Local Binary Pattern and Edge Histogram Descriptor for Dynamic Hand Gesture Recognition

Salah Eddine AGAB, Fatma zohra CHELALI

Speech communication and signal processing laboratory,
Faculty of Electronic and computer sciences.
University of Sciences and Technology Houari Boumediene (USTHB)
Box n°: 32 El Alia, Algiers 16111, Algeria

E-mail : agab11@hotmail.fr , Chelali_zohra@yahoo.fr

Abstract **-** In recent years, hand gesture recognition has gained significant attention, researchers are now developing Human-Computer Interfaces that are simple, efficient, and do not require any intermediary device between user and computing environment. In this context, we present a vision-based dynamic gesture recognition system that uses a set of textural features to characterize the gestures, and two variants of Artificial Neural Network to predict the labels in classification phase. Experiments done on a user-independent database with a simple background show that used methods are efficient and perform well with higher accuracy than some previous works.

Keywords **-** Artificial Neural Network, CS-LBP, EHD, hand gesture recognition, LBP, LBP^{riu2}.

I. INTRODUCTION

In last decades, Human-Computer Interaction was mainly based on keyboards and mouses, but due to the development of software and hardware, these devices seem nowadays obsolete with an unintuitive interface. Eventually, researchers focused on designing interfaces capable of satisfying the expected performance and intuitiveness such as interfaces based on gestures [1]. Gesture recognition has become then a very active research area in computer science, which employs mathematical algorithms to understand human gesture and make Human-Computer Interaction (HCI) as simple and natural as Human-Human Interaction (HHI), gestures that originate from the hand or face convey lot of meaningful informations and are instinctively used by humans when communicating [2].

Early gesture recognition systems used wearable devices, such as data gloves to determine the position and shape of the hand, then transmit this data to the computing environment for processing, these methods are very efficient in characterizing hand gestures and achieve good recognition rates but they are expensive and require the user to wear special equipment whenever they want to interact with a computer [3]. However, vision-based approaches do not require any mechanical device, the hand gesture is directly captured using a camera, so users can interact with the computer without any contact, this approach allows natural and flexible interaction but it requires additional processing to detect and characterize the hand region, thus robust and fast methods are needed. Vision based hand gesture recognition can be used in many applications such as robot control, video games, navigating in virtual environment, change radio station in a car, or in medical systems by allowing the medical staff to use machines and interact with instruments without any contact which can minimize contamination and spread of an infection [4].

Hand gestures can be either static or dynamic, the static form, also called posture, is represented by single image containing the spatial aspect of the gesture such as the orientation and position of the hand, whereas the dynamic form is defined by a sequence of postures over time, so characterizing dynamic gestures is harder as it requires the study of both spatial and temporal aspects of the hand gesture [5]. Researchers developed many algorithms to characterize and classify hand gestures in both forms, in [6], authors implemented a real-time dynamic hand gesture recognition system using a Kinect camera, this system is composed of 3 main components, first is gesture spotting using skeleton tracking, second is features extraction where the dynamic gesture is represented according to the orientation feature, and the last component is classification phase using Support Vector Machine classifier, this system obtained 95.42% accuracy rate on a recorded dataset of 10 numbers and 26 alphabet letters. In [7], a dynamic gesture based system for robot navigation is proposed, the robot uses a RGB-D camera to capture both depth and color images, which are used for 3D hand tracking, the resulting motion trajectory is converted to a feature vector and classified using Hidden Markov Model, this system obtained a 98% recognition rate when tested on dataset containing 6 gestures to control the actions of robot performed by right hand and one gesture performed by left hand to control the speed of robot. In [8], authors proposed a gesture recognition system that is able to classify digits from 11 to 20 performed against uniform and complex backgrounds, the algorithm is based on chain code to capture the hand flow and create a motion curve in spatiotemporal volume, for classification, Principal Component Analysis was used to achieve 94% accuracy. Authors in [9] developed a new method for dynamic gesture recognition using the position, and angle of fingertips and finger roots, this system uses a skin color-based threshold for image segmentation which is YCbCr spatial threshold, the segmented image is characterized using edge detection, then, the coordinates of fingertip and finger roots are extracted from the contours map, finally, classification step is performed using a binary decision tree, this method was tested on a recorded database containing 9 dynamic gestures and obtained 97% accuracy. In [10], a performance comparison between appearance features such as Local Binary Pattern (LBP) and geometric moments (GM) for gesture recognition was performed, the classification phase used Hidden Markov Model, results show that LBP features performed better then GM, authors also investigated the combination of both features and added Principal Component Analysis to reduce the dimension of resulting features, this combination recognized all gesture from test data except one giving a recognition rate of 99.75%. In [11], a newly improved LBP is proposed with several advantages such as better illumination consistency and better robustness on constant regions, this descriptor was paired with Adaboost algorithm for classifying a recorded set of 12 gestures representing digits from 0 to 10 and "ok" gesture, this system proved to be very efficient and robust. In [12], authors presented a new implementation to improve the LBP operator by drawing radial lines from the center of the image at fixed angles, feature vector represents the histogram computed for each radial line drawn on the input image, experiments using SVM classifier show very good recognition rates for a recorded dataset where all gestures were correctly classified. In [13], a gesture recognition system based on Local Binary Pattern and Support Vector Machine classifier is proposed, the performance of this system was tested on NUS gesture database which contains 10 gestures with 24 repetitions, but to improve training efficiency, authors added 51 repetitions per class resulting in a total of 75 repetitions, the average recognition rate obtained was equal to 93%.

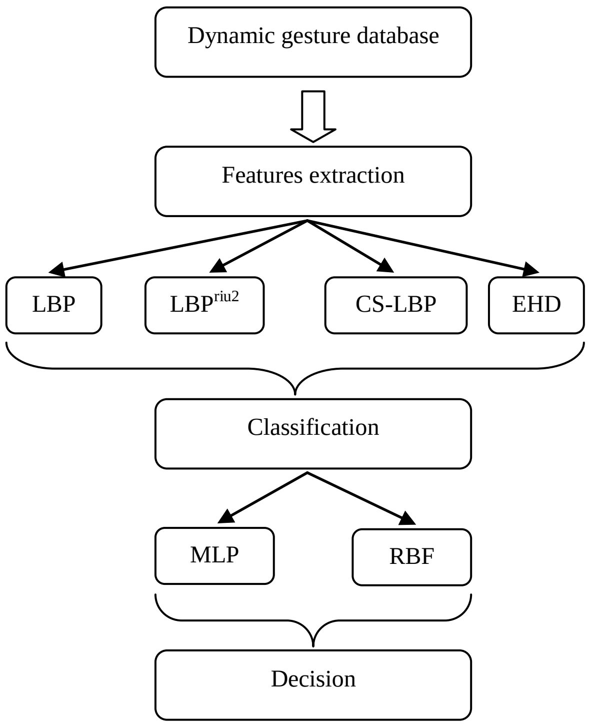


Fig. 1. Block diagram of the proposed gesture recognition system

In this paper, we present a recognition system for dynamic gestures in video sequences recorded using a single camera with low resolution, this dataset contains four distinct gestures repeated 12 times, and each repetition is performed by a different person, to characterize these gestures, we used mainly descriptors capable of retrieving textural features namely the Local Binary Pattern (LBP), Rotation Invariant and Uniform LBP (LBP^{riu2}), Center Symmetric LBP (CS-LBP), and Edge Histogram

Descriptor (EHD). For classification, we compared the performance given by two Artificial Neural Networks (ANN) which are MultiLayer Perceptron (MLP) and Radial Basis Function neural network (RBF). Fig.1 shows the main steps of the proposed system.

The contributions of this study consist of : (1) comparing the performance and speed of several texture-based descriptors for features extraction, and two ANN for classification, (2) illustrating the efficiency of LBP and LBP^{riu2} descriptors with only four neighbors, (3) demonstrating the performance gain for EHD when used with Sobel filters for edge detection step, (4) demonstrating the importance of applying texture-based descriptors to segmented images with grayscale foreground.

The remainder of this paper is structured as follows. Section II describes the theory of the methods used to characterize our gestures, MLP and RBF classifiers are presented in Section III. Section IV describes the dynamic gesture database, as well as, the obtained results, and finally, Section V concludes this paper.

II. FEATURES EXTRACTION

The characterization module generates a descriptor vector that describes the gesture using different types of features such as color, shape, texture or motion [14], this vector simplifies the discrimination between the sequences of all classes, the descriptors used in our paper extract textural features.

A) Local Binary Pattern (LBP)

LBP descriptor was presented by Ojala et al. in 1996; its main idea is to characterize each pixel of the image based on the grayscale values of its neighbors [15].

Initially, the image is divided into 3×3 overlapping blocks, and for each block, the eight neighbor pixels are thresholded by the central pixel, the resulting value is ‘1’ if the neighbor is superior and ‘0’ otherwise. The thresholded values are, then, multiplied by powers of two and summed to give the LBP code for that window. Fig. 2 summarizes the computation of the LBP code for 3×3 neighborhood [15].

Example

7	1	2
9	4	5
4	5	3

Binary

1	0	0
1	■	1
1	1	0

Weights

1	2	4
128	■	8
64	32	16

LBP = 1+8+32+64+128 = 233

Fig. 2. Computation of the LBP operator.

The LBP descriptor for the whole image represents the histogram accumulated from the LBP codes of all windows and its size is then 2^8=256.

The original LBP was later extended to support different number of neighbors by using a circular and symmetric neighborhood defined by the radius R and number of sampling points P [16] as shown in (1).

$$LBP_{P,R} = \sum_{P=0}^{P-1} s\left(g_p - g_c\right) \times 2^P \tag{1}$$

where

$$s(x) = \begin{cases} 1 \text{ if } x \geq 0 \\ 0 \text{ if } x < 0 \end{cases} \tag{2}$$

g_c represents the value of the central pixel and g_p represents the value of the neighbor pixel. The descriptor size given by this new LBP is equal to 2^P.

B) Rotation invariant and uniform LBP (LBP^{riu2})

The LBP^{riu2} was developed to provide rotation invariance with few spatial transitions and very compact histogram [16]. As for the original LBP descriptor, the neighbors are thresholded by the center pixel to give a binary matrix. If there are at most two transitions 0 to 1 or 1 to 0 then the patterns are considered as “uniform” and the LBP^{riu2} operator equals to the sum of all eight binary values as shown in (3).

$$LBP_{P,R}^{riu2} = \begin{cases} \sum_{0}^{P-1} s\left(g_p - g_c\right) & \text{if } U\left(LBP\right) \leq 2 \\ P+1 & \text{otherwise} \end{cases} \tag{3}$$

Where :

$$\begin{aligned} U\left(LBP\right) = \sum_{p=1}^{P} &\left| s\left(g_p - g_c\right) - s\left(g_{p-1} - g_c\right) \right| \\ &+ \left| s\left(g_{P-1} - g_c\right) - s\left(g_0 - g_c\right) \right| \end{aligned} \tag{4}$$

The LBP^{riu2} descriptor for the whole image represents the histogram accumulated from the LBP codes of all windows and its size is P+2. Fig. 3 shows two examples of computation of the LBP^{riu2}, the first with uniform patterns and the second with non-uniform patterns.

Example				Binary			
8	1	3		1	0	0	Uniform pattern
8	5	2		1	■	0	Binary code = 10001111
6	7	9		1	1	1	2 transitions so LBP^{riu2} = 5

Example				Binary			
7	1	2		1	0	0	Non-Uniform pattern
9	4	5		1	■	1	Binary code = 10010111
4	5	3		1	1	0	4 transitions so LBP^{riu2} = 9

Fig. 3. Computation of the LBPriu2 operator.

C) Center-Symmetric LBP (CS-LBP)

Heikkilä et al. introduced the CS-LBP in 2006. It provides many new features compared to the original LBP such as robustness on flat regions and to illumination changes and a shorter histogram [17].

Instead of comparing the neighbor pixels to the center pixel, the CS-LBP operator compares each pixel to the symmetrical pixel with respect to the center pixel, the result is then thresholded by a small value T, the operator is thus given by :

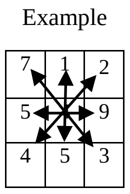
$$\text{CS-LBP}_{P,R} = \sum_{p=0}^{(P/2)-1} s\left(g_p - g_{p+(P/2)} - T\right) \times 2^p \tag{5}$$

Where :

$$s(x) = \begin{cases} 1 \text{ if } x \geq 0 \\ 0 \text{ if } x < 0 \end{cases} \tag{6}$$

g_p and $g_{p+(P/2)}$ represent the pairs of center-symmetric pixels. The histogram produced from the image is very compact ($2^P/2$) compared to the original LBP (2^P) but it can still provide very good image characterization [17].

The computation steps of the CS-LBP code for 3×3 neighborhood are presented in Fig. 4.



CS-LBP = 1+0+0+8 = 9

Fig. 4. Computation of the CS-LBP operator with T=0.

D) Edge Histogram Descriptor (EHD)

The EHD is a widely used method for shape detection, which represents the frequency of occurrence of five different types of edges in a local area of an image [18].

Initially, the image is divided into N×N sub-images, and for each sub-image, five histograms are generated for all five types of edges namely (horizontal, vertical, diagonals and non-directional) as shown in Fig. 5.

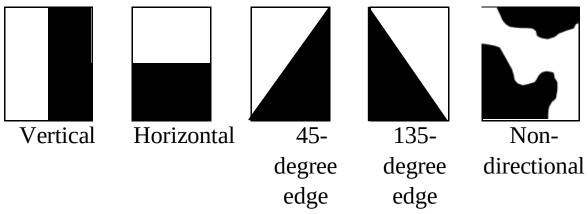


Fig. 5. The five edges used in the EHD.

The fastest method to extract the five histograms is to use digital filters approximating the five types of edges as shown in Fig. 6, so each sub-image is convolved with all five filters giving five edge strengths as demonstrated in (7) [18].

1	-1	1	1	$2^{½}$	0	0	$2^{½}$	2	-2
1	-1	-1	-1	0	$-2^{½}$	$-2^{½}$	0	-2	2
Vertical		Horizontal		45° edge		135° edge		Non-directional	

Fig. 6. Digital filters used for edge detection.

$$S = I * K \tag{7}$$

Where I represents the image, K represents one of the five digital filters and S the strength of the corresponding edge.

Once all edge strengths extracted from the image, five histograms representing five edges are accumulated for each sub-image, the concatenation of these histograms define EHD for the whole image and its size is then N×N×5.

All the histograms extracted using the descriptors described above characterize a single image but since we are working with a dynamic database and each sequence contains 55 frames, therefore, the descriptor vector characterizing a complete sequence is defined as the concatenation of individual histograms to simulate the temporal evolution of the frames. Fig. 7 shows a histogram example of a sequence extracted using the CS-LBP descriptor.

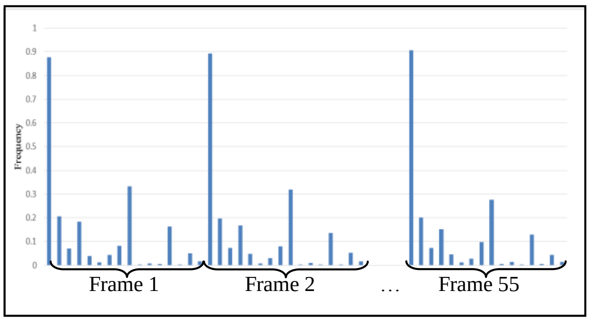


Fig. 7. Histogram of one sequence using CS-LBP descriptor.

III. ARTIFICIAL NEURAL NETWORK (ANN) FOR GESTURE RECOGNITION

ANN is a tool that uses mathematical operations to model the human's biological neurons; so basically, it is a web of interconnected processing elements 'PEs' that sum the signals using a nonlinear function to produce an output. This output becomes then the input for another PE [19].

Many ANN are used in literature for pattern recognition, in our paper, we opted for the MultiLayer Perceptron (MLP) and the Radial Basis Function (RBF) networks.

A) MultiLayer Perceptron (MLP)

The MLP consists of at least three layers : an input layer, a hidden layer constituted of nonlinear sigmoid PEs and an output layer constituted of linear identity PEs. The topology of the MLP is shown in Fig. 8.

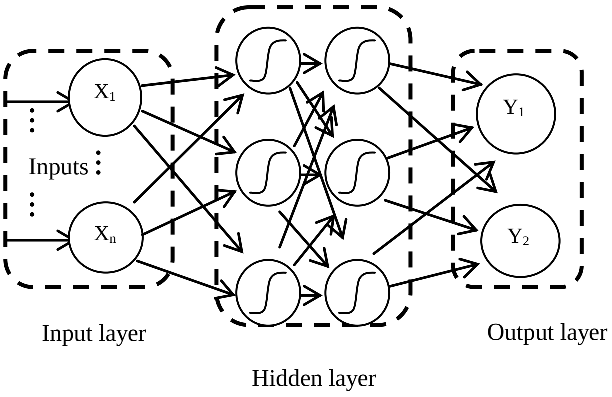


Fig. 8. MLP architecture with n inputs, two hidden layers, and two outputs.

Each PE calculates the sum of its weighted input signals and uses an activation function to map the results into the desired range [20] :

$$\mathrm{y} = \theta\left(\sum_{\mathrm{i}=0}^{\mathrm{n}} w_i x_i\right) \tag{8}$$

where y represents the output of the PE, x_i are the inputs, w_i are the weights, n is the number of inputs and θ is the activation function which can be linear, sigmoid or Gaussian [20].

- *Conjugate gradient backpropagation with Powell-Beale restarts*

The backpropagation approach is used to minimize the mean squared error 'MSE' between the desired output and the actual output of the PE by readjusting the weights and biases, the PE sums then the inputs with the new weights to give new output that is closer to the desired output, the error between the outputs is computed according to (9).

$$E(n) = \frac{1}{2}\sum_{j=1}^{m}\left(d_j(n) - y_j(n)\right)^2 \tag{9}$$

where m represents the total number of the output neurons, n is the current training sample, d and y are, respectively, the desired output and the actual output.

The conjugate gradient methods reset the search direction every N iterations where N is the number of variables (weights and biases), but to improve the training efficiency, Beale et al. proposed an approach that restarts when the orthogonality between two consecutive gradients is lost, so if (10) is satisfied, the search direction is restarted [21].

$$\left| g_{k-1}^{T} \times g_k \right| \geq 0.2 \| g_k \|^2 \tag{10}$$

where k is the current iteration and $g_k = \partial E(n) / \partial w_{ij}(n)$

B) Radial Basis Function (RBF)

Like the MLP, the RBF is a multilayer neural network but has only one hidden layer, also, the RBF network learns faster due to the absence of the iterative method [22].

The hidden layer is composed of RBF neurons that transform the inputs using a nonlinear activation function, each PE calculates the distance between the input and its center using a gaussian function as show in (12), the resulting value is bigger when the input is closer to the center [23]. This distance is defined by :

$$d(x) = \| x - \mu \| \tag{11}$$

$$\phi(x) = e^{-\left(\frac{d(x)}{2\sigma}\right)^2} \tag{12}$$

where x is the input vector, μ and σ are, respectively, the center and the width of the gaussian function.

The output layer calculates the linear combination of its inputs [23].

$$y(x)=\sum_{i=1}^{N} w_i \phi_i(x) \tag{13}$$

where x is the input vector, w_i are the weights of the connections between the hidden and the output layers and N is the number of the RBF neurons.

The RBF neural network uses hybrid learning where the gaussian parameters (center and width) are determined using unsupervised learning, then, the weights connecting the hidden and output layers are computed using supervised approach [23].

IV. SIMULATION RESULTS

We present in this section the simulation results obtained using the proposed gesture recognition system, all tests were performed on a laptop equipped with 4 GO of ram and i3-4030u processor.

The first task consists of extracting the feature vectors containing the relevant informations from our video sequences using different descriptors namely LBP, LBP^{riu2}, CS-LBP and EHD. Then, the recognition rate is computed using two classifiers, the MLP and the RBF neural networks. The next step consists of optimizing our classifiers and descriptors by finding the optimal values of the parameters to obtain the best recognition rate possible.

A) Dataset description

We used in our paper the dynamic gesture database available in [24] which contains 4 gestures as shown in Fig. 9, each class includes 12 video sequences performed by 12 different persons, and 55 frames were extracted from each sequence giving a total of 2640 images. All images have the same low resolution of 62×58 pixels, and all images have been converted from RGB to grayscale.

To evaluate the performance of our system, we used 6 sequences from each gesture to train our classifiers (MLP and RBF) and we used the 6 remaining sequences for the test phase.

B) Implemented technique

The first step consists of characterizing each sequence from our database using different descriptors described in section II. Each technique provides us with a unique vector with a different size from the other descriptors. For example, the CS-LBP produces a very compact histogram of 16 values per frame, so, one sequence of 55 frames is represented by 880 values, in comparison, basic LBP produces a histogram of 256 values, which results in a very long vector of 14080 values per sequence. The second step consists of dividing the resulting vectors into training data and test data, in our case; we used half the data for training and the remaining half for testing, meaning that six sequences from each class were used for either train or test. For the last step, the average recognition rate was computed for each descriptor using MLP or RBF; these classifiers will use training data to build a prediction model that will be used to predict the class of each sequence from test dataset, the number of correct predictions divided by the total number of test sequences represents the average recognition rate.

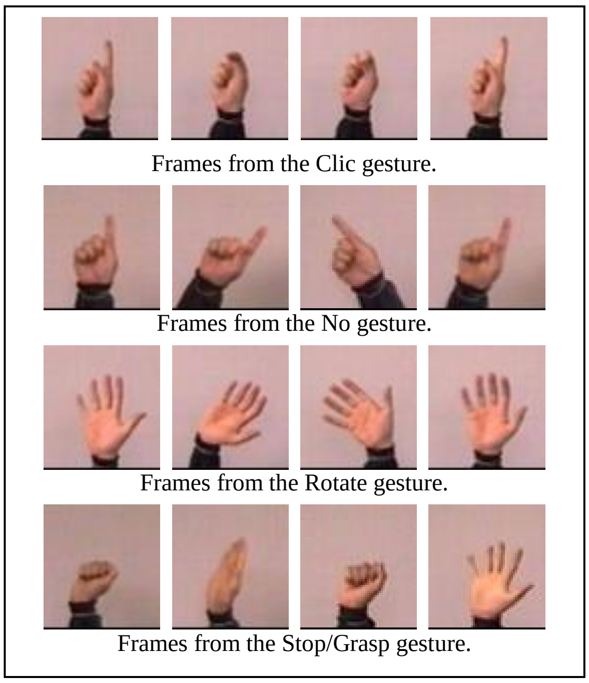

Frames from the Clic gesture.
Frames from the No gesture.
Frames from the Rotate gesture.
Frames from the Stop/Grasp gesture.

Fig. 9. Few frames from each gesture.

For features extraction, we applied the basic LBP and LBP^{riu2} descriptors to all sequences using a radius equals to 1 and 8 neighbors (R=1, P=8), Fig. 10 shows an example using basic LBP descriptor applied to a sequence from Clic gesture, we notice that the LBP code that characterizes this sequence is very long.

To reduce the size of this vector, we investigate a variant of the basic LBP with similar radius and 4 neighbors (R=1, P=4) which reduces total size of the vector by a factor of 2^4. These two configurations were also used for the LBP^{riu2}.

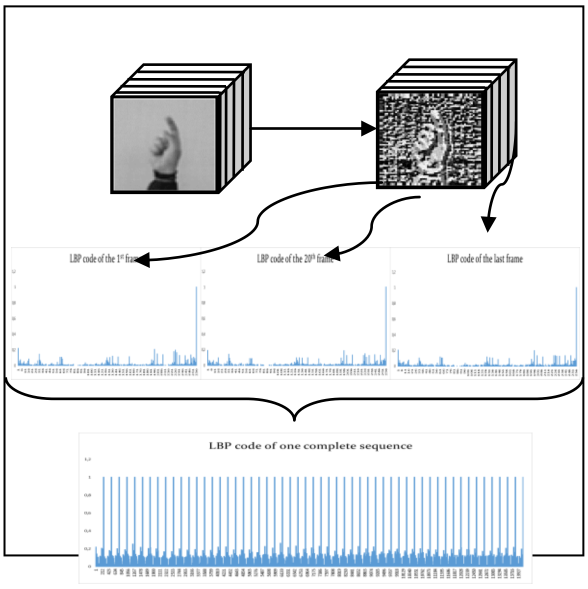

LBP code of the 1st frame
LBP code of the 20th frame
LBP code of the last frame
LBP code of one complete sequence

Fig. 10. Basic LBP applied to a sequence from 'Clic' gesture.

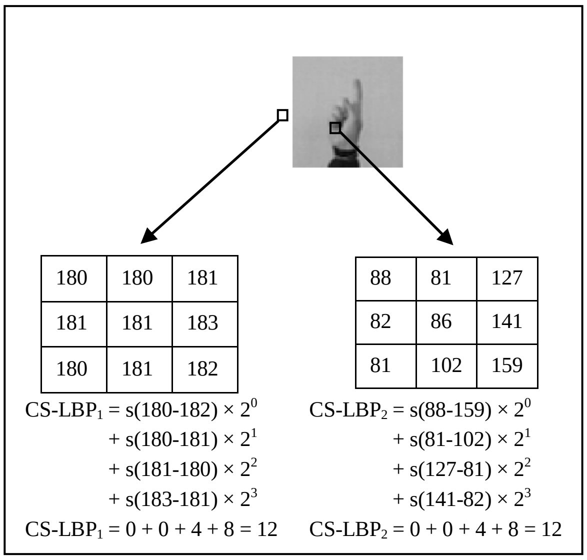
For CS-LBP, we used the standard configuration which produces a compact histogram similar in size to the 4 neighbors configuration of basic LBP, and to improve robustness on flat regions, we investigate different values of thresholds.

Left table of Fig. 11 shows a small region extracted from a frame resenting 'Clic' gesture, grayscale values in this region are supposed to be constant because the background is uniform but because of the noise and the low resolution. We notice that it is not the case which means that flat regions will contribute to the histogram with wrong and random values.

For example, we compare the CS-LBP code of two small regions, the first one on the left represents a flat region, whereas the second one represents an edge, we notice that the code for both regions is equal, yet these regions are completely different, to solve this problem, a small threshold is considered to make the code on flat regions equals to 0.

For EHD, we extracted five types of edges using the 2×2 filters defined in [18], resulting gradient image is shown in Fig. 12 we can see that the hand's shape is not well defined, to solve this problem, we investigate the 3×3 Sobel filters, which are widely used in edge detection.

When we compare gradients from both filters, we notice that the obtained image using Sobel filters have clearer edges.



180	180	181
181	181	183
180	181	182

$$CS\text{-}LBP_1 = s(180-182)\times 2^0 + s(180-181)\times 2^1 + s(181-180)\times 2^2 + s(183-181)\times 2^3$$
$$CS\text{-}LBP_1 = 0+0+4+8 = 12$$

88	81	127
82	86	141
81	102	159

$$CS\text{-}LBP_2 = s(88-159)\times 2^0 + s(81-102)\times 2^1 + s(127-81)\times 2^2 + s(141-82)\times 2^3$$
$$CS\text{-}LBP_2 = 0+0+4+8 = 12$$

Fig. 11. Comparison between CS-LBP code of a flat region and edge.

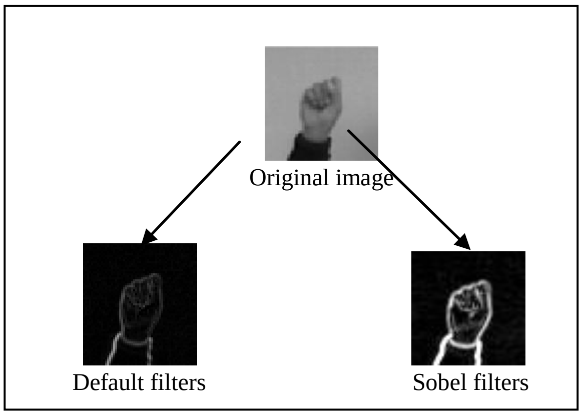


Fig. 12. Comparison between default and Sobel filters.

For test phase, we tuned the parameters of our classifiers to obtain the most optimized configuration that provides best recognition rate. For the MLP, we varied the number of neurons in the hidden layer in the range [1:1:200], and for the RBF, the width of the gaussian function was varied in the range [0.01:0.05:10] and the number of neurons in the hidden layer was varied in the range [1:1:100].

Finally, we investigate the addition of a widely used and important pre-processing step which is image segmentation, in the first experiments, we directly extracted features from grayscale images, so the only pre-processing performed was conversion from RGB to grayscale, we added a segmentation step before features extraction to verify if better results can be achieved. There are many developed algorithms for image segmentation based on either region or edge [25], we opted for a region-based method that is widely used due to its simplicity and speed, this method is called k-means clustering, its main idea is to split the input image into several groups or clusters where the average squared distance between points in the same cluster is minimal, in our case, we chose two clusters (gesture and background). This algorithm was improved to a faster and more accurate version called k-means++ [26]. Fig. 13 shows the result of splitting an image into 2 clusters using k-means++ method, we notice that pixel values of the segmented image are equal to '0' for the background and '1' for the foreground, so in the features extraction phase, the three LBP descriptors will not be able to analyze the structure of the hand in foreground because all pixel values are equal to '1', instead, the LBP descriptors will produce codes equal to 0, to solve this problem, the grayscale image and segmented image are combined to produce a background with values equal to '0' and foreground with grayscale values, this approach is shown in the right part of Fig. 13.

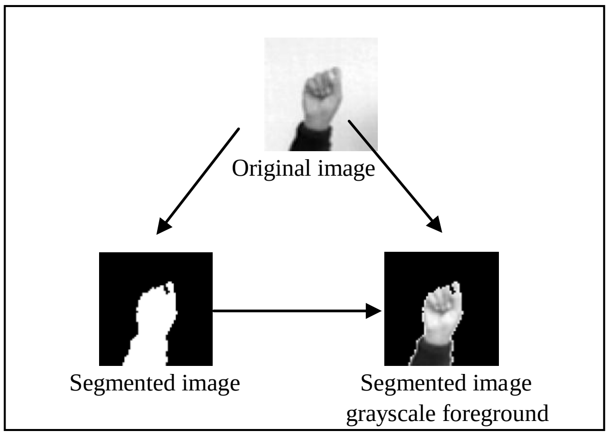


Fig. 13. Image segmentation using K-means++ approach

C) Simulation results and discussions

Recognition rates were computed for each descriptor with various configurations and using either MLP or RBF classifiers with different parameters. Fig. 14 shows obtained accuracy rates for basic LBP descriptor with 8 neighbors and a radius equals to 1 (P=8, R=1) combined to MLP classifier with different number of neurons in hidden layer and conjugate gradient with Powell-Beale restarts, we compared recognition rates obtained using this method with other training techniques such as gradient descent and resilient backpropagation, best accuracy rates and execution times are shown in Table I.

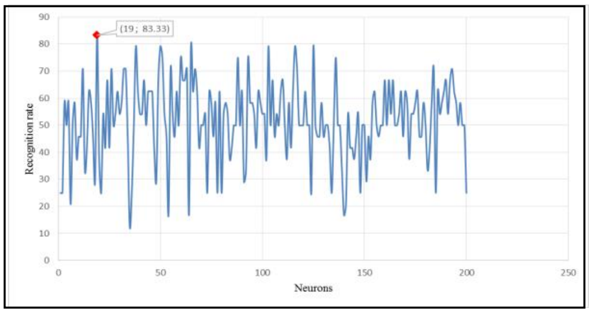


Fig. 14. Recognition rates for basic LBP descriptor using the MLP.

TABLE I. RECOGNITION RATES AND EXECUTION TIMES OBTAINED USING THE BASIC LBP AND DIFFERENT TRAINING FUNCTIONS.

Training function	**MLP**		
	Recognition rate	***Training time***	***Testing time***
Gradient Descent	83.33	21.4	0.25
Resilient Backpropagation	75	2.9	0.25
Powell-Beale restarts	83.33	5.6	0.25

Results in Table I show that a similar recognition rate of 83.33% was achieved when using either gradient descent or conjugate gradient but with big difference in training time, while the conjugate gradient took 5.6 seconds, the gradient descent spent seconds to train the model. Moreover, the resilient backpropagation achieved 75% recognition rate with very fast training time of seconds, so the conjugate gradient with Powell-Beale restarts was used as a training function for MLP since it provides very good recognition rate and fast training and testing times.

Fig. 15 shows results obtained using the same architecture but with RBF classifier for which the spread and number of neurons in hidden layer were varied.

Expermiments done using the basic LBP with 8 neighbors show good recognition rates for both the MLP and the RBF approaches, where a RR of about 83.33% was achieved for the MLP with 19 neurons in the hidden layer, and a higher RR of 87.5% was achieved using the RBF network for spread = 2.91 and 22 neurons in the hidden layer.

Time processing for both classifiers was long due to the large size of the feature vector given by this descriptor, as stated above, for the 8 neighbors configuration the size of the vector for one sequence contains 55×256 = 14080 values. To solve this problem, we used a variant with only 4 neighbors which gives a compact vector 24×55 = 880, results are shown in Table II.

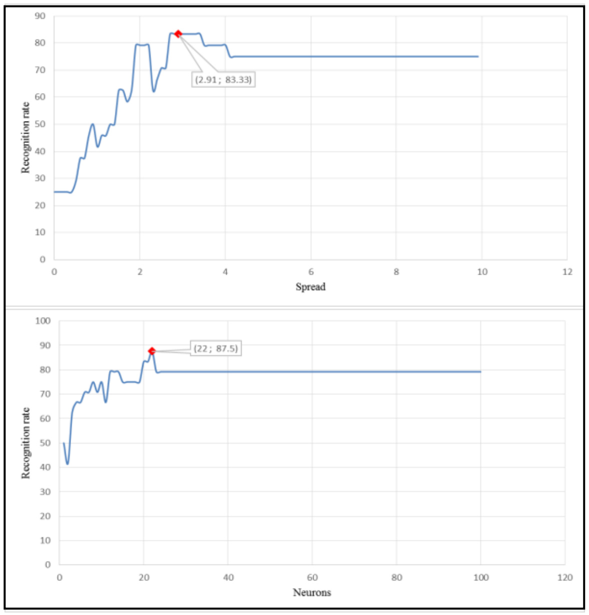


Fig. 15. Recognition rates for the basic LBP descriptor using RBF.

Table II. Recognition rates obtained using the basic LBP.

Features extraction method	**Classifiers**	
	MLP	***RBF***
$LBP_{8,1}$	83.33	87.5
$LBP_{4,1}$	91.67	91.67

We notice that the use of basic LBP with 4 neighbors achieves a high reduction of features vector size and improves the RR obtained. We run the same tests using the LBP^{riu2}, obtained results are presented in Table III.

TABLE III. RECOGNITION RATES OBTAINED USING LBP^{RIU2}.

Features extraction method	**Classifiers**	
	MLP	***RBF***
$LBP_{8,1}^{riu2}$	83.33	75
$LBP_{4,1}^{riu2}$	91.67	95.83

Recognition rates obtained for the LBP^{riu2} are better than the basic LBP. Using the 4 neighbors variant with the RBF classifier achieves a maximum RR of 95.83%.

Fig. 16 shows the obtained results for the CS-LBP descriptor with different values of threshold 'T'.

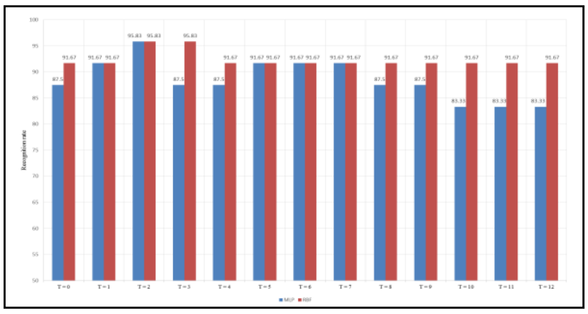


Fig. 16. Recognition rates for CS-LBP with different thresholds.

The maximum accuracy rate for the CS-LBP is about 95.83% and it was obtained using the MLP and RBF for a threshold 'T = 2'. Fig. 17 shows the recognition rates achieved for the EHD when using the filters defined in Fig. 6 and Sobel filters.

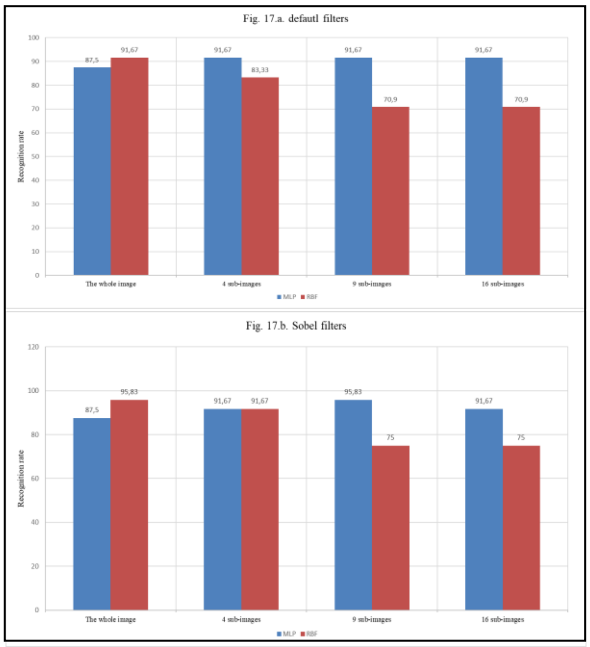


Fig. 17. Recognition rates for the EHD using MLP and RBF.

We notice that recognition rates obtained with Sobel filters are better for both classifiers MLP and RBF, the best RR achieved by both classifiers is 95.83% wheras the highest RR achieved by the default filters is 91.67%.

Table IV shows a comparison between recognition rates obtained for grayscale and segmented images, architecture 1 column show recognition rates for each descriptor applied directely to grayscale images without any additional pre-processing, architecture 2 column present results for all descriptors applied to segmented images using k-means++ algorithm, and finally, results of the combination of segmented and grayscale images are presented in the architecture 3 column.

TABLE IV. RECOGNITION RATES OBTAINED USING SEGMENTED IMAGES.

Features extraction method	Architecture 1		Architecture 2		Architecture 3	
	MLP	***RBF***	***MLP***	***RBF***	***MLP***	***RBF***
$\boldsymbol{LBP_{8.1}}$	83.3	87.5	54.1	66.6	75	91.6
$\boldsymbol{LBP_{4.1}}$	91.6	91.6	54.1	50	83.3	91.6
$\boldsymbol{LBP_{8.1}^{riu2}}$	83.3	75	79.1	70.8	83.3	87.5
$\boldsymbol{LBP_{4.1}^{riu2}}$	91.6	95.8	62.5	79.1	75	91.6
CS-LBP	95.8	95.8	62.5	75	95.8	95.8
EHD	91.6	95.8	45.8	83.3	58.3	91.6

We notice that combining the background of segmented image and the foreground of grayscale image before applying any feature extraction method improves recognition rates for all descriptors. But when compared to features extraction without segmentation, we notice mixed results, for the 8 neighbors configuration of both basic LBP and LBP^{riu2}, the third architecture performed better than the first giving a maximum recognition rate of 91.67% for basic LBP and 87.5% for LBP^{riu2}. For CS-LBP and the 4 neighbors configuration of basic LBP, similar results were obtained for first and third architectures with a recognition rate of 95.83% for CS-LBP and 91.67% for basic LBP. Finally, the first architecture performed better than the third when used with the 4 neighbors LBP^{riu2} and EHD.

Tables V and VI show the processing time taken by our system to extract and then classify relevant features from all sequences using the MLP and RBF classifiers, respectively.

TABLE V. TIME PROCESSING FOR EACH DESCRIPTOR FOR MLP CLASSIFIER (IN SECONDS).

Descriptor	Feature extraction	Train time	Test time	Total
$\boldsymbol{LBP_{8.1}}$	1.5	5.6	0.25	**78**
$\boldsymbol{LBP_{4.1}}$	0.84	2.6	0.21	**43**
$\boldsymbol{LBP_{8.1}^{riu2}}$	1.02	2.2	0.24	**51**
$\boldsymbol{LBP_{4.1}^{riu2}}$	0.69	2.1	0.21	**35**
$\boldsymbol{CS\text{-}LBP_{T=2}}$	0.22	3	0.24	**13**
$\boldsymbol{EHD_{N=3}}$	0.2	3.1	0.24	**13**

The feature extraction time represents the time taken by a descriptor to charecterize and transform a video sequence of 55 frames to a vector descriptor, this dataset contains a global of 48 video sequences. The time taken by classifiers to construct a classification model based on the given training data is referred to as train time, wheras, the time taken by this model to predict the class label of all sequences of test data is called test time. The results shown in these Tables are obtained using the first architecture, when adding the segmentation step before characterization, features extraction time increases only by 0.03 seconds, so times for second and third architectures were not included.

TABLE VI. TIME PROCESSING FOR EACH DESCRIPTOR FOR RBF CLASSIFIER (IN SECONDS).

Descriptor	Feature extraction	Train time	Test time	Total
$\boldsymbol{LBP_{8.1}}$	1.5	3.5	0.4	**76**
$\boldsymbol{LBP_{4.1}}$	0.84	2	0.38	**43**
$\boldsymbol{LBP_{8.1}^{riu2}}$	1.02	1.7	0.4	**52**
$\boldsymbol{LBP_{4.1}^{riu2}}$	0.69	1.4	0.38	**35**
$\boldsymbol{CS\text{-}LBP_{T=2}}$	0.22	1.09	0.37	**12**
$\boldsymbol{EHD_{N=1}}$	0.18	0.95	0.37	**10**

We notice from these Tables that MLP classifier has longer training times than RBF but takes less time when testing, these results were consistent for all descriptors. We notice that LBP with 8 neighbors takes a long time for training using either MLP or RBF, which is, mainly, due to the large data size 14080 values to charecterize one sequence, and by reducing the neighbors to only 4, there is a huge reduction in data size 880 values for one sequence resulting in a substantial reduction in training time. Best timings were obtained using CS-LBP and EHD for both classifiers, where the MLP takes 3 seconds for training and 0.24 seconds for testing, the RBF takes only 1 second for training and 0.37 seconds for testing.

Table VII describes a comparison of our results with other works for the same database.

TABLE VII. COMPARISON WITH PREVIOUS WORKS.

Reference	Method	Number of features	RR (%)
[27]	3DHOG + 3DHOOF	4095	91.67
[28]	Contour-Based Similarity Images	Not mentioned	85.4
[29]	$LBP_{4.1}^{riu2}$ + RBF	330	95.83
[29]	EHD	275	95.83
Proposed method	**K-means++ and CS-LBP**	**880**	**95.83**
Proposed method	**K-means++ EHD + RBF**	**275**	**91.67**

From our analysis, we noticed that the three descriptors perform well on dynamic hand gesture database using MLP and RBF classifiers, where CS-LBP and EHD give good efficiency with processing time of around 10 seconds, we also notice that adding a segmentation phase before features extraction improves the recognition rates for some descriptors.

V. CONCLUSION

In this paper, we have studied and implemented a dynamic gesture recognition system with uniform background, the used database contains video sequences with a very low resolution, and each sequence is performed by different persons.

Different characterization techniques were implemented and evaluated by two variants of the ANN, which are the MLP network and the RBF network. Very good recognition rates were obtained for the LBP^{riu2}, the CS-LBP and the EHD using Sobel filters where almost all sequences were correctly classified giving a maximum accuracy rate of 95.83%, and 91.67% for the basic LBP. The MLP network outperformed the RBF network when using the LBP^{riu2} but for the other descriptors, they performed similarly.

For future work, we aim to elaborate an automatic recognition system for dynamic gestures using uniform and complex background. In addition, other efficient classifiers could be investigated such as deep learning to ameliorate the recognition rate.

VI. REFERENCES

[1] M. Panwar, and P. Singh Mehra, Hand gesture recognition for human computer interaction, *International Conference on Image Information Processing,* Shimla, India, 2011, 1-7.

[2] S. S. Rautaray, and A. Agrawal, Real Time Gesture Recognition System for Interaction in Dynamic Environment, *Journal of Procedia Technology*, 4, 2012, 595-599.

[3] X. Wang, J. Jiang, Y. Wei, L. Kang, and Y. Gao, Research on Gesture Recognition Method Based on Computer Vision, *2nd International Conference on Electronic Information Technology and Computer Engineering (EITCE)*, China, 2018, 1-6.

[4] J. P. Wachs, M. Kölsch, H. Stern, and Y. Edan, Vision-based hand-gesture applications, *ACM Communications, 54(2)*, 2011, New York, USA, 60-71.

[5] Q. de Smedt, *Dynamic hand gesture recognition – From traditional handcrafted to recent deep learning approaches*, University of Lille 1, Sciences and Technologies, 2017.

[6] Y. Chen, B. Luo, Y. L. Chen, G. Liang, X. Wu, A real-time dynamic hand gesture recognition system using kinect sensor, *IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Zhuhai, China, 2015, 2026-2030.

[7] D. Xu, Y. Chen, C. Lin, X. Kong and X. Wu, Real-time dynamic gesture recognition system based on depth perception for robot navigation, *IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Guangzhou, China, 2012, 689-694.

[8] R. Rokade-Shinde and J. Sonawane, Dynamic hand gesture recognition, *International Conference on Signal and Information Processing (IConSIP)*, Vishnupuri, India, 2016, 1-4.

[9] C. Mengmeng, F. Zhiquan and L. Min, Dynamic Gesture Recognition Based on the Trend of Key Points, *2nd International Conference on Multimedia and Image Processing (ICMIP)*, Wuhan, China, 2017, 98-102.

[10] M. Hruz, J. Trojanova, M. Zelezny, Local binary pattern based features for sign language recognition, *Pattern Recognition and Image Analysis, 21(3)*, 2011, 398-401.

[11] Y. Ding, H. Pang, X. Wu and J. Lan, Recognition of hand-gestures using improved local binary pattern, *International Conference on Multimedia Technology*, Hangzhou, China, 2011, 3171-3174.

[12] J. Megha, J. Padmaja, and D. Doye, Radially Defined Local Binary Patterns for Hand Gesture Recognition, *International Journal of Advanced Trends in Computer Science and Engineering, 4(4)*, 2015, 44-48.

[13] H. Lahiani, M. Neji, Hand Gesture Recognition System Based on LBP and SVM for Mobile Devices, *11th International Conference on Computational Collective Intelligence*, Hendaye, France, 2019, 283-295.

[14] A. Rosenfeld, K. Fu, Pattern Recognition and Image Processing, *IEEE Transactions on Computers, 25(12)*, 1976, 1336-1346.

[15] T. Ojala, M. Pietikainen, and D. Harwood, A comparative study of texture measures with classification based on feature distributions, *Pattern Recognition, 29(1)*, 1996, 51-59.

[16] T. Ojala, M. Pietikainen, T. Mäenpää, Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns, *IEEE Transactions on pattern analysis and machine intelligence, 24(7)*, 2002, 971-987.

[17] M. Heikkilä, M. Pietikäinen, C. Schmid, Description of Interest Regions with Center-Symmetric Local Binary Patterns, *5th Indian Conference on Computer Vision, Graphics and Image Processing (ICVGIP)*, Madurai, India, 2006, 58-69.

[18] C. S. Won, D. K. Park, S. J. Park, Efficient Use of MPEG7 Edge Histogram Descriptor, *Etri Journal, 24(1)*, 2002, 23-30.

[19] J. C. Principle, N. R. Euliano, W. C. Lefebvre, *Neural and adaptive Systems: Fundamentals Through Simulation* (John Wiley and Sons, 1999).

[20] A. K. Jain, J. Mao, K. M. Mohiuddin, Artificial Neural Networks : A Tutorial, *IEEE Transitions on Computer, 29(3)*, 1996, 31-44.

[21] M. J. D. Powell Restart Procedures for the Conjugate Gradient Method, *Mathematical Programming, 12(1)*, 1977, 241-254.

[22] I. Yilmaz, O. Kaynar, Multiple regression, ANN (RBF, MLP) and ANFIS models for Predection of Swell Potential of Clayey Soils, *Expert Systems with Applications, 38(5)*, 2011, 5958-5966.

[23] G. M. Foody, Supervised Image Classification by MLP and RBF neural networks with and without an exhaustively defined sets of classes, *International Journal of Remote Sensing, 25(15)*, 2004, 3091-3104.

[24] http://www.idiap.ch/resource/gestures/

[25] N. M. Zaitoun, M. J. Aqel, Survey on Image Segmentation Techniques, *International Conference on Communications, management, and Information technology (ICCMIT'2015)*, Prague, Czech Republic, 2015, 797-806.

[26] D. Arthur, S. Vassilvitskii, k-Means++: the advantages of careful seeding. *Proceedings of the eighteenth annual ACM-SIAM symposium on discrete algorithms*, Louisiana, USA, 2007, 1027-1035.

[27] S. E. Agab, F.Z. Chelali, HOG and HOOF Spatio-Temporal Descriptors for Gesture Recognition, *IEEE International Conference on Signal, Image, Vision and their applications (SIVA)*, Guelma, Algeria, 2018, 1-7.

[28] S. Nasri, A. Behrad, and F. Razzazi, A Novel Approach for Dynamic Hand Gesture Recognition using Contour-Based Similarity Images, *International Journal of Computer Mathematics, 92(4)*, 2015, 662-685.

[29] S. E. Agab, F.Z. Chelali, Dynamic hand gesture recognition based on textural features, *IEEE International Conference on Advanced Electrical Engineering (ICAEE)*, Algiers, Algeria, 2019, 1-6.