DrawInAir: A Lightweight Gestural Interface Based on Fingertip Regression

Gaurav Garg(✉), Srinidhi Hegde, Ramakrishna Perla, Varun Jain, Lovekesh Vig, and Ramya Hebbalaguppe

TCS Research, Gurgaon, India
{ga.gaurav,sri.hegde,r.perla,varun.in,lovekesh.vig, ramya.hebbalaguppe}@tcs.com

Abstract. Hand gestures form a natural way of interaction on Head-Mounted Devices (HMDs) and smartphones. HMDs such as the Microsoft HoloLens and ARCore/ARKit platform enabled smartphones are expensive and are equipped with powerful processors and sensors such as multiple cameras, depth and IR sensors to process hand gestures. To enable mass market reach via inexpensive Augmented Reality (AR) headsets without built-in depth or IR sensors, we propose a real-time, in-air gestural framework that works on monocular RGB input, termed, *DrawInAir*. *DrawInAir* uses fingertip for writing in air analogous to a pen on paper. The major challenge in training egocentric gesture recognition models is in obtaining sufficient labeled data for end-to-end learning. Thus, we design a cascade of networks, consisting of a CNN with *differentiable spatial to numerical transform* (DSNT) layer, for fingertip regression, followed by a *Bidirectional Long Short-Term Memory* (Bi-LSTM), for a real-time pointing hand gesture classification. We highlight how a model, that is separately trained to regress fingertip in conjunction with a classifier trained on limited classification data, would perform better over *end-to-end* models. We also propose a dataset of 10 egocentric pointing gestures designed for AR applications for testing our model. We show that the framework takes 1.73 s to run end-to-end and has a low memory footprint of 14 MB while achieving an accuracy of 88.0% on egocentric video dataset.

Keywords: Egocentric gestures · Coordinate regression
Augmented reality

1 Introduction

Most popular interfaces in HMDs/Smartphones are speech and gestures. However, the accuracy of speech recognition tends to suffer in an industrial or an outdoor setting due to ambient noise [1]. To this end, gestural interfaces are

G. Garg, S. Hegde, R. Perla and V. Jain—Contributed equally.

L. Leal-Taixé and S. Roth (Eds.): ECCV 2018 Workshops, LNCS 11134, pp. 229–240, 2019.
https://doi.org/10.1007/978-3-030-11024-6_15

preferred in the areas of human-computer interaction and human-robot interaction [1–3] as one does not require sophisticated skills to communicate, and they enable wider accessibility without bias on speech accents. However, real-time gesture tracking and recognition in First Person View (FPV) for wearable devices is still a challenging task (refer Fig. 1). Expensive AR devices such as the Microsoft *HoloLens*, *Daqri* and *Meta* Glasses are equipped with gestural interface powered by a variety of on-board sensors including a depth sensor and customized processors making the product expensive and unaffordable for mass adoption.

Fig. 1. Users performing egocentric *in-air* gestures in complex backgrounds such as outdoor environments, reflective backgrounds and different lighting conditions. Note: Variations in the speed of gestures and gesture trajectories between individuals are some of the issues that affect *in-air* hand gesture recognition [4].

In this paper we propose a novel gestural framework without the need of specialized hardware that would provide mass accessibility of gestural interfaces to the most affordable *video-see-through* HMDs such as *Wearality Sky* (50 USD) and *Google Cardboard*[1] (15 USD). These devices provide immersive AR experiences with the help of stereo rendering of the smartphone camera feed. The immediate applications are industrial inspection and repair, tele-presence, and FPV photography. *Google Cardboard* still employs primitive modes of user interaction, that is magnetic trigger and conductive lever, and any development is restricted to the hardware and sensors available on a smartphone. Hence, we aim to design pointing gesture based user interaction for frugal HMDs/smartphones.

3D CNNs and RNNs are found to be effective in analysis of egocentric gestures. However, these networks are highly reliant on the large scale video dataset and pixel-level depth information while training, often hindering real-time performance. In this work, we present a neural network architecture comprising of a base CNN and a *differentiable spatial to numerical transform* (DSNT) [18] layer followed by a Bidirectional Long Short-Term Memory (Bi-LSTM). The layer transforms the heatmap from CNN, that is rich in spatial information, to output spatial location of fingertip. The Bi-LSTM effectively captures the dynamic motion of user gesture that aids in classification. Feeding the fingertip keypoints to the Bi-LSTM, as opposed to traditional approaches of inputting

[1] https://vr.google.com/cardboard/.

featuremaps or images, reduces the computational cost in classification. Our key contributions are:

1. We propose *DrawInAir*, a neural network architecture, consisting of a base CNN and a DSNT network followed by a Bi-LSTM, for efficient classification of user gestures. It works in real-time, uses only RGB image sequence with no depth information, and can be ported on mobile devices due to low memory footprint.
2. ***EgoGestAR***: a dataset of spatio-temporal sequences representing 10 gestures suitable for AR applications. We have published the dataset online at: https://github.com/varunj/EgoGestAR.

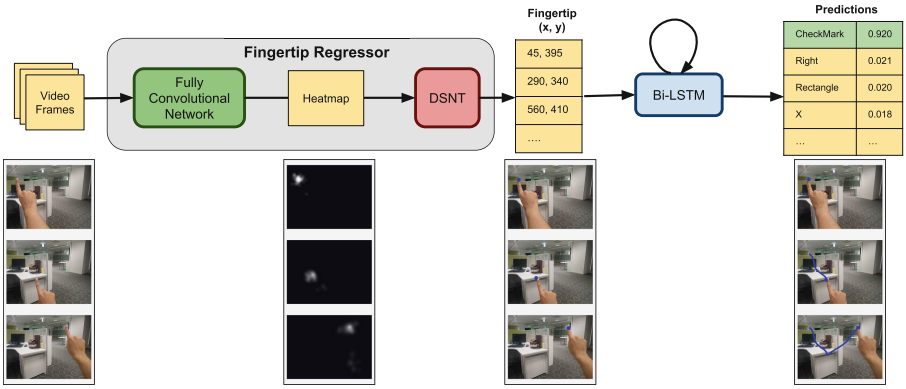

Fig. 2. *DrawInAir* framework. *DrawInAir* comprises a *Fingertip Regressor* module which accurately localizes the fingertip (the fingertip is analogous to a pen-tip in HCI) and a Bi-LSTM network is used for classification of fingertip detections on subsequent frames into different gestures (Images at the bottom show input/output at different stages).

2 Related Work

Despite being intuitive and natural, gestures are prone to inherent ambiguity which makes them a topic of interest to the research community [5]. Most of the early gesture recognition frameworks involve either (i) low-level image analysis such as detection of contours, texture, segmentation, histograms [6] or (ii) vision approaches such as feature extraction, object detection followed by tracking, and classification [7].

Recently using CNNs for object classification and detection has shown to give promising results. Huang *et al.* [8] proposed bi-level cascade CNNs approach for hand and key point detection in egocentric view using *HSV* color space information. Tompson *et al.* [9] proposed a pipeline for real-time pose recovery of human hands from a single depth image using a CNN. Coming to the gesture classification methods, in [10], Liu *et al.* presented two real-time third-person hand gesture recognition systems - (i) utilizing the stereo camera hardware setup with

DTW classifier and (ii) using dual-modality sensor fusion system with HMM classifier. Dardas *et al.* [11] presented a system for hand gesture recognition via bag-of-features and multi class Support Vector Machines (SVM). The Randomized Decision Forest classifier has also been explored for hand segmentation [9] and hand pose estimation [12]. Jain *et al.* [?] have shown the efficacy of using LSTM networks for the classification of 3-dimensional gestures.

In a recent work, Hegde *et al.* [1] discussed simple hand swipe gestures for Google Cardboard in egocentric view using GMM based modeling of skin pixel data. Further, this work was extended in [13] for accurate hand swipe classification. Implementing such ad-hoc recognizers is very challenging when the number and type of gestures increase. This is due to high inter class similarity among the gesture classes [14]. Unlike the works [15–17], which use RGB-D inputs to recognize multi pose gestures and occluding fingers in egocentric view, our proposed framework focuses on computationally efficient pointing pose-based gesture recognition using just RGB data.

In our work, we specifically deal with pointing finger gestures which requires detecting fingertip coordinates. We are inspired by the recent work of Nibali *et al.* [18] which proposed DSNT layer for numerical coordinate regression for estimating human body joints position. But they use a fully convolutional networks (FCN), a stacked hourglass network and other complex networks for generating heatmaps which makes their method slow in comparison to ours.

3 DrawInAir

A recent trend in the deep learning community has been to develop *end-to-end* models that learn several intermediate tasks simultaneously. While this has obvious benefits for learning joint tasks like object detection, regression and classification, it is reliant on the presence of sufficient labelled data to learn all the tasks in a pipeline.

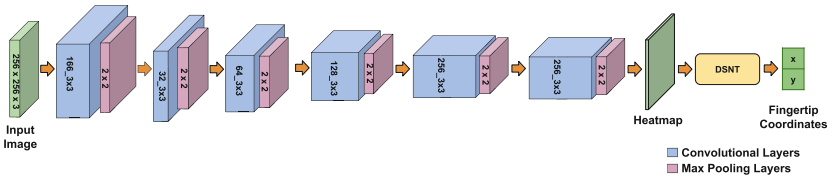

Fig. 3. Overview of our proposed *fingertip regressor* architecture for fingertip localization. The input to the network is $3 \times 256 \times 256$ sized RGB images. The network consists of 6 convolutional blocks, each with different convolutional layers followed by a max-pooling layer. Then we have a convolutional layer to output a heatmap which is input to DSNT. Finally, we get 2 coordinates denoting fingertip spatial location.

We, hence, propose a pointing hand gestural framework in egocentric view with limited labelled classification data. We focus on classifying the *point gesture*

motion patterns into different gestures. Figure 2 shows the blocks which are: (a) *the Fingertip Regressor* that takes an RGB input image and accurately localizes the fingertip, (b) a Bi-LSTM network for classification of the fingertip detection on subsequent frames into different gestures.

We assume that the subjects are stationary while performing gestures to interact with the device. Slight errors introduced due to the head movement can be rectified by post-processing the *Fingertip Regressor* output and by the Bi-LSTM network used in classification. Bi-LSTM also has the ability to handle unexpected impulses/peaks arising in gesture pattern due to false detections or fingertip localizations for short duration.

3.1 Fingertip Regression

Estimating human pose by localizing human joints has been an important study in computer vision. Toshev *et al.* [19] propose *DeepPose*, which formulates the human pose estimation problem as a CNN based regression over body joints. In a similar context, we employ a CNN architecture followed by DSNT layer [18] (refer Fig. 3) for localizing fingertip by regressing over the coordinates, (x, y), of the fingertip.

Differentiable Spatial to Numerical Transform (DSNT): The proposed architecture consists of a CNN that produces a heatmap, $\boldsymbol{Z}$, containing the spatial information of fingertip location. The heatmap is passed on to a *differentiable spatial to numerical transform* (DSNT) layer which transforms the heatmap to numerical coordinates of the fingertip location. The DSNT layer has no trainable parameters, preserves the differentiability and generalizes spatially, hence allowing the entire network to learn by back-propagation. DSNT normalizes the heatmap $\boldsymbol{Z}$ to $\hat{\boldsymbol{Z}}$ such that all the elements of normalized heatmap are non-negative and sum to one. After normalization, the heatmap coordinates are scaled such that the top-left corner of the heatmap is at $(-1, -1)$ and bottom-right is at $(1, 1)$. This is followed by outputting the expected coordinates in the scaled coordinate system with normalized heatmap, $\hat{\boldsymbol{Z}}$, as probability distribution map.

For training the network we use Euclidean loss as follows:

$$\mathcal{L}(\hat{\boldsymbol{Z}}, \boldsymbol{p}) = \|\boldsymbol{p} - DSNT(\hat{\boldsymbol{Z}})\|_2 + \lambda \mathcal{L}_{reg}(\hat{\boldsymbol{Z}}) \tag{1}$$

where $\boldsymbol{p}$ is the ground truth coordinates and λ is a regularization constant. $DSNT(\hat{\boldsymbol{Z}})$ is the expected scaled coordinates that is produced by the DSNT layer. Nibali *et al.* [18], suggest different regularizers, $\mathcal{L}_{reg}$, for training the network. We find that using Kullback-Leibler divergence (KLD) as regularizer gave us the best results. Thus, we have $\mathcal{L}_{reg}$ as follows:

$$\mathcal{L}_{reg}(\hat{\boldsymbol{Z}}, \boldsymbol{p}) = KLD(\hat{\boldsymbol{Z}} \| \mathcal{N}(\boldsymbol{p}, \sigma_t^2)) \tag{2}$$

where σ_t^2 is a variance hyper-parameter of a target normal distribution, $\mathcal{N}$. This regularizer encourages the heatmap to resemble a isotropic target Gaussian distribution.

3.2 Gesture Classification

The localization network discussed in the previous section outputs the spatial location of the fingertip (x, y), which is then fed as an input to our gesture classification network. Since we use the gestures that have only pointing fingers, the classification task reduces to analyzing the motion of the fingertip. Thus, we input (x, y) coordinate instead of the entire frame to the network. Motivated by the effectiveness of LSTMs [20] in learning long-term dependencies of sequential data [21], we employ a Bi-LSTM [22] network for the classification of gestures. We found that Bi-LSTM performs better than LSTM for classification as it processes the sequence in both forward and reverse direction.

We found the raw fingertip coordinates from the *fingertip regressor* to be noisy. This is due to the relative motion of head and hand of the user in an egocentric setting. Thus, we applied smoothing operation on the sequence of fingertip points as an egocentric correction measure (refer Fig. 4). We used Savitzky-Golay filter [23] on the fingertip sequence with window size of 15 and polynomial order 1 yielding the best classification accuracies on applying this filter. This filter operates by increasing the signal-to-noise ratio without greatly distorting the signal. The entire framework is also adaptable to videos/live feeds with variable length frame sequences. This is particularly important as the length of gestures depends on the user performing it.

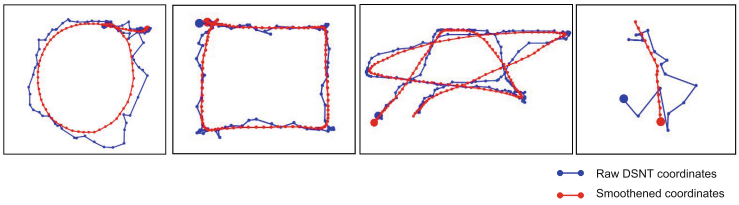

Fig. 4. Effect of smoothing for egocentric correction. (Left to right) Output of Savitzky-Golay filter [23] for samples of classes – Circle, Square, Star and Up respectively. The highlighted point in each gesture indicates the starting position of the gesture.

4 Datasets

4.1 Hand Dataset

We use the SCUT-Ego-Finger benchmark Dataset [8] for training the base CNN followed by DSNT layer model. Twenty four subjects in different environments (such as basketball field, canteen, teaching building, library, lake) contributed to the dataset to gather variations in illumination conditions, background and to address challenges such as variation in hand shape, hand color diversity, and motion blur. The dataset includes 93,729 frames with corresponding labels including hand candidate bounding boxes and index finger key point coordinates.

4.2 EgoGestAR Dataset

To train and evaluate the proposed Bi-LSTM architecture, we present ***EgoGestAR***: a spatio-temporal sequence dataset for AR wearables. The dataset includes spatial patterns representing 10 gestures and inspired by industrial applications, we divided the gestures patterns primarily into 3 categories. (a) 4 swipe gesture patterns (*Up, Down, Left, and Right*) for navigating/selecting user preferences in AR HMDs. (b) 2 gesture patterns (*Rectangle and Circle*) for RoI highlighting in user's FoV for tele-support applications. (c) 4 gesture patterns (*Checkmark: Yes, Caret: No, X: Delete, Star: Bookmark*) for evidence capture in inspection, maintenance and repair applications.

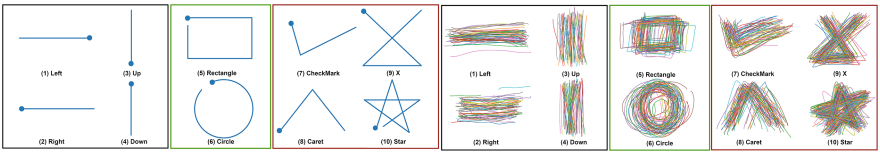

Fig. 5. EgoGestAR dataset: The first 3 columns show standard sequences shown to the users before the data collection and the last 3 columns (captured at a resolution of 640×480) depict the variations in the data samples. The highlighted point in each sequence indicates the starting position of the gesture.

We collected the data from 50 subjects in our research lab with ages in the range 21 to 50 with average age 27.8 years. The dataset consists of 2500 gesture patterns where each subject recorded 5 samples of each gesture. The gestures were recorded by mounting a 10.1 in. display HP Pro Tablet to a wall. The gesture pattern drawn by a user's index finger on a touch interface application with position sensing region was stored. The data was captured at a resolution of 640×480. Figure 5 describes the standard input sequences shown to the users before data collection and a sample subset of gestures from the dataset showing the variability introduced by the subjects. Detailed statistics of the EgoGestAR dataset is available at https://github.com/varunj/EgoGestAR.

5 Experiments and Results

Since the framework comprises of a cascade of two networks, we evaluate each network performance individually and then present the results of the entire pipeline. We use an 8 core Intel(R) Core(TM) i7-6820HQ CPU, 32 GB memory and an Nvidia Quadro M5000M GPU machine for experiments. The models are trained using Tensorflow v1.6.0.

5.1 Training

Fingertip Localization: We first train the *fingertip regressor* using the SCUT Ego-finger dataset (refer Sect. 4.1). Out of the 24 subjects in the dataset, we choose 17 subjects' data for training with a validation split of 70:30, and 7 subjects' data (24,155 images) for testing the networks. We use *Adam* optimizer with a learning rate of 6×10^{-5}. We set the hyper-parameters λ and σ_t to 1 and 4 respectively.

Classification: We then use *EgoGestAR* dataset (discussed in Sect. 3.2) for training and testing of the Bi-LSTM and also an LSTM network for classification. During training, we use 2000 gesture patterns in the training set. These patterns are fed as input to the *Bi-LSTM* layer consisting of 30 hidden units. The *forward* and *backward* outputs are multiplied before passing it to a fully connected layer with 10 output scores that correspond to each of the 10 gestures. We use a *softmax* activation function and *cross-entropy* loss for training the Bi-LSTM network. We train both the networks using *Adam* optimizer with learning rate of 0.001, a batch size of 32 and validation split of 80:20.

5.2 Performance Evaluation

Framework Evaluation: The average Euclidean loss in predicting the fingertip coordinates by the *fingertip regressor* is 1.147 on an input image of resolution 256×256. The mean absolute regression error is found to be 23.73 pixels for our approach. Table 1 presents comparison of the proposed LSTM and Bi-LSTM approach with DTW [10] and SVM [24]. We see that Bi-LSTM outperforms the

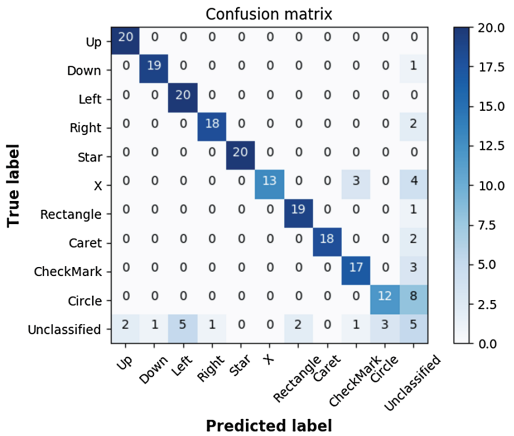

Fig. 6. The overall performance of our proposed framework on 240 egocentric videos (22 per class) captured using a smartphone based *Google Cardboard* head-mount. The gesture is detected when the predicted probability is more than 0.75. Accuracy of the model is 88%, ignoring unclassified class and 82.27% otherwise.

traditional approaches that are being used for similar classification tasks. Since the proposed approach is a series of different networks, the overall classification accuracy in real-time will vary depending on the performance of the earlier network used in the pipeline. Therefore, we evaluate the entire framework using 240 egocentric videos captured with a smartphone based Google Cardboard head-mount. Dataset and demos are available at https://ilab-ar.github.io/DrawInAir/. The overall framework achieved an accuracy of 88% on this dataset (as shown in Fig. 6).

Runtime and Memory Analysis: Table 2 shows the time profile of the proposed framework. The entire model has a very small memory footprint of 14 MB without compression and could be easily ported to mobile devices for testing.

Table 1. Performance of different classification methods on our proposed fingertip sequence dataset, EgoGestAR. Note that these results are observed on sequence data and not on hand gesture videos.

Method	Precision	Recall (Accuracy)	F_1 score
DTW [10]	0.763	0.749	0.756
SVM [24]	0.938	0.922	0.929
LSTM [20]	0.808	0.788	0.798
Bi-LSTM [22]	**0.967**	**0.966**	**0.966**

Table 2. Run-time analysis of different modules (with different inputs) of the framework. The input image resolution is 256×256 for the entire analysis. Note: the entire pipeline time is calculated starting from the first frame into the regressor to the prediction at the end of the entire video.

Module	Fingertip regressor (per frame)	Gesture classifier (per sequence)	Entire pipeline (per video)
Time (in s for unit input)	0.0096	0.0314	1.73

6 Discussion and Comparison

On deeper analysis, we observe that the *X (Del)* gesture is slightly correlated with the *CheckMark* since the difference in them is due to a triangle in the bottom of *X (Del)* gesture. Hence, due to variations in how users perform gestures and occlusion in users' hand, we observe a drop in their classification accuracies. Our framework is limited to a single finger in the user FoV and the accuracy drops if multiple fingers are present at roughly the same distance or on using any gesture different from pointing gesture. Figure 7 shows some cases, such as presence of multiple fingers (in case of reflection), where *DrawInAir* gives low accuracies. But our framework robustly detects and classifies fingertip of any of the fingers (even

if the user is wearing nail paint or has minor finger injuries) provided it is the only finger in the user FoV. Our framework is also robust to variations in starting position of the gestures in frame, hand sizes and skin colors. The framework can accommodate a number of pointing gestures as per the requirements of FPV application, making it generic for all touch-less interaction systems.

Fig. 7. Misclassified cases. Our framework fails to detect fingertip accurately in the cases of (a) reflective surfaces in the background, (b) near skin pixel background, and (c) very low illumination conditions.

Table 3. Analysis of gesture recognition accuracy and latency of various models against the proposed DrawInAir. We compared and evaluated all the end-to-end methods against ours on the 240 egocentric videos.

Method	Accuracy (%)	Time taken (in s)
Tsironi *et al.* [21]	32.14	0.76
VGG16 + LSTM [25]	58.37	**0.69**
C3D [26]	66.71	1.19
DrawInAir	**88.00**	1.73

We compared our framework against a few *end-to-end* baseline architectures used for video classification to highlight the importance of modular architectures, such as ours (see Table 3). We train these methods on our egocentric video dataset with a train, validation and test data split of 50:25:25. In [25], 2D CNNs are used to extract features of individual frames and then these frame-level features are encoded as video descriptors followed by training a classifier to predict the labels. Donahue *et al.* [26] use 3D CNNs to extract features of video clips. Then, clip features are aggregated into video descriptors for classifier training. As we can see, methods proposed by Tran *et al.* [25] and Donahue *et al.* [26] do not perform well as the data has high inter class similarity.

Tsironi *et al.* [21] propose *end-to-end* gesture classification method that works with differential image input to convolutional LSTMs. They use LSTMs to capture body parts motion involved in the gestures performed in second-person perspective. This method gave us a very low accuracy, even after fine-tuning the model on our egocentric video dataset. The possible reason for this behaviour could be that our data involved varying background and no static reference to the camera. Sharma *et al.* [27] propose attention based video classification that performed poorly owing to the high inter-class similarity which posed challenges

in classification with the limited data available for *end-to-end* training. For such fine-grained classification tasks, we require features from a very small portion of the entire frame, that is, the fingertip location. In our scenario, since the fingertip location is known, training an attention model appears redundant.

7 Conclusion

We present an in-air gestural interface, *DrawInAir* to enable researchers to incorporate hand gestures in frugal HMDs. *DrawInAir* achieves an average accuracy of 88.0% when tested on EgoGestAR dataset. We have tested the two networks in the pipeline on an egocentric hand gesture video dataset to ensure robust fingertip detection and accurate gesture classification. The entire framework works just with monocular RGB data at real-time and can be used with frugal AR devices without any sensor fusion. Gestural interface with RGB data alone helps to facilitate mass market reach in frugal HMDs. Given that the model size is 14 MB, our framework is small enough to be ported on a resource constrained smartphone/HMD.

References

1. Hegde, S., Perla, R., Hebbalaguppe, R., Hassan, E.: Gestar: real time gesture interaction for AR with egocentric view. In: International Symposium on Mixed and Augmented Reality. IEEE (2016)
2. Hürst, W., Van Wezel, C.: Gesture-based interaction via finger tracking for mobile augmented reality. Multimed. Tools Appl. **62**(1), 233–258 (2013)
3. Waldherr, S., Romero, R., Thrun, S.: A gesture based interface for human-robot interaction. Auton. Robots **9**(2), 151–173 (2000)
4. Yang, C., Han, D.K., Ko, H.: Continuous hand gesture recognition based on trajectory shape information. Pattern Recogn. Lett. **99**(1), 39–47 (2017)
5. Wobbrock, J.O., Wilson, A.D., Li, Y.: Gestures without libraries, toolkits or training: a $1 recognizer for user interface prototypes. In: Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology, pp. 159–168. ACM (2007)
6. Freeman, W.T.: Dynamic and static hand gesture recognition through low-level image analysis. US Patent 5,454,043, 26 Sept 1995
7. Liu, K., Kehtarnavaz, N.: Real-time robust vision-based hand gesture recognition using stereo images. J. Real-Time Image Process. **11**(1), 201–209 (2016)
8. Huang, Y., Liu, X., Jin, L., Zhang, X.: Deepfinger: a cascade convolutional neuron network approach to finger key point detection in egocentric vision with mobile camera. In: IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 2944–2949. IEEE (2015)
9. Tompson, J., Stein, M., Lecun, Y., Perlin, K.: Real-time continuous pose recovery of human hands using convolutional networks. ACM Trans. Graph. (ToG) **33**(5), 169 (2014)
10. Liu, K., Kehtarnavaz, N., Carlsohn, M.: Comparison of two real-time hand gesture recognition systems involving stereo cameras, depth camera, and inertial sensor. In: SPIE Photonics Europe, International Society for Optics and Photonics, paper no. 91390C (2014)

11. Dardas, N.H., Georganas, N.D.: Real-time hand gesture detection and recognition using bag-of-features and support vector machine techniques. IEEE Trans. Instrum. Meas. **60**(11), 3592–3607 (2011)
12. Keskin, C., Kıraç, F., Kara, Y.E., Akarun, L.: Hand pose estimation and hand shape classification using multi-layered randomized decision forests. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012. LNCS, vol. 7577, pp. 852–863. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33783-3_61
13. Mohatta, S., Perla, R., Gupta, G., Hassan, E., Hebbalaguppe, R.: Robust hand gestural interaction for smartphone based AR/VR applications. In: IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 330–335. IEEE (2017)
14. Long Jr, A.C., Landay, J.A., Rowe, L.A.: Implications for a gesture design tool. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 40–47. ACM (1999)
15. Cao, C., Zhang, Y., Wu, Y., Lu, H., Cheng, J.: Egocentric gesture recognition using recurrent 3D convolutional neural networks with spatiotemporal transformer modules. In: The IEEE International Conference on Computer Vision (ICCV) (2017)
16. Sridhar, S., Oulasvirta, A., Theobalt, C.: Interactive markerless articulated hand motion tracking using RGB and depth data. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2456–2463 (2013)
17. Jang, Y., Noh, S.T., Chang, H.J., Kim, T.K., Woo, W.: 3D finger cape: clicking action and position estimation under self-occlusions in egocentric viewpoint. IEEE Trans. Visual. Comput. Graphics **21**(4), 501–510 (2015)
18. Nibali, A., He, Z., Morgan, S., Prendergast, L.: Numerical coordinate regression with convolutional neural networks. arXiv preprint arXiv:1801.07372 (2018)
19. Toshev, A., Szegedy, C.: DeepPose: human pose estimation via deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1653–1660 (2014)
20. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
21. Tsironi, E., Barros, P., Wermter, S.: Gesture recognition with a convolutional long short-term memory recurrent neural network. In: Proceedings of the European Symposium on Artificial Neural Networks Computational Intelligence and Machine Learning (ESANN), pp. 213–218 (2016)
22. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural Netw. **18**(5), 602–610 (2005)
23. Savitzky, A., Golay, M.J.: Smoothing and differentiation of data by simplified least squares procedures. Anal. Chem. **36**(8), 1627–1639 (1964)
24. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: Liblinear: a library for large linear classification. J. Mach. Learn. Res. **9**(8), 1871–1874 (2008)
25. Donahue, J., et al.: Long-term recurrent convolutional networks for visual recognition and description. ArXiv e-prints arXiv:1411.4389, November 2014
26. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3D convolutional networks. ArXiv e-prints arXiv:1412.0767, December 2014
27. Sharma, S., Kiros, R., Salakhutdinov, R.: Action recognition using visual attention. ArXiv e-prints arXiv:1511.04119 (2015)