Beyond the Touch: a Web Camera based Virtual Keyboard

Časlav Livada, Miro Proleta, Krešimir Romić, Hrvoje Leventić
University of J.J. Strossmayer, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek
Kneza Trpimira 2B, HR-31000 Osijek, Croatia
clivada@ferit.hr

***Abstract*—This paper presents the concept of web camera based virtual keyboard. In order to achieve full functionality of the proposed concept digital image processing methods are being used, such as binarization, morphological operation and filtering. Real time image acquisition is performed by a mid-range quality web camera in order to test the sturdiness of algorithm design. The objective of this research paper is to identify the method of simultaneous fingertip and keyboard character recognition in order to produce a fully functional keyboard for user input.**

***Index terms*—** real-time image processing; object detection; character recognition; fully functional keyboard

I. Introduction

A. Monochromatic image binarization

In order to transform the color image into binary image, all chromatic components have to be removed. Thus only leaving grayscale values, i.e. pixels having values from $0-255$ where 0 is the brightest part of the image an 255 the darkest part. This is performed by the following equation:

$$0.2989 * R + 0.5870 * G + 0.1140 * B \tag{1}$$

Next step is filtering with Median filter which is making a reconstruction X^* of the degraded image Y in that way so that every pixel in X^* is gained by [1]:

$$x^*_{i,j=med(y_k)}, \forall\,(i,j) \in \Omega, k = i * N + j \tag{2}$$

where y_k represents the structural element. Result of the equation can be seen on Figure 1.

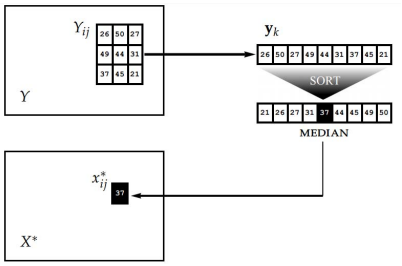


Figure 1. Structural element

In order to finish the binarization process, a threshold must be defined above which all pixels are turned black and all pixels below the threshold are turned white. In this research Otsu threshold determination method is used. This method assumes that the image consists of two pixel categories, details and background.

Threshold t is shown on Equation 3 [2] by maximizing the function σ_B^2 from set of pixels $x_0\,[0,1,\ldots,t-1]\,, x_1\,[t,t+1,\ldots,L-1]$ where L is level number, r_q is intensity level, p_q is pixel intensity and t is threshold value.

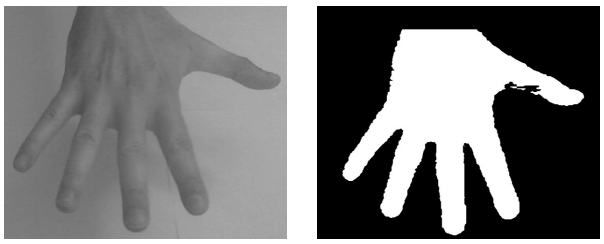
$$\sigma_B^2 = \omega_0\,(\mu_0 - \mu_T)^2\,\omega_0\,(\mu_1 - \mu_T)^2 \tag{3}$$

where equation coefficients are

$$\omega_0 = \sum_{q=0}^{k-1} p_q\,(r_q) \qquad \omega_1 = \sum_{q=k}^{L-1} p_q\,(r_q)$$

$$\mu_0 = \frac{\sum_{q=0}^{k-1} q p_q\,(r_q)}{\omega_0} \qquad \mu_1 = \frac{\sum_{q=k}^{L-1} q p_q\,(r_q)}{\omega_1}$$

Using Otsu method the relevant part of the binary image is extracted. Result of monochromatic image binarization on human hand image can be seen on Figure 2. Binary image on Figure 2, (b) was made using the median structural element $[3,3]$, stretching of grayscale values and Otsu threshold determination method.



(a) Monochromatic image (b) Binary image

Figure 2. Binarization of image showing the human hand

B. Morphological operations

Morphological operations consist of dilation and erosion. Combining the order of performing the dilation and erosion leads to *closing* and erosion and dilation leads to *opening*. *Closing* is a procedure of joining separated regions that belong to the same object and removing the unnecessary pixels to prepare the image for edge detection and upcoming processing [3]. Main deficiency of *closing* is closing of the nearby objects

into one region and not removing the noise that occurs during the process. *Opening* is a method used for lowering the noise in the image. In the process of *opening* erosion is first applied and will make a gap between poorly-connected objects while dilation will restore edge pixels.

1) Dilation: Dilation is the process of adding pixels to the edges of the observed image objects and is performed when detected edges are not emphasized enough. Adding pixels leads to linking of regions of space smaller than the size of the structural elements, edge reinforcement, object increase and filling hole in objects. Dilation can be explained by Equation 4.

$$A \oplus B = \{z \in E \vee (B)_z \cap A \neq \emptyset\} \tag{4}$$

where A is binary image, E is Euclidean space, B is a structural element, B_z is a structural element with the origin in z.

Pseudo code of dilation:

Dilation3x (source, dilation image, mask)
if mask = 1
mask = {{255,255,255}
{255,255,255}
{255,255,255}}
Initialise pointers for pixels of source and dilated image
For every row of source image {
For every column of source image {
Matching pixel (row, column)
If matching pixel = *background* pixel (value 0) {
Set mask over him, if the pixel values of mask and source image match:
background put in front layer
Write corresponding pixel in dilated image

2) Erosion: Erosion is the proces of removing pixels from object edges. Removing the pixels, the gap between objects is growing and detection is easier. Edges of the observed object can be precisely detected but holes on object can occur which are corrected by dilation. Erosion is shown with Equation 5.

$$A \oplus B = \{z \in E \vee (B)_z \subseteq A \tag{5}$$

Pseudo code of erosion:

For every row of source image {
For every column of source image {
Matching pixel (row, column)
If matching pixel = *front layer* pixel (value 255) {
Set mask over him, if the pixel values of mask and source image match:
Matching pixel set to background (value 0)
Write corresponding pixel in eroded image

II. Web based virtual keyboard

Idea of this article is to present a method for keyboard input via recognizing location of keyboard keys and fingertips with webcam [4]. Web camera is attached to the monitor and is scanning the area in front of the monitor where the keyboard presented on Figure 3 is located.

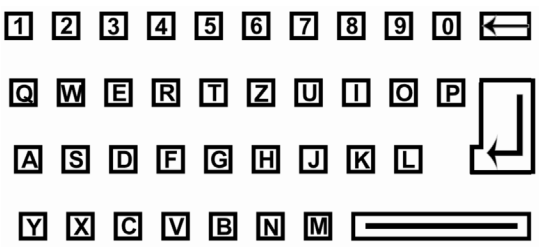


Figure 3. Keyboard

Camera is scanning the keyboard image situated on desk in front of the monitor. This algorithm has to single out the object, i.e. finger and locate the letters on the printed out keyboard. Isolation of region of interest (ROI), in this case, separation of the fingertip, location and mode of virtual keyboard is shown in next sections.

A. Fingertip separation

In order to separate the fingertip needed for key selection from the virtual keyboard erosion and dilation have to be applied on binary image of the human hand, Figure 2 b). Using erosion with disc structural element with the radius 20 px all regions that are smaller than the used structural element are destroyed. The value of 20 px is set manually and represents the average radius of the fingertip based on 10 different people. After this operation part of the human hand with fingers removed can be seen on Figure 4.

Figure 4. Region isolation

The next procedure is addition of pixels on the object edges, i.e. dilation. Dilation is used to fill the edges with the disc structural elements with the radius of 40 px. This value is set manually based on the test made during the coding process. Figure 5 shows object as a result of dilation.

Next step is to subtract object on Figure 5 from the object on Figure 2 b). The result of the region of interest extraction, i.e. fingertip extraction is shown on Figure

B. Blob analysis

After separating the region of interest and displaying the image, it is necessary to frame the desired area in order to access the necessary size information measurement. This operation is called *blob analysis* [5]. The starting point of the observed object is placed in the upper left corner and based

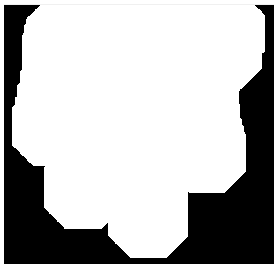


Figure 5. Object after dilation

Figure 6. ROI extraction

on the set origin, a boundary frame is drawn which can be rectangular, circular or polygonal. In this article rectangular boundary frame, i.e. bounding box is used. Observed from the boundary frame starting point, the width of the boundary frame equals the farthest coordinate of the object observed by the X axis, while the height equals the farthest coordinate of the object observed by the Y axis, Figure 7.

$$\begin{bmatrix} x_1 & y_1 & w_1 & h_1 \\ x_2 & y_2 & w_2 & h_2 \end{bmatrix}$$

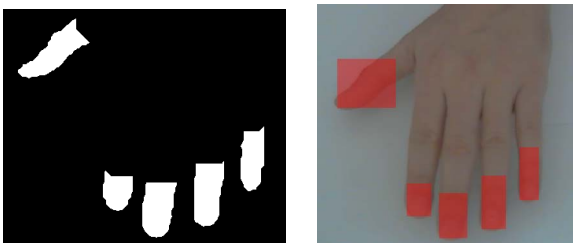
Fig. 7: Boundary frame

In order to place boundary frame on fingers, some parameters must be determined, like maximum and minimum area of the observed object, maximum number of objects on the binary image and the shape of the boundary frame. The area of the observed Object is the number that tells us how many pixels the object is made of. By setting the minimum and maximum surface of the object, some type of filtering of smaller or larger objects on the binary image can be performed. The boundary frame is displayed on all observed binary image objects, Figure 8.

The coordinates of the rectangular shape centroid are calculated according to the Equation 6.

$$x_{bb} = \frac{x_{min} + x_{max}}{2} \qquad (6)$$
$$y_{bb} = \frac{y_{min} + y_{max}}{2}$$

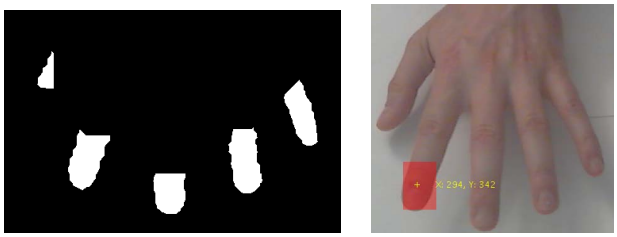
where x_{min} and x_{max} represent the closest and finest coordinate of the observed object per x axis, where y_{min} and y_{max} represent the closest and finest coordinate of the observed object per y axis.



(a) Extracted regions (b) Boundary box on fingers

Figure 8. Binarization of image showing the human hand

The maximum number of observed objects is set to 1 because only one finger is used to select letters from the keyboard. Observed object data is stored in the table and is required for keyboard functionality. Final images showing the isolated fingertips and one fingertip with bounding box are shown on Figure 9.



(a) Extracted fingertips (b) Extracted fingertip with shown centroid

Figure 9. Binarization of image showing the human hand

C. Keyboard recognition

The next step in creating a functional virtual keyboard is to recognize a single character as an object from sketched keyboard. Figure 3 shows the used keyboard and by using the morphological operation of *opening* on the keyboard image, every character is transformed into a separated object, Figure 10.

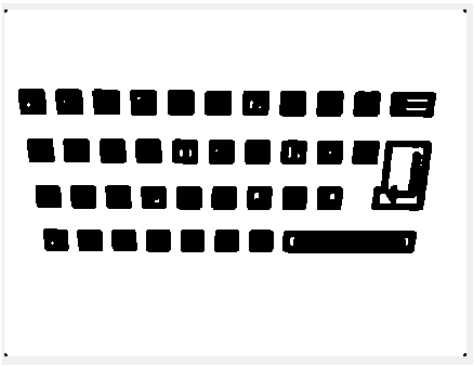


Figure 10. Keyboard with applied *opening* morphological oper-ator

Figure 10 shows that all characters and all objects are separated. By setting the boundary boxes on characters with blob analysis, information about height, width and location of each character is calculated and are stored in table shown on Table I.

TABLE I. AN EXAMPLE OF BOUNDARY BOX RECORDS

X	Y	Width	Height
25	114	39	36
37	182	38	34
48	246	37	33
59	307	36	31
76	114	39	36
87	182	37	34
96	247	37	32
106	307	35	32

where X and Y represent the coordinates of the x, y left upper corner of the object.

Figure 11 shows keyboard with set boundary boxes on top of isolated characters.

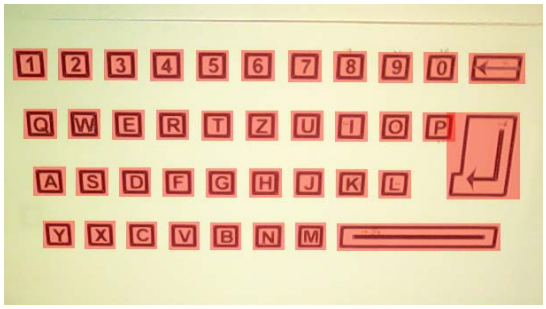


Figure 11. Bounding box records in table

D. Rectangular barrier overlap ratio

Fingertip and keyboard characters data is used for determining the *overlap ratio* given by Equation 7.

$$Overlap\ Ratio = \frac{area\,(A \cap B)}{area\,(A \cup B)} \tag{7}$$

where A is fingertip bounding box and B is keyboard character bounding box. Overlap ratio is calculated only if there is overlapping between the two bounding boxes. Overlap ratio is equal to the ratio of the cross section and union of two bounding boxes and the value can be 0 or 1. If there is no overlapping the value is 0 and if the fingertip covers the keyboard character, the value is 1.

E. Keyboard functionality

For full functionality of the keyboard, conditions are set to determine whether there is a overlapping between any character and fingertip. The default value of the *overlap ratio* is set to 0.1 due to the overlapping of fingertips from two fingers.

After the *overlap ratio* is met, a timer is called before the keystroke simulation is performed. The counter is used used because it is difficult to identify the trigger, i.e. pressing of a particular character via a web camera. If the counter had not been used, the hovering of the finger above each character would simulate the pressure and undesired letters would be printed out. Counter counts for 1 second and calls a function that will once again check for *overlap ratio* between the same character and fingertip. If the *overlap ratio* is still present, a *java.awt.robot* function is invoked that can print the character from the selected character in any Windows interface program. After the character is printed out, the flag status is set for each character in order to avoid replicating the pressure simulation.

III. CONCLUSION

In this paper a virtual keyboard that works via a web camera is presented. The webcam is positioned on the monitor is capturing the area below the monitor where the keyboard is printed on the paper. By positioning finger above a letter of a printed keyboard, a letter appears in the selected program on computer. The operating principle is shown on Figure 12.

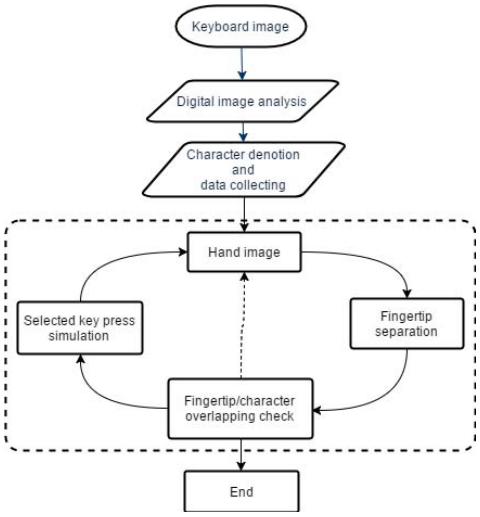


Figure 12. Virtual Keyboard Diagram

One drawback of the virtual keyboard design method is the character acquisition speed and the sensitivity to the room lighting. These problems will be dealt with in the future iterations of research.

REFERENCES

[1] B. Novoselac, V.; Zovko-Cihlar, "Image noise removal by vector median filter," *Proceedings ELMAR-2012*, pp. 57–62, 2012.

[2] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, man, and cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[3] B. Jain, R.;Kasturi R.; Schunck *Machine Vision, McGraw-Hill, New York*, 1995.

[4] M. Kölsch, M.; Turk, "Keyboards without keyboards: A survey of virtual keyboards," *Proceedings of Sensing and Input for Media-centric Systems*, 2002.

[5] O. Marques *Practical Image and Video Processing Using MATLAB, John Wiley and Sons*, 1995.