Calibration of a plenoptic camera for use in three-dimensional particle tracking

by

Rachel L. Bierasinski

Submitted in Partial Fulfillment
of the
Requirements for the Degree

Master of Science

Supervised by

Jonathan D. Ellis and Douglas H. Kelley

Department of Mechanical Engineering
Arts, Sciences and Engineering
Edmund A. Hajim School of Engineering and Applied Sciences

University of Rochester
Rochester, New York

2014

Biographical Sketch

Rachel L. Bierasinski received a Bachelor of Science degree with distinction in Mechanical Engineering at the University of Rochester in 2013. During her undergraduate career, Rachel studied for a semester abroad in Berlin, Germany at Humboldt University and the Technische Universität Berlin. Rachel was also honored as a Hajim Scholar in 2012. After graduation she worked as a Design Engineer Intern at Amphenol RF in Danbury, Connecticut from June to August of 2013. Immediately following her summer internship, she began her graduate studies as part of the Graduate Engineering at Rochester (GEAR) Program, an accelerated combined degree program. She conducted her research in three-dimensional particle tracking and plenoptic cameras under the advisement of Professors Douglas H. Kelley and Jonathan D. Ellis.

Contributors and Funding Sources

This work was supervised by a dissertation committee consisting of Professors Douglas H. Kelley (co-advisor) and Jonathan D. Ellis (co-advisor) of the Department of Mechanical Engineering and Professor Miguel A. Alonso of the Institute of Optics. This dissertation was undertaken and completed by the student without outside funding support. Work completed by others and online code used in this research have been appropriately referenced.

Dedication

Dedicated to my father

Robert H. Bierasinski

Acknowledgments

Firstly, I thank my committee members, Professors Miguel A. Alonso, Douglas H. Kelley and Jonathan D. Ellis, for their continuous support and encouragement throughout my graduate career. Secondly, I thank the members of the Precision Instrumentation Group and the MixingLab for their assistance throughout my research, including borrowing equipment and continual advice. Thirdly, I thank Adalberto Perez for machining the camera mount used for calibration. Thank you to everyone that has reviewed my thesis. I am grateful for my friends who on countless occasions listened to me about my challenges and progress of my research and academic courses. Finally, I thank my family Terri, Lisa, and Christopher, for believing in me and supporting me in my endeavors.

Abstract

Plenoptic cameras, also known as light field cameras, have been implemented in depth estimation techniques [1, 2]. A key component to the plenoptic camera is a microlens array that eliminates the need for multiple cameras employed in other three-dimensional particle tracking techniques. The plenoptic camera captures a four-dimensional light field that is both a function of the spatial position as well as angular direction of the light ray striking the sensor [3]. A conventional camera loses the angular information of the light rays in the final image, because it records the average intensity of the light rays [1]. A plenoptic camera retains the angular information of the light rays, and as a result multiple views can be extracted via sub-aperture imaging [4]. These views can be used in place of a multiple camera array that conventional three-dimensional particle tracking methods require. Such methods include tomographic particle image velocimetry and synthetic aperture particle image velocimetry [5–8].

The raw files of a plenoptic camera used in this research, called the "Lytro", contain information in the metadata, including lens properties and image properties. A major component embedded in the raw files is a depth map that can be extracted when the file format is thoroughly understood. This depth map can be used in the calibration process by relating the values in the depth map to distance, which can be completed using calibration targets set at different distances from the camera. The work completed during this research produced such calibration and also explores the properties of a plenoptic camera, depth estimation techniques using a plenoptic camera, the file format, and camera calibration.

Table of Contents

List of Figures

1 Introduction

1.1 The Rationale for Three-Dimensional Particle Tracking

Two-dimensional particle tracking methods such as Lagrangian particle tracking (LPT) and the Eulerian counterpart particle image velocimetry (PIV) are commonly used in fluid dynamics to measure the velocity fields and other flow properties of individual particles in turbulent fluid flow [9]. Mapping a full three-dimensional velocity field to quantify fluid mechanical systems is ideal [10]. However, most current particle tracking methods, especially PIV, are typically used to measure the two-dimensional velocity fields [5,9,10]. Traditional methods alone, such as PIV, cannot describe the topology of the naturally three-dimensional turbulent fluid flows, because these methods only capture two-dimensional velocity fields [11].

The ability to validate numerical simulations of complex flows as well as to quantify experimental three-dimensional flows is important [6, 11] because the resolution of computational models have not only surpassed experimental models, but have also increased the separation between the predicted and observed phenomena in fluid flow [12]. Additionally, the ability to capture the full three-dimensional velocity field is critical for quantifying coherent flow structures in turbulent fluid flow [5]. Furthermore, to completely describe turbulent flow, which naturally exhibits three-dimensional behavior, it is necessary to determine the three-dimensional velocity field [5]. The velocity fields and path lines can be used when studying issues associated with mixing [10]. Three-dimensional particle tracking can not only be used to quantify turbulent flow but can also be used to analyze insect swarms [13], as well as bird flight and the combustion of flames [12].

1.2 State of the Art

Current three-dimensional particle tracking utilizes multiple conventional cameras simultaneously in techniques such as defocusing PIV, tomographic PIV, synthetic aperture PIV, and stereoscopic particle tracking velocimetry (PTV) [5, 7, 8]. There have also been significant efforts to reduce the number of cameras required in three-dimensional particle tracking techniques to a single camera [5, 8, 10, 11]. The pros and cons for each method are described and tabulated in Table 1.1.

1.2.1 Multiple Camera Methods

Defocusing PIV, or defocusing digital PIV, utilizes the defocus blur of particles to extract depth information [5–8, 14]. In general, a mask is placed in front of one or more cameras, and the mask usually has three apertures which are shifted from the optical axis of the camera and arranged as an equilateral triangle [6, 7, 10]. Thus, a single particle will appear at multiple locations on the image plane in which the separation between particles is related to the distance from the camera [6–8]. Figure 1.1 illustrates a schematic of an experimental setup used for defocusing PIV that utilizes three CCD sensors and three lenses each having a pinhole mask, illustrated in Figure 1.2. The three-dimensional spatial coordinates of the particle can be determined by first combining the images from all of the cameras on to a common coordinate system. Then an algorithm can be implemented that computes the particle coordinates and depth based on the geometry of the mask [5–7].

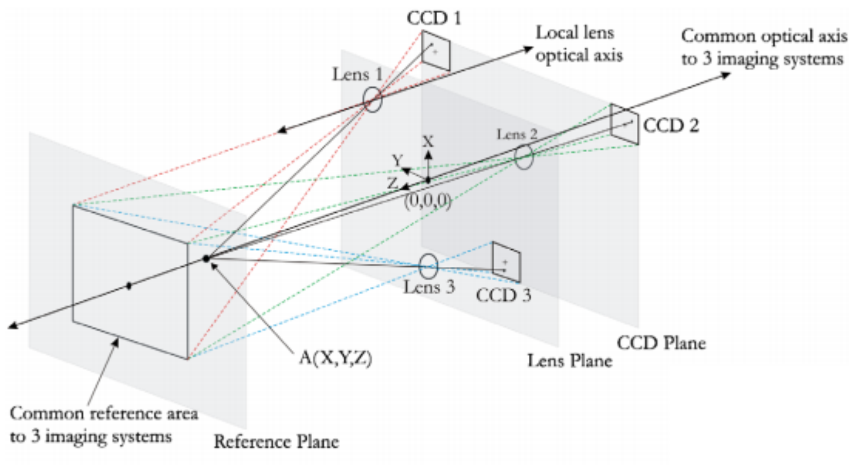


Figure 1.1: Experimental setup for defocusing digital PIV utilizing three CCD sensors and three lenses. Each lens has a mask with three apertures arranged as an equilateral triangle as shown in Figure 1.2 [14].

This technique is severely limited in seeding density, which is the number of particles per pixel (ppp) or per unit volume. Seeding density is one measure of the performance of the particle tracking method. More particles mean more data points, thus a higher seeding density is better. However, too many particles can create issues in particle detection and tracking, because the fluid becomes too crowded with particles. Defocusing PIV requires a low-seeding density because individual particle coordinates need to be resolved [5–8], and past simulations and experiments

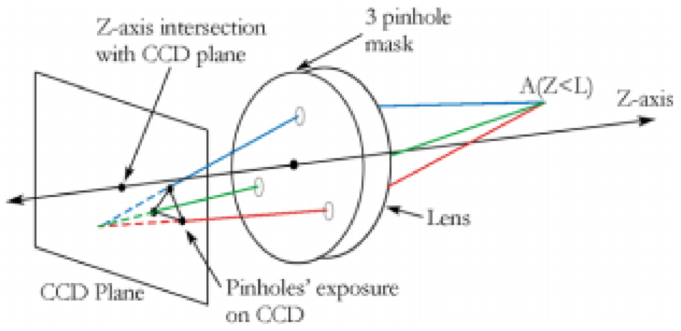


Figure 1.2: Schematic of pinhole mask placed in front of a lens used in defocusing digital PIV [14].

report seeding densities ranging from 0.034 to 0.038 ppp for measurement volumes ranging from $100 \times 100 \times 100$ mm^3 to $150 \times 150 \times 150$ mm^3 [6]. A high-seeding density would make it impossible to find particle image pairs during triangulation due to occluded particles [8]. However, this method has advantages including simple equipment and ease of analysis [5].

Tomographic PIV typically uses three to six cameras to capture multiple view points of a volume seeded with illuminated particles, where optical tomography algorithms are used to reconstruct the volume [5–8]. An experimental setup is illustrated in Figure 1.3. The spatial positions of the particles are determined through cross-correlation or a multiplicative algebraic reconstruction technique [5, 6, 8]. The main disadvantage with this method is that the measurement volume is significantly smaller than that of other particle tracking methods and is expensive; nonetheless, tomographic PIV allows for higher seeding densities, up to 0.08 ppp [5–7], and can handle overlapping particles [8].

Similar to that of tomographic PIV, synthetic aperture PIV, also known as light field imaging, utilizes an array of eight or more synchronized cameras to

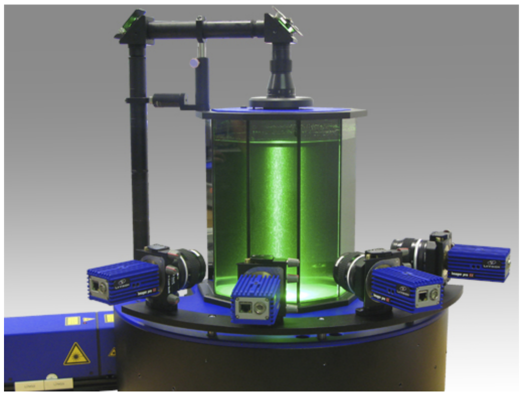


Figure 1.3: Experimental setup for tomographic PIV; containing four cameras viewing the measurement volume at different angles [15].

record multiple perspectives of the measurement volume. The images captured are recombined through refocusing algorithms to obtain multiple focus planes [5–7]. A camera array used in synthetic aperture PIV experiments by Truscott, *et al.* [12] is shown in Figure 1.4. A particle with a high intensity is in focus on the plane of interest, whereas particles with low intensities are out-of-focus [5, 6]. To form the three-dimensional intensity field, a refocusing algorithm is applied through the entire measurement volume and any out-of-focus particles are removed by applying a threshold. The particle coordinates are then computed by applying a cross-correlation algorithm to the reconstructed three-dimensional intensity field [5, 6].

Synthetic aperture PIV has the ability to resolve large volumes with a large particle seeding density that have been recorded as 0.05 ppp for a $40 \times 40 \times 30$ mm^3 volume and 0.17 ppp for a $50 \times 50 \times 10$ mm^3 volume [5]. Unfortunately, this method uses far more cameras and is more expensive than any of the other methods described [5–7]. A typical synthetic aperture PIV experiment also requires a minimum of four hours for calibration and data acquisition and an additional twelve hours to perform synthetic aperture refocusing [12].

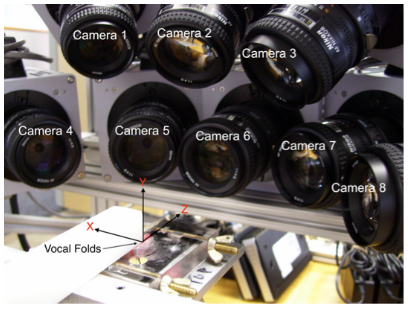


Figure 1.4: Experimental setup for synthetic aperture PIV, utilizing a large array of cameras for viewing the measurement volume at different angles [12].

Stereoscopic PTV also uses multiple cameras to capture multiple views of the same scene [7, 10]. A schematic for stereoscopic PTV used to measure wake-phenomena in a wind-turbine is illustrated in Figure 1.5. Particles are individually triangulated and tracked from frame to frame using conventional algorithms such as Lagrangian particle tracking, and then are reconstructed into three-dimensions [10]. The order of operations can be reversed where the three-dimensional coordinates of each particle are determined first by reconstructing the images of each camera, and the particles' trajectories are then determined using Lagrangian particle tracking techniques [9]. Either order of operations is sufficient, but Ouellette and Xu [9] state that particle seeding density is decreased by a factor of the added dimension simplifying the tracking problem, so it is preferred to track in three dimensions for a higher seeding density. The downfall with stereoscopic particle tracking velocimetry is that the calibration is a long and difficult process [10]. This method does allow for larger measurement volumes [7] and a past experiment reported 600 to 720 particles in a 512 $\times$ 512 pixel image [16].

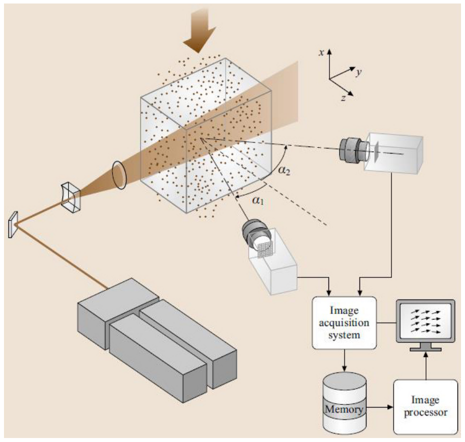
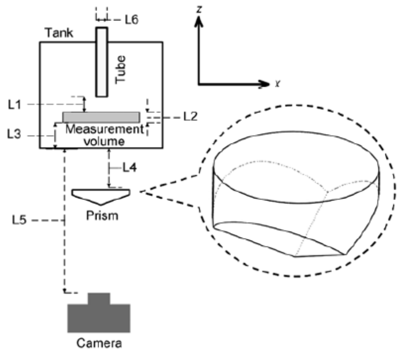


Figure 1.5: Schematic for stereoscopic PTV used to measure three-dimensional wind-turbine wake phenomena [17].

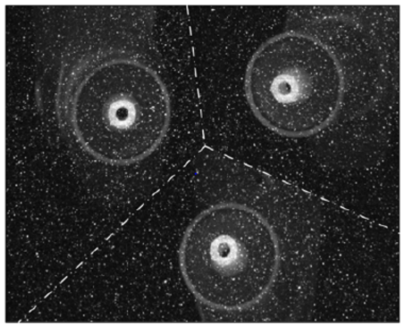
1.2.2 Single Camera Methods

There have also been advances in three-dimensional particle tracking which have reduced the number of cameras required to one. Most single camera three-dimensional particle tracking techniques utilize a camera called a plenoptic camera, also known as a light field camera. A plenoptic camera is able to record the spatial coordinates of the intensity distribution as well as retain the angular information of the light rays that create the image, whereas a conventional camera is only able to record the spatial coordinates [5, 7, 11, 18]. A second single camera technique combines a conventional camera and a three-vision prism to capture three views simultaneously [8]. Holographic PIV is another method that only requires a single camera to calculate particle positions [5, 7, 19]. Defocusing PIV, as previously explained, has also been successfully implemented with one camera [5,6,10].

Gao, *et al.* [8] introduced a single camera method that utilizes a three-vision prism. The prism is a spherical lens with one flat surface and one three-face-cone shaped surface. This prism is placed between the camera and the measurement volume as shown in Figure 1.6(a), and each face of the prism captures a different view of the object as illustrated in Figure 1.6(b). The investigators processed their data in a similar manner to three-dimensional particle tracking velocimetry or tomographic PIV. Before performing any data processing, the images must first be separated into the three individual views and any background noise must be subtracted from the images to increase the signal-to-noise ratio for improved particle identification.



(a)



(b)

Figure 1.6: Figure 1.6(a) illustrates a schematic for a single camera particle tracking method which utilizes a three-vision prism placed between the camera and the measurement volume. Figure 1.6(b) shows a sample particle image taken with the system shown in Figure 1.6(a). The dashed lines represent the boundaries between the different views [8].

After other necessary image pre-processing and volumetric calibration, particle triangulation or particle intensity reconstruction is used to obtain the three-dimensional particle positions. Velocities are determined using a three-dimensional particle tracking velocimetry algorithm. This method has recorded a spatial resolution of 0.07, 0.07, and 0.06 mm in the X, Y, and Z directions, respectively. An advantage of this method is that particle triangulation does not need self-calibration, though numerous image pre-processing steps are required, increasing the time required for data processing. Additionally, this method also has a very low seeding density of 0.01 ppp.

Holographic PIV determines the three-dimensional position of particles by analyzing the interference pattern that is formed by the light reflected from particles and a reference beam that is incident on the volume [5–7, 19]. The phase of the light wave diffracted by the particle can be traced by utilizing the interference pattern, which ultimately results in the distance between the particle and the sensor in the measurement volume [5–7]. Firstly, the hologram is recorded by overlapping the scattered particle light and a reference wave on a photographic plate as illustrated in Figure 1.7(a). Secondly, the hologram is illuminated with the reference wave and a virtual image is produced by diffraction from the interference pattern as shown in Figure 1.7(b). Finally, a real image of the particles is reconstructed by tracing a conjugate reference wave as illustrated in Figure 1.7(c).

Meng, *et al.* [20] has calculated the maximum seeding density allowed for a test volume with a depth of 40 mm to be 20 particles per mm^3. This technique has been performed through digital analyses and with film, but each method has disadvantages [5–7]. Data processing of the film based method is difficult and time consuming, whereas the digital platform is significantly easier to use at the expense of lower resolution [5–7, 20]. Film has better resolution than digital methods because of the size of the film [7], and a standard holographic plate can store up to 50 Gbytes of data, which cannot be achieved by digital methods [19]. However, experimentalists have been transitioning from film to digital holographic PIV and researching ways to improve digital holographic PIV [20].

Belden, *et al.* [6] and Lynch, *et al.* [5] briefly mention that a single camera can be used in defocusing PIV. The method for extracting the three-dimensional positions of

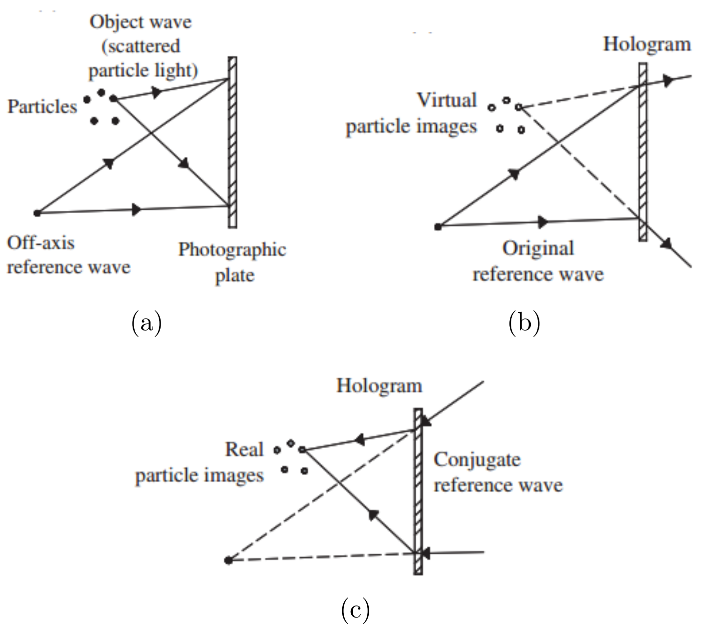


Figure 1.7: Process of holographic PIV to reconstruct particle coordinates. Figure 1.7(a) illustrates how the hologram of the particles is recorded. Figure 1.7(b) shows how the virtual image is produced. Figure 1.7(c) illustrates how the real image of the particles is reconstructed [19].

particles is similar to that of a multiple camera system. Willert, *et al.* [10] explain how defocusing PIV is applied to a single camera system. A mask is placed in front of the camera lens that has three apertures arranged in the shape of a triangle as illustrated in Figure 1.8. This causes a projection of multiple blurred images from one point source, in which the separation distance of the blurred images is directly related to the distance between the particle and the imaging system. To calculate the remaining two coordinates, the geometrical centers of self-similar image pairs and trigonometry are used. Particle densities between 0.005 and 0.025 particles per mm^3 have been recorded in experimental models [21]. Although defocusing PIV can be used in a single camera system that is simple to calibrate, the aperture used significantly limits how much light is collected, thereby restricting when and where this method

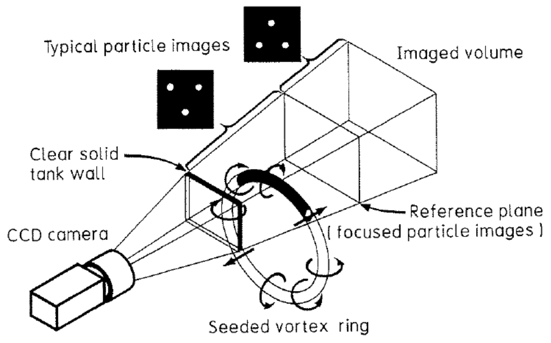


Figure 1.8: Schematic for single camera defocusing PIV where a three pinhole mask arranged as an equilateral triangle is placed in front of the main lens of the camera [10].

can reasonably be applied [5].

Plenoptic cameras utilize a microlens array, which is placed between the sensor and the main lens. They are also beneficial for macro imaging that is most commonly used in the laboratory [5]. Multiple views can be created from a plenoptic camera by properly sampling the correct pixel behind each microlens [1], and the number of views attainable is equivalent to the number of pixels behind each microlens [5]. This is somewhat similar to that of the three-vision prism, but plenoptic cameras can offer significantly more views. Plenoptic cameras will be explained in more detail in the following chapter.

Members from a research group at the University of Auburn [22] performed tomographic PIV with a simulated plenoptic camera, as illustrated in Figure 1.9, thus eliminating the need for multiple cameras. They created a synthetic particle field consisting of 20 particles in a $5 \times 5 \times 5$ mm^3 volume and is illustrated in Figure 1.10(a). An image taken with the simulated plenoptic camera is shown in Figure 1.10(b). The image was used with a multiplicative algebraic reconstruction technique to reconstruct the particle fields as illustrated in Figure 1.10(c). A three-dimensional cross-correlation algorithm can be used to determine the velocity fields in the reconstructed volume [5, 11]. This simulation can have a particle density of up to 0.001 ppp to ensure accurate results [5, 22]. Although these custom plenoptic cameras only require a onetime calibration of the microlens, the file sizes are very large which increases the time required for post-processing which is approximately 12 hours [5].

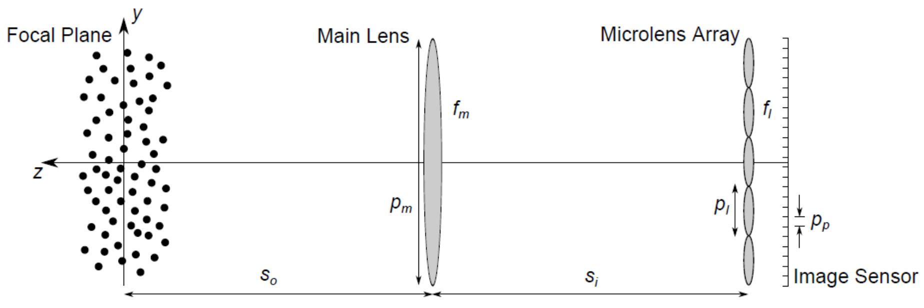


Figure 1.9: Schematic for tomographic PIV with a plenoptic camera [5].

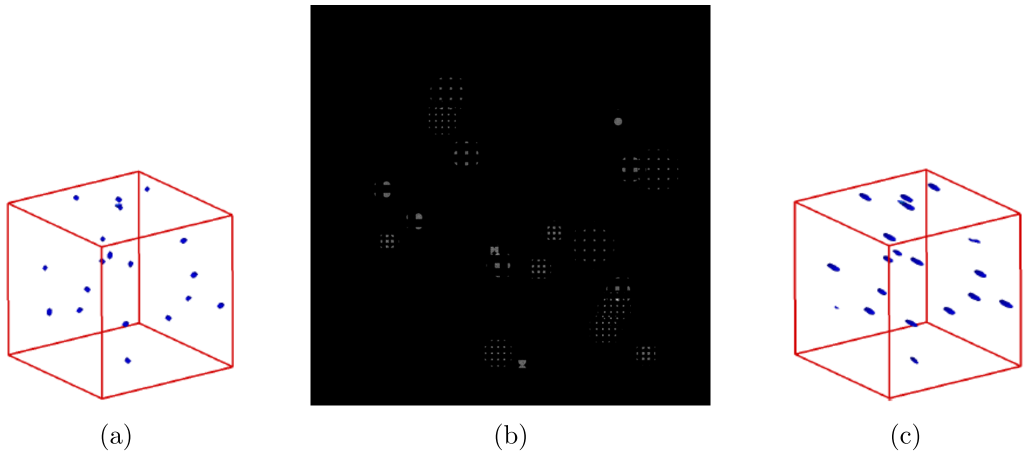


Figure 1.10: Reconstruction of a synthetic particle field using a plenoptic camera. Figure 1.10(a) illustrates the synthetic particle field. Figure 1.10(b) shows the image taken by the simulated plenoptic camera. Figure 1.10(c) illustrates the reconstructed particle field [22].

Cenedese, *et al.* [7] introduced another method of three-dimensional particle tracking with a plenoptic camera. The basic concept of this method is that the minimum size of the diameter of a particle corresponds to that particle being on the plane of focus. Figure 1.11 shows four computationally refocused images of the same scene to obtain separate images with different planes of focus. The spherical particles have the same diameter and were placed at different distances. By comparing a particle by itself (in the same row) in the refocused images the particle diameter changes in size.

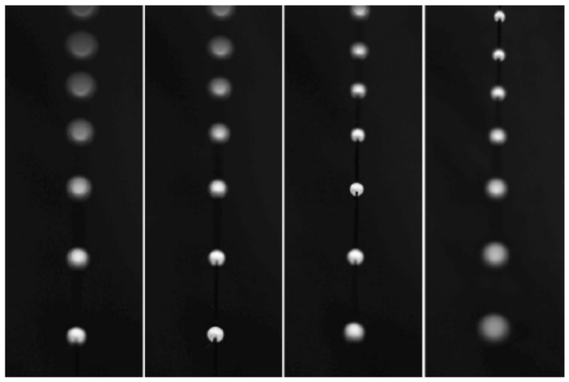


Figure 1.11: Four computationally refocused images of the same scene. Spherical particles with the same diameter at different distances from the camera have different planes of focus [7].

Each refocused image has a different particle at a different depth that is in-focus which illustrates the principle that particles at different depths have different planes of focus. Thus the distance between the particle and the camera plane can be related to the image of the particle with minimum area through appropriate calibration. Once the positions of the particles are extracted it is possible to reconstruct two consecutive scenes taken from the plenoptic camera into a volume, and then apply Lagrangian particle tracking algorithms to obtain a velocity field.

The main advantage for this system is that a single camera is required to obtain multiple images of a scene containing different focal planes. However this method has a low frame rate of 1 frame per second which restricts its application.

Another novel method for three-dimensional particle image velocimetry using a plenoptic camera is described by Skupsch and Brücker [18] which utilizes the concept of synthetic aperture PIV. They use a microlens array placed in front of a camera sensor and illuminate their measurement volume with five equally spaced light sheets, in which tracer particles from all of the light sheets are simultaneously imaged onto the sensor as illustrated in Figure 1.12.

First, each lenslet in the microlens array is considered a sub-image, and then all sub-images are superimposed in order to have the particle of interest line up to the central sub-image. Thus, each sub-image has a corresponding shift vector so that the particle falls on top of the particle in the central sub-image. Next, all the shifted sub-images are summed and normalized. Finally, a threshold is applied to remove any background noise that corresponds to out-of-focus particles. This procedure is repeated for every particle of interest, which creates multiple shift-maps for each refocus plane, thus refocusing on different depths.

Skupsch and Brücker [18] note that one can use their proposed imaging system for particle tracking velocimetry as well. This method has a spatial resolution of 1.3 mm, which is better than that of commercial plenoptic cameras [18]. Additionally this method is robust, easy to implement and does not require fine tuning, but the apparatus has a more complex illumination system. Another disadvantage is that this method can only cover a small range of angles, thus reducing the angular resolution.

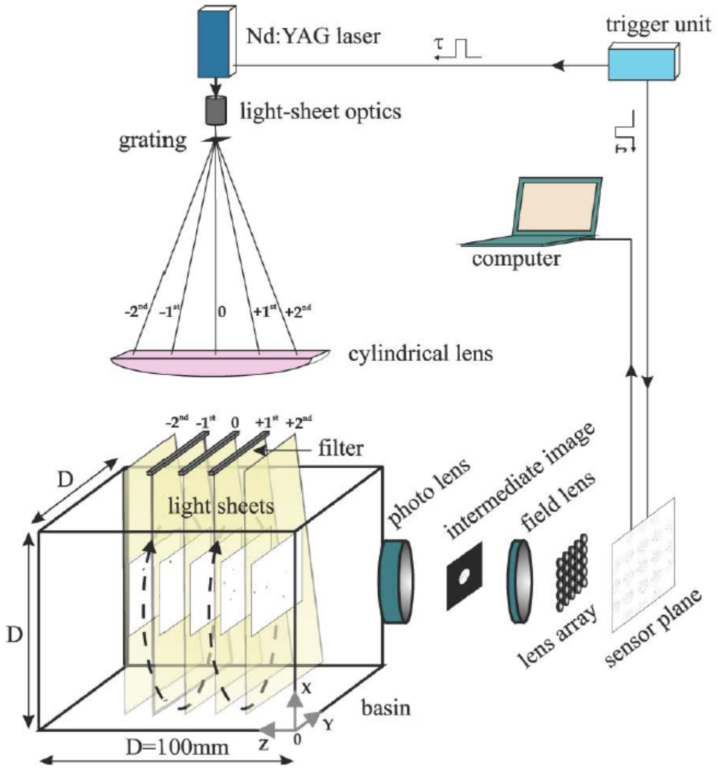


Figure 1.12: Schematic for synthetic aperture with a plenoptic camera. The tracer particles from all five light sheets are simultaneously imaged onto the sensor plane [18].

Table 1.1: Tabulation of pros and cons for current three-dimensional particle tracking methods.

Method	**Pros**	**Cons**
Multiple Camera Methods		
Defocusing PIV	-Simplicity of equipment -Ease of analysis	-Requires a low-seeding density -Cannot handle overlapping particles
Tomographic PIV	-Higher seeding density -Can handle particles that overlap	-Measurement volume is small -Complicated setup and control -Expensive -Limited number of viewing angles
Synthetic Aperture PIV	-Can see occluded particles -High spatial resolution -Large measurement volumes -High seeding density -Simple and robust algorithms	-Uses several cameras -Requires a lot of time and money
Stereoscopic PTV	-Large measurement volumes	-Calibration is difficult and lengthy -Low seeding density
Single Camera Methods		
Holographic PIV	-Film can be densely sampled -Film has better resolution because of film size -Digital is user friendly	-Image processing of holographic plates is difficult -Complex optics -Limited to small measurement volumes -Digital has a lower resolution than film
Defocusing PIV	-Calibration is straightforward -Simple to use	-Apertures reduce the amount of light collected
Tomographic PIV	-Microlens only calibrated once	-Large file size increases post-processing time
Gao, *et al.* [8]	-Particle triangulation does not need self-calibration	-Multiple image pre-processing steps required
Skupsch, *et al.* [18]	-Spatial resolution is better than commercial plenoptic cameras -Robust and easy to implement -No fine tuning of system	-More complex illumination system -Small ranges of angles covered
Cenedese, *et al.* [7]	-One camera captures multiple depth planes	-Slow frame rate

2 The Plenoptic Camera

A conventional camera collects light across the lens aperture, and the structure of the light varies depending on what sub-region of the lens aperture it strikes. The final image is an average of all possible images seen from different sub-regions of the lens aperture. Therefore, at the camera sensor, any angular information about the light is lost because the sensor only records the average of all the light rays from the different viewing angles [1]. Figure 2.1 shows a ray diagram for a single lens system; note how two rays at different angles, θ and ϕ, recombine at the image plane, thereby losing any angular information of the light rays.

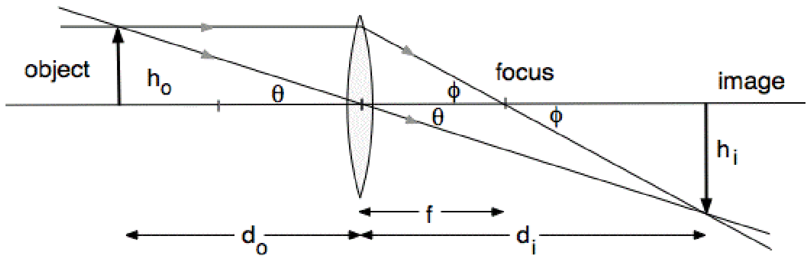


Figure 2.1: Ray diagram depicting two rays at angles θ and ϕ of a conventional camera [23].

A plenoptic camera is very similar to a conventional camera, except that the plenoptic camera utilizes a microlens array placed between the main lens and the sensor [1, 4, 24]. The microlens array is used to retain the angular information of the light rays striking different areas of the lens aperture [1, 4, 25]. Specifically, each microlens captures a perspective view of the scene being imaged from that position on the microlens array [24] and measures the amount of light along each incident ray striking that microlens [25]. Figure 2.2 demonstrates how a microlens array separates the converging rays, thus retaining the angular information of the light rays in the final image.

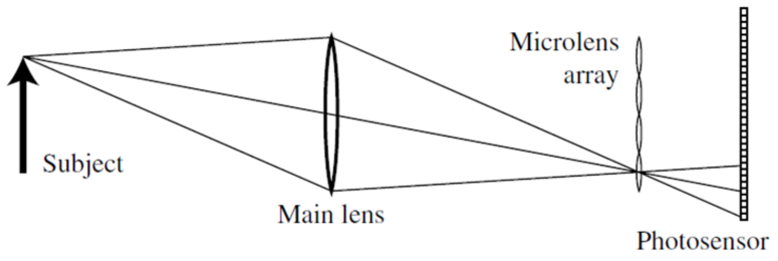


Figure 2.2: Ray diagram of a plenoptic camera, illustrating how the microlens array retains the angular information of the light rays by separating the converging rays [4].

A plenoptic camera records what is referred to as the light field, $I(x, y, \phi_x, \phi_y)$, which captures the intensity of light as a function of the two-dimensional spatial position, x and y, and the respective angular directions with respect to the optical axis, ϕ_x and ϕ_y, of the light rays [4, 24, 25]. Additionally, blurring at the sensor is dependent on the object position. Figure 3.2 is a ray diagram of a plenoptic camera for three different particle positions. Note that the rays shown are only a subset of all the possible rays. A particle that is on the object plane, shown in blue, is in-focus on the image plane. Whereas, a particle that is left of the object plane, shown in red, focuses before the sensor plane blurring the image at the sensor. Similarly, a particle right of the object plane, shown in green, focuses after the sensor plane and its image is also blurred at the sensor. Therefore, objects at different depths will appear blurred if they are not on the object plane and in-focus if they are on the object plane.

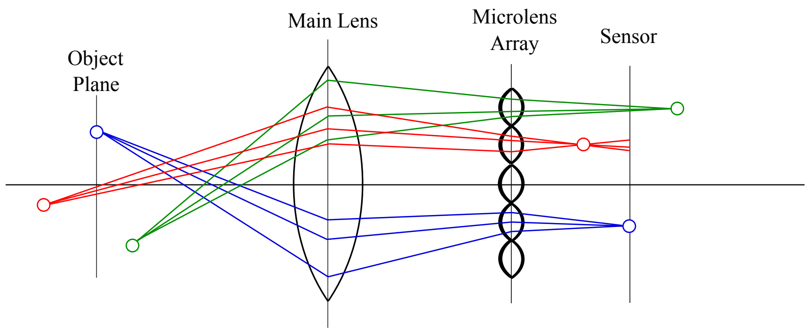


Figure 2.3: Ray diagram of a plenoptic camera for three different particle positions, illustrating how their blur at the image plane is dependent on their position relative to the object plane. Each color represents a different particle, and the rays shown are a subset of all possible rays. Adapted from Hahne, *et al.* [26].

Figure 2.4 illustrates a simple orthogonal microlens array, where x and y indicate the global spatial position of the microlens array, and v_x and v_y is the local pixel position within each microlens. Microlens arrays are available in various shapes and sizes ranging from square to circular shaped lenslets [27] and orthogonal to hexagonal grid arrays [27,28]. As an example, Figure 2.5(a) shows a raw light field image taken with an orthogonal microlens array with square shaped lenses. Figure 2.5(b) shows an enlarged view of the raw light field image, which is boxed in red in Figure 2.5(a).

When a hexagonal array is used, it is easier to post-process images if the data are dehexed prior to any analysis [29]. Dansereau, *et al.* [29] dehexed their hexagonal grid array following the methods described in *H_2O:Reversible Hexagonal-Orthogonal Grid Conversion by 1-D Filtering* [30]. A MatLab toolbox is available that is formulated in *Decoding, Calibration and Rectification for Lenselet-Based Plenoptic Cameras* and dehexes the hexagonal data as well as provides calibration and rectification for Lytro cameras [31]. A brief example using this toolbox is described in the Appendix.

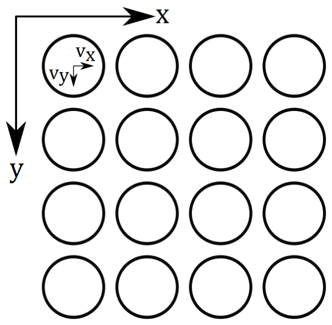


Figure 2.4: Schematic of an orthogonal microlens array, where x and y denote the spatial position of the microlens, and v_x and v_y represent the pixel coordinates within each microlens.

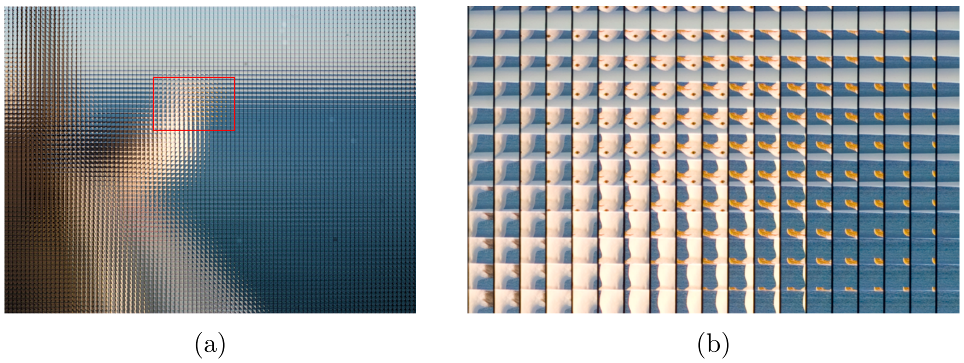


Figure 2.5: Raw light field image of a seagull taken with an orthogonal microlens array with square lenses [32]. Figure 2.5(a) images the full light field of the seagull. Figure 2.5(b) enlarged part of Figure 2.5(a) (boxed in red) to show the square microlenses.

If an orthogonal microlens array has $P \times P$ microlenses and $N \times N$ pixels behind each microlens, there will be $N \times N$ number of views and each view will have $P \times P$ pixels [24]. Views are equivalent to looking through different sub-apertures on the main lens, because all the light rays that pass through the sub-aperture of interest are focused through related pixels under different microlenses [4]. The top image of Figure 2.6 illustrates how the light rays that pass through a pixel also pass through its parent microlens and through a corresponding sub-aperture on the main lens. The bottom image of Figure 2.6 demonstrates how all the light that passes through a sub-aperture on the main lens focuses through related pixels under the microlenses. The views correspond to the angular dimensions of the light field, ϕ_x and ϕ_y, and because the views are created by selecting the appropriate pixel positions, v_x and v_y, the angular dimensions are therefore analogous to the pixel position [5]. Due to this analogy, the intensity of the light field can be written as a function of x, y, v_x, and v_y as $I(x, y, v_x, v_y)$.

In order to reconstruct each view, the same respective pixel must be taken from each microlens [1, 4]. Figure 2.7 illustrates this concept, where one view was formed by extracting a pixel near the top of each microlens of Figure 2.5(a) and a second view was formed by selecting a pixel near the bottom of each microlens. The two views formed are images seen through two different sub-apertures on the main lens, thus these two views are not identical as seen in the vertical parallax between the two views. The bottom image in Figure 2.7 shows an increase in the length of the sky in the image by Δ, thus exhibiting vertical parallax when compared to the top image where the sky is of length d_r.

After all possible views are extracted; it is possible to use those views in three-dimensional particle tracking methods as described earlier, instead of using multiple cameras. This approach reduces the amount of time required for camera calibration as well as cost, because only one camera is required. Custom plenoptic cameras can be made similar to the ones created by Adelson, *et al.* [1] and Fahringer, *et al.* [11]. There are also plenoptic cameras available for purchase such as the Lytro

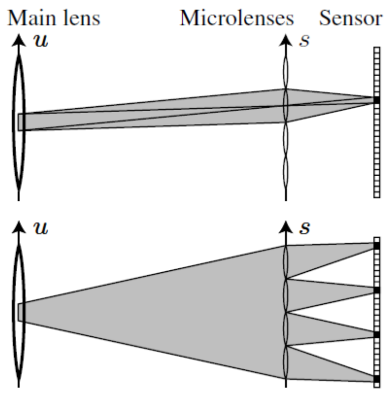


Figure 2.6: *Top:* Light that passes through a pixel also passes through its parent microlens and through its corresponding sub-aperture on the main lens. *Bottom*: Light that passes through the sub-aperture on the main lens is focused through related pixels under the microlenses [4].

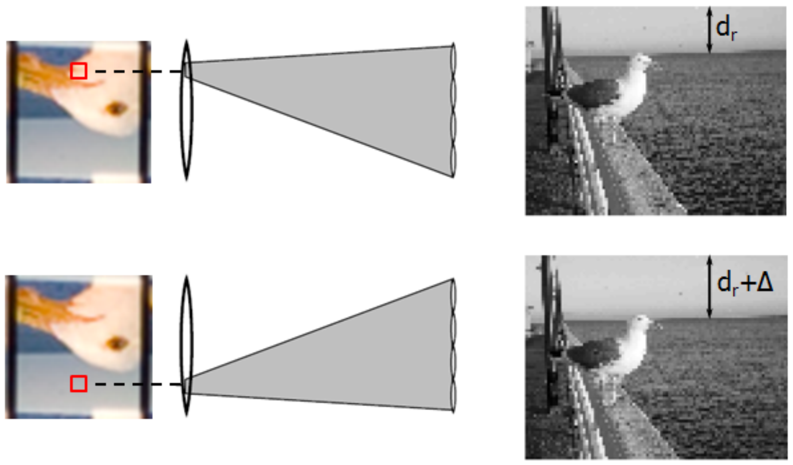


Figure 2.7: Two views (depicted on right) from two sub-apertures on the main lens extracted from the raw light field image in Figure 2.5(a) by selecting the pixel boxed in red behind each microlens (shown on left). Note the vertical parallax, Δ, between the two views when comparing the length of the sky in the image. Image adapted from Ng, *et al.* [4].

Light Field Camera or their newest generation camera the Lytro Illium, which are more oriented towards photography [33]. Raytrix also sells 3D light field cameras that are primarily used in industry and research settings, but come with a higher price tag [34]. In this work a Lytro camera was calibrated to a depth look up table embedded in the metadata of the camera, which will be explained in detail in the following chapter.

Furthermore, the number and pixel size of microlenses used effects the lateral resolution and depth of field respectively. Consider a plenoptic camera with a sensor of a fixed size for two different microlens arrays. Let one microlens array be a 6 × 6 array of microlenses that are each 3 × 3 pixels as illustrated in Figure 2.8(a). This array would provide a 3 × 3 array of views that are each 6 × 6 pixels as shown in Figure 2.8(b). Let the second array be a 3 × 3 array of microlenses that are each 6 × 6 pixels as shown in Figure 2.9(a). This array would result in a 6 × 6 array of views that are each 3 × 3 pixels as illustrated in Figure 2.9(b).

The 6 × 6 array of microlenses has a lower lateral resolution than that of the 3 × 3 array, because the associated views have fewer pixels than the views from the 3 × 3 array of microlenses. The 3 × 3 array of microlenses has a smaller depth of field than that of the 6 × 6 array, because it has only 9 views whereas the 6 × 6 microlens array has 36 views. Therefore, when holding the sensor size constant the lateral resolution increases as the number of microlenes increases because the pixel size of each view increases. However, the depth of field decreases as the number of microlenses increases as shown in Figure 2.10, because the pixel size of each microlens decreases resulting in fewer possible views. A smaller depth of field corresponds to a smaller depth resolution. This concept should be considered when implementing a plenoptic camera in three-dimensional particle tracking methods to determine the lateral and depth resolution of the system.

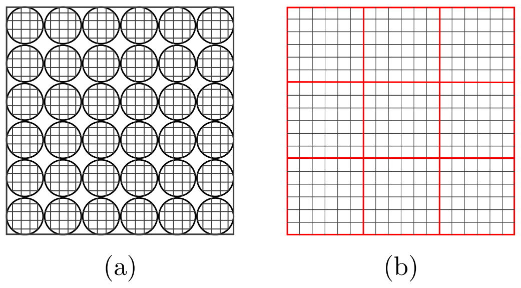


Figure 2.8: Figure 2.8(a) illustrates a 6 × 6 array of microlenses that are each 3 × 3 pixels. Figure 2.8(b) shows the resulting 3 × 3 array of views and each view is 6 × 6 pixels

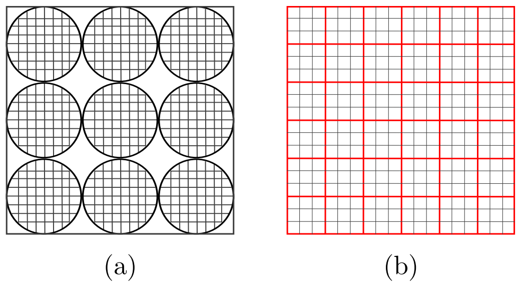


Figure 2.9: Figure 2.9(a) shows a 3 × 3 array of microlenses that are each 6 × 6 pixels. Figure 2.9(b) shows the resulting 6 × 6 array of views and each view is 3 × 3 pixels

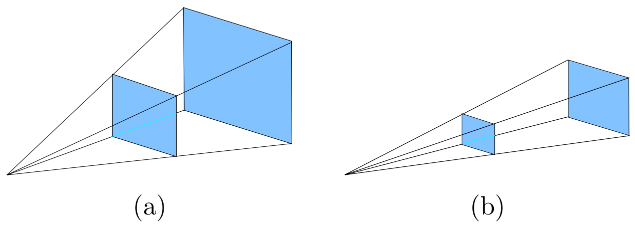


Figure 2.10: The depth of field for the 6 × 6 array of microlenses shown in Figure 2.10(a) is smaller than the depth of field for the 3 × 3 microlens array shown in Figure 2.10(b).

3 Depth Estimation Techniques

3.1 Two-Dimensional Cross-Correlation

There are other methods that have been used to extract the depth information from images captured with a plenoptic camera, and could be utilized as part of a three-dimensional particle tracking algorithm. Binocular stereo systems are widely used to retrieve depth information from a scene through two-dimensional cross-correlation but this requires two cameras [1]. Additionally, binocular stereo systems create challenges due to correspondence problems, and only exploit parallax along one axis, and thus cannot be used for depth estimations for contours parallel to that axis [1]. In order to reduce these issues, a third camera should be implemented [1], which is counter-productive if the goal is to utilize only one camera.

Adelson and Wang [1] explain a depth estimation technique that performs single lens stereo analyses through the use of a plenoptic camera. They take advantage of the multiple views that a plenoptic camera can produce by extracting the appropriate subpixels from each microlens, thereby forming an array of images. Image displacement is estimated between all possible image pairs by using a least-squares method. An image pair is simply two adjacent views. For example, if there is a 3×3 array of views, as shown in Figure 3.1, there will be six image pairs in the horizontal direction and six image pairs in the vertical direction totaling twelve possible image pairs. The least-squares estimator, h, is used to calculate an image displacement estimate and is equivalent to

$$h = \frac{\sum_P (I_x I_{v_x} + I_y I_{v_y})}{\sum_P (I_x^2 + I_y^2)}, \tag{3.1}$$

where P is the integration patch, $I = I(x, y, v_x, v_y)$ is the four-dimensional intensity, the subscripts signify differentiation with respect to the corresponding dimension, x and y denote the spatial dimensions, and v_x and v_y denote the viewing dimensions. A large integration patch is recommended by Adelson and Wang [1], to reduce noise.

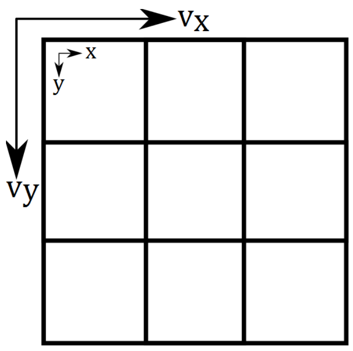


Figure 3.1: A 3×3 array which represent views, where x and y denote the spatial dimensions and v_x and v_y denote the viewing dimensions.

Confidence for the image displacement estimate in x is

$$c_x = \sum_P I_x^2, \tag{3.2}$$

and similarly the confidence in y is

$$c_y = \sum_P I_y^2. \tag{3.3}$$

Regions of rapid change or that have large contours would result in high confidence, whereas areas that are smooth or featureless would result in low confidence. Thus, a region with primarily horizontal contour would have low confidence in x, but a high confidence in y and *vice versa* for images exhibiting primarily vertical contour.

After completing the displacement analysis by selecting the image displacement estimate, h, with the highest confidence, h can be used to extract depth. To do this, basic geometrical optics are employed to calculate the depth. Figure 3.2 illustrates the geometry of single lens stereo as used by Adelson and Wang [1]. After implementing the lens equation and using similar triangles, it follows that

$$\frac{1}{d} = \frac{1}{F} - \frac{1}{f}(1 - \frac{h}{v}) \tag{3.4}$$

where F represents the focal length of the lens, f is the distance between the lens and sensor plane, d is the distance to a particular point object, v is the displacement of the aperture, and h is the image displacement estimate of the object on the sensor plane calculated using Equation 3.1. Unfortunately, this method cannot be implemented when using a Lytro Light Field Camera, because Lytro, for proprietary reasons, does not provide all the optical properties that are required to compute depth using Equation 3.4. Specifically, the distance between the lens and the sensor of the camera needs to be known or calculated, which cannot be done with the information that Lytro provides. This method however would be easy to implement if a custom plenoptic camera were used because all optical properties of the system would be known, including the distance between the lens and the sensor and the focal length of the lens.

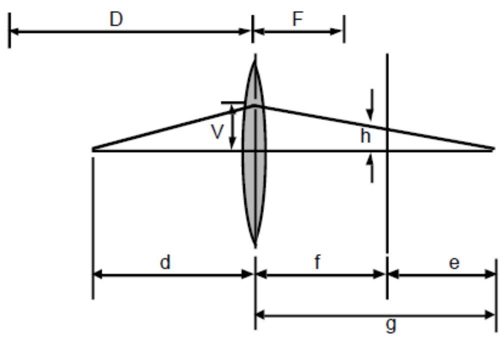


Figure 3.2: Schematic of single lens stereo [1].

3.2 Refocusing

The concept of extracting depth by refocusing was briefly explained in the Introduction. If a particle is in the plane of focus then the diameter will be minimized. The system should be calibrated by relating the change in diameter of a focused particle to the distance of the plane of focus [7]. This calibration will thereby associate particle diameter to depth, and, with proper calibration, the depth of the particle of interest can be determined by finding the correct image, where the particle of interest has the minimum area [7]. Quantitatively, digital refocusing requires a summation of shifted versions of views known as sub-aperture images [25, 35].

Georgiev, *et al.* [35] explain how to perform refocusing by first creating an array of all possible views and then from that array selecting a reference image. Figure 3.3 shows the subset of the possible views that can be taken from Figure 2.5(a) and the reference image which is the center view boxed in red. The square microlenses are approximately 74 × 74 pixels in size, thus there are 5476 views available. For simplicity, only 25 views are used in this example, and were created by taking combinations of 10, -10, 20 and -20 pixels away from the center pixel of each microlens in both the v_x and v_y directions.

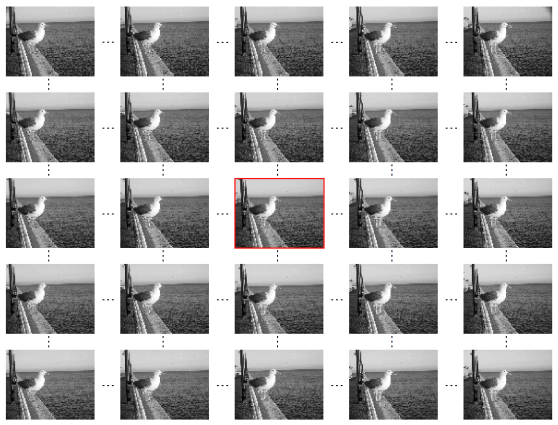
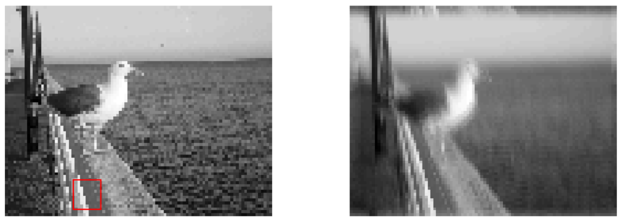


Figure 3.3: Subset of all 5476 possible views from Figure 2.5(a) where the reference image chosen is boxed in red.

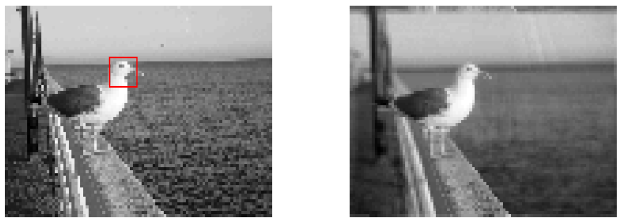
A region of interest (ROI) is selected on the reference image to which the rest of the images will be matched. A two-dimensional shift, S, is performed for each image, other than the reference image, to align the image with the ROI. A normalized two-dimensional cross-correlation, $c(x, y)$, is calculated between every point of each image and the appropriate channel of the chosen ROI. The shift that results in the highest cross-correlation is used for the final shift of that particular image. Each image will have a corresponding shift vector, (dx, dy), for the chosen ROI. Each image is then shifted by its calculated shift vector, which then are uniformly blended with the reference image to create a refocused image. The refocused image refocuses on the plane of the objects in the chosen ROI. This procedure is useful because coplanar objects will have the same shift [35].

This process was performed with Figure 3.3 for two different ROIs, refocusing at two different depth planes, as illustrated in Figure 3.4. To refocus on the railing in front of the seagull, the ROI was chosen to be on the railing in front of the seagull as shown in red in the left image of Figure 3.4(a). The right image of Figure 3.4(a) illustrates the image after refocusing on to the railing. Next, the ROI was chosen as the seagull's head as shown in red in the left image of Figure 3.4(b) to refocus on the seagull's head. The right image of Figure 3.4(b) is the associated refocused image which shows that the image is refocused on the head of the seagull. It is clear that Figure 3.4(a) is properly refocused on the railing and Figure 3.4(b) is refocused on the seagull's head.

To create various intermediate depth planes for refocusing, shifts should be calculated for the foreground of the image, S_f, and for the background, S_b. This is similar to how the seagull was refocused to two different object planes in the image. The intermediate depth plane shift, S_D, can be obtained by linear interpolation using



(a)



(b)

Figure 3.4: Figure 3.4(b) shows the ROI in red on the railing in front of the seagull (left) and the refocused image (right). Figure 3.4(b) shows the ROI in red on the head of the seagull (left) and the refocused image (right).

$$S_D = S_f + D(S_b - S_f), \tag{3.5}$$

where D represents the depth plane, and ranges between 0 and 1 [35]. When D is equivalent to 0, the resulting image would be refocused on the foreground and when D is equivalent to 1 the image would be refocused on the background of the scene.

3.3 Fourier Slice Photography

Another method of generating refocused images of a four-dimensional light field is called Fourier slice photography. This method is significantly more complex than the two-dimensional cross-correlation method, but is also more robust. Ng [36] explains that the Fourier slice photography theorem is where an image is the inverse two-dimensional Fourier transform of a dilated two-dimensional slice of the four-dimensional Fourier transform of the light field. The Fourier slice theorem results in a photograph, P_α,

$$P_\alpha = \frac{1}{f^2} F^{-2} \circ (S_2^4 \circ B_\alpha^{-T}) \circ F^4, \tag{3.6}$$

where a two-dimensional slice, $S_2^4 \circ B_\alpha^{-T}$, is extracted from the four-dimensional Fourier transform of the light field F^4. A four-dimensional change of basis, B_α, is equivalent to a shear, and its inverse transpose. $B_\alpha^{-T}$are equivalent to

$$B_\alpha = \begin{bmatrix} \alpha & 0 & 1-\alpha & 0 \\ 0 & \alpha & 0 & 1-\alpha \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad B_\alpha^{-T} = \begin{bmatrix} 1/\alpha & 0 & 0 & 0 \\ 0 & 1/\alpha & 0 & 0 \\ 1-1/\alpha & 0 & 1 & 0 \\ 0 & 1-1/\alpha & 0 & 1 \end{bmatrix} \tag{3.7}$$

where α is equal to f'/f, and f' is the depth to the desired refocus plane and f is the distance between the sensor and the lens. Then a two-dimensional inverse Fourier transform, F^{-2} is applied and the resulting image is scaled by a factor of $\frac{1}{f^2}$, where f is the distance between the lens and the sensor. Note, the operator "$\circ$" represents a function composition. A function composition nests two or more functions to create a single function; for example, $(f \circ g)(x) = f(g(x))$ [37].

Figure 3.5 shows the Fourier slice photography algorithm to create refocused images. The top left image is the raw light field image, and the top right image is the light field image after a four-dimensional Fourier transform. The bottom right image is one after the appropriate change of basis is performed and the desired slice is extracted. After a two-dimensional inverse Fourier transform, a final refocused photograph at the new virtual film plane is retrieved, as shown in the bottom left image.

This method of Fourier slice photography for refocusing has been implemented in a MatLab script by Yu [38]. The MatLab script refocuses images for given α values and creates the four-dimensional light field from an array of images, where each image was taken by a camera in a larger array of cameras. Thus, if a plenoptic camera is used, all of the possible views must be extracted to be used in the script. The α value chosen to refocus an image is also associated with the depth of a particle that has the minimum area in that image. Therefore, this MatLab script can also be used to create refocused images to be used in the depth estimation analyses as described by Cenedese, *et al.* [7].

The two-dimensional cross-correlation or Fourier slice photography can provide refocused photographs of a scene to be used in estimating depth following the method explained by Cenedese, *et al.* [7]. Once depth is extracted for the particle of interest,

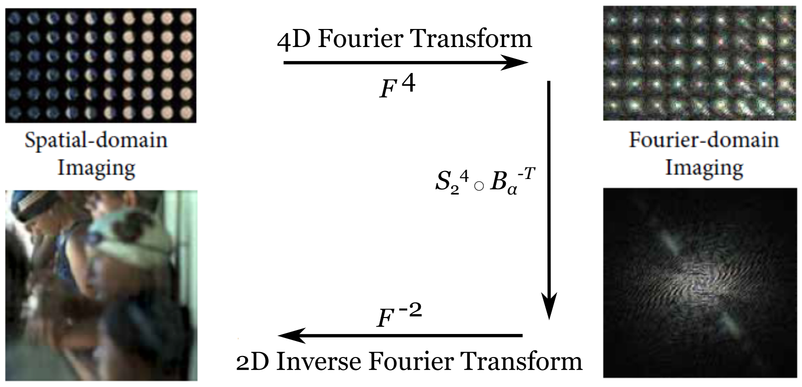


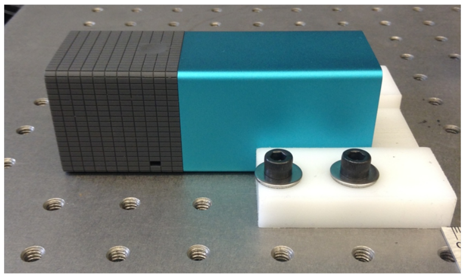
Figure 3.5: Fourier slice photography algorithm where the top left image is a subset of the light field image, the top right image is the four-dimensional Fourier transform, the bottom right image is the extracted slice after a change of basis, and finally the bottom left image is the refocused photograph, adapted from Ng [36].

standard Lagrangian particle tracking methods can be used to create a complete three-dimensional particle track [7]. It may be possible through calibration to relate D in the two-dimensional cross-correlation method [35] or α in the Fourier slice photography [4, 25] to meters. This has been explored for Fourier slice photography, and is described in the next chapter on camera calibration.

4 Calibration

4.1 The Apparatus

A custom camera mount was machined to hold the camera in place during calibration, due to the lack of mounting threads on the Lytro camera. The Lytro is shown secured in the camera mount in Figure 4.1(a). Two 38 cm by 38 cm LEGO® baseplates were fixed to an optical table, one in front of the other to create a 76 $\times$ 38 cm canvas. The camera mount was also fixed to the optical table, and placed flush to the leading baseplate. A Siemens' star focus target pattern was used as a calibration target and was attached to columns made from LEGO® bricks so that they can be secured to the baseplates. Any background that was not the calibration target in the scene was covered in black fabric to allow for a clear distinction between the calibration targets and the background. This set up is shown with example calibration target positions in Figure 4.1(b).


(a)

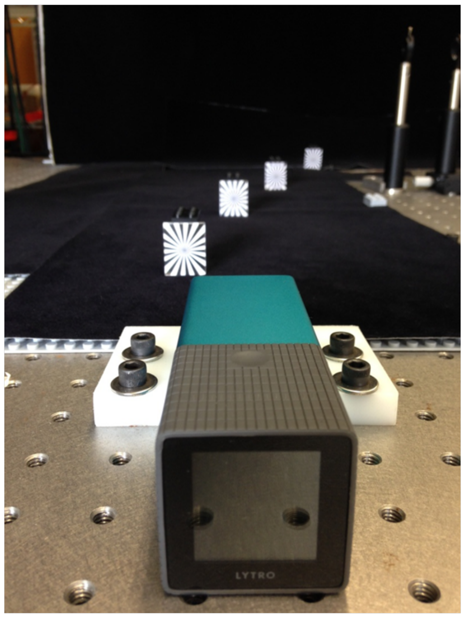

(b)

Figure 4.1: Figure 4.1(a) shows the custom camera mount to secure the Lytro in place. Figure 4.1(b) shows the calibration set up with example calibration target positions.

4.2 Methods

4.2.1 Fourier Slice Refocusing

In this work, Fourier slice refocusing was first used in an attempt to calibrate the Lytro camera to the α parameter by implementing the MatLab script created by Yu [38]. The code was used to refocus images from the Lytro camera for various α values. Four Siemens focus targets were placed at four different positions within the view of the camera, at 120 mm, 332 mm, 520 mm, and 737 mm from the camera plane as shown in Figure 4.2. The images were then refocused for a range of α values from -1 to 1, with steps of 0.1, and between -0.1 and 0.1 the steps size was reduced to 0.01. For each calibration target a region of interest was selected, from which the standard deviation and the mean were calculated. Figure 4.3 shows an array of the same region of interests selected for different α values used.

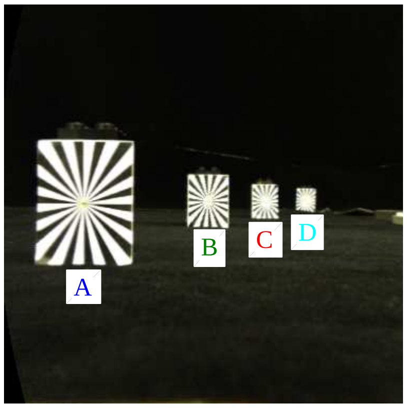


Figure 4.2: View of four Siemens star focus targets A, B, C, and D used to test calibration for α, and placed at 120 mm, 332 mm, 520 mm, and 737 mm from the camera plane, respectively.

The standard deviation *versus* α, the mean *versus* α, and standard deviation normalized by the mean *versus* α were plotted for all four columns as shown in Figure 4.4. A high sigma value would correspond to the target being in focus. When refocusing for various α values it is expected that each calibration target would have a corresponding α value to which it is best in focus. However, Figure 4.4 illustrates that refocusing for α using the MatLab code created by Yu [38] does not provide a clear distinction as to which α is best for each target. All of the calibration targets have a peak sigma around the same α value, which shows that using this code to calibrate for α is not possible. It is necessary that α can be resolved for different depth layers, which would require that the four calibration targets have different α values.

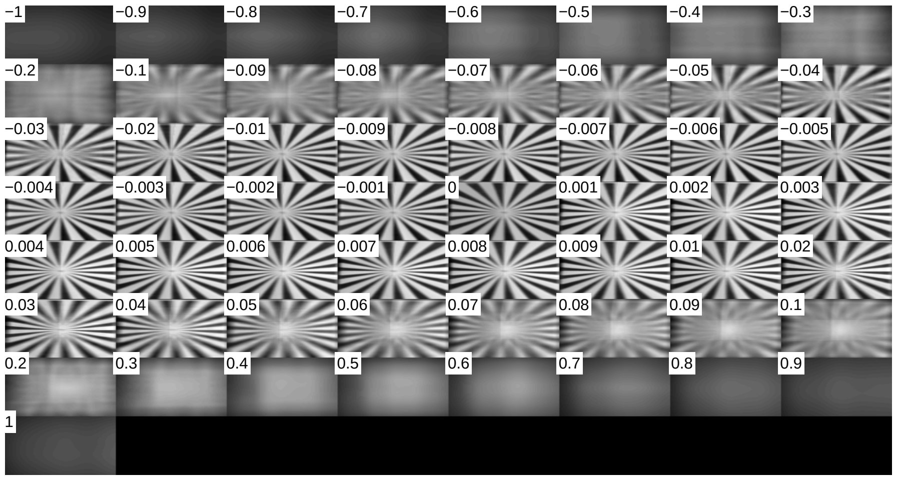


Figure 4.3: This figure illustrates an array of the same ROI from the same target from one Lytro image refocused for different values of α.

These errors seen in the refocusing code could be due to the type of calibration targets used. The Siemens focus targets exhibit a change in spatial frequency with a lower spatial frequency on the edge of the target and a higher spatial frequency at the middle of the target. This high spatial frequency causes the center of the target to always appear blurry. When the standard deviation is used to determine whether the target is in focus after performing Fourier slice refocusing, the blurry center of the target might skew the results. In order to test this, a checkerboard grid which has a lower and constant spatial frequency was used, and the Fourier slice refocusing process was repeated for the same α values. The four targets were also placed in the same positions as before at 120 mm, 332 mm, 520 mm and 737 mm as shown in Figure 4.5(a). The results for sigma, the mean, and sigma normalized by the mean were plotted against α for each of the calibration targets as illustrated in Figure 4.6(a)

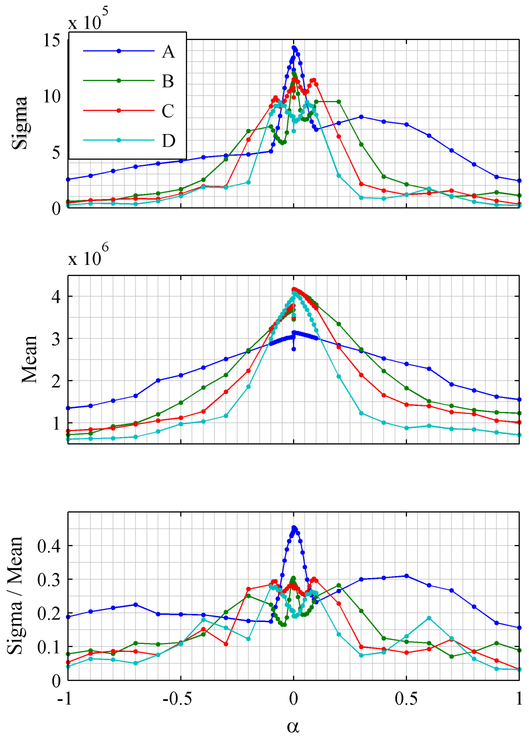


Figure 4.4: Graphs of standard deviation (top), mean (middle), and standard deviation normalized by the mean (bottom). All are plotted against α.

Similar to the Siemens star focus targets, the checkerboard targets have a peak sigma value at the same value of α. Each checkerboard target did not result in a different α value, however, this could be due to their position from the camera plane instead of the type of target. The refocusing calibration process was repeated again for checkerboard targets, but within 100 mm from the camera at 75 mm, 83 mm, 91 mm, and 98 mm from the camera plane. The positions of the four targets is shown in Figure 4.5(b) and the results are shown in Figure 4.6(b). The results also show that the α values cannot be resolved for different depths, therefore the issue with this method for calibration is related to the code used. There are discrepancies in the change of basis matrix, B_{α}, described in Ng's PhD thesis [36] and what Yu's MatLab code [38] uses which is based off of Ng's PhD thesis. An investigation should be done on Yu's MatLab code to determine what is causing the refocusing errors, or a custom Fourier slice refocusing code should be created. The custom code should then be compared to the results from Yu's MatLab code to further determine where the errors are originating from. However, there is a different method to calibrate the camera and is explained in detail in the following section.

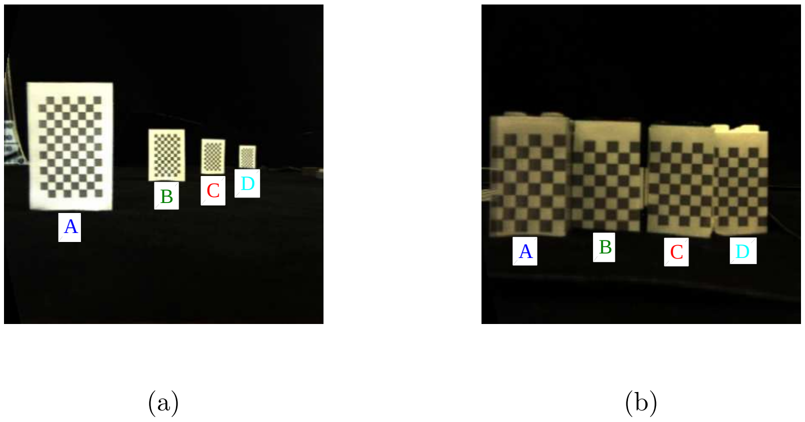
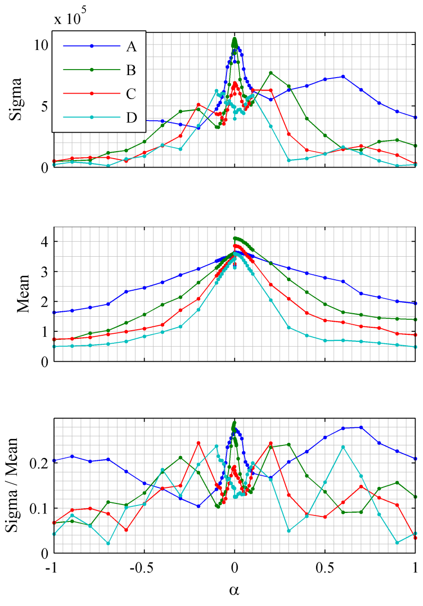
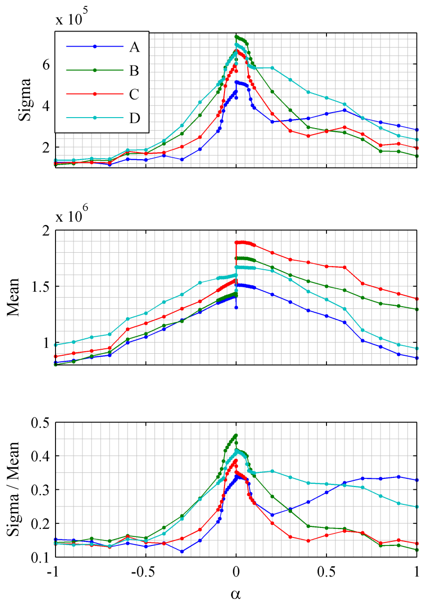


Figure 4.5: Four checkerboard targets A, B, C, and D used to re-test calibration for α. Figure 4.5(a): Targets placed at 120 mm, 332 mm, 520 mm, and 737 mm from the camera plane. Figure 4.5(b): Targets placed at 75 mm, 83 mm, 91 mm, and 98 mm from the camera plane.



(a)



(b)

Figure 4.6: Figure 4.6(a): Graphs for checkerboard targets placed at 120 mm, 332 mm, 520 mm, and 737 mm from the camera plane. Figure 4.6(b): Graphs for checkerboard targets placed at 75 mm, 83 mm, 91 mm, and 98 mm from the camera plane. Standard deviation (top), mean (middle), and standard deviation normalized by the mean (bottom) for the checkerboard targets. All are plotted against α.

4.2.2 Lytro's Depth Map

The depth map retrieved from Lytro's raw lfp file has unknown units, so it is important to calibrate the depth map to meters. To do this a very simple calibration is required. First, the calibration targets, overlaid on LEGO® bricks, were gradually moved by one LEGO® unit starting one LEGO® unit away from the camera mount, and a picture was captured for every new position. One LEGO® unit is equivalent to 7.8 mm. A total of ninety-four positions were captured as shown in Figure 4.7.

Knowledge of the Lytro file format is critical for properly extracting the depth

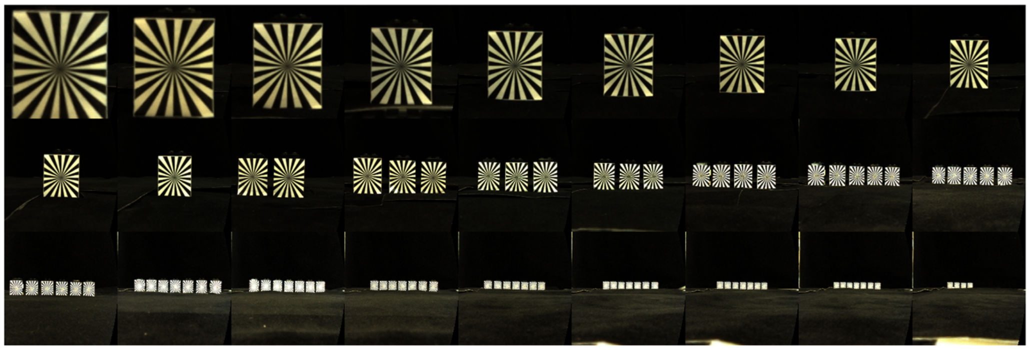


Figure 4.7: Array of images of the calibration targets captured during the Lytro camera calibration.

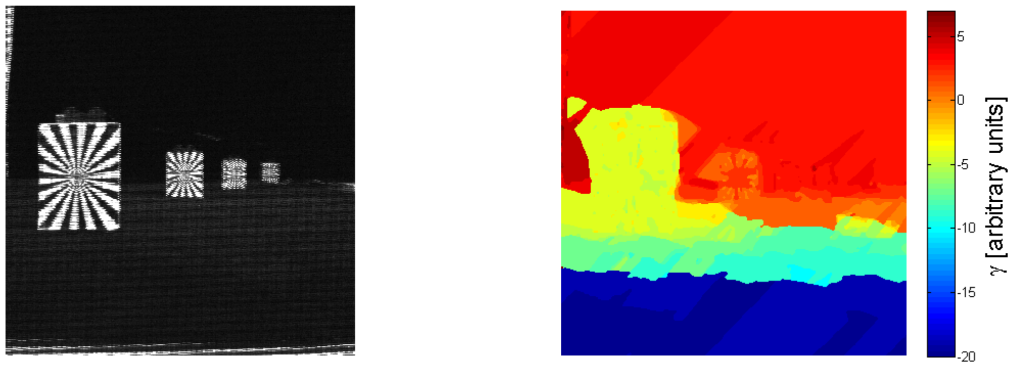
map; the file format is explained in detail in the Appendix. A custom MatLab script was written to extract the depth map for each raw stack lfp file and is included in the Appendix. There are also other programs available online, such as N. Patel's lfptools [39] that can be used to extract the metadata of a raw lfp image file from the Lytro camera, but does not output the depth map for the newest Lytro software version, which is a main component needed for this research. This depth map is labeled as depthLUT in the metadata of a raw stack lfp file and provides information on the size of the depth map, the representation, and the image tag as shown in Table 4.1.

Table 4.1: depthLUT listed in the metadata of a raw stack lfp file where width and height are the size of the depth map, representation is the file format for the depth map, and imageRef is the image tag associated with the depth map.

```
"depthLut" : {
  "width" : 328,
  "height" : 328,
  "representation" : "raw",
  "imageRef" : "sha1-b78f6366b83c99b7e05c5f27180e6509866673d7"
},
```

The script parses through the file, reads the metadata, and then searches the metadata for the size of the depthLUT, because the size of the depthLUT will depend on the software version when the picture was taken. The component in the metadata that contains the depthLUT is then located in the metadata by ascertaining the component length that is equal to the height of the depthLUT times the width of the depthLUT times four. The factor of four is used to take the data type of the depthLUT into account, which is a double array that is float32 equivalent to 4 bytes.

Figure 4.8(a) shows a raw stack lfp file of four focus targets, and Figure 4.8(b) is the corresponding depth map that has been obtained from the MatLab script. The units of the depth map are unknown, but through this calibration a correlation can be made between the values of the depth map and meters. After extracting the depth map for each file the depth map value, γ, for each LEGO® row was calculated by selecting a region of the calibration target and taking the average. This process was repeated for every file, and Figure 4.9 shows four example files and the region of interest selected for each target in the view of the camera.



(a) (b)

Figure 4.8: Figure 4.8(a) is a raw light field image of four focus targets and Figure 4.8(b) is the corresponding depth map (380 × 380 pixels) ranging from -20 to 7.

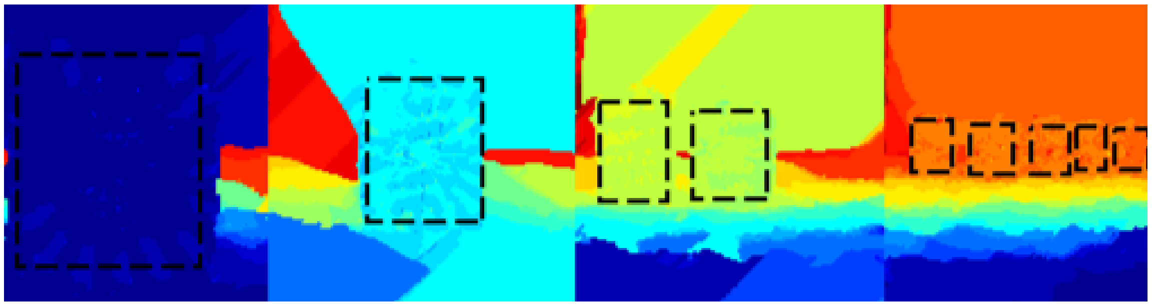


Figure 4.9: Example of regions selected (boxed) in four different depth maps used to calculate the depth map value.

4.3 Results

A direct correlation between the depth map and LEGO® unit can be determined. The measured depth value, γ, was plotted against the distance from the camera to the calibration target, l, as shown in Figure 4.10. The error bars were calculated from the standard deviation of the ROI used to calculate the γ value. Figure 4.10 clearly shows that the camera is more sensitive when the object is closer to it. Conversely, as the object is moved away from the camera, the ability for the camera to resolve objects that are at different depths declines. Eventually, when the object is moved far enough away from the camera at approximately 250 mm, the camera can no longer distinguish the depth of objects placed at different distances from the camera.

By inspection, it is clear that the measurements shown in Figure 4.10 resemble a one-term exponential function

$$\gamma = Ae^{Bl} \tag{4.1}$$

where

$$A = -36.76 \text{ mm and } B = -0.017 \text{ mm}^{-1},$$

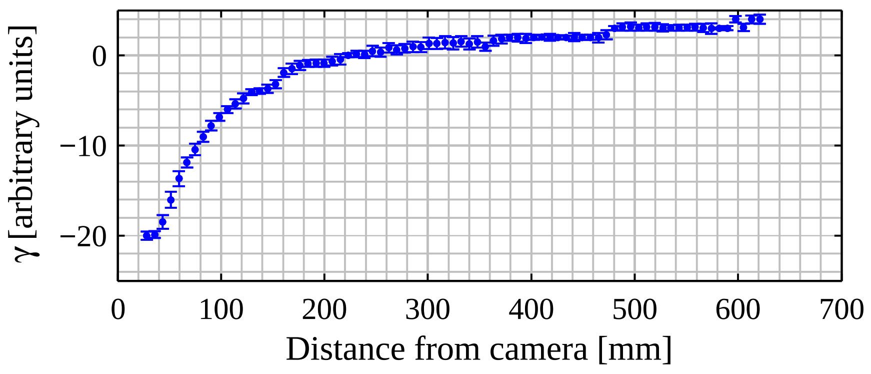


Figure 4.10: Measured depth values, γ, plotted against distance to the calibration target from the camera. Error bars were calculated from the standard deviation of the ROI used to calculate γ

or possibly a two-term exponential function

$$\gamma = Ae^{Bl} + Ce^{Dl} \tag{4.2}$$

where

$$A = -0.679 \text{ mm}, \; B = 0.003 \text{ mm}^{-1},$$
$$C = -33.66 \text{ mm, and } D = -0.014 \text{ mm}^{-1}.$$

Considering that an optical system is involved, the lens equation may be more reasonable. The lens equation is

$$\frac{1}{f} = \frac{1}{l'} - \frac{1}{l}, \tag{4.3}$$

where, l is the object distance, l' is the image distance, and f is the focal length. Equation 4.3 can be used to fit to the calibration curve shown in Figure 4.10 by introducing three coefficients, A, B, and C such that Equation 4.3 becomes the modified lens equation

$$\frac{1}{f} = \frac{1}{\frac{\gamma}{C} - A} - \frac{1}{l - B}, \tag{4.4}$$

where

$$\gamma = Cl'.$$

The modified lens equation can be rearranged, solving for γ to obtain

$$\gamma = C\left[A + \frac{l - B}{1 + \frac{l-B}{f}}\right] \tag{4.5}$$

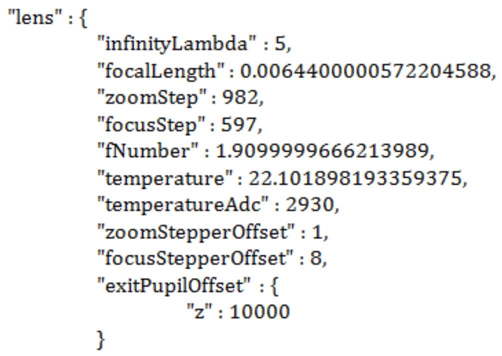
where

$$A = -6.291 \text{ mm}, B = -24.98 \text{ mm and } C = 0.024 \text{ arbitrary units/mm}.$$

The focal length used for fitting can be found in the metadata under a section called "lens" as shown below in Table 4.2. The focal length of the system, called "focalLength" in the metadata, is equivalent to 0.00644 m or 6.44 mm. The measurements were fit to Equation 4.1, Equation 4.2 and Equation 4.5 and shown plotted with the original measurements in Figure 4.11. The one-term exponential had an R-squared value of 0.8744, whereas the two-term exponential and the modified lens equation had larger R-squared values of 0.9952 and 0.9893 respectively. Thus, the two-term exponential and the modified lens equation fit the measurements better than the one-term exponential function. Although the two-term exponential has the highest R-squared value, the modified lens equation would be a more reasonable function to choose, because the fit coefficients have a physical relationship to the optical system, and the two-term exponential function does not.

Specifically the coefficients A and B in Equation 4.5 are lateral shifts applied to the image and object distances, because the exact location of the principal planes for the optical system is unknown. Since object distance was measured from the camera plane and not the principal plane the lateral shift B adjusts for this; and B is negative which means it is adding the additional distance from the principal plane inside the Lytro camera to the camera plane for the new object distance $l - B$. The lateral shift A applied to the image distance adjusts for the location of the second principal plane inside the Lytro camera. The final coefficient C relates image distance to the measurement values γ. Furthermore, the units of the coefficients and their magnitude are reasonable, thus Equation 4.5 is the best fit for the calibration relationship.

Table 4.2: Lens properties embedded in the Lytro metadata which includes the focal length of the system



```
"lens" : {
        "infinityLambda" : 5,
        "focalLength" : 0.0064400000572204588,
        "zoomStep" : 982,
        "focusStep" : 597,
        "fNumber" : 1.9099999666213989,
        "temperature" : 22.101898193359375,
        "temperatureAdc" : 2930,
        "zoomStepperOffset" : 1,
        "focusStepperOffset" : 8,
        "exitPupilOffset" : {
                "z" : 10000
        }
```

To determine the resolution of the depth estimates, the derivative of Equation 4.5 was taken. These results are shown in Figure 4.12. The resolution of the depth estimates is larger when the object is close to the camera. As the object is moved farther away from the camera the resolution rapidly declines to nearly zero as seen in Figure 4.12. Therefore, it would be best to keep the measurement volume close to the camera, within approximately 100 mm from the camera, when performing depth estimation analyses in order to prevent any ambiguities in depth estimates when objects are far from the camera. When the object is significantly far from the camera, over 100 mm away, the camera can no longer resolve the different depths of objects placed in different locations.

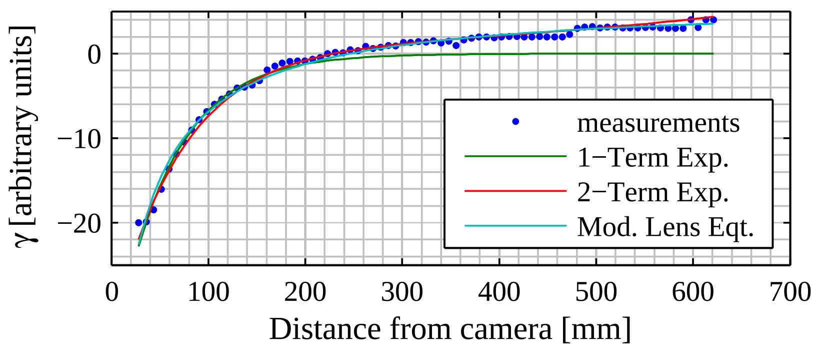


Figure 4.11: Measured values plotted with the results of fitting three equations, a one-term exponential, a two-term exponential, and the modified lens equation.

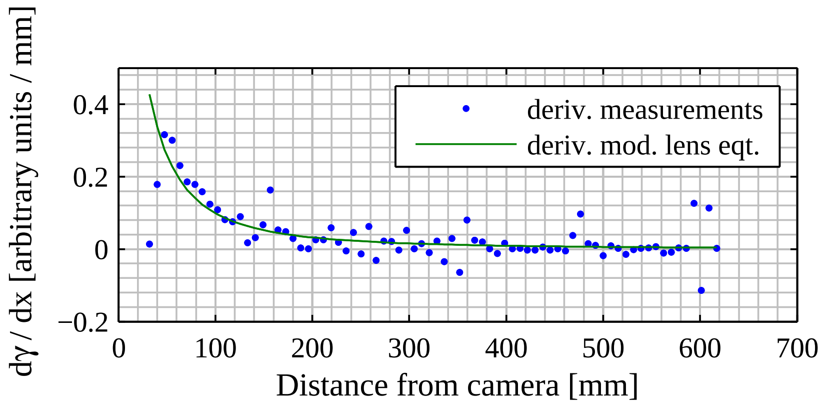


Figure 4.12: The derivative of Equation 4.5 is plotted to show the resolution of the depth values from the final calibration curve.

5 Conclusion

Three-dimensional particle tracking with a single camera can greatly reduce the amount of time required for calibration and image processing, which for some multiple camera systems requires a minimum of sixteen hours for calibration, data acquisition, and particle tracking combined [12]. Additionally the cost of three-dimensional particle tracking with a plenoptic camera is significantly reduced, because most methods described in the Introduction used three or more cameras which is expensive. Two methods for depth estimation with a plenoptic camera were explored. Two-dimensional cross-correlation is a simple method that can be used to refocus photographs at various depth planes [35], whereas Fourier slice photography is a more sophisticated method that uses Fourier transforms, image slicing, and inverse Fourier transforms to create a refocused photograph [25, 36].

When the file format of a raw light field image is understood, important data and parameters such as the depth map or the focal length of the system can be acquired. The file format has been explained in detail by Kučera [40], but, in general, the raw light field image contains three major sections: the package, metadata, and components. These sections provide a large amount of detailed information about the camera and the image that was taken.

Finally, it has been shown that the depth map embedded in the raw light field stack file can be calibrated to depth through the lens equation, where the final calibration curve can be represented by Equation 4.5. Additionally, when analyzing for sensitivity, it was shown in Figure 4.12 that the camera is more sensitive to depth when the object is close to the camera, and the ability for the camera to distinguish different depths rapidly declines when the object is moved farther away from the camera. Thus, when particle tracking methods are employed with a Lytro camera the measurement volume should be relatively close to the main lens of the camera. When attempting to calibrate the Lytro camera to α through Fourier slice refocusing it has been shown that the available source code [38] is not able to resolve α for various depth planes and has discrepancies with Ng's PhD thesis from which it is based off of [36].

5.1 Future Work

The resolution of the depth estimates decreases as the object is moved further away from the main lens of the camera, and was illustrated in Figure 4.12. When the object is close to the camera between 0 and 100 mm the resolution ranges between 0.4 to 0.1 mm^{-1}, respectively. As the object moves further from the camera the resolution rapidly declines to nearly 0 mm^{-1}. Thus, if the Lytro camera were to be used in an experiment, the measurement volume should be placed within 100 mm of the camera plane to ensure the various depths of the particles can be resolved. Since the measurement volume should be within 100 mm from the camera, the depth of the measurement volume is limited to 100 mm. That is the same as the depth of the measurement volume used in defocusing PIV (100 mm) [6], and larger than the depths used in synthetic aperture PIV (10 mm to 30 mm) [5].

Once the Lytro camera's depth map is successfully calibrated to distance, completed in this work, the next steps are to perform simple three-dimensional particle tracking methods. For example, objects such as LEGO® bricks can be suspended in a measurement volume and moved gradually between each consecutive picture taken. Using the depth maps available in those images and the associated calibration curve defined by Equation 4.5, the depths of the LEGO® bricks could be determined. Furthermore, standard Lagrangian particle tracking techniques can be used to create the particle tracks. Once the simple particle tracking methods are successfully established, more complex particle tracking experiments can be introduced. For example the Lytro camera could be placed at 45° and 100 mm from a mixing tank to simulate three dimensions as illustrated in Figure 5.1. The results from the Lytro camera could then be compared to results from a standard camera.

A more in depth investigation should be done to determine the primary cause of error when using the publicly available code for Fourier slice refocusing by Yu [38]. It has been shown that the type of calibration target and the position relative to the camera did not affect the results when determining the α values, therefore the errors are related to the code used. Additionally, discrepancies were found between the change of basis matrix, B_α, stated by Ng [36] and what was used by Yu [38]. The change of basis matrix should be further explored, and a custom script should be created to perform Fourier slice photography to be compared with the code made available by Yu [38].

Furthermore the frame rate for a Lytro camera is very slow, approximately

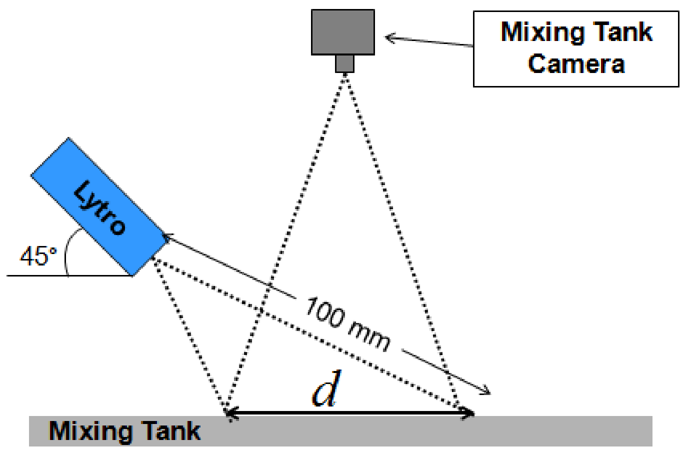


Figure 5.1: Example future experiment with the Lytro camera placed at 45° and 100 mm from a mixing tank, and compared to the results from a standard camera. The field of view for the two cameras used, d, should overlap.

1 frame per second [7]. This slow frame rate restricts where this camera can be reasonably implemented for three-dimensional particle tracking to experiments that analyze slow fluid flow. If the fluid being analyzed moves too quickly, it can be deduced that the Lytro will not be able to accurately measure the positions and the trajectories of the tracer particles.

In order to accurately conclude if the Lytro camera would be a better solution for three-dimensional particle tracking, the particle seeding density would need to be determined and compared to common seeding densities used by other methods. For example, synthetic aperture PIV has used particle seeding densities of 0.05 to 0.17 ppp depending on the size of the measurement volume [5], whereas tomographic PIV has used particle seeding densities up to 0.08 ppp, and defocusing PIV used seeding densities between 0.034 to 0.038 ppp, which are also dependent on the size of the measurement volume [6].

At this point, only the distance of the object from the camera can be determined from Lytro camera's depth map. Therefore, further action is required to determine the remaining two spatial coordinates of the object in the scene, which can be done through additional calibration. The height and width of the calibration targets used is known, and the image height and width of the calibration targets in the recorded image from which magnification, M, can be calculated by

$$M = \frac{h_i}{h_o}, \tag{5.1}$$

where h_i is the image height or width and h_o is the object height or width. An image of a target close to the camera as shown in Figure 5.2(a) and an image of the same sized targets far from the camera as shown in Figure 5.2(b) would appear smaller far from the camera. The two images also have the same pixel size so the magnification in the height or width could also be related to the distance of the calibration target, which would result in a second calibration curve or mapping function relating magnification to distance. This secondary calibration could be used in later experiments to determine the two remaining spatial coordinates, and thus allow for the Lytro camera to be used in three-dimensional particle tracking techniques.

Additionally, modifying the Lytro camera to take video would be an interesting topic to explore. If the Lytro camera would be able to be modified to take video, then the three-dimensional particle tracking could be done in real time instead of having to take a series of photos. However, at this time, the resources are not available to explore this option.

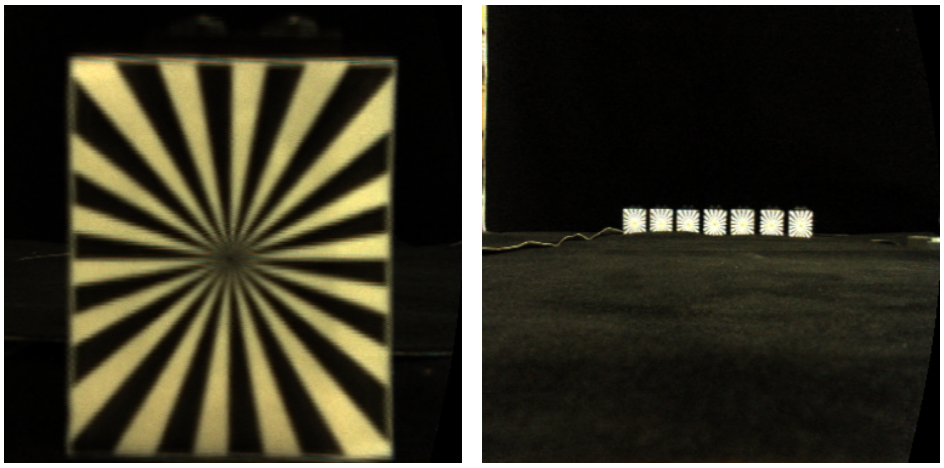


Figure 5.2: Figure 5.2(a): Image of a target close to the camera. Figure5.2(b): Image of targets far from the camera.

Bibliography

[1] E. Adelson and J. Wang. Single lens stereo with a plenoptic camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):99–106, 1992.

[2] T. Bishop and P. Favaro. Plenoptic depth estimation from multiple aliased views. In *2009 IEEE 12th International Conference on Computer Vision Workshops*, 2009.

[3] S. Wanner, J. Fehr, and B. Jähne. Generating epi representations of 4d light fields with a single lens focused plenoptic camera. In G. Bebis *et al.*, editor, *Advances in Visual Computing*, volume 6938 of *Lecture Notes in Computer Science*, pages 90–101. Springer Berlin Heidelberg, 2011.

[4] R. Ng, M. Levoy, M. Brédif, G. Duval, M. Horowitz, and P. Hanrahan. Light field photography with a hand-held plenoptic camera. *Stanford Technical Report*, 2005.

[5] K. Lynch, T. Fahringer, and B. Thurow. Three-dimensional particle image velocimetry using a plenoptic camera. In *50th AIAA Aerospace Sciences Meeting*, Nashville, Tennessee, January 2012.

[6] J. Belden, T.T. Truscott, M. Axiak, and A.H. Techet. Three-dimensional synthetic aperture particle image velocimetry. *Measurement Science and Technology*, 21(12), 2010.

[7] A. Cenedese, C. Cenedese, F. Furia, M. Marchetti, M. Moroni, and L. Shindler. 3d particle reconstruction using light field imaging. In *16th Int Symp on Application of Laser Techniques to Fluid Mechanics*, Lisbon, Portugal, July 2012.

[8] Q. Gao, H. Wang, and J. Wang. A single camera volumetric particle image velocimetry and its application. *Science China Technological Sciences*, 55(9), September 2012.

[9] N. Ouellette and H. Xu. A quantitative study of three-dimensional Lagrangian particle tracking algorithms. *Experiments in Fluids*, 40:301–313, 2006.

[10] C.E. Willert and M. Gharib. Three-dimensional particle imaging with a single camera. *Experiments in Fluids*, 13:353–358, 1992.

[11] T. Fahringer and B. Thurow. Tomographic reconstruction of a 3-d flow field using a plenoptic camera. In *42nd AIAA Fluid Dynamic Conference and Exhibit*, New Orleans, Louisiana, June 2012.

[12] T. Truscott, J. Belden, J.R. Nielson, D.J. Daily, and S.L. Thomson. Determining 3d flow fields via multi-camera light field imaging. *Journal of Visualized Experiments*, 73, March 2013.

[13] D.H. Kelley and N.T. Ouellete. Emergent dynamics of laboratory insect swarms. *Scientific Reports*, 3:1–7, 2013.

[14] L. Kajitani and D. Dabiri. A full three-dimensional characterization of defocusing digital particle image velocimetry. *Measurement Science Technology*, 16(3):790–804, March 2005.

[15] LaVision. Flowmaster tomographic piv. http://tinyurl.com/lt9oz9c. [Online, accessed 11-November-2014].

[16] Y. Ge, S.S. Cha, and J. Park. Study of particle tracking algorithms based on neural networks for stereoscopic tracking velocimetry. *Optics and Lasers in Engineering*, 44(6):623–636, June 2006.

[17] Sandia National Laboratories. Sandia is developing a Doppler global velocimetry system to understand fundamental wind-turbine wake phenomena. http://tinyurl.com/mnzxj44, March 2014. [Online, accessed 11-November-2014].

[18] C. Skupsch and C. Brücker. Multiple-plane particle image velocimetry using a light-field camera. *Optics Express*, 21(2):1726–1740, January 2013.

[19] K.D. Hinsch. Holographic particle image velocimetry. *Measurement Science and Technology*, 13(7):R61–R72, July 2002.

[20] H. Meng, G. Pan, Y. Pu, and S. Woodward. Holographic particle image velocimetry: from film to digital recording. *Measurement Science and Technology*, 15(4):673–685, April 2004.

[21] F. Pereira and M. Gharib. Defocusing digital particle image velocimetry and the three-dimensional characterization of two-phase flows. *Measurement Science and Technology*, 13(5):683–694, May 2002.

[22] T. Fahringer and B. Thurow. 3d particle position reconstruction accuracy in plenoptic piv. In *52nd Aerospace Sciences Meeting*, National Harbor, Maryland, January 2014.

[23] Physclips and UNSW School of Physics. Lenses and images. http://tinyurl.com/mrvkbsf. [Image Online; accessed 27-October-2014].

[24] M. Levoy. Light fields and computational imaging. *IEEE Computer*, 39(8):46–55, August 2006.

[25] R. Ng. Fourier slice photography. *ACM Transactions on Graphics*, 24(3):735–744, July 2005.

[26] C. Hahne, A. Aggoun, S. Haxha, V. Velisavljevic, and J.C.J. Fernández. Light field geometry of a standard plenoptic camera. *Optics Express*, 22(22):26659–26673, November 2014.

[27] Sùss MicroOptics. Microlens arrays. www.suss-microoptics.com/shop. [Online; accessed 13-November-2013].

[28] C. Perwaß and L. Wietzke. Single lens 3d-camera with extended depth-of-field. In *Proc. SPIE 8291, Human Vision and Electronic Imaging XVII*, February 2012.

[29] D.G. Dansereau, O. Pizarro, and Williams S.B. Decoding, calibration and rectification for lenselet-based plenoptic cameras. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1027–1034, June 2013.

[30] L. Condat, B. Forster-Heinlein, and D. Van DeVille. H_2o: Reversible hexagongal-orthogonal grid conversion by 1-d filtering. In *2007 IEEE International Conference on Image Processing*, volume 2, pages 73–76, 2007.

[31] D.G. Dansereau. Light field toolbox for Matlab v0.2. http://tinyurl.com/psyzb52, May 2013. [Online; accessed 30-September-2013].

[32] T. Georgiev. Gallery and lightfield data. http://www.tgeorgiev.net/795.jpg. [Image Online; accessed 28-May-2014].

[33] Lytro. About us. https://www.lytro.com/about/. [Online; accessed 30-October-2014].

[34] Raytrix GmbH. Applications of raytrix cameras. http://tinyurl.com/kfmqynf. [Online; accessed 30-October-2014].

[35] T. Georgiev and C. Intwala. Light field camera design for integral view photography. *Adobe Technical Report*, 2006.

[36] R. Ng. *Digital Light Field Photography*. PhD thesis, Stanford University, July 2006.

[37] C. Stover and E.W. Weisstein. Composition. http://mathworld.wolfram.com/Composition.html. [Online from MathWorld, accessed 15-November-2014].

[38] Z. Yu. Tutorials: Matlab implementation of Fourier slice photography. http://www.eecis.udel.edu/ zyu/, 2013. [Online, accessed 30-September-2014].

[39] N. Patel. lfptools. https://github.com/nrpatel/lfptools, 2013. [Online; accessed 02-June-2014].

[40] J. Kučera. Lytro meltdown. http://tinyurl.com/kscm7zf. [Online; accessed 02-October-2014].

A The Lytro File Format

To properly extract the embedded information in Lytro's raw image files, called lfp files, one needs to understand the specific file format. Kučera [40] has explained the Lytro file format in depth, which will be illustrated in this section. There are two types of lfp files, a raw lfp and a stack lfp. The raw lfp file contains raw sensor data, frame metadata, and private metadata. The stack lfp files contain a depth look up table and one or more pre-rendered images. These file components are assembled in a specific pattern, leading with a header, version number, component length, image tag, the component data, and zero padding.

The header is a constant eight bytes, where the fourth byte determines the component type. The components can be either a package, metadata, or component denoted by **P**, **M**, and **C**, respectively. The package is always the first component in the file and the metadata are second in the file. Any remaining components are tallied after the package and the metadata. After the eight bytes of header, another four bytes, denoted by **VE**, is used for what is most likely the version of the file format and is a big endian integer. The next four bytes give the length of data in the component, denoted by **CL** as a big endian integer. The name of the component given by an image tag of the form sha1-* can be up to 80 bytes long, and is padded with zeros if the image tag length is less than 80 bytes. The remaining bytes of the component section are the actual data which has the length found by **CL**. Each component section is padded with zeros so that the header, version number, component length, image tag, component data, and zero padding is a multiple of sixteen. See Table A.1 for the specific file format of a component in tabular form.

Table A.1: Specific file format of a component in an lfp file where green represents the first eight bytes of the header, blue represents the next eight bytes, yellow represents the eighty bytes of the image tag, white represents the component data, and red represents the zero padding.

‰	L	F	_	0D	0A	1A	0A	VE	VE	VE	VE	CL	CL	CL	CL
s	h	a	1	-	00	00	00	00	00	00	00	00	00	00	00
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
--	--	--	--	--	--	--	--	--	--	--	--	00	00	00	00

B MatLab Code to Extract a Depth Map

```
function[dm] = depthmap(filename)
% Usage: [dm] = depthmap(filename)
% Given the name of a Lytro *-stk.lfp stack file, depthmap reads and parses
% the file to find the depth look up table in the metadata called "depthLUT"
% and returns a depth map in an array called "dm".

% Written 03 October 2014 by Rachel Bierasinski, based on read_lfp.m by
% Douglas H. Kelley and following the file format found in
% http://optics.miloush.net/lytro/Default.aspx

fid = fopen(filename); % *-stk file
header_length = 8; % number of bytes in header
version_length = 4; % number of bytes in versions
sha1_length = 80; % hash length
cl = 4; % number of bytes for component length
line_delim = char(10); % line deliminater
col_delim = ':'; % column deliminater

% -=- Main Header -=-
fread(fid,1,'char'); % FFFD
char(fread(fid,3,'char')'); % LFP
dec2hex(fread(fid,version_length,'char')); % version number
fread(fid,1,'int32','ieee-be');
fread(fid,1,'int32','ieee-be'); % component length

% -=- Read Metadata -=-
LFM=read_metadata(fid);

% -=- Get depth look up table size from metadata -=-
md = LFM.data;
d_loc = strfind(md,'depthLut'); % find "depthLut" in metadata
d = md(d_loc:d_loc+162); % grab whole depthLut section
w_loc = strfind(d,'width'); % find "width" from depthLut
dlines = strfind(d,line_delim);
w_linelims=dlines([ find(dlines<w_loc,1,'last') ...
    find(dlines>w_loc,1,'first') ]); % grab whole line
[~,w]=strtok(d(w_linelims(1):w_linelims(2)),col_delim);
w=str2double(strtrim(strrep(strrep(w,col_delim,''),',',''))); % depthLut
width
h_loc=strfind(d,'height'); % find "height" in depthLut
h_linelims=dlines([ find(dlines<h_loc,1,'last') ...
    find(dlines>h_loc,1,'first') ]); % grab whole line
[~,h]=strtok(d(h_linelims(1):h_linelims(2)),col_delim);
h=str2double(strtrim(strrep(strrep(h,col_delim,''),',',''))); % depthLut
height

%-=- Loop to read remaining sections -=-
secs = struct('type', [], 'typecode', [], 'length', [], 'sha1', [],
'name',...
    [], 'data', []);
N=1;
while ~feof(fid)
    secs(N) = read_section(fid,LFM,h,w);
    if strcmp(secs(N).name,'depth') == 1 % stop once we find depth map
        break
    end
    N = N+1;
end
fclose(fid);

dlut = strcmp({secs.name},'depth');
n = find(dlut==1); % find component labeled depth
dm = secs(n).data; % read the data in this component
dm = reshape(dm,[h w])'; % create the depth map

function LFM = read_metadata(fid)
%function to read the metadata in the stk-lfp file

header_length = 8; % length of header
version_length = 4; % version length
sha1_length = 80; % length of hash
cl = 4; % number of bytes of component length

LFM = struct('type', [], 'typecode', [], 'length', [], 'sha1', [], 'name',...
    [], 'data', []);
fread(fid,1,'char');
LFM.typecode = char(fread(fid,3,'char')'); % LF_
dec2hex(fread(fid,version_length,'char')); % version number
fread(fid,1,'int32','ieee-be');
LFM.length = fread(fid,1,'int32','ieee-be'); % component length
LFM.sha1 = char(fread(fid,sha1_length,'char')'); % Image hash
if strcmp(LFM.typecode,'LFM')
    LFM.name = 'metadata';
    LFM.data = char(fread(fid,LFM.length,'char')');
    LFM.type = 'stk-lfp.json';
else
    error('Sorry, cannot find metadata')
end
num = (LFM.length+version_length+header_length+sha1_length+cl);
if rem(num,16)~=0
    num = floor(((num/16)+1))*16-
(LFM.length+version_length+header_length+sha1_length+cl);
else
    num = floor(num)-
(LFM.length+version_length+header_length+sha1_length+cl);
end
fseek(fid,num,'cof'); % skip padding


function sec = read_section(fid,TOC,LUT_width,LUT_height)
%function slightly modified from the read_section function in read_lfp to
%read remaining sections

header_length = 8; % length of header
version_length = 4; % version length
sha1_length = 80; % length of hash
cl = 4; % number of bytes for component length
line_delim = char(10); % linefeed
col_delim = ':'; % separates names from columns in json data
jpegmagic = hex2dec({'FF', 'D8', 'FF', 'E0', '00', '10', '4A', '46', '49',
'46'})'; % magic number for jpegs
```

```
sec = struct('type', [], 'typecode', [], 'length', [], 'sha1', [], 'name',...
    [], 'data', []);

fread(fid,1,'char');
sec.typecode = char(fread(fid,3,'char')'); % LF_
dec2hex(fread(fid,version_length,'char')); % version number
fread(fid,1,'int32','ieee-be');
sec.length = fread(fid,1,'int32','ieee-be'); % component length
sec.sha1 = char(fread(fid,sha1_length,'char')'); % Image hash

if exist('TOC','var') && ~isempty(TOC) % if we have a TOC, use it for naming
    sha1loc=strfind(TOC.data,sec.sha1(1:45)); % find section sha1 in TOC
    TOClines=strfind(TOC.data,line_delim); % where the lines end
    linelims=TOClines([ find(TOClines<sha1loc,1,'last') ...
        find(TOClines>sha1loc,1,'first') ]); % find the whole line
    sec.name=strrep(strtrim(...
        strtok(TOC.data(linelims(1):linelims(2)),col_delim)),'"',''); % grab
the name
end
if strcmp(sec.name,'imageRef')
    if sec.length==LUT_height*LUT_width*4 % Test for Lytro depth map
        sec.type='depthLut';
        sec.name='depth';
        sec.data=fread(fid,sec.length/4,'float32')';
    else sec.length~=LUT_height*LUT_width*4;
        sec.data=fread(fid,sec.length*2/3,'ubit12=>uint16','ieee-be')'; %
read an image if named imageRef
    end
else
    sec.data=char(fread(fid,sec.length,'char')'); % otherwise read text
end
if sec.length > numel(jpegmagic) && ... % Check for the magic bytes to see if
its a jpg
        all(jpegmagic==double(sec.data(1:numel(jpegmagic))))
    sec.type='LFP_JPEG';
    sec.name='image';
end
if strcmp(sec.name,'blockOfImagesRef') % Check for h264 block
    sec.type='LFP_BLOCK_OF_IMAGES';
end
if ~strcmp(sec.name,'imageRef') % Assume anything else that isn't called
imageRef is plain text json
    sec.type='LFP_JSON';
end
num = (sec.length+version_length+header_length+sha1_length+cl);
if rem(num,16)~=0
    num = floor(((num/16)+1))*16-
(sec.length+version_length+header_length+sha1_length+cl);
else
    num = floor(num)-
(sec.length+version_length+header_length+sha1_length+cl);
end
fseek(fid,num,'cof'); % skip padding
```

C Decoding and Rectifying a Lytro Camera

Source code called Light Field Toolbox that is available for download on MatLab's website [31] created by Dansereau based on *Decoding, Calibration, and Rectification for Lenselet-Based Plenoptic Cameras* [29]. This toolbox was used with the Lytro camera in this research to decode the images as well as to remove lens distortions through calibration. First, white images are pre-processed to find the centers of each microlens and to build a white image database. Figure C.1 shows an example of the centers found in a white image of the Lytro camera used in this research.

A 19 $\times$ 19 grid with 3.6 mm spacing and imaged with ten diverse poses was used during the calibration process that completes corner identification as shown in Figure C.2, including parameter initialization, parameter optimization without lens distortion, then with lens distortion, and a final stage of parameter refinement [31]. The calibration process from the Light Field Toolbox results in a lenslet grid model, a plenoptic intrinsic model, and distortion parameters. The lenslet grid model contains information on the rotation, spacing and offset of the lenslet images on the sensor. The plenoptic intrinsic model is a 5$\times$5 matrix that relates pixel index to an undistorted ray. Finally, the distortion parameters include information on radial

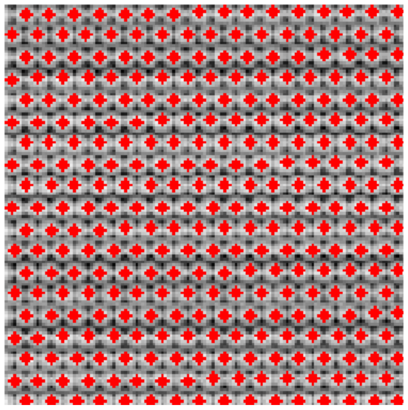


Figure C.1: The centers, shown in red, of microlenses in a white image from a Lytro camera.

distortion in the ray direction.

The calibration process and the calibration results from the Light Field Toolbox are further explained in the tutorial included with the toolbox [31]. Further detail is also provided in *Decoding, Calibration, and Rectification for Lenselet-Based Plenoptic Cameras* by Dansereau [29]. After the calibration is completed, the light field images taken can be decoded with rectification to remove any lens distortions. The results of image rectification are shown in Figure C.3, where Figure C.3(a) shows the original image and Figure C.3(b) shows the same image after lens distortions are removed through rectification. It is clear the rectification improves the image by removing the warping around the edges of the image, which is clearly visible in the unrectified image Figure C.3(a).

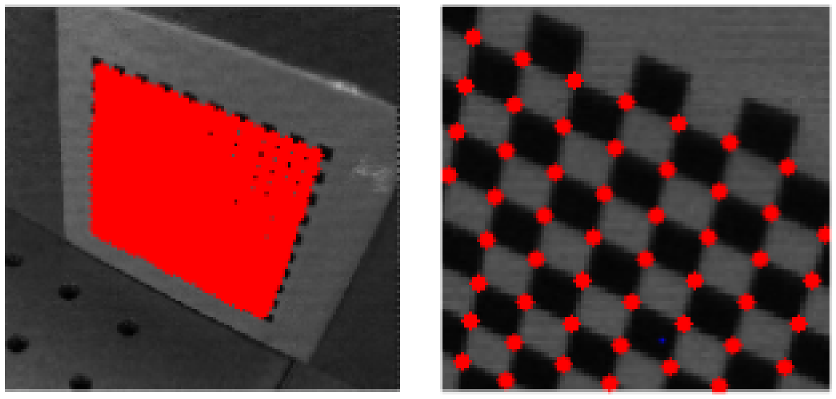


Figure C.2: Results of the checkerboard corner finding step completed by the Light Field Toolbox. The image on the right is an enlargement of the image on the left.

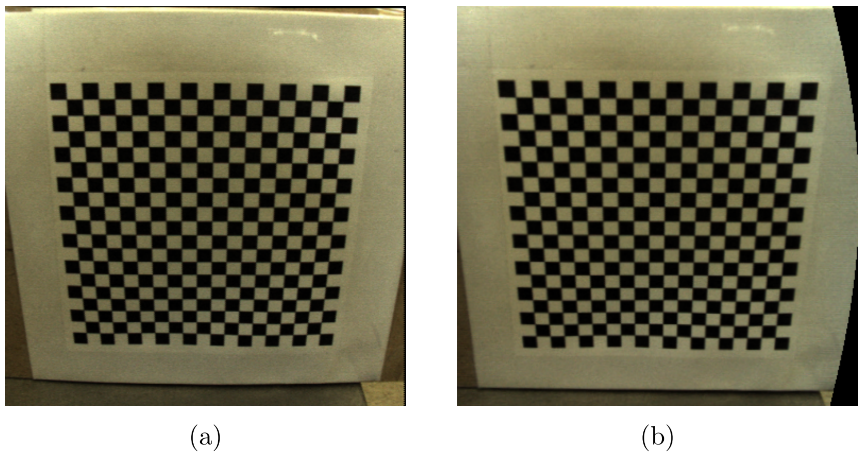


Figure C.3: Figure C.3(a) is an unrectified image of the calibration grid prior to any calibration and Figure C.3(b) is a rectified image after camera calibration to remove lens distortions. Figure C.3(b) shows visible signs of improvement from the original especially on the edges where the warping is removed.