FISH DETECTION AND CLASSIFICATION IN UNDERWATER VIDEOS

Marina Alonso Poal
malonsopoal@gmail.com

Ponç Palau Puigdevall
ppalaupuigdevall@gmail.com

Escola Tècnica Superior d'Enginyeria de Telecomunicacions de Barcelona (ETSETB)
Universitat Politècnica de Catalunya (UPC)

ABSTRACT

Automatic fish detection and classification in underwater videos is essential for studying and monitoring different species with environmental purposes. However, this is a really challenging task, for underwater environment includes illumination changes as well as drastic movement of aquatic plants. Additionally, fishes' appearance varies in time and we put special emphasis on detecting Serranus Scriba*, a fish species notorious for its excellent camouflaging skills. In this paper we introduce an automatic system including state-of-the-art techniques that aim at tackling those problems. However, as the performance of our system may not be adequate enough for direct fish detection and classification, we conclude that it is to be used for semi-automatic labelling of underwater videos.*

Keywords: fish detection, deep learning, underwater video, background subtraction, semi-automatic labelling

1. INTRODUCTION

Monitoring fish in unconstrained conditions is essential for the study of different species. The collected information helps scientists study their behavior and estimate its number and species among other characteristics. Manual processing of underwater videos is extremely expensive and time consuming. Therefore, there is an increasing interest on automatic processing of underwater videos for fish detection and classification, and especially applying deep learning techniques. Some recent interesting automatic approaches are presented in [1] and [2].

In this paper, we present an automatic system for fish detection and classification that combines both traditional computer vision techniques with recent deep learning approaches. Our method consists of a background subtraction module followed by a fish tracking step and a fish classification module which implements a state-of-the-art detector, the RetinaNet [3].

2. DATASET

The dataset we have used for this study has been provided by *Institut Mediterrani of Estudis Avançats* (IMEDEA) and *Institut de Ciències del Mar (*ICM). This dataset consists of 41 underwater videos of different duration each, recorded in the Mediterranean locations of Cala Egos and Andratx. These locations lie on the coast of Spanish Balearian island of Mallorca.

These high-resolution videos (1920x1080) contain all kinds of underwater conditions addressing complex variability in the scenes. Most of the variabilities are due to complex and dynamic backgrounds, such as extreme aquatic plants movement, blurred captures and luminosity variations. Moreover, the fishes in those videos appear camouflaged in the foreground and their appearance changes as they swim in front of the camera.

Although for this work we handled almost 60 hours of different underwater videos, we only labelled 15 minutes of them, for labelling was not the purpose of our research and it required a specialist's knowledge. This labelling however was strategic. We name keyframe to each frame that appears every 4500 frames, and for each keyframe and its temporal surroundings, a specialist observed and indicated if and where a Serranus Scriba appeared. Having those annotations, we labelled with bounding boxes both Serranus Scriba and other fishes using the Computer Vision Annotation Tool (CVAT). In total, we obtained 24400 frames where 5300 Serranus Scriba and 7700 fishes from other species were spotted.

3. SYSTEM OVERVIEW

The proposed system has three main and differentiated modules. Firstly, the raw video is introduced in the Background Subtraction module, where motion information is used to extract the foreground objects, modelling the background with a Gaussian Mixture Model (GMM) and refining its output by applying morphological filters. The output of this first module are binary masks for

each frame, where a hull represents each fish. In the second module, Fish Tracking, each fish is tracked over frames using a simple centroid tracking approach. Then, their movement is analyzed, and some false fishes are discarded if they do not perform a fish characteristic motion (we assume fishes move more than a determinate number of pixels per frame). The output of this second module are bounding boxes with fish identification for each fish at each frame. Finally, in the third block, Fish Classification, those bounding boxes are inferred at a state-of-the-art object detector, RetinaNet, which classifies the fishes, taking into account its appearance at different frames, into Serranus Scriba, other fishes or simply discards them, and refines the bounding boxes. **Figure 1** summarizes the proposed system architecture.

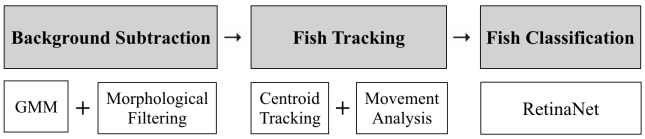

Fig. 1. Proposed system architecture.

3.1. Background Subtraction

Background subtraction is implemented by initially modelling background pixels in the video frames using GMMs [4]. GMM is an effective approach that takes into account motion data, but it does not exploit local information of the pixels, resulting into noisy outputs. To tackle this problem, two strategies are implemented, the first one is to apply gaussian filters before GMM to slightly smoothen the image and the second one is to apply a series of morphological filters after the background subtraction.

3.1.1 Gaussian Mixture Model

GMM models each pixel as an adaptive mixture of Gaussians, and it classifies each pixel based on whether the Gaussian distribution that represents its most effectively is considered part of the background model. In our study, two classes are considered in the GMM, background and foreground (fishes will be later classified), and ideally background should compress everything but the fishes. This background subtraction approach provides good results when the background is quite still and there are no abrupt illumination changes. However, when aquatic plants move a lot or there are drastic illumination changes, this approach misclassifies a large amount of background pixels as foreground.

Our GMM implementation depends on the following four main parameters: number of frames used as memory to adapt the model, number of Gaussian Mixtures for each pixel, background ratio and noise strength (standard deviation of the brightness on each color channel). Those parameters have been fine-tuned to fit the majority of the data, so they are not optimal for each specific video.

3.1.2 Morphological Filters

Morphological filters are operators applied through image filtering to grow or shrink image regions, as well as to remove or fill-in image region boundary pixels. These basic operators process objects in the image based on the characteristics of the selected structuring element. These structuring elements can have different shapes (squared, disk, line) and different sizes, which work as parameters that need to be fine-tuned.

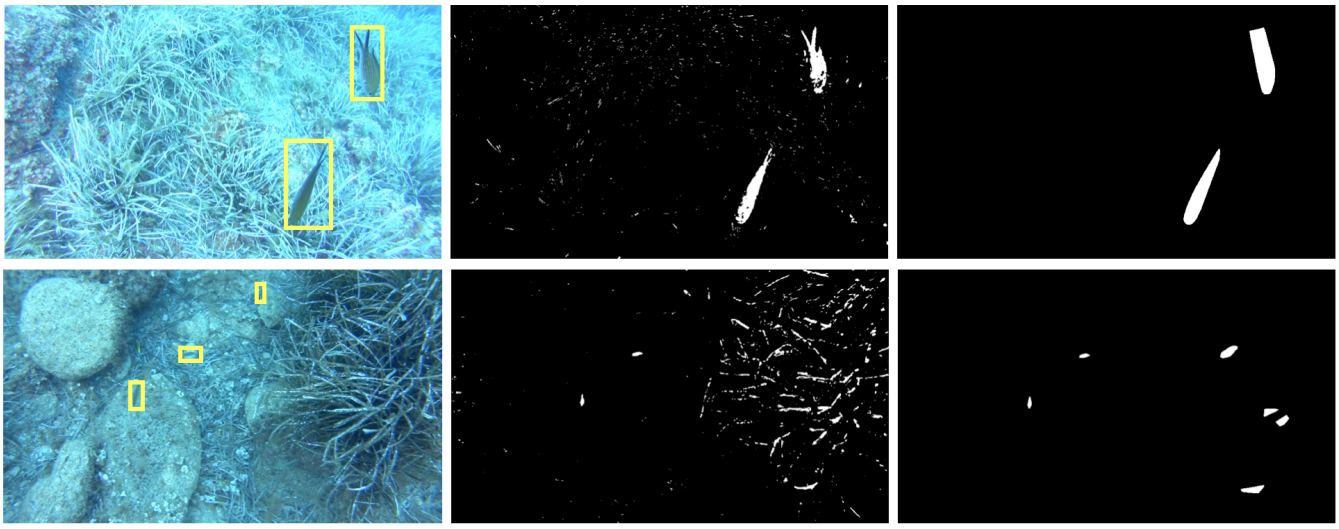

Fig. 2. Example of the Background Subtraction block's performance on different frames. Left to right, original video with ground truth, GMM and morphological filters. The first example shows how the algorithm has successfully detected both fishes, while in the second example, where aquatic plants were moving a lot, the system failed to fit them in the background model. Hence, it wrongly detected some of the aquatic plants parts as fishes. Additionally, as the morphological filter parameters are quite restrictive to remove all those false positives, sometimes they also delete true positives. The second example shows that morphological operators missed one fish at the top center of the image.

In our research, we applied the following concatenation of morphological filters to exploit the local information of the extracted foreground. First, a reconstruction filter with an opening is applied to remove unconnected pixels, followed by a closing with a small structuring element to connect the remaining components. The same process is repeated again but with larger structuring elements to get rid of little objects and strengthen the residual ones. Finally, the minimum hull of the remaining objects is computed to simplify its appearance. **Figure 2** shows one positive and negative example of the outputs of the first module compared to the ground truth.

3.1.3 Optical Flow

GMMs needs some training frames to model the background. Ideally, those frames should not contain any foreground object, but in our automatic system, these training frames are chosen as the determined number of the initial frames, which may contain some fish instances. Additionally, in some cases, GMM fails to model correctly the background, for if the variance that models every gaussian is too large, it will fail to detect any foreground.

Therefore, to compensate those disadvantages in the background subtraction module, we also introduce in our system an optical flow estimation. Optical flow [4] is a 2D motion vector in the video footage caused by the 3D motion of the displayed objects. There are various methods to estimate optical flow, but for this project we opted for implementing a state-of-the-art deep learning approach, the FlowNet2.0 [5]. FlowNet2.0 is an end-to-end architecture, stacked by FlowNetCorr and FlowNetS. FlowNetS simply stacks two sequentially adjacent images as input, while in FlowNetCorr, two images are convoluted separately, and are combined by a correlation layer. In a spatial pyramid network, the authors trained one deep network for each level independently to compute the flow update, estimating motions in a coarse-to-fine manner.

However, although we implemented FlowNet2.0 and successfully managed to inference the videos from our dataset, we haven't yet integrated the optical flow in our whole system.

3.2. Fish Tracking

Fish tracking module involves firstly tracking the fish with a simple centroid tracking and then analyzing each fish's trajectory to reject objects that do not have a fish-like movement. In the future, a deep learning approach for tracking based on the Re^3: Real Time Recurrent Regression [6], which is currently being implemented by another researcher at our project group, will be integrated on our proposed system.

3.2.1 Centroid Tracking

We perform tracking by computing the four simple steps based on, as its name suggests, the distances between centroids. First, we accept the bounding box coordinates corresponding to each hull from the previous module and compute its centroid. Followingly, we compute the Euclidian distance between the centroids at the current frame and all of the centroids of the already identified objects from previous frames. The tracked object centroid is updated to their new centroid locations based on the new centroid with the smallest Euclidean distance and finally, if we cannot match an object to any of the existing objects for a determinate number of subsequent frames, we will deregister that object. Equivalently, if there are more input detections than existing objects being tracked, we register the new object.

This tracking approach is quite simple and has the following main downsides. It requires object detection for each frame of the video, which can be computationally expensive. This also means that it does not try to detect by tracking, so it does not correct bad detections. Additionally, it does not handle overlapping objects well and because of the Euclidian distance, it can cause some centroids to swap IDs, which is not desired. Despite these three downsides, it still gives provides good results.

3.2.2 Movement Analysis

The algorithm we apply to analyze the movement is quite straightforward. While studying our dataset, we realized that the majority of the fishes moved quite fast throughout the frame, which distinguished their movement from that of the aquatic plants'.

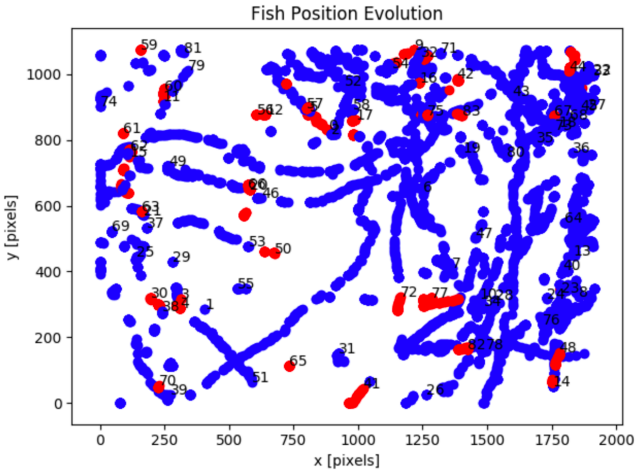

Fig. 3. Representation of the trajectory of the detected objects in a certain time slot. Red dots represent the position of the rejected objects at each frame.

Therefore, the algorithm consists of computing the total pixel displacement of the object from the moment it appears until it disappears and dividing it by the number of frames in which that object appears. This can be seen as a kind of mean displacement per frame. If that value is not larger than a threshold, the object is discarded. This threshold is a parameter that needs to be fine-tuned, for the system is sensitive to this parameter. At this moment, this threshold is set to 5 pixels/frame. **Figure 3** represents the trajectory of the detected objects in a certain time slot.

3.3. Fish Classification

For this project, we used RetinaNet as an object detector, because at this moment it is the network providing best results on COCO dataset and although it is not the fastest network, it has quite a fast performance. Moreover, real-time performance was discarded for this project, as it was not necessary.

RetinaNet was first introduced in 2017 by Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollár, P., which developed it at Facebook Artificial Intelligence Research (FAIR) under the Detectron paradigm. While looking for the reason why one-stage detectors could not beat two-stage detectors in accuracy, they discovered that there was an extreme foreground-background class imbalance problem, that only two-stage detectors were managing. However, this led to a higher computationally complexity and therefore a higher inference time. The authors of RetinaNet had the intuition that solving this imbalance in a one-stage detector would enable it to perform at least as good as a two-stage detector.

The foreground/background imbalance problem is tackled in RetinaNet by introducing a new loss, named Focal Loss, which mainly consists on applying a modulating term to the traditional cross entropy loss to focus learning on d hard negative examples. Therefore, all the easy examples the network is certain about contribute less to the loss so that the learning can be focused on the few challenging cases. The focal loss is defined as:

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{otherwise,} \end{cases}$$

$$\mathrm{FL}(p_t) = -\alpha_t (1 - p_t)^{\gamma} \log(p_t).$$

The focal loss introduces two new parameters with respect to the traditional cross entropy loss. The most significant one is γ, which forces the network to focus on hard examples. The second parameter is α_t, which is the offset class imbalance of number of examples. Note that if γ=0 and α_t=1, the focal loss turns into a simple cross-entropy loss. **Figure 4** shows the focal loss for different γ values compared to the classic cross entropy loss.

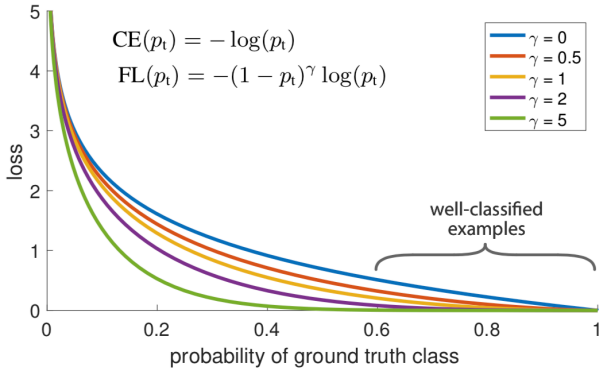

Fig. 4. Focal loss for different γ values compared to the classic cross entropy loss. It introduces two new parameters with respect to the traditional cross entropy loss. The most significant one is γ, that forces the network to focus on hard examples. The second parameter is α_t, which is the offset class imbalance of number of examples.

Mainly, RetinaNet is a network composed of a backbone network (Feature Pyramid Network) built on top of ResNet [7], responsible for extracting the feature maps of an entire image, and two subnetworks: one responsible for object classification and the other responsible for the bounding box regression. In this section we will briefly comment the four main components of the RetinaNet network. **Figure 5** provides a visualization of the complete architecture.

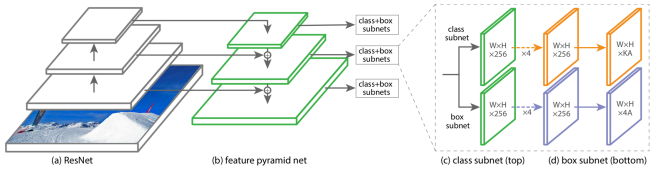

Fig. 5. RetinaNet architecture. RetinaNet is a network composed of a backbone network (Feature Pyramid Network) built on top of ResNet, and two subnetworks: object classification and bounding box regression.

RetinaNet as Object Detector

Firstly, we proposed RetinaNet to perform fish detection on single images. In this section, we summarize the achieved results following this approach. Pretrained weights of RetinaNet-50 were available from [8], so we checked how this pretrained model performed on our data. As the results were not promising, we concluded that the pretrained model should be trained. Since the number of data examples was rather low, our approach is based on applying transfer learning in Classification and Regression modules of RetinaNet, leaving the Backbone network (Resnet50) and FPN frozen. Several models were trained through the course of this work, but only the most significant ones are summarized in this paper: model 1 and model 2. The difference between these models is the layers that have been retrained as well as the amount of data used to train each model.

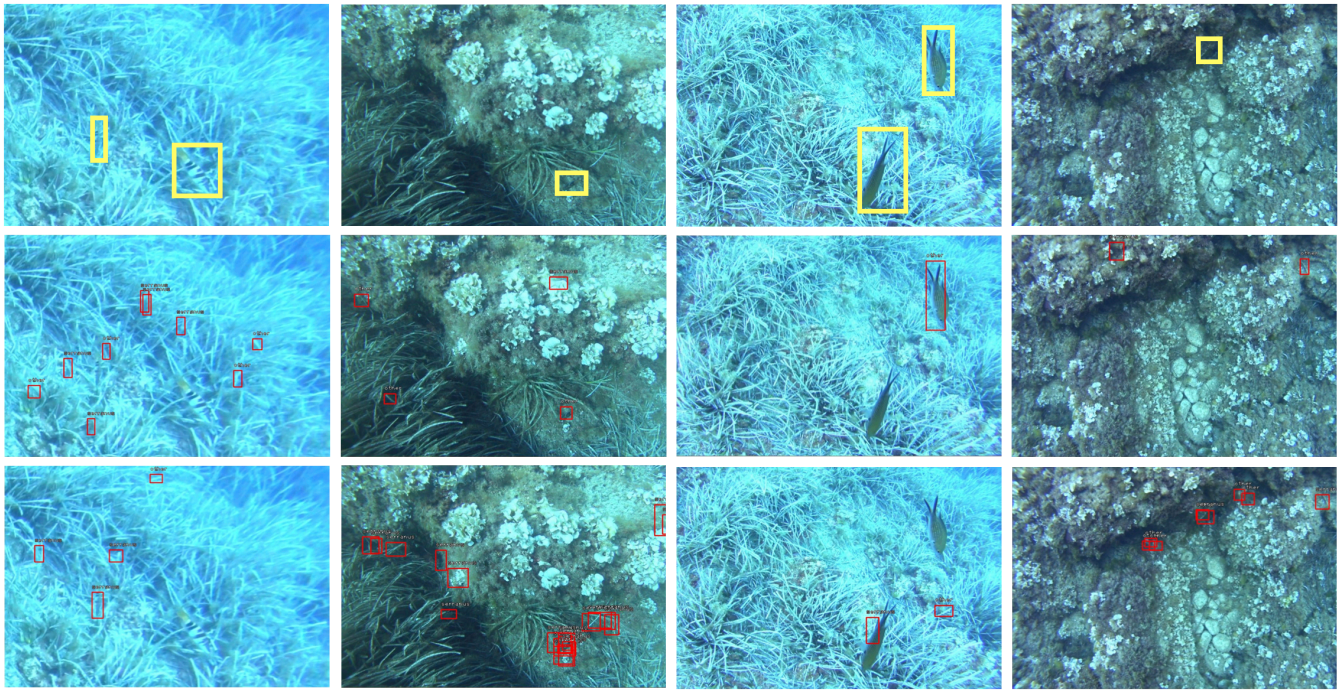

Fig. 6. Example of the RetinaNet's performance on different videos. Example of RetinaNet performance on validation images. Column 1 and 3 belong to Andratx videos, columns 2 and 4 belong to Cala Egos videos (validation). First row shows ground truth images, second row shows results of model 1 and third row shows results of model 2. Analyzing these results, it is clear that model 2 has less precision than model 1. Contrarily, model 5 has higher recall, it detects serranus in the first image while model 1 does not detect it, the same phenomenon can be appreciated in the last column. Having said that, model 2 is more appropriate for the problem we are trying to solve because in a situation where a Serranus is static it may detect it (while model 2 will not). Moreover, the high number of false positives can be reduced with further processing based on motion.

A) Model 1

This model is the result of training entirely the Classification and Regression modules of RetinaNet, which are FCN (Fully Convolutional Networks), but with very few labels (only CalaEgos8L and CalaEgos9L). These videos provide 2256 positive instances (Serranus Scriba). Out of this amount, 1818 examples are used for training and 438 for validation.

B) Model 2

This model is the result of training the 2 last convolutional layers of the Classification module and the last convolutional layer of the Regression module. The number of samples to train this model is considerably higher than for model 1. In the data used to train this model, there are 5300 positive samples, from which 4000 are used for training and 1300 for validation. Notice that in this model fewer layers are trained. This decision is based on the fact that if all the regression module is retrained, the effectiveness of the Regression module trained on COCO is lost. The same reasoning applies to the Classification module. For this model, we provide both qualitative and quantitative results.

Figure 6 exemplifies the result of model 1 on frames from different videos and compares its performance with the one from model 2. **Figure 7** shows the precision and recall curve and the F score evolution during epochs for the second model. Despite the obtained results, we conclude that RetinaNet is not able to solve the detection task with acceptable metrics. This is due to the fact that we are facing of a very challenging task, detecting fishes from single images that camouflage in the background, even for humans.

Finally, due to the above stated reasons in this section, we relied on a detection based on motion, while we rely on a CNN (ResNet) to perform fish classification, without regression.

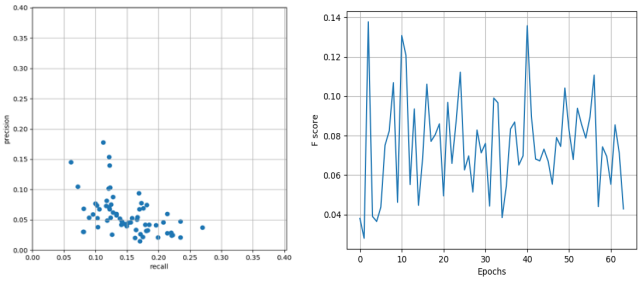

Fig. 7. Model 2 training results. On the left, precision and recall scatter plot for different epochs. On the right, temporal evolution of F score metric as a function of the epochs. Considering these results, model weights for epochs 22 (where recall is maximum) and 40 (where F score is maximum) have been evaluated.

4. RESULTS

Results of each block have been discussed in its correspondent section. In this part of the document, we expose the overall system results, that is to say, concatenating the background subtraction and fish tracking module with RetinaNet. This approach took each hull resulting from the BS module and made a forward pass with RetinaNet. As it was expected, this approach did not work properly because RetinaNet is trained with full-resolution images instead of just crops of images around fishes (the output of the background subtraction module). Thus, the following two options are considered. One option would be training RetinaNet with cropped images around the bounding boxes. This could refine the bounding boxes proposed by the BS module. Another option could be a CNN such as Resnet to perform image classification given the hulls that the BS module outputs. This option would rely on the BS module to perform the detection task.

As this project is not over yet and the qualitative results provided by the integrated system at this moment are quite poor, no quantitative results have been computed so they are not shown in this paper.

5. FUTURE WORK

The authors dedicated a limited amount of time to this project corresponding to 5 ECTS for each author. Thus, there is still many work ahead, for results could end up being far better than they are currently.

First of all, a module before the background subtraction should be introduced, responsible for camera motion compensation, as it is sometimes unstable. The background subtraction module would improve its results if the FlowNet2.0 estimation of the optical flow was integrated in the system. Additionally, it would be interesting to try to apply some state-of-the-art deep learning-based background subtractors such as the BScGAN [9], a conditional Generative Adversarial Network. In the tracking module, instead of the simple centroid tracking approach presented on this paper, the Re^3 tracking implementation that is currently being performed by another group member should be integrated on our system to enhance and correct both tracking and therefore, detection. The performance of the classifier would improve drastically if it was trained with more data, so more videos should be carefully labelled. Finally, quantitative results of the integrated system such as accuracy, recall, mean Average Precision (mAP) or Intersection over Union (IoU) should be computed, to be able to fairly compare different approaches, as well as fine-tune some hyperparameters.

7. CONCLUSIONS

In this paper, we have presented an automatic fish detection and classification system for underwater videos. This system consists of three main blocks: background subtraction, fish tracking and fish classification, and it combines traditional computer vision techniques such as GMMs with state-of-the-art approaches like the RetinaNet detector. Due to the hard environment conditions such as drastic movement of aquatic plants, illumination changes and fish camouflage, as well as the fact that the project is not over yet, the system's performance is pretty mediocre. For this reason, we do not consider that it could work as a stand-alone automatic fish detector and classifier. Instead, our system would be extremely useful as a semi-automatic labeling tool, because, as it has a qualitative high recall, it notices some fishes that even humans struggle to detect. That being said, there is still a large amount of work ahead.

ACKNOWLEDGEMENTS

This paper has emanated from the Master in Telecommunication Engineering (MET) and Master in Advanced Telecommunication Technologies (MATT) subject *Introduction to Research*, supervised by Prof. Ramon Morros and Prof. Elisa Sayrol, whose supervision is acknowledged by the authors. The authors also acknowledge the work of Benjamin Dorra, a French graduate student who made important contributions to the project and institutions IMEDEA and ICM for providing the dataset.

REFERENCES

[1] Salman, A., Siddiqui, S. A., Shafait, F., Mian, A., Shortis, M. R., Khurshid, K., ... & Schwanecke, U. (2019). Automatic fish detection in underwater videos by a deep neural network-based hybrid motion learning system. ICES Journal of Marine Science.

[2] Moniruzzaman, M., Islam, S. M. S., Bennamoun, M., & Lavery, P. (2017, September). Deep learning on underwater marine object detection: a survey. In International Conference on Advanced Concepts for Intelligent Vision Systems (pp. 150-160). Springer, Cham.

[4] Stauffer, C., & Grimson, W. E. L. (1999). Adaptive background mixture models for real-time tracking. In Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149) (Vol. 2, pp. 246-252). IEEE.

[5] Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., & Brox, T. (2017). Flownet 2.0: Evolution of optical flow estimation with deep networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2462-2470).

[6] Gordon, D., Farhadi, A., & Fox, D. (2018). Re^3: Real-Time Recurrent Regression Networks for Visual Tracking of Generic Objects. IEEE Robotics and Automation Letters, 3(2), 788-795.

[7] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[8] https://github.com/yhenon/pytorch-retinanet

[9] Bakkay, M. C., Rashwan, H. A., Salmane, H., Khoudour, L., Puigtt, D., & Ruichek, Y. (2018, October). BSCGAN: deep background subtraction with conditional generative adversarial networks. In 2018 25th IEEE International Conference on Image Processing (ICIP) (pp. 4018-4022). IEEE.