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

Abstract:

The contribution of air bubbles emerging from breaking waves to the gas exchange between atmosphere and ocean is still a controversial topic. Therefore, it is necessary to estimate bubble concentrations as well as their distribution in order to determine their importance to the gas exchange.
In this thesis, a new measuring technique has been developed, where bubbles and their radii can be automatically and reliably detected in a wind-wave facility. Lighting bubbles from below allows the detection of bubbles that are very close to the water surface.
Two separate algorithms have been developed to solve the bubble detection problem. The first is a deep learning approach that works best for measurements with a low bubble concentration. The second one uses a classical machine learning and image processing approach to detect bubbles and estimate their radii. It was designed to perform well for images with high bubble concentrations. Localizing bubbles in the third dimension was also made possible using a depth from focus technique.

Zusammenfassung:

Der Beitrag der durch brechende Wellen entstehende Luftblasen zum Gasaustausch zwischen Ozean und Atmosphäre ist umstritten. Deshalb ist es wichtig, sowohl die Konzentration als auch die Verteilung dieser Luftblasen abzuschätzen, um eine kräftige Aussage über ihren Beitrag zum Gasaustausch treffen zu können. In dieser Arbeit wurde eine neue Messmethode sowie bildverarbeitungsbasierte Algorithmen entwickelt, mit deren Hilfe Blasen und deren Radien zuverlässig und automatisiert in einem Wind-Wellenkanal detektiert werden können. Eine Beleuchtung der Luftblasen von Unten ermöglicht die Messung der Blasen, die sich in der Nähe von der Wasseroberfläche befinden.
Zur Auswertung wurden zwei Algorithmen entwickelt. Der erste basiert sich auf einen Deep-Learning Ansatz und eignet sich gut für Messungen mit kleinen Blasenkonzentrationen, während der zweite klassische Machine Learning- und Bildverarbeitungsmethoden zur Blasendetektion bei hohen Blasenkonzentrationen verwendet. Die Lokaliesierung der Blasen in der dritten Dimension konnte durch die Kalibrierung der Tiefenschärfe bestimmt werden.

Contents

1 Introduction

Air-water gas exchange is important in many contexts in nature and engineering. Climate change is probably one of the most prevailing topics involving air-water exchange between the atmosphere and ocean [NIW, 2016].

The transfer rate of most gases between the atmosphere and ocean is controlled by processes just beneath the water surface. Wind blowing over the sea is the main driving force for all relevant physical processes. When the water surface is highly turbulent, gases can be more rapidly transferred toward or away from the surface. In particular, turbulent regions can generate breaking waves which in turn create bubbles that trap air from the atmosphere into the ocean [Terray et al., 1996]. These bubbles enlarge the air-water interface due to their additional surface under water. However, this does not necessarily mean that gas exchanges will occur for the whole bubble lifetime. In fact, smaller bubbles can reach equilibrium with the surrounding water and stop exchanging gases [Mischler, 2014]. Therefore, determining bubble radii is crucial to properly model bubble induced gas exchange.

Previously developed bubble measuring methods in wind wave facilities include light scattering based methods [Jähne et al., 1984], depth from focus methods [gei, 1995] and more recently, light field methods [Mischler, 2014]. Other in situ methods such as Al-Lashi et al. [2016a] and Leifer et al. [2003] introduced instruments for imaging bubbles within breaking waves. These methods rely on thresholding and edge detection algorithms to produce binary images. Further techniques such as Hough transoform [Duda and Hart, 1972] are applied to extract bubble distributions. Many of these methods, however, are very sensitive to background illumination, which either reduces their accuracy [Al-Lashi et al., 2016b] or demands more manual preprocessing.

The goal of this work is to develop a measurement technique that determines the bubble spectrum in a given measurement volume, i.e. determine bubble concentrations as a function of their radii in a well determined volume. In particular, this work uses recent developments in object detection algorithms, while relying on machine learning for bubble classification in order to develop an accurate and robust bubble measurement technique.

In this work, first the relevant theoretical basics are discussed in chapter 2 starting with the physics behind capturing bubble images, followed by basics in image processing. Next, discuss related measurement techniques are briefly discussed in chapter 3 and their importance to this work. Chapter 4 describes the experimental setup in detail. The proposed algorithm that analyses bubble images is described thoroughly in chapter 5.

2 Theory

This chapter explains the theoretical concepts relevant to this thesis. First, the physics behind the developed method is explained in section 2.1, in particular how bubbles interact with light. Next, the mathematical basics necessary for image processing such as Fourier theory and edge detection are discussed in section 2.2. Section 2.3.1 explains the principle behind machine learning that the method developed in this work relies on for classification. Finally, section 2.3 formally introduces the object detection problem and the chosen criteria for evaluation.

2.1 Bubble physics

2.1.1 Reflection and Refraction

Reflection is the abrupt change in the direction of propagation of a light beam that strikes the boundary between two different media. Assuming the incoming light ray makes an angle θ_1 with the normal of a plane tangent to the boundary, then the reflected ray makes an angle θ_1' with this normal and lies in the same plane as the incident ray and the normal. The reflection law is shown in figure 2.1a and is described as

$$\theta = \theta' \tag{2.1}$$

Refraction is the change in direction of propagation of a wave when the wave passes from one medium into another with a different index of refraction. The angle of the reflected beam is shown in figure 2.1b is given by Snell's law:

$$n_1 \sin(\theta_1) = n_2 \sin(\theta_2) \tag{2.2}$$

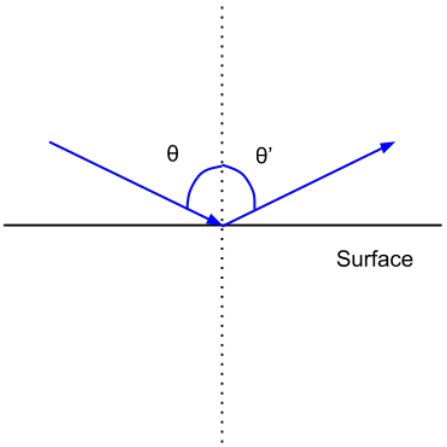
Where n_1 and n_2 are the indices of refraction of the media.

Total reflection occurs when angle of the incident beam is larger than the critical angle:

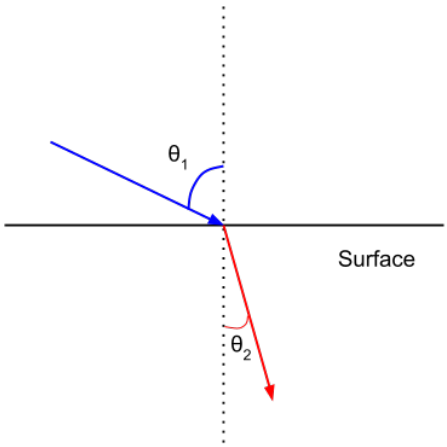
$$\theta_c = \arcsin\left(\frac{n_2}{n_1}\right) \tag{2.3}$$

2.1.2 Bubble-light interaction

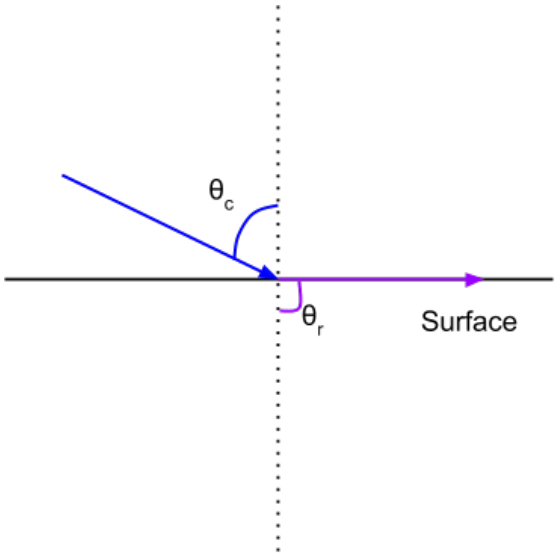
Since the smallest bubble radius (around 50-100 μm) is much larger than the wavelength of the light source (around 600 nm), diffraction within the air bubble can be neglected. Mie scattering effects are also only relevant for radii within the same



(a) Reflection: incident light beam gets reflected with the same angle relative to surface normal.

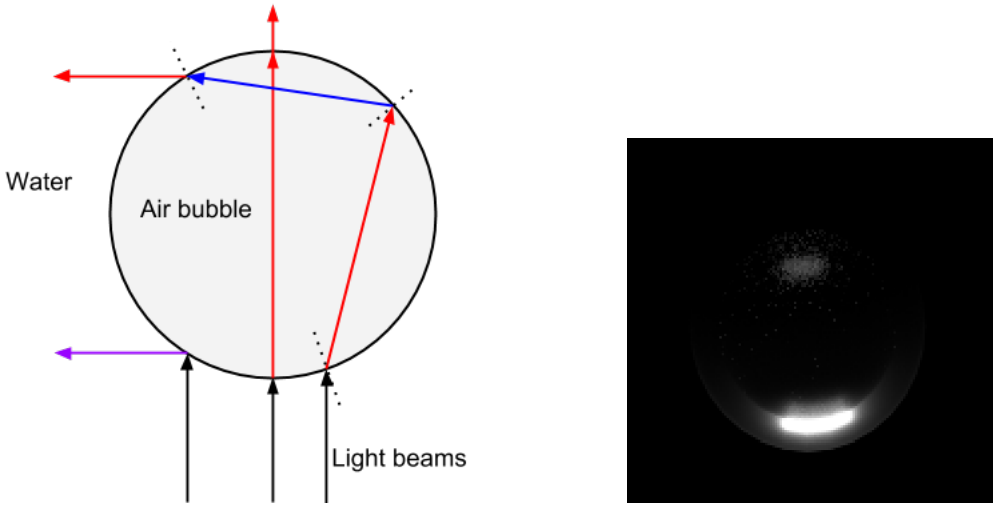


(b) Refraction: refracted light beam gets refracted according to Snell's law.



(c) Critical angle is reached when refracted angle has 90° to the surface normal.

Figure 2.1: Reflection and refraction laws.



(a) Light beams inside a bubble. Reflected beams' intensities inside the bubble are described by Fresnel's law.

(b) Simulated bubble with ray tracing.

Figure 2.2: Interaction of a light bubble with light rays

order of magnitude as the wavelength [Demtröder, 2013] so it can be neglected as well. Therefore, only refraction and reflection laws will be considered.

Figure 2.2a shows refracted rays inside an air bubble. Since the outside medium (water) is more dense, total refraction occurs at the lower bubble boundary. Simulation in figure 2.2b also shows that when a bubble is lit from bellow, two peaks can be observed. The lower peak is strong because it arises from total reflection on the lower bubble boundary, whereas the upper peak is much weaker. The second peak arises from internal (partial) reflections within the bubble and can be best explained by the Fresnel equations [Demtröder, 2013]. The different peaks characteristics will be exploited by the proposed algorithm in order to compute the bubble's depth and radius.

2.2 Image processing

In the following an image is represented as a two dimensional signal written as a matrix $\boldsymbol{g}$. $g_{m,n}$ 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 2.3.

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

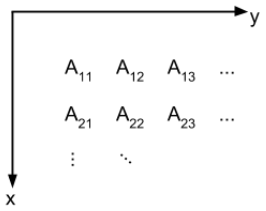


Figure 2.3: Notation and coordinate system

particular frequency contained in the spatial domain image. The *continuous* two-dimensional Fourier transform is defined as

$$\mathscr{F}\{g(\mathbf{x})\} = \hat{g}(\mathbf{k}) = \int_{-\infty}^{\infty} g(\mathbf{x}) \exp\left(-2\pi \mathrm{i}\mathbf{k}^T\mathbf{x}\right) \mathrm{d}\mathbf{x} \tag{2.4}$$

and the inverse Fourier transform

$$\mathscr{F}^{-1}\{\hat{g}(\mathbf{k})\} = g(\mathbf{x}) = \int_{-\infty}^{\infty} \hat{g}(\mathbf{k}) \exp\left(-2\pi \mathrm{i}\mathbf{k}^T\mathbf{x}\right) \mathrm{d}\mathbf{x} \tag{2.5}$$

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

Images however are discrete two dimensional signals, therefore the *Discrete* Fourier transform or DFT is needed, which is defined as

$$\mathrm{DFT}\{g_{m,n}\} = \hat{g}_{u,v} = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} g_{m,n} \exp\left(-\frac{2\pi \mathrm{i} m u}{M}\right) \exp\left(-\frac{2\pi \mathrm{i} n u}{N}\right) \tag{2.6}$$

Similarly, the inverse 2-D DFT is defined as

$$\mathrm{IDFT}\{\hat{g}_{u,v}\} = g_{m,n} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \hat{g}_{u,v} \exp\left(\frac{2\pi \mathrm{i} m u}{M}\right) \exp\left(\frac{2\pi \mathrm{i} n u}{N}\right) \tag{2.7}$$

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, the continuous convolution is defined as follows

$$(g \star h)(\mathbf{x}) = \int_{-\infty}^{\infty} h(\mathbf{x}') g(\mathbf{x} - \mathbf{x}') \mathrm{d}\mathbf{x} \tag{2.8}$$

and the discrete two dimensional convolution as

$$g'_{m,n} = \sum_{m'=0}^{M-1} \sum_{n'=0}^{N-1} h_{m',n'} g_{m-m',n-n'} \tag{2.9}$$

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

$$\mathscr{F}\{g \star h\} = NM\hat{h}\hat{g} \tag{2.10}$$

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

Edges of images are typically extended with zero values (i.e. zero padding). This introduces an error when applying filters at the image border so borders will mostly be excluded 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: $\hat{h}(\mathbf{k}) = \hat{h}(|\mathbf{k}|)$

In this work, Gaussian filters will be used 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

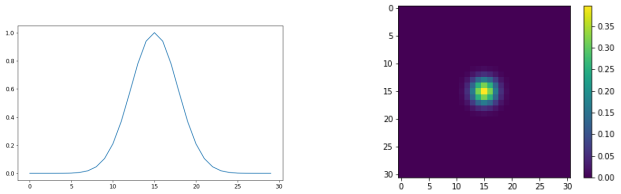
$$G_m = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(m-\mu)^2}{2\sigma^2}\right) \tag{2.11}$$

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

Figure 2.4 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.

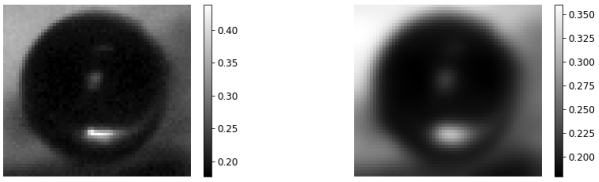
2.2.4 Edges and Derivation

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.



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

(b) 2D Gaussian signal with $\mu_x = \mu_y = 15$ and $\sigma_x = \sigma_y = 4$



(c) Original image

(d) After convolution with 2D Gaussian mask

Figure 2.4: Gaussian smoothing filter

(a) Original bubble image

(b) Sobel filter filter applied.

Figure 2.5: Computing image derivative. Edges are detected by applying a derivative filter.

In continuous space, a partial derivative operation is defined as

$$\nabla = \left[\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right] \tag{2.12}$$

and its corresponding Fourier transform is

$$\mathscr{F}\{\nabla\} = 2\pi \mathrm{i}\mathbf{k} \tag{2.13}$$

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

$$\mathbf{H} = \begin{bmatrix} \dfrac{\partial^2}{\partial x^2} & \dfrac{\partial^2}{\partial x \partial y} \\ \dfrac{\partial^2}{\partial y \partial x} & \dfrac{\partial^2}{\partial y^2} \end{bmatrix} \tag{2.14}$$

and its Fourier transform is

$$\mathscr{F}\{\mathbf{H}\} = -4\pi^2 \mathbf{k}\mathbf{k}^T \tag{2.15}$$

Edge detectors can be implemented as filters h that operate on a two dimensional grid.

In this work, the Sobel filter masks as defined in equation (2.16) will be used to compute derivatives in x and y directions.

$$S_x = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}, \qquad S_y = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & 2 \\ 1 & 0 & -1 \end{bmatrix} \tag{2.16}$$

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

$$S = \sqrt{(S_x \star G)^2 + (S_y \star G)^2} \tag{2.17}$$

where G is the input image. Figure 2.5 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 2.6 shows that in a neighborhood with ideal orientation gray values change in one direction only. Generally, the direction of local orientation

Figure 2.6: Ideal local neighborhood described by a unit vector $\tilde{\mathbf{n}}$

Condition	**Rank(J)**	**Description**
$\lambda_1 = \lambda_2 = 0$	0	Local neighborhood has constant values
$\lambda_1 \geq 0, \lambda_2 = 0$	1	The local neighborhood is a simple neighborhood with ideal orientation.
$\lambda_1 \geq 0, \lambda_2 \geq 0$	2	Gray values change in all directions

Table 2.1: Interpretation of the eigenvalues of a structure tensor

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

$$g(\mathbf{x}) = g(\mathbf{x}^T \tilde{\mathbf{n}}) \tag{2.18}$$

From this principle the structure tensor can be derived[**?**]

$$\mathbf{J} = \int w(\mathbf{x} - \mathbf{x}') \left(\nabla g(\mathbf{x}') \nabla g(\mathbf{x}')^T \right) \mathrm{d}\mathbf{x}' \tag{2.19}$$

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

$$J_{pq} = \int_{-\infty}^{\infty} w(\mathbf{x} - \mathbf{x}') \left(\frac{\partial g(\mathbf{x}')}{\partial x'_p} \frac{\partial g(\mathbf{x}')}{\partial x'_q} \right) \mathrm{d}\mathbf{x}' \tag{2.20}$$

Orientation can be obtained from

$$\tan(2\theta) = \frac{2J_{12}}{J_{11} - J_{22}} \tag{2.21}$$

The importance of the structure tensor stems from the fact the eigenvalues (which can be ordered $\lambda_1 \geq \lambda_2 \geq 0$ and the corresponding eigenvectors summarize the distribution of the gradient within the window defined by w. Table 2.1 shows how eigenvalues can be interpreted.

For discrete images, Sobel filters as defined in equation 2.16 are used for derivation. As for smoothing, a Gaussian smoothing mask as introduced in section 2.2.3 is used. Computing the elements of a structure tensor for an image G therefore requires following steps:

1. $G_x = S_x \star G$
   $G_y = S_y \star G$

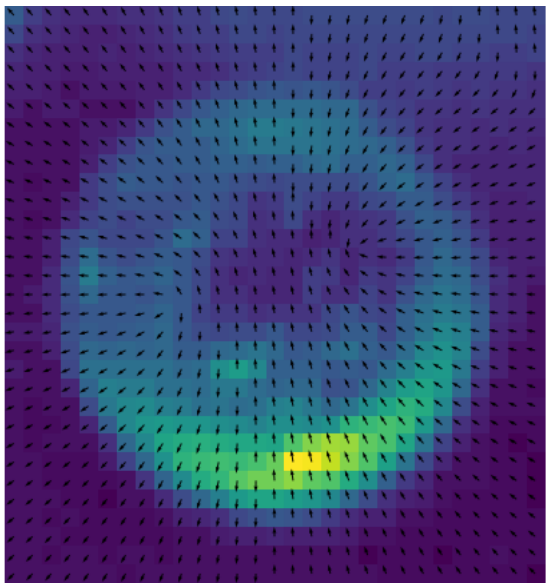


Figure 2.7: Orientation angle using sobel filters for derivation and a Gaussian mask with $\sigma = 1$ for smoothing.

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

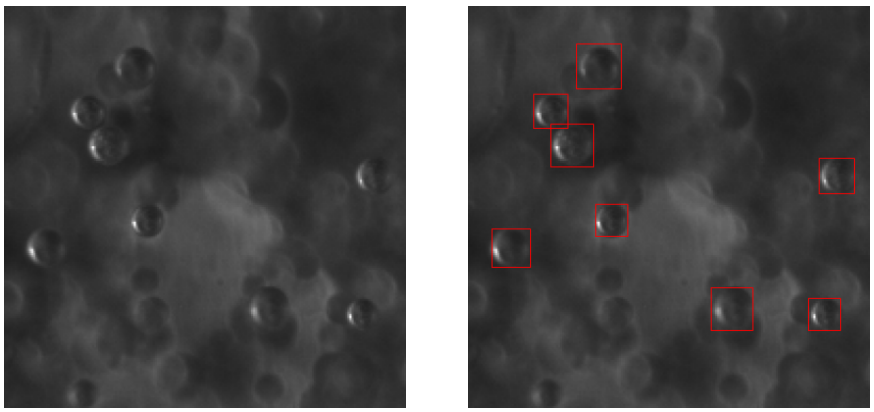
Where M is a smoothing mask and S_x and S_y are derivation masks in x and y directions respectively.

Figure 2.7 shows an extracted orientation from a bubble image using the structure tensor.

2.3 The object detection problem

The proposed algorithm recognizes bubbles (classification) in an image and estimates their respective centers and radii (localization). This problem of classification and localization is known as the object detection problem. Figure 2.8 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 [Ren et al., 2015], where many candidate regions are first extracted and then classified and single evaluation methods such as YOLO [Redmon et al., 2016], 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 this bubble detection problem yields very bad results (see chapter 5). In this work, a machine learning based approach is used for the classification part, whereas localization is performed with classical image processing methods.



(a) Input image. (b) Output format.

Figure 2.8: Output of an object detection algorithm. Bounding boxes around correctly classified bubbles are returned.

2.3.1 Classification

In this work, machine learning techniques are used mainly for signal classification purposes. For instance, both of the proposed algorithms rely on binary classification using a random forest classifier and a convolutional neural network. This section explains 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 is discussed in chapter 5.

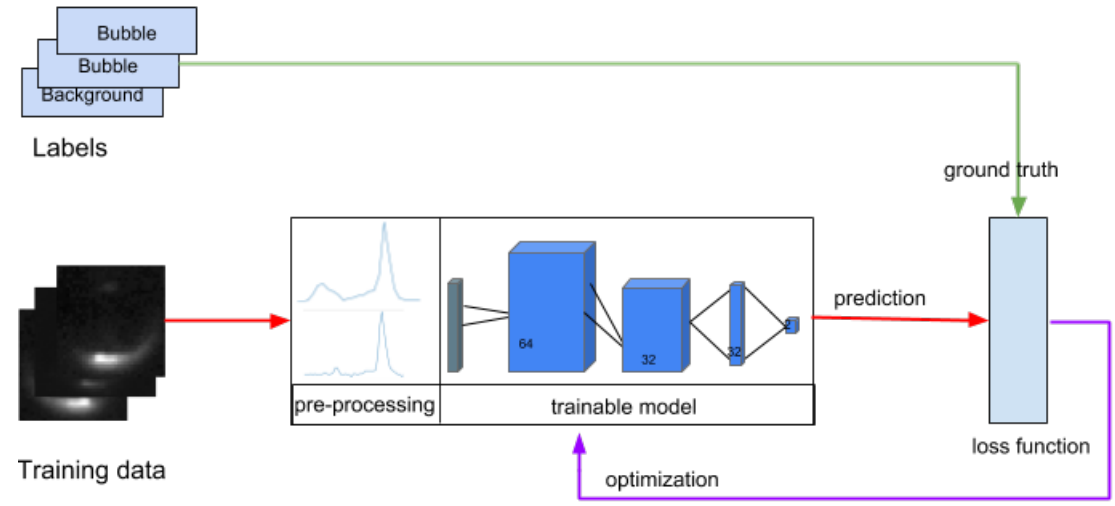
Machine learning is commonly defined as the practice of using algorithms to parse data, learn from it, and then make a determination or prediction about something. In this case, this means training the model with a large number of bubble and background instances, in order to predict whether a given signal corresponds to a bubble or not. Figure 2.9 summarizes the principle behind machine learning.

Terminology

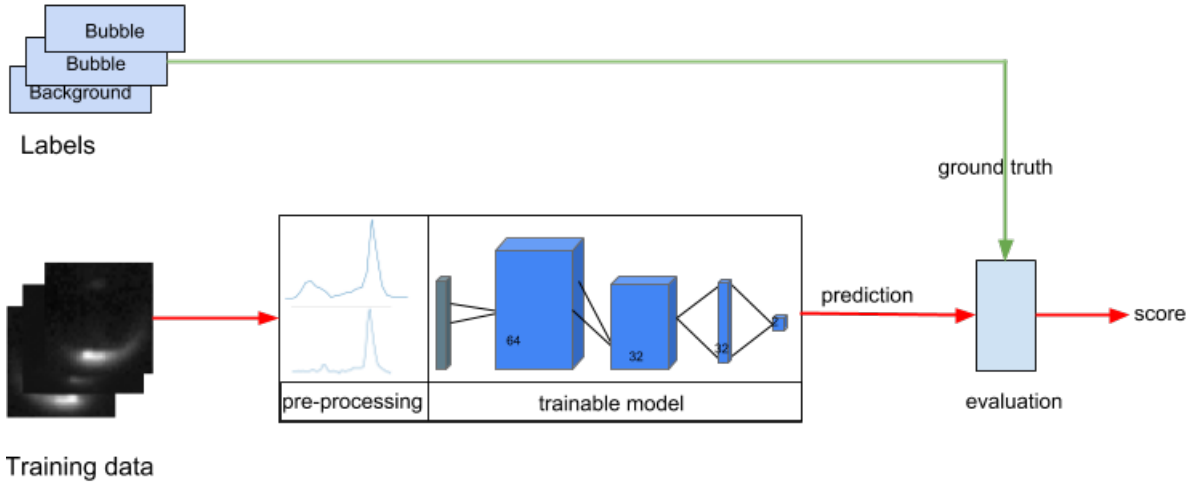
- **Training** is changing the model's parameters based on the model's output. Training is also referred to as optimization. When training data is annotated, the process of training is also called supervised training.

- **Testing** is assigning a score to a trained model based on annotated testing data. The score is typically produced 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 one wants 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



(a) Training



(b) Testing

Figure 2.9: Training and testing in machine learning

is equivalent to training by definition, an independent data set is needed to compute the final score of the algorithm, i.e. validation data.

Logistic Regression Classifier

Logistic regression is despite its name a linear model for classification, where the probabilities describing the possible outcome of a single trial are modeled using a logistic function (sigmoid curve):

$$lr(x) = \frac{A}{1 + e^{-k(x-x_0)}} \tag{2.22}$$

where x_0 is the x-value of the sigmoid's midpoint, A is the curve's maximum value and k is the steepness of the curve. This model is very simple and can be trained with relatively few data points. Its main drawback is its inability to generalize to complex data with high number of features. This classifier is used as a preprocessing step to generate training data for the more demanding convolutional neural network.

Random Forest Classifier

Random forest classifiers operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes [Ho, 1995]. So random forest is an ensemble method in which a classifier is constructed by combining several different independent base classifiers. Their main advantage over a simple decision tree is the ability to describe the data well without overfitting. This method is well suited for small sized and balanced data set, given the chosen features describe the data well. It also allows an estimation of feature importance in the classification, something that neural networks are known to be lacking. Its main drawback is its tendency to generate deep trees during training, which makes evaluation computationally expensive. See (cite Pattern Recognition buch 2016) for a more in depth description of random forests.

Convolutional Neural Network

First introduced by [Krizhevsky et al., 2017], a convolutional neural networks, or CNN is a class of deep, feed-forward artificial neural networks. Convolutional networks were inspired by biological processes in that the connectivity pattern between neurons resembles the organization of the animal visual cortex. Individual neurons activate only when stimulated in a restricted region of the visual field known as the receptive field. The receptive fields of different neurons partially overlap such that they cover the entire visual field. CNNs are known for their ability to generalize well to large, mutlidimensional data. They also require very little preprocessing. For instance, neural networks learn the filters needed to extract features using back propagation (i.e. stochastic gradient descent) to optimize convolutional and dense

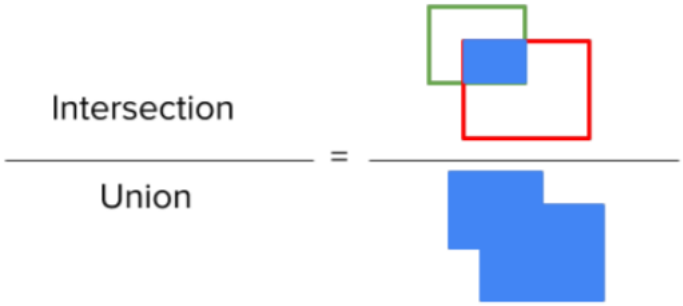


Figure 2.10: Definition of intersection over union. Green represents predicted bounding box and red is the ground truth. Areas are drawn in blue

layers within the neural network. This eliminates the need to first compute features and then pass them to the classifier as is done by classical machine learning algorithms such as random forests.

The drawback of using a CNN is the need of a large amount of data and computational power for proper training. It is also not possible to extract feature importance from a trained CNN, which makes CNNs resemble black boxes that are hard to interpret. Goldberg and Hirst [2017] explain CNNs in more details.

2.3.2 Evaluation Criteria: mAP@p-IoU

The object detection algorithm using the *intersection over union at p mean average precision* criteria as explained in figure 2.10. As the name suggests, the area of intersection between the predicted and the ground truth bounding boxes is computed and then divided by the 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 a threshold p is chosen such that IoU $>= p$ corresponds to a correct prediction or *true positive*.

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

$$precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{2.23}$$

$$recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{2.24}$$

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 the proposed algorithms perform binary classification, *mean average precision* is equivalent to *average precision*.

Note: This criteria only shows how well the algorithm 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

This chapter briefly discusses two similar bubble measurement techniques that aim to estimate bubble concentrations in water. The first is a master's thesis by [Flotow, 2017], where a bright field method was used in a wind wave facility, so that bubbles appeared as dark circles on camera, and the second is a field measurement technique [Al-Lashi et al., 2016a], where bubbles were artificially lit from below, yielding similar images to our own.

3.1 Bright field method

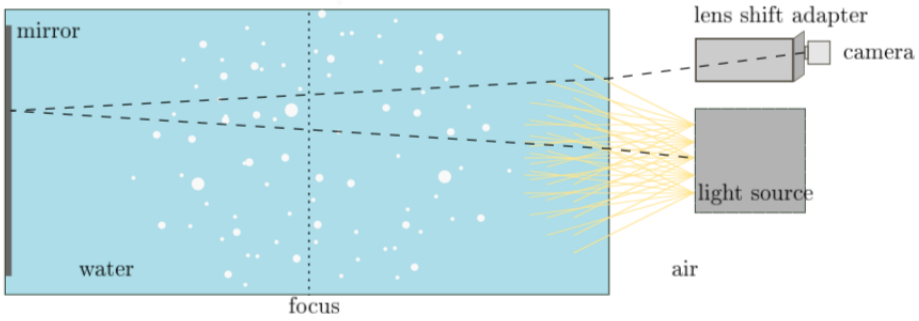
As an improved method from Mischler [2014], this technique uses a bright field method as shown in figure 3.1. Bubbles in this experimental setup are backlit, so that they appear completely dark except for one spot in the center (see section 2.1). Note the need to use a lens shift adapter because the 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).

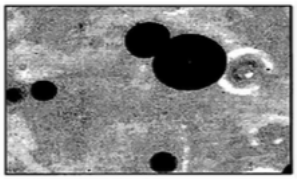
The images obtained by this method are arguably the most important advantage of this technique, since it reduces bubbles to simple circles, making them relatively easy to detect, whereas our algorithms need to recognize significantly more complex patterns than mere circles. In contrast to our method however, the bright field technique constrains how close the camera can get to the water surface. Also, Flotow [2017] 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 our newly developed method because it offered an estimation of 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

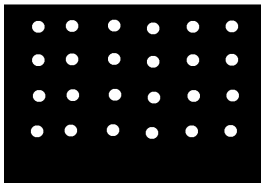
This technique submerges an optical imaging instrument (figure 3.2a) underwater and illuminates a thin slice of water of volume 4cm $\times$ 4cm $\times$ 5mm. This method is particularly interesting because it produces images similar to our own. However,



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



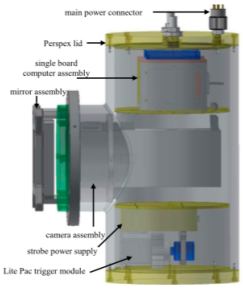
(b) Sample result image



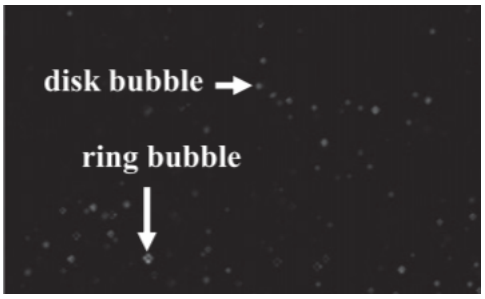
(c) calibration target

Figure 3.1: Bright field method

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 [Duda and Hart, 1972] [Al-Lashi et al., 2016a].



(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

4 Experimental Setup

4.1 Requirements

In order to improve on previous works Flotow [2017], section 3), we require that the developed method in this work fulfills following criteria:

- **R1: Minimal distance to water surface**: This requirement defined the experimental setup. As shown in figures 4.1 and 4.2, lighting bubbles from below gives more freedom for the camera to be closer to the water surface. This is more apparent when the camera faces the surface with a certain angle.

- **R2: Consistent image results**: Bubbles can look very different under different lighting conditions. For instance, bubbles might appear as dark disks or bright rings (chapter 3) depending how they are lit. It is therefore required that the angle between camera and light source is always constant. This requirement guarantees a consistent pattern for bubbles across different water tanks.

- **R3: Images must contain all possibly retrievable information**. From simulation (figure 2.2b) it is known that bubbles are characterized by two peaks. Both of these – especially the upper, dimmer one – are required to be visible. This requires the camera to have a high signal-to-noise ratio and/or the light source to be very bright.

4.2 Aquarium Setup

The goal of this setup is to capture bubble images that can be used to prototype the proposed algorithm, so the experiment was built in a transparent $1m \times 1m \times 1m$ water tank as shown in figure 4.1 where most of the parameters could be changed such as lighting angle, camera and bubble concentration independently from each other. Measurement results are discussed in next chapter and image analysis is discussed in chapter 5.

In order to satisfy requirements in section 4.1, parameters were set according to table 4.1.

We chose the Basler A1920 mainly because of its high signal-to-noise ratio, and therefore fulfilling requirement R3 given enough light. Its main drawback is its relatively low frame rate at the highest possible resolution, which makes bubble tracking more difficult, which in turn makes radius calibration more difficult (see

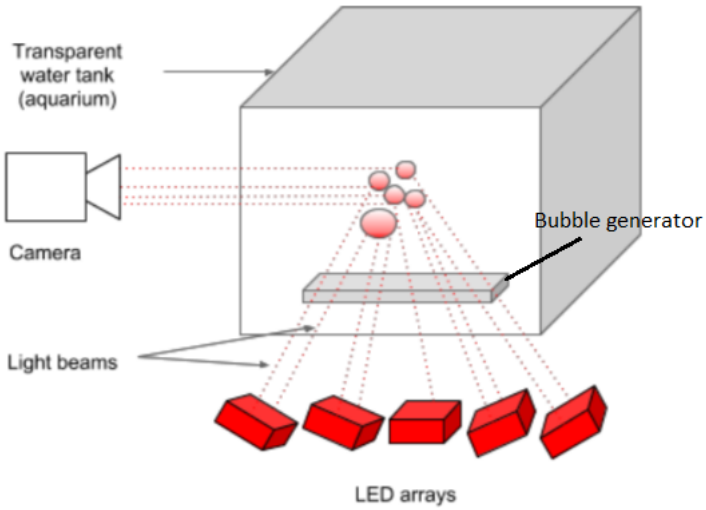


Figure 4.1: Experimental setup illustrating the 90° angle between the light source (LED array) and the camera. Note how LEDs are also oriented towards the camera's field of view to maximize the amount of light reaching the camera and therefore reduce noise.

section 5.4). Compared to an available high-speed camera with comparable maximum resolution, (PCO Dimax, 1,2kHz at 2000×2000 resolution) the chosen camera also has a smaller sensor. This means that the magnification factor is higher and therefore smaller bubbles can be better detected. A 100 mm macro lens was used in this setup. The large focal length yields a large magnification factor. Bubbles were produced by a bubble generator. Bubble radii therefore depend on the amount of air flowing through the bubble generator, measured in liter per minute or LPM. We acquired two sets of measurements, the first with a low bubble concentration, (1,8 LPM air flow), and the second with a much higher bubble concentration (3 LPM).

4.3 Aeolotron Setup

The Aeolotron is an experimental annular wind-wave facility with a diameter of about 10m and a width of about 60cm. Typical water height is 1m. Four wind engines generate reference wind speeds of up to 17m/s, enough to generate breaking waves that can produce air bubbles. Figure 4.2 shows a schematic view of the wind facility as well as the experimental setup.

The goal of this setup is to apply the knowledge gained from the previous setup on bubbles emerging from breaking waves, so the setup from the aquarium was reproduced at the Aeolotron, while keeping as many unchanged parameters as possible (e.g. same camera, same light source same lens etc...).

Also a bubble generator was used at Aeolotron in order to verify that the algorithm developed for the previous setup still works with measurements from the

Camera	Basler A1920-155um
Focal length	100 mm
F-number	5,6
Exposure time	100 us
Gamma	0,8
Resolution [px]	1920 x 1200
Sensor size	11.3 mm x 7.1 mm
Air flow	1,8 LPM / 3 LPM
Magnification factor	30 um/px
Framerate	100 Hz
Resulting field of view	5,7cm x 3,6 cm

Table 4.1: Parameters for aquarium setup

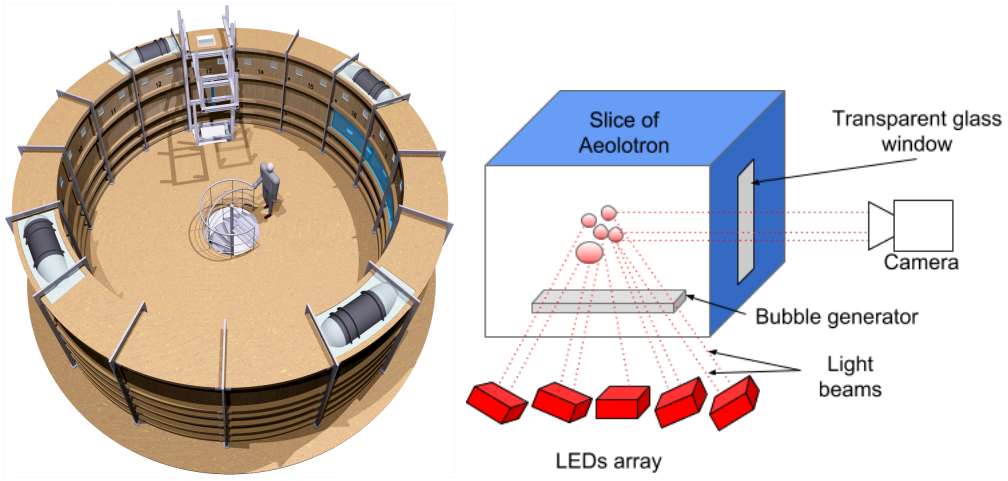
Camera	Basler A1920-155um
Focal length	100 mm
F-number	5,6
Exposure time	130 us
Gamma	0,7
Resolution [px]	1920 x 1200
Sensor size	11.3 mm x 7.1 mm
Air flow	1,8 LPM
Magnification factor	10 um/px
Framerate	100 Hz
Resulting field of view	1,9 cm x 1,2 cm

Table 4.2: Parameters for Aeolotron setup

Aeolotron setup. Some parameters such as the focal distance and the angle between the light source and bubbles were inevitably changed, which means new calibration measurements were needed. Table 4.2 summarizes the setup's parameters.

4.4 Measurement result

Figures 4.3, 4.4 and 4.5 show different images captured using a bubble generator in two different water tanks. For low bubble concentrations, images look very similar across the two different setups. However, when the concentration is very high (figure 4.4) bubbles change dramatically. The curvature can be easily distinguished and scattered light from other bubbles gives bubbles a more circular shape. Also, the background is much brighter for a high bubble concentration. Figure 4.4b shows a bubble so large that it lost its spherical shape. Such bubbles are too rare and will be classifier as background with no further considerations. The exact bubble shape will be discussed further in chapter 5. Full size, raw measurements can be found in



(a) Aeolotron wind wave facility [Krall, 2013]

(b) Experimental setup

Figure 4.2: Aeolotron facility and experimental setup

appendix.

4.5 Calibration

4.5.1 Depth of Field

Since the measurement volume needs to be determined, it is not only necessary to detect bubbles in an image, but also to determine their depth, i.e. distance to the focal plane. With the experimental setup shown in figure 4.6, bubbles can be trapped and moved along the depth axis.

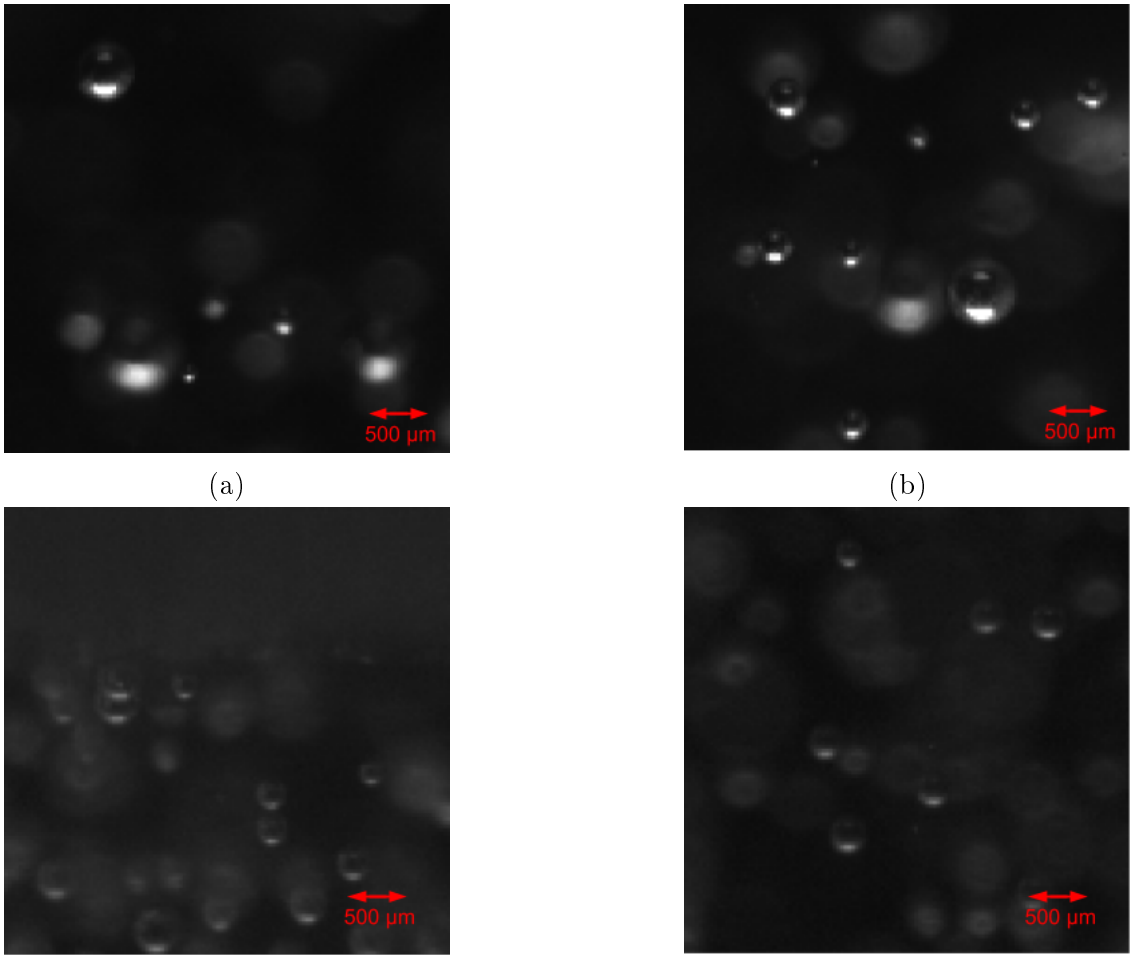
Figure 4.7 shows images captured with this setup. Note how blurriness varies among images depending on the distance to the focal plane. In section 5.4 we discuss how to extract depth form these images. The setup is not perfectly stable, so resulting images were not perfectly aligned. As a workaround, additional images were captured with a different light source at each position to make other surrounding objects visible, e.g. armature of calibration device that can later on be used for tracking and image stabilization.

This calibration technique is practical, because only relative distances to the focal plane are needed for depth calibration. This is important because the goal of calibration is to determine the measurement volume only and not necessarily the absolute position of the bubbles to the camera.

4.5.2 Radius

Although simulation shows that the distance between the two dominant peaks in a bubble is proportional to the bubble radius, the real radius cannot be determined

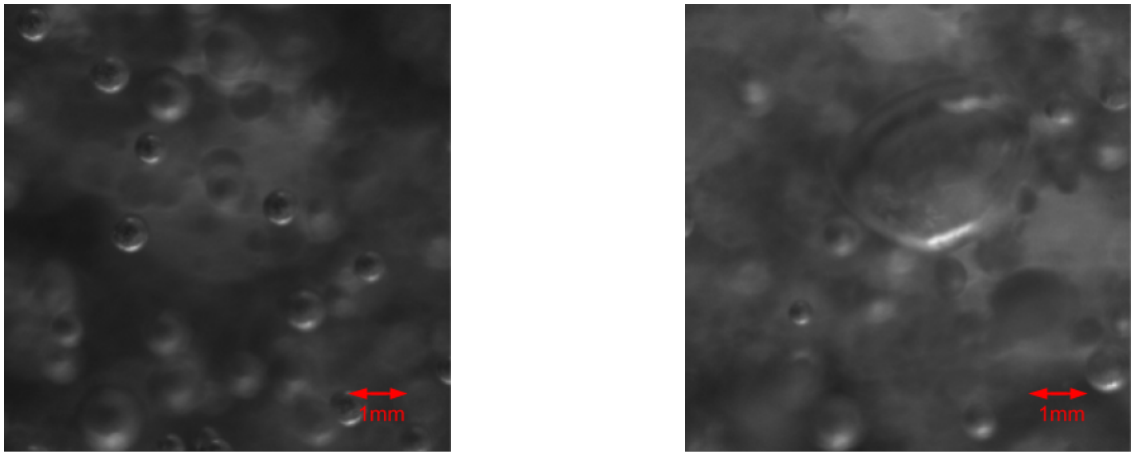
from measurement results alone (figure 4.3 and 4.5). So lighting needs to be changed in such a way that the real radius becomes measurable, i.e. the whole bubble becomes visible. In a similar way to the light field method Mischler [2014], an additional light source and a mirror were mounted in order to light up bubbles from behind (figure 4.8). Then, images were acquired at a frame rate of 200 FPS, where the second light source was turned on every second frame. Results are shown in figure 4.9. In section 5.4 bubbles lit from below are identified with those lit from behind in order to estimate the proportionality factor between the real radius and the radius computed by the proposed algorithms.



(a) (b)

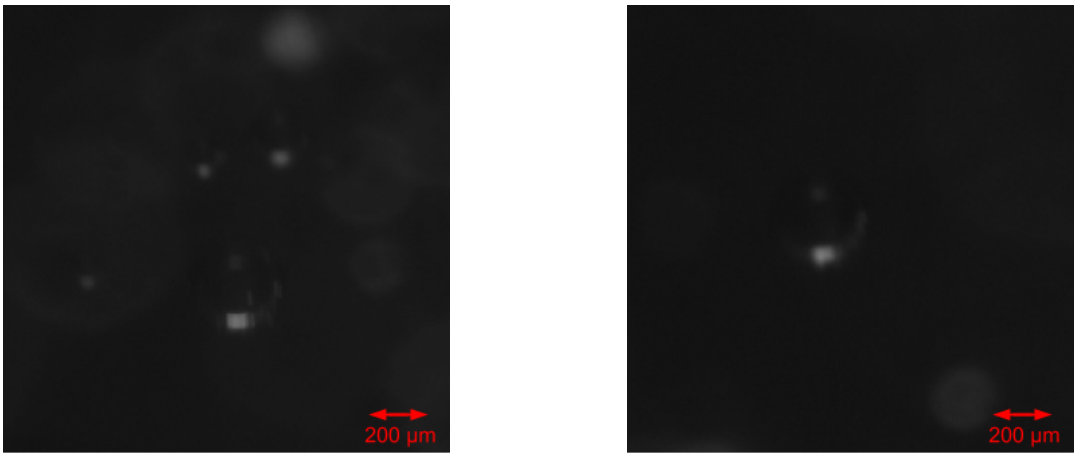
(c) (d)

Figure 4.3: Sample image captured with aquarium setup for a low bubble concentration for different camera setups. Note how bubbles are characterized by one bright peak at the bottom and one dimmer upper peak, as expected from simulation.



(a) (b)

Figure 4.4: Sample images captured with aquarium setup for a high bubble concentration. The upper peak is not visible anymore. However, bubble curvatures are more recognizable, even for smaller bubbles



(a) (b)

Figure 4.5: Image captured with Aeolotron setup. Bubbles were created by a bubble generator. Magnification is now larger than with the aquarium setup.

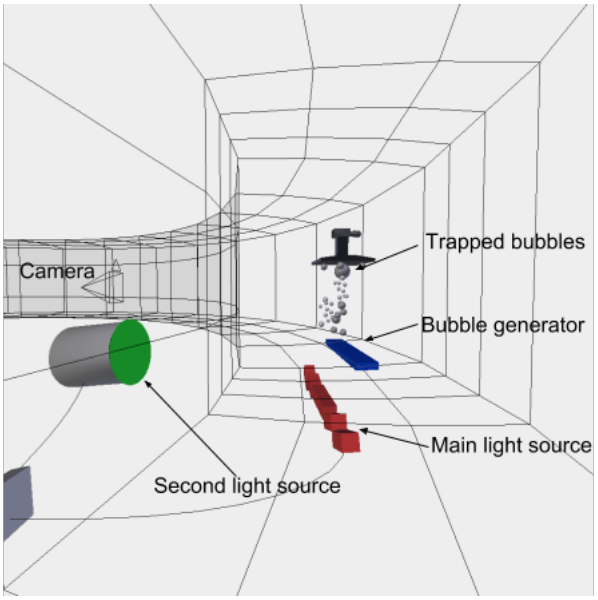
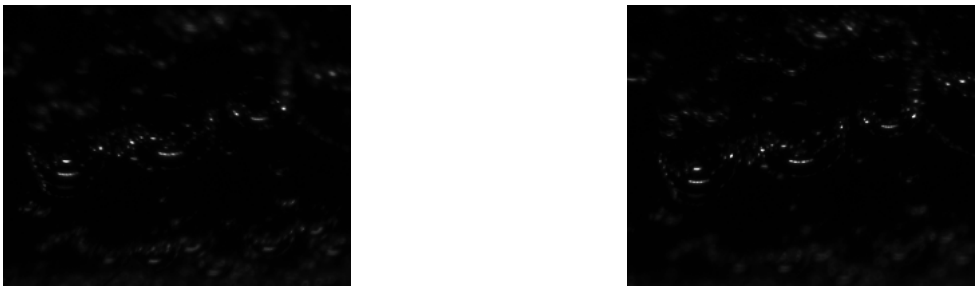


Figure 4.6: Setup for depth of field calibration. A high friction stick attached to a micrometer screw traps bubbles and moves them away from the focal plane.



(a) (b)

Figure 4.7: Two images from depth of field calibration setup at different distances from focal plane.

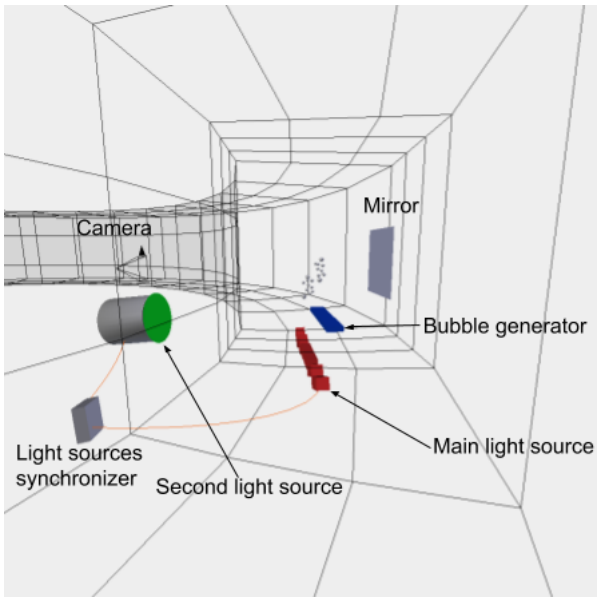
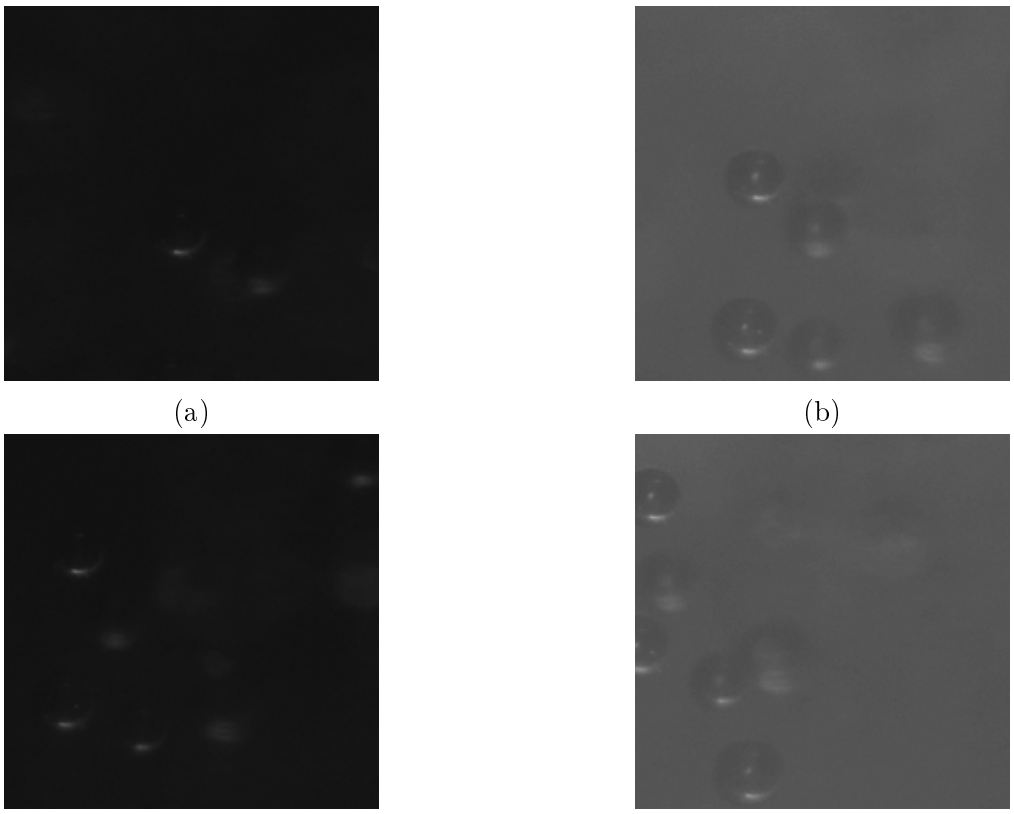


Figure 4.8: A view from inside the Aeolotron wind wave facility for radius calibration. The additional light source (green) is synchronized with the main light source (red).



(a) (b)

(c) (d)

Figure 4.9: Radius calibration images. (a) and (c) are lit from below. (b) and (d) are lit from below and from behind, where the real bubble radius becomes visible.

5 The Algorithm

This chapter discusses two important object detection algorithms that can classify bubbles in an image and estimate their radii. The first algorithm, which is referred to as *BubbleProfiles* or simply algorithm 1 is optimized to work for small bubble concentration (e.g. figure 4.3) whereas algorithm 2, also referred to as *BubbleCurves* is better suited for bubble images with very high concentrations (figure 4.4).

5.1 Motivation

Section 2.3 mentions that the state of the art object detection algorithms are based on deep learning algorithms such as Faster R-CNN [Ren et al., 2015] and YOLO [Redmon et al., 2016]. Training these algorithms with the data from this work (simulated and real) did not yield good results. Training a pre-trained R-CNN algorithm (i.e. transfer learning) is not meaningful either because most pre-trained models have large anchors (or rank regions boxes) and are fine-tuned to work for real world photographs that contain sparse objects such as humans, cars and pets.

The acquired data from this work however, is very dense. And although bubbles can have comlex pattern, these patterns are far simpler than typical objects found in photographs. This led us to the conclusion that such algorithms are not well suited for detecting small and dense objects (images with a low bubble concentration are also considered to have dense objects, since they contain several hundred bubble instances per image that are very close to each other). This suspicion was confirmed by Zhe [2018] who also added that these algorithms not only rely on a large amount of training data, but also require the data to be rich in features, which does not apply to this work's data.

Interestingly though, scaling down anchors and training an R-CNN algorithm (cite rcnn, Projektpraktikum?) from scratch did perform slightly better, with an mAP@0.5IoU of 68%. This however, is far from ideal, so developing a different approach was necessary to achieve better results.

5.2 BubbleNet

This algorithm detects bubbles in an image with low bubble concentration and estimates their radii. The algorithm extracts candidate signals from images, classifies these signals and then computes a radius from a signal. A vanilla version of the algorithm is described in algorithm 1. Note that the algorithm returns a list of squares (i.e. bounding boxes around bubbles) so that its performance can be evalu-

ated. Later on, only bubble radii, i.e. bounding boxes' widths and bubbles' depths are needed to construct a bubble spectrum.

Algorithm 1 BubbleProfiles

```
1: Input Image with low bubble concentration G.
2: Output List of squares Sq, List of depths Dep
3: Obtain local maxima loc_max in G
4: for Each lm in loc_max do
5:     Extract vertical profile signal v_p with length L.
6:     if is_bubble_BubbleNet(v_p) then
7:         Compute radius r and center c from signal using equation 5.1
8:         Compute real radius r' using radius calibration equation 5.6
9:         Compute depth dp using equation 5.5
10:        append(Sq, toBoundingBox(r', c))
11:        append(Dep, dp)
12:    end if
13: end for
```

5.2.1 Radius from Bubble

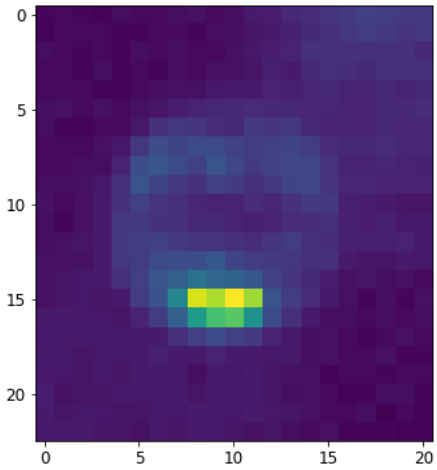
A closer inspection of bubbles shows that a bubble can be described well by a vertical cross section going through the middle of the bubble. For instance, figure 5.1 shows that a vertical profile v_p is characterized by a two salient peaks: A bright peak that emerges from total reflection as explained in figure 2.2 as well as a dimmer, yet still visible second peak at a distance d from the first peak. Furthermore, simulation shows that the distance d is proportional to the real bubble radius (figure 5.2). Note that in figure 5.2 the distance d was computed on rendered images, whereas the radius is a given parameter that is determined before rendering. The step wise increase in d can be explained by sampling, i.e. when the radius increases by an amount smaller than a pixel, d will either not change or increase by exactly 1 pixel.

In order to reach a better precision, a Gaussian fit around the two peaks is performed. Here it is necessary to initialize the fit routine with reasonable values, otherwise both fits would converge towards the same function. So the signal is first smoothed with a 1D Gaussian filter, and then the absolute maximum m_1 within the signal is determined that corresponds to the first peak. The second peak corresponds to the highest local maximum m_2 that comes right after m_1. Finally, the positions of m_1 and m_2 are used as initial parameters for a Gaussian fit around the peaks. The final result is given by

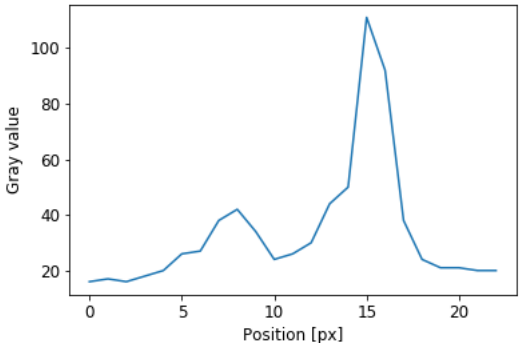
$$r_{\text{measured}} = \frac{|\mu_1 - \mu_2|}{2} \tag{5.1}$$

for the radius and

$$c_{\text{measured}} = \left(\frac{|\mu_1 + \mu_2|}{2}, v_{p,y} \right) \tag{5.2}$$



(a) Enlarged image of a bubble from a low concentration image.



(b) Vertical bubble profile. The second peak is much dimmer than the first, but still bright enough to be distinguished from background noise.

Figure 5.1: A small air bubble from an image with low bubble concentration.

for the bubble center where μ_1 and μ_2 are the resulting parameters from the Gaussian fit that correspond to the first and second peak respectively.

From previous considerations it becomes clear that a radius calibration is needed to determine the real radius from the measured radius. This is discussed in sections 4.5.2 and 5.4.2.

5.2.2 Signal extraction

So far, it has been discussed how the radius can be determined from a given vertical cross section of a bubble. However, the question remains, how do we know the signal corresponds to a bubble in the first place? Answering this question is by far more challenging than determining the bubble radius and the solution will be discussed thoroughly in the following sections.

As measurement results show (section 4.4), extracting local maxima from an image gives good starting points for vertical profile extraction. Starting from a local maximum, the signal must reach a few pixels downwards l_d to cover the whole peak and a certain distance upwards l_u until the second peak is reached. l_d and l_u are hyperparameters that add up to the total signal length L. The choice of l_u depends on the largest bubble radius that need to be detected. Al-Lashi et al. [2016a] showed that bubbles large enough to loose their spherical shape are very rare and their contribution to gas transfer can be neglected. Also, Flotow [2017] showed that

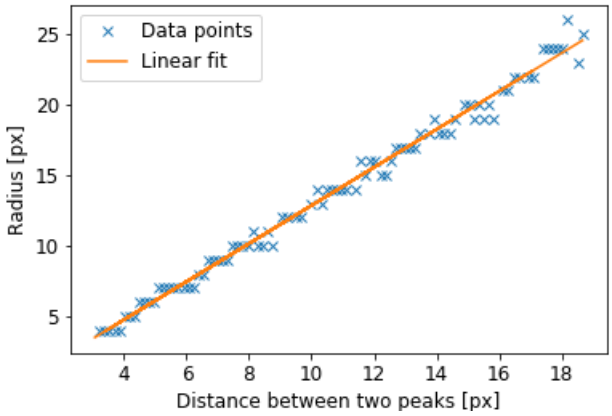


Figure 5.2: Simulated distance between two peaks as a function of bubble radius.

the maximum bubble diameter observed at the Aeolotron facility lies at around 500 μm (i.e. 50 pixels). As for l_d, a constant value of 4 pixels is good enough to cover the large peak, because the value returned by the fit usually lies within 1 to 2 pixels away from the local maximum.

5.2.3 Signal classification

After having extracted a one dimensional signal from a potential bubble, the signal is classified using a convolutional neural network called *BubbleNet*.

BubbleNet architecture

The architecture of BubbleNet, showin in figure 5.3 is inspired from Li et al. [2017], where the authors used a 1D convolutional neural network to classify electrocardiogram signals (ECG). BubbleNet contains two convolutional layers with MaxPool layers in between and two dense layers, where the last dense layer is also the output. The first convolutional layer has a relatively large receptive field. The idea behind it is to account for the large variations of the peak's widths. For the next layers, a similar pattern to Li et al. [2017] is followed and a small receptive field is chosen for the last convolutional layer, aiming to capture salient features at a higher abstraction level.

Note that the neural network has a fixed input size of 20 gray values. This means that resizing the vertical profiles is often necessary before passing them to the CNN. BubbleNet is therefore trained with resized inputs in order to make it robust against such preprocessing operations.

Since the bubble radius is not known prior to vertical profile extraction, not one but n signals with length L_s are extracted, where

$$r_{min} \leq L_s \leq L \tag{5.3}$$

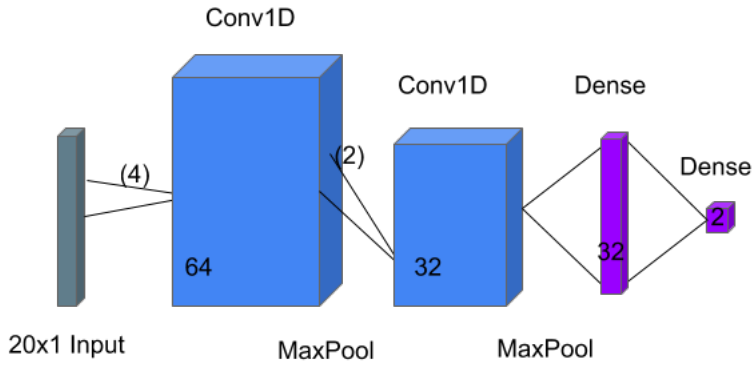


Figure 5.3: Architecture of BubbleNet. Numbers in parentheses refer to the kernel size. The rest refers to the layer size.

and then the signal with the highest bubble probability (raw classifier output) is kept. Signals that are classified as background obviously do not get further considerations. r_{min} is the smallest detectable bubble radius and it depends on the magnification factor. For the Aeolotron setup, $r_{min} = 90\mu m$ was used.

Training Data

The most straightforward approach to generate training data is to annotate images by hand, preferably with an accuracy of no more than 2 pixels. However, this can quickly become very tedious work, especially when hundreds of thousands of images are needed to train a neural network with over 10^3 parameters. Instead, training data were generated as follows:

1. Annotate 300 bubbles only by hand.
2. Train logistic regression classifier with 300 annotated and 300 simulated images using manually computed features.
3. Predict 100 000 bubble instances with logistic regression classifier for use as training data.
4. Use data augmentation to generate a total of 200 000 vertical profiles.
5. Extract 200 000 background (non bubble) images to balance data.

Step 2 and 3 are performed to generate training data. Following features have been used:

- First peak's gray value p_1
- Second peak's gray value p_2
- p_1/p_2

- Peaks distance d

The annotated images are fairly similar in size, lighting conditions and camera settings (Gamma value, gain, exposure time etc..). The reason for that, is because we want to optimize the logistic regression classifier for a high recall, regardless of its precision. So at this stage, it is only desired to detect as few false positives as possible, even if that means neglecting potential true positives, which in turn lowers the precision. This is obviously not a good approach to classify bubbles in general, however, the goal at this stage is to merely generate accurately annotated data.

After predicting 10^5 true positive bubble instances from the measurements in step step 3, images were processed further to generate more data. Processing images with the goal of generating more data is called data augmentation and it is essential to teach the network the desired invariance and robustness properties [Ronneberger et al., 2015]. Data augmentation included changing the gamma value, with the intent to make the model robust against images taken with different camera settings, resizing (upsampling and downsampling), and horizontal flipping.

Having generated more than $4 \cdot 10^5$, the neural network can now be trained.

Results

First, the data was shuffled data and split it into training (80%) and testing (20%). After training for 14 epochs, the testing accuracy converged to 96.8%, which is a very good result for a binary classifier.

For validation, 20 more manually annotated small images were used, each containing from 10 to 20 bubbles. Figure 5.4 illustrates the performance of algorithm 1. The algorithm achieved mAP@.5IoU = 91.5% and mAP@.3IoU = 93%. Note that this is slightly worse than the classification accuracy result. This is due to the occasionally unprecise radius computation, which gets penalized by the mAP@p-IoU criteria.

5.3 BubbleCurves

This algorithm detects bubbles in images with high bubble concentrations and estimates their radii. In these images, bubble curvatures are much better distinguishable than on low bubble concentrations images. In algorithm 2, first candidates are picked, classified and then their radii are determined.

5.3.1 Bubble Classification

In a similar way to algorithm 1, candidate bubbles are picked based on local maxima in the input image. Then a window with fixed size of 10×10 pixels is extracted around the local maximum. The window size is a hyperparameter and it was chosen, such that it covers enough curvature for large bubbles while keeping its size as small as possible, so that no more than one full (small) bubble is contained in this window.

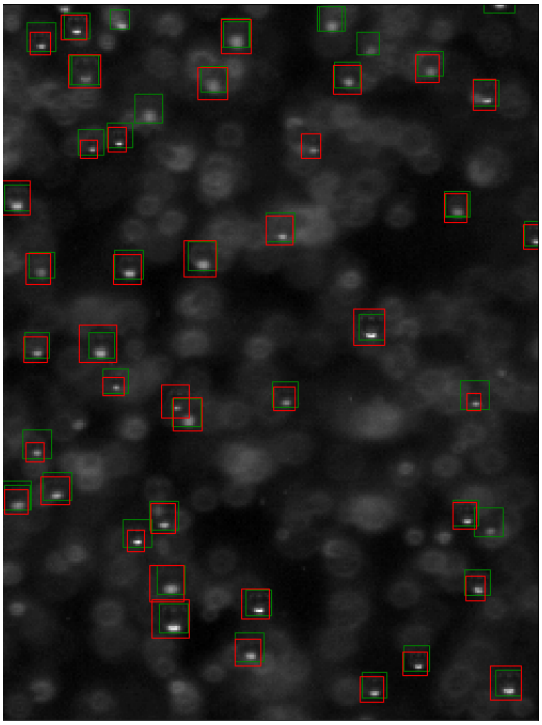


Figure 5.4: Testing of algorithm 1. Red: ground truth, green: predicted. Note how in this image, ground truth annotation was not done perfectly, so IoU results were likely slightly underestimated.

Algorithm 2 BubbleCurves

Input Image with a high bubble concentration G.
Output List of rectangles Rec, List of depths Dep
Obtain local maxima loc_max in G
for Each lm in loc_max **do**
    Extract image window W around lm with size 10×10.
    if is_bubble_RandomForest(W) **then**
        get bubble curvature cur
        Compute radius r and center c from cur using a circle fit.
        Compute real radius r' using equation 5.1
        Compute depth d using equation 5.5
        append(Rec, toBoundingBox(r', c))
        append(Dep, d)
    end if
end for

Orientation from structure tensor, the eigenvalues of the structure tensor (see section 5.3.2) as well as the eigenvalues of the Hessian matrix are used as features to classify bubbles using a random forest classifier.

The choice of these features is justified as follows. Bubbles are expected to show a strong orientation inwards around the local maximum. Therefore the orientation is computed using the structure tensor with the expectation to see a vertical orientation. The eigenvalues of the structure tensor show whether this orientation is significant or not. Finally, with the eigenvalues of the hessian matrix the curvature around the local maximum can be determined, which is expected to be concave.

As for classification, a random forest classifier classifier (section 2.3.1) because it is well suited for determining thresholds by using decision trees that are particularly robust to inclusion of irrelevant features, which is important when the window size is larger than the considered bubble.

5.3.2 Radius from Orientation

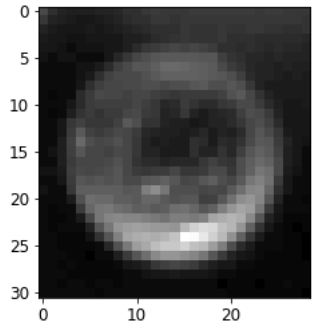
The idea behind determining the bubble radius is to start from a point on the bubble edge and then "walk" along this edge and keep track of the path. After gathering enough points along this path, a circle fit is performed to determine the bubble's radius and center.

Orientation

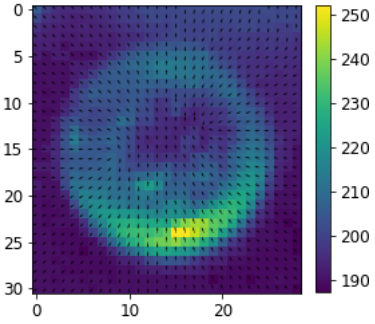
Figure 5.5 shows the result of applying the structure tensor to a single bubble. Orientation arrows describe the orientation as expected around the strongest peak in the image. Since it is only possible to compute the double angle using the structure tensor, the direction of the arrows is not important. In fact, a direction flip around the middle of the image can be observed. When considering the eigenvalues, it can be observed that both eigenvalues are close to zero around the bubble center and on the upper part of the bubble (figure 5.5c and 5.5d). This means that orientation around this area is homogeneous (ref theory struct tensor EV interpretation) so results form these area are likely not valid. This is more apparent in figure 5.6, where orientation on the upper side is different from the one around the brightest peak in the image. For these reasons, only orientation along the bubble edge up to the middle of the bubble will be used, i.e. where the larger eigenvalue is significantly larger than zero.

Curve from Orientation

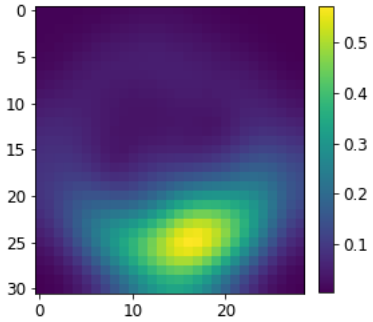
Assuming that the starting point is part of the bubble's lower edge, the orientation at the current pixel is computed and then points are sampled on a line segment along this orientation. The sampling method is taken from Bopp [2018]: A square is drawn around each point on the sampling line, and then a weighted average of the gray values within the rectangle is performed, where weights are intersection



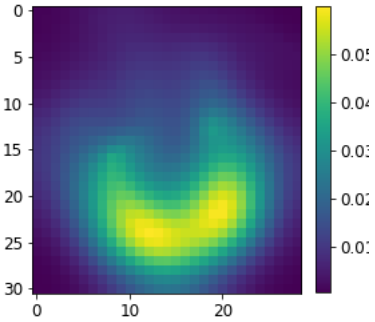
(a) Original image



(b) Orientation from structure tensor



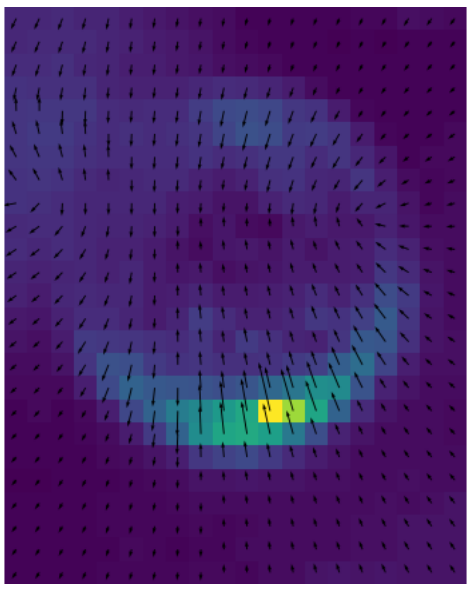
(c) Larger eigenvalues of the structure tensor



(d) Smaller eigenvalues of the structure tensor

Figure 5.5: Structure tensor eigenvalues and orientation on a single bubble from a high bubble concentration image.

Figure 5.6: Applying the structure tensor on a slightly smaller and differently lit bubble. Significance of orientation is represented by the arrows' length. Orientation from the left side and the upper part is not meaningful and can't be used.



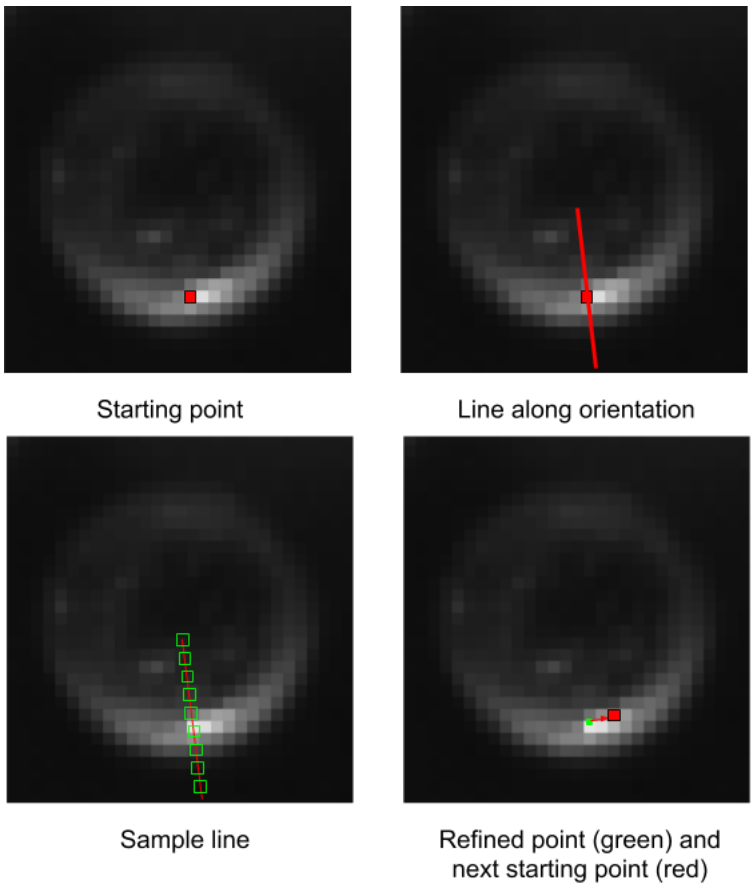


Figure 5.7: Extracting curve from bubble using orientation and neighborhood sampling.

areas between neighboring pixels and the drawn rectangle. This sampling step is illustrated in figure 5.8 After sampling, a Gaussian fit is performed along the fit line, where the mean is initialized with the current position. Next, the fit result is added to the final curve and the next starting point is chosen by moving one step to the right. Figure 5.7 summarizes how a bubble curve is extracted using orientation.

Finally, a least squares fit is used to find the center $c = (a, b)^T$ and the radius r from the circle equation

$$\sqrt{((x-a)^2 + (y-b)^2)} = r^2 \tag{5.4}$$

The fit and sampling results are shown in figure 5.9

Results

The final result is shown in figure 5.10. Although the circle matches the bubble's shape well, it is clear that the radius has been underestimated. Similarly to algorithm 1, radius calibration is addressed in section 5.4.2

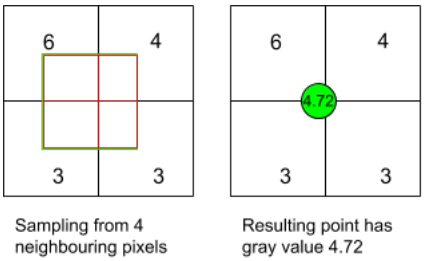
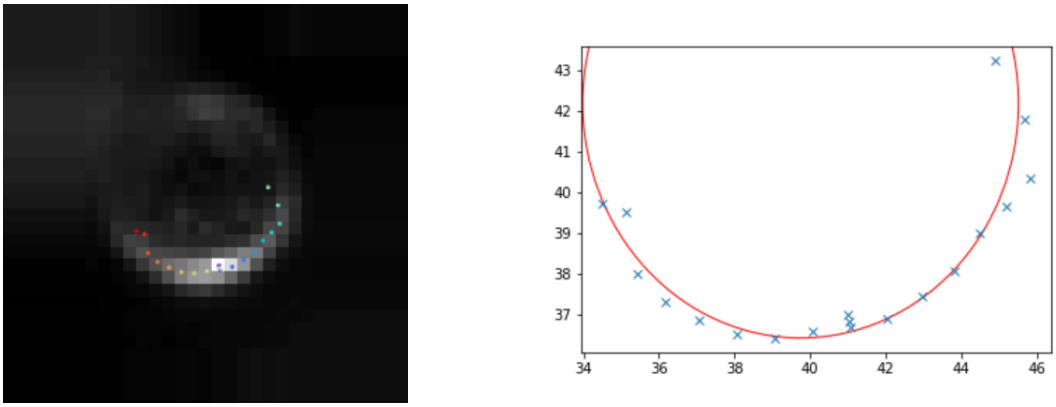


Figure 5.8: Gray value sampling along sampling line



(a) Sampling result.

(b) Circle fit result.

Figure 5.9: Sampling and fit Note that the circle works well, even though only half a circle is extracted from the bubble at most.

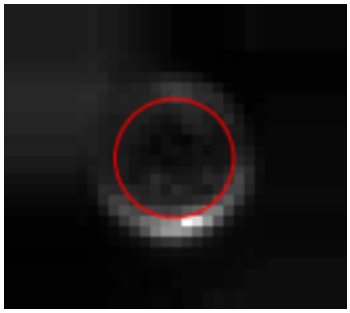


Figure 5.10: Final result of algorithm 2 applied to a single bubble

Using a validation set of 30 small images, each containing 30 to 40 bubbles, this algorithm achieved mAP@.5IoU = 89.3%, which is slightly worse than algorithm 1. This is in part due to the high rate of overlap between bubbles, making curvature extraction less precise.. Nevertheless, the algorithm is reliable enough to be usable for bubble detection in high concentration images.

5.4 Calibration

5.4.1 Depth of Field

Determining the distance between a bubble and the focal plane can be deduced from bubble blurriness. When considering a vertical profile of the bright lower peak (for both low and high bubble concentration images), it becomes clear that the peak's width increases with increasing distance from the focal plane (figure 5.11a). Therefore blurriness can be described with the peak's width, given by the sigma parameter of a Gaussian fit around the peak. Theoretically, a linear dependency between blurriness and the distance to the focal plane would be expected. However, the light source has a certain size that has to be accounted for, which gives bubble peaks a certain minimum width at focus.

Furthermore, blurriness depends on the radius. For instance, larger bubbles tend to be visible for larger distances from the focal plane (up to 10 centimeters), whereas small bubbles are only visible for no more than 2 centimeters. Figure 5.11b shows measured blurriness (in terms of σ) as a function of radius and distance from the focal plane.

Therefore, a fourth degree polynomial is chosen to describe the blurriness-depth dependency and a linear function to describe the radius dependency. The resulting calibration equation is

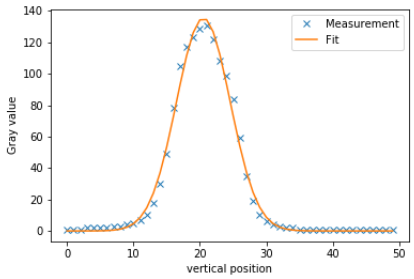
$$f(r, d) = b \cdot r + a_1 \cdot d + a_2 \cdot d^2 + c \tag{5.5}$$

Where $a_1 =, a_2 =, b =, c =$ were used for Aeolotron setup. Figure 5.11c shows the two dimensional calibration function alongside the measurements.

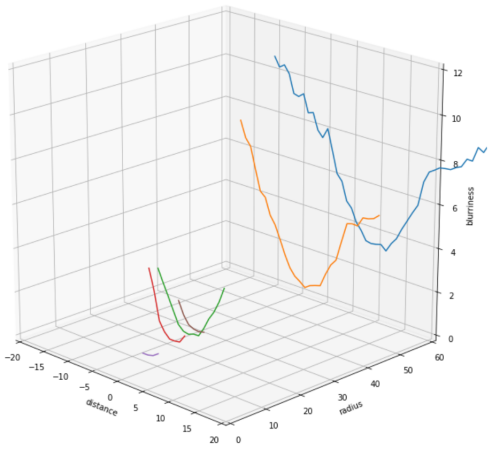
Note that only measurement results up to a certain threshold σ_{th} were kept and used for the fit function 5.5.

5.4.2 Radius

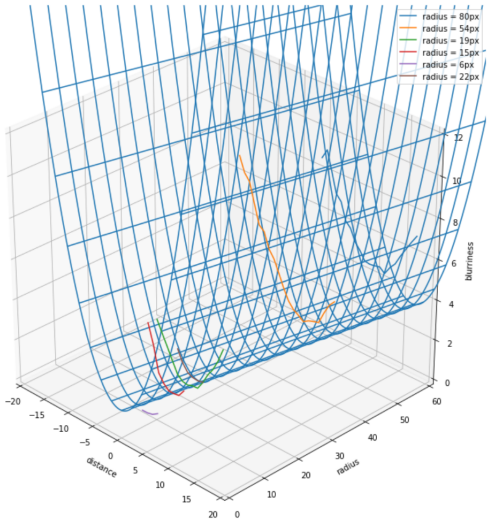
The measurement technique developed in this work uses a light source to light up bubbles from below and captures images with a camera placed at 90° to the light source. The resulting images are characterized by a lower peak (or bright lower curvature for high concentration bubbles) as discussed in section **??**. From these images alone, it is not clear whether the observed edge corresponds to the real bubble edge or not. Also, algorithm 2 clearly underestimates the observed radius (figure 5.10).



(a) Vertical peak profile and Gaussian fit with $\sigma = 4.04$



(b)



(c) Sampling result.

Figure 5.11: Depth calibration

In order to correct for this effect, calibration measurements as described in section 4.5.2 are performed.

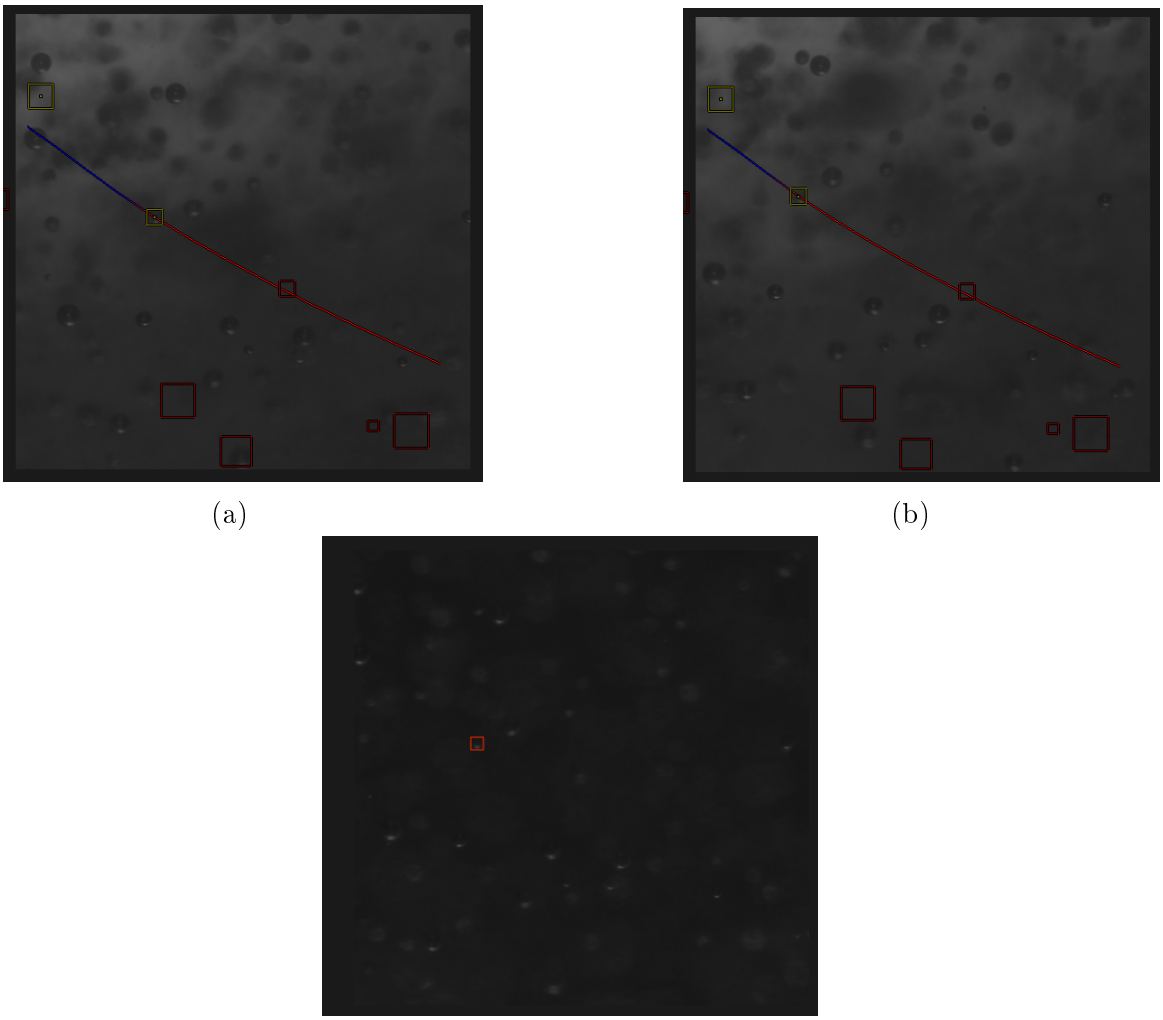
Back lit bubbles need to get identified with those lit from below. So images are acquired with back lit bubbles only, then tracking is applied to the images to track a few bubbles of different sizes and depths. The radii of the tracked bubbles is also computed for each position, i.e. from each image. From the tracked back lit bubbles, the position of the same bubble lit from below can be roughly estimated. Figure 5.12 illustrates the principle behind identifying bubbles from different images with different light sources.

Determining the radius of bubbles lit from below is done using algorithm 1. As for the corresponding back lit bubbles, the radius is determined as follows: First, the derivative image is computed and then the position of the bubble center is estimated using the characteristic bright peak around the bubble center. Then, several horizontal profiles that go through or are very close to the bubble center are extracted. Next, the local maxima from the extracted profiles is determined and a Gaussian fit around them is applied in order to determine the local maximum with a subpixel accuracy. Finally, the bubble diameter is given by the distance between the two local maxima that are farthest apart from each other. Figure 5.13 illustrates the described steps above.

Figure 5.14 shows the radius determined by algorithm 1 of bubbles lit from below as a function of the radius of back lit bubbles. The error bars are standard deviations, i.e. statistical errors arising from multiple radius computations of the same bubble at different depths and different locations. The resulting radius calibration equation is

$$r' = a \cdot r + b \tag{5.6}$$

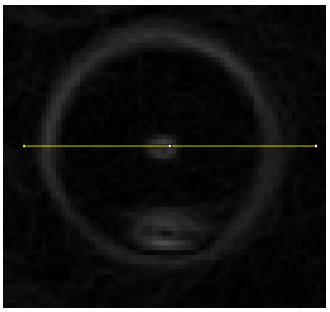
Where $a = 0.95$ and $b = 4.54$ were used for the Aeolotron setup.



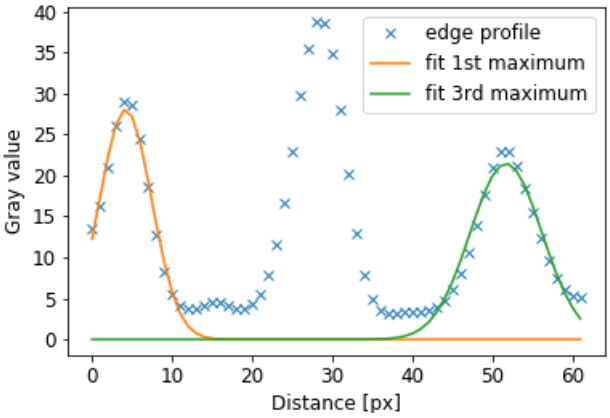
(a) (b)

(c)

Figure 5.12: Identifying a bubble from images with different lighting. Images were acquired at 200 FPS. Image 5.12c was taken after 5.12a and before 5.12b. Matching bubble in 5.12c is marked in red



(a) Horizontal profiles are extracted from the derivative image



(b) The distance between the farthest local maxima gives the diameter

Figure 5.13: Determining the radius of a back lit bubble

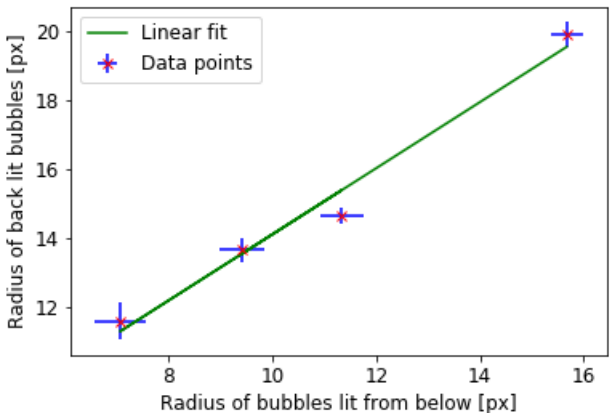


Figure 5.14: Measured radius as a function of real radius.

6 Conclusion

A bubble measuring technique has been developed with special emphasis on the ability to measure bubbles close to the water surface and developing a robust bubble detection algorithm.

Measurement setup: Artificially generated bubbles were lit from below, which resulted in images where bubbles are characterized by two peaks for a low bubble concentrations, and images where bubbles had a more apparent curvature around the bubble edge for high bubble concentrations. Keeping the angle between the camera and the light source at 90° insured a consistent pattern across the different used water tanks. The drawback of this method is that only medium to large bubbles can be detected since only total reflections from the bubble's lower edge are visible.

Algorithm 1: This algorithm uses a deep learning approach. So generating a large amount of annotated training data was necessary. This was achieved through simulation as well as through annotating measurement results with a logistic regression classifier, i.e. a simpler classifier was optimized to detect true positive bubble instances only, which were then used to train the convolutional neural network. This algorithm performed well when measured by the mAP@0.5IoU criteria. Also, the variation within the validation dataset was strong, therefore it has been verified that this algorithm is robust. Using a deep learning approach however comes with an inherent limit, which is the lack of interpretability of the model.

Algorithm 2: Images with large bubble concentrations show bubbles with a nicely recognizable curvature. This observation was used in this algorithm to classify bubbles and determine their radii. For classification, features that extract orientation and edge's concavity from a potential bubble were used. The radius is then determined from the bubble's lower half. This algorithm performed slightly worse than algorithm 1. The validation dataset was also not as varied as the one used for the previous algorithm. Therefore this algorithm is less robust against large changes in camera settings and lighting conditions. The advantage however, is that features can be easily interpretable since they were chosen manually.

Calibration: It has been shown that the width of the brightest peak at the bubble's lower edge can be used to determine the bubble's depth, i.e. location in the third dimension. As for the radius calibration, an additional light source was used to capture bubbles as dark circles, track them, compute their radii and then find a function that describes their radii with the ones computed with algorithms 1 and 2. In both calibration setups real bubbles were used, i.e. no calibration target that emulates bubbles was needed, which contributed positively to calibration accuracy.

Appendix

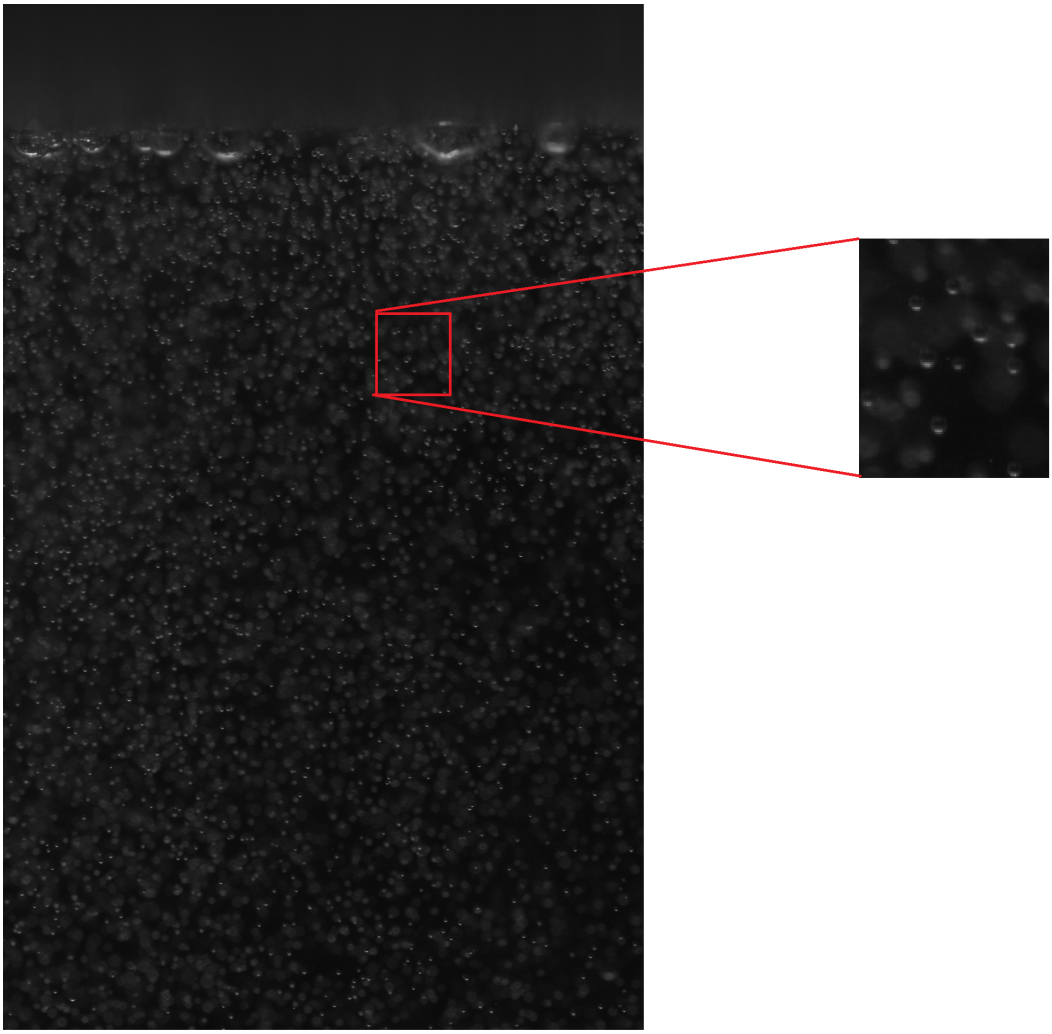


Figure 6.1: Image captured close to the water surface with the aquarium setup. Most bubbles show the expected two peaks, except for those very close to the water surface, where bubble curvatures is also visible.

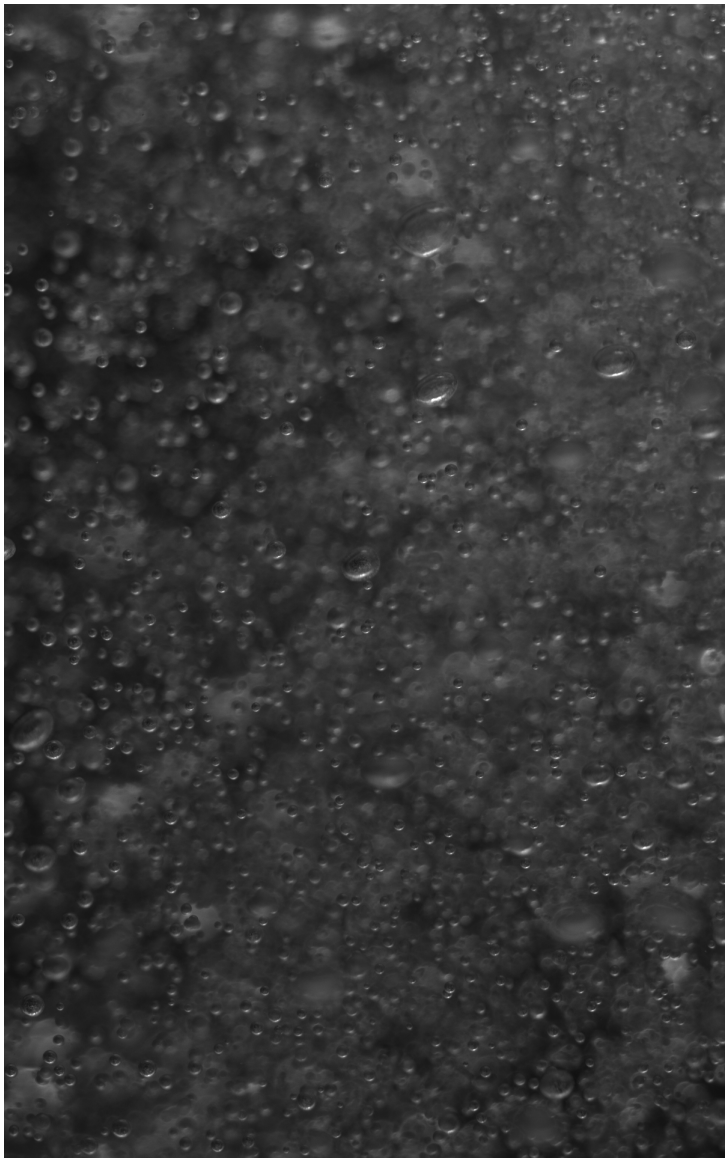


Figure 6.2: Image with high bubble concentration at aquarium setup using a bubble generator.

7 Lists

7.1 List of Figures

7.2 List of Tables

8 Bibliography

Measurements of bubble size distributions with an optical technique based on depth from focus, December 1995. Zenodo. doi: 10.5281/zenodo.10400. URL https://doi.org/10.5281/zenodo.10400.

Gas exchange and climate. *Water and Atmosphere*, 14, 2016. URL https://www.niwa.co.nz/sites/niwa.co.nz/files/import/attachments/gas.pdf.

Detecting small, densely distributed objects with filter-amplifier networks and loss boosting. *CoRR*, abs/1802.07845, 2018. URL http://arxiv.org/abs/1802.07845. Withdrawn.

Raied S. Al-Lashi, Steve R. Gunn, and Helen Czerski. Automated processing of oceanic bubble images for measuring bubble size distributions underneath breaking waves. *Journal of Atmospheric and Oceanic Technology*, 33(8):1701–1714, 2016a. doi: 10.1175/JTECH-D-15-0222.1. URL https://doi.org/10.1175/JTECH-D-15-0222.1.

Raied S. Al-Lashi, Steve R. Gunn, and Helen Czerski. Automated processing of oceanic bubble images for measuring bubble size distributions underneath breaking waves. *Journal of Atmospheric and Oceanic Technology*, 33(8):1701–1714, 2016b. doi: 10.1175/JTECH-D-15-0222.1. URL https://doi.org/10.1175/JTECH-D-15-0222.1.

Maximilian Bopp. *Air-flow and stress partitioning over wind waves in a linear wind-wave facility*. PhD thesis, Institue of Environmental Physics, Heidelberg University, 2018.

W. Demtröder. *Experimentalphysik 2: Elektrizität und Optik*. Springer-Lehrbuch. Springer Berlin Heidelberg, 2013. ISBN 9783642299438. URL https://books.google.de/books?id=lIYBlwEACAAJ.

Richard O. Duda and Peter E. Hart. Use of the hough transformation to detect lines and curves in pictures. *Commun. ACM*, 15(1):11–15, January 1972. ISSN 0001-0782. doi: 10.1145/361237.361242. URL http://doi.acm.org/10.1145/361237.361242.

Leonie Flotow. Bubble characteristics from breaking waves in fresh water and simulated seawater. Master's thesis, Institue of Environmental Physics, Heidelberg University, 2017.

Yoav Goldberg and Graeme Hirst. *Neural Network Methods in Natural Language Processing*. Morgan & Claypool Publishers, 2017. ISBN 1627052984, 9781627052986.

Tin Kam Ho. Random decision forests. In *Proceedings of the Third International Conference on Document Analysis and Recognition (Volume 1) - Volume 1*, ICDAR '95, pages 278–, Washington, DC, USA, 1995. IEEE Computer Society. ISBN 0-8186-7128-9. URL http://dl.acm.org/citation.cfm?id=844379.844681.

Bernd Jähne, Thomas Wais, and Michael Barabas. A new optical bubble measuring device; a simple model for bubble contribution to gas exchange. In W. Brutsaert and G. H. Jirka, editors, *Gas transfer at water surfaces*, pages 237–246. Reidel, Reidel, 1984. doi: 10.1007/978-94-017-1660-4_22.

Kerstin Krall. *Laboratory investigations of air-sea gas transfer under a wide range of water surface conditions*. PhD thesis, Institue of Environmental Physics, Heidelberg University, 2013.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, May 2017. ISSN 0001-0782. doi: 10.1145/3065386. URL http://doi.acm.org/10.1145/3065386.

Ira Leifer, Gerrit de Leeuw, and Leo H. Cohen. Optical measurement of bubbles: System design and application. *Journal of Atmospheric and Oceanic Technology*, 20(9):1317–1332, 2003. doi: 10.1175/1520-0426(2003)020<1317:OMOBSD>2.0.CO;2. URL https://doi.org/10.1175/1520-0426(2003)020<1317:OMOBSD>2.0.CO;2.

Dan Li, Jianxin Zhang, Qiang Zhang, and Xiaopeng Wei. Classification of ecg signals based on 1d convolution neural network. *2017 IEEE 19th International Conference on e-Health Networking, Applications and Services (Healthcom)*, pages 1–6, 2017.

Wolfgang Mischler. *Systematic Measurements of Bubble Induced Gas Exchange for Trace Gases with Low Solubilities*. PhD thesis, Heidelberg University, 2014.

J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, June 2016. doi: 10.1109/CVPR.2016.91.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 91–99. Curran Associates, Inc., 2015. URL http://papers.nips.cc/paper/

5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal-networks
pdf.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015. URL http://arxiv.org/abs/1505.04597.

Eugene Terray, M A. Donelan, Y.C. Agrawal, William Drennan, Kimmo Kahma, Albert Williams, Paul Hwang, and S A. Kitaigorodskii. Estimates of kinetic energy dissipation under breaking waves. 26:792–807, May 1996.

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 .....