Real-Time Hand Tracking and Gesture Recognition Using Semantic-Probabilistic Network

Mykyta Kovalenko, Odessa National Polytechnic University (ONPU), Ukraine, kov.nikit@gmail.com; Svetlana Antoshchuk, Odessa National Polytechnic University (ONPU), Ukraine, svetlana_onpu@mail.ru; Juergen Sieck, University of Applied Sciences Berlin (HTW), Germany, j.sieck@htw-berlin.de

Abstract—In this paper we propose a system for real-time hand gesture recognition for use as a human-computer interface. Firstly, hand detection is performed using a Viola-Jones algorithm. We use the Continuously Adaptive Mean Shift Algorithm (CAMShift) to track the position of each detected hand in each video frame. The hand contour is then extracted using a Border Following algorithm, which is preceded by skin-colour thresholding, performed in HSV colour space. Finally, a semantic-probabilistic network is introduced, which uses an ontological gesture model and Bayesian network for gesture recognition. We then demonstrate the effectiveness of our technique on a training data set, as well as on a real-life video sequence.

Keywords—HCI; hand detection; gesture recognition; tracking; ontology; probabilistic network

I. INTRODUCTION

With the passing of time our view of human-computer interaction constantly changes and evolves. The traditional keyboard and mouse are slowly replaced by touch- and sensor-screens. However, instinctive and direct communication between human and machines is still a challenging task. Gesture recognition is an area of on-going research in computer vision. Body language has always been an important way of communication among humans, providing emphasis and complementing the oral speech or even being used instead of it. Thus, automatic gesture recognition system could be used for improving human-machine interaction. This kind of human-machine interfaces would allow a human user remote control of a variety of devices via hand gestures.

Many approaches to gesture recognition have been developed. Hand gesture recognition is performed through a curvature space method in [1], which involves finding the boundary contours of the hand. This is a robust approach that is scale, translation and rotation invariant on the hand pose, but it is computationally demanding. In [2], a vision-based hand pose recognition technique using skeleton images is proposed, in which a multi-system camera is used to pick the centre of gravity of the hand and points with farthest distances from the centre, providing the locations of the finger tips, which are used to obtain a skeleton image, and finally for gesture recognition. A technique for gesture recognition for sign language interpretation has been proposed in [3].

Zhao et al. [4] explained virtual reality system based on hand gesture. Gaussian distribution was used for building complexion model, YCbCr colour space was used for segmentation purpose and Fourier descriptor as a feature vector, BP neural network was utilised for recognition and all these resulted in an improved recognition rate. Highlight and shadow segmentation results were, however, found to not be perfect. Guan and Zheng [5] introduced a novel approach to pointing gesture recognition based on binocular stereo vision in which users need to wear special clothes or markers and which was found to be suitable for both, left and right handed users. Freeman and Weissman [6] explained television control application by hand gesture. Here, the user uses only one gesture: the open hand facing the camera for controlling the television. Sepehri et al. [7] proposed algorithms and applications for using the hand as an interface device in virtual and physical spaces.

In Real-time Vision based Hand Gesture recognition systems, hand tracking and segmentation are the most important and challenging steps towards gesture recognition. Uncontrolled environment, lighting condition, skin colour detection, rapid hand motion and self-occlusions are the challenges that need to be considered while capturing and tracking the hand gesture [8]. Various researchers are working on hand tracking and segmentation to make it robust to achieve a natural interface with machines. Bao et al. [9] introduced a new robust algorithm called Tower method for hand tracking module where skin colour was considered for hand gesture tracking and recognition.

Most of the modern gesture recognition systems use additional hardware, in addition to simple video camera. A good example is the Kinect, which uses a combination of a colour camera and two depth sensors. Another example is the LeapMotion, which uses a set of two infra-red cameras. Limited range of operation is a known disadvantage of both approaches.

Our goal was to develop a system capable of detecting and recognising complex hand gestures in real-time, using only a simple colour-camera with no additional hardware.

II. SYSTEM ORGANISATION

The developed system has a very complex architecture with different components. Fig. 1 shows the overview of the developed hand localisation system for input stream acquisition and Region of Interest (ROI) extraction.

A camera is used to capture a video sequence of the hand at a size of 640x480 pixels, which is a trade-off between image quality and processing time. Each of the video frames is then processed as follows.

This input image (converted into greyscale) is used for the detection of the hand candidates. The Viola-Jones approach is used for hand-candidate detection because of its speed and good detection rate [10].
Once the positions of the hand candidates have been located on the image, the tracking of the hand position is performed in the consecutive frames using the CAMShift algorithm.

At each position, the colour-thresholding inside the ROI is performed in a HSV colour space in order to extract the hand shape. The contour is extracted using Suzuki’s Border Following technique. The contour is then analysed to detect finger tips.

Visual descriptors are calculated and then used as an input to the probabilistic network in order to recognise the gesture.

III. HAND DETECTION AND TRACKING

At first we have to find hand-candidates. The Viola-Jones approach is used for the detection of hand-candidates. The main reason for the selection of the Viola-Jones approach is speed and its good detection rate [10]. Once the positions of the hand candidates have been located in the image, each of the candidates is verified by using skin-colour histograms.

A. Hand-candidate detection

The aim of the hand-candidate detection process is to determine relatively small areas of the image where a hand could be located. This process should be as fast as possible.

That means we have to be able to process the entire frame in real time, in less than 40 milliseconds.

The selected Viola-Jones approach fulfils all the basic conditions. The approach is primarily used for face detection [10]; however, it has also been used for hand detection [11].

The Viola-Jones object detector operates on greyscale images. In addition, a sliding-window approach is used to examine each image location for the presence of a hand. The classifier itself is built as a cascade of weak classifiers. These weak classifiers are constructed using AdaBoost on a large set of 2D HoG features, which can be computed quickly by using integral images (each image element on the integral image contains the sum of all the pixels to its upper left on the original image). Weak classifiers are trained to have a very high detection rate (99.5% in our case) and a low false positive rate (2.5% in our case). This means that at each stage of the classification most of the real hands will be passed to the subsequent classifiers in the cascade, while a part of the non-hands will be rejected. Thus, most of the non-hand areas will be rejected in the first several stages of the cascade, which, together with fast feature computation, makes real-time hand detection possible.

We used the OpenCV [12] implementation of the Viola-Jones object detector. Figure 2 shows an example of hand-candidate detection in a real-time video-stream, captured from a webcam.

B. Hand tracking

We use the Continuously Adaptive Mean Shift Algorithm (CAMShift) to track the position of a hand in each video frame [13]. This is an extended motion-tracking algorithm of the Mean Shift algorithm in the continuous images sequence which is a non-parametric method based on the gradient of dynamic distribution of probability density function. The basic idea is to apply Mean Shift operations to all the frames in a video sequence, using the mass centre and size of search window obtained in the previous frame as the initial value of search window in the next frame, and achieving the target tracking by iteration.
The CAMShift algorithm mainly uses information of colour probability distribution of moving targets in video images to achieve the tracking purpose [14], which can effectively solve objective deformation and objective shelter problems with high computational efficiency.

Our implementation is capable of simultaneously tracking multiple objects in a video with relative ease. An example of hand tracking can be seen on figure 3. Here we can see that two hands were detected on the first frame. They are bounded by different coloured rectangles and then tracked throughout the video sequence.

At this stage we also calculate several simple visual descriptors, such as the hand’s movement speed, movement direction, orientation angle and track of movement. All the extracted descriptors will be used later in the process of gesture recognition.

C. Skin-colour detection

Having extracted the portion of the scene around the hand, whether by detection or tracking, our next task is to segment the hand in the image from the background. Skin-colour detection aims at determining whether a colour pixel has the colour of human skin or not. This type of classification should overcome difficulties such as different skin tones (white, pink, yellow, brown and black), scene illuminations and the fact that background pixels can have the same colour as skin [15].

The problem of the RGB colour space is that it does not provide the correct information about skin colour due to the problem of luminance effects. HSV provides colour information as Hue (or colour-depth), Saturation (or colour-purity) and intensity of the Value (or colour-brightness). Hue refers to the colour of red, blue and yellow and has the range of 0 to 360. Saturation means purity of the colour and takes the value from 0 to 100%. Value refers the brightness of the colour and provides the achromatic idea of the colour. From this colour space, H and S will provide the necessary information about the skin colour.

The skin in channel H is characterized by values between 0 and 50, in the channel S from 0.23 to 0.68 for Asian and Caucasian ethnics [16]. The parameters were chosen after extensive experimentation in a large number of images. After we perform colour-segmentation, we can extract the contour of a hand using the Border following algorithm [17] and perform contour analysis.

D. Finger tips extraction

Having extracted the contour, the next step is to extract key features in order to detect more complex hand gestures. Different techniques were considered. One of the studies suggested extracting the top half of the hand region, which presumably contains fingers [18]. In another study a contour analysis was performed to locate sharp turns of the contour line, which were assumed to be fingertips. We have established that using the information about fingertips is the most advantageous approach, which provides a good balance between performance and gesture-complexity.

To locate fingertips the extracted contour is analysed to find convexity information. To understand the shape of an object convex hull and convexity defects information can be used.

The convex hull of a set of points $S$ in the Euclidean space is the smallest convex set that contains all the set of given points. For example, when this set of points is a bounded subset of the plane, the convex hull can be visualized as the shape formed by a rubber band stretched around this set of points. Each point $x_i$ in $S$ is assigned a weight or coefficient $\alpha_i$ in such a way that the coefficients are all non-negative and sum to one, and these weights are used to compute a weighted average of the points. Expressing this as a single formula, the convex hull is the set:

$$\text{conv}(S) = \left\{ \sum_{i} \alpha_i x_i : \sum_{i} \alpha_i = 1 \land \alpha_i \geq 0 \right\}$$

A convex hull is drawn around the contour of the hand, such that all contour points are within the convex hull. This creates an envelope around the hand contour.

When the convex hull is drawn around the contour of the hand, it fits a set of contour points of the hand within the hull.
It uses a minimum of points to form the hull to include all contour points inside or on the hull and maintain the property of convexity. This causes the formation of defects in the convex hull with respect to the contour drawn on the hand. A defect is present wherever the contour of the object is away from the convex hull drawn around the same contour. Convexity defects are valley points. In particular, convexity defects are sequences of contour points between two consecutive contour vertices on the contour hull. These convexity defects are very useful in providing information about the location and state of the fingers. The fingers are therefore described as sets of two points: the coordinates of the palm centre and the coordinate of the fingertip. The palm centre is calculated as the contour’s centre of mass. Figure 5 shows an example of the implemented algorithm being used in our system. A convex hull, build around the hand contour can be seen, as well as the extracted fingertips. From that we can extract the necessary information about the fingers, for instance the amount of detected fingers and their angles, relative to each other. This allows discerning between different hand-postures like “Fist”, “Peace sign”, “Thumbs up”, etc.

IV. GESTURE RECOGNITION

To recognise the hand gesture we developed a semantic-probabilistic network [19], which is a modified Bayesian network, constructed with the help of a domain ontology, organised by a decomposition of each programmed gesture into several levels of abstraction. For that we use the Cambridge Hand Gesture Data set [20].

A. Building the ontology

The ontology contains a structure set of concepts and their relations, which can be used to describe aspects of the field of application and perform logical inference. The structure of a hand gesture is organised in a hierarchical ontology by decomposing it into three levels of abstraction.

The first level of abstraction contains the visual descriptors, computed from the video data: hand movement speed, track and the angle of movement, the orientation angle, the number of detected fingers and the relative angle between each finger.

The second level, the macro-gesture, describes the hand’s movement on a larger scale, its overall state, and is derived from the hand’s movement speed and movement direction. States like “Static”, “Swipe left” or “Swipe up” are considered to be macro-gestures.

On the level of micro-gestures we consider the amount of detected (outstretched) fingers and their angles, relative to each other. This allows discerning between different hand-postures like “Fist”, “Peace sign”, “Thumbs up”, etc.

Finally, at the highest level, the information about both micro- and macro-gestures is combined to determine the overall complex gesture. For some gestures, the information from the lowest level (for example, movement track) is also considered.

The gesture ontology is constructed semi-automatically, using a supervised-training approach, where the visual descriptors for each training sample are extracted and matched automatically, while the more complex hierarchy of macro- and micro-gestures is constructed manually.

B. Constructing a Bayesian network

The ontology serves as a basis for constructing a network of Bayesian inference. The structure of a Bayesian network consists of a directed acyclic graph (DAG) \( G \), whose connectivity matrix defines the conditional dependence and independence relationships among its constituent nodes \( X \) and hence defines the form of the conditional probability tables [21].

The visual descriptors serve as input nodes for the Bayesian network. The nodes of the network are represented as discrete states, with the input nodes quantised to several different values.

Training the network structure requires a means of searching the space of all possible DAGs over the set of nodes \( X \) and a scoring function to evaluate a given structure over the training data \( D \).

For training the network structure we chose a \( K2 \) algorithm [21] — a greedy search technique, which starts from an empty network but with an initial ordering of the nodes. In order to compute the score of a candidate network over the training data while avoiding over-fitting we used a Bayesian Information Criterion (BIC) [22], which approximates the marginal likelihood using a Minimum Description Length (MDL) approach.

Figure 6. A fragment of network to recognise a gesture “Swipe left”.

Figure 5. Example of finger tips extraction

271
An example of a constructed Bayesian network can be seen in Figure 6.

Once the network structure has been trained, parameters can be estimated through a maximum likelihood estimation using the log-likelihood of the training data.

After constructing and training the Bayesian network is ready for hand gesture recognition in a real-time video.

V. RESULTS

Firstly, we constructed the domain ontology to be used for probabilistic network construction. The training data for the network was taken from several sources, including the Cambridge Hand Gesture Data set, as well as several video sequences, recorded manually using a webcam and a Smartphone camera.

At the current stage, a total of ten gestures were programmed to be recognised by our system: swipe in 4 directions, a clenched fist and a hand showing numbers from one to five.

We then performed an experimental testing using live video from a webcam to see how the system performs in real-time. Figure 7 shows an example of recognising several simple gestures. Here we only count the amount of raised fingers and a hand showing numbers from one to five.

First of all we tested the hand detection and tracking modules. To measure the effectiveness of hand detection and tracking, we employ the measures of sensitivity and precision [23]. Figure 8 shows the principle idea of the measures. Here, region A is the ground truth data, and region B is the result of the hand detection/tracking, as implemented in our system.

The sensitivity measure for one image is calculated as follows:

\[
Sens = \frac{|TP|}{|TP| + |FN|},
\]

where TP is the amount of true positives – pixels that were correctly classified as belonging to the region, where the hand is located; and FN is the amount of false negatives – pixels that belong to the hand region, but were not detected.

The precision for a video frame is measured as

\[
Prec = \frac{|FP|}{|FP| + |TP|},
\]

where FP is the amount of false positives – pixels, that were incorrectly classified as belonging to the hand area.

For every frame of every video sequence we detected and tracked the position of a hand, compared it to the ground truth by calculating the sensitivity and precision measures. We then calculated the average values of precision and sensitivity, which are 0.88 and 0.97 correspondingly.

Next we evaluated the performance of our gesture recognition module. The results of the experimental testing are shown in Table 1.

As illustrated, the movement (swiping) gestures are recognised with a high accuracy rate. The clenched fist is sometimes confused with a “Thumbs up”, hence the lower detection rate. The same applies for “Digit Five” gesture, which sometimes gets confused with the “Digit Four”.

We have noted that images taken under insufficient light (especially using the webcam) have led to the incorrect results. In these cases the failure mainly results from the erroneous segmentation of some background portions as hand region.
TABLE I. TESTING RESULTS

<table>
<thead>
<tr>
<th>Gesture</th>
<th>Detection rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Swipe Left</td>
<td>94%</td>
</tr>
<tr>
<td>Swipe Right</td>
<td>96%</td>
</tr>
<tr>
<td>Swipe Down</td>
<td>98%</td>
</tr>
<tr>
<td>Swipe Up</td>
<td>98%</td>
</tr>
<tr>
<td>Clenched Fist</td>
<td>88%</td>
</tr>
<tr>
<td>Gesture Digit 1</td>
<td>97%</td>
</tr>
<tr>
<td>Gesture Digit 2</td>
<td>96%</td>
</tr>
<tr>
<td>Gesture Digit 3</td>
<td>94%</td>
</tr>
<tr>
<td>Gesture Digit 4</td>
<td>96%</td>
</tr>
<tr>
<td>Gesture Digit 5</td>
<td>89%</td>
</tr>
</tbody>
</table>

Our algorithm appears to perform well with somewhat complicated backgrounds, being on par with or surpassing other approaches, as long as there are not too many background pixels with skin-like colours.

VI. CONCLUSION AND FUTURE WORK

We proposed a fast and simple algorithm for a hand gesture recognition problem. The system initially locates the hand candidates using the Viola-Jones approach and tracks the detected hands using the CAMShift algorithm. The contour of a hand is then extracted with skin-colour thresholding. Finally, finger-tips are detected.

The information about the hand, such as movement speed and direction, angle and finger-count is used to construct a semantic model for every gesture. The gesture is recognised by using a Bayesian network.

Our algorithm can be extended to recognise a broader set of gestures. Particularly, to consider the movements of each individual finger to detect more complex gestures involving finger movement.

The segmentation portion of our algorithm is still simple, and has to be improved. However, we should note that the segmentation problem in a general setting is an open research problem itself.

The employed Viola-Jones detector generates a substantial amount of false positives and should also be modified to filter these out. A possible solution could be using better training data for the detector or using a SIFT algorithm.

The semantic-probabilistic network introduced above is not limited just to gesture recognition. In addition it can also be used for various other applications, for example, human behaviour recognition [19].

REFERENCES


