# Incorporating Recognition in Catfish Counting Algorithm Using Artificial Neural Network and Geometry

**Ibrahim Aliyu[1], Kolo Jonathan Gana[2], Aibinu Abiodun Musa[3], Mutiu Adesina Adegboye[4], and Chang Gyoon Lim[1*]**

[1] Department of Computer Engineering, Chonnam National University
50 Daehakro, Yeosu, Jeonnam, South Korea
[e-mail: ibal2010@yahoo.com, cglim@jnu.ac.kr]
[2] Department of Electrical and Electronics Engineering, Federal University of Technology, Minna
Niger State, Nigeria
[e-mail: jgkolo@futminna.edu.ng]
[3] Department of Mechatronics Engineering, Federal University of Technology, Minna
Niger State, Nigeria
[e-mail: maibinu@gmail.com]
[4] Department of Computer Engineering, Federal University Oye-Ekiti, Ekiti State, Nigeria
[e-mail: mutiu.adegboye@fuoye.edu.ng]
*Corresponding author: Chang Gyoon Lim

*Received July 13, 2020; revised October 5, 2020; accepted December 7, 2020; published December 31, 2020*

## Abstract

One major and time-consuming task in fish production is obtaining an accurate estimate of the number of fish produced. In most Nigerian farms, fish counting is performed manually. Digital image processing (DIP) is an inexpensive solution, but its accuracy is affected by noise, overlapping fish, and interfering objects. This study developed a catfish recognition and counting algorithm that introduces detection before counting and consists of six steps: image acquisition, pre-processing, segmentation, feature extraction, recognition, and counting. Images were acquired and pre-processed. The segmentation was performed by applying three methods: image binarization using Otsu thresholding, morphological operations using fill hole, dilation, and opening operations, and boundary segmentation using edge detection. The boundary features were extracted using a chain code algorithm and Fourier descriptors (CH-FD), which were used to train an artificial neural network (ANN) to perform the recognition. The new counting approach, based on the geometry of the fish, was applied to determine the number of fish and was found to be suitable for counting fish of any size and handling overlap. The accuracies of the segmentation algorithm, boundary pixel and Fourier descriptors (BD-FD), and the proposed CH-FD method were 90.34%, 96.6%, and 100% respectively. The proposed counting algorithm demonstrated 100% accuracy.

***Keywords:*** Aquaculture, Catfish, Counting Algorithm, Digital Image Processing, ANN

**http://doi.org/10.3837/tiis.2020.12.014** **ISSN : 1976-7277**

## 1. Introduction

Fish products contribute a significant amount of protein to human nutrition- about 16% of the human diet worldwide. Also, its consumption has dramatically increased from approximately 36.82 million tons circa 1960 to 199.74 million tons by 2015 [1]. Likewise, the demand for aquaculture software, supplies, and equipment has increased-it was estimated to exceed US$ 80 billion in 2016 [1-3]. In Nigeria, catfish production has increased from 3 million tons per annum in 2001 to more than 30 million tons per annum in 2006 [4]. But Nigeria still imports approximately 900 000 tons of fish each year which costs about US$ 1 billion due to shortages in the production sector [5-7]. One of the problems faced by the production sector in the country is accurate counting as it is essential for accurate stocking of ponds, feeding planning, marketing and fast delivery of fish [8, 9].

Most catfish counting in rural areas of Nigeria is performed manually, which involves counting and sorting the fish by hand [10]. However, the manual counting is prone to mistakes, occasional omissions, fatigue and may lead to stress and death of the fish. Another method of fish counting is the use of a container to estimate the number of catfish, which is also inaccurate [10]. Inaccurate estimation affects both the hatchery and the customer because it can lead to over- or under-feeding or payment. Even though several automatic counting systems have been devised, the cost of acquiring such commercial counting equipment is very high: US$ 75,000 and greater [3].

Various techniques have been proposed for fish counting such as mechanical, optical and digital image techniques (DIP). The techniques associated with DIP include pixels counting, video tracking, machine vision, etc. However, machine vision is mainly suitable for fishing or underwater counting [11]. Video tracking technique also increases the cost of counting, slows counting speed and is computationally intensive.

Despite some of the disadvantages of Fish counting using DIP, it is still a viable solution in catfish farms as it would reduce counting time, minimize fish stress, and facilitated proper financial planning [12]. Thus, to harness the benefit of DIP, one of the major issues to improve in fish counting is accuracy, which is affected by factors such as noise, overlapped and other objects other than fish in the pond [13, 14]. As the DIP basically employed pixel counting, there is a problem of object recognition to substantiate which area pixel is associated with fish as noise or variation in illumination or any other object in the image could erroneously be considered as fish. Besides, the flexibility of counting fish of any size is critical in attaining high accuracy in DIP.

Hence, this study is aimed at developing a catfish recognition and counting algorithm to improve pixel counting by recognizing the catfish before counting. It also addresses the issue of overlapping by formulating a new counting algorithm that is flexible to count fish of any size. Therefore, the main contributions of this paper that differ from other works is in at least one of the following points:

- The development of Catfish pre-processing, segmentation, and feature extraction techniques.
- Introduction of chaincode to improve catfish recognition accuracy using shallow neural networks with few trainable parameters.
- Formulation of new counting algorithm that is flexible to count fish of any size while addressing the problem of overlapping

The remainder of the paper is organized as follows: In Section 2, an overview of related works is presented. Section 3 provides the study methodology, Section 4 presents the results and discussion, and Section 5 provides the study conclusions.

## 2. Related Works

Newbury, Culverhouse, and Pilgrim used the top view of an artificial fish image to estimate fish count using neural networks [15]. High accuracy was achieved, but artificial fish were used and there was no background noise and no movement. Yada and Chen used a weighing technique to count seedling fry, but it was too slow, only a few milligrams were accurately weighed, and required a controlled environment [16]. Friedland et al. proposed a system that counted fish eggs using commercial software in which a dilution-erosion algorithm was employed to segment acquired images [17]. This work is particularly important because it attempted egg counting, which has issues with overlap as in catfish counting. Alver, Tennøy, Alfredsen, and Øie developed an autonomous rotifer sampling and counting system. However, accuracy figures were not presented [18]. A fully automated acoustic method for counting and sizing fish on fish farms using Dual-Frequency Identification Sonar (DIDSON) was proposed by Han, Honda, Asada, and Shibata [19]. The advantage of the proposed system is its ability to be applied in fish farms for counting and sizing, whereas DIDSON was initially designed for counting fish underwater. Toh et al. used blob detection to analyze the number of fish in a video by summing the fish in each frame and taking the average [20]. However, only five fish in overlap could be handled by this approach. Zheng and Zhang proposed a new method of counting fish using a digital image processing and a fuzzy artificial neural network (ANN) [21]. The method was capable of recognizing overlapped and differently sized fish.Loh, Raman, and Then developed Aquatic Tool Kits (ATK) for estimating the number of larval and juvenile fish using pixel counting [2]. However, the proposed method might not be flexible to count Fish of various size as the average area of a single larva is used to obtain the count

In another study, Labuguen et al. proposed an automated system using edge detection for fish fry counting and behavior analysis [13]. The method demonstrated an average accuracy of 95%. Luo, Li, Wang, Li, and Sun proposed a system for fish recognition and counting using a machine learning technique and statistical methods based on a set of six rules [22]. But a low accuracy of approximately 75% was recorded. Duan et al. proposed a non-invasive method of counting the live pelagic eggs of marine fish using an image analysis method [23]. The technique can be tried in ponds where contamination changes the watercolor. Raman et al. proposed a low-cost machine learning technique for efficient fish counting and growth studies using ANN [3]. However, the authors only stated that the boundary detection algorithm would be applied to count through the connected regions for overlaps case but there was no clear or robust explanation on how this would be performed.

More recently, Lumauag and Nava employed blob analysis and Euclidean filtering in DIP technique to estimate fish population [24]. The results of their experiment recorded an average counting and detection accuracy of 91% and 94%, respectively. Besides, Kaewchote et al. proposed an image recognition technique that utilized local binary pattern and RGB color feature extraction with an ensemble tree and random forest as classifiers for larvae counting [25]. However, a large root mean square error (MSE) of 14.43 was recorded mainly due to overlapping of shrimps. The study Lainez and Gonzales [26] proposed an automated fingerling counting algorithm using image processing method and convolution neural network (CNN). However, the CNN architecture requires a large trainable parameter. Garcial et al. [8] proposed an automatic fingerling counting algorithm which recorded an average error of 2.65% when

compared with the manual counting. Meanwhile, a compressive review on image processing techniques for fisheries application was conducted by Awalludin et al. [9]. The authors observed that research publications with the application of image processing methods are still very limited.

Based on the reviewed literatures, issues such as counting other objects present in the water or counting noise caused by variations in illumination and flexible counting of fish of any size with high accuracy need to be addressed. Hence, this study aims to develop a catfish recognition and counting algorithm to improve pixel counting by recognizing the catfish before counting and the formulation of a new strategy for addressing overlapping in counting.

## 3. Catfish Counting Algorithm

The study methodology is comprised of five steps: image acquisition, image pre-processing, image segmentation, feature extraction, classification, and counting, as well as an evaluation of the accuracy and mean square error of the results. **Fig. 1** shows the block diagram of the methodology in which an image is acquired, pre-processed, and segmented, features are then extracted for recognition and counting. Details of each unit in the block diagram are provided in the subsequent subsections.

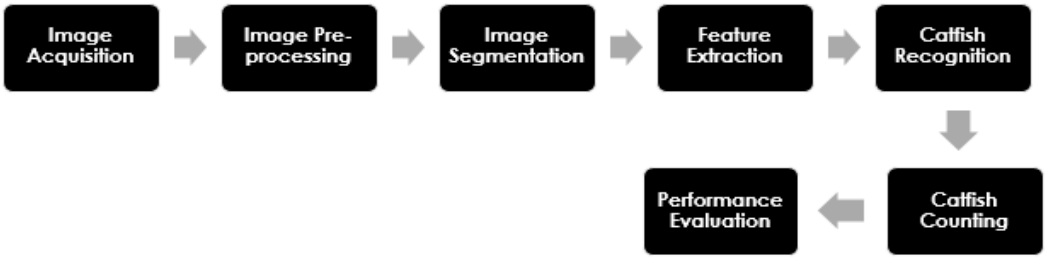



**Fig. 1.** The proposed methodology for catfish counting

### 3.1 Image Acquisition

The image acquisition unit consisted of a camera that captured multiple images of a group of catfish in a container at an interval of approximately 3 s. The multiple image capture improved accuracy and reduced the issue of overlapping that occurred because of the catfish moving around in the water. The 8-megapixel camera was positioned vertically above the fish container.

### 3.2 Fish Image Pre-processing

Water contamination due to feed and other factors is a common problem in a fish pond. A pre-processing algorithm was implemented to successfully achieve a higher accuracy of counting

in such an aquatic background, as shown in **Fig. 2**.

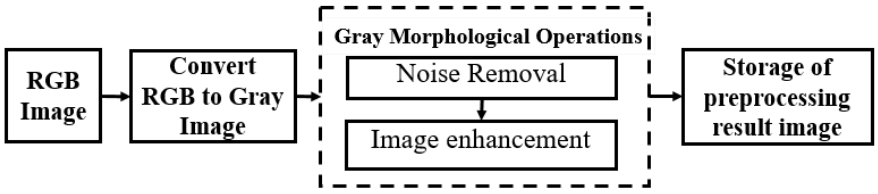



**Fig. 2.** Image pre-processing algorithm for noise removal and color conversion consisting of two operations: color conversion and gray morphological operation

After capture, the image was pre-processed to filter the fish and remove the noise from the image. The image was first converted from an RGB to a grayscale image using Equation (1).

$$I\ (x,y) = 0.2989 \times R + 0.5870 \times G + 0.1140 \times B \tag{1}$$

where R, G, and B are the red, green, and blue, respectively, colors of the RGB color space. Gray morphological operations were then performed using contrast enhacement and a media filer to enchance and remove noise from the image. Contrast enhancement mapped the intensity values in the grayscale level image, $I(x,y)$, to new values. The resulting values were saturated at low and high intensities of 1% of the data, which improved the contrast of the resulting image. The median filtering operation ranked neighboring pixels according to brightness (intensity), and the middle (median) value became the new value for the pixel under consideration. This filtering smoothed the noisy regions and preserved the boundary structures between the regions. The median was obtained using Equation (2):

$$\mathrm{I}_m(x,y) = \underset{(x,y)\epsilon S_{xy}}{\mathrm{median}}\{I(x,y)\} \tag{2}$$

where $I(x,y)$ is the noisy gray image, and $S_{xy}$ is a sub-image (region) of the image, $I(x,y)$. The image, $I_m(x,y)$, was then stored for the segmentation process.

### 3.3 Image Segmentation Algorithm

In the segmentation algorithm, the image underwent three operations: thresholding, morphological operations, and edge segmentation. The process is illustrated in **Fig. 3**. The key operations in the algorithm are described as follows.

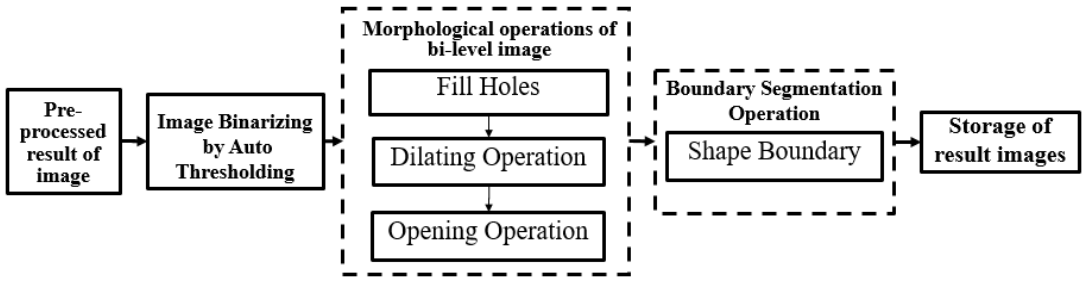



**Fig. 3.** Image segmentation algorithm key operations: auto-thresholding, morphological operations, and edge detection operation

### 3.3.1 Auto-thresholding

Image binarization was performed using Otsu's thresholding to correct the variations in the mean gray level that may arise due to factors such as unequal light exposure. Otsu's thresholding method was employed to binarize the image, $I_m(x, y)$, into the foreground (catfish with pixels = 1) and background (= 0) by iterating through all the possible threshold values and calculating an optimal threshold to separate the two classes so that their intra-class variance was minimal while their inter-class variance was maximal. The exhaustive search for the threshold is the sum of the weighted variance of the two classes as obtained by Equation (3):

$$\sigma^2_{I_m(x,y)}(t) = \omega_0(t)\sigma_0^2(t) + \omega_1(t)\sigma_1^2(t) \tag{3}$$

where $\omega_0$ and $\omega_1$ are the probabilities of the two classes, background and catfish, respectively, separated by a threshold, $t$, and the variances of these two classes are $\sigma_0^2$ and $\sigma_1^2$. The probabilities $\omega_0(t)$ and $\omega_1(t)$ were obtained using the histograms expressed by Equations (4) and (5), respectively:

$$\omega_0(t) = \sum_{i=0}^{t-1} p(i) \tag{4}$$

$$\omega_1(t) = \sum_{i=t}^{L-1} p(i) \tag{5}$$

where L is the grayscale layers of the image, $I_m(x, y)$. The resulting image, $I_t(x, y)$, was stored for the next stage and is given as follows:

$$I_t(x, y) = \sigma^2_{I_m(x,y)}\,. \tag{6}$$

### 3.3.2 Morphological operations

After the initial binarization by thresholding, the fish objects' holes and some minor noise still existed, and portions of the catfish were truncated. To correct these issues, firstly, a fill hole operation was executed using Equation (7):

$$I_H(x, y) = \left[R_{I_t(x,y)}(F)\right]^t \tag{7}$$

where $R_{I_t(x,y)}$ is the reconstruction of the input image (mask), $I_t(x, y)$, from $F$, and $F$ is the marker given as Equation (8):

$$F\,(x, y) = \begin{cases} 1 - \mathrm{I}_t(x.y), if\ (x,y)\ is\ on\ the\ border\ of\ \mathrm{I}_t(x,y) \\ 0, otherwise \end{cases} \tag{8}$$

An area opening operation was subsequently performed to remove small objects. In the opening operation, $S$ was eroded by $I_H(x, y)$, followed by a dilation of the result by $S$. The opening operation is given as Equation (9):

$$I_H(x,y) \circ S = (I_H(x,y) \ominus S) \oplus S \tag{9}$$

where $S$ is the structuring element determined experimentally. In other words, the opening operation can be express as Equation (10):

$$I_H(x,y) \circ S = u\{(S)_z \subseteq I_H(x,y)\} \tag{10}$$

where $u$ is the union of all sets. The erosion was executed to "shrink" or "thin" objects in the image to diminish the false connections that appear as overlap. The erosion of $I_H(x,y)$ by $S$ is given as Equation (11):

$$I_H(x,y) \ominus S = \{(S)_z \subseteq I_H(x,y)\} \tag{11}$$

where $S \subseteq I_H(x,y)$ indicates that $S$ is a subset of $I_H(x,y)$. Equation (11) indicates that the erosion of $I_H(x,y)$ by $S$ is the set of all points $z$ such that $S$, translated by $z$, is contained in $I_H(x,y)$. Consequently, the erosion can be expressed as follows:

$$I_H(x,y) \ominus S = \{(S)_z \; n \; {I_H}^c = \emptyset\} \tag{12}$$

Here, erosion is the set of all element origin locations where no part of $S$ overlaps the background, $I_H(x,y)$. The dilation in the opening operation "grows" and "thickens" objects in an image. The extent of the thickening is controlled by a shape referred to as a structuring element. The dilation is given as follows:

$$I_H(x,y) \oplus S = \left\{\left(\hat{S}\right)_z \; n \; I_H \neq \emptyset\right\} \tag{13}$$

where $\emptyset$ is the empty set and $\hat{S}$ is the structuring element. Therefore, the resultant image, $I_o(x,y)$, is given as Equation (14):

$$I_o(x,y) = x \circ y \quad . \tag{14}$$

### 3.3.3 Edge detection operation

After the morphological operations, the shape boundary was obtained using edge detection. Based on the following criteria, the edge detection operation searched for places in the images where the intensity changed rapidly:

1. It found the places where the first derivative of the intensity was greater in magnitude than a specified threshold.

2. Then, it found the places where the second derivative of the intensity had a zero crossing. The first-order derivative is the gradient, $(g)$, of the 2D function $I_O(x,y)$, defined as follows:

$$v_{I_o(x,y)} = \left(\frac{g_x}{g_y}\right) = \left(\frac{\frac{\partial f}{\partial x}}{\frac{\partial f}{\partial y}}\right). \tag{15}$$

The magnitude is given as follows:

$$\nabla I_O = mag(\nabla I_O) = ({g_x}^2 + {g_y}^2)^{\frac{1}{2}}, \tag{16}$$

and the angle at the maximum rate of change is

$$\alpha(x,y) = \left(\frac{g_y}{g_x}\right). \tag{17}$$

The second-order derivatives were computed using the Laplacian as shown below:

$$\nabla^2 I_O(x,y) = \frac{\partial^2 f I_O(x,y)}{\partial x^2} + \frac{\partial^2 I_O(x,y)}{\partial y^2}. \tag{18}$$

The edge detected was then traced using a green outline, as presented in Section 4.2.

### 3.4 Feature Extraction Algorithm

After segmentation, the feature extraction algorithm was executed to obtain the shape descriptors of the fish. The algorithm extracted the shape features using a chain code representation and Fourier descriptors (CH-FD). The details of the CH-FD process are discussed in the subsequent section. **Fig. 4** illustrates the feature extraction algorithm.

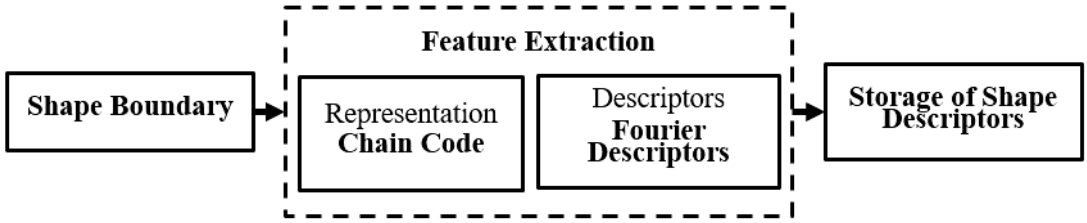



**Fig. 4.** Feature extraction algorithm using a chain code representation and Fourier descriptors (FDs)

The shape boundary pixel was obtained from the segmented catfish. The boundary pixels were then used to derive the chain code representation of the shape. The chain code was then processed using a discrete Fourier transform (DFT) implemented in the form of a fast Fourier transform (FFT). The results of these processes were termed as the Fourier descriptors (FDs).

#### 3.4.1 Chain Code Representation

The chain code boundary representation was based on eight connectivity segments. First, the boundary pixels were designated as segmented, as discussed in Section 3.3. The chain code of the boundary pixels was then generated. **Fig. 5** shows the trace boundary of a catfish on a grid, from which the chain code was obtained.

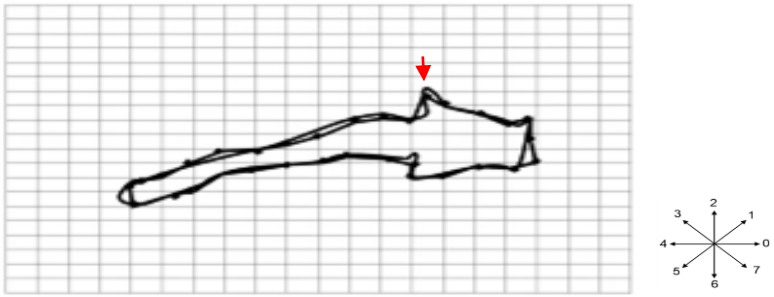



**Fig. 5.** Catfish chain code representation

In this study, the new chain code generation function was developed based on a straight-line gradient. Considering the figure on a line graph, the difference between two points (pixel boundary) on the y-axis ($\Delta y$) was given by

$$\Delta y = y_2 - y_1 , \tag{19}$$

and the difference between two points on the x-axis (pixel boundary) was given as

$$\Delta x = x_2 - x_1 . \tag{20}$$

The gradient ($\delta$) of any line segment in **Fig. 5**, can, therefore, be expressed as

$$\delta = \frac{\Delta y}{\Delta x} . \tag{21}$$

Using (19)– (21), the chain code ($\eta$) can be expressed mathematically as (22)

$$= \begin{cases} 0, & \delta == 0 \; and \; \Delta y > 0 \; and \; \Delta x == 0 \\ 1, & \delta > 0 \; and \; \Delta x > 0 \\ 2, & \delta > 0 \; and \; \Delta y > 0 \\ 3, & \delta < 0 \; and \; \Delta y > 0 \; and \; \Delta x < 0 \\ 4, & \delta == 0 \; and \; \Delta y == 0 \; and \; \Delta x < 0 \\ 5, & \delta > 0 \; and \; \Delta y < 0 \; and \; \Delta x < 0 \\ 6, & \delta < 0 \; and \; \Delta y < 0 \; and \; \Delta x == 0 \\ 7, & \delta < 0 \; and \; \Delta y < 0 \end{cases} \tag{22}$$

We applied a similar technique for optimal fish feeding strategy [27].

### 3.4.2 Fourier Descriptors (FDs)

The boundary chain code was further processed to obtain the FDs of the boundary. The chain code obtained is denoted as $\eta$ ($k$), where K is the sequence of the point. A DFT was implemented in the form of an FFT using

$$a(u) = \sum_{k=0}^{K-1} \eta(k) e^{-\frac{j2\pi uk}{K}} \tag{23}$$

where $\eta(k)$ is the chain code and the complex coefficients, $a(u)$, are the FDs of the boundary. The value of $\eta(k)$ can be obtained using the inverse Fourier transform of these coefficients:

$$\eta(k) = \frac{1}{K} \sum_{k=0}^{K-1} a(u) e^{\frac{j2\pi uk}{K}} . \tag{24}$$

For $k = 0, 1, 2, 3..K-1$, $k$ still ranges from 0 to $K-1$, i.e., the same number of points exists in the approximate boundary, but not as many terms are used in the reconstruction of each point.

### 3.4.3 Feature Selection

The FDs provide effective shape descriptors with efficient tradeoffs between the speed of recognition, accuracy, and the compactness of the representation. Additionally, few coefficients are sufficient to describe complex objects, which can be normalized to make them invariant to translation, scale, and rotation. In this study, ten FDs were applied to both the boundary pixels (BW-FD) and boundary chain code (CH-FD). Hence, the Fourier coefficients used only the first ten coefficients to compute the inverse as per Equation (25):

$$\eta(k) = \frac{1}{10} \sum_{u=0}^{9} a(u) e^{\frac{j2\pi uk}{10}} . \tag{25}$$

The FDs were then used as a dataset to train an ANN. The dataset contained catfish and non-cat samples from which the FDs were extracted. Each sample consists of ten attributes described by the ten FDs.

## 3.5 Catfish Recognition using an ANN

The catfish recognition system was developed using a feedforward neural network (FFNN). The FFNN implemented in this study was based on a multilayer perceptron (MLP), which had the ten FDs as inputs, five hidden layers, and an output layer. The ANN parameters, i.e., the number of hidden neurons, the training cycle, and the training algorithm, were experimentally acquired to achieve optimal performance. **Fig. 6** shows the designed ANN model.

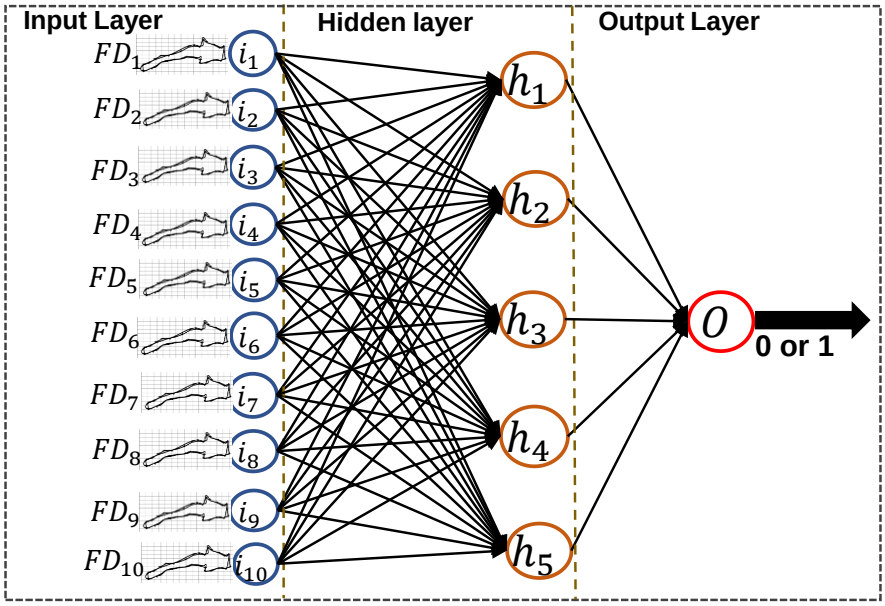



**Fig. 6.** ANN recognition model consisting of an input layer, hidden layer, and output layer

Based on the design, the number of input parameters ($k$) is ten and the number of neurons in the hidden layer ($j$) is five. Thus, the input to output structure of the model is formulated as:

$$i_j = b_1 + \sum_{i=1}^{k} w_{ij} \times FD_i \tag{26}$$

where $i_j$ denotes the weighted sum of the $j$ hidden neuron, $b$ denote the hidden neuron bias, $w_{ij}$ is the synaptic weights of the neuron connection between the input and the hidden neuron, while $FD_i$ is the input. The output of each hidden neuron is express as:

$$h_j = f(i_j) \tag{27}$$

and the output layer neuron is given by:

$$O_k = b_2 + \sum_{j=1}^{m} w_{jk} \times h_j \tag{28}$$

where $b_2$ is denote output neuron bias, $w_{jk}$ is the synaptic weights of the neuron connection between $m_{th}$ hidden neuron to the output neuron. The activation employed is tan sigmoid function, which limits the range of the output to 1 or 0, where 1 is the activator for catfish, and 0 is for non-catfish. The mathematical model of the function is given as:

$$f(O_k) = \frac{2}{1 + e^{-2O_k}} - 1 \tag{29}$$

where $y$ is the input and $f(O_k)$ is the output.

### 3.6 Catfish Counting Algorithm using Geometry

This study developed a new counting approach that is based on the geometry of the fish. This approach was used to estimate the possible number of fish in a cluster of connected or overlapped fish. **Fig. 7** shows the counting algorithm.

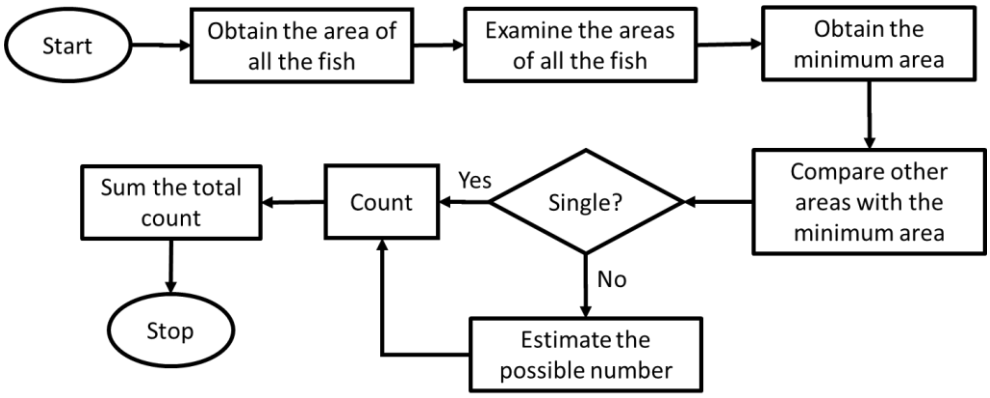



**Fig. 7.** Fish counting algorithm flowchart

The first step in the counting method was to determine all of the areas of fish in the image, including the overlapped fish, by singly extracting them from the background. These obtained areas were used to form an array. The minimum area of the array was then estimated and used to determine the proportion by which each of the areas differed from the minimum area. Since the fish were of homogenous sizes, the difference in the area was assumed to be not more than 1.5 times the minimum area. Any area larger than 1.5 times the minimum area was considered an overlap. Further estimates were then performed on the overlapped fish to determine their possible number. Mathematically, this method can be expressed as follows:

The ***number of fish*** ($\lambda$) in an area is given by

$$\lambda \begin{cases} 1 & if\ a < 1.5(\min[A]) \\ 2 & if\ 1.5(\min[A]) < a < 2.5(\min[A]) \\ \vdots & \vdots \\ \vdots & \vdots \\ k & if\ (k-0.5)\min[A] < a < (k+1.5)\min[A] \end{cases} \tag{30}$$

where [A] is the array of the fish area, a is the area of fish in [A], and min[A] is the minimum area of fish in [A].

The ***total sum of fish*** ($\varphi$) is given as

$$\varphi = \sum_{i=1}^{N} (\lambda_i) \tag{31}$$

where N is the number of elements in [A].

## 4. Performance Evaluation and Results

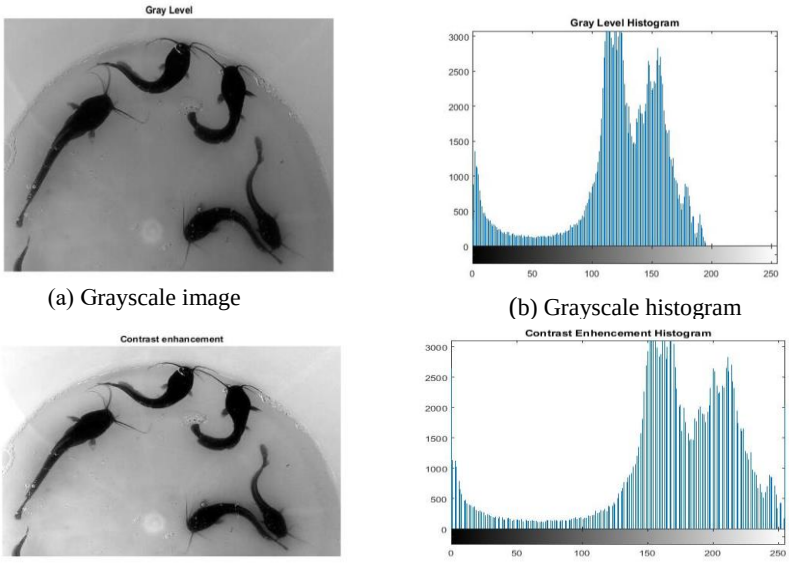
The performance of the recognition and counting algorithms was evaluated for sensitivity, specificity, and accuracy and then compared with the results obtained using the CH-FD and BD-FD approaches.

### 4.1 Image Pre-processing Performance Analysis

**Fig. 8** shows the original image in RGB, which was first converted from RGB to gray level. The RGB was based on a coordinate system where each color appeared in its primary spectral components of red, green, and blue, represented by 24 bits. The gray level encodes the image using 8 bit/pixel data, which allows 256 different grey levels. A pixel value of 0 represents 'black,' while a pixel value of 255 represents white. Intermediate values between 0 and 255 represent the varying shades of gray. The simplicity provided by grayscale made it suitable for this study because the image processing used the concept of comparing region pixels [28]. Using grayscale data with real scalar algebraic operations was sufficient to detect the shapes required for this study. On the contrary, the use of RGB would require a more complex computation using an algebraic vector to differentiate colors.

**Fig. 8.** Original image in RGB

**Fig. 9**(a) shows the resulting image after conversion to the gray levels. The histogram of the gray level image in **Fig. 9**(b) revealed that most of the pixels were concentrated within the 100- to 200-pixel range. The contrast enhancement spread the pixel concentration from approximately 140 to the maximum 255, as shown in the histogram in **Fig. 9**(d). The contrast enhancement mapped the intensity values in the grayscale level image, $I\ (x, y)$, to new values, thus increasing the contrast of the output image, as shown in **Fig. 9**(c). This pre-processing step was utilized on more than one image and proved to be effective.




(a) Grayscale image

(b) Grayscale histogram

(c) Contrast-enhanced image

(d) Contrast enhancement

**Fig. 9.** Grayscale conversion and enhancement

### 4.1.1 Median Filter Performance Analysis

A median filter was further applied to smooth the noisy regions while preserving the structures of the boundaries of the fish image. At the edge boundary, the output pixel was computed such that a white pixel from a black region near the edge was not used in the computation. Likewise, a black pixel from an adjacent white region near the edge was not used in the computation of the output pixel because these alternating pixels found in the adjacent regions were noise.

**Fig. 10**(a) shows an unfiltered binary image with salt-and-pepper noise and blurred edges. **Fig. 10**(b) shows the effect of the median filter on the image; unlike smoothing using averaging, the median filter preserved the edge structures while simultaneously smoothing the uniform regions. The filter also removed the salt-and-pepper noise and most other small artifacts.

The complexity of the two operations, RGB to grey conversion and grey morphological operation, in the pre-processing are both linear and thus amount to *O(2n)*, where *n* is the number of images. This means that the relationship between the input and the preprocessing operations done to complete its execution is linear. Thus, by ignoring the constant the final complexity is *O(n)*

(a) Unfiltered binary image (b) Filtered binary image

**Fig. 10.** Effect of the median filter

### 4.2 Image Segmentation Algorithm Performance Analysis

The image segmentation process involved three steps: image binarization by Otsu thresholding, morphological operations, and contour detection. Segmentation was applied, as discussed in Section 3.0. **Fig. 11**(a) shows the binarization of the image, which resulted in the creation of holes, some minor noise, and cuts on the image that further thin the fish.


(a) Binary image (b) Complement of the binary image

**Fig. 11.** Image segmentation

From the complement of the image in **Fig. 11**(b), the two fish at the top were joined with a thin line. Morphological operations were conducted to remove the thin line and holes and to fix the possible cuts, as shown in **Fig. 12**.

The edge boundaries were successfully traced using contour detection and superimposed

on the binary image, as shown in **Fig. 13**. The contour detection technique searched for boundary pixels with substantial intensity changes and operated based on the boundary consisting of a set of points that were connected and located on the outline of the object. The contour detector used the binary image as an input and computed the objects' boundaries by hierarchically maintaining records of the holes inside the parent object. This information was extracted and traced onto each of the fish components, as shown in **Fig. 13**(a), (b), (c), and (d).

**Fig. 12.** Morphological operation on binary image

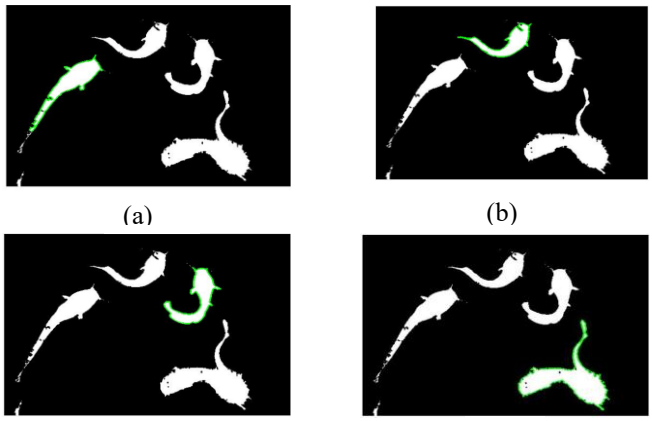



**Fig. 13.** Contour detection and tracing

The overall performance of the segmentation algorithm was determined using a supervised evaluation method. In supervised evaluation, a ground truth (GT) is generated, to which the segmented image (SI) is compared. In this study, the pixels of the GT—the image before segmentation—and the SI were extracted and compared. **Fig. 14** shows a sample of the GT and SI.

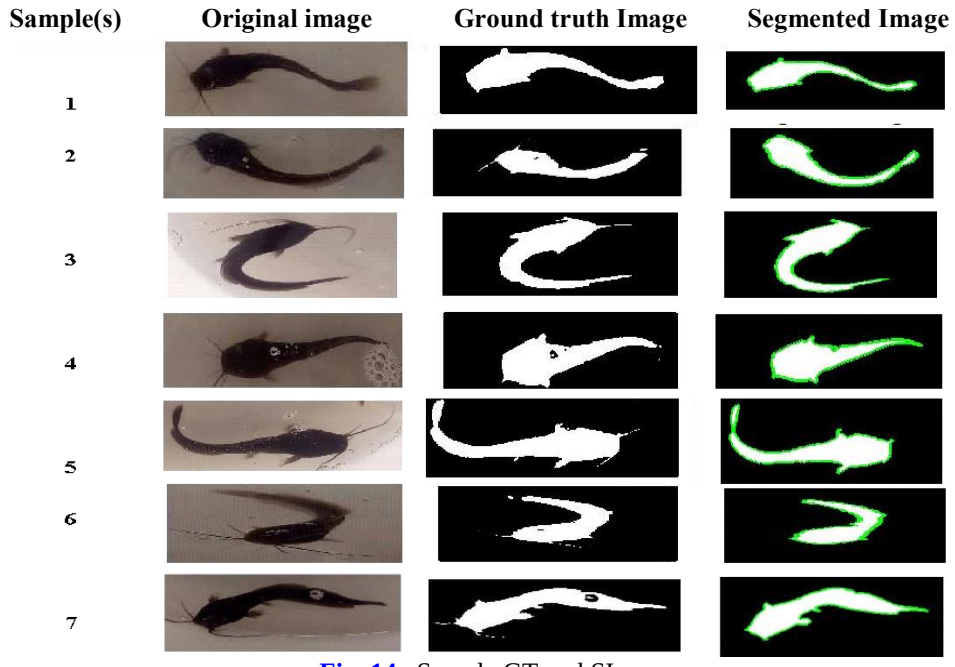



**Fig. 14.** Sample GT and SI

The pixel in both the GT image and the SI was extracted, as shown in **Table 1**. The GT pixels in all samples were greater in number than those of the SI because the application of erosion during the segmentation process reduced the images' pixels.

**Table 1.** Pixels of the detected GT and SI

| Sample | GT | SI | Accuracy (%) |
|---|---|---|---|
| 1 | 2477 | 2183 | 88.13 |
| 2 | 3299 | 2927 | 88.72 |
| 3 | 6538 | 5939 | 90.83 |
| 4 | 3351 | 3023 | 90.21 |
| 5 | 6562 | 5927 | 90.32 |
| 6 | 2346 | 2080 | 88.66 |
| 7 | 6171 | 5808 | 94.11 |
| 8 | 5254 | 4840 | 92.12 |
| 9 | 5182 | 4509 | 87.01 |
| 10 | 4228 | 3787 | 89.57 |

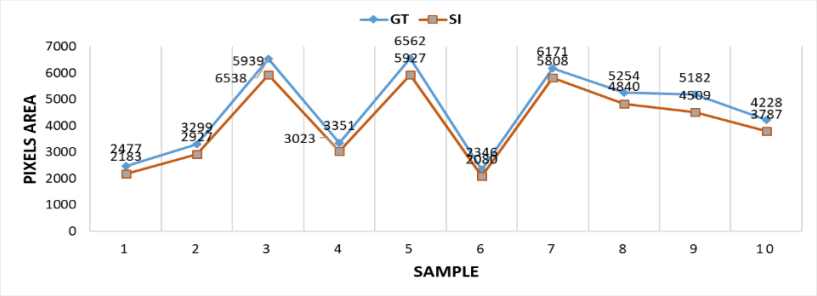



**Fig. 15.** Segmentation algorithm performance

**Fig. 15** shows the segmentation performance against the GT. The red curve represents the GT, while the blue curve represents the segmented plot. This analysis revealed that the segmentation algorithm effectively segmented the image. The overall performance of the segmentation algorithm ($\mathcal{E}$) is the ratio of the pixel in the detected SI ($\chi$) to the pixel in the GT ($\Upsilon$):

$$\mathcal{E} = \frac{\chi}{\Upsilon} \times 100 \tag{32}$$

Hence, the performance of the segmentation algorithm using the detected pixels that matched the GT was 90.34 %, resulting from the morphological operation that was performed on the image to reduce the false connections between the fish.

Considering the complexity of the segmentation algorithm, the thresholding and morphological operations complexities are both $O(n)$, while the edge segmentation has the complexity of $O(n^2)$. Therefore the overall complexity is $O(n^2)+O(2n)$. However, as the sample tends to infinity, the significance of $O(2n)$ becomes negligible. Thus the segmentation algorithm complexity is $O(n^2)$

### 4.3 ANN Fish Recognition Algorithm Performance Analysis

The ANN was trained with CH-FD and BD-FD. The dataset used contained 1368 images of Catfish and Non-catfish, as shown in **Table 2**. The training dataset was divided into 60% training, 20% validation, and 20% testing samples. These datasets for both the CH-FD and BD-FD were obtained from the image samples used previously.

**Table 2.** Datasets

| Object features | Training data (%) | Testing data (%) | Validation data (%) | Datasets |
|---|---|---|---|---|
| Catfish | 60 | 20 | 20 | 684 |
| Non-catfish | 60 | 20 | 20 | 684 |
| Total Datasets | | | | 1368 |

The performance of the network classifier was measured in terms of accuracy, sensitivity, and specificity.

#### 4.3.1 Performance analysis of the ANN recognition system using CH-FD

After training the network, the overall classification/recognition rate and accuracy of the network were evaluated. During the training, the network divided the input and the target data into three different sample types: training, validation, and testing. The training samples were used to train the network, and the weights were adjusted based on the output errors. The validation samples were used to measure the network generalization and halt training when the generalization stopped improving, while testing samples were used to independently measure the performance of the network after the training. The network was trained and tested using different threshold values, which varied from 0.1 to 1.0. For the training using CH-FD, the percentages of correctly classified catfish and non-catfish were 98.3% and 98.7%, respectively. Consequently, the optimal features for the training data yielded 98.6% accuracy, 98.7% sensitivity, and 98.3% specificity. For the CH-FD testing, the percentages of correctly classified catfish and non-catfish were both 100%. Thus, testing data yielded 100% for accuracy, sensitivity, and specificity. From the results of the experiments shown in **Table 3**, the highest accuracies were obtained at threshold 0.6 and 0.5. Even though 0.6 thresholds gave a better training result, but it takes about 13 epochs to generalize while 0.5 took about 7 epochs. Thus, 0.5 is adopted for the model and recorded 0.004088 MSE.

#### 4.3.2 Performance Analysis of the ANN using BD-FD

The BD-FD model optimal performance was attained at epoch 7 and 0.027086 MSE. The training results using the BD-FD data demonstrated that the optimal features yielded 97.5% accuracy, 95.0% sensitivity, and 100.0% specificity. The testing results show that the optimal features yielded 96.6% accuracy, 94.2% sensitivity, and 99.0% specificity. From the results of the experiments in **Table 3**, the highest testing accuracy of 96.6% was obtained at a threshold of 0.8.

**Table 3.** CH-FD and BD-FD experimental results

| Exp. | Threshold | TRAINING | | TESTING | |
|---|---|---|---|---|---|
| | | BD-FD | CH-FD | BD-FD | CH-FD |
| 1 | 0.1 | 91.3 | 97.2 | 92.7 | 97.6 |
| 2 | 0.2 | 96.0 | 98.2 | 94.1 | 99.5 |
| 3 | 0.3 | 95.2 | 97.9 | 93.2 | 99.5 |
| 4 | 0.4 | 94.1 | 98.4 | 94.9 | 99.5 |
| 5 | **0.5** | 96.1 | **98.5** | 95.6 | **100** |
| 6 | **0.6** | 95.2 | **98.6** | 96.1 | **100** |
| 7 | 0.7 | 95.5 | 98.6 | 95.1 | 99.5 |
| 8 | **0.8** | **97.5** | 98.1 | **96.6** | 98.5 |
| 9 | 0.9 | 94.7 | 97.8 | 93.7 | 98.0 |
| 10 | 1.0 | 63.2 | 85.9 | 66.3 | 83.3 |

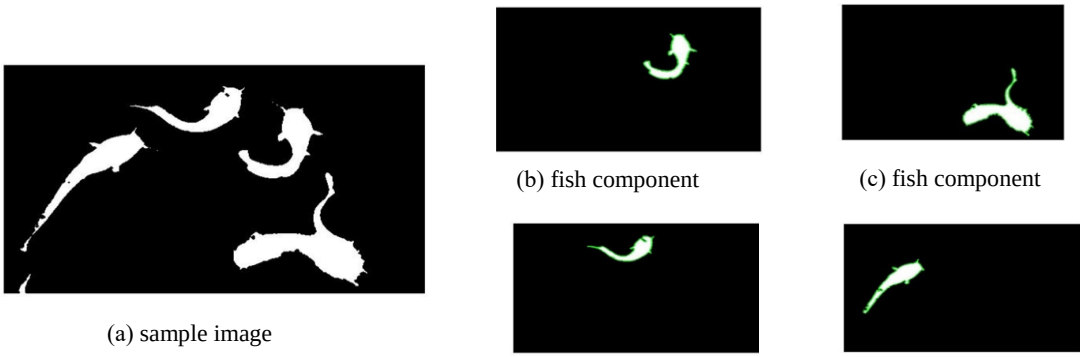
**Table 4** presents a comparison between the ANN performance using CH-FD and BD-FD. The CH-FD performed better in both training and testing with an accuracy of 98.5% and 100%, respectively. While the BD-FD recorded the performance of 97.5% and 96.6% for training and testing, respectively. Compacting the boundaries pixels using chain code improved the accuracy by normalizing and making the boundary pixel invariant to rotation, translation, and scaling. Conversely, the boundary pixels without the application of the shape representation scheme were not invariant to rotation, translation, and scaling and this affected the accuracy of the recognition. On the other hand, the training complexity of the ANN is estimated as $O(n^4)$ [29].

**Table 4.** ANN performance using CH-FD and BD-FD

| PERFORMANCE | TRAINING | | TESTING | |
|---|---|---|---|---|
| | BD-FD | CH-FD | BD-FD | CH-FD |
| True Positive (TP) | 454 | 472 | 97 | 103 |
| False Negative (FN) | 24 | 6 | 6 | 0 |
| True Negative (TN) | 479 | 471 | 101 | 102 |
| False Positive (FP) | 0 | 8 | 1 | 0 |
| Sensitivity ($\frac{TP}{TP+FN}$) | 95.0% | 98.7% | 94.2% | 100% |
| Specificity ($\frac{TN}{TP+FN}$) | 100% | 98.3% | 99.0% | 100% |
| Accuracy ($\frac{TP+TN}{TP+TN+FP+FN}$) | 97.5% | **98.5%** | 96.6% | **100%** |

### 4.4 Catfish Counting Algorithm Performance Analysis

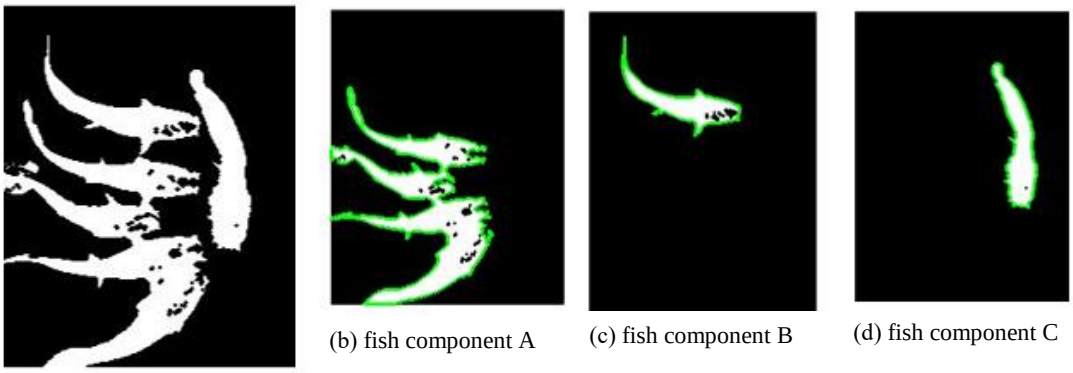
The first step in the counting scheme was obtaining the pixel area of the fish in the image by extracting each area from the background, including the overlapped fish. **Fig. 16** and **17** show the results of this step for two images. **Fig. 16**(b), (d), and (e) are the fish components of a single fish, while **Fig. 16**(c) contained overlap. Likewise, **Fig. 17**(c) and **Fig. 17**(d) each contained a single fish component, while **Fig. 17**(b) contained an overlapped component. These areas were stored in an array, and the set conditions from Section 3.0 were applied to obtain the catfish count.



(a) sample image (b) fish component (c) fish component (d) fish component C (e) fish component D

**Fig. 16.** Counting Analysis

The areas of each extract were obtained and analyzed. **Fig. 16** shows that the fish components A, B, C, and D had pixel area values of 4452, 3253, 8104, and 3975, respectively. The least pixels were in fish component B with a pixel area of 3253. Components A, D, and C were 1.37, 1.22, and 2.49 of component B and, hence, represented fish counts of one, one, and two, respectively. Resultingly, the total number of fish in sample 1, including the minimum area (component B), was five. Overlap only occurred with the two fish in component C.



(a) sample image (b) fish component A (c) fish component B (d) fish component C

**Fig. 17.** Counting Analysis II

**Fig. 17** shows that the fish components A, B, and C had pixel area values of 7583, 1826, and 2351, respectively. The minimum pixels in the array was fish component B, with a pixel area of 1826. Components A and C were 4.15 and 1.28 of this minimum, and, therefore, represented fish counts of four overlapped fish and a fish, respectively. The total number of fish in sample 2, including the minimum area, component B, was therefore six. The overlap only occurred in component B with four fish.

**Table 5** shows the overall result and performance of the counting algorithm. The counting demonstrated 100% accuracy for the two sample images because the fish in both images only overlapped side-by-side. The low water level in the container and the low number of fish contributed to the high accuracy.

**Table 5.** Counting Algorithm Performance

| Sample image | Fish component | Number of fish | Number overlapped | Number of Fish | Number of Fish counted |
|---|---|---|---|---|---|
| 1 | A | 1 | | 5 | 5 |
| | B | 1 | | | |
| | C | 2 | 2 | | |
| | D | 1 | | | |
| 2 | A | 4 | 4 | 6 | 6 |
| | B | 1 | | | |
| | C | 1 | | | |

Hence, the factors $(\tau)$ that affect accuracy can be summarized as the depth of water $(h)$ the population of fish in the container $(p)$ and the size of the container $(c)$. This can be expressed as follows:

$$\tau \, \alpha \frac{1}{phc} \tag{33}$$

The overall complexity of the proposed counting algorithm is estimated as *O(4n).* As the number of the *n* tends to infinity, the constant can be ignored, thus the complexity becomes *O(n).*

## 5. Conclusion

This study developed a catfish recognition and counting algorithm consisting of five steps: image acquisition, pre-processing, segmentation, feature extraction and recognition, and counting. The segmentation algorithm using detected pixels demonstrated 90.34% accuracy when compared to the GT, due to the morphological operation that was performed on the image to reduce the false connections between the fish. The boundary features from the SI were extracted using CH-FD and BD-FD and were then used as the dataset for training an ANN. The CH-FD method demonstrated 100% accuracy, while 96.6% accuracy was obtained by the BD-FD approach. These results show that compacting the boundary pixels using chain code improved the accuracy by normalizing and making the boundary pixels invariant to rotation, translation, and scaling. The boundary pixels without the application of this shape representation scheme were not invariant to rotation, translation, and scaling. The counting demonstrated 100% accuracy for the two sample images because the fish in both images only overlapped side-by-side. The low water level in the container and the low number of fish contributed to the high accuracy. Hence, the factors that affect accuracy can be summarized as the depth of water $(h)$, the population of fish in the container $(p)$, and the size of the container $(c)$. Despite the contribution of this paper, more experiments are needed to establish the optimal values regarding water depth, fish population and size of the container. Deep learning can also be explored to recognize overlap fish patterns from other possible objects or illumination that may affect recognition accuracy. As the developed counting algorithm requires at least a single fish that is not overlapped, further studies are required to eliminate the need for a single fish while maintaining high accuracy.

## Acknowledgment

This work was supported in part by the TETFUND Institution-Based Research Intervention (IBRI) Fund of the Federal University of Technology, Minna, Nigeria. Reference No: TETFUND/FUTMINNA/2016-2017/6th BRP/01 and in part by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education (2017R1D1A1B03035988).

## References

[1] H. Ritchie and M. Roser, "Seafood Production," *OurWorldInData*, 2019. Article (CrossRef Link)
[2] B. C. Loh, V. Raman, and P. H. Then, "First Prototype of Aquatic Tool Kit: Towards Low-Cost Intelligent Larval Fish Counting in Hatcheries," in *Proc. of IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing*, pp. 192-195, Dec. 2011. Article (CrossRef Link)
[3] V. Raman, S. Perumal, S. Navaratnam, and S. Fazilah, "Computer Assisted Counter System for Larvae and Juvenile Fish in Malaysian Fishing Hatcheries by Machine Learning Approach," *Journal of Computer*, vol. 11, no. 5, pp. 423-431, 2016. Article (CrossRef Link)
[4] K. Potongkam and J. Miller, "Catfish Hatchery and Production Manual," National Special Programme for Food Security, 2006. [Online]. Available: https://docplayer.net/43237829-Catfish-hatchery-and-production-manual.html
[5] O. Oluwemimo and A. Damilola, "Socio-economic and policy issues determining sustainable fish farming in Nigeria," *International Journal of Livestock Production*, vol. 4, no. 1, pp. 1-8, 2013. Article (CrossRef Link)
[6] M. A. Adegboye, A. M. Aibinu, J. G. Kolo, A. M. Orire, T. A. Folorunso, and I. A. Aliyu, "Vibration Processing Analysis Approach to the Development of Fish Feeding Regime System, " *Computing, Information System, Development Informatics & Allied Research Journal*, vol. 8, no.1, pp. 141-148, Mar. 2017. Article (CrossRef Link)
[7] E. Daniel, "Quality fingerlings: Hatcheries to the rescue," *The Nation*, [Online]. Available: http://thenationonlineng. net/quality-fingerlings-hatcheries-to-the-rescue
[8] E. Awalludin, T. Arsad, and H. W. Yussof, "A Review on Image Processing Techniques for Fisheries Application," in *Proc. of Journal of Physics: Conference Series*, vol. 1529, p. 052031, 2020. Article (CrossRef Link)
[9] V. Garcia, D. A. SantAna, V. A. G. Zanoni, M. C. B. Pache, M. H. Naka, P. L. F. Albuquerque, T. Lewandowski, A. D. S. Oliveira, J. Victor, A. Rozales, M. W. Ferreira, E. Q. A. Queiroz, J. C. M. Almanza, and H. Pistori, "A new image dataset for the evaluation of automatic fingerlings counting," *Aquacultural Engineering*, vol. 89, p. 102064, 2020. Article (CrossRef Link)
[10] FAO, "Mass production of fry and fingerlings of the African catfish clariasgariepinus," 2016. [Online]. Available: http://www.fao.org/docrep/field/003/ac182e/ac182e03.htm
[11] T. W. Huang, J. N. Hwang, and C. S. Rose, "Chute based automated fish length measurement and water drop detection," in *Proc. of 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1906-1910, May. 2016. Article (CrossRef Link)
[12] I. Aliyu, K. J. Gana, A. A. Musa, J. Agajo, A. M. Orire, F. T. Abiodun, and M. A. Adegboye, "A Proposed fish counting algorithm using image processing technique," *ATBU Journal of Science and Technology and Education*, vol. 5, no. 1, pp. 1-11, 2017. Article (CrossRef Link)
[13] R. Labuguen, E. J. P. Volante, A. Causo, R. Bayot, G. Peren, R. M. Macaraig, N. J. C. Libatique, and G. L. Tangonan, "Automated fish fry counting and schooling behavior analysis using computer vision," *IEEE 8th International Colloquium on Signal Processing and its Applications (CSPA)*, pp. 255-260, May 2012. Article (CrossRef Link)
[14] J. Ni, Z. Khan, S. Wang, K. Wang, and S. K. Haider, "Automatic detection and counting of circular shaped overlapped objects using circular hough transform and contour detection," *2016 12th World Congress on Intelligent Control and Automation (WCICA)*, pp. 2902-2906, 2016. Article (CrossRef Link)

[15] P. F. Newbury, P. F. Culverhouse, and D. A. Pilgrim, "Automatic fish population counting by artificial neural network," *Aquaculture*, vol. 133, no. 1, pp. 45-55, 1995. Article (CrossRef Link)
[16] S. Yada and H. Chen, "Weighing type counting system for seedling fry," *Nippon Suisan Gakkaishi*, vol. 63, no. 2, pp. 178-183, 1997. Article (CrossRef Link)
[17] K. Friedland, D. Ama-Abasi, M. Manning, L. Clarke, G. Kligys, and R. Chambers, "Automated egg counting and sizing from scanned images: rapid sample processing and large data volumes for fecundity estimates," *Journal of Sea Research*, vol. 54, no. 4, pp. 307-316, 2005. Article (CrossRef Link)
[18] M. O. Alver, T. Tennøy, J. A. Alfredsen, and G. Øie, "Automatic measurement of rotifer Brachionus plicatilis densities in first feeding tanks," *Aquacultural engineering*, vol. 36, no. 2, pp. 115-121, 2007. Article (CrossRef Link)
[19] J. Han, N. Honda, A. Asada, and K. Shibata, "Automated acoustic method for counting and sizing farmed fish during transfer using DIDSON," *Fisheries Science*, vol. 75, no. 6, pp. 1359-1367, 2009. Article (CrossRef Link)
[20] Y. Toh, T. Ng, and B. Liew, "Automated fish counting using image processing," in *Proc. of 2009 International Conference on Computational Intelligence and Software Engineering*, pp. 1-5, 2009. Article (CrossRef Link)
[21] X. Zheng and Y. Zhang, "A fish population counting method using fuzzy artificial neural network," in *Proc. of 2010 IEEE International Conference on Progress in Informatics and Computing (PIC)*, pp. 225-228, 2010. Article (CrossRef Link)
[22] S. Luo, X. Li, D. Wang, J. Li, and C. Sun, "Automatic Fish Recognition and Counting in Video Footage of Fishery Operations," in *Proc. of 2015 International Conference on Computational Intelligence and Communication Networks (CICN)*, pp. 296-299, 2015. Article (CrossRef Link)
[23] Y. Duan, L. H. Stien, A. Thorsen, Ø. Karlsen, N. Sandlund, D. Li, Z. Fu, and S. Meier, "An automatic counting system for transparent pelagic fish eggs based on computer vision," *Aquacultural Engineering*, vol. 67, pp. 8-13, 2015. Article (CrossRef Link)
[24] R. Lumauag and M. Nava, "Fish Tracking and Counting using Image Processing," in *Proc. of IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM)*, pp. 1-4, 2018. Article (CrossRef Link)
[25] J. Kaewchote, S. Janyong, and W. Limprasert, "Image recognition method using Local Binary Pattern and the Random forest classifier to count post larvae shrimp," *Agriculture and Natural Resources*, vol. 52, no. 4, pp. 371-376, 2018. Article (CrossRef Link)
[26] S. M. D. Lainez and D. B. Gonzales, "Automated Fingerlings Counting Using Convolutional Neural Network," in *Proc. of IEEE 4th International Conference on Computer and Communication Systems (ICCCS)*, pp. 67-72, 2019. Article (CrossRef Link)
[27] M. A. Adegboye, A. M. Aibinu, J. G. Kolo, I. Aliyu, T. A. Folorunso, and S. H. Lee, "Incorporating intelligence in fish feeding system for dispensing feed based on fish feeding intensity," *IEEE Access*, vol. 8, pp. 91948-91960, 2020. Article (CrossRef Link)
[28] O. M. Olaniyan, M. A. Adegboye, I. A. M. Bolaji Omodunbi, and T. Badmus, "A decision support system for automatic fertilizer application to tomato plants using artificial neural network," *FUW Trends in Science & Technology*, vol. 3, no. 2, pp. 600-604, 2018.
[29] Kaper, "Computational Complexity Of Neural Networks," 2020. [Online]. Available: https://kasperfred.com/series/computational-omplexity/computational-complexity-of-neural networks

**Ibrahim Aliyu** received his B.Eng and M.Eng Degree from the Dept. of Computer Engineering, Federal University of Technology, Minna, Nigeria, in 2014 and 2018, respectively. He is currently a Ph.D. candidate at Dept. of Computer Engineering, Chonnam National University, Yeosu, Korea, under the supervision of Prof. Chang Gyoon Lim through Korean Government Scholarship Program. His research interest includes Machine Learning, Cloud Computing, Blockchain, Network Security, SDN and BCI.

**Jonathan G. Kolo R** received the B.Eng., M.Sc. and Ph.D. degrees in Electrical/Electronic Engineering from Ahmadu Bello University, Zaria, in Electrical and Electronic Engineering from University of Lagos, Lagos, Nigeria and University of Nottingham, Malaysia Campus, Semenyih, Malaysia in 1994, 2002 and 2013, respectively. He's currently an Associate Professor with the Electrical and Electronic Engineering Department, Federal University of Technology, Minna, Nigeria. His research interests include wireless sensor networks, data compression, intelligent system design, wireless communications, and artificial intelligence.

**Abiodun M. Alibinu** received the Ph.D. degree from International Islamic University Malaysia in 2010. He is presently a Professor with the Department of Mechatronics Engineering, Federal University of Technology, Minna, Nigeria. His research interests include Digital signal and Image processing, Instrumentation and measurement, Intelligent system design, and artificial intelligence with an emphasis on artificial neural networks and Genetic Algorithm.

**Mutiu A. Adegboye** received the B.Eng. and M.Eng. Degrees in computer engineering in 2014 and 2018, respectively, from Federal University of Technology, Minna, Nigeria. He's currently pursuing Ph.D. degree in engineering at Robert Gordon University, Aberdeen, United Kingdom through PTDF scholarship. Since 2018, he has been a Lecturer with the Computer Engineering Department, Federal University, Oye-Ekiti, Nigeria. His research interest includes computer vision, signal processing, artificial intelligence.

**Chang Gyoon Lim** received his Ph.D. in Dept. of Computer Engineering in Wayne State University, U.S.A. in 1997. Since September of 1997, he has been working for the department of Computer Engineering, Chonnam National University, Yeosu, Korea, as a professor. He was a director of Home Robot Center in Gwangju Techno Park. He is a director of Korean Society for Internet Information. He plays the role of Gwangju-Jeonnam Cloud Computing Leaders Forum. His current research interests include BCI, Machine Learning, Soft Computing, IoT, Cloud Computing, and Embedded Software.