very Good Q
12						
13 <b>Algorithm 1 image B</b>		Poor Quality	Poor Quality	Poor Quality	Very Poor Qu	Very Poor Qu
14 <b>Algorithm 2 image B</b>		Poor Quality	Very Poor Qu	Poor Quality	Very Poor Qu	Good Quality
15 <b>Algorithm 3 image B</b>		Excellent Qua	Poor Quality	Good Quality	Very Good Q	Excellent Qua
16 <b>Algorithm 4 image B</b>		Excellent Qua	Very Good Q	Excellent Qua	Very Good Q	Very Good Q
17						
18 <b>Algorithm 1 image C</b>		Very Poor Qu	Very Poor Qu	Poor Quality	Very Poor Qu	Good Quality
19 <b>Algorithm 2 image C</b>		Good Quality	Poor Quality	Good Quality	Poor Quality	Poor Quality
20 <b>Algorithm 3 image C</b>		Very Good Q	Good Quality	Good Quality	Very Good Q	Excellent Qua
21 <b>Algorithm 4 image C</b>		Very Good Q	Excellent Qua	Very Good Q	Excellent Qua	Good Quality

### 5.6.9 Mean opinion scores

After completing the screening, the mean opinion score (MOS) of the statistical distribution of the measured for each test image is computed as:

$$MOS = \sum_{i=1}^5 i \cdot p(i) \quad \text{eq (12)}$$

Where  $i$  is the grade of subject participants and  $p(i)$  is the grade probability. The standard deviation is calculated in eq (13) to show how spread out the mean opinion scores is. The relationship between the estimated mean values based on a sample of the population and the true mean values of the entire population is given by the confidence interval of estimated mean.

$$\sigma^2 = \sum_{i=1}^5 (i - MOS)^2 \cdot p(i) \quad \text{eq (13)}$$

$$\bar{X} \pm \alpha \sigma / \sqrt{n} \quad \text{eq (14)}$$

Confidence	$\alpha/2$	Z score
90%	0.05	1.65
95%	0.025	1.96
99%	0.005	2.58

where  $\alpha$  is a degree of confidence,  $n$  corresponds to the number of subjects,  $\sigma$  is the standard deviation of a single test across participants and  $\bar{X}$  is a sample mean that estimates the best point of the confident interval. The confidence interval means that if the same test is repeated for a large number of times using random sample of the population, then the confidence interval will contain the true mean value. We compute our confidence intervals for an  $\alpha$  equal to 1.96, which corresponds to a degree of confidence of 95%. The results from the study clearly shows that the experiments' entire range of quality levels are satisfactory [31]. Also, the confidence intervals are reasonably small, hence proving that the response required from each participant were accurate and consistent with their grading [32]. This is equivalent to say that we are 95%

confident that the population mean or the true mean value lies within the confidence interval calculated by the eq (14).

#### **5.6.10 Results and Analysis**

A detailed statistical analysis of the subjective results was processed using excel tool to obtain the MOS by averaging all the respondents' grades. Also 95% confidence interval was computed. Fig 5. 18 shows the distribution of age and gender for subjective studies. Fig 5. 19 shows the gender representation of the subjective studies.

Fig 5. 20, shows the familiarity of image quality amongst the participants. This is carried out using questionnaire survey before starting the screening test. It illustrates that a high number of participants have a good knowledge of image quality for this studies, a total of 9, 16, 8, 2 and 0 have excellent, good, fair, Don't Know and Bad knowledge of image quality, respectively.

The knowledge of Holographic 3D amongst the participants are shown in Fig 5. 21. A large number of subjects in the studies were unaware of Holographic 3D, whereas 3, 9, 9 and 3 of the subjects expressed their knowledge as excellent, good, fair and bad, respectively.

Through the series of experiments conducted, we analysed all the respondents' feedback and observed how each of the respondents assessed each image using different algorithms. The results were plotted in graph and tabular formats to show the respondents' rated assessment of the image quality.

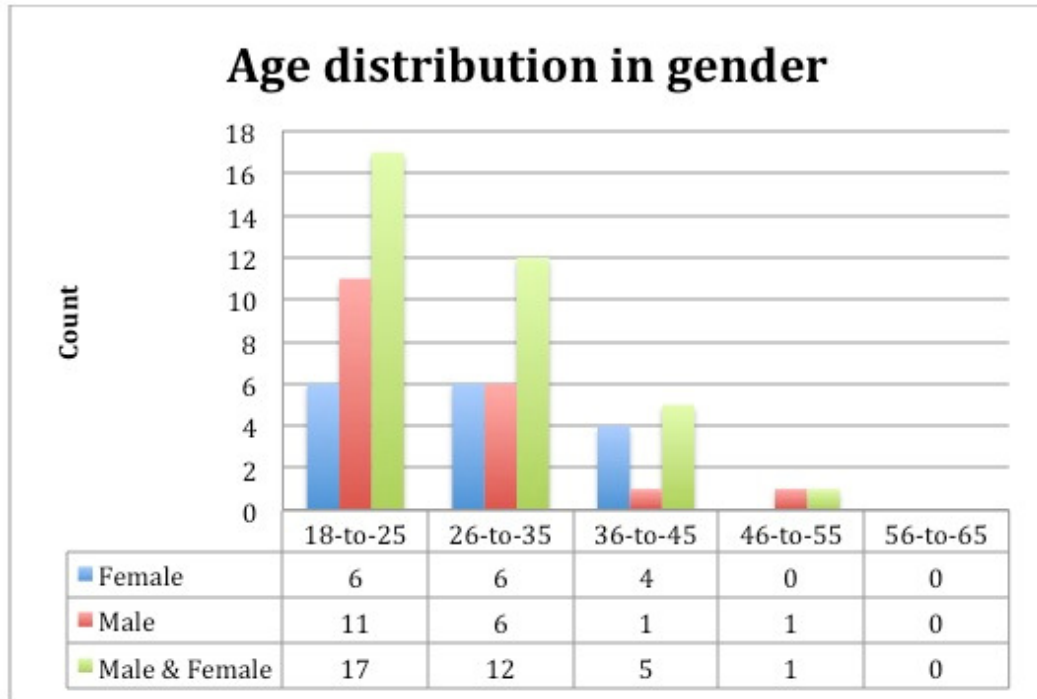


Fig 5. 18 : Participants distribution in term of age and gender using bar chart

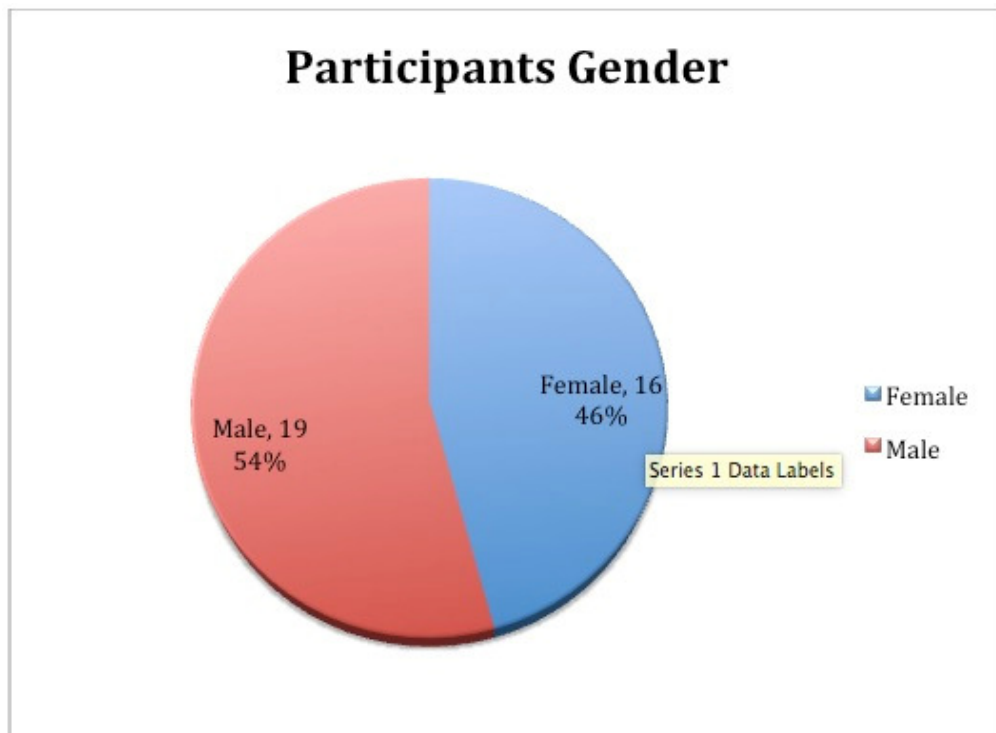


Fig 5. 19 : Gender representation in pie chart



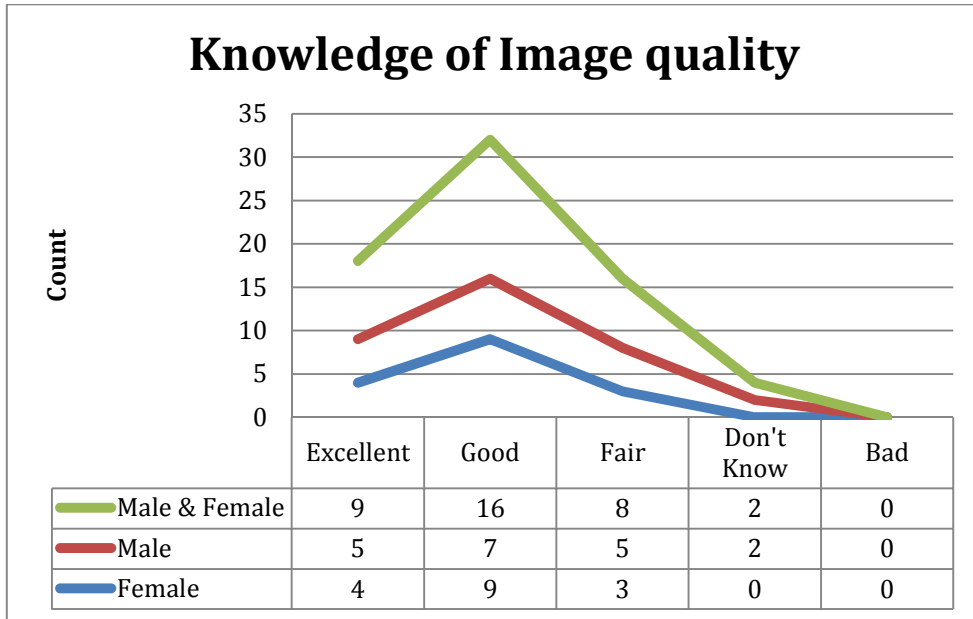


Fig 5. 20 : Representation of image quality in term of gender in line graph

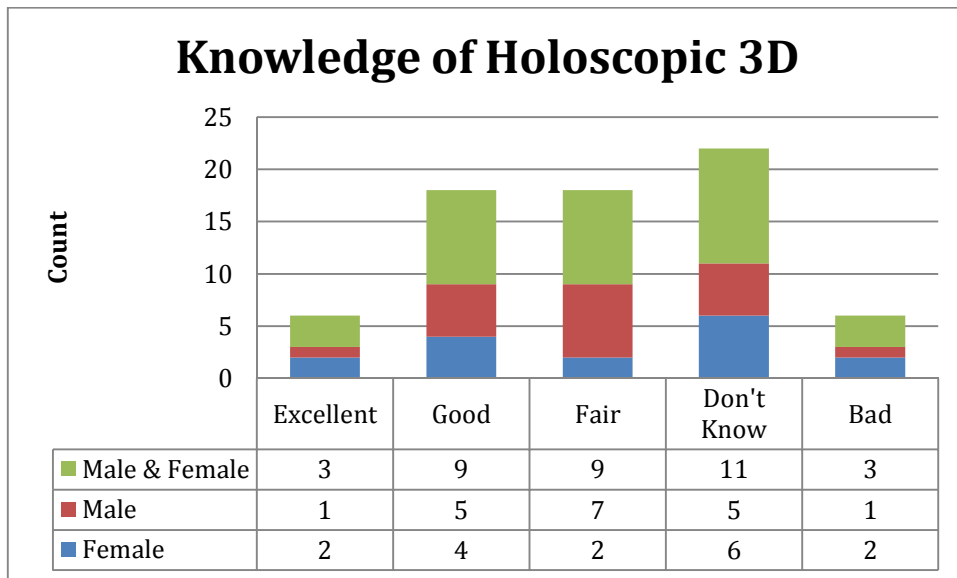


Fig 5. 21 : Knowledge of Holographic 3D in bar chart

**Algorithm 1: (Native VP refocusing)**

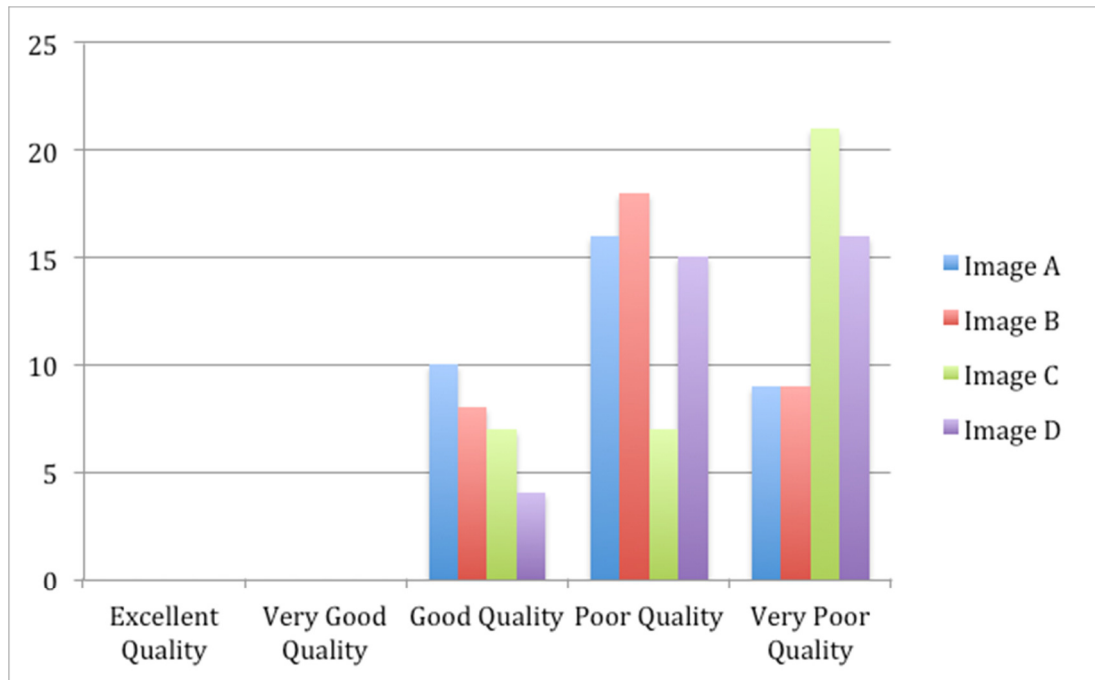


Fig 5. 22 : MOS values for image quality using Algorithm 1 represented in bar chart

Table 5. 6 : Mean, Standard Deviation for Algorithm 1 computed

Algorithm 1	Image A	Image B	Image C	Image D
Excellent Quality	0	0	0	0
Very Good Quality	0	0	0	0
Good Quality	10	8	7	4
Poor Quality	16	18	7	15
Very Poor Quality	9	9	21	16
<b>MOS</b>	<b>2.03</b>	<b>2</b>	<b>1.6</b>	<b>1.66</b>
Standard Deviation	2.302	2.06	2.77	2.208
95% Confidence Interval	0.762	0.682	0.917	0.731
Confidence interval outside -	1.267	1.317	0.682	0.928
Confidence interval outside +	2.792	2.682	2.517	2.391

Table 5. 7: Confidence interval for all the test images using Algorithm 1

Name	Confidence level	Confidence interval
Image A	95%	$(1.267 < \mu < 2.792) = 0.95$
Image B	95%	$(1.317 < \mu < 2.682) = 0.95$
Image C	95%	$(0.682 < \mu < 2.517) = 0.95$
Image D	95%	$(0.928 < \mu < 2.391) = 0.95$

Fig 5. 22 shows the MOS of image quality for algorithm 1 using all the test images. For each image, the MOS is computed and presented in both graph and tabular formats. Table 5. 6 shows the computed mean, standard deviation and confidence interval for each image rendered with algorithm 1. Table 5. 8 presents the MOS values, standard deviation and confidence interval associated with algorithm 2. Table 5. 10 and Table 5. 12 present the MOS, standard deviation, and confidence interval using all the test images for algorithms 3 and 4, respectively. Fig 5. 23, Fig 5. 24 and Fig 5. 25 show the MOS in bar chart format for algorithms 2, 3 and 4, respectively. These figures better visualise algorithms in terms of the image quality.

In image A, the confidence limits are  $2.03 \pm 1.96 \frac{2.302}{\sqrt{35}}$ , or  $2.03 \pm 0.762$ .

The 95% confidence interval for the population mean  $\mu$  lies between 1.267 and 2.792. This means that the probability that the population's mean image quality for algorithm 1 on image A lies between 1.267 and 2.792 is about 95% or 0.95. We can write this as  $p(1.267 < \mu < 2.792) = 0.95$ . In other words, we are 95% confident that the population's mean (or true mean) quality of image A using algorithm 1, lies between 1.267 and 2.792. The population's mean image quality associated with algorithm 1 for all the four test images are presented in Table 5.7. Table 5. 9, Table 5. 11 and Table 5. 13 show the computed 95% confidence intervals associated with the algorithms 2, 3, and 4, respectively.

**Algorithm 2: (Full resolution refocusing)**

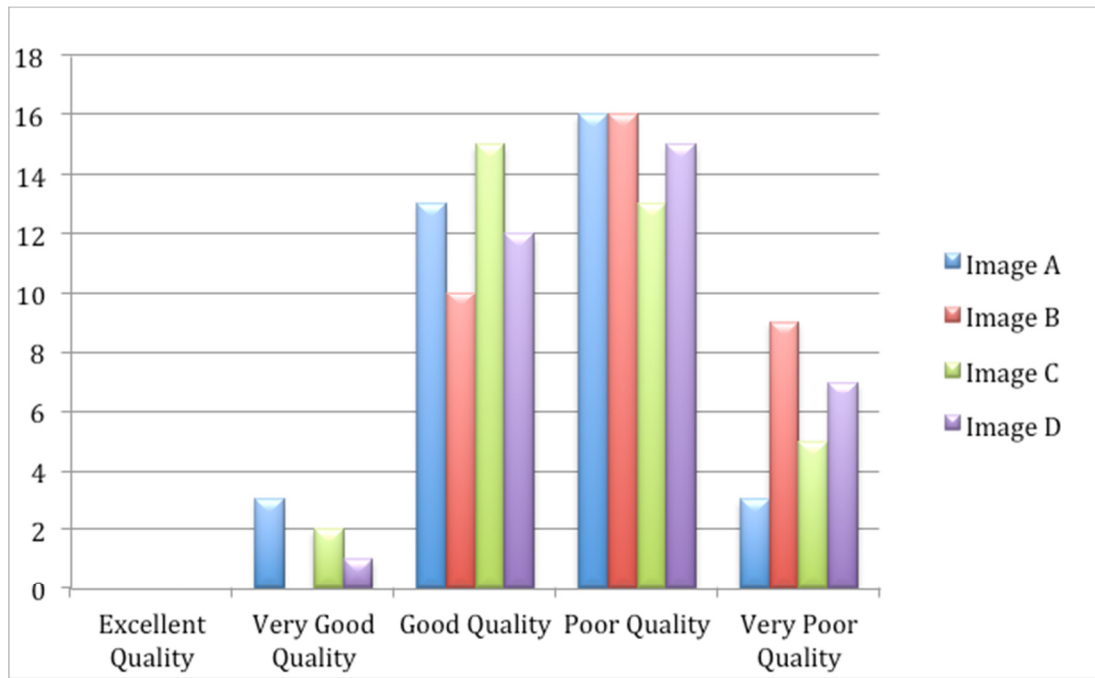


Fig 5. 23 : MOS values for image quality using Algorithm 2 represented in bar chart.

Table 5. 8 : Mean, Standard Deviation and Confidence interval for Algorithm 2 presented.

Algorithm 2	Image A	Image B	Image C	Image D
Excellent Quality	0	0	0	0
Very Good Quality	3	0	2	1
Good Quality	13	10	15	12
Poor Quality	16	16	13	15
Very Poor Quality	3	9	5	7
<b>MOS</b>	<b>2.46</b>	<b>2.03</b>	<b>2.40</b>	<b>2.20</b>
Standard Deviation	2.056	2.303	2.2181	2.268
95% Confidence Interval	0.681	0.762	0.734	0.751
Confidence interval outside -	1.778	1.267	1.665	1.448
Confidence interval outside +	3.141	2.792	3.134	2.951

Table 5. 9 : Confidence interval for algorithm 2 on all the test images.

Name	Confidence level	Confidence interval
Image A	95%	$(1.778 < \mu < 3.141) = 0.95$
Image B	95%	$(1.267 < \mu < 2.792) = 0.95$
Image C	95%	$(1.665 < \mu < 3.134) = 0.95$
Image D	95%	$(1.448 < \mu < 2.951) = 0.95$

**Algorithm 3: (Full resolution with blend)**

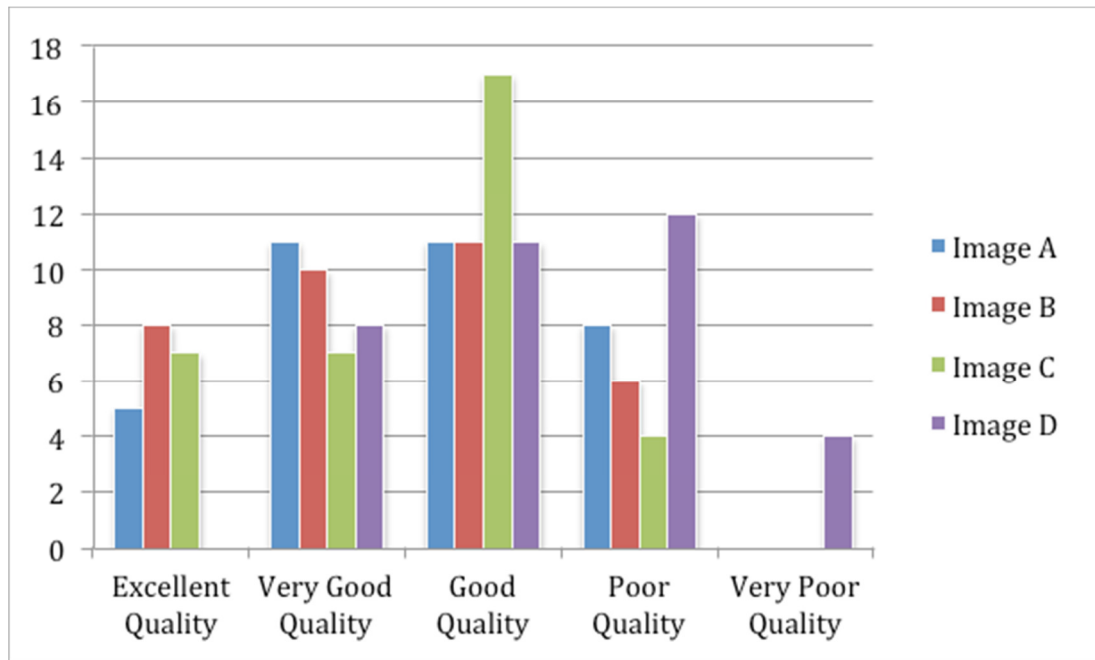


Fig 5. 24 :MOS values for image quality using Algorithm 3 represented in bar chart.

Table 5. 10: Mean, Standard Deviation, Confidence interval for algorithm 3 computed

Algorithm 3	Image A	Image B	Image C	Image D
Excellent Quality	5	8	7	0
Very Good Quality	11	10	7	8
Good Quality	11	11	17	11
Poor Quality	8	6	4	12
Very Poor Quality	0	0	0	4
<b>MOS</b>	<b>3.37</b>	<b>3.57</b>	<b>3.48</b>	<b>2.66</b>
Standard Deviation	2.72	2.495	2.86	2.63
95% Confidence Interval	0.901	0.826	0.947	0.871
Confidence interval outside -	2.468	2.745	2.538	1.788
Confidence interval outside +	4.271	4.398	4.433	3.531

Table 5. 11 : Confidence interval computed for algorithm 3

Name	Confidence level	Confidence interval
Image A	95%	$(2.468 < \mu < 4.271) = 0.95$
Image B	95%	$(2.745 < \mu < 4.398) = 0.95$
Image C	95%	$(2.538 < \mu < 4.433) = 0.95$
Image D	95%	$(1.788 < \mu < 3.531) = 0.95$

**Algorithm 4: (SPA with smart filters - (proposed method))**

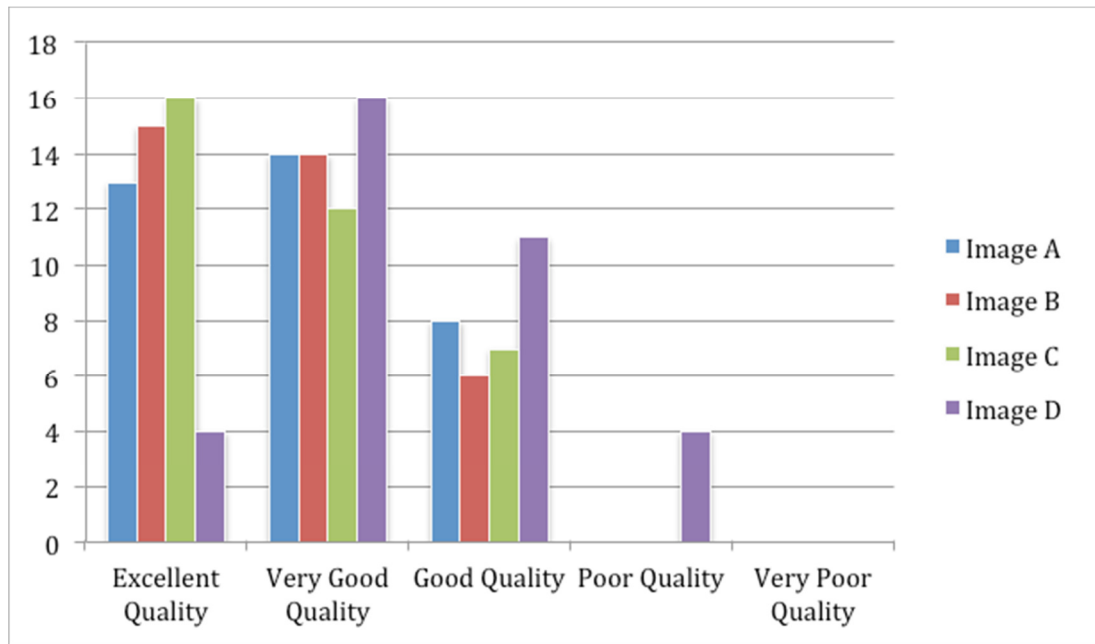


Fig 5. 25: MOS value for image quality on all the test images using Algorithm 4

Table 5. 12 : Mean, Standard deviation, Confidence interval for algorithm4 computed

Algorithm 4	Image A	Image B	Image C	Image D
Excellent Quality	13	15	16	4
Very Good Quality	14	14	12	16
Good Quality	8	6	7	11
Poor Quality	0	0	0	4
Very Poor Quality	0	0	0	0
<b>MOS</b>	<b>4.14</b>	<b>4.26</b>	<b>4.26</b>	<b>3.57</b>
Standard Deviation	2.49	2.39	2.6	2.16
95% Confidence Interval	0.824	0.791	0.861	0.715
Confidence interval outside -	3.315	3.468	3.398	2.854
Confidence interval outside +	4.964	5.051	5.121	4.285

Table 5. 13 : Confidence interval computed for algorithm 4 represented in tabular fashion

Name	Confidence level	Confidence interval
Image A	95%	$(3.315 < \mu < 4.964) = 0.95$
Image B	95%	$(3.468 < \mu < 5.051) = 0.95$
Image C	95%	$(3.398 < \mu < 5.121) = 0.95$
Image D	95%	$(2.854 < \mu < 4.285) = 0.95$

The confidence intervals are calculated for each image using one of the four algorithms. We can therefore calculate the probability that the population's mean image quality for each image is between the range of A and B, as shown above for each algorithm in tabular format. For more in-depth evaluation, we took a step further by averaging the MOS for all four images used by one particular algorithm, then calculating standard deviation for that particular algorithm as well as those for the other three. This is to calculate the confidence interval of image quality associated with each individual algorithm on all the test images in the study (See table 5.15).

MOSA is the mean score value associated with each algorithm as computed by averaging all the MOS for that particular algorithm on different test images (see eq (15)).

$$MOSA_j = \frac{\sum_{i=1}^{MOSn} MOS_j}{MOSn} \quad \text{eq (15)}$$

where MOS is the mean opinion score of the test image *i* for algorithm *j* and MOSn is the number of images. This formulae computes the mean opinion score for each individual algorithm on all test images amongst 35 participants. Standard deviation of each algorithm is computed using eq (16). Then the confidence intervals are computed for each algorithm.

$$M_j^2 = \frac{\sum (MOS_j - MOSA_j)^2}{MOSn - 1} \quad \text{eq (16)}$$

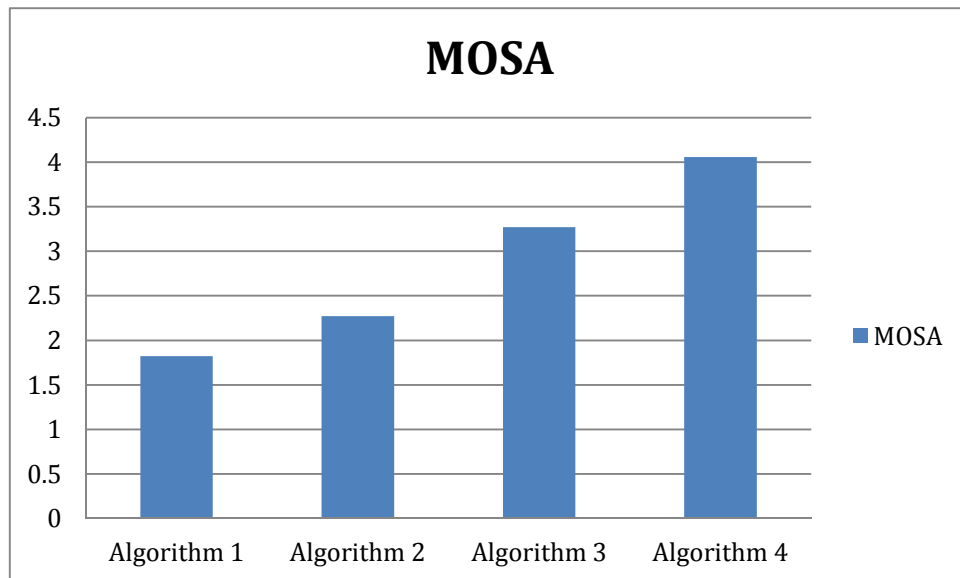
**Table 5. 14: MOSA, standard deviation and 95% confidence interval for four algorithms computed**

Name	MOS Algorithm 1	MOS Algorithm 2	MOS Algorithm 3	MOS Algorithm 4
Image A	2.03	2.46	3.37	4.14
Image B	2	2.03	3.57	4.26
Image C	1.6	2.4	3.48	4.26
Image D	1.66	2.2	2.66	3.57
MOSA	1.82	2.27	3.27	4.05
Standard Deviation	0.150	0.115	0.516	0.326
95% confidence Interval	0.049	0.038	0.171	0.108
Confidence Interval outside -	1.772	2.234	3.098	3.949
Confidence Interval outside +	1.872	2.310	3.441	4.165

**Table 5. 15 : Confidence interval for four algorithms**

Name	Confidence level	Confidence interval
Algorithm 1	95%	$(1.772 < \mu < 1.872) = 0.95$
Algorithm 2	95%	$(2.234 < \mu < 2.310) = 0.95$
Algorithm 3	95%	$(3.098 < \mu < 3.441) = 0.95$
Algorithm 4	95%	$(3.949 < \mu < 4.165) = 0.95$

The MOSA obtained from all four algorithms are presented in Table 5.14, where the algorithm 4(the proposed algorithm) scored the highest in image quality amongst the other algorithms that were used in the studies. The MOSA associated with the algorithm 4 lies between 3.949 and 4.165 with 95% confidence. Algorithm 1 scored the lowest in terms of image quality with confidence interval  $(1.772 < \mu < 1.872)$ . Fig 5. 26 shows the MOSA values in histogram format.



**Fig 5. 26 : MOSA values for image quality**



## **5.7 Conclusion**

In this chapter smart filters were discussed. Both pre-processing and post-processing stages were equally important in imaging systems, as pre-processing allows to enhance visual data whereas post-processing allows to do various adjustments to the final image to improve the quality and resolution.

Pre-processing played a very vital role at the beginning of the process, where it effectively removed any unnecessary noise on the HI. This was done with a simple but effective process, which suppresses the noise as well as keeping the details of the overall HI by avoiding the effect of over sharpening. This process does not put too much emphasis on the edges, making it easier for the human eyes to pick up, but it is far more similar to unsharpened masking. The results were shown in Fig 5. 5 to illustrate the effects when compared with the original HI. The overall structure of the processed HI in Fig 5. 5(b)-(e) was well-defined with high contrast that brought out every small detail, hence achieved a better result compared to the original. This made a prominent impact on refocused images, bringing out structural details on the focused regions of the image. Otherwise, small structural details were completely faded away with recursive or iterative shift and integration of different views.

Post-processing smoothing method is called “smoothing via  $L_0$  gradient minimisation”. This smoothing mechanism has effectively removed parts of the noise, unwanted details and even of slight blurriness that was introduced in refocusing stage. The final results of smart filtering was compared, in section 5.6, with those presented earlier in chapter 4. It clearly shows the improvement in terms of image quality and resolution. Also comprehensive experiments were carried and it defined the right weighting value for lambda, which was applied on refocused images to maintain a more photorealistic look, and successful achieving the high quality image without artifacts.

Also, the subjective image quality assessment experiments were conducted to evaluate the human perception for the quality of each image rendered under different refocusing algorithms. The work addressed image quality assessment on refocusing algorithms using the subjective test. Subjective results were

analysed to demonstrate the most effective refocusing algorithms for determining image quality in holographic 3D content. The results show that the output of the proposed method (SPA with smart filters) correlates strongly with overall viewer perception of image quality. The results also claim that the algorithm 1 scores the weakest in terms of subjective image quality assessment.

## **5.8 References**

- [1] S. Lertrattanapanich and N. Bose, "High resolution image formation from low resolution frames using Delaunay triangulation.," *Image Process. IEEE Trans.*, vol. 11, no. 12, pp. 1427–1441, Jan. 2002.
- [2] R. R. Schultz, L. Meng, and R. L. Stevenson, "Subpixel Motion Estimation for Super-Resolution Image Sequence Enhancement," *J. Vis. Commun. Image Represent.*, vol. 9, no. 1, pp. 38–50, Mar. 1998.
- [3] T. Georgiev and A. Lumsdaine, "Superresolution with plenoptic camera 2.0," *Adobe Syst. Inc. Tech. Rep*, vol. 2009, no. April, pp. 1–9, 2009.
- [4] T. Georgiev and A. Lumsdaine, "Reducing Plenoptic Camera Artifacts," *Comput. Graph. Forum*, vol. 29, no. 6, pp. 1955–1968, Sep. 2010.
- [5] R. Hummel, B. Kimia, and S. Zucker, "Deblurring gaussian blur," *Comput. Vision, Graph. Image Process.*, vol. 38, no. 1, pp. 66–80, 1987.
- [6] L. Xu, C. Lu, Y. Xu, and J. Jia, "Image smoothing via L0 gradient minimization," *Proc. 2011 SIGGRAPH Asia Conf. - SA '11*, vol. 30, no. 6, pp. 174–175, 2011.
- [7] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," *Comput. Vision*, 1998. Sixth ..., pp. 839–846, 1998.
- [8] F. Durand and J. Dorsey, "Fast bilateral filtering for the display of high-dynamic-range images," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 257–266, 2002.
- [9] S. Paris and F. Durand, "A fast approximation of the bilateral filter using a signal processing approach," *Comput. Vision–ECCV 2006*, vol. 2006, pp. 568–580, 2006.
- [10] Z. Farbman, R. Fattal, D. Lischinski, and R. Szeliski, "Edge-preserving decompositions for multi-scale tone and detail manipulation," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 67–77, Aug. 2008.
- [11] K. Subr, C. Soler, and F. Durand, "Edge-preserving multiscale image decomposition based on local extrema," *ACM Trans. Graph.*, vol. 28, no. 5, pp. 147–156, Dec. 2009.
- [12] M. Kass and J. Solomon, "Smoothed local histogram filters," *ACM Trans. Graph.*, vol. 29, no. 4, pp. 100–110, Jul. 2010.

- [13] B. Weiss, "Fast median and bilateral filtering," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 519–526, 2006.
- [14] J. Chen, S. Paris, and F. Durand, "Real-time edge-aware image processing with the bilateral grid," *ACM Trans. Graph.*, vol. 26, no. 3, pp. 103–112, 2007.
- [15] P. Choudhury and J. Tumblin, "The trilateral filter for high contrast images and meshes," *ACM SIGGRAPH 2005 Courses*, pp. 1–11, 2005.
- [16] R. Fattal, "Edge-avoiding wavelets and their applications," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 22–32, Jul. 2009.
- [17] Z. Farbman, R. Fattal, and D. Lischinski, "Diffusion maps for edge-aware image editing," *ACM SIGGRAPH Asia 2010 Pap. - SIGGRAPH ASIA '10*, vol. 29, no. 6, pp. 145–155, 2010.
- [18] J. Baek and D. E. Jacobs, "Accelerating spatially varying Gaussian filters," *ACM SIGGRAPH Asia 2010 Pap. - SIGGRAPH ASIA '10*, vol. 29, no. 6, pp. 169–178, 2010.
- [19] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *Pattern Anal. Mach. Intell. IEEE Trans.*, vol. 12, no. 7, pp. 629–639, Jul. 1990.
- [20] M. J. Black, G. Sapiro, D. H. Marimont, and D. Heeger, "Robust anisotropic diffusion," *Image Process. IEEE Trans.*, vol. 7, no. 3, pp. 421–432, Jan. 1998.
- [21] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," *ACM Trans. Graph.*, vol. 1, no. 212, pp. 309–314, 2004.
- [22] Y. Li, J. Sun, C. Tang, and H. Shum, "Lazy snapping," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 303–308, 2004.
- [23] J. Liu, J. Sun, and H.-Y. Shum, "Paint selection," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 69–76, Jul. 2009.
- [24] S. Maji, N. Vishnoi, and J. Malik, "Biased normalized cuts," *Comput. Vis. Pattern Recognit.*, pp. 2057–2064, Jun. 2011.
- [25] P. Arbelaez and M. Maire, "Contour detection and hierarchical image segmentation," *Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 898–916, May 2011.

- [26] A. Criminisi, T. Sharp, C. Rother, and P. P'erez, "Geodesic image and video editing," *ACM Trans. Graph.*, vol. 29, no. 5, pp. 134–149, Oct. 2010.
- [27] R. C. Gonzales and R. E. Woods, "Digital image processing, 1993." Addison-Wesley Publishing Company, pp. 196-199, 1993.
- [28] Y. Wang, J. Yang, W. Yin, and Y. Zhang, "A new alternating minimization algorithm for total variation image reconstruction," *SIAM J. Imaging Sci.*, vol. 1, no. 3, pp. 248–272, 2008.
- [29] A. George and A. Prabavathy, "A Survey On different approaches used in image quality assessment," *IJCSNS*, vol. 3, no. 2, 2014.
- [30] S. Bae, T. Pappas, and B. Juang, "Subjective Evaluation of Spatial Resolution and Quantization Noise Trade offs," *Image Process. IEEE*, vol. 18, no. 3, pp. 495–508, 2009.
- [31] P. Mohammadi, "Subjective and Objective Quality Assessment of Image: A Survey," *arXiv Prepr. arXiv*, no. June, pp. 1–50, 2014.
- [32] S. Grgić, M. Grgić, and B. Zovko-Cihlar, "Subjective Assessment of Picture Quality," *Dubrovnik*, no. 97, 1997.
- [33] J. Redi, H. Liu, H. Alers, R. Zunino, and I. Heynderickx, "Comparing subjective image quality measurement methods for the creation of public databases," in *IS&T/SPIE Electronic Imaging*, 2010, p. 752903.
- [34] I. Recommendation, "500-11,'Methodology for the Subjective Assessment of the Quality of Television Pictures,' Recommendation ITU-R BT. 500-11," *ITU Telecom. Stand. Sect. ITU*, 2002.
- [35] I. Recommendation, "P.910, 'Subjective video quality assessment methods for multimedia applications,' ITU-T Recommendation P.910," *Geneva, ITU*, 2008.
- [36] I. Recommendation, "BT.814-1, 'Specification and alignment procedures for setting of brightness and contrast of displays,' ITU-R Recommendation BT.814-1," *ITU*, 1994.
- [37] I. Recommendation, "BT.1129-2, 'Subjective assessment of standard definition digital television (SDTV),' ITU-R Recommendation BT.1129-2," *ITU*, 1998.
- [38] A. Lumsdaine and T. Georgiev, "Full resolution lightfield rendering," *Indiana Univ. Adobe Syst. Tech. Rep*, no. January, pp. 1–12, 2008.

- [39] T. Georgiev and A. Lumsdaine, "Focused plenoptic camera and rendering," *J. ...*, vol. 12, no. 2, pp. 1–28, 2010.
- [40] R. Ng, M. Levoy, M. Brédif, and G. Duval, "Light field photography with a hand-held plenoptic camera," *Comput. Sci. ...*, vol. 2, no. 11, pp. 1–11, 2005.

## **6. Chapter Six**

### **Integration and Object Segmentation**

This chapter presents the integration technique of different display technologies such as 2D, stereoscopic 3D and autostereoscopic 3D displays. All displays are put under test to see if the Holographic 3D content can be reformatted and replayed on various displays. As each display has different requirements to perform their optimal results, it is important to take the display specifications into consideration during rendering. We know that holographic 3D content contains a richer representation of the scene and with proposed (SPA with smart filters) method could render images with good resolution. Therefore, combining this method with the whole system of post-production of holographic 3D content rendering makes a complete package including the object segmentation. This will benefit content creators, as a single format of holographic 3D content is reformat-able to various other formats. The displays considered are listed below:

- LG 47 Inches, 2D
- LG 47 Inches (3D polarised stereoscopic)
- Alioscopy 47 Inches (Autostereoscopic Multiview 3D Display)

The contents are acquired using Arri ALEXA XT, which is modified by placing a microlens array in front of the image sensor. This camera captures video sequences, whereas in the previous chapters it was focused on still holographic 3D images.

Finally, object extraction is described using stereo pair images. The stereo pair images are rendered from Holographic 3D content and disparity map is calculated to extract objects at a particular distance within the captured scene. Detailed experiments are conducted to show the findings as well as the integration of different display technologies in this chapter.

## **6.1 Camera**

The first holographic 3D camera was assembled with a Canon 5D II camera, which was part of the 3D VIVANT project developments. It consists of an adapter that contains a microlens array to simulate the fly's eyes and a relay lens to relay the holographic 3D image onto the sensor. This included optical distortions such as barrel effect, which was discussed in chapter 3. This design was then extended to accommodate the Alexa camera.

The first version is a modular solution with a holographic camera adapter for Alexa [7]. This component is placed at an intermediate position between camera body and objective lens. It contains the microlens array, which generates a plenoptic replica of the scene. The manipulation of the lens is different. Focus setting is no longer a matter of setting the focus dial to the correct value. The focus dial is to be kept in a predefined position rather than pulling the focus while scene objects move. Also the aperture setting is kept in a fixed and predefined position, which is given by specifications of the microlens array. This type of holographic 3D camera is flexible. It is very easy to replace the holographic tube and the front lens in order to generate different styles of holographic 3D images (number of views, depth budget, interocular distance, re-focusable range, etc.).





**Fig 6. 1 : (a) First version of holoscopic adapter for Alexa camera. (b) A holoscopic 3D camera prototype based on Canon 5D MKII [7].**

The first version of Alexa camera images were captured at a resolution of 2048 x 1080 pixels and the results are shown in this chapter to see the effect that they may have caused on the image rendering as well as on the image quality. Fig 6. 2 shows a frame of the sequence recorded with the Alexa camera using the adapter with master prime 50mm lens.

Second version of the holoscopic 3D camera images is fully integrated [7]. The microlens array is mounted in closer proximity to the sensor. Hence the physical dimensions of the camera are almost identical with regular camera. Specifications and operation of the camera are similar to the first version and the adapter. The optical quality, however, is greatly improved. As the design gets along without a relay lens, vignetting and distortions are reduced. The fixed mounting position (factory setting) leads to tighter tolerances for positional parameters of the microlens array. Therefore, the calibration of the system is easier and more durable. The second version, namely, Alexa XT camera, captures at resolutions of 3424 x 2202 pixels resulting in more microlenses, which makes the final rendered frames at a higher resolution than it does in the previous version. It is important to point out here that the main lens image plane is formed in front of microlens arrays allowing it to capture a portion of the scene from different perspectives.

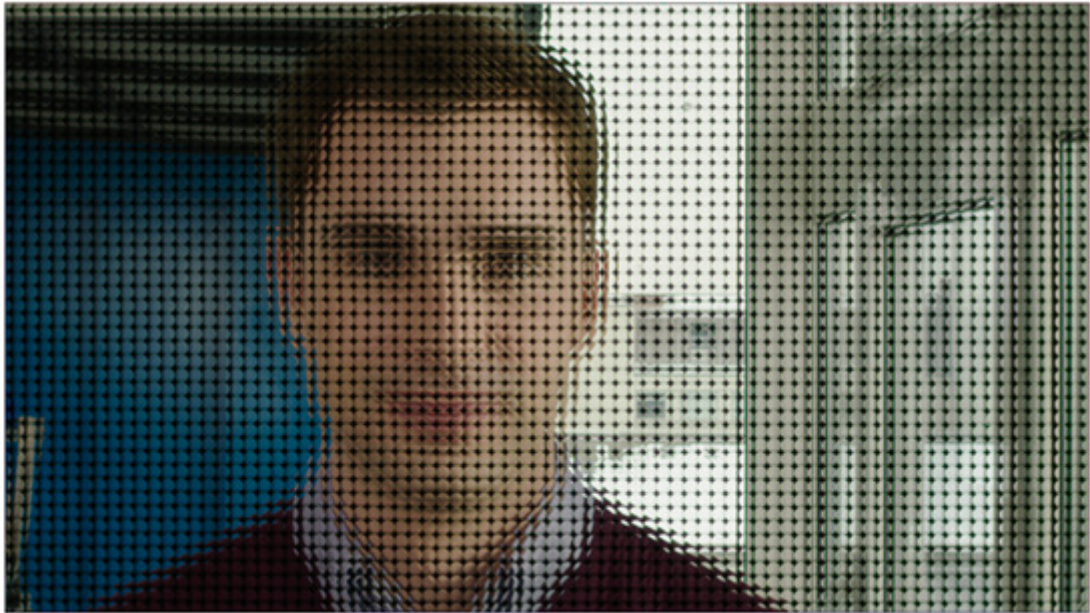


Fig 6. 2 : Sample of acquired holographic 3D sequence from Alexa camera at resolution of 2880 x 1620 pixels.

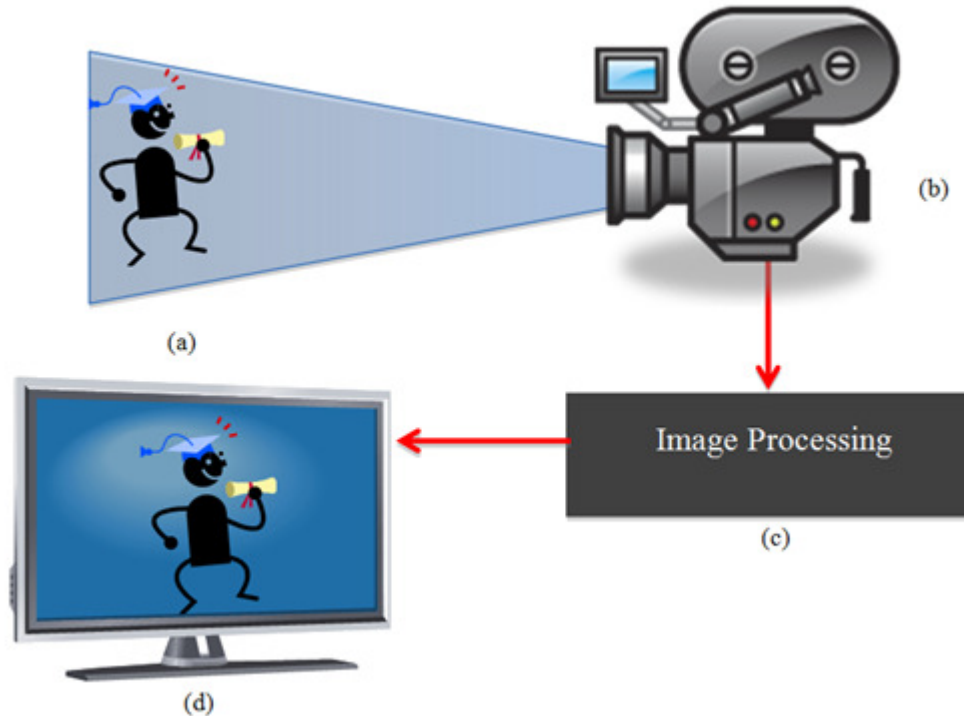


Fig 6. 3 : Alexa XT camera with microlens built close to the image sensor and capture frames at resolution of 3424 x 2202 pixels [7].

### 6.1.1 2D Display

LG 47 inches 2D TV is selected to illustrate the integration process from holographic 3D capture to 2D display. For the purpose of simplicity, the workflow is graphically illustrated in Fig 6. 4, which involves capturing, processing and visualisation. Once the content is captured with holographic 3D

camera, the acquired holographic 3D content is converted into 2D format that is replayed on 2D displays.



**Fig 6. 4 : The workflow block diagram. (a) Scene capture with (b) holoscopic camera. (c) Frames are rendered in post-production and playing on 2D display (d).**

In the capturing stage, both position and angular information of the scene is recorded in the image sensor. This allows users to change the focus after the image is acquired in the post-production stage. It can also extract the depth of field without decreasing the aperture.

Rendering process involves up-sampling, shift and integration with smart filter to extract only one 2D view from the record content to be replayed on LG TV. It is worth mentioning that each element image (EI) is at resolution of  $37 \times 37$  pixels, which is due to the microlens pitch being equal to  $250\mu\text{m}$ . Therefore the number of microlenses involved in the capturing stage is equal to the total number of pixels along horizontal and vertical directions divided by the resolution of EI. The total number of EIs along horizontal and vertical directions are equal to 92 and 59, respectively.

To extract a single view, it is vital to define the size of sub-image (SI) under each microlens, because a different size of SI determines a different image plane as discussed earlier in chapter 4. The pickup position of SI under a microlens is another essential element that needs to be considered. This is because extracting SI from the corners of EI makes the final image to be darker and noisier due to the circular aperture of the camera, while the microlenses are square (see Fig 6. 5). Increasing the circular aperture of the main lens will result in an over exposure of light to its neighbouring microlenses. Decreasing the aperture, on the other hand, will increase the dark areas in the corners of the every EI by allowing less light through. Therefore, keeping the acceptable aperture is ideal in the capturing stage, as shown in Fig 6. 5(b). Also it is important to avoid the corners of the EI during rendering, which will have a greater impact on the final rendered image.

As shown in Fig 6. 5(b), each EI is inverted by 180 degrees, which is due to the main lens image plane set in front of the microlens array. Therefore, EIs are rotated by 180 degrees before up-sampling, shift and integration process. This accurately integrates all the rays across other EI at particular depth given the SI size. The results are shown in Fig 6. 6, where  $SI = 10$ , number of different views of  $8 \times 8$  are used, pickup position along  $j = 8$  and  $i = 8$ ,  $\sigma = 6$ , kernel =  $[10 \times 10]$  and  $\lambda = 0.005$ . Fig 6. 6 shows the final rendered image, which is achieved by applying pre-processing, high resolution refocusing and post-processing, smart filters.

### **6.1.2 Experimental Results and Observations**

In pre-processing stage, an effective and efficient technique is applied to remove unnecessary noises that may affect the image quality in the rendering process. Then, high resolution refocusing technique is performed on the pre-processed frame to extract one view from location  $j=8$  and  $i = 8$  under every EI image with  $SI = 10$  as mentioned above and will remain the same throughout the whole sequence. Finally post-processing smoothing mechanism is applied to remove the noises, which are introduced in high resolution refocusing. During this smoothing process, edges are also preserved and efficiently enhanced. All the sequence frames get applied the same process one after the other,

converting frame sequences to video, while at the same time they are being processed.

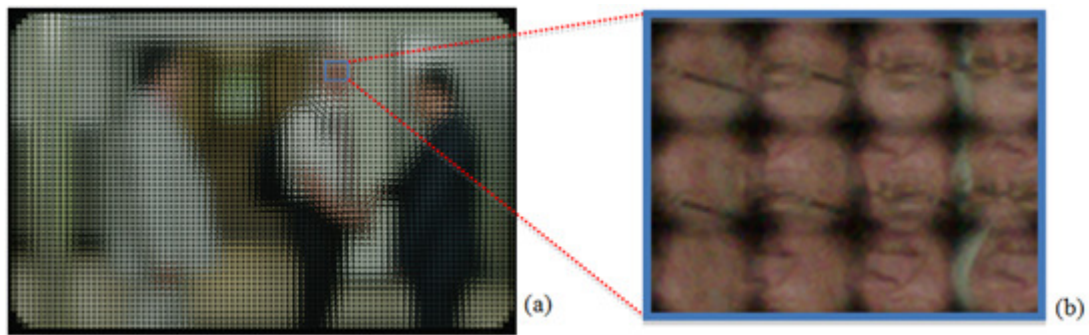


Fig 6. 5 : (a) A frame sample of a sequence, (b) the magnified version.



Fig 6. 6 : (a) 2D result of rendered frame no.14 and (b) rendered frame no.18.

After successfully converting 2D rendered sequence to video, it is then sent to the appropriate display. The processed video sequence is replayed on the LG TV using a movie player to show whether the holographic content can deliver 2D videos, as there are so many researchers who have only attempted to play still images but not video. Therefore, this is the first experiment to successfully capture, process and display holographic 3D video.

## 6.2 3D Stereoscopic Display

In this section, a single aperture holographic 3D camera is used to generate high-resolution stereoscopic 3D content. Using a single aperture camera in stereoscopic 3D production will reduce the complexity of dual cameras calibration. There has been intensive research in reducing complexity of stereoscopic 3D systems, yet there is not a successful method that overcomes

this challenge. In this section, the generation of stereoscopic 3D content from holographic 3D content is investigated. The results are replayed on LG 40 inches stereoscopic 3D display and the image quality and depth perception are analysed.

The steps involved in rendering stereoscopic from 3D holographic are explained in this section and also, perception of depth is analysed with anaglyph visualisation tool to give a detailed analysis in generating better stereo. The same holographic 3D content of the 2D content is used here for rendering stereoscopic 3D content. As mentioned earlier it has the capability of capturing the whole scene in true 3D in 2D format as shown in Fig 6. 5.

In rendering stage, the method relies on up-sampling, shift and integration of viewpoints with a smart filter to extract only one view of stereo. In order to extract the second viewpoint from the same frame, a different position along x-axis within the same EI is selected. The distance of a second view from first view is directly related to the baseline in stereoscopic 3D production, whereas this can be controlled by selecting different starting points on the x-axis during the rendering process. However, the baseline is fixed in stereoscopic 3D production. Also if the baseline distance is incorrect during the capture, then the whole scene needs to be shot again; otherwise there is not a correct depth information. This is expensive and time consuming, but this issue is reduced with holographic 3D camera.

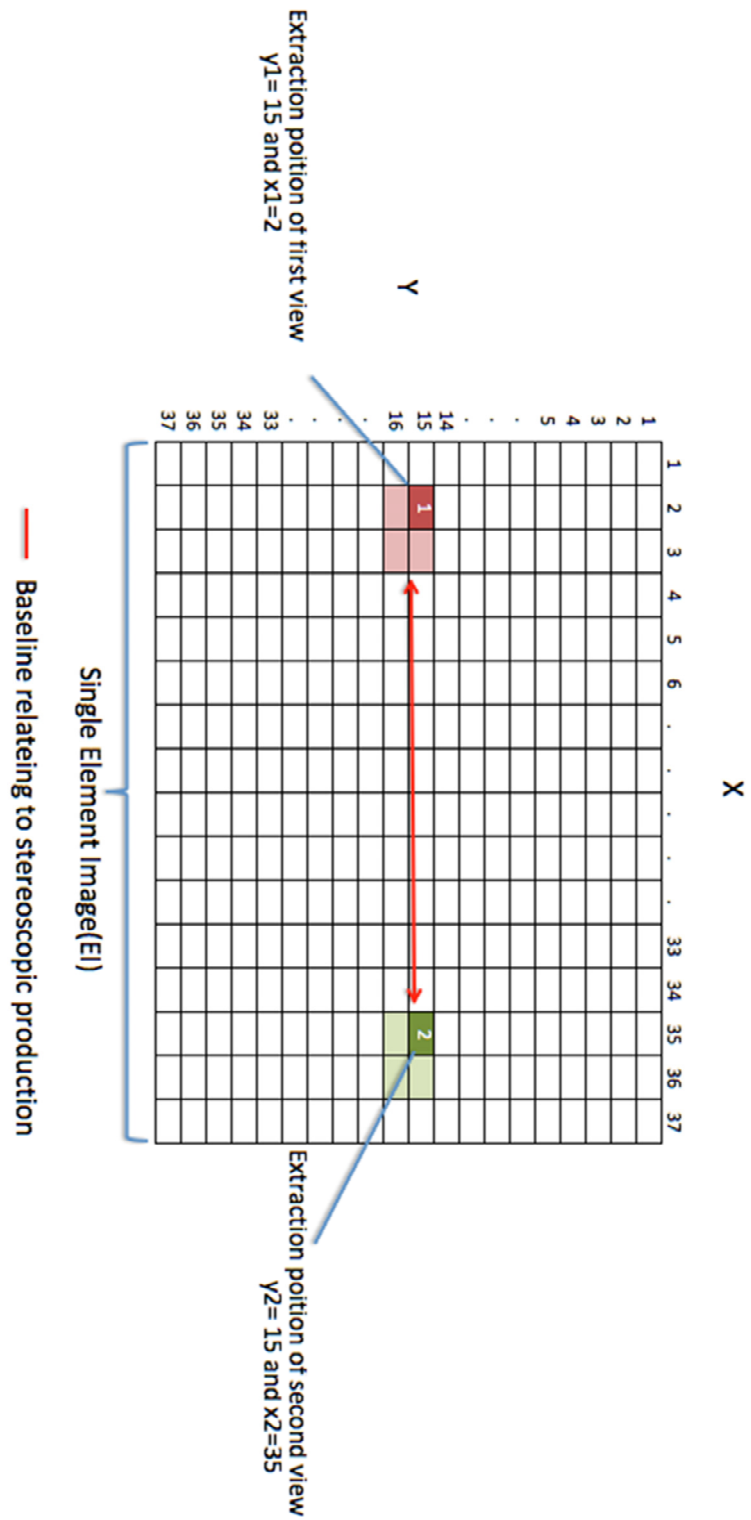


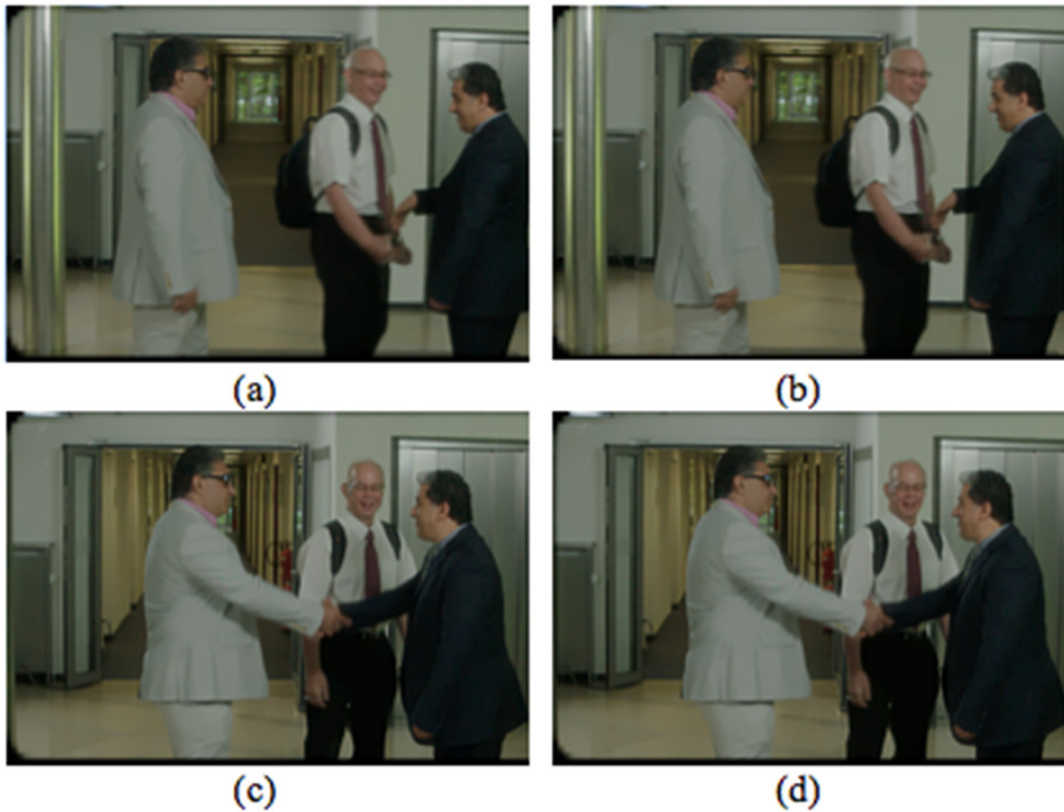
Fig 6. 7 : Illustration of baseline distance in single Element Image (EI).

For instance, the element images have resolution of 37 by 37 pixels and 92 by 59 element images in total. To extract view one, the position of x-axis ( $x_1$ ) and y-axis ( $y_1$ ) inside the element image are 4 and 15, respectively. On the other hand, the second view of y-axis ( $y_2$ ) stays constant, because in stereoscopic 3D capturing both cameras are setup side by side in the straight line along y-axis. This is why rendering the second view from holographic 3D content requires us to set the position of  $y_2 = y_1$  to keep in the same line along y direction. But the x-axis ( $x_2$ ) in the second view is changed to 35 depending on the suitability of the scene and  $SI = 2$ , as shown in Fig 6. 7.

### **6.2.1 Experimental Results and Observations**

Stereoscopic 3D content creation from holographic 3D content is discussed as shown in Fig 6. 8. The parameters remain the same as 2D image creation in section 6.1.1 . However, the position of x-axis and y-axis are different here. In the first view, position of x-axis and y-axis in Fig 6. 8(a) equals to 4 and 15, respectively. In second view, however, position of x-axis and y-axis are 24 and 15, respectively, as shown in Fig 6. 8(b). In this experiment the parameters are fixed throughout, to simplify the rendering process.





**Fig 6. 8 : Rendered sequences - (a) is the first rendered frame no. 3 of the sequence and (b) is the second view. (c) is the first view and (d) is the second view of the rendered frame no. 18<sup>th</sup> of the sequence.**



**Fig 6. 9 : Stereo 3D resulting images in anaglyph view. (a) Result of Fig 6.8(a)-(b) and (b) is result of Fig 6.8(c)-(d).**

The first view in Fig 6. 8(a) and (c) represents as left frame and the other two (b) and (d) as the right frame in stereoscopic 3D concept. Achieved results offer reasonably good image quality as well as stereo 3D depth parallax (?). This experiment confirms that initial stereoscopic 3D production requirement is met by holographic 3D camera, which is clearly the simplest and most efficient way of

capturing 3D content compared to any other 3D acquisition systems. Having said this, holographic 3D camera is still in its initial stage compared to Stereoscopic 3D cameras in the context of acquiring depth perception. This is due to wider ocular distance in stereoscopic production that increases the 3D depth.



**Fig 6. 10 : Stereo 3D image pair rendered from a holographic 3D image**

The rendered stereoscopic 3D image is shown in Fig 6. 10, which is replayed on the LG 3D TV. This experiment aims to exhibit the integration of stereoscopic 3D technology with holographic 3D imaging technology which confirms holographic 3D capturing that can serve the desire for a single capturing device of the future with further improved optical parameter, such as wider ocular distance and bigger image sensor.

### **6.2.2 Analysis and Discussion**

The Canon 5D Mark II camera has a bigger image sensor, capturing at resolution of 5K, enabling to see the impact of depth perception on final rendered images. The highest motion captures resolution out in commercial sector at present is 4K, which recently came to attraction. It has also been known that NHK from Japan manufactured 8k motion capturing camera as well as an 8k display and presented them in IBC 2012. This experiment shows that in the near future, with image sensor vastly becoming bigger and storage cheaper, it could be the most desirable solution. This may affect the stereo images generated from holographic capturing using the 5K resolution camera image. It is because this approach is very clearly focused at simple and efficient way of capturing real 3D content.

Notice that in Fig 6. 11 the EIs do not hold darker areas around corners. This is due to the square aperture designed for the camera to take full use of the sensor without covering any areas in EIs [7]. In this case, widely apart views can be used to give a wider ocular distance in stereo generation. The microlens array used is 250 $\mu$ m capturing resolution of 80 x 80 pixels containing 68 by 44 EIs in total. Hence, the same steps are taken in rendering the stereo views from Fig 6. 11.

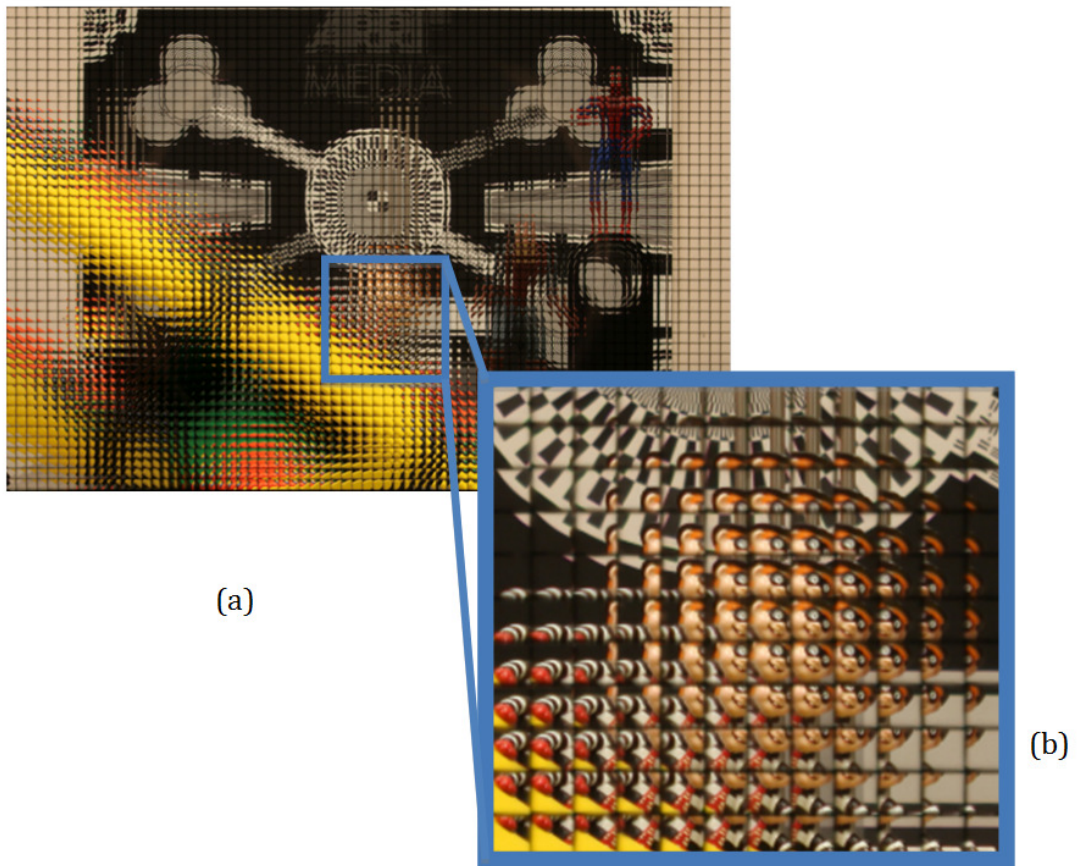
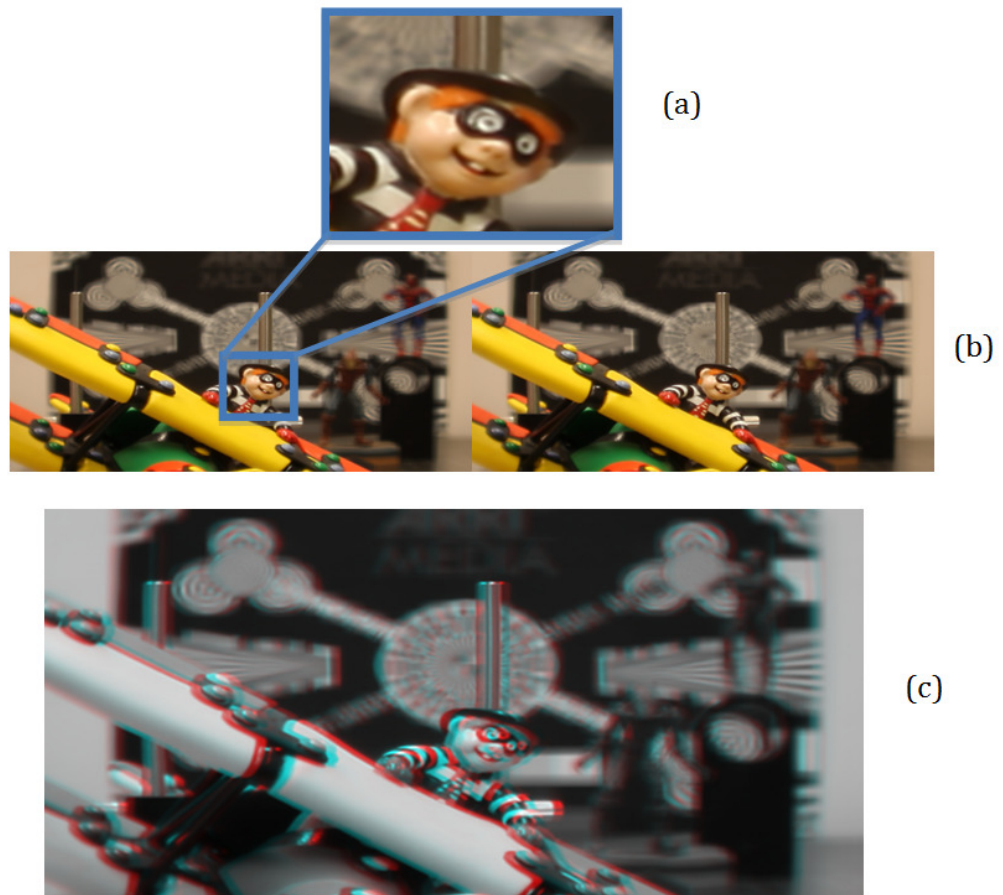


Fig 6. 11 : (a) shows the captured image from holographic camera at resolution of 5466 x 3588. (b) Shows a small portion of (a) high lighting it with blue square around it.

### 6.2.3 Experimental Results and Observations

Results of stereo 3D are shown in Fig 6. 12 with parameters SI= 12, x-axis and y-axis of first view equal to 2 and 35, respectively. While in the second view the x-axis is 42 and y-axis is 35. In both views 24 views are used in up-sampling, shift and integration processing in order to digitally put the image plane on the foreground object and this can be observed from the results. The final rendered

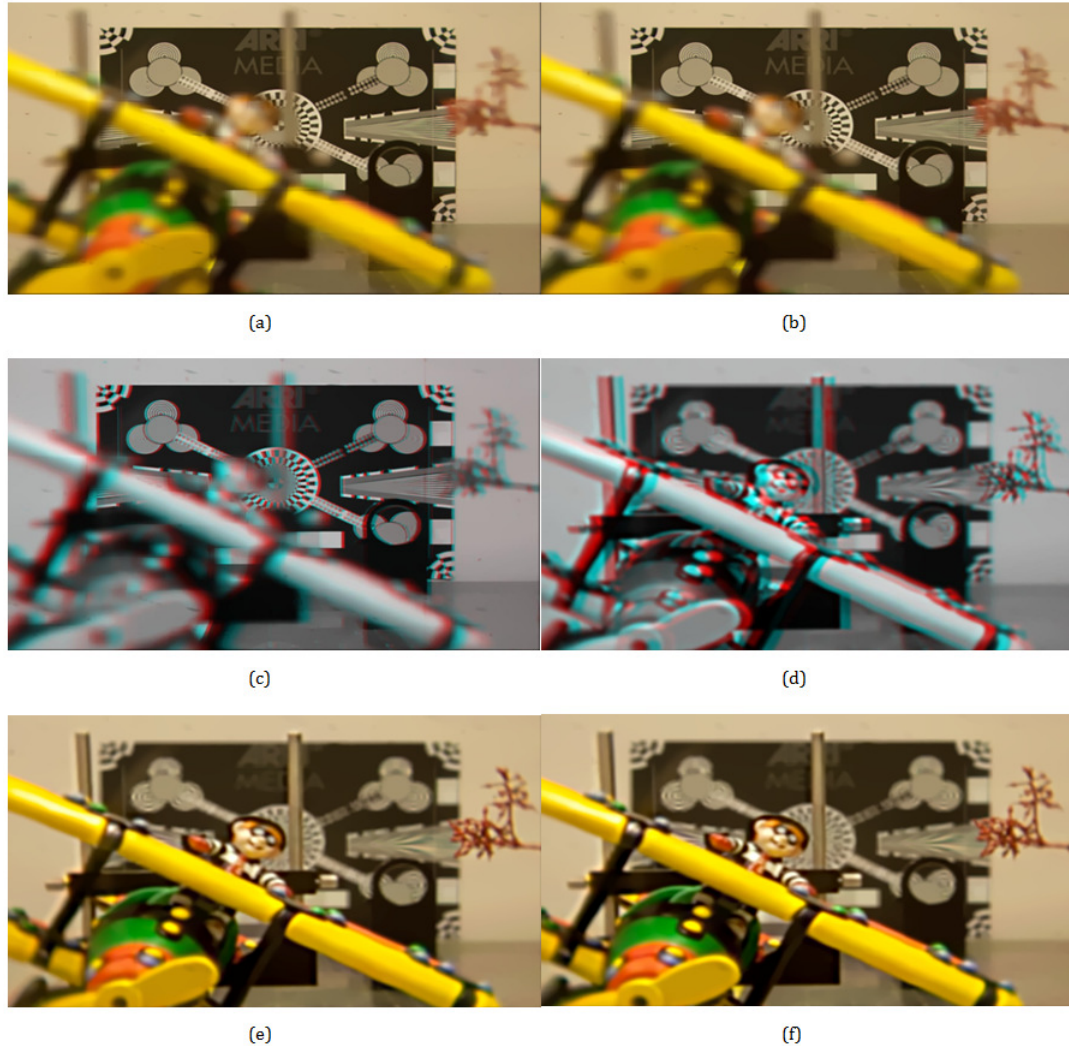
results are observed with richer and better contrast as shown in Fig 6. 12(a). Every small detail of the object can visually be seen with the naked eyes. The anaglyph viewing is also shown to point out the effect of 3D in Fig 6. 12(c). A better 3D effect is observed when analysing with anaglyph glasses as well as with polarised 3D.



**Fig 6. 12 :** The result of stereo 3D image from holographic content. (a) Crop area of the first view. (b) Shows both stereo views and (c) is the anaglyph result of the both view in (b).

Image of chapter 5 are also used in this experiment to show result of stereo 3D with two focuses, first focused at the background and the second at the foreground. The parameters used in extracting the two views of Fig 6. 13(a)-(b) are  $SI = 2$ ,  $\delta = 6$  and 7 by 7 different views with ( $\lambda=0.007$ ) as mentioned earlier in chapter 5, but the only difference is the x-axis and y-axis of both views. There is 9-pixel differences in between the first and second views. In the first view, x-axis and y-axis are 3 and 10, whereas they are 12 and 10 at the second view. Fig

6. 13 (b) shows the anaglyph results of (a)-(b) with better 3D depth perception. In Fig 6. 13(e)-(f) image plane is set on the foreground and results of the two views are presented in anaglyph in Fig 6. 13 (d).



**Fig 6. 13 : The rendered stereo 3D images from holographic 3D images with (a)-(b) focusing at background and (e)-(d) at foreground object. (c)-(d) are the results of both in anaglyph.**

After observing the results, the bigger image sensor offers better result compared to the result achieved in Fig 6. 9; because the depth perception is more visible in Fig 6. 13, as shown in 3D anaglyph presentation. The quality of results achieved is satisfying both in 2D as well as in 3D in Fig 6. 13. This comes to show that with bigger image sensor, as it may become the future in motion picture capturing, the future of stereoscopic 3D capturing may become the technique of the past with holographic 3D capturing [7]. This approach is very clearly focused at simple and efficient way of capturing real 3D content without

having to go through the complex dual camera calibration that is in stereoscopic production. The rendered images or videos are successfully displayed on LG 3D TV to demonstrate the capabilities of holographic content in integrating the existing technologies as one system [7].

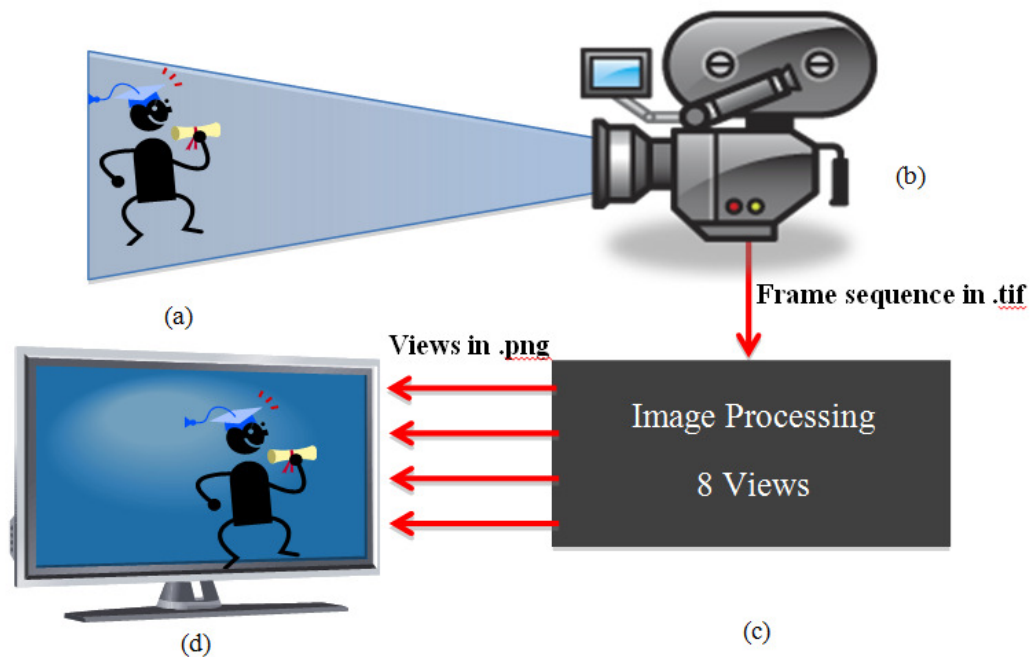
### **6.3 Autostereoscopic 3D Display**

Alioscopy developed a multiview 3D display that shows 3D content without glasses within 100-degree viewing angle and 30-degree viewing zone. The user can freely walk around the display within the 30-degree viewing zone undisturbed 3D content without jumping views; however, in 100-degree viewing angle the 3D content is visible but views are observed with jumping effect. The 3D display has its own 3D mixer player that requires 8 different perceptive views of the same scene in order to correctly display the content in 3D. Furthermore, the viewers are not required to wear any special glasses to experience 3d depth.



**Fig 6. 14 : Display size 47inches multiview display from Alioscopy**

As mentioned above, it is expensive to capture images even with dual camera. Now, taking it further to capture with 8 cameras will not only increase the cost widely, but also involve complicated calibration of 8 cameras. Furthermore, the multi-camera setup is not very mobile in comparison to stereoscopic production, which makes it very difficult for content producers. Therefore, holographic capturing is proposed in generating 8 views for such systems without having the need of complex and expensive multi camera calibration to produce content. This approach will show the capability of delivering content on motion pictures for multiview display. Also, this experiment will be the first of its kind in attempting to generate multi views from holographic camera on video. Fig 6. 15 shows the workflow from capturing to display.



**Fig 6. 15 : The workflow graphically. (a) Scene capture with (b) holoscopic camera. (c) Frames are rendered in post-production and playing on 2D display (d).**

The capturing process described earlier in section 6.1.1 is used in generating 2D as well as 3D stereoscopic views from holographic 3D capturing. Hence, the same holographic 3D content is chosen to produce content for multiview display, in

order to show that a single exposure of the scene with holoscopic camera can deliver content for multiple display technologies out in the market.

Note that the rendering process that takes the same steps in generating stereo view in section 6.2 but with more than two views are required in here. Therefore, different perception views from holoscopic 3D content are rendered using up-sampling, shift and integration approach and displayed on multiview screen using its player. Multiview 3D display used here does not necessarily requires the views to have large baseline; otherwise, it would have been difficult to produce content knowing that the baseline is limited at this stage. This is due to small image sensor size that limits the use of larger microlens with wide viewing angle. The rendering process is explained in example below where only 20 pixels are considered under each EIs for the sake of simplicity and also shown graphically in Fig 6.16.

**Example:** element images (EIs) are of resolution 20 by 20 pixels and 6 views are extracted from each EI. Notice that the position of view one on x-axis ( $x_1$ ) and y-axis ( $y_1$ ) inside the element image are 3 and 10, respectively. On the other hand, the view two of y-axis ( $y_2$ ) stays constant as well as all that of the other views, which will avoid misalignment of views projection causing bad 3D effect when it is displayed. The x-axis position is different for each view, but have the same distance from each other as shown in Fig 6.16. The difference between view one pixel 1 to view two pixel 1 are two pixels, where it is the same with view two with view three and the others too. The positions of views are represented with different colors in Fig 6.16, each color presenting one view from different perspective, using up-sampling, shift and integration process.



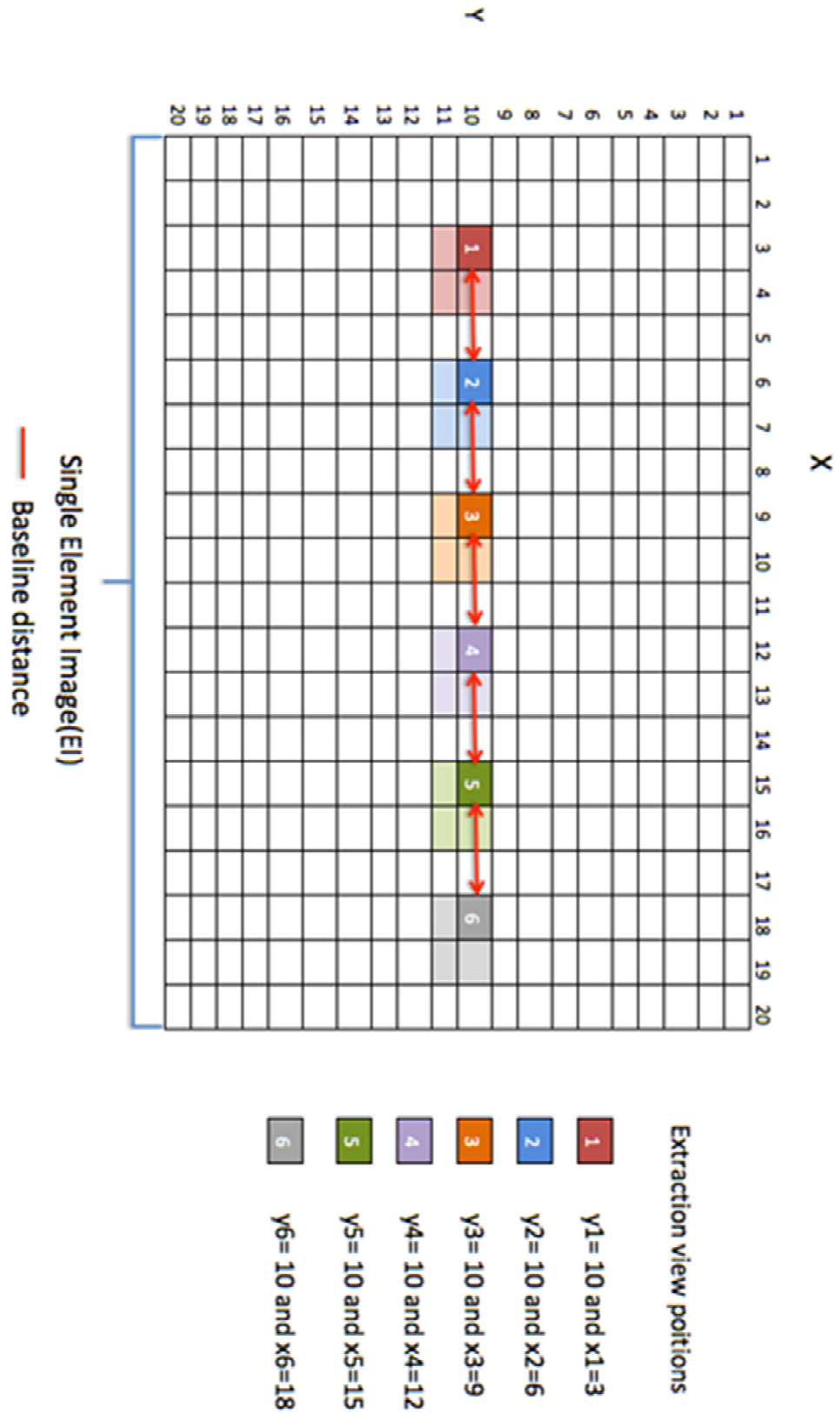


Fig 6.16 : Single EI with position of 6 views

### 6.3.1 Experimental Results and Observations

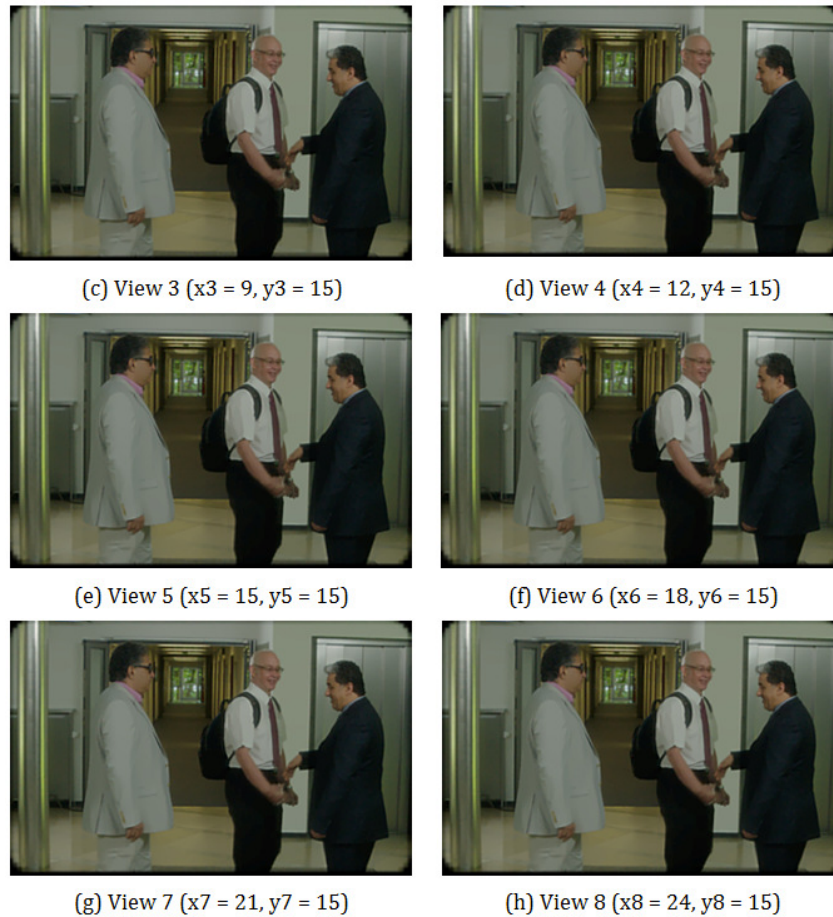
Results of 8 views from holographic 3D image are shown in fig 6.17. Each view is kept at the same distance from each other during rendering. In pre-processing, each frame is processed with  $\sigma = 6$ , kernel = [10x10], then up-sampling, shift and integration process is carried out with SI=10 and 8x8 number of different views. Finally, post-processing is performed to remove any blur in the views and also noises with  $\lambda = 0.005$ . Also note here that parameters are set constant throughout as well as in the 8 extracted views from each frame. Differences between the views are 3 pixels as it is shown below in Fig 6. 17 with each views' axis position under each EI.



(a) View 1 (  $x_1 = 3, y_1 = 15$  )



(b) View 2 (  $x_2 = 6, y_2 = 15$  )



**Fig 6. 17 : Illustration of 8 extracted views from single holographic frame no 3.**

Finally, all the rendered frames of 8 views are replayed on the multiview 3D display from Alioscopy using their own players to process and display in 3D. The results are shown in Fig 6. 18 taking picture of the display from different viewing points to show motion parallax. Looking at the depth of the scene, it seems all the objects are inside the screen, feeling like looking outside through a window. This is because in the holographic camera the image plane of the main lens is in front of the micro-lens array, where each microlens is rotated by 180 degrees on this centre axis. Therefore, the depth is observed inside the screen. The depth can be outside the screen with the image plane of the main lens in the capturing stages. During capturing, the image plane of the main lens needs to be behind microlens array, creating virtual image plane behind the micro-lens array. This is also tested here with Canon camera images while results are played on the display to observe the difference in depth. Result in Fig 6. 19

shows the scene depth as outside of screen giving more realistic 3D effect as the objects in scene move closer to the observer.



**Fig 6. 18 : Rendered multiview 3D content displayed on Alioscopy multiview screen.**



Fig 6. 19a : Results are demonstrated on the multiview display when focusing on the background



Fig 6. 19b : demonstrates the output results of holographic content on multiview display with focusing digital set to foreground object.

#### 6.4 Object Extraction Based On Depth map

Research on depth extraction from multi-view imaging systems has been extensive. However, the depth extraction from 3D holographic imaging systems is still in its infancy. Recent developments have been carried out in the past few years, where the number of depth extraction algorithms have been developed and compared to existing methods [1][2][3][4][8][9].

One of the methods is based on energy minimisation problem that seeks a pixel disparity map between sub-images(SI). The minimisation is accomplished using the graph cuts approach. It enables one to extract small set of points for which the depth is estimated with high accuracy. This sets of points are used to pose constraints to the minimisation problem that lead to more accurate estimation

of the overall depth[1][3]. However, this method requires binary mask that is manually created for each scene whose depth is accurately estimated. This approach may not be very practical in real life applications where for each image, a binary mask is necessary to be generated through depth map.

In another method the depth is obtained by viewpoint image extraction and a hybrid algorithm combining both multi-baseline and neighborhood constraint and relaxation techniques with feature block pre-selection in disparity analysis. This method is based on the distribution of the sample variance in sub-dividing non-overlapping blocks [4]. This algorithm is very time consuming due to its complexity in calculating depth information from numerous viewpoints when compared to generating depth map from stereo. Depth map results from stereo are to acceptable standard with few flitter process, which then can be used for object extraction in post-production.

## **6.5 Proposed holographic 3D object segmentation**

A new approach is proposed to extract object base on simple stereo geometry that requires two views for disparity estimation. The two views are generated from holographic content with wider baseline as mentioned earlier in section 6.2. It involves pre-processing, up-sampling, shift and integration and finally the post-processing. Furthermore, the two views are used to calculate disparity that is required to extract the desired object from the scene.

In order to extract a desired object from the holographic content, the right size of SI needs to be considered during the stereo rendering process; because the size of SI determines the image plane in the real world. Therefore, It is important to keep the desired object in-focused in stereo views, which leads to precise disparity estimation and also the extraction of that particular object being in-focused on the final image.

Firstly, two perceptive views are generated from holographic content. Then, disparity of the two views is calculated. For the sake of simplicity and effectiveness, a correlation-based block-matching method is used here[4]. The basic idea of the block-matching method is to locate a candidate block in the second view that can be the best match in the first target view. Two views,  $V_1$

and  $V_2$ , are used;  $(x,y)$  are the coordinates of the point being analyzed.  $V_1(x,y)$  denotes the intensity of the point  $(x,y)$ ,  $w$  denotes the local block size used in matching,  $d$  is the disparity and  $R$  is the search range in the second view associated with the target view. The correlation matching criteria used is sum of the square difference(SSD), as this is more effective compared with sum of absolute difference (SAD) and cross correlation (CC), used in [6]. The SSD function can be mathematically described as :

$$SSD(R) = \sum_{x,y \in w} [V_1(x,y) - V_2(x + R,y)]^2 \quad \text{eq( 6.1)}$$

In general, the algorithm can be mathematically described as finding out the best matching position  $(x+R,y)$  in the second image where SSD(R) function has minimum;

$$d = \arg\{\min\{SSD(R)\}\} \quad \text{eq (6.2)}$$

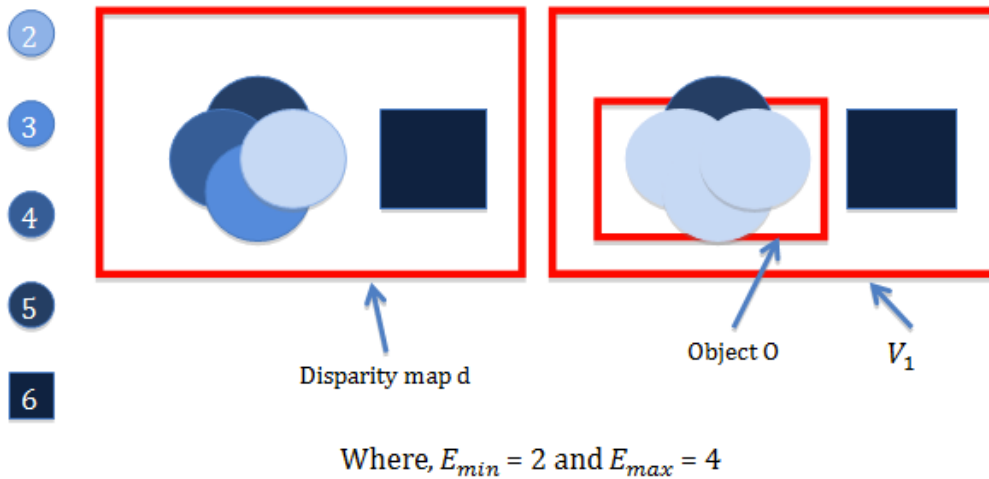


Fig 6. 20 :An Illustration of block matching methods

Now, disparity  $d$  obtained from the above equations is acquired. Then an algorithm is formulated (see eq (6.3)) mathematically to describe the extraction of object  $O$  from the view  $V_1$  image, where  $E_{min}$  and  $E_{max}$  stand for E minimum disparity and E maximum disparity, respectively.

$$O_{(x,y)} = V_{1(x,y)} \quad E_{min} \leq d_{(x,y)} \leq E_{max} \quad \text{eq (6.3)}$$

Disparity Range



Extracted object  $O$

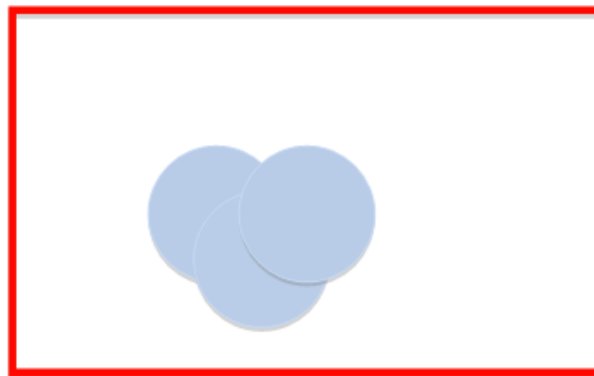


Fig 6. 21 : Demonstration of extraction of object from  $V_1$  based on disparity information.

The values of  $E_{min}$  and  $E_{max}$  are set according to the desired object that needs extracting, whereas it varies from one image to another in the scene.

### 6.6 Experimental Results and Observations

The experiment analyses the feasibility of the object extraction algorithm based on disparity from stereo 3D Images. The rendered stereo 3D views of section 6.2 are used in this experiment, where the differences between the two views are to the largest baseline distance.

Example1: EIs are at resolution of 80 by 80 pixels and 68 by 44 element images in total at resolution of 5466 x 3588 pixels with 250 $\mu$ m. View  $V_1$  of x-axis (x1)



and y-axis ( $y_1$ ) inside the element image are 2 and 35, respectively, while view  $V_2$  of y-axis  $y_2 = y_1$  and x-axis ( $x_2$ ) = 42, and  $SI = 12$ . The disparity map result is shown in Fig 6. 22(b) with  $w = 15$ ,  $R = 60$  and SSD and object O is extracted from a given disparity range of  $E_{min} = 18$  and  $E_{max} = 25$  as show in Fig 6. 22(d). The final result is observed with small errors that do not belong to the object O. This is due to the accuracy obtained from correlation matching criteria used, namely sum of the square difference (SSD).

Example2:  $SI=2$  and EI at resolution of 29 by 29 pixels with  $90\mu\text{m}$ . View  $V_1$   $x_1=3$ ,  $y_1=10$  and view  $V_2$  of  $x_2 = 12$  and  $y_2 = 10$ . Both focused planes are presented, one focused at the foreground object and second focused at background. Disparity of both are calculated with the same parameters  $w= 9$  and  $R=25$ . Result of object O is presented in Fig 6. 23 with a given disparity range of  $E_{min} = 18$  and  $E_{max} = 25$ .

In Fig 6. 23a, result of disparity is not efficient to use in extracting the object due to numerous errors in the disparity map. Therefore, the image plane is digitally changed at object O, the result of which shows an improvement compared with Fig 6. 23a(c). The extraction of object O is more effective with disparity range of  $E_{min} = 6$  and  $E_{max} = 12$  in Fig 6. 23b.

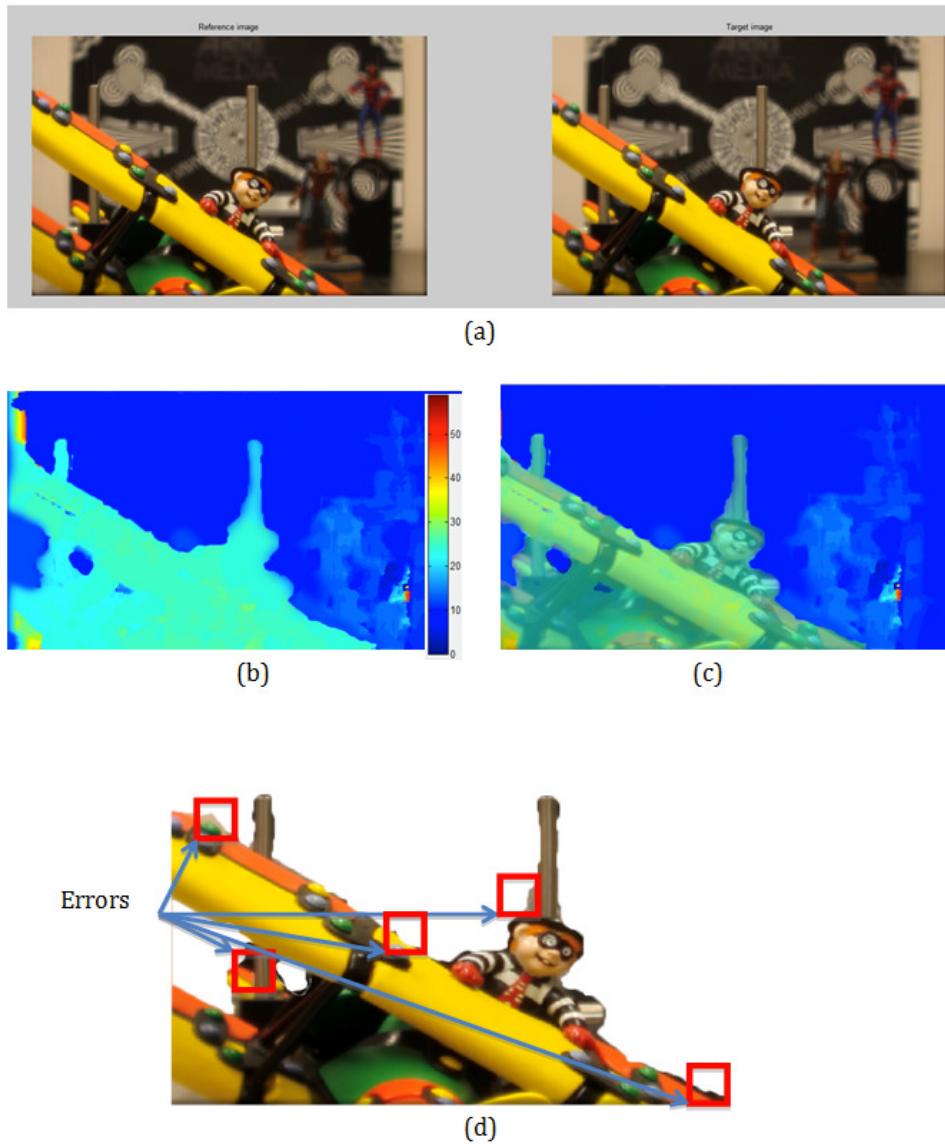


Fig 6. 22 : (a) show stereo views, (b) is disparity result with  $w=15$ ,  $R= 35$ . (c) illustrates object O overlaid with disparity  $d$  of (b) and (d) is the final result of object O

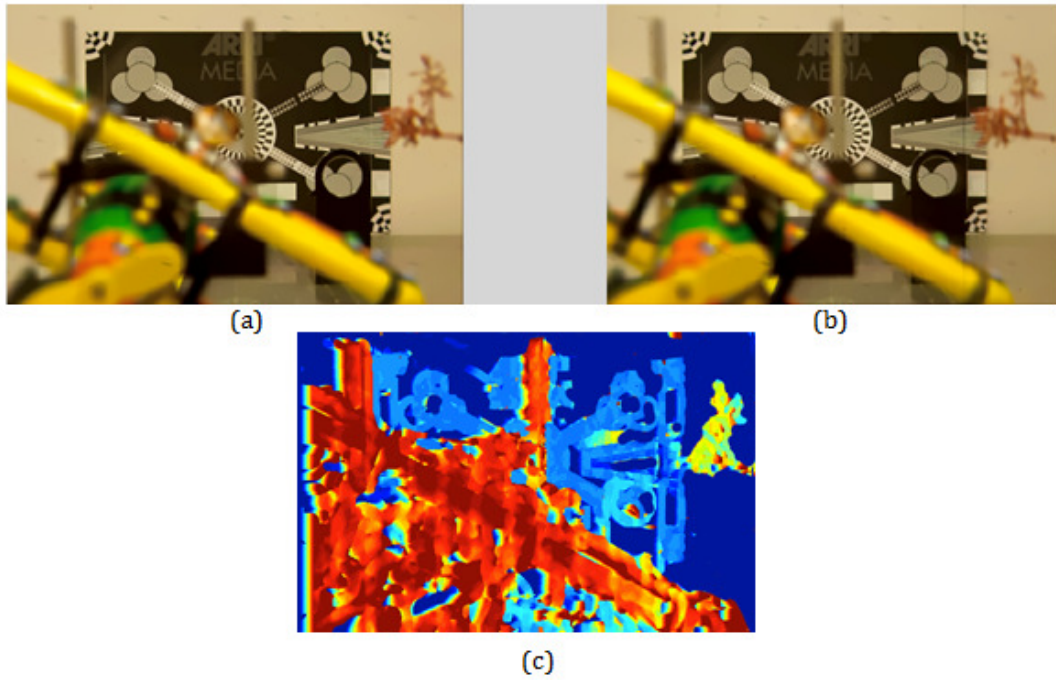


Fig 6. 23a: (a) and (b) is stereo views and (c) is the disparity result of the stereo view.

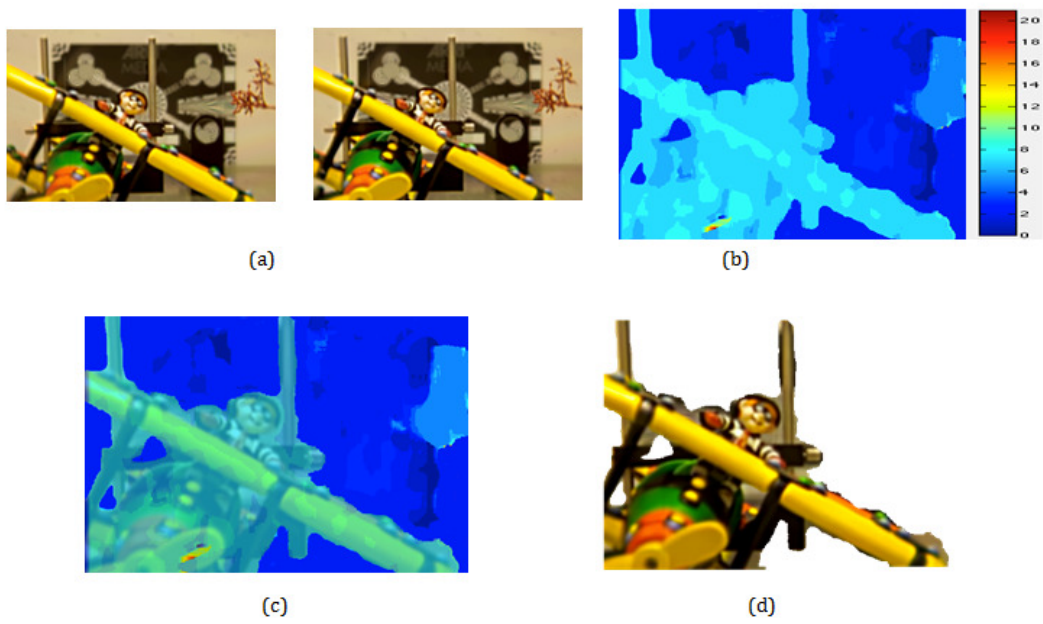
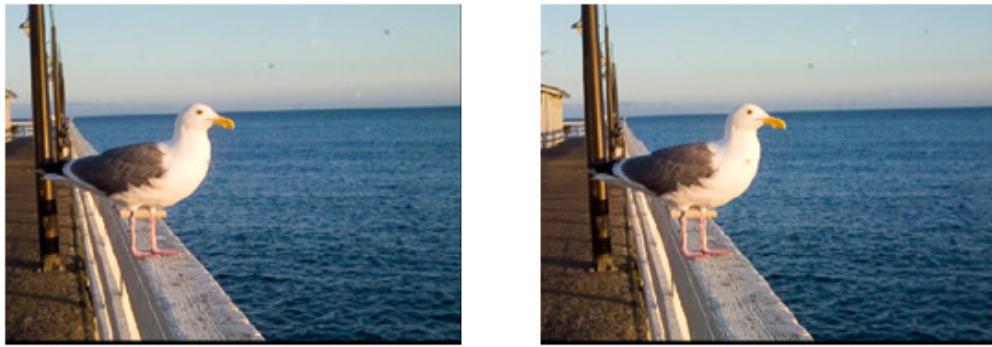
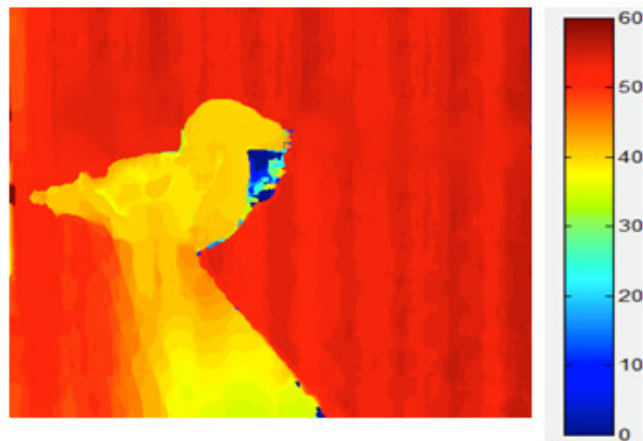


Fig 6. 23b : Illustration of object extraction using disparity map.



(a)

(b)



(c)



(d)

Fig 6. 24 : The result of object 0 is within disparity ranging from  $E_{min}= 43$  and  $E_{max} = 36$  shown in (d).

**Example3:** SI=9 and EI at resolution of 74 by 74 pixels.

V\_1 (x1=3, y1=10), V\_2(x2=12, y2 = 3), w=19, R=60.

Disparity is used to extract object O in Fig 6. 24 with a given disparity range of E\_min= 43 and E\_max = 36.

This method shows an effective way of extracting object. However, high resolution of EI and long baseline is preferred in disparity estimation, where high accuracy is achieved in extracting objects. Also it ignores poor SI from EI in generating the stereo views, which will increase the quality on the views and also accuracy in disparity estimation. As object extraction algorithm is directly related to the accuracy of disparity, objects are extracted with minimum errors as can be seen in the experimental section.

## **6.7 Conclusions**

This chapter has presented the integration process of different display technologies by providing content to all cross platform displays from holographic 3D content. This experiment is the first of its kind ever attempted on holographic 3D videos whereas recent developments have been carried out on still images. Also, clear data flow chain is explained in an attempt to achieve content for each display technology. The above experiment proved that holographic technology did deliver 2D content for normal display but it can also fulfil the stereoscopic 3D display. Therefore, this experiment will show how holographic content can meet the demands of existing 3D stereo technologies in the market.

The resolutions achieved are reasonably acceptable; however, the depth on stereoscopic display is slightly small. This is due to the size limitation of the image sensor on the camera, as there is a trade off in increasing the baseline with the resolution of views. Therefore, results with larger image sensor showed promising outcome in terms of resolution as well as in depth perception. Large image sensor in this experiment has made improvement both in 3D stereoscopic display as well as in multiview display.

Single aperture camera is used to generate 3D content for different types of displays. As this comes to show, cameras with a bigger image sensor may become the future in motion picture capturing; for which this 3D content capturing may become the future. This approach is very clearly focused on finding a simple and efficient way of capturing real 3D content without having to go through the complex dual or multi camera calibrations.

Object extraction performance is analysed and shows that the disparity analysis is of great importance in achieving correct boundary of an object from the scene. The work in this chapter is mainly concerned with developing algorithms to show the simplest way of performing the object extraction from holographic content. A comprehensive experiment is carried out to prove the proposed approach in extracting objects using the MATLAB simulation environment.

## **6.8 References**

- [1] M. G. Zarpalas, D., Biperis, I., Fotiadou, E., Lyka, E., Daras, P., & Strintzis, "Depth estimation in integral images by anchoring optimization techniques.," *J. Disp. Technol.*, pp. 1–6, 2011.
- [2] O. AbdulFatah, A. Aggoun, M. Nawaz, and J. Cosmas, "Depth mapping of integral images using a hybrid disparity analysis algorithm," 2012 IEEE Int. Symp. on. IEEE, Broadband Multimed. Syst. Broadcast., pp. 1–4, 2012.
- [3] D. Zarpalas and E. Fotiadou, "Anchoring Graph Cuts Towards Accurate Depth Estimation in Integral Images," *J. Disp. Technol.*, vol. 8, no. 7, pp. 405–417, 2012.
- [4] E. Alazawi, A. Aggoun, M. Abbod, O. A. Fatah, and M. R. Swash, "Adaptive depth map estimation from 3D integral image," 2013 IEEE Int. Symp. Broadband Multimed. Syst. Broadcast., pp. 1–6, Jun. 2013.
- [5] C. Wu, "Depth extraction from unidirectional integral image using a modified multibaseline technique," *Proceedings of SPIE. Spie*, pp. 135–145, 2002.
- [6] C. Wu, A. Aggoun, S. Y. Kung, and M. McCormick, "Depth measurement from integral images through viewpoint image extraction and a modified multibaseline disparity analysis algorithm," *J. Electron. Imaging*, vol. 14, no. 2, p. 23018, 2005.
- [7] A. Aggoun, P. Nunes, J. Steurer, and M. Meier, "Report on Integration issues and prototype system," 2013.
- [8] H. Zhou, X. Li, and A. H. Sadka, "Nonrigid Structure-From-Motion From 2-D Images Using Markov Chain Monte Carlo," *Multimedia, IEEE Trans.*, vol. 14, no. 1, pp. 168–177, Feb. 2012.
- [9] H. Zhou, A. M. Wallace, and P. R. Green, "Efficient tracking and ego-motion recovery using gait analysis," *Signal Processing*, vol. 89, no. 12, pp. 2367–2384, 2009.

## **7. Chapter Seven**

### **Conclusion and Further work**

This thesis presented and discussed number of holographic 3D post-production algorithms, which includes an improvement to digital refocusing for focus correction, depth map extraction, 3D stereoscopic and multiview content creating from holographic 3D images as well as 3D image segmentation.



## **7.1 Conclusion**

A detailed literature review on the existing 3D imaging technologies has been presented and it has been concluded that the simplest form of presenting the true-3D is holographic 3D imaging system. In addition, this has opened up the possibilities such as changing the focus plane after capturing as well as 3D depth information. Holographic 3D imaging has demonstrated that it can deliver a rich viewing sensation that is eye fatigue free and without a need for any headgear glasses.

State-of-the-art digital refocusing techniques remain with limitations such as unnatural artifacts. A novel approach was proposed in this research that effectively refocuses using low-resolution orthographic images to form a higher resolution image. In addition, new interpolation approach is incorporated to improve the visual quality of the final image using the VP approach. As a result the final image looks more like a natural photograph without unnatural artifacts. In addition, all-in-focus image and depth information algorithms have been proposed. An all-in-focus image is generated by analyzing the depth plane of each window of elemental image blocks using contrast estimation, where the highest contrast extracts the focused window from different depth planes. The computational experiments were illustrated and discussed on both UIIs and OIIs to show the enhancement using the VP method with interpolation.

However, the achievable image quality did not yet satisfy the quality standard of commercial camera quality such as FHD or 4K. Further research was carried out to improve this revolutionary approach for solving the focus-plane challenge. As a result, a new method of refocusing based on high resolution views was proposed that uses the up-sampling, shift, and integration, which is achieved by sub pixel adjustment (SPA) method. This method extracts multiple pixels from each element image instead of one pixel when using VP refocusing approach. The resolution of each view is much higher, which results in a higher resolution of the final refocused image. A detailed experiment has been carried out that illustrated the effectiveness of the method. Consistency was maintained and a benchmark was established by using the same holographic 3D content throughout all experiments, to make comparisons of the results with other

methods more analogously. In addition, sets of different holographic 3D content were used to compare with the established benchmark. The resulting images and user testing confirms that the proposed method produced acceptable quality of resolution subjectively. Yet at this stage, artifacts on the final image still remained visible to some extent.

As a result, further improvement on the SPA method was made in order to cut the artifacts down to its minimum, where a delta value was introduced in the equation in section 4.4 to work more effectively. This allows a controllable degree of light rays intersecting amongst VPs enabling an accurate integration of pixels from all the views. The SPA method blurs out the artifact regions of the final image with a given value of delta. The delta value controls the blurring on the final image. Experimental results have shown improvement on the final refocused images, which is achieved from using SPA method. Also, the seagull image from 'Todor' was used in this experiment, where the optical elements of the camera were modified to reduce the artifacts using rendering with blending algorithm. The seagull image with the proposed (SPA) rendering created all-in-focus image with promising results, whereas the same parameters used with up-sampling, shift, and integration resulted in minor artifacts in the final rendered image. The resulting images achieved in both methods have shown an improvement in removing the artifacts but returned the final image with unwanted noise.

The final rendered images still contained unwanted noise and blurring using SPA. This is due to recursive shift and integration of views. Therefore, a proposed smart filtering approach in this research was introduced for the purpose of removing unwanted noise and blurring effect. This is to further improve the SPA method. Pre-processing as well as post-processing techniques titled as 'smart filters' have been proposed to apply to both stages. Both pre-processing and post-processing stages are equally important in imaging systems as pre-processing allows to enhance visual data, whereas post-processing allows to do various adjustments to the final image to improve the quality and resolution.

Pre-processing played a very vital role at the beginning of the process that effectively removed any unnecessary noise in the holographic 3D content. This was done with a simple but effective process that suppresses the noise as well as keeping the details of the overall holographic 3D content by avoiding the effect of over sharpening. This process does not put too much emphasis on the edges, making it easier for the human eyes to pickup; however it is similar to unsharpened masking method. The results were observed with well-defined contrast bringing out every small detail, which appeared better compared with the original one. Therefore, this made prominent impact on refocused images, bringing out structural details on the focused regions of the image. Otherwise, small structural details would completely fade away with recursive or iterative shift and integration of different views.

Post-processing with powerful smoothing method is incorporated as called “smoothing via  $L_0$  gradient minimisation”. This smoothing mechanism effectively removed parts of the noise, unwanted details and even slight blurriness that was introduced in the refocusing stage. The final result clearly confirmed improvements in terms of image quality and resolution. Experiments have been carried out to clearly point out the right weighting value for lambda. With smart filters, the refocusing algorithm has achieved high quality refocused images without artifacts and noises. A comprehensive subjective assessment of image quality was conducted to evaluate the human perception. The four different test holographic 3D images used in the subjective assessment were rendered with four different refocusing algorithms for like-to-like comparison. This aimed to address quality of the resulting images rendered with the refocusing algorithms during the subjective test. The test results were analysed to demonstrate the most effective method amongst four algorithms to determine the best image quality. The results show that the output of the proposed methods correlate strongly with overall viewer perception of image quality.

Content integration for different 3D display technologies has been presented by converting holographic 3D images to all cross-platform multiview 3D displays. This was the first time an attempt ever made to playback holographic 3D videos

on stereo and multiview 3D displays. The recent developments were carried out on still images. Also, clear data flow chain was defined and discussed in an attempt to achieve content for each display technology. The resulting resolution was acceptable; however, the depth perception on stereoscopic 3D display was slightly small due to the size limitation of the image sensor on the camera, as there was trade-off in increasing the baseline with the resolution of views. But, the results with larger image sensor showed a promising outcome in terms of resolution as well as in-depth perception. Large image sensor in the experiment showed improvement both in stereoscopic and multiview 3D displays.

Single aperture camera was used to generate holographic 3D content for different types of 3D displays. The experiments suggest that cameras with bigger image sensor may become the future of motion picture capturing. In other words, the future of the present capturing may become the technology of the past with holographic 3D imaging. This approach is very clearly focused at simple and efficient way of capturing real 3D content without having to go through the complex dual or multi camera calibrations.

Furthermore, object extraction performance has been analysed for disparity, which shows a significant importance in capturing the correct object from the scene. The proposed method was developed to show the simplest way of performing the object extraction from holographic 3D content. The results have proved a promising outcome with minor errors that need further improvement on disparity analysis stage.

## **7.2 Further work**

A number of techniques might be adopted to further improve the performance of the digital refocusing. Here are some examples:

1. To remove the conventional demosaicing algorithm that is provided with camera, and to process the raw sensor values from camera with different commercial and non-commercial demosaicing algorithms to see the impact of the final results.
2. After observing the impact on final image with different demosaicing algorithms, a new demosaicing algorithm specifically for holographic 3D content should be developed. This process is known as one of the important, yet challenging steps in reconstructing full RGB pixels from raw sensor values. The process may remove the antialiasing blur when resizing the image in a more efficient way than the existing conventional demosaicing methods do.
3. To develop a single view extraction on camera chip to allow real-time viewing experience. This will help producers and camera operators in content capturing of holographic 3D content.
4. To implement demosaicing algorithm on the camera chip combined with single view extraction to allow high quality resolution on camera viewing capability.
5. To implement holographic 3D content rendering on GPU to increase the processing time in post-production.
6. To improve the optical parameter of the camera, for example, through increasing the viewing angle of microlens array that will increase the baseline distance both in 3D multiview and 3D stereoscopic views from holographic 3D content. As a result, better depth perception may be pragmatic.
7. Object extraction (also known as object segmentation) may be improved if improved views are generated with demosaicing algorithm. Also an improved cross-correlation matching such as multi baseline matching should be implemented to enhance the disparity map, which will reduce the errors on the final extracted object. Furthermore, growing in the region of

interest algorithm can also be implemented to help in effectively extracting the desired object with given disparity map information.

### 7.3 Appendix A

#### 7.4 Sample Questionnaire

Gender

<input type="checkbox"/> Male	<input type="checkbox"/> Female
-------------------------------	---------------------------------

Age:

<input type="checkbox"/> 18-to-25	<input type="checkbox"/> 26-to35	<input type="checkbox"/> 36-to-45	<input type="checkbox"/> 46-to55	<input type="checkbox"/> 56-to-65
-----------------------------------	----------------------------------	-----------------------------------	----------------------------------	-----------------------------------

What is your knowledge of image quality?

	Excellent	Good	Fair	Don't Know	Bad
Answer	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

What is your knowledge of Holoscopic 3D?


	Excellent	Good	Fair	Don't Know	Bad
Answer	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

How important is image quality for you?


	Very Important	Important	Less Important	Don't Know	Not Important
Answer	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

**Section 1: Image A**


Algorithm 1: Please rate

	Excellent	<input type="checkbox"/>
	Good	<input type="checkbox"/>
	Fair	<input type="checkbox"/>
	Don't	<input type="checkbox"/>
	Bad	<input type="checkbox"/>


Algorithm 2: Please rate

	Excellent	<input type="checkbox"/>
	Good	<input type="checkbox"/>
	Fair	<input type="checkbox"/>
	Don't	<input type="checkbox"/>
	Bad	<input type="checkbox"/>

Algorithm 3: Please rate

	Excellent	<input type="checkbox"/>
	Good	<input type="checkbox"/>
	Fair	<input type="checkbox"/>
	Don't	<input type="checkbox"/>
	Bad	<input type="checkbox"/>

Algorithm 4: Please rete

	Excellent	<input type="checkbox"/>
	Good	<input type="checkbox"/>
	Fair	<input type="checkbox"/>
	Don't	<input type="checkbox"/>
	Bad	<input type="checkbox"/>



**Section 2: Image B**

Algorithm 1: Please rate

	Excellent	<input type="checkbox"/>
	Good	<input type="checkbox"/>
	Fair	<input type="checkbox"/>
	Don't	<input type="checkbox"/>
	Bad	<input type="checkbox"/>

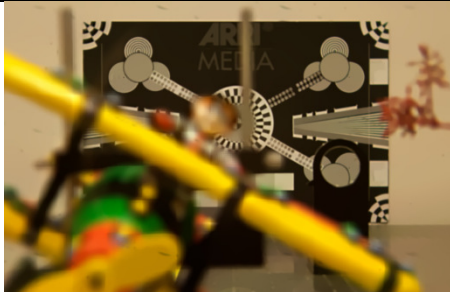
Algorithm 2: Please rate

	Excellent	<input type="checkbox"/>
	Good	<input type="checkbox"/>
	Fair	<input type="checkbox"/>
	Don't	<input type="checkbox"/>
	Bad	<input type="checkbox"/>

Algorithm 3: Please rate

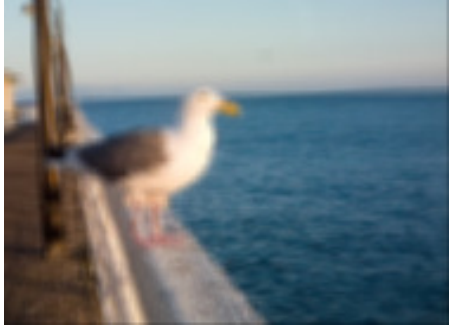
	Excellent	<input type="checkbox"/>
	Good	<input type="checkbox"/>
	Fair	<input type="checkbox"/>
	Don't	<input type="checkbox"/>
	Bad	<input type="checkbox"/>

Algorithm 4: Please rate


	Excellent	<input type="checkbox"/>
	Good	<input type="checkbox"/>
	Fair	<input type="checkbox"/>
	Don't	<input type="checkbox"/>
	Bad	<input type="checkbox"/>

**Section 3: Image C**


Algorithm 1: Please rate

	Excellent	<input type="checkbox"/>
	Good	<input type="checkbox"/>
	Fair	<input type="checkbox"/>
	Don't	<input type="checkbox"/>
	Bad	<input type="checkbox"/>

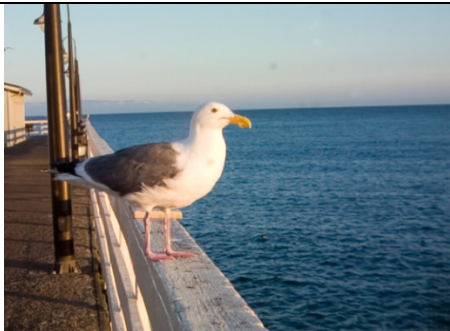
Algorithm 2: Please rate

	Excellent	<input type="checkbox"/>
	Good	<input type="checkbox"/>
	Fair	<input type="checkbox"/>
	Don't	<input type="checkbox"/>
	Bad	<input type="checkbox"/>

Algorithm 3: Please rate

	Excellent	<input type="checkbox"/>
	Good	<input type="checkbox"/>
	Fair	<input type="checkbox"/>
	Don't	<input type="checkbox"/>
	Bad	<input type="checkbox"/>

Algorithm 4: Please rate

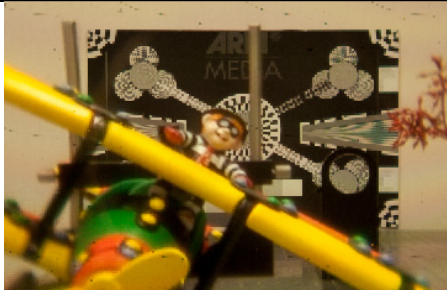
	Excellent	<input type="checkbox"/>
	Good	<input type="checkbox"/>
	Fair	<input type="checkbox"/>
	Don't	<input type="checkbox"/>
	Bad	<input type="checkbox"/>

**Section 4: Image D**


Algorithm 1: Please rate

	Excellent	<input type="checkbox"/>
	Good	<input type="checkbox"/>
	Fair	<input type="checkbox"/>
	Don't	<input type="checkbox"/>
	Bad	<input type="checkbox"/>

Algorithm 2: Please rate

	Excellent	<input type="checkbox"/>
	Good	<input type="checkbox"/>
	Fair	<input type="checkbox"/>
	Don't	<input type="checkbox"/>
	Bad	<input type="checkbox"/>

Algorithm 3: Please rate

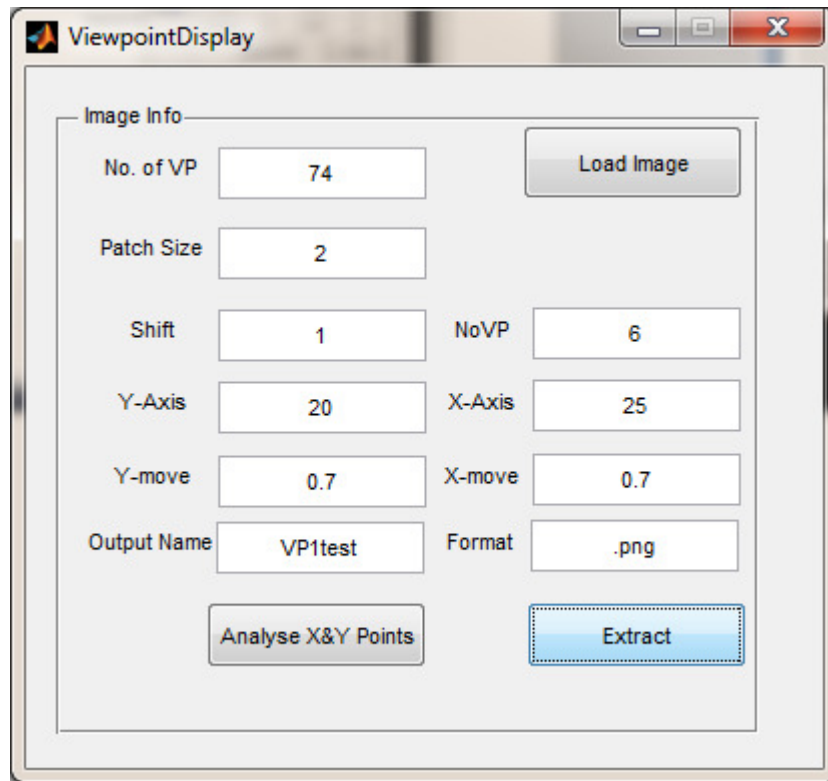
	Excellent	<input type="checkbox"/>
	Good	<input type="checkbox"/>
	Fair	<input type="checkbox"/>
	Don't	<input type="checkbox"/>
	Bad	<input type="checkbox"/>

Algorithm 4: Please rate

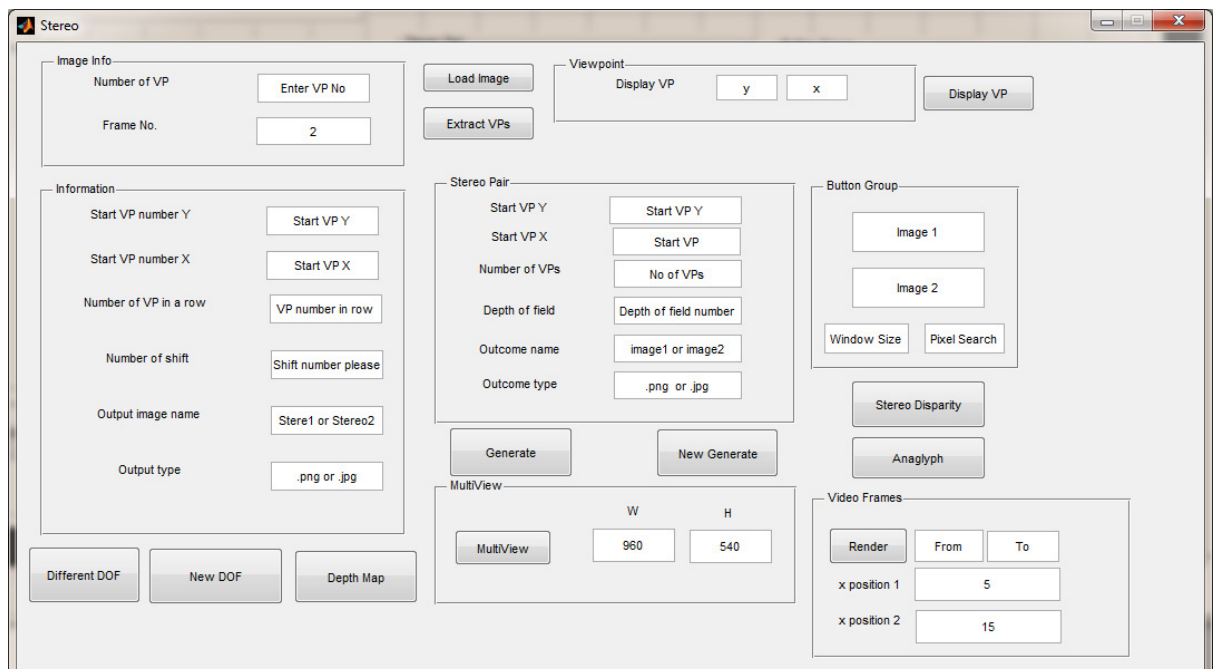
	Excellent	<input type="checkbox"/>
	Good	<input type="checkbox"/>
	Fair	<input type="checkbox"/>
	Don't	<input type="checkbox"/>
	Bad	<input type="checkbox"/>

Thank you for your time and participation. Hope you've enjoyed it.

Appendix B: Developed GUI using Matlab application



Representation of GUI A



Representation of GUI B

## Appendix C: Matlab Scripts for GUI A

```

function varargout = ViewpointDisplay(varargin)
% VIEWPOINTDISPLAY MATLAB code for ViewpointDisplay.fig
%   VIEWPOINTDISPLAY, by itself, creates a new VIEWPOINTDISPLAY
or raises the existing
%   singleton*.
%
%   H = VIEWPOINTDISPLAY returns the handle to a new
VIEWPOINTDISPLAY or the handle to
%   the existing singleton*.
%
%   VIEWPOINTDISPLAY('CALLBACK',hObject,eventData,handles,...)
calls the local
%   function named CALLBACK in VIEWPOINTDISPLAY.M with the given
input arguments.
%
%   VIEWPOINTDISPLAY('Property','Value',...) creates a new
VIEWPOINTDISPLAY or raises the
%   existing singleton*. Starting from the left, property value
pairs are
%   applied to the GUI before ViewpointDisplay_OpeningFcn gets
called. An
%   unrecognized property name or invalid value makes property
application
%   stop. All inputs are passed to ViewpointDisplay_OpeningFcn
via varargin.
%
%   *See GUI Options on GUIDE's Tools menu. Choose "GUI allows
only one
%   instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help
ViewpointDisplay

% Last Modified by GUIDE v2.5 15-Jan-2013 18:16:28

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                  'gui_Singleton',   gui_Singleton, ...
                  'gui_OpeningFcn',   @ViewpointDisplay_OpeningFcn,
                  ...
                  'gui_OutputFcn',    @ViewpointDisplay_OutputFcn,
                  ...
                  'gui_LayoutFcn',    [] , ...
                  'gui_Callback',     []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

```

```
% --- Executes just before ViewpointDisplay is made visible.
function ViewpointDisplay_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)
% varargin   command line arguments to ViewpointDisplay (see
VARARGIN)

% Choose default command line output for ViewpointDisplay
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes ViewpointDisplay wait for user response (see
UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = ViewpointDisplay_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

function NoVPin_Callback(hObject, eventdata, handles)
% hObject    handle to NoVPin (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of NoVPin as text
%         str2double(get(hObject,'String')) returns contents of
NoVPin as a double

% --- Executes during object creation, after setting all properties.
function NoVPin_CreateFcn(hObject, eventdata, handles)
% hObject    handle to NoVPin (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

end

```
function PatchSize_Callback(hObject, eventdata, handles)
% hObject    handle to PatchSize (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of PatchSize as text
%         str2double(get(hObject,'String')) returns contents of
PatchSize as a double

% --- Executes during object creation, after setting all properties.
function PatchSize_CreateFcn(hObject, eventdata, handles)
% hObject    handle to PatchSize (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function Yaxis_Callback(hObject, eventdata, handles)
% hObject    handle to Yaxis (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Yaxis as text
%         str2double(get(hObject,'String')) returns contents of Yaxis
as a double

% --- Executes during object creation, after setting all properties.
function Yaxis_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Yaxis (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function Xaxis_Callback(hObject, eventdata, handles)
% hObject    handle to Xaxis (see GCBO)
```

```

% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Xaxis as text
%         str2double(get(hObject,'String')) returns contents of Xaxis
as a double

% --- Executes during object creation, after setting all properties.
function Xaxis_CreateFcn(hObject, eventdata, handles)
% hObject handle to Xaxis (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in LoadImage.
function LoadImage_Callback(hObject, eventdata, handles)
% hObject handle to LoadImage (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
[filename,pathname]=uigetfile('*.jpg','Select an image File');
%image=VideoReader(fullfile(pathname,filename));
image=imread(fullfile(pathname,filename));
handles.pathname=pathname;
handles.filename=filename;
handles.image=image;
guidata(hObject,handles);

% --- Executes on button press in ExtractVP.
function ExtractVP_Callback(hObject, eventdata, handles)
% hObject handle to ExtractVP (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
integralImage=handles.image;
NoVP=str2num(get(handles.NoVP1,'String'));
PatchSize=str2num(get(handles.PatchSize,'String'));

Yaxis=str2num(get(handles.Yaxis,'String'));
Xaxis=str2num(get(handles.Xaxis,'String'));

YMove=str2num(get(handles.YMove,'String'));
XMove=str2num(get(handles.XMove,'String'));

NoVPin= str2num(get(handles.NoVPin,'String'));

if NoVPin==0
    NoVPin=floor((((NoVP/PatchSize)-1)/2)*PatchSize);

```



```

else
    NoVPin= str2num(get(handles.NoVPin, 'String'));
end

NoShift= str2num(get(handles.shift, 'String'));

if NoShift==0
    NoShift=floor(((NoVP/PatchSize)-1)/2);
    t=1:NoShift:NoVPin;
    balance=(size(t,2));
elseif NoShift==1
    NoShift=0;
    balance=NoVPin;
else
    NoShift= str2num(get(handles.shift, 'String'));
    t=1:NoShift:NoVPin;
    balance=(size(t,2));
%     balance=NoVPin;
end

%         if         str2num(get(handles.shift, 'String'))==0         &&
str2num(get(handles.NoVPin, 'String'))==0
%     NoVPin= floor(NoVPin/NoShift);
% end
ext = get(handles.ext, 'String');
OutName= get(handles.OutName, 'String');

%[VP1]=         HD2image_new2_sharp(integralImage,NoVP,PatchSize,
Yaxis,Xaxis,NoVPin,YMove,XMove,NoShift,balance);

[VP1]=         HD2image_new2(integralImage,NoVP,PatchSize,
Yaxis,Xaxis,NoVPin,YMove,XMove,NoShift,balance);

if NoShift==0
    pia=1;
else
    pia=NoShift;
end
pic=floor(NoVPin/pia);
pic=floor(pic*pia);
VP1=VP1(pic:size(VP1,1)-((pic*2)+pia),pic:size(VP1,2)-
((pic*2)+pia),:);
%VP1=imresize(VP1,[882 1330], 'bicubic');
%H = fspecial('disk',3);1920x1080
%VP1 = imfilter(VP1,H,'replicate');
% H = fspecial('unsharp');
% %VP1(:, :, 1) = imfilter(VP1(:, :, 1),H,'replicate');
% %VP1(:, :, 2) = imfilter(VP1(:, :, 2),H,'replicate');
% VP1(:, :, 3) = imfilter(VP1(:, :, 3),H,'replicate');
% % H = fspecial('disk',2);
% % VP1(:, :, 3) = imfilter(VP1(:, :, 3),H,'replicate');
% % %VP1(:, :, 1) = imfilter(VP1(:, :, 1),H,'replicate');
% h = fspecial('gaussian', [50 50]);
% VP1 = imfilter(VP1,h,'replicate');

imwrite(VP1, [OutName ext], 'tiff', 'Compression', 'none');
%imwrite(VP1, [OutName ext]);
figure(331);imshow(VP1);

```

```

function YMove_Callback(hObject, eventdata, handles)
% hObject    handle to YMove (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of YMove as text
%        str2double(get(hObject,'String')) returns contents of YMove
as a double

% --- Executes during object creation, after setting all properties.
function YMove_CreateFcn(hObject, eventdata, handles)
% hObject    handle to YMove (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%        See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function XMove_Callback(hObject, eventdata, handles)
% hObject    handle to XMove (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of XMove as text
%        str2double(get(hObject,'String')) returns contents of XMove
as a double

% --- Executes during object creation, after setting all properties.
function XMove_CreateFcn(hObject, eventdata, handles)
% hObject    handle to XMove (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%        See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in Analyse.
function Analyse_Callback(hObject, eventdata, handles)
% hObject    handle to Analyse (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

```

```

integralImage=handles.image;
NoVP=str2num(get(handles.NoVP1,'String'));
PatchSize=str2num(get(handles.PatchSize,'String'));

Yaxis=str2num(get(handles.Yaxis,'String'));
Xaxis=str2num(get(handles.Xaxis,'String'));

YMove=str2num(get(handles.YMove,'String'));
XMove=str2num(get(handles.XMove,'String'));

% NoViewpoint= str2num(get(handles.VPNoInRow,'String'));
% NoShift= str2num(get(handles.text33,'String'));
% ext = get(handles.ext,'String');
% OutName= get(handles.OutName,'String');

[Image]=Analyse(integralImage,NoVP,YMove,XMove,Yaxis,Xaxis);

figure(1);imshow(Image);

function OutName_Callback(hObject, eventdata, handles)
% hObject    handle to OutName (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of OutName as text
%        str2double(get(hObject,'String')) returns contents of
OutName as a double

% --- Executes during object creation, after setting all properties.
function OutName_CreateFcn(hObject, eventdata, handles)
% hObject    handle to OutName (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function ext_Callback(hObject, eventdata, handles)
% hObject    handle to ext (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of ext as text
%        str2double(get(hObject,'String')) returns contents of ext
as a double

% --- Executes during object creation, after setting all properties.

```

```
function ext_CreateFcn(hObject, eventdata, handles)
% hObject    handle to ext (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function shift_Callback(hObject, eventdata, handles)
% hObject    handle to shift (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of shift as text
%       str2double(get(hObject,'String')) returns contents of shift
as a double

% --- Executes during object creation, after setting all properties.
function shift_CreateFcn(hObject, eventdata, handles)
% hObject    handle to shift (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function NoVP1_Callback(hObject, eventdata, handles)
% hObject    handle to NoVP1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of NoVP1 as text
%       str2double(get(hObject,'String')) returns contents of NoVP1
as a double

% --- Executes during object creation, after setting all properties.
function NoVP1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to NoVP1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.  
% See ISPC and COMPUTER.  
if ispc && isequal(get(hObject,'BackgroundColor'),  
get(0,'defaultUicontrolBackgroundColor'))  
    set(hObject,'BackgroundColor','white');  
end
```

## Appendix D: Matlab code for SPA

```

function [VP1]= HD2image_new(integralImage, MicroImageSize, combineVP,
ss, ii, NoOfVP, YMove, XMove, shift, balance)

%name=integralImage;

%integralImage=imread([integralImage typeEx1]);
% combineVP= str2num(get(handles.PatchSizeDisp, 'String'));
% ss= str2num(get(handles.DisplayViewpoint_Y, 'String'));
% ii= str2num(get(handles.DisplayViewpoint_X, 'String'));
% NoOfVP=str2num(get(handles.NoOfVPDisp, 'String'));
%
% MicroImageSize = str2num(get(handles.NumberOfVP, 'String'));
%shift=0;
% %handles.image=Image;
% %guidata(hObject, handles);
% %numViewPoint=str2num(get(handles.NumberOfVP, 'String'));
shift1=1;
numViewPoint=MicroImageSize;

NoOfVP=round(NoOfVP);
shift=round(shift);
if shift==0
    shift=1;
end

%NoViewpoint= str2num(get(handles.VPNoInRow, 'String'));
%NoShift= str2num(get(handles.Shift, 'String'));
%type = get(handles.typeEx, 'String');
%pathname1='D:\obaid\Database (Second)\Quadruped
Mammals\DepthV\HDImages\';
%OutImageName= get(handles.OutputName, 'String');
num_lens_x=floor(size(integralImage,2)/numViewPoint)*combineVP;
num_lens_y=floor(size(integralImage,1)/numViewPoint)*combineVP;

%VP1=
im2double(zeros((num_lens_y+((shift+1)*NoOfVP))*NoOfVP, ((num_lens_
x+((shift+1)*NoOfVP))*NoOfVP), 3));
%VP1=
im2uint16(zeros((num_lens_y+((shift1+1)*NoOfVP)), (num_lens_x+((shift
1+1)*NoOfVP)), 3));
VP1=
im2double(zeros((num_lens_y+((shift1+1)*(NoOfVP+2))), (num_lens_x+((s
hift1+1)*(NoOfVP+2))), 3));
num_elem_pixc=floor(size(integralImage,2)/numViewPoint)*
numViewPoint;%Number of complete cylindrical microlenses with pixels
num_elem_pixr=floor(size(integralImage,1)/numViewPoint)*
numViewPoint;%Number of complete cylindrical microlenses with pixels
for r = 1:1:NoOfVP

    rowpoint=0;
    yy=0;
    brr=0;
    for VPr =(r+(ss-1)):numViewPoint:num_elem_pixr-(numViewPoint)
        yy=yy+1;

        for rr=0:(combineVP-1)
            rowpoint=rowpoint+1;

```

```

        y=(yy*(combineVP-1)+1)-rr+yy;
        VPr (rowpoint)=(VPr+floor (brr))+rr;
        %VPr (y-1)=VPr+rr;

    end
    brr=brr+YMove;
end
%-----

%           VPr=VPr+((combineVP-1)*(r-1));

%-----
for i = 1:1:NoOfVP
    bcc=0;
    colpoint=0;
    xx=0;
    for VPc=(i+(ii-1)):numViewPoint:num_elem_pixc-(numViewPoint)
        xx=xx+1;
        for cc=0:(combineVP-1)
            colpoint=colpoint+1;
            x=(xx*(combineVP-1)+1)-cc+xx;
            VPcc (colpoint)=(VPc+floor (bcc))+cc;
            %VPcc (x-1)=VPc+cc;

        end
        bcc=bcc+XMove;
    end
    shift1=NoOfVP;
%-----

%           VPcc1=VPcc+((combineVP-1)*(i-1));

%-----

    VP=integralImage (VPr,VPcc,:);
%           VP=imresize (VP, [(size (VP,1)*NoOfVP)
(size (VP,2)*NoOfVP)]);
    cj=floor (size (VP,2));%Number of complete cylindrical
microlenses with pixels
    ri=floor (size (VP,1));%Number of complete cylindrical
microlenses with pixels
    %VPii = r:size (VP1,1);
    % VPjj = i:size (VP1,2);
    VPi = r+((r-1)*(shift1-1)):size (VP1,1);
    VPj = i+((i-1)*(shift1-1)):size (VP1,2);
    %VPN= (VP)/(balance*balance);
    VPN= im2double (VP)/(NoOfVP*NoOfVP);
    iii=1:ri;
    jjj=1:cj;
    VP1 (VPi (iii),VPj (jjj),:)=(VP1 (VPi (iii),VPj (jjj),:))+
(VPN (iii, jjj, :));

end
end
end

```

## Appendix E: Matlab code for GUI B

```

function varargout = Stereo(varargin)
% STEREO MATLAB code for Stereo.fig
%     STEREO, by itself, creates a new STEREO or raises the
existing
%     singleton*.
%
%     H = STEREO returns the handle to a new STEREO or the handle
to
%     the existing singleton*.
%
%     STEREO('CALLBACK',hObject,eventData,handles,...) calls the
local
%     function named CALLBACK in STEREO.M with the given input
arguments.
%
%     STEREO('Property','Value',...) creates a new STEREO or raises
the
%     existing singleton*. Starting from the left, property value
pairs are
%     applied to the GUI before Stereo_OpeningFcn gets called. An
%     unrecognized property name or invalid value makes property
application
%     stop. All inputs are passed to Stereo_OpeningFcn via
varargin.
%
%     *See GUI Options on GUIDE's Tools menu. Choose "GUI allows
only one
%     instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Stereo

% Last Modified by GUIDE v2.5 05-Sep-2014 14:12:18

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                  'gui_Singleton',  gui_Singleton, ...
                  'gui_OpeningFcn', @Stereo_OpeningFcn, ...
                  'gui_OutputFcn',  @Stereo_OutputFcn, ...
                  'gui_LayoutFcn',  [] , ...
                  'gui_Callback',    []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
end

% --- Executes just before Stereo is made visible.
function Stereo_OpeningFcn(hObject, eventdata, handles, varargin)

```



```
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Stereo (see VARARGIN)

% Choose default command line output for Stereo
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Stereo wait for user response (see UIRESUME)
% uiwait(handles.figure1);

end

% --- Outputs from this function are returned to the command line.
function varargout = Stereo_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

end

function StartVPY_Callback(hObject, eventdata, handles)
% hObject    handle to StartVPY (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of StartVPY as text
%        str2double(get(hObject,'String')) returns contents of
StartVPY as a double

end

% --- Executes during object creation, after setting all properties.
function StartVPY_CreateFcn(hObject, eventdata, handles)
% hObject    handle to StartVPY (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

end

function NumberOfVP_Callback(hObject, eventdata, handles)
% hObject    handle to NumberOfVP (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of NumberOfVP as
text
%          str2double(get(hObject,'String')) returns contents of
NumberOfVP as a double
end

% --- Executes during object creation, after setting all properties.
function NumberOfVP_CreateFcn(hObject, eventdata, handles)
% hObject    handle to NumberOfVP (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%          See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

end

function VPNoInRow_Callback(hObject, eventdata, handles)
% hObject    handle to VPNoInRow (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of VPNoInRow as text
%          str2double(get(hObject,'String')) returns contents of
VPNoInRow as a double
end

% --- Executes during object creation, after setting all properties.
function VPNoInRow_CreateFcn(hObject, eventdata, handles)
% hObject    handle to VPNoInRow (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%          See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

end

function Shift_Callback(hObject, eventdata, handles)
% hObject    handle to Shift (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Shift as text
%          str2double(get(hObject,'String')) returns contents of Shift
as a double
end
```

```
% --- Executes during object creation, after setting all properties.
function Shift_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Shift (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

end

function OutputName_Callback(hObject, eventdata, handles)
% hObject    handle to OutputName (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of OutputName as
text
%         str2double(get(hObject,'String')) returns contents of
OutputName as a double

end

% --- Executes during object creation, after setting all properties.
function OutputName_CreateFcn(hObject, eventdata, handles)
% hObject    handle to OutputName (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

end

function typeEx_Callback(hObject, eventdata, handles)
% hObject    handle to typeEx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of typeEx as text
%         str2double(get(hObject,'String')) returns contents of
typeEx as a double
end

% --- Executes during object creation, after setting all properties.
function typeEx_CreateFcn(hObject, eventdata, handles)
% hObject    handle to typeEx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
```

```

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if      ispc      &&      isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

end

% --- Executes on button press in LoadImage.
function LoadImage_Callback(hObject, eventdata, handles)
% hObject      handle to LoadImage (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
[filename,pathname]=uigetfile('*.jpg','Select an image File');
image=VideoReader(fullfile(pathname,filename));
%image=imread(fullfile(pathname,filename));
handles.pathname=pathname;
handles.filename=filename;
handles.image=image;
guidata(hObject,handles);
end

% --- Executes on button press in DifferentDepth.
function DifferentDepth_Callback(hObject, eventdata, handles)
% hObject      handle to DifferentDepth (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
%integralImage=handles.image;
%integralImage=handles.image;
MicroImageSize = str2num(get(handles.NumberOfVP, 'String'));
%l = floor(MicroImageSize/2);
%Image = uint8(zeros(size(integralImage)));
%   for i = 1+1:MicroImageSize:size(integralImage,1)-1
%       for j = 1+1:MicroImageSize:size(integralImage,2)-1
%           %MicroBlock=flipud(img(i-1:i+1,j-1:j+1,:));
%           %Image(i-1:i+1,j-1:j+1,:)=MicroBlock(:,:);
%           MicroBlock=integralImage(i-1:i+1,j-1:j+1,:);
%           block=uint8(zeros(size(MicroBlock)));
%           block(:, :, 1)=flipud(MicroBlock(:, :, 1));
%           block(:, :, 2)=flipud(MicroBlock(:, :, 2));
%           block(:, :, 3)=flipud(MicroBlock(:, :, 3));
%           Image(i-1:i+1,j-1:j+1,:)=block(:,:);
%       end
%   end
%handles.image=Image;
%guidata(hObject,handles);
%imwrite(Image, 'InvertedImage.jpg');
%integralImage=Image;
%numViewPoint=str2num(get(handles.NumberOfVP, 'String'));

numViewPoint=MicroImageSize;
StartVPnumbery=str2num(get(handles.StartVPY, 'String'));
StartVPnumberx=str2num(get(handles.StartVPX, 'String'));
NoViewpoint= str2num(get(handles.VPNoInRow, 'String'));
NoShift= str2num(get(handles.Shift, 'String'));
type = get(handles.typeEx, 'String');

```

```

OutImageName= get(handles.OutputName, 'String');

subimages=handles.subimages;

%-----
    for iii= 0:NoShift

        numViewpoint1= 7;

        cj=floor(size(subimages{1,1},2)* NoViewpoint);%Number of
complete cylindrical microlenses with pixels
        ri=floor(size(subimages{1,1},1)* NoViewpoint);%Number of
complete cylindrical microlenses with pixels
        %ImageOutName= ([OutImageName num2str(NumIntegral) '-']);
        %VP=0;
        %shift=6;
        %shift=0;
        VP=
im2double(zeros((size(subimages{1,1},1)*(NoViewpoint))+((NoViewpoint
-1) * (iii+1)),(size(subimages{1,1},2)*(NoViewpoint))+((NoViewpoint-
1) * (iii+1)),3));

        for i = 1: NoViewpoint
            %iii=(i-1)*shift;
            VPi = i+((i-1)*iii):size(VP,1);
            %I = 1+(numViewpoint1 -i);
            for j = 1:NoViewpoint
                %jjj=(j-1)*shift;
                %J = 1+(numViewpoint1 -j);
                VPj = j+((j-1)*iii):size(VP,2);
                VPN=
im2double(imresize(subimages{i+StartVPnumbery, j+StartVPnumberx},
[ri,cj]))/(NoViewpoint*NoViewpoint) ;
                ii=1:ri;
                jj=1:cj;

VP(VPi(ii),VPj(jj),:)=(VP(VPi(ii),VPj(jj),:))+ (VPN(ii,jj,:)));

                %VP(VPi,VPj,:)=subimages{i+19,j+19};

            end
        end
        p=(NoViewpoint-1)*iii + ((NoViewpoint-iii)* (NoViewpoint-
1)/2);
        %p=(NoViewpoint-1)*iii;
        VP= VP(p: (size(VP,1)-p),p: (size(VP,2)-p),: );
        imwrite(VP,([OutImageName num2str(iii) type]));
    end

end

```

end

```

% --- Executes on button press in DisplayVP.
function DisplayVP_Callback(hObject, eventdata, handles)
% hObject      handle to DisplayVP (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
subimages=handles.subimages;
%DisplayVPnum=0;
DisplayVPnum_Y= str2num(get(handles.DisplayViewpoint_Y,'String'));
DisplayVPnum_X= str2num(get(handles.DisplayViewpoint_X,'String'));
figure(1);imagesc(subimages{DisplayVPnum_Y,DisplayVPnum_X});
figure(2);imshow(subimages{DisplayVPnum_Y,DisplayVPnum_X});

end

function DisplayViewpoint_Y_Callback(hObject, eventdata, handles)
% hObject      handle to DisplayViewpoint_Y (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

%   Hints:     get(hObject,'String') returns contents of
DisplayViewpoint_Y as text
%             str2double(get(hObject,'String')) returns contents of
DisplayViewpoint_Y as a double
end

% --- Executes during object creation, after setting all properties.
function DisplayViewpoint_Y_CreateFcn(hObject, eventdata, handles)
% hObject      handle to DisplayViewpoint_Y (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

% --- Executes on button press in ExtractVPs.
function ExtractVPs_Callback(hObject, eventdata, handles)
% hObject      handle to ExtractVPs (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
integralImage=handles.image;
FrameNo = str2num(get(handles.FrameNo,'String'));
integralImage=read(integralImage,FrameNo);
%integralImage=handles.image;

```

```

MicroImageSize = str2num(get(handles.NumberOfVP, 'String'));

    %handles.image=Image;
    %guidata(hObject, handles);
    numViewPoint=str2num(get(handles.NumberOfVP, 'String'));
    numViewPoint=MicroImageSize;
    %StartVPnumber=str2num(get(handles.StartVPY1, 'String'));
    %NoViewpoint= str2num(get(handles.VPNoInRow, 'String'));
    %NoShift= str2num(get(handles.Shift, 'String'));
    %type = get(handles.typeEx, 'String');

    %OutImageName= get(handles.OutputName, 'String');

    num_elem_pixc=floor(size(integralImage,2)/numViewPoint)*
    numViewPoint;%Number of complete cylindrical microlenses with pixels
    num_elem_pixr=floor(size(integralImage,1)/numViewPoint)*
    numViewPoint;%Number of complete cylindrical microlenses with pixels
    %ImageOutName= ([OutImageName num2str(NumIntegral) '-']);
    subimages=cell(numViewPoint,numViewPoint);
    %vpno=0;
    for r = 1 : numViewPoint
        %ImageOutName= ([OutImageName num2str(r) '-']);
        %ImageOutName1= ([OutImageName num2str(r) '-']);

        for i = 1 :numViewPoint
            %vpno= vpno+1;

            ViewPointPositionc = i:numViewPoint:num_elem_pixc;
            ViewPointPositionr = r:numViewPoint:num_elem_pixr;
            %extract the viewpoint from the integralImage and store
it onto
            %--eval(['VP_' num2str(i)
            '=integralImage(ViewPointPositionr,ViewPointPositionc,:);'])%extract
the viewpoint i
            %--VP = eval(['VP_' num2str(i)]);
            %VP = imresize(VP,[540 960],'bilinear');
            %% enter the function of histogram equalisation
            %VP=histeq(VP);

            VP=integralImage(ViewPointPositionr,ViewPointPositionc,:);
            %imwrite(VP,([ImageOutName1 num2str(i) type]));% write
the viewpoint image i

            subimages{r,i}=VP;% also store the viewpoint i images
            clear VP;
        end
    end

    handles.subimages=subimages;
    guidata(hObject, handles);

end

function DisplayViewpoint_X_Callback(hObject, eventdata, handles)
% hObject handle to DisplayViewpoint_X (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

```

```

% Hints:      get(hObject,'String')      returns      contents      of
DisplayViewpoint_X as text
%            str2double(get(hObject,'String')) returns contents of
DisplayViewpoint_X as a double

end
% --- Executes during object creation, after setting all properties.
function DisplayViewpoint_X_CreateFcn(hObject, eventdata, handles)
% hObject      handle to DisplayViewpoint_X (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if      ispc      &&      isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

% --- Executes on button press in Generate.
function Generate_Callback(hObject, eventdata, handles)
% hObject      handle to Generate (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
MicroImageSize = str2num(get(handles.NumberOfVP,'String'));
numViewPoint=MicroImageSize;
StartVPnumberY=str2num(get(handles.StatVPY1,'String'));
StartVPnumberX=str2num(get(handles.StartVPX1,'String'));
NoViewpoint= str2num(get(handles.VPNoInRow1,'String'));
NoShift= str2num(get(handles.Shift1,'String'));
type = get(handles.typeEx1,'String');
OutImageName= get(handles.OutputName1,'String');
%NoIntegratedImages = cell(NoViewpoint, 1);
subimages=handles.subimages;
cj=floor(size(subimages{1,1},2)* NoViewpoint);%Number of complete
cylindrical microlenses with pixels
ri=floor(size(subimages{1,1},1)* NoViewpoint);%Number of complete
cylindrical microlenses with pixels

%NoIntegratedImages = cell(NoViewpoint, 1);
%NoIntegratedImagesF = cell(1,1);

%-----

        %[NoIntegratedImagesR]=
VPIntegrationR(subimages,StartVPnumberR,NoViewpoint,NoShift,Ri);
        VP=
im2double(zeros((size(subimages{1,1},1)*(NoViewpoint))+((NoViewpoint
-1)

```



```
(NoShift+1)), (size(subimages{1,1},2)*(NoViewpoint))+((NoViewpoint-1)
* (NoShift+1)),3));
```

```
    for i = 1: NoViewpoint
        VPi = i+((i-1)*NoShift):size(VP,1);
        upsamplei=0;
        iup=0;
        for ii=1:size(subimages{1,1},1)
            iup=(NoViewpoint*(ii-1))+1;
            for iii=0:(NoViewpoint-1)
                upsamplei(iup+iii)=ii;
            end
        end
        %iii=(i-1)*shift;
        %VPi = i+((i-1)*iii):size(VP,1);
        %I = 1+(numViewpoint1 -i);
        for j = 1:NoViewpoint
            %jjj=(j-1)*shift;
            %J = 1+(numViewpoint1 -j);
            upsamplej=0;
            jup=0;
            for jj=1:size(subimages{1,1},2)
                jup=(NoViewpoint*(jj-1))+1;
                for jjjj=0:(NoViewpoint-1)
                    upsamplej(jup+jjjj)=jj;
                end
            end
            VPj = j+((j-1)*NoShift):size(VP,2);
            %VPj = j+((j-1)*iii):size(VP,2);
            VPN=
im2double(subimages{i+StartVPnumberY,j+StartVPnumberX})/(NoViewpoint
*NoViewpoint) ;
            ii=1:ri;
            jj=1:cj;
```

```
VP(VPi(ii),VPj(jj),:)=(VP(VPi(ii),VPj(jj),:))+
(VPN(upsamplei(ii),upsamplej(jj),:)));
```

```
    %VP(VPi,VPj,:)=subimages{i+19,j+19};
```

```
    end
end
```

```
%NoIntegratedImages2{1}=NoIntegratedImagesC{1};
imwrite(VP,([OutImageName 'newful' type]));
imwrite((imresize(VP,0.3)),([OutImageName 'new' type]));
```

```
end
```

```

% --- Executes on button press in DepthMap.
function DepthMap_Callback(hObject, eventdata, handles)
% hObject    handle to DepthMap (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
BW=11;%window size
l=floor(BW/2);
NoShift= str2num(get(handles.Shift,'String'));
type = get(handles.typeEx,'String');
OutImageName= get(handles.OutputName,'String');
NBN = 4;
%NB=floor([size(subimages{1},1)/BW size(subimages{1},2)/BW]);%number
of blocks line-columns
%BlockValue =(size(InterpolatedVPNo,1),1);
InterpolatedVPNo1 = cell(NoShift,1);
InterpolatedVPNo1_col = cell(NoShift,1);
for ii = 1: NoShift
    InterpolatedVPNo1{ii}=rgb2gray((im2double(imread([OutImageName
num2str(ii-1) type]))));
    InterpolatedVPNo1_col{ii}=imread([OutImageName      num2str(ii-1)
type]);
    InterpolatedVPNo1_coll{ii}=imread([OutImageName  '1'  num2str(ii-
1) type]);

end
HighResolution=          zeros(size(InterpolatedVPNo1{1},1),
size(InterpolatedVPNo1{1},2));
HighResolution_col=      uint8(zeros(size(InterpolatedVPNo1{1},1),
size(InterpolatedVPNo1{1},23),3));
HighResolution_coll=     uint8(zeros(size(InterpolatedVPNo1{1},1),
size(InterpolatedVPNo1{1},23),3));

for i=1+1:BW:size(InterpolatedVPNo1{1},1) -1
    for j=1+1:BW:size(InterpolatedVPNo1{1},2)-1
        for          NoInterpolatedVP          =          1          :
NoShift%10%size(InterpolatedVPNo1,1)

            Block=      InterpolatedVPNo1{NoInterpolatedVP}(i-1:i+1,j-
1:j+1,:);
%           I = double(Block);
%           [y x] = size(I);
%
%           Hv = [1 1 1 1 1 1 1 1 1]/9;
%           Hh = Hv';
%
%           B_Ver = imfilter(I,Hv);%blur the input image in
vertical direction
%           B_Hor = imfilter(I,Hh);%blur the input image in
horizontal direction
%
%           D_F_Ver = abs(I(:,1:x-1) - I(:,2:x));%variation of the
input image (vertical direction)
%           D_F_Hor = abs(I(1:y-1,:) - I(2:y,:));%variation of the
input image (horizontal direction)
%
%           D_B_Ver = abs(B_Ver(:,1:x-1)-B_Ver(:,2:x));%variation
of the blurred image (vertical direction)

```

```

%           D_B_Hor = abs(B_Hor(1:y-1,:)-B_Hor(2:y,:));%variation
of the blurred image (horizontal direction)
%
%           T_Ver = D_F_Ver - D_B_Ver;%difference between two
vertical variations of 2 image (input and blurred)
%           T_Hor = D_F_Hor - D_B_Hor;%difference between two
horizontal variations of 2 image (input and blurred)
%
%           V_Ver = max(0,T_Ver);
%           V_Hor = max(0,T_Hor);
%
%           S_D_Ver = sum(sum(D_F_Ver(2:y-1,2:x-1)));
%           S_D_Hor = sum(sum(D_F_Hor(2:y-1,2:x-1)));
%
%           S_V_Ver = sum(sum(V_Ver(2:y-1,2:x-1)));
%           S_V_Hor = sum(sum(V_Hor(2:y-1,2:x-1)));
%
%           blur_F_Ver = (S_D_Ver-S_V_Ver)/S_D_Ver;
%           blur_F_Hor = (S_D_Hor-S_V_Hor)/S_D_Hor;
%
%           BlockValue(NoInterpolatedVP) =
max(blur_F_Ver,blur_F_Hor);

           BlockMax = max(max(Block));
           BlockMin = min(min(Block));
           BlockValue(NoInterpolatedVP) = ((BlockMax - BlockMin)/
(BlockMax + BlockMin));
           neighbors=cell(NBN,1);
           l=floor(BW/2);
           siz = size(InterpolatedVPNo1{1,1});

           if i>BW+1
           neighbors{1}=[i-BW,j];
           else
           neighbors{1}=NaN;
           end
           if j<siz(2)-BW-1
           neighbors{2}=[i,j+BW];
           else
           neighbors{2}=NaN;
           end

           if i<siz(1)-BW-1
           neighbors{3}=[i+BW,j];
           else
           neighbors{3}=NaN;
           end

           if j>BW+1
           neighbors{4}=[i,j-BW];
           else
           neighbors{4}=NaN;
           end

           if NBN>4
           if i>BW+1&j>BW+1
           neighbors{5}=[i-BW,j-BW];
           else
           neighbors{5}=NaN;

```

```

end
if i>BW+1&j<siz(2)-BW-1
neighbors{6}=[i-BW,j+BW];
else
    neighbors{6}=NaN;
end

if i<siz(1)-BW-1&j<siz(2)-BW-1
neighbors{7}=[i+BW,j+BW];
else
    neighbors{7}=NaN;
end

if i<siz(1)-BW-1&j>BW+1
neighbors{8}=[i+BW,j-BW];
else
    neighbors{8}=NaN;
end

end

if NBN>8
    if j>2*BW+1
neighbors{9}=[i,j-2*BW];
else
    neighbors{9}=NaN;
end
if i>2*BW+1
neighbors{10}=[i-2*BW,j];
else
    neighbors{10}=NaN;
end

if j<siz(2)-2*BW-1
neighbors{11}=[i,j+2*BW];
else
    neighbors{11}=NaN;
end

if i<siz(1)-2*BW-1
neighbors{12}=[i+2*BW,j];
else
    neighbors{12}=NaN;
end

end

if NBN>12
if i<siz(1)-1&j>2*BW+1
neighbors{13}=[i+BW,j-2*BW];
else
neighbors{13}=NaN;
end

if i>BW+1&j>2*BW+1

```

```

neighbors{14}=[i-BW, j-2*BW];
else
    neighbors{14}=NaN;
end

if i>2*BW+1&j>BW+1
neighbors{15}=[i-2*BW, j-BW];
else
    neighbors{15}=NaN;
end

if i>2*BW+1&j<siz(2)-BW-1
neighbors{16}=[i-2*BW, j+BW];
else
    neighbors{16}=NaN;
end

if i>BW+1&j<siz(2)-2*BW-1
neighbors{17}=[i-BW, j+2*BW];
else
    neighbors{17}=NaN;
end

if i<siz(1)-BW-1&j<siz(2)-2*BW-1
neighbors{18}=[i+BW, j+2*BW];
else
    neighbors{18}=NaN;
end

if i<siz(1)-2*BW-1&j<siz(2)-BW-1
neighbors{19}=[i+2*BW, j+BW];
else
    neighbors{19}=NaN;
end

if i<siz(1)-2*BW-1&j>BW+1
neighbors{20}=[i+2*BW, j-BW];
else
    neighbors{20}=NaN;
end

end
%-->

if NBN>20

    if i<siz(1)-BW*2-1&j>BW*2+1
        neighbors{21}=[i+2*BW, j-2*BW];
    else
        neighbors{21}=NaN;
    end

    if i>BW*2+1&j>BW*2+1
        neighbors{22}=[i-2*BW, j-2*BW];
    else
        neighbors{22}=NaN;
    end

    if i>BW*2+1&j<siz(2)-BW*2-1

```



```

%           nBlockValue = min(blur_F_Ver,blur_F_Hor);
           nblockMax = max(max(nblock));
           nblockMin = min(min(nblock));
           nBlockValue = ((nblockMax - nblockMin)/
(nblockMax + nblockMin));

           end
           Tblocks(n) = nBlockValue;
       end

           BlockValue(NoInterpolatedVP) =
BlockValue(NoInterpolatedVP)+sum(sum(Tblocks));

           end
           [value index]= max(BlockValue);
           HighResolution(i-1:i+1,j-1:j+1,:)=
InterpolatedVPNo1{index}(i-1:i+1,j-1:j+1,:);
           HighResolution_col(i-1:i+1,j-1:j+1,:)=
InterpolatedVPNo1_col{index}(i-1:i+1,j-1:j+1,:);
           HighResolution_coll(i-1:i+1,j-1:j+1,:)=
InterpolatedVPNo1_coll{index}(i-1:i+1,j-1:j+1,:);
           depth(i,j)= index;

       end
   end

Z1=zeros(size(InterpolatedVPNo1{1},1),size(InterpolatedVPNo1{1},2));
for i=1+1:BW:size(InterpolatedVPNo1{1},1)-1
    for j=1+1:BW:size(InterpolatedVPNo1{1},2)-1

        %depthvalue=0;
        %depthvalue=(2 * 29/10)/depth(i,j);

        %Z1(i-1:i+1,j-1:j+1)=depthvalue;
        Z1(i-1:i+1,j-1:j+1)=depth(i,j);

    end
end
HighResolution_col(:,:,1)=medfilt2(HighResolution_col(:,:,1),[13
13]);
HighResolution_col(:,:,2)=medfilt2(HighResolution_col(:,:,2),[13
13]);
HighResolution_col(:,:,3)=medfilt2(HighResolution_col(:,:,3),[13
13]);

figure(6);imagesc(Z1);
figure(7);imagesc(HighResolution);
figure(11);imagesc(HighResolution_col);
figure(111);imagesc(HighResolution_coll);
save depthinformation

end

% --- Executes on button press in StereoDisparity.
function StereoDisparity_Callback(hObject, eventdata, handles)
% hObject    handle to StereoDisparity (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB

```

```

% handles      structure with handles and user data (see GUIDATA)
%MicroImageSize = str2num(get(handles.NumberOfVP, 'String'));
%numViewPoint=MicroImageSize;
method = 'SSD';
corrWindowSize=str2num(get(handles.WindowSize, 'String'));
dMax= str2num(get(handles.PixelSearch, 'String'));
dMin = 0;
%NoShift= str2num(get(handles.Shift1, 'String'));
leftImage = get(handles.StereoImage2, 'String');
rightImage= get(handles.StereoImage1, 'String');

%NoIntegratedImages = cell(NoViewpoint, 1);
%subimages=handles.subimages;
% Grab the image information (metadata) of left image using the
function imfinfo
leftImageInfo=imfinfo(leftImage);
% Grab the image information (metadata) of right image using the
function imfinfo
rightImageInfo=imfinfo(rightImage);
% Since Dense Matching is applied on a grayscale image, determine if
the
% input left image is already in grayscale or color
if(getfield(leftImageInfo, 'ColorType')== 'truecolor')
% Read an image using imread function, convert from RGB color space
to
% grayscale using rgb2gray function and assign it to variable
leftImage
    leftImage=rgb2gray(imread(leftImage));
else if(getfield(leftImageInfo, 'ColorType')== 'grayscale')
% If the image is already in grayscale, then just read it.
    leftImage=imread(leftImage);
    else
        error('The Color Type of Left Image is not acceptable.
Acceptable color types are truecolor or grayscale.');
```



```

% Check to see if both the left and right images have same number of
rows
% and columns
if(nrLeft==nrRight && ncLeft==ncRight)
else
    error('Both left and right images should have the same number of
rows and columns');
end
% Convert the left and right images from uint8 to double
leftImage=im2double(leftImage);
rightImage=im2double(rightImage);
% Check the size of window to see if it is an odd number.
if (mod(corrWindowSize,2)==0)
    error('The window size must be an odd number.');
```

end

```

% Check whether minimum disparity is less than the maximum
disparity.
if (dMin>dMax)
    error('Minimum Disparity must be less than the Maximum
disparity.');
```

end

```

% Create an image of size nrLeft and ncLeft, fill it with zeros and
assign
% it to variable dispMap
dispMap=zeros(nrLeft, ncLeft);
% Find out how many rows and columns are to the left/right/up/down
of the
% central pixel based on the window size
win=(corrWindowSize-1)/2;
% The objective of CC, NCC and ZNCC is to maximize the
% correlation score, whereas other methods try to minimize
% it.
maximize = 0;
if strcmp(method, 'NCC') || strcmp(method, 'ZNCC')
    maximize = 1;
end
%tic; % Initialize the timer to calculate the time consumed.
for(i=1+win:1:nrLeft-win)
    % For every row in Left Image
    for(j=1+win:1:ncLeft-win-dMax)
        % For every column in Left Image
        % Initialize the temporary variable to hold the previous
        % correlation score
        if(maximize)
            prevcorrScore = 0.0;
        else
            prevcorrScore = 65532;
        end
        % Initialize the temporary variable to store the best
        matched
        % disparity score
        bestMatchSoFar = dMin;
        for(d=dMin:dMax)
            % For every disparity value in x-direction
            % Construct a region with window around central/selected
            pixel in left image
            regionLeft=leftImage(i-win : i+win, j-win : j+win);
            % Construct a region with window around central/selected
            pixel in right image
            regionRight=rightImage(i-win : i+win, j+d-win :
            j+d+win);
```

```

% Calculate the local mean in left region
meanLeft = mean2(regionLeft);
% Calculate the local mean in right region
meanRight = mean2(regionRight);
% Initialize the variable to store temporarily the
correlation
% scores
tempCorrScore = zeros(size(regionLeft));
% Calculate the correlation score
if strcmp(method, 'SAD')
    tempCorrScore = abs(regionLeft - regionRight);
elseif strcmp(method, 'ZSAD')
    tempCorrScore = abs(regionLeft - meanLeft -
regionRight + meanRight);
elseif strcmp(method, 'LSAD')
    tempCorrScore = abs(regionLeft -
meanLeft/meanRight*regionRight);
elseif strcmp(method, 'SSD')
    tempCorrScore = (regionLeft - regionRight).^2;
elseif strcmp(method, 'ZSSD')
    tempCorrScore = (regionLeft - meanLeft - regionRight
+ meanRight).^2;
elseif strcmp(method, 'LSSD')
    tempCorrScore = (regionLeft -
meanLeft/meanRight*regionRight).^2;
elseif strcmp(method, 'NCC')
    % Calculate the term in the denominator (var: den)
    den =
sqrt(sum(sum(regionLeft.^2))*sum(sum(regionRight.^2)));
    tempCorrScore = regionLeft.*regionRight/den;
elseif strcmp(method, 'ZNCC')
    % Calculate the term in the denominator (var: den)
    den = sqrt(sum(sum((regionLeft -
meanLeft).^2))*sum(sum((regionRight - meanRight).^2)));
    tempCorrScore = (regionLeft -
meanLeft).*(regionRight - meanRight)/den;
end
% Compute the final score by summing the values in
tempCorrScore,
% and store it in a temporary variable signifying the
distance
% (var: corrScore)
corrScore=sum(sum(tempCorrScore));
if(maximize)
    if(corrScore>prevcorrScore)
        % If the current disparity value is greater than
        % previous one, then swap them
        prevcorrScore=corrScore;
        bestMatchSoFar=d;
    end
else
    if (prevcorrScore > corrScore)
        % If the current disparity value is less than
        % previous one, then swap them
        prevcorrScore = corrScore;
        bestMatchSoFar = d;
    end
end
end
% Store the final matched value in variable dispMap
dispMap(i, j) = bestMatchSoFar;

```

```

    end
end
% Stop the timer to calculate the time consumed.
%timeTaken=toc;
figure(3);imagesc(dispMap);
%figure(4); imshow(dispMap);
figure(5); image(dispMap);
save infodatacheck

end

function StartVP1_Callback(hObject, eventdata, handles)
% hObject    handle to StartVPY1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of StartVPY1 as text
%        str2double(get(hObject,'String')) returns contents of
StartVPY1 as a double
end

% --- Executes during object creation, after setting all properties.
function StartVP1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to StartVPY1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%        See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

function VPNoInRow1_Callback(hObject, eventdata, handles)
% hObject    handle to VPNoInRow1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of VPNoInRow1 as
text
%        str2double(get(hObject,'String')) returns contents of
VPNoInRow1 as a double
end

% --- Executes during object creation, after setting all properties.
function VPNoInRow1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to VPNoInRow1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB

```

```
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%      See ISPC and COMPUTER.
if      ispc      &&      isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

end

function Shift1_Callback(hObject, eventdata, handles)
% hObject      handle to Shift1 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Shift1 as text
%          str2double(get(hObject,'String')) returns contents of
Shift1 as a double
end

% --- Executes during object creation, after setting all properties.
function Shift1_CreateFcn(hObject, eventdata, handles)
% hObject      handle to Shift1 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%      See ISPC and COMPUTER.
if      ispc      &&      isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

function OutputName1_Callback(hObject, eventdata, handles)
% hObject      handle to OutputName1 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of OutputName1 as
text
%          str2double(get(hObject,'String')) returns contents of
OutputName1 as a double
end

% --- Executes during object creation, after setting all properties.
function OutputName1_CreateFcn(hObject, eventdata, handles)
% hObject      handle to OutputName1 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%      See ISPC and COMPUTER.
```

```

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

```

```

function typeEx1_Callback(hObject, eventdata, handles)
% hObject    handle to typeEx1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of typeEx1 as text
%         str2double(get(hObject,'String')) returns contents of
typeEx1 as a double
end

```

```

% --- Executes during object creation, after setting all properties.
function typeEx1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to typeEx1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

```

```

function WindowSize_Callback(hObject, eventdata, handles)
% hObject    handle to WindowSize (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of WindowSize as
text
%         str2double(get(hObject,'String')) returns contents of
WindowSize as a double

end

```

```

% --- Executes during object creation, after setting all properties.
function WindowSize_CreateFcn(hObject, eventdata, handles)
% hObject    handle to WindowSize (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

```

end

```
function PixelSearch_Callback(hObject, eventdata, handles)
% hObject    handle to PixelSearch (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of PixelSearch as
text
%          str2double(get(hObject,'String')) returns contents of
PixelSearch as a double
end
```

```
% --- Executes during object creation, after setting all properties.
function PixelSearch_CreateFcn(hObject, eventdata, handles)
% hObject    handle to PixelSearch (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
```

```
function StereoImage1_Callback(hObject, eventdata, handles)
% hObject    handle to StereoImage1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of StereoImage1 as
text
%          str2double(get(hObject,'String')) returns contents of
StereoImage1 as a double
end
```

```
% --- Executes during object creation, after setting all properties.
function StereoImage1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to StereoImage1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
```

```
function StereoImage2_Callback(hObject, eventdata, handles)
```

```

% hObject    handle to StereoImage2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of StereoImage2 as
text
%          str2double(get(hObject,'String')) returns contents of
StereoImage2 as a double

end
% --- Executes during object creation, after setting all properties.
function StereoImage2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to StereoImage2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

% --- Executes on button press in MultiView.
function MultiView_Callback(hObject, eventdata, handles)
% hObject    handle to MultiView (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
MicroImageSize = str2num(get(handles.NumberOfVP, 'String'));
numViewPoint=MicroImageSize;
StartVPnumber=13;
NoViewpoint= 7;
StartVPnumber1 = 18;
NoShift= 4;
H = str2num(get(handles.H, 'String'));
W = str2num(get(handles.W, 'String'));
type = get(handles.typeEx, 'String');
OutImageName= get(handles.OutputName, 'String');
Multi=9;

subimages=handles.subimages;
%-----
    for iii= 1:Multi
%         if iii >= Multi
                StartVPnumber= StartVPnumber + 1;
%
%         else
                StartVPnumber= (NoViewpoint*iii);
%         end

%-----
        cj=floor(size(subimages{1,1},2)* NoViewpoint);%Number    of
complete cylindrical microlenses with pixels
        ri=floor(size(subimages{1,1},1)* NoViewpoint);%Number    of
complete cylindrical microlenses with pixels

        %ImageOutName= ([OutImageName num2str(NumIntegral) '-']);

```

```

        %VP=0;
        %shift=6;
        %shift=0;
        VP=
im2double(zeros((size(subimages{1,1},1)*(NoViewpoint))+((NoViewpoint
-1)
*(NoShift+1)),(size(subimages{1,1},2)*(NoViewpoint))+((NoViewpoint-1)
* (NoShift+1)),3));

        for i = 1: NoViewpoint
            %iii=(i-1)*shift;
            VPi = i+((i-1)*NoShift):size(VP,1);
            %I = 1+(numViewpoint1 -i);
            for j = 1:NoViewpoint
                %jjj=(j-1)*shift;
                %J = 1+(numViewpoint1 -j);
                VPj = j+((j-1)*NoShift):size(VP,2);
                VPN=
im2double(imresize(subimages{i+StartVPnumber1,j+StartVPnumber},
[ri,cj]))/(NoViewpoint*NoViewpoint) ;
                ii=1:ri;
                jj=1:cj;

VP(VPi(ii),VPj(jj),:)=(VP(VPi(ii),VPj(jj),:))+ (VPN(ii,jj,:)));

                %VP(VPi,VPj,:)=subimages{i+19,j+19};

            end
        end
        %-----

        %NoIntegratedImages2{1}=NoIntegratedImagesC{1};
        %p=(NoViewpoint-1)*iii + ((NoViewpoint-iii)* (NoViewpoint-
1)/2);
        %NoIntegratedImages2{iii}=          NoIntegratedImages2{iii}(p:
(size(NoIntegratedImages2{iii},1)-p),p:
(size(NoIntegratedImages2{iii},2)-p),: );
        VP=imresize(VP, [H W]);
        imwrite(VP,([OutImageName 'multi' num2str(iii) type]));

    end

end

function W_Callback(hObject, eventdata, handles)
% hObject      handle to W (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of W as text

```



```

%         str2double(get(hObject,'String')) returns contents of W as
a double
end

% --- Executes during object creation, after setting all properties.
function W_CreateFcn(hObject, eventdata, handles)
% hObject    handle to W (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc      &&      isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

function H_Callback(hObject, eventdata, handles)
% hObject    handle to H (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of H as text
%         str2double(get(hObject,'String')) returns contents of H as
a double

end
% --- Executes during object creation, after setting all properties.
function H_CreateFcn(hObject, eventdata, handles)
% hObject    handle to H (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc      &&      isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

function StartVPX_Callback(hObject, eventdata, handles)
% hObject    handle to StartVPX (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of StartVPX as text
%         str2double(get(hObject,'String')) returns contents of
StartVPX as a double
end

% --- Executes during object creation, after setting all properties.
function StartVPX_CreateFcn(hObject, eventdata, handles)

```

```

% hObject    handle to StartVPX (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if    ispc    &&    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

% --- Executes on button press in NewDOF.
function NewDOF_Callback(hObject, eventdata, handles)
% hObject    handle to NewDOF (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

MicroImageSize = str2num(get(handles.NumberOfVP,'String'));
%l = floor(MicroImageSize/2);
%Image = uint8(zeros(size(integralImage)));
%    for i = 1+1:MicroImageSize:size(integralImage,1)-1
%        for j = 1+1:MicroImageSize:size(integralImage,2)-1
%            %MicroBlock=flipud(img(i-1:i+1,j-1:j+1,:));
%            %Image(i-1:i+1,j-1:j+1,:)=MicroBlock(:,:,:);
%            MicroBlock=integralImage(i-1:i+1,j-1:j+1,:);
%            block=uint8(zeros(size(MicroBlock)));
%            block(:,:,1)=flipud(MicroBlock(:,:,1));
%            block(:,:,2)=flipud(MicroBlock(:,:,2));
%            block(:,:,3)=flipud(MicroBlock(:,:,3));
%            Image(i-1:i+1,j-1:j+1,:)=block(:,:,:);
%        end
%    end
%    %handles.image=Image;
%    %guidata(hObject,handles);
%imwrite(Image, 'InvertedImage.jpg');
%integralImage=Image;
%numViewPoint=str2num(get(handles.NumberOfVP,'String'));

numViewPoint=MicroImageSize;
StartVPnumberY=str2num(get(handles.StartVPY,'String'));
StartVPnumberX=str2num(get(handles.StartVPX,'String'));
NoViewpoint= str2num(get(handles.VPNoInRow,'String'));
NoShift= str2num(get(handles.Shift,'String'));
type = get(handles.typeEx,'String');
OutImageName= get(handles.OutputName,'String');
subimages=handles.subimages;
cj=floor(size(subimages{1,1},2)* NoViewpoint);%Number of complete
cylindrical microlenses with pixels
ri=floor(size(subimages{1,1},1)* NoViewpoint);%Number of complete
cylindrical microlenses with pixels
%-----
    for iii= 0:NoShift

        %numViewpoint1= 7;

        %ImageOutName= ([OutImageName num2str(NumIntegral) '-']);

```

```

%VP=0;
%shift=6;
%shift=0;
VP=
im2double(zeros((size(subimages{1,1},1)*(NoViewpoint))+((NoViewpoint-1) * (iii+1)),(size(subimages{1,1},2)*(NoViewpoint))+((NoViewpoint-1) * (iii+1)),3));

for i = 1: NoViewpoint
    VPi = i+((i-1)*iii):size(VP,1);
    upsamlei=0;
    iup=0;
    for ii=1:size(subimages{1,1},1)
        iup=(NoViewpoint*(ii-1))+1;
        for iii=0:(NoViewpoint-1)
            upsamlei(iup+iii)=ii;
        end
    end
    %iii=(i-1)*shift;
    %VPi = i+((i-1)*iii):size(VP,1);
    %I = 1+(numViewpoint1 -i);
    for j = 1:NoViewpoint
        %jjj=(j-1)*shift;
        %J = 1+(numViewpoint1 -j);
        upsamlej=0;
        jup=0;
        for jj=1:size(subimages{1,1},2)
            jup=(NoViewpoint*(jj-1))+1;
            for jjjj=0:(NoViewpoint-1)
                upsamlej(jup+jjjj)=jj;
            end
        end
        VPj = j+((j-1)*iii):size(VP,2);
        %VPj = j+((j-1)*iii):size(VP,2);
        VPN=
im2double(subimages{i+StartVPnumbery,j+StartVPnumberx})/(NoViewpoint *NoViewpoint) ;
        ii=1:ri;
        jj=1:cj;

VP(VPi(ii),VPj(jj),:)=(VP(VPi(ii),VPj(jj),:))+
(VPN(upsamlei(ii),upsamlej(jj),:)));

        %VP(VPi,VPj,:)=subimages{i+19,j+19};

    end
end
p=(NoViewpoint-1)*iii + ((NoViewpoint-iii)* (NoViewpoint-1)/2);
%p=(NoViewpoint-1)*iii;
VP= VP(p: (size(VP,1)-p),p: (size(VP,2)-p),: );
imwrite(VP,([OutImageName '1' num2str(iii) type]));
end

```

```

end

% --- Executes on button press in Generate1.
function Generate1_Callback(hObject, eventdata, handles)
% hObject    handle to Generate1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
MicroImageSize = str2num(get(handles.NumberOfVP, 'String'));
numViewPoint=MicroImageSize;
StartVPnumberY=str2num(get(handles.StatVPY1, 'String'));
StartVPnumberX=str2num(get(handles.StartVPX1, 'String'));
NoViewpoint= str2num(get(handles.VPNoInRow1, 'String'));
NoShift= str2num(get(handles.Shift1, 'String'));
type = get(handles.typeEx1, 'String');
OutImageName= get(handles.OutputName1, 'String');
%NoIntegratedImages = cell(NoViewpoint, 1);
subimages=handles.subimages;
cj=floor(size(subimages{1,1},2)* NoViewpoint);%Number of complete
cylindrical microlenses with pixels
ri=floor(size(subimages{1,1},1)* NoViewpoint);%Number of complete
cylindrical microlenses with pixels

        %ImageOutName= ([OutImageName num2str(NumIntegral) '-']);
        %VP=0;
        %shift=6;
        %shift=0;
        VP=
im2double(zeros((size(subimages{1,1},1)*(NoViewpoint))+((NoViewpoint
-1)
(NoShift+1)), (size(subimages{1,1},2)*(NoViewpoint))+((NoViewpoint-1)
* (NoShift+1)),3));

        for i = 1: NoViewpoint
            %iii=(i-1)*shift;
            VPi = i+((i-1)*NoShift):size(VP,1);
            %I = 1+(numViewpoint1 -i);
            for j = 1:NoViewpoint
                %jjj=(j-1)*shift;
                %J = 1+(numViewpoint1 -j);
                VPj = j+((j-1)*NoShift):size(VP,2);
                VPn=
im2double(imresize(subimages{i+StartVPnumberY,j+StartVPnumberX},
[ri,cj]))/(NoViewpoint*NoViewpoint) ;
                ii=1:ri;
                jj=1:cj;

VP(VPi(ii),VPj(jj),:)=(VP(VPi(ii),VPj(jj),:))+ (VPn(ii,jj,:)));

                %VP(VPi,VPj,:)=subimages{i+19,j+19};

        end
end

```

```

imwrite(VP, ([OutImageName 'ful' type]));
imwrite((imresize(VP,0.3)), ([OutImageName type]));

end

function StartVPX1_Callback(hObject, eventdata, handles)
% hObject    handle to StartVPX1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of StartVPX1 as text
%        str2double(get(hObject,'String')) returns contents of
StartVPX1 as a double
end

% --- Executes during object creation, after setting all properties.
function StartVPX1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to StartVPX1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

% --- Executes on button press in anaglyph.
function anaglyph_Callback(hObject, eventdata, handles)
% hObject    handle to anaglyph (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
leftImage = get(handles.StereoImage2,'String');
rightImage= get(handles.StereoImage1,'String');

hIdtc = vision.ImageDataTypeConverter;
hCsc = vision.ColorSpaceConverter('Conversion','RGB to intensity');
leftI3chan = step(hIdtc,imread(leftImage));
leftI = step(hCsc,leftI3chan);
rightI3chan = step(hIdtc,imread(rightImage));
rightI = step(hCsc,rightI3chan);

figure(177), clf;
imshow(rightI3chan), title('Image');

figure(288), clf;
imshow(cat(3,rightI,leftI,leftI)), axis image;
title('anaglyph');
anaglyph=(cat(3,rightI,leftI,leftI));

```

```

imwrite(anaglyph, 'Anaglyph.jpg');
end

function StatVPY1_Callback(hObject, eventdata, handles)
% hObject      handle to StatVPY1 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of StatVPY1 as text
%         str2double(get(hObject, 'String')) returns contents of
StatVPY1 as a double
end

% --- Executes during object creation, after setting all properties.
function StatVPY1_CreateFcn(hObject, eventdata, handles)
% hObject      handle to StatVPY1 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject, 'BackgroundColor'),
get(0, 'defaultUiControlBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end
end

% --- Executes on button press in VideoRendering.
function VideoRendering_Callback(hObject, eventdata, handles)

MicroImageSize = str2num(get(handles.NumberOfVP, 'String'));
numViewPoint=MicroImageSize;
StartVPnumberY=str2num(get(handles.StartVPY, 'String'));

StartVPnumberX=str2num(get(handles.xposition1, 'String'));
StartVPnumberX1=str2num(get(handles.xposition2, 'String'));

NoViewpoint= str2num(get(handles.VPNoInRow, 'String'));
NoShift= str2num(get(handles.Shift, 'String'));
type = get(handles.typeEx, 'String');
OutImageName= get(handles.OutputName, 'String');
writerObj = VideoWriter([OutImageName '1' type], ...
                        'Uncompressed AVI');
writerObj1 = VideoWriter([OutImageName '2' type], ...
                        'Uncompressed AVI');

open(writerObj1);
open(writerObj);

% pathname=handles.pathname;
% filename=handles.filename;
Image=handles.image;
StartFrameNo = str2num(get(handles.StratFrame, 'String'));

```

```

EndFrameNo = str2num(get(handles.EndFrame, 'String'));

%integralImage=read(integralImage,FrameNo);
%integralImage=handles.image;

for start = StartFrameNo : EndFrameNo
integralImage=read(Image,start);
% at the end writeVideo(writerObj,integralImage);
num_elem_pixc=floor(size(integralImage,2)/numViewPoint)*
numViewPoint;%Number of complete cylindrical microlenses with pixels
num_elem_pixr=floor(size(integralImage,1)/numViewPoint)*
numViewPoint;%Number of complete cylindrical microlenses with pixels
subimages=cell(numViewPoint,numViewPoint);
%vpno=0;
    for r = 1 : numViewPoint
        %ImageOutName= ([OutImageName num2str(r) '-']);
        %ImageOutName1= ([OutImageName num2str(r) '-']);

        for i = 1 :numViewPoint
            %vpno= vpno+1;

            ViewPointPositionc = i:numViewPoint:num_elem_pixc;
            ViewPointPositionr = r:numViewPoint:num_elem_pixr;
            %extract the viewpoint from the integralImage and store
it onto
                %--eval(['VP_' num2str(i)
'='integralImage(ViewPointPositionr,ViewPointPositionc,:);'])%extract
the viewpoint i
                %--VP = eval(['VP_' num2str(i)]);
                %VP = imresize(VP,[540 960],'bilinear');
                %% enter the function of histogram equalisation
                %VP=histeq(VP);

VP=integralImage(ViewPointPositionr,ViewPointPositionc,:);
                %imwrite(VP,([ImageOutName1 num2str(i) type])) ;% write
the viewpoint image i

                subimages{r,i}=VP;% also store the viewpoint i images
                clear VP;
            end
        end
        % only viewpointpoint extraction;

        cj=floor(size(subimages{1,1},2)* NoViewpoint);%Number of
complete cylindrical microlenses with pixels
        ri=floor(size(subimages{1,1},1)* NoViewpoint);%Number of
complete cylindrical microlenses with pixels

        %ImageOutName= ([OutImageName num2str(NumIntegral) '-']);
        %VP=0;
        %shift=6;
        %shift=0;
        VP=
im2double(zeros((size(subimages{1,1},1)*(NoViewpoint))+(NoViewpoint
-1)

```

```

(NoShift+1)), (size(subimages{1,1},2)*(NoViewpoint))+((NoViewpoint-1)
* (NoShift+1)),3));
    VP2=
im2double(zeros((size(subimages{1,1},1)*(NoViewpoint))+((NoViewpoint
-1)
(NoShift+1)), (size(subimages{1,1},2)*(NoViewpoint))+((NoViewpoint-1)
* (NoShift+1)),3));
    for i = 1: NoViewpoint
        %iii=(i-1)*shift;
        VPi = i+((i-1)*NoShift):size(VP,1);
        %I = 1+(numViewpoint1 -i);
        for j = 1:NoViewpoint
            %jjj=(j-1)*shift;
            %J = 1+(numViewpoint1 -j);
            VPj = j+((j-1)*NoShift):size(VP,2);
            VPN=
im2double(imresize(subimages{i+StartVPnumberY, j+StartVPnumberX},
[ri,cj]))/(NoViewpoint*NoViewpoint) ;
            VPN1=
im2double(imresize(subimages{i+StartVPnumberY, j+StartVPnumberX1},
[ri,cj]))/(NoViewpoint*NoViewpoint) ;
            ii=1:ri;
            jj=1:cj;

VP(VPi(ii),VPj(jj),:)=((VP(VPi(ii),VPj(jj),:))+ (VPN(ii,jj,:)));

VP2(VPi(ii),VPj(jj),:)=((VP2(VPi(ii),VPj(jj),:))+ (VPN1(ii,jj,:)));

        %VP(VPi,VPj,:)=subimages{i+19,j+19};

    end
end

%     imwrite(VP,([OutImageName 'ful' type]));
%     imwrite((imresize(VP,0.3)),([OutImageName type]));
writeVideo(writerObj,VP);
writeVideo(writerObj1,VP2);

end
close(writerObj);
close(writerObj1);

end

function StratFrame_Callback(hObject, eventdata, handles)
% hObject     handle to StratFrame (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

```



```
% Hints: get(hObject,'String') returns contents of StratFrame as text
%          str2double(get(hObject,'String')) returns contents of
StratFrame as a double
end

% --- Executes during object creation, after setting all properties.
function StratFrame_CreateFcn(hObject, eventdata, handles)
% hObject    handle to StratFrame (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%          See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

function EndFrame_Callback(hObject, eventdata, handles)
% hObject    handle to EndFrame (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of EndFrame as text
%          str2double(get(hObject,'String')) returns contents of
EndFrame as a double
end

% --- Executes during object creation, after setting all properties.
function EndFrame_CreateFcn(hObject, eventdata, handles)
% hObject    handle to EndFrame (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%          See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

function FrameNo_Callback(hObject, eventdata, handles)
% hObject    handle to FrameNo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of FrameNo as text
%          str2double(get(hObject,'String')) returns contents of
FrameNo as a double
end
```

```
% --- Executes during object creation, after setting all properties.
function FrameNo_CreateFcn(hObject, eventdata, handles)
% hObject    handle to FrameNo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if    ispc    &&    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

function xposition1_Callback(hObject, eventdata, handles)
% hObject    handle to xposition1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of xposition1 as
text
%         str2double(get(hObject,'String')) returns contents of
xposition1 as a double
end

% --- Executes during object creation, after setting all properties.
function xposition1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to xposition1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if    ispc    &&    isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

function xposition2_Callback(hObject, eventdata, handles)
% hObject    handle to xposition2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of xposition2 as
text
%         str2double(get(hObject,'String')) returns contents of
xposition2 as a double
end

% --- Executes during object creation, after setting all properties.
function xposition2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to xposition2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%      See ISPC and COMPUTER.
if      ispc      &&      isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
```

## Appendix E: Matlab code for extracting VP

```

% --- Executes on button press in ExtractVPs.
function ExtractVPs_Callback(hObject, eventdata, handles)
% hObject      handle to ExtractVPs (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
integralImage=handles.image;
FrameNo = str2num(get(handles.FrameNo, 'String'));
integralImage=read(integralImage,FrameNo);
%integralImage=handles.image;
MicroImageSize = str2num(get(handles.NumberOfVP, 'String'));

    %handles.image=Image;
    %guidata(hObject,handles);
numViewPoint=str2num(get(handles.NumberOfVP, 'String'));
numViewPoint=MicroImageSize;
%StartVPnumber=str2num(get(handles.StartVPY1, 'String'));
%NoViewpoint= str2num(get(handles.VPNoInRow, 'String'));
%NoShift= str2num(get(handles.Shift, 'String'));
%type = get(handles.typeEx, 'String');
%OutImageName= get(handles.OutputName, 'String');
num_elem_pixc=floor(size(integralImage,2)/numViewPoint)*
numViewPoint;%Number of complete cylindrical microlenses with pixels
num_elem_pixr=floor(size(integralImage,1)/numViewPoint)*
numViewPoint;%Number of complete cylindrical microlenses with pixels
%ImageOutName= ([OutImageName num2str(NumIntegral) '-']);
subimages=cell(numViewPoint,numViewPoint);
%vpno=0;
    for r = 1 : numViewPoint
        %ImageOutName= ([OutImageName num2str(r) '-']);
        %ImageOutName1= ([OutImageName num2str(r) '-']);
        for i = 1 :numViewPoint
            %vpno= vpno+1;
            ViewPointPositionc = i:numViewPoint:num_elem_pixc;
            ViewPointPositionr = r:numViewPoint:num_elem_pixr;
            %extract the viewpoint from the integralImage and store
it onto
            %--eval(['VP_' num2str(i)
            '=integralImage(ViewPointPositionr,ViewPointPositionc,:);'])%extract
the viewpoint i
            %--VP = eval(['VP_' num2str(i)]);
            %VP = imresize(VP,[540 960],'bilinear');
            %% enter the function of histogram equalisation
            %VP=histeq(VP);

VP=integralImage(ViewPointPositionr,ViewPointPositionc,:);
            %imwrite(VP,([ImageOutName1 num2str(i) type])) ;% write
the viewpoint image i

            subimages{r,i}=VP;% also store the viewpoint i images
            clear VP;
        end
    end
handles.subimages=subimages;
guidata(hObject,handles);
end

```