3D Surface Reconstruction Based On Plenoptic Image

by

Haiqiao Zhang

A thesis submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Auburn, Alabama
August 1, 2015

Keywords: Light-field, deconvolution, distance estimation, 3D-volume reconstruction

Abstract

Light-field cameras, also known as plenoptic cameras, have recently become available to the consumer market. Plenoptic cameras can collect the 4D light field information and represent it in a single photograph. This thesis mainly concerns distance estimation based on the photographs taken by light-field camera. A light-field camera is implemented by placing a micro-lens array the main lens and micro-sensors. Each micro-lens record the light information arriving along the rays, not only the amount of the light also the direction. The light-field picture contains more than 2D information, which includes not only the spatial data, but also the angle data. Light-field images record a 4D matrix which contains intensity, position and direction of incoming rays. With this information one can refocus the image with different focal depths to generate a focal stack plane. Each refocused image is a simple 2D image, and the focal stack becomes 3D based on a stack of 2D refocused images. This information can be used to estimate the distance of different objects in the original image. We also combined a segmentation algorithm into deconvolution processing to improve the performance of deconvolution. Various images with different characteristics were used to test the performance and to compare to other depth estimation techniques.

Acknowledgments

I would like thank my family, my father and my mother, for their support and encouragement. I came from China and could not be with my family in the past two years. My parents have made countless sacrifices for me, and have provided me with steady guidance and encouragement. They gave me all the love and support. I could not finish my graduate study without them. This dissertation is dedicated to them. I feel tremendously lucky to have had the opportunity to be the student of Dr. Stanley Reeves, and I would like to thank him for his support. Two years ago, I knew nothing about this thesis and even could not use MATLAB very well. My first class was Digital Signal Processing taught by Dr. Reeves. He is the most serious and intelligent professor I have ever seen. I feel so lucky to be his student to become a quality graduate student. I recognize it is not the most important thing to be a clever man but to be a hard-working one. I would like to also thank Dr. Brian Thurow and Dr. Thomas Denney for serving as committee members. I would also like to thank Paul Anglin for the MATLAB implementation of the deconvolution algorithm used in this work and for the discussions regarding various challenges in applying the technique. He was very patient to deal with every single question I asked no matter how easy or hard it was. I really thank him for his help and his selfless dedication during the whole research. I also would like to thank Jeffrey Thomas Bolan for his help in the setup of the toolkit and the data taken. He was always passionate to answer the questions about the toolkit and sent me the newest version of the toolkit. At the end, I would like to thank God for sending these people into my life and to help me to improve myself. Thank all of you in my life!

Table of Contents

List of Figures

Chapter 1

Introduction

A light-field camera is a camera that uses a micro-lens array to capture 4D light field information on a two dimensional plane[17]. A micro-lens array is inserted in front of the image sensor to record the rich information of the light source. An array of micro-lenses record position and direction information to measure the intensity and direction of every incoming ray. Based on the 4D light field information recorded by the light field camera, the data contains information about different focal depths for different refocused planes as well as various perspectives.

This thesis proposes a depth estimation algorithm based on deconvolution[20]. Based on the advantages of light-field images, the algorithm uses the refocused image with different focal depths to generate the focal stack. But first image stack must be processed to make each point stand out at each focal plane. The method treats this as image blurring that translates the problem into a problem of deblurring an image. So the method can use deconvolution to deal with this problem because image with blur is modeled by a convolution. Here, the method uses introduce the FFT to improve the deconvolution operation. The FFT algorithm offers significant improvement in computational efficiency to compute the convolution and deconvolution. To improve the performance of the deconvolution algorithm, the author considered two segmentation algorithms. After comparing two segmentation algorithm, the author selected the watershed segmentation.

To compare the performance of the deconvolution algorithm, the author considered two comparison algorithms, depth from blur and stereo algorithm. The light field image can be refocused at different focal depths. When the object in the image on focus, the edge of the object is clean and sharp after filtering. The author utilized a measure of the edge strength to achieve depth estimation. The stereo algorithm result were provided by William Roberts, a student of Dr. Brian Thurow in the Department of Aerospace Engineering at Auburn University. This is a new stereo algorithm which is based on the plenoptic image. The author also chose three groups of experiments to compare all three algorithms in different ways.

Chapter 2

Light Field Imaging

2.1 Introduction and History

Conventional cameras are do not record most of the information about the light entering the camera. For example, if we think about the light deposited at one pixel, a photograph does not give us information on how the light coming from one part of the lens differs from the light coming from another. The differences turned out to be significantly important to solving focus problems of conventional cameras.

Light field imaging is gradually becoming of interest in image acquisition. Leonardo da Vinci gave the earliest definition of a light field camera model in the early 16th century. He imagined an "imaging device capturing every optical aspect of a scene". He also defined light rays as "radiant pyramids" where the light contains all the information necessary to recreate any view arriving at any point in space. That means radiant pyramids can intersect each other. Michael Faraday published the idea that light is similar to magnetic fields in 1864[17]. He mentioned that there must exist other wavelengths of light which are not limited to the visible wavelength range 400nm to 700nm, even though we cannot see them. A "light field" camera was demonstrated by Gabriel Lippmann[14].

In the 20th century, a light field camera takes the form of a digital camera that depends on a new perspective. It captures rays instead of pixels, a light field rather than light. After taking the photo, users can focus anywhere in the photo on the computer. Light field cameras can show the difference between each side of the lens (vectors of light where they originate, and how they are manipulated using lenses). Light fields can be defined as the geometrical distribution of light rays flowing in space. The total geometric distribution of light as a plenoptic function over the 5D space of rays[27] — 3D for each spatial position and 2D for each direction of flow.

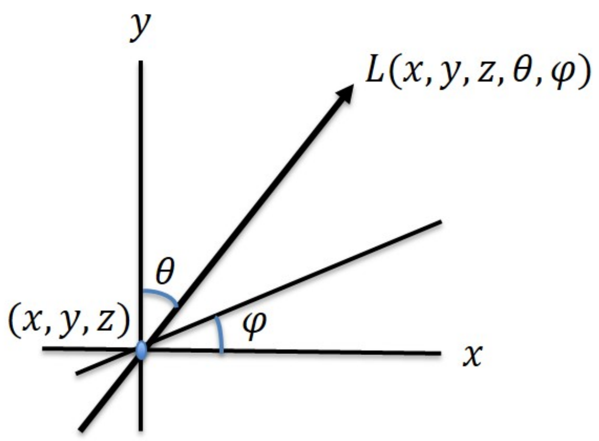


Figure 2.1: geometric construction of five-dimension space of rays

2.2 Camera Fundamentals

A light-field camera is a camera that places a micro-lens array in front of an image sensor to capture 4D light field information on a two dimensional plane[11]. The light-field camera allows one to capture the ray space which holds rich information such as intensity, position and direction of incoming rays. To understand the light-field camera, first consider the diagram of a conventional camera shown below in Figure 2.2.

A conventional camera will have an image sensor, a main lens, and a focal plane as shown in the figure above. The general principle of a conventional camera is very simple. It uses an optical lens to gather the scene produced by the front light,which then focuses the light on the photosensitive element. In order to focus the light onto the photosensitive element, we can adjust the lens to capture light from different directions. Also for this reason, the conventional camera only can take one focal depth for one image.

Compared to a conventional camera, a plenoptic camera has a main photographic lens, a light-field sensor and a digital image sensor. The light-field sensor consist of a CCD sensor

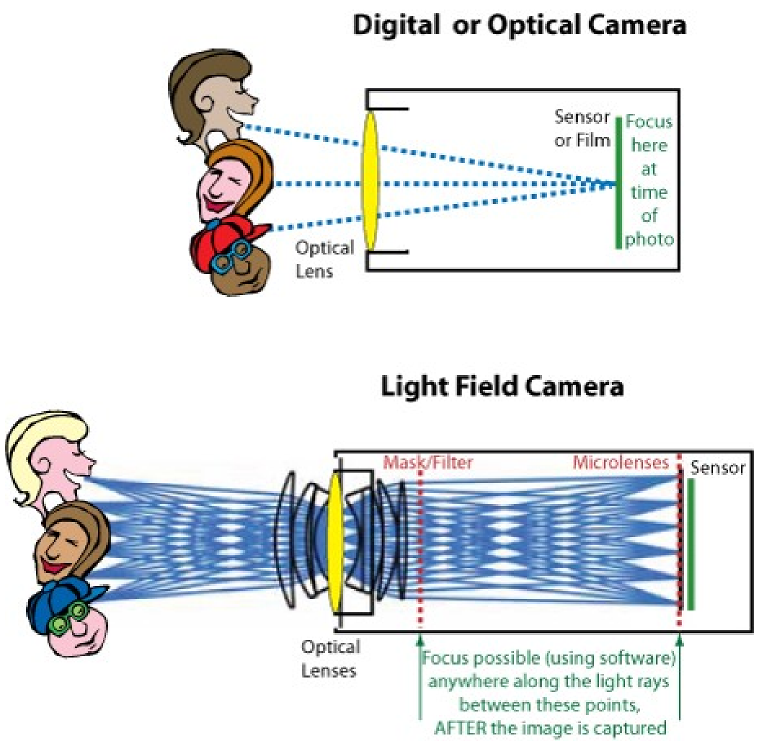

Digital or Optical Camera
Sensor or Film
Focus here at time of photo
Optical Lens
Light Field Camera
Mask/Filter
Microlenses
Sensor
Optical Lenses
Focus possible (using software) anywhere along the light rays between these points, AFTER the image is captured

Figure 2.2: Comparison between conventional digital camera and Light-field camera

and a micro-lens array. The CCD sensor is common in cameras. It will convert light coming from the outside world into electrons and record it in cells. It reads the values from each cell and records it as a digital value. A micro-lens array is attached to a standard sensor, it divides the pixels from the CCD into areas, showing the image at different angles.

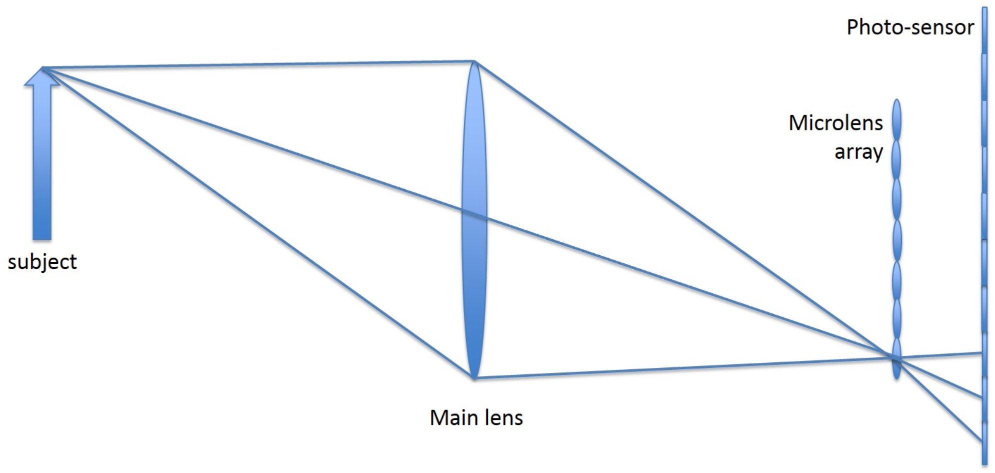


Figure 2.3: Simplified diagram of imaging with a plenoptic camera

After the light passes through the main lens, it will then hit the micro-lens array. Pixels behind the micro-lens array still only records intensity information of the light[19].Direction information of light is recorded because the location of pixels are relate to micro-lens array. For example, if the size of a micro-lens array is 10 by 10 and the pixel array on the sensor is 50 by 50, each micro-lens has 25 pixels. The intensity information goes through 25 different position on the main lens and reaches the micro-lens and is then recorded by these pixels. Thus, simply using a micro-lens array and a photoelectric sensor is equivalent to recording all the light information through the main lens. In post processing,we can trace the light rays back to complete the refocusing, because the light propagation in free space can be uniquely expressed by two planes and four coordinates (four-dimensional volume, called a light field). The imaging process is only computing a two-dimensional integral on this four-dimensional light. A light field camera records the four dimensional light field. Different image depth of focus is accomplished by a two-dimensional integral under different circumstances.

Figure 2.4: Raw plenoptic image from sample experiment

Of course, the disadvantages of a light field camera are also obvious, like insufficient spatial resolution. It is apparent, because the two-dimensional image is recorded by the

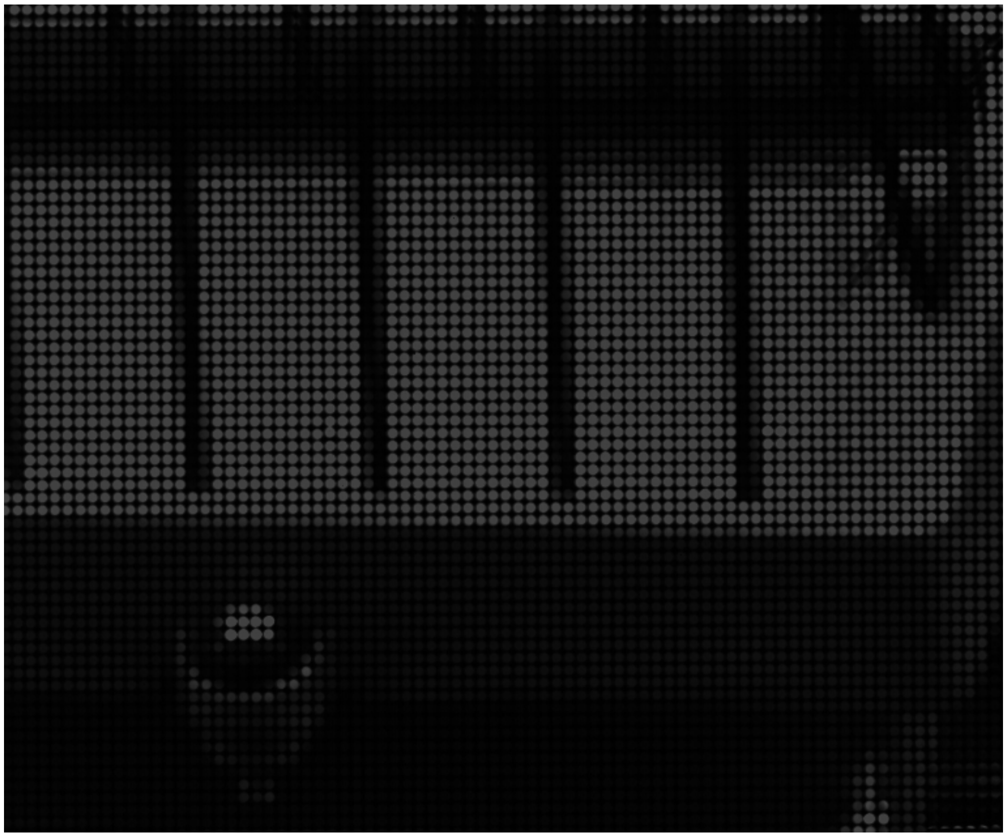


Figure 2.5: Raw plenoptic image from sample experiment with zoomed inset

conventional camera with the same number of pixels, and the full number of pixels are used. A light field camera records four-dimensional images and generates two-dimensional images by doing integrals. The integration process will lose part of the information and reduce a number of pixels to a single pixel in a two-dimensional image, which causes insufficient spatial resolution. Spatial resolution is then proportional to the number of lenses in the micro-lens array, so that a trade-off is necessary between improving the spatial resolution and decreasing the angular resolution.

2.3 Light Field Principle

Since the focal length of the micro-lenses is much smaller than the focal length of the main lens, therefore the main lens can be considered to be at infinity of the micro-lens mirror. Thus the dark green vertical bar area on the main lens just focuses on one pixel through the micro-lens. Micro-lenses are very small compared to the main lens, approximately hundredths of the main lens, so we can consider one pixel to be one sample of all rays inside the blue line. In this way, it will record a light ray inside the camera. Similarly, other pixels also correspond to one light ray. Because of the positional relationship between the main lens and the micro-lens array sensors, pixels behind each micro-lens can be seen as a sample of light coming through the main lens.

Since the position of each pixel is fixed, the position of each micro-lens corresponding to a pixel is fixed. Because light travels in straight lines, it is easy to get the light direction information. For example, assume there are four micro-lenses behind the main lens and eight sensors behind each micro-lens. Red sensor corresponds to the red zone on the main lens in Figure 2.6. If the coordinate of the micro-lens and the coordinate of the red zone on the main lens is 7, the red sensor recordings this light can be expressed as $L(7, s)$. In Figure 2.6, these four light values are $L(4, u)$.

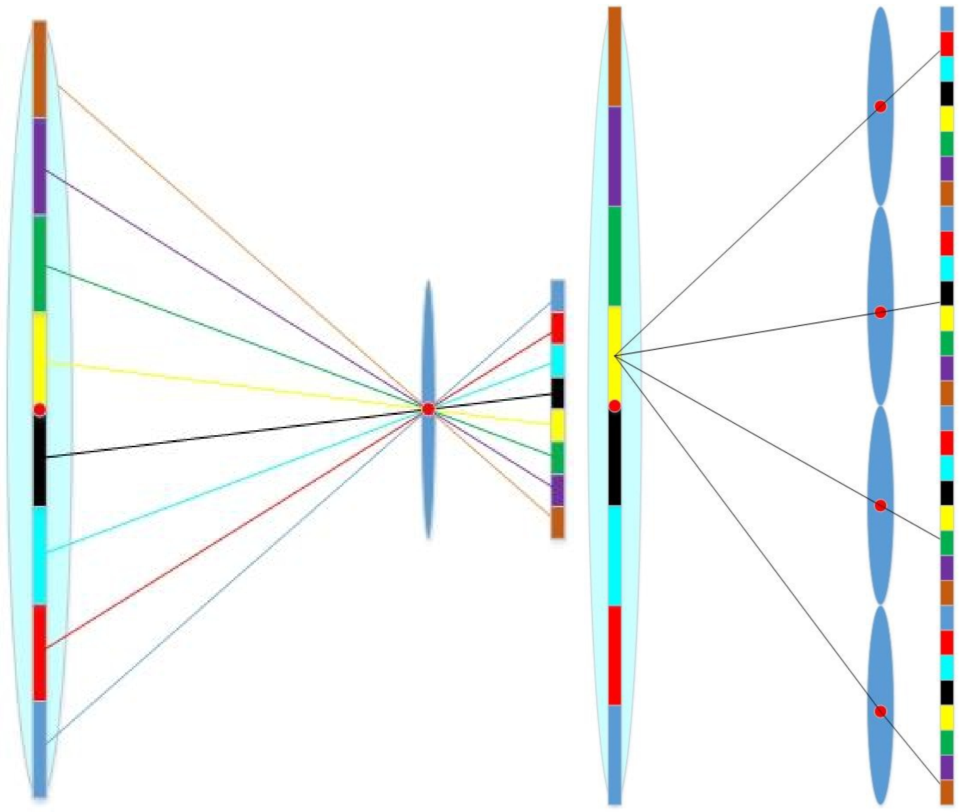
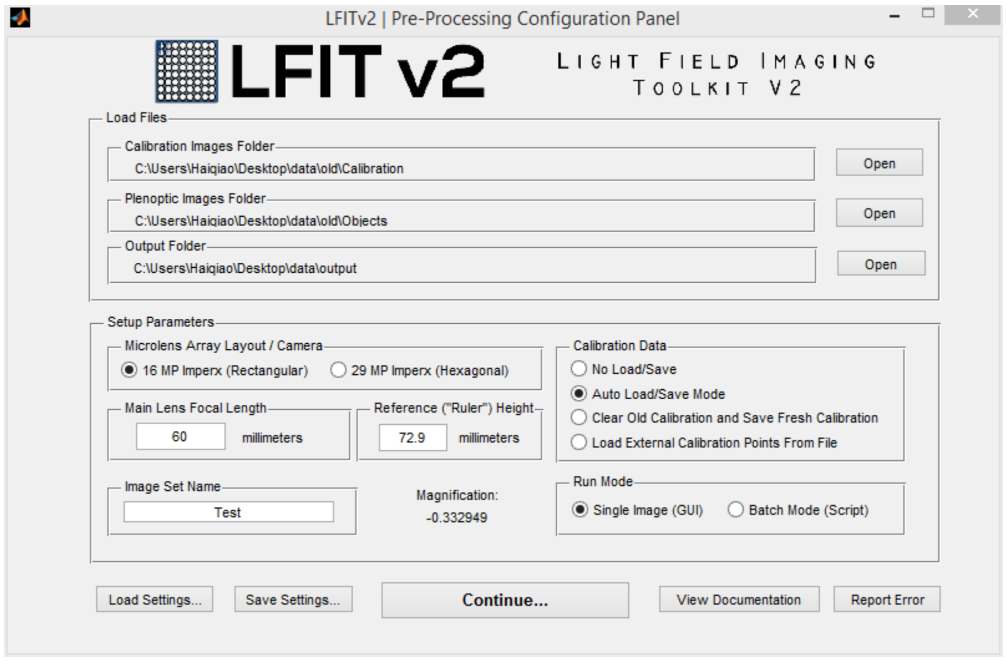


Figure 2.6: Light-field camera image principle

2.4 The Light Field Imaging Toolkit

2.4.1 Setup Steps

All the raw plenoptic images are processed first by the Light Field Imaging Toolkit (LFIT), which was developed by Jeffrey Bolan. This is a powerful tool to process the plenoptic image and a major contribution to this research. LFIT is composed of dozens of functions to implement various processes, with a fully functioning graphical user interface (GUI) shown in Figure 2.7



LFITv2 | Pre-Processing Configuration Panel

LFIT v2 LIGHT FIELD IMAGING TOOLKIT V2

Load Files

Calibration Images Folder: C:\Users\Haiqiao\Desktop\data\old\Calibration

Plenoptic Images Folder: C:\Users\Haiqiao\Desktop\data\old\Objects

Output Folder: C:\Users\Haiqiao\Desktop\data\output

Setup Parameters

Microlens Array Layout / Camera: [✓ 16 MP Imperx (Rectangular)] 29 MP Imperx (Hexagonal)

Main Lens Focal Length: 60 millimeters

Reference ("Ruler") Height: 72.9 millimeters

Image Set Name: Test

Magnification: -0.332949

Calibration Data: No Load/Save [✓ Auto Load/Save Mode] Clear Old Calibration and Save Fresh Calibration Load External Calibration Points From File

Run Mode: [✓ Single Image (GUI)] Batch Mode (Script)

Figure 2.7: LFIT pre-processing GUI window

The initial step is to set up the parameters required in the GUI window. In addition to the raw image required, calibration images are also required. LFIT will averages over one hundred calibration images to reduce the influence of noise on the calibration. After averaging process, the user is prompted to identify three calibration points according to a specified pattern. This step will identify the centroid of each spot through each micro-image, with a result such as that shown in Figure 2.9.

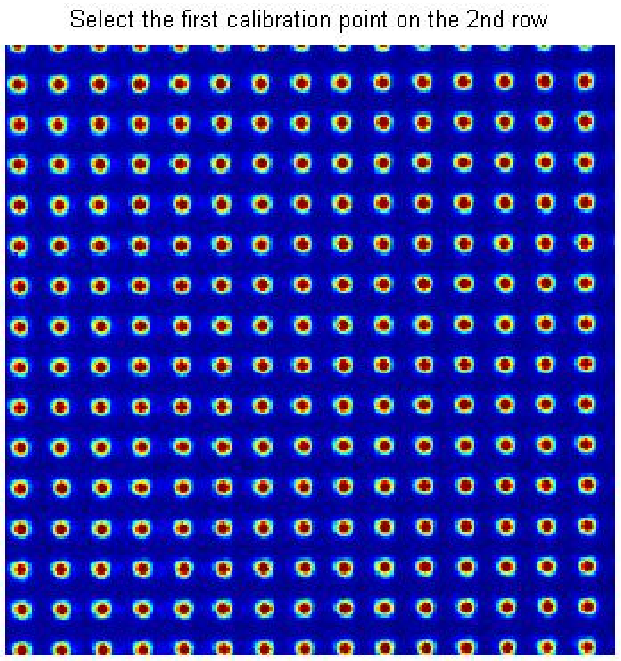


Figure 2.8: Known (u,v) coordinates and desired uniform (u,v) coordinates for microimage

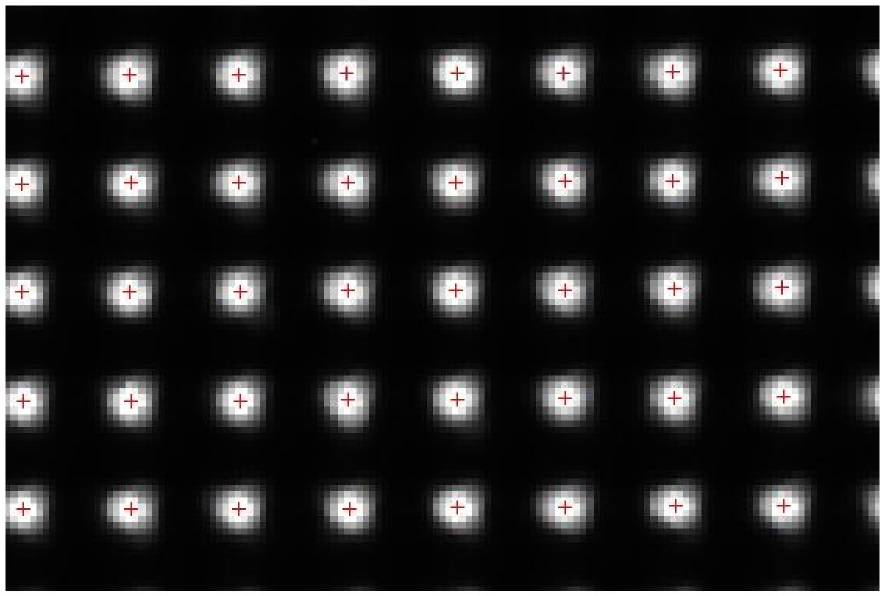


Figure 2.9: Locate the center of each microlens

2.4.2 4D Radiance Array

As we mentioned before, this field is parameterized by five dimensions: the spatial location (x, y, z) and the angle of propagation (θ, ϕ)[11]. But by using the light field camera, the light field function only needs four dimensions to record all this information, defined by $L_f\,(x, y, \theta, \phi)$. In order to capture the additional angular dimension, a light field camera adds a micro-sensor array in front of the image sensor to record all four dimensions of information shown in one image. The purpose of capturing the additional two dimensions of data is to allow the camera to compute a synthetic image plane flexibly from the acquired light field. Each pixel on the image sensor corresponds to a spatial location defined in the image as (s, t). At the same time, each pixel also corresponds to a certain angle that light rays passing through the camera to the certain micro-sensor define as (u, v).

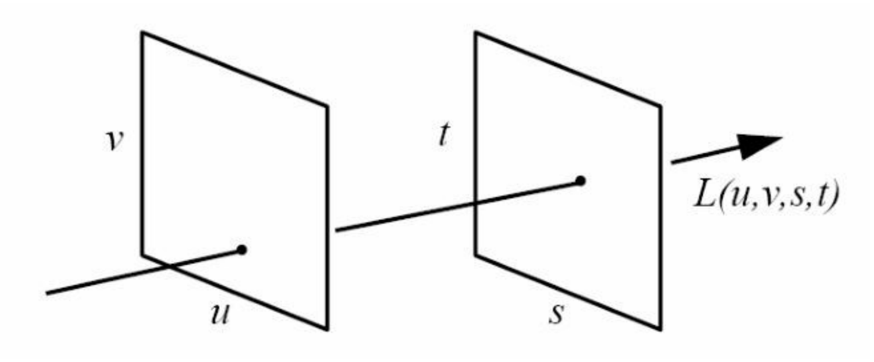


Figure 2.10: The two-plane parameterization for a 4D light field

Each microsensor corresponds to a spatial location defined in the image as (s, t). That means, there always is a certain value of (s, t) associated with a certain microsensor. The (s, t) coordinates can be defined for the entire raw image. At the same time, each pixel also corresponds to a certain angle that light rays passing through the camera to the certain microsensor define as (u, v). That means each pixel on the micro-image corresponds to a different (u, v) value.

2.5 Digital Refocusing

2.5.1 Mathematical Model

Consider the original raw plenoptic image Figure 2.4. The camera is physically focused on the lamp in the Figure 2.4. When all the light rays coming from the subject converge to the sensor, the sensor will record the light rays as a clear and focused image. Figure 2.11 shows the difference between in focus and out of focus. The light field camera is able to change the out of focus image to in focus image when all settings are fixed including the film, the lens and the subject. This is impossible for a conventional camera.

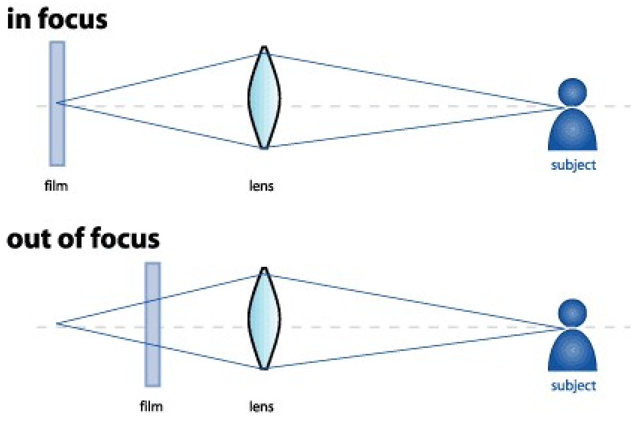


Figure 2.11: Simple schematic explanation of in focus and out of focus

As we mentioned, the light field camera can record the four dimensional information of the light rays, so it is possible to recalculate a new focal plane based on the nominal focal plane of the camera lens. This is the process of refocusing. The main idea of refocusing is described below in Figure 2.12.

There are two light source s R and R', located at different focal planes of the camera. The distance from light source R to the main lens is object distance, given by d_o. The blue lines in the figure represent how the light rays come from the light source R through the main lens to the micro-lens S. We can see that the green lines all converge onto the micro-lens S, which means the light source is in focus. The distance from main lens to micro-lens is image

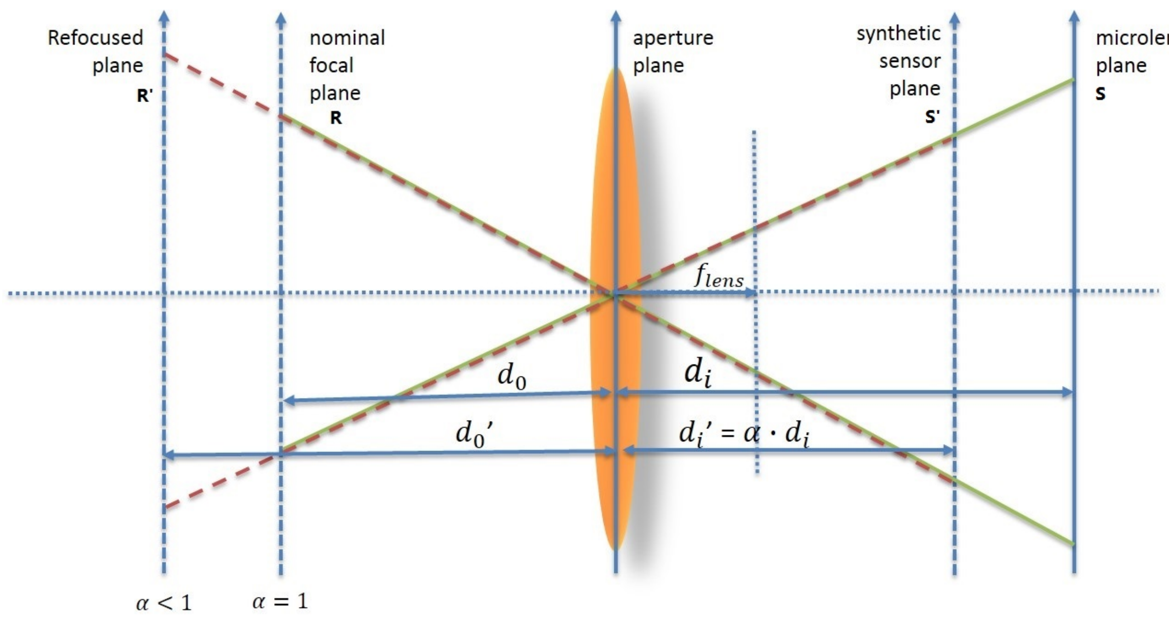


Figure 2.12: Schematic depicting process of computational refocusing

distance, given by d_i. The object distance of light source R' is d_o' and the image distance is d_i'. Here, the distance from light source R' to the main lens is further compared to the light source R, in other words, the light rays from R' will converge to the new plan plane S'. The red line in the Figure indicates the path of a light ray from the light source R'. Note that plan plane S, the micro-lens plane, is also the focal plane of the camera. Plane S' is the synthetic sensor plane, which is synthetic for out of focus point R' but can be in focus in the refocused image. The depth of the refocused plane is controlled by the depth parameter α.

Focusing at different depths corresponds to changing the separation between the lens and the film plane, resulting in a shearing of the trajectory of the integration lines on the ray-space[23]. We can consider the photograph focused at a new film depth of F' by expressing $L_{F'}(x', u)$ in terms of $L_F(x, u)$.

Figure 2.13 below is a geometric construction that illustrates how a ray parameterized by the x' and u planes for $L_{F'}$ may be re-parameterized by its intersection with planes x and u for L_F. By similar triangles, the illustrated ray that intersects the lens at u and the film plane at x' also intersects the x plane at $u + (x' - u)\frac{F}{F'}$. Although the diagram only shows

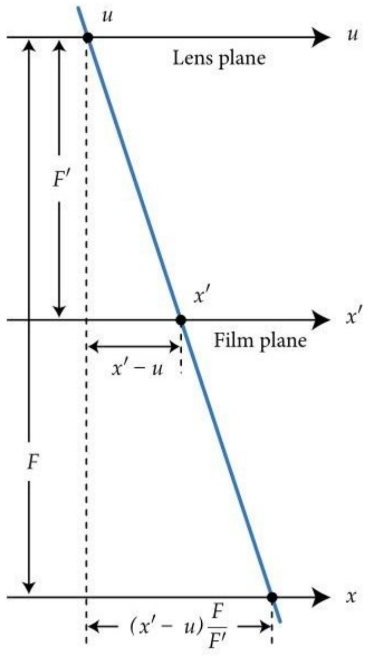


Figure 2.13: Schematic depicting process of computational refocusing

the 2D case involving x and u, the y and v dimensions share an identical relationship. As a result, if we define $\alpha = \frac{F}{F'}$ as the relative depth of the film plane,

$$L_{F'}\left(x', y', u, v\right) = L_F\left(u + \frac{x' - u}{\alpha}, v + \frac{y' - v}{\alpha}, u, v\right)$$

$$= L_F\left(u\left(1 - \frac{1}{\alpha}\right) + \frac{x'}{\alpha}, v\left(1 - \frac{1}{\alpha}\right) + \frac{y'}{\alpha}, u, v\right).$$

This equation [14] formalizes the 4D light field that can be used to focus at different depths.

The ideal set of rays contributing to a pixel in digital refocusing is the set of rays that converge on that pixel in a conventional camera focused at the desired depth. Here, I reference the software that was developed by the Aerospace Engineering department in Auburn University. Refocusing is conceptually a summation of dilated and shifted versions of the sub-aperture images over the entire (u, v) aperture. The pixel at position (x, y) in the sub-aperture image is given by $L_F^{(u,v)}(x, y)$. Digital refocusing can be implemented by shifting and adding the sub-aperture images of the light field, in other words,$L_F^{(u,v)}\left(u\left(1-\frac{1}{\alpha}\right)+\frac{x'}{\alpha}, v\left(1-\frac{1}{\alpha}\right)+\frac{y'}{\alpha}\right)$ is simply the sub-aperture image $L_F^{(u,v)}$, dilated by a factor of α and shifted by a factor of $\left(u\left(1-\frac{1}{\alpha}\right), v\left(1-\frac{1}{\alpha}\right)\right)$.

The problem occurs when the shift in neighboring sub-aperture images differs by more than one pixel, so edges in one sub-aperture may not be blended smoothly into the neighbor image during the summation process. A solution used in this software is to super-sample the aperture plane. The super-sampling rate is chosen so that the minimum shift is less than one output pixel.

2.5.2 Focal Stack Generation

To refocus one image containing several objects with different focal depths requires more than one α value. To record all α settings and the corresponding refocused image at different focal depths, a focal stack is generated from a single raw plenoptic image. The focal stacks are typically defined by a fixed range of depth parameter α and the number of the images in the stack, which are spaced uniformly in α. To find the corresponding α value for different objects in the image, we can select the in focus slice of the object from the focal stack. The depth of the refocused focal plane can be determined by a maximum sharpness of the selected object. Each slice in the focal stack corresponds to a differentα value which indicate the location of the synthetic focal plane. The number of this slice also indicate the relationship of object distance and image distance. The the relationship is shown in Figure **??**.

So the equation of this relationship can be defined by

$$\frac{1}{d_o}+\frac{1}{d_i}=\frac{1}{f_{mainlens}}$$

Here, d_o corresponds to object distance, d_i corresponds to image distance, and $f_{mainlens}$ to focal length of the main lens. The image distance for an experiment can be easily measured by using the magnification[30] to calculate or just record during the experiment. So the refocus equation can be rewritten as

$$\frac{1}{d'_o} + \frac{1}{\alpha \cdot d_i} = \frac{1}{f_{mainlens}}$$

$$d'_o = \left(\frac{1}{f_{mainlens}} - \frac{1}{\alpha \cdot d_i} \right)^{-1}$$

The above d'_o is the depth location of the refocused focal plane. So if we set an appropriate range for α, then we can calculate the depth location of the refocused focal plane for each layer of the focal stack.

2.5.3 Alpha Scaling

When we do the refocus processing, we need to set an α to determine the focal plane, that means we need to decide which part of the original image we should focus on. So we can set a range of α to include all parts of the original image depth. And at the same time, we also set the step to determine the distance between each focal plane. As stated earlier, the depth parameter α is used to define the location of a new focal plane. A diagram below in Figure 2.14 illustrates the relation between object distance and image distance. The blue lines correspond to the nominal focal plane where alpha $\alpha = 1$. When refocusing beyond the nominal focal plane, the α value should be less than 1 to get the corresponding focal plane. The red line in Figure 2.12 illustrates this situation. When the desired refocused plane is closer to the camera, the α value should be larger than 1 to get the corresponding focal plane.

As a demonstration we choose the range of α to be [0.65 2.2] and the step to be 200. We show several refocused images from the focal stack in the figure. In this image, the first object is a book that is 0.95 m from the main lens, the second object is a lamp and is 1.30 m far from the main lens, the third object is a bucket and is 1.85 m from the main lens. The camera is focused at 1.4m. The figure shows the refocused image when focused on each of the three objects as well as for $\alpha = 2.2$.

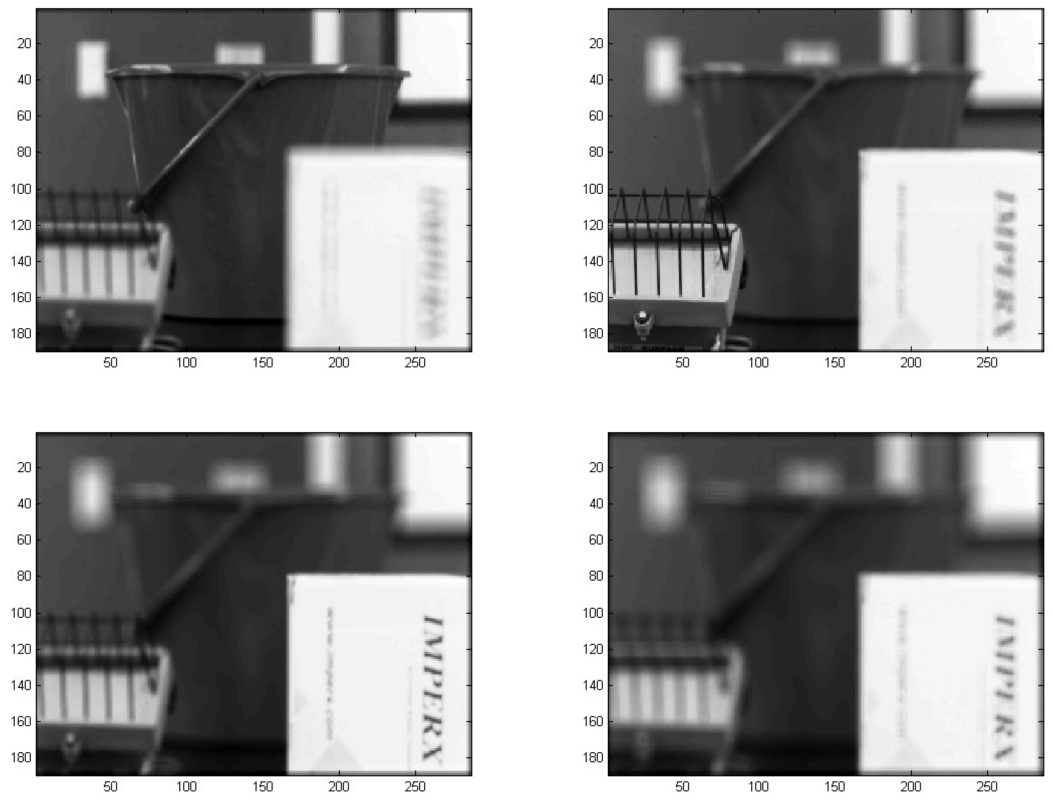


Figure 2.14: Direct implementation of refocusing equation with different α. $\alpha = 0.7$ (top right), $\alpha = 1$(top left), $\alpha = 1.62$ (bottom left), $\alpha = 0.3$ (bottom right)

Chapter 3
Depth Estimation

3.1 Depth from blur

Depth can be estimated from blurred images in a focal stack. When the object is displaced significantly from the current focal plane, a blurred image will be formed on the image sensor. The larger the depth displacement is, the more defocused the image is [6]. Using this feature and combining it with some basic image processing, we can calculate the depth. Fortunately, light field camera can obtain 4D information and reconstruct a 3D focal stack from the 4D information. Thus, this 3D focal stack corresponds to actual spatial planes. In this way, it is not necessary to process real-world planes; instead we only need to deal with a digital focal stack. We only need to process digital focal planes, and then the result in real space can be found by this correspondence [13]. This is the fundamental idea of the depth from blur algorithm.

A focused image contains more high-frequency energy, but a defocused image is smoother so that it has less high-frequency energy. When we take a photo of a friend, we want their face to be clear so that every edge of the face is sharp; however, when we change the focal length of the optical system, the focal plane of this image is changed. In this case, the face shows more blur across edges, which reflects a reduction in high-frequency energy. Therefore, an effective way to determine whether an image is focused is to measure the high -frequency energy.

There are several high frequency detection operators, like the Roberts operator, the Sobel operator, the Prewitt operator and the Laplacian operator. All of these operators have their own specific benefits. For example, the Roberts operator is good for sharp low-noise images, the Sobel operator can lower the noise, and the Prewitt operator is insensitive to noise and can extract high-frequency components even when the value changes slowly. I chose the Laplacian operator as a high frequency detection method in this technique, because the Laplacian operator can detect gradients not only in the horizontal or vertical direction but in any direction around the selected pixel. Other operators cannot do this, and the pixels in the blurred image are not only blurred in two directions, but in all directions. Thus, the Laplacian operator is the best operator for this technique. After calculating the gradients of every point in every plane, I can plot gradient energy of every point in the (x, y) plane along the z direction. The first step is more like an edge detection, to eliminate the out-of-plane objects and maintain the edge information of the in focus object. I assume coordinates in the image are (x, y), and planes are located in the z direction. If the object is within the depth range of the focal stack, there must be a peak in the gradient curve, where the horizontal axis is in the z direction and the vertical axis is gradient energy of the pixel of the object. Once the peak is found, the horizontal coordinate of this peak is the position of the focused image in the focal stack.

The depth from blurred image algorithm can be briefly described by the following steps:

- Use Toolkit to generate the refocused image focal stack $f_l\left(x, y, z\right)$ with range of depth parameter alpha.
- Filter refocused image focal stack with the high-frequency detection operators in (x, y) frame by frame.
- Square every pixel in focal stack $f_l - (x, y, z)$ to get ${f_l}^2 - (x, y, z)$, then filter every resulting (x, y) plane with $L \times L$ size window, $g\left(x, y, z\right) = {f_l}^2\left(x, y, z\right) * \Pi\left(x, y\right)$.
- Pick out maximum absolute value in z direction for each pixel (x, y).

The steps above are enough to find depth,; however, this method only works well near edges and not on smooth objects. I used a Gaussian filter to spread the gradient value, so that the gradient value on the edge can affect pixels nearby. In this way, when an object is located before another object, the smooth part between edges would be affected by edges nearby. Thus the position of this part is determined as between two objects. It is true that applying this method reduces the spatial resolution of the depth measurement, and errors may be significant, but this approach at least assigns some values.

3.2 Depth Estimation from Stereo Algorithm

The main idea of the stereo algorithm is to triangulate two images of the same scene point to recover depth. Two images of the same scene taken from slightly displaced viewpoints are called stereo images. Here, the stereo images are generated based on a raw plenoptic image by the toolkit and are called perspective images.

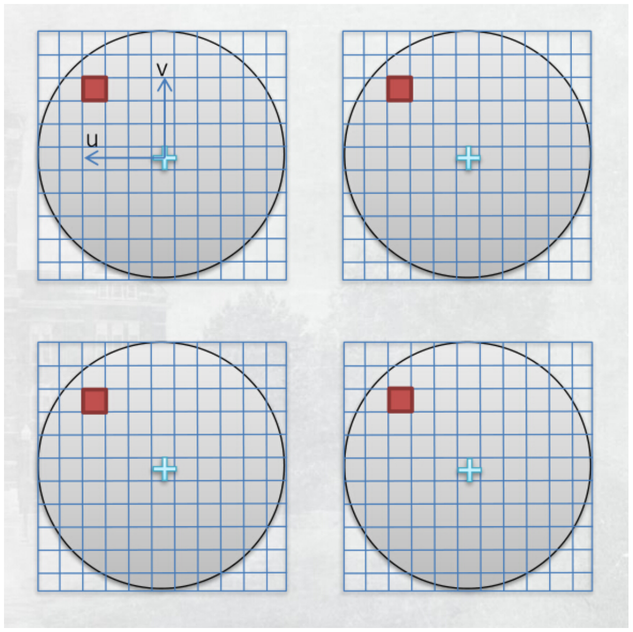


Figure 3.1: Simplified diagram of perspective view generation

As mentioned before, the light field has four dimensions to record light information, defined by $L_f(x, y, \theta, \phi)$. As shown in Figure 3.1, the coordinates (s, t) correspond to each micro-lens on the micro-lens array and the coordinates (u, v) correspond to image aperture[12]. Each (s, t) coordinate is associated with a specific micro-lens and (u, v) coordinate is associated with a specific pixels underneath a micro-lens. So, using a raw plenoptic image to generate the perspective image is analogous to extracting the same pixel which has the same (u, v) location beneath each micro-lens. The example left perspective image and right perspective image are shown in Figure 3.2

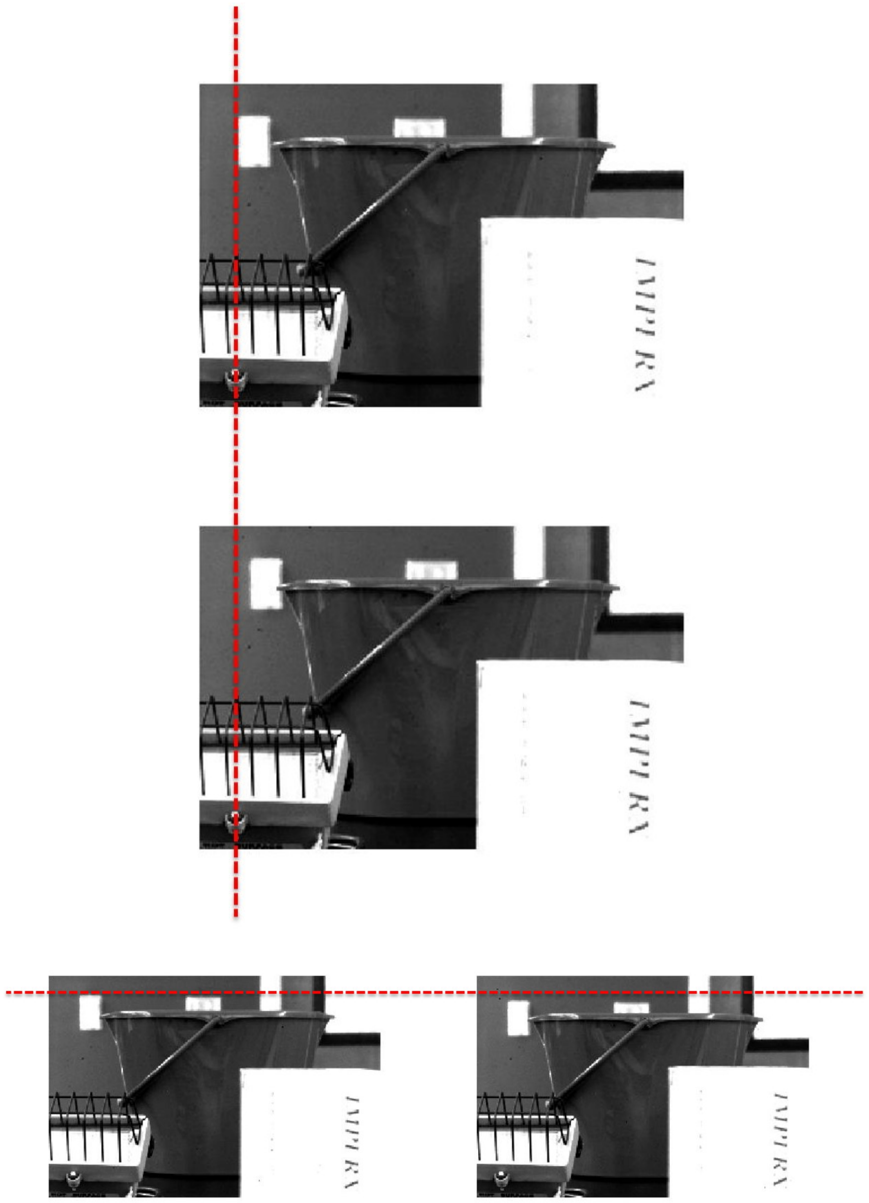


Figure 3.2: Two example perspective views of the sample scene

The stereo problem[2] is usually broken in to two sub-problems: extraction of depth information from stereo pairs and use of depth data to visualize the world scene in three-dimensions by a suitable projection technique[16]. To illustrate how a typical stereo imaging system operates, let us take a look at the stereo algorithm model for obtaining perspective images.

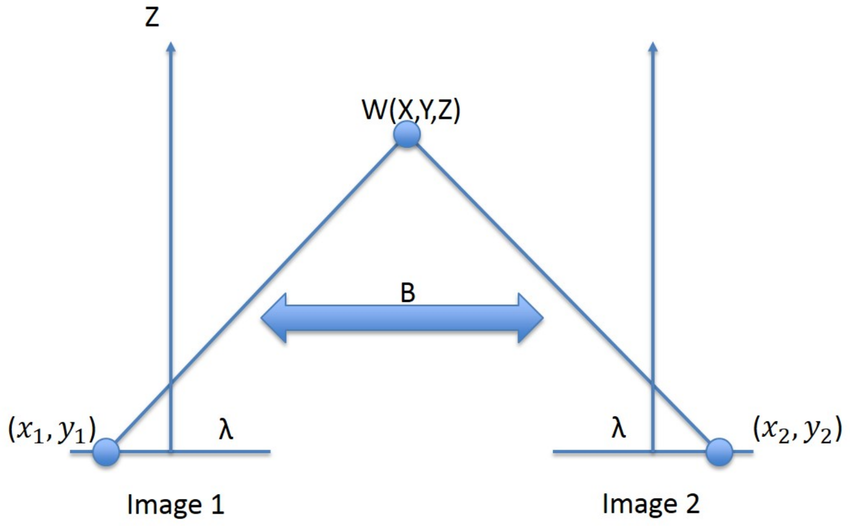


Figure 3.3: geometric schematic explanation of traditional stereo algorithm

From Figure 3.3, we can relate the stereo problem to a mathematical model:

$$X_1 = \frac{x_1}{\lambda}(\lambda - Z_1), X_2 = \frac{x_2}{\lambda}(\lambda - Z_2)$$

According to the triangular calculation, $X_2 = X1 + B$ and $Z_1 = Z_2 = Z$, so the above formula can be rewritten by

$$X_1 + B = \frac{x_2}{\lambda}(\lambda - Z_2)$$

So the depth can be calculated as

$$Depth = (z - \lambda) = \frac{\lambda B}{x_1 - x_2} = \frac{K}{x_1 - x_2}$$

where $x_1 - x_2$ is the disparity[8] of the image. Thus Depth is inversely proportional to $x_1 - x_2$, where x_1 and x_2 are pixel coordinates of the same world point when projected onto the stereo image planes. The disparity is the difference between the x coordinate of two corresponding points. It is typically encoded with a gray scale image (closer points are brighter). Disparity is higher for points closer to the camera. Based on the calculated disparities, a disparity map is obtained. The lighter shades (greater disparities) represent regions with less depth as opposed to the darker regions which are further away from us.

The problem of finding x_1 and x_2 in the stereo pairs is done by a stereo matching technique. The goal of stereo matching algorithms is to find matching locations in the left and right images, specifically to find the coordinates of the pixel on the left and right images which correspond to the same world point. This is also called the correspondence problem. A common approach to finding correspondences is to search for local regions that appear similar. This approach requires trying to match a window of pixels on the left image with a corresponding sized window on the right image.

Here the stereo algorithm is provided by William Roberts. In his algorithm, the stereo matching is done by normalized cross-correlation which uses a small template window surrounding a comparison pixel to compare with a larger search window at the same location on the perspective image. The disparity estimate is based on the surrounding correlation surface. Defining a confidence coefficient improves the reliability of the disparity estimation. Depth is then calculated from disparity.

3.3 Depth Estimation from Deconvolution Algorithm

3.3.1 Model of Image

Deconvolution is an algorithm-based process used to reverse the effects of convolution on recorded data. Given an observed blurry image y, the deconvolution problem can be defined as[3]:

$$y = f * x$$

Convolution is a linear operator, and thus the equation above can be written as:

$$y = C_f x$$

where C_f is an $N \times N$ convolution matrix. In the frequency domain, where $F(\nu,,\omega)$, the Fourier transformation of C_f is a diagonal matrix:

$$Y(\nu,\omega) = F(\nu,\omega)\, X(\nu,\omega)$$

Here (X, Y) are the frequency representation of (x, y) and are coordinates in the frequency domain. If the matrix is a full rank matrix, and no noise is involved in the imaging process, the simplest approach to deconvolve y is:

$$X(\nu,\omega) = Y(\nu,\omega)\,/F(\nu,\omega)$$

This, however, is very rarely stable enough. For example, the inverse is not defined in frequencies (ν,ω) for which $F(\nu,\omega) = 0$. Even in case $|F(\nu,\omega)|$ is not exactly 0 but small, the inverse is very sensitive to noise. If Y includes some noise then:

$$Y(\nu,\omega) = F(\nu,\omega)\, X(\nu,\omega) + N(\nu,\omega)\, X(\nu,\omega)$$

then the inverse will produce:

$$Y(\nu,\omega) = X(\nu,\omega) + N(\nu,\omega)\, X(\nu,\omega)\,/F(\nu,\omega)$$

3.3.2 Point Spread Function

Image blurring is based on the concept of a three-dimensional point spread function (PSF). A PSF describes the response of an imaging system to a point source or point object. A more general term for the PSF is a system's impulse response, the PSF being the impulse response of a focused optical system[29]. Ideally, the PSF chosen will completely characterize the imaging system. For the imaging systems considered here, the PSF will resemble a double cone where the apex represents the point where objects are optimally focused.

The PSF of an imaging system directly impacts image refinement using deconvolution algorithms [33]. A point input can be represented as a single pixel in the image. So the point spread function is based on a small point source infinitely diffusing in the object space. It cannot perfectly capture the energy of light into a three-dimensional image of the point, because the imaging system only can collect a fraction of the light that spread out. Thus, the point spread function is formally defined as the three-dimensional diffraction pattern generated by an ideal point source of light.

The point spread function also can be defined as a mathematical model of diffraction. An ideal point spread function can have perfect symmetry. That means an ideal point spread function is generally symmetric both in axial (above and below the x-y plane) and radial (rotationally about the z-axis) directions.

Generally, a PSF can be created in two ways: analytically and experimentally[15]. Because the imaging system responds to a point input, a PSF could be derived by the camera geometry and optical path. But this analytical way is hard to achieve due to the complex imaging system. So we take an alternate approach to find a PSF by imaging a point source experimentally[5, 25]. This approach is based on the experimental data including the calibration and an approximate point source. This approach makes the experiment more precise and reliable because it accounts for changing the input or the setup. Consider a hypothetical point source of light located at the nominal focal plane of the plenoptic camera. We only want one micro-lens to be illuminated with the right setup. That is the point source that would illuminate a single spatial sample while the rays strike the main lens at all angles for a complete angular sampling. After appropriate setup, we capture a whole white raw plenoptic image. Only allow the center micro-lens to be illuminated and zero all other micro-lenses. Then use Toolkit with calibration and this single particle image to generate a focal stack. This focal stack of the modified raw image constitutes the three dimensional PSF used.

A vertical slice view of the 3D PSF is given below in Figure 3.4.

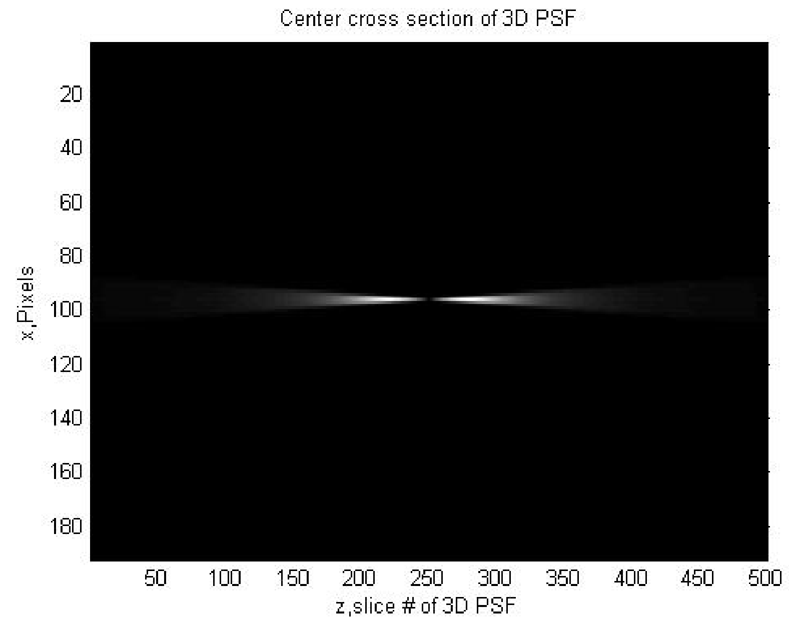


Figure 3.4: x-z slice of PSF focal stack

Based on the conventional imaging mode (wide field, confocal, transmitted light), the PSF has a different shape and structure. To describe the PSF in three dimensions, it is common to apply a coordinate system of three axes where x and y are parallel to the focal plane of the object and z is parallel to the optical axis of the lens. The PSF appears as concentric rings in the x-y plane, and resembles an hourglass in the x-z and y-z planes. At the nominal focal plane, the three-dimensional PSF only is shown by a single pixel. Moving away from the focal plane, the PSF will spread out to a larger circle. This feature is shown in two example results below in Figure 3.5

From the intensity curve, we can see that at the focal plane, the intensity is concentrated to 1 at the center of the focal stack. As the plane moves away from the center focal plane, the intensity decreases rapidly to 0. It has a sharp peak at the center which reveals the fact

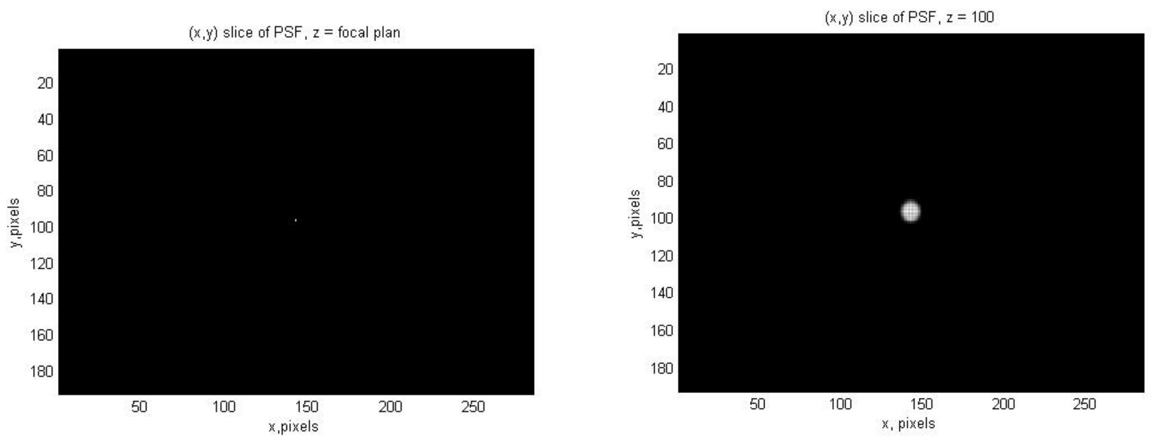


Figure 3.5: X-z slice of PSF focal stack. Left figure is center plane in focal stack, right figure is z-direction in focal stack

that the light information will spread out in the real world. These all can demonstrate that this approach can create an ideal PSF closer to the real way light spreads.

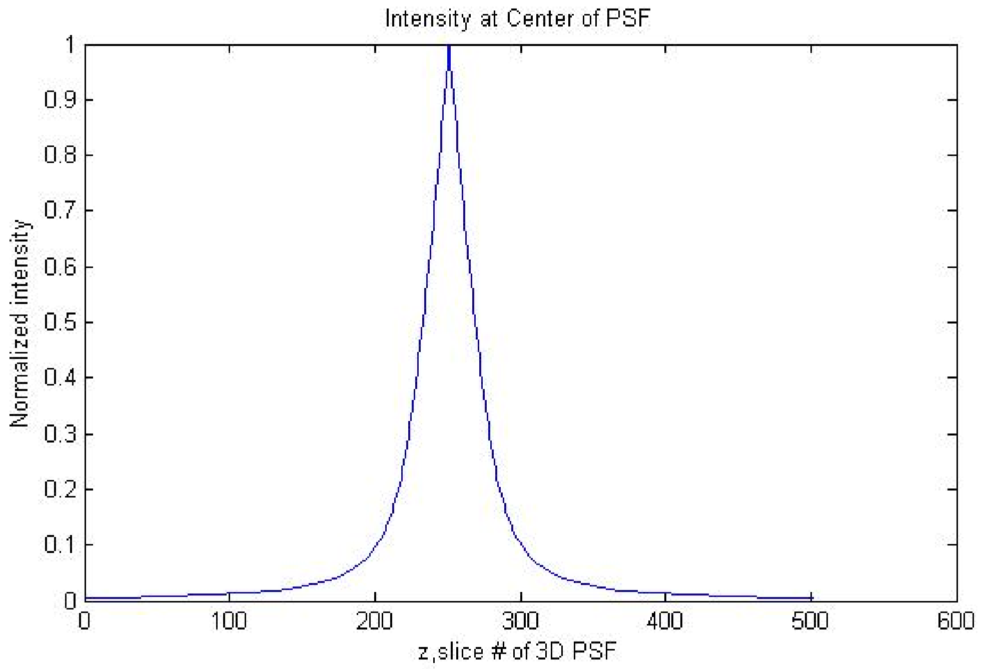


Figure 3.6: Intensity profile of PSF with respect to depth

3.3.3 Three-Dimensional Deconvolution

In image processing, deconvolution is used for image restoration in many applications[32]. In other words, deconvolution is used for getting rid of the effect of blur in the image, by applying a mathematical algorithm. By doing so, the user can get clearer and sharper images of specific focal planes. Every light source emits scattered light, so this can lead to a blurry signal in the real world. To overcome this problem, the author used three dimensional deconvolution here.

For two dimensional images, a blurred image $g(x, y)$ can be estimated by the convolution of a linear shift-invariant blur which is characterized by the PSF $h(x, y)$ on original image $f(x, y)$. So the processing of recovering the true image $f(x, y)$ requires the deconvolution of the blurred image $g(x, y)$ with PSF $h(x, y)$. Here, we assume the PSF $h(x, y)$ is known explicitly prior to the deconvolution. So this problem becomes the classical linear image restoration problem.

Figure 3.7 shows a three dimensional small structure of a glowing square with a blurry surrounding simulated by MATLAB.

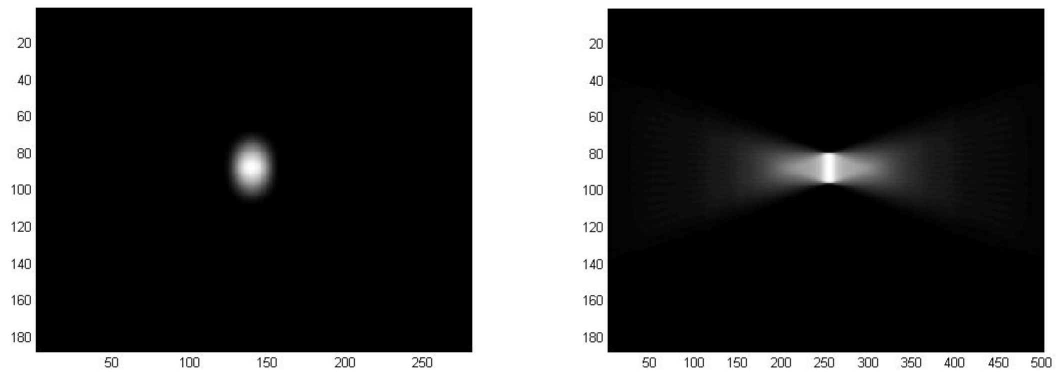


Figure 3.7: x-y slice of PSF focal stack (left) and y-z slice of PSF focal stack (right)

Due to spread light, the x-y slice view shows some blur around the object which represents the distortion of the object by the optical system. This is an irreversible process by an optical system. The x-z view looks like two cones facing each other's apexes.

For three-dimensional deconvolution, the goal of deconvolution is to eliminate the useless information with the help of a mathematical calculation[15]. With the powerful tool of deconvolution it is possible to eliminate out-of-focus energy. Here, we want to concentrate the energy back to where it spread out from. From the mathematical analysis, the blur comes from the convolution of the original image and PSF. Blur can be viewed as energy from the light source spread out. When we record the light source out of focus, the image looks blurry. The performance of the deconvolution is seen by the fact that the energy which spread out before is back to the former place. As we mentioned, the PSF $h(x, y, z)$ is formally defined as the three-dimensional diffraction pattern. So the blurred image $g(x, y, z)$ can also be defined as a three-dimensional diffraction pattern. Additional, because we want to estimate the distance of the object in the image, this process must not only eliminates the noise or the effect of the PSF, but it also eliminates the energy coming from other objects when we focus on one object. That means we can see the object when it is in focus during the z-stack after deconvolution without interference from other objects.

For three-dimensional deconvolution, formation of the z-stack should be started from in front of the object of interest in the image, where relevant structures are still out of focus. The same is true for the back end. Compared to a dark object or image, a bright object in the image can concentrate the energy well during the deconvolution.

All these improvements are based on high-performance hardware and software. Of course, the better the original pictures are, the better the deconvolution image will be.

Recollect that image restoration refers to the minimization or even removal of known or unknown degradations in an image. This includes deblurring of images degraded by the limitations of the sensor or filtering of noise from the acquisition process, and correction of geometric distortions or non-linearity due to sensors [20]. If the imaging system is linear, the image of an object can be expressed as:

$$g\left(x,y\right)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}h\left(x,y;\alpha,\beta\right)f\left(\alpha,\beta\right)d\alpha d\beta+\eta\left(x,y\right)$$

where $\eta(x,y)$ is the additive noise function, $f(\alpha,\beta)$ is the object, $g(x,y)$ is the image, and $h(x,y;\alpha,\beta)$ is the PSF.The ";" is used to distinguish the input and output pairs of coordinates here. A typical image restoration problem can be formed based on a PSF, a blurred image, and the statistical properties of the noise and the factors affecting the noise contribution.

Deconvolution based on FFTs offers a fast and efficient way of removing the out-of-plane energy from volumetric estimates generated from plenoptic data[18, 7]. FFT-based deconvolution is based on the convolution theorem of the Fourier Transform:

$$f(x,y,z) * h(x,y,z) \xrightarrow{\Im} F(\omega_x,\omega_y,\omega_z)\, H(\omega_x,\omega_y,\omega_z)$$

If we consider the noise in the optical system with the imaging model mentioned above,

$$g(x,y,z) = f(x,y,z) * h(x,y,z) + \eta(x,y,z)$$

So based on the Fourier Transform, the imaging model with noise can be defined as,

$$g(x,y,z) = f(x,y,z) * h(x,y,z) + \eta(x,y,z) = \Im^{-1}[F(\omega_x,\omega_y,\omega_z)\, H(\omega_x,\omega_y,\omega_z) + N(\omega_x,\omega_y,\omega_z)]$$

This equation provides the means to recover the original image $f(x,y,z)$ from the noisy image $g(x,y,z)$ by simple division in the frequency domain,

$$\hat{F}(\omega_x,\omega_y,\omega_z) = \frac{G(\omega_x,\omega_y,\omega_z)}{H(\omega_x,\omega_y,\omega_z)} = F(\omega_x,\omega_y,\omega_z) + \frac{N(\omega_x,\omega_y,\omega_z)}{H(\omega_x,\omega_y,\omega_z)}$$

Deconvolution based on FFT offers a fast and efficient way of removing the out-of-plane energy from volumetric estimates generated from plenoptic data. The speed of computation is the first advantage of FFT-based deconvolution. Compared to direct computation of convolution, FFT-based convolution transfers the computation from time domain to frequency domain. In the frequency domain, the operations decrease from N^2 to $O(N \log N)$. At the same time, the PSF cannot guarantee a perfect reconstruction of the volume any longer because it falsely assumes the volume is periodic. Furthermore, the convolved image is corrupted by some level of noise, which can be amplified considerably during deconvolution. A compromise is typically chosen in order to limit the impacts of noise amplification and PSF mismatch by viewing the image as a random process and then minimizing the expected value of the squared error:

$$e^2 = E\left(f - \hat{f}\right)^2$$

Then result of this minimization is the Wiener filter given by

$$\hat{F}\left(\omega_x, \omega_y, \omega_z\right) = \left[\frac{H^*\left(\omega_x, \omega_y, \omega_z\right)}{|H^*\left(\omega_x, \omega_y, \omega_z\right)|^2 + \frac{S_\eta\left(\omega_x, \omega_y, \omega_z\right)}{S_f\left(\omega_x, \omega_y, \omega_z\right)}}\right] G\left(\omega_x, \omega_y, \omega_z\right)$$

where S_η and S_f represent the power spectrum of the noise and the original image volume, respectively. The inverse filter cannot handle noise well [21]. Zero or near zero terms can cause significant noise amplification. So here we use a more helpful filter, the regularized three-dimensional Wiener filter. The regularized three-dimensional Wiener filter replaces the ratio of the power spectra by a regularization parameter K,

$$\hat{F}\left(\omega_x, \omega_y, \omega_z\right) = \left[\frac{H^*\left(\omega_x, \omega_y, \omega_z\right)}{|H^*\left(\omega_x, \omega_y, \omega_z\right)|^2 + K}\right] G\left(\omega_x, \omega_y, \omega_z\right)$$

Here, K is a regularization parameter of the ratio of the power spectra. When K is set to zero, the system is noiseless. To illustrate the performance of K during the deconvolution, a test case is set where two white targets with different focal depth are placed. The following figures show the different focal depths for these two targets. The right target is on focus in the left figure, which focuses at $\alpha = 1$, and the left target is on focus in the right figure, which focuses at $\alpha = 0.95$ (Figure 3.8).

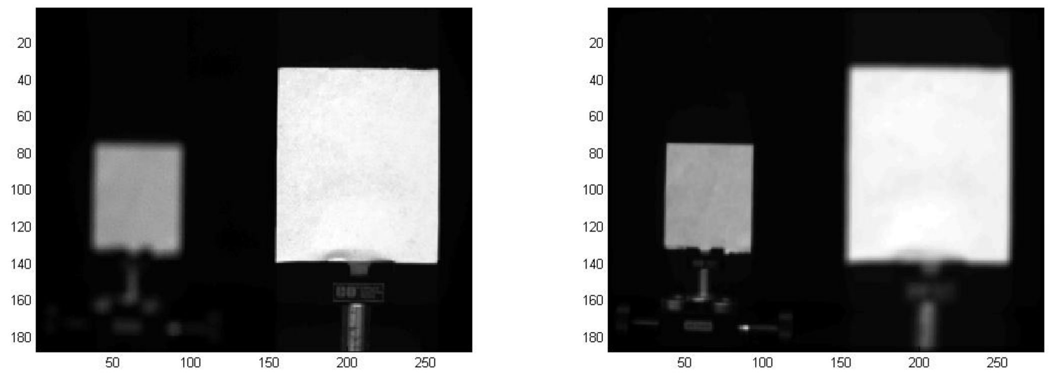


Figure 3.8: Refocus calculation with $\alpha = 1$ (left) and $\alpha = 0.95$ (right)

The corresponding focal stack is generated by the toolkit with the plenoptic image. The resulting focal stack is shown in Figure 3.9, which clearly depicts the blur resulting from out-of-plane points.

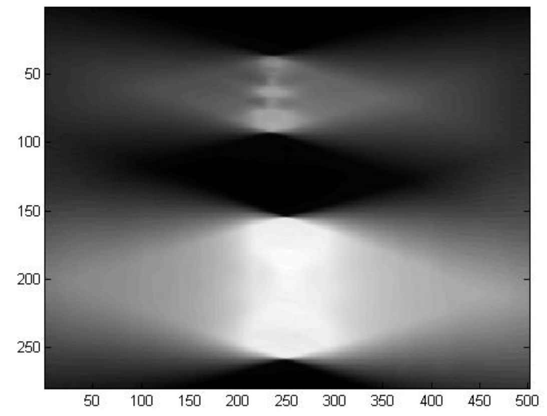


Figure 3.9: y-z slice of sample experiment image in focal stack (equal to top view of focal stack)

The following figure compares the results of the deconvolution with various levels of the regularization parameter K. This regularization parameter is a constant that is important for deconvolution to get an appropriate result. The regularization parameter needs to be adjusted to the requirement of deconvolution and the smoothness of the objects in the image.

In the following figures, utilizing regularization values K are $K = 0.1$(left), $K = 0.0001$ (center) and $K = 0.000001$(right).

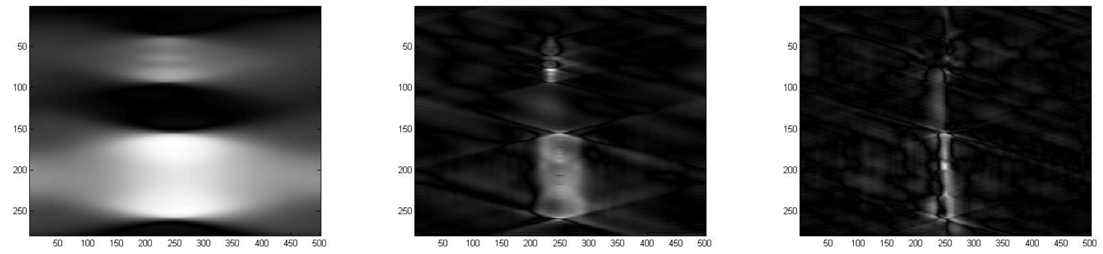


Figure 3.10: Deconvolution result with different regularization values. From left to right, $K = 0.1$ (left),$K = 0.0001$ (center), $K = 0.000001$ (right)

A large regularization parameter will reduce high-frequency noise. But if the regularization parameter is too large, the result will have a bad reconstruction because the large regularization reduces too much, like the result shown in the left figure. A small regularization parameter will maintain this high frequency content, but also maintain the noise. If the regularization parameter is too small, the deconvolution cannot deblurr very well, and the quality of the reconstruction will be reduced, like the result shown in the right figure. Selecting the appropriate regularization parameter gives a clear result, with significantly reduced blur. The following figures show intensity of one arbitrarily selected point in the image shown above. Z-stack direction is the positive direction of x-axis in the following figures that represents the distance relationship between the targets and the camera in real world. The y-axis represents the intensity of the light source coming from the real world.

In Figure 3.11, the x axis for these four figures is the z-direction in the focal stack and the y axis is the intensity value of pixels. The upper left shows the intensity of an arbitrary pixel in the z direction in the original focal stack. We can see there is a flat region in the center which represents the range of light spread out from the focal plane. The upper right figure is the result from the deconvolution with regularization parameter $K = 0.1$. In this case, the regularization parameter is not large enough to get a sharp peak, which means the energy is still not concentrated at the focal plane. The lower left figure is the result from

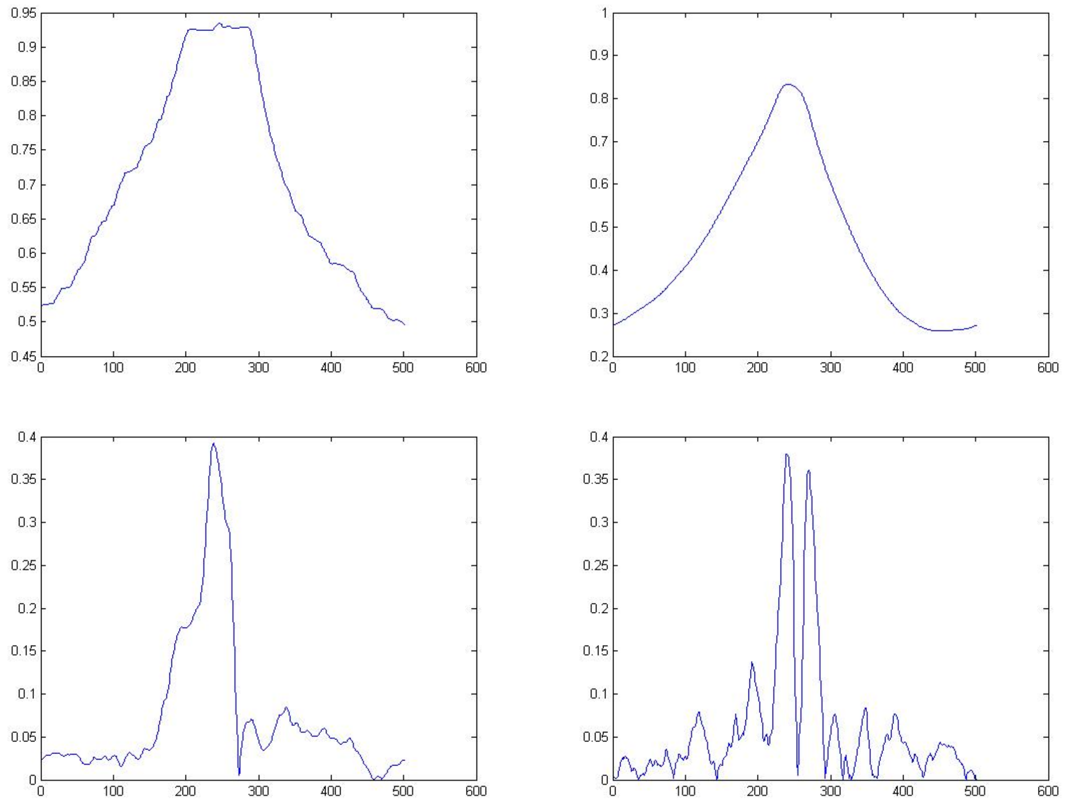


Figure 3.11: Plots of intensity for one selected point of deconvolution result with different regularization value K. Original intensity change during the z direction (top left), $K = 0.1$ (top right), $K = 0.0001$ (bottom left), $K = 0.000001$ (bottom right)

regularization parameter $K = 0.0001$. From this result, only one sharp peak appears. The side band of this curve is small enough compared to the peak value. The lower right figure is the result from regularization parameter $K = 0.000001$. Here, it still has sharp peaks but more than one. All these four results verify the truth that selecting an appropriate level of regularization can get a clear and identifiable result which is better for the reconstruction.

As we mentioned before, the z direction in the focal stack represents the frame location in the focal stack which is numbered by the depth parameter alpha. In other words, if we know the location of the refocused object in the focal stack, we also can know the alpha value for this object. After doing the deconvolution, the location of the peak value of each pixel is the location we want. So according to the equation above, we can use the alpha value to calculate the depth of the object in the real world.

3.4 Segmentation

3.4.1 Introduction

Two ways humans perceive the outside world is hearing and vision. Vision gets the most information from an image. In an image, people are often only interested in some of these items. These objects usually occupy a certain area and some features, such as gray level, outline, color, and texture are different from the background and the surrounding area. These features can be very significant but also can be very subtle, imperceptible to the human eye. Development of computer image processing technology allows people to use computers to access and process image information. Image recognition technology has been successfully applied in many fields. Among them, note recognition, license plate recognition, character recognition (OCR), and fingerprint recognition are already familiar to everyone. Image recognition can use image segmentation, whose role is to reflect the real situation, occupying different regions, with different characteristics of target separately, and form digital features. Image segmentation is a prerequisite steps of an image recognition system [28]. The quality of image segmentation directly affects the entire image processing chain and can even determine their success or failure. Therefore, segmentation is critical.

Why do we need segmentation? Because the limitations of the deconvolution process are too significant to get a good result. Deconvolution is based on the PSF to deblur the image so that we can then estimate the depth [4]. We can imagine that if the targets in one image are too close to each other, the spread of the point source will affect and be affected by each other too much. This interference will appear during the deconvolution processing. We cannot change the light source, but we can change what kind of image we will process. So we need the segmentation to separate the objects in one image to process them individually. This will reduce the effect of neighbor point sources.

Segmentation decomposes an image into several non-overlapping areas each of which has the same nature and significance. Good segmentation should have the following three characteristics:

- Segmented areas are similar in some certain features (such as gray scale, color, texture, and so on), have connectivity inside and do not have too many holes;
- The segmentation of adjacent areas is based on differences in the same property;
- Regional boundaries are clear.

Most existing image segmentation methods can satisfy the characteristics mentioned above[22]. However, if we emphasize distinguishing regions with similar properties, segmented regions are prone to produce more small holes and uneven edges. If we emphasize significant similarities of properties among areas, segmentation is much more likely to yield merged areas which have different properties. Therefore, choosing an appropriate segmentation method is required to find a reasonable compromise.

Thus, we see that the basic problem is dividing the image into a plurality of regions in the aforementioned conditions. Typically, for a monochrome image segmentation algorithm based on one characteristic of gray value processing, we consider discontinuity and similarity.

In the first category characteristic, it is assumed boundaries of these regions are completely different from each other, allowing boundary detection which is based on local discontinuities. Edge-based segmentation is the main method used in this category. Region segmentation method in the second category is based on a predefined set of criteria so that one image is divided into several similar regions.

3.4.2 Region Growing

Region growing is based on the growth of pre-defined criteria to group the pixels or sub-region into a larger area [**?**]. The basic method selects from the group of "seed" points [1] and then add pixels which are similar to predefined properties of neighboring seeds to each seed to form the growing area, such as a specific range of gray scale or color[24].

We can usually base our choice on the nature of the problem to select one or more sets of starting points. When prior information is not available, this process calculates the same characteristics at each pixel. Eventually, during the growth process, feature sets emerge that are allocated to each pixel region. If the results of these calculations show a set of values, then those characteristic of pixels similar to the center pixels of these groups can be used as a seed.

If use the connectivity properties are not used in the region-growing region, the individual descriptors will produce erroneous results. For example, consider a random arrangement of only three different pixel gray scale values. Pixels with the same gray level combination forming a "region", regardless of connectivity, will have a meaningless result.

Another problem of growing a region is the representation of termination rules[9]. When there are no pixels that satisfy the join criteria for an area, the region will stop growing. Other criteria which enhance regional capacity growing algorithm utilize candidate pixel, similarity, the shape of the growing area and so on. The utilization of such descriptors is based on the assumption that a model of expected results is at least partially available.

Let $f(x,y)$ denote an input array. A gray scale based region growing algorithm can be stated as follows,

- Find a start seed from input array $f(x,y)$ for each region as a starting point for each region needed to grow;
- From the start seed, check its neighbor pixels and compare all eight neighbor pixels to the center seed. If the gradation difference is smaller than a predetermined threshold value, they will be merged;
- From the newly merged pixel as center pixel, check a new pixel neighborhood until the region cannot be further expanded;
- Continue to scan until no additional pixels are found, which ends the whole growth process.

A key point of region growing is to select the appropriate growth Characteristic. Most of the region growing algorithms use local characteristics of the image. Growth criteria are formulated according to different principles, and the use of different criteria will affect the result of growth. The result of the basic region growing algorithm is also affect significantly by the selected start seed. Based on this, we can improve the algorithm by first setting the gray-scale threshold value to zero, which defines the pixels with the same gradation to be part of the same area. Calculate the average gray level of all the adjacent regions, and combine the adjacent region with the minimum gradation difference. Repeat the last step till a predetermined termination criterion is satisfied. This technique has limitations when dealing with an image full of gray scale details.

Here I used the region growing algorithm based on statistical properties of the gray distribution area. This algorithm considers gray distribution similarity criteria to determine the merger as a growth area.

- The image is divided into small areas that do not overlap each other

- Compare histogram of adjacent regions, to merge regions with similar features.

- Repeatedly merge each area in turn until the termination criterion is satisfied.

Here I used the smoothed-difference test, $\max |h_1(z) - h_2(z)|$, where $h_1(z)$ and $h_2(z)$ denote the histograms of two adjacent regions.

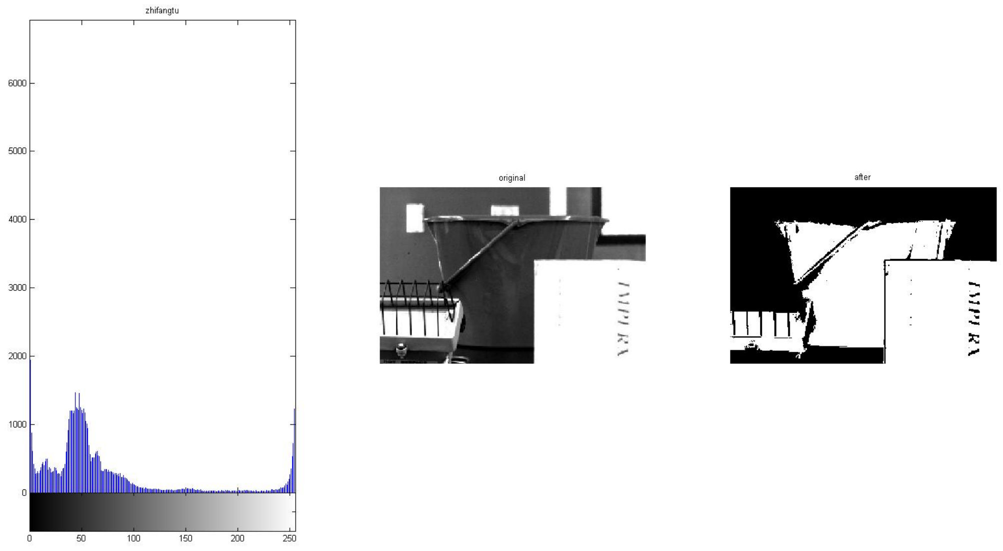


Figure 3.12: Segmentation result with Region Growing method. Left figure is the histogram of same gray scale pixels. Center image is the original image. The right figure is the result.

In Figure 3.12, the left figure shows the histogram of the experimental image, and the center is the original experimental image. The right figure is the result using the statistical–based region growing algorithm. But the principle problem of this algorithm is always the need to select the first seed. I set three seeds here to get the three target regions growing.

3.4.3 Watershed segmentation

Watershed segmentation algorithm is a method borrowed from morphology theory. This method treats an image as a topographic map of the topology, in which the pixel gray scale value corresponds to terrain height. High gray value corresponds to peaks, low gray value corresponds to valleys. Water is always flowing towards low-lying areas. Eventually, each local low-lying areas, known as an absorbent basin, will fill with water and be segregated into different absorption basins. Ridges between the suction basins are called the watershed[31]. When water flows from the watershed, the possibilities of water moving to different suction basins are equal. This idea can be applied to image segmentation to find different absorption basins and watersheds in a gray scale image. The target to be split is composed of these different absorption basin and watershed areas [24].

Watershed segmentation algorithm is an adaptive segmentation algorithm [10]. Low-lying basins experience water absorption. The height is equivalent to a threshold level. When the threshold value is increased, the water level rises. When the water rises at a watershed, the water-absorbent watershed basin eventually overflow into another basin.

The watershed threshold selection algorithm is simple and has excellent performance. It is better able to extract the object contour. However, due to the need for gradient information, the original signal and noise will cause many false local minima in the gradient figure, leading to an over-segmentation phenomenon.

We can set one minimum point belongs to a region. See another point as a drop of water, water droplets will fall to a single minimum value if these points placed on anywhere. Water at this point will flow to more than one such minimum point with equal possibility. The minimum for a specific area which satisfy condition (b) called catchment basin or watershed. The peak point which satisfies the condition (c) forming crest lines on the ground surface is called the dividing lines or watershed lines[26].

Here we set an experimental image shown below in Figure 3.13. The image used here as input for segmentation is a perspective image without shift. In other word, this input image is not a refocused image but is generated by the perspective function with zero angle. All points are selected from the center of each micro-lens and then interpolated. All this is done by the perspective function in LFIT.

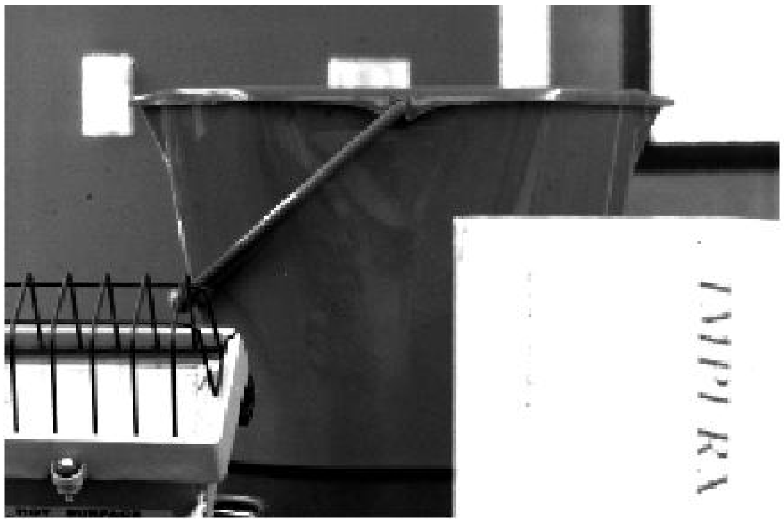


Figure 3.13: Selected experiment image

As we mentioned, the main objective of these concepts in a segmentation algorithm is to find the dividing line. We can calculate the gradient of the whole image to get the dividing line. Following is the result after the gradient calculation:

Then, we can directly use the watershed transform provided in MATLAB, $L = watershed\left(A\right)$, based on gradient modulus.

From the above Figure 3.15, we can see the defect of watershed algorithm is usually due to local noise and irregular gradients that cause excessive segmentation. A method for controlling excessive segmentation is based on the concept of a marker. A marker is a connected component that belongs to an image. The object of interest associated with the marker is known as an internal marker and a tag associated with the background is called an external marker. If we define an internal marker as: (1) the point with higher elevation surrounded by a region; (2) those points of a connected component are formed in a region; (3) all points that are connected components have same gray scale. After applying

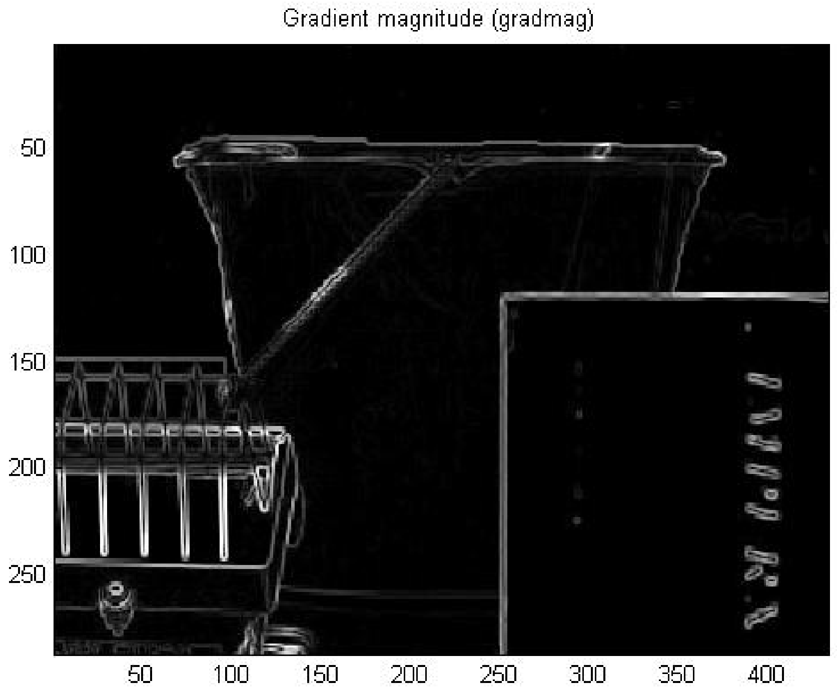


Figure 3.14: The result of gradient calculation

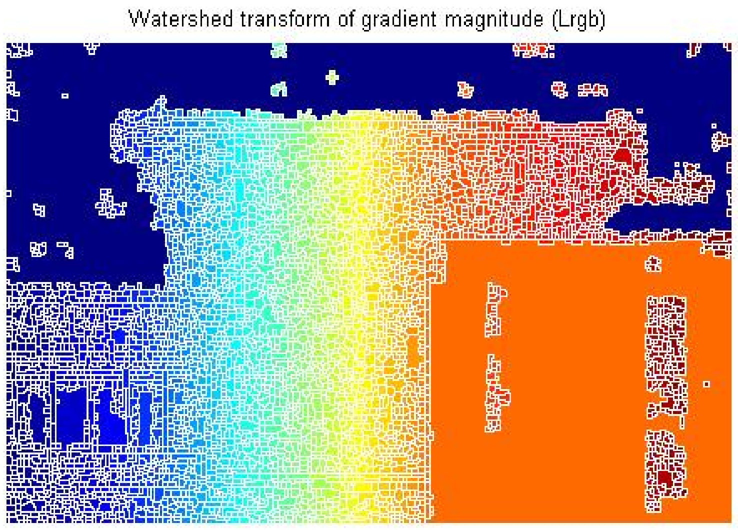


Figure 3.15: Watershed transform of gradient magnitude which is over segment

the watershed algorithm, the dividing line was defined as an external marker. The external marker effectively segments different regions, each region containing a single internal tag and part of the background. So these areas will be classified as a single object and the background. This example uses morphological reconstruction techniques to mark the foreground object, first using an open operation which can remove some small targets. The result is shown below in Figure 3.16,

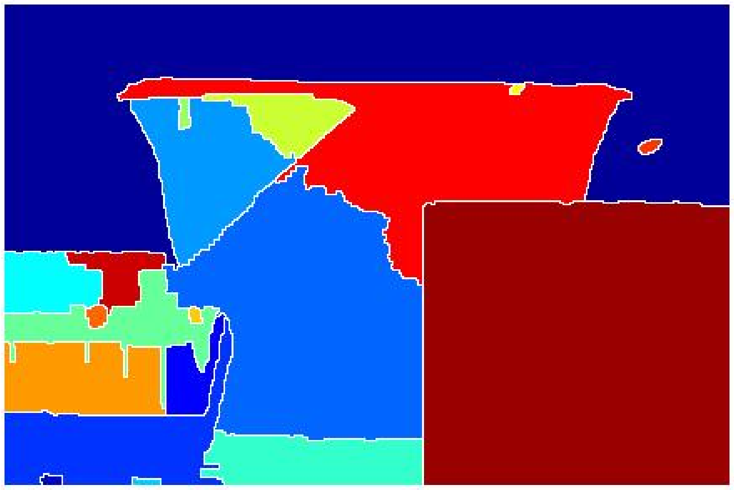


Figure 3.16: Improved result of watershed segmentation method

Chapter 4

Result of Depth Estimation

4.1 Simple Target Result

To compare the result of the algorithms mentioned above, three experiments were devised to compare the performance of three different algorithms. First, a simple one–target plenoptic image was reconstructed to compare the basic performance of three different algorithms. The first plenoptic image is shown below.

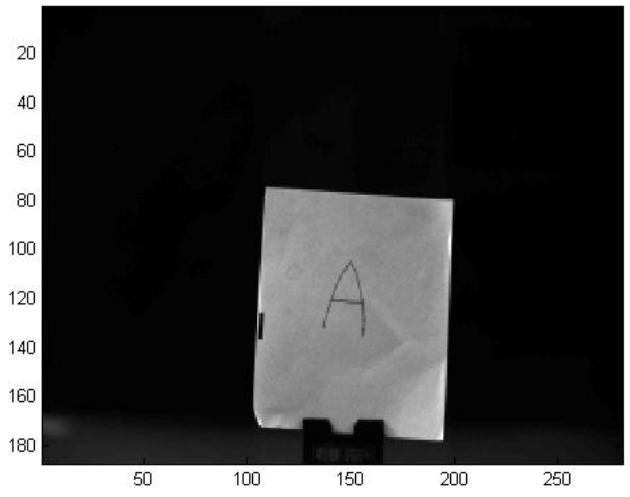


Figure 4.1: Single object refocused image example on focal plane $z = 251$

For a test case, the object was taken imaged at the focal plane at $depth = 22.50cm$, where the reference ruler height is $140mm$. The focal stack generated by LFIT consisted of $z = 501$ refocused images with $\alpha \subseteq (0.15, 1.85)$ range. Here, we use x and y coordinates to represent each refocused image in a three-dimensional focal stack. The third dimension is represented by the z coordinate, which indicates the location of each refocused image in the focal stack. Recalling the depth parameter α, $\alpha = 1.00$ indicates the target is at the nominal focal plane, $depth = 22.50cm$ in this case. According to the α range in this example, $\alpha = 1.00$ indicates that the refocused image is located at the center of the focal stack, $z = 251$ in this case.

4.1.1 Result of depth from blur algorithm

For the depth from blur algorithm, the first step is to filter the refocused image with a Laplacian, which shows the edge information in the image:

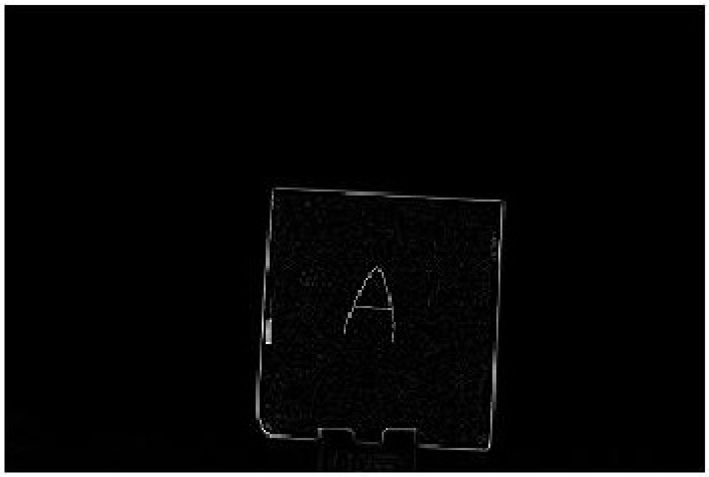
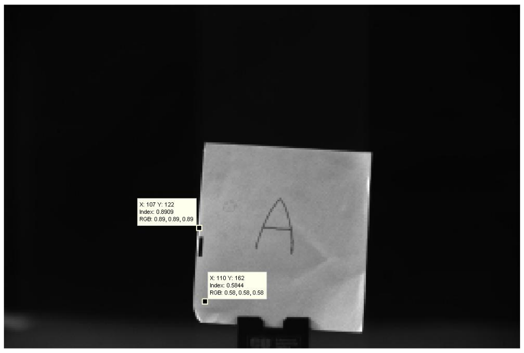


Figure 4.2: Edge information from filtering the original image with Laplacian

This algorithm only uses the edge information to calculate the depth for the whole image. Following is the depth estimation result using the depth from blur algorithm.

Next, the maximum intensity in the focal stack is found for all points in the image, and the background information is ignored. The result shows the inside points of the object still have depth information that is similar to the depth estimation of the edge points. To consider this result further, a plots of the intensity information of one inside point of the object along the z direction is shown:



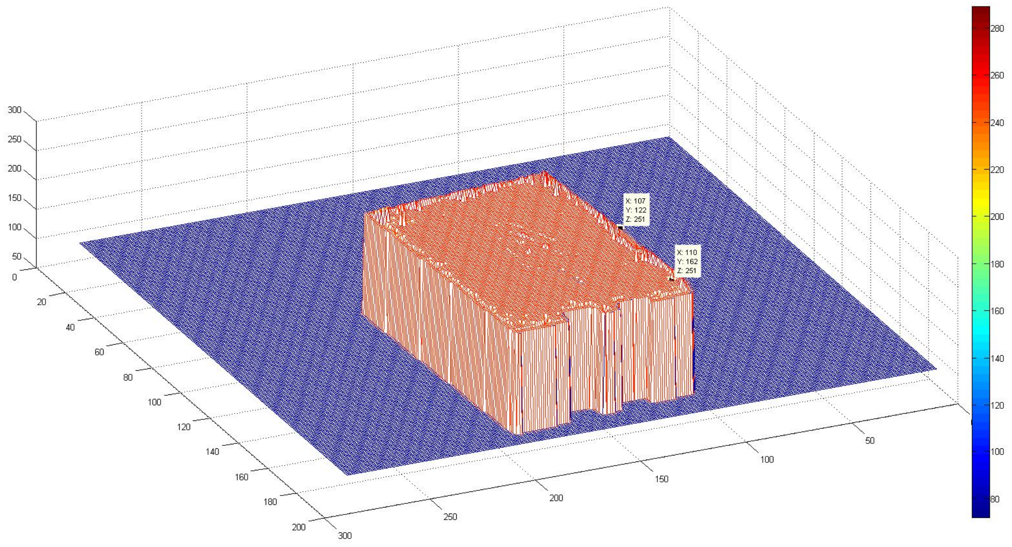


Figure 4.3: Top figure shows the original image. Bottom figure is the depth estimation result of depth from blur. The coordinate of the top point is (107,122) which is selected from an edge, and the corresponding plane in the focal stack is $z = 251$ shown as bottom figure. The coordinate of the bottom point is (110,162) which is selected to be on the object, and the corresponding plane in the focal stack is $z = 251$.

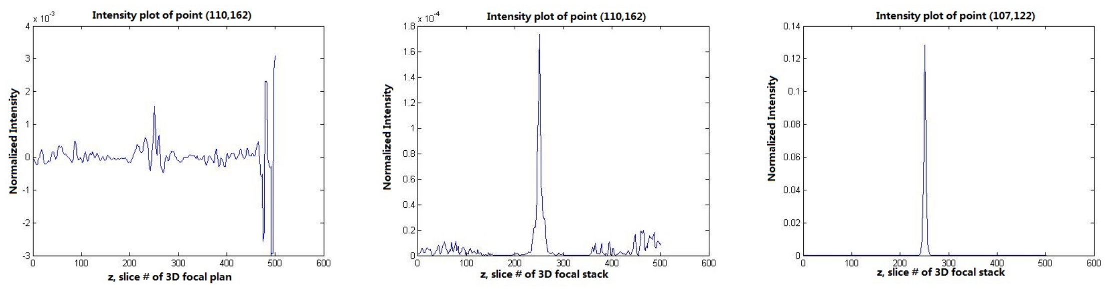


Figure 4.4: Left and middle figures are the intensity profile of selected point $(110, 162)$ with respect to depth; Right figure is intensity profile of edge point $(107, 122)$ with respect to depth.

The left and middle figure in Figure 4.4 are the intensity plots of one point selected from the smooth area of the target. The difference between these two figures is that the left one represents the intensity after filtering the original focal stack. The middle one represents the intensity after the whole depth from blur process. But the common characteristic of these two figures is that the maximum intensity is too small compared to the right figure. The right figure is the intensity plot of one edge point after the depth from blur process. Such a small intensity is very susceptible to interference by nearby points that would confuse the real depth. So the intensity values of smooth area points are not reliable for depth estimation. This behavior will be confirmed by the next experiments.

4.1.2 Result of stereo algorithm

The stereo algorithm used various viewing perspectives on the main lens to calculate the depth. The depth calculation is based on the disparity. From the above discussion, the disparity estimate is based on the characteristics of the correlation of surrounding pixels. The disparity map is generated by finding the peak of the normalized cross-correlation coefficient first and then averaging the disparity estimates together. Then the depth estimation result can be calculated from the disparity map. Figure 4.5 shows the disparity map of a simple target plenoptic image, and Figure 4.6 is the depth estimation result of the simple target.

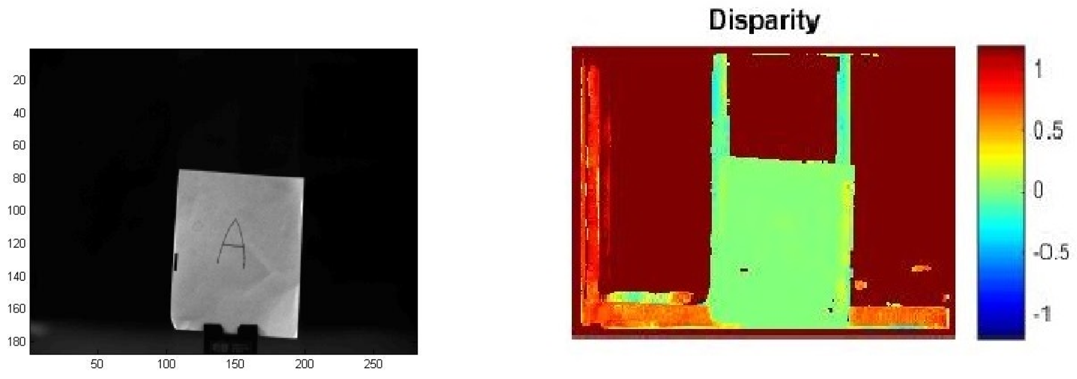


Figure 4.5: Left figure is original image. Right figure is disparity map for a simple target. In the disparity map, red indicates near and blue indicates far.

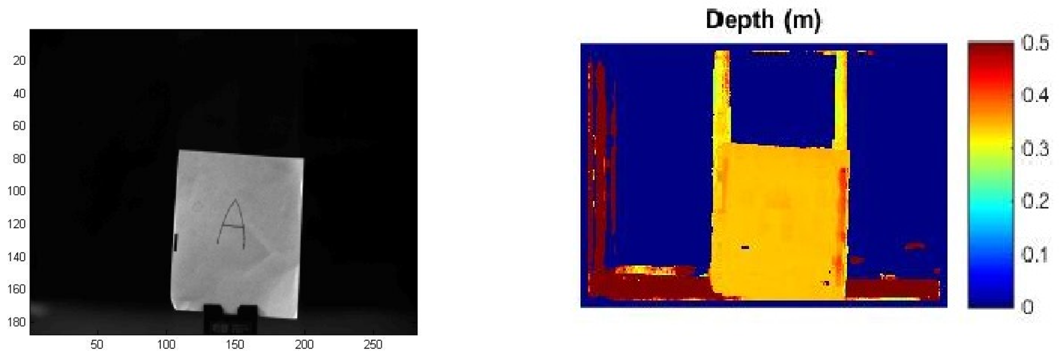


Figure 4.6: Left figure is original image. Right figure is depth estimation result for simple target.

From the depth result, we can see the stereo algorithm is very sensitive to the light source. The odd vertical lines attached to the target are due to the light source recorded by the CCD sensor coming from different angles. The background in the experimental plenoptic image is black which is dark enough to contrast with the light from the target. So when we use different perspective views of the experimental image, the “smear” of light from different perspectives is more distinguishable. This is a sensor problem, and the black background makes this problem more sensitive. Without the sensor problem, the performance of the stereo algorithm will be more accurate. This algorithm is effective because it is similar to human eyes.

4.1.3 Result of deconvolution algorithm

With the same plenoptic image, the performance of the deconvolution algorithm is examined. To explain the performance visually, three deconvolution results are obtained utilizing different regularization values K, (from left to right: $K = 0.01$, $K = 0.0001$, $K = 0.000001$).

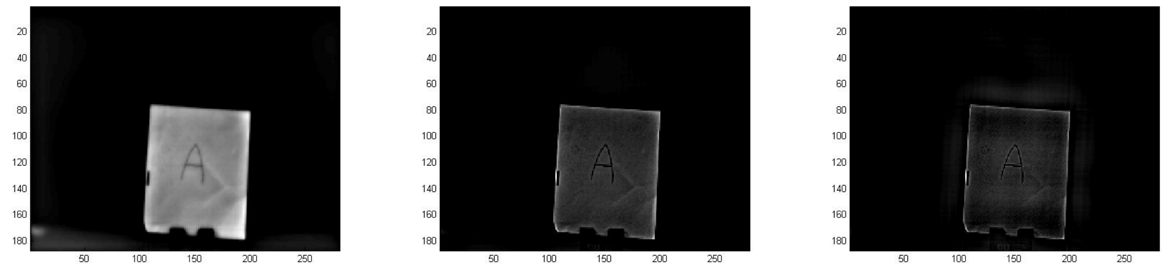


Figure 4.7: Comparison of the deconvolution utilizing regularization values $K = 0.01$(left), $K = 0.0001$(center), and $K = 0.000001$(right)

The results become dark when the regularization values K decrease. When K is large, the deconvolution result does not differ too much from the original image at the same focal plane. When K is small, artifacts appear around the edge as shown in the right figure of Figure 4.7. To analyze the change in the z direction of the focal stack, the following figures show y-z slices of the focal stack after processing the deconvolution utilizing different regularization values K.

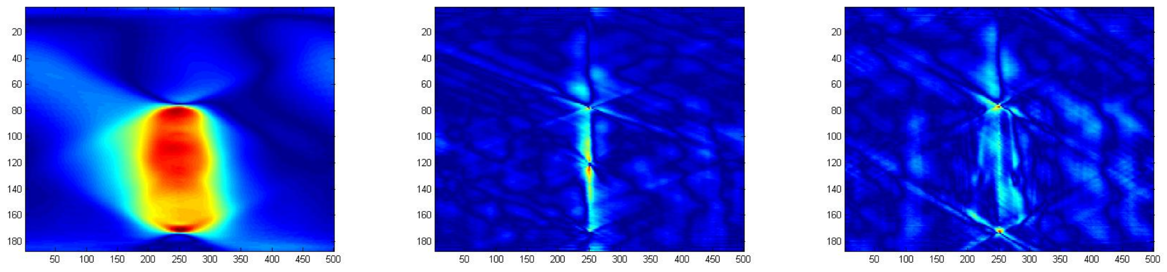


Figure 4.8: Comparison of y-z plane slice of the object focal stack at different points with different regularization values $K = 0.01$(left), $K = 0.0001$(center), and $K = 0.000001$(right)

In the figures above, the color represents different intensity levels. The bright color and red represent higher intensity, while the dark color and blue represent lower intensity. We can see the deconvolution results with $K = 0.0001$ can concentrate energy very well, represented in the middle figure. The intensity distribution in the left figure utilizing $K = 0.01$ is still spread out. And when utilizing very small regularization value $K = 0.000001$, artifacts remain in the result. By calculating the maximum value of intensity and its location, we can build the depth estimation surface by finding the maximum intensity in the focal stack along the z direction as shown in Figure 4.9 and Figure 4.10.

In this depth estimation result, five points (x, y, z) were selected from different parts of the image. Here, z coordinate represents the plane number in the focal stack which relates to the depth parameter α. We set 501 steps in the focal stack, which divides the depth to 501 slices. The object in the figure is in focus, which means the plane of this object in the focal stack should be at the center ($z = 251$). We can see a large flat region with the same depth $z = 251$. But it still has some random spikes around the corner. This is because the original has some undefined noise and the algorithm cannot eliminate all of the noise during the process because the spread energy is not fully localized to the original light source location.

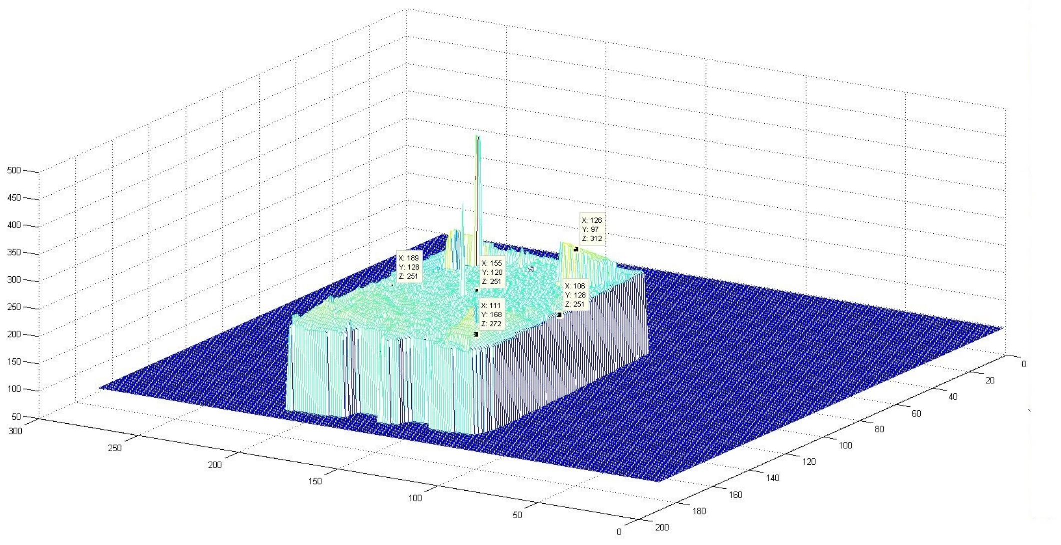


Figure 4.9: Direct depth estimation result from deconvolution algorithm, side view

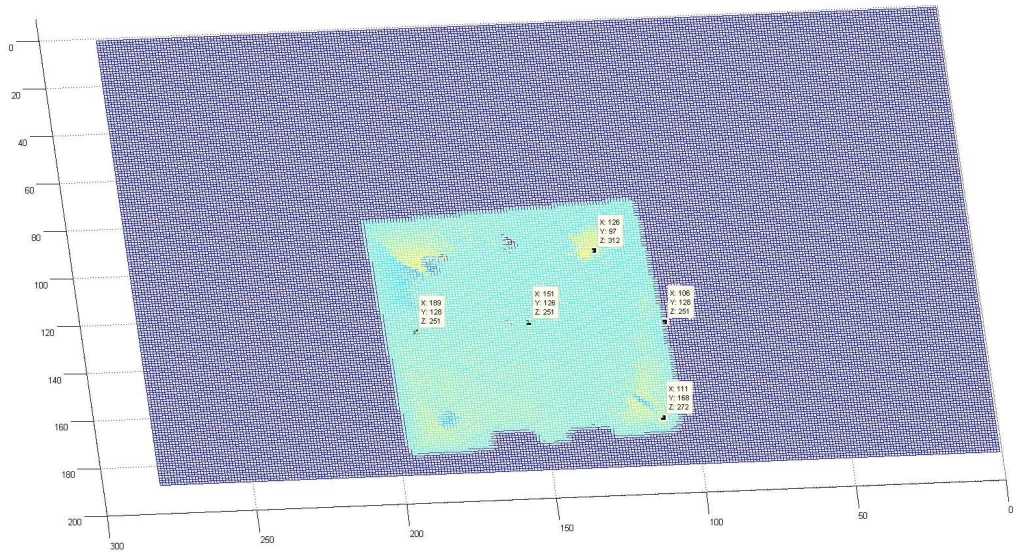


Figure 4.10: Direct depth estimation result from deconvolution algorithm, top view. In this figure, the dark blue represents the background which is far away from the camera lens. The light blue represent the object which is near to the camera lens.

But from this result, it can be seen that with a smooth area in the image the deconvolution can still get an accurate depth.

4.2 Multi-target Result

For multiple targets in a single image, we can test performance when dealing with the point spread effect from nearby targets. We chose two smooth targets with different depths shown below.

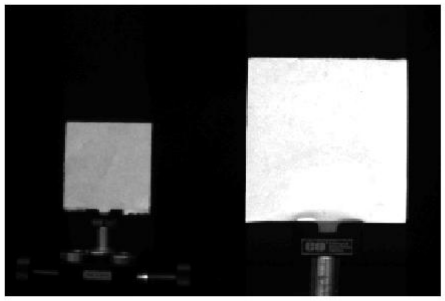
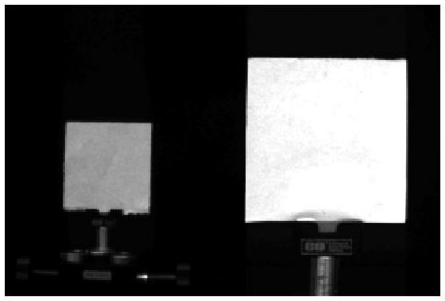


Figure 4.11: Sample figure of Multiple targets in a single image

In this test case, the focal length for the main lens is 50mm. The camera CCD is $36 \times 24mm$ and 4904×3280 pixels, so we can calculate that the magnification is 0.1863. The distance from the camera to the right side target is 280mm and to the left side target is 466mm. Using the refocus computation formula, the depth parameters for these two targets are $\alpha_{right} = 1$ and $\alpha_{left} = 0.95$. In Section 2.3.5, it was already discussed that the farther away from the camera, the smaller the depth parameter α is. The focal stack generated by LFIT consists of $z = 501$ refocused images with $\alpha \subseteq (0.35, 1.65)$. According to the α range defined here, the locations of the two objects in the focal stack are $z_{right} = 251$ and $z_{left} = 240$.

4.2.1 Result of depth from blur algorithm

Figure 4.11 is the result using the depth from blur algorithm. We can see a lot of spikes on the targets. From the data shown in the figure, we can see the edges still have a better estimation result than the center points.



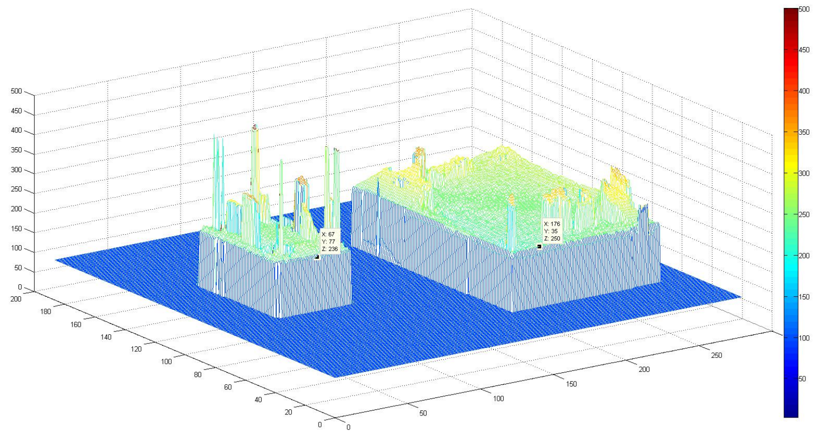
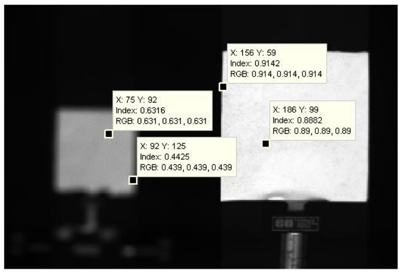


Figure 4.12: Depth estimation result by using depth from blur algorithm

From Figure 4.12, most of the area on the objects has a green color, which represents the same depth (here, green is equal to $z = 251$). We plot an intensity curve again to examine the conclusion we drew above. We selected four points, (75, 92), (92,125), (156, 59), (186, 99), from the test image. Two points were selected from the edge of the objects, (92,125) and (156, 59). The other two points were selected from smooth areas on the objects.

From the following Figure 4.13, we can see the intensity is still very small, which means this intensity is not reliable. From the single -target experiment, we mentioned that the estimates in the smooth area on objects are unreliable due to the too small intensity value of these points. We did not consider the information about inside points reliable from utilizing the depth from blur algorithm. Because these points have really small intensity values, they were too easily affected by nearby points.



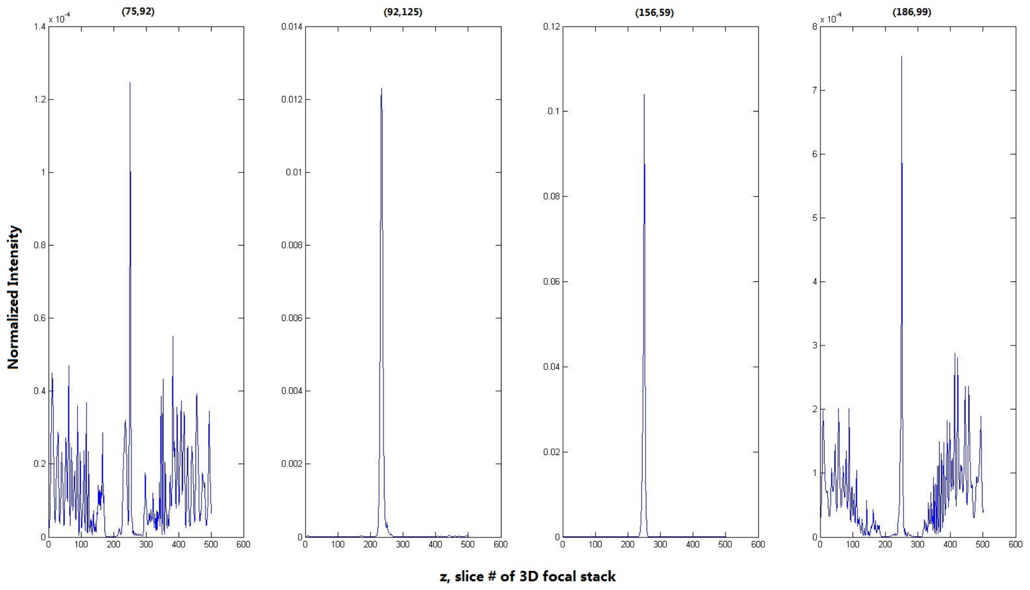


Figure 4.13: Intensity profile of selected points with respect to depth

4.2.2 Result of stereo algorithm

For the second plenoptic image experiment, the disparity map is shown in Figure 4.14

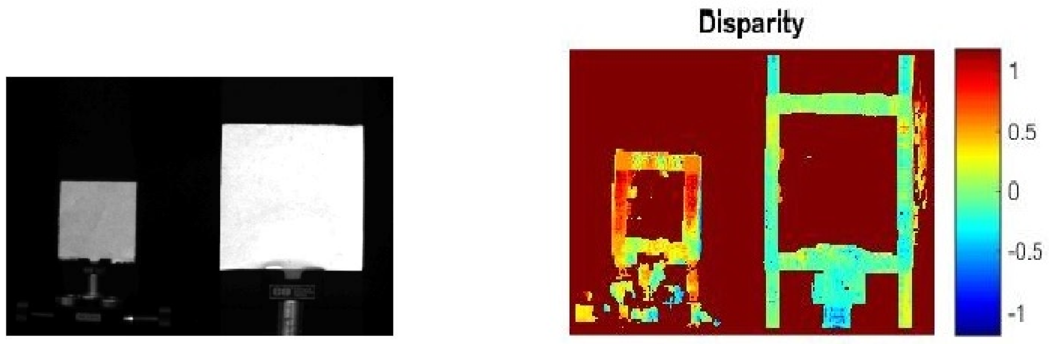
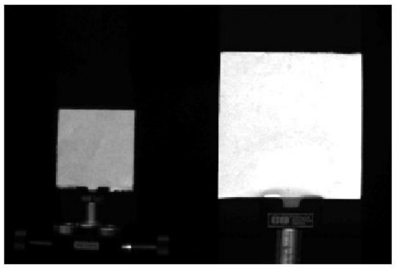


Figure 4.14: Left figure is original image. Right figure is disparity map for multi targets. In the disparity map, red indicates near and blue indicates far

For this experimental plenoptic image, the smooth areas are missed in the disparity map. This is because the generation of disparity map is based on the correlation coefficient between the search window and the template window. When the window is located at the center of one target of the experimental plenoptic image, the perspective images from different angles are all same. Therefore, it cannot determine whether the target whether is shifted or not when the target is large and monochrome. In other words, the edge points can be estimated in depth, but it cannot work on the center points.

Also, the anomalous vertical lines appear around the right object. From the raw image, we can see the right object is in front and much brighter than the left one. This also manifests the sensor problems that occurred in the first result. The bright object may cause the light to smear in different perspective views due to the correlation method picking up whatever pattern is in the smear and treating it as if it were a real feature.



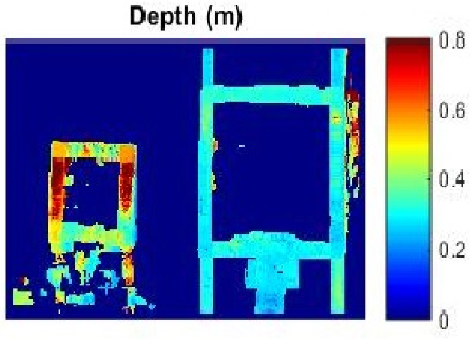


Figure 4.15: Left figure is original image. Right figure is depth estimation result for multiple targets.

4.2.3 Result of deconvolution algorithm

For the deconvolution algorithm, the deconvolution step is essential to the whole process. After the deconvolution step, we plot the result of depth estimation by finding the maximum value in the z direction. The results are shown in the following figure. From the color in the figure, we can tell the two objects are not too far away from each other, but in fact these two objects should have enough distance between them to distinguish the depth. From the top view of the result, there are many error points. From the original plenoptic image, we know the large object is closer to the camera lens than the small object, so the large objects are brighter than the small ones. Except for the noise, most of the errors may comes from energy bleeding from the other object. In other words, we need to consider whether the effect of the point spread comes from the points of nearby objects.

To verify this hypothesis, we separated the objects and performed the deconvolution separately. The following figure is a y-z slice of the right target in the original plenoptic image. The left image in the figure is the y-z slice of the focal stack before deconvolution, and the right one is the y-z slice after deconvolution.

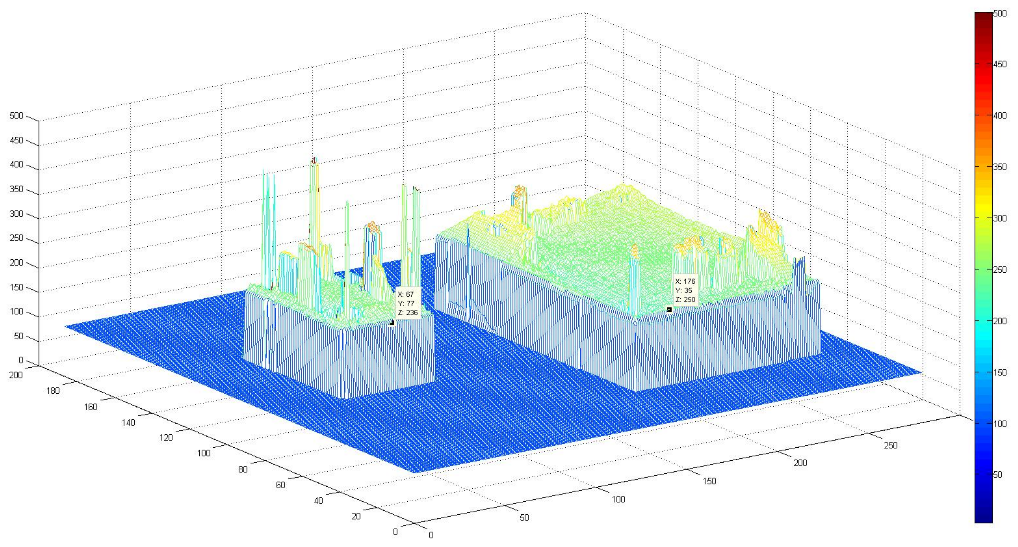


Figure 4.16: Depth estimation result by using deconvolution, side view

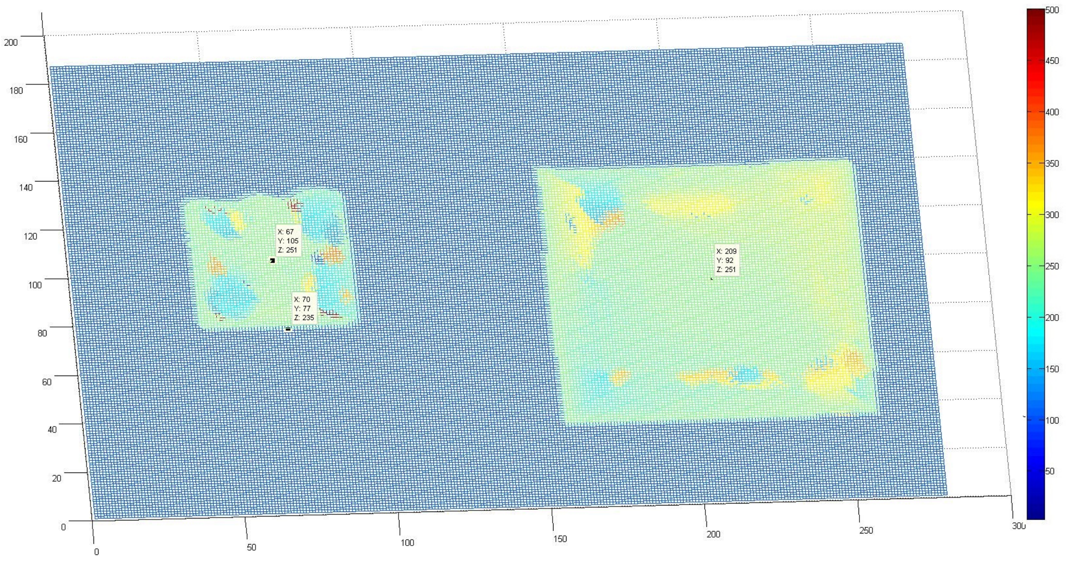


Figure 4.17: Depth estimation result by using deconvolution, top view

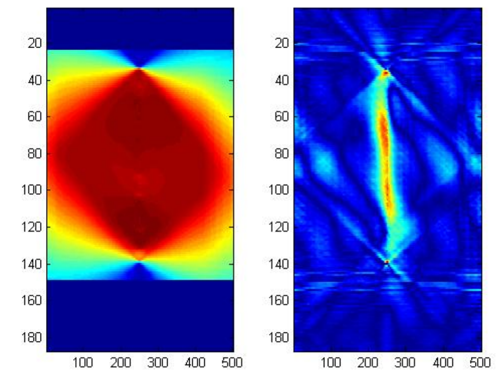


Figure 4.18: y-z slice of large object in focal stack after segmentation. Left image shows the y-z slice of the large object in the original focal stack; right figure shows the y-z slice after deconvolving separately.

We can see the performance of deconvolution step is reasonable based on the deconvolution theory. Based on this result, we can estimate the depth by using maximum values as shown in Figure 4.19.,

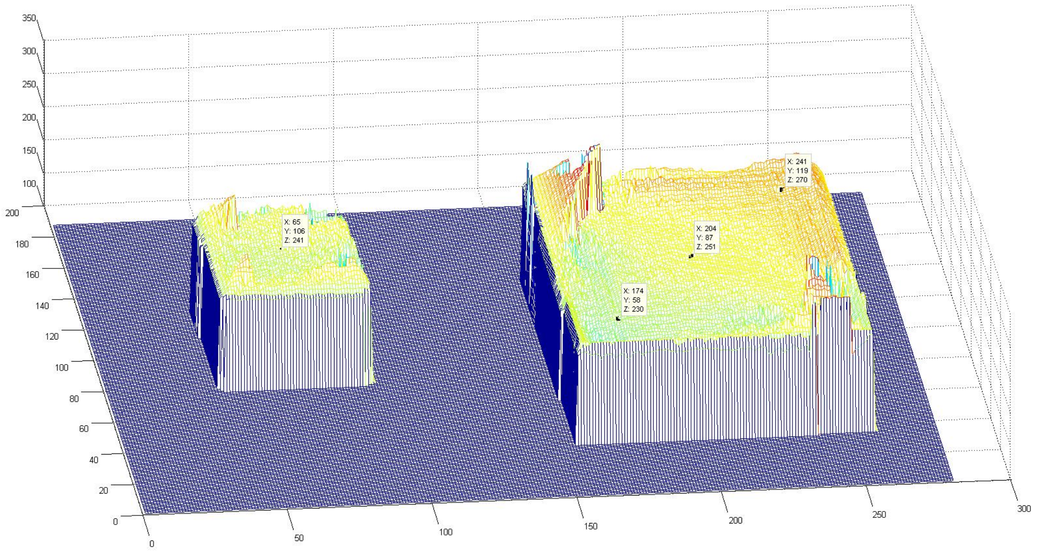


Figure 4.19: Depth estimation using deconvolution algorithm after segmentation, side view

From the two results above, the z coordinates for each center point are ($z_{left} = 241$) and ($z_{right} = 251$). Most points have a correct estimation, and the interference from large

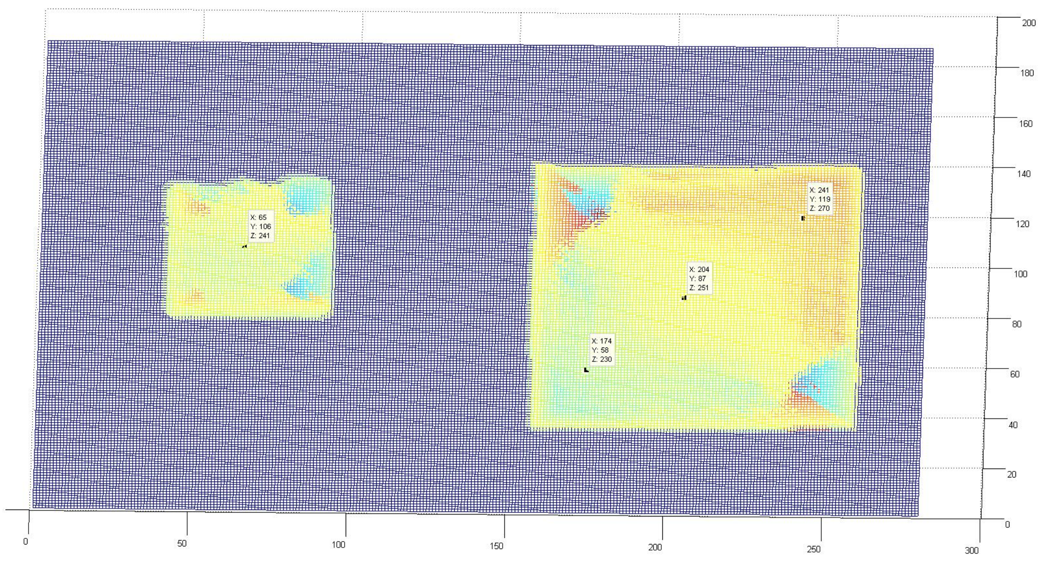


Figure 4.20: Depth estimation using deconvolution algorithm after segmentation, top view

and bright objects to small dark objects is reduced. From the top view of the result, there are still some errors in both targets. However, looking into the detailed data, we can find a regular pattern for this estimation, that is,

$$z_{center} = z_{correct} = \frac{z_{left} + z_{right}}{2}$$

This means the energy shift is symmetric after the deconvolution. Because we use the segmentation which separates each subject to process individually, the energy will shift inside of a single target. This process prevent the energy of one target from shifting to other objects to influence the depth estimation of other objects, but this also leads to the energy which belongs to each object being reconstructed with some level of error. In addition, this image still has noise affecting the direction of the energy shift. Due to this pattern, we can average the estimate results by averaging the results around the center which are based on the formula above to recalculate the distance for each point. The result is shown below.

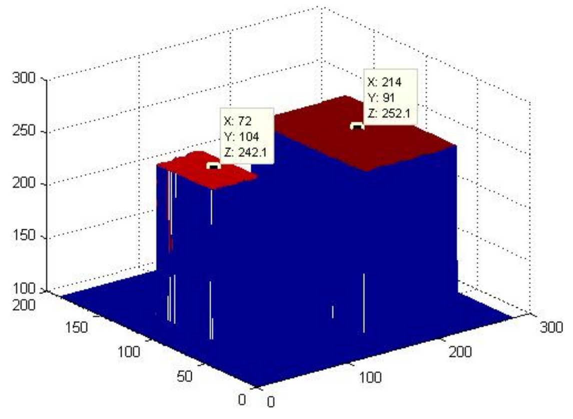


Figure 4.21: Averaging distance estimate result for deconvolution algorithm. The correct distance estimate result, for small object (left) is $z = 242.1$ and large object (right) is $z = 252.1$.

4.3 Large Target Result

The most difficult part of depth estimation is that the experimental image has a large and smooth area. When the algorithms calculate the depth, they all need to reveal spatial and depth changes in the image. If the experimental image is smooth, it is hard to find spatial differences and hard to locate the target in the depth dimension. We can imagine that it is hard to tell where the wall is if an image only records a whole white smooth wall. When we deal with this kind of image, edge points are very important. No matter how large the smooth area is in the image, the depth from blur algorithm still can estimate the depth for edge points if there is an edge in the image. Following figure is the experiment image we used.

This test contained three objects with different focal depth. The raw image shows the left object is near the camera, the center object is in focus and the right one is farther away. As we calculated above, the corresponding slice numbers in the focal stack are $(279, 251, 235)$ (from left to right). These correspond to $z_{left} = 279$, $z_{center} = 251$, $z_{right} = 235$.

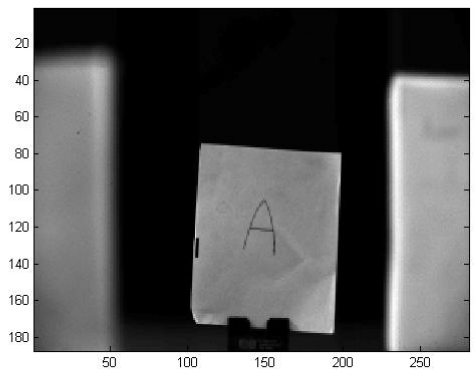


Figure 4.22: Experimental plenoptic image with large and smooth object

4.3.1 Result of depth from blur algorithm

For the depth from blur algorithm, edges still can be used here and the results are shown below. From the data cursor shown in Figure 4.23, we can see the correct depth can be calculated for the edge point, but center points are full of errors.

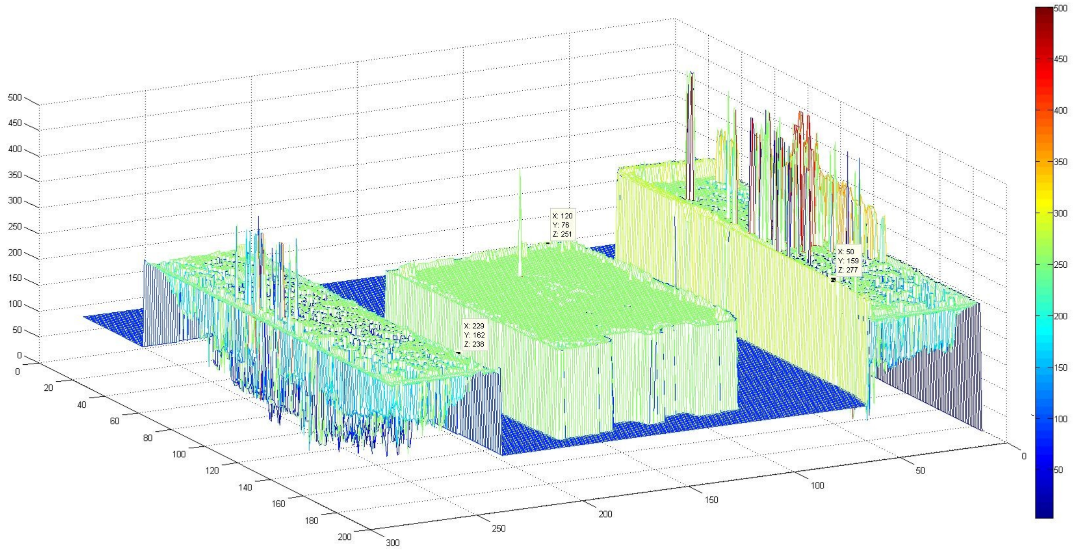


Figure 4.23: Distance estimate result of depth from blur image

4.3.2 Result of stereo algorithm

For the stereo algorithm, the window size determines the results of the depth estimation of a large target. Compared to the second experiment which contains two different size objects, the experimental image used here is three large objects. If we change to a large window, the result is similar to the result of simple target. The following figure shows the result of disparity map.

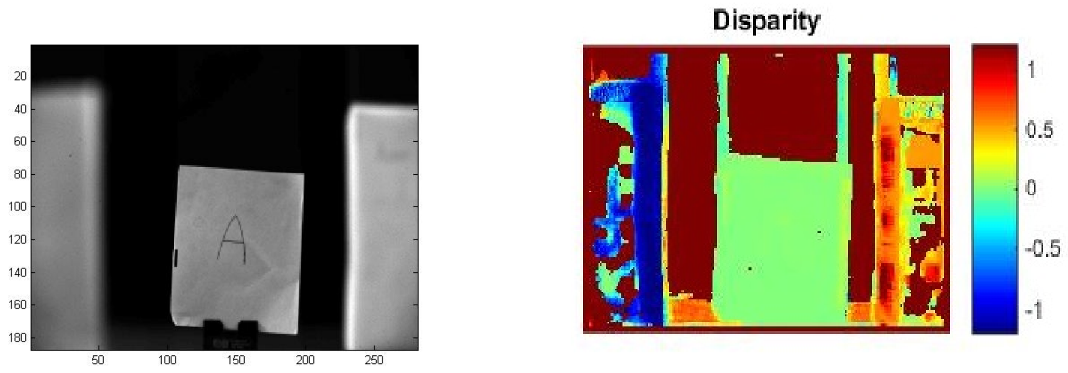


Figure 4.24: Left figure is original image. Right figure is disparity map for large target. In the disparity map, red indicates near and blue indicates far.

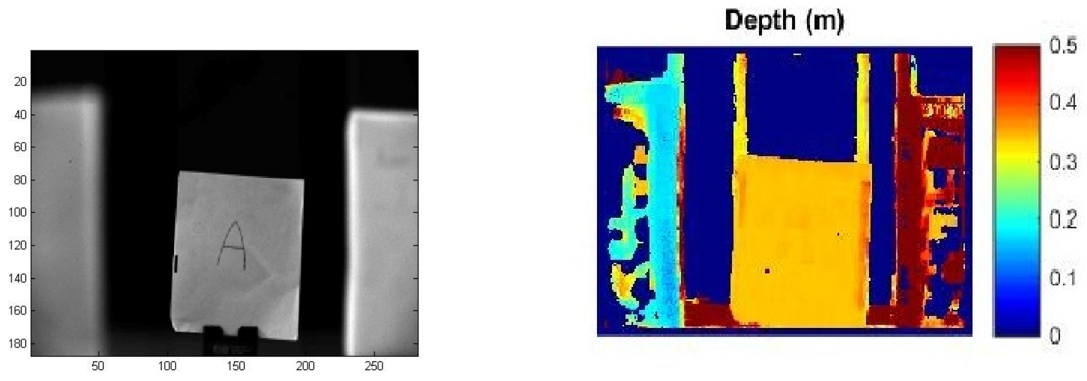


Figure 4.25: Left figure is original image. Right figure is depth estimation result for large target.

The left object and the right object in the image have unexpected results here. From the raw image, we know these two objects are two smooth surfaces. The perspective views of the left edge of the left object and the right edge of the right object are always the same white color. This means there is not enough angle information for this edge from the perspective view to compute the correlation coefficient, which certainly will affect the result of the disparity map and the depth estimation result. For the large or smooth object, the stereo algorithm cannot provide a reliable result. However, this does not mean the stereo algorithm is not useful. The stereo algorithm is a powerful tool for dealing with a more complex plenoptic image. For a complex image, there is more difference in perspective views and therefore better performance in the results. This also illustrates that the practical applications of stereo algorithms are better than other algorithms. The results obtained by the stereo algorithm are more reliable because this algorithm appears to be more robust under a variety of image conditions than the depth from blur algorithm and the deconvolution algorithm.

4.3.3 Result of deconvolution algorithm

Now we consider the result of the deconvolution algorithm. We can see the result is smooth and all depth estimations are correct. This result tests the ability of deconvolution when dealing with a smooth area. Deconvolution is able to estimate depth in smooth regions because it does not rely on the edge information but also on the center information. The deconvolution algorithm can put the energy which spreads out back to the original location. The deconvolution algorithm performs well when dealing with smooth areas. To see the performance of deconvolution, we first process the deconvolution algorithm using the whole image.

The same problem appears when dealing with multiple objects. In Figure 4.26 and Figure 4.27, the center points show the correct distance estimation for the whole target. From the center object, we can see the error spikes that appear. To test the hypothesis made

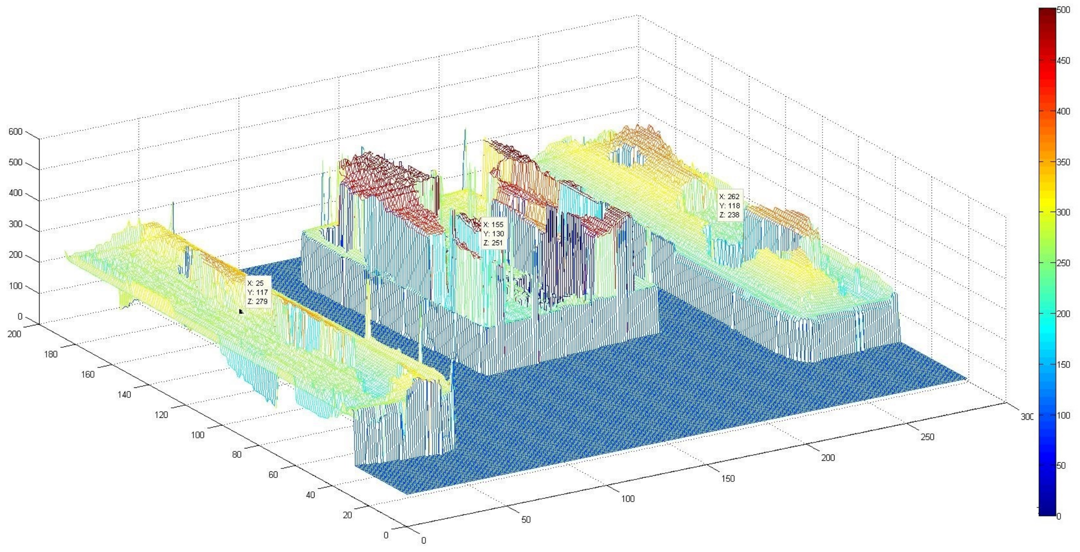


Figure 4.26: Distance estimate result of deconvolution algorithm without segmentation, side view

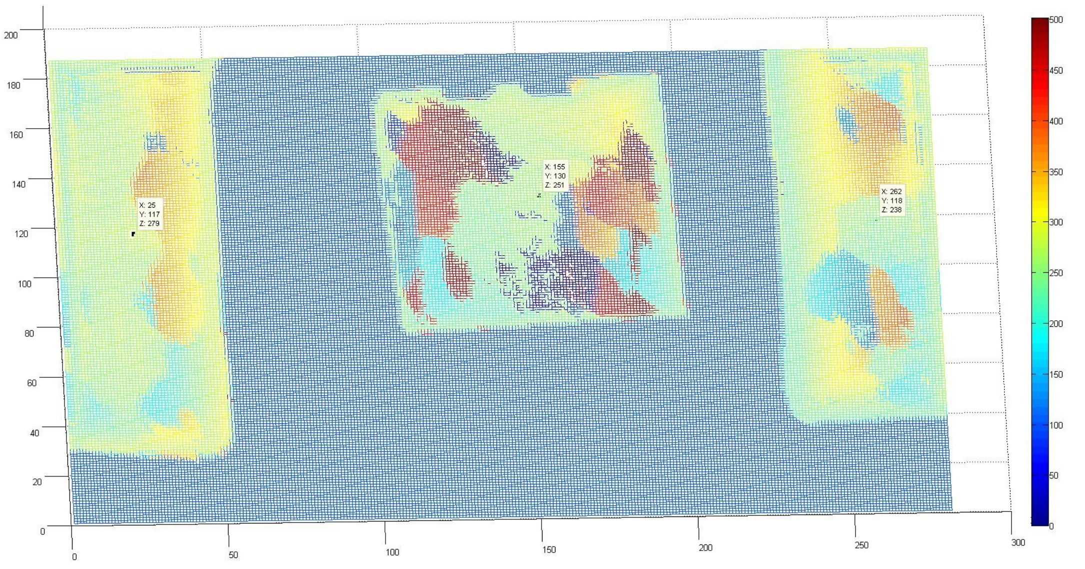


Figure 4.27: Distance estimate result of deconvolution algorithm without segmentation, top view

in the last section, we used segmentation before deconvolution to re-calculate the distance for this image.

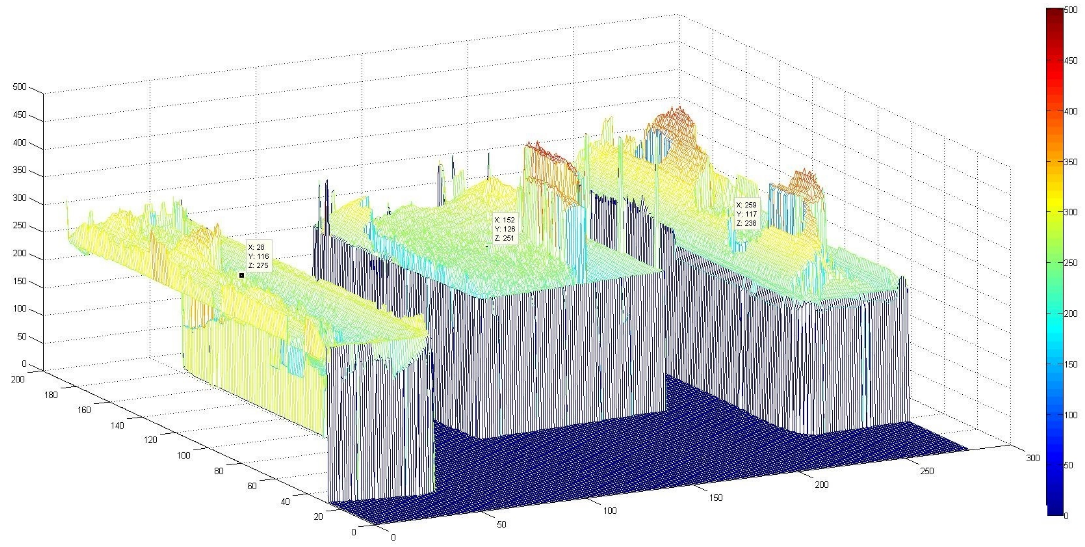


Figure 4.28: Distance estimate of deconvolution algorithm with segmentation, side view

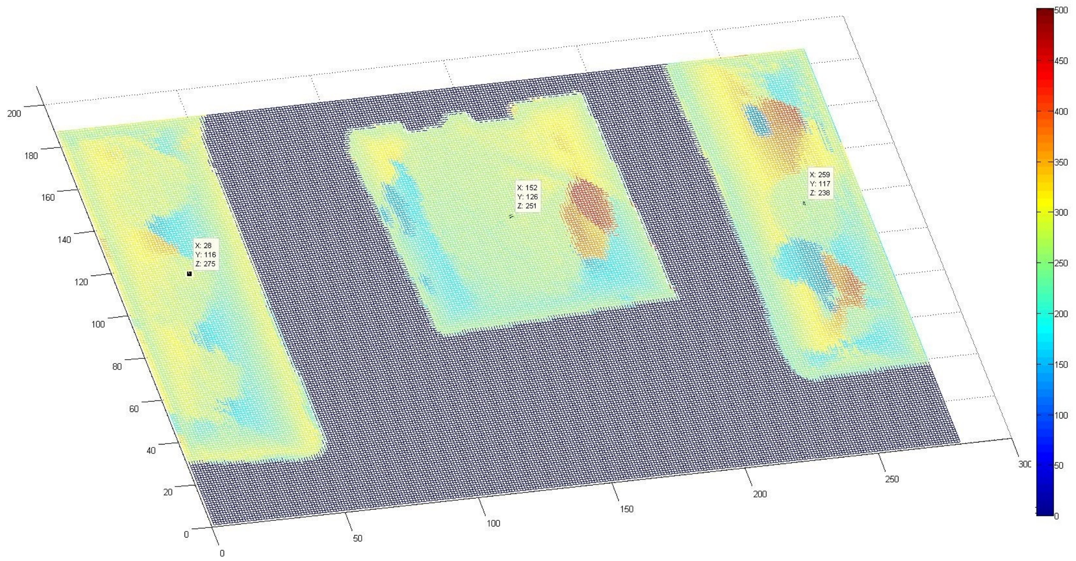


Figure 4.29: Distance estimate of deconvolution algorithm with segmentation, top view

We can see a significant change in the center object in Figure 4.28 and Figure 4.29. The center object is much smoother. For the two side objects, there is not a significant difference. This is because these two objects are too close to the edge, and the sharp boundary is not suited for the point spread function. The segmentation prevents the energy from being lost during the deconvolution, so we can average the distance around the center to regenerate the distance result.

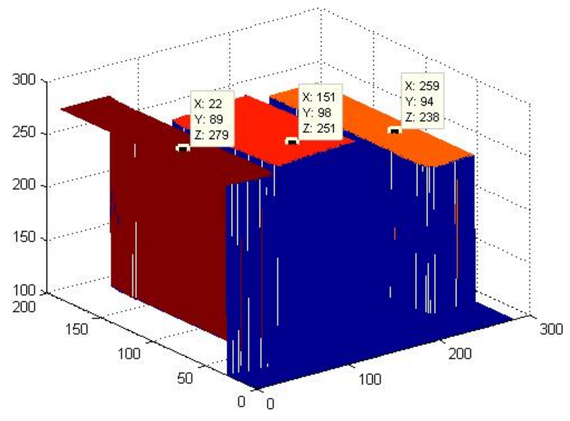


Figure 4.30: Averaging result of distance estimate of deconvolution algorithm

The noise from the sensor system is unpredictable. We took five raw images with the same scene and same settings of the three large targets from the experimental scene. After processing, we plotted the three depth estimation results of all five experiments, and the results are shown below in Figure 4.31. The five colors represent five image experiments and three depth estimations for each. The red line is an ideal result where the real depth equals the estimation result, which also corresponds to the result of one of the experiments. The largest percentage error here is below 5%. The results are all different from each other, which demonstrates that the noise from the sensor is random and unpredictable. This also increases the difficulty of estimating the correct depth. If we can filter out the noise to a degree, we can get a better result.

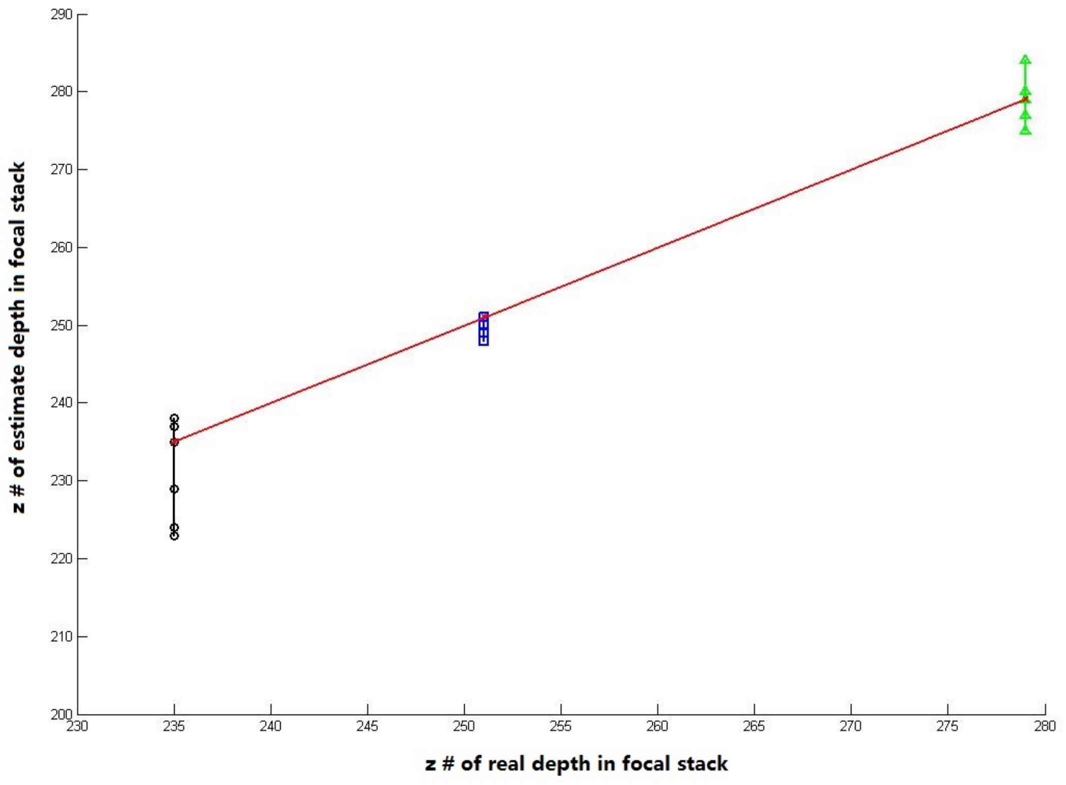


Figure 4.31: Performance of deconvolution algorithm with random noise

Chapter 5

Conclusions and Future Work

In this study, a new deconvolution-based algorithm has been developed to estimate the depth. This idea mainly comes from the properties of the light field camera, which can contain four-dimensional information. The most important feature of this algorithm is that it only needs one image to implement the depth estimation. With a light field camera, it can not only generate different perspective views of the same scene, but also can be refocused at different depths of objects in one plenoptic image.

The deconvolution algorithm is a new way to estimate the depth from a plenoptic image. From the above discussion, we can see the performance of different algorithms. To emphasize the comparison, the deconvolution algorithm can get a better depth estimation result than the other two algorithms. The deconvolution algorithm only needs one view of the scene, making the computations much easier than other algorithms. The limitation of deconvolution is still very obvious. There still are many errors appearing in the results. When the target is dark and near a bright target, it is very susceptible to interference by the point spread from the bright target or the noise in the optical system, which negatively affects the results.

Further work is required to find a way to reduce or even eliminate the effect from nearby light sources. The deconvolution algorithm provides a different approach to depth estimation and a way to exploit the potential of light ray tracing. The method involves a complicated study due to the various sources of error, including noise. Further work should address noise filtering.

Bibliography

[1] R. Adams and L. Bischof. Seeded region growing. *IEEE Trans. Pattern Anal. Machine Intell*, pages 641–647.

[2] E.H. Adelson and J.Y.A. Wang. Single lens stereo with a plenoptic camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):99106.

[3] Frdo Durand William T. Freema Anat Levin, Rob Fergus. Deconvolution using natural image priors. *Massachusetts Institute of Technology, Computer Science and Artificial Intelligence Laboratory*, 2009.

[4] Lauren Barghout and Lawrence W. Lee. Perceptual information processing system. *Paravue Inc. U.S. Patent Application*, 2003.

[5] T. E. Bishop and P. Favaro. The light field camera: Extended depth of field, aliasing, and superresolution. *IEEE Trans. Pattern Anal. Mach. Intell*, 34.

[6] P. Favaro and S. Soatto. Learning shape from defocus. *Computer Vision ECCV Lecture Notes in Computer Science*, pages 823–824.

[7] Intwala C. Babakan S. Georgiev, T. and A. Lumsdaine. Unified frequency domain analysis of light-field cameras. *ECCV*, 2008.

[8] M. Schlosser J. Jachalsky and D. Gandolph. Confidence evaluation for robust, fast-converging disparity map refinement. *IEEE International Conference on Multimedia and Expo (ICME)*, page 13991404.

[9] Mathurin Body Mohand-Said Hacid. Jianping Fan, Guihua Zeng. Seeded region growing: an extensive and comparative study. *Pattern Recognition Letters*, 26.

[10] P. Soille L. Vincent. Watersheds in digital spaces: an efficient algorithm based on immersion simulations,. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 13.

[11] M. Levoy and P. Hanrahan. Light field rendering. *Proc. 23rd Annu. Conf. Comput. Graph. Interact. Tech. - SIGGRAPH 96*, page 3142, 1996.

[12] Zhang Z. Levoy, M. and I McDowell. Recording and controlling the 4d light field in a microscope using microlens arrays. *Journal of Microscopy*, 235(2):144,162.

[13] H.-Y. Lin and C.-H. Chang. Depth recovery from motion and defocus blur. *Image Analysis and Recognition*, 4142.

[14] G. Lippmann. preuves rversibles. photographies intgrales. *Comptes-Rendus Acad.des Sci*, page 446451, 1908.

[15] S. Yang N. Cohen A. Andalman K. Deisseroth M. Broxton, L. Grosenick and M Levoy. Wave optics theory and 3-d deconvolution for the light field microscope. *Opt. Express*, 21.

[16] Stefano Mattoccia. Stereo vision: Algorithms and applications. *University of Bologna*, 2013.

[17] M.levoy. Light fields and computational imaging. *Computer (Long. Beach. Calif)*, 39:44–55, 2006.1.

[18] R Ng. Fourier slice photography. *[ACM SIGGRAPH 2005 Papers], SIGGRAPH 05*, page 735744.

[19] R. Ng. Digital light field photography. page 1203, 2006.

[20] S. J. Reeves P. Anglin and B. S. Thurow. Characterization of plenoptic imaging systems and efficient volumetric estimation from plenoptic data. 2014.

[21] S. J. Reeves P. Anglin and B. S. Thurow. Direct fft-based volumetric reconstruction from plenoptic data. *IEEE Transactions on Image Processing*, 2015.

[22] Chenyang; Prince Jerry L. Pham, Dzung L.; Xu. Current methods in medical image segmentation. *Annual Review of Biomedical Engineering*, page 315337.

[23] G. Duval M. Horowitz R. Ng, M. Levoy and P. Hanrahan. Light field photography with a hand-held plenoptic camera. *Informational*, page 111, 2005.

[24] Richard E. Woods Rafael C. Gonzalez. *Digital Image Processing Third edition*. 2008.

[25] R. A. Raynor. Rangefinding with a plenoptic camera. *Air Force Institute of Technology*, 2014.

[26] J.B.T.M. Roerdink and A. Meijster. The watershed transform: definitions, algorithms, and parallelization strategies. *Fundamenta Informaticae*, 14.

[27] Jonathan Shade. Approximating the plenoptic function. *UW CSE Technical Report*, 2002.

[28] Linda G. Shapiro and George C. Stockman. Computer vision. *Air Force Institute of Technology*.

[29] Ronneberger O. Nitschke R. Driever W. Temerinac-Ott, M. and H. Burkhardt. Spatially-variant lucy-richardson deconvolution for multiview fusion of microscopical 3d images. *Biomedical Imaging: From Nano to Macro, 2011 IEEE International Symposium on*, page 899 904.

[30] C. Thomason. Determination of the image distance in a plenoptic camera. *Auburn University*, 2014.

[31] Luc Vincent and Pierre Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13.

[32] L. H. Schaefer W. Wallace and J. R. Swedlow. A working persons guide to deconvolution in light microscopy. *Biotechniques*, 31.

[33] Filip Sroubek Xiang Zhu1 and Peyman Milanfa. Deconvolving psfs for a better motion deblurring using multiple images. *ECCV*, 2012.