# Real Time Fingers and Palm Locating using Dynamic Circle Templates

Mokhtar M. Hasan
Computer Science Department
Faculty of Science
Banaras Hindu University

Pramod K. Mishra
Computer Science Department
Faculty of Science
Banaras Hindu University

## ABSTRACT
Real time and interactive systems require a high speed processing of input images or input signals and the response should be within acceptable time limit, these systems must have a remarkable response time for their trained actions and for that reason it is called real time systems, we have proposed in this paper a novel approach for real time fingers and palm detection by using the dynamic circle templates for capturing the hand structure, we have captured the structure of the hand object which includes the fingers and palm and we also located the fingertips , finger bases and palm center as well as the structure of each, we have sort the fingers sequence and have been indexed properly from left to right using our novel finger sorting algorithm, our proposed algorithm has shown a significant accuracy as well as the time required for this operation which is 82 milliseconds for fingers/palm detecting out of segmented hand object, our proposed algorithm can detect the fingers without any prior assumption for hand direction and without any limitation for the number of fingers used or their poses as well.

## General Terms
Gesture Recognition System, Template Matching, Geometric Features.

## Keywords
Finger Tracking, Palm Tracking, Finger Detection, Palm Detection, Hand Gesture Recognition, Interactive Systems.

## 1. INTRODUCTION
In order to reduce the burden of learning how to operate particular devices, control over our home appliances, playing a video game with our bare hands, entertainment and augmented reality [1], gesture and voice recognition are considered as promising alternative approaches to surmount such difficulties and the human-device interaction will be similar to human-human interaction which is done through the human organs like hand gesturing, head movement, face expressions, voice communication and overall body pose.

Human hand has intensive mount of information which can be used for achieving acceptable recognition; these information needs some lopping operations to extract principal components that have a great affection on a correct classification process and abandon the information that has a small impact. All the algorithms that applied hitherto are trying to simulate the human eye-brain ability ocular-based language tracking and recognition which is a silly task for this creature, from this point, plenty of algorithms have started to track and detect these fingers.

Hand gesture systems have been classified roughly into two major categories, which is vision based and glove based systems, each one can be used to build 2D or 3D model, a 3D estimation out of 2D cluttered image is done in [2]; they consider this problem as indexing problem and a large database is used for this purpose to find the corresponding 3D matching of 2D presented gesture.

Human gesture recognition has many challenges, illumination, rotation, translation, and scaling challenges, illumination challenges can be cured by using of normalized RGB as well as converting the RGB color model to other models that have brightness parameter which can be ignored to get rid of this illumination problem such as HSV, YCbCr, HSL and YIQ. Translation and scaling affections are dependent on correct segmentation/clustering of the hand object and can be cured by trimming and fixing the image to a standard scale respectively, rotation affection is considered of the major problem and need a special consideration, many systems have been built by employing a large number of samples per posture for overcome of rotation problem; 7*40 total number of gestures are used as training patterns in [3], tenfold of training patters are used in [4], and hundreds of images is used in [5], thousands of training gestures have been employed in [6], this increasing number of training patterns can be reduced by unifying the direction of the hand posture as suggested by us in [7]. The latter challenge can be classified as posture challenges [8], other kind of challenge is the system challenge which is the speed of the system, the reaction should be with acceptable time [8] for the reaction commenced and so, the system can be classified as real time system in this case.

We have applied a novel approach for hand fingers detection and tracking using dynamic circle templates and then we have extracted the fingers of the hand and the palm as well, our system achieved 82 milliseconds for detecting the hand fingers from the segmented hand object.

## 2. RELATED WORK
The authors in [9] has used the color information mixed with motion information for extracting of hand region, the fingers have been detected by looking for the high value curvature since the open fingers has this property, so, by looking for this angles they classified the fingers, in [10] the fingertips are detected by locating the hemisphere cap, and the palm center is located by finding the point that has the maximum distance to the closest region boundary.

In [11], the hand image is segmented by using of histogram-based skin classifier, after that; the fingertips are located by

finding the pixels that represent the peaks along the contour premises, in [12], a search window has been decided for hand fingers instead of expansion the search along the hand arm, so, they hand fingers should be within this window for continue of processing, he has decided the hand palm center in the same as in [10], and the fingertips also the same. In [13] the fingers have been detected by using of both of the Kohonen Self-Organized Feature Map (SOFM) and the Growing Neural Gas(GNG), which has the property of finding the structure of a shape, and in this case, hand shape and fitting the hand shape all over with the neurons, this paper has a limitation of using left hand only with up righted direction.

## 3. MORPHOLOGICAL OPERATIONS

Morphology operations consist of two basic artifacts, dilation and erosion [14], these two operations represent the spine for the morphological operations and any other named operations are derived from those.

Let us start with the formal definition of these operations, the best judge that can tell us about the meaning of these two terms is Oxford Definition [15], the dilation is defined there as "to become or to make something larger, wider or more open", and the erosion operation is defined there as "to gradually destroy the surface of something through the action of wind, rain, etc", these definitions are in general, for image processing, the object lied inside the image can be dilated and eroded by the impact of a filter which called structuring element (SE), which dilates and erodes the object in respect of a reference point which controls the trigger of these operations.

### 3.1 Morphological Operation Notations

Let $A \in Z^2$, which is the 2D space of the (x, y), and also let $B \in Z^2$ be the SE which controls the structure of the morphological operations, then for any binary image I where $A \subseteq I$ , dilation and erosion can be defined as in (1) and (2) respectively.

$$A \oplus B = \left\{ z \mid [(\widehat{B})_z \cap A] \subseteq A, \exists\, b \in B \right\} \quad (1)$$

$$A \ominus B = \{ z \mid [(B)_z \cap A] \subseteq A,\ \forall\, b \in B \} \quad (2)$$

Where $\oplus$ and $\ominus$ are represent the dilation and erosion respectively, formula (1) can be interpreted as the reflection of the SE B should hit A somewhere, at least one point, whereas formula (2) is interpreted as the SE B should hit A at all its points, which means should be fully contained by A, where $\widehat{B}$ is the reflection of B and this reflection is alternative for complement operation, this reflection can be written as (3):

$$\widehat{B} = \{ w \mid w = -b,\ \forall\, b \in B \} \quad (3)$$

And $(B)_z$ is the shifting of B by z offset, and can be written as (4):

$$(B)_z = \{ w \mid w = b + z,\ \forall\, b \in B \} \quad (4)$$

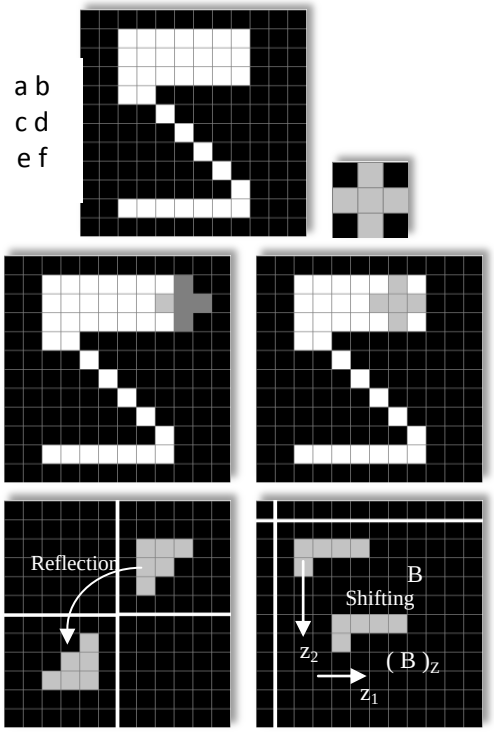
For $Z^2$space, the definition can be expanded to (5) and (6):

$$\widehat{B} = \{(w_1, w_2) \mid (w_1, w_2) = (-x, -y), \forall\, (x, y) \in B \} \quad (5)$$

$$(B)_z = \{(w_1, w_2) \mid (w_1, w_2) = (z_1, z_2) + (x, y),\ \forall\, (x, y) \in B \} \quad (6)$$

Ocularly speaking, Figure 1 shows the dilation, erosion, reflection and shifting respectively.

We have employed the morphological operations for improving an existing algorithm for 2D hole filling and we have applied this hole filling (hand interior noise) after the segmentation operation for preparing a noise-free hand area.




a) original image, b) structuring element, c) dilation done by at least one hit, d) erosion is done by completely containing, e) reflection, f) shifting.

**Figure 1.Illustrating of some Operations.**

## 4. HAND SEGMENT EXTRACTION

In this step, the hand is extracted as one object from the input image, we have applied image segmentation using color based model for human skin detection by RGB to HSV color space conversion in order to extract the hand object from a cluttered background, this segmentation algorithm has a significant property which is color seeking algorithm, we have chosen a specific values for each of H and S components for human-skin pigment detection, the following formula (7) shows the application of the selected segmentation method.

$$I(x,y) = \begin{cases} \text{hand cluster} & \text{if } 0 \le H(x,y) \le 0.1228 \\ & \text{and } 0.2666 \le S(x,y) \le 0.8777 \\ \text{background cluster} & \text{otherwise} \end{cases} \quad (7)$$

Where I(x, y) is the input image, H(x, y) is the hue value of the corresponding pixel and S(x, y) is the saturation value of the same for HSV color space. Figure 2 shows the segmentation operation of a cluttered background image.

## 5. INTERIOR NOISE REMOVAL

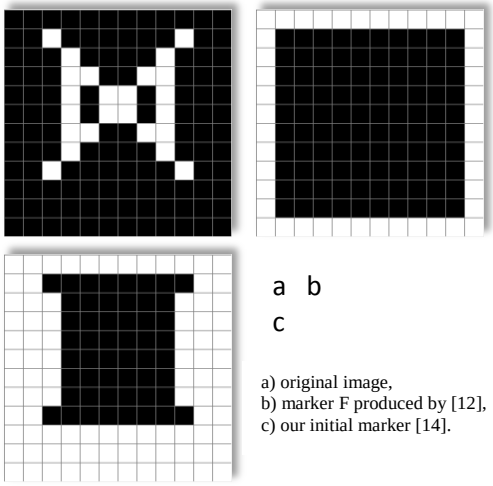
We can define the hand interior noise as a background pixel within the premises of a hand object, all the segmentation operations do not produce a perfect object, some noises are interposes, so, in order to covers up this shortage and makes the hand object with no interior noise; we have applied Speed Border Image Initial Algorithm (SBIIA) which is suggested by us [16], this algorithm is applied by using the reconstruction operation of the morphological operations with an initial marker employing, we have applied this algorithm using Equations (1), (5) and (6) for dilation operation by the help of reconstruction technique, this algorithm has the operation done with a remarkable speed, this algorithm was a

modification for the algorithm found in [14] which is BIIA; we have modified this algorithm in two different places, the initial marker is improved so that the convergence will be more faster and second place by using dynamic SEs technique instead of one SE, which are threshold SE and non-threshold SE, these two SE's have solved a concealed problem at the corners when there is a diagonal border for the object, our initial marker takes the following form (8).

$$F(x,y) = \begin{cases} 1 & \text{if } \mathrm{BorderPath}(x,y) \text{ is true} \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

Where $\mathrm{BorderPath}(x,y)$ is true if there is a background pixels from (x, y) to any of the two horizontal end edges of the image, which are (0, y) or (width-1, y), and is false if such direct path does not existed, Figure 3 illustrates this idea by presenting the marker F produced by the original algorithm and our marker for a given image.

**Figure 2.HSV segmentation operation.**



a b
c

a) original image,
b) marker F produced by [12],
c) our initial marker [14].

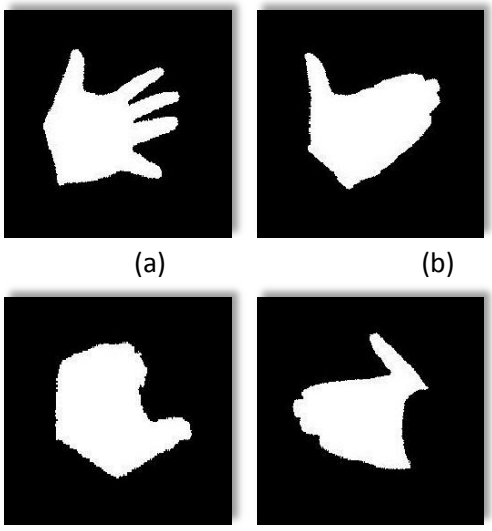
**Figure 3.The Initial Marker for a Given Image.**

After that initial marker, we have applied equation (9) [14] for construction operation in order to find the holes of a given object.

$$H = [R^{D}_{A^c}(F)]^c \quad (9)$$

Where H denotes the final image output, R is the reconstruction operation, D is the dilation operation, A is the input image, and F is the marker that will be updated in each epoch of the algorithm, for more details about how this method has been applied, you can refer to [16]. Figure 4 shows several hand posture samples, Table 1 shows comparison in a time factor between the two algorithms.

As seen, SBIIA is faster and can go along with real time applications, and this speed can be enhanced as the image size reduced. Figure 5 shows a text image taken as an example for application of our SBIIA algorithm.



(a) (b)

(c) (d)

**Figure 4.Four Image Samples after Filling Operation**.

**Table 1: Time Factor Performance for BIIA and SBIIA.**

| Input | Dimension | BIIA (millisecond) | SBIIA (millisecond) | speed ratio % |
|---|---|---|---|---|
| **Fig. 4a** | 250x250 | 675 | 318 | 47 |
| **Fig. 4b** | 250x250 | 459 | 262 | 57 |
| **Fig. 4c** | 250x250 | 606 | 87 | 14 |
| **Fig. 4d** | 250x250 | 550 | 54 | 9 |

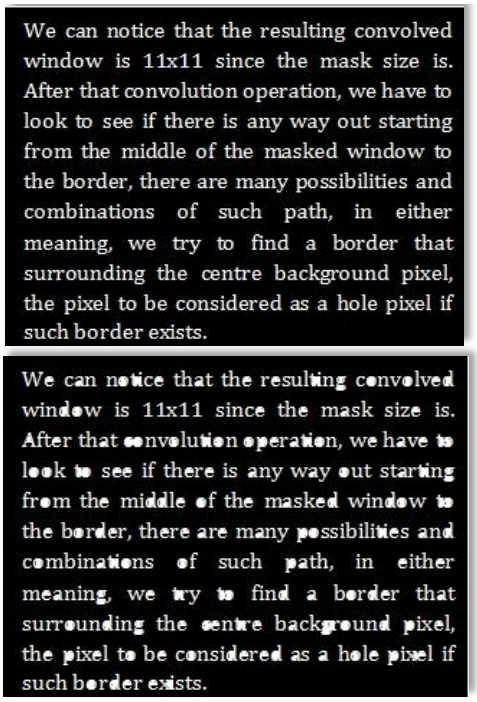



**Figure 5.SBIIA Implantation on Image Document.**

# 6. FINGER DETECTION

The human hand consists of two main components, palm and five fingers, as noticed from Figure 4a, the shape of fingers is different from the palm's shape, and this difference has enabled us to formulate a new algorithm for fingers/palm detection and locating, and this can enable us to use of geometric features for latter stages of recognition purposes.

Our point that inspired from morphological erosion operation is, the width of the fingers is differ from that one for palm, and the circle that fits and goes inside the palm cannot be inside the finger, so, we have used the two different circle templates that can penetrate inside the finger and/or palm in order to decide the fingers and palm area as well.

## 6.1 Dynamic Circle Templates

We have applied two different circle templates, the first one is used to decide the fingers and the another one for palm, the size of these templates are decided according to picture size, if we trim and scale the image to be a specific standard size; circle templates size can be unified, let us call them as FCT (finger circle template) and PCT (palm circle template), the size of FCT could be any size that fits the small finger, while the PCT size should be larger than FCT and should not fitsat finger's area, Figure 6 shows a selection of FCT and PCT.

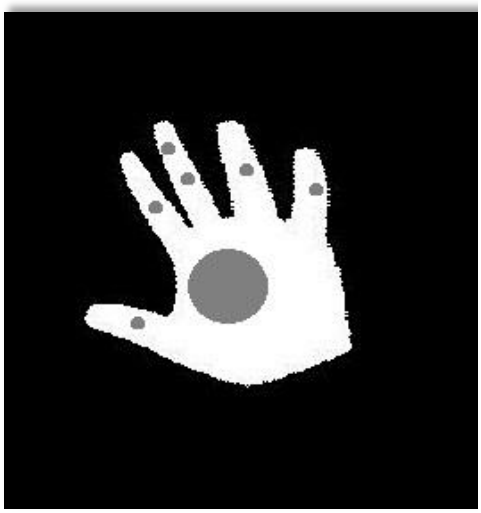


**Figure 6.The Infiltrating of FCT and PCT.**

As seen by Figure 6, the big gray circle represents the PCT that can interpose the palm without any affection on fingers area, on the other hand, the FCT can infiltrate the finger premises easily, but there is one small problem which is they can do the same with palm premises as well, this problem is addressed later.

Both of FCT and PCT take the circular form which has no angle or direction like rectangle or triangle, Figure 7 shows FCT and PCT with diameter 9 and 21 respectively and this diameter should be odd for circle balance purpose and the reference point can be localized as the center which will employed in the feature calculations stagesince each dot is formed by its center point (reference point).

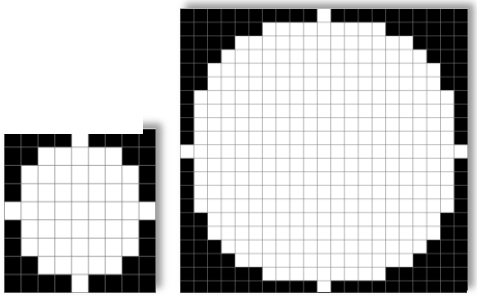
## 6.2 Infiltrating Hand Object

After selection of the FCT and PCT, these two templates should be applied on the hand object; we have chosen a FITmatching mode in which the matching is true if the template is completely contained by the hand object as seen by Figure 6.

### *6.2.1 Infiltrating With FCT*

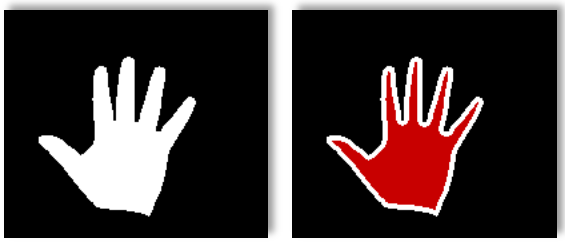
At this step, the entire hand object will be processed under the FCTmatching; the result of this process is shown in Figure 8.

As seen by Fig.8b; this shows the output after applying FCT, but, we can speed up this operation by redefining the template matching operation and converting this overlapping filling tonon-overlapping filling which in turns reduces the number of pixels that will be processed later and speeds up the processing; which means the output for each template will be the reference point (center point) rather than whole template, and any other matching within this matched template will be neglected, the procedure of matching is that the reference point of new template should lies outside any other matched template, Figure 9 shows the result of reference point filling (non-overlapping template matching at reference point).



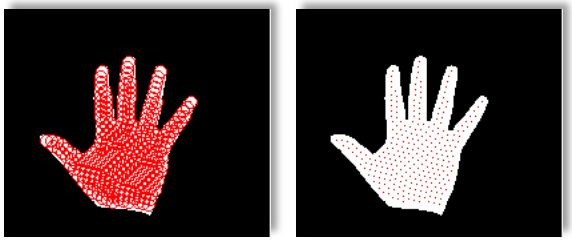
a) FCT, b) PCT.

**Figure 7.The Structure of FCT and PCT.**



a) segmented handb) fitted hand.

**Figure 8.Fully Template Matching for FCT.**



a) circle matching b) dotting by circle centres

**Figure 9.Dotted Application of FCT.**

As seen by Figure 9, whenever a new point is considered for matching, this point should lie outside of any other matched templates hitherto; this can be written as (10):

$$\text{matching}(A, T) = \{\, \text{rp}(z) \mid A^c \cap T = \emptyset \text{ and space}(z) \} \quad (10)$$

Where

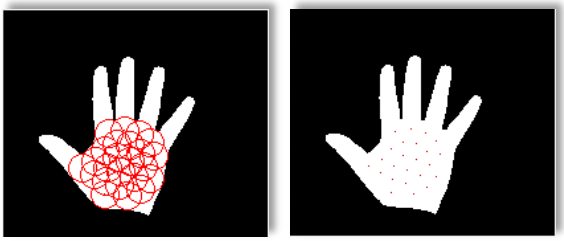
$$\text{space}(z) = \{\text{abs}(z_i - z_j) \geq r |\ z_i, z_j \in z, \forall\, i \neq j \text{ and } z_i = (x_i, y_i)\} \quad (11)$$

Where radius r in (11) is the radius of the FCT, and Figure 9b came from the collecting of the reference points for the corresponding matched FCT.

### 6.2.2 *Infiltrating with PCT*

Now, we have to apply PCT in order to locate the palm area by applying the latter equations (10) and (11) on the segmented image but of course with r of PCT, the result will be shown in Figure 10.

As seen by latter figure, the palm area has been located and we have applied the same latter equation with PCT's respective radius as we mentioned.



a) input hand b) PCT application

**Figure 10.PCT Application on Hand Object.**

### 6.2.3 *Merging Operation*

After application of FCT and PCT, a merge operation must take place; this merge operation is done to decide the fingers premises, notice that the result of application of PCT must dominate and control the merge operation, and these palm dots should oust the finger(s) dots according to a specific threshold which is the radius of the PCT incremented few number of pixels to push the FCT dots away inside their fingers, this merging operation can be written as follows (12):

$$\text{hand} = \text{merge}(P, F) = \{ Z = (P \cup F) | \text{abs}(p_i - f_j) > r, \forall I, j \} \quad (12)$$

Where P is the palm circles and F is the finger circles and r is the ousting threshold, this formula controls the selection of the fingers dots, while taking all the palm dots since the palm dots cannot infiltrate within fingers premises but the opposite is not true; this is the solution of the problem addressed in section 6.1, Figure 11 shows the result of merging operation between each of Figure 9b and Figure 10b with keeping all the points of latter mentioned figure.

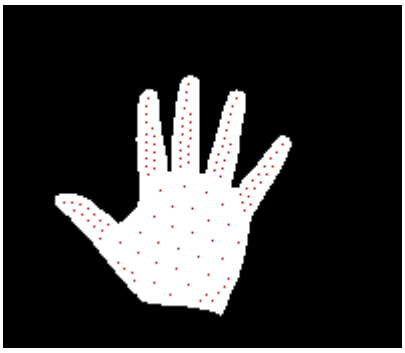


**Figure 11.Merging of Selected Fingers Dots to Palm Dots.**

We can improve the speed of the sections (6.2.1, 6.2.2, and 6.2.3) by applying the PCT first, at this point we have the palm area covered, and then we apply the FCT with fingers dots ousting at the same time if they trespassing the palm premises, and in this way we have applied both sections (6.2.1, and 6.2.3) in one basic loop; this can be done be rewriting (11) as in (13).

$$\text{dotted(Fingers)} = \{z_i \mid \text{abs}(z_i - z_j) \geq r_j, \forall\ z_j \in I, \text{ and } i \neq j\} \quad (13)$$

Where $z_i = (x_i, y_i)$ and $r_j$ is the radius of $z_j$ and I is the input image and $z_j$ represents the stored reference point for matched palm/fingers and $z_i$ are the new presented FCT's reference point.

## 6.3 Finger Locating

After this point, we have to construct the fingers from the final collected data hitherto, this is done by labeling the finger dots according to their distances from each other and any two finger dots can be union together in one single group if the reference point of one of them touches the border of other FCT, but, however, some unwanted groups will come out, Figure 12 shows the potential groups coming out of Figure 11.

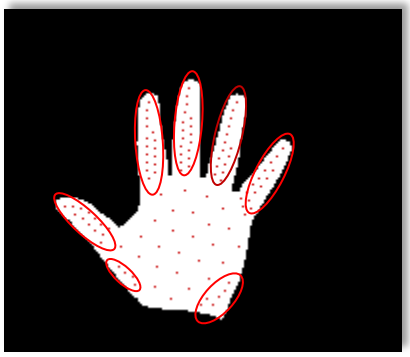


**Figure 12.Finger Dots Labeling.**

As noticed above, our algorithm will produce 7 groups for that particular image, and this number is varying depends upon the image shape, however, let us assume the fingers occupied 1 to 5 fingers number, groups 6 and 7 should be neglected, we can remove these unwanted groups by two approaches:

(1) Considering number of dots in a group: calculating number of points in each group and by applying thresholding technique to remove the non-finger groups which heir number of dots less than predefined threshold.

(2) Considering group trend length: calculating the farthest group point from the nearest touched palm dot, this distance should be greater than specific threshold for deciding the finger group; otherwise, this is not a finger and should be neglected, it is worth to mention that the distance should be calculated from the nearest touched palm dot and not from the palm center since if we consider the palm center the calculated distance will not reflect the finger length which is the threshold value.

The first technique yields a bad classification especially with small size photos especially with the thin fingers which produces small number of dots, but, however, the second technique more reliable and produces correct finger labeling. The idea of the second technique can be written as the following algorithm in which the palm center is considered

for calculating the max distance and min distance is calculated from the nearest touched palm dot:

**Algorithm 1:** Removing non-finger groups.

**Input:** $(x_i, y_i)$ for each of palm and fingers-yet groups,G is the total number of fingers-yet groups, $T_g$ is the total number of dots in group g, threshold for distance.

**Output:**removing of non-fingers group.

**Method:**

**Step1:** *[calculate the center of the palm area]*

$$(xc, yc)_{palm} = \frac{(\sum_i x_i, \sum_i y_i)}{n}, \forall (x_i, y_i) \in palmdots$$

**Step 2:***[find the max distance of each group point from that center]*

$$D_g = \max_{\forall i \in T_g, g \in G} \sqrt{(x_{g,i} - xc_{palm})^2 + (y_{g,i} - yc_{palm})^2}$$

**Step 3:***[find the min distance from the group points calculated in step 2 from the nearest palm point]*

$$d_g = \min_{\forall i \in palmdots\ , g \in G} \sqrt{(x_{D_g} - xi_{palm})^2 + (y_{D_g} - yi_{palm})^2}$$

**Step 4:***[deciding the finger group]*

if $(d_g > threshold)$then announce this as a finger group

else neglect this group

**Step 5***: [stop]*

Stop.

The threshold for Algorithm 1 has been set to 25. Figure 13 shows the graphical application of Algorithm 1.

As noticed in Figure 13, the groups that have the distance 13 and 18 will be removed since they not represent any finger and the other groups will survive. So, after the application of Algorithm 1, Figure 14 shows different samples.

## 6.4 Finger Tips and BasesLocating

Now, we have collected the information regarding the palm and finger areas respectively, we guess it is clear and obvious how to get the finger tips and bases from these collected data.

Mathematically speaking, this can be accomplished by finding out the farthest and closest group dots compared with the palm center, which can be written as (14) and (15):

$$Tip_g = \max_{\forall i \in T_g} \sqrt{(x_{g,i} - xc_{palm})^2 + (y_{g,i} - yc_{palm})^2}$$

$$\forall g \in G \quad (14)$$

And

$$Base_g = \min_{\forall i \in T_g} \sqrt{(x_{g,i} - xc_{palm})^2 + (y_{g,i} - yc_{palm})^2}$$

$$\forall g \in G \quad (15)$$

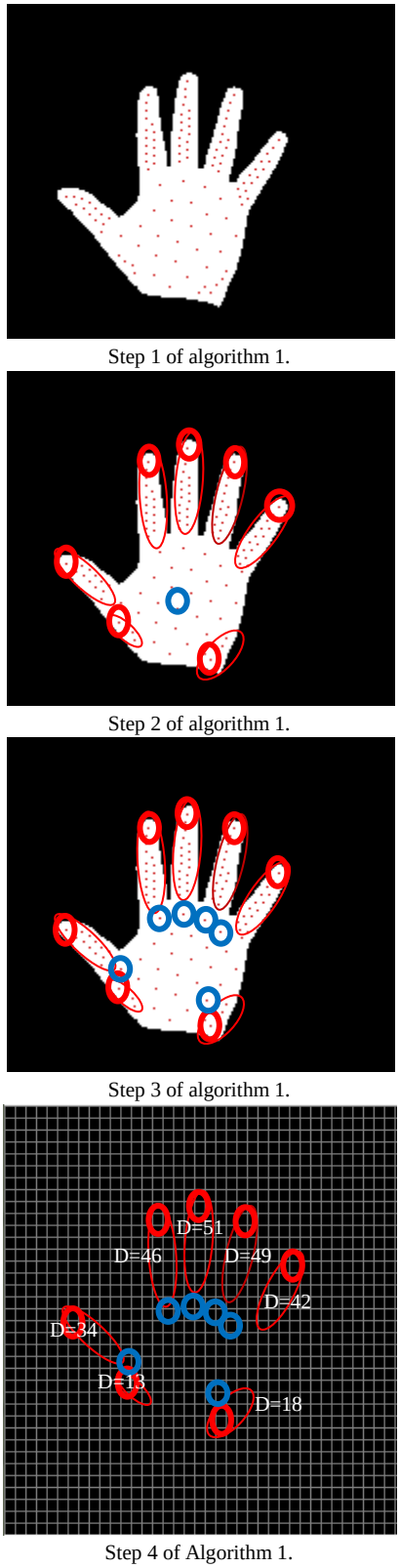
However, there is another way for calculating the finger base, which is by finding the difference between the fingertip and finger center, and that difference will work as a guide toward the finger base by reflecting it around the finger center which will be considered as original point axis, this can be written as (16):

$$Base_g(x, y) =$$

$$Centre_g(x, y) + R[Tip_g(x, y) - Centre_g(x, y)] \quad (16)$$

Where $R[B] = (\hat{B})$ (17)

The (17) is defined as in (3).Where g is the group number, and $T_g$ is the total points in group g, the following figure shows the finger tips and bases that have been detected by our algorithm by using the two latter techniques for base locating, Figure 15 shows some samples with the two different tools as mentioned above.




Step 1 of algorithm 1.

Step 2 of algorithm 1.

Step 3 of algorithm 1.

Step 4 of Algorithm 1.

**Figure 13.Application of Algorithm1.**

## 6.5 Fingers Sorting

The sequence of the collected fingers are subject to first in line-dot encountered during the FCT application, i.e. the output is not in ordered, since the input hand can be in any rotation angle and same pose may take different finger sequence, Figure 16 shows some hand postures with their corresponding sequence produced and Figure 17 gives more comprehensive details about how the initial finger sequences come like Figure 16 since the processing direction is row major.

So, after this brief explanation of the initial fingers order, now, we need to find out the correct order of the fingers and re-sorting the fingers according to the natural hand sequence of fingers, we have sorted the fingers from left to right so we can unify the latter extracted fingers features and this unification will help us for fingers/hand pose classification, for example, consider Figure 16(a and d), our sorting will give the thumb finger (first finger on the left side) zero number which means will be the first finger in the fingers sequence regardless of hand rotation angle, even the hand is upside down like Figure 16d, so, this will unify the latter extracted features.

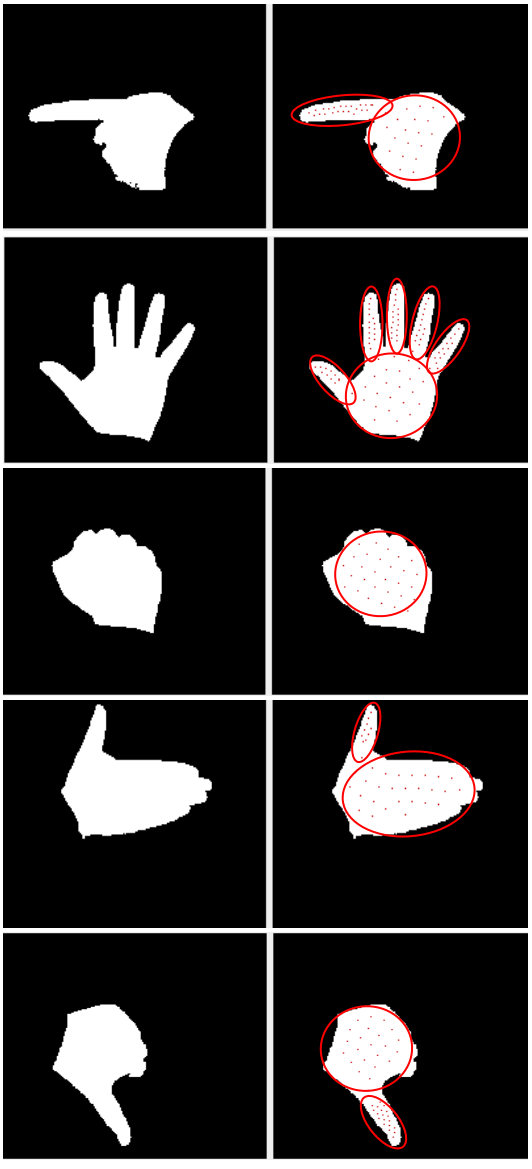
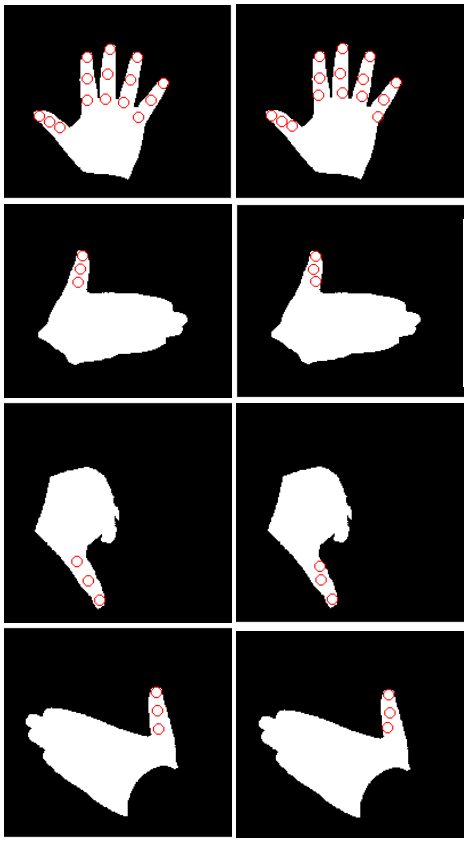


**Figure 14.Input Image and its Corresponding Labeling.**




a, c, e,and g are calculated using distance measure; b, d, f,and h are calculated using reflection measure

**Figure 15.Finger Tip, Centre, and Base Locating.**

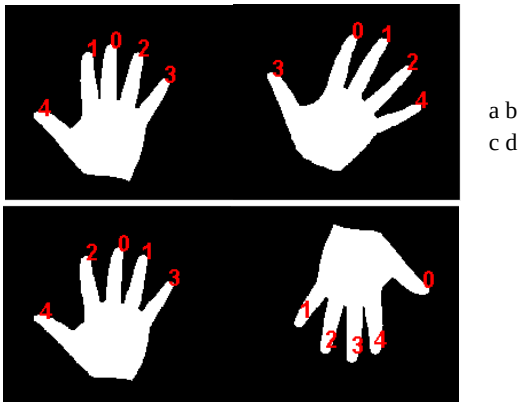



**Figure 16. Fingers initial Sequence.**

We have developed a new algorithm for sorting these fingers which is Fingers Sorting Algorithm (FSA) as shown later. We have considered a practical example by application of FSA using the hand object of Figure 17; we have to show the finger base locations and the palm center as calculated from the latter mentioned figure in Table 2.

**Table 2: (x, y) Coordinates for the Finger Bases and Palm Center for Figure 17.**

| Finger number | x | y |
|---|---|---|
| **0** | 89 | 83 |
| **1** | 73 | 84 |
| **2** | 105 | 86 |
| **3** | 118 | 99 |
| **4** | 49 | 108 |
| **Palm Center** | 89 | 114 |

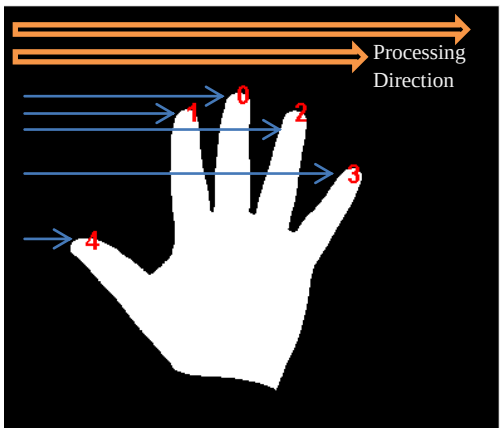



**Figure 17.Encountering of Fingers Dots.**

Then, by applying steps(1-4) of FSA algorithm, we can get the following matrices in Table 3, 4 and5 as follows:

**Table 3: Distance Matrix from Step 1 of FSA.**

| Finger number | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 0 | 0 | 16 | 16 | 33 | 47 |
| 1 | 16 | 0 | 32 | 47 | 33 |
| 2 | 16 | 32 | 0 | 18 | 60 |
| 3 | 33 | 47 | 18 | 0 | 69 |
| 4 | 47 | 33 | 60 | 69 | 0 |

**Table 4: Opposite Matrix fromStep 2 and 3 of FSA.**

| Finger number | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 0 | -16 | 16 | -16 | -14 | 14 |
| 1 | 16 | -16 | 16 | 14 | -14 |
| 2 | 16 | 16 | -16 | -15 | 13 |
| 3 | 17 | 15 | 18 | -18 | 9 |
| 4 | 31 | 33 | 28 | 22 | -33 |

**Table 5: Neighborhood Matrix fromStep 2 and 4 of FSA.**

| | 0 | 1 |
|---|---|---|
| 0 | 1 | 2 |
| 1 | 0 | 4 |
| 2 | 0 | 3 |
| 3 | 2 | -1 |
| 4 | 1 | -1 |

So, Table 3 gave us the first neighbor of each fingers which have different cell color, table 4 gave us the second opposite neighbor which is not necessarily the second minimum value from the finger, you can check finger 1 (index finger) in table 3; his first neighbor is the middle finger(finger 0), but the second neighbor should be in opposite side of this first neighbor and if we consider the second minimum value which is 32 (from distance matrix); this will lead that the index finger has two surroundings which are middle and ring finger?! How these come? So, we have calculated this opposite matrix in order to overcome this issue, for now and as seen by table 4, the second neighbor finger for finger number 1 (index finger) is finger number 4 (thumb finger) which is correct, as the rest are the same, and we can notice that fingers numbered (3, and 4) has not corresponding in Table 3 which means they are border fingers.

Now, in order to determine which one of those 3 and 4 fingers are left and right most fingers, the their angles are calculated as mentioned in step 5 of FSA, which is as table 6:

**Table 6: Angles of each of the Border Fingers.**

| Finger | Angle |
|---|---|
| 3 | 333 |
| 4 | 188 |

This means that finger 4 has the minimum angle and he is the left angle of the hand object and the other finger is the right finger, Figure 18 gives more explanation about calculation of the fingers angle by using computer view for xy-coordinates, it is worth to mention the center coordinate point when calculating that degree is not fixed and it can be one of the four corner points of the image, this is decided by finding the sum of two vectors formed by the two border fingers since if we have just one finger's hand then will be no need for renumbering and the numbering starts from two or more fingers, then; after finding the sum of these two vectors; their sign (x vector and y vector) will decide the original coordinate pixel which will be one of the four corners as we mentioned. Now, Figure 18 shows these angles as calculated form the algorithm in case of the original coordinate pixel was palm center, and Figure 19 shows different samples after renumbering, one more note; we can consider the palm center as original point but this is not working in one case which is the hand is horizontal and pointing toward the right side because of that we have mentioned the vector remedy which has no shortcoming.

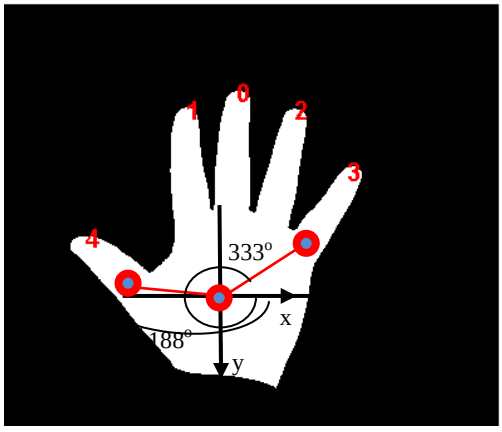



**Figure 18.Left and Right Fingers Discrimination.**

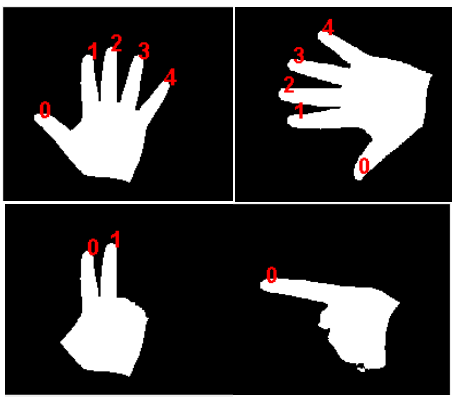



**Figure 19.Indexing of the Fingers.**

**Algorithm 2:** Fingers Sorting Algorithm (FSA)

**Input:** $\text{Base}_{\text{finger}}[t]$ and $\text{Centre}_{\text{Palm}}$, where t is thetotal number of fingers.

**Output:** Fingers[i] sorted

**Method:**

**Step 1:** *[calculate the distance matrix]*

$$\text{distance}[i][j] = \sqrt{(\text{Base}_{\text{finger}}[i] - \text{Base}_{\text{finger}}[j])^2} \quad \forall\, i, j = 1 \dots t$$

**Step 2:** *[choosing the first neighborhood for each finger i]*

$$\text{neighbourhood}_{i,1} = j \text{ index of}\left(\min_{\forall\, j \in t \text{ and } i \neq j} |\text{distance}_{i,j}|\right)$$

**Step 3:** *[calculate the opposite neighbor matrix for all i, j]*

$$\text{opposite}_{i,j} = \text{abs}(\text{distance}_{i,j}) - \text{abs}(\text{distance}_{\text{neightbour }i,1,j})$$

**Step 4:** *[finding the second neighborhood that should be apposite to first one for each finger i]*

$$\text{neighbourhood}_{i,2} = \begin{cases} j \text{ index of}\left(\min_{\forall\, j \in t \text{ and } i \neq j} |\text{distance}_{i,j}|\right), & \text{if opposite}_{i,j} < 0 \\ -1, & \text{otherwise} \end{cases}$$

**Step 5:** *[finding the left most and right most finger]*

$$\emptyset_i = \tan^{-1}\left(\frac{(\text{base}_i(y) - \text{centre}_{\text{palm }(y)})}{(\text{base}_i(x) - \text{centre}_{\text{palm }(x)})}\right), \forall\, i \in t$$

$$\text{left most finger index} = \text{index of}\left(\min_{\forall i \in t} |\emptyset_i|\right)$$

$$\text{right most finger index} = \text{index of}\left(\max_{\forall i \in t} |\emptyset_i|\right)$$

**Step 6:** *[sorting the fingers by tracking the neighborhood array]*

```
left=left most finger index
repeat the following for all i= 1, …, t-1
    right=the other parameter of neighborhood_left
    update Finger[i] data according to the left and right
        indices
    left=right
end repeat
```

**Step 7:** *[finish]*

Stop, Finger is ordered from left most finger to the last most finger.

# 7. EXPERIMENTAL RESULTS

We have applied our system using a computer web camera, we have captured several photos and we have applied our suggested algorithm, we have listed the performance speed as well as the results of the finger detection, our method is suitable for real time applications since it has a remarkable processing time of approximately 328 milliseconds per posture, this time includes segmentation operation which is applied by using HSV color space conversion, and also the hole filling algorithm which consumes more time, and the fingers/palm detection, we have captured nine different poses; all these poses are processed correctly, Figure 20 shows the original and the processed postures.

As seen by latter figure, we have marked the fingertips as well as the center of the finger, and the palm center, these labeling is done automatically by our software without any manual interference.

# 8. CONCLUSION AND FUTURE WORK

We have applied a novel approach for fingers/palm detecting and sorting, we have obtained the fingertip location, finger base location, finger center, and palm location using dynamic circle templates with non-overlapping matching for reducing the system processing time, with noise removal, the hand fingertips are located with their bases coordinates and the palm is located as well, we have traced the fingers and labeled them from left to right using our novel algorithm form finding the two neighborhood fingers, the first neighborhood finger is caught by minimum distance from each finger, while the other one is caught by finding the minimum distance in the opposite direction since the neighborhood fingers cannot be at the same direction, the following situation shows the reason behind this as in Figure 21.

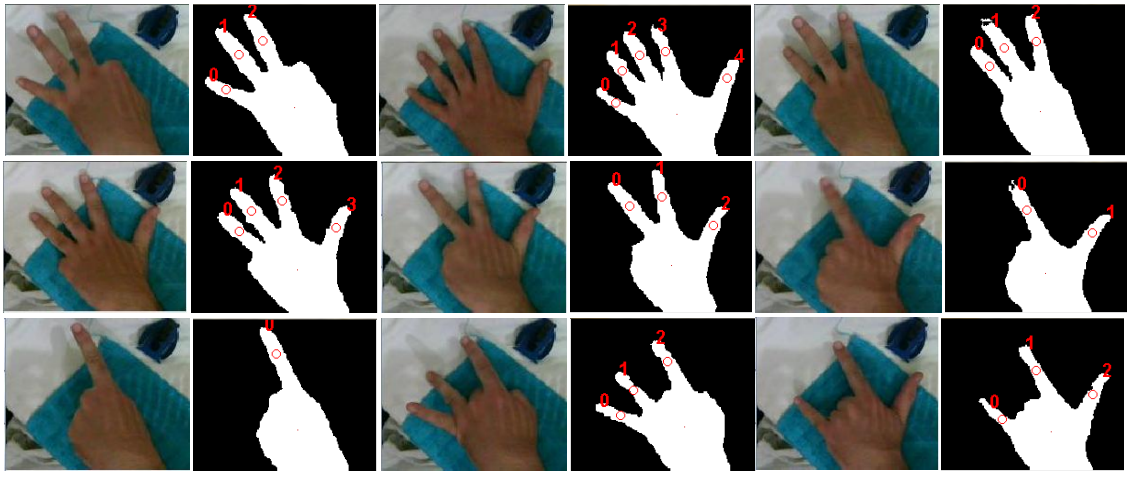
As seen by Figure 21, the nearest two fingers to the middle finger are ring and small fingers since d1 and d2 are less than d3, but those fingers are not the neighborhood of the mentioned finger since the actual neighborhoods are those correspond to d1 and d3 since they are in opposite direction, so, in our algorithm of FSA, we have considered the minimum distance from the middle finger which is the ring finger, and the other finger is the minimum distance but in the opposite side which is thumb since the index finger is folded.

However, after this sorting operation we have put all the fingers in a unified sequence from left to right by considering the two border fingers and calculating their angles, and the finger that corresponds to the smallest angle is the left most fingerand the other one is the right most finger, these angles are calculated depending on the original coordinate pixel that have chosen, this original coordinate pixel is selected by finding the sum of the two vector formed by those border fingers and depending on the x y sign of that resulted vector the original coordinate pixel will be decided, and all the other fingers are numbered accordingly, we have achieved average processing time was 82 milliseconds, and the average of the total running time for the three stages of the system was 328 milliseconds which is suitable for real time applications.

Our system can be applied without any limitation on hand direction, even we can use the left and the right hand as well, we can also apply some modification in order to use both hands and extract the fingertips as well, the modification can be done by making initial separation of the collected palm dots, after that we can go through labeling the finger dots and then; each group of finger dots can be attached to a certain palm according to the minimum error criteria, so, now we have two hands data and the processing will be the same for each hand.

Also our algorithm can be applied for finger tracking easily, by keep tracking of the current fingertips locations, the latter step is a traditional step [12] by calculating the distance between each two consecutive shots, and by assigning the nearest fingertip location the tracking will be done successfully.

And also the system can be applied for identifying the raised finger and to name each one of them, this accomplished by extracting the features from each finger's collected data and these features can be used for identifying purpose; we can define the most important feature suggested by us which is the distance between the projected base pixel on the hand main direction line and the palm center, this feature proved its uniqueness and discrimination and can be used with other feature to provide a robust algorithm for finger identification, this other feature can be the angle or distance between thumb base (after identification from the former feature) and other finger's bases and this feature can ensure to classify the fingers correctly, or between the small finger (after identification from the former distance feature) and other finger's bases.




Each pair represent the input image and the its correspondingoutput image after processing and featuresextracting and fingersindexing.

**Figure 20.Application of ourSuggested Algorithm.**

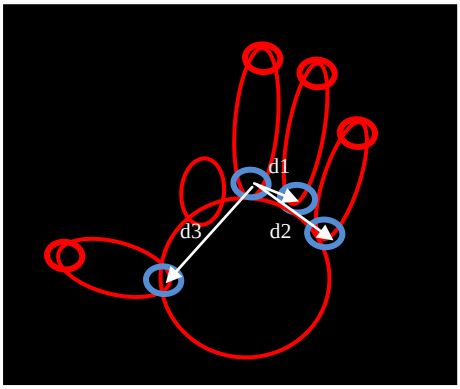



**Figure 21.The Reason behind Application of FSA.**

Other feature can be the angle formed between the line of the main direction of the hand and the line formed by base finger-palm center, this angle can be used as a classification feature as well. However, after identification of the raised finger, we can go with the hand recognition which will be an easy task since we have the number of fingers and the name of each one of them, this can be done by using a simple deterministic finite automata (DFA) as a classifier for considering all the combinations that come out which are 32 combinations including non-finger's hand, we have used Core i3 Intel @ 2.13GHz on HP Pavilion 6, Windows 7 with java J2SE language as implementation language.

## 9. ACKNOWLEDGMENTS

Our thanks to the experts who have contributed towards development of the template.

## 10. REFERENCES

[1] Hasan, M. M., and Mishra, P. K. 2011. Comparative Study for Construction Of Gesture Recognition System, International Journal of Computer Science and Software Technology, vol. 4(1), pp.15-21.

[2] Athitsos, V., and Sclaroff, S. 2003. Estimating 3D Hand Pose from a Cluttered Image", IEEE Conference on Computer Vision and Pattern Recognition, vol. 2,pp.432-9.

[3] Li, V. 2005. Vision Based Gesture Recognition System with High Accuracy, Department of Computer Science, The University of Tennessee, Knoxville.

[4] Just, A., Rodriguez, Y.,and Marcel, S.2006. Hand Posture Classification and Recognition using the Modified Census Transform, In Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition (AFGR 2006), Southampton , UK, April.

[5] Chang, C. C., Chen, J. J., Tai, W. K., and Han, C. C. 2006. New Approach for Static Gesture Recognition, Journal of Information Science and Engineering, vol. 22, pp. 1047-1057.

[6] Mackie, J., and McCane, B. Finger Detection with Decision Trees, University of Otago, Department of Computer Science.

[7] M. Hasan, M. M. Mishra, P. K. Direction Analysis Algorithm using Statistical Approach, SPIE International Conference for Digital Image Processing (ICDIP 2012), Malaysia, in press.

[8] Hasan, M. M., and Mishra, P. K. 2011. Brightness Factor Matching For Gesture Recognition System Using Scaled Normalization, International Journal of Computer Science & Information Technology (IJCSIT), vol 3(2), pp.35-46.

[9] Lee, D., and Lee, S. G.2011. Vision-Based Finger Action Recognition by Angle Detection and Contour Analysis, ETRI Journal, Vol.(33),pp.415-422.

[10] Oka, K., Sato, Y., and Koike, H. 2001. Real-time Tracking of Multiple Fingertips and Gesture Recognition for Augmented Desk Interface Systems, IEEE International Conference on Automatic Face and Gesture Recognition, pp.429-434.

[11] Malik, S. 2003. Real-time Hand Tracking and Finger Tracking for Interaction, Project Report, 2003.

[12] Oka, K. and Sato, Y. 2002. Real-Time Fingertip Tracking and Gesture Recognition, IEEE Computer Graphics and Applications, vol. 22(6).

[13] Stergiopoulou, E., Papamarkos, N. 2009. Hand gesture recognition using a neural network shape fitting technique, ELSEVIER Engineering Applications of Artificial Intelligence, vol. 22,pp.1141–1158.

[14] Gonzalez, R. C., and Woods, R. E. 2009. Digital Image Processing",Pearson Prentice Hall, First Impression.

[15] Oxford Dictionary, 2011.

[16] Hasan, M. M., and Mishra, P. K. 2012. Improving Morphology Operation for 2D Hole Filling Algorithm, International Journal of Image Processing, vol. 6(1), pp.1-12.

[17] Li, X. 2003. Gesture Recognition Based On Fuzzy C-Means Clustering Algorithm, Department Of Computer Science The University Of Tennessee Knoxville.

[18] Swain, M., and Ballard, D. 1991. Indexing via Color Histograms", Intern. Journal of Computer Vision, vol. 7, pp. 11-332.

[19] Wachs, J., Kartoun, U., Stern, H., and Edan, Y. 1999. Real-Time Hand Gesture Telerobotic System using Fuzzy C-Means Clustering, Department of Industrial Engineering and Management, Ben-Gurion University of the Negov.

[20] Available at, http://www.gamedev.net/ community/forums/topic.asp?topic_id=18309.

[21] Yang, T., and Xu, T. 1994. Hidden Markov Model For Gesture Recognition, The Robotics Institute Carnegie Mellon University Pittsburgh, Pennsylvania 15213.

[22] Phu, J. J., and Tay, Y. H. 2006. Computer Vision Based Hand Gesture Recognition Using Artificial Neural Network, Faculty Of Information And Communication Technology, University Tunku Abdul Rahman (Utar), Malaysia.

[23] Amor, H. B., Ikemoto, S., Minato, T., and Ishiguro, H. 2003. Learning Android Control Using Growing Neural Networks, Department Of Adaptive Machine Systems Osaka University,Osaka, Japan.

[24] Triesch, J., and Malsburg, C. 1996. Robust Classification Of Hand Postures Against Complex Backgrounds, IEEE Computer Society, Second International Conference On Automatic Face And Gesture Recognition.

[25] Marcel, S., Bernier, O., Viallet, J., and Collobert, D. 1999. Hand Gesture Recognition Using Input–Output Hidden Markov Models, France Telecom Cnet 2 Avenue Pierre Marzin 22307 Lannion, France.

[26] The AF Research Laboratory, 2009. Neural Networks,language and cognition, Elsevier, Neural Networks 22,pp. 247_257.

[27] Gunes, H., Piccardi, M., andJan, T. 2007. Face and Body Gesture Recognition for a Vision-Based Multimodal Analyzer, Computer Vision Research Group, University of Technology, Sydney (UTS).

[28] Lu, Y., Lu, S., Fotouhi, F., Deng, Y., and Brown, S. J. 2000. A Fast Genetic K-Means Clustering Algorithm, Wayne State University, Kansas State University Manhattan, USA.

[29] Ibraheem, N. A., Khan, R.Z. 2012. Vision Based Gesture Recognition using Neural Networks Approaches: A Review. International Journal of human Computer Interaction (IJHCI) ), vol.3(1), pp.1-14.

[30] Heisele, B., Ho, P., and Poggio, T. 2001. Face Recognition with Support Vector Machines: Global versus Component-based Approach, Massachusetts Institute of Technology Center for Biological and Computational Learning Cambridge.

[31] Keerthi, S. S., Chapelle, O.,and DeCoste, D. 2006. Building Support Vector Machines with Reduced Classifier, Complexity, Journal of Machine Learning Research 8 (1-22), 2006.