A NOVEL METHOD FOR AUTOMATIC FISH COUNTING BASED ON MACHINE LEARNING USING MORPHOLOGICAL FEATURES

A Thesis

by

Ahad Aghapour

Submitted to the
Graduate School of Sciences and Engineering
In Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in the
Department of Computer Science

Özyeğin University
August 2020

A NOVEL METHOD FOR AUTOMATIC FISH COUNTING BASED ON MACHINE LEARNING USING MORPHOLOGICAL FEATURES

Approved by:

Prof. H. Fatih Uğurdağ, Advisor
Department of Electrical and Electronics Engineering
Özyeğin University

Prof. A. Tanju Erdem
Department of Electrical and Electronics Engineering
Özyeğin University

Asst. Prof. Tacha Serif
Department of Computer Engineering
Yeditepe University

Date Approved: 20 August 2020

To my beloved advisor, family, friends, and teachers

ABSTRACT

This thesis offers a machine learning based computer vision approach for counting fish in a non-stop running narrow water stream (or equally on a conveyor belt or similar mechanism) in a fish farm. We call this “continuous operation”. Such setup requires that counting is done extremely fast, which we call “real-time” operation. To our knowledge, our fish counting solution is the only such method, namely, continuous and real-time. The hardest subproblem here is correctly counting “overlapped fish” and yet be flexible in terms of fish sizes. Most of the previous works are not real-time due to their inefficient approach to counting overlapped fish. Our superiority in speed mostly stems from selecting fewer and computationally less expensive morphological features (which we feed to machine learning). Although the state-of-the-art is pretty accurate (98.9%), we were able to improve it to 99.4% for even very difficult test cases. Since our solution is continuous unlike previous work in the literature, we had to also solve the problem of “merging fish” (overlapped or not) lying at the boundary of consecutive frames.

ÖZETÇE

Bu tez, bir balık çiftliğinde durmaksızın akan dar bir su akışında (veya aynı şekilde bir taşıma bandı veya benzer bir mekanizmada) balıkları saymak için makine öğrenimi tabanlı bir yapay görme yaklaşımı sunmaktadır. Bu tür işlemlere “sürekli işlem” denmektedir. Böyle bir kurulum, sayma işleminin son derece hızlı yapılmasını gerektirir ve buna "gerçek zamanlı" işlem denir. Teze konu olan balık sayma çözümümüz de bu tür sürekli ve gerçek zamanlı bir yöntemdir. Buradaki en zor alt problem “üst üste binen balıkları” doğru bir şekilde saymak ve her bir balığın boyutları konusundaki esnekliktir. Önceki çalışmaların çoğu, üst üste binen balıkları sayma konusundaki verimli olmayan yaklaşımları nedeniyle gerçek zamanlı olmamıştır. Hızdaki üstünlüğümüz, daha az sayıda ve hesaplama açısından daha ucuz morfolojik özelliklerin (makine öğrenimine beslediğimiz) seçilmesinden kaynaklanmaktadır. Günümüzün gelişmiş teknolojisi oldukça doğru sonuçlar vermesine rağmen (%98,9), çok zor test koşullarında bile bu oranı %99,4'e çıkardık. Çözümümüz literatürdeki önceki çalışmalardan farklı olarak süreklilik arz ettiği için, ardışık görüntülerde yer alan “balıkların birleştirilmesi” (üst üste binmiş olsun ya da olmasın) sorununu da çözmek zorunda kaldık.

ACKNOWLEDGMENTS

I want to thank my supervisor, Prof. H. Fatih Uğurdağ for providing guidance and feedback throughout this work and also in my life. Without him, I would have accomplished much less. I also want to thank my wife and my parents for being patient throughout various stages of life and especially during this work. I cannot also forget to thank my dear friend Hamed for always encouraging me, providing guidance, and being a sounding board when required. In addition, I want to thank my managers Ali Baran and Rıfat Demircioğlu during my internship at their company, Yongatek, where the problem statement of this thesis emerged. Finally, I would like to express my sincere gratitude to Özyeğin University's Faculty of Engineering for the full scholarship they offered throughout my master's studies.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

One of the main concerns of the highest importance in modern aquaculture is counting the aquatic creatures. The reason for the significance of this area in the hatcheries is that knowledge of the number of fish enables managing the fish farms through short-term and long-term planning. This involves a vast range of necessary knowledge from simple issues such as the amount of food required for the existing fish in the farm in specific time slots to deciding about the long-term goals like increasing the number of fish, signing new contracts of providing fish as a vendor, Etc. Some studies suggest that the farm fish have to be counted at least four times in their life span: two times by the hatchery (seller) and the farmer (buyer), at least one time by the farmer to assess the survival rate of the fish of each size in each pole or cage and one time before delivering the fish to the market [1].

The issue of counting fish has not always been as easy as today. Before designing machines for counting fish, they had to anesthetize the water creatures by giving them drugs, and then they put them in buckets and weighted them. Then, by dividing the total weight over mean weight, they could estimate the number of fish. To have the mean weight, they had to weigh them and reach an acceptable average weight randomly. In addition to taking a long time (taking up to several days or even weeks) and effort, this process was not suitable for the fish as it put stress on them, resulting in them losing their appetite. This could cause a growth decline and even death. Losing a fish means that the specific fish is lost as a number in the production rate and that all the work and food

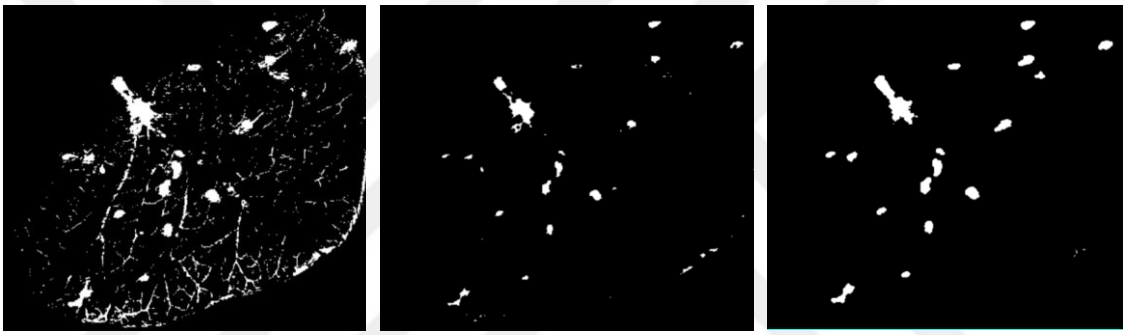
consumed for that fish is wasted. In addition to the mentioned disadvantages, this method is also not as accurate as it is needed in today's aquaculture business. The accuracy in this method may even be lower than ±10, which is not even close to the acceptable rate[1]. The importance of fish counting is not restricted to only fish farming. However, it may be extended to areas like protecting the sea wildlife and monitoring endangered aquatic species in lakes, rivers, seas, and probably oceans.

Of course, when it comes to counting, the importance of accuracy is more than any other factor. There have been many efforts to accomplish the goal of accurate fish counting throughout the past years, some of which have involved proposing novel methods based on interdisciplinary research. There has been a significant improvement in this field over the past four decades, resulting in a massive increase in the productivity of fish farms, thus providing human society with more seafood resources. Leading works in this area involve using knowledge of electrical engineering [1][2][3], optical engineering [4], acoustic and sonar engineering [5][6][7] and computer science [8][9][10].

The applications of methods of counting are not restricted to fish counting only. There are some notable works in counting, including different applications in diverse fields, from farming to traffic control. As the use of computer science in the area is becoming undeniably important, here we categorize the issue of counting to two types of counting through computer science methods (mainly machine learning) and through other methods.

Some notable works in the field of counting have used computer science methods other than machine learning techniques. These works may include attempts to count things other than fish [8][9][10][11][12].

In [8], computer vision, image processing, and pattern recognition is used to count soybean leaf aphids through by taking advantage of images with an HSI color system. In their work, after getting the RGB photo from the camera, the software part is begun. The RGB photo is changed to the HIS system photo. The resulted image is then segmented via using a specific threshold. Then, the image's veins are removed through morphological erosion, and leaf edges are then removed. The final step is to reduce the segments knowing each leaf has only one petiole. This is shown in Figure 1.



Gray image after segmentation in HSI | Image after corrosion | Image after removal of leaf edge

Figure 1: The segmentation method used in [8]

[9] proposes counting people in crowds by using images sensors and segmenting foreground objects from the background objects and introduces an algorithm that calculates bounds on the number of people in each region of the projection. They have used an existing prototype including 8 image sensor nodes which are all connected to a central node which enables them to count people in a real-time fashion.

[10] provides a similar method implemented in Raspberry Pi microcomputer to detect cars in lanes using remote video cameras through an automatic counting

algorithm. Using image processing techniques and statistical analysis has been explored in this work. They subtract the background from the acquired video and then take advantage of machine learning techniques to do the counting part. They have also compared intrusive and non-intrusive methods for counting the cars. They have mentioned Pneumatic tubes, an intrusive method that enables the users to detect the type of cars, their speed, and of course, count them. However, there are some disadvantages to this method, some of which are the sensitivity of the pneumatic tubes to temperature and also their easily damage-likelihood. As a result of these disadvantages, they have proposed video vehicle counting as a non-intrusive method. They use video frames to extract vehicle images, identify the lanes via background subtraction, and detect the gradient edges. As a result of some imperfections in the foreground (e.g., existing unnecessary holes in the common background), they used machine learning techniques. The machine learning method which they use is supervised machine learning. They use it to train the model with the dataset, and they have defined two classes of "existing vehicle in the frame" and "no vehicle in the frame". The simplifications have enabled them to use even cheap micro-computers to implement the method to do the low computational work required for this job. As a result, they have their four steps of getting the image, analyzing the image, detecting the objects through supervised machine learning and pattern recognition, and finally reporting the count. Obviously, as a result of the mentioned classification policy, they have 0 and 1's as the outputs of the classifier function. Then combining the counts of binary strings in all the lanes gives the total number of the cars. An example of the resulted string is shown in Figure 2.

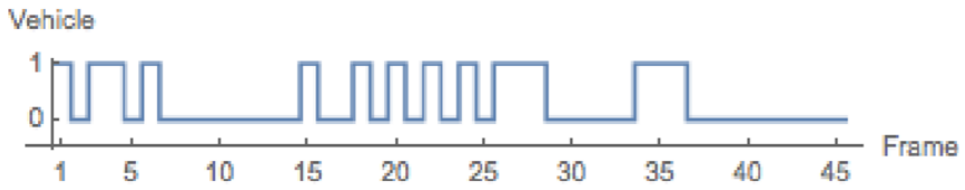


Figure 2: Binary vehicle counting used in [10]

[11] uses image processing algorithms and statistical analysis to detect and count larvae from a microscopic image. Their method enables them to count the larvae as small as 30-300 micrometers. The images are captured with a firewire camera that is installed on a microscope, and the images are captured then. Their used system is shown in Figure 3.

Figure 3: Implemented system in [11]

Through three Laplacian filters and five layers of the median filter, the image is then enhanced. The result of the enhanced image of their work is brought in Figure 4. After applying adaptive thresholding and removing the background, the resulted image is brought in Figure 5.

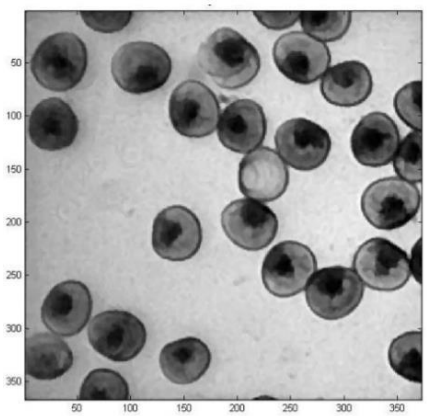


Figure 4: The enhanced image in [11]

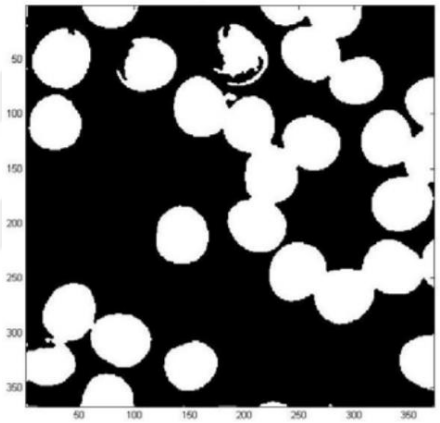


Figure 5: Segmented image in [11]

Two issues addressed in their work are 1- The larvae which are connected to each other in one segment and 2- the larvae on the edge of the image. Both problems are solved through labeling techniques in [11]. These two issues are shown in Figure 6.

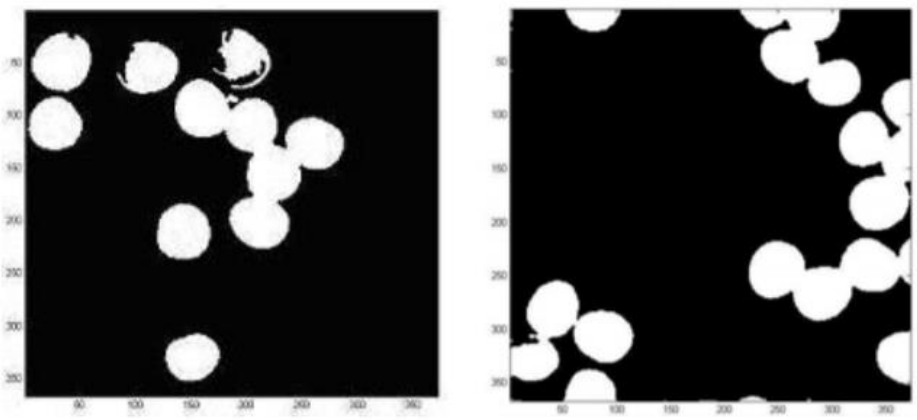


Figure 6: The two problematic images addressed in [11]

According to the mentioned problems, each segment is labeled as containing either one or multiple larvae. After that, each object's area is estimated via pixel counting to estimate the number of larvae in them. In their example, they have labeled twenty images. They have mapped them into a statistical histogram. This histogram contains 100 values for which each interval's width represents one-hundredth of the maximum area. The main frequency of such a histogram is determined by the fast Fourier transform (FFT) shown in Figure 7.

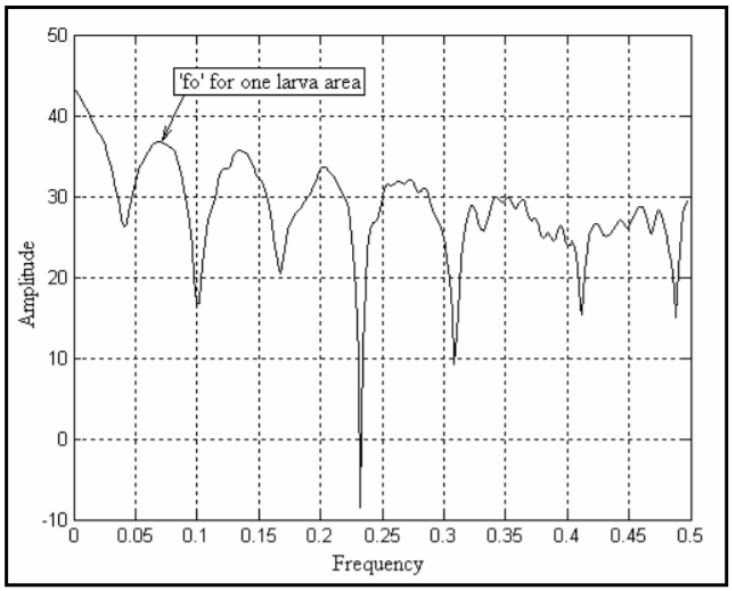


Figure 7: FFT of the histogram of objects in [11]

The performance of different texture-based used with support vector machine is investigated in [12] to find a proper algorithm for distinguishing bicycles and pedestrians and therefore, to estimate the traffic of bicycles. As in some other works discussed above, they convert images from RGB to HSV. The visionary properties, such as saturation, are obtained and quantized, forming a 100 bin histogram of two-by-two values of saturation and hue. As they aim to count people and bicycles, they consider a test of 200 images (100 for each) for training, but for forming the vocabulary features 162 bike images and 140 human images are used. This part of the work is done through using SIFT features, Pyramidal HOG features, and via different levels if scales using SVM algorithms. After that, the training is done, and the results would be either 1 (existing bike) or 0 (no bike). A schematic of the work done in [12] is shown in Figure 8. It worth mentioning that their method can increase accuracy by up to 89%.

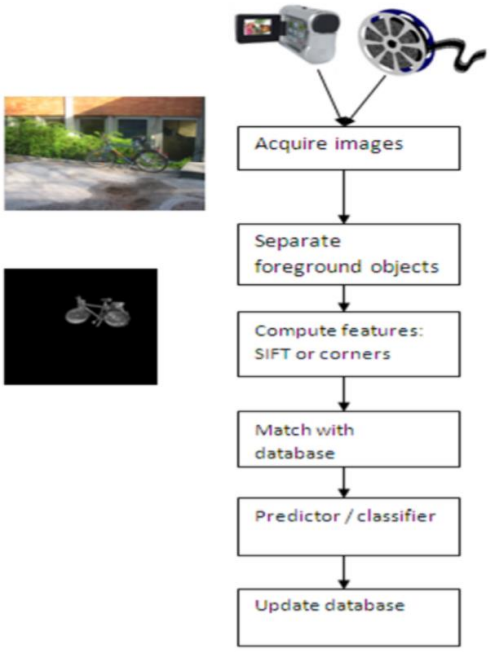


Figure 8: Schematic of the work done in [12]

Some works include counting fish with methods other than machine learning [13][14][15][23]. In [13], the methodology is based on applying a Bayesian filtering technique that enables tracking of objects whose number may vary over time. The overview of their work is presented in Figure 9. The results of such work enable marine biologists to investigate the effectiveness of transposition mechanisms.

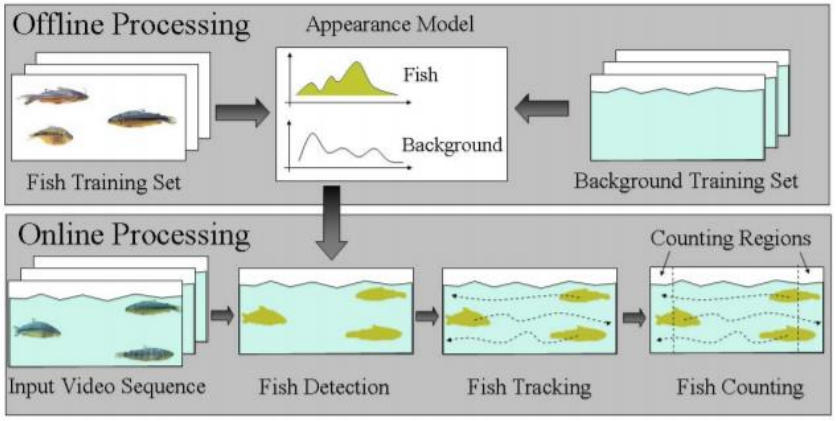


Figure 9: Fish counting method of [13]

The work in [13] is mainly discussing a proposed solution to the problem of object tracking in counting. The reason for object tracking to be one of their main concerns is that they tend to count the number of fish in the water stream as they are freely moving around and changing their positions. They have to detect and localize the object in the captured videos recursively. Some issues to be considered while designing a system with the main concerns mentioned in [13] are: arbitrary number of fish (as the fish may enter and exit the concerned area), arbitrary fish movements and behaviors (as the fish may cover each other in the images), arbitrary size and orientation of the fish (as the fish may be of any size and shape and direction in the images), 3D motions (rather than people moving on the streets, fish can move along the three axis), environmental changes causing the images quality to vary, etc. Their method does not require specific hardware for the fish to be counted, although it requires high-quality cameras and light sources. The model used in their work consists of linking image pixels to foregrounds or backgrounds for describing the scene's appearance and modeling it. They use a likelihood function for detecting the fish, which is used to model the probability of a hypothesis in an observed image. They also have tested their results to show that their used algorithm is suitable for accurate fish counting. They also claim that their method can be used to gather information on some features of the fish such their ability to swim, their swimming time and speed and their peak flow traffic, resulting in being able to foresee fish trajectories, in different environments and conditions. However, their highest achieved accuracy is 81%, which may not be adequate for reliable business tasks and programming.

The method proposed in [14] includes using image processing techniques to implement an embedded system in Raspberry Pi hardware for counting fish. Although their work may introduce some theoretical novelty, the technique requires an enclosed environment to offer accuracy and cannot be used for real-time counting. They propose

two methods to provide the counting strategy with the mentioned requirement: one is when a camera is placed above a channel connecting two ponds. The other consists of placing multiple cameras to cover an entire area in a matrix fashion. It worth mentioning that the work in [14] takes advantage of linear classification, and like other works, they consider using thresholds for blobbing the images. They only consider two classes: fish and no-fish (i.e. image containing the aquarium background). The overview of their proposed system is presented in Figure 10.

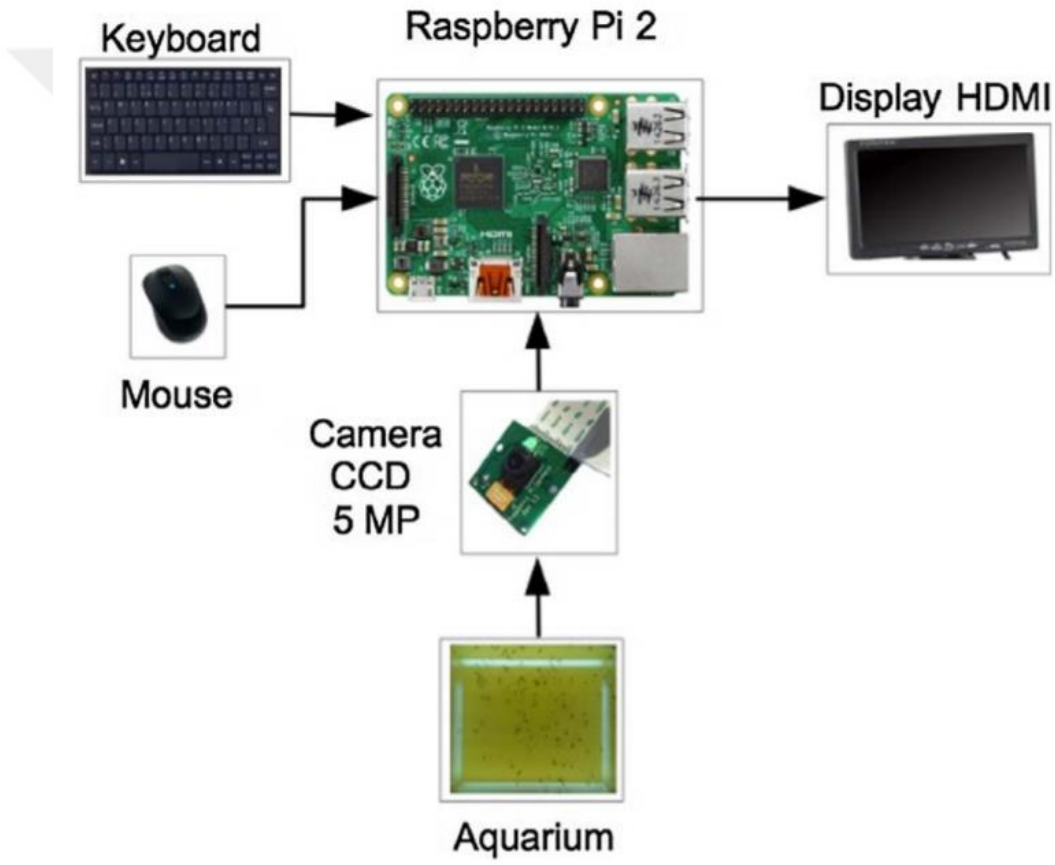


Figure 10: Overview of the work in [14]

The variables affecting the quality of the image acquisition for the sake of fish counting in [14] are the size and type of fish species, the mobility models of the fish and also physical characteristics such as water level, lighting flux, area of water and the last

but not the least: camera image resolution. In this work, digital image processing is used to aid the counting process. They use different images to add accuracy to the counting process, causing some delay in the work. The images should be altered due to the aquarium's vibrations, movements of the water and fish, and turbidity of the water and image reflections on the aquarium walls. To overcome the issues related to these features, a filtering process has been used. The whole process is done recursively to add up the number of fish and give the total number at the end. According to the above, this method does not provide a real real-time solution to the fish counting problem. They have also compared their results with the results of the LS-SVM method. Their highest accuracy rate is 95.57%. Some of their test results are brought in Table 1.

Table 1: Some test results in [14]

Real amount of fish in the aquarium	Fish counted by the proposed system	System Accuracy (%)	Fish counted SVM (Fan and Liu, 2013)	LS-SVM Accuracy (%) (Fan and Liu, 2013)
350	337	96.29	340	97.14
300	287	95.67	291	97.00
250	237	94.80	240	96.00
200	192	96.00	194	97.00
150	139	92.67	142	94.67
100	98	98.00	98	98.00
Average accuracy		95.57		96.64

The use of computer vision methods for analyzing underwater videos has been explored in [16], based on two different algorithms: matching blob shape features and histogram matching. In their work, they have included a video processing system with three subsystems for analyzing the video textures, detecting the fish, and finally tracking

the modules. For the detection of the fish itself, they use two algorithms and combine their results. It worth mentioning that the two algorithms are performed independently. The task of fish tracking is the responsibility of CamShift algorithm. An example of the information provided by the system designed in [16] is brought in Figure 11. The two blubs indicating the spaces where their system has noticed the presence of the fish.

Figure 11: Annotations as displayed in a processed video

The subsystem's aim for analyzing the texture and water color is to indicate the brightness, smoothness, and color of the water. This information will then help the system to decide on how to estimate the number of fish in the corresponding water environment. Their fish detection system has chosen some algorithm to do the processing required for handling large video files in a short time. They suggest that the previously used algorithms, which include subtracting a reference image, may consume so much time and processing power. Thus, they used a moving average procedure. One of the shortages to their procedure is that it does not provide good results for the outdoor scenarios. For the tracking system, they have used two algorithms, one of which is based on blob shape

matching, and the other is based on matching the generated histograms. To support their work, they have done some experiments concluding that they can reach the accuracy of up to 89.5% in the fish tracking algorithm. They claim that the near 10% inaccuracy is due to ignoring using complicated fish identification algorithms, which they avoided to decrease the processing time. They also have brought the overall counting results, which claim they have reached 90% of accuracy.

Other methods of counting fish without taking advantage of machine learning include using knowledge of electrical, optical, or sonar/acoustic engineering [3]. In [3], they provide a method of using acoustic sensors when optical, mechanical, and conductivity sensors do not provide good feedback due to their short range of applicability. They specifically investigate the use of split beam transducer, which can detect the acoustic size of the objects and their 3D position. In [3], microphone sensors have taken advantage rather than accelerometers to generate an electrical signal and turn that into a voltage that enables the counting trigger.

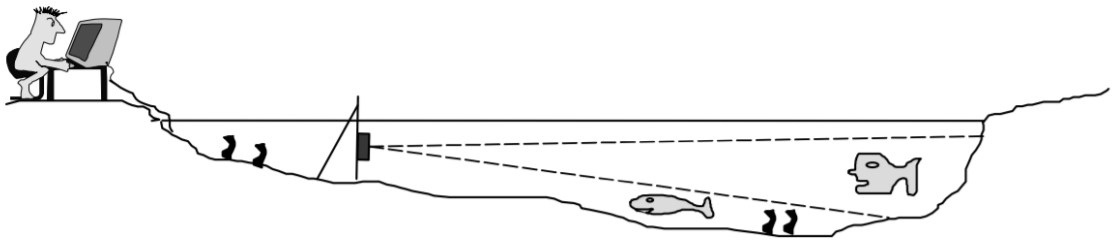


Figure 12: Schematic of the system in [3]

They have used this method to address the differences in counting fish in the river. The schematic of their work is brought in Figure 12. By using single echo detectors (SED) and because if a sound pulse hits one fish, it will return in the same amount of time, and with the same pulse shape, they have developed their fish counting system. On their way,

they had some problems, such as improving the returned pulse by applying some filters. They have also added an image analysis to their procedure to address other problems caused by the signal detection system. After receiving the pulses, they have to perform the classification to determine if the returned pulse is back from hitting a fish or other elements such as debris, lures, swimmers, water turbid, stones or even birds that dive into the river to get food. They have different features to do this categorizing based on them, some of which are the velocity, the direction change of the tracks, etc. Figure 13 is a simple diagram of the steps taken in their work.

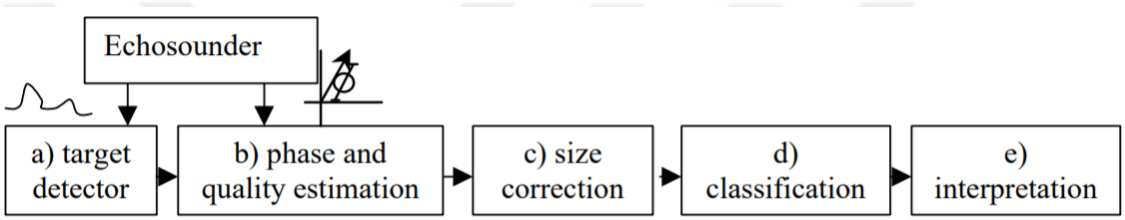


Figure 13: Steps taken in [3]

During the past years, the use of machine learning in automatic counting methods has taken a lot of attention [17]. [17] proposes a novel method of crowd counting based on learning-to-rank frame work, assuming that any sub-image of a crowd image is guaranteed to contain the same number or less number of people in it. This is offered as an answer to the problem of lack of crowd scene datasets and helps in taking advantage of Neural networks.

To have a comprehensive and complete literature review, here we bring the method used in [17] for crowd counting. This may seem irrelevant at the beginning of our work of fish counting. However, it should be mentioned as it gives a meaningful understanding of the methods used in crowd counting and can be inspiring for other

categories of counting. The techniques for crowd counting have been recent including convolutional neural networks (CNN). In their work, [17] has studied a method of self-supervising procedure to train the neural network better and improve performance. They use the trick of cropping the images into sub-images, ensuring them to have the same number of persons or less in the sub-images, giving them more guaranteed data for training purposes. This is shown in Figure 14.

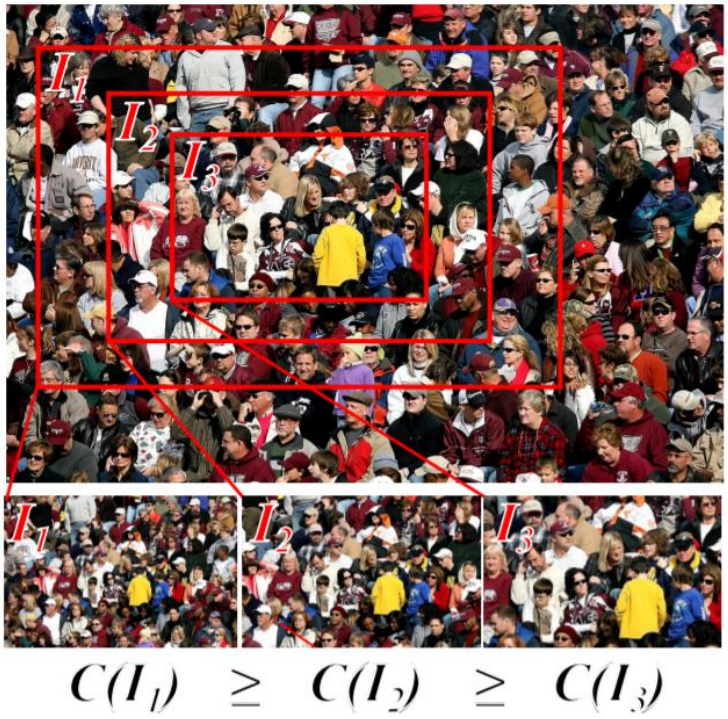


Figure 14: The cropping technique used in [17]

The use of morphological features through image processing, has been explored for type recognition, classification, and counting of fish through image processing and machine learning methods. Some works like [15] consider using morphological features for image segmentation and counting the number of fish, where the authors explore their algorithm based on the endpoints of the skeleton of the fish. They extract the skeleton of

the fish-based image thinning method. The proposed method also encounters inaccuracy when given images containing some fish on top of each other.

The experiment in [15] is set to be done in an aquarium, where a waterproof camera is inserted into the experiment platform and captures vertical video. There is also some LED around the camera lens to illuminate the water by the required amount. The captured video is then converted to a digital signal and sent to the computer unit. The experimental setup in [15] is shown in Figure 15.

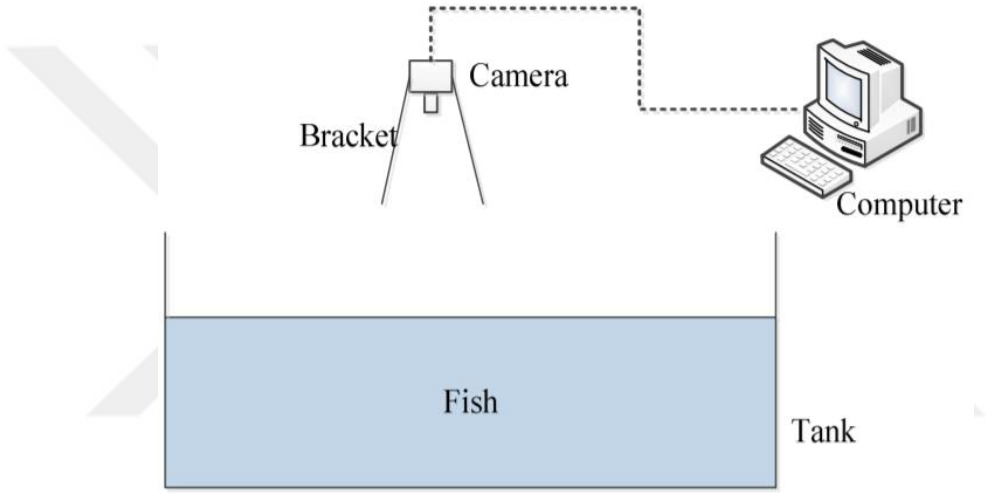


Figure 15: Experimental setup in [15]

The procedure consists of some of the same steps in similar works, such as converting the color images to gray-scale ones and removing the noise through median filters. The focus in their work is on the contour of the fish blobs rather than other information. Then the images are segmented via threshold segmentation algorithm, but with the adjusted, carefully-chosen parameters to give the best segmentation results. The main gras-scale image and the corresponding segmentation results of their work are brought in Figure 16.

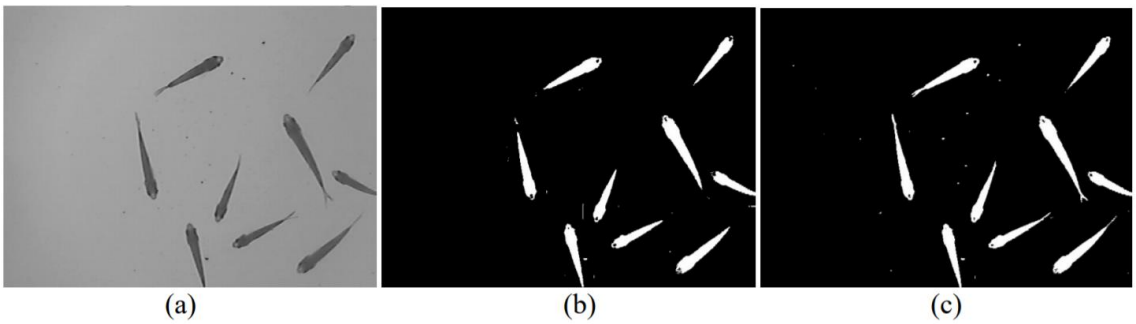


Figure 16: Different steps in segmentation in [15]

After performing the segmentation, they extract the skeleton information through image processing methods. To do so, they have to perform a so-called thinning algorithm. The results of thinning the image are brought in Figure 17.

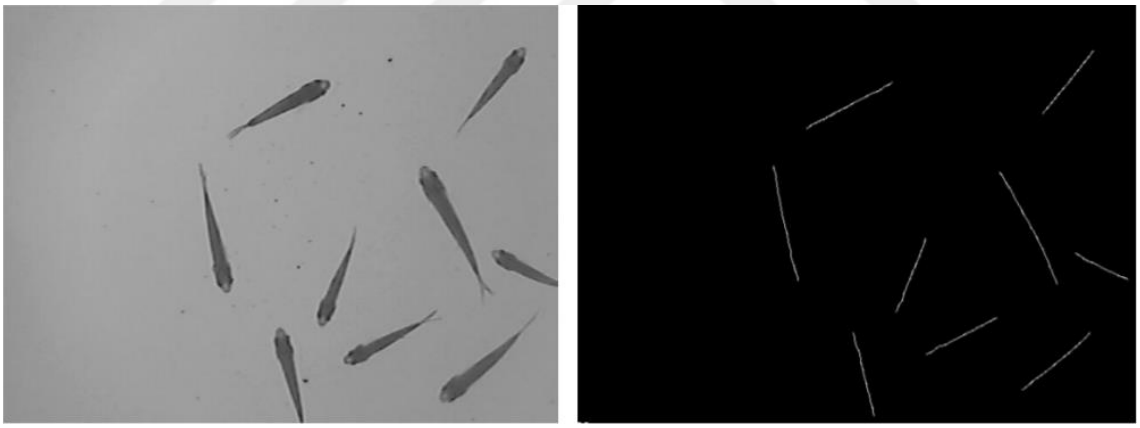


Figure 17: The gray-scale image and the resulted skeleton image in [15]

After performing all the image processing steps, they can start the fish counting procedure by counting the skeleton endpoints. They have also proposed some procedures for avoiding miscalculation of the crowds. They use the logical rule of considering n/2+1 fish if the number of endpoints in the connected figure of skeletons is n (n>4). There are

some problems with their work such as huge amount of time needed for pre-processing the images before starting the counting procedure, avoiding the method proposed in [15] to be used for real-time counting.

The method proposed in [19] is based on using morphological features to do image erosion and find the background image in order to do background subtraction and achieve a binary image. This binary image is then used to find the area of blobs and estimate the number of fish according to the median area of fish which is updated after every step. This method is only applicable to fish of the same size and shape.

They tend to process individual frames of the captured video separately and sum up the number of fish in each of them. They estimate the area of each fish blob and then estimate the number of fish in each of them. The procedure of converting the color images to gray-scale is the same. They also increase the contrast of the image by some methods. They compare their method with other methods such as threshold extraction and edge detection. To obtain a proper background extraction, they use a method called the background estimation. This method involves closing the frame with a proper subtracting element to be used for background subtraction. The result of their method is brought in Figure 18.

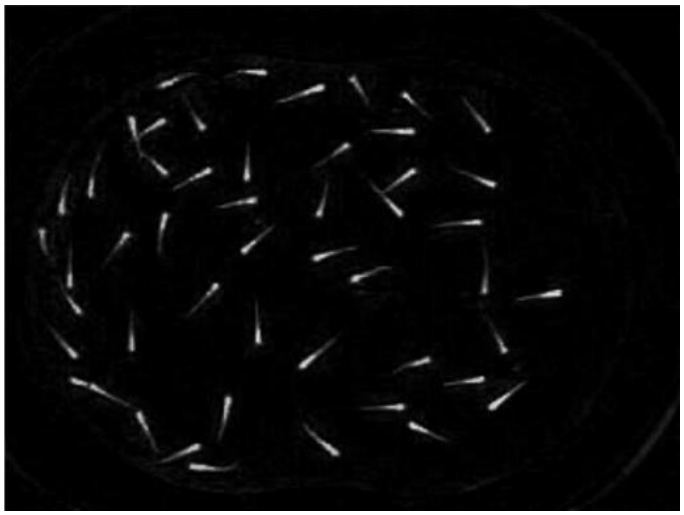


Figure 18: Background subtraction result in [19]

After performing the above image processing tasks, they start counting the fish through estimating the blobs and considering the average fish area for the special type of fish being counted.

As discussed above, use of morphological features has been explored in machine vision methods of counting rather than machine learning techniques and most of the works involving machine learning include training data sets of images of blobs of fish (or other counting objects) via a specific algorithm rather than first using image processing techniques in order to extract the quantities of suitable morphological features and then training the data sets involving vectors of morphological features via proper choices of algorithms which are most appropriate for the intended data set.

In [20], the authors have proposed using data sets of morphological features to train a model to predict the number of fish. However, the choice of morphological features has not been in a fashion that it reflects the best possible correlation between the data set and the number of fish in the images of blobs (one of the examples of such features used in this work is bonding box width and height, which is not a good choice in comparison

to the ratio of width to height). The overall choice of these features has resulted in less accuracy than what could be achieved through good choice of features. They have considered using support vector machine (SVM) algorithm for their work. They have chosen SVM because it does not need large data sets for the training phase. They also have used least-square SVM as it includes the same advantages of the SVM in an optimized fashion.

In their practical setup, they obtain the images via a video camera from the scenes illuminated by a light source. Their light system consists of a pair of LEDs that are installed on both sides of their aquarium. The camera is installed on top of the area. They have used mostly the same procedures for gray-scaling the image and doing the background subtraction. They have also removed the noise from their images though thresholding. Different steps of their work are brought in Figure 19.

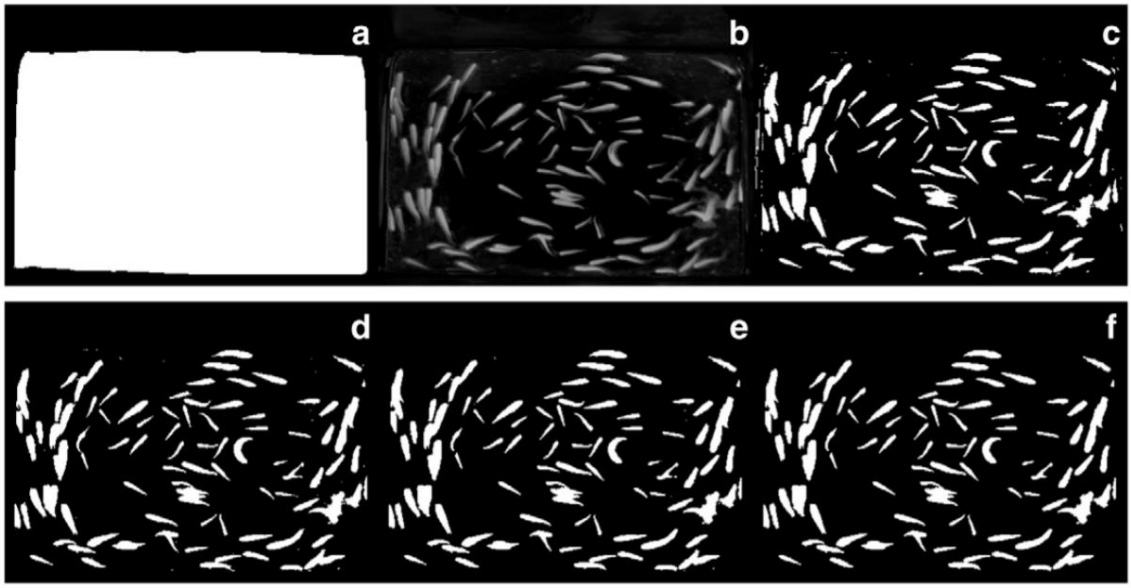


Figure 19: Image processing steps in [20]: (a) mask, (b) original image, (c) binarized image, (d) ‘AND’ image, (e) filtered image through median operation, and (f) ultimate result.

After performing the mentioned procedures, they have used feature extraction on the skeleton-like images of the fish blobs. They finally use these results to do the counting process.

In order to take advantage of both machine learning and image processing, we propose a method in this work, in which we define and use suitable and most optimal morphological features to train the dataset via machine learning algorithms in a fashion that the algorithms perform at their best. Therefore, the resulting counting process performs at its fastest, giving the most accurate results compared with the mentioned works. This wise choice of features frees us from restrictions like the necessity of having the same size and shape fish, enabling us to use the method for every mixed set of fish with different sizes and shapes.

Also our method provides less complicated and less-steps process of counting enabling both real-time and continuous fish counting as a result of high speed of our counting process resulted from good choice of features enabling us to use machine learning algorithms like random forest which turns out to be one the best and fastest possible choices (both in train and predict) for the nature of our problem and also due to using line scan camera rather than area scan camera (used in all of the above works).

CHAPTER II

BACKGROUND

This section explains image processing and morphological operations, features, some of which them used in this work and then it explains some of the machine learning algorithms used in this work.

2.1 Image Processing

A digital image is a two-dimensional function, $f(x, y)$, where x and y are planes and spatial coordinates, and the value of x and y is the intensity or gray-level of the image at that points. When x, y, and amplitude value f, all are finite and discrete we can call it a digital image [18]. In other words, an image is a two-dimensional array, which arranged in rows and columns.

$$f(x,y)=\begin{bmatrix} f(0,0) & f(0,1) & f(0,2) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & f(1,2) & \dots & f(1,N-1) \\ \vdots & \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & f(M-1,2) & \dots & f(M-1,N-1) \end{bmatrix}$$

The finite number of elements composed the digital image, each of these elements has a specific value at a particular location, which called a pixel.

Digital Image Processing, processing the digital image by using a digital computer. By using computer algorithms, in order to get the enhanced image and then

extract some useful information and analyses of the image.

2.1.1 Morphological Image Processing

The morphology word denotes a branch of biology, which from the structure of plants and animals. Binary images may have a lot of imperfections, morphological image processing try to remove these imperfections by using some technics and operations. Sometimes we can apply these technics to greyscale images. Morphological image processing is a collection of operation which is non-linear and related to the shape of an image [21].

2.1.1.1 Dilation

The dilation of A by B, denoted $A \oplus B$, is defined as

$A \oplus B = \{z \mid (\hat{B})_z \cap A \neq \varnothing\}$,

where A and B are set in Z^2.

Dilation changes the inner and outer boundaries pixel value to one and inserts a layer of pixel to the inner and outer boundaries. The gaps between different regions become smaller, and small holes enclosed by a single region and small holes in the boundaries of a region are filled.

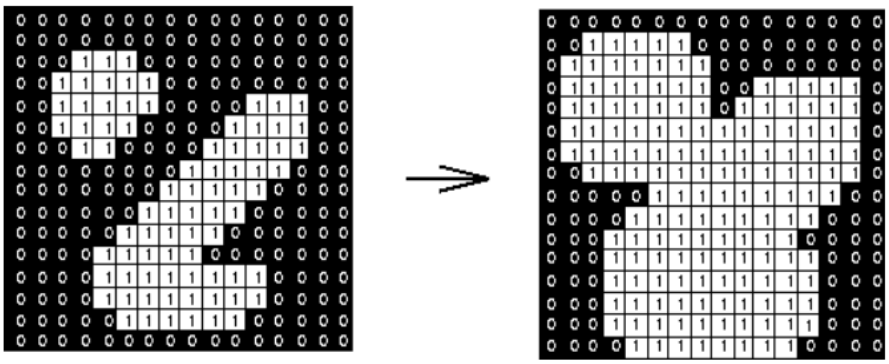


Figure 20: Dilation: a 3×3 square structuring element

Figure 21: Left image is binary image and the right image is the result after dilation with 2*2 structuring element

2.1.1.2 Erosion

The erosion of A by B, denoted $A\Theta B$, is defined as

$A\Theta B = \{z \mid (B)_z \subseteq A\}$,

where A and B are set in Z^2 .

Erosion shrinks a binary image by changing the value of pixels from the inner and outer boundaries of the region to 0 and causes the gaps and holes in the different regions to become bigger and small artifacts are eliminated [21].

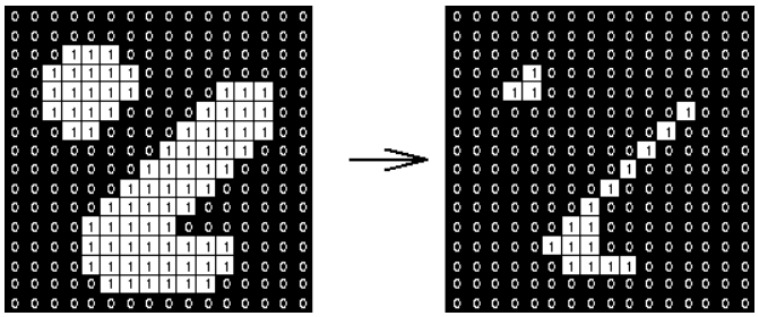


Figure 22: Erosion: a 3×3 square structuring element

Figure 23: Left image is binary image and the right image is the result after erosion with 2*2 structuring element

2.1.1.3 Opening

Opening eliminates thin protrusions and smoothed the contour of a binary image. The opening of an image A by a structuring element B show as below:

$$A \circ B = (A \Theta B) \oplus B.$$

The opening can open up gaps between objects connected by a thin bridge of pixels. It makes each regions erosion and afterward restored by dilation [21].

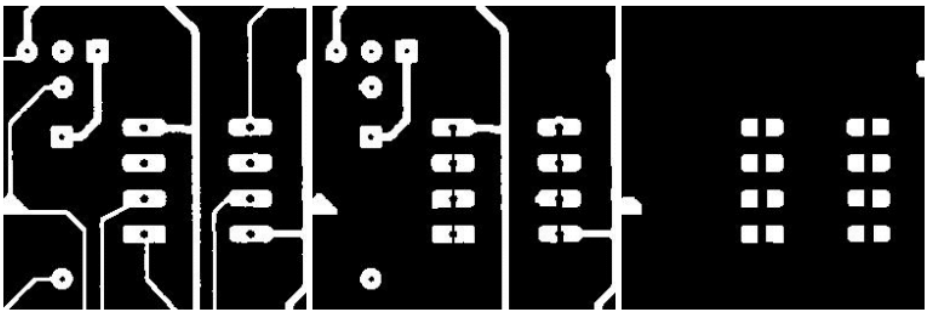


Figure 24: Left image is original binary image, the center image is opened image by 5X5 square, right image is opened image by 9X9 square.

2.1.1.4 Closing

Closing fills, the holes in the region while keeping the initial region size. The closing of an image A by a structuring element B show as below [21]:

$$A \bullet B = (A \oplus B) \Theta B.$$

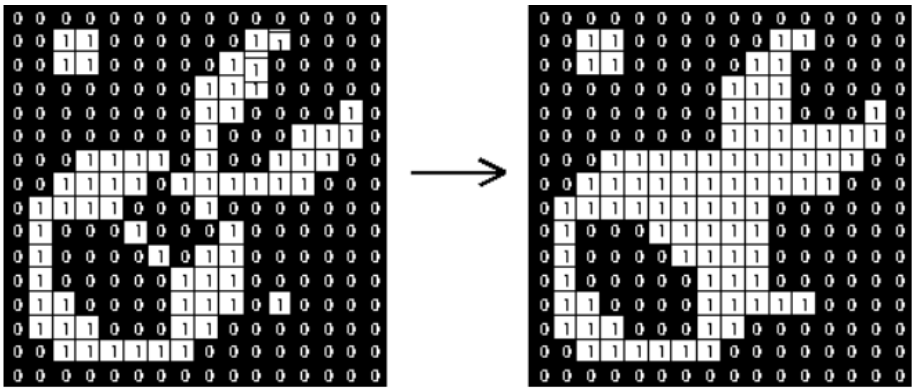


Figure 25: Closing with a 3x3 square structuring element

2.1.2 Morphological Features

This section explained some morphological features which some of them used in this work directly and some others didn't used directly but used indirectly.

2.1.2.1 Area

Area is the number of pixels in the blob and calculated from the contour of shape as follows:

$$Area(Polygon) = \frac{\sum_{i \in Contours} (x_i + x_{i-1}).(y_i - y_{i-1})}{2}$$

$$Area(ellipse) = \pi.ab$$

Figure 26: Area example

2.1.2.2 *Convex Area*

The convex area is the area of the convex hull that encloses the object.

Figure 27: Convex Area example

2.1.2.3 *Area Rate*

We found that we can make use of the "area" feature indirectly by turning it into a more meaningful indicator and use it to train the model alongside some other features, instead of using it directly to determine the number of fish by just a simple division. It is defined as $A_{rate} = A / \bar{A}$, in which $\bar{A}$ is the average area and A is the area of the blob defined for the contour around the periphery of a blob indicated by points of $(x_i, y_i), i \in contours$, as

$$A = \frac{\sum_{n \in contours} (x_n - x_{n-1})(y_n - y_{n-1})}{2}$$

2.1.2.4 *Perimeter*

Perimeter is the number of pixels in the boundary of the shape. For Polygon Shapes, the Perimeter is calculated from the shape's contour as:

$$p = \sum_i \sqrt{(x_i - x_{i-1})^2 . (y_i - y_{i-1})^2}$$

Figure 28: Perimeter of a shape

2.1.2.5 Convex Perimeter

Convex perimeter is the perimeter of the convex hull that encloses the object.

Figure 29: Convex perimeter show in blue circle

2.1.2.6 Roughness

In addition to using area indirectly, we can make use of the perimeter indirectly as well. This feature indicates how rough a blob is and is found by $R = P / P_{convex}$, in which P is the perimeter of the blob and P_{convex} is the convex perimeter defined as the perimeter of the smallest convex hull which encloses the blob. Obviously, roughness has a minimum value of 1 for a smooth convex object.

2.1.2.7 Elongation

Elongation is the ratio of the length and width of the object bounding box.

$$Elongation = \frac{width_{bounding-box}}{length_{bounding-box}}$$

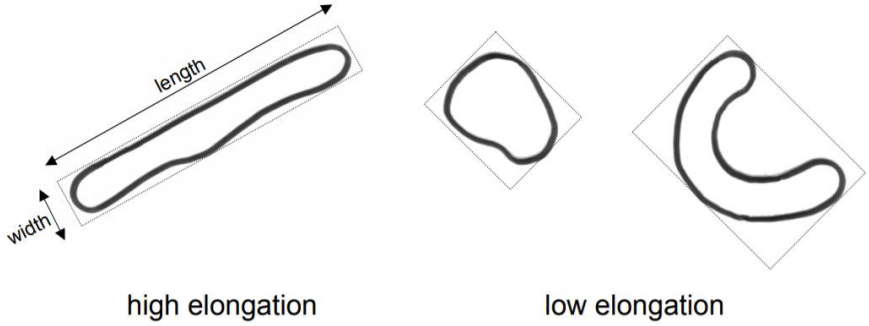


Figure 30: Example of high and low elongation

2.1.2.8 Compactness

Compactness is defined as a measurement of how much a shape is compact as close to a circle and it is defined as the ratio of the area of a circle with the same perimeter of the considered shape to its area, thus after some mathematical manipulation, we have compactness as

$$C = \frac{P^2}{4\pi A}$$

Obviously, the minimum amount value of 1 is for the circle and as the shape becomes more convoluted with irregular boundaries it becomes bigger.

Figure 31: An example of various compactness value

2.1.2.9 Roundness

Roundness describes the Shape's resemblance to a circle. The Roundness factor of a Shape will approach 1.0 the closer the Shape resembles a circle. The Roundness factor of a square is $2/\pi \approx 0.636$. Roundness is defined as

$$Roundness = \frac{Diameter^2}{MaximumDiameter^2} = \frac{4 * \frac{Enclosed\_Area}{\pi}}{MaximumDiameter^2}$$

where Maximum Diameter is the diameter of the minimal enclosing circle, i.e. the smallest circle enclosing all points of the Shape, and the Diameter is the diameter of a circle with the same area as the area enclosed by the Shape contour. This area is larger than the reported Area parameter if the Shape contains holes.

2.1.2.10 Minimal Enclosing Circle

Calculating the minimal enclosing circle is CPU intensive. Therefore, we didn’t use this feature in our work.

Figure 32: Minimal enclosing circle

2.2 Machine Learning

Tom Mitchell (1998) Well-posed Learning Problem: A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.

Some technics in machine learning inspired from human and animal brain how they learn and derive from the efforts of psychologists to make more precise their theories of animal and human learning through computational models [22].

2.2.1 Types of Machine Learning Algorithms:

2.2.1.1 Supervised Machine Learning

In supervised machine learning, there is a human-labeled data to train and generate a model to predict the unseen situation. Examples of Supervised Learning: Regression, Decision Tree, Random Forest, KNN (K-Nearest Neighbor), Logistic Regression, etc.

2.2.1.2 Unsupervised Machine Learning

In contrast in unsupervised machine learning there is not a human-labeled data to train but the model tries to find the undetected patterns in the data. Examples of Unsupervised Learning: K-means, Apriori algorithm.

2.2.1.3 Semi-Supervised Machine Learning

Semi-supervised machine learning is a combination of both, supervise and unsupervised machine learning model which use a small amount of labeled data with a large amount of unlabeled data to training and generating the model.

2.2.1.4 Reinforcement Machine Learning

Reinforcement machine learning enables an agent to learn in an interactive environment with try and error using feedback from its actions. Reinforcement machine learning using rewards and punishment for positive and negative behavior. Example of Reinforcement Learning: Markov Decision Process.

2.2.2 Some of Machine Learning Algorithms

In this section, I will explain the common machine learning algorithm which applied to solve the data problems.

2.2.2.1 Linear Regression

The linear regression estimates the real values based on continuous variables. It fits the best line for predicting the new value.

$$Y = a \times X + b$$

where Y is the dependent variable, a is the slope, X is the independent variable, and b is intercept, which a and b derived based on minimizing the sum of squared difference of distance between the data value and line.

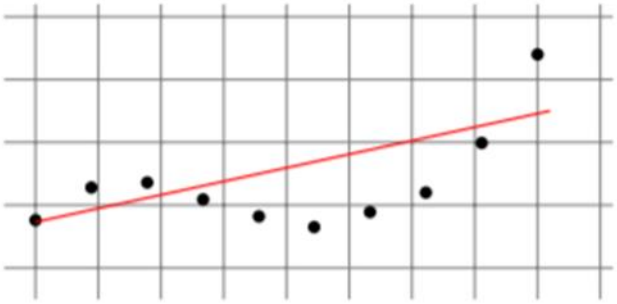


Figure 33: Fitting a line on data points

2.2.2.2 Logistic Regression

Unlike the logistic regression name, it is a classification algorithm, not a regression algorithm. Because it is a classification algorithm it is used to estimate discrete values based on independent variables. In other words, it predicts the binary values and predicts the probability of occurrence of an event by using a logic function. The output of the logistic regression model will be between 0 and 1.

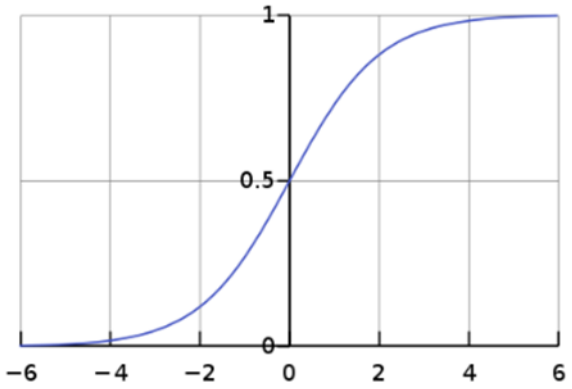


Figure 34: Sigmoid function

2.2.2.3 Decision Tree

The decision tree is a supervised machine learning, which works for continuous and categorical dependent variables, but mostly used for classification problems. The trained model predicts the input data during decision rules inferred from the data. Decision tree split the data to subsets and nodes are a splitting point based on a certain feature from the data and recursively for each subset generate new tree nodes, we continue to splitting the data until reach a point where we have optimized with maximum accuracy.

For example, in Figure 35 we have 3 features (Humidity, Outlook, Wind) firstly calculate the entropy of each feature then split them during the entropy of each feature.

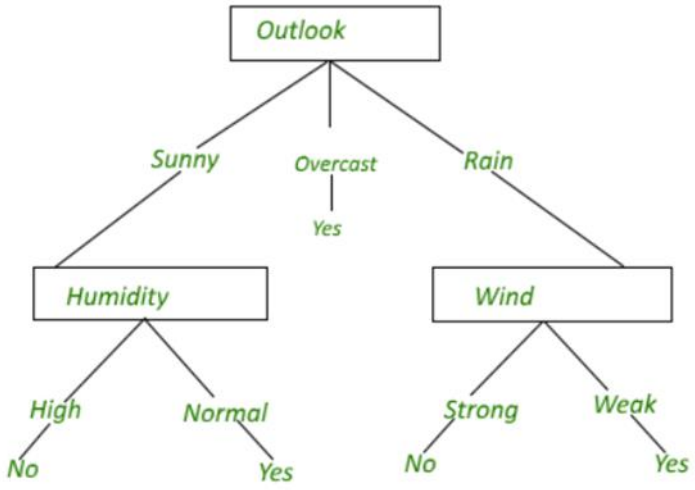


Figure 35: Sample of decision tree

The cost function for decision tree regression is:

$$E = \sum (Y - \hat{Y})^2$$

where Y is True labels and Y-hat is predicted value by the model.

The cost function for classification is:

$$E = \sum (p_k \times (1 - p_k))$$

where p_k are the training instances of class k.

2.2.2.4 Random Forest

Random forest is a collection of decision trees because of that, it's known as random forest, each tree has a vote during the prediction, then during the majority of the vote random forest make a final decision. Figure 36 shows that.

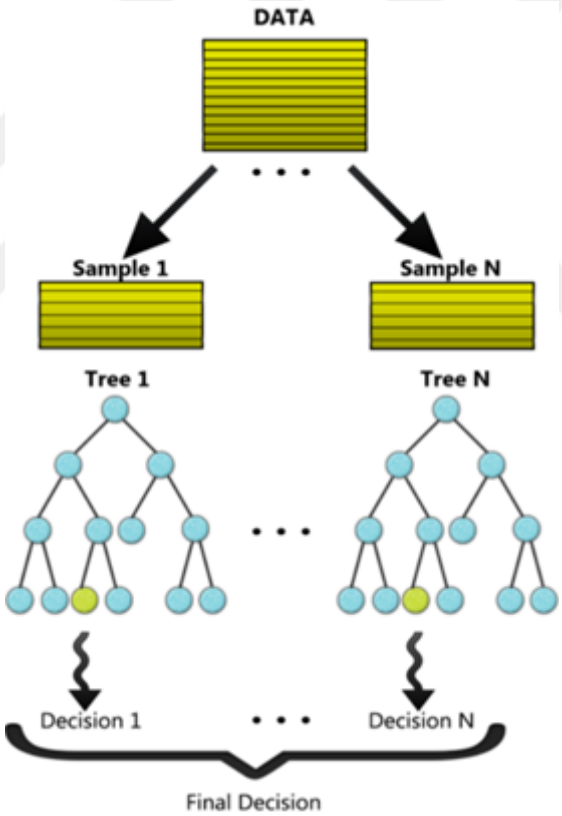


Figure 36: Random Forest

2.2.2.5 SVM (Support Vector Machine)

The SVM algorithm is very famous and applied for classification and regression problems. It’s appropriate to linear predictors in high dimensional feature spaces because

the SVM algorithm tackles the complexity challenge for searching “large margin” separators [23]. SVM finds the best possible lines which maximize the margin length, Figure 37 shows that.

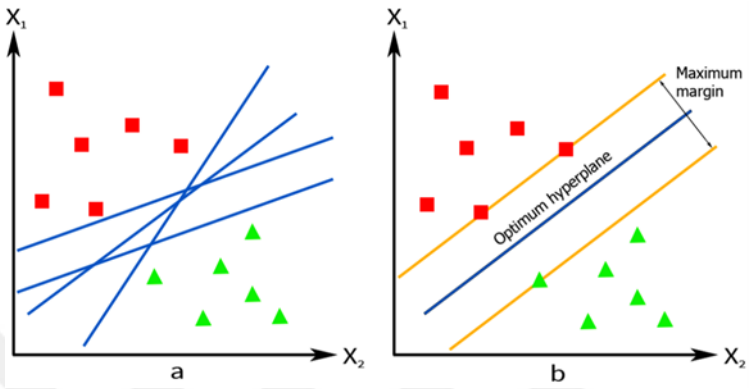


Figure 37: The best possible margin length

2.2.2.6 Naive Bayes

Naive Bayes is a probabilistic machine learning algorithm and based on Bayes Theorem. Naive Bayes is very simple, despite the simplicity it is fast, accurate, and reliable. It’s used in a variety of classification problems.

Conditional probability is a probability of an event occurring given that event has occurred.

$$P(A \mid B) = \frac{P(B \mid A).P(A)}{P(B)}$$

P(A|B) is the posterior probability of class.

P(A) is the prior probability of class.

P(B|A) is the likelihood which is the probability of predictor given class.

P(B) is the prior probability of predictor.

2.2.2.7 KNN (k- Nearest Neighbors)

K-Nearest Neighbor algorithm can be applied for both, classification and regression problems. But it mostly used for classification problems. The idea is to memorize the training set in feature space and then predict the new instance by a majority vote of k closest neighbors in the training set which the number of k is should be an odd number [23]. If we select a big number for k, the computational will be very expensive. Figure 38 shows an example of KNN.

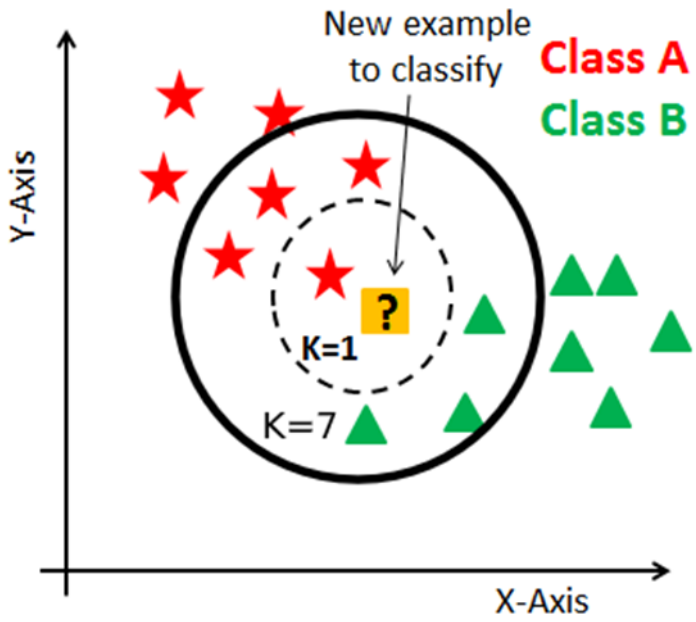


Figure 38: Example of KNN

2.2.2.8 K-Means

One of the unsupervised algorithms, which is used for clustering, is K-means algorithm. It is one of the most popular "clustering" algorithms, by using K-Means the clustering task change into optimization problems [23].

Figure 39 shows an illustration of K-means algorithm in two-dimensional data clustering, which (a) is data points in two-dimensional, (b) shows 3 random location of clustering we called them centroids, (c) shows 2D spaces are divided into three regions, (d) shows each centroid move to the center of the data points, (e) shows the newly updated location of centroids, the (c) and (d) steps are repeated until convergence occurs and (f) show the final result.

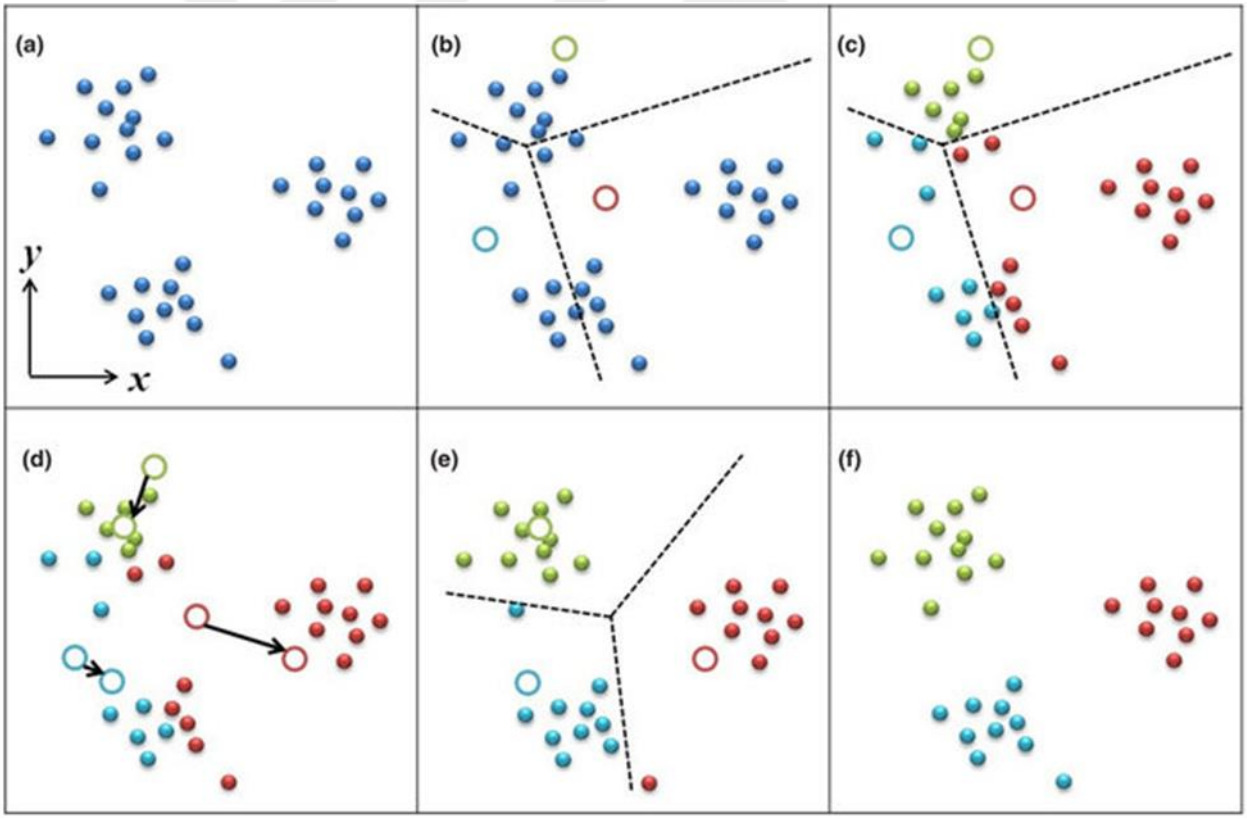


Figure 39: A schematic illustration of the K-means algorithm

2.2.2.9 PCA (Principal Component Analysis)

In some situation, we have a lot of features in the training set, but all of them is not useful in the training because they are uncorrelated data. The main idea of PCA is to reduce and eliminate the un-useful features and sort the features during their importance. By doing that the dimensionality of the data is reduced and the training task will be fast. The principal components are the eigenvectors of a covariance matrix. Figure 40 shows PCA how to work.

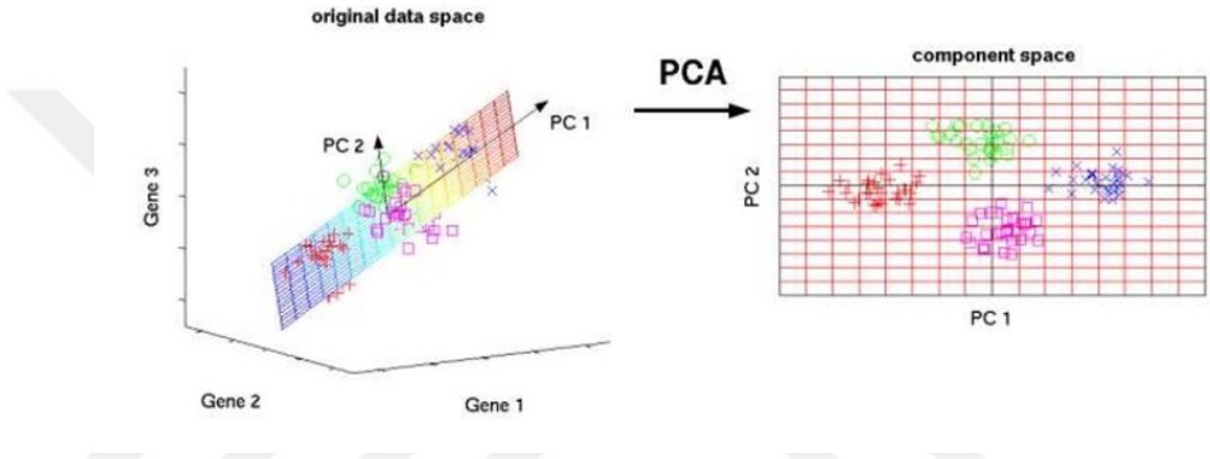


Figure 40: PCA

CHAPTER III

THE PROCESS AND RELATED CHALLENGES

This section introduces the process of capturing the images of fish, counting process, and proposed methods and solutions.

3.1 The Process

This section introduces the overall process of capturing the images of fish in the stream of water and preparing those images for the process of counting.

As it is seen in Figure 41, the fish are passing by a camera in the stream of water and the camera captures consecutive images of them which are then prepared for the other steps of processing and counting.

Use of area scan camera results in the necessity of adaptive thresholding to produce binary images and loads an extra heavy process on the job, resulting in the whole counting process to take more time and fail in being real-time. In comparison to area scan camera, in our work, using an innovative use of the camera, we have removed the extra steps mentioned above, giving us the advantage of real-time counting, which is absent in all of the other works.

Figure 41: The fish are counted while passing by the camera in the stream of water, depicting the power of our method for continuous and simultaneous counting

Figure 42 brings an example of the images used in our work, which is taken by proper choice of pixels and imaging frequency.

The points mentioned above, alongside other minor reasons show the power of the robust design taken into consideration in order to enable both real-time and continuous fish counting providing the fish farming with a much less time consuming and mush easier technique in cheaper total costs.

Figure 42: An example of images taken by the camera

As mentioned previously, our process is independent of the fish size and type and our method can be used for any type and size of fish. This method is designed to work directly with the camera and take the images and do the processing on them in a real-time fashion, resulting in giving the number of counted fish in real-time. After grabbing taking the images from the camera, the median filter is applied to remove small artifacts and then by applying binary thresholding, binary images are produced. Through performing image processing techniques on such images, the connected regions are specified to acquire the fish images and their contour specifications as arrays.

In addition to that, the method is capable of processing pre-saved images (with different formats such as tiff or jpg) on the disk. According to the chosen length of the images, the system waits for the sub-images from the camera and forms the complete

images, saves them to the disk and starts processing them according to the model (which will be discussed in the following sections) in order to give the fish count in that image. An example of processed image is shown in Figure 43. As it is seen, our process consists of segmenting each image into groups of 1, 2, 3 or more fish (segmented images enclosed by green rectangle for 1, red rectangle for 2, blue rectangle for 3, and black rectangle for more than 3 fish) and counts each of them which results in giving the total count of fish in the image. Our work is also can take advantage of making the dataset by saving the segments of the images into the disk, while one wants to make a data set of the images of the fish in that fish farm in order to use the data set in the feature extracting and training process to adapt the method with that type and size of fish. The morphological features of each segment are then found and saved in an excel file and will be used later for training the model.

Figure 43: An example of processed image

To help understanding the working of the process, a simplified summary is brought in Figure 44, while Figure 45 shows an example of the segments found in different frame images.

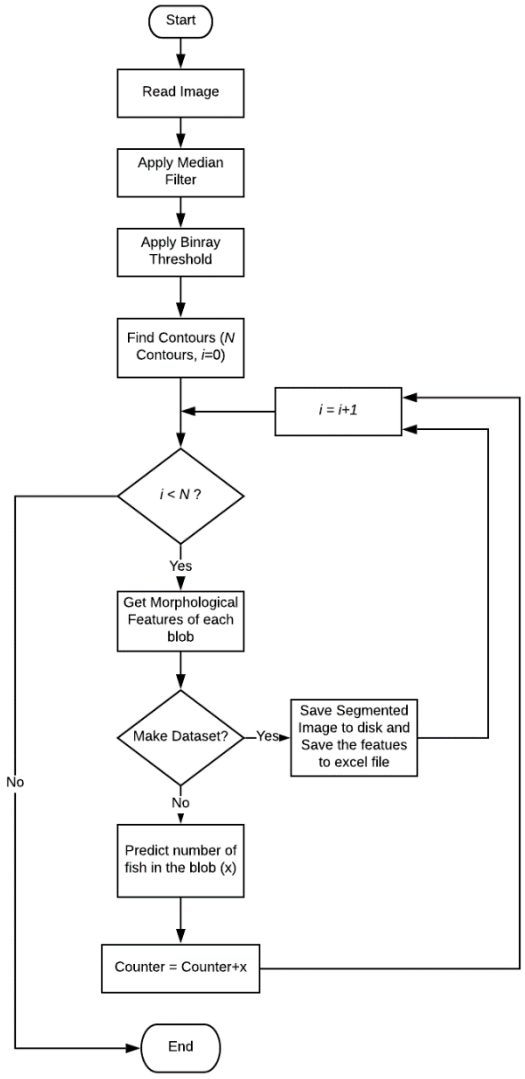


Figure 44: Step-by-step flowchart of the counting procedure in the software

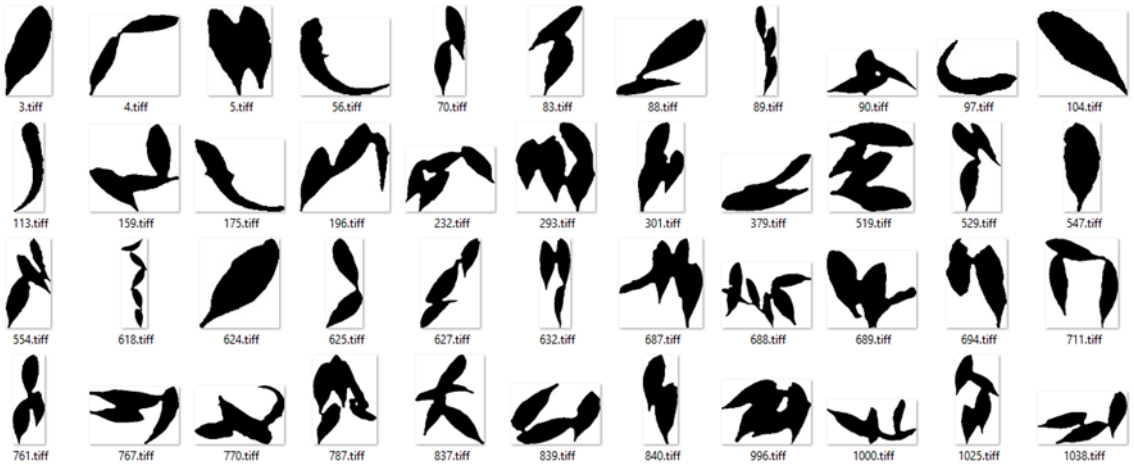


Figure 45: Example of image segments resulted from image processing techniques

3.2 The Counting Process Challenges

Here, we go through the two most important problems which may interfere with the counting algorithms that are based on image processing applications. The introduction of these two problems make it possible to better understand the reasons for which we use the proposed method which is going to be discussed in next section and show the advantages of the method alongside with the novelties used in it.

A. Consecutive Frames Problem

As mentioned in previous section, the process consists of counting fish in each frame consisting of attached one-pixel-length images taken of the flow of fish with a certain frequency, thus making it possible for some fish to be repeated in two consecutive frames. This results in repeating the process for some number of fish which are existing in both of the frames twice, causing inaccuracy in the overall result of the counting process. An example of such repetition is brought in Figure 46 (as the emphasis is on the bottom of

the first frame and top of second frame, the two frames has been cut to only include the parts around the frame borders).

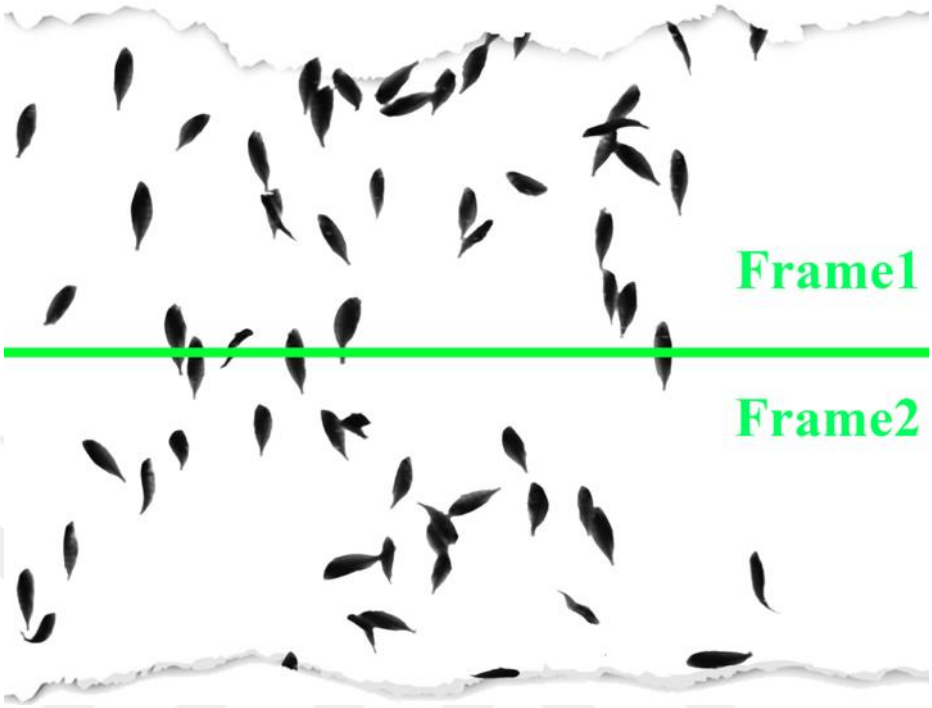


Figure 46: Example of shared fish images in two consecutive frames

B. Multiple-fish Segments (Overlapping) Problem

To be able to take advantage of the real-time continuity of the counting process, we have to give up the control over the style that fish are entering the counting mechanism and pass in front of the line-scan camera. This makes us unable to be sure about the number of fish in each segment (here better called blobs), as fish can appear beside each other or on top of each other, resulting in to have blobs of different numbers of fish which are beside or on top pf each other in different fashions and directions. This means that by only knowing the number of blobs, one is unable to know the number of the fish and even is unable to have a rough estimation of the number which is even close to the true number of the fish.

C. Composite Problem (Combination of Problems A and B)

When blobs consisting of multiple fish are shared between two consecutive frames, not only the same fish are repeated between the two frames, but also the real count of the fish in each blob is not certainly known.

3.3 Proposed Methods and Solutions

To address the two mentioned problems (and some more which are not mentioned here as they are less effective on the counting accuracy), we have proposed some solutions. These solutions are introduced in this section and then are compared with each other in terms of ease and accuracy through extensive numerical results and discussion.

A1. Addressing the Consecutive Frames Problem through Mathematical Approach

In order to approach this problem, in two consecutive frames, one must care about the first frame bottom most row of pixels and the second frame top most row of pixels as the possible repetition of objects in two consecutive frames would be related to these aforementioned rows. In this part (III-A) we focus on the simple case where the shared blobs between the two consecutive frames contain only one fish each. We address the more complicated case of probable-multiple fish in some shared blobs (i.e. the composite problem) in the corresponding section of III-C. After scanning these two rows of pixels in the grayscale images, we can plot the black-white intensity diagram of each separately, as the white background is interpreted as 250 and black pixels (which show the existence of fish in our work) are interpreted as zero. The color intensity diagrams of the two

considered rows are brought in Figure 47 left and right. For a better understanding, without loss of generality, we binarize the diagrams by setting the threshold of 200, i.e. the we consider more than 200 as white background and set it to zero and consider less than 200 as object and set it to 1 (The binarized diagrams are shown in Figure 48 left and right). Then we convert the diagrams into a more meaningful one as follows: color alteration from white to black (0 to 1) is considered to be 1 and color alteration from white to black (1 to 0) is considered as -1. The resulted diagrams are shown in Figure 47 left and Figure 47 right. Now the number of the number of 1 and -1 bars in the two diagrams is critical in specifying the number of shared fish between the two borders. We name the number of bars of 1 and -1 for the first frame as n_1 and n_2 also the number of bars of 1 and -1 for the second frames as n_3 and n_4. The exact number of fish shared on the border of two consecutive frames will then be

$$Number\ of\ shared\ \ fish = \max\left(n_1, n_2\right) + \max\left(n_3, n_4\right) - \min\left(\max\left(n_1, n_2\right), \max\left(n_3, n_4\right)\right)$$

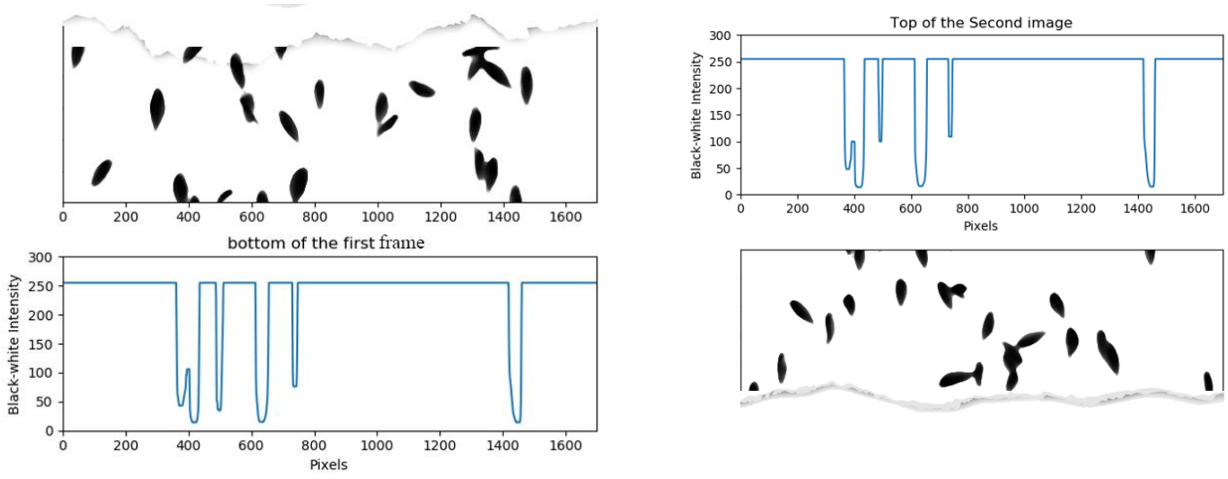


Figure 47: Left and right are Color intensity for the bottom of the first frame and the top of the second frame

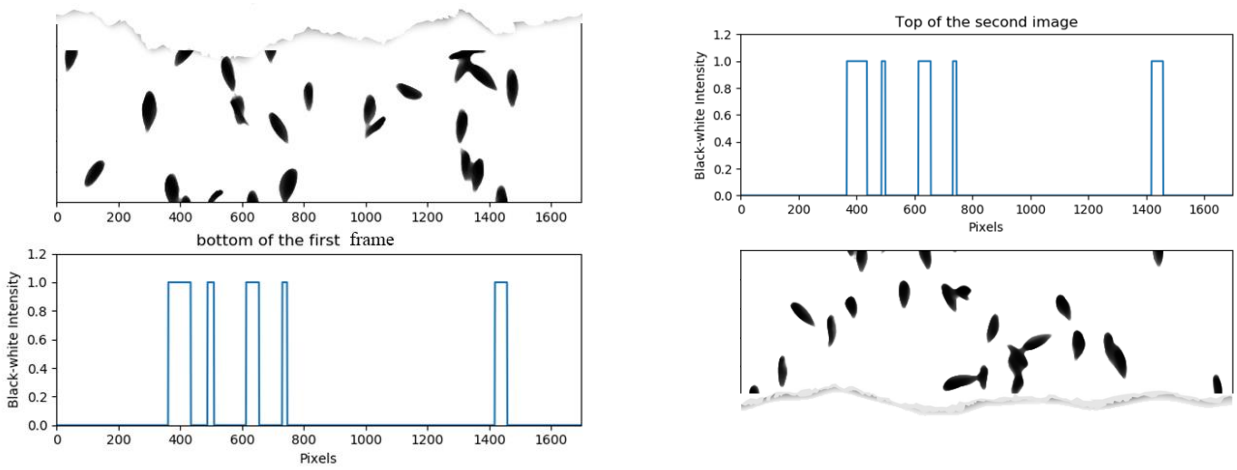


Figure 48: Left image and right image - Binarized corresponding diagrams

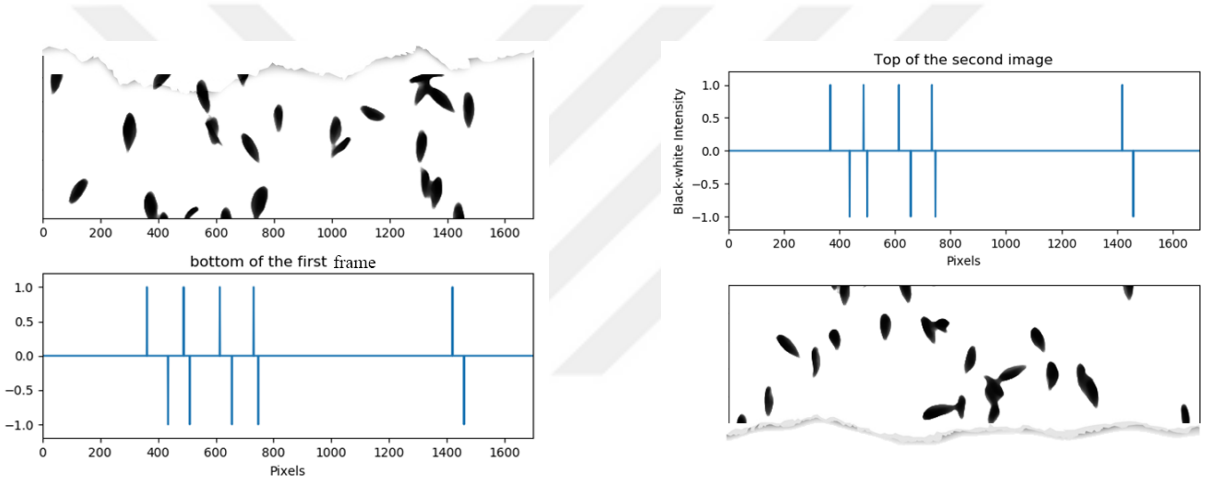


Figure 49: Left and right images - “1” and “-1” corresponding diagrams

It is obvious that the visualized interpretation above is brought here to help better understanding the issue and the logic behind the proposed equation for the reader, thus the diagrams are not used in the programming process.

A2. Addressing the Consecutive Frames Problem through Merging

The other way to solve the problem in section III-A is that instead of the mentioned mathematical approach we can merge the shared blobs of fish in the previous image with the next image. For that purpose, a special algorithm is designed as follows: firstly, we

define two arrays, one for saving the bounding-box of shared images in the first frame as vectors (named *ImageArray*) and the other one for saving the contour specifications of the bounding boxes (named *ContoursArray*). These contour specifications include location of the upper right corner of the bounding-box in the Cartesian coordinates (x, y) and also the width (w) and the height of the rectangular bounding-box (h). Then, before starting the analysis of the second frame, the *ImageArray* and *ContoursArray* of the previous image are checked. If the two arrays are null, it means there has not been any shared image of fish with the current image, thus continuing with the analysis of the new image, but if they are not null, merging between the two frames should be done as follows: from the *ContoursArray*, the maximum height (h) of the bounding-boxes is found and a blank image is made with the same height as h and normal width of the images; then each image from the *ImageArray* is copied to that blank image in the correct place (by taking advantage of the *ContoursArray*), starting from the bottom. Now the new filled image and the second frame can be concatenated to form a new frame which does not have any shared images of fish with the previous frame. Figure 50 shows an example of such a process.

Figure 50: An example of merging

B. Addressing the Multiple-fish Segments Problem by ML (Machine Learning)

As mentioned previously, most of the previous works on the fish counting which are based on image processing, use some techniques to determine the blobs and then using the area of each blob and the mean value of the area of a single fish of that type, they determine total number of the fish. This is simply not accurate at all, as not only the mean value is not adequate because the variations of the areas may be large, but also this is based on the assumption that all the fish in the images pass in front of the camera in a certain style for which the average area is assumed and also it is assumed that the fish are not covering each other in front of the camera and they all are beside each other with the minimal amount of blockage in the image. A simple example of such inaccuracy in counting via this method is brought in Table 2.

Table 2: Huge inaccuracy in using only the area feature

Fish Blob	Predicted Fish Number with area of blobs
	2
	1

To address such a problem, we propose to take advantage of machine learning and instead of some works which use the images of blobs directly to train the model and use that model to predict the number of fish directly on the images of the blobs for which the number of fish need to be counted, we propose using feature extraction. This enables the method to work faster both in training and predicting phases. Also using the suitable morphological features adds up to the accuracy of the counting job significantly. Therefore, we first chose the most related morphological features and used different combination sets for training procedures and through Principal Component Analysis (PCA) methods, we came up with the best features that result in the highest accuracy for the counting job.

Thus, the following morphological features are chosen to be used in the training and predicting procedures.

3.3.1.1 Area rate

We found that we can make use of the “area” feature indirectly by turning it into a more meaningful indicator and use it to train the model alongside some other features, instead of using it directly to determine the number of fish by just a simple division. It is defined

as $A_{rate} = A / \bar{A}$, in which $\bar{A}$ is the average area and A is the area of the blob defined for the contour around the periphery of a blob indicated by points of $(x_i, y_i), i \in contours$, as

$$A = \frac{\sum_{n \in contours} (x_n - x_{n-1})(y_n - y_{n-1})}{2}$$

3.3.1.2 Roughness

In addition to using area indirectly, we can make use of the perimeter indirectly as well. This feature indicates how rough a blob is and is found by $R = P / P_{convex}$, in which P is the perimeter of the blob and P_{convex} is the convex perimeter defined as the perimeter of the smallest convex hull which encloses the blob. Obviously, roughness has the minimum value of 1 for a smooth convex object.

3.3.1.3 Compactness

This is defined as a measurement of how much a shape is compact as close to a circle and it is defined as the ratio of the area of a circle with the same perimeter of the considered shape to its area, thus after some mathematical manipulation, we have compactness as

$$C = \frac{P^2}{4\pi A}$$

Obviously, the minimum amount value of 1 is for the circle and as the shape becomes more convoluted with irregular boundaries it becomes higher.

To show the idea behind this and also to give an insight of the reason behind using these morphological features among other features, Table 3 shows the values of selected morphological features for the same data set of Figure 45 correspondingly.

Table 3: Values of morphological features of corresponding dataset of Figure 45

#	Compactness	Roughness	Area Rate	Perimeter	Area	Average Area	label
3	1.800	1.049	0.711	269.664	3,214.500	4,524.083	1
4	4.861	1.123	1.122	556.902	5,077.000	4,524.083	2
5	2.261	1.276	1.223	396.434	5,532.500	4,524.083	2
56	4.490	1.195	0.553	375.706	2,502.000	4,524.083	1
70	4.013	1.259	1.288	541.990	5,825.000	4,524.083	2
83	3.638	1.272	1.104	477.772	4,993.500	4,524.083	2
88	4.171	1.340	1.552	648.340	8,020.500	5,168.242	2
89	5.629	1.253	1.919	837.612	9,919.000	5,168.242	3
90	3.291	1.152	1.367	540.642	7,067.000	5,168.242	3
97	4.012	1.237	0.250	255.380	1,293.500	5,168.242	1
104	2.135	1.058	0.534	272.191	2,762.000	5,168.242	1
113	4.153	1.102	0.371	241.723	1,119.500	3,018.898	1
159	4.145	1.353	2.504	627.428	7,558.500	3,018.898	3
175	4.344	1.124	0.698	339.120	2,106.500	3,018.898	1
196	5.278	1.429	2.346	685.453	7,083.500	3,018.898	3
232	3.863	1.337	3.424	712.566	10,460.500	3,054.612	4
293	4.477	1.732	3.508	776.465	10,716.000	3,054.612	4
301	3.140	1.326	2.730	573.630	8,339.500	3,054.612	3
379	3.906	1.288	1.653	495.245	4,997.000	3,023.373	2
519	5.469	1.765	3.577	867.980	10,963.000	3,065.086	4
529	4.878	1.400	2.633	703.394	8,071.500	3,065.086	3
547	1.812	1.071	0.790	234.894	2,422.500	3,065.086	1
554	5.714	1.513	2.265	706.080	6,943.500	3,065.086	3
618	9.104	1.306	4.590	1,271.751	14,137.500	3,080.246	5
624	1.933	1.065	1.039	278.877	3,201.000	3,080.246	1
625	3.549	1.156	1.418	441.387	4,368.000	3,080.246	2
627	5.209	1.367	3.803	875.661	11,715.000	3,080.246	3
632	5.753	1.463	2.994	816.500	9,222.000	3,080.246	3
687	4.589	1.454	3.863	831.068	11,977.000	3,100.173	4
688	10.848	1.972	6.004	1,592.906	18,613.000	3,100.173	6
689	3.566	1.426	2.639	605.470	8,181.000	3,100.173	3
694	3.636	1.415	2.089	543.973	6,475.500	3,100.173	2
711	6.610	1.626	3.150	900.583	9,764.000	3,100.173	3

761	4.147	1.330	2.375	619.387	7,362.000	3,100.173	3
767	5.583	1.589	3.478	869.696	10,781.000	3,100.173	3
770	5.136	1.467	1.978	629.127	6,133.000	3,100.173	3
787	5.635	1.698	3.884	923.318	12,040.000	3,100.173	5
837	6.186	1.493	3.485	930.188	11,130.500	3,193.718	4
839	5.178	1.657	2.946	782.448	9,409.500	3,193.718	3
840	2.777	1.214	2.467	524.333	7,879.000	3,193.718	2
996	2.915	1.386	3.159	608.541	10,110.500	3,200.289	4
1000	6.227	1.466	2.928	856.222	9,369.500	3,200.289	3
1025	5.071	1.514	3.287	818.666	10,518.000	3,200.289	4
1038	6.145	1.531	2.907	847.553	9,302.000	3,200.289	3

Using the effective features found above for training the model through an effective algorithm, one can address the problem of multiple-fish blobs and determine how many fish are in each blob to add up to the total count.

C1. Addressing the Composite Problem by Mathematical Approach and ML

To offer a solution for solving the problem of having shared blobs containing multiple fish between two consecutive frames, we offer a combination of the mathematical approach offered in III-A1 and the machine learning approach offered in III-B; We first consider counting the number of fish in the part of a shared blob in the first image and part of the blob in the second image and then we use the following equation:

$$Number\ of\ shared\ \ fish = m_1 + m_2 - \min\left(m_1, m_2\right)$$

where m_1 and m_2 are the number of fish in the first part of the shared blob and second part of the shared blob consecutively, counted by machine learning approach.

C2. Addressing the Composite Problem through Merging and ML

An alternative method to solve the composite problem is to combine the merging and machine learning (introduced in III-B) solutions. To perform that we use the merging as introduced in III-A2 and then when we have separate frames without sharing any fish images, we use machine learning to count the number of fish in each one.

D. Addressing the Whole Process of Counting

In summary, we can propose two methods that can be used to count the fish continuously in the stream of water, the first approach which is the combination of mathematical approach and machine learning and the second one is combination of merging approach and machine learning. Both of these approaches will introduce better accuracy that the methods offered by others.

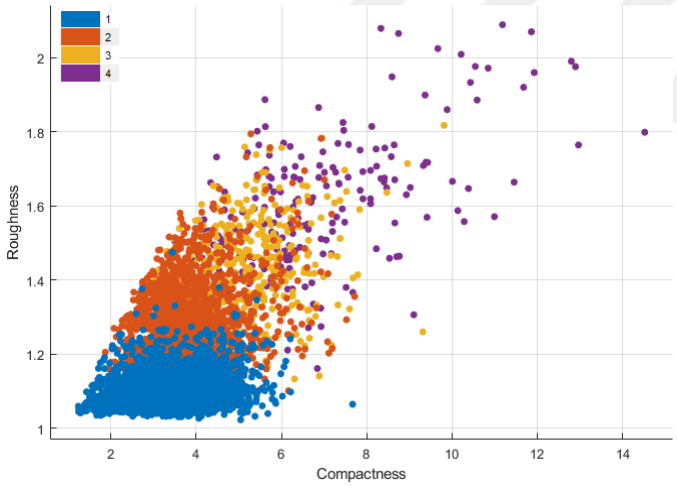
3.4 Numerical Results and Discussion

To show the effect of using different classification and training methods, the results of testing the datasets using K-fold cross validation to train and test the dataset through different classification methods of support vector machine (SVM), decision tree, ensemble, K-nearest neighbor (KNN), random forest, logistic regression and multilayer perceptron in Table 4. It is seen that for our data set, the SVM method introduces the most accurate results and thus in the ultimate version of our software we have implemented the model trained by this method for counting the fish.

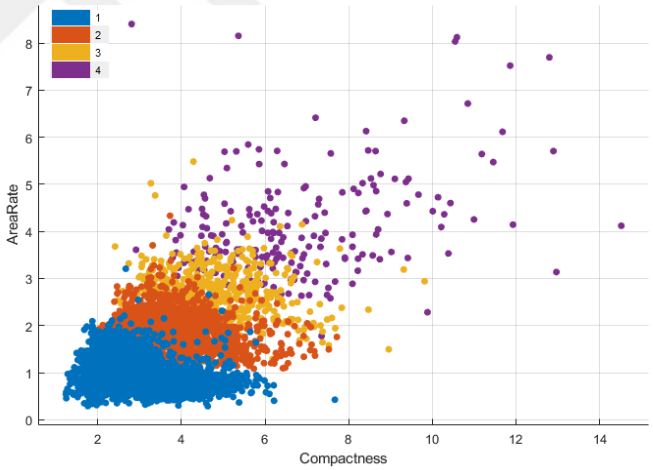
Table 4: Comparison of accuracy of different classification methods in dataset

Machine Learning Algorithm	SVM	KNN	Ensemble	Logistic Regression	Decision Tree	Random Forest	Multilayer Perceptron
Accuracy	99.1 %	98.9 %	98.5 %	98.37 %	98.77 %	98.91 %	98.94 %

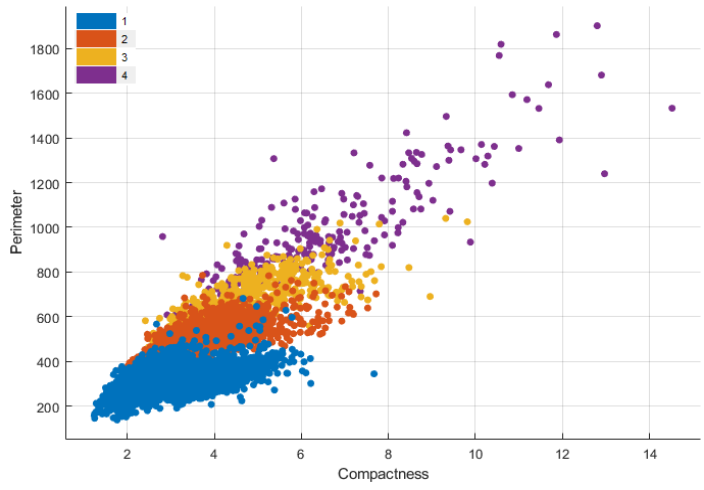
To show the reason behind using the features introduced previously, here we bring the results of classifying for some data set into classes of "1", "2", "3", and "4 and more", using K-fold cross validation by SVM in Figure 51. Here we are comparing the three features that we selected previously with the perimeter feature which is used in some of the previous works. The results show that the correlation of the other three features with each other are helping classifying the data in a better manner.



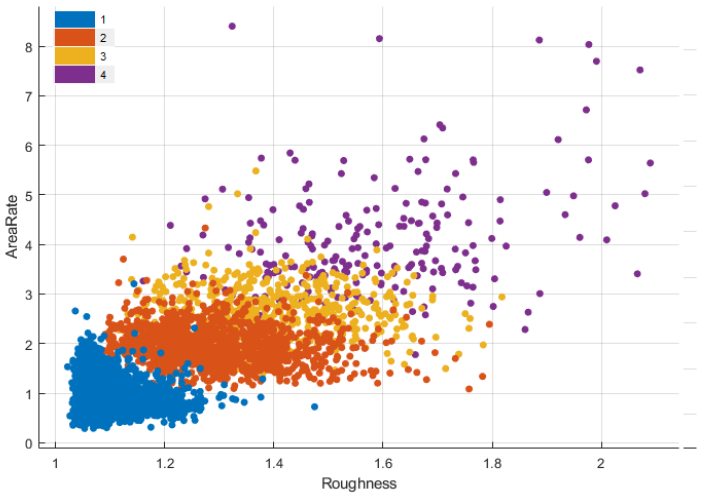
Compactness and Roughness



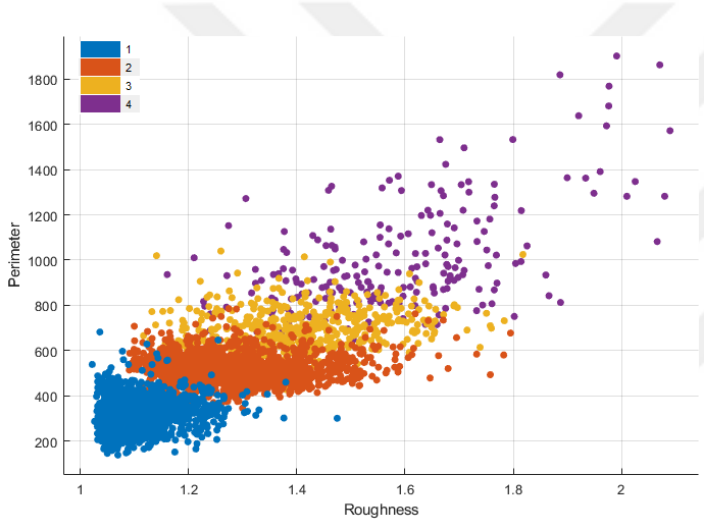
Compactness and Area-Rate



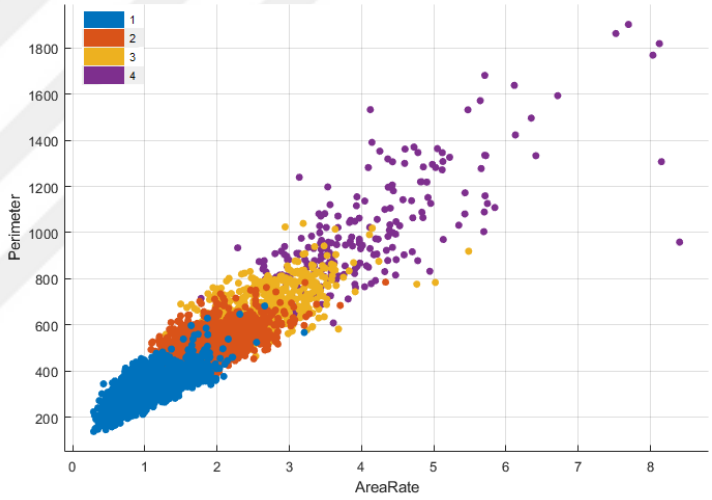
Compactness and Perimeter



Roughness and Area-Rate



Roughness and Perimeter



Area-Rate and Perimeter

Figure 51: Difference between the scatter plots for different combination of features

Table 5 shows the comparison between the two proposed approaches in III-D. It is seen that the second approach (i.e. combination of merging and machine learning) has a better accuracy rate and introduces the accuracy of 99.44%, while the other approach (i.e. combination of mathematical approach and machine learning) is also giving us a high accuracy rate which is much better than the previous works. As a result of this, we have

chosen to integrate the second approach into our hardware for the final version of the fish counting machine.

Table 5: Comparison between our two proposed novel approaches for fish counting

		Set1	Set2	Set3	Set4	average
	Real Fish Numbers	490	473	465	24300	
Approach 1	Counts	486	471	462	25457	
	Difference	4	2	3	1157	
	Error Rate	0.816326531	0.422833	0.645161	4.761317	
	Accuracy	99.18367347	99.57717	99.35484	95.23868	98.33859
Approach 2	Counts	489	470	463	24060	
	Difference	1	3	2	240	
	Error Rate	0.204081633	0.634249	0.430108	0.987654	
	Accuracy	99.79591837	99.36575	99.56989	99.01235	99.43598

Eventually, Table 6 is presenting real, in-field results of testing our fish counting machine. The tests have been conducted in some fish farms in Turkey and the results of counting are compared carefully with the precise results of precise manual counting. As the table shows, our counting machine is providing even better results than the ones from Table 4 and Table 5, as the test samples chosen for the training phase were considered to be chosen out of the worst possible cases and so the results in Table 4 and Table 5 are for the cases which may not be present all together at the same time in a real course of water stream conducted to the fish counting machine, resulting in our machine to be more reliable and ready for the worst possible scenarios. Here, also the counting time is included to show that whatever the water flow through the fish counting machine is, it does not affect the precision of counting by our machine.

Table 6: Real in-field results of the fish counting machine

Row	Manually Counted	Machine Counted	Error	Error Percentage	Counting Time
1	29004	28658	-346	-1.19%	13 min.
2	13633	13676	43	0.32%	6 min.
4	20700	20482	-218	-1.05%	10 min.
5	42637	42303	-334	-0.78%	13 min.
6	62564	61730	-834	-1.33%	17 min.
7	59455	58890	-565	-0.95%	16 min.
8	62564	61672	-892	-1.43%	16 min.
9	59455	58941	-514	-0.86%	14 min.
10	185338	177296	-8042	-4.34%	25 min.
11	26487	26486	-1	0.00%	13 min.
12	26487	26310	-177	-0.67%	10 min.
13	26487	26520	33	0.12%	12 min.
14	26487	26368	-119	-0.45%	9 min.
15	26487	26376	-111	-0.42%	9 min.

CHAPTER IV

CONCLUSION AND FUTURE WORK

In this thesis, we proposed some novel approaches for counting the fish in fish farms. Each of the proposed approaches offers a higher accuracy for the job of fish counting than the previous works, while due to higher accuracy, one of the approaches is even more ideal for the job than the other one. The fish counting machine is designed based on a combination of image processing techniques and machine learning methods (via choosing the best-suited morphological features for the fish counting job). This results in finding the best approach of classification, training, and counting the fish. Using our novel approach allows performing fish counting in real-time and in a continuous manner with the lowest amount of processing time and highest accuracy.

In this work, in chapter one, we have explored the previous works on this issue. These works include attempts to do the fish counting job through mechanical, electrical, and acoustic methods in the older works. Recent studies have explored the possibility of using machine learning algorithms and computer vision methods for this job. In this chapter, we have also studied some previous works which are not focusing on counting the fish, but due to their relevance to the counting job and the methods they have used, they worth being mentioned to have a more complete literature review. These works can also be inspiring for the ones who are exploring the possibility of using counting methods for the sake of counting other things than the ones intended in the corresponding original work.

In chapter two, we have brought some fundamental concepts which are used in further chapters. We have gone through machine learning algorithms and computer vision methods. We have introduced the conceptual aspects of morphological features and how they can be used in training the mentioned machine learning algorithms.

In chapter three, the main problem of this work is introduced. The problem is explored by mentioning the features that we want in the defined counting job and the challenges ahead of this work and the reasons the previous works have not been able to address such problems. Different issues on the way of accomplishing the goal are introduced and studied in detail. In the same chapter, we brought our proposed methods to address the studied problems and the additional issues that each of the methods may bear into the fish counting job have been explored. In this chapter, at the end, an ultimate method is proposed which combines the best aspects of each of the methods and addresses all the problems. The accuracy and advantages of the proposed methods have been proven through extended numerical results. The results demonstrate the superiority of our method against the previous works.

Although due to the high accuracy of 99.44% in the fish counting and also the advantages that our method offers to the job, this study seems to have addressed the best solutions and have come up with an ultimate solution for the real-time fast fish counting, but as in the world of science and technology, one can always improve the previous works, the author does not hesitate to say that this study can be even further improved, as the machine learning techniques and computer vision methods are advancing every day. Therefore, with the improvement in the computer science world, newer procedures can be explored to be used for this job and result in even higher accuracy rates and higher speeds with the use of lower processing power.

BIBLIOGRAPHY

[1] N. J. C. Strachan and F. J. Nicholson, “A novel fish counter for fish farms,” *Aquac. Eng.*, vol. 9, pp. 405–410, 1990.

[2] J. J. Sheppard and M. S. Bednarski, “Utility of single-channel electronic resistivity counters for monitoring river herring populations,” *North Am. J. Fish. Manag.*, vol. 35, pp. 1144–1151, 2015.

[3] H. Balk and T. Lindem, “Fish detection in rivers with split-beam sonars,” *Proc. Scand. Symp. Physics Acoustic.*, 2002.

[4] D. Li, Y. Hao, and Y. Duan, “Nonintrusive methods for biomass estimation in aquaculture with emphasis on fish: a review,” *Rev. Aquac.*, pp. 1–22, 2019.

[5] P. Roux and J. De Rosny, “Multiple scattering in a reflecting cavity: Application to fish counting in a tank,” *J. Acoust. Soc. Am.*, vol. 109, pp. 2431–2431, 2001.

[6] P. Jonsson, I. Sillitoe, B. Dushaw, J. Nystuen, and J. Heltne, “Observing using sound and light – a short review of underwater acoustic and video-based methods,” *Ocean Sci. Discuss.*, vol. 6, pp. 819–870, 2009.

[7] G. Rakowitz, M. Tušer, M. Říha, T. Jůza, H. Balk, and J. Kubečka, “Use of high-frequency imaging sonar (DIDSON) to observe fish behaviour towards a surface trawl,” *Fish. Res.*, vol. 123–124, pp. 37–48, 2012.

[8] W. Z. Shen, C. L. Zhang, and Z. L. Chen, “Research on automatic counting soybean leaf aphids system based on computer vision technology,” *Proc. Int. Conf. Machine Learning and Cybernetics*, pp. 1635–1638, 2007.

[9] D. B. Yang, H. H. González-Baños, and L. J. Guibas, “Counting people in crowds with a real-time network of simple image sensors,” *Proc. IEEE Int. Conf. Comput. Vis.*, pp. 122–129, 2003.

[10] S. Meany, E. Eskew, R. Martinez-Castro, and S. Jang, “Automated vehicle counting using image processing and machine learning,” *Proc. Heal. Monit. Struct. Biol. Syst.*, vol. 10170, p. 101703G1–7, 2017.

[11] A. Flores, P. Crisóstomo, and J. López, “Peruvian scallop larvae counting system using image processing techniques,” *Proc. Int. Caribb. Conf. Devices, Circuits Syst. (ICCDCS)*, pp. 8–11, 2008.

[12] G. Somasundaram, V. Morellas, and N. Papanikolopoulos, “Counting pedestrians and bicycles in traffic scenes,” *Proc. IEEE Conf. Intell. Transp. Syst. (ITSC)*, pp. 299–304, 2009.

[13] E. F. Morais, M. F. M. Campos, F. L. C. Pádua, and R. L. Carceroni, “Particle filter-based predictive tracking for robust fish counting,” *Proc. Brazilian Symp. Comput. Graph. Image Process.*, pp. 367–374, 2005.

[14] J. M. Hernández-Ontiveros et al., “Development and implementation of a fish counter by using an embedded system,” *Comput. Electron. Agric.*, vol. 145, pp.

53–62, 2018.

[15] J. Le and L. Xu, “An Automated Fish Counting Algorithm in Aquaculture Based on Image Processing,” *Proc. Int. Forum on Mech., Cont. and Autom.*, pp. 358–366, 2017.

[16] C. Spampinato, Y. H. Chen-Burger, G. Nadarajan, and R. B. Fisher, “Detecting, tracking and counting fish in low quality unconstrained underwater videos,” *Proc. Int. Conf. Comput. Vis. Theory Appl. (VISAPP), vol. 2*, pp. 514–519, 2008.

[17] X. Liu, J. Van De Weijer, and A. D. Bagdanov, “Leveraging Unlabeled Data for Crowd Counting by Learning to Rank,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 7661–7669, 2018.

[18] R. Gonzalez and R. Woods, *Digital image processing*, Addison-Wesley, 1993.

[19] Y. H. Toh, T. M. Ng, and B. K. Liew, “Automated fish counting using image processing,” *Proc. Int. Conf. on Computational Intelligence and Software Engineering (CiSE)*, pp. 1–5, 2009.

[20] L. Fan and Y. Liu, “Automate fry counting using computer vision and multi-class least squares support vector machine,” *Aquaculture*, vol. 380–383, pp. 91–98, 2013.

[21] N. Efford. *Digital Image Processing: A Practical Introduction Using Java. Pearson Education*, Addison-Wesley, 2000.

[22] S. Lemm, B. Blankertz, T. Dickhaus, and K. R. Müller, “Introduction to machine learning for brain imaging,” *Neuroimage*, vol. 56, pp. 387–399, 2011.

[23] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: From theory to algorithms*, Cambridge University Press, 2014.

VITA

Ahad Aghapour received his B.Sc. degree in software engineering from Applied Science University of Jahad, Tabriz, Iran in 2010. Throughout his undergraduate studies, he was a freelance programmer and participated in many projects as a developer. Afterward, he worked at ASR-Gooyesh-Pardaz company as a senior programmer for around 6 years in Tehran, Iran. Later on, he pursued his master's degree in the field of image processing and computer vision under the supervision of Professor H. Fatih Uğurdağ at Özyeğin University, Istanbul, Turkey. His research interests include machine/deep learning, computer vision, and image processing. During his studies at Özyeğin, he has submitted a journal paper and a patent of fish counting machine in Turkey.