SIMULATION OF PHASE MEASURING DEFLECTOMETRY OF FREEFORM SURFACES

by

Chao-Hsiung Tseng

A Thesis Submitted to the Faculty of the

JAMES C. WYANT COLLEGE OF OPTICAL SCIENCES

In Partial Fulfillment of the Requirements

For the Degree of

MASTER OF SCIENCE

In the Graduate College

THE UNIVERSITY OF ARIZONA

2019

THE UNIVERSITY OF ARIZONA
GRADUATE COLLEGE

As members of the Master's Committee, we certify that we have read the thesis prepared by *Chao-Hsiung Tseng*, titled *Simulation of Phase Measuring Deflectometry of Freeform Surfaces* and recommend that it be accepted as fulfilling the dissertation requirement for the Master's Degree.

James Schwiegerling Date: 6/21/2019

Daewook Kim Date: 6/21/2019

Rongguang Liang Date: 06/21/2019

Final approval and acceptance of this thesis is contingent upon the candidate's submission of the final copies of the thesis to the Graduate College.

I hereby certify that I have read this thesis prepared under my direction and recommend that it be accepted as fulfilling the Master's requirement.

James Schwiegerling Date: 6/21/2019
Master's Thesis Committee Chair
James C. Wyant College of Optical Sciences

DEDICATION

To my parents, Ying-chung and Mei-hui, for their love and support through the completion of my masters.

Table of Contents

List of Figures

Abstract

Optical three-dimensional shape metrology has become a key technology in scientific and industrial applications. Phase Measuring Deflectometry (PMD)is one optical three-dimensional shape metrology technique which is based on two-dimensional fringe patterns measurements for specular reflecting surfaces.

There are several configurations of PMD to measure the arbitrary specular surfaces. Here, a single camera is used to capture the reflected image of a single LCD monitor to construct the deflectometry system. Distance laser sensors, multiple cameras, and multiple monitors will not be considered here. This investigation focuses on creating simulated PMD images for an arbitrary specular surface. Such images are useful for testing slope calculations and surface reconstruction algorithms. System geometry calibration and an inverse ray-tracing algorithm are explored.

This thesis demonstrates the preliminary results of PMD for a flat mirror, a concave mirror and a freeform surface with the phase shifting method. The specific feature of the image simulation shows the inverse ray-tracing can deduce the captured image correctly. Included is a discussion about the ambiguity of fringe numbers and the uncertainty of the phase value calculation with insufficient fringe sampling.

Chapter 1: Introduction

1.1 Metrology methods for optical surface measurement

The stringent requirements of optical manufacturing make optical fabrication and metrology technologies progress rapidly. Nowadays, optics with high precision fabrication can be finished by the computer controlled figuring with feedback from optical metrology. The correction for subtle surface variation in the polishing process highly relies on precise metrology techniques.

Optical metrology for surfaces can be categorized as contact profilometry and contact-free profilometry. Contact profilometry (Song & Vorburger, 1991) is an older but more accepted method to measure an arbitrary surface profile. A mechanical tip is dragged along the surface in this method, and the tip deflections are measured by using mechanical, electrical, or optical transducers. Contact profilometry can measure to the atomic scale with an atomic force microscope as contact stylus. However, the relatively long measuring time and potential for damage of the testing surface by the stylus tip are the main drawbacks associated with this technique.

Optical probing profilometry is an example of a non-contact measuring technique that uses an optical probe to map surface topography by sensing the best focus position on the testing objects. Confocal microscopy is a common example of optical probing topography. The technique uses a spatial filter at the confocal plane of a microscope objective to increase the signal to noise ratio of images with height distribution determination for the testing surface. Maximum signal occurs when the surface is conjugate to the spatial filter. The slow speed of data acquisition due to the single point detection still limits the improvement of this method.

For non-contact optical profilometry, interferometry is the most common and widely used non-contact method in optical metrology. Interferometry can quickly measure broad regions of the test part, so it avoids the speed issues associated with the previously described techniques. However, there are some limitations on the application of interferometry. Interferometry detects the optical path difference between the reference surface and testing surface to obtain a high accuracy profile measurement. Fizeau interferometry is one of the most common configurations. The common path of the reference beam and testing object beam is the advantage of Fizeau interferometry to reduce the influences from system vibration. However, the requirement of specific null reference surface for measuring freeform optics, like a computer generated hologram (Frecher, 1976) (Cai, Zhou, Zhao, & Burge, 2013), is necessary and expensive.

White light interferometry is also the other choice for evaluating the interference intensity profile via assessing the temporal coherence of a light source and vertical scanning (Wyant, 2002). The basic concept of the white light interferometer is measuring the sum of all the fringe intensities and the broadband spectrum of the white light source can ensure the precise profile measurement. However, the measurement time consuming of the white light interferometer is still a big issue due to the lateral scanning for the testing object.

Deflectometry is a non-contact profilometry for measuring the specular optical surface that will be explored in more detail below. The principle of deflectometric techniques is detecting the lateral displacement of reflected light from a testing object to obtain the slope information of the testing profile. The original deflectometry technique is the Foucault Knife Edge Test in 1858 (Malacara, Foucault, Wire, and Phase Modulation Tests, 2007). The knife edge creates a shadow pattern that is analyzed to understand the topography of the test object. There are several related versions of the Foucault Knife Edge Test, which include the Wire Test, the Ronchi Test (Mansuripur, 1997) (Malacara, Ronchi Test, 2007) and the Hartmann Test (Malacara, Hartmann, Hartmann-Shack, and Other Screen Tests, 2007). These tests are actually replacing the knife edge as a thin wire and a binary grating in the Wire Test and Ronchi Test respectively. The Hartmann Test, introduced by Johannes Hartmann in 1900, is using a point light source and grid mask to sampling surface at one time. An improvement of the Hartmann Test which replaces the grid mask with a lenslet array at the pupil plane is called Shark-Hartmann Test (Platt & Shack, 2001) (Neal, Copland, & Neal, 2002).

PMD is a new deflectometric method developed in 2004 (Knauer, Kaminski, & Hausler, 2004). The basic principle of PMD is using a digital camera to capture the specular reflection of a testing object with a structured light source from an LCD monitor.Details of this method are discussed in detail in the next section.

1.2Phase Measuring Deflectometry

Phase Measuring Deflectometry(PMD) is one of the optical three-dimensional shape metrology techniques based on two-dimensional fringe phase measurement, especially for specular reflecting surfaces (Werling, Mai, Heizmann, & Beyerer, 2009). The fundamental principle of PMD is the law of reflection. Figure 1 shows the scheme of PMD. Since there are many possible height and slope combinations to explain the phase point observed by a single camera, the reflected light from the testing object in a well-measured system needs to be traced, then the vectors of the surface normal are determined to solve the ambiguity between slope and height. The topography of the testing surface is reconstructed from the measured slopes with numerical calculations, and iterative height reconstructions can achieve the self-consistent shape results (Olesch, Faber, & Hausler, 2011).

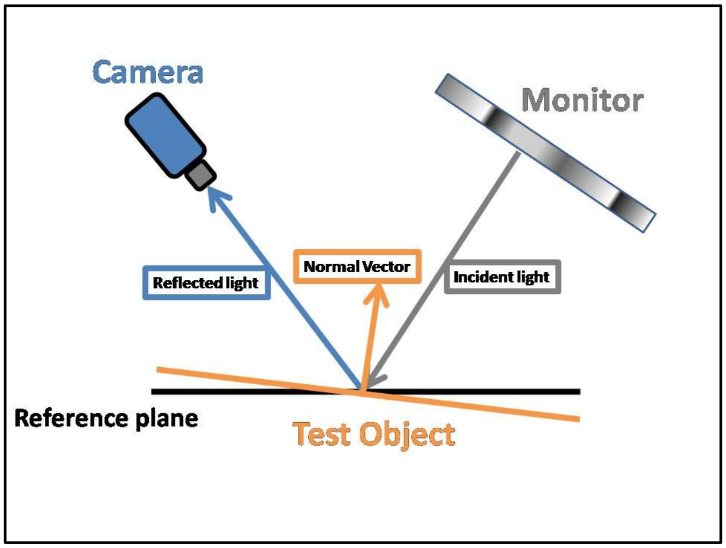

Figure 1 The measurement principle of the PMD is based on the law of reflection.

The followings are the steps of standard measurement of PMD.

(1) Set up the camera and monitor properly to ensure the field of view can cover the region of interest for a test object, adjust or tip/tilt so that the reflection on the captured image of fringe patterns from the monitor can be recorded.

(2) Capture the phase shifted fringe images from the testing object.

(3) Analyze the fringe patterns with phase unwrapping to retrieve the phase map and calculate the distribution of the slopes in x and y-direction

(4) Reconstruct the height distribution from the slopes information with a two-dimensional integration process.

The basic setup of PMD includes a digital camera and an LCD monitor with computer-generated fringe patterns. The camera records the fringe patterns from the monitor after reflection from a specular test surface. The shape of the test object is reconstructed by solving the inverse ray-tracing problem from the captured images. For different requirements in the application of PMD, the system may also use additional monitors, cameras or distance sensors. Figure 2 shows the additional patterns in the optical path of configuration (Huang, Idir, Zuo, & Asundi, 2018). In Figure 2(a), the reflected ray is determined by the two intersection points with shifting the monitor (Petz & Tutsch, 2003). In Figure 2(b), the configurations use an additional distance sensor to measure the reference distance and solve the ill-posed problem (Li, Sandner, Gesierich, & Burke, 2012). Figure 2(c) and (d) shows the multiple cameras and monitors, which are used to reduce the discrepancies of the calculation of the normal vectors for the test object (Knauer, Kaminski, & Hausler, 2004).

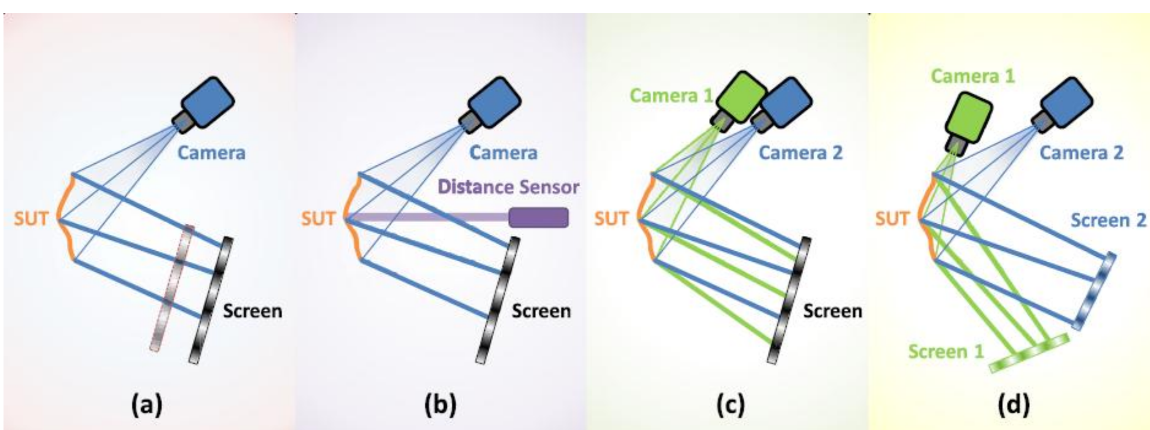

Figure 2 Some other types PMD setups: (a) monoscopic PMD with shifted screens,(b) monoscopic PMD with a point distance sensor,(c) stereoscopic PMD, and(d) multi-camera PMD with several screens serving different camera. (Huang, Idir, Zuo, & Asundi, 2018)

1.3Challenges in Deflectometry

The precise measurement of the deflectometry configuration plays an important role in PMD. Measurement of the absolute position of the illumination light source, an observation point, and the test optic is required to high accuracy to minimize the uncertainty of the slope calculation. The low-order aberrations like defocus, astigmatism and coma is easily generated via geometry bias.

Camera distortion as a mapping of the captured image in the real-world coordinate system also needs to be carefully considered. The viewing perspective is a further issue when mapping the captured image in the deflectometry system. Mapping correction is required in the data processing to avoid a projection error. For the system geometry,low-order aberrations affect the image mapping more than high-order aberrations.

The system calibration is the most important part of the deflectometry process to ensure reliable measurement results. There are different calibration methods to improve the accuracy of measurements in deflectometry. Due to the limitation of lab equipment, the possible method using a distance sensor is excluded. The calibration method is discussed further in Chapter 3.

Chapter 2: Principle of Phase Measuring Deflectometry

2.1 Fringe Images on Monitor

The fringes created for phase information calculation consist of sinusoidal patterns with a single frequency. To obtain two-dimensional slope information, sinusoidal fringe patterns in both the X and Y directions are projected. The determination of frequencies for the sinusoidal pattern is dependent on whether the two adjacent fringes in the reflected image can be distinguished. Since the radius of curvature for testing objects are different, the frequencies also vary with different directions. Table 1 shows the values of frequencies for 4 different type testing objects.

Figure 3 and Figure 4 shows the sinusoidal fringe patterns used in the experiments. To know the incoming source position for each pixel in images, the corresponding pixel with the same phase value on the monitor needs to be found. To derive the position of the corresponding monitor pixel, the phase difference per pixel is calculated. The total phase of the sinusoidal patterns in x and y-directions are determined and this value is divided by the number of pixels of the monitor. Figure 5 and Figure 6 shows the phase information in x and y-direction.

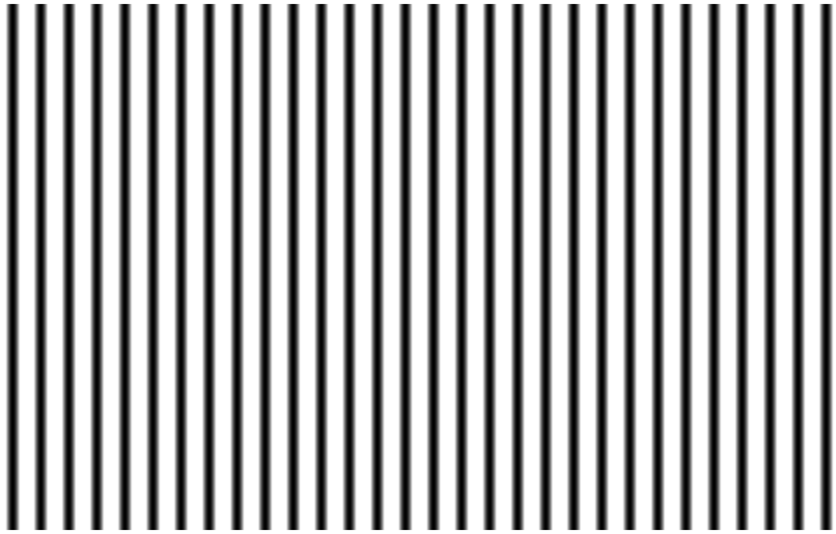

Figure 3 The sinusoidal fringe patterns in the x-axis. The exact number of fringes is dependent on the resolution of fringe patterns on the testing objects.

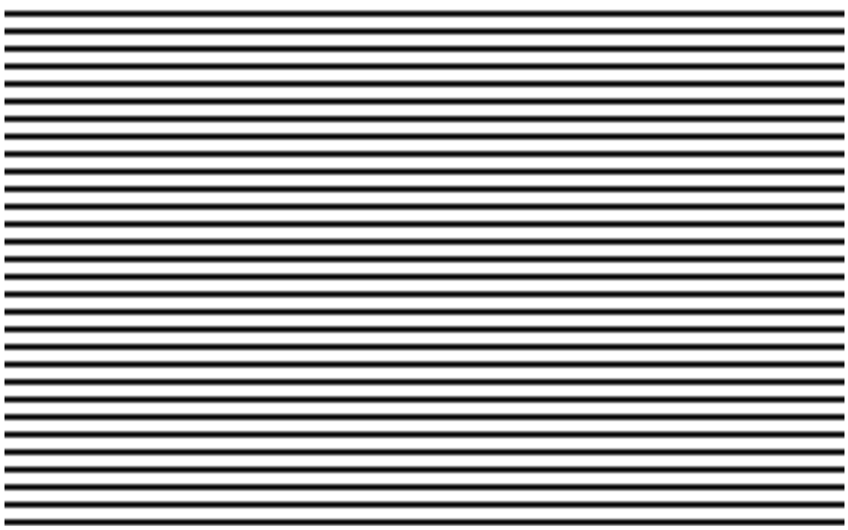

Figure 4 The sinusoidal fringe patterns in the y-axis. The exact number of fringes is dependent on the resolution of fringe patterns on the testing objects.

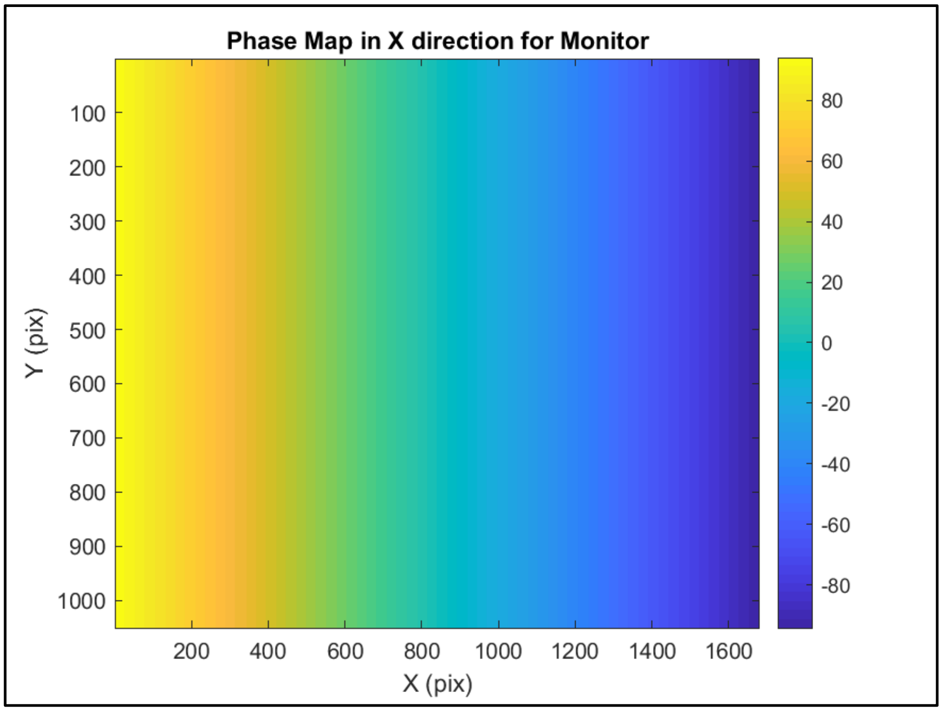

Figure 5 The phase map in the x-direction for Monitor after phase unwrapping. Noticed that the x and y-axis are the monitor pixel in x and y-direction respectively.

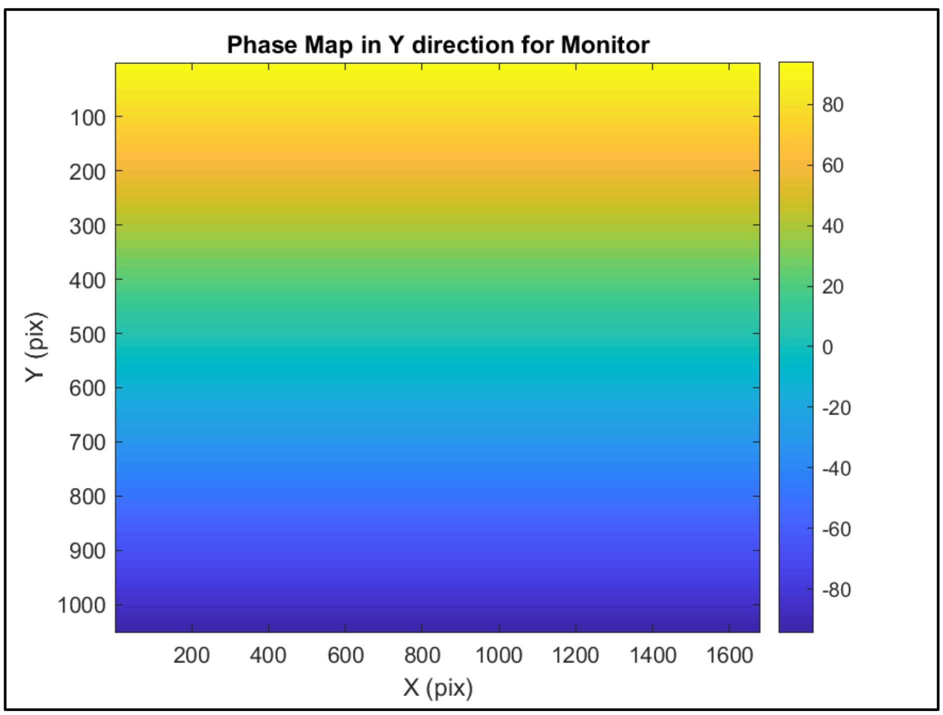

Figure 6 The phase map in y-direction for Monitor after phase unwrapping. Noticed that the x and y-axis are the monitor pixel in x and y-direction respectively.

2.2 Phase Shifting and Unwrapping

The basic idea of PMDis to use the phase shifting method to detect the test object's profile information, and then use phase unwrapping to restore the phase information and match the same phase value of the corresponding pixel on the monitor. Figure 7 and Figure 8 shows the captured image taken from the flat mirror with x and y-direction fringe pattern respectively.

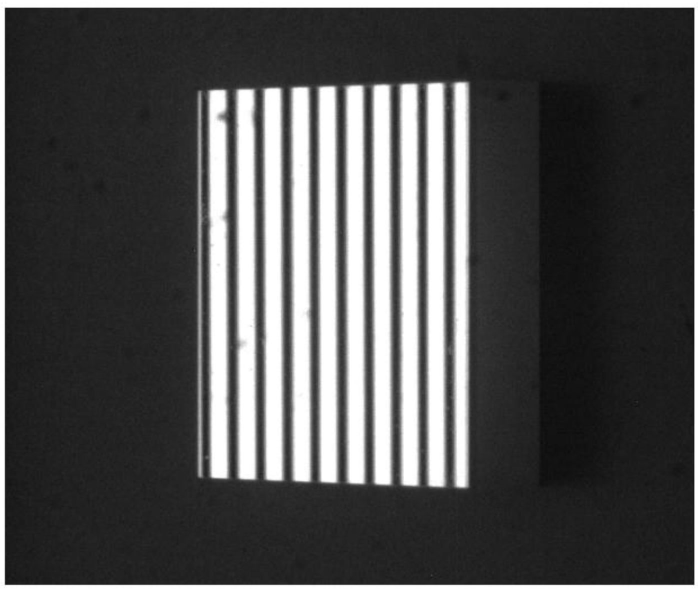

Figure 7 The captured image of the flat mirror as a testing object with x-direction fringe pattern on the monitor.

The crossed fringe intensity I(x,y) from Figure 7 and 8 can be expressed as

$$I_n(x,y) = a(x,y) + b(x,y)\cos[\varphi(x,y) + \frac{2n\pi}{N}], n = 1,2,\dots, N-1$$

where x and y are the orthogonal coordinates of the screen, $a(x,y)$ is the background, $b(x,y)$is the modulation, and $\varphi(x,y)$ is the phase of x and y directional sinusoidal fringes, respectively.The wrapped phase $\varphi^{\omega}(x,y)$ can be calculated as

$$\varphi^{\omega}(x,y) = -\arctan\frac{\sum_{n=0}^{N-1} I_n \sin(\frac{2n\pi}{N})}{\sum_{n=0}^{N-1} I_n \cos(\frac{2n\pi}{N})}$$

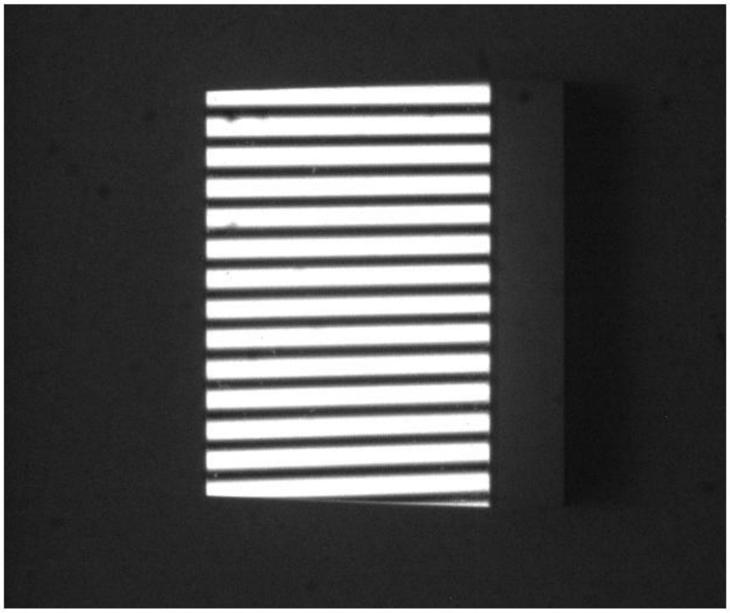

Figure 8 The captured image of the flat mirror as a testing object with y-direction fringe pattern on the monitor.

Typically, the phase shifting method uses three or four steps in shifting to collect the phase difference information. However, the small number of steps can cause phase error and non-smooth phase map. One way to solve this sampling problem is by increasing the number of shifting steps. Here, eight shift steps with a π/4 difference is used to avoid the sampling issue.

After the fringe demodulation, the phase values are wrapped within[-π, π]. To calculate slopes from these measurements, these wrapped phases need to be unwrapped to absolute phases. Here, a two-dimensional phase unwrapping algorithm (Ghiglia & Romero, 1994)is used to restore the phase information. This algorithm deals with the area out of the region of interest by solving the weighted unwrapping problem. The required phase values in PMD are absolute phases, and the one to one mapping could be constructed between captured image and the monitor projection.

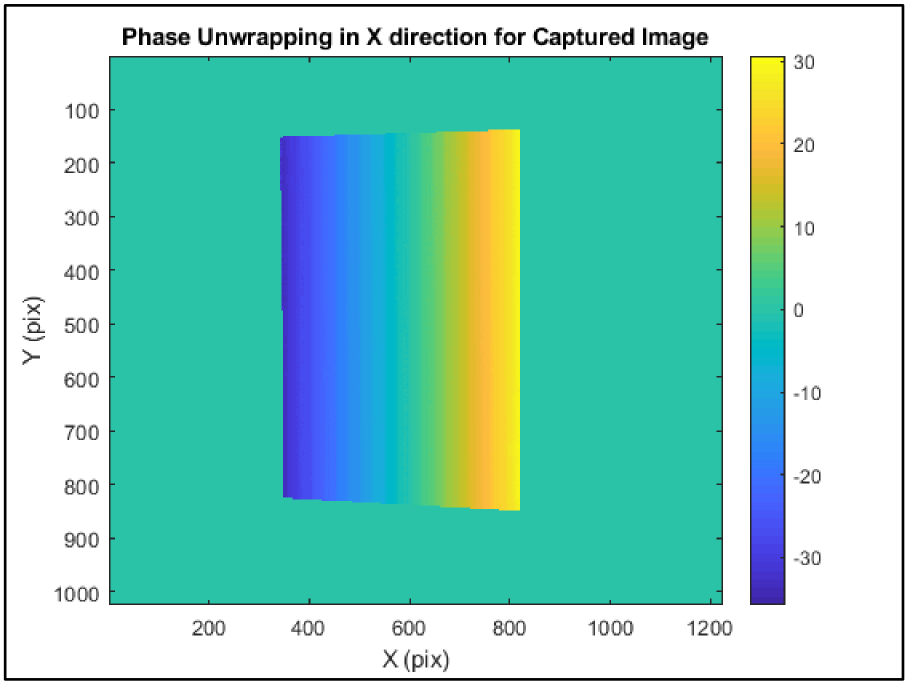

Figure 9 The phase map in the x-direction of the captured image for a flat mirror after phase unwrapping.

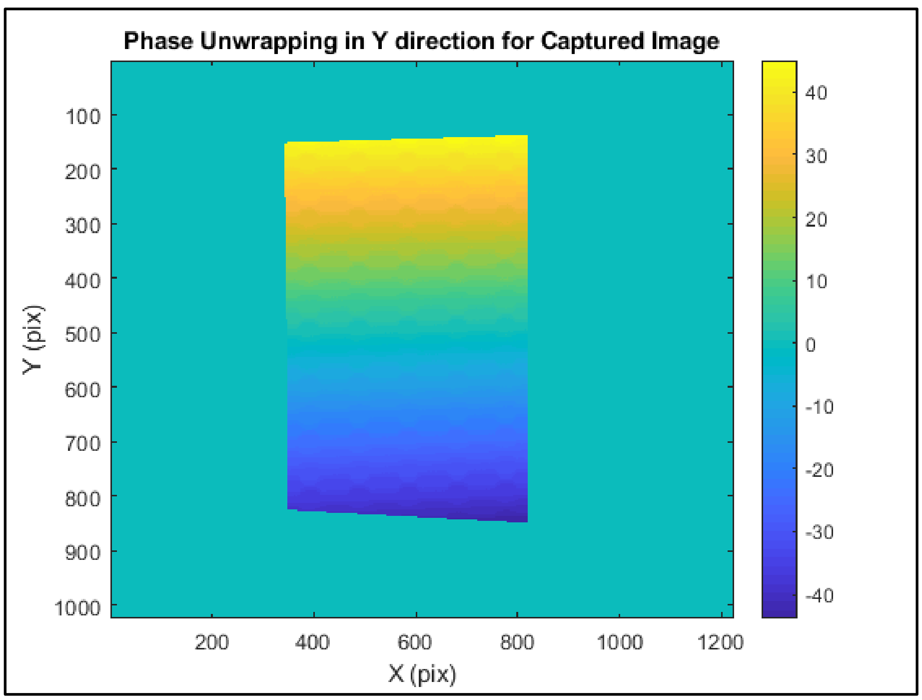

Figure 10 The phase map in the y-direction of the captured image for the flat mirror after phase unwrapping.

To construct the one to one phase mapping correctly a marker fiducial to flag the relative positions of a specific pixel on the testing object and monitor. This technique is discussed in detail in Chapter 3. Figure 11 and Figure 12 shows the phase unwrapping results in x and y-direction, respectively. Due to the phase unwrapping algorithm cannot exclude out of the region, the area out of flat mirror still shows phase value in phase map, but it would not affect us to retrieve the absolute phase value.

2.3 Slope Calculation

After obtaining the absolute phase values from phase unwrapping, the corresponding pixel on the monitor can be determined since the period of the fringe pattern on the monitor is known. As mentioned in Section 1, the phase difference per pixel can be calculated if geometrical system measurement is trusted. Once the angle between the monitor and the reference plane, where the test objects is located, is measured, then the position of the corresponding monitor pixel for each captured image pixel with the phase-marked intersection point can be derived. Again, this is discussed in detail in Chapter 3. The connections between these two pixels are the incident light vectors, which are used to determine the normal vectors of testing objects.

For the reflected light, we corrected the keystone effect and derived the reflected light vector via the assumption of the pinhole camera model. We will discuss the keystone effect correction in detail in Chapter 3.

Once we obtained the incident light vector (denoted asV_i) and reflected light vector (denoted asV_r), we normalized V_i and V_r:

$$v_i = \frac{V_i}{\|V_i\|}, \text{ and } v_r = \frac{V_r}{\|V_r\|}$$

According to the reflected law, the testing surface normal vectors N can be calculated by

$$N = -(v_i + v_r) = \begin{pmatrix} N_x \\ N_y \\ N_z \end{pmatrix}$$

where N_x, N_y, and N_z are the x, y and z components of the surface normal N. The measured surface x and y slopes (S_x,S_y) are therefore calculated as

$$S_x = -\frac{N_x}{N_z}$$

$$S_y = -\frac{N_y}{N_z}$$

2.4 Shape Reconstruction

Once the slopes (S_x,S_y) and in-plane coordinates (x,y) are calculated, the shape can be reconstruction from these gradient data. The height distribution z is reconstructed from the calculated coordinates (x,y) and slopes (S_x,S_y). We can express the two-dimensional integration process as

$$z = f(x, y, S_x, S_y)$$

where f stands for the two-dimensional function. There are mainly three types of reconstruction methods: Zonal reconstruction (Fried, 1977), Modal reconstruction (Dai, 1996) and Piecewise reconstruction (Ettl, Kaminski, Knauer, & Hausler, 2008). Here we use the Modal reconstruction method to restore the surface profile. This reconstruction method is based on analytical models. Via taking the first derivatives in x and y-direction of the analytical expressions, the coefficients of polynomials can be approximated by fitting the measured slopes with analytical slopes. There are several used models include Zernike, Chebyshev, and Legendre polynomials. When the coefficients of polynomials are estimated from the slope fitting, the height distribution of the surface profile can be calculated by using the coefficients from the slope fitting.

Here we used the non-rotational polynomial with fourth-order as our fitting surface. That is, for a location on the surface defined by coordinates x and, the height above a reference plane is by

$$z = \sum_{i,j}^{i+j\leq 4} a_{ij} x^i y^j$$

There are several ways to obtain the converge result with fitting parameters, Dr. Kim's group provided a novel model-free iterative data-processing approach to improves the accuracy of surface reconstruction (Graves, Choi, Zhao, & Kim, 2018). The initial heigth distribution of test surface is assuming a flat plane (the reference plane). Once the first estimate is obtained from surface reconstruction, the estimate of height distribution was taken to be the initial height distribution, and re-calculated the x and y slopes again in iterative approach to obtain convergent fitting parameters. Figure 11 and Figure 12 show the results of surface reconstruction with one and five iterations respectively. However, it does not show significant change of the fitting shape.

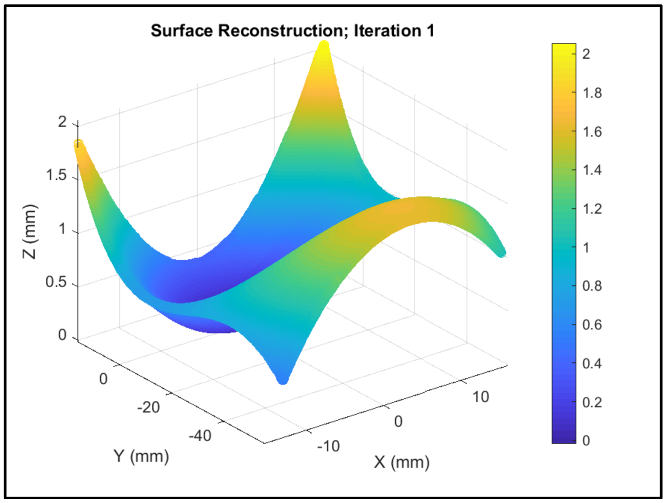

Figure 11 The shape reconstruction with one iteration.

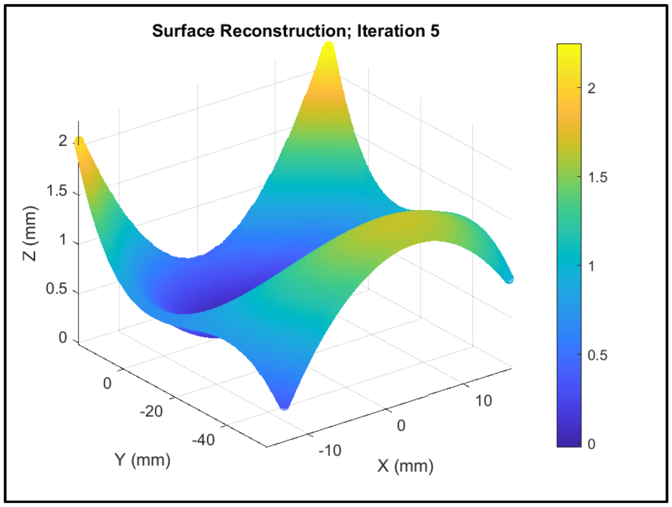

Figure 12 The shape reconstruction with five iterations.

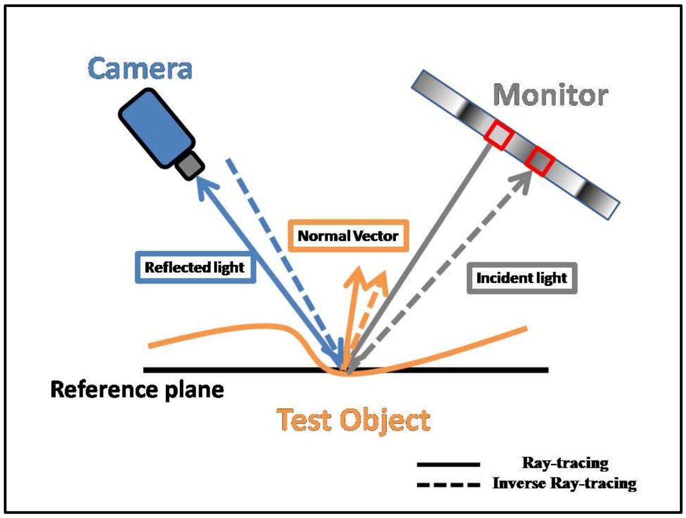

Figure 13 The scheme of ray-tracing and inverse ray-tracing in PMD.

2.5 Image Simulation

Once the converged fitting parameters have been calculated, the image can be simulated by fitting height distribution by reverse reflected vector calculation with system geometrical parameters. The reverse vectors calculation is re-calculating the reflected vectors via Snell's law for final height information from fitting result. Figure 11 shows the scheme of ray-tracing and inverse ray-tracing in PMD.

The first step of the reverse ray-tracing calculation is starting from the incident light vectors. Since the real world coordinates of camera are the same, the incident light vectors only change by the different values of test object's height, which is the result of the polynomial fitting. The real world coordinates for new test object coordinate in (x,y,z) is:

$$x = \text{x value of test object after re} - \text{projection}$$
$$y = \text{y value of test object after re} - \text{projection}$$
$$z = \text{4th order polynomial fitting result for each x and y}$$

The subtraction between camera and new test object contribute to the new incident light vector. The next step is re-calculating the normal vectors of test object surface for the new height distribution of new test object surface. The normal vector of a surface can be obtained from the gradient of the surface. The real world coordinates for new normal vector coordinate of test object surface in (x,y,z) is:

$$x = \text{derivatives of test surface in X direction}$$
$$y = \text{derivatives of test surface in Y direction}$$

$$z = 1$$

Note that z value always equal to 1 due to the derivatives of two-dimensional surface in Z direction is always 1. Once new normal vectors have been calculated, the new reflected light vectors also can be calculated via Snell's law:

$$v_r = 2 * N - v_i$$

where all three vectors are normalized. As the new reflected light vectors have been obtained, the next step would be calculating new intersection points between new reflected light vectors and LCD monitor. Before calculating the intersection points, the equation of LCD monitor must be written. The formula of a plane in Cartesian coordinates is:

$$Ax + By + Cz + D = 0$$

where the (A, B, C) is the normal vector components in x, y, and z directions. D is the constant satisfied a known point $r_p = (x_p, y_p, z_p)$ in this plane: $D = -Ax_P - By_p - Cz_p$. To express the equation of the LCD monitor in this case, the parameters of system geometry have to be used. The followings are the values of plane equation's parameters B, C and D for LCD monitor:

$$RA = \text{angle between refer plane to monitor} - \text{angle between camera and normal vector of refer plane}$$

$$B = -y_p * \cos(RA) + z_p * \sin(RA)$$

$$C = -y_p * \cos(RA) - z_p * \sin(RA)$$

$$D = -B * y_p - C * z_p$$

The definition of $r_p = (x_p, y_p, z_p)$ will be shown in Chapter 3. As the equation of LCD monitor has been defined, the intersection points of reflected light vectors can be derived. Consider a starting point $r_0 = (x_0, y_0, z_0)$ in three-dimensional space and a propagation direction $r_d = (x_d, y_d, z_d)$, where the r_d has been normalized to a unit vector, then the propagation equation (here is the reflected light vector) at magnitude t is written as

$$r(t) = r_0 + t * r_d$$

Solving the t value with the plane equation of monitor and propagation equation, the answer can be derived as following (Schwiegerling, 2019):

$$t = \frac{-(Ax_0 + By_0 + Cz_0 + D)}{Ax_d + By_d + Cz_d}$$

where the starting point $r_0 = (x_0, y_0, z_0)$ in this case is the point on new test object and the propagation direction $r_d = (x_d, y_d, z_d)$ is the reflected light vector. After the magnitude t has been solved, the intersection point on the monitor can be obtained.

The last step would be calculating the corresponding intensity value of the intersection point on the monitor. Since the period of sinusoidal fringe patterns is known, the corresponding intensity value can be derived from the following equation:

$$\text{intensity in X direction} = 1 + \cos(2 * \pi * (\text{intersection point in X direction}) * \text{fringe numbers in X})$$

$$\text{intensity in Y direction} = 1 + \cos(2 * \pi * (\text{intersection point in Y direction}) * \text{fringe numbers in Y})$$

Once the intensity in X and Y directions have been calculated, the image simulation has been finished. The image simulation results are presented in Chapter 4.

Chapter 3: System GeometryMeasurement and Calibration

3.1 Keystone Effect Correction

The keystone effect, also known as the tombstone effect, is caused by attempting to project an image onto a surface at an angle. It is a distortion of the image dimensions, making it look like an architectural keystone. In this case, the distortion suffered by the image depends on the angle of the camera to the reference plane. To correct the keystone effect, a checkerboard calibration grid is used and then the grid in x and y directions is fit by a linear function. To mark the corner of the checkerboard, the toolbox "Computer Vision" in MATLAB is used to detect the corners and to make the calibration grid. Figure 13 shows the checkerboard after detecting the corners and making the calibration grid. Figure 14 shows the grid distribution in the x-direction with linear fitting. Figure 15 shows the grid distribution in the y-direction with linear fitting.

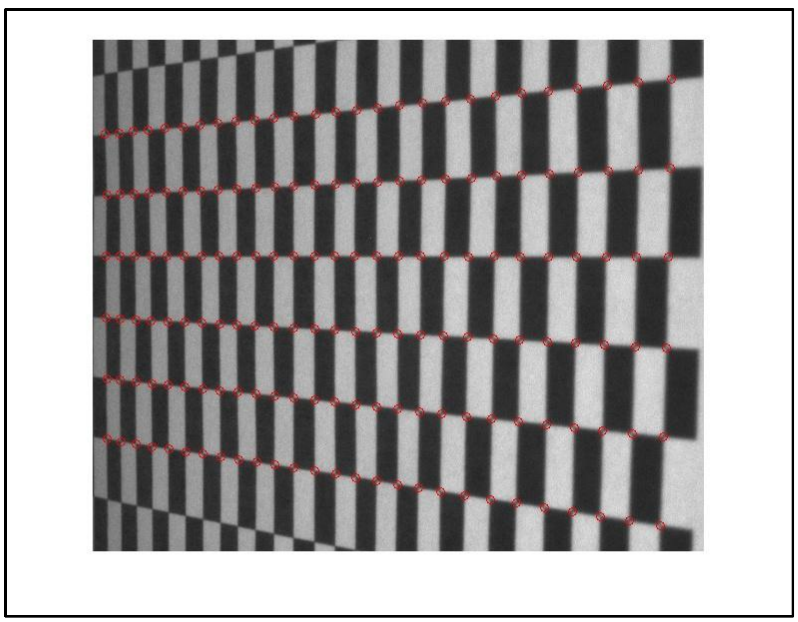

Figure 14 The calibration checkerboard and detected corner points with the red circle. It's been finished by 'detectCheckerboardPoints' function in MATLAB 2018.

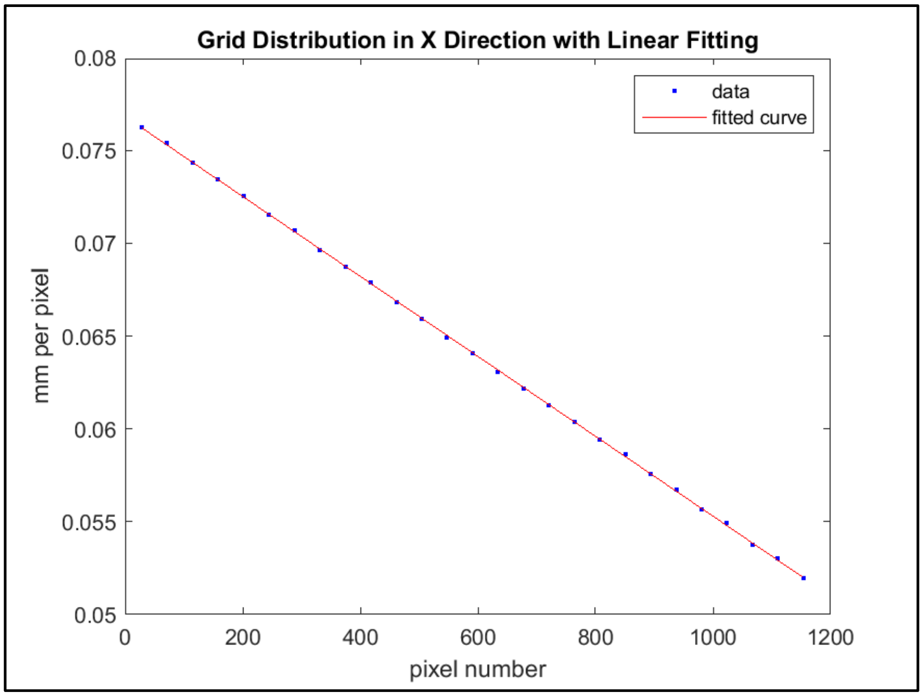

Figure 15 The grid distribution in the x-direction with linear fitting.

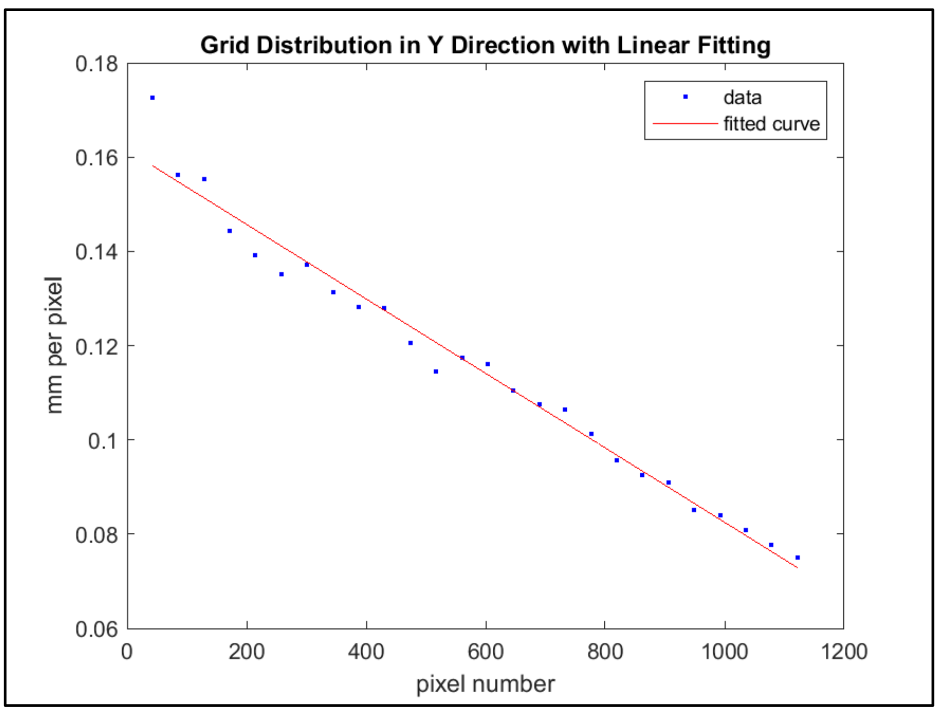

Figure 16 The grid distribution in the y-direction with linear fitting.

The re-projection of the captured image is converting the trapezoid to a rectangle shape. Figure 16 shows an example of a trapezoid area in the captured image and shows its re-projection in Figure 17, which is a rectangle shape. In Figure16, the size of checkerboard in calibration is 4.75 mm squares. The blurred part in Figure 16 is due to that area being out of focus. The camera lens has a limited depth of focus, and it makes partial area out of focus as the coverage area is too big.

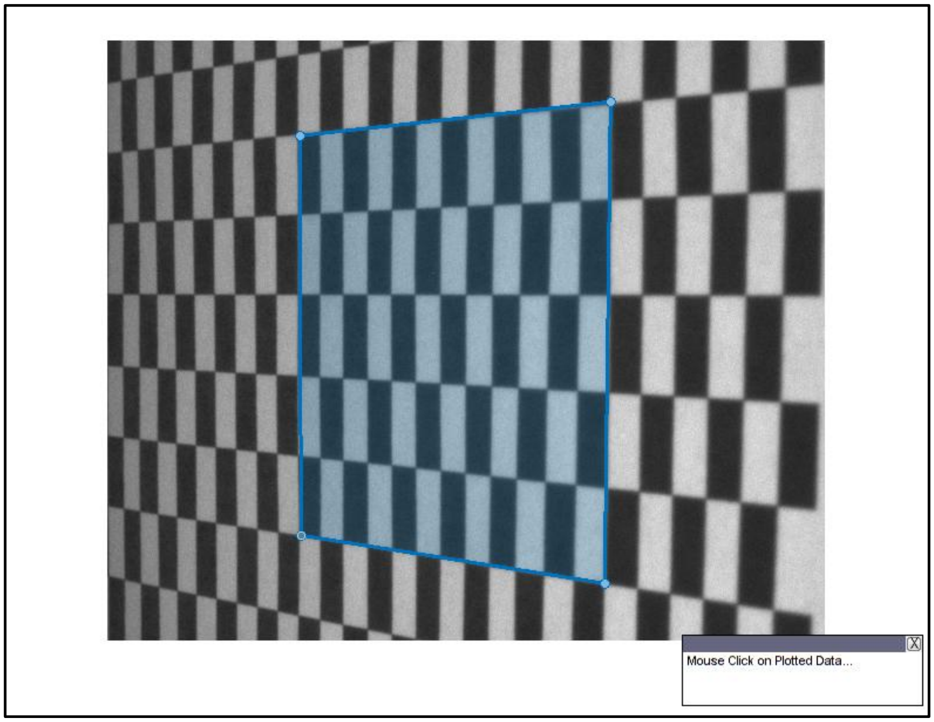

Figure 17 The trapezoid area in the captured image.

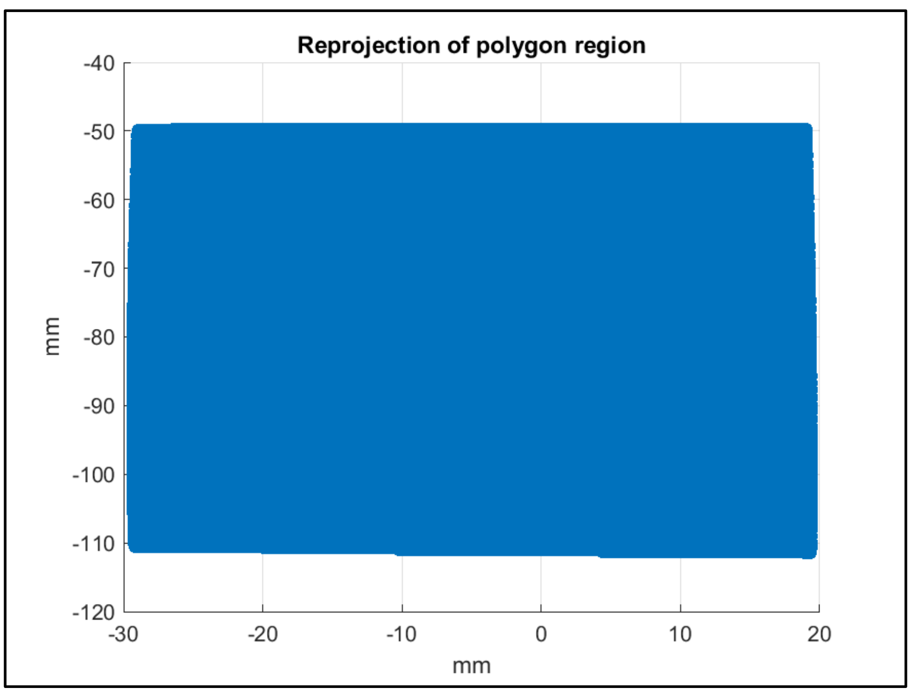

Figure 18 The re-projection of the trapezoid area from the captured image.

3.2 Geometrical Measurements

The experimental setup consists of an LCD monitor with a resolution of 1680×1050pixels and a CMOS camera with a resolution of 1224×1024 pixels. To describe the relative positions of three components (monitor, camera and test object), an imaginary reference plane at z = 0 in the real world coordinates is assumed. The center of test object is nominally the origin point of the reference plane. The camera is placed at a distance of about 300 mm from the testing object with 45 degrees respect to the normal vector of the reference plane. The position of the camera can change is needed to measure the objects with a small radius of curvature or low contrast in captured images. The definition of an LCD monitor's position in our experiment is dependent on the position of the fiducial marker shown in the monitor. The angle between the LCD monitor and the reference plane is 62.5 degrees, and the monitor is placed at a distance of about 150 mm from the test object.

Figure 18 shows the experimental setup of the whole system.

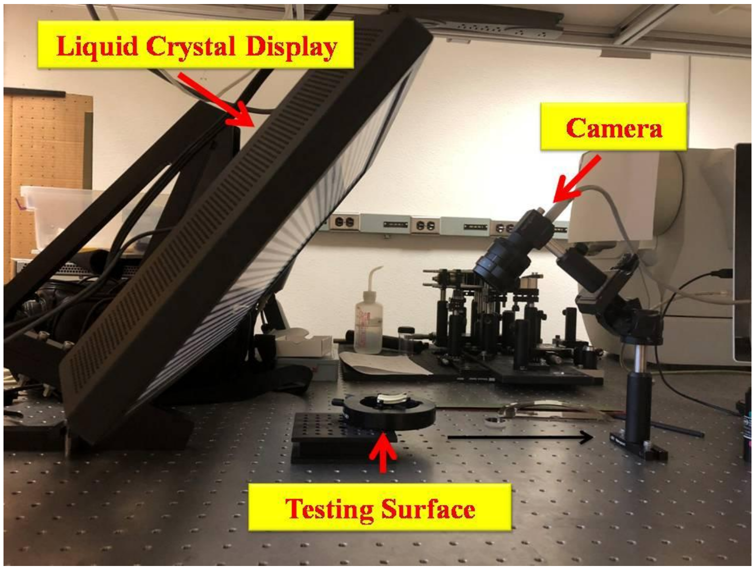

Figure 19 The experimental setup of PMD

3.3 Ray Tracing for Camera and Monitor

The ray tracing for the incident and reflected light vectors are based on the geometrical information of camera and monitor in real-world coordinates. The alignment of the whole system is assumed to be well done, which means both the monitor and camera are not rotated with respect to the y-axis in real-world coordinates. Before calculating the incident and reflected vectors, a fiducial marker is shown on the monitor to be a reference point. Figure 19 shows the cross marker used in the calibration. The fiducial marker shown in monitor, testing objects and camera are the three reference points to determine the relative positions of each other. Figure 20 shows the fiducial marker in the captured image of the flat mirror. Figure 21 shows the fiducial marker in the captured image of the concave mirror in the Polaroid SX-70 Land camera. In this case, the cross marker moves to right direction slightly due to the small radius of curvature in the central region for the test object. Figure 22 shows the fiducial marker in the captured image of the variable anamorphic cylindrical lens.

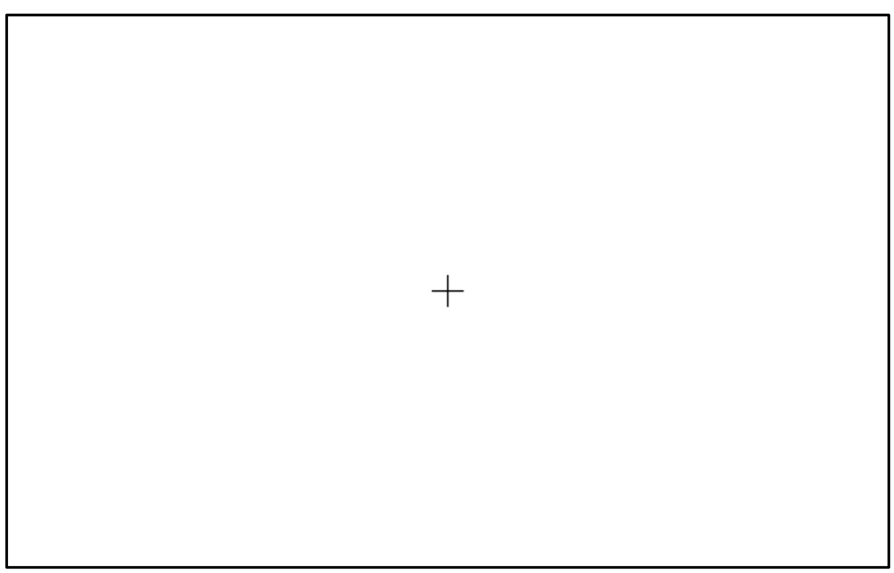

Figure 20 The fiducial marker shown on the monitor to be a calibration flag.

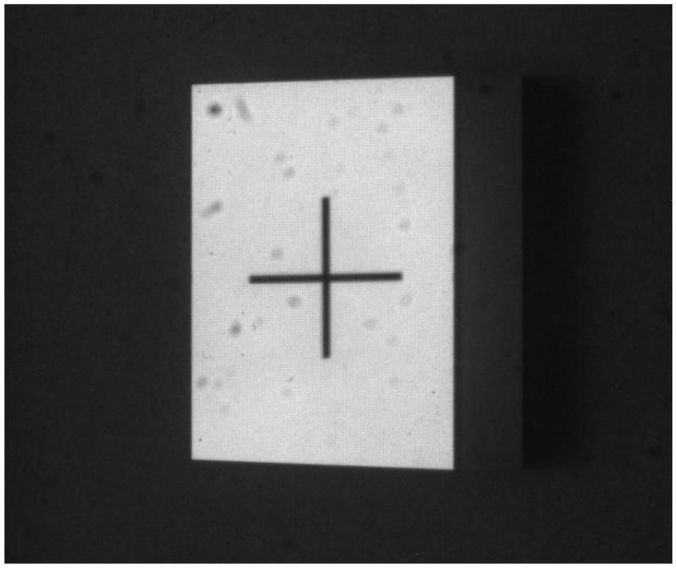

Figure 21 The captured image of the fiducial marker on the flat mirror.

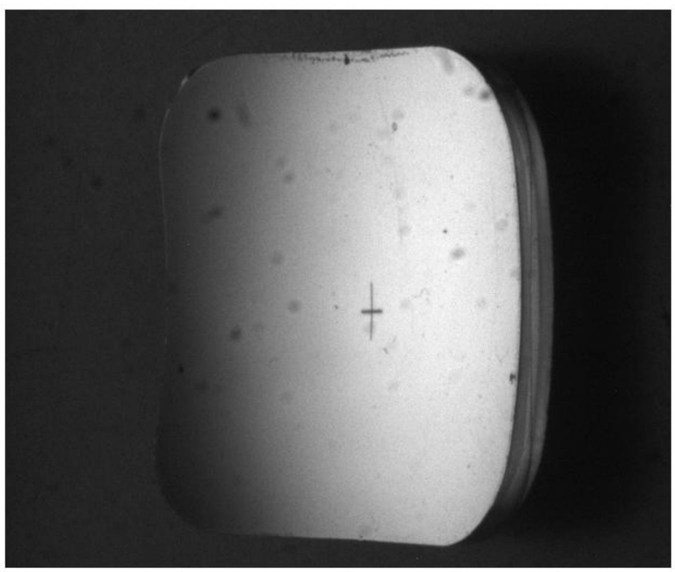

Figure 22 The captured image of the fiducial marker on the concave mirror in the Polaroid SX-70 Land camera.

Note that there are two cross markers shown in the captured image due to the saddle shape of this test object. One of the cross markers is simply chosen as the fiducial marker in the deflectometry process.

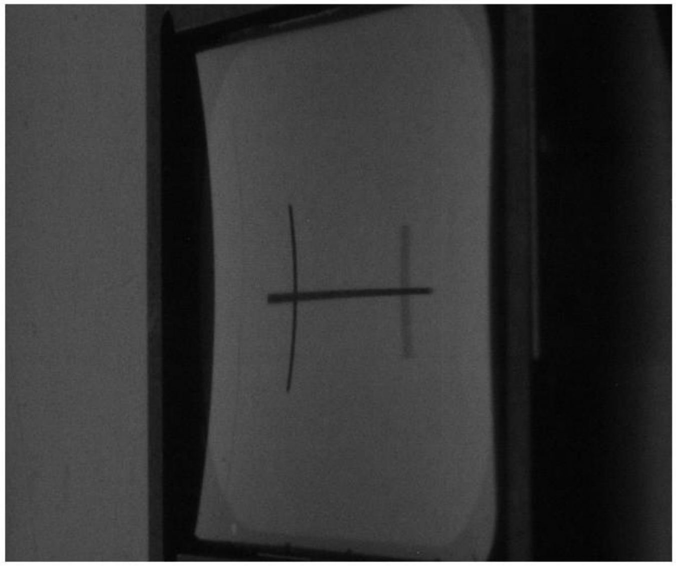

Figure 23 The captured image of the fiducial marker on the variable anamorphic cylindrical lens.

For the incident vectors from testing object to monitor, the real world coordinates for one of the captured image pixels in (x,y,z) is:

$$x = \text{fiducial} + (\text{Phase}(i,j) - \text{Phase}(\text{origin})) * (\text{phase per monitor pixel in the } x - \text{axis})$$

$$y = \text{fiducial} + (\text{Phase}(i,j) - \text{Phase}(\text{origin})) * (\text{phase per monitor pixel in the } y - \text{axis}) * \cos(\text{angle}) * (\text{monitor pixel size})$$

$$z = \text{fiducial} + (\text{Phase}(i,j) - \text{Phase}(\text{origin})) * (\text{phase per monitor pixel in the } y - \text{axis}) * \sin(\text{angle}) * (\text{monitor pixel size})$$

where the parameter "angle" means the angle between monitor and reference plane. The parameter "Phase" means the phase value calculated from the phase unwrapping method for the captured image. To compare the two phase maps between the monitor and captured an image, the sign of phase values have to be modified in the same way. According to the explanation of phase value per pixel in Chapter 2, the quantity can be used in the distance calculation of the corresponding monitor pixel of a captured image pixel. The parameter (i, j) means the pixel number of captured image in x and y-direction, respectively. The coordinate of fiducial for the monitor is (x,y,z) =:

$$x = 0$$

$$y = (\text{distance from the testing object to monitor}) * \sin(\text{incident angle})$$

$$z = (\text{distance from the testing object to monitor}) * \cos(\text{incident angle})$$

Here the fiducial point on the monitor is the known point $r_p = (x_p, y_p, z_p)$ in Chapter 2. For the reflected vectors from testing object to the camera, the real world coordinates for one of the captured image pixels in (x,y,z) is:

$$x = \text{fiducial} + (i - \text{pixelX(origin)}) * (\text{camera pixel size})$$

$$y = \text{fiducial} + (j - \text{pixelY(origin)}) * (\text{camera pixel size}) * \cos(\text{incident angle})$$

$$z = \text{fiducial} + (j - \text{pixelY(origin)}) * (\text{camera pixel size}) * \sin(\text{incident angle})$$

where the "origin" means the center of the image sensor in the camera. The parameter "pixelX" and "pixelY" mean the pixel number of original point in x and y directions. The assumption of a pinhole camera model is used. The property of the pinhole camera model makes the contact points of the reflected vector in image sensor invert and revert from the position in the captured image. The coordinate of fiducial for camera is (x,y,z) =:

$$x = 0$$

$$y = (\text{distance from the testing object to the camera}) * \sin(\text{incident angle})$$

$$z = (\text{distance from the testing object to the camera}) * \cos(\text{incident angle})$$

Once the system geometry has been built up well, calculation of the slopes of testing surface in x and y-direction with ray tracing can begin.

Chapter 4: Experimental Results

4.1 Flat Mirror

A flat mirror is used as the first test object. The mirror is a square-shaped mirror with 1-inch length. Detailed information about the configuration is given below. Figure 23 shows the top view of the experimental setup. The center of the mirror is the origin point, and the x-axis and y-axis have been defined as in the figure. The z-axis is the cross product of x and y-axis. According to Figure 23, the real world coordinates of the CMOS camera is (0, -239mm, 244mm). The angle between the monitor and the reference plane is 62.5 degree. The number of fringes shown on the monitor is 100 in the x-direction, and 60 in the y-direction.

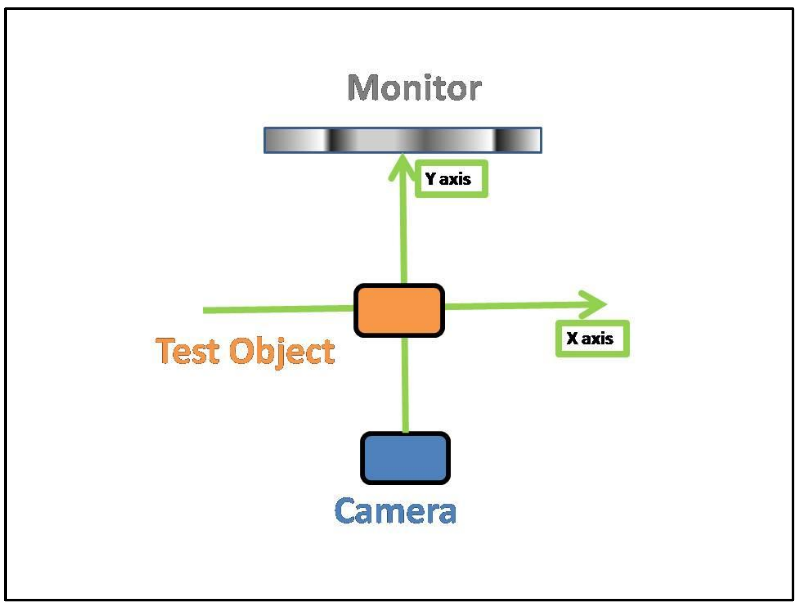

Figure 24 The top view of the experimental setup. The center of the mirror has been taken as the original point, and the x-axis and y-axis have been defined in the figure. The z-axis is the cross product of x and y-axis.

After we defined the real world coordinate, the deflectometry process is initiated. The 8-step phase shifting method is used, and a fourth-order xy polynomial set is used to fit the surface. Below are the phase map in the X and Y direction(Figure 22 and 23), the surface reconstruction via 4-th two-dimensional order polynomial fitting (Figure 24), and the image simulation for fringe patterns in X and Y direction respectively (Figure 25-28).

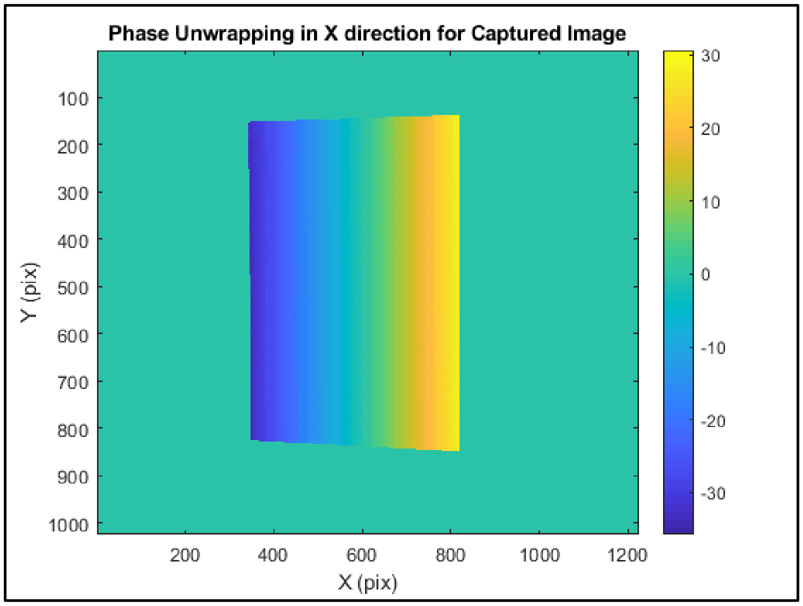

Figure 25 The phase map in x-direction after phase unwrapping for a flat mirror.

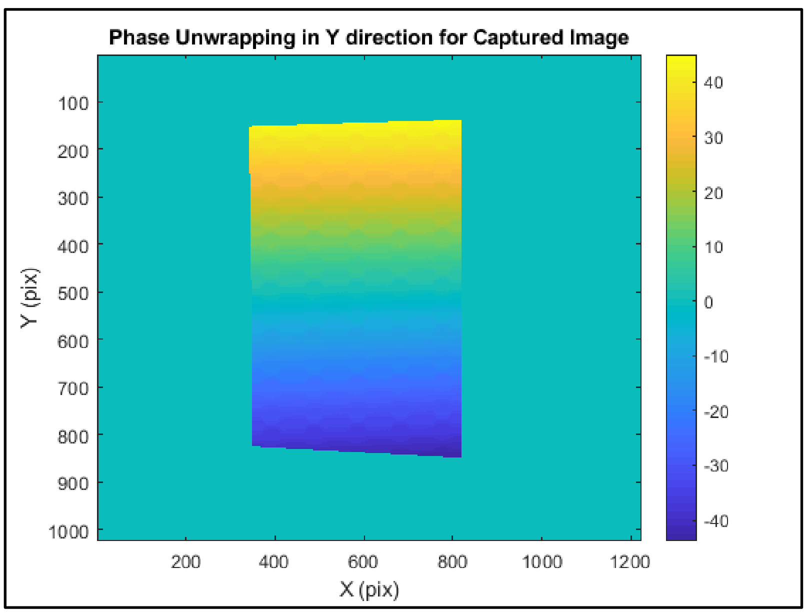

Figure 26 The phase map in y-direction after phase unwrapping for a flat mirror.

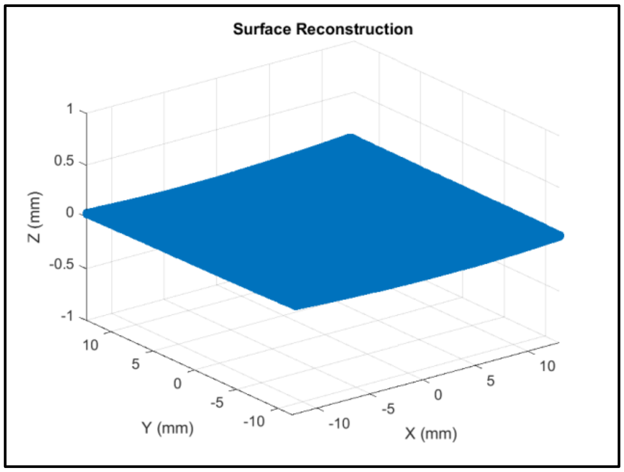

Figure 27 The surface reconstruction for flat mirror via 4-th two-dimensional order polynomial fitting.

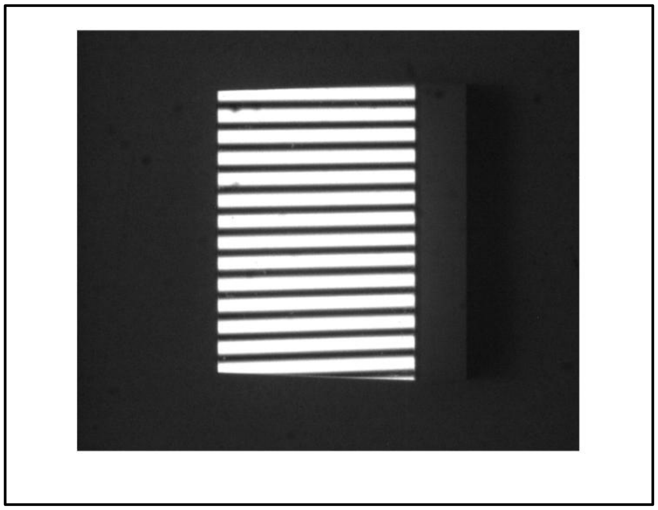

Figure 28 The fringe patterns of the captured image in x-direction on a flat mirror

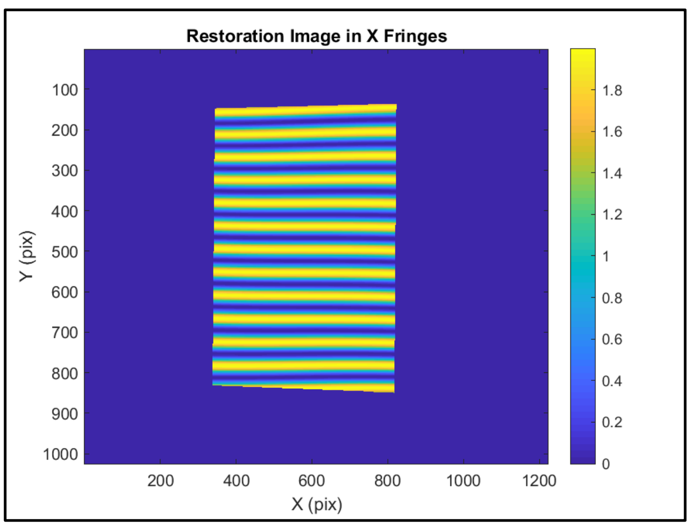

Figure 29 The image simulation of fringe pattern in x-direction on a flat mirror

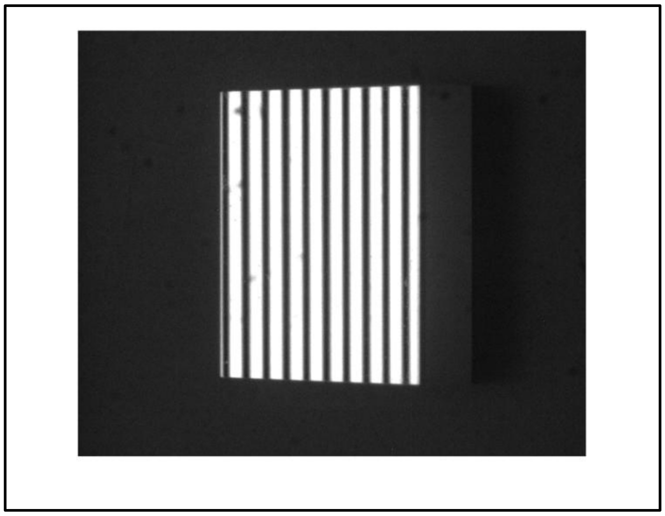

Figure 30 The fringe patterns of the captured image in y-direction on a flat mirror

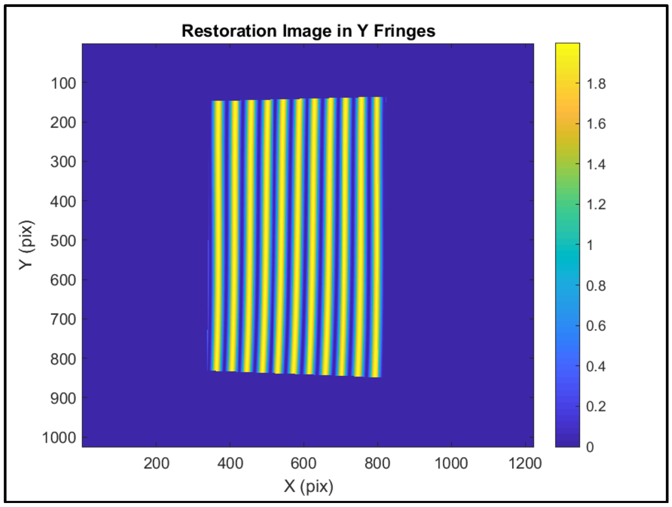

Figure 31The image simulation of fringe pattern in y-direction on a flat mirror

The fitting result with 4th order xy polynomials is following:

$$z = 0.00055 * y - 0.00016 * x + 0.00028 * x^2$$

Here the parameters' values smaller than 0.0001have been excluded due to the limitation of the system sensitivity. The image simulation is based on the height distribution (z values) from the above equation to reproduce the captured image. The simulated captured image in the y-direction is not the same as seen in the real captured image. This issue is discussed in Chapter 5.

4.2 Concave Mirror of the Polaroid SX-70 Land camera

The second test object examined is the concave mirror in the Polaroid SX-70 Land camera (Baker, 1975) (Plummer, 1982). The concave mirror in this design is used to reduce the Petzval field curvature and correct the distortion. It is an aspherical surface. In this case, we used the same configuration as in the flat mirror. The real world coordinates of the CMOS camera are(0, -239mm, 244mm). The angle between the monitor and the reference plane is 62.5 degree. The number of fringes shown on the monitor is 100 in the x-direction, and 60 in the y-direction.

Again, the 8-step phase shifting method is used, as well as a fourth-order xy polynomial fit to the surface. Below the phase map in x and y-direction (Figure 29 and 30), the surface reconstruction via 4-the two-dimensional order polynomial fitting (Figure 31), and the image simulation for fringe patterns in x and y-direction are shown, respectively (Figure 32-35).

The fitting result with 4th order xy polynomials is following:

$$z = -0.0087 * y + 0.0028 * y^2 - 0.00032 * x + 0.00012 * xy + 0.0047 * x^2$$

Since the concave mirror is rotational symmetric, the tip and tilt of the whole system could be observed if the mirror is being rotated. The following result is the 4th order xy polynomials fitting with 90° rotation:

$$z = -0.0064 * y + 0.0030 * y^2 - 0.0035 * x - 0.00015 * xy + 0.0044 * x^2$$

The comparison of two equations shows the tip/tilt from system geometry is small. The x and y terms show the tilt/tilt aberration from system geometry and test surface. Since there is no estimate of uncertainty for fitting method, it is hard to conclude the significance of error from system geometry for fitting results.

Again, the parameters' values smaller than 0.0001 have been excluded due to the limitation of the system sensitivity. After substituting the new z values from fitting, the image simulations are similar to the captured image. Their features are discussed in detail in Chapter 5.

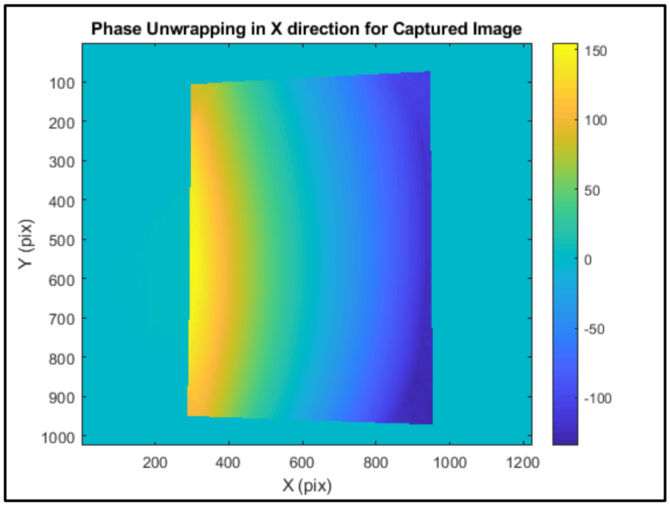

Figure 32The phase map in x-direction after phase unwrapping for a concave mirror.

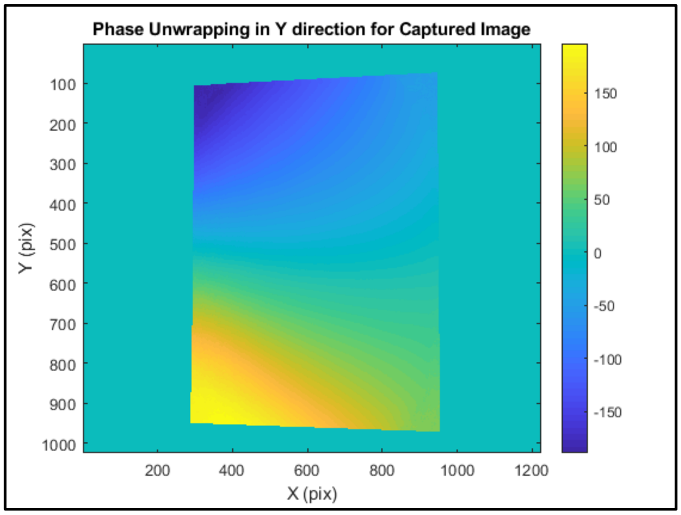

Figure 33The phase map in y-direction after phase unwrapping for the concave mirror.

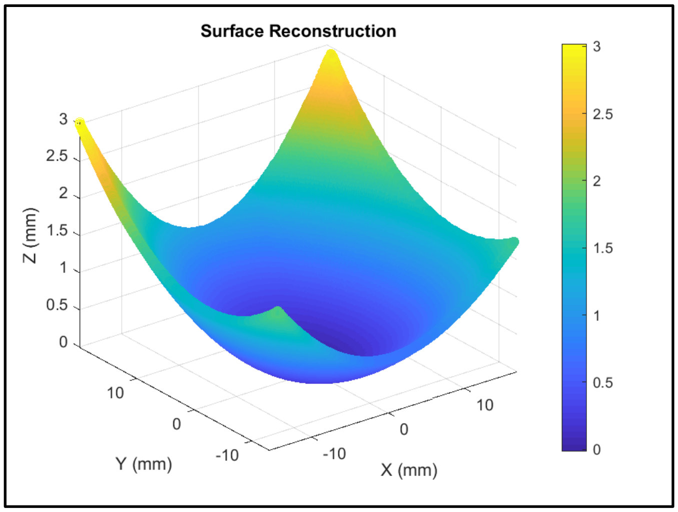

Figure 34The surface reconstruction for concave mirror via 4-th two-dimensional order polynomial fitting.

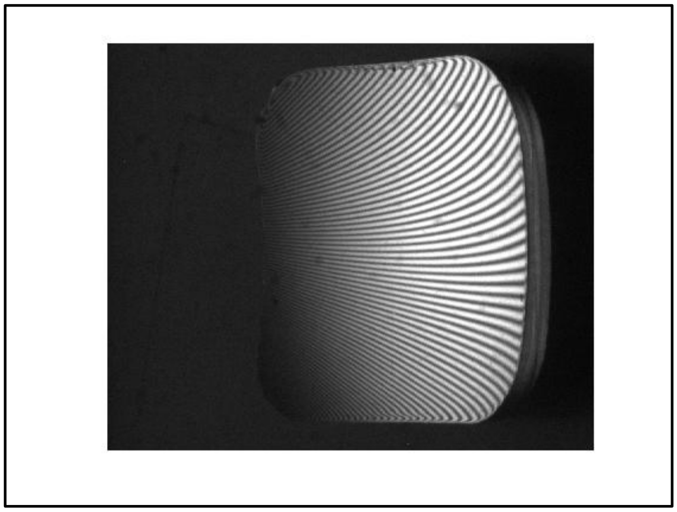

Figure 35The fringe patterns of the captured image in x-direction on the concave mirror

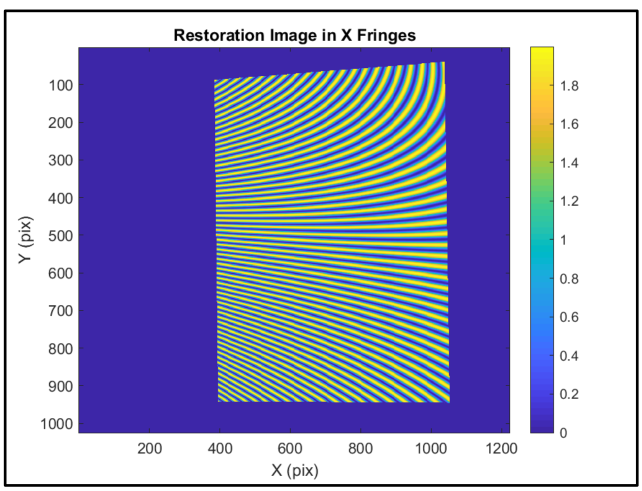

Figure 36The fringe patterns of image simulation in x-direction on the concave mirror

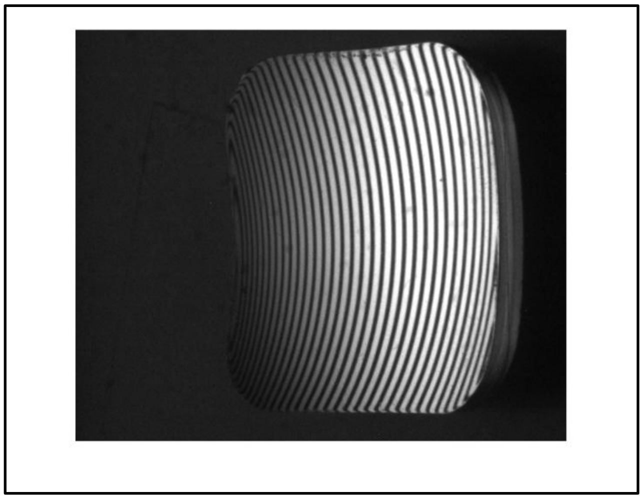

Figure 37 The fringe patterns of the captured image in y-direction on the concave mirror

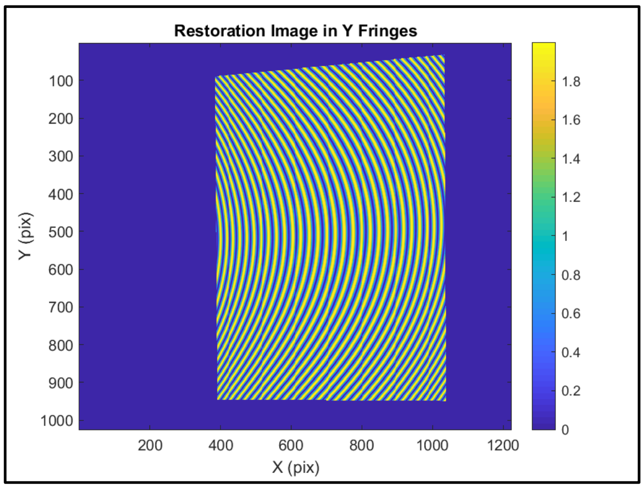

Figure 38 The fringe patterns of image simulation in y-direction on the concave mirror

4.3Variable Anamorphic Cylinder (VAC) lens with variation in 45 degrees

The third test object examined is the variable anamorphic cylinder lens with 45 degrees variation (Humphrey, 1973). An anamorphic lens can generate variable cylindrical lens power, and the variable cylindrical lens rotational alignment could choose the incremental viewpoints through its surface. In this case, a different configuration than in the case of the flat and aspheric mirrors since the low reflection of the lens. The real world coordinates of the CMOS camera are(0, -293mm, 94mm). The angle between the monitor and the reference plane is 62.5 degree. The number of fringes shown on the monitor is 100 in the x-direction, and 100 in the y-direction.

Again, the 8-step phase shifting method is used, along with a fourth-order xy polynomial fit to the surface. The phase map in x and y-direction (Figure 36 and 37), the surface reconstruction via 4-th two-dimensional order polynomial fitting (Figure 38), and the image simulation for fringe patterns in x and y-direction are shown, respectively (Figure 39-42).

Although there are two components of one anamorphic lens set, their shapes are anti-symmetric. Therefore, we only present one of the components results. We could find the feature from image simulation is similar to from the captured image, except for the fringe number. The simulation is discussed in detail in detail at Chapter 5.

The fitting result with 4th order xy polynomials is following:

$$z = -0.00174 * y + 0.0017 * y^2 + 0.0023 * x - 0.00012 * xy + 0.0021 * x^2 + 0.00021 * x^2y$$

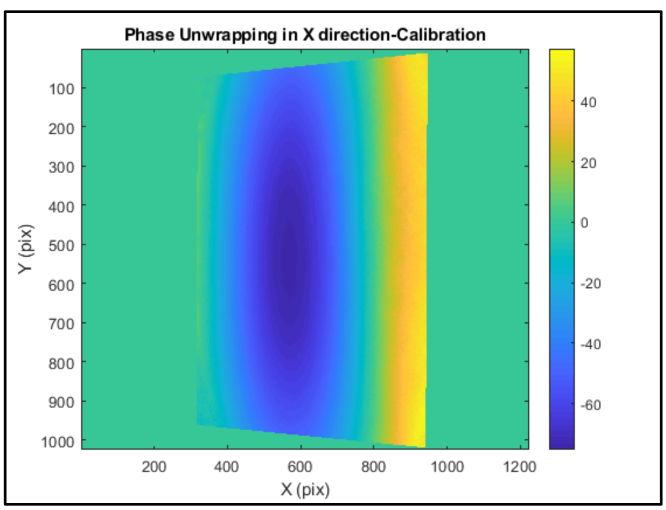

Figure 39The phase map in x-direction after phase unwrapping for VACwith variation in 45 degrees

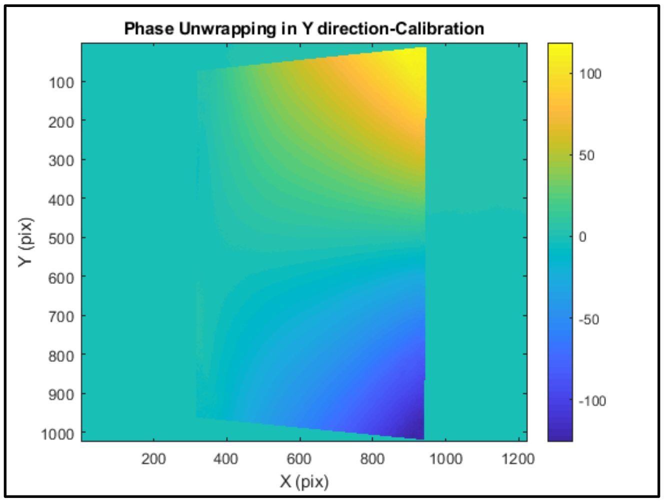

Figure 40The phase map in y-direction after phase unwrapping for VACwith variation in 45 degrees

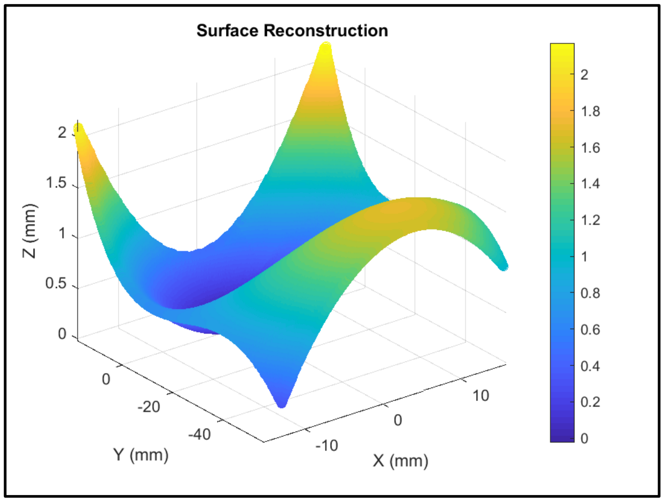

Figure 41The surface reconstruction for VAC with variation in 45 degrees via 4-th two-dimensional order polynomial fitting.

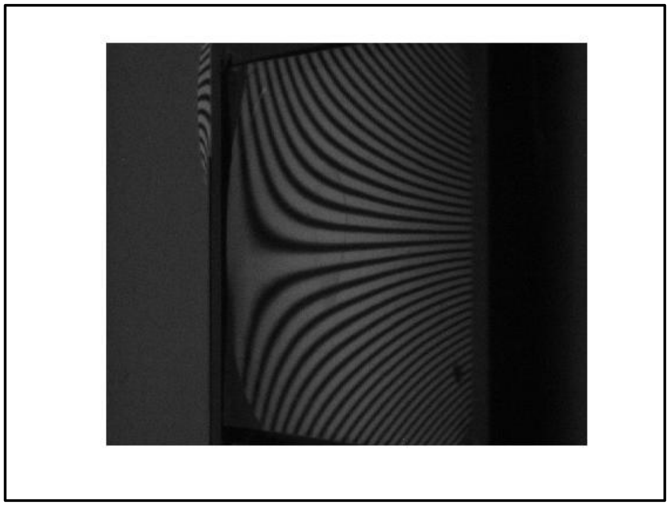

Figure 42 The fringe patterns of the captured image in x-direction on VAC with variation in 45 degrees

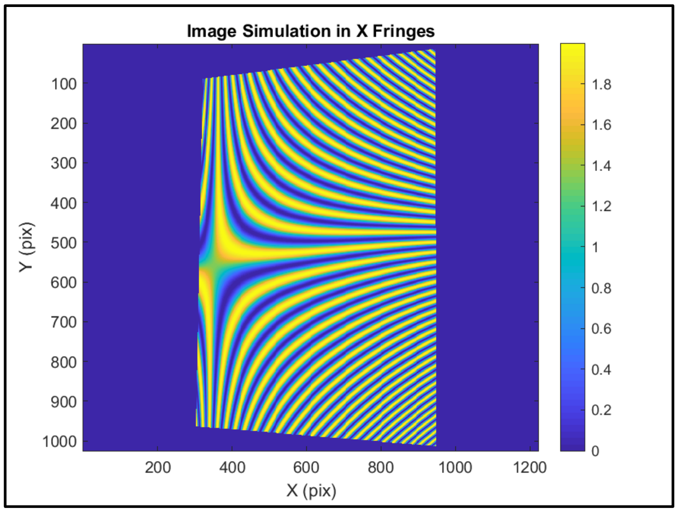

Figure 43 The fringe patterns of image simulation in x-direction on VAC with variation in 45 degrees

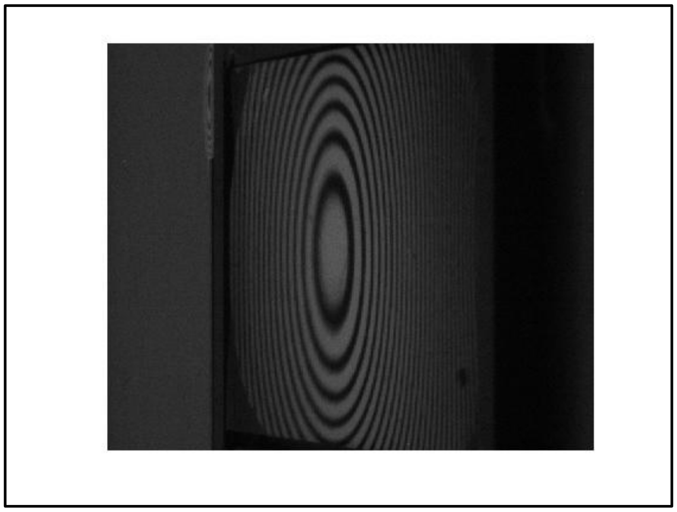

Figure 44 The fringe patterns of the captured image in y-direction on VAC with variation in 45 degrees

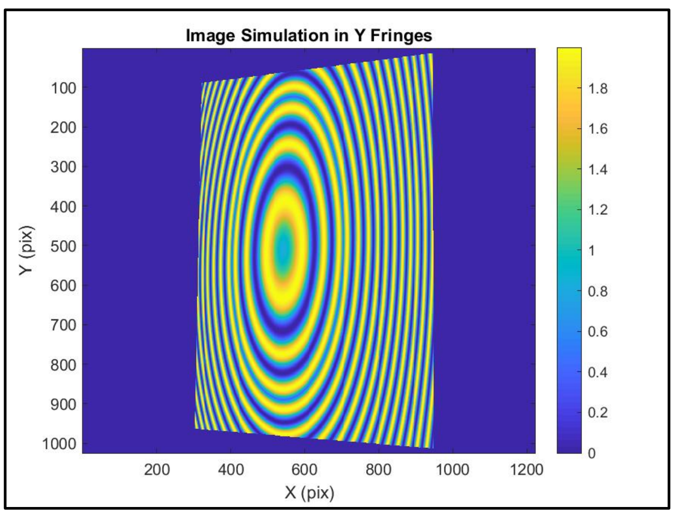

Figure 45 The fringe patterns of image simulation in x-direction on VAC with variation in 45 degrees

4.4Variable Anamorphic Cylinder (VAC) lens with variation in 90 degrees

The last test object examined is the variable anamorphic cylinder lens with 90 degrees variation (Alvarez, 1967) (Alvarez & Humphrey, 1970) (Humphrey, 1973). This lens set is similar to the previous one but with variation in 90 degrees. The real world coordinates of the CMOS camera are(0, -293mm, 94mm). The angle between the monitor and the reference plane is 62.5 degree. The numbers of fringes shown on the monitor are both 100 in the X and Y directions.

The previously described techniques are used to calculate the phase map in x and y-direction (Figure 44 and 45), the surface reconstruction via 4-th two-dimensional order polynomial fitting (Figure 46), and the image simulation for fringe patterns in x and y-direction, respectively (Figure 47-50). Although there are two components of one anamorphic lens set, their shapes are symmetric. Therefore, we only present one of the components results.

The fitting result with 4th order xy polynomials is following:

$$z = -0.0023 * y + 0.0004 * x - 0.00045 * xy - 0.0024 * x^2 + 0.00012 * x^2 y$$

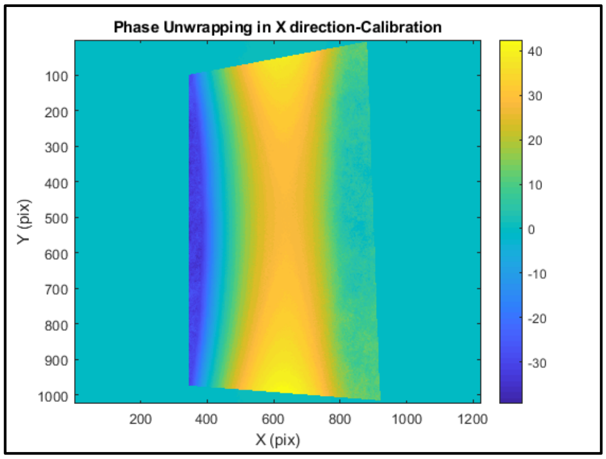

Figure 46 The phase map in x-direction after phase unwrapping for VAC with variation in 90 degrees

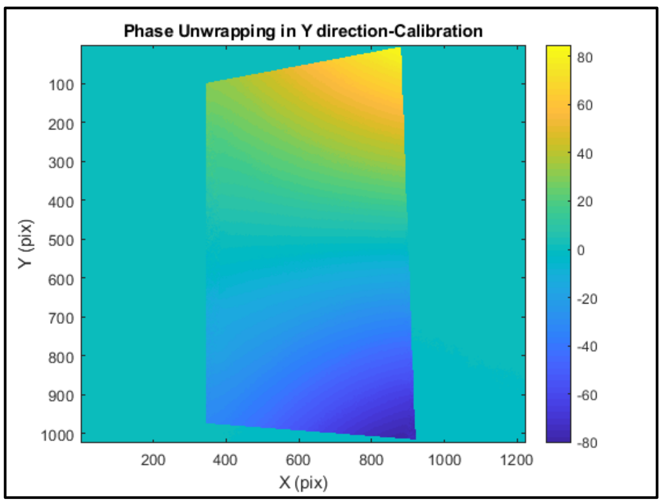

Figure 47 The phase map in y-direction after phase unwrapping for VAC with variation in 90 degrees

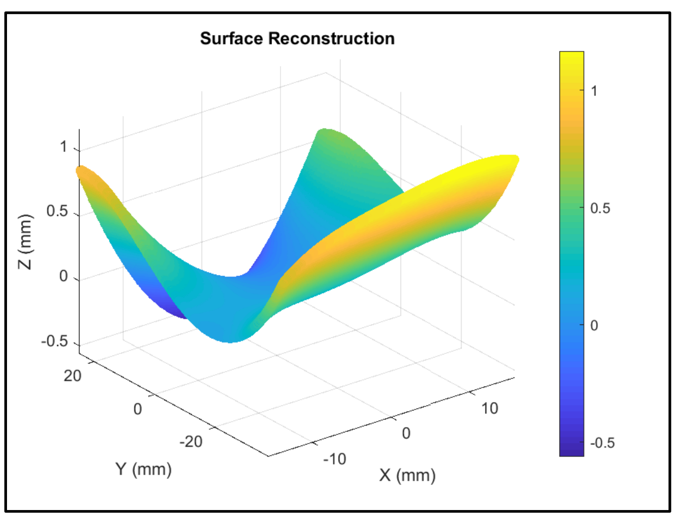

Figure 48The surface reconstruction for VAC with variation in 90 degrees via 4-th two-dimensional order polynomial fitting.

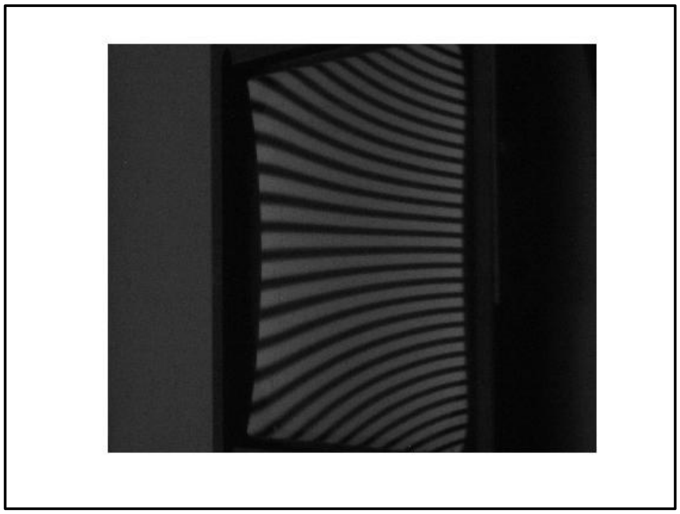

Figure 49 The fringe patterns of the captured image in x-direction on VAC with variation in 90 degrees

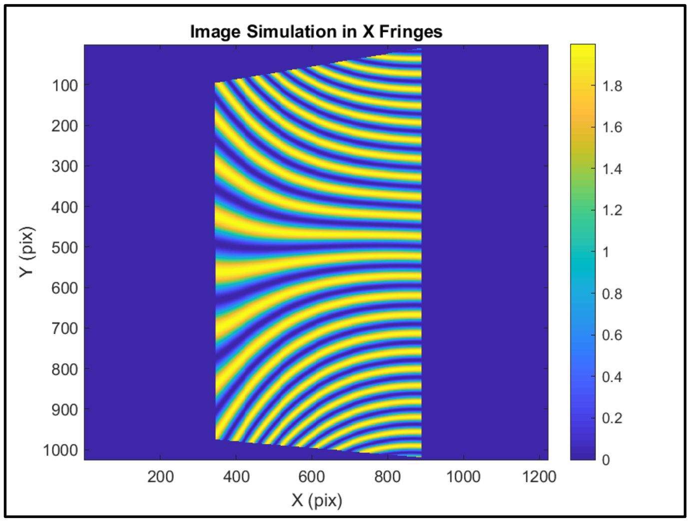

Figure 50 The fringe patterns of image simulation in x-direction on VAC with variation in 90 degrees

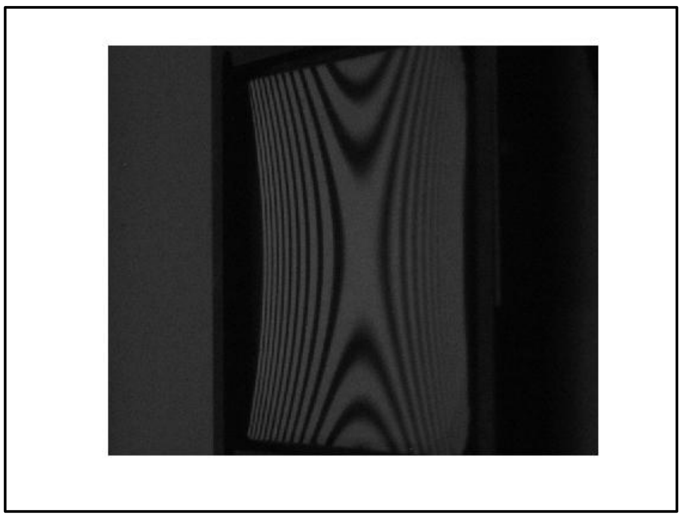

Figure 51The fringe patterns of the captured image in y-direction on VAC with variation in 90 degrees

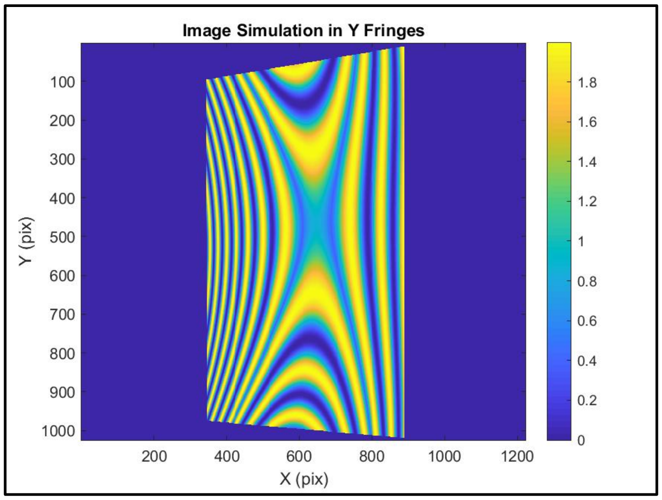

Figure 52 The fringe patterns of image simulation in x-direction on VAC with variation in 90 degrees

Chapter 5: Summary

5.1 Discussion

From the results in Chapter 4, it is observed that the image simulation is quite similar to the originally captured image, but there are still some differences in the features. The most obvious examples are for the flat mirror in the y-direction shown in Figure 27 and 28, and the comparison of fringes number in Figures 39-42. The following are the possible reasons to explain the phenomenon.

(1) Limitation of the depth of focus

The depth of focus in deflectometry plays the most significant role in the ambiguity of the fringe number. When the camera is focused on the testing surface, the measurement gets the best spatial resolution. However, the angular resolution cannot be the best at the same time due to the defocusing of the screen patterns. If the camera is focused on the reflection of the fringe patterns, the measurement can get the best angular resolution, but the spatial resolution becomes worse. In practice, the camera is usually focused on the testing object, and it will cause the ambiguity of fringe number due to the defocus of the camera. The ambiguity of the fringe number would cause the wrong phase calculation after phase unwrapping, and induce the wrong slope calculation. The image simulation would be affected by the ambiguity as processing the inverse ray tracing from screen fringes.

(2) Insufficient fringes density in phase shifting

The sampling of fringe patterns is another reason for ambiguity of fringe number. The insufficient sampling of the fringe patterns would cause the non-smooth phase map after phase unwrapping, and the wrong phase map causes the slope calculation error. Figure 51 is the interception of the 2D phase map with insufficient fringe sampling phase shifting. Figure 52 is the interception of the 2D phase map with sufficient fringe sampling phase shifting.

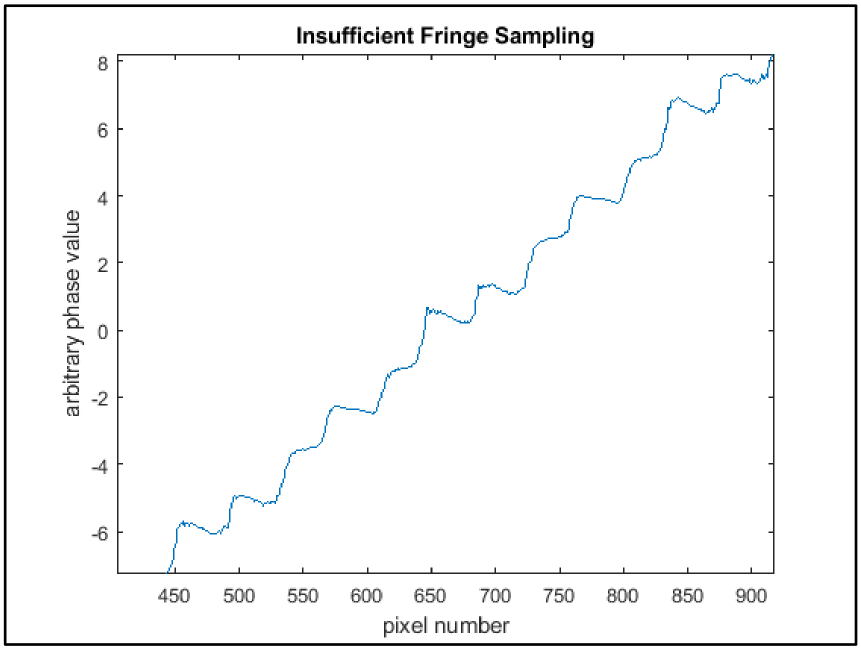

Figure 53 The interception of 2D phase map with insufficient fringe sampling phase shifting

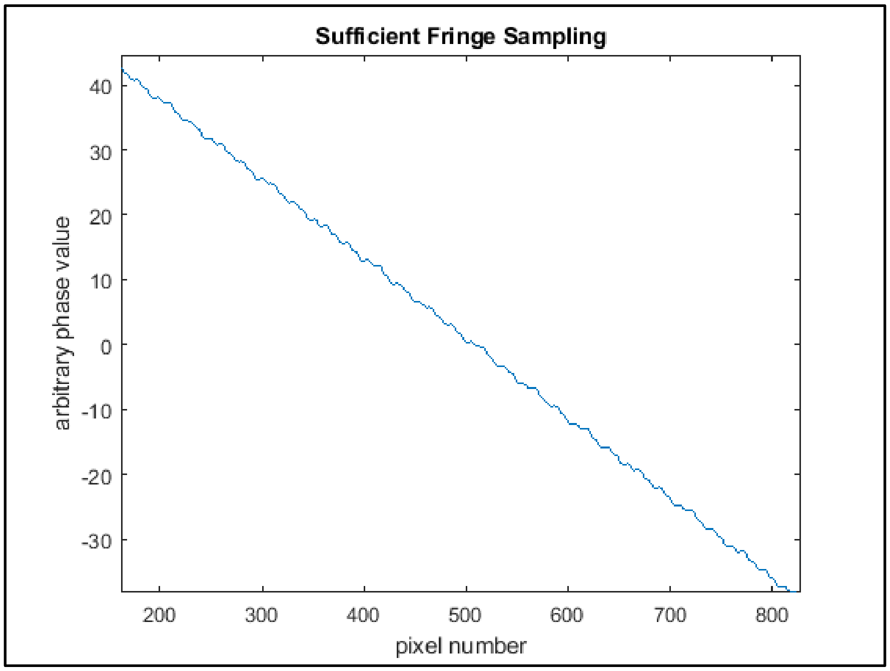

Figure 54 The interception of 2D phase map with sufficient fringe sampling phase shifting.

The difference between insufficient and sufficient fringe sampling causes the wrong phase calculation due to the uncertainty of inverse ray-tracing. This effect can be avoided by adapting to more fringes to cover the testing object as phase shifting processing.

5.2 Future Work

System geometry calibration is a critical procedure for PMD. There are two parts of the calibration process, geometry calibration, and instrumental calibration. A flat mirror with markers can usually be a reference and complete the geometric calibration. However, serious distortion of the camera lens may cause a problem in image re-projection. A linear fitting to correct the keystone effect was used, but this technique still needs to be improved in accuracy. The self-calibration procedure for arbitrary specular surfaces will be our next step (Olesch, Faber, & Hausler, 2011).

The other improvement will be for instruments. The distance laser tracker will improve the accuracy of the system geometry, especially in the reference plane. By using the laser tracker, we can define more than one reference planes to reduce the height-slope ambiguity (Knauer, Kaminski, & Hausler, 2004). Additional cameras and monitors can also help to reduce the height-slope ambiguity

Screen imperfection also needs to be considered (Petz & Tutsch, 2005). However, this has not yet been well addressed and carefully calibrated. In our setup, the resolution of the screen is good enough to show the fringe patterns resolvable, but with the further development in system geometry, the screen imperfection will need to be carefully considered.

5.3 Conclusion

PMD is a powerful tool used for measure the arbitrary specular surface. The high dynamic range and low-cost measurement make this technique to be applied in several scientific applications for deformation, curvature, and shape measurement. This paper demonstrated the preliminary results ofPMDfor the flat mirror, concave mirror, and variable anamorphic cylinder lens. The image simulation shows the inverse ray-tracing can deduce the captured image correctly. Accurate calibration for system geometry and the limitation of system resolution still needs to be improved of the PMD technique. The experimental result shows the validity of this method to sense the high dynamic range 3D shape of the arbitrary specular surface.

Appendix A: Source Code:

```
%Note: x,y in array is [x,y]
%           in matrix is A(y,x)
%Unit: mm

close all
clear
clc

global O2CZ;                                                        %
object to camera Z axis
global O2CY;                                                        %
object to camera Y axis
global mm2pixel_camera;                                             %
camera pixel size; unit:mm
global displayX;                                                    % mm;
(inch/mm)*(inch)
global displayY;                                                    % mm;
(inch/mm)*(inch)
global Xfringes;                                                    %
fringes number in X
global Yfringes;                                                    %
fringes number in Y
global display_x_pixel;                                             % pixel
number of display in X
global display_y_pixel;                                             % pixel
number of display in Y
global MarkerX;                                                     %
Camera Y axis
global MarkerY;                                                     %
Camera X axis
global Xpix;                                                        %
Camera Y axis
global Ypix;                                                        %
Camera X axis
global Load_Cali_Horizon_file;
global Load_Cali_Vertical_file;
global Load_Cali_Marker_file;
global Load_Cali_Stripe_file;
global Load_Keystone_file;

global ang_normal2camera;
global ang_display2plane;

ang_display2plane = deg2rad(ang_display2plane);

marker = [MarkerX MarkerY];

pitchX = displayX/Xfringes;
pitchY = displayY/Yfringes;
```

```
py=1:display_x_pixel;
px=1:display_y_pixel;
[xx,yy] = meshgrid(px,py);

PhiX_D =phase_unwrap(-2*pi*(xx*Xfringes/display_x_pixel));
PhiY_D =phase_unwrap(-2*pi*(yy*Yfringes/display_y_pixel));


mm2pixel_display_x = displayX/display_x_pixel;
mm2pixel_display_y = displayY/display_y_pixel;
%disp(mm2pixel_display_x);

number = 8;

imOrig = imread(Load_Keystone_file);
[imagePoints, boardSize] = detectCheckerboardPoints(imOrig);
for i = 1:length(imagePoints)
for j = 1:length(imagePoints)
if (mod(i,number)==6)
            line1(floor(i./number)+1,1) = imagePoints(i,1);
            line1(floor(i./number)+1,2) = imagePoints(i,2);
end

if (mod(i,number)==5)
            line2(floor(i./number)+1,1) = imagePoints(i,1);
            line2(floor(i./number)+1,2) = imagePoints(i,2);
end
end
end

squareSize = 4.75; % in millimeters

for i=1:length(line1)
    xscale(i) = squareSize/sqrt((line1(i,1)-line2(i,1))^2+(line1(i,2)-
line2(i,2))^2);
if i < length(line1)
    yscale(i) = squareSize/sqrt((line1(i+1,1)-
line1(i,1))^2+(line1(i+1,2)-line1(i,2))^2);
end
end

xls = linspace(line1(1,1),line1(length(line1),1),length(xscale));
yls = linspace((line1(1,1)+line1(2,1))/2,(line1(length(line1)-
1,1)+line1(length(line1),1))/2,length(yscale));
fx=fit(xls',xscale','poly1');
fy=fit(yls',yscale','poly1');
cx1 = fx.p1;
cx2 = fx.p2;
cy1 = fy.p1;
cy2 = fy.p2;
i1o = line1(length(line1),2);
j1o = line1(length(line1),1);

WorldX = zeros(Xpix,Ypix);
WorldY = zeros(Xpix,Ypix);
```

```
%WorldZ = zeros(Ypix,Xpix);

for i = 1:Xpix                  % x axis in my world
for j = 1:Ypix              % y axis in my world
        i1 = i-i1o;
        j1 = j-j1o;
        WorldX(i,j) = (i1*(cx1*j1+cx2))*(-1);
if (j > 1)
        WorldY(i,j) = (cy1*j1 + cy2)*(-1) + WorldY(i,j-1);
else
        WorldY(i,j) = (cy1*j1 + cy2)*(-1);
end
end
end

load(Load_Cali_Marker_file);
figure;
imshow(cmd);
for i=1:5
            shg
            dcm_obj = datacursormode(1);
            set(dcm_obj,'DisplayStyle','window',...
'SnapToDataVertex','off','Enable','on')
            waitforbuttonpress
            P_marker(i)= getCursorInfo(dcm_obj);
end

xOrig = P_marker(1).Position(2);
yOrig = P_marker(1).Position(1);

maskx = [P_marker(2).Position(2) P_marker(3).Position(2)
P_marker(4).Position(2) P_marker(5).Position(2)];
masky = [P_marker(2).Position(1) P_marker(3).Position(1)
P_marker(4).Position(1) P_marker(5).Position(1)];

mask = poly2mask(masky,maskx,Xpix,Ypix);
mask = double(mask);
for i = 1:Xpix % Y axis
for j = 1:Ypix % X axis
if (mask(i,j) == 0)
           mask(i,j) = 0.01;
end
end
end

cxOrig = Xpix/2;
cyOrig = Ypix/2;

ang_normal2camera = atan(O2CY/O2CZ); %normal vector on mirror and
camera; unit:radian

load(Load_Cali_Stripe_file);
plot(v1(:,512));

for i = 1:4
```

```
            shg
            dcm_obj = datacursormode(1);
            set(dcm_obj,'DisplayStyle','window',...
'SnapToDataVertex','off','Enable','on')
            waitforbuttonpress
            P_h(i)= getCursorInfo(dcm_obj);
end

patternXlength =
WorldX(round((P_h(2).Position(1)+P_h(1).Position(1))/2),512) -
WorldX(round((P_h(4).Position(1)+P_h(3).Position(1))/2),512);
ratioX = pitchX/patternXlength-1;  % angle of display and flat plane;
unit:radian
dist_mirror_display = sqrt(O2CY^2 + O2CZ^2)*ratioX;


% Use a cursor to select four points that will crop out a portion of
the
% data and analyze the cropped portion of the data
load(string(Load_Cali_Horizon_file(1)));
load(string(Load_Cali_Horizon_file(2)));
load(string(Load_Cali_Horizon_file(3)));
load(string(Load_Cali_Horizon_file(4)));
load(string(Load_Cali_Horizon_file(5)));
load(string(Load_Cali_Horizon_file(6)));
load(string(Load_Cali_Horizon_file(7)));
load(string(Load_Cali_Horizon_file(8)));

load(string(Load_Cali_Vertical_file(1)));
load(string(Load_Cali_Vertical_file(2)));
load(string(Load_Cali_Vertical_file(3)));
load(string(Load_Cali_Vertical_file(4)));
load(string(Load_Cali_Vertical_file(5)));
load(string(Load_Cali_Vertical_file(6)));
load(string(Load_Cali_Vertical_file(7)));
load(string(Load_Cali_Vertical_file(8)));

h1=double(h1); %Horizontal Fringe pi/4
h2=double(h2); %Horizontal Fringe pi/2
h3=double(h3); %Horizontal Fringe 3pi/4
h4=double(h4); %Horizontal Fringe pi
h5=double(h5); %Horizontal Fringe 5pi/4
h6=double(h6); %Horizontal Fringe 3pi/2
h7=double(h7); %Horizontal Fringe 7pi/4
h8=double(h8); %Horizontal Fringe 2pi

v1=double(v1); %Vertical Fringe pi/4
v2=double(v2); %Vertical Fringe pi/2
v3=double(v3); %Vertical Fringe 3pi/4
v4=double(v4); %Vertical Fringe pi
v5=double(v5); %Vertical Fringe 5pi/4
v6=double(v6); %Vertical Fringe 3pi/2
v7=double(v7); %Vertical Fringe 7pi/4
v8=double(v8); %Vertical Fringe 2pi
```

```
%x=1:Xpix;
%y=1:Ypix;
%[X,Y]=meshgrid(y,x);

h_sin =
h1*sin(2*pi*1/8)+h2*sin(2*pi*2/8)+h3*sin(2*pi*3/8)+h4*sin(2*pi*4/8)+h5*
sin(2*pi*5/8)+h6*sin(2*pi*6/8)+h7*sin(2*pi*7/8)+h8*sin(2*pi*8/8);
h_cos =
h1*cos(2*pi*1/8)+h2*cos(2*pi*2/8)+h3*cos(2*pi*3/8)+h4*cos(2*pi*4/8)+h5*
cos(2*pi*5/8)+h6*cos(2*pi*6/8)+h7*cos(2*pi*7/8)+h8*cos(2*pi*8/8);
v_sin =
v1*sin(2*pi*1/8)+v2*sin(2*pi*2/8)+v3*sin(2*pi*3/8)+v4*sin(2*pi*4/8)+v5*
sin(2*pi*5/8)+v6*sin(2*pi*6/8)+v7*sin(2*pi*7/8)+v8*sin(2*pi*8/8);
v_cos =
v1*cos(2*pi*1/8)+v2*cos(2*pi*2/8)+v3*cos(2*pi*3/8)+v4*cos(2*pi*4/8)+v5*
cos(2*pi*5/8)+v6*cos(2*pi*6/8)+v7*cos(2*pi*7/8)+v8*cos(2*pi*8/8);

PhiX = phase_unwrap(-atan2(h_sin,h_cos),mask);
PhiY = phase_unwrap(-atan2(v_sin,v_cos),mask);

figure,imagesc(PhiX),title('Phase Unwrapping in X direction for
Captured Image'),xlabel('X (pix)'),ylabel('Y
(pix)'),colorbar,saveas(gcf,'PhiX.png');
figure,imagesc(PhiY),title('Phase Unwrapping in Y direction for
Captured Image'),xlabel('X (pix)'),ylabel('Y
(pix)'),colorbar,saveas(gcf,'PhiY.png');

% Construct Normal vector for calibration

ratio_phase2pixelX = display_x_pixel/(max(PhiY_D(:,1))-
min(PhiY_D(:,1)));
ratio_phase2pixelY = display_y_pixel/(max(PhiX_D(1,:))-
min(PhiX_D(1,:)));

% original point in the cross mark
y_d = dist_mirror_display*sin(ang_normal2camera);
x_d = 0;
z_d = dist_mirror_display*cos(ang_normal2camera);
y_c = (-1)*O2CY;
x_c = 0;
z_c = O2CZ;

Sx = zeros(Xpix,Ypix);
Sy = zeros(Xpix,Ypix);
M_X = zeros(Xpix,Ypix);
M_Y = zeros(Xpix,Ypix);
M_Z = zeros(Xpix,Ypix);
C_X = zeros(Xpix,Ypix);
C_Y = zeros(Xpix,Ypix);
C_Z = zeros(Xpix,Ypix);
D_X = zeros(Xpix,Ypix);
D_Y = zeros(Xpix,Ypix);
D_Z = zeros(Xpix,Ypix);
modnumber = 30;
```

```
%coefficient of display plane equation
RA = ang_display2plane - ang_normal2camera; % Rotation angle to the
correct normal vector of the plane
B = -y_d*cos(RA)+z_d*sin(RA);
C = -y_d*sin(RA)-z_d*cos(RA);
D = -(B*y_d + C*z_d);


for i = 1:Xpix % Y axis
for j = 1:Ypix % X axis
if (mask(i,j) == 0.01)
            Sx(i,j) = 0;
            Sy(i,j) = 0;
else

     M_Y(i,j)= WorldY(i,j) - WorldY(xOrig,yOrig);
     M_X(i,j)= WorldX(i,j) - WorldX(xOrig,yOrig);
if(M_Y(i,j) == 0) M_Y(i,j)=0.000001; end
if(M_X(i,j) == 0) M_X(i,j)=0.000001; end


     D_Y(i,j)= y_d - (PhiX(i,j)-
PhiX(xOrig,yOrig))*ratio_phase2pixelY*cos(ang_display2plane)*mm2pixel_d
isplay_y;
     D_X(i,j)= x_d + (PhiY(i,j)-
PhiY(xOrig,yOrig))*ratio_phase2pixelX*mm2pixel_display_x;
     D_Z(i,j)= z_d + (PhiX(i,j)-
PhiX(xOrig,yOrig))*ratio_phase2pixelY*sin(ang_display2plane)*mm2pixel_d
isplay_y;
     C_Y(i,j)= y_c + (j-yOrig)*mm2pixel_camera*cos(ang_normal2camera);
     C_X(i,j)= x_c + (i-xOrig)*mm2pixel_camera;
     C_Z(i,j)= z_c + (j-yOrig)*mm2pixel_camera*sin(ang_normal2camera);

     Dx(i,j) = (D_X(i,j)-M_X(i,j))/norm([D_X(i,j)-M_X(i,j) D_Y(i,j)-
M_Y(i,j) D_Z(i,j)-M_Z(i,j)]);
     Dy(i,j) = (D_Y(i,j)-M_Y(i,j))/norm([D_X(i,j)-M_X(i,j) D_Y(i,j)-
M_Y(i,j) D_Z(i,j)-M_Z(i,j)]);
     Dz(i,j) = (D_Z(i,j)-M_Z(i,j))/norm([D_X(i,j)-M_X(i,j) D_Y(i,j)-
M_Y(i,j) D_Z(i,j)-M_Z(i,j)]);
     Cx(i,j) = (C_X(i,j)-M_X(i,j))/norm([C_X(i,j)-M_X(i,j) C_Y(i,j)-
M_Y(i,j) C_Z(i,j)-M_Z(i,j)]);
     Cy(i,j) = (C_Y(i,j)-M_Y(i,j))/norm([C_X(i,j)-M_X(i,j) C_Y(i,j)-
M_Y(i,j) C_Z(i,j)-M_Z(i,j)]);
     Cz(i,j) = (C_Z(i,j)-M_Z(i,j))/norm([C_X(i,j)-M_X(i,j) C_Y(i,j)-
M_Y(i,j) C_Z(i,j)-M_Z(i,j)]);

     Nx(i,j) = Dx(i,j) + Cx(i,j);
     Ny(i,j) = Dy(i,j) + Cy(i,j);
     Nz(i,j) = Dz(i,j) + Cz(i,j);
     Sx(i,j) = -Nx(i,j)/Nz(i,j);
     Sy(i,j) = -Ny(i,j)/Nz(i,j);
end
end
end
```

```
%%%%%%%%%%%%%%%%%%%%%%% Fitting with Polynomial %%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%% From Jim's book p.173   %%%%%%%%%%%%%%%%%%
Sx = Sx(find(Sx));
Sy = Sy(find(Sy));
b = [Sx; Sy];

order = 4;
xs = M_X(find(M_X));
ys = M_Y(find(M_Y));

[sizexR, sizexC] = size(xs);
[sizeyR, sizeyC] = size(ys);
if (sizexC ~= 1) || (sizeyC ~= 1)
    fprintf( 'Inputs of fit2dPolySVD must be column vectors' );
return;
end
if (sizeyR ~= sizexR)
    fprintf( 'Inputs vectors of fit2dPolySVD must be the same length' );
return;
end
numVals = sizexR;

% number of combinations of coefficients in resulting polynomial
numCoeffs = (order+2)*(order+1)/2;
dx = zeros(numVals, numCoeffs);
dy = zeros(numVals, numCoeffs);
column = 1;
for xpower = 0:order
for ypower = 0:(order-xpower)
if (xpower == 0)
            dx(:,column) = 0;
else
            dx(:,column) = xpower*xs.^(xpower-1) .* ys.^ypower;
end
        column = column + 1;
end
end
column = 1;
for xpower = 0:order
for ypower = 0:(order-xpower)
if (ypower == 0)
            dy(:,column) = 0;
else
            dy(:,column) = ypower*ys.^(ypower-1) .* xs.^xpower;
end
        column = column + 1;
end
end

A = [dx;dy];
A(:,1) = [];
coeffs = inv(A.'*A)*A.'*b;
coeffs1 = [0; coeffs];
zbar1 = eval2dPoly(xs, ys, coeffs1);
v = linspace(1,10,length(zbar1));
```

```
figure,scatter3(xs,ys,zbar1),xlim([min(xs) max(xs)]),ylim([min(ys) max(ys)]),zlim([min(zbar1) max(zbar1)]),xlabel('X (mm)'),ylabel('Y (mm)'),zlabel('Z (mm)'),title('Surface Reconstruction'),saveas(gcf,'FR1.png');

RIX = zeros(Xpix,Ypix);
RIY = zeros(Xpix,Ypix);
dxx = zeros(Xpix,Ypix);
dyy = zeros(Xpix,Ypix);
m_i_x = zeros(Xpix,Ypix);
m_i_y = zeros(Xpix,Ypix);
m_i_z = zeros(Xpix,Ypix);
d_i_x = zeros(Xpix,Ypix);
d_i_y = zeros(Xpix,Ypix);
d_i_z = zeros(Xpix,Ypix);


for i = 1:Xpix % Y axis
for j = 1:Ypix % X axis
if (mask(i,j) ~= 0.01)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Simulation from Patent information %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% concave mirror
     tmp_y = M_Y(i,j);
     tmp_x = M_X(i,j);
     tmp_z = eval2dPoly(tmp_x, tmp_y, coeffs1);

%New unit vector from concave mirror to camera
     InCx_cm(i,j) = (C_X(i,j)-tmp_x)/norm([C_X(i,j)-tmp_x C_Y(i,j)-tmp_y C_Z(i,j)-tmp_z]);
     InCy_cm(i,j) = (C_Y(i,j)-tmp_y)/norm([C_X(i,j)-tmp_x C_Y(i,j)-tmp_y C_Z(i,j)-tmp_z]);
     InCz_cm(i,j) = (C_Z(i,j)-tmp_z)/norm([C_X(i,j)-tmp_x C_Y(i,j)-tmp_y C_Z(i,j)-tmp_z]);

     nx(i,j) = -derivativeX(tmp_x, tmp_y, coeffs1);
     ny(i,j) = -derivativeY(tmp_x, tmp_y, coeffs1);
     nz(i,j) = 1;

%unit vector for normal vector of concave mirror
     Nz_cm(i,j) = nz(i,j)/norm([nx(i,j) ny(i,j) nz(i,j)]);
     Nx_cm(i,j) = nx(i,j)/norm([nx(i,j) ny(i,j) nz(i,j)]);
     Ny_cm(i,j) = ny(i,j)/norm([nx(i,j) ny(i,j) nz(i,j)]);
%unit vector for reflected light from concave mirror
     unit_scale = 2*(Nx_cm(i,j)*InCx_cm(i,j) + Ny_cm(i,j)*InCy_cm(i,j) + Nz_cm(i,j)*InCz_cm(i,j));
     Rx_tmp = unit_scale*Nx_cm(i,j)-InCx_cm(i,j);
     Ry_tmp = unit_scale*Ny_cm(i,j)-InCy_cm(i,j);
     Rz_tmp = unit_scale*Nz_cm(i,j)-InCz_cm(i,j);
     Rx_cm(i,j) = Rx_tmp/norm([Rx_tmp Ry_tmp Rz_tmp]);
     Ry_cm(i,j) = Ry_tmp/norm([Rx_tmp Ry_tmp Rz_tmp]);
     Rz_cm(i,j) = Rz_tmp/norm([Rx_tmp Ry_tmp Rz_tmp]);
%convert intersection point (concave mirror) from concave mirror
```

```
coordinate to marker coordnate
     m_i_x(i,j) = tmp_x;% + (central_index(1) -
P_marker.Position(1))*mm2pixel_mirror;%mirror intersection point x in
P_marker coordinate
     m_i_y(i,j) = tmp_y;% - (central_index(2) -
P_marker.Position(2))*hmax(1)/vmax(1)*mm2pixel_mirror; %mirror
intersection point y in P_marker coordinate
     m_i_z(i,j) = tmp_z;

     t2(i,j) = -
(B*m_i_y(i,j)+C*m_i_z(i,j)+D)/(B*Ry_cm(i,j)+C*Rz_cm(i,j));
%disp(t2(i,j));
     d_i_z(i,j) = m_i_z(i,j) + t2(i,j)*Rz_cm(i,j);
     d_i_x(i,j) = m_i_x(i,j) + t2(i,j)*Rx_cm(i,j);
     d_i_y(i,j) = m_i_y(i,j) + t2(i,j)*Ry_cm(i,j);

%sqrt((d_i_x-x_d)^2 + (d_i_y-y_d)^2 + (d_i_z-z_d)^2);
      dxx(i,j) = (d_i_x(i,j)-x_d)/mm2pixel_display_x + MarkerX;
if (d_i_y(i,j) < y_d)
      dyy(i,j) =  sqrt((d_i_y(i,j)-y_d)^2 + (d_i_z(i,j)-
z_d)^2)/mm2pixel_display_y;
else
      dyy(i,j) = -sqrt((d_i_y(i,j)-y_d)^2 + (d_i_z(i,j)-
z_d)^2)/mm2pixel_display_y;
end
      dyy(i,j) = dyy(i,j) + MarkerY;

      F_X = 1+cos(2*pi*(dxx(i,j)*Xfringes/display_x_pixel)); %
Horizontal Fringes
      F_Y = 1+cos(2*pi*(dyy(i,j)*Yfringes/display_y_pixel)); % Vertical
Fringes

      RIX(i,j) = F_X; %Restoration Image in X Fringes
      RIY(i,j) = F_Y; %Restoration Image in Y Fringes
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  END
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%

end
end
end
figure,imagesc(RIX),title('Restoration Image in X Fringes'),xlabel('X
(pix)'),ylabel('Y (pix)'),colorbar,saveas(gcf,'RIX.png');
figure,imagesc(RIY),title('Restoration Image in Y Fringes'),xlabel('X
(pix)'),ylabel('Y (pix)'),colorbar,saveas(gcf,'RIY.png');
```

References

Alvarez, L. W. (1967). *Patent No. 3305294.* United States.

Alvarez, L. W., & Humphrey, W. E. (1970). *Patent No. 3507565.* United States.

Baker, J. G. (1975). *Patent No. 3860940.* United States.

Cai, W., Zhou, P., Zhao, C., & Burge, J. (2013). Analysis of wavefront errors introduced by encoding computer-generated holograms. *Appl. Opt. Issue 52* , 8324-31.

Dai, G.-m. (1996). Modal wave-front reconstruction with Zernike polynomials and Karhunen-loeve functions. *J Opt Soc Am A, Vol.121* , 8-25.

Ettl, S., Kaminski, J., Knauer, M., & Hausler, G. (2008). Shape reconstruction from gradient data. *Appl Opt, Vol.47* , 2091-7.

Frecher, A. (1976). Computer-generated Holograms for Testing Optical Elements: Error Analysis and Error Compensation. *Opt.Acta Int. J. Opt. Issue 23* , 347-365.

Fried, D. (1977). Least-square fitting a wave-front distortion estimate to an arry of phase-difference measurements. *J Opt Soc Am A, Vol.67* , 370-5.

Ghiglia, D. C., & Romero, L. A. (1994). Robust two-dimensional weighted and unweighted phase unwrapping that uses fast transforms and iterative methods. *JOSA A, Vol.11, Issue.1* , 107.

Graves, L. R., Choi, H., Zhao, W., & Kim, D. (2018). Model-free deflectometry for freeform optics measurement using an iterative reconstruction technique. *Optics Letters, Vol.43* , 2110.

Huang, L., Idir, M., Zuo, C., & Asundi, A. (2018). Review of phase measuring deflectometry. *Optics and Lasers in Engineering, Issue 107* , 247-257.

Humphrey, W. E. (1973). *Patent No. 3751138.* United States.

Knauer, M., Kaminski, J., & Hausler, G. (2004). Phase measuring deflectometry: a new approach to measure specular free form surfaces. *International Society for Optics and Photonics* , 366-376.

Li, W., Sandner, M., Gesierich, A., & Burke, J. (2012). Absolute optical surface measurement with deflectometry. *SPIE Optical Engineering and Applications* , 84940G.

Malacara, D. (2007). Foucault, Wire, and Phase Modulation Tests. *Optical Shop Testing* , 275-313.

Malacara, D. (2007). Hartmann, Hartmann-Shack, and Other Screen Tests. *Optical Shop Testing* , 361-394.

Malacara, D. (2007). Ronchi Test. *Optical Shop Testing* , 317-350.

Mansuripur, M. (1997). The Ronchi Test. *Opt. & Photonics News, Issue 8* .

Neal, D., Copland, J., & Neal, D. (2002). Shack-Hartmann wavefront sensor precision and accuracy. *International Society for Optics and Photonics* , 148-160.

Olesch, E., Faber, C., & Hausler, G. (2011). Deflectometric self-calibration for arbitrary specular surfaces. *Proceedings of the 125th annual meeting of the DGaO A* .

Petz, M., & Tutsch, R. (2003). Measurement of optically effective surfaces by imaging of gratings. . *Proceedings of SPIE* , 288-94.

Petz, M., & Tutsch, R. (2005). Reflection grating photogrammetry: a technique for absolute shape measurement of specular free-form surfaces. *Optics and Photonics, SPIE* , 59691D.

Platt, B., & Shack, R. (2001). History and principles of Shack-Hartmann wavefront sensing. *J. Refract. Surg* , S573-7.

Plummer, W. T. (1982). Unusual optics of the Polaroid SX-70 Land camera. *Applied Optics, Vol.21* , 196.
Pope, D. R. (2000). Progressive Addition Lenses: History, Design, Wearer Satisfaction and Trends. *OSA Technical Digest* (p. 342). Santa Fe: Optical Society of America.
Schmit, J., Creath, K., & Wyant, J. C. (2007). Surface Profilers, Multiple Wavelength, and White Light Interferometery . In D. Malacara, *Optical Shop Testing* (p. 670). Hoboken, New Jersey: John Wiley & Sons.
Schwiegerling, J. (2019). *OPTICAL SPECIFICATIONS, FABRICATION AND TESTING: Lecture Notes.*
Smith, W. J. (2000). *Modern Optical Engineering.* New York: McGraw-Hill.
Song, J., & Vorburger, T. (1991). Stylus profiling at high resolution and low force. *Appl. Opt, Issue 30* , 42-50.
Su, P., Parks, R., Wang, L., Angel, R., & Burge, J. (2010). Software configurable optical test system: a computerized reverse Hartmann test. *Appl. Opt. Issue 49* , 4404-12.
Thompson, K. P., & Rolland, J. P. (2012). Freeform Optical Surfaces: A Revolution in Imaging Optical Design. *Optics and Photonics News Vol.23 Ossue 6* , 30-35.
Werling, S., Mai, M., Heizmann, M., & Beyerer, J. (2009). Inspection of specular and partially specular surfaces. *Metrol Meas Syst, Issue 16* , 415-31.
Williamson, R. (2011). *Field Guide to Optical Fabrication.* Bellingham, Washington: SPIE.
Wyant, J. (2002). White light interferometry . *International Society for Optics and Photonics* , 98-107.
Yoder, Jr., P. R. (2010). Mounting Optical Components. In M. Bass, *Handbook of Optics* (p. 37.3). New York: The McGraw-Hill Companies, Inc.