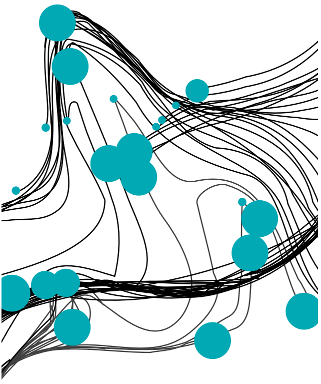
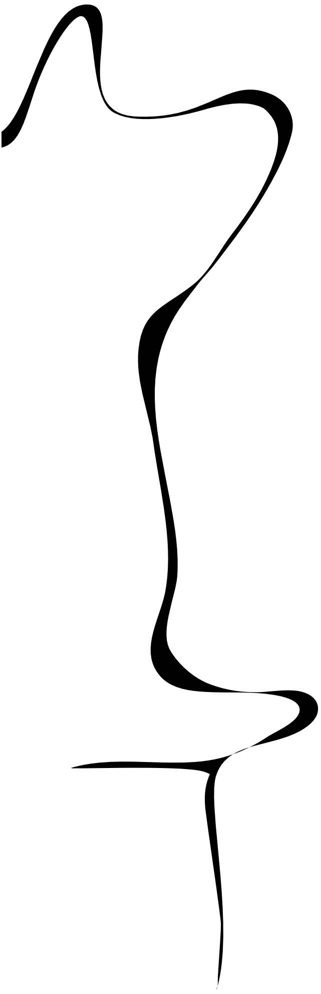


USING COMPUTER VISION FOR MONITORING THE PEELING PROCESS OF SEMICONDUCTING INSULATION SHIELD OF MEDIUM VOLTAGE CABLES



R. (Ruud) van den Berg

MSC ASSIGNMENT

Committee:
T.R. Elderhorst, MSc
dr. ir. E. Dertien
ir. L. Nies
dr. ir. C. Salm

May, 2023

013berg2023
Robotics and Mechatronics
EEMCS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Abstract

Alliander is a company that manages power grid connections to about 3.3 million clients. Switching from fossil fuels to renewable energy puts an additional load on the power grid. Therefore, Alliander must expand the capacity of the power grid. Expanding the power grid entails the construction of power connections over long distances, which are realized by splicing voltage cables. Splicing medium voltage cables is labor-intensive, so more qualified technicians are needed to improve and expand the power grid. Unfortunately, technicians are hard to come by due to the tension in the job market.

Before splicing two medium-voltage cables, a technician must prepare the cable ends. Part of this preparation process is peeling the insulation shield from the insulator. The Cable Splicing Robot project at the University of Twente aims to automate part of the splicing process. This cable splicing robot project involves a control system. This control system needs a monitoring system to measure the peeling process. This thesis project focuses on developing this monitoring system.

Peeling the insulator shield off the cable results in a chip. This chip resembles information about the peeling process. So, the monitoring system observes the chip and extracts information about the peeling process using computer vision. This information is suitable as a measured value for the cable splicing robot.

Validation results show that the monitoring system deviates less than 4 % from readings manually measured with a caliper. Technicians monitor the chip by visual inspection, which is much less accurate than using a caliper for measuring the chip. The monitoring system thus proves to be valid concerning accuracy. The monitoring system must also deliver measurements fast enough for the control system to function well. The sample time is 105 milliseconds. The monitoring system also proves valid concerning sample time.

Contents

1 Introduction

The Dutch electric power grid can be divided into two parts. One part is the high-voltage power grid for transporting power across the Netherlands. The other part consists of several medium-voltage to low-voltage grids to distribute the power regionally. The grid managers are responsible for managing these power distribution grids. Alliander[1] is one of these grid managers and manages power grid connections to about 3.3 million clients[2]. Over 40,000 km[2] of this power grid consists of medium voltage cables. Switching from fossil to renewable energy will increase electric power production and usage. Distributing this additional power puts an additional load on the grid. Therefore, the capacity of the power grid must be largely extended.

Splicing medium-voltage cables requires the use of cable joints. Installing these cable joints is labor intensive. The consequence is that Alliander struggles to connect new clients to the grid and process requests for increasing the power rating of existing grid connections. More qualified technicians are needed to speed up improving the power grid. Unfortunately, technicians are hard to come by due to the high tension in the job market[22]. As a result, extending the power grid its capacity is impeded.

This thesis project focuses on splicing medium-voltage cables. Medium-voltage cables usually have one or three conductors. Such a cable consists of an aluminum conductor surrounded by insulation material. A semiconducting layer encapsulates the insulation material to distribute the electric field homogeneously. A copper wire mesh protects the cable from mechanical stress. Finally, an outer cable jacket protects the cable against moisture.

By splicing two cables, a technician must first prepare the cable ends. He removes the outer cable jacket and folds the copper wire mesh backward to expose the semiconducting layer. He then removes the semiconducting layer by peeling it away with a peeling tool. This peeling tool has a blade whose cutting depth can be adjusted. The technician rotates this peeling tool around the cable to cut the semiconducting layer from the insulator. Finally, he removes part of the insulation material to expose the conductor. After the technician has prepared the cable ends, he connects the conductors of the two cables using a metal cable bushing. Then he adds a heat shrink tube, which is designed to withstand high voltages, over the cable bushing. He then adds a wire mesh to protect the cable joint from mechanical stress. Finally, he adds a heat shrink tube to protect the cable joint against moisture.

One difficult task when preparing cable ends is removing the semiconducting layer, or insulation shield, from the insulator. The Cable Splicing Robot project[5] at the University of Twente aims to automate this task. This way, the need for new technicians is reduced while speeding up extending the power grid its capacity. That project involves a control system to control the peeling process. This control system needs a monitoring system to measure the peeling process. An important parameter for a proper peeling process is the depth setting of the blade. The blade must be set deep enough to completely remove the semiconducting material while removing as little insulating material as possible.

1.1 Goal

Splicing two cables involves a number of processing steps. Each of them has its own quality aspect. The peeling process is the primary task of the robot project Univeraity of Twente. Therefore, this thesis project focuses on monitoring this part of the splicing. Currently, a technician monitors the blade depth by visual inspecting the chip that is peeled off. Therefore, the idea arose to use computer vision for the monitoring system for measuring the peeling process. So the goal of this thesis project is to arrive at a monitoring system based on computer vision. A solution can be found by answering the following questions.

> *"Which quality aspects in the peeling process can be inspected by means of computer vision, and which principles for computer vision are needed for this?"*

This general research question can be narrowed down to more specific sub-questions.

1. *"How to quantify the quality of the peeling process?"*
2. *"What is a suitable camera and illumination setup?"*
3. *"Which computer vision principles are suitable for measuring the quality aspects?"*
4. *"What are suitable performance indicators for the computer vision system?"*
5. *"What is a suitable test setup to measure the performance of the system?"*

1.2 Outline

In Chapter 2 the current workflow of Alliander to prepare medium-voltage cables is discussed and the peeling tool used by Alliander is shown. Alliander its current workflow is narrowed down to the process of peeling the semiconducting insulation shield, what this thesis project is about. Chapter 3 briefly reviews some computer vision methods aiming at visual quality inspection tasks. Chapter 4 describes the research methodology and the corresponding theory. The integration of the computer vision system into a prototype is described in detail in Chapter 5. Chapter 6 presents results of tests that are conducted with the computer vision methods discussed in Chapter 3. These results are subjectively assessed and the most promising computer vision method is selected for the monitoring system. In Chapter 7, final implementation of the monitoring system is objectively assessed and validated. This section also presents the validation results. Also, some limitations and recommendations are discussed. Chapter 8 finalizes the report with a conclusion.

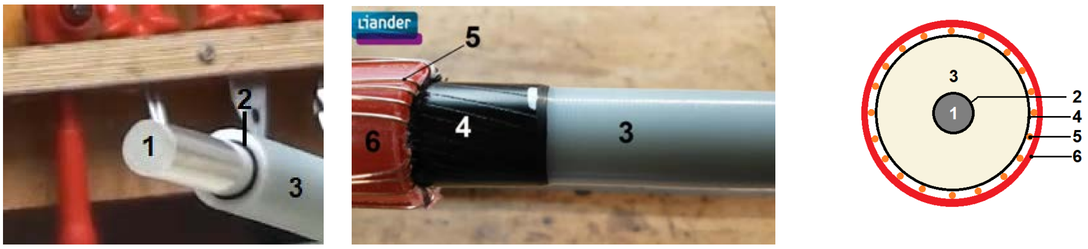
2 Problem analysis

In this chapter, the construction of a medium-voltage cable is described. The current workflow at Alliander for splicing two cables is explained. During the process of splicing the two cables, important quality aspects must be taken into account. One of these quality aspects is what the monitoring system is designed for.

2.1 The construction of a medium-voltage cable

The construction of medium-voltage cables (source: Alliander and [20]) is shown in Figure 1 with the numbered parts listed below.

1. aluminum conductor (metallic colored)
2. conductor shield, made of semiconducting material (black colored)
3. insulation (white colored)
4. insulation shield, made of semiconducting material (black colored)
5. copper wire mesh
6. PVC outer cable jacket (red colored)



(a) *front view with conductor*

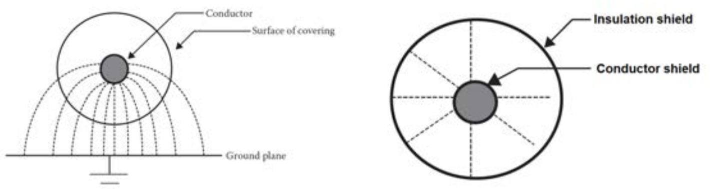
(b) *peeled section*

(c) *cross section*

Figure 1: *Two pictures of prepared cable ends (source: Alliander) and an illustration of a cross section showing the construction of an XLPE cable.*

The insulation material is made of cross-linked polyethylene[20], or XLPE for short, and must withstand high electric potential. Without precautions, this potential can result in the electric field concentrating on one region in the insulator, as shown by the field lines going from the conductor to the ground in Figure 2a. The field lines are concentrated on one region in the insulation material, which is not problematic for low voltages[20]. However, with medium-voltage cables, the electric field is preferably distributed evenly over the insulation material, which reduces the risk of the insulation material breaking[20]. This is achieved by applying a semiconducting layer, or shield, to the insulation material, 4 in Figure 1b. As a result, the electric field is distributed more evenly over the insulation material, as shown in Figure 2b.

Extending a cable using a cable connector impacts the distribution of the electric field. Therefore, it is very important that cable connectors are installed properly. So, technicians cannot rush and have to take their time for installing these cable connectors.



(a) *Concentrated on part of the insulation* (b) *Evenly distributed across the insulation*

Figure 2: *How electric field lines are distributed over the insulation material.[20]*

2.2 Current workflow for installing cable connectors

Currently, technicians splice cables entirely by hand by following the steps below (source: Alliander):

- Prepare the cable ends.
- Connect both cables using metal cable connectors.
- Install the insulation and shields on the metal cable connector.

"Improperly prepared cable ends provide inherent initiation of failures", according to William Thue[20]. As a result, many cable joints fail. To clarify how a cable end is prepared, this step is narrowed down to smaller steps, summarized below (source: Alliander):

- Remove the outer cable jacket.
- Fold away the copper ground wire mesh.
- Smooth the semiconducting layer by heating it with a blowtorch.
- Peel away te insulation shield from the insulator.
- Remove part of insulation material to make room for the metal cable connector.

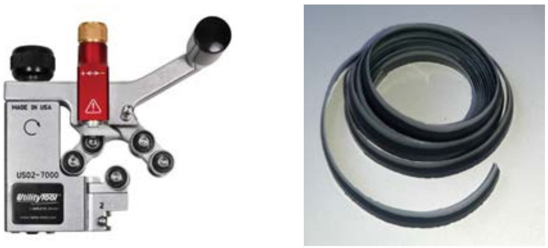
The cable splicing robot project[5] focuses on automating the process of peeling away the insulation shield.

2.3 Quality aspects

By preparing a cable end, some quality aspects must be met. These quality aspects were established by inquiry with Alliander, and are discussed below. First, the technician assures that the tools and cable are clean and that the cable is not bent. Then, he smooths the semiconducting layer by heating it with a blowtorch. This removes the pattern, left behind in the semiconducting material by the wire mesh. When the cable is clean, straight, and smooth, the technician mounts the peeling tool on the cable and carefully starts peeling it by rotating the tool around the cable. Figure 3 shows the peeling tool and a piece of chip that is peeled off. The semiconducting layer must be fully removed while leaving the insulator thick enough to withstand the high electric potential. The amount of material being peeled off can be set by adjusting the depth of the blade of the peeling tool. This depth setting influences the appearance of the chip. So, the chip is a good indicator for the blade depth and thus, the amount of material being peeled off(source: Alliander). Therefore, the technician monitors the peeling process by visually inspecting the chip. Finally, he fluently cuts the chip loose from the cable such that the transition from insulation material to semiconducting material is smooth, as shown in figure 4a. The quality aspects discussed are summarized in Table 1.

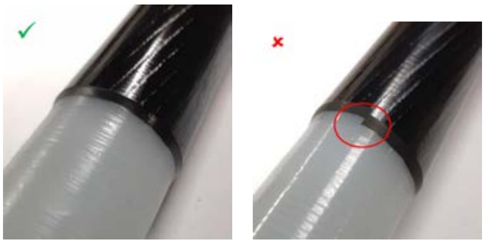
Table 1: *The quality aspects to be met during cable preparation.*(source: Alliander)

aspect	description
1	the peeling tool is clean.
2	the cable is straight before peeling.
3	the insulation shield is smoothed by heating.
4	no semiconducting material is left on the insulation during peeling.
5	no thin sections and cavities arise during peeling.
6	the transition from insulation to insulation shield is smooth as shown in Figure 4a.



(a) *Ripley-tools US2-series* (b) *Piece of chip.*

Figure 3: *The peeling tool and a piece of chip that is peeled off of the cable.*



(a) *Fluent transition.* (b) *With sharp corner.*

Figure 4: *Two examples of a cable from which the chip is removed.*(source: Alliander)

2.4 Peeling quality aspect

As described earlier, the peeling quality is monitored during peeling py visually inspecting the chip. In a successful peeling process, the chip consists of two parts, namely 'semiconductor' and 'insulator'. The relative width of the semiconductor is considered to be a good indicator for the quality of the peeling process. Denote the width of the semiconductor and insulator as W_{semi} and W_{insu} respectively, then the ratio is defined by Equation 1.

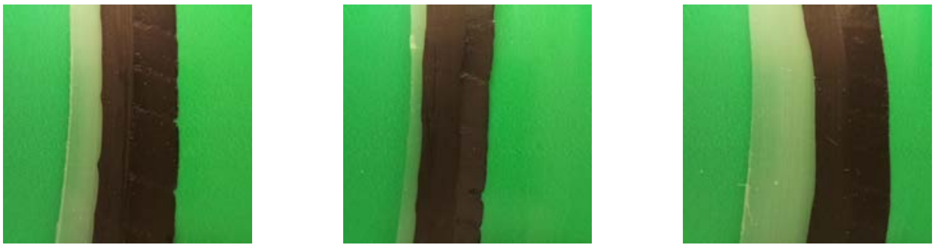
$$ratio = \frac{W_{semi}}{W_{semi} + W_{insu}} \tag{1}$$

As a norm, this ratio should be 2/3. Therefore, a good quantitative measure for the quality of the peeling process can be defined by a quality factor Q, defined in Equation 2.

$$Q = \left| \frac{W_{semi}}{W_{semi} + W_{insu}} - \frac{2}{3} \right| \tag{2}$$

This quality factor should be kept as close to zero as possible for a high quality peeling process.

Figure 5 shows three example chips. The chip should have a ratio of 2/3 semiconducting material, as shown in Figure 5a. When the blade is set too shallow, semiconducting material may remain on the insulation material. This results in a mostly black chip, as shown in Figure 5b. When the blade is set too deep, too much insulation material gets removed. This results in much white on the chip, as shown in Figure 5c. The thickness of the semiconducting layer is not exactly the same everywhere. This results in a periodic pattern on the ratio of the chip. Therefore, the chip is allowed to have a margin.



(a) *Proper blade depth* $Q \approx 0$ (b) *Too shallow* $Q > 0$ (c) *Too deep* $Q > 0$

Figure 5: *Three chips that result from different blade depth settings. Colors are associated with: semi-conductor (black), insulator (semi transparent white), and background (green)*

3 Monitoring system: computer vision for quality inspection

In Chapter 2, a quantitative measure for the quality of the peeling process has been established. The goal of the monitoring system is to obtain this measure in a computer vision system. The physical scene consists of three objects, as Figure 5 shows. Namely ”background”, ”insulation” and ”semiconductor”. These three objects must be distinguished using computer vision methods. Then, the ratio can be measured. This ratio can be used for defect detection.

3.1 Visual quality inspection systems

In a visual quality inspection system, a scene is illuminated with a light source and observed with a camera. The images captured by the camera can be conditioned such that subsequent image processing tools can work better. Then, the conditioned image can be segmented, in which the image is partitioned into segments that correspond to the objects in the scene. Features of these segments can be extracted, resulting in descriptive data. A defect detector can use this data to detect defects in the scene. The architecture of the monitoring system described above is illustrated in Figure 6.

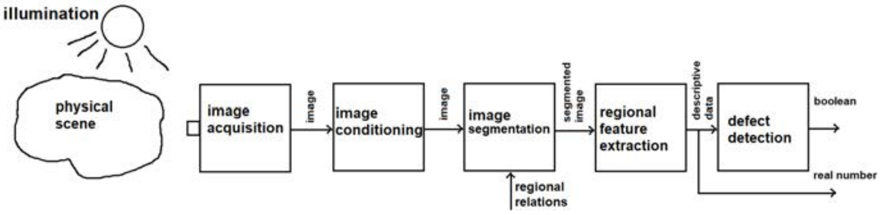


Figure 6: *Fundamental architecture of a visual monitoring system.*

3.2 Inspection of the peeling process

The Visual quality inspection systems must be integrated into the existing peeling tool. A schematic illustration of how the monitoring system can be realized is shown in Figure 7.

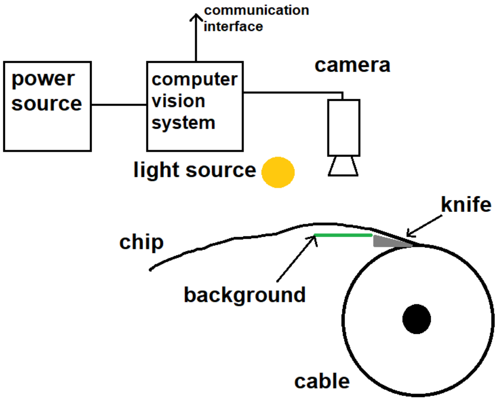


Figure 7: *Setup of the monitoring system.*

3.3 Camera and illumination setup

The physical scene must be illuminated for a camera to observe it. There could be ambient light, but this is not a light source to rely on. Therefore, it is best to have an artificial light source to illuminate the scene. A suitable camera-illumination setup must thus shield off ambient light and illuminate the scene in a controlled way. The semiconductor and insulator might have nearly overlapping grayscale values which makes it difficult to distinguish them in a grayscale image. For this reason, it is advisable to use color images. A design aspect is the background color to use. This color must be contrasting with the colors of the semiconductor and the insulator. Among other colors, green is appropriate.

Another design aspect is the focal distance of the camera lens. The monitoring system must be small, since it will be integrated on a peeling robot in the main project. Therefore, the camera must be able to focus closely.

The chip constantly moves in front of the camera, which may result in motion blur. To limit motion blur, the exposure time must be very short. There are three ways to achieve this. One way is by increasing the sensitivity of the sensor. However, the susceptibility to image noise also increases here. The second way to achieve this is by using a lens with a high diameter aperture, which transfers much light from the scene to the sensor. Unfortunately, this in combination with focusing closely, results in the depth of field being very narrow. The third way to achieve short exposure times is by using a very bright light source. Then, the image sensor its sensitivity can be kept low and the lens its aperture can be kept small, increasing the depth of field.

Figure 8 illustrates how the camera and light source can be positioned to properly capture images from the scene. The light source is positioned such that no direct light enters the lens of the camera.

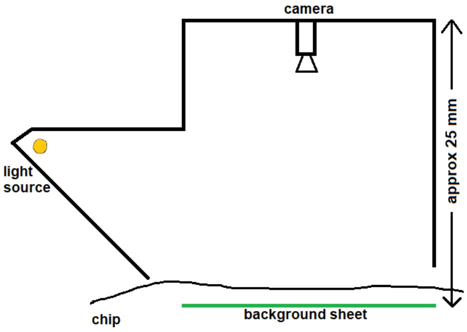


Figure 8: *Drawing of camera-illumination support structure.*

3.4 Image conditioning

After image acquisition, images may not yet be optimal for further processing. Sub-optimal images may be optimized using image conditioning methods. A selection of image conditioning methods is listed below.

- Contrast enhancement[6]
- Histogram equalization[6][12]
- Color adjustment[23]
- Noise reduction and sharpening[23][12]

These image conditioning methods can help make computer vision tools perform at best. Contrast enhancement is expected to be advantageous as it can be used to normalize the rgb values between 0 and 1, so that the images become less sensitive to changes in illumination and diffuse reflection coefficients of the material. Therefore, contrast enhancement is expected to work satisfactorily.

3.5 Image segmentation

After the image is conditioned, the objects in the image must be partitioned into segments that correspond to the objects in the scene by extracting specific features that are distinctive for the objects. There are many image segmentation methods and a few methods that can contribute to this project are listed below.

- Edge detection[12]
- Segmentation by multi-thresholding[12]
- Segmentation by pixel classification based on probabilistic model (Bayesian classification[18])
- Deep learning based semantic segmentation[14] (Artificial Neural Network)

The edges, detected by an edge detector, could be used to divide the image into segments.

3.5.1 Edge detection

Objects can be separated based on the sharp change in brightness on the edges[12] of those objects. These edges can be identified by an edge detector. One of the more popular edge detectors is the Canny edge detector[12], of which several free implementations are available. The UT version is based on Gaussean-based image derivatives. Edge detection is expected to be insensitive to unevenly illuminated scenes. However, scratches on the chip are expected to show up as falsely detected edges. Note that edge detection does not exploit the color information in RGB images as they operate on grayscale images. OpenCV and Matlab use the transformation defined in Equation (3) to map color images (RGB) to grayscale images (Y).

$$Y = 0.299R + 0.587G + 0.114B \tag{3}$$

Mappings with other coefficients may improve edge detection quality a bit. For example, when the background is green, a good mapping could be $Y = 1G$ then. In other words, pick the green channel and use it as a grayscale image. Unfortunately, it turned out that edge detection works insufficiently, as shown in section 6.1. No mapping has been found for the edge detector to perform sufficiently.

3.5.2 Segmentation by multi-thresholding

Images can be segmented directly using multi-thresholding. This method is based on classifying pixels based on their grayscale value. Pixels are assigned to a particular class if their grayscale value falls within the range specified for that class. This method is sensitive to unevenly illuminated objects because pixels are classified incorrectly if their grayscale value falls within the range specified for a different class. So, underexposed and overexposed areas may be segmented incorrectly. Multi-thresholding usually operates on grayscale images, but can also be implemented in color space by treating the color channels separately. Multi-thresholding in color space could have been tested in this project, but it is a small step towards Bayesian classification. Therefore, no effort is spent on multi-thresholding in color space.

3.5.3 Segmentation by pixel classification based on probabilistic model

In Bayesian pixel classification[12, 14, 18], pixels are considered to belong to classes. Each class corresponds to an object in the scene. A pixel is assigned to the class with the maximum probability, based on the observed RGB values of that pixel. Usually, Gaussian distributions are assumed for the probability densities[12, 14, 18]. Therefore, the segmentation is based on the following class-dependent parameters:

- The prior probability of a class
- The expected RGB values of a class
- The covariances between the RGB channels of a class.

These parameters can be inferred from an initial image. The parameters resemble relations between the color channels of the pixels. Since the parameters depend heavily on the illumination of the scene, the illumination conditions must be constant.

Pixel classification using Bayes approach works as follows. Let the 3D vector $\vec{x}_{r,c}$ denote the RGB values of the pixel at position (r,c) in the image. And let the variable k denote the class. That is:

- $k = 1$: semiconductor
- $k = 2$: insulator
- $k = 3$: background

In the Bayes approach, classification of a pixel is done by selecting the most likely class $\hat{k}_{r,c}$ given the color of the pixel. Mathematically this is expressed by Equation 4.

$$\hat{k}_{r,c} = \underset{k=1,2,3}{\arg\max}\, P(k|\vec{x}_{r,c}) \tag{4}$$

$P(k|\vec{x}_{r,c})$ is the posterior probability of having a class k under the condition that the color of the pixel is $\vec{x}_{r,c}$. This probability can be calculated using Bayes rule for conditional probabilities, defined by Equation (5):

$$P(k|\vec{x}_{r,c}) = \frac{P(\vec{x}_{r,c}|k)P(k)}{P(\vec{x}_{r,c})} \tag{5}$$

$$P(\vec{x}_{r,c}) = \sum_{k=1}^{3} P(\vec{x}_{r,c}|k)P(k) \tag{6}$$

To use Bayes' formula, some parameters must first be inferred from a manually segmented initial image. Subsequent images can then be segmented automatically using the parameters inferred from that initial image.

The prior probabilities of the three classes $P(k)$ are calculated by taking the percentage of pixels belonging to each class with respect to the total number of pixels.

$$P(k) = \frac{N_k}{N} \tag{7}$$

of which

$$\begin{aligned} N_k &= \text{number of pixels belonging to class } k \\ N &= \text{total number of pixels} \end{aligned}$$

The multivariate normal probability density $P(\vec{x}_{r,c}|k)$ of a pixel $\vec{x}_{r,c}$, given its class, can be calculated using the mean $\vec{\mu}$ and covariance matrix C inferred from the manually segmented image[14].

$$P(\vec{x}_{r,c}|k) = \frac{1}{\sqrt{|C_k|(2\pi)^d}} exp\left(-\frac{1}{2}(\vec{x} - \vec{\mu}_k)C_k^{-1}(\vec{x}_{r,c} - \vec{\mu}_k)^T\right) \tag{8}$$

of which

$$\begin{aligned} d &= \text{Number of dimensions in } \mathbb{R}\text{. In this case } d = 3 \\ \vec{\mu}_k &= \text{Mean pixel color values in } \mathbb{R}^3 \text{ for class } k \\ C_k &= \text{Covariance matrix in } \mathbb{R}^{3\times3} \text{ for class } k \end{aligned}$$

The mean pixel color $\vec{\mu}_k$ can be obtained for each class k by calculating the arithmetic mean of all pixels at positions (r, c) in k, denoted by Equations (9)[14, 15].

$$\vec{\mu}_k = \frac{1}{N_k} \sum_{\forall r,c \in k} \vec{x}_{r,c} \qquad \vec{\mu}_m \in \mathbb{R}^3 \tag{9}$$

The variance of the color channels and the covariance between the color channels must also be calculated. Image pixels can be represented as a row of column vectors, or a matrix $X \in \mathbb{R}^{3xN}$. This way, the covariance matrices C_k can be calculated for each class k using Equation (10) [14, 15].

$$C_k = \frac{1}{N_k - 1}(X_k - \vec{\mu}_k)(X_k - \vec{\mu}_k)^T \qquad C_k \in \mathbb{R}^{3\times3} \tag{10}$$

Using Bayes' theorem, the probability that a pixel belongs to class k, can be calculated for all pixels using Bayes' formula (5). Classification is done for all pixels by picking the class with highest probability.

Appendix A clarifies Bayesian classification with an example.

3.5.4 Deep learning based semantic segmentation

Segmentation based on deep learning[21] is a promising state-of-the-art method that should not be left undiscussed. Convolutional Neural Networks are often used on images[14]. Training such a deep network requires many manually segmented images as training data. Although this method looks promising, it is time-consuming to generate enough training data. When something changes in the monitoring system, like illumination conditions or the camera, the neural network must be trained again with new training data. Creating this new training data again consumes much time. Also, the images in this thesis project can be so well-conditioned that deep learning is unneeded. Therefore, no effort is spent on deep learning based semantic segmentation.

3.6 Contextual information about the scene

Progressive insight revealed that the insulation material of the chip is a bit transparent. This results in insulation material, which is normally white, appearing a bit greenish in the image. As a result, the insulation material has features that are somewhat similar to the background. After an input image is segmented, some parts of insulation material and background are expected to be misclassified. Fortunately, the black semiconducting material will likely be segmented correctly. This knowledge can be used as contextual information to improve segmentation quality.

The contextual information includes two relations. First, there is no insulation material on the right-hand side of the semiconducting material. Second, a limited amount of insulation material is on the left-hand side, in close proximity of the semiconducting material. This contextual information describes regional relations that can serve in improving segmentation performance. The contextual image can be used to create new prior probability distributions $P_{r,c}(k)$ that depend on the location in the image. Bayesian classification can then be applied for a second iteration. A method for creating new prior probabilities is described below.

Input: A segmented image with segments *semi*, *insu*, and *bgnd*.
Output: Position dependent prior probabilities $P_{r,c}(semi)$, $P_{r,c}(insu)$, and $P_{r,c}(bgnd)$.

1. Determine the connected components of the segment *semi*.
2. From the components, determined in step 1, select the largest component. The result is stored in a binary image *bw_semi*.
3. Determine the pixel-dependent prior probability for class *semi*:

$$P_{r,c}(semi) = \begin{cases} 1 & \text{for all } (r,c) \in bw\_semi \\ 0 & \text{elsewhere} \end{cases}$$

4. Set the pixel-dependent prior probability for class *insu* to zero for all pixels: $P_{r,c}(insu) = 0$
5. For each row r in the image:
    5.1 Determine the width W_r of the semi-conductor zone as represented by the rows in *bw_semi*.
    5.2 Define an interval $[c_1, c_1 + W_r]$, positioned directly left from the semi-conductor zone, and with a width W_r.
    5.3 Define a function on this interval that smoothly increases from 0 to 1.
    $P_{r,c}(insu) = \frac{1}{2} - \frac{1}{2}\cos\left(\pi \frac{c-c_1}{W_r}\right)$ for $c \in [c_1, c_1 + W_r]$
6. Spatially smooth the prior probabilities $P_{r,c}(semi)$ and $P_{r,c}(insu)$ with a rectangular point spread function with a size of 1×15 pixels.
7. Set the prior probability for the background:
   $P_{r,c}(bgnd) = 1 - P_{r,c}(semi) - P_{r,c}(insu)$

Since the chip moves during image acquisition, new prior probability distributions must be generated for each new input image. An example of using contextual information is shown in Appendix B.

3.7 Regional description

After segmentation, the image area is divided into three regions: background, semiconductor, and insulator. The next step is to derive parameters from these regions. This is called regional description. The parameters that matter here are the widths of the semiconductor and the insulator regions. Assuming that the orientation of the camera is aligned such that the chip moves in the vertical direction of the image, the widths are easily obtained by counting the number of pixels that spans the semiconductor and the insulator regions for each row in the image. Let W_{semi} and W_{insu} denote these width calculated for each row. The ratio can be calculated by Equation (1). Then, the global ratio is calculated as the averages over the rows in the lower quarter of the image area.

3.8 Defect detector

In the context of this project, the system has only to decide whether or not the blade depth is set correctly based on the ratio of the chip. As discussed in the quality aspects in section 2, the ratio of the chip is allowed to deviate from a 2/3 ratio, or 67 % ratio, quite a bit. So, the error in ratio the chip is allowed to have is chosen to be 10 %-point. In other words, the peeling process will be considered defective when the quality factor Q, defined by Equation 2, exceeds 0.1.

3.9 Requirements

For the prototype to be usable, at least two requirements must be met. First, it must be an order of magnitude more accurate than the human eye. As stated earlier, a deviation of 10 % is acceptable. Second, it must process the input images quickly enough for the control system to function properly. A sample rate of about 10 Hz is considered adequate. In summary:

- Accuracy of 10 %
- Sample rate 100 ms ± 10 %.

4 Research methodology

Chapter 3 outlined general principles for visual inspection. Chapter 3 also defined the functional architecture suitable for the current application. It listed three segmentation methods that might work well for the current problem. In chapter 6, the results are subjectively assessed and the best segmentation method is used in the prototype.

4.1 prototype

A prototype, serving as a demonstrator, will be constructed by integrating the monitoring system on the existing peeling tool used at Alliander. This prototype is then capable of monitoring the blade depth setting during the peeling process. It will also be used to produce test data for the experiments and validation. Debugging the system involves peeling many cables, which results in lots of waste. Therefore, one cable is peeled with three blade depth settings; too deep, correct and too shallow. This results in three chips listed below.

- One with the blade set just right. Ratio about 67 %.
- One with the blade set too deep. Ratio much smaller than 67 %.
- One with the blade set too shallow. Ratio much higher than 67 %.

These chips can be used over and over again by manually pulling them through the prototype, preventing lots of waste.

4.2 Image acquisition for testing

For each chip, 200 images are captured over a 75 cm long section of the chip. These images are stored on a computer so that they can be reused. Reusing images is also beneficial for the experiments because the exact same input data is used for the three image segmentation methods. So, the outcome of the experiments is not affected by differences in the input data. This results in an image data set of 600 images. The camera its brightness must settle after it is turned on, so the first 20 images are expected to be unusable.

4.3 Preliminary experiments

In a preliminary experiment, the image segmentation methods are tested. The 200 images acquired earlier from the chip with a correct ratio, are processed. The results of this experiment are subjectively assessed and the most promising segmentation method will be selected to be integrated in the prototype. The prototype with the selected segmentation method will then be validated.

4.4 Validation of the prototype

After the best image segmentation method is selected, the monitoring system must be validated. The data extraction part is added to the computer vision system to automatically measure the ratio of the chips. The automatically measured ratios are stored in a log file. This results in three log files. One for each chip.

The three chips are also manually measured on 50 predefined locations using a caliper. The chip is also measured on the exact same section as measured with the caliper. The prototype is validated by comparing the automatically measured ratios and manually measured ratios. The ratios are plotted and subjectively assess to assure that no erroneous measurements ended up in the results. For a true validation, the differences in automated measurements and manual measurements must be assessed objectively. To validate the monitoring system the mean ratios, logged in the six log files are compared for the automatically acquired measurements and manually acquired measurements. Estimating the ratio based on visual inspection is not accurate at all. Therefore, an estimation error of 10 %-point is considered appropriate.

5 Implementation of the prototype

A prototype will be developed that is able to monitor the blade height setting based on the chip and show a binary measure of the peeling process, ”good” or ”bad”, via a signaling LED. The ratio must be communicated to the control system, developed in the Cable Splicing Robot project[5] at the University of Twente. So the prototype developed in this thesis project will not propel the tool nor control the blade height. These requirements are summarized below.
The project must

- result in a prototype that monitors the blade height setting based on the chip.
- show a binary measure of the process, ”good” or ”bad” via an LED.
- communicate the ratio of the chip to an external device.
- be able to work on a 5 volt power source.

An important note is that the cable is stationary and that the demonstrator must be revolved around the cable to peel the semiconducting layer.

5.1 Design choices

There is a ready made tool with which the technician can peel the cable. This tool can be improved with a monitoring system which checks the peeling process. This monitoring system can show that the peeling process goes wrong by means of an LED that show whether or not the blade height is set properly. In this section, the parts needed to build the monitoring system are discussed.

5.1.1 Platform

The images acquired by the camera must be processed on a platform. Some examples of a platform are summarized below. The pro’s and con’s are noted as well to clarify the choices.

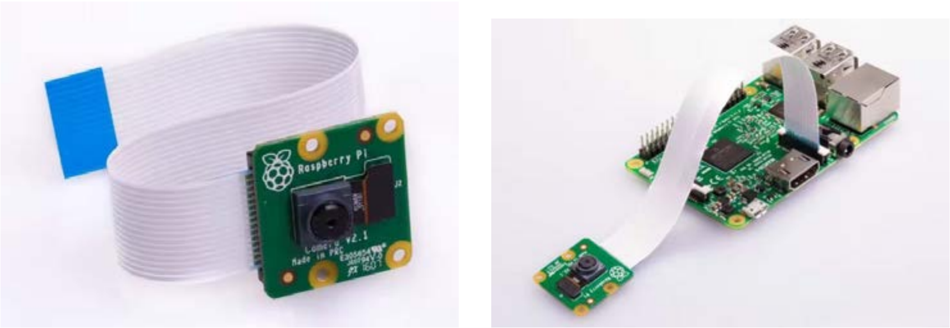
- FPGA/CPLD
  + Extremely fast
  − Difficult to write programs for
- Micro controller board
  + Cheap
  − Difficult to connect a camera to
  − Low processing power
- Single board computer
  + Adequate processing power
  + Easy to write software for

From these platforms, a single board computer comes out well. There are many single board computers that run Linux. The Raspberry Pi[10] is a well known single board computer that is suitable and it is chosen for this project. In addition, the Raspberry Pi foundation provides a standard operating system, Raspberry Pi OS[11] and it has a large community[9] of Raspberry Pi users.

5.1.2 Camera

For acquiring images from the chip, a camera that can focus closely is needed. The camera does not need to be very fast nor have high resolution. Some examples of cameras are summarized below as well as the pro’s and con’s.

- 1D photodiode array
  + cheap
  − Additional electronics must be developed
  − May miss parts of the chip
- Consumer grade camera, webcam
  + Cheap
  − No close focus
- Industrial camera
  + Close focus
  − Very expensive



(a) *Close up* (b) *Connected to Raspberry Pi*

Figure 9: Pictures of the Raspberry Pi camera*[8]*.

- UVC camera module with manual (close) focus
  + Close focus
  + Works on any platform with Linux/Windows or Mac OS installed
- Dedicated Raspberry Pi camera[8]
  + Close focus
  + Cheap
  + Small
  + Guarantee that it works with the Raspberry Pi
- 3D camera / depth image camera
  + Possibility to measure texture
  − Expensive
  − Much more work to write algorithms for

The Raspberry Pi camera module comes out best. This camera module is small, can focus closely, and can be connected to the Raspberry Pi. A picture of the Raspberry Pi camera is shown in figure 9.

5.1.3 Conditioned light

The computer vision algorithms are developed in lab conditions. When used outside the lab, lighting conditions may be different and the computer vision its robustness cannot be guaranteed anymore. To solve this, light must be conditioned. In other words, ambient light must be shielded off and artificial light must be created by a light bulb or an LED. Some illumination solutions are summarized below.

- Use an OLED display to illuminate the chip from the back
  + All sorts of background colors and patterns can be set
  − Difficult to implement
  − Takes valuable space between the chip
  − The insulating material may absorb too little light, making segmentation difficult.
- Use one LED as a point light source.
  + Easy to use.
  − May be unable to illuminate the scene evenly.
- LED bar, illuminating the scene from the side
  + The structure can be built such that light does not reflect into the camera directly
  + Easy to construct.
- Diffuse light source. Multiple LED's with diffusion sheet
  + Prevent white spots in the captured image due to reflections
  − Bit more difficult to implement

From the options above, the LED bar comes out best. In this project, a 1W LED filament is taken from a commercial household LED bulb. The color temperature is specified at 4000k, which corresponds to common daylight. Three 24V DCDC converters are used to generate the voltage needed to power the LED filament. To make it easier for the computer vision algorithms to distinguish the chip from the background, a green colored background sheet is used. A picture of the LED bar is shown in figure 10.

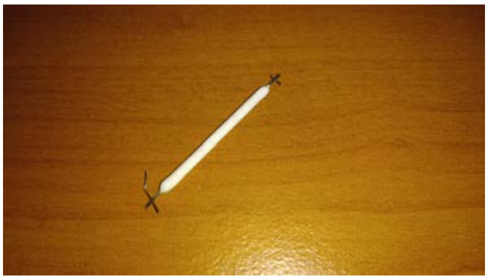


Figure 10: *Picture of a LED bar taken from a commercial LED filament bulb.*

5.1.4 Software

Designing the computer vision algorithms is done on a personal computer or laptop. The algorithms must then be ported to the Raspberry Pi. The programming language is thus preferred to be cross-platform. Below is a summary of some programming languages.

- Matlab[16]
  + Easy to use for prototyping
  − Does not run on a raspberry pi
- Python[7]
  + Relative easy to use
  + Cross-platform, scripts run on Windows as well as on the Raspberry pi
  + Installed on Raspberry Pi OS at standard[11]
- C, C++
  + Efficient
  − Takes much more time to write a program
- HTML, CSS
  + Suitable for writing a web app for communicating with the outside world
  − Not usable for image processing

The Python programming language[7] is chosen because it is cross-platform and easy to use. Python comes with the Integrated Development Evelopment Thonny[3]. Also, Python is installed on Raspberry Pi OS at standard[11] and it works with the OpenCV[13] library which has many computer vision tools. Matlab[16] is used, for conducting experiments, testing algorithms and visualizing results for the report.

5.1.5 Mechanics

To mount the equipment on the tool, additional mechanical parts must be made. Below is a summary of possible solutions for the mechanics.

- Custom parts milled in a metal workshop
  + Very robust
  + All the equipment is available
  − Labor intensive
  − Expensive
- 3D printed parts
  + Easy
  + Cheap
  + Complex geometries can be made
  + Printer does the work. Little manual labor.
  − Not very robust
- Manually engineer the parts
  − Labor intensive
  − Inaccurate

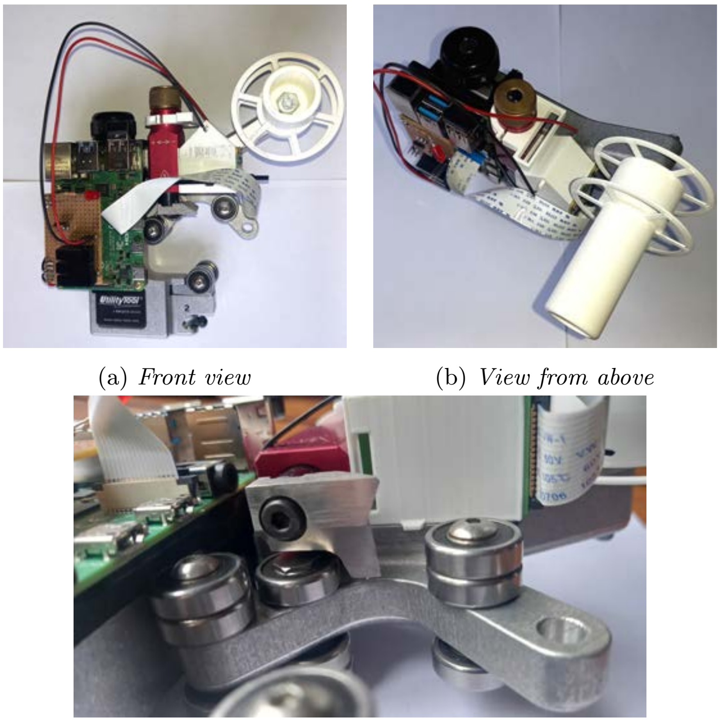
3D printed parts may not be as robust as metal machined parts, they are still robust enough and are well suitable for a prototype. Very complex geometries can be made that are impossible to machine on a milling machine or lathe. It also saves a lot of time to manufacture parts because the 3D printer does the work. Therefore, it is chosen to manufacture most parts using a 3D fused filament printer. The parts are designed in SolidWorks[19] since this software package is easy to work with and it is provided by the University of Twente.

5.1.6 Accessing the platform

The software must be installed on the Raspberry Pi. Accessing a Raspberry Pi from a Windows computer is often done using Putty[17]. This is a command line interface to communicate directly to the OS. A primitive graphical user interface can be set up using Xming[4]. However, the graphical user interface is slow and thus not suitable for streaming images. Fortunately, the Raspberry Pi also has an HDMI interface to connect a monitor to.

5.1.7 Summary of design choices

The system is built using a Raspberry Pi with a Raspberry Pi camera. The computer vision algorithms are written in Python with the OpenCV library. The system will eventually be powered from a battery used for the control system developed in the main project. However, during the development process, a 15 W Raspberry Pi USB-C power supply is used. Mechanical parts are manufactured using a 3D printer. Three photos of the prototype, with the monitoring system mounted on it, are shown in figure 11.



(a) *Front view*

(b) *View from above*

(c) *Close up of the blade*

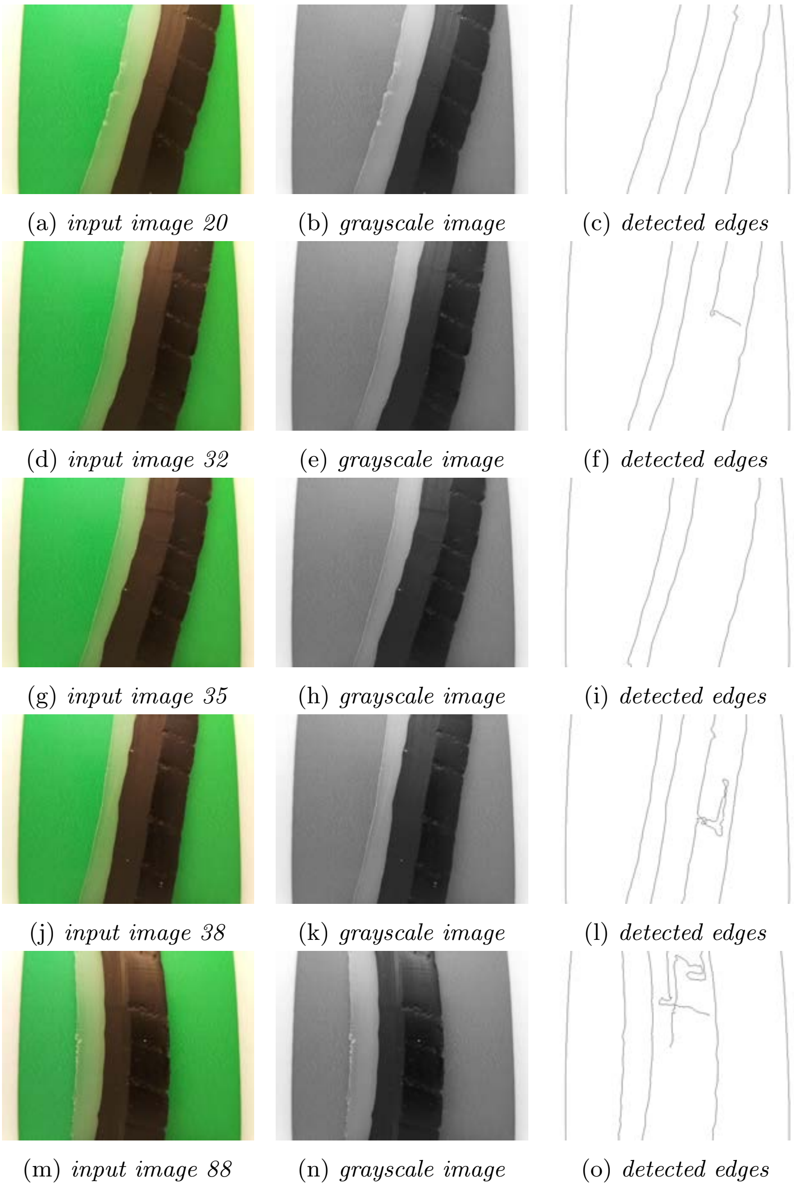
Figure 11: *Pictures of the monitoring system mounted on the tool.*

6 Results of the preliminary experiments

In this section, 180 of the images are processed with the three image feature extraction methods. Showing the results of all 180 images is cumbersome, so a selection of five results are shown and assessed.

6.1 Edge detection

Five input images are shown in the left column in figure 12. These images are converted to grayscale images prior to edge detection using the mapping noted by Equation (3), as shown in the center column. The right column show the detected edges. An important note is that part of the camera support structure is in sight of the camera, which results in two white regions in the input images.



(a) *input image 20* (b) *grayscale image* (c) *detected edges*

(d) *input image 32* (e) *grayscale image* (f) *detected edges*

(g) *input image 35* (h) *grayscale image* (i) *detected edges*

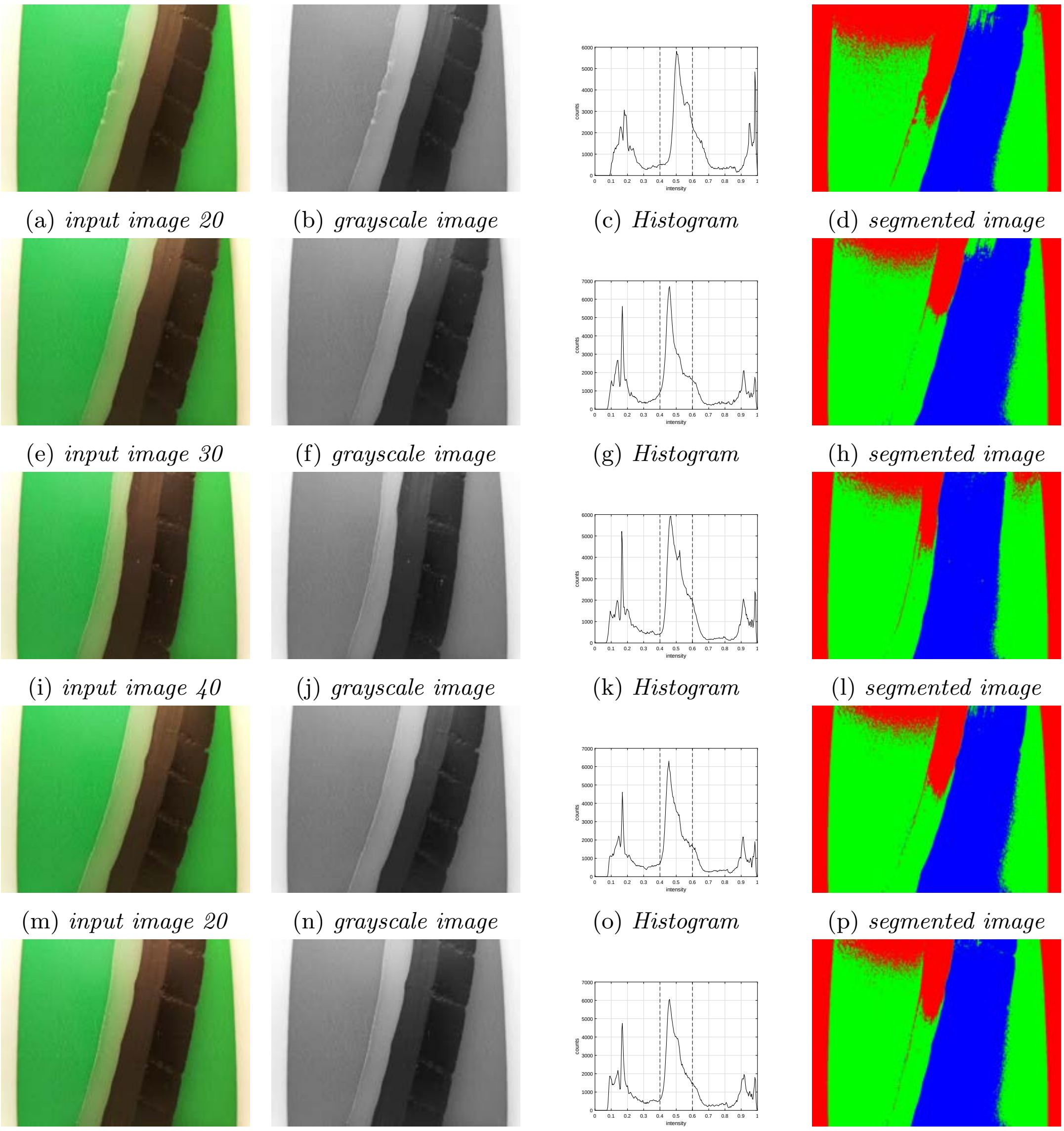
(j) *input image 38* (k) *grayscale image* (l) *detected edges*

(m) *input image 88* (n) *grayscale image* (o) *detected edges*

Figure 12: *Results of the edge detection algorithm applied on five images.*

6.2 Multi-Thresholding

Five selections of input images are shown in the leftmost column in figure 13. The images are converted to grayscale prior to segmentation, as shown in the left center column. The histograms of the grayscale images is shown in the right center column with the thresholds noted by two dashed lines. The results are shown in the rightmost column.



(a) *input image 20* (b) *grayscale image* (c) *Histogram* (d) *segmented image*

(e) *input image 30* (f) *grayscale image* (g) *Histogram* (h) *segmented image*

(i) *input image 40* (j) *grayscale image* (k) *Histogram* (l) *segmented image*

(m) *input image 20* (n) *grayscale image* (o) *Histogram* (p) *segmented image*

(q) *input image 60* (r) *grayscale image* (s) *Histogram* (t) *segmented image*

Figure 13: *Results of multi-thresholding applied on five images. The thresholds are indicated by the vertical lines in the histograms.*

6.3 Bayesian pixel classification

First, one image is manually segmented to get the parameters needed for applying Bayes' theorem. Second, these parameters are used to segment the rest of the images. The image shown in figure 14 is manually segmented.

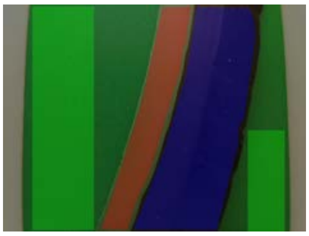


Figure 14: *the 25th image manually segmented.*

The white areas on the left and right in the image are caused by the walls of the camera support structure that come in sight of the camera. These areas can be trimmed off. Although not all pixels are segmented, the number of pixels will be enough to get the parameters needed. The pixels that are segmented are plotted in 3D space by using their color channels as Cartesian coordinates, as shown in Figure 15

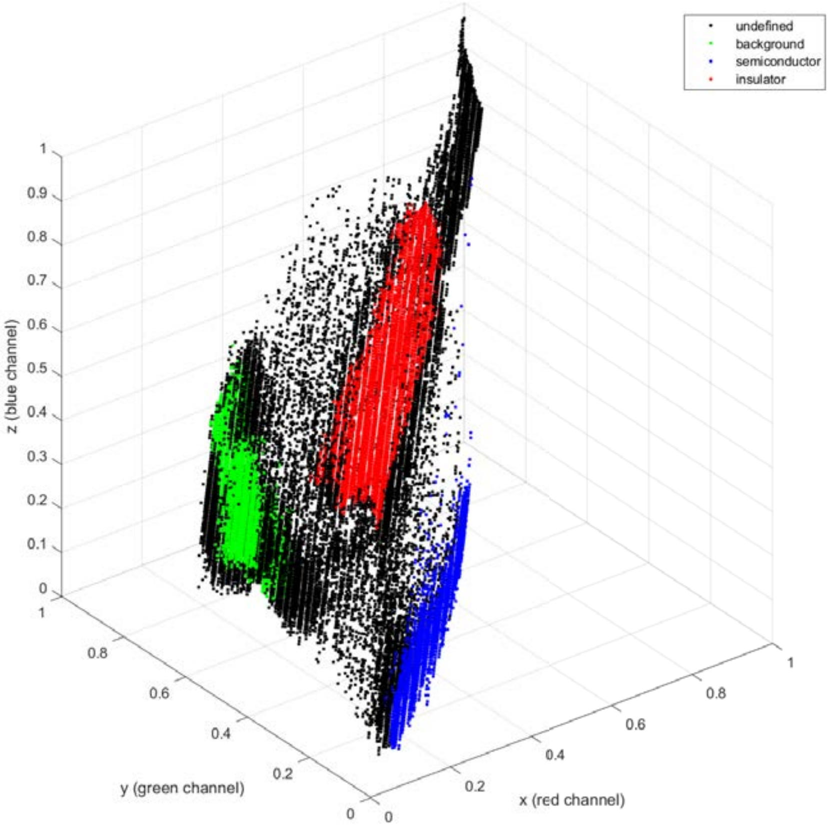


Figure 15: *All image pixels plotted 3D space based on their color channels. The black undefined dots are missed in manually classification.*

The pixels are separable by Bayesian classification since they are not mixed together. The mean pixel value $\vec{\mu}_m$ and covariance matrices C_m can now be calculated based on the manually classified pixels. These parameters $\vec{\mu}_m$ and C_m can best be visualized to clarify what they mean. See Figure 16.

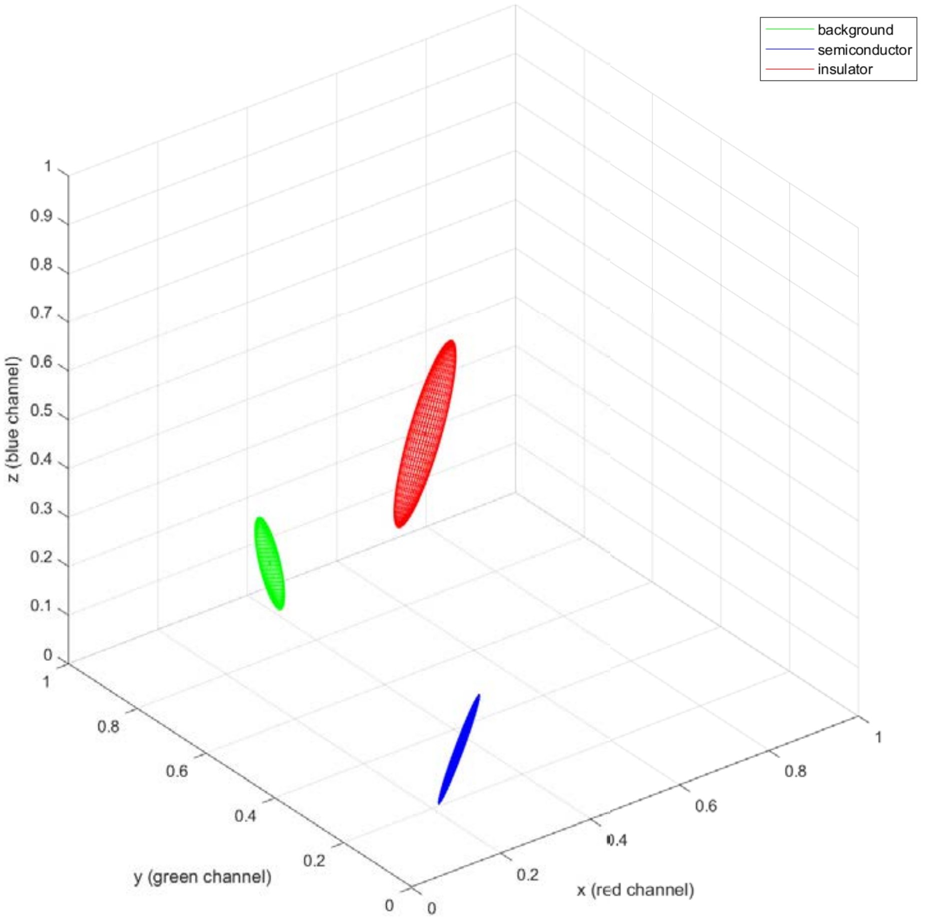
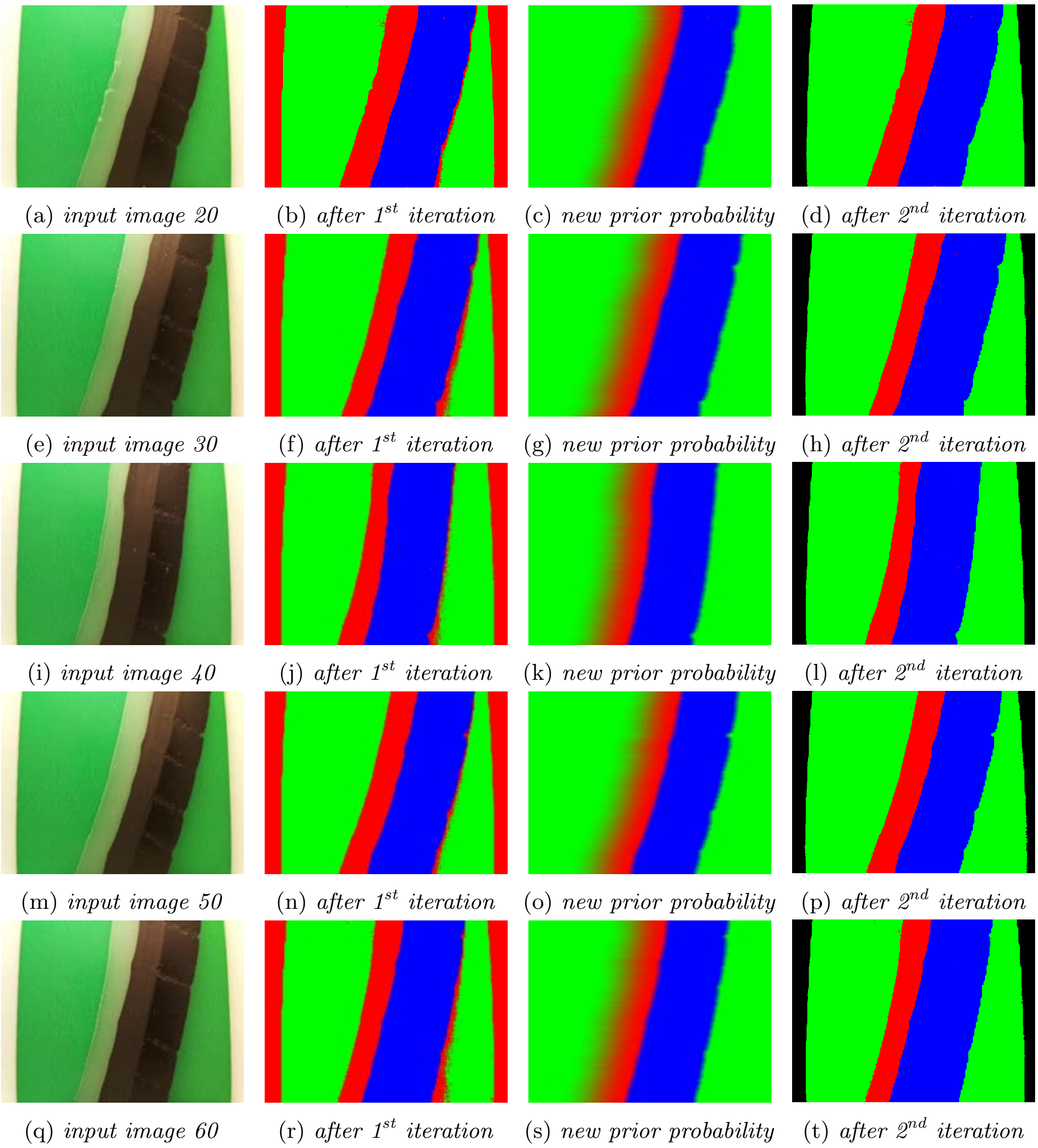


Figure 16: *The covariance matrices and mean pixel values visualized as ellipsoids.*

The covariance matrices C_m describe the exact shape of the ellipsoids. The mean pixel values $\vec{\mu}_m$ describe the location of the ellipsoids in space. Please note the similarities between Figures 15 and 16. The ellipsoids enclose about 31.7 % of the colored dots in Figure 15.

Figure 12 show a selections of five input images. The left center column show the results after the first segmentation iteration. These results are used to generate new prior probability distributions shown in the right center column. Then, the input images are segmented a second time - second iteration - resulting in the segmented images in the rightmost column.



(a) *input image 20* (b) *after 1st iteration* (c) *new prior probability* (d) *after 2nd iteration*

(e) *input image 30* (f) *after 1st iteration* (g) *new prior probability* (h) *after 2nd iteration*

(i) *input image 40* (j) *after 1st iteration* (k) *new prior probability* (l) *after 2nd iteration*

(m) *input image 50* (n) *after 1st iteration* (o) *new prior probability* (p) *after 2nd iteration*

(q) *input image 60* (r) *after 1st iteration* (s) *new prior probability* (t) *after 2nd iteration*

Figure 17: *Results of Bayesian classification applied on five images.*

6.4 Discussion

The results of the Edge detection method, shown in Figure 12, demonstrate that most true edges are detected correctly. Unfortunately, there are also edges that are spurious. It is difficult to distinguish the true edges from the spurious edges, making this method fragile.

The results from the multi-thresholding method are shown in figure 13. The histograms reveals three peaks, which create the impression that the three segments are separable. However, the colors of the insulation material and background are too similar to be distinguished by multi-thresholding entirely. So this method is also fragile.

In Bayesian classification, one initial image is manually segmented and the pixels, shown in Cartesian coordinates in Figure 15, are separable. After the first classification iteration, the images are segmented reasonable already. As a result, the new prior probability distribution functions are of high quality, resulting in that the segmentation results after the second iteration are even better. So, this method shows promising.

In short: The edge detector often detects spurious edges. Segmentation by multi-thresholding is not able to separate the objects in the image correctly. By using Bayesian classification, images are already reasonable segmented after the first iteration and even better after the second iteration. Therefore, Bayesian classification is used in the prototype. The prototype is validated in section 7.

7 Validation

The results in section 6 showed that Bayesian classification works best for segmenting the images. Therefore, this method is implemented in the prototype. The prototype is validated by measuring tree chips with the prototype and manually using a caliper. The ratios measured by the prototype and measured manually should show similar results. As noted in section 4, the technician estimates the ratio by visual inspection. Measuring it with a caliper is an order of magnitude more accurate than eyeballing. So, a difference lower than 10 %-point is considered valid.

Not all of the image-plane is used to measure the ratio. Otherwise, some section of the chip may be measured multiple times. Therefore, a small region of interest (ROI) of forty lines of pixels is used, as shown in Figure 18. This region of interest is used for measuring the ratio.

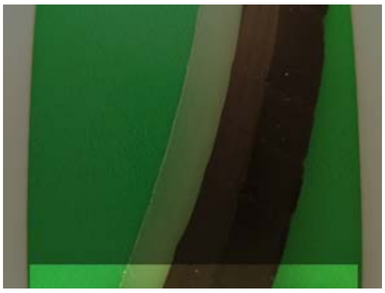


Figure 18: *The brighter part on the bottom shows the region of interest (ROI).*

The three chips, shown in Figure 19a, are marked with a white gel-ink pencil on 50 locations about 10 mm apart. The width of the chip is measured on the regions of interest, encircled in red in Figure 19b.

First, images are captured with the prototype. A small python script allows the user to positioning the chip and manually capture images on specific locations. The white marks on the chips are kept just out of view. This way, the monitoring system is forced to measure the ratio on the pre-specified locations marked with the gel-ink pencil. Figure 19c shows one captured image. Measuring all three chips on the 50 predefined locations results in 150 measurements.

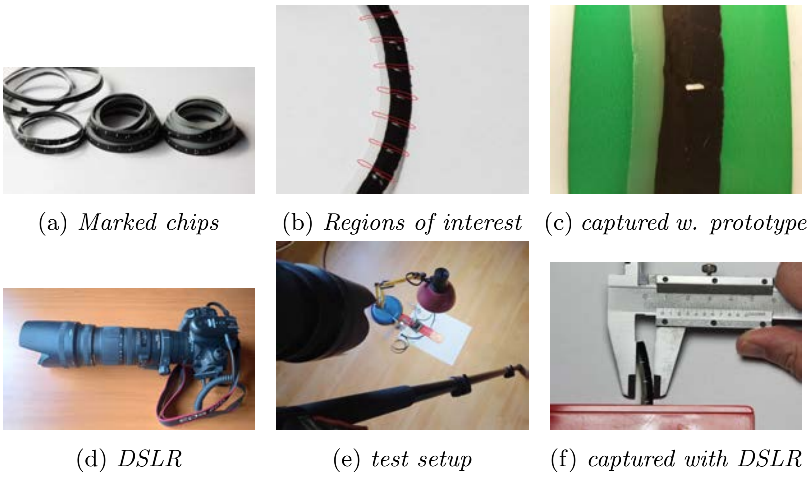
Then, the chips are measured manually with a caliper. The chip is secured such that the caliper can be positioned accurately. A photo is taken using a Canon 60D DSLR with a 70-200mm F2.8 lens and a 1.4x teleconverter, shown in Figure 19d. This way, the caliper can be read accurately from the photo and human mistakes can be detected and rectified afterwards. Figure 19e shows the setup for manually measuring the chips. Measuring the ratio of the chip involves two measurements. One is measuring the width of the semiconducting material. The second is measuring the total width of the chip. Capturing these measurements on all 50 locations of the three chips results in 300 photos. One of these 300 photos is shown in Figure 19f. These 300 photos result in 150 measurements.

Finally, these manually acquired measurements are superimposed on the automatically obtained measurements from the prototype. The prototype is valid with respect to ratio measurements if the ratios differ less than 10 %-point.

The monitoring system is part of a control system. So the performance with respect to sample time is also validated. Since it may happen that the chip moves through the monitoring system faster than anticipated, parts of the chip may be missed. This can be solved by enlarging the ROI, shown in figure 18. Howerver, this will result in longer sample time. Therefore, the performance is measured with the ROI consisting of 40, 60, 80, and 100 lines of pixels. The most important criterion is that the sample time is about 100 milliseconds with a ROI of 40 lines op pixels. The performance is considered valid if the sample time is not higher than 100 milliseconds plus 10 %, or 110 ms with a ROI of 40 lines of pixels. The sample time is measured by letting the prototype measure the ratio 1000 times and logging the time stamp. Then, the average sample time and standard deviation are calculated. A Raspberry Pi is a small computer with a full-blown operating system on it, which is not considered to be hard real-time. So, jitter is expected to be present.

7.1 Validation with respect to ratio measurements

The mean ratio in the ROI, measured with the prototype, is plotted in blue. The manually measured ratios are superimposed in red on the ratios measured with the prototype. The two lines are expected to follow about the same periodic pattern.



(a) *Marked chips* (b) *Regions of interest* (c) *captured w. prototype*

(d) *DSLR* (e) *test setup* (f) *captured with DSLR*

Figure 19: *Images of the validation process.*

7.1.1 Comparison with the blade set right

If the blade height is set right, the chip should in theory show an average ratio of about 67 %. The ratios measured with the prototype and with the caliper are shown in figure 20.

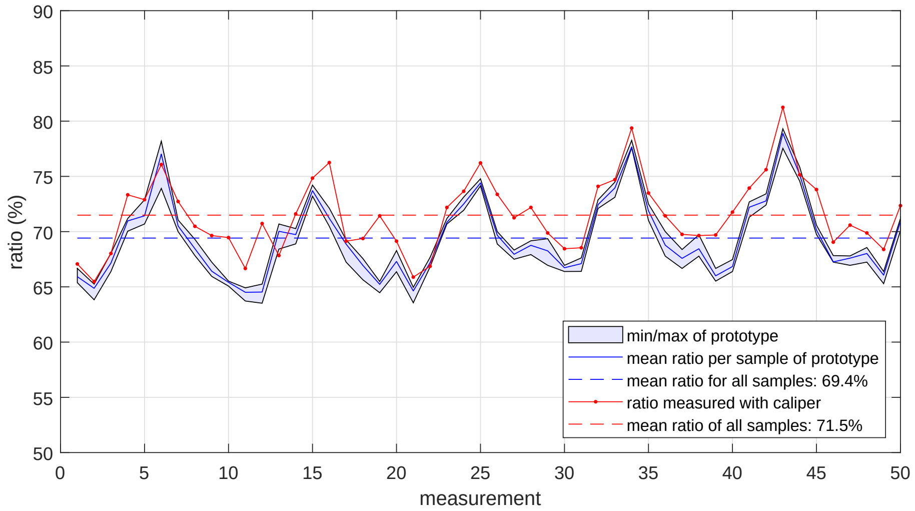


Figure 20: *Measured ratios with the blade set correctly. Should be about 67 %.*

The mean ratios measured with the prototype and manually with a caliper differ about 2 % in absolute terms, or 2 %-point.

7.1.2 Comparison with the blade set too deep

If the blade height is too deep, the ratio of the chip should clearly be below 67 %. The ratios measured with the prototype and manually with a caliper are shown in figure 21.

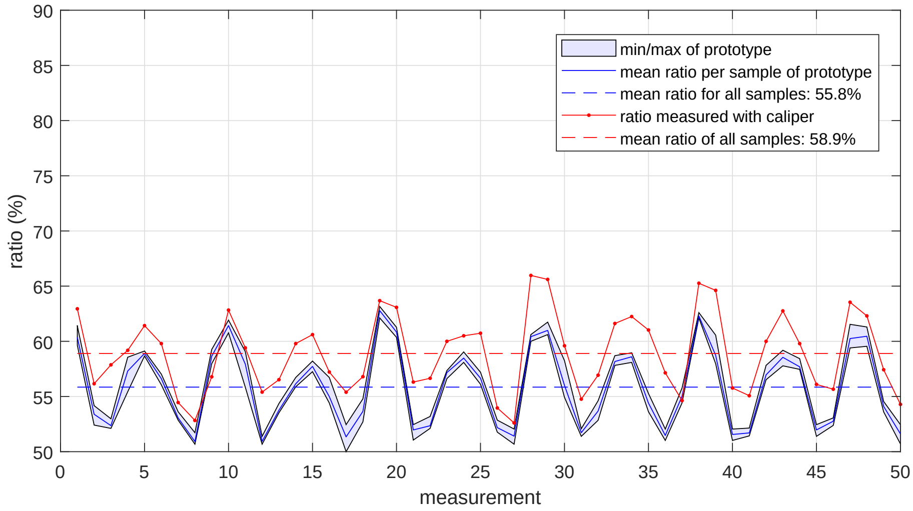


Figure 21: *Measured ratios with the blade set too deep.*

The mean ratios measured with the prototype and manually with a caliper differ about 3 %-point.

7.1.3 Comparison wit the blade set too shallow

If the blade height is too shallow, the ratio of the chip should be clearly above 67 %. The ratios are shown in figure 22.

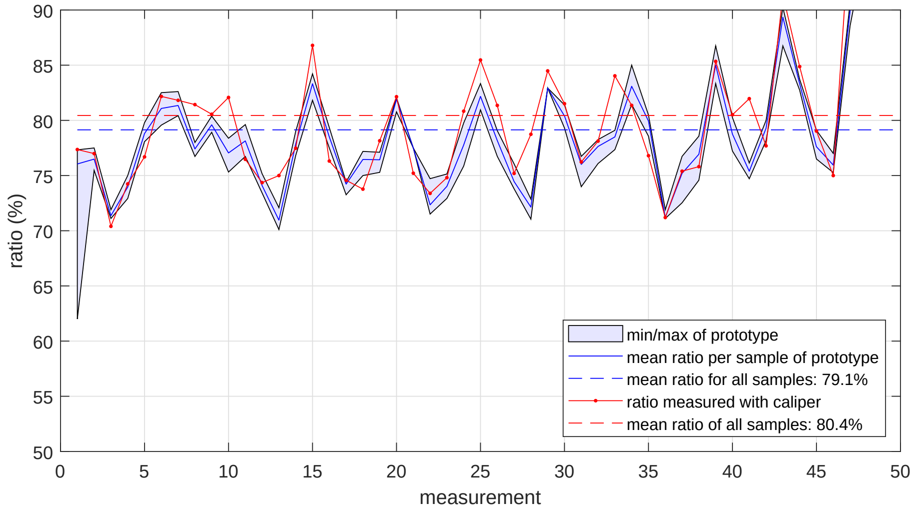


Figure 22: *Measured ratios with the blade set too shallow.*

The mean ratios measured with the prototype and manually with a caliper differ about 1 %-point.

7.1.4 Interpretation of the results

The ratios show a periodic pattern as expected and the waveforms are in phase, showing that the measurements are acquired correctly. The ratios measured with the prototype deviate about 1 tot 3 %-point from the ratios manually measured with a caliper.

7.2 Validation with respect to performance

The performance is visualized by plotting the sample times as blue dots. The average sample time with the standard deviation are shown by three vertical line through the dots.

Performance with 40 image lines
Figure 23 shows the results with a region of interest of 40 lines of pixels. This is the ROI shown in Figure 18.

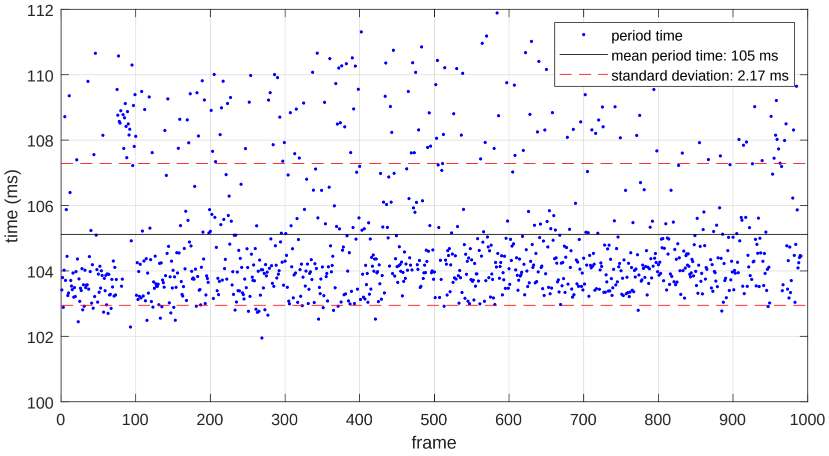


Figure 23: *Processing time by processing 40 image lines.*

Performance with 60 image lines
Enlarging the ROI to 60 lines of pixels results in a sample time of 153 milliseconds as shown in Figure 24.

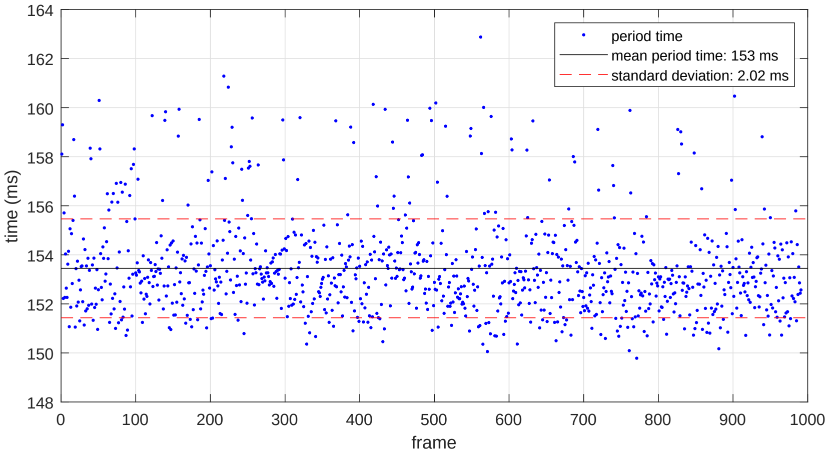


Figure 24: *Processing time by processing 60 image lines.*

Performance of 80 image lines

Enlarging the ROI ever further to 60 lines of pixels results in a sample time of 183 milliseconds as shown in Figure 25.

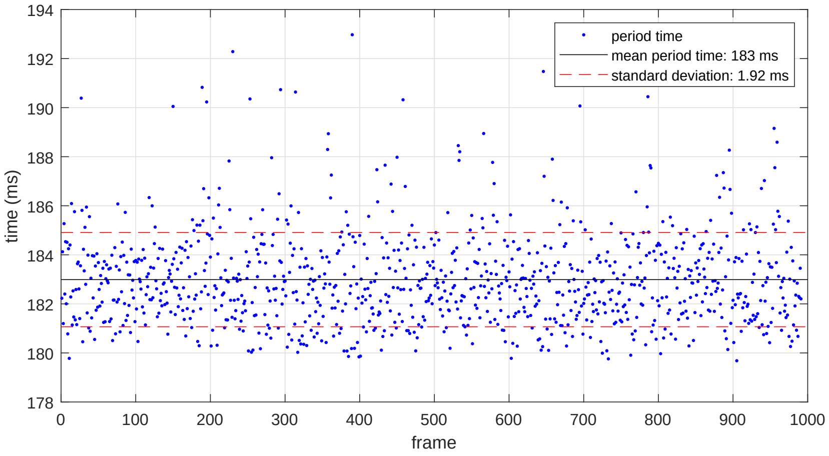


Figure 25: *Processing time by processing 80 image lines.*

Performance of 100 image lines

The largest ROI tested in this experiment is 100 pixels high and the results are shown in Figure 26.

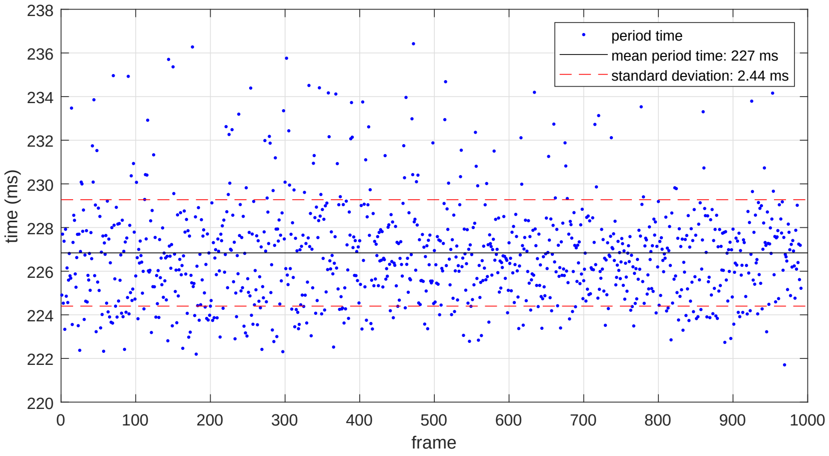


Figure 26: *Processing time by processing 100 image lines.*

7.2.1 Interpretation of the results

The sample time with a region of interest of 40 pixels high is 105 milliseconds. This is faster than the 110 milliseconds so the prototype is valid with respect to performance with a ROI of 40 pixels high. However, when the ROI is enlarged, the performance drops.

7.3 discussion

These results show that the prototype is valid with respect to ratio measurements. What stands out is that the average ratios, measured with the caliper, are above the average ratios measurements with the prototype. This is especially noticeable in Figure 21. An explanation can be that there is a static error in positioning the caliper. In particular, holding the caliper near the chip has to be done carefully to prevent squeezing the chip since it is very flexible. It may happen that the caliper is set too wide to not to touch the chip. Suppose that a constant width is added to the caliper reading. This adds a constant to both the numerator and denominator of the ratio, defined by Equation 1, pushing it towards one a bit. This effect increases when the true ratio goes further away from 1 towards 0.5. Comparing the three plots in Figures 20, 21, and 22 supports this.

An additional cause of errors is misalignment of the caliper, as shown in Figure 27. This picture is magnified a lot and the gap in between the caliper and the semiconductor is only a fraction of a millimeter. But, it does increase the ratio.

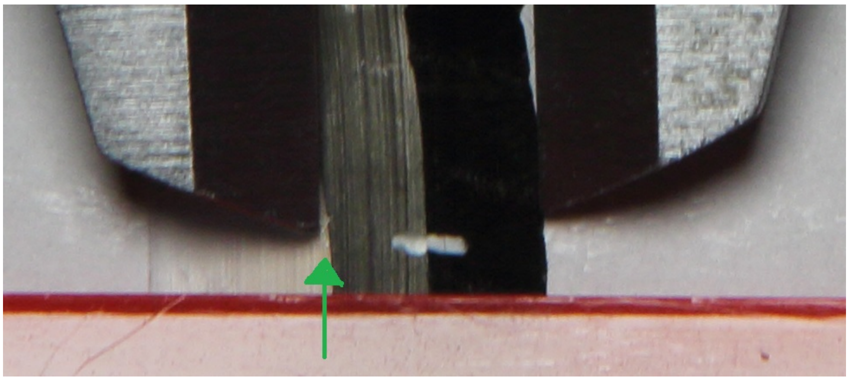


Figure 27: *Misalignment of the caliper shown by the green arrow.*

The relation between the height of the ROI and the sample time can be visualized by putting the sample times in a bar graph shown in Figure 28. The relation turned out to be about linear. Each pixel line adds roughly 1 ms to the sample time.

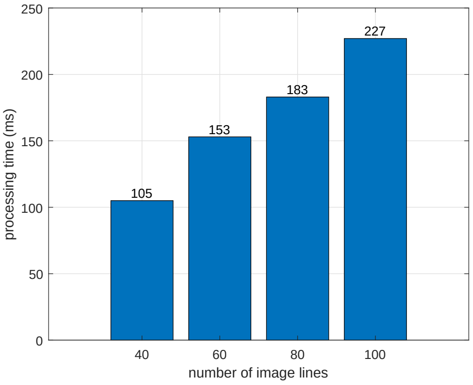


Figure 28: *All processing times visualized in a bar graph.*

7.3.1 Limitations

Frame rate
The tool acquires images at a rate of nearly 10 frames per second. The region of interest is about 2.5 mm wide. Therefore the chip is not recommended to move faster than about 25 millimeters per second. Otherwise, parts of chip can be missed. A solution is to enlarge the region of interest. However, more data must then be processed, slowing down the sample rate.

Illumination of the chip
Due to the high brightness of the LED bar, the camera needs about 20 images to settle to the correct image brightness. As a result, the prototype may produce erroneous measurements for the first two seconds.

Static human error in validation
Measuring the ratio manually with a caliper may introduce a static human error. It may be noticed that the measurements with the caliper are higher compared to the ratios measured with the prototype. This is especially visible in Figure 21, where the red line is constantly above the blue one. Although the validation results are good, it can be supported by letting different people measure the ratios of the chips used in this project and use these results to compare the automatically measured results to.

Surface finish of the cable
The blade may wear out over time and become blunt. This may result in a rough surface finish of the cable. Since the tool measured the upper side of the chip, it is not able to capture information about the cable surface finish, nor can it detect the physical state of the blade.

7.3.2 Recommendations

Surface finish of the cable
The prototype is unable to monitor the surface finish of the cable. More research can be done on what effects the physical state of the blade has on the cable surface finish and if the chip resembles information about the physical state of the blade. For example, a blunt knife might result in rough edges on the chip, which may be detectable by additional computer vision algorithms.

Time delay
The camera is not located exactly on the blade, but a small distance after it. As a result, there is a time delay from the control imput to the blade to the measurement output. This time delay is linearly proportional to the angular velocity of the peeling tool around the cable. Therefore, it is recommended to keep the angular velocity constant. Otherwise, the volatility of this delay has to be taken into account.

8 Conclusion

In this project, a prototype is developed to demonstrate that computer vision works for monitoring the peeling process. A signaling LED shows whether or not the ratio of the chip is right. The validation results show that the measurements acquired by the monitoring system deviates less than 4 %-point from the manually acquired measurements. In this thesis project, the research question was:

> *"Which quality aspects in the peeling process can be inspected by means of computer vision, and which principles for computer vision are needed for this?"*

Several quality aspects are mentioned in Table 1. However, the correct depth setting of the blade is the most important one. Measuring the blade depth setting is considered the most critical quality aspect. This quality aspect can be inspected by applying computer vision based on Bayesian pixel classification.

"How to quantify the quality of the peeling process?"

An important quality aspect of the peeling process is the depth setting of the blade. This quality aspect can be quantified by calculating a quality factor, which is the absolute value of the ratio of semiconducting material to the total width of the chip, minus the desired ratio of 2/3. The lower this quality factor the higher the quality. Therefore, the aim is to keep this quality factor as close to 0 as possible.

"What is a suitable camera and illumination setup?"

The monitoring system must fit in a limited space and the chip must be illuminated properly. A camera from which the lens its focal distance can manually be set within 2 centimeters is used. In this case a Raspberry Pi camera is chosen. An LED bar is taken from a commercial LED filament light bulb is used to illuminate the scene. A green background sheet is used, which makes it easier to distinguish the chip from the background.

"Which computer vision principles are suitable for measuring the quality aspects?"

The images can be processed well using Bayesian pixel classification. Applying Bayesian a second time with improved prior probability distributions based on the first iteration, resulted in even better image segmentation performance. The ratio of the chip can then automatically be measured from the segmented images.

"What are suitable performance indicators for the computer vision system?"

To measure the performance of the monitoring system, three pieces of chip are measured with the monitoring system and manually with a caliper. The monitoring system also measured the sample time. This resulted in two performance indicators. One is the error between the ratio measured with the monitoring system and measured with the caliper. The second performance indicator is the average sample time and the jitter on it.

"What is a suitable test setup to measure the performance of the system?"

A prototype is constructed by integrating the monitoring system in an existing peeling tool. This way, the researcher can test different computer vision algorithms by pulling chips through the prototype, as well as by peeling real cables to make experimenting more realistic.

In summary, in this thesis project, the following has been achieved:

- A computer vision based monitoring system is developed to quantify the quality of the peeling process.
- This quantity, acquired by the monitoring system, deviates less than 4 %-points from the quantity acquired manually with a caliper.
- This quantity can serve as sensor input for the control system in the Cable Splicing Robot project[5].
- A prototype is engineered to demonstrate the monitoring system.
- The monitoring system is ready to be used in the Cable Splicing Robot project[5].

References

[1] Alliander. *Alliander start site.* [Online; accessed 06-March-2023]. URL: https://www.alliander.com/.

[2] Alliander. *Factsheet kerngegevens Alliander.* 2022.

[3] Aivar Annamaa and contributors. *Thonny, Python IDE for beginners.* [accessed 04-November-2022]. URL: https://thonny.org/.

[4] Harrison C. *Xming X Server for Windows.* [accessed 20-October-2022]. URL: http://www.straightrunning.com/XmingNotes/.

[5] Tim Elderhorst Edwin Dertien. *cable splicing robot.* [accessed 8-March-2023]. URL: https://www.ram.eemcs.utwente.nl/research/projects/cable-splicing-robot.

[6] Nick Efford. *Digital Image Processing, a practical introcution using Java.* 1st ed. Addison-Wesley, Peason Education, 2000. ISBN: 0-201-59623-7.

[7] Python Software Foundation. *Python.* [accessed 04-November-2022]. URL: https://www.python.org/.

[8] Raspberry Pi Foundation. *Raspberry Pi camera module V2.* [accessed 24-November-2022]. URL: https://www.raspberrypi.com/products/camera-module-v2/.

[9] Raspberry Pi Foundation. *Raspberry Pi community forum.* [accessed 09-March-2023].

[10] Raspberry Pi Foundation. *Raspberry Pi home page.* [accessed 19-October-2022]. URL: https://www.raspberrypi.com/.

[11] Raspberry Pi Foundation. *Raspberry Pi OS.* [accessed 19-October-2022]. URL: https://www.raspberrypi.com/software/.

[12] Ferdinand van der Heijden. *Image based measurement systems.* 1st ed. Wiley, 1995. ISBN: 978-0471950622.

[13] Intel Corporation Willow Garage Itseez. *OpenCV home page.* [accessed 20-October-2022]. URL: https://opencv.org/.

[14] Bischop Christopher M. *Pattern Recognition and Machine Learning.* Springer, 2011.

[15] Mathworks. *Covariance.* [accessed 17-March-2023]. URL: https://nl.mathworks.com/help/matlab/ref/cov.html.

[16] Mathworks. *Makers of Matlab and Simulink.* [accessed 04-November-2022]. URL: https://nl.mathworks.com/.

[17] Tatham S. Alexandra L. Harris B. Nevis J. *PuTTY: a free SSH and Telnet client.* [accessed 20-October-2022]. URL: https://www.chiark.greenend.org.uk/~sgtatham/putty/.

[18] J.M.H. Lenoir P.M.E.M. van der Grinten. *Statistische procesbeheersing.* Het Spectrum B.V., 1973.

[19] Dassault Systèmes. *Solid Works.* [accessed 24-October-2022]. URL: https://www.solidworks.com/.

[20] William Thue. *Electric Power Cable Engineering.* 3rd ed. CRC Press, 2012. ISBN: 978-1439856451.

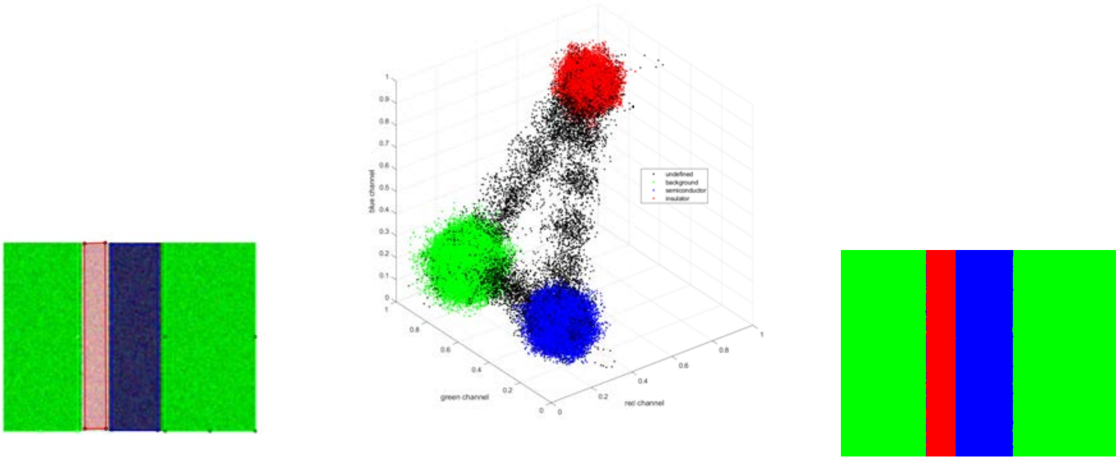
[21] Irem Ulku and Erdem Akagündüz. "A Survey on Deep Learning-based Architectures for Semantic Segmentation on 2D Images". In: *Applied Artificial Intelligence* 36.1 (Feb. 2022). DOI: 10.1080/08839514.2022.2032924. URL: https://doi.org/10.1080%2F08839514.2022.2032924.

[22] Uitvoeringsinstituut Werknemersverzekeringen. *spanningsindicator.* [accessed 7-March-2023]. URL: https://www.werk.nl/arbeidsmarktinformatie/dashboards/spanningsindicator.

[23] Gonzales R.C. Woods R.E. *Digital Image Processing.* 3rd ed. Pearson Education, Prentice Hall, 2008. ISBN: 978-0135052679.

A Example of segmentation based on Bayesian classification

Suppose, an artificially generated example image is manually segmented, as shown in Figure 29a. The pixels can then be positioned in 3D space by using their three color channels as Cartesian coordinates. The result is that the three classes pile up in three specific locations in 3D space, as shown in Figure 29b.



(a) *Manually segmented* (b) *Three classes in 3D space* (c) *Segmented.*

Figure 29: *Example of what statistical inference implies. The black dots are missed in manual segmentation.*

The dark gray, or near black color of the semiconducting material results in the pixels being closely located near the origin, as shown by the blue point cloud in Figure 29b. The light gray, or near white insulating material results in pixels located closely to (1,1,1), shown by the red point cloud in Figure 29b. The green background pixels have low coordinate values for the red and blue channel and high for green. The mean location $\vec{\mu}_m$ for each class is what is calculated by equation 9. The covariance matrices C_m, calculated by equation 10, resemble the shape of the point clouds.

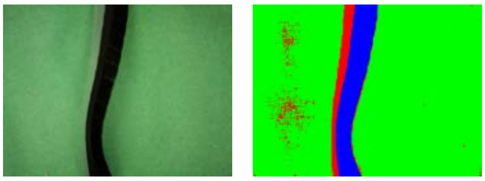
The hypothesis is that the pixels in consecutive images end up on about the same locations when plotted in 3D space. In other words, the pixels are separable, which implies that they can be classified using Bayes' theorem. Applying Bayes' theorem on the example image results in the segmented image 29c. Although it is a bit unfair to apply Bayesian classification on the same image as used for inferring the parameters, this example shows promising results.

Important note: In this example, the point clouds appear concentrated on spherical clusters in 3D space. It may create the impression that it is possible to separate pixels by defining square boxes in space and classify new pixels based on in which box they end up. However, in real life, the point clouds may not be circular at all and may stretch out through space, meaning that it is not possible anymore to define square boxes enclosing pixels of specific classes. This is why the idea arose of using Bayes' theorem to deal with that. As long as the points do not mix, Bayes' theorem should be able to separate them.

B Example of using contextual information

The prior probability, defined by equation 7, is constant over the whole image plane. So, at least two details are not included in the prior probability. One, insulation material is always on one side of the semiconducting material. Two, the insulating material will unlikely be wider than the semiconducting material. These details can be embedded in the prior probability.

An input image, shown in Figure 30a, is segmented with uniform prior probability. This results in a segmented image of moderate quality, as shown in Figure 30b. The background sheet and insulating material contain misclassified pixels. Fortunately, the black semiconducting material is classified correctly. Therefore, this segment is used to create new prior probability distributions for the three classes.

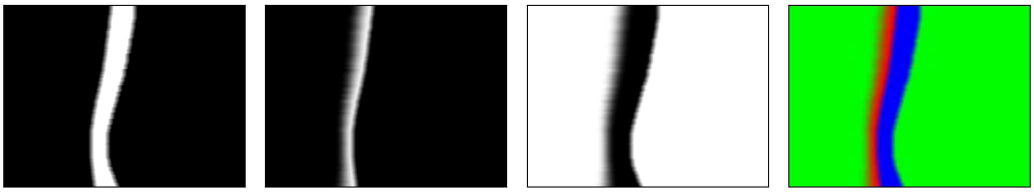


(a) *Image of the chip.* (b) *Segmented image.*

Figure 30: *Example of an image of a chip and the first segmentation iteration.*

The semiconducting material is segmented pretty well and can by itself suit as a prior probability distribution function, depending on the location in the image expressed in row r and column c coordinates. To leave some room for reclassifying pixels, the probability distribution function is smeared out a bit, resulting in the prior probability distribution function $P_{r,c}(semiconductor)$ shown in Figure 31a. This new prior probability replaces the initial prior probability $P(semiconductor)$ and is now dependent on the pixel its location.

Insulator material will likely not be on the right-hand side of the semiconductor. So the prior probability of a pixel being insulator material is zero on the right-hand side of the semiconductor and on the semiconductor itself. Insulator material will likely be on the left-hand side of the semiconductor. The insulator cannot be wider than the semiconductor material. So, the probability close to the semiconductor should be 1 and steadily decrease towards 0 further away from the semiconductor. Figure 31b shows the new prior probability $P_{r,c}(insulator)$. The new prior probability for the background is 1 minus the probability distributions for the semiconductor and insulator and is shown in figure 31c. For clarity, the three new prior probabilities are visualized as one color image, shown in figure 31d.



(a) $P_{r,c}(semicon)$ (b) $P_{r,c}(insulator)$ (c) $P_{r,c}(background)$ (d) $P_{r,c}(k)$

Figure 31: *The new prior probabilities dependent on the location, row r and column c, in the image. Figure 31d combines the three prior probabilities for visualization.*

Segmenting image 30a a second time with the new prior probability density function results in the segmented image in figure 32.

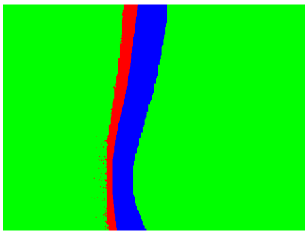


Figure 32: *The image segmented.*