Expert Systems With Applications 136 (2019) 217–229

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Fingertip detection and tracking for recognition of air-writing in videos

Sohom Mukherjee[a,*], Sk. Arif Ahmed[b], Debi Prosad Dogra[c], Samarjit Kar[b], Partha Pratim Roy[d]

[a] Department of Electrical Engineering, National Institute of Technology Durgapur, Durgapur 713209, India
[b] Department of Mathematics, National Institute of Technology Durgapur, Durgapur 713209, India
[c] School of Electrical Science, Indian Institute of Technology Bhubaneswar, Bhubaneswar 751013, India
[d] Department of Computer Science and Engineering, Indian Institute of Technology Roorkee, Roorkee 247667, India

ARTICLE INFO

Article history:
Received 9 September 2018
Revised 3 March 2019
Accepted 16 June 2019
Available online 17 June 2019

Keywords:
Air-writing
Hand pose detection
Fingertip detection and tracking
Handwritten character recognition
Human-computer interaction (HCI)

ABSTRACT

Air-writing is the process of writing characters or words in free space using finger or hand movements without the aid of any hand-held device. In this work, we address the problem of mid-air finger writing using web-cam video as input. In spite of recent advances in object detection and tracking, accurate and robust detection and tracking of the fingertip remains a challenging task, primarily due to small dimension of the fingertip. Moreover, the initialization and termination of mid-air finger writing is also challenging due to the absence of any standard delimiting criterion. To solve these problems, we propose a new writing hand pose detection algorithm for initialization of air-writing using the Faster R-CNN framework for accurate hand detection followed by hand segmentation and finally counting the number of raised fingers based on geometrical properties of the hand. Further, we propose a robust fingertip detection and tracking approach using a new signature function called distance-weighted curvature entropy. Finally, a fingertip velocity-based termination criterion is used as a delimiter to mark the completion of the air-writing gesture. Experiments show the superiority of the proposed fingertip detection and tracking algorithm over state-of-the-art approaches giving a mean precision of 73.1% while achieving real-time performance at 18.5 fps, a condition which is of vital importance to air-writing. Character recognition experiments give a mean accuracy of 96.11% using the proposed air-writing system, a result which is comparable to that of existing handwritten character recognition systems.

1. Introduction

With the emergence of virtual and augmented reality, the need for the development of natural human-computer interaction (HCI) systems to replace the traditional HCI approaches is increasing rapidly. In particular, interfaces incorporating hand gesture-based interaction have gained popularity in many fields of application viz. automotive interfaces (Ohn-Bar & Trivedi, 2014), human activity recognition (Rohrbach et al., 2016) and several state-of-the-art hand gesture recognition approaches have been developed (Molchanov, Gupta, Kim, & Kautz, 2015; Rautaray & Agrawal, 2015). However, hand motion gestures as such are not sufficient to input text. This necessitates the need for the development of touch-less air-writing systems which may replace touch and electromechanical input panels leading to a more natural human-computer interaction (HCI) approach.

A vision-based system for the recognition of mid-air finger-writing trajectories is not a new problem and substantial work has been done in the past two decades. One of the early works by Oka, Sato, and Koike (2002) used a sophisticated device with an infrared and color sensor for fingertip tracking and recognition of simple geometric shapes trajectories. In Amma, Georgi, and Schultz (2012), inertial sensors attached to a glove were used for continuous spotting and recognition of air-writing. Recently, Misra, Singha, and Laskar (2017) have developed a hand gesture recognition framework that can recognize letters, numbers, arithmetic operators as well as 18 printable ASCII characters using a red marker placed on the tip of the index finger for fingertip detection. In spite of satisfactory performance in terms of accuracy of character trajectory recognition, the above approaches using cumbersome motion sensing and tracking hardware devices impose

* Corresponding author.
E-mail addresses: sm.20150025@btech.nitdgp.ac.in (S. Mukherjee), arif.1984.in@ieee.org (Sk.A. Ahmed), dpdogra@iitbbs.ac.in (D.P. Dogra), samarjit.kar@maths.nitdgp.ac.in (S. Kar), proy.fcs@iitr.ac.in (P.P. Roy).

https://doi.org/10.1016/j.eswa.2019.06.034
0957-4174/

many behavioral constraints on the user. For example, wearing data gloves on the hand may change the user's natural handwriting pattern and is often considered as an undesirable burden by many users. During the past few years, several research works have been carried out addressing the problem of air-writing recognition using depth sensors such as Microsoft Kinect and special hardware namely the Leap Motion Controller from Leap Motion,[1] for input. Chang, Garcia-Hernando, Tang, and Kim (2016) have proposed a Spatio-Temporal Hough Forest for the detection, localization and recognition of mid-air finger-writing in egocentric depth video. Chen, AlRegib, and Juang (2016) have developed a robust system for air-writing recognition using the Leap Motion Controller for marker-free and glove-free finger tracking. However, the high cost and limited availability of such sophisticated hardware to the majority of users render them unsuitable for real-world applications.

The work that is closest to ours is by Huang, Liu, Zhang, and Jin (2016) where they use a two stage CNN-based fingertip detection framework for recognition of air-writing in egocentric RGB video. However, capturing egocentric video requires head-mounted smart cameras or mixed reality headsets such as Google Glass, Microsoft HoloLens and Facebook Oculus, which may not be available to all users. Therefore, to make our system more general, we use video inputs from a standard laptop camera or a web-cam for the air-writing application. This makes the task even more challenging due to the presence of the face in the video frames, which being a moving object of similar skin tone as the hand, makes the detection of the hand and hence the fingertip much more complicated.

To address these issues we propose a new system for air-writing recognition in videos using a standard laptop camera or a web-cam for the video input. The key challenges in this task are: (1) writing hand pose detection for initialization of air-writing; (2) accurate fingertip detection and tracking in real-time; (3) recognition of air-writing character trajectory. The task is fairly difficult due to several factors such as hand shape deformation, fingertip motion blur, cluttered background and variation in illumination.

To address the aforementioned challenges, our work makes the following contributions:

- We propose a new writing hand pose detection algorithm for the initialization of air-writing using the Faster R-CNN framework for accurate hand detection followed by hand segmentation and finally counting the number of raised fingers based on geometrical properties of the hand.
- We propose a robust fingertip detection approach using a new signature function called distance-weighted curvature entropy.
- We propose a fingertip velocity-based termination criterion which serves as a delimiter and marks the completion of the air-writing gesture.

The rest of the paper is organized as follows. Section 2 provides a review of the related work. Section 3 presents the proposed air-writing recognition algorithm detailing its four stages, namely writing hand pose detection, fingertip detection, fingertip tracking and trajectory generation, and character recognition. In Section 4, details of the implementation, dataset and experimental analysis are presented. Finally, in Section 5, conclusions are drawn and the scope for future development of the proposed approach is discussed.

2. Related work

Recognizing mid-air finger-writing is a difficult and open problem in computer vision, due to the various challenges outlined in the previous section. We review the previous research work related to this work as follows: (1) fingertip detection and tracking; (2) air-writing recognition; and (3) the existing hand datasets related to air-writing recognition.

2.1. Fingertip detection and tracking

Fingertips detection and tracking has been an active topic in the fields of HCI and augmented reality (AR) using color as well as depth cameras. A model-based approach for 3D hand tracking (de La Gorce, Fleet, & Paragios, 2011; Tang, Chang, Tejani, & Kim, 2017) may be used as a preceding step for fingertip detection, but these approaches involve high computational cost and require a large amount of training data. This makes them unsuitable for our real-time application. In the model-less approach, the hand silhouette is first segmented using color, depth or motion cues and then the fingertips are detected from the extracted binary hand mask. Liang, Yuan, and Thalmann (2012) used a distance metric from hand palm to the contour furthest points to localize candidate fingertip points. Krejov and Bowden (2013) extended the distance concept employing it with the natural structure of hand using a geodesic distance. This improved the localization of fingertips in hand configurations where previous methods failed. In Lee and Hollerer (2007), the authors used the contour curvature as a cue to detect fingertips, exploiting the fact that fingertips are high curvature points, compared to other parts of the hand.

However, all the above approaches suffer from the drawback that the fingertip detection depends largely on the hand segmentation, and suffer severely from poor segmentation results while making use of solely color, depth and motion cues (Zhang, Ye, Jin, Feng, & Xu, 2013). The problem can be solved by accurate detection of hands as a preceding step for hand segmentation. In Mittal, Zisserman, and Torr (2011), a skin-based detector, a deformable part models (DPM)-based hand shape detector and a context detector are combined to detect hands in still images, but this method is time consuming for real-time applications due to the sliding window approach. In Li and Kitani (2013), hand detection and pixel-level segmentation from egocentric RGB videos using local appearance features have produced good results, but the performance is affected by variation in illumination.

Recent advances in deep learning based methods are giving excellent results on general object detection tasks. However, hand detection remains a challenging task and numerous works have been done in recent times. The Region-based CNN (R-CNN) framework is applied in Bambach, Lee, Crandall, and Yu (2015) to detect hands in egocentric videos. This method achieves promising performance in complex environments, but is slow due to CNN-based feature computation for redundant overlapping region proposals. A recent work by Deng et al. (2018) proposes a CNN-based joint hand detection and rotation estimation framework, using on an online derotation layer embedded in the network. In Roy, Mohanty, and Sahay (2017), a two-step framework is proposed for detection and segmentation of hands using a Faster R-CNN based hand detector followed by a CNN based skin detection technique. An experimental survey of existing hand segmentation datasets, state-of-the art methods as well as three new datasets for hand segmentation can be found in Khan and Borji (2018). Deep learning based methods have been extended to fingertip detection in Huang et al. (2016). They use a two-stage framework consisting of Faster R-CNN based hand detection followed by CNN-based fingertip detection. More recently, Wu, Li, Cheng, Zhang, and Jin (2017) proposed a framework called YOLSE (You Only Look what You Should See), that uses a heatmap-based fully convolution network for multiple fingertip detection from single RGB images in egocentric view.

Tracking a small object such as the fingertip from an image accurately and robustly remains a great challenge and to

[1] https://www.leapmotion.com/.

the best of our knowledge no standard method for fingertip tracking has yet been proposed. Mayol, Davison, Tordoff, Molton, and Murray (2004) and Kurata, Okuma, Kourogi, and Sakaue (2001) have used template matching and mean-shift respectively for hand tracking in constrained environments, from images captured using a wearable camera. Stenger, Thayananthan, Torr, and Cipolla (2006) gives a tracking framework, which is applied to the recovery of three-dimensional hand motion from an image sequence using a hierarchical Bayesian filter. In Kolsch and Turk (2004), a fast tracking method has been proposed for non-rigid and highly articulated objects such as hands. It uses KLT features along with a learned foreground color distribution for tracking in 2D monocular videos. However, these methods cannot deal with long-term continuous tracking problems such as change in object appearance and object moving in and out of the frame, since they are designed and evaluated for short video sequences. The Tracking-Learning-Detection (TLD) framework (Kalal, Mikolajczyk, & Matas, 2012) has been proposed for long-term tracking. TLD works well when there are frequent variations in the appearance of an object, but is slow compared to many other tracking algorithms. Therefore, long-term tracking of small objects remains a challenging task.

2.2. Air-writing recognition

Some methods have been proposed in the literature for the recognition of air-writing, treating it as a spatial-temporal signal. In Chen et al. (2016), various modifications of Hidden Markov Models (HMMs) have been used for the recognition of air-writing generated using Leap Motion Controller. An attention-based model, called attention recurrent translator has been used in Gan and Wang (2017) for in-air handwritten English word recognition, which gives performance comparable to the connectionist temporal classification (CTC). Kane and Khanna (2017) have proposed an equipolar signature (EPS) technique for a vision-based mid-air unistroke character input framework that is resistant to variations in scale, translation and rotation. Some recent works have also addressed the problem of 3D air signature recognition and verification (Behera, Dogra, & Roy, 2017; 2018). Behera, Dogra, and Roy (2018) presents a method for analyzing 3D air signatures captured using the Leap Motion Controller, with the help of a new geometrical feature extracted from the convex hull enclosing a signature.

In recent years, deep convolutional neural networks (CNNs) have achieved great success in handwritten character recognition, beating benchmark performances by wide margins (Ciregan, Meier, & Schmidhuber, 2012; Cireşan & Meier, 2015). The multi-column deep neural network proposed in Ciregan et al. (2012) reaches near-human performance on the MNIST handwritten digits dataset. A variation of CNNs called DeepCNet, first proposed by Graham (2013), won first place in the ICDAR 2013 Chinese Handwriting Recognition Competition (Yin, Wang, Zhang, & Liu, 2013). Further advances in CNN architectures namely DropWeight (Xiao, Yang, Ahmad, Jin, & Chang, 2017) and new training strategies such as DropDistortion (Lai, Jin, & Yang, 2017) and DropSample (Yang, Jin, Tao, Xie, & Feng, 2016) have led to improved performance of CNNs in online handwritten character recognition tasks.

2.3. Related datasets

At present, there isn't any benchmark dataset in the area of finger air-writing research, to evaluate the performance of the methods. Recently, some datasets have been released for the recognition of air-writing in egocentric view. The EgoFinger dataset (Huang et al., 2016) containing 93,729 labelled RGB frames from 24 egocentric videos and captured from 24 different individuals, is the most extensive among such datasets. Other datasets for the recognition of egocentric hand gesture include the EgoHands dataset (Bambach et al., 2015), containing 48 videos of egocentric interactions between two people, with pixel-level ground-truth annotations for 4800 frames and 15,053 hands. RGB-D datasets are comparatively more numerous. The EgoGesture (Zhang, Cao, Cheng, & Lu, 2018) is a multi-modal large scale dataset for egocentric hand gesture recognition containing 2081 RGB-D videos, 24,161 gesture samples and 2,953,224 frames acquired from 50 distinct subjects. In Suau, Alcoverro, López-Méndez, Ruiz-Hidalgo, and Casas (2014) a real-time fingertip detection method has been proposed with fingertip labeled RGB-D dataset. However, none of these datasets are suitable for our proposed air-writing application using web-cam video input.

3. Proposed algorithm

A flowchart for the proposed air-writing recognition system is shown in Fig. 1. The proposed algorithm can be broken into the following four components: (1) writing hand pose detection; (2) fingertip detection in each frame; (3) fingertip tracking and character trajectory generation; (4) recognition of air-writing characters.

3.1. Writing hand pose detection

The detection of writing hand pose and its recognition from other gestures is an integral step towards air-writing initialization since, unlike conventional handwriting which has the pen-down and pen-up motion, air-writing does not have such a delimited sequence of writing events. In this work, we define the writing hand pose to be a single raised finger with the assumptions that the finger is free from occlusions and is not manipulating any object. We detect a writing hand pose and discriminate it from a non-writing hand pose by counting the number of raised fingers. To this end we propose a four-fold approach consisting of hand detection, hand region segmentation, localization of the hand centroid, followed by utilizing geometrical properties of the hand to count the number of raised fingers.

3.1.1. Hand detection

The Faster R-CNN (FRCNN) (Ren, He, Girshick, & Sun, 2015) framework has been used for hand detection. Faster R-CNN is a state-of-the-art object detection algorithm that removes dependency on external hypothesis generation methods, as in case of Fast R-CNN (Girshick, 2015). The first step involves using the output of an intermediate layer of a CNN pretrained for the task of classification (called base network) to generate a convolutional feature map. The original Faster R-CNN used VGG-16 network (Simonyan & Zisserman, 2014) pretrained on ImageNet, but since then many different networks with a varying number of weights have been used. The base network returns a convolutional feature map that forms the input to the Region Proposal Network (RPN). The RPN is a fully convolutional network that takes the reference bounding boxes (anchors) and outputs a set of object proposals. This is followed by region of interest (RoI) pooling for extraction of features from each of the object proposals. Finally, we use these features for classification using a region-based convolutional neural network (R-CNN). After putting the complete FRCNN model together, we train it end-to-end as a single network, with four losses, RPN classification loss, RPN regression loss, R-CNN classification loss and R-CNN regression loss. The details of implementation and training of the proposed Faster R-CNN based hand detector is given in Section 4.2.

3.1.2. Hand region segmentation

Hand region segmentation is achieved using a two-step approach viz. skin segmentation and background subtraction and the

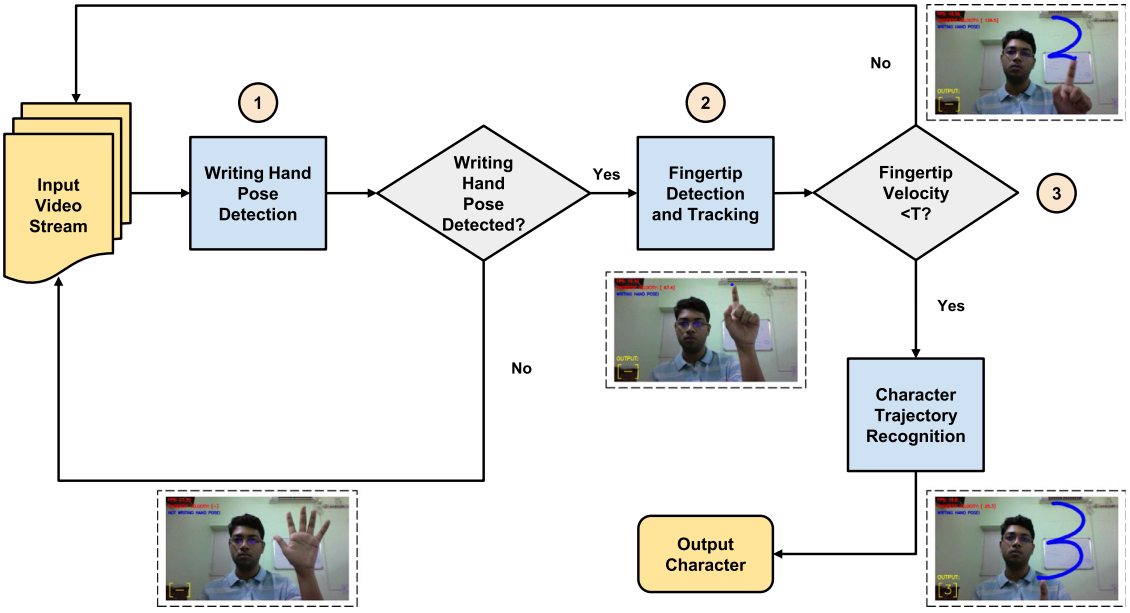


Fig. 1. Flowchart of the proposed air-writing recognition system. The three main contributions in the work have been highlighted by numbering them in the order they appear in the pipeline.

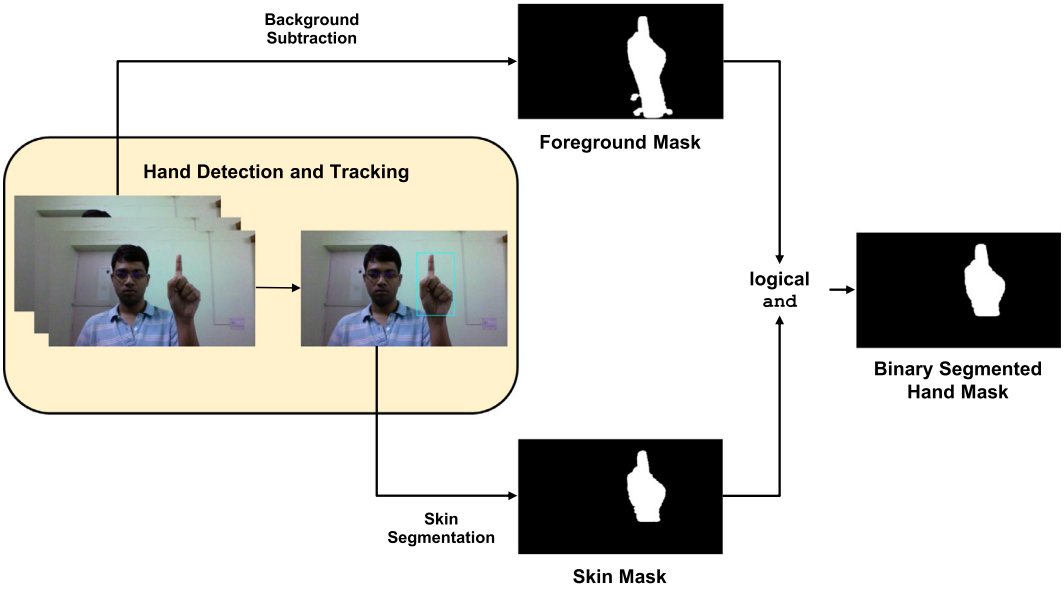


Fig. 2. Hand detection and segmentation.

final binary hand image is obtained as an aggregation of the two. In spite of the success of semi-supervised segmentation algorithms namely GrabCut for hand segmentation (Bambach et al., 2015), we find that such algorithms are computationally expensive and hence not suitable for our purpose. The proposed algorithm is found to work well in real-time and gives fairly accurate segmentation results. Fig. 2 shows a schematic diagram for the proposed hand segmentation algorithm.

Skin Segmentation: We propose a skin color filtering approach based on a static skin color model in the YCbCr color space for skin segmentation. Although skin colors vary considerably across races, it has been observed that the skin chrominance has only a small range across different skin types, while the skin luminance varies considerably. This forms the basis of the static skin color model (Phung, Bouzerdoum, & Chai, 2005). Based on extensive experimentation (Phung et al., 2005), we extracted the narrow band $77 \leq Cb \leq 127$ and $133 \leq Cr \leq 173$ as our static skin color model and discarded the Y component to exclude the luminance information. A frame of the video sequence acquired by the web-cam is in RGB color space, and we can convert it to YCbCr color space using the following equation (Wu & Kang, 2016):

$$\begin{bmatrix} Y \\ Cb \\ Cr \\ 1 \end{bmatrix} = \begin{bmatrix} 0.2990 & 0.5870 & 0.1140 & 0 \\ -0.1687 & -0.3313 & 0.5000 & 128 \\ 0.5000 & -0.4187 & -0.0813 & 128 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \\ 1 \end{bmatrix} \quad (1)$$

We construct a skin filter based on the YCbCr static skin color model and filter the candidate hand bounding box, obtained from detection phase to get the binary skin mask I_{h1} as follows:

$$I_{h1}(x, y) = \begin{cases} 1 & \text{if } 77 \leq Cb \leq 127 \text{ and } 133 \leq Cr \leq 173 \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

Fig. 3. Hand region segmentation in skin tone background demonstrating a situation where skin segmentation fails partially and segmentation performance is improved due to background subtraction. (a) Input RGB image from web-cam. (b) Foreground mask. (c) Skin mask. (d) Binary segmented hand image.

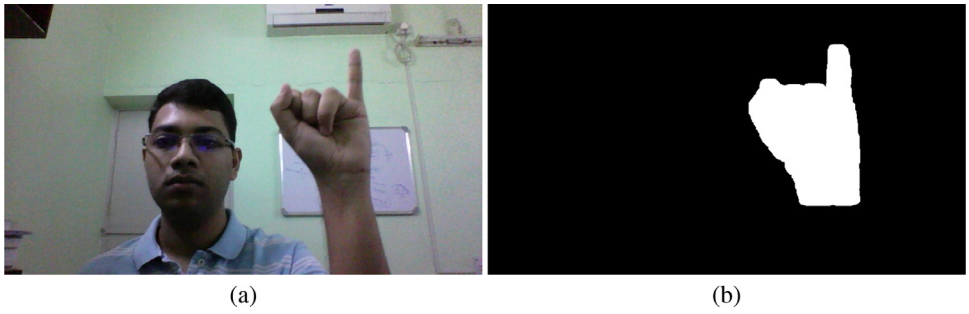


Fig. 4. Hand region segmentation demonstrating a situation where the person performs air-writing gesture using little finger. (a) Input RGB image from web-cam. (b) Binary segmented hand image.

Fig. 5. Hand centroid localization using distance transform. (a) Binary hand image. (b) Distance transform (the pixel having the maximum intensity is highlighted with a red marker).

Skin color filtering method is chosen for skin segmentation over other methods namely Gaussian classifiers and the Bayesian classifier, in spite of their slightly better performance, since they increase the computational load significantly and hence are not suitable for real-time applications.

Background Subtraction: The background subtraction step is used to remove any skin colored object (not a part of the hand) that may be present inside the detected hand bounding box. Fig. 3 shows one such situation where skin color-based segmentation partially fails (due to the presence of skin tone background) and background subtraction proves to be useful in improving the segmentation performance. We use the adaptive Gaussian Mixture Model (GMM) based background subtraction algorithm proposed in Zivkovic (2004) for this purpose. The main advantage of GMM is that it can reach real-time processing. Let the binary foreground mask obtained from this step be I_{h2}.

The final binary hand image is obtained by a logical and operation between the binary skin mask I_{h1} and the binary foreground mask I_{h2} as follows:

$$I_h = I_{h1} \wedge I_{h2} \tag{3}$$

The above approach using an aggregation of two segmentation algorithms gives us an accurate hand segmentation result which is free from noise due to moving objects in the scene which are not the hand as well as any other object inside the detected hand bounding box which is skin colored but does not belong to the hand. The segmented hand image I_h is then filtered applying the morphological operations of dilation $\oplus$ and erosion $\ominus$, as described in Eq. (4), where I_h is the segmented hand image, I_{hf} is the resulting filtered image and K is a circular disk structuring element of radius 3 with the anchor at its center.

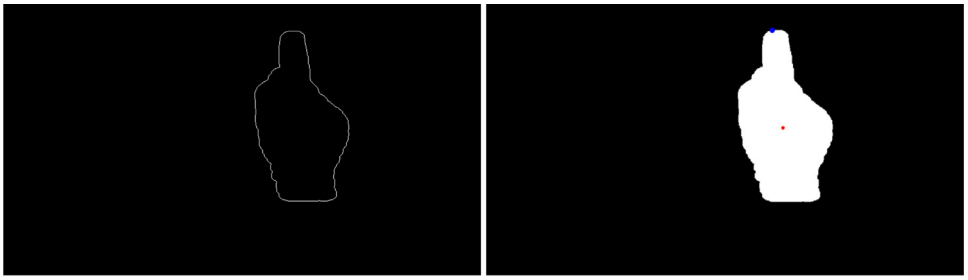
$$I_{hf} = ((I_h \oplus K) \ominus K) \oplus K \tag{4}$$

These morphological operations are employed in order to fill spaces in the foreground object (hand) with dilation, and then clean any background noise with an erosion. Finally, the hand silhouette is regularized using dilation. Lastly, we sort all connected regions in I_{hf} by area, and extract the connected region having the maximum area as the hand region. Fig. 4 shows a case where the person performs air-writing gesture using little finger. This demonstrates the robustness of our segmentation algorithm to fingers of different shapes and sizes.

3.1.3. Hand centroid localization

We employ two algorithms to find the initial estimates of the hand centroid and the final centroid is calculated as the mean of the two.

The method of distance transform is used to get the first estimate of the centroid (x_{c1}, y_{c1}). In the distance transform image, each pixel is represented by its distance from the nearest boundary pixel. Fig. 5 shows an example of distance transform. Fig. 5a shows a binary hand image and Fig. 5b shows the distance transform image of the former. The Euclidean distance has been used to measure the distance between a pixel and its nearest boundary pixel.



(a) (b)

Fig. 6. Fingertip detection. (a) Hand contour s extracted from the binary hand image in Fig. 5a. (b) Binary hand image with centroid (highlighted with a red marker) and fingertip (highlighted with a blue marker).

As it can be seen in the Fig. 5b, the center point of the hand has the largest distance (depicted by maximum intensity). Therefore, the pixel having the maximum intensity in the distance transform image (Fig. 5b), is taken to be the centroid.

The second estimate of the centroid (x_{c2}, y_{c2}) is found using the concept of image region moments. The image moments M_{ij}, for an image with pixel intensities $I(x, y)$, can be calculated as:

$$M_{ij} = \sum_x \sum_y x^i y^j I(x, y) \tag{5}$$

The centroid (x_{c2}, y_{c2}) is given by:

$$x_{c2} = \frac{M_{10}}{M_{00}},\ y_{c2} = \frac{M_{01}}{M_{00}} \tag{6}$$

where M_{10}, M_{01}, M_{00} are the moments of the binary hand region obtained from the previous segmentation step.

The final centroid (x_c, y_c) is obtained as the mean of the two initial estimates.

3.1.4. Counting raised fingers

We use the geometrical properties of the hand to count the number of raised fingers. From the centroid of the hand (x_c, y_c) we draw a circle of maximum radius such that it lies entirely within the hand, i.e., not including any background pixel. We call this circle as the inner circle of maximum radius and take its radius to be r. Now with the same center, we construct another circle of radius $R = M_r \times r$ to intersect all the raised fingers. Here, M_r is a magnification factor which is chosen to be 1.5 based on the geometrical shape of the hand and extensive experimentation (Wu & Kang, 2016).

We say that the circle of radius R intersects the hand whenever it crosses from a background pixel to a foreground pixel and vice-versa. Let the number of such intersections be n. Therefore, taking into account the fact that the circle intersects each finger twice and excluding the two intersections with the wrist, the number of raised fingers is given by:

$$N = \frac{n}{2} - 1 \tag{7}$$

Finally, if $N = 1$ i.e., there is a single raised finger, the writing hand pose is detected and the air-writing is initialized, otherwise, the screen is cleared and the entire writing hand pose detection algorithm is repeated on successive frames until a match of $N = 1$ is found.

3.2. Fingertip detection

Although recent CNN-based approaches can give good results for many object detection tasks, detecting the fingertip directly in RGB videos remains challenging, primarily due to its extremely small dimension. Once we have segmented the hand region and localized the hand centroid, we find the fingertip position from the segmented hand using a new signature function, called distance-weighted curvature entropy.

A signature function is a 1-D functional representation of a boundary and can be generated in many ways. The basic idea behind this is to reduce the original 2-D boundary representation to a 1-D function, that is easier to describe. We propose a novel signature function called as distance-weighted curvature entropy which is a product of the distance of each contour point of the segmented binary hand region from the hand centroid and the curvature entropy at each contour point. Fingertips are high curvature points and are also distant from the hand center. We use these two distinctive features of fingertips for their accurate detection.

The first factor of the proposed signature function is a scale in-variant measure of the curvature of a contour presented in Feldman and Singh (2005) and is referred to as curvature entropy u. As shown in Fig. 5a, the segmented hand is represented as a binary image. Let s be a planar contour curve extracted from the segmented hand region of the binary image. We use the border tracing algorithm proposed in Suzuki et al. (1985) for retrieving the hand contour from the binary hand image. The contour retrieval algorithm extracts the outer boundary of the hand region using pixel connectivity of 8 (two pixels are connected if their edges or corners touch). The extracted contour curve s has been illustrated in Fig. 6a. If s is of length L and is sampled at n uniformly spaced points, then the sampling interval will be $\triangle s = L/n$. We consider that between any two points along the sampled contour curve, the tangent direction changes by an angle α, called the *turning angle*. The curvature κ is the change in the tangent direction as we move along the curve and hence may be approximated by the ratio between the turning angle α and the sampling interval $\triangle s$:

$$\kappa(s) \approx \frac{\alpha}{\triangle s} \tag{8}$$

Following the derivation in Feldman and Singh (2005), the curvature entropy u may be approximated as follows:

$$u(\kappa(s)) \propto -\cos(\triangle s \cdot \kappa(s)) \tag{9}$$

Therefore, we find that the curvature entropy u is locally proportional to the curvature $\kappa(s)$ and is also scale-invariant. This allows us to localize high curvature points, namely the fingertip in our case, without the need for exploring different scales.

The second factor of the signature function will be a distance function $\delta(s)$ which is equal to the distance between each contour point and the centroid of the hand (x_c, y_c).

We define the signature function of a contour $\Psi(s)$ as the product of the curvature entropy of the contour $u(\kappa(s))$ and the distance function $\delta(s)$:

$$\Psi(s) = u(\kappa(s)) \cdot \delta(s)^{\gamma} \tag{10}$$

where the parameter γ is a weight for the distance term in the signature function and is tuned during experiments. The distance

(a) Initial trajectory. (b) First iteration. (c) Second iteration. (d) Final iteration.

Fig. 7. Trajectory smoothing for an initially distorted character trajectory.

term weighted by the parameter γ eliminates any high curvature points along the hand contour that are not fingertips, thereby reducing false positives caused by irregularities in the hand contour.

Since the fingertip will have a high value of curvature entropy and will be also faraway from the hand centroid, the signature function $\Psi(s)$, which may be considered as an 1-D array, will have a maximum at the fingertip position (see Fig. 6b). Therefore, the coordinates of the fingertip (x_f, y_f) may be calculated as follows:

$$(x_f, y_f) = s(\arg\max_s \Psi(s)) \tag{11}$$

3.3. Fingertip tracking and character trajectory generation

The third step in the air-writing recognition system is the generation of characters from trajectories formed by sequentially detected fingertip locations. For this, we need to track the fingertips over successive frames starting from the frame in which the writing hand pose is detected until a stopping criterion is reached. This will generate the required character trajectory. Following this, we apply a trajectory smoothing algorithm to remove noise on account of poor fingertip detection or hand trembling.

3.3.1. Fingertip tracking

Detection and tracking of the hand over successive frames is integral to the fingertip detection and tracking performance. Experiments show that using the Faster R-CNN hand detector for every frame is computationally costly and leads to frame rates much lower than real-time performance. Therefore, the KCF tracking algorithm (Henriques, Caseiro, Martins, & Batista, 2015) is used for tracking of the detected hand region. The tracker is initialized with the Faster R-CNN hand detector output and re-initialization is done at an interval of t frames. Re-initialization is necessary in this case since the hand being a highly non-rigid object is difficult to track over a long time. Taking the interval for re-initialization t to be equal to 50 is experimentally found to give the best compromise between tracking accuracy and frame rate.

Tracking a fingertip robustly and accurately is fairly challenging due to its small dimension. However, it is essential to an air-writing application since any erroneous track results in a distorted character. Following the success of CNN-based detection-tracking of fingertips in Huang et al. (2016), we use a detection-based fingertip tracking approach. In each frame, once the hand region is obtained, fingertip detection is carried out as discussed in Section 3.2. Experiments prove that the proposed detection-based tracking method gives the best result in terms of tracking accuracy and speed while state-of-the-art tracking algorithms either suffer from poor frame rate or the fingertip track is lost after only a few frames.

3.3.2. Termination criterion

The termination or delimiting criterion is important to an air-writing system since there is no pen-up motion, as in case of traditional online handwriting. Once the termination criterion is satisfied, the character trajectory is assumed to be complete and the trajectory is passed to the following smoothing and recognition stages. We use the velocity of the detected fingertip as the natural stopping criterion for the air-writing recognition system. The rationale behind this choice is that the velocity of the fingertip while writing the character will be high compared to when the writing is complete and the fingertip will be nearly at rest. Let the coordinates of the fingertip for the r^{th} frame be (x_r, y_r), and the coordinates for the $(r+1)^{th}$ frame be (x_{r+1}, y_{r+1}). Then the displacement d of the fingertip over one frame is given by the Euclidean distance between the two points and the velocity v is the product of the displacement and the frame rate f in fps as shown in the following equations:

$$d = \sqrt{(x^{r+1} - x^r)^2 + (y^{r+1} - y^r)^2} \tag{12}$$

$$v = d \cdot f \tag{13}$$

When the velocity $v < \tau$ the stopping criterion is satisfied. Here τ is a threshold and experiments reveal that a value of $\tau = 40$ gives best results for most users. Besides, in case a distorted trajectory is plotted on account of hand trembling or any other disturbance, the user can manually reach the stopping criterion by providing a non-writing hand pose, which also clears the screen and then again initialize the writing sequence by a writing hand pose in following frames.

3.3.3. Trajectory smoothing

Poor fingertip detection or trembling of hands may lead to the formation of uneven or distorted character trajectory, resulting in misclassification of characters in the recognition stage. Thus, smoothing of the character trajectory is an essential post-processing step in order to attenuate the noise.

Let T be the character trajectory obtained from the preceding steps. We propose a simple iterative smoothing algorithm that replaces a trajectory point T_i by the average value of the two neighboring points T_{i-1} and T_{i+1}, based on the condition that the distance between the point T_i and its preceding point T_{i-1} is greater than a threshold λ. The process is repeated over the entire trajectory until the difference between curvature entropy of the trajectory $u(T)$ for two consecutive iterations is less than a tolerance ϵ. The stopping criterion for the iteration is based on the concept of curvature entropy of a curve described in Section 3.2. It is found experimentally that the average distance between two consecutive points of a character trajectory lies close to 1. Therefore, the value of λ is experimentally chosen as 5, to give a trade-off between accuracy and speed. Also, experiments show that the tolerance ϵ kept at 0.4 gives fairly a good performance in real-time. The proposed smoothing algorithm gives competitive results compared to others, namely the Ramer–Douglas–Peucker algorithm (Misra et al., 2017) at a significantly lower computational cost. Visualizations for successive iterations involved in smoothing of an initially distorted character trajectory is shown in Fig. 7.

3.4. Character recognition

Following the recent success of deep convolutional neural networks (CNNs) on handwritten character recognition tasks, we use

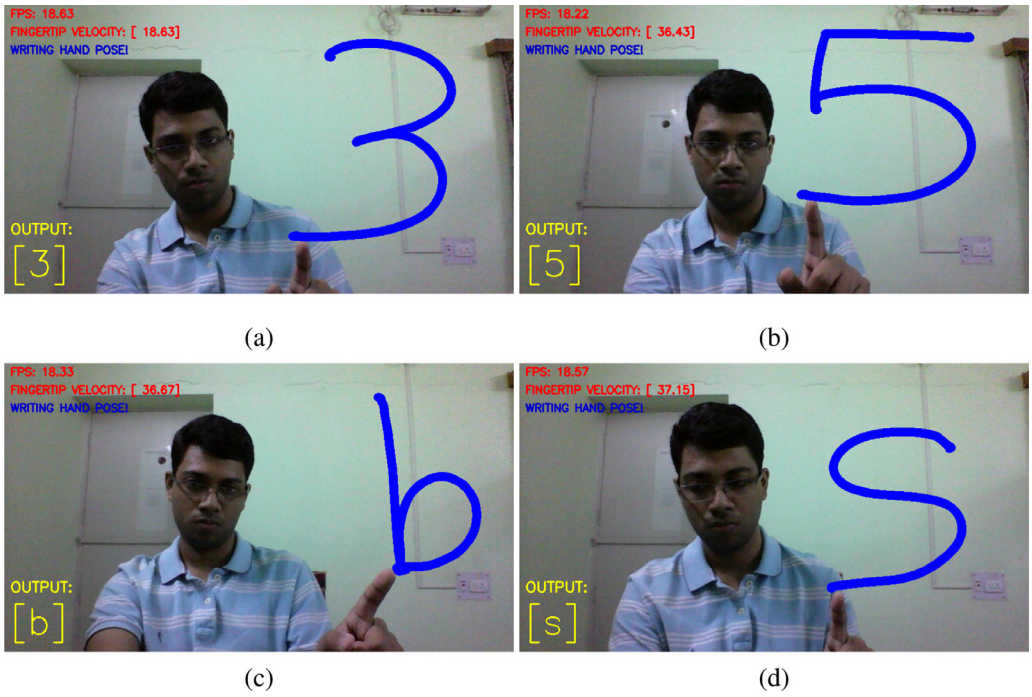


Fig. 8. Air-writing character trajectories of digits and letters along with hand pose description, predicted outputs and instantaneous frame rates using the proposed air-writing recognition system.

the AlexNet architecture (Krizhevsky, Sutskever, & Hinton, 2012), pre-trained on the EMNIST dataset (Cohen, Afshar, Tapson, & van Schaik, 2017) for the recognition of the air-writing character trajectories. It is a well-known fact that accuracy and inference time for deep neural networks (DNNs) are in a hyperbolic relationship, i.e., a small increment in accuracy costs a lot of computational time (Canziani, Paszke, & Culurciello, 2016). Therefore, for all practical applications requiring real-time operation, we need to consider the classification accuracy as well as inference time of DNNs. Since the primary goal of this work is to develop a real-time system for air-writing recognition, we choose the AlexNet architecture over other state-of-the-art CNN architectures such as VGG-16 (Simonyan & Zisserman, 2014), ResNet (He, Zhang, Ren, & Sun, 2016), Inception-v3 (Szegedy, Vanhoucke, Ioffe, Shlens, & Wojna, 2016) and Inception-v4 (Szegedy, Ioffe, Vanhoucke, & Alemi, 2017). This is because AlexNet offers a reasonably high accuracy with very low inference time. Detailed experiments analysing the performance of all these architectures in terms of classification accuracy as well as inference time have been provided in Section 4.4. Fig. 8 shows the final character recognition results for four different characters using the proposed framework.

4. Experiments

In this section, we present extensive experiments to demonstrate the superior performance of the proposed fingertip detection and tracking algorithm and state-of-the-art recognition performance for air-writing character trajectories. All experiments are performed on a machine equipped with a single NVIDIA GeForce GTX 1080 and an Intel Core i7-4790K Processor with 16GB RAM. Fig. 9 shows the different stages involved in performing a complete air-writing gesture by an user using the proposed framework.

4.1. Dataset preparation

The dataset contains air-writing videos with corresponding fingertip trajectories by 5 subjects. The trajectories have been recorded for 36 characters: numbers (from 0 to 9) and English alphabet letters (from a to z), from 5 persons with 10 samples per character per person. We use a typical web-cam for recording the air-writing videos which have backgrounds of varying complexity and illumination levels. In total, 1800 different air-writing character trajectories (10 samples each of 36 characters from 5 persons) have been recorded. Each trajectory consists of a set of points (x, y, t), where (x, y) is the position of the fingertip at time t. Our dataset differs from egocentric fingertip detection datasets viz. the SCUT EgoFinger Dataset (Huang et al., 2016) in the fact that the video sequences have been captured using a web-cam and hence contains the subject's face as well (which is mostly absent in case of egocentric videos). This makes the task more challenging.

4.2. Hand detection and tracking

It is essential to evaluate the hand detection and tracking performance, since it forms the first stage of the algorithm. To train the proposed Faster R-CNN based hand detector, we have created a dataset consisting 25,000 images collected from the EgoFinger dataset (Huang et al., 2016) and the EgoHands dataset (Bambach et al., 2015) with annotated hand regions. Our hand image dataset is split into training, validation, and test sets consisting of 15,000, 3750 and 6250 images, respectively. We use Inception-v2 model (Szegedy et al., 2016) trained on the Microsoft COCO dataset (Lin et al., 2014) as the base network for Faster R-CNN. The network is trained end-to-end using stochastic gradient descent with momentum. The momentum value used is 0.9 and the learning rate starts from 0.0002 and decreases to 0.00002 after 900,000 steps. We use the PASCAL VOC 2007 criteria (Everingham, Van Gool, Williams, Winn, & Zisserman, 2007) for scoring detections (detections are considered true positives if the intersection over union between the predicted bounding box and the ground truth bounding box is greater than 0.5) and obtain an average precision (AP) of 87.6% on the test set.

For hand tracking, we compare three state-of-the-art tracking algorithms, viz. KCF (Henriques et al., 2015), TLD (Kalal et al., 2012)

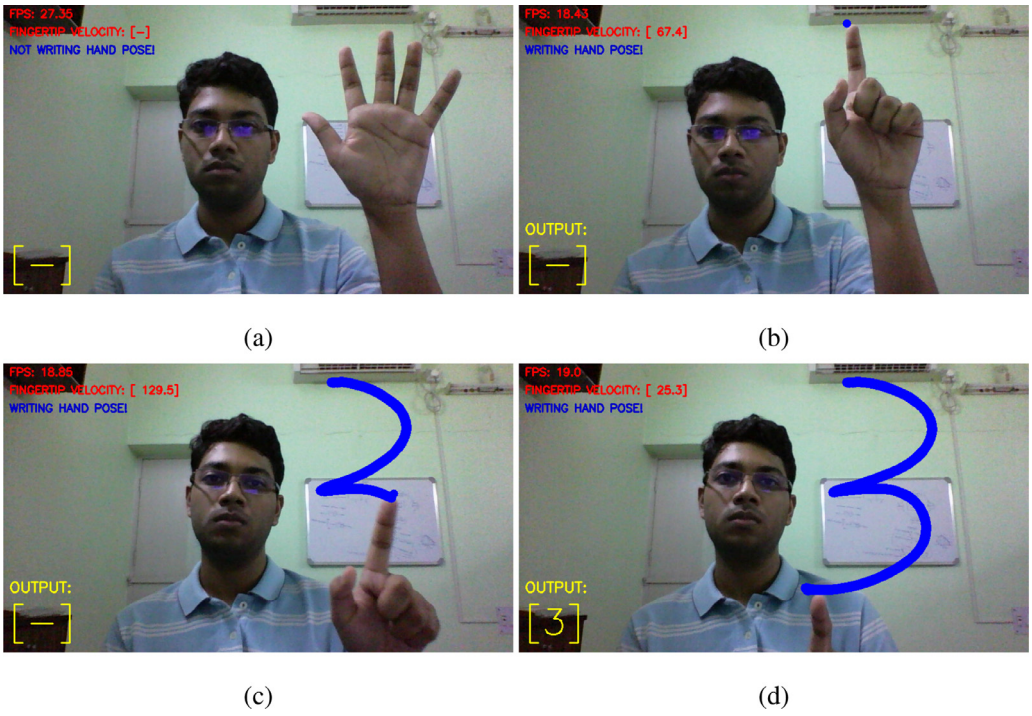


Fig. 9. Different stages of air-writing using the proposed framework. (a) Non-writing hand pose, therefore, fingertip detection does not occur. (b) User starts writing, writing hand pose is detected. The fingertip is detected and tracked over successive frames. (c) User is in the process of writing a character. Character recognition does not occur since the termination criterion is not satisfied. (d) Fingertip velocity is less than the threshold. Therefore, the termination criterion is satisfied, and the character '3' is recognized.

Table 1
Hand tracking performance on the test set in terms of overlap (IoU) and precision following the OTB-2013 benchmark. The proposed and **best** results are highlighted for each column.

Tracker	OPE		TRE		Speed (fps)
	IoU	Precision	IoU	Precision	
Proposed (KCF-based)	58.7	71.6	60.4	73.5	**27.1**
TLD (Kalal et al., 2012)	**60.2**	**74.1**	**62.5**	**76.3**	14.5
MIL (Babenko et al., 2011)	48.7	58.4	49.7	58.9	16.1

and MIL (Babenko, Yang, & Belongie, 2011), initialization in each case being done with the detected hand regions using FRCNN. As pointed out earlier re-initialization of the tracker is done at an interval of 50 frames to enable long-term tracking. We test the hand tracking performance for 1200 video sequences from our air-writing dataset.

We use the OTB-2013 (Wu, Lim, & Yang, 2013) benchmark to evaluate our hand tracking results. The OTB-2013 benchmark considers the average per-frame success rate at different values of thresholds. For a particular frame, if the intersection-over-union (IoU) between the estimate produced by a tracker and the ground-truth is greater than a certain threshold, the tracker is successful in that frame. For comparing different trackers, we use the area under the curve of success rates for different values of the threshold. The overlap (IoU) and precision scores for OPE (one pass evaluation) and TRE (temporal robustness evaluation) have been reported in Table 1. In OPE, the tracker is run once on each video sequence, from the start to the end, while in TRE, the tracker is started from twenty different starting points, and run until the end from each.

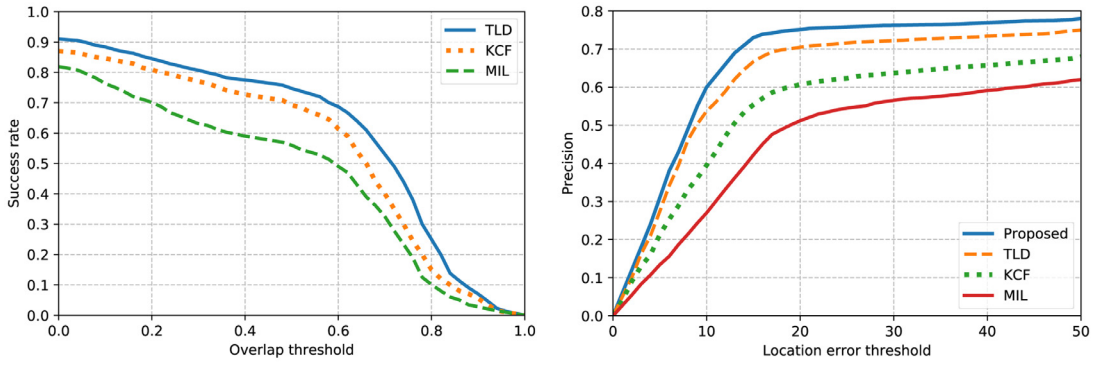
While MIL clearly suffers from the drifting problem, TLD gives superior tracking performance in terms of precision, but has a very poor frame rate. Experiments on the test set reveal that Faster R-CNN based hand detection followed by KCF tracking gives the best compromise between precision of tracking and frame rate, and therefore we use it for hand tracking. Fig. 10a shows the summarized hand tracking performance.

4.3. Fingertip detection and tracking

A test set of 1200 video sequences from our air-writing dataset has been used for evaluation of fingertip detection and tracking performance. For fingertip detection, the distance weighting parameter γ, referred to in Section 3.2, is tuned for maximum detection accuracy. The value of γ is varied in the range [1, 5] with steps of 0.5, the maximum detection accuracy occurring at 2.5.

For fingertip tracking, we use the precision curve proposed in Henriques et al. (2015) as the evaluation metric. A frame may be considered correctly tracked if the predicted target center is within a distance threshold of the ground truth. Precision curves show the percentage of correctly tracked frames for a range of distance thresholds. This is particularly suitable for our fingertip tracking application since the fingertip is best represented by a point and not a bounding box.

A tracker having higher precision at low thresholds is more accurate, while a lower precision at high thresholds implies that the target is lost. When a representative precision score is needed, we choose 15 pixels as the threshold instead of the standard practice of choosing 20 pixels (Henriques et al., 2015), since the finger-



(a) Success plots for OPE. (b) Precision plots for OPE.

Fig. 10. Comparison of tracking performances. (a) Hand tracking. (b) Fingertip tracking.

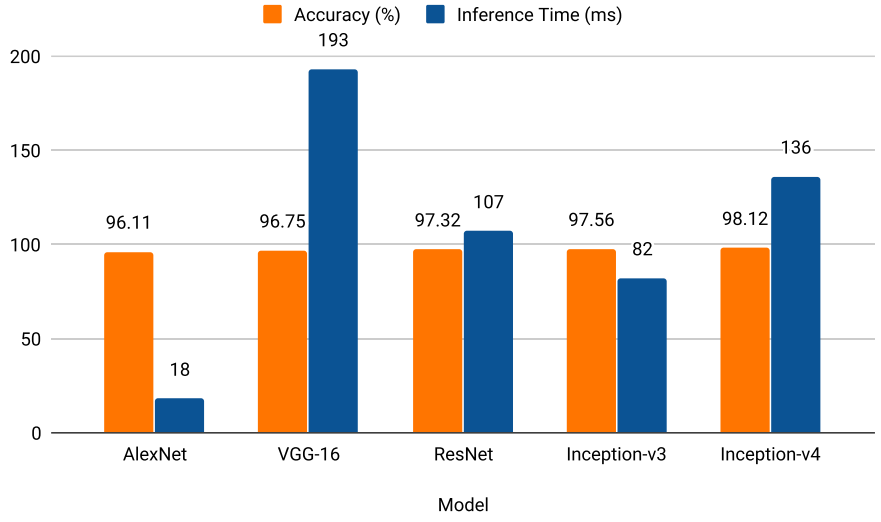


Fig. 11. Comparative performance analysis of state-of-the-art CNN architectures on air-writing character recognition in terms of accuracy and inference time.

Table 2
Fingertip tracking performance on the test set in terms of mean precision (15 px.). The proposed and **best** results are highlighted for each column.

Tracker	Mean Precision (15 px.)	Speed (fps)
Proposed (Tracking-by-detection)	**73.1**	18.5
KCF (Henriques et al., 2015)	55.4	**26.4**
TLD (Kalal et al., 2012)	66.7	10.6
MIL (Babenko et al., 2011)	42.4	12.1

tip tracking performance largely determines the resulting character trajectory, and hence the lower tolerance. The precision scores have been reported in Table 2. The results clearly show the superior performance of the proposed fingertip tracking algorithm over state-of-the-art trackers. Fig. 10b shows the summarized fingertip tracking performance.

4.4. Air-writing character recognition

For character recognition experiments, the entire air-writing dataset consisting of 1800 video sequences is used as the test set. We analysed the performance of several state-of-the-art CNN architectures such as AlexNet, VGG-16, ResNet, Inception-v3 and Inception-v4 in terms of classification accuracy as well as inference time of the trained model. Each CNN architecture is pre-trained on the EMNIST dataset and evaluated on our test set. Fig. 11 shows a comparison of the accuracies and inference times for all the above architectures. Inception-v4 offers the best performance in terms of accuracy while AlexNet has the least inference time. However, it is important to note that the accuracy of AlexNet is only slightly lower than Inception-v4 (a difference of 2.01%) while its inference time is less the one-seventh of Inception-v4. This clearly shows that the AlexNet architecture is the best choice for our real-time application since it offers the best trade-off between accuracy and inference time.

The character recognition model based on AlexNet architecture and pre-trained on the EMNIST dataset gives an air-writing character recognition accuracy of 96.11% on the test set. The normalized[2] confusion matrix for character recognition results in Fig. 12 shows that most characters have been correctly recognized, but some characters having similar shape and appearance were confused by the system, such as *'1'-'7', '3'-'8', 'a'-'d', 'g'-'q', 'j'-'i', 'l'-'1', 'n'-'m', 'o'-'0', 'r'-'n', 'w'-'v'*. Actually it has been found that these similar looking characters are often confused by humans too. The erroneous character recognition results might be improved by better models for handwritten character recognition using context such as words to decide the correct character.

Our experiments using the proposed air-writing recognition system revealed that there is a relationship between the speed of writing and the overall accuracy. We perform an exhaustive quantitative study to understand this effect of writing speed on overall accuracy. For this experiment, the actual speed of the fin-

[2] Normalization is done by dividing each row of the original confusion matrix by its class size 50.

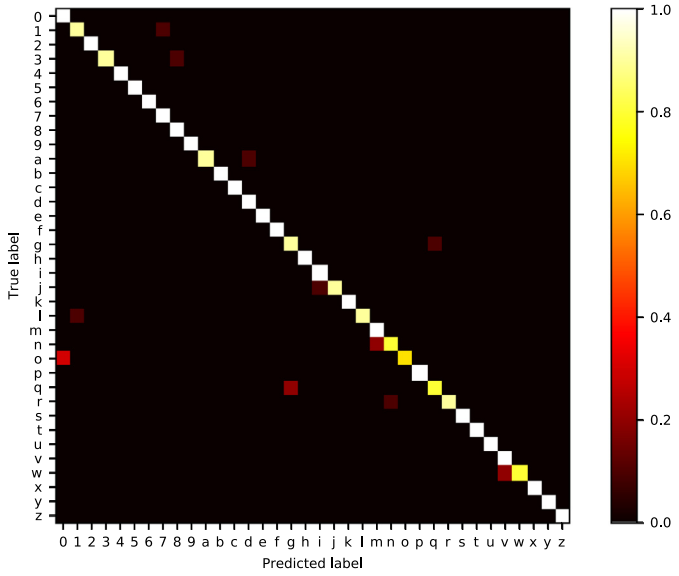


Fig. 12. Normalized confusion matrix for air-writing character recognition on 1800 test videos (36 character classes and 50 samples per class) using the proposed framework.

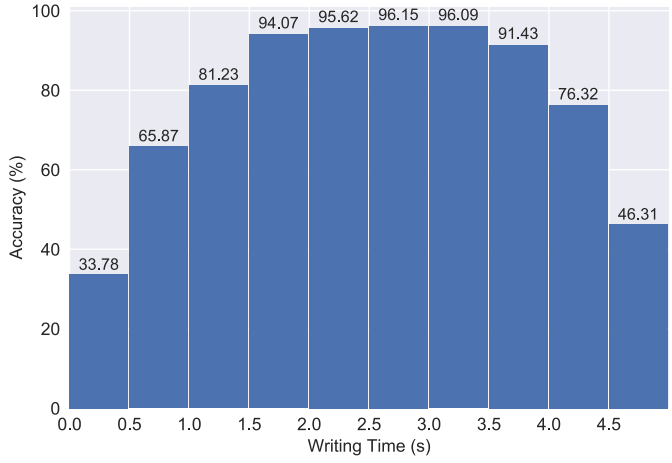


Fig. 13. Quantitative analysis of the effect of writing speed on overall accuracy.

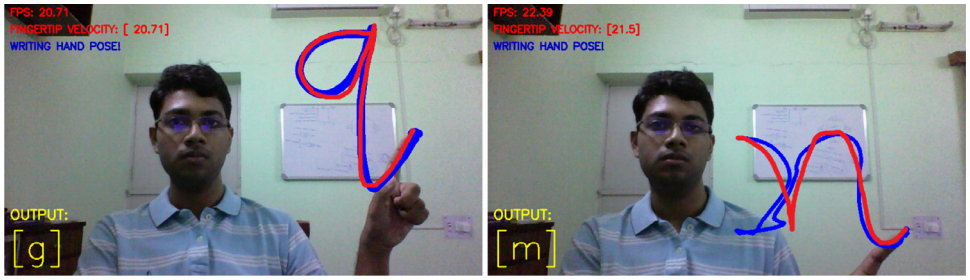
gertip is needed rather than the fingertip velocity obtained in Section 3.3.2 (because our algorithm for calculation of fingertip velocity depends on fingertip detection, and might be affected by very fast movement of the fingertip). However, obtaining the actual speed of the fingertip requires the use of additional sensors and is cumbersome. Instead, we use the time for writing a character (in seconds) as a measure of the writing speed. As illustrated in Fig. 13, we consider 10 equally sized bins (each of size 0.5 s) for writing time in the range [0.0, 5.0] s. For example, the first bin corresponds to writing time in the range [0.0, 0.5] s, the second bin to [0.5, 1.0] s, and so on. We record 360 air-writing character trajectories (10 samples each of 36 characters) from a single person for this study. For each character, we collect 10 samples from the person, each sample belonging to one of the bins. Therefore, we obtain 36 characters for each bin of writing time, and calculate the character recognition accuracy for each bin. Fig. 13 shows a comparison of accuracies of all bins of writing time. It is clear that the overall accuracy attains a maximum value for an intermediate range of speed (corresponding to writing time in the range 1.5 to 3.5 s) while it is low for very small values of speed (since the fingertip velocity drops lower than the threshold velocity leading to premature termination and erroneous character trajectory recognition) as well as for very high values of speed (since the fingertip detection fails in some of the frames due to extremely fast hand motion). It might also be noted that the maximum accuracy occurs when the person writes at natural speed, which corresponds to a writing time of about 2.5 s.

It is important to note that the frame rate displayed on the top left corner of each output image is the instantaneous frame rate for that particular frame. These instantaneous values of frame rates are averaged over the time interval of the air-writing gesture (including the inference time for character recognition) to get the average frame rate for a single video. Finally, we aggregate frame rates of all the test videos to get the frame rate values shown in Table 2. Therefore, the frame rate of the proposed system (18.5 fps) incorporates character recognition inference time as well, and demonstrates its real-time performance.

5. Conclusion

In this paper, we presented a new framework for the recognition of mid-air finger writing using web-cam video as input. We proposed a new writing hand pose detection algorithm for the initialization of air-writing. Further, we used a novel signature function called distance-weighted curvature entropy for robust fingertip detection and tracking. Finally, a fingertip velocity based termination criterion was used as a delimiter to mark the completion of the air-writing gesture. Extensive experiments on our air-writing dataset revealed the superior performance of the proposed fingertip detection and tracking approach over state-of-the-art trackers while achieving real-time performance in terms of frame rate. Character recognition results are impressive as well.

The proposed air-writing recognition framework can find applications in HCI as a virtual touch-less text entry interface. A key application may be in smart home automation for gesture-controlled smart home devices. This is analogous to home automation hubs such as the Amazon Echo, which uses voice commands to control



(a) (b)

Fig. 14. Failure cases of the proposed air-writing recognition system. (a) Failure in character recognition. (b) Failure in fingertip detection and tracking. The tracked fingertip trajectory is shown in blue color and the ground truth trajectory is shown in red color.

smart home devices. Instead, our framework (with suitable hardware implementation) can be used to take a fingertip trajectory based visual command (using a smart camera) as input to perform a particular task, based on the visual command or keyword. For e.g., a keyword *'b5'* may be used to turn on the fifth light of the bedroom.

Some failure cases however exist for the proposed air-writing recognition system, as depicted in Fig. 14. In Fig. 14a, it is clearly seen that the failure results due to misclassification by the character recognition model. The failure case in Fig. 14b is however more subtle, as it results from poor fingertip detection and tracking. This can be accounted for by the fact that, when the hand is not completely inside the frame, the hand detector performance deteriorates. This in turn degrades the fingertip detection and tracking performance. Moreover, when the hand is largely occluded, the geometrical features of the hand used for fingertip detection are also no longer valid.

Another limitation of the proposed air-writing recognition framework is, it is suited mainly for gestures with the finger and palm lying in the same plane, with the palm held vertically or near-vertically. For example, our fingertip detection algorithm performs poorly when the fingertip points to the camera. This constraint is imposed due to the typical RGB web-cams that do not include depth information. As a future work, we aim to incorporate recent advances in 3D hand pose estimation (Panteleris, Oikonomidis, & Argyros, 2018; Zimmermann & Brox, 2017) and hand depth map estimation (Nicodemou, Oikonomidis, Tzimiropoulos, & Argyros, 2018) from monocular RGB images into our fingertip detection algorithm, to be able to capture finger movements in the 3D space. This in turn would lead to more robust fingertip detection and air-writing recognition performance. Moreover, we would like to extend the proposed framework to utilize the spatio-temporal as well as motion features of the air-writing trajectories for the recognition of characters. We would also like to extend our framework to the recognition of words as well as signatures, which can have potential application in touch-less and marker-less biometric authentication systems.

Declaration of Competing Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Credit authorship contribution statement

Sohom Mukherjee: Conceptualization, Methodology, Software, Investigation, Data curation, Writing - original draft, Visualization. **Sk. Arif Ahmed:** Conceptualization, Methodology, Validation, Data curation, Visualization. **Debi Prosad Dogra:** Conceptualization, Validation, Data curation, Writing - review & editing, Supervision. **Samarjit Kar:** Formal analysis, Resources, Writing - review & editing, Supervision, Project administration. **Partha Pratim Roy:** Formal analysis, Writing - review & editing, Supervision, Project administration.

Acknowledgements

The authors would like to thank the Editor and the anonymous reviewers for their constructive and valuable suggestions to enhance the quality of the manuscript.

Funding

This research did not receive any grant from funding agencies in the public, commercial, or not-for-profit sectors.

Informed consent

Informed consent was obtained from all individual participants included in the study.

References

Amma, C., Georgi, M., & Schultz, T. (2012). Airwriting: Hands-free mobile text input by spotting and continuous recognition of 3d-space handwriting with inertial sensors. In *Wearable computers (ISWC), 2012 16th international symposium on* (pp. 52–59). IEEE.

Babenko, B., Yang, M.-H., & Belongie, S. (2011). Robust object tracking with online multiple instance learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 33*(8), 1619–1632.

Bambach, S., Lee, S., Crandall, D. J., & Yu, C. (2015). Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions. In *Computer vision (ICCV), 2015 IEEE international conference on* (pp. 1949–1957). IEEE.

Behera, S. K., Dogra, D. P., & Roy, P. P. (2017). Analysis of 3d signatures recorded using leap motion sensor. *Multimedia Tools and Applications*, 1–26.

Behera, S. K., Dogra, D. P., & Roy, P. P. (2018). Fast recognition and verification of 3d air signatures using convex hulls. *Expert Systems with Applications, 100*, 106–119.

Canziani, A., Paszke, A., & Culurciello, E. (2016). An analysis of deep neural network models for practical applications. arXiv:1605.07678.

Chang, H. J., Garcia-Hernando, G., Tang, D., & Kim, T.-K. (2016). Spatio-temporal hough forest for efficient detection–localisation–recognition of fingerwriting in egocentric camera. *Computer Vision and Image Understanding, 148*, 87–96.

Chen, M., AlRegib, G., & Juang, B.-H. (2016). Air-writing recognition part ii: Detection and recognition of writing activity in continuous stream of motion data. *IEEE Transactions on Human-Machine Systems, 46*(3), 436–444.

Ciregan, D., Meier, U., & Schmidhuber, J. (2012). Multi-column deep neural networks for image classification. In *Computer vision and pattern recognition (CVPR), 2012 IEEE conference on* (pp. 3642–3649). IEEE.

Cireşan, D., & Meier, U. (2015). Multi-column deep neural networks for offline handwritten chinese character classification. In *Neural networks (IJCNN), 2015 international joint conference on* (pp. 1–6). IEEE.

Cohen, G., Afshar, S., Tapson, J., & van Schaik, A. (2017). Emnist: An extension of mnist to handwritten letters. arXiv:1702.05373.

Deng, X., Zhang, Y., Yang, S., Tan, P., Chang, L., Yuan, Y., & Wang, H. (2018). Joint hand detection and rotation estimation using cnn. *IEEE Transactions on Image Processing, 27*(4), 1888–1900.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. (2007). The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

Feldman, J., & Singh, M. (2005). Information along contours and object boundaries.. *Psychological Review, 112*(1), 243.

Gan, J., & Wang, W. (2017). In-air handwritten english word recognition using attention recurrent translator. *Neural Computing and Applications*, 1–18.

Girshick, R. (2015). Fast r-cnn. In *2015 IEEE international conference on computer vision (ICCV)* (pp. 1440–1448). doi:10.1109/ICCV.2015.169.

Graham, B. (2013). Sparse arrays of signatures for online character recognition. arXiv:1308.0371.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).

Henriques, J. F., Caseiro, R., Martins, P., & Batista, J. (2015). High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 37*(3), 583–596.

Huang, Y., Liu, X., Zhang, X., & Jin, L. (2016). A pointing gesture based egocentric interaction system: Dataset, approach and application. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops* (pp. 16–23).

Kalal, Z., Mikolajczyk, K., & Matas, J. (2012). Tracking-learning-detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 34*(7), 1409–1422.

Kane, L., & Khanna, P. (2017). Vision-based mid-air unistroke character input using polar signatures. *IEEE Transactions on Human-Machine Systems, 47*(6), 1077–1088.

Khan, A. U., & Borji, A. (2018). Analysis of hand segmentation in the wild. arXiv:1803.03317.

Kolsch, M., & Turk, M. (2004). Fast 2d hand tracking with flocks of features and multi-cue integration. *Computer vision and pattern recognition workshop, 2004. CVPRW'04. conference on*. IEEE. 158–158.

Krejov, P., & Bowden, R. (2013). Multi-touchless: Real-time fingertip detection and tracking using geodesic maxima. In *Automatic face and gesture recognition (FG), 2013 10th IEEE international conference and workshops on* (pp. 1–7). IEEE.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097–1105).

Kurata, T., Okuma, T., Kourogi, M., & Sakaue, K. (2001). The hand mouse: Gmm hand-color classification and mean shift tracking. In *Recognition, analysis, and tracking of faces and gestures in real-time systems, 2001. proceedings. IEEE ICCV workshop on* (pp. 119–124). IEEE.

de La Gorce, M., Fleet, D. J., & Paragios, N. (2011). Model-based 3d hand pose estimation from monocular video. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 33*(9), 1793–1805.

Lai, S., Jin, L., & Yang, W. (2017). Toward high-performance online hccr: A cnn approach with dropdistortion, path signature and spatial stochastic max-pooling. *Pattern Recognition Letters, 89*, 60–66.

Lee, T., & Hollerer, T. (2007). Handy ar: Markerless inspection of augmented reality objects using fingertip tracking. In *Wearable computers, 2007 11th IEEE international symposium on* (pp. 83–90). IEEE.

Li, C., & Kitani, K. M. (2013). Pixel-level hand detection in ego-centric videos. In *Computer vision and pattern recognition (CVPR), 2013 IEEE conference on* (pp. 3570–3577). IEEE.

Liang, H., Yuan, J., & Thalmann, D. (2012). 3d fingertip and palm tracking in depth image sequences. In *Proceedings of the 20th ACM international conference on multimedia* (pp. 785–788). ACM.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., … Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision* (pp. 740–755). Springer.

Mayol, W. W., Davison, A. J., Tordoff, B. J., Molton, N., & Murray, D. W. (2004). Interaction between hand and wearable camera in 2d and 3d environments. In *Proc. british machine vision conference: 2* (p. 5).

Misra, S., Singha, J., & Laskar, R. (2017). Vision-based hand gesture recognition of alphabets, numbers, arithmetic operators and ascii characters in order to develop a virtual text-entry interface system. *Neural Computing and Applications*, 1–19.

Mittal, A., Zisserman, A., & Torr, P. H. (2011). Hand detection using multiple proposals.. In *BMVC* (pp. 1–11).

Molchanov, P., Gupta, S., Kim, K., & Kautz, J. (2015). Hand gesture recognition with 3d convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops* (pp. 1–7).

Nicodemou, V. C., Oikonomidis, I., Tzimiropoulos, G., & Argyros, A. (2018). Learning to infer the depth map of a hand from its color image. arXiv:1812.02486.

Ohn-Bar, E., & Trivedi, M. M. (2014). Hand gesture recognition in real time for automotive interfaces: A multimodal vision-based approach and evaluations. *IEEE Transactions on Intelligent Transportation Systems, 15*(6), 2368–2377.

Oka, K., Sato, Y., & Koike, H. (2002). Real-time fingertip tracking and gesture recognition. *IEEE Computer Graphics and Applications, 22*(6), 64–71.

Panteleris, P., Oikonomidis, I., & Argyros, A. (2018). Using a single rgb frame for real time 3d hand pose estimation in the wild. In *2018 IEEE winter conference on applications of computer vision (WACV)* (pp. 436–445). IEEE.

Phung, S. L., Bouzerdoum, A., & Chai, D. (2005). Skin segmentation using color pixel classification: analysis and comparison. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 27*(1), 148–154.

Rautaray, S. S., & Agrawal, A. (2015). Vision based hand gesture recognition for human computer interaction: a survey. *Artificial Intelligence Review, 43*(1), 1–54.

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems* (pp. 91–99).

Rohrbach, M., Rohrbach, A., Regneri, M., Amin, S., Andriluka, M., Pinkal, M., & Schiele, B. (2016). Recognizing fine-grained and composite activities using hand-centric features and script data. *International Journal of Computer Vision, 119*(3), 346–373.

Roy, K., Mohanty, A., & Sahay, R. R. (2017). Deep learning based hand detection in cluttered environment using skin segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 640–649).

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556.

Stenger, B., Thayananthan, A., Torr, P. H., & Cipolla, R. (2006). Model-based hand tracking using a hierarchical bayesian filter. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 28*(9), 1372–1384.

Suau, X., Alcoverro, M., López-Méndez, A., Ruiz-Hidalgo, J., & Casas, J. R. (2014). Real-time fingertip localization conditioned on hand gesture classification. *Image and Vision Computing, 32*(8), 522–532.

Suzuki, S., et al. (1985). Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing, 30*(1), 32–46.

Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. A. (2017). Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI: 4* (p. 12).

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2818–2826).

Tang, D., Chang, H. J., Tejani, A., & Kim, T.-K. (2017). Latent regression forest: structured estimation of 3d hand poses. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 39*(7), 1374–1387.

Wu, G., & Kang, W. (2016). Robust fingertip detection in a complex environment. *IEEE Transactions on Multimedia, 18*(6), 978–987.

Wu, W., Li, C., Cheng, Z., Zhang, X., & Jin, L. (2017). Yolse: Egocentric fingertip detection from single rgb images. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 623–630).

Wu, Y., Lim, J., & Yang, M.-H. (2013). Online object tracking: A benchmark. In *Computer vision and pattern recognition (CVPR), 2013 IEEE conference on* (pp. 2411–2418). IEEE.

Xiao, X., Yang, Y., Ahmad, T., Jin, L., & Chang, T. (2017). Design of a very compact cnn classifier for online handwritten chinese character recognition using dropweight and global pooling. arXiv:1705.05207.

Yang, W., Jin, L., Tao, D., Xie, Z., & Feng, Z. (2016). Dropsample: A new training method to enhance deep convolutional neural networks for large-scale unconstrained handwritten chinese character recognition. *Pattern Recognition, 58*, 190–203.

Yin, F., Wang, Q.-F., Zhang, X.-Y., & Liu, C.-L. (2013). Icdar 2013 chinese handwriting recognition competition. In *Document analysis and recognition (ICDAR), 2013 12th international conference on* (pp. 1464–1470). IEEE.

Zhang, X., Ye, Z., Jin, L., Feng, Z., & Xu, S. (2013). A new writing experience: Finger writing in the air using a kinect sensor. *IEEE MultiMedia, 20*(4), 85–93.

Zhang, Y., Cao, C., Cheng, J., & Lu, H. (2018). Egogesture: A new dataset and benchmark for egocentric hand gesture recognition. *IEEE Transactions on Multimedia*.

Zimmermann, C., & Brox, T. (2017). Learning to estimate 3d hand pose from single rgb images. In *International conference on computer vision: 1* (p. 3).

Zivkovic, Z. (2004). Improved adaptive gaussian mixture model for background subtraction. In *Pattern recognition, 2004. ICPR 2004. proceedings of the 17th international conference on: 2* (pp. 28–31). IEEE.