*Research Article*

# A New Fingertip Detection Method Using the Top-Hat Transform

**Pawin Prasertsakul* and Toshiaki Kondo**
School of Information, Computer and Communication Technology,
Sirindorn International Institute of Technology, Thammasat University
Rangsit Campus, Khlong Nueng, Khlong Luang, Pathum Thani 12120, Thailand

## Abstract

This paper presents a new fingertip detection method that is based on the top-hat transform. A hand in an image is assumed to be segmented in advance. The palm of the hand is obtained using the morphological opening operation. Fingertips are then obtained as the opening residue, namely, the difference between the input image and the palm image. The performance of this approach is compared with three other techniques: (1) the convex hull algorithm, (2) the Kanade-Lucas-Tomasi (KLT) feature tracker, and (3) the SUSAN corner detector. Simulation results show that the method (1) is ineffective for closed hands. The methods (2) and (3) tend to respond falsely to many non-fingertip points. By contrast, the proposed method can detect fingertips with more than 90% success rates even for closed hands where the fingers are in contact with each other.

**Keywords**: Fingertip detection; Top-hat transform; Convex hull; KLT feature tracker; SUSAN corner detector.

## 1. Introduction

Fingertip detection is an important part in human-computer interaction (HCI). It may work like a computer mouse, but remotely without contacting. There are demands on the technique for robustly detecting fingertips using a regular camera at low cost. Many researchers have been working on it.

Handy AR, proposed by T. Lee and T. Hollerer, uses fingertips for camera calibration instead of a calibration cardboard [1]. This work uses the Curvature-based algorithm to detect fingertips from the hand contour. This algorithm requires iterative calculations, typically several times, in order to detect fingertips for various sizes of hand since it detects non-fingertips initially.

A virtual keyboard application proposed by E. Posner et al. has a process of fingertip detection [2]. They find a fingertip by analyzing the geometric relationship between three consecutive pixels along the contour of a finger. This approach may detect a false spot as a fingertip if there is a curvature contour looking similar to that of a finger due to the inaccuracy of hand segmentation.

C. Qian et al. recently presented a method for hand tracking [3]. They use a depth sensor to track the hand. In this work, they detect fingers on 2D *x*-*y* plane and 1D z direction separately. To find fingertips, they iteratively compute the geodesic distances of *x*-*y* plane. However, this method is complex and requires specific equipment to process while the authors' objective is to devise a simple and low-cost method.

H. Kato and T. Kato developed a smartphone application which includes fingertips detection and augmented reality in order to control its application launchers [4].

***Correspondence** : cattypkung@gmail.com

They use the convex hull algorithm to make fast fingertip detection [5]. By finding the difference between the hand region and its convex hull region, the convex deficiency is defined for making points where a transition is made in or out. However, this approach can detect fingertips of opened hands only.

The corner detectors, such as the Kanade-Lucas-Tomasi (KLT) feature tracker and the SUSAN locate fingertips indirectly by detecting corners in an image [6], [7]. The KLT finds the corners by finding 2 non-zero eigenvalues while the SUSAN finds the corners by counting white pixels inside the circular mask that applies to the input image to give the corner score. However, these two approaches tend to detect non-fingertips as fingertips. Also, they often fail to detect fingertips while the hand is in closed gesture.

An existing fingertip detection method is proposed in A. Chaudhary et al. [8]. An input hand image is extracted to a binary image. Then the intensity values of the binary image are modified by assigned value from 1 to 255 following the direction of the hand from the wrist to the fingers. However, this method can detect fingertips only when the fingers are separated.

In this paper, we propose a new method for detecting fingertips. We have evaluated the proposed method by comparing it with three traditional approaches that are the convex hull algorithm, the KLT feature tracker, and the SUSAN corner detection. Experimental results show that the proposed method can detect fingertips successfully in various gestures, including closed hands that the traditional approaches cannot handle. The computational time of the proposed method is comparative to those of the three traditional methods.

The rest of this paper is organized as follows. Section 2 describes steps of proposed method. All experimental results and their descriptions are shown in Section 3. Finally, we conclude everything in Section 4.

## 2. Materials and methods

Our approach is divided into 4 steps. It begins with producing a binary image from the input hand. From the binary image, the top-hat transform is used to separate fingers from the palm. Then we find fingertip positions from both fingers and palm. The outline of the proposed method is shown in Fig.1.

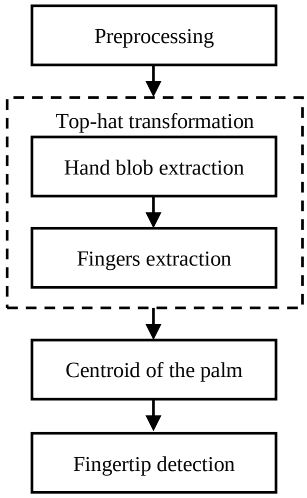



**Fig.1.** Proposed method flow chart.

### 2.1 Preprocessing

First, we segment the hand from an input image using color information [9], [10]. The segmentation result is represented as a binary image where 1 is assigned to the hand region and 0 to the background.

### 2.2 Top-hat transformation

The proposed method is based on the top-hat transform [11]. The process begins with extracting the hand blob by using morphological opening with a circular structuring element (SE). After that, the hand blob is subtracted from the hand region to retrieve fingers, which can be expressed as in Eq. (1),

$$F = H - (H \circ SE) \quad (1)$$

where $F$ is a result of the top-hat transform, $H$ is the hand region, $SE$ is a circular structuring element of diameter 69 pixels, and $H \circ SE$ indicates morphological opening, that is, erosion followed by dilation. This operation removes fingers and extracts the palm region only. From the results of the top-hat transform, 5 largest regions are selected as fingers.

### 2.3 Centroid of the palm

To find fingertip positions, an additional feature is required. It is called centroid of the palm. This feature can be found from the hand blob region following this paper [12].

### 2.4 Fingertip detection

We use the centroid of the palm in the previous step as a reference point for finding fingertip positions. The method begins with the extraction of the contour of each finger. Then we measure the distance between the reference point and the contour. The furthest pixel is considered as the fingertip position.

## 3. Results and discussion

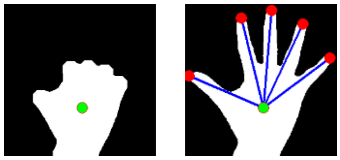
The experiments were performed by using MATLAB on a laptop with 2.3GHz CPU and 4GB RAM. To get the input image, we collected different 4 hand gestures with the same orientation. There are opened, semi-opened, semi-closed, and closed hand gestures. The image resolution is 640 by 640 pixels. In this paper, we use 25 different hands.

### 3.1 Fingertip detection using top-hat

Fig. 4 demonstrates how the top-hat transform is used in the proposed method. Fig. 2(a) shows an input hand image. Fig. 2(b) exhibits the result of opening, i.e., erosion followed by dilation. Fig. 2(c) shows the opening residue, that is, the difference between Fig. 2(a) and Fig. 2(b). We then find the center of gravity of the hand blob in Fig. 3(a). We use this point as a reference for locating five fingertips. The fingertip is defined as the furthest point in each finger segment from the reference point (Fig. 3(b)).

(a) (b) (c)

**Fig.2.** Raw results of the top-hat transform (a) hand (b) hand blob and (c) fingers.



(a) (b)

**Fig.3.** Final results of the top-hat transform (a) center of gravity and (b) fingertips.

### 3.2 Discussion in results

Figs. 4 to 7 are fingertip detection results in different 4 hand gestures. Sub-Figs (a) represent the convex hull results, Sub-Figs (b) represent the KLT feature tracker results, Sub-Figs (c) represent the SUSAN corner detector results, and Sub-Figs (d) represent the proposed method results.

The convex hull is sensitive to the hand gesture. When the fingers are separated widely, only the fingertip positions are in the result (Fig. 4(a)). Otherwise, when the fingers are closer, the convex part differs from the opened hand condition. For this reason, the wrong detected fingertips occur (Figs. 5(a), 6(a), and 7(a)).

Non-fingertip positions in the KLT results are detected because the KLT confuses between convex and concave corners. For opened, semi-opened, and semi-closed gestures, the KLT is successful in detecting fingertips but sometimes detects non-fingertip positions (Figs. 4(b), 5(b), and 6(b)). Only in the closed gesture does the KLT hardly detect fingertips (Fig. 7(b)). The fingertips on index and ring fingers are often lost because this method considers that these two fingers are the middle finger.

In case of the SUSAN corner detection, we compare the detection result with the KLT. The SUSAN detects less non-fingertip positions since it does not confuse convex and concave corners (Figs. 4(c), 5(c), and 6(c)). Sometimes it detects non-fingertip positions because the corner score at that position is slightly higher than the threshold value. The SUSAN almost successfully detects fingertips for the closed gesture (Fig. 7(c)). It misses a fingertip because the SUSAN may consider the two fingers to be the same finger.

The proposed method can detect all the fingertips for every hand gesture (Figs. 4(d), 5(d), 6(d), and 7(d)). This method rarely detects non-fingertip positions since the distance between the reference point and the pixel at the fingertip is higher than the distance between the reference point and the pixel at a non-fingertip.

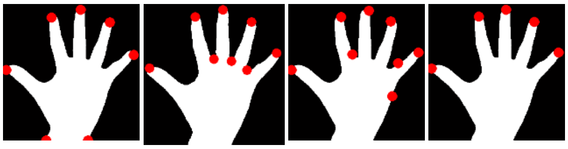

(a) (b) (c) (d)

**Fig.4.** Opened gesture (a) Convex hull (b) KLT (c) SUSAN and (d) Proposed method.

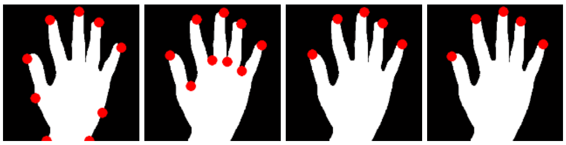

(a) (b) (c) (d)

**Fig.5.** Semi-opened gesture (a) Convex hull (b) KLT (c) SUSAN and (d) Proposed method.


(a) (b) (c) (d)

**Fig.6.** Semi-closed gesture (a) Convex hull (b) KLT (c) SUSAN and (d) Proposed method.

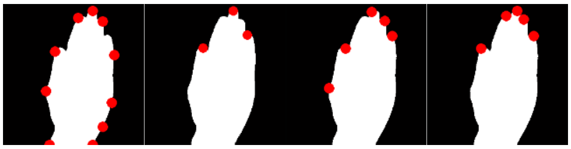

(a) (b) (c) (d)

**Fig.7.** Closed gesture (a) Convex hull (b) KLT (c) SUSAN and (d) Proposed method.

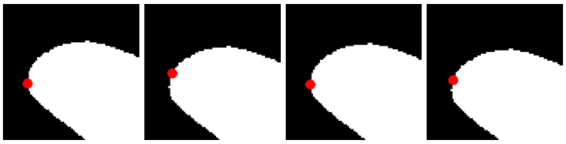
### 3.3 Evaluation

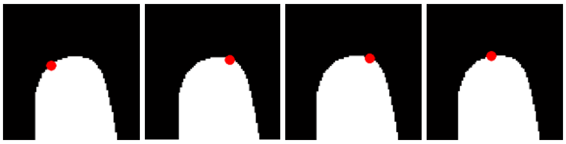
To evaluate the performance we compare the accuracy of position that each method detected by defining the target position at the middle of the fingertip. Figs. 8 to 12 show fingertips which are cropped from the final results of the opened gesture. We evaluate the performance from this gesture because every method can detect the fingertips easily.

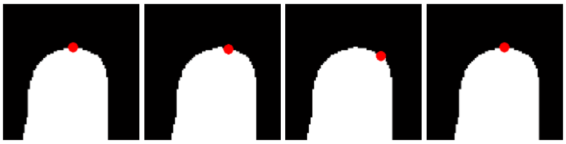
In the convex hull, three fingertips are detected approximately at the target position (Figs. 8(a), 10(a) and 11(a)). The other two fingertips are detected far from the target position (Figs. 9(a) and 12(a)).

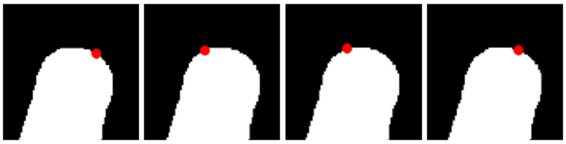
In the KLT corner detection, three fingertips are detected approximately at the target position (Figs. 8(b), 10(b), and 12(b)) while the other two fingertips are detected slightly off target position (Figs. 9(b) and 11(b)). The SUSAN can detect two fingertips approximately at the target point (Figs. 8(c) and 12(c)). These two methods tend to detect at the position where the curvature is maximal. Therefore, the detected points are often off the target position (Figs. 9(b), 11(b), 9(c), 10(c), and 11(c)).

In the proposed method, all fingertips are detected approximately at the target point since the detected point and the target point are the furthest points from the center of gravity (Figs. 8(d), 9(d), 10(d), 11(d), and 12(d)). Consequently, all fingertips detected by the proposed method are good approximations of the target point.


(a) (b) (c) (d)
**Fig.8.** Thumb (a) Convex hull (b) KLT (c) SUSAN and (d) Proposed method.


(a) (b) (c) (d)
**Fig.9.** Index finger (a) Convex hull (b) KLT (c) SUSAN and (d) Proposed method.


(a) (b) (c) (d)
**Fig.10.** Middle finger (a) Convex hull (b) KLT (c) SUSAN and (d) Proposed method.


(a) (b) (c) (d)
**Fig.11.** Ring finger (a) Convex hull (b) KLT (c) SUSAN and (d) Proposed method.


(a) (b) (c) (d)
**Fig.12.** Little finger (a) Convex hull (b) KLT (c) SUSAN and (d) Proposed method.

As a further evaluation, we compare the computation times of the four approaches. We apply each approach to the 25 different hand images and calculate the average and the standard deviation.

Table 1 shows a computation time table. The convex hull algorithm is the fastest since the algorithm has a few steps that comprise finding the convex hull and subtracting it from the input to retrieve the convex deficiency. Finally, the points are marked on the hand contour where a transition is made in or out of a component of the convex deficiency. The KLT feature tracker is slightly slower than the convex hull even when we slide the window along the contour of the segmented hand image. The SUSAN corner detector is the slowest since we have to count the number of pixels inside the circular mask for every spot when the circular mask is on the input image. The proposed method is slightly slower than the convex hull and the KLT. It mostly spends time for finding the center of gravity and distance measurement.

**Table1.** Computation time.

| | **AVG (Sec)** | **SD (Sec)** | **MIN (Sec)** | **MAX (Sec)** |
|---|---|---|---|---|
| **Convex** | 0.1712 | 0.0089 | 0.1544 | 0.2006 |
| **KLT** | 0.2296 | 0.0359 | 0.1538 | 0.3579 |
| **SUSAN** | 0.6708 | 0.0866 | 0.4536 | 0.8221 |
| **Proposed** | 0.3763 | 0.0146 | 0.3518 | 0.4087 |

The numerical results are also compared in sensitivity. In each approach, we record a number of the input image which is success to detect fingertips on thumb, index, middle, ring, and little fingers. Since the proposed method has exactly 5 detected points, to make a fair comparison the existing methods are optimized by limiting the number of detected points to be 5, except for the convex hull algorithm.

For the convex hull, false positives at the border of image are not considered as errors because they frequently occur and they can be removed easily. The convex hull succeeds to detect all fingertips for opened and semi-opened gestures. However, the semi-opened gesture starts to have false positives. Some fingertips are lost when the hand is in semi-closed or closed gesture especially the thumb and little fingers. In Table 2, we use additional information which is precision because there is no score calculation in this method. Therefore, we cannot limit the number of detected points to 5, and thus we may have more false positives than for other techniques.

For the KLT feature tracker, we adjust the size of the window to make all results in each hand become optimized since the KLT is very sensitive to the size of the hand. The detected points are limited to 5 by selecting the 5 largest eigenvalues $\lambda_2$. Table 3 shows a poor performance by the KLT. This is because the KLT cannot distinguish between convex and concave corners. It produces detected pointed at both fingertips and non-fingertips e.g. the root of the finger. After we select the 5 largest eigenvalues, the points at the root of the finger are selected since their eigenvalues are larger than the points at the fingertips. Index and ring fingers are not detected for any hands in the closed gesture. As we mentioned in previous section, the KLT considers these two fingers to be the middle finger.

Unlike the KLT, the SUSAN corner detector can distinguish convex corners from concave corners. The number of detected points is limited to 5 by selecting the 5 highest corner scores.

Table 4 shows that the SUSAN is ineffective for fingertip detection. The low sensitivity indicates that the SUSAN often fails to detect fingertips, resulting in a high false negative (FN).

Table5 shows that the proposed method successfully locates 5 fingertips when the hand is opened, semi-opened, and semi-closed. Since we select the 5 largest components in the top-hat transform, we rarely detect a non-fingertip as a fingertip. Thus, the false positive (FP) is very low, which helps to reduce the false negative (FN) and achieve high sensitivity. Some of the fingertips of closed gesture are missed because two adjacent fingers are so close that they are segmented together as one blob. The proposed method is ineffective when there is no concave corner between the adjacent fingers.

**Table 2**. Numerical results of the convex hull algorithm.

| Gesture \ Finger | Thumb | Index | Middle | Ring | Little | Sensitivity | Precision |
|---|---|---|---|---|---|---|---|
| Opened | 25 | 25 | 25 | 25 | 25 | 100% | 100% |
| Semi-opened | 25 | 25 | 25 | 25 | 25 | 100% | 77.1% |
| Semi-closed | 23 | 25 | 25 | 25 | 11 | 87.2% | 68.1% |
| Closed | 24 | 25 | 25 | 24 | 12 | 88% | 60.9% |

**Table 3**. Numerical results of the KLT feature tracker.

| Gesture \ Finger | Thumb | Index | Middle | Ring | Little | Sensitivity |
|---|---|---|---|---|---|---|
| Opened | 15 | 20 | 11 | 12 | 11 | 55.2% |
| Semi-opened | 15 | 18 | 13 | 11 | 20 | 61.6% |
| Semi-closed | 18 | 20 | 11 | 14 | 23 | 68.8% |
| Closed | 18 | 0 | 22 | 1 | 14 | 44% |

**Table 4**. Numerical results of SUSAN corner detector.

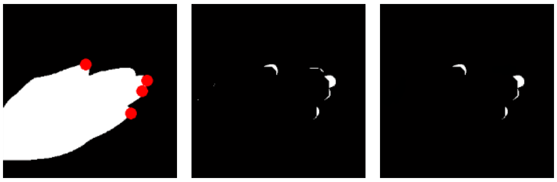
| Finger<br>Gesture | Thumb | Index | Middle | Ring | Little | Sensitivity |
|---|---|---|---|---|---|---|
| Opened | 21 | 16 | 16 | 14 | 18 | 68% |
| Semi-opened | 20 | 13 | 12 | 16 | 16 | 61.6% |
| Semi-closed | 19 | 17 | 14 | 15 | 22 | 69.6% |
| Closed | 20 | 8 | 20 | 5 | 18 | 56.8% |

**Table 5**. Numerical results of the proposed method.

| Finger<br>Gesture | Thumb | Index | Middle | Ring | Little | Sensitivity |
|---|---|---|---|---|---|---|
| Opened | 25 | 25 | 25 | 25 | 25 | 100% |
| Semi-opened | 25 | 25 | 25 | 25 | 25 | 100% |
| Semi-closed | 24 | 23 | 25 | 24 | 25 | 96.8% |
| Closed | 25 | 21 | 25 | 21 | 24 | 92.8% |

### 3.4 Discussion of errors

In this section, we describe a closed-hand case for which the proposed method fails to detect fingertips. Fig. 13(a) illustrates that one of the fingertips is missed. The reason for it is that the fingertip is too small after subtracting the hand blob as shown in Fig. 13(b). If the subtraction result is too small, it may be discarded in the thresholding step that follows (Fig. 13(c)).



(a) (b) (c)

**Fig.13.** An error case in closed hand gesture (a) detection result (b) fingers (c) fingers after threshold.

Fig.14(a) shows a miss detection in the semi-closed hand gesture. For this case, after the hand blob is subtracted from the hand region, there is a finger that makes contact with its adjacent finger. Thus, the method misunderstands and the two fingers are considered to be one finger (Fig.14(b)). It concludes that one of the fingertip positions cannot be found.

(a) (b)

**Fig.14.** An error case in semi-closed hand gesture (a) detection result (b) fingers that thumb and index are contracted.

## 4. Conclusion

Although the convex hull algorithm and the KLT tracker are fast, they commit numerous false detections. Simulation results reveal that the convex hull algorithm is effective only for opened hands where the fingers are spread widely. This is a serious limitation when the hand recognition is concerned. Meanwhile, the KLT feature tracker and the SUSAN corner detector respond not only to fingertips but also to the bases of the fingers. This means we need to introduce an extra process for distinguishing convex corners from concave corners. Simulation results also show that the KLT feature tracker and the SUSAN corner detector respond to the points where the curvature is locally maximal, rather than detecting the real fingertips. The proposed method, on the other hand, can successfully locate the fingertips of the hands in various gestures, including a closed hand. Also, it is faster than the SUSAN corner detector.

## 5. Acknowledgment

This research is financially supported by Thailand Advanced Institute of Science and Technology (TAIST), National Science and Technology Development Agency (NSTDA), Tokyo Institute of Technology, and Sirindhorn International Institute of Technology (SIIT), Thammasat University (TU). A special acknowledgement is also extended to the Center of Excellence in Biomedical Engineering, SIIT, TU.

## 6. References

[1] T. Lee and T. Hollerer, Handy AR: Markerless Inspection of Augmented Reality Objects Using Fingertip Tracking, IEEE Int. Symposium Wearable Computers, pp. 83-90, Oct 2007.

[2] E. Posner, N. Starzicki, and E. Katz, A Single Camera Based Floating Virtual Keyboard with Improved Touch Detection, IEEE Convention of Electrical and Electronics Engineers in Israel, pp. 1-5, Nov 2012.

[3] C. Qian, X. Sun, Y. Wei, X. Tang, and J. Sun, Realtime and Robust Hand Tracking from Depth, CVPR2014 Computer Vision and Pattern Recognition, Jun 2014.

[4] H. Kato and T. Kato, A Marker-less Augmented Reality Based on Fast fingertip Detection for Smart Phones", IEEE Int. Conf. Consumer Electronics, pp. 127-128, Jan 2011.

[5] R. C. Gonzalez and R. E. Woods, Digital Image Processing 3rd Edition, Prentice Hall, chapter 9, pp. 669-671, 2008.

[6] J. Shi and C. Tomasi, Good Features to Track, IEEE Conf. Computer Vision and Pattern Recognition, pp. 593-600, Jun 1994.

[7] S. M. Smith and J. M. Brady, SUSAN – a New Approach to Low Level Image Processing, Int. Journal of Computer Vision, Vol. 23, Issue 1,pp. 45-78, May 1997.

[8] A. Chaudhary, J. L. Raheja, and S. Raheja, A Vision Based Geometrical Method to Find Fingers Positions in Real Time Hand Gesture Recognition, Journal of Software, Vol. 7, No. 4, pp. 861-869, April 2012.

[9] P. Prasertsakul, T. Kondo, T. Phatrapornnant, and T. Isshiki, A Robust Hand Segmentation Method Based on Color and Background Subtraction", ICICTES Int. Conf. Embedded System Applications, Jan 2014.

[10] J. J. de Dios and N. Garcia, Face Detection Based on a New Color Space YCgCr, ICIP Int. Conf. Image Processing, Sep 2003.

[11] R. C. Gonzalez and R. E. Woods, Digital Image Processing, 3rd edition", Prentice Hall, Chapter 9, pp. 690-694, 2008.

[12] Y. Liao, Y. Zhou, and H. Zhou, Fingertips Detection Algorithm Based on Skin Colour Filtering and Distance Transformation", QSIC Int. Conf. Quality Software, pp. 276-281, Aug 2012.

## 7. Appendix

### 7.1 Convex hull algorithm

We assume that a hand is already segmented in the preprocessing step. First, the convex hull theory is utilized to find a convex set from a hand region. To obtain the convex hull of the hand region, 4 structuring elements are used by a morphological algorithm Eq. (2).

$$X_k^i = \left(X_{k-1}^i \circledast SE^i\right) \cup A \qquad (2)$$

starting with $X_0^i = A$ where $X$ is the result of each structuring element, $A$ is the hand region and $X_{k-1}^i \circledast SE^i$ means the morphological algorithm is applied to the binary image by using structuring element $SE$ following Fig. 1.

| | | |
|---|---|---|
| 1 | 0 | 0 |
| 1 | 0 | 0 |
| 1 | 0 | 0 |

| | | |
|---|---|---|
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

| | | |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 0 | 1 |
| 0 | 0 | 1 |

| | | |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 1 | 1 |

**Fig.1.** Structuring elements.

After each structuring element is used iteratively until its output is converged (i.e. $X_k^i = X_{k-1}^i$), the algorithm lets $D^i = X_k^i$ then calculates the convex hull by merging all converged outputs of each structuring element together which is described as

$$C(A) = \bigcup_{i=1}^{4} D^i \quad (3)$$

where $C(A)$ is the convex hull of the hand region $A$.The convex deficiency is defined by finding the difference between the convex hull of the hand region and the hand region. These two concepts are used to partition the hand contour by following the hand contour and marking the points where a transition is made in or out of a component of the convex deficiency. These marking points represent fingertip positions.

### 7.2 KLT feature tracker

The Kanade–Lucas–Tomasi (KLT) feature tracker can be also used for detecting corners in a binary image. For this, we first construct the gradient covariance matrix C as shown in Eqs. (4) and (5). The matrix is a null matrix where there is no boundary of a hand present in the image (Eq. (5)). $I(x, y)$ represents a binary input image at pixel coordinates $(x, y)$ . We use the Sobel operators for obtaining the partial derivatives $I_x$ and $I_y$ of the image $I$. The local summation $\Sigma$ is conducted within a block of size 35 by 35 pixels. By solving the characteristic equation Eq. (6), we finally obtain two eigenvalues $\lambda_1$ and $\lambda_2$ of the gradient covariance matrix where $\lambda_1 \geq \lambda_2$. When the smaller eigenvalue $\lambda_2$ is greater than a certain threshold level, we consider it as a corner corresponding to the tip of a finger.

$$\boldsymbol{A} = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \quad (4)$$

$$\boldsymbol{C} = \begin{cases} \boldsymbol{A}, & I_x(cw_x, cw_y) \text{ or } I_y(cw_x, cw_y) \neq 0 \\ 0, & otherwise \end{cases} \quad (5)$$

$$det(\lambda \boldsymbol{I} - \boldsymbol{C}) = 0 \quad (6)$$

where $I_x(cw_x, cw_y)$ and $I_y(cw_x, cw_y)$ are pixels at the center of the windows on the partial derivatives $I_x$ and $I_y$, $\lambda$ represent two eigenvalues, and $I$ is the 2x2 identity matrix.

### 7.3 SUSAN corner detector

The SUSAN corner detector was originally developed for finding corners in a grayscale image. We apply the SUSAN detector to binary images. If the input image is changed into the binary image, there are only two regions of information which are white region (i.e. target) and black region (i.e. non-target). Since hand region is represented in the white region, the SUSAN is applied on the white region only. For detecting corners, we use a circular mask whose diameter is 15 pixels. We then count the number of white pixels inside the circular mask. If the counted number is less than half of the number of pixels within the mask *n*, this counted number is used to calculate the corner score, *CS*, as shown in Eq. (7). If the number of white pixels is larger than or equal to half of *n*, the *CS* is given zeros. Finally, we determine whether the pixels are fingertips or not by comparing with a threshold value.

$$CS = \begin{cases} (n/2) - A, & A < n/2 \\ 0, & otherwise \end{cases} \quad (7)$$

where $CS$ is the corner score, $A$ the number of white pixels inside the circular mask, and $n$ the number of pixels within the circular mask.