Multimedia Tools and Applications (2022) 81:40311–40342
https://doi.org/10.1007/s11042-022-12870-8

Long Hands gesture recognition system: 2 step gesture recognition with machine learning and geometric shape analysis

Pavel A. Popov [1,2] · Robert Laganière [1]

Received: 22 December 2020 / Revised: 18 March 2021 / Accepted: 10 March 2022 /
Published online: 7 May 2022

Abstract
A two stage real-time hand gesture recognition system is presented. It combines a machine learning trained detection step with a colour processing contour shape validation step. The detection step is done with either Adaboost Cascades or Support Vector Machines using HOG features. The system achieves a low false positive rate and a sufficient true positive rate necessary for robust real-time performance. It performs well compared to MobileNets a state of the art Neural Network for mobile real-time applications.

Keywords SVM · Adaboost · HOG · Hand gesture recognition · MobileNets · Neural network · Shape signature · Contour shape analysis · Adaptive colour segmentation

1 Introduction

Hand Gesture Recognition is an area of research which focuses on recognizing and tracking human hands in video. The goal is to allow users to use their hands to issue powerful and intuitive commands in user interfaces and to improve the state of human computer interaction. People communicate quite a bit with their hands and use their hands to interact with the world. The question of just how we communicate with our hands is of continued cognitive and psychological research interest [3, 9, 10, 21, 37]. By solving the challenges that still surround

✉ Pavel A. Popov
Pavel.Popov@nrc–cnrc.gc.ca

Robert Laganière
laganier@eecs.uottawa.ca

[1] University of Ottawa, Ottawa, ON K1N 6N5, Canada

[2] National Research Council, Ottawa, ON K1A 0R6, Canada

hand gesture recognition human new avenues of creativity and human computer functionality will be opened.

The first step to creating powerful user interfaces is to have robust and reliable hand gesture recognition. We propose a system in this work that is designed to achieve two goals so that it can be viable for user interfaces. These goals are:

- Real-time multi scale hand gesture recognition from a video input
- A very low false positive rate for gesture recognition.

Our proposed Gesture Recognition system, called Long Hands, is a 2 step system with each step being designed to achieve one of these goals.

The first goal is achieved in the first step with machine learning. The system has 2 options for multi scale detection, Adaboost Cascades with HOG Features, and Support Vector Machines with HOG Features. A dataset of hand gestures was collected for the task. This first step detects hand gestures anywhere in an input frame and provides hand position estimates for the validation functions of step 2 of the system.

The second step achieves the second goal of a low false positive rate. To be a viable option for a user interfaces the amount of objects that a gesture recognition system rejects becomes much more important than the amount of gestures that it recognizes. While in the worst case a delay in reading commands from a user may be acceptable, false positives triggering unwanted commands is generally unacceptable. The second step uses colour processing to adaptively segment the user's hands. These segmentations are processed by shape analysis tests that confirm the correct gestures and reject incorrect gestures or objects mistaken for gestures.

Our proposed gesture recognition system works using only 2D video input to recognize hand gestures in real-time. Our work is of interest to biometry applications because it achieves its high discrimination capabilities using hand models which take advantage of hand geometry. Biometry has been extensively investigated in the context of facial recognition [13, 16, 81, 91], and much success has been achieved with a focus on geometry. Hand gesture detection would be a great complement to face recognition in both security and user interface applications because identifying and tracking hands is a natural extension of facial recognition in order to determine what a human subject is doing. Landmarking is a very important element of biometric facial recognition systems and our gesture recognition and hand tracking system makes significant efforts to have stable hand landmarks which would be an asset for a biometric verification system. It is worth noting that computational load is a concern in facial recognition systems and having computationally lightweight gesture recognition systems makes using both in a single system much more feasible. In addition it can be seen that facial recognition, having been a long studied problem, benefits from publically available datasets that allow for comparative work. There is currently a lack of large publically available datasets for the 2D video-only gesture recognition problem which makes comparisons between research approaches very difficult. The work presented in this paper addresses the issues of computational load and shows that real-time GPU-independent solutions to the 2D video-only hand gesture recognition problem are possible. It also contributes datasets for the evaluation of said systems to allow the research community to be able to do comparative work for 2D video only static hand gesture recognition, and 2D video-only hand tracking.

2 Related work

Hand Gesture Recognition is a diverse field of research that encompasses both static and dynamic hand gestures as well as hand tracking. Whatever the goal of a hand gesture recognition system may be, there are some common elements. Good features are needed to either detect hands or recognize hand gestures, and some sort of classification is usually used. Real-time performance is also desired for user interface applications.

2.1 Features

Colour is a prevalent feature in hand gesture recognition and tracking methods [2, 6, 11, 26, 28, 32, 38, 39, 57, 59, 71, 77, 78, 80, 86, 90, 95–97]. Colour features are often used to segment hands. Skin detectors and colour histograms are both interchangeable terms for defining a range of colours to use for segmentation which can be pre defined [2, 6, 26, 28, 32, 39, 59, 71, 77, 80, 86, 95–97] or defined adaptively [38, 57, 78, 90, 95] also adapts to the colour of user's hand by using k-means clustering and colour to partition input images. [11, 95] use simple background subtraction to segment user's hands which can be thought of as a difference of colour.

Depth is also used because it is particularly useful for segmentation. The availability of cheap depth sensors have made it a viable option [8, 75, 76, 95] use depth thresholding to segment the nearest object in their methods. Depth is also used for 3D hand gesture recognition in [5, 8, 35, 54, 55, 70, 87]. [44] uses a pseudo depth image feature achieved with a CNN model.

Building upon the results of segmentation, geometric shape features offer a way to analyze the contour shapes of hand gestures [6, 8, 11, 17–20, 29, 34, 38, 39, 57, 71–76, 95, 96] all use geometric shape features. The works in [8, 11, 34, 39, 57, 72–74, 95, 96] all use some sort of edge or contour based analysis. A variety of geometric tests are used in [8, 11, 57, 72–74, 95, 96], relating to convexity defect and contour curvature analysis, to find the location of the palms and extended visible fingers of hands. A variety of logic and shape tests are used directly on the contour boundaries and the convexity defects to determine fingertip and palm locations in [8, 11, 95, 96], extended visible finger count in [11, 57, 95], gesture pose in [8, 11, 57, 72–74, 95] and [8, 57] are even able to label the extended fingers.

Another way to analyze hand contours is by using shape signatures as was done in [7, 17, 38, 75, 76]. The works in [17, 38, 75, 76] use 1 dimensional shape signatures for classification. [81] proposes a slightly different shape signature related method for counting fingers. The authors use binary skeletons of hand contours with a normalized radial function. A trained mathematical model validates skeletal branches as fingers.

Another major category of features used in hand gesture recognition is vectorizable features generated by mathematical transforms. Vectorizable features are needed in order to use many powerful machine learning methods. HOG features are used in [28, 30, 47, 80, 82, 89, 98]. Research continues in improving HOG with [47] proposing a skin colour modification called SCHOG for hand detection. Haar features have been used in the work of [26, 30, 48, 72–74]. The authors of [26, 48] have also proposed improvements to Haar features specific for their applications. FREAK features are used for hand detection in [84]. [92] and [14] use SURF features, and [14] also uses SIFT. Several methods use descriptors to vectorize segmented contour shapes. Contour Sequences are used in [18–20, 29] and Hu moments are used in [29, 71, 90]. The work in [6] uses Fourier Descriptors, and [17] uses wavelet transform. These descriptors take contours of variable length and encode the shape information into a vector of

fixed length. The works presented in [75, 76] both use near convex decomposition to threshold finger shapes in the contour shape signatures and these shapes are then indexed by size and location to make fixed length shape signature vectors. Haralick features are used in [90] which describe texture information.

Some other types of features have been proposed in various works for hand gesture recognition and related applications. These can generally categorized into motion and statistics features. Motion features are used in [2, 38, 71, 97] combine motion information with skin or skin and depth segmentation [41] reduces the complexity of its motion path features by calculating the Log Path Signature. The authors of [59, 84] use statistical features to find the areas of interest based on saliency and entropy. Hough features are used in [51] for ASL recognition.

2.2 Classification methods

Classification methods are a common ground for Hand Gesture recognition systems. Whether the application is static or dynamic gestures, or a related field, many works make use of similar classification approaches.

The Adaboost Cascade of Weak Classifiers and Adaboost inspired methods have continued to be used for detection and recognition tasks [15, 22, 30, 40, 48, 53, 72–74, 87, 89, 98]. Adaboost was used for hand detection in [75], with classification being handled by PCA. Rautaray and Agrawal use Haar feature cascades in several works [72–74], for hand detection. Nguyen et al. [53] also use a Haar Cascade as part of their gesture recognition system. [40] and [89] both do complete gesture recognition using Adaboost [89] even proposes a multiclass structure for Adaboost. Messom and Barczak [48] propose diagonal Haar features to make Adaboost more robust to rotation. A number of other classifiers have been used with the multistage idea. Cascades of Support Vector Machines have been used in [22, 98] and Cascades of weak regressors have been used in [15, 87]. Both of the regressor works have similar training methods as Adaboost, designed to minimize the error in each stage.

SVMs have become a popular classification method for Hand Gesture Recognition and related fields. SVMs have been used for hand detection [22, 47, 56, 60, 98], and hand gesture recognition [5, 6, 14, 19, 26, 42, 52, 59, 60, 71, 79, 80, 82, 92]. This list of works is not exhaustive and many more examples exist in the literature. SVMs are simple to train and with linearly separable data, or a good choice of a non-linear kernel, and they generally perform well. The major drawback is that the more samples are used to train an SVM classifier the larger the model tends to become. This is because the training is done by finding the optimal margin of separation between the difficult to classify samples, which are referred to as Support Vectors. Larger sample sizes produce more of these support vectors which increase the computational cost of the model. Therefore keeping an SVM model small enough to run in real-time can be a challenge. SVMs are highly adaptive to classification problems. Many works use SVMs to recognize static hand gestures [6, 14, 19, 26, 42, 52, 53, 60, 79, 80, 82, 92], which means recognizes the shape features of a hand gesture in one instance or frame. SVMs have likewise been adapted for dynamic hand gesture recognition [5, 71], which requires the classification of motion features as well as shape features from a series of frames. SVMs are also very versatile, in [79] an SVM is combined with a CNN feature extractor in order to classify static hand gestures.

Neural Networks are a competing classification method which can automatically select features to use for classification. This reduces the repeated trial and error associated with feature selection. Neural networks require more data for training compared to other methods

but they have the advantage of keeping the model the same size even if the dataset increases. Once a Neural Network has been trained with a large enough dataset such as ImageNet, it can be reused and adapted for a different problems with a process called retraining. Neural Networks have been used to solve hand gesture recognition problems [1, 4, 17, 18, 20, 23–25, 29, 31, 32, 36, 41, 44, 46, 49, 51, 54, 55, 86, 90, 93, 94, 96, 99]. Methods have been proposed for both static [1, 17, 18, 20, 23, 29, 31, 32, 36, 44, 46, 51, 86, 90, 94, 96] and dynamic hand gestures [4, 41, 49, 93, 99]. Neural Networks have also been used to fit 3D hand models for hand pose estimation [54, 55] which can be used for either static or dynamic gestures. Hand Detection is performed in [77] which achieves very good skin segmentation with a neural network. [65] uses neural networks to both temporally segment and classify dynamic hand gestures. Hwang and Lee [29] propose a system specifically for a user interface. [94] classifies ego centric hand gestures and finds the locations of visible fingertips for UI motivations. User interfaces are also a motivation for the work done in [1, 4, 17, 23, 31, 32, 36, 49, 54, 99]. [23] also uses ego centric data for their work. Koller et al. in [36] present how to use weakly labeled data to train neural networks on larger datasets. This work achieves success in static gesture recognition that is independent of large changes in orientation of the hand poses, and it also uses a large set of hand pose classes. [99] and [4] analyze motion paths to allow users to draw numbers and letters. Swipe gestures are recognized by the system of Molchanov et al. [49]. [54, 55] actively fit 3D models to the user's displayed hand. MobileNets [25] is a very interesting and powerful recent neural network classifier capable of running in real-time on mobile devices. It has been used for comparisons in a range of applications. Computational savings compared to other networks are achieved with separable convolutions. Mobilenets has been built to reduce computation load with real-time applications in mind, making it an excellent candidate for comparisons with our work.

2.3 Relevance to proposed hand gesture recognition system

The Hand Gesture Recognition system presented in this work makes significant use of Adaboost Cascades with HOG features and Support Vector Machines with HOG features to achieve multiscale gesture detection. HOG Features are used because they are invariant to light and scale transformations. A MobileNets Neural Network model is used for a comparison system to evaluate the performance of the Gesture Recognition system to state of the art neural network solutions for mobile vision applications.

The validations functions of the system use adaptive colour processing with 2D colour histograms to segment the user's skin from the background. Geometric shape features of contours and their shape signatures are used to validate and invalidate gesture detections generated by machine learning trained multiscale detectors.

Depth data is not used by our system. The system presented here is designed strictly for 2d video. Depth could be an interesting direction for further research.

Some works which achieve good results use methods that only work for uniform backgrounds [18, 27, 30, 42, 44, 51, 54, 57, 72–74, 78, 80, 82, 86]. Other methods work in cluttered and complex backgrounds [1, 2, 4–6, 14, 19, 20, 22, 23, 26, 28, 29, 32, 36, 38–41, 43, 47, 53, 55, 56, 59, 70, 71, 75–77, 83–85, 89, 90, 92–95, 97, 98] which dramatically increases their usability for user interfaces. The system presented in this paper was designed to perform hand gesture recognition in complex backgrounds.

Consulting recent literature on hand gesture recognition and hand tracking reveals some very interesting needs in the research community. Progress continues in 3D and depth

depended methods such as [58] however these cannot be used in a strictly 2D video only context. Datasets continue to be small as is the case for [1, 27, 31, 79, 90] or publically unavailable [58] or the method is only designed for uniform backgrounds [31, 58]. [31] use two datasets one that is small at only 960 samples per class, which the authors boost by data augmentation, and the other is bigger with 2000 samples per class, however it is comprised of near infrared images and is not applicable to a 2D video only context. The authors of [90] also use data augmentation to enlarge a small dataset. The datasets of [31, 90] are significantly smaller than the dataset presented in this paper for video only static hand gesture recognition which has 8000–10,000 images per class. Furthermore if the authors of [4, 90] can make use of data augmentation to boost their dataset size, similar benefits can also be gained from augmenting the much larger dataset presented in this paper. Another interesting observation is that known methods still continue to be used such as ELM [90], which is a simple type of neural network, and SVM [79]. This demonstrates that there is still interest in using more well known methods for solving hand gesture recognition problems likely due to simpler implementation and lighter computational load. Additionally the focus on CNN based methods indicates that many newer works will be GPU dependent, which means that GPU independent solutions that have lower computational load can make significant contributions to the research field, and are also more attractive from a practical implementation perspective.

With regard to hand tracking similar research needs are demonstrated. [58, 88] are depth dependent and thus not applicable in a 2D video only context. However [58] correct for motion blur in their depth based tracking method with a gyroscope, which echoes the idea that multiple collaborating tracking strategies achieve better results, an idea present in the tracking extension of the Gesture Recognition system presented in this paper. [12, 45] present 2d hand tracking approaches. [45] presents a Kernelized Correlation Filter and [12] uses a Lucas-Kanade optical flow with YCrCb colourspace skin segmentation. Both however do not provide datasets, with [12] preferring to use user test cases for evaluation. Mueller et al. [50] present an interesting well performing method which fits a 3D hand model to 2D video. This method has significant drawbacks because it requires 3D data to train and improve. Thus it is fundamentally a 3D method that cannot work in a 2D vision only context because it relies on 3D data. Furthermore it is GPU dependent which will means increased hardware cost for implementation compared to computationally lighter CPU only methods. Lastly Kapidis et al. [33] is an interesting work for the different but related context of ego centric gesture recognition. In the context of our system the user faces a camera and makes gestures. In egocentric gesture recognition the user wears a camera and gestures are recognized from a user point of view perspective. This work uses a large dataset which is publically available, however the dataset is designed for a different use case context. Nevertheless this system presents a great direction for future work. However the main thing of interest in this article is that they use a combination of Kalman Filter and Hungarian algorithm for their tracker to achieve performance comparable to neural networks by using only a fraction of the input for model training. This demonstrates the research interest in computationally lighter solutions to hand tracking problems. The tracking is done using detections produced by a CNN which ultimately makes their system GPU dependent and thus computationally heavy. In light of the need for datasets and computationally lighter solutions the gesture recognition system presented in this work with its hand tracking capabilities, and its datasets, represent important contributions to the research field.

3 Gesture recognition algorithm overview

The Gesture Recognition Algorithm detects hand gestures in real-time from a video, anywhere in the video frame. There are two main steps. The first is a detection step, which uses machine learning trained detectors to locate potential hand gestures. The second is the validation step, which uses shape information to either validate or reject gesture detections (Fig. 1).

The detection step is done with either HOG Adaboost Cascades, or HOG SVMs. The validation step is performed using both geometric contour shape analysis, and contour shape signature analysis. A hand gesture that has been detected, and then validated, is said to have been recognized by the system.

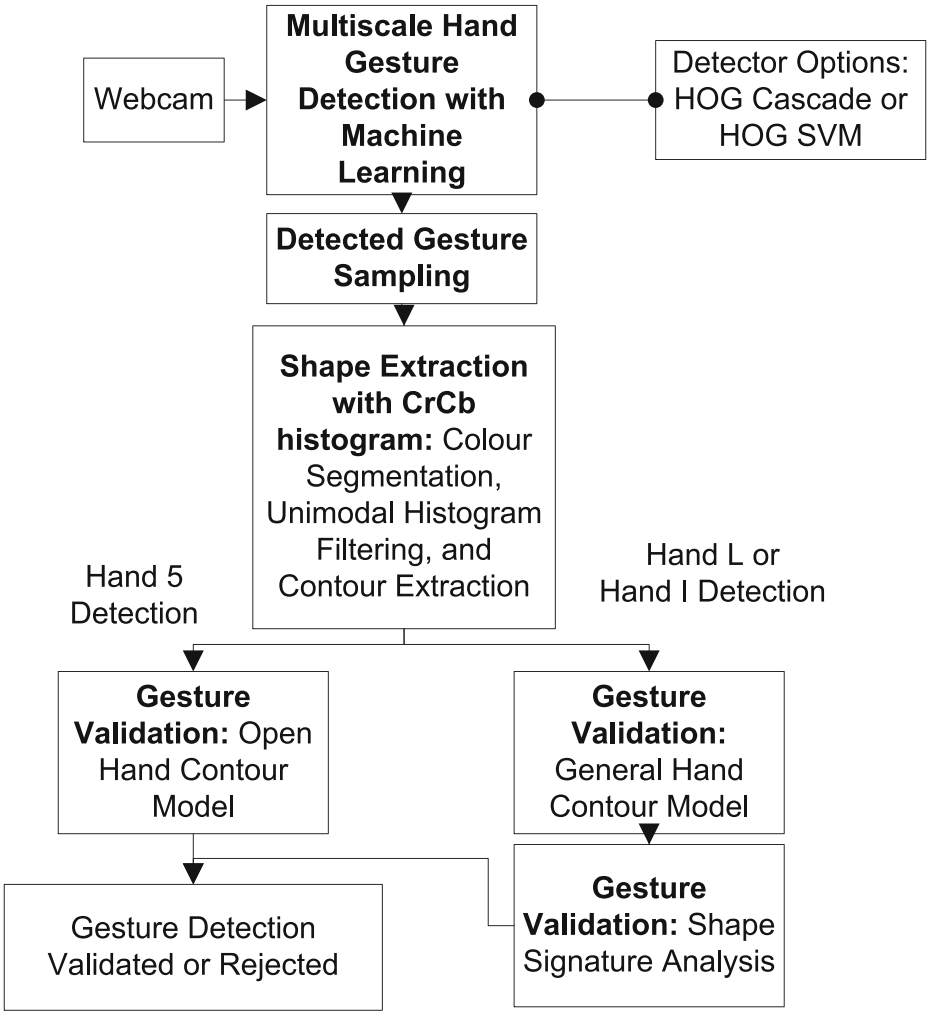

Fig. 1 Gesture Recognition System Flowchart

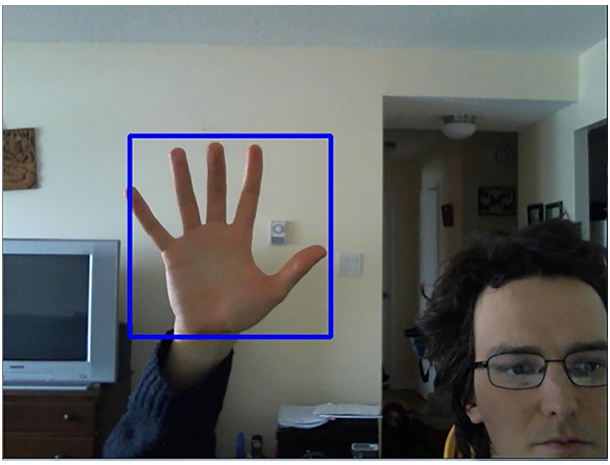

Fig. 2 Example of Multiscale Detection using Machine Learning

3.1 Multiscale hand gesture detection with machine learning

Machine learning is used to provide an estimate of a hand gesture location (Fig. 2). These detections are processed by the validation functions to filter out false positives and to keep true positives. 2 types of multiscale detectors were considered: Adaboost cascades with HOG features, and Support Vector Machines with HOG features.

3.2 Gestures used

There are 3 hand gestures used for the hand gesture recognition system. The gestures are indicated in Fig. 3.

3.3 HOG feature

Two different configurations of the system were implemented, one using Adaboost detectors and the second using SVM detectors. Both types of detectors used Histogram of Oriented Gradient (HOG) features. The features catalogue the gradient changes and their orientation within neighbourhoods of pixels. This creates different signatures from different areas in images which can then be used for classification. HOG features were used because they are

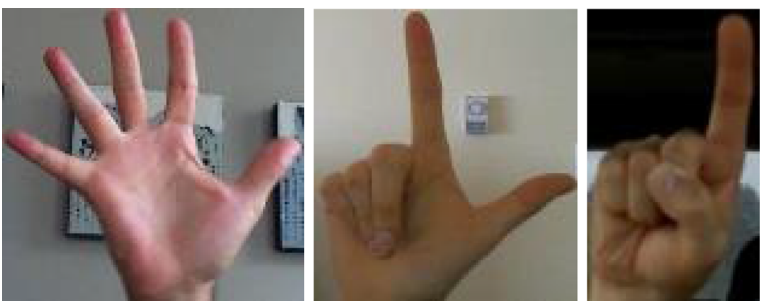

Fig. 3 Gestures recognized by the system. Hand 5 (Left) Hand L (Center) Hand I (Right)

light and scale transformation invariant when used with multiscale processing, which means they are very useful for detections in complex backgrounds.

3.4 HOG Cascade configuration

One Adaboost HOG Cascade detector was trained for each of the 3 gestures (Fig. 4). The detectors were trained with raw large backgrounds as negative images, and cropped gestures images as positives. Each detector was trained on positive images of its gesture vs. the negative images. The training dataset was 7813 images for hand 5, 6401 for hand L, 6958 for hand I, and 35,483 negative backgrounds. This corresponded to 80% of the overall dataset.

3.5 HOG SVM configuration

The training of the HOG SVMs was done with positive hand gesture samples vs. negative patches. The negative frames were not used whole, and were instead broken down into 160 × 160 non overlapping patches. The 35,483 negative backgrounds generated 247,148 negative background patches. Because the goal was to create a real-time detector, linear kernels were used. The entire training dataset was randomly sampled to train a 1500 hand 5 positive versus 4000 background patch negative SVM configuration. This configuration performed well in online testing. This is henceforth referred to as the small sample 1.5:4 ratio configuration. Using the entire training dataset resulted in larger and slower models that did not perform well.

Using multiple SVM stages was investigated as an option to increase the performance of the system. The idea is to use several SVM detectors in sequence to refine the detections. 4 configurations were investigated, a 1 stage configuration, a 1 stage run twice, a 2 stage configuration, and a 3 stage configuration.

The SVMs are run in sequence for each gesture (with the exception of the 1 stage configuration) in the following manner (Fig. 5):

The initial detections of the 1st (lowest) stage are progressively filtered out by the higher stages, and only the best detections are kept for the validation step.

The idea behind using multiple SVMs was that if the negative patch samples were partition by the magnitude of their HOG, a better partitioning of the vector space can be achieved with multiple linear SVMs as opposed to one. Multiple linear decision boundaries can be combined to make better decision surfaces in the vector space. The dimensionality of the problem, and thus the SVM model, can be reduced according to [22]. The other benefit is that more of the dataset of negative patches can be used for the training, thus creating a more robust detector.

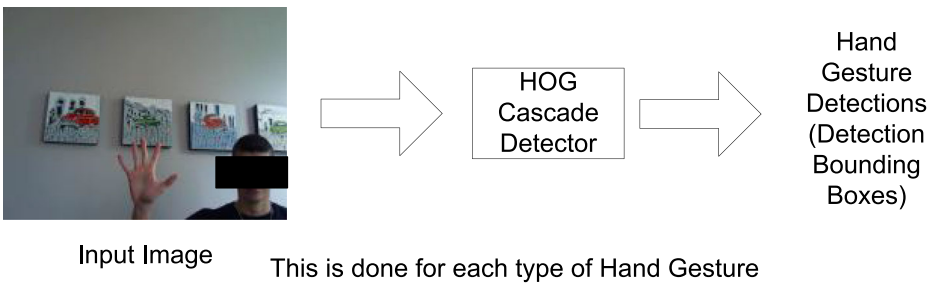

Fig. 4 HOG Adaboost Gesture Detection Architecture

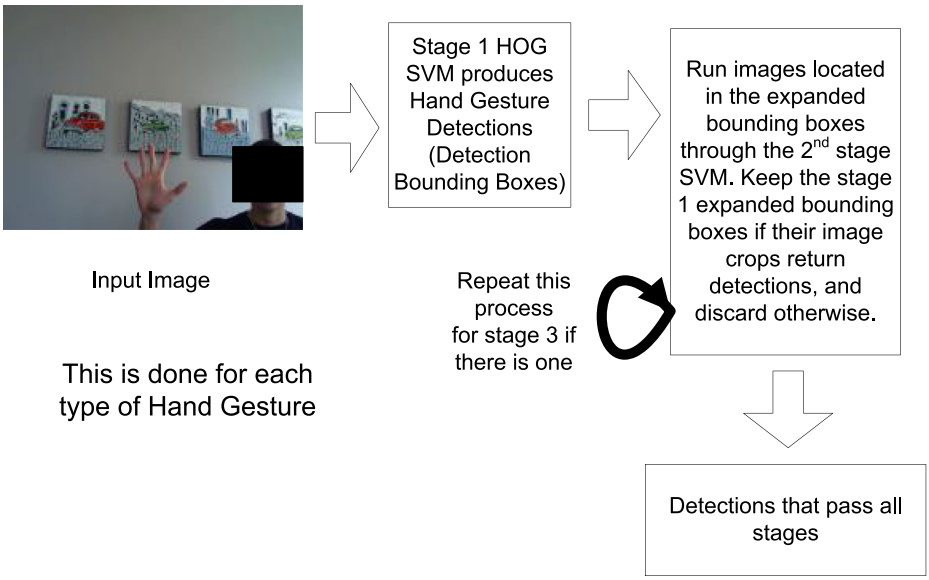

Fig. 5 HOG SVM Gesture Detection Architecture

Reducing the dimension increases the speed of the system because reduced dimension means fewer support vectors.

Experimentally it was determined that even a 1 stage SVM produces better detection results with a 2nd pass from its 1 stage SVM. This second pass serves to eliminate some or sometimes all of the false detections, and leaves fewer and better detections to be processed by the validation step (Fig. 6).

In order to make multiple stages the negative patches were partitioned according to magnitude of the HOG gradient. All the trained SVMs used the same 1500 randomly selected hand 5 patches and 4000 randomly selected negative patches from each gradient partition used. A performance comparison was done with the test dataset of all 3 Multistage SVM configurations being evaluated and the 1 stage SVM configuration (Fig. 7 and Table 1).

As stages are added there is a trade off. The false positive rate decreases as you add a second stage, and so does the true positive rate. It was decided that the 2 stage configuration would be used because it had the lowest false positive rate. Qualitatively, it also had better

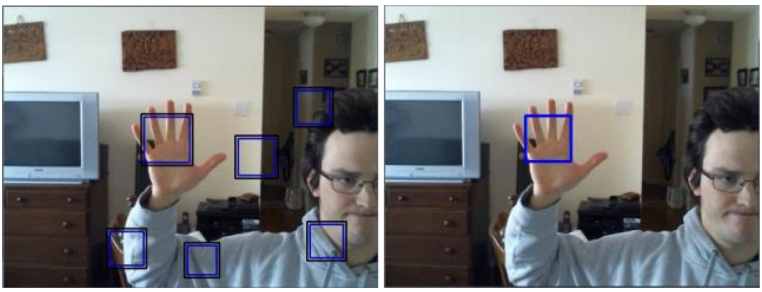

Fig. 6 The left image shows the detections after 1 pass with a 1 stage HOG SVM. The right image shows the detections that remain after a 2nd pass with the same 1 stage HOG SVM

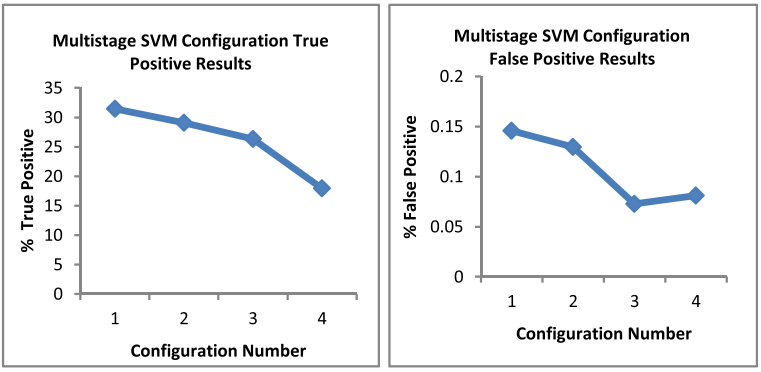

Fig. 7 Multistage SVM result graphs

Table 1 Multistage SVM results

Configuration Number	Number of Stages	True Positive Rate %	False Positive Rate %
1	1 Stage System	31.47233202	0.145619286
2	1 Stage run 2x	29.0513834	0.129439366
3	2 Stage	26.33399209	0.072809643
4	3 Stage	17.93478261	0.080899604

localized bounding boxes than the other configurations. These boxes crop the visible hand gestures more closely. The results of the 3 stage configuration indicate that using more than 2 stages means a significant drop in performance. The 2 stage SVM configuration was selected as the best multistage configuration, and it was used to build the 3 gesture SVM Gesture Recognition system.

3.6 Detected gesture sampling

Each type of gesture detection has its own specific sampling pattern. The validation functions are run using each sampling point. This validates true gesture detections and rejects false detections (Fig. 8).

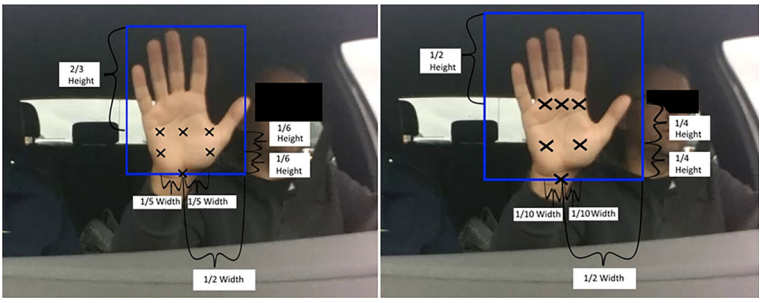

Fig. 8 Example of Detection Boxes Sampling Patterns used for the Cascade(Left) and SVM(Right) configuration

3.7 Gesture validation

The gesture validation procedure is applied on each detection box. The procedure takes as input sample points provided by a given detection box, and the video frame. The procedure then segments out the user's hands and the corresponding contours are evaluated to determine if the shapes are correct for the detected hand gestures. The validation procedure uses the Open Hand Model to validate hand 5 gesture. For the hand L and hand I gestures the extended fingers are counted with the General Hand Model, and shape signature analysis. Hand L, and Hand I gestures, must have 2, and 1 fingers, respectively.

4 Affects of using the validation functions

The affects of the validation functions have been tested using the test dataset. Here is a side by side comparison of the Cascade and SVM configurations (Table 2).

The validation functions significantly reduce the false positive rate for all gestures. While they also reduce the true positive rate, the low false positive rate is required for user interface applications. The true positive rates are none the less high enough to make the systems responsive in real-time online operation.

4.1 Shape extraction with CrCb histogram: Unimodal histogram filtering and back projection

The validation procedure is run at every sample point provided by a detection box using the input frame (Fig. 9).

At a given sample point the 30 × 30 area around it is used to calculate its colour signature. The YCrCb colour space is used because it is very good at isolating light intensity from colour information. The CrCb 2 dimensional histogram, otherwise known as the colour signature of the area is found. This colour signature will be later used to segment the frame. The CrCb 2d histogram or colour signature is a 2d binary map that records whether or not a given Cr and Cb colour value pair is present in the sampling area defined by the ROI.

The 1d histograms of the Cr and Cb channels are also found, and from this the Cummulative Distribution Functions (CDFs) are calculated. The CDFs of the Cr and Cb

Table 2 Affects of using the validation functions (TP = True Positive, FP = False Positive)

	Without Validation Functions		With Validation Functions Best Configuration		With Validation Functions without Unimodal Filtering		With Validation Functions with Unimodal Filtering	
	Cascade System		Cascade System		Cascade System		Cascade System	
Gesture	% TP	% FP	% TP	% FP	% TP	% FP	% TP	% FP
hand 5	84.68	2.217	31.67	0.0485	22.18	0.04045	31.67	0.04854
hand L	92.49	49.41	25.58	0.8943	26.43	1.718	25.58	0.8943
hand I	98.88	46.4	16.19	1.008	15.91	0.9846	10.5	0.6988
	SVM System		SVM System		SVM System		SVM System	
Gesture	% TP	% FP	% TP	% FP	% TP	% FP	% TP	% FP
hand 5	97.13	81.89	26.33	0.07281	18.083	0.0809	26.33	0.07281
hand L	91.82	81.13	20.51	1.828	19.6	3.946	20.51	1.828
hand I	100	96.26	10.11	2.898	12.62	3.954	10.11	2.898

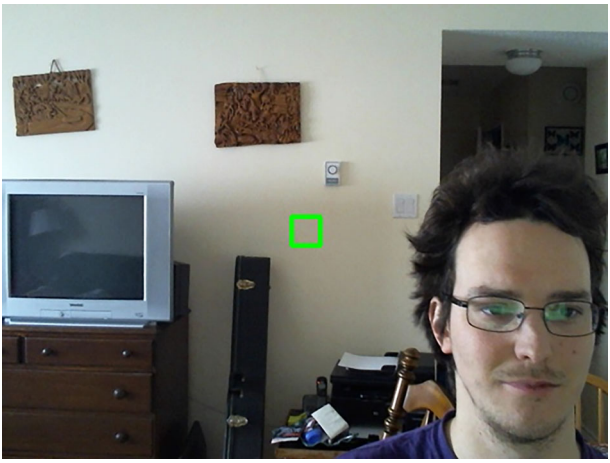

Fig. 9 Sampling zone example. Zone outlined in green

channels are evaluated to determine if the colour distribution that their histograms represent is unimodal. If both 1d Cr and Cb histograms are unimodal then the 2d colour signature is augmented with unimodal histogram filtering (Fig. 10).

Unimodal histogram filtering is done by taking the limits of the positive regions of increase in the CDFs and using these limits to draw a box on the 2d colour signature. This adds colour pairs to the colour signature which are highly similar to the ones already registered, which will improve segmentation. If unimodality is not detected for both Cr and Cb histograms then the original 2d colour signature is used instead.

Then the back projection is performed. The CrCb colour signature is used as a look up table in order to segment the input image.

Figure 11 demonstrates back projection. The CrCb signature that is generated from the colour information in the ROI, indicated in green on the left image, is used as a look-up table to segment that same image. Every pixel is evaluated of the input image, and if its Cr and Cb values correspond to a CrCb value pair present in the colour signature that pixel is labeled as 1 in the output back projection, and as 0 otherwise.

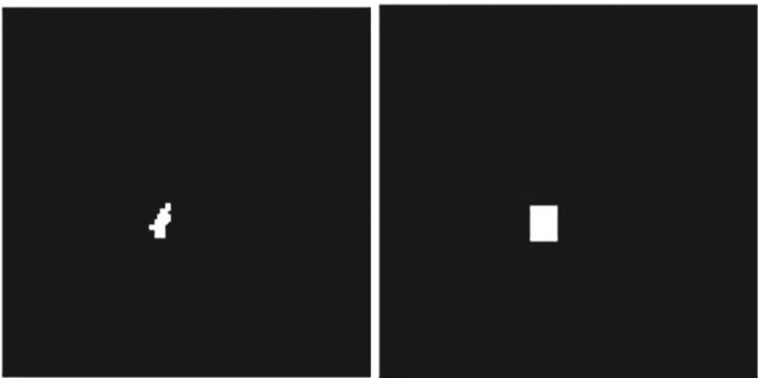

Fig. 10 Left: Original 2d CrCb histogram. Right: With box filling in missing colour information for the colour mode

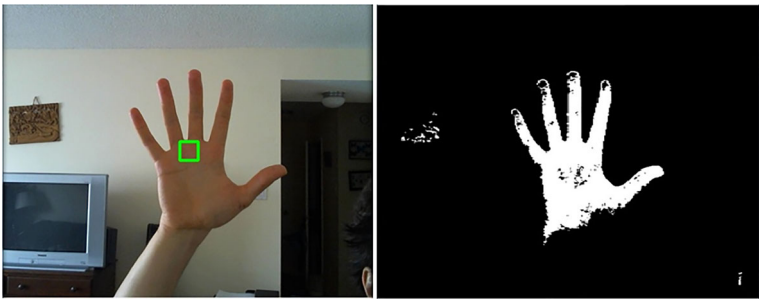

Fig. 11 Performing Backprojection

Unimodal filtering fixes problem areas in the segmentation. The colour sampling often under represents the range of colours of a user's hand. This results in missing areas of the user's hand contour such as fingers or pieces of the palm. This is due to colour pairs being missing in the CrCb colour signature (Fig. 12).

The problem areas are highlighted in red in the above back projection. The areas which likely contain useful colour information in the histogram are also highlighted. Unimodal filtering fills in this missing colour information, and it only triggers when the colours sampled form a unimodal distribution. Using the filtering produces results like this (Fig. 13):

The first back projection is the original, and the second back projection is done after unimodal histogram filtering is applied.

4.2 Affects of unimodal histogram filtering

The system was tested to determine how unimodal histogram filtering affects the performance of the validation functions. The affects of Unimodal filtering are summarized in Table 2.

Unimodal filtering has an overall positive effect on both systems, particularly the hand 5 gesture. It also has positive effects on the hand L gesture, with only a negligible drop in true positive rate for the Cascade configuration being the only trade off for an improved false positive rate. The hand I gesture performance is improved for the SVM configuration, but only negligibly improved for the Cascade configuration. The negligible improvement in false positive rate comes at over a 5% drop in true positive rate. Because the false positive rate was already quite low for the hand I gesture the unimodal filtering wasn't used for the hand I validation function of the Cascade system. It was only used for the hand 5 and hand L gestures

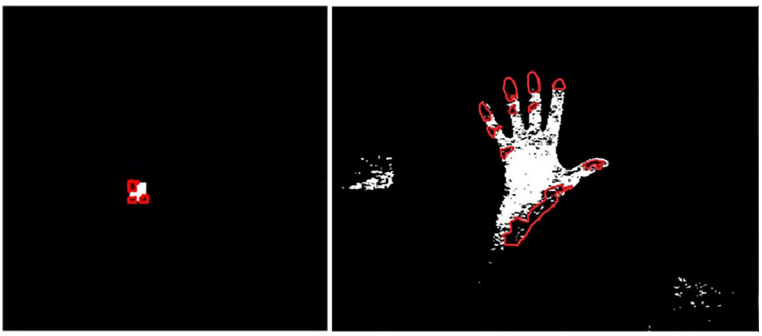

Fig. 12 CrCb histogram (left) and corresponding back projection (right)

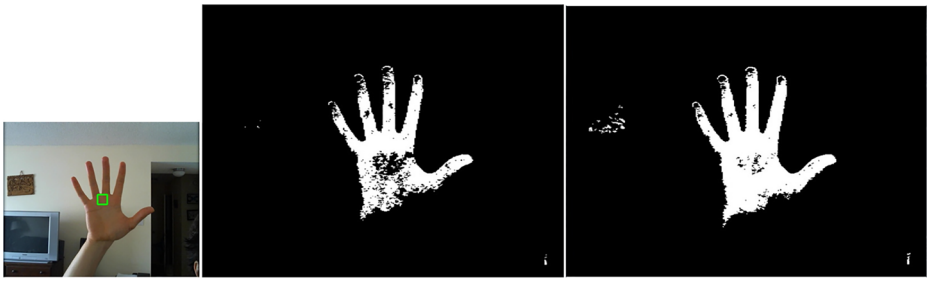

Fig. 13 Improved backprojection with unimodal filtering

for the HOG Cascade Gesture Recognition system. It was used for all 3 gestures in the SVM configuration.

4.3 The open hand model

The Open Hand model is a series of geometric tests that are applied to a given contour shape to determine if it is a hand 5 gesture. It takes a sample point and a back projection as input and outputs whether a contour was found that has a valid hand 5 gesture shape. The hand 5 gesture is an open hand with 5 visible extend fingers. Due to space constraints the details of the model equations will be omitted here. However the full equations of the model are available at [67].

In order to be valid a given contour must contain the sample point. The first step is to find the contour's palm. The biggest convexity defect is found and then the model discards all other convexity defects whose depth is less than 6.6% of the biggest defect depth. A potential hand 5 shape must have between 3 and 8 kept convexity defects. The inner points of these defects are used to find a minimum enclosing circle which is declared to be the palm (Fig. 14).

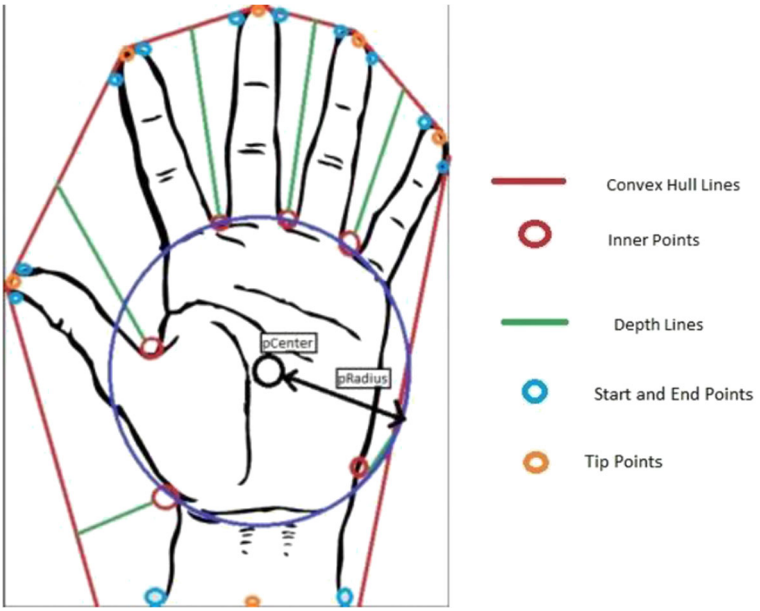

Fig. 14 Open Hand Model Palm Location

The tip points are found by averaging the start and end points of adjacent convexity defects. The angles of each tip point to its corresponding inner points are also measured (Fig. 15).

The model then goes through each pair of adjacent convexity defects and their corresponding tip points to determine if the contour protrusion that they represent corresponds to an extended finger (Fig. 16).

The finger basepoint is halfway between two adjacent inner points. The shifted finger base point is the point on the palm circle which has the same angle to the palm center as the finger base point. The knuckle point is halfway between the tip point and the shifted finger base point.

All of the relationships that are presented are evaluated with the model equations to determine if a given tip point represents a valid extended visible finger. This enforces the angle and proportion limits of the model.

Once the valid finger points of a contour have been found the number of consecutive finger points are found. Consecutive finger points are adjacent valid finger points that form an angle between themselves and their intermediate inner point of 45 degrees of less. A contour shape is declared to be a hand 5 shape if it has either 4 or 5 valid finger points or at least 3 consecutive finger points. These conditions are designed to ensure quick recognition in the presence of noise. If a contour shape is found to be a hand 5 shape then the open hand model returns true.

4.4 The general hand model

The general hand contour model is a series of geometric tests that is used for validating the hand L and hand I gestures. The model takes a sample point and a back projection as input and outputs the visible extended finger count of a valid hand contour if one is found. Due to space constraints the details of the model equations will be omitted here. The full models equations are available at [67].

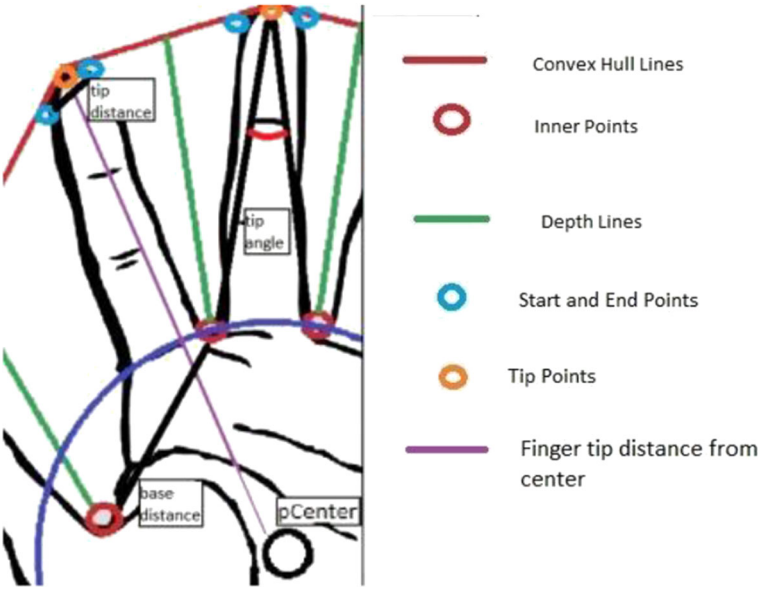

Fig. 15 Open Hand Model Finger Processing (1)

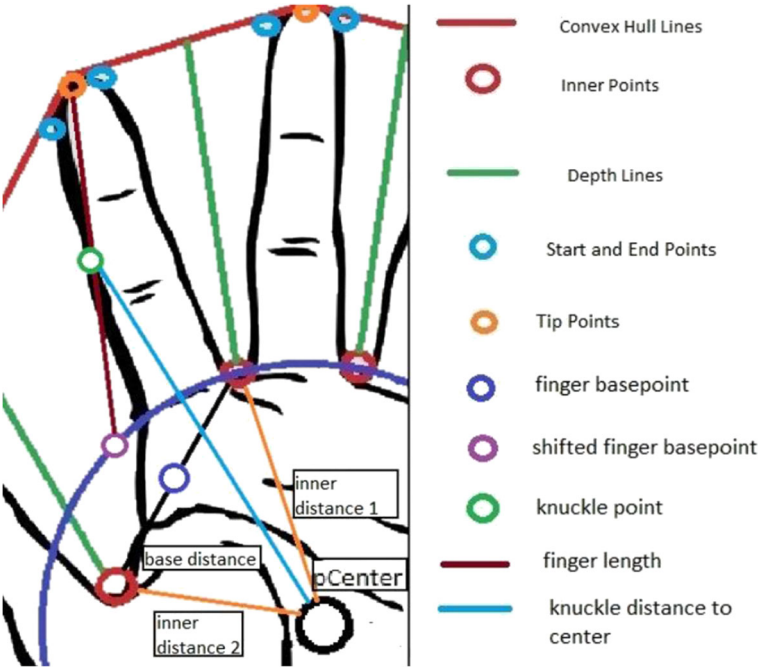

Fig. 16 Open Hand Model Finger Processing (2)

A valid hand contour must contain the sample point. A valid contour must also be at least 120 points in size. (Our system runs on 640 × 480 frames). If no valid contour is found then the model terminates evaluation. The convexity defects of the contour are filtered in the same way as in the open hand model but defects with defect depths of less than 10.0% of the biggest defect depth are discarded. Any number of kept convexity defects is valid.

In the general hand model the palm is determined more precisely. The initial inner points are filtered to find a better localized palm. This overcomes the effects of forearms and biceps on the hand contours. The inner points of the kept convexity defects are used to find a minimal enclosing circle. The radius of this circle is referred to as the current radius. The center is referred to as the minimal enclosing circle center. The average of all the inner points is calculated. Each inner point has a distance to the average of all inner points and a distance to the center of the minimal enclosing circle (test distance). The general idea is that an inner point is declared to be a palm point if it is closer to the average of all the inner points than the center of the minimal enclosing circle (Fig. 17).

Palm points must also form a general circular shape. The angle between 3 consecutive palm points must greater than 30° otherwise the palm point located at the angle is discarded. If there are at least 5 palm points remaining than these palm points and their minimum enclosing circle are the contour's palm. Otherwise the original inner points of the kept convexity defects are used as the palm instead.

Next the generalized hand contour model determines if there are any visible extended fingers. The model counts the number of extended fingers in a similar manner to the open hand model. Further details of this will be omitted for brevity.

If the generalized hand model is successful in finding a valid contour, it will output the visible extended finger count. It can also give the contour itself, as well as palm and fingertip locations.

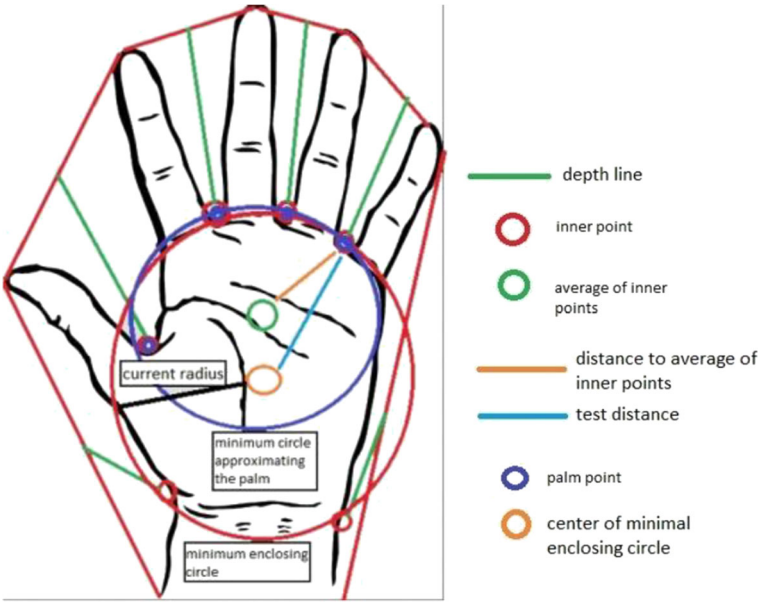

Fig. 17 General Hand Model Palm Location (1)

4.5 Shape signature analysis

Shape signature analysis is a secondary finger counting method used to count the extend number of visible fingers. It verifies the number found with the general hand model. This method takes as input the hand contour found by the general hand model, and outputs a finger count.

The center of mass of the hand contour is found by using the distance transform. Then PCA is used to find the principle axis. Then the maximum inscribed circle from the center of mass to the hand contour is found (Fig. 18). Using the center of mass and the principle axis, the contour's shape signature is plotted. Distance from the center of mass is normalized by the radius.

A threshold of 1.5 is applied. Cuts are made when large vertically oriented convexity defects are found in the thresholded finger contours. For a defect to be considered large it needs to have

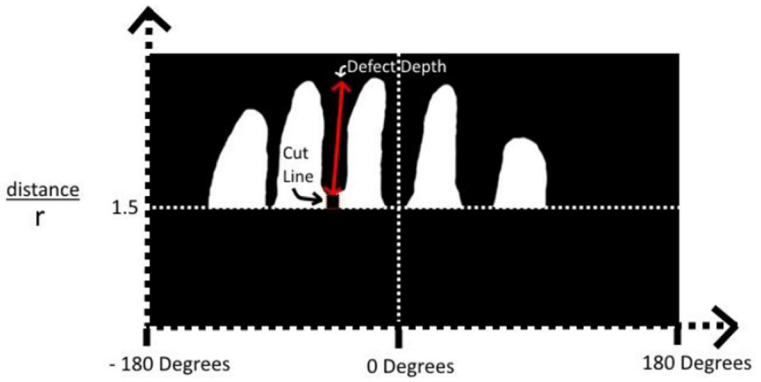

Fig. 18 Separating finger contours

a y-value difference that is equal or greater than 70% of the largest y-value difference of the corresponding contour from which it was extracted. A cut is then made from the convexity defect's inner depth point to separate the contour. This allows finger contours that are still connected to be separated. The resulting finger contours are counted, and this result is compared to the number of fingers a given hand gesture must have, and the number of fingers found by the general hand model. If all three values match, the hand gesture is declared valid, otherwise it is invalid. Further details regarding shape signature analysis can be found at [67].

5 Results

5.1 Gesture recognition demonstration

The Gesture Recognition system was implemented with both a HOG Cascade detection step, and a HOG SVM detection step. Here are some recognition result images of the Cascade configuration system (Fig. 19).

Here are some gesture recognition results of the Gesture Recognition system with SVM detectors (Fig. 20). The images show the filtering of the bounding boxes in the two stage system.

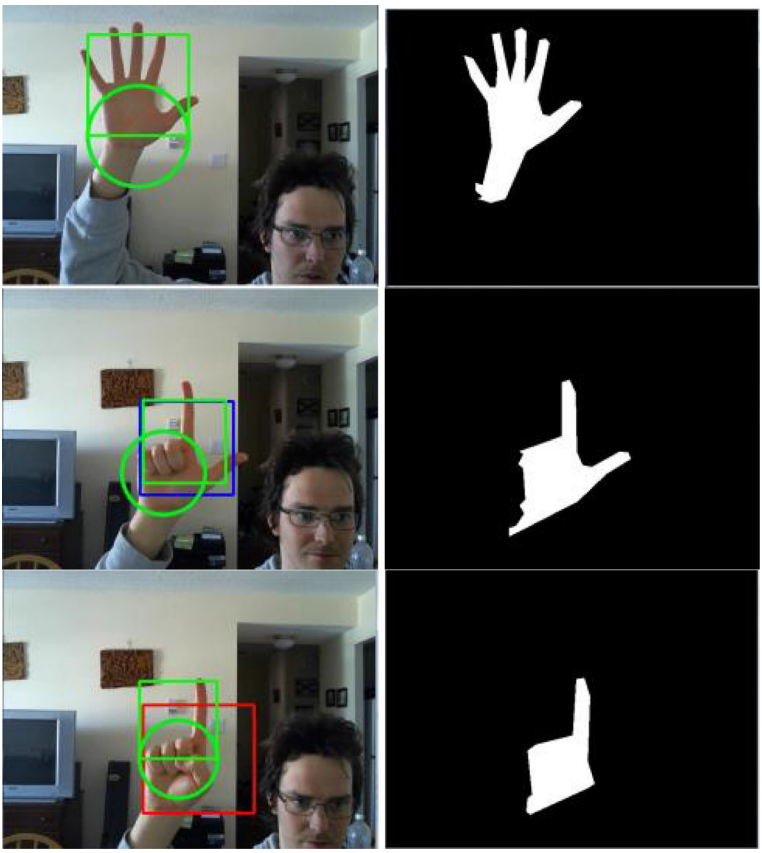

Fig. 19 Examples of Gesture Recognition results achieved with HOG Cascade System

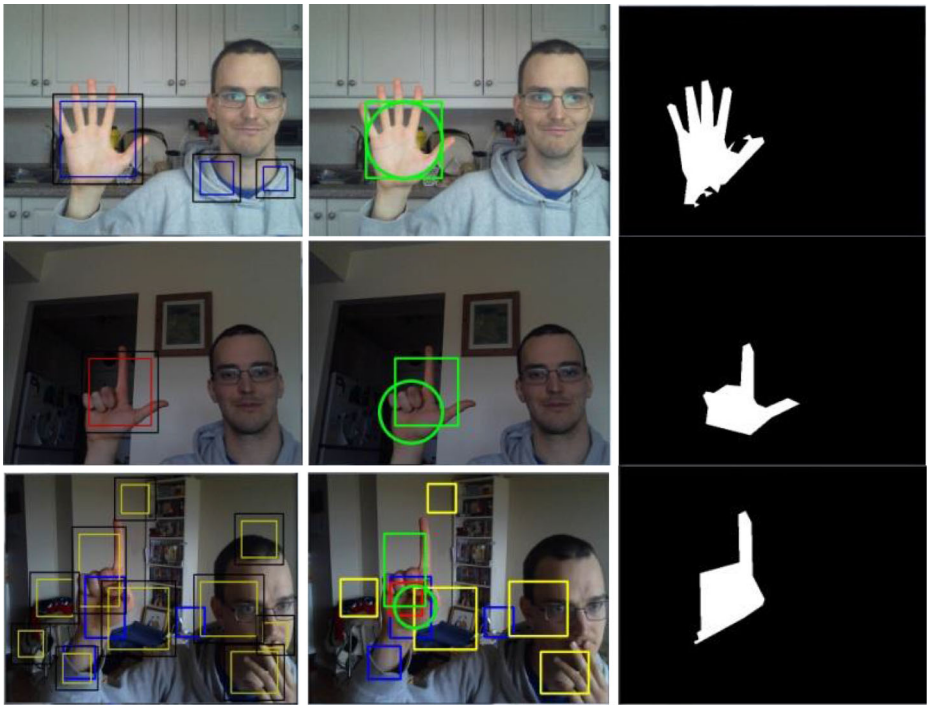

Fig. 20 Examples of Gesture Recognition results achieved with HOG SVM System

A video demonstration of the HOG Cascade GR system can be found here [64] and a demonstration of the SVM system can be found here [65]. The systems are responsive enough and discriminative enough for UI applications.

5.2 Gesture recognition testing

The HOG Cascade GR system and the HOG SVM GR system were tested using the test dataset (Table 3). This dataset is comprised of 2024 hand 5 samples, 1638 hand L samples, 1791 hand I samples, and 8824 full frames of backgrounds. The gesture samples were cropped images. Testing was done by randomly placing them over top of a background image. The true positive and false positive results for HOG Cascade and HOG SVM Gesture Recognition systems are presented in Table 4.

A rough estimate of the execution time of the experiment with the Cascade system was 1 h 44 min. The SVM system testing took approximately 4 h 50 min. It can be seen that the SVM system is indeed more computationally costly, but its speed can be improved with optimization.

Both of the configurations can run at real-time or close to real time. The Cascade configuration can run at real-time, and the SVM configuration can be used as a real-time system because it only has a minor delay in terms of responsiveness when it runs online with a web camera. The runtime of the SVM system's experiment on the testing dataset is roughly 3 times longer than the Cascade based system. Both of these systems can be used as real-time systems, even though the SVM configuration has a delay in responsiveness. This has been achieved on a Intel Core i7 laptop computer without a dedicated effort for optimization.

Table 3 Design-of-Experiment chart for Gesture Recognition system testing

Controlled Variables	Uncontrolled Variables	Process	Measured Variables
- Class of images. 3 gesture classes and 1 background class are used. - Which gesture class or background class gets displayed - Conditions under which a gesture image gets displayed: Generic office type background. No faces or users present, just the background and the gesture.	- Where a gesture image is placed onto the generic background. This is done with deterministic random number generation, and thus replicable. - When faces or people appear in the background class images. There are many samples of the background class which contain faces or people, this is not known or controlled during runtime.	1. Generate the sample. For the gesture class samples this means taking the image and placing it randomly onto the generic office background image. For background class samples no further processing is needed. 2. Process the sample with the evaluated gesture recognition system. 3. Record the result and compare to the known label.	- What kind of detection is produced, which gesture class is detected if a gesture is detected, otherwise a given sample is classified as a background class sample.

Therefore it is a safe assumption that if optimized the SVM system would run in real-time as well.

During testing on a few occasions a gesture would be validated using a contour of an incorrect object, but because the gesture class would correspond to the correct gesture class of the sample image, it would register as a true positive. All systems were tested using the same testing method so the performance comparisons made are still valid, and this event did not occur often. This is however something that can be improved in future work.

The criteria for success in this project is that a hand gesture should be detected after being visibly presented in front of a camera for a couple of seconds, and the false positive rate should be very low. These two criteria need to be achieved for a robust system. Narrowing recognition time to approximately 1 s means that in a real-time system running at 30 frames per second, at least 1 frame with a hand gesture out of 30 should be detected to meet this requirement. Which corresponds to a true positive rate of 3.3%. This true positive rate has been achieved for each gesture in each configuration. A low false positive rate is also important for a robust system. While the Cascade system outperforms the SVM system, both systems achieve very low false positive rates.

Table 4 Confidence Threshold vs. Performance for MobileNets. The best false positive rate with a true positive rate of 3.3% or greater is shown in bold

Confidence Threshold vs. Gesture Recognition Performance for MobileNets Comparison system						
Threshold	hand 5 TP %	hand 5 FP %	hand L TP %	hand L FP %	hand I TP %	hand I FP %
0.95	95.50395	6.285899	62.02686	13.58751	94.8074	14.95156
0.96	94.66403	4.400938	57.57021	10.57504	93.8024	11.75163
0.97	93.92292	2.742497	52.1978	7.374284	92.9648	9.012228
0.98	92.53953	1.577542	43.58974	4.157841	90.3964	6.106082
0.99	89.08103	0.776636	**29.9145**	**1.4984**	85.0363	3.144354
0.999	71.39328	0.121349	2.197802	0.007845	47.2362	0.317612
0.9995	**62.3024**	**0.0485**	0.671551	0	**34.618**	**0.1588**

5.3 Comparison to MobileNets

The performance of the system was compared to a popular neural network for mobile real-time applications proposed by Google called MobileNets [25].

The a comparison system was made that used a MobileNets V2 model with 128 × 128 window size for multiclass gesture classification along with a simple sliding window approach. Because the HOG SVM and the HOG Cascade detectors both use this approach this allows for a very good comparison to be made. Testing of the MobileNets system was done using the same test dataset in the same manner as the Cascade and SVM systems (Table 4).

The MobileNets model was trained with all of the hand 5, hand L, and hand I samples of the training dataset which is 7813, 6401, and 6958, respectively. It was also trained with 10,896 randomly selected negative patches of the approximately 250,000 negative background patches generated from the negative backgrounds of the training dataset. This was done in order to achieve approximately a 1.5:2 ratio with the hand 5 samples. This way some class balance can be achieved, while still benefiting from the large quantity of negative background images in the dataset.

The results of the MobileNets system are presented in Table 4 under *MobileNets 0.99 Best Global Threshold System*. The execution time of the MobileNets system on the test dataset was roughly 90 h. The experiment was run on the same Intel Core i7 laptop computer as the other systems. The machine has a Nvidia GTX950M graphics card which has a 5.0 CUDA compute compatibility. The MobileNets model outputs a confidence threshold that a certain evaluated image sample is of a given class. The confidence threshold is an important parameter that affects the performance of the system. A comparison was made of the performance using different confidence thresholds (Fig. 21).

0.99 was deemed to be the best confidence threshold. It achieved a low false positive rate while maintaining high recognition performance. Increasing the threshold beyond 0.99 saw a dramatic decrease in true positive rate particularly for the hand L gesture.

The results of all three systems can be seen in the following comparison table. MobileNets is compared to both systems using the same false positive rate as each gesture of the Cascade and SVM systems. A corresponding confidence threshold necessary for MobileNets to achieve this performance is shown as well (Table 5).

The Cascade and SVM Gesture recognition systems both achieve a generally better performance for false positive rate than the best global confidence threshold MobileNets system. Both systems outperform the global threshold MobileNets system with lower false

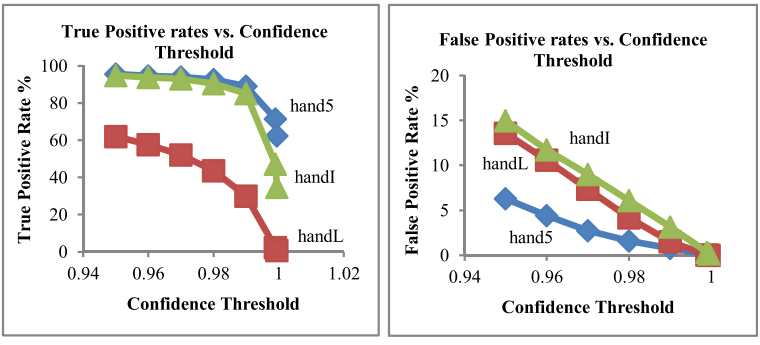

Fig. 21 MobileNets Comparison System True Positive and False Positive result graphs. The MobileNets Comparison system has an mAP of 0.748

Table 5 Comparison table for Cascade and SVM systems to MobileNets

	Cascade System		MobileNets Best Individual Thresholds System			MobileNets 0.99 Best Global Threshold System	
Gesture	% TP	% FP	% TP	% FP	Confidence Threshold Necessary	% TP	% FP
hand 5	31.67	0.04854	**62.3**	0.04854	0.9995	89.08	0.7766
hand L	**25.58**	0.8943	18.68	0.8943	0.9936	29.91	1.498
hand I	16.19	1.008	**56.47**	1.008	0.9968	85.04	3.144
mean Precision	82.41		**87.08**			82.09	
mean Recall	24.48		45.82			**68.01**	
mean Accuracy	89.92		92.77			**94.62**	
	SVM System		**MobileNets Best Individual Thresholds System**			**MobileNets 0.99 Best Global Threshold System**	
Gesture	% TP	% FP	% TP	% FP	Confidence Threshold Necessary	% TP	% FP
hand 5	26.33	0.07281	**65.33**	0.07281	0.9993	89.08	0.7766
hand L	20.51	1.828	**31.61**	1.828	0.9888	29.91	1.498
hand I	10.11	2.898	**81.74**	2.898	0.9908	85.04	3.144
mean Precision	63.51		**82.78**			82.09	
mean Recall	21.69		59.56			**68.01**	
mean Accuracy	89.73		93.61			**94.62**	
					Cascade System	**SVM System**	**MobileNets System**
Approximate total runtime on test dataset					**1 h 44 min**	4 h 50 min	94 h 4 min
Approximate average runtime per frame					**434 ms**	1210 ms	23,541 ms

positive rates for hand 5 and hand I gestures. The HOG Cascade system even beats MobileNets for the hand L gesture, while the SVM system trails by 0.3%. In addition in order to achieve the same false positive rate as the Cascade system the MobileNets system must use a confidence threshold of 0.9995 which reduces the true positive rate of the hand L gesture bellow the required 3.3% for a real-time system. If different confidence thresholds were used for each gesture, the MobileNets system beats the Cascade system in terms of Hand 5 and Hand I performance, but it is not be able to best the Hand L gesture performance. Linear Interpolation reveals that in order to match the false positive rate of the hand L gesture the MobileNets system must have a confidence threshold of 0.993647 which achieves a lower true positive rate of 18.68% than the 25.58% achieved by the HOG Cascade Gesture Recognition system.

A major aspect to note is that the run time of the MobileNets system is significantly greater than that of the HOG Cascade and HOG SVM systems while using the same simple sliding window detection strategy. This highlights the importance of using a proposal generator in order to make a viable real-time system with a MobileNets neural network.

Low false positive rates were achieved with both systems for all 3 gestures. These rates can be achieved with high performing CNNs only when using very high confidence thresholds. Furthermore true positive rates were achieved with both systems that are significantly higher than the 3.3% required in order to have responsiveness in real-time contexts. The Cascade

system is able to beat MobileNets outright in some cases. A well performing system is achieved for the real-time context at a fraction of the computational cost of MobileNets. Both Cascade and SVM configurations require much less computation time to processes the test dataset.

To summarize, the SVM gesture recognition system is a very close contender to .99 threshold MobileNets in terms of performance as a real-time system because of its low false positive rates. The Adaboost Cascade system beats the .99 threshold MobileNets system outright. Using separate confidence thresholds would allow MobileNets to outperform the Cascade system for Hand 5 and Hand I gesture recognition but not for Hand L. In addition using MobileNets requires a proposal generation architecture in order to run as a real-time system, which would also perform well so long as the necessary architecture is in place. However both Cascade and SVM systems are computationally much lighter than MobileNets which means that there is trade-off between speed and classification performance.

5.4 Usability as a UI using hand tracking

The Gesture Recognition system in the Cascade configuration has been successfully combined with hand tracking (Fig. 22). The system is used to initialize a hand tracking algorithm that is based on the general hand model [66]. The hand tracking algorithm can also operate on its own from fixed location initialization [61].

Hand tracking is initiated using the hand 5 gesture and proceeds to track the hand in subsequent frames until it can no longer locate the hand. Segmentation of the input frame is done with the same back projection technique as for the gesture recognition using the retained colour signature. The tracking algorithm uses two collaborative tracking strategies. Tracking of the hand is done by analyzing the contours with the General Hand Model and performing a model sensitive nearest neighbour search between successive frames. Tracking of the fingertips is done using template matching. The fingertip template trackers are guided by the contour based tracking which delimits smaller search regions for the trackers which allows the 2-strategy tracking algorithm to run in real-time. The template tracking in turn provides stable fingertip tracking, more stable that what the contour tracking can provide. The stable fingertip tracking allows the algorithm to be used to make responsive UI. Demonstrations of how the hand tracking algorithm can be used for UI applications can be found at [62], which is a Paint application for Android, and at [63], which is a controller for an FPS video game called Halo: Combat Evolved.

5.5 Hand tracking testing

The Gesture Recognition system with Hand Tracking was tested using a challenging dataset of 2D videos where users presented a hand 5 gesture for 20–30 s before displaying a sequence of extended visible fingers (Table 6). The users were free to move anywhere they wished provided that they did not occlude their faces with their hands, and the hand remained in the frame while presenting the sequence. They were also free to present the specified finger counts in any manner that they wished. The videos were taken in indoor laboratory settings with complex backgrounds. The dataset was annotated frame by frame with the user's hand location, if present, being outlined with a bounding box, and the number of visible extended fingers was recorded.

Performance was measured by the algorithm's abilities to track the hand within the bounding box and by the ability of the contour tracking and the template tracking to count

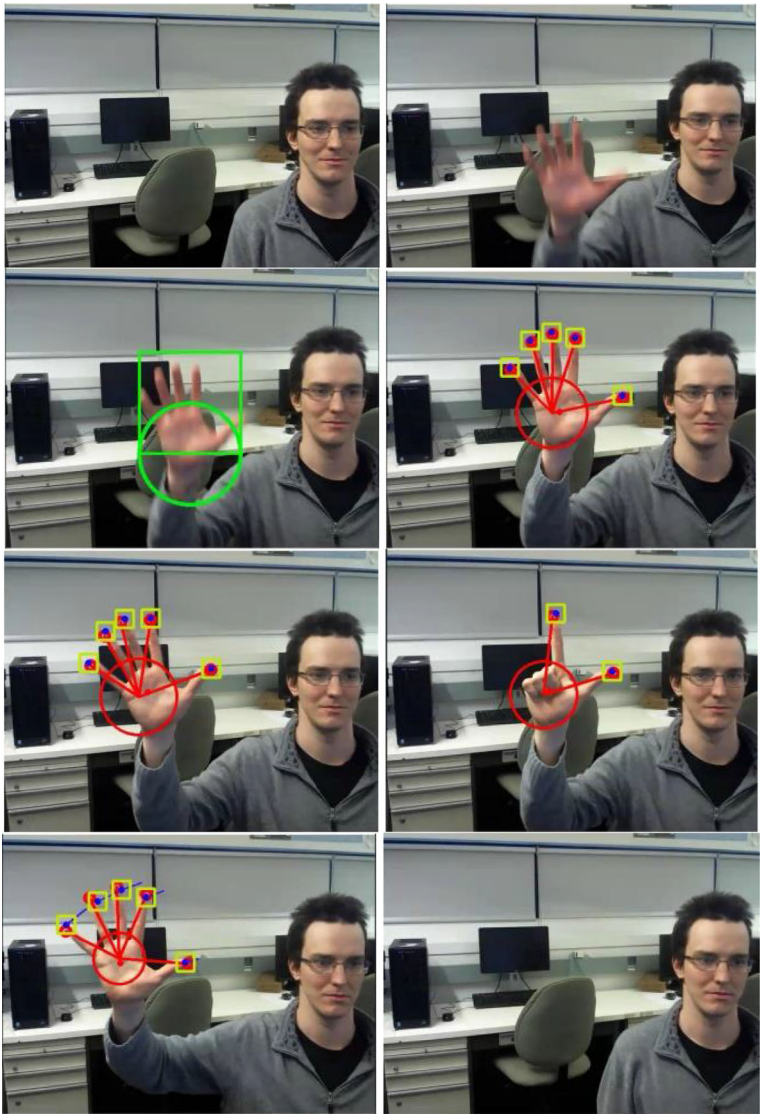

Fig. 22 Example of operation of GR with Hand Tracking system

the correct number of extended fingers. The system is not given any initial location on which to commence tracking, and must rely on its recognition capabilities to find the initial hand location. This evaluates unconstrained tracking performance (Table 7).

The Gesture Recognition with Hand Tracking system achieved good tracking results in a challenging unconstrained setting achieving an 89% success rate. Instances where tracking was not successful could be due to tracking failing during a given video and it taking some time to re-locate the hand and continue tracking, or due to the response to the initial hand registration and tracking being delayed due to difficulty of recognition in a complex environment, or due to the algorithm tracking the wrong object and the tracking target was outside the

Table 6 Design-of-Experiment chart for Gesture Recognition and Hand Tracking testing

Controlled/Known Variables	Uncontrolled Variables	Process	Measured Variables
- Whether or not a hand is present in a given frame. - How many fingers are displayed if a hand is present in a frame. - Where a hand is if it is present in a frame. - The general script of the videos: 30 s of the hand-5 gesture displayed anywhere on the screen, followed by a sequence of displayed extended fingers for each user. The user can move their hand freely provided that the motion is smooth and the hand doesn't occlude with the user's face.	- Whether the user shows their face in a given frame. - Whether or not there are other people in the background. Other people are present in the background in some instances in the dataset.	1. Load a given video from the dataset. 2. Load a given frame from the video 3. Process the frame with the Hand Gesture Recognition and Hand Tracking system. 4. Record if a tracked hand was found within the known location bounding box. 5. Record how many extended fingers were found. 6. Repeat steps 2–5 for all frames of the video in consecutive order. 7. Repeat the process for all videos of the dataset.	- If a hand is successfully tracked in the correct location within a frame. - If the correct number of extended fingers were detected.

area of the known target as per the frame by frame annotation. 89% percent is never the less a very good result for unconstrained tracking in a complex environment. The finger counting success rate of the two tracking strategies may seem low, however it must be remembered that

Table 7 Performance of the Gesture Recognition and Hand Tracking System

Video Number	Successful Hand Tracking	Successful Finger Counting with Contour Tracking	Successful Finger Counting with Template Tracking
1	0.9663	0.7243	0.6753
2	0.7527	0.6081	0.5194
3	0.9880	0.7954	0.5797
4	0.9777	0.7984	0.6706
5	0.9835	0.7102	0.5741
6	0.9753	0.8571	0.8519
7	0.9819	0.7418	0.6698
8	0.9795	0.7855	0.6188
9	0.9949	0.7237	0.6822
10	0.7417	0.4466	0.3883
11	0.4583	0.3031	0.3360
12	0.8824	0.7093	0.5824
13	0.9020	0.6544	0.7389
14	0.8874	0.6941	0.6558
15	0.9531	0.7685	0.6960
16	0.9500	0.4406	0.3828
17	0.7718	0.6328	0.6025
18	0.8003	0.3494	0.4138
19	0.9777	0.6610	0.6918
Average	0.8908	0.6529	0.5963

noise in the contour boundary can add false fingers or eliminate fingers from the count which in turn leads to a false result for a given frame. The remaining fingertips can still be useful for issuing UI commands. Although noise reduction is an excellent area for future work, this tracking system still achieves excellent results which enable UI applications.

The affect of motion blur in the user's hand on the initial registration of the hand was also investigated. The experiment was run a second time with the system only becoming active after 15 frames of the hand 5 gesture had been present, with the assumption that the user's hand would be relatively stationary at this point and blur would be minimal. As it can be seen in the following table this had the affect of minor improvement across all three testing criteria which demonstrates that motion blur is an important consideration in the constrained context of 2D video only Gesture Recognition and Hand Tracking (Table 8).

The resulting video [66] shows how the Gesture Recognition system can initiate and recover the hand tracking and how this tracking is robust and real-time. The system runs in real-time on a CPU without a dedicated optimization effort. The hand tracking algorithm has also been used in a variety of applications including [62, 63]. The combined Hand Gesture Recognition and Tracking system demonstrates the usability of the algorithm for UI, bringing robust hand gesture recognition and tracking in a 2D video only real-time setting.

5.6 Datasets

The complete dataset used for training and testing the Gesture Recognition component of the system is available at [68]. It contains 5 gesture classes with 8000–10,000 samples per class. This allows for comparison between methods requiring large datasets and those which only require small datasets. Because there are currently no large publicly available datasets for 2d vision only hand gesture recognition this will be an excellent opportunity for comparison work. Existing datasets of gestures for 2d vision are often too small [19, 20, 32, 36, 60, 92], publically unavailable [6, 22, 23, 26, 47, 56, 89, 95, 98], or are only designed for hand detection and not gesture recognition [77]. The datasets in [77] lack pose annotations for different classes and cannot be used without further annotation.

The dataset used for testing the hand tracking component of the extended system is available at [69]. This dataset has 19 annotated videos from 5 different users. The dataset corresponds to roughly 20,000 sequentially annotated frames indicating the general location of a user's hand and the amount of visible extended fingertips. This dataset will be a welcome addition to the research community for comparative work, especially since there are very few such datasets available for the context of hand tracking in 2D video.

Table 8 Performance of the Gesture Recognition and Hand Tracking System

Experiment	Description	Successful Hand Tracking	Successful Finger Counting with Contour Tracking	Successful Finger Counting with Template Tracking
1	GR with Hand Tracking System	0.8908	0.6529	0.5963
2	GR with Hand Tracking System after 15 frames of hand-5 gesture	0.9001	0.6637	0.6144

6 Conclusion

A real-time static hand gesture recognition system was presented for user interface applications. The system achieves great low positive results comparable to high performing CNNs, with true positive results that are more than sufficient for use in live real-time applications. The system achieves these results at a fraction of the computational cost of high performing CNNs in a CPU only 2D video context without depending on GPU-acceleration. The system was extended to include robust hand tracking and a variety of prototype user interface applications are presented to prove the usability of the system as a UI. We believe this system is an important milestone in 2D video only static hand gesture recognition and tracking for low cost hardware. The proven UI applications demonstrate its practicality for user control input, especially with low cost hardware, which can be investigated in future work. Furthermore the two datasets that were produced represent a major contribution to the state of the art in an area that has thus far had a lack of publically available datasets and comparative testing methods. The publically available datasets will allow for comparative work to be done for 2D video only static hand gesture recognition and hand tracking thus enabling further progress and innovation in the research area. Future work can involve improving the true positive rate for even more responsiveness, adding more gestures to the system, and exploring new applications with hand tracking.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11042-022-12870-8.

Acknowledgements The authors would like to thank the Natural Sciences and Engineering Research Council, Mitacs, and Ontario Graduate Scholarships, for their support in funding this project.

Funding Open access funding provided by National Research Council Canada.

References

1. Adithya V, Rajesh R (2020) A deep convolutional neural network approach for static hand gesture recognition. Procedia Comput. Sci. 171(2019):2353–2361
2. Alon J, Athitsos V, Yuan SQ, Sclaroff S (2009) A unified framework for gesture recognition and spatiotemporal gesture segmentation. Pattern Anal Mach Intell IEEE Trans 31(9):1685–1699
3. Bernardis P, Gentilucci M (2006) Speech and gesture share the same communication system. Neuropsychologia 44(2):178–190
4. Bhowmick S, Kumar S, Kumar A (2015) Hand gesture recognition of English alphabets using artificial neural network. 2015 IEEE 2nd Int. Conf. Recent Trends Inf. Syst., pp. 405–410
5. Chen Y, Zhang W (2016) Research and Implementation of Sign Language Recognition Method Based on Kinect, pp 1947–1951

6. Chen L-M, Hsieh CT, Hung K-M, Yeh C-H, Ke C-Y (2013) A real time hand gesture recognition system based on DFT and SVM. In: Inf. Sci. Digit. Content Technol. (ICIDT), 2012 8th Int. Conf., vol. 284–287, pp 490–494
7. Chernyshov V, Mestetskiy L (2016) Real-time hand detection using continuous skeletons. Pattern Recognit. Image Anal. 26(2):368–373
8. Chochai P, Mekrungroj T, Matsumaru T (2014) Real-time gesture recognition with finger naming by RGB camera and IR depth sensor. In: 2014 IEEE Int. Conf. Robot. Biomimetics, IEEE ROBIO 2014, pp 931–936
9. Ciuffani BM (2017) Non-verbal Communication and Leadership The impact of hand gestures used by leaders on follower job satisfaction. In: 9th IBA Bachelor Thesis Conference
10. Cook SW (2018) Enhancing learning with hand gestures: Potential mechanisms, 1st ed., vol. 69. Elsevier Inc
11. Costanzo C, Iannizzotto G, La Rosa F (2003) VirtualBoard: real-time visual gesture recognition for natural human-computer interaction. Proc. Int. Parallel Distrib. Process. Symp. 00, no. C:8
12. Crisnapati PN, Setiawan M, Wikranta Arsa IGN, Devi Novayanti P, Wibawa MS, Oka Ciptahadi KG (2019) Real-time hand palm detection and tracking augmented reality game using lucas kanade optical flow combined with color blob detection. In: 2019 1st Int. Conf. Cybern. Intell. Syst. ICORIS 2019, no. August, pp 263–268
13. Dagnes N, Vezzetti E, Marcolin F, Tornincasa S (2018) Occlusion detection and restoration techniques for 3D face recognition: a literature review. Mach Vis Appl 29(5):789–813
14. Dardas NH, Georganas ND (2011) Real-time hand gesture detection and recognition using bag-of-features and support vector machine techniques. Instrum Meas IEEE Trans 60(11):3592–3607
15. Dollár P, Welinder P, Perona P (2010) Cascaded pose regression. Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit: 1078–1085
16. Fan X, Jia Q, Huyan K, Gu X, Luo Z (2016) 3D facial landmark localization using texture regression via conformal mapping. Pattern Recognit Lett 83:395–402
17. Fu X, Zhang T, Bonair C, Coats ML, Lu J (2015) Wavelet enhanced image preprocessing and neural networks for hand gesture recognition. In: Proc. - 2015 IEEE Int. Conf. Smart City, SmartCity 2015, Held Jointly with 8th IEEE Int. Conf. Soc. Comput. Networking, Soc. 2015, 5th IEEE Int. Conf. Sustain. Comput. Communic, pp 838–843
18. Ghosh DK, Ari S (2011) A static hand gesture recognition algorithm using k-mean based radial basis function neural network. In: 2011 8th Int. Conf. Information, Commun. Signal Process., no. i, pp 1–5
19. Ghosh DK, Ari S (2015) Static hand gesture recognition using mixture of features and SVM classifier. In: 2015 Fifth Int. Conf. Commun. Syst. Netw. Technol, pp 1094–1099
20. Ghosh DK, Ari S (2016) On an algorithm for Vision-based hand gesture recognition. Signal, Image Video Process. 10(4):655–662
21. Goldin-Meadow S (1999) The role of gesture in communication and thinking. Trends Cogn Sci 3(11):419–429
22. Guo Y, Cheng J, Pang J, Guo J (2013) Real-time hand detection based on multi-stage HOG-SVM classifier. In: 20th IEEE International Conference on Image Processing (ICIP), pp 4108–4111
23. Han M, Chen J, Li L, Chang Y (2016) Visual hand gesture recognition with convolution neural network. In: 2016 IEEE/ACIS 17th Int. Conf. Softw. Eng. Artif. Intell. Netw. Parallel/Distributed Comput. SNPD 2016, pp 287–291
24. Hasan H, Abdul-Kareem S (2014) Static hand gesture recognition using neural networks. Artif Intell Rev 41(2):147–181
25. Howard AG et al (2017) MobileNets: efficient convolutional neural networks for mobile vision applications
26. Hsieh C-C, Liou D-H (2015) Novel Haar features for real- time hand gesture recognition using SVM. J Real-Time Image Process 10(2):357–370
27. Hu K, Yin L, Wang T (2019) Temporal interframe pattern analysis for static and dynamic hand gesture recognition. Proc. - Int. Conf. Image Process. ICIP 2019-Septe:3422–3426
28. Huang DA, Ma M, Ma WC, Kitani KM (2015) How do we use our hands? Discovering a diverse set of common grasps. Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit 07-12-June:666–675
29. Hwang CL, Lee HW (2011) The command control by hand gesture with Hu and contour sequence moments and probability neural network. Conf Proc - IEEE Int Conf Syst Man Cybern:2056–2061
30. Inmoonnoy V, Ketcham M (2017) The message notification for patients care system using hand gestures recognition, no. 2
31. Islam MZ, Hossain MS, Ul Islam R, Andersson K (2019) Static hand gesture recognition using convolutional neural network with data augmentation. In: 2019 Jt. 8th Int. Conf. Informatics, Electron. Vision, ICIEV 2019 3rd Int. Conf. Imaging, Vis. Pattern Recognition, icIVPR 2019 with Int. Conf. Act. Behav. Comput. ABC 2019, pp 324–329

32. Jalab HA, Omer HK (2015) Human computer interface using hand gesture recognition based on neural network. In: Inf. Technol. Towar. New Smart World (NSITNSW), 2015 5th Natl. Symp, pp 1–6
33. Kapidis G, Poppe R, Van Dam E, Noldus L, Veltkamp R (2019) Egocentric hand track and object-based human action recognition. In: Proc. - 2019 IEEE SmartWorld, Ubiquitous Intell. Comput. Adv. Trust. Comput. Scalable Comput. Commun. Internet People Smart City Innov. SmartWorld/UIC/ATC/SCALCOM/IOP/SCI 2019, pp 922–929
34. Kerdvibulvech C (2015) Hand tracking by extending distance transform and hand model in real-time. Pattern Recognit. Image Anal. 25(3):437–441
35. Khamis S, Taylor J, Shotton J, Keskin C, Izadi S, Fitzgibbon A (2015) Learning an efficient model of hand shape variation from depth images. Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit 07-12-June: 2540–2548
36. Koller O, Ney H, Bowden R (2016) Deep hand: how to train a CNN on 1 million hand images when your data is continuous and weakly labelled. In: 2016 IEEE Conf. Comput. Vis. Pattern Recognit, pp 3793–3802
37. Krauss RM, Chen Y, Chawla P (1996) Nonverbal behavior and nonverbal communication: what do conversational hand gestures tell us? Adv Exp Soc Psychol 28, no. C:389–450
38. Lee J, Lee Y, Lee E, Hong S (2004) Hand region extraction and Gesture recognition from video stream with Complex Background Through Entropy Analysis. Hand, The, pp 1513–1516
39. Lekova AK, Dimitrova MI Hand gestures recognition based on lightweight evolving fuzzy clustering method. In: Image Information Processing (ICIIP), 2013 IEEE Second International Conference on, 2013, pp 505–510
40. Li H, Yang L, Wu X, Xu S, Wang Y (2012) Static hand gesture recognition based on HOG with kinect. In: Proc. 2012 4th Int. Conf. Intell. Human-Machine Syst. Cybern, vol 1. IHMSC 2012, pp 271–273
41. Li C, Zhang X, Jin L LPSNet: A novel log path signature feature based hand gesture recognition framework. In: Openaccess.Thecvf.Com, pp 631–639
42. Liu Y, Gan Z, Sun Y (2008) Static hand gesture recognition and its application based on Support Vector Machines, pp 517–521
43. Liu Z, Hu F, Luo D, Member I, Wu X, Senior I (2014) Visual gesture recognition for human robot interaction using dynamic movement primitives
44. Liu J, Furusawa K, Tateyama T, Iwamoto Y, Chen YW (2019) An improved hand gesture recognition with two-stage convolution neural networks using a hand color image and its pseudo-depth image. Proc. - Int. Conf. Image Process. ICIP 2019-Septe:375–379
45. Liu C, Li Z, Zhang C, Yan Y, Zhang R (2019) An Improved hand tracking algorithm for Chinese sign language recognition. In: Proc. 2019 IEEE 3rd Inf. Technol. Networking, Electron. Autom. Control Conf. ITNEC 2019, no. Itnec, pp 229–232
46. Mekala P, Fan J, Lai W-C, Hsue C-W (2013) Gesture recognition using neural networks based on HW/SW cosimulation platform. Adv Softw Eng 2013:2
47. Meng X, Lin J, Ding Y (2012) An extended HOG model: SCHOG for human hand detection. In: 2012 Int. Conf. Syst. Informatics, ICSAI 2012, no. Icsai, pp 2593–2596
48. Messom CH, Barczak AL (2009) Stream processing for fast and efficient rotated Haar-like features using rotated integral images. Int J Intell Syst Technol Appl 7(1):40
49. Molchanov P, Gupta S, Kim K, Kautz J (2015) Hand gesture recognition with 3D convolutional neural networks. In: 2015 IEEE Conf. Comput. Vis. Pattern Recognit. Work, pp 1–7
50. Mueller F et al (2018) GANerated hands for real-time 3D hand tracking from monocular RGB. Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit :49–59
51. Munib Q, Habeeb M, Takruri B, Al-Malik HA (2007) American sign language (ASL) recognition based on Hough transform and neural networks. Expert Syst Appl 32(1):24–37
52. Nguyen DD, Le HS (2016) Kinect Gesture Recognition: SVM vs. RVM. In: Proc. - 2015 IEEE Int. Conf. Knowl. Syst. Eng. KSE 2015, pp 395–400
53. Nguyen VT, Le TL, Tran TH, Mullot R, Courboulay V (2014) Hand posture recognition using Kernel Descriptor. Procedia Comput. Sci. 39, no. C:154–157
54. Nölker C, Ritter H (2002) Visual recognition of continuous hand postures. IEEE Trans. Neural Networks 13(4):983–994
55. Oberweger M, Wohlhart P, Lepetit V (2015) Hands deep in deep learning for hand pose estimation
56. Pang Y, Member S, Zhang K, Yuan Y, Member S, Wang K (2014) Distributed object detection with linear SVMs, vol. 44, no. 11, pp. 2122–2133
57. Panwar M (2012) Hand gesture recognition based on shape parameters. In: Computing, Communication and Applications (ICCCA), 2012 International Conference on, 2012, pp. 1–6
58. Park G, Argyros A, Lee J, Woo W (2020) 3D hand tracking in the presence of excessive motion blur. IEEE Trans. Vis. Comput. Graph. 26(5):1891–1901

59. Pisharady P, Vadakkepat P, Loh A (2013) Attention based detection and recognition of hand postures against complex backgrounds. Int J Comput Vis 101(3):403–419
60. Pisharady PK, Vadakkepat P, Loh AP (2013) Attention based detection and recognition of hand postures against complex backgrounds. Int J Comput Vis 101(3):403–419
61. Popov PA (2015a) Long hands gesture recognition system. [Online]. Available: https://www.youtube.com/watch?v=OcgXYQFNkIU
62. Popov PA (2015b) GR paint app presentation HD. [Online]. Available: https://www.youtube.com/watch?v=8e2xZpFhcIY
63. Popov PA (2015c) Long hands gesture recognition controller for halo. [Online]. Available: https://www.youtube.com/watch?v=NuVAW6wihZ8
64. Popov PA (2018a) GR cascade system demonstration. [Online]. Available: https://www.youtube.com/watch?v=tlVnWRIQf9A
65. Popov PA (2018b) GR SVM System Demonstration. [Online]. Available: https://www.youtube.com/watch?v=qlMepz8soL8
66. Popov PA (2019) Hand gesture recognition and tracking. [Online]. Available: https://www.youtube.com/watch?v=6ICK1ELZmfE
67. Popov PA (2020) Real-time 2D static hand gesture recognition and 2D hand tracking for human-computer interaction. [Online]. Available: https://doi.org/10.20381/ruor-25778
68. Popov PA (2020a) Long hands gesture recognition: gesture recognition dataset. [Online]. Available: http://www.site.uottawa.ca/research/viva/projects/handgesturerecognition/index.html#dataset_gesture
69. Popov PA (2020b) Long hands gesture recognition and hand tracking: hand tracking dataset. [Online]. Available: http://www.site.uottawa.ca/research/viva/projects/handgesturerecognition/index.html#dataset_handtrack
70. Qian C, Sun X, Wei Y, Tang X, Sun J (2014) Realtime and robust hand tracking from depth. Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit :1106–1113
71. Raheja AMJL, Chaudhary A, Raheja JL, Mishra A, Chaudhary A (2016) Indian sign language recognition using SVM. Pattern Recognit Image Anal 26(2):434–441
72. Rautaray SS, Agrawal (2012) A Design of gesture recognition system for dynamic user interface. In: Technology Enhanced Education (ICTEE), 2012 IEEE International Conference, pp 1–6
73. Rautaray SS, Agrawal A (2011) Interaction with virtual game through hand gesture recognition. Multimedia, Signal Process. Commun. Technol. (IMPACT), 2011 Int. Conf., pp. 244–247
74. Rautaray SS, Agrawal A (2012) Real Time Multiple Hand Gesture Recognition System for Human Computer Interaction. Int J Intell Syst Appl 4(5):56–64
75. Ren Z, Yuan J, Zhang Z (2011) Robust hand gesture recognition based on finger-earth mover's distance with a commodity depth camera. Proc 19th ACM Int Conf Multimed - MM 11:1093
76. Ren Z, Yuan J, Meng J, Zhang Z (2013) Robust part-based hand gesture recognition using kinect sensor. IEEE Trans Multimed 15(5):1110–1120
77. Roy K, Mohanty A, Sahay RR (2017) Deep learning based hand detection in cluttered environment using skin segmentation, pp 640–649
78. Rumyantsev O, Merati M, Ramachandran V (2012) Hand Sign recognition through palm gesture and movement. Image Process
79. Sahoo JP, Ari S, Patra SK (2019) Hand gesture recognition using PCA based deep CNN reduced features and SVM classifier. In: Proc. - 2019 IEEE Int. Symp. Smart Electron. Syst. iSES 2019, pp 221–224
80. Sefat MS, Shahjahan M (2015) A hand gesture recognition technique from real time video. 2nd Int. Conf. Electr. Eng. Inf. Commun. Technol. iCEEiCT 2015, no. May, pp. 21–23
81. Sepas-Moghaddam A, Pereira FM, Correia PL (2020) Face recognition: A novel multi-level taxonomy based survey. IET Biometrics 9(2):58–67
82. Sheenu, Joshi G, Vig R (2015) Histograms of orientation gradient investigation for static hand gestures. Int. Conf. Comput. Commun. Autom. ICCCA 2015, pp. 1100–1103
83. Spruyt V, Ledda A, Philips W (2012) Real-time hand tracking by invariant hough forest detection. Proc. - Int. Conf. Image Process. ICIP:149–152
84. Spruyt V, Ledda A, Philips W, G. University-telin-ipi-iminds (2013) Real-time, long-term hand tracking with unsupervised initialization. In: Image Processing (ICIP), 2013 20th IEEE International Conference, pp 3730–3734
85. Sridhar S, Mueller F, Oulasvirta A, Theobalt C (2015) Fast and robust hand tracking using detection-guided optimization. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. 07-12-June:3213–3221
86. Stergiopoulou E, Papamarkos N (2009) Hand gesture recognition using a neural network shape fitting technique. Eng Appl Artif Intell 22(8):1141–1158
87. Sun X, Wei Y, Liang S, Tang X, Sun J (2015) Cascaded hand pose regression. Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit 07-12-June:824–832

88. Sun Y, Liang X, Fan H, Imran M, Heidari H (2019) Visual hand tracking on depth image using 2-D matched filter. 2019 UK/China Emerg. Technol. UCET 2019, pp. 17–20
89. Tian F, Hu QC, Zhang TN (2016) A hand gesture detection for multi class cascade classifier based on gradient. Proc. - 5th Int. Conf. Instrum. Meas. Comput. Commun. Control. IMCCC 2015, no. 1, pp. 1364–1368
90. Uwineza J, Ma H, Li B, Jin Y (2019) Static hand gesture recognition for human robot interaction, vol 11741. Springer International Publishing, LNAI
91. Vezzetti E, Marcolin F, Tornincasa S, Ulrich L, Dagnes N (2018) 3D geometry-based automatic landmark localization in presence of facial occlusions. Multimed Tools Appl 77(11):14177–14205
92. Wang F, Zhou L, Cui Z, Li H, Li M (2016) Gesture recognition based on BoF and its application in human-machine interaction of service robot. In: 6th Annu. IEEE Int. Conf. Cyber Technol. Autom. Control Intell. Syst. IEEE-CYBER 2016, vol 2016, pp 115–120
93. Wu D et al (2016) Deep dynamic neural networks for multimodal gesture segmentation and recognition. IEEE Trans. Pattern Anal Mach Intell 38(8):1583–1597
94. Wu W, Li C, Cheng Z, Zhang X, Jin L YOLSE : Egocentric fingertip detection from single RGB images, pp 623–630
95. Yeo H-S, Lee B-G, Lim H (2015) Hand tracking and gesture recognition system for human-computer interaction using low-cost hardware. Multimed Tools Appl 74(8):2687–2715
96. Yewale SK, Bharne PK (2011) Hand gesture recognition using different algorithms based on artificial neural network. 2011 Int. Conf. Emerg. Trends Networks Comput. Commun., no. 1998, pp. 287–292
97. Yuan Q, Sclaroff S, Athitsos V (2005) Automatic 2D hand tracking in video sequences. In: Proc. 7th IEEE Work. Appl. Comupter Vis, pp 250–256
98. Zhao Y, Song Z, Wu X (2012) Hand detection using multi-resolution HOG features. In: 2012 IEEE Int. Conf. Robot. Biomimetics, ROBIO 2012 - Conf. Dig, pp 1715–1720
99. Zhiqi Y (2016) Gesture learning and recognition based on the Chebyshev polynomial neural network. In: Proc. 2016 IEEE Inf. Technol. Networking, Electron. Autom. Control Conf. ITNEC 2016, pp 931–934