2018 24th International Conference on Pattern Recognition (ICPR)
Beijing, China, August 20-24, 2018

Applying hand gesture recognition and joint tracking to a TV controller using CNN and Convolutional Pose Machine

Yueh Wu and Chien-Min Wang
Institute of Information Science, Academia Sinica
yuehwu1994@gmail.com, cmwang@iis.sinica.edu.tw

***Abstract*—This paper introduces a novel TV control simulation system that recognizes hand gestures and track hand joints based on Convolutional Neural Networks (CNN) and Convolutional Pose Machines (CPM). The system provides users with an intuitive means of controlling television functions through hand gestures. Moreover, based on relative position and angle of fingers, users can manipulate an onscreen cursor and continuously modify volume & channels at varying speed. We achieved 95.8 percent testing accuracy in 19 gestures with 4 subjects, and average 11 & 45 fps while conducting CPM and CNN respectively.**

***Keywords*—Gesture recognition, TV control system, Convolutional Pose Machine, Hand joint**

I. Introduction

Television has become a nearly ubiquitous feature of daily life, and remote controls are now a standard feature in most televisions [2]-[11]. However, traditional point-click-style TV controllers have some drawbacks: they are easily lost or broken, and depend on batteries which can wear out, and current trends in controller design focus on voice or gesture inputs [4].

Hand gesture-based TVs controllers can be implemented using simple RGB cameras, IR sensors [10] or depth maps [7] to capture hand movement. However, these approaches are still subject to two challenges. First, static hand-based recognition techniques for TV controllers cannot effectively differentiate between subtle differences in hand positions, thus limiting the number of unique gestures and functions a controller system can accommodate. To address this issue, Jeong [2], Shimada [3] and Chu [11] based discrimination on hand motion as well as position, but this requires increased execution time and negatively impacts the user experience. Secondly, television users are used to controllers which allow them to control variables such as brightness and volume on a relatively fine-grained continuum, and button-based controllers allow for users to increase/decrease a given variable until they find the desired setting. However, current hand-based TV controllers do not allow for such continuous increase/decrease by maintaining a specific gesture.

To tackle with the first issue, we propose using Convolutional Neural Networks [12] in our hand gesture recognition TV controller system. This would allow us to use more gestures with a high degree of recognition accuracy in single frames. For the second issue, we use the Convolutional Pose Machine [13] to track 21 hand joints. This information could be processed to extract the relative direction and angle of the fingers to establish gestures that increase/decrease by degree at different speeds. Moreover, they also could be used to simulate clicking buttons and moving the cursor.

Our system runs in near real-time at about 11 fps for CPM and about 45 fps for CNN gesture recognition, and achieves an accuracy rate of 95.8% for nineteen gestures without assistance of 3D depth mapping.

The remainder of this paper is organized is as follows. Section II reviews the literature on TV control systems, Section III describes the proposed system, and Section IV describes and evaluates experiments conducted. Finally, conclusions and future directions and proposed in Section V.

II. Related Work

Freeman et al. [1] first proposed the concept of using hand gestures to remotely control television set functions. They used visual feedback from the TV, with the users hand motion used to control an onscreen cursor. Ionescu et al. used the angle and number of hands/fingers to recognize static hand gestures using a 3D camera [7]. Lee et al. [5] proposed a smart TV interaction system and explored the potential of personalized TV-based services using facial and gesture recognition. These three studies were all designed to use static hand gestures and, while allowed for remote gesture-based control, the number of gestures was limited.

In addition to static methods, [2], [3], [11] emphasized the use of dynamic gesture recognition, using motion paths to recognize instructions in the form of hand-drawn symbols. Jeong et al. [2] proposed a gesture drawing system with an on-screen cursor and a single camera. Shimada et al. [3] used an 8-digit quantizing code based on hand movement, allowing the system to recognize various gestures based on the sequence of digits. Chu and Su [11] used a Kinect sensor to recognize digit-based inputs from hand movement. A motion-based hand system is more natural form of human-computer interaction [4] and could allow a greater number of gestures to be classified, thus enhancing controllability. However, such an approach requires additional execution time, which has a negative impact on the user experience.

Their are other kinds of recognition method in TV controlling systems [4] [6] [8] [9] [10]. Park et al [8] studied the voice control system, but interference is still an issue in realistic environment with noise and sound from TVs. Luna et al [9] proposed recognition based on wrist gestures through a smartwatch which could classify six different gestures and map them to commonly invoked TV control functions, but the need for the smartwatch introduced a significant degree of inconvenience. Park and Lee [10] implemented IR-LED

978-1-5386-3788-3/18/$31.00 ©2018 IEEE

remote controllers based on cameras sensing infrared light from a moving hand-held controller, but this solution still required the user to hold a controller, thus providing minimal improved convenience over a traditional remote control. In [4] [6], the authors proposed hand gesture-based TV controlling systems which focus on different modes of TV interaction. Watanabe and Miyake [6] introduced remote touching point design while Lian and Wang [4] sought to minimize power consumption and computational cost for absent and inactive users using the finite-state machine (FSM).

In contrast, the present work proposes a computer-based simulation of a 2D RGB TV control system that could recognize 19 static gestures based on the use of convolutional neural networks [12]. This allows a greater variety of signal functions than the traditional icon-based recognition systems. Because gestures are recognized in individual frames, the execution time required is lower than in other motion-based systems. We also apply Convolutional Pose Machine (CPM) [13] to track hand joints, allowing users to continuously scale parameter values up or down at different rates - a function not addressed by earlier work. In addition to CNN and CPM, we also develop a mechanism to extract specific commands in series of gestures from individual frame, thus facilitating gesture prediction.

III. System Flow

Our proposed system involves four steps (Fig. 1.). The first part preprocesses the original RGB input into hand contours. These contours are then used as CNN inputs for gesture prediction. We then design a mechanism to extract effective TV commands rather than simply relying on frame-by-frame gestures. Finally, hand joint tracking is used to control parameter adjustment, clicking and cursor movement once the system recognizes the Control command.

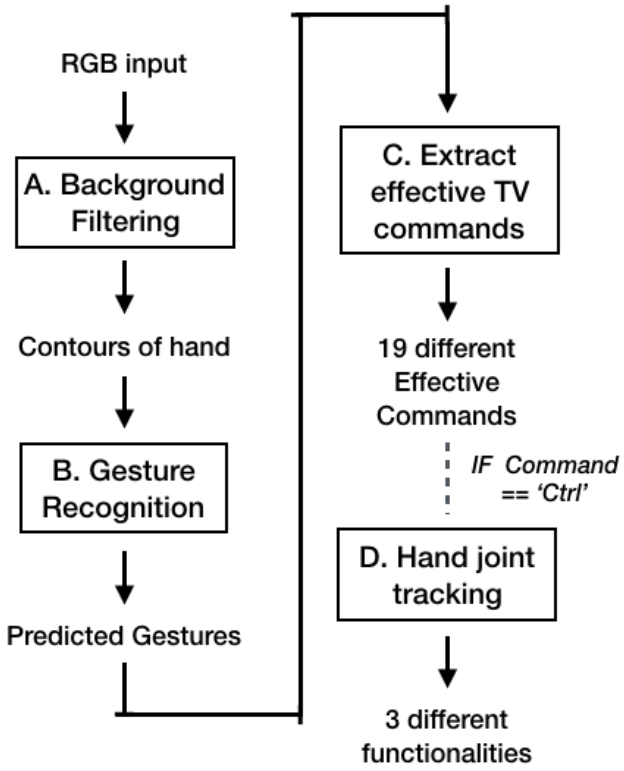


Fig. 1. Proposed system flow

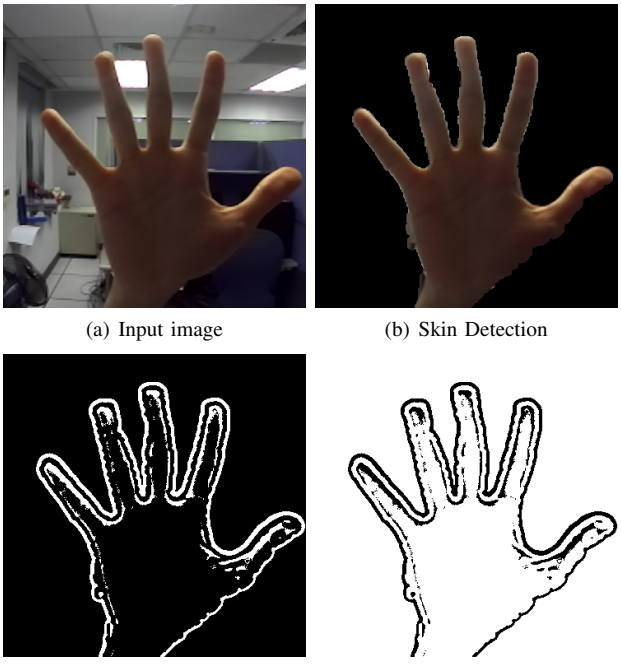
A. Background Filtering

1) Skin Detection: We identify the users hand by skin color to filter unnecessary background, thus the system requires ordinary lighting condition to operate. To integrate the Convolutional Pose Machine for hands described in [12], we set the input size to 368x368 pixels with 3 RGB channels.

We first convert the input (Fig 2(a)) to the HSV color space [14] and specify a skin mask range by upper and lower boundaries. We then apply erosions and dilations to remove false-positive skin parts to obtain a binary mask with a white skin region. The last step is to convert the white region to its original color in RGB. (Fig 2(b))

2) Extract hand contour: Even with skin detection to extract the hand object, it is preferable to exclude the RGB values of other hands because of different subjects or brightness. To this end, and to simplify the number of parameters, we choose contour [16] as region of interest.

We first convert the skin image in Fig. 2 to a gray color space. Adaptive thresholding [17] is then applied with a gaussian weighted sum as the threshold value. This extracts the hand contour (Fig 2(c)). Finally, we use inverted binary thresholding to turn the contour black (Fig. 2(d)). This preprocessing method is based on [15].



(a) Input image (b) Skin Detection

(c) Hand Contour (d) Contour by int-binary threshold

Fig. 2. (a)-(d) Illustration of input preprocessing process.

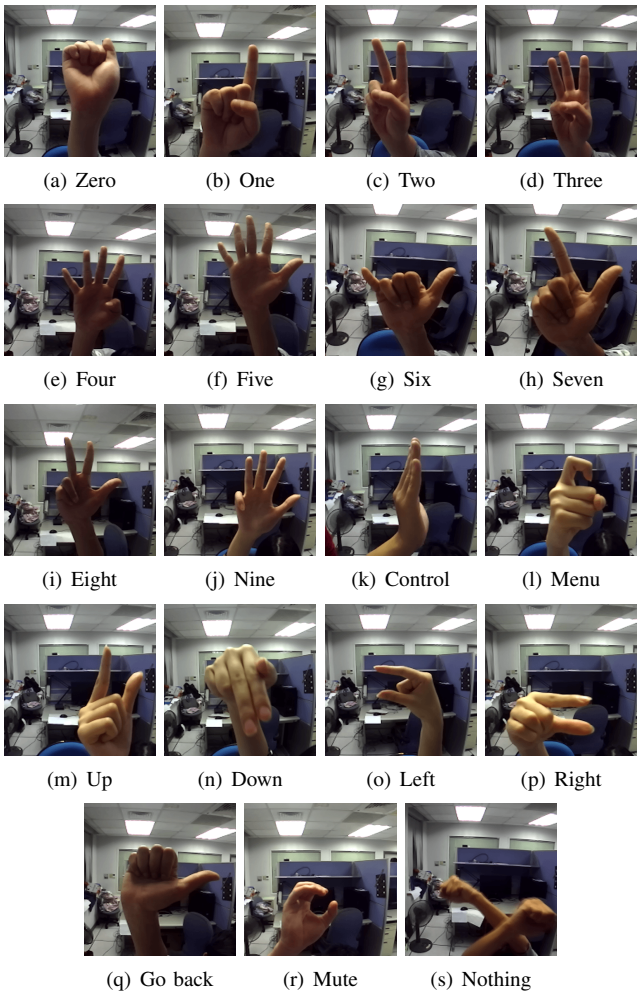


Fig. 3. (a)-(s) Illustration of nineteen hand gestures.

B. Gesture recognition

1) Introduction to gestures: To simulate TV controlling, we mapped a vocabulary of nineteen common gestures to various TV control functions. These nineteen gestures are illustrated in Figure 3.

- Number :
  Numbers from zero to nine are used to simulate channel changing.
- Control :
  This gesture is used to activate the convolutional pose machine for hand joint tracking to simulate clicking, TV cursor movement and continuous parameter value increase/decrease at varying speeds. (See Sec. III.C)
- Menu :
  Access to the Menu page in TV.
- Direction :
  Up, Down, Left, and Right are used to simulate option changing in the Menu.
- Go back :
  Return to the previous channel/menu level.
- Mute/ Unmute :
  Mute or unmute audio.
- Nothing :
  Undefined gestures or skin detection. Include human faces, hand waving, etc.

2) CNN architecture: We resize the input image to half its original size (184x184 pixels). We then set deeper convolution and pooling layers to reduce the total number of parameters. The network has 6 convolution layers and 3 max-pooling layers (where 2 convolution layers are followed by 1 pooling layer). The 6 layers respectively have 8, 8, 16, 16, 32, and 32 layers. Each set of 6 layers is applied with zero-padding to maintain the original input size. Except for the first max-pooling layer, which has a kernel size of 2x2 to preserve information in early stage, the kernel size of all the other 11 layers is 5x5. After flattening, we used two fully connected layers before applying a sigmoid function to classify the nineteen gestures. As for the activation function, we used rectified linear units (ReLU) in all convolutional and pooling layers, and hyperbolic tangent function in 2 fully connected layers. Moreover, batch normalization [18] is used after every convolutional and fully connected layer to deal with internal covariate shift.

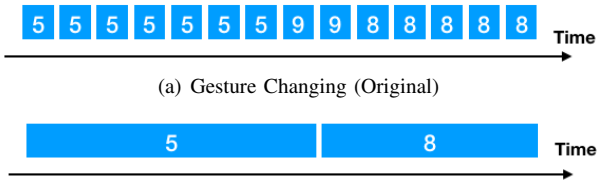
This model was trained from scratch without using pre-trained network. To ensure the model is generalized to other users, we applied cross validation, shuffling and validation split. Moreover, in order to minimize validation lost, *training epoch* and *patient* are set. Detailed training process is explained in section IV.

C. Extract effective TV commands

Simply extracting TV commands by frames after CNN raises several issues. First of all, random gestures might be misinterpreted as intentional commands. For example, the system might predict ”Five” if the user randomly waves his/her hand in front of the screen. Next, after recognizing the users intention, the system would predict repetitive and divided gestures by frames, and produce unwanted predictions between gestures (see Fig. 4(a)).

We address these issues as follows. First, we have to confirm that the users gesture is an intentional command, rather than random movement. Secondly, we need to exclude unwanted prediction between gesture transformation and integrate repetitive and divided predictions as effective TV commands. This section demonstrates how to distinguish user intention using a by two-state (Active and Inactive) finite state machine, and then explains how we exclude unwanted predictions and exclude low-confidence gestures in the Active State. The ideal prediction result is shown in Figure 4(b).

1) Distinguish user intention: To allow our system to distinguish intentional gesture commands from random movement, we designed two states: Active State triggers gesture recognition, while Inactive State is not interpreted as a command. State transformation is achieved using two thresholds, *thrA* and *thrB*. The system initially defaults to Inactive State. If the system detect skin and predicts any gesture besides Nothing for subsequent *thrA* frames, the system switches to Active State. Otherwise, if the system fails to detect skin or predicts ”Nothing” for subsequent *thrB* frames, it remains or



(a) Gesture Changing (Original)

(b) Gesture Changing(Ideal)

Fig. 4. Gesture prediction by frames. Figure 4(a) shows the repetitive and unwanted prediction issue due to frame by frame prediction, but we hope to exclude them by achieving the state shown in Fig.4(b)

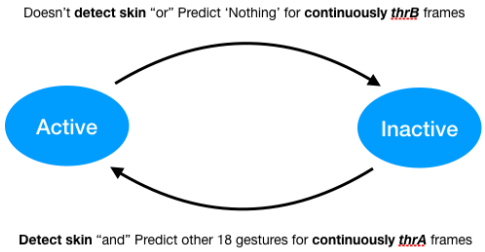


Fig. 5. Finite state machine used to address gesture prediction problems. Two thresholds, *thrA* and *thrB*, are used to differentiate between intentional and random gestures

switches in Inactive State. Our experiments respectively set *thrA* and *thrB* at 5 and 15. Figure 5 illustrates the finite state machine.

2) Exclude unwanted prediction and withhold low-confidence gestures: While the system is in Active State, it may produce unwanted predictions during gesture transformation. For example, if users want to change to channel 58, the system might obtain Nothing or Nine (Fig. 6) when the user changes his/her hand gestures from Five to Eight. To avoid this issue, we set a threshold *thrC* where, if the same gesture has been predicted for continuously *thrC* frames, the prediction is confident and can be extracted as an effective TV command instead of repetitive or divided gestures. Otherwise, if the gesture is predicted for fewer than *thrC* frames, it is considered an unwanted prediction. In our experiment, *thrC* is set at 7, equivalent to about 0.3 seconds.

D. Hand joint tracking

This section introduces the mechanism used to simulate three control functions: continuous parameter increase/decrease at different speeds, clicking and TV cursor movement.

The system recognizes a "Control" pose (Fig. 7(a)) to active convolutional pose machine (CPM) [13] to track hand joints (Figs. 7(b,c)). CPM integrated pose machine framework [19] that denotes pixel in images with a sequence of multi-class predictors, and convolutional architecture to enabling end-to-end joint training in this model. It provides an advantage in that it only requires a 2D camera to precisely track the fingers and joints, rather than a 3D depth map, allowing for the inclusion of a wide range of based the relative position and angle of fingers for hand-based TV control systems. To achieve adequate precision, we utilized pre-trained model and

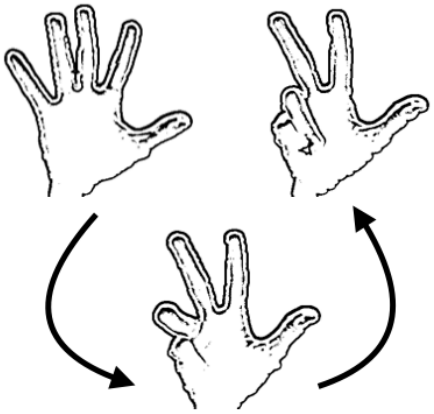
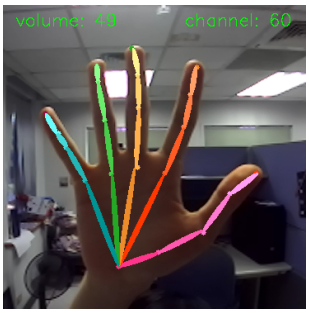


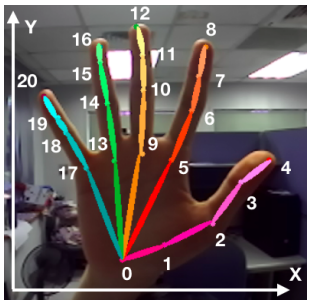
Fig. 6. As the users gesture changes from Five to Eight, the system might predict Nothing or Nine due to the position of the ring fingers and pinky fingers. The threshold *thrC* is used to address this issue

(a) Control Pose

(b) Original Image



(c) Image with joints

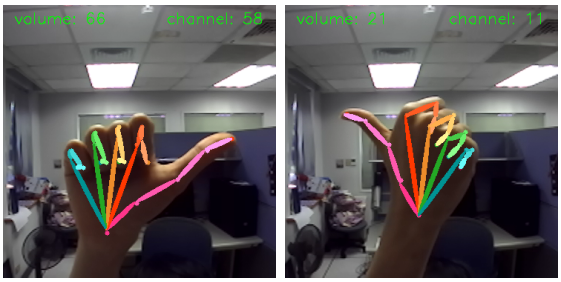


(d) position of twenty-one joints

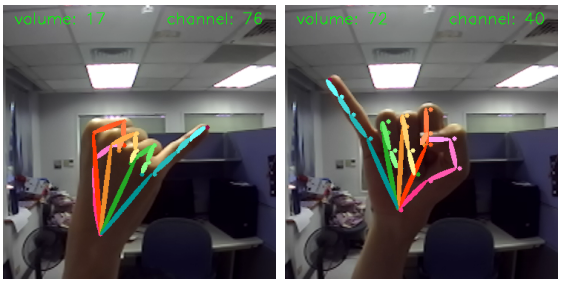
Fig. 7. Figure (a) shows the pose to activate convolutional pose machine (CPM) to track hand joints. Figures (b) and (c) respectively show the before and after images of CPM activation. Figure (d) shows the relative position of the 21 hand joints

set the input size to 368x368 pixels, identical to that used in [13]. As shown in Fig. 7(d), CPM for twenty one joints and edges with different colors to distinguish fingers.

1) Continuous parameter increase/decrease at different speeds: To achieve this function, we calculate the hand vectors of two joints to simulate two-stage (i.e., every 5 frames and every 1 frame) changes in degree as follows:



(a) Add Volume (b) Minus Volume



(c) Add Channel (d) Minus Channel

Fig. 8. Increase/Decrease volume and channel by hand joints

$$\angle 55^0 < tan^{-1}(\overrightarrow{A_0A_x}) < \angle 90^0 \quad (+1 \text{ every } 5 \text{ frames})$$
$$\angle 0^0 < tan^{-1}(\overrightarrow{A_0A_x}) < \angle 55^0 \quad (+1 \text{ every } 1 \text{ frames})$$
$$\angle 90^0 < tan^{-1}(\overrightarrow{A_0A_x}) < \angle 125^0 \quad (-1 \text{ every } 1 \text{ frames})$$
$$\angle 125^0 < tan^{-1}(\overrightarrow{A_0A_x}) < \angle 180^0 \quad (-1 \text{ every } 5 \text{ frames})$$

$(\overrightarrow{A_0A_x})$ denotes hand vector, and "0" & "x" represent the index of hand joints based on Fig. 7(d). We set $\overrightarrow{A_0A_4}$, where x = 4, as vector of thumbs. As for $\overrightarrow{A_0A_{20}}$, where x = 20, we set it as the vector of the little finger. Figures 8(a)-(d) show the operation of the channel changing function. It is possible to scroll parameters such as brightness, contrast and volume in finer increments by including additional hand angles. However, the point of this study is simply to show the feasibility of using CPM for TV control, thus we only include two parameters (channel and volume) at two different speeds.

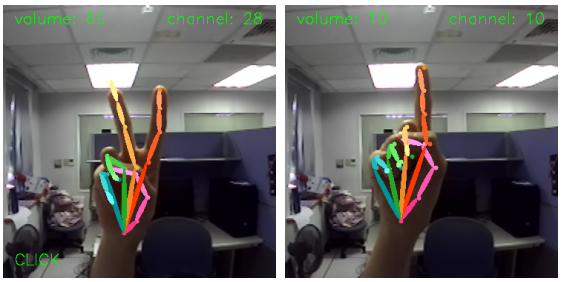
2) Clicking: We simulate the clicking function in a smart TV by the Y axis position of joints. The system detects a click if only the middle and index fingers are extended (Fig. 9(a)).

3) Cursor movement: We simulate the clicking function in a smart TV by the Y axis position of joints. The system detects a click if only the middle and index fingers are extended (Fig. 9(a)).

To leave hand tracking mode, the user simply removes his/her hand from the screen, and the system returns to gesture recognition mode.

IV. Experimental Evaluation

To evaluate the practical effectiveness of the proposed system, we tested CNN and CPM in terms of CNN recognition accuracy and frame per second (fps).



(a) Clicking (b) Cursor

Fig. 9. Clicking and cursor movement

A. Recognition Accuracy

We first collected a hand gesture dataset by asking 4 participants to make a series of hand movements, producing a set of 1000 continuous frames per gesture, and a total dataset of 76,000 (1000*19*4) frames for training and testing.

We used four-fold cross validation to ensure the user-specific model could be generalized to other users. Data from three of our four subjects were used as the training & validation set, while the remaining subject was used as testing set. The experiment was conducted four times to calculate the average accuracy of the four subjects. As for the other settings, shuffling and a 20% validation split were applied to the training set. We set the *training epoch* = 35 and *patient* = 10 based on minimum validation lost. Our results are summarized Table I, showing an average testing accuracy of 95.8% for the nineteen gestures.

Subject	1	2	3	4	**Average**
Accuracy	0.945	0.978	0.976	0.932	**0.958**

TABLE I. Testing accuracy of four subjects

We compared 19000 predicted frames from the testing set with the actual results, and compiled the results from our four subjects to produce a confusion matrix (Fig. 10). Eight and Nine produced accuracy results below 90% because of the relative difficultly of recognizing the associated contour differences of the fingers. However, all other gestures achieved accuracy results above 90%, and thirteen achieved accuracy results above 95%.

B. Latency evaluation

We calculated the maximum, minimum, average, first quartile and third quartile frame rates of 500 frames each in the hand tracking and hand recognition stages.

A frame rate of about 40-45 fps indicates real-time hand recognition without latency. However, the average frame rate in the hand tracking stage is about 11 fps, indicating slightly latency of about 0.1 seconds, which does not impact joint tracking for volume and channel control or for cursor movement.

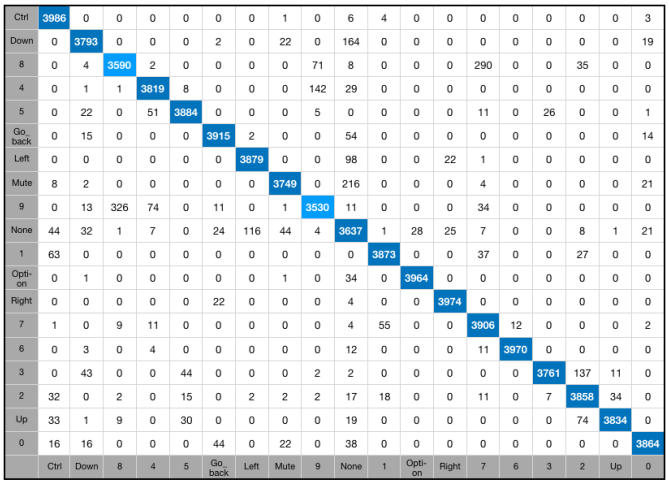


Fig. 10. Confusion matrix of the nineteen hand gestures.

	Min	Q_1	Q_3	Max	Ave
Tracking (CPM)	10.4	10.76	10.90	21.84	**11.33**
Recognition (CNN)	35.47	39.20	41.32	83.40	**44.59**

TABLE II. FRAME RATE FOR HAND TRACKING AND HAND RECOGNITION

V. CONCLUSIONS AND FUTURE WORKS

This paper proposes a novel RGB-based simulation for a TV remote control system. Existing motion-based approaches suffer from various limitations including longer execution times, requiring 3D depth mapping, or limited gesture recognition. The proposed method only requires a 2D camera, uses static CNN and is capable to reliably recognizing 19 separate gestures. Moreover, we apply 2D hand tracking by Convolutional Pose Machines [13] on 21 hand joints. Unlike previously proposed approaches, this method allows users to continuously increase/decrease various television control parameter values by holding a static gesture, and allows users to intuitively and remotely click buttons and move an onscreen cursor through hand movement. We provide a state change mechanism to deal with repetitive frame-based prediction and to distinguish intentional commands from random hand movement. We demonstrate the practical effectiveness of the proposed system to reliably recognize gestures and track hand joints in real time, producing average recognition accuracy of 95.8% and with a joint tracking frame rate of about 11 fps.

Future work will focus on analyzing kernel parameters and applying contour convex of hand to further improve recognition accuracy of gestures with similar contour. In addition, the proposed approach could potentially be applied to other types of graphical displays monitors, for the development of new applications by Convolutional Pose Machine (CPM) and CNN. Finally, we will continue to refine the CPM hand tracking algorithm of hand tracking on CPM to further reduce latency.

REFERENCES

[1] W. T. Freeman and C. D. Weissman, *Television control by hand gestures*, in Proceeding of IEEE International Workshop on Automatic Face and Gesture Recognition, Zurich, Switzerland, pp. 179-183, June 1995.

[2] S. Jeong, J. Jin, T. Song, K. Kwon, and J. W. Jeon, *Single-Camera Dedicated Television Control System using Gesture Drawing*, IEEE Trans. on Consumer Electronics, Vol. 58, No. 4, pp. 1129-1137, November 2012.

[3] Shimada, A., Yamashita, T. and Taniguchi, R., *Hand gesture based TV control system-Towards both user- & machchine-friendly gesture applications*, Proc. Frontiers of Computer Vision, (FCV), Jan 2013.

[4] S. Lian, W. Hu, and K. Wang, *Automatic user state recognition for hand gesture based low-cost television control system*, IEEE Trans. Consum. Electron., vol. 60, no. 1, pp. 107-115, Feb 2014.

[5] S.-H. Lee, M.-K. Sohn, D.-J. Kim, B. Kim, and H. Kim, *Smart tv interaction system using face and hand gesture recognition*, in Consumer Electronics (ICCE), 2013 IEEE International Conference on. IEEE, pp. 173-174, 2013.

[6] K. Watanabe, Y. Miyake, N. Nakamichi, T. Yamada, and T. Ozeki, *Remote touch pointing for smart tv interaction*, in Consumer Electronics (GCCE), 2014 IEEE 3rd Global Conference on. IEEE, pp.232-235, 2014.

[7] D. Ionescu, B. Ionescu, C. Gadea, and S. Islam, *An intelligent gesture interface for controlling TV sets and set-top boxes*, in Proceeding of 6th IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI), pp. 159-164, May 2011.

[8] J.-S. Park, G.-J. Jang, J.-H. Kim, S.-H. Kim, *Acoustic interference cancellation for a voice-driven interface in smart TVs*, IEEE Trans. on Consumer Electronics, Vol. 59, No. 1, pp. 244-249, February 2013.

[9] Mateus M. Luna, Thyago P. Carvalho, Fabrizzio Alphonsus A. M. N. Soares, *Wrist Player: A Smartwatch Gesture Controller for Smart TVs*, IEEE Annual Computer Software and Applications Conference, pp. 337-341, September 2017.

[10] Yunjung Park and Minho Lee, *Cost Effective Smart Remote Controller Based on Invisible IRLED Using Image Processing*, IEEE International Conference on Consumer Electronics (ICCE), pp. 434-435, March 2013.

[11] Tsai-Te Chu and Chung-Yen Su, *A Kinect-based Handwritten Digit Recognition for TV Remote Controller*, IEEE International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS), pp. 414-419, November 2012.

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton, *Imagenet classification with deep convolutional neural networks*, NIPS, 2012.

[13] Shih-En Wei, Varun Ramakrishna, Takeo Kanade and Yaser Sheikh, *Convolutional pose machines* CVPR, 2016.

[14] George H. Joblove and Donald Greenberg, *Color spaces for computer graphics*,Proceedings of the 5th annual conference on Computer graphics and interactive techniques, pp. 20-25, Aug 1978.

[15] Abhishek Singh,*asingh33/CNNGestureRecognizer: CNNGestureRecognizer (Version 1.3.0)*, Zenodo. http://doi.org/10.5281/zenodo.1064825, Nov.2017.

[16] B. Leibe and B. Schiele, *Analyzing appearance and contour based methods for object categorization*, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings. Jul. 2003.

[17] Francis H. Y. Chan, F. K. Lam, and Hui Zhu, *Adaptive Thresholding by Variational Method*, IEEE transaction on image processing, pp. 468-473, Mar. 1998.

[18] Sergey Ioffe and Christian Szegedy, *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*, Proceedings of the 32nd International Conference on Machine Learning (ICML-15) Year: 2015. Pages: 448-456.

[19] V. Ramakrishna, D. Munoz, M. Hebert, J. Bagnell, and Y. Sheikh, *Pose Machines: Articulated Pose Estimation via Inference Machines.* In ECCV, 2014