Biometric Cattle Identification Approach Based on Weber's Local Descriptor and AdaBoost Classifier

Tarek Gaber[a,b,c,1,*], Alaa Tharwat[c,d], Aboul Ella Hassanien[c,e], Vaclav Snasel[f]

[a] *Faculty of Computers and Informatics, Suez Canal University, Ismailia, Egypt*
[b] *IT4Innovation, VSB-TU of Ostrava, Ostrava, Czech Republic*
[c] *Scientific Research Group in Egypt (SRGE), http://www.egyptscience.net*
[d] *Faculty of Engineering, Suez Canal University, Ismailia, Egypt*
[e] *Faculty of Computers and Information, Cairo University, Egypt*
[f] *FEECS, Dept. of Computer Science and IT4Innovations, VSB-TU of Ostrava, Czech Republic*

Abstract

In this paper, we proposed a new and robust biometric-based approach to identify head of cattle. This approach used the Weber Local Descriptor (WLD) to extract robust features from cattle muzzle print images (images from 31 head of cattle were used). It also employed the AdaBoost classifier to identify head of cattle from their WLD features. To validate the results obtained by this classifier, other two classifiers (k-Nearest Neighbor (k-NN) and Fuzzy-k-Nearest Neighbor (Fk-NN)) were used. The experimental results showed that the proposed approach achieved a promising accuracy result (approximately 99.5%) which is better than existed proposed solutions. Moreover, to evaluate the results of the proposed approach, four different assessment methods (Area Under Curve (AUC), Sensitivity and Specificity, accuracy rate, and Equal Error Rate (EER)) were used. The results of all these methods showed that the WLD along with AdaBoost algorithm gave very promising results compared to both of the k-NN and Fk-NN algorithms.

Keywords: Cattle identification, Weber Local Descriptor (WLD), k-Nearest Neighbor, Fuzzy-k-Nearest Neighbor, Muzzle print images, dimensionality reduction, feature extraction, AdaBoost classifier, Animal identification

---

*Corresponding author

Email addresses: `tmgaber@gmail.com` (Tarek Gaber), `engalaatharwat@hotmail.com` (Alaa Tharwat), `aboitcairo@gmail.com` (Aboul Ella Hassanien), `Vaclav.Snasel@vsb.cz` (Vaclav Snasel)

[1] My present address is FEECS, Dept. of Computer Science, VSB-TU of Ostrava, Ostrava - Poruba, 708 33, Czech

1. Introduction

Cattle identification and traceability are very crucial to control safety policies of animals and management of food production. Many international organizations, e.g. food safety and world animal health, have formally recognized the significant values of the development of the animal identification and traceability systems and they further actively promoted for these systems (Schroeder and Tonsor, 2012). Such values include (a) controlling the widespread of the animal diseases by identifying and detecting infected animals, (b) reducing losses of livestock producers by controlling the diseases, (c) decreasing the government cost by the control, intervention, and eradication of the outbreak diseases (Bowling et al., 2008). Therefore, especially after the discovery of the Bovine Spongiform Encephalopathy (BSE), advanced animal identification and traceability systems were evolved and deployed by big beef exporters and have been increasingly used by ranked beef importing countries (Schroeder and Tonsor, 2012).

Marchant (2002) reported that animal identification can be achieved using many different methods which could be classified as mechanical, electronic, and biometric. The mechanical class includes methods such as ear notching, ear tags, branding, and tattoos. Nonetheless, as reported in (Shadduck and Golden, 2002; Allen et al., 2008), the mechanical-based identification suffers from a number of limitations. The ear notching method is not suitable for large-scale identification systems. The ear tag methods (metal clips and plastic tags) are not so expensive, but they may cause animal infections (Allen et al., 2008). The branding and tattoo methods are not achieving a relatively good accuracy as in one herd, all head of cattle are identically branded. Thus, they are not useful to uniquely differentiate between various head of cattle in the same herd. In addition, these methods take more time than other modern techniques (Shadduck and Golden, 2002).

Animal identification systems based on electronic methods (Marchant, 2002; Shanahan et al., 2009) used *Radio Frequency Identification* (RFID) to identify animals. These methods are mainly based on attaching two devices with the animals. One device contains a unique identification number and the other is the reading device which reads and interprets animals code (the unique identification number). When a code is scanned, the reading device sends it to a database for future actions. The main limitation of this method is that the attached devices may get lost, removed, or damaged (Marchant, 2002).

The third method is the biometric-based animal identification (Shadduck and Golden, 2002; Jiménez-Gamero et al., 2006; Rusk et al., 2006; Corkery et al., 2007; Allen et al., 2008; Barry et al., 2008; Gonzales Barron et al., 2008; Rojas-Olivares et al., 2011; Adell et al., 2012). Similar to biometric-based human identification, a number of biometric animal have proposed to uniquely identify animals. Retina-based identification systems (Rusk et al., 2006; Allen et al., 2008; Barry et al., 2008; Gonzales Barron et al., 2008; Adell et al., 2012) depend on the retinal image recognition (RIR) which utilizes the fact that the retina vessels of each head of cattle is a unique identifier. DNA-based methods (Jiménez-Gamero et al., 2006) were also proposed to identify meat products that were produced from a given specific animal. Although this method, in case of head of cattle, gives a higher identification rate than the other methods, it is intrusive, and not cost-effective and it could last days or weeks to obtain the identification result (Rusk et al., 2006). Other biometric-based methods include animal facial recognition (Shadduck and Golden, 2002; Corkery et al., 2007) and muzzle-based identification (Minagawa et al., 2002; Noviyanto and Arymurthy, 2012; Awad et al., 2013; Noviyanto and Arymurthy, 2013).

The muzzle-based animal identification is based on the fact that the muzzle pattern or nose print of different animals of the same species are mostly unique (Baranov et al., 1993; Gonzales Barron et al., 2008). Thus, it is concluded that muzzle print is similar to a human's fingerprint. The muzzle-based approach is a very promising way for cattle identification as it can achieve a high accuracy (e.g. 90.6% in (Noviyanto and Arymurthy, 2012)). Using this approach, there is no need to attach or insert external parts within the animals. Moreover, it complies with most countries legal rules.

In the muzzle-based identification system, extracting discriminative features from the muzzle images is a very important step. Local invariant features are good ones as they are robust against many challenges such as noise, illumination, transformation, rotation, and occlusion. There are two methods to extract the local invariant features: sparse descriptor (Lowe, 1999) and dense descriptor (Chen et al., 2010). In the former method, the interest points (keypoints), are first detected, then a local patch, around these keypoints, is constructed, and finally invariant features are extracted. *Scale Invariant Feature Transformation* (SIFT) is considered one of the most well-known algorithms in the sparse descriptor type (Lowe, 1999). In the dense descriptor-based methods, local features are extracted from every pixel (pixel by pixel) over the input image. Examples of this method include *Local Binary Pattern* (LBP) and *Weber Local Descriptor* (WLD) (Ojala et al., 2002; Chen et al., 2010).

In this paper, a muzzle-based cattle identification approach was proposed. This approach consists of three phases: feature extraction, feature reduction, and classification. In the first phase, the WLD algorithm was used to extract local features. In the second phase, the Linear Discriminant Analysis (LDA) technique was used to reduce the features and further to discriminate between different images of various head of cattle. In the classification phase, three classifiers (AdaBoost, k-Nearest Neighbor (k-NN), and Fuzzy k-NN (Fk-NN)) were used to match between unknown cattle images and trained or labeled images and then based on the highest accuracy results, the best classifier was recommended for the cattle identification system.

The rest of the paper is organized as follows. Section 2 summarizes the related work of the cattle identification system based on information technology. Section 3 gives overviews of the techniques and methods used for the proposed approach while Section 4 describes our proposed approach in detail. Experimental results and discussion are introduced in Section 5 and Section 6, respectively. Finally, conclusions are summarized in Section 7.

2. Related Work

There are a number of the muzzle-based cattle identification approaches (Minagawa et al., 2002; Noviyanto and Arymurthy, 2012; Awad et al., 2013; Noviyanto and Arymurthy, 2013; Tharwat et al., 2014). These approaches used different techniques to extract biometric features from muzzle images. Minagawa et al. (2002) proposed the first cattle identification approach in which the joint pixels of the grooves were extracted by applying the image processing techniques, i.e. filtering, binary transforming, and thinning. The identification was then achieved by matching the joint pixels of a cattle image to the others or to itself. The experiments of their proposed approach were conducted on a database of 43 head of cattle and achieved minimum matching scores at 12% and maximum scores at 60%. The results also showed that the identification accuracy was around 30%.

The Speed Up Robust Features (SURF) and its variant (U-SURF) feature extraction techniques were used in (Noviyanto and Arymurthy, 2012). Noviyanto et al. used 15 muzzle print images in their experimental scenarios (10 images were used in the training phase, and five images were used in the testing phase). The SURF-based method was found superior to U-SURF-based one as the former achieved 90% identification accuracy against rotation conditions.

Awad et al. (2013) used SIFT technique to detect the interesting points of muzzle images for the purpose of cattle identification. To improve the robustness of their proposed approach, they applied the *RANdom SAmple Consensus* (RANSAC) algorithm along with the output of SIFT technique. In their experiment, they used six images for each head of cattle and in total their database includes 90 images ($6 \times 15 = 90$). They achieved 93.3% accuracy of cattle identification.

Also, Noviyanto and Arymurthy (2013) applied the SIFT technique to muzzle patterns lifted on paper in order to achieve cattle identification. To improve the identification performance of their system, they also proposed a new matching refinement technique based on the keypoint of the orientation information.

They tested the proposed system using a database composed of 160 muzzle images left on papers and taken from 20 head of cattle. The achieved accuracy results using SIFT only were equal to 0.0167 *Equal Error Rate* (EER) whereas using SIFT along with the proposed new matching refinement technique minimized the EER to be 0.0028.

Tharwat et al. (2014) used the LBP technique for the feature extraction phase of a muzzle-based cattle identification approach. The LBP was used as it extracts robust texture features which are invariant to rotation and occlusion of the images. They also used LDA to (a) address LBP high dimensionality problem, and (b) discriminate between different classes, thus improving the accuracy of their proposed system. For the identification phase, they tested four different classifiers (Nearest Neighbor, k-Nearest Neighbor (k-NN), Naive Bayes, and Support Vector Machine (SVM)). The results showed that their proposed approach achieved 99.5% identification accuracy.

3. Preliminaries

This section gives overviews of the techniques, algorithms, and methods used in the design of the proposed approach.

3.1. Weber Local Descriptor (WLD)

The WLD technique is an image descriptor technique which describes an image as a histogram of gradient orientations and differential excitations (Chen et al., 2010). It is originally inspired by Weber's Law where Ernst Weber, in the 19^{th} century, observed that the ratio between an increment threshold and the background intensity is constant and this can be formally expressed as follows:

$$\frac{\Delta I}{I} = k \tag{1}$$

where ΔI represents the increment threshold, I refers to the initial intensity or an image background, and k denotes the constant value even if I is changing. The fraction $\frac{\Delta I}{I}$ is known as *Weber law or Weber fraction* (Chen et al., 2010).

In WLD algorithm, features are extracted from each pixel in an image. In general, WLD algorithm consists of three steps, finding differential excitations, gradient orientations, and building the histogram. For each pixel in the input image, the differential excitation is first computed and the gradient orientation is then calculated to extract local features. Finally, a WLD histogram is built by combining differential excitation and gradient orientation for each pixel (Chen et al., 2010). These steps are further explained below.

3.1.1. Differential Excitation (ξ):

A differential excitation (ξ) of a pixel is calculated as follows:

1. Calculating the difference between the pixel x_c (the center pixel) and its neighbors using Equation (2) (Chen et al., 2010).

$$\nu_s^{00} = \sum_{i=0}^{p-1}(\Delta x_i) = \sum_{i=0}^{p-1}(x_i - x_c) \tag{2}$$

   where $x_i(i = 0, 1, \ldots, p-1)$ represents the intensity of the i^{th} neighbors of x_c and p refers to the number of neighbors. An illustrative example, inspired by the one in (Chen et al., 2010), is given in Figure 1 to show how the differential excitation is calculated. As shown in the figure, there are eight neighbors to x_c, where $p = 8$. To calculate the differential excitation and the orientation, four filters, f_{00}, f_{01}, f_{10}, and f_{11} are used to calculate $\nu_s^{00}, \nu_s^{01}, \nu_s^{10}$, and ν_s^{11}, respectively, where, ν_s^{00} represents the difference between x_c and its neighbors as shown in Equation (2), $\nu_s^{01} = x_c$, $\nu_s^{10} = x_5 - x_1$, and $\nu_s^{11} = x_7 - x_3$.

2. Computing the ratio between the differences, ν_s^{00}, and the intensity of the current pixel, $\nu_s^{01} = x_c$. This can be achieved using Equation (3).

$$G_{ratio}(x_c) = \nu_s^{00}/\nu_s^{01} \tag{3}$$

3. Applying the arc-tangent function on $G_{ratio}(.)$ to get the differential excitation of (x_c), as shown in Equation (4).

$$\xi(x_c) = G_{arctan}[G_{ratio}(x_c)] = \arctan\left[\nu_s^{00}/\nu_s^{01}\right] = \arctan\left[\sum_{i=0}^{p-1}\left(\frac{x_i - x_c}{x_c}\right)\right] \tag{4}$$

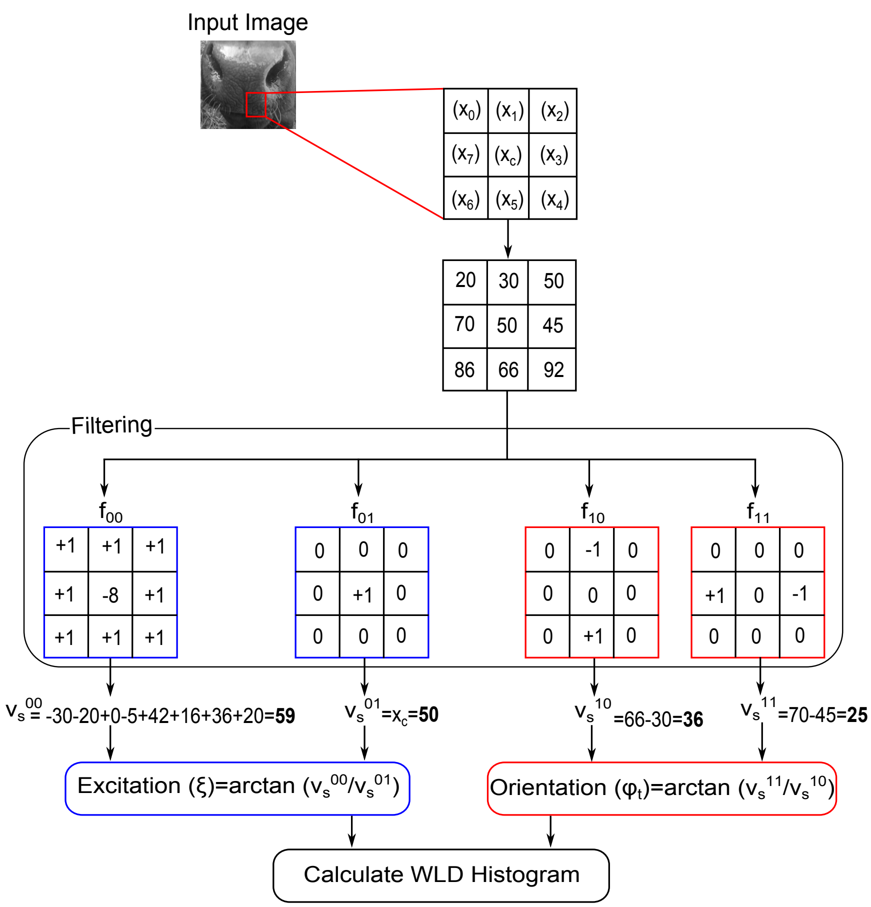


Figure 1: Illustration of the computation of the WLD algorithm.

3.1.2. Orientation (ϕ_t):

The orientation of a pixel (x_c) is computed as follows:

1. Computing the gradient orientation of the current pixel, x_c, by calculating

the changes in the horizontal and vertical directions as follows:

$$\theta(x_c) = \arctan\left(\frac{\nu_s^{11}}{\nu_s^{10}}\right) = \arctan\left(\frac{x_7 - x_3}{x_5 - x_1}\right) \tag{5}$$

2. Quantizing the gradient orientation by transforming it into T dominant orientation. This is achieved by first mapping θ to $\acute{\theta}$ as follows:

$$\acute{\theta} = \arctan2(\nu_s^{11}, \nu_s^{10}) + \pi \tag{6}$$

where

$$\arctan2(\nu_s^{11}, \nu_s^{10}) = \begin{cases} \theta, & \nu_s^{11} > 0 \; and \; \nu_s^{10} > 0 \\ \pi - \theta, & \nu_s^{11} > 0 \; and \; \nu_s^{10} < 0 \\ \theta - \pi, & \nu_s^{11} < 0 \; and \; \nu_s^{10} < 0 \\ -\theta, & \nu_s^{11} < 0 \; and \; \nu_s^{10} > 0 \end{cases} \tag{7}$$

where $\theta \in [-\pi/2, \pi/2]$ and $\acute{\theta} \in [0, 2\pi]$.

3. Finally, the quantization function is calculated as in Equation (8) (Chen et al., 2010).

$$\phi_t = f_q(\acute{\theta}) = \frac{2t}{T}\pi \; , \; \text{ and } \; t = mod\left(\left\lfloor \frac{\acute{\theta}}{2\pi/T} + 0.5 \right\rfloor, T\right) \tag{8}$$

3.1.3. WLD Histogram:

The WLD histogram is computed, as shown in Figure (1), using the values of both the Differential Excitation (ξ_j) and Orientation (ϕ_t) at each pixel. In other words, this histogram consists of (ξ_j, ϕ_t), $j = 0, 1, \ldots, N-1$ and $t = 0, 1, \ldots, T-1$, where N represents the dimensionality of an image and T denotes the number of the dominant orientation. The steps of WLD algorithm are summarized in Algorithm 1.

3.2. Linear Discriminant Analysis (LDA)

LDA is a well-known dimensionality reduction technique in machine learning applications. LDA aims to find a linear combination of features which linearly separates two or more classes. Formally, LDA attempts to find a transformation

Algorithm 1 : WLD Algorithm

Initialize the size of the patch or sub-region, (e.g. 3×3, 5×5, 7×7, etc.).
Divide the images into patches or sub-regions.
Compute the Differential Excitation (ξ) as follows:
for all pixels in an image **do**
    Compute the difference between the center or current pixel (x_c) and all its surrounding pixels as follows, $\nu_s^{00} = \sum_{i=0}^{p-1} (\Delta x_i) = \sum_{i=0}^{p-1} (x_i - x_c)$.
    Compute the ratio between ν_s^{00} and x_c as follows, $G_{ratio}(x_c) = \frac{\nu_s^{00}}{\nu_s^{01}} = \sum_{i=0}^{p-1} \left(\frac{\Delta x_i}{x_c}\right)$.
    The final function will be as follows, $\xi(x_c) = \arctan(G_{ratio}) = \arctan\left[\sum_{i=0}^{p-1} \left(\frac{\Delta x_i}{x_c}\right)\right] = \arctan\left[\sum_{i=0}^{p-1} \left(\frac{x_i - x_c}{x_c}\right)\right]$.
end for
Compute Gradient Orientation ($\acute{\theta}$).
for all pixels in an image **do**
    Compute the changes in horizontal and vertical directions of the current pixel (x_c) as follows, $\theta(x_c) = \arctan\left[\frac{\nu_s^{11}}{\nu_s^{10}}\right] = \arctan\left[\frac{x_7 - x_3}{x_5 - x_1}\right]$.
    Now $\theta \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$, to get more texture information, θ mapped to $\acute{\theta} \in [0, 2\pi]$, so $\acute{\theta}$ will be as follows, $\acute{\theta} = \arctan2(\nu_s^{11}, \nu_s^{10}) + \pi$, where $\arctan2(\nu_s^{11}, \nu_s^{10})$ is calculated as in Equation (7).
    Compute the quantization function as follows, $\phi_t = (2t/T)\pi$.
end for
Compute WLD histogram ($WLD(\xi_j, \phi_t)$), where $j = 0, 1, \ldots, N-1, t = 0, 1, \ldots, T-1$.

matrix, W, that maximizes the Fisher's formula, $J(W) = \left|\frac{W^T S_b W}{W^T S_w W}\right|$, where $S_w = \sum_{j=1}^{c} \sum_{i=1}^{N_j} (x_i^j - \mu_j)(x_i^j - \mu_j)^T$ represents the within-class scatter matrix, where ${x_i}^j$ is the i^{th} sample of class j, μ_j is the mean of class j, c is the number of classes, and N_j is the number of samples in class j, $S_b = \sum_{j=1}^{c} (\mu_j - \mu)(\mu_j - \mu)^T$ is the between-classes scatter matrix, where μ refers to the mean of all classes, and W is the transformation matrix of LDA (Roth and Steinhage, 1999). The solution of Fisher's formula is a set of eigenvectors (V) and eigenvalues (λ) of W

and the LDA space consists of the eigenvectors which have higher eigenvalues. In our proposed approach, LDA was used to discriminate between different classes, where a class represents a head of cattle and each class consists of seven images (samples).

3.3. Classifiers

In the proposed approach, described in Section 4, a number of classifiers were used to achieve the identification of cattle. A brief summary about these classifiers is given below.

3.3.1. AdaBoost

AdaBoost (Adaptive Boosting) is a classifier ensemble algorithm consisting of a number of weak learners. A weak learner (classifier) is a simple, fast, and easy to implement classifier such as single level decision tree or simple neural networks (Kuncheva, 2014). The main idea of an ensemble classifier is to individually train its weak learners and then combine their decisions/predictions to determine a final decision. In other words, in an ensemble classifier, e.g. AdaBoost, a large margin classification is produced by iteratively combining a small number of the weighted-weak learners to construct a strong classifier.

A brief description of the AdaBoost classifier is as follows. As shown in Algorithm 2, the parameters of AdaBoost classifier are first initialized. As shown in the algorithm, the weights of all samples (w) are equal and they will be adjusted for each iteration. For each iteration (t), the training samples are selected based on these weights (w), and these samples are used to build the weak learner (C_t). The resubstitution error rate[2] of the current weak learner (ϵ_t), produced from the training data, is then calculated. If the error rate is more than 0.5, the weights (w) are reinitialized and the error rate is recalculated again. The

[2] In other words, it is the estimation of error based on the difference between the predicted values and the true labels of the training set.

Algorithm 2 : AdaBoost (Adaptive Boosting) Classifier

1: Given a training set $X = (x_1, y_1), \ldots, (x_N, y_N)$, where y_i represents the label of sample $x_i \in X$ and N denotes the total number of samples in the training set.

2: Initialize the parameters of AdaBoost classifier, the total number of iterations (T), type of weak learners, learning rate (λ), the weights w_j^i of each training sample, where w^i represents the weights of the i^{th} iteration, and $w^i = [w_1^i, \ldots, w_N^i], w_j^i \in [0, 1], \sum_{j=1}^{N} w_j^i = 1$. Usually the weights are initialized to be equal as follows, $w_j^1 = \frac{1}{N}, j = 1, \ldots, N$.

3: **for** $t = 1$ **to** T **do**

4: Take a sample D_t from X using distribution w^t.

5: Use the distribution D_t to train the weak learner (C_t) with a minimum error (ϵ_t), where $\epsilon_t = \sum_{j=1}^{N} w_j^t l_j^t$, and $l_j^t = 1$ if C_t misclassifies x_j; otherwise, $l_j^t = 0$.

6: **while** $\epsilon_t >= 0.5$ **do**

7: Reinitialize the weights to $w_j^t = \frac{1}{N}, j = 1, \ldots, N$.

8: Recalculate ϵ_t.

9: **end while**

10: Compute the weight of each weak learner (α_t) as follow, $\alpha_t = \frac{\epsilon_t}{1-\epsilon_t}$.

11: Update the weights of the training samples to be used in the next iteration $(t + 1)$ as follows:

$$w_j^{t+1} = \frac{w_j^t \alpha_t^{(1-l_j^t)}}{\sum_{i=1}^{N} w_i^t \alpha_t^{(1-l_i^t)}}, \; j = 1, 2, \ldots, N \quad (9)$$

12: **end for**

13: Final AdaBoost classifier: $H_{final} = \sum_{t=1}^{T} \alpha_t C_t(x)$.

weight of current weak learner, ($\alpha_t \in (0,1)$), is then calculated. As shown in the algorithm (step number nine), increasing the error rate increases the weight of the weak learner (α_t). The weights of the training samples are then updated at the end of each iteration to be used in the next iteration (this can be seen at the 10^{th} step of the algorithm). As shown in Equation (9), if the j^{th} sample is misclassified then $l_j^t = 1$; otherwise $l_j^t = 0$. Since, the weight of the weak learner (α_i) is less than one, thus the new weights (w_j^{t+1}) of the correctly classified samples will be decreased; otherwise the weights will be increased. In each iteration, the AdaBoost will focus on the misclassified patterns and the procedure is repeated for many iterations until the performance is satisfied (Kuncheva, 2014).

To classify an unknown sample (x_{test}), all weak learners of the AdaBoost classifier are used as shown in Equation (10). The score of each class is calculated and then assigns the class that has a maximum score to the unknown sample.

$$\mu_t = \sum_{C_t(x_{test})=\omega_t} ln(\frac{1}{\alpha_t}) \ , \forall \ t = 1, 2, \dots, T \tag{10}$$

where T represents the maximum number (a positive integer) of the iterations and it ranges from a few dozen to a few thousand, $C_t(x_{test})$ denotes the weak learner, μ_t represents the score of a class ω_t, and α_t refers to the weight of the t^{th} weak learner.

The performance of the AdaBoost algorithm is controlled by a parameter called *Learning rate*, (λ), or step size which is a numeric value ranged from 0 to 1. This parameter determines how fast or slow the algorithm will move towards the optimal solution. If λ is large, the algorithm accuracy may oscillate around the optimal solution without reaching to it. If λ is too small, there is a need for many iterations to converge to the optimal solution. More discussions about AdaBoost parameters are given in Section 5.

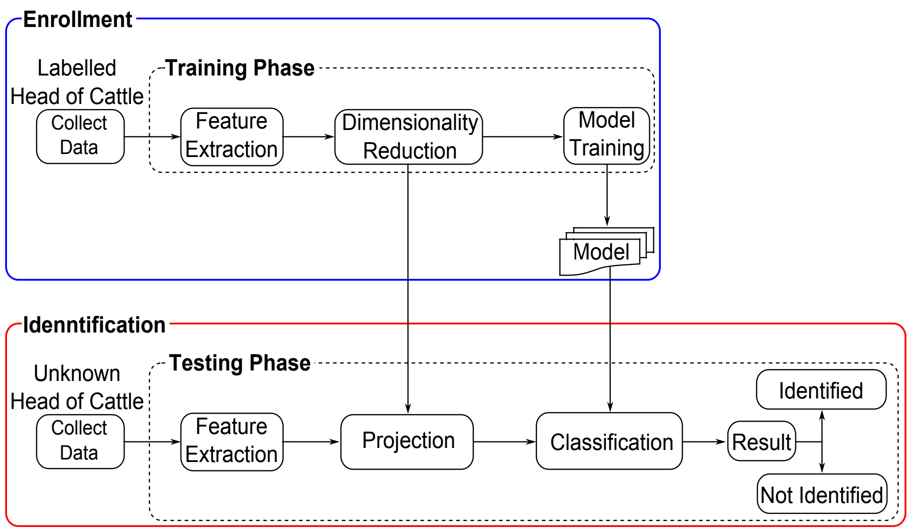


Figure 2: A block diagram of the proposed cattle identification system using muzzle print images.

3.3.2. Other Classifiers

k-Nearest Neighbor (Fix and Hodges Jr, 1951) and Fuzzy-k-NN (Keller et al., 1985) were also used to test the performance of the AdaBoost algorithm. The k-Nearest Neighbor (k-NN) is one of the oldest and simplest methods for pattern classification algorithms. It was first introduced by Fix and Hodges Jr (1951). The performance of the k-NN algorithm crucially depends on the distance metric to identify the nearest neighbors. Thus, the distance metric must be carefully chosen according to the problem being solved. The fuzzy k-NN (Fk-NN) classifier (Keller et al., 1985) is based on assigning a membership value to an unlabeled pattern. This value provides the system with information to determine a more accurate decision. Thus, the Fk-NN assigns a class membership to a test pattern rather than assigning the vector to a particular class.

4. Proposed Cattle Identification System

This section describes the proposed approach in detail. Generally speaking, the approach depends on using the WLD algorithm to extract robust features and then using the AdaBoost classifier to recognize the input muzzle print image of a given cattle. The approach, as illustrated in Figure 2, generally consists of three phases: feature extraction, feature reduction, and classification. These phases are explained below.

4.1. Feature Extraction Phase

The WLD algorithm, given in Algorithm 1 was adapted to achieve the feature extraction phase of the proposed approach. As shown in Figure 2, WLD was used to extract the features from all the training images in the training phase to construct a feature matrix. In the testing phase, the WLD also applied to extract the features from each an unknown or a test image. The extracted features are represented as a vector.

4.2. Feature Reduction Phase

The output of the *feature extraction phase* is usually a high dimension features vector (see Table 1). To use these features vectors in the classification/identification phase, there will be a high computational cost and time-consuming process, thus affecting the performance of the proposed approach. To address these issues, LDA algorithm, described in Section (3.2), was applied on the output of the feature extraction phase. In other words, the LDA was applied to the feature matrix which computed in the training phase to find the LDA space that reduces the dimension of the training data and separate different classes (head of cattle in this case). The feature vector of an unknown image was then projected on the LDA space to reduce its dimension before starting the classification phase.

4.3. Classification Phase

Finally, in the classification phase, the proposed system gives a decision about whether an input (i.e. unknown) muzzle image is for cattle previously stored in the database of the system or not. Generally, machine learning-based classifiers use a set of features in order to differentiate each object within a database. In this paper, a supervised learning classifier (AdaBoost) was used. As shown in the algorithm, the feature matrix, after projection onto the LDA space, and the labels of the training samples represent the input to the AdaBoost classifier. The AdaBoost classifier was then built by training one weak learner in each iteration and calculating the weight of that weak learner.

To automatically identify head of cattle from its muzzle image (i.e. an unknown cattle), all weak learners were used to classify the unknown image. The weighted voting method was then used to calculate the score of each class, and assign the class with the maximum score to the unknown image. Hence, the image is said to be identified. Otherwise, if all scores were lower than a threshold, then the image is said to be not identified.

5. Experimental Results

5.1. Dataset Description

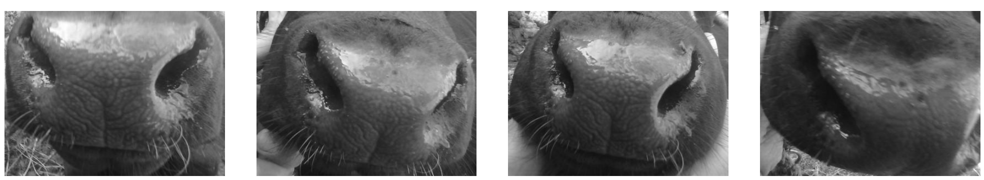


Figure 3: A sample of cattle images with different orientation of the same cattle.

The proposed cattle identification approach was evaluated using 217 gray level muzzle print images collected from 31 head of cattle (7 images for each head of cattle). These images were collected under different transformations: illumination, rotation, quality levels and image partiality. The size of all these images is 300×400 pixels, Figure 3 shows examples of these images. Moreover, these images were used without performing any preprocessing operation such as gray scaling, cropping, histogram equalization, etc. This was done to evaluate the robustness of the feature extraction algorithm. The dataset was randomly divided into two sets: training and testing. During the training phase, for each head of cattle, the number of training images was increased from 1, 2, 3, 4, 5, and 6 muzzle images whereas in the testing phase the remaining images (one muzzle image) of this head of cattle was used.

5.2. Experiment Setup

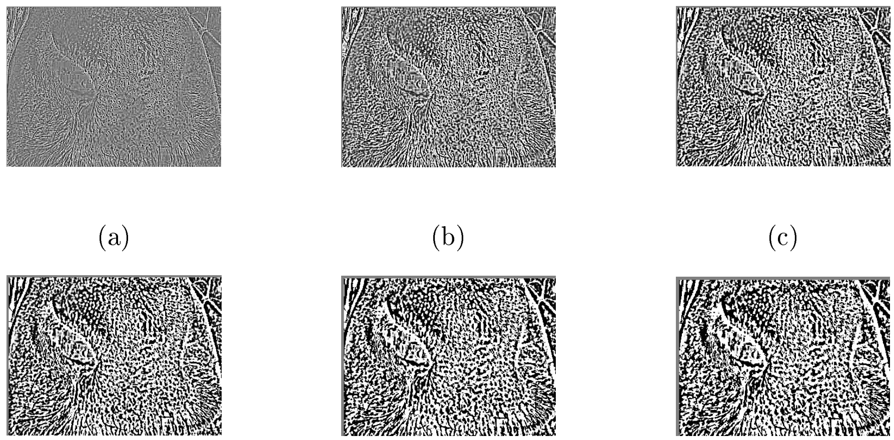
The experiments in this paper were conducted using a PC with Intel(R) Core(TM) i5-2400 CPU @ 3.10 GHz, and 4.00 GB RAM. The Matlab platform was used and it was run under windows 32-bit operating system. Prior to evaluating the proposed approach, we run a number of pre-experiments to tune up the parameters of all algorithms that are used in the proposed approach. The following subsections explain the tuning process of these parameters and their impact on the results presented in Section 5.

5.2.1. Parameters Tuning

In our approach, there are different parameters affecting the overall results. In this section, an overview of the parameters configured during the different phases of our approach is given. This includes WLD parameters used in the feature extraction phase, and AdaBoost, k-NN, and Fk-NN classifiers used in the classification phase.

5.2.1.1. ***WLD Parameters.*** The patch size is a very important parameter affecting the accuracy and CPU time of the WLD algorithm. A number of experiments, using different patch sizes for WLD, were conducted to investigate the impact of the WLD patch size on the cattle identification rate. Figure 4 shows WLD features extract using different patch size. The features extracted from each experiment were then used for the classification using the AdaBoost, k-NN, and Fk-NN classifiers to evaluate the identification rate. Table 1 summarizes the identification rate and the CPU time obtained when different patch sizes were used.

5.2.1.2. ***AdaBoost Parameters.*** The tuning of AdaBoost parameters (weak learners type, number of weak learners (iterations), and learning rate (λ)) used in our proposed approach are explaining in this section.



(a) (b) (c) (d) (e) (f)

Figure 4: WLD features using different patch sizes, (a) 3×3, (b) 5×5, (c) 7×7, (d) 9×9, (e) 11×11, (f) 13×13.

Table 1: Length of feature vector, CPU time, and identification rates (in %) of head of cattle using WLD features using different training images and different sizes' of sub-images.

Patch size	**No. of Training Images**						**Length of Feature Vector**	**CPU Time (Secs)**
	6	**5**	**4**	**3**	**2**	**1**		
3×3	96.8	96.8	94.6	92.7	92.9	80.1	119301	0.54934
5×5	**100**	96.8	**98.9**	92.7	**93.6**	**85.5**	118604	0.5437
7×7	**100**	**98.4**	97.9	92.7	89.7	74.7	117909	0.524767
9×9	93.6	93.6	92.7	92.7	81.3	84.4	117216	0.5245
11×11	96.7	96.8	93.6	90.3	88.4	71	116525	0.521
13×13	93.6	96.8	89.3	90.3	86.5	83.3	115836	0.5153

Bold fonts indicate best identification rate within each number of training images.

- **Type of Weak Learners:** To evaluate the effect of this parameter on the results of our approach, a number of experiments were conducted using two types of weak learners: *Tree, and Discriminant.* As shown in Figure 5, the results of these experiments showed that the error rate of the Discriminant learner is less than that of the Tree learner. These results were obtained when $\lambda = 0.1$ (default value), and the number of weak learners was 200. Also, the results presented in Table 2 shows that the Discriminant learner reached to the minimum error more faster than the Tree learner did.

Table 2: A comparison between the CPU time of the AdaBoost classifier when using Discriminant and Tree learner where (λ)=0.1, and the number of weak learners =200.

Type of Weak Learner	CPU Time (Secs)
Discriminant	0.20605
Tree	0.86898

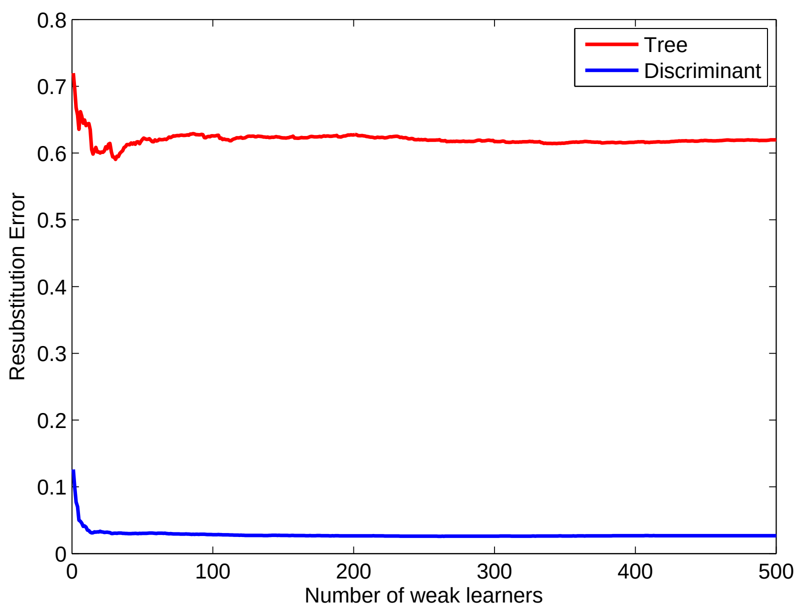


Figure 5: Resubstitution error curves of AdaBoost classifier using two types of weak learners, Tree and Discriminant, where the learning rate=0.1.

- **Number of Weak Learners:** To tune this parameter, a number of experiments were run to investigate its effect on the resubstitution error[3].

[3]The resubstitution error is the error rate obtained from running an algorithm on the

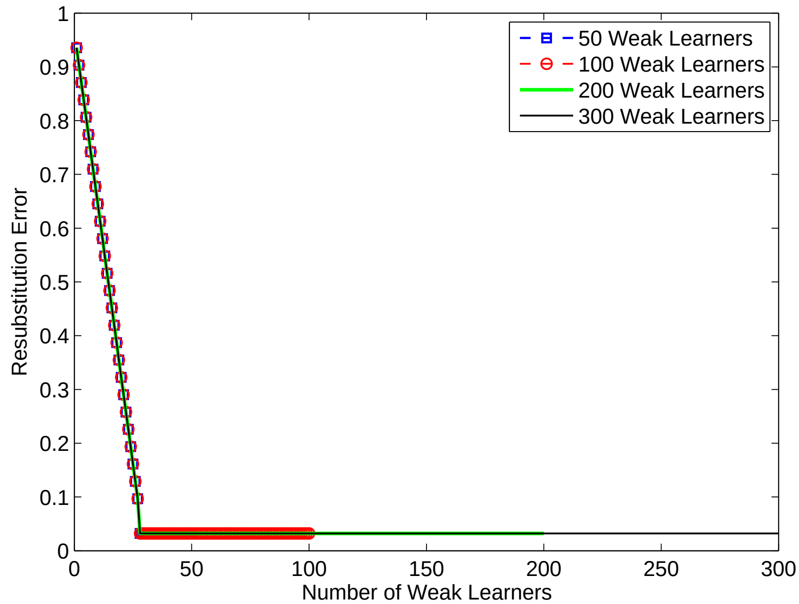


Figure 6: Resubstitution error curves of AdaBoost classifier using different numbers of weak learners (iterations), at learning rate=0.1, and the type of learner is Decision Tree.

The results of these experiments are shown in Figure 6 from which it can be seen that, when choosing 50, 100, 200 and 300 weak learners, the resubstitution error is approximately 0.19, 0.16, 0.13, and 0.12, respectively. These results were obtained when *the learning rate=0.1* and the type of the weak learner was the *Tree learner*. It can also be noticed that, when the number of the weak learners was increased, the accuracy was also increased until it reached an extent at which increasing the number of the learners did not affect the accuracy. On the contrary, the CPU usage time was increased without achieving noticeable progress in the accuracy (this is summarized in Table 3).

From Figure 6 and Table 3, it can be concluded that: (1) when using 200

training data

and 300 weak learners for the AdaBoost classifier, the difference of the error rate is small, (2) the error rate is approximately stable starting from 200 Tree learners to 300 Tree learners, and (3) the running time, using 300 iterations, is higher than that of using 200 iterations.

Table 3: The CPU time of the AdaBoost classifier when using a different number of iterations, when the weak learner is Tree and (λ)=0.1.

Number of Weak Learners	**Time (Secs)**
50 Weak Learners	0.2364
100 Weak Learners	0.44583
200 Weak Learners	0.9245
300 Weak Learners	1.36194

- **Learning Rate (λ):** To tune this parameter, some experiments were conducted at different values of λ while the other parameters were Tree learner, and the number of the iterations = 200. The results of these experiments are illustrated in Figure 7. This figure shows that the AdaBoost classifier with low learning rates (0.05 and 0.01) resulted in high error values. The reason behind this is that the classifier with a low learning rate takes more iterations to reach the optimal solution. Moreover, it can be remarked that increasing the learning rate (0.5 and 0.8) made the error rate fluctuated up and down more than other learning rates until it reached to the minimum error rate and the classifier, in this case, maybe not stable and will not reach to the minimum error. Moreover, Table 4 shows that the CPU time, taken by the AdaBoost classifier with different learning rates, was approximately the same when the same number of iterations was used.

5.2.1.3. ***k-NN and Fk-NN Parameters.*** Both of k-NN and Fk-NN classifiers may have different values of k. This value is always odd value to enable the voting to be smaller than the number of training images in each class (head of cattle). For example, if the number of the training images of each class is three,

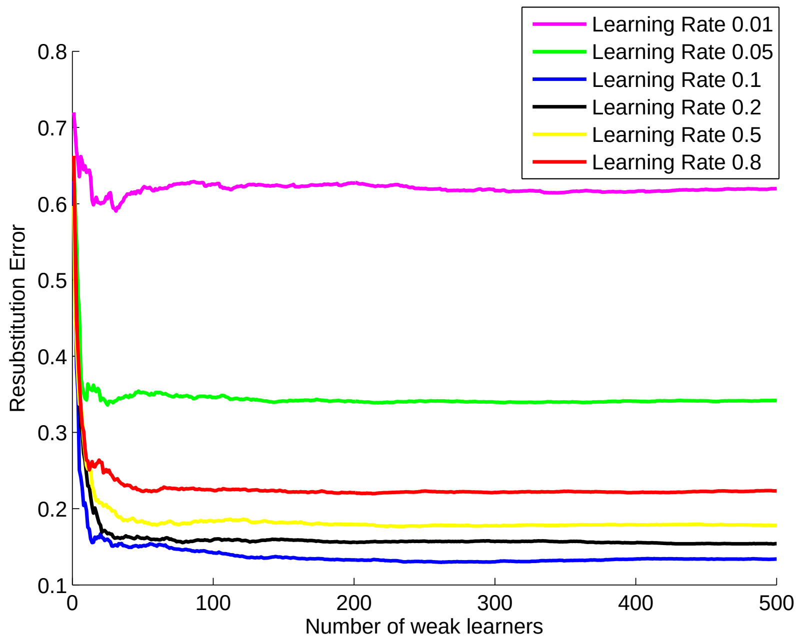


Figure 7: Resubstitution error curves of AdaBoost classifier when using different learning rates, Decision Tree learner, and the number of iterations are 200.

Table 4: The CPU time of AdaBoost classifier when using different learning rates, while Tree learner and 200 iterations were used.

Learning Rate (λ)	**Time (Secs)**
$\lambda = 0.8$	0.8933
$\lambda = 0.5$	0.8984
$\lambda = 0.2$	0.8772
$\lambda = 0.1$	0.8328
$\lambda = 0.05$	0.88179
$\lambda = 0.01$	0.856

thus it does not make sense to set k =7. If this happens, the k-NN classifier will select the nearest seven objects and make a vote on it to determine the class label of an unknown pattern, but this is not true as there are four objects out of seven are wrong. To investigate this, some experiments were run to check the accuracy and the CPU time under different values of k. Table 5 summarizes the results of these experiments. It can be noticed that the accuracy of k-NN and Fk-NN classifiers were the same and it decreased when the value of k decreased. In addition, when increasing k, the CPU time were slightly increased in both classifiers.

Table 5: Recognition rate and CPU time of k-NN and Fk-NN classifiers using different k values and using six training images.

Classifier	Recognition Rate (in %)			CPU Time (Secs)		
	k=1	k=3	k=5	k=1	k=3	k=5
k-NN	96.77	100	100	0.0749	0.0779	0.0814
Fk-NN	96.77	100	100	0.07818	0.0818	0.085

5.3. Experimental Scenarios and Their Results

Three experimental scenarios were designed to evaluate our proposed approach. The aim of the first scenario was to investigate the accuracy of our approach when changing the number of the training images. The second and the third scenarios were designed to test the robustness of the approach against rotation and occlusion, respectively. The second and third scenarios were considered because of the following reason. Firstly, as reported in (Dahlborn et al., 2013), the animals need to be restrained when mechanical or electrical methods are used, while using biometric-based identification no need to restrain animals. Secondly, unlike the human case, the animals are not fully controlled, thus the captured images may be rotated in different angles or partially occluded. Considering these issues, the proposed approach investigated their potential effective on the accuracy of the cattle identification. In all experiments, three classifiers, AdaBoost, k-NN, and Fk-NN, have been applied to the features extracted using the WLD algorithm. The AdaBoost was used with parameters: *learning rate=0.1*, *Discriminant learners = 200*, and both k-NN and Fk-NN were used with the parameter k=5.

In the first scenario, AdaBoost, k-NN, and Fk-NN, were used to (1) understand the effect of changing the number of training data on the identification accuracy and (2) evaluate the performance stability over the standardized data. The number of training images was ranged from one to six images. Table 6 and Figure 8 summarize the identification rate and CPU time obtained from this scenario.

Table 6: Identification rates (in %) and CPU time of the proposed approach using AdaBoost, k-NN, Fk-NN classifiers. The rate was calculated for different number of training images while the CPU time was computed when four training images were used.

Classifiers	**No. of Training Images**						**CPU Time (Secs) using**
	6	**5**	**4**	**3**	**2**	**1**	**(four Training Images)**
AdaBoost	100	96.8	98.9	92.7	93.6	85.5	0.27
Fk-NN	100	96.8	97.9	92.7	92.4	85.5	0.04781
k-NN	100	95.2	96.8	92.7	91.2	84.3	0.27

In the second scenario, testing against image rotation, the training and testings images consist of four and three images, respectively. The testing images were rotated in the following angles: ($0°$, $15°$, $30°$, $45°$, $-15°$, $-30°$, $-45°$) as shown in Figure 9. The rotated testing images were matched with the training images for the identification. Table 7 summarizes the results obtained from this scenario.

In the third experiment scenario, testing against the image occlusion, the used images were four and three for the training and the testing, respectively. As depicted in Figure 10, the testing images were first occluded, vertically and horizontally with different percentages, and used for the identification. Table 7 summarizes the results obtained from this scenario.

6. Discussion

This section introduces a reasoning and discussion about the results presented in Section 5.

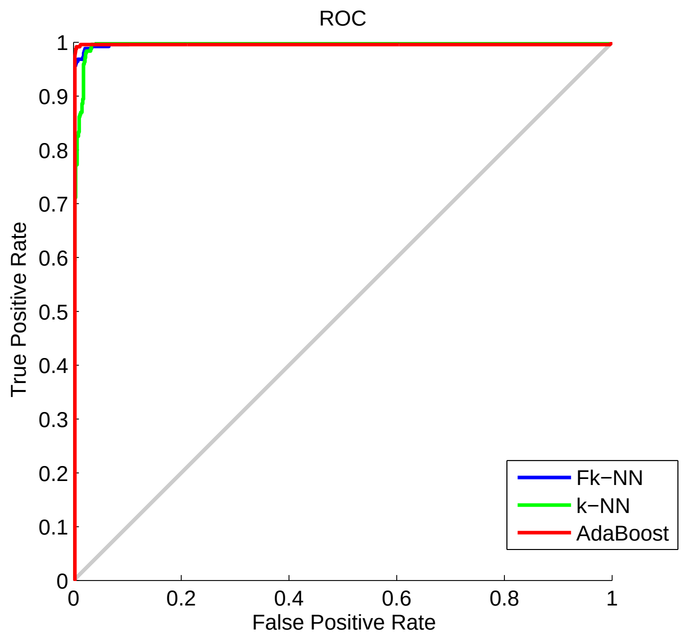


Figure 8: ROC curves for cattle identification based on AdaBoost, Fk-NN, and k-NN classifiers using four training images.

Table 7: Accuracy (in %) of cattle identification when muzzle print images were rotated in different angles and occluded in different percentages.

Classifier	**Angles of Rotation (°)**							**Percentage of Occlusion (%)**			
								Vertical		**Horizontal**	
	0	**15**	**30**	**45**	**-15**	**-30**	**-45**	**10**	**20**	**10**	**20**
AdaBoost	98.9	95.7	93.6	89.2	97.6	94.6	92.5	96.8	94.69	95.7	93.6
***k*-NN**	96.8	94.6	92.5	86	96.8	94.6	88.2	94.6	91.4	94.6	92.5
F*k*-NN	97.9	94.6	93.6	88.2	95.7	94.6	89.3	94.6	92.5	95.7	92.5

6.1. Parameter Tuning

As described in Section 5.2, a number of experiments were run to determine the best parameters' values for all the techniques used in our approach. For the

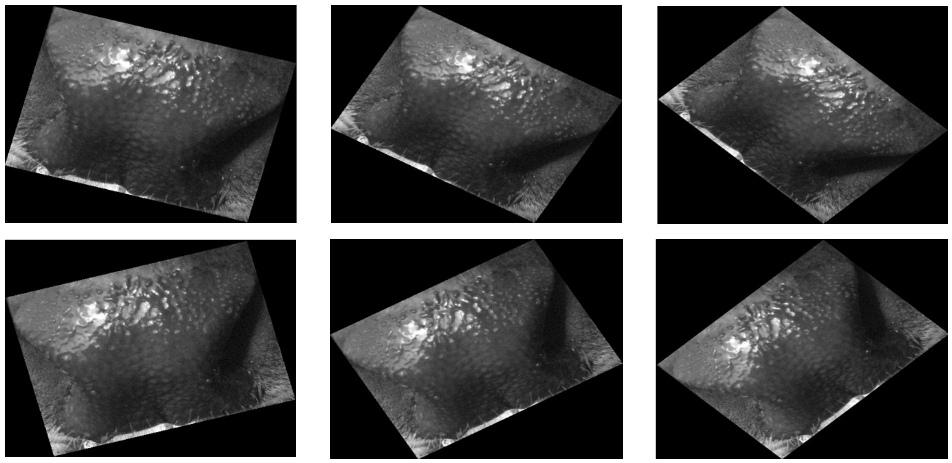


Figure 9: A sample of different images with different orientations of the same cattle.

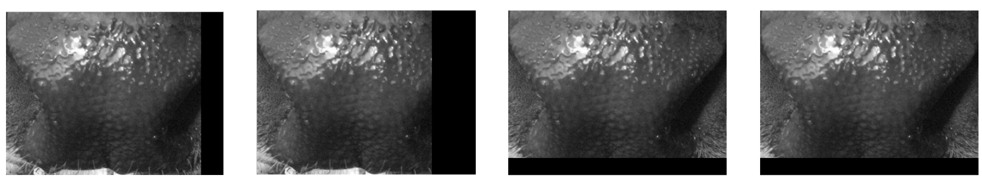


Figure 10: A sample of occluded muzzle print images, the top row (a and b) represents the vertical occlusion, while the bottom row (c and d) represents the horizontal occlusion.

WLD technique, based on the results described in Table 1, it was found that the most suitable size for the patch parameter was 7×7. This is because it allowed our approach to achieve an accuracy rate significantly better than the other sizes. Moreover, it can be noticed that increasing the patch size led to decreasing the length of the feature vectors, consequently decreasing the CPU time for classification. Thus, the 7×7 patch size did not take more CPU time comparing with the other patch sizes (e.g. 3×3 and 5×5).

Also, the patch size was affecting the length of produced features vectors. When it was changed from 3×3 to 13×13, as can be seen in Table 1, the length of the vectors ranged from 119301 to 115836 and this caused a high-dimension problem. Hence, the LDA was used to reduce such high dimensionality and further extracts more discriminative features.

For the AdaBoost classifier, the experiments, conducted to determine its best parameters for the accuracy and the CUP time (see Section 5.2.1), showed the following remarks. Firstly, the Discriminant weak learner was better than Tree weak learner as the former was faster than the latter in reaching the minimum resubstitution error. Secondly, the best accuracy rate and the least CPU time taken were achieved when the number of weak learners was *200 learners*. Thirdly, when the learning rate was decreased, more CPU time was taken to reach the optimal solution. Also, when the learning rate was increased, the error was ranged from up to down and the best learning rate was =0.1. For the k-NN and Fk-NN classifiers, as can be seen from the results described in Section 5.2.1, when the k parameter was changed from value to another, it did not affect the CPU time and the best accuracy was achieved when $k=3$ and $k=5$.

6.2. Experiment Scenarios Discussion

From the results of the first scenario, summarized in Table 6 and depicted in Figure 8, the following remarks can be drawn. Firstly, the features extracted by the WLD algorithm enabled our approach to achieve a very good identification rate using the three used classifiers. Secondly, using more training images led to a high recognition rate. This is very important to avoid the problem of a high variance[4]. As reported in (Brain et al., 1999), using more training images will decrease the variance, hence decreases the overfitting. Thirdly, the AdaBoost classifier achieved the best accuracy rate comparing with the k-NN and Fk-NN classifiers. Nonetheless, the AdaBoost took the highest CPU time which is not a problem nowadays due to the advance in the high-speed computers. The AdaBoost classifier achieved the highest accuracy because of two main reasons. (1) as mentioned in Section 3.3.1, the AdaBoost is an ensemble classifier consisting of other weak learners. Combining the outputs of all these classifiers may help to increase the accuracy while k-NN and Fk-NN are single classifiers.

[4]The variance is the error from sensitivity to small variations in training samples

(2) the AdaBoost classifier assigns high weights to the samples which are critical or misclassified during the iterations of AdaBoost classifier.

From the results of the second scenario, see Table 7, it can be claimed that our proposed approach is robust against image rotation. This is because when the images were rotated in different angles, the identification rate, achieved by the three classifiers, did not go below 86% and the AdaBoost classifier achieved the best recognition rate in all angles comparing with the other two classifiers.

Also, from the experimental results obtained from the third scenario and summarized in Table 7, it is proven that our approach is robust against image occlusion (10% and 20 % of the original image). Although this occlusion, the recognition rate of all the used classifiers was above 91%. Under 20% occlusion of the test images, horizontally or vertically, the best accuracy was achieved by the AdaBoost classifier. On the other hand, the k-NN classifier has given the lowest accuracy rate.

6.3. Assessment of the Results

To assess the results obtained by our proposed approach, four benchmark assessment methods (sensitivity and specificity, accuracy rate, Area Under Curve (AUC), and Equal Error Rate (EER)) were used. The results of these assessments are summarized in Table 8. From this table, the following remarks can be drawn. Firstly, as the sensitivity (i.e. True Positive Rate (TPR)) of the AdaBoost was better than both of the k-NN and Fk-NN classifiers, hence, the AdaBoost classifier could be used to correctly identify head of cattle. Secondly, both of the AdaBoost and Fk-NN classifiers achieved specificity (True Negative Rate (TNR)) better than that of the k-NN classifier. This means that the AdaBoost and Fk-NN are robust against unauthorized cattle identification. Thirdly, based on the value of the sensitivity and specificity of the three classifiers, see Table 8, and the AUC shown in Figure 8, the AdaBoost classifier along with the WLD is better to be used for cattle identification. Last but not least, based on the EER[5] results given in Table 8, it can be concluded that the AdaBoost is a good classifier for cattle identification as it achieved the minimum EER compared with k-NN and Fk-NN classifiers.

Table 8: A comparison between AdaBoost, Fk-NN, and k-NN classifiers based on different assessment methods (four training images were used).

Assessment Methods	**AdaBoost**	**Fk-NN**	**k-NN**
Accuracy (AC) (in %)	98.9	97.9	96.8
Sensitivity (TPR)	0.9841	0.9683	0.9683
Specificity (TNR)	0.9836	0.9836	0.9672
Area Under Curve (AUC)	0.983	0.976	0.969
Equal Error Rate (EER)	0.0035	0.0046	0.0073

6.4. Performance Analysis

The performance of the proposed approach was evaluated using two ways: the CPU time to get the results and a comparison with the most related work.

For the CPU time, from Table 6, it can be noticed that the AdaBoost took the highest CPU time. This is due to the fact that this algorithm needs to run 200 weak learners on each cattle image and then combines the results of these weak learners to get the final result. However, as discussed above, the best results were obtained when the AdaBoost was used. In addition, thanks to the advance in the parallel computing and the super-computing, this issue could be addressed in the real-time implementation.

To further prove that our approach is better than other related work, as illustrated in Table 9, a comparison with the most related work (Minagawa et al., 2002; Noviyanto and Arymurthy, 2012; Awad et al., 2013) was conducted. From this table, it can be remarked that although our approach used the largest dataset (217 images), at the same time it achieved the best accuracy results.

[5] The EER represents the failure rate when FPR and TNR are approximately the same

This is because of two reasons: the use of the WLD algorithm which extracts discriminative features (WLD algorithm is discussed in more detail in Section 3.1) and the strong AdaBoost classifier.

Table 9: A comparison between our proposed cattle identification method and some of state-of-the-art methods in terms of, identification accuracy, size of database images, and feature extraction methods.

Authors	**Feature Extraction Method**	**Database Images**	**Results**
(Minagawa et al., 2002)	Joint Pixels	43 images	30%
(Noviyanto and Arymurthy, 2012)	SURF	15 images for each animal	90%
(Awad et al., 2013)	SIFT	15 animals (6 images each)	93.3%
Our Proposed Approach	WLD	31 animals (7 images each)	99%

6.4.1. WLD vs LBP vs SIFT

As mentioned in Section 1, there are two main methods to extract local invariant features: dense and sparse methods. To justify why WLD was chosen as a feature extraction technique in this work, a comparison between two dense methods: LBP and WLD, is presented. Another comparison between WLD and SIFT is conducted to show the difference between the dense and sparse methods.

WLD vs LBP: The WLD is different from the LBP in three ways. Firstly, the WLD is more robust than LBP against image rotation. This is because the LBP algorithm firstly builds statistics on the local patterns while the WLD firstly computes the salient patterns and then builds statistics on these salient patterns with the gradient orientation of the current pixel. In other words, the WLD algorithm not only concentrates on the position or statistics of the patterns (differential excitation), but also computes the orientation gradient of each pixel and then combines the differential excitation and the orientation into a WLD histogram. On the other hand, the LBP calculates only statistics about the local patterns without taking orientation into its consideration. Hence, the WLD is more robust against rotation than LBP. Secondly, WLD is more efficient than LBP against noisy pixels and illumination changes. This occurs because the LBP codes are calculated by comparing the pixels with their surrounding pixels, while, in the WLD, the ratio of the intensity differences to the current pixel is calculated as in Equation (4). For this reason, WLD reduces the influence of noisy pixels as well as the effects of illumination change as reported in (Chen et al., 2010). Thirdly, the time complexity of LBP is simpler than WLD. As reported in (Chen et al., 2010), the time complexity for WLD is $O(C_1mn)$ while the time complexity for LBP is $O(C_2mn)$, where m and n are the dimensions of the image, C_1 is a constant and it represents the computation of each pixel in WLD, and C_2 is a constant and it represents the computation of each pixel in LBP. The computation of C_1 in WLD consists of several additions, divisions, and filtering with arctangent function, while C_2 in LBP consists of only several additions. Hence, LBP is a little faster than WLD. However, using the supercomputer and the parallel computing, the time complexity is not a problem as long as WLD could give a high accuracy.

WLD vs SIFT: The WLD is better than the SIFT in three ways. Firstly, WLD is robust than SIFT to capture local features. This is because SIFT algorithm extracts the features around the selected keypoints while, in the WLD algorithm, the features are extracted from each pixel. This means that WLD is able to capture more local salient features and identify small objects and patterns (i.e. more efficient). Secondly, WLD has only the patch size parameter that needs to be tuned to improve the robustness of WLD. While in SIFT algorithm, there are many parameters (peak threshold, the number of angles, and the number of bins, levels of scale space) which need to be tuned (Lowe, 1999; Noviyanto and Arymurthy, 2013). Thirdly, the time complexity of WLD is more efficient than SIFT. As reported in (Chen et al., 2010), the time complexity for SIFT is computed using, $O(C_1(\alpha\beta)mn + C_2k_1 + C_3k_2st + C_4k_2st)$, where $C_1, C_2, C_3,$ and C_4 represent four constants, k_1 is the number of keypoint candidates, k_2 is the number of keypoints, s and t refer to the size of the support regions for each keypoint, and α and β are the levels of octave [6] and scales of each octave, respectively. Comparing the time complexity of SIFT and WLD, descried earlier, it can be seen that WLD is more efficient than SIFT.

6.5. Further Discussion

When using a large cattle database images, it is expected that our approach would be suitable to highly identify head of cattle. This is due to the fact that the cattle muzzle pattern is much similar to the human fingerprint pattern mentioned (Baranov et al., 1993). Also, the WLD was used in (Gragnaniello et al., 2013) to detect the human liveness using a large dataset of human fingerprint images. Therefore, it is expected that our proposed approach, using the WLD, would also be able to identify head of cattle in case of using a large data set of cattle muzzle images.

Head of cattle could also be identified using dynamic frames (video) to support real-life scenarios in a farm. The dynamic frames have been used to identify human though capturing different biometrics, such as face and gait biometrics, which were then fused using independent biometric methods to improve the accuracy (Zhou and Bhanu, 2006; Liu and Sarkar, 2007). Similarly, video frames could be utilized to identify head of cattle to improve the accuracy. This could be achieved by applying fusion approach on different types of biometric, such as face, muzzle print, and retina. It is expected that integrating the video frame and the fusion approach could support the nature (uncontrollability) of the animals during the identification process real-time scenarios. This further could be also used for tracing animals activities such as eating, drinking, and movement, or any behavior change.

[6] Octave is a scale space. For example, the first octave starts with the original dimension of the image, and the scale of the image will be one-half in the next octave and so on (Lowe, 1999).

7. Conclusion and Future Work

In this paper, a new approach for cattle identification using muzzle print images was proposed. This approach used the Weber Local Descriptor (WLD) to extract texture features which are robust against rotation, noise, and illumination. It also utilized the LDA algorithm to reduce the dimensions of feature vectors and to increase the discrimination between different classes (head of cattle). Three classifiers (AdaBoost, k-NN, and Fk-NN) were used to achieve the cattle identification. The parameters of used techniques were first tuned to determine the ones achieving the best results in terms of accuracy and performance. The experimental results obtained when the WLD has patch size $= 7 \times 7$, the AdaBoost has Discriminant weak learner, 200 weak learners, and learning rate $= 0.1$, and $k = 5$ for both of the k-NN and the Fk-NN classifiers. Using these parameters and four training images, the best classifier was the AdaBoost achieved $\approx$99% accuracy whereas the k-NN gave the minimum accuracy. The results were assessed using different methods (sensitivity, specificity, AUC, and EER). Moreover, the sensitivity, specificity, and AUC of the proposed approach were approximately 0.9841, 0.9836, and 0.983, respectively, which reflects the robustness of the proposed approach. In addition, the proposed approach achieved a low error rate (≈ 0.0035). Furthermore, the results of the proposed approach were proven to be superior to the most related work. In the future work, our approach will be evaluated against a larger database of cattle images. Also, we will investigate the idea of fusing two cattle biometrics: muzzle and face.

8. ACKNOWLEDGMENT

This paper has been elaborated in the framework of the project "New creative teams in priorities of scientific research", reg. no. CZ.1.07/2.3.00/30.0055, supported by Operational Program Education for Competitiveness and co-financed by the European Social Fund and the state budget of the Czech Republic and supported by the IT4Innovations Center of Excellence project (CZ.1.05/1.1.00/02.0070), funded by the European Regional Development Fund and the national budget of the Czech Republic via the Research and Development for Innovations Operational Program and by Project SP2015/146 Parallel processing of Big data 2 of the Student Grant System, VSB Technical University of Ostrava.

Adell, N., Puig, P., Rojas-Olivares, A., Caja, G., Carné, S., Salama, A. A., 2012. A bivariate model for retinal image identification in lambs. Computers and Electronics in Agriculture 87 (0), 108 – 112.

Allen, A., Golden, B., Taylor, M., Patterson, D., Henriksen, D., Skuce, R., 2008. Evaluation of retinal imaging technology for the biometric identification of bovine animals in northern ireland. Journal of Livestock science 116 (1), 42–52.

Awad, A. I., Zawbaa, H. M., Mahmoud, H. A., Nabi, E. H. H. A., Fayed, R. H., Hassanien, A. E., 2013. A robust cattle identification scheme using muzzle print images. In: Proceedings Federated Conference on Computer Science and Information Systems (FedCSIS), Kraków, Poland. IEEE, pp. 529–534.

Baranov, A., Graml, R., Pirchner, F., Schmid, D., 1993. Breed differences and intra-breed genetic variability of dermatoglyphic pattern of cattle. Journal of Animal Breeding and Genetics 110 (1-6), 385–392.

Barry, B., Corkery, G., Gonzales-Barron, U., Donnell, K. M., Butler, F., Ward, S., 2008. A longitudinal study of the effect of time on the matching performance of a retinal recognition system for lambs. Computers and Electronics in Agriculture 64 (2), 202 – 211.

Bowling, M., Pendell, D., Morris, D., Yoon, Y., Katoh, K., Belk, K., Smith, G., 2008. Review: Identification and traceability of cattle in selected countries outside of north america. The Professional Animal Scientist 24 (4), 287–294.

Brain, D., Webb, G., Richards, D., Beydoun, G., Hoffmann, A., Compton, P., 1999. On the effect of data set size on bias and variance in classification learning. In: Proceedings of the Fourth Australian Knowledge Acquisition Workshop, University of New South Wales. pp. 117–128.

Chen, J., Shan, S., He, C., Zhao, G., Pietikainen, M., Chen, X., Gao, W., 2010. Wld: A robust local image descriptor. IEEE Transactions on Pattern Analysis and Machine Intelligence 32 (9), 1705–1720.

Corkery, G., Gonzales-Barron, U. A., Butler, F., McDonnell, K., Ward, S., 2007. A preliminary investigation on face recognition as a biometric identifier of sheep. Transactions of the ASABE 50 (1), 313–320.

Dahlborn, K., Bugnon, P., Nevalainen, T., Raspa, M., Verbost, P., Spangenberg, E., 2013. Report of the federation of european laboratory animal science associations working group on animal identification. Laboratory animals 47 (1), 2–11.

Fix, E., Hodges Jr, J. L., 1951. Discriminatory analysis-nonparametric discrimination: consistency properties. Tech. rep., DTIC Document.

Gonzales Barron, U., Corkery, G., Barry, B., Butler, F., McDonnell, K., Ward, S., 2008. Assessment of retinal recognition technology as a biometric identification. Journal of Computers and Electronics in Agriculture 60 (2), 156–166.

Gragnaniello, D., Poggi, G., Sansone, C., Verdoliva, L., 2013. Fingerprint liveness detection based on weber local image descriptor. In: IEEE Workshop on Biometric Measurements and Systems for Security and Medical Applications (BIOMS), 2013. IEEE, pp. 46–50.

Jiménez-Gamero, I., Dorado, G., Muñoz-Serrano, A., Analla, M., Alonso-Moraga, A., 2006. Dna microsatellites to ascertain pedigree-recorded information in a selecting nucleus of murciano-granadina dairy goats. Small Ruminant Research 65 (3), 266–273.

Keller, J. M., Gray, M. R., Givens, J. A., 1985. A fuzzy k-nearest neighbor algorithm. IEEE Transactions on Systems, Man and Cybernetics SMC-15 (4), 580–585.

Kuncheva, L. I., 2014. Combining pattern classifiers: methods and algorithms. John Wiley & Sons.

Liu, Z., Sarkar, S., 2007. Outdoor recognition at a distance by fusing gait and face. Image and Vision Computing 25 (6), 817–832.

Lowe, D. G., 1999. Object recognition from local scale-invariant features. In: The proceedings of the seventh IEEE international conference on Computer vision, 1999. Vol. 2. Ieee, pp. 1150–1157.

Marchant, J., 2002. Secure animal identification and source verification. JM Communications, UK. Copyright Optibrand Ltd., LLC.
URL `http://www.optibrand.com/uploadedfiles/Animal_ID.pdf`

Minagawa, H., Fujimura, T., Ichiyanagi, M., Tanaka, K., 2002. Identification of beef cattle by analyzing images of their muzzle patterns lifted on paper. Publications of the Japanese Society of Agricultural Informatics 8, 596–600.

Noviyanto, A., Arymurthy, A. M., 2012. Automatic cattle identification based on muzzle photo using speed-up robust features approach. In: Proceedings of the 3^{rd} European Conference of Computer Science, ECCS, Montreux, Switzerland. pp. 110–114.

Noviyanto, A., Arymurthy, A. M., 2013. Beef cattle identification based on muzzle pattern using a matching refinement technique in the sift method. Journal of Computers and Electronics in Agriculture 99 (1), 77–84.

Ojala, T., Pietikainen, M., Maenpaa, T., 2002. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (7), 971–987.

Rojas-Olivares, M., Caja, G., Carné, S., Salama, A., Adell, N., Puig, P., 2011. Retinal image recognition for verifying the identity of fattening and replacement lambs. Journal of animal science 89 (8), 2603–2613.

Roth, V., Steinhage, V., 1999. Nonlinear discriminant analysis using kernel functions. In: Advances in Neural Information Processing Systems. MIT Press, pp. 568–574.

Rusk, C. P., Blomeke, C. R., Balschweid, M. A., Elliott, S. J., Baker, D., 2006. An evaluation of retinal imaging technology for 4-h beef and sheep identification. Journal of Extension 44 (5), Feature Articles // 5FEA7.

Schroeder, T. C., Tonsor, G. T., 2012. International cattle {ID} and traceability: Competitive implications for the {US}. Food Policy 37 (1), 31 – 40. URL http://www.sciencedirect.com/science/article/pii/S0306919211001229

Shadduck, J., Golden, B., 2002. Retinal imaging in secure identification and source verification of livestock. In: Proceedings ID/INFO Expo 2002, National Institute for Animal Agriculture.

Shanahan, C., Kernan, B., Ayalew, G., McDonnell, K., Butler, F., Ward, S., 2009. A framework for beef traceability from farm to slaughter using global standards: An irish perspective. Journal of Computers and Electronics in Agriculture 66 (1), 62–69.

Tharwat, A., Gaber, T., Hassanien, A. E., Hassanien, H. A., Tolba, M. F., 2014. Cattle identification using muzzle print images based on texture features approach. Proceedings of the 5^{th} International Conference on Innovations in Bio-Inspired Computing and Applications IBICA 2014, 303, 217–227.

Zhou, X., Bhanu, B., 2006. Feature fusion of face and gait for human recognition at a distance in video. In: 18th International Conference on Pattern Recognition, 2006. ICPR 2006. Vol. 4. IEEE, pp. 529–532.