Static hand gesture recognition using neural networks

Haitham Hasan · S. Abdul-Kareem

Published online: 12 January 2012
© Springer Science+Business Media B.V. 2012

Abstract  This paper presents a novel technique for hand gesture recognition through human–computer interaction based on shape analysis. The main objective of this effort is to explore the utility of a neural network-based approach to the recognition of the hand gestures. A unique multi-layer perception of neural network is built for classification by using back-propagation learning algorithm. The goal of static hand gesture recognition is to classify the given hand gesture data represented by some features into some predefined finite number of gesture classes. The proposed system presents a recognition algorithm to recognize a set of six specific static hand gestures, namely: Open, Close, Cut, Paste, Maximize, and Minimize. The hand gesture image is passed through three stages, preprocessing, feature extraction, and classification. In preprocessing stage some operations are applied to extract the hand gesture from its background and prepare the hand gesture image for the feature extraction stage. In the first method, the hand contour is used as a feature which treats scaling and translation of problems (in some cases). The complex moment algorithm is, however, used to describe the hand gesture and treat the rotation problem in addition to the scaling and translation. The algorithm used in a multi-layer neural network classifier which uses back-propagation learning algorithm. The results show that the first method has a performance of 70.83% recognition, while the second method, proposed in this article, has a better performance of 86.38% recognition rate.

Keywords  Gesture recognition · Hand gestures · Artificial neural network · Human–computer interaction · Computer vision

H. Hasan (✉) · S. Abdul-Kareem
Faculty of Computer Science and Information of Technology, University of Malaya, 50603 Kuala Lumpur, Malaysia
e-mail: haitham@siswa.um.edu.my
S. Abdul-Kareem
e-mail: sameem@um.edu.my.com
1 Introduction

Since the first introduction of computer into the modern era, it has penetrated into all corners of our personal and social lives as a key element revolutionizing our way of living. Surfing the web, typing a letter, playing a video game or storing and retrieving personal or official data are just a few examples of the use of computers or computer-based devices. Due to increase in mass production and constant decrease in price of personal computers, they will even influence our everyday life more in near future. Nevertheless, in order to efficiently utilize the new phenomenon, myriad number of studies has been carried out on computer applications and their requirement of more and more interactions. For this reason, human–computer interaction (HCI), among others, has been considered to be a lively field of research in the last few years (Just 2006).

Since the introduction of the most common input devices, they have not changed considerably, probably due to the fact that the existing devices are still sufficiently applicable and effective. However, it is also well known that computers have been tightly integrated into everyday life with the constant introduction of new applications and hardware in the recent years (Symonidis 2000). Nowadays, majority of the HCI is based on mechanical devices such as keyboard, mouse, joystick or game-pad, but a growing interest in a class of methods based on computational vision has been emerged due to ability to recognize human gestures in a natural way (Manresa et al. 2000). Recently, use of human movements, especially hand gesture, has become an important part of Human Computer Intelligent Interaction (HCII), which serves as a motivating force for research in modelling, analyzing and recognizing the hand gestures. Many techniques developed in HCII can also be extended to other areas such as surveillance, robot control and teleconferencing (Wu and Huang 1999). The significance of the problem can be easily illustrated by the use of natural gestures applied together with verbal and non verbal communications (Dadgostar et al. 2005).

1.1 Hand gesture analysis

There are three main approaches in the hand gesture analysis: glove based analysis, vision based analysis, and analysis of drawing gestures (Ionescu et al. 2005). The first approach employs sensors (mechanical or optical) attached to a glove that transducers finger flexions into electrical signals for determining the hand posture. The relative position of the hand is determined by an additional sensor, this sensor is normally a magnetic or an acoustic sensor attached to the glove. For some data-glove application, look-up table software toolkits are provided with the glove to be used for hand posture recognition (Symonidis 2000). The second approach is, vision based analysis, it is based on the way human beings perceive information about their surroundings, yes it is probably the most difficult to implement in a satisfactory way. Several different approaches have been tested so far. One is to build a three dimensional model of the human hand. The model is matched to images of the hand by one or more cameras and parameters corresponding to palm orientation and joint angles are estimated. These parameters are then used to perform gesture classification (Symonidis 2000). The third approach is analysis of drawing gestures, which usually involves the use of a stylus as an input device. Analysis of drawing gestures can also lead to recognition of written text. The vast majority of hand gesture recognition work has used mechanical sensing, most often for direct manipulation of a virtual environment and occasionally for symbolic communication. Sensing the hand posture (static gesture), mechanically has a range of problems, however, including reliability, accuracy and
electromagnetic noise. Visual sensing has the potential to make gestural interaction more practical, but potentially embodies some of the most difficult problems in machine vision (Symonidis 2000).

1.2 Background studies

Static gesture recognition is a pattern recognition interface. The first step, before using any standard pattern recognition technique, is feature extraction. In this recognition, features correspond to the most discriminated information contained in the lighting conditions. A fair amount of research has been done on different aspects of this approach. Some of the approaches and concepts in connection with gesture recognition are introduced and evaluated in this paper.

In line with this trend, Freeman and Roth (1995) have jointly introduced a method to recognize hand gestures, based on a pattern recognition technique developed by McConnell employing histograms of local orientation. Their method uses the orientation histogram as a feature vector for gesture classification and interpolation. The method is simple and fast to compute, and it offers some robustness to scene illumination changes. However, this approach provides power and robustness at the expense of speed. In this method, a pattern recognition system is applied along with transformation, $T$, which converts image or image sequence into a feature vector, which is then compared with the feature vectors of a training set of gestures. Euclidean distance metric is used for this purpose. Another system for classification of hand postures against complex backgrounds in grey-level images has also been suggested by Triesch and von der Malsburg (1996). The presented system employs elastic graph matching which has already been successfully employed for face recognition. Elastic matching of a model graph $M$ to an image means to search for a set $X_N$ of node positions simultaneously satisfying two constraints. The local image information attached to each node must match the image icon around the position where the node is placed. However, the distances between the matched node positions must not differ too much from the signal distances. The correct classification of this system is 86.2% in comparison to others. Naidoo et al. (1998) are members of another group of scholars who have jointly suggested a system that recognizes static hand images against complex backgrounds. In this method, a Support Vector recognition system is used to classify hand postures as gestures. Support Vector Machine (SVM) is a linear machine with some very good properties. The main idea of an SVM in the context of pattern classification is to construct a hyperplane as the decision surface. The hyperplane is constructed in a way to maximize the margin of separation between positive and negative examples. The SVM uses a coupled approach based on statistical learning theory known as natural risk minimization, which minimizes an upper bound on the minimization error (i.e. a parameter estimation heuristic seeks parameter values and minimizes the rate of “risk” or “loss” that the model on the training data has). Yet, another hand gesture recognition system has also been developed by Licsar and Sziranyi (2002) based on the shape analysis of static gestures for HCI. This appearance-based recognition system uses MFD (Modified Fourier Descriptors) for the classification of hand shapes. This is also an example-based system which includes two phases, namely: training and running. In the training stage, the user shows the system one or more examples of hand gestures, and the system stores the carrier coefficients of the hand shape. In the running phase, the computer compares the current hand shape with each of the stored shapes by coefficients. The best gesture match is selected by the nearest-neighbour method with distance metric of Symonidis (2000) has also presented an approach for static hand gesture recognition using orientation histograms.
as a simple and fast algorithm developed to work on a workstation. This approach recognizes static hand gestures, namely, a subset of American Sign Language (ASL). Pattern recognition system uses a transform that converts an image into vector, then it compares the vector with the feature vectors of a set of gestures. The final system is implemented with a network. Hu et al. (2003) have developed a new visual gesture recognition method for the human–machine Interface of mobile robot operation. The interface uses seven static hand gestures each of which represents an individual control command for the motion control of the remote robot. And adaptive object segmentation with color image in HLS representation is used and recognition is made by edge codes, matching, and skeletonising. A new approach has been presented by Chang et al. (2006) which recognizes static gestures based on Zernike moments (ZMs) and pseudo-Zernike moments (PZMs). This approach includes four stages. In the first step, an input static gesture is segmented into a binary hand silhouette via the modified color segmentation approach. In the second step, the binary hand silhouette is recommended with a Minimum Bounding Circle (MBC). In the third step, binary hand silhouette is decomposed in the finger part and palm part by morphological operations according to the radius of the MBC. The ZMs and PZMs of the important finger and palm parts are respectively computed based on the centre of MBC. Finally, the nearest-neighbour techniques are used to perform the matching between an input feature vector and stored feature vector for static gesture identification. The proposed technique has recognition rate of 95%. Just (2006) has proposed to apply an approach previously used for face recognition to the hand posture (static gesture) classification and recognition tasks. The feature is based on the local parametric pixel operator: modified Census Transform (MCT) and information invariant. For the classification and recognition processes, a linear classifier is trained, using a set of feature lookup-benchmark in the field of posture recognition. Parvini and Shahabi (2007) have jointly proposed an approach recognizing static and dynamic hand gestures by analyzing the raw streams generated by the sensors attached to human hands. The recognition of a sign is based on the observation of all forms of hand signs and finger-joint movements from a starting posture to a final posture. In this approach, the concept of the Range of Motion is utilized for each joint movement. Range of Motion (ROM) is a quantity which defines the movement by measuring the angle from the starting position of the axis to its position at the end of the full range of the movement.

1.3 Comparison with other approaches

Not many vision based approaches have been reported for static hand gestures considered in this work. The original contributions of this work are, a novel technique to develop a vision based gesture recognition algorithm to recognize a set of static hand gestures with scaling, translation and rotation actions and movements with the idea of using two types of features: hand contour as a boundary based feature in the first method and the complex moments as a region based feature in the second method. Unlike the earlier methods, we find that hand shape has not been explicitly considered as a possible feature. It was reported that use of hand shape makes it easier for the gesturer to remember commands (Kinnebrock 1995). In our study six static hand gestures in the proposed method; the proposed algorithm uses the multi-layer perception of neural network-based classifier. Gestural interfaces based on vision technologies are the most common way for the construction of advanced man-machine interfaces. However, in this method, the size of the image is used instead of the dedicated acquisition devices.
2 Gesture recognition

Humans naturally use gesture to communicate, it has been constricted that young children can readily learn to communicate with machine before they learn to talk (Kjrlfdn 1997). Gestures are frequently use by means for communication, they are used for everything from pointing at person to get their attention, to conveying information about space and characteristics. There is evidence that gestures are not merely used as qualifiers for oral communication but are actually part of the main language generation process (Shet 2003). Gestures in modern societies are everything from a smile or a present to hand and arm movements. Most people add meaning to their words by drawing pictures with their hands. In most cases this is subconsciously and is therefore hard to suppress. They do it even they speak on the phone or when they talk to themselves. Deaf people might use sign language as a sole means to communicate in diverse fields such as military or aerobics use arm signals to recognize commands (Winnem 1999). Gestures play a major role in many aspects of human life. Gesturing probably universal, there has been no report of a community that does use gesture. Gestures are a crucial part of everyday conversation such as in Greek paintings, Indian Miniatures or European paintings. Gestures play a role in religious or spiritual rituals such as the Christian sign of the cross. In Hinduism and Buddhism, a mudra (Sanskrit, literally seal) symbolic gesture made with the hand or fingers. Each mudra has a specific meaning, playing a central role in Hindu and Buddhist photography. An example is the Vitarka mudra. It is done joining the tips of the thumb and the index together, while keeping the fingers straight (Gesture 2007).

2.1 Scope of the problem

Gestural interfaces based on vision technologies are the most way for the construction of advanced man-machine interfaces, but size of images instead of dedicated acquisition devices. There are three main problems: segmentation of the hand, tracking and recognition of the hand posture of (feature extraction and classification) (Just 2006). (a) Segmentation: in most of the literature, hand segmentation has been performed using a controlled (uncluttered) background, using a known ground (i.e. background subtraction), using segmentation by motion, using color segmentation (i.e. skin color filtering). Using controlled or backgrounds can be problematic in dynamic environments where backgrounds can change over time, and thus are non-realistic. Motion can be difficult to apply due to artefacts of motion cause by light and camera motions. Color segmentation is a fast and robust approach to hand segmentation that works well under lighting conditions and against unknown backgrounds. It cannot use if the background behind the hand is close to the color of the segmentation. (b) Tracking: articulated objects (such as the hand) are more difficult to track than the rigid objects. The analysis process of the human hand is further articulated by the fact that the hand is a nor-rigid articulated structure, changes in shape in various ways. The four fingers are adjacent to other that leads to self-occlusions. To overcome these difficulties, tracking has been facilitated through the use of special markers (colored gloves or color marked gloves) and a combination of color and shape constraints. Tracking can be done using on single camera (mono) or using multiple cameras (stereo). To track the hand in 3D, a model of the hand is needed. Hand tracking needs to update well chosen parameters through consecutive images. And this problem is strongly tied with hand segmentation in each image alone the tracking process (Just 2006). (c) Recognition: the major difficulties of hand posture (static gesture) and gesture recognition are feature extraction and the recognition itself. Because hand postures and gestures are highly variable from one person to another, it is essential to capture their essence-their invariant properties and use this information to represent them.
The features must optimally distinguish the variety of hand gestures or postures from each other and make recognition of similar gestures of postures simpler. Another problem related to hand gestures recognition is gesture segmentation (or gesture spotting). Hand gesture spotting consists of determining the start and end points of a particular gesture. Gesture spotting needs also to deal with the rejection of unknown gestures (Just 2006).

2.2 Gesture definition

The following definition of gesture can be bound the Oxford-advance learners dictionary: gesture—a movement of a part of the body, especially, the hand or hand intended to suggest a certain meaning (Lamar 2001). The word gesture is used for many different phenomena involving human movement, especially of the hands and arms, only some of these are interactive or communicative (Nehaniv et al. 2005). There is a difference between gestures and pure functional movements which can be achieved with other actions, too. Movements which show or symbolize something and contain a message are called gestures. For example, steering a car is a pure functional movement without information in the context of gestures describing the size of a round object by circling the hand, contains information about how big the object is. Writing down words on a sheet of paper is a pure functional movement. The words contain the information, not the movement of the hand. The writing can be exchanged by typing; therefore writing is not a gesture (Gestures 2007).

2.3 Hand gestures

Hand gestures, i.e. gestures performed by one or two hands, is the most numerous category of gestures due to the ability of the human hand to acquire a huge number of clearly discernible configurations, the fact of importance for the sign languages (Gesture 2007). According to different application scenarios, hand gestures can be classified into several categories such as conversational gestures, controlling gestures, manipulative gestures, and communicative gestures. Sign language is an important case of communicative gestures. Since sign languages are highly structural, they are very suitable of acting as a test-bed for vision algorithms. At the same time they can also be a good way to help the disabled to interact with computers. Controlling gestures are the focus of current research in Vision-Based Interface (VBI). Virtual objects can be located by analysing pointing gestures. Some display control applications demonstrate the potential of pointing gestures in HCI. Another controlling gesture is navigating gesture. Instead of using wands, the orientation of hands can be captured as a 3 D directional input to navigate the virtual environments (VEs). The manipulative gesture will serve a natural way to interact with virtual objects. Tele-operation and virtual assembly are good examples of applications. Communicative gestures are subtle in human interaction, which involves psychological studies; however, vision-based motion capturing techniques can help those studies (Wu and Huang 1999). Generally gestures can be classified into static gestures and dynamic gestures. Static gestures are usually described in terms of hand shapes and dynamic gestures are generally described according to hand movements (Chang et al. 2006).

2.3.1 Static gestures

Static gesture or hand posture, adopting the posture definition used by Liang (Lamar 2001): posture is a specific combination of hand position, orientation, and flexion observed at some time instance Posture of static gestures are not time varying signals, so they can be completely
analysed using only one or a set of images of the hand took in a specific time. Good examples of postures are the facial information like a smile or an angry face, and the hand postures for OK or STOP hand signs, which a simple picture is enough for complete understanding.

2.3.2 Dynamic gestures

The word gesture is used to describe dynamic gestures, according to definition used by Liang (Lamar 2001); Gesture is a sequence of postures connected by motions over a short time span. A gesture can be thought of as a sequence of postures. In a video signal the individual frames define the postures and the video sequence defines the gesture. The head No and Yes, and hand goodbye or come here gestures can only be recognized taking the temporal context information being good examples of dynamic gestures.

2.4 Hand gesture recognition

Since the first study on gesture recognition conducted by Guo et al. (1998) during the mid seventies, many practical applications and theories have been suggested by various researchers. Gesture recognition has adapted to various gestural studies ranging from facial gestures to complete human body actions and reactions. Because of its wide range of applications such as human–computer interfaces, sign language interpretation and visual surveillance, gesture recognition is an important topic in the studies of computer-based vision. Both spatial variation and temporal variation among gesture samples make this recognition difficult. Different subjects, for instance, have different hand appearances and different gestures have different paces (Kim and Cipolla 2007).

However, serious work on gesture recognition began as early as 1992 when the first frame grabbers for color video inputs were available which could grab color images in real time. Improvements in color recognition and segmentation brought about development of gesture recognition (Shet 2003), because real time performance is a prerequisite for HCI. Until a few years ago, nearly all works in this area used mechanical devices to sense shape of the hand. The most common device was a thin glove equipped with bending sensors of various kinds such as the Data Glove. Often a magnetic field sensor was used to detect hand position and orientation, but that was notorious for having problems in the electro magnetically noisy environments of most computer labs (Kjrlfdrn 1997).

A primary goal of gesture recognition research, however, is to create a system which can identify specific human gestures and use them for information conveyance or for device control (Guitar 2002).

2.5 The basics of gesture recognition

The general gesture recognition process in systems of any type can be broken down into the following components as shown in Fig. 1 (Winnem 1999).
The first stage, as displayed in the figure, is mostly concerned with the hardware of the system and the way data for the recognition process is gathered (in the form of bitmaps or lists of vertices). The second stage is a preprocessing stage. In this stage, edge-detection, as well as smoothing and other filtering processes, can occur. This stage prepares the data for the main computational stage for feature extraction. Some systems might never use the term feature, but somewhere along the line they will find a way of quantifying their input. The features of the input are then evaluated in one or more of several possible ways to make a decision about which gesture the system is most likely subjected to in the fourth stage, also known as evaluation stage. Nevertheless, all systems will have a limited set of gestures that they can recognize at any given time (Winem 1999).

2.6 Basics of image processing and neural networks

Gesture recognition and gesture-based interaction are becoming an increasingly attractive research subject in HCI. To implement gesture-based HCI it needs capturing the necessary information to infer what gesture is performed by the use. Recognizing gestures is complex task which involves many aspects such as pattern recognition and machine learning even psycholinguistic studies, to achieve this task there must be a great knowledge in the following areas: computer imaging feature extraction neural network. The purpose of this is to give an idea of, and some introduction to each one of them.

2.7 Computer imaging

Computer imaging is a fascinating and exciting area to be involved in today (Umbaugh 1998). The advent of the information super highway, with its ease of use via the World Wide Web, combined with the advances in computer power have brought the world into our offices and into our homes. One of the most interesting aspects of this information revolution is the ability to send and receive complex data that transcend ordinary written text visual information transmitted in the form of digital images is becoming a major method of communication in the modern age. Computer imaging can be defined as the acquisition and processing of visual information by computer. Computer imaging can be divided into two primary categories: computer vision image processing in the computer vision application the processed (output) images are for use by a computer, whereas in Image Processing applications the output images are for human consumption. The human visual system and the computer as a vision system have varying limitations and strengths and the computer imaging specialist need to be aware of the functionality of these two very different systems.

2.8 Segmentation

Segmentation is the initial stage for any recognition process whereby the acquired image is broken up into meaningful regions or segments. The segmentation process is not primarily concerned with what the regions represent but only with the process of partitioning the image. In the simplest case (binary images) there are only two regions: a foreground (object) region and a background region. In gray level images there may be many types of region or classes within the image for example in a natural scene to be segmented there may be regions of clouds, ground, building and trees (Awcock and Thomas 1995). Segmentation subdivides and image into its constituent parts of objects the level to which this subdivision is carried depends on the problem being solved. That is segmentation should stop when the objects of interest in an application have been isolated (Gonzalez and Woods 2001). There are two
main approaches to segmentation: 1. Pixel-based or local methods that includes: edge detection and boundary detection. 2. Region-based or global approaches, which include region merging and splitting and threshold (Awcock and Thomas 1995). Thresholding The simplest image segmentation problem occurs when and image contains an object having homogeneous intensity and a background with different intensity levels (Pitas 1998). Thresholding techniques is to partition the image histogram by using a single threshold, T. Segmentation is then accomplished by scanning the image pixel by pixel and labelling each pixel as object or background depending on whether the gray level of that pixel is greater of less than the value of T (Gonzalez and Woods 2001).

2.9 Noise reduction

Spatial filters can be effectively used to remove various types of noise in digital images. These spatial filters typically operate on small neighbourhoods, $$(3 \times 3)$$ to $$(11 \times 11)$$. Many spatial filters are implemented with convolution masks. Because a convolution mask operation provides a result that is a weighted sum of the values of a pixel and its neighbours, it is called a linear filter. The mean filters are essentially averaging filters; they operate on local groups of pixels called neighbourhoods and replace the centre pixel with an average of the pixels in this neighbourhood. This replacement is done with a convolution mask (Umbaugh 1998). The median filter is a nonlinear filter. A nonlinear filter has a result that cannot be found by a weighted sum of the neighbourhood pixels as done with convolution mask. However the median filter operates on a local neighbourhood after the size of the local neighbourhood is defined the centre pixel is replaces with the median or centre value present among its neighbours rather than by average (Umbaugh 1998). The median filter disregards extreme values (high or low) and does not allow them to influence the selection of a pixel value which is truly representative of the neighbourhood. It is therefore good at removing isolated extreme noise pixels (often known as salt and pepper noise), while substantially retaining spatial detail. However its performance deteriorates when the number of nose pixels is more than half the number of pixels in the window (Awcock and Thomas 1995).

2.10 Edge detection

Edges are basic image features. They carry useful information about object boundaries which can be used for image analysis object identification and for image filtering applications as well (Pitas 1998). Edge detection methods are used as a first step in the line detection process and also used to find complex object boundaries by marking potential edge points corresponding to places in an image where changes in brightness occur. After these edge points have been marked they can be merged to form lines and object outlines. Edge detection operations are based on the idea that edge information in an image is found by looking at the relationship a pixel has with its neighbours with widely varying gray levels it may represent an edge point. In other words an edge is defined by a discontinuity in gray-level values. Ideally an edge is caused by changes in color or texture or by the specific lighting conditions present during the image acquisition process (Umbaugh 1998).

2.10.1 Sobel operator

Sobel operator is recognized as one of the best simple edge operators it utilizes two $$(3 \times 3)$$ masks (Awcock and Thomas 1995). The Sobel edge detection masks look for the
horizontal and vertical directions and then combine this information into a single metric (Fig. 2).

These masks are each convolved with the image. At each pixel location there are two numbers $S_1$ corresponding to the result from the row mask, and $S_2$ from the column mask; these numbers are used to compute two metrics, the edge magnitude and the direction which are defined as follows (Umbaugh 1998).

\[
\text{Edge Magnitude} = \sqrt{S_1^2 + S_2^2} \quad (1)
\]

\[
\text{Edge Direction} : \tan^{-1} \left( \frac{S_1}{S_2} \right) \quad (2)
\]

2.10.2 Prewitt operator

Prewitt operator is similar to the Sobel operator but with different mask coefficients the masks are defined as follows: the edge magnitude and the direction are defined as follows (Umbaugh 1998) (Fig. 3).

\[
\text{Edge Magnitude} = \sqrt{P_1^2 + P_2^2} \quad (3)
\]

\[
\text{Edge Direction} : \tan^{-1} \left( \frac{P_1}{P_2} \right) \quad (4)
\]

2.10.3 Laplacian operator

The Laplacian operator is a second order derivative operation that has zero crossing (e.g. transition from positive to negative and vice versa) (Gonzalez and Woods 2001; Pitas 1998). Laplacian masks are rotationally symmetric, which means edges at all orientations contribute to result. They are applied by selecting one mask and convolving it with the image. The sign of the result (positive or negative) from two adjacent pixel locations provides directional information and tells as which since of the edge is brighter (Umbaugh 1998).

2.11 Coordinate normalization

The idea of coordinate normalization is to map the scaled hand image coordinates to the standard size ranging between $[-1, +1]$ (Musa 1998). The purpose of this step is to keep domain
of image coordinates fixed and irrelevant to the original size. The condition of keeping the domain of coordinates within limited boundaries will effectively satisfy the convergence of higher ordered moments so the scaled image coordinates \((x, y)\) will be transformed into the normalized set \((x, y)\) that can be considered as a standard version of the original coordinated \((x, y)\). By using centre of image each pixel coordinates \((x_s, y_s)\) value are mapped to the domain \([-1, +1]\) this process can be done using the following equations:

\[
X_n = \left(\frac{2}{W - 1} \times X\right) - 1 \quad (5)
\]

\[
Y_n = \left(\frac{2}{H - 1} \times Y\right) - 1 \quad (6)
\]

where \(H\) and \(W\) are the height and width of the scaled image respectively.

2.12 Feature extraction

Feature extraction is part of the data reduction process and is followed by feature analysis. One of the important aspects of feature analysis is to determine exactly which futures are important (Umbaugh 1998). Feature extraction is a complex problem and often the whole image or transformed image is taken as input. The goal of feature extraction is to find the most discriminate information in the recorded images. Feature extraction operates on two-dimensional image array but produces a list of descriptions or a feature vector (Awcock and Thomas 1995; Huang 1998). Mathematically a feature is an n-dimensional vector with its components computed by some image analysis. The most commonly used visual cues are color, texture, shape, spatial information and motion in video. For example the color may represent the color information in an image such as color histogram color binary sets, color coherent vectors. The \((n)\) components of a feature may be derived from one visual cue or from composite cues such as the combination of color and texture (Huang 1998). Selecting good features is crucial to gesture recognition since hand gestures are very rich in shape variation, motion and textures. For static hand gesture recognition, although it is possible to recognize hand posture by extracting some geometric features such as fingertips, finger directions and hand contours, such features are not always available and reliable due to self-occlusion and lighting conditions. There are also many other non-geometric features such as color, silhouette and textures, however, they are inadequate in recognition. Since it is not easy to specify features explicitly the whole image or transformed image is taken as the input and features are selected implicitly and automatically by the recognizer (Wu and Huang 1999).

2.12.1 Contour detection

Contour detection in real images is a fundamental problem in many computer vision tasks. Contours are distinguished from edges as follows. Edges are variations in intensity level in a gray level image whereas contour is salient coarse edges that belong to objects and region boundaries in the image. By salient is meant that the contour map drawn by human observers include these edges as they are considered to be salient. However the contours produced by different humans for a given image are not identical when the images are of complex natural (Joshi and Sivaswamy 2004). The usefulness of contour detection in a great number of applications has been well established and demonstrated. Indeed this operation is of great help for further image analysis and scene understanding. There are many kinds of edge detectors most of them are based on derivative operators that give a high response at the contour points and a low response in homogeneous areas. The oldest and simplest edge detector is undoubtedly the digital gradient operator (Boussaid and Beghdaddi 1999). Contour detection can be
implemented in a simple manner as follows: (1) Compute a gradient map this gradient computation must performed in two orthogonal directions using Sobel mask. (2) Incorporate the surround influence on the gradient map this can be implemented as a convolution operation with an appropriate isotropic mask as shown in Fig. 4. (3) Binaries the output of the second stage using a standard procedure (Joshi and Sivaswamy 2004). For two-column wide figures use.

2.12.2 Invariant features

Features associated with images are called invariant if they are not affected by certain changes regarding the object viewpoint. It is widely accepted that invariant features would be independent of modifiers such as translation, scaling, rotation and light conditions. Ideally invariant features should recognize objects whose geometry can change either because the object is moving in relation to the camera is articulated or because different viewpoints cause different patterns in 2D image. Usually these modifiers are not independent of each other and therefore they often happen simultaneously. It is also agreed that there is not truly pure invariant feature. Rather there are features that are more or less robust to none or more modifiers (Barczak and Dadgostar 2005). The need for invariant features rises in many practical problems as illustrated in the following example. 1. Speech Recognition: the adopted features should be independent of speakers. Hence features should be insensitive to speaker dependent data. 2. Speaker Recognition: features should be invariant to spoken speech. 3. Image Recognition: features should be invariant under rotation, translation, scaling and the illumination angle used in generating the image (Abd Alrazak 2004).

2.12.3 Complex moments (CMs)

The notation of complex moment was introduced by Abo-Mustafa and Psaltis (1984) as a simple and straightforward way to derive moment invariants. The complex moment of order \( m \) is defined as:

\[
C_m = \int \int (x + iy)^m \mu(x, y) dxdy
\]

where \( i = \sqrt{-1} \) and \( \mu(x, y) \) is the real image intensity function. The complex moments have been proposed as a solution to different pattern recognition problems. The CMs are very simple and quite powerful in providing an analytic characteristic for moments invariant. Moment invariant is a set of moment values extracted from the image state such that their values are invariant to rotation of the image data the value of CM could be considered as moments invariant that can be computed from group of CMs for the same object at different
resolution (Musa 1998). Moment invariant can be used as a feature of classification and recognition of an object. Complex moments have two parts. Real part and imaginary part however the computation of their values decomposes into two directions; x-axis moment which represents real part direction and y-axis moment for the imaginary part direction (Musa 1998).

It well known that the moment sets can offer a powerful description on the geometrical distribution of the material within any region of interest. The low order of complex moments has meanings very relevant to some well known physical quantities (Musa 1998). Zero-order moments represent the total mass of the image. 1st order moments together with zero-order moments assign the centre of the mass of the image. 2nd order moments together with zero-order moment of inertia. 3rd order and 4th order moments are used for computing statistical quantities known as skews and kurtosis, respectively. While the higher nth-order moments give additional statistical and structural information about the image. The computation for complex moment should involve the calculation of its real and imaginary components. Then nth-order complex moment (Mi) for the hand image of size \((n \times m)\) is calculated according to the following equation:

\[
M_i = \sum_{x=0}^{n-1} \sum_{y=0}^{m-1} Z_n^i a(x, y)
\]

where \((i)\) indicates moments order

\[Z_n = X_n + iY_n\]

is a complex number \(a(x, y)\) represent a pixel value at the position, \((x, y)\) may be (ON or OFF) pixel The calculation of complex moments may require long computation time, to reduce this the relation below will be used:

\[
Z^n = Z^{n-1} \cdot Z
\]

where \(Z = x + iy\) is the complex number When the real and imaginary parts of \(Z^n\) and \(Z^{n-1}\) are assumed complex numbers could be written as:

\[
Z^n = R_n + iI_n
\]

and

\[
Z^{n-1} = R_{n-1} + iI_{n-1}
\]

By taking into consideration the case \(Z^0\) and \(Z^1\), it is simple to calculate that \(R_0 = 1\); \(I_0 = 0\); \(R_1 = X\); \(I_1 = Y\) Substituting the value of \(Z_n, Z_{n-1}\) and \(Z_1\) in (7) yields:

\[
R_1 = R_{n-1}X - I_{n-1}Y
\]

\[
I_1 = I_{n-1}X + R_{n-1}Y
\]

These equations indicate that knowing the components of \(Z^{n-1}\) will be directly used to compute the components of \(Z^n\) (Musa 1998).

2.13 Artificial neural networks

An Artificial neural network is an information processing system that has certain performance characteristics in common with biological neural networks. Artificial neural networks have been developed as generalizations of mathematical models of human cognition or neural biology based on the assumptions that (Fausett 1994). 1. Information processing occurs at many simple elements called neurons. 2. Signals are passed between neurons over connection links. 3. Each connection link has associated weight, which in a typical neural net, multiplies
the signal transmitted. 4. Each neuron applies an activation function (usually nonlinear) to its net input (sum of weighted input signals) (Fausett 1994; Picton 2000).

2.14 Artificial neuron

A neuron is an information processing unit that is fundamental to the operation of neural network. There are three basic elements of the neuron model, as described here: 1. A set of synapses or connecting links, each of which is characterized by a weight or strength of its own. Specifically, a signal $x_j$ at the input of synapses $j$ connected to neuron $k$ is multiplied by the synaptic weight. 2. An adder for summing the input signals, weighted by the respective synapses the neuron; the operations described here constitute a linear combiner. 3. An activation function for limiting the amplitude of the output of a neuron. The activation function is also referred to, in the literature as a squashing function that is squashes (limits) the permissible amplitude range of the output signal to some finite value. Typically the normalized amplitude range of the output of a neuron is written as the closed unit interval $[0, 1]$ or alternatively $[-1, 1]$ (Haykin 1994).

2.15 Types of activation functions

The basic operation of an artificial neuron involves summing its weighted input signal and applying an output or activation function. For the input unit this function is the identity function,

I. Identity function

$$f(X) = X \text{ for all } X$$  \hspace{1cm} (14)

Single layer nets often use a step function to convert the net input which is a continuously valued variable to an output unit that is a binary (1 or 0) or bipolar (1 or $-1$) signal. The binary step function is also known as the threshold function (Fausett 1994).

II. Binary step function (with threshold $\theta$):

Sigmoid functions (S-shaped curves) are useful activation functions. The logistic function and the hyperbolic tangent functions are the most common. They are especially advantageous for use in neural nets trained by back-propagation. If the range of the function is between 0 and 1 it is called the binary sigmoid, but if the range of the function is between $-1$ and 1 it is called the bipolar sigmoid (Fausett 1994).

III. Binary sigmoid

$$f(x) = \frac{1}{1 - \exp(-\delta x)}$$  \hspace{1cm} (15)

$$g(x) = 2f(x) - 1 = \frac{2}{1 + \exp(-\delta x)} - 1$$  \hspace{1cm} (16)

2.16 Neural networks architectures

The arrangement of neuron into layers and the connection patterns within and between layers is called the network architecture. Neural network are often classified as single layer or multilayer (Picton 2000).
2.16.1 Single-layer neural networks

A single layer network has one layer of connection weights. Often the units can be distinguished as input units which receive signals from the outside world and output units from which the response of the network can be read. In the typical single layer networked the input units are fully connected to output units but are not connected to other input units and the output units are not connected to other output units (Fausett 1994).

2.16.2 Multi-layer network

A multi-layer network is a network with one or more layers (or levels) for nodes (the so-called hidden units) between the input units and the output units there is a layer of weights between two adjacent levels of units (input, hidden, and output). Multilayer networks can solve more complicated problems than can single layer networks but training may be more difficult. However in some cases training may be more successful because it is possible to solve a problem that a single layer network cannot be trained to perform correctly at all. Multilayer Perceptron (MLP) neural network is a good tool for classification purposes (Fausett 1994; Picton 2000).

2.17 Back-propagation learning algorithm

Basically the error back-propagation process consists of two passes through the different layers of the network a forward pass and a backward pass (Haykin 1994; Kinnebrock 1995). The algorithm is as follows (Fausett 1994):

(1) Step 0. Initialize weights. (Set to small random values)
(2) Step 1. While stopping condition is false do steps 2–9
(3) Step 2. For each training pair do steps 3–8
Feed-forward:

(4) Steps 3. Each input unit \( (X_i, i = 1, \ldots, n) \) receives input signal \( X_i \) And broadcasts this signal to all units in the layer above (the Hidden units).
(5) Steps 4. Each hidden unit \( (Z_j, j = 1, \ldots, p) \) sums its weighted input signals,

\[
Z_{in} = V_{oj} + \sum_{i=1}^{n} X_i V_{ij}
\]  

(17)

\( V_{0j} \) Bias on hidden unit j.
\( V_{ij} \) Weight between input unit and hidden unit. Applies its activation function to compute its output signal

\[
Z_j = f(Z_{inj})
\]  

(18)

And send this signal to all units in the layer above (output units).
(6) Steps 5. Each output unit \( (Y_k, k = 1, \ldots, m) \) sums its weighted input signals,

\[
y_{in_k} = W_{ok} + \sum_{j=1}^{p} Z_j W_{jk}
\]  

(19)
$W_{ok}$  Bias on output unit $k$.
$W_{jk}$  Weight between hidden unit and output unit.
And applies its activation function to compute its output signal,

$$y_k = f(y_{in_k})$$  \hspace{1cm} (20)

Back propagation of error:
(7) Step 6. Each output unit $(Y_k, k = 1, \ldots, m)$ receives a target pattern corresponding to the input training pattern, computes its error information term, computes its error information term,

$$\delta_k = (t_k - y_k)f(y_{in_k})$$  \hspace{1cm} (21)

calculates its weight correction term (used to update $W_{jk}$ later),

$$\Delta W_{jk} = \delta_k Z_j$$  \hspace{1cm} (22)

Calculates its bias correction term (used to update $W_{ok}$ later ),

$$\Delta W_{ok} = \delta_k$$  \hspace{1cm} (23)

And sends $\delta_k$ to units in the layer below.

(8) Step 7. Each hidden unit $(z_j, j = 1, \ldots, p)$ sums its delta inputs (from unit in the layer above),

$$\partial_{in_j} = \sum_{k=1}^{m} \partial_k W_{jk}$$  \hspace{1cm} (24)

Multiplies by the derivative of its activation function to calculate its error information term,

$$\partial_j = \partial_{in_j} f(Z_{in_j})$$  \hspace{1cm} (25)

Calculates its weight correction term (used to update $V_{ij}$ later ),

$$\Delta V_{ij} = \partial \delta_j X_i$$  \hspace{1cm} (26)

And calculates its bias correction term (used to update $V_{oj}$ later),

$$\Delta V_{oj} = \partial \delta_j$$  \hspace{1cm} (27)

Update weights and biases:
(9) Step 8. Each output unit $(Y_k, k = 1, \ldots, m)$ updates its bias and weights $(j = 0, \ldots, p)$:

$$W_{jk}(new) = W_{jk}(old) + \Delta W_{jk}$$  \hspace{1cm} (28)

Each hidden unit $(Z_j, j = 1, \ldots, p)$ updates its bias and weights $(i = 0, \ldots, n)$:

$$V_{ij}(new) = V_{ij}(old) + \Delta V_{ij}$$  \hspace{1cm} (29)

(10) Step 9. Test stopping condition.
2.18 Advantages of neural computing (Symonidis 2000)

There are a variety of benefits that an analyst realizes from using neural networks in their work:

1. Pattern recognition is a powerful technique for harnessing the information in the data and generalizing about it. Neural nets learn to recognize the patterns which exist in the data set.
2. The system is developed through learning rather than programming. Programming is much more time consuming for the analyst and requires the analyst to specify the exact behaviour of the model. Neural nets teach themselves the patterns in the data freeing the analyst for more interesting work.
3. Neural networks are flexible in a changing environment. Rule based systems or programmed systems are limited to the situation for which they were designed—when conditions change they are no longer valid. Although neural networks may take some time to learn a sudden drastic change they are excellent at adapting to constantly changing information.
4. Neural networks can build informative models where more conventional approaches fail. Because neural networks can handle very complex interactions they can easily model data which is too difficult to model with traditional approaches such as inferential statistics or programming logic.
5. Performance of neural networks is at least as good as classical statistical modelling, and better on most problems. The neural networks build models that are more reflective of the structure of the data in significantly less time.

3 Methodology

This will introduce a proposed vision-based static hand gesture recognition algorithm to recognized the chosen set of six static hand gestures as commands used for HCI, these gestures are defined as: (Open, Close, Cut, Paste, Maximize, Minimize). The proposed algorithm consists of three basic stages: pre-processing feature extraction and classification. As known the main problems of 2D object recognition are the size changing, translation of position and rotation by angle from the principle axes. In this algorithm two methods are proposed in the first method the hand contour is extracted as a geometric feature this method treats the problems of size changing and translation (in some cases). The second method treats the problem of rotation in addition to previous problems by using the hand complex moments feature. In the pre-processing segmentation. The extracted features are entered to the last stage (neural networks using the supervised back-propagation learning algorithm), and this stage is responsible for recognizing and deciding to which class the hand gesture belongs. The main efforts in this work and main contribution are proposed in the feature extraction and classification parts. Its known that to have a robust classification its necessary to choose properly the features as well as the correct way to present the features to the classifier. This work explores mainly this topic finding proper features and proper way to represent them that matches the classifier characteristics.

The proposed hand gesture recognition algorithm consists of two methods: neural network with hand contour neural network with hand complex moments. Either of these methods is formed by four sequential stages: hand image capture, image preprocessing, feature extraction, and classification, as seen in Fig. 5.
The role of the preprocessing module is to segment the pattern of interest (hand) from the background, removing noise and any other operation which will contribute to defining a compact representation of the pattern. In the training phase, the feature extraction module finds the appropriate features for representing the input patterns and the classifier is trained using the back-propagation algorithm. In the testing phase the trained classifier will assign the input pattern (test hand gesture image) to one of the pattern classes under consideration based on the measured features. The next sections present a detailed explanation of the proposed methods.

3.1 Hand gestures image database

The starting point of this study was the creation of a database with all the hand gesture images that would be used for training and testing. The construction of such a database is clearly dependent on the application. Each gesture represents a gesture command mode. These commands are widely used in various application programs. Therefore, these gestures are allowed to be flexible and natural so that they can successfully be applied to a wide range of people and situations. The gestures images are real images with different sizes acquired using digital camera and taken from one subject. The database consists of 30 images for training set, with five samples for each gesture, and 84 images derived from the two remaining samples of 6 gesture classes with scaling effect, translation, and rotation.

3.2 Neural network with hand contour

As shown in Fig. 6, the recognition process consists of two phases: training (learning) and classification (testing):

A. Training phase

(1) Capturing Gesture Image from training set.
(2) Segmentation to isolate hand object from the background.
(3) Noise reduction to remove any noise from the segmented image.
(4) Edge detection to find the hand gesture image boundaries.
(5) Contour detection as a geometric feature.
(6) Calculating the height offset and width offset as a general feature.
(7) Saving feature image as a training pattern.

B. Testing phase

(1) Capturing Gesture Image from testing set.
(2) Segmentation to isolate hand object from the background.
(3) Noise reduction to remove any noise from the segmented image.
(4) Edge detection to find the gesture image boundaries.
(5) Contour detection as a geometric feature.
(6) Calculating height offset and width offset as a general feature.
(7) Recognized by neural networks.
3.2.1 Preprocessing

The primary goal of the preprocessing stage is to ensure a uniform input to the classification network. This stage includes hand segmentation to isolate the foreground (hand gesture) from the background removing any noises caused by segmentation process using special filters. This stage also includes edge detection to find the final shape of the hand.

3.2.2 Hand segmentation

The hand image is segmented from the background. The segmentation process, however, should be fast, reliable, and consistent, and able to produce the best quality output possible giving the constraints, which must produce an image suitable to recognize the gesture of the hand. Gesture recognition requires a more accurate segmentation. In this work a Thresholding algorithm is used for segmentation of the gesture image. The pseudo code of the algorithm is shown in Fig. 7.

3.2.3 Noise reduction

Once the hand gesture image has been segmented, a special filter is applied. The goal of applying a filter is to eliminate all the single white pixels on a black background, and all the single black pixels on a white foreground. To accomplish this goal, a median filter is applied to the segmented image. In Fig. 8 the pseudo code for the median filter algorithm is given.

3.2.4 Edge detection

For recognizing static gestures in our system, the model parameters are derived from description of the shape, and the boundary of the hand is extracted for further processing. Therefore,
several different edge-detector operators were tried including Sobel and Prewitt. Another attempt was using a Laplacian edge detector which gives thin edges. Finally, Sobel has been chosen for its very good results.

3.3 Gesture feature extraction

The objective of the feature extraction stage is to capture and discriminate the most relevant characteristics of the hand gesture image for recognition. The selection of good features can strongly affect the classification performance and reduce the computational time. These selected features, consequently, result in an easier classification task. The features used must be suitable for the application and the applied classifier. In a proposed method two types of features are extracted, namely, the hand contour as a geometric feature and height width offsets of the hand image as assistant general features.

3.3.1 Geometric feature (hand contour)

One of the most important geometric features for static hand gesture recognition is hand contour (Fig. 9). Figure 10 shows the pseudo code for contour detection algorithm.

Once the contour map is produced, feature image scaling takes place. Feature image scaling is a very simple method which reduces the feature image size by selecting several rows and columns. For all the images in the training set, a scaling is performed. The result is feature image with 32, the number of the rows, and 32, the number of the columns. The pseudo code of the process is displayed in Fig. 11. After the feature image scaling, a feature gesture image must be prepared before entering the last stage, i.e. classifier stage. The preparing step includes shifting the hand gesture section to the origin point (0.0) to solve translation problem.
Fig. 9  Shows the pseudo code for Sobel edge detection algorithm

Input: Segmented gesture image with Sobel edge detector
Output: Edge magnitude of the gesture image
FOR RM=1 To MaskWidth
FOR CM= 2 To MaskHeight
FULL RowMask
FULL ColumnMask
EDN FOR
END FOR
FOR each pixel in the Gesture Image
CONVOLUTE with RowMask
RETURN (S1)
CONVOLUTE with ColumnMask
RETURN (S2)
END FOR
EdgeMagnitude = SQRT (S1^2+S2^2)

Fig. 10  The pseudo code for contour detection

Input: Gesture Image has passed the Segmentation and Edge Detection Process
Output: A Gesture Image with Contour Detection
FOR RM=1 To MaskWidth
FOR CM= 1 To MaskHeight
FULL Contour_Mask
END FOR
END FOR
FOR each pixel in the Gesture Image
CONVOLUTE Gradients from Edge Detector with Contour_Mask
RETURN Surround_Influence Image
END FOR
FOR each pixel in the Surround_Influence Image
IF pixel value > “specific threshold” THEN
Mark each pixel in the Surround_Influence Image as ContourELSE
Mark it as lack Contour.
END IF
END FOR
END

Fig. 11  The pseudo code for reducing feature gesture image from different size to (32 × 32)

Input: Feature Gesture image with variable size
Output: Feature Gesture image with fixed size (32 x32)
SET NewWidth TO 32
SET NewHeight TO 32
DW = (NewWidth / OldWidth)
DH = (NewHeight / OldHeight)
FOR Hi=0 TO NewHeight-1
FOR Wi=0 TO NewWidth-1
NewX=Round (Wi/DW)
New Y=Round(Hi/DH)
GET Value if is inside (NewX,NewY)
PUT in position (Wi,Hi)
END FOR
END FOR

3.3.2 General features

General features describe the hand image offsets that serve as an additional information such as height and width of the hand image. The general features will be represented as a (6 × 6) matrix: its indices are (2) to power (0, 1, 2, 3, 4, 5) as shown in Fig. 12. This matrix will hold values of two (ones) with all remaining zero values. The two (ones) positions represent height
Fig. 12 Example of general futures ($6 \times 6$) matrix

<table>
<thead>
<tr>
<th></th>
<th>2$^1$</th>
<th>2$^2$</th>
<th>2$^3$</th>
<th>2$^4$</th>
<th>2$^5$</th>
<th>2$^6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2$^0$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2$^1$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2$^2$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2$^3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2$^4$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2$^5$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

and width of the hand gesture. For example as shown in the figure above if the hand gesture has a width equal to (8) or near to this value (by using threshold value), and the height equal to (32) or near to this value, the position of ($2^3$) will hold a (one) value, the same thing for the height the position of ($2^5$) will hold a (one) value. This feature matrix is compounded with contour feature vector as a composed features to produce a new feature vector with (1060) elements then it is passed to the back-propagation neural network as its input vector.

3.4 Neural network based classifier

The last stage of the proposed system is the classification. The right choice of the classification algorithm to be used in a gesture recognition system is highly dependent on the properties and the format of the features that represent the gesture image. In this work a standard back-propagation neural network is used to classify gestures. The network consists of three layers; the first layer consists of neurons that are responsible for inputting a hand gesture sample into the neural network. The second layer is a hidden layer. This layer allows neural network to perform the error reduction necessary to successfully achieve the desired output. The final layer is the output layer with one node per class. Typically the number of neurons in this layer is determined by the size of the set of desired outputs, with each possible output being represented by a separate neuron. The structure of this particular back-propagation neural network is illustrated in Fig. 13.

There are (6) outputs from the neural network each output represents index for one of the six hand gesture images classes. The highest index value (in the testing phase) will represent the recognized gesture image. For the recognition process, five neural networks with same structure are used. The parameters for the multilayer neural networks are shown in Table 1. In back-propagation as in Sect. 2.16 a hand gesture sample is propagated through the multilayer neural network, producing an output. This output is compared to the desired output, giving an error rate for the output layer. Because the error rate of a neuron is a function of the error rates of all the units that use its output, the error rates of the layer directly below the output layer can now be found. These error rate calculations will continue to propagate through the network in a backward fashion, until the error rates for all the neurons have been found. Each neuron will then make slight weight adjustments in order to minimize its error signal. This pattern is repeated until the error rate of the output layer reaches a minimum value. This process is then repeated for the next input value, until all of the input values have been processed (Fig. 14).

3.5 Neural networks with complex moments

Figure 15 shows the block diagram of the second proposed method for recognition algorithm. The figure demonstrates the steps of preprocessing operations. The sample for each hand gesture is trimmed by clipping the body of the hand gesture for empty columns.
and lines, normalizing the size of the clipped gesture to standard size. In a feature extraction stage, the complex moment value is extracted.

3.6 Preprocessing

In addition to segmentation and noise reduction processes as in previous method, the preprocessing involves another operations that includes image trimming, scaling and coordinate normalization.

3.6.1 Image trimming

The image of hand gesture may include an additional empty lines and columns that have no data (space lines) these empty lines should be eliminated by tracing from outside margins towards inside and stopping at the first occurrence of the (On-pixel) at each side of the four edges.

3.6.2 Image scaling

The dimensions of the hand gesture images are varying due to capturing process therefore the image size is adjusted to fixed size \((250 \times 250)\) in order to facilitate the calculation of the next stage.

---

**Table 1** Parameters for the five multilayer neural network

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input layer</td>
<td>1,060 nodes</td>
</tr>
<tr>
<td>Hidden layer</td>
<td>100 nodes</td>
</tr>
<tr>
<td>Output layer</td>
<td>6 nodes</td>
</tr>
<tr>
<td>Stop error</td>
<td>0.01</td>
</tr>
<tr>
<td>Learning rate</td>
<td>0.9</td>
</tr>
</tbody>
</table>

**Fig. 13** Gesture recognition network architecture
3.6.3 Coordinate normalization

After image scaling the coordinate normalization process is take place to map the scaled hand image coordinates to the standard size ranging between $[-1, +1]$ by using the equations in previous section (Sect. 2.11) the origin point coordinates will be at the center of the image as shown in Fig. 16.

3.7 Feature extraction

In this proposed method for gesture recognition the complex moments are used as a moment invariant which is a set of moment values extracted from the image data that their values are invariant to scaling translation and rotation of the image data. Feature extraction stage includes the calculation of complex moments. For every hand gesture in the database, these features are calculated.

3.7.1 Complex moments calculation

For each hand gesture image the complex moments from (zero-order) to (ninth-order) will be calculated (by using the equations in Sect. 2.11 so each feature vector will have ten values, as shown in Fig. 17. With many experiments it found that this (ten values) is enough to represent each hand gesture and to make it recognizable. In order to make the complex moments values in a small range before they are entered to neural network classifier they should be scaled to small range from [0 to 1] this will facilitate the classification process.
3.7.2 Complex moments algorithm

The computation for complex moments illustrated in the pseudo code is as shown in Fig. 18.

3.8 Neural network based classifier

As in previous method a multilayer neural network (with the same structure) is used for recognizing the hand gesture. The input of this network is a feature vector with ten values, the parameters of this neural network are shown in Table 2. The remaining details about recognition process is the same as in Sect. 3.4.
Fig. 16 Central coordinates normalization

![Diagram showing central coordinates normalization](image)

Fig. 17 The complex moments feature vector

![Diagram showing complex moments feature vector](image)

Fig. 18 Complex moments calculation algorithm

```
Input: Normalized Gesture Image
Output: Feature vector with (n) complex moments values
FOR MomOrder= 0 To MomentNumber
    SET Mom(MomentOrder).RealPart=0
    SET Mom(MomentOrder).ImgPart=0
END FOR
SET Real0=1
SET Imag0=0
FOR Row=0 TO ImageHeight-1
    FOR Col=0 TO ImageWidth-1
        Mom(0).RealPart =Mom(0).RealPart + Array(Row,Col)
        END FOR
    END FOR
FOR MomOrder=1 To MomentNumber
    FOR Row=0 TO ImageHeight-1
        FOR Col=0 TO ImageWidth-1
            RealT=Real0*Row-Imag0*Col
            ImgT=Imag0*Row+Real0*Col
            Mom(MomOrder).RealPart=Mom(MomOrder).RealPart + RealT*Array(Row,Col)
            Mom(MomOrder).ImgPart=(MomOrder).ImgPart + ImgT*Array(Row,Col)
        END FOR
    END FOR
MomValue(MomOrder)=SQRT(SQR(MomOrder).RealPart + SQR(Mom(MomOrder).ImgPart))
Real0=RealT
Imag0=ImagT
END FOR
```
Table 2  Parameters for the five multilayer neural networks used in hand complex moments

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input layer</td>
<td>10 nodes</td>
</tr>
<tr>
<td>Hidden layer</td>
<td>100 nodes</td>
</tr>
<tr>
<td>Output layer</td>
<td>6 nodes</td>
</tr>
<tr>
<td>Stop error</td>
<td>0.01</td>
</tr>
<tr>
<td>Learning rate</td>
<td>0.9</td>
</tr>
</tbody>
</table>

4 Experiments and results

Hand gesture recognition is a decision concerning the gesture of the user which cannot be made unless the gesture or recognition algorithm passes through several operations on unknown hand gesture. These operations are required before deciding to which class the gesture belongs. Hand gesture recognition algorithm includes two methods, extracting the hand contour as a geometric feature and treating the problem of rotation in addition to previous problems using the hand complex moments feature. Each of these methods consists of two phases: training phase and testing phase. The training phase works on the gesture used to train neural networks while the testing phase works on the hand gestures used to decide to which class they belong which will not be used in the training phase. Both phases consist of two main processes which are preprocessing and feature extraction. In this section the results obtained from applying the proposed hand gesture recognition algorithm to both methods and the effects of each of the processes are presented. For each hand gesture of the six selected gesture classes, seven samples are taken under different light conditions (natural and artificial light conditions). In the following sections the effect and the performance of the algorithm will be presented and discussed.

4.1 Preprocessing stage effects (contour with NN)

In this stage many processes are performed on the hand gesture image to make it ready for the following feature extraction stage. The effects of these operations will be explained in the following subsections.

4.1.1 Hand gesture segmentation

All the techniques that are used in this paper are based on hand shape. The acquiring color gesture image is segmented to isolate the foreground hand from the background.

4.1.2 Noise reduction

The segmented hand image may contain some noises that will affect the result values produced by the feature extraction stage. Hence, the values of the computed features will be different for the same image if it contains noise. So using the median filter will reduce the noise as much as possible.

4.1.3 Edge detection

The Edge detection process is performed using Sobel operator objects presented in an image.
4.2 Feature extraction

The hand gesture image that has passed through image preprocessing stage is fed to the feature extraction stage to compute the feature information about the image. As soon as contour feature is computed the image size is adjusted so that each hand gesture image size becomes \((32 \times 32)\). Image resizing makes the system faster, this operation will reduce the negative affects of the size change. the general features (height offset and width offset) will be computed implicitly.

4.3 Training phase

In this phase, the composite feature vectors computed earlier and stored in a feature images database are fed to the next stage of our system as inputs. These feature vectors are used to train the neural networks. The learning process for the five multilayer neural networks is accomplished by using the following parameters as shown in Table 3. The neural networks are trained through successive epochs (iteration), after each epoch the square error over the validation set is computed. the training results as shown in Fig. 19, which show the convergence of learning algorithm for back-propagation neural networks and the learned samples with respect to the number of epochs for each network.

4.4 Testing phase

After training neural networks, performance is estimated by applying the testing set to the network inputs and computing the classification errors. The used activation function is binary-sigmoid which holds outputs constantly between 0 and 1. Then the test gesture feature image will be entered into the first neural network. In this phase, if the network succeeds to recognize the gesture, the test operation is stopped. If this network does not recognize the gesture features, the second network will be activated and so on. If all networks fail to identify the features, gesture not recognized message will appear to announce the failure in recognition. In this phase, 56 hand gesture images are used to test the system with different light conditions and with scaling and translation effects. The system now is able to recognize and classify any unknown gestures if they are in the original database. Each gesture has a table of recognition results and with neural network outputs for one gesture image the performance of the proposed system is evaluated based on its ability to correctly recognized gestures to their corresponding input gestures, the metric that is used to accomplish this job is called the recognition rate. the recognition rate is defined as the ratio of the number of correctly

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>1,060 nodes</td>
</tr>
<tr>
<td>Hidden layer</td>
<td>100 nodes</td>
</tr>
<tr>
<td>Output</td>
<td>6 nodes</td>
</tr>
<tr>
<td>No. of images for each training set</td>
<td>6</td>
</tr>
<tr>
<td>Stop error</td>
<td>0.01</td>
</tr>
<tr>
<td>Learning rate</td>
<td>0.9</td>
</tr>
</tbody>
</table>
recognized gestures to the total number of input gestures as shown in equation

\[
\text{Recognition rate} = \frac{\text{No. of correctly recognized gestures}}{\text{Total number of input gestures}} \times 100\% \quad (30)
\]

A summary of all recognition results and the recognition rates for each of the six hand gestures is presented by Table 4, and the recognition rates with each class are shown in Fig. 20.

4.5 Preprocessing stage effect (complex moments with NN)

The preprocessing stage in this method include: image trimming followed by normalization process. the effect of these operation will be presented in the next sections.
Table 4 Summary of the recognition results and the recognition rates used in hand contour method

<table>
<thead>
<tr>
<th>Gesture meaning</th>
<th>Number of test gesture</th>
<th>Successful recognition</th>
<th>Recognition rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>8</td>
<td>4</td>
<td>50</td>
</tr>
<tr>
<td>Close</td>
<td>9</td>
<td>7</td>
<td>77</td>
</tr>
<tr>
<td>Cut</td>
<td>9</td>
<td>8</td>
<td>88</td>
</tr>
<tr>
<td>Paste</td>
<td>10</td>
<td>6</td>
<td>60</td>
</tr>
<tr>
<td>Maximize</td>
<td>10</td>
<td>8</td>
<td>80</td>
</tr>
<tr>
<td>Minimize</td>
<td>10</td>
<td>7</td>
<td>70</td>
</tr>
<tr>
<td>Total</td>
<td>56</td>
<td>40</td>
<td>70.83</td>
</tr>
</tbody>
</table>

Fig. 20 Percentages of correct recognition using hand contour

4.5.1 Image trimming effect

The hand gesture filtered image may contain unused space surrounding the hand gesture so the image trimming process is used to extract the hand gesture from its background.

4.5.2 Coordinate normalization

After scaling each image size to fixed size (250 × 250) the coordinates for the hand image are normalized between [−1, +1].

4.6 Complex moments calculation

Each hand gesture in the training set will have a feature vector of ten values and these values represent the complex moments starting with zero order up to nine order. Tables 5 and 6 present an example of the results of these computations, before and after normalization process.

4.7 Training phase

After the computation of feature vectors, each containing (10) translation, scaling and rotation invariant elements characterized the complex moments for the hand gesture, five similar neural network classifiers are trained with a data set containing (30) features vectors, these vectors were computed from a set of (5) examples for each hand gesture performed by one
subject. The learning process for the back-propagation neural networks is accomplished by using the following parameter for each one as shown in Table 7. The training results as shown in Fig. 21 show the convergence of learning algorithm for three back-propagation neural networks (first, fourth and fifth) and the learned samples with respect to the number of epochs for each network.

4.8 Testing phase

After training the neural networks, performance is estimated by applying the testing set on the networks inputs, and computing the classification error. the activation function used is binary-sigmoid which holds outputs always between 0 and 1. The testing process as in the Sect. 4.4 of the previous method. in this phase, (84) hand gesture images are used to test the system. Each one of the six hand gestures has a number of samples in different light conditions and with effects of scaling, translation and rotation. Each gesture has a table of recognition results and with neural network outputs for one gesture image (as am example) a summary of all the recognition results and the recognition rates for each of the six static

### Table 5 Complex moments values

<table>
<thead>
<tr>
<th>Moment order</th>
<th>M0</th>
<th>M1</th>
<th>M2</th>
<th>M3</th>
<th>M4</th>
<th>M5</th>
<th>M6</th>
<th>M7</th>
<th>M8</th>
<th>M9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>45660</td>
<td>0.11310</td>
<td>2167.2</td>
<td>2330.2</td>
<td>538.9</td>
<td>1448.3</td>
<td>55.3</td>
<td>57.6</td>
<td>1691.2</td>
<td>8819</td>
</tr>
<tr>
<td>Close</td>
<td>44652</td>
<td>0.02355</td>
<td>953.7</td>
<td>1248.5</td>
<td>2342.8</td>
<td>1971.3</td>
<td>1261.6</td>
<td>731.1</td>
<td>1656.5</td>
<td>905.8</td>
</tr>
<tr>
<td>Cut</td>
<td>30916</td>
<td>0.16550</td>
<td>7553.2</td>
<td>1300.5</td>
<td>2793.3</td>
<td>1545.0</td>
<td>3230.4</td>
<td>1249.5</td>
<td>4125.3</td>
<td>1247.2</td>
</tr>
<tr>
<td>Paste</td>
<td>37186</td>
<td>0.14053</td>
<td>4680.8</td>
<td>1900.6</td>
<td>965.4</td>
<td>1161.9</td>
<td>1418.5</td>
<td>1190.7</td>
<td>408.5</td>
<td>616.5</td>
</tr>
<tr>
<td>Maximize</td>
<td>28965</td>
<td>0.17819</td>
<td>7142.2</td>
<td>3383.1</td>
<td>8998.0</td>
<td>752.4</td>
<td>7099.1</td>
<td>2109.6</td>
<td>10954.9</td>
<td>1894.8</td>
</tr>
<tr>
<td>Minimize</td>
<td>43866</td>
<td>0.08460</td>
<td>5710.4</td>
<td>1938.4</td>
<td>3020.8</td>
<td>2963.1</td>
<td>2404.2</td>
<td>2420.0</td>
<td>3076.2</td>
<td>3117.8</td>
</tr>
</tbody>
</table>

### Table 6 Complex moments values after normalization

<table>
<thead>
<tr>
<th>Moment order</th>
<th>M0</th>
<th>M1</th>
<th>M2</th>
<th>M3</th>
<th>M4</th>
<th>M5</th>
<th>M6</th>
<th>M7</th>
<th>M8</th>
<th>M9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>1</td>
<td>0.63</td>
<td>0.28</td>
<td>0.68</td>
<td>0.05</td>
<td>0.48</td>
<td>0.07</td>
<td>0.02</td>
<td>0.15</td>
<td>0.28</td>
</tr>
<tr>
<td>Close</td>
<td>0.97</td>
<td>0.13</td>
<td>0.12</td>
<td>0.36</td>
<td>0.26</td>
<td>0.66</td>
<td>0.17</td>
<td>0.30</td>
<td>0.15</td>
<td>0.29</td>
</tr>
<tr>
<td>Cut</td>
<td>0.67</td>
<td>0.92</td>
<td>1</td>
<td>0.38</td>
<td>0.31</td>
<td>0.52</td>
<td>0.45</td>
<td>0.51</td>
<td>0.37</td>
<td>0.40</td>
</tr>
<tr>
<td>Paste</td>
<td>0.81</td>
<td>0.78</td>
<td>0.61</td>
<td>0.56</td>
<td>0.10</td>
<td>0.39</td>
<td>0.19</td>
<td>0.49</td>
<td>0.03</td>
<td>0.19</td>
</tr>
<tr>
<td>Maximize</td>
<td>0.63</td>
<td>1</td>
<td>0.94</td>
<td>1</td>
<td>0.25</td>
<td>1</td>
<td>0.87</td>
<td>1</td>
<td>0.60</td>
<td></td>
</tr>
<tr>
<td>Minimize</td>
<td>0.96</td>
<td>0.47</td>
<td>0.75</td>
<td>0.57</td>
<td>0.33</td>
<td>1</td>
<td>0.40</td>
<td>1</td>
<td>0.28</td>
<td>1</td>
</tr>
</tbody>
</table>

### Table 7 Parameters of back-propagation neural networks

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>10 lodes</td>
</tr>
<tr>
<td>Output</td>
<td>6 nodes</td>
</tr>
<tr>
<td>No. of images for each training set</td>
<td>6</td>
</tr>
<tr>
<td>Stop error</td>
<td>0.01</td>
</tr>
<tr>
<td>Learning rate</td>
<td>0.9</td>
</tr>
</tbody>
</table>
hand gestures is presented in Table 8, these results are obtained by using the formula (30) and the recognition rates with each class is shown in Fig. 22.

5 Conclusions and suggestions for future work

This paper addresses the problem of static hand gesture recognition. In this research, a computer vision algorithm is proposed that recognizes the six selected static hand gestures, namely, Open, Close, Cut, Paste, Maximize, Minimize, used for HCI. A supervised back-propagation multi-layer feed-forward neural network is used for recognition, with two types of features: hand contour and the hand complex moments. The recognition is made, however, without the need for using special hardware such as gloves. This section summarizes the thesis evaluating the main results, and showing the main contributions of this work. This section also presents possible future directions for further research on the topic.

It is difficult to find a highly accurate hand gesture recognition system that is capable of working under various conditions including varying illumination complex background and effects of scaling, translation and rotation by specific angles. However, this static hand gesture recognition algorithm for both methods, mentioned earlier, performs well in recognizing the six static hand gestures, with some instances of failure for both methods. In general, the conclusions resulted from the foregoing analyses are:
Table 8  Summary of the recognition results and the recognition rates used in hand complex moments

<table>
<thead>
<tr>
<th>Gesture meaning</th>
<th>Number of test gesture</th>
<th>Successful recognition</th>
<th>Recognition rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>15</td>
<td>15</td>
<td>100</td>
</tr>
<tr>
<td>Close</td>
<td>15</td>
<td>12</td>
<td>80</td>
</tr>
<tr>
<td>Cut</td>
<td>12</td>
<td>10</td>
<td>83.33</td>
</tr>
<tr>
<td>Paste</td>
<td>12</td>
<td>9</td>
<td>75</td>
</tr>
<tr>
<td>Maximize</td>
<td>15</td>
<td>14</td>
<td>93.33</td>
</tr>
<tr>
<td>Minimize</td>
<td>15</td>
<td>13</td>
<td>86.66</td>
</tr>
<tr>
<td>Total</td>
<td>84</td>
<td>70</td>
<td>86.38</td>
</tr>
</tbody>
</table>

Fig. 22  Percentages of correct recognition using complex moments

Neural network with hand contour The results obtained through applying this method are presented in Fig. 19. According to the results, this method takes a short time for convergence of learning algorithm of the neural networks. In Table 4, it can be observed that the method has sufficient accuracy with recognition rate of 70.83% but is not invariant to translation cases. Using pure hand contour is not enough for successful recognition results. It can be used to calculate feature vectors with parameters such as the area of hand shape and hand perimeters, and to find some relation between them.

Neural network with hand complex moments As displayed in Fig. 21, this method needs a long time for convergence of learning how to train the neural networks. The performance of this method, as shown in Table 8, with recognition rate of 86.38% and with effects of scaling and translation and rotation by specific angles, is significantly better than the previous method for recognizing the six gestures. For both methods, however, the same gesture in different light conditions is not correctly recognizable because the distance between the camera and the hand gesture is not fixed.

5.1 Suggestions for future work

Potential applications of a static hand gesture recognition algorithm include: recognition of sign language, controlling the TV by hand gestures, and HCI. Based on the work presented in this paper, the following problems are suggested for further researches in future:

Recognition of dynamic gestures Gestures such as waving or wagging a finger can make a HCI much more intuitive.
Use of both hands In virtual reality (VR), applications where a particular gesture of the right hand such as “extended index finger” is assigned the command of “move forward” gesture of the left hand can be used as modifier to regulate the speed.

Usage of flow menus To practically use the hand gestures in HCI, more gestural commands will be needed. Menus can be used to create gesture command sets. Some kinds of commands would be more reasonably input by static hand gestures (hand postures).

Usage of hybrid methods Such as Hidden Markov Model (HMM) with Artificial Neural Network (ANN).

References

Boussaid K, Behgdaddi A (1999) A contour detection method based on some knowledge of the visual system mechanisms, vision interface. Rrois-Rivieres, Canada


Winnemöller H (1999) Practical gesture recognition for controlling virtual environments. Project for Bachelor of Science (Honours) of Rhodes University