Real-Time Hand Gesture Detection and Recognition Using Bag-of-Features and Support Vector Machine Techniques

Nasser H. Dardas and Nicolas D. Georganas, *Fellow, IEEE*

***Abstract*—This paper presents a novel and real-time system for interaction with an application or videogame via hand gestures. Our system includes detecting and tracking bare hand in cluttered background using skin detection and hand posture contour comparison algorithm after face subtraction, recognizing hand gestures via bag-of-features and multiclass support vector machine (SVM) and building a grammar that generates gesture commands to control an application. In the training stage, after extracting the keypoints for every training image using the scale invariance feature transform (SIFT), a vector quantization technique will map keypoints from every training image into a unified dimensional histogram vector (bag-of-words) after K-means clustering. This histogram is treated as an input vector for a multiclass SVM to build the training classifier. In the testing stage, for every frame captured from a webcam, the hand is detected using our algorithm, then, the keypoints are extracted for every small image that contains the detected hand gesture only and fed into the cluster model to map them into a bag-of-words vector, which is finally fed into the multiclass SVM training classifier to recognize the hand gesture.**

***Index Terms*—Bag-of-features, grammar, hand gesture, hand posture, human computer interaction, K-means, object detection, object recognition, scale invariant feature transform (SIFT), support vector machine (SVM).**

I. Introduction

HAND GESTURES provide a natural and intuitive communication modality for human–computer interaction. Efficient human computer interfaces (HCIs) have to be developed to allow computers to visually recognize in real time hand gestures. However, vision-based hand tracking and gesture recognition is a challenging problem due to the complexity of hand gestures, which are rich in diversities due to high degrees of freedom (DOF) involved by the human hand. In order to successfully fulfill their role, the hand gesture HCIs have to meet the requirements in terms of real-time performance, recognition accuracy, and robustness against transformations and cluttered background. To meet these requirements, many gesture recognition systems used the help of colored markers or data gloves to make the task easier [1]. However, using of markers and gloves sacrifices the user's convenience. In this paper, we focus on bare hand gesture recognition without help of any markers and gloves.

Detecting and tracking hand gestures in a sequence of images help in extracting hand region. Thus, processing time will be reduced and accuracy will be increased as the features of that region will represent the hand gesture only. Skin color [39]–[41] is a significant image feature to detect and track human hands. However, color-based methods face the challenge of removing other objects with similar color such as face and human arm. To solve this problem, we proposed a new technique to detect hand gestures only using face subtraction, skin detection, and hand posture contour comparison algorithm. We used the Viola–Jones method [11] to detect face, and this method is considered the fastest and most accurate learning-based method. The detected face will be subtracted by replacing face area with a black circle. After subtracting the face, we detected the skin area using the hue, saturation, value (HSV) color model since it has real-time performance, and it is robust against rotations, scaling, and lighting conditions. Then, the contours of skin area were compared with all the loaded hand gesture contours to get rid of other skin-like objects existing in the image. The hand gesture area only was saved in a small image, which will be used in extracting the keypoints by scale invariance feature transform (SIFT) algorithm.

SIFT features, proposed by Lowe [2], are features (keypoints) extracted from images to help in reliable matching between different views of the same object, image classification, and object recognition. The extracted keypoints are invariant to scale, orientation and partially invariant to illumination changes, and are highly distinctive of the image. Therefore, the SIFT is adopted in this paper for the bare hand gesture recognition. However, SIFT features are of too high dimensionality to be used efficiently. We proposed to solve this problem by the bag-of-features approach [17], [18] to reduce the dimensionality of the feature space.

In the training stage, hand gestures training images can be represented by sets of keypoint descriptors, but the numbers of keypoints from the images are different and lack meaningful ordering. This creates difficulties for machine learning methods such as the multiclass support vector machine (SVM) classifier that require feature vectors of fixed dimension as input. To address this problem, we used the bag-of-features approach, which has several steps. The first step is extracting the features

Manuscript received November 17, 2010; revised April 13, 2011; accepted April 22, 2011. Date of publication August 15, 2011; date of current version November 9, 2011. The Associate Editor coordinating the review process for this paper was Dr. Sethuraman Panchanathan.

The authors are with the University of Ottawa, Ottawa, ON K1N 6N5, Canada (e-mail: ndard076@uottawa.ca.).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIM.2011.2161140

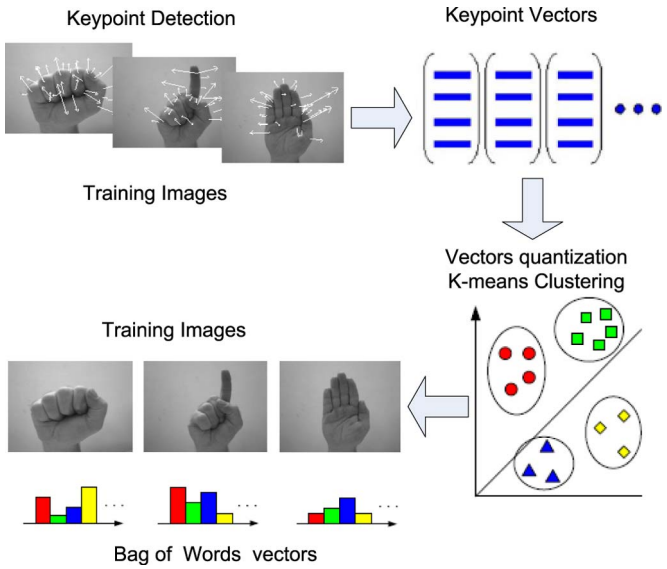


Fig. 1. Generating a bag-of-words.

(keypoints) of hand gesture training images using the SIFT algorithm. The next step is using the *vector quantization* (VQ) technique [3], which clusters the keypoint descriptors in their feature space into a large number of clusters using the K-means clustering algorithm and encodes each keypoint by the index of the cluster (codevector) to which it belongs. This VQ maps keypoints of every training image into a unified dimensional histogram vector after K-means clustering as shown in Fig. 1. Finally, each cluster is considered as a visual word (codevector) that stands for a particular local pattern shared by the keypoints in that cluster. Therefore, the clustering algorithm constructs a visual word vocabulary (codebook) representing several local patterns in the training images. The size of the vocabulary is determined by the number of clusters (codebook size), which can vary from hundred to over tens of thousands. Each training image can be described as a "bag-of-words" vector by mapping the keypoints to a visual words vector. With the described feature representation, we can train multiclass SVM classifier.

After the training stage, the testing stage can be run to recognize hand gestures captured from a webcam or a video file regardless of its resolution size. First, the hand gesture is detected using our approach. Then, a small image (50×50 pixels) that contains the detected hand gesture only will be used in extracting the keypoints to reduce time processing and increase the accuracy of recognition as the keypoints extracted will represent the hand gesture only and will be used as the input for the cluster and multiclass SVM classifier models, which were built in the training stage, to recognize the hand gesture. In this way, the background is subtracted, and the system will be robust against cluttered background.

After detecting and recognizing hand postures, our system will build a grammar that generates gesture commands, which can be used to control or interact with an application or a videogame.

The major contributions of this paper are as follows.

1) We proposed a new real-time approach for bare hand posture detection and tracking using face subtraction, skin detection, and hand posture contour comparison algorithm.
2) We have achieved real-time performance and accurate recognition for the detected hand postures using bag-of-features model and multiclass SVM.
3) We built a grammar that generates a large number of gesture commands used to control an application or videogame by tracking and monitoring the size or scale of the detected hand posture, its movement direction, and the transitions among postures.

The paper is organized as follows: Section II introduces related works; Section III describes our system in details, including the training stage to build the cluster and the SVM classifier models and the testing stage for recognition; Section IV provides experimental results; Section V compares performance of our approach with other approaches; Section VI explains how our system generates gesture commands; the last section gives the conclusion of our method.

II. Related Work

There are mainly two categories for Vision-based hand gesture recognition, which are the 3-D (D) hand model-based methods and the appearance-based methods [4]. Many approaches that used the 3-D hand model-based technique [26], [35]–[37] depend on the 3-D kinematic hand model with considerable *DOF* and calculate the hand parameters by comparison between the input frames and the 2-D appearance projected by the 3-D hand model. This will be suitable for realistic interactions in virtual environments. The 3-D hand model-based technique provides a rich description that permits a wide class of hand gestures. However, since the 3-D hand models are articulated deformable objects with many DOFs, a huge image database is required to deal with the entire characteristic shapes under several views. Another drawback is the difficulty of feature extraction and inability to handle singularities that occur from unclear views.

Appearance-based techniques extract image features to model the visual appearance of the hand and compare these features with the extracted features from the video frames as our approach. They have real-time performance because of the easier 2-D image features that are used. A simple method, searching for skin colored regions in the image, was used in [5]. However, this method has some shortcomings; first, it is very sensitive to lighting conditions. Secondly, it is required that no other skin-like objects exist in the image.

In [6], scale-space color features are used to recognize hand gestures, which are based on feature detection and user independence. However, the system shows real-time performance only when no other skin-colored objects exist in the image.

The authors of [7] obtained a clear-cut and integrated hand contour to recognize hand gestures and then computed the curvature of each point on the contour. Due to noise and unstable illumination in the cluttered background, the segmentation of integrated hand contour had some difficulty.

The eigenspace is another technique, which provides an efficient representation of a large set of high-dimensional points using a small set of basis vectors. However, eigenspace methods are not invariant to translation, scaling, and rotation.

TABLE I
PERFORMANCE OF ROBUST FEATURE DETECTION METHODS: SIFT, PCA-SIFT, AND SURF

Method	Time	Scale	Rotation	Illumination
SIFT	Good	Best	Best	Common
PCA-SIFT	Good	Common	Good	Good
SURF	Best	Good	Common	Best

There have been a number of research efforts recently on local invariant features [8]–[10]. In [8], Adaboost learning algorithm and SIFT features were used to achieve in-plane rotation invariant hand detection. In addition, a sharing feature concept was used to speed up the testing process and increase the recognition accuracy. Therefore, efficiency of 97.8% was achieved. However, several features such as a contrast context histogram had to be used to achieve hand gesture recognition in real time. In [9], [10], Haar-like features were applied for hand detection. Haar-like features concentrate more on the information within a certain area of the image rather than each single pixel. To enhance classification accuracy and attain real-time performance, the AdaBoost learning algorithm, which can adaptively choose the best features in each step and combine them into a strong classifier, can be used.

With the learning-based object detection technique proposed by Viola and Jones [11], the bare hand detection without any restriction on the background evolves dramatically [12]. The detection method attains robust detection, but needs a large training time for obtaining the cascaded classifier. In addition, hand detection with the Viola–Jones detector can be done with about 15° in-plane rotations compared to 30° on face [13]. Even though rotation invariant hand detection can be achieved using the same Adaboost framework in a way of treating the problem as a multiclass classification problem, the training process requires much more training images and more computational power for both training and testing.

It has been found that the keypoints provide an efficient image representation for tasks varying from image classification to object recognition. Keypoints are salient image patches that include rich local information of an image. Keypoints are detected by robust feature detection methods like SIFT [2], its variant principal component analysis (PCA)-SIFT [14] and speeded up robust features (SURF) [15]. In [2], SIFT was used for extracting distinctive invariant features from images that can be invariant to image scale and rotation. Then, it was widely applied in image mosaic, recognition, and retrieval. Then, in [14], PCA was used to normalize gradient patch instead of histograms. It turned out that PCA-SIFT-based local descriptors were also distinctive and robust to image deformations. In [15], robust features (SURF) were speeded up, and integral images were used for image convolutions and Fast-Hessian detector. The experiments showed that it was very fast and worked properly. In [19], many experiments were conducted to evaluate performance of: SIFT, PCA-SIFT, and SURF. Table I summarizes their results.

Table I shows that SIFT is fast for low resolution images and invariant to scale and rotation and partially to illumination changes. SURF is the fastest and has good performance as the same as SIFT, but it is not stable to rotation changes. PCA-SIFT has advantages in illumination changes, while not in scale changes.

Since the keypoints (features), which represent the detected hand gestures only, were extracted in real time using the SIFT algorithm and were invariant to scale and orientation, and the training images had been captured under different lighting conditions, our system has real-time performance and is robust against scale, rotation, illumination changes, and cluttered background.

Keypoint features can be used in their raw format for direct image matching [16], or vector-quantized keypoints features into a representation like the bag-of-words representation of text documents. There were a lot of researches using this vector-quantized, or bag-of-features representation, for image classification [16]–[18].

Recently, bag-of-features representations have shown outstanding performance for action and gesture recognition [42]–[44]. They permit to recognize a rich set of actions ranging from simple periodic motion (waving, running) to interactions (kissing, shaking hands) [45]–[48]. However, "bag-of-features" approaches exclusively rely on the dense local motion features. They do not have the relations between the features in the spatial and the temporal domains. These are significant correlation, which are useful for recognition. There were a lot of researches on expanding "bag-of-features" to include the spatial relation in the context of object categorization [17], [49]–[52]. In [53], the spatiotemporal pyramid was employed as a representation and at the same time integrated the spatiotemporal relation among visual features with their appearance information. In [54], a spatiotemporal interest point detector was proposed based on 1-D Gabor filters. In [55], a detector was proposed based on the determinant of the space-time Hessian matrix. In [56], local features were calculated from temporal self-similarity matrices of actions. In [57], a local space-time descriptor was proposed based on space-time gradients.

In our approach, we used spatiotemporal (space-time) correlation between every two consecutive frames in a video sequence in terms of the transition among recognized postures and their locations to develop our system into dynamic hand gesture recognition. The timing relation for transition among recognized postures is monitored for every two consecutive frames of a video sequence by saving every two consecutive recognized postures in two states in a queue, where the previous recognized posture is saved in the old state of the queue, while the current recognized posture is saved on the new state. The movement direction or space relation between every two postures recognized from every two consecutive frames is tracked by monitoring the difference between the two locations of the previous and current recognized hand postures. This part of our contributions will be discussed thoroughly in Section VI.

III. SYSTEM OVERVIEW

Our hand gesture recognition system consists of two stages: the offline training and the online testing, which is extended from our system in [25] by using face detection and subtraction

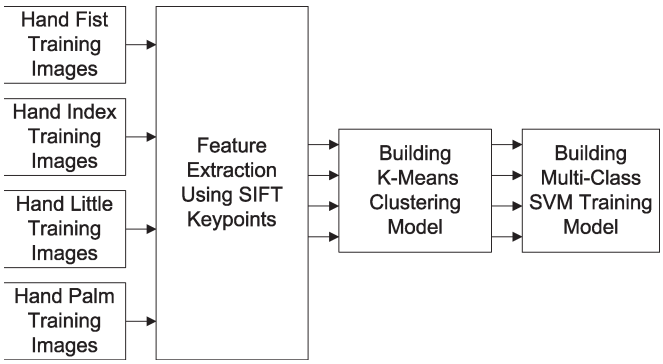


Fig. 2. Training stage.

and hand gesture detection. The cluster and multiclass SVM classifier models will be built in the training stage and will be used in the testing stage to recognize hand gestures captured from a webcam.

A. Training Stage

The training stage model is shown in Fig. 2. Before building the bag-of-features model, we captured 100 training images for each hand gesture, which are the fist, index, palm, and little finger gestures, for different people, scales, and rotations and under different illuminations conditions to increase the robustness of the multiclass SVM classifier and the cluster model. The system can also recognize any other gestures, such as two, three, and five. All the training images illustrate the hand gestures without any other objects, and the background has no texture or objects (white wall). In this way, we guarantee that all the keypoints extracted from training images using the SIFT algorithm will represent the hand gesture only. Note that we can reduce image processing time by reducing the number of keypoints. This is achieved by reducing the image resolution and converting training images to the *portable gray map* (PGM) format. The image processing time is not too important in the training stage as it is in the testing stage. For the case that we did not use hand detection algorithm, the size of training images have been reduced to 320×240 pixels and converted them into PGM format to coincide with the size of images captured from the video file in the testing stage. We repeated the training stage with another 160×120 pixels training images size to build another cluster and multi-class SVM classifier models to repeat the testing stage with the 160×120 pixels frame size. While for the case of using our hand detection algorithm, we built new cluster and multi-class SVM classifier models after reducing the size of training images to 50×50 pixels and converted them into PGM format to coincide with the size of the small image (50×50 pixels) that contains the detect hand gesture only for every frame captured from the video file in the testing stage.

The bag-of-features model is built using features extraction, learning a "visual vocabulary" by k-means clustering, quantizing features using visual vocabulary, and finally representing images by frequencies of "visual words," as will be discussed in the following:

1) Features Extraction Using Scale Invariant Feature Transform (SIFT): Features based on the SIFT algorithm are invariant to scale and rotation and can be extracted in real time for low resolution images. They are extracted in four stages. The first step computes the locations of potential interest points in the image by detecting the maxima and minima of a set of difference of Gaussian filters applied at different scales all over the image. Then, these locations are refined by getting rid of points of low contrast. An orientation is then assigned to each key point based on local image features. Finally, a local feature descriptor is computed at each keypoint. This descriptor is based on the local image gradient, transformed according to the orientation of the keypoint to provide orientation invariance. The size of the feature vector depends on the number of histograms and the number of bins in each histogram. In Lowe's original implementation [2] a 4-by-4 patch of histograms with 8 bins each is used, generating a 128-dimensional feature vector.

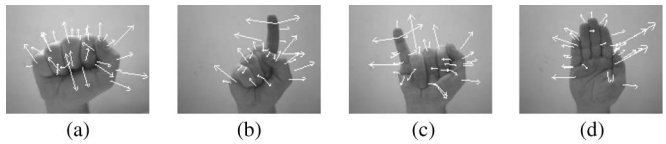


Fig. 3. 640×480 training images. (a) Fist with 35 features. (b) Index with 41 features. (c) Little finger with 38 features. (d) Palm with 75 features.

We used the SIFT algorithm to extract the keypoints (vectors) for each training image. Fig. 3 shows some training images with their keypoints. The number of keypoints decreases when the hand gets away from the camera and increases when the hand comes closer to the camera, because the area of the hand increases. For the same distance from the camera, we notice that the palm gesture has the maximum number of keypoints as it has the largest area. We can increase the number of training images to train the system as we wish for all the hand gestures for different people with different scales, orientations, and illumination conditions. The more training images used with different illumination conditions, the more accurate in building k-means cluster and SVM models since extracted features for training images using SIFT are invariant to scale, orientation, and partially to illumination changes. Therefore, the time will increase for building the cluster model in the training stage. However, this will not affect the testing stage speed.

2) K-Means Clustering: Clustering is the division of a set into subsets, called clusters, so that elements in the same cluster are similar in some sense. It is an approach of unsupervised learning algorithm and an ordinary method for statistical data analysis applied in several fields, such as machine learning, pattern recognition, image analysis, data mining, and bioinformatics.

Among many different types of clustering, in this paper, we used the k-means clustering algorithm [20]. The number of the clusters (codebook size) will be determined depending on the structure of the data. There will be a sort of compromise for how to choose vocabulary size or number of clusters. If it is too small, then each bag-of-words vector will not represent all the keypoints extracted from its related image. If it is too large, then there will be quantization artifacts and overfitting because of insufficient samples of the keypoints extracted from the training image.

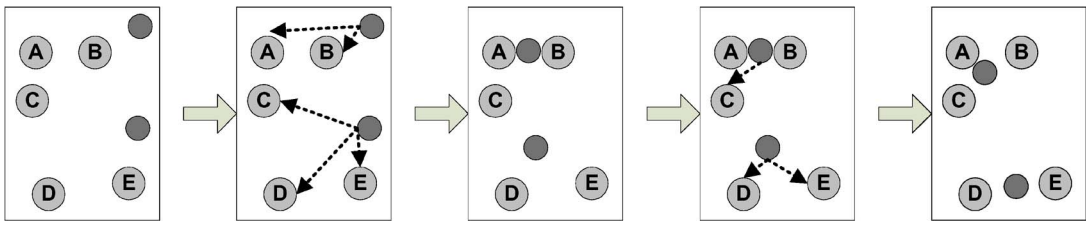


Fig. 4. K-means clustering with two clusters.

In the training stage, when the training images contain only hand gestures on a white background, the keypoints that are extracted will represent the hand gesture only, and this will not exceed 75 keypoints for the palm gesture, which has the largest number of keypoints. From this information, we know that the number of clusters must be larger than 75. Therefore, the training stage provides the minimum number of clusters that we can use. In the testing stage, the webcam will capture other objects besides the hand gesture such as the face and background. The keypoints will be around 400 keypoints of all the objects in the image. We chose the value 750 as the number of clusters (visual vocabularies or codebook) to build our cluster model.

The first step in k-means clustering is to divide the vector space (128-dimensional feature vector) into k clusters. K-means clustering starts with k randomly located centroids (points in space that represent the center of the cluster) and assigns every keypoint to the nearest one. After the assignment, the centroids (codevectors) are shifted to the average location of all the keypoints assigned to them, and the assignments are redone. This procedure repeats until the assignments stop changing. Fig. 4 shows this process in action for five keypoints: A, B, C, D, and E and two clusters.

Once this is done, each feature vector (keypoint) is assigned to one and only one cluster center that is in the nearest distance with respect to the Euclidean metric in 128-dimensional feature vectors. The keypoints that are assigned to the same cluster center will be in the same subgroup so that after clustering, we have k disjoint subgroups of keypoints. Therefore, k-means clustering decreases dimensionality for every training image with n keypoints $(n \times 128)$ to $1 \times k$, where k is number of clusters.

The keypoint vectors for each training image will be used to build the cluster model using k-means clustering. The number of clusters (codebook) will represent the number of centroid in the cluster model. Finally, the cluster model will build codevectors equal to the number of clusters assigned (k) and each codevector will have 128 components, which is equal to the length of each keypoint. Then, the keypoints of each training image will be fed into the k-means clustering model to reduce its dimensionality into one bag-of-words vector with components equal to the number of clusters (k). In this way, each keypoint, extracted from a training image, will be represented by one component in the generated bag-of-words vector with value equal to the index of the centroid in the cluster model with the nearest Euclidean distance. The generated bag-of-words vector, which represents the training image, will be grouped with all the generated vectors of other training images that have the same hand gesture and labeled with the same number, and this label will represent the class number. For example, label or class 1 for the fist training images, class 2 for index training images, class 3 for little training images, and class 4 for palm training images.

3) Building the Training Classifier Using Multiclass SVM: After mapping all the keypoints that represent every training image with its generated bag-of-words vector using k-means clustering, we fed every bag-of-words vector with its related class or label number into a multiclass SVM classifier to build the multiclass SVM training classifier model.

SVM is a group of related supervised learning methods used for classification and regression. It carries out classification by creating an N-dimensional hyperplane that optimally divides the data into two groups. SVM classifiers are closely related to neural networks. Actually, a SVM classifier model using a sigmoid kernel function is the same as the two-layer, perceptron neural network.

In the SVM literature, a predictor variable is known as an attribute, and a transformed attribute that is used to define the hyperplane is known as a feature. The operation of selecting the most appropriate representation is called as feature selection. A group of features that describes one case is known as a vector. Therefore, the objective of SVM modeling is to find the optimal hyperplane that divides clusters of vectors in such a way that cases with one class of the target variable are on one side of the plane and cases with the other classes are on the other side of the plane. The vectors near the hyperplane are the support vectors.

Even though SVMs were initially intended as binary classifiers, other methods that deal with a multiclass problem as a single "all-together" optimization problem exist [21], but are computationally much more costly than solving several binary problems.

A variety of approaches for decomposition of the multiclass problem into several binary problems using SVMs as binary classifiers have been proposed. In our implementation, multiclass SVM training and testing are performed using the library for SVM described in [22]. This library supports multiclass classification and uses a *one-against-one* (OAO) approach for multiclass classification in SVM [23]. For the M-class problems (M being greater than 2), the OAO approach creates M(M-1)/2 two-class classifiers, using all the binary pair-wise combinations of the M classes. Each classifier is trained using the samples of the first class as positive examples and the samples of the second class as negative examples. To combine these classifiers, the Max Wins method is used to find the resultant class by selecting the class voted by the majority of the classifiers [24].

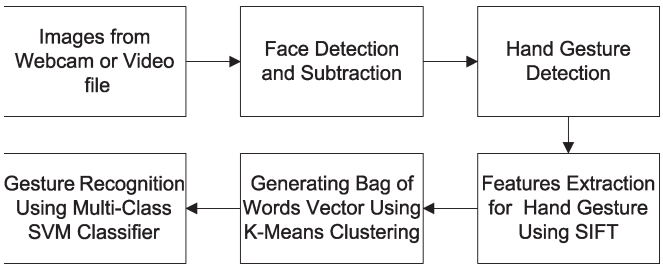


Fig. 5. Testing stage.

B. Testing Stage

Fig. 5 shows the testing stage, which is extended from our previous work [25] by using face detection and subtraction and hand gesture detection. After capturing frames from webcam or video file, we detected the face and subtracted it before using a skin detection and hand posture contour comparison algorithm because the skin detection will detect the face and the face's contours very close to the fist hand gesture contours. To get rid of other skin-like objects existing in the image, we make sure that the contours of the detected skin area comply with the contours of any hand gestures contours to detect and save the hand gesture only in a small image (50 $\times$ 50 pixels). Then, the keypoints were extracted from the small image that contains the hand gesture only and will be fed into the cluster model to map them into a "bag-of-words" vector and finally this vector will be fed into multiclass SVM training classifier model to recognize the hand gesture.

1) Face Subtraction: We used the skin detection and contour comparison algorithm to detect the hand gesture, and this algorithm can also detect the face because the face has a skin color and its contours like the hand fist gesture contours. To get rid of face area, we detected the face using Viola and Jones method [11] and then subtracted the face before applying the skin detection algorithm to detect the hand gesture only by replacing face area with a black circle for every frame captured. The Viola and Jones algorithm has a real-time performance, achieving accuracy as the best published results [11].

First, we loaded a statistical model, which is the XML file classifier for frontal faces provided by OpenCV to detect the faces from the frames captured from a webcam during the testing stage. The XML file classifier, which is a cascade of classifiers working with Haar-like features, was trained with a few hundreds of sample views of human face, called positive examples that were scaled to the same size such as 30 $\times$ 30 pixels and negative examples, which are random images of the same size. The term "cascade" in the classifier indicates that the resultant classifier includes several simpler classifiers that are applied subsequently to a region of interest until at some stage, the candidate is discarded for image regions least likely to be a face or all stages are accepted for those that represent a face as shown in Fig. 6. These classifiers use simple rectangular features, called Haar-like features as in Fig. 7.

The Haar feature used in a particular classifier is determined by its shape, position within the region of interest, and the scale. The existence of a Haar feature is found by subtracting the average dark-region pixel value from the average light-region pixel value. If the difference is above a threshold, which is set during learning, that feature will be existed. The Haar-like features make use of the following information as distinctive features that can be used to detect the face as shown in Fig. 8:

- Eyes are dark (eyebrows and shadows).
- Cheeks and forehead are bright.
- Nose is bright.

After a classifier is trained, a set of features are extracted and distinctive features that can be used to classify the face are selected. The classifier can be applied to a region of interest in an input image. The classifier outputs a "1" if the region is probable to show the face and "0" otherwise. To search for the face in the entire image, the search window can be moved across the image and check every pixel using the classifier. The classifier can be simply "resized" to be able to find the objects of interest at various scales, which is more efficient than resizing the image itself. Therefore, to locate an object of an unknown size in the image, the search process should be repeated several times at different sizes.

Once the face had been detected by the XML file classifier for every frame captured, we replaced the detected face area with a black circle to remove the face from the skin area. In this way, we make sure that the skin detection will be for hand gesture only as shown in Fig. 9.

2) Hand Gesture Detection: Detecting and tracking human hand in a cluttered background will enhance the performance of hand gesture recognition using bag-of-features model in terms of accuracy and speed because the keypoints extracted will

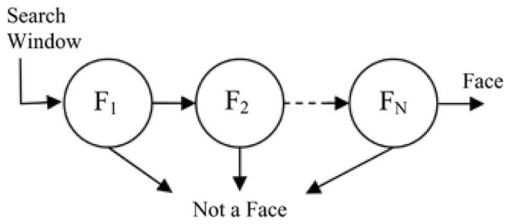


Fig. 6. Cascade of classifiers.

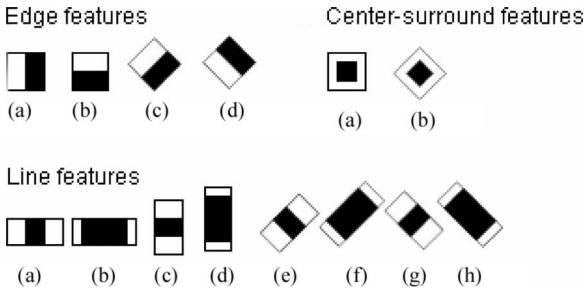


Fig. 7. Set of Haar-like features.

Fig. 8. Detecting a human face using Haar-like features.

Fig. 9. Face detection and subtraction.

Fig. 10. Templates of hand gestures.

represent the hand gesture only. Moreover, we will not be confined with the frame resolution size captured from a webcam or video file, because we will always extract the keypoints of the small image (50 × 50 pixels) that contains the detected hand gesture area only, not the complete frame. In this way, the speed and accuracy of recognition will be the same for any frame size captured from a webcam such as 640 × 480, 320 × 240, or 160 × 120, and the system will be also robust against cluttered background because we process the detected hand gesture area only. The small image size (50 × 50 pixels) that contains the detected hand gesture area only has to be complied with the training images size of training stage.

For detecting hand gesture using skin detection, there are different methods including skin color-based methods. In our case, after detecting and subtracting the face, skin detection and contour comparison algorithm was used to search for the human hands and discard other skin-colored objects for every frame captured from a webcam or video file. Before capturing the frames from a webcam, we loaded four templates of hand gestures as shown in Fig. 10, fist, index, little, and palm to extract their contours, and saved them for comparison with the contours of skin detected area of every frame captured. After detecting skin area for every frame captured, we used contour comparison of that area with the loaded hand gestures contours to get rid of other skin-like objects that exist in the image. If the contour comparison of skin detected area complies with any one of the stored hand gesture contours, a small image (50 × 50 pixels) will enclose the hand gesture area only, and that small image will be used for extracting the keypoints using SIFT algorithm.

Skin detection is an important issue in many Computer Vision applications such as face detection, recognition, facial expression extraction, face and hand tracking, hand gesture recognition, virtual reality, and other many recognition systems.

There have been several studies and different approaches proposed on skin color modeling and recognition. The objective of skin color detection is to make a decision rule that will differentiate between skin and nonskin pixels based on color components. In order to get appropriate distinction between skin and nonskin regions, color transformation to separate luminance from chrominance is needed.

Although the input image is generally in RGB format, skin detection methods usually use color components in the color spaces, such as HSV, YCbCr, TSL, or YIQ. That is because RGB components are dependent on the lighting conditions. The overlap between skin and nonskin pixels can be decreased using color space transformation. That will help in skin-pixel classification and provide robust parameters against lighting conditions changes. It has been noticed that skin colors vary more in intensity than in chrominance [27]. Therefore, the luminance component can be dropped for skin classification. HSV is one of the color spaces most commonly used for skin detection. According to Zarit *et al.* [28], HSV provides the best performance for skin-pixel detection.

In our implementation we used the *HSV* color model since it has shown to be one of the most adapted to skin-color detection [28]. It is also compatible with the human color perception. Furthermore, it has real-time performance and robust against rotations, scaling, and lighting conditions and can tolerate occlusion well.

The HSV color space is obtained by a nonlinear transformation of the fundamental RGB color space. The transformation between RGB and HSV is described in detail in [29]. *Hue* (H) is a component that represents pure color such as pure yellow, orange, or red, whereas *saturation* (S) provides a measure of the degree to which a pure color diluted by white light [30]. *Value* (V) attempts to represent brightness along the grey axis such as white to black, but as brightness is subjective, it is thus difficult to measure [30].

Experiments have shown that skin colors of individuals cluster closely in the color space for all people from different ethnicities, i.e., color appearances in human faces and hands vary more in intensity than in chrominance [31], [32]. Thus, removing the intensity V of the original color space and working in the chromatic color space (H,S) provide invariance against illumination conditions. In [33], it had been noted that discarding the Value (V) component and only using the Hue and Saturation components, can still permit for the detection 96.83% of the skin pixels.

From a classification approach, skin-color detection can be considered as a two class problem: skin-pixel versus nonskin-pixel classification. There are many classification techniques such as thresholding, Gaussian classifier, and multilayer perceptron [34]. In our implementation, we used the thresholding method, which has the least time on computation compared with other techniques, and this is required for real-time application. The basis of thresholding classification is to find the range of two components H and S in HSV model as we discarded the Value (V) component. Usually, a pixel can be viewed as being a skin pixel when the following threshold ranges are simultaneously satisfied: $0° < \mathrm{H} < 20°$ and $75 < \mathrm{S} < 190$.

Once the skin area had been detected, we found contours of the detected area and then compared them with the contours of the hand gestures templates. If the contours of the skin area comply with any of the contours of the hand gestures templates, then, that area will be the region of interest by enclosing the detected hand gesture with a rectangle, which will be used in tracking the hand movements and saving hand gesture in a small image (50 × 50 pixels) for every frame captured as shown in Fig. 11. The small image will be used in extracting the keypoints to recognize hand gesture.

3) Hand Gesture Recognition: We converted the small image (50 × 50 pixels) that contains the detected hand gesture

Fig. 11. Small images of detected hand gestures.

only for every frame captured into a PGM format to reduce the time needed in extracting the keypoints. For every small PGM image, we extracted the keypoints (vectors) using the SIFT algorithm. The keypoints will be fed into the k-means clustering model that was built in the training stage to map all the keypoints with one generated vector (bag-of-words) with components (k), which is equal to the number of clusters (k) used in the training stage. Each feature vector in the keypoints will be represented by one component in the generated vector with value equals to the index of centroid in the cluster model with nearest Euclidean distance. Finally, the generated bag-of-words vector will be fed into the multiclass SVM training classifier model that was built in the training stage to classify and recognize the hand gesture.

IV. Experimental Results

In this section, we will discuss results of our experiments conducted on our image data set and on public image data set.

A. Results on Our Image Data Set

We tested four hand gestures: the fist gesture, the index gesture, the little finger gesture, and the palm gesture. The camera used for recording video files in our experiment was a low-cost Logitech QuickCam webcamera that provides video capture with different resolutions such as 640 × 480, 320 × 240, and 160 × 120, at 15 frames-per second, which is adequate for real-time speed image recognition.

We conducted the experiment to recognize hand gestures for two cases. First case was without face subtraction and hand gesture detection as done in our work [25], and the other case was with hand gesture detection. There are two restrictions for the first case [25]. First, since the SIFT algorithm is real time for low resolution images, the resolution of video frames in the testing stage must not exceed 320 × 240 pixels. While the other restriction in the testing stage has to be done with a white wall as a background because the keypoints extracted in the testing stage will increase dramatically with cluttered background, and this will make it difficult to recognize the keypoints of hand gestures from the other keypoints of cluttered background.

While for the second case, there is no restriction in the testing stage to run it against the resolution of video frames and cluttered background because the region of interest will be determined by face subtraction and hand gesture detection, and that region, which contains the hand gesture only, will be saved in a small image used for extracting the keypoints of hand gestures only for every frame captured.

1) Hand Gesture Recognition Without Face Subtraction and Hand Gesture Detection: Ten video files with the resolution of 320 × 240 had been recorded for each hand gesture: fist, index, little, and palm using the Logitech webcam. The length of each video file was 100 images. The hand gestures were recorded with different scales and rotations and without any object in the background. The 40 video files were recorded under different illuminations conditions. Fifty images of each video file were recorded with natural fluorescent lighting condition, while for the others, it had been installed an extra incandescent light bulb to create a tungsten lighting condition. The test had been run for the 40 video files to evaluate the performance of the multiclass SVM classifier model for each gesture.

TABLE II
Performance of the Multiclass SVM Classifier Without Other Objects (320 × 240 Pixels)

Gesture Name	Number of frames	Correct	Incorrect	Recognition Time (Second/frame)
Fist	1000	972	28	0.06330
Index	1000	989	11	0.06332
Little	1000	963	37	0.06331
Palm	1000	995	5	0.06333

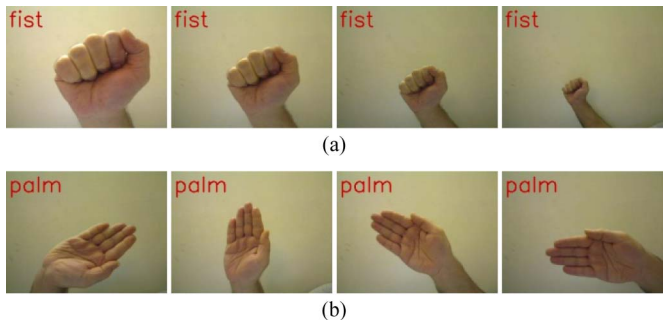


Fig. 12. Hand gesture recognition without any object against. (a) Scale. (b) Rotation.

Table II shows the performance of the multiclass SVM classifier for each gesture with testing against scale, rotation, and illumination. We repeated the experiment with other ten video files for each hand gesture with the resolution of 160 × 120. Only about 0.0492 s was required to recognize hand gesture for every frame with the same accuracy as 320 × 240 pixels videos results. During the real-time testing with live input from each video file, there was no detectable pause and latency to recognize the hand gestures. It appears that the system shows excellent results in terms of accuracy and speed because the features of hand gestures were extracted in real time for using the SIFT algorithm and were invariant to scale and orientation. In addition, since the training images had been captured under different lighting conditions, the multiclass SVM classifier is robust against illumination changes. The system wrote the gesture name for each frame captured in the above left of the image as shown in Fig. 12.

Another ten video files with the resolution of 320 × 240 had been recorded for each hand gesture: fist, index, little finger, and palm using the Logitech webcam. The length of each video file was 100 images. The hand gestures were recorded with different scales, rotations, and illuminations conditions and

TABLE III
PERFORMANCE OF THE MULTICLASS SVM CLASSIFIER WITH OTHER OBJECTS (320 × 240 PIXELS)

Gesture Name	Number of frames	Correct	Incorrect	Recognition Time (Second/frame)
Fist	1000	951	49	0.06994
Index	1000	979	21	0.06991
Little	1000	935	65	0.06992
Palm	1000	991	9	0.06997

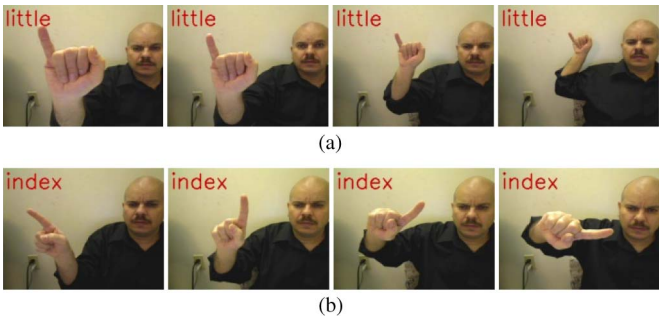


Fig. 13. Hand gesture recognition with other objects against. (a) Scale. (b) Rotation.

with other objects in the background such as the face. The test was run for the 40 video files to evaluate the performance of the multiclass SVM classifier model for each gesture. The trained multiclass SVM classifier shows a certain degree of robustness against scale, rotation, and illumination. Table III shows the performance of the multiclass SVM classifier for each gesture with testing against scale, rotation, and illumination. The recognition time increased because the extracted keypoints increased from other objects. We repeated the experiment with other ten video files for each hand gesture with the resolution of 160 × 120. Only about 0.0512 s was required to recognize hand gesture for every frame with the same accuracy as 320 × 240 videos results. Fig. 13 shows some correct samples for hand gesture recognition using multiclass SVM classifier for little and index gestures with testing against scale, rotation, and illumination.

2) Hand Gesture Recognition With Face Subtraction and Hand Gesture Detection: Ten video files with the resolution of 640 × 480 had been recorded for each hand gesture: fist, index, little finger, and palm using a commercial grade webcam. The length of each video file was 100 images. The hand gestures were recorded with different scales, rotations, and illuminations conditions and with a cluttered background. The test was run for the 40 video files to evaluate the performance of the multiclass SVM classifier model for each gesture. The trained multiclass SVM classifier shows a certain degree of robustness against scale, rotation, illumination, and cluttered background. Table IV shows the performance of the multiclass SVM classifier for each gesture with testing against scale, rotation, illumination, and cluttered background. The recognition time did not increase with cluttered background or increasing the resolution of video file because the keypoints will be extracted from the small image (50 × 50 pixels) that contains the detected hand gesture only. We repeated the experiment with other ten video files for each hand gesture with the resolution of 320 × 240 pixels; the same time was needed to recognize every frame with the same accuracy as 640 × 480 videos results because the keypoints were extracted in both cases from the small image that contains the hand gesture only. Fig. 14 shows some correct samples for hand gesture recognition using multiclass SVM classifier for the four hand postures with testing against scale, rotation, illumination, and cluttered background.

TABLE IV
PERFORMANCE OF THE MULTICLASS SVM CLASSIFIER WITH CLUTTERED BACKGROUND (640 × 480 PIXELS)

Gesture Name	Number of frames	Correct	Incorrect	Recognition Time (Second/frame)
Fist	1000	967	33	0.017
Index	1000	990	10	0.017
Little	1000	956	44	0.017
Palm	1000	994	6	0.017

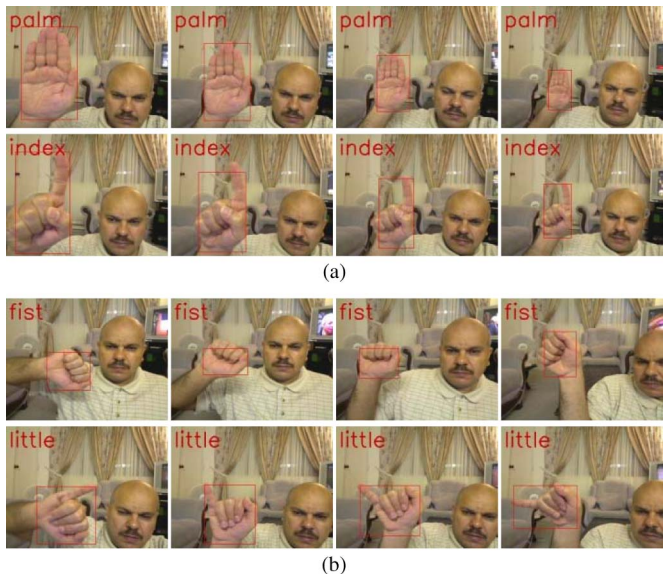


Fig. 14. Hand gesture detection and recognition with cluttered background against. (a) Scale. (b) Rotation.

From Table IV, we notice that the recognition time for every frame had been reduced dramatically to 0.017 s for any video resolution size used because the time spent for extracting the keypoints was only for the small image (50 × 50 pixels) that contains the detected hand gesture. Moreover, the recognition accuracy increased because the keypoints extracted represent the hand gesture only. While for the case without hand gesture detection as in [25], it needs more time because the keypoints were extracted for the whole objects in the image.

We repeated the same experiment for the case face subtraction and hand gesture detection with different resolution sizes for the small image that captures the detected hand gesture only. Table V shows the recognition time needed for every small image size, which contains the hand gesture only.

TABLE V
RECOGNITION TIME NEEDED WITH CLUTTERED BACKGROUND FOR DIFFERENT SMALL IMAGES SIZES USED TO CAPTURE HAND GESTURE ONLY

Size of small images (Pixels)	Recognition Time (Second/frame) (640×480 Pixels)
50×50	0.017
80×80	0.017
90×90	0.034
130×130	0.034
140×140	0.049
170×170	0.049
180×180	0.063
210×210	0.063

Fig. 15. Added hand postures to our system.

We notice from Table V that if we increase the size of the small image that captures the detected hand gesture only from 50 × 50 pixels to 80 × 80 pixels, the recognition time is the same.

To make sure of the robustness of our system, we added three new hand postures defined in ASL (American Sign Language) to our system, which are l (L), W (three), and Y as shown in Fig. 15.

In the training stage, we captured 100 training images with a size of 50 × 50 for every new hand posture and added them to the four previous hand postures training images. The 700 training images for seven hand postures were used to build a new cluster and multiclass SVM classifier models that can classify seven hand postures, which are fist, index, little, palm, l, W and Y. For every new hand posture, we recorded ten video files of 100 frames length for every video file with different scales, rotations, and illuminations conditions and with a cluttered background to test them with the previous 40 video files of fist, index, little, and palm postures. The seven hand postures contours were loaded to detect hand posture in every frame of 70 video files after face subtraction and skin detection. Then, the test was run for the 70 video files to evaluate the performance of the new multiclass SVM classifier model for each hand posture. The multiclass SVM classifier gave excellent recognition results as shown in Table VI. Fig. 16 shows some correct samples for the three new hand postures: l (L), W (three), and Y.

B. Results on Public Image Data Set

The public database images used for testing in this paper is the Sebastien Marcel database [58], which is a benchmark

TABLE VI
PERFORMANCE OF THE MULTICLASS SVM CLASSIFIER WITH CLUTTERED BACKGROUND FOR SEVEN POSTURES (640 × 480 PIXELS)

Gesture Name	Number of frames	Correct	Incorrect	Recognition Time (Second/frame)
Fist	1000	964	36	0.017
Index	1000	988	12	0.017
Little	1000	951	49	0.017
Palm	1000	993	7	0.017
l (L)	1000	958	42	0.017
W(Three)	1000	975	25	0.017
Y	1000	962	38	0.017

Fig. 16. Hand gesture detection and recognition for l (L), W (three), and Y postures.

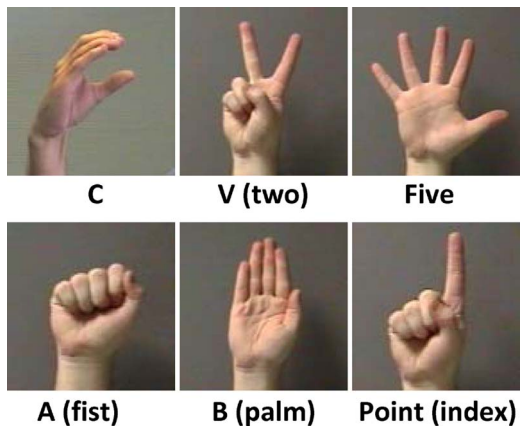


Fig. 17. Hand postures used in Marcel database images.

database in the field of hand gesture recognition. This database contains (100 × 100 pixels) color images of six hand postures in ASL, which are C, A (fist), Five, Point (index), B (palm), and V (two) as shown in Fig. 17, performed by different people against uniform and complex backgrounds. Therefore, we have to add three new postures to our system, which are C, Five, and V (two) postures.

In the training stage, we captured 100 training images with a size of 50 × 50 for every new hand posture, which are C, Five,

TABLE VII
PERFORMANCE OF THE MULTICLASS SVM CLASSIFIER WITH PUBLIC IMAGE DATA SET (100 × 100 PIXELS)

Posture Name	Number of Images	Correct	Incorrect	Recognition Accuracy	Recognition Time (Second/frame)
Fist	97	92	5	94.85%	0.017
Palm	102	100	2	98.04%	0.017
C	112	108	4	96.43%	0.017
V(Two)	95	91	4	95.79%	0.017
Five	134	127	7	94.78%	0.017
Index	119	116	3	97.48%	0.017

and V (two), and added them to the three previous hand postures training images to build a new cluster and multi-class SVM classifier models that can classify six postures. The 600 training images for six postures contain only hand posture on a white background, therefore the extracted keypoints will represent the hand posture only.

In the testing stage, we used the new cluster and multiclass SVM classifier models that were built in the training stage to classify the six hand postures provided by Sebastien Marcel database. The six hand posture contours were loaded to detect hand postures in test images after face subtraction and skin detection. The multiclass SVM classifier gave excellent recognition results on this benchmark database as shown in Table VII. The overall recognition accuracy is 96.23%. As we mentioned before, the recognition time will be always 0.017 s for each frame regardless of the frame size used because the keypoints will be extracted from the small image (50 × 50 pixels) that contains the detected hand gesture only.

Fig. 18 shows some correct samples for the six hand postures provided by Sebastien Marcel database using our approach. In the next section, a performance comparison of our approach with Sebastien Marcel [58] approach and other approaches will be discussed.

Fig. 18. Hand posture detection and recognition for the six hand postures of Marcel database images using our approach.

V. COMPARISON WITH OTHER APPROACHES

We selected some papers [10], [58]–[62] that had a real-time performance and discussed their recognition time and accuracy to compare their performance with our approach. In [59], the hand gesture was detected using skin color approach. Features for all the detected hand gestures used were extracted based on Haar Wavelet Representation and stored in a database. During the recognition process, a measurement metric was utilized to measure the similarity between the features of a test image and those in the database. In [60], hand detection with Adaboost was used to trigger tracking and recognition. Then, adaptive hand segmentation was executed during detection and tracking with motion and color cues. Finally, scale-space feature detection was applied to find palm-like and finger-like structures. Hand gesture type was determined by palm–finger configuration. In [61], first hand gesture was detected based on Viola–Jones method. Then, the Hu invariant moment feature vectors of the detected hand gesture were extracted, and a SVM classifier was trained for final recognition. In [10], the posture was detected using Haar-like features and the AdaBoost classifier. Based on the cascade classifiers, a parallel cascade structure was implemented to classify different hand postures.

TABLE VIII
PERFORMANCE COMPARISON WITH OTHER HAND POSTURE RECOGNITION APPROCHES

Paper	Number of Postures	Recognition Time (Second/Frame)	Recognition Accuracy	Frame Resolution	Number of Test Images/Posture	Scale	Rotation	Lighting Changes	Background
[59]	15	0.4	94.89%	160×120	30	Not discussed	Invariant	Not discussed	Wall
[60]	6	0.09 – 0.11	93.8%	320x240	195-221	Not discussed	Not discussed	Not discussed	Cluttered
[61]	3	0.1333	96.2%	640x480	130	Invariant	±30°	Variant	Different
[10]	4	0.03	90.0%	320x240	100	Invariant	±15°	Invariant	White wall
[62]	8	0.066667	96.9%.	640x480	300	Variant	Invariant	Not discussed	Not discussed
[58]	6	Not discussed	93.4 %	100×100	57-76	Invariant	Not discussed	Invariant	Wall
[58]	6	Not discussed	76.1 %	100×100	38-58	Invariant	Not discussed	Not discussed	Cluttered
Our method	10	0.017	96.23%	640x480 or any size	1000	Invariant	Invariant	Invariant	Cluttered

In [62], it acquired and preprocessed video image frame from camera, then extracted the normalized moment of inertia features and Hu invariant moments of gestures to constitute feature vector, which is inputted into SVM to achieved classification results. In [58], a neural network model was used to recognize a hand posture in an image. A space discretization based on face location and body anthropometry was used to segment hand postures.

A. Performance Comparison With Other Approches Based on Their Own Image Data Set

Table VIII shows the performance for all other approaches mentioned above based on their own image data set besides our approach in terms of recognition time and accuracy, resolution, number of tested images, and background and if it is invariant against scale, rotation, and different lighting conditions.

B. Performance Comparison With Other Approches Based on the Same Public Image Data Set

In [10], [60], [61], Viola and Jones's method [11] was used to detect hand posture before recognition. Therefore, we created six Haar classifiers to test detection for six postures in Sebastien Marcel database images [58] before recognition. We used OpenCV to train and create six classifiers for C, A (fist), Five, Point (index), B (palm), and V (two) postures using Haar-like features in Haar Training. Each classifier is trained with 1000 positive images for each hand posture and 3500 negative images. The result of training is six xml classifiers files. We loaded the trained xml classifiers simultaneously to detect the hand postures in the Sebastien Marcel database images. Around 40% of those database images have complex or cluttered background while the others have wall background with different colors. The detection results were not so successful. The classifiers did not detect six postures for all the test images that have cluttered background. While the classifiers detected six postures successfully for around 50% of the images that have wall background with some difficulty to detect the palm posture. The classifiers failed for other 50% because the rotations for hand postures are more than $\pm 15^{\circ}$. Therefore, we left with around 30% of the total images for recognition after detection, which will be the maximum recognition rate for algorithms used in [10], [60], [61]. To validate the performance of the six Haar classifiers, we tested them to detect our hand postures captured from a webcam. The 6 Haar classifiers detected our six hand postures successfully if the hand posture's rotation is less than $\pm 15^{\circ}$ and if the background is wall like the background used in Figs. 12 and 13. The palm classifier had some difficulty to detect the palm posture. Also, the six Haar classifiers failed to detect all the postures if the background is cluttered like the background used in Figs. 14 and 16.

Haar-like features have been used successfully in face classification [11]; however, hand detection and tracking in [13], [63], [64] have not been so successful. The major reason for this is Haar-like features are not invariant against rotation [65]. The Haar features that describe faces are insensitive to small angle changes as much as 30° from the vertical [13]. The person's head is normally aligned vertically with respect to gravity. Thus, rotational sensitivity is no major problem for faces. However, hands are not naturally aligned with the horizontal or vertical axes. Therefore, it is difficult to model them with traditional Haar-like features. As we discussed in Section III-B1, the face has strong Haar-like features related to shading, which facilitates the face detection in cluttered background. However, Haar-like features are not sufficient to describe the hand postures because the most significant features of a hand posture are the shape of the hand rather than internal hand shades and textures [66], which makes the hand posture detection difficult in cluttered background. In [10], a white wall background was used to detect hand postures using Haar classifiers.

Fig. 19. Skin detection for public images with cluttered background.

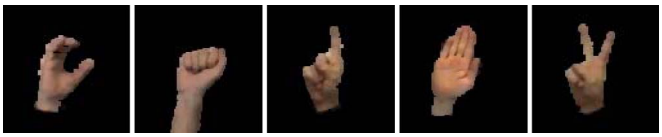


Fig. 20. Skin detection for postures alone for public images with wall and cluttered background.

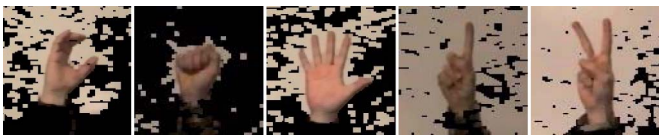


Fig. 21. Skin detection for public images with wall background.

In [59], [62], skin color detection method was used to detect hand postures before recognition. There are many constrains for their method to detect hand postures only using skin color detection such as no face and skin-like objects exist in the background and the front arm of the user has to be covered by clothes [59]. The figures in [59], [62] captured only the hand postures without any skin-like objects or the face. The background used in [59] is the wall only without any other objects. In [62], the algorithm for skin detection was not discussed at all. Therefore, we used algorithm used in [59] and our algorithm to detect skin color only in the Sebastien Marcel database images [58] before recognition without face subtraction and hand contours comparison algorithm because our algorithm was not used in [59], [62]. We found that the performance of our algorithm to detect skin color only is the same as the performance of algorithm used in [59]. The skin detection algorithm detected six postures with other skin-like objects for around 70% of the public test images in [58] that have cluttered background as shown in Fig. 19 for some samples. While the skin detection algorithm detected six postures alone without other objects for around 50% of the images that have wall background. Fig. 20 shows some samples for hand posture detection alone that have wall and cluttered background. Fig. 21 shows some samples where the skin detection did not detect hand posture alone for the public test images that have wall background. Therefore, we left with around 42% of the total images for recognition after detection, which will be the maximum recognition rate for algorithms used in [59], [62].

VI. Building Gesture Commands (Grammar)

A hand gesture is an action, which consist of a sequence of hand postures. The rules for the composition of hand postures into various hand gestures can be determined by a grammar [38]. Since we extended the testing stage that was used in our previous work [25] by using face detection and subtraction and hand gesture detection, we built a grammar for our system that generates 160 gesture commands, which can be used to control or interact with an application or a videogame instead of keyboard or mouse, by sending events to be executed such as double click, close, open, go left, right, up, or down, and so on. Those gesture commands can be generated by three ways. First, by observing the gesture transitions from posture to posture, such as from fist to index, fist to palm, fist to little, fist to l (L), fist to W (three), fist to Y, fist to C, fist to five, fist to two or stay fist, and so on for the other postures. For each posture, we have ten transitions to other states; thus, we have 100 events, or gesture commands can be sent from all of the transitions among of the ten postures. Second, by observing the direction of movement for each posture: up, down, left, or right. We have four directions or gesture commands for each posture or 40 gesture commands for ten postures. We did not take the case of no movement for each posture as it is counted already in the first way when the transition occurred from fist to fist or palm to palm and so on. Finally, by observing the hand posture size or scale: when it comes close (zoom in) or far away (zoom out) from camera. As we have two cases for each posture, we have 20 gesture commands for the ten postures. Therefore, from the three ways, we have totally 160 gesture commands that can be sent to interact with an application or video game. In the following sections, we will explain how our system can generate gesture commands using three ways.

A. Transitions Among Postures

This way depends on saving every two postures recognized from every two consecutive frames of a video sequence in a two states of a queue: the new state of the queue, which holds the current or new posture recognized from a video or webcam, and the old state of the queue, which holds the previous or old recognized posture. The first posture recognized from the first frame will be saved in the two states of the queue because they are empty. The next recognized posture will be saved in the new state after transferring its posture state to the old state. Thus, for every frame captured, the posture state of the old state is emptied. Then, the posture state of the new state is transferred to the old state. Finally, the current posture recognized from the frame will be saved in the new state. By observing the two states for every two consecutive frames captured, the system will keep monitoring the transitions among every two consecutive recognized postures and generates a specific gesture command for a specific transition among recognized postures. Fig. 22 shows all the transition states of fist posture with all other postures.

B. Movement Direction for Each Posture

This way depends on tracking the movement direction of the detected posture using the rectangle, which captures the detected hand posture only, and the transition of recognized posture still the same such as palm to palm. Once the hand posture is detected from each frame by the rectangle and the transition of recognized posture still the same, the coordinates X and Y point, which are located in the middle of the rectangle, are recorded. The system will always monitor the absolute difference of distance between the two points of rectangle in the X and Y coordinates for every two successive frames that have the same posture. If the absolute difference of distance in

Fig. 22. Transitions among postures.

the X direction is larger than absolute difference of distance in the Y direction, then the hand posture is moved left or right. If the difference of distance in the X direction is positive, then the hand posture is moved right and if it is negative, then the hand posture is moved left. If the absolute difference of distance in the Y direction is larger than the absolute difference of distance in the X direction, then the hand posture is moved up or down. If the difference of distance in the Y direction is positive, then the hand posture is moved down, and if it is negative, then the hand posture is moved up. Fig. 23 shows all the movement direction cases of palm posture.

C. Distance From Camera for Each Posture

This way depends on tracking the size of height for the rectangle, which captures the detected hand posture only, and the transition of recognized posture still the same. Once the hand posture is detected from each frame by the rectangle and the transition of recognized posture still the same such as little to little, the height of the rectangle is recorded. The system

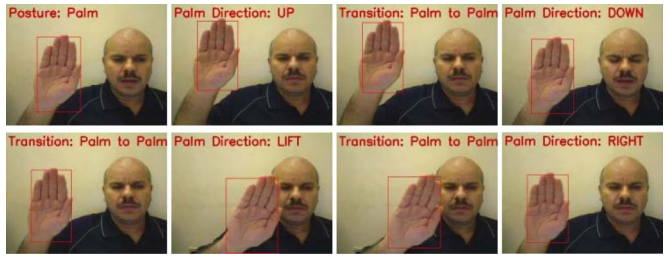


Fig. 23. Movement direction cases of palm posture.

Fig. 24. Zoom cases of little posture.

will always monitor the difference between the heights of two rectangles for every two successive frames that have the same posture. If the difference of rectangle height between the new frame and the previous frame is positive, then the posture gets closer to camera (zoom in), and if the difference is negative, then the posture gets away from camera (zoom out). Fig. 24 shows all the zoom cases of little posture.

VII. Conclusion

In this paper, we have described a real-time system that consists of three modules: hand detection and tracking using face subtraction, skin detection and contour comparison algorithm, posture recognition using bag-of-features and multiclass SVM, and a grammar that generates a large number of gesture commands by monitoring the scale of the detected hand posture, its movement direction, and the transitions among postures. In the training stage, after extracting the keypoints for all the training images using SIFT algorithm, we perform VQ on the keypoints for every training image using a k-means clustering to map them into a unified dimensional bag-of-words vector, which is used as input vector for building the multiclass SVM classifier model. Then, the multiclass SVM classifier will be used in the testing stage to classify the detected hand posture captured from a webcam after constructing visual words vector for keypoints of the small image (50×50 pixels) that contains the detected hand gesture only. The testing stage proves the effectiveness of the proposed scheme in terms of accuracy and speed as the keypoints extracted represent the detected hand gesture only. Experiments show that the system can achieve satisfactory real-time performance regardless of the frame resolution size as well as high classification accuracy of 96.23% under variable scale, orientation and illumination conditions, and cluttered background. Three important factors affect the accuracy of the system, which are the quality of the webcam in the training and testing stages, the number of the training images, and choosing number of clusters to build the cluster model.

References

[1] A. El-Sawah, N. Georganas, and E. Petriu, "A prototype for 3-D hand tracking and gesture estimation," *IEEE Trans. Instrum. Meas.*, vol. 57, no. 8, pp. 1627–1636, Aug. 2008.

[2] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[3] A. Bosch, X. Munoz, and R. Marti, "Which is the best way to organize/classify images by content?" *Image Vis. Comput.*, vol. 25, no. 6, pp. 778–791, Jun. 2007.

[4] H. Zhou and T. Huang, "Tracking articulated hand motion with Eigen dynamics analysis," in *Proc. Int. Conf. Comput. Vis.*, 2003, vol. 2, pp. 1102–1109.

[5] B. Stenger, "Template based hand pose recognition using multiple cues," in *Proc. 7th ACCV*, 2006, pp. 551–560.

[6] L. Bretzner, I. Laptev, and T. Lindeberg, "Hand gesture recognition using multiscale color features, hierarchical models and particle filtering," in *Proc. Int. Conf. Autom. Face Gesture Recog.*, Washington, DC, May 2002.

[7] A. Argyros and M. Lourakis, "Vision-based interpretation of hand gestures for remote control of a computer mouse," in *Proc. Workshop Comput. Human Interact.*, 2006, pp. 40–51.

[8] C. Wang and K. Wang, *Hand Gesture Recognition Using Adaboost With SIFT for Human Robot Interaction*, vol. 370. Berlin, Germany: Springer-Verlag, 2008.

[9] A. Barczak and F. Dadgostar, "Real-time hand tracking using a set of co-operative classifiers based on Haar-like features," *Res. Lett. Inf. Math. Sci.*, vol. 7, pp. 29–42, 2005.

[10] Q. Chen, N. Georganas, and E. Petriu, "Real-time vision-based hand gesture recognition using Haar-like features," in *Proc. IEEE IMTC*, 2007, pp. 1–6.

[11] P. Viola and M. Jones, "Robust real-time object detection," *Int. J. Comput. Vis.*, vol. 2, no. 57, pp. 137–154, 2004.

[12] S. Wagner, B. Alefs, and C. Picus, "Framework for a portable gesture interface," in *Proc. Int. Conf. Autom. Face Gesture Recog.*, 2006, pp. 275–280.

[13] M. Kolsch and M. Turk, "Analysis of rotational robustness of hand detection with a Viola-Jones detector," in *Proc. 17th ICPR*, 2004, pp. 107–110.

[14] Y. Ke and R. Sukthankar, "PCA-SIFT: A more distinctive representation for local image descriptors," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2004, pp. II-506–II-513.

[15] H. Bay, A. Ess, T. Tuytelaars, and L. Gool, "SURF: Speeded up robust features," *Comput. Vis. Image Understand. (CVIU)*, vol. 110, no. 3, pp. 346–359, 2008.

[16] W. Zhao, Y. Jiang, and C. Ngo, "Keyframe retrieval by keypoints: Can point-to-point matching help?" in *Proc. 5th Int. Conf. Image Video Retrieval*, 2006, pp. 72–81.

[17] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2006, pp. 2169–2178.

[18] Y. Jiang, C. Ngo, and J. Yang, "Towards optimal bag-of-features for object categorization and semantic video retrieval," in *Proc. ACM Int. Conf. Image Video Retrieval*, 2007, pp. 494–501.

[19] L. Juan and O. Gwun, "A comparison of SIFT, PCA-SIFT and SURF," *Int. J. Image Process. (IJIP)*, vol. 3, no. 4, pp. 143–152, 2009.

[20] D. J. C. MacKay, *Information Theory, Inference, and Learning Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[21] J. Weston and C. Watkins, "Multi-class support vector machines," in *Proc. ESANN*, M. Verleysen, Ed., Brussels, Belgium, 1999.

[22] C.-C. Chang and C.-J. Lin, *LIBSVM: A Library for Support Vector Machines*, 2001. [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm

[23] C.-W. Hsu and C.-J. Lin, "A comparison of methods for multi-class support vector machines," *IEEE Trans. Neural Netw.*, vol. 13, no. 2, pp. 415–425, Mar. 2002.

[24] J. H. Friedman, "Another approach to polychotomous classification," Dept. Statist., Stanford Univ., Stanford, CA, 1997.

[25] N. Dardas, Q. Chen, N. Georganas, and E. Petriu, "Hand gesture recognition using bag-of-features and multi-class support vector machine," in *Proc. 9th IEEE Int. Workshop HAVE*, Phoenix, AZ, Oct. 16–17, 2010, pp. 1–5.

[26] J. M. Rehg and T. Kanade, "Visual tracking of high DOF articulated structures: An application to human hand tracking," in *Proc. Eur. Conf. Comput. Vis.*, 1994, pp. 35–46.

[27] J. Yang, W. Lu, and A. Waibel, "Skin-color modeling and adaptation," in *Proc. ACCV*, 1998, pp. 687–694.

[28] B. Zarit, B. Super, and F. Quek, "Comparison of five color models in skin pixel classification," in *Proc. ICCV Int. Workshop Recog., Anal. Tracking Faces Gestures Real-Time Syst.*, 1999, pp. 58–63.

[29] A. Ford and A. Roberts, "Color space conversions," Westminster Univ., London, U.K., Aug. 11, 1998.

[30] R. Gonzalez, R. Woods, and S. Eddins, *Digital Image Processing Using MATLAB*. Englewood Cliffs, NJ: Prentice-Hall, 2004.

[31] H. Jun and Z. Hua, "A real time face detection method in human-machine interaction," in *Proc. ICBBE*, 2008, pp. 1975–1978.

[32] W. Kelly, A. Donnellan, and D. Molloy, "Screening for objectionable images: A review of skin detection techniques," in *Proc. IMVIP*, 2008, pp. 151–158.

[33] Q. Zhu, C.-T. Wu, K.-T. Cheng, and Y.-L. Wu, "An adaptive skin model and its application to objectionable image filtering," in *Proc. ACM Multimedia*, 2004, pp. 56–63.

[34] K. Nallaperumal, S. Ravi, C. N. K. Babu, R. K. Selvakumar, A. L. Fred, C. Seldev, and S. S. Vinsley, "Skin detection using color pixel classification with application to face detection: A comparative study," in *Proc. IEEE Int. Conf. Comput. Intell. Multimedia Appl.*, 2007, vol. 3, pp. 436–441.

[35] A. J. Heap and D. C. Hogg, "Towards 3-D hand tracking using a deformable model," in *Proc. 2nd Int. Face Gesture Recog. Conf.*, Killington, VT, Oct. 1996, pp. 140–145.

[36] Y. Wu, J. Y. Lin, and T. S. Huang, "Capturing natural hand articulation," in *Proc. 8th Int. Conf. Comput. Vis.*, Vancouver, BC, Canada, Jul. 2001, vol. II, pp. 426–432.

[37] B. Stenger, P. R. S. Mendonça, and R. Cipolla, "Model-based 3D tracking of an articulated hand," in *Proc. Brit. Mach. Vis. Conf.*, Manchester, U.K., Sep. 2001, vol. I, pp. 63–72.

[38] Q. Chen, N. Georganas, and E. Petriu, "Hand gesture recognition using Haar-like features and a stochastic context-free grammar," *IEEE Trans. Instrum. Meas.*, vol. 57, no. 8, pp. 1562–1571, Aug. 2008.

[39] L. Bretzner, I. Laptev, and T. Lindeberg, "Hand gesture recognition using multi-scale colour features, hierarchical models and particle filtering," in *Proc. 5th IEEE Int. Conf. Autom. Face Gesture Recog.*, 2002, pp. 405–410.

[40] S. McKenna and K. Morrison, "A comparison of skin history and trajectory-based representation schemes for the recognition of user-specific gestures," *Pattern Recognit.*, vol. 37, no. 5, pp. 999–1009, May 2004.

[41] K. Imagawa, H. Matsuo, R. Taniguchi, D. Arita, S. Lu, and S. Igi, "Recognition of local features for camera-based sign language recognition system," in *Proc. Int. Conf. Pattern Recog.*, 2000, vol. 4, pp. 849–853.

[42] H. Jhuang, T. Serre, L. Wolf, and T. Poggio, "A biologically inspired system for action recognition," in *Proc. IEEE ICCV*, 2007, pp. 1–8.

[43] J. Niebles, H. Wang, and L. Fei-Fei, "Unsupervised learning of human action categories using spatial-temporal words," *Int. J. Comput. Vis.*, vol. 79, no. 3, pp. 299–318, Sep. 2008.

[44] C. Schüldt, I. Laptev, and B. Caputo, "Recognizing human actions: A local SVM approach," in *Proc. ICPR*, 2004, pp. 32–36.

[45] A. Gilbert, J. Illingworth, and R. Bowden, "Fast realistic multi-action recognition using mined dense spatio-temporal features," in *Proc. IEEE ICCV*, 2009, pp. 925–931.

[46] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," in *Proc. IEEE Conf. CVPR*, 2008, pp. 1–8.

[47] M. Marszalek, I. Laptev, and C. Schmid, "Actions in context," in *Proc. IEEE Conf. CVPR*, 2009, pp. 2929–2936.

[48] L. Yeffet and L. Wolf, "Local trinary patterns for human action recognition," in *Proc. IEEE ICCV*, 2009, pp. 492–497.

[49] S. Savarese, J. Winn, and A. Criminisi, "Discriminative object class models of appearance and shape by correlations," in *Proc. IEEE Conf. CVPR*, 2006, pp. 2033–2040.

[50] A. Agarwal and B. Triggs, "Hyperfeatures: Multilevel local coding for visual recognition," in *Proc. ECCV*, 2006, pp. 30–43.

[51] M. Marszalek and C. Schmid, "Spatial weighting for bag-of-features," in *Proc. IEEE Conf. CVPR*, 2006, pp. 2118–2125.

[52] K. Grauman and T. Darrell, "The pyramid match kernel: Discriminative classification with sets of image features," in *Proc. IEEE ICCV*, 2005, pp. 1458–1465.

[53] Z. Zhao and A. Elgammal, "Spatiotemporal pyramid representation for recognition of facial expressions and hand gestures," in *Proc. IEEE Int. Conf. Autom. Face Gesture Recog.*, 2008, pp. 1–6.

[54] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features," in *Proc. IEEE Int. Workshop Visual Surveillance Perform. Eval. Tracking Surveillance*, 2005, pp. 65–72.

[55] G. Willems, T. Tuytelaars, and L. Van Gool, "An efficient dense and scale invariant spatio-temporal interest point detector," in *Proc. ECCV*, 2008, pp. 650–663.

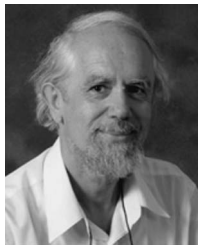
[56] I. Junejo, E. Dexter, I. Laptev, and P. Pérez, "Cross-view action recognition from temporal self-similarities," in *Proc. ECCV*, 2008, pp. 293–306.

[57] A. Kläser, M. Marszalek, and C. Schmid, "A spatio-temporal descriptor based on 3D-gradients," in *Proc. BMVC*, 2008, pp. 995–1004.

[58] S. Marcel, "Hand posture recognition in a body-face centered space," in *Proc. Conf. Human Factors Comput. Syst. (CHI)*, 1999, pp. 302–303.
[59] W. Chung, X. Wu, and Y. Xu, "A real time hand gesture recognition based on Haar wavelet representation," in *Proc. IEEE Int. Conf. Robot. Biomimetics*, 2009, pp. 336–341.
[60] Y. Fang, K. Wang, J. Cheng, and H. Lu, "A real-time hand gesture recognition method," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2007, pp. 995–998.
[61] L. Yun and Z. Peng, "An automatic hand gesture recognition system based on Viola-Jones method and SVMs," in *Proc. 2nd Int. Workshop Comput. Sci. Eng.*, 2009, pp. 72–76.
[62] Y. Ren and C. Gu, "Real-time hand gesture recognition based on vision," in *Proc. Edutainment*, 2010, pp. 468–475.
[63] A. L. C. Barczak, F. Dadgostar, and C. H. Messom, "Real-time hand tracking based on non-invariant features," in *Proc. IEEE Instrum. Meas. Technol. Conf.*, Ottawa, ON, Canada, 2005, pp. 2192–2197.
[64] A. Micilotta and R. Bowden, "View-based location and tracking of body parts for visual interaction," in *Proc. BMVC*, 2004, pp. 849–858.
[65] C. Messom and A. Barczak, "Fast and efficient rotated haar-like features using rotated integral images," in *Proc. ACRA*, 2006, pp. 1–6.
[66] L. Shi, Y. Wang, and J. Li, "A real time vision-based hand gestures recognition system," in *Proc. 5th ISICA*, 2010, pp. 349–358.

Nasser H. Dardas received the B.Sc. degree in electrical engineering—communication from Jordan University of Science and Technology, Irbid, Jordan, in 1988, the M.Sc. degree in electrical engineering—communication from University of Jordan, Amman, Jordan, in 1999, and the M.A.Sc. degree in electrical engineering—computer networks from Carleton University, Ottawa, ON, Canada, in 2008. He is currently working toward the Ph.D. degree in electrical and computer engineering at University of Ottawa, Ottawa, ON, Canada.

His research interests include visual object detection and tracking, and image processing. His current research topic is focused on vision-based hand gesture detection and recognition in real time.



Nicolas D. Georganas (F'90) received the Dipl.Ing. degree in electrical and mechanical engineering from the National Technical University of Athens, Athens, Greece, in 1966 and the Ph.D. degree (*summa cum laude*) in electrical engineering from the University of Ottawa, Ottawa, ON, Canada, in 1970.

In 2004, he became the Founding Editor-in-Chief of the *ACM Transactions on Multimedia Computing, Communications, and Applications*. He has published over 360 technical papers and is the coauthor of the book *Queueing Networks—Exact Computational Algorithms: A Unified Theory by Decomposition and Aggregation* (Cambridge, MA: MIT Press, 1989). His research interests include multimedia communications, collaborative virtual environments, Web telecollaboration applications, intelligent Internet sensors and appliances, and telehaptics.

Dr. Georganas is a Fellow of the Engineering Institute of Canada, the Canadian Academy of Engineering, and the Royal Society of Canada. He was the recipient of the Killam Prize for Engineering in 2002, the Queens Golden Jubilee Medal in 2003, the Canadian Award in Telecommunications Research in 2006, the first IEEE Canada Computer Medal, the ORION Leadership Award, and an Honorary Doctorate degree from the National Technical University of Athens in 2007. In 2004, he received an Honorary Doctorate degree from the Technical University Darmstadt, Darmstadt, Germany. In 2005, he was recognized as a Pioneer of Computing in Canada by the International Business Machines Corporation Centre of Advanced Studies. In 2007, he was invested as Officer of the Order of Canada: the highest honor for a Canadian. From 2007–2009, he was Secretary of the Academy of Science, Royal Society of Canada.