Towards The Development of Virtual Keyboard: An Activity Recognition Approach

Muhammad Haroon Yousaf, Hafiz Adnan Habib, Kanza Azhar, Fawad Hussain, Muhammad Rizwan, Malik Muhammad Asim

Department of Computer Engineering, University of Engineering and Technology Taxila, 47050, Pakistan
haroon.yousaf@uettaxila.edu.pk

Abstract: This paper presents an activity recognition based scheme for the development of the virtual keyboard for mobile and portable computing devices. Keystrokes activities are modeled in terms of the joints movements of fingers using kinematic model of hand. Generic activity recognition system is employed for keystroke detection and recognition. Hands activities on a flat surface are captured using mobile's secondary camera. Scheme composed of two steps: first, low-level features are extracted in terms of joints trajectories estimation using finger joints localization and optical flow calculation. Second, trajectories are further interpreted into feature vectors. Feature vectors are trained and classified using Hidden Markov Models leading towards the keystrokes recognition. Real-time implementation covers the accurate detection and recognition of 78 keys of the keyboard, thus providing enriched set of keys to the users of mobile devices.

[Yousaf MH, Habib HA, Azhar K, Hussain F, Rizwan M, Asim MM. **Towards the Development of Virtual Keyboard: An Activity Recognition Approach**. *Life Sci J* 2013;10(10s):275-282] (ISSN:1097-8135). http://www.lifesciencesite.com. 45

Key words: Virtual keyboard, activity recognition, trajectory interpretation, hidden markov models

1.Introduction

In last decade, a significant increase has been observed in the use of mobile and portable devices e.g. cell phones, PDA's, iPhone etc. Variety of applications running on these devices required textual input e.g. SMS, Email, document editors etc. But size of these devices has been reduced due to the mobility and portability issues, resulting in elimination or reduction of workspace for input data units in form of tiny keypads. Frequent use of above said applications needs a user-friendly workspace to input textual data. Textual input to mobile devices through tiny keypads is tedious, hectic and cumbersome task. Whereas, technology evolution intend to facilitate users with reduced system intricacy. Users are much comfortable with QWERTY full size traditional keyboard.

Researchers paid numerous attentions on this real problem by providing different methods or devices as virtual keyboards. A virtual keyboard can be defined as a touch typing device that does not have a physical manifestation of the sensing areas. Up-till now many designs of virtual keyboards have been proposed including the both sensor and vision based methods or devices for keystroke recognition. Sensor based methods and devices include Finger Joint Wearable Keypad [1], Chording Glove [2], FingerRing [3], VType [4], Samsung Scurry [5] and Senseboard [6]. But these sensor based solutions are proved to be intrusive. Vision based approaches are employed by Visual Panel [7], VKPC [8] and VKB [9].

Along with all these above said devices or methods, few other computer vision based algorithms have been proposed for the detection and recognition of keystrokes. In [10] researches presented a cost effective, user friendly, portable solution based on image capturing, character identification and device emulation modules named as VistaKey. A 3D optical ranging based virtual keyboard was proposed in [11]. System consists of pattern projector and 3D range camera detecting the fingertips by depth curves. Work presented in [11] was further extended by constructing 3D hand shape model in [12]. An intrusive wearable 3D vision based augmented reality keyboard is proposed in [13]. In our previous work [14], gesture recognition based virtual keyboard system has been proposed and implemented, but its real-time implementation raised the memory and computational complexity on a mobile platform.

Although all these devices or methods have been proposed for the development of virtual keyboard, but there are some key metrics and characteristics associated with virtual keyboards to find out the device or method's performance. These metrics includes number of discrete keys and fingers, finger key mapping, finger-key switch time, user familiarity, estimated bandwidth and visual appearance of keyboard. Keeping in view the existing virtual keyboard systems; lack of familiarity, less bandwidth in terms of characters per minute and integration with state of the art new mobile devices still a challenge for the researchers.

2.Activity Recognition based Virtual Keyboard System

Human activity recognition from videos is one of the most promising computer vision applications. Human activity recognition has been proposed and implemented in applications e.g. human activity analysis in surveillance videos [15], human machine interfacing using hand gestures [16], camera mouse control [17], game system using multi-modal user interfaces [18] and robotic control [19] etc. A detailed survey of techniques related to representation, recognition and learning of human activities is presented in [20].

In this paper, activity recognition based virtual keyboard has been proposed for keystroke detection and recognition. A flat surface is used as the workspace for the generation of keystrokes instead of physical keypads. Fig. 1 shows the comparison between traditional keypads and activity recognition based virtual keyboard.

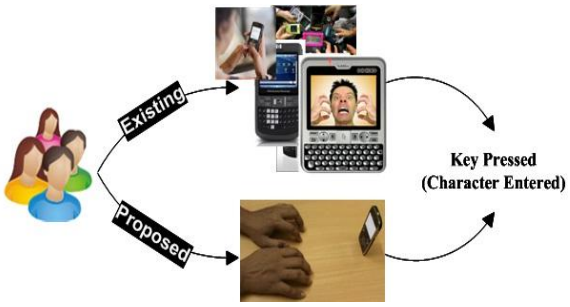

Fig. 1 Comparison b/w Traditional Keypads and Activity Recognition based Virtual Keyboard

Research work implements the activity recognition scheme on the video data captured by the secondary camera of mobile device. Ignoring the borderline between actions and activities, generic activity recognition scheme discussed in [20] is summarized as illustrated in Fig. 2. Concise low-level features are extracted from the input video data. As discussed in [20], background subtraction, blob extraction, optical flow, point trajectories and filter responses can be used as low-level features. These features are further processed to model and recognize the activities performed in the workspace. Activities can be modeled and recognized using non-parametric, parametric and volumetric approaches.

Before proceeding to the architecture of the proposed system, hand modeling is accomplished for the keystroke detection and recognition. Instead of considering each joint of the human hands as in [14], hand anatomy and kinematic model of hand [21] is utilized to represent the hands in terms of fingers and joints. Fig. 3 represents the kinematic hands model used for the development of virtual keyboard. Fig. 3 also describes the naming and localization of fingers and joints used in virtual keyboard system. As it is clear that, all ten fingers take part in the keystroke generation process, so these fingers are named according to there anatomical names as pinky, ring, middle, index and thumb. But fingers act as the operators in the proposed activity recognition system, so we have set of ten discrete operators involved in keystrokes recognition activity as shown in Fig. 3.

$$O = \{O_1, O_2, O_3 \ldots\ldots\ldots O_{10}\}$$

Fig. 2 Generic Activity Recognition System

Each operator is further divided into joints based on their anatomical naming conventions. Joints in the human hands are named according to their location on the hand as metacarpophalangeal (MCP) (i.e. joining fingers to the palm), interphalangeal (IP) (i.e. joining finger segments) and carpometacarpal (CMC) (i.e. connecting the metacarpal bones to the wrist). In the gesture recognition based virtual keyboard [14], both MCP's and IP's were considered in the representation of keystrokes, which results in the trajectory estimation for 38 joints of human hands, 19 joints for each hand. In this research work, due to modeling of hand using kinematic model, it concludes that MCP joints didn't take part more effectively in keystrokes generation so are ignored. Only IP's joints are considered, resulting in total 28 joints for both hands. For convenience IP joints are represented using red, green and blue colors in Fig. 3 e.g. each fingertip is named as J_D (Distal Joint of finger), joint between fingertip and middle of the finger as J_M (Middle Joint of finger) and joint at the middle of the finger as J_P (Proximal Joint of finger). In the development of virtual keyboard, 28 joints are represented as J_{ON} , where O = 1,2,3,…….10 representing the operator/finger and N = 1 to 3 for all operators instead of O_5 and O_6 for which N=1 to 2 due to the absence of middle joint in the thumbs. For example J_{23} and J_{82} represent the distal joint or fingertip of the ring finger of left hand and middle joint of the middle finger of the right hand respectively.

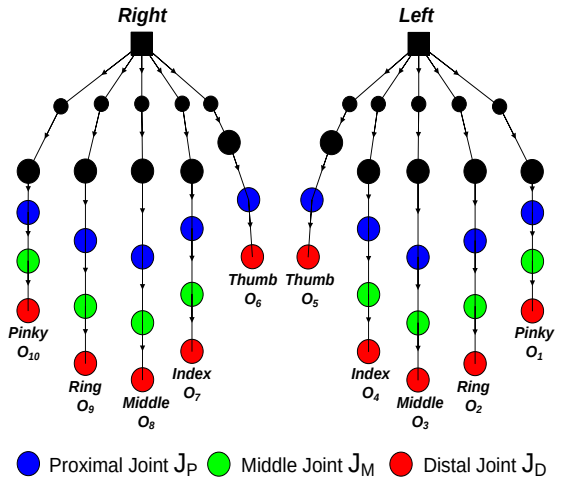

Fig. 3 Kinematic Model of Hands for Virtual Keyboard System

Talking about the number of discrete keys to be operated in the virtual keyboard, this paper covers the detection and recognition of 78 discrete keys instead of 64 keys recognized in [14]. So

$$K = \{K_1, K_2, K_3 \ldots\ldots\ldots\ldots K_L\}$$ Where L = 78

K_i represents alphanumeric characters, control keys and special characters, Where i = 1, 2, 3 ….78. Due to the absence of the physical layout of the keyboard, standard typing layout constraint is imposed regarding hand and finger locations for proper finger/operator-to-key mapping scheme. Where using QWERTY keyboard user positioned the four fingers of the left hand on A, S, D and F and four fingers of right hand on J, K, L and **;** keys. Each finger scoped the keys in both horizontal and vertical directions. For example for a QWERTY keyboard without function keys and bottom line of keys, each finger has the four key-press possibilities in vertical direction. Proper operator-to-key mapping table is shown in the Table 1 showing the keys- press possibilities for each operator. Table 1 also depicts that set of two/three joints are involved in any of the key-press event. Detailed architecture of proposed scheme is demonstrated in Section III.

The architecture of the proposed scheme is modular based in the generic activity recognition system of Fig. 1. Algorithmic detail of the proposed system is described in Fig. 4. Data acquisition phase comprises of video sequence of human hands activities, captured on a flat surface without any physical keyboard layout using camera.

Table 1. Operator-to-Key Mapping Table

Operator	Finger Associated	Possible Keys
O_1	L_Pinky	Q, A, Z, Caps Lock, Shift, Home, End, 1, !, `, ~
O_2	L_Ring	W, S, X, 2, @
O_3	L_Middle	E, D, C, 3, #
O_4	L_Index	R, F, V, T, G, B, 4, 5, $, %
O_5	L_Thumb	Space
O_6	R_Thumb	Space
O_7	R_Index	Y, H, N, U, J, M, 6, 7, ^, &
O_8	R_Middle	I, K, ,, <, 8, 9, *, (
O_9	R_Ring	O, L, ., >, 0,)
O_{10}	R_Pinky	Enter, Backsp, Arrow Keys, ;, :, ‘, , ?, /, [,], {, }, P, -, _, +, =, \, |

3.Architecture of Activity Recognition based virtual keyboard

Proposed system has real-time capabilities so sequence consists of 30 frames/sec. Therefore, the entire subsequent sequence of steps in the algorithm is performed on the frame-by-frame basis. Video data is passed to subsequent phases; feature extraction and activity recognition. Feature extraction phase aims to localize finger joints and estimate the trajectories for all 28 joints involved in the keystrokes generation process. Whereas, activity recognition phase interprets the estimated trajectories and recognize the activities in form of keystrokes detection and recognition.

A. Low-Level Feature Extraction

As discussed earlier, this phase results in the estimation of the trajectories for each joint JON. Trajectory estimation step requires the proper detection and localization of joints. After frame extraction from the video sequence, hand detection is performed using background subtraction as implemented in [22]. Algorithm used for hand detection computes the background statistics when it is static, with hands not present in the workspace. To compute the robust background statistics,

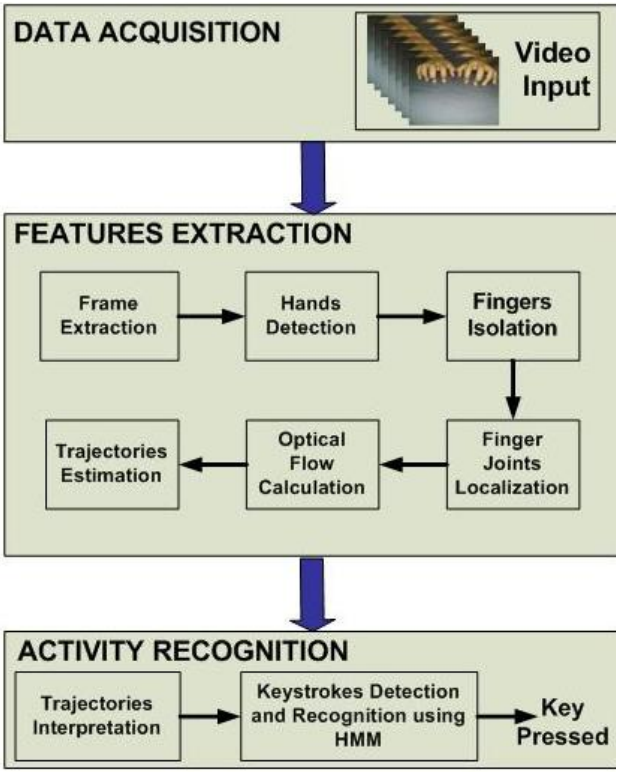

Fig. 4 Algorithmic of Proposed System

3-4sec of video with static background is sufficient. First and second-order statistics are computed as in (1) and (2):

$$F_{B,\mu}(x,y,c) = \frac{1}{T}\sum_{t=1}^{T}[F_{B,t}(x,y,c)] \quad (1)$$

$$F_{B,\sigma^2}(x,y,c) = \frac{1}{T-1}\sum_{t=1}^{T}[F_{B,t}(x,y,c) - F_{B,\mu}(x,y,c)]^2 \quad (2)$$

Where $F_{B,t}(x,y,c)$ is the color image corresponding to frame at time t and x, y shows the spatial coordinates of the corresponding frame. T is the total number of frames considered for the computation of background statistics. $F_{B,\mu}(x,y,c)$ and $F_{B,\sigma^2}(x,y,c)$ are mean and variance images respectively. These background statistics are used to calculate the quadratic difference. Quadratic difference is used in the generation of foreground image which is the hands image. Fig. 5a shows the sample image of hands from the video sequence and Fig. 5b shows the hands region detected from the frame. If there is any other objects present in the workspace other than hands, connected component labeling scheme can be used to find out the larger component as the hands in the workspace. Once the hands are detected, next is to isolate the fingers for the localization of joints. Finger identification scheme [19] and Component labeling and contour tracing algorithm [23] are implemented jointly for the finger isolation and joints localization. Finger isolation is performed using the MCP's detection for the each finger. MCP's plays the partitioning role in between fingers. For joints detection and localization, region properties e.g. area, perimeter, convexity and centroids of each finger are processed to locate the spatial coordinates of J_D, J_M and J_P for each finger. Fig. 5c and 5d shows the joints localization on the hands contours and original frame respectively. Joints localization results in matrix D comprised of location for each joint. Each component of D contains the spatial coordinates of respective joints.

$$D = \begin{bmatrix} J_{11} & J_{12} & J_{13} \\ . & . & . \\ . & . & . \\ J_{51} & J_{52} & \Phi \\ J_{61} & J_{62} & \Phi \\ J_{71} & J_{72} & J_{73} \\ . & . & . \\ . & . & . \\ J_{O1} & J_{O2} & J_{O3} \end{bmatrix} \quad \text{Where O} = 10$$

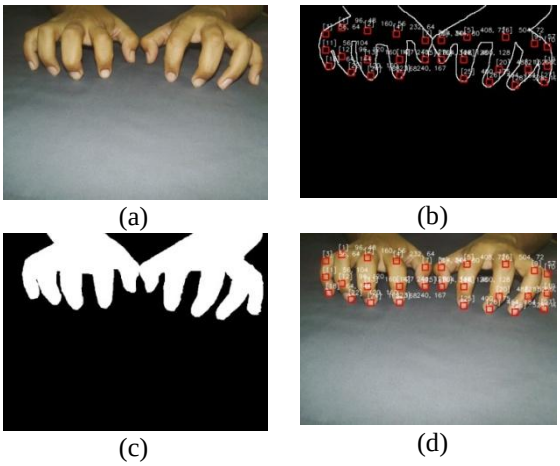

Fig. 5 Hands Detection and Joints Localization Results

Now the tracking of these joints is required for the trajectories estimation. As mentioned in [20], optical flow is one of the promising techniques as the low-level feature extraction in the activity recognition system. It tells about the apparent motion of the individual pixels in the image plane. In proposed

system optical flow calculation for every pixel is not necessary. Therefore, optical flow is computed to estimate trajectory for every joint location in the subsequent frames. For each joint two components of optical flow are computed; X and Y components where X component describes the apparent motion of respective joint in horizontal direction and Y components tells the motion in vertical direction. Cumulatively, we have two trajectory matrices as follows. Where X_{11} means the trajectory of J_1 (proximal joint) of O_1 (left pinky) along horizontal direction. Whereas, Y_{11} means the trajectory of J_1 (proximal joint) of O_1 (left pinky) along vertical direction.

$$T_X = \begin{bmatrix} X_{11} & X_{12} & X_{13} \\ . & . & . \\ . & . & . \\ X_{51} & X_{52} & \Phi \\ X_{61} & X_{62} & \Phi \\ X_{71} & X_{72} & X_{73} \\ . & . & . \\ . & . & . \\ X_{O1} & X_{O2} & X_{O3} \end{bmatrix} \quad T_Y = \begin{bmatrix} Y_{11} & Y_{12} & Y_{13} \\ . & . & . \\ . & . & . \\ Y_{51} & Y_{52} & \Phi \\ Y_{61} & Y_{62} & \Phi \\ Y_{71} & Y_{72} & Y_{73} \\ . & . & . \\ . & . & . \\ Y_{O1} & Y_{O2} & Y_{O3} \end{bmatrix}$$

$$\text{Where } O = 10$$

Fig. 6 shows the horizontal and vertical trajectories estimated for all joints of O_7 recorded for a period of about 26 seconds.

As discussed in section II, each operator has horizontal and vertical scopes for key-press representing horizontal and vertical trajectories respectively. After experimentation horizontal trajectories of all joints of an operator causes irrelevance and redundancy. So horizontally, trajectories of only proximal joints are considered. Similarly as thumbs have no scope in horizontal direction due to one-to-one operator–to-key relationship, so no horizontal trajectories are required for both thumbs too. Keeping in view the above said scenario, overall trajectories data will be reduced, which also reduces the system computational complexity, so modified trajectories matrix will be as *T*.

Equation clearly indicates the less number of trajectories required for the keystroke detection and recognition. Fig. 7 shows the reduced version of the trajectories data of O_7 shown in Fig. 6. These trajectories are further passed to activity recognition phase for keystroke detection and interpretation.

$$T = \begin{bmatrix} X_{11} & Y_{11} & Y_{12} & Y_{13} \\ . & . & . & . \\ . & . & . & . \\ \Phi & Y_{51} & Y_{52} & \Phi \\ \Phi & Y_{61} & Y_{62} & \Phi \\ X_{71} & Y_{71} & Y_{72} & Y_{73} \\ . & . & . & . \\ . & . & . & . \\ X_{O1} & Y_{O1} & Y_{O2} & Y_{03} \end{bmatrix} \quad \text{Where } O = 10$$

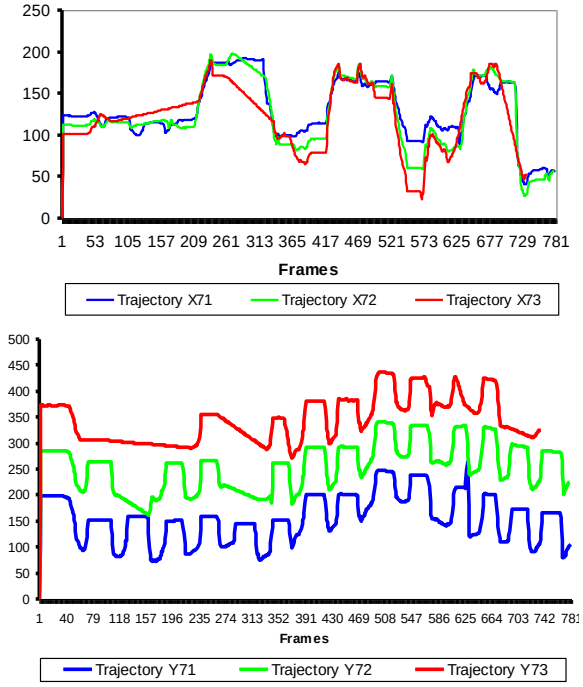

Fig. 6. Horizontal and Vertical Trajectories for O7 (index finger of right hand)

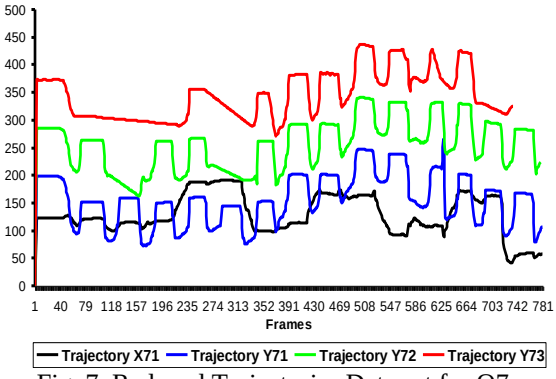

Fig. 7. Reduced Trajectories Data set for O7

B. Activity Recognition

Activity recognition phase targets the detection and recognition of keystrokes from trajectories estimated in section III-A. For keystroke detection and recognition, supervised learning scheme is adopted. Fig. 8 describes the algorithm for training and classification of trajectories for keystroke recognition activity.

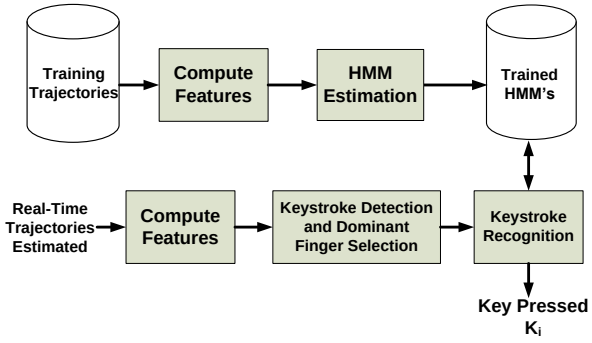

Fig. 8. Keystroke Detection and Recognition Algorithm

In the training phase, trajectories are interpreted into feature vectors comprising of the key spatial, pathic and probabilistic information. As the decision making in typing activity is dependant on the temporal data. Because typing activity exists in both space and time – it's a spatiotemporal model. It is reasonable to assume that the temporal length of an activity will vary amongst different typists. Thus we use a parametric method, the Hidden Markov Model [24-27] for proposed system. Feature vectors are employed for the estimation of HMM's. Key-press activity is defined as a sequence of spatial and directional changes which are the observation symbols. Each key is mapped to one Hidden Markov Model; it means total 78 HMM's designed for the proposed virtual keyboard. Traditional Baum-Welch [24] and the Viterbi Path Counting [24] algorithms were adopted to train the Hidden Markov Models in Left Right connectivity with 9 to 13 states.

In the real-time classification phase, trajectories estimated from real-time video data are interpreted into feature vectors. For keystroke detection, as in [14] the concept of dominant finger/operator is introduced. Dominant finger is decided on basis of proximal joint J_{i1} where i = 1,2,3,…..10 representing the operators. Feature vector is tested using a logic based technique over J_{i1}, resulting in possible key-press event by i i.e. respective finger/operator if it occurs. Dominant operator not only decides the keystroke detection activity but also reduces the further computation complexity. Once an operator's dominancy is decided, in keystroke recognition phase its not compulsory to test respective feature vector for all 78 HMM's. Then only HMM's relevant to that particular operator will be tested. Using the above said algorithm, keystrokes detected and recognized for the set of trajectories of O_7 in Fig. 7 are shown in Fig. 8. Black diamond symbols on the blue color trajectory Y_{71} shows the possibility of the keystrokes detected. Respective feature vectors of X_{71}, Y_{72} and Y_{73} at that particular time are processed to recognize the keystrokes.

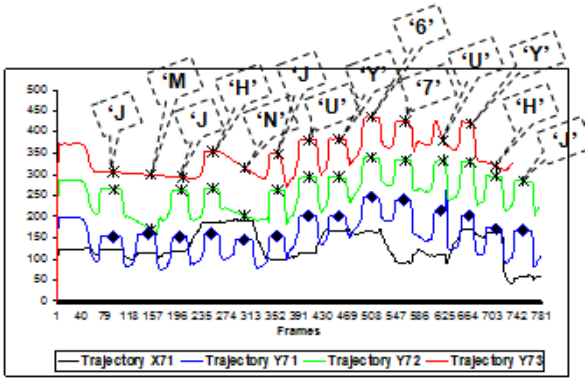

Fig. 9 Keystrokes Detected and Recognized for Trajectories of O7

4.Experimentation results and discussion

For real-time implementation, video database of 30 videos of 30 different undergraduate students typing on a flat surface were recorded in laboratory setup. After the low-level features extraction and trajectories estimation, classifier is trained using algorithm in section IIIB for 25 videos. While remaining 5 are used for testing purpose by ways of three-fold cross validation. System accuracy is evaluated on the basis of key parameters regarding development of virtual keyboard discussed in section I. These parameters include number of discrete keys and operators, operator-key-mapping, estimated bandwidth, and user familiarity. As its evident from the discussion that system operates 78 discrete keys using 10 operators. Scheme obtains the bandwidth of 4keys/sec, satisfying the standard typing speed of 250 characters per min. Scheme almost shows the 100% results for keystroke detection. Whereas, recognition accuracy is discussed in two ways; first recognition results for different keys and secondly operator-to-key mapping recognition accuracy. Table 2 shows the percentage recognition results for few alphanumeric and special keys using data samples collected from the video database recorded for experimentation.

Recognition accuracy is also evaluated across operator-to-key mapping using Table 1.

Table 2 Recognition Results for Different Keys using HMM Training and Classification

Key	Training Data	Test Data	Correct Data	Error	Recognition (%)
'Q'	74	14	13	1	**92.86**
'W'	69	16	16	0	**100**
'E'	54	12	12	0	**100**
'R'	77	15	14	1	**93.33**
'T'	70	12	11	1	**91.67**
'Y'	56	20	18	2	**90**
'U'	67	13	12	1	**92.31**
'I'	60	16	16	0	**100**
'O'	43	9	9	0	**100**
'P'	47	9	9	0	**100**
'1'	33	7	7	0	**100**
'2'	28	6	6	0	**100**
'3'	32	7	7	0	**100**
'4'	33	7	7	0	**100**
'5'	34	7	6	1	**85.71**
'6'	29	6	5	1	**83.33**
'7'	30	5	4	1	**80**
'8'	30	7	6	1	**85.71**
'9'	32	7	6	1	**85.71**
'0'	33	8	8	0	**100**
','	15	4	4	0	**100**
'.'	17	5	5	0	**100**
'/'	12	5	4	1	**80**
'+'	12	4	3	1	**75**
'['	15	3	3	0	**100**
SPC	50	22	20	2	**90.91**
Total	**1052**	**246**	**231**	**15**	**93.9**

Fig. 10 shows the % keystroke recognition results for all 10 operators. Fig. 10 also shows the cumulative results for keystroke recognition upto 94.1%. Keystroke recognition results show the poor recognition results for O_1 and O_{10} due to higher operator-to-key ratio.

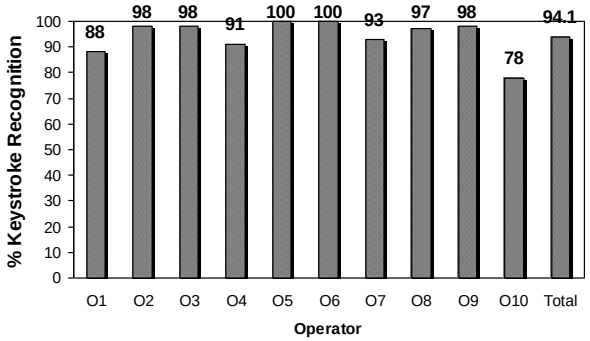

Fig. 10 Keystroke Recognition Results

5 Conclusion

Virtual keyboard system proposed in this paper provides significant amount of new algorithmic and practical content with respect to [14].System is based on a non-intrusive and software centric mechanism rather than hardware centric mechanism by using secondary camera of mobile resulting in a low-cost system. Number of discrete keys is increased from 64 to 78 thus providing more functionality to the users. System made use of less number of joints and as well as trajectories data set is reduced to almost half. It helped in the improvement of computational time and memory complexities of the system. Good user acceptance rate in terms of ease, functionality, ergonomics, non-intrusiveness and bandwidth can be achieved by deployment of this system on mobile platforms like; Android, Symbian, Rim OS and Windows Mobile etc.

Acknowledgment

Authors are thankful to postgraduate research fellows affiliated with the *video and image processing laboratory* for their suggestions and support in this research work. Authors are also thankful to the students and peers involved in the video dataset collection for the project.

References

1. Rehman,A. Alqahtani, S. Altameem, A. Saba, T. (2013) Virtual Machine Security Challenges: Case Studies, International Journal of Machine Learning and Cybernatics, DOI 10.1007/s13042-013-0166-4.
2. Rosenberg, R.: Computing without Mice and Keyboards: Text and Graphic Input Devices for Mobile Computing; PhD Thesis, Dept. of Computer Science, University College, London, 1998.
3. Saba, T. and Rehman, A. (2012). Effects of Artificially Intelligent Tools on Pattern Recognition, International Journal of Machine Learning and Cybernetics, vol. 4(2), pp. 155-162.
4. Saba, T. Rehman, A. and Sulong, G. (2010) An Intelligent Approach to Image Denoising Journal of Theoretical and Applied Information Technology, vol. 17 (1), pp. 32-36.
5. Samsung Scurry; PCWorld article at http://www.pcworld.com/article/70568/virtual_keyboards_let_you_type_in_air.html
6. Senseboard, http://www.senseboard.com
7. Saba, T. and Altameem, A. (2013) Analysis of Vision based Systems to Detect Real Time Goal

Events in Soccer Videos , Applied Artificial Intelligence, vol. 27(7), pp. 656-67.
8. Virtual Devices, Inc., VKPC; http://www.virtualdevices.net
Tech's Virtual Keyboard at http://www.vkb-support.com
9. Nadeeka Samanthi Wijewantha, Chanaka Amarasekara VISTAKEY: A Keyboard Without A Keyboard – A Type Of Virtual Keyboard IEE Eleventh Annual Conference Sri Lanka 2004.
10. Huan Du, Thierry Oggier, Felix Lustenberger, A Virtual Keyboard Based on True-3D Optical Ranging , British Machine Vision Conference 2005.
11. Sulong, G. Saba, T. and Rehman, A. (2010). Dynamic Programming Based Hybrid Strategy for Offline Cursive Script Recognition. IEEE Second International Conference on Computer and Engineering, vol. 2, pp. 580-584.
12. Lee, M. And Woo, W. ARKB: 3D vision-based Augmented Reality Keyboard, International Conference on Artificial Reality and Telexisitence, pp. 54-57, 2003.
13. Rehman, A. and Saba, T. (2012). Off-line Cursive Script Recognition: Current Advances, Comparisons and Remaining Problems . Artificial Intelligence Review, vol. 37(4), pp:261-268, DOI: 10.1007/s10462-011-9229-7.
14. Rehman, A. and Saba, T. (2012). Evaluation of Artificial Intelligent Techniques to Secure Information in Enterprises. Artificial Intelligence Review, DOI. 10.1007/s10462-012-9372-9.
15. M Elarbi-Boudihir, A Rehman, T Saba (2011) Video motion perception using optimized Gabor filter, International Journal of Physical Sciences vol. 6(12), pp. 2799-2806.
16. Luca Lumbardi, Marco Porta, Adding Gestures to Ordinary Mouse Use: a New Input Modality for Improved Human- Computer Interaction 14th International Conference on Image Analysis and Processing (ICIAP 2007)
17. Hwan Heo Eui, Chul Lee, Kang Ryoung Park, Chi Jung Kim, Mincheol Whang, A realistic game system using multi-modal user interfaces IEEE Transactions on Consumer Electronics, Vol. 56, No. 3, August 2010
18. Xiaoming Yin, Ming Xie Finger identification and hand posture recognition for human–robot interaction Journal of Image and Vision Computing 25 (2007) 1291–1300.
19. Pavan Turaga, Rama Chellappa, V. S. Subrahmanian, Octavian Udrea, Machine Recognition of Human Activities: A survey IEEE Transactions on Pattern Analysis and Machine Intelligence Vol. 18, No. 11, November 2008
20. Saba,T. Rehman, A. and Elarbi-Boudihir, M. (2011). Methods and Strategies on off-line Cursive Touched Characters Segmentation: A Directional Review . Artificial Intelligence Review, Springer, DOI 10.1007/s10462-011-9271-5.pp:45-54
21. Fre´de´ric Jean, Alexandra Branzan Albu, The visual keyboard: Real-time feet tracking for the control of musical meta-instruments , Journal of Signal Processing: Image Communication 23 (2008) 505– 515.
22. Rehman, A. and Saba, T. (2012).Neural Network for Document Image Preprocessing Artificial Intelligence Review, DOI: 10.1007/s10462-012-9337-z.
23. L. R. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, Proc. IEEE, vol. 77, no. 2, pp. 257–285, Feb. 1989.
24. A. Rehman, F. Kurniawan and T. Saba (2011) "An Automatic Approach for Line detection and Removal without Characters Smash-up". Imaging Science Journal, vol. 59(3), pp. 171-182
25. Rehman, A. and Saba, T. (2011). Performance Analysis of Segmentation Approach for Cursive Handwritten Word Recognition on Benchmark Database . Digital Signal Processing, vol. 21(3), pp. 486-490.
26. MSM Rahim, S.A.M. Isa, A. Rehman and T. Saba (2013) Evaluation of Adaptive Subdivision Method on Mobile Device, 3D-Research (Springer), Vol.4 (4), DOI. 10.1007/3DRes.02(2013)4.
27. Saba, T. and Rehman,A. (2012). Machine Learning and Script Recognition, Lambert Academic Publisher, pp. 124-129, ISBN-10: 3659111708.

2/28/2013