

# **Analysis, Acquisition, and Processing of Light Field for Computational Photography**

*Chia-Kai Liang*

*June 2009*

*Graduate Institute of Communications Engineering*

*National Taiwan University*

*Taipei, Taiwan, R.O.C.*





# Contents

<b>Abstract</b>	<b>xiii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Overview . . . . .	2
1.2 Preliminary of Light Field . . . . .	4
<b>2 Light Transport in Photography</b>	<b>7</b>
2.1 Previous Work . . . . .	8
2.2 General Light Transport Operators . . . . .	10
2.2.1 Parameterization . . . . .	10
2.2.2 Light Ray Traveling . . . . .	12
2.2.3 Occlusion . . . . .	14
2.2.4 Refraction of the Lens . . . . .	16
2.2.5 Diffuse Shading . . . . .	16
2.2.6 Accumulation of Transformations . . . . .	19
2.3 Combined Transformation in Photography . . . . .	20
2.3.1 Derivation without Aperture . . . . .	20
2.3.2 Derivation with Aperture . . . . .	22
2.4 Interpretation of Photographic Effects . . . . .	25
2.4.1 Perspective Transformation . . . . .	25
2.4.2 Defocus Blur . . . . .	27
2.4.3 Vignetting Effect . . . . .	33
2.4.4 Summary . . . . .	34
2.5 Spectrum Analysis of the Transformed Light Field . . . . .	34

2.5.1	Refocusing using Light Field . . . . .	35
2.5.2	Fusion of All-Focused Images . . . . .	37
2.5.3	Parameter Settings for Efficient Light Field Sampling . . . . .	40
2.6	Depth Detection of Light Field . . . . .	44
2.6.1	Overview . . . . .	45
2.6.2	Focusness Measurement . . . . .	45
2.6.3	Detection as Optimization . . . . .	46
2.6.4	Experimental Results . . . . .	48
2.6.5	Complexity . . . . .	51
2.7	Discussion . . . . .	51
2.7.1	Extension to the 3D Space . . . . .	52
2.7.2	New Aperture Design . . . . .	53
2.7.3	New All-Focused Fusion Method . . . . .	53
2.8	Summary . . . . .	54
<b>3</b>	<b>Light Field Acquisition</b>	<b>55</b>
3.1	Previous Work . . . . .	55
3.1.1	Light Field Acquisition . . . . .	55
3.1.2	Coded Aperture Imaging . . . . .	57
3.1.3	Multiple-Exposure Imaging . . . . .	58
3.1.4	Illumination Multiplexing . . . . .	58
3.2	Programmable Aperture . . . . .	59
3.2.1	Sequential Light Field Acquisition . . . . .	59
3.2.2	Light Field Multiplexing . . . . .	60
3.2.3	Prototypes . . . . .	64
3.2.4	Performance Comparison . . . . .	69
3.2.5	Discussion . . . . .	70
3.2.6	Related Work . . . . .	71
3.2.7	Limitation and Future Direction . . . . .	72
3.3	Photometric Calibration . . . . .	73
3.3.1	Previous Methods . . . . .	74

3.3.2	Proposed Algorithm . . . . .	75
3.4	Multi-View Depth Estimation . . . . .	79
3.4.1	Previous Methods . . . . .	79
3.4.2	Main Concept . . . . .	80
3.4.3	Definitions of Energy Terms . . . . .	81
3.4.4	Optimization . . . . .	84
3.4.5	Performance Analysis . . . . .	85
3.4.6	Discussion . . . . .	88
3.5	Summary . . . . .	95
<b>4</b>	<b>Applications of the Light Field Cameras</b>	<b>97</b>
4.1	View Interpolation . . . . .	97
4.2	Digital Refocusing . . . . .	100
4.3	Feature-Based Refocusing . . . . .	101
4.4	Summary . . . . .	104
<b>5</b>	<b>Efficient Processing</b>	<b>107</b>
5.1	Tile-Based Belief Propagation . . . . .	107
5.1.1	Belief Propagation: Preliminary . . . . .	108
5.1.2	Belief Propagation: Cost Analysis . . . . .	110
5.1.3	Proposed Algorithm . . . . .	111
5.1.4	Cost Analysis of the Proposed Algorithm . . . . .	115
5.1.5	Performance of the Proposed Algorithm . . . . .	117
5.1.6	Discussion . . . . .	122
5.2	Fast Message Construction . . . . .	123
5.2.1	Hardware Implementation . . . . .	129
5.2.2	GPU Implementation . . . . .	130
5.2.3	Discussion . . . . .	134
5.3	Noise-Aware Demultiplexing . . . . .	134
5.3.1	Formulation . . . . .	135
5.3.2	Optimization . . . . .	136

5.3.3	Results . . . . .	136
5.3.4	Discussion . . . . .	137
5.4	Summary . . . . .	138
<b>6</b>	<b>Conclusion</b>	<b>139</b>
	<b>Bibliography</b>	<b>142</b>
	<b>Publication and Honors</b>	<b>157</b>

# List of Figures

1.1	Overview of the dissertation. . . . .	2
1.2	Different parameterizations of the light field. . . . .	5
1.3	The light field in the camera. . . . .	6
2.1	The light field parameterizations in light transport analysis. . . . .	10
2.2	The light field parameterization in the camera. . . . .	11
2.3	Transformation of the light field due to the light ray traveling. . . . .	13
2.4	Modulation of the light field due to occlusion. . . . .	15
2.5	Transformation of the light field due to lens refraction. . . . .	17
2.6	The parameterization function $G(v)$ . . . . .	18
2.7	A simple example of the light transport in photography. . . . .	20
2.8	Illustration of the image formation process in light field. . . . .	24
2.9	The effect of the aperture to the image formation. . . . .	28
2.10	The effect of the occluded object to the defocused image. . . . .	32
2.11	The light field spectrums with and without the aperture blocking function. . . . .	37
2.12	The spectrum of the light fields under different focus settings. . . . .	41
2.13	The minimal sampling rates along the $f_v$ axis. . . . .	42
2.14	The relation of the sampling rate and the focus settings. . . . .	43
2.15	Definition of the depth detection problem. . . . .	44
2.16	Detected depths of the two datasets using different methods. . . . .	47
2.17	The refocused and all-focused images from the synthetic datasets. . . . .	49
2.18	The refocused and all-focused images from the real dataset. . . . .	50
3.1	Sequential light field acquisition using the programmable aperture. . . . .	59

3.2	Multiplexing the light field by using the programmable aperture. . . . .	61
3.3	The noise characteristics of Nikon D70 DSLR. . . . .	62
3.4	Performance improvement by multiplexing (1/2). . . . .	64
3.5	Performance improvement by multiplexing (2/2). . . . .	65
3.6	The first prototype: rotary panel. . . . .	66
3.7	The second prototype: pattern scroll. . . . .	67
3.8	The third prototype: liquid crystal array. . . . .	68
3.9	Reducing the aperture size without affecting the angular resolution and the multiplexing. . . . .	71
3.10	The effect of the photometric distortion. . . . .	73
3.11	The flow and results of the photometric calibration. . . . .	76
3.12	Application of the proposed algorithm to the dataset of other light field cameras. . . . .	78
3.13	Overview of the proposed multi-view depth estimation algorithm. . . .	81
3.14	The effect of the cross bilateral filtering. . . . .	84
3.15	Depth maps of the captured light field dataset (1/2). . . . .	86
3.16	Depth maps of the captured light field dataset (2/2). . . . .	87
3.17	The results of the test dataset <b>Tsukuba</b> . . . . .	89
3.18	The results of the test dataset <b>Veuns</b> . . . . .	90
3.19	The results of the test dataset <b>Teddy</b> . . . . .	91
3.20	The results of the test dataset <b>Cones</b> . . . . .	92
3.21	The results of other multi-view depth estimation algorithms. . . . .	93
3.22	Effects of the post-processing algorithms. . . . .	94
4.1	Image interpolated without depth information. . . . .	98
4.2	The illustration of the proposed depth-assisted view interpolation. . . .	99
4.3	Image interpolated with depth information. . . . .	100
4.4	Digital refocusing of a transparent object. . . . .	101
4.5	Digital refocused image with the original angular resolution $4 \times 4$ . . . .	102
4.6	Digital refocused image with the boosted angular resolution $25 \times 25$ . . .	103
4.7	Feature refocusing (1/2). . . . .	105

4.8	Feature refocusing (2/2). . . . .	106
5.1	The message and belief construction. . . . .	109
5.2	Belief propagation for a single tile. . . . .	111
5.3	The flow of the tile-based belief propagation. . . . .	114
5.4	Level-C data reuse for calculating the data terms. . . . .	115
5.5	The bandwidth consumption considering the iteration numbers. . . . .	118
5.6	The disparity maps of <b>Venus</b> from different BP algorithms. . . . .	120
5.7	The disparity maps of <b>Tsukuba</b> from different BP algorithms. . . . .	121
5.8	The robust functions commonly used as the smoothness terms in MRF. . . . .	127
5.9	The hypotheses and the final message. . . . .	127
5.10	(a) The generation of $H[i]$ . (b) The generation of hypotheses. . . . .	128
5.11	The processing elements. . . . .	129
5.12	The MSE of different multiplexing schemes. . . . .	137





# List of Tables

3.1	Performance comparison between the conventional camera, the plenoptic camera, and the programmable aperture camera. . . . .	69
5.1	The pseudo code of the tile-based belief propagation. . . . .	113
5.2	The comparison of the memory and bandwidth consumption. . . . .	116
5.3	The energy of the solutions using different BP algorithms. . . . .	119
5.4	The pseudo code of the message construction in original BP. . . . .	123
5.5	The pseudo code of the existing fast message construction algorithm. . .	124
5.6	The pseudo code of the proposed fast, hardware-oriented message construction algorithm. . . . .	125
5.7	Comparison of gate count. . . . .	129
5.8	The execution time of the test dataset <b>Tsukuba</b> . . . . .	132
5.9	The execution time of the test dataset <b>Cones</b> . . . . .	132



# Abstract

Photography is an abstruse skill. Taking a perfect photo needs a great deal of knowledge in aesthetics, physics, optics, and even electronics and also requires a lot of patience. In this dissertation, we examine the process of photography using 4D light field representation. This new approach leads to a novel framework to describe the image formulation, a new device to capture high dimensional visual data, and many new applications.

In the first part of the dissertation, we analyze the light transport of the light rays in the picture-capturing process and show that several photographic effects, including magnification, focusing, and vignetting, can better be explained in the 4D light field domain or the dual 4D frequency domain than in the 2D image domain. We also show this framework can be applied to many new applications, such as digital refocusing, all-focused fusion, light field camera parameter setting, depth detection without correspondence matching, and so forth.

In the second part of the dissertation, we present a new compact device, called programmable aperture, to capture the light field without moving the camera or losing the image resolution. The device is more flexible, inexpensive, easier to calibrate than the previous light field cameras. It also enables the multiplexing of the light field to improve the data quality. We show several different implementations of the programmable aperture and compare the performance of different imaging devices.

We then remove the inherent defects of the captured light field by two novel post-processing algorithms. The photometric calibration algorithm can automatically remove the vignetting-alike effects without any reference object. The multi-view depth estimation generates per-view depth maps from the light field. It utilizes accurate occlusion model and cross-bilateral filtering to efficiently achieve high quality results. The

combination of the device and the algorithms produce a distortion free, high spatial- and high angular- resolutions light field with auxiliary depth information of the scene. We demonstrate several applications using the captured light field, including view interpolation, digital refocusing, and a novel feature-based refocusing.

In the third part of the dissertation, we describe two spinoff topics. These two topics are not only related to the practical light field acquisition and applications, but are also very important to other computer vision and signal processing problems. The first topic is the computational and storage bottleneck of the global optimization algorithms. We make a complete analysis of the bandwidth and memory cost of the belief propagation and then propose a new tile-based belief propagation algorithm. While its performance is very close to the original belief propagation algorithm, the memory and bandwidth requirements are reduced by orders of magnitude. We also propose a new message construction method which can be applied to all robust smoothness functions. These two algorithms enable efficient VLSI or GPU implementations and make the global optimization more affordable to the resource-limited platforms for real-time mobile applications.

The second topic is about the signal demultiplexing. Traditional demultiplexing method assumes that the noises are independent and identically distributed, which is invalid in the image formulation. We reformulate the demultiplexing process in a probabilistic framework and show that when the noise is dependent to the sensed signal, the MAP estimation is equivalent to a L1 regularized least-square problem. The simulation results show that the signal recovered by the proposed algorithm has a higher signal-to-noise ratio than that recovered by the traditional demultiplexing method. We believe this new formulation can also be applied to other applications that require demultiplexing.

In summary, this dissertation presents several novel and important solutions to analyze, acquire, and efficiently utilize the high dimension, high quality, and high resolution visual data. We show that by using the light field representation, the photographic effects can be described in a way more close to the real functionality of the camera. We present a novel backward-compatible device that can capture high resolution light field data without moving the device itself or using complex optics, and demonstrate several

novel applications of the light field data. Finally, we propose a modified belief propagation algorithm which removes the fundamental memory, bandwidth, and computation bottlenecks of the original algorithm, and a new noise-aware demultiplexing algorithm which has a better performance than the traditional one.



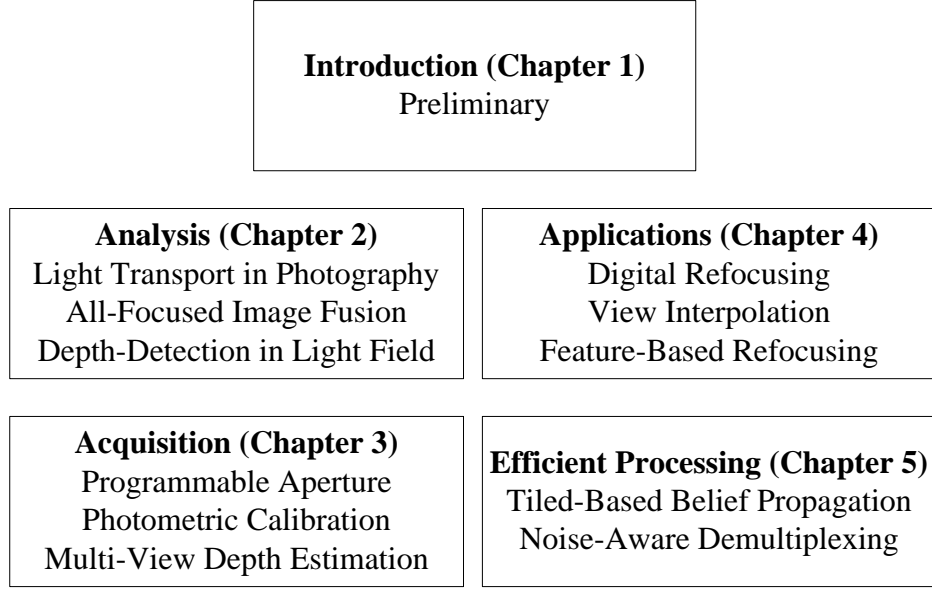
# Chapter 1

## Introduction

Light field has become a popular tool for computer graphics, computer vision, image understanding, image processing, signal processing, geometric optics, and data compression. By representing the radiance carried by a light ray as a sample of the 4D signal, we can analyze the behavior of the light ray in a principled, signal-processing framework. This new representation has enabled numerous novel applications and research topics such as image-based rendering, image-based modeling, precomputed radiance transfer, and so forth. It also has been used to re-formulate many fundamental problems in the visual signal analysis, such as the light transport process in rendering, the inversion of the image formulation, motion analysis, etc. The qualitative results of the old methods can now be quantitatively described using the terminology in the signal processing.

In this dissertation, we will focus on a specific topic, *photography*. The photography is traditionally modeled as a 2D projection of the 3D world. Many natural effects in the image formation process are modeled as independent distortions to a virtual, perfect image. Although CCD or CMOS sensors are widely used in commercial cameras now, these fundamental models never change.

Regarding this, we re-formulate the whole image formation process in the light field domain, which results in an unified light transport model for the photographic effects, and inspires a new device to capture the light field. We show that the new model and new device can enable several novel applications.



**Figure 1.1:** *Overview of the dissertation.*

## 1.1 Overview

The overview of this dissertation is given in Figure 1.1. In the next section of this chapter, we first give the preliminary knowledge about the light field, including the basic mathematical terminology and the history of its development. Since each chapter (or each section in the same chapter) covers a unique research topic, the relevant prior arts are described at the beginning of each chapter/section.

Chapter 2 focuses on the analysis of the light transport in photography. We derive a new model to describe the image formulation using the light transport theory. Using this model, we show that all photographic effects, including focusing, magnification, vignetting, etc, are all results of the transformation, modulation, and projection of the light field. A nice property of this model is that some effects can be explained in the 4D frequency domain. We show an application that uses the concept of auto-focusing to estimate the possible depth planes from the spectrum of the light field, and de-convolves the light field spectrum to recover the all-focused image. We also show that the analysis can be used to specify the parameters of the light field cameras.

In Chapter 3, we will describe a novel device called programmable aperture to acquire the 4D light field. Programmable aperture can dynamically and precisely control



the transmittance of the aperture plane. This enables the camera to sequentially capture the light field without sacrificing the image resolution like other light field cameras do. Also the quality degradation problem can be efficiently resolved by using the multiplexing technique.

Then we present two post-processing algorithms that are essential to produce high quality light field. The first algorithm can automatically remove the complex photometric distortion from the light field without relying on any specific reference object. A more consistent rendering result can be thus obtained from the calibrated light field. The second algorithm exploits the multi-view nature of the light field data to infer accurate per-view depth maps. The estimated depth maps can be applied to several applications, such as anti-aliasing and z-keying. These two algorithms can be applied to not only our light field data, but also those captured by other light field cameras as well.

In Chapter 4 we demonstrate several applications using the captured light field and the estimated depth maps. The first application is view interpolation. While this application is not novel, we focus on the details of efficient implementation, such as how to combine depth maps with light field in the rendering pipeline. We then describe our re-focusing algorithm which combines both the light field and the depth maps to generate plausible and physically correct results. The third application, called feature-based re-focusing, uses robust feature matching to perform homography estimation. This makes user able to adjust the focus plane in a more intuitive way.

Before concluding this dissertation, in Chapter 5 we present two efficient processing methods that are not only related to the light field, but also relevant to other computer vision and signal processing problems. The first one is a modified belief propagation algorithm, dubbed tile-based belief propagation. By only preserving the messages along the boundaries of the tiles, this algorithm can significantly reduce the bandwidth and memory requirement by at least an order, while the performance of the message passing is unaffected. Therefore, this modified algorithm is much more suitable for GPU and VLSI implementation than the original algorithm. Also a fast message construction for parallel processors is described.

The second algorithm is a noise-aware demultiplexing algorithm. Traditionally,

multiplexing and demultiplexing are considered as linear and signal-independent procedures, but in the imaging sensing process the noise depends on the signal. We show that according to the statistical analysis, the optimal noise-dependent solution can be obtained by solving a L1-regularized least square problem.

## 1.2 Preliminary of Light Field

The visual information is a result of the complex interactions between the lighting, object geometry, materials, observer location, and the characteristic of the human visual systems (or of the imaging sensors). To generate a photorealistic image, all the parameters must be precisely modeled or specified, and a great amount of the computations are required to simulate the process of the interactions.

However, if we focus on the geometric optics<sup>1</sup>, the overall effects of those interactions are simply re-direction, scattering, and attenuation of light rays. The pioneering work of Adelson and Bergen shows that distribution of all light after the complex interactions can be uniquely described by a seven-dimensional *plenoptic function* [1]. Specifically, the radiance  $R$  carried by a light ray is a sample of the plenoptic function:

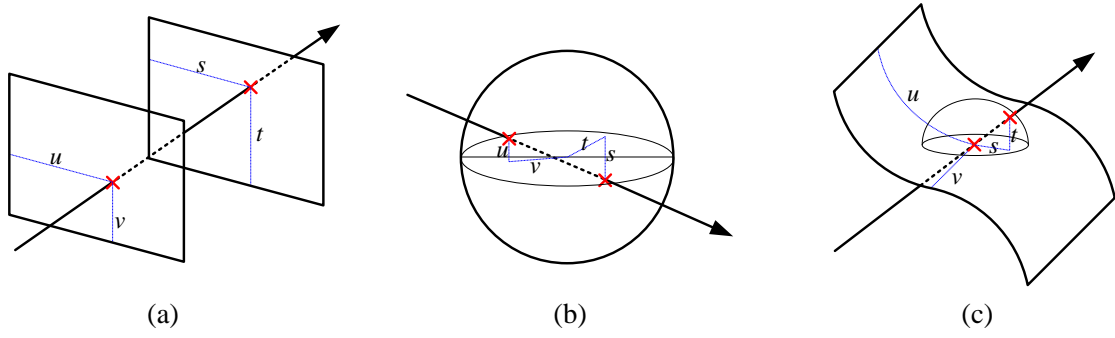
$$R = P(V_x, V_y, V_z, \theta, \phi, \lambda, t), \quad (1.1)$$

where  $(V_x, V_y, V_z)$  are the coordinates of the viewing position,  $(\phi, \lambda)$  are the spherical coordinates of the viewing angle,  $\lambda$  is the wavelength, and  $t$  is the time. A set of these variables can uniquely represent a light ray and thus  $P$  is a 7-to-1 mapping function. Once the whole plenoptic function is recorded, we can re-create all visual content without simulating the interactions.

However, the plenoptic function is expensive and difficult to sample, store, or even reconstruct. Therefore, several simplifications must be imposed. First, one can remove the time axis and focus on the static scene. Second, one can utilize the trichromacy of the human vision system to remove the wavelength axis. The 7-D plenoptic function now becomes three 5-D plenoptic functions, one for each color channel. However, it is still too expensive to sample and store a 5D signal.

---

<sup>1</sup>The lights are incoherent and objects are much larger than the wavelength of light.



**Figure 1.2:** *Different parameterizations of the light field.*

(a) *The plane-plane parameterization, (b) the sphere-sphere parameterization, and (c) the manifold-sphere parameterization.*

If we always stand outside the convex hull of the scene, the observed radiance along a ray is constant since the light ray can never interact with other objects. In other words, there is redundant information in the 5D function. We do not have to specify the viewing position but only the light rays. In the 3D world, every light ray can be parameterized by only four coordinates <sup>23</sup>.

This property is first exploited in [2], where the 4D function is called a *photic field*. Later this function is termed as *light field* in [3] and *lumigraph* in [4]. These methods attempt to achieve the same goal, representing the radiance along the rays in an empty space by a four dimensional function:

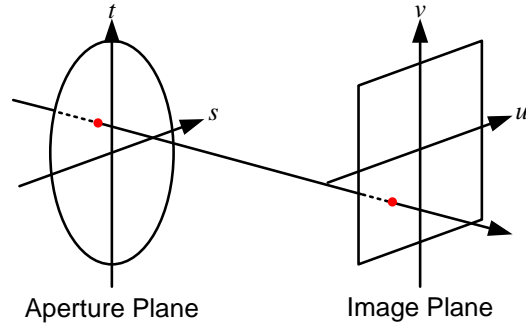
$$R = L(u, v, s, t). \quad (1.2)$$

We will use term light field in the rest of the dissertation.

The light field can be parameterized in many different ways as shown in Figure 1.2. We can parameterize a light ray by its intersections to two planes (Figure 1.2(a)), or we can parameterize it by its intersections to a sphere enclosing the scene (Figure 1.2(b)). Generally speaking, a light ray can be parameterized by the coordinates of its intersection points with two parametric 2D manifolds (Figure 1.2(c)), as long as each light ray only intersects the manifold once. Therefore, the choice of the manifold

<sup>2</sup>Each ray may have 2 different directions but since we only care about the rays emitted outward the convex hull, this ambiguity is removed.

<sup>3</sup>We will describe many different parameterization methods later.



**Figure 1.3:** *The light field in the camera.*

depends on the shape of the convex hull of the scene.

Our main interest is the light fields inside the camera, which can be intuitively and uniquely represented by the plane-plane parameterization, as shown in Figure 1.3. Each light ray in the camera must pass thorough the aperture at some point to enter the camera, and must hit some point on the image plane. The origins of these two planes are defined as the intersections to the optical path.

Given this representation, we attempt to answer many questions in this dissertation. How this light field is related to the light field outside the camera? How a 2D photo is generated from the 4D light field? How to capture this 4D light field with a high resolution and high quality? Can we estimate the scene from the light field? Can we do this efficiently? In the next chapter, we give the theoretical analysis of photography process using the light field representation in the next chapter.

## Chapter 2

# Light Transport in Photography

Different effects in the image formation process are traditionally described by many independent mathematical models. For example, the basic pinhole camera model describes the perspective projection. Objects farther from the camera would be smaller on the image than the objects closer to the camera [5]. The out-of-focus objects usually look blurry. This is usually described by a spatially variant box or Gaussian filter on the image plane. The size of the filter kernel can be derived analytically according to the object depth and camera parameters [6]. The image is usually dimmer at the corner, which is modeled by the vignetting effect [7]. For a natural image with many effects, we usually assume there is a perfect reference image and model those effects as separate degradations to the reference image<sup>1</sup>.

However, if we examine the mechanism of a camera, it actually does not do anything about the filtering or projection<sup>2</sup>. The camera first refracts and blocks the light rays. Then the imaging sensor simply integrates all incoming radiances together into an irradiance value. The footprints of the light rays which are infinitesimally small do not scale with the traveling distance, and thus the light rays are not perspectively projected. The light rays also do not interact with each other so there is no filtering involved.

In other words, while the operation of a camera is extremely simple, people on the contrary use many separate models to describe the various effects in the captured

---

<sup>1</sup>These effects may be applied in a specific order, which was seldom discussed in the past.

<sup>2</sup>We focus on the optics part, not the digital image processing pipeline in the commercial digital cameras.

photo. In this chapter, we present the complete light transport from the object surface to the sensor and show that the photography process can be described by simple transformations and modulations of the light field. The resulting single equation can fully describe several photographic effects together. Both the pinhole and lens camera can be described by the proposed model.

Furthermore, since the light field is a signal and all operations are linear, we can move the whole analysis to the frequency domain. The spectrum of the light field has a very distinct Structure which depends on the scene geometry and the camera parameters. We can use the spectrum to decide the parameters of the cameras or light field cameras. Still, all the photographic effects can be described in the frequency domain as well, while some of them would even have a more intuitive explanation. Although the full light transport theory is more general than the content we present, our work fills the missing piece of the whole transport process.

In the following, we first review the important work and introduce the preliminary concepts and notation. Then, we analyze the transportation of the light rays in the imaging/photography process and derive a combined light field transformation operator. In a sequel, we present how to interpret the photographic effects by the transformation, in both the primal and frequency domains. We also utilize the spectrum of the light field to other applications. In the end, we discuss the future work and conclude this chapter.

## 2.1 Previous Work

The interactions between the light rays and the environment can be exactly described by the rendering equation [8], as long as all parameters of the environment are known. However, these parameters are difficult to obtain and the brute force simulation is usually computationally extensive. Therefore, people are getting more interest in knowing that given the local or statistic property of the environment, is it possible to predict rendering results or reduce the required computation resource for simulation? or is it possible to estimate the environment parameters from the captured images rendered by the nature forces?

The simplest property is the Shannon sampling theory [9]. Because the final goal

of simulating the rendering equation is to generate a visually plausible image, one can specify the sampling rate and the reconstruction filter according to bandwidth of the signal. For example, when a diffuse textured surface is uniformly shaded, the image can be considered as a warping of the texture map due to the perspective transform [10].

Later people have found that other phenomena in rendering can also be interpreted in a signal-processing framework if we consider the light rays as samples of the 4D light field signal. For example, in [11] and [12] it is shown that the shape of the light field signal depends on the depth range<sup>3</sup> of the scene. However, the inter-object interaction like occlusion is ignored. In our analysis we show that the occlusion due to the aperture plays an important role in generating natural images. Similar analysis is also used to address the aliasing problem in the automultiscopic 3D displays [13].

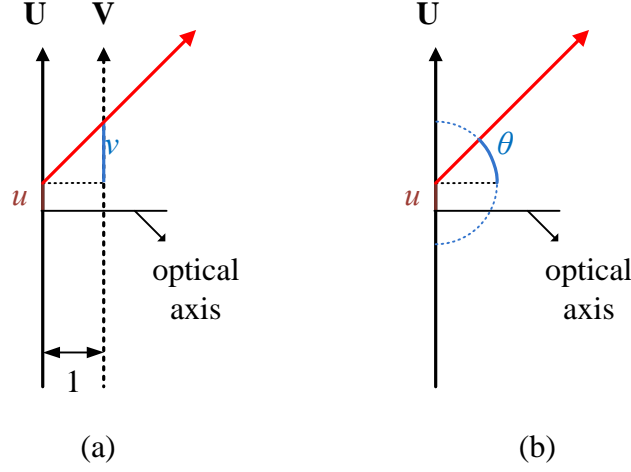
In [14] and [15] a complete framework for forward and inverse rendering is presented. In this framework, the reflected light field is described as a convolution of the lighting (incident light field) and BRDF or a product of the spherical harmonic coefficients of the lighting and BRDF. This seminal work not only analyzes the posedness of the inverse rendering problems but also shows that the expensive rendering equation could be efficiently evaluated or stored in a low-dimensional space in many cases [16]. The following numerous researches in the precomputed radiance transfer (PRT) all attempt to further exploit this property [17]. Other effects like the shadow casting is also analyzed in a similar principle [18].

In [19], phenomena including occlusion, shading, surface curvature, and transport are jointly analyzed by a single signal-processing framework. The framework shows that these phenomena either shear or convolve the light field spectrum, and they can be accumulated together. We follow this line and focus on the effects caused by the lens and aperture of the camera, the final elements in the light transport before reaching the image plane. Finally, in [20] a first-order analysis is proposed to address the cases difficult to model in the Fourier domain.

The light transport analysis is also used in designing many light field cameras. In

---

<sup>3</sup>The distance between the object and the reference plane.



**Figure 2.1:** *The light field parameterizations in light transport analysis.*

(a) *The plane-plane parameterization.* (b) *The plane-sphere parameterization.*

[21], the modulation effect of the aperture is analyzed, but it does not consider the whole light transport process. In [22] the light transport equation similar to what we will develop is presented. However, it only discusses the applications of the light field cameras and totally neglects the effect of the aperture. On the contrary, we present a complete analysis here, which can interpret all photographic effects in traditional cameras and other devices.

## 2.2 General Light Transport Operators

In this section we briefly present the notation and the general light transport operations. We perform our analysis in the 2D space to make the formulation elegant. In this way, we only need two 1-D parametric manifolds to represent a light ray and thus the light field is 2 dimensional. We shall discuss how to extend the analysis to the 3D cases later.

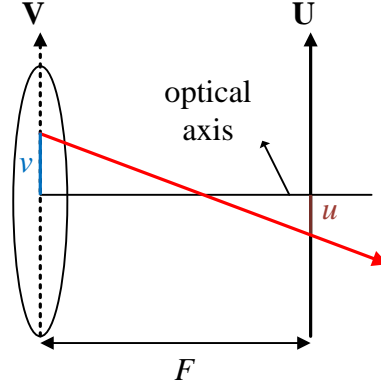
### 2.2.1 Parameterization

We will use many different parameterizations in our analysis<sup>4</sup>. When the light rays propagate in the space, we use the parameterizations shown in Figure 2.1 to represent

---

<sup>4</sup>Note that the plane becomes 1D in the 2D space.





**Figure 2.2:** *The light field parameterization in the camera.*

the light field. In the first plane-plane parameterization we used, the light rays intersect one plane **U** at point  $u$ , and another plane **V** parallel to the first one at point  $v$ . The distance between the two planes is 1 unit and both are perpendicular to the optical path of the camera. The coordinate of the first intersection point is defined with respect to the optical path, and the coordinate of the second intersection point is defined with respect to  $u$ . In other words, for each light ray, the intersection point  $u$  forms a local frame for the plane **V**.

The second plane-sphere parameterization uses one intersection point  $u$  and the angle  $\theta$  with respect to the central direction to represent a light ray. These two parameterizations are linked by  $v = \tan \theta$  and because the tangent function can be linearized when  $\theta \approx 0$ , they are equivalent around the central direction.

After the light rays enter the camera, we use another plane-plane parameterization shown in Figure 2.2. The two planes are defined at the physical aperture plane and the image plane. The distance between these two planes is  $F$ . We will later describe how to transform the parameterization in 2.1(a) to this parameterization.

The light field  $L([u, v]^T)$  is a 2D signal that represents the radiance values carried by the light rays. We will also analyze the spectrum of the light field in the Fourier domain. For the plane-plane parameterization, the spectrum  $\mathcal{L}([f_u, f_v]^T)$  is obtained by

the Fourier transform:

$$\mathcal{L} \left( \begin{bmatrix} f_u \\ f_v \end{bmatrix} \right) = \int_{u=-\infty}^{\infty} \int_{v=-\infty}^{\infty} L \left( \begin{bmatrix} u \\ v \end{bmatrix} \right) \exp(-j2\pi(f_u u + f_v v)) du dv, \quad (2.1)$$

where  $j = \sqrt{-1}$ . To make the equations concise, we also define  $\mathbf{u} = [u, v]^T$  and  $\mathbf{f} = [f_u, f_v]^T$

### 2.2.2 Light Ray Traveling

Assume the light rays of interest are emitted from a surface, we can define the  $\mathbf{U}$  at the surface and have the initial light field  $L_0([u, v]^T)$ . If we sample the light field at  $d$  away from the surface by setting the new reference plane  $\mathbf{U}$  at  $d$ , as shown in Figure 2.2, the sampled light field  $L_1([u, v]^T)$  contains the information exactly the same as  $L_0$  does, but these information are represented in a different way. By simple derivations, we have  $L_1([u, v]^T) = L_0([u - vd, v]^T)$ , which can be written in a matrix form:

$$L_1 \left( \begin{bmatrix} u \\ v \end{bmatrix} \right) = L_0 \left( \begin{bmatrix} 1 & -d \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \right) = L_0(\mathbf{T}_d \mathbf{u}), \quad (2.2)$$

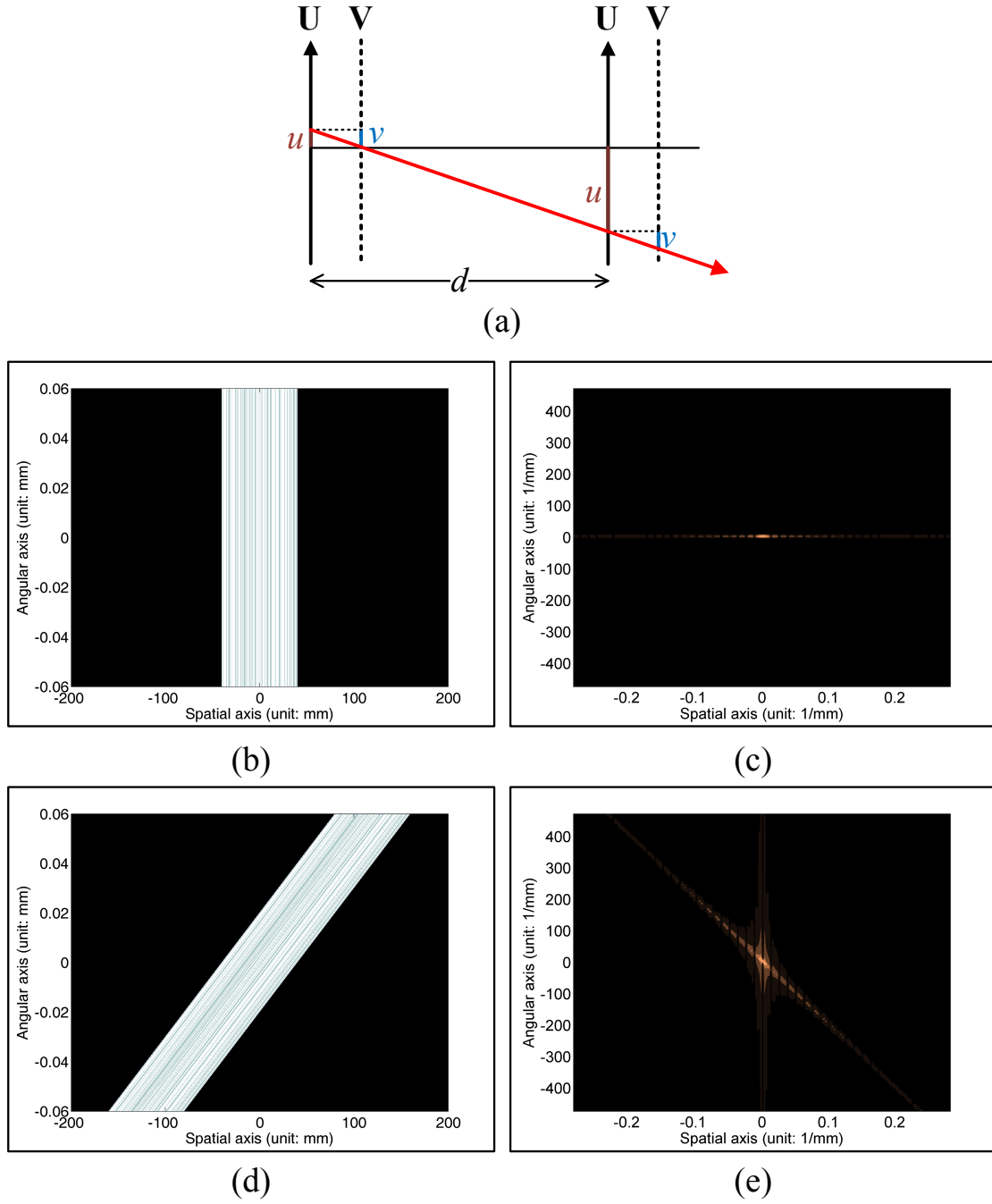
where  $\mathbf{T}_d$  is a  $2 \times 2$  matrix and a function of  $d$ .

According to the Fourier linear transformation theorem, the spectrum of the new light field is defined as:

$$\mathcal{L}_1(\mathbf{f}) = \frac{1}{|\det(\mathbf{T}_d)|} \mathcal{L}_0(\mathbf{T}_d^{-T} \mathbf{f}) = \mathcal{L}_0 \left( \begin{bmatrix} 1 & 0 \\ d & 1 \end{bmatrix} \mathbf{f} \right) \quad (2.3)$$

**Toy example 1:** In Figure 2.3 we show a simple example of the light ray traveling. First, we align the plane with a planar Lambertian object. Because the brightness of this object is direction-invariant, the light field can be described by a texture function  $T(x)$  and thus  $L_1([u, v]^T) = T(u)$ , as shown in Figure 2.3(b).

Because the light field has no angular variation, all its energy would be on the  $f_v = 0$  subspace of the full spectrum. In other words, the power spectrum of the light field would look like a horizontal line in the 2D space, as shown in Figure 2.3(c). One can also think that the light field is obtained by convolving  $T(u)$  with a uniform function along  $v$ , and thus the spectrum is a delta function along  $f_v$ .



**Figure 2.3:** Transformation of the light field due to the light ray traveling.  
 (a) The illustration of the propagation process. (b) The initial light field at the object surface. (c) The spectrum of the initial light field. (d) The transformed light field. (e) The spectrum of the transformed light field.

We then propagate the light rays by 2000 units, update the reference plane, and re-sample the light field. The result is shown in 2.3(d). We can see that the light field is skewed due to the transformation  $T_d$ . One can also treat the light field as a 2D image, and the transformation  $T_d$  is an affine transformation to the image. According to Equation 2.3, the spectrum of the light field is transformed as well, as shown in Figure 2.3(e)<sup>5</sup>. However, the energy of signal still lies on a slice of the 2D space. In other words, the light ray traveling does not convolve the light field in the spatial domain or Fourier domain.

### 2.2.3 Occlusion

When a light ray is blocked by the object during traveling, the radiance of the light ray becomes zero. This effect can be described as a modulation of a binary occlusion function  $B$  to the light field:

$$L_1(\mathbf{u}) = L_0(\mathbf{u})B(\mathbf{u}). \quad (2.4)$$

When  $B(\mathbf{u}) = 0$ , it means the light ray represented by  $\mathbf{u}$  is blocked by the occluder, as shown in Figure 2.4.

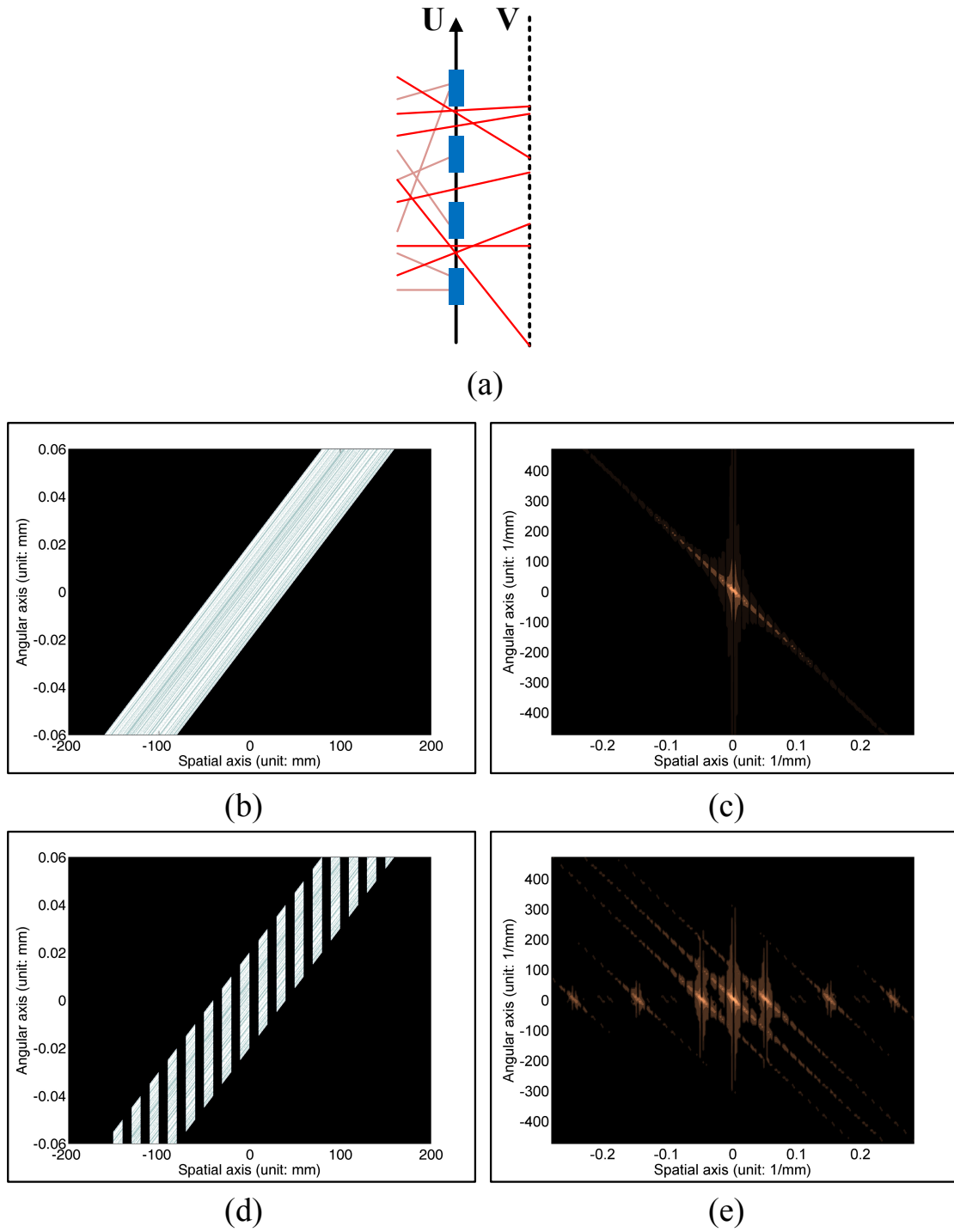
A modulation in the primal domain results in a convolution in the frequency (dual) domain. Let  $\otimes$  denote the convolution operator, then the spectrum of the blocked light field is

$$\mathcal{L}_1(\mathbf{f}) = \mathcal{L}_0(\mathbf{f}) \otimes \mathcal{B}(\mathbf{f}). \quad (2.5)$$

**Toy example 2:** We use the propagated light field in the toy example 1 on page 12, as shown in Figure 2.4(b) as the input. We place a fence as a blocker at the reference plane. The blocked light field is shown in Figure 2.4(d). We can see that the light field is modulated by a rectangle function along the  $u$  axis. The spectrum of the light field is therefore convolved by a sinc function, as shown in Figure 2.4(e). We can see while the original spectrum has only one lobe, the convolved spectrum has multiple side lobes which are repetitive along  $f_u$ .

---

<sup>5</sup>There are some residual errors which are due to the truncation of the signal



**Figure 2.4:** Modulation of the light field due to occlusion.

(a) The illustration of the occlusion process. (b) The light field before blocking. (c) The spectrum of the light field before blocking. (d) The light field after blocking. (e) The spectrum of the light field after blocking.

### 2.2.4 Refraction of the Lens

The general refraction of the light rays follows the Snell's law, which cannot not be expressed linearly in the light field representation. However, here we focus on the refraction of the lens in the camera, which can be elegantly represented using the lens parameters. For example, if we define the  $\mathbf{U}$  plane at the thin lens of focal length  $f$ , as shown in Figure 2.5, the  $u$  coordinate of the light ray is unaffected by the lens, but because the light ray is bent by the lens, the  $v$  coordinate becomes  $v - u/f$  [23]. We can see that this is equivalent to a transformation to the light field:

$$L_1 \left( \begin{bmatrix} u \\ v \end{bmatrix} \right) = L_0 \left( \begin{bmatrix} 1 & 0 \\ 1/f & 1 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \right) = L_0(\mathbf{R}_f \mathbf{u}). \quad (2.6)$$

The spectrum of the transformed light field can be derived similar to the case in Section 2.2.2.

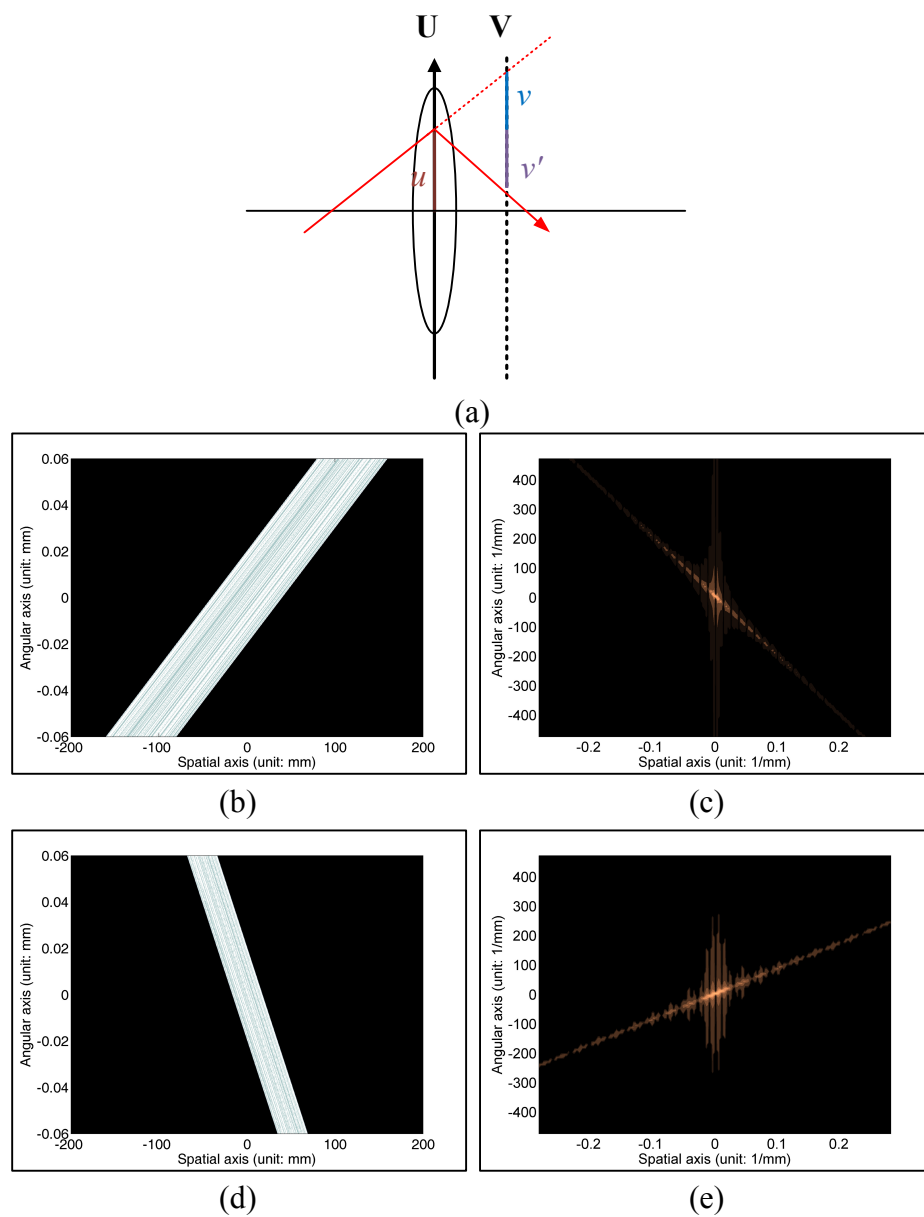
**Toy example 3:** We again use the propagated light field in the toy example 1 as the input. We place a lens with  $f = 600$  at the reference plane as shown in Figure 2.5(a). We can see that the light field is modified by a linear transformed. However, it is not convolved or modulated. The power of the spectrum still lay on a 1D slice in the 2D frequency space. Only the slope of this slice is changed.

### 2.2.5 Diffuse Shading

We consider only the shading by a planar Lambertian surface. For a diffuse reflector, the reflected light field has no angular variation. The outgoing radiance is the integral of the incoming radiance per unit area times the surface albedo  $\rho$

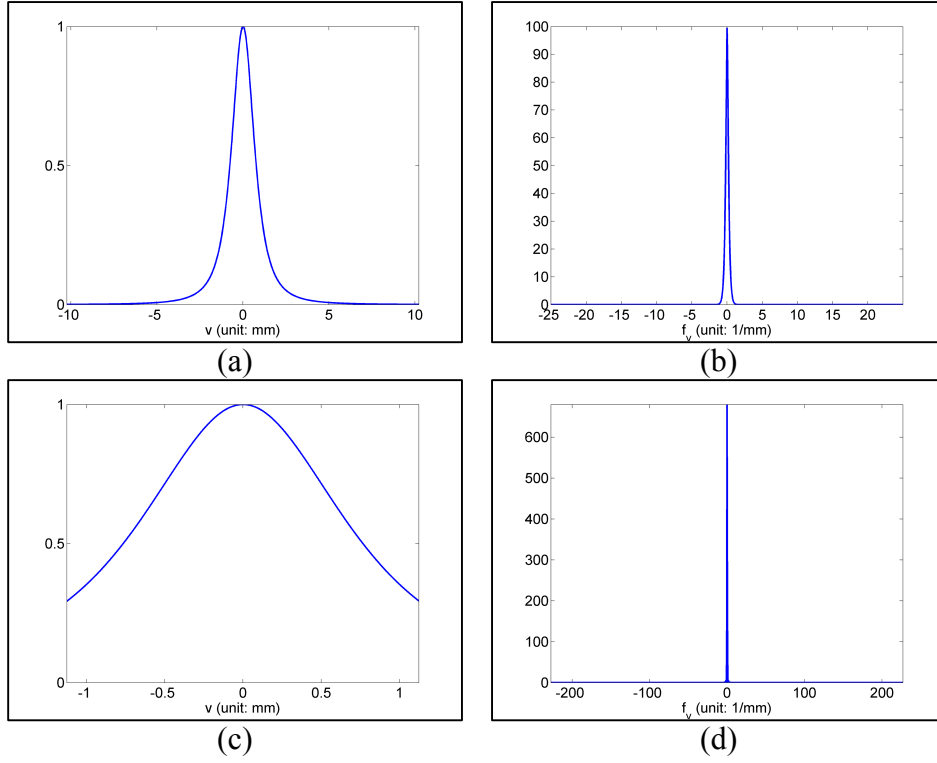
$$\begin{aligned} L_1([u, v]^T) &= L_1(u) = \rho \int L_0(\mathbf{u}) dA \\ &= L_1(u) = \rho \int_{v=-\infty}^{\infty} L_0(\mathbf{u}) (1 + v^2)^{-1.5} dv \\ &= L_1(u) = \rho \int_{v=-\infty}^{\infty} L_0(\mathbf{u}) G(v) dv, \end{aligned} \quad (2.7)$$

where  $G(v)$  is a parameterization function as defined in [19]. We can see that the reflected light field is obtained by first modulating the incoming light field with a pa-



**Figure 2.5:** Transformation of the light field due to lens refraction.

(a) The illustration of the refraction process. (b) The light field before refraction. (c) The spectrum of the light field before refraction. (d) The light field after refraction. (e) The spectrum of the light field after refraction.



**Figure 2.6:** The parameterization function  $G(v)$ .

parameterization function, and then integrating along the angular axis. This integration is also known as a canonical *projection operator*.

As shown in [24], this projection operator in the primal domain is equivalent to taking a lower-dimensional slice in the frequency domain. For the completeness, we derive the 2D case below. Let  $Y(u)$  be a projected 1D signal of  $X([u, v]^T)$ , then the spectrum  $\mathcal{Y}(f_u)$  is defined as

$$\begin{aligned}
 \mathcal{Y}(f_u) &= \int_{u=-\infty}^{\infty} Y(u) \exp(-j2\pi f_u u) du \\
 &= \int_{u=-\infty}^{\infty} \int_{v=-\infty}^{\infty} X(\mathbf{u}) \exp(-j2\pi(f_u u + 0v)) du dv \\
 &= \mathcal{X}([f_u, 0]^T) = [\mathcal{X}(\mathbf{f})]_{f_v=0}.
 \end{aligned} \tag{2.8}$$

Therefore, the spectrum of  $L_1$  is the slice of  $\mathcal{L}_0 \otimes \mathcal{G}$  along the  $f_v$  axis:

$$\mathcal{L}_1(f_u) = \rho[\mathcal{L}_0(\mathbf{f}) \otimes (\mathcal{G}(f_v) \delta(f_u))]_{f_v=0}. \tag{2.9}$$

Note that  $G$  is constant along  $u$  so the spectrum is a Dirac delta function of  $f_u$ .



We should elaborate more the property of  $G$ , which is depicted in Figure 2.6(a). It is a smooth band-limited function and thus the energy falls on the low frequency bands. Moreover, in the case we are interested in (the light transport in image formation), the range of  $\nu$  is very small, usually within  $(-1.2, 1.2)$ , as shown in Figure 2.6(c). In this case, the range of the spectrum is much larger than the bandwidth of the signal, and thus  $\mathcal{G}(f_\nu)$  would be approximately a delta function (Figure 2.6(d)). Therefore, the effect of the convolution in Equation 2.9 is negligible.

### 2.2.6 Accumulation of Transformations

Because the operators presented above are linear, they can be accumulated together. For example, assume that there are three different parameterized light fields  $L_0$ ,  $L_1$ , and  $L_2$ . The reference plane of  $L_0$  is at the left, that of  $L_1$  is  $d_1$  away, and the that of  $L_2$  is  $d_2$  away from  $L_1$ 's reference plane. Using Equation 2.2 and the associativity rule,  $L_2$  and  $L_0$  are related by:

$$L_2(\mathbf{u}) = L_1(\mathbf{T}_{d_2}\mathbf{u}) = L_0(\mathbf{T}_{d_1}(\mathbf{T}_{d_2}\mathbf{u})) = L_0((\mathbf{T}_{d_1}\mathbf{T}_{d_2})\mathbf{u}). \quad (2.10)$$

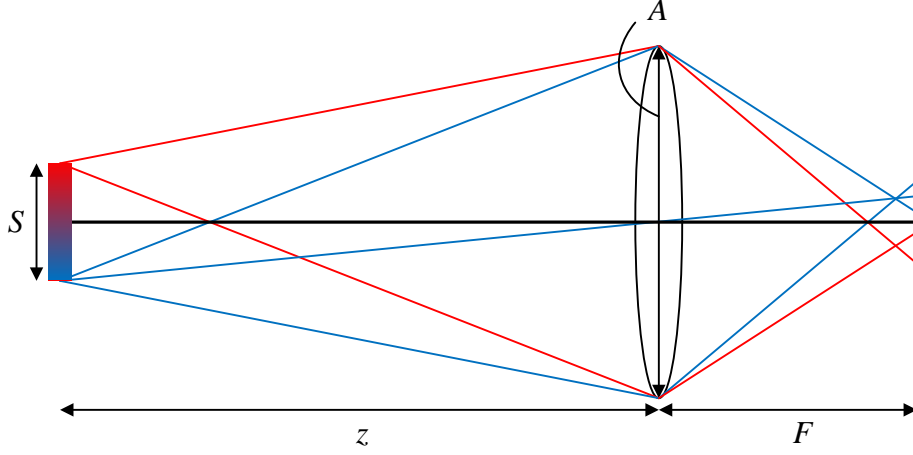
The  $(\mathbf{T}_{d_1}\mathbf{T}_{d_2})$  is the accumulated operator that describes the transformation of the light field when the light ray travel  $d_1 + d_2$  away from the origin.

We can also use the fact that the reference planes of  $L_2$  and  $L_1$  are  $d_1 + d_2$  away and obtain

$$L_2(\mathbf{u}) = L_0(\mathbf{T}_{d_1+d_2}\mathbf{u}). \quad (2.11)$$

With simple calculation one can obtain  $\mathbf{T}_{d_1+d_2} = \mathbf{T}_{d_1}\mathbf{T}_{d_2}$ . That is, the accumulated operator is equivalent to the direct operator.

Therefore, as the light rays propagate, refract, and block in the space during traversal, we can simply accumulate all the operators sequentially without directly deriving the operator that describes the overall light transport effect. While the modulation is involved in the blocking and diffuse shading operators, we will see that these modulations can be carried with the transformation together.



**Figure 2.7:** A simple example of the light transport in photography.

## 2.3 Combined Transformation in Photography

We have presented the basic operators in light transport, now we use them to analyze the light transport process in photography. Without loss of generality, we use scene and camera setup shown in Figure 2.7 for derivation. The distance between the lens and the image plane is  $F$ , and the diameter of the aperture is  $A$ . The initial light field  $L_i$  is measured at  $z$  units away from the lens.

### 2.3.1 Derivation without Aperture

The light rays first propagate  $z$  units to reach the lens, and thus the light field is transformed by  $\mathbf{T}_d$ . Then the light rays are bent by the lens and the light field is transformed by  $\mathbf{R}_f$ . Finally the light rays propagate to the image sensor and the light field is transformed by  $\mathbf{T}_F$ . Let's ignore the effect of the aperture for now and accumulate these operators together, then we can obtain the final light field  $L_p$ :

$$\begin{aligned}
 L_p \left( \begin{bmatrix} u \\ v \end{bmatrix} \right) &= L_i \left( \mathbf{T}_d \mathbf{R}_f \mathbf{T}_F \begin{bmatrix} u \\ v \end{bmatrix} \right) \\
 &= L_i \left( \begin{bmatrix} 1 - \frac{z}{f} & -Fz(\frac{1}{z} + \frac{1}{F} - \frac{1}{f}) \\ \frac{1}{f} & 1 - \frac{F}{f} \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \right). \quad (2.12)
 \end{aligned}$$

Therefore, the light field in the camera is a linear transformation of the light field of the outside world. Although we place a specific object in the scene to simplify the description, any light field can be transformed into the camera using this transformation. We call the accumulated operator  $\mathbf{T}_d \mathbf{R}_f \mathbf{T}_F$  *photographic operator*, which is denoted by  $\mathbf{C}_{zfF}$ . Note that in the  $(2, 2)$  entity of  $\mathbf{C}_{zfF}$ , we can see a term very similar to the well-known lens equation. This term indicates that the lens equation is implicitly embedded in our framework.

If the sensitivity of the sensor is direction-invariant, then the irradiance value it records is equivalent to the reflected radiance from a diffuse surface. Therefore, the captured image  $I_p(u)$  can be defined using Equation 2.7:

$$I_p(u) = \int_{v=-\infty}^{\infty} L_p(\mathbf{u}) G(v) dv. \quad (2.13)$$

The constant sensitivity factor is removed here for simplicity.

The spectrum of the transformed light field can be easily derived using the Fourier linear transformation theorem:

$$\begin{aligned} \mathcal{L}_p(\mathbf{f}) &= \mathcal{L}_i(\mathbf{C}_{zfF}^{-T} \mathbf{f}) \\ &= \mathcal{L}_i \left( \begin{bmatrix} 1 - \frac{F}{f} & -\frac{1}{f} \\ Fz(\frac{1}{z} + \frac{1}{F} - \frac{1}{f}) & 1 - \frac{z}{f} \end{bmatrix} \begin{bmatrix} f_u \\ f_v \end{bmatrix} \right). \end{aligned} \quad (2.14)$$

The spectrum of the captured image is

$$\mathcal{I}_p(f_u) = [\mathcal{L}_p(\mathbf{f}) \otimes (\mathcal{G}(f_v) \delta(f_u))]_{f_v=0} \approx \mathcal{L}_p(f_u, 0). \quad (2.15)$$

The approximation is valid because compared with other effects, the convolution effect due to  $\mathcal{G}$  is very small. Therefore, the spectrum of the image is the slice of the light field spectrum along the line  $f_v = 0$ . This approximation can simplify the analysis in certain cases.

### 2.3.2 Derivation with Aperture

In the derivations above the aperture is ignored. The aperture can be considered as an occluder at the lens plane. Its blocking function  $B_A(\mathbf{u})$  is a binary field, that is,

$$B_A(\mathbf{u}) = \text{rect}\left(\frac{u}{A}\right) = \begin{cases} 1, & |u| < \frac{A}{2} \\ 0, & \text{otherwise} \end{cases} \quad (2.16)$$

There are two sets of discontinuities in this binary field, one is along the line  $u - A = 0$  and the other one is along the line  $u + A = 0$ . This function is a box function along  $u$  and constant along  $v$ , and thus its spectrum is defined as

$$\mathcal{B}_A(\mathbf{f}) = A \text{sinc}(Af_u) \delta(f_v) = (\pi f_u)^{-1} \sin(\pi A f_u) \delta(f_v). \quad (2.17)$$

When the light rays reach the lens, the light field is transformed by  $R_f T_d$  and also modulated by  $B_A$ , and then propagate. Because the order of modulation and transformation can be exchanged, the real light field in the camera  $L_r$  is:

$$L_r(\mathbf{u}) = L_p(\mathbf{u}) B_A(\mathbf{T}_F \mathbf{u}) = L_i(\mathbf{C}_{zfF} \mathbf{u}) B_A(\mathbf{T}_F \mathbf{u}), \quad (2.18)$$

and the spectrum is

$$\mathcal{L}_r(\mathbf{f}) = \mathcal{L}_i(\mathbf{C}_{zfF}^{-T} \mathbf{f}) \otimes \mathcal{B}_A(\mathbf{T}_F^{-T} \mathbf{f}). \quad (2.19)$$

However, this spectrum cannot be easily expressed because the discontinuities of  $B_A$  is not aligned with the principle axes. We resolve this problem by reparameterizing the light field using the parameterization in Figure 2.2. We use the variables with a widehat to denote the light field with the in-camera parameterization. For a light field  $L_a$  defined using the first parameterization with the reference plane at the image plane, it can be represented as  $\widehat{L}_b$  in the new parameterization. These two light fields are linked by a linear transformation:

$$L_b(\mathbf{u}) = \frac{1}{F} L_a \left( \begin{bmatrix} 1 & 0 \\ F^{-1} & -F^{-1} \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \right) = \frac{1}{F} L_a(\mathbf{P}_F \mathbf{u}). \quad (2.20)$$

The constant factor  $1/F$  is Jacobian of the transformation<sup>6</sup>. Again this factor is not important to the analysis and will be dropped for simplicity.

---

<sup>6</sup>The Jacobian of  $T_d$  and  $R_f$  are both 1.

The blocking function  $B_A(\mathbf{T}_F \mathbf{u})$  under the in-camera parameterization becomes

$$\hat{B}_A(\mathbf{u}) = \text{rect}\left(\frac{v}{A}\right) = \begin{cases} 1, & |v| < \frac{A}{2} \\ 0, & \text{otherwise} \end{cases} \quad (2.21)$$

We can see the discontinuities are along  $v - A = 0$  and  $v + A = 0$ , and the spectrum has a simpler form as in Equation 2.17 now. Therefore, if we reparameterize Equation 2.18, we obtain:

$$\begin{aligned} \hat{L}_r(\mathbf{u}) &= L_p(\mathbf{P}_F \mathbf{u}) \hat{B}_A(\mathbf{u}) \\ &= L_i(\mathbf{C}_{zfF} \mathbf{P}_F \mathbf{u}) \hat{B}_A(\mathbf{u}) \\ &= L_i\left(\begin{bmatrix} -\frac{z}{F} & z(\frac{1}{z} + \frac{1}{F} - \frac{1}{f}) \\ \frac{1}{F} & \frac{1}{f} - \frac{1}{F} \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}\right) \hat{B}_A(\mathbf{u}) \\ &= L_i(\hat{\mathbf{C}}_{zfF} \mathbf{u}) \hat{B}_A(\mathbf{u}) \end{aligned} \quad (2.22)$$

We can see that this light field is the transformed light field modulated with a simple binary function. The spectrum of  $\hat{L}_r$  is

$$\begin{aligned} \hat{\mathcal{L}}_r(\mathbf{f}) &= \frac{1}{|\det \hat{\mathbf{C}}_{zfF}|} \mathcal{L}_i(\hat{\mathbf{C}}_{zfF}^{-T} \mathbf{f}) \otimes \hat{\mathcal{B}}_A(\mathbf{f}) \\ &= F \mathcal{L}_i\left(\begin{bmatrix} 1 - \frac{F}{f} & 1 \\ Fz(\frac{1}{z} + \frac{1}{F} - \frac{1}{f}) & z \end{bmatrix} \begin{bmatrix} f_u \\ f_v \end{bmatrix}\right) \otimes \hat{\mathcal{B}}_A(\mathbf{f}). \end{aligned} \quad (2.23)$$

The image captured with the finite aperture is

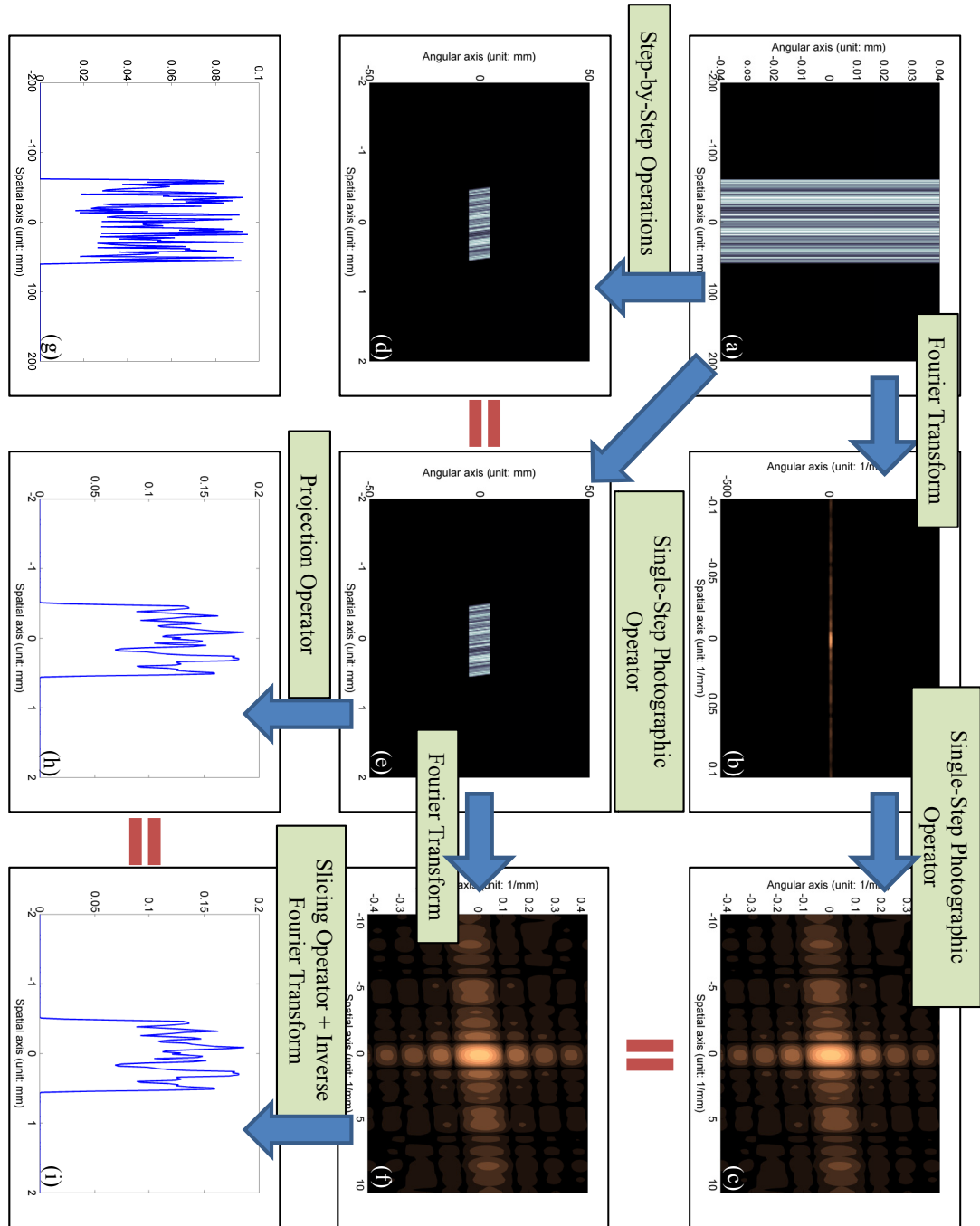
$$I_r(u) = \int_{v=-\infty}^{\infty} \hat{L}_r(\mathbf{u}) G(v) dv = \int_{v=-\infty}^{\infty} \hat{L}_r(\mathbf{u}) G\left(\frac{u-v}{F}\right) dv \quad (2.24)$$

Again the modulation effect of  $G$  is negligible, and the spectrum of the image is:

$$\mathcal{J}_r(f_u) \approx [\hat{\mathcal{L}}_r(\mathbf{f})]_{f_v=0} = [\mathcal{L}_i(\hat{\mathbf{C}}_{zfF}^{-T} \mathbf{f}) \otimes (\text{Asinc}(Af_v) \delta(f_u))]_{f_v=0}. \quad (2.25)$$

In other words, the spectrum of the 1D image is obtained by first transforming the 2D spectrum of the light field by  $\hat{\mathbf{C}}_{zfF}^{-T}$ , convolving it with the aperture blocking function, and then finally extracting the 1D spectrum along the line  $f_v = 0$ . We will call this step *slicing* operator.

**Toy example 4:** A example using the setting in Figure 2.7 is given in 2.8. The planar Lambertian object is at  $z = 6000$ , the focal length  $f$  of the lens is 50, and the focus plane



**Figure 2.8:** Illustration of the image formation process in light field.

is set at  $z = 4000$  (i.e.,  $F = 1/(1/50 - 1/4000)$ ). Figure 2.8(a) and (b) show the original light field  $L_i$  sampled at the object surface and its spectrum  $\mathcal{L}_i$ , respectively. Because the surface is Lambertian, the spectrum is a delta function along  $f_v$ .

Figure 2.8(d) is the light field modified by sequentially using the  $T_z$ ,  $R_f$ ,  $T_F$ ,  $P_F$  and  $B_A$  operators, and Figure 2.8(e) is the light field  $\hat{L}_r$  obtained using the combined photographic operator (Equation 2.22). We can see the light fields obtained by these two methods are equivalent<sup>7</sup>.

We can also perform the transform in the frequency domain using Equation 2.23, the resulting spectrum  $\hat{\mathcal{L}}_r$  is shown in Figure 2.8(c). Compare to Figure 2.8(f), the Fourier transform of Figure 2.8(e), the transformation is correct.

Finally, we can either obtain the 1D image  $I_r$  from the light field using the projection operator (Equation 2.24), as shown in Figure 2.8(h), or from the light field spectrum using the slicing operator (Equation 2.25), as shown in 2.8(i). These two results are almost identical; we can see compare to the original texture function (2.8(g)), the image is a little blurry due to defocus.

## 2.4 Interpretation of Photographic Effects

The equations in the previous section describe all processes in photography. In this section we show how those equations are related to the mathematical models used to describe the separate photographic effects, such as blurring due to defocus.

### 2.4.1 Perspective Transformation

It is well known that in the pinhole camera model, a scene point at  $(x, z)$  is projected to  $(-xF/z)$  on the image plane<sup>8</sup>. This projection can be described as a perspective transformation using the homogenous coordinate. Here we show that this effect is already implicitly embedded in the photographic operator.

First let's focus on case of the pinhole camera. A pinhole camera has an infinitesimally small aperture, which means  $A \rightarrow 0$  and  $\hat{B}_A(\mathbf{u})$  becomes a delta function of  $v$ . Put

<sup>7</sup>There are some errors due to re-sampling in the simulation process.

<sup>8</sup>Note that we perform the analysis in the 2D space for now.

this into Equation 2.22 and 2.24, we have

$$\widehat{L}_r(\mathbf{u}) = L_i(\widehat{\mathbf{C}}_{zfF}\mathbf{u})\delta(v) \quad (2.26)$$

and

$$I_r(u) = [\widehat{L}_r(\mathbf{u})]_{v=0}. \quad (2.27)$$

Put these two equations together then we have

$$\begin{aligned} I_r(u) &= \widehat{L}_r\left(\begin{bmatrix} u \\ 0 \end{bmatrix}\right) \\ &= L_i\left(\begin{bmatrix} -\frac{z}{F} & z(\frac{1}{z} + \frac{1}{F} - \frac{1}{f}) \\ \frac{1}{F} & \frac{1}{f} - \frac{1}{F} \end{bmatrix} \begin{bmatrix} u \\ 0 \end{bmatrix}\right) = L_i\left(\begin{bmatrix} -\frac{z}{F}u \\ \frac{1}{F}u \end{bmatrix}\right) \end{aligned} \quad (2.28)$$

If we assume a diffuse object is at depth  $z$  and its surface albedo is defined by a texture map  $T(x)$ , then we have  $L_i([u, v]^T) = T(u)$  and

$$I_r(u) = T\left(-\frac{z}{F}u\right) \Rightarrow I_r\left(-\frac{F}{z}u\right) = T(u), \quad (2.29)$$

which is exactly the equation of the perspective transform. In other words, the pinhole camera captures a specific subset of the light rays, or a slice of the light field. This 1D slice is stretched on the image plane and the stretch factor is a function of  $z$  and  $F$ .

Now let's see a more general case where the camera is a lens camera and the object is in-focus. In that case,  $1/z + 1/F - 1/f = 0$  and  $\widehat{\mathbf{C}}_{zfF}$  becomes

$$\widehat{\mathbf{C}}_{zfF} = \begin{bmatrix} -\frac{z}{F} & 0 \\ \frac{1}{F} & \frac{1}{f} - \frac{1}{F} \end{bmatrix}, \quad (2.30)$$

and put this into Equation 2.22 we have

$$\widehat{L}_r(\mathbf{u}) = L_i\left(\begin{bmatrix} -\frac{z}{F}u \\ \frac{1}{F}u + \left(\frac{1}{f} - \frac{1}{F}\right)v \end{bmatrix}\right) \widehat{B}_A(\mathbf{u}). \quad (2.31)$$

Again, when the object is Lambertian, all light rays reaching  $u$  on the image plane come from the same scene point  $(-zu/F, z)$ , only the emitting angles are different. In other words, for a lens camera, when the object is in-focus, the perspective transformation is still valid.



A great feature of our analysis is that this conclusion can also be derived in the Fourier domain. When the object is in-focus, the convolution effect in Equation 2.25 can be removed<sup>9</sup>. Then we can combine  $1/z + 1/F - 1/f = 0$ , Equation 2.23 and Equation 2.25 and obtain

$$\mathcal{J}_r(f_u) = \mathcal{L}_i \left( \begin{bmatrix} 1 - \frac{F}{f} & 1 \\ 0 & z \end{bmatrix} \begin{bmatrix} f_u \\ 0 \end{bmatrix} \right) = \mathcal{L}_i \left( \begin{bmatrix} \left(1 - \frac{F}{f}\right) f_u \\ 0 \end{bmatrix} \right), \quad (2.32)$$

where  $1 - \frac{F}{f} = F(-\frac{1}{z}) = -\frac{F}{z}$  and thus  $\mathcal{J}_r(f_u) = \mathcal{L}_i(-\frac{F}{z}f_u, 0) = \mathcal{T}(-\frac{F}{z}f_u)$ . By simple variable substitution, we have

$$\mathcal{T}(f_u) = \mathcal{J}_r(-\frac{z}{F}f_u) \quad (2.33)$$

We can see that the spectrum of the image is the warped version of the surface texture. In practice,  $F \ll z$ , and thus a frequency component of the texture function is always mapped to a higher frequency component on the image. A simple example is that when a person walk away from the camera (increasing  $z$ ), the pattern on his/her cloth becomes denser.

In summary, when the object is in-focus, the shape or the texture of the object is warped/stretched when reaching the image plane. The stretching factor is uniform and independent of the angular variation. It only depends on the object depth  $z$  and the camera parameter  $F$ . The result is identical to that of the perspective transformation.

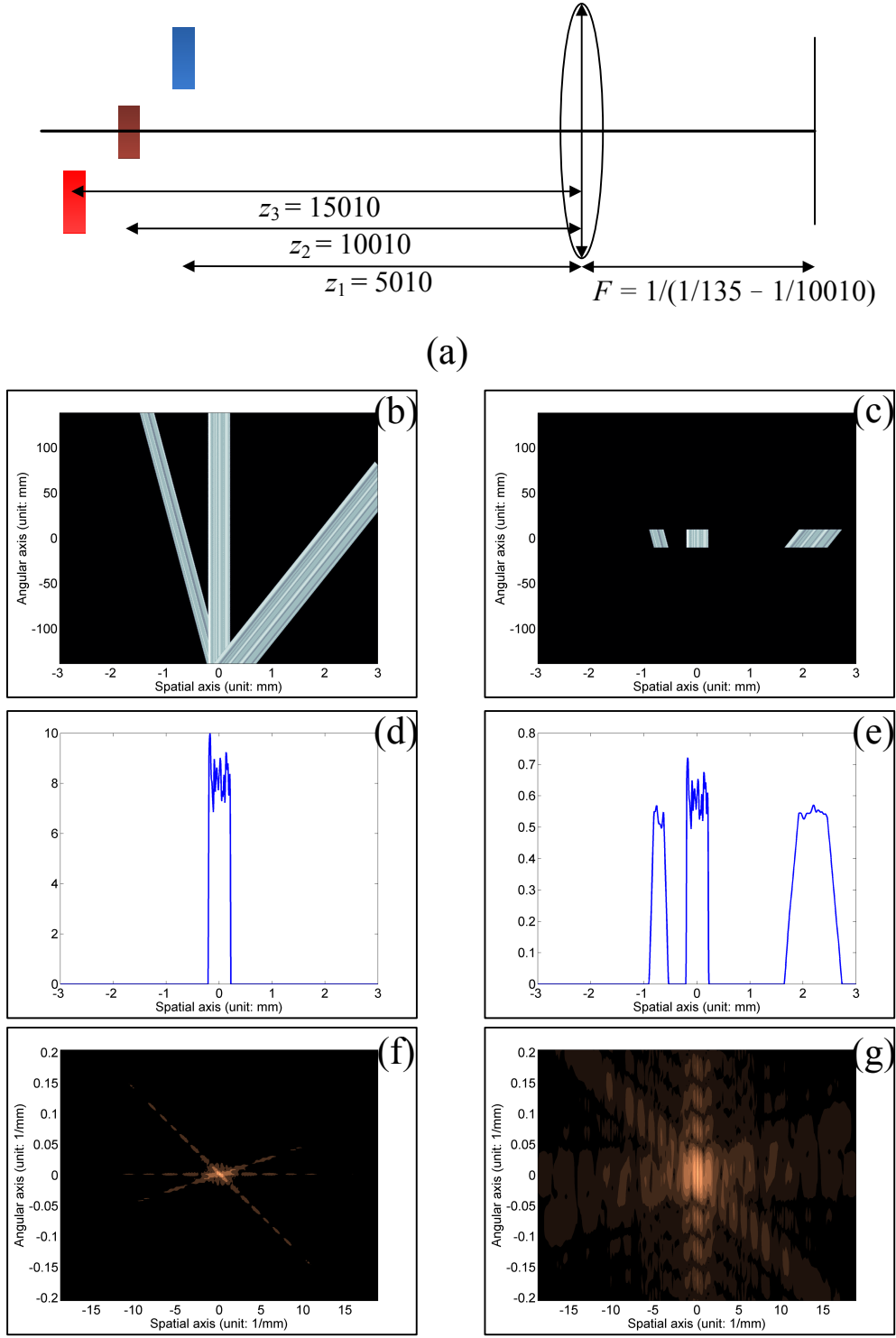
## 2.4.2 Defocus Blur

The defocused objects appear blurry in the image. This is often modeled by a box or Gaussian filter to the image, where the filter parameters are functions of the depth and the camera parameters [6]. This filter is also called the point spread function (PSF). Here we show that this filtering effect is actually a combination of the transformation, modulation, and projection operators to the light field.

In the analysis above we rely on the forward light transport, here we analyze this blur effect using the backward transport. In the backward transport, the transformation oper-

---

<sup>9</sup>This will be elaborated more in the next subsection. For a pin-hole camera, all objects are in-focus and thus the following analysis is also valid.



**Figure 2.9:** The effect of the aperture to the image formation.

- (a) The camera and scene setup.  $f = 135$ ,  $A = 20$  or infinite. (b) The transformed light field when the aperture size is infinite ( $\hat{L}_p$ ). (c) The transformed light field when the aperture size is 20 ( $\hat{L}_r$ ). (d) The observed image when the aperture size is infinite ( $I_p$ ). (e) The observed image when the aperture size is 20 ( $I_r$ ). (f) The spectrum of (b) ( $\hat{L}_p$ ). (g) The spectrum of (c) ( $\hat{L}_r$ ).

ators including the light ray traveling and refraction are reversed. That is, if  $L_1(\mathbf{Mu}) = L_0(\mathbf{u})$ , one can immediately have  $L_0(\mathbf{u}) = L_1(\mathbf{M}^{-1}\mathbf{u})$  and  $\mathcal{L}_0(\mathbf{f}) = \mathcal{L}_1(\mathbf{M}^T\mathbf{f})$ <sup>10</sup>. The backward representation of Equation 2.14 is

$$\mathcal{L}_i(\mathbf{f}) = \mathcal{L}_p \left( \begin{bmatrix} 1 - \frac{z}{f} & \frac{1}{f} \\ -Fz(\frac{1}{z} + \frac{1}{F} - \frac{1}{f}) & 1 - \frac{F}{f} \end{bmatrix} \begin{bmatrix} f_u \\ f_v \end{bmatrix} \right). \quad (2.34)$$

For a Lambertian object like the one in the toy examples, all energies of  $\mathcal{L}_i$  fall on the  $f_v = 0$  slice. Using Equation 2.34, we can see the mapping of these energies is

$$\mathcal{L}_i \left( \begin{bmatrix} f_u \\ 0 \end{bmatrix} \right) = \mathcal{L}_p \left( \begin{bmatrix} (1 - \frac{z}{f}) f_u \\ -Fz(\frac{1}{z} + \frac{1}{F} - \frac{1}{f}) f_u \end{bmatrix} \right). \quad (2.35)$$

As we have derived in the previous subsection, when the object is in-focus, the energies fall on the  $f_v = 0$  slice again. However, when the object is out-of-focus,  $\frac{1}{z} + \frac{1}{F} - \frac{1}{f} \neq 0$ , then the energy will never fall on the  $f_v = 0$  line except the DC term ( $f_u = 0$ ). As a result, the spectrum of the image obtained using Equation 2.15 is zero except the DC term. This means if the object is not perfectly in-focus, it would be totally absent in the captured image.

This result is contradictory to our daily experience. When something is out-of-focus, it simply appears blurry in the captured image. The truth is, the blocking effect due to the finite aperture is the main cause of the defocused blur. In Equation 2.25, the spectrum is convolved with a sinc function along the  $f_v$  direction. Therefore, the slice is spread out along  $f_v$ . The spread energies touch the  $f_v = 0$  and thus appear on the spectrum of the image. Therefore, the image contains the information of the out-of-focus objects. In the following, we derive this effect explicitly.

Let  $\widehat{\mathcal{L}}_p$  be the light field in the camera without aperture under the in-camera parameterization. We first use the backward transport to link  $\widehat{\mathcal{L}}_p$  and  $\mathcal{L}_i$ :

$$\mathcal{L}_i(\mathbf{f}) = \widehat{\mathcal{L}}_p \left( \begin{bmatrix} -\frac{z}{F} & \frac{1}{F} \\ z(\frac{1}{z} + \frac{1}{F} - \frac{1}{f}) & \frac{1}{f} - \frac{1}{F} \end{bmatrix} \begin{bmatrix} f_u \\ f_v \end{bmatrix} \right). \quad (2.36)$$

and again when the  $L_i$  is defined on a Lambertian surface, the mapping of the energies

<sup>10</sup>The Jacobian is omitted here for simplicity.

is

$$\mathcal{L}_i \left( \begin{bmatrix} f_u \\ 0 \end{bmatrix} \right) = \widehat{\mathcal{L}}_p \left( \begin{bmatrix} -\frac{z}{F} f_u \\ z(\frac{1}{z} + \frac{1}{F} - \frac{1}{f}) f_u \end{bmatrix} \right). \quad (2.37)$$

That is, before the convolution with  $\widehat{\mathcal{B}}_A$ , the energies fall on the line  $f_v = -F(\frac{1}{z} + \frac{1}{F} - \frac{1}{f})f_u$ . Therefore, the spectrum of the image given in Equation 2.25 becomes

$$\begin{aligned} \mathcal{J}(f_u) &= \widehat{\mathcal{L}}_p \left( \begin{bmatrix} f_u \\ -F(\frac{1}{z} + \frac{1}{F} - \frac{1}{f}) f_u \end{bmatrix} \right) \text{Asinc} \left( -AF \left( \frac{1}{z} + \frac{1}{F} - \frac{1}{f} \right) f_u \right) \\ &= \mathcal{L}_i \left( \begin{bmatrix} -\frac{F}{z} f_u \\ 0 \end{bmatrix} \right) \text{sinc} \left( -AF \left( \frac{1}{z} + \frac{1}{F} - \frac{1}{f} \right) f_u \right), \end{aligned} \quad (2.38)$$

Now we can see that the image is a filtered signal of the stretched surface texture signal.

It should be noted that when the object is in-focus, this equation is identical to Equation 2.33. Moreover, when the object is out-of-focus, the energy of the object does not shift in the frequency space. It is only attenuated by the sinc function. This means when the object is out-of-focus to a lens camera, the perspective transformation is still valid. While this phenomenon is hard to describe in the image-based approach, we can easily model it using light transport analysis.

If the texture is a delta function,  $\mathcal{L}_i$  is uniform and so is  $\widehat{\mathcal{L}}_p$  uniform. The image spectrum of this delta function is the frequency response function of the system:

$$\mathcal{H}(f_u) = \text{Asinc} \left( -AF \left( \frac{1}{z} + \frac{1}{F} - \frac{1}{f} \right) f_u \right). \quad (2.39)$$

and the response function is

$$H(u) = \frac{1}{F|\frac{1}{z} + \frac{1}{F} - \frac{1}{f}|} \text{rect} \left( \frac{u}{AF|\frac{1}{z} + \frac{1}{F} - \frac{1}{f}|} \right) \quad (2.40)$$

which is exactly the traditional box filter of the defocus blur [25]. The power of an out-of-focus scene point is uniformly spread to a box of width  $AF|\frac{1}{z} + \frac{1}{F} - \frac{1}{f}|$  on the image plane. Also note that the signal is stretched according to the camera parameters. As the depth of the object changes, the degree of blur and the perspective effects also change. This is explicitly expressed in Equations 2.24 and 2.25.

An example is shown in Figure 2.9. The scene contains three non-overlapping planar objects and the camera is focused at the middle one. If the aperture size is infinite,

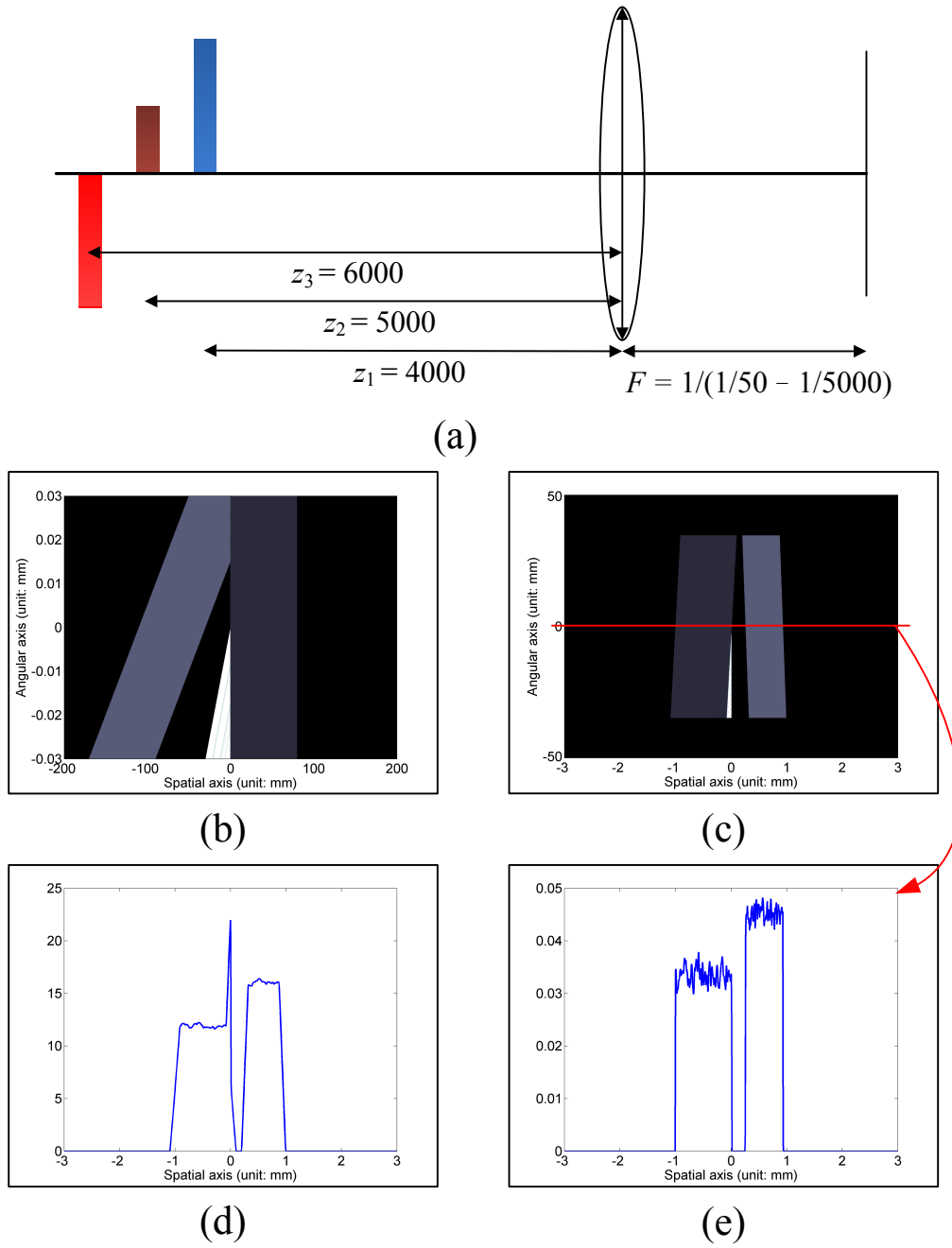
only the in-focus object will appear in the image, as shown in Figure 2.9(d). Note that this image is obtained by applying the inverse Fourier transform to the spectrum obtained using Equation 2.15.

While our derivation is performed in the frequency domain, it can also be performed in the primal domain. When an object is in-focus, its light field signal will be constant along  $v$ , and the image is simply the scaled version of the texture function. That is, no blur is introduced. When the object is out-of-focus, a sample on the texture appears slant in the transformed light field. The range of spread depends on the aperture size, and it is easy to show that the result will be identical to that obtained in the frequency domain, and thus identical to the traditional image-based analysis.

Compared to our method, the image-based approaches have several limitations. They usually assume that the scene is first projected to the image plane using the pin-hole model, and then blurred using the PSF [6]. However, the box filter is not correct at the object boundaries, where a pixel is affected by several objects with different depths, and the box filter is truncated. While there exists a model to describe this case, its analytic form is complex [26]. Additionally, for the scene points occluded in the perfect pinhole image can contribute to the defocused image, which can not be easily explained using the image-based models.

For example, suppose that the scene contains three objects, and their depths ( $z$ ) are 4000, 5000, and 6000, respectively as shown in Figure 2.10(a). The middle object is totally occluded by the near object. The camera is focused at  $z = 5000$ . We can see that in the transformed light field (Figure 2.10(c)), the middle object is present (appear white), and thus it contributes to the final image. We can see this effect clearly in Figure 2.10(d), where a peak representing the object at  $z = 5000$  is at the right boundary of the defocused frontal object.

However, if we start the analysis from the pinhole image (Figure 2.10(e)), the middle object is totally absent in the image. As a result, no matter how accurate the blur kernel is, the real image can never be obtained from the pinhole image. This observation suggests two key points. First, the analysis in the light field space is more accurate to the occlusion events. Second, if the full light field data is available, we can better



**Figure 2.10:** The effect of the occluded object to the defocused image.

(a) The light field sampled at  $z = 4000$ . (b) The transformed light field. The aperture size is 70. (c) The resulting image. (d) The pinhole image.

describe the image formulation process.

Another interesting thing is that in the pinhole camera,  $A \rightarrow 0$  and  $\widehat{\mathcal{B}}_A$  becomes a uniform function. As a result, the energy at any point is uniformly spread along  $f_v$  without any decay. Therefore, all objects are always in-focus. Changing the camera parameters  $F$  only changes the scaling factor of the image, as we have described in Section 2.4.1.

### 2.4.3 Vignetting Effect

In the previous analysis, the parameterization function  $G$  is ignored. In Equation 2.24, we can see that the effect of  $G$  varies as the image coordinate changes. Suppose that the input light field  $L_i$  is uniform, then the resulting image is

$$V(u) = \int_{-A/2}^{A/2} G\left(\frac{u-v}{F}\right) dv, \quad (2.41)$$

where  $V$  is the so-called vignetting field. In the traditional image-based method [5], it is approximated by a  $\cos^3 \theta$  decay where  $\theta$  is defined as

$$\theta = \frac{u}{F}. \quad (2.42)$$

However, this approximation is valid only when the aperture size  $A$  is infinitesimally small<sup>11</sup>. It is easy to show that when  $A \rightarrow 0$ , Equation 2.41 becomes

$$V(u) = G\left(\frac{u}{F}\right) = \frac{1}{\left(1 + \left(\frac{u}{F}\right)^2\right)^{3/2}} = \cos^3(\theta). \quad (2.43)$$

When the aperture size is finite, the vignetting field changes with  $A$ . While the analytic form is tedious, generally speaking, the vignetting field becomes smoother as the aperture size increases, and eventually becomes flat.

We can also perform the analysis in the frequency domain. When  $L_i$  is uniform, only the DC component of the spectrum is non-zero. As a result, according to Equation 2.25, only the DC component of the image will be non-zero and the image will appear uniform. However, because the light field is modulated by  $G$  before the projection

---

<sup>11</sup>In the textbook [5], the vignetting field is a  $\cos^4$  decay field in the 3D space. In the 2D space here, it becomes a  $\cos^3$  decay field.

operation, its spectrum is convolved by  $\mathcal{G}$  before the slicing operation. Since  $G$  is a function of  $u$ , the energy at the DC component will be spread along  $f_u$ , and thus there are non-zero energies at other frequency bands.

As shown in Figure 2.6,  $G$  is a very smooth function, and thus the degree of the spread will be very small. In other words, the spectrum of the vignetting field will only contain extremely low-frequency components. This conclusion matches our intuition. Again when  $A \rightarrow 0$ , the result is consistent with the image-based approach.

#### 2.4.4 Summary

We have shown that the models for several photographic effects are implicitly embedded in Equations 2.22, 2.23, 2.24, and 2.25. The perspective transform of the image is due to the photographic transformation to the light field, the defocus blur is due to the modulation of the aperture function and the slicing operation, and the vignetting effect is due to the modulation of the parameterization function. They can also be equally or better explained in the frequency domain.

It is easy to see that those effects are all functions of the camera parameters  $(f, A, F)$  and the position of the object  $z$ . Therefore, we can not adjust the degree of one effect without affecting others. On the contrary, in the existing image-based analyses, these effects are usually discussed separately. This means our method is more accurate and closer to the real device than the existing ones.

### 2.5 Spectrum Analysis of the Transformed Light Field

We have showed that the energy of the light field from a Lambertian object fall on a slice along  $f_v = 0$  in the frequency domain. The photographic operator maps this slice to a slice whose slope depends on the object distance and camera parameters. The image formation process extracts a slice along  $f_v = 0$  in the light field spectrum. Many photographic effects can be easily explained using our formulation in either primal or the dual (frequency) domain.

In this section, we turn our attention to the full spectrum of the transformed light



field, which now can be obtained by many light field cameras ([21], [27], [28] to name a few). We first show that the digital refocusing operation can be easily derived using our framework. Second, we show that the spectrum of the light field has a very unique structure and further exploit this structure for the fusion of all-focused image, and for the parameter setting of the light field cameras.

### 2.5.1 Refocusing using Light Field

In [24] it is showed that if we have a light field in the camera, we can generate images with different focus settings. This refocusing operation can be done efficiently in the frequency domain by extracting the 1D spectrum along the slice of the object to be focused<sup>12</sup>.

While the derivation in [24] focuses on the mathematical theorems and, here we describe the refocusing operation using the complete light transport work and precisely link the camera parameters and the refocusing operations together.

Equation 2.18 and 2.19 show that the light field in the camera is a modulation of a transformed light field and a aperture blocking function. An ideal refocusing system should allow the users to change all camera parameters, including  $f$ ,  $F$ , and  $A$ . Let's first focus on adjusting the first two parameters and assume the new focus length is  $af$  and the new distance between the lens and the image plane is  $bF$ . One can use backward transport to first reverse  $R_f T_F P_F$  and then apply the new forward operators to the light field:

$$\widehat{\mathcal{L}}'_r(\mathbf{f}) = \widehat{\mathcal{L}}_r((\mathbf{R}_f \mathbf{T}_F \mathbf{P}_F)^{-1}(\mathbf{R}_{af} \mathbf{T}_{bF} \mathbf{P}_{bF})\mathbf{f}), \quad (2.44)$$

where the accumulated refocusing operator is

$$\mathbf{P}_F^{-1} \mathbf{T}_F^{-1} \mathbf{R}_f^{-1} \mathbf{R}_{af} \mathbf{T}_{bF} \mathbf{P}_{bF} = \begin{bmatrix} \frac{1}{b} & 1 + F \left( \frac{1}{af} - \frac{1}{f} \right) - \frac{1}{b} \\ 0 & 1 \end{bmatrix}. \quad (2.45)$$

Therefore, given the new camera parameters, we can transform the light field using Equations 2.44 and 2.45 to transform the light field. The spectrum along  $f_v = 0$  of the

---

<sup>12</sup>In the 3D space, and light field is 4 dimensional and a 1D slice becomes a 2D hyperplane. However, the concept and derivation are similar to the 2D case we discuss here.

transformed light field would be the spectrum of the refocused image. One can also use the backward transport to transform the slicing operator to the original light field. The backward transport operator of Equation 2.45 is

$$\begin{bmatrix} b & 0 \\ 1 - b - bF\left(\frac{1}{af} - \frac{1}{f}\right) & 1 \end{bmatrix}. \quad (2.46)$$

As the analysis in Section 2.4.2, the slicing operator will become slant. Therefore, the spectrum of the image will be

$$\mathcal{J}_{r,a,b}(f_u) = \widehat{\mathcal{L}}_r \left( \begin{bmatrix} bf_u \\ \left(1 - b - bF\left(\frac{1}{af} - \frac{1}{f}\right)\right) f_u \end{bmatrix} \right). \quad (2.47)$$

When this slice overlaps with the energy of a specific object, that object would appear sharp in the image.

It should be noted that if we further assume  $a = 1$  as in [24], then the accumulated operator is

$$\mathbf{P}_F^{-1} \mathbf{T}_F^{-1} \mathbf{T}_{bF} \mathbf{P}_{bF} = \begin{bmatrix} \frac{1}{b} & 1 - \frac{1}{b} \\ 0 & 1 \end{bmatrix}, \quad (2.48)$$

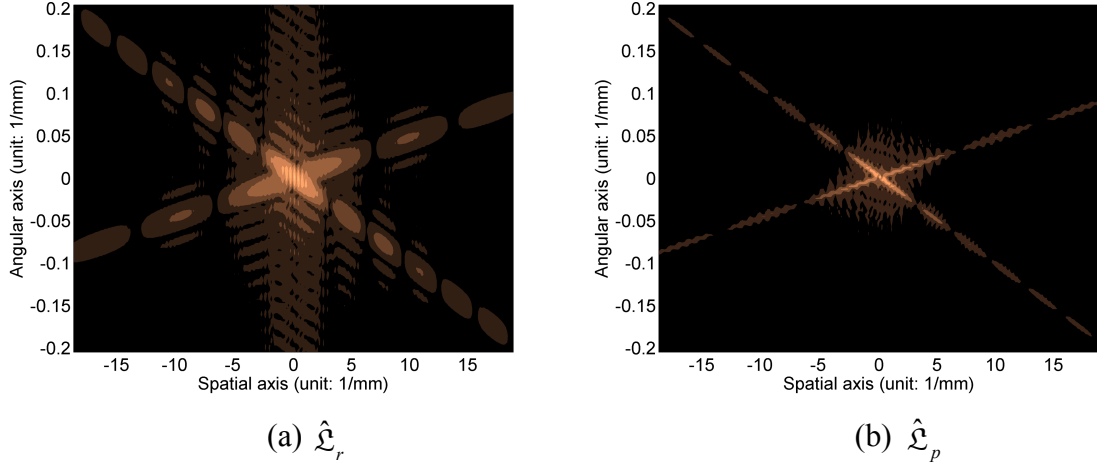
which is identical to the result in [24]. In other words, the refocusing operation in [24] is a simplified version of Equation 2.47.

Changing the aperture size is more difficult than changing other parameters. Since  $B_A$  is a binary modulation function, the blocked signals are irrecoverably lost. Therefore, one can only reduce the size of the aperture in the refocusing operation. However, since the signal of the light field is very regular, it might be possible to extrapolate the lost signals.

One can also think of this problem in the frequency domain. The blocking function convolves the light field spectrum, as described in Equation 2.23. To remove it, we can apply a deconvolution filter to the spectrum:

$$\widehat{\mathcal{L}}_p = \widehat{\mathcal{L}}_r \otimes^{-1} \widehat{\mathcal{B}}_A. \quad (2.49)$$

Since  $\widehat{\mathcal{B}}_A$  is a delta function along  $f_u$ , we can only perform the deconvolution along the  $f_v$  direction. That is, the frequency bands with different  $f_u$  coordinates can be processed



**Figure 2.11:** *The light field spectrums with and without the aperture blocking function. (a) The spectrum of the light field transformed by the photographic operator. The scene contains two objects at different depths. (b) The spectrum of the aperture-free light field.*

independently. However,  $\hat{\mathcal{B}}_A$  is approximately a low-frequency function, and thus many high frequency (along the  $f_v$  axis) contents are lost. Therefore, a trivial deconvolution is ill-conditioned, which can not perfectly recover  $\mathcal{L}_r$  even when the data is noise-free.

While it is difficult totally remove the effect of the aperture, we can do that to some degree. One particular application is to generate an all-focused image from the convolved spectrum  $\hat{\mathcal{L}}_r$ , as we describe in the following.

## 2.5.2 Fusion of All-Focused Images

Now we know that for a static scene, the images of different focus settings are equivalent to the different slices in the light field spectrum. Without loss of generality, we can assume  $a = 1$  in Equation 2.47 and have

$$\mathcal{I}_b(f_u) = \hat{\mathcal{L}}_r([bf_u, (1-b)f_u]^T). \quad (2.50)$$

Let  $f_x$  denote  $bf_u$ , we have

$$\mathcal{I}_b\left(\frac{f_x}{b}\right) = \hat{\mathcal{L}}_r\left(\left[f_x, \frac{(1-b)f_x}{b}\right]^T\right). \quad (2.51)$$

This equation means that when we change the refocus parameter  $b$ , the frequencies of the same  $f_u$  but different  $f_v$  coordinates are captured by different refocused images.

The aperture blocking function spreads the energies of the light field along  $f_v$ . Assume there are two objects at two different depths  $z_1$  and  $z_2$ , and their initial light fields defined at the object surface are  $L_1$  and  $L_2$ . Let  $\hat{L}_{p,1}$  and  $\hat{L}_{p,2}$  denote the light field transformed using the photographic operators  $\hat{\mathbf{C}}_{z_1 fF}$  and  $\hat{\mathbf{C}}_{z_2 fF}$ , respectively. Then according to Equation 2.23, we have

$$\hat{\mathcal{L}}_r(\mathbf{f}) = (\hat{\mathcal{L}}_{p,1}(\mathbf{f}) + \hat{\mathcal{L}}_{p,2}(\mathbf{f})) \otimes \hat{\mathcal{B}}_A. \quad (2.52)$$

An example is illustrated in Figure 2.11(a). The combined spectrum is convolved with the aperture blocking function and thus blurred. Our goal is to recover the spectrum unaffected by the aperture function, as shown in Figure 2.11(b).

According to Equation 2.37, their energies will fall on two lines with different slopes in  $\hat{\mathcal{L}}_p$ . If we set  $b_1 F = (f^{-1} - z_1^{-1})^{-1}$  and  $b_2 F = (f^{-1} - z_2^{-1})^{-1}$ , and apply them to Equation 2.51, we have

$$\mathcal{J}_{b_1} \left( \frac{f_x}{b_1} \right) = \hat{\mathcal{L}}_r \left( \left[ f_x, \frac{(1-b_1)}{b_1} f_x \right]^T \right), \quad (2.53)$$

$$\mathcal{J}_{b_2} \left( \frac{f_x}{b_2} \right) = \hat{\mathcal{L}}_r \left( \left[ f_x, \frac{(1-b_2)}{b_2} f_x \right]^T \right). \quad (2.54)$$

We can see that by properly stretching spectrums of the refocused images, their frequency components map to those with identical  $f_u$  coordinate and different  $f_v$  coordinates.

According to our analysis in Section 2.4.2, we know that because of the convolution of  $\hat{\mathcal{B}}_A$  along  $f_v$ , the defocused object is present in the image. The decay of the energy depends on the  $f_v$  coordinate of the mapped slice in  $\hat{\mathcal{L}}_r$ . In the case of refocused image, the analysis is still valid with slight modifications. In image  $I_{b_1}$ , the object at  $z_1$  is well-focused, and vice versa. The decay of the energy now becomes the difference of the  $f_v$  coordinates of these two slices. Let  $b_{12}$  denote  $(1-b_1)b_1^{-1} - (1-b_2)b_2^{-1}$ , and by Equations 2.38, 2.52, Equations 2.53 and 2.54 become:

$$\mathcal{J}_{b_1} \left( \frac{f_x}{b_1} \right) = \hat{\mathcal{L}}_{p,1} \left( \left[ f_x, \frac{(1-b_1)}{b_1} f_x \right]^T \right) + \hat{\mathcal{L}}_{p,2} \left( \left[ f_x, \frac{(1-b_2)}{b_2} f_x \right]^T \right) \text{sinc}(Ab_{12}f_x), \quad (2.55)$$

$$\mathcal{J}_{b_2} \left( \frac{f_x}{b_1} \right) = \hat{\mathcal{L}}_{p,1} \left( \left[ f_x, \frac{(1-b_1)}{b_1} f_x \right]^T \right) \text{sinc}(Ab_{12}f_x) + \hat{\mathcal{L}}_{p,2} \left( \left[ f_x, \frac{(1-b_2)}{b_2} f_x \right]^T \right). \quad (2.56)$$

We can re-write these two equations together in a matrix form. Removing all frequency indices in  $\mathcal{J}$  and  $\hat{\mathcal{L}}$  for simplicity, we have:

$$\begin{bmatrix} \mathcal{J}_1 \\ \mathcal{J}_2 \end{bmatrix} = \begin{bmatrix} 1 & \text{sinc}(Ab_{12}f_x) \\ \text{sinc}(Ab_{12}f_x) & 1 \end{bmatrix} \begin{bmatrix} \mathcal{L}_{p,1} \\ \mathcal{L}_{p,2} \end{bmatrix} = \mathbf{M}_{b_{12}} \begin{bmatrix} \mathcal{L}_{p,1} \\ \mathcal{L}_{p,2} \end{bmatrix}. \quad (2.57)$$

Therefore, we can see that when there are two objects at different depths, the refocused images, or the equivalent samples in the light field after the convolution of the blocking function, are the linear combination of the aperture-free light field. As a results, as long as  $\mathbf{M}_{b_{12}}$  is invertible, we can recover the aperture-free light field, which is impossible to capture using a physically limited device.

Whether  $\mathbf{M}_{b_{12}}$  is invertible depends on  $f_x$  and  $b_{12}$ . When  $f_x = 0$ , the condition number of  $\mathbf{M}_{b_{12}}$  is infinite, which means we cannot recover the individual DC components of two objects from two refocused images. The conditional number decreases as  $f_x$  and  $b_{12}$  increase. This means that it is easier to separate two light fields at high frequencies, or when the distances between these two objects are large. However, at a reasonable spectrum resolution,  $\mathbf{M}_{b_{12}}$  is always invertible except at the DC term.

If we only attempt to generate an all-focused image, we can perform the inversion of Equation 2.57 at all frequencies but the DC term. Then we can combine the recovered  $\mathcal{L}_{p,1}$  and  $\mathcal{L}_{p,2}$ , and take the DC term of either  $\mathcal{J}_1$  or  $\mathcal{J}_2$ . The spectrum generated in this way contains both  $L_1$  and  $L_2$ , and thus the image is all-focused.

The analysis we present can be easily generalized to multiple objects. The matrix  $\mathbf{M}$  in Equation 2.57 would become  $N \times N$  when dealing with  $N$  objects at different depths, but the condition number still follows our analysis: higher frequencies lead to easier inversion and the individual DC terms can never be recovered. Several examples can be found in Figures 2.17 and 2.18.

In [29] an image-based approach with a similar goal is derived. However, its derivation is painstaking because the relation we showed in the light field domain cannot be easily characterized in the image domain. The algorithm in [29] requires several deconvolution passes and has convergence issues. On the contrary, our method is very easy

to explain in the light field spectrum domain. Different frequency bands (with different  $f_u$  coordinates) can be solved independently, each by a single matrix multiplication.

### 2.5.3 Parameter Settings for Efficient Light Field Sampling

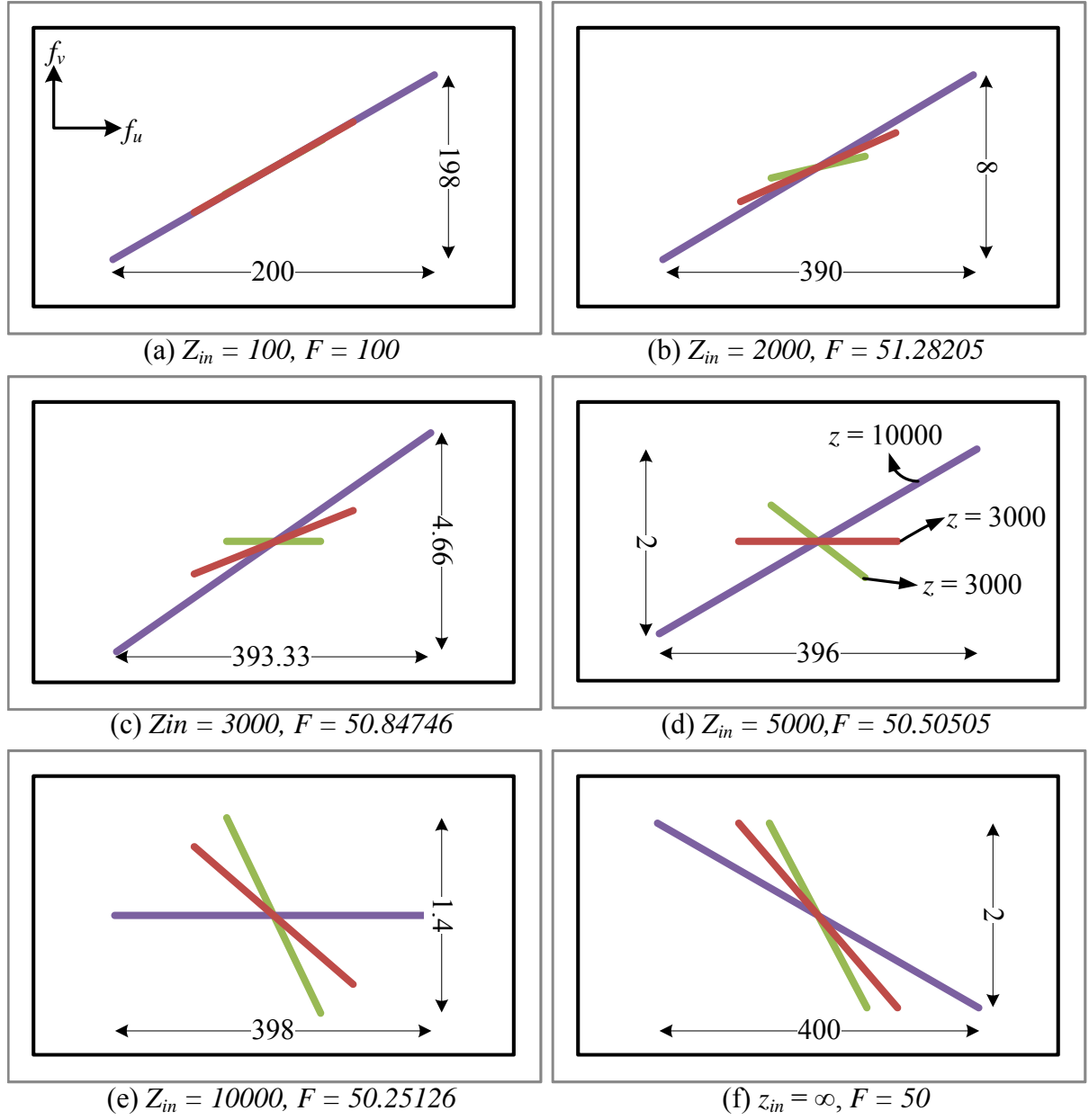
The spectrum of the transformed light field has several unique properties. First, for Lambertian scenes, the energies concentrate on a 1D slice. Second, the slices of different objects have different slope in the transformed light field as shown in Equation 2.37. Here we discuss how to specify the camera parameters to efficiently sample the light field.

We use a simple example to start our analysis. Suppose that the scene contains three object at  $z_1 = 3000$ ,  $z_2 = 5000$ , and  $z_3 = 10000$ , respectively. We also assume the bandwidth of the texture functions on those objects are the same. We set  $f = 50$  and vary the focus setting  $F$  to obtain the transformed light field, as illustrated in Figure 2.12.

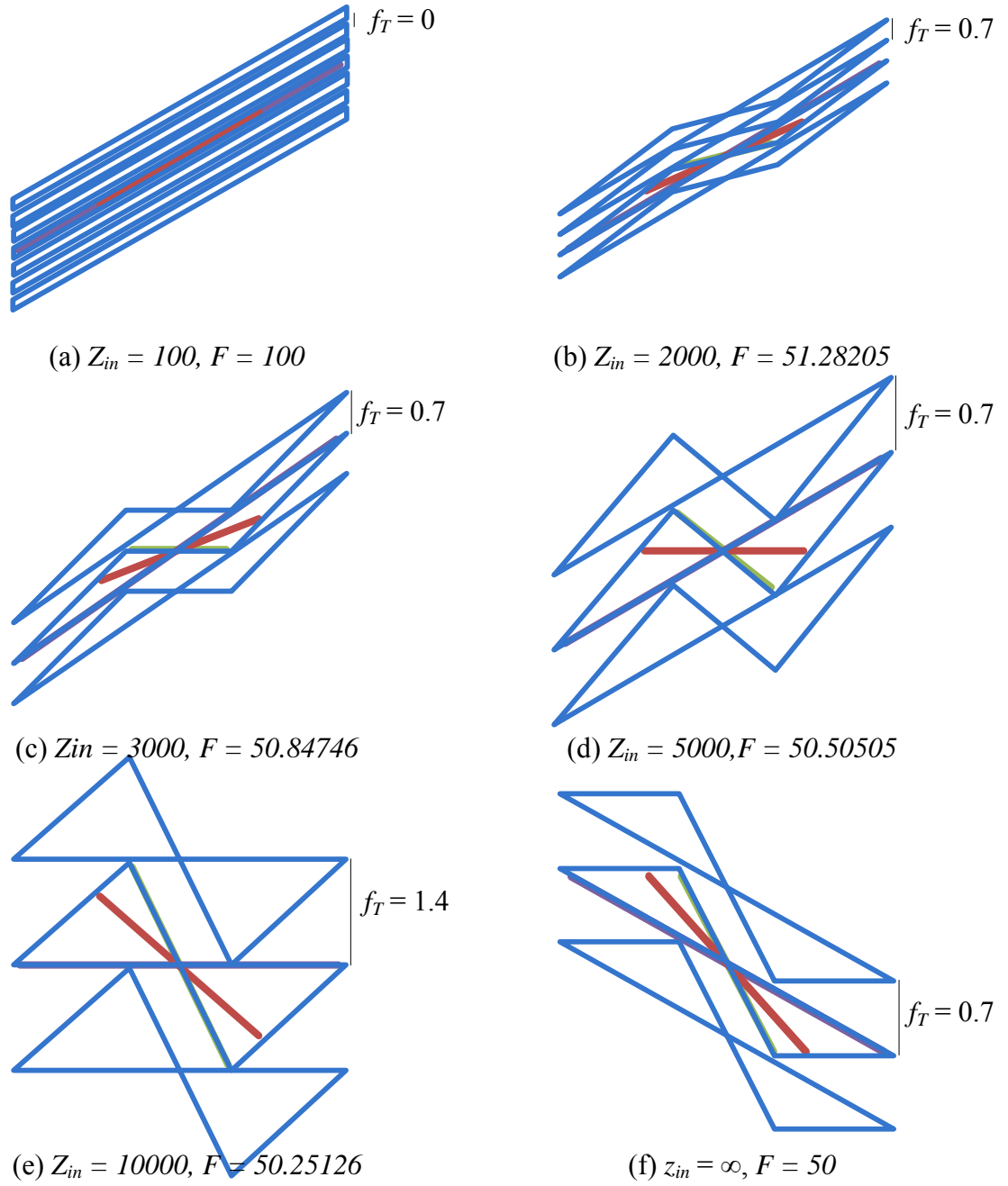
We can see that when  $F$  is specified such that  $z_{in} = z_i$ , the energy slice of the corresponding object falls on  $f_v = 0$ , and thus it can be perfectly recorded by the slicing operator given in Equation 2.25. Also note that the Nyquist sampling rate along the  $f_u$  axis depends on the focus setting. When the in-focus object is the nearest one (Figure 2.12(c)), the sampling rate can be smaller than the case when the in-focus object is the farthest one ( $z_{in} = z_3$ ). This is because a specific frequency component on the far object is mapped to a higher frequency in the light field spectrum transformed by the photographic operator than that on the near object.

For light field cameras, we also have to sample along the  $v$  axis (the aperture plane). According to the sampling theory, the Nyquist rate is twice of the bandwidth. At the first glance, the spectrum in Figure 2.12(e) has the minimal bandwidth along  $f_v$  and thus requires minimal sampling rate.

However, the shape of the light field spectrum is very unique. We can see that it is like a laid sandglass (two symmetric wedges sharing a single vertex at the origin), as shown in Figure 2.13. If we measure the *effective bandwidth* of the spectrum and perform the sampling, we can see that the required bandwidth is much smaller than

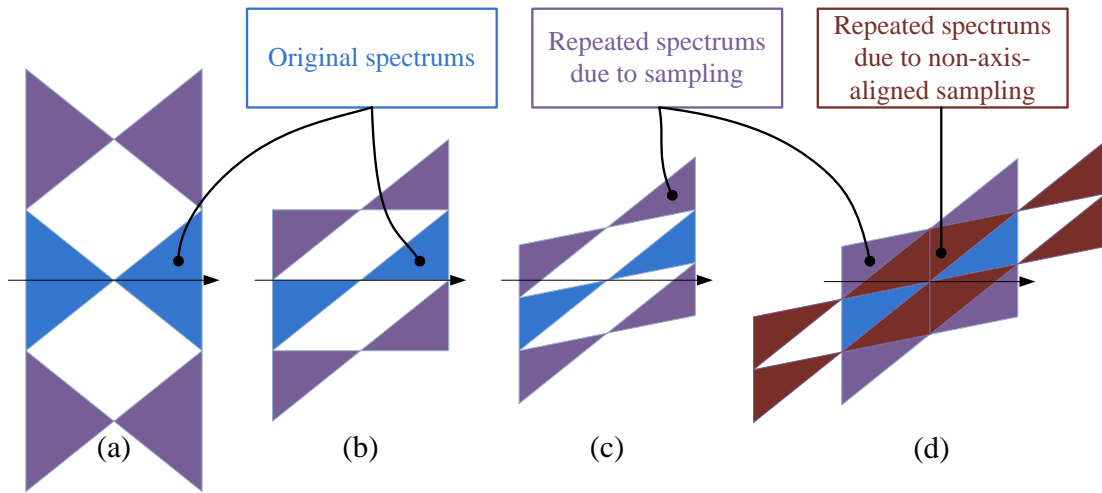


**Figure 2.12:** The spectrum of the light fields under different focus settings. The camera setting are given with the indices, and the bandwidth along  $f_u$  and  $f_v$  are given in the Figures.



**Figure 2.13:** The minimal sampling rates along the  $f_v$  axis.





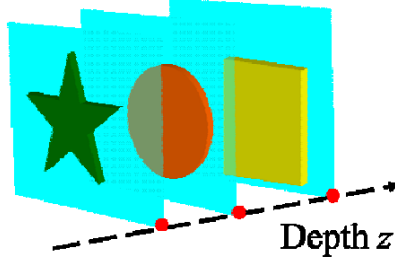
**Figure 2.14:** *The relation of the sampling rate and the focus settings.*

*The perform sampling along the  $v$  axis and thus the spectrum of the light field would repeat along the  $f_v$  axis. (a) The focus plane is within the depth range. (b) The focus plane is at the most frontal object. (c) The focus plane is in front of all objects. (d) Higher spectrum packing using non-axis-aligned sampling.*

the Nyquist rate. In this case, the spectrum in Figure 2.12(e) on contrary requires the highest sampling rate. When the camera is focused extremely close to the lens, all slices will overlap together, and the Nyquist rate becomes zero. However, this situation may be undesirable for many applications, such as digital refocusing.

The analysis above assumes the bandwidth of the surface texture is limited. In real case, the texture function is usually unlimited. The general rule is *adjust the focus plane to be extremely close to the lens or to infinity*, as illustrated in Figure 2.14. When the focus plane is set within the depth range of the scene, the energies are distributed in all quadrants in the frequency space, and thus the minimal effective bandwidth is at most half of the bandwidth. However, once the focus plane is set outside the depth range, the energies are always within two of four quadrants. As a result, the efficient bandwidth can be much smaller. The price of the efficient sampling is that a non-axis-aligned reconstruction filter must be applied. However, for many applications such as digital refocusing, reconstruction of the full light field is unnecessary.

If the non-axis-aligned sampling scheme is allowed, one can utilize the sandglass



**Figure 2.15:** *Definition of the depth detection problem.*

*Three objects located at three planes perpendicular to the  $z$  axis (optical axis of the camera). Our algorithm finds the depths of these planes as marked by the three dots.*

structure to achieve a even tighter energy packing, as shown in Figure 2.14(d). However, for the light field cameras, especially for the proposed programmable aperture camera in the next Chapter, the available sampling rate along the  $u$  axis is usually very high, and hence this complex sampling scheme is unnecessary.

While the efficient sampling of the light field has been presented in [11], here we successfully link it to the camera parameters. A similar conclusion about the parameter setting according to the complex geometric optics is given in [30].

## 2.6 Depth Detection of Light Field

We have showed that the energies of the objects at different depths are located at different 1D slices in the 2D frequency domain. Also these slices can be extracted by the refocusing operator defined in Section 2.6.4. For the slice without object, it only contains the energies spread from the slices with objects due to the convolution of  $\hat{\mathcal{B}}_A$  as described in Equation 2.23.

Because  $\hat{\mathcal{B}}_A$  has only one global peak at the original, all spread energies are always smaller than original ones. Therefore, the powers of the slices containing object should be much higher than those containing no object. This effect is particularly obvious in the high frequency bands since  $\hat{\mathcal{B}}_A$  decays very fast.

In the following we develop an efficient depth detection algorithm based on this concept. Instead of finding the depths of different regions in the scene, we extract the

depths of different planes at which objects are located. In other words, for a scene with objects located at  $(x, y, z)$ 's, we find the  $z$ 's of the objects without solving for their  $x$  and  $y$  positions. Figure 2.15 illustrates this idea. As described later, the proposed method is very useful for many applications. This work was first presented in [31]<sup>13</sup>.

Our proposed method is quite different from traditional auto-focus (AF) algorithms in two ways. First, AF algorithms only determine a best focus value for a specific region, but our algorithm detects all depths. Second, AF algorithms are image-based, which usually use several heuristics to measure the sharpness of the image. On the contrary, since we have a completed 4D light field so we can estimate the depths more precisely.

### 2.6.1 Overview

We say that the *focusness* of a depth is high if there is some object located at this depth. We first describe the proposed measurement for the focusness. This measurement is a weighted energy ratio of different frequency bands in the spectrum of the refocused images. It can be evaluated easily from the light field spectrum.

However, objects such as grounds or forests may spread over a wide range of depths in a real scene. Therefore, we re-formulate our objective function to address this issue.

### 2.6.2 Focusness Measurement

We directly extend our analysis above to the 3D cases and the coordinates on the lens plane and the image plane are  $(u, v)$  and  $(x, y)$ , respectively. Therefore, according to Equations 2.47 and 2.48, we have<sup>14</sup>:

$$\mathcal{I}_b(f_x, f_y) = \mathcal{L}([(1-b)f_x, (1-b)f_y, bf_x, bf_y]^T). \quad (2.58)$$

When there is an object located at the plane of depth  $z = (f^{-1} - bF^{-1})^{-1}$ , the high

---

<sup>13</sup>This possible utilization of the light field analysis was mentioned in [24] as a *Fourier range finder* but was never developed in the past.

<sup>14</sup>The subscript  $r$  and widehats are removed below for simplicity since the real data is always modulated by  $B_A$

frequency component of this spectrum should be larger than those of the other slices which have no object in focus.

To reduce the computation, only the energy along  $f_x$  and  $f_y$  axes is taken into account. That is, we extract two 1D spectrums from  $\mathcal{J}_b$ :

$$\begin{aligned}\mathcal{J}_b^x(f_x) &= \mathcal{L}([(1-b)f_x, 0, bf_x, 0]^T), \\ \mathcal{J}_b^y(f_y) &= \mathcal{L}([0, (1-b)f_y, 0, bf_y]^T),\end{aligned}\tag{2.59}$$

and then calculate the power spectrum  $P_b(f)$ :

$$P_b(f) = (\mathcal{J}_b^x(f))^2 + (\mathcal{J}_b^y(f))^2.\tag{2.60}$$

The direct summation of  $P_b(f)$  over the whole spectrum (total energy) is not a good measurement of focusness for mainly two reasons. First, changing  $b$  does not much alter the energy of low frequency components. Only the details (high frequencies) are lost when the object is out-of-focus. Second, noise and aliasing may dominate the energy at high frequencies near the Nyquist rate.

To alleviate these problems, we use a multi-band processing method. The power spectrums  $P_b(f)$  is split equally into 8 bands. Denote the energy in these 8 bands by  $E_0(b), E_1(b), \dots, E_7(b)$ . The high bands  $E_1$ - $E_6$  are emphasized with proper weighting, and their summation are normalized by the lowest band  $E_0$ . The highest band  $E_7$  is ignored since it contains mostly noise. Denote the measurement of focusness by  $G(b)$ :

$$G(b) = \frac{1}{\log E_0(b)} \sum_{i=1}^6 w_i \log E_i(b).\tag{2.61}$$

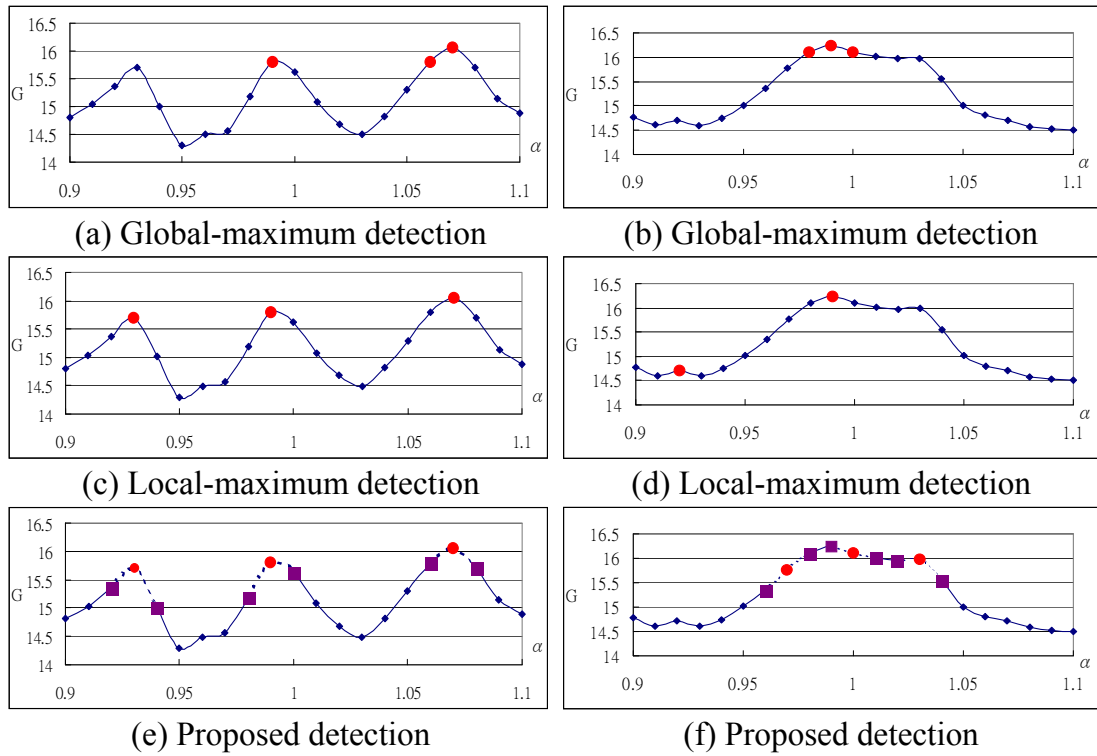
We tried many different settings of  $w_i$  and found that  $w_i = i$  gives the best results.

### 2.6.3 Detection as Optimization

Figure 2.16(a) and (b) show the plots of  $G$  over  $b$  for our two synthetic datasets, which are designed for two extreme cases. It is obvious that  $G(b)$  is high when there is some object at depth  $b^{15}$ . However, simple local or global maximum detection may not result in good selection.

---

<sup>15</sup>We can related  $b$  to the real depth using the lens equation.



**Figure 2.16:** *Detected depths of the two datasets using different methods. (a)(b) global-maximum, (c)(d) local-maximum, and (e)(f) our proposed algorithm. The circles denote the selected depths, the squares are the neighboring depths, and the dotted segments are the partially focused regions.*

Figure 2.16(a) shows the  $G$  of dataset 1 with completely discrete depth distribution. The objects in this space are located at three planes with  $b = 0.93, 0.99$ , and  $1.07$ . The curve shows three local peaks exactly at these points. On the other hand, Figure 2.16(b) shows the  $G$  curve of dataset 2 with a continuous distribution of depths. The objects spread a range of depths from  $b = 0.97$  to  $1.03$ . The  $G$ 's in this range are globally higher than those in others, but there is only one peak.

These two extreme cases reveal that simple depth-detection algorithms solely based on local-or global-maximum detection would not work. That is, if we take local maximums as the output, we can succeed in the discrete-depth case, but fail to handle the continuous-depth case. On the contrary, the global maximum detection works well in the continuous-depth case, but not in the discrete depth one.

Note that when there is an object at  $b$ ,  $G(b)$  is larger than its neighbors. In addition, the neighboring  $G$ 's are also affected by this object. This effect should be taken into account. Instead of finding global or local maximums, we try to solve the following maximum-cover problem:

**Maximum-Cover Problem:** *Given depths  $b_1, b_2, \dots, b_N$ , and corresponding  $G$  factors  $G_1, G_2, \dots, G_N$ , find  $K$  indexes  $D_1, D_2, \dots, D_K$  such that*

$$\sum_{i=1}^K (\lambda G_{D_i-1} + G_{D_i} + \lambda G_{D_i+1}) \quad (2.62)$$

*is maximized, under the constraint that the selected  $D_i$  are separated by at least 3. The parameter  $\lambda$  is between 0 and 1.*

The constraint is to ensure that the neighbors of every selected depth do not overlap to the neighbors of other selected depths. In our experiments the  $\lambda$  is set to 0.5 and  $K$  is set to 3. This problem can be solved efficiently by dynamic programming.

## 2.6.4 Experimental Results

We first perform experiments using the previous synthetic datasets so the exact depths of the objects are known. The resolution for these dataset is  $16 \times 16 \times 256 \times 256$ . For Equation 2.59, spectrum is re-sampled by a Kaiser-Bessel filter with width 2.5. For Equation 2.62,  $N$  is set to 21, corresponding to  $b = 0.90, 0.91, \dots, 1.10$ .

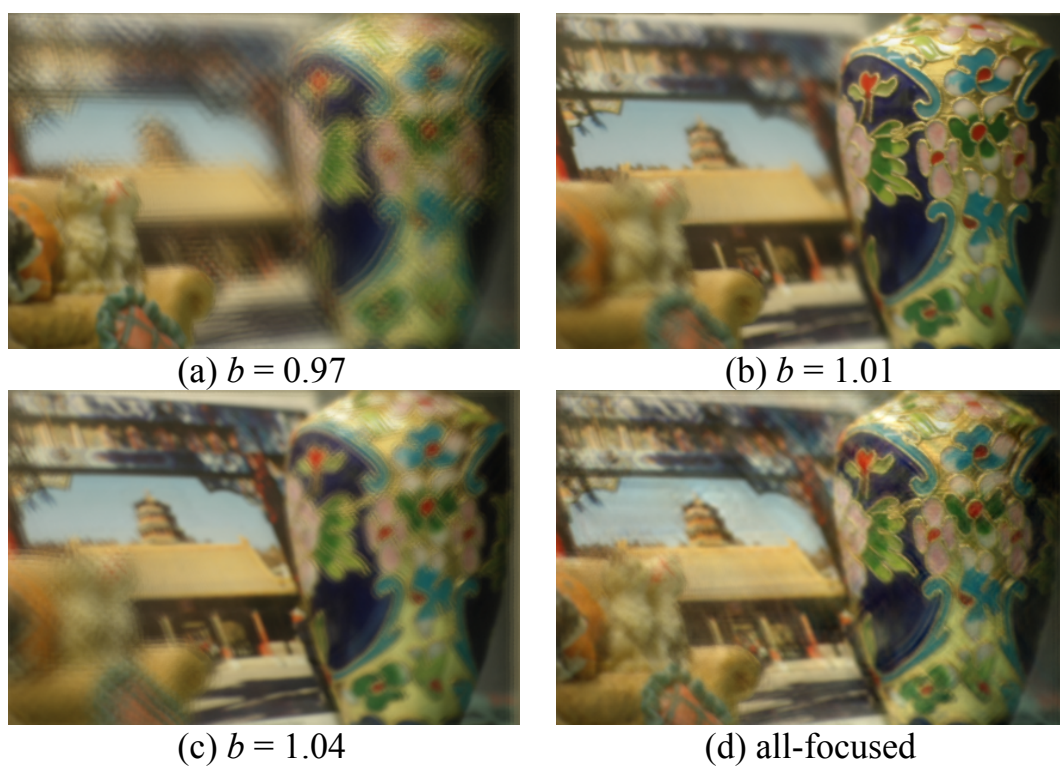


**Figure 2.17:** *The refocused and all-focused images from the synthetic datasets.  $b$  denotes the detected depths and all-focused images are generated from the synthetic aperture photographs without human interaction.*

Then we perform similar experiments using real dataset captured by programmable aperture camera described in Chapter 3. The depth distribution of this dataset is neither completely discrete nor completely continuous. Instead, it is composed of an object located at  $b=0.97$  and a region of objects through  $b=1.01$  to  $b=1.04$ . The resolution of this dataset is down-sampled to  $4 \times 4 \times 256 \times 256$ .

The resulted  $G$  curves of the two synthetic datasets are shown in Figure 2.16. The global-maximum method fails when the depth distribution is discrete, since the object at 1.07 also pulls up  $G(1.06)$ , as shown in Figure 2.16(a). On the other hand, the local-maximum method fails when the depth distribution is continuous, as shown in Figure 2.16(d). There are objects distributed from 0.97 to 1.03, but only  $G(0.99)$  is a local peak. For both cases, our proposed algorithm works well.

Using the detected depths, we can automatically generate the synthetic aperture photos, as shown in Figure 2.17 (a-c) and 2.17 (e-g). The results on real dataset are presented in Figure 2.18. Our algorithm successfully selects the depths where objects are located at.



**Figure 2.18:** *The refocused and all-focused images from the real dataset.*



Using the fusion technique presented in Section , we can generate all-focused images, as shown in Figure 2.17 (d) and (h), and Figure 2.18 (d). The little ghosting effect is due to aliasing. Thanks to the proposed depth detection algorithm, we do not have to manually specify the object depths as it was in the analysis. We also do not have to resolve the per-pixel depth maps to fuse the refocused images. All operations can be efficiently done in the frequency domain.

### 2.6.5 Complexity

Calculating the  $G$  factor for each depth takes  $O(S)$ , where  $S$  is the width (or height) of the image. Calculating  $N$  factors takes  $O(NS)$ . Our selection algorithm takes  $O(NK)$ , so the total time cost is  $O(NS + NK)$  and dominated by  $O(NS)$ . Compared with the FFT for obtaining  $\mathcal{L}$ , which takes  $O(S^4 \log S)$ , and the IFFT for refocused photo, which takes  $O(S^2 \log S)$ , the complexity of our algorithm is negligible.

In our experiment, 4D FFT takes 10 seconds, but we can pre-calculate it only once. For some scenario where the 4D FFT is redundant or undesirable, we can use a different approach to evaluate Equation 2.59. In  $\mathcal{J}^x$  and  $\mathcal{J}^y$ , two of four dimensions are simply DC components, which can be easily extracted by the projection operator. Therefore, we can generate two 2D signals  $I_{ux}$ ,  $I_{vy}$  by projecting the 4D signal along  $v, y$  and  $u, x$  axes respectively.  $\mathcal{J}^x$  and  $\mathcal{J}^y$  can be obtained from the 2D FFT of  $I_{ux}$  and  $I_{vy}$ .

Our experiments were performed on a PC with Pentium-4 3GHz CPU and 1GB RAM. Generating each refocused image takes 1 second and the all-focused image takes 3 seconds. Our depth detection only takes 0.25 seconds.

## 2.7 Discussion

In this section we discuss the possible extensions and applications of the present light transport framework.

### 2.7.1 Extension to the 3D Space

While our analyses are performed in the 2D space, it is straightforward to extend them to the 3D space (where the light field becomes four dimensional). For certain operators such as surface shading, the Fourier transformation must be replaced with the spherical harmonics [14]. Fortunately, most operators we use, including light ray traveling, lens refraction, and reparameterization, can be easily mapped to the 4D light field by simply including two additional coordinates to the equations. One coordinate defined on the  $\mathbf{U}$  plane is orthogonal to the  $u$  axis and the other defined on the  $\mathbf{V}$  plane is orthogonal to the  $v$  axis. Therefore, we can use our framework to map the light field into the camera. Also, because the range of the angular axis is seriously truncated by the aperture blocking function, unlike the case in the camera array that spans a large angular range, we can assume the scene is Lambertian for a single camera (or light field camera).

An interesting thing is that in practice, the camera parameters can be different in different axes. For example, the focal length of the lens may be direction-variant, and thus the refraction Equation 2.6 becomes:

$$L_1 \left( \begin{bmatrix} u \\ v \\ s \\ t \end{bmatrix} \right) = L_0 \left( \begin{bmatrix} 1 & 0 & 0 & 0 \\ f_u^{-1} & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & f_s^{-1} & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ s \\ t \end{bmatrix} \right), \quad (2.63)$$

where  $s$  and  $t$  are the new coordinates in the 4D space and  $f_u$  and  $f_s$  are the focal lengths along the  $u$  and  $s$  axes, respectively. As a result, when an object is in-focus in one direction, it would be out-of-focus in others. This is actually the basic model for the astigmatic effect. In fact, the variation of each entity in Equation 2.63 or in the overall photographic operator maps to a specific effect in the optics system. For example, if  $F$  is different in two axes, the image plane is skewed. A mathematical framework of these variations is presented in [32], which does not link the equations with the physical property of the camera like we do. Combine our analysis with that in [32], it may be possible to design new optics for novel imaging applications.

### 2.7.2 New Aperture Design

We have shown that the effect of the aperture is the spread of energies in the frequency domain. While for the traditional aperture, the spread energies are always decayed, it may be possible to design new aperture to amplify the energies. For example, if  $\hat{\mathcal{B}}(\mathbf{f}) = f_v^2 \delta(f_u)$ , then all frequency components are always amplified in the image spectrum. However, the aperture has its physical limitation, such as it can neither negative the light rays nor amplify them.

One can on the other hand change the objective. For example, in [21] the low-pass aperture function is replaced with an all-pass one so the deconvolution becomes well-posed. In [33] the aperture is designed such that the PSF for each depth is very distinct from those for others. Other application-specific designs could possibly be found using our framework.

### 2.7.3 New All-Focused Fusion Method

We have presented an all-focused fusion method and a new depth-detection method. While both of them can be efficiently in the frequency domain, it may be possible to combine them into a single one.

When the angular sampling rate is high, the number of the possible depths is always much fewer than the angular samples. Also, the samples of the aperture-free light field axis is very sparse along the  $f_v$ . Therefore, one may re-formulate Equation 2.57

$$\mathbf{L}_p = \arg \min (\mathbf{L}_r - \mathbf{M}\mathbf{L}_p)^2 + \lambda |\mathbf{L}_p|^p, \quad (2.64)$$

where  $\mathbf{L}_r$  are the frequencies in the observed light field of a specific  $f_u$  band,  $\mathbf{L}_p$  are the corresponding frequencies in the aperture-free light field, and  $\mathbf{M}$  is the convolution matrix generalized from Equation 2.57. We use the  $L_p$  norm to enforce the sparseness of the  $\mathbf{L}_p$ <sup>16</sup>. Therefore, if this objective function can be minimized, we can simultaneously solve the depth detection and all-focused problems. To make it more robust, the inter-frequency continuity could be considered.

---

<sup>16</sup>For the definition of the sparseness, please refer to [34].

## 2.8 Summary

In this chapter we have presented a complete framework for analyzing the light transport effect in photography. We have shown that the photographic effects result from the combination of the transformation and modulation of the light field, and these operations can be expressed by a single equation. All the operations can be easily mapped to the frequency domain, and the photographic effects can therefore be explained in a new way.

We have applied the spectrum analysis of the light transport to many applications, including digital refocusing, fusion of all-focused images, and parameter setting for the light field cameras. A novel depth-detection algorithm based on the analysis is also proposed. It can efficiently find possible object planes without performing per-pixel depth estimation.

The proposed framework explains many photographic effects and has many potential applications beyond the ones we have presented in this chapter. We hope it can inspire more researches in light transport analysis and image formulation.

# Chapter 3

## Light Field Acquisition

In the chapter, we propose a complete to acquire high quality light field data. After a brief review of the previous work, we describe a novel device, called programmable aperture, for light field acquisition. The programmable aperture is much more flexible and inexpensive than the previous devices. It can capture the light field with the spatial resolution identical to the sensor resolution, and with an adaptive angular resolution.

We then present two new post-processing algorithms to address inherently distortions. The first algorithm can automatically estimate and calibrate the photometric distortion of the light field. The second algorithm can infer the scene geometry from the light field<sup>1</sup>.

### 3.1 Previous Work

The programmable aperture is inspired by several previous techniques for computational photography and light field acquisition. Here we describe the most relevant ones. Please refer to [36], [37], [38], [39], [40] for the complete survey.

#### 3.1.1 Light Field Acquisition

Light field is a four dimensional signal which cannot be directly captured by a 2 dimensional sensor array. The simplest method moves a single camera over a 2D manifold

---

<sup>1</sup>The content in this chapter was presented in [35], [28].

to capture the light field sequentially. The position of the camera on the manifold can be controlled precisely using a camera gantry [3], or estimated from the captured images by structure-from-motion (SFM) algorithms [4], [41]. This method is very slow because the position of the camera or the object has to be altered before each exposure. Either gantry or the SFM algorithm only works well in a controlled environment.

The second method captures the dynamic light field by putting numerous cameras into a camera array [42], [43], [44]. Each camera can capture a subset of the light field at video-rate. Although this method can acquire high quality light field, the camera array and its back-end control and storage machine are very cumbersome and expensive.

The third method, which is most related to ours, inserts an *optical light ray reparameterizer* into the camera to avoid the angular integration operation we mentioned in Chapter 2. This idea dates back nearly a century ago, termed *integral photography* or *parallax panoramagrams*, and are realized using either a fly-eye array [45] or a slit plate [46], [47].

Recently compact implementation and theoretical analysis were paid much attention. In a *plenoptic camera*, for example, a microlens array is placed at the image plane inside the camera [48], [27]. The resulting tiny image behind each microlens records the angular resolution of the light rays, while in the integral photography, the image behind each microlens records the spatial distribution. Veeraraghavan et al. replaced the split plate with a cosine-mixture mask to improve the light-collection efficiency [21]. They also showed that this operation is in fact a modulation of the light field, and that the *heterodyned* light field can be recovered by demodulation. Using the light transport theory, the reparameterizers inside the camera can be replaced by those outside the camera with equivalent effects [49], [22]. This can possibly make a better tradeoff between the spatial and angular resolutions.

However, no matter which reparameterizer is applied, the principle of those devices are the same: manipulating the 4D light field spectrum by modulation or transformation to make it fit in a 2D sensor slice. Therefore, all these devices share the following drawbacks. First, the spatial resolution, or the spectrum bandwidth along the spatial axis, is traded from the angular resolution. To capture a 3 megapixel photo with reasonable

angular resolution, like 5-by-5, would require a sensor array with 75 million sensors, which is still very difficult, if not impossible, to make in the near future.

Second, inserting masks or optical elements in a camera automatically imposes a fixed sampling pattern. The angular and spatial resolutions cannot be changed unless the component is replaced. These components are usually permanently installed and cannot be easily removed from the camera to capture regular images. Third, the reparameterizers inside the camera must be made accurately at micrometer scale. Therefore they are usually very expensive and require elaborate calibration<sup>2</sup>.

### 3.1.2 Coded Aperture Imaging

The coded aperture technique is one of the most useful tools in computational photography. In the beginning, this technique was used to improve the data quality for X-ray astronomy [50], [51]. Then it is found that many other applications can be achieved by modulating the light rays in a controlled way using the coded aperture. For example, the exposure time can be modulated by a coded aperture to preserve the high spatial frequencies in the motion blurred photo and make the deblurring process well-posed [52]. Similar concept is also applied to remove the refocused blur [21]. One can also capture high-dynamic-range or multi-spectral images with spatially variant exposure control [53], [54]. The field-of-view of the image can be split by using a volumetric light attenuator [55] and the stereoscopic images can be captured by a single static camera [56]. A color-filter aperture can be used to generate inter-channel parallax cue in a single exposure [57].

One particularly relevant work uses a coded aperture to improve the performance of the depth-from-defocus algorithm [33]. The pattern on the aperture is designed such that the ambiguities between the blur kernel of different scales are minimized. It is demonstrated that the rough geometry of a near-Lambertian scene can be estimated from a single image. In contrast, our method directly captures the complete 4D light field and estimates the depth from it when necessary.

Several concurrent projects discussed in Section 3.2.6 also use the concept similar

---

<sup>2</sup>For example, the microlens array used in [27] costs US \$1500 dollars.

to our programmable aperture.

### 3.1.3 Multiple-Exposure Imaging

Another techniques commonly adopted in computational photography is the multiple-exposure technique. The basic concept is to capture the scene several times sequentially, or simultaneously by using beam splitters and camera arrays. At each exposure a subset of the imaging parameters, such as lighting or viewpoint, are made different. Then a novel image or other additional information (e.g., alpha matte), is obtained by computation. Some applications include depth-edge detection and NPR rendering [58], matting estimation [59], flash/non-flash image enhancement [60], [61], noise/blurred image restoration [62], and so forth.

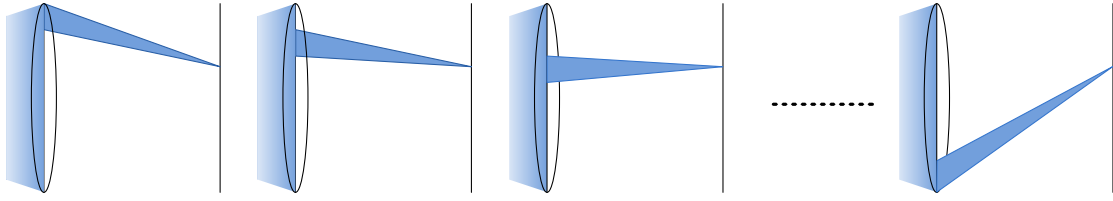
It should be noticed that due to the rapid exposure ability of the digital sensors, many techniques mentioned above can be easily implemented on the commercial cameras. For example, Senkichi et al. split a given exposure time into a number of time steps and sampled one image in each step. These images are then registered into a single clean, blur-free image [63].

### 3.1.4 Illumination Multiplexing

Capturing and analyzing the appearance of a scene under different lightings is an active topic for image-based rendering and computer vision. Since the dimension of the signal (a 4D incident light field or an 8D reflectance field) is much higher than that of the sensor (a 2D photo sensor array), the signal must be captured sequentially, one subset at a time. One can *multiplex* several subsets of the signal together in each exposure, and then recover the individual subsets by computation [64], [65].

The proposed programmable aperture makes it possible to capture a subset of the light field inside the camera at each exposure. Therefore, the multiplexing technique can be naturally introduced and thus the signal-to-noise ratio of the captured light field can be significantly increased.





**Figure 3.1:** Sequential light field acquisition using the programmable aperture.

## 3.2 Programmable Aperture

In a traditional camera one can only adjust the size of the aperture to adjust the depth of field (DOF) of the photo. However, the captured photo is always a 2D projection of the 3D scene, or the 2D projection of the 4D light field. That is, the angular information (in our case, the signal variation on the aperture plane) is irrecoverably lost.

However, if we modify the *shape* of the aperture so that only the light rays arriving in a small specified region of the aperture can pass through the aperture, we can avoid the angular integration. More specifically, if the aperture blocks all light rays but those around a specific point  $v_0$ , the transmittance of the aperture  $T(v)$  can be described as

$$T(v) = \begin{cases} 1 & \text{if } \|v - v_0\| < w \\ 0 & \text{otherwise} \end{cases}, \quad (3.1)$$

where  $w$  is the width of the aperture. Put this into Equation 2.24, we have

$$I_{v_0}(u) = \int_{-\infty}^{\infty} L(u, v) T(v) dv = \int_{v_0-w}^{v_0+w} L(u, v) dv. \quad (3.2)$$

We can see that if  $w \rightarrow 0$ ,  $I(u) = L(u, v_0)$ , and the captured image is a subset of the light field. If  $w > 0$ , the captured image is equivalent to a subset of the filtered light field. Specifically, the light field is filtered by a box filter of width  $w$ , and then sampled by the sensor array. In the following, we call  $I_v$  as a *light field image*.

### 3.2.1 Sequential Light Field Acquisition

By capturing images with different aperture shapes, as shown in Figure 3.1, we acquire a complete light field. Note that unlike previous devices that manipulate the light rays

after they enter the camera [27], [21], [22], our method blocks all undesirable light rays in the very beginning and leaves a subset of light field to be sampled. Thus the spatial resolution of the light field is the same as the sensor resolution. For the method to take effect, a *programmable aperture* is needed. Its transmittance  $T(v)$  has to be spatially variant and controllable.

One may think that the programmable aperture can be trivially implemented by replacing the lens module with a volumetric light attenuator [55]. However, according to our light transport analysis in Chapter 2, removing the lens would significantly increase the bandwidth of the light field along the angular axis and thus we need to capture more images. In other words, the lens can bend the light rays and consequently compress the spectrum. Therefore, by carefully selecting the in-focus plane and lens module, we can properly reshape the spectrum to reduce aliasing, as described in Section 2.5.3. That is also the reason why most light field cameras preserve the lens module.

### 3.2.2 Light Field Multiplexing

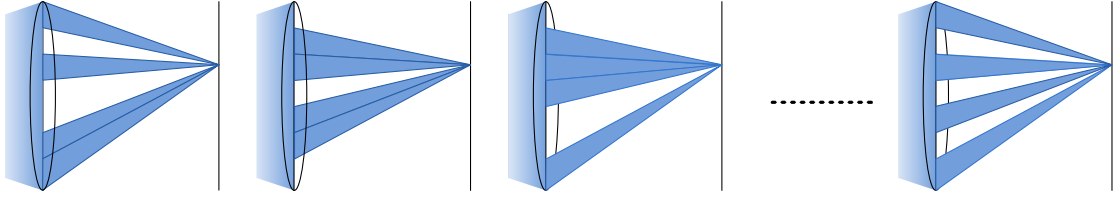
A light field with angular resolution  $N_a$  requires  $N_a$  exposures, one for each  $v_i$ . Compare with traditional photography, the light-collection efficiency is decreased because only a small aperture is open at each exposure (the size of the aperture should be less than the sampling grid) and each exposure time is only  $1/N_a$  of the total acquisition time. As a result, given the same acquisition time, the captured images are noisier than those captured by conventional cameras.

To solve this problem, we *multiplex* the light field images at each exposure. Specifically, one can aggregate multiple light field samples at each exposure by opening multiple regions of the aperture. Because the radiances of the light rays are additive, the individual signals can be recovered afterwards.

At each exposure, the captured image  $M_v$  is a linear combination of  $N_a$  light field images (Figure 3.2):

$$M_v(u) = \sum_{k=0}^{N_a-1} w_{vk} I_k(u), \quad (3.3)$$

where  $I_k(u)$  is defined in Equation 3.2 and  $w_{vk}$  is the weight of each light field images in the current exposure. Due to the physical limitation, we cannot amplify the power



**Figure 3.2:** *Multiplexing the light field by using the programmable aperture.*

of light rays or generate a negative light rays, and thus  $w_{vk} \in [0, 1]$ . The weights for a single exposure can be represented by a vector  $\mathbf{w}_v = [w_{v0}, w_{v1}, \dots, w_{v(N_a-1)}]$  and is referred to as a *multiplexing pattern* since  $\mathbf{w}_v$  is physically realized as a spatially variant mask on the aperture plane. After  $N_a$  exposures with  $N_a$  different multiplexing patterns, we can recover the light field by *demultiplexing* the captured images. Specifically, all captured samples at the same pixel location can be represented by a  $N_a$ -D vector  $\mathbf{m}(u) = [M_0(u), M_1(u), \dots, M_{N_a-1}(u)]$  and similarly all true light field signals at the spatial index  $u$  can be represented by  $\mathbf{i}(u) = [I_0(u), I_1(u), \dots, I_{N_a-1}(u)]$ , then we have

$$\mathbf{m} = \mathbf{W}\mathbf{i}, \quad (3.4)$$

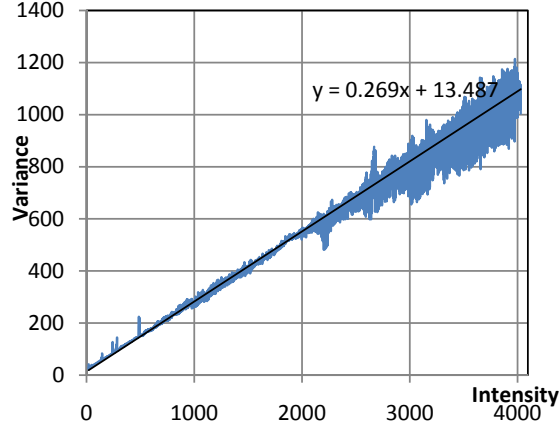
where  $\mathbf{W}$  is an  $N_a \times N_a$  matrix and each row of  $\mathbf{W}$  is a multiplexing pattern  $\mathbf{w}_v$ . The demultiplexing process recovers  $\mathbf{I}$  by

$$\mathbf{i} = \mathbf{W}^{-1}\mathbf{m}. \quad (3.5)$$

We can see that  $\mathbf{W}$  must be invertible to make the demultiplexing feasible.

By intuition, we should one as many as regions as possible (i.e., maximize  $\|\mathbf{w}_v\|$ ) to allow the sensors to gather as much light as possible. In practice, however, noise always involves in the acquisition and complicates the design of the multiplexing patterns. In the case where the noise is independent and identically-distributed (i.i.d.), Hadamard-code based patterns are best in terms of the quality of the demultiplexed data [66], [64], [35].

However, for digital sensors, including CCD and CMOS, the noises are often correlated with the input signal and the camera parameters [67], [68]. For example, the variance of the *shot noise* grows linearly with the number of the incoming photons and the variance of the *dark current noise* grows linearly with exposure time. In this case,



**Figure 3.3:** *The noise characteristics of Nikon D70 DSLR.*

*The variance of the noise grows linearly with the intensity. The blue points are the observed variances and the black line is the fitted noise model.*

using the Hadamard-code based patterns actually degrades the data quality [65]. Another drawback of the Hadamard-code based patterns is that they only exist for certain size (i.e., only certain  $N_a$ ).

Instead, we search for the optimal multiplexing patterns according to the noise characteristics of the camera. According to [66], the mean-square-error (MSE) of the recovered signal from demultiplexing (Equation 3.5) is:

$$MSE = \frac{\sigma^2}{N_a} \text{Trace}((\mathbf{W}^T \mathbf{W})^{-1}), \quad (3.6)$$

where  $\sigma^2$  is the variance of the noise in the measured signal ( $\mathbf{m}$ ). The variance depends on the input signal and the camera parameters and can be approximately modeled by two components. The first one  $\sigma_c^2$  is a constant value and the second one  $\sigma_p^2$  is proportional to the received irradiance of the sensor. Let  $\sigma_0^2$  be the variance of the second noise when the received irradiance value is one unit.

$$\sigma^2 = \sigma_c^2 + \sigma_p^2 = \sigma_c^2 + P\sigma_0^2. \quad (3.7)$$

Both  $\sigma_c^2$  and  $\sigma_0^2$  are measured as follows. We fix the camera in front of a LCD screen. The screen display an achromatic intensity ramp (both devices are linearized). We then capture 20 images of the screen and take the average of those images as the ground truth image. The variances of the signals of different intensities are then obtained and then

$\sigma_c^2$  and  $\sigma_0^2$  are estimated by a simple least-square fitting. One result is shown in Figure 3.3. Given the model, the MSE becomes

$$MSE = \frac{\sigma_c^2 + P\sigma_0^2}{N_a} \text{Trace}((\mathbf{W}^T \mathbf{W})^{-1}). \quad (3.8)$$

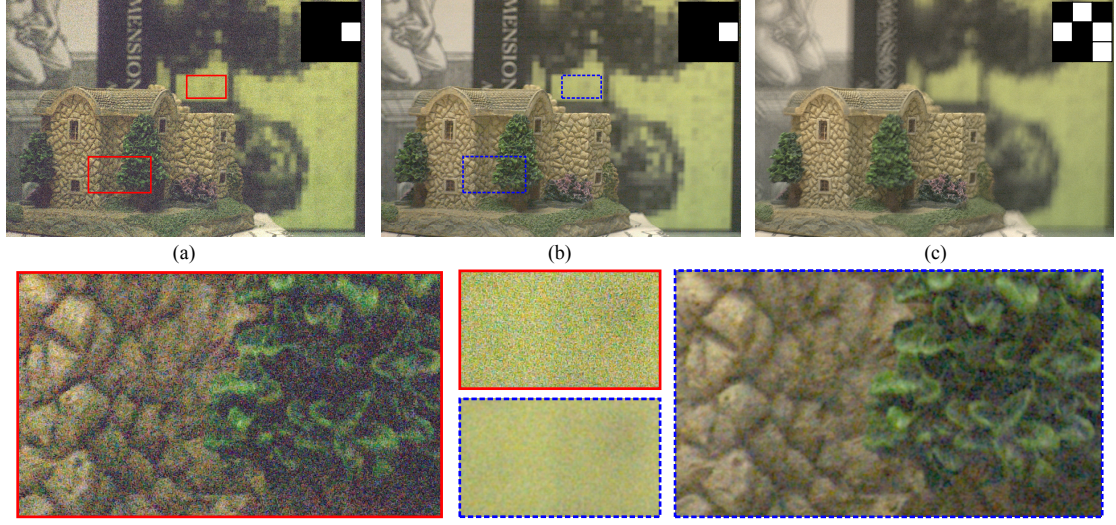
The intensity of the multiplexed signal  $\mathbf{m}$  depends on not only the multiplexing pattern  $\mathbf{W}$  but also the real light field signal  $\mathbf{i}$ . Since the latter term is dynamic, we fix it as the middle of the dynamic range. We also assume the effective transmittance of the multiplexing pattern is fixed (i.e.,  $\|\mathbf{w}_v\|$  is a constant  $\|\mathbf{w}\|$  for all  $v$ ). Then the MSE will only depend on the chosen multiplexing patterns:

$$MSE = \frac{\sigma_c^2 + \|\mathbf{w}\|\sigma_0^2}{N_a} \text{Trace}((\mathbf{W}^T \mathbf{W})^{-1}). \quad (3.9)$$

The optimal multiplexing patterns should minimize Equation 3.9. There are two methods to perform the minimization. When the angular sampling rate is small (like less than 20) and the patterns are binary (i.e.,  $w_{vk} \in \{0, 1\}$ ), the optimal  $\mathbf{W}$  can be obtained by a simple brute-force search. The solution obtained is guaranteed to be globally optimal under the given constraints and assumptions.

When the desirable angular sampling rate is large, since the constraints are convex and Equation 3.9 is differentiable, the minimization can be formulated as a *constrained convex optimization* problem and solved by the *projected gradient method*, as described in [69]. We surprisedly find that most entities in the multiplexing patterns thus obtained are very close to zero or one, and therefore we enforce them to be binary. This is because the binary masks can be made more accurately in practice. The MSE is only slightly affected (less than 0.1%).

The exact quality improvement due to multiplexing depends on many factors, such as the light field signal of interest. Therefore the MSE of the demultiplexed is spatially and angularly variant. However, on average, the MSE is reduced by a factor of  $O(\sqrt{N_a})$  by using the optimal multiplexing. Two results are shown in Figure 3.4 and 3.5. We can clearly see the demultiplexed light field is much better than the one captured without multiplexing.



**Figure 3.4:** Performance improvement by multiplexing (1/2).

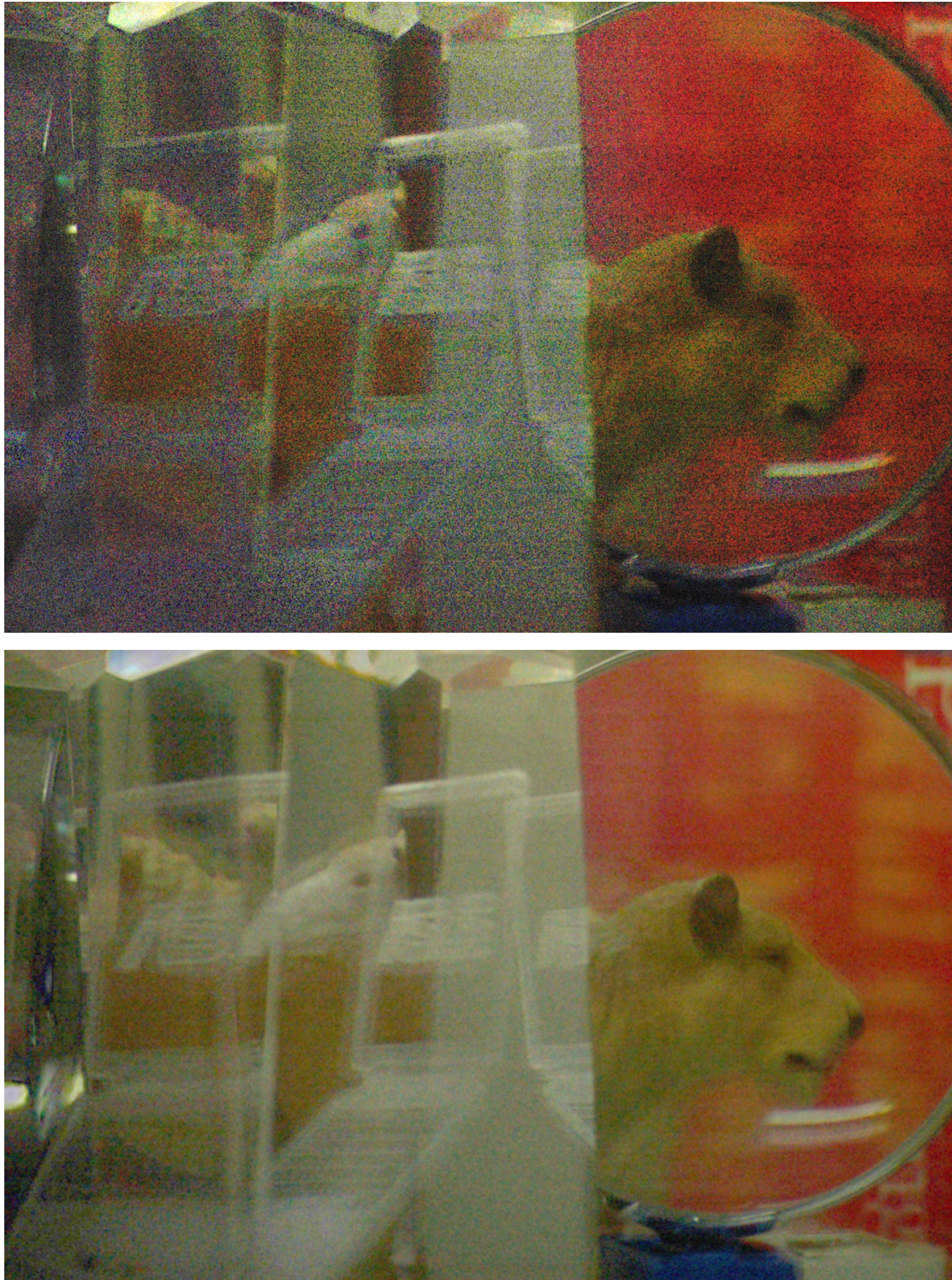
(a) Light field image captured without multiplexing. (b) Demultiplexed light field image. (c) Image captured with multiplexing. The insets in (a-c) show the corresponding multiplexing patterns. (Bottom row) Close-up of (a) and (b). The angular resolution is 3 by 3 and the total exposure time is 90ms for both cases.

### 3.2.3 Prototypes

The programmable aperture can be implemented in several ways. In this subsection we describe three prototypes we have built in chronological order. The prototypes are all installed into a Nikon D70 DSLR camera to capture the data used in experiments. We use a Nikkor 50mm f/1.4D lens module. For simplicity, we dismount the lens module from the camera and insert the programmable aperture in between them. Hence the distance between the lens and the sensor is lengthened and the focus range is shortened as compared to the original camera. However, as we shall see later, this problem can be easily solved because the second and third devices can be made small enough to place into the normal camera.

The first prototype is called *rotary panel*, as shown in Figure 3.6. It is made of a circular transparent plastic plane of the radius 139mm. The designed multiplexing patterns are manually pasted on the panel. There are 16 multiplexing patterns can be placed on a single rotary panel. All the pieces are aligned using the positioning mechanisms, which can translate and rotate independently. This prototype works fine but has some

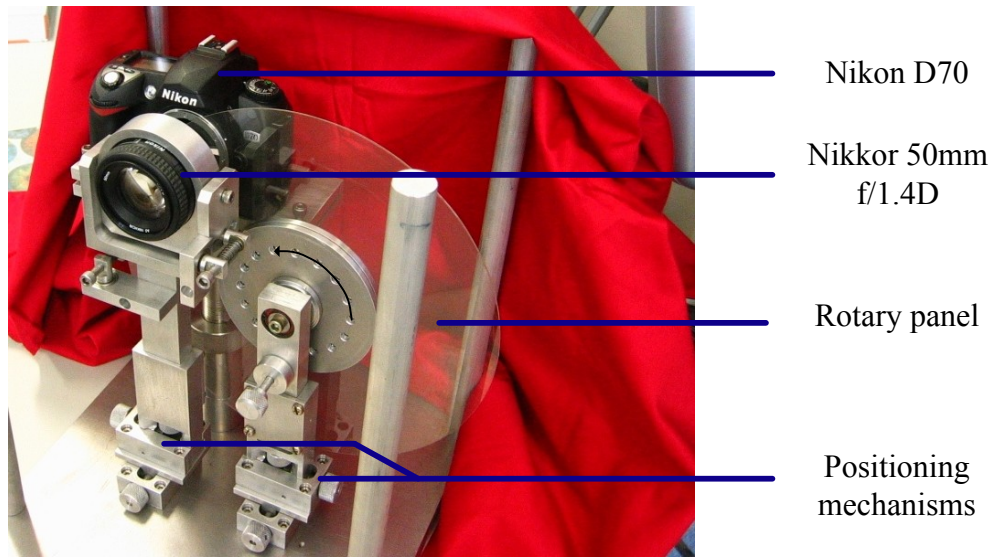




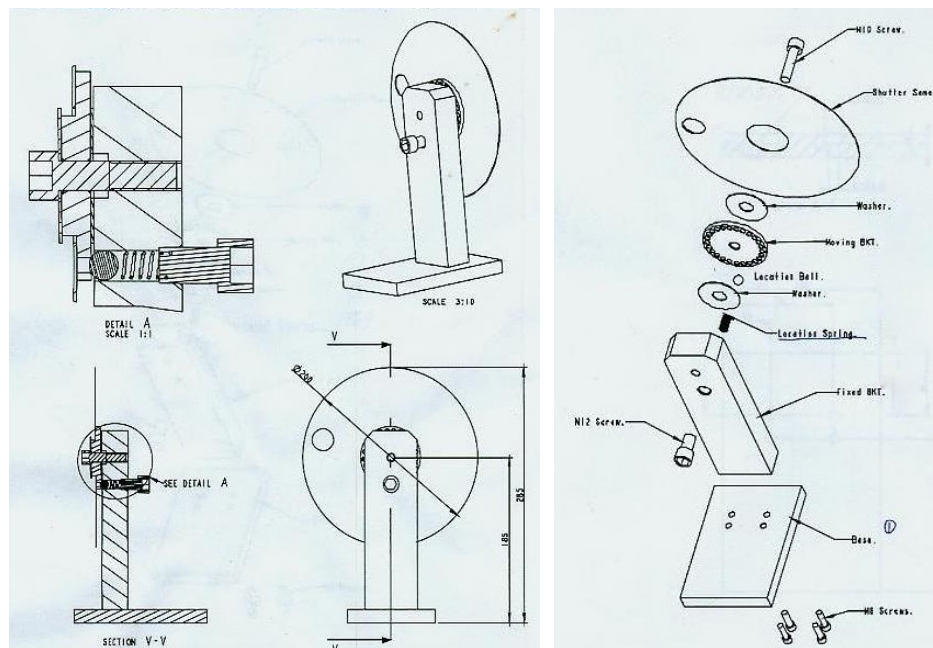
**Figure 3.5:** *Performance improvement by multiplexing (2/2).*

*(Top) Light field image captured without multiplexing. (Bottom) The corresponding demultiplexed light field image. The angular resolution is 5 by 5 and the total exposure time is 500ms for both cases.*





(a)

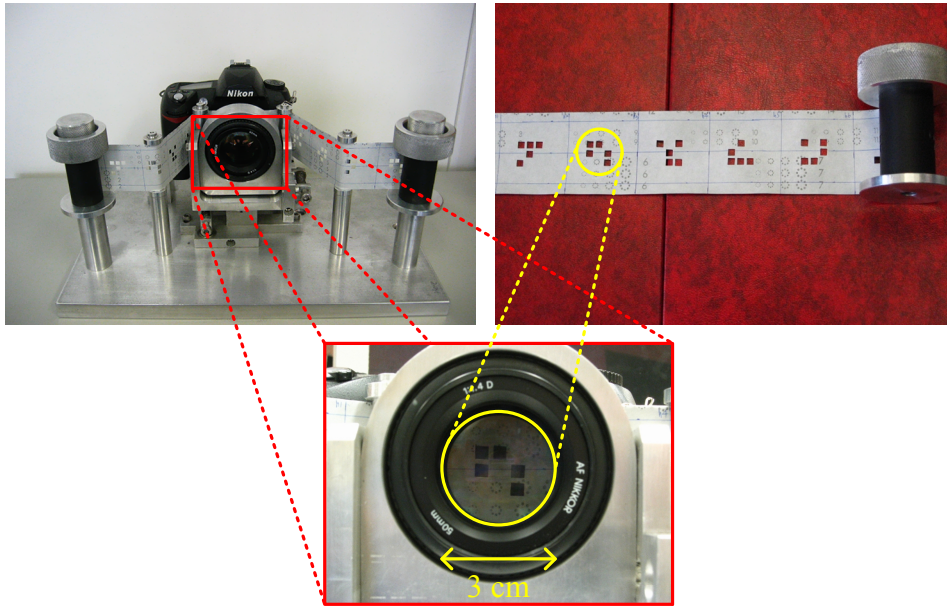


(b)

**Figure 3.6:** The first prototype: rotary panel.

(a) The photo of the prototype in action. (b) The schematic diagrams of the rotary panel.





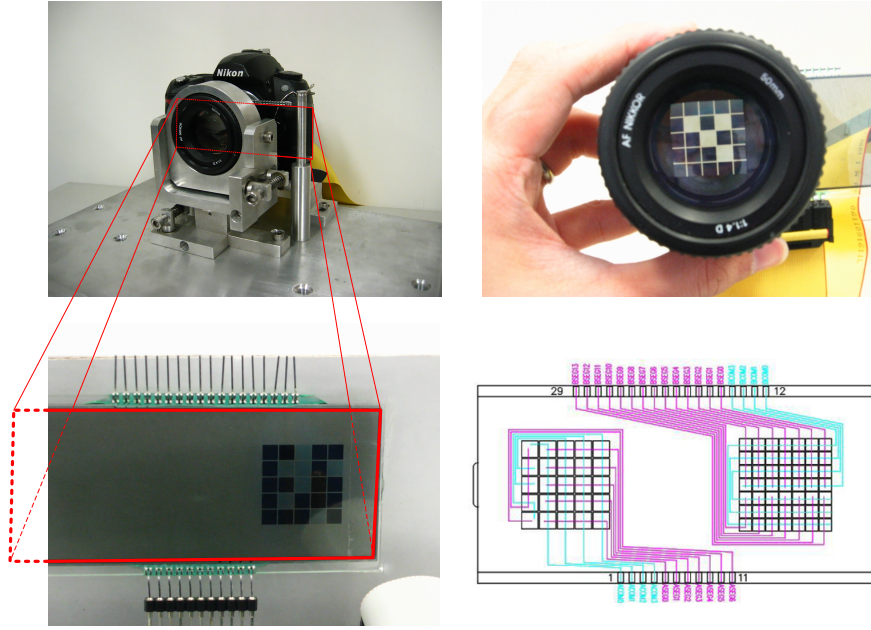
**Figure 3.7:** *The second prototype: pattern scroll.*

drawbacks. First, the rotary panel is huge and cannot be inserted into the camera body. Second, the registration of the multiplexing patterns on the circular moving object is difficult. As result, this prototype works at best a proof of concept.

The second prototype is called *pattern scroll*, which is an opaqued slit of paper. The paper we choose was originally used for film protection and thus has a extremely low transmittance. The aperture patterns are manually cut and scrolled across the optical path. The scroll is long enough to include tens of multiplexing patterns or traditional aperture shapes. This quick and dirty method is simple and performs well. Because the transition between multiplex patterns can be very fast, it can easily capture high quality multiplexed data. The cost for the paper is less than 3 US dollars<sup>3</sup>.

There are few minor issues of the pattern scroll. First, the block cell ( $w_{vk} = 0$ ) cannot stay on the pattern scroll if it loses support. We solve this problem by leaving a gap between each cell. Second, in operation, there could be some drifting errors in the position of the patterns. In our hand-made device, the errors are less than 0.5mm and do not seriously affect the acquisition. In rare cases, these errors cause a narrow, halo-like effect around the strong edges in the demultiplexed images. However, those

<sup>3</sup>Actually the 3 dollar is for the film (Ilford FP4 plus 125) because we cannot buy the paper alone



**Figure 3.8:** *The third prototype: liquid crystal array.*

effects can be easily identified and removed. Also we believe the drifting error would not be a problem to the industrial manufacturers.

The third prototype, as shown in Figure 3.8, is made up of a liquid crystal array (LCA) controlled by a Holtek HT49R30A-1 micro control circuit that support C language. The LCA is customized by a local company and has two different resolutions:  $5 \times 5$  and  $7 \times 7$ . The LCA is much easier to program and mount than the previous two prototypes, and the multiplexing pattern is not limited to be binary. The price for each piece of the LCA is 0.7 US dollars, and the control circuits cost roughly 3.5 US dollars.

However, the light rays can leak from the liquid crystal cells that cannot be completely turned off (i.e.,  $T(v)$  is always non-zero) and from the gaps (used for routing) in between the cells. In our experiments we compensate the leakage by capturing an extra image with all liquid crystal cell turned off and subtracting it from all other images. One can also put this image into the demultiplexing process and the linear system (Equation 3.5) becomes over-determined. However, this method costs more computation and we did not observe any noticeable quality improvement from it.

We use the Nikon Capture software to control the camera. All the images are saved

Device	Aperture size	#shot	Single exposure duration	SNR of the light field samples	SNR of the refocused image	Angular $\times$ spatial resolution
Conventional camera with small aperture	$A$	1	$T$	$\frac{P}{\sqrt{\sigma_0^2 + \sigma_c^2}}$	–	$1 \times M^2$
Conventional camera with large aperture	$N_a^2 A$	1	$T$	$\frac{N_a^2 P}{\sqrt{N_a^2 \sigma_0^2 + \sigma_c^2}}$	–	$1 \times M^2$
Plenoptic camera	$N_a^2 A$	1	$T$	$\frac{N_a^2 P}{\sqrt{N_a^2 \sigma_0^2 + \sigma_c^2}}$	$\frac{N_a^3 P}{\sqrt{N_a^2 \sigma_0^2 + \sigma_c^2}}$	$N_a^2 \times (M^2 / N_a^2)$
Plenoptic camera with $N_a^2 M^2$ sensors	$N_a^2 A$	1	$T$	$\frac{P}{\sqrt{\sigma_0^2 + \sigma_c^2}}$	$\frac{N_a P}{\sqrt{\sigma_0^2 + \sigma_c^2}}$	$N_a^2 \times M^2$
Programmable aperture camera	$A$	$N_a^2$	$T / N_a^2$	$\frac{N_a^{-2} P}{\sqrt{\sigma_0^2 / N_a^2 + \sigma_c^2}} (S_1)$	$\frac{N_a^{-1} P}{\sqrt{\sigma_0^2 / N_a^2 + \sigma_c^2}}$	$N_a^2 \times M^2$
PAC with multiplexing	$\approx N_a^2 A / 2$	$N_a^2$	$T / N_a^2$	$\approx N_a S_1 / 2$	$\approx N_a^2 S_1 / 2$	$N_a^2 \times M^2$

**Table 3.1:** *Performance comparison between the conventional camera, the plenoptic camera, and the programmable aperture camera.*

in linear raw format<sup>4</sup>. The original image pipeline in the Nikon D70 contains several non-linear operations, such as de-noising and tone mapping, which may corrupt the linearity of the signal and make it difficult to evaluate the performance of our post-processing algorithms. Therefore, we only perform per-channel linear scaling for white balance and simple demosaicing. These two steps can be done either before or after demultiplexing.

### 3.2.4 Performance Comparison

An important question is, despite the visual plausibility, how to quantitatively compare our programmable with other light field cameras, and even with the traditional camera? Unfortunately, we have no access to other light field cameras and the results reported in previous literatures were all based on different criterions.

Here we make a theoretical analysis based on the noise model we just described (Eq. 3.7). Instead of comparing with all existing light field cameras that follow the same

---

<sup>4</sup>The Nikon raw format is not fully public. Here we use dcraw to convert it to the raw binary data format.

principle, we argue that because no light rays is blocked or attenuated in the plenoptic camera [48], [27], it should be superior to other mask-based light field cameras [46], [47], [21]. Therefore, we focus on the comparison between the programmable aperture, the plenoptic camera, and the traditional camera in the follow analysis.

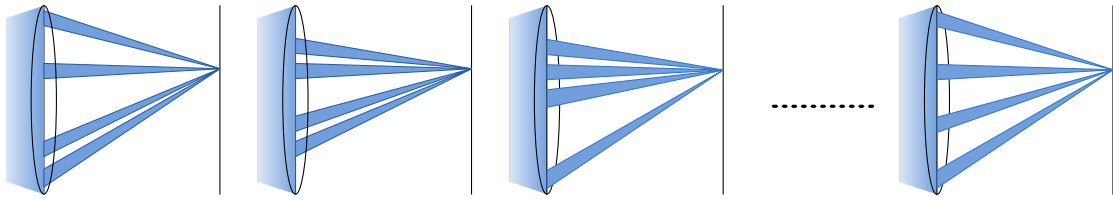
Without loss of generality, we assume the default number of sensors in these devices is  $M^2$ , and the angular resolution for capturing a single database is  $N_a^2$ . The total exposure duration for capturing a single dataset is fixed. Therefore, each exposure time in our device is  $1/N_a^2$  of the total exposure.

We measure the signal-to-noise ratio (SNR) of these devices and list the results in Table 3.1. The image captured by a conventional camera with a large aperture has the best quality, but it has a very shallow depth of field and thus requires precise focus setting. A light field image is equivalent to the image captured by a conventional camera with a small aperture and thus its quality is worse than the formal case. An interesting thing is that although light rays emitted from an in-focus scene point are recorded by  $N_a^2$  light field samplers, when the digital refocusing is performed, these  $N_a^2$  are averaged and the noises are suppressed. Therefore the SNR of the refocused image is increased by a factor of  $N_a$  and becomes competitive to the regular images.

The plenoptic camera is slightly better than the programmable aperture camera at the same angular and spatial resolution. Nevertheless, in that case it requires  $N_a^2 M^2$  sensors. To capture a light field of the same resolution as the programmable aperture camera, the plenoptic camera would require an array of nearly 100M sensors, which would be difficult and expensive, if not impossible, to make.

### 3.2.5 Discussion

The proposed light field acquisition scheme does not require a high resolution sensor. Therefore, it can be implemented on web cameras, cell-phone cameras, surveillance cameras, etc. The image captured by previous light field cameras must be decoded before visualization. In contrast, the image captured by our device can be directly displayed. Even when the multiplexing is applied, the in-focus region remains sharp, as shown in Figure 3.4 (c).



**Figure 3.9:** *Reducing the aperture size without affecting the angular resolution and the multiplexing.*

Another advantage of the programmable aperture is that the sampling grid and the pre-filtering kernel are decoupled. Therefore, the aperture size ( $w$  in Equation 3.1) can be chosen regardless of the sampling rate, as shown in Figure 3.9. In contrast, in previous light field cameras these two parameters are entangled. This unique flexibility of the proposed device is useful in many cases. For example, when the scene is suitable for depth estimation (layered or near-Lambertian), one can choose a smaller aperture size to maximize the sharpness and preserve the details in the captured light field (minimizing the filtering effect). The aliasing can be later removed by depth-aware view interpolation, as we describe in Section 4.1. Also, the sampling lattice on the lens plane is no longer to rectangular and can even be adjusted dynamically. One can also follow the analysis in Section 2.5.3 to specify the camera parameters for efficient sampling.

### 3.2.6 Related Work

There are a few concurrent works that use the ideas similar to the programmable aperture for other applications. In [70] Mohan et al. showed that by using the programmable aperture to capture a low angular-resolution light field, the spatial resolution of the in-focused regions can be increased. However, this application is somewhat limited since it only works at the in-focused regions. Also in that work, the programmable aperture is implemented by manually changing the mask at the aperture plane.

Dou and Favaro combined the defocus cue and disparity cue between the light field images captured by the programmable aperture to estimate the scene geometry and the clean image in [71]. However, the accuracy improvement due to the defocus cue is unclear. Also, they did not implement a real programmable aperture but only used a tradi-

tional aperture to perform the experiment. The data required for these two applications can be easily obtained by our device and hopefully the invention of the programmable aperture can inspire more novel applications in the future.

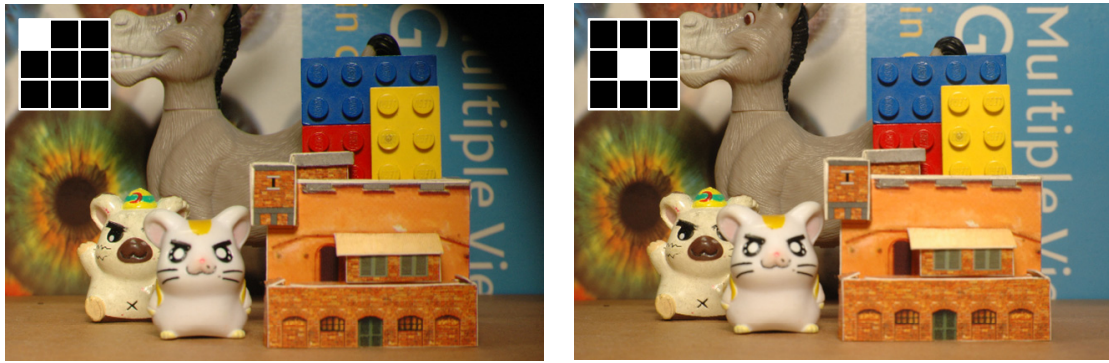
Hasinoff and Kutulakos used the multiple-exposure technique to capture several different focused images with the maximal aperture setting [72]. These images are then combined into a high quality all-in-focused image. However, the quality of the combination relies on the performance of the depth-from-defocus algorithm. Because this method also segments the exposure time budget, the image quality is actually worse than that of the image refocused from the light field. Nagahara et al. move the camera along the optical path during the exposure and thus make the blur kernel depth-invariant [73]. The quality of the all-in-focused image depends on the deblur algorithm, and all depth information are lost. This method can also be optically achieved by the wavefront coding technique [74].

### 3.2.7 Limitation and Future Direction

The proposed device has great performance and flexibility, but it requires the scene and the camera to be static because the data are captured sequentially. However, as we mentioned above, the sharpness of the in-focus regions are unaffected by multiplexing. Hence our system can capture a moving in-focus object amid static out-of-focus objects and then recover the light field and scene geometry of the static object as described in Section 3.4.

On the other hand, other devices capture the light field in one exposure at the expense of spatial resolution. However, it should be pointed out that the proposed method is complementary to the existing ones. We can place a cosine mask or a microlens array near the image plane to capture a coarse angular resolution light field and use the programmable aperture to provide the fine angular resolution needed.

Multiplexing a light field is equivalent to transforming the light field to another representation by basis projection. While our goal is to obtain a reconstruction with minimal error from a fixed number of projected images ( $M_v(u)$  in Equation 3.3), an interesting direction of future research is to reduce the number of images required for



**Figure 3.10:** *The effect of the photometric distortion.*

*The images are two of the nine light field images of a static scene. The insets show the corresponding aperture shape in exposure.*

reconstruction. The compressive sensing theory states that if a signal of dimension  $n$  has a sparse representation, we can use fewer than  $n$  projected measurements to recover the full signal [34]. We have followed this line and performed some small scale experiments, but the results were not satisfactory. We believe it is because that although the light field might be sparse when the angular and spatial dimensions are jointly considered [75], the programmable can only perform the random projection over the angular dimensions. To perform the full 4-dimensional projection, we may place multiple programmable apertures at different positions in the optical path. The design may be similar to the volumetric light attenuator in [55].

### 3.3 Photometric Calibration

The light fields captured by either our programmable aperture or other light field cameras have a noticeable photometric distortion. The light field images corresponding to the boundary of the aperture would appear very different to that corresponding to the center of the aperture, as shown in Figure 3.10.

While being termed as vignetting collectively, this photometric distortion is attributed to several sources: the cosine fall-off [5], the blocking of the lens diaphragm [7], pupil aberrations [76], etc. Because this distortion breaks the common photometric consistency assumption, it must be removed or it can obstruct view interpolation, depth

estimation, and many other applications.

The exact physical model of the vignetting effect is difficult to construct. In general, a simplified model that describes the ratio between the distorted light field image  $I_v^d(u)$  and the clean image  $I_v(u)$  by a  $2(D-1)$ -degree polynomial function  $f_v(u)$  is adopted:

$$I_v^d(u) = f_v(u)I_v(u) = \left( \sum_{i=0}^{D-1} a_{vi} \|u - \mathbf{c}_v\|_2^{2i} \right) I_v(u), \quad (3.10)$$

where  $\{a_{vi}\}$  are the polynomial coefficients,  $\mathbf{c}_v$  is the vignetting center, and  $\|\cdot\|_2$  is the Euclidean distance<sup>5</sup>. The function  $f_v$ , called *vignetting field*, is a smooth field across the image. It is large when the distance between  $u$  and  $\mathbf{c}_v$  is small and gradually decreases as the distance increases. Our goal is to estimate the parameters of the function, including  $\{a_{vi}\}$  and  $\mathbf{c}_v$ , and then to recover the distortion-free image  $I_v(u)$ .

### 3.3.1 Previous Methods

Because the number of unknown variables in Eq. 3.10 is larger than the number of observations, the estimation problem is inherently ill-posed. A straightforward method is to capture a uniformly lit object so the distortion-free image  $I_v(u)$  becomes a priori. However, the vignetting field changes when the camera parameters, including the focus, aperture size, and lens module, are adjusted. It is impractical to perform the calibration whenever a single parameter is changed.

Existing photometric calibration methods that require no specific reference object generally make two assumptions to make the problem tractable [7], [77]. First, the scene points have multiple registered observations with different levels of distortions. Second, the vignetting center  $\mathbf{c}_v$ . The first assumption is usually valid in panoramic imaging where the panorama is stitched from many images of the same view point. The second assumption is valid in most traditional camera, where the optics and the sensors are symmetric along the optical path. Some recent methods remove the first assumption by exploiting the edge and gradient priors in natural images [78], but the second assumption is still needed.

---

<sup>5</sup>The coordinates are normalized to  $(0, 1)$ .



However, both assumptions are inappropriate for the light field images for two reasons. First, the registration of the light field images taken from different view points requires an accurate per-pixel disparity map that is difficult to obtain from the distorted inputs. Second, in each light field image, the parameters,  $\{a_{vi}\}$  and  $\mathbf{c}_v$ , of the vignetting function, are image-dependent and coupled. Therefore, simultaneously estimating the parameters and the clean image is an under-determined nonlinear problem.

### 3.3.2 Proposed Algorithm

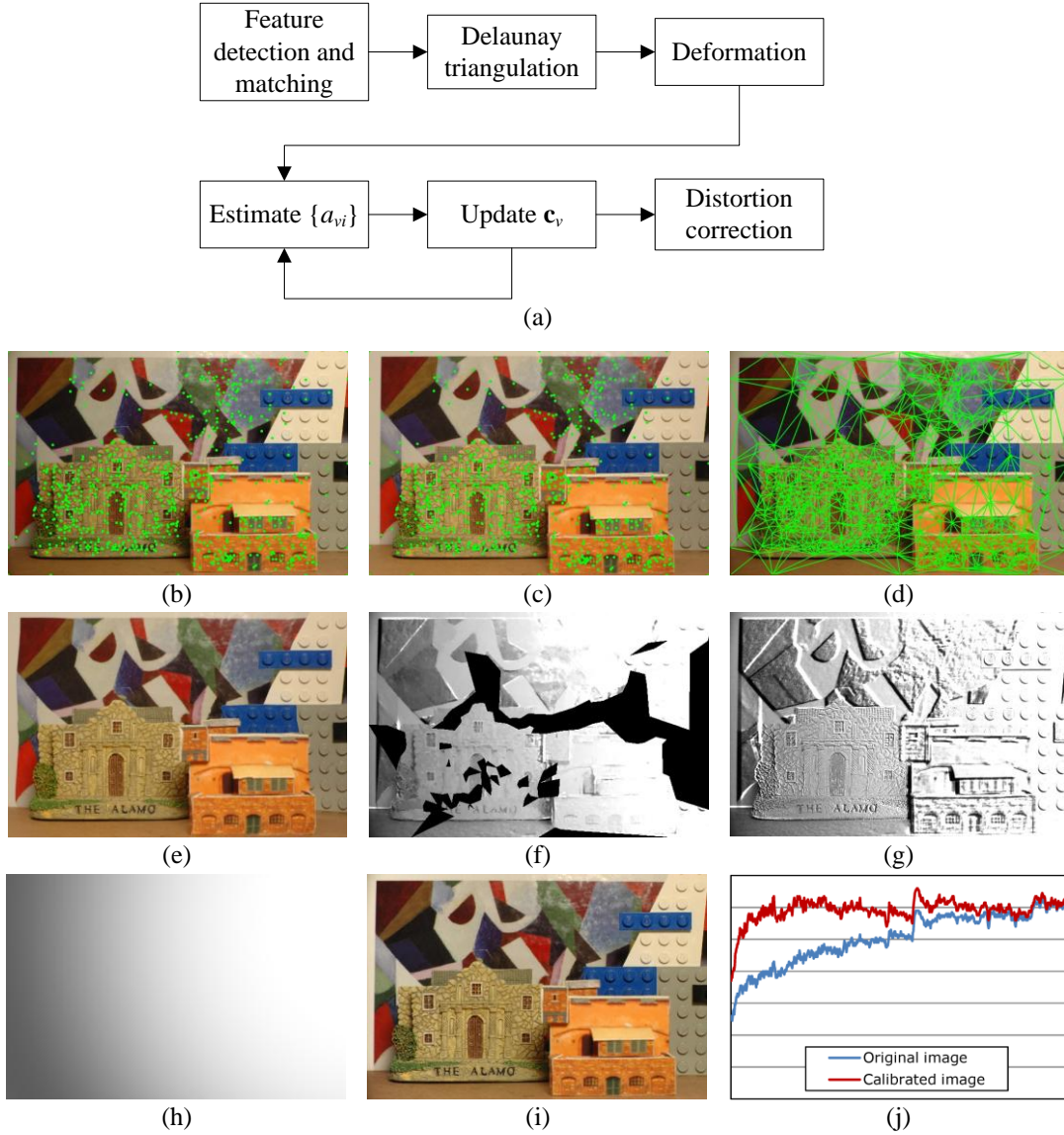
Here we propose the first algorithm to automatically calibrate the photometric distortion of the light field images. The key idea is that the light field image closer to the center of the optical path (i.e.,  $v = 0$ ) has less distortion. Therefore, we can assume  $I_0^d \approx I_0$ ,<sup>6</sup> and then approximate other  $I_v$ 's by properly transforming  $I_0$  to estimate other vignetting fields. As we mentioned, while the perfect registration is impossible, a proper outlier-rejection method must be applied.

The flowchart of the proposed algorithm is shown in Figure 3.11(a) along with an example. For an input  $I_v^d$ , we first use the SIFT method, which is well immune to local photometric distortions [79], to detect the feature points and find their valid matches in  $I_0$  (Figures 3.11 (b) and (c)). Next, we apply the Delaunay triangulation to the matched points in  $I_v^d$  to construct a mesh (Figure 3.11 (d)). For each triangle  $A$  of the mesh, we use the displacement vectors of its three vertices to determine an affine transform. By affinely warping all triangles, we obtain an image  $I_v^w$  from  $I_0$  (Figure 3.11 (e)).

The warped image  $I_v^w$  is close enough to the clean image  $I_v$  unless there are triangles including objects of different depths or incorrect feature matchings. Such erroneous cases can be effectively detected and removed by measuring the variance of the associated displacement vectors. By dividing the distorted image  $I_v^d$  with the warped image  $I_v^w$  and excluding the outliers, we have an estimated vignetting field, as shown in Figure 3.11 (f). We also show the vignetting field estimated from the image without warping in Figure 3.11 (g). Compare those two images we can find that the warping can effectively find a rough approximation of the smooth vignetting field, and the outliers around the

---

<sup>6</sup>One can also use other single-image vignetting algorithms to calibrate it.



**Figure 3.11:** The flow and results of the photometric calibration.

(b) The image to be correct  $I_u^d$ . Not the left side is darker. (c) The reference image  $I_0$ . Matched points in (b) and (c) are marked as green points. (d) Triangulation of the matched features in (b). (e) The image  $I_v^w$  warped from (c) based on the triangular mesh. (f) The approximated vignetting field with suspicious areas (black) removed. (g) The vignetting field approximated without warping. (h) The estimated parametric vignetting field. (i) The calibrated image  $I_v$ . (j) The intensity profile of the 420th scanline before and after the calibration.

depth discontinuities are successfully excluded.

After we have the approximation of the vignetting field, we estimate the parametric vignetting function (Equation 3.10) by minimizing an objective function  $E(\{a_{vi}\}, \mathbf{c}_v)$ :

$$E(\{a_{vi}\}, \mathbf{c}_v) = \sum_u \left( \frac{I_v^d(u)}{I_v^w(u)} - f_v(u) \right)^2. \quad (3.11)$$

This objective function is non-linear because  $\{a_{vi}\}$  and  $\mathbf{c}_v$  are coupled. Therefore, we minimize this function iteratively. Given an initial estimate, we first fix the vignetting center  $\mathbf{c}_v$ . This makes the Equation 3.11 linear in  $\{a_{vi}\}$ , which can be easily solved by a least square estimation. Then, we fix  $\{a_{vi}\}$  and update  $\mathbf{c}_v$ . This is done by a gradient descent method. We want to find a displacement  $\mathbf{d}_v$  such that  $E(\{a_{vi}\}, \mathbf{c}_v + \mathbf{d}_v)$  is minimized.

Specifically, let  $r_i$  denote the distance between  $u_i = (x_i, y_i)$  and  $\mathbf{c}_v = (c_{v,x}, c_{v,y})$ , the  $N$ -D vector  $\mathbf{r} = [r_1, r_1, \dots, r_N]^T$  denote the distances between all points  $\{u_i | i = 1, \dots, N\}$  and  $\mathbf{c}_v$ . We also denote  $\mathbf{I}_v$  as the vector of estimated vignetting field, that is, the ratio  $I_v^d / I_v^w$ . Because  $\mathbf{c}_v$  is the only variable, we can re-define the vignetting function  $f_v(u)$  as a vector function  $\mathbf{f}(\mathbf{c}_v) = [f_v(u_1), f_v(u_2), \dots, f_v(u_N)]^T$ . Then Equation 3.11 is equivalent to the L2 norm of the error vector  $\boldsymbol{\varepsilon}$ :  $\|\boldsymbol{\varepsilon}\| = \|\mathbf{I}_v - \mathbf{f}(\mathbf{c}_v)\|$ . Then the optimal displacement  $\mathbf{d}_v$  at iteration  $t$  can be obtained by solving the normal equation:

$$\mathbf{J}^T \mathbf{J} \mathbf{d}_v = -\mathbf{J}^T \boldsymbol{\varepsilon}_{t-1}, \quad (3.12)$$

where  $\mathbf{J}$  is the Jacobian matrix ( $\mathbf{J} = \frac{d\mathbf{f}}{d\mathbf{c}_v}$ ) and  $\boldsymbol{\varepsilon}_{t-1}$  is the error vector of the previous iteration. We here set  $D = 4$  and thus the Jacobian is:

$$\frac{d\mathbf{f}}{d\mathbf{c}_v} = \begin{bmatrix} -(x_0 - c_{v,x})[2a_1 + 4a_2r_0^2 + 6a_3r_0^4] & -(y_0 - c_{v,y})[2a_1 + 4a_2r_0^2 + 6a_3r_0^4] \\ -(x_1 - c_{v,x})[2a_1 + 4a_2r_1^2 + 6a_3r_1^4] & -(y_1 - c_{v,y})[2a_1 + 4a_2r_1^2 + 6a_3r_1^4] \\ \vdots & \vdots \\ -(x_N - c_{v,x})[2a_1 + 4a_2r_N^2 + 6a_3r_N^4] & -(y_N - c_{v,y})[2a_1 + 4a_2r_N^2 + 6a_3r_N^4] \end{bmatrix}. \quad (3.13)$$

Note that this Jacobian is evaluated using the vignetting center obtained in the previous iteration and the coefficients estimated in this iteration. In this way the convergence



**Figure 3.12:** Application of the proposed algorithm to the dataset of other light field cameras.

The light field is captured using the heterodyned light field camera in [21] (Left) The original light field image. (Right) The calibrated and re-synthesized light field image.

speed is increased. We also find that because the number of parameter is small, we can sub-sample the image to reduce the computation. Usually 1000 to 2000 inlier samples are sufficient. For each image, we perform 50 iterations and choose the parameters with minimal objective value as the result. One obtained vignetting field is shown in Figure 3.11 (h).

Finally, we divide  $I_v^d$  by  $f_v$  to recover the clean image  $I_v$ , as shown in Figure 3.11 (i). One scanline profile is also shown in Figure 3.11 (j) for comparison. We can see that the recovered image has much less distortions.

It should be emphasized that the light field data captured by all light field cameras have photometric distortion to some degrees. Our photometric calibration can be applied to those data as well, as one example shown in Figure 3.12. The data in the example is captured using the heterodyned light field camera [21]. It is very noisy and has serious distortion. We apply our algorithm to estimate the vignetting field and then estimate the scene geometry using the method described in the next section, and finally perform the view-interpolation to re-synthesize the image at the same view point. We can see that the re-synthesized image is much cleaner than the original one.

### 3.4 Multi-View Depth Estimation

Images corresponding to new viewpoints or focus settings can be rendered from the captured light field by re-sampling. However, the quality of the rendered image is dictated by the bandwidth of the light field, which strongly depends on the scene geometry [11], [12]. As we have analyzed in Chapter 2, a scene with higher depth range requires a higher angular resolution for aliasing-free rendering. Although one can adjust the angular resolution of the programmable aperture camera, a high angular sampling rate requires a long capture duration and a large storage, which may not be always affordable.

Here we propose a multi-view depth estimation algorithm to generate view-dependent depth maps for view interpolation. By depth-dependent view interpolation, we can greatly reduce the angular sampling rate for the near-Lambertian scene. The estimated depth maps can also benefit other applications, such as z-keying, matting [57], robot vision, and so on.

#### 3.4.1 Previous Methods

Previously the aliasing is removed by pre-filtering [3] or post-filtering [80]. In this way the out-of-focus objects are blurred. However, while the visual quality is improved, these methods implicitly require the depth range of the scene but did not fully utilize this information. In [11] the image is segmented into many blocks and each block is assigned an optimal depth value. If the user wants the best visual quality, this method would require the per-pixel depth value, which is exactly what we want to estimate.

The multi-view depth estimation problem is similar to the traditional stereo correspondence problem [81]. However, the visibility reasoning is extremely important for multi-view depth estimation since the occluded views should be excluded from the depth estimation. Previous methods that determine the visibility by hard constraint [82] or greedy progressive masking [83] can easily be trapped in local minima because they cannot recover from incorrect occlusion guess.

### 3.4.2 Main Concept

Inspired by the symmetric stereo matching algorithm [84], we alleviate this problem by iteratively optimizing 1) a view-dependent depth map  $D_v$  for each image  $I_v$  and 2) an occlusion map  $O_{vw}$  for each pair of neighboring images  $I_v$  and  $I_w$ . If a scene point projected onto a point  $u$  in  $I_v$  is occluded in  $I_w$ , it does not have a valid correspondence. When this happens, we can set  $O_{vw}(u) = 1$  to exclude it from the matching process. On the other hand, if the estimated correspondence  $u'$  of  $u_w$  in  $I_w$  is marked as invisible, that is,  $O_{wv}(u') = 1$ , the estimate is unreliable.

Depth and occlusion estimation is formulated as a discrete labeling problem [85]. For each pixel  $u_v$ , we need to determine a discrete depth value  $D_v(u) \in \{0, 1, \dots, d_{max}\}$  and a binary occlusion value  $O_{vw}(u) \in \{0, 1\}$ . More specifically, given a set of light field images  $\mathcal{J} = \{I_v\}$ , we want to find a set of depth maps  $\mathcal{D} = \{D_v\}$  and a set of occlusion maps  $\mathcal{O} = \{O_{vw}\}$  to minimize the energy functional defined by

$$\begin{aligned} E(\mathcal{D}, \mathcal{O} | \mathcal{J}) = & \sum_v \{E_{dd}(D_v | \mathcal{O}, \mathcal{J}) + E_{ds}(D_v | \mathcal{O}, \mathcal{J})\} \\ & + \sum_v \sum_{w \in \mathcal{N}(v)} \{E_{od}(O_{vw} | D_v, \mathcal{J}) + E_{os}(O_{vw})\}, \end{aligned} \quad (3.14)$$

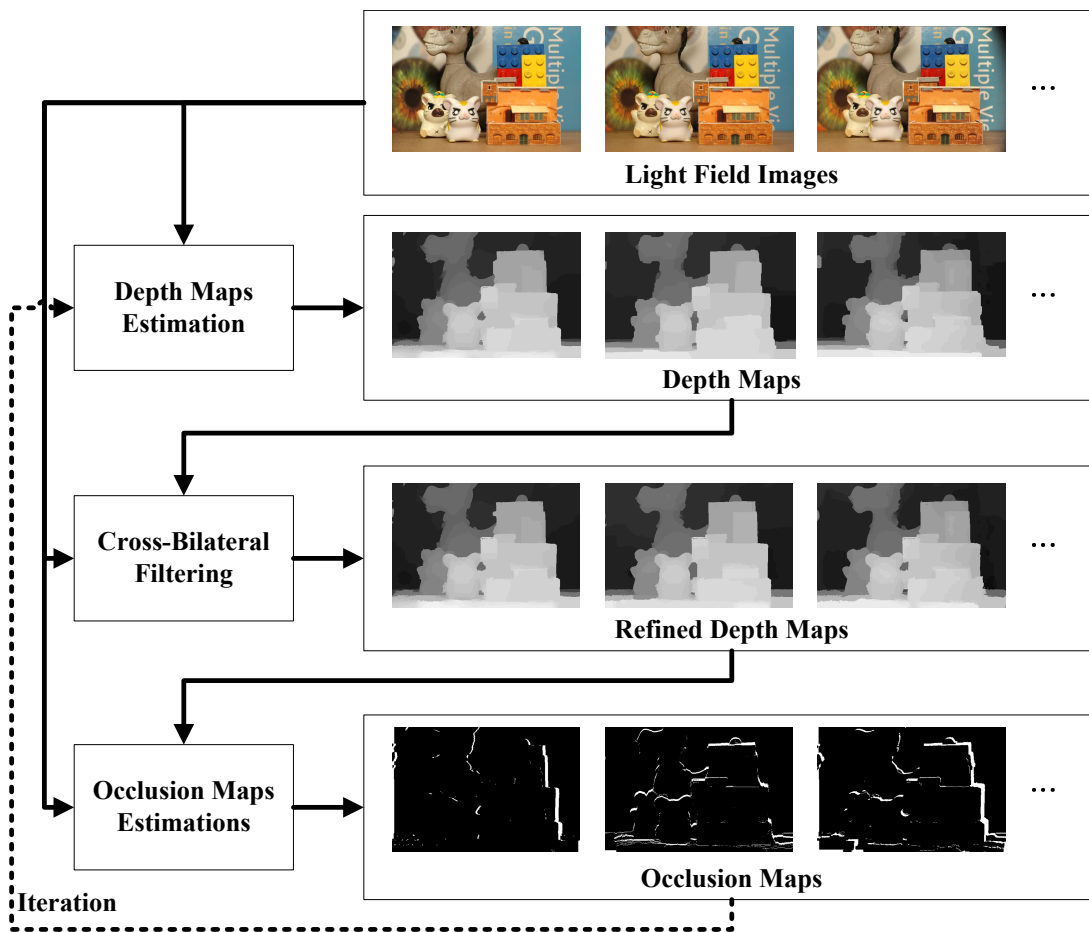
where  $E_{dd}$  and  $E_{ds}$  are the data term and the smoothness (or regularization) term, respectively, of the depth map, and  $E_{od}$  and  $E_{os}$  are the data term and the smoothness term, respectively, of the occlusion map.  $\mathcal{N}(v)$  is the set of the viewpoints which are close to  $v$ <sup>7</sup>.

The intuition of our formulation is as follows. The optimal depth maps and occlusion maps should be compatible to each other. Given a perfect depth map, it is easy to know if a specific point will be occluded when the viewpoint is changed, and the answer should be the same as that from the perfect occlusion map. Therefore, the energy of the depth maps should depend on not only the input images, but also the occlusion maps, and vice versa.

However, this dependency makes the optimization intractable. The occlusion map cannot be easily estimated from the observation alone, and the depth maps can not be precisely estimated when the occlusion reasonable is absent. Therefore, we minimize

---

<sup>7</sup>In our case, this is equivalent to the light field images captured with closer aperture centers



**Figure 3.13:** Overview of the proposed multi-view depth estimation algorithm.

the energy iteratively. In each iteration, we first fix the occlusion maps and minimize  $E_{dd} + E_{ds}$  by updating the depth maps and then fix the depth maps and minimize  $E_{od} + E_{os}$  by updating the occlusion maps.

The overview of the proposed algorithm to minimize Equation 3.14 is shown in Figure 3.13. In the following subsections, we describe the definitions of energy terms, and then the method we use to minimize them in details.

### 3.4.3 Definitions of Energy Terms

The energy terms should be defined in the way such that it obeys the fundamental rules in vision (e.g., the color and intensity consistency), and also follows the prior knowledge (e.g., the depth variation of the object should be smooth). Inevitably, some hand-tuned

parameters, such as weighting coefficients and the thresholds, would present in the function, but this would not be a problem in practical as long as the optimal parameters do not vary significantly from one dataset to another.

Before proceeding, let  $\alpha$ ,  $\beta$ ,  $\gamma$ ,  $\zeta$ ,  $\eta$ ,  $\rho$ ,  $\gamma_c$  and  $\gamma_s$  denote the coefficients;  $K$  and  $T$  denote the thresholds. These parameters are empirically determined and fixed in the experiments. The data term  $E_{dd}$  is a unary function<sup>8</sup>,

$$E_{dd}(D_v|\mathcal{O},\mathcal{I}) = \sum_u \left\{ \sum_{w \in \mathcal{N}(v)} (\bar{O}_{vw}(u)C(I_v(u), I_w(x + D_{vw}(x))) + \alpha O_{vw}(x + D_{vw}(u))) \right\}, \quad (3.15)$$

where  $\bar{O}_{vw}(u) = 1 - O_{vw}(u)$ , which can be considered as a *non-occlusion* map.  $D_{vw}$  is the disparity vector on the image plane corresponding to the depth value  $D_v$ .

$C(I_a, I_b)$  is the dissimilarity measurement between the color or brightness of two pixels  $a$  and  $b$ . Here we use two different measurements. In the early iterations, because the occlusion maps are either absent or inaccurate, we use a window-based rank and census transform [86]. For each pixel, we pack all neighboring pixels into a vector and replace the intensity values with their ranks in the vector. A simple example is as follows:

$$[120, 128, 178, 160, 175, 182, 140, 20, 25] \rightarrow [2, 3, 7, 5, 6, 8, 4, 0, 1]. \quad (3.16)$$

This operation is also called a *rank* filter. It has been shown in [87] that the rank filter is the most robust filter against noise, vignetting, and other distortions. Without the knowledge of the occlusion or object segmentation, it gives the best initial guess.

When the occlusion maps are at hand in the later iterations, we replace the rank filter with the Birchfield-Tomasi per-pixel measurement [88] that is wildly used in most stereo estimation algorithms. For both measurement functions, they are clamped by a threshold  $K$ . In this way, the outliers, such as specularity, which cannot be matched would not cause a large penalty in the energy function.

The intuition of the  $E_{dd}$  is as follows. When the corresponding pixel of  $u$  in  $I_w$  is non-occluded ( $\bar{O}_{vw}(u) = 1$ ),  $I_v(u)$  should look similar to the corresponding pixel

---

<sup>8</sup>Each entity in the summation depends on a single node (pixel) in the graph (image).



$I_w(x + D_{vw}(x))$ , so the value of the dissimilarity measurement should be small. On the other hand, if the estimated corresponding pixel is impossible to be observed in  $I_v$ , this estimation should be penalized.

The definition of the pairwise smoothness term  $E_{ds}$  is based on a generalized Potts model:

$$E_{ds}(D_u|\mathcal{O}, \mathcal{I}) = \sum_{\substack{(u,s) \in \mathcal{P}, \\ O_v(u)=O_v(s)}} \beta \min(|D_v(u) - D_v(s)|, T), \quad (3.17)$$

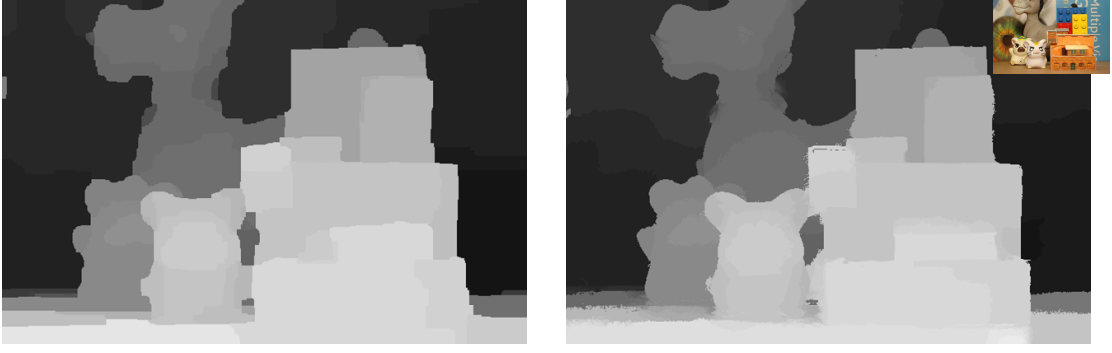
where  $\mathcal{P}$  is the set of all pairs of neighboring pixels and  $O_v = \bigcap O_{vw}$ , which is true only when  $u_v$  is occluded in all other images. This energy function encourages the depth to be piecewise smooth. But when the depth difference is larger than a threshold, the penalty should be uniform. It is because when this happens, these two pixels should belong to two different objects, and there should be no correlation between the depth values of two independent objects.

In some stereo estimation algorithm, the smoothness term is modulated with the color difference of two pixels. However, we find that when the occlusion reasoning is accurate, the modulation does not give any noticeable performance gain.

Next, we describe the energy terms involved in the second step in one iteration. Because the depth maps  $\mathcal{D}$  are fixed in this step, the prior of an occlusion map can be obtained by warping the depth map. Specifically, let  $W_{vw}$  denote a binary map. The value of  $W_{vw}(u)$  is 1 when the depth map  $D_w$  warped to the viewpoint  $v$  is null at  $u_v$  and 0 otherwise. If  $W_{vw}(u) = 1$ ,  $u_v$  might be occluded in  $I_w$ . With this prior, the data term  $E_{od}$  is formulated as:

$$\begin{aligned} E_{od}(O_{uv}|D_u, \mathcal{I}) &= \sum_u (\bar{O}_{vw}(u) C(I_v(u) - I_w(u + D_{vw}(u))) \\ &\quad + \gamma O_{vw}(u) \\ &\quad + \zeta |O_{vw}(u) - W_{vw}(u)|), \end{aligned} \quad (3.18)$$

where the first term biases a pixel to be non-occluded if it is similar to its correspondence, the second term penalizes the occlusion ( $O = 1$ ) to prevent the whole image from being marked as occluded, and the third term favors the occlusion when the prior  $W_{vw}$  is true.



**Figure 3.14:** *The effect of the cross bilateral filtering.*

(Left) The estimated map at the first iteration where the occlusion map is absent. (Right) The cross bilateral filtered depth map. The inset shows the light field image used in the filtering.

Finally, the definition of the smoothness term  $E_{os}$  is based on the Potts model:

$$E_{os}(O_{vw}) = \sum_{(u,s) \in \mathcal{P}} \eta |O_{vw}(u) - O_{vw}(s)|. \quad (3.19)$$

This function suggests that the occlusion map should be piecewise smooth.

### 3.4.4 Optimization

The solution of the energy minimization is a maximum a posteriori (MAP) estimate of a Markov random field (MRF), for which high-performance algorithms have been recently developed.

We use the MRF optimization library from Middlebury [89]. Among all algorithms provided in the library, both the alpha-expansion graph cut [85], loopy belief propagation [90], and the tree-reweighted message passing [91] perform well, but the last one gives slightly better results at the cost of execution time.

Specifically, the tree-reweighted message passing (TRW-S) first partition the MRF into several spanning trees which fully cover all nodes in the MRF. A reweighted max-product belief propagation is performed for each tree and finally the beliefs from all trees are then combined into the final belief.

The proposed algorithm would require many iterations because in the beginning the occlusion map is far from perfect. To speedup the convergence, we make an assumption

that a depth map should be smooth if the associated image is smooth. Therefore, we can apply a modified cross bilateral filtering [92] to the depth maps at the end of each iteration to improve their quality.

Specifically, given the depth map  $D_v$  and the image  $I_v$ , we first build a data cost volume  $C(u, d)$  based on the depth value  $D_v(u)$ :

$$C(u, d) = \min((d - D_v(u))^2, \rho D_{max}). \quad (3.20)$$

Then, a L1-norm bilateral filter is applied to each slice of the cost value. The filtered cost volume  $C'(u, d)$  is

$$C'(u, d) = \frac{\sum_{s \in N(u)} f_c(u, s) f_s(u, s) C(s, d)}{\sum_{s \in N(u)} f_c(u, s) f_s(u, s)}, \quad (3.21)$$

where

$$f_c(u, s) = \exp\left(-\frac{|I(u) - I(s)|}{\gamma_c}\right), \quad (3.22)$$

and

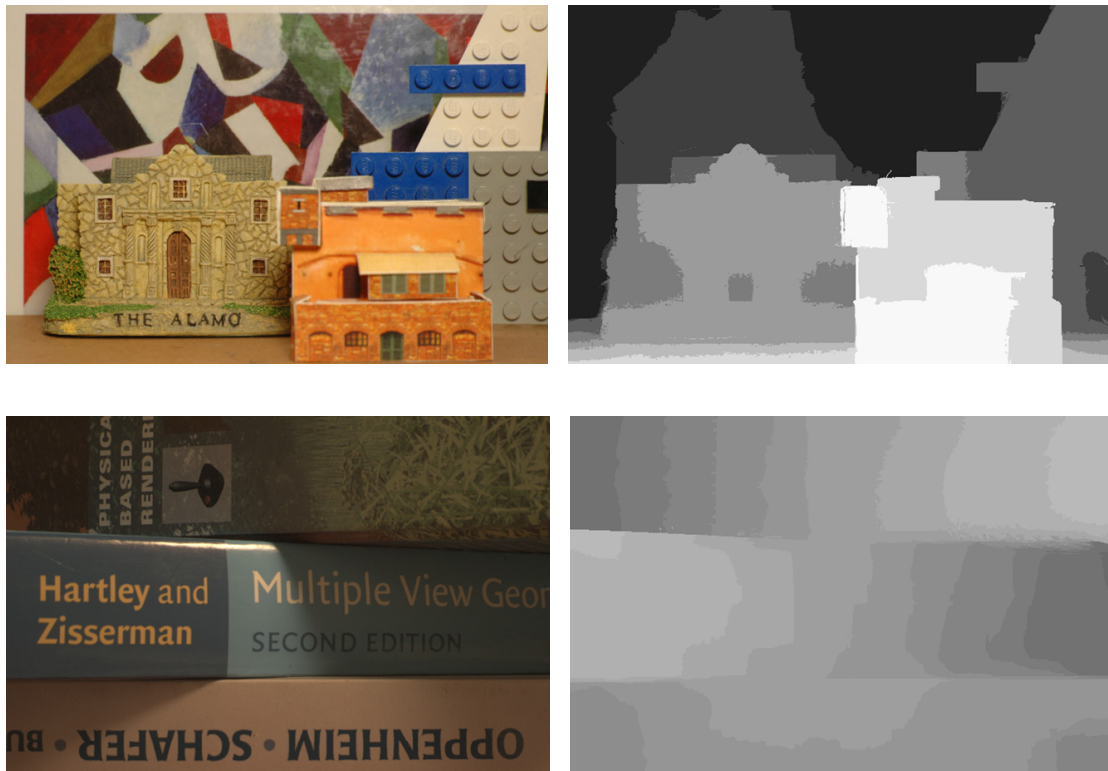
$$f_s(u, s) = \exp\left(-\frac{|u - s|}{\gamma_s}\right). \quad (3.23)$$

In other words, we use the color difference as well as the spatial distance to weight the contribution of the neighboring pixels. Then, the minimal value in the filtered cost volume is the new depth value. We find that this refined depth map can address many errors at the depth discontinuities in the early iterations, as shown in Figure 3.14. When the occlusion maps become accurate in the later iterations, the cross bilateral filter can be removed. Using the cross-bilateral filtering, we usually only require 3 iterations to obtain the a reasonable result.

### 3.4.5 Performance Analysis

The recovered depth maps of the light field we captured in shown in Figure 3.15 and 3.16. We can see that our algorithm works well for layered scenes, slanted planes, textless objects, and even furry objects.

Because our algorithm is designed for the problem of dense multi-view depth estimation, for which a public benchmark is not available, we apply our algorithm to public



**Figure 3.15:** *Depth maps of the captured light field dataset (1/2).*



**Figure 3.16:** *Depth maps of the captured light field dataset (2/2).*

stereo benchmark on the Middlebury website<sup>9</sup>. Our algorithm achieves an average rank of 8.2 when only two images are used in the optimization and 7.0 when all images are used. The results are shown in Figure 3.17, 3.18, 3.19, and 3.20. This score is the fifth best one in a pool of 41 algorithms at the time of updating. Our algorithm is the best one among all algorithms that do not utilize image over-segmentation and plane fitting. However, most of those algorithms cannot be trivially applied to the multi-view dataset.

We find that in the top algorithms including ours, the errors are usually due to the aliasing around the object boundaries. Assign a single depth to those pixels that may be covered by more than one object is not reasonable so the score is somewhat biased.

Finally, we compare our algorithm with other multi-view depth estimation algorithms [83], [82]. The results of [83] and [82] are shown in Figure 3.21. We can see that our algorithm is much better than them. It is interesting that our basic formulation is very similar to that in [83], but they use a greedily occlusion reasoning without iteration because they think it may not converge. Here we show that the convergence is always possible, at least in all of our experiments.

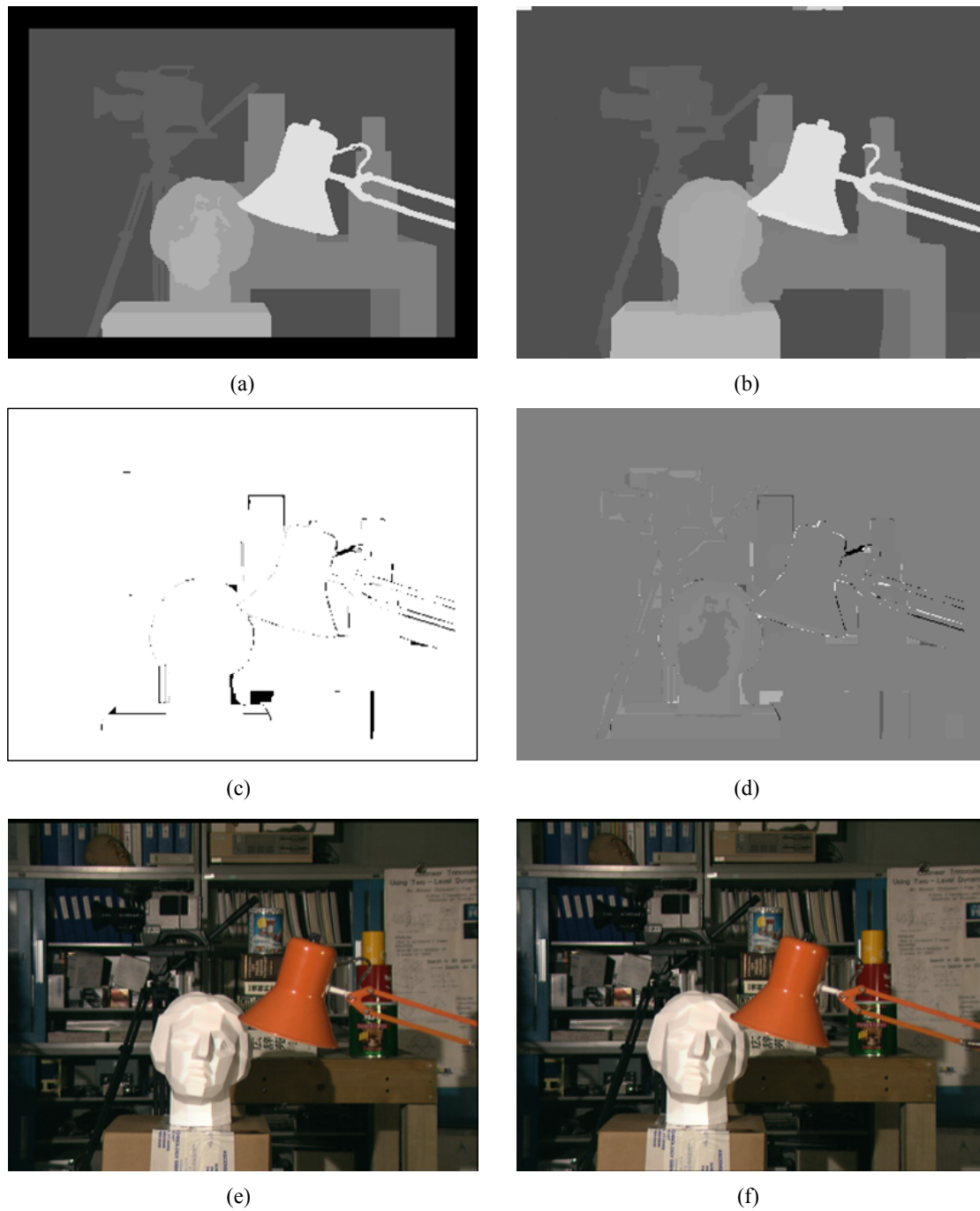
### 3.4.6 Discussion

The light field images captured by our programmable aperture camera have several advantages for depth estimation. First, the viewpoints of the light field images are well aligned with the 2D grid on the aperture, and thus the depth estimation can be performed without camera calibration. Indeed, all the results present here are obtained without applying any geometric calibration. Second, the disparity corresponding to a depth value can be adjusted by changing the camera parameters without any additional rectification as required in camera array systems. Finally, unlike depth-from-defocus methods [93], [33], there is no ambiguity in the scene points behind and in front of the in-focus object.

It should be emphasized that both the light field data and the post-processing algorithms are indispensable for generating plausible photographic effects. To illustrate this, we apply a single light field image and its associated depth map to the Photoshop

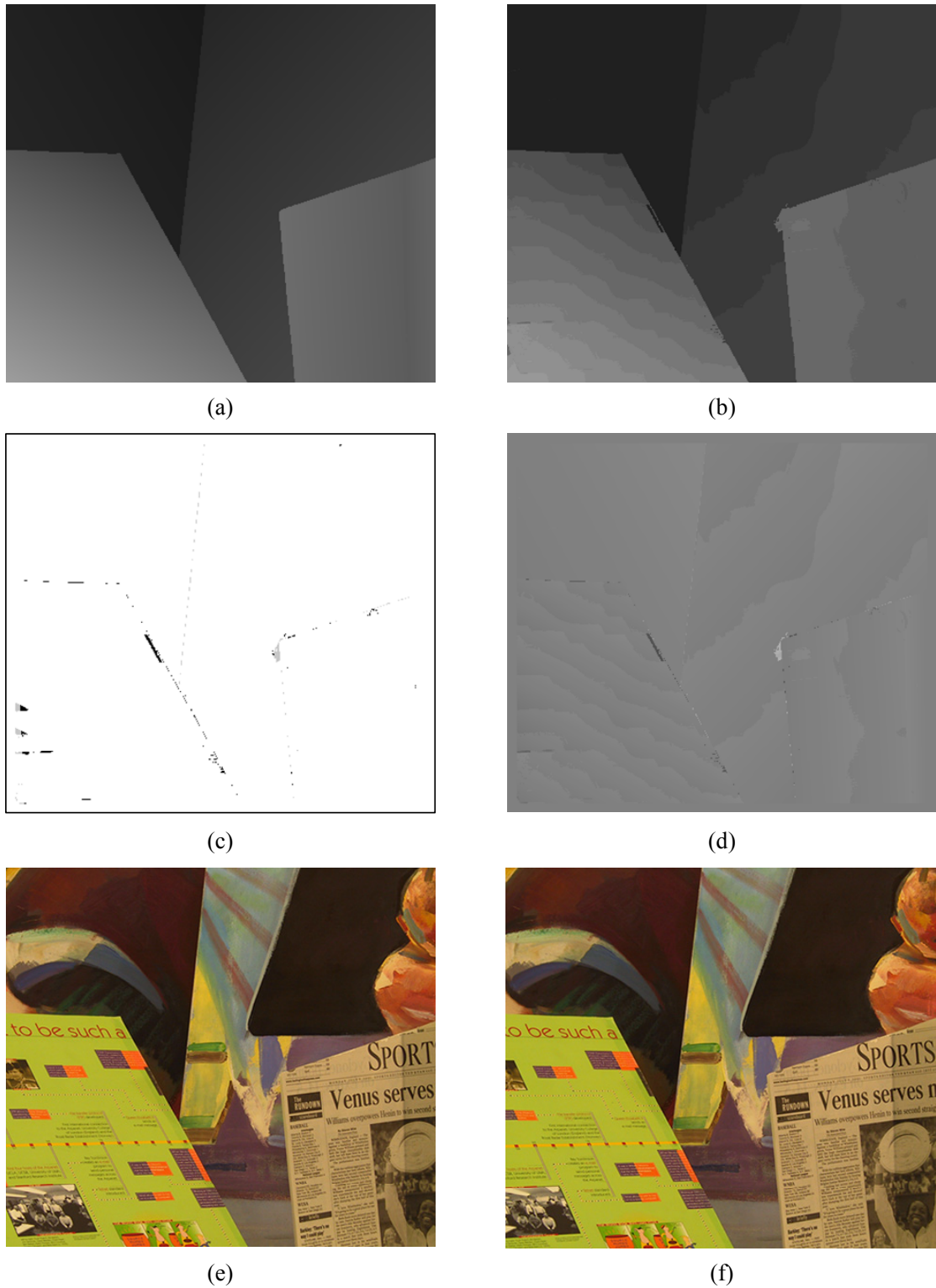
---

<sup>9</sup><http://vision.middlebury.edu/stereo/>



**Figure 3.17:** *The results of the test dataset **Tsukuba**.*

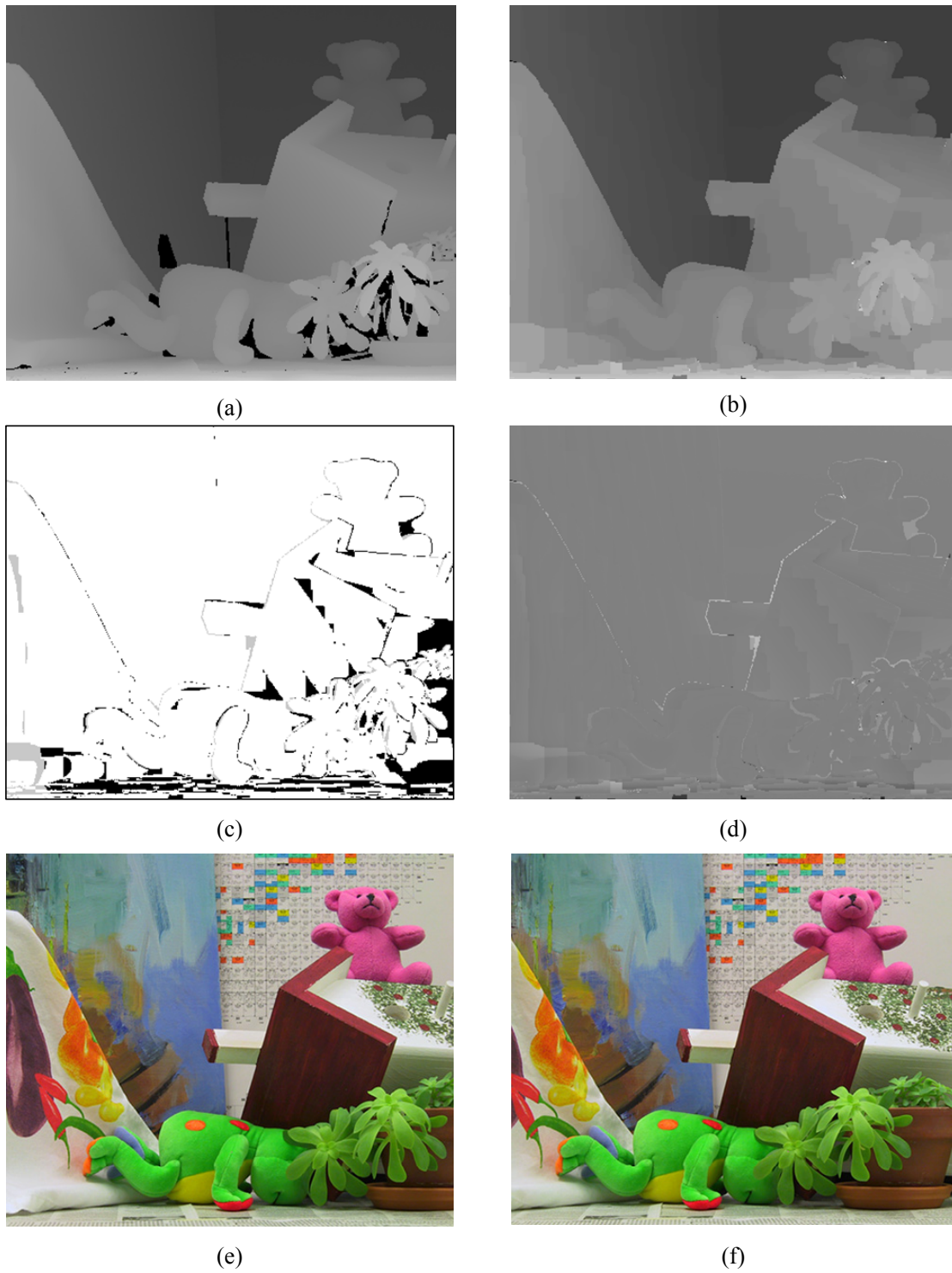
(a) *The groundtruth disparity map, (b) the estimated disparity map, (c) the bad pixels (absolute disparity error  $> 1.0$ ), (d) the signed disparity error map, (e) the left input image, and (f) the right input image.*



**Figure 3.18:** *The results of the test dataset **Veuns**.*

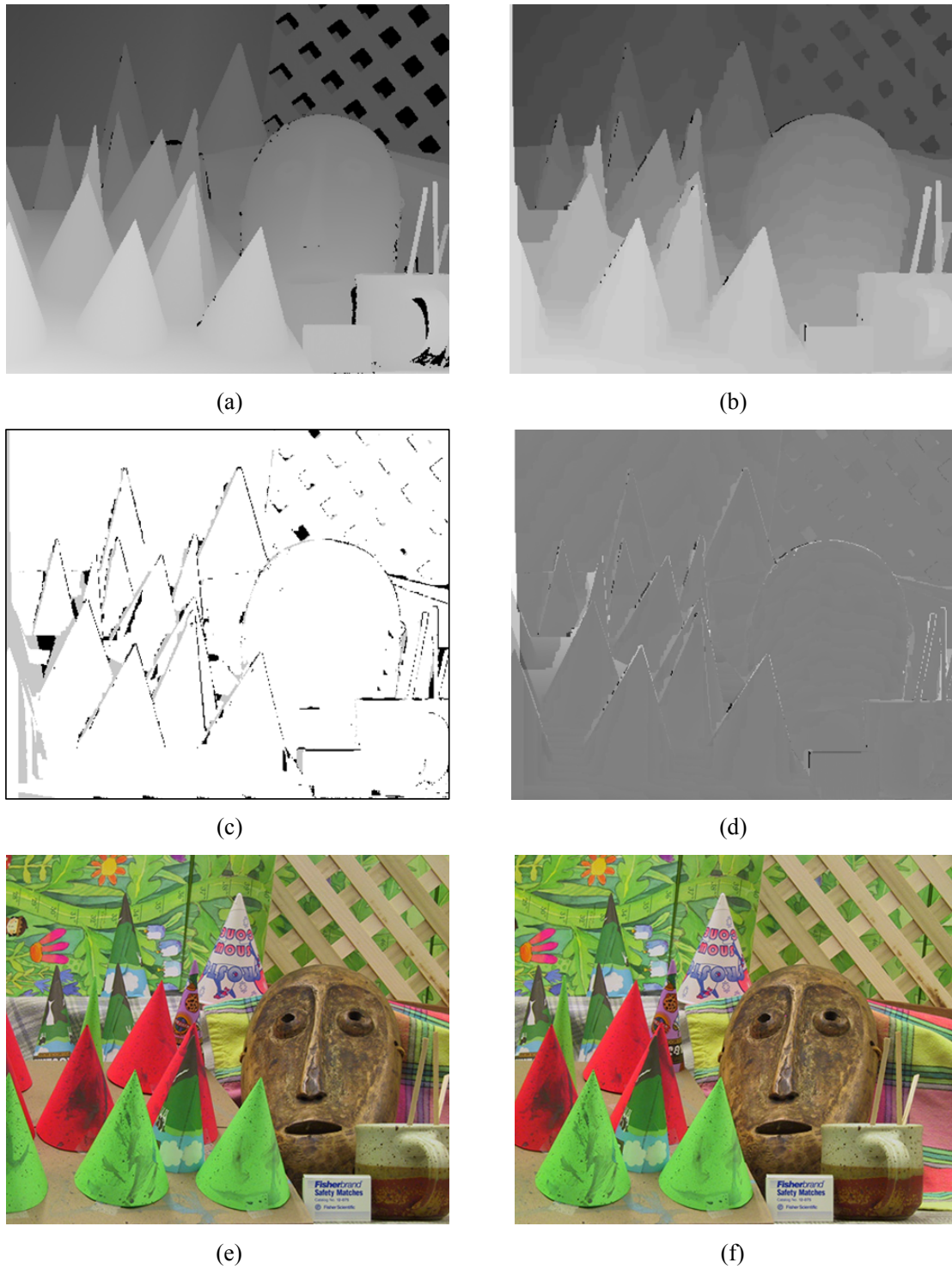
(a) *The groundtruth disparity map, (b) the estimated disparity map, (c) the bad pixels (absolute disparity error  $> 1.0$ ), (d) the signed disparity error map, (e) the left input image, and (f) the right input image.*





**Figure 3.19:** *The results of the test dataset **Teddy**.*

(a) *The groundtruth disparity map, (b) the estimated disparity map, (c) the bad pixels (absolute disparity error  $> 1.0$ ), (d) the signed disparity error map, (e) the left input image, and (f) the right input image.*



**Figure 3.20:** *The results of the test dataset **Cones**.*

(a) *The groundtruth disparity map, (b) the estimated disparity map, (c) the bad pixels (absolute disparity error  $> 1.0$ ), (d) the signed disparity error map, (e) the left input image, and (f) the right input image.*

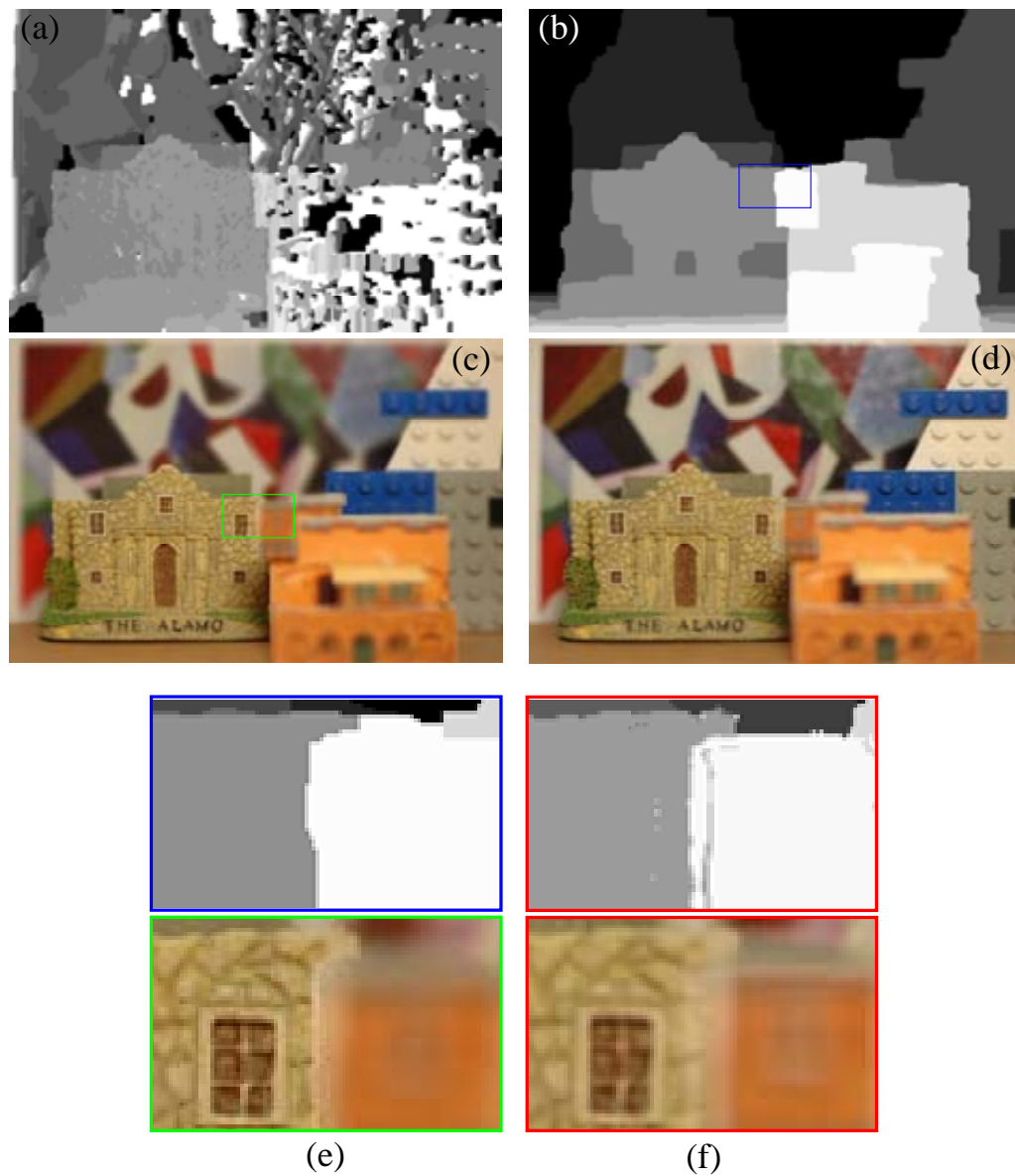


(a)



(b)

**Figure 3.21:** *The results of other multi-view depth estimation algorithms.*  
(a) *The result of [83].* (b) *The result of [82].*



**Figure 3.22:** *Effects of the post-processing algorithms.*

(a) *Depth map estimated without photometric calibration and occlusion reasoning.* (b) *Depth map estimated without occlusion reasoning.* (c) *Defocusing by the Photoshop Lens Blur tool.* (d) *Defocusing using the light field and depth maps.* (e) *Close-up of (b) and (c).* (f) *Corresponding close-up of (d) and Figure 3.15.*

Lens Blur tool to generate a defocused image. The result shown in Figure 3.22 (c) contains many errors, particularly at the depth discontinuities, Figure 3.22 (e). In contrast, our results, Figure 3.22 (d) and (f), are more natural. The boundaries of the defocused objects are semi-transparent and thus the objects behind can be partially seen. Also we can see that without photometric calibration, the result of the depth estimation would be very bad.

### 3.5 Summary

In this chapter we have described a system for capturing light field using a programmable aperture camera with an optimal multiplexing scheme. Along with the programmable aperture, we have also developed two post-processing algorithms for photometric calibration and multi-view depth estimation. To our best knowledge, this system is the first single-camera system that generates light field at the same spatial resolution as that of the sensor, with adjustable angular resolution, and free of photometric distortion. In addition, the programmable aperture is fully backward compatible with conventional apertures.

While we have focused on the light field acquisition in this work, the programmable aperture camera can be further exploited for other applications. For example, it can be used to realize a computational camera with a fixed mask.



## Chapter 4

# Applications of the Light Field Cameras

In this chapter, we describe the applications using the light field captured by the programmable aperture camera and the depth information estimated by the proposed multi-view depth estimation algorithm. We first present a method to synthesize a high-quality image of a novel viewpoint. Then, we present a method to synthesize high quality refocused image. Finally, we present a new interface to adjust the focus settings in real-time based on the feature matching. There are several other applications such as fusion of all-focused images and depth-detection are presented in Chapter 2. It is because they can be easier explained in the proposed light transport framework.

### 4.1 View Interpolation

Given the light field  $L(u, v)$ , we can easily generate a image of a new viewpoint by properly sampling the light field. Because the captured light field samples are discrete, we must apply a reconstruction filter to obtain the continuous signal. However, as we have analyzed in Chapter 2, the optimal reconstruction filter depends on the scene geometry. If the trivial bilinear filter is used, serious ghosting effect due to aliasing is observed, as shown in Figure 4.1. In this case, the raw angular resolution is  $3 \times 3$ , which is far below the Nyquist rate.



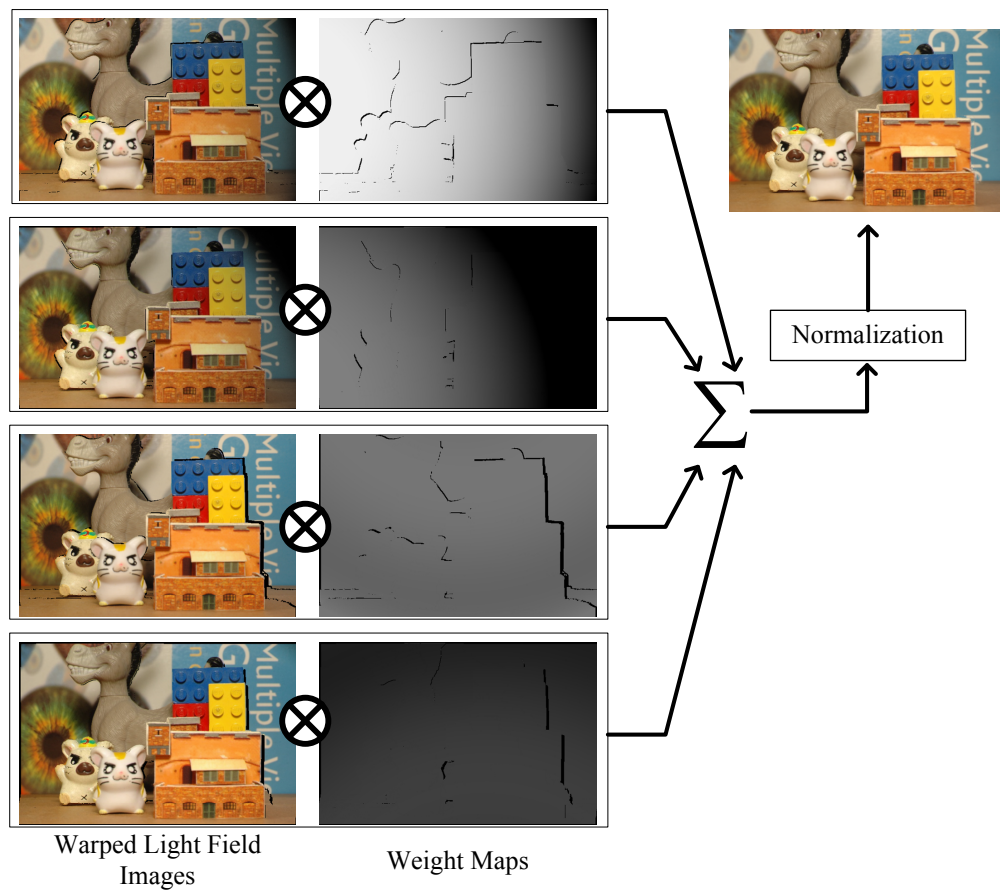


**Figure 4.1:** *Image interpolated without depth information.*

Previously the aliasing was removed using filtering [3], [12], and the out-of-focus objects become blurry in the resulting image. Here we strive to obtain a clean, all-focused image. We use a modified projective texture mapping method [94]. The flow of the proposed algorithm is shown in Figure 4.2. Given a viewpoint, four closest images are warped according to their associated depth maps obtained by the algorithm described in 3.4. The warped images are then blended together. The weight of each image is inversely proportional to the distance between its viewpoint and the given viewpoint. Furthermore, although the photometric distortions in the light field are already removed using the algorithm described in 3.3, the pixels which belong to the regions with serious distortions are much noisier than others. Therefore, we also weight each pixel by its vignetting function. After all images are warped, we combine them together and normalize all weights to obtain the final image.

One result is shown in Figure 4.3, which has the same viewpoint to the one in Figure 4.1. We can see that the ghosting effect is greatly suppressed without blurring. In addition, although the specified viewpoint is very close to that of an existing light field image, our distortion-aware weighting scheme can further remove the small photomet-





**Figure 4.2:** The illustration of the proposed depth-assisted view interpolation.



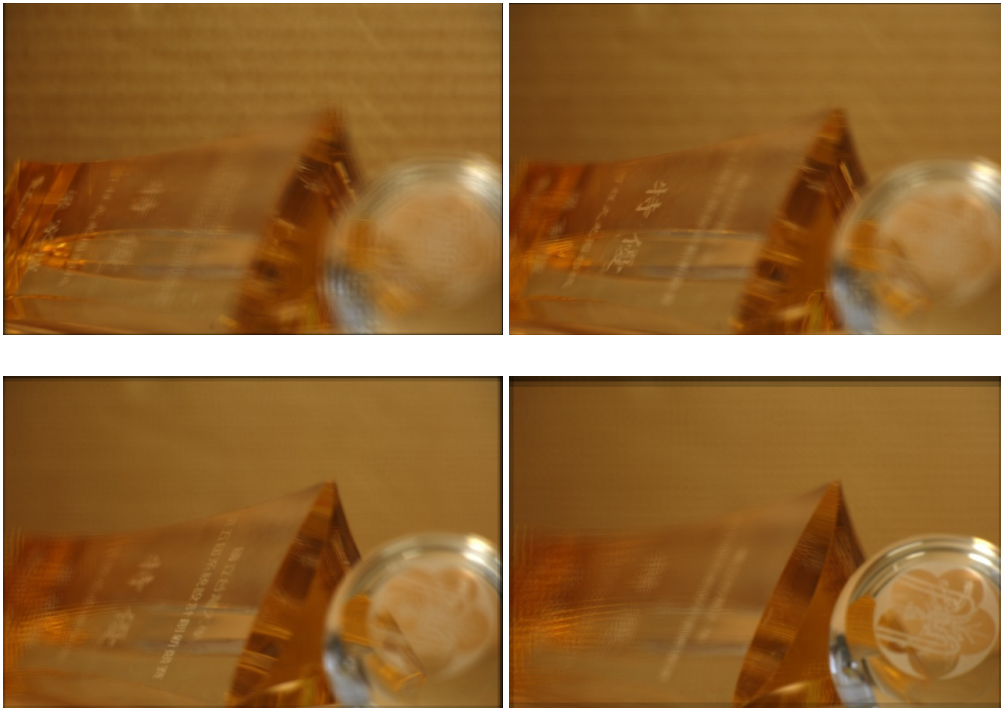
**Figure 4.3:** *Image interpolated with depth information.*

ric distortions in the light field images.

## 4.2 Digital Refocusing

Given the light field in the camera, we can easily generate the images of different focus planes by transforming the light field before the integration [12], or by changing the integration directions [24]. One example is shown in Figure 4.4. This scene contains a transparent object in front of a nearly uniform background. The geometry of this scene is difficult to estimate. However, since our acquisition method does not impose any restriction on the scene, we can capture the light field with  $4 \times 4$  and generate faithful refocused images.

However, as we described above, the captured light field is usually under-sampled and thus has aliasing. In the refocused image, the aliasing results in high frequency patterns in the defocused regions as shown in Figure 4.5. In the previous section we have presented an algorithm to synthesize distortion-free images. Here we utilize it to synthetically increase the angular resolution to reach the Nyquist rate. For example, the



**Figure 4.4:** *Digital refocusing of a transparent object.*

Nyquist rate of the scene in Figure 4.5 can be calculated according to the analysis in Chapter 2 and is roughly  $20 \times 20$ . We increase the angular resolution to  $25 \times 25$  and then perform digital refocusing, as shown in Figure 4.6. We can see that the out-of-focus objects are blurry now while the in-focus objects remain sharp.

### 4.3 Feature-Based Refocusing

We have demonstrated that the high quality refocusing effects can be created from the captured light field and estimated depth maps. However, specifying the correct focus is a difficult and tedious job for ordinary users. They need to understand the scene structure and perspective transformation. In most cases, the users have to fully sweep all possible focus settings and pick the images with the right focus.

Here we propose a more intuitive interface for this task. First, let's review the refocusing Equation 2.48. When the distance between the lens and the image plane  $F$



**Figure 4.5:** *Digital refocused image with the original angular resolution  $4 \times 4$ .*

is replaced with  $bF$ , the refocused image  $I_{bF}$  is defined by

$$I_{bF}(u) = \int L(u/b + (1 - 1/b)v, v) dv. \quad (4.1)$$

Because the distance between the lens and the image plane is changed, the magnification factor of the image is changed as well. It is desirable to compensate this change by magnifying the refocused image by a factor of  $b$ :

$$I'_{bF}(u) = I_{bF}(bu) = \int L(u + (b - 1)v, v) dv = \sum I_v(u + (b - 1)v). \quad (4.2)$$

We can see that given the refocus parameter  $b$ , the equation above is equivalent to first translating the light field image  $I_v$  by  $(b - 1)v$  and then accumulating all translated light field images together.

Furthermore, we know that for  $I_0$ , the translation vector is always 0. If we know that for a point on the object, its coordinate in  $I_v$  and  $I_0$  are  $u_v$  and  $u_0$ , respectively. When we want this object to be in-focus,  $u_v$  must move to  $u_0$ , and thus

$$\begin{aligned} u_v + (b - 1)v &= u_0, \\ b &= \frac{u_0 - u_v + v}{v}. \end{aligned} \quad (4.3)$$



**Figure 4.6:** *Digital refocused image with the boosted angular resolution  $25 \times 25$ .*

In other words, for each object to be refocused, if we can find a matched pair between  $I_0$  and another one of other light field images, the refocus parameter can be obtained using Equation 4.3.

The problem we need to solve now is to efficiently find the matched points on the specified object. We first extract the SIFT features [79] in  $I_0$  and  $I_v$ . The features in each image are stored in an independent k-d tree for efficient query. When the user move the cursor on the image, we continuously gather all the feature points in  $I_0$  around the cursor, and find their matched points in  $I_v$ . While the SIFT features are very robust, wrong matchings may still occur and the features may not belong to the same object. Therefore, we use RANSAC to find the optimal translational vector. Given the translational vector, Equation 4.3 can be easily solved.

The snapshots of the program are shown in Figure 4.7 and Figure 4.8. To illustrate the matching results we show the displacement vectors of the matched pairs in the images. The red vectors are the inliers in the RANSAC estimation and the green ones are the outliers. We can see that all outliers are successfully removed and the refocus setting is correct in all those results. Because we only perform feature matching between

two images, the computational cost of the matching very small. For the light field of angular resolution  $5 \times 5$  in Figure 4.8, the un-optimized program runs at 27fps on a desktop with a Intel 2.4Ghz CPU and 2GB memory.

## 4.4 Summary

In this chapter we have shown many applications of the light field. Using the programmable aperture, we can capture the light field and then generate many post-exposure effects, such as digital refocusing, which are impossible in the traditional photography. We have shown that the depth information can successfully alleviate the ghosting effect due to aliasing. The proper parameters for the anti-aliasing can be easily obtained using the analysis in Chapter 2.

We have also provided users a new interface to perform refocusing, which is also an application that did not exist in the past. By using feature matching, user can easily specify the focus plane in a drag-and-click manner.

Besides the applications shown above, there are many trivial applications that we do not present here, such as z-keying and natural image matting. We also believe that the fruitful light field data and the accurate depth information can enable applications that one can never imagine before.





**Figure 4.7:** *Feature refocusing (1/2).*



**Figure 4.8:** *Feature refocusing (2/2).*



# Chapter 5

## Efficient Processing

In this chapter, we will present two algorithms that are not only related to the light field analysis and acquisition, but are very important to the fundamental problems in computer vision and signal processing. We first present a new belief propagation method called tile-based belief propagation that can perform energy minimization to gigapixel images without sacrificing the performance and the convergence speed. The proposed method can be easily mapped to parallel processors. The experimental results show that the graphics processing unit (GPU) and VLSI implementations are 8 to 100 times faster than the CPU implementation.

We then propose a new noise-aware demultiplexing algorithm. The demultiplexing is formulated as a maximum a posteriori estimation. It can be casted into an L1 regularized energy minimization problem. The experimental results show that the proposed method is better than the traditional linear demultiplexing method.

### 5.1 Tile-Based Belief Propagation

Many problems in low-level visions can be formulated as a maximum a posteriori (MAP) estimation on a Markov random field (MRF). For each pixel or patch, we estimate an entity that best matches the observation (image) and the compatible to the prior knowledge, such as surface smoothness. To name a few applications, the entities can be the pixel values in a noise-suppressed image [90], the disparity vector of the disparity

map [85], [82], the source image indices of the stitched image [89], etc.

The MRF can be represented by an undirected graph  $G = (V, E)$  consisting of a set of  $n$  nodes (vertices)  $V = \{v_1, v_2, \dots, v_n\}$  and the edges  $E$  that connecting the neighboring nodes. Let  $N(\mathbf{p}) = \{\mathbf{q} \in V \mid (\mathbf{p}, \mathbf{q}) \in E\}$  denote the neighbor of  $\mathbf{p}$ . Each node is associated with a random variable  $X_{\mathbf{p}}$  taking values  $x_{\mathbf{p}}$  in some sample space  $\mathcal{X}_{\mathbf{p}}$ . The sample space is discrete in many applications. That is,  $\mathcal{X}_{\mathbf{p}} := \{0, 1, \dots, L_{\mathbf{p}} - 1\}$ .

If we consider the simple 4-connected or 8-connected neighboring system, the energy function<sup>1</sup> of a set of random variables  $\mathbf{X} = \{x_{\mathbf{p}} \mid \mathbf{p} \in V\}$  is

$$E(\mathbf{p}) = \sum_{\mathbf{p} \in V} E_d(x_{\mathbf{p}}) + \sum_{(\mathbf{p}, \mathbf{q}) \in E} E_s(x_{\mathbf{p}}, x_{\mathbf{q}}). \quad (5.1)$$

where  $E_d$  is an unary data term and  $E_s$  is a pairwise smoothness term.

As we have mentioned in Chapter 3, the performance of the application depends on the definitions of the energy terms and the chosen optimization algorithms. Finding the set of random variables that minimizes this energy function is generally an NP-hard problem [85]. However, recently many algorithms have been proposed to efficiently find the local minimal solution in polynomial time with guaranteed optimum property. Those algorithms includes alpha-expansion graph cut [85], loopy belief propagation [90], tree-reweighted message passing [91] and many other variants [95].

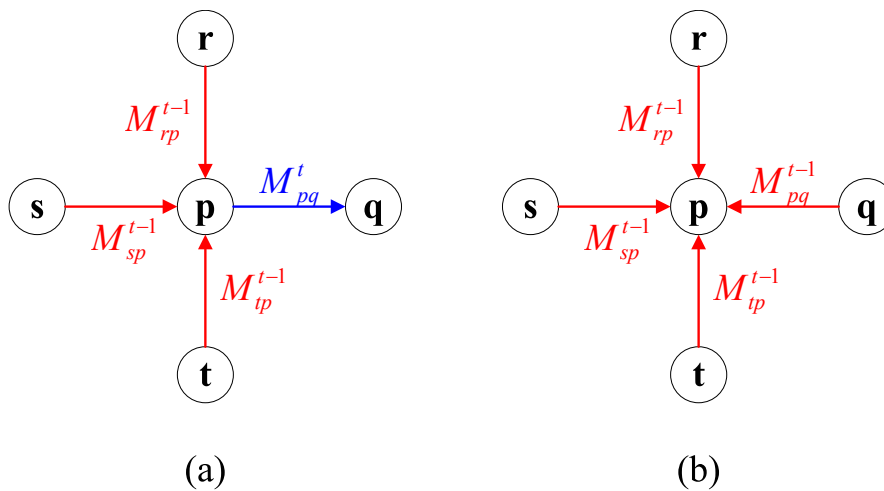
While the single-thread software implementation of these algorithms works efficiently and effectively, the hardware implementation on VLSI or other parallel processors is much less addressed in the past. Also those global optimization algorithms usually require a great amount of memory and multi-pass data access, which are both very expensive for memory- and bandwidth-limited embedded systems.

### 5.1.1 Belief Propagation: Preliminary

Belief propagation (BP) is one of the most successful MAP-MRF optimization algorithms. Given a graph  $G$  which is defined above, each node passes the outgoing *messages*, which is updated according to the incoming messages and the energy functions, to its neighbors. For example, in the min-sum/max-product BP, the message sent by  $\mathbf{p}$

---

<sup>1</sup>The negative log of the joint probability density function or the potential functions of cliques.



**Figure 5.1:** The message and belief construction.

(a) An outgoing message is constructed using three incoming messages. (b) The belief is constructed using all incoming messages.

to  $q$  at the  $t$ -th iteration is  $m_{(p,q)}^t$ , which is a  $L$ -D vector<sup>2</sup>. The message is computed as (Figure 5.1(a)):

$$m_{(p,q)}^t(x_q) = \min_{x_p \in \mathcal{X}} (E_s(x_p, x_q) + E_d(x_p) + \sum_{s \in N(p) \setminus q} m_{(s,p)}^{t-1}(x_p)), \quad (5.2)$$

where  $N(p) \setminus q$  denotes the neighbors of  $p$  other than  $q$ . After  $T$  iterations a belief vector  $b_p$  is computed as (Figure 5.1(b))

$$b_p(x_p) = E_d(x_p) + \sum_{s \in N(p)} m_{(s,p)}^T(x_p). \quad (5.3)$$

Finally, at each node a label  $x_p^* = \arg \min b_p(x_p)$  is chosen independently. The intuition behind the belief propagation is as follows. The incoming messages to  $p$  give the suggestions from other nodes about being assigned a label  $x$ , and the outgoing messages combine the opinion and the self-judgment into a new suggestion to other nodes. Through iterations the nodes far away from each can more or less receive the suggestions from each other. As a result, a global optimal decision can be obtained.

It has been shown that when the graph contains no loop, BP can find the global optimal solution, and when the graph contains loops, BP can find a strong local optimal

<sup>2</sup>We assume that  $L_p = L$  for all  $p$  and all random variables use the same sample space.

solution<sup>3</sup>. While graph cut can sometimes find a better solution, it requires the smoothness to be submodular. On the contrary, BP has no such restriction. The theoretical analysis of the BP can be found in [96], [97], [98].

### 5.1.2 Belief Propagation: Cost Analysis

It can be easily seen that BP has several advantages for hardware implementation. First, it is highly parallel. Each node can load the messages from the previous iteration, operate independently, and generate new messages. Second, it only use simple operations such as additions and comparisons. Therefore the processing element (PE, or *kernel* in GPU programming) can be made very small. Third, the memory access is regular. On the contrary, while the graph cut can be parallelized by using the push-relabel algorithm as shown in [99], each PE has many branches and thus is much more complex.

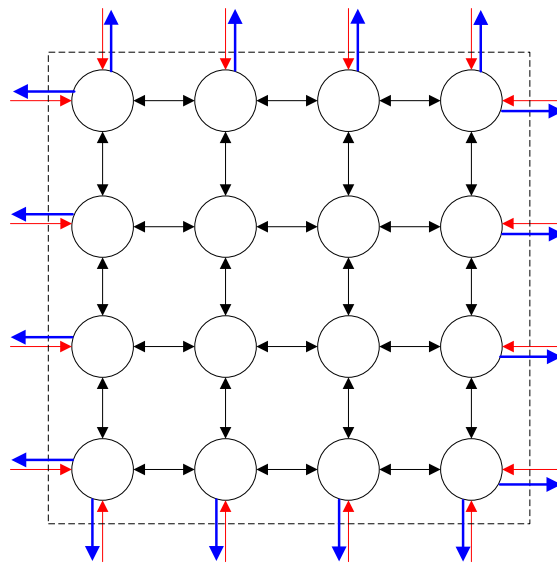
However, BP cannot be efficiently implemented in hardware due to the great amount of memory, bandwidth, and computation. We now analyze these three issues in the following.

**Memory:** Like all other global optimization algorithms, we need to store  $nL$  data terms. The smoothness terms, which are usually globally uniform, can be analytically defined or tabularized. Also, we need to store  $4nL$  messages for the 4-connected MRF. Therefore, to perform BP we need to store  $5nL$  messages. For example, in stereo estimation of a 720p image with 100 disparity values and assume each element takes one byte, BP totally takes 460,800,000 bytes (439.45 MB), 500 times the image size. In [100] the authors propose to compress the messages using various techniques. However, the number of the messages is unaffected; the decompression requires extra computation and latency. In [101] the image is segmented into many independent patches. The memory can be greatly reduced but the global optimality is totally lost. The results of the so-called block-based BP would be poor in many applications.

**Bandwidth:** The bandwidth of BP depends on the scheme of the message passing. Using the most efficient BPM scheme in [102], each node loads three messages and the data terms and saves one new message. In each iteration each node is processed 4 times,

---

<sup>3</sup>Remember that finding the global optimal solution is NP-hard.



**Figure 5.2:** *Belief propagation for a single tile.*

so totally BP requires  $4 \times n \times (3 + 1 + 1) \times L = 20nL$  data transferring in one iteration. Using the 720p stereo problem as the example again and assume BP converges when  $T = 50$ , for 25fps data, the bandwidth is 2,304,000,000,000 (2.0 tera) bytes per second. Obviously this daunting bandwidth requirement is infeasible for the existing hardware, as described in [103].

**Computation:** Although in constructing a new message (Equation 5.2) only additions and comparisons are used, the number of operations is large. The arithmetic complexity of the direct implementation is  $O(L^2)$ . For some analytic smoothness terms such as linear and quadratic model, the message can be constructed in  $O(L)$  by using the min-convolution operation [104]. However as we will see, this method is restricted and introduces a large latency.

Our contribution is to make BP one step toward efficient hardware implementation by developing a new message passing scheme and a new fast message construction algorithm.

### 5.1.3 Proposed Algorithm

In this section we propose a new message passing scheme to remove the memory and bandwidth bottleneck. Let's review the construction of the message (Figure 5.1(a) and

Equation 5.2), we find that an outgoing message is uniquely determined by the incoming messages, the data terms, and the smoothness terms. If these entities are known, the outgoing message is a redundant element that is only used to drive the propagation.

This property is also observed in [104]. In that work, the nodes are separated into two non-connecting sets<sup>4</sup> and in each iteration, only the messages toward one set is calculated. Therefore the storage for the message can be halved. However, this scheme cannot be applied to BPM.

We find that this property can be further exploited. Consider a tile consisting of  $B \times B$  nodes ( $B = 4$  in Figure 5.2). We can see that, for this tile, the knowledge about the outside world purely comes from the incoming messages (red arrows), no matter how those messages are constructed and if they are correct. Using the incoming messages, the data terms of the nodes, and the smoothness terms, we can perform belief propagation in this tile, and generate the belief and labels for the nodes in the tile without accessing other data. In other words, when we focus on optimization of a small tile, we can drop all data terms and messages of the outside but only those messages passing to the tile.

If we want to perform belief propagation outside the tile, we need the knowledge about the tile. However, the outgoing messages (blue arrows in Figure 5.2) are enough to give this knowledge. We do not need to know the data terms in the tile, or even the messages between the nodes in the tile (black arrows in Figure 5.2).

We can generalize this concept to the whole image. If we segment the image into many non-overlapping tiles and perform the BP independently, the global optimal property is lost. However, if we maintain the messages bridging the neighboring tiles, when the belief propagation in the tile is still driven by the messages from other tiles. Therefore the global optimality shall be preserved.

According to this idea, we propose a tile-based belief propagation. The pseudo code of the algorithm is given in Table 5.1 and the flow is illustrated in Figure 5.3. We first scan the image in a raster scan order. At each time a  $B \times B$  tile is processed. We load all messages, which we call *boundary messages*, sending to the tile, and the data terms

---

<sup>4</sup>That is, a node in one set does not connect to another node of the same set.

```

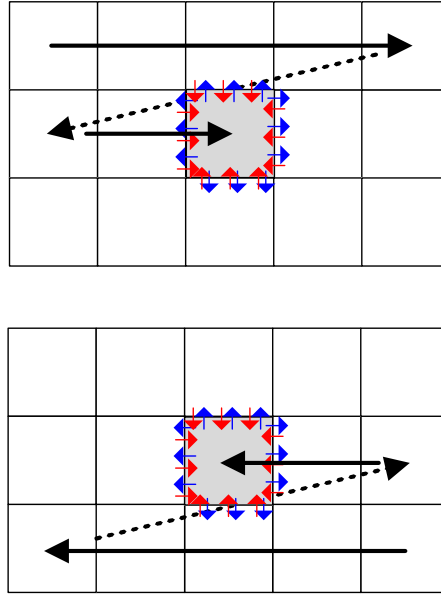
function  $\{x_p^*\} \leftarrow \text{TileBasedBP}(E_d, E_s, B, T_o, T_i)$ 


---


1   for  $t_o=1, \dots, T_o$ 
2       loop through all tiles in a raster scan order
3           Load  $\{M_{pq}^{t_o-1}\}$  for  $p \notin C$  and  $q \in C$ ; //C is the current tile.
4           Load  $E_d(x_p)$  for  $p \in C$ ;
5           for  $t_i=1, \dots, T_i$ 
6                $\{M_{pq}^{t_i}\} \leftarrow \text{BPinOneTile}(\{M_{pq}^{t_i-1}\}, E_d, C)$ ;
7               Store  $\{M_{pq}^{T_i}\}$  for  $p \in C$  and  $q \notin C$ ;
8           loop through all tiles in an inverse raster scan order
9               Load  $\{M_{pq}^{t_o-1}\}$  for  $p \notin C$  and  $q \in C$ ;
10              Load  $E_d(x_p)$  for  $p \in C$ ;
11              for  $t_i=1, \dots, T_i$ 
12                   $\{M_{pq}^{t_i}\} \leftarrow \text{BPinOneTile}(\{M_{pq}^{t_i-1}\}, E_d, C)$ ;
13              if  $(t_o = T_o)$  Obtain  $b(x_p)$  and  $\{x_p^*\}$  for  $p \in C$ ;
14              else Store  $\{M_{pq}^{T_i}\}$  for  $p \in C$  and  $q \notin C$ ;
15  return  $\{x_p^*\}$ ;

```

**Table 5.1:** The pseudo code of the tile-based belief propagation.



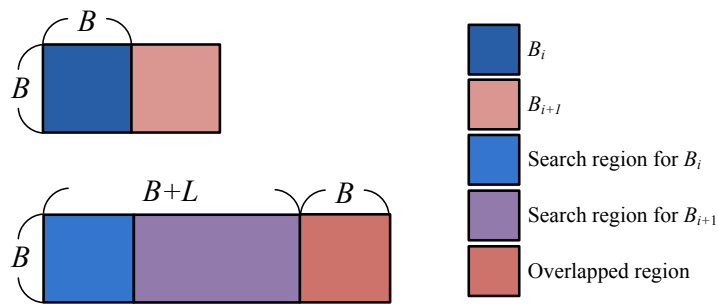
**Figure 5.3:** *The flow of the tile-based belief propagation.*

of the nodes from the memory. Then we perform belief propagation within this tile (the `BPinOneTile` function in the pseudo code). Note that the boundary messages are fixed. After a  $T_i$  iterations or after the messages in the tiles are stable, we drop all messages inside the tiles and store all outgoing boundary messages to the memory.

However, if we only scan the image once, the results of the optimization are biased. This is because for all tiles, the boundary messages from the right and the bottom tiles are missed. Therefore, we re-scan the image in an inverse raster scan order. In this way the bias is compensated. It should be noted that the bias problem must be addressed in most BP algorithms.

Finally, it should be emphasized that the tile-based BP is different from the hierarchical BP [104] in many perspectives. The hierarchical BP merges many nodes in the graph to build a smaller graph and then performs BP in a hierarchical fashion, from small graphs to the large graphs. In this way the convergence speed can be increased. On the contrary, in the tile-based BP, the nodes in the tile are not merged into a single one, and thus the distinctness of the nodes is preserved. Each tile is analogous to a *filter box*. The input boundary messages are non-linearly filtered by the MRF in the tile into the output boundary messages.





**Figure 5.4:** Level-C data reuse for calculating the data terms.

$B_i$  and  $B_{i+1}$  are two successive blocks in the left image. Because most of their search region in the right image is overlapped, besides the block  $B_{i+1}$  itself, only  $B^2$  pixels are needed to be loaded.

### 5.1.4 Cost Analysis of the Proposed Algorithm

Before we show the performance of the tile-based BP, we first show its advantages for hardware implementation. Specifically, we analyze the memory and bandwidth requirement of the proposed algorithm. We will see both costs can be greatly reduced, and also the tile-based operation can enable regular pipeline to further reduce the bandwidth and memory requirement.

**Memory:** We do not need to store the messages inside the tile. In each new iteration, the messages inside can be re-generated from the boundary messages and data terms. Therefore, the off-chip storage of the messages becomes:

$$\frac{n}{B^2} \times 4BL = \frac{4nL}{B}. \quad (5.4)$$

The reduction is a factor of  $B$ . Therefore, the bigger the tile, the larger memory reduction we obtain. For  $B = 32$ , the memory for the messages becomes 0.03125% of the original size.

**Bandwidth:** The bandwidth of the tile-based BP is

$$2(nL + \frac{4nL}{B} + \frac{4nL}{B})T_o = (2nL + 16\frac{nL}{B})T_o, \quad (5.5)$$

where the first term is for the data term, the second term is for the incoming boundary messages of tiles, and the third term is the outgoing boundary messages of tiles.

		Original BP	Proposed tile-based BP				Proposed tile-based BP + data reuse			
Block(tile)-size ( $B^2$ )		—	$8^2$	$16^2$	$32^2$	$64^2$	$8^2$	$16^2$	$32^2$	$64^2$
Internal memory	Data terms	16	1,024	4,096	16,384	65,536	1,280	4,864	18,944	74,752
	Messages	64	4,096	16,384	65,536	262,144	4,096	16,384	65,536	262,144
	Total	80	5,120	20,480	81,920	327,680	5,120	20,480	81,920	327,680
	Factor (to the same tile size)		100%	100%	100%	100%	125%	119%	116%	114%
External memory	Data terms	4,915,200	4,915,200				0			
	Messages	19,660,800	2,457,600	1,228,800	614,400	307,200	2,457,600	1,228,800	614,400	307,200
	Total	24,576,000	7,372,800	6,144,000	5,529,600	5,222,400	2,457,600	1,228,800	614,400	307,200
	Factor (to original BP)	100.00%	30%	25%	22.5%	21.25%	10%	5%	2.5%	1.25%
Bandwidth	Data terms	19,660,800	4,915,200				614,400			
	Messages	78,643,200	4,915,200	2,457,600	1,228,800	614,400	4,915,200	2,457,600	1,228,800	614,400
	Total	98,304,000	9,830,400	7,372,800	6,144,000	5,529,600	5,529,600	3,072,000	1,843,200	1,228,800
	Factor (to original BP)	100%	10%	7.5%	6.25%	5.625%	5.625%	3.125%	1.875%	1.25%

**Table 5.2:** The comparison of the memory and bandwidth consumption.

We can see that now the memory and the bandwidth costs are dominated by the storage of the data terms. However, because the algorithm is tile-based, several pipeline and data reuse techniques used in the video compression can be applied [105] to achieve lower bandwidth by using slightly more internal memory.

For example, in stereo estimation, the data terms  $E_d$  is calculated from the intensity difference of the left and right correspondences. Because the proposed algorithm is performed in tile-based, when performing BP for one tile, we can load the images and construct the data terms of the next tile. Using the level-C memory reuse scheme [105], we only need to load  $B^2$  bytes for calculating the data cost of one tile, as shown in Figure 5.4. Therefore, the external memory for the data terms are eliminated and the overall bandwidth decreases from  $(2nL + 16\frac{nL}{B})T_o$  to:

$$2(2n + \frac{8nL}{B})T_o = (4n + 16\frac{nL}{B})T_o, \quad (5.6)$$

where the first term is for the image pixels.

Using this technique, we can further reduce the memory and bandwidth consump-

tion, as shown in Table 5.2. Compared with to original BP, the external memory is reduced by a factor of 10 to 80 and the bandwidth is reduced by a factor of 18 to 80.

The analysis above does not consider the convergence property. In BPM [102], one usually have to perform 30 to 50 iterations to obtain results. In the tile-based BP, there are two iteration coefficients (Table 5.1), the maximal inner iteration  $T_i$  and the maximal outer iteration  $T_o$ . The first coefficient  $T_i$  increases linearly with the tile size. This is because if the BP within the tile is not converged, the outgoing boundary messages would not be representative enough, and thus the overall performance would degrade, as we will show in the next subsection. However,  $T_i$  does not affect the memory and the bandwidth costs. Usually,  $T_o = 2$  or 3 gives a very nice results.

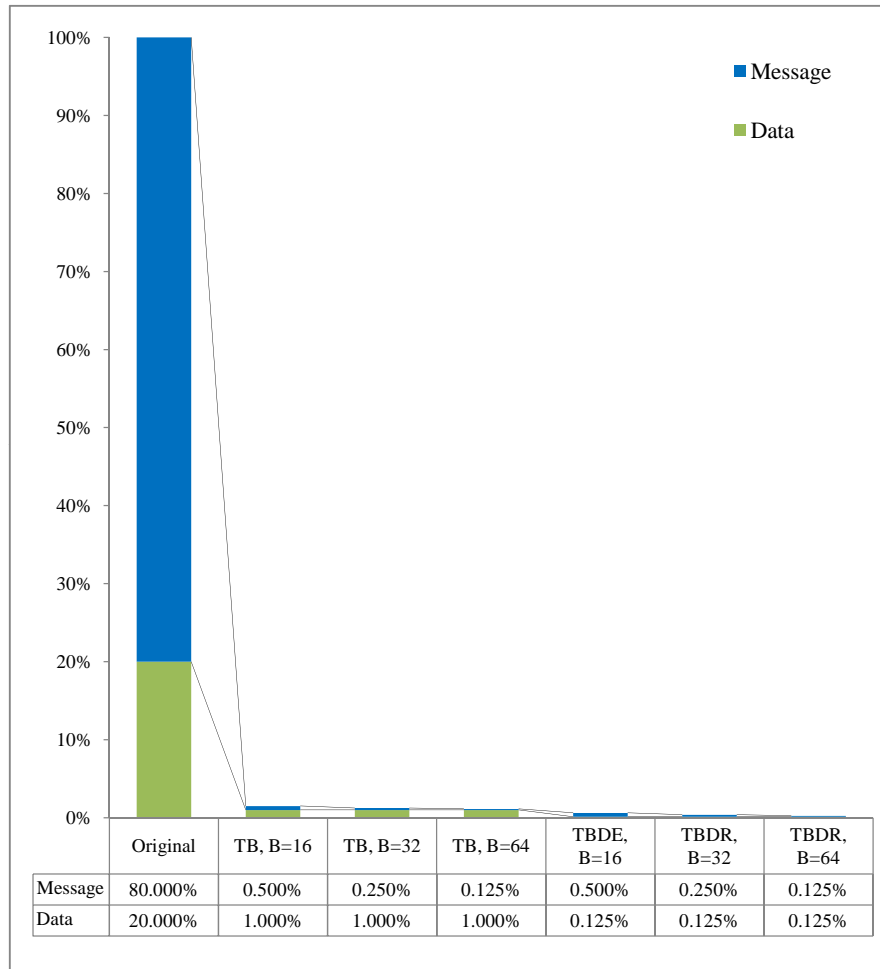
The bandwidth cost after the iteration number is considered is shown in Figure 5.5. We can see that our algorithm only requires roughly 1% of the original bandwidth. Using data reuse, the bandwidth can be further reduced to 0.25%, only 1/400 of the original cost.

### 5.1.5 Performance of the Proposed Algorithm

We have shown that the proposed algorithm can significantly reduce the memory and bandwidth. Here we use the stereo estimation to show that performance-wise, our algorithm is much better than the block-based BP [101] and comparable to the original BP.

In the experiments we use the stereo dataset on the Middlebury benchmark website. Since our goal is to compare the performance of difference BP algorithms, we adopt the simple data and smoothness term definitions according to the definitions in [85]. The parameters such as the weighting coefficients and thresholds are fixed in the experiments.

Different BP algorithms require different number of iterations. To reach the performance limitation of the algorithms, we empirically choose the number of the iterations to be large enough such that all algorithms converge for all dataset. For the original BP (BPM) and the block-based BP,  $T$  is 50; for the tile-based BP, inner iteration  $T_i$  is set as the block size  $B$ , and  $T_o$  is 10 (In fact, setting  $T_o = 3$  gives visually plausible results).

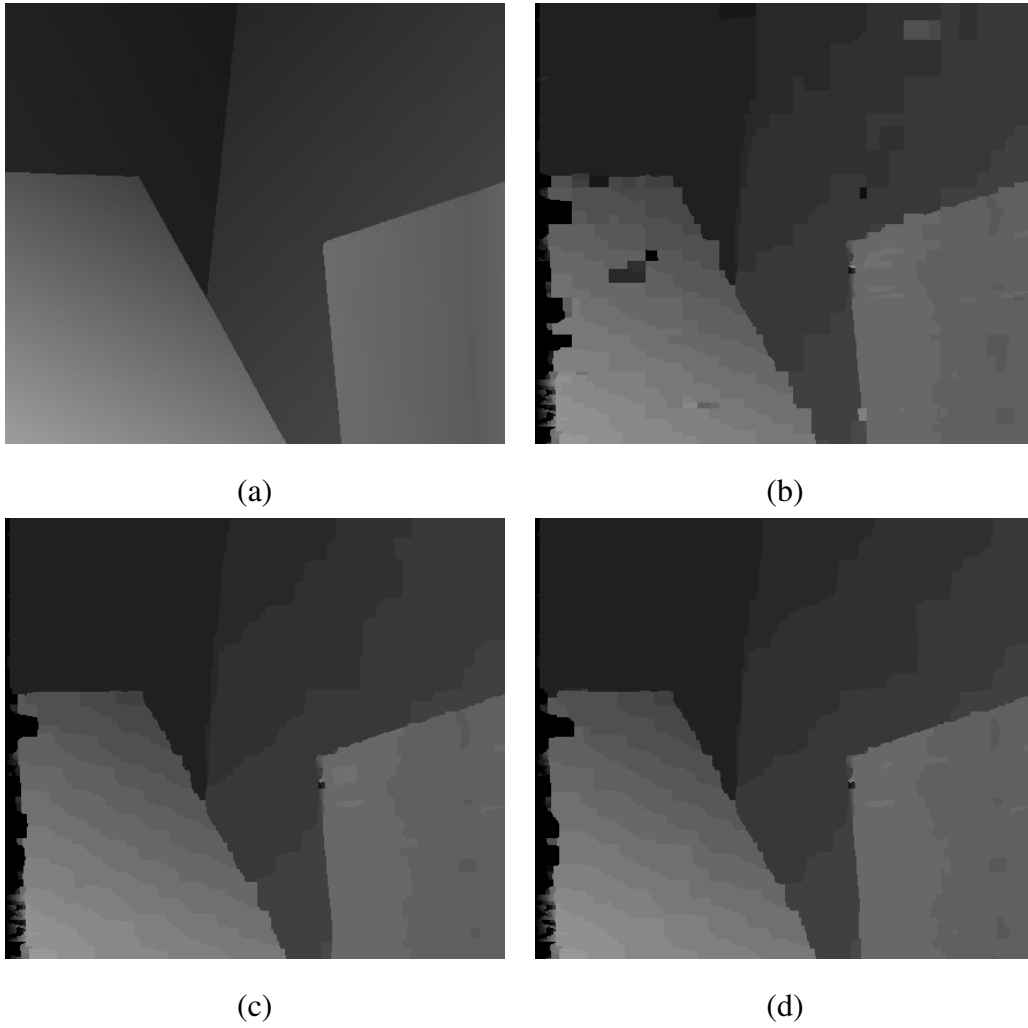


**Figure 5.5:** *The bandwidth consumption considering the iteration numbers.*  
*In this figure the original BP, the tile-based (TB) BP, and the TB BP with data reuse (TBDR) of different block sizes ( $B = 16, 32$ , and  $64$ ) are compared.*

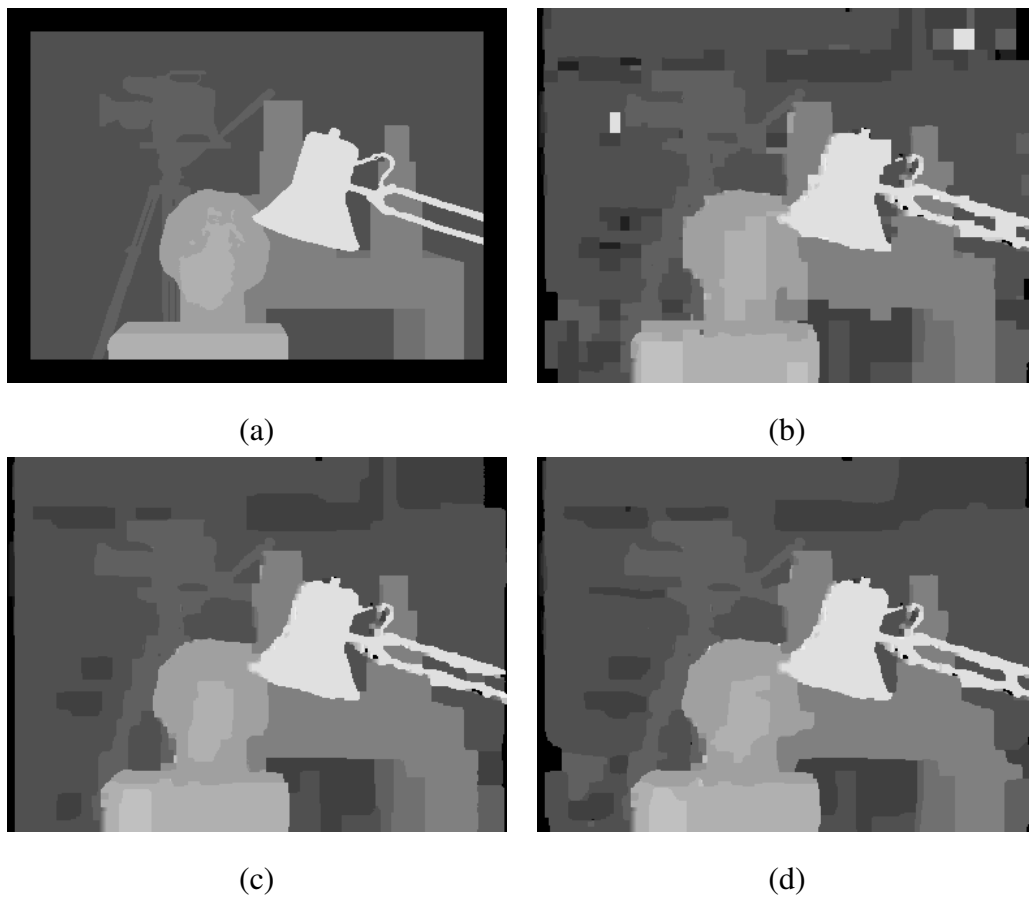
		<i>Tsukuba</i>	<i>Teddy</i>	<i>Cones</i>	<i>Venus</i>	Average
Original BP		333466	2227224	2466383	802006	
Block-based BP	$B = 16$	386849	2393924	2646827	935663	
		16.01%	7.48%	7.32%	16.67%	11.87%
	$B = 32$	351942	2308736	2554895	853686	
		5.54%	3.66%	3.59%	6.44%	4.81%
	$B = 64$	350840	2271426	2520843	826655	
		5.21%	1.98%	2.21%	3.07%	3.12%
Tile-based BP	$B = 16$	328257	2233962	2477568	814453	
		<b>-1.56%</b>	0.30%	<b>0.45%</b>	<b>1.55%</b>	0.19%
	$B = 32$	330647	2230923	2478763	815300	
		<b>-0.85%</b>	<b>0.17%</b>	0.50%	1.66%	0.37%
	$B = 64$	341915	2238193	2483135	817561	
		2.53%	0.49%	0.68%	1.94%	1.41%

**Table 5.3:** The energy of the solutions using different BP algorithms.

The percentages indicate the increased energy to the energy of the original BP. The bold numbers are the energies that are lower than the original BP. The red numbers are the best results.



**Figure 5.6:** *The disparity maps of Venus from different BP algorithms.*  
 (a) *Ground truth, (b) the block-based BP, (c) the original BP, and (d) the tile-based BP.*  
*The block size ( $B$ ) is 32.*



**Figure 5.7:** *The disparity maps of **Tsukuba** from different BP algorithms.*  
 (a) *Ground truth*, (b) *the block-based BP*, (c) *the original BP*, and (d) *the tile-based BP*.  
 The block size ( $B$ ) is 16.

The energy obtained from different BP algorithms are shown in Table 5.3. We can see that the performance of our algorithm is very close to that of the BPM. The achieved energies can sometimes be lower than that obtained by the original BP. On average, the energy variation is 0.19% for  $B = 16$ , 0.37% for  $B = 32$ , and 1.41% for  $B = 64$ . On the contrary, the energy from the block-based BP, which removes all the edges between the blocks, is very bad. All the results are much worse than those of the proposed method.

We also visually compare the resulting disparity maps from different BP algorithms, as shown in Figure 5.6 and 5.7. We can see that the disparity maps obtained by the tile-based BP and those obtained by the original BP are very similar. On the contrary, the disparity maps obtained by the block-based BP are very noisy. Many textureless regions in the image rely on the smoothness constraint to guide the disparity estimation. However, in the block-based BP the smoothness constraint across the blocks are removed, we cannot infer the disparity values for the isolated textureless blocks. It should be noted that we use simple data and smoothness terms in the evaluation here so the results are worse than the state-of-the-art algorithms. However, while these state-of-the-art algorithms use more complex energy functions in the MRF, they all use belief propagation to perform the optimization. Therefore, our technique can be easily adopted in their algorithm for efficient hardware implementation.

### 5.1.6 Discussion

The proposed tile-based BP algorithm can successfully reduce the memory and bandwidth requirement by a factor of 80 to 400 and still retain the property of the global optimality. We find that the tweaking of the parameters such as the damping factor in the message passing, plays a more important role in the convergence speed and the final energy value. Also, the tile-based BP can also be applied with other BP algorithms. For example, we can apply tile-based BP in each level of the hierarchical BP [104] to reduce the memory and bandwidth cost. The initial experimental results suggest that the performance is unaffected.



BuildMessageOriginalParallel(Mq, Mr, Ms, Mu, D, V)	<i>Latency</i>
<b>for</b> l=0,...,L-1 <b>in parallel</b>	$2D_A$
H[l] = (Mu[l]+Mr[l])+(Ms[l]+D[l]);	
<b>end</b>	
<b>for</b> l = 0,...,L-1 <b>in parallel</b>	$D_A +$
<b>for</b> m = 0,...,L-1 <b>in parallel</b> 1 ADD	$\log_2(L)D_C$
M_part[l][m] = H[m] + $\lambda$ V[m][l];	
<b>end</b>	
<b>end</b>	
<b>for</b> m = 0,...,L-1 <b>in parallel</b> $\log_2(L)$ CMP	
Mq[l] = min(M_part[l][m], Md[l]);	
<b>end</b>	

**Table 5.4:** The pseudo code of the message construction in original BP.

## 5.2 Fast Message Construction

The tile-based BP has greatly reduced the bandwidth and memory costs of the BP algorithm. However, the computational complexity of the message construction is still a bottleneck to be solved. Let's repeat the message construction equation here:

$$m_{(\mathbf{p}, \mathbf{q})}^t(x_{\mathbf{q}}) = \min_{x_{\mathbf{p}} \in \mathcal{X}} (E_s(x_{\mathbf{p}}, x_{\mathbf{q}}) + E_d(x_{\mathbf{p}}) + \sum_{\mathbf{s} \in N(\mathbf{p}) \setminus \mathbf{q}} m_{(\mathbf{s}, \mathbf{p})}^{t-1}(x_{\mathbf{p}})). \quad (5.7)$$

Note that the last term in the summation can be reused for constructing different entities. Therefore, constructing a  $L$ -D message needs  $L(3 + L)$  additions and  $L(L - 1)$  comparisons, and the computational complexity for the whole image is proportional to  $O(nL^2)$ . In stereo estimation of a VGA-sized image pair with 60 disparity values, a single BPM iteration requires 4.64G additions and 4.35G comparisons.

The pseudo code of the maximally parallelized version of the original message construction is shown in Table 5.4. The function loads three incoming messages Mu, Mr, Ms, the local data term D, and the smoothness term V (i.e., the tabularized  $E_s(x_{\mathbf{p}}, x_{\mathbf{q}})$ ),

BuildMessageEfficient(Mq, Mr, Ms, Mu, D, V)	<i>Latency</i>
<b>for</b> l=0,...,L-1 <b>in parallel</b>	$2D_A$
H[l] = (Mu[l]+Mr[l])+(Ms[l]+D[l]);	
<b>end</b>	
<b>for</b> l = 1,...,L-1 <b>L-1 ADD and L-1 CMP</b>	$2(L-1) \times (D_A + D_C)$
M[l] = min(M[l-1]+ $\lambda$ , M [l]);	
<b>end</b>	
<b>for</b> l = L-2,...,0 <b>L-1 ADD and L-1 CMP</b>	
M[l] = min(M[l+1]+ $\lambda$ , M [l]);	
<b>end</b>	
<b>//Minimum truncation phase</b>	$\log_2(L)D_C + D_A + D_c$
M_min=M[0];	
<b>for</b> l = 1,...,L-1 <b>in parallel</b> $\log_2(L)$ <b>CMP</b>	
M_min = min(Md[l] , M_min);	
<b>end</b>	
<b>for</b> l = 0,...,L-1 <b>in parallel</b> <b>1 ADD and 1 CMP</b>	
Mq[l] = min(M[l], M_min+ $\lambda T$ );	
<b>end</b>	

**Table 5.5:** The pseudo code of the existing fast message construction algorithm.

BuildMessageProposed(Mq, Mr, Ms, Mu, D, V)	<i>Latency</i>
<b>for</b> l=0,...,L-1 <b>in parallel</b>	$2D_A$
$H[l] = (Mu[l]+Mr[l])+(Ms[l]+D[l]);$	
<b>end</b>	
<b>//The local minimum cost comparison phase</b>	$D_A + \log_2(2T-1)D_C$
<b>for</b> l = 0,...,L-1 <b>in parallel</b> <b>1 ADD</b>	
<b>for</b> m = -T+1,...,T-1 <b>in parallel</b>	
$M\_part[l][l+m] = H[l+m] + \lambda V[l-m];$	
<b>end</b>	
<b>end</b>	
<b>for</b> l = 0,...,L-1 <b>in parallel</b>	
<b>for</b> m = -T+1,...,T-1 <b>in parallel</b> $\log_2(2T-1)$ <b>CMP</b>	
$M[l] = \min(M\_part[l][l+m], M[l]);$	
<b>end</b>	
<b>end</b>	
<b>//The global minimum threshold phase</b>	$\log_2(L)D_C$
<b>for</b> l = 0,...,L-1 <b>in parallel</b> $\log_2(L)$ <b>CMP</b>	
$M\_min = \min(M\_part[l][l], M\_min);$	
<b>end</b>	
<b>//Global truncation phase</b>	$D_A + D_C$
<b>for</b> l = 0,...,L-1 <b>in parallel</b> <b>1 ADD and CMP</b>	
$Mq[l] = \min(M[l], M\_min + \lambda K);$	
<b>end</b>	

**Table 5.6:** The pseudo code of the proposed fast, hardware-oriented message construction algorithm.

and generates the outgoing message  $M_q$ . The for loops that can be performed in parallel are annotated with the keyword “in parallel”. Let  $D_A$  and  $D_C$  denote the latency of the adder and comparer, respectively. The latency of each sub-routine is annotated with the blue words, and the overall latency is shown at the right.

In original BP, the messages can be calculated by first generating all  $L^2$  hypotheses (M\_part in Table 5.4), and then finding the  $L$  minimal final entities from them in parallel. The latency of this operation is  $\log_2(L)D_C$ . However, it requires  $L^2$  temporary registers and operators. If we give up the parallelism, it takes  $O(L^2)$  cycles.

The software-efficient algorithm in [104] is shown in Table 5.5<sup>5</sup>. The entities are updated sequentially in two forward-backward passes and then truncated by the threshold. The sequential operation is suitable for software but causes a great amount of latency and low throughput in hardware. In hardware implementation, it is actually much slower than the parallelized original BP.

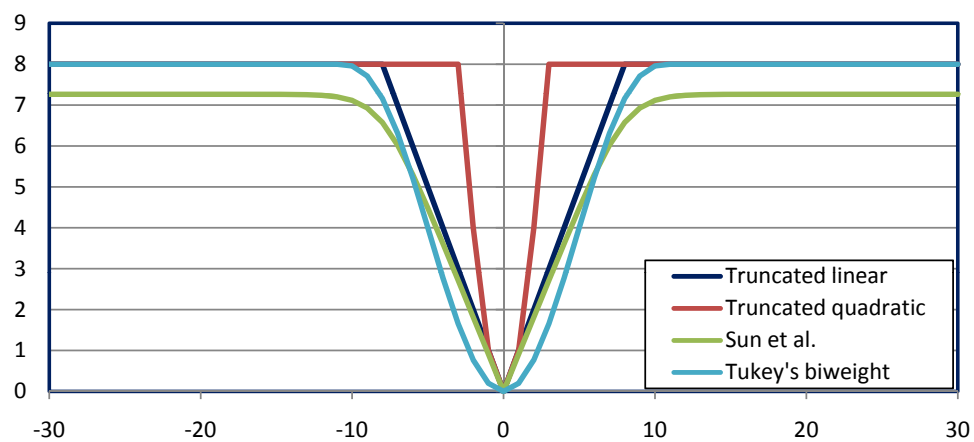
While the algorithm in [104] only applies to the linear or quadratic smoothness terms, we find that in many stereo estimation applications or many other low-level vision applications, the smoothness terms are usually *robust to outliers* [106], [107], [108], [84]. That is, the smoothness terms grows as the value of  $|x_p - x_q|$  increases, but as the value becomes larger, the increasing rate gradually decreases. After the value reaches a threshold, the smoothness term becomes constant. This is because when two labels becomes too distinct from each other, they should have no correlation so a adding constant penalty would be more proper than adding an arbitrary large penalty.

Several typical robust functions used in disparity estimation, denoising, and tone mapping applications are shown in Figure 5.8. We can see that after a threshold  $T$ , the robust functions all become constant. We attempt to utilize this property in the message construction. Let  $H(x_p) = E_d(x_p) + \sum_{s \in N(p) \setminus q} m_{(s,p)}^{t-1}(x_p)$  and assume the smoothness term is a robust function  $E_s(x_p, x_q) = \rho(x_p - x_q)$ , then Equation 5.7 becomes

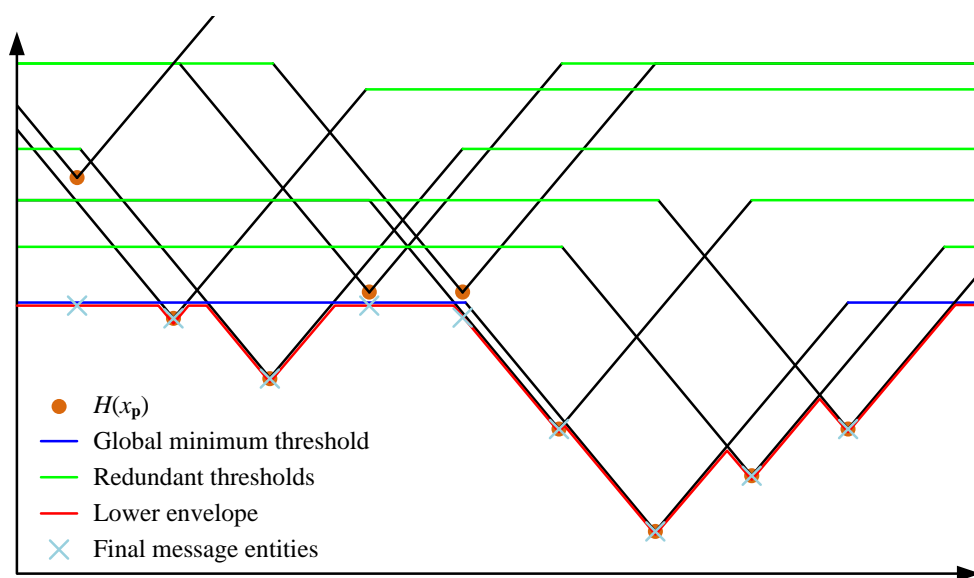
$$m_{(p,q)}^t(x_q) = \min_{x_p \in \mathcal{X}} (\rho(x_p - x_q) + H(x_p)). \quad (5.8)$$

We can consider the hypotheses generated by a single  $H(x_p)$  as a signal and plot all signals together, as shown in Figure 5.9. Each signal has a single peak is constant when

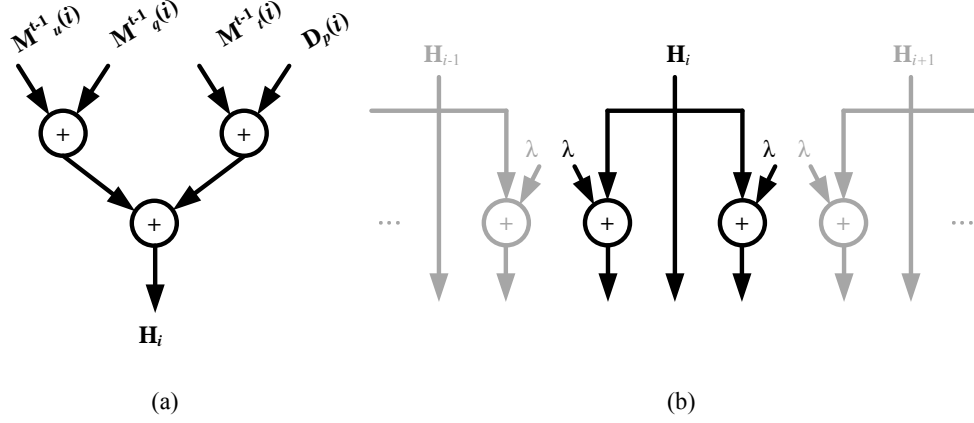
<sup>5</sup>This is the version for the linear truncated model. Please refer to [104] for the quadratic model.



**Figure 5.8:** The robust functions commonly used as the smoothness terms in MRF.



**Figure 5.9:** The hypotheses and the final message.

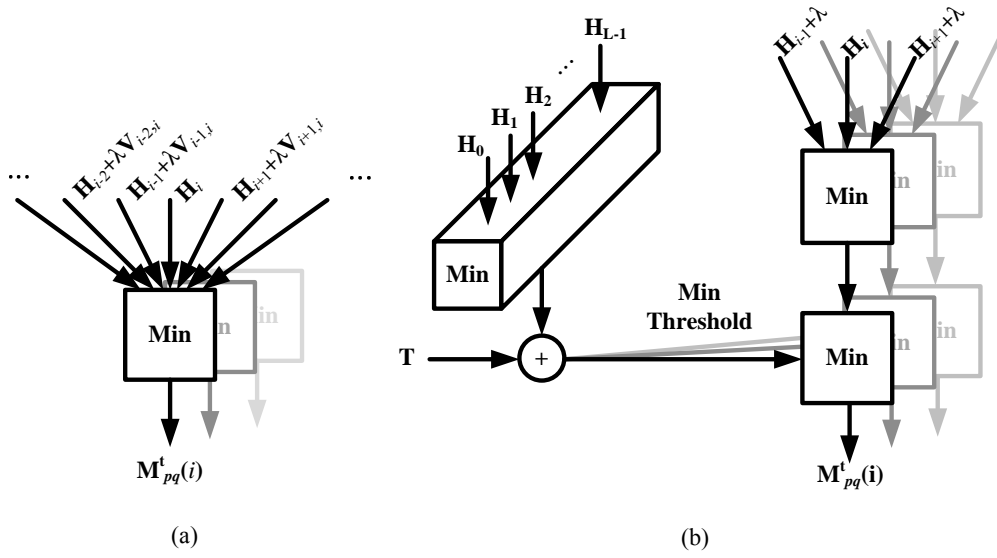


**Figure 5.10:** (a) The generation of  $H[i]$ . (b) The generation of hypotheses.

the index is  $T$  away from the peak.

The final message is determined by the lower envelope (shown in red) of all signals. One important observation is that *only the global minimum threshold generated by the smallest  $H(x_p)$  will present in the envelope*. Therefore, all other constant hypotheses can be ignored. Therefore, the number of comparisons can be reduced from  $O(L^2)$  to  $O((2T - 1)L + L) = O(2TL)$  or  $O(L)$  when  $T$  is a small constant. In many applications,  $T$  much smaller than  $L$  and thus the computation is greatly reduced. Note that this reduction of the computation does not change the result of the message construction, and thus it is applicable to all robust functions.

The pseudo code of the proposed algorithm is shown in Table 5.6. The smoothness term table becomes an one dimensional array, and  $K$  is the constant penalty value. In the local minimum cost comparison phase, all non-constant hypotheses are generated and compared. The result is a  $L$ -D temporal message. In the global minimum threshold phase, we find the smallest  $H(x_p)$  (M\_min). The final message is the minimum of the temporal message and  $M_{\min} + \lambda K$ . It should be noted that these two phases can be processed in parallel.



**Figure 5.11:** The processing elements.

(a) The original PE and data flow. (b) The proposed PE and data flow. The grey items have the same structure but different inputs and work in parallel.

UMC90nm@10ns L=64	Original	Proposed	Reduction ratio
Gate count	533.8k	30k	94.38%

**Table 5.7:** Comparison of gate count.

### 5.2.1 Hardware Implementation

We implement a VLSI processing element based on the proposed algorithm. In the PE, we set  $T = 2$ . The generation of  $H[i]$  and the non-constant hypotheses are shown in Figure 5.10. Each  $H[i]$  can be generated by an adder tree and all hypotheses are generated in parallel. The PE's for the next step in the message construction is shown in Figure 5.11. We can see that in the original BP, each PE has many fan-in and thus the comparer tree is very large. On the contrary, in the proposed algorithm, each PE only has 3 input and thus the latency and size can be very small.

We synthesize the circuits by using the UMC90nm library with the critical path constraint 10ns. This means that the synthesized circuit can construct a message within

10ns. Compared with the original message computation method, the proposed method can reduce 94.4% of the gate count, as shown in Table 5.7. This is not only due to the number of the operations is smaller, but also due to the fact that each hypothesis has to drive fewer operators.

## 5.2.2 GPU Implementation

We implement this idea using CUDA (Compute Unified Device Architecture), a library developed by NVIDIA Corp. for programming the GPU in a C-like language [109], [110]. CUDA makes the GPU programming almost identical to the standard CPU programming and makes the data transferring between the main memory and the graphics memory much easier than before. As a result, it has been widely applied to scientific and financial computing, signal processing, machine learning, video processing, etc<sup>6</sup>.

**Preliminary on CUDA:** A GPU is consisted of many multiprocessors, and each of them is consisted of many PE's, a shared memory, and a shared texture cache. For instance, the NVIDIA 8800GTS which we used in the experiment has 12 multiprocessors and each has 16 KB shared memory and 8 PE's. Each PE in a multiprocessor executes the same instruction in every cycle on different data. The communication between the multiprocessors is through the global device memory, but the operation of the multiprocessors are not synchronized.

In the CUDA programming environment, the programmer specifies the degree of parallelism by controlling the number of *threads*. The programmer also specifies a collection of threads that require synchronization or sharing memory as a *block*. The collection of all blocks in a single execution is called a *grid*. All threads in the same grid have identical instruction, and use the thread ID and block ID to perform data addressing.

A collection of threads can be executed on the GPU in parallel. The threads in a block are executed by a single multiprocessor at a given time. A warp is a collection of threads that are scheduled for execution simultaneously. The warp size is fixed for a specific GPU. However, the number of the threads can be much larger than the total

---

<sup>6</sup>Many examples can be found at <http://www.nvidia.com/>.



available resource. The CUDA library and GPU automatically perform the scheduling to hide the latency due to memory (device memory) access and task switching. Since the memory access is usually the performance bottleneck, one should maximize the thread number as long as the memory access does not grow exponentially.

**Our Implementation :** We implement the BPM algorithm using CUDA. In each iteration, the BPM algorithm propagates the messages from left to right, from bottom to top, from top to bottom, and from right to left. The updated messages are immediately used by the neighboring node in constructing the outgoing messages. Let assume the image size is  $N \times N$ . Because when sending the messages in one direction, all rows (or columns) are independent of each other. Therefore, the job can be partition into  $N$  blocks.

However, if we use the method in [104] to construct the message, each block can only has one thread due to the sequential nature of the algorithm. On the contrary, our algorithm can be mapped  $L + 1$  threads. The first  $L$  threads first generate  $H[i]$ . After a synchronization, The first  $L$  threads perform the local minimum cost comparison phase in Table 5.6, and the last thread performs the global minimum threshold phase. After a synchronization of the two phases, the first  $L$  threads perform the global truncation phase <sup>7</sup>.

**Results:** We use the disparity estimation problem as the benchmark. We use NVIDIA 8800GTS on a computer with Intel 3.0GHz CPU and 2GB memory to perform the experiments. The execution times are averaged over 10 runs.

We implement four different message construction methods and use the BPM [102] as the message passing scheme. We use the linear truncated function as the smoothness since all other robust functions would be extremely slow on CPU. In the first implementation, we perform the min-convolution method [104] on CPU, which is the most efficient method on CPU. In the second implementation, we perform the min-convolution on GPU. Each row/column is executed by a thread. In the third one, we perform the proposed message construction method on GPU. Each entity of the message is constructed by a thread, and the global minimum threshold phase is performed by another

---

<sup>7</sup>In the experiment we find that the synchronization steps are not necessary because all operations are very simple.

<b>Tsukuba</b> 384×288, $L=16$	BPM+ distance transform CPU	BPM+ distance transform GPU	BPM+ proposed message construction GPU	BPM+ proposed message construction + scan GPU
Calculating data term	17.72	0.03		
<i>Speedup factor</i>	1.00	663.70		
Memory transfer	0.00	27.02		
One BPM iteration	173.94	102.31	21.75	19.46
<i>Speedup factor</i>	1.00	1.70	8.00	<b>8.94</b>
Generate depth map	10.52	0.03		
<i>Speedup factor</i>	1.00	361.34		
Total (10 BP iterations)	1767.63	1050.20	244.56	221.63
<i>Speedup factor</i>	1.00	1.68	7.23	<b>7.98</b>

**Table 5.8:** The execution time of the test dataset *Tsukuba*.

<b>Cones</b> 450×375, $L=60$	BPM+ distance transform CPU	BPM+ distance transform GPU	BPM+ proposed message construction GPU	BPM+ proposed message construction + scan GPU
Calculating data term	17.72	0.03		
<i>Speedup factor</i>	1.00	583.79		
Memory transfer	0.00	29.70		
One BPM iteration	771.43	528.74	113.92	112.94
<i>Speedup factor</i>	1.00	1.46	6.77	<b>6.83</b>
Generate depth map	44.32	0.03		
<i>Speedup factor</i>	1.00	1523.14		
Total (10 BP iterations)	7776.39	5317.16	1168.99	1159.11
<i>Speedup factor</i>	1.00	1.46	6.65	<b>6.71</b>

**Table 5.9:** The execution time of the test dataset *Cones*.

single thread. Each row/column is processed in parallel.

In the final implementation, instead of using a single thread to perform the global minimum threshold phase, we use the original  $L$  threads to do it after the local minimum cost comparison phase is done. The global minimum threshold can be found in  $O(\log_2(L))$  cycles instead of  $O(L)$  by using the scan-alike operator [111]<sup>8</sup>. We make this implementation to understand the bottleneck of the proposed method.

The averaged execution time of different BP implementations are for two Middlebury datasets listed in Tables 5.8 and 5.9, respectively. We perform 10 BPM iterations in all implementations<sup>9</sup>. The execution times of different steps are shown separately, and the unit is ms. For the GPU implementations, we need to transfer the data from the main memory to the video memory, which requires additional time.

For the dataset Tsukuba, the disparity range is 16. If we directly implement the min-convolution method on GPU, the speedup factor is rather limited, only 1.68x. It is because the number of the threads does not fill the computation power of the GPU. Many cycles are wasted in waiting the data transferring between the video memory and the local memory.

Our proposed method can better utilize the GPU resource, and thus the speedup factor is much larger (7.23x). When the scan-alike operator is applied, the speedup factor can be further increased (7.98x). This suggests that the bottleneck of the proposed method is the global minimum threshold phase. Therefore, the best method is to assign  $2L$  threads for each node. The first  $L$  threads perform the local minimum cost comparison phase, and another  $L$  threads perform the global minimum threshold phase. While this is what we actually do in the VLSI implementation, this job partition is difficult in the CUDA programming model.

For the dataset Cones, the disparity range is 60, and therefore the memory consumption and the bandwidth requirement is larger than the previous dataset. The task-switching would be also more frequent. The direct implementation of the min-convolution

---

<sup>8</sup>In the terminology of the VLSI design, we use a comparator tree to find the global minimum threshold.

<sup>9</sup>For 2 iterations, the results are already much better than those of most stereo algorithms for the real-time applications.

on GPU gives less performance gain than it did in the Tsukuba dataset because more time wasted in data transferring. Our method is also slightly affected, but still much faster than the min-convolution. The overall speedup factor is 6.71 when the scan-alike operator is applied.

Note that our proof-of-concept programs are not fully optimized. The performance can be further improved if the memory conflict problem is addressed, the data is stored at the cache-enabled constant or texture memory, and if the whole program is perfectly mapped to the shaders. However, the ratio between the original and the proposed method should be only slightly affected.

### 5.2.3 Discussion

To summarize, we propose a fast message construction algorithm to remove the computation bottleneck in parallelizing the belief propagation. By exploiting the property of the robust function, a great amount of redundant operations are removed.

The proposed message construction algorithm can be used in any BP algorithm. If we combine it with the proposed tile-based BP algorithm (Section 5.1), we would have a low-complexity, low-memory, and low-bandwidth belief propagation implementation. This makes it possible to perform high-performance global optimization algorithm to high-resolution images on a memory limited parallel platform.

## 5.3 Noise-Aware Demultiplexing

In Section 3.2.2 we apply multiplexing to improve the quality of the acquired light field. We show because the noise of the imaging sensors is not i.i.d, the multiplexing patterns must be optimized according to the noise characteristics.

However, while the noise is considered in designing the multiplexing patterns, the demultiplexing does not consider the effect of the noise. Because the intensity of the acquired samples are different, some samples are noisier than others and less reliable. In this section, we exploit this property and present a new noise-aware demultiplexing algorithm.

### 5.3.1 Formulation

We formulate the demultiplexing process as an estimation of the MAP solution of the probability function. Let's first review the general multiplexing equation:

$$\mathbf{y} = \mathbf{W}\mathbf{x} + \mathbf{n}, \quad (5.9)$$

where  $\mathbf{y} = [y_1, y_2, \dots, y_N]^T$  is the observation vector,  $\mathbf{W}$  is the multiplexing matrix where the  $i$ -th row  $\mathbf{w}_i$  is the multiplexing pattern for the  $i$ -th observation,  $\mathbf{x} = [x_1, x_2, \dots, x_N]^T$  is the vector of the true signals, and  $\mathbf{n}$  is the noise vector. The demultiplexing process can be casted as a MAP estimation of the random variable:

$$\mathbf{x}^* = \arg \max_{\mathbf{x}} P(\mathbf{x} | \mathbf{y}; \mathbf{W}), \quad (5.10)$$

according to the Bayesian rule and assume the true signal is uniformly distributed, we have

$$P(\mathbf{x} | \mathbf{y}; \mathbf{W}) = \frac{P(\mathbf{y} | \mathbf{x}; \mathbf{W})}{P(\mathbf{y})} \propto P(\mathbf{y} | \mathbf{x}; \mathbf{W}). \quad (5.11)$$

Because each observation is independently sampled,

$$P(\mathbf{y} | \mathbf{x}; \mathbf{W}) = \prod_i^N P(y_i | \mathbf{x}; \mathbf{W}). \quad (5.12)$$

The goal of demultiplexing is to recover the most probable signal  $\mathbf{x}^*$  from the observation, that is,

$$\mathbf{x}^* = \arg \max_{\mathbf{x}} \prod_i^N P(y_i | \mathbf{x}; \mathbf{W}). \quad (5.13)$$

In the following we focus on the visual signal such as the light field [28] or the reflectance field [65] captured by imaging sensors. Therefore, we use the noise model described in section 3.2.2 and [67], [68]. For a true signal with magnitude  $P$ , the noise of the observed signal is normally distributed with zero mean and variance  $\sigma^2$  as

$$\sigma^2 = \sigma_c^2 + \sigma_p^2 = \sigma_c^2 + P\sigma_0^2. \quad (5.14)$$

Put this into Equation 5.9, we have

$$P(y_i | \mathbf{x}; \mathbf{W}) = (2\pi(\sigma_c^2 + (\mathbf{W}\mathbf{x})_i\sigma_0^2))^{-0.5} \exp\left(-\frac{1}{2} \frac{(y_i - (\mathbf{W}\mathbf{x})_i)^2}{\sigma_c^2 + (\mathbf{W}\mathbf{x})_i\sigma_0^2}\right). \quad (5.15)$$

If we put Equation 5.15 into Equation 5.13 and take the negative log, we have

$$\begin{aligned}
\mathbf{x}^* &= \arg \min_{\mathbf{x}} -\log \prod_i^N P(y_i | \mathbf{x}; \mathbf{W}) \\
&= \arg \min_{\mathbf{x}} -\sum_i^N \log P(y_i | \mathbf{x}; \mathbf{W}) \\
&= \arg \min_{\mathbf{x}} \sum_i^N \left( \frac{(y_i - (\mathbf{W}\mathbf{x})_i)^2}{\sigma_c^2 + (\mathbf{W}\mathbf{x})_i \sigma_0^2} + \log \left( 2\pi\sigma_c^2 \left( 1 + \frac{(\mathbf{W}\mathbf{x})_i \sigma_0^2}{2\pi\sigma_c^2} \right) \right) \right) \\
&\approx \arg \min_{\mathbf{x}} \sum_i^N \left( \frac{(y_i - (\mathbf{W}\mathbf{x})_i)^2}{\sigma_c^2 + (\mathbf{W}\mathbf{x})_i \sigma_0^2} + \left( \frac{\sigma_0^2 (\mathbf{W}\mathbf{x})_i}{2\pi\sigma_c^2} \right) \right), \tag{5.16}
\end{aligned}$$

where the last equality comes from the first-order approximation of the logarithm function. The solution that minimizes this objective function is the MAP solution. We can see that the observations are not equally important in this formulation. When the multiplexed signal  $(\mathbf{W}\mathbf{x})_i$  is larger, the variance of the noise is larger. As a result, the estimation error  $(y_i - (\mathbf{W}\mathbf{x})_i)^2$  is less reliable and is weighted less than others. Also a L1 regularization term is present in the objective function. Therefore, the problem becomes a *weighted L1 regularized optimization* problem. It should be noted that when  $\sigma_0^2 = 0$ , the objective function becomes the typical L2 norm equation and the optimal solution would be  $\mathbf{W}^{-1}\mathbf{y}$ , same as the result in the traditional demultiplexing.

### 5.3.2 Optimization

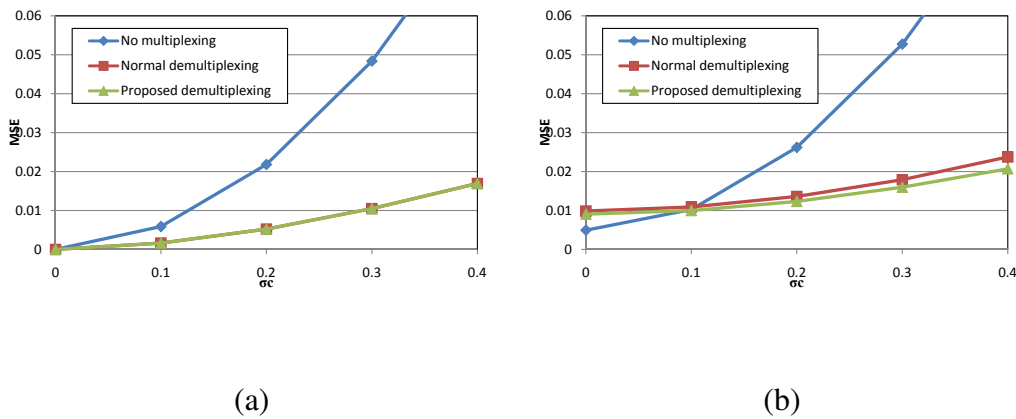
In Equation 5.16, the estimation error is weighted by the unknown signal and thus cannot be directly evaluated. However, we can assume  $\sigma_c^2 + (\mathbf{W}\mathbf{x})_i \sigma_0^2 \approx \sigma_c^2 + y_i \sigma_0^2$  and have

$$\mathbf{x}^* = \arg \min_{\mathbf{x}} \sum_i^N \left( \frac{(y_i - (\mathbf{W}\mathbf{x})_i)^2}{\sigma_c^2 + y_i \sigma_0^2} + \frac{\sigma_0^2 (\mathbf{W}\mathbf{x})_i}{2\pi\sigma_c^2} \right). \tag{5.17}$$

This is a standard L1 regularized least-square problem. Note that this approximation is unbiased, and we can substitute the obtained  $\mathbf{x}^*$  back to Equation 5.16 and iteratively perform the optimization.

### 5.3.3 Results

We use the interior-point method in [112] to perform the optimization. It uses preconditioned conjugate gradient method to compute the search step and thus converges much



**Figure 5.12:** *The MSE of different multiplexing schemes.*

We perform the simulation with different noise characteristics. (a)  $\sigma_0 = 0$ , and (b)  $\sigma_0 = 0.4$ .

faster than other methods.

We perform a synthetic experiment here. The true signal is 25-D and the dynamic range of each sample is  $(0, 1)$ . We use the optimal multiplexing patterns described in Section 3.2.2 to obtain the multiplexed signal. The mean-square-error (MSE) of the recovered signals are shown in Figure 5.12. We can see that when there is no signal-dependent noise, the results of the normal demultiplexing and the proposed demultiplexing methods are identical and better than the signal sampled without multiplexing. When there is only shot noise ( $\sigma_0 > 0$  and  $\sigma_c = 0$ ), the signal without multiplexing is the best, as described in [66]. However, for the general cases ( $\sigma_0 > 0$  and  $\sigma_c > 0$ ), the signals recovered by the proposed algorithm are consistently better than that recovered by the normal demultiplexing method. The average MSE reduction is 11.0%.

### 5.3.4 Discussion

In this section we have propose a new noise-aware demultiplexing method. We have formulated the demultiplexing as a MAP estimation and showed that it is equivalent to a L1-regularized least-square problem. The experimental results show that the proposed method can recover a signal with lower errors.

While the simulation results look promising, we believe this method can be extended

in many ways. First, we assume the signal is uniform-distributed so our algorithm is not context-aware. For example, in programmable aperture photography, the angular information of each spatial location is independently multiplexed, but it is possible to jointly demultiplex the spatially neighboring samples to further improve the data quality. Second, if the statistics of the signal is known, it is possible to optimize the multiplexing patterns using the same Bayesian framework.

## 5.4 Summary

In this chapter we have developed the algorithms that can not only benefit the light field acquisition but also other computer vision applications. We have presented a modified belief propagation scheme and a fast message construction algorithm. These contributions make it possible to perform global optimization on large images in a highly parallel fashion.

We also have presented a noise-aware demultiplexing algorithm. By formulating the problem in a probabilistic framework, we have shown that the optimal demultiplexing is equivalent to minimizing a L1-regularized least-square problem. This algorithm can be used to many different applications, including illumination multiplexing and capturing the light field and the reflectance field.



## Chapter 6

### Conclusion

In this dissertation, we have applied the light transport theory to analyze the effect of the camera to the light field in a signal-processing framework. The analysis shows that the light ray traveling, the lens refraction, and the radiance integration of the sensor, can be described by a single photographic operator and a slicing operator.

Our model shows that there is no filtering applied to the light rays in the photography process. The light field inside the camera is a transformed and modulated light field of the outside world. This model can explain many photographic effects in the image formation process, which were described by many independent models. The perspective transform of the scene results from the shearing of the light field due to light ray traveling and the lens refraction. The defocus blur on the image plane is a combination of the convolution in the frequency domain and the slicing operator. These effects can also be naturally expressed in the frequency domain. The results obtained by our model are quantitatively identical to the traditional methods.

We have applied the derived model to other applications. First, we show that the operation of the digital refocusing is equivalent to a linear transform operator to the light field. Second, we build a novel method to fuse several different focused images into an all-focused one. This method requires no per-pixel depth map estimation. It can be performed efficiently in the light field frequency domain. Third, we use the spectrum analysis to suggest the parameter settings of the light field cameras. Finally, we formulate a novel depth-detection problem and propose a new algorithm to solve it.

Without searching for the correspondences between the images of different viewpoints, we directly estimate the possible depths from the spectrum of the light field. While the focusness measurements in the spectrum are not fully reliable, we re-formulate the problem as a max-cover problem and efficiently solve it using dynamic programming.

We have proposed a novel device, dubbed programmable aperture, to capture the light field. By adjusting the transmittance of the aperture dynamically, we can sequentially capture the light field without moving the camera. To improve the acquisition efficiency, we optically multiplex the light field and then recover the light field signal by demultiplexing. The optimal multiplexing scheme is found from an optimization process. This method can capture high quality light field with the spatial resolution identical to the sensor resolution, which is orders of magnitude higher than the other light field cameras. The angular resolution, the sampling grid, and the pre-filter kernel are all fully adjustable. Also it requires no expensive micron scale manufacture and tedious calibration.

The 4D light field data captured by our programmable aperture and other light field cameras have unique distortions to be removed. We have therefore presented novel algorithms to address them. First, we analyzed the photometric distortion in the light field and showed that it is fundamentally different to the one in traditional images. We then proposed a novel algorithm that uses feature matching and iterative optimization to estimate and remove the photometric distortion.

Second, we have proposed a novel multi-view depth estimation algorithm to infer the scene geometry from the captured light field. We have formulated the program as a MAP-MRF problem and used an accurate occlusion model to describe the multi-view occlusion events. The problem is efficiently solved in an iterative fashion and the cross-bilateral filtering is used to increase the convergence speed. The experimental results showed that the proposed algorithm is comparable to the state-of-the-art algorithms.

We have used the estimated depth maps to assist the image-based rendering. More specifically, during the view interpolation, each light field sample is warped according to its depth value. In this way the ghosting effect is removed. We combined this technique with the proposed light transport analysis to synthetically increase the angular

sampling rate to reach the Nyquist rate and thus remove the aliasing in the refocused images.

Manually specifying the refocusing parameters is a tedious task. To address this issue, we have proposed a novel feature-based refocusing interface. By using robust feature matching and homography estimation, we allowed users to intuitively change the focus setting and obtain the refocused image in real-time.

Efficient processing of the high resolution and high dimensional multi-view or light field data is a challenge and important issue. The memory, bandwidth, and computation requirements of most algorithms grow exponentially to the input data. We have proposed a tile-based belief propagation algorithm and a parallel message construction algorithm to solve this problem. The proposed methods successfully remove most redundant computations and data in the message passing operation while the performance is unaffected. It can be easily mapped on the parallel hardware architectures such as VLSI circuit and GPU. Finally, we have proposed a new noise-aware demultiplexing algorithm. By formulating the noise-dependent multiplexing process in a probabilistic framework, we can find the MAP solution by solving a L1-regularized least-square problem.

In summary, this dissertation covers several topics about the efficient analysis, acquisition, and processing of the high dimensional, high resolution visual data. The light transport analysis re-formulates the whole image formation process, and the proposed devices and algorithms can efficiently generate excellent results. We believe many proposed techniques can be readily applied to the industry, such as the programmable aperture, the feature-based refocusing, and the modified belief propagation algorithm. We also hope that the theoretical part of the dissertation can inspire new researches in the computer vision, computer graphics, and image processing.



# Bibliography

- [1] Edward H. Adelson and James R. Bergen, “The plenoptic function and the elements of early vision,” *Computational Models of Visual Processing*, pp. 3–20, 1991.
- [2] Parry Moon and Domina Eberle Spencer, *The photic field*, MIT Press, 1981.
- [3] Marc Levoy and Pat Hanrahan, “Light field rendering,” in *SIGGRAPH ’96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, New York, NY, USA, 1996, pp. 31–42, ACM Press.
- [4] Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen, “The lumigraph,” in *SIGGRAPH ’96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, New York, NY, USA, 1996, pp. 43–54, ACM Press.
- [5] Berthold K.P. Horn, *Robot Vision*, MIT Press, 1986.
- [6] Shree K. Nayar, Masahiro Watanabe, and Minori Noguchi, “Real-time focus range sensor,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 12, pp. 1186–1198, Dec 1996.
- [7] Dan B Goldman and Jiun-Hung Chen, “Vignette and exposure calibration and compensation,” in *ICCV ’05: Proc. the 10th IEEE International Conference on Computer Vision*, 2005, pp. 899–906.
- [8] James T. Kajiya, “The rendering equation,” *SIGGRAPH Comput. Graph.*, vol. 20, no. 4, pp. 143–150, 1986.

- [9] Alan V. Oppenheim and Alan S. Willsky, *Signals and Systems, 2nd Edition*, Prentice-Hall, 1997.
- [10] Paul S. Heckbert, *Fundamentals of Texture Mapping and Image Warping*, Ph.D. thesis, University of California at Berkeley, 1989.
- [11] Jin-Xiang Chai, Shing-Chow Chan, and Heung-Yeung Shuang and Xin Tong, “Plenoptic sampling,” in *SIGGRAPH ’00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, New York, NY, USA, 2000, pp. 307–318, ACM Press/Addison-Wesley Publishing Co.
- [12] Aaron Isaksen, Leonard McMillan, and Steven J. Gortler, “Dynamically reparameterized light fields,” in *SIGGRAPH ’00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, New York, NY, USA, 2000, pp. 297–306, ACM Press/Addison-Wesley Publishing Co.
- [13] Matthias Zwicker, Wojciech Matusik, Frédo Durand, and Hanspeter Pfister, “Antialiasing for automultiscopic 3d displays,” in *EGSR ’06: Proc. the 17th Eurographics Symposium on Rendering*, 2006.
- [14] Ravi Ramamoorthi and Pat Hanrahan, “A signal-processing framework for inverse rendering,” in *SIGGRAPH ’01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, New York, NY, USA, 2001, pp. 117–128, ACM.
- [15] Ravi Ramamoorthi and Pat Hanrahan, “A signal-processing framework for reflection,” *ACM Transactions on Graphics*, vol. 23, no. 4, pp. 1004–1042, 2004.
- [16] Ravi Ramamoorthi and Pat Hanrahan, “Frequency space environment map rendering,” *ACM Trans. Graph.*, vol. 21, no. 3, pp. 517–526, 2002.
- [17] Jaakko Lehtinen, “A framework for precomputed and captured light transport,” *ACM Transactions on Graphics*, vol. 26, no. 4, pp. 13, 2007.

- [18] Ravi Ramamoorthi, Melissa Koudelka, and Peter Belhumeur, “A fourier theory for cast shadows,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 2, pp. 1–8, 2005.
- [19] Frédo Durand, Nicolas Holzschuch, Cyril Soler, Eric Chan, and François X. Sillion, “A frequency analysis of light transport,” in *SIGGRAPH ’05: ACM SIGGRAPH 2005 Papers*, New York, NY, USA, 2005, pp. 1115–1126, ACM Press.
- [20] Ravi Ramamoorthi, Dhruv Mahajan, and Peter Belhumeur, “A first-order analysis of lighting, shading, and shadows,” *ACM Transactions on Graphics*, vol. 26, no. 1, pp. 2, 2007.
- [21] Ashok Veeraraghavan, Ramesh Raskar, Amit Agrawal, Ankit Mohan, and Jack Tumblin, “Dappled photography: mask enhanced cameras for heterodyned light fields and coded aperture refocusing,” *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 26, no. 3, pp. 69, 2007.
- [22] Todor Georgiev, Chintan Intwala, and Derin Babacan, “Light-field capture by multiplexing in the frequency domain,” Adobe technical report, Adobe Systems Incorporated, 2007.
- [23] A. Gerrard and Jennings Michael Burch, *Introduction to Matrix Methods in Optics*, Courier Dover Publications, 1994.
- [24] Ren Ng, “Fourier slice photography,” in *SIGGRAPH ’05: ACM SIGGRAPH 2005 Papers*, New York, NY, USA, 2005, pp. 735–744, ACM.
- [25] Yoav Y. Schechner and Nahum Kiryati, “Depth from defocus vs. stereo: How different really are they?,” *International Journal of Computer Vision*, vol. 39, no. 2, pp. 141–162, 2000.
- [26] Naoki Asada, Hisanaga Fujiwara, and Takashi Matsuyama, “Seeing behind the scene: Analysis of photometric properties of occluding edges by the reversed projection blurring model,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 2, pp. 155–167, 1998.

- [27] Ren Ng, Marc Levoy, Mathieu Brédif, Gene Duval, Mark Horowitz, and Pat Hanrahan, “Light field photography with a hand-held plenoptic camera,” CSTR 2005-02, Stanford University, April 2005.
- [28] Chia-Kai Liang, Tai-Hsu Lin, Bing-Yi Wong, Chi Liu, and Homer Chen, “Programmable aperture photography: Multiplexed light field acquisition,” *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 27, no. 3, pp. 55:1–55:10, 2008.
- [29] Akira Kubota, Kiyoharu Aizawa, and Tsuhan Chen, “Direct filtering method for image based rendering,” in *Proc. International Conference on Image Processing*, 2005, vol. 1, pp. 1–4.
- [30] Andrew Lumsdaine and Todor Georgiev, “Full resolution lightfield rendering,” Adobe technical report, Adobe Systems Incorporated, 2008.
- [31] Yi-Hao Kao, Chia-Kai Liang, Li-Wen Chang, and Homer Chen, “Depth detection of light field,” in *Proc. International Conference on Acoustics, Speech, and Signal Processing*, 2007, vol. 1, pp. 893–897.
- [32] Andrew Adams and Marc Levoy, “General linear cameras with finite aperture,” in *EGSR '07: Proc. the 18th Eurographics workshop on Rendering*, 2007.
- [33] Anat Levin, Rob Fergus, Frédo Durand, and William T. Freeman, “Image and depth from a conventional camera with a coded aperture,” *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 26, no. 3, pp. 70, 2007.
- [34] David L. Donoho, “Compressed sensing,” *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.
- [35] Chia-Kai Liang, Gene Liu (a.k.a Chi Liu), and Homer Chen, “Light field acquisition using programmable aperture camera,” in *Proc. International Conference on Image Processing*, 2007, vol. 5, pp. 233–236.
- [36] Takanori Okoshi, *Three-dimensional imaging techniques*, Academic Press New York, 1976.



- [37] David Roberts, “History of lenticular and related autostereoscopic methods,” *White paper, Leap Technologies, LLC*, 2003.
- [38] Cha Zhang and Tsuhan Chen, “A survey on image-based renderingxrepresentation, sampling and compression,” *Signal Processing: Image Communication*, vol. 19, no. 1, pp. 1–28, 2004.
- [39] Jack Tumblin and Ramesh Raskar, “State of the art report (STAR): Computational photography,” in *Proc. ACM/Eurographics 2006*, 2006.
- [40] Shree K. Nayar, “Computational cameras: Redefining the image,” *IEEE Computer Mag., Special Issue on Computational Photography*, pp. 62–70, Aug 2006.
- [41] Reinhard Koch, Marc Pollefeys, Luc Van Gool, Benno Heigl, and Heinrich Niemann, “Calibration of hand-held camera sequences for plenoptic modeling,” in *ICCV ’99: Proc. the Seventh IEEE International Conference on Computer Vision*, 1999, vol. 1, pp. 585–591.
- [42] Jason C. Yang, Matthew Everett, Chris Buehler, and Leonard McMillan, “A real-time distributed light field camera,” in *EGRW ’02: Proc. the 13th Eurographics workshop on Rendering*, 2002, pp. 77–86.
- [43] Cha Zhang and Tsuhan Chen, “A self-reconfigurable camera array,” in *EGRW ’04: Proc. the 15th Eurographics workshop on Rendering*, 2004, vol. 4, pp. 243–254.
- [44] Bennett Wilburn, Neel Joshi, Vaibhav Vaish, Eino-Ville Talvala, Emilio Antunez, Adam Barth, Andrew Adams, Mark Horowitz, and Marc Levoy, “High performance imaging using large camera arrays,” in *SIGGRAPH ’05: ACM SIGGRAPH 2005 Papers*, New York, NY, USA, 2005, pp. 765–776, ACM.
- [45] Gabriel Lippmann, “Epreuves reversible donnant la sensation du relief,” *Journal of Physics*, vol. 7, pp. 821–825, 1908.

- [46] Herbert E. Ives, "Parallax panoramagrams made with a large diameter lens," *Journal of the Optical Society of America*, vol. 20, no. 6, pp. 332–342, June 1930.
- [47] Herbert E. Ives, "Parallax panoramagrams made with a large diameter concave mirror," *Journal of the Optical Society of America*, vol. 20, no. 11, pp. 597–600, Nov. 1930.
- [48] Edward H. Adelson and John Y. A. Wang, "Single lens stereo with a plenoptic camera," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 99–106, 1992.
- [49] Todor Georgiev, Ke Colin Zheng, Brian Curless, David Salesin, Shree Nayar, and Chintan Intwala, "Spatio-angular resolution tradeoff in integral photography," in *EGRW '06: Proc. the 17th Eurographics workshop on Rendering*, 2006.
- [50] Gerald K. Skinner, "X-ray imaging with coded masks," *Scientific American*, vol. 259, no. 2, pp. 84–89, 1988.
- [51] Stephen R. Gottesman and E. E. Fenimore, "New family of binary arrays for coded aperture imaging," *Applied Optics*, vol. 28, no. 20, pp. 4344–4352, 1989.
- [52] Ramesh Raskar, Amit Agrawal, and Jack Tumblin, "Coded exposure photography: motion deblurring using fluttered shutter," *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 25, no. 3, pp. 795–804, 2006.
- [53] Shree K. Nayar and Vlad Branzoi, "Adaptive dynamic range imaging: Optical control of pixel exposuregoldman05s over space and time," in *ICCV '03: Proc. the Ninth IEEE International Conference on Computer Vision*, Oct 2003, vol. 2, pp. 1168–1175.
- [54] Yoav Y. Schechner and Shree K. Nayar, "Uncontrolled modulation imaging," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Jun 2004, vol. 2, pp. 197–204.

- [55] Assaf Zomet and Shree K. Nayar, “Lensless imaging with a controllable aperture,” *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pp. 339–346, 2006.
- [56] Hany Farid and Eero P. Simoncelli, “Range estimation by optical differentiation,” *Journal of the Optical Society of America A*, vol. 15, no. 7, pp. 1777–1786, 1998.
- [57] Yosuke Bando, Bing-Yu Chen, and Tomoyuki Nishita, “Extracting depth and matte using a color-filtered aperture,” *ACM Transactions on Graphics*, vol. 27, no. 5, 2008.
- [58] Ramesh Raskar, Kar-Han Tan, Rogerio Feris, Jingyi Yu, and Matthew Turk, “Non-photorealistic camera: depth edge detection and stylized rendering using multi-flash imaging,” in *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*, New York, NY, USA, 2004, pp. 679–688, ACM Press.
- [59] Neel Joshi, Wojciech Matusik, and Shai Avidan, “Natural video matting using camera arrays,” *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 25, no. 3, pp. 779–786, 2006.
- [60] Elmar Eisemann and Frédo Durand, “Flash photography enhancement via intrinsic relighting,” in *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*, New York, NY, USA, 2004, pp. 673–678, ACM Press.
- [61] Georg Petschnigg, Richard Szeliski, Maneesh Agrawala, Michael Cohen, Hugues Hoppe, and Kentaro Toyama, “Digital photography with flash and no-flash image pairs,” in *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*, New York, NY, USA, 2004, pp. 664–672, ACM Press.
- [62] Lu Yuan, Jian Sun, Long Quan, and Heung-Yeung Shum, “Image deblurring with blurred/noisy image pairs,” *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 26, no. 3, pp. 1, 2007.
- [63] Cho Senkichi, Moriya Toshio, Hase Toshinori, Miyata Yuichi, and Kubota Hidetoshi, “Device and method for correcting camera-shake and device for detecting camera shake,” 2003, JP Patent No. 2003-138436.

- [64] Yoav Y. Schechner, Shree K. Nayar, and Peter N. Belhumeur, “A theory of multiplexed illumination,” in *ICCV '03: Proc. the Ninth IEEE International Conference on Computer Vision*, 2003, vol. 2, pp. 808–815.
- [65] Andreas Wenger, Andrew Gardner, Chris Tchou, Jonas Unger, Tim Hawkins, and Paul Debevec, “Performance relighting and reflectance transformation with time-multiplexed illumination,” in *ACM SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*, New York, NY, USA, 2005, pp. 756–764, ACM Press.
- [66] Martin Harwit and Neil J. A. Sloane, *Hadamard Transform Optics*, Academic Press, New York, 1979.
- [67] HP components group, “Noise sources in CMOS image sensors,” Technical report, Hewlett-Packard Company, 1998.
- [68] Yanghai Tsin, Visvanathan Ramesh, and Takeo Kanade, “Statistical calibration of CCD imaging process,” in *ICCV '01: Proc. the Eighth IEEE International Conference on Computer Vision*, 2001, pp. 480–487.
- [69] Netanel Ratner and Yoav Y. Schechner, “Illumination multiplexing within fundamental limits,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2007.
- [70] Ankit Mohan, Xiang Huang, Ramesh Raskar, and Jack Tumblin, “Sensing increased image resolution using aperture masks,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2008.
- [71] Qingxu Dou and Paolo Favaro, “Off-axis aperture camera: 3d shape reconstruction and image restoration,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2008.
- [72] Samuel W. Hasinoff and Kiriakos N. Kutulakos, “Light-efficient photography,” in *Proc. 10th European Conference on Computer Vision*, 2008, pp. 45–59.

- [73] Hajime Nagahara, Sujit Kuthirumma, Changyin Zhou, and Shree K. Nayar, “Flexible depth of field photography,” in *Proc. 10th European Conference on Computer Vision*, 2008.
- [74] Jr. Edward R. Dowski and W. Thomas Cathey, “Extended depth of field through wave-front coding,” *Applied Optics*, vol. 34, no. 11, pp. 1859–1866, 1995.
- [75] Anat Levin, William Freeman, and Frédo Durand, “Understanding camera trade-offs through a bayesian analysis of light field projections,” MIT-CSAIL-TR-2008-049, Massachusetts Institute of Technology, 2008.
- [76] Manoj Aggarwal, Hong Hua, and Narendra Ahuja, “On cosine-fourth and vignetting effects in real lenses,” in *ICCV ’01: Proc. the Eighth IEEE International Conference on Computer Vision*, 2001, vol. 1, pp. 472–479.
- [77] Anatoly Litvinov and Yoav Y. Schechner, “Addressing radiometric nonidealities: A unified framework,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2005, vol. 2, pp. 52–59.
- [78] Yuanjie Zheng, Jingyi Yu, Sing Bing Kang, Stephen Lin, and Chandra Kambhamettu, “Single-image vignetting correction using radial gradient symmetry,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2008.
- [79] David G. Lowe, “Distinctive image features from scale-invariant keypoints,” *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.
- [80] Jason Stewart, Jingyi Yu, Steven J. Gortler, and Leonard McMillan, “A new reconstruction filter for undersampled light fields,” in *EGSR ’03: Proc. the 14th Eurographics workshop on Rendering*, 2003, pp. 150–156.
- [81] Daniel Scharstein and Richard Szeliski, “A taxonomy and evaluation of dense two-frame stereo correspondence algorithms,” *International Journal of Computer Vision*, vol. 47, no. 1-3, pp. 7–42, 2002.

- [82] Vladimir Kolmogorov and Ramin Zabih, “Multi-camera scene reconstruction via graph cuts,” *Proc. European Conference on Computer Vision*, vol. 3, pp. 82–96, 2002.
- [83] Sing Bing Kang and Richard Szeliski, “Extracting view-dependent depth maps from a collection of images,” *International Journal of Computer Vision*, vol. 58, no. 2, pp. 139–163, 2004.
- [84] Jian Sun, Yin Li, Sing Bing Kang, and Heung-Yeung Shum, “Symmetric stereo matching for occlusion handling,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2005, vol. 2, pp. 399–406.
- [85] Yori Boykov, Olga Veksler, and Ramin Zabih, “Fast approximate energy minimization via graph cuts,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 11, pp. 1222–1239, 2001.
- [86] Ramin Zabih and John Woodfill, “Non-parametric local transforms for computing visual correspondence,” in *Proc. 3rd European Conference on Computer Vision*. 1994, vol. 2, pp. 151–158, Springer.
- [87] Heiko Hirschmüller and Daniel Scharstein, “Evaluation of cost functions for stereo matching,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2007.
- [88] Stan Birchfield and Carlo Tomasi, “A pixel dissimilarity measure that is insensitive to image sampling,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 4, pp. 401–406, 1998.
- [89] Richard Szeliski, Ramin Zabih, Daniel Scharstein, Olga Veksler, Vladimir Kolmogorov, Aseem Agarwala, Marshall Tappen, and Carsten Rother, “A comparative study of energy minimization methods for markov random fields,” in *Proc. European Conference on Computer Vision*, 2006, vol. 2, pp. 16–29.
- [90] William T. Freeman, Egon C. Pasztor, and Owen T. Carmichael, “Learning low-level vision,” *International Journal of Computer Vision*, vol. 40, no. 1, pp. 25–47, 2000.

- [91] Vladimir Kolmogorov, “Convergent tree-reweighted message passing for energy minimization,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 10, pp. 1568–1583, 2006.
- [92] Qingxiong Yang, Ruigang Yang, James Davis, and David Nister, “Spatial-depth super resolution for range images,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2007.
- [93] Paul Green, Wenyang Sun, Wojciech Matusik, and Frédo Durand, “Multi-aperture photography,” *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 26, no. 3, pp. 68, 2007.
- [94] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik, “Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach,” in *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, New York, NY, USA, 1996, pp. 11–20, ACM.
- [95] Nikos Komodakis, Nikos Paragios, and Georgios Tziritas, “MRF optimization via dual decomposition: Message-passing revisited,” in *ICCV 2007: Proc. the 11th International Conference on Computer Vision*, 2007.
- [96] Yair Weiss, “Correctness of local probability propagation in graphical models with loops,” *Neural Computation*, vol. 12, no. 1, pp. 1–41, 2000.
- [97] Yair Weiss and William T. Freeman, “On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs,” *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 736–744, 2001.
- [98] Jonathan Yedidia, William T. Freeman, and Yair Weiss, “Constructing free-energy approximations and generalized belief propagation algorithms,” *IEEE Transactions on Information Theory*, vol. 51, no. 7, pp. 2282–2312, 2005.
- [99] Andrew DeLong and Yuri Boykov, “A scalable graph-cut algorithm for N-D grids,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

- [100] Tianli Yu, Ruei-Sung Lin, Boaz Super, and Bei Tang, "Efficient message representations for belief propagation," in *ICCV '07: Proc. the 11-th IEEE International Conference on Computer Vision*, 2007.
- [101] Yu-Cheng Tseng, Nelson Yen-Chung Chang, and Tian-Sheuan Chang, "Low memory cost block-based belief propagation for stereo correspondence," in *Proc. IEEE International Conference on Multimedia and Expo*, 2007, pp. 1415–1418.
- [102] Marshall Tappen and William T. Freeman, "Comparison of graph cuts with belief propagation for stereo, using identical MRF parameters," in *ICCV '03: Proc. the Ninth IEEE International Conference on Computer Vision*, 2003, pp. 900–907.
- [103] Tung-Chien Chen, Shao-Yi Chien, Yu-Wen Huang, Chen-Han Tsai, Ching-Yeh Chen, To-Wei Chen, and Liang-Gee Chen, "Analysis and architecture design of an HDTV720p 30 frames/s H.264/AVC encoder," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 6, pp. 673–688, 2006.
- [104] Pedro F. Felzenszwalb and Daniel P. Huttenlocher, "Efficient belief propagation for early vision," *International Journal of Computer Vision*, vol. 70, no. 1, pp. 41–54, 2006.
- [105] Jen-Chieh Tuan, Tian-Sheuan Chang, and Chein-Wei Jen, "On the data reuse and memory bandwidth analysis for full-searchblock-matching VLSI architecture," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 1, pp. 61–72, 2002.
- [106] Michael J. Black and Anand Rangarajan, "On the unification of line processes, outlier rejection, and robust statistics with applications in early vision," *International Journal of Computer Vision*, vol. 19, no. 1, pp. 57–91, 1996.
- [107] Guillermo Sapiro Michael J. Black, David Marimont, and David Heeger, "Robust anisotropic diffusion," *IEEE Transactions on Image Processing*, vol. 7, no. 3, pp. 421–432, 1998.
- [108] Frédo Durand and Julie Dorsey, "Fast bilateral filtering for the display of high-dynamic-range images," in *SIGGRAPH '02: Proceedings of the 29th annual*



*conference on Computer graphics and interactive techniques*, New York, NY, USA, 2002, pp. 257–266, ACM.

- [109] Nvidia Corporation, “NVIDIA CUDA compute unified device architecture programming guide,” Documentation, Nvidia Corporation, 2007.
- [110] John D. Owens, Mike Houston, David Luebke, Simon Green, John E. Stone, and James C. Phillips, “GPU computing,” *Proceedings of the IEEE*, vol. 96, no. 5, pp. 879–899, 2008.
- [111] Mark Harris, “Parallel prefix sum (scan) with CUDA,” Documentation, Nvidia Corporation, 2007.
- [112] Seung-Jean Kim, Kwangmoo Koh, Michael Lustig, and Stephen Boyd, “A method for large-scale  $\ell_1$ -regularized least squares,” *IEEE Journal on Selected Topics in Signal Processing*, vol. 1, no. 4, pp. 606–617, 2007.



# Publication and Honors

This section lists my publications and major honors during the master and PhD programs before the dissertation is finished (November 2008).

## Journal Papers

- [1] P.-S. Tsai, **C.-K. Liang**, T.-H. Huang, and H. H. Chen, “Image enhancement for backlight-scaled TFT-LCD displays,” *IEEE Trans CSVT*, accepted.
- [2] **C.-K. Liang**, T.-H. Lin, B.-Y. Wong, C. Liu, and H. H. Chen, “Programmable aperture photography: multiplexed light field acquisition,” *ACM Trans Graphics (Proc. SIGGRAPH 2008)*, vol. 27, no. 3, pp. 55:1-55:10.
- [3] **C.-K. Liang**, L. Chang, and H. H. Homer, “Analysis and compensation of rolling shutter effect,” *IEEE Trans Image Processing*, vol. 17, no. 8, pp. 1323-1330, Aug. 2008.
- [4] H. H. Chen, **C.-K. Liang**, Y.-C. Peng, and H.-A. Chang, “Integration of digital stabilizer with video codec for digital video cameras,” *IEEE Trans. CSVT*, vol. 17, no. 7, pp. 801-813, Jul. 2007. *2008 IEEE Circuits and Systems Society CSVT Best Paper Award.*
- [5] Y.-E. Wu, **C.-K. Liang**, C.-T. Tsao, K.-S. Hsu, Y.-J. Lin, “Digital video content service-license usage management platform,” *CCL Technical Journal*, no.110, pp. 44-52.

## Conference Papers

- [6] C.-C. Cheng, **C.-K. Liang**, Y.-C. Lai, H. H. Chen, and L.-G. Chen, "Analysis of belief propagation for hardware realization," *Proc. SIPS 2008*, to appear.
- [7] T.-H. Huang, **C.-K. Liang**, S.-L. Yeh, and H. H. Chen, "JND-based enhancement of perceptibility for dim images," *Proc. ICIP'08*, to appear.
- [8] Y.-H. Chen, **C.-K. Liang**, and Y.-Y. Chuang, "Depth-of-field matting," in *Proc. CVGIP'08*.
- [9] P.-S. Tsai, **C.-K. Liang**, and H. H. Chen, "Image quality enhancement for low backlight TFT-LCD displays," in *Proc. ICIP'07*, San Antonio, Texas, Sept. 16-19, 2007, vol. 3, pp.473-476.
- [10] **C.-K. Liang**, G. Liu, and H. H. Chen, "Light field acquisition using programmable aperture camera," in *Proc. ICIP'07*, San Antonio, Texas, Sept. 16-19, 2007, vol. 5, pp. 233-236.
- [11] Y.-H. Kao, **C.-K. Liang**, and H. H. Chen, "Depth detection of light field," in *Proc. ICASSP'07*, Honolulu, Hawaii, USA, Apr. 2007, vol. 1, pp. 893-896.
- [12] C.-C. Liu, **C.-K. Liang**, K.-S. Hsu, C.-P. Kuan, and H. H. Chen, "Digital rights management for DVB-H Mobile TV," in *Europe-China Conference on Intellectual Property in Digital Media*, Shanghai, China, Oct. 2006.
- [13] **C.-K. Liang**, C.-C. Liu and H. H. Chen, "A robust DRM system on the DVB Multimedia Home Platform," in *Proc. IEEE CCNC'06*, vol. 1, Las Vegas, Nevada, USA, Jan. 2006. pp. 605-609.
- [14] L.-W. Chang, **C.-K. Liang**, and H. H. Chen, "Analysis and correction of rolling shutter distortion for CMOS image sensor arrays," in *Proc. IS-COM'05*, Kaohsiung, Taiwan, Nov. 2005.
- [15] **C.-K. Liang**, Y.-C. Peng and H. H. Chen, "Rolling shutter distortion correction," in *SPIE Proc. VCIP'05*, vol. 5960, Beijing, China, July, 2005. pp. 1315-1322.
- [16] Y.-C. Peng, **C.-K. Liang**, H.-A. Chang, C.-J. Kao and H. H. Chen, "Integration of image stabilizer and video encoder for digital video cameras," in *Proc. ISCAS'05*, vol. 5, Kobe, Japan, May 2005, pp. 4871-4874.
- [17] **C.-K. Liang**, Y.-C. Peng, H.-A. Chang, C.-C. Su and H. H. Chen, "The effect of digital image stabilization on coding performance," in *Proc. ISIMP'04*, Hong Kong, China, Oct. 2004, pp. 402-405.

## Patents

- [18] M.-Y. Wu, **C.-K. Liang**, C.-C. You, and B. Sung, “Method for enhanced removing noise of digital image and apparatus thereof,” Taiwan, USA, and Japan pending.
- [19] M.-Y. Wu, **C.-K. Liang**, and B. Sung, “Face detection system and method,” Taiwan and USA pending.
- [20] H. H. Chen, **C.-K. Liang**, D. Yeh, and B. Sung, “Digital image stabilization method,” Taiwan and USA pending.
- [21] M.-Y. Wu, B. Sung, C.-C. Yu, and C.-K. Liang, “Method and apparatus thereof for enhancing digital image,” Taiwan and USA pending.
- [22] **C.-K. Liang**, H. H. Chen, D. Yeh, and H. Wang, “Camera using programmable aperture,” US 2008/0124070 A1. Taiwan pending.
- [23] H. H. Chen, **C.-K. Liang** and C.-C. Liu, “Management system for digital broadcast rights and a method thereof,” US 2007/0189530 A1, Taiwan I-281825.

## In Preparation

- [24] C.-C. Cheng, **C.-K. Liang**, Y.-C. Lai, H. H. Chen, and L.-G. Chen, “Fast belief propagation process element for high-quality stereo estimation,” 4-page conference paper.
- [25] C.-C. Cheng, **C.-K. Liang**, Y.-C. Lai, H. H. Chen, and L.-G. Chen, “Efficient memory architecture for tile-based belief propagation,” 4-page conference paper.
- [26] **C.-K. Liang**, C.-C. Cheng, Y.-C. Lai, H. H. Chen, and L.-G. Chen, “Parallel and low-memory belief propagation for early vision,” 8-page conference or journal paper.

## Major Honors

- [1] Travel Grant for SIGGRAPH 2008 from the foundation from the advancement of outstanding scholarship (US \$1930).
- [2] 2008 IEEE Circuits and Systems Society CSVT Best Paper Award (US \$2000).
- [3] 2007 5th Ennovation Contest First Prize (US \$3000).
- [4] 2005 MiTac Technology Scholarship (US \$400 per month).
- [5] 2005 Class A Scholarship, Graduate Institute of Communication Engineering, National Taiwan University (US \$400 per month).
- [6] 2004 Class A Scholarship, Graduate Institute of Communication Engineering, National Taiwan University (US \$400 per month).

## Press

- [1] SIGGRAPH 2008: The quest for more pixels, *Hack a Day*, Aug. 2008.
- [2] Story about the digital refocusing using the programmable aperture camera, *The China TV news*, Aug. 2008.
- [3] Story about the digital refocusing using the programmable aperture camera, *The Liberty Times*, Aug. 2008.
- [4] Story about the best paper award, *National Taiwan University Newsletter*, vol. 927. July 2008.
- [5] Story about the programmable aperture, *TMBA Journal*, vol. 70, Aug. 2007.