Lappeenranta University of Technology
School of Engineering Science
Intelligent Computing Major

Master's Thesis

Violetta Shevchenko

FISH DETECTION FOR SPECIES RECOGNITION

Examiners:	Associate Professor Arto Kaarna
	Professor Vladimir Pilidi
Supervisors:	Associate Professor Arto Kaarna
	D.Sc. Tuomas Eerola

ABSTRACT

Lappeenranta University of Technology
School of Engineering Science
Intelligent Computing Major

Violetta Shevchenko

Fish detection for species recognition

Master's Thesis

2017

57 pages, 32 figures, and 7 tables.

Examiners: Associate Professor Arto Kaarna
Professor Vladimir Pilidi

Keywords: fish detection, moving object detection, fish classification, pattern recognition

Counting and tracking fish populations is important for conservation purposes as well as for the fishing industry. The fish counting typically occurs in rivers where the passing fish are counted either manually or automatically. Various automatic fish counters exist, based on such principles as resistivity, light beams and sonar. However, such methods typically cannot make distinction between fish and other passing objects, and moreover, cannot recognize different species. Computer vision techniques provide an attractive alternative for building a more robust and versatile fish counting systems. In this work the fish detection system, which provides the fish characterization for recognition purposes, was proposed. The results showed that by choosing an appropriate background subtraction method, it is possible to achieve a satisfying detection accuracy of 80% and 60% for two used datasets.

PREFACE

I wish to thank my supervisors, Associate Professor Arto Kaarna and D.Sc Tuomas Eerola, and the examiner, Professor Vladimir Pilidi, for their support and assistance during this research work. I also want to thank the organization "Kymijoen vesi ja ympäristö ry" for the provided video materials.

Finally, thank you to everyone who guided and supported me during this thesis work.

Lappeenranta, May 2017

Violetta Shevchenko

CONTENTS

LIST OF ABBREVIATIONS AND SYMBOLS

BS	Background Subtraction
PDF	Probability Density Function
GMM	Gaussian Mixture Model
KDE	Kernel Density Estimation
EM	Expectation-Maximization
ViBe	Visual Background Extractor
IoU	Intersection over Union
BG	background
FG	foreground
LBP	Local Binary Patterns
TP	true positive
TN	true negative
FP	false positive
FN	false negative
p	probability
$v(x)$	value of a pixel x
$I(x, y, t)$	intensity value of the pixel (x, y) at the time t
I_x	derivative of the image in the x-direction
I_y	derivative of the image in the y-direction
I_t	temporal derivative of the image
P	precision
R	recall
F_1	F1 score

1 INTRODUCTION

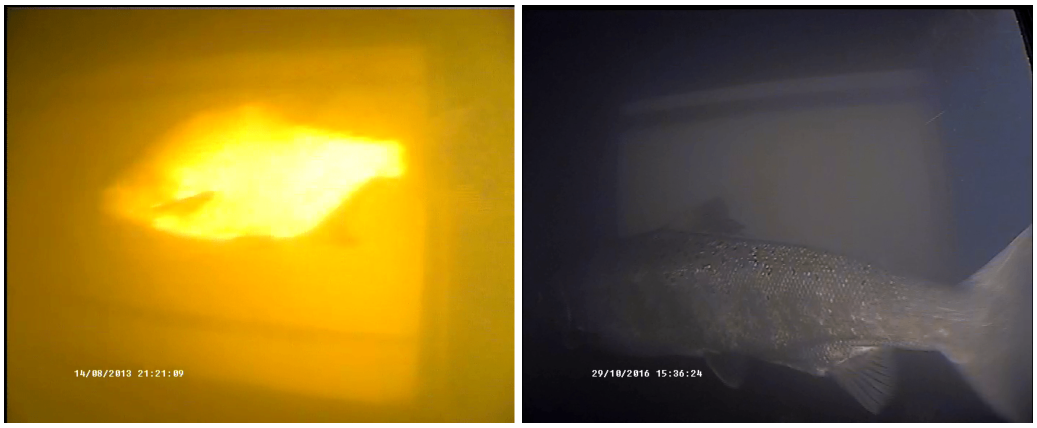
1.1 Background

Counting and tracking fish populations is important for conservation purposes as well as for the fishing industry. Usually, biologists determine the situation under the water by casting nets for collecting and examining fish or by human underwater observation [1]. Such methods are not able to provide a comprehensive observation, cannot capture the real fish state and behavior, and require much time and expenses. In addition, there is a big risk to kill or damage fish and their habitat during the collection process. These drawbacks force researchers to develop alternative ways of marine life observation [2].

One of the most commonly used approaches for underwater observation is to collect videos and analyse them to solve different tasks of marine life monitoring, for instance, fish counting. Many systems for underwater fish observation located in rivers and fish farms involve the manual counting of the passing fish, while the automation can speed up the process significantly, reduce the costs associated with the involvement of human experts and provide an opportunity for fish recognition.

Various automatic fish counters already exist, such as resistive counters that rely on the fact that the resistivity of a fish is lower than that of water [3], optical counters that use light beams [3], and hydroacoustic counters that operate using the principles of sonar [4]. Such approaches, based on different physical principles, cope with the task of counting passing objects quite successfully. However, such methods typically cannot make distinction between fish and other passing objects, and moreover, cannot recognize different species. Computer vision techniques provide an attractive alternative for building a more robust and versatile fish counting systems.

This thesis focuses on computer vision methods. The problem of fish counting is divided into two subtasks: the first subtask is to detect moving objects in videos, and the second subtask is to distinguish fish from another objects. The possibility to recognize the species of fish is also considered. The data for the research contains videos from muddy water with various types of illumination and visibility. The videos were collected in real environment by organization "Kymijoen vesi ja ympäristö ry" in 2013 and 2016. Example frames from the videos are shown in Figure 1. The goal is to find a method that can be applied to videos with different quality.



(a) (b)

Figure 1. Example images: (a) High illumination; (b) Low illumination.

1.2 Objectives and Delimitations

The aim of this work is to perform a comprehensive review of existing methods for object detection and recognition suitable for passing fish, to select the most promising ones, and to develop a system for detecting fish in underwater videos. Also, the possibility to perform fish species recognition is studied in the scope of the thesis. In total, the final goal of the project is to implement an application that is able to detect fish in the video, distinguish fish from other passing objects and extract features useful for fish classification. For that purpose, the following questions are considered:

- "How a moving object can be detected in the video sequences?"
  - "What kind of methods have been used for the task of fish detection?"
  - "Which methods have shown the best results?"
- "How fish can be recognized in the video sequences?"
  - "What features are needed for a successful fish recognition?"
  - "Is it possible to extract necessary features from the available data?"

Because of the lack of information about camera parameters and setup, the problem of camera calibration and estimation of the real fish size are not considered.

1.3 Structure of the Thesis

This thesis is organized as follows. Section 2 presents the existing computer vision methods for detecting moving objects. It comprises a common information about methods, their features, benefits and disadvantages. The usual process of moving object detection is presented in this section as well. It provides an overview of articles devoted to fish detection and the list of challenges which may occur during this detection. Section 3 contains material describing the problem of object recognition, especially the task of fish recognition.

In Section 4, the proposed approach for the fish detection is presented. The key steps of the process along with the applied methods and techniques are defined. The description of the conducted experiments, available data, evaluation methods and obtained results is given in Section 5.

Section 6 contains the discussion of the thesis, its results and perspectives for the future work. The last Section 7 draws a conclusion from the whole thesis.

2 DETECTING MOVING OBJECTS IN VIDEOS

2.1 Object Detection Methods

Video surveillance systems are currently widely used for analysing video sequences and applied, for example, in security control [5], traffic monitoring [6], anomaly detection [7] and animal monitoring and conservation [8]. Some systems include human intervention, but involving of a human operator may slow down the process of video analysis significantly and make the task impracticable. Therefore, the creation of autonomous video surveillance systems is one of the most actively developing areas of computer vision. The term autonomous in this case implies that the system receives a video as an input and performs both the low-level tasks, like motion detection, and high-level decisions like object recognition, without human expert involvement. [9]

Regardless the type of high-level task, the crucial step which must be done for further analysis is the moving object detection. It separates moving objects, which form the foreground, from the stationary background. Different approaches for the moving object detection from videos can be classified in the following way [10]:

- **Background subtraction (BS)** is commonly used in static scenes. It uses a modelled background image and subtract the current image pixel-by-pixel from it to detect regions of motion. With each new image a background model is updated to adapt to scene changes.
- **Temporal differencing** is usually applied in cases where the camera is moving and the background is not stable. Moving regions are detected by taking pixel-by-pixel difference of consecutive frames in a video sequence.
- **Statistical approaches** are a modification of BS methods, that use statistical information of each pixel and compare pixels' statistics to identify foreground.
- **Optical flow** can be used with the moving camera and moving background, but are computationally complex and time consuming. The basic idea is to compute the apparent velocity and direction of every pixel to find moving objects.

A typical process of moving object detection can be divided into five steps as shown in Figure 2 [11]. The first step is background modelling, which aim is to define a model describing the background scene. The next step is foreground detection which distinguish foreground pixels from the background by using the defined background model. During the following step pixel level processing, including, for example, morphological operations and low pass filter, is performed to remove the noise. Then the foreground pixels are grouped into connected regions and the bounded boxes of the regions are calculated. Finally, after a set of individual regions is obtained, all false regions, which do not correspond to objects, are removed based on, for example, size.

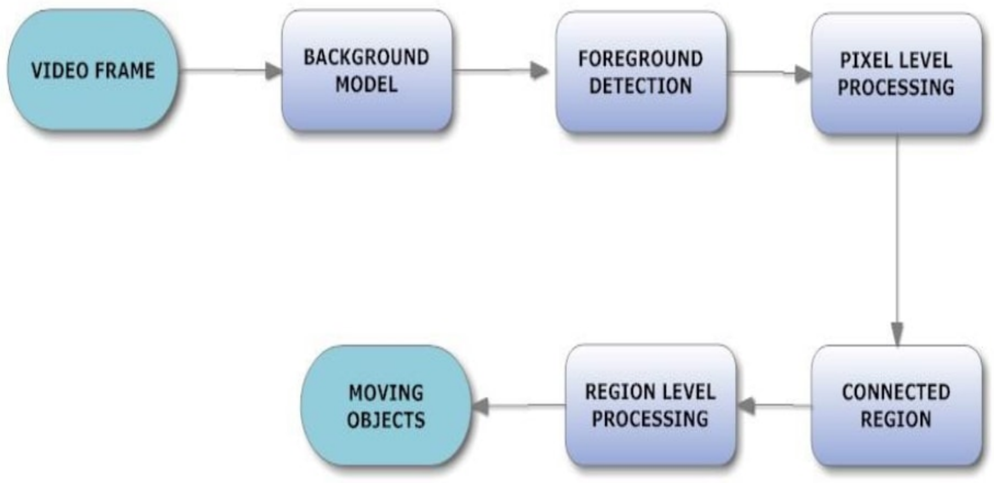


Figure 2. Framework of Moving Object Detection System [9].

2.2 Background Subtraction

BS methods are widely used for solving the task of moving object detection. These algorithms are relatively fast, which makes it possible to detect objects in real time. Moreover, they do not require a lot of memory and show robust results. BS approach is mostly applicable when both the camera and the background are stable, but can be easily modified for the cases with moving camera or background [12, 13]. Due to its benefits, BS methods have gained popularity among researches, which led to the emergence of a large number of different algorithms [14–16].

Most of the BS methods are implemented by the same scheme, which consists of three main steps (Figure 3):

- **Background modelling**. This is the first step of every BS algorithm. For detecting foreground objects, a background image that does not contain any moving objects needs to be estimated. For that purpose, a background model is built using a several number of successive frames.

- **Foreground detection**. After the background image is obtained, all subsequent frames are extracted from that model in order to find pixels that do not belong to the background and therefore can be considered as a foreground.
- **Background update**. During the detection process a background model must be updated with each new frame, because in real cases a background does not remain still throughout the whole video. The model must be adaptive for various changes (e.g. some moving object does not move for a long time) so that they do not affect the result of detection.

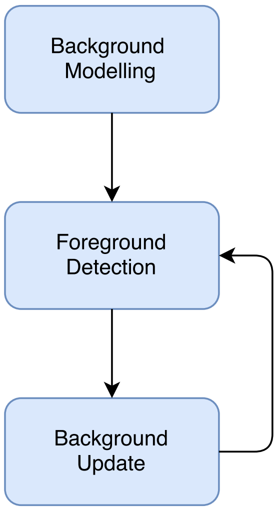


Figure 3. BS process.

Different BS methods are usually classified by the way they initialize a background model and how they compute the distance between background and foreground pixels [17, 18]. In accordance with this type of classification, the most commonly used BS techniques are the following:

- **Basic Motion Detection**. A background is represented by a single image taken when there is no motion or estimated by a use of temporal median filter [19, 20]. Difference between foreground and background pixels is calculated on the basis of pixels' colors.
- **One Gaussian**. The idea of this approach is that background pixels are modelled with a probability density function (PDF) and the distribution is considered

to be normal [21]. The parameters of PDF are estimated with a number of training frames. In this case, a pixel is assumed to be a foreground if it has low probability.

- **Gaussian Mixture Model (GMM)**. In contrast to the previous method, GMM based methods make use of a several Gaussians in modelling process [22].
- **Kernel Density Estimation (KDE)**. KDE is a non-parametric statistical approach, in which the PDF is estimated directly from the data [23].
- **Other methods**. There is a vast variety of different techniques that are based on, for example, fuzzy logic [24], neural networks [25], codebook [26] and eigenvalues and eigenvectors [27].

2.3 Fish Detection

Object detection algorithms have various applications. For instance, computer vision techniques provide a great opportunity to make animal monitoring more accurate, less time consuming and fully automated. In particular, different approaches and methods of moving object detection have been used for the task of fish detection, which is the fundamental step in building a fish observation system.

Palazzo et al. [28] proposed an approach for object detection that imply explicit modelling both the background and the foreground for each frame. This allows to avoid misdetections when the background is not stationary or when target objects have the same color as the background. The latter problem is also solved by introducing texture information in models as well as the color. The algorithm was evaluated on real underwater videos and showed high and stable performance.

In [29], an Expectation-maximization (EM) algorithm was proposed for fish detection. Given intensity images of fish (Figure 4), a certain threshold is selected, such that pixels with higher intensities are assumed to belong to the fish. Then the sets of x and y locations corresponding to these pixels are defined (Figure 5). The shape of each fish is assumed to be a multivariate Gaussian, then an image is modelled as GMM. And finally, the parameters of GMM, including the number of fish, are estimated using an EM algorithm. The method was tested on southern bluefin tuna fish, an individual fish was modelled by using a two-dimensional Gaussian distribution with x and y pixel locations as an input.

Spampinato et al. [2] presented an automated video processing system for underwater video surveillance. The system is able to solve the tasks of texture and color analysis,

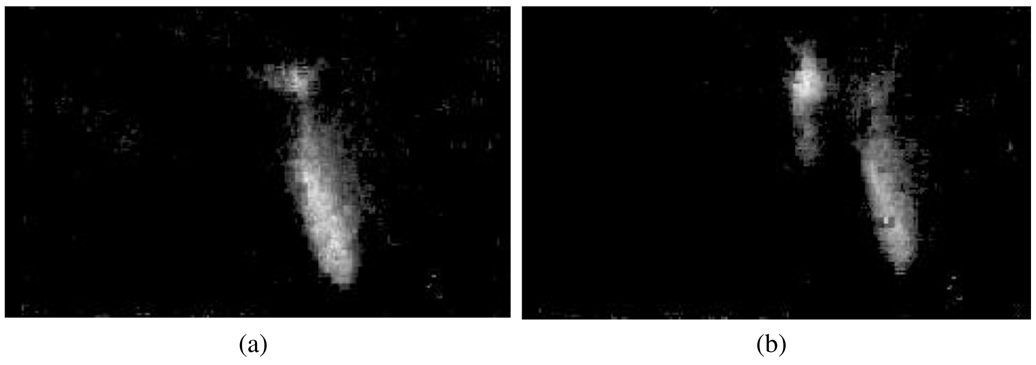


Figure 4. Intensity images: (a) Frame showing a single fish; (b) Frame showing two fish [29].

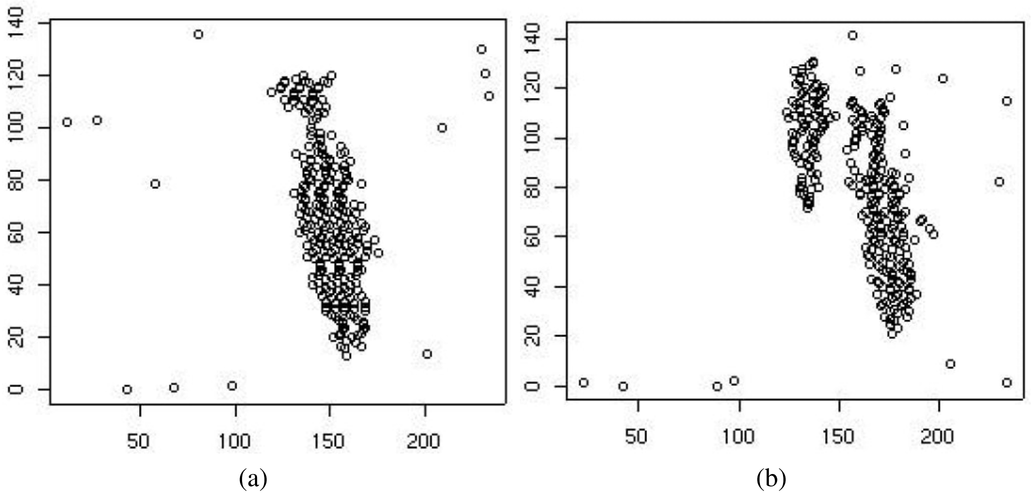


Figure 5. x and y pixel locations: (a) Frame showing a single fish; (b) Frame showing two fish [29].

fish detection and fish tracking by using three image processing subsystems. The first one calculates the average texture and color properties of the frame. It considers the brightness, smoothness and identifies the green tone. Fish detection subsystem is based on a BS approach. It uses the combination of moving average algorithm with the Adaptive GMM [30] which shows high performance on both stable and dynamic background scenes. The last subsystem tracks the fish throughout the video. The proposed system was tested on 20 underwater video sequences and the results were considered to be excellent when compared to other similar methods. An example of the system outcome is shown in Figure 6.

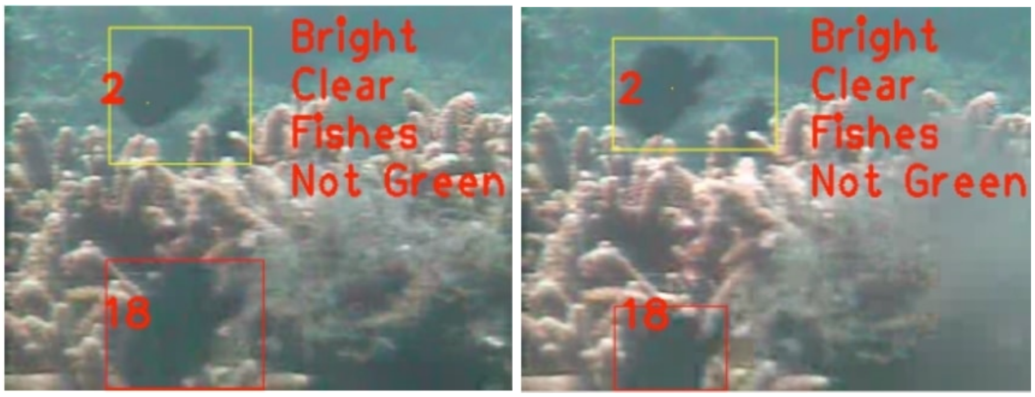


Figure 6. Tracking example with two frames grabbed at two consecutive times [2].

2.4 Challenges in Fish Detection

Despite the variety of the proposed approaches, it is a challenging task to find a robust and high performance method for fish detection. Usually video sequences may contain various attributes that make the task more difficult. Some of the possible problems connected with the detection are listed below [10]:

- **Illumination changes**. During the video capturing the illumination may change gradually (e.g. intensity of the sun changes throughout the day) or suddenly (e.g. the lamp has been turned off), so the background model must take the light into account.
- **Dynamic background**. Often the background is not a stable scene and the algorithm should distinguish moving objects of interest from those that can be regarded as background.

- **Occlusion**. Object of interest cannot be fully seen and detected on some frames if it is occluded by other objects.
- **Clutter**. It hinders the process of background modelling and complicates the segmentation.
- **Camouflage**. Target objects may have the similar color and pattern as a background, which makes them hardly distinguished even by the human.
- **Camera motions**. If the camera is unstable it may affect the quality of the captured video.
- **Noise**. Video may contain noise which also affects the results.
- **Environmental issues**. The quality of water, underwater garbage, presence of moving plants, etc. may also hamper the fish detection.

In the case of fish detection, different types of videos are used for experiments (Figure 7). In low quality video, there is noise and fish do not differ sufficiently from the background in the terms of color. High quality video on the opposite allows all fish to be distinguished easily, but that may cause clutter and occlusions. Illumination changes and sudden appearance of glares are common attributes of underwater videos as well. This kind of troubles should be considered while choosing an appropriate detection method.

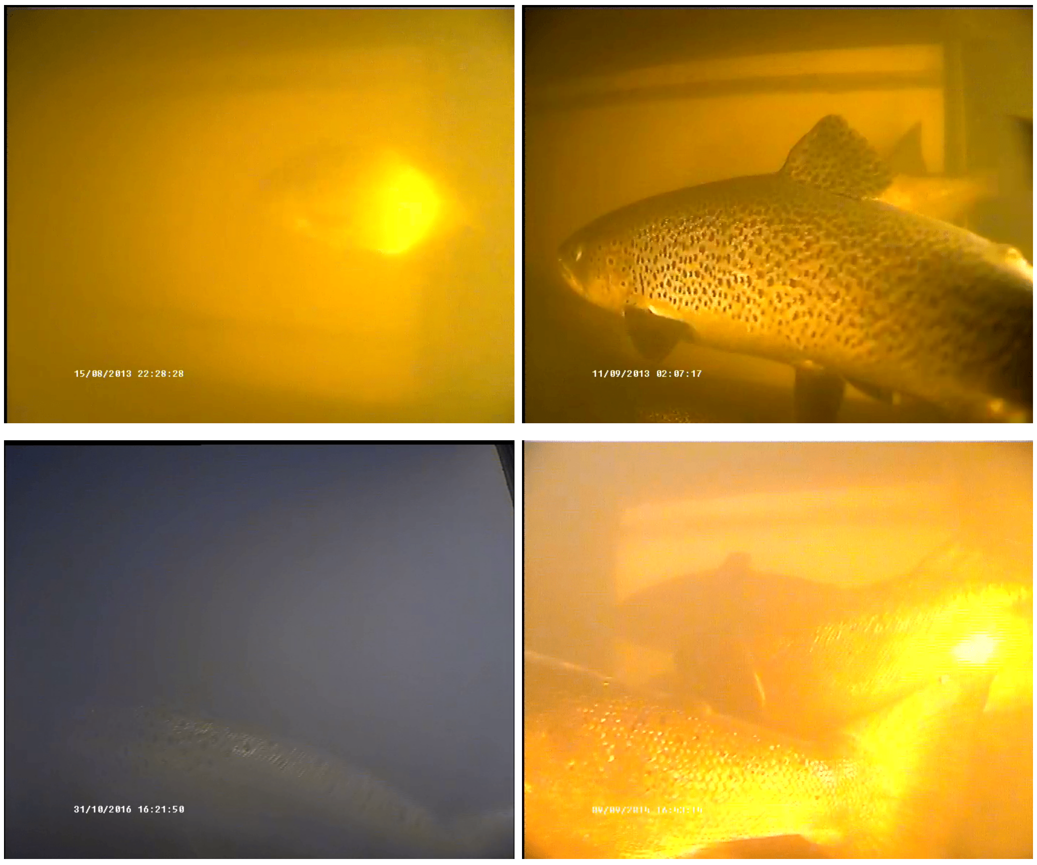


Figure 7. Examples of difficult cases for fish detection.

3 OBJECT RECOGNITION

3.1 Background

Object recognition is a scientific discipline, which is applied in various spheres connected with data analysis [31]. In brief, it is a process of classification, where an object is categorized into certain class. As for computer vision, object recognition is a key part of many image or video analysis systems. Given a single or multiple images, a system needs to determine what is depicted. Usually, an additional information is available, like database of all possible objects, that is classes are known a priori.

To be able to distinguish objects in different classes, there should be some criterion, on the basis of which a separation can be made [32]. The most obvious decision is to consider two objects to be of the same class if they are similar to each other. For that purpose, an object is represented by a number of features, and then objects are similar if they are close enough in the feature space. Features need to be carefully selected since the success of classification directly depends on them. The choice of features should be such that within one class objects would be very close to each other, while features of objects from different classes will differ significantly.

In general, the process of object recognition from video can be divided in three steps (Figure 8). At first, an object is detected, then its features are extracted and, finally, the classification is performed. An overview of object detection techniques was given in Section 2, feature extraction is explained further in the text and the classification task is out of scope of this work.

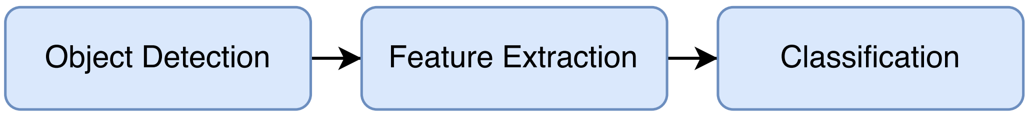


Figure 8. General scheme of object recognition.

When dealing with video, which is basically a series of successive frames, the range of possible features is limited by that information which can be extracted from the images. In case of video with moving objects, for example the following features can be used for characterization:

- **Size**. The basic properties which can be calculated for an object is its width, height and area.

- **Shape**. Various shape properties, such as object roundness, centroid and orientation, can be used to describe the object.
- **Color** features, such as color histograms, are one of the most commonly used properties for describing an object.
- **Texture**. Some objects have a specific texture, such as a unique fish scale pattern, which can be used to recognize them.
- **Motion**. An object can be characterized by the way it is moving in the video. Examples of such features are velocity and direction.

3.2 Fish Recognition

Moving object detection methods allow to find, track and count fish automatically, but simple fish detection is not enough for building a fully autonomous surveillance system. To provide an accurate and detailed information about the underwater environment, it is usually required to recognize all passing species, so basically each passing fish need to be classified. Object recognition is a challenging task, especially in the case of fish classification, because underwater videos are usually of low quality, contains noise and can have a low contrast. That makes the process of feature extraction quite complicated and hamper the classification itself.

Alsmadi et al. [33] presented a system, that takes an image of fish as an input, recognize it and categorized into "poison" or "non-poison" family. In the work much attention is paid for feature extraction, which consists in various size and shape measurement. At first, anchor points are detected as shown in Figure 9. They are used to calculate the geometry of the fish, from which such features as width, height, mouth length, head angle and others are extracted. A neural network with back propagation algorithm is used for classification. The accuracy of the classification varies for different fish families and lies between 75% and 95%.

Spampinato et al. [34] proposed a way to extend capabilities of the fish monitoring system described in [2] and to refine it into an automatic fish classification system. An outcome of the detection subsystem is a bounding box (Figure 10), from which a fish contour is extracted (Figure 11) and later used for feature extraction. Authors suggest to use affine invariant features to simplify the classification, because a fish can be at any orientation

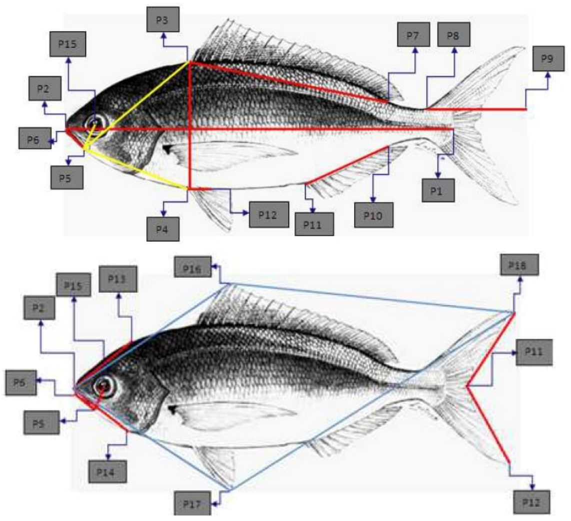


Figure 9. Anchor point locations [33].

and position relative to the camera. For that purpose, an affine transformation (Figure 12) is used to represent a 3D shape of a fish and to get affine invariant features. Such features are: texture that is received from the grey-level histogram, the Gabor filters and the grey-level co-occurrence matrices, and boundary features. To reduce the number of obtained features Principal Component Analysis is used and then Discriminant Analysis is applied for the classification.

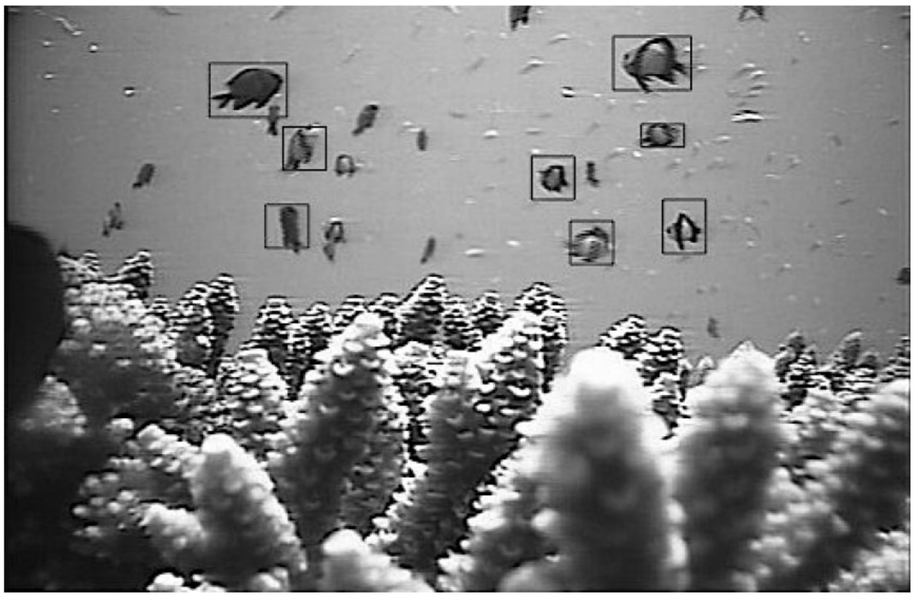


Figure 10. Output of the detection system [34].

Hsiao et al. [35] describe a distributed real-time underwater video observational system that can be used for monitoring of a coral reef ecosystem. Using a large amount of col-

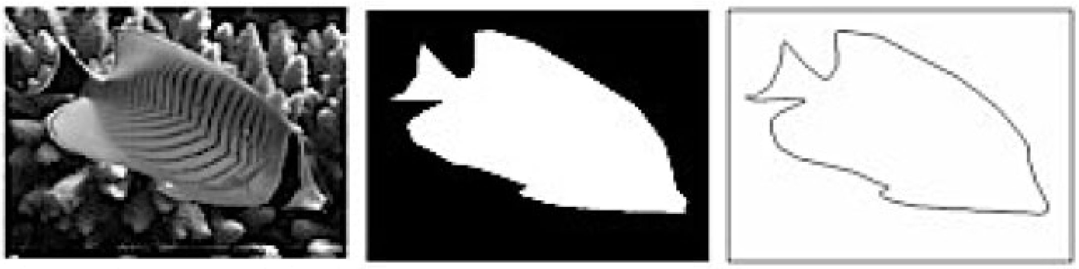


Figure 11. Fish contour [34].

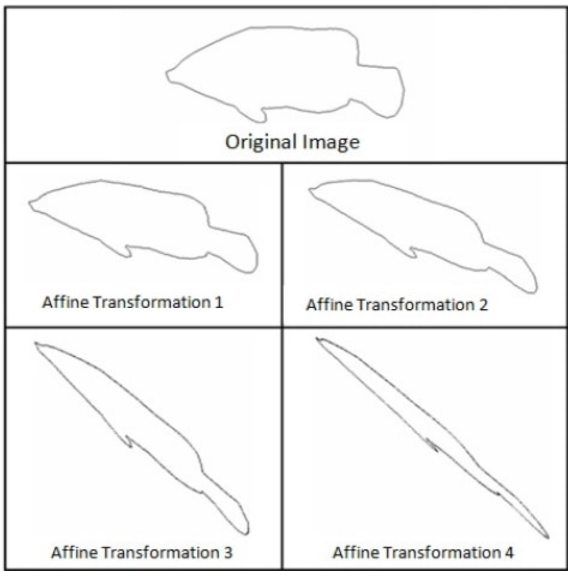


Figure 12. Affine transformation of fish contour [34].

lected videos, a fish category database has been built for recognition purpose. To recognize fish species on the video, at first, BS methods are used for fish detection and then a maximum probability of partial ranking method is applied. This method is based on a sparse representation-based classification, in which an image represents a sparse linear combination of all training images with weighting coefficients as unknowns. An identification process defines whether an image has one of the species from the database or it is a new one. The proposed method was proven to perform better than other popular methods. The described approach is shown in Figure 13.

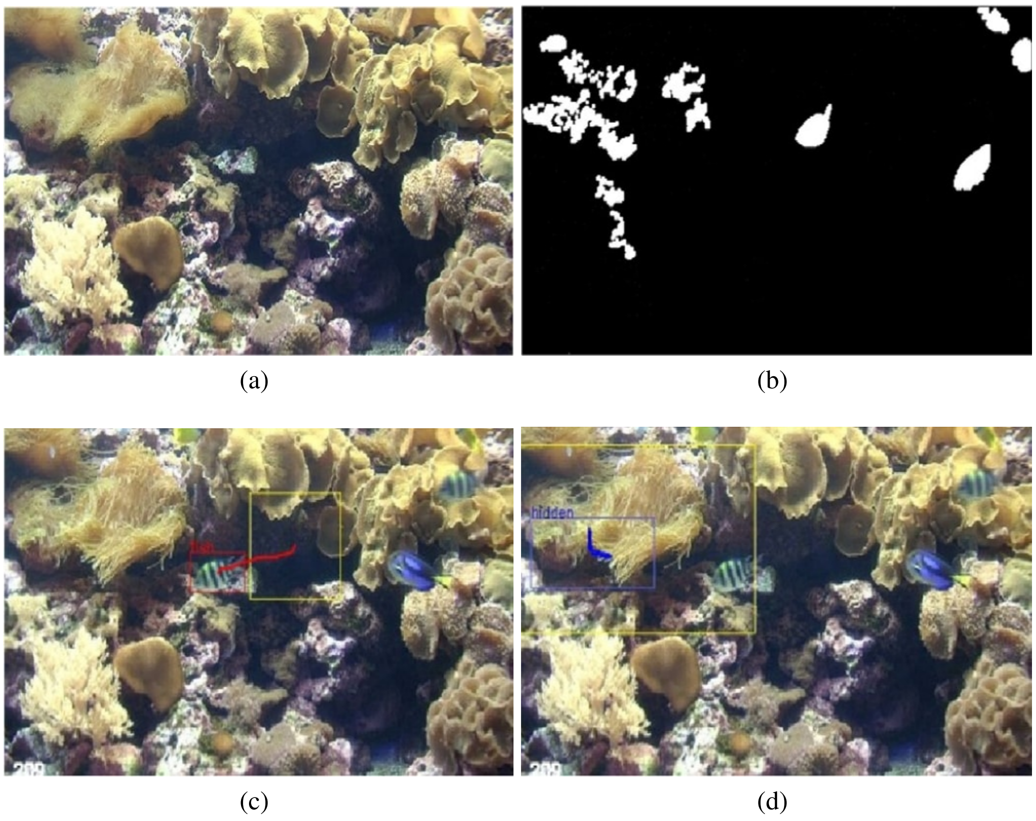


Figure 13. (a) The background model; (b) Detected moving objects; (c) The object (red box) is classified as "fish"; (d) the object (blue box) is classified as "non-fish" object [35].

The work of Chuang et al. [36] comprises the results of comparison of two fish feature extraction methods, namely the supervised and unsupervised approaches. The main difference between these approaches is that supervised methods use predefined features like shape or color, while unsupervised methods learn the features directly from images. Both approaches are tested by using a hierarchical partial classifier. The experiments showed that unsupervised methods achieve better performance on low quality videos and imbalanced fish distribution.

According to the reviewed articles, the most commonly used features of fish are size, shape, color and texture, so some of these features can be taken into account while choosing appropriate features for fish classification. Nevertheless, the choice of algorithm for classification is not obvious and should be based on the results of fish detection and extracted features.

4 PROPOSED APPROACH

The main objective of this thesis is to build a system that is able to detect passing fish in the underwater videos and extract features which may be used for classification purposes. Considering the reviewed works, the most promising approach for fish detection is BS as it is commonly used and shows good results. Also some statistical methods can be included to improve the performance. Temporal differencing may be less effective as in the regarded case the background is mostly stable and camera is fixed, what makes the advantages of this approach inessential. Optical flow methods show worse results, when foreground objects do not differ significantly in color from the background. But they can be used as an additional tool to describe the motion after it has been detected. Therefore, in this work BS methods are used to develop a robust system for fish detection.

The whole approach can be divided into the following steps: BS, Object Detection and Feature Extraction (Figure 14). The application takes a video as an input and returns a set of features as an output. Detailed descriptions of each of the subsystems are presented below.

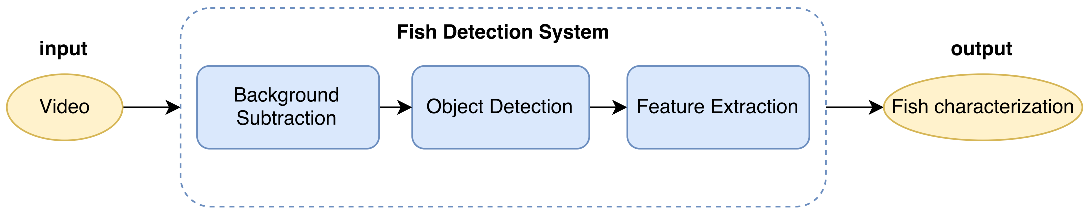


Figure 14. Proposed approach.

4.1 Background Subtraction

To detect fish in a video, firstly, all moving objects must be separated from the background. For solving this task, a BS approach was chosen as the most appropriate one. It is relatively simple, fast and able to cope with the most possible challenges. A BS algorithm takes a raw video, processes it and returns a set of binary maps, where "0" represent background pixel and "1" refers to the foreground (Figure 15). This process may be arranged in real time, when each frame is analysed instantly after reading it. In this case, first few frames are used only for building the model, while in all other frames the motion can be detected.

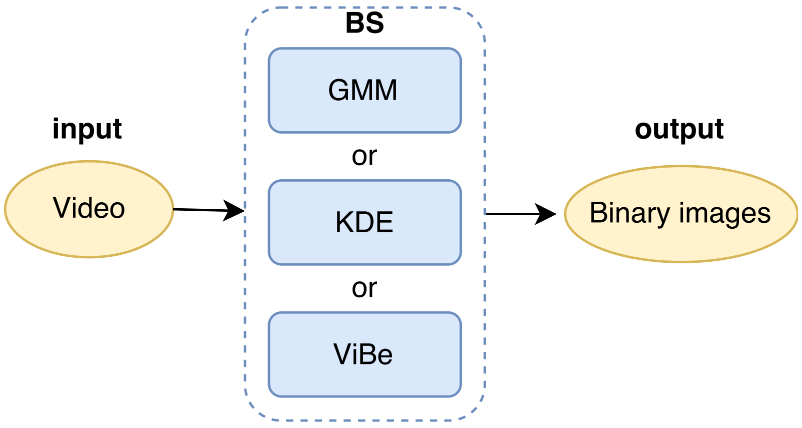


Figure 15. BS subsystem structure.

While choosing the specific BS method the following criteria were considered: accuracy, robustness, ability to cope with noise, moving background and illumination changes, computational speed and the simplicity of implementation. An extensive survey on the performance of various methods is given in [18, 37]. Based on the gathered information, three algorithms were chosen for BS: Adaptive GMM, KDE and Visual Background Extractor (ViBe).

4.1.1 Adaptive Gaussian Mixture Model

The first selected algorithm is based on the GMM and described in [30, 38]. One of the most important properties of this algorithm is that it presents the color of each pixel using multiple Gaussians and automatically chooses the number of Gaussians. This ability makes it fully adaptable for sudden and gradual scene changes. As in typical fish counting videos the illumination is changing and some moving objects stay motionless for a long time, the choice of this algorithm seems to be reasonable.

The proposed GMM method refers to the statistical BS approach, which means that the background image, due to its regular behaviour, is assumed to be well described by some statistical model. In this case, a scene model is built by estimating a PDF for each pixel. As pixels usually have complex distributions, authors suggested to use GMM instead of one PDF. Those pixels, which do not fit the model, are considered as a foreground.

The background model is denoted by $p(\overrightarrow{x}|BG)$, where $\overrightarrow{x}$ is a pixel value in some colorspace. A training set $\mathcal{X}$ is used for model estimation and $\widehat{p}(\overrightarrow{x}|\mathcal{X}, BG)$ stands for the obtained model. At each moment of time t the following decision must be made: whether the pixel belongs to the background or to the foreground. For this, the Bayesian decision B is

$$B = \frac{p(BG|\overrightarrow{x}^{(t)})}{p(FG|\overrightarrow{x}^{(t)})} = \frac{p(\overrightarrow{x}^{(t)}|BG)p(BG)}{p(\overrightarrow{x}^{(t)}|FG)p(FG)}. \tag{1}$$

Typically nothing is known about the foreground. So $p(FG)$ is set equal to $p(BG)$ and the distribution is assumed to be uniform $p(\overrightarrow{x}^{(t)}|FG) = c_{FG}$. Then the pixel is referred to the background if

$$p(\overrightarrow{x}^{(t)}|BG) > c_{thr}, \tag{2}$$

where $c_{thr} = Bc_{FG}$ is a threshold value.

Since we expect that the scene may vary over time, it is essential that the training set should adjust to changes by adding or eliminating samples. So at time t the training set is $\mathcal{X}_T = \{x^{(t)}, ..., x^{(t-T)}\}$, where T is a selected time period. The set $\mathcal{X}_T$ and the model itself are recalculated with each new sample. As our goal is to detect the foreground, some of the samples will contain foreground pixels, which should not affect on the update of the background. So in general, the proposed GMM approach with M components is described as

$$\widehat{p}(\overrightarrow{x}|\mathcal{X}_T, BG + FG) = \sum_{m=1}^{M} \widehat{\pi}_m \mathcal{N}(\overrightarrow{x}; \widehat{\overrightarrow{\mu}}_m, \widehat{\sigma}_m^2 I), \tag{3}$$

where $\widehat{\overrightarrow{\mu}}_1, ..., \widehat{\overrightarrow{\mu}}_M$ and $\widehat{\sigma}_1^2, ..., \widehat{\sigma}_M^2$ are the estimates of means and variances respectively, I is the identity matrix and $\widehat{\pi}_1, ..., \widehat{\pi}_M$ are non-negative weights. The parameters are updated recursively by the following rules:

$$\begin{aligned} \widehat{\pi}_m &\longleftarrow \widehat{\pi}_m + \alpha(o_m^{(t)} - \widehat{\pi}_m) \\ \widehat{\overrightarrow{\mu}}_m &\longleftarrow \widehat{\overrightarrow{\mu}}_m + o_m^{(t)}(\alpha/\widehat{\pi}_m)\overrightarrow{\delta}_m \\ \widehat{\sigma}_m^2 &\longleftarrow \widehat{\sigma}_m^2 + o_m^{(t)}(\alpha/\widehat{\pi}_m)(\overrightarrow{\delta}_m^T \overrightarrow{\delta}_m - \widehat{\sigma}_m^2), \end{aligned} \tag{4}$$

where $\overrightarrow{\delta}_m = \overrightarrow{x}^{(t)} - \widehat{\overrightarrow{\mu}}_m$, $\alpha \approx 1/T$ and $o_m^{(t)}$ is the ownership that defines the closeness between a sample and a component.

A detailed description of this approach is presented in [30] along with the explanation of how the number of components are selected automatically, so the algorithm can be implemented.

4.1.2 Kernel Density Estimation

Zivcovic et al. [38] introduced a new non-parametric BS method based on KDE. In contrast to the equation describing GMM approach (Equation 3), the KDE is given by

$$\widehat{p}(\overrightarrow{x}|\mathcal{X}_T, BG + FG) = \frac{1}{TV}\sum_{m=t-T}^{t}\mathcal{K}(\frac{\|\overrightarrow{x}^{(m)} - \overrightarrow{x}\|}{D}) = \frac{k}{TV}$$
$$\mathcal{K}(u) = \begin{cases} 1 & \text{if } u < 1/2, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

where $\mathcal{K}(u)$ is the kernel function, V is the volume of the kernel which is a hypersphere with diameter D.

This method is proven to be reasonably effective, when the number of foreground pixels is relatively small. Typical fish detection datasets include videos, where usually only one moving fish is presented in each frame, thus the foreground fraction is quite limited. This justifies the use of the KDE method.

4.1.3 Visual Background Extractor

In [39], a universal BS algorithm, called ViBe, is presented. It is proven to be one of the most effective techniques for moving object detection. The general idea of the algorithm is the following: for each pixel, there is a set of model values that were taken in the past from the same pixel location or its neighbourhood. This set is later compared with the captured pixel value and, according to the result of comparing, the pixel is marked as background or foreground. The set is updated randomly and each detected background pixel affects models of its neighbours.

To completely describe an algorithm, one needs to determine what the background model is, how it must be initialized and updated. In ViBe a model for pixel located at x is denoted by $\mathcal{M}(x)$ and defined as a set of background samples:

$$\mathcal{M}(x) = \{v_1, ...v_N\}, \tag{6}$$

where $v_1, ...v_N$ are pixel values taken in previous frames. In order to detect foreground pixels, for each x a sphere $S_{\mathcal{R}}(v(x))$ of predetermined radius $\mathcal{R}$ centred in $v(x)$ is defined. Then the intersection between the sphere and the model is calculated. Finally, if the number of intersected elements exceeds a certain threshold $\sharp_{min}$, pixel is classified as a background:

$$\sharp(S_{\mathcal{R}}(v(x)) \cap \mathcal{M}(x)) \geq \sharp_{min}, \tag{7}$$

where $\sharp$ is the cardinality of a set. This approach is resistant to outliers, which allows to achieve more reliable results.

Unlike many other techniques, ViBe is able to initialize a background model by using a single frame. It speeds up the process significantly, allows to start the detection as soon as possible and, moreover, makes it possible to process short videos effectively. As only one frame is used, there is no temporal information available. That is why some additional assumptions must be made. Authors suggested, that pixels in the neighbourhood share a similar distribution, so a pixel can be modelled using the values of its neighbours:

$$\mathcal{M}^0(x) = \{v^0(y|y \in N_G(x))\}, \tag{8}$$

where $M^0(x)$ is a background model for pixel x at time $t = 0$, $N_G(x)$ denotes a neighbourhood of x and locations y are chosen randomly.

One of the crucial features of each BS algorithm is its ability to adapt to different changes. For that reason, almost in every algorithm there is a step on which the model is updated. The update process in ViBe is based on several principles:

- Pixels belonging to the foreground are never included to the model.
- The choice of the sample to be replaced is random.
- Not all background pixels are updated each time. The choice of the pixels is also random.
- An information used for pixel update affects the models of its neighbours as well.

A simple implementation of ViBe written in pseudocode is presented in Algorithm 1.

Authors state, that this technique allows ViBe to instantly adjust to most possible situations, such as fast or gradual change of illumination, appearance of new background objects or sudden background movements. It also helps to quickly correct negative consequences in the case, when the first frame used for background initialization contained objects of motion.

Algorithm 1 : ViBe [39].

```
Read the first frame.
for all pixels $x$ in frame do                              ▷ Background initialization
    Randomly choose $N$ pixels from the neighbourhood of $x$.
    Store these values $v_1, ...v_N$ in the pixel model $\mathcal{M}(x)$.
end for

for all remaining frames do
    for all pixels $x$ do                                   ▷ Foreground detection
        Compare current pixel value $v(x)$ to the background model $\mathcal{M}(x)$.
        Find the number $num$ of close samples in the model.
        if $num \geq \#_{min}$ then
            Classify pixel as a background.
                                                            ▷ Background update
            Get random sample from pixel model and update it with the current pixel
            value.
            Randomly choose one neighbour and update its model.
        else
            Classify pixel as a foreground.
        end if
    end for
end for
```

4.2 Object Detection

Outcomes of the BS subsystem are binary images with the detected foreground. The next step is to find fish on the images. For that purpose, several post-processing techniques are applied. After all objects are detected, only those objects which correspond to fish are selected, and, finally, bounding boxes of these objects are received. This process is illustrated in Figure 16.

4.2.1 Post-Processing for Background Subtraction

As it was mentioned before, videos usually have various artefacts which complicate their analysis. For this reason, the raw output of BS algorithm generally contains a lot of noise in the form of foreground pixels which actually belong to the background and vice versa (Figure 17(a)). This kind of distortions must be eliminated to exclude their influence on the subsequent analysis. The post-processing for BS consists of three steps:

- **Median filter** is a non-linear filter used in image processing, normally for noise

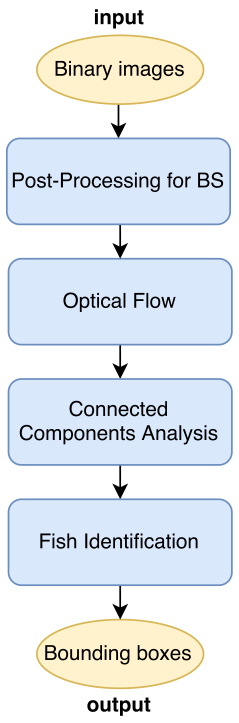


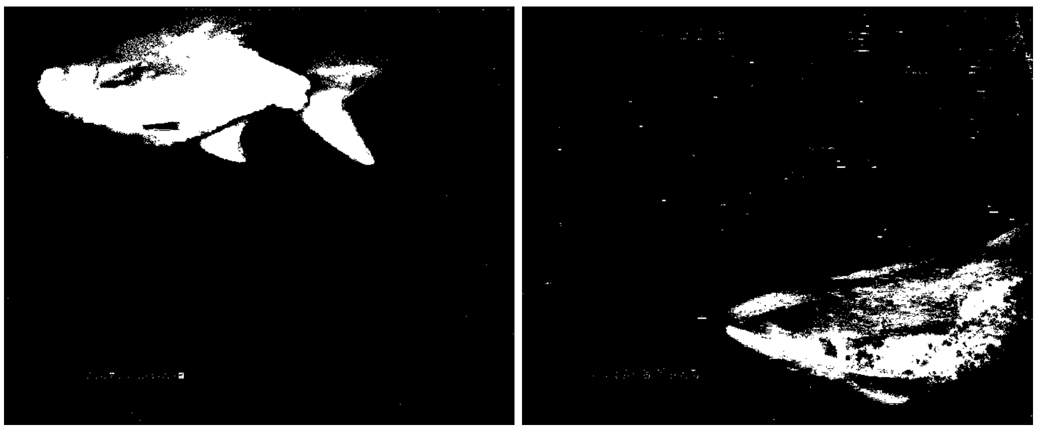
Figure 16. Object Detection subsystem structure.

reduction [40]. This filter takes a neighbourhood of a pixel, finds its median, and replaces the current pixel value with the median value. The main feature of the median filter is that it reduces impulse noise and preserves edges. Misdetected foreground pixels appears as white spots on the black background, which looks like an impulse noise. Therefore, median filter is a logical choice in this case.

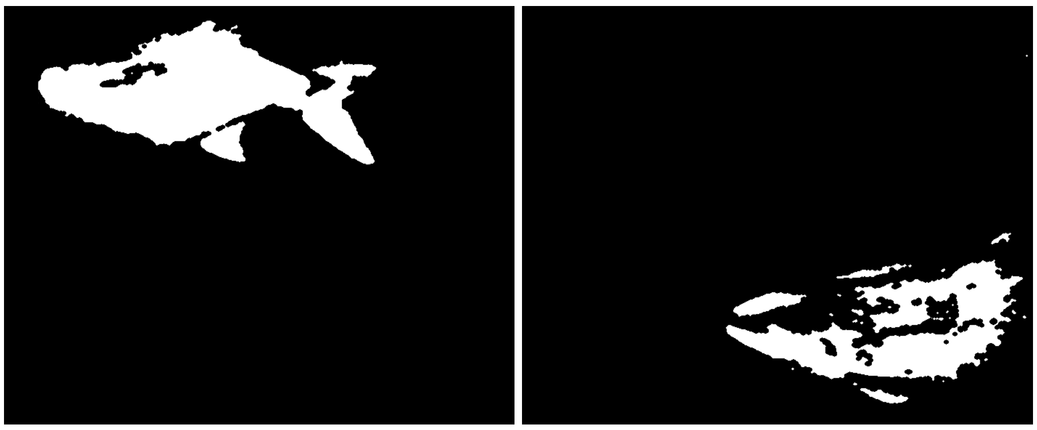
- **Morphological opening** is a morphological operation composed of a consecutive application of erosion and dilation with the same structuring element [41]. This operation removes small objects, which are most likely noise, but does not alter big objects. In this way, opening is used as a final step in noise removal.

- **Morphological closing**. In contradistinction to opening, closing operation is a successive usage of dilation and erosion [41]. Most common application of closing is filling the holes in objects. Thus, background pixels inside an object, which were misclassified to foreground, will be painted with white color after closing operation.

The presented approach is a common pattern used in post-processing for BS [42]. It helps to get rid of the vast majority of misclassified pixels which facilitates the further analysis. An example of the processed frames is given in Figure 17(b).



(a)



(b)

Figure 17. Post-processing: (a) Raw frames with noise; (b) Frames after post-processing.

4.2.2 Optical Flow

Although BS is applied for moving object detection, it results in finding the regions of motion, but does not provide any specific information about the motion itself. In many cases, it may be useful to know how exactly the objects are moving. For instance, different types of objects can be distinguished according to the direction of their movement. Or, on the contrary, several components can be considered to belong to the same object, if they move with the same speed and direction. The outputs of motion analysis will be used in connected components analysis.

Optical Flow provides an opportunity to measure a motion from video frames [43]. The technique is based on assumptions, that pixel values do not vary significantly between consecutive frames and that pixels in one neighbourhood move in a similar way. This hypothesis is called brightness constancy constrain with the pixel displacement (dx, dy) after time dt:

$$I(x, y, t) \approx I(x + dx, y + dy, t + dt). \tag{9}$$

By expanding with Taylor series, we get

$$I(x + dx, y + dy, t + dt) = I(x, y, t) + \frac{\partial I}{\partial x}dx + \frac{\partial I}{\partial y}dy + \frac{\partial I}{\partial t}dt + O^2, \tag{10}$$

where O^2 are the second and higher order terms, which are neglected. From this equation, it follows that:

$$\frac{\partial I}{\partial x}dx + \frac{\partial I}{\partial y}dy + \frac{\partial I}{\partial t}dt = 0, \tag{11}$$

which can be transformed into:

$$\begin{aligned} &\frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t}\frac{dt}{dt} = 0 \\ &\frac{\partial I}{\partial x}V_x + \frac{\partial I}{\partial y}V_y + \frac{\partial I}{\partial t} = 0 \\ &I_x V_x + I_y V_y = -I_t, \end{aligned} \tag{12}$$

where V_x and V_y are the components of the image velocity.

An optical flow implementation based on work of Farnebäck [44] is used in the proposed method. It takes a series of binary images and finds optical flow vectors for all pixels in each frame (Figure 18). On the basis of the obtained vectors, their magnitude and direction are computed.

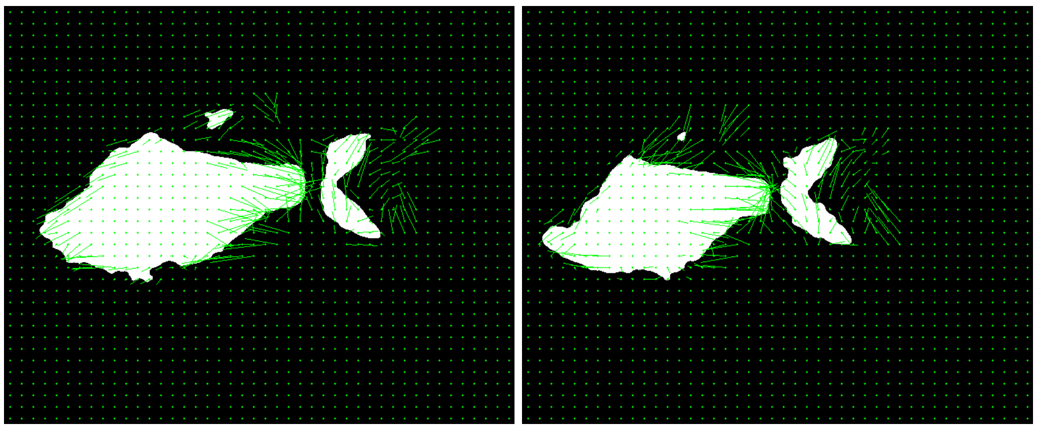


Figure 18. Optical flow found for two consecutive frames.

4.2.3 Connected Component Analysis

On this stage the remained objects are detected. At first, all connected components are found together with their properties like area, centroid point, bounding box, average velocity and direction. For that purpose, a connected component labelling algorithm is used. It is an algorithm that takes a binary image, finds regions of connected foreground pixels and marks each region with its own label. Then the features of each region are calculated.

Unfortunately, BS methods are not ideal and objects may still contain holes and be disconnected even after post processing. Such objects are detected as a set of a certain number of regions (Figure 19(a)). To correct this, all regions belonging to one object must be combined into one (Figure 19(b)). In order to determine whether two regions belong to the same object or not, the following heuristics were proposed. Two regions are combined, if several or all the conditions listed below are satisfied:

- The distance between the two centres is smaller than a selected threshold value.
- The height of the combined region is smaller than a threshold.
- The length of the combined region is smaller than a threshold.
- Two regions are moving with a similar velocity.
- Two regions are moving in the same direction.

Conditions which are used for region combination depend on the specific task and can be selected individually. For instance, when the approximate length of the fish is known, this

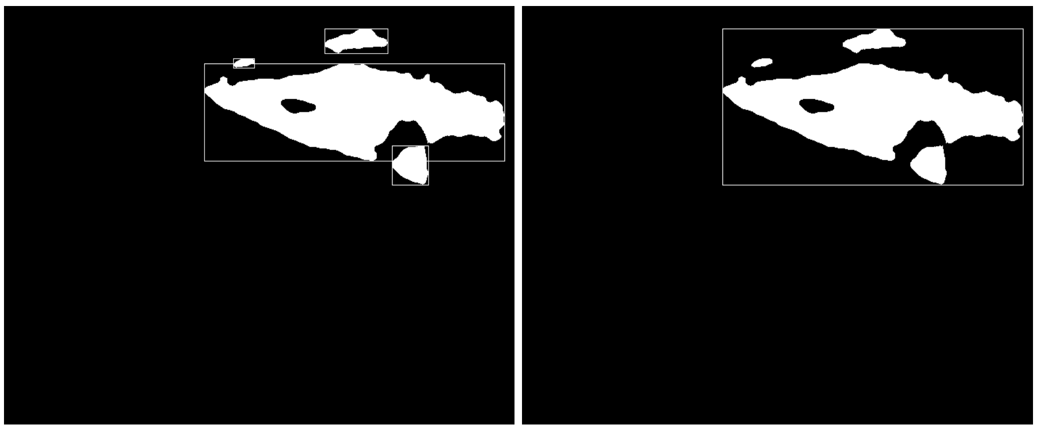


Figure 19. Combination of regions: (a) Initial regions; (b) Combined regions.

value can be selected as a maximum length threshold value. The same situation is with the height. When there is no prior information about the fish, these conditions should not be set because of the inability to select a threshold. As another example, one can consider the situation when many objects are moving with the same speed and direction. Now the last two conditions become useless because they do not help to distinguish objects.

4.2.4 Fish Identification

From the previous step a set of detected objects is received. This set may still contain noise because in real underwater videos fishes are not the only passing objects. Such items as garbage, plants and various marine life would be detected as well. So the last step of fish detection is fish identification, which means the separating fish from everything else. To distinguish objects, some prior information about fish is needed.

Typically, an approximate fish size can be estimated from videos. That is why, at first, all objects which are too small to be a fish are eliminated. This is implemented by introducing a threshold value for the minimum area of the region. Next, fish in underwater videos usually move in one specific direction. Thus, all objects that pass in the opposite direction can be considered as non-fish. These known facts simplify the process of fish identification, but of course do not guarantee an ideal differentiation. An example of object distinguishing is shown in Figure 20.

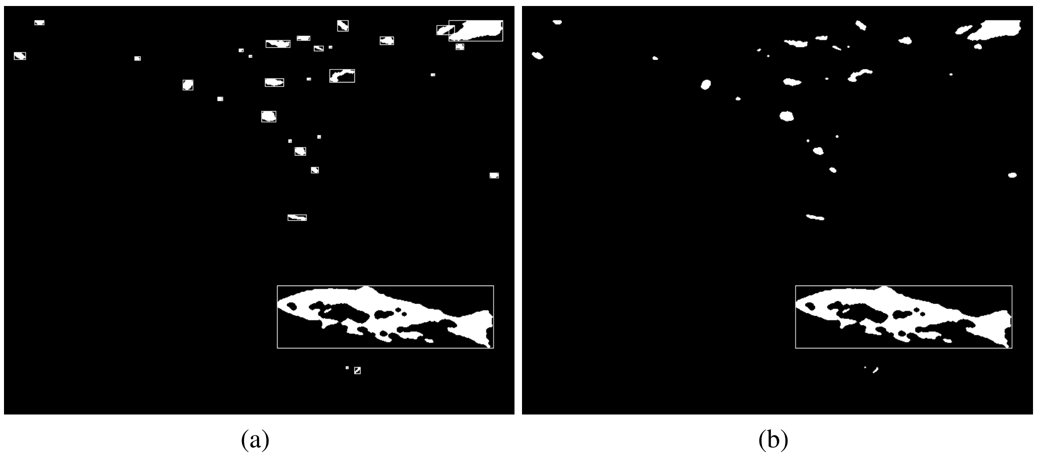


Figure 20. Object distinguishing: (a) All detected objects; (b) Detected fish.

4.3 Feature Extraction

The last part of fish detection system is feature extraction. As the result of detection will be later used for fish classification, the detection system should provide fish characterization as an outcome (Figure 21). Considering the experience of other fish recognition projects and according to the obtained detection results, it was decided to describe a fish in terms of its size and texture.

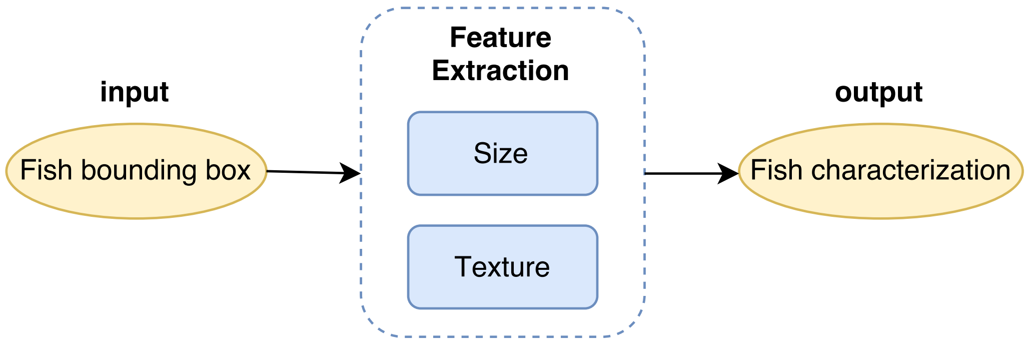
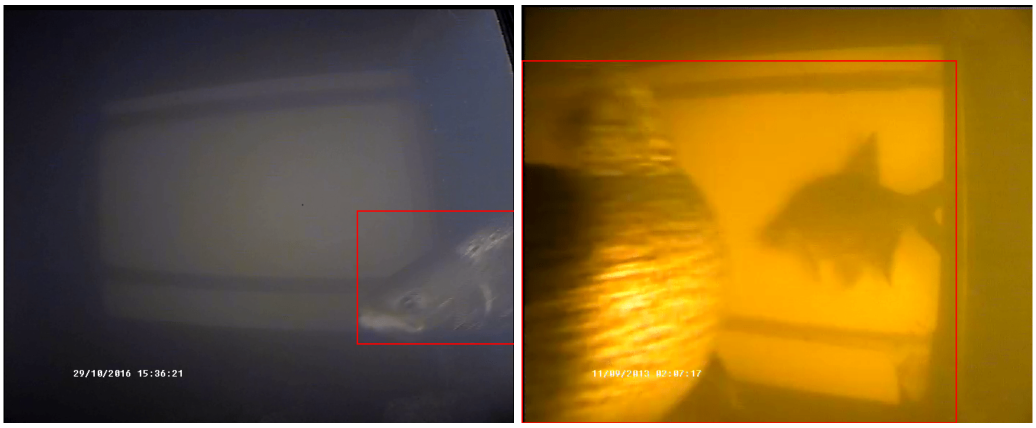


Figure 21. Feature Extraction subsystem structure.

4.3.1 Size

The simplest fish feature is the size. It can be measured by finding the width and the height of the fish bounding box. Here, the size is measured in pixels because to obtain the real fish size a camera calibration is needed. In videos the same fish is typically captured in multiple frames, so several bounding boxes are obtained. They may contain unreliable information, as when the fish is entering or leaving the field of view of the camera, only a part of it is seen (Figure 22(a)). Also a noise and incorrect detections may be included into measurements, which complicates the size estimation (Figure 22(b)). Thus, the fish size cannot be estimated by simple averaging, some extra assumptions are needed.



(a) (b)

Figure 22. Challenges in size estimation: (a) Only a part of the fish is captured; (b) Incorrect detection.

Depending on the frame rate and the time, during which the fish appears in the video, the number N of frames containing the whole image of the fish can be selected. Thus, only these frames participate in size estimation. So, N biggest values of bounding boxes are selected, then the resulting size of fish is calculated as the median value. Assuming that the number of noisy measurements is lower than the real ones, use of median instead of mean should reduce the influence of noise.

Although size estimation is regarded as a simple problem, there are many factors that may make this task impossible. For instance, if there is not a single frame in the video where the fish appears entirely. Or when all the detected bounding boxes contain wrong information, for example, due to occlusion. In these cases, size estimation needs additional knowledge or cannot be realized at all.

For classification purposes, raw values of fish height and width may not be helpful. If several fishes pass at different distances from the camera, their size in pixels may not coincide, although they are equal in the real life. Such features confuse classification algorithms, so some relative values, such as the height-to-length ratio, can be used.

4.3.2 Texture

Another fish feature that may be useful for species recognition is the texture. Texture characterizes the appearance of a certain area in an image. It is commonly used for object recognition and, specifically, for fish recognition, as various fish species have their own textures of scales. For using a texture as a fish feature it needs to represented in some numerical form. One of the possible forms is Local Binary Patterns (LBP) [45].

LBP is a feature vector which describes a texture, the steps for its construction are given in Algorithm 2.

Algorithm 2 : LBP.

```
The whole image or a specific region of interest is divided into cells (e.g. $16\times16$ pixels in a cell).
for all cells $c$ in the image do
    for all pixels $x$ in a cell $c$ do
        Visit 8 neighbours of $x$ clockwise.
        for all neighbours $n$ of a pixel $x$ do
            if $v(x) > v(n)$ then
                Write "0".
            else
                Write "1".
            end if
        end for
        Convert the obtained 8-digit binary number to integer.
    end for
    Compute a histogram over cell according to the obtained integer numbers.
end for
Concatenate histograms for all cells in the image. The resulted histogram gives the feature vector.
```

5 EXPERIMENTS AND RESULTS

5.1 Datasets

The fish detection system implemented in this work was tested on two datasets. Dataset 1 was collected by "Kymijoen vesi ja ympäristö ry" organization in 2013. It consists of four videos with different types of illumination and different water qualities. Dataset 2 was gathered by the same organization in 2016. It comprises six videos with various levels of visibility. Example frames from both datasets can be seen in Figure 23. A general description of the datasets is given in Tables 1 and 2.

Table 1. Videos from Dataset 1.

Video	**Resolution**	**Frames**	**Annotations**	**Species**
1	704×576	101	28	-
2	704×576	83	27	Bream
3	704×576	108	14	Bream
4	704×576	73	19	Bream

Table 2. Videos from Dataset 2.

Video	**Resolution**	**Frames**	**Annotations**	**Species**
1	704×576	138	34	Salmon
2	704×576	264	47	Whitefish
3	704×576	334	217	Salmon
4	704×576	240	33	Salmon
5	704×576	289	108	Salmon
6	704×576	275	97	Salmon

Both datasets consist of video sequences of passing fishes in the real underwater environment. The general scene of the camera position used in video capturing is shown in Figure 24 [46]. Fish enter the tube from the entering place, denoted as point A, and pass in front of the camera located at point B. The camera has angle of view of 72°, and the average distance between the camera and the fish is 55 cm. Nevertheless, in most of the videos fish pass at the distance which differs from the average value significantly. This fact should affect the size estimation process.

Some prior knowledge can be extracted from the datasets and used for tuning the parameters of the methods. All fish in videos move from right to left, which simplifies the process

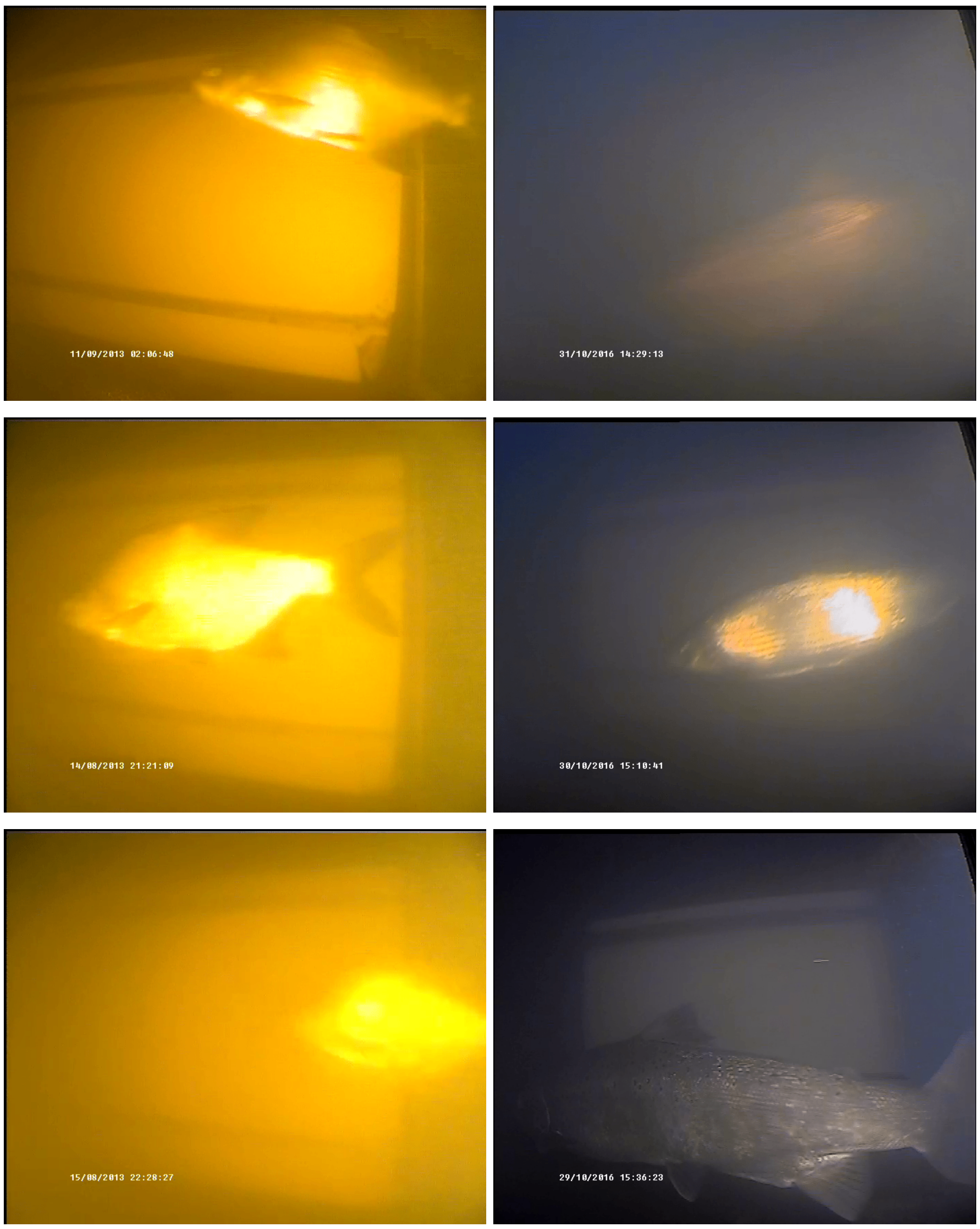


Figure 23. Example frames. The frames from the first column belong to Dataset 1, the second column frames are taken from Dataset 2.

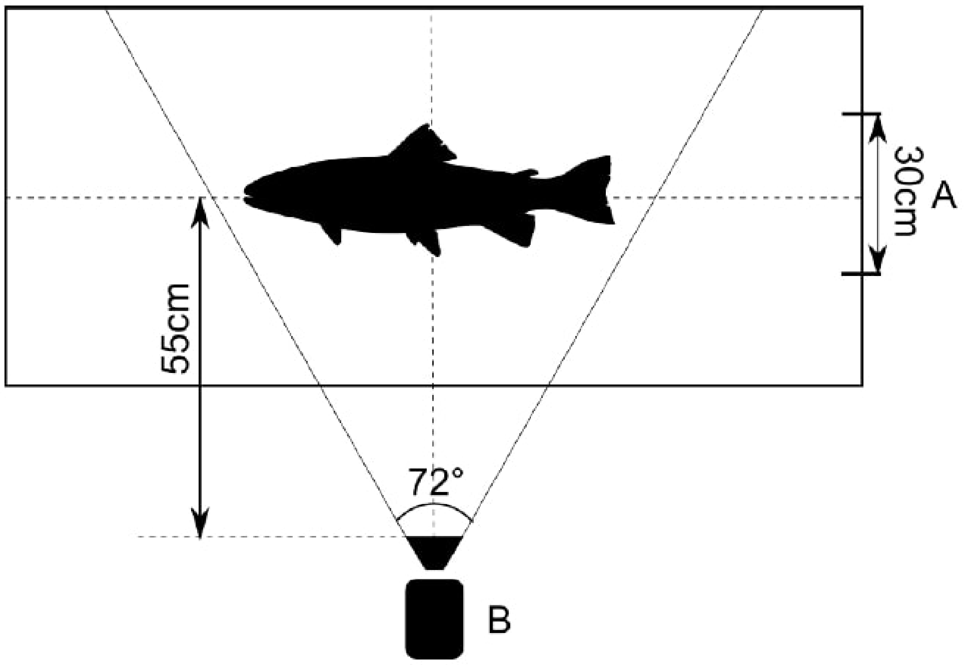


Figure 24. Scene of the camera position [46].

of fish identification described in Section 4.2.4. Moreover, each video contains only one passing fish in each frame, which also facilitate the detection.

To evaluate the performance of the implemented methods, a ground truth data needs to be generated. A ground truth is a desired ideal result, which can be obtained in various ways. The ground truth for the provided datasets was generated manually. Each video was annotated frame-by-frame with bounding boxes around fish (Figure 25). Unfortunately, the accuracy of the annotation is not perfect. Because of the low video quality, fish is not easily distinguished from the background, and it is hard to detect edges of the fish by the human eye. Therefore, a certain permissible error should be taken into account during the evaluation.

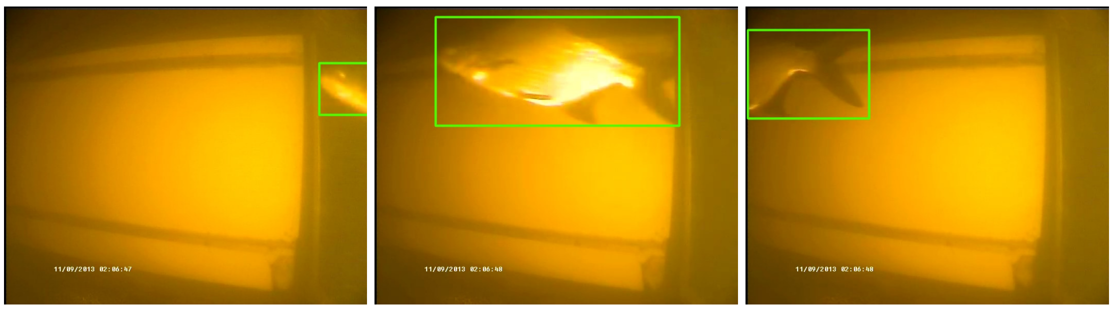


Figure 25. Example of annotated frames from Dataset 1. Green rectangles in the frames are the ground truth bounding boxes.

5.2 Evaluation Criteria

The proposed fish detection system can be based on either of the three BS methods described in Section 4.1. To select the most accurate implementation, their detection results need to be compared according to the defined evaluation criteria. This section includes the description of the evaluation process.

The result of the fish detection is evaluated in accordance with the ground truth. For each video the number of true positive (TP), true negative (TN), false positive (FP) and false negative (FN) frames is counted. The explanation of these values is given in Table 3. Rows of the table defines whether the frame contains fish or not, while columns correspond to the result of detection: if the fish was detected on the frame or not.

Table 3. Explanation of TP, TN, FP and FN values.

	Fish was detected	**Fish was not detected**
Frame contains fish	True Positive	False Negative
Frame does not contain fish	False Positive	True Negative

When counting the number of TP frames, it is necessary to make sure that the fish was detected correctly. That is, the found bounding box should match the ground truth bounding box. The most common evaluation metric used in object detection is the Intersection over Union (IoU), also known as Jaccard Index. It determines how two bounding boxes are similar to each other, and, therefore, is used to calculate the accuracy of the detection. By selecting a threshold value, the detected bounding box can be considered as the correct one, if its IoU value is higher than this threshold. IoU is defined as a ratio:

$$IoU = \frac{A_i}{A_u}, \tag{13}$$

where A_i is the area of intersection of the detected and ground truth bounding boxes, while A_u is the area of their union (Figure 26).

Finally, for comparing the results of detection, the following measures were calculated:

- **Precision** (P) determines how useful the detection is. High precision means that the algorithm returns more true detected fish than false detected.

$$P = \frac{TP}{TP + FP} \tag{14}$$

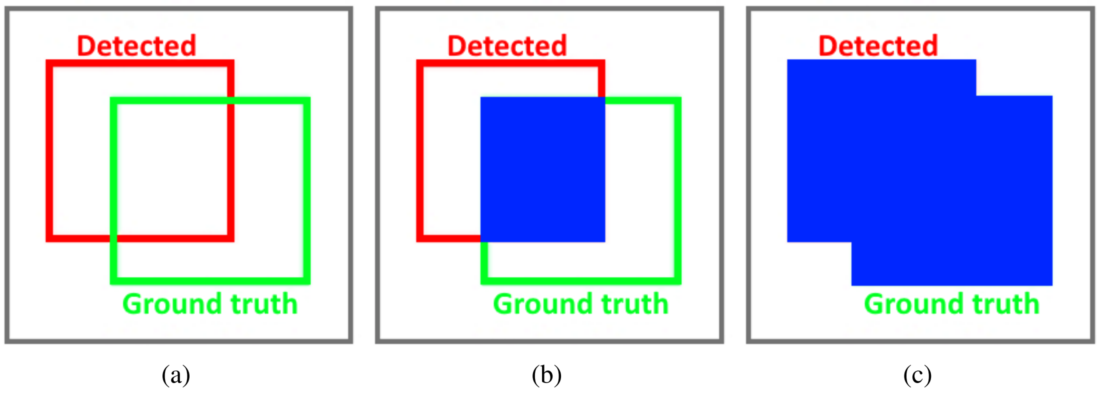


Figure 26. IoU: (a) Detected (red) and ground truth (green) bounding boxes; (b) Area of intersection (blue); (c) Area of union (blue).

- **Recall** (R) defines how many truly detected fish were returned.

$$R = \frac{TP}{TP + FN} \tag{15}$$

- **F1 score** (F_1) is a harmonic mean of P and R. It defines the detection accuracy.

$$F_1 = 2\frac{P \times R}{P + R} \tag{16}$$

Although the outcome of the proposed system is the fish characterization, it is not possible to evaluate the accuracy of fish size estimation and texture extraction due to the lack of available data and ground truth. Therefore, this thesis concentrates mainly on the evaluation of the detection accuracy, while the efficiency of the fish characterization extraction can be considered in the future work.

5.3 Implementation and Method Parameters

The fish detection system proposed in this work was implemented in Python language version 2.7.12 using the OpenCV 3.2.0 library [47].

The implementations of Adaptive GMM and KDE methods introduced in Section 4.1 were taken from the OpenCV library and were used with default parameters. The default parameters are suitable for the current task, because they allow the algorithms to be both sensitive and noise-resistant. The descriptions of these implementations are given in [48] and [49] respectively. The third method, ViBe, was implemented manually with default parameters specified in [39]. The only parameter that was set individually for each video is radius $\mathcal{R}$, since it is responsible for the sensitivity of the algorithm. The lower values of $\mathcal{R}$ allows to detect motion even when the colors of background and foreground are very close. But lower values of $\mathcal{R}$ also increases the noise. That is why the $\mathcal{R}$ value was chosen for each dataset in accordance with how much the fish stands out against the background in videos (Table 4).

Table 4. Radius $\mathcal{R}$ for ViBe algorithm. The visibility of fish in the datasets is shown in Figure 23.

Dataset	$\mathcal{R}$
1	15
2	10

Table 5 contains parameters used for Post-Processing for BS: the size of the median filter, the size of the opening structuring element and the size of the closing structuring element. These values were chosen empirically, according to the level of noise and misdetections remained after BS. Parameters used in Object Detection are indicated in Table 6. After Post-Processing for BS all objects, whose area is smaller than $area1$ value, are deleted. Then the regions are combined using the threshold values $dist$ and $height$. And, finally, united objects, with area smaller than $area2$, are deleted.

Table 5. Parameters for Post-Processing for BS. Three values in each cell refer to Adaptive GMM, KDE and ViBe algorithms respectively.

Dataset	$median$			$opening$			$closing$		
	GMM	*KDE*	*ViBe*	*GMM*	*KDE*	*ViBe*	*GMM*	*KDE*	*ViBe*
1	5	3	3	5	5	5	7	7	7
2	5	3	3	7	5	5	7	7	7

Table 6. Parameters for Object Detection. The value in each cell refers to all the three algorithms: Adaptive GMM, KDE and ViBe. All values are given in pixel units.

Dataset	$area1$	$area2$	$dist$	$height$
1	100	500	450	320
2	200	800	450	320

5.4 Results

During the experiments two datasets were tested with three BS methods. Table 7 includes the results of the experiments, where detected bounding boxes are compared to the ground truth, and the detection is assumed to be successful if their IoU value is higher than 0.5. The choice of such a threshold is due to the fact that the accuracy of a ground truth annotation is not perfect, so IoU values cannot be close to 1. This parameter can vary depending on how accurately fish bounding boxes should be detected.

Table 7. Results of the experiments with threshold for IoU = 0.5. Three values in each cell refer to P, R and F_1 respectively. Best results according to F_1 are in bold.

Video	**GMM**			**KDE**			**ViBe**		
Dataset 1	P	R	F_1	P	R	F_1	P	R	F_1
1	0.82	0.82	0.82	0.89	0.89	0.89	0.96	0.89	**0.93**
2	0.24	0.44	0.31	0.80	0.74	0.77	0.95	0.74	**0.83**
3	0.73	0.79	**0.76**	0.50	0.57	0.53	0.67	0.57	0.62
4	0.05	0.11	0.06	0.25	0.32	0.28	0.70	0.74	**0.72**
Overall	0.35	0.55	0.42	0.63	0.67	0.65	0.85	0.76	**0.80**
Dataset 2									
1	0.78	0.62	**0.69**	0.58	0.41	0.48	0.50	0.41	0.45
2	0.43	0.49	0.46	0.81	0.64	**0.71**	0.82	0.57	0.67
3	0.61	0.74	**0.67**	0.65	0.51	0.57	0.54	0.45	0.49
4	0.90	0.85	**0.87**	0.52	0.36	0.43	0.50	0.36	0.42
5	0.54	0.46	**0.50**	0.22	0.07	0.11	0.43	0.17	0.24
6	0.86	0.59	**0.70**	0.07	0.03	0.04	0.50	0.09	0.16
Overall	0.64	0.63	**0.63**	0.53	0.33	0.41	0.54	0.33	0.41
Overall for both datasets	0.58	0.62	**0.60**	0.55	0.38	0.45	0.60	0.39	0.48

It can be seen, that both Adaptive GMM and ViBe algorithms show acceptable results on different videos, while KDE method show average results for almost all videos. Since the quality of the video and the presence of various attributes (e.g. noise, illumination changes, background movements) can vary greatly between and within the datasets, the detection results also differ significantly. In general, results for Dataset 1 are better than for Dataset 2. This may be because of the fact, that videos from the second dataset are darker, and fish on most of the frames stays in shade and is not fully seen. Also videos from Dataset 2 include situations, when the fish stays motionless during a long set of frames and BS algorithm cannot detect it. Examples of the detection results can be seen on Figures 27 and 28. Figures 29, 30 and 31 show how the accuracy of the detection depends on the IoU threshold value.

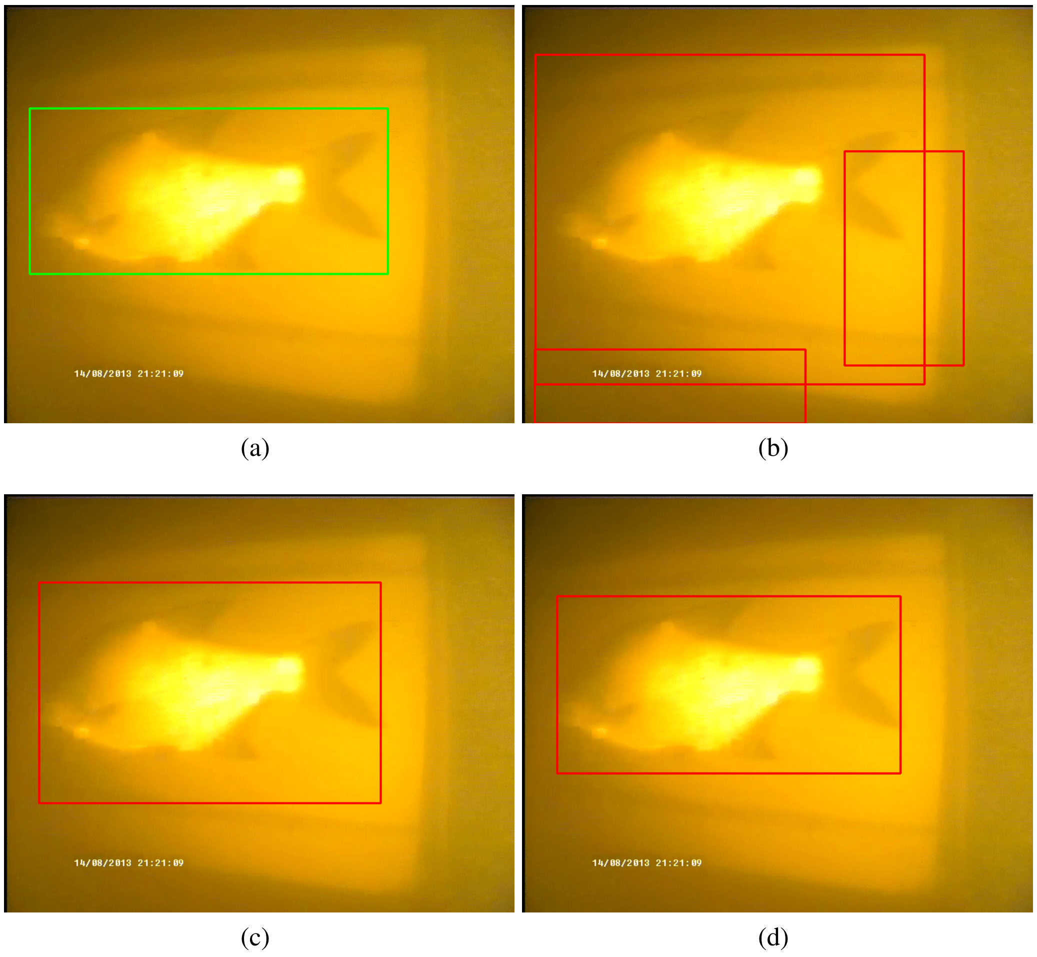


Figure 27. The result of detection (Video 2 from Dataset 1): (a) Ground truth; (b) Adaptive GMM; (c) KDE; (d) ViBe.

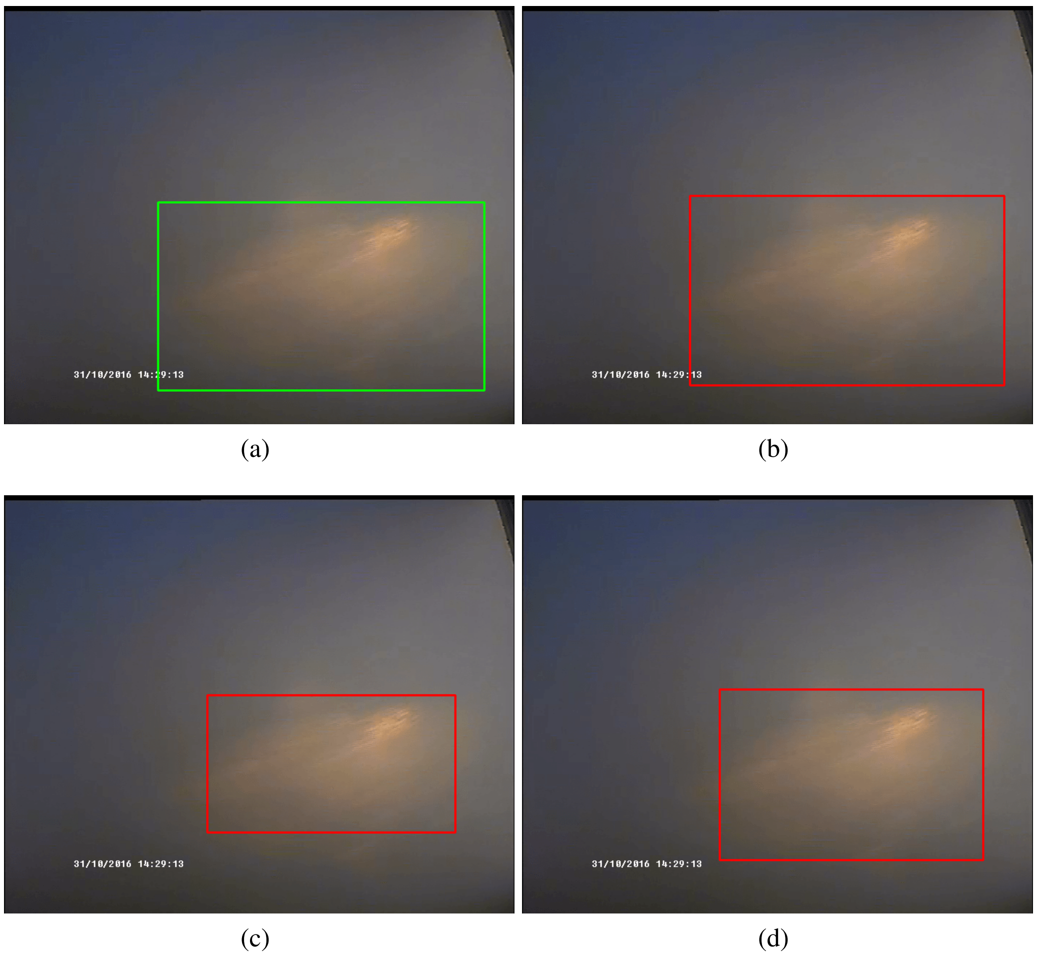


Figure 28. The result of detection (Video 1 from Dataset 2): (a) Ground truth; (b) Adaptive GMM; (c) KDE; (d) ViBe.

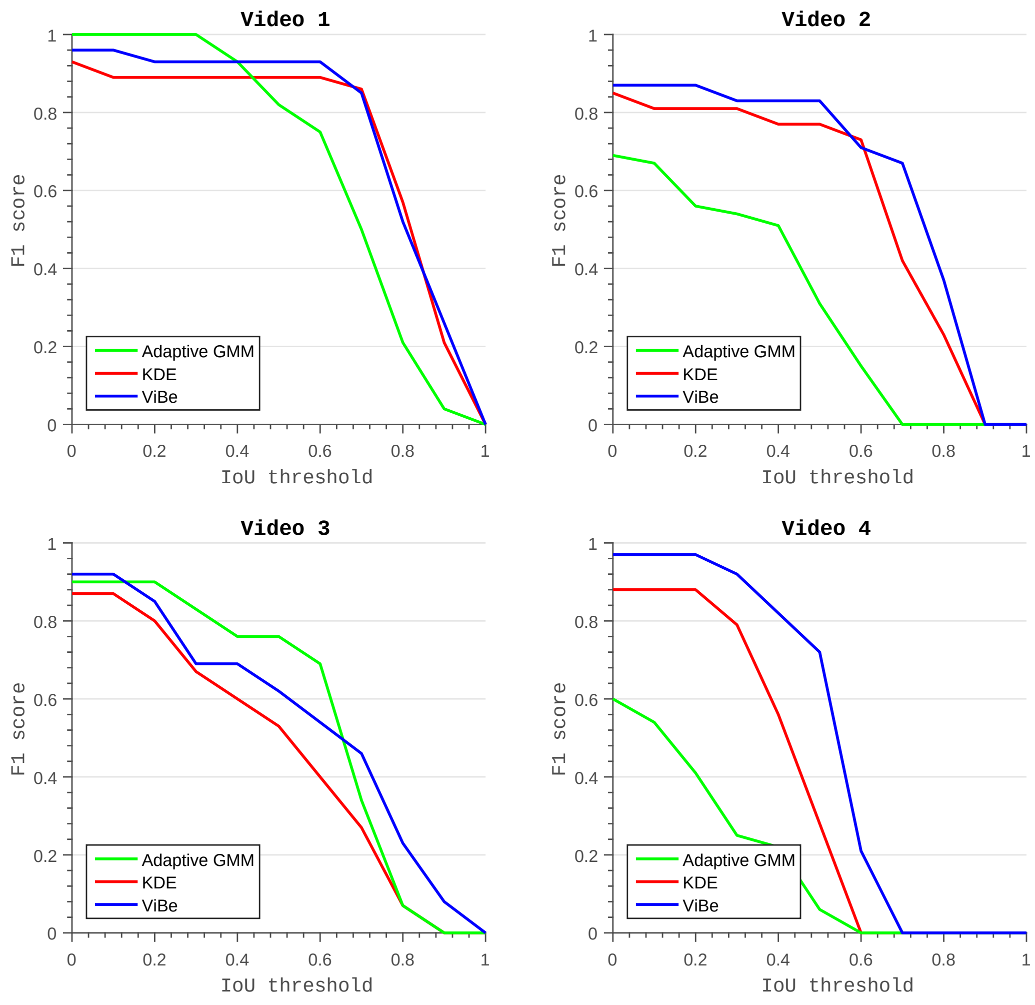


Figure 29. The dependence of the F1 score on the IoU threshold for Dataset 1.

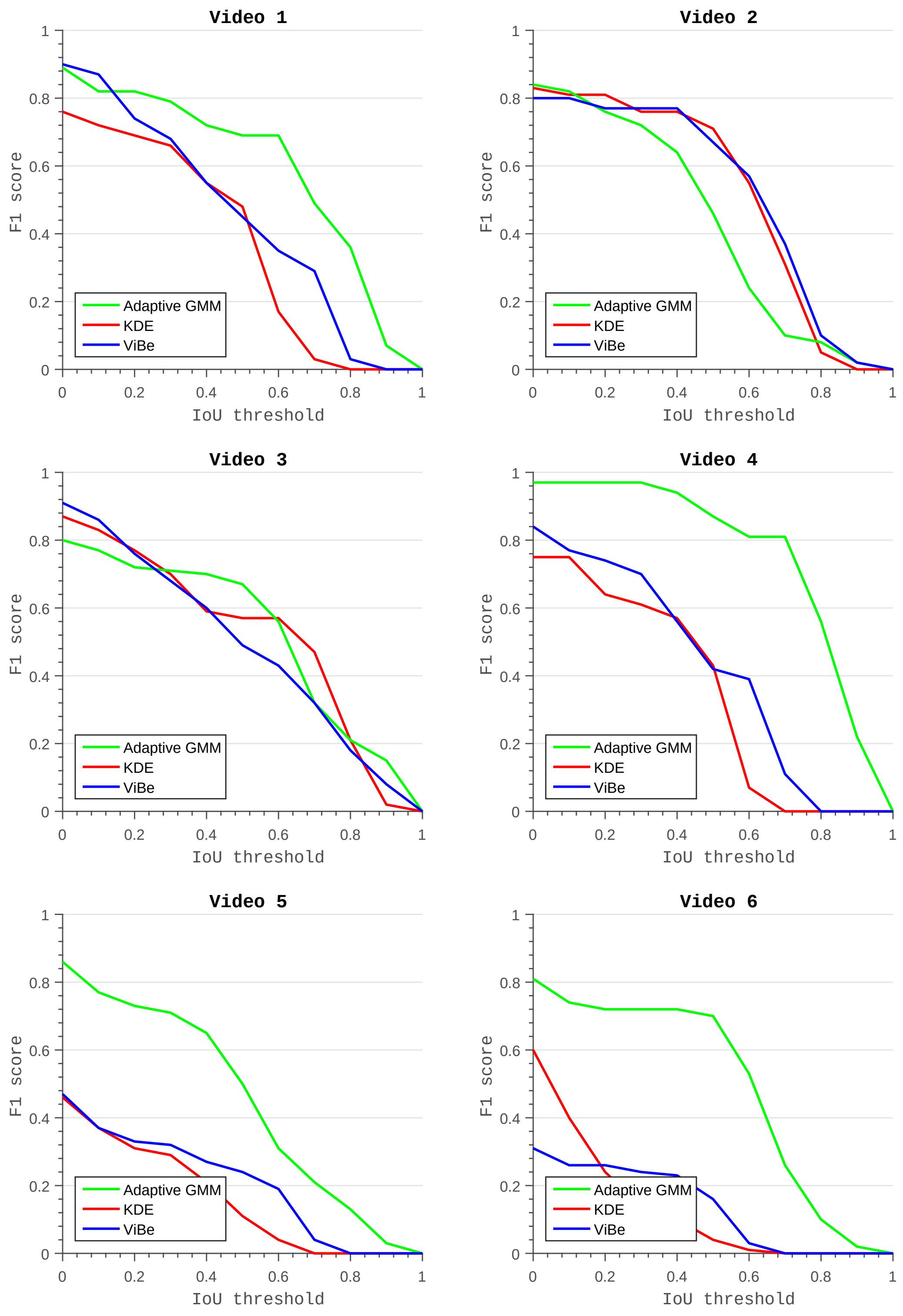


Figure 30. The dependence of the F1 score on the IoU threshold for Dataset 2.

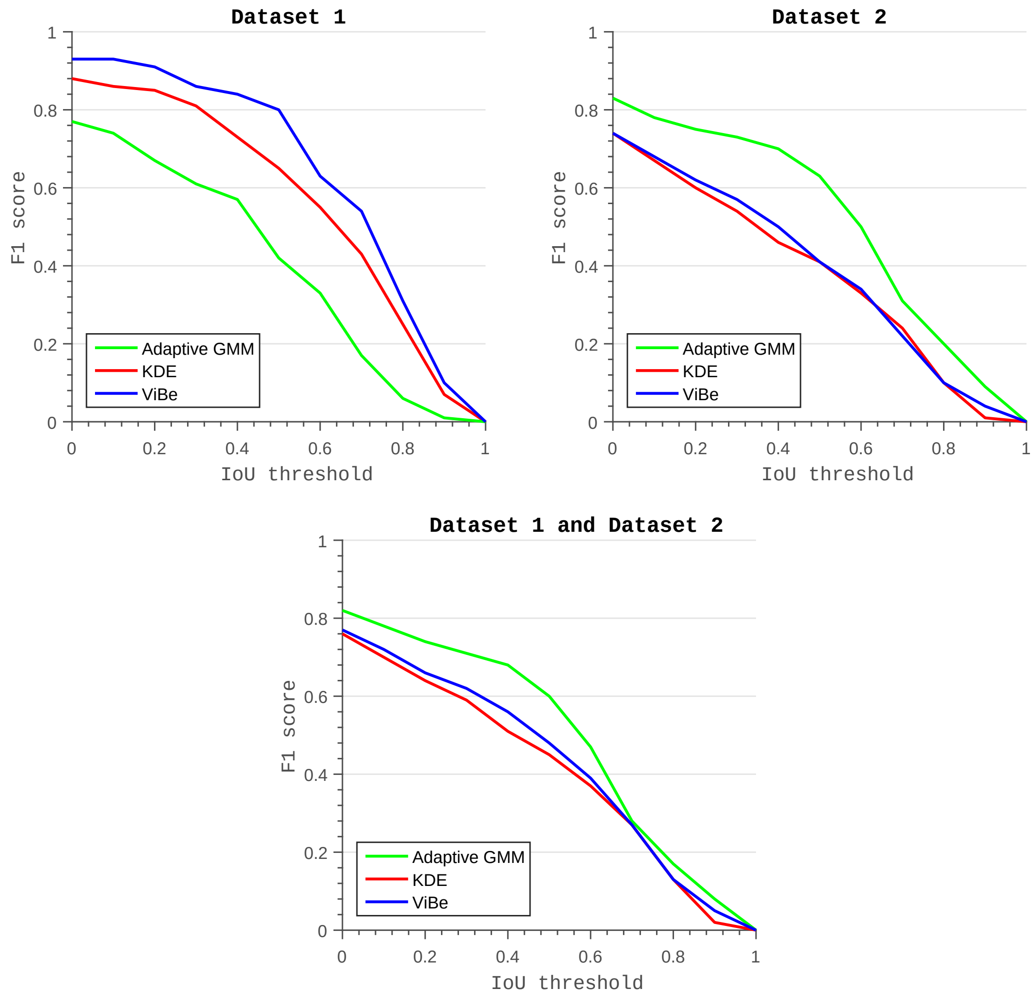


Figure 31. The dependence of the F1 score on the IoU threshold for overall results.

Experiments also include the texture extraction step, in which the LBP feature vector is found for the detected fish. The results of the texture extraction for different fish species are shown in Figure 32. Although the quality of the videos does not allow to extract precise textures of the real fish scale patterns, a certain tendency on the histograms can be seen. Histograms of bream species have higher bins around 200-250 values, while salmon histograms have peaks in the range from 0 to 40. The lack of videos with different species and the fact, that the general quality, lightness and level of noise differ significantly for two datasets, prevents an accurate analysis of results. But it can be assumed that the texture can be possibly used for classification of these two species.

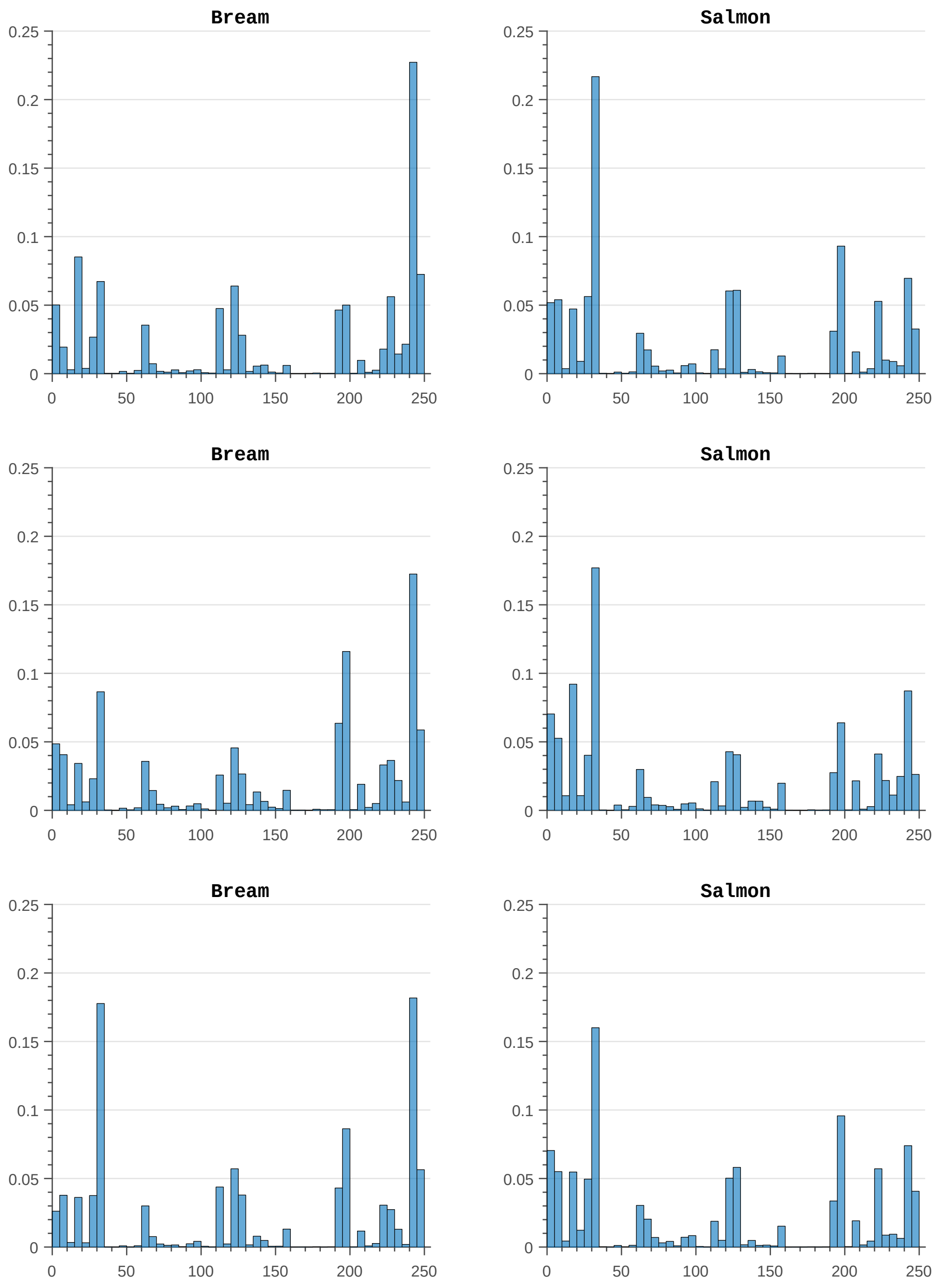


Figure 32. Normalized histograms representing LBP feature vectors. The histograms from the first column belong to Dataset 1, the second column histograms belong to Dataset 2.

6 DISCUSSION

6.1 Results

The goal of this work was to build a system, which is able to detect fish in underwater videos and extract fish characteristics that are useful for recognition purposes. The proposed system detects moving objects using the BS approach, distinguish fish from other objects and characterize a fish in terms of its size and texture. The system was tested on two datasets and evaluated by the detection accuracy.

Considering the obtained results, it can be concluded, that the proposed approach can be successfully applied for the fish detection. Selection of a suitable BS method should be based on the type of the videos. ViBe method shows better resistance for noise and ability to adapt for illumination changes and background movements. Adaptive GMM method is capable to detect motion even when the color of foreground is close to the background, but its results may contain a lot of noise. KDE shows acceptable results on different videos, but may be less effective on the used datasets than the other methods. As a conclusion, it can be said that Adaptive GMM is the most motion-sensitive algorithm, ViBe is the most noise-resistant, while KDE shows an average between detection and noise resistance.

The main problem that obstructs the fish detection is the quality of the video. In cases where the fish is distinguishable against the background, the detection accuracy reaches 72-93% (Table 7). Unfortunately, most of the underwater videos have a low level of visibility, and therefore the fish is barely noticeable. For these cases, the accuracy fluctuates between 50% and 87% (Table 7). Also, the provided datasets do not contain enough videos with various fish species, so it is impossible to evaluate, how the extracted fish features describe different species. These issues may be considered in the future work.

6.2 Future Work

An outcome of the fish detection system is the fish characterization, which can be used for fish species recognition. Therefore, future work can be concentrated mainly on the classification task. Given the sufficient number of training results, a classification system based on fish size and texture can be build. Also, the proposed process of size estimation can be improved by introducing the information about camera parameters, which allows to perform the camera calibration.

7 CONCLUSION

In this thesis, the system for detecting fish in underwater videos were proposed. The three BS algorithms, called Adaptive GMM, KDE and ViBe, were used for moving object detection. The results showed that depending on the type of the video, all reviewed BS methods can be successfully applied for fish detection. The main parameters on which the BS algorithm can be chosen is the presence of noise, illumination changes, level of visibility and presence of moving background. As the result, the fish detection system returns the fish characterization, which includes the estimated fish size and its texture properties. These features can be later used for species classification.

REFERENCES

[1] Carl Schlieper. *Research methods in marine biology*. Sidgwick & Jackson, 1972.

[2] Concetto Spampinato, Yun-Heh Chen-Burger, Gayathri Nadarajan, and Robert B Fisher. Detecting, tracking and counting fish in low quality unconstrained underwater videos. In *Proceedings of the Third International Conference on Computer Vision Theory and Applications*, volume 2, pages 514–519. Citeseer, 2008.

[3] JL Thorley, DMR Eatherley, AB Stephen, I Simpson, JC MacLean, and AF Youngson. Congruence between automatic fish counter data and rod catches of atlantic salmon (salmo salar) in scottish rivers. *ICES Journal of Marine Science: Journal du Conseil*, 62(4):808–817, 2005.

[4] Helge Balk. *Development of Hydroacoustic Methods for Fish Detection in Shallow Water*. PhD thesis, Faculty of Mathematics and Natural Science, University of Oslo, 2001.

[5] Robert T Collins, Alan J Lipton, Takeo Kanade, Hironobu Fujiyoshi, David Duggins, Yanghai Tsin, David Tolliver, Nobuyoshi Enomoto, Osamu Hasegawa, Peter Burt, et al. A system for video surveillance and monitoring. 2000.

[6] Suman Srinivasan, Haniph Latchman, John Shea, Tan Wong, and Janice McNair. Airborne traffic surveillance systems: video surveillance of highway traffic. In *Proceedings of the ACM 2nd International Workshop on Video Surveillance & Sensor Networks*, pages 131–135. ACM, 2004.

[7] Nils T Siebel and Steve Maybank. The advisor visual surveillance system. In *Proceedings of the 8th European Conference on Computer Vision Workshop Applications of Computer Vision*, volume 1, 2004.

[8] Peter Christiansen, Kim Arild Steen, Rasmus Nyholm Jørgensen, and Henrik Karstoft. Automated detection and recognition of wildlife using thermal cameras. *Sensors*, 14(8):13778–13793, 2014.

[9] Kinjal A Joshi and Darshak G Thakore. A survey on moving object detection and tracking in video surveillance system. *International Journal of Soft Computing and Engineering*, 2(3):44–48, 2012.

[10] Soharab Hossain Shaikh, Khalid Saeed, and Nabendu Chaki. *Moving Object Detection Using Background Subtraction*. Springer, 2014.

[11] Shireen Y Elhabian, Khaled M El-Sayed, and Sumaya H Ahmed. Moving object detection in spatial domain using background removal techniques-state-of-art. *Recent patents on computer science*, 1(1):32–54, 2008.

[12] Yaser Sheikh, Omar Javed, and Takeo Kanade. Background subtraction for freely moving cameras. In *Proceedings of the 2009 IEEE 12th International Conference on Computer Vision*, pages 1219–1225. IEEE, 2009.

[13] Vijay Mahadevan and Nuno Vasconcelos. Background subtraction in highly dynamic scenes. In *Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–6. IEEE, 2008.

[14] Tejinder Thind Bharti. Background subtraction technique review. *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, 2(3):166–168, 2013.

[15] Massimo Piccardi. Background subtraction techniques: a review. In *Proceedings of the 2004 IEEE international conference on Systems, man and cybernetics*, volume 4, pages 3099–3104. IEEE, 2004.

[16] Thierry Bouwmans. Recent advanced statistical background modeling for foreground detection-a systematic survey. *Recent Patents on Computer Science*, 4(3):147–176, 2011.

[17] Yannick Benezeth, Pierre-Marc Jodoin, Bruno Emile, Hélène Laurent, and Christophe Rosenberger. Comparative study of background subtraction algorithms. *Journal of Electronic Imaging*, 19(3):033003–033003, 2010.

[18] Andrews Sobral and Antoine Vacavant. A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos. *Computer Vision and Image Understanding*, 122:4–21, 2014.

[19] Rita Cucchiara, Costantino Grana, Massimo Piccardi, and Andrea Prati. Detecting moving objects, ghosts, and shadows in video streams. *IEEE transactions on pattern analysis and machine intelligence*, 25(10):1337–1342, 2003.

[20] Janne Heikkilä and Olli Silvén. A real-time system for monitoring of cyclists and pedestrians. *Image and Vision Computing*, 22(7):563–570, 2004.

[21] Christopher Richard Wren, Ali Azarbayejani, Trevor Darrell, and Alex Paul Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transactions on pattern analysis and machine intelligence*, 19(7):780–785, 1997.

[22] Chris Stauffer and W Eric L Grimson. Adaptive background mixture models for real-time tracking. In *Proceedings of the 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 246–252. IEEE, 1999.

[23] Ahmed Elgammal, David Harwood, and Larry Davis. Non-parametric model for background subtraction. In *Proceedings of the 6th European Conference on Computer Vision*, pages 751–767. Springer, 2000.

[24] Hongxun Zhang and De Xu. Fusing color and texture features for background model. *Fuzzy Systems and Knowledge Discovery*, pages 887–893, 2006.

[25] Lucia Maddalena and Alfredo Petrosino. A self-organizing approach to background subtraction for visual surveillance applications. *IEEE Transactions on Image Processing*, 17(7):1168–1177, 2008.

[26] Kyungnam Kim, Thanarat H Chalidabhongse, David Harwood, and Larry Davis. Real-time foreground–background segmentation using codebook model. *Real-time imaging*, 11(3):172–185, 2005.

[27] Nuria M Oliver, Barbara Rosario, and Alex P Pentland. A bayesian computer vision system for modeling human interactions. *IEEE transactions on pattern analysis and machine intelligence*, 22(8):831–843, 2000.

[28] Simone Palazzo, Isaak Kavasidis, and Concetto Spampinato. Covariance based modeling of underwater scenes for fish detection. In *Proceedings of the 2013 20th IEEE International Conference on Image Processing (ICIP)*, pages 1481–1485. IEEE, 2013.

[29] Fiona H Evans. Detecting fish in underwater video using the em algorithm. In *Proceedings of the 2003 International Conference on Image Processing (ICIP)*, volume 3, pages III–1029. IEEE, 2003.

[30] Zoran Zivkovic. Improved adaptive gaussian mixture model for background subtraction. In *Proceedings of the 17th 2004 International Conference on Pattern Recognition (ICPR)*, volume 2, pages 28–31. IEEE, 2004.

[31] Sergios Theodoridis and Konstantinos Koutroumbas. *Pattern recognition (4th Edition)*. Academic press, 2003.

[32] Richard O Duda, Peter E Hart, and David G Stork. *Pattern classification (2nd Edition)*. New York: Wiley, 2001.

[33] Mutasem K Alsmadi, Khairuddin B Omar, Shahrul A Noah, and Ibrahim Almarashdeh. Fish recognition based on robust features extraction from size and shape measurements using neural network. *Journal of Computer Science*, 6(10):1088, 2010.

[34] Concetto Spampinato, Daniela Giordano, Roberto Di Salvo, Yun-Heh Jessica Chen-Burger, Robert Bob Fisher, and Gayathri Nadarajan. Automatic fish classification for underwater species behavior understanding. In *Proceedings of the first ACM international workshop on Analysis and retrieval of tracked events and motion in imagery streams*, pages 45–50. ACM, 2010.

[35] Yi-Hao Hsiao, Chaur-Chin Chen, Sun-In Lin, and Fang-Pang Lin. Real-world underwater fish recognition and identification, using sparse representation. *Ecological Informatics*, 23:13–21, 2014.

[36] Meng-Che Chuang, Jenq-Neng Hwang, and Kresimir Williams. Supervised and unsupervised feature extraction methods for underwater fish species recognition. In *Proceedings of the 2014 ICPR Workshop on Computer Vision for Analysis of Underwater Imagery (CVAUI)*, pages 33–40. IEEE, 2014.

[37] Yong Xu, Jixiang Dong, Bob Zhang, and Daoyun Xu. Background modeling methods in video analysis: A review and comparative evaluation. *CAAI Transactions on Intelligence Technology*, 1(1):43–60, 2016.

[38] Zoran Zivkovic and Ferdinand Van Der Heijden. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern recognition letters*, 27(7):773–780, 2006.

[39] Olivier Barnich and Marc Van Droogenbroeck. Vibe: A universal background subtraction algorithm for video sequences. *IEEE Transactions on Image processing*, 20(6):1709–1724, 2011.

[40] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing (3rd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2006.

[41] Linda Shapiro and George C Stockman. Computer vision. *Prentice Hall*, 2001.

[42] Insaf Setitra and Slimane Larabi. Background subtraction algorithms with post-processing: A review. In *Proceedings of the 2014 22nd International Conference on Pattern Recognition (ICPR)*, pages 2436–2441. IEEE, 2014.

[43] Steven S. Beauchemin and John L. Barron. The computation of optical flow. *ACM computing surveys (CSUR)*, 27(3):433–466, 1995.

[44] Gunnar Farnebäck. Two-frame motion estimation based on polynomial expansion. *Image analysis*, pages 363–370, 2003.

[45] Li Wang and Dong-Chen He. Texture classification using texture spectrum. *Pattern Recognition*, 23(8):905–910, 1990.

[46] Ekaterina Lantsova, Tatiana Voitiuk, Tatiana Zudilova, and Arto Kaarna. Using low-quality video sequences for fish detection and tracking. In *Proceedings of the 2016 SAI Computing Conference (SAI)*, pages 426–433. IEEE, 2016.

[47] OpenCV 3.2.0 Python Tutorials. `http://docs.opencv.org/3.2.0/d6/d00/tutorial_py_root.html`, 2016. [Online; Accessed: May 2017].

[48] OpenCV cv::BackgroundSubtractorMOG2. `http://docs.opencv.org/3.2.0/d7/d7b/classcv_1_1BackgroundSubtractorMOG2.html`, 2016. [Online; Accessed: May 2017].

[49] OpenCV cv::BackgroundSubtractorKNN. `http://docs.opencv.org/3.2.0/db/d88/classcv_1_1BackgroundSubtractorKNN.html`, 2016. [Online; Accessed: May 2017].