2018 24th International Conference on Pattern Recognition (ICPR)
Beijing, China, August 20-24, 2018

Fish Detection from Low Visibility Underwater Videos

Violetta Shevchenko, Tuomas Eerola, and Arto Kaarna
Machine Vision and Pattern Recognition Laboratory
School of Engineering Science
Lappeenranta University of Technology
Lappeenranta, Finland
Email: violetteshev@gmail.com, tuomas.eerola@lut.fi, arto.kaarna@lut.fi

***Abstract*—Counting and tracking fish populations is important for conservation purposes as well as for the fishing industry. Various non-invasive 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 paper we present the fish detection framework for noisy videos captured in water with low visibility. For this purpose, we compare three background subtraction methods for the task. Moreover, we propose necessary post-processing steps and heuristics to detect the fish and separate them from other moving objects. 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 challenging datasets. The proposed method will form a basis for the future development of fish species identification methods.**

I. Introduction

The problem of tracking and studying fish populations is crucial for fishing conservation and industry. However, this task is complicated and usually requires large amount of time and expenses. The traditional way to determine the situation under the water is casting nets for collecting and examining fish or direct human underwater observation [1]. Such methods are not able to provide a comprehensive observation, cannot capture the real fish state and behavior, and demands expensive equipment and human resources e.g. in examining spring migration of fish. 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 and freshwater life observation [2].

One of the most commonly used approaches for underwater observation is to collect videos and analyze 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.

The fish counting task is the simplest part of underwater observation and various automatic fish counters already exist. They usually implement approaches, based on different physical principles, 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]. These fish counters 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 fish species. Therefore, researcher nowadays tend to use computer vision techniques to build more robust and versatile fish counting systems.

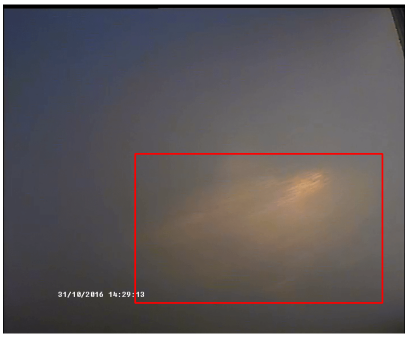


Fig. 1: Fish detection from a low visibility video.

The crucial step in building computer vision-based counting systems is fish detection. However, it is often a challenging task to find a robust and high-performance method for moving object detection. Usually video sequences contain various attributes that make the task even more difficult. Some of the possible problems connected with the detection are illumination changes, dynamic background, camera motion, occlusion, noise, low visibility, etc. The recent works in the field of fish detection have shown that Background Subtraction (BS) methods are able to cope with the most of distortions that are typical for underwater videos (see e.g. [2]). These methods are based on background modeling which makes it possible to distinguish foreground pixels belonged to moving objects.

In this paper, the system for detecting fish in low visibility underwater videos is presented. The system utilizes background subtraction to detect moving objects and various heuristics to distinguish passing fishes from all other objects.

978-1-5386-3788-3/18/$31.00 ©2018 IEEE

The proposed system was tested on two datasets collected in natural environment. The data contain videos from muddy water with various types of illumination and visibility (see Fig. 1). The proposed method can be applied to videos with different quality.

II. Related Work

Some methods to detect and count fish in underwater videos can be found in the literature. In [5], an Expectation-maximization (EM) algorithm was proposed for fish detection. The shape of each fish is assumed to be a multivariate Gaussian and images are modeled as Gaussian Mixture Model (GMM). The parameters of GMM, including the number of fish, are estimated using an EM algorithm.

A fully automated video processing system for underwater video surveillance was presented in [2]. This approach combines BS methods with tracking algorithms and can be applied to both stable and dynamic background scenes.

Fish counting systems can be improved into fully autonomous surveillance systems by adding fish recognition implementation. A method proposed in [6] allows to calculate the geometry of the fish, which consists in various size and shape measurement, and categorize fish into ”poison” or ”non-poison” families. System described in [7] uses affine invariant features, such as texture that is received from the gray-level histogram, the Gabor filters and the gray-level co-occurrence matrices, to classify fish. The work [8] comprises the results of comparison of two fish feature extraction methods, namely the supervised that use predefined features like shape or color and unsupervised that learn the features directly from images.

In [9], a method similar to ours was proposed. The method contains an adaptive background model to obtain fish silhouettes, as well as, a classification method to distinguish fish from other moving objects. However, the video data considered in the study is of high quality and the fish are easily distinguishable from the background.

III. Fish Detection System

A. Background Subtraction

The first step of the proposed system is moving object detection. To solve this problem we utilize Background Subtraction (BS) methods as they are relatively simple and fast methods and able to cope with various types of movements in video sequences. 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. Most of the BS methods are implemented by the same scheme, which consists of three main steps:

- Background modeling. 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 several number of successive frames.
- Foreground detection. After the background image is obtained, all subsequent frames are processed based on the 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 it is useful to update the background model with each new frame, since typically the background does not remain unchanged throughout the whole video. The model must be adaptive to various changes so that they do not affect the result of detection.

An extensive survey on the performance of various BS methods is given in [10], [11]. Three algorithms were chosen for further study based on accuracy, robustness, ability to cope with noise, moving background and illumination changes, and computational speed: Adaptive Gaussian Mixture model GMM [12], Kernel Density Estimation (KDE) [13] and Visual Background Extractor (ViBe) [14].

1) Adaptive Gaussian Mixture Model: The first selected algorithm is based on the GMM and described in [12], [13]. This method refers to the statistical BS approach which means that the background, due to its regular behavior, is assumed to be well described by a statistical model. In this case, a scene model is built by estimating a probability density function (PDF) for each pixel. As pixels usually have complex distributions, GMM is used instead of a single normal distribution.

Pixels which do not fit to the model are considered as a foreground. One of the most important properties of this algorithm is the ability to automatically choose the number of Gaussians which makes it fully adaptive 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 is justified.

2) Kernel Density Estimation: In [13], a BS method based on KDE was introduced. It is a non-parametric statistical approach in which the PDF is estimated directly from the data. This method is proven to be reasonably effective, when the number of foreground pixels is relatively small. This is the case with typical fish detection videos, where usually only one moving fish is presented in each frame and, therefore, only small fraction of pixels belong to the foreground.

3) Visual Background Extractor: A universal BS algorithm, called ViBe, was presented in [14]. It has been shown 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 have been taken from the same pixel location or its neighborhood of earlier frames. This set is compared with the captured pixel value and based on 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 neighbors. ViBe can instantly adjust to most possible situations, such as fast or gradual change of illumination, appearance of new background objects or sudden background movements. It is also able to quickly correct negative consequences in the case, when the first frame used for background initialization contains objects of motion. Another key feature of ViBe is that a background model is initialized by using a single frame. It speeds up the initialization significantly, allows to start the detection from the second frame and, makes it possible to process short videos effectively.

B. Fish Detection

Outcomes of the BS subsystem are binary images with the detected foreground. To separate fish from all other moving objects the following steps are proposed:

1) Post-Processing for BS: The raw output of BS algorithm generally contains a large amount of noise in the form of wrongly detected foreground or background pixels. In order to eliminate these distortions three methods are applied. Median filter is used to reduce misdetected foreground pixels that appear as impulse noise. Morphological opening is further used to remove small objects which are most likely noise, but do not alter large objects. And finally, morphological closing is used to fill the holes inside objects. The presented approach is a common procedure used in post-processing for BS [15]. It helps to get rid of the vast majority of misclassified pixels which facilitates the further analysis.

2) Optical Flow: Although BS methods find the regions of motion, they do not provide any specific information about the motion itself. However, in many cases, it is useful to know how 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. To measure movement we utilize optical flow [16]. The approach is based on assumptions that pixel values do not vary significantly between consecutive frames and that pixels in one neighborhood move in a similar way. An optical flow implementation from [17] is used in the proposed method. It takes a series of binary images and finds optical flow vectors for all pixels in each frame.

3) Connected Component Analysis: On this step all connected components are extracted and a set of properties are computed for each component. These properties include area, centroid point, bounding box, average velocity and direction of motion obtained from optical flow. BS methods are not ideal and objects may still be disconnected after post processing. Such objects are detected as multiple regions (connected components). To correct this, all regions belonging to one object must be combined into one. In order to determine whether two regions belong to the same object, the following heuristics were proposed. Two regions are combined, if at least two of the following conditions are satisfied:

- The distance between the two centers 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 to the same direction.

Conditions which are used for region combination depend on the specific task and can be selected individually.

4) Fish Identification: The last step of fish detection is fish identification, that is distinguishing fish from other objects, such as garbage, plants and various marine life. To distinguish objects, some prior information about fish is needed. Typically, an approximate fish size can be estimated from videos. This can be utilized to remove all objects which are too small to be a fish.

The size of the fish is measured by finding the width and the height of the fish bounding box. In videos the same fish is typically captured in multiple frames, and therefore, several bounding boxes are obtained. Depending on the frame rate and the time, during which the fish appears in the video, the number N of frames containing the whole fish can be selected. Thus, only these frames participate in size estimation. N biggest values of bounding boxes are selected and the 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 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.

In addition to size, the movement is used for fish identification. Fish counting typically occurs in a fish pass where fish move in one specific direction. Moreover, this direction is typically opposite to the flow direction of water. Therefore, all objects that pass in the opposite direction can be considered as non-fish.

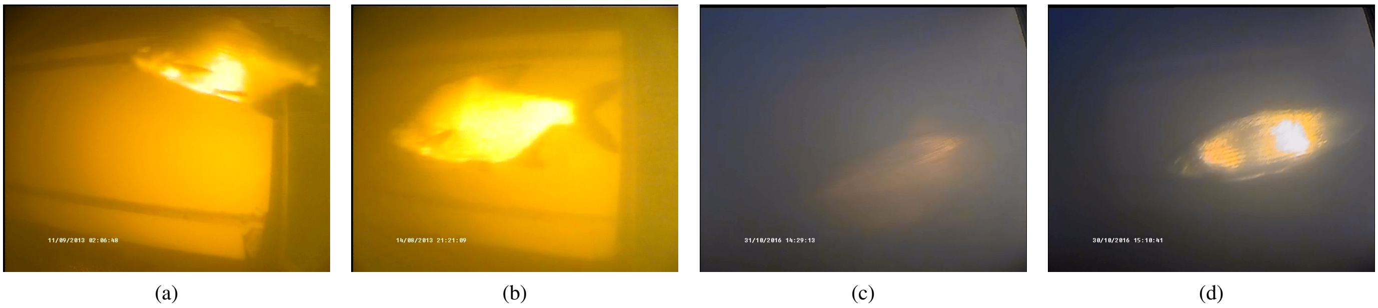
IV. Experiments

A. Datasets

The proposed fish detection system was tested on two challenging datasets. Dataset 1 was collected by ”Kymijoen vesi ja ympäristö ry” organization in 2013. It consists of four videos with different types of natural illumination and various water qualities. Dataset 2 was gathered by the same organization in 2016. It comprises six videos with various levels of visibility. Both datasets consist of video sequences of passing fishes in the real underwater environment. Number of frames in videos varies from 73 to 334 and the number of detections from 14 to 217. Example frames from both datasets can be seen in Fig. 2.

The general scene of the camera position used in video capturing is shown in Fig. 3 [18]. 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.

To evaluate the performance of the implemented methods, the ground truth data were generated by annotating each video frame-by-frame with bounding boxes around fish. The accuracy of the annotation is not perfect. Because of the



(a) (b) (c) (d)

Fig. 2: Example frames: (a)-(b) Dataset 1; (c)-(d) Dataset 2.

low video quality, fish is not easily distinguished from the background, and it is hard to detect edges of the fish even by the human eye. Therefore, a certain permissible error was taken into account during the evaluation.

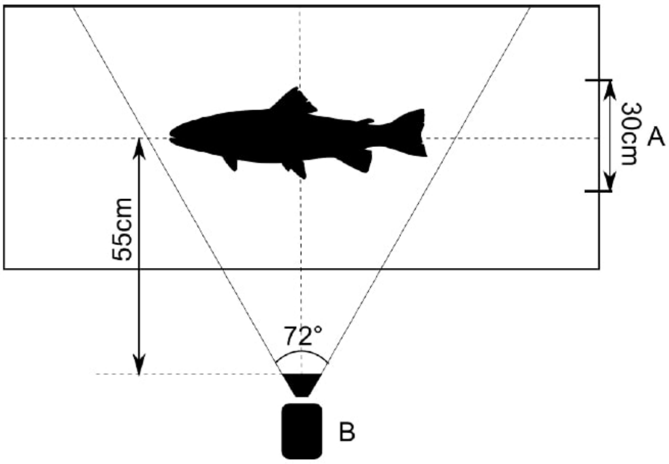


Fig. 3: Scene of the camera position [18].

B. Evaluation Criteria

The fish detection system can be based on either of the three BS methods. To select the most accurate implementation, their detection results were compared. The results of the fish detection were evaluated in accordance with the ground truth. For each video the number of true positive, true negative, false positive and false negative frames were counted. Based on these values 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.

$$P = \frac{TP}{TP + FP} \tag{1}$$

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

$$R = \frac{TP}{TP + FN} \tag{2}$$

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

$$F_1 = 2\frac{P \times R}{P + R} \tag{3}$$

To make decision whether detection should be considered as true positive, the Intersection over Union (IoU) metric was used. It determines how two bounding boxes are similar to each other, and, therefore, is used to calculate the accuracy of the detection. IoU is defined as

$$IoU = \frac{A_i}{A_u}, \tag{4}$$

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. By selecting a threshold value, the detected bounding box can be considered as correct, if its IoU value is higher than the threshold.

C. Implementation

During the experiments three BS methods were evaluated with the both datasets. Parameters used for post-processing for BS were chosen empirically, according to the level of noise and misdetections remained after BS. The implementations of Adaptive GMM and KDE methods were taken from the OpenCV library [19] 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 third method, ViBe, was implemented manually with default parameters specified in [14]. The only parameter that was set individually for each dataset was radius $\mathcal{R}$, since it is responsible for the sensitivity of the algorithm. The $\mathcal{R}$ value was chosen in accordance with how much the fish stands out against the background in videos. The $\mathcal{R}$ values were 15 and 10 for Dataset 1 and Dataset 2 respectively. In the connected component analysis two regions were combined if the distance between the two centres was smaller than a selected threshold value and the height of the combined region was smaller than a threshold. Threshold values were chosen in accordance with the average size of the fish in datasets.

D. Results

Table I 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. A relatively low threshold value is used due to the fact that the accuracy of a ground truth annotation is not perfect. The overall values were obtained by computing P, R, and F_1 values over all frames in the datasets. Examples of the detection results can be seen on Figs. 4 and 5. Fig. 6 shows how the accuracy of the detection depends on the IoU threshold value.

TABLE I: Results of the experiments with threshold for IoU of 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 mediocre 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 remarkably between and within the datasets, the detection results also differ significantly. In general, results for Dataset 1 are better than for Dataset 2. This is because of the fact that videos in Dataset 2 are darker and fish is not fully seen on most of the frames. Also videos from Dataset 2 include situations where the fish stays motionless during a long set of frames and which makes it difficult for BS based methods to detect them.

It can be seen that the 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.

V. Conclusion

In this paper a framework to detect fish in low visibility underwater videos was proposed. The proposed system detects moving objects using the BS approach and distinguish fish from other objects. The system was tested on two challenging datasets and evaluated by the detection accuracy. Considering the obtained results, it can be concluded, that the proposed approach can be applied for the fish detection. However, application of approach should be carefully evaluated before its use. 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. 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 video quality. In cases where the fish is distinguishable from the background, the detection accuracy reaches 72-93%. 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%. The accuracy could be improved by developing better imaging system containing more sensitive camera and better illumination. Future work will include detection a computer vision method for fish species recognition.

Acknowledgment

The authors would like to thank the organization ”Kymijoen vesi ja ympäristö ry” for the provided video materials.

References

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

[2] C. Spampinato, Y.-H. Chen-Burger, G. Nadarajan, and R. 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*, vol. 2, 2008, pp. 514–519.

[3] J. Thorley, D. Eatherley, A. Stephen, I. Simpson, J. MacLean, and A. 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*, vol. 62, no. 4, pp. 808–817, 2005.

[4] H. Balk, “Development of hydroacoustic methods for fish detection in shallow water,” Ph.D. dissertation, Faculty of Mathematics and Natural Science, University of Oslo, 2001.

[5] F. H. Evans, “Detecting fish in underwater video using the em algorithm,” in *Proceedings of the 2003 International Conference on Image Processing (ICIP)*, vol. 3. IEEE, 2003, pp. III–1029.

[6] M. K. Alsmadi, K. B. Omar, S. A. Noah, and I. Almarashdeh, “Fish recognition based on robust features extraction from size and shape measurements using neural network,” *Journal of Computer Science*, vol. 6, no. 10, p. 1088, 2010.

[7] C. Spampinato, D. Giordano, R. Di Salvo, Y.-H. J. Chen-Burger, R. B. Fisher, and G. 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*. ACM, 2010, pp. 45–50.

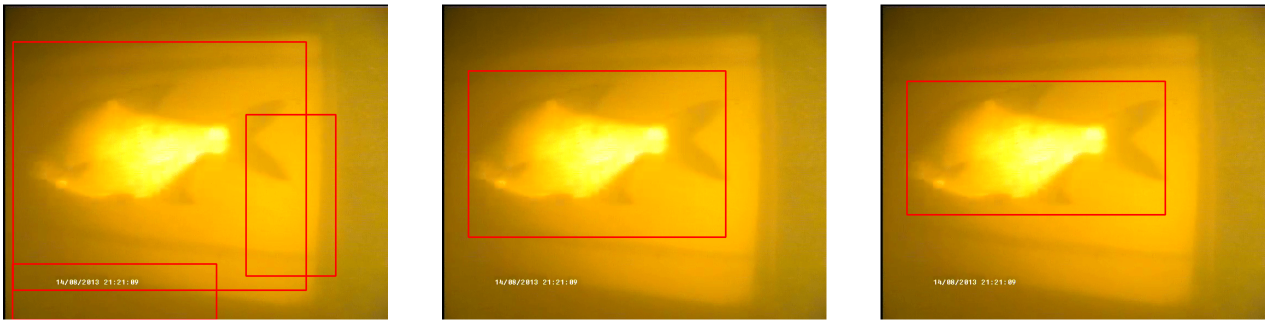
[8] M.-C. Chuang, J.-N. Hwang, and K. Williams, “Supervised and unsupervised feature extraction methods for underwater fish species recognition,” in *Proceedings of the 2014 ICPR Workshop on Computer Vision for Analysis of Underwater Imagery (CVAUI)*. IEEE, 2014, pp. 33–40.

[9] P. Forczmański, A. Nowosielski, and P. Marczeski, “Video stream analysis for fish detection and classification,” in *Soft Computing in Computer and Information Science*, 2015, pp. 157–169.

[10] A. Sobral and A. Vacavant, “A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos,” *Computer Vision and Image Understanding*, vol. 122, pp. 4–21, 2014.

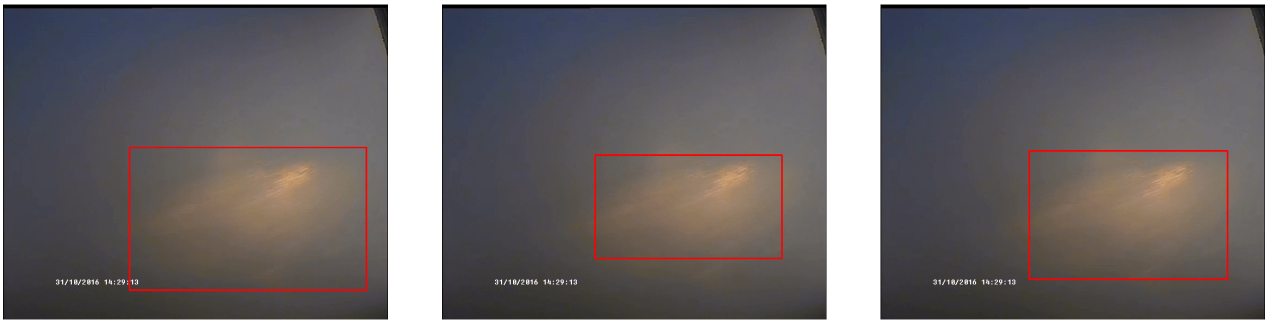
[11] Y. Xu, J. Dong, B. Zhang, and D. Xu, “Background modeling methods in video analysis: A review and comparative evaluation,” *CAAI Transactions on Intelligence Technology*, vol. 1, no. 1, pp. 43–60, 2016.

[12] Z. Zivkovic, “Improved adaptive gaussian mixture model for background subtraction,” in *Proceedings of the 17th 2004 International Conference on Pattern Recognition (ICPR)*, vol. 2. IEEE, 2004, pp. 28–31.



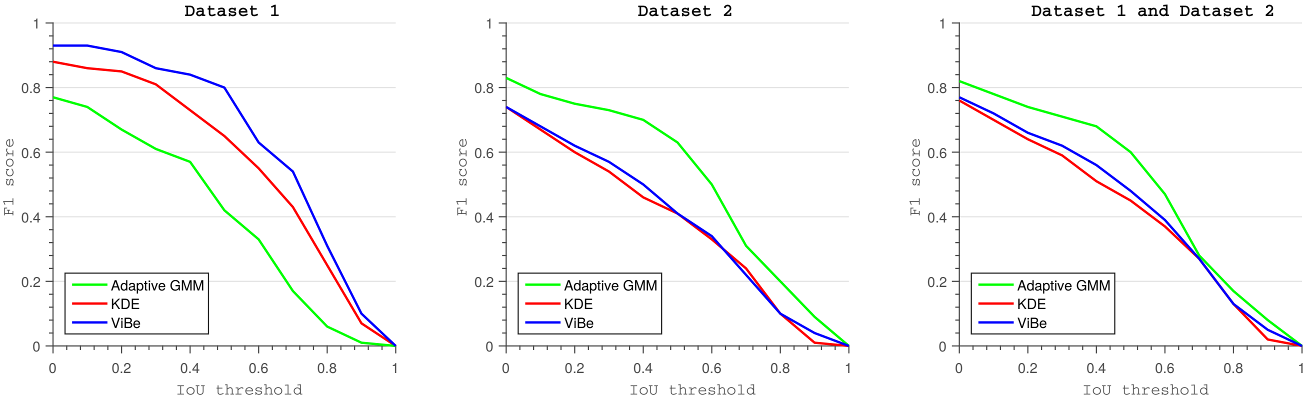
(a) (b) (c)

Fig. 4: The result of detection (Dataset 1): (a) Adaptive GMM; (b) KDE; (c) ViBe.



(a) (b) (c)

Fig. 5: The result of detection (Dataset 2): (a) Adaptive GMM; (b) KDE; (c) ViBe.



(a) (b) (c)

Fig. 6: The dependence of the F1 score on the IoU threshold for overall results.

[13] Z. Zivkovic and F. Van Der Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction," *Pattern recognition letters*, vol. 27, no. 7, pp. 773–780, 2006.

[14] O. Barnich and M. Van Droogenbroeck, "Vibe: A universal background subtraction algorithm for video sequences," *IEEE Transactions on Image processing*, vol. 20, no. 6, pp. 1709–1724, 2011.

[15] I. Setitra and S. Larabi, "Background subtraction algorithms with post-processing: A review," in *Proceedings of the 2014 22nd International Conference on Pattern Recognition (ICPR)*. IEEE, 2014, pp. 2436–2441.

[16] S. S. Beauchemin and J. L. Barron, "The computation of optical flow," *ACM computing surveys (CSUR)*, vol. 27, no. 3, pp. 433–466, 1995.

[17] G. Farnebäck, "Two-frame motion estimation based on polynomial expansion," *Image analysis*, pp. 363–370, 2003.

[18] E. Lantsova, T. Voitiuk, T. Zudilova, and A. Kaarna, "Using low-quality video sequences for fish detection and tracking," in *Proceedings of the 2016 SAI Computing Conference (SAI)*. IEEE, 2016, pp. 426–433.

[19] (2016) OpenCV 3.2.0 Python Tutorials. [Online]. Available: https://docs.opencv.org/3.2.0/d6/d00/tutorial_py_root.html