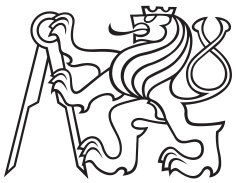
Master Thesis



Czech Technical University in Prague

F3

Faculty of Electrical Engineering
Department of Cybernetics

Edge Detection and 3D Reconstruction Based on the Shape-from-Focus

Jakub Cmíral

Supervisor: Ing. Pavel Krsek, Ph.D.
May 2019

Acknowledgements

I am grateful to my supervisor Dr. Pavel Krsek for guiding me and enabling me to comprehend a little what is research. I appreciate the help of other members of the Robotic Perception Group at CIIRC ČVUT.

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instruction for observing the ethical principles in the preparation of university theses.

Prague, May 23, 2019

Abstract

The work stems from the industrial project which aims to build the highly precise micro components assembly machine. The components are positioned via locating the edges in the image. The overview of the edge detection techniques and the design of the Shape-from-Focus algorithm in the microscopic environment are presented.

The images used were captured with telecentric optics with a shallow Depth-of-Field. The Shape-from-Focus algorithm is developed together with the 3D convolutional mask and approximation of the surface in the textureless areas. The developed 3D convolutional filter is based on the seconds derivative of the image function.

Various edge detection techniques are used in experiments to calibrate the camera and to refocus the optics. The experiments also show the surface reconstruction obtained by the Shape-from-Focus algorithm.

Keywords: 3D reconstruction, edge detecion, Shape-from-Focus, telecentric lens

Supervisor: Ing. Pavel Krsek, Ph.D.
Czech Technical University in Prague,
Czech Institute of Informatics, Robotics and Cybernetics
Jugoslávských partyzánů 1580/3,
Prague 6, 166 36,
Czech Republic

Abstrakt

Má práce vychází z průmyslového projektu, jehož cílem je postavit stroj pro přesnou manipulaci mikrokomponenty. Zmíněné mikrokomponenty jsou sledovány na základě hledání hran v obraze. Má práce popisuje přehled postupů používaných pro detekci hran v obraze a zároveň návrh algoritmu pro rekonstrukci povrchu mikrokomponent pomocí Shape-from-Focus v mikroskopickém prostředí.

Použité obrázky byly pořízeny kamerou s telecentrickým objektivem s malou hloubkou ostrosti. Vyvinul jsem Shape-from-Focus algoritmus, který používá 3D konvoluční masku pro detekci hran a je schopný aproximovat povrchy bez struktury. Vyvinutá 3D konvoluční maska je založena na druhé derivaci obrazové funkce.

V pokusech popisujících kalibraci kamery a pro opětovné zaostření optické soustavy byly použity rozličné metody pro detekci hran v obraze. V pokusech se také prezentují výsledky rekonstrukce povrchu pomocí navrženého Shape-from-Focus algoritmu.

Klíčová slova: 3D rekonstrukce, hledání hran v obraze, Shape-from-Focus, telecentrický objektiv

Překlad názvu: Detekce hran a 3D rekonstrukce na základě rozostření

Contents

Figures

MASTER'S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Cmíral Jakub** Personal ID number: **434665**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Cybernetics**

Study program: **Cybernetics and Robotics**

Branch of study: **Robotics**

II. Master's thesis details

Master's thesis title in English:

Edge Detection and 3D Reconstruction Based on the Shape-from-Focus

Master's thesis title in Czech:

Detekce hran a 3D rekonstrukce na základě rozostření

Guidelines:

1. Study the existing edge detection and 3D reconstruction (Shape-from-Focus) algorithms.
2. Develop a 3D reconstruction algorithm based on the Shape-from-Focus.
3. Prepare methods for processing obtained 3D data.
4. Test developed algorithms on real data.
5. Write proper documentation.

Bibliography / sources:

[1] Milan Sonka, Vaclav Hlavac, and Roger Boyle. Image Processing, Analysis and Machine Vision. Thomson, 3rd edition, ISBN 978-0-495-08252, 2007.
[2] Hartley, Richard and Zisserman, Andrew. Multiple view geometry in computer vision. Cambridge University, 2nd edition, ISBN 0-521-54051-8, 2003.
[3] Said Pertuz, Domenec Puig, and Miguel Angel Garcia. Analysis of focus measure operators for shape-from-focus. Pattern Recognition, 46(5):1415 – 1432, 2013.

Name and workplace of master's thesis supervisor:

Ing. Pavel Krsek, Ph.D., Robotic Perception, CIIRC

Name and workplace of second master's thesis supervisor or consultant:

Date of master's thesis assignment: **09.01.2019** Deadline for master's thesis submission: **24.05.2019**

Assignment valid until: **30.09.2020**

Ing. Pavel Krsek, Ph.D.	doc. Ing. Tomáš Svoboda, Ph.D.	prof. Ing. Pavel Ripka, CSc.
Supervisor's signature	Head of department's signature	Dean's signature

III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

Date of assignment receipt	Student's signature

© ČVUT v Praze, Design: ČVUT v Praze, VIC

Chapter 1

Introduction

My work stems from the industrial project Control Platform for High-Accuracy Microelectronics (CoPA) supported by the Technological Agency of the Czech Republic (TAČR). The project aims to build a precise assembly machine for micro-components. Example of such a component is a laser transmitter for optical communication. The machine must be able to precisely pick the components from the source position, place them, and weld them to the target position. The machine will be applied in production.

Neither the position of components nor the position of welding place are precisely known in advance. Both places are observed by a camera. The component and the welding place are localized in the image. Both the component and the welding place are designed in a way that their approximate positions are obtained via detecting placement markers. The precise position of the component is given by edges in the placement region. The detection requires one pixel precision on the source side and possibly a subpixel precision on the target side. My contribution to the project is an edge detection mechanism for the precise micro-components placement.

The camera on the target side has a shallow Depth-of-Field and the images could be slightly defocused. I was interested in how defocussing the image affects the subpixel edge detection. I conducted experiments on how the defocussing of an edge affects the confidence of its position (Section 5.1.2). I decided to detect the edges in the whole sequence of defocused images. I obtained a sequence of images with detected edges and the value of confidence of the edge position. I discussed the results with my supervisor and we decided to apply the Shape-from-Focus algorithm on the sequence to reconstruct the shape of the micro-components.

My thesis is primarily focused on the Shape-from-Focus algorithm introduced in the following Section 1.1. The other work related to the Shape-from-Focus algorithm is discussed in Chapter 2. The edge detection serves as an introduction to this algorithm. The edge detection in the images is a well-established problem in image processing and it is described in Section 3.1. My Shape-from-Focus algorithm is presented in detail in Section 3.3. Chapter 4 describes the used hardware and software. Chapter 5 shows and discusses the experiments.

1.1 Principle description

A camera with a limited Depth-of-Field (DoF) observes the scene. The camera is moving in such a way that its DoF flows thru the scene from the closest to the furthest point in the scene. The images from the camera are taken in equidistant distances. Such images create an image sequence. The Shape-from-Focus algorithm is a 3D reconstruction technique which seeks for the best-focused texture in depth for each pixel in the sequence. The result of Shape-from-Focus reconstruction is a depth map with the same size as the original image. A single fully in-focus image can be extracted from the sequence based on the depth map. The simple Shape-from-Focus algorithm requires a textured area to work correctly. My images have a lot of textureless areas. On the other hand, they provide a lot of well-defined edges and the surface between edges can be considered flat due to the manufacturing process of the components, i.e. the foundation is a flat circuit board, and the components are shaped as a cuboid or a frustum. An example of the used image is displayed in Figure 1.1.

The Shape-from-Focus algorithm was modified to detect the edges in the sequence, which results in the edge depth map. The prior knowledge about the flatness of the surface between edges was used and the edge elements were connected by a full triangulation. The resulting triangulation provided a sufficient depth estimation in the textureless areas in the case of flat surfaces.

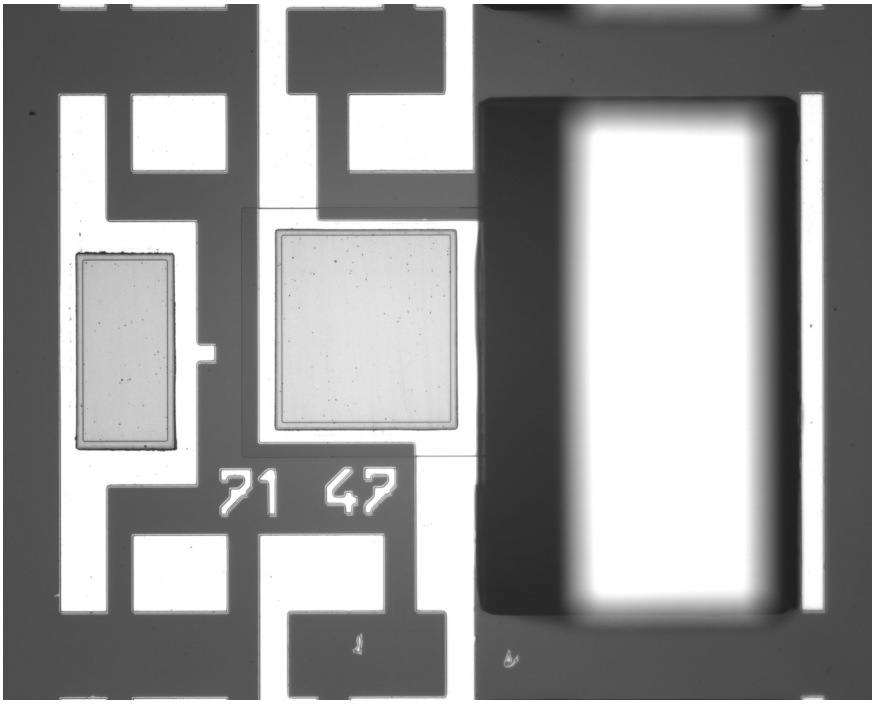


Figure 1.1: Example of my image with large textureless areas and high contrast edges.

1.2 Task formulation

The aim my master thesis was to create and to test the algorithm for 3D reconstruction in microscopic images with textureless areas based on the Shape-from-Focus algorithm. The task can be formulated as follows:

1. Study the edge detection techniques.
2. Find or create my edge detection algorithm in a sequence of differently focused images.
3. Smooth the edges positions in depth.
4. Approximate the surface between the edges and find the depth in textureless areas.
5. Remove the implausible surfaces and reassign texture/color to the plausible ones.

Chapter 2

Related work

The Shape-from-Focus algorithm is a 3D surface reconstruction technique for the optical systems with a shallow Depth-of-Field. The algorithm is also called Shape-from-Defocus or the Depth-From-Focus in literature. The algorithm has multiple modifications. The Shape-from-Focus algorithm modification could be a use of the different operators for focus measuring, minimizing the variance of the depth, deconvolution of the Point Spread Function ect.

Pertuz [2] compares some of the commonly used focus measuring operators for the Shape-from-Focus algorithm. The compared operators are the image gradient, Laplacian-based masks, gray-level variance, the wavelet transform. Pertuz suggests using the relative quality of the reconstructed surface to compare the quality of the reconstruction. The modified Laplacian has the overall best performance in low noise images. However, Pertuz states that it is hard to compare operator performance since the operator response depends on the imaging conditions.

Nayar [3] also suggests to use the sum of the modified Laplacian (MLap) for the depth estimations. Nayar says that his method can directly be applied to the smooth textured surfaces. He also states that a special illumination technique is required for the textureless surface.

Moeller [4] uses Nayar's MLap for the initial depth estimation in his work. Moller proposes a function mapping the depth function to the energy function. The resulting energy function is numerically minimized. The resulting minimized energy provides a smooth depth map. Moller also provides the implementation of the energy minimizing algorithm. However, the implementation does not behave well in textureless areas.

Faro [5] tries to find an inverse convolution of the Point Spread Function. Faro tries to find the operator with the lowest energy cost. His approach works well in the textured areas. From his experiments, we can see that his approach does not deal with textureless areas.

Suwajanakorn [6] suggests a two-step Depth-from-Focus algorithm for a device held in hand. Suwajanakorn's algorithm first aligns the images and then it performs the auto-calibration with the depth reconstruction. The calibration part is based on the guess of the focal length of a lens. The cameras used in my work are equipped with a telecentric lens and guessing its focal length is not possible (i.e. it is close to infinity).

Tang [7] tries to find a depth flow between two differently focused images. Tang's approach uses only two images and they must preserve a tiny blur condition explained in the article. My scene does not satisfy this condition if only two images of the scene are used.

The Shape-from-Focus algorithm can also be modified to extract a single fully in-focus image from the collection of differently focused images. This modification is commonly used in the micro and product photography of a static scene. This technique allows to extend the Depth-of-Field of the lens and it allows to overcome limitation caused the diffraction of the light in the iris. This technique is implemented in professional cameras and camera software nowadays.

The capability of collecting the series of defocus images is present in Nikon D850 [8]. Nikon calls the functionality as the Focus Stacking. Nikon suggests joining images together using Adobe CC software [9]. Software tool made by HeliconSoft can also be used [10].

Chapter 3

Theory and proposed solution

3.1 Edge detection techniques overview

My scene is illumined by a monochromatic light source and the camera used is also monochromatic. The images provided by this setup are intensity images $I(x, y)$.

The edges in the image are defined as the pixels with a rapid intensity change between the neighboring pixels [11]. The section of the intensity image perpendicular to the edge creates an edge function. An example of the edge function is displayed Figure 3.1. The edges in images are discrete samples of the edge on a real object in the scene. The discrete edge samples are called edge elements or *edgels*.

The edge detection process seeks for the intensity changes in the edge function to find the position of the edge or the edge element in the image. There are three main groups of edge detectors:

1. Finding the extrema of the first derivative (Prewitt [12] or Sobel [13] operators, Canny [14], etc.)
2. Finding the zero-crossing of the second derivative (Laplacian of Gaussian [15] [16])
3. Approximation of the edge or the edge function by a parametric model [17] [18]

My thesis primarily focuses on the first two groups (i.e., first and second derivatives). The reason is that the derivatives can be expressed as discrete convolutional masks and they provide a fast edge detection over the whole image.

The last group of methods is interesting when it comes to precise edge detection. Those methods provide a refinement of the edge position and the relative position of the edge must be known a priori. Those methods are well suited for the CoPA project. However, those methods are not ideal for the Shape-from-Focus with the 5M pixel images because of their computational complexity.

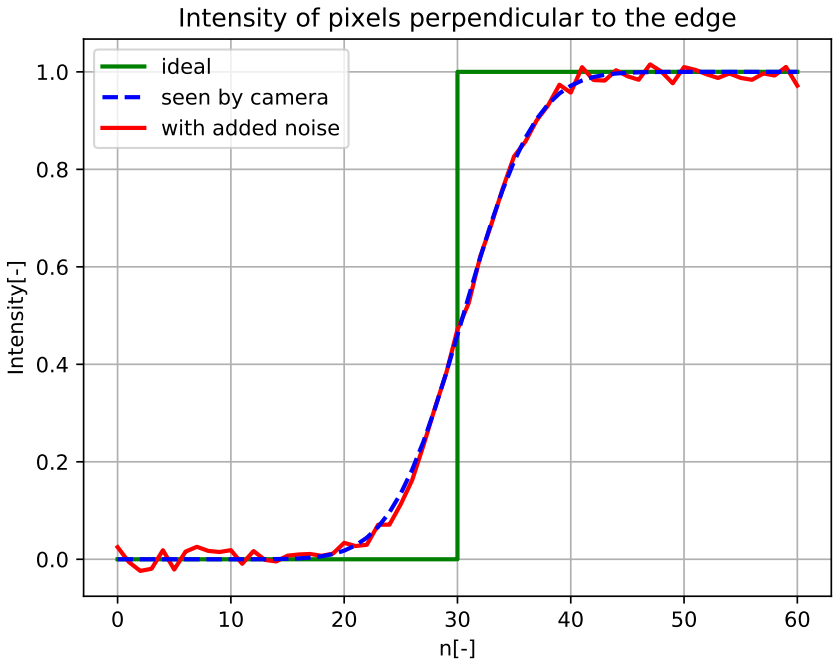


Figure 3.1: Example of the edge function, where n is a relative position in the image. The green line represents an ideal edge. The blue dashed line is an edge as seen in camera. It takes into account a defocus and optics imperfections. The red line adds a Gaussian noise to represent the noise of a camera sensor.

3.1.1 Intensity function derivative

The derivative edge detectors seek for the local extrema (i.e., maxima or minima) of the first derivative of the intensity image function $I(x, y)$. The image function is assumed to be continuous in this case. The first derivative is expressed as a gradient of the image function:

$$\nabla I(x,y) = \left(\frac{\partial I(x,y)}{\partial x}, \frac{\partial I(x,y)}{\partial y} \right), \tag{3.1}$$

where ∂ represents a partial derivative of a function. Such a function has a magnitude of:

$$||\nabla I(x,y)|| = \sqrt{\left(\frac{\partial I(x,y)}{\partial x} \right)^2 + \left(\frac{\partial I(x,y)}{\partial y} \right)^2}, \tag{3.2}$$

and a direction:

$$\phi = \tan^{-1} \left(\frac{\partial I(x,y)}{\partial x} \Big/ \frac{\partial I(x,y)}{\partial y} \right). \tag{3.3}$$

The images, which are used, are discrete functions and their derivatives can be expressed in terms of the finite differences based on the continuous case from Equation (3.1). I will take into account only the symmetrical variant of

the finite differences. The finite differences approximation of a single variable function derivative $f'(x)$ is expressed as:

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h} \approx [-0.5, 0.0, 0.5] * f[x], \tag{3.4}$$

where $*$ represent convolution, and $f[x]$ is a discrete function. Equation (3.1) is expressed in terms of the finite differences in Section 3.1.2 as a Prewitt and Sobel operators.

The problem with applying the first derivative directly to the image is the operator response to the noise. This problem can be mitigated by filtering the image before computing the derivative. The filtering of the image is usually expressed as a convolution with a filter kernel h and the image $I(x, y)$. The derivative and the convolution are commutative. Filtering the image and the derivative can be combined into a single convolutional operator (∇h):

$$\nabla(h * I(x, y)) = (\nabla h) * I(x, y). \tag{3.5}$$

An example of the derivation operator, Equation (3.4), response to edge functions from Figure 3.1 is displayed in Figure 3.2. The edge is in the middle of the graph, $n = 30$. As you can see, there are two local extrema around

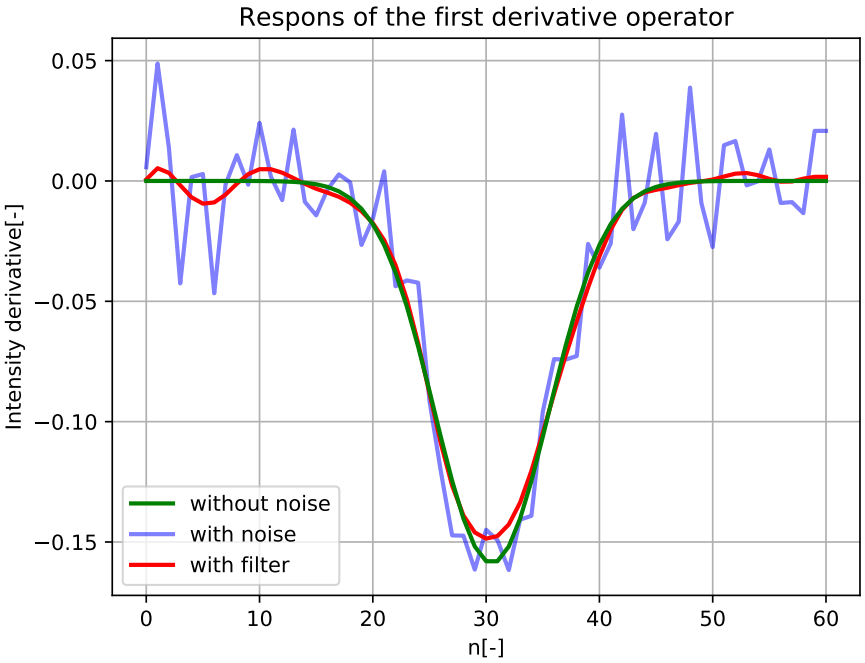


Figure 3.2: Example of the first derivative operator response to edge functions from Figure 3.1. The green line is a response to the edge function without the noise. The light blue is a response to the function with the noise. The red line is a response to the noisy edge function with a Gaussian filtering in prior to calculating the derivative.

the edge position in the response to the noisy edge function, the light blue line. The correct edge position is hidden in-between these extrema. As I mentioned before, this can be overcome by filtering the edge function before calculating the derivative; the filtered response is shown as the red line. The extrema position of response to the filtered edge corresponds to the extrema of the response to the edge without noise, the green line.

3.1.2 First derivative convolutional operators

In this section, I focus on two discrete edge detection operators, Prewitt and Sobel, and briefly describe a Canny edge detector.

Prewitt and Sobel operators

Prewitt and Sobel operators are one of the simplest convolutional masks h for an approximation of the derivative of the image $I(x, y)$. The shape of Prewitt and Sobel operator masks is usually 3×3 or 5×5. Prewitt and Sobel operator masks are expressed in the x and y direction separately and their 3×3 masks look like:

$$h_x = \begin{bmatrix} -1 & 0 & 1 \\ -\xi & 0 & \xi \\ -1 & 0 & 1 \end{bmatrix}, \; h_y = \begin{bmatrix} -1 & -\xi & -1 \\ 0 & 0 & 0 \\ 1 & \xi & 1 \end{bmatrix}, \tag{3.6}$$

if $\xi = 1$ it is called Prewitt mask, and if $\xi = 2$ it is Sobel mask. The only notable difference between them is that Sobel mask is more directionally sensitive than the Prewitt's. The masks h_i is convolved with the image $I(x, y)$ as:

$$G_i(x, y) = h_i * I(x, y), \tag{3.7}$$

where $i \in \{x, y\}$. The conjoin magnitude of the intensity change in x and y direction is given by:

$$G_I(x, y) = \sqrt{G_x^2(x, y) + G_y^2(x, y)}, \tag{3.8}$$

There are different variants of these two masks. However, they do not provide much a difference in the edge detection compare to these two, so I will not mansion them.

Prewitt and Sobel operators are usually used if one is interested in the magnitude of changes in the image and do not care about the direction of change as much.

The edges in the original image $I(x, y)$ are estimated to be on the positions where the magnitude of the intensity change $G_I(x, y)$ is sufficiently higher than the given threshold.

Canny edge detection algorithm

The Canny edge detection algorithm is one of the most popular techniques for edge detection due to its simplicity and reliability.

The Canny edge detection algorithm combines the image filtration and the gradient computation together with the edgel pooling. The pooling is more sophisticated compared to the thresholding of the gradient magnitude. The most commonly used implementation of the algorithm can be summarized into 7 steps, courtesy [11]:

1. Filter the image with a Gaussian filter with a standard deviation σ.
2. Estimate the local edge normals from each pixel.
3. Find the location of the edges via non-maximal suppression.
4. Compute the magnitude of the edge.
5. Threshold edges in the image with hysteresis to eliminate fake edges.
6. Repeat (1) to (5) for ascending values of σ.
7. Collect the final information about edges at multiple scales using a feature synthesis.

3.1.3 Second derivative edge detection

The second derivative of the image function $f(x, y)$ is expressed as:

$$\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2}\,, \tag{3.9}$$

where ∇^2 is the Laplace operator (Laplacian).

Equation (3.9) is not a vector as in a case of the first derivative, but it is a scalar. The Laplacian operator is rotation-invariant. This property allows the operator to have the same response for differently oriented edges. The edges in the image are detected at the positions where Function (3.9) crosses the zero.

The problem with the second derivative is an even higher influence of the noise compared to the first derivative. This problem can be overcome by filtering the image I with the Gaussian filter G before computing the derivative. The filtering and the derivative can be joined to create a single operator similarly as in a case of the first derivative:

$$\nabla^2 (G * I) = (\nabla^2 G) * I. \tag{3.10}$$

Finally, $(\nabla^2 G)$ is called the Laplacian of Gaussian (LoG) operator.

Usually, only the symmetrical filters are taken into account, i.e. the standard deviation σ is the same for both axis. The LoG operator is obtain from Equation (3.9):

$$\nabla^2 G(x,y) = \nabla^2 \, e^{-\frac{x^2+y^2}{2\sigma_{xy}^2}} = \frac{x^2+y^2-2\sigma_{xy}^2}{\sigma_{xy}^4}\, G(x,y), \tag{3.11}$$

where σ_{xy} is a standard deviation of the Gaussian filter in x and y axis. The discrete convolutional operator is constructed from Equation (3.11).

The shape of the operator is based on the blur effect of the filter. The recommended size of the operator mask is $\geq 4\lceil\sigma_{xy}\rceil$ [11]. The operator response to the homogeneous input must be a zero to ensure the property of the Laplacian. The operator could look like:

$$\nabla^2 G(x,y) = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 \\ 1 & 2 & -16 & 2 & 1 \\ 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}. \tag{3.12}$$

An example of the one dimensional Laplacian operator response to edge functions $f(x)$ from Figure 3.1 is displayed in Figure 3.3. The used operator looks like $\nabla f(x) \approx [1, -2, 1] * f[x]$, where $f[x]$ is a discrete edge function. The reason for using one dimension is easier visualization.

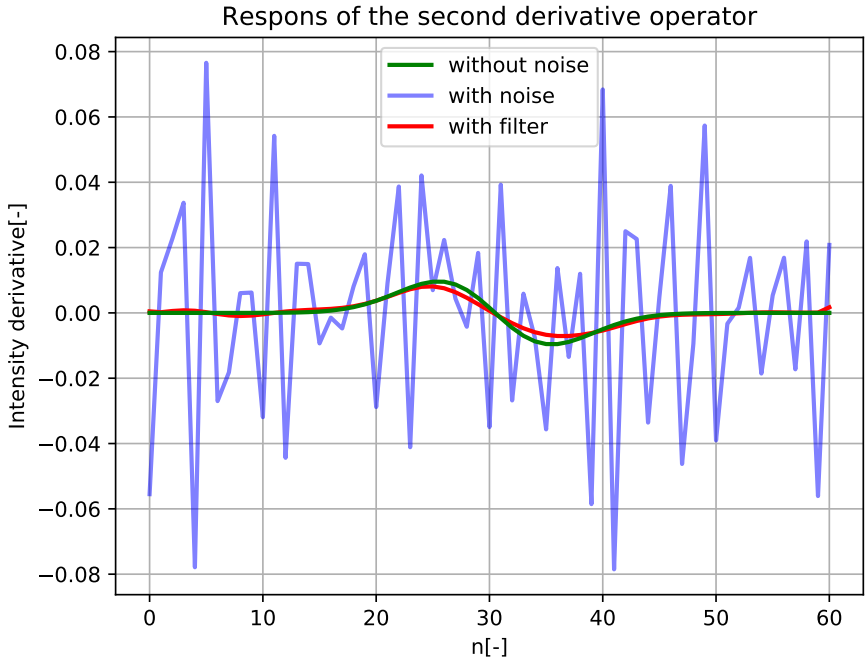


Figure 3.3: Example of the 1D second derivative operator response to edge functions from Figure 3.1. The green line is a response to signal without noise. The blue line is a response to edge with noise. The red line is a response to the noisy edge with a gaussian filtering in prior to calculating the derivative.

As shown in Figure 3.3, the response to the edge with noise (the blue line) crosses the zero multiple times with a magnitude higher than the response to the noise-free edge (the green line). Such behavior makes usage of the Laplacian operator without filtering the edge function impossible. The red line shows the response with filtering in before calculating the derivative. The response almost corresponds to the response without noise (the green line).

It could happen that some less significant zero crossings are present even after the filtering of the image function. Such a less significant zero crossing could be seen in a close proximity to $n = 0 \rightarrow 10$. The edge is on a position where the response crosses the zero with a high-enough significance, i.e. $n = 30$.

3.1.4 Parametric model approximation of edge function

The camera observes a transition between the low and high intensity value influenced by the camera chip and the optics. The influence of the camera chip and the optics can be described as a physics-based mathematical model. The parametric models build on the idea of such a model (dashed blue line in Figure 3.1) underlay the noisy edge function obtain by a real camera (red line in same Figure 3.1).

The edge functions are modeled as S curve, Polynomial or other similar curves [17]. The parametrization allows the subpixel precision of the edge detection (i.e., one can find the function extrema). The edge function can be described as a S curve, for example:

$$f(x) = \frac{L}{1 + e^{-k(x-x_0)}}\,, \tag{3.13}$$

where L stretches the function in y axis, k stretches the function in x axis, and x_0 is a position of the inflection point. The inflection point is at a position where the S curve changes a sign of the first derivative or where the second derivative is a zero. The edge position is assumed in a place of the inflection point if the optical system is considered to have a symmetrical response to the image blur, similarly to the algorithm described in Sections 3.1.2 and 3.1.3.

3.1.5 Parametric modeling of the edge shape

The edglels are detected in the image. Prior knowledge about the edge shape in the image is used to refine the edge position based on the detected edgels. The edge could resemble a line, $y = kx+c$, or a circle, $(y-y_0)^2+(x-x_0)^2 = r^2$, for example. The shape detection algorithm can be summarized into two steps.

1. Find the edge element candidates using one of previously mentioned edge detection methods.

2. Find the optimal model parameters of the edge shape based on the candidates.

There are also more complex models such a facets [18]. They build on the idea that the pixel's neighborhood can be represented as a piecewise continuous function. However, I will not describe them any further because they are not used in my work in any way.

3.2 Camera descriptions

3.2.1 Telecentric lens

The camera used in my experiments is equipped with a telecentric lens. Telecentric lens behaves differently comapred to the conventional lens. Figure 3.4 shows the difference between the conventional lens and the telecentric one.

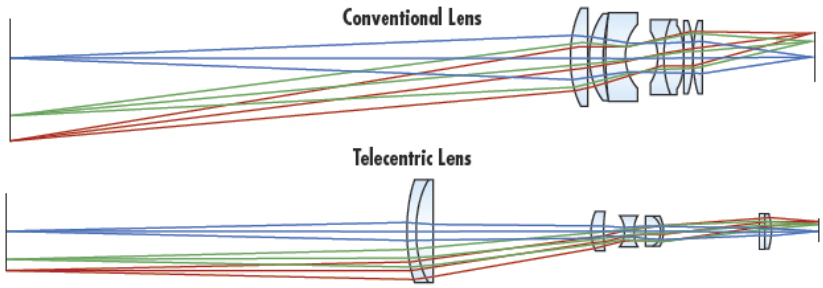


Figure 3.4: Comparison of the Field-of-View between a conventional lens and a telecentric one. Courtesy of [1].

The conventional lens replicates a lens in an animal eye. The eye lens has an angular Field-of-View (FoV), which means the magnification decreases with the distance. This property allows animals to perceive depth. However, if the object is observed from two different distances, its size in the image changes. The conventional lens could be modeled by a central projection [11]:

$$\alpha\vec{u} = \mathbb{K}\vec{X}, \tag{3.14}$$

where $\vec{u} = (u, v, 1)^\top$ is a point in the image in homogeneous coordinates, $\mathbb{K}$ is a 3×3 camera matrix, $\vec{X} = (x, y, z)^\top$ is point in a scene, and α is a scalar.

The telecentric lens, on the other hand, has a non-angular FoV, which means that the magnification remains the same for all distances. The telecentric lens model could be described by a homography H:

$$s\vec{X} = H\vec{X}' = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \vec{X}', \tag{3.15}$$

where $\vec{X} = (x_i, y_i, 1)^\top$ are the coordinates in the image, $\vec{X}' = (x'_i, y'_i, 1)^\top$ are the coordinates of corresponding points in the object space, s is a scalar.

The H parameters are obtained by calibration. The calibration pattern with known dimensions could be used to do such a thing. Equation (3.15) can be rewritten for a known position of the calibration pattern:

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} x' & y' & 1 & 0 & 0 & 0 & x'x & y'x & 1 \\ 0 & 0 & 0 & x' & y' & 1 & x'y & y'y & 1 \end{bmatrix} \bar{H}, \tag{3.16}$$

where $\vec{X} = (x_i, y_i, 1)^\top$ are coordinates of ith calibration points in the image, $\vec{X}' = (x'_i, y'_i, 1)^\top$ are the coordinates of corresponding points on the calibra-

tion pattern, and $\bar{H} = (h_{11}, h_{12}, h_{13}, h_{21}, h_{22}, h_{23}, h_{31}, h_{32}, h_{33})^\top$. The homography is found from at least 4 calibration points by solving Equation 3.16.

3.2.2 Lens distortion

The camera lens can cause a non-linear distortion in the image. The distortion will convert lines on the viewed object to curves in the image. The distortion must be evaluated and corrected to provide an accurate measurement.

The simplest way how to evaluate the distortion is to find a linear mapping, described in Section 3.2.1, between a flat calibration pattern and the corresponding image points. The distances between the calibration points in the image and the points projected by the linear mapping will identify if any lens distortion is present or not.

The used lens has almost no non-linear distortion as it turns out in the experiment in Sec. 5.1.3. I decided to not use the non-linear distortion model. So, I will not describe the non-linear model for distortion correction. The radial and the tangential distortion models can be found in [11].

3.3 My Shape-from-Focus algorithm

First, I will describe my algorithm flow and later on, I will focus on each of the individual points of the algorithm. The algorithm can be summarized into 6 steps:

1. Create the image stack 3.3.1 from the image sequence.
2. Convolve the image stack with my 3D convolution mask and extract the edge depth map.
3. Smooth the edge depth map with the median and mean filter over connected edge elements in the local neighborhood.
4. Compute full triangulation over the edge depth map.
5. Remove implausible triangles.
6. Assign the depth and the original texture to all elements of the edge depth map covered by leftover triangles for the visualization.

3.3.1 Image stack

The images are taken as a sequence in which the Depth-of-Field flows from the closest to the furthest point in the scene. Each image in the sequence corresponds to a different position of the camera. The image stack is created from the image sequence. The images are stacked at the top of each other in such way that their orientation is the same and their order is given by their position in the sequence. The image stack will be called to as $I(x, y, z)$ in the following text. The x and y axis of the image has the origin in the top left

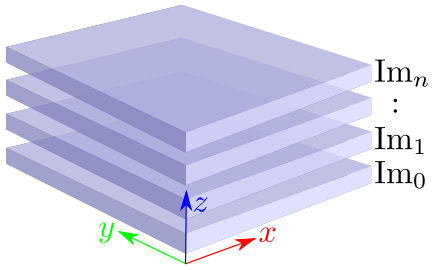


Figure 3.5: Sketch of the image stack $I(x, y, z)$.

corner. The x and y axis represent the rows and the columns of the image respectively. The z coordinate of the image stack represents the ascending depth at which the image was captured. The image stack sketch is displayed in Figure 3.5.

3.3.2 My convolutional mask

My convolutional mask seeks for the zero crossings in the second derivative of the image stack $I(x, y, z)$. The convolutional mask is spanning over all three dimensions of $I(x, y, z)$ and its kernel is three dimensional (3D). I will focus on how to build such a convolutional mask.

My Shape-from-Focus algorithm is inspired by a theoretical shape of the Point Spread Function (PSF) [19] [20] [21] and Laplacian of Gaussian (LoG) operator, see Section 3.1.3. The PSF is a physics-based mathematical model describing how the point-like source or object response looks in the imagining system. The PSF is a complex function in its nature. However, the PSF can be approximated by a simpler function.

The PSF approximation used in the book '*Principles of Optics: Electromagnetic Theory of Propagation, Interference and Diffraction of Light*' [19] is parametrized by three parameters. The parameters are the emission wavelength, the numerical aperture, and the refractive index of the immersion medium. These imaging system parameters are hard to obtain and the parametrization is also too complex for my purposes. Articles [20] [21] model the PSF as a Gaussian mixture. Article [20] also suggest that the PSF can be approximated by a single Gaussian.

I decided to use the single Gaussian approximation:

$$\begin{aligned} PSF_{app}(\vec{X}) &= C_{app}\ \exp\left(-\frac{x^2+y^2}{2\sigma_{xy}^2} - \frac{z^2}{2\sigma_z^2}\right) \\ &= (8\pi^3\sigma_{xy}\sigma_z)^{-0.5}\ \exp\left(-\frac{x^2+y^2}{2\sigma_{xy}^2} - \frac{z^2}{2\sigma_z^2}\right), \end{aligned} \tag{3.17}$$

where C_{app} is a normalization constant, σ is a standard deviation of the Gaussian. The σ is assumed to be the same for x and y axis and different for the depth axis z.

The second derivative of the image $I(x, y)$ is given by Equation (3.9). The derivative is extended into 3rd dimension z, and its equation is:

$$\nabla^2 I(x, y, z) = \frac{\partial^2 I(x, y, z)}{\partial x^2} + \frac{\partial^2 I(x, y, z)}{\partial y^2} + \frac{\partial^2 I(x, y, z)}{\partial z^2}, \tag{3.18}$$

where $I(x, y, z)$ is the image stack.

With the PSF approximation, I can now move on to creating the convolutional mask combining Equations (3.17) and (3.18). The convolution mask $ELOG$ is given by:

$$\nabla^2 (PSF_{app} * I(x, y, z)) = (\nabla^2 PSF_{app}) * I(x, y, z) = ELOG * I(x, y, z), \tag{3.19}$$

where $ELOG$ is derived using the associative property of the convolution and the derivative. The normalization constant C_{app} in PSF_{app} is independent of the position in the mask $\vec{X}$ and it does not add any useful information for creating the convolutional mask so it can be omitted. The reason for omitting C_{app} is that I am looking for a magnitude of the change and multiplying the response of $ELOG$ mask by constant changes the threshold for which the edgels are considered significant enough by the same amount. The $ELOG$ convolutional operator equation writes as:

$$ELOG = \frac{\left(\sigma_z^4 \left(x^2 + y^2\right) - 2\sigma_z^4 \sigma_{xy}^2 + \sigma_{xy}^4 \left(z^2 - \sigma_z^2\right)\right)}{\sigma_z^4 \sigma_{xy}^4} \exp\left(-\frac{x^2 + y^2}{2\sigma_{xy}^2} - \frac{z^2}{2\sigma_z^2}\right). \tag{3.20}$$

The discrete convolutional mask is found based on the Equation (3.20) The resulting mask must preserve the properties of the second derivative. The important property of the mask $ELOG$ is the zero response to the homogeneous input I_h. This can be achieved by calculating the response R of the mask to I_h and by subtracting R/n from each element in $ELOG$, where n is the number of the elements in $ELOG$. This approximation is sufficient for my purposes.

3.3.3 Extracting the edge depth map

The edge depth map $d(x, y)$ is a sparse matrix which represents the depth z of the detected edges in each element. The depth element is marked as edgeless if the element does not contain any detected edge. The depth map $d(x, y)$ reduces the z axis of the image stack $I(x, y, z)$ into a single value representing the depth of the edgels. Each element of $d(x, y)$ contains at most one depth. The $d(x, y)$ does not allow to represent multiple edges in a single element.

The edge depth map $d(x, y)$ is extracted after the image stack is convolved with $ELOG$ mask using the Equation (3.19). The stack convolved with $ELOG$ is called as $L(x, y, z)$. The $ELOG$ mask represents the second derivative. The edges in the stack are the positions where the response to the mask crosses the zero between the neighboring pixels with enough steepness.

The zero crossings are found by shifting all pixels of $L(x, y, z)$ in xy plane by one in all plausible positive direction. All neighboring points to the point (x, y) are at positions $(x + 1, y)$, $(x, y + 1)$, and $(x + 1, y + 1)$. The rest of neighboring pixels are overlapping with the previous ones[1]. The shifted stack is refered to as $L_{>>i}(x, y, z)$, where $i \in \{x, y, xy\}$ specifies the direction of the shift. The edges in the stack are on positions where two conditions are satisfied:

1. The sign of $L(x, y, z)$ differs from the sign of $L_{>>i}(x, y, z)$. The implementation should also take in account $L(x, y, z) = 0$.

2. The significance value $||L(x, y, z) - L_{>>i}(x, y, z)||$ is higher than the given threshold. The threshold value is set according to the noise and signal ratio.

The elements which do not satisfy those two conditions are marked as edgeless.

3.3.4 Median and mean smoothing of the depth map

I suppose that the ideal edge and its edge elements (edgels) are smooth and continuous in all directions. The elements of the edge depth map $d(x, y)$ estimated in previous Section 3.3.3 are adjusted according to those assumptions.

The elements in $d(x, y)$ have some deviation in depth caused by the convolutional mask response to the noise in the image. The deviation of a single edgel can be mitigated based on the locally connected edgels. The depth smoothing algorithm is suggested. The algorithm also removes too small edges during its process. The algorithm is described as:

For all edgels $\boldsymbol{p}_i$ in $d(x, y)$ do:

1. Find all edgels connected to $\boldsymbol{p}_i$ in the given mask. The 8 neighborhood connection is considered and the size of the mask is given by the user.

2. If the number of connected edgels is smaller then minimal length of the edge than remove all of those edgels and continue with the next edgel.

3. Update the depth of $\boldsymbol{p}_i$ using the median or mean depth of the connected edgels.

The algorithm runs twice over the depth map $d(x, y)$. In the first run, the algorithm removes points which are far away from the neighboring ones in depth using the median. In the second run, the algorithm smooths the depth using the mean. The result is a new filtered depth map of edges $d_f(x, y)$.

[1] If the point $(x - 1, y - 1)$ is considered as a starting point, the point (x, y) is present in its neighborhood so the negative shifting can be omitted.

3.3.5 Surface approximation

My images contains a lot of edges and flat textureless surfaces in-between them as it was discussed in Section 1.1. I make use of this assumption about the surface flatness and connect all the occupied elements of the filtered depth map $d_f(x, y)$ from Section 3.3.4 to get a surface approximation in textureless areas.

As I mentioned before, a single element of $d_f(x, y)$ either contains a depth of the edgel or it is marked as edgeless. The x and y positions of occupied elements in $d_f(x, y)$ create a set of 2D points Ξ. The indexes from Ξ to $d_f(x, y)$ are kept in a set χ.

I use the Delaunay algorithm [22] to create triangulation of the surface. Delaunay triangulation connects all points in a given point set Ξ by non-occluding triangles. The triangulation creates a convex surface. Delaunay triangulation maximizes the minimal angle of all the possible triangles. Each triangle created by the triangulation is described by three indexes to the Ξ. The depth of the triangles verities is given by $d_f(x, y)$ using χ.

3.3.6 Removing of implausible triangles

My images are taken from the top view over the scene and the telecentric optics are employed. Due to these reasons, some triangles obtained by Delaunay triangulation introduced in Section 3.3.5 are not observable in the real conditions and they will be removed. The non-observable triangles are the ones which are almost perpendicular to the wafer foundation and their sides are almost as long as the height of the observed components.

The triangles are described by three points A, B, and C. The normal vector of the triangle $\vec{n}_t$ is:

$$\vec{n}_t = (n_{tx}, n_{ty}, n_{tz})^\top = (C - A) \times (B - A). \tag{3.21}$$

I suppose that the wafer foundation lies in a plane that is parallel with the xy plane of the image stack $I(x, y, z)$ and its normal vector is $\vec{n}_f = (0, 0, 1)^\top$. The angle ϕ between the $\vec{n}_t$ and $\vec{n}_f$ is given by equation:

$$\cos(\phi) = \frac{\vec{n}_t \times \vec{n}_f}{||\vec{n}_t||\,||\vec{n}_f||} = \frac{n_{tz}}{||(C - A) \times (B - A)||}. \tag{3.22}$$

The condition for removing the almost perpendicular triangles is $|\cos(\phi)| < t$, where t is chosen based on the maximal acceptable triangle angle, for example $t = 0.1$.

The orientation of triangle vertices is not specified nor is their normal vector. The triangles have two normal vectors $\vec{n}_t$ which are collinear and they are facing the opposite direction (i.e. their angle differs by 180°), so the triangles with ϕ close to $-90°$ must also be removed. The condition is given by a cosine of the angle, so it works for both normals the same.

3.3.7 Reassign the texture to triangulated surface

The final step of the algorithm is to assign the depth and the original texture to missing elements of the filtered depth map $d_f(x, y)$ from Section 3.3.4 using the filtered triangles from Section 3.3.6. The result of this algorithm is a new depth map $d_c(x, y)$ and a texture map $c(x, y)$. The position x and y in $c(x, y)$ corresponds to the same positions in $d_c(x, y)$. The shape of $d_c(x, y)$ and $c(x, y)$ is the same as the shape of a single image $I(x, y)$.

Each element of $d_c(x, y)$ is checked if it lies inside any triangle. If $d_c(x, y)$ is inside of the triangle, it is assigned with the depth. The triangle is imagined as a plane in 3D:

$$(a, b, c)^\top S + d = 0, \tag{3.23}$$

where $\vec{n}_p = (a, b, c)^\top = (C - A) \times (B - A)$, $S = (x, y, d_c(x, y))^\top$, and $d = -\vec{n}_p^\top A$. The depth $d_f(x, y)$ is given by Equation (3.23):

$$d_c(x, y) = -\frac{d + ax + by}{c}. \tag{3.24}$$

The elements of $c(x, y) = L(x, y, \lfloor d(x, y) \rceil)$, where $\lfloor \bullet \rceil$ is rounding to the integer value.

Chapter 4

Implementation and Hardware description

4.1 Assembly machine description

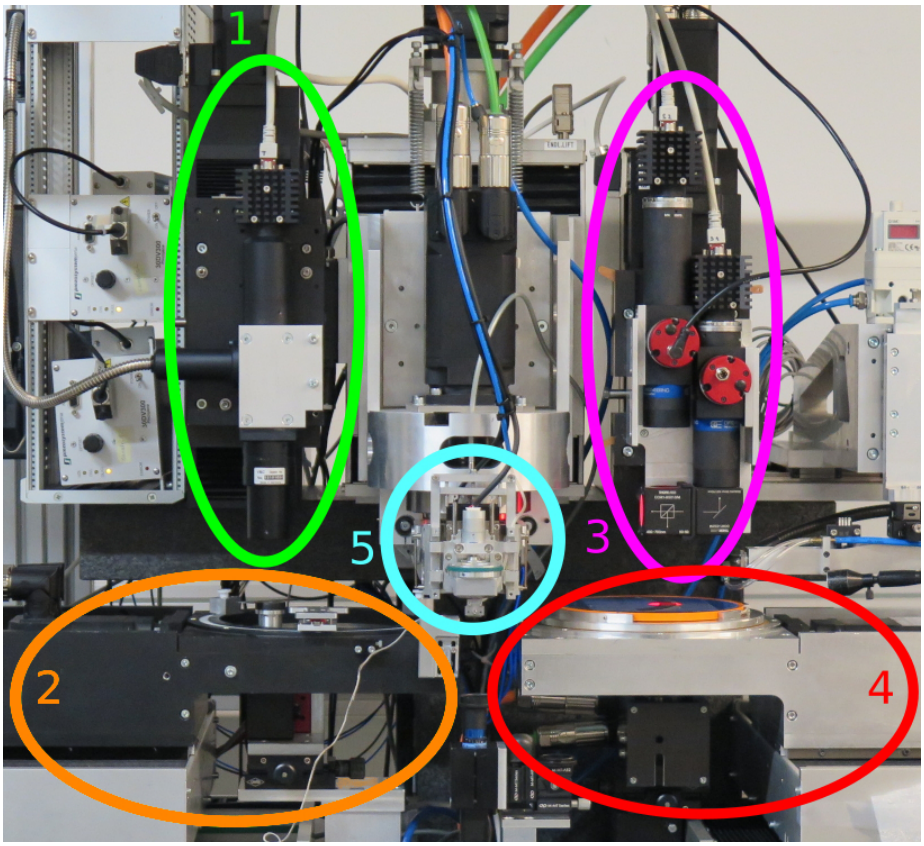


Figure 4.1: Assembly machine. The target camera (1) is in the green ellipse, the target wafer bench (2) is in the orange ellipse, the dual cameras for finding the components (3) are in the magenta ellipse, the bench with component wafer (4) is in the red ellipse, and the component picker (5) is in the cyan circle.

The micro-module assembly machine was provided by an industrial partner via a project called Control Platform for High-Accuracy Microelectronics (CoPA). My thesis uses the machine's optical system.

You can see a picture of the assembly machine in Figure 4.1. The machine is laying on a big marble block. The marble block mass makes the machine resistible to small vibrations caused by the movement of the axis and it allows the sub-micron precision placement of the components.

The machine has two working benches with three linear axes, and a rotary axis with a wafer holder at the end. The bench in the red ellipse carries a wafer with components (source wafer). The bench in the orange ellipse holds a target wafer. The benches are designed in such a way that the image plane is always parallel with the wafer plane.

The component and the target wafer are observed by cameras. Two cameras in the magenta ellipse are looking at the component source. Each of these two cameras has a different magnification. One on the right has 0.5× magnification. It scans a larger area, finds and provides a map with missing components. The other one has 3.5× magnification. It provides an exact position of the component for the picker. The camera in the green ellipse is observing the target wafer. The target camera has a 10× magnification. The target camera has a shallow Depth-of-Field compared to the height of the components, which is well-suited for the Shape-from-Focus algorithm. All cameras are equipped with a telecentric lens.

The picker, the cyan circle in the picture, picks the component, rotates them into the correct position, places them and holds them at the assembly position until the component is welded by a laser.

Dramatical changes to the provided machine configuration were not possible. On the other hand, it was possible to connect a notebook to the machine network, control the movement of the axes, and capture images from the installed cameras.

(a) : Camera. Courtesy of [23]

(b) : Telecentric lens.

Figure 4.2: Camera and lens used on the target side.

Target camera and lens description

The camera is shown in Figure 4.2a. The machine uses JAI GO-5000M-PGE camera [23] equipped with the telecentric lens on the target side. The camera have a 1 inch sensor with 2560×2048 pixels.

Figure 4.2b shows the lens. The lens magnification is 10×. The Field-of-View for the 1 inch camera sensor is 1.2×1.0 mm. The lens was used on the precedent machine and no technical description of it is known.

The lens consists of three elements. The elements are the tubus with a C mount at the top, the central part with the light source and the beamsplitter, and the tubus with lenses at the bottom. The light source is telecentric (i.e. its light rays are parallel). The beamsplitter serves as a reflective mirror for the light source. The beamsplitter aims the light rays coming from the side to the direction of the optical axis of the lens.

4.2 Software

The algorithms proposed in Section 3 were implemented in Python 3.5. C++ is used for communication with the assembly machine and for capturing the images. The 3D models are exported to Polygon File Format (PLY). The software will be described here only briefly because it not the primary subject of my thesis.

Python language is chosen because of its wide availability, its large number of libraries, and its open-source nature. The used libraries are SciPy with NumPy [24], SkImage [25], OpenCV [26], and MatPlotLib [27]. The libraries provide most of the needed mathematical operations and they also take care of the visualization of results.

The cameras used on the machine are based GigE vision protocol implemented by Pleora [28] called eBUS. eBUS is C++ SDK which provides full control over the camera. The communication with the machine axes is possible via ZeroMQ library [29]. The communication messages are requests describing where to move the specific axis, and replies describing if the movement was successful or not.

In its simplest form, the PLY format describes the point cloud with colors and the connection between those points using triangles. PLY description allowed me to export my depth and texture map with little to no adjustment. Python PLY library is available as open source under GNU General Public License, version 3. The open-source mesh processing tool MeshLab [30] could be used to open the PLY files.

Chapter 5

Experiments

I prepared experiments regarding the topic of the edge detection, and the results of the surface reconstruction obtained by my Shape-from-Focus algorithm. Edge detection experiments provide a simplified background to the Shape-from-Focus algorithm. The experiments are performed using the camera on the target side of the assembly machine.

Two edge detection experiments are related to the lens distortion evaluation and the relative distance between the camera and the observed edge by the sharpness of a given region.

The first experiment described in Section 5.1.2 shows how moving the Depth-of-Field affects the value of sharpness of the edge. The first experiment finds the sharpest image with a calibration pattern for evaluating the amount of distortion in the optical system. The first experiment is also interesting for the CoPA project. It shows the capability of the machine to refocus the image during the operation without the intervention of an operator.

The second experiment described in Section 5.1.3 evaluates the optical system distortion. The evaluation of the optical distortion is important to preserve straight lines in the scene. The evaluation of the optical distortion also plays an important role in the high accuracy measuring and it is needed in the CoPA project.

The last set of experiments (Section 5.2) covers the Shape-from-Focus algorithm. Here, I include the images used for creating reconstructed surfaces. Subsequently, I present the response of the *ELOG* mask to the image stack. Finally, I discuss the reconstruction results.

5.1 Edge detection

The dot calibration pattern is used for testing the edge detection techniques and for the camera calibration. The pattern is described later in Section 5.1.1. The calibration pattern is assumed to be almost parallel with the image plane.

The images were taken by the camera at different distances from the calibration pattern. The images create the same image stack $I(x, y, z)$ as the one described in Section 3.3.1. The number of images taken was 50. The spacing between each image (z direction) in $I(x, y, z)$ was $0.320\,\mu$m. Figure 5.1 shows an example of these images.

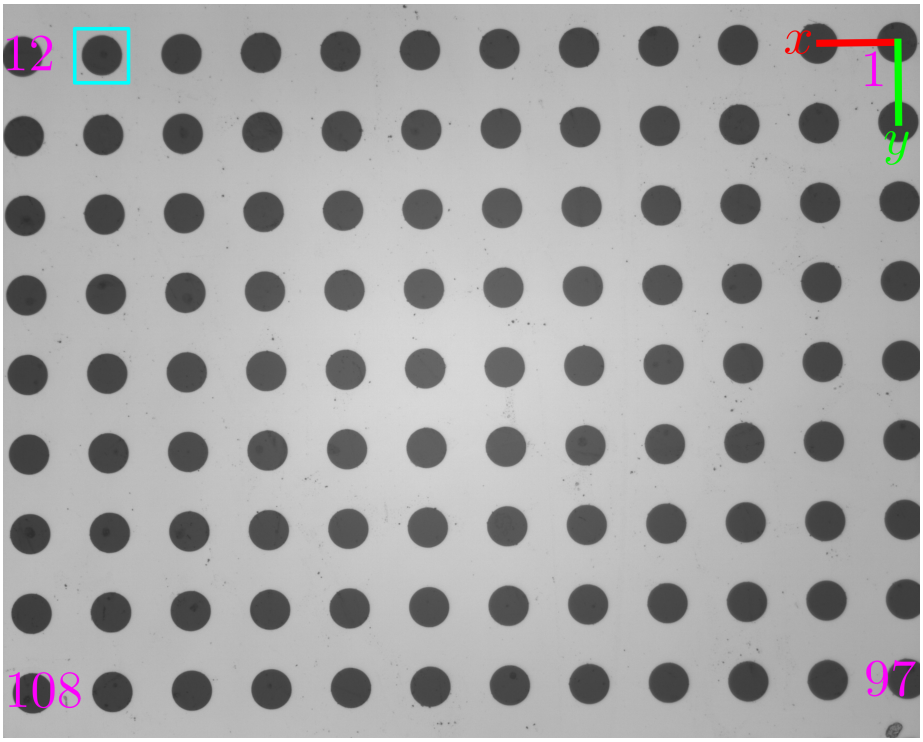
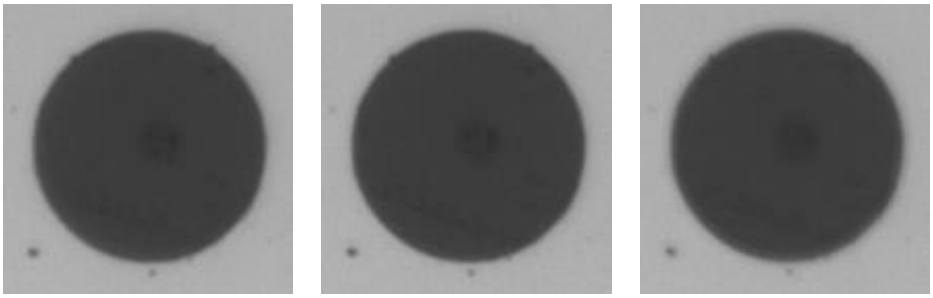


Figure 5.1: Picture of the calibration pattern. The pattern is an array of dots with the diameter 62.5 μm and the spacing 125 μm. The cyan square shows the region where experiment from Section 5.1.2 was performed. The x and y axes shows the chosen calibration pattern coordinate system. The magenta numbers denotes the ordering of calibration dots used in Section 5.1.3



(a) : $n = 0$ **(b) :** $n = 24$ **(c) :** $n = 50$

Figure 5.2: Cyan region from Figure 5.1 zoom in. The cuts are taken from the different positions n of the camera. This Figure illustrates the difference between the focused and almost focused images.

5.1.1 Calibration target

The used calibration pattern is a multi-frequency grid distortion target made by Edmund Optics [31]. The pattern has multiple sizes of dots and multiple spacings between them. The central part of the pattern with the smallest dots is used for the experiments due to the limited Field-of-View.

The dots used for the experiments have a diameter of 62.5 μm and their spacing between centers is 125.0 μm. The camera Field-of-View is 1.2×1.0 mm and it fits in 12×9 dots. The manufacturing tolerance of the dot diameter is ±2.5 μm. The tolerance of the spacing between centers is also ±2.5 μm. The tolerances are given with regards to the manufacturing process. The manufacturing process errors are not random and they fluctuate during the production. The true geometrical errors are smaller and they are bellow the manufacturing ones.

5.1.2 Depth from the sharpness of the region

This experiment shows how the image blur is used to find the sharpest image in the sequence or to estimate the depth. The sharpest image is used for the distortion evaluation and it is used to prove the concept of the Shape-from-Focus algorithm. As I mentioned before, some methods could be employed for focusing the image during the machine operation.

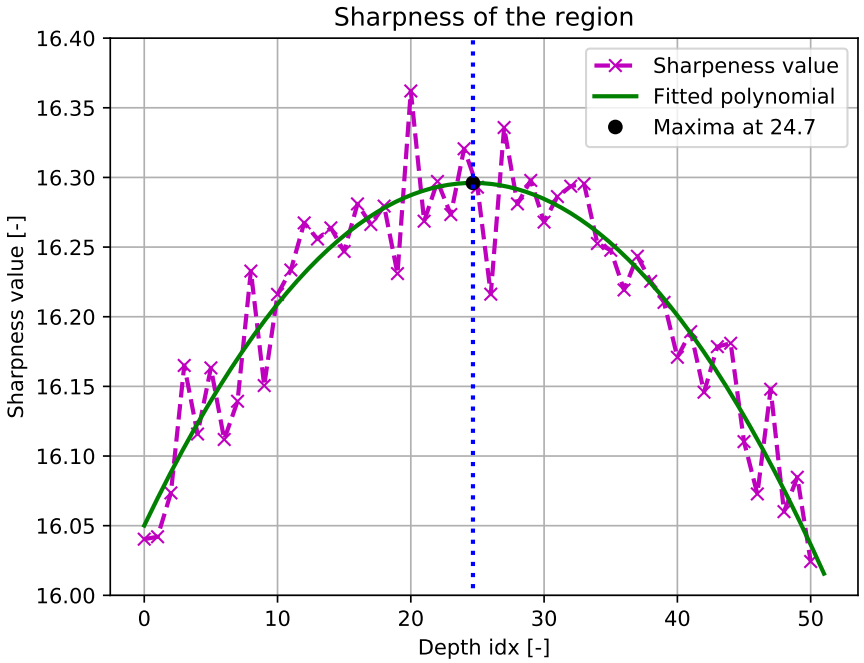


Figure 5.3: Sharpness of the region. The dashed magenta line shows the sharpness value Υ. The green line is a fitted parabola. The dotted blue line is a position of the parabola maxima.

Figure 5.2 shows the zoom-in of the green region from Figure 5.1. This region of the image was used to evaluate the sharpness in this experiment. The sharpness of the region was given by a sum of the magnitude of gradients $G_I(x, y)$ obtained by Sobel filter described in Section 3.1.2.

The sharpness of the region Υ was given as:

$$\Upsilon = \frac{1}{s} \sum_{x,y} G_I(x, y), \tag{5.1}$$

where $\sum_{x,y}$ denotes the sum over all elements in the region, and s is a number of elements in the region.

The Υ for the given region in the stack is show in the Figure 5.3 as a dashed magenta line. The Υ is influenced by the noise in the image. The noise can be mitigated by fitting an appropriate function over the data. The graph of Υ resembles a parabolic function in its local maxima. I decide to fit the parabola to the data (the green line) and find its local maxima. The parabola equation is $y = ax^2 + bx + c$, and its local maxima is at $x_{max} = -b/(2a) = 24.7$.

The maximum is used to identify the sharpest image in the stack. The experiment identified the best focused image as the image at the position $n = 24$, or $n = 25$.

5.1.3 Lens distortion

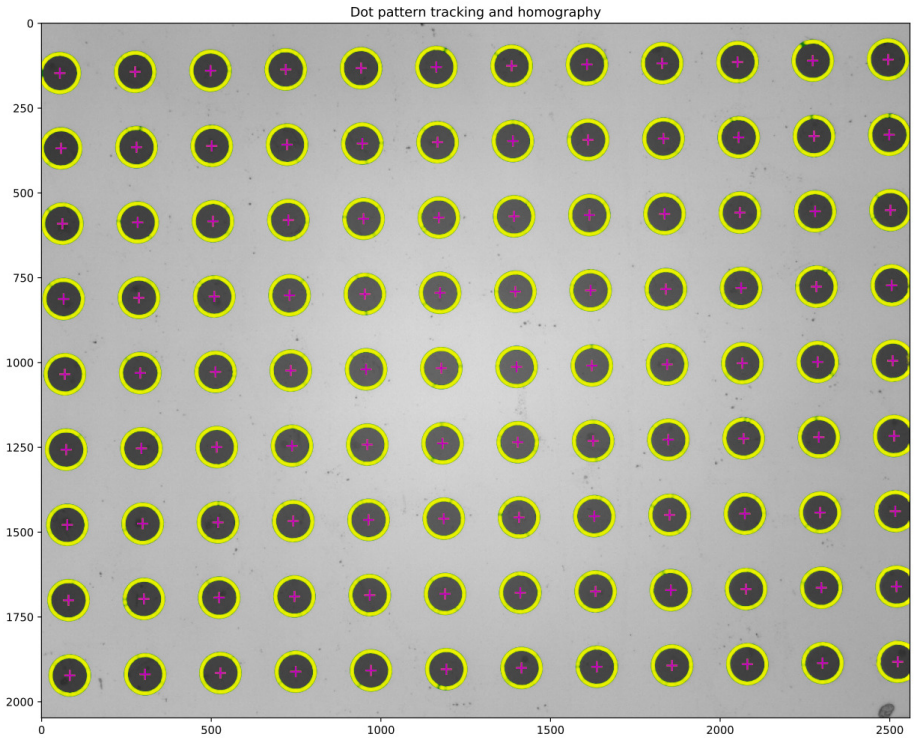


Figure 5.4: Tracked dot calibration pattern. The green dots are the candidate edgels. The yellow line shows the circles fitted to candidate edgels. The magenta cross shows the centers of fitted yellow circles

In this experiment, I seek for the camera calibration and the evaluation of the optical distortion. I used the dot calibration pattern described in Section 5.1.1 to do job. I placed the calibration pattern perpendicularly to

the optical axis. The precision of the placement is given by the machine tolerances. For calibrations, I used the sharpest image found in previous Section 5.1.2. The image is shown in Figure 5.1 together with the dots coordinate system and the ordering of the dots pattern.

In order to calibrate the camera, the dots must be localized in the image first. The simplified dot pattern localization algorithm goes as follows:

1. Find the edges in the image using the Canny edge detector, see Section 3.1.2.
2. Find the approximate circle center positions using extended Hough transformation [32].
3. Find the edgels of the circle by extracting the edge function among multiple lines. The line is spanning from the center estimated by Hough transformation in equidistant angle around the circles and the edgels are detected via the maximal magnitude of the first derivative.
4. Fit a circle in the edgels found in the previous point (3).
5. Return the radius and the center of all fitted circles.

The center and diameter of the circles found by the algorithm are used in the evaluation of the distortion. The localized circles and their estimated centers are shown in Figure 5.4. The circle centers are arranged to the coordinate system shown in the Figure 5.1. The circles radius mean is 56.16 pixel, its standard deviation is 0.063 pixel, its maxima is 56.30 pixel, and its minima is 55.88 pixel. Such a low deviation of the radius indicates a correct detection and localization of the dots.

The homography between the calibration pattern and the image is estimated using Equation (3.15) and (3.16). The resulting homography is:

$$H = \begin{bmatrix} -1775.21 & 29.33 & 2494.78 \\ 28.90 & 1775.52 & 106.92 \\ -2.29 \cdot 10^{-4} & 2.01 \cdot 10^{-4} & 1.00 \end{bmatrix}. \tag{5.2}$$

The h_{31} and h_{32} are close to the zero, which means that the homography represents an almost pure rotation and translation. The h_{11} and h_{22} are similar up to the decimal place and the sign, so the scaling in both axis is considered the same. Those properties of H indicate that the pattern was placed almost perpendicularly to the optical axis.

The h_{11} or h_{22} is used to convert the dot diameter from pixels r_{px} to millimeters r_{mm}:

$$r_{mm} = \frac{r_{px}}{h_{11}}. \tag{5.3}$$

Using only the h_{11} or h_{22} is sufficient because the value of h_{12} and h_{21} is low compare to h_{11} or h_{22}. Equation (5.3) is used to evaluate the dimension of the calibration pattern given by manufacturer in Section 5.1.1. The dot diameter is $62.5 \pm 2.5\,\mu$m. The obtained radius mean is $63.24\,\mu$m, its standard

deviation is 0.03 μm, and the difference between the smallest and the largest radius is 0.2 μm. These dimensions are below the manufacturing tolerances of the calibration pattern.

The dot grid given by the manufacturer is reprojected to the image using the homography in Equation (5.2). The mean distance between the localized dots centers and the centers projected by homography is 0.137 pixel, its standard deviation is 0.081 pixel, and its maxima is 0.418 pixel. All distance magnitudes are well under 0.5 pixel and it indicates that there is almost no distortion present in the image. The errors emphasized 500 times are shown in Figure 5.5 as arrows spanning from the localized centers to the projected ones. The arrows do not have any preferred direction. Probably, the most of the observed displacements are caused by the noise in the image. So, even if there is any distortion, it is almost impossible to correct without improvements to the optical system.

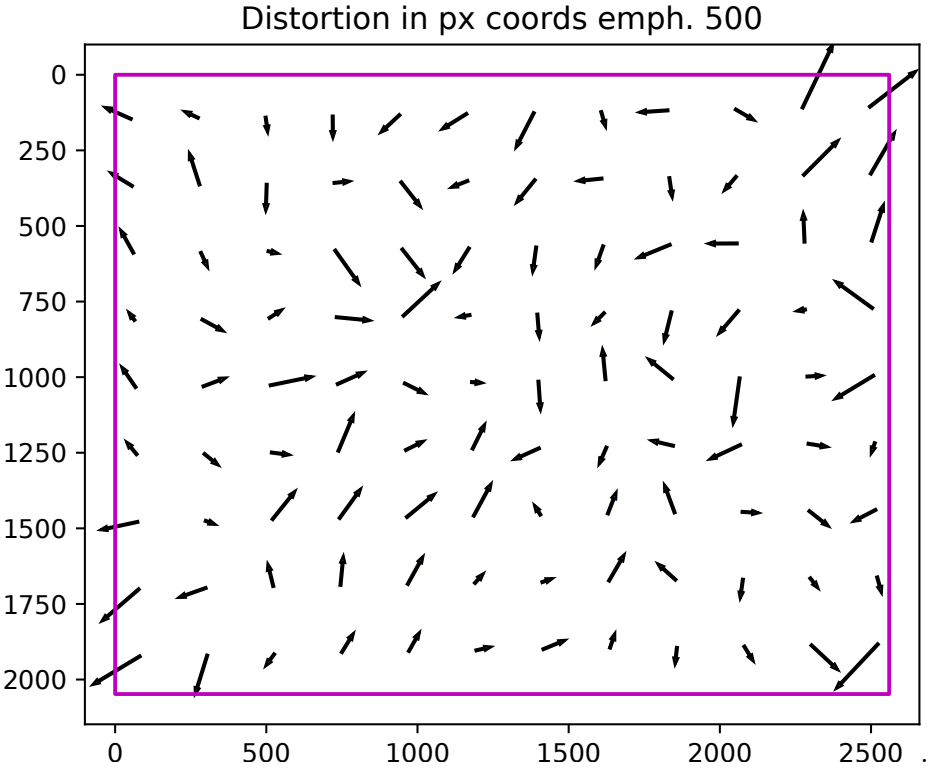


Figure 5.5: Arrows show the error direction between the centers of dots found by pattern tracking and the centers of dots projected to image by the homography. Their size is emphasized 500 times. The magenta box shows the original image shape.

5.2 Shape-from-Focus

I prepared the set of experiments regarding my Shape-from-Focus algorithm. I used two scenes to demonstrate the capabilities of the algorithm. First, I describe the layout of both scenes. Afterwards, I show and discuss the shallow Depth-of-Field of the optical system in the images of the second scene. I demonstrate the response of my 3D convolution mask ($ELOG$). Finally, I discuss the results of the Shape-from-Focus reconstruction are shown.

5.2.1 Scenes description and used images

First scene

The first scene is a flat circuit board with the dust and scratches. The images of the scene are captured as the images stack described in Section 3.3.1. The camera captured 70 images in equidistant distances. The distance between two images in the stack is 8 μm.

Figures 5.8 show one focused image in depth 20 and one blurred image in depth 40. This scene shows how the Shape-from-Focus algorithm preserves the flatness of the surface and how the algorithm can capture the height of imperfections (i.e. dust and scratches).

Second scene

The second scene is an integrated circuit board with micro-components. The scene is more complex and I will describe it in more details.

The sketch of the scene side view is in Figure 5.6. The scene has four components in it. The components are a laser (the blue box) and a monitoring diode (the green box), a reflective mirror (the magenta shape), and a lens (the black shape). The components are placed on the wafer foundation (the red line). The foundation is the integrated circuit board. The components and their wiring are welded or glued to the circuit board.

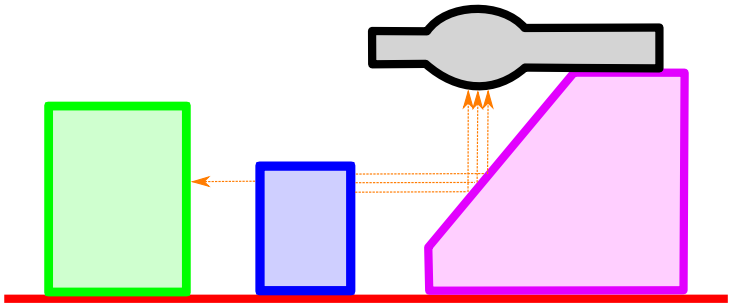


Figure 5.6: Sketch of the scene side view. The green box represents monitoring diode. The blue box represents laser diode. The magenta shape represents reflective mirror. The black shape represents lens. The orange arrows represents the light emission of the laster diode. The red line is a foundations.

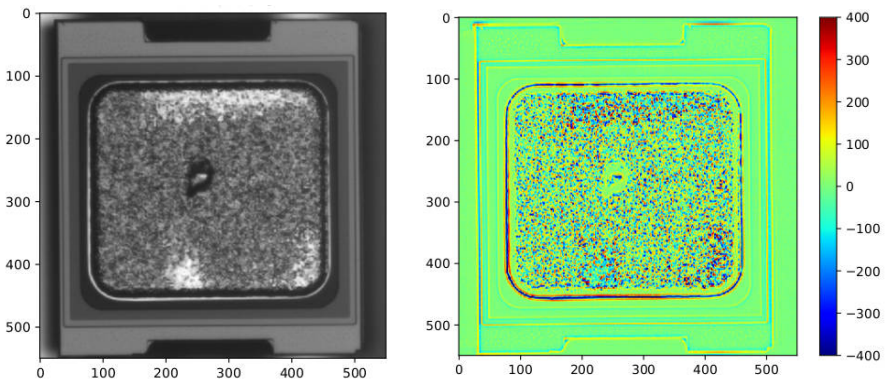
The 150 images are captured in equidistant heights spanning 480.0 μm. The captured images create the image stack $I(x, y, z)$ described in Section 3.3.1. The height difference (z direction) between two images is 3.2 μm. The tallest component, the mirror, has 220.0 μm. The lowest point of the image stack is set 180.0 μm below the foundation. The highest point of the image stack is set 180.0 μm over the top surface of the mirror.

Four examples of images from the stack are shown in Figures 5.11 and 5.12. The monitoring diode is on the left side of the image. The laser diode is in the middle of the image. The laser diode has a wire and a number 137 on the top. The reflective mirror is on the right side of the image. The integrated circuit was not completed. The lens on the top of the mirror and some wiring are missing.

The light source rays are emitted directly from the lens and are perpendicular to the foundation. The reflective mirror is placed in such a way that it reflects the light from the side of the laser diode. Only the reflexion of the laser diode side is visible on the mirror surface. The mirror surface it self is invisible.

The images in Figures 5.11 and 5.12 show that there are many high contrast edges and a lot of textureless area in between. It is assumed that the area between the edges is flat. The images also show that the Depth-of-Field (DoF) of the telecentric lens is tiny compare to the components height. These properties of the scene and images are important when it comes to my Shape-from-Focus algorithm.

5.2.2 Response of convolutional mask



(a) : Monitoring diode **(b) :** Response

Figure 5.7: Response of my convolutional mask in the region of monitoring diode.

The experiment demonstrates the response of the convolutinal mask $ELOG$ from Section 3.3.2 to the monitoring diode region in the image stack $I(x, y, z)$ of the second scene.

The mask shape and its parameters must be found in first. Multiple parameters and shapes of the mask were tested. The best fitting shape $x \times y \times z$

of the convolutional mask was 7×7×11. The variance parameters were $\sigma_{xy} = \sigma_z = 1.0$.

The convolutional mask behaves similarly to the second derivative. The second derivative of the edge function crosses the zero at the position of the edge. So the mask response around the edge should also cross the zero.

Figure 5.7a shows the region of the monitoring diode on the 68th image of the $I(x, y, z)$. The edges of the monitoring diode are clearly visible in the image. Figure 5.7b shows the response of the $ELOG$ mask to the diode region. The color transition between yellow and red is a positive value response of the mask. The shades of blue are the negative value responses of the mask. The shades of green are the responses around the zero.

Comparing the image in Figure 5.7a with the mask response in Figure 5.7b, it is visible that, the zero crossings are at same positions as the edges (transition between blue and red color), and the zero responses are in the textureless areas (green color). As I said before, this behavior of the $ELOG$ mask is expected and the mask acts similarly to the second derivative.

5.2.3 Surface reconstruction

I will present the results of the surface reconstruction. The surface is obtained from the Shape-from-Focus algorithm suggested in Section 3.3. The shape and parameters of the used convolutional mask are the same for both scenes. The shape $x \times y \times z$ of the mask is 7×7×11. The mask variance parameters are $\sigma_{xy} = \sigma_z = 1.0$.

Both reconstructed surface models are included on the attached CD, see Appendix A.

First scene

Figures 5.9 show the results of the surface reconstruction. The reconstructed surface looks flat from a distance. No large triangles in the middle of the scene are missing. Some triangles on the fringe are missing. The missing triangles are caused by the undefined response of the convolutional mask around the fringe regions. The mask estimates the edge in wrong heights and the triangles are removed in the filtration process.

Figures 5.10 show the details of the surface with and without the texture. The estimated surface is flat with some peaks and wallies. The peaks and wallies are caused by dust and scratches on the surface. Some missing triangles are also visible. The removed triangles are almost perpendicular to the scene. These triangles were removed in a triangle filtration process. When the surface with and without texture are compared, it is visible that the texture hides some imperfections of the surface.

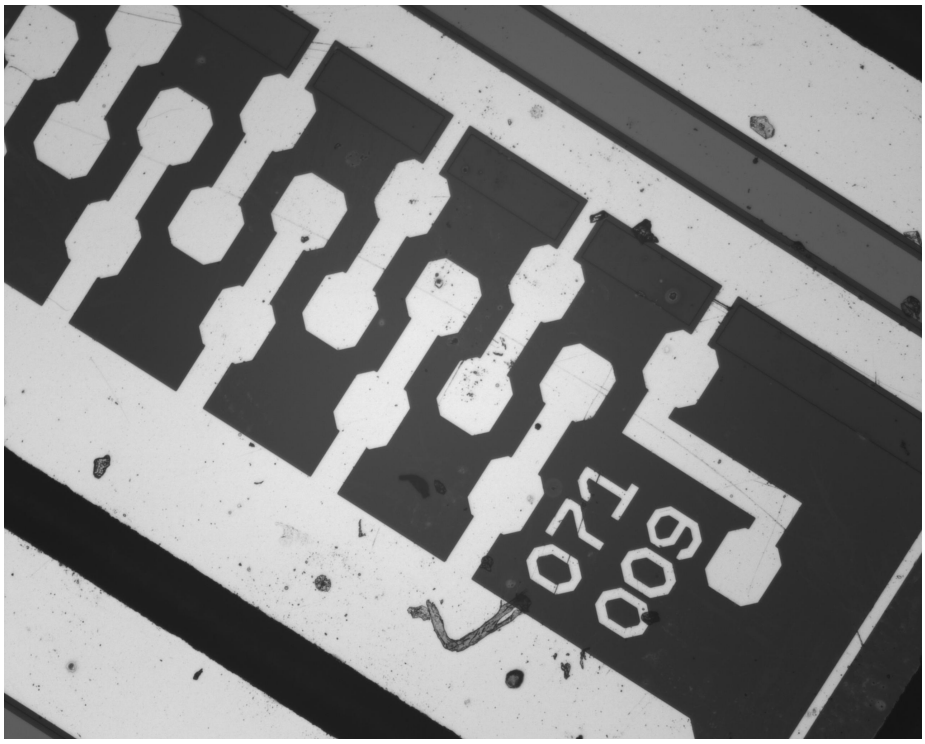
Second scene

The reconstructed surface is in Figures 5.13. The components are described in Section 5.2.1, the depth of metal plating, and the laser diode top metallic

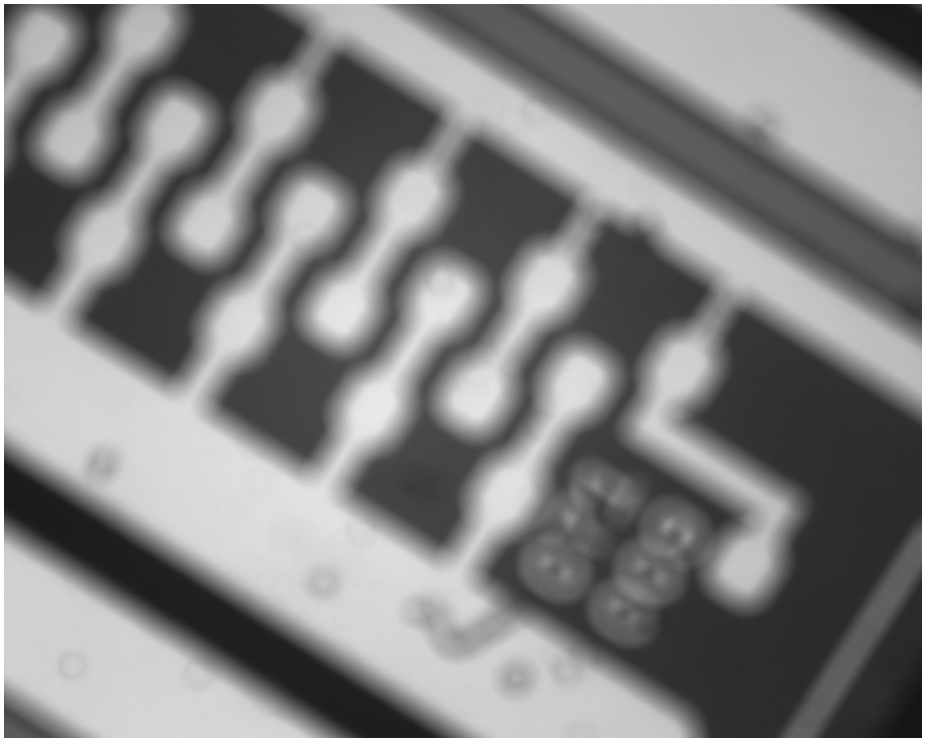
contact are clearly visible. The monitoring diode is on the left side, the laser diode with wiring is in the middle and the top of the mirror is on the right side in Figure 5.13a. The the triangles corresponding to sides of components are removed in a triangle filtration. The removed triangles are almost parallel to the optical axis and their side are too large.

Figure 5.14a shows the detail of the circuit board in the region of the metal plating. The black region is a wafer foundation surface. The metal plating surface is white. The metal plating is above the foundation surface. This behavior is expected due to the manufacturing process of the circuit board. Figure 5.14b shows the detail of the laser diode top metallic contact. The top contact is directly above the laser emission place of the diode. It is visible that the top contact is above the top surface of the laser diode.

There is a wire at the top of the laser diode. The wire connects the diode and the foundation. The wire is made out of gold, it has a circular section and it is glossy. The glossy cylindrical surface of the wire makes it hard to detect. The wire is approximated only by its sides. The depth map provides only a single depth for each element. The wire occludes the foundation and it makes foundation invisible in that part. The triangles connected to the side of the wire were removed in the triangle filtration.

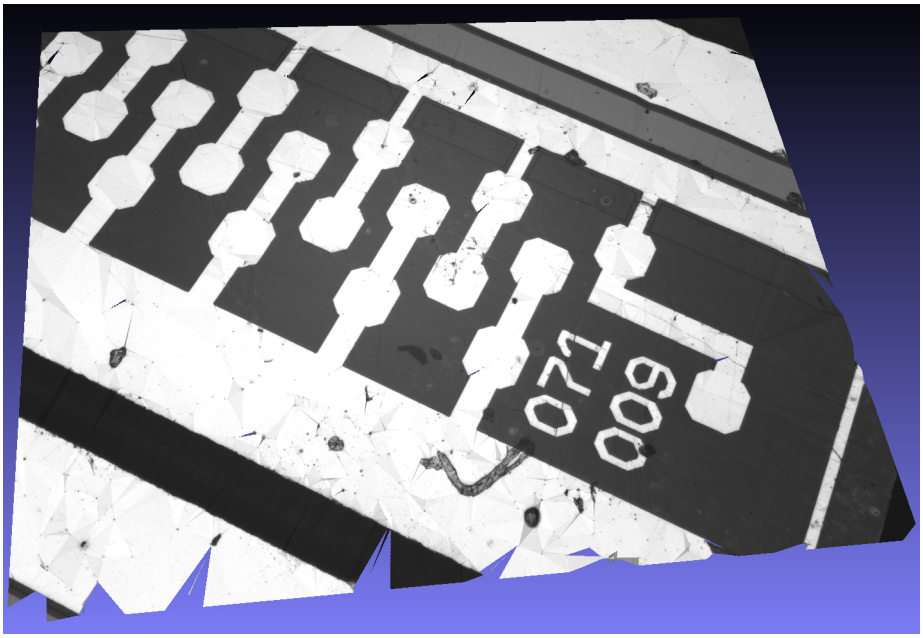


(a) : Focused surface.

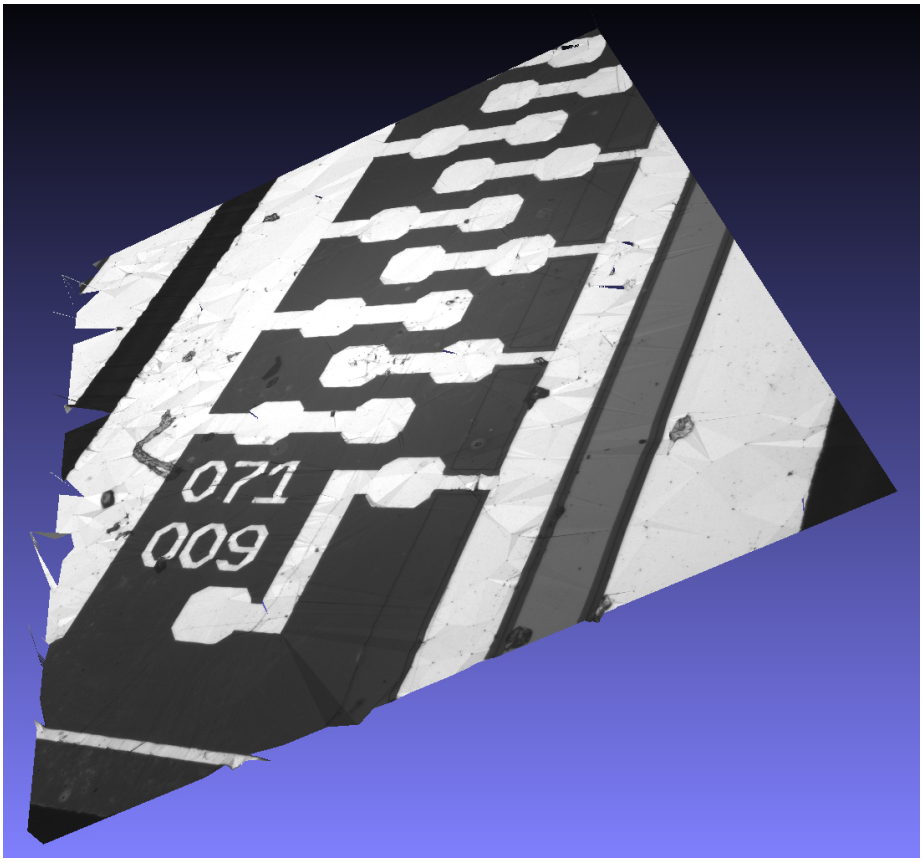


(b) : Defocused surface.

Figure 5.8: Examples of images used as and input to my Shape-from-Focus algorithm. Flat surface with dust and scratches.

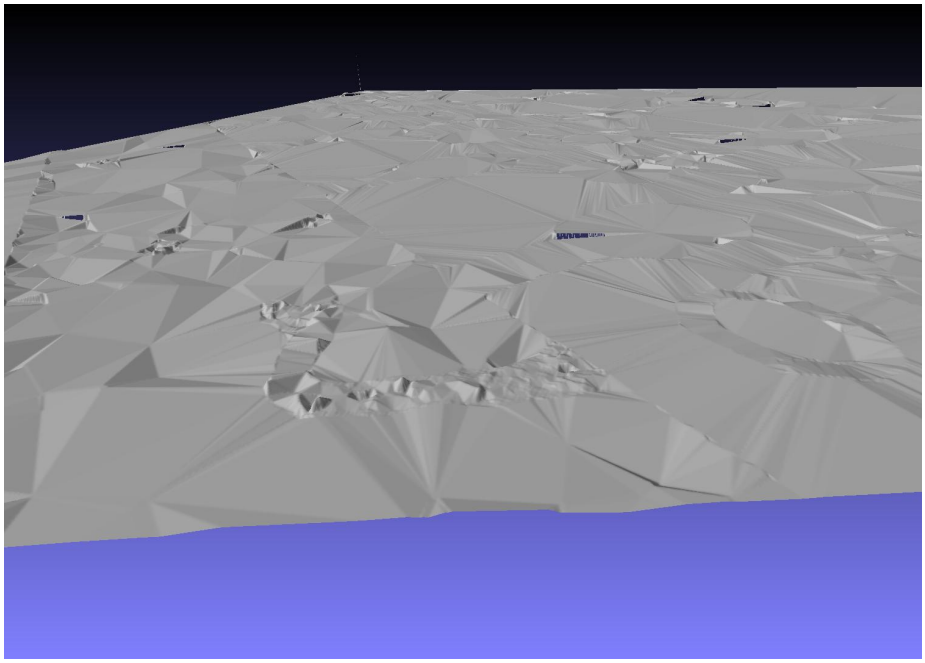


(a) : Surface without texture.

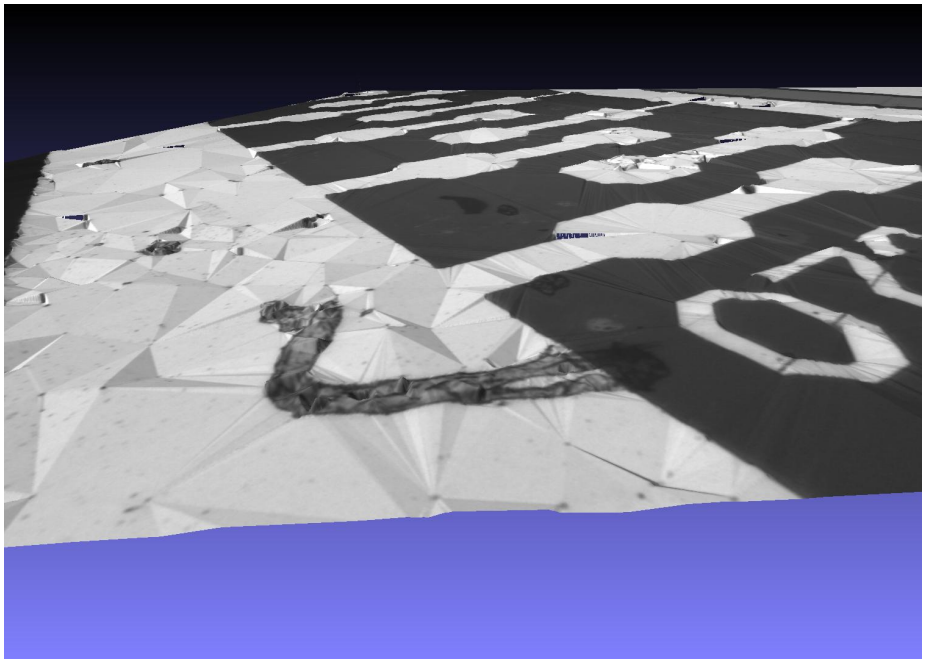


(b) : Surface with texture

Figure 5.9: Surface of the testing circuit board with dust and sketches.

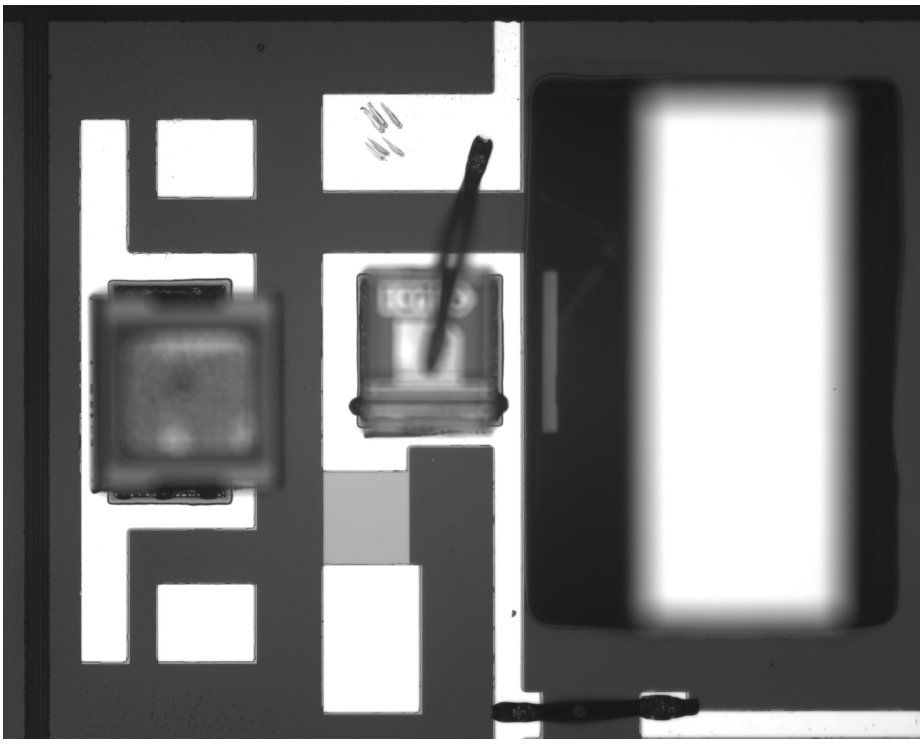


(a) : Surface without texture.



(b) : Surface with texture

Figure 5.10: Detail of the surface of the flat circuit board with dust and sketches.

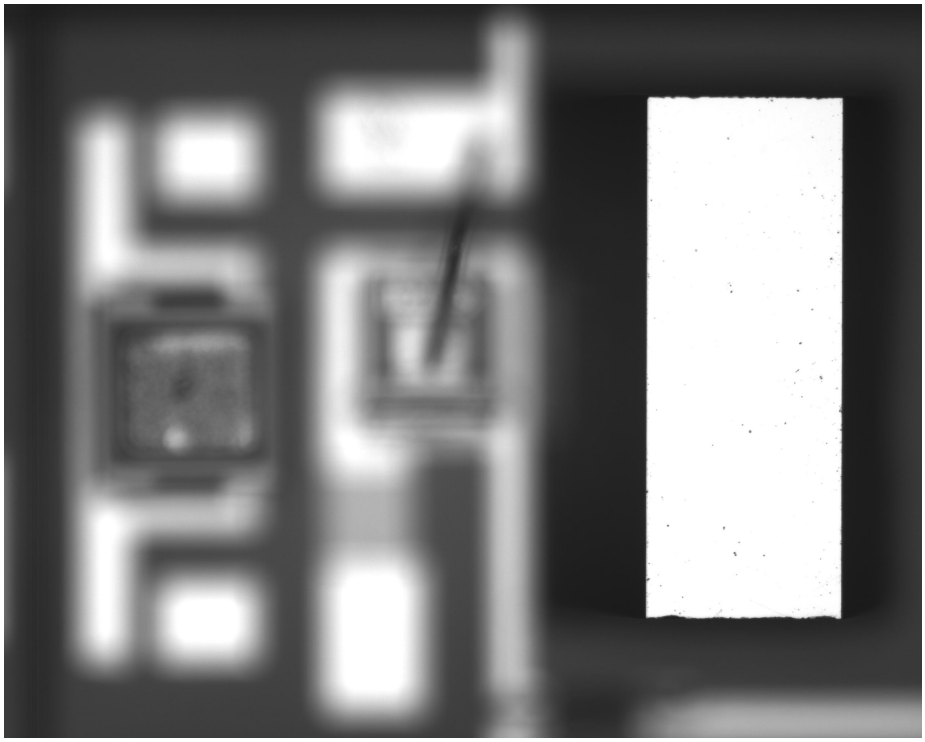


(a) : The wafer foundation is in focus.

(b) : The laser diode is in focus.

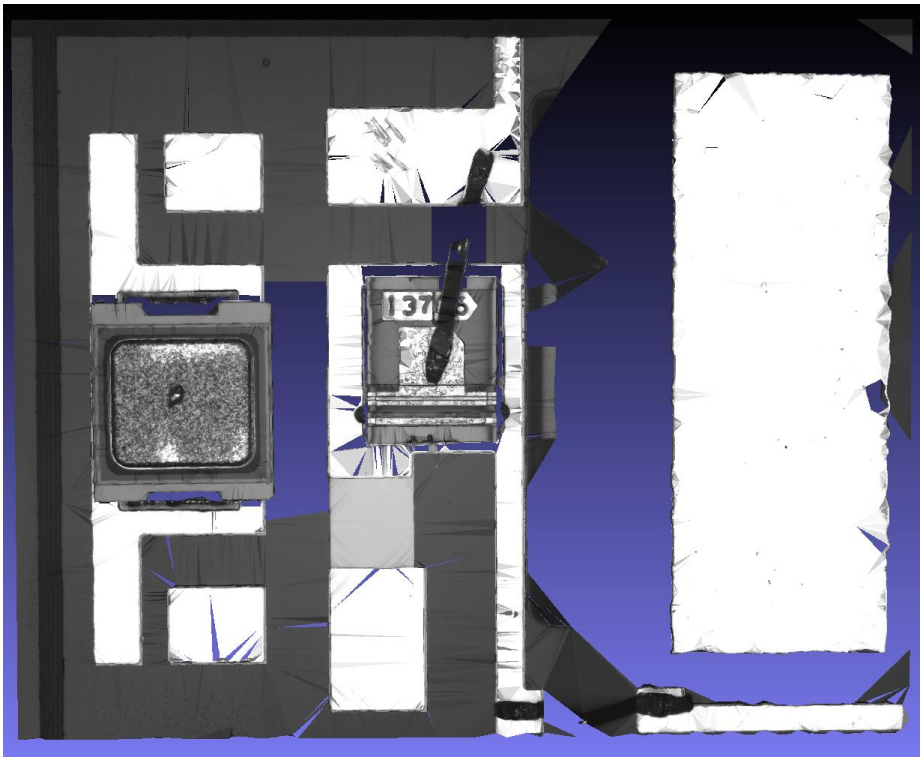
Figure 5.11: Examples of images used as and input to my Shape-from-Focus algorithm. Circuit board with components. Part 1.

(a) : The monitoring diode is in focus.

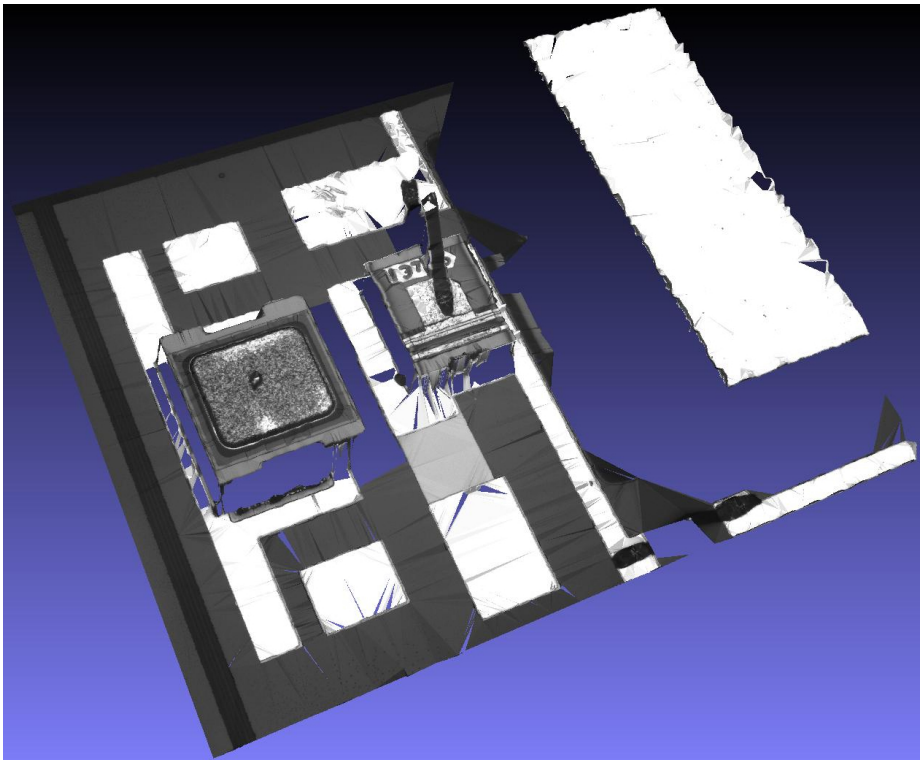


(b) : The top of a mirror is in focus.

Figure 5.12: Examples of images used as and input to my Shape-from-Focus algorithm. Circuit board with components. Part 2.

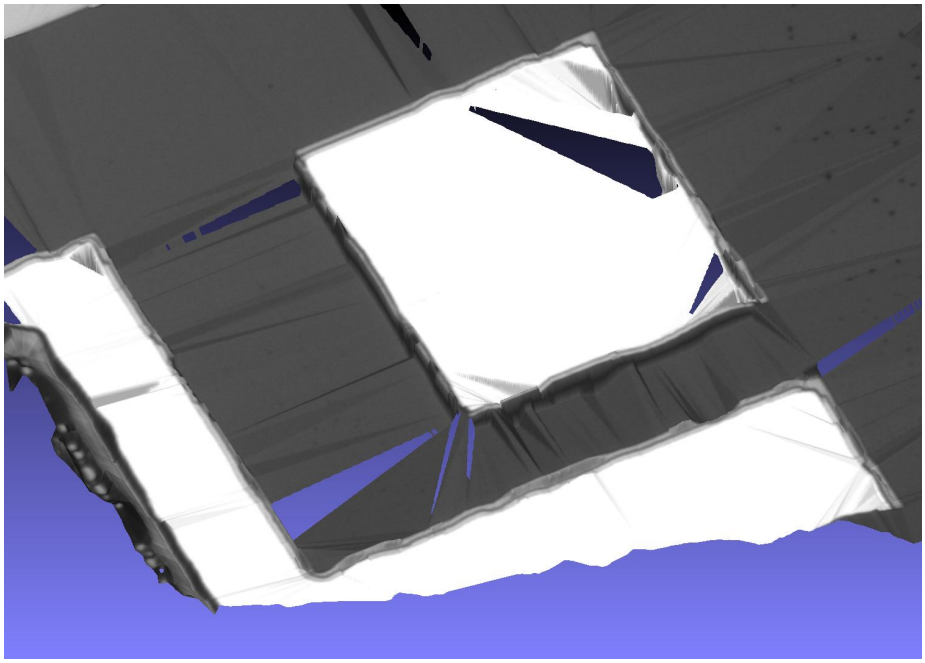


(a) : Top view.

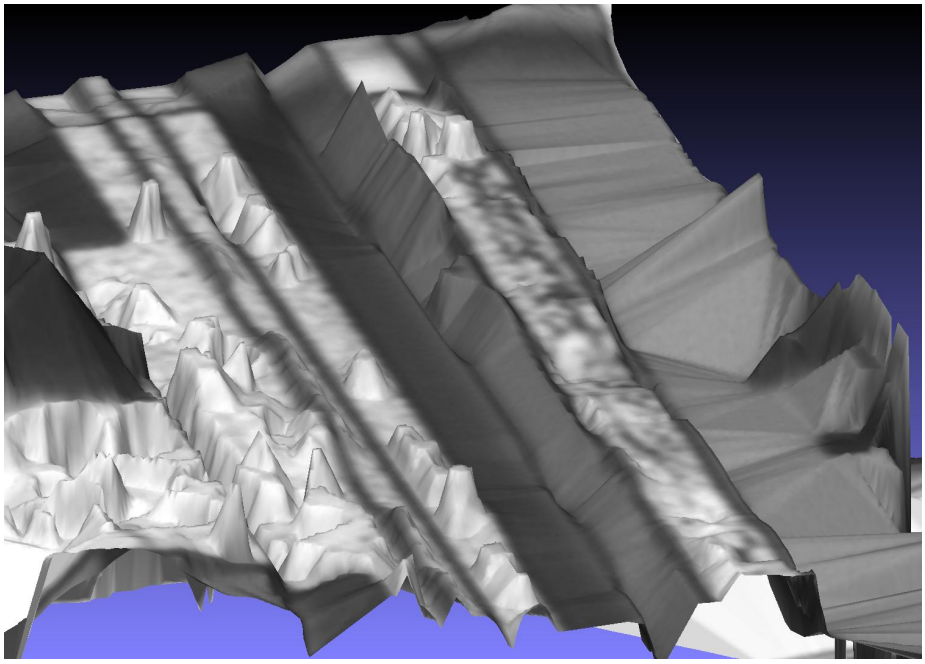


(b) : Bottom left corner view (according to (a)).

Figure 5.13: Surface of the circuit board with micro-components in a different views.



(a) : Metal plating.



(b) : Laser diode top metallic contact

Figure 5.14: Detail of the surface of the circuit board with components.

Chapter 6

Conclusions and future work

I had two main tasks in my thesis. The first task was to study different edge detection techniques. The second task was to develop the Shape-from-Focus algorithm for the images with high contrast edges and large textureless areas.

The thesis summarizes the commonly used edge detection techniques and the design of the Shape-from-Focus algorithm. The described edge detection techniques serve as an introduction to the Shape-from-Focus algorithm.

The edge detection experiments show how the edge detection can be used to focus the camera and they also show how the edge detection is used in the calibration of the optical system.

The main contribution of my thesis is the design of the Shape-from-Focus algorithm, which uses the second derivative 3D convolutional mask, and which provides the depth estimation in the textureless areas. Another contribution of my thesis is the calibration of the optical system on the provided machine.

Edge detection

The described edge detection techniques were the first and second derivative (Section 3.1.2 and 3.1.3) of the image function, and parametric models for the edge approximation (Section 3.1.4 and 3.1.5). Edge detection techniques were demonstrated in two experiments.

The first experiment (Section 5.1.2) showed that it is possible to extract the depth of the image region using the sum of the gradients. The experiment also showed that the sharpness value was affected by the noise and some final approximation was needed.

The second experiment (Section 5.1.3) showed how the edge detection is used in the optical system calibration. The experiment used a dot calibration pattern. The dots were tracked by Hough transformation. The center position was refined using the parametric model of the edge (Section 3.1.5). The optical system distortion was evaluated using homography between the dot pattern in a scene and the dot pattern projected into the image. The mean distance between the center of the dot projected from the scene using the homography and localized centers in the image was 0.137 pixel. The experiment showed that there is almost no non-linear distortion in the optical system.

Shape-from-Focus algorithm

The design of the 3D convolutional mask was discussed in Section 3.3.2. The mask approximated the second derivative and it was inspired by the theoretical shape of the Point Spread Function. The mask was used to estimate the position of the edges in depth in the image stack (Section 3.3.1). The estimated edge positions created the edge depth map (Section 3.3.3). The edge depth map had a single entry for each pixel and it was treated as an image (Section 3.3.3). The edges in the depth map were smoothed using median and mean filtering (Section 3.3.4). The surface approximation was obtained by Delaunay triangulation (Section 3.3.5). The final surface was filtered and reassigned with the texture (Section 3.3.6 and 3.3.7).

Models of two different surfaces obtained by the Shape-from-Focus algorithm were presented in Section 5.2.3. The first surface was a flat circuit board with dust particles and scratches. The second surface was a partially assembled circuit board. The first surface demonstrated the ability of the Shape-from-Focus algorithm to preserve the flat surface and the height of imperfections at the same time. The second surface showed that the Shape-from-Focus algorithm could find a surface in different heights and it can also preserve the roughness of their surface. Both surface models are included on the attached CD, see Appendix A.

Future work

One idea for achieving a better surface reconstruction is to replace the developed 3D convolutional mask with a convolutional neural network (CNN). I expect that the CNN will better describe the influence of the neighboring edges and it will improve the edge detection. The influence of the neighboring edges is omitted in the 3D convolutional mask. I already experimented with the CNN with some positive results. However, it did not provide better results in comparison to my convolutional mask.

Another idea for achieving more precise surface reconstruction is to employ the global optimization inspired by articles [4] or [7] to smoothen the reconstructed surface.

Bibliography

[1] "Telecentric lens description, Edmund Optics Inc.." https://www.edmundoptics.co.uk/resources/application-notes/imaging/advantages-of-telecentricity/. Online; Accessed: 2019-05-22.

[2] S. Pertuz, D. Puig, and M. A. Garcia, "Analysis of focus measure operators for shape-from-focus," *Pattern Recognition*, vol. 46, no. 5, pp. 1415 – 1432, 2013.

[3] S. K. Nayar and Y. Nakagawa, "Shape from focus," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, pp. 824–831, Aug. 1994.

[4] M. Moeller, M. Benning, C. Schönlieb, and D. Cremers, "Variational depth from focus reconstruction," *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 5369–5378, 2015.

[5] P. Favaro and S. Soatto, "A geometric approach to shape from defocus," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 3, pp. 406–417, 2005.

[6] S. Suwajanakorn, C. Hernandez, and S. M. Seitz, "Depth from focus with your mobile phone," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3497–3506, June 2015.

[7] H. Tang, S. Cohen, B. Price, S. Schiller, and K. N. Kutulakos, "Depth from defocus in the wild," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4773–4781, July 2017.

[8] "Nikon Focal Stacking D850, Nikon Corporation." https://nps.nikonimaging.com/technical_solutions/d850_tips/useful/focus_stacking/. Online; Accessed: 2019-05-22.

[9] "Adobe CC, Adobe Inc.." https://www.adobe.com/cz/creativecloud.html. Online; Accessed: 2019-05-22.

[10] "Helicon Focus, Helicon Soft Limited." https://nps.nikonimaging.com/technical_solutions/d850_tips/useful/focus_stacking/. Online; Accessed: 2019-05-22.

[11] M. Šonka, V. Hlaváč, and R. Boyle, *Image Processing, Analysis and Machine Vision*. Toronto, Canada: Cengage Learning, 4 ed., 2015. 3rd edition 2008, Chinese translation 2003, 2nd edition 1998, 1st edition 1993.

[12] B. S. Lipkin and A. Rosenfeld, eds., *Picture Processing and Psychopictorics*. Orlando, FL, USA: Academic Press, Inc., 1970.

[13] N. Kanopoulos, N. Vasanthavada, and R. L. Baker, "Design of an image edge detection filter using the sobel operator," *IEEE Journal of Solid-State Circuits*, vol. 23, pp. 358–367, April 1988.

[14] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, pp. 679–698, Nov 1986.

[15] D. Marr and E. Hildreth, "Theory of edge detection," *Proceedings of the Royal Society of London Series B*, vol. 207, pp. 187–217, 1980.

[16] G. Sotak and K. Boyer, "The laplacian-of-gaussian kernel: A formal analysis and design procedure for fast, accurate convolution and full-frame output," *Computer Vision, Graphics, and Image Processing*, vol. 48, no. 2, pp. 147 – 189, 1989.

[17] S. Kraus, "Měření rozměru cCD kamerou se submikronovou přesností," Master's thesis, České vysoké učen´ i technické v Praze, Fakulta elektrotechnická, Katedra řídící techniky, Centrum strojového vnímání, Karlovo nám. 13, 121 35 Praha 2, Česká republika, 1999. In Czech.

[18] R. M. Haralick and L. Watson, "A facet model for image data," *Computer Graphics and Image Processing*, vol. 15, no. 2, pp. 113–129, 1981.

[19] M. Born, E. Wolf, A. B. Bhatia, P. C. Clemmow, D. Gabor, A. R. Stokes, A. M. Taylor, P. A. Wayman, and W. L. Wilcock, *Principles of Optics: Electromagnetic Theory of Propagation, Interference and Diffraction of Light*. Cambridge University Press, 7 ed., 1999.

[20] D. K. Samuylov, P. Purwar, G. Székely, and G. Paul, "Modelling point spread function in fluorescence microscopy with a sparse gaussian mixture: trade-off between accuracy and efficiency," *IEEE Transactions on Image Processing*, pp. 1–1, 2019.

[21] K. Ahi, "Mathematical modeling of thz point spread function and simulation of thz imaging systems," *IEEE Transactions on Terahertz Science and Technology*, vol. PP, pp. 1–8, 10 2017.

[22] B. Delaunay, "Sur la sphère vide. a la mémoire de georges voronoï," *Bulletin de l'Académie des Sciences de l'URSS. Classe des sciences mathématiques et na*, vol. PP, pp. 793–800, 1934.

[23] "JAI GO-5000M-PGE, JAI Inc.." `https://www.jai.com/products/go-5000m-pge`. Online; Accessed: 2019-05-22.

[24] E. Jones, T. Oliphant, P. Peterson, *et al.*, "SciPy: Open source scientific tools for Python," 2001–. [Online; Accessed: 2019-05-22].

[25] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011. Online; Accessed: 2019-05-22.

[26] Itseez, "Open source computer vision library." `https://github.com/itseez/opencv`, 2015. Online; Accessed: 2019-05-22.

[27] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing In Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.

[28] "eBUS SDK, Pleora Techologies Inc.." `https://www.pleora.com/products/ebus-sdk`. Online; Accessed: 2019-05-22.

[29] F. Akgul, "Zeromq." `http://zeromq.org/`, 2013. Online; Accessed: 2019-05-22.

[30] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia, "MeshLab: an Open-Source Mesh Processing Tool," in *Eurographics Italian Chapter Conference* (V. Scarano, R. D. Chiara, and U. Erra, eds.), The Eurographics Association, 2008.

[31] "Multi-Frequency Grid Distortion Targets, Edmund Optics Inc.." `https://www.edmundoptics.com/f/multi-frequency-grid-distortion-targets/12312/`. Online; Accessed: 2019-05-22.

[32] Y. X. Qiang and Q. Ji, "A new efficient ellipse detection method," in *in International Conference on Pattern Recognition 2002*, p. 20957, 2002.

Appendix A

Structure of CD

The CD contains the thesis and 3D models of both scenes from Section 5.2. The thesis in PDF is in a root directory of the CD. The models are saved as Polygon File Format (PLY). The open-source mesh processing tool MeshLab [30] could be used to open the PLY files. The 3D models are located in directory **3D_models**. The first scene model is called **scene_1_flat_surface.ply**. The second scene model is called **scene_2_components_surface.ply**.

```
├─ 3D_models
│  ├─ scene_1_flat_surface.ply
│  └─ scene_2_components_surface.ply
└─ thesis.pdf
```