Imperial College of Science, Technology and Medicine
Department of Electrical and Electronic Engineering

Estimation of the Plenoptic Function from a Swiped Image

Michael Lawson

Submitted in part fulfilment of the requirements for the degree of
Doctor of Philosophy in Electrical and Electronic Engineering of
the Diploma of Imperial College, October 2018

Statement of originality

I declare that this thesis, and the research it contains, is the product of my own work under the guidance of my thesis supervisors: Professor Pier Luigi Dragotti and Mike Brookes. Any ideas or quotations from the work of other people, published or otherwise, are fully acknowledged in accordance with the standard referencing practices of the discipline. The material of this thesis has not been submitted for any degree at any other academic or professional institution.

Copyright declaration

The copyright of this thesis rests with the author and is made available under a Creative Commons Attribution Non-Commercial No Derivatives licence. Researchers are free to copy, distribute or transmit the thesis on the condition that they attribute it, that they do not use it for commercial purposes and that they do not alter, transform or build upon it. For any reuse or redistribution, researchers must make clear to others the licence terms of this work.

Abstract

The image blurring that results from moving a camera whilst the shutter is open is normally regarded as undesirable. However, the blurring of the images contains hidden information which can be extracted to give information about the captured scene. Information encapsulated within this blur allows the light rays present within the scene to be recovered. Given the correct recovery of the light rays that form a blurred image, a sharp image of the scene can be generated for an arbitrary camera location. Therefore, rather than resharpening an image with motion blur, the goal of the work contained within this thesis is to recover the light rays within the scene

Recovery of the light rays within the scene is achieved by using a layer based model to represent objects within the scene as planes at discrete depths, and by using an extension to the level set method, to segment the blurred image into planes at different depths. Knowledge of the layer boundaries is then used to recover the surfaces of the layers, and hence a model of the scene in three dimensions. The algorithm described in this thesis has been successfully tested on real and synthetic images constructed from fronto-parallel planes, with the scene being recovered accurately from all positions within the original path of the camera.

Acknowledgements

I would like to express my sincere thanks to:

- My supervisors Mike Brookes and Pier Luigi Dragotti for their invaluable advice and support throughout the last four years.

- Everybody in the Communications and Signal Processing research group at Imperial. Without the friendship from the people on the 8th floor I would not have enjoyed doing my PhD nearly as much as I did.

- My family for always supporting me and letting me live at home for the first 2 years of the PhD.

Quotes

"I have made this letter longer than usual, only because I have not had the time to make it shorter. "

Blaise Pascal

"If you haven't lost sleep over writing a journal paper, you haven't done it correctly. "

Pier Luigi Dragotti

Contents

Statement of originality **i**

Copyright declaration **iii**

Abstract **v**

Acknowledgements **vii**

1 Introduction **7**

1.1 Motivation and Objectives 7

1.2 Outline of thesis 12

1.3 Original Contributions 12

1.4 Publications 13

2 Literature review **15**

2.1 Introduction 15

2.2 The plenoptic function 16

2.2.1 The plenoptic function and simplifications 16

2.2.2 The pinhole camera model 21

2.2.3 Reconstruction of the Plenoptic function 23

2.3 Depth from motion blur 28

2.4 Conclusion 30

3 Ideal plane EPI recovery **32**

3.1 Introduction 32

3.2 Problem setup 34

3.3 Single plane EPI recovery 37

3.3.1 The single plane swiped image 37

3.3.2 Single plane algorithm overview 38

3.3.3 Determining plane geometry 38

3.3.4 Determining plane surface texture 39

3.4 EPI recovery from a multiple plane swiped image 39

3.4.1 The multiple plane swiped image 39

3.4.2 Multiple plane algorithm overview 43

3.4.3 Multiple plane algorithm implementation 45

3.5 Uniqueness of the solution 46

3.6 Sinusoidal texture EPI recovery 47

3.6.1 The sinusoidal texture swiped image 47

3.6.2 Sinusoidal texture algorithm overview 51

3.6.3 Determining the sections for local recovery 52

3.6.4 Determining the texture frequency 53

3.6.5 Determining the switchpoint locations 54

3.7 Ideal plane algorithm results 56

3.7.1 Uniform texture plane results 56

3.7.2 Sinusoidal texture plane results 58

3.8 Conclusion 59

4 EPI recovery from real swiped images 61

4.1 Introduction 61

4.2 Algorithm Overview 63

4.2.1 Problem setup 63

4.2.2 Algorithm Description 65

4.3 EPI recovery from a swiped image 67

4.3.1 Recovering depth layers 67

4.3.2 Recovering plane silhouettes 71

4.3.3 Recovering plane surfaces 82

4.3.4 Plane recovery optimisation 87

4.3.5 EPI recovery 92

4.4 Conclusion 93

5 Evaluation Results 95

5.1 Introduction 95

5.2 Testing setup 95

5.3 Algorithm results 98

5.4 Algorithm error resilience 105

5.5 Conclusion 107

6 Conclusion **108**

6.1 Summary of Thesis Achievements 108

6.2 Future Work 109

6.2.1 Real Images 110

6.2.2 Acquisition constraints 111

6.2.3 Swiped image and depth fusion 112

6.2.4 Fluttering shutter 112

Bibliography **114**

List of Tables

5.1 Comparison of recovered images of the scene against ground truth images. . . . 103

5.2 Comparison of the SSIM results when comparing the ground truth EPI mid-point image against the recovered EPI mid-point image and against a blind deblurring algorithm [94]. 105

5.3 Comparison of recovered images of the scene against ground truth images. The true disparity of the foreground plane is 20 pixels. 106

5.4 Comparison of recovered images of the scene against ground truth images as the swiped image SNR is decreased. 107

List of Figures

1.1 An example of the range of camera movement available to view an object of interest from all angles. 8

1.2 A 'swiped image' of a scene, formed by moving a camera between two points with the shutter open (synthesised from cropped planes in [72, 38]). 10

2.1 The light rays of a plenoptic function. An image is created in the image sphere at (θ_v, ϕ_w) by light rays travelling from objects in the scene to the camera at location (V_x, V_y, V_z) at time τ and with wavelength λ (after [2]). 16

2.2 The light rays of a plenoptic function. An image is created in the image plane at (v, w) by light rays travelling from objects in the scene to the camera at location (V_x, V_y, V_z) at time τ and with wavelength λ (after [2]). 17

2.3 Rays in a light field are defined by where they pass through two parallel planes (after [52]). 18

2.4 A grid of cameras can capture a sample of the light field of a scene (taken from [92]). 19

2.5 (a) shows a view from above of a point in the scene; the rays from the scene-point to the camera for two camera positions, t_1 and t_2 intersect the image line at v_1 and v_2 relative to an origin that depends on t. The camera focal length is f. In (b), the relationship between v and t defines a straight line in the EPI whose gradient equals f/z (after [11]). 20

2.6 A panoramic photo taken by a consumer smartphone is created by stitching together photos taken as the camera is rotated. 20

2.7 A light ray forms an image at v as it intersects the image line when going from the scene surface to the camera. 21

2.8 A slice of the EPI for a real scene is shown. A vertical slice of this EPI results in a row of an image of the scene for a given camera location. In an EPI, steeper lines will occlude lines with shallower gradients (image taken from [91]). 23

2.9 The layer based model of a scene. The real life scene surface has a continuously varying depth (shown as a dashed line), but it can be adequately approximated by a series of fronto-parallel planes at discrete depths (shown as a solid line). . . 24

2.10 It has been shown for example by [69] that a real scene (such as (a)) can be well represented by a small number of discrete depth layers (such as (b)). Image taken from [69]. 25

2.11 The EPI of a scene can be decomposed into volumes grouped by depth. The images of the scene for different camera positions are seen on the left. Layers within the scene are mapped to volumes within the EPI with a common trajectory, as shown on the right (image taken from [5]). 26

2.12 The spectrum of an EPI [30] is bounded by lines defined by the minimum and maximum depths present in the scene as shown in (a). (b) shows the low pass filtering effect due to the finite pixel resolution. (c) shows the minimum sampling density of the plenoptic function to avoid aliasing in the plenoptic spectrum (image taken from [30]). 27

2.13 In [17], the image in (a) is analysed to create a depth map in (b) from motion blur. The background in (c) and the foreground in (d) are deblurred separately to produce a sharp image (image taken from [17]). 29

3.1 (a) shows a swiped image created by opening the camera shutter as the camera moves. (b) shows the images we aim to reconstruct by recovering the EPI of (a). 33

3.2 (a) and (b) show two plane scenes with uniform surface textures. Occlusions are present in (a), but not in (b). 33

3.3 An example of a scene with two planes. 35

3.4 A row of the swiped image, (a), created by integrating the EPI shown in (b) for a two plane image with constant plane texture surfaces. Each switchpoint is indicated by a white line in the EPI and by a cross in the row of the swiped image. 36

3.5 A switchpoint is labelled by the plane edge combination which created it. 37

3.6 (3.5) is derived from similar triangles in the scene geometry. The same process is used to derive (3.2) - (3.4). 38

3.7 A flowchart of the steps in the algorithm to recover an EPI from a single plane swiped image with uniform surface texture. In step (a), we determine scene geometry. In step (b), we determine the plane surface texture. Finally, in step (c), we reconstruct the EPI of the scene. 39

3.8 The dashed lines indicate the implied plane limits from the location of the switchpoints in the EPI. 40

3.9 A histogram of distances between switchpoints. As the planes in the scene are fronto parallel, the distance between switchpoints are repeated. Repeated switchpoints are identified and used to hypothesise planes in the swiped image. In the scene represented by this histogram, there are possible planes with a switchpoint difference of 144 and 350 pixels. 43

3.10 The swiped image appears to be generated from a single plane scene, but the EPI contains two planes with shared switchpoints. The further plane from the camera has twice the distance from the camera as the closer plane, but half the surface intensity. 47

3.11 The swiped image is created from a scene containing a single fronto-parallel plane with a sinusoidal texture surface. 48

3.12 The swiped image (right) formed from the EPI (left) of a fronto-parallel plane with a sinusoidal texture surface. 50

3.13 A flowchart of the steps in the algorithm to recover an EPI from a single plane swiped image. In step (a), we determine valid windows to detect plane texture frequency. In step (b), we estimate plane texture frequency. In step (c), we determine scene geometry. Finally, in step (d), we reconstruct the EPI of the scene. 52

3.14 A sampled horizontal slice of a simulated swiped image of a fronto-parallel plane with a sinusoidal texture (i.e. $N_s = 2$). 55

3.15 (a) shows a real swiped image created by opening the camera shutter as the camera moves. (b) shows the EPI recovered from (a). (c) and (d) show recreated single images of the scene from two different camera positions. 57

3.16 An example of EPI reconstruction from a swiped image. The images in (a) are combined into a swiped image (b), from which the EPI can be recovered and used to synthesise new images as in (c). 59

4.1 A swiped image made of planes with a uniform texture surface is shown in (a), whereas a swiped image made of planes that model a real life scene are shown in (b). The corresponding rows from the centre of the swiped images, shown in (c) and (d), demonstrate why the complexity of recovering the EPI increases for a real scene. Detection of switchpoints is more uncertain in a real scene, as it is unknown whether variations in intensity in the swiped image are caused by plane occlusion or surface texture. Furthermore, the switchpoints cannot be exactly paired in the real image to obtain the scene geometry. 62

4.2 A view of the scene with two planes from above, where the plane furthest from the camera is the background plane. A light ray from a plane to the camera at position t forms an image at v. The camera is swiped from x_{01} to x_{02} with the shutter open. 63

4.3 A row of the swiped image, (a), created by integrating the EPI shown in (b) for a two plane image with realistic plane texture surfaces. 65

4.4 A flowchart of the steps in the algorithm to recover an EPI from a swiped image. The switchpoints in the swiped image are shown in the image at step (a). The silhouettes of planes at each depth are shown at step (b). The surface of the planes at each depth are shown at plane (c). 66

4.5 $I_I(v, w)$, shown in (a), leads to the image of switchpoints M_s, shown in (b), which contains the repeated outlines of planes. The autocorrelation histogram shown in (c) has a large peak for each depth, which in this image is a single depth with a disparity of 30. 68

4.6 The technique in [17] is applied to the simulated swiped image in (a). This results in a depth likelihood distribution for each pixel. The peak of the depth distribution for each pixel is chosen as an estimated depth to form the estimated depth map shown in (b). 70

4.7 An image segmented into two regions (a), and the surface $\phi(\mathbf{x})$ that defines the boundary between the two regions (b) [66]. 72

4.8 (a) shows the initial ϕ boundary for a noisy image of an object and (b) shows this object being successfully segmented. 74

4.9 Pixels of a swiped image containing two planes shown in (a). In (b) green coloured observations are pixels from the foreground plane and red coloured observations are pixels from the background plane. High likelihood labelled pixels are used to predict a model to classify other pixels within the image by depth. 76

4.10 Pixels of a swiped image containing three planes at different depths shown in (a). The probability pixels are at depth 3, the background, is shown in (b), the probability pixels are at depth 2 is shown in (c) and the probability pixels are at depth 1 is shown in (d). 77

4.11 (a) shows the initial ϕ boundary for a blurred object and (b) shows this object being successfully segmented. The white outline segments the plane silhouette for the rightmost camera position. 81

4.12 (a) shows the **A** matrix for a scene with two planes without considering the effects of occlusion. (b) shows **A** for the same scene when occlusion is considered. The elements for the closer plane (on the right), are unchanged, but some elements for the further plane (on the left) are set to zero. 85

4.13 Each row of the swiped image, I_I shown in (a), can be formed by a row of the sharp image of each plane in $\tilde{\mathbf{v}}$ multiplied by **A**. The resultant row of the swiped image created in (b) is indicated in (a) by arrows. In this figure, there are two depths present within the image, related to a red and blue plane shown within $\tilde{\mathbf{v}}$. 86

4.14 The swiped image in (a) leads to the new views of the scene shown in (b) and (c). In (b), least squares estimation is used to obtain the estimate of the plane surfaces, leading to artifacts. ISTA is used to regularise the estimated plane surfaces in (c), minimising these artifacts. Swiped image constructed from cropped planes in [72, 38]. 87

4.15 The swiped image in (a) is mirrored, leading to (b). In (c), a rendered view of the scene taken from an EPI generated from (a) is seen, with artifacts at the edge of the background plane. In (d), a rendered view of the scene taken from an EPI generated from (b) is seen without artifacts caused by boundary effects. Swiped image constructed from cropped planes in [72, 38]. 88

4.16 Image of recovered planes at depth 3 (b), depth 2 (c), and depth 1 (d), from swiped image (a) when the depths and planes outlines are correct. When plane silhouettes are incorrect the recovered planes at depth 3 (e), depth 2 (f), and depth 1 (g) are incorrect. 89

4.17 The recreated swiped image from the recovered planes (b) from the swiped image (a) are estimated wrongly due to occlusion due to the swipe. 90

5.1 Swiped images were captured using camera mounted on a motorised camera slider, with the scene created from printed fronto-parallel planes arranged to form a diorama. 96

5.2 (a) shows a swiped image with two layers. (b) shows the depth probability at each disparity using the technique in Section 4.3.1. (c) shows how the number of depth layers detected varies with D_t for this image. The value of D_t is set empirically, as if it is too high then only a single layer is detected, and if it is too low then spurious depths are detected. 97

5.3 (a) shows the initial guess of Γ as a white rectangle, and (b) shows the Γ for level set method extension after 20 iterations. The leftmost plane is a background plane, so ϕ is always negative in that area. 97

5.4 (a) are simulated swiped images, which are made from two planes at different depths to form a diorama, constructed from cropped planes in [72, 38]. (b) are the ground truth depth maps, from the mid-point of the camera swipe. (c) are the estimated depth maps using the technique from [17], and (d) are the estimated depth maps using the technique in this thesis at the mid-point of the camera swipe as a comparison of techniques (taken from [50]). 98

5.5 Creation of new views of the scene from a generated swiped image. (a) shows a swiped image of a scene, (b) shows an image of the detected switchpoints, (c) shows the recovered planes from furthest on the left to closest on the right, with each depth separated by a white line, and (d) and (e) show new views of the scene rendered from different perspectives. 99

5.6 Column (a) are simulated swiped images, which are made from two planes at different depths, constructed from cropped planes in [72, 38]. Column (b) is the estimated plane silhouettes, in descending depth order, with column (c) being the recovered plane surfaces. From the recovered planes, we recover the EPIs of the scenes, and so can create novel views of the scene for different camera position, such as in column (d). 100

5.7 Column (a) are real swiped images, which are made from two planes at different depths to form a diorama, constructed from cropped planes in [72, 38]. Column (b) is the estimated plane silhouettes, in descending depth order, with column (c) being the recovered plane surfaces. From the recovered planes, we recover the EPIs of the scenes, and so can create novel views of the scene for different camera position, such as in column (d). 100

5.8 The EPI in (b) is recovered from the swiped image in (a) using the proposed algorithm from Section 4.3. 101

5.9 (a) are real swiped images, which are made from three planes at different depths to form a diorama, constructed from cropped planes in [72, 38]. (b) are the estimated plane silhouettes, in descending depth order, with (c) being the recovered plane surfaces. From the recovered planes, we recover the EPIs of the scenes, and so can create novel views of the scene for different camera position, such as in (d) and (e) (taken from [50]). 102

5.10 (a) are simulated swiped images containing four planes, which are made from four planes at different depths to form a diorama, constructed from cropped planes in [72, 38]. (b) are the estimated plane silhouettes, in descending depth order, with (c) being the recovered plane surfaces. From the recovered planes, we recover the EPIs of the scenes, and so can create novel views of the scene for different camera position, such as in (d) and (e) (taken from [50]). 103

5.11 A graph plotting the SSIM of the recovered EPI against the ground truth EPI as the maximum disparity in the swiped image is varied (taken from [50]). . . . 104

5.12 Column (a) are simulated swiped images, which are made from two planes at different depths to form a diorama, constructed from cropped planes in [72, 38]. Column (b) are the images from (a) deblurred using the software from [39], which implements the technique from [94]. Column (c) are the newly rendered images from the EPI estimated from (a) using the techniques outlined in this thesis. . . 105

5.13 A plot of recovered EPI SSIM against detected disparity for the foreground plane, where the true disparity for the foreground plane is 20 pixels. 106

5.14 A plot of recovered EPI SSIM against SNR of the swiped image. 107

6.1 An example of a real image of a scene, where the EPI shown in (a) results in the swiped image shown in (b). Although it has been assumed in this thesis that the layer based model may be used, real scenes such as this have continuous variations in depth. Furthermore, it has been assumed that colour can be used to classify pixels by depth, whereas this scene has similar colours occurring at different depths. 110

6.2 (a) is a swiped image of a scene, whilst (b) and (c) are coded exposures. (e) to (g) are the deblurred images of the swiped images above. (e) exhibits artifacts in the deblurred image, whereas (f) and (g) do not, as there are no zeros in the Fourier transform of the filter for the coded exposure (taken from [71]). 113

List of Abbreviations

- AWGN - Additive White Gaussian Noise
- CGI - Computer Generated Imagery
- DIBR - Depth Image Based Rendering
- EPI - Epipolar Plane Image
- GPU - Graphics Processing Unit
- IBR - Image Based Rendering
- LDA - Linear Discriminant Analysis
- RGB - Red Green Blue
- RGB - Red Green Blue Depth
- SNR - Signal-to-noise ratio
- SSIM - Structural Similarity Index
- 2D - Two Dimensional
- 3D - Three Dimensional

List of Symbols

- $\mathbf{A}$ - Swiping matrix
- $\mathbf{A}_p$ - Swiping matrix for plane p
- c^- and c^+ - Averages in level set
- C_b - Averages for blurred segments
- d - Indexes the segments of $I_I(v)$ between consecutive switchpoints
- $d_d(v, w, l)$ - Depth probability distribution for each pixel in the swiped image, calculated using [17]
- D_t - Threshold for plane detection in autocorrelation of switchpoints
- $E(\Omega^+, \Omega^-)$ - Energy in level set
- f - Focal length of camera
- $G(z)$ - Annihilating filter for moments of $p_I{}'[k]$
- $H_{\mathbf{u}}(z)$ - Annihilating filter for $\mathbf{P_I}$ matrix
- $h_c(l)$ - Combined histogram of depth likelihoods
- $\bar{h}_c(l)$ - Smoothed combined histogram of depth likelihoods
- $h_d(l)$ - Histogram of depth likelihood derived from [17]
- h_p - Blurring kernel for depth p

- $I(v, w, t)$ - EPI
- $I(v, w, s, t)$ - Light field
- $I(V_x, V_y, V_z, v, w, \lambda, \tau)$ - Cartesian form of plenoptic function
- $I(V_x, V_y, V_z, \theta_v, \phi_w, \lambda, \tau)$ - Angular form of plenoptic function
- $I_I(w, v)$ - Swiped image
- $I_{I,p}$ - Swiped image of plane p
- $\hat{I}_I(v)$ - A stream of differentiated Diracs at the switchpoint locations
- k - Pixel index in sampled swiped image
- k_a - Moving average filter
- l - Lag (disparity) of switchpoints
- $L_{-,1}$ - The probability that the pixel if ϕ is negative at that location fits the LDA generated model at the depth plane the pixel is in
- $L_{-,2}$ - The maximum probability that the pixel if ϕ is negative at that location fits the LDA generated model at the depth plane the pixel is not in
- M_s - Image of switchpoints
- N - Number of pixels
- N_s - Number of sinusoids
- p - Plane index
- P - Number of planes
- $p_I[k]$ - Pixels of swiped image
- $p_I{}'[k]$ - Filtered pixels of swiped image
- $\mathbf{P_I}$ - Toeplitz matrix created from pixels of swiped image

- q_s - Phase of sinusoidal texture surface
- $R_{vv}(l)$ - Autocorrelation for each row of switchpoints
- R_l - Threshold for depth layer confidence
- s_p - Curvllinear coordinate of point on plane
- t - Camera horizontal spatial coordinate (x-axis) in light field and EPI
- T_I - Spacing between pixels
- u - Camera vertical spatial coordinate (y-axis) in light field and EPI
- u_n - Weighted exponential component of a signal
- v - Image plane horizontal spatial coordinate (x-axis)
- $\tilde{\mathbf{v}}$ - Sharp image of planes
- $V(\mathbf{x})$ - Image of depth probabilities
- $V_{b_{p-}}(\mathbf{x})$ and $V_{b_{p+}}(\mathbf{x})$ - Contributions to swiped segments from each plane
- $V_c(\phi(\mathbf{x}))$ - Approximation of swiped image
- v_m - Horizontal field of view of the camera
- $\mathbf{v}_{ip}$ - ith pixel labelled with depth p
- $\tilde{\mathbf{v}}_p$ - Sharp image of plane p
- V_s - Number of switchpoints
- V_x - Camera horizontal spatial coordinate (x-axis) in full plenoptic function
- V_y - Camera vertical spatial coordinate (y-axis) in full plenoptic function
- V_z - Camera depth spatial coordinate (x-axis) in full plenoptic function
- $V_\phi(\mathbf{x})$ - Plane mask

- w - Image plane vertical spatial coordinate (y-axis)
- w_a - Length of the moving average filter
- w_m - Vertical field of view of the camera
- x - World horizontal spatial coordinate (x-axis)
- $\mathbf{x}$ - Pixel index in image of depth probabilities
- x_{ij} - Plane end points
- x_v - Pixel index in the v axis
- x_w - Pixel index in the w axis
- y - World vertical spatial coordinate (y-axis)
- z - World depth spatial coordinate (z-axis)
- z_p - Depth of pth plane
- α_m - The frequencies reproduced by the e-spline sampling kernel, $\beta_{\mathbf{u}}(k)$
- $\beta_{\mathbf{u}}$ - E-spline sampling kernel
- Γ - Boundary in level set
- θ_v - Polar angle in image sphere
- κ - Mean curvature of surface
- λ -Wavelength
- λ_f - Spacing between frequencies
- λ_1 - Weighting of $\frac{\partial E}{\partial \phi}$
- λ_2 - Weighting of mean curvature
- μ - Switchpoint mean

- μ_p - Sample mean
- ρ - Order of linear component of signal
- σ^2 - Switchpoint variance
- $\mathbf{\Sigma}_p$ - Sample covariance
- τ -Time index
- τ_I - Moments of a signal
- ϕ - Surface of level set function
- φ_I - Sampling kernel for swiped image acquisition
- ϕ_w - Azimuth angle in image sphere
- $\phi_+(\mathbf{x})$ - ϕ at the current iteration where the value of ϕ at $\mathbf{x}$ is positive
- $\phi_-(\mathbf{x})$ - ϕ at the current iteration where the value of ϕ at $\mathbf{x}$ is negative
- $\xi_d(t)$ - Heaviside step function
- ω_n - Plane surface texture frequencies
- Ω^- and Ω^+ - Regions in level set

Chapter 1

Introduction

1.1 Motivation and Objectives

An increasing quantity of visual media is generated every year, as cameras proliferate from standalone devices to being included in most consumer electronics, such as smartphones and tablets. Camera modules have become inexpensive enough over time to be a required part of any smart device. Despite increases in the quantity and quality of images produced, the mainstream format remains unchanged from two dimensional colour pictures. Extending media consumption to three dimensions is frequently used as a way to increase the immersiveness of the scene [46]. Many modern televisions incorporate 3D technology, and stereoscopic virtual reality headsets are becoming steadily more popular [32].

Potential commercial applications of the 3D view of a scene include virtual reality shopping, the viewing of sculptures in art installations or the following of figures in sports events [40, 10]. With 3D information about the scene, it becomes possible for the viewer to have control over the point of view of the camera and can explore multiple perspectives, with an example of an object able to be viewed from all angles being seen in Figure 1.1. These applications are still being rapidly developed as more complex representations of scenes become available.

Capturing a scene in three dimensions has required images of the scene at multiple viewpoints.



Figure 1.1: An example of the range of camera movement available to view an object of interest from all angles.

Advances in technology have enabled great increases in the availability of digital storage and transmission bandwidth, but the amount of data generated by taking a large number of views of a scene is orders of magnitude greater than that required for a 2D representation of the scene. In addition, capturing a multiview representation of a scene requires specialised equipment, such as a large array of cameras or a camera with multiple lenses or sensors. For instance, many films in cinemas are now shot using 3D camera, with two cameras that are spaced apart at roughly the distance between human eyes. Plenoptic cameras, featuring a multi-lens array and a grid of sensors, have recently become available to capture multiple views of the scene from a single camera [64].

Whilst there are many methods for capturing 3D views of a scene, most of these are not practical for widespread consumer devices. Having an array of cameras increases the cost of a consumer device with the additional number of camera modules included, with the benefit being limited because of the small size of the device. Plenoptic cameras using lens arrays are also bulky and expensive. Although in recent years technology improvements have resulted in a reduction of

the cost of cameras, the low margins and the aim of minimising size has meant that almost all current smartphones have a single camera, and the smartphones containing multiple cameras overwhelmingly have only two cameras. Consumer devices that contain two or more cameras are able to produce a depth map of the scene, but are limited in recovering views of the scene from different perspectives by the typically very small distance between the cameras. In this work, we focus on a technique to obtain a view of the scene in which the viewer is able to move a virtual camera within the scene to change the point of view. This is obtained using a camera with a single lens and sensor, a setup found widely on consumer devices.

In addition to consumer uses, although the financial and space constraints for industrial devices are typically not as high, there exist potential uses for this technology for uses where there are limitations in acquiring images. Non-consumer applications could include remote devices, such as deep sea remote vehicles or micro-satellites, where the small and remote nature of the technology mean that grids of cameras, and transmitting large arrays of images are not feasible. In addition, many remote devices have limited control over movement, as a remote probe may be beyond the direct control of human operators, therefore making the recovery of the scene from multiple viewpoints useful for surveying larger areas.

There are two existing methods of giving a viewer the ability to change their view of a scene: the model-based approach and Image Based Rendering (IBR) [76]. The model-based approach is where a view of the scene is created using models of objects, light sources and textures [74]. This is the method used for animating CGI films and 3D video games. In the model-based approach, a 3D model of the scene is created, either by creating a scene entirely within a computer or by estimating the locations of objects within a real world scene. Each of the objects has a surface with a texture, and is lit using a modelled light source within the scene. The advantage of using this approach is the ability to create new views by projecting the objects to wherever the virtual camera is situated.

Use of the model-based method is highly computationally intensive, with expensive GPU units needed to render modern computer games at an acceptable frame rate, and large scale rendering farms used to create animated movies [74]. Estimating the geometric model of a real world

scene is a well-studied, but very difficult problem. Various methods have used range finding equipment, such as laser distance measurement or ultrasonic sensors, or have alternately used an estimate from computer vision algorithms [61, 42, 73]. With complex scenes containing many objects, it is not easy to build a complete model of the scene.

An alternative to the model-based approach is Image Based Rendering [60]. For Image Based Rendering the scene is represented not as a model of a scene, but as rays of light captured by cameras at different viewpoints. To accomplish this, the scene must be captured by taking images with an array of cameras, or, for a static scene, with a single movable camera taking multiple images. To create novel views of the scene with a virtual camera placed at an unsampled location at which an image is unavailable, the light rays that make up the image are interpolated from those captured by images taken from real camera locations. Although computationally less intensive than the model-based method, a large number of images are normally required to render the scene accurately. Previous work has been undertaken to exploit the redundancy in this array of images to obtain accurate results with fewer images [16].

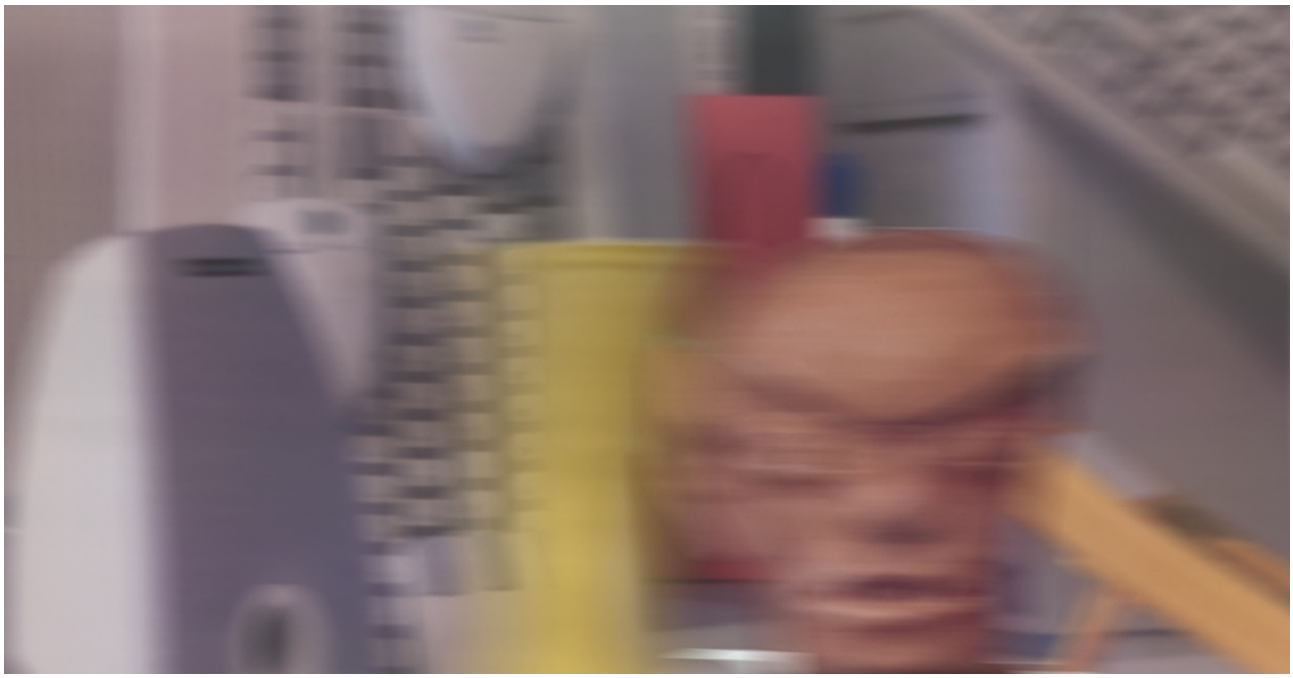


Figure 1.2: A 'swiped image' of a scene, formed by moving a camera between two points with the shutter open (synthesised from cropped planes in [72, 38]).

A common feature of previous work in representing a scene from multiple perspectives is that multiple input images are required. In this thesis, the goal is to use a single input image to

recover a representation of a scene that encapsulates all perspectives as a camera travels along a line. This is known specifically as the Epipolar Plane Image (EPI) of a scene, which will be discussed in more detail in Chapter 2. Figure 1.2 shows a synthesised image that would result from moving a camera along a horizontal line while the shutter is held open. The image blurring that results from moving a camera whilst the shutter is open is normally regarded as undesirable. However, the blurring of the images contains hidden information which can be extracted to give information about the captured scene. A 'swiped image' is one that includes motion blur arising from motion of the camera with the shutter open. A swiped image is generated from the integration of the light rays present in the scene as the camera moves. Motion blur within the swiped image is due to the combination of many light rays arriving at different viewpoints. Rather than merely resharpening the image, the goal of this thesis is to retrieve the images at these different viewpoints. Due to the integration of all the light rays received during the exposure, this problem of recovering the EPI is a challenging inverse problem. If the light rays that created the swiped image can be determined, then new views of the scene can be synthesised by creating a virtual camera at any chosen position.

The technique presented in this thesis assumes a highly controlled acquisition, where the swiped image is obtained using a motorised camera slider. This is the ideal case for camera movement, reducing the number of free parameters that need to be estimated for views of the scene to be reconstructed. A key contribution of this thesis is a more accurate forward model of the acquisition of the swiped image, which uses knowledge of the scene geometry and the movement of the camera. Although this is not something that can be casually obtained by using a hand movement with a typical consumer device, the forward model of the acquisition could be further developed by adapting it to incorporate knowledge of non-uniform camera movement. A method of estimating camera movement already exists using the inertial sensors and accelerometers in smartphones, and are already used for high precision inputs to games [83].

1.2 Outline of thesis

The outline of this thesis is as follows: Chapter 2 presents a literature review of the state of the art in research in this area. Chapter 3 demonstrates an algorithm that allows new views of a scene to be rendered from a swiped image of a scene containing uniform texture planes. This is extended in Chapter 4 which discusses a method for rendering new views of a scene which contains planes representing real objects from a swiped image. Chapter 5 evaluates the results of the algorithm from Chapter 4 on both synthetic and real images. Finally, Chapter 6 concludes and explores possible directions of future work on this subject.

1.3 Original Contributions

This thesis includes the following original contributions:

1. **Recovery of EPI for multiple fronto-parallel planes with uniform texture surfaces** - A scene with multiple fronto-parallel planes will create a swiped image that is comprised of a piecewise linear signal. We show that it is possible to find the EPI for a scene that leads to a unique EPI given an exhaustive combinatorial search. The complexity of this exhaustive search is reduced by eliminating impossible combinations.

2. **Recovery of EPI for ideal single planes with sinusoidal texture surfaces** - A technique is outlined that enables the recovery of an EPI given a scene with a single plane that has a surface which is a sum of sinusoids. Moreover, through the use of finite rate of innovation (FRI) theory, we show that the EPI can be recovered after the signal has been subsampled after being distorted through an acquisition kernel.

3. **Depth estimation from a swiped image** - Estimation of the depths present within a swiped image is achieved through the use of the detection of 'switchpoints' within a swiped image. We infer scene geometry by correctly pairing switchpoints caused by the motion of the camera whilst the shutter is open.

4. **Plane texture recovery** - Existing methods for deblurring a swiped image recover a single sharp image. Areas of objects which are occluded by objects closer to the camera cannot be seen in a deblurred image. Using the proposed algorithm, we recover all the plane surface textures that are seen at any point during the camera swipe, enabling a viewer to 'look behind' occluding objects.

5. **Plane recovery optimisation** - Existing methods for obtaining depth from motion blur usually operate in estimating depth in a window around each pixel. Therefore boundaries of depth layers are usually poorly defined. In the proposed algorithm, we optimise the boundaries of the recovered planes to produce the maximum amount of colour separation between the depth layers.

6. **Recovery of EPI for real life images** - Recent techniques have been developed to recover the EPI of a scene, enabling alias free rendering of novel views of scenes from a grid of images. In Chapter 4 we demonstrate a technique where the EPI of a scene can be recovered from a single swiped image generated from a camera moving with the shutter open. Furthermore, from this EPI, novel views of the scene can be rendered from an arbitrary camera position.

1.4 Publications

The work in this thesis has led to the following publications:

1. Michael Lawson, Mike Brookes, and Pier Luigi Dragotti, "Capturing the plenoptic function in a swipe," *SPIE Optical Engineering+ Applications*, pages 99710O- 99710O. International Society for Optics and Photonics, SPIE, 2016 [48]

2. Michael Lawson, Mike Brookes, and Pier Luigi Dragotti, "Identifying a multiple plane plenoptic function from a swiped image," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1423-1427. IEEE, 2017. [49]

3. **Submitted for publication in IEEE Transactions on Computational Imaging** Michael Lawson, Mike Brookes, and Pier Luigi Dragotti, Scene Estimation from a Swiped Image. In *IEEE Transactions on Computational Imaging (submitted)*, 2018. [50]

Chapter 2

Literature review

2.1 Introduction

To be able to render new images of a scene from a single image, a method of characterising the visual information in a scene is needed. The visual data which is acquired by viewing the scene from different viewpoints may be parameterised by the plenoptic function. The plenoptic function is a way of representing a scene not by the objects that it contains, but by the light rays that are present. The plenoptic function was first introduced by Adelson and Bergen in [2], and describes a scene in terms of light rays observed by a camera at an arbitrary location.

In this chapter we discuss the plenoptic function, how the dimensionality of the full plenoptic function may be reduced, and include an overview of recent results in Section 2.2. In Section 2.3, the state of the art of detecting depth from images with motion blur is discussed. Finally, the conclusion is in Section 2.4.

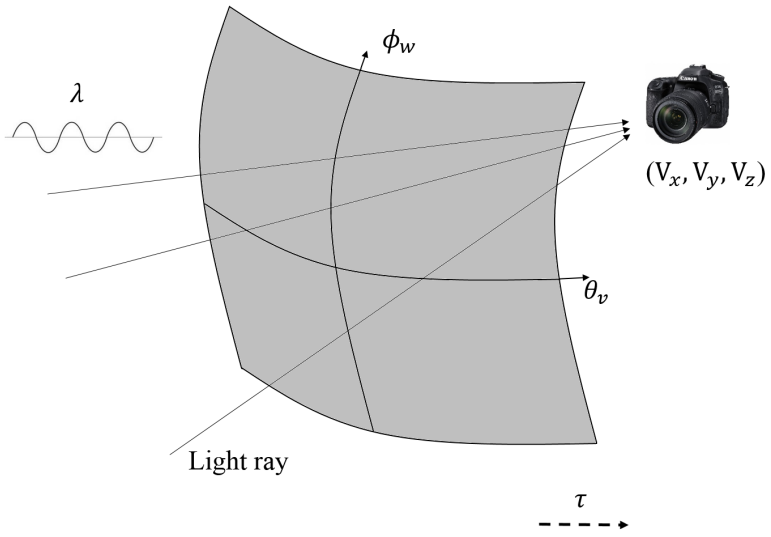


Figure 2.1: The light rays of a plenoptic function. An image is created in the image sphere at (θ_v, ϕ_w) by light rays travelling from objects in the scene to the camera at location (V_x, V_y, V_z) at time τ and with wavelength λ (after [2]).

2.2 The plenoptic function

2.2.1 The plenoptic function and simplifications

The plenoptic function, first introduced by Adelson and Bergen in [2], expresses the intensity of the light rays emanating from a point in a scene at (x, y, z) as a function of seven variables, namely: (V_x, V_y, V_z), the viewing position of the camera; (θ_v, ϕ_w), the pixel location within the image plane in angular coordinates; λ, the wavelength of the light ray; and τ is the time. The scene in the plenoptic function is defined by the intersections of the light rays with a surface. With this definition of the scene, it is assumed that there is no fading of the light rays due to range, and there are no changed of medium to cause refraction. As a result, the plenoptic function can be written as:

$$I = I_7(V_x, V_y, V_z, \theta_v, \phi_w, \lambda, \tau). \tag{2.1}$$

When a camera takes a sharp image of the scene, it is obtaining a sample of the plenoptic

function, as shown in Figure 2.1. When there is access to the continuous version of the plenoptic function, an image from any location can be rendered by the selection of the light rays needed to create the required image. Use of angular coordinates enables the light ray to come from any point in space towards the camera, but when only an image plane is necessary the light rays may be parameterized by Cartesian coordinates, as shown in Figure 2.2. This approach is commonly used in computer graphics and machine vision algorithms for simplicity. Use of the Cartesian coordinate system to define an image plane with the axes (v, w) results in the following representation of the plenoptic function:

$$I = I_7(V_x, V_y, V_z, v, w, \lambda, \tau) \tag{2.2}$$

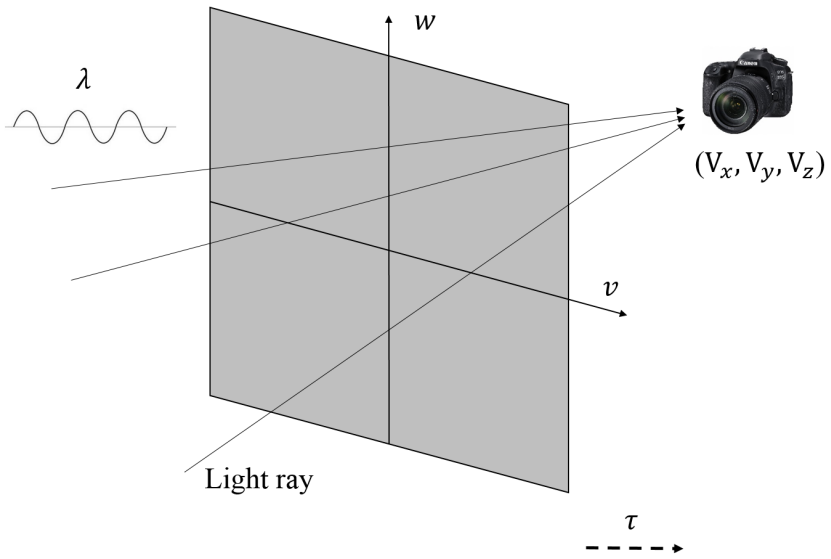


Figure 2.2: The light rays of a plenoptic function. An image is created in the image plane at (v, w) by light rays travelling from objects in the scene to the camera at location (V_x, V_y, V_z) at time τ and with wavelength λ (after [2]).

The full plenoptic function is complicated to analyse due to its high dimensionality. Having seven dimensions makes derivations of mathematical properties difficult. Furthermore, a very large number of sharp images are required to sample all the dimensions of the plenoptic function adequately, with the minimum sampling density without aliasing being analysed in [88, 89].

Despite recent advances in the field of data storage, the number of images required to adequately sample the plenoptic function constitutes an overly large amount of data, and also means that practical samplings of the plenoptic function require a lot of time and effort.

As the full plenoptic function is so complex to analyse, most work undertaken in this area has applied restrictions to the plenoptic function in order to reduce its dimensionality. Assumptions about the scene can be made to reduce the number of dimensions, such as constraining the camera movement or discarding τ by assuming the scene is static.

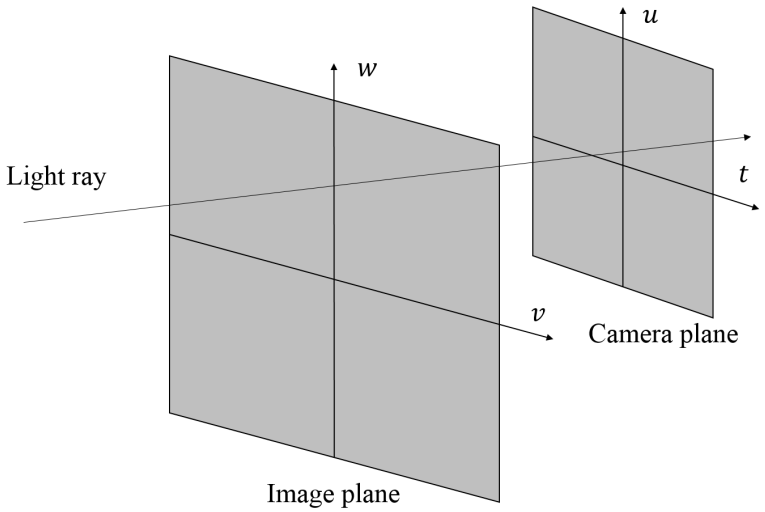


Figure 2.3: Rays in a light field are defined by where they pass through two parallel planes (after [52]).

A widely used simplification to the plenoptic function is the light field, which was introduced by Levoy and Hanarahan [52], and is alternatively known as the lumigraph [33]. In the light field, time and wavelength are disregarded, camera motion is restricted to a plane and the scene is assumed to lie within a bounding box, limiting the light field plenoptic function to 4 parameters:

$$I = I_4(t, u, v, w). \tag{2.3}$$

Each ray of light is defined by its intersections with two parallel planes, as shown in Figure 2.3.

The plane with the coordinates (v, w) is the image plane, and the plane with the coordinates (t, u) is the camera plane.

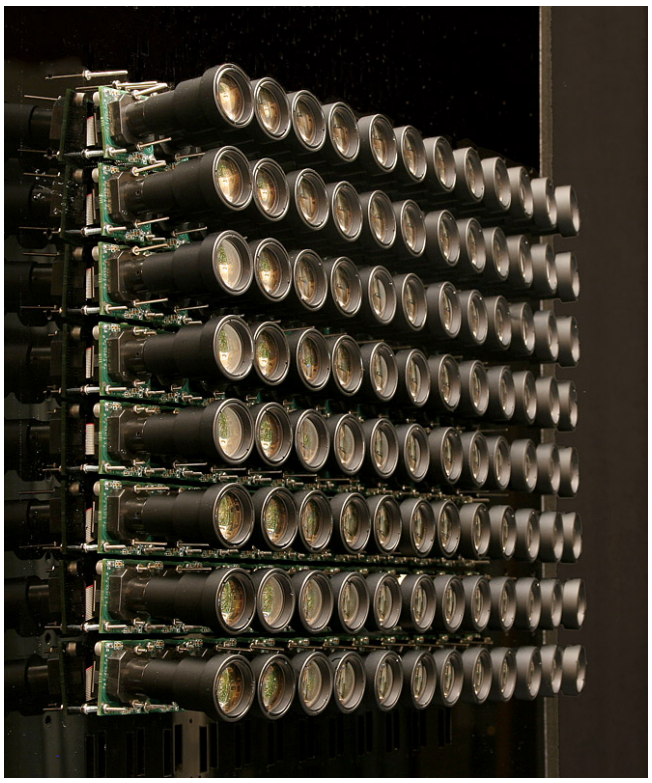


Figure 2.4: A grid of cameras can capture a sample of the light field of a scene (taken from [92]).

A sampling of the light field of a scene can be accomplished by using a grid of images of the scene [63, 64]. A grid of images may be obtained either by a dedicated plenoptic camera or by using an array of cameras, such as that shown in Figure 2.4, to take multiple pictures of the scene.

A further simplification to the plenoptic function can be made by restricting the camera movement to a horizontal line; this leads to the Epipolar Plane Image (EPI) [11]. In the EPI model, wavelength and time are also omitted, resulting in a 3 parameter function $I(t, v, w)$:

$$I = I_3(t, v, w) \tag{2.4}$$

where the origin of the image plane now moves with the camera, i.e. $v = w = 0$ denotes a light

ray that is perpendicular to the camera plane for any value of t. In the EPI, a point in space, such as shown in Figure 2.5(a), becomes a line in the EPI, as shown in Figure 2.5(b), which has a gradient inversely proportional to the depth of the point.

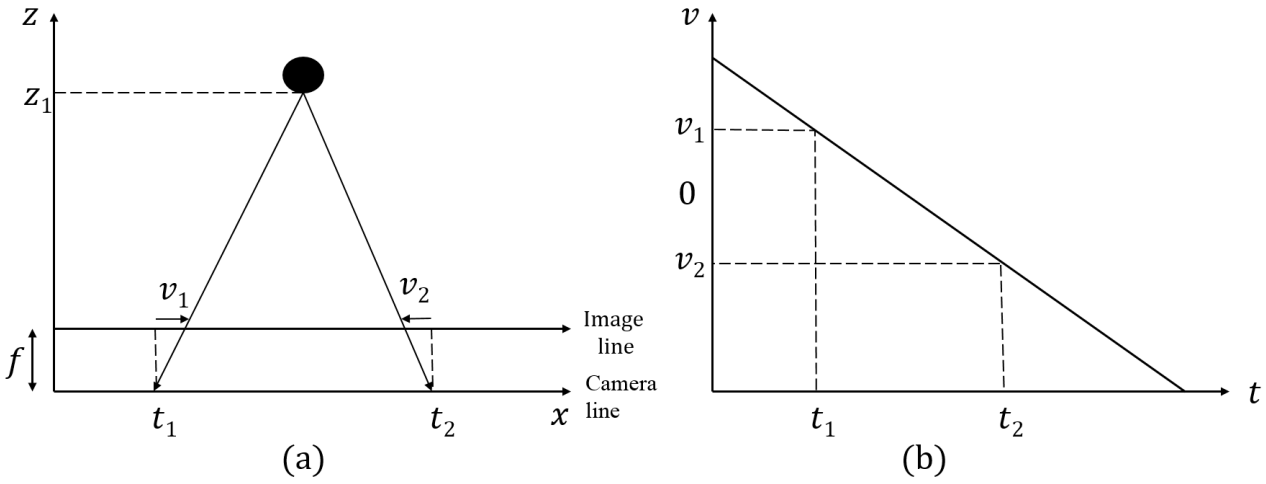


Figure 2.5: (a) shows a view from above of a point in the scene; the rays from the scene-point to the camera for two camera positions, t_1 and t_2 intersect the image line at v_1 and v_2 relative to an origin that depends on t. The camera focal length is f. In (b), the relationship between v and t defines a straight line in the EPI whose gradient equals f/z (after [11]).

In the most restricted form of the plenoptic function, the camera location is fixed; this has been called panoramic photos or mosaic photos [96]. Here, the camera rotates whilst fixed in a location, with multiple images being stitched together to form an image of a scene with a wide field of view. This form of images is widely available in consumer devices because it can be created without using specialist hardware, as illustrated in Figure 2.6, which shows a panoramic photo taken with a smartphone.



Figure 2.6: A panoramic photo taken by a consumer smartphone is created by stitching together photos taken as the camera is rotated.

2.2.2 The pinhole camera model

When analysing the EPI of a scene, a pinhole camera model can be used. Figure 2.7 illustrates the scene viewed from above in which a point in space has coordinates (x, z), the camera centre has coordinates $(t, 0)$. In this view, the image plane forms a line at $z = f$, the focal length, which is the distance between the camera and the image plane [11]. As a ray of light goes from a point in the scene to the camera, an image is formed where the light intersects with the image plane, which can be seen in Figure 2.7.

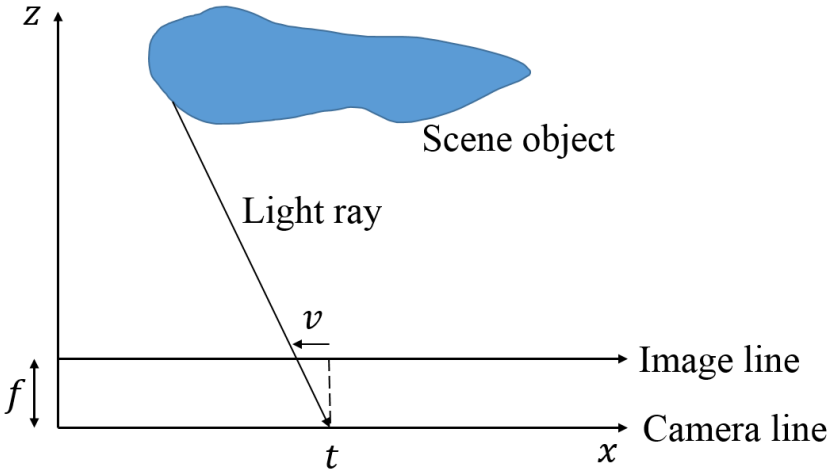


Figure 2.7: A light ray forms an image at v as it intersects the image line when going from the scene surface to the camera.

In the EPI model, the camera can only move in a horizontal direction along the x axis, as illustrated in Figure 2.7, the light ray from a point in the scene will intersect the image plane at a point whose x-coordinate, v, depends on the camera position, t. A point within the scene will become a straight line in the EPI [11], with the mapping being:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \mapsto \begin{pmatrix} t \\ v \\ w \end{pmatrix} = \begin{pmatrix} t \\ fx/z - ft/z \\ fy/z \end{pmatrix}. \tag{2.5}$$

If the camera is moved to a different position, the light ray from the same point that arrives at

the camera will intersect the image line in a different location. Similar triangles can be used to find the shift in the position of the point in the image, given the shift in camera position:

$$v' = v - \frac{f(t' - t)}{z} \tag{2.6}$$

$$w' = w \tag{2.7}$$

where v' and w' are the new position of the intersection of the light ray and the image line after the camera has been shifted from location t to location t'.

Equation (2.6) demonstrates that a shift in camera position will produce a corresponding shift in the perceived position of a point within the scene. Since w does not vary with t, each horizontal line in the image can be treated independently. Therefore, under the assumption that plane surfaces are lambertian, the equation can be written [11]:

$$I(t', v', w') = I(t, v - \frac{f(t' - t)}{z}, w). \tag{2.8}$$

A point in space corresponds to a line within the EPI, with a gradient that is proportional to $1/z$, and (2.8) shows this has an intensity which is constant along the line. The perceived shift in the position of a point for a given camera shift is known as its disparity.

When there is more than one object within the EPI, a closer object will occlude a further object. Within the EPI, this means that lines with a steeper gradient will occlude lines with a shallower gradient; this is illustrated in Figure 2.8 (reproduced from [91]) which shows part of the EPI from a real scene.

The properties outlined above demonstrate that the EPI of a scene is highly structured; this property can be used for the reconstruction of an EPI from incomplete information, such as a swiped image, or in situations such as a low-resolution sampling of the EPI [5, 29, 28, 69].



Figure 2.8: A slice of the EPI for a real scene is shown. A vertical slice of this EPI results in a row of an image of the scene for a given camera location. In an EPI, steeper lines will occlude lines with shallower gradients (image taken from [91]).

2.2.3 Reconstruction of the Plenoptic function

Recovery of the plenoptic function of a scene is a problem that has been well studied in recent years. The parametrisation of a scene by the plenoptic function has, in particular, been used to perform Image Based Rendering (IBR), in which an image is rendered from a new viewpoint by estimating the light rays from samples of the plenoptic function [60].

Real-life scenes contain a range of depths within them that are continuous for non-planar objects within the scene. When using Image Based Rendering, a representation of depths of objects within a scene has been employed in which every point in a scene surface has a defined (x, y, z) position. However, artifacts can appear in new rendered images that are caused by the varying depths of objects and complex occlusion ordering [95]. A way of reducing the complexity of the modelling of objects within the scene is to represent them as fronto-parallel planes with discrete depths. This is known as a layer based model and is illustrated in Figure 2.9 in which

the continuously varying depth (dashed line) has been quantised to three discrete depth layers. Each point on a scene surface is assigned to the layer with the closest depth.

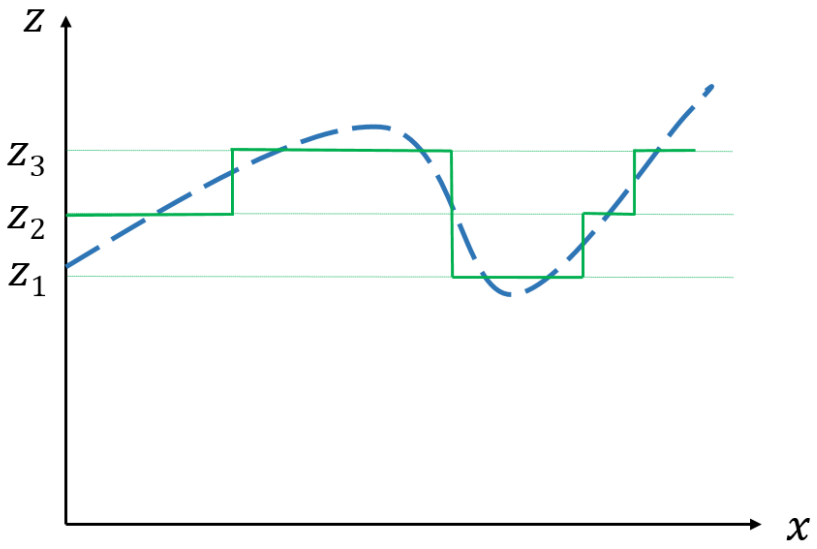
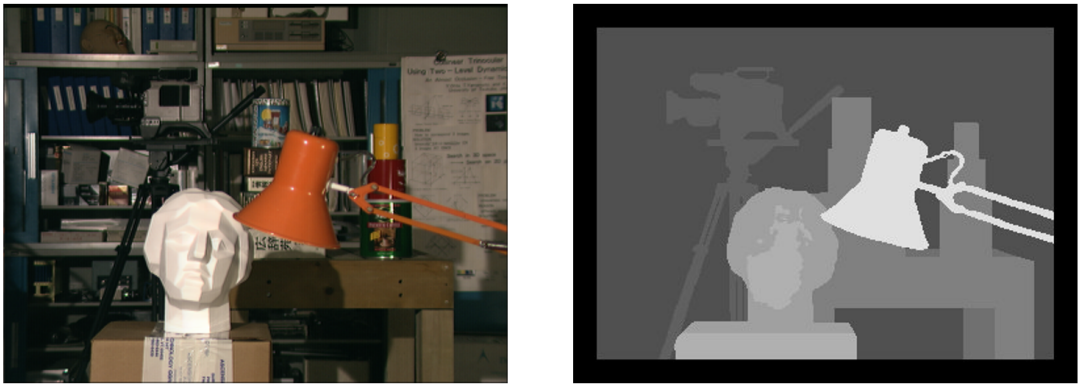


Figure 2.9: The layer based model of a scene. The real life scene surface has a continuously varying depth (shown as a dashed line), but it can be adequately approximated by a series of fronto-parallel planes at discrete depths (shown as a solid line).

The use of a layer based representation was first introduced by Wang and Adelson [88, 89] and it has been shown that photorealistic images can be reconstructed using a layer-based model of the scene in which object depths are coarsely quantised [5, 29, 28, 69]. An example is the scene in Figure 2.10(a) in which the depths can be represented by the layers in Figure 2.10(b). The modelling of the scene as fronto-parallel planes enables the bandwidth of the plenoptic function to be defined, therefore establishing the minimum sampling density for new views to be rendered without aliasing, and makes the occlusion ordering explicit [88, 89]. Previous work has also added variation in the depth of layers for each pixel in the plane, to represent real life object depths more closely [3].

Representation of a scene in the form of depth layers has been used for many different applications in multiview imaging. Disparity compensated and shape adaptive wavelet coding has used depth layers for wavelet compression of light fields [19], whilst object based compression [27] has been used to segment plenoptic video sequences into IBR objects, which can be rendered separately.



(a) (b)

Figure 2.10: It has been shown for example by [69] that a real scene (such as (a)) can be well represented by a small number of discrete depth layers (such as (b)). Image taken from [69].

Due to the large number of applications and the benefits of employing the layer based model, the decomposition of an EPI into planes at discrete depths has been well studied, but it remains a challenging problem that is not fully defined. When access to the ground truth light field is available, segmentation has been achieved to a highly accurate level by using manual methods [77, 27]. However, this method is not scalable as it requires a human to do the segmentation manually.

Other methods for segmenting the depths layers within the image have included using graph cuts [12, 55, 93], k-means clustering [89] and linear subspace results [45]. Commonly used are also the methods of energy minimisation with active contour methods [43] and the level set method [6, 7]. With active contour methods, the energy is minimised by an explicitly defined contour, or "snake", and forces are defined which act upon the contour to align it with plane boundaries. Contrasting with this, the level set method does not explicitly define the boundaries of the regions being segmented, with the boundary being defined implicitly as the zero level of a surface, with the area of the region being segmented defined as the area in which the surface is below zero, upon which forces from the image act. This has an advantage when used for segmenting a scene into planes, as the correct number of segments is not typically known in advance.

In a scene using a layer based model, a layer at a depth corresponds to a plenoptic hypervolume

within the plenoptic function [5], where all points in the layer will have the same trajectory within the plenoptic function, such as the layers in the EPI in Figure 2.11, which shows individual images on the left and a section through the complete EPI on the right. These are known as plenoptic manifolds in higher dimensional plenoptic representations. When analysing an EPI, the scene has been decomposed into layers in [22], which results in trajectories which are termed EPI tubes or EPI lines.

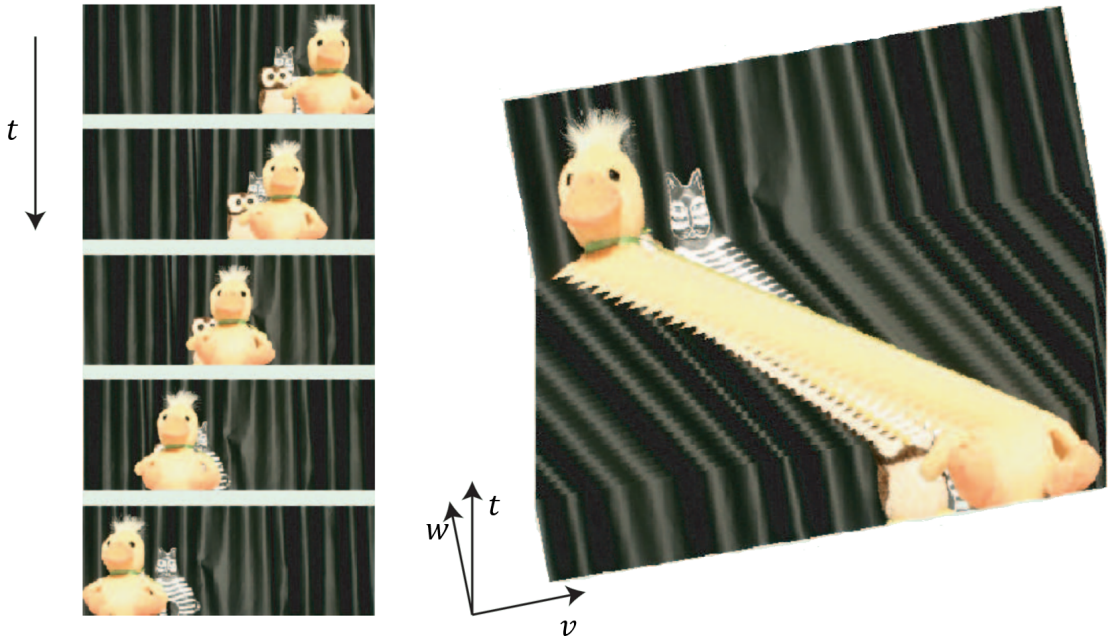


Figure 2.11: The EPI of a scene can be decomposed into volumes grouped by depth. The images of the scene for different camera positions are seen on the left. Layers within the scene are mapped to volumes within the EPI with a common trajectory, as shown on the right (image taken from [5]).

Multiple techniques have been developed to recover the plenoptic function of a scene from a sampled version. The recovery of the light field of a scene from a grid of images was investigated in [63, 64], whilst the recovery of the EPI from a set of images taken along a line has been studied in, for example, [30, 16, 24].

One widely used approach is to use local interpolation, where a point in the plenoptic function is estimated by the weighted interpolation of a number of nearby samples from available images [20, 33, 52, 15, 81]. When depth information in the scene is known, a novel view can be generated by interpolating each pixel along the EPI lines for the EPI case [11]. When the light

field is used, quadrilinear interpolation is employed [33, 52]. The disadvantage of this method is that the spacing of the available images must be very close, otherwise the new images rendered will include artifacts. The appearance of these artifacts may be mitigated by blending between the boundaries of layers, known as alpha matting [77], as the artifacts are most visible at the edges of the layers.

An issue with rendering new views of a scene from a grid or line of acquired images is the appearance of holes in the novel views. The holes are created when there are no available pixels in any of the input images to interpolate from. Gaps in rendered images degrade the appearance of the new images, so hole filling algorithms have been devised. The technique used in [65, 58] fills the hole by extrapolating pixel values from only regions located in the background by using a depth map to distinguish the background from foreground regions in the proximity of holes.

A further technique has been developed to recover the light field from a focal stack, in which the light field of a scene can be estimated from a set of images taken with different focuses [53]. This enables, for example, the refocusing of an image after it has been taken.

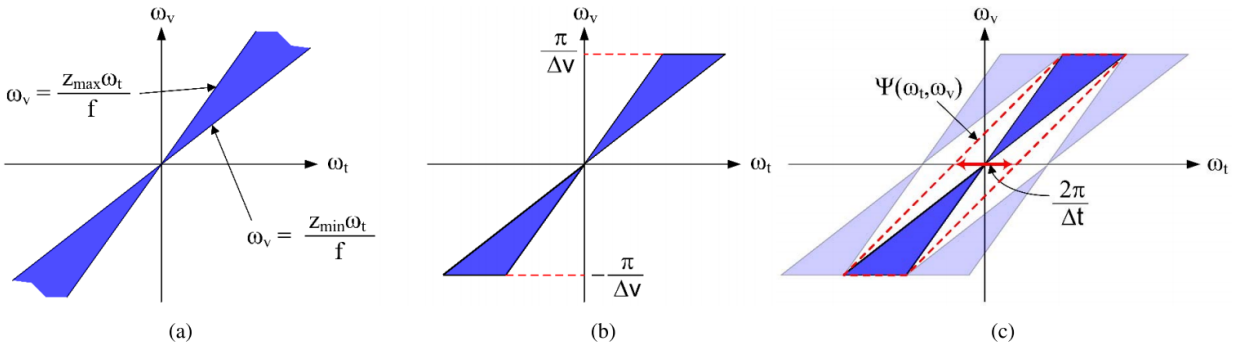


Figure 2.12: The spectrum of an EPI [30] is bounded by lines defined by the minimum and maximum depths present in the scene as shown in (a). (b) shows the low pass filtering effect due to the finite pixel resolution. (c) shows the minimum sampling density of the plenoptic function to avoid aliasing in the plenoptic spectrum (image taken from [30]).

When recovering the EPI from a set of images taken along a line, the spectral properties of the plenoptic function have been studied and the minimum sampling density of the EPI that is necessary to create images from novel viewpoints without aliasing [16, 30] has been established. Specifically, it has been shown that real images of complex scenes can be rendered using a layer-based model of the scene in which object depths are at discrete intervals [5, 29, 28, 69]. The

advantage of the layer based model is that it makes occlusion ordering explicit, and allows the spectral properties of the scene to be well defined. The advantage of the layer based method for analysis of the plenoptic spectrum, obtained by taking the 2-D Fourier transform of the EPI, is that the plenoptic spectrum is bounded by lines defined by the minimum and maximum depth layers, as visualised in Figure 2.12.

2.3 Depth from motion blur

Recovery of the EPI from a swiped image is related to the problem of recovering depth from motion blur. Without the multiple images available in the cases discussed in Section 2.2.3, the locations of objects within the swiped image must be estimated from the blur. Blur within an image can vary spatially, owing to variations in, for example, defocus blur [97] or motion blur. Accurate estimation of blur for each region of an image has been used to create a sharp image of the scene from a single perspective, known as blind deblurring or blind deconvolution [21].

Over the past two decades, a number of studies have addressed the issue of analysing motion blur, e.g. [17, 23, 21, 51]. Typically the motion blur is assumed to be due to the movement of objects within a scene, whilst the camera remains stationary. The most common objective of estimating depth in motion blurred images is to deblur an image with multiple depths [79, 34, 54, 68]. Objects that are thought likely to be motion blurred are rapidly moving points of interest such as a vehicle or person.

Estimating the depths from motion blur present in only a single image is a difficult task, as a sharp image of the scene is not available for comparison. Estimation of depths has been achieved by making assumptions about the statistical properties of the image such as by analysis of the Fourier transform of pixels on a window in [17], by comparing the horizontal and vertical gradient distributions in [51], and by intensity profile analysis in [54]. Other methods have used energy minimisation approaches, such as [21] which uses dictionary based filters for energy minimisation, or [79] which uses an auxiliary window-based depth estimation algorithm. A drawback in [79], is that it requires the user to select an appropriate initial area to estimate

depth in.

In [23] the depth estimation is based on the values of the alpha map. The technique relies on the pre-computation of a good alpha matte and has the assumption that a scene consists of a foreground object moving across a background. In contrast to many other methods, [34] does not assume spatially variant blur kernels, but camera motions, known as the motion density function. This enables a wide range of camera motions to create blur, such as horizontal, vertical and roll.

In [17], the level of blur present within the image in Figure 2.13(a) is estimated to create a depth map in Figure 2.13(b) from motion blur. This technique calculates the Fourier transform of windows around each pixel to find the location of zeroes, and compares this against the expected zeroes at different levels of blur to find the likelihood for each depth. The background in Figure 2.13(c) and the foreground in Figure 2.13(d) are deblurred using the depth estimate for each region to produce a composite sharp image. The advantages of the technique in this paper is that the technique works with only a horizontal window of pixels, but it can be inaccurate in regions with little texture.

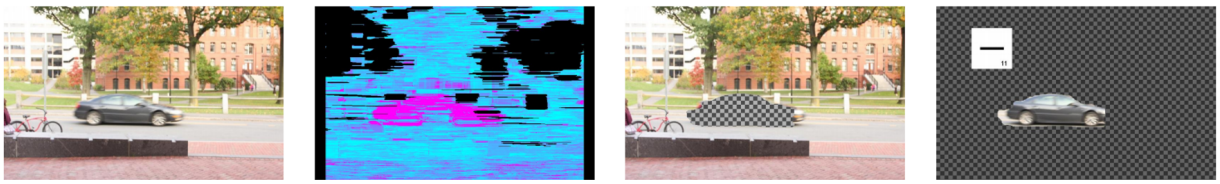


Figure 2.13: In [17], the image in (a) is analysed to create a depth map in (b) from motion blur. The background in (c) and the foreground in (d) are deblurred separately to produce a sharp image (image taken from [17]).

The paper in [51] attempts to use the statistics of the distribution of gradients within an image to estimate blur. Motion blur is modelled as 1D filtering with a box function along the direction of the movement of the subject in the scene during the exposure period. Each window within the image is artificially blurred in the other dimension to find the distribution of gradients that best matches the distribution of gradients in the direction of the motion blur. This technique

works well, but is dependent upon the assumption that similar blur in horizontal and vertical directions will produce similar gradient distributions, which may not be the case in image regions containing vertical or horizontal edges.

The approaches above have limitations due to the limited information available from a single image. As most of the approaches estimate the level of blur in a window around each pixel, the speed of the algorithms are slow and scales with image resolution. In addition to this, as only the image with motion blur is available, natural image priors are used which may not match correctly to areas of the underlying sharp image due to local features. A further drawback to many existing techniques is that deblurring images with different depths may lead to artifacts in the resultant deblurred image. This is because each depth is deblurred with a different point spread function, so the boundaries between depths do not accurately model two depths layers blurring into each other.

When a camera shutter is opened during the movement of an object within the scene, a low pass filtering effect is observed. Instead of an open shutter, a fluttering shutter [71] on a camera, with the exposures being coded of varying lengths has been employed to build a model of a scene. The fluttering shutter does not require any additional hardware, but the shutter fluttering must be carefully modulated to achieve good results.

2.4 Conclusion

In conclusion, this chapter introduces the concept of the plenoptic function in which a scene is represented by the light rays emanating from objects. It is shown that the plenoptic function is a useful framework for representing a scene from different viewpoints, and this has been employed to generate novel views of a scene with the technique of Image Based Rendering.

The full plenoptic function of seven dimensions has been introduced, and it has been shown that, by the use of appropriate assumptions, the dimensionality of the full plenoptic function may be reduced to four dimensions for the light field, or to three dimensions for the EPI.

The layer based model has been introduced, and it has been shown that this is an effective method of representing a complex scene. Furthermore, the advantages of using a layer based model to represent a scene when using plenoptic function have been discussed.

Finally, in the absence of multiple input images, an overview of the techniques of estimating depth from motion blur has been presented, along with the application of this estimate to deblurring.

Chapter 3

Ideal plane EPI recovery

3.1 Introduction

In Chapter 2, we noted that obtaining the plenoptic function of a scene directly is difficult in practice and requires either an array of cameras or a multi lens camera [64]. Both of these options significantly increase the cost, and so are rarely used in consumer electronics. The goal of the technique described in this chapter is to capture the EPI of a scene from a single image taken with a conventional consumer camera or a smartphone. The entire information of the EPI, between the camera swipe start and end locations, is encapsulated in a single swiped image that is captured by a camera moving uniformly along a straight line with the shutter open; an example swiped image being shown in Figure 3.1(a). Given a swiped image, such as the one shown in Figure 3.1(a), the goal is to recover the EPI of the scene. With a correctly recovered EPI, novel scene views may be created that match images of the scene from different viewpoints along the camera line, as illustrated by Figure 3.1(b).

This novel technique is different from the traditional deblurring methods that are employed in many consumer electronics, as the reconstructed pictures of the scene are created for an arbitrary camera location, within the range of camera movement. This contrasts with the non-exact sharp image from a single viewpoint that results from a mathematical model used for the well-studied process of deblurring [62, 75]. Traditionally, blur in an image is thought of as

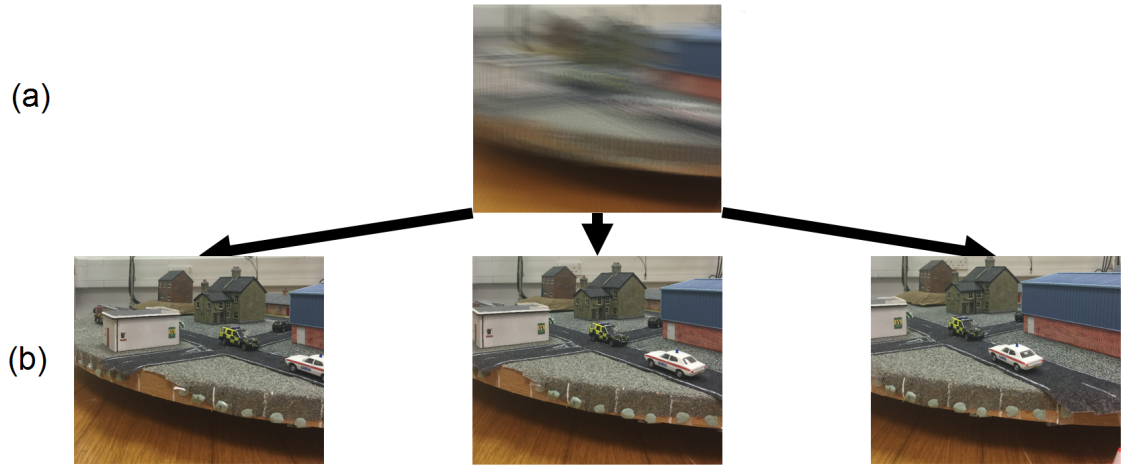


Figure 3.1: (a) shows a swiped image created by opening the camera shutter as the camera moves. (b) shows the images we aim to reconstruct by recovering the EPI of (a).

a problem to be removed from the image, however our technique instead makes use of blur as way of extracting information about the light rays in the scene.



Figure 3.2: (a) and (b) show two plane scenes with uniform surface textures. Occlusions are present in (a), but not in (b).

Obtaining the EPI of a real world scene is a complex task, so in this chapter we first aim to obtain the EPI from a swiped image of a simpler form of scene. The complexity of a real scene is due to objects at different depths in the scene occluding each other, and also due to the surface textures on the objects. Therefore, in this chapter, we focus on the problem of recovering the EPI for a scene in which each object is planar and has a uniform texture. Images of scenes with two uniform plane objects are shown in Figure 3.2, with occlusions present in Figure 3.2(a) and

absent in Figure 3.2(b).

We first solve the problem for the simplest case in which the scene contains only a single planar object; in this case no occlusions are possible. We then extend this technique to more complex scenes, in which there are multiple planar objects which may occlude each other. The problem setup with multiple planes is closer to that of a real world scene, and for that reason we use the assumptions from Section 2.2.3 to restrict the scene to fronto-parallel planes. The occlusions of planes by other planes within the scene increases the complexity of recovering the EPI. However, scene geometry can be inferred by the locations of occlusions seen in the EPI.

We then show that it is possible to recover the EPI of a single plane under more challenging conditions, where the plane has a complex texture. In this problem formulation, the problem of recovering the EPI from a single swiped image is mapped to that of reconstructing signals with a finite rate of innovation (FRI) [87, 82, 9, 25, 86], and we demonstrate that the exact recovery of the EPI of such a scene is possible. This EPI can then be used to generate unblurred images from an arbitrary camera location within the range of the camera swipe during the image acquisition.

The chapter is organised as follows: in Section 3.2 the problem formulation and the process of capturing the swiped image are described. In Section 3.3 the process of recovering the EPI from the swiped image of a single plane is described. In Section 3.4, we extend this technique to a scene with multiple planes. In Section 3.6, we recover the EPI of a scene with a single plane with a sinusoidal texture surface. In Section 3.5, we discuss the uniqueness of the solutions to recovering an EPI from a swiped image, and in Section 3.7, we show the application of the algorithms from this chapter on real images. Finally, we draw conclusions in Section 3.8.

3.2 Problem setup

For this chapter we consider a scene that comprises P fronto-parallel planes, with a uniform texture on each plane. Figure 3.3 illustrates the scene viewed from above for the case of $P = 2$. Without loss of generality, we consider only a 2D slice of the EPI by taking $y = w = 0$. In

the scene a point in space has coordinates (x, z) and the camera centre has coordinates $(t, 0)$. The horizontal limits of plane p are at x_{p1} and x_{p2}, and its depth is z_p. The number of planes is unknown in advance of the swiped image acquisition. In this scene, the focal length defines the image plane at $z = f$. The surface texture intensity of plane p is denoted as P_{I_p}, with the background to the scene being assumed to be black (a surface texture intensity of 0).

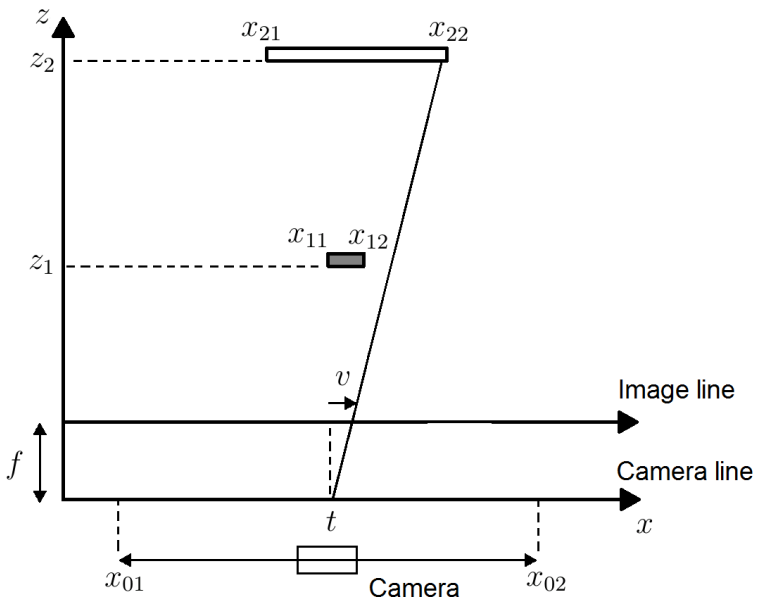


Figure 3.3: An example of a scene with two planes.

We suppose that the camera is swiped from x_{01} to x_{02} at a constant velocity along the camera line, which, without loss of generality we take to be $y = z = 0$. It is convenient to define an additional plane, the zeroth plane, at a depth of $z_0 = 0$. This plane can be viewed as a masking plane; the setup is equivalent to the camera swiping from minus infinity to plus infinity with the shutter open, but only being able to see the scene through the zeroth plane, between x_{01} to x_{02}. A swiped image of a scene with this setup is illustrated in Figure 3.4(a).

A slice, $I_I(v)$, of the resultant swiped image is shown in Figure 3.4(a) and the corresponding EPI we aim to retrieve in Figure 3.4(b), plotted against the camera position, t. The slice, $I_I(v)$, is obtained by integrating the EPI horizontally over the range $x_{01} < t < x_{02}$:

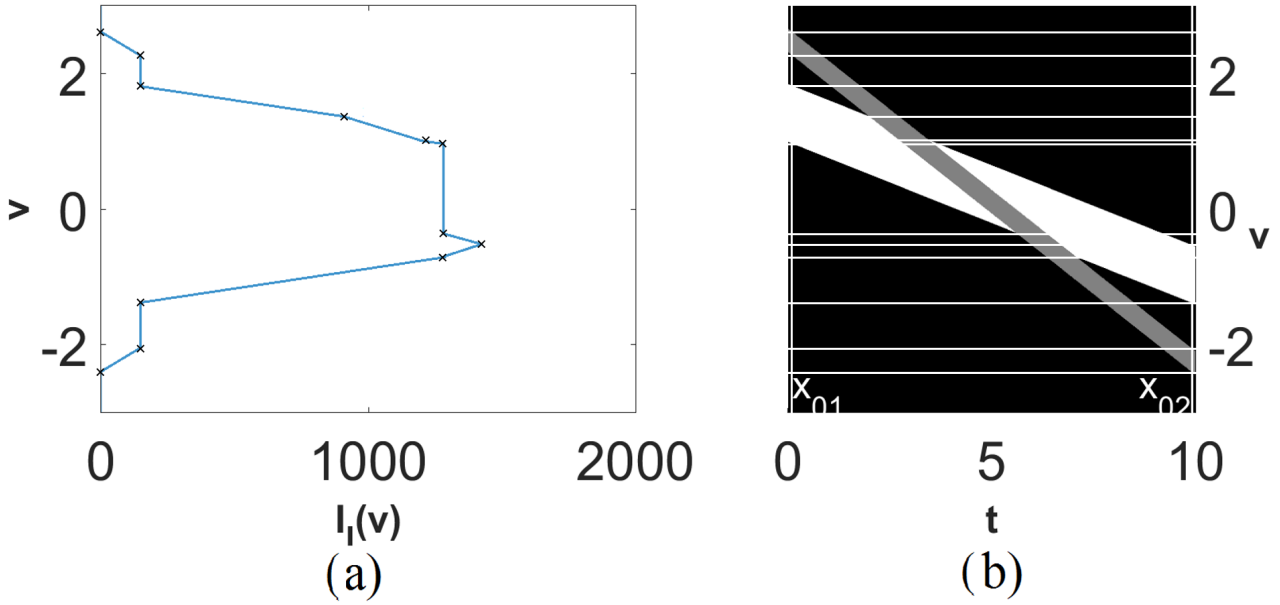


Figure 3.4: A row of the swiped image, (a), created by integrating the EPI shown in (b) for a two plane image with constant plane texture surfaces. Each switchpoint is indicated by a white line in the EPI and by a cross in the row of the swiped image.

$$I_I(v) = \int_{x_{01}}^{x_{02}} I(v,t)dt \tag{3.1}$$

where $I(v,t)$ is a horizontal slice of the EPI of the scene. Thus Figure 3.4(a) is obtained by integrating Figure 3.4(b) horizontally.

A "switchpoint" in the swiped image is created every time one plane occludes or disoccludes another in the EPI, with the total number of switchpoints in the swiped image being V_s. This can be seen in Figure 3.4(b) where the darker plane occludes and disoccludes the lighter plane as the camera is swiped. The switchpoint locations in v are indicated in the EPI in Figure 3.4(b) by the horizontal white lines and in Figure 3.4(a) by crosses. Switchpoints in the idealised case result in discontinuities in the gradient of $I_I(v)$. This occurs when plane edges (x_{pi}, z_p) and (x_{qj}, z_q) are aligned at the camera position $t = t_{pi,qj}$, and the aligned points are at $v = v_{pi,qj}$ in the swiped image, with the labelling illustrated by Figure 3.5. We therefore conclude that

identifying the switchpoints of a scene will allow us to recover a valid plane geometry of the scene, and hence a valid EPI. The case of multiple scenes resulting in the same swiped image is discussed in Section 3.5.

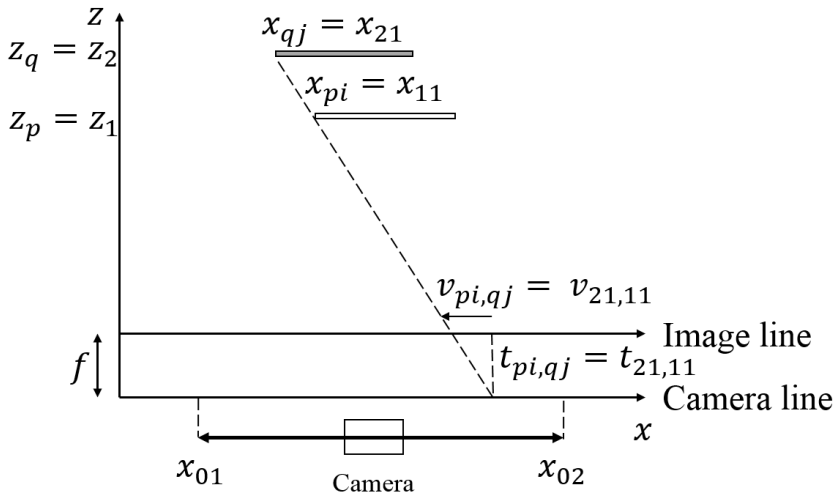


Figure 3.5: A switchpoint is labelled by the plane edge combination which created it.

3.3 Single plane EPI recovery

3.3.1 The single plane swiped image

In this section, we aim to recover the EPI of a single plane swiped image, restricting the scene to $P = 1$, to avoid the complexity that is introduced by planes occluding each other.

The resulting integrated image $I_I(v)$ is made of 5 sections that are related to the limits of integration. The four segment boundaries, or switchpoints, in the swiped image are the values of v where the zeroth plane, at x_{01} and x_{02}, occludes or disoccludes the scene plane in the EPI. By using similar triangles, shown in Figure 3.6, the switchpoints can be related to the scene geometry using the equations:

$$v_{11,02} = f(x_{11} - x_{02}){z_1}^{-1} \qquad (3.2) \qquad\qquad v_{11,01} = f(x_{11} - x_{01}){z_1}^{-1} \qquad (3.3)$$

$$v_{12,02} = f(x_{12} - x_{02}){z_1}^{-1} \qquad (3.4) \qquad\qquad v_{12,01} = f(x_{12} - x_{01}){z_1}^{-1} \qquad (3.5)$$

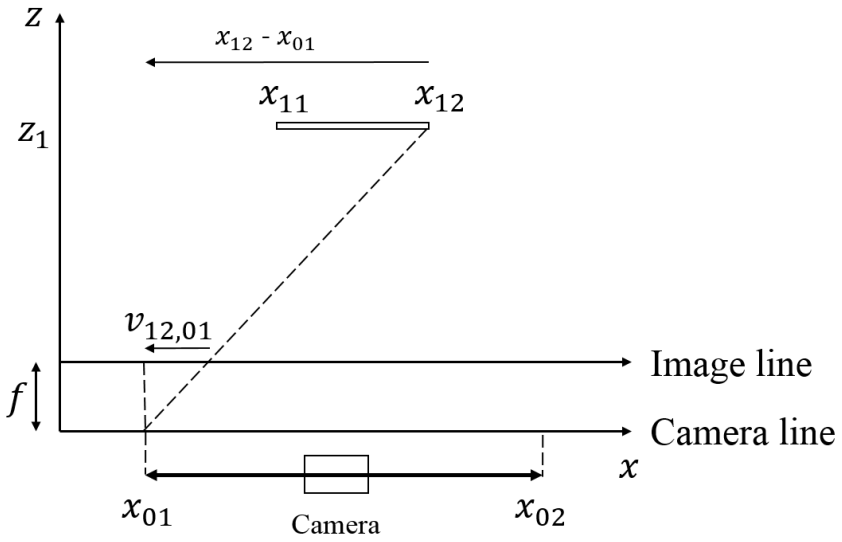


Figure 3.6: (3.5) is derived from similar triangles in the scene geometry. The same process is used to derive (3.2) - (3.4).

3.3.2 Single plane algorithm overview

We leverage our knowledge of the single plane swiped image from Section 3.3 to create an algorithm that can retrieve the EPI from a swiped image. The algorithm in the following consists of the following steps. First we detect the switchpoint locations, and from that the plane geometry (step (a) in Figure 3.7). We then calculate the plane surface texture (step (b) in Figure 3.7). Finally, new images of the scene may be rendered from any camera location within the original swipe range (step (c) in Figure 3.7) using the information gained from the previous steps.

3.3.3 Determining plane geometry

Combining (3.2) with (3.3) allows x_{11} and z_1 to be found, which is the location for the leftmost limit of the plane. Combining (3.4) with (3.5) allows x_{12} and z_1 to be found, which is the location for the rightmost limit of the plane. Figure 3.8 shows the plane limits as the solutions to these equations. We assume that the camera swipe is large enough such that $v_{11,01} < v_{12,02}$.

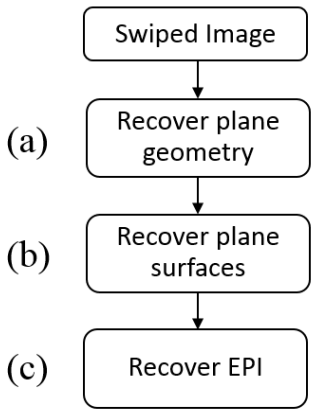


Figure 3.7: A flowchart of the steps in the algorithm to recover an EPI from a single plane swiped image with uniform surface texture. In step (a), we determine scene geometry. In step (b), we determine the plane surface texture. Finally, in step (c), we reconstruct the EPI of the scene.

3.3.4 Determining plane surface texture

The value of the swiped image between $v_{11,01}$ and $v_{12,02}$ is used to calculate the plane surface texture, with the equation being:

$$P_{I_1} = \frac{I_I(v_{11,01})f}{(x_{12} - x_{11})z_1} \tag{3.6}$$

where P_{I_1} is the intensity of the plane surface texture for $p = 1$ and $I_I(v_{11,01})$ is the value of the swiped image at switchpoint $v_{11,01}$. Given knowledge of the plane geometry and the plane surface texture, we may now reconstruct the EPI of the scene.

3.4 EPI recovery from a multiple plane swiped image

3.4.1 The multiple plane swiped image

We now consider the swiped image of a scene in which there is more than one plane. With multiple planes present in the scene, the number of switchpoints in the EPI increases. The

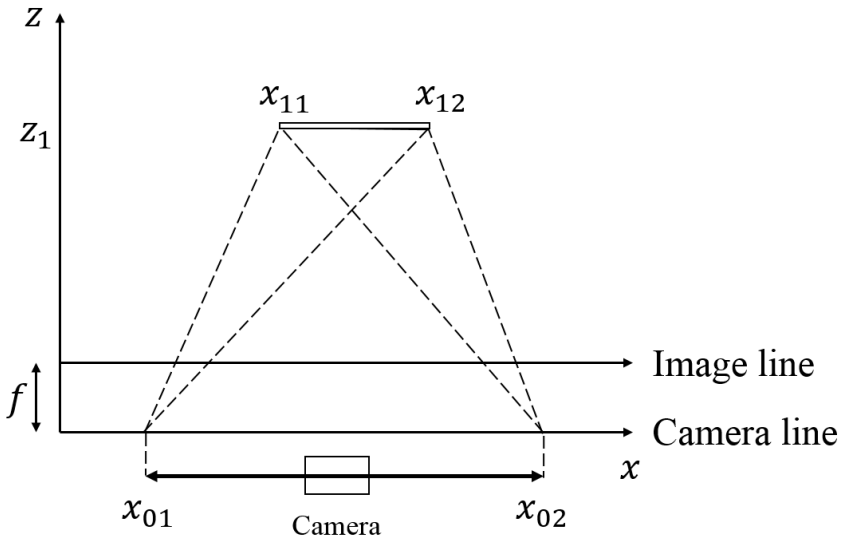


Figure 3.8: The dashed lines indicate the implied plane limits from the location of the switchpoints in the EPI.

technique employed in Section 3.3, is therefore not practical as it requires prior knowledge of the switchpoint ordering.

From similar triangles in Figure 3.3 we can write:

$$\frac{v_{pi,qj}}{f} = \frac{x_{pi} - x_{qj}}{z_p - z_q} \tag{3.7}$$

for each switchpoint. We can then rearrange this equation into the linear form:

$$x_{pi} - \bar{v}_{pi,qj} z_p - x_{qj} + \bar{v}_{pi,qj} z_q = 0 \tag{3.8}$$

where $\bar{v} = \frac{v}{f}$ is the normalised version of v. Assuming, without loss of generality, that $p > q$, for $0 \leq q < p \leq P$ and $1 \leq i, j \leq 2$, we have a total of $2P^2 + 2P$ equations in $3P$ unknowns. Of these, the $4P$ equations with $q = 0$, have the simpler form:

$$x_{pi} - \bar{v}_{pi,0j} z_p - x_{0j} = 0 \tag{3.9}$$

which can be rewritten as:

$$x_{pi} - \bar{v}_{pi,0j} z_p = x_{0j}. \tag{3.10}$$

Thus, for example, when $P = 2$, we get a total of 12 equations in 6 unknowns where $piqj \in (1101, 1201, 1102, 1202, 2101, 2201, 2102, 2202, 2111, 2211, 2112, 2212)$. These equations can be arranged into matrix form as follows:

$$\begin{pmatrix}
1 & 0 & -\bar{v}_{11,01} & 0 & 0 & 0 \\
0 & 1 & -\bar{v}_{12,01} & 0 & 0 & 0 \\
1 & 0 & -\bar{v}_{11,02} & 0 & 0 & 0 \\
0 & 1 & -\bar{v}_{12,02} & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & -\bar{v}_{21,01} \\
0 & 0 & 0 & 0 & 1 & -\bar{v}_{22,01} \\
0 & 0 & 0 & 1 & 0 & -\bar{v}_{21,02} \\
0 & 0 & 0 & 0 & 1 & -\bar{v}_{22,02} \\
-1 & 0 & \bar{v}_{21,11} & 1 & 0 & -\bar{v}_{21,11} \\
-1 & 0 & \bar{v}_{22,11} & 0 & 1 & -\bar{v}_{22,11} \\
0 & -1 & \bar{v}_{21,12} & 1 & 0 & -\bar{v}_{21,12} \\
0 & -1 & \bar{v}_{22,12} & 0 & 1 & -\bar{v}_{22,12}
\end{pmatrix} \cdot \begin{pmatrix} x_{11} \\ x_{12} \\ z_1 \\ x_{21} \\ x_{22} \\ z_2 \end{pmatrix} = \begin{pmatrix} x_{01} \\ x_{01} \\ x_{02} \\ x_{02} \\ x_{01} \\ x_{01} \\ x_{02} \\ x_{02} \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Due to the limited swipe length, not all switchpoints are visible in every swiped image. Each switchpoint, $\bar{v}_{pi,qj}$ occurs in only one equation, so if the switchpoint is not present, the corresponding equation may be omitted. Provided that at least $3P$ switchpoints are present, the $3P$ unknowns that define the scene geometry can be determined. The difficulty in this prob-

lem is that switchpoints from the swiped image need to be labelled correctly, as the order of switchpoints in the swiped image is undetermined. This is similar to the problem of unlabelled sensing, where a number of samples of a signal are taken at unknown times. If the times that the samples are taken at are correctly labelled, then the signal can be correctly characterised [85]. Similarly in this situation, once the switchpoints have been labelled correctly, the problem is reduced to that of solving a set of linear equations. When the solution is under determined the physical significance of this is that a plane edge is not able to be determined as it is not seen during the entire swipe, as it either goes off the edge of the image, or is hidden behind another plane during the entire camera swipe. This possibility is discussed in Section 3.5, where there are multiple scenes which could lead to the same swiped image. In this case, the plane is assumed to be the scene with the least width, to result in a single answer that is created from the simplest type of scene.

The swiped image value at switchpoint n can be calculated by the sum of the contributions of the planes at the switchpoint, with the equation being:

$$I_I(v_n) = \sum_{p=1}^{P} P_{w_p}(v_n) P_{I_p} \tag{3.11}$$

where $P_{w_p}(v_n)$ is the size of plane in the t axis in the EPI at switchpoint n between t_1 and t_2, minus the occlusions from closer planes, P_{I_p} is the intensity of plane p and $I_I(v_n)$ is the value of the swiped image at switchpoint n. There will be an Equation (3.11) for each switchpoint in the swiped image, i.e. for $1 <= n <= 2P^2 + 2P$.

Given $I(v)$ and x_{0i}, the goal is to determine P, x_{pi}, z_p and P_{I_p} for $1 \leq p \leq P$ and $1 \leq i \leq 2$. From this information, it is possible to reconstruct the EPI of the scene.

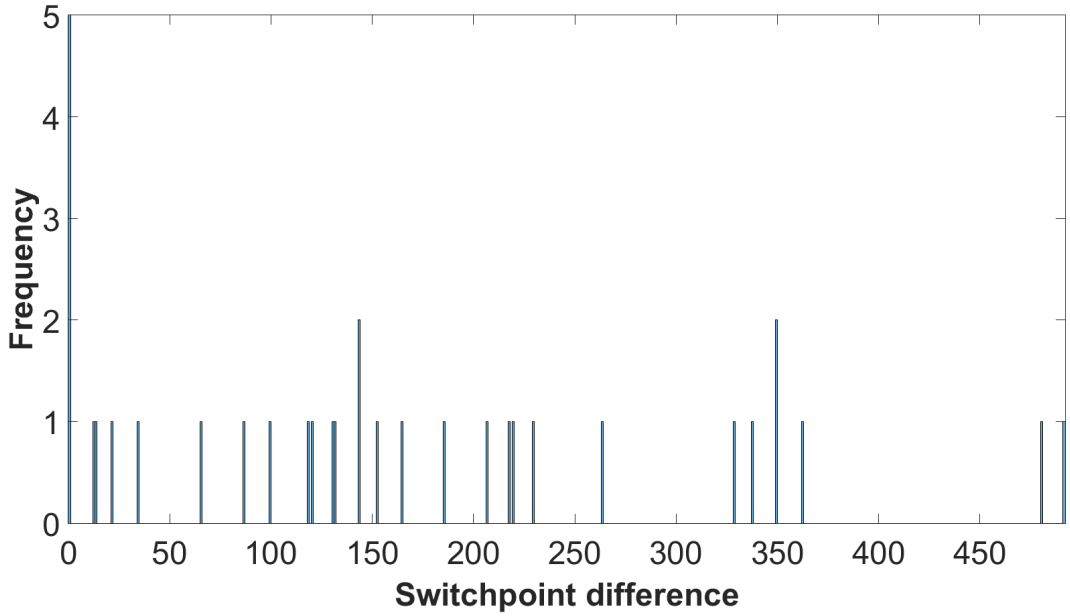


Figure 3.9: A histogram of distances between switchpoints. As the planes in the scene are fronto parallel, the distance between switchpoints are repeated. Repeated switchpoints are identified and used to hypothesise planes in the swiped image. In the scene represented by this histogram, there are possible planes with a switchpoint difference of 144 and 350 pixels.

3.4.2 Multiple plane algorithm overview

We use the knowledge of the underlying structure of the scene to label switchpoints, with only labels that lead to valid scenes being considered to reduce search space. Within the swiped image, the switchpoints created by the alignment between the scene planes and the zeroth plane can be used to hypothesise planes. When a plane is seen unoccluded at both extremes of camera range, the distance between the switchpoints caused by the camera swipe will be equal, so switchpoint pairs with equal distances may be used to hypothesise planes, as seen in Figure 3.9. The hypothesised planes are confirmed by checking if the switchpoints created by the occlusion or disocclusion of one hypothesised plane by another are present within the swiped image. Once these planes have been detected, remaining switchpoints are labelled in a combinatorial approach. We can recover the EPI for a scene, given the condition that each plane is seen unoccluded at one end of the camera swipe range. Once the switchpoints have been labelled and verified, the swiped image can be reconstructed to find the values of the planes and check the algorithm result correctness.

The number of planes in the scene is unknown initially, and is guessed by the number of switchpoints present in the image. We first assume that given the total switchpoints detected that the simplest scene, the one with the least number of possible planes, is the correct one. If after all possible combinations are exhausted for scenes with that number of planes, we increase the number of assumed planes and repeat the algorithm.

The algorithm can be described as follows:

1. Assume the number of planes, P, within the scene is the minimum possible given the number of switchpoints, V_s, to satisfy: $2P^2 + 2P > V_s$.

2. Create a histogram of distances between switchpoint locations to find repeated differences.

3. For each repeated switchpoint difference, hypothesise that the switchpoints are $v_{p1,01}$, $v_{p2,01}$, $v_{p1,02}$, $v_{p2,02}$. These define the scene geometry (x_{p1}, x_{p2} and z_p) of a hypothetical plane p.

4. Create a list of hypothesised plane pair combinations, where one plane is plane p and the other plane is plane q.

5. For each pair combination of hypothesised planes, find the v location of the switchpoints that should occur by the planes occluding each other ($v_{pi,qj}$ for $1 \leq i, j \leq 2$), which are present within the swiped image if $t_{pi,qj}$ is within the range $x_{01} < t < x_{02}$. If these switchpoints exist then evidence for these planes exist.

6. If all switchpoint have been accounted for, recreate swiped image $I_I(v)$ and find plane surface textures by solving (3.11).

7. If the number of switchpoints from hypothesised planes with evidence for existence is less than V_s, hypothesise additional planes using sets of three remaining switchpoints.Two of these switchpoints are assumed to be either $v_{p1,01}$ and $v_{p2,01}$ or $v_{p1,02}$ and $v_{p2,02}$. If the switchpoints are $v_{p1,01}$ and $v_{p2,01}$ then the additional switchpoint is assumed to be $v_{p2,q1}$. If the switchpoints are $v_{p1,02}$ and $v_{p2,02}$ then the additional switchpoint is assumed to be $v_{p1,q2}$.

8. For each pair combination of hypothesised planes, find the v location of the switchpoints that should occur by the planes occluding each other ($v_{pi,qj}$ for $1 \leq i, j \leq 2$), which are present within the swiped image if $t_{pi,qj}$ is within the range $x_{01} < t < x_{02}$. If these switchpoints exist then evidence for these planes exist.

9. Recreate swiped image $I_I(v)$ and find plane surface textures P_{I_p} by solving (3.11).

10. If swiped image cannot be recreated, increase the number of hypothesised total planes in the scene.

3.4.3 Multiple plane algorithm implementation

In real images of scenes, it cannot be assumed that the switchpoint locations lie at the exact values of v that would be expected from the scene geometry. This is because of small errors in camera alignment, and nearest pixel rounding errors for the switchpoint locations. To enable pairs of differences to be detected correctly, the position of the switchpoints is modelled as having a normal distribution, with mean $\mu = \bar{v}_{qi,pj}$ and the variance σ^2 chosen based on the detected level of switchpoint uncertainty. The switchpoint difference distribution can be written as:

$$\mu_{1-2} = \mu_1 - \mu_2$$
$$\sigma^2_{1-2} = \sigma^2_1 + \sigma^2_2$$

where μ is the mean value of a switchpoint and σ^2 is the variance of a switchpoint. To find repeated differences, the Hellinger distance between the switchpoint distribution was calculated:

$$H^2 = 1 - \sqrt{\frac{2\sigma_1\sigma_2}{\sigma_1^2 + \sigma_2^2}} e^{-\frac{1}{4}\frac{(\mu_1-\mu_2)^2}{\sigma_1^2+\sigma_2^2}}$$

If the Hellinger distance for two distances is below a threshold, the differences are considered to

be matched for the purposes of this algorithm. The threshold was chosen based on experimental data, with the chosen value being high enough to exclude most false positives, but low enough to not exclude correct repeated differences.

The technique outlined in this section can determine the plenoptic function given a number of assumptions. It is assumed that the swiped image acquired is unique to a scene geometry, and that each plane is seen unoccluded at either extreme of the camera range of movement.

3.5 Uniqueness of the solution

In the algorithms proposed in this chapter, we implicitly assume that a swiped image maps to only one possible EPI. This assumption is not true for all possible scenes, as there are swiped images which can be generated from more than one valid EPI.

However, in practice, this situation is highly unlikely. For a scene to lead to the same EPI as another scene, such as in Figure 3.10, multiple switchpoints must share the same location, requiring exact alignment of planes. Furthermore the product of plane surface intensity and depth for two different planes must be equal. Given the total possible number of scene configurations, the configurations leading to a swiped image with multiple possible EPIs are an extremely small number. Using the algorithm in Section 3.4, if there is more than one EPI validated as correct after the combinatorial search, the EPI with the planes closest to the camera is chosen as the solution. We select the simplest form of scene as the chosen one. When the plane chosen is the closest possible, given the swiped image, this results in the plane being the smallest, and therefore least complex. The solution of the smallest plane is again chosen in an underdetermined system. When the edge of a plane is unseen, we assume it to be the edge that results in the plane with the least width. By choosing this solution, we get a specific EPI from each swiped image.

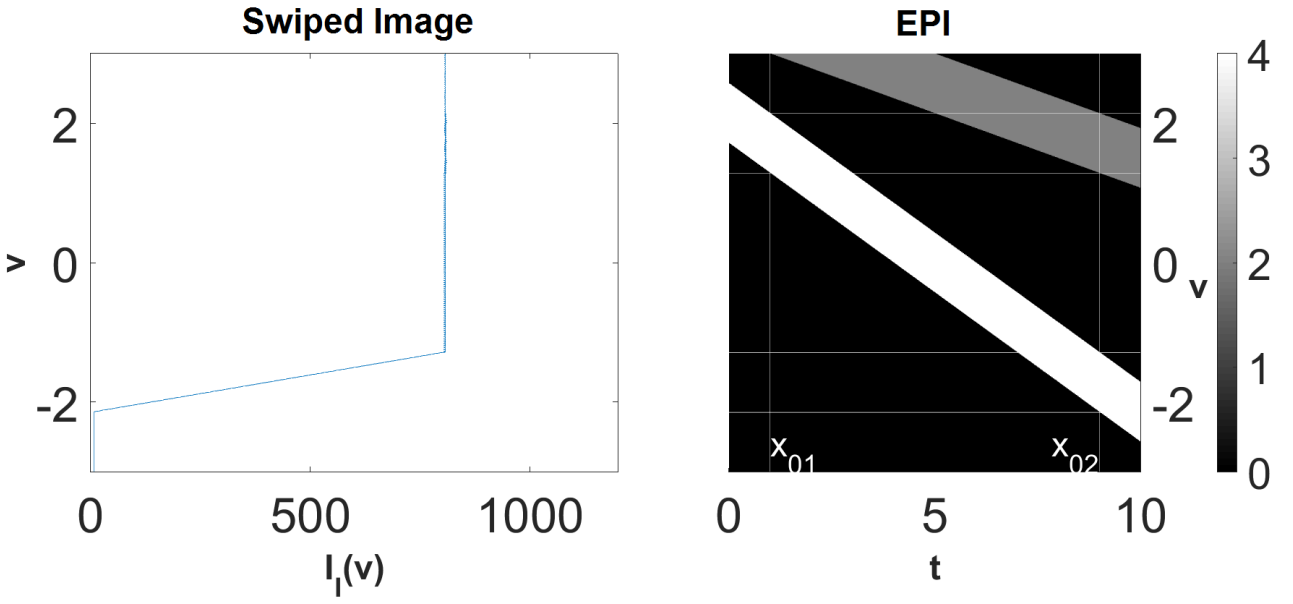


Figure 3.10: The swiped image appears to be generated from a single plane scene, but the EPI contains two planes with shared switchpoints. The further plane from the camera has twice the distance from the camera as the closer plane, but half the surface intensity.

3.6 Sinusoidal texture EPI recovery

3.6.1 The sinusoidal texture swiped image

In Sections 3.3 and 3.4, we assumed that the scene consists of planes with a uniform texture. In this section, we show that it is also possible to recover the EPI of a single fronto-parallel plane with a more complex texture, namely a sum of sinusoids. The scene in this problem setup is illustrated by Figure 3.11. In this case, we assume that we do not have access to the underlying swiped image, only a version that has been sampled and distorted by an acquisition kernel. This is to model acquiring the image using a limited resolution camera with a distorting lens.

The reason that a sum of sinusoids is used as a plane texture is because this is a more complex texture than a uniform plane texture, whilst still resulting in a highly structured swiped image. As all signals can be expressed as a sum of sinusoids, there are many existing signal processing techniques to analyse these signals. We leverage existing techniques, such as Finite Rate of Innovation (FRI) sampling, to find the scene geometry and plane surface texture.

In the absence of occlusions, using (3.7) every light ray that arrives at pixel location v satisfies

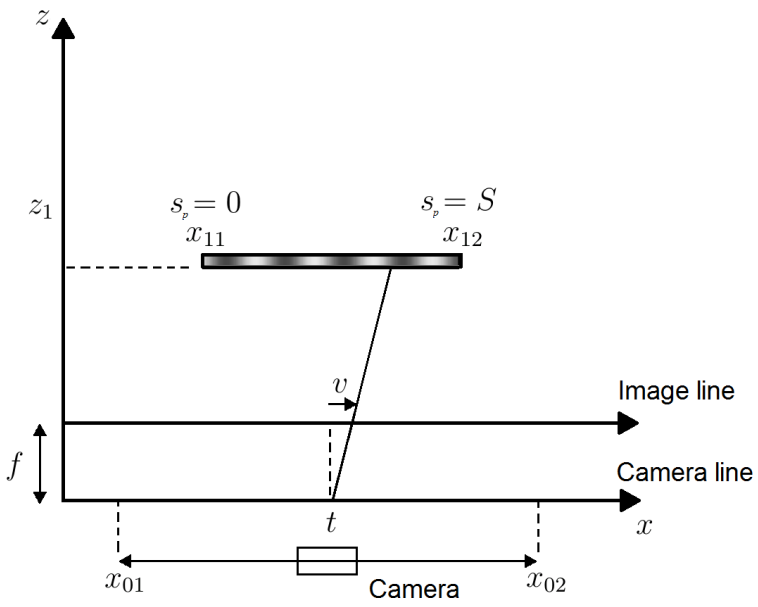


Figure 3.11: The swiped image is created from a scene containing a single fronto-parallel plane with a sinusoidal texture surface.

the equation

$$fx - ft - vz = 0 \tag{3.12}$$

where f is the focal length of the camera. When the plane texture is a sum of sinusoids, this ray originates from a point on the scene surface with curvilinear coordinate s_p, given by:

$$s_p(v,t) = t + q_s$$

where

$$q_s = x_{11}\frac{vz_1}{f}$$

When no part of a plane is occluded, the intensity of rays arising from a point on the plane surface can be expressed as a sum of complex sinusoids:

$$I(s_p) = A + \sum_{n=1}^{N_s} B e^{j\omega_n s_p} \tag{3.13}$$

which results in the image intensity:

$$I(v,t) = A + \sum_{n=1}^{N_s} B_n e^{j\omega_n s_p(v,t)} = A + \sum_{n=1}^{N_s} B_n e^{j\omega_n (t+q_s)} \tag{3.14}$$

where ω_n is a frequency of a complex sinusoid on the texture surface, and N_s is the number of sinusoids. Equation 3.14 is valid for:

$$f\frac{(x_1 - t)}{z_1} < v < f\frac{(x_2 - t)}{z_2} \tag{3.15}$$

the range of v in which the plane surface is seen.

The resultant swiped image may be obtained by integrating (3.14) over the range $x_{01} < t < x_{02}$, shown in Figure 3.12, which results in:

$$I_I(v) = \int_a^b A + \sum_{n=1}^{N_s} B_n e^{j\omega_n (rt+q)} dt = \Big[At + \sum_{n=1}^{N_s} \frac{B_n}{j\omega_n r} e^{j\omega_n (rt+q)} \Big]_a^b \tag{3.16}$$

where the integration limits, a and b, depend both on the value of v and on the scene geometry, and are given by:

$$a = \min(\max((x_{11} - z_1)\frac{v}{f}, x_{01}), x_{02})$$
$$b = \min(\max((x_{12} - z_1)\frac{v}{f}, x_{01}), x_{02})$$

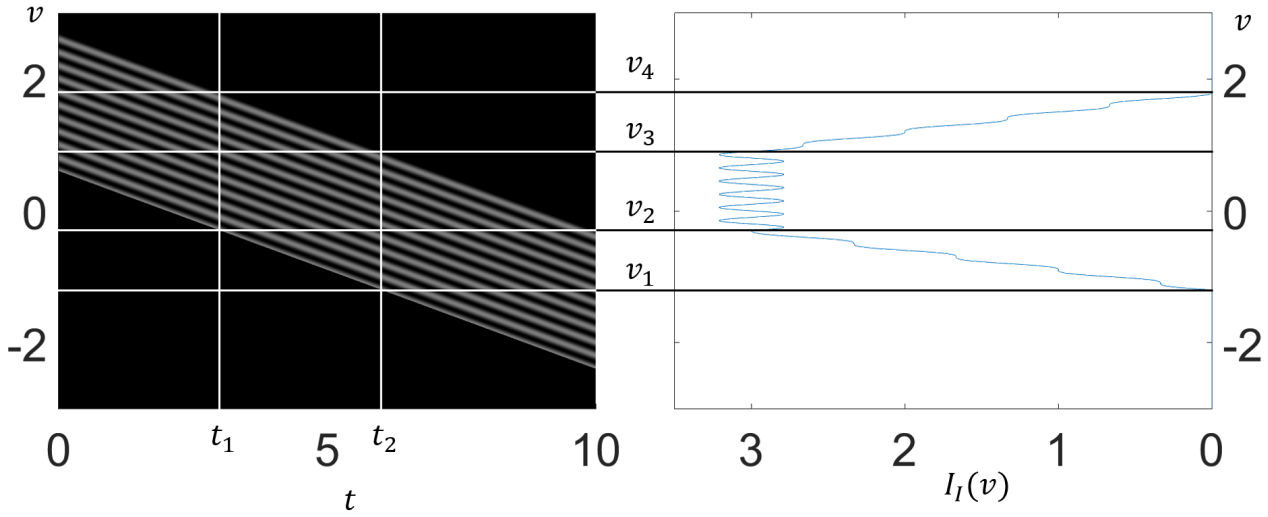


Figure 3.12: The swiped image (right) formed from the EPI (left) of a fronto-parallel plane with a sinusoidal texture surface.

In practice, when acquiring the image with a digital camera, the continuous function $I_I(v)$ cannot be measured directly and only the pixels $p_I[k]$ of the swiped image are seen. Information in $I_I(v)$ is distorted due to the blur introduced by the camera lens (modelled by its point spread function) and the limited density of imaging sensors. This process is similar to the way that acquisition is modelled in traditional sampling theory, and the pixels of one row of the swiped image are given by the inner product:

$$p_I[k] = \int_{-\infty}^{\infty} \varphi_I(\frac{v}{T_I} - k) I_I(v) dv \triangleq \left\langle \varphi_I(\frac{v}{T_I} - k), I_I(v) \right\rangle$$

where p_I are the pixels, k is the pixel index, T_I is the spacing between pixels and $\varphi_I(v)$ is the point spread function of the camera or sampling kernel.

Given the pixels $p_I[k]$ of the swiped image acquired, the goal is to retrieve the information

contained within the EPI. Recovering the frequencies of the sinusoids in the texture surface, the amplitude and phase of the sinusoids in the texture surface, and the locations of the switchpoints allows the EPI to be exactly determined. With knowledge of the EPI, new views of the scene can be created for arbitrary camera locations.

Using the layer based model from Section 2.2.3, we consider the case where the plane in the scene is fronto-parallel. Under this assumption $I_I(v)$ can be written as follows:

$$I_I(v) = \sum_{d=1}^{3} \sum_{n=1}^{N_s} B_{d,n} e^{j\omega_{d,n} v + \psi_{d,n}} \xi_d(v) + \sum_{\rho=0}^{1} A_{d,\rho} t^\rho \xi_d(v) \tag{3.17}$$

where d indexes the segments of $I_I(v)$ between consecutive switchpoints, N_s is the number of complex exponentials in the texture, $\xi_d(t)$ is the Heaviside step function, ρ is the order of the linear component and $\psi_{d,n}$ is the phase of the sinusoid.

$I_I(v)$ is a piecewise signal in which each segment is a sum of sinusoids; it can therefore be recovered from the pixels $p_I[k]$ using FRI theory [87, 82, 9, 25, 86, 8]. It is assumed here that the point spread function $\varphi_I(v)$ is an exponential reproducing kernel, as in [25], but less restrictive choices are also available [86]. In what follows, we show that it is possible to retrieve the parameters of $I_I(v)$, that is the frequency of the texture and the switch points by adapting to this case the method in [8].

3.6.2 Sinusoidal texture algorithm overview

We leverage our knowledge of the single plane swiped image from Section 3.6 to create an algorithm that can retrieve the EPI from a swiped image. The algorithm in the following consists of the following steps. First we detect the sections in the image which consist of only a sum of sinusoids, to enable the detection of plane texture frequencies (step (a) in Figure 3.13). The plane texture frequency is then determined from the detected sections (step (b) in Figure 3.13). The switchpoint locations are then estimated to find the locations of the plane extremities (step

(c) in Figure 3.13). Finally, new images of the scene may be rendered from any camera location within the original swipe range (step (d) in Figure 3.13) using the information gained from the previous steps.

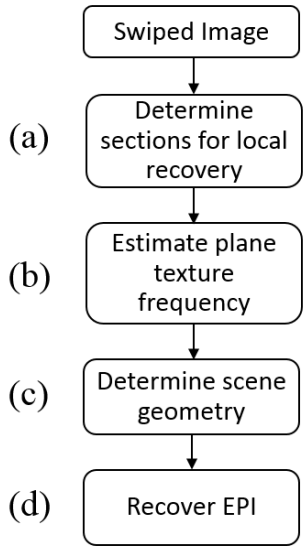


Figure 3.13: A flowchart of the steps in the algorithm to recover an EPI from a single plane swiped image. In step (a), we determine valid windows to detect plane texture frequency. In step (b), we estimate plane texture frequency. In step (c), we determine scene geometry. Finally, in step (d), we reconstruct the EPI of the scene.

3.6.3 Determining the sections for local recovery

It has been shown in [8] that if a sampled signal $p_I[k]$ is a sum of weighted exponentials $p_I[k] = \sum_{n=1}^{N_s} a_n u_n^k$ then a filter with the z transform $H_{\mathbf{u}}(z) = \prod_{n=1}^{N_s}(1 - u_n z^{-1})$ will annihilate the signal $p_I[k]$. This can be expressed in matrix form as:

$$\mathbf{P_I h_u} = \begin{pmatrix} \vdots & \vdots & \dots & \vdots \\ p_I[0] & p_I[-1] & \dots & p_I[-N_s] \\ p_I[1] & p_I[0] & \dots & p_I[-N_s+1] \\ \vdots & \vdots & \ddots & \vdots \\ p_I[N_s] & p_I[N_s-1] & \dots & p_I[0] \\ \vdots & \vdots & \dots & \vdots \end{pmatrix} \begin{pmatrix} h_{\mathbf{u}}[0] \\ h_{\mathbf{u}}[1] \\ \vdots \\ h_{\mathbf{u}}[N_s] \end{pmatrix} = 0$$

where $p_I[k]$ is the finite difference of the pixel values, $\mathbf{P_I}$ is a $(N_s+1)\times(N_s+1)$ Toeplitz matrix created with $2N_s+1$ samples of $p_I[k]$, and $\mathbf{h_u}$ defines the impulse response of $H_{\mathbf{u}}(z)$.

If the $2N_s+1$ samples of $p_I[k]$ used to form $\mathbf{P_I}$ include contributions from only N_s sinusoids, then $\mathbf{P_I}$ will be rank deficient. A sliding window over $p_I[k]$ can be used to find window positions which contain only contributions from pure sinusoids, undistorted by switchpoints. Using the finite difference of the pixels to eliminate the constant term, denoted by A in (3.17), we can analyse the signal as a sinusoids centered around zero. Windows in the segment between $\bar{v}_{11,01}$ and $\bar{v}_{12,02}$ can be used to detect the frequency of the sinusoidal component. The frequency found, by using these windows in Section 3.6.4, represents the frequency of the sinusoidal texture surface on the plane.

3.6.4 Determining the texture frequency

The frequency of the sine waves in each rank deficient window can be found by constructing the annihilating filter $H_{\mathbf{u}}(z) = \prod_{n=1}^{N_s}(1-u_n z^{-1})$, and finding the roots of the filter. The roots will be pairs of the form $z = e^{j\omega_n}$ and $z = e^{-j\omega_n}$, where ω_n is the nth surface texture frequency. The phase and amplitude of the sinusoids in the rank deficient window can then be found by constructing a Vandermonde system [87]. For example, for the case $N_s = 1$, the Vandermonde system can be written as:

$$\frac{1}{2}\begin{pmatrix} e^{j\omega_1 k} & e^{-j\omega_1 k} \\ e^{j\omega_1(k+1)} & e^{-j\omega_1(k+1)} \end{pmatrix}\begin{pmatrix} B\varphi(-\omega_1)e^{j\psi_1} \\ B\varphi(\omega_1)e^{-j\psi_1} \end{pmatrix} = \begin{pmatrix} p[k] \\ p[k+1] \end{pmatrix}.$$

3.6.5 Determining the switchpoint locations

Once the annihilating filter ($H_{\mathbf{u}}(z)$) has been found for the $\mathbf{P_I}$ matrix, it can be applied to the entire sequence of pixels to form a new signal [8]:

$$p_I{}'[k] = h_{\mathbf{u}}[k] * \left\langle \varphi_I(\frac{v}{T} - k), I(v) \right\rangle = \left\langle \varphi_I(\frac{v}{T} - k) * \beta_{\mathbf{u}}(\frac{v}{T} - k), \hat{I}_I(v) \right\rangle$$

where $p_I{}'[k]$ is the sampled pixels of the swiped image after the frequencies ω_n have been filtered out by $h_{\mathbf{u}}[k]$, $\hat{I}_I(v)$ is a stream of differentiated Diracs at the switchpoint locations, and $\beta_{\mathbf{u}}$ is an e-spline created from the sinusoids of freqs ω_N [8]. Here, the fact that $p_I[k] = \left\langle \varphi(v-k), I_I(v) \right\rangle$ is also used.

This equation shows that the resulting signal $p_I{}'[k]$ is the equivalent of the inner product between differentiated Diracs at the switch points, and a new sampling kernel $\varphi_I{}'(t) = \varphi_I(\frac{v}{T} - k) * \beta_{\mathbf{u}}(\frac{v}{T} - k)$, which can be shown to still be able to reproduce exponentials [8].

To find the location of the switchpoints the moments of the signal can be calculated:

$$\tau_I[m] = \sum_{k \epsilon Z} c'_{m,k} p_I{}'[k]$$

where $c'_{m,k}$ fulfills:

$$\sum_{k \epsilon Z} c_{m,k}^{'} \varphi_I{}^{'}(v-k) = e^{\alpha_m v}$$

where α_m are the frequencies reproduced by the e-spline sampling kernel, $\beta_{\mathbf{u}}(k)$.

The moments of $p_I{}^{'}[k]$ are a set of weighted exponential, and can therefore be annihilated by a filter of the form:

$$G(z) = \prod_{d=1}^{4} (1 - e^{\lambda_f v_d} z^{-1})^{8N_s}$$

where λ_f is the spacing between frequencies in α_m, and where the roots of the filter $G(z)$ can be used to find the switchpoint locations. Finally, the values of x_{11}, x_{12}, and z_1 can be found from Eq. (3.2) to (3.5).

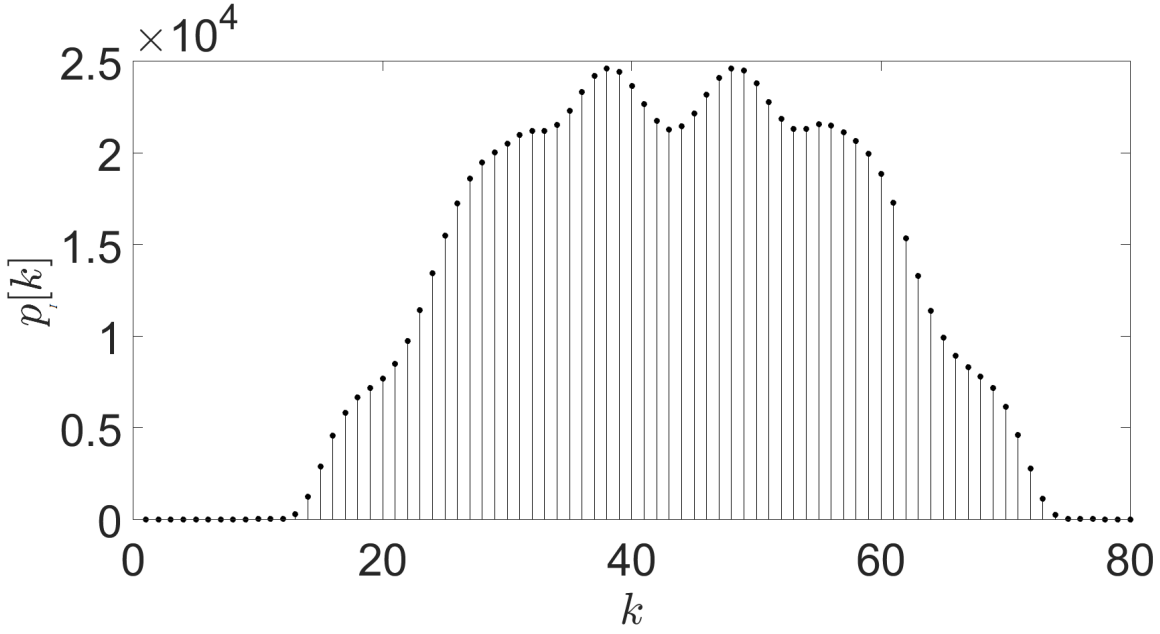


Figure 3.14: A sampled horizontal slice of a simulated swiped image of a fronto-parallel plane with a sinusoidal texture (i.e. $N_s = 2$).

Once the plane geometry and surface texture of the plane are determined, the image of the scene is rendered at camera location x_{02} using equation (3.17). To create the EPI, for each

camera position the plane in the image is shifted by an amount determined by (3.7). Given the estimated scene, the EPI for single plane scene is recovered for camera positions within the range $x_{01} < t < x_{02}$.

3.7 Ideal plane algorithm results

To verify the algorithms proposed in Sections 3.3, 3.4 and 3.6, experimental verification was performed. A DSLR camera attached to a sliding mount was moved between two points. A black board was used as the scene background. A large number of still images of the scene were taken as the camera slid between two points, which were linearly combined to form a swiped image, such as that shown in Figure 3.15(a). To reduce the impact of noise in the black background, pixel values below a certain threshold were set to 0. The threshold was chosen from the inspection of the real captured image samples, where the threshold was above the maximum value of the samples that were known to be of the black background.

3.7.1 Uniform texture plane results

The algorithm proposed in Section 3.4 was verified using MATLAB simulations, and by using real swiped images. MATLAB simulations were performed by using a Monte Carlo simulation, with the scene containing between 1 and 4 planes, with a black background. The depth, width and texture surface intensities were set from a uniform random distribution. For each scene, the EPI recovered was compared against the simulated ground truth EPI, as well as the reconstructed swiped image from the recovered EPI against the ground truth swiped image. It was verified that an EPI could be recovered from each scene that created a swiped image that was consistent with the original swiped image.

Real swiped images were created using printed sheets of paper, with a constant surface texture on each, that were setup as fronto-parallel planes. The impact of noise was reduced by the application of median filtering, which preserved the edges in the swiped image for switchpoint

detection. The horizontal size of the swiped image was 4000 pixels, and the length of the median filter was 20 pixels. A row of pixels from the centre of the swiped image was used for analysis, as the image was assumed not to vary in the vertical direction. The number of planes within the scene were 2 and 3, with different surface texture intensities for each plane, and with each plane edge being seen at some point during the camera swipe to avoid underdetermined systems. To check the validity of the recovered EPI, it was compared against the original still images that created the real swiped image.

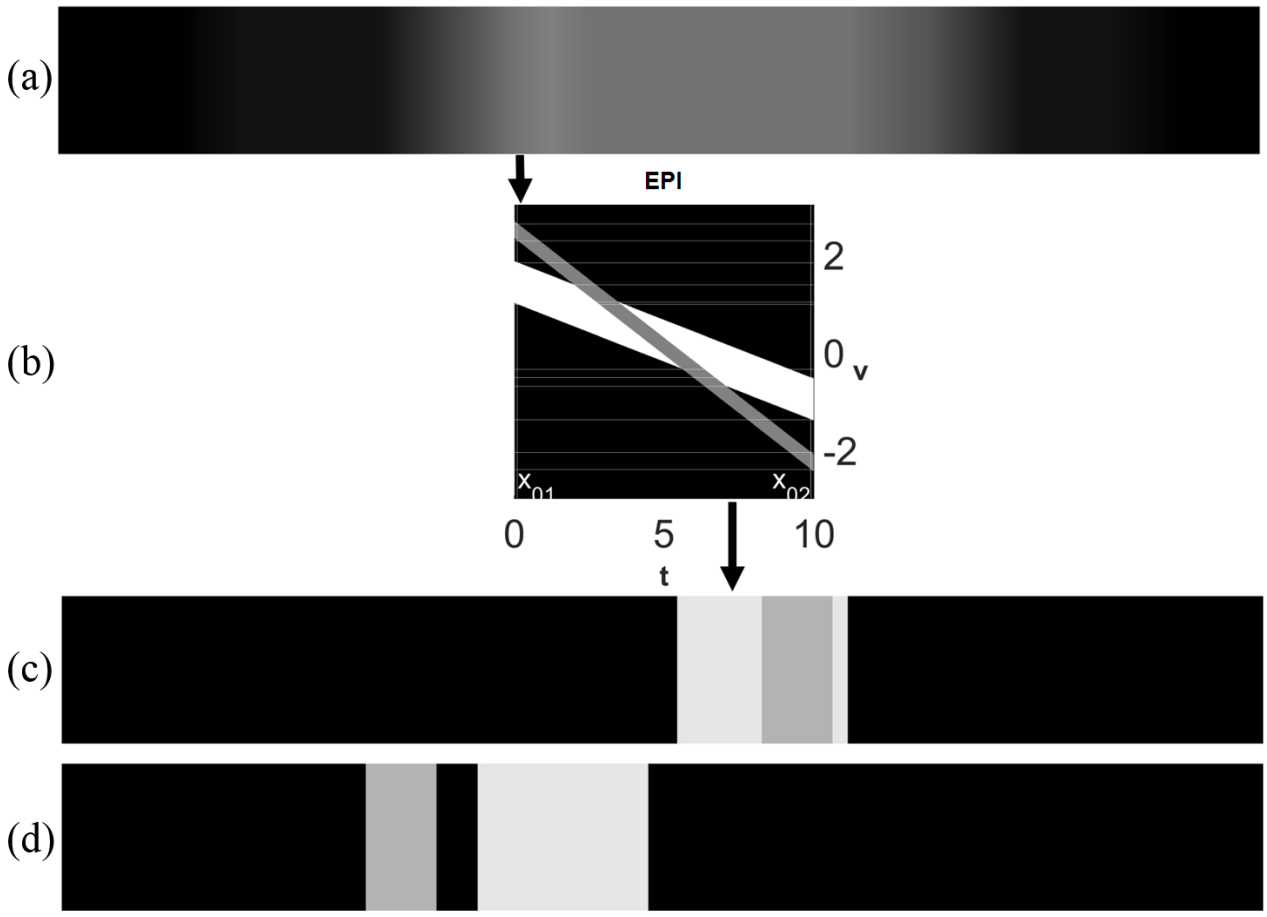


Figure 3.15: (a) shows a real swiped image created by opening the camera shutter as the camera moves. (b) shows the EPI recovered from (a). (c) and (d) show recreated single images of the scene from two different camera positions.

The algorithm proposed in Section 3.4 was verified as recovering the location of the planes in Figure 3.15 to within an accuracy of 0.1% of the true value, and the texture surface intensity to within 1.5% of the true value, as well as other setups fulfilling the conditions from Section 3.4. Using the information recovered, the EPI of the scene could be successfully reconstructed,

with the EPI being shown in Figure 3.15(b), and new images of the scene were generated, as shown in Figure 3.15(c) and Figure 3.15(d).

3.7.2 Sinusoidal texture plane results

A printed paper sheet of a sinusoidal texture surface, with a single frequency, was set up as a fronto-parallel plane to verify the algorithm. A large number of still images were taken, of which a subset are shown in Figure 3.16(a), and added together to form a swiped image of the scene. To simplify the analysis of the scene, there was no intensity variation in the vertical direction, which allowed the lens distortion to be modelled as a 1D convolution.

Figure 3.16(a) shows three of the original images which contain include in addition to the sinusoidal texture. In Figure 3.16(b), the resulting swiped image, which is very blurred, is shown, however it still contains all the information necessary to retrieve the original EPI. The original swiped image was subsampled by a factor of 50 to create a swiped image width of only 80 pixels. The noise present in the images resulted in the amplitude of the signal deviating from its ideal value. During the acquisition noise in the image is due to Gaussian noise, caused by the temperature of the image sensor, as well as shot noise, caused by the statistical quantum fluctuations of photons [57]. However, the averaging effect of combining multiple still images to form a swiped image reduces the impact of this noise, therefore the dominant source of "noise" in the real image is caused by the non-fully black nature of the background, and the non-perfect printing process for the plane texture surfaces. The frequency of the sinusoid could be estimated from any valid sliding window that excluded the switchpoints and contained only contributions from the sinusoid. To obtain the most accurate result, the frequency of the sinusoid was estimated for each valid sliding window and the results were averaged to produce a final estimate. Hard thresholding of background samples outside the length of the sampling kernel also was employed to reduce the impact of noise.

Use of the technique outlined in Section 3.6 showed that the wavelength of the sinusoid recovered from the swiped image in Figure 3.16(b) was 6.227cm, compared to the true wavelength of 6.272cm, within 1 percent of its true value. The locations of the switchpoints could be recovered

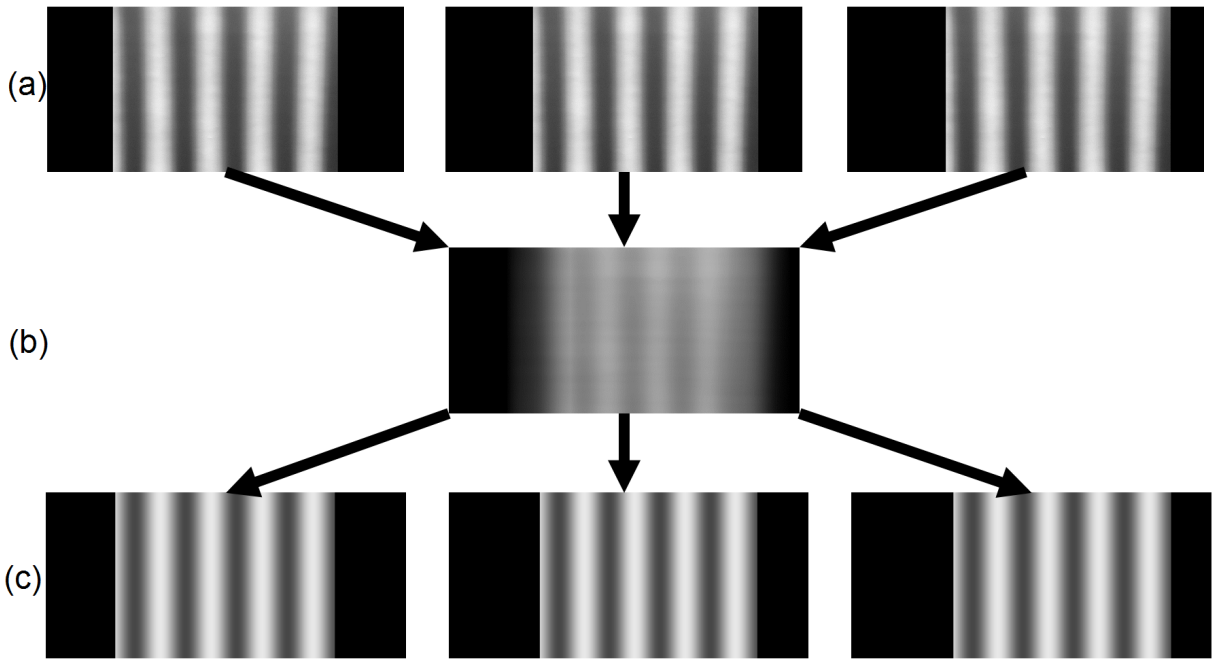


Figure 3.16: An example of EPI reconstruction from a swiped image. The images in (a) are combined into a swiped image (b), from which the EPI can be recovered and used to synthesise new images as in (c).

within half a pixel. From the recovered values, the EPI of the scene could be reconstructed. This EPI can be used to generate an image of the scene from an arbitrary camera location, as shown in Figure 3.16(c).

3.8 Conclusion

In this chapter, we demonstrate that the EPI of a scene comprising one or more uniform-texture fronto-parallel planes can be reconstructed exactly from a swiped image of the scene. We further demonstrate that it is possible to reconstruct the EPI given a more challenging form of scene, in which the swiped image is that of a flat plane with a texture that is a finite sum of sinusoids. This is possible given knowledge of the distortion introduced by the camera lens and despite the image being sampled.

We also show that there exist some swiped images that could have been produced from multiple EPIs. However, the scenes which result in a swiped image that suffering from this type of ambiguity require precise alignment of planes and textures, so are not likely given all the

possible scene configurations. Given a swiped image which could be captured from multiple possible scenes, one of the scenes is chosen to recover the EPI. Therefore the output of the algorithms in this chapter always results in an EPI that is consistent with the swiped image we have as an input.

Chapter 4

EPI recovery from real swiped images

4.1 Introduction

Previously, in Chapter 3, the problem of reconstructing the EPI is analysed for scenes comprising planes with uniform surfaces. In this chapter, the goal of recovering the EPI from a swiped image is extended to that of scenes that reflect real word environments. The techniques from Chapter 3 cannot be directly reused for this situation, as scenes in real world images almost never contain only uniform texture fronto-parallel planes. Therefore, the approach used in Chapter 3 must be modified to cope with planes with more complicated textures.

Extending the problem setup from the one in Chapter 3 to that of a real world scene is again achieved by modelling the scene as comprising layers at different depths, but with the plane surface textures corresponding to that of the objects within the scene, which has been shown to be a good approximation of real life scenes [5, 29, 28, 69]. The approach of detecting scene geometry using a combinatorial approach to match expected switchpoints cannot be extended to real world scenes, as switchpoints in the swiped image are harder to detect because the plane surfaces are varying. This difference can be seen in Figure 4.1, where Figure 4.1(c) is a row of a swiped image with uniform texture planes (Figure 4.1(a)), and Figure 4.1(d) is a row of a swiped image with planes that model a real life scene (Figure 4.1(b)).

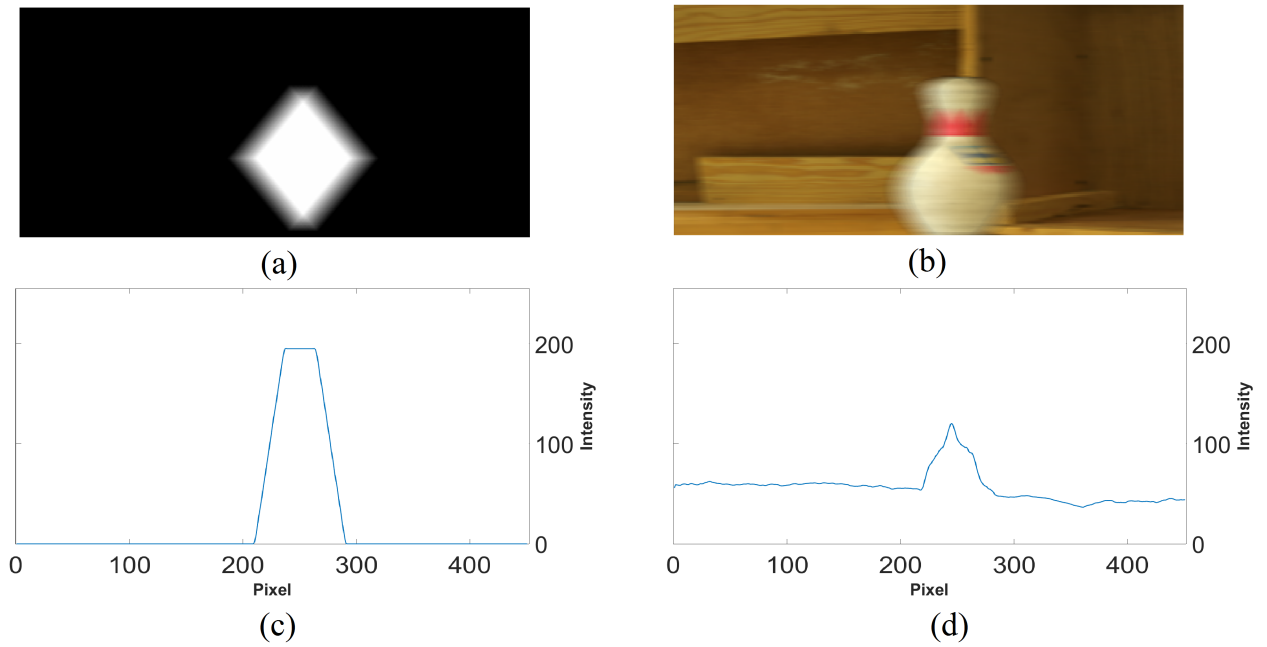


Figure 4.1: A swiped image made of planes with a uniform texture surface is shown in (a), whereas a swiped image made of planes that model a real life scene are shown in (b). The corresponding rows from the centre of the swiped images, shown in (c) and (d), demonstrate why the complexity of recovering the EPI increases for a real scene. Detection of switchpoints is more uncertain in a real scene, as it is unknown whether variations in intensity in the swiped image are caused by plane occlusion or surface texture. Furthermore, the switchpoints cannot be exactly paired in the real image to obtain the scene geometry.

The challenges of recovering the EPI of a real world scene include that of estimating the depths present within the scene, finding the silhouette of objects at these depths, recovering the texture surfaces of these objects, and finally rendering new images of the scene using the recovered objects and scene geometry. The challenge of estimating the depths present within the scene is addressed by creating a histogram of estimated depths within the image and choosing peaks as depths present within the scene. In this chapter an extension to the level set method whereby the outline of depth layers can be extracted is demonstrated. The outline of the depth layers can then be used to recover the surface intensities of the planes. Once the boundaries of the objects within the scene have been determined, the scene texture is determined, and the EPI is recovered.

The assumption that is used throughout this thesis is that the scene is static during the acquisition of the swiped image. This assumption will not always be true for a real world image, as the exposure interval for the shutter may be several seconds. Dynamic scenes complicate the

analysis of the swiped image, as the increased or decreased levels of motion blur could mislead the estimated depths within the image. This problem has much in common with the problem of obtaining a panoramic photograph, as discussed in Section 2.2.1, where the panoramic photograph aims to create a wider view of the scene than the field of view of the camera. As the process of creating a panoramic photograph using a single camera consumer device involves rotating the camera slowly, dynamic scenes also cause errors [31]. Methods for reducing the errors caused by dynamic scenes could include having a swiped image acquisition with a short exposure time, to minimise the possibility of movement, or discarding scenes in which movement in the scene is detected.

The outline of this chapter is as follows: Section 4.2 presents an overview of the proposed algorithm. Section 4.3 describes in detail the proposed method that allows the EPI to be calculated from the swiped image. Finally, we conclude in Section 4.4.

4.2 Algorithm Overview

4.2.1 Problem setup

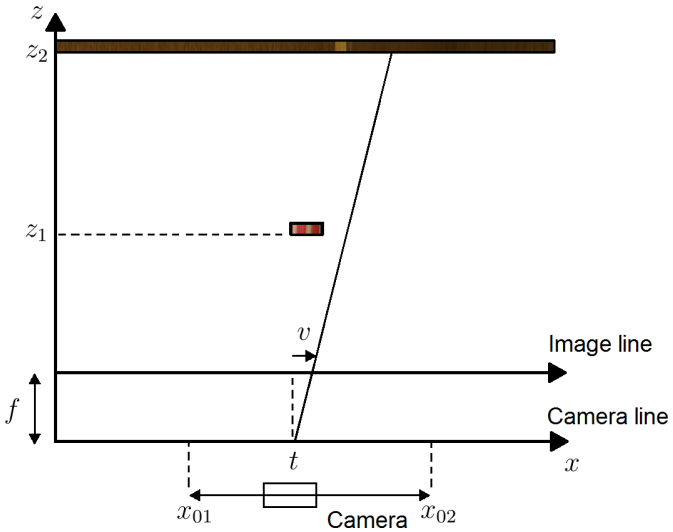


Figure 4.2: A view of the scene with two planes from above, where the plane furthest from the camera is the background plane. A light ray from a plane to the camera at position t forms an image at v. The camera is swiped from x_{01} to x_{02} with the shutter open.

To gain insights into the proposed algorithm, we first consider the idealised setting from Chapter 3 shown in Figure 4.2, and consider what needs to be changed to adapt this to a realistic scene. Our goal is to retrieve the EPI of a realistic scene from a swiped image. We achieve this by retaining the layer-based model in which we assume that a complex real-world scene is modelled as comprising P fronto-parallel planes. Each plane has a surface intensity corresponding to that of the real-life scene and the depth, or distance from the camera, of plane p is denoted z_p. This is illustrated in Figure 4.2, which shows a view from above for the case of $P = 2$. We assume that there is a plane that fills the field of view of the camera at all points during the swipe, which we term the "background plane", to ensure that the swiped image consists of only light rays from planes in the scene. The background plane is always the plane furthest from the camera, and is labelled as z_P. Using planes as depth layers is convenient for rendering and occlusion ordering.

As in Chapter 3, a camera is swiped from x_{01} to x_{02} at a constant velocity along a horizontal line parallel to the planes. We define the disparity, l_p, of each plane to be the shift in its location within the image as the camera moves between x_{01} and x_{02}:

$$l_p = \frac{f(x_{02} - x_{01})}{z_p}. \tag{4.1}$$

Given the swiped image $I_I(v, w)$, and knowledge of x_{01} and x_{02}, we wish to recover the EPI of the scene. A slice, $I_I(v)$, of the resultant swiped image is shown in Figure 4.3(a) and the corresponding EPI we aim to retrieve in Figure 4.3(b), plotted against the camera position, t. The slice, $I_I(v)$, is again obtained by using (3.1), integrating the EPI over the range $x_{01} < t < x_{02}$

A "switchpoint" in the swiped image is created every time one plane occludes or disoccludes another in the EPI. This can be seen in Figure 4.3(b) where the foreground plane occludes and disoccludes the background plane as the camera is swiped. Switchpoints in the idealised case result in discontinuities in the horizontal gradient of $I_I(v, w)$. In real images, switchpoints are

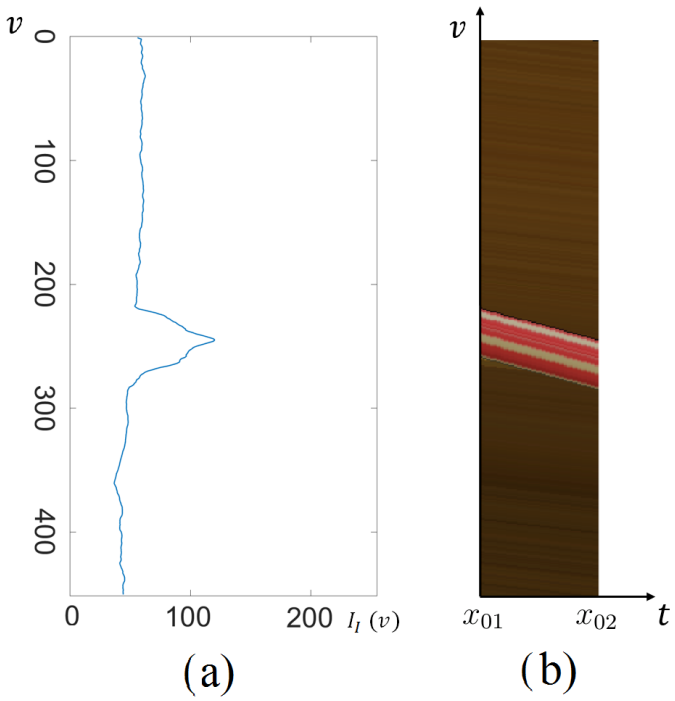


Figure 4.3: A row of the swiped image, (a), created by integrating the EPI shown in (b) for a two plane image with realistic plane texture surfaces.

detected by peaks in the rate of change of the horizontal gradient of $I_I(v, w)$ that are above a set threshold. Switchpoints are harder to detect in realistic images, as variations in $I_I(v, w)$ may be caused either by plane occlusion or by plane textures. The switchpoints that are caused by the zeroth plane will form an outline of each plane within the scene at the plane location at the start and end of the camera swipe. We therefore conclude that if we take the derivative of $I_I(v, w)$, its peaks will allow us to retrieve the planes and their depths [49].

4.2.2 Algorithm Description

Leveraging the analysis of the previous section for the idealised plane case, we present an algorithm that is able to retrieve the EPI from a real swiped image. The algorithm described in this chapter consists of four steps. In the first step, the depths of layers that are present within the swiped image are estimated (step (a) in Figure 4.4). This is achieved by finding peaks in the histogram of estimated disparities. The silhouette and location of the planes within the swiped image at each depth layer is then estimated (step (b) in Figure 4.4). An extension to the level set method which accounts for the amount of blur present within each

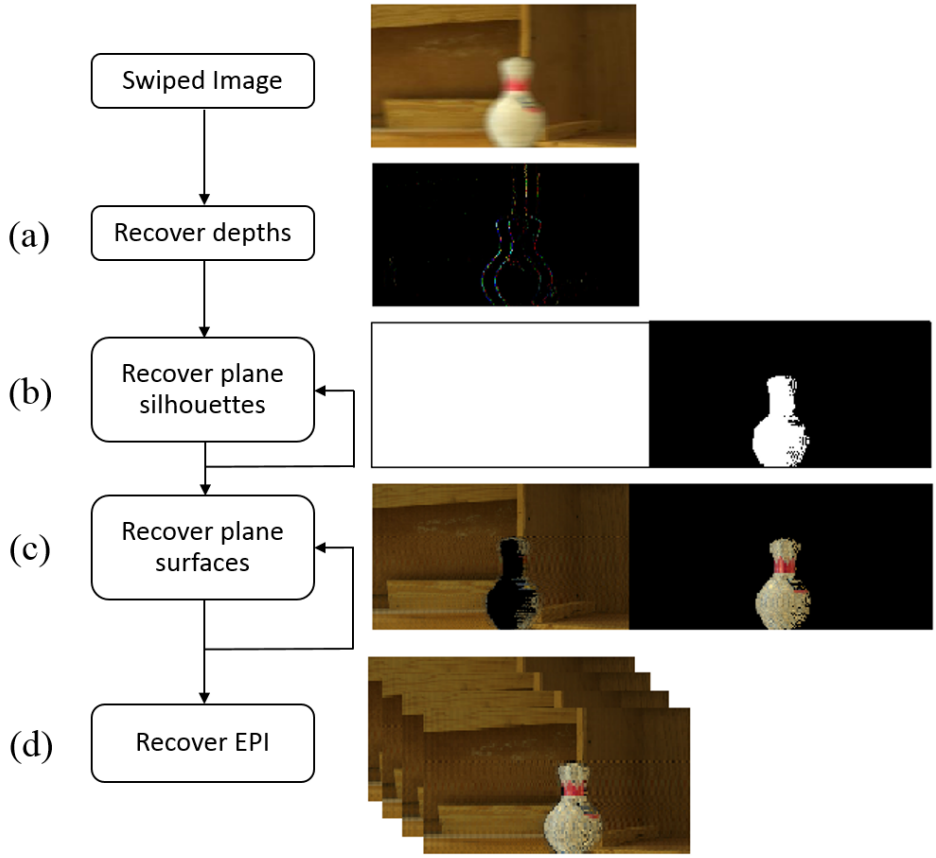


Figure 4.4: A flowchart of the steps in the algorithm to recover an EPI from a swiped image. The switchpoints in the swiped image are shown in the image at step (a). The silhouettes of planes at each depth are shown at step (b). The surface of the planes at each depth are shown at plane (c).

depth layer is used to establish a silhouette of each plane at the start of the camera swipe. The surface texture of each plane at each depth layer is then recovered (step (c) in Figure 4.4) using the estimated depths, plane silhouettes and plane locations to deblur from the swiped image. Finally new images of the scene may be rendered from any camera location within the original swipe range (step (d) in Figure 4.4) using the estimated depths, plane surface textures and plane locations. These four steps are described in detail in the following section.

4.3 EPI recovery from a swiped image

4.3.1 Recovering depth layers

To recover the depths of layers present in the swiped image, we first analyse the switchpoints derived from the swiped image. As was seen in Figure 3.4, a switchpoint is created in a given row in a swiped image every time a plane occludes or disoccludes another plane in the EPI as the camera is swiped. Switchpoints are found by detecting discontinuities in the horizontal intensity gradient of the swiped image. The occlusion of plane p by the zeroth plane will result in a pair of switchpoints separated by a distance, l_p, inversely proportional to the distance, z_p, of the plane from the camera. The detection of switchpoints by taking the horizontal intensity gradient of the swiped image increases the noise, leading to the detection of spurious switchpoints. However, in practice these spurious switchpoints are randomly distributed, whereas the switchpoints caused by a plane are always separated by a set distance, enabling detection of correct switchpoints. Switchpoints are plotted to form an image of switchpoints, $M_s(u, v)$, which is used to infer scene geometry. We aim to find the depth layers present within the scene by finding switchpoints which are separated by a set distance across multiple rows of the image of switchpoints. As we make no assumptions about the shape of the planes at this stage, we detect the relative distance between switchpoints by calculating the horizontal autocorrelation for each row. The horizontal autocorrelation for each row of the image of switchpoints can be calculated as:

$$R_{vv}(l, w) = \sum_{v \epsilon \mathbb{Z}} M_s(v, w) M_s(v - l, w) \tag{4.2}$$

where $R_{vv}(l, w)$ is the autocorrelation for each row of the image of switchpoints, M_s is the image of switchpoints and l is the lag. As the swiped image is an RGB colour image, we detect switchpoints for each colour channel independently. To prevent switchpoints matching with switchpoints from other colour channels, we calculate (4.2) separately for each RGB channel. We take the mean of the autocorrelation for each colour channel in each row to get the average

autocorrelation for the whole image, $\bar{R}_{vv}(l)$.

This is illustrated in Figure 4.5, which shows a swiped image in Figure 4.5(a), and the estimated image of switchpoints in Figure 4.5(b). The plane surface texture is white, which generates equal switchpoints in each of the RGB channels, resulting in the same autocorrelation for each channel in each row. The average autocorrelation, $\bar{R}_{vv}(l)$, of the rows of the switchpoint image is shown in Figure 4.5(c).

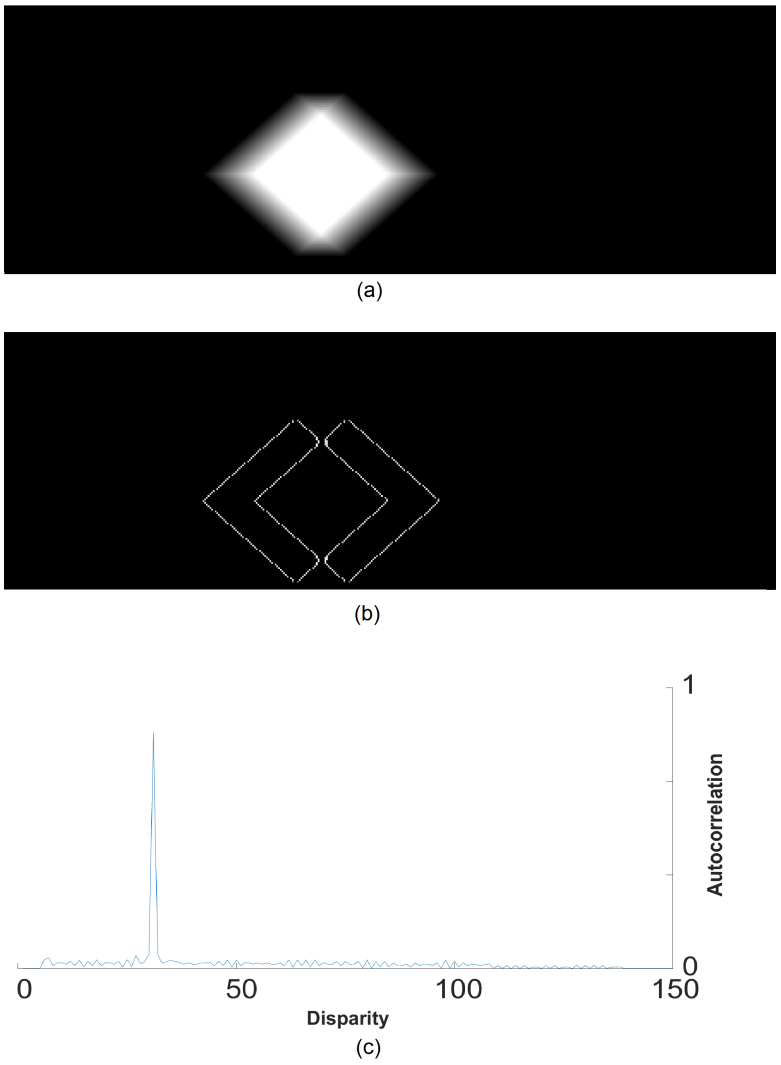


Figure 4.5: $I_I(v, w)$, shown in (a), leads to the image of switchpoints M_s, shown in (b), which contains the repeated outlines of planes. The autocorrelation histogram shown in (c) has a large peak for each depth, which in this image is a single depth with a disparity of 30.

Peaks in the autocorrelation indicate the shift due to motion blur of the planes within the image of switchpoints, and hence the depths within the swiped image. The lag in (4.2) is also referred to as the disparity of the planes. Although planes occluding each other at the start

or end of the camera swipe will supress the switchpoints of the more distant plane, we assume that sufficient switchpoints from each plane will remain to correctly estimate depths. This assumption is not always true for every scene, but if a depth layer is occluded for much of a swiped image acquisition, it will have only a small impact on the resultant EPI. The depths of the planes are then calculated from the disparity as:

$$z_p = f\frac{x_{01} - x_{02}}{l_p} \tag{4.3}$$

where l_p is the disparity of plane p at depth z_p within the swiped image, and f is the focal length of the camera. Using (4.3) requires knowledge of camera focal length and movement. From (4.2), we then generate a histogram of estimated disparities within the image.

To increase the robustness of detecting the correct depth layers, we create a disparity likelihood distribution for each pixel using the method outlined in [17]. The range of the distribution is the calculated maximum and minimum disparities of the layers within the scene. This procedure is illustrated in Figure 4.6, where Figure 4.6(a) is the swiped image. The peak of the likelihood distribution for each pixel is plotted as an image in Figure 4.6(b). We sum the likelihood distributions for all the pixels in the swiped image to create an additional histogram of disparities within the image, shown in Figure 4.6(c):

$$h_d(l) = \sum_{v=-v_m}^{v_m} \sum_{w=-w_m}^{w_m} d_d(v, w, l) \tag{4.4}$$

where $h_d(l)$ is the histogram of depth likelihoods derived from [17], v_m is the horizontal field of view of the camera, w_m is the vertical field of view of the camera, and $d_d(v, w, l)$ is the depth probability distribution for each pixel in the image, calculated using [17]. Similar to Figure 4.5, the switchpoint image and its average autocorrelation are shown in Figure 4.6(d) and (e). The histograms from Figure 4.6(c) and Figure 4.6(e) are normalised and added together to form a

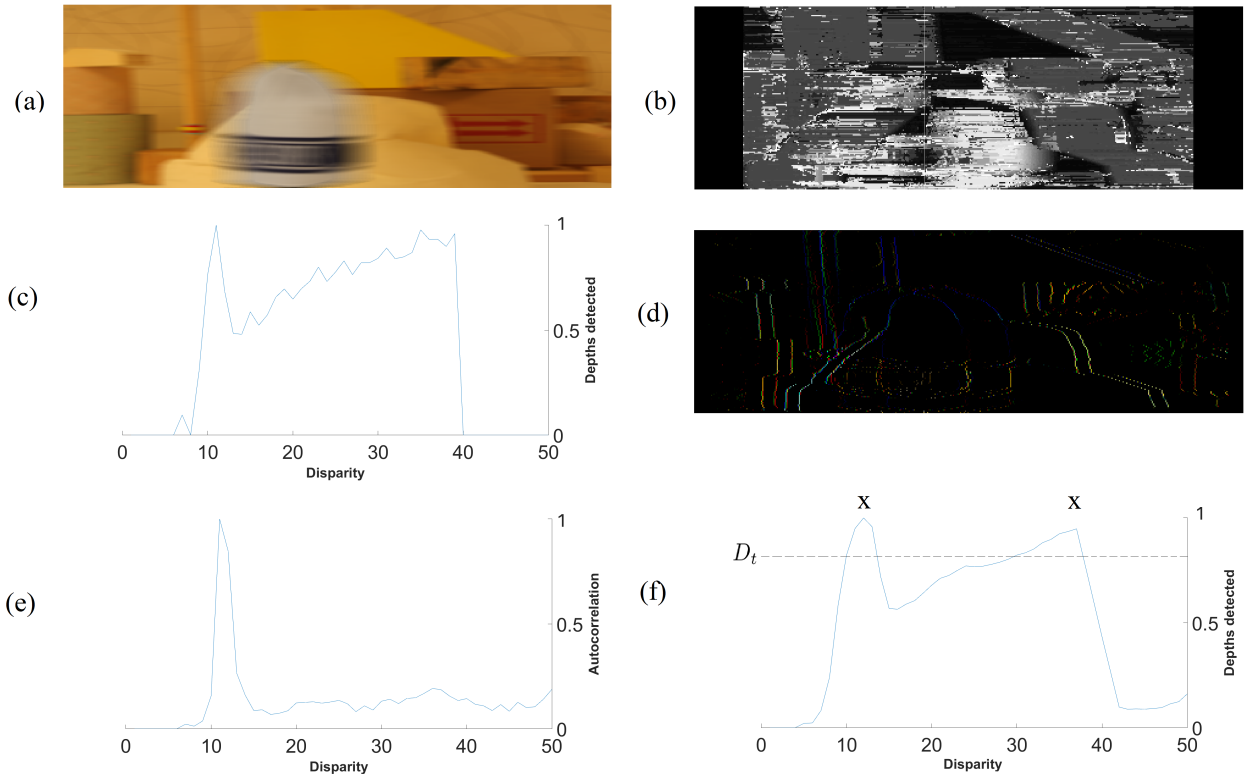


Figure 4.6: The technique in [17] is applied to the simulated swiped image in (a). This results in a depth likelihood distribution for each pixel. The peak of the depth distribution for each pixel is chosen as an estimated depth to form the estimated depth map shown in (b). The histogram of estimated disparities is shown in (c), and is created by summing the likelihood distributions for all pixels within the swiped image. The image of switchpoints (d) is found from the swiped image, which contains repeated outlines of planes. The autocorrelation histogram shown in (e) has a peak for each disparity. The two histograms are combined and smoothed with a moving average filter in (f) to produce a peak at each disparity above the threshold D_t, indicated by a dotted line. The disparity of each peak, indicated by an x, determines the depth of the estimated layers in the swiped image.

single histogram of disparities detected within the image:

$$h_c(l) = \frac{h_d(l)}{\max(h_d(l))} + \frac{\bar{R}_{vv}(l)}{\max(\bar{R}_{vv}(l))} \tag{4.5}$$

where $h_c(l)$ is the combined histogram of depth likelihoods, which is then smoothed by a moving average filter of length w_a, shown in Figure 4.6(f):

$$\bar{h}_c(l) = h_c(l) \circledast k_a \tag{4.6}$$

$$k_a = [\frac{1}{w_a}\frac{1}{w_a}...\frac{1}{w_a}\frac{1}{w_a}] \tag{4.7}$$

where $\bar{h}_c(l)$ is the smoothed combined histogram of depth likelihoods, k_a is the moving average filter and w_a is the length of the moving average filter. w_a was chosen to have the small value of 5, to eliminate local peaks without merging closely spaced layers together. Using this histogram, the number of depth layers is estimated as the number of peaks which are above a fraction, D_t, of the highest peak. The value of D_t is determined empirically. The depth of each layer is calculated from the location of the corresponding peak.

In a general scene with zero fronto-parallel planes, the depth histogram from the autocorrelation of the switchpoints will not have a single peak for each object, rather a spread of smaller peaks caused by switchpoints at continuous depths. In addition, the depth likelihood histogram derived from [17], will have a spread of smaller peaks. This is handled by smoothing the combined histogram with moving average filtering, quantising the depths in the scene into a discrete number of depth layers.

4.3.2 Recovering plane silhouettes

Once the depths of the layers present within the image have been found, the silhouettes of the planes at each depth are extracted. To obtain the EPI, the objects present within the scene that are at different distances from the camera must be segmented from each other. The use of the layer based model makes depth ordering explicit, so we aim to find the silhouettes of each of the planes that represent objects at each depth within the scene. For convenience, we choose to recover the plane silhouettes when the camera is at position x_{02}, the rightmost end of the swipe.

For a sharp image, segmenting into regions of interest based on pixel intensity can be achieved

using the level set method. The level set method was introduced by Osher and Sethian to track the movement of a front whose speed depends on the local curvature [67]. This front is defined as the locus of the zero level of a surface. The level set method has been used to segment images into different sections, in order to identify features of interest such as crystal growth [67, 66], flame propagation [67, 70] and cells [59].

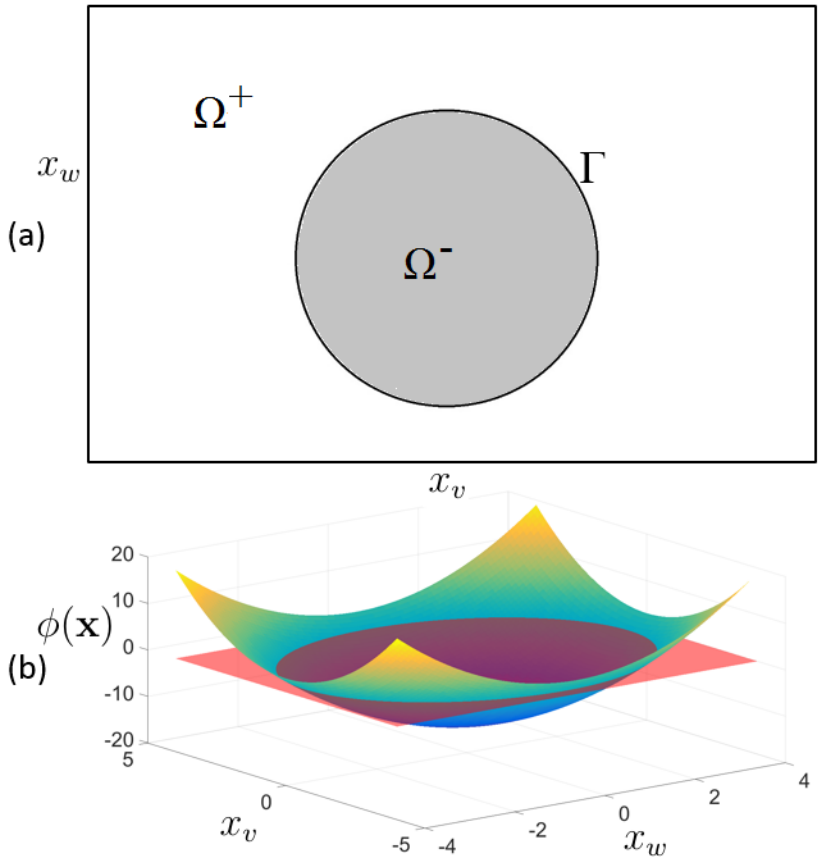


Figure 4.7: An image segmented into two regions (a), and the surface $\phi(\mathbf{x})$ that defines the boundary between the two regions (b) [66].

Consider an image $V(\mathbf{x})$, comprising of two regions which we wish to segment, as illustrated in Figure 4.7(a). $\mathbf{x}$ is a pixel index in the image, defined as the vector $\mathbf{x} = [x_v x_w]$, where x_v is a pixel index in the v axis and x_w is a pixel index in the w axis. To find the boundary, Γ, we define it as a function of ϕ, where $\phi(\mathbf{x})$ is a 2D surface that is a function of $\mathbf{x}$, such that:

$$\begin{aligned} \phi(\mathbf{x}) < 0 &\leftrightarrow x \in \Omega^- \\ \phi(\mathbf{x}) > 0 &\leftrightarrow x \in \Omega^+ \\ \phi(\mathbf{x}) = 0 &\leftrightarrow x \in \Gamma \end{aligned} \tag{4.8}$$

where $\mathbf{x}$ is a pixel index in the image. In this way, the two regions are defined by the single function, $\phi(\mathbf{x})$. The advantage of defining the regions Ω^- and Ω^+ using ϕ is that Γ is not explicitly defined, so regions can merge or split without hard decisions being made.

Typically the level set method searches for a boundary Γ between the regions, Ω^- and Ω^+, that minimises the energy of the function [84]:

$$E(\phi) = \int_{\Omega^+(\phi)} |V(\mathbf{x}) - c^+|^2 dx + \int_{\Omega^-(\phi)} |V(\mathbf{x}) - c^-|^2 dx \tag{4.9}$$

where $V(\mathbf{x})$ is a suitable function of the image that is to be segmented, $\mathbf{x}$ indexes the pixel, c^- is the mean of the pixel values in the $\Omega-$ region and c^+ is the mean of the pixel values in the $\Omega+$ region. Use of (4.8) demonstrates that a change in ϕ produces a change in the boundaries of Ω^+ and Ω^-, with (4.9) showing that this changes the energy of the segmentation.

To minimise the energy of (4.9), we evolve the function $\phi(\mathbf{x})$ over a time variable, τ, using a steepest descent method [84]. The objective of the level set method is to find the segmentation of the two regions that results in the lowest possible energy.

It is shown in [84] that ϕ can be evolved by gradient descent to minimise $E(\phi)$:

$$\frac{\partial \phi}{\partial \tau} = -\frac{\partial E}{\partial \phi}.$$

This PDE can be solved numerically by updating ϕ as follows:

$$\phi^{\tau+1} = \phi^{\tau} - \Delta\tau \frac{\partial E}{\partial \phi}$$

where [84]:

$$\frac{\partial E}{\partial \phi(\mathbf{x})} = -(|V(\mathbf{x}) - c^{-}(\phi^{\tau})|^2 - |V(\mathbf{x}) - c^{+}(\phi^{\tau})|^2)$$

where ϕ^{τ} is ϕ at iteration τ. At each iteration, τ, we calculate the updated areas of Ω^{+} and Ω^{-} and then calculate the mean of $V(\mathbf{x})$ within these two regions to obtain c^{+} and c^{-}.

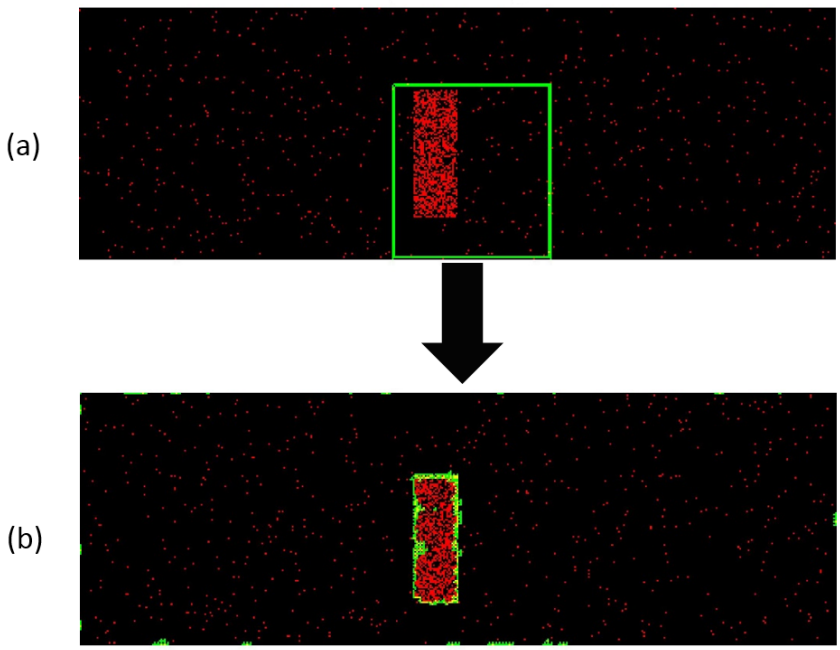


Figure 4.8: (a) shows the initial ϕ boundary for a noisy image of an object and (b) shows this object being successfully segmented.

When there is noise in the image, the level set algorithm may segment small, scattered regions which are not of interest. To counteract this, we penalise the mean curvature of $\phi(\mathbf{x})$, which is a local measure of the curvature of a surface for each pixel in ϕ. Figure 4.8(b) shows how a noisy image of an object is successfully segmented from a noisy image starting from an initial boundary shown in Figure 4.8(a). As a result, a region will not be segmented if the penalty

caused by the increase in curvature of $\phi(\mathbf{x})$ outweighs the decrease in energy of (4.9). Following [78, 66], we define the mean curvature of $\phi(\mathbf{x})$ as:

$$\kappa = \nabla \frac{\nabla \phi}{|\nabla \phi|}.$$

Therefore, the update equation becomes [84]:

$$\phi^{\tau+1} = \phi^{\tau} - \lambda_1 \frac{\partial E}{\partial \phi} + \lambda_2 \kappa \tag{4.10}$$

where λ_1 and λ_2 are the weights attached to the level set energy and curvature respectively.

Given sufficient iterations, the image is segmented into two regions based on the intensity of the pixel values. To successfully use the level set function to obtain a region of interest, the image being segmented must have features in the regions that allow them to be distinguished from each other. Previous implementations of the level set function have used the intensity of the pixels [66].

In our algorithm, the level set method is used to find a silhouette of each plane at each depth within the image determined by the technique in Section 4.3.1. To find features to segment using the level set method, we use a technique derived from Linear Discriminant analysis (LDA). This technique is used to build a model to find the common features of the pixels belonging to each depth layer. LDA is a well-known technique to extract features by dimension reduction, and has been used for applications such as image retrieval and face recognition [80, 47, 56]. The features used in this technique are the values of red, green and blue colour channels (RGB) for the pixels within the colour swiped image, which is transformed into a probability of belonging to each of the depth layers. The RGB values of the pixels are used rather than just the greyscale values as many realistic swiped images contain planes with different colour surface textures, but with similar greyscale values. In these situations use of RGB enables the planes at different

depths to be better distinguished from each other.

A model is generated based on the depth map from Section 4.3.1, in which each pixel is labelled with a depth layer label. To identify the pixels that have a sufficiently high likelihood of being labelled correctly, only the pixels with the ratio of the highest depth likelihood over the second highest depth likelihood above a set value, R_l, are chosen to build the model. Therefore, the inputs to build the model are a list of pixels, $\mathbf{x}_{p,i}$, labelled with the depth layer p to which they have been assigned. This is shown in Figure 4.9, where Figure 4.9(a) is the swiped image, and Figure 4.9(b) is a three-dimensional scatter graph showing the RGB value of pixels with labels represented by the two colours green and red.

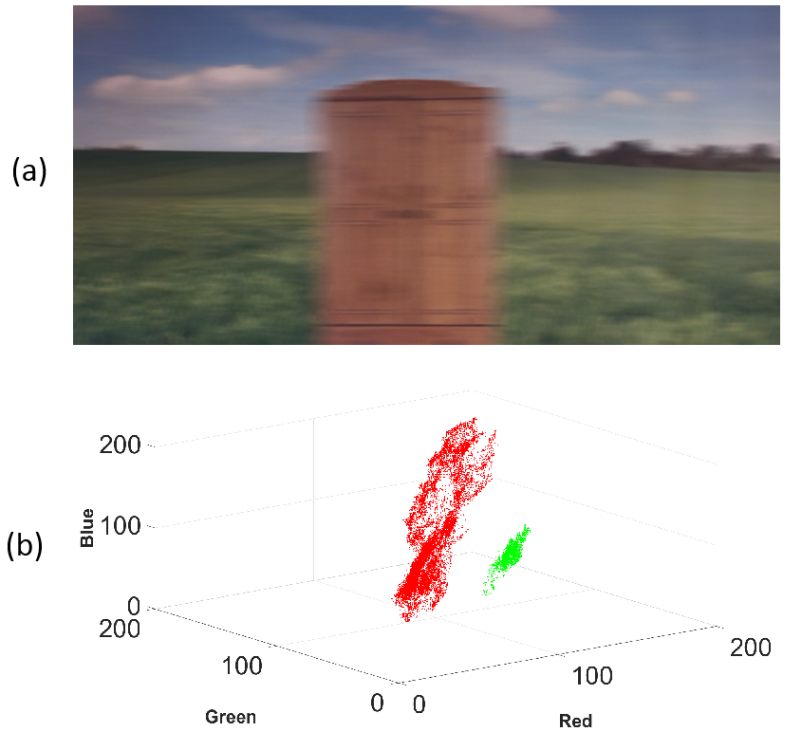


Figure 4.9: Pixels of a swiped image containing two planes shown in (a). In (b) green coloured observations are pixels from the foreground plane and red coloured observations are pixels from the background plane. High likelihood labelled pixels are used to predict a model to classify other pixels within the image by depth.

We assume that the intensity distribution of pixels belonging to depth p follows a multivariate Gaussian distribution, which is characterised by the sample mean and covariance of the pixels for each label. The sample mean is calculated as [41]:

$$\boldsymbol{\mu}_p = \frac{1}{N_p} \sum_{i=1}^{N_p} I_I(\mathbf{x}_{p,i}) \tag{4.11}$$

where $\boldsymbol{\mu}_p$ is the mean of the pixels labelled with p, N_p is the number of pixels labelled p and the column vector $I_I(\mathbf{x}_{p,i})$ is the RGB intensity of the ith pixel that was labelled with p from the swiped image. The sample covariance for each label is calculated as [41]:

$$\boldsymbol{\Sigma}_p = \frac{1}{N_p - 1} \sum_{i=1}^{N_p} (I_I(\mathbf{x}_{p,i}) - \boldsymbol{\mu}_p)(I_I(\mathbf{x}_{p,i}) - \boldsymbol{\mu}_p)^T \tag{4.12}$$

where $\boldsymbol{\Sigma}_p$ is the covariance for label p.

A likelihood is assigned to each pixel in the swiped image that it belongs to each depth layer:

$$V(\mathbf{x}|p) = \frac{1}{(2\pi|\boldsymbol{\Sigma}_p|)^{\frac{1}{2}}} e^{-\frac{1}{2}(I_I(\mathbf{x})-\boldsymbol{\mu}_p)^T(\boldsymbol{\Sigma}_p)^{-1}(I_I(\mathbf{x})-\boldsymbol{\mu}_p)}. \tag{4.13}$$

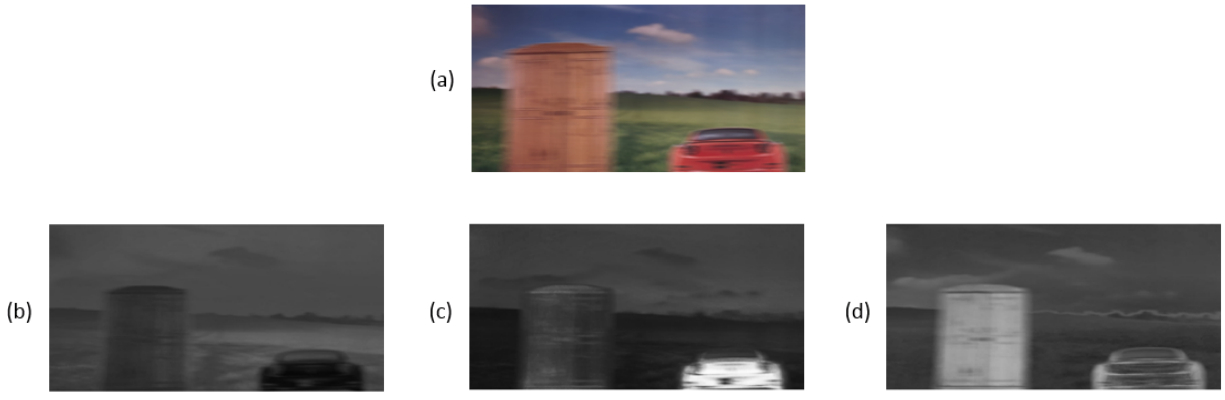


Figure 4.10: Pixels of a swiped image containing three planes at different depths shown in (a). The probability pixels are at depth 3, the background, is shown in (b), the probability pixels are at depth 2 is shown in (c) and the probability pixels are at depth 1 is shown in (d).

The equation in (4.13) is used to transform each pixel in I_I, seen in Figure 4.10(a), into a

likelihood of belonging to each depth layer, $V(\mathbf{x}|p)$ seen in Figure 4.10(b) to (d). Each pixel in each image is a scalar between 0 and 1, and the extended level set method operates on these images to get the silhouettes of planes at each depth. This transform makes the outline of each depth plane more reliable, as mislabelled pixels in the depth map are reclassified.

Now that we have the likelihood images, $V(\mathbf{x}|p)$, for each layer, p, we apply the level set method on each depth to find the silhouette of each plane.

An issue with the use of the canonical level set method for blurred images is that the blur introduced by camera motion with the shutter open produces a boundary between segments that is not sharp. Attempts to segment blurred images become more challenging as the blur within the images increases. We therefore modify the level set method to account explicitly for the blur at each depth to segment the swiped image into depth layers.

To segment the swiped image, we consider each of the depths independently. We consider all colour channels together, as different colour channels cannot have different scene geometry. The plane that is the furthest away from the camera, z_P, is defined as the background plane, and is assumed to cover the entire background of the swiped image. Therefore, for this depth layer ϕ_P is uniformly negative. We assume for now that the planes z_1 to z_{P-1} do not occlude each other, but occlude only the background. The case of planes occluding each other is discussed in Section 4.3.4.

For each non-background plane, we model the situation as the plane at that depth, z_p, occluding only the background plane. All other planes are considered as part of the background. We define a different level set surface $\phi_p(\mathbf{x})$ to determine the silhouette of each non-background plane. To achieve that we introduce the following change to the calculation of energy for the level set method, defining the energy of the image at depth p as:

$$E_p(\phi_p(\mathbf{x})) = \lambda_2 \kappa_p + \lambda_1 \int |V(\mathbf{x}|p) - V_{c_p}(\phi_p(\mathbf{x}))|^2 dx \tag{4.14}$$

where $V_{c_p}(\mathbf{x})$ is the best approximation for the image $V(\mathbf{x}|p)$ using two regions with set values, blurred with a known blurring kernel that depends on the depth of the plane.

The approximation of $V(\mathbf{x}|p)$ is obtained from $\phi_p(\mathbf{x})$ as follows: we first introduce

$$V_{\phi_p}(\mathbf{x}) = \begin{cases} 1 & \phi_p(\mathbf{x}) < 1 \\ 0 & \text{otherwise} \end{cases}$$

where $V_{\phi_p}(\mathbf{x})$ is the current estimate of the silhouette of the depth layer given ϕ_p. Moreover,

$$V_{b_{p-}}(\mathbf{x}) = h_p * V_{\phi_p}(\mathbf{x})$$
$$V_{b_{p+}}(\mathbf{x}) = h_p * (1 - V_{\phi_p}(\mathbf{x}))$$

where h_p is a blurring kernel that is a box function with a length determined by that of the depth of the layer being segmented. It is assumed that the blur kernel for each depth is a box. It is possible to have other blur kernels due to defocus blur [54, 53] and due to distortions from the lens. Obtaining depth from defocus has been previously studied in [54, 53], but we assume that all objects within the scene are at perfect focus, as the range of depths is small compared to many scenes studied in depth by defocus, and therefore the effects of motion blur will dominate.

The value of $\mathbf{c}_{b_p}$ is chosen to minimise:

$$\|\mathbf{V}_{b_p}\mathbf{c}_{b_p} - \mathbf{v}_p\|^2$$

where

$$\mathbf{V}_{b_p} = \begin{bmatrix} V_{b_{p-},1} & V_{b_{p+},1} \\ V_{b_{p-},2} & V_{b_{p+},2} \\ \vdots & \vdots \\ V_{b_{p-},n} & V_{b_{p+},n} \end{bmatrix}$$

and

$$\mathbf{c}_{b_p} = \begin{bmatrix} c_{b_{p-}} \\ c_{b_{p+}} \end{bmatrix}$$

where n is the number of pixels, $V_{b_{p-},1}, V_{b_{p-},2}$ ... $V_{b_{p-},n}$ are the elements of $V_{b_{p-}}(\mathbf{x})$ and $V_{b_{p+},1}$, $V_{b_{p+},2}$ ... $V_{b_{p+},n}$ are the elements of $V_{b_{p+}}(\mathbf{x})$, and $\mathbf{v}_p$ is the elements in $V(\mathbf{x}|p)$ formed into a column vector. Finally, the best approximation to $V(\mathbf{x}|p)$ is given by:

$$\mathbf{V}_{c_p} = \mathbf{V}_{b_p}\mathbf{c}_{b_p}$$

Given the equation for the energy of the segmentation, (4.14), the change in energy due to a change in the value of $\phi_p(\mathbf{x})$ is determined. For each pixel, the value of $\phi_p(\mathbf{x})$ is changed to $-\phi_p(\mathbf{x})$, thereby reversing the sign of $\phi_p(\mathbf{x})$ for that pixel and changing the boundary. Given the new boundary, the new energy is calculated. This gives a numerical method for estimating the partial derivative:

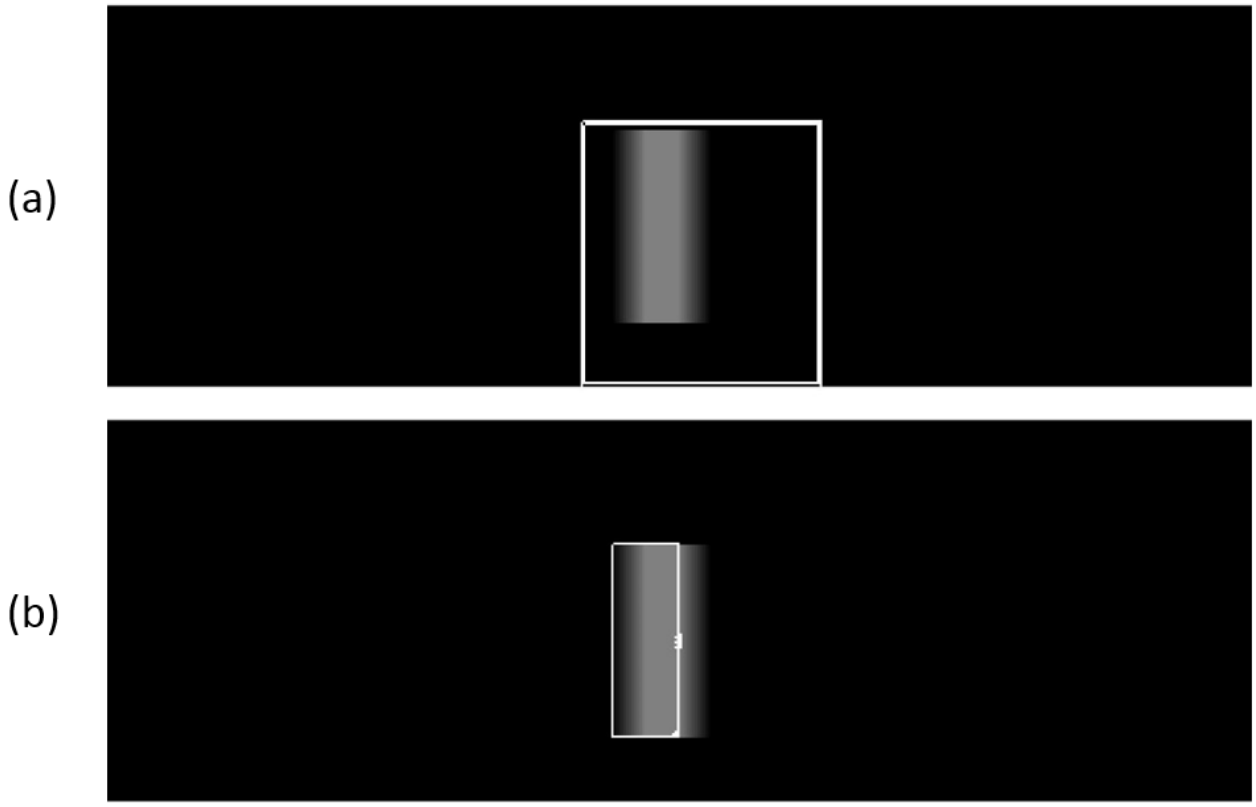


Figure 4.11: (a) shows the initial ϕ boundary for a blurred object and (b) shows this object being successfully segmented. The white outline segments the plane silhouette for the rightmost camera position.

$$\frac{\partial E_p}{\partial \phi_p}(\mathbf{x}) = \frac{E_p(\phi_{+_p}(\mathbf{x})) - E_p(\phi_{-_p}(\mathbf{x}))}{|\phi_p(\mathbf{x})|} \tag{4.15}$$

where $\phi_{+_p}(\mathbf{x})$ is the ϕ_p at the current iteration where the value of ϕ_p at $\mathbf{x}$ is positive and $\phi_{-_p}(\mathbf{x})$ is the ϕ_p at the current iteration where the value of ϕ_p at $\mathbf{x}$ is negative.

In principle the energy of the entire image must be calculated for every pixel being considered. To reduce the amount of calculation required, it is assumed that only pixels within a predefined distance of the boundary are likely to reduce the energy with a sign change, and so only these pixels need to be considered. This is known as the narrow band level set method [1].

Using (4.15) in (4.10), we evolve each non-background ϕ_p, as in Figure 4.11, for a fixed number of iterations to estimate the plane silhouette for each non-background plane.

4.3.3 Recovering plane surfaces

Once the depths in the scene have been found using the method in Section 4.3.1, and the plane silhouettes and locations using the method in Section 4.3.2, the texture surfaces of the planes may be recovered.

For the simplest scene type, in which there is only a single plane, a sharp image of the plane can be turned into the swiped image by a simple linear convolution of the sharp plane image with a 1D box function with a length equalling the disparity of the plane due to the depth. Therefore, to recover the surface of the original plane from the swiped image, the well-studied techniques in the field of deblurring will produce a good result.

When we consider a more realistic scene comprising multiple planes, the process of recovering the plane surfaces becomes more complicated. This is because as the camera is swiped, planes at different depths will have different disparities in the swiped image and may occlude each other due to their positioning within the scene. The effect of these occlusions turns the problem of recovering plane surfaces from a linear one to a non-linear one, making the recovery more difficult. To recover the plane surfaces, we first consider a situation including planes of different depths which do not occlude each other.

Planes within the scene at different depths will have a disparity during the camera swipe inversely proportional to their depth, and will have different occlusions depending on depth, so are recovered separately. To get the forward model of the effect of camera swiping, we define a matrix $\mathbf{A}_p$ that when multiplied by a row of the sharp image of a plane will produce a row of the swiped image of the plane:

$$\mathbf{A}_p \tilde{\mathbf{v}}_p = \mathbf{I}_{I,\,p} \tag{4.16}$$

where $\mathbf{A}_p$ is the swiping matrix, measuring $n \times m$, where n is the width of the swiped image, m is $n + b$, where b is a buffer the size of the maximum disparity expected in the swiped image,

to account for planes wider than the width of the image. $\tilde{\mathbf{v}}_p$ is a transposed row of the sharp image of plane p measuring m by 1 and $\mathbf{I}_{I,\,p}$ is a transposed row of the swiped image of plane p measuring n by 1.

The $\mathbf{A}_p$ matrix in (4.16) is populated with the following elements to model the effect of swiping the camera:

$$a(i,i) = \begin{cases} 0.5(2L-(L-M)^2)/L & \text{if } M=0 \\ 0.5/L & \text{if } M>0 \end{cases}$$

$$a(i,i-1) = \begin{cases} 0.5(L-M)^2/L & \text{if } M=0 \\ 0 & \text{if } M>0 \end{cases}$$

$$a(i,j) = \begin{cases} 0 & \text{if } M=0 \\ 1/L \text{ for } i-M<j<i & \text{if } M>0 \end{cases}$$

$$a(i,i-M) = \begin{cases} 0 & \text{if } M=0 \\ \frac{0.5(1+2L-2M-(L-M)^2)}{L} & \text{if } M>0 \end{cases}$$

$$a(i,i-M-1) = \begin{cases} 0 & \text{if } M=0 \\ \frac{0.5(L-M)^2}{L} & \text{if } M>0 \end{cases}$$

where L is the disparity, M is floor(L), i is the row index, and j is the column index.

To find the forward model of a multiple-plane scene, we concatenate the $\mathbf{A}_p$ matrices and $\tilde{\mathbf{v}}_p$ vectors for each plane to form:

$$\mathbf{A} = [\mathbf{A}_1, \mathbf{A}_2 ... \mathbf{A}_P] \tag{4.17}$$

and

$$\tilde{\mathbf{v}} = [\tilde{\mathbf{v}}_\mathbf{1}^T; \tilde{\mathbf{v}}_\mathbf{2}^T; ...\tilde{\mathbf{v}}_\mathbf{P}^T]^T \tag{4.18}$$

where $\mathbf{A}$ is the swiping matrix for the entire scene and $\tilde{\mathbf{v}}$ is a concatenated row of the sharp image of planes at each depth layer. Use of these matrices gives the equation:

$$\mathbf{A}\tilde{\mathbf{v}} = \hat{\mathbf{I}}_I \tag{4.19}$$

where $\hat{\mathbf{I}}_I$ is defined as:

$$\hat{\mathbf{I}}_\mathbf{I} = \sum_{p=1}^{P} \mathbf{I}_{I,\,p} \tag{4.20}$$

and represents the swiped image from the planes in the absence of the effects of occlusions.

The difficulty of recovering the scene is due to the non-linear effects of the occlusions on the swiped image. To account for the occlusions of the planes on the swiped image, we modify the $\mathbf{A}_p$ matrices in depth order, taking advantage of the explicit depth ordering of the layer based model. $\mathbf{A}_1$ is related to the closest plane to the camera, and so cannot be occluded by another plane, and remains unchanged. The entries of $\mathbf{A}_2$ which are calculated to be occluded by plane 1, by our knowledge of plane silhouette and location from Section 4.3.2, are deleted. This process is repeated for each layer in turn, with the elements occluded by planes closer to the camera changed to zero. The $\mathbf{A}$ matrix for the situation where the effect of occlusions is disregarded is shown in Figure 4.12(a). The $\mathbf{A}$ matrix incorporating the effects of occlusions is shown in Figure 4.12(b), where some of the elements for the plane further from the camera have been changed to zero due to occlusion.

The forward model in which (4.19) is used to obtain a row of the swiped image is illustrated in



Figure 4.12: (a) shows the **A** matrix for a scene with two planes without considering the effects of occlusion. (b) shows **A** for the same scene when occlusion is considered. The elements for the closer plane (on the right), are unchanged, but some elements for the further plane (on the left) are set to zero.

Figure 4.13. A row of the image in Figure 4.13(a) is generated from the multiplication of the **A** matrix with the concatenated sharp images of planes, $\tilde{\mathbf{v}}$, shown in Figure 4.13(b).

The **A** matrix is different for each row due to variations of the scene geometry with vertical image position. Using the minimum solution to the following equation, we estimate each row of the original images of the planes from the swiped image using:

$$\|\mathbf{A}\tilde{\mathbf{v}} - \mathbf{I}_I\|^2. \tag{4.21}$$

If the swiped image is a colour image, (4.21) is repeated for each RGB colour channel to estimate each colour component of the original planes. The equation in (4.21) is not well posed due to low pass filtering effects of swiping the camera. In situations such as this, standard least

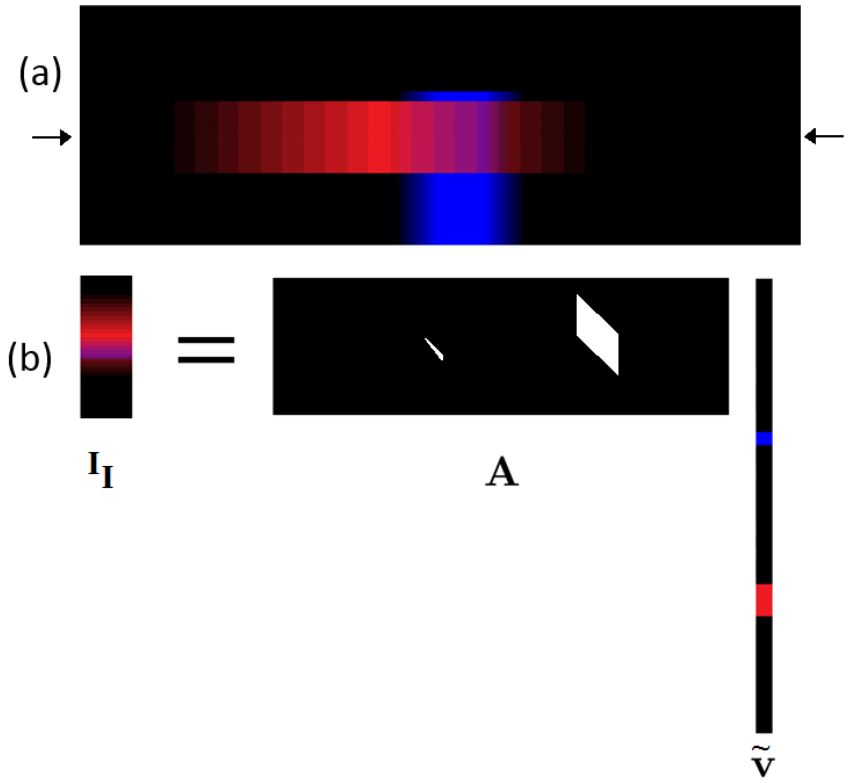


Figure 4.13: Each row of the swiped image, I_I shown in (a), can be formed by a row of the sharp image of each plane in $\tilde{\mathbf{v}}$ multiplied by $\mathbf{A}$. The resultant row of the swiped image created in (b) is indicated in (a) by arrows. In this figure, there are two depths present within the image, related to a red and blue plane shown within $\tilde{\mathbf{v}}$.

squares estimation can produce unwanted effects due to the large norm of the least squares solution. To obtain the best estimate of the original planes, we use the regularisation method of the iterative shrinkage thresholding algorithm (ISTA) to stabilize the solution [4, 18]. The results of a rendered view from a recovered EPI generated from Figure 4.14(a) are seen in Figure 4.14(b) when estimating the planes with least squares estimation and Figure 4.14(c) when using ISTA.

When estimating plane surfaces directly from the swiped image, boundary effects from the edge of the background plane are seen. To alleviate this, the swiped image and the $\mathbf{A}$ for each row were "mirrored", with a mirrored extension on each horizontal edge. The mirrored sections recovered planes containing the artifacts were then discarded when recovering the EPI. This process is illustrated by Figure 4.15, with the EPI generated from Figure 4.15(a) resulting in a rendered view in Figure 4.15(c) that contains artifacts at the edge of the background plane.

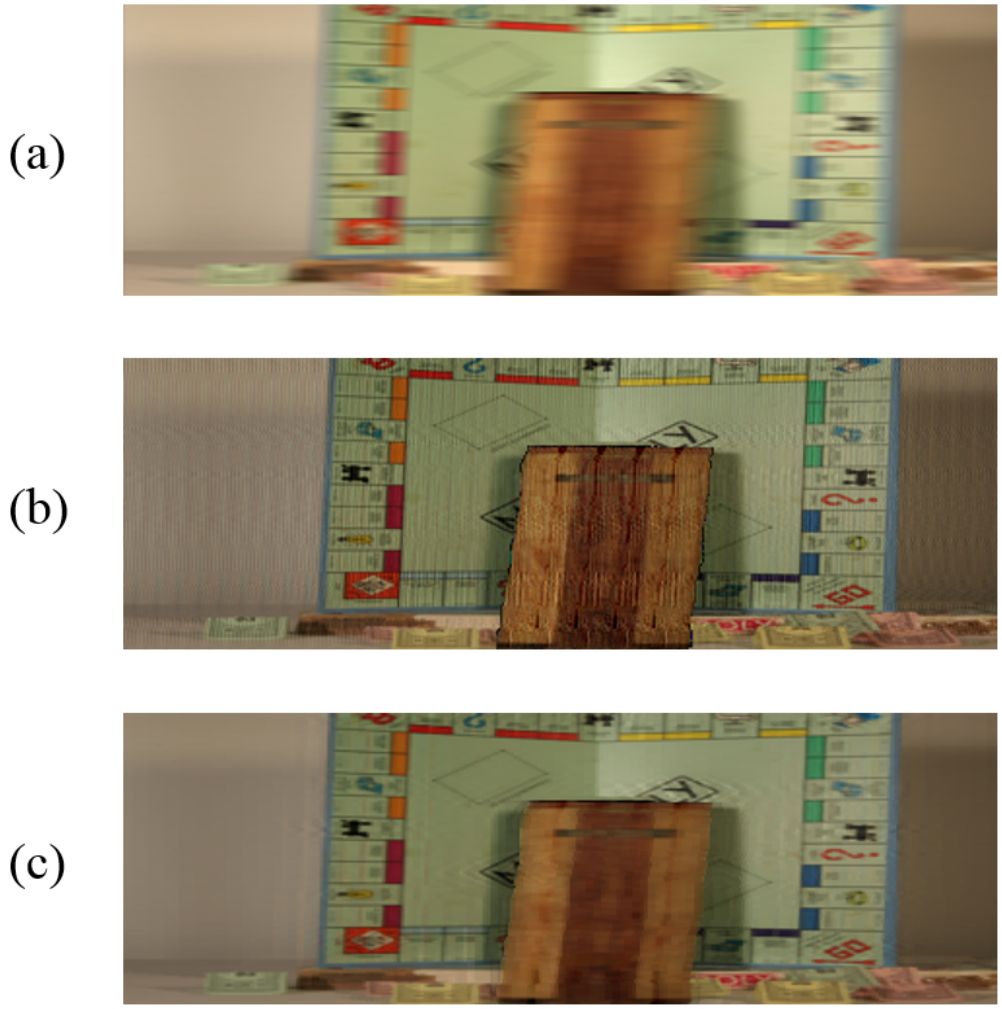


Figure 4.14: The swiped image in (a) leads to the new views of the scene shown in (b) and (c). In (b), least squares estimation is used to obtain the estimate of the plane surfaces, leading to artifacts. ISTA is used to regularise the estimated plane surfaces in (c), minimising these artifacts. Swiped image constructed from cropped planes in [72, 38].

The swiped image in Figure 4.15(a) is mirrored in Figure 4.15(b), resulting in the rendered view in Figure 4.15(d) that does not have artifacts caused by boundary effects.

4.3.4 Plane recovery optimisation

The technique outlined above can be used to recover the plane surfaces within the image, given an estimate of the plane silhouettes from Section 4.3.2. In this section, we describe a further method of refining the recovery of the planes. The reasons for using this are twofold, with the

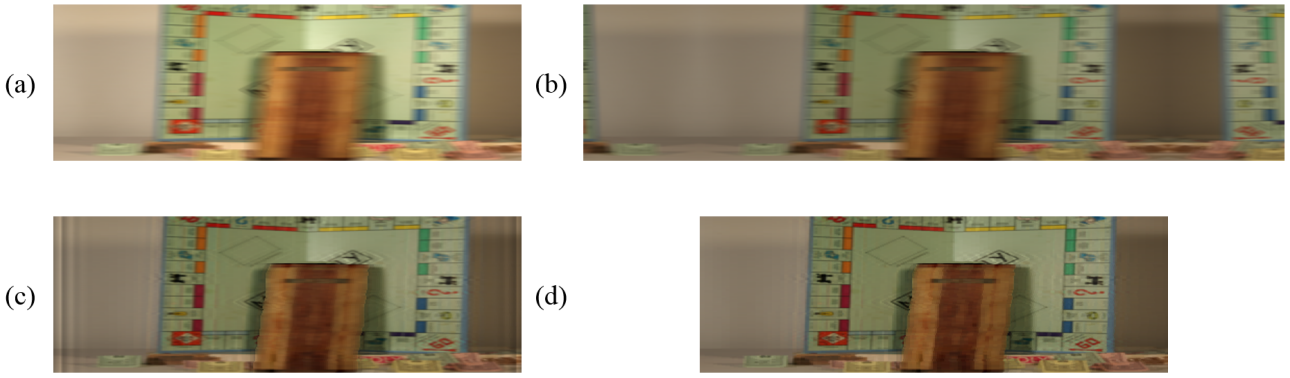


Figure 4.15: The swiped image in (a) is mirrored, leading to (b). In (c), a rendered view of the scene taken from an EPI generated from (a) is seen, with artifacts at the edge of the background plane. In (d), a rendered view of the scene taken from an EPI generated from (b) is seen without artifacts caused by boundary effects. Swiped image constructed from cropped planes in [72, 38].

first being that the assumption from Section 4.3.2, that non-background planes only occlude the background and not each other, is not generally true in real world images. Therefore, we get a good result when non-background planes do not overlap, but inaccuracies in the plane silhouettes when they do. The second is that the more accurate the plane silhouettes are, the more accurate the recovered plane surfaces and therefore recovered EPI will be. When the planes in the image are segmented wrongly, the content of the planes will leak into each other in $\tilde{\mathbf{v}}$, as can be seen in Figure 4.16(e). This leakage does not occur when plane silhouettes are correct, such as in Figure 4.16(b). Minimising the error in the boundary estimate will minimise this leakage.

The level set method in Section 4.3.2 operates on a transform of the swiped image, the probability of each pixel belonging to each depth. When non background planes occlude each other in the swiped image, then a pixel can belong to more than one plane. Use of the algorithm from Section 4.3.2 on the 3 plane swiped image in Figure 4.17(a), will result in incorrect plane silhouettes where the planes overlap, resulting in the recreated swiped image shown in Figure 4.17(b).

To solve the problem of overlapping planes, we use the level set method again, but now calculate ϕ using $\tilde{\mathbf{v}}$, the image of recovered planes. This method uses the recovered planes, where a change in the silhouette of a plane results in a different plane surfaces at other depths due to occlusion.

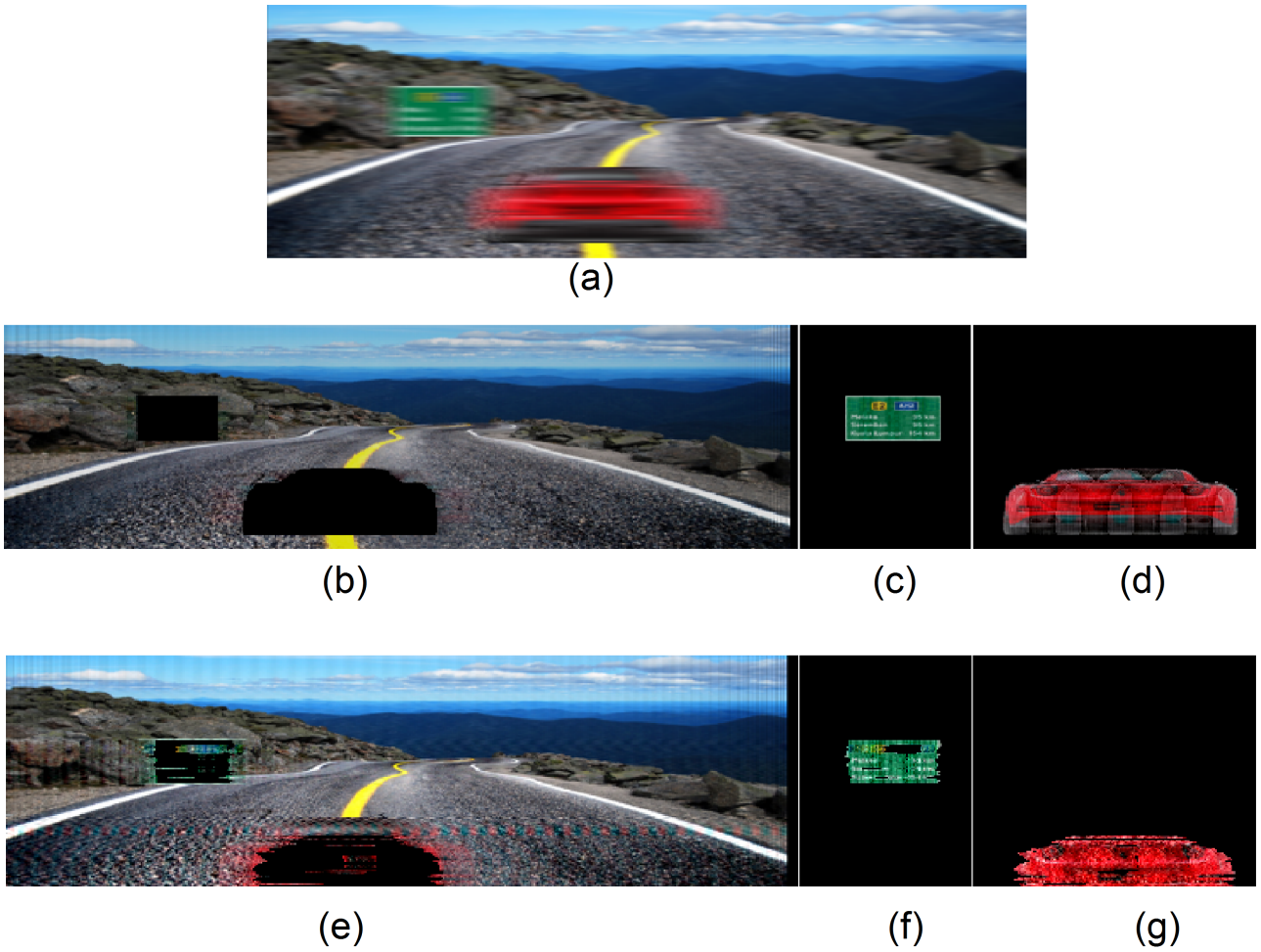


Figure 4.16: Image of recovered planes at depth 3 (b), depth 2 (c), and depth 1 (d), from swiped image (a) when the depths and planes outlines are correct. When plane silhouettes are incorrect the recovered planes at depth 3 (e), depth 2 (f), and depth 1 (g) are incorrect.

Therefore, the ϕ for each depth cannot be considered separately, and so ϕ is defined as:

$$\phi = [\phi_1, \phi_2, ...\phi_P]$$

where ϕ_1, ϕ_2 and ϕ_P are the level set surfaces for depth layers 1,2 and P respectively. As the recovered planes are separated by depth, any pixel in the swiped image can belong to a combination of planes. For this method, we use the RGB value of a pixel $\tilde{\mathbf{v}}(\mathbf{x})$ in a recovered plane, and compare it to the RGB distributions of the planes at each depth. If the recovered pixel closely matches the colours of the planes at its depth, then we assume the pixel is correctly

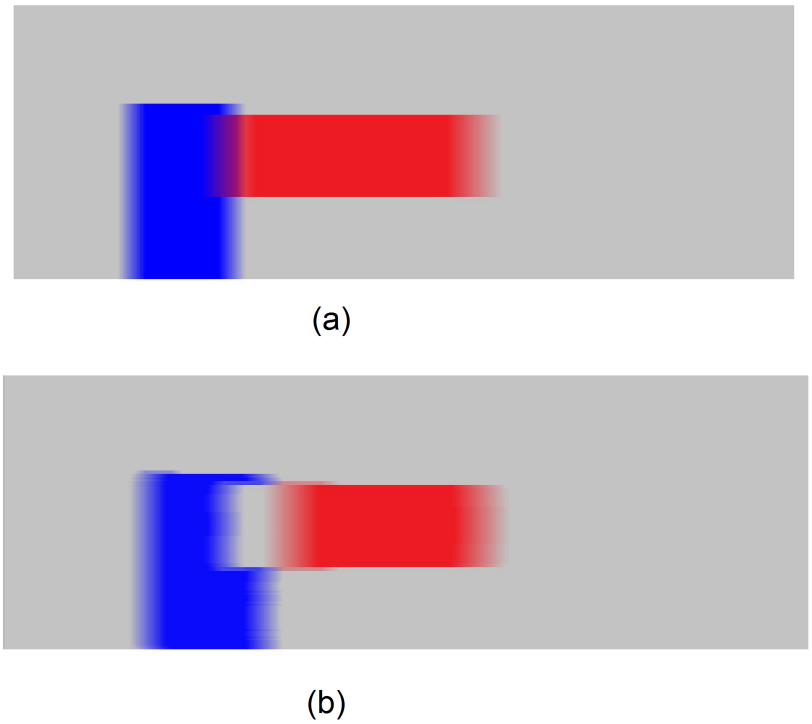


Figure 4.17: The recreated swiped image from the recovered planes (b) from the swiped image (a) are estimated wrongly due to occlusion due to the swipe.

part of the plane silhouette. If the recovered pixel matches more closely the colours in a plane at another depth, we assume it is not part of the plane silhouette.

To assess how closely the colour of the pixel matches the colour of the plane around it, we calculate the mean and sample covariance of the pixels in a window around the pixel:

$$\boldsymbol{\mu}_L(\mathbf{x}) = \frac{1}{N_L} \sum_{i=1}^{N_L} \tilde{\mathbf{v}}(\mathbf{x}_{L,i}) \tag{4.22}$$

where $\boldsymbol{\mu}_L(\mathbf{x})$ is the mean of the pixels, where $\phi(\mathbf{x})$ is negative, in a window around $\mathbf{x}$, N_L is the number of pixels, where $\phi(\mathbf{x})$ is negative, in a window around $\mathbf{x}$ and $\tilde{v}(\mathbf{x}_{L,i})$ is the ith pixel, where $\phi(\mathbf{x})$ is negative, in a window around the pixel $\mathbf{x}$. The sample covariance for each label is calculated by [41]:

$$\mathbf{\Sigma}_L(\mathbf{x}) = \frac{1}{N_L - 1}\sum_{i=1}^{N_L}(\tilde{\mathbf{v}}(\mathbf{x}_{L,i}) - \boldsymbol{\mu}_L(\mathbf{x}))(\tilde{\mathbf{v}}(\mathbf{x}_{L,i}) - \boldsymbol{\mu}_L(\mathbf{x}))^T \tag{4.23}$$

where $\mathbf{\Sigma}_L(\mathbf{x})$ is the covariance of the pixels, where $\phi(\mathbf{x})$ is negative, in the window around $\mathbf{x}$. We then calculate the likelihood the pixel at $\mathbf{x}$ matches the distribution:

$$M_L(\mathbf{x}) = \frac{1}{(2\pi|\mathbf{\Sigma}_L(\mathbf{x})|)^{\frac{1}{2}}} e^{-\frac{1}{2}(\tilde{\mathbf{v}}(\mathbf{x})-\boldsymbol{\mu}_L(\mathbf{x}))^T(\mathbf{\Sigma}_L(\mathbf{x}))^{-1}(\tilde{\mathbf{v}}(\mathbf{x})-\boldsymbol{\mu}_L(\mathbf{x}))} \tag{4.24}$$

where $M_L(\mathbf{x})$ is the likelihood the pixel at $\mathbf{x}$ matches distribution defined by $\mathbf{\Sigma}_L(\mathbf{x})$ and $\boldsymbol{\mu}_L(\mathbf{x})$. The energy at the pixel is defined as:

$$E_L(\mathbf{x}) = -\ln(M_L(\mathbf{x})). \tag{4.25}$$

We aim to minimise the energy at each pixel to provide the optimal segmentation, with this terminology derived from commonly used energy minimisation techniques [14, 13].

To find the likelihood the pixel matches the planes at other depths, we consider the recovered planes as being composed of the recovered plane at each depth, as in (4.18). We repeat (4.22),(4.23), (4.24) and (4.25) for a window with the same coordinates as $\mathbf{x}_{L_i}$ for each depth $\tilde{\mathbf{v}}_1$, $\tilde{\mathbf{v}}_2$ ... $\tilde{\mathbf{v}}_P$ to obtain E_{L_1}, E_{L_2} ... E_{L_P}, a measure of the energy of the pixel at $\mathbf{x}$ against the pixels in a window at each depth.

To evolve the level set function we calculate the change in energy as ϕ changes is:

$$\frac{\partial E(\mathbf{x})}{\partial \phi(\mathbf{x})} = -\frac{(E_{L,p=d} - \min(E_{L,p\neq d}))}{|\phi(\mathbf{x})|} \tag{4.26}$$

where d is the depth that pixel $\mathbf{x}$ belongs to in $\phi(\mathbf{x})$, $E_{L,p=d}$ is the energy from the depth that $\mathbf{x}$ belongs to and $\min(E_{L,p\neq d})$ is the minimum energy from a depth that $\mathbf{x}$ does not belong. To calculate (4.26), for each pixel $\phi(\mathbf{x})$ where the sign is positive we must recalculate the $\mathbf{A}$ matrix and therefore the recovered planes for when the sign of $\phi(\mathbf{x})$ is negative to compare the colours of the recovered pixel.

As in Section 4.3.2, ϕ is evolved iteratively using a steepest descent algorithm to find plane silhouettes and therefore plane surfaces. We initialise ϕ using the plane silhouettes in Section 4.3.2. After each iteration $\mathbf{A}$ is recalculated based on ϕ and the recovered planes are re-estimated.

Using (4.26) for each pixel relies on having a good initial estimate of plane boundaries, so the plane silhouettes must be close to the correct answer before this procedure is applied. Furthermore, a different $\tilde{\mathbf{v}}$ needs to be recovered for each pixel within each depth, creating a high amount of complexity. The fast marching method is used to limit the pixels tested to near the boundaries, limiting the speed of the boundary change.

4.3.5 EPI recovery

Once the surface intensities of the planes have been obtained, this information is combined with the knowledge of the plane silhouettes and plane geometry from the switchpoints to reconstruct the EPI of the scene. To render an image for a novel camera position, we select the new camera distance from x_{02}, and use (4.3) to find the shift of the location of the planes within the new image. Once the planes have been shifted, we create the new image by copying the pixels where $\phi(\mathbf{x})$ is negative from each depth layer into the corresponding pixels of the new image. This process repeats for each depth, beginning with the furthest plane, to account for occlusions. The full EPI recovery algorithm is summarised in Algorithm 1.

Algorithm 1 Compute EPI from swiped image

1: Find switchpoints in swiped image
2: Calculate horizontal autocorrelation of image to estimate number of layers and layer depths
3: Transform pixels in swiped image to probabilities for each depth
4: Initialise plane silhouettes
5: **for** Iterations less than threshold **do**
6: Calculate $\frac{dE}{d\phi}$ using depth probabilities and evolve plane silhouettes using extension to level set method
7: **end for**
8: **for** Iterations less than threshold **do**
9: Evolve plane silhouettes using recovered plane surfaces
10: **end for**
11: Recreate EPI using scene geometry from depths, plane positions and plane surfaces

4.4 Conclusion

In conclusion, this chapter has presented a method to retrieve the EPI from a single swiped image. The method is derived from a depth-based layer model of the scene, in which a real scene is represented as a finite number of planes at different depths. Using this model, we are able to identify switchpoint locations in the swiped image, caused by planes occluding each other during the camera swipe. By finding the peaks in the autocorrelation of the image of switchpoints, combined with an existing method of generating a depth likelihood for each pixel, the depths of the layers in the scene are estimated.

To determine the silhouettes of each plane, we model the likelihood that a pixel is at a depth as a function of its RGB value. A variation of the level set method is employed to segment the likelihood for each depth. The variation on the level set method finds the plane silhouettes by accounting for the level of blur present within each depth layer. The silhouette for each depth is segmented independently.

The next step is to recover the plane surfaces from the silhouettes of each plane, the depth layers and the original swiped image. This is achieved by defining a matrix, $\mathbf{A}$, that when multiplied by a row of the sharp images of the planes at each depth will produce a row of the swiped image of the plane. We estimated the sharp images of planes at each depth using an iterative shrinkage algorithm (ISTA) to solve the matrix equation with $\mathbf{A}$ and the swiped image.

The final step before reconstructing the EPI is to optimise the segmentation of the silhouettes of the recovered planes by changing the silhouettes of the planes to maximise the colour separation between the layers.

Chapter 5

Evaluation Results

5.1 Introduction

To evaluate the algorithm proposed in Chapter 4, it was tested on both simulated and real images. The proposed algorithm was tested on scenes with fronto-parallel planes with realistic texture surfaces. The algorithm was able to correctly estimate the depths, segment the plane silhouettes and recover plane surfaces, leading to the recovery of the EPI for multiple scenes containing two and three planes.

The chapter is organised as follows: Section 5.2 describes the setup for testing the algorithm to recover the EPI. Section 5.3 evaluates the results of using the algorithm. Finally we conclude in Section 5.5.

5.2 Testing setup

To evaluate the algorithm proposed in Chapter 4, we tested it on both simulated and real images. To create real swiped images, fronto-parallel images at different depth layers were arranged to form a diorama that resembles a real life scene. A CineMoco camera slider was used to create a swiped image by moving a DSLR camera between two points with the shutter



Figure 5.1: Swiped images were captured using camera mounted on a motorised camera slider, with the scene created from printed fronto-parallel planes arranged to form a diorama.

open, as illustrated by Figure 5.1.

The number of depth layers was calculated as in Section 4.3.1. The parameter D_t was chosen empirically as $D_t = 0.7$. If the value is too high then too few or zero depths are detected. If the parameter is set too low then spurious depths are detected. This is illustrated by Figure 5.2, where a swiped image in Figure 5.2(a) produces the depth probability histogram in Figure 5.2(b). The number of depths detected from the image varies with the value of D_t, as shown in Figure 5.2(c). The boundary of the planes in the swiped image was segmented using a ϕ defined by the method proposed in Section 4.3.2, as illustrated by Figure 5.3. The level set method is initialised with multiple segments, as shown in Figure 5.3(a), in which the initial boundary contains a set percentage of the energy within each depth. The boundary is then evolved using the level set method, resulting in the plane boundary shown in Figure 5.3(b).

The run time of the algorithm on MATLAB running on a computer with an Intel i7-3770 CPU and 8GB of RAM when two depth layers were present was approximately 6 minutes for a swiped

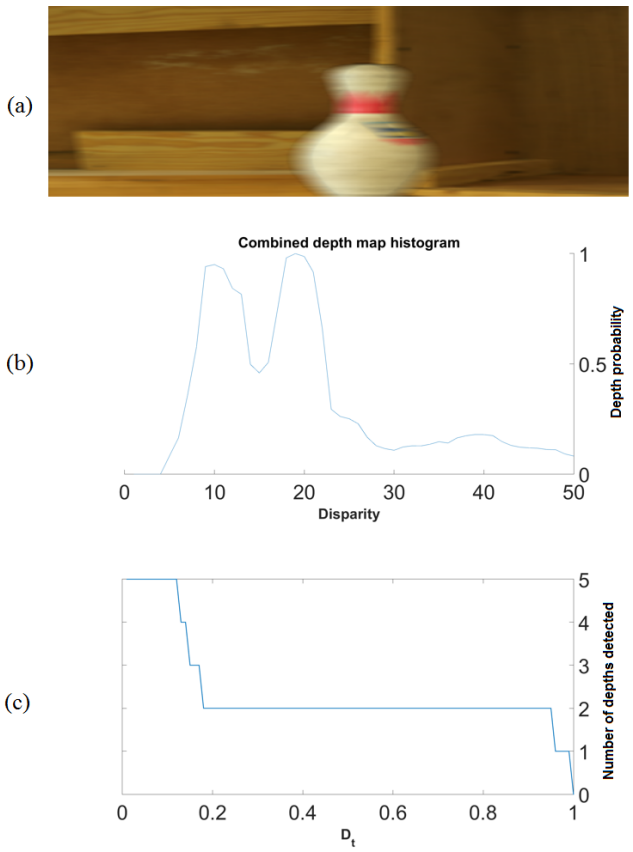


Figure 5.2: (a) shows a swiped image with two layers. (b) shows the depth probability at each disparity using the technique in Section 4.3.1. (c) shows how the number of depth layers detected varies with D_t for this image. The value of D_t is set empirically, as if it is too high then only a single layer is detected, and if it is too low then spurious depths are detected.

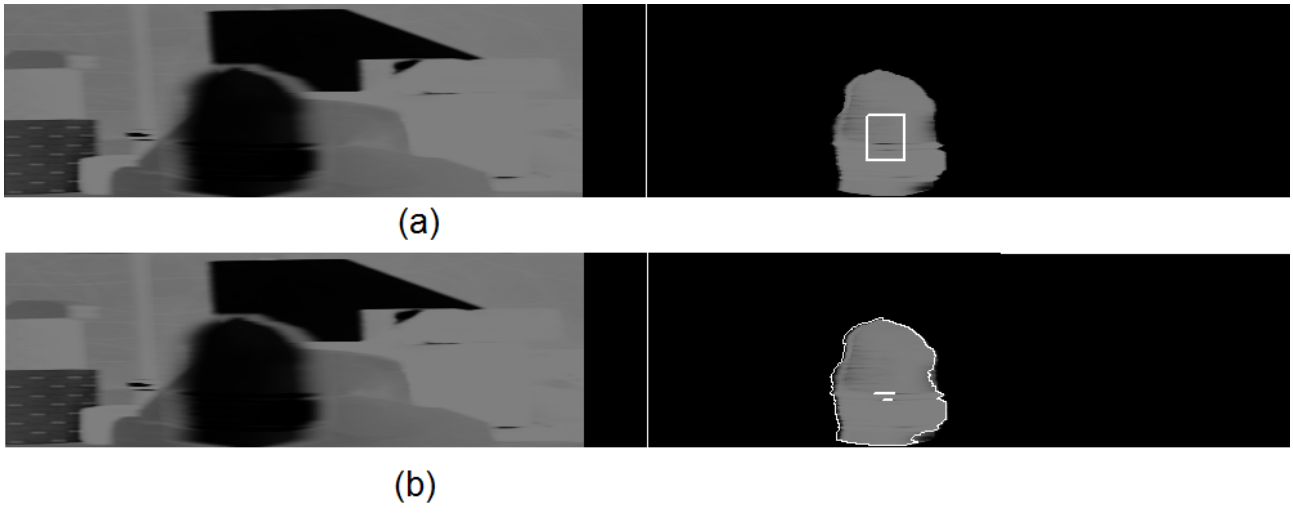


Figure 5.3: (a) shows the initial guess of Γ as a white rectangle, and (b) shows the Γ for level set method extension after 20 iterations. The leftmost plane is a background plane, so ϕ is always negative in that area.

image of dimension 450 by150 pixels [50]. Most of the complexity of the algorithm comes from the optimisation step described in Section 4.3.4, as the plane surfaces have to be estimated for each change in the sign of $\phi(\mathbf{x})$. As the optimisation step is run on each depth layer, the complexity of the algorithm is proportional to $O(P)$ [50].

5.3 Algorithm results

The depth maps that we estimate using the proposed technique are shown in Figure 5.4. Figure 5.4(a) shows the swiped images, with Figure 5.4(b) showing the ground truth depth map from the mid-point of the camera swipe. Figures 5.4(c) and (d) compare the results from [17] with the depth map resulting from the mid-point in the camera swipe of the recovered EPI. As the technique in Chapter 4 uses the algorithm from [17] to obtain an initial estimate of the depth in the image, the improved depth map from our technique justifies the iterative updates of the depth silhouettes using colour information [50].

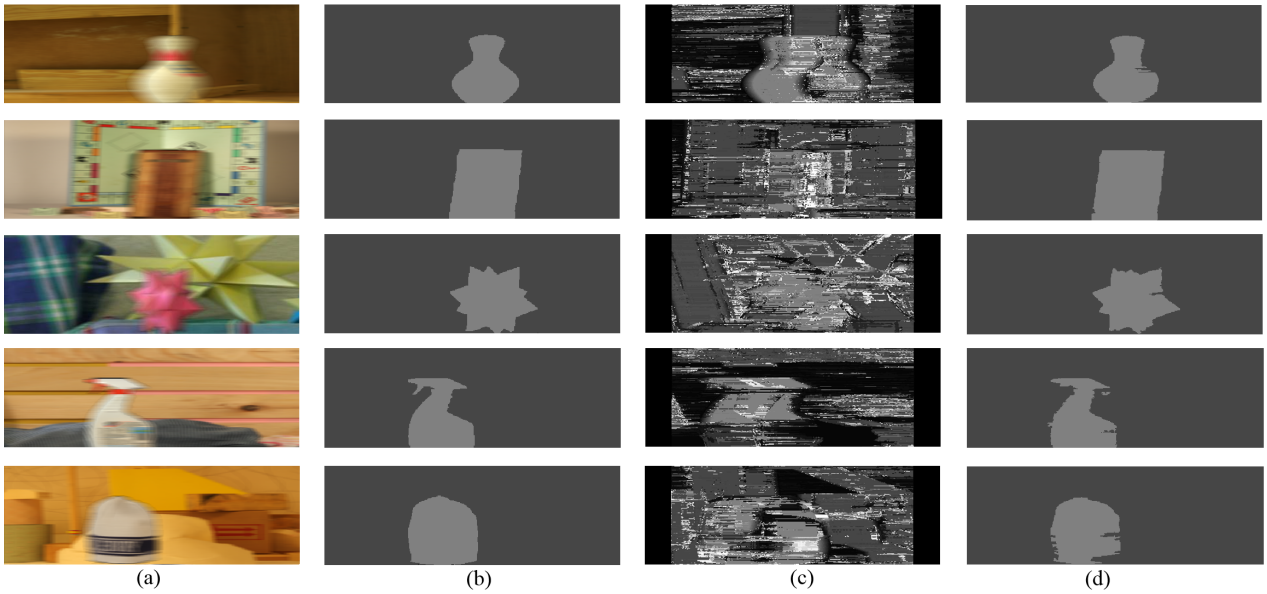


Figure 5.4: (a) are simulated swiped images, which are made from two planes at different depths to form a diorama, constructed from cropped planes in [72, 38]. (b) are the ground truth depth maps, from the mid-point of the camera swipe. (c) are the estimated depth maps using the technique from [17], and (d) are the estimated depth maps using the technique in this thesis at the mid-point of the camera swipe as a comparison of techniques (taken from [50]).

Given a swiped image (e.g. Figure 5.5(a)), the depths in the image can be detected using the

switchpoints (Figure 5.5(b)). The plane surfaces can then be recovered (Figure 5.5(c)), allowing the EPI to be recovered, and new images of the scene generated for arbitrary intermediate camera positions (Figure 5.5(d) and (e)) that lie within the original swipe.

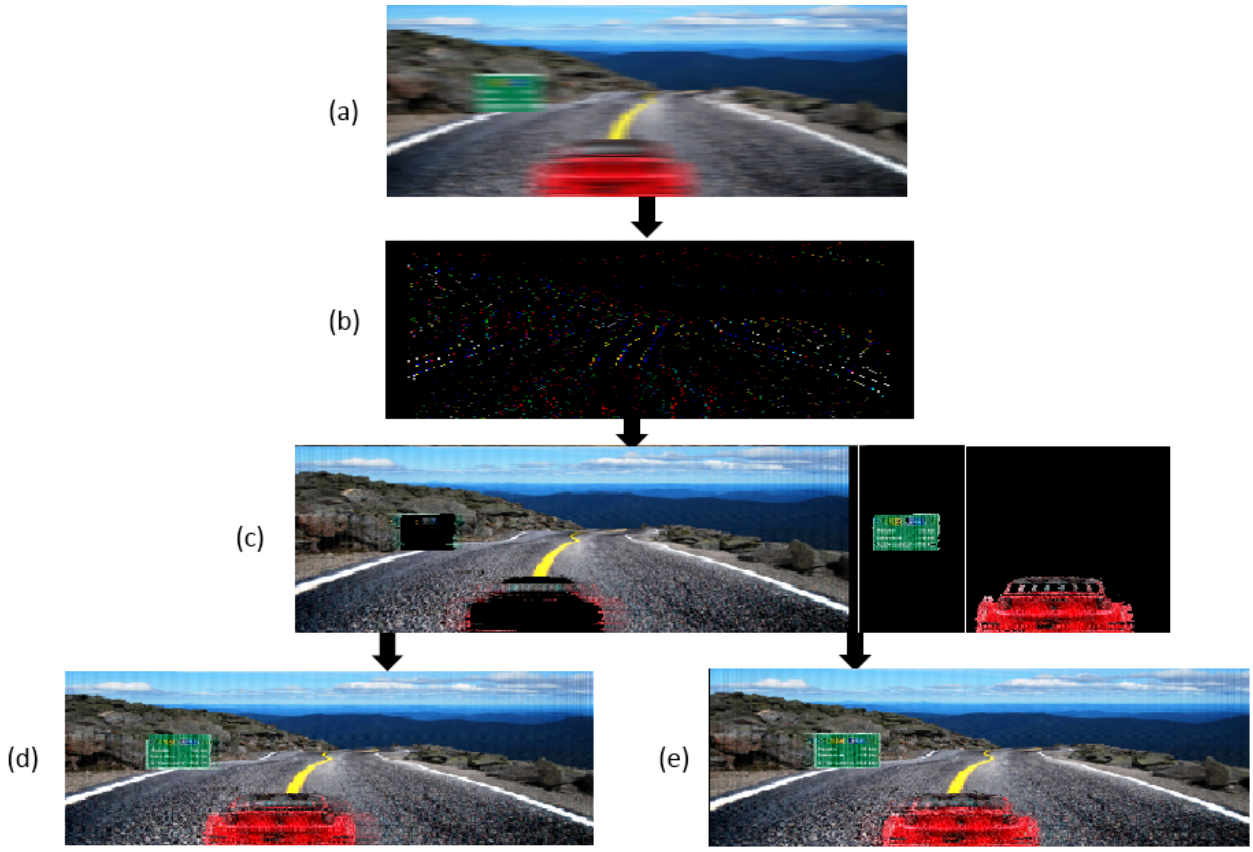


Figure 5.5: Creation of new views of the scene from a generated swiped image. (a) shows a swiped image of a scene, (b) shows an image of the detected switchpoints, (c) shows the recovered planes from furthest on the left to closest on the right, with each depth separated by a white line, and (d) and (e) show new views of the scene rendered from different perspectives.

In the recovered planes, areas which are always occluded during the swipe cannot be recovered as they are never seen. This can be seen in Figure 5.5(c), where those parts of the background plane that are never visible in Figure 5.5(a) cannot be recovered, leading to a hole in the recovered background plane. These holes are smaller in width than the objects occluding them due to the motion of planes during the swipe.

The results of the algorithm on a dataset of simulated images are shown in Figure 5.6, and with real images in Figure 5.7, demonstrating the validity of the algorithm under a variety of scenes. In each case the figure includes (a) the original swiped image, (b) the layer silhouettes, (c) the

extracted layers and (d) a resynthesized image. The recovered EPI from the upper example of Figure 5.6 is shown in Figure 5.8, which includes synthesized images from four different camera viewpoints.

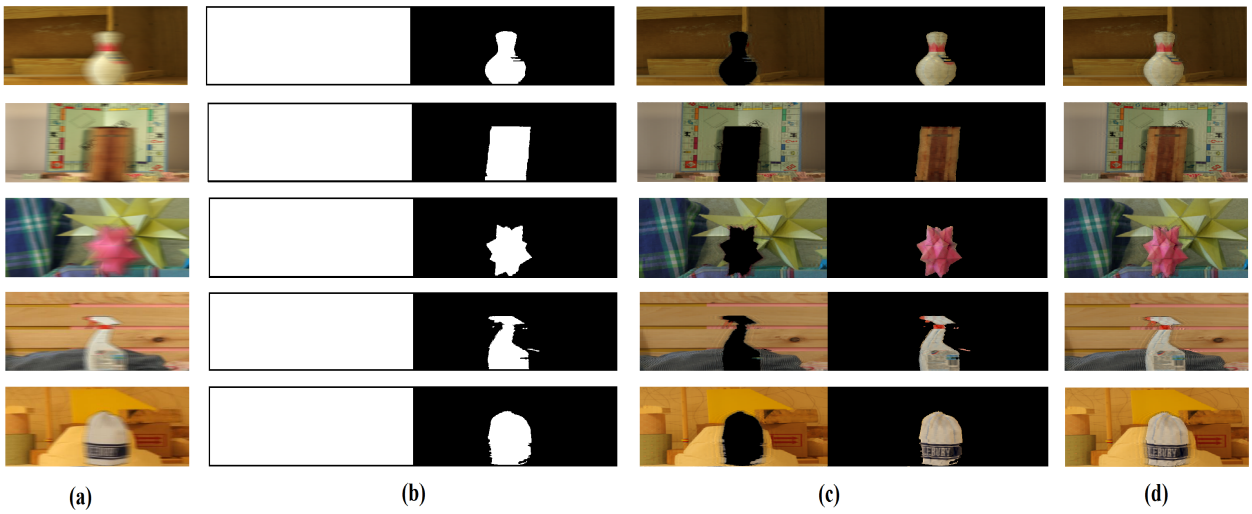


Figure 5.6: Column (a) are simulated swiped images, which are made from two planes at different depths, constructed from cropped planes in [72, 38]. Column (b) is the estimated plane silhouettes, in descending depth order, with column (c) being the recovered plane surfaces. From the recovered planes, we recover the EPIs of the scenes, and so can create novel views of the scene for different camera position, such as in column (d).



Figure 5.7: Column (a) are real swiped images, which are made from two planes at different depths to form a diorama, constructed from cropped planes in [72, 38]. Column (b) is the estimated plane silhouettes, in descending depth order, with column (c) being the recovered plane surfaces. From the recovered planes, we recover the EPIs of the scenes, and so can create novel views of the scene for different camera position, such as in column (d).

A further set of recovered EPIs can be seen in Figure 5.9 where there are three planes present within the scene. The scenes shown in Figure 5.9 have 3 depths present within the image. Some of the pixels in the final example in Figure 5.9 include contributions from all three planes. This

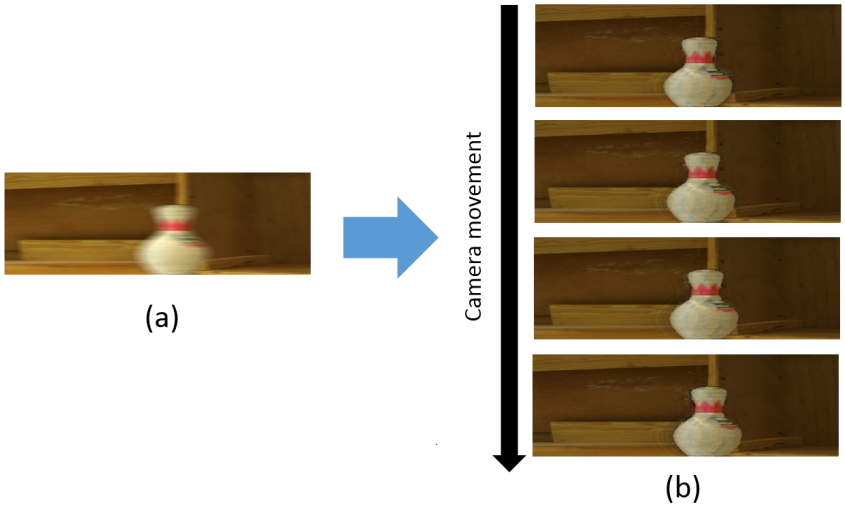


Figure 5.8: The EPI in (b) is recovered from the swiped image in (a) using the proposed algorithm from Section 4.3.

can be seen from the leftmost panel of image Figure 5.9(c) which includes recovered background pixels that lie between the two foreground objects. This would not be possible using the traditional methods of segmenting and deblurring by depth. The results of the algorithm on a simulated swiped image with four depths is shown in Figure 5.10. Simulated images are used in this case, as it is challenging to get all the planes within focus for a real image with four depth layers, whilst having enough of a difference in disparity to enable detection of each layer [50].

This demonstrates that the technique can be extended to more complex scenes that include multiple objects, as would be typically seen in real world swiped photographs.

To provide a metric to assess the accuracy of the EPI recovery, sharp images of scenes were taken in addition to swiped images. These were used as ground truth data to compare against the results of the recovered EPI from the swiped images. The structural similarity (SSIM) index [90] was used to compare the recovered images in the EPI, and the recovered swiped image from the created EPI, to the ground truth images, for swiped images created in MATLAB. The ground-truth EPI is available for synthetic images created in MATLAB as the integration of the EPI is used to create the synthetic swiped image. SSIM is chosen as a measurement of the perceived quality of the recovered EPI, as it compares the structure of two images, meaning wrongly estimated plane silhouettes will result in lower SSIM scores than correctly estimated

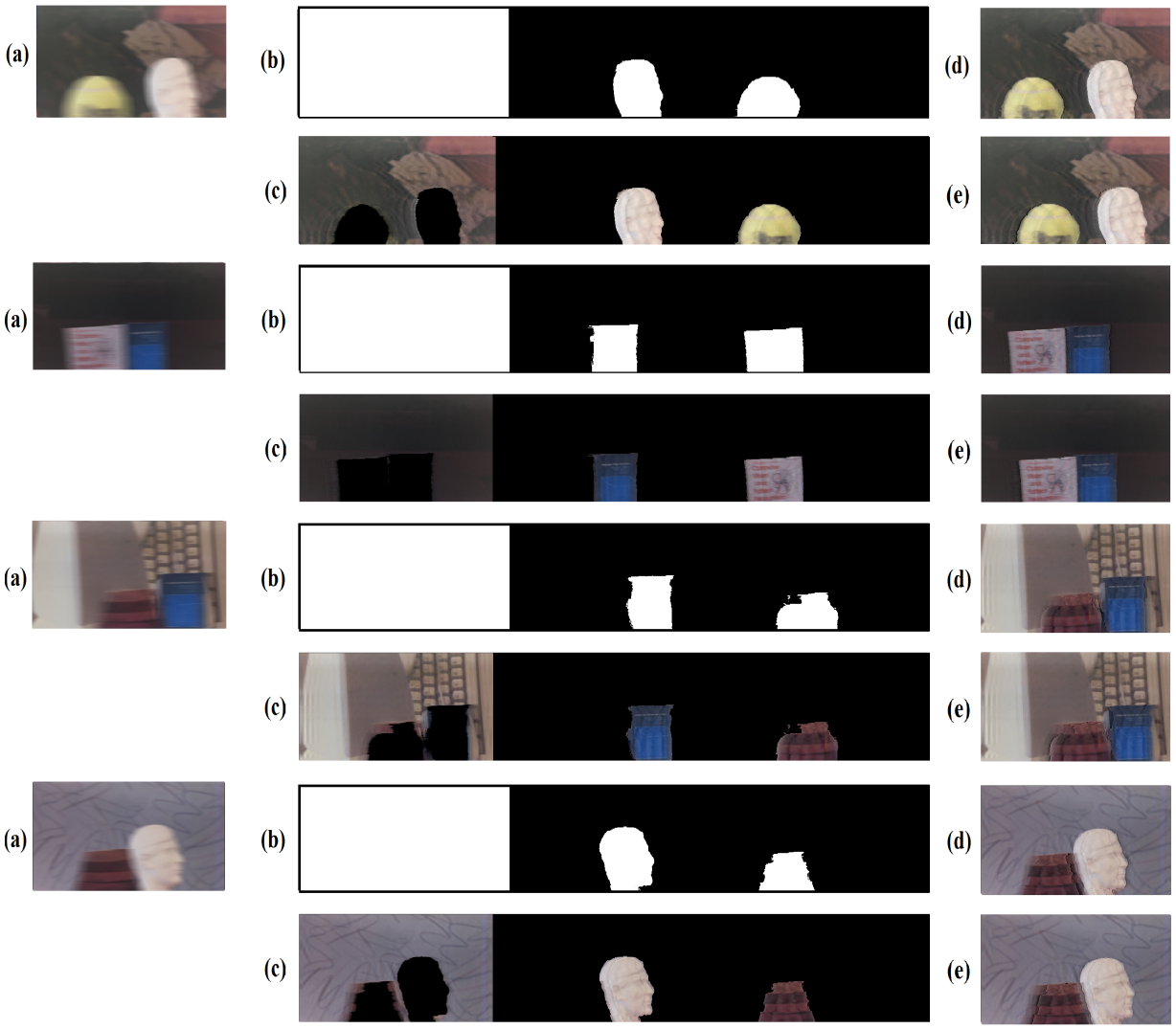


Figure 5.9: (a) are real swiped images, which are made from three planes at different depths to form a diorama, constructed from cropped planes in [72, 38]. (b) are the estimated plane silhouettes, in descending depth order, with (c) being the recovered plane surfaces. From the recovered planes, we recover the EPIs of the scenes, and so can create novel views of the scene for different camera position, such as in (d) and (e) (taken from [50]).

ones. The results of the proposed EPI recovery algorithm are shown in Table 5.1. The central column gives the SSIM of the recovered EPI while the rightmost column gives the SSIM of a swiped image recreated from the recovered EPI. To create the reconstructed swiped image, the recovered EPI was integrated between x_{01} and x_{02}. It can be seen that, although there are small errors in the recovered EPI, the recreated swiped image is, in all cases, almost perfect.

Table 5.1 shows results for different swipe lengths corresponding to maximum disparities of 10, 20 or 30 pixels. The results of this are plotted in Figure 5.11. From the simulations performed,

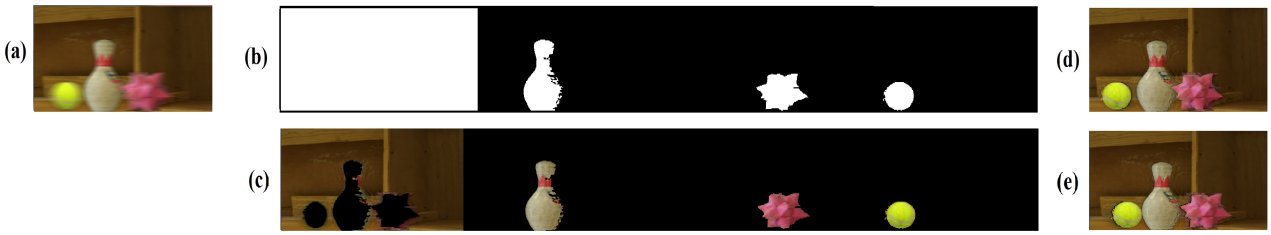


Figure 5.10: (a) are simulated swiped images containing four planes, which are made from four planes at different depths to form a diorama, constructed from cropped planes in [72, 38]. (b) are the estimated plane silhouettes, in descending depth order, with (c) being the recovered plane surfaces. From the recovered planes, we recover the EPIs of the scenes, and so can create novel views of the scene for different camera position, such as in (d) and (e) (taken from [50]).

Swiped image	Layers disparity	Recovered EPI SSIM	Recovered swiped image SSIM
Pin	5 10	0.993	1.000
Pin	10 20	0.988	1.000
Pin	15 30	0.985	1.000
Monopoly	5 10	0.982	1.000
Monopoly	10 20	0.969	1.000
Monopoly	15 30	0.958	1.000
Moebius	5 10	0.975	1.000
Moebius	10 20	0.957	1.000
Moebius	15 30	0.939	1.000
Laundry	5 10	0.964	0.998
Laundry	10 20	0.904	0.993
Laundry	15 30	0.882	0.995
Lampshade	5 10	0.982	1.000
Lampshade	10 20	0.970	0.999
Lampshade	15 30	0.959	1.000

Table 5.1: Comparison of recovered images of the scene against ground truth images.

it can be seen that there is a trade-off between having a larger camera swipe, which enables an EPI which encompasses more camera locations, and having an EPI with more accuracy. This makes intuitive sense, since with all deblurring algorithms, the larger the amount of blur the harder it is to recover a sharp image [50].

Table 5.1 demonstrates that the swiped image can be almost perfectly reconstructed by integrating the recovered EPI, even from an EPI that is not perfect, linking back to Section 3.5. In Section 3.5, it was demonstrated that different EPIs can, under some circumstances, result in identical swiped images. The scenes which lead to ambiguous swiped images in Section 3.5

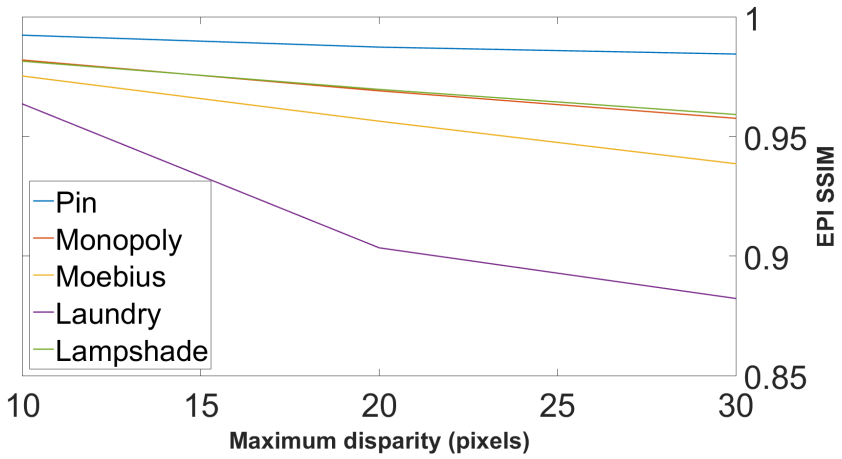


Figure 5.11: A graph plotting the SSIM of the recovered EPI against the ground truth EPI as the maximum disparity in the swiped image is varied (taken from [50]).

require exact alignment of switchpoints and planes, making it unlikely that any one scene will lead to an ambiguous EPI. In contrast, all swiped images in real life scenes can potentially be created from a range of scenes. The low pass filtering effect of the swiping of the camera leads to the smoothing of the plane textures, which does not affect the recovered planes for uniform texture surfaces, as the uniform plane textures are already smooth. Out of the multiple possible scenes that lead to a swiped image, the solution is chosen by use of the optimisation of plane silhouettes by colour separation in Section 4.3.4 and by calculating plane surfaces using an Iterative Shrinkage Thresholding Algorithm to minimise the norm of the solution in Section 4.3.3.

Figure 5.12 compares the proposed method with a recent blind deblurring algorithm [94]. Although [94] creates a view of the scene from only a single perspective, as shown in Figure 5.12(d), it can be compared with a rendered image from the recovered EPI using the technique from this thesis, as shown in Figure 5.12(b) as well as with the ground truth in Figure 5.12(c), with a quantitative assessment included in Table 5.2. An attempt to create an EPI from a single deblurred image would have the disadvantage against the proposed technique of not recovering plane surfaces which are occluded during part of the swipe and introducing holes in the rendered image as the planes are shifted. From the comparison, it can be seen that the technique from this thesis performs well against the blind deblurring algorithm, especially at the boundaries between planes, where there are more artifacts present in the deblurred images [50].



Figure 5.12: Column (a) are simulated swiped images, which are made from two planes at different depths to form a diorama, constructed from cropped planes in [72, 38]. Column (b) are the images from (a) deblurred using the software from [39], which implements the technique from [94]. Column (c) are the newly rendered images from the EPI estimated from (a) using the techniques outlined in this thesis.

Swiped image	Recovered image from EPI SSIM	blind deblurring SSIM
Pin	0.988	0.921
Monopoly	0.972	0.777
Moebius	0.961	0.846
Laundry	0.912	0.639
Lampshade	0.972	0.929

Table 5.2: Comparison of the SSIM results when comparing the ground truth EPI mid-point image against the recovered EPI mid-point image and against a blind deblurring algorithm [94].

5.4 Algorithm error resilience

Additional experiments were performed to find resilience of the algorithm to errors in the estimation of scene parameters. The effect that error in the estimate of depths of layers had on the quality of the reconstructed EPI was tested. Using simulated images in MATLAB, the technique from Chapter 4 was applied, using ground truth depths and plane silhouettes. The detected depth of the foreground plane was varied from the true value, and the resulting change in the accuracy of the reconstructed EPI was measured by using SSIM. The results are shown in Table 5.3, and are plotted in Figure 5.13. Figure 5.13 illustrates that the recovered EPI image quality gradually decreased as the detected depth diverges from the true depth.

Estimated foreground plane disparity	Recovered EPI SSIM	Recovered swiped image SSIM
15	0.980	1.00
16	0.981	1.00
17	0.983	1.00
18	0.986	1.00
19	0.987	1.00
20	0.988	1.00
21	0.987	1.00
22	0.985	1.00
23	0.982	1.00
24	0.980	1.00
25	0.977	1.00

Table 5.3: Comparison of recovered images of the scene against ground truth images. The true disparity of the foreground plane is 20 pixels.

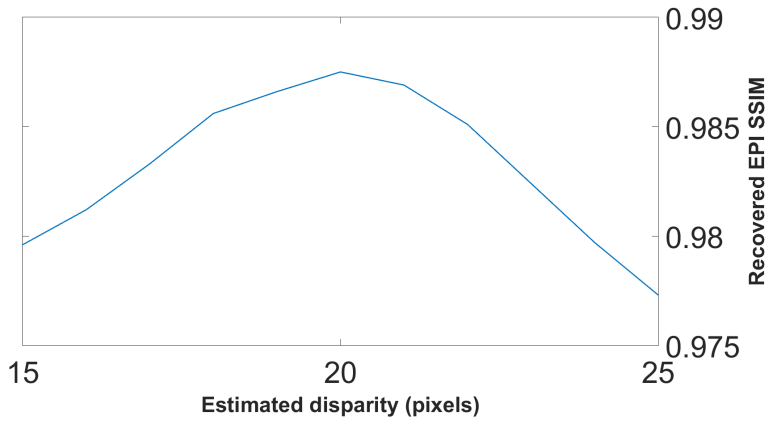


Figure 5.13: A plot of recovered EPI SSIM against detected disparity for the foreground plane, where the true disparity for the foreground plane is 20 pixels.

To find how robust the technique of Chapter 4 was to noise, experiments were performed using simulated swiped images with noise added. Additive White Gaussian Noise (AWGN) was used to model the effects of shot noise and quantum noise during the swiped image acquisition [57]. The results in Table 5.3 show that increasing the amount of noise in the swiped image in Figure 5.14 decreases the accuracy of the algorithm when comparing the results against the ground truth images. Figure 5.14 illustrates that as the Signal to Noise Ratio (SNR) of the swiped image is decreased, the SSIM of the recovered EPI falls slowly at first, and then more rapidly. The performance of the algorithm under these noise conditions is good, as the lower values of SNR tested would be much worse than could be achieved by a modern digital camera. Although the noise at lower SNR ratios causes some pixels to be misclassified, and for the pixels correctly labelled at a depth to differ from their true value, the RGB grouping technique of

Swiped image SNR	Recovered EPI SSIM	Recovered swiped image SSIM
70	0.987	1.00
60	0.987	1.00
50	0.984	1.00
40	0.924	0.998
30	0.552	0.964

Table 5.4: Comparison of recovered images of the scene against ground truth images as the swiped image SNR is decreased.

Section 4.3.2 proved to be robust to noise, leading to still accurate plane silhouettes.

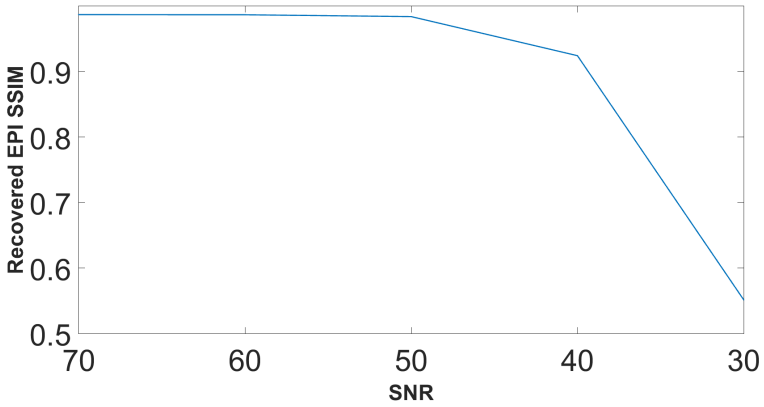


Figure 5.14: A plot of recovered EPI SSIM against SNR of the swiped image.

5.5 Conclusion

In conclusion, the algorithm to recover the EPI of a realistic scene from a swiped image has been tested on real and synthetic images, and the performance of the algorithm has been demonstrated. The swiped images for synthetic images were created using MATLAB, and for real images were created using printed planes and captured using a camera mounted on a motorised camera slider.

The recovered EPIs from testing were evaluated numerically using SSIM, and compared against a simulated EPI of the scene. Furthermore, the resynthesised swiped images created from the recovered EPI were compared to the original swiped images and were found to be an almost exact match.

Chapter 6

Conclusion

This chapter summarises the achievements of this thesis in Section 6.1, and discusses possible directions of future work on this subject matter in Section 6.2.

6.1 Summary of Thesis Achievements

The overall goal of the work described in this thesis is to recover the EPI of a scene from a single swiped image. This contrasts with existing methods of EPI recovery which require multiple images of the scene. Since the EPI encapsulates knowledge of all the light rays coming from the scene, its recovery permits the rendering of new views from arbitrary viewpoints.

Chapter 3 considered the simplified case of a scene containing one of more fronto-parallel planar objects having uniform texture. We presented an algorithm for the recovery of the EPI for a swiped image of such a scene. To achieve this we assign labels to the features in the swiped image that correspond to the planes that created them. When there is only a single plane present in the scene then the order of labels is pre-determined and the scene geometry and therefore the plane surface can be recovered directly. When there are multiple planes present, we used a combinatorial approach to identify the correct labelling. We take advantage of the knowledge of the scene geometry to eliminate plane labelling combinations that are not

physically possible, and recreate the swiped image of the scene to confirm the feasibility of the remaining possibilities. We have shown that it is possible, in rare situations, for identical swiped image to arise from more than one alternative scene. In cases where this occurs our algorithm consistently selects one of the possibilities.

We have further shown in Chapter 3 that it is possible to recover the EPI of a scene in which there is a single plane whose surface texture is a sum of sinusoids. Moreover, in cases of a poor resolution image acquired with a distorting sampling kernel, we showed that it is possible to use finite rate of innovation theory to recover the locations of the switchpoints to sub-pixel precision.

Chapter 4 presented an algorithm to recover the EPI from a swiped image of a scene containing fronto-parallel planes whose surface textures correspond to objects from a real world scene. In the proposed algorithm, the outlines of planes at different depths are segmented. An extension to the level set method which accounts for the amount of blur present within each depth layer is used to establish a silhouette of each plane at the start of the camera swipe. We have then designed a novel algorithm to recover the surface of the planes for each depth layer. Finally, we have optimised the outlines of the recovered planes by evolving the plane silhouettes of the recovered planes for each depth to achieve the maximum colour separation.

In Chapter 5, numerical results demonstrate the validity of the proposed EPI recovery algorithm. We have shown that the EPI that is recovered from the swiped image leads to an almost perfect reconstruction of the original swiped image, meaning that the solution identified by the proposed algorithm is consistent with the original data.

6.2 Future Work

In conclusion, we discuss in this section some possible directions for future work on this subject matter. Extensions to the work presented in this thesis could focus on relaxing the constraints on the acquisition of the swiped image, or alternatively techniques to improve the rendering of the new images.

6.2.1 Real Images

In Chapter 4 the use of the layer based approach from [29] and [69] was discussed for modelling a real scene as a series of fronto-parallel planes.



Figure 6.1: An example of a real image of a scene, where the EPI shown in (a) results in the swiped image shown in (b). Although it has been assumed in this thesis that the layer based model may be used, real scenes such as this have continuous variations in depth. Furthermore, it has been assumed that colour can be used to classify pixels by depth, whereas this scene has similar colours occurring at different depths.

An implicit assumption in Section 4.3.2 was that each layer has a distribution of RGB intensities so that it is possible to classify pixel depths on the basis of colour and intensity. The assumption of global colour consistency throughout a layer may be unrealistic and may lead to incorrect segmentation when a single layer contains objects having very different colours, as illustrated by the swiped image in Figure 6.1(b), whose corresponding EPI in Figure 6.1(a) contains similar colours at different depth layers. There are two ways in which this assumption can be modified

in order to improve the region segmentation accuracy. The first is to assume only local colour consistency within a layer so that a single layer will have different RGB intensity distributions in different regions of the image. The second modification is to incorporate additional features, such as texture features, into the distribution that characterizes a layer; this would allow adjacent objects to be distinguished when they are at different depths even if they have very similar colours.

6.2.2 Acquisition constraints

In this thesis, the problem of rendering new views of a scene when it is captured as a swiped image is analysed with a number of constraints. The constraints that are imposed upon the acquisition are that, while the shutter is open, the camera moves in a straight line at a constant velocity and that the start and end locations of this line are exactly known. By relaxing these constraints, a more general algorithm for plenoptic function recovery might be created. The advantage of this would be that the swiped image could be captured more easily using a hand-held camera.

The framework for rendering new views of the scene in this thesis is that of the EPI. Many previous methods for creating new views of a scene have used the light field model, where the camera can move in two dimensions rather than being constricted to a line. The acquisition of swiped images obtained by moving a camera in two dimensions may be investigated in the future to enable reconstruction of a light field from a single images. Although work in this area promises more freedom in viewing the scene from novel camera positions, the movement of the camera becomes more complicated, adding more unknowns when going from a swiped image to a new rendered view. For a single exposure, an infinite number of camera movement possibilities are available. The ultimate goal of this technique would be to capture the light field of a scene from a person moving a camera freeform in a roughly circular motion.

Acquisition constraints could be further relaxed to enable the recovery of a scene when the camera moves in a curved plane or even a spherical motion with the shutter open, such as the

movement demonstrated in Figure 1.1. This would have the advantage of enabling the viewing of an object from a wide range of angles, letting the viewer "look behind" an object of interest.

6.2.3 Swiped image and depth fusion

When recovering the EPI from the swiped image in Chapter 4, the estimation of the depths and the silhouettes of the depth layers formed key steps in the algorithm. The estimation of the layers in the image is achieved in this thesis only using the swiped image, but an alternative approach is by using instead a depth sensing camera. This combination is often known as RGB-D camera, where a camera captures an RGB image together with per-pixel depth information [36]. Recently, consumer devices such as the Microsoft Kinect have become available to mainstream consumers with a low price point [35, 37]. These provide accurate and high resolution depth images, in contrast to depth cameras of the past, which were often highly expensive and only able to offer very low resolution images.

Use of a swiped depth map from an RGB-D camera would allow the fusion of the image based techniques with additional depth information to create new views of the scene. For traditional image based rendering methods, this is often referred to as Depth Image Based Rendering (DIBR), and is investigated in [26, 44]. Use of an externally generated depth map may increase the accuracy of the layer depth estimation, especially in areas of smooth scene textures where the cues used to estimate the depth are absent. However, the requirement of an additional depth camera may limit the applicability of any developments in this area, as it is still a specialised sensor not available on most consumer devices.

6.2.4 Fluttering shutter

In the current implementation of the recovery of the EPI from a swiped image, the camera moves between two points with the shutter open for the entire movement. This condition simplifies the mathematical representation of the swiped image, as it is the integration of the EPI between the start and end of the camera swipe. Previous work in this area, [71], has

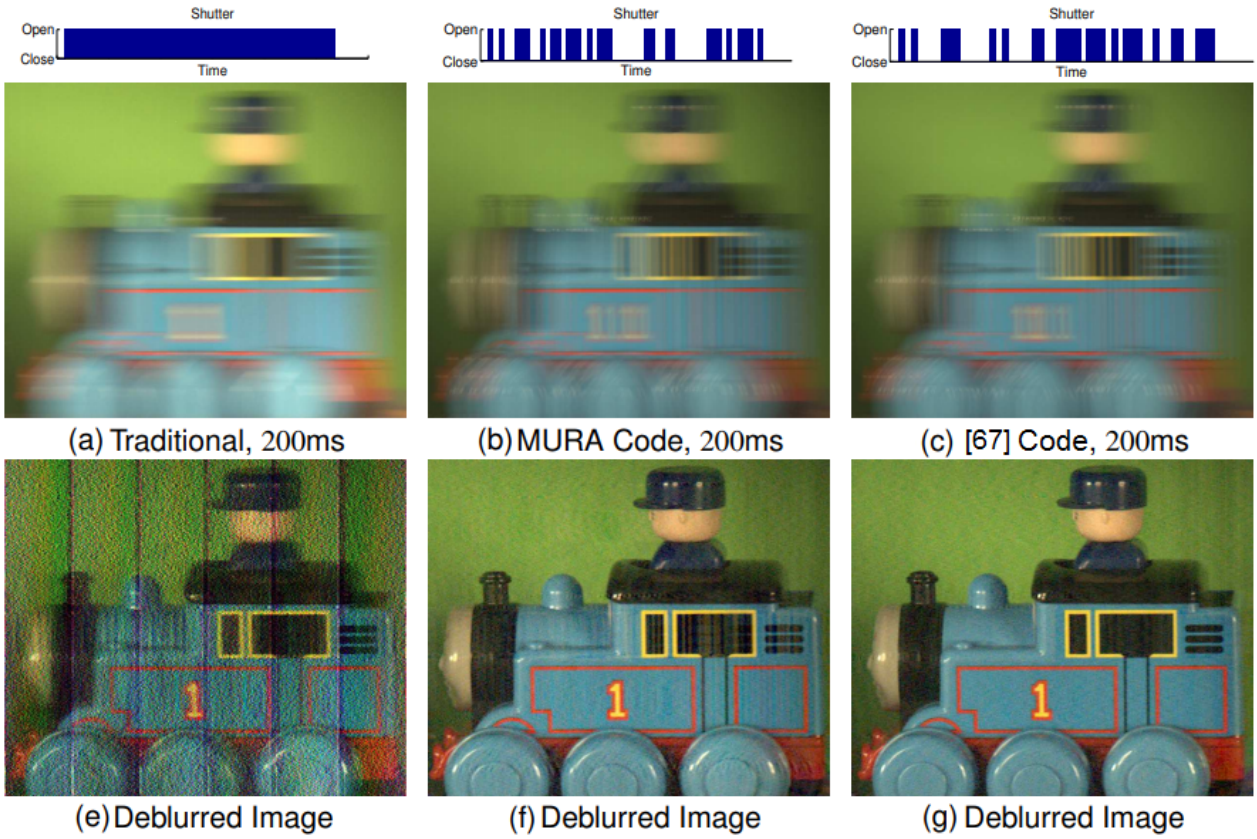


Figure 6.2: (a) is a swiped image of a scene, whilst (b) and (c) are coded exposures. (e) to (g) are the deblurred images of the swiped images above. (e) exhibits artifacts in the deblurred image, whereas (f) and (g) do not, as there are no zeros in the Fourier transform of the filter for the coded exposure (taken from [71]).

used a fluttering shutter, where the exposure is coded. The fluttering shutter enables the more accurate recovery of the sharp image. A low pass filter in the time domain has a sinc response in the frequency domain, with the null points of the sinc frequency response implying that the information at these frequencies cannot be recovered, even given perfect knowledge of the blurring kernel. In [71], the fluttering shutter uses a coded exposure pattern to take a series of exposures, which form a single blurred image. The image is deblurred using knowledge of the coded exposures to produce a single sharp image of the scene. The artifacts caused by deblurring the single exposure of Figure 6.2(a) are seen in Figure 6.2(e), whereas the coded exposures seen in Figure 6.2(b) and Figure 6.2(c) lead to deblurred images without artifacts in Figure 6.2(f) and Figure 6.2(g).

For future work, the swiped image may be obtained by a fluttering shutter, to reduce artifacts in the recovered plane surfaces. We currently reduce these artifacts by use of the ISTA algorithm,

but this comes at the trade-off of high frequency features in the recovered plane surfaces. To achieve this change, the **A** matrix from Section 4.3.3 would have to be altered to reflect the coded exposure. The advantage of the algorithms proposed in this thesis are that no specialised equipment needs to be used, as a swiped image can be generated with a consumer camera keeping the shutter open as it moves. This advantage is sustained for the case of a fluttering shutter, as the coded exposure only needs software changes, not hardware.

Bibliography

[1] D. Adalsteinsson and J. A. Sethian, "A fast level set method for propagating interfaces," *Journal of Computational Physics*, vol. 118, no. 2, pp. 269–277, 1995.

[2] E. H. Adelson and J. R. Bergen, *The plenoptic function and the elements of early vision.* Vision and Modeling Group, Media Laboratory, Massachusetts Institute of Technology, 1991.

[3] S. Baker, R. Szeliski, and P. Anandan, "A layered approach to stereo reconstruction," in *Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition (Cat. No.98CB36231)*, Jun. 1998, pp. 434–441.

[4] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[5] J. Berent and P. L. Dragotti, "Plenoptic manifolds," *IEEE Signal Processing Magazine*, vol. 24, no. 6, pp. 34–44, 2007.

[6] ——, "Segmentation of epipolar-plane image volumes with occlusion and disocclusion competition," in *IEEE Workshop on Multimedia Signal Processing.* IEEE, 2006, pp. 182–185.

[7] ——, "Unsupervised extraction of coherent regions for image based rendering." in *Proc. British Machine Vision Conf.*, 2007, pp. 1–10.

[8] J. Berent, P. L. Dragotti, and T. Blu, "Sampling piecewise sinusoidal signals with finite rate of innovation methods," *IEEE Transactions on Signal Processing*, vol. 58, no. 2, pp. 613–625, 2010.

[9] N. Bi, M. Z. Nashed, and Q. Sun, "Reconstructing signals with finite rate of innovation from noisy samples," *Acta applicandae mathematicae*, vol. 107, no. 1-3, pp. 339–372, 2009.

[10] B. Bideau, R. Kulpa, N. Vignais, S. Brault, F. Multon, and C. Craig, "Using virtual reality to analyze sports performance," *IEEE Computer Graphics and Applications*, vol. 30, no. 2, pp. 14–21, 2010.

[11] R. C. Bolles, H. H. Baker, and D. H. Marimont, "Epipolar-plane image analysis: An approach to determining structure from motion," *International Journal of Computer Vision*, vol. 1, no. 1, pp. 7–55, 1987.

[12] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," in *Proc. IEEE Int. Conf. Computer Vision*, vol. 1, Sep. 1999, pp. 377–384 vol.1.

[13] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 9, pp. 1124–1137, 2004.

[14] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 23, no. 11, pp. 1222–1239, 2001.

[15] C. Buehler, M. Bosse, L. McMillan, S. Gortler, and M. Cohen, "Unstructured lumigraph rendering," in *Proc. Conf. on Computer Graphics and Interactive Techniques*. ACM, 2001, pp. 425–432.

[16] J.-X. Chai, X. Tong, S.-C. Chan, and H.-Y. Shum, "Plenoptic sampling," in *Proc. Conf. on Computer Graphics and Interactive Techniques*. ACM Press/Addison-Wesley Publishing Co., 2000, pp. 307–318.

[17] A. Chakrabarti, T. Zickler, and W. T. Freeman, "Analyzing spatially-varying blur," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2010, pp. 2512–2519.

[18] A. Chambolle, R. A. De Vore, N.-Y. Lee, and B. J. Lucier, "Nonlinear wavelet image processing: variational problems, compression, and noise removal through wavelet shrinkage," *IEEE Transactions on Image Processing*, vol. 7, no. 3, pp. 319–335, 1998.

[19] C.-L. Chang, X. Zhu, P. Ramanathan, and B. Girod, "Light field compression using disparity-compensated lifting and shape adaptation," *IEEE Transactions on Image Processing*, vol. 15, no. 4, pp. 793–806, Apr. 2006.

[20] S. E. Chen and L. Williams, "View interpolation for image synthesis," in *Proc. Conf. on Computer Graphics and Interactive Techniques*. ACM, 1993, pp. 279–288.

[21] F. Couzinie-Devy, J. Sun, K. Alahari, and J. Ponce, "Learning to estimate and remove non-uniform image blur," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*. IEEE, 2013, pp. 1075–1082.

[22] A. Criminisi, S. B. Kang, R. Swaminathan, R. Szeliski, and P. Anandan, "Extracting layers and analyzing their specular properties using epipolar-plane-image analysis," *Computer Vision and Image Understanding*, vol. 97, no. 1, pp. 51–85, 2005.

[23] S. Dai and Y. Wu, "Motion from blur," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*. IEEE, 2008, pp. 1–8.

[24] M. N. Do, D. Marchand-Maillet, and M. Vetterli, "On the bandwidth of the plenoptic function," *IEEE Transactions on Image Processing*, vol. 21, no. 2, pp. 708–717, 2012.

[25] P. L. Dragotti, M. Vetterli, and T. Blu, "Sampling moments and reconstructing signals of finite rate of innovation: Shannon meets Strang-Fix," *IEEE Transactions on Signal Processing*, vol. 55, no. 5, pp. 1741–1757, 2007. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4156380

[26] C. Fehn, "Depth-image-based rendering (dibr), compression, and transmission for a new approach on 3d-tv," in *Stereoscopic Displays and Virtual Reality Systems XI*, vol. 5291. International Society for Optics and Photonics, 2004, pp. 93–105.

[27] Z.-F. Gan, S.-C. Chan, K.-T. Ng, and H.-Y. Shum, "Am object-based approach to plenoptic videos," in *Proc. IEEE Int. Symp. Circuits and Systems*, May 2005, pp. 3435–3438 Vol. 4.

[28] A. Gelman, J. Berent, and P. L. Dragotti, "Layer-based sparse representation of multiview images," *EURASIP Journal on Advances in Signal Processing*, vol. 2012, no. 1, pp. 1–15, 2012.

[29] A. Gelman, P. L. Dragotti, and V. Velisavljevic, "Multiview image coding using depth layers and an optimized bit allocation," *IEEE Transactions on Image Processing*, vol. 21, no. 9, pp. 4092–4105, 2012.

[30] C. Gilliam, P. L. Dragotti, and M. Brookes, "On the spectrum of the plenoptic function," *IEEE Transactions on Image Processing*, vol. 23, no. 2, pp. 502–516, 2014. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6672000

[31] D. Gledhill, G. Y. Tian, D. Taylor, and D. Clarke, "Panoramic imaginga review," *Computers & Graphics*, vol. 27, no. 3, pp. 435–445, 2003.

[32] I. Goradia, J. Doshi, and L. Kurup, "A review paper on oculus rift & project morpheus," *International Journal of Current Engineering and Technology*, vol. 4, no. 5, pp. 3196–3200, 2014.

[33] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen, "The lumigraph," in *Proc. Conf. on Computer Graphics and Interactive Techniques*. ACM, 1996, pp. 43–54.

[34] A. Gupta, N. Joshi, C. L. Zitnick, M. Cohen, and B. Curless, "Single image deblurring using motion density functions," in *Proc. European Conf. on Computer Vision*. Springer, 2010, pp. 171–184.

[35] J. Han, L. Shao, D. Xu, and J. Shotton, "Enhanced computer vision with microsoft kinect sensor: A review," *IEEE Transactions on Cybernetics*, vol. 43, no. 5, pp. 1318–1334, 2013.

[36] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "Rgb-d mapping: Using depth cameras for dense 3d modeling of indoor environments," in *Proc. Int. Symp. on Experimental Robotics (ISER)*. Citeseer, 2010.

[37] ——, "Rgb-d mapping: Using kinect-style depth cameras for dense 3d modeling of indoor environments," *The International Journal of Robotics Research*, vol. 31, no. 5, pp. 647–663, 2012.

[38] H. Hirschmuller and D. Scharstein, "Evaluation of cost functions for stereo matching," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Jun. 2007, pp. 1–8.

[39] J. JIA, "Robust deblurring software," http://www.cse.cuhk.edu.hk/ leojia/deblurring.htm, accessed: 2018-09-27.

[40] S.-A. A. Jin, "The roles of modality richness and involvement in shopping behavior in 3d virtual stores," *Journal of Interactive Marketing*, vol. 23, no. 3, pp. 234–246, 2009.

[41] R. Johnson and D. Wichern, *Applied Multivariate Statistical Analysis*, ser. Applied Multivariate Statistical Analysis. Pearson Prentice Hall, 2007.

[42] T. Kanade and M. Okutomi, "A stereo matching algorithm with an adaptive window: Theory and experiment," in *Proc. IEEE Int. Conf. on Robotics and Automation*. IEEE, 1991, pp. 1088–1095.

[43] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321–331, 1988.

[44] P. Kauff, N. Atzpadin, C. Fehn, M. Müller, O. Schreer, A. Smolic, and R. Tanger, "Depth map creation and image-based rendering for advanced 3dtv services providing interoperability and scalability," *Signal Processing: Image Communication*, vol. 22, no. 2, pp. 217–234, 2007.

[45] Q. Ke and T. Kanade, "A subspace approach to layer extraction," in *Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR)*, vol. 1, Dec. 2001, p. I.

[46] J. Konrad and M. Halle, "3-D displays and signal processing," *IEEE Signal Processing Magazine*, vol. 24, no. 6, pp. 97–111, Nov. 2007.

[47] W. Krzanowski, P. Jonathan, W. McCarthy, and M. Thomas, "Discriminant analysis with singular covariance matrices: methods and applications to spectroscopic data," *Applied statistics*, pp. 101–115, 1995.

[48] M. Lawson, M. Brookes, and P. L. Dragotti, "Capturing the plenoptic function in a swipe," in *SPIE Optical Engineering+ Applications*. International Society for Optics and Photonics, 2016, pp. 99 710O–99 710O.

[49] ——, "Identifying a multiple plane plenoptic function from a swiped image," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 1423–1427.

[50] ——, "Scene estimation from a swiped image," *IEEE Transactions on Computational Imaging (submitted)*, 2018.

[51] A. Levin, "Blind motion deblurring using image statistics," in *Advances in Neural Information Processing Systems*, 2007, pp. 841–848.

[52] M. Levoy and P. Hanrahan, "Light field rendering," in *Proc. Conf. on Computer Graphics and Interactive Techniques*. ACM, 1996, pp. 31–42.

[53] H. Lin, C. Chen, S. Bing Kang, and J. Yu, "Depth recovery from light field using focal stack symmetry," in *Proc. IEEE Int. Conf. on Computer Vision*, 2015, pp. 3451–3459.

[54] H.-Y. Lin and C.-H. Chang, "Depth from motion and defocus blur," *Optical engineering*, vol. 45, no. 12, p. 127201, 2006.

[55] M. H. Lin and C. Tomasi, "Surfaces with occlusions from layered stereo," in *Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition*, vol. 1, Jun. 2003, p. I.

[56] C. Liu and H. Wechsler, "Gabor feature based classification using the enhanced fisher linear discriminant model for face recognition," *IEEE Transactions on Image processing*, vol. 11, no. 4, pp. 467–476, 2002.

[57] J. Lukas, J. Fridrich, and M. Goljan, "Digital camera identification from sensor pattern noise," *IEEE Transactions on Information Forensics and Security*, vol. 1, no. 2, pp. 205–214, 2006.

[58] G. Luo, Y. Zhu, Z. Li, and L. Zhang, "A hole filling approach based on background reconstruction for view synthesis in 3d video," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2016, pp. 1781–1789.

[59] E. Maitre, T. Milcent, G.-H. Cottet, A. Raoult, and Y. Usson, "Applications of level set methods in computational biophysics," *Mathematical and Computer Modelling*, vol. 49, no. 11, pp. 2161–2169, 2009.

[60] L. McMillan and G. Bishop, "Plenoptic modeling: An image-based rendering system," in *Proc. Conf. on Computer Graphics and Interactive Techniques*. ACM, 1995, pp. 39–46.

[61] G. Medioni and R. Nevatia, "Segment-based stereo matching," *Computer Vision, Graphics, and Image Processing*, vol. 31, no. 1, pp. 2–18, 1985.

[62] S. Nayar and M. Ben-Ezra, "Motion-based motion deblurring," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 6, pp. 689–698, 2004.

[63] R. Ng, "Fourier slice photography," in *ACM Transactions on Graphics (TOG)*, vol. 24, no. 3. ACM, 2005, pp. 735–744.

[64] R. Ng, M. Levoy, M. Brédif, G. Duval, M. Horowitz, and P. Hanrahan, "Light field photography with a hand-held plenoptic camera," *Computer Science Technical Report (CSTR)*, vol. 2, no. 11, pp. 1–11, 2005.

[65] K.-J. Oh, S. Yea, and Y.-S. Ho, "Hole filling method using depth based in-painting for view synthesis in free viewpoint television and 3-d video," in *Picture Coding Symposium (PCS)*. IEEE, 2009, pp. 1–4.

[66] S. Osher and R. P. Fedkiw, "Level set methods: an overview and some recent results," *Journal of Computational Physics*, vol. 169, no. 2, pp. 463–502, 2001.

[67] S. Osher and J. A. Sethian, "Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations," *Journal of Computational Physics*, vol. 79, no. 1, pp. 12–49, 1988.

[68] C. Paramanand and A. N. Rajagopalan, "Depth from motion and optical blur with an unscented kalman filter," *IEEE Transactions on Image Processing*, vol. 21, no. 5, pp. 2798–2811, 2012.

[69] J. Pearson, M. Brookes, and P. L. Dragotti, "Plenoptic layer-based modeling for image based rendering," *IEEE Transactions on Image Processing*, vol. 22, no. 9, pp. 3405–3419, 2013.

[70] H. Pitsch, "A consistent level set formulation for large-eddy simulation of premixed turbulent combustion," *Combustion and Flame*, vol. 143, no. 4, pp. 587–598, 2005.

[71] R. Raskar, A. Agrawal, and J. Tumblin, "Coded exposure photography: motion deblurring using fluttered shutter," in *ACM Transactions on Graphics (TOG)*, vol. 25, no. 3. ACM, 2006, pp. 795–804.

[72] D. Scharstein and C. Pal, "Learning conditional random fields for stereo," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Jun. 2007, pp. 1–8.

[73] D. Scharstein and R. Szeliski, "High-accuracy stereo depth maps using structured light," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1. IEEE, 2003, pp. I–I.

[74] W. J. Schroeder, B. Lorensen, and K. Martin, *The visualization toolkit: an object-oriented approach to 3D graphics*. Kitware, 2004.

[75] Q. Shan, J. Jia, and A. Agarwala, "High-quality motion deblurring from a single image," in *ACM Transactions on Graphics (TOG)*, vol. 27, no. 3. ACM, 2008, p. 73.

[76] H. Shum and S. B. Kang, "Review of image-based rendering techniques," in *Visual Communications and Image Processing*. International Society for Optics and Photonics, 2000, pp. 2–13.

[77] H.-Y. Shum, J. Sun, S. Yamazaki, Y. Li, and C.-K. Tang, "Pop-up light field: An interactive image-based modeling and rendering system," *ACM Transactions on Graphics (TOG)*, vol. 23, no. 2, pp. 143–162, 2004.

[78] P. Smereka, "Semi-implicit level set methods for curvature and surface diffusion motion," *Journal of Scientific Computing*, vol. 19, no. 1, pp. 439–456, 2003.

[79] M. Sorel and J. Flusser, "Space-variant restoration of images degraded by camera motion blur," *IEEE Transactions on Image Processing*, vol. 17, no. 2, pp. 105–116, 2008.

[80] D. L. Swets and J. J. Weng, "Using discriminant eigenfeatures for image retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 831–836, 1996.

[81] K. Takahashi, "Theoretical analysis of view interpolation with inaccurate depth information," *IEEE Transactions on Image Processing*, vol. 21, no. 2, pp. 718–732, 2012.

[82] V. Y. Tan and V. K. Goyal, "Estimating signals with finite rate of innovation from noisy samples: A stochastic algorithm," *IEEE Transactions on Signal Processing*, vol. 56, no. 10, pp. 5135–5146, 2008.

[83] R. J. Teather and I. S. MacKenzie, "Position vs. velocity control for tilt-based interaction," in *Proceedings of Graphics Interface 2014*. Canadian Information Processing Society, 2014, pp. 51–58.

[84] R. Tsai, S. Osher *et al.*, "Review article: Level set methods and their applications in image science," *Communications in Mathematical Sciences*, vol. 1, no. 4, pp. 1–20, 2003.

[85] J. Unnikrishnan, S. Haghighatshoar, and M. Vetterli, "Unlabeled sensing: Solving a linear system with unordered measurements," in *Proc. Allerton Conf. on Communication, Control, and Computing*. IEEE, 2015, pp. 786–793.

[86] J. A. Uriguen, T. Blu, and P. L. Dragotti, "Fri sampling with arbitrary kernels," *IEEE Transactions on Signal Processing*, vol. 61, no. 21, pp. 5310–5323, 2013. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6578165

[87] M. Vetterli, P. Marziliano, and T. Blu, "Sampling signals with finite rate of innovation," *IEEE Transactions on Signal Processing*, vol. 50, no. 6, pp. 1417–1428, 2002. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1003065

[88] J. Y. A. Wang and E. H. Adelson, "Layered representation for motion analysis," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Jun. 1993, pp. 361–366.

[89] ——, "Representing moving images with layers," *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 625–638, Sep. 1994.

[90] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[91] S. Wanner and B. Goldluecke, "Globally consistent depth labeling of 4D light fields," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Jun. 2012, pp. 41–48.

[92] B. Wilburn, N. Joshi, V. Vaish, E.-V. Talvala, E. Antunez, A. Barth, A. Adams, M. Horowitz, and M. Levoy, "High performance imaging using large camera arrays," in *ACM Transactions on Graphics (TOG)*, vol. 24, no. 3. ACM, 2005, pp. 765–776.

[93] J. Xiao and M. Shah, "Motion layer extraction in the presence of occlusion using graph cuts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1644–1659, Oct. 2005.

[94] L. Xu, S. Zheng, and J. Jia, "Unnatural l0 Sparse representation for natural image deblurring," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Jun. 2013, pp. 1107–1114.

[95] C. Zhang and T. Chen, "Spectral analysis for sampling image-based rendering data," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 11, pp. 1038–1050, Nov. 2003.

[96] ——, "A survey on image-based rendering representation, sampling and compression," *Signal Processing: Image Communication*, vol. 19, no. 1, pp. 1–28, 2004.

[97] S. Zhuo and T. Sim, "On the recovery of depth from a single defocused image," in *Proc. Int. Conf. on Computer Analysis of Images and Patterns*. Springer, 2009, pp. 889–897.