AirGestAR: Leveraging Deep Learning for Complex Hand Gestural Interaction with Frugal AR Devices

Varun Jain* — IIIT Delhi, India
Ramakrishna Perla† — TCS Research, India
Ramya Hebbalaguppe‡ — TCS Research, India

Abstract

Hand gestures provide a natural and an intuitive way of user interaction in AR/VR applications. However, the most popular and commercially available devices such as the Google Cardboard and Wearality[1] still employ only primitive modes of interaction such as the magnetic trigger, conductive lever and have limited user-input capability. The truly instinctual gestures work only with inordinately priced devices such as the Hololens, Magic Leap, and Meta Glasses[2] which use proprietary hardware and are still not commercially available. In this paper, we explore the possibility of leveraging deep learning for recognizing complex 3-dimensional marker-less gestures (*Bloom, Click, Zoom-In, Zoom-Out*) in real-time using monocular camera input from a single smartphone. This framework can be used with frugal smartphones to build powerful AR/VR systems for large scale deployments that work in real-time, eliminating the need for specialized hardware. We have created a hand gesture dataset to train LSTM networks for gesture classification and published the same online. We also demonstrate the performance of our proposed method in terms of classification accuracy and computational time.

Keywords: Complex gesture classification, deep learning, dynamic 3D gesture, monocular vision, smartphones.

Index Terms: H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Artificial, augmented, and virtual realities; H.5.2 [Information Interfaces and Presentation]: User Interfaces—Input devices and strategies; I.2.10 [Artificial Intelligence]: Vision and Scene Understanding—Video analysis; I.4.8 [Image Processing and Computer Vision]: Scene Analysis—Time-varying imagery

1 Introduction

Wearable AR/VR devices such as the Microsoft Hololens, Daqri Smart Helmet, Meta Glasses have been exceedingly popular in recent years. The user interaction modalities used in such devices point to the fact that hand gestures form an intuitive means of interaction in AR/VR applications. These devices use a variety of on-board sensors and customized processing chips which often ties the technology to complex and expensive hardware. These devices are tailor made to perform a specific function and are not readily available due to their exorbitant prices. Generic platforms such as the Microsoft Kinect and Leap Motion Controller provide the much needed abstraction but fare poorly in direct sunlight, incandescent light and outdoor environments due to the presence of infrared radiation [4] and in the presence of reflective surfaces such as a thick glass and under water.

Smartphones today are equipped with powerful processors and high-quality optics, providing an appealing and readily available platform for AR/VR applications. These advances have introduced several low-cost alternatives such as the Google Cardboard and Wearality which are video-see-through devices for providing an immersive VR experience [1]. Using a stereo-rendered camera feed and overlaid images, audio and text, these devices can also be extended for AR applications. The main motive of using these frugal headsets (Google Cardboard or Wearality with an Android smartphone) was their economic viability, portability and easy scalability to the mass market. However, accessibility of sensors to these HMDs is limited to the sensors available on the attached smartphone. Current versions use a magnetic trigger or a conductive lever to trigger a single event hence curtailing the richness of possible user interaction.

Hegde et al. [8] conducted a feasibility study for ranking available modes of interaction for frugal Google Cardboard set-up such as magnetic trigger, conductive lever, head tracking, and speech interface. Their study revealed that *(i)* frequent usage of magnetic trigger and conductive lever leads to wear and tear of the device, *(ii)* head tracking is inconvenient and shifts the focus away from object of interest in user Field of View (FoV), *(iii)* speech recognition is inaccurate in industrial outdoor setting due to ambient noise. Hand gestures were the preferred mode of interaction as they reduce human-effort and are effective in interacting with the surrounding environment. Recent work by S. Mohatta et al. [16] presented an efficient algorithm for gesture recognition in First Person View (FPV), which focuses on recognizing a four swipe model (*Left, Right, Up and Down*) for smartphones through monocular camera vision. Despite the novelty and usability of the method, it is constrained to specific use-cases and lacks robustness under realistic conditions because of skin color dependency.

Recent advances in deep learning have provided computer vision models that are robust to intra class variations and often surpass human abilities in performing detection and classification tasks [7]. Motivated by this, we propose a framework for detecting and classifying four complex 3D hand gestures (*Bloom, Click, Zoom-In, Zoom-Out* as described in Figure 1) in FPV for AR applications involving single RGB camera input with-out having built-in depth sensors. This overcomes the limitations with existing techniques and opens avenues for rich user-interaction on frugal devices. The summary of key contributions are as follows:

1. A marker-less FPV gesture classification framework (Refer Figure 3) that includes *(i)* work by Zimmermann et al. [25] for estimating 3D hand pose from single RGB image, and *(ii)* proposed LSTM [9] architecture for accurately classifying the motion of interest points to detect the gesture being performed.

2. An AR application using Google Cardboard and an Android smartphone that detects these gestures.

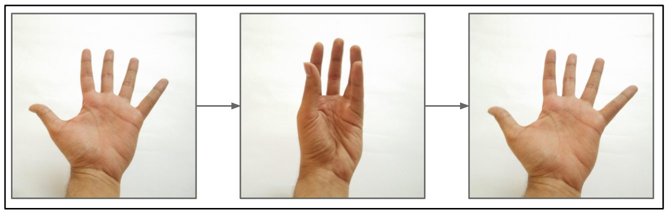
3. A dataset of *Bloom, Click, Zoom-In, Zoom-Out* gestures for training LSTM networks. It has been published online[3] for the benefit of community.

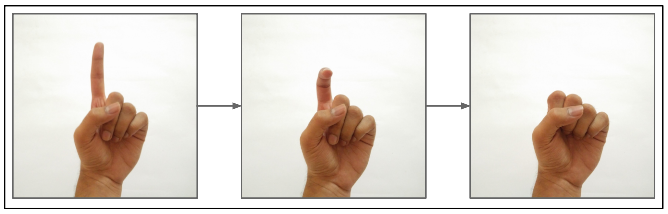
*e-mail: varun14170@iiitd.ac.in
†e-mail: r.perla@tcs.com
‡e-mail: ramya.hebbalaguppe@tcs.com

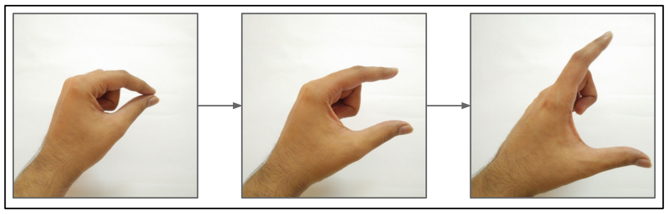
[1]https://vr.google.com/cardboard/ and http://wearality.com/

[2]https://www.microsoft.com/hololens, https://www.magicleap.com/, and https://www.metavision.com/

[3]Demo and Dataset available at http://varunj.github.io/airgestar


(a) Bloom


(b) Click


(c) Zoom-In

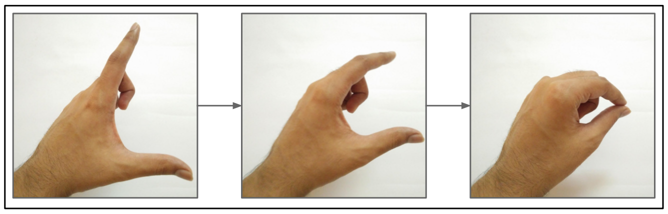

(d) Zoom-Out

Figure 1: 3D gestures supported by our framework. (a) *Bloom*: for a menu display operation, (b) *Click*: for a select/hold operation, (c) *Zoom-In*: for zooming into a scene, and (d) *Zoom-Out*: for zooming out of a scene. (a) and (b) have been inspired by the Microsoft Hololens.

2 Related Work

The potential of hand gestures as an interaction metaphor for AR applications on mobile phones has been extensively explored in the past [10]. An in-depth survey of earlier works on hand gesture recognition was done by Mitra and Acharya [15]. Buchmann et al. [3] used marker based finger tracking for interaction in AR applications. Most of the work until this point was based on skin color feature for hand segmentation and interest point detection which is followed by optical flow for tracking. Dynamic Time Warping [12] and Hidden Markov Chains [20] were extensively used for gesture classification. Song et al. [21] use multi-stage random forest pipeline that jointly classifies hand shapes and regresses depth of the hand from a single RGB-D camera. But, the focus of their study remains static gestures and estimating the depth of hand as a whole.

In a recent study, Mohatta et al. [16] presented an efficient algorithm that runs on smartphones, while focusing on recognizing a four swipe model (*Left, Right, Up and Down*) through single camera set-up. They considered cues such as skin color, hand contour segmentation, and motion tracking that effectively deal with FPV constraints caused due to AR wearable device. Despite having robust methods for tracking and gesture classification, most of the earlier attempts in gesture recognition from FPV lack in real-time performance. Gesture recognition through an AR wearable include the following challenges: *(i)* change in surrounding illumination conditions and dynamic background with skin like colors causes algorithms with skin feature dependency to fail, *(ii)* close proximity of hand to the wearable renders blurry images, which, in turn, makes tracking difficult and reduces classification accuracy.

Overcoming these challenges, deep learning approaches have proven to be very accurate in recent times and are being used in a wide variety of problems involving egocentric videos to recognize user activities and hand gestures. The recent research on estimating disparity from stereo images by Luo et al. [13] and Mayer et al. [14] boasts of high performance and accuracy on vision benchmarks such as KITTI [5], but uses stereo camera set-up and is only robust in outdoor environments with larger objects, performing poorly along the edges of objects. Hence, they give sub-par results on finger joints where the scale of change in disparity is extremely diminutive.

Kitani et al. [11] presented pixel-level hand detection in egocentric view using local appearance and global illumination models to effectively detect hand regions over a diverse set of imaging conditions. Further, Tsironi et al. show how LSTM architecture [23] can increase the accuracy of classifying temporal gestures over Convolutional Neural Networks (CNNs). Molchanov et al. [17, 18] detect temporal 3D gestures such as *tap, pinch, expand* using sophisticated 3D CNNs but take depth input from sensors such as the Microsoft Kinect that restricts its applications in head mounted devices. In a similar context, Tompson et al. [22] and Oberweger et al. [19] perform hand pose recovery with high accuracy by using depth information from a specialized hardware.

In this work, we detect complex 3D hand gestures (*Bloom, Click, Zoom-In, Zoom-Out*) in FPV using RGB data as the input, with-out additional depth information. We explored the use of recent work by Zimmermann et al. [25] for estimating 3D hand pose. It is followed by our proposed LSTM architecture for accurately classifying motion patterns to detect the type of gesture being performed by the user.

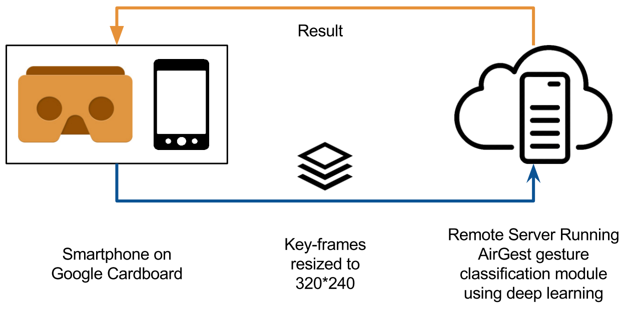

Result
Smartphone on Google Cardboard
Key-frames resized to 320*240
Remote Server Running AirGest gesture classification module using deep learning

Figure 2: System architecture: each smartphone sends down-scaled video frames to a server which runs the *AirGestAR* gesture recognition framework. The result is then communicated back to the device.

3 Proposed Framework

In this section, we describe the proposed framework for recognizing four complex 3D hand gestures (*Bloom, Click, Zoom-In, Zoom-Out*) in FPV with dynamic background setting. Figure 2 describes the major steps of our proposed set-up which includes *(i)* a smartphone with Google Cardboard device running the AR application streams video frames to the server at 25 *FPS*. Each frame is down-scaled to 320x240 resolution to achieve real-time performance by reducing the computational time, *(ii)* the server decomposes the video stream

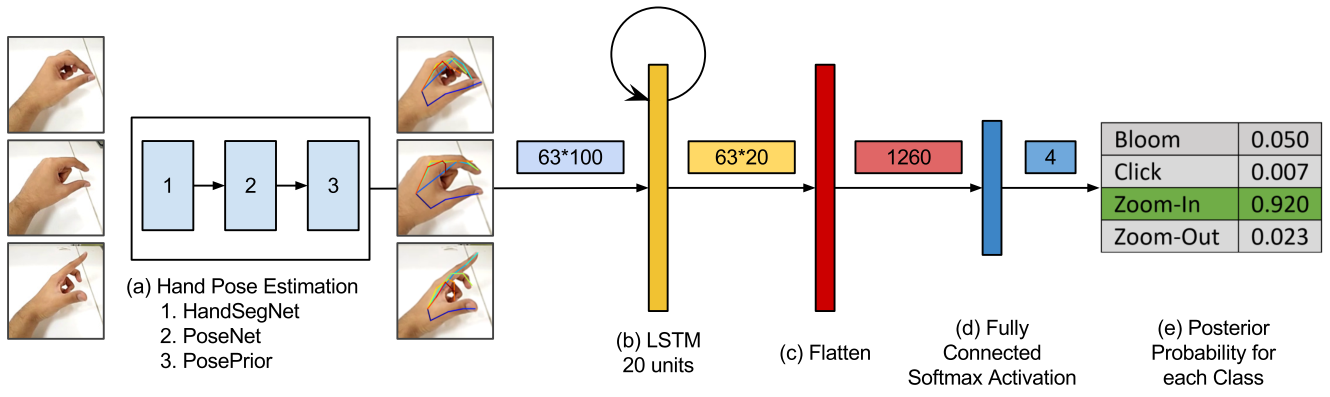


Figure 3: The proposed *AirGestAR* neural network framework for gesture recognition (Note output size after each layer). The pipeline is demonstrated via sample frames of a *zoom-in* gesture instance traced through each step in the framework. *(a)* The input video frames are first fed to hand pose estimation model proposed by Zimmermann and Brox [25]. The 21 keypoints detected by the network are shown as an overlay on the input video frames. *(b)* to *(e)* shows our proposed LSTM architecture. An LSTM layer consisting of 20 LSTM cells takes input 3 coordinates for each of the 21 key-points and 100 frames at a time. At the end, a fully connected layer outputs posterior probabilities corresponding to each of the 4 gestures.

into frames and feeds it into the *AirGestAR* gesture recognition framework (Figure 3) sequentially. A constant batch size of 100 is maintained as required by the LSTM layer, *(iii)* in case of a gesture-detection event, the recognized gesture is communicated to the smartphone to trigger the pre-defined task in the AR application.

The individual blocks of the *AirGestAR* architecture (described in Figure 3) *(1)* hand pose estimation and *(2)* gesture classification are explained in the following sections:

3.1 Hand Pose Estimation

Zimmermann and Brox in their recent work [25] proposed a deep learning approach that estimates 3D hand pose from a single RGB image overcoming the challenges caused due to unavailability of depth information. The work focuses on estimating 21 keypoints on hand present in the user FoV. The 21 hand keypoints include 4 key points per finger and one keypoint close to the wrist. Their approach detects these 21 keypoints and learns a network-implicit 3D articulation prior with the help of following three networks as shown in Figure 3(a):

1. HandSegNet [24,25]: A segmentation network to localize hand within the image.
2. PoseNet [24]: Given segmented hand mask as the input, it localizes 21 hand keypoints by estimating 2-dimensional scoremaps for each keypoint, containing likelihood information about its spatial location.
3. PosePrior [25]: Network to estimate the most likely 3D hand structure conditioned on the score maps obtained from PoseNet.

In this work, we use pre-trained models of aforementioned networks in order to estimate 21 keypoints of user hand. These networks are trained using a large-scale 3D hand pose dataset [25] based on synthetic hand models. The dataset includes 41,000 photo-realistic renderings of 20 different subjects performing 39 unique actions. All hands present in the dataset lie in range of 40cm to 65cm from the camera center which is ideal for FPV use-cases. The light position and intensities are randomized and the images are saved using a lossy JPEG compression with losses of up to 40%. The background is chosen at random from 1231 images and the camera location is chosen randomly in a spherical vicinity around the hand for each frame ensuring the robustness of the model to external factors.

3.2 Gesture Classification

The hand pose estimation discussed in previous section outputs 3D coordinate values for each of the 21 keypoints detected on the hand. We use this data as an input to our gesture classifier network as shown in Figure 3(b). Motivated by the ability and efficiency of LSTM neural networks in learning long-term dependencies of sequential data [23], we propose a novel LSTM network based architecture for the task of gesture classification using spatial location of hand keypoints in video frames. This idea of inputting just 3D coordinate values of the hand pose in modeling the keypoints' variation across frames helps in achieving real-time performance of the framework by reducing computational cost.

Figure 3(b) to 3(e) describes the proposed network architecture for gesture classification task denoting output shape after every layer. Each gesture input is sampled into 100 frames spread evenly across the duration for feeding into the network, making the input of size 63x100 (3 coordinate values for each of the 21 keypoints) to the LSTM layer. The LSTM layer consisting of 20 LSTM cells tries to learn long-term dependencies and patterns in the sequence of coordinates during network training. The LSTM layer is followed by a flattening layer that makes the data one-dimensional. It is then followed by a fully connected layer with 4 output scores that correspond to each of the 4 gestures. Since the task is to classify 4 gesture classes, we use *softmax* activation function. *Softmax* interprets these output scores as unnormalized log probabilities and squashes the output scores to be between 0 and 1 using the following equation:

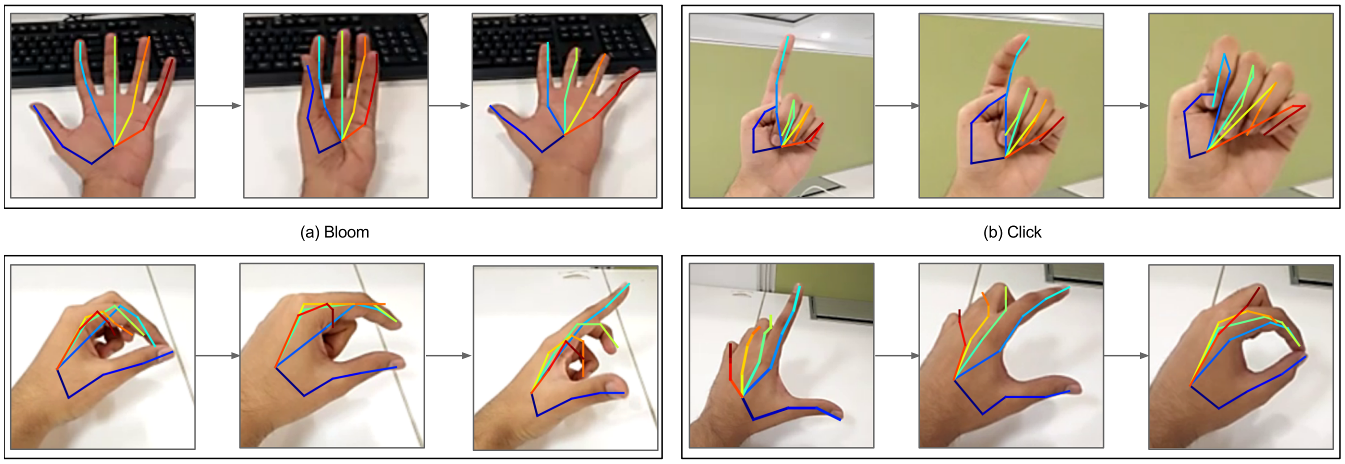
$$\sigma(s)_j = \frac{e^{s_j}}{\sum_{k=0}^{K-1} e^{s_k}} \tag{1}$$

where K denotes number of classes, s is a Kx1 vector of scores, an input to *softmax* function, and j is an index varying from 0 to $K-1$. $\sigma(s)$ is Kx1 output vector denoting the posterior probabilities associated with each gesture. The cross-entropy loss L_i of i^{th} training sample of the batch is computed by

$$L_i = -h * log(\sigma(s)) \tag{2}$$

where h is a 1xK vector denoting one-hot label[4] of the input. Finally, the mean of L_i over all the training examples of the batch is computed and used in network back-propagation to fine tune the model in training.

[4]https://en.wikipedia.org/wiki/One-hot



(a) Bloom (b) Click

(c) Zoom-In (d) Zoom-Out

Figure 4: 21 Key-points as detected by hand pose estimation network [25] are shown as an overlay on input images. Each of the 5 connections from wrist to fingertip is colored differently, variation in shade denoting each joint.

4 Experiments and Results

4.1 Dataset and Experimental Set-up

A major factor that has hampered the advent of deep learning in the task of recognizing temporal hand gestures is lack of available large-scale datasets to train neural networks on. To our knowledge, there exists no dataset, other than the one mentioned in this paper, that provides annotated data of a wide range of temporal hand gestures. We created a dataset of *Bloom, Click, Zoom-In, Zoom-Out* gestures captured in egocentric view. It has 480 videos which includes 100 videos per gesture in train set and 20 video per gesture in test set. The videos were recorded using a OnePlus X Android device mounted on a Google Cardboard. High quality videos are captured at a resolution of 320x240, and at 30 *FPS*. We have published the same online [5] for the benefit of research community.

6 subjects with varying skin color were involved in the data collection, ages ranging from 21 to 55. The videos were recorded in different places (outdoor, indoor, living room, office setting, cafeteria) in order to gather maximum variation in color composition, lighting conditions, and dynamic background scenes. Each gesture lasted for an average of 4.1 seconds; the most complex *bloom* taking the average of 5 seconds and the simpler *zoom* gestures taking an average of 3.5 seconds.

We use a server powered by an Intel Xeon E5-2650 processor, 128 GB memory and an Nvidia Tesla K20 GPU for training and testing. The models are trained using Keras v2.05, Tensorflow v0.11.0RC0.

4.2 Evaluation and Discussion

In this section, we evaluate the performance of our proposed framework. The pre-trained model of hand pose estimation proposed by Zimmermann and Brox [25] was used in our experiments in order to estimate the hand pose by detecting 21 keypoints of hand. Figure 4 shows the 21 keypoints detected by the network as an overlay on the input images while testing the system. Each of the 5 connections from wrist to fingertip is colored with a different color, variation in shade denoting each joint of finger. The 3D coordinate values of these 21 keypoints are then fed to proposed LSTM network for gesture classification.

We use proposed hand gesture dataset to train and test the LSTM classification network. While training, each of the 400 videos from the train set is sampled into 100 frames spread evenly across the duration for feeding into the LSTM network. With a batch size of 5 and a validation split of 70 : 30, the network is trained for 300 epochs taking around 11 hours on the GPU set-up mentioned in Section 4.1. We achieved an accuracy of 97.5% on the validation split while training the network. Further, we tested the model on a test set of 80 videos. Table 1 shows a confusion matrix for the experiments. We achieved an accuracy of 93.75% with 2 cases of mis-classification out of 80. The presence of a gesture is detected when the probability of a gesture is more than 70% using the following equation:

$$\max_{0\leq i\leq 3} \sigma(s)_i \geq 0.70 \tag{3}$$

where $\sigma(s)_i$ is the predicted probability for the i^{th} class. The recognized gesture is communicated to the smartphone. Otherwise, no gesture-detection is reported.

Table 1: Confusion matrix for the proposed framework yielding an accuracy of 93.75% with 2 cases of mis-classification out of 80.

Predicted Gesture / True Gesture	Bloom	Click	Zoom In	Zoom Out	Unclassified
Bloom	20	0	0	0	0
Click	0	16	0	2	2
Zoom-In	0	0	20	0	0
Zoom-Out	0	0	0	19	1

Our LSTM-only architecture is capable of delivering frame rates of up to 107 on GPU implementation. However, the hand pose estimation network works at 9 *FPS*. To ensure maximum throughput of the combined framework, we allow the hand pose estimation network to drop frames; the latest frame received at the server is fed to the network. We interpolate 3D coordinate values before feeding them to our LSTM network to get 100 data-points. This enables our framework to dynamically adapt to GPU performance, hence minimizing the recognition time after completion the gesture. As a result, the average response time of the proposed framework is found to be 0.8*s* on the GPU configuration discussed in Section 4.1.

[5] http://varunj.github.io/airgestar

On deeper analysis, we find that the click and the zoom-out gestures are highly correlated since both involve movement of index finger towards the palm. The bloom and the zoom-out gestures fare well due to their unique nature and the fact that the bloom gesture exploits most of the 21 keypoints being detected. The rest are not classified due to the explicit threshold set by us. We observe that lowering this threshold or removing it causes the classification of actions that were not intended to denote any input.

5 Future Work

In future, we would extend the system to make it more robust by evaluating new architectures and incorporating more training examples. The system relies heavily on the trained model provided by Zimmermann and Brox [25]. Their network was not able to recognize the *click* gesture as it appears on Microsoft Hololens due to the nature of dataset that the model had been trained on. Hence, we had to redefine the gesture by making the fist face the camera, which is not intuitive. To this end, we would like to explore the possibility of tuning the model with more synthetic images that are a better representation of gestures commonly used in FPV applications.

We also intend to use Generative Adversarial Networks (GANs) [6] to augment the dataset. We are experimenting with larger datasets that would complement the growing needs of AR applications. We would also like to look at the possibility of exploiting powerful System on Chip (SoC) architectures to port the Tensorflow models to an embedded board such as the ODROID [6] board. This has been successfully explored by Baraldi et al. [2]. This would remove the wearable device's dependency on external GPU servers for computation and make it possible for large scale deployment of framework, even in network constrained environment.

6 Conclusion

We presented a novel approach for marker-less dynamic 3D hand gesture recognition in ego-centric videos. The proposed framework works with only RGB data, with-out depth information. This can enable wider reach of frugal devices for AR applications. Our network can recognize 4 intuitive hand gestures (*Bloom, Click, Zoom-in and Zoom-out*) in real-time and has the potential to be extended for more complex recognition tasks by fine tuning the models using more realistic hand gesture data. We have also published the gesture dataset used in training and testing of the LSTM network. We demonstrated the performance of our proposed approach under realistic conditions and also reported turn-around-time and accuracy of gesture recognition.

References

[1] A. Amer and P. Peralez. Affordable altered perspectives: Making augmented and virtual reality technology accessible. In *IEEE Global Humanitarian Technology Conference (GHTC 2014)*. IEEE, oct 2014.

[2] L. Baraldi, F. Paci, G. Serra, L. Benini, and R. Cucchiara. Gesture recognition in ego-centric videos using dense trajectories and hand segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 688–693, 2014.

[3] V. Buchmann, S. Violich, M. Billinghurst, and A. Cockburn. Fingartips: Gesture based direct manipulation in augmented reality. In *Proceedings of the 2Nd International Conference on Computer Graphics and Interactive Techniques in Australasia and South East Asia*, GRAPHITE '04, pages 212–221, New York, NY, USA, 2004. ACM.

[4] P. Fankhauser, M. Bloesch, D. Rodriguez, R. Kaestner, M. Hutter, and R. Siegwart. Kinect v2 for mobile robot navigation: Evaluation and modeling. In *2015 International Conference on Advanced Robotics (ICAR)*. IEEE, jul 2015.

[5] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[6]http://www.hardkernel.com/main/main.php

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[7] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*. IEEE, dec 2015.

[8] S. Hegde, R. Perla, R. Hebbalaguppe, and E. Hassan. GestAR: Real time gesture interaction for AR with egocentric view. In *2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR-Adjunct)*. IEEE, sep 2016.

[9] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, Nov. 1997.

[10] W. Hürst and C. van Wezel. Gesture-based interaction via finger tracking for mobile augmented reality. *Multimedia Tools and Applications*, 62(1):233–258, jan 2012.

[11] C. Li and K. M. Kitani. Pixel-level hand detection in ego-centric videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3570–3577, 2013.

[12] K. Liu, N. Kehtarnavaz, and M. Carlsohn. Comparison of two real-time hand gesture recognition systems involving stereo cameras, depth camera, and inertial sensor. *Real-time Image And Video Processing 2014*, 9139, 2014.

[13] W. Luo, A. G. Schwing, and R. Urtasun. Efficient deep learning for stereo matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5695–5703, 2016.

[14] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4040–4048, 2016.

[15] S. Mitra and T. Acharya. Gesture recognition: A survey. *Trans. Sys. Man Cyber Part C*, 37(3):311–324, May 2007.

[16] S. Mohatta, R. Perla, G. Gupta, E. Hassan, and R. Hebbalaguppe. Robust hand gestural interaction for smartphone based AR/VR applications. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, mar 2017.

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

[18] P. Molchanov, X. Yang, S. Gupta, K. Kim, S. Tyree, and J. Kautz. Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4207–4215, 2016.

[19] M. Oberweger, P. Wohlhart, and V. Lepetit. Hands deep in deep learning for hand pose estimation. *arXiv preprint arXiv:1502.06807*, 2015.

[20] L. Rabiner and B. Juang. An introduction to hidden markov models. *ieee assp magazine*, 3(1):4–16, 1986.

[21] J. Song, G. Sörös, F. Pece, S. R. Fanello, S. Izadi, C. Keskin, and O. Hilliges. In-air gestures around unmodified mobile devices. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, pages 319–329, New York, NY, USA, 2014. ACM.

[22] J. Tompson, M. Stein, Y. Lecun, and K. Perlin. Real-time continuous pose recovery of human hands using convolutional networks. *ACM Transactions on Graphics (ToG)*, 33(5):169, 2014.

[23] E. Tsironi, P. Barros, and S. Wermter. 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)*, pages 213–218, 2016.

[24] S.-E. Wei. *Convolutional Pose Machines: A Deep Architecture for Estimating Articulated Poses*. PhD thesis, Carnegie Mellon University Pittsburgh, PA, 2016.

[25] C. Zimmermann and T. Brox. Learning to estimate 3d hand pose from single RGB images. *CoRR*, abs/1705.01389, 2017.