TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY

MASTER THESIS (45 EC):

Bubble Segmentation by Convolutional Neural Networks

This report was made in accordance with the TU/e Code of Scientific Conduct for the Master thesis.

Author: H.B. Tuls, 1536168

Thesis Supervisors: dr. N. J. Dam
dr. N.O. Jaensson MSc
dr. T.A.M. Homan

Graduation Committee: dr. N. J. Dam
dr. N.O. Jaensson MSc
dr. T.A.M. Homan
dr. ir. K. Tiels

Master/Department: Mechanical Engineering
Research Group: Power & Flow

Eindhoven
1/1/2025

Abstract
The measure of the interfacial shape and trajectory of bubbles in dense bubbly flows has been a challenging topic in research. In the past few years, studies have focused on Convolutional Networks for this application. Previous studies have predominantly relied on the well-known Mask R-CNN. In this study, we expand the scope by exploring the application of two additional instance segmentation networks: the YOLAct++ and SOLOv2. When the contours become discontinuous or bubbles overlap to form clusters, even well-trained state-of-the-art networks struggle. Therefore, we made a proposal on how to extend such networks with ConvLSTM layers after the backbone, for continuous feature aggregation, and a strategy to reconstruct bubble shapes with a LSTM-based Graph Neural Network structure that infers continuous motion. To work towards these proposed architectures, this paper demonstrates a method of capturing bubble images by opposing stereo vision. Through a slight misalignment we show how to complete contours and estimate depths by triangulation. Apart from the YOLAct++, the CNNs show the ability to estimate whether a bubble is isolated, occluded or occluding, by which a proposed reconstruction method based on previous motion should be feasible. The SOLOv2 and Mask R-CNN show comparable capabilities in this classification task. The Mask R-CNN produces the most accurate masks.

Keywords: Bubble segmentation, Bubbly flows, CNN, Instance segmentation.

---

1 Introduction

In recent years, the field of image processing has undergone a fundamental shift in the approach of one of its major tasks, the recognition and outlining of objects in images. This task is known as image segmentation, the grouping together of pixels that belong to particular features in an image, like all pixels that belong to a human and those belonging to the background. Previously, the segmentation of images relied on manually crafted algorithms. Nowadays, due to the advent of artificial intelligence, data with known content is used to train Neural Networks (NNs) for autonomous image processing. This data-driven approach has revolutionized the field, enabling more adaptable and efficient segmentation across a wide range of applications. Driven by an abundance of accessible data and computational power, caused by the ongoing digitalization, the development of complex and large Deep Neural Networks (DNNs) has become reality. For instance, Google Images utilizes advanced DNNs to enhance image search capabilities, enabling users to find visually similar images and recognize objects within photos. ChatGPT, developed by OpenAI, demonstrates how DNNs excel in comprehending and generating natural language, facilitating nuanced and intelligent conversations. Apple's Face ID employs deep learning to securely recognize faces for authentication. Moreover, platforms like Instagram and TikTok leverage AI-driven filters and enhancements, powered by neural networks, to provide real-time, creative image and video effects. Neural networks are thriving and have established a dominant role in augmenting and surpassing conventional digital image-processing techniques.

In the field of fluid mechanics, DNNs have proved promising as well. They are increasingly employed to tackle both theoretical and practical obstacles. One of these obstacles, and the subject of this thesis, is the measure of the spatiotemporal variation of the interfacial shape and trajectory of bubbles in bubbly flows. Multiphase flows are relevant to sectors such as the chemical industry, energy sector, biomedical industry, and certain systems within the petroleum industry, as bubbly flows are integral to some of their processes. From bubble column reactors and electrolytic cells to the bubbly flows within nuclear reactors, these bubbly flows are evident in numerous processes that involve gas-liquid reactions, boiling, or cavitation. Understanding bubble behavior is crucial for enhancing efficiency and optimizing the performance of these applications.

The scientific community is interested in the shapes and progression of bubbles, as they play a significant role in understanding the fundamentals of fluid mechanics and transport phenomena within multiphase flows. Moreover, the shape of the bubbles reveals properties of the bubbly flow. Small bubbles are typically spherical, while larger bubbles can take on almost any shape, including ellipsoids, dimpled ellipsoidal caps, skirted bubbles, and more complex forms such as aspherical caps and wobbling bubbles. Several studies in the past relate these kinds of bubble shapes to domains of a combination of dimensional numbers such as the Reynolds, Eotvos, Morton, and Galilei numbers [1,2].

Bubbles are typically studied using a technique called shadowgraphy, which involves photographing the bubbly flow with a video camera. Afterward, the progression of individual bubbles can be followed by segmenting each frame. To enhance the contrast of the bubble contours, a high-speed camera is positioned opposite a light source. A diffuser is typically placed in front of the light source to achieve homogeneous illumination, as shown in the typical setup in Figure 1a. One exemplary image that is captured with this technique is shown in Figure 1b.

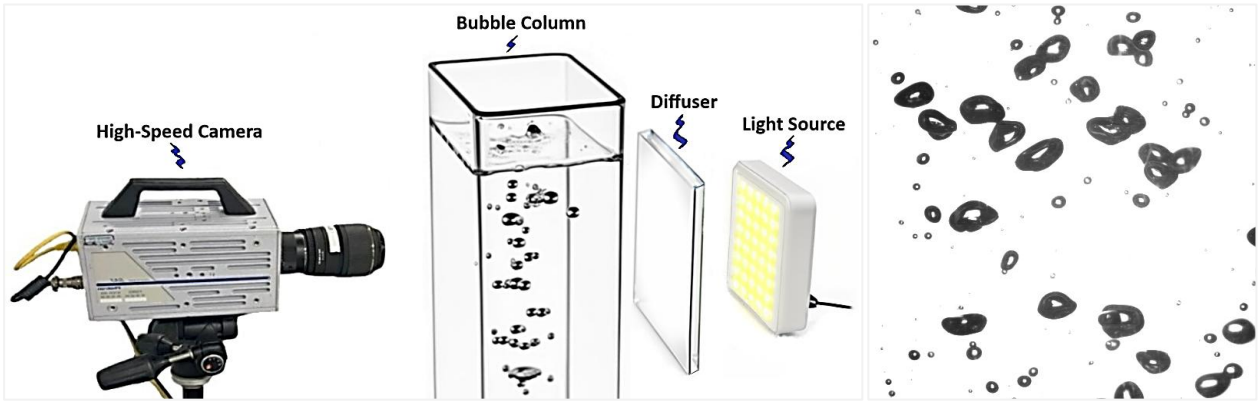

Figure 1. *(a) A typical shadowgraphy setup, including a high-speed camera capturing images of a bubble column illuminated by a light source with a diffuser to ensure homogeneous backlighting. The setup illustration is generated using Generative AI models guided by text, images, and sketches. (b) A representative bubble image produced during this thesis.*

For images like these, the isolated bubbles can generally be segmented efficiently through conventional methods. Popular approaches include thresholding methods (e.g., Otsu's method), region-based techniques (e.g., Watershed), and edge detection methods such as Canny. Segmentation methods such as these rely on specific assumptions about distinct objects. For instance, color thresholding assumes a certain uniformity in the color of each object, while edge detection relies on noticeable gradients at object boundaries. However, the success of these techniques depends on how well the asserted assumption holds, which is often not the case across the full extent of the object or contour. For bubbles that overlap or have incomplete contours, the uniformity, contrast, or gradient is often insufficient to capture the true forms.

When these conventional techniques underperform locally, incorporating prior knowledge about the shape can improve segmentation. For example, the Circle Hough Transform can detect circular features even when the edges are noisy, incomplete, or otherwise imperfect. Cui et al. [3] applied the Circle Hough Transform to restore the shapes of sub-millimeter bubbles, which are expected to be spherical. This assumption is valid for sub-millimeter bubbles, as their high area-to-volume ratio ensures that the surface tension dominates over the external forces such as drag, buoyancy, and inertial forces. More commonly, researchers complete hidden contours by employing the Hough Ellipse Transform [4-6]. However, this premise that bubbles have ellipsoidal shapes is weak for dynamic bubbly flows. In such flows, interactions between bubbles and the dominance of external forces over surface tension cause significant deformations, resulting in irregularly shaped bubbles.

The shape-fitting methods described above are examples of algorithmic approaches to segmentation. A wide range of algorithmic segmentation methods exists, including active contours (snakes), feature-based clustering, and graph-based segmentation. However, these advanced methods often require user assistance, involve numerous parameters that need careful fine-tuning, and rely on expertise. Hence, researchers have increasingly turned to Convolutional Neural Networks (CNNs) for their segmentation tasks. Unlike conventional methods, CNNs learn directly from data, eliminating the need for user assistance and parameter tuning. By training on large datasets, CNNs can identify and segment bubbles with remarkable accuracy, even under challenging conditions such as overlapping bubbles and incomplete boundaries. As a result, the segmentation of bubbles with CNNs has become an active and ongoing area of scientific research. Outsourcing segmentation to CNNs has already enhanced accessibility, which in turn has improved the ability to compare experimental data with numerical simulations. Moreover, the ability to track and analyze bubbles in real-time offers practical applications, particularly for tuning and monitoring industrial processes.

In this report, we aim to contribute to the exploration and advancement of Convolutional Neural Networks for segmenting bubbly flows, with a particular emphasis on achieving more accurate segmentation and precise tracking of non-spherical bubbles. Disregarding the optimization procedure, as it is well-established, the performance of CNNs heavily depends on the network architecture and the quality and quantity of the training data [7,8]. Hence, this study draws upon these two key aspects. The primary objective, therefore, is to develop a dataset and propose a CNN architecture capable of accurately capturing bubble flows with complex shapes and overlaps, as observed in shadowgraphy.

2 CNN models for bubble segmentation

In this chapter, we will investigate different CNNs that have been applied already or show potential, and discuss how spatial and temporal relations could be exploited. After this analysis, we will propose an architecture and reconstruction strategy for overlapped bubbles. The primary objective of any neural network is to draw reliable inferences by generalizing patterns during training rather than relying on spurious correlations embedded in the training set. This is why evaluation on an unseen test-set is a cornerstone of the training process. We would like our CNN to rely on relations such as the clear contours of bubbles, spatial and temporal consistency, or even principles grounded in fluid dynamics and light tracing, not on pitfalls of memorization and false generalizations of incidental patterns. Modern CNNs have thoughtful architectures that go far beyond the black-box approach of a vanilla CNN. They are designed to enforce certain approaches in correspondence with the task at hand. Convolutional layers, encoder/decoder, attention, embeddings, skip connections and RNN-cells are some examples of the fundamental building blocks to do so. The challenge ahead is in finding the best composition tailored for bubble segmentation.

2.1 Existing architectures and application for bubble segmentation

Advanced CNNs have intermediate products in their pathway to enforce a procedure. The popular Mask R-CNN for example, which has the Region Proposal Network (RPN), Region of Interest (RoI) Align and a segmentation branch to ensure regional-based segmentation. The RPN generates candidate object bounding boxes (proposed regions) for the RoI Align to extract features from, with whom the Segmentation Branch predicts a binary mask for each detected object [9]. The architecture is shown in Figure 2.

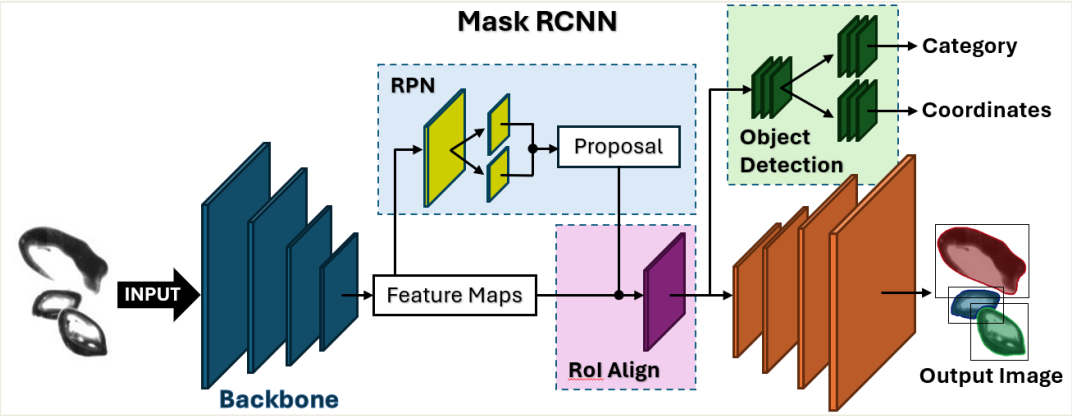

Figure 2. *Architecture of Mask R-CNN: Extracts features, generates region proposals, and outputs object masks and classifications.*

The Mask R-CNN is a popular network for instance segmentation and is used by many researchers for bubble segmentation. Kim and Park [10] trained this Mask Region-based Convolutional Neural Network (Mask R-CNN) for bubbles and achieved an AP_{50} (Averaged Precision for IoU ≥ 0.5) of 98% on the test-set. Even when tested with bubble-swarm flows not included in the training set, the model detects more than 95% of the bubbles. The segmentation of the submillimeter bubbles by Cui et al. [3] was achieved by fine-tuning the Mask R-CNN model of Kim and Park. They used transfer learning, a common practice for CNNs, which builds on pre-trained models to save computational resources. This approach is particularly effective as it harnesses the knowledge embedded in previously trained models, allowing them to adapt to their specific case by fine-tuning with a small dataset.

The study of Malakhov et al. [11] explored an application for bubbles during boiling at various pressures. They utilized the Mask R-CNN and a U-Net architecture as well, to detect bubbles located on a heated wall until the moment of their departure and subsequent lift-off. The U-Net, named after its U-shaped architecture structure, is a CNN that was developed for biomedical image segmentation by Ronneberger et al. [12] in 2015. It consists of a contraction path (down-sampling) and an expansion path (up-sampling) that are connected with skip connection, as shown in Figure 3. The U-Net provides *semantic segmentation*, which means it does not capture the instances of each category, it defines the bubbles from the background but does not distinguish them from each other. To capture the bubbles as individuals, you need *instance segmentation*. However, you could describe the separation of the objects as another semantic segmentation task.

Therefore, Hessenkemper et al. [5] applied a double U-Net. As the name suggests, it consists of two independent U-Nets. The first U-Net (UNetL3) is trained to distinguish between the foreground and background, identifying all pixels that belong to bubbles. The second U-Net (UNetL5) is trained to find all pixels that belong to the boundaries between overlapping bubbles. The method proved better than the Mask R-CNN on the validation set but performed significantly

worse on additional data from different imaging conditions. The UNetL3 still accurately captures the outline of the bubbles, but the second U-Net has trouble finding the correct intersections. Consequently, the high AP_{50}, which is close to 1 for the test-set, decreases to approximately 0.7.

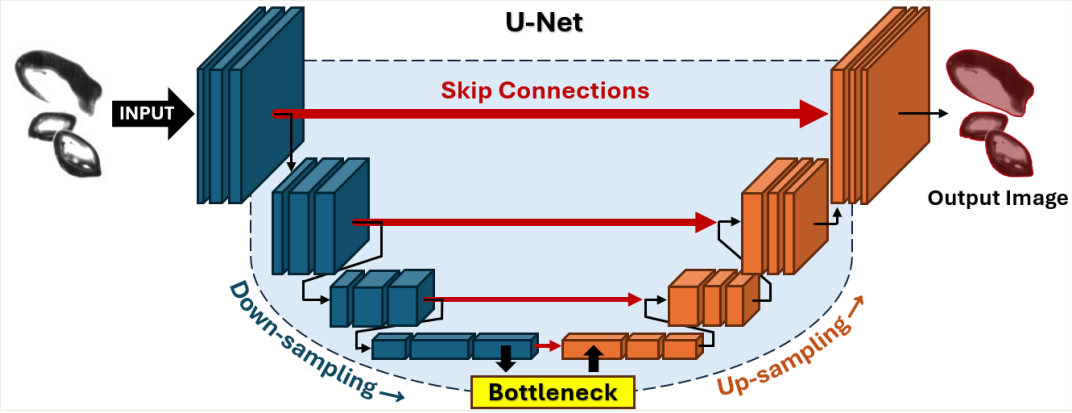

Figure 3. *Architecture of U-Net: Combines down-sampling and up-sampling with skip connections for precise image segmentation.*

A second approach utilized by Hessenkemper et al., is the StarDist, which is particularly effective for segmenting closely packed or overlapping objects. It uses star-convex polygons to segment and predict the shapes by estimating the distances from their centers to their boundaries. A three-layer U-Net is used as the backbone of the StarDist, together with 64 radial directions for the star-convex polygons. The StarDist achieves a higher AP than the Mask R-CNN and shows a comparable robustness for the additional set, but captures less accurate borders than the UNetL3. They encountered this drawback by utilizing the StarDist, and dilate the StarDist results until they fit the bubble mask of the UNetL3 as a post-processing step. Upon utilizing the thus far mentioned CNNs on an image of our own, the robustness of the StarDist is confirmed, see Figure 4.

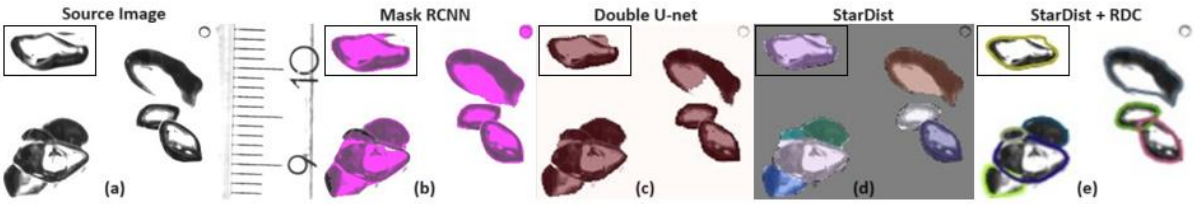

Figure 4. *Inference of an image (a) by the Mask R-CNN (b), the Double U-Net (c), the StarDist (D) and the StarDist with a bubble reconstruction method called RDC as trained by Hessenkemper et al. [5]*

The field of instance segmentation offers a much broader scale of CNNs than those currently applied or reviewed for bubble segmentation. Sharma et al. [13] recently published a thorough survey on object instance segmentation. They have outlined the most popular CNNs and demonstrated their performance when trained on standard datasets such as COCO and Cityscapes. However, it is hard to relate such results to our application. Some CNNs excel on one dataset but underperform on another. Additionally, the performance is highly influenced by the depth of the networks [8]. While deeper networks may excel on these benchmarks, they often come with significant computational costs and inefficiencies. Compact networks, on the other hand, can achieve comparable results with reduced resource demands, making them more practical for tasks like bubble segmentation where efficiency and adaptability are desirable. This trade-off is rarely reflected in standard evaluations, making it hard to draw conclusions for our specific application. Investigating all these CNNs exceeds the scope and capabilities of our research. Therefore, we have selected two additional lightweight CNNs to assess alongside the Mask R-CNN, namely the YOLAct and the SOLOv2.

Proposal-based methods like YOLAct and Mask R-CNN rely on bounding boxes to segment objects, making them effective for tasks involving clearly defined regions. YOLAct ("You Only Look At CoefficienTs") [14] extends the YOLO framework for object detection by representing object masks using a small set of mathematical coefficients of a fitting function instead of dense pixel-by-pixel masks. In contrast, proposal-free methods of SOLOv2 ("Segmenting Objects by Locations"), directly cluster pixels without relying on bounding boxes. The lightweight architecture ensures faster training and inference. The SOLOv2 [15] is well-suited for dense and complex images where pixel-level precision is paramount, as it segments objects by predicting instance-aware masks for each location independently.

The interest in lightweight CNNs for bubble segmentation is due to their efficiency and real-time processing capabilities, which also reduce the demand on hardware resources. Bubble segmentation typically involves simpler visual features, making large, complex models unnecessary. Moreover, small networks are easier to train on limited data-sets and deploy in practical applications. If needed, one can enhance the segmentation accuracy of a network with methods such as PointRend or use a cascaded version of the networks. PointRend refines segmentation results by focusing on boundary regions with a high-resolution iterative rendering approach, improving the precision of object edges [16]. Cascaded networks, on the other hand, utilize multiple stages of segmentation, progressively refining the output at each stage. While both methods significantly enhance segmentation quality, they come with added complexity and increased inference time.

2.2 How can spatial relations be exploited?

Convolutional Neural Networks excel at leveraging spatial relationships within structured data, such as images, making them particularly effective for tasks involving consistent and localized features. The model inferences shown in Figure 4, demonstrate that CNNs are able to close contours to some extent, even when applied to unseen data from a different setup. Notably, StarDist excels in completing contours, resembling a capacity similar to the human tendency to complete contours.

In complex scenes, deciding which objects to segment or how to organize them hierarchically can be challenging. In contrast, bubble segmentation is more straightforward. Bubbly flows are typically set against a homogeneous background and lack other objects. There is no ambiguity between features when deciding what should be segmented or considered background, nor are there intricate hierarchical relationships that influence grouping. The focus is solely on identifying the bubbles as distinct entities, with segmentation guided primarily by local features such as edges and contours. Therefore, advanced techniques like attention mechanisms and transformers, while powerful for tasks involving complex contextual, hierarchical and global dependencies, are overkill and do not address the challenges relevant to bubble segmentation. Although transformers might offer potential in scenarios with dense overlapping bubbles or significant interference from other objects, their advantages are unlikely to justify the additional computational overhead. In addition, transformers require a large amount of training data [13]. For applications involving additional objects, transfer learning with targeted examples provides a more practical and efficient solution. Moreover, previously utilized models have shown that spatial relations of features are generally well captured by lightweight CNNs.

Besides the planar dimensions of the image, there is another spatial dimension to consider. Knowledge of the depth can be a valid method to distinguish objects when planar clustering fails. Researchers have recently ventured into methods that directly estimate the focus position from a single image. The depth of a bubble can be indicated by the focus blur, size, shadow intensity, and by its occlusions and overlaps with other bubbles. General depth-map related neural networks, show great performance in estimated depth from real-world scenes. Monocular depth estimation relies on perspective geometry, object scaling, texture gradients, and learned scene semantics, using visual cues such as edges, shading, and sharpness to estimate relative distances. However, our shadowgraphy images of bubbles offer limited cues to estimate the depth.

Whang et al. [17] designed a Generative Adversarial Network called FocGAN that discriminates and eliminates out of focus-droplets from shadowgraph droplet images. This research shows the ability of Convolutional-based networks to distinguish out-of-focus objects. For similar images, Zhang et al. [18] employed discrete depth estimation using the EfficientNet CNN to determine the depth of polystyrene particles, blood cells, and plankton. The network classifies particle depths into discrete categories corresponding to predefined intervals, with 9 intervals for polystyrene particles and blood cells and 7 intervals for plankton. The EfficientNet achieved accuracies of 99.7% for polystyrene particles, 99.31% for blood cells, and 96.48% for plankton. Other networks are able to get similar accuracy, but EfficientNet significantly outperforms them in inference and training time.

Discrete intervals are a logical approach because captured defocus is tied to the resolution. We can only speak of a “Depth” of Field because we perceive a circle of confusion within a pixel to be in focus. Within this range, variations in blur are indistinguishable. Nonetheless, Zhang et al. [6] used a Multi-input Residual Convolutional Neural Network (MRCNN) to estimate the depth of isolated bubbles directly from light-field images. Their network combines a standardized crop of a single bubble image extracted from the bubble flow and an approximate refocusing parameter as inputs. The network outputs a refined refocus parameter that quantifies the focus shift needed for optimal clarity at a specific depth. The input parameter is precomputed by comparing the computed sharpness of the bubble against a

calibrated mapping. However, the network's output is a continuous value that exhibits a relative error of less than 1.0 % and an average absolute error of 1.03 mm.

Depth estimation through depth-from-defocus benefits from the otherwise unwanted blur associated with a shallow depth of field. Nonetheless, we would like to investigate if a depth estimation of our bubbles could aid in distinguishing bubble clusters. Furthermore, integrating a network such as EfficientNet for depth estimation could enhance models that leverage temporal continuity by utilizing this additional dimension.

2.3 How can temporal relations be exploited?

Although segmenting each image in isolation is currently the most common approach for bubble segmentation, there is significant potential to improve accuracy by incorporating information from preceding and subsequent frames. If our network only uses previous results, it will still allow for real-time inference.

There are many approaches to extend image-based CNNs for sequences. A straightforward method involves processing temporal patterns by utilizing tensors instead of matrices, essentially stacking a sequence of images. This approach is similar to how color channels are processed for 2D-CNNs. While these so-called 3D CNNs can capture temporal consistency across frames, they are computationally intensive and memory-demanding. Another more sophisticated approach could be the exploitation of spatiotemporal transformers, which is an emerging yet pioneering technology [13]. However, adaptation to specialized tasks like bubble segmentation remains largely unexplored and would presumably be overkill, as it requires significant development and a vast amount of training data.

To integrate previous results with new inferences, current feature maps can be warped by past outputs, functioning in a manner similar to a Recurrent Neural Network (RNN) [19]. However, due to the vanishing gradient problem inherent in RNNs, applying techniques like Convolutional LSTMs or GRUs is more suitable. These memory units can be integrated into different stages of the network, such as at the feature levels within the encoder or decoder [20]. Integrating ConvLSTM/GRU in the encoder stage can enhance global context and potentially improve detection. In the decoder, it could enhance the mask through spatiotemporal consistency. Alternatively, they can be integrated before the Softmax layer, which corresponds to a temporal filtering of the result [21]. The bottleneck (between encoder and decoder) is often the most suitable location. This is due to the fact that the bottleneck provides the most compact and abstract representation of the input data. A ConvLSTMs here focuses on learning global temporal dynamics while avoiding an overload of spatial details.

There is however another approach possible that leverages continuity instead of memory. Models like MaskTrack R-CNN [22] and Video Object Segmentation (VOS) [23] use optical flow to track and refine instance masks over time. There are also networks that rely on optical flow as an additional input, like DeVOS [24] for semantic segmentation. Making distinct segments based on optical flow is similar to how humans group objects moving in the same direction and at the same speed, a phenomenon known as the Gestalt principle of common fate. Unfortunately, it is poorly applicable for bubbly flow. High contrast, Non-Rigid Motion, and light artifacts make it difficult to compute reliable optical flow.

2.4 Reconstruction of occluded contour

The trajectories of individual bubbles are derived from the spatiotemporal displacement of the center point of the segments. Therefore, you need the complete contour of every bubble on every image. It is common practice to deal with any occlusions by estimating the occluded part of the hidden contours. Consequently, if we want more precise tracking we need better estimations. Hessenkemper et al. [5] extended the StarDist with Radial Distance Correction (RDC) to reconstruct occluded contours. StarDist generates star-convex polygons by predicting radial distances from object centers but does not correct occluded contour. The RDC method uses a feedforward neural network, trained on synthetic images with varying overlaps (10%-90%), to extend the shortened radial distances at occluded borders based on visible parts. You could say the method estimates the occluded form based on smooth continuation of the radii. While this method improved upon the previously applied ellipse fitting, it has limitations. Hessenkemper et al. suggest that the use of image sequences has the potential to improve the occluded bubbles reconstruction, for the hidden bubble contours may be better visible at an earlier or later moment.

Generative networks that utilize sequential data for inference are already widely employed across diverse research fields. Xing-Wei et al. [25] explore the application of Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) for predicting unsteady flow fields. By training on datasets generated from numerical simulations (OpenFOAM) and experimental Particle Image Velocimetry (PIV), they demonstrate the effectiveness of LSTM and GRU in capturing

temporal and spatial flow field characteristics. LSTM performs well in velocity field predictions, while GRU excels in pressure field predictions. LSTM has significant potential for our reconstruction task due to its ability to capture temporal evolutions, such as the dynamic contours of bubbles. With sufficient training data from simulations and experiments, they are likely to infer underlying causal relationships more effectively than the predictive capabilities of the RDC method.

Reconstructing the radii of convex shapes instead of pixel masks, as demonstrated by Hessenkemper et al., could simplify the reconstruction, reducing the need for a ConvLSTM to a standard LSTM that just predicts radii. Normalizing the area to exclude size variations from the equation could further streamline the process, enabling the network to predict shapes based on the generalization of contour propagation. However, incorporating predictions from previous time steps for sequential occlusions might introduce cascading errors. In such cases, false predictions or discontinuous detections could propagate and lead to inaccuracies in future bubble reconstructions. Furthermore, the assumption that incorporating nearby bubbles into the prediction framework is unnecessary for accurate predictions has yet to be substantiated.

2.5 Proposed design

The field of instance segmentation is broad, but video segmentation is even more expansive. Research has been conducted on occlusion-aware segmentation of images and sequences in real-world scenes, where tasks like depth estimation, optical flow, and LSTMs are key components. Based on the discussions in the previous subsections, we consider depth estimation and strategic convLSTM placement between the backbone (encoder) and mask head (decoder) to be the most promising approaches. If we only place ConvLSTM layers at the bottleneck, we minimize added complexity and ensure real-time processing capability [21].

Choi et al. [26] studied different models to infer the optical flow of bubbly flows. They noted that, while all models accurately measure the unsteady velocities of individual bubbles, their accuracy rapidly diminishes when the bubbles form clusters. Optical flow methods are vulnerable to occlusions and motion boundaries, due to a lack of object-level information [27]. Therefore, we do not include guided optical flow for the instance segmentation task, as it adds complexity without significant benefit. We expect that the velocimetric distinction is less decisive than continuous feature aggregation. Moreover, clustered bubbles may lack sufficient distinction in their optical flow, and the motions can be misleading due to bubble deformation (wobbling) or errors caused by light artifacts, as evidenced by the limitations of the optical flow models applied by Choi et al.

To achieve robustness for unseen bubbly flows, it is beneficial to design a multi-stream architecture that separates distinct tasks and optimizes their individual performance. However, it is equally important for individual streams to support transfer learning, enabling users to adapt them to their specific setups. For this reason, we propose to decouple the reconstruction task from the primary segmentation task. The architecture for the segmentation task is shown in Figure 5 and the reconstruction task is depicted in Figure 6.

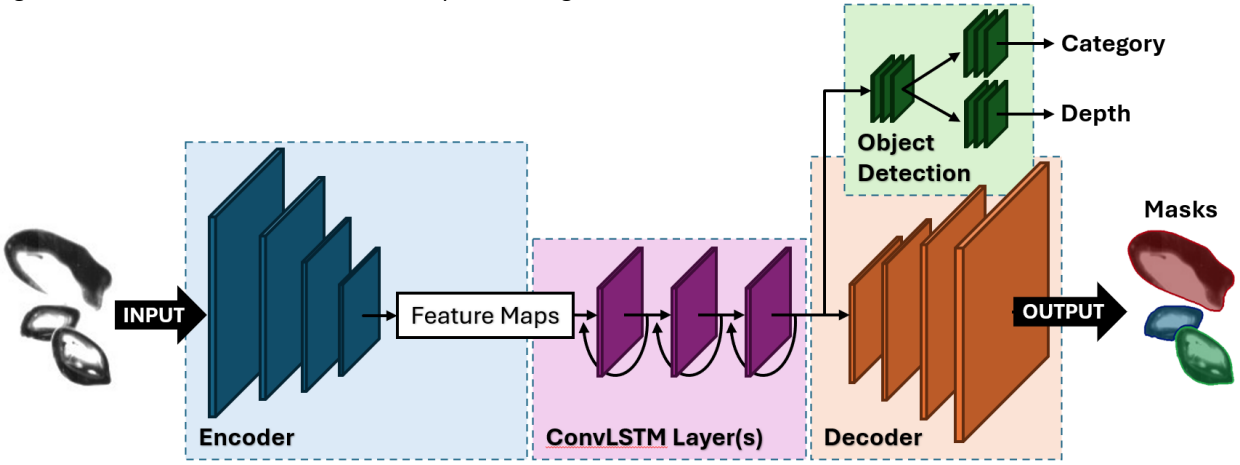

Figure 5. *Proposed architecture for bubble segmentation, featuring an encoder for feature extraction, ConvLSTM layers for temporally consistent feature aggregation, and a decoder with dual branches for object detection and depth estimation, resulting in segmented masks.*

We will investigate whether the CNN should incorporate depth estimation. For this thesis, we plan to introduce depth awareness into the segmentation categories as proof of concept. We will manually assign the following labels: singular, occluding, occluded, and occluding-occluded bubbles. With these labels, each bubble is tagged with a relative order of depth to the bubbles whose contours it intersects. This approach is expected to enhance feature aggregation within the bottleneck. If successful, future designs could integrate an output branch after the bottleneck for discrete or continuous depth estimation, like the branching by Nekrasov et al. [28], or decouple the two tasks. The reconstruction of the three-dimensional interface of the bubbles has not been addressed so far. Although monocular and stereovision-based 3D shape recovery for bubbles is an active research area [29,30], such implementations are still likely to rely on accurate 2D segmentation and the reconstruction of occluded forms.

The most simplistic way to add a reconstruction mechanism for the occluded bubbles is by adding a second upsampling branch, trained on the occluded segments. However, we see more potential in a lightweight decoupled approach such as the RDC method that Hessenkemper et al. employed. This method, however, has a previously unmentioned limitation. Star-convex polygons only capture bubbles that are star-convex, meaning that a straight line from the center to any point on the boundary lies entirely within the shape. While this condition is most often met, it is not guaranteed, particularly in cases of complex deformations. Lin et al. [31], employed a Graph Neural Network (GNN) to infer predictions from moving graph nodes as segments, which do not have such limitations. Inferring nodal mesh-like graphs for a Graph Neural Network (GNN) has proven to be a powerful approach for solving complex physical problems and making predictions in Finite Element Method (FEM) and Computational Fluid Dynamics (CFD) [32,33].

The GNN proposed by Lin et al. combines the motion of the visible regions with the previously reconstructed parts of the 3D node segments to predict the motion of occluded regions using an LSTM-enhanced architecture, by which you update the status quo. This so-called OcclusionFusion could potentially be applied to bubble segmentation if the segments are treated as planar layers with relative depths that can occlude each other. Generating graph representations from segmented bubbles and computing nodal motion from previous graphs as a preprocessing step would enable the use of this LSTM-based GNN. The prediction relies on previously observed shapes, which is both its strength and its limitation when occlusion is present from the outset. Therefore, reconstruction methods based on shape generalization, such as the RDC approach of Hessenkemper et al., cannot be entirely disregarded. Nonetheless, we propose this network, as described and illustrated in Figure 6, as a promising direction for future research. As shown in the figure, the GNN uses a U-shaped architecture with downsampling and upsampling operations.

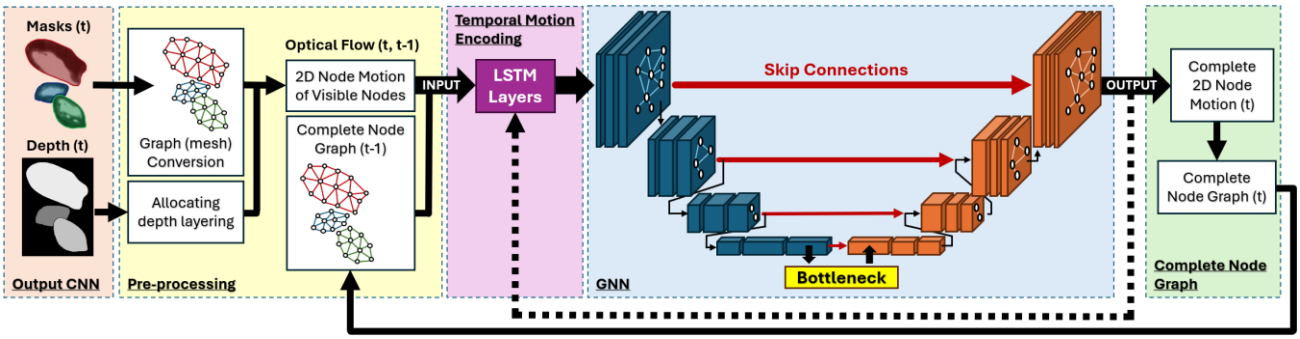

Figure 6. *Proposed pipeline for occluded bubble reconstruction, combining segmentation preprocessing, temporal motion encoding with LSTMs, and a U-shaped GNN for predicting occluded node motion and reconstructing occluded regions.*

The LSTMs that are part of the segmentation and reconstruction pipeline are sensitive to overfitting. A large labeled data set will be needed. We will need sequences of segmented bubbles with complete forms despite occlusions. Typically, the training set is composed of experimental and/or synthetic data. To narrow the scope of this project, we limit our focus to requiring experimental data that would be needed for the proposed design. Additionally, we will explore candidates for the backbone and decoder by investigating the previously mentioned lightweight CNN models and determining whether the estimation of relative order of overlap is possible. We will continue with a single encoder stage and use the categories to estimate the depth order for contour-crossing bubbles. With regard to obtaining suitable training data, we give a head start by exploring a method by which occluded shapes can be recovered for future contour estimations.

3 Experimental setup

This thesis builds upon a series of initiatives by the Power & Flow Group within the Department of Mechanical Engineering at TU/e, aimed at studying bubbles using a typical shadowgraphy setup, as previously shown in Figure 1a. The dataset for this research will be obtained from a bubble column originally designed and built by Creusen [34] in 2020. This bubble column was first utilized by Josso [35], whose setup relied on a single high-speed camera. Consequently, this setup cannot track the 3D position, capture occluded shapes, or reconstruct the geometry of the bubbles.

Multi-camera systems have already been deployed for the tracking and 3D reconstruction of bubbles. She et al. [36] utilized wide-baseline stereo setups to quantify and monitor methane fluxes escaping from natural reservoirs on the seafloor at a depth of 1000 meters. They applied epipolar geometry to identify stereo matching pairs, whose corresponding points were triangulated to fit 3D ellipsoids and estimate bubble volumes. While a wide-baseline stereo setup reveals the 3D position and, to some extent, the bubble shape, it does not capture occluded forms. Moreover, the accuracy of this method declines rapidly when higher bubble densities are encountered, leading to biases in both the center locations and the ellipsoid fitting. This could be circumvented by increasing the number of views, thereby improving the likelihood of unobstructed observations of individual bubble instances.

Recently, Hessenkemper et al. [37] used a multi-camera system with up to four cameras on an octagonal water tank to overcome the aforementioned limitations of segmentation caused by occluded bubbles. Their paper focuses on detection and tracking, achieved by matching detected contours across views by a Siamese Neural Network. To compute the center location of each bubble in all views, they utilized the previously mentioned StarDist CNN, which reconstructs the hidden parts to determine the centers. Consequently, the training data must include complete ground truth annotations. To overcome this challenge, they recorded and segmented single bubble trajectories. Subsequently, they stacked multiple trajectories with sufficient three-dimensional spacing to create artificial sequences with known ground truths, mimicking a bubble swarm.

In their previous study [5], Hessenkemper et al. compiled bubbles in a similar manner but disregarded the depth. This approach was also applied by Homan, the initiator of this project, and Deen [38] as it allows for the rapid generation of large datasets. While this method provides a practical solution, it is more desirable to work with real images. Building on the previously discussed methods and as an extension of these, we will explore the reconstruction of overlapped bubbles using opposing stereo vision. The reconstruction of occluded bubbles should be possible, as long as occluded parts are not occluded by a third in the opposite image.

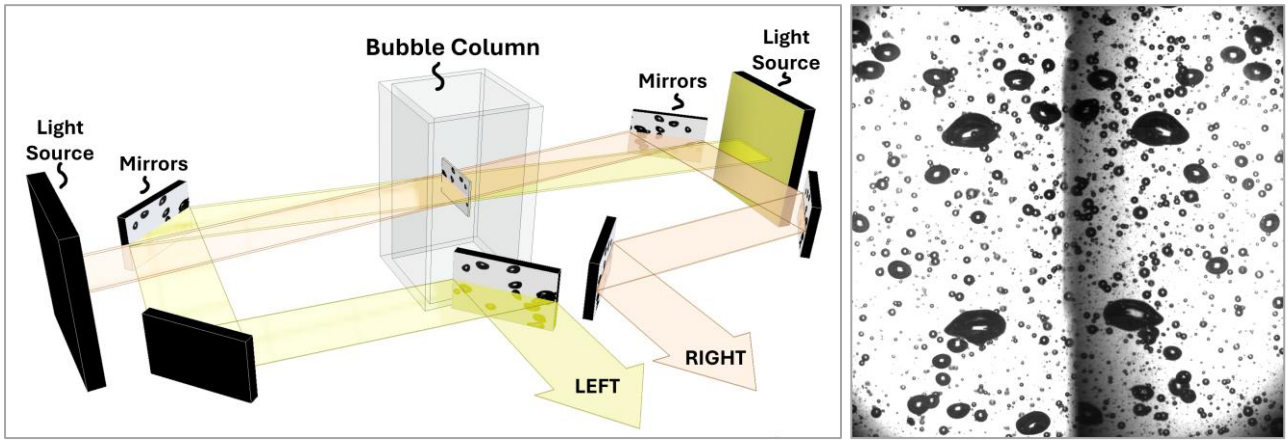

Figure 7. *(a) Schematic of the opposing stereo vision setup for a bubble column, utilizing mirrors and light sources to capture bubbles from opposing perspectives. (b) Example of a captured image showing front and back views of bubbles with similar contours despite the misalignment.*

3.1 Stereo imaging method

Due to the limited availability of high-speed cameras and their continuous demand within the department, an alternative solution was implemented. An image doubler from LaVision, a stereoscopic device, was used to split the image into left and right sections onto a single camera sensor. Using first-surface mirrors, as illustrated in Figure 7, this setup enabled the simultaneous capture of opposite views with a single camera. Bubble formation is initiated at the bottom of the tank by a plate configured with a 3x3 or 5x5 array of holes, each with a diameter of 1 mm. The airflow is provided by an air supply system and controlled with a Mass Flow Controller (MFC), which maintains a constant mass flow into the chamber beneath the perforated plate.

The necessity of aligning a light source collinearly with each view prevents us from capturing the true backside. To avoid interference from the equipment, a slight angle must be introduced to capture both sides, as shown in Figure 7. This misalignment alters the perspective, causing shape distortion proportional to the angle of misalignment. Consequently, the shape is not fully represented in the opposing image. If the bubbles were flat in depth, the change in perspective would result in simple horizontal stretching or compression, enabling the segment to be reconstructed. However, depth variations cause the rotation of free-form bubbles to produce uneven horizontal shifts and the appearance or disappearance of contours along the sides. Consequently, the shape cannot be accurately recovered using affine transformations, such as stretching or compressing to align it with the opposing captured image.

Fortunately, some factors work to our advantage. If the shapes are symmetrical around the axis of misalignment, the depth-induced distortions become less noticeable. E.g., a cylinder rotated around its axis shows no apparent distortion. By this principle, the sphericity of the bubbles is a beneficial feature. Another advantage is the relatively low depth of the bubbles compared to the overall setup, which keeps the uneven horizontal shifts relatively small. For instance, a long cylinder viewed from the top would exhibit significant horizontal contour shifts with even a small change in perspective, whereas a short cylinder experiences much less distortion. Figure 7 provides a preview confirming that the distortion is minimal. In order to keep the discrepancy to a minimum, the setup is built with the smallest possible misalignment angle.

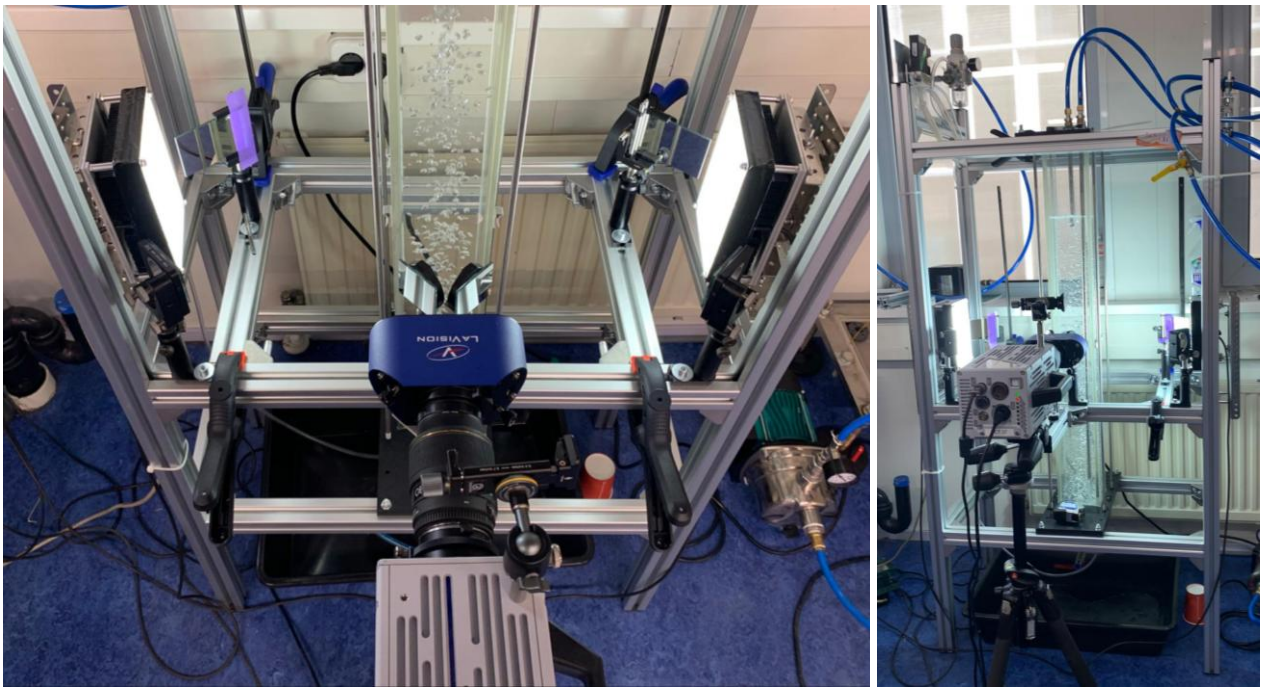

Figure 8. *(a) Realized setup for opposite stereo vision of a bubble column using mirrors. (b) The same setup, enhanced with tubes inserted from above, allowing flow to be introduced through pumping.*

3.2 Setup and Equipment

The realized experimental setup, built around the bubble column of Creusen [34], is depicted in Figure 8a. The mirrors are positioned on optical posts that are mounted on a horizontal structure of aluminium extrusion profiles. The second and third mirrors on both sides are equipped with adjustable mounts for precise alignment. The LaVision image doubler is mounted on a Sigma EX 105mm f/2.8 Macro lens. In a part of the recordings, a 2x teleconverter is used. The converter or lens is connected to the F-mount of the camera. The high-speed camera used is the Photron FASTCAM SA3 model 120K-M3, offering a resolution of 1024x1024 pixels at 2000 frames per second.

A custom light source was developed using a 200 W LED-PCB floodlight plate (DC 29-42V). To protect the frosted glass diffuser from the LED's heat, a protective glass is placed directly over the LED panel. Above the frosted glass diffuser, a rough-polished plexiglass panel serves as the final diffuser. The entire assembly is enclosed with black aluminum tape to retain light within the stack. The two light sources were connected to a programmable DC power supply to provide a stable direct current, eliminating fluctuations typically caused by grid power frequency.

The camera is focused on a protractor triangle that is submerged into and kept in the middle of the bubble column, as shown in Figure 9a. In the same figure, two protractor triangles are visible on the outside of the column. They are needed to determine the misalignment angle. Figure 9b presents a typical calibration image captured with the high-speed camera prior to recording.

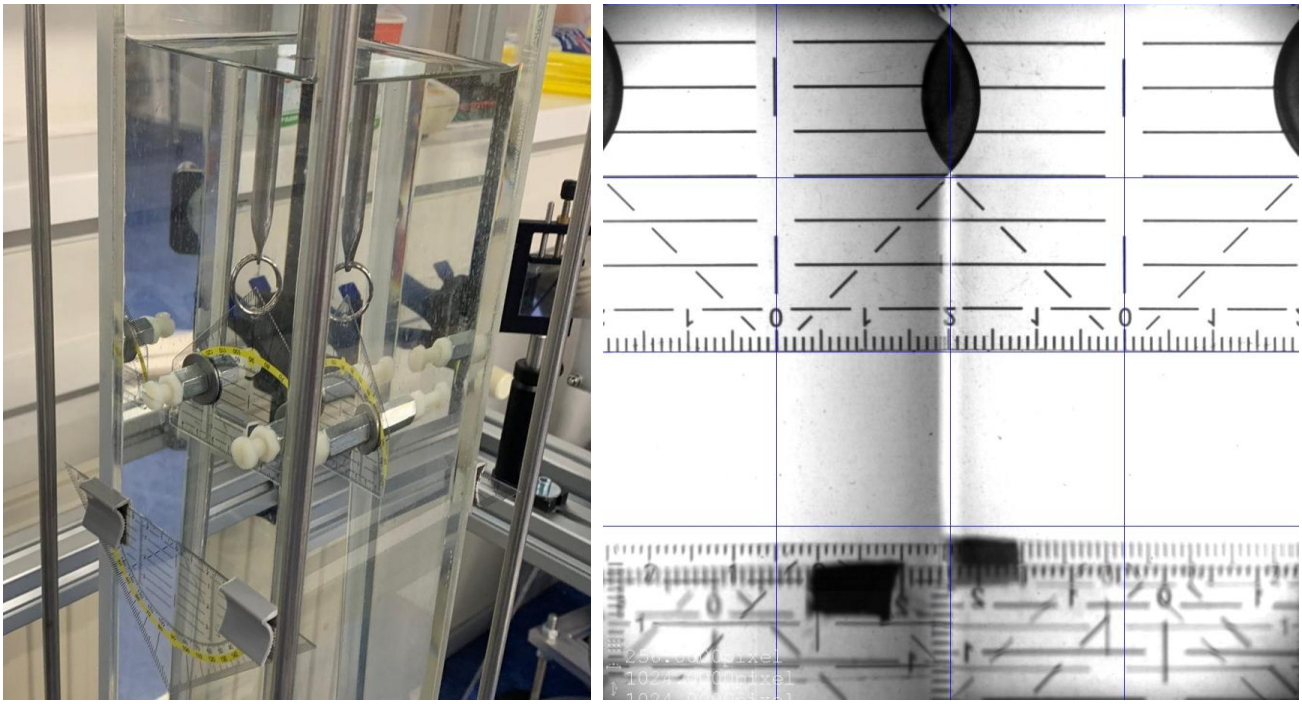

Figure 9. *(a) The experimental setup showing the bubble column with a protractor triangle submerged inside and two others attached to the outside surfaces on both sides to determine the misalignment angle. (b) A typical calibration image captured with the high-speed camera, featuring the protractor triangles used for alignment and scaling.*

3.3 Experiments

During this thesis project, experiments were conducted from three different perspectives. The first is a close-up view of the bubbles captured using a 2x teleconverter, covering a width of 4 cm. The second is a more distant view, achieved without a teleconverter, capturing a width of 8 cm. The third consists of images taken without mirrors, using one of the lamps and the camera without the LaVision doubler, capturing the full column width. A diverse dataset was created by varying apertures, LED voltages, and flow rates, with a constant exposure time equal to the frame interval. The MFC has provided mass flows up to 2 L/min. The mass flow of air is slightly different but can be computed using appropriate gas correction factors.

Hessenkemper et al. [5] mention that, although they did not apply this approach, the training data should reflect cases with background flow to some extent, as flow could influence the appearance of the bubbles. Following this advice, we implemented three types of flow: upward, downward, and swirl. These flows were created by circulating the water of the column via two stainless steel tubes inserted from above, as shown in Figure 8b. Upward flow was created by suction from the top of the bubble column and discharge at the bottom, while downward flow was achieved vice versa. The swirl was created by another long tube with holes along its side and a capped end. A centrifugal pump with a flow rate of 3500 l/h was used for all three configurations. In videos with upward flow, the bubbles in the center of the column can be observed to rise faster due to the core flow. Video recordings of downward flow show some bubbles stagnating and being pushed downward. The effects of the swirl are evident in the videos, where bubbles exhibit noticeable horizontal shifts. Typical images of bubbles in upward flow and swirl are shown in Figure 10a and 10b, respectively.

While using the pump, we discovered that excessive throttling could temporarily make the deionized (DI) water appear cloudy. In low-pressure zones, dissolved gases in the DI water are released, forming microbubbles. The dataset is extended with videos of this phenomenon which could make the CNN more resilient to polluted media (see Figure 10c). The scientific community often investigates surfactants in bubbly flows because surfactants reduce surface tension, altering bubble dynamics and stability. Therefore, we recorded videos with the addition of one of the most popular surfactants, Sodium Dodecyl Sulfate (SDS), at concentrations of 0.8, 1.6, 2.4, 3.2, and 4.8 mM/kg, as used for the first five concentrations tested by Vega and Montanero [39]. A typical image is shown in Figure 10d. Lastly, additional videos were recorded, featuring elements such as droplets on the outside wall, a halogen lamp on one side, and the water surface within the frame. In the GitHub repository [40], one can find a typical video for each of the aforementioned experiments. The captured dataset contains approximately 4 million images from 400 videos, with a total size of about 3 TB.

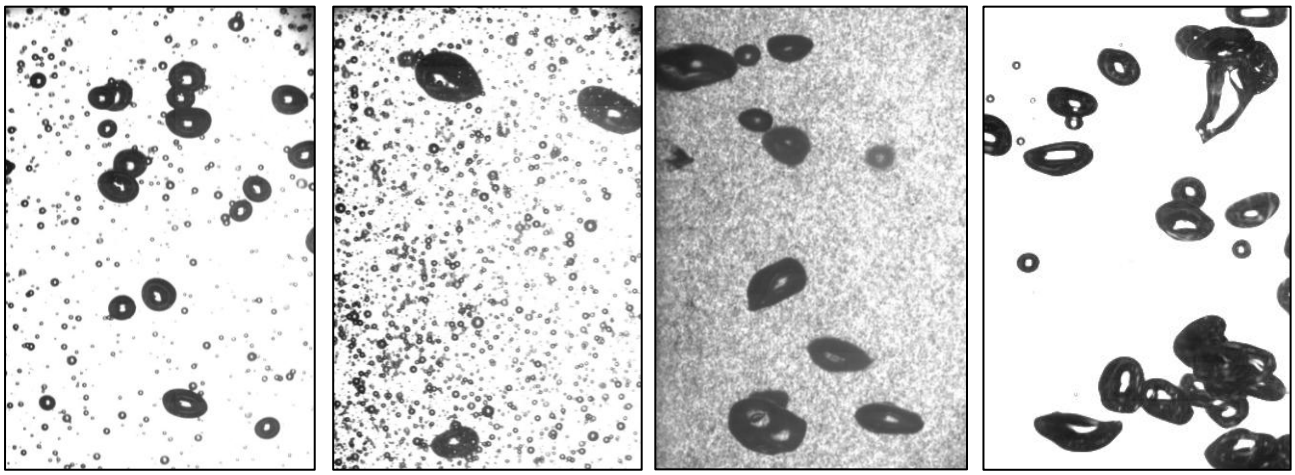

Figure 10. *A typical image of bubbles: (a) in upward flow, (b) in swirl flow, (c) with microbubbles formed by dissolved gases in deionized water, and (d) with surfactants (SDS at 4.8 mM/kg).*

4 Data processing

The experimental setup has yielded a substantial amount of data. To manage this data efficiently, we have developed several JupyterLab scripts. For instance, the script "MakeVideosFromImageSets.ipynb" generates MP4 videos for all 400 recordings. Another script combines all these videos from different sets into a single compilation, with directory names displayed as subtitles. This video provides an overview of the experimental results and is stored together with the obtained data. The high-speed camera used in the experiments exhibits fixed pattern noise. This is particularly noticeable in darker images. This noise can be significantly reduced by applying a black-level image correction. For the most recent 86 videos, this correction was applied internally using the camera's PFV4 software. Some short sets from certain recordings were manually segmented and labeled. More importantly, standalone images were segmented to create the training set. The processed images, along with the necessary JupyterLab scripts, are available in the GitHub repository [40]. Each JupyterLab script includes a description of its purpose and the operations it performs.

To aid the manual segmentation process, we applied several custom scripts to generate clearer visualizations of the original images. To address the fixed pattern noise and vignetting, we applied flat-field corrections to the images. Additionally, we generated a representation with the background subtracted for better contrast during segmentation, as illustrated in Figure 11b. The background image was created through a multi-step process using all frames from the video. First, at each pixel location, a median filter was applied across the frames to remove outliers caused by moving bubbles, effectively isolating the static features of the scene. Then, at each pixel position, the mode of the remaining pixel intensities was calculated, identifying the most common intensity value. This final step ensured that the background image represented the most persistent features of the video.

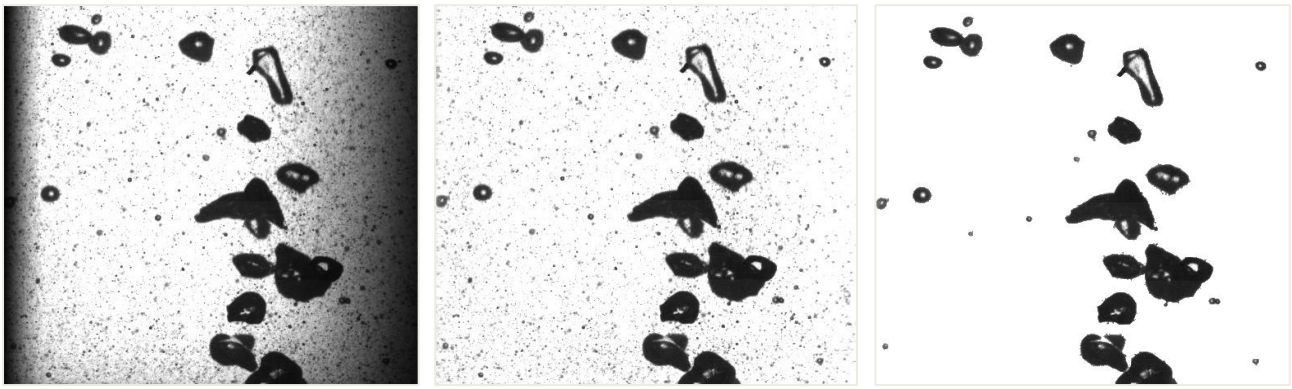

Figure 11. *(a) Original image showing numerous small, noisy bubbles. (b) Enhanced contrast after flat-field and background subtraction. (c) Processed image with erosion steps to highlight significant bubbles for segmentation.*

For images captured with additional pumped flow, numerous small, noisy bubbles are visible (see Figure 11a). Choosing what to include and exclude for segmentation is difficult. Therefore, we generated an additional visualization by applying Otsu's thresholding to the flat-fielded, background-subtracted image, producing a bitmask. This mask was then subjected to two erosion steps to remove small spherical bubbles. After repairing the remaining segments, the updated mask was applied to the original image to emphasize the remaining features (see Figure 11c). Such visualizations can be used to assist in identifying the bubbles that should be segmented.

The choice of two iterations is based on the analysis shown in Figure 12a, which illustrates the accuracy of the area of circles with subpixel diameters and varying center locations as captured by a camera. The accuracy is determined by dividing the pixel-covered area by the actual geometric area of the circle. The graph considers all possible subpixel center positions for each discrete diameter, using a step size of 0.1 subpixels. Examples of such bitmasks are shown in Figure 12b and Figure 12c, where green represents activated sensor pixels and yellow represents inactive ones. Objects that are completely eroded within two iterations are deemed too unreliable for meaningful analysis. The two erosion steps use a cross-shaped 3x3 kernel. For further details, see the corresponding code in the GitHub repository [40], where a comprehensive analysis is conducted on a similarly noisy image.

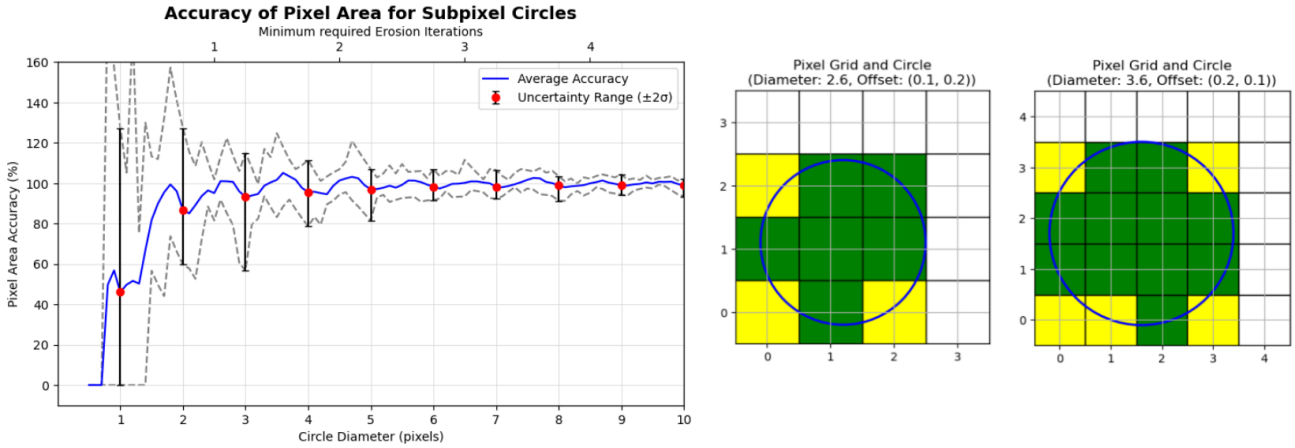

Figure 12. *Analysis of the subpixel circle accuracy area as captured by a camera across various diameters and offsets. (a): The graph compares the pixel-covered area to the true geometric area for circles of varying diameters and subpixel center positions. The uncertainty (±2σ) reflects variations in subpixel offsets. (b, c): Visual examples show how subpixel circles (blue outlines) are imaged by the pixel grid. Green pixels are active, while yellow pixels are intersected by the circle but are considered inactive.*

4.1 Segmentation and Labeling

We developed a custom labeling tool in Python to address the need for segmenting and labeling bubbles across entire sequences, rather than limiting the process to individual images. Creating our own program also enabled us to incorporate tailored image processing techniques. The user interface, illustrated in Figure 13, displays multiple segmentation methods in the top row.

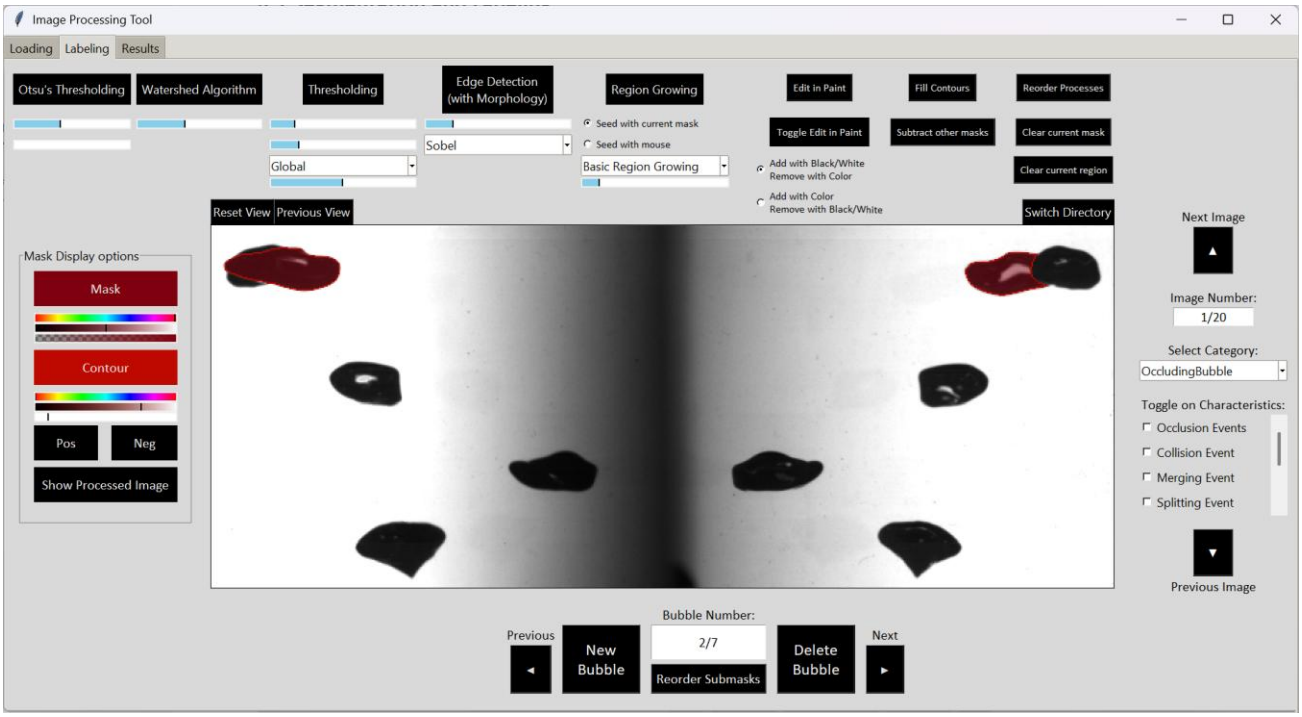

Figure 13. *User interface of the custom bubble segmentation tool for sequences, featuring multiple image processing techniques such as Otsu's Thresholding, Watershed Algorithm, and Edge Detection, accessible via the top toolbar.*

The front and back sides of the captured bubbles are segmented together as a single image and separated afterward. The masks for each bubble in each image are stored in a folder structure where the folder names match the names of the processed images. For each bitmask image, a corresponding .json file is generated, containing the label and additional characteristics for further analysis. After segmentation and labeling, a JupyterLab code is used to convert the folders into a unified .json file for the training and test sets. This file also includes supplementary information, such as roundness, mask width, and mean image intensity, to support further analysis.

4.2 Shape Recovery

Once both sides are segmented, the shape of occluded bubble parts can be reconstructed if they are visible on the opposite side. To illustrate our method, we present a front and backside image from which parts of the bitmasks are intentionally removed. To identify corresponding points in the two images, one image is flipped to simulate binocular vision. Since the front and back sides of the bubbles appear similar because of their transparency, this approach is effective. Each bubble pair is isolated by applying the corresponding masks to both images. We apply Feature Detection and Description algorithms, commonly used in image stitching, to match corresponding points for the incomplete bubble. Although standard methods such as SIFT, SURF, and ORB were tested, identifying keypoints with Harris Corner Detection yielded significantly more reliable matches. These keypoints are then described using the Scale-Invariant Feature Transform (SIFT) to generate robust descriptors. This approach is well-suited to the task since Harris Corner Detection excels at detecting contours like those on the outlines of the bubble, which are the most stable features for matching.

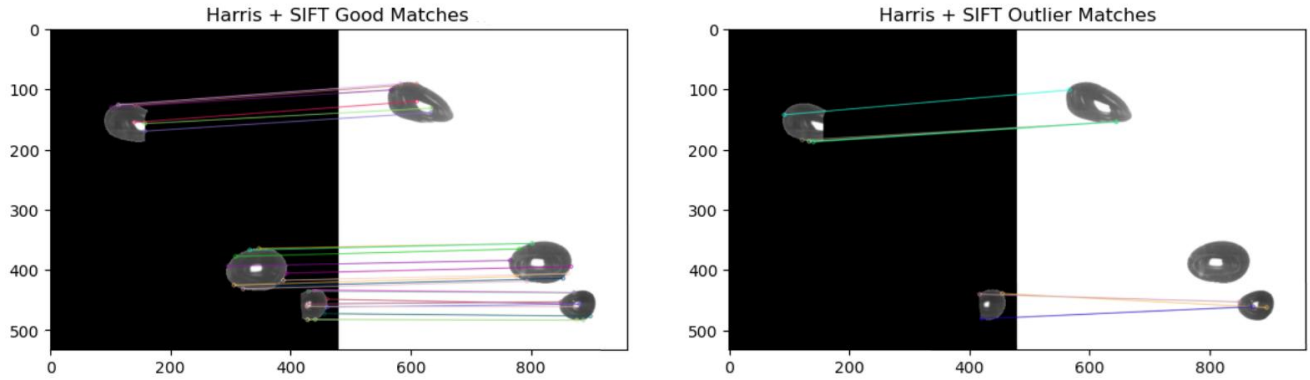

Figure 14. *(a) Successful feature matches between bubble images using Harris Corner Detection and SIFT. (b) The outlier matches that are considered incorrect.*

If we draw lines between the resulting matching points, as shown in Figure 14, we are able to filter out mismatches based on deviations in angle and length. While more advanced clustering methods could be employed, using thresholds has proven both effective and sufficient for this purpose. However, as demonstrated in Figure 14, good matches may occasionally be filtered out. This is not problematic as long as enough valid matches remain. From the filtered matching points, we compute transformation matrices using *partial affine estimation*. This transformation allows us to warp one mask to match the perspective of the other. Figure 15 illustrates a pixel-wise comparison between the original mask and the transformed mask. With these completed shapes, the bubble centers can be accurately estimated.

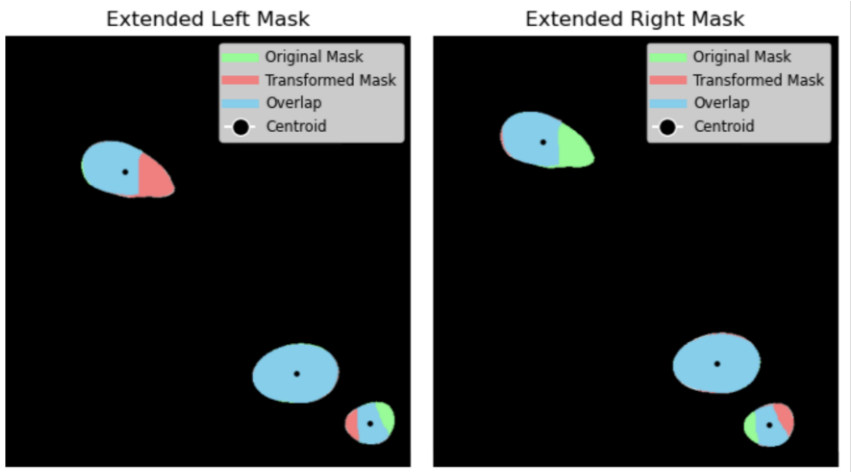

Figure 15. *Pixel-wise comparison of original and transformed bubble masks for both the (a) left side and (b) right side.*

In Figure 16, both the original and transformed masks are converted into star-convex polygons to identify the regions that require reparation. These regions are determined by detecting the angles where the transformed polygon's radii are bigger and where the squared error between the radii of the original and transformed polygons exceeds a specific threshold. The yellow marked regions in the second and third plots of Figure 16 fulfill these two conditions. These ranges of angles are refined based on the squared errors of the gradient of the radii, shown in the fourth plot of Figure 16. In

this graph, we search near the initial borders for the point where the values first cross a specified threshold and remain below it for at least 20 consecutive points while the squared error of the radii remains below its mean squared error. This method effectively identifies where the original contour ceases to follow the contour of the transformed polygon. This is where the gradient of the squared error becomes flat. The regions that are visually highlighted in yellow on the plots in Figure 16 are clearly marking the areas of significant deviation.

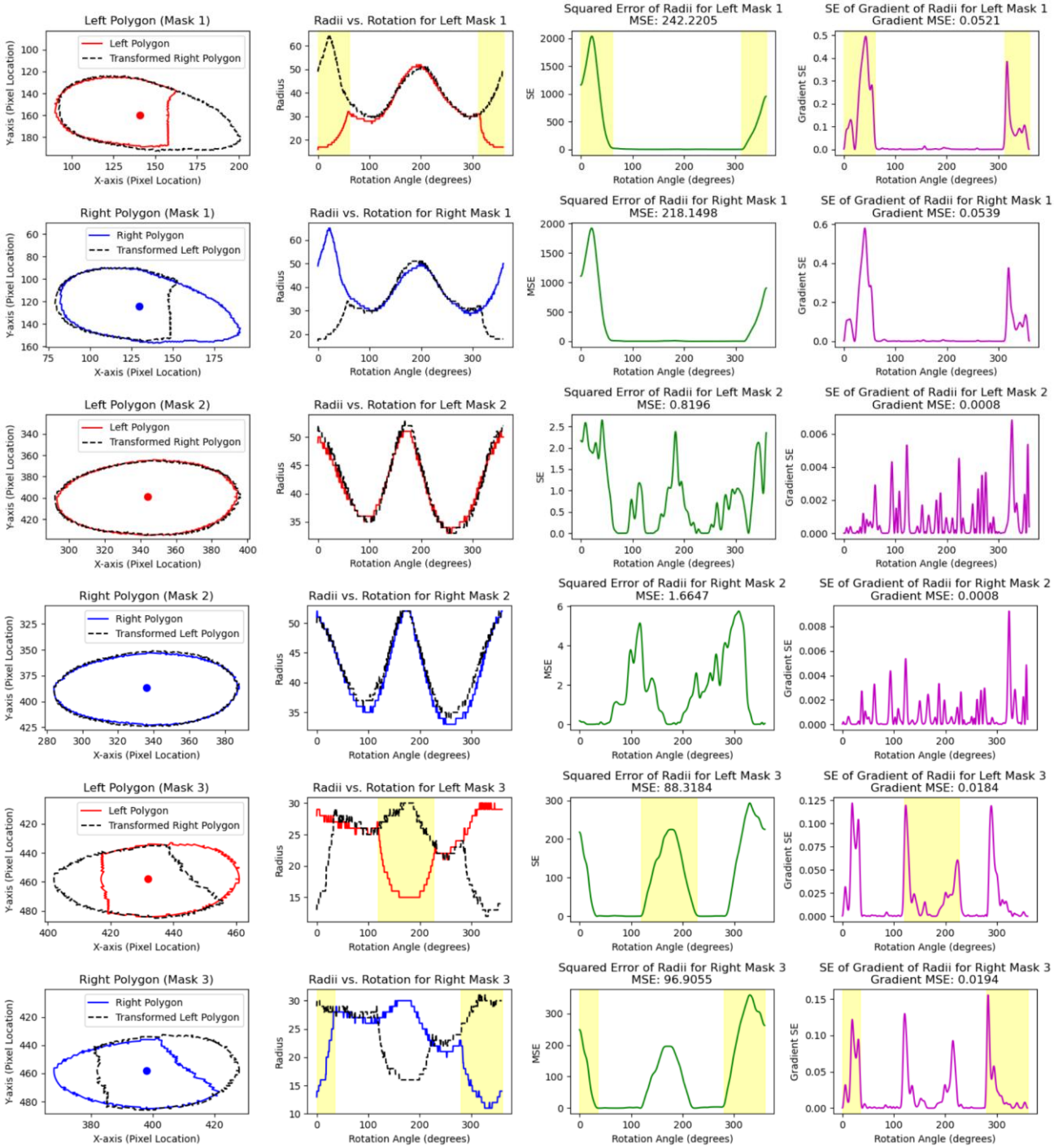

Figure 16. *Analysis of shape for three pairs of masks. The first column shows the original and transformed polygon shapes. The second column presents the radii of the polygons relative to their centers, measured clockwise starting from the horizontal axis. The third column illustrates squared error of radii, and the fourth column shows the gradient of the squared error. These plots help identify regions where significant deviations occur, marked in yellow, aiding in the accurate reconstruction of bubble contours.*

To repair the masks in the marked regions with smooth transitions, we compute scaling factors at the start and end of each region to match the transformed radii to the original lengths at these points. We then calculate linear scaling functions that interpolate between these factors, gradually adjusting the transformed radii across the region. This approach ensures a smooth transition from the original to the adjusted values, preventing abrupt changes that could result in unrealistic shapes. See Figure 17 for the results of such repair for the masks of the three previously selected regions.

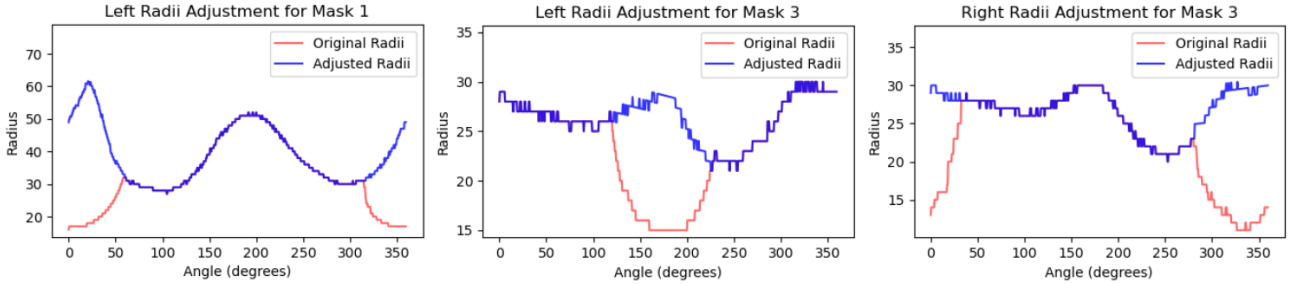

Figure 17. Adjusted radii for three regions. Each graph compares the original radii of the incomplete contour (red) to the adjusted radii of the completed contour (blue). The adjustments create smooth transitions between the existing and reconstructed parts.

The original mask can be extended by creating and filling the specific region of the polygon contour. The repaired masks, shown in Figure 17, achieve Intersection over Union (IoU) values of 0.95, 0.98, and 0.98, respectively. For exclusively the repaired regions, the IoU values are 0.84, 0.89, and 0.92. The bubble image that is used for this example was captured at an early stage and has a greater misalignment than typical. In tests using a typical image captured with our setup, IoU values above 0.9 for repaired regions are commonly achieved. However, the results are highly dependent on obtaining accurate matching points, which becomes more challenging for bubbles represented by fewer pixels. Consequently, human supervision remains essential.

4.3 Depth Recovery

Although the misalignment is considered a disadvantage for the accuracy of the bubble reconstruction, it enables the possibility of depth recovery. In typical stereo vision, a larger viewing angle improves the accuracy of depth estimation by allowing more precise triangulation based on horizontal disparities. For the bubbles, this should be done with respect to the center of the complete contour. Triangulation becomes particularly challenging when dealing with relatively small angles and limited resolution. Therefore, we have validated our approach with an image of an object with a known geometry. Figure 18a shows a plate lowered into the column, featuring a central bolt and two square bolt patterns measuring 30x30 mm and 70x70 mm, designed for use with and without a teleconverter, respectively. Figure 18b shows an image captured by the camera using the 2x teleconverter. Similar images were captured for most of the sessions.

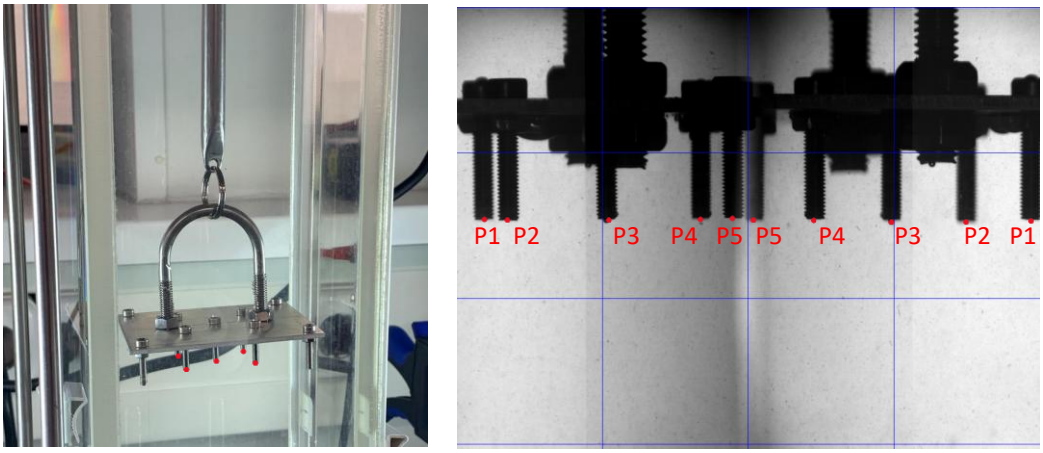

Figure 18. *Test for depth recovery. (a) Shows a plate with a configuration of bolts of known dimensions placed into the bubble column. (b) Shows the Image that is captured with a 2x teleconverter.*

By utilizing the image with the protractor triangles, as previously shown in Figure 9b, the angles of both cameras can be calculated. By incorporating the distance from the camera to the center of the bubble column, we can establish a configuration equivalent to one without mirrors, as illustrated in Figure 19. With the camera's intrinsic parameters (K matrices), the relative positions of the cameras defined by the rotation (R) and translation (t) matrices, and the 2D matching points, the real-world 3D coordinates can be accurately computed through standard triangulation.

Figure 18b has been separated into a left and right side. It is important to note that we mirrored the coordinates vertically for both images. Since the setup includes three mirrors, the two image sections shown in Figure 9b and Figure 18b are actually mirrored views. The result of the triangulation is presented in Figure 20a. More details can be found in the corresponding JupyterLab script available on the GitHub repository [40].

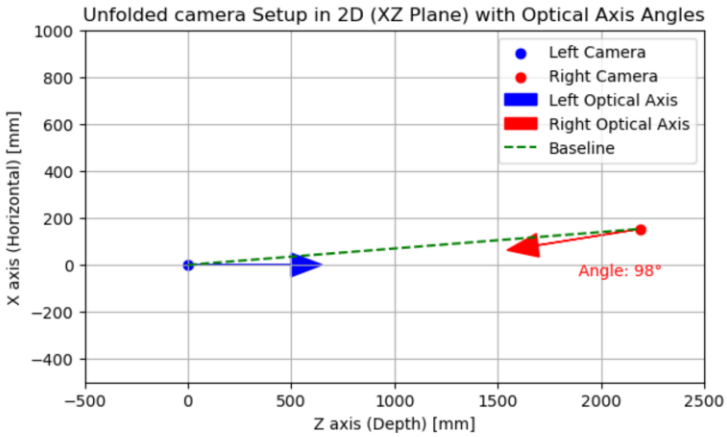

Figure 19. *Camera setup in the 2D XZ plane illustrating the optical axis angles of both cameras relative to the baseline, simulating a mirror-free configuration. This setup enables triangulation for 3D coordinate reconstruction.*

The results shown in Figure 20a reveal inaccuracies in the depth, which is the dimension we aim to uncover. These errors are likely caused by inaccuracies in the parameters used to compute the camera positions, specifically the camera angles and distances. Additionally, cumulative errors can arise from factors such as the placement of the protractor triangles, the allocation of pixel coordinates, and minor inaccuracies in the test object itself. Due to the small misalignment, even slight deviations in the disparities have a significant impact on the depth estimation. Furthermore, the individual distances from the center of the column to the cameras are estimated, as the use of multiple mirrors prevents direct measurement along a straight line.

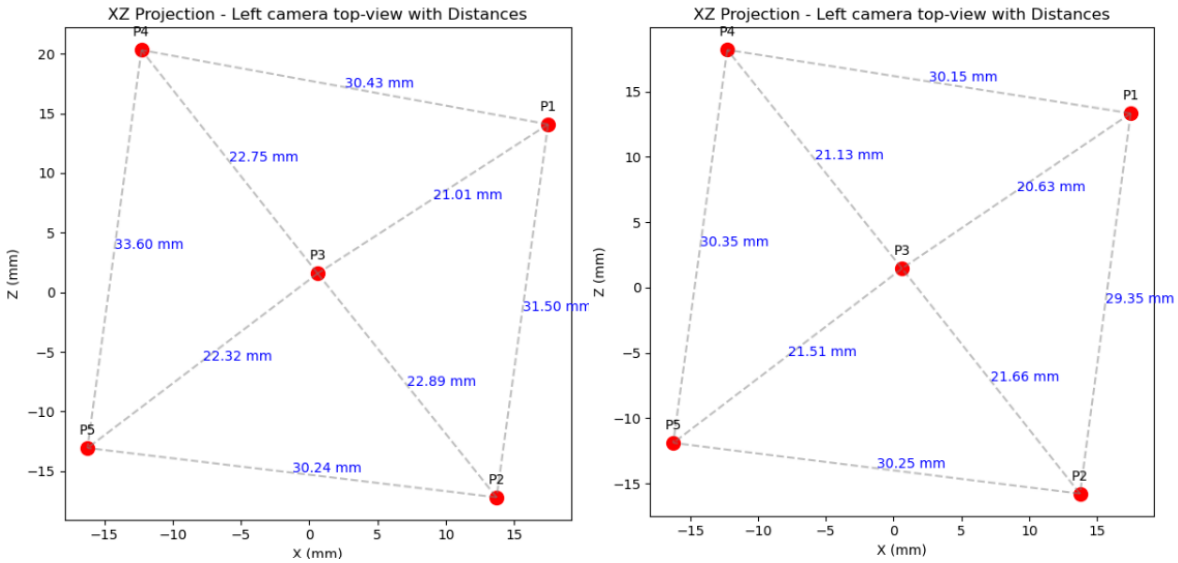

Figure 20. *XZ projection of the left camera's top-view with distances between key points that should match a 30x30 mm square. (a) Initial results show depth deviations caused by minor errors in camera parameters and test setup alignment. (b) Corrected results after applying an optical angle adjustment.*

For the bubble segments, the X-coordinates are not of primary interest, as they can be extracted directly from the mask. Once calibrated with an object of known geometry, the depth can be computed with reasonable accuracy. An optical angle correction of +0.8 degrees was applied to obtain the result shown in Figure 20b. Upon reinspection of the tool, we discovered that the residual skewness still visible in Figure 20b was also present in the tool itself. While our setup may not provide the precision required for all research applications, it is sufficiently accurate for depth estimation alongside our segmentation task.

5 CNN model evaluation

In this chapter, we test depth awareness in bubble segmentation by categorizing bubbles as singular, occluding, occluded, and occluding-occluded using Mask R-CNN and SOLOv2. We then evaluate their segmentation performance together with that of YOLAct++. The CNNs used in this chapter are detailed in Table 1. The backbones are selected based on comparable inference times, as documented in the GitHub repositories of the respective models. We will utilize the dataset from HessenKemper et al. [5], complemented by our own data. Of this combined set, 80% is allocated for model training and the remaining 20% is reserved for validation. Additionally, we selected and set aside a small subset of images from the experimental data of Hessenkemper et al. These images, differing from the main training and validation data, are specifically intended to test the models' robustness on out-of-distribution data.

Table 1. *Convolutional Neural Networks that are evaluated in this chapter.*

Model	*Inference Time*	*Backbone*	*Method*
Mask R-CNN	55 ms	ResNet50-FPN	Traditional Region-based
SOLOv2	54 ms	ResNet50-FPN	Grid-based
YOLAct++	42 ms	ResNet101-FPN	Coefficient Fitting

5.1 Metrics

The performance of segmentation models is often evaluated using the *Average Precision* (AP) metric, which highlights the trade-off between *Precision* and *Recall*. This performance score can be computed for various *Intersection over Union* (IoU) thresholds:

$$\text{IoU} = \frac{|P \cap G|}{|P \cup G|} = \frac{\text{Area of Overlap}}{\text{Area of Union}}, \tag{4.1}$$

where P represents the 1-pixels of the predicted mask and G those of the ground truth. The IoU quantifies the pixel-based overlap between the predicted segmentation mask and the ground truth mask. A certain IoU can be assigned as a threshold for what would be a correct prediction for the computation of the Precision and Recall:

$$\text{Precision} = \frac{TP}{TP + FP}, \tag{4.2}$$

$$\text{Recall} = \frac{TP}{TP + FN}. \tag{4.3}$$

In these equations, TP stands for True Positives, representing correctly predicted regions that meet the IoU threshold. FP denotes False Positives, which are incorrectly predicted regions that do not match the ground truth. The FN stands for False Negatives, referring to ground truth regions that were missed by the model.

The output of the Softmax layer provides values between 0 and 1, which act as confidence scores. To compute precision as a function of recall, confidence thresholds are used that must be met for a prediction to be considered. The model's *Precision* (the accuracy of the predictions) and recall (the proportion of actual positives correctly identified) are calculated at each threshold. As the confidence threshold is lowered, recall increases because more predictions are considered, but precision typically decreases due to the rise in false positives. This results in a curve, called the Precision-Recall curve, that typically starts high in precision at low recall levels and declines as recall increases, illustrating the trade-off between these two metrics. The AP is computed as the area under the Precision-Recall curve at a specific IoU threshold (e.g. IoU=0.5). Generally this is done in the discrete form:

$$\text{AP}_{\text{IoU}=0.5} = \frac{1}{N}\sum_{i=1}^{N} \text{Precision}\left(\text{Recall}_i\right), \tag{4.4}$$

where Precision is a function of a Recall value at point i, and N is the total number of recall points. The term *Mean Average Precision* (mAP) refers to the average AP over a range of IoU thresholds (e.g., from 0.50 to 0.95 in steps of 0.05). These metrics form the core of the COCO evaluation framework, widely used for benchmarking object detection models.

5.2 Depth-aware labels

A crucial aspect of our proposed design is the determination of the order of depth of crossing contours. It is yet to be explored if the relationship between overlapping bubbles can be accurately predicted and if this enhances segmentation performance. In Figure 21a, we show the Average Precision (AP) of the set with and without our label strategy for the Mask R-CNN and SOLOv2. Adding more labels results in a lower Average Precision because the model must learn to distinguish between a greater number of categories. In addition, the results of the unseen out-of-distribution set can be observed in Figure 21b.

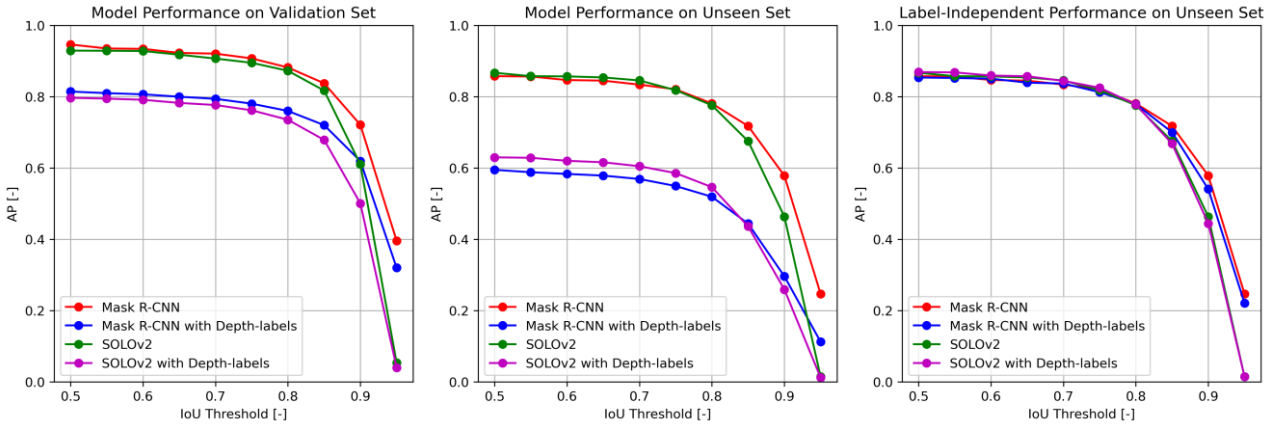

Figure 21. *Performance Analysis of Depth-aware Labeling on CNN Models: (a) AP comparison on validation set with and without depth labels for Mask R-CNN and SOLOv2. (b) Performance on unseen datasets, emphasizing the model robustness (c) Performance on unseen datasets excluding false categorization.*

The AP curves in Figure 21a indicate how well our extended labeled CNNs execute the depth-aware categorization compared to those models trained for a single category, namely *Bubble*. The AP in Figure 21b does the same for an unseen set. Moreover, Figure 21b indicates the extent to which the model generalizes rather than merely memorizing the training data. The height of the AP curves in both graphs indicate that the models are reasonably able to recover the depth-related categories. However, to find out if these distinctions improve the segmentation tasks we should not account for false categorization. We are after all most interested in how well bubbles are detected and captured. Therefore, we excluded the false labeling for the AP computation in Figure 21c. Those AP curves show that the performance in segmenting is not notably affected by the depth-related categories.

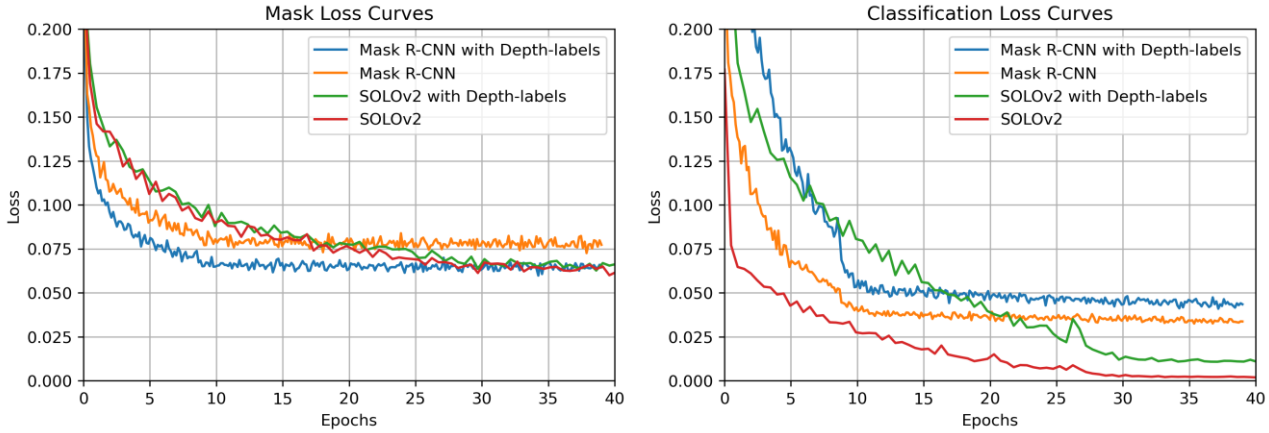

Figure 22. *Training Loss Curves of the Mask R-CNN and SOLOv2 with one label vs. depth-aware labels for (a) the segmentation and (b) the classification.*

Arpit et al. [7], demonstrated that a CNN designed for classification tasks converges significantly faster during training when using actual categories as opposed to random categories. Although the minima are of equal quality, the network trained on meaningful labels tends to generalize, whereas the one trained on random labels resorts to memorization. This suggests that when labels carry inherent and coherent meaning, the model is more likely to generalize rather than memorize. Modern CNNs, which are trained on vast amounts of data, are able to uncover the fundamental principles that define each category. The high AP curves for unseen data as observed in Figure 21b, together with the fast-converging training losses in Figure 22, indicate that the depth-related labels promote generalization rather than memorization.

5.3 Comparison on inference

The part that makes models such as the Mask R-CNN, SOLOv2 and YOLAct++ unique typically lies in the network structure after the backbone. Generally, a larger backbone improves performance, if there is enough training data to support the extended model. The backbone is the kick-off module of the architecture, used to extract and encode features from the data given by the input layer. In the early layers, it filters for core (low) features like edges and corners, which are combined into higher-level features like shapes the further you advance in the layers. The CNN models we consider are published with different backbones to match user requirements and preferences. Because YOLAct++ is a very lightweight model compared to the others, we use a larger backbone, as noted in Table 1. However, this deeper backbone did not result in improved performance compared to the other lightweight models, as shown in Table 2. In this table the performance of the trained models is summarized. The weights of the models are shared in the GitHub repository [40].

Table 2. *The mAP and IoU scores for all evaluated models, with IoU calculated as the mean of combined instance masks per image.*

Model	*Categories*	*mAP of Evaluation Set*	*mAP of Out-of-Distribution Set*	*Image-based IoU of Evaluation Set*	*Image-based IoU of Out-of-Distribution Set*
Mask R-CNN	1	0.840	0.739	0.949	0.928
Mask R-CNN	4	0.723	0.484	0.950	0.928
SOLOv2	1	0.786	0.703	0.925	0.905
SOLOv2	4	0.666	0.494	0.929	0.910
YOLAct++	4	0.160	0.128	0.729	0.699

Figure 23a-c presents typical inference results from the trained Mask R-CNN, SOLOv2, and YOLAct++, respectively. The segmented image is taken from the validation set. The hierarchy of performance is evident from the quality of the generated masks. The differences in categorization performance and mask quality, as depicted in Table 2, are clearly reflected in these images. Additional inference results, including examples from both the validation and out-of-distribution sets, are provided in Appendix A.

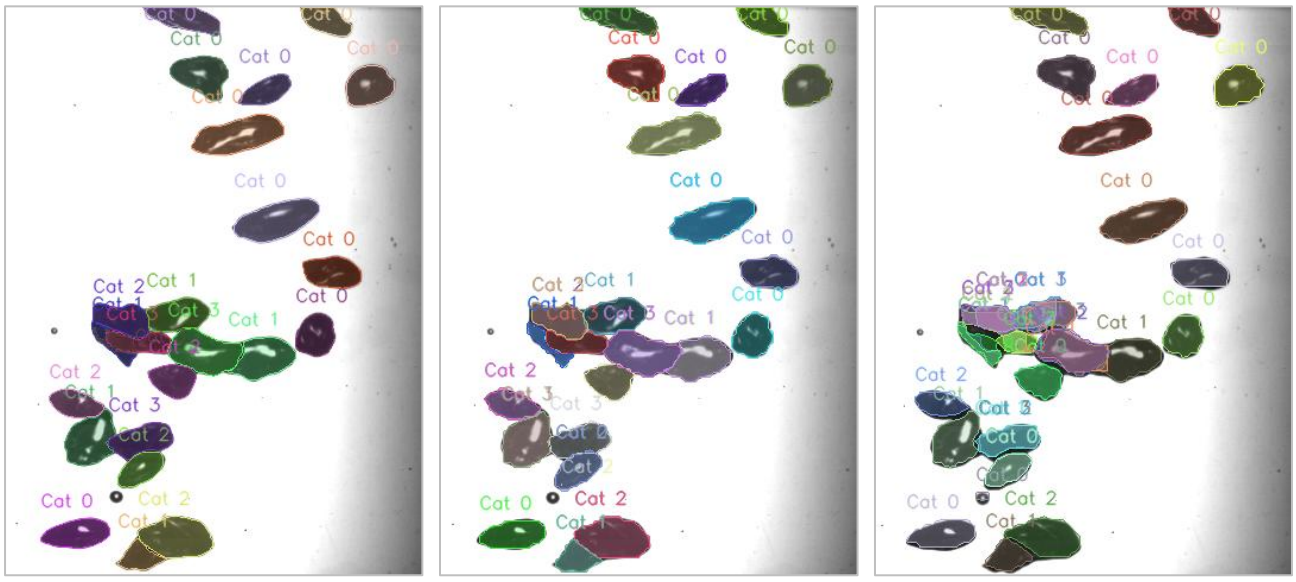

Figure 23. Segmentation results for an image of the validation set of the (a) Mask R-CNN, (b) SOLOv2, and (c) YOLAct++. The bubbles are categorized into four classes: singular [0], occluded [1], occluding [2], and occluding-occluded [3].

6 Discussion

The specific objective of this thesis is to achieve more accurate segmentation of non-spherical, less predictable bubble shapes in dense bubble flows, as the segmentation of isolated bubbles generally poses no particular difficulty. Aggregating detected features into complete bubble instances is essential. However, CNNs tend to memorize the training set when faced with less predictable spatial arrangements caused by occlusions and light artifacts. Therefore, we consider performance on the unseen set particularly important. Based on our analysis, Mask R-CNN emerged as the top-performing model in our lightweight tests, demonstrating efficiency for both in-distribution and out-of-distribution data. This success can be attributed primarily to its superior mask generation capabilities. SOLOv2, on the other hand, performed comparably in detection and classification across both seen and unseen data, making it a strong competitor. In contrast, YOLAct++ proved unsuitable for bubble segmentation and particularly our classification tasks, delivering subpar performance in both areas.

As shown in the previous chapter, the Mask R-CNN and SOLOv2 both demonstrate the ability to categorize depth-aware labels. Determining which bubble is occluding and which is occluded is most apparent from the continuation of the contours at the intersection. Consequently, label mismatches are often accompanied by poor mask quality, as can be observed in Figure 23. The comparison of loss curves in Figure 22 of the Mask R-CNN and the SOLOv2 has shown that the inclusion of depth-aware labels required the models to learn more complex relationships. However, the additional labels do not appear to affect the mask quality beyond the variation normally observed across training sessions, as the IoU values in Table 2 are comparable for both the evaluation and the Out-of-Distribution (OoD).

By utilizing stereo imaging, we developed a method to reconstruct the contour segments visible in the opposite image, even in the presence of a more challenging misalignment in the exemplary image in Chapter 4. The high IoU of 0.9 for reconstructed parts demonstrates the potential of this method to accurately recover the occluded form. Consequently, this allows for more precise center estimation, which facilitates more accurate depth tracking through triangulation.

The submillimeter precision achieved in recovering the dimensions of the test object, as presented in Chapter 4.3, exceeded our expectations. Depth recovery in binocular stereo vision is highly sensitive to errors at small viewing angles and low resolutions. However, triangulation at a 170-degree angle yields significantly more accurate depth estimates compared to a 10-degree setup. This follows directly from the Stereo Vision Depth Formula for 2D parallel cameras with equal focal lengths:

$$Z = \frac{f \cdot B}{x_l - x_r} \tag{6.1}$$

In this formula, the depth (Z) is computed using the focal length (f), the baseline (B), and the disparity ($x_l - x_r$), where x_l and x_r represent the offsets from the optical axis of each camera. A visual interpretation is given in Figure 24. At a baseline of zero (or zero-degree angle), no disparity exists for any coordinate, and depth cannot be calculated. By increasing the baseline, the disparity increases steadily. Similarly, increasing the angle between the cameras also amplifies the disparity of coordinates, reaching its maximum at 180 degrees. At this point, any coordinate on an intermediate plane appears as mirrored in the opposing view. Except for points on the coincident optical axis, where the disparity is zero, each depth corresponds to a unique disparity, unlike in the zero-degree setup. In other words, in contrast to the zero-degree configuration, moving an object between the opposing cameras causes it to increase in size in one view while decreasing in the other. The 10-degree misalignment in the 180-degree setup is therefore not equivalent to a 10-degree misalignment in a zero-degree setup. Due to the larger disparities at wider angles, errors in pixel position result in minor changes in the calculated depth. Therefore, the 170-degree configuration offers a distinct advantage over a 10-degree setup, achieving highly accurate depth recovery even at low resolution.

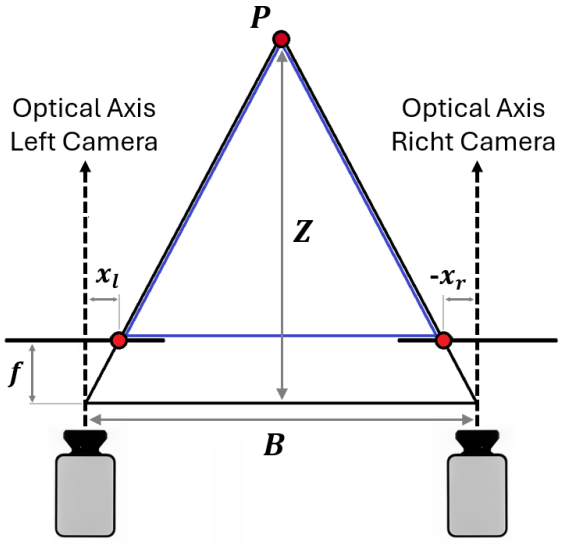

Figure 24. *Schematic representation of parallel binocular stereo vision. Two cameras, separated by a baseline (B), capture the projection of a point (P) onto their image planes at coordinates x_l and x_r. The disparity (x_l - x_r), along with the focal length (f) and the baseline (B) are used to calculate the depth (Z) of point P relative to the cameras using the Stereo Vision Depth Formula.*

Several factors related to the training process, dataset, and experimental setup have influenced the results of this study. The most relevant aspect and their implications are outlined below:

- **Training:** The randomness inherent in the stochastic gradient descent (SGD) process, including minibatch selection, results in variability in converging to minima in the loss landscape across training sessions. This variability may lead to better or worse outcomes depending on how the search space is explored. As highlighted by Arpit et al. [7], memorization and generalization in deep neural networks (DNNs) are strongly influenced by the network architecture, optimization procedure, and training data. In this study, optimization of the training process was not a focus; instead, the training processes of existing models were adopted. Given the computational constraints, limited data, and the scope of this thesis, pretrained weights were employed through transfer learning. This approach is widely recognized for its efficiency in weight initialization and reduced training time, making it a practical and effective starting point for the experiments conducted in this study.

- **Dataset:** Despite efforts to introduce diversity, the experimental data were captured using a single prime lens, with only minimal variation in field of view and depth of field. Additionally, the experiments were limited to a single water–air flow condition, further reducing the variability of the data. The segmentation of the experimental data involved manual assistance, making the masks subject to interpretation. Furthermore, limited bubble density and the small quantity of labeled data posed challenges, necessitating the supplementation of synthetic datasets

- **Experimental setup:** Unforeseen misalignment in the stereo imaging setup introduced unintended perspective distortions, adversely affecting shape recovery and depth estimation. The skewness of the aluminum frame further disrupted the mirror alignment, as visible in the slight tilt of the protractor triangles between the left and right sides in Figure 9b. Additionally, the lack of known dimensions led to a reliance on estimations, which further constrained the accuracy of triangulation and resulted in a strong dependence on calibration.

7 Conclusion

In this study, a new method is presented to recover occluded contours for the tracking of non-spherical bubbles within bubbly flows. By employing an opposing stereo camera setup, we reconstructed occluded bubble shapes by capturing their contours from the opposite side. With this reconstruction, the bubble center can be computed in a manner similar to that of non-occluded bubbles. We demonstrated that the 3D positions of the centers can be recovered through triangulation. Even with a small misalignment and a relatively low resolution, depth recovery by our setup achieves a reasonable level of precision after calibration. A substantial dataset was acquired with this setup, consisting of air-water bubbly flows under various mass flow rates and lighting conditions. Additionally, the data includes induced liquid flows, such as swirling flows, and surfactant-supplemented bubbly flows. To process the data, a Python program was developed to segment bubbles in both individual images and image sequences. The processed images, along with the program, are shared in the GitHub repository for reference and reuse.

In addition to the commonly used Mask R-CNN, we have considered SOLOv2 and YOLAct++ for bubble segmentation. To enable a fair comparison, we chose backbones such that the resulting inference times across the models were similar. As anticipated, all models successfully captured individual closed bubbles. However, they encountered challenges with unclosed contours and overlapping bubbles in dense flows. Of the three models, the Mask R-CNN achieved the most accurate segmentations on both the validation and out-of-distribution sets. The SOLOv2, however, delivered competitive results in both segmentation and classification on these datasets. As a follow-up to this research, we propose enhancing one of these two CNNs by incorporating ConvLSTM layers within the bottleneck. This modification would leverage temporal relationships to achieve more consistent feature aggregation over time, as discussed in Chapter 2.

In general, our study demonstrates that CNNs can recover the hierarchy of bubbles in cases of occlusion. With masks and relative depth as outputs, the CNN provides crucial inputs for predicting unseen shapes based on previous motion. We discussed a potential method for predicting the motion of previously unoccluded contours using an LSTM-based Graph Neural Network, developed by Lin et al. [31]. For subsequent data, bubble shapes that become occluded can be inferred as nodal sheets sliding either in front of or behind one another.

Instance segmentation by Convolutional Neural Networks involves a wide variety of methods and has been extensively applied and explored in research. For the specific challenges of segmenting bubbly flows, many opportunities remain unexplored. The following recommendations are based on the limitations discussed in Chapter 6, the proposed design in Chapter 2, and our vision for future improvement and research.

- **Shape repair**: The method we developed for shape repair is a valid approach to recovering the shape, but errors can occur during matching, warping, and radius smoothing. Matching methods, clustering techniques, and smoothing operations can be further explored. Because of the recovered 3D positions of individual bubbles, you could stack singular bubbles on top of each other from one view without creating spatially impossible sequences. By mimicking occlusion in this way, you could train a Dual-Input CNN to perform occluded shape repair instead of relying on our method.

- **Depth categories:** Further research is needed to determine whether CNNs have the ability to recover discrete or continuous depth from the defocus of bubbles. It is worth investigating whether an included or aggregated depth estimation task enhances the CNN's segmentation capabilities. For the proposed LSTM-based GNN, such output could allow the nodal sheets to account for out-of-plane motion as well.

- **ConvLSTM-layers and LSTM-based GNN:** The directions taken in the thesis were motivated by the proposed design presented in Chapter 2. Implementing ConvLSTM layers in the bottleneck, or training an LSTM-based GNN to recover occluded forms through motion, will require significantly larger datasets comprising image sequences rather than single frames. We developed a tool for the manual segmentation of sequences, but manual annotation remains a cumbersome and labor-intensive process. Nonetheless, we strongly recommend further exploration of these methods, as they show great potential for efficient inference. In addition to segmenting experimental data, we recommend extending the training set with synthetic sequences featuring denser flow. For the training of the ConvLSTM layers, it is paramount that the feature aggregation is difficult, which could be achieved by stacking singular bubbles or employing simulation-based approaches.

- **Synthetic data:** The impact of combining synthetic and experimental data should be examined in greater detail. Kim and Park observed that an experimental dataset predominates in enhancing the ability to recognize the actual bubble shapes, while adding synthetic bubble images to the training set improves the ability to separate overlapping bubbles. Some efforts to generate synthetic data were undertaken during the thesis, but these efforts have been abandoned in favor of the experimental data. A few 3D OpenFOAM simulations have been executed with the solver InterIsoFoam to accurately capture the interface between fluids. We applied extensive Dynamic Mesh Refinement near the interface in order to keep the mesh coarse in all undisturbed regions. Although the results were promising, the computational needs were a dealbreaker. Nonetheless, it is still worth mentioning for any subsequent project. To better distribute the computational effort, Dynamic Load Balancing (DLB) should be employed, and the simulation scope should be minimized as much as possible. With an application like Blender, a lighting setup can be applied to render shadowgraphy-like images of the CFD results after smoothing the mesh interfaces, similar to the approach of Koch et al. [29]. The use of simulation data can even enable subpixel edge detection.

- **Optimization:** Detailed optimization processes were beyond the scope of this project. However, specific areas worth investigating include the effects of freezing layers, data augmentation strategies, hyperparameter tuning, and preprocessing techniques such as a flat-field correction. These efforts could significantly enhance the performance and generalization capabilities of the neural network. Additionally, we encourage future research to focus on analyzing results to better understand the challenges the CNNs encounter during inference. This understanding could help guide further refinements of the architecture and training strategies.

We hope that the findings and recommendations in this study inspire new initiatives and advancements in bubble segmentation, shape repair, and tracking individual bubbles within bubbly flows. Additionally, we encourage further validation and refinement of the concepts, methods, and experimental setup presented in this report. Segmentation of bubbly flow remains a significant challenge in related studies. Through this work, we aimed to contribute to the groundwork for autonomous processing, which has the potential to open new pathways for research and innovation in fluid mechanics and related fields.

8 References

[1] R. Clift, J. R. Grace and M. E. Weber, Bubbles, Drops and Particles, 1978, pp. 169-173.

[2] M. K. Tripathi, K. C. Sahu and R. Govindarajan, “Dynamics of an initially spherical bubble rising in quiescent liquid,” *Nature Communications,* pp. 1-9, 17 februari 2015.

[3] Y. Cui, C. Li, W. Zhang, X. Ning, X. Shi, J. Gao and X. Lan, “A deep learning-based image processing method for bubble detection, segmentation, and shape reconstruction in high gas holdup sub-millimeter bubbly flows,” 2022.

[4] S. Zhang, H. Li, K. Wang and T. Qiu, “Accelerating intelligent microfluidic image processing with transfer deep learning: A microchannel droplet/bubble breakup case study,” 2023.

[5] H. Hessenkemper, S. Starke, Y. Atassi, T. Ziegenhein and D. Lucas, “Bubble identification from images with machine learning methods,” 2022.

[6] H. Zhang, J. Li, N. Sun, H. Li and Q. Hang, “Multi-input residual convolution neural network for three-dimensional reconstruction of bubble flows from light field images,” 2024.

[7] D. Arpit, S. Jastrzebski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio and S. Lacoste-Julien, “A Closer Look at Memorization in Deep Networks,” 2017.

[8] I. Goodfellow, Y. Bengio and A. Courville, Deep Learning, MIT Press, 2016.

[9] K. He, G. Gkioxari, P. Dollár and R. Girshick, “Mask R-CNN,” in *In Proceedings of the IEEE International Conference on Computer*, 2017.

[10] Y. Kim and H. Park, “Deep learning-based automated and universal bubble detection and mask extraction in complex two-phase fows,” 2021.

[11] I. Malakhov, A. Seredkin, A. Chernyavskiy, V. Serdyukov, R. Mullyadzanov and A. Surtaev, “Deep learning segmentation to analyze bubble dynamics and heat transfer during boiling at various pressures,” 2023.

[12] O. Ronneberger, P. Fischer and T. Brox, “U-Net: Convolutional Networks for Biomedical,” in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2015.

[13] R. Sharma, M. Saqib, C. T. Lin and M. Blumenstein, “A Survey on Object Instance Segmentation,” *SN Computer Science,* 2022.

[14] D. Bolya, C. Zhou, F. Xiao and Y. J. Lee, “YOLACT++ Better Real-time Instance Segmentation,” 2019.

[15] X. Wang, R. Zhang, T. Kong, L. Li and C. Shen, “SOLOv2: Dynamic and Fast Instance Segmentation,” 2020.

[16] A. Kirillov, Y. Wu, K. He and R. Girshick, “PointRend: Image Segmentation as Rendering,” 2019.

[17] Z. Wang, F. He, H. Zhang, P. Hao, X. Zhang and X. Li, “Characterization of the in-focus droplets in shadowgraphy systems via deep learning-based image processing method,” 2022.

[18] X. Zhang, H. Wang, W. Wang, S. Yang, J. Wang, J. Lei, Z. Zhang and D. Zhao, “Particle field positioning with a commercial microscope based on a developed CNN and the depth-from-defocus method,” 2022.

[19] R. Gadde, V. Jampani and P. V. Gehler, “Semantic Video CNNs through Representation Warping,” 2017.

[20] A. Arbelle, S. Cohen and T. R. Raviv, “Dual-Task ConvLSTM-UNet for Instance Segmentation of Weakly Annotated Microscopy Videos,” 2021.

[21] A. Pfeuffer, K. Schultz and K. Dietmayer, “Semantic Segmentation of Video Sequences with Convolutional LSTMs,” 2019.

[22] L. Yang, Y. Fan and N. Xu, “Video Instance Segmentation,” 2019.

[23] N. Xu, L. Yang, D. Yue, Y. Liang, J. Yang and T. S. Huang, “Video Object Segmentation (VOS),” 2018.

[24] V. Fedynyak, Y. Romanus, B. Hlovatskyi, B. Sydor, O. Dobosevych, I. Babin and R. Riazantsev, “DeVOS: Flow-Guided Deformable Transformer for Video Object Segmentation,” 2024.

[25] X.-W. Zhang, J.-Q. Chen, K. Zhang, S.-X. Zhang en H.-X. He, „Predictions for Unsteady Flow Fields With Deep,” IEEE Access, 2024.

[26] D. Choi, H. Kim and H. Park, “Bubble velocimetry using the conventional and CNN-based optical flow algorithms,” 2022.

[27] S. Yuan, L. Luo, Z. Hui, C. Pu, X. Xiang, R. Ranjan and D. Demandolx, “UnSAMFlow: Unsupervised Optical Flow Guided by Segment Anything Model,” 2024.

[28] V. Nekrasov, T. Dharmasiri, A. Spek, T. Drummond, C. Shen and . I. Reid, “Real-Time Joint Semantic Segmentation and Depth Estimation Using Asymmetric Annotations,” 2019.

[29] M. Koch, J. M. Rosselló, C. Lechner, W. Lauterborn, J. Eisener and R. Mettin, “Theory-assisted optical ray tracing to extract cavitation-bubble shapes from experiment,” 2021.

[30] L. Cheng, C. Xu, J. Li and B. Zhang, “3D measurement method of bubbles based on edge gradient segmentation of light field images,” 2022.

[31] W. Lin, C. Zheng, J.-H. Yong and F. Xu, “OcclusionFusion: Occlusion-aware Motion Estimation for Real-time Dynamic 3D Reconstruction,” 2022.

[32] R. Zhao, S. Liu, J. Liu, N. Jiang and Q. Chen, “A two-stage CFD-GNN approach for efficient steady-state prediction of urban airflow and airborne contaminant dispersion,” 2024.

[33] C. Jiang and N.-Z. Chen, “Graph Neural Networks (GNNs) based accelerated numerical simulation,” 2023.

[34] B. H. Creusen, “Experimental study of a small bubble column.,” Tech. rep. Technical University, 2020.

[35] J. B. Josso, “Multi-bubble interaction: Bubble column assembly, first tests and Matlab analysis,” Tech. rep. Technical University Eindhoven, 2021.

[36] M. She, T. Weiß, Y. Song, P. Urban, J. Greinert and K. Köser, Marine bubble flow quantification using wide-baseline stereo photogrammetry, 2022.

[37] H. Hessenkemper, L. Wang, D. Lucas, S. Tan, R. Ni and T. Ma, 3D detection and tracking of deformable bubbles in swarms with the aid of deep learning models, 2024.

[38] T. Homan and N. G. Deen, Deep learning bubble segmentation on a shoestring, 2024.

[39] E. J. Vega and J. M. Montanero, “Influence of a surfactant on bubblebursting,” 2023.

[40] B. Tuls, “Bubble Segmentation,” Github, [Online]. Available: https://github.com/HBTuls/BubbleSegmentation.

Appendix A: Inferences of the CNN models

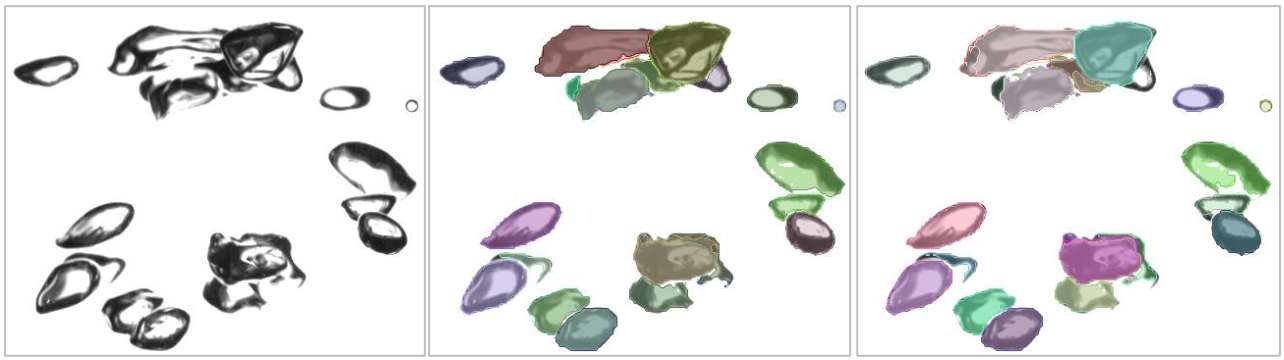

Figure 25. Segmentation inference for an out-of-distribution image: (a) input image; (b) SOLOv2 with depth-related labels (singular, occluding, occluded, and occluding-occluded bubbles); (c) SOLOv2 without depth-related labels.

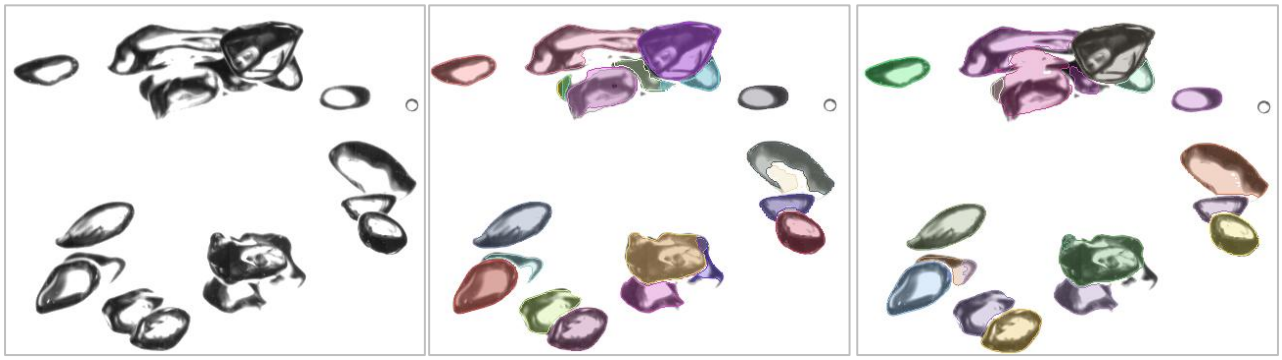

Figure 26. Segmentation inference for an out-of-distribution image: (a) input image; (b) Mask R-CNN with depth-related labels (singular, occluding, occluded, and occluding-occluded bubbles); (c) Mask R-CNN without depth-related labels.

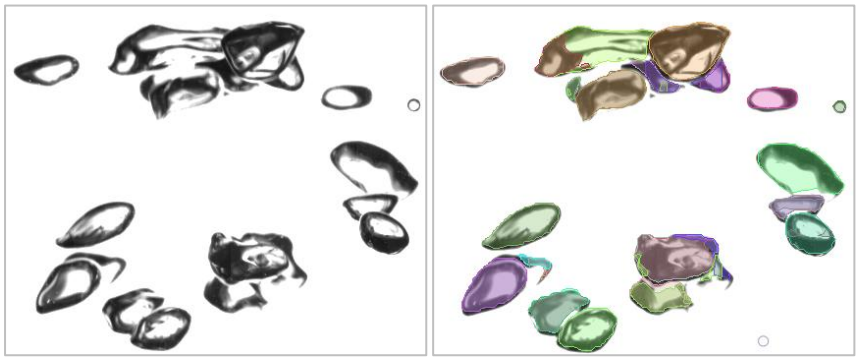

Figure 27. Segmentation inference for an out-of-distribution image: (a) input image; (b) YOLACT++ with depth-related labels (singular, occluding, occluded, and occluding-occluded bubbles).

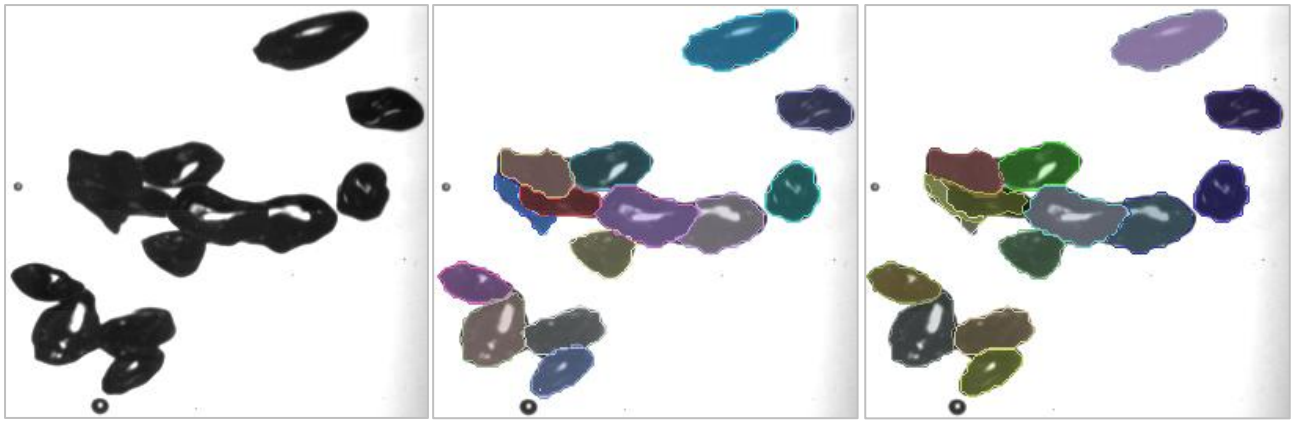

Figure 28. Segmentation inference for an image of the validation set: (a) input image; (b) SOLOv2 with depth-related labels (singular, occluding, occluded, and occluding-occluded bubbles); (c) SOLOv2 without depth-related labels.

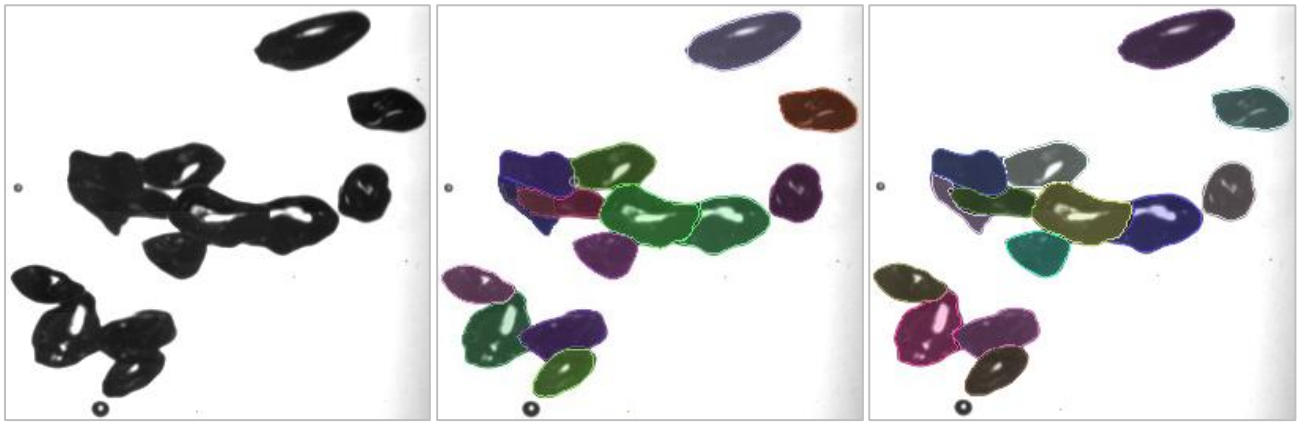

Figure 29. Segmentation inference for an image of the validation set: (a) input image; (b) Mask R-CNN with depth-related labels (singular, occluding, occluded, and occluding-occluded bubbles); (c) Mask R-CNN without depth-related labels.

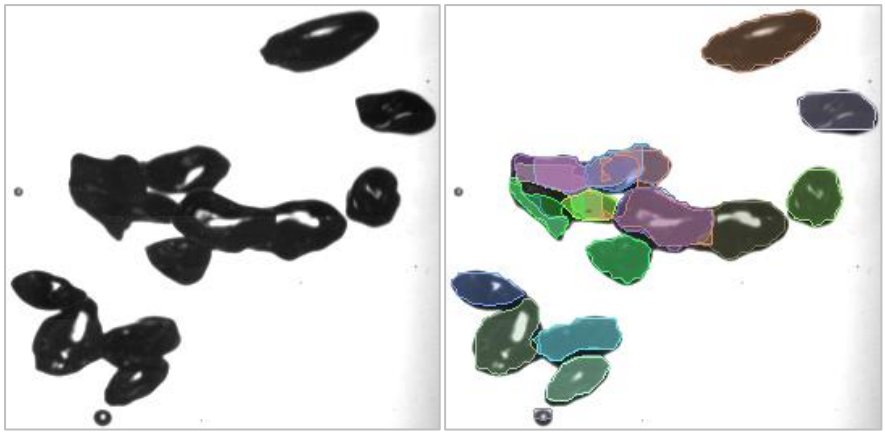

Figure 30. Segmentation inference for an image of the validation set: (a) input image; (b) YOLACT++ with depth-related labels (singular, occluding, occluded, and occluding-occluded bubbles).