Department of Physics and Astronomy

University of Heidelberg

Master thesis

in Computer Engineering

submitted by

Habib Gahbiche

born in Sousse

2018

(Title)

(of)

(Master thesis)

This Master thesis has been carried out by Habib Gahbiche

at the

Institute of Environmental Physics

under the supervision of

Prof. Dr. Bernd Jähne

and

Prof. Dr. Karl-Heinz Brenner

Institute of Computer Engineering

(Titel der Masterarbeit - deutsch):

(Abstract in Deutsch, max. 200 Worte. Beispiel: **?**)
Lorem ipsum dolor sit amet, consectetur adipisici elit, sed eiusmod tempor incidunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquid ex ea commodi consequat. Quis aute iure reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint obcaecat cupiditat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi. Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.
Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi.

(Title of Master thesis - english):

(abstract in english, at most 200 words. Example: **?**)
Lorem ipsum dolor sit amet, consectetur adipisici elit, sed eiusmod tempor incidunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquid ex ea commodi consequat. Quis aute iure reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint obcaecat cupiditat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi. Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.
Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi.

Contents

1 Introduction

This is my intro

2 Theory

In this chapter we explain the theoretical concepts relevant to this thesis. We start with explaining the physics behind our method in section 2.1, in particular how bubbles interact with light. Next, we discuss the mathematical basics necessary for image processing such as Fourier theory and convolution in section 2.2. Section 2.3.1 explains the principle behind machine learning that our method relies on for classification. Finally, section 2.3 formally introduces the object detection problem and our chosen criteria for evaluation.

2.1 Bubble physics

2.2 Image processing

In the following we represent an image as a two dimensional signal written as a matrix $\boldsymbol{g}$. Therefore, $g_{m,n}$ denotes the pixel (i.e. picture element) at the m-th row corresponding to the n-th column. The chosen coordinate system is described in figure 2.1.

2.2.1 Fourier theory

The Fourier transform is an important image processing tool which is used to decompose an image into its since and cosine components. The output of the transformation represents the image in the Fourier or frequency domain, while the input image is the spacial domain. In the Fourier domain image, each point represents a particular frequency contained in the spatial domain image. The *continuous* two-

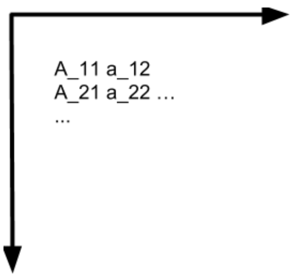


Figure 2.1: Notation and coordinate system

dimensional Fourier transform is defined as

$$\mathscr{F}\{g(\mathbf{x})\} = \hat{g}(\mathbf{k}) = \int_{-\infty}^{\infty} g(\mathbf{x})\exp\left(-2\pi\mathrm{i}\mathbf{k}^T\mathbf{x}\right)\mathrm{d}\mathbf{x} \tag{2.1}$$

and the inverse Fourier transform

$$\mathscr{F}^{-1}\{\hat{g}(\mathbf{k})\} = g(\mathbf{x}) = \int_{-\infty}^{\infty} \hat{g}(\mathbf{k})\exp\left(-2\pi\mathrm{i}\mathbf{k}^T\mathbf{x}\right)\mathrm{d}\mathbf{x} \tag{2.2}$$

Where $\mathbf{x}$ and $\mathbf{k}$ are the two dimensional space and frequency vectors respectively.

Images however are discrete two dimensional signals, we therefore need to apply the *Discrete* Fourier transform or DFT, defined as

$$\mathrm{DFT}\{g_{m,n}\} = \hat{g}_{u,v} = \frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} g_{m,n}\exp\left(-\frac{2\pi\mathrm{i}mu}{M}\right)\exp\left(-\frac{2\pi\mathrm{i}nu}{N}\right) \tag{2.3}$$

Similarly, the inverse 2-D DFT is defined as

$$\mathrm{IDFT}\{\hat{g}_{u,v}\} = g_{m,n} = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} \hat{g}_{u,v}\exp\left(\frac{2\pi\mathrm{i}mu}{M}\right)\exp\left(\frac{2\pi\mathrm{i}nu}{N}\right) \tag{2.4}$$

2.2.2 Convolution

Convolution is one of the most important operations in signal processing. Convolving two signals g and h produces a third signal that expresses how the shape of one is modified by the other. Formally, we define the continuous convolution as follows

$$(g \star h)(\mathbf{x}) = \int_{-\infty}^{\infty} h(\mathbf{x}')g(\mathbf{x}-\mathbf{x}')\mathrm{d}\mathbf{x} \tag{2.5}$$

and the discrete two dimensional convolution as

$$g'_{m,n} = \sum_{m'=0}^{M-1}\sum_{n'=0}^{N-1} h_{m',n'}g_{m-m',n-n'} \tag{2.6}$$

One important property of convolution is that we can express it as a multiplication in the Fourier domain.

$$\mathscr{F}\{g \star h\} = NM\hat{h}\hat{g} \tag{2.7}$$

This property, together with the fast Fourier implementation of the Fourier transform allows a fast computation of convolutions.

At the edge of the image, we typically extend the image with zero values (i.e. zero padding). This introduces an error when applying filters at the image border and we will mostly exclude the border when using filters (see chapter 5 for more details).

2.2.3 Smoothing

Smoothing an image means convolving an image with a smoothing filter. A smoothing or averaging filters must ideally fulfill following conditions

1. Zero-shift: $\Im(\hat{h}(\mathbf{k})) = 0$
2. Preservation of mean value: $\hat{h}(0) = 1$
3. Monotonous decrease: $\hat{h}(k_1) \leq \hat{h}(k_2)$ for $k_2 > k_2$
4. Isotropy: $\hat{h}(\mathbf{k}) = \hat{h}(|\mathbf{k}|)$ stimmt das ??

In this work, we will be using Gaussian filters for one and two dimensional smoothing. Although Gaussian filters are not ideal, e.g. isotropy is violated for small standard deviations, it is still a good approximation for an ideal low pass filter. Computing the Fourier transform (for convolution) is also faster for a Gaussian filter. The m-th component of a one dimensional Gaussian filter mask can be obtained from the Gaussian function

$$G_m = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(m-\mu)^2}{2\sigma^2}\right) \tag{2.8}$$

Where μ is the mean, i.e. Gaussian peak's position and σ is the standard deviation, i.e. peak's width.

Figure 2.2 show a Gaussian curve in one and two dimensions as well as the result of convolving an image with a Gaussian filter mask. Note how the image becomes blurry, i.e. large wave numbers have been suppressed.

2.2.4 Edges and Derivation

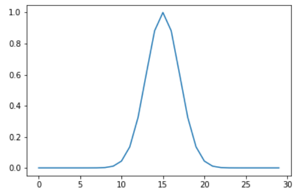
An edge can be defined as a set of continuous pixel positions where an abrupt change of intensity (i.e. gray value) occurs. Therefore, edge detection is based on differentiation, where in discrete images differentiation is replaced by discrete differences that are mere approximation to differentiation. There is also the need to not only know where edges are, but also how strong they are. Figure **??** shows that in the one dimensional case, edges can be detected by applying first and second derivatives to the signal.

In continuous space, a partial derivative operation is defined as

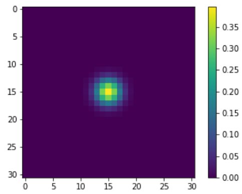
$$\nabla = \left[\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right] \tag{2.9}$$

and its corresponding Fourier transform is

$$\mathscr{F}\{\nabla\} = 2\pi \mathrm{i}\mathbf{k} \tag{2.10}$$



(a) 1D Gaussian signal with $\mu = 15$ and $\sigma = 2$



(b) 2D Gaussian signal with $\mu_x = \mu_y = 15$ and $\sigma_x = \sigma_y = 2$

(c) Original image

(d) After convolution with 2D Gaussian mask

Figure 2.2: Gaussian smoothing filter

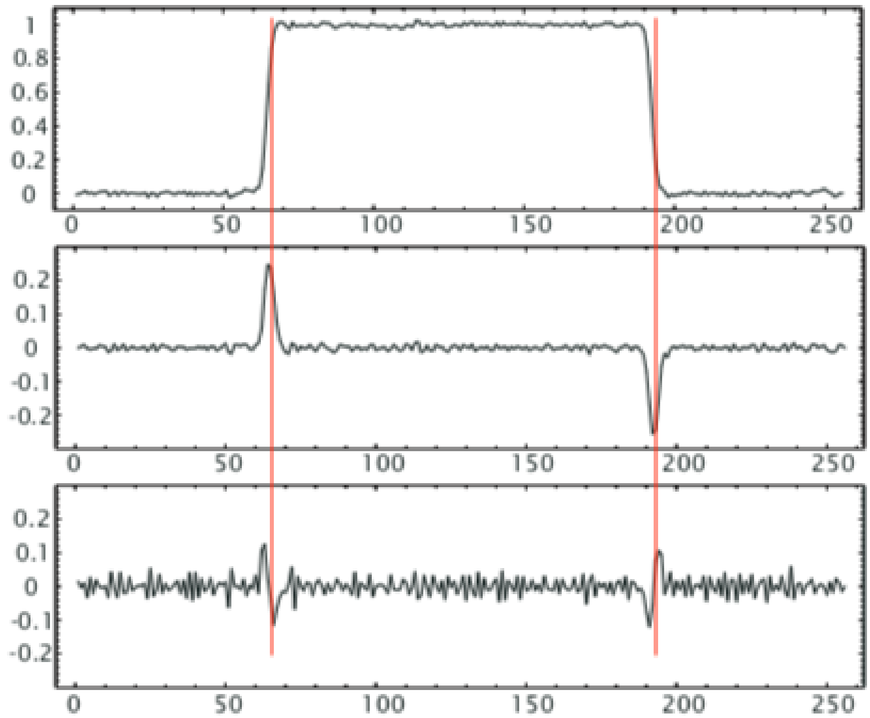


Figure 2.3: Original 1D signal. First derivative. Second derivative

For the second derivative we need to consider all possible combinations of second order partial differential operators of a two dimensional signal. The resulting 2×2 matrix is called the Hessian matrix

$$\mathbf{H} = \begin{bmatrix} \dfrac{\partial^2}{\partial x^2} & \dfrac{\partial^2}{\partial x \partial y} \\ \dfrac{\partial^2}{\partial y \partial x} & \dfrac{\partial^2}{\partial y^2} \end{bmatrix} \tag{2.11}$$

and its Fourier transform is

$$\mathscr{F}\{\mathbf{H}\} = -4\pi^2 \mathbf{k}\mathbf{k}^T \tag{2.12}$$

Edge detectors can be implemented as filters h that operate on a two dimensional grid. From the above equations we can derive the general properties for these filters:

1. Zero-shift:
   - $90°$ phase shift for first order derivative, implying $\Im\{\hat{h}\} \neq 0$ and an antisymmetric filter mask, i.e $h_{-n} = -h_n$
   - a second order derivative operator must be symmetric in order to satisfy the zero shift property, i.e. $h_{-n} = h_n$

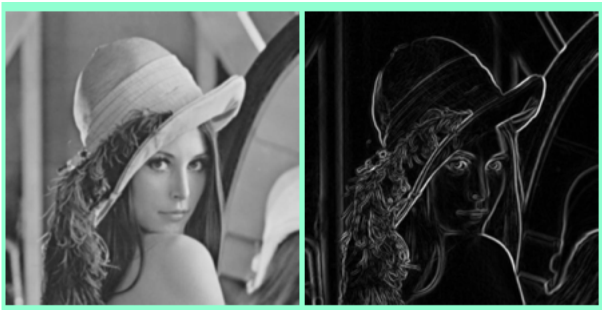


Figure 2.4: Left: original image. Right: gradient image

2. Suppression of mean value: $\hat{h}(k_i = 0) \Leftrightarrow \sum_{\mathbf{n}} h_{\mathbf{n}} = 0$
3. isotropy: For good edge detection, the edge detector's response must not depend on the direction of the edge.
   - first order derivative $\hat{h}(\mathbf{k}) = \pi i k_i \hat{b}(|\mathbf{k}|)$
   - second order derivative $\hat{h}(\mathbf{k}) = \pi^2 k_i^2 \hat{b}(|\mathbf{k}|)$

   where k_i denotes the wave number in the i-th direction and b is an isotropic smoothing filter that fulfills the conditions

$$\hat{b}(\mathbf{0}) = 1, \qquad \nabla_k \hat{b}(|\mathbf{k}|) = \mathbf{0} \tag{2.13}$$

In this work we will be using the Sobel filter masks as defined in equation (2.14) in order to compute derivatives in x and y directions.

$$S_x = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}, \qquad S_y = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & 2 \\ 1 & 0 & -1 \end{bmatrix} \tag{2.14}$$

At each point in the image, the resulting gradient approximations can be combined to give the gradient magnitude image S using:

$$S = \sqrt{(S_x \star G)^2 + (S_y \star G)^2} \tag{2.15}$$

where G is the input image. Figure 2.4 shows the result of applying the derivation operator on an image. Note how edges have different magnitude depending on their strength.

2.2.5 Orientation and Structure Tensor

Although derivation is useful to determine the gradient magnitude and its direction in an image, it doesn't tell us much about gradient directions in a specific neighborhood of a point. Figure 2.5 shows that in a neighborhood with ideal orientation gray values change in one direction only. Generally, the direction of local orientation

Figure 2.5: Ideal local neighborhood described by a unit vector $\tilde{\mathbf{n}}$

can be denoted with a unit vector $\tilde{\mathbf{n}}$. If we orient the coordinate system along the principal directions, the gray values become a one dimensional function and a simple neighbourhood can be represented by

$$g(\mathbf{x}) = g(\mathbf{x}^T\tilde{\mathbf{n}}) \tag{2.16}$$

The drawback of this representation however, is that it cannot distinguish between neighborhoods with constant values and isotropic orientation distribution. So if we define the optimum orientation as the orientation that shows the least deviations from the directions of the gradient, we can expressed it using the unit vector $\tilde{\mathbf{n}}$ as

$$\nabla g^T\tilde{\mathbf{n}} = \cos[\angle(\nabla g, \tilde{\mathbf{n}})] \qquad \Leftrightarrow \qquad \left(\nabla g^T\tilde{\mathbf{n}}\right)^2 = |\nabla g|^2\cos^2[\angle(\nabla g, \tilde{\mathbf{n}})] \tag{2.17}$$

We can see that this quantity is maximized when the orientation is along the unit vector $\tilde{\mathbf{n}}$, i.e. when ∇g and $\tilde{\mathbf{n}}$ are either parallel or antiparallel. Therefore, the following integral is maximized in a local neighborhood:

$$\int w(\mathbf{x}-\mathbf{x}')\left(\nabla g(\mathbf{x}')^T\tilde{\mathbf{n}}\right)^2 \mathrm{d}\mathbf{x}' \tag{2.18}$$

where the window function w determines the size and shape of neighborhood around a point $\mathbf{x}$ in which the orientation is averaged. The maximization problem must be solved for each point $\mathbf{x}$, so we can write the maximization problem as follows:

$$\tilde{\mathbf{n}}^T\mathbf{J}\tilde{\mathbf{n}} \to \max \tag{2.19}$$

From equation 2.18 and 2.19 we can define **the structure tensor** as

$$\mathbf{J} = \int w(\mathbf{x}-\mathbf{x}')\left(\nabla g(\mathbf{x}')\nabla g(\mathbf{x}')^T\right)\mathrm{d}\mathbf{x}' \tag{2.20}$$

The pq-th component of this tensor is therefore given by

$$J_{pq} = \int_{-\infty}^{\infty} w(\mathbf{x}-\mathbf{x}')\left(\frac{\partial g(\mathbf{x}')}{\partial x'_p}\frac{\partial g(\mathbf{x}')}{\partial x'_q}\right)\mathrm{d}\mathbf{x}' \tag{2.21}$$

Rotating equation 2.19 into principle coordinate system yields:

$$\begin{bmatrix} n'_1 & n'_2 \end{bmatrix}\begin{bmatrix} J'_{11} & 0 \\ 0 & J_{22} \end{bmatrix}\begin{bmatrix} n'_1 \\ n'_2 \end{bmatrix} = J' = J'_{11}n'_1 + J'_{22}n'_2 \to \max \tag{2.22}$$

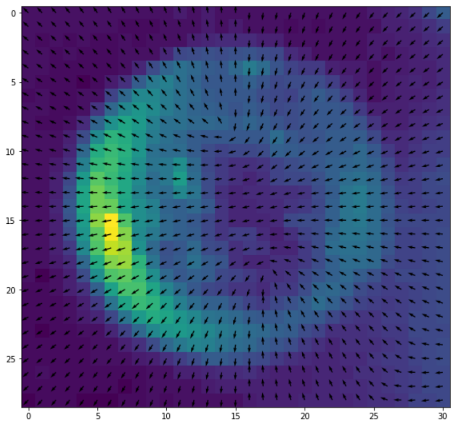


Figure 2.6: Orientation angle using sobel filters for derivation and a Gaussian mask with $\sigma = 1$ for smoothing.

We can see that J' is maximized for $\tilde{\mathbf{n}} = [1 \quad 0]^T$ (assuming $J'_{11} > J'_{22}$), where the maximum value is J_{11}, so solving this problem is equivalent to solving the eigenvalue problem for $\mathbf{J}$. We can then extract the orientation θ as follows:

$$\begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} J_{11} & J12 \\ J_{21} & J_{22} \end{bmatrix} \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \tag{2.23}$$

Using trigonometric identities, this yields

$$\tan(2\theta) = \frac{2J_{12}}{J_{11} - J_{22}} \tag{2.24}$$

For discrete images, we use Sobel filters as defined in equation 2.14 for derivation and a Gaussian smoothing mask introduced in section 2.2.3. Computing the elements of a structure tensor for an image G therefore requires following steps:

1. $G_x = S_x \star G$
   $G_y = S_y \star G$
2. $J_{11} = M \star (G_x \times G_x)$
   $J_{12} = J_{21} = M \star (G_x \times G_y)$
   $J_{22} = M \star (G_y \times G_y)$

Where M is a smoothing mask and S_x and S_y are derivation masks in x and y directions respectively.

Figure 2.6 shows an extracted orientation from a bubble image using the structure tensor.

2.3 The object detection problem

Our proposed algorithm recognizes bubbles (classification) in an image and estimates their respective centers and radii (localization). This problem of classification and

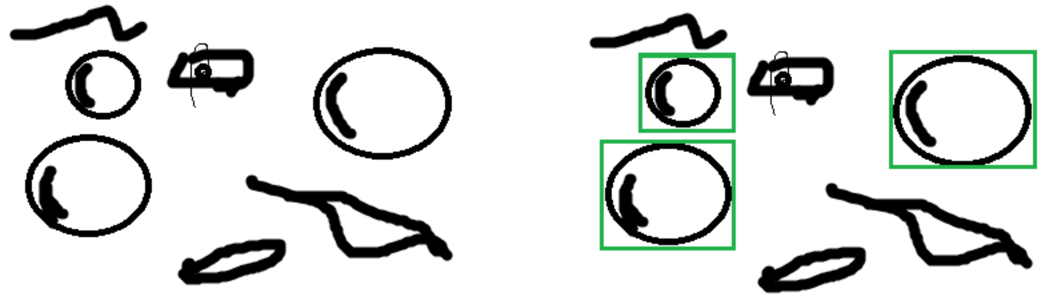


Figure 2.7: Left: Input image. Left: Output of an object detection algorithm dawn as bounding boxes around bubbles

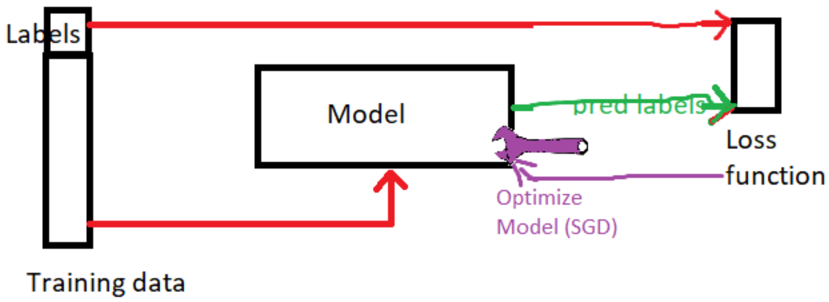
localization is known as the object detection problem. Figure 2.7 shows a typical output of an object detection algorithm.

There has been a lot of progress in this field thanks to deep learning algorithms (see next section) that rely on training a relatively complex model a large amount of annotated data. The sate of the art algorithms include region based methods such as Faster R-CNN (cite **?**), where many candidate regions are first extracted and then classified and single evaluation methods such as YOLO (cite **?**) where bounding boxes and class probabilities are estimated with one single neural network in a single evaluation. Although these algorithms perform very well on typical photographs and support between 1000 and 9000 different classes, applying them to our problem yields very bad results (see chapter 5). We still however opt for a machine learning based approach for the classification part, and then perform localization with more straightforward methods.

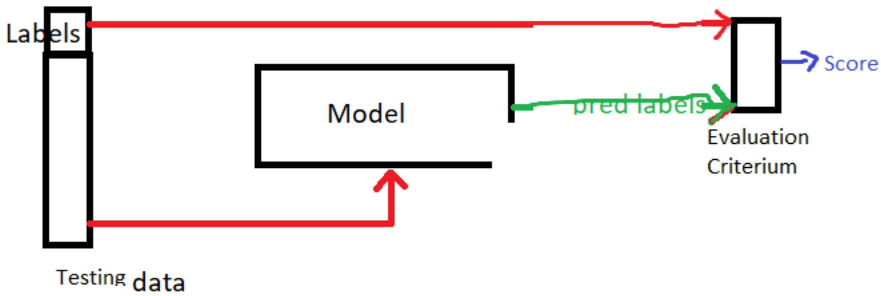
2.3.1 Classification and Machine Learning

In our work, we use machine learning techniques mainly for signal classification purposes. For instance, both of the proposed algorithms rely on binary classification using logistic regression and convolutional neural network. In this section, we explain the principle behind machine learning and the main idea behind the chosen model. A more thorough argumentation about the chosen architecture and model-specific parameters can be found in chapter 5.

Machine learning is commonly defined as the practice of using algorithms to parse data, learn from it, and the make a determination or prediction about something. In our case, this means training our model with a large number of bubble and background instances, in order to predict whether a given signal corresponds to a bubble or not. Figure 2.8 summarizes the principle behind machine learning.



(a) Training



(b) Testing

Figure 2.8: Training and testing in machine learning

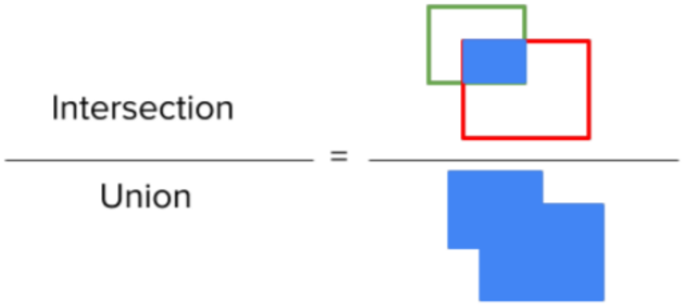


Figure 2.9: Definition of intersection over union. Green represents predicted bounding box and red is the ground truth. Areas are drawn in blue

Terminology

- **Training** is changing the model's parameters based on the model's output. Training is also referred to as optimization. When training data is annotated, we speak of supervised training.

- **Testing** is assigning a score to a trained model based on annotated testing data. The score is typically by an evaluation criteria and is independent of the model itself.

- **Validation** is a form of testing, so it assigns a score to a trained model. Validation is needed when we want to choose the best model among some trained candidate models based on their testing performance. Since changing a parameter of the algorithm (in this case the whole model) based on the output is equivalent to training by definition, we need an independent data set to compute the final score of the algorithm, i.e. validation data.

Evaluation Criteria: IoU@p-mAP

We evaluate our object detection algorithm using the *intersection over union at p mean average precision* criteria as shown in figure 2.9. As the name suggests, we compute the area of intersection between the predicted and the ground truth bounding box and then divide it by union area of the two boxes. This means that IoU $=1$ corresponds to a perfect prediction and IoU $=0$ is a complete miss (assuming original images were annotated perfectly). So we choose a threshold p, so that IoU $>= p$ or simply IoU@p corresponds to a correct prediction or *true positive*.

Mean average precision is the average of the maximum precision at different recall values, where **precision** and recall are defined as

$$precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{2.25}$$

$$recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{2.26}$$

From the above definition it becomes clear that precision measures how accurate the predictions are and recall measures how well all the positives can be found.

Since our algorithm performs binary classification, *mean average precision* is equivalent to *average precision* in our case.

Note: This criteria only shows how well the algorithms is at detecting bubbles from an image, and does not necessarily make a strong statement on how well the measurement technique as a whole performs at estimating bubble size distributions in a water.

3 Related Work

In this chapter we briefly discuss two similar bubble measurement techniques that aim to estimate bubble concentrations in water. The first is a master's thesis by (cite Leonie), where bubbles, where a bright field method was used in lab, so that bubbles appeared as dark circles on camera, and the second is a field measurement technique (cite R. Al-Lashi, S. Gunn), where bubbles were artificially lit from below, yielding similar images to our own.

3.1 Bright field method

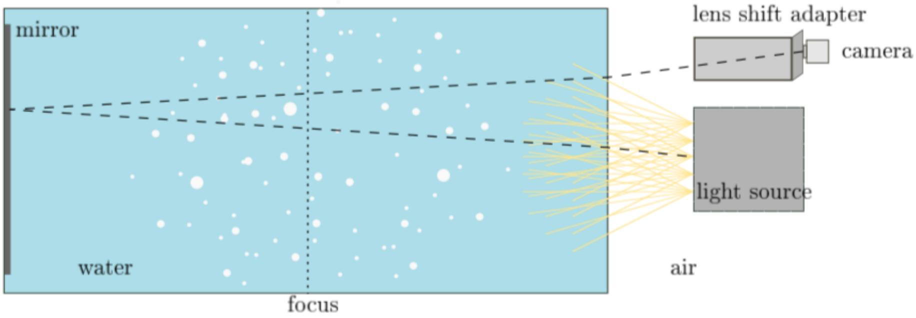
As an improved method from (cite Michler), this technique uses a bright field method as shown in figure **??**. Bubbles is this experimental setup are backlit, so that they appear completely dark except for one spot in the center (see section 2.1). Note the need to use a lens shift adapter because the incoming rays are not parallel.

Depth of field calibration was based on the edges between bubbles and background. The higher the edge's magnitude, the closer the bubble is to the focus plane. A calibration target made of hollow circles was used to calibrate the depth of field (figure 3.1c).

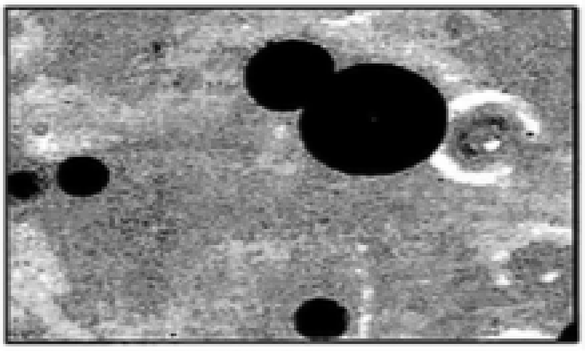
The images obtained by this method are arguably the most important advantage of this technique, since it reduces bubbles to simple circles, making them relatively easy to detect, whereas our algorithms need to recognize significantly more complex patterns than mere circles. In contrast to our method however, the bright field technique constrains how close the camera can get to the water surface. Also, (cite Leonie) does not define a criteria that measures how well the circle detection algorithm performs, so the assumption is that precision is close to perfect, which introduces an error to bubble concentration measurement that was essentially neglected. Nevertheless, this work was very important for newly developed method because it offered an estimation of the boundary conditions, such as the largest possible bubble radius and the largest possible bubble concentration, that were important to determine the different hyperparameters of our algorithm and therefore contributing to the accuracy of our algorithms.

3.2 Bubble optical imaging instrument

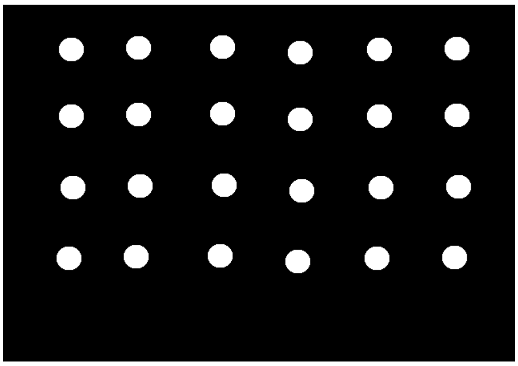
This technique submerges an optical imaging instrument (figure 3.2a) underwater and illuminates a thin slice of water of volume 4cm $\times$ 4cm $\times$ 5mm. We found this method particularly interesting because it produces images similar to our own.



(a) Experimental setup of a light field method at Aeolotron facility

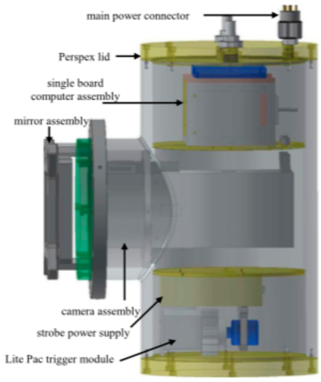


(b) Sample result image

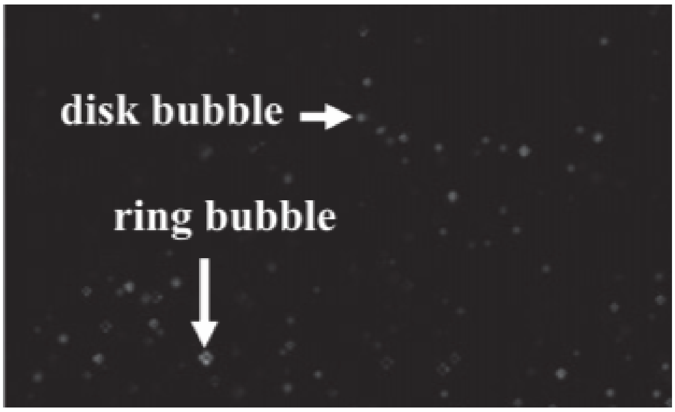


(c) calibration target

Figure 3.1: Bright field method



(a) Bubble optical imaging instrument. The submerged device



(b) Sample result image containing two types of bubbles: filled circles (disks) and rings.

Figure 3.2: Bubble detection with the bubble optical imaging instrument

However, applying the algorithm from this paper did not yield good results. This is most likely due to the fact that our method does not illuminate the bubbles from above, making the bubble's circular shape not easily detectable with a Hough transform (cite Hough transform, Al-Lashi)

4 Experimental Setup

4.1 Requirements

4.2 Aquarium

4.3 Aeolotron

4.4 Calibration

5 The Algorithm

5.1 BubbleNet

5.2 Curvature based

5.3 Calibration

Appendix

A Lists

A.1 List of Figures

A.2 List of Tables

Erklärung:

Ich versichere, dass ich diese Arbeit selbstständig verfasst habe und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Heidelberg, den 30.09.2018 . . . .