International Journal of Computer Science and Software Engineering (IJCSSE), Volume 4, Issue 12, December 2015
ISSN (Online): 2409-4285 www.IJCSSE.org Page: 315-322

Using Kinect System and OpenCV Library for Digits Recognition

Bassem Khelil[1] and Hamid Amiri[2]

[1, 2] Electrical Engineering Department, National Engineering School of Tunis, SITI-LAB, Tunis, Tunisia

[1]*khelil.bassem@gmail.com*, [2]*hamidlamiri@gmail.com*

ABSTRACT

The recently developed depth sensors, e.g., the Kinect sensor, have provided new opportunities for human-computer interaction (HCI). Although great progress has been made by leveraging the Kinect sensor, e.g., in human body tracking, face recognition and human action recognition, robust hand gesture recognition remains an open problem. Compared to the entire human body, the hand is a smaller object with more complex articulations and more easily affected by segmentation errors. It is thus a very challenging problem to recognize hand gestures. This paper proposes hand static gesture recognition digits 0-5 successfully captured through Kinect, which acts as a sign language for conveying information, by using OpenCV library.

Keywords: *Hand Gesture Recognition, OpenCV, Depth image, Microsoft Kinect.*

1. INTRODUCTION

The Microsoft Kinect is a low cost device that combines an RGB camera with a depth sensor. Depth information is inferred by comparing a scene illuminated by a structured IR speckle dot pattern against a calibrated reference. Specifically, a depth image is calculated internally by comparing the spacing of the returned dots against known values at specific depths [1]. The end result is an inexpensive RGB-D (red, green, blue, and depth) sensor, a device that has proven useful in various Human Computer Interface problems from finger counting to gesture recognition.

One particularly interesting application of such sensors is the classification of gestures for automatic sign language recognition. Implementations have been created for various sign language alphabets and words including Greek, English, and Japanese [2]. While such sign language dictionaries are well beyond the scope of this project, our goal is to implement a Kinect based gesture recognition system that is capable of resolving numerical digits from 0 through 5 as well as more general single hand gestures.

2. Microsoft Kinect

We present our experience of using a Kinect depth camera for recognition of some common gestures. Kinect interprets a 3D scene information using a projected infrared structured light [4] (fig.1). This 3D scanner system called Light Coding employs a variant of image-based 3D reconstruction. The Kinect sensor is a horizontal bar connected to a small base with a motorized pivot and is designed to be positioned lengthwise above or below the video display. It also has a RGB camera.

The depth sensor consists of an infrared laser projector combined with a monochrome CMOS sensor, which captures video data in 3D under any ambient light conditions. The depth images taken by Kinect for a particular scene is shown in fig. 2. The depth map is visualized here using colour gradients from white (near) to blue (far). The monochrome depth sensing video stream is in VGA resolution with 2,048 levels of sensitivity.

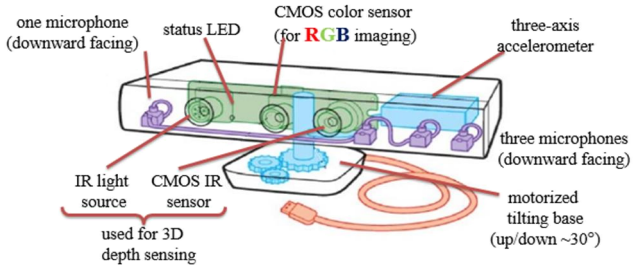

Fig. 1. Microsoft Kinect.

3. Literature Review

Numerous papers have been written on the subject of generalized hand gesture recognition using RGB-D sensors. One technique considered early on was proposed by Du [3].

Fig. 2. Depth Image.

The idea is to trace the contours of a hand silhouette and detect concavities and convexities. The convexity points correspond to the finger tips, while the concavity points define the region between fingers. A simple classifier is defined for the digits 0 to 5 by counting the number of these points. However, this descriptor and classification scheme are already applied to Kinect sensor information for gesture recognition. So, instead, the Malima [4] paper is considered. As with the Du method, the scheme proposed in Malima has simple, yet robust feature selection and classification. In addition, it is scale, translation, and rotation invariant. However, their method is applied to images captured with an RGB camera, where segmentation is performed by thresholding the color channels based on skin color values. In principle, though, the circle-based descriptor is applicable to depth images as well. This is one of the descriptors implemented as part of the project. It is described in greater detail in the following section.
Unfortunately, although simple and relatively robust, the Du and Malima systems are limited to the counting digits, 0 to 5. They are both analyzing local contour features of the hand outline, but do not recognize the shape of the hand as a whole. So, more generalized gesture methods were explored.

One such paper is the covariance matrix approach in Guo [7]. This approach performs human action recognition by looking at a sequence of whole body silhouettes over time, a silhouette tunnel, captured by the Kinect. A 13 dimensional feature vector is defined, where 3 values are row, column, and time, 8 values are based on the shape of the silhouette (described later), and the last 2 are a measure of the temporal similarity. However, this feature vector is general enough to work with hand silhouettes in addition to full bodies. So, it is possible to modify the shape feature vector to work with static hand gestures by reducing the dimensionality. This can be accomplished by removing the time dependence and temporal similarity terms. The result is a generalized 10 dimensional feature vector applicable to static shape recognition. The specific implementation and classification details are defined in the next section.
Fourier descriptors were also researched. A nice survey of Fourier based shape representations is provided in [6]. Out of all the methods surveyed, the centroid (central) distance Fourier descriptor provides the best results. This method is then used by Kulshreshth [5] to perform gesture recognition with the Kinect. Surprisingly, however, they limit themselves to just recognizing digits as in finger counting. It is implied that the comparatively low resolution of the depth images from the Kinect is the limiting factor. In addition, there is no testing done on the optimal number of contour points to sample. So, we chose to explore the central distance Fourier descriptor further.

4. Proposed Approach

The development environment consisted of the Microsoft Kinect for image capture, the OpenCV library for image processing, OpenNI for interfacing with the Kinect, and a third party NITE library for hand tracking. The figure 3 below illustrates the flowchart.

4.1 Preprocessing

The preprocessing steps include: image capture, hand localization, silhouette generation, and finding the center of the hand. There are three available stream sources for image capture from the Kinect: RGB, the raw infrared (IR) dot pattern, and the processed depth image.

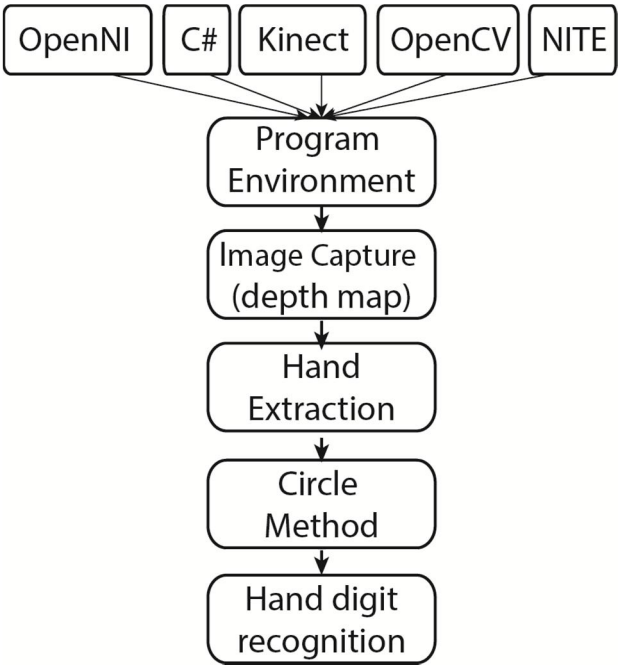

Fig. 3. Overall workflow.

4.1.1 IR Stream

Our initial method used the raw IR stream for image capture. Hand localization is performed by cropping to a rectangle around the hand coordinates provided by the NITE library. A 3x3 median filter is applied to the resulting image, followed by global thresholding with Otsu's Method [8] using the openCV function, cv2.threshold().

Otsu's Method is a binary thresholding scheme that separates the intensity values into two classes (foreground and background in this case). The problem is that the intensity ranges of the two classes may overlap in the histogram of the image. Intuitively, the basic idea is to find the value which best splits the histogram into two distinct, well-separated histogram regions. Practically, this is done by finding the threshold value on the histogram, K ∈ {0…255}, which maximizes the between-class variance, σ2 (k) [9]:

where $\sigma^2(k) = P_B(k)P_F(k)(m_B(k) - m_F(k))^2$

$P_B(k) = \sum_{i=0}^{k} p_i$ =*Prob pixel assigned to background*

$P_F(k) = \sum_{i=k+1}^{255} p_i = 1 - P_{B=}$*Prob pixel assigned to foreground*

$m_B(k) = \frac{1}{P_B}\sum_{i=0}^{k} ip_i$ =*mean of the background pixels*

$m_F(k) = \frac{1}{P_F}\sum_{i=k+1}^{255} ip_i$ =*mean of the foreground pixels*

However, although processing the IR stream produces useful hand silhouette images, there is no depth information included. Segmentation is based on the intensity values of the IR image itself, which is quite dim. So, this segmentation method only provides good results in the case where the user's hand is separated far enough from the body and is shown against a white background. Instead, we switched to using the resolved depth stream.

4.1.2 Depth Stream

a) Image Capture

- Capture a raw depth image from the Kinect and quantize to 0 through 255
- Background Elimination [3]
  - ✓ Invert the depth values
  - ✓ Assign the background pixels to 0
  - ✓ (0 = background, 255 = foreground)

b) Hand Localization

Finer Thresholding

- Use hand location provided by NITE tracker **(Px, Py)**
- Find the average depth value in 3x3 window centered at (Px, Py)
- Threshold to a certain range around this average depth value (empirical)
- Crop to box around hand using depth at (Px, Py) [11]

$$scale = \frac{13000}{\text{Depth(Px, Py)}}$$

  - ✓ Constant 13000 found empirically
  - ✓ Proportional to the area of bounding box

$$Top\ Left\ Pixel = (Px - 10 * scale * 0.5, Py - 10 * scale * 0.6)$$

  - ✓ This formula scales an initial 10x10 box centered at (Px, Py) to the appropriate dimensions based on the depth
  - ✓ Constants 0.5 and 0.6 take into account that most hand silhouettes are more tall than wide

$$\text{Width} = \text{Height} = 10 * \text{Scale}$$

c) Generate Silhouette

- Trace the contours using OpenCV's findContour function.

- Only keep the contour with the largest area
- Fill in the contour to generate a hand silhouette

d) Locate Center of Hand Silhouette (COG)

$$x_{ctr} = \frac{\sum_{i=0}^{k} x_i}{k}, y_{ctr} = \frac{\sum_{i=0}^{k} y_i}{k}$$

Where (xi,yi) = silhouette coordinates, (xctr, yctr) = center of hand, and k = total number of pixels in the hand.

4.2 Circle Method

This detection method is based on Malima et al [4]. The idea is to draw a circle centered at (xctr, yctr), which cuts through all the fingers of the hand. The radius of this circle needs to be large enough that it encompasses more than just the palm, but small enough that no fingers are missed. The intersection of the circle with the silhouette forms the basis of the finger detection scheme. The circumference of the circle is traced for discrete values of theta from 0 to 360. For each theta, the algorithm assigns a value of 0 or 1 depending on whether or not the underlying pixel is in the silhouette of the hand. The result, shown in figure 4, is a 1D vector (descriptor) for each gesture. (Note: The vectors on the right have been stretched out in two dimensions for visualization purposes only.)

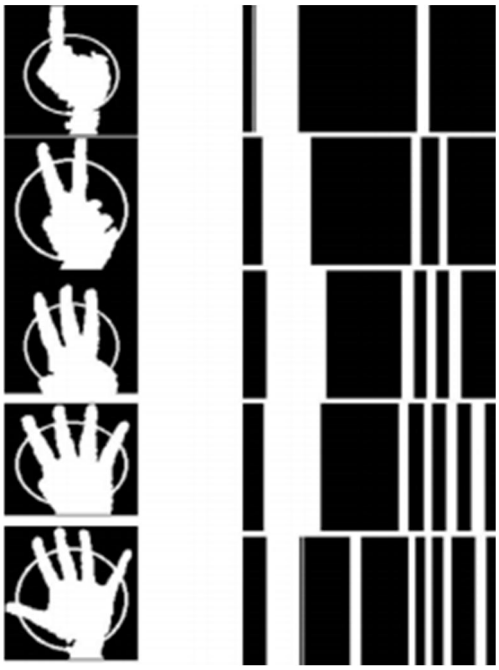

Fig 4. Circle Method Descriptor.

1) Find the distance between the center of the hand at (xctr, yctr) and the furthest point on the silhouette
2) Define five circles with radii scaled at 0.65, 0.67, 0.7, 0.73, 0.75 of this maximum distance
3) Generate feature vector defined previously for each circle
4) Count the number of discontinuities and map to the appropriate finger count for each circle
5) Apply majority rule to classify the gesture.

Because the decisions are based on a scaled value of the distance computed in Step 2, this classification scheme is scale invariant. Rotation invariance is the result of using a circle for feature selection. This method is robust given a perfectly segmented hand. However, the classifier is limited to counting the numbers between 0 and 5 and not adaptable to more generalized gestures. Therefore, we tested other method such as covariance matrix and Fourier descriptor.

4.3 Covariance Matrix Approach

As previously stated in the Literature Review, the Covariance Matrix approach proposed in Guo [7] can be easily adapted for use with static hand gestures.

4.3.1 Create the Feature Vector

- Compute the 10-dimensional feature vector adapted from Guo [7]:
- **f**(x,y)= [x, y, de, dw, dn, ds, dne, dsw, dse, dnw]
  - ✓ x = col, y = row
  - ✓ d = Euclidean distance from (x,y) to the nearest boundary point in the specified direction
  - ✓ east, west, north, south, northeast, southwest, southeast, northwest
- Scale Invariance
  - ✓ Divide each spatial feature by the square root of the silhouette area
- Compute the Covariance Matrix:
  - ✓ $cov(f(s)) = \frac{1}{|s|}\sum_{(x,y)\in S}(f(x,y) - \mu_F)(f(x,y) - \mu_F)^T$
  - ✓ *S*=area silhouette

- ✓ $\mu_F = \sum_{(x,y)\in S} \frac{1}{|s|} f(x,y)$ = mean feature vector for silhouette

4.3.2 Build a Dictionary

For a set of images of the same gesture, compute the Covariance Matrix and add it to the dictionary with the same label. Repeat for all desired gestures.

4.3.3 Classification

As noted in Guo [6], the set of all covariance matrices lie on a Riemannian manifold. So, a Euclidean distance measure cannot be used. Instead, the distance between two covariance matrices on this manifold is defined as:

$$(c, c') = \sqrt{\sum_{k=1}^{10} (\ln \lambda_k (c, c'))^2}$$

Where $\lambda_k(c, c')$ are the generalized eigenvalues of c and c'
C=covariance Matrix of new sample, c'=Reference from Dictionary.

4.4 Fourier Shape Descriptor

The Fourier Shape descriptor used in this paper is based on the work in Kulshreshth [5]. The basic idea is to first create a set of Euclidean distances between the center of the hand and equidistant points along the contour [see Fig 5]. The magnitude of the Fourier Transform of this set forms a unique shape signature, which can be used for generalized gesture classification. In addition, this descriptor is rotationally invariant. Shifts in the silhouette contour points, which is the cause of rotation, will be appear as phase delays in frequency domain. However, since only the magnitude of the Fourier coefficients is considered, the phase (or equivalently, the rotation) is ignored. So, this method is rotationally invariant while remaining computationally fast.

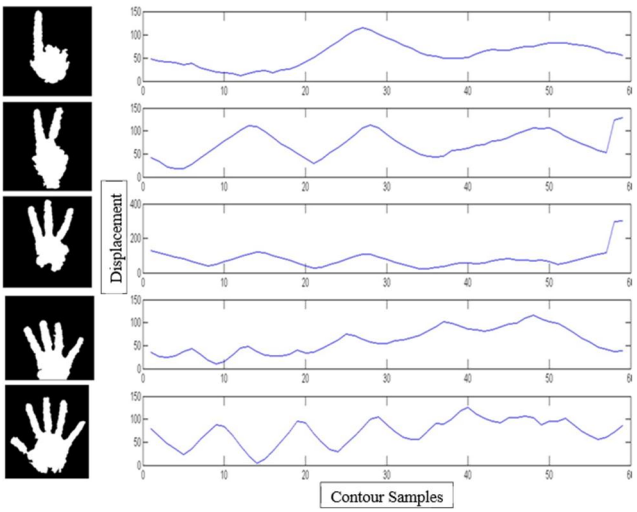

Fig. 5. Centroid Displacement Diagram (Full contour before sampling).

4.4.1 Generating the Fourier Descriptor

- Sample N = 16 equidistant points along the full contour of the hand (Fig 5)
- Calculate Euclidean distance between each sample point and the center of the hand.

$$r[n] \text{ for } n = 1..N$$

- Take the magnitude of the N point DFT of these points

$$Abs(FT\{r[n]\}) = a[m] \text{ for } m = 1..N$$

- Normalize the Fourier coefficients by the DC value (Scale Invariance)
- Keep the first 7 normalized coefficients (skip DC, which is always 1)

This is the Fourier Descriptor for a single shape.

4.4.2 Make a Dictionary

For a set of images of the same gesture, compute the average Fourier shape descriptor and add it to the dictionary with the same label. Repeat for all desired gestures.

4.4.2 Classification

Compute the Fourier descriptor for each new sample. Compare it with each stored gesture in the dictionary using a Euclidean distance measure. The label of the minimum distance is the desired gesture.

5. Experimental results

5.1 Comparison between Covariance Matrix and Fourier Descriptor

5.1.1 Testing Setup

- Covariance Matrix Methods implemented in MATLAB
- 600 images = 6 gestures, labeled 0 to 5, 100 images per gesture
- Vertical Orientation (No Rotation)

5.1.2 Confusion matrices for the Covariance Matrix and Fourier Descriptor Methods

Table 1: Confusion Matrix for Covariance Matrix Method

Actual Data \ Predicted(CCR=89.5%)	0	1	2	3	4	5
0	1	0	0	0	0	0
1	0	0.85	0	0.11	0	0.4
2	0	0	0.85	0.15	0	0
3	0	0	0.13	0.87	0	0
4	0	0.08	0	0	0.84	0.08
5	0	0	0	0.04	0	0.96

Table 2: Confusion Matrix for Fourier Descriptor

Actual Data \ Predicted(CCR=90.6%)	0	1	2	3	4	5
0	1	0	0	0	0	0
1	0	1	0	0	0	0.4
2	0	0.04	0.85	0.11	0	0
3	0	0	0	0.88	0.12	0
4	0	0	0	0.07	0.93	0
5	0	0	0	0	0.22	0.78

5.1.3 Results

As is evident in the tables above, both methods are quite effective. However, during additional testing, we found that rotation in the input images decreased the effectiveness of the covariance matrix method. This result is reasonable since the covariance matrix is not rotationally invariant. One simple solution is to first find the orientation of the hand, re-center the silhouette vertically, and then run the covariance matrix method. However, because the Fourier descriptor is already rotationally invariant and much faster computationally, it was chosen as the primary shape signature and subsequently ported to C++/OpenCV for real time classification with the Kinect.

5.2 Determining the Optimal DFT Length

5.2.1 Testing Setup (Matlab)

- 600 images = 6 gestures, labeled 0 to 5 (Same as previous test)
- N = 16, 32, 64, and 128 point DFT
- N/2 – 1 = 7, 15, 31, 63 Coefficients kept

Table 3: Decisions made with N/2-1 of Coefficients

N-FT length \ Gestures	0	1	2	3	4	5	CCR
16	1	0	0	0	0	0	0.906
32	0	1	0	0	0	0.4	0.85
64	0	0.04	0.85	0.11	0	0	0.69
128	0	0	0	0.88	0.12	0	0.45

5.2.2 Results

The 16 point DFT results in the highest correct classification rate (CCR), where CCR is calculated as the average of the values in each row (i.e. across all gestures). Interestingly, additional testing revealed that the global optimum for our testing data was actually an 18 point DFT. However, we kept N = 16, a power of 2, to maximize the efficiency of the Fourier transform computation. Recall that N is also the number of points along the contour of the hand silhouette. Surprisingly, increasing the number of samples actually decreases the CCR. This is a bit counter intuitive. So, we decided to investigate methods for utilizing a higher order DFT while maintaining lower dimensionality.

5.3 Investigating Fourier Coefficient Pruning

One suggested alteration is to only keep the first few coefficients of a 128 point DFT. The idea is that throwing out Fourier coefficients of such a high order DFT will make the overall contours smoother (remove noise), while retaining the overall shape.

Table 4: Results of 128 Point DFT Coefficient Pruning

Feature Vector length \ Gestures	0	1	2	3	4	5	CCR
7	0.80	0.76	0.42	0.38	0.07	0.04	0.41
15	0.76	0.76	0.42	0.38	0.07	0.04	0.40
31	0.71	0.75	0.41	0.40	0.07	0.04	0.40
63	0.73	0.87	0.46	0.46	0.03	0.14	0.45

From the table above, it appears that the suggested alteration is ineffective. One reason may be that there simply aren't enough contour points (i.e. N > # contour points). For example, when the hand is far from the camera, the resulting silhouette is small, resulting in fewer contour points.

5.4 Additional Gestures (beyond counting fingers)

5.4.1 Testing Setup (C++/OpenCV)

- Fourier Descriptor
- 4000 images, 10 gestures, 400 images per gesture Dictionary
- IR Depth Images (prior to using the Depth Stream)
- Real time testing with the Kinect using IR segmentation scheme

5.4.2 Results

The Fourier shape description method was implemented using the IR depth data. The segmentation was poor when the user held the hand in front of the body. However, for the limited case where the hand was positioned away from the body against a remote background, we were able to create a fairly accurate gesture recognition system which included more generalized shapes. This was due in part to the comparatively higher resolution silhouettes captured by the processed IR images. The lack of depth integration in the segmentation ultimately forced us to abandon this approach. However, it provided compelling evidence that the Fourier descriptor has great potential when paired with a higher resolution depth sensor.

6. Future Work

- Applying a threshold method for unassigned gestures
- Ability to add new gestures to the dictionary
- Trying a new distance metric (Mahalanobis)
- Trying a more sophisticated Classification scheme
- Using two hands simultaneously for combined gestures
- Adding more gestures from the single hand sign language alphabet

7. Conclusion

In conclusion, the Covariance Matrix Approach was able to perform gesture recognition with a CCR of 89.5% in MATLAB. Furthermore, we were able to implement it in C#/OpenCV to run in real time. In addition, we successfully implemented temporal dampening to minimize fluctuations in the classification by buffering the results and showing the mode over the last 30 frames (one second).
Unfortunately, because of time constraints, we were unable to add more gestures beyond the usual counting numbers from 0 to 5 after switching from the IR channel to the Depth stream. The only additional gestures added were alternate versions of the same 0 to 5 digits to account for the different ways that people might perform those gestures. Even so, we believe that this codebase could form the basis of a generalized gesture recognition system. Unlike the Du and Malima methods, the Fourier descriptor is looking at the overall shape, not just local contour information. So, the fact that it can successfully resolve any group of 6 shapes suggests that, in principle, there should be no issues adding any new gestures to the dictionary (provided they have a different enough silhouette).

REFERENCES

[1] J. Han, L. Shao, D. Xu and J. Shotton, "Enhanced Computer Vision With Microsoft Kinect Sensor: A Review," Cybernetics, IEEE Transactions on , vol.43, no.5, pp.1318,1334, Oct. 2013.
[2] J. Suarez, R.R. Murphy, "Hand gesture recognition with depth images: A review," RO-MAN, 2012 IEEE , vol., no., pp.411,417, 9-13 Sept. 2012.
[3] H. Du and T. To, "Hand Gesture Recognition Using Kinect," Boston University, 2011.
[4] A. Malima,. E. Ozgur and M. Cetin, "A Fast Algorithm for Vision-Based Hand Gesture Recognition for Robot Control," Signal Processing and Communications Applications, 2006 IEEE 14th , vol., no., pp.1,4, 17-19 April 2006.
[5] A . Kulshreshth, C. Zorn and J.J LaViola, "Poster: Real-time markerless Kinect based finger tracking and hand gesture recognition for HCI," 3D User Interfaces (3DUI), 2013 IEEE Symposium on, vol., no., pp.187, 188, 16-17 March 2013.
[6] D. Zhang and G. Lu, "A comparative study of Fourier descriptors for shape representation and retrieval," in Proc. 5th Asian Conference on Computer Vision, 2002.
[7] K. Guo, P. Ishwar and J. Konrad, "Action Recognition in Video by Covariance Matching of Silhouette Tunnels," Computer Graphics and Image Processing

(SIBGRAPI), 2009 XXII Brazilian Symposium on , vol., no., pp.299,306, 11-15 Oct. 2009.
[8] A Threshold Selection Method from Gray-Level Histograms," Systems, Man and Cybernetics, IEEE Transactions on , vol.9, no.1, pp.62,66, Jan. 1979.
[9] C. Rafael Gonzalez and E. Richard Woods, Digital Image Processing (3rd Edition). Prentice-Hal l, Inc., Upper Saddle River, NJ, USA, 2006.

AUTHOR PROFILES:

Bassem Khelil received the Computer Science Engineers Diploma from National Engineering School of Tunis (ENIT) in 2009 and the Master degree in Automatic and Signals Processing from ENIT in 2010. He is now a PHD student in Signal, Image and Information Technology at ENIT. His research interest includes Hand Gesture Recognition and 3D image processing.

Hamid Amiri received the Diploma of Electrotechnics, Information Technique in 1978 and the PhD degree in 1983 at the TU Braunschweig, Germany. He obtained the Doctorates Sciences in 1993. He was a Professor at the National School of Engineer of Tunis (ENIT), Tunisia, from 1987 to 2001; from 2001 to 2009 he was at the Riyadh College of Telecom and Information. Currently, he is again at ENIT. His research is focused on: Image Processing, Speech Processing, Document Processing and Natural language processing.