# A Real-Time Letter Recognition Model for Arabic Sign Language Using Kinect and Leap Motion Controller v2

Miada A. Almasre, Hana Al-Nuaim

Department of Computer Science, King AbdulAziz University, Jeddah, Saudi Arabia

***Abstract—*** *The objective of this research is to develop a supervised machine learning hand-gesturing model to recognize Arabic Sign Language (ArSL), using two sensors: Microsoft's Kinect with a Leap Motion Controller. The proposed model relies on the concept of supervised learning to predict a hand pose from two depth images and defines a classifier algorithm to dynamically transform gestural interactions based on 3D positions of a hand-joint direction into their corresponding letters whereby live gesturing can be then compared and letters displayed in real time.*

*This research is motivated by the need to increase the opportunity for the Arabic hearing-impaired to communicate with ease using ArSL and is the first step towards building a full communication system for the Arabic hearing impaired that can improve the interpretation of detected letters using fewer calculations.*

*To evaluate the model, participants were asked to gesture the 28 letters of the Arabic alphabet multiple times each to create an ArSL letter data set of gestures built by the depth images retrieved by these devices. Then, participants were later asked to gesture letters to validate the classifier algorithm developed. The results indicated that using both devices for the ArSL model were essential in detecting and recognizing 22 of the 28 Arabic alphabet correctly 100 %.*

***Keywords— Hand gesture recognition, Arabic Sign Language, Kinect version 2, Leap Motion Controller, skeleton***.

## I. INTRODUCTION

One of the current research areas in the field of computer vision is automatic hand-gesture recognition, especially in domains that serve hearing-impaired or deaf individuals. Hand gestures use the palms, finger positions, and shapes to create forms referring to different letters and phrases. Traditional vision-based methods that estimate hand poses lack robust image detection because they exploit a certain global-image feature like the contour or silhouette, which rely totally on input that could be imperfect and are basically 2D renderings of 3D hand poses [1]. These traditional methods rely on template matching, where the hand pose is obtained by the nearest-neighbor search in a vast set of templates, each of which contains a matching descriptor that does not allow full encoding of all needed information to extract complete pose parameters [2-4].

Current research tests different technologies that can be used to detect and interpret hand signs, such as tracking hand devices based on spatial-temporal features, hand-parsing-based color devices, and colored gloves or 3D hand-reference models [5-7]. Many devices used for gesture recognition map the user's body or hand gestures to specific features such as the whole hand, finger position, or velocity. It seems that most research focuses on exploiting these devices, but they rarely investigate their combined capabilities [8]. Microsoft Kinect and LMC, for example, are two of the latest devices used to create natural user interfaces (NUIs), which enhance communication between the user and the devices, relying on hand and body gestures and movements [8-9].

Microsoft Kinect relies on depth technology, which allows users to deal with any system via a Web camera through an NUI in which the user uses hand gestures or verbal commands to recognize objects without the need to touch the controller [9]. In 2013, LMC released a device that a user can use to interact with objects on screens through hand gestures. This device is a combination of hardware and software that tracks the movement of hands and fingers and then converts them into a 3D input [11-12].

This research follows the concept of supervised learning to predict the hand pose from two depth images and define a classifier that relies on hand-joint direction. In addition, it investigates how exploiting both Kinect's and LMC's skeleton detection is useful in the recognition of gestures used for Arabic Sign Language (ArSL) and whether their detection can improve letter interpretation with fewer calculations. We used 3D positions of one hand's joints and presented it as a 3D model. The proposed model is a real-time recognition model for any letter, and it is the first step toward building a full communication system for the hearing impaired.

## II. LITERATURE REVIEW

Hand detection and recognition are the main communication phases between humans and computers. In the hand-

detection phase, some researchers rely on a hand kinematic model and use it to mold the hand's shape as a skeleton or 3D hand shape [13]. The hand recognition or classification phase must actually pass through two steps: (a) hand-feature extraction, which uses many parameters to detect the hand-shape details by relying on the application type and goal, and (b) the gesture classification or recognition process, which translates the sign [14]. The algorithms and calculation methods used in these steps will vary depending on whether the gesture is static or dynamic [13-14].

The recent trend in pose estimation seems more efficient and robust compared to those that rely on global-image features because it fuses individual estimations obtained by many weak pose estimators that gather a subset of the pose parameters based on part of the input [15-16] to reconstruct the complete pose. This recent trend has been implemented in many projects in various fields including those related to sign-language recognition. Many projects have attempted to enhance the communication process with the hearing impaired; some examples include Web sites, mobile application or desk top like SignSpeak, the ClassAct, Tawasoul (Connect) and Maktabi (my office) application [17-20].

All such projects are used to translate and improve communication between the signer and hearing communities through vision-based sign-language interpretation or video-tracker technology. However, because of the lack of a standard dictionary (and even before an established sign language), home signs have been used by hearing-impaired communities as means of communication with each other [20]. These home signs were not part of a unified language but were still used as familiar motions and expressions employed within families or regions [20-21].

In 2014, a technique was introduced to predict the 3D joint positions from the depth images and the parsed hand parts [22]. Consequently, they enhanced the multimodal predictions produced by the previous regression-based method without making the calculations more complicated [22]. The prediction accuracy was significantly improved by the proposed method compared to other experiments that employed the joint locations from the depth images [22]. Since they presented a method that works on single depth images (static), they planned to analyze the dynamic constraints of the joint positions and track the hand-joint positions in the successive input images [22].

In Japan's Nara Women's University, researchers proposed a finger-spelling recognition method using LMC to provide an alternative method of voice input for the hearing impaired [23]. Different experiments were conducted to apply the genetic algorithm [23]. A quasi-optimal solution with a recognition rate of 82.71% was obtained, and therefore, the researchers recommended including the finger spelling of two letters with movement and that the decision tree employed should not be fixed [23].

A limited amount of research has been conducted over the last 5 years within the Arab world regarding the gesture recognition of ArSL. Nevertheless, a few exhaustive researches have been conducted to enhance a full recognition system that can be employed as a means of communication for the hearing impaired [24]. Each research usually focused on one part of gesture recognition: alphabet letters, isolated words, or continuous phrases [24].

As for letter recognition, El-Bendary, Zawbaa, Daoud, ella Hassanien, and Nakamatsu [25] at 2010, developed an Arabic-alphabet sign translator capable of identifying 30 characters in Arabic alphabet, with an accuracy of up to 91.3%. They employed rotation, scale, and translation features extracted from a video of signs, producing a text representing the letter [25]. In addition, Hemayed and Hassanien at 2010 presented a model that converted ArSL signs to voice [26]. As well as, an LMC device was introduced to implement a system for Arabic-alphabet sign recognition [27]. Only one signer was responsible for making 28 Arabic-alphabet signs, which was repeated 10 times for a total of 2,800 frames of data but they recommended two devices for more accuracy [27].

[28-29], Regarding isolated word recognition, [28] Shanableh and Assaleh (2007) presented various applications to recognize isolated ArSL gestures. Different spatiotemporal feature-extraction techniques were used with temporal features of a video-based gesture extracted through forward image predictions. The k-nearest-neighbors algorithm (KNN) and the Bayesian classifier both is used, and the results showed higher classification performances, ranging from 97% to 100% recognition rates [28].

[30-32] for continuous phrase recognition, [30] Assaleh at 2010, used a system based on spatiotemporal features and HMM models, and the result was a 94% average of word-recognition rate. However, recognized phrases were not always grammatically functional or were sometimes incoherent linguistically [30].

Indeed, to produce powerful recognition systems that are close to real-life accuracy for ArSL, researchers need to improve the accuracy of error detection and correction for two-way communication translation. When the size of the vocabulary is large, the difficulty of obtaining strong accuracy within the recognition system increases, but the system becomes more relevant [21]. The important point of real-life applications is to get better recognition accuracy with shorter processing times.

## III. RESEARCH OBJECTIVES

The recognition of ArSL poses some problems as with all sign languages due to the fact that the letters of ArSL are signed using gestures that are not always detected easily.

Sometimes there are difficulties with reading hand parts, such as the joints, the palm, the fingertip, etc. “fig. 1”. In such cases, predicting the meaning of a sign is difficult.

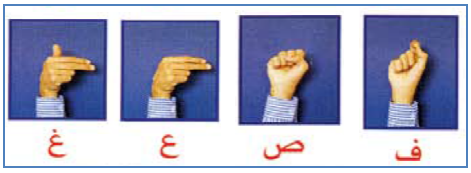



*Fig.1: Some ArSL issues.*

Some new sensor devices, such as LMC and Kinect, have some limitations that require further development. Using both devices may provide more details and enhance any hand gesture recognition system, especially if more devices are placed in different settings [21].

To build a translator system using several sensors, we need a compatible algorithm for keeping track of all high-precision parts. Therefore, the main objective of this research is to develop a supervised machine learning hand-gesturing model to recognize ArSL using Kinect with LMC. This paper describes the first step (phase) the researcher must take when constructing a Sign Language interpreter system, which uses a novel approach by combining Kinect V2 with LMC to assess the accuracy of ArSL classifications.

## IV. KINECT AND LMC

Microsoft’s Kinect Version 2.0, was used to track standing skeleton with high-depth fidelity. It has an RGB camera, voice recognition capability, face-tracking capabilities, and access to the raw sensor records [33] and [34].

The infrared (IR) emitter and IR depth sensor work together with the PrimeSense Chip. Through this chip, Kinect has a wider sensing range and tracks 24 joint points “fig. 2” [35-36].

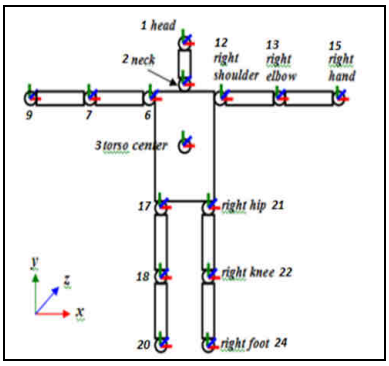



*Fig. 2. 24 joint points that Kinect detect 1*

LMC Version 2.0 was used to provide a skeletal tracking model that offers information about hands, fingers and overall hand tracking data even if the hands cross over each other “fig.3”. LMC uses a right-handed Cartesian coordinate system, and its application programming interface (API) measures different physical quantities’ units, such as measuring distance using millimeters, time using microseconds, and an angle using radians [37].

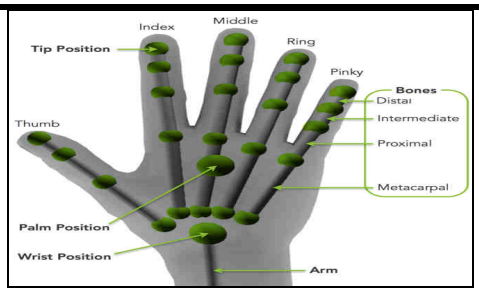



*Fig. 3: Hand skeleton points that LMC retrieves*

LMC has an interactive box that is a cube inside an inverse pyramid “fig. 4” that defines a rectilinear area (“rectilinear” means related to a straight line; it may refer to a rectilinear grid, a tessellation of the Euclidean plane) within the LMC field of view “fig. 4”. The size of this box is determined by the LMC field of view and the user’s setting in the application. When the device is placed on any table horizontally, the original point in the plane (0,0,0) is the center of the LMC, so it can detect any user’s hand above it [34]. However, many researchers, such as in our case, rotate the LMC so that a user’s hand gesture becomes more natural; instead of a hand hovering over the device, it can simply gesture normally in front of it. Thus, the user normally stands and makes sign language gestures.

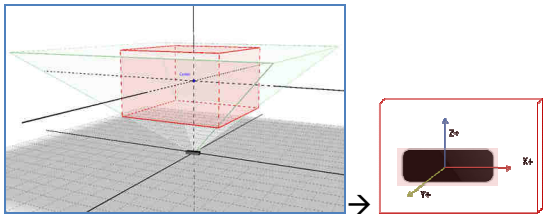


*Fig. 4. Rotate the view field (horizontal to vertical) plane.*

Like Kinect, LMC has an open-source library (SDK Leap Version2). This library provides APIs that retrieve many details about the hand, such as the direction and the index of each finger or each bone in the finger.

## V. THE PROPOSED MODEL

The proposed model of this research is to detect and recognize the hand gestures of ArSL letters using supervised learning and natural user interface libraries with the Kinect SDK Version 2 and LMC Kit as follows:

1.1. The devices calibration (joints):

Kinect and LMC give the same type of data that we need (depth of object) based on a skeleton algorithm. Thus, we get data about several objects (user hand and body) with high accuracy. Initially, the two devices are placed in the same direction with 30 centimeters between them “fig. 5”.

*Fig. 5. The tow device’s initial place.*

Kinect tracks the joints of the user's body, while LMC tracks the hand details of the same user. In this experiment, we used Microsoft Visual Studio with C# and SDK Version 2 of Kinect and LMC using a Windows Media3D namespace to present a 3D model of a hand that provided 3D object transformation, such as rotation and zooming.

Calibration, between two devices in our research, means drawing one skeleton from the details retrieved from two devices. The skeleton drawing is actually lines between joints using a vector class. We converted the length for each vector from meters, used by Kinect, to millimeters, used by Leap, to have standard length units in millimeters. This class exists in the leap SDK and vector3D class that exist in the .Net C# library. Moreover, the skeleton part that we need is just the upper part of the user's body (head, neck, shoulder, arm, spin, and hand with fingers). The vector is a structure that represents a three-component mathematical vector or point, such as a direction or position in a 3D space. Fig. 6 shows the vector3D with drawing the complete skeleton of the user.


*Fig. 6:Defined vectors in the two devices with skeleton appear on user.*

We stored the hand information in an XML file instead of using tables in the relational database "fig.9".

Fig. 7 presents the first perspective structure that explains the main concept of hand detection and recognition for ArSL letters using Kinect and LMC.

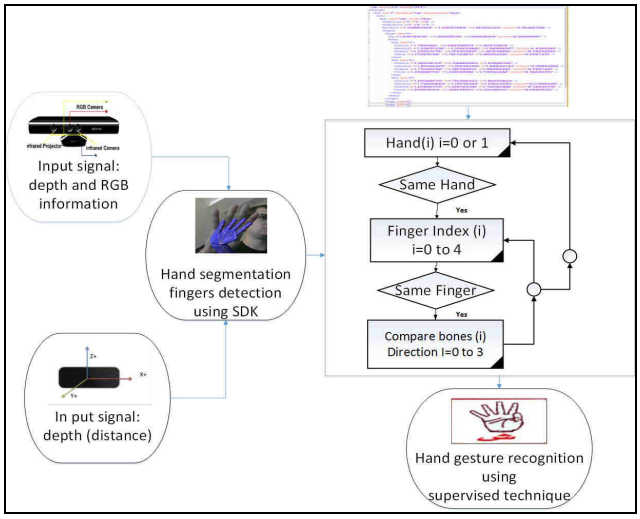



*Fig. 7. The general process of the experiment level 0.*

1.2. The testing environment

The data collection was done in a room with white light. The four participants (two males and two females) their age between fifteen to forty years to capture different hand sizes, styles, speeds, and hand orientations for each gesture. They were not deaf; rather, they were able to mimic sign gestures.

Each participant stood in a position from 1 to 2 meters away from the front of the Kinect device, and 30 to 50 cm away from the front of the LMC.

Kinect and LMC were mounted on a table 70 cm from the floor. The two devices were placed on the same horizontal surface and 50 cm apart. The background behind each participant was different because it was captured in different places. Fig. 8 presents two users in the testing environment with the main interface of the application that we built to apply our methodology.

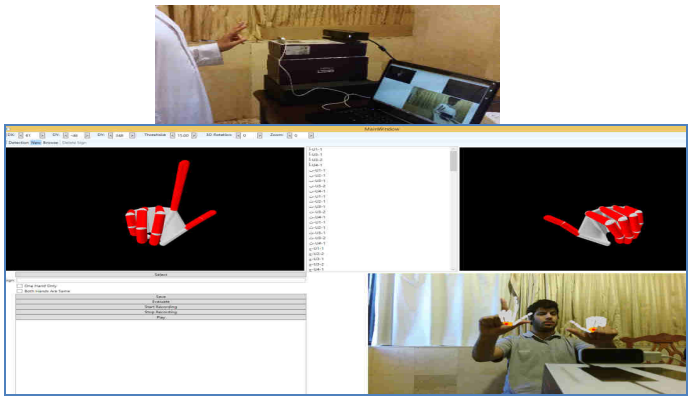


*Fig. 8. Experimental environment with non-deaf participants with the main window interface.*

The capturing procedure used Microsoft Kinect and LMC as an input device, and all of the signals that the LMC device captured were activated, but only the depth and color frames were activated in Kinect. Then, all of the users were asked to gesture all letters continuously by making the "new" gesture and then click on the "select" button for each letter.

For letters with more than one movement, the process of recording used then the comparison made on frames, where each frame represents one sign.

An evaluation button is used to export any live sign to a Microsoft Excel sheet with all of the calculations between this unknown gesture and all stored gestures for classifying the gesture and checking if it is a match with any existing gesture.

1.3. Data set collection

To evaluate the calculation process, we collected an experimental dataset with all 28 ArSL letters. Each of the four participants performed each letter at least two times. Thus, there were 28 × 4 × 2 = 224 signs for all letters. The average for one user was 28 gestures in 20 minutes.

The depth data hand gesture dataset collected for these cases using LMC and Kinect consisted of 3D coordinate values (x,y,z) and an RGB image. Bone direction was the main

feature, which was used to compare the hand gestures. The four other features were used to draw each bone finger as a 3D model: the fingertip, the bone center point, the bone from point, and the bone to point.

We attempted to find out how LMC and Microsoft Kinect would recognize and translate ArSL from the depth images of the sensor to ordinary speech or text.

Fig. 9 shows the features of the first bone in the thumb; it is an XML file that is a snapshot of the sample of the dataset for the letter "أ". We attempted to find out how LMC and Microsoft Kinect would recognize and translate ArSL from the depth images of the sensor to ordinary speech or text.

The data file contains many tags that describe the letter, such as:

- Sign name: the name of the gesture that the user enters in the class name field
- Hand: contains a tag for recognizing the left or right hand, palm position, palm direction, and wrist point finger. These are the values with respect to the x, y, and z axes that were obtained from LMC.
- Fingers: the finger index, the (x,y,z) coordinates, and distance of the fingertip
- Bones: the bone index, direction, center point, and from point and to point with respect to the (x,y,z) coordinates and distance

*Fig. 9:Gesture dataset XML file.*

The threshold value is a point or value that the gesture accuracy can affect. Through experimentation, researchers can determine a threshold that is useful for reducing the computation in the search process, as it becomes the point separating the accepted from the rejected values. For instance, if the value is under the threshold value, the gesture is accepted; otherwise, the gesture is rejected if it goes beyond the threshold. The threshold value used in this research was from 15 degrees because when the hand leans more than this value, the sign will be different.

For each joint point, we drew a line between each corresponding point (vector). Each vector has a corresponding direction and an angle, and each angle was measured based on the corresponding x, y , z angles. These angles are the main factor for a comparison between two gestures. To clarify the calculation process, the following example presents a certain value for a live gesture compared with a stored one "Fig. 10".

The first value (point A in the live gesture) represents the first bone in the thumb: Point (A) (x1, y1, z1) = (1.78, 0.44, 1.95). If we need to compare it with the same bone in the first stored sign, which is (x2, y2, z2) = ( 0.92, 0.66, 1.51), we have to apply the difference between each value, as in the following equation:

X=ABS(180 * x1 /360-180*x2/360) = 49.59

Y =ABS(180 * y1 /360-180*y2/360) = 12.56

Z =ABS(180 * z1 /360-180*z2/360) = 25.

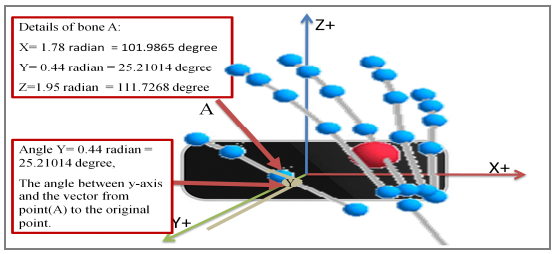



*Fig. 10:Clarify skeleton on device.*

Thus, the result shows the value more than the threshold, so the live gesture did not match the first stored value. Keep in the mind that the comparison was between the same index of the hand, finger, and bone.

1.4. Training phase

Using the distance or depth value for each hand skeleton point stored in an XML file, each bone direction in an unknown live gesture was compared with the same bone index in the stored signs. If more than one gesture gave an accepted value, we would calculate the average of the comparison bones and then accept the minimum one.

"Fig. 11" shows a snapshot from the training phase for the gesture of the letter "ب", pronounced "ba". The first participant clicked on the "new" button using a mouse, and then the participant made the gesture of the letter and clicked "select" to fix the gesture; next, the participant clicked "save" to store it. The participant repeated the same process for all letters, and so on for all participants. The testing phase "detection" had enough information to classify or recognize the gesture as in "fig. 12". The information about gestures appears under the left-hand model, where the sign text is, for example: "ت-user1-2" means that the gesture is "ت," pronounced "ta." It matched the stored one made by participant number 1 in his second attempt "1-2." In addition, a timestamp appeared to give us some indication about the response time in milliseconds.

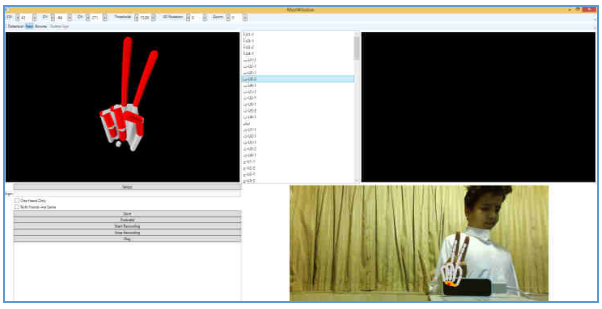


*Fig. 11:A snapshot from the training phase.*

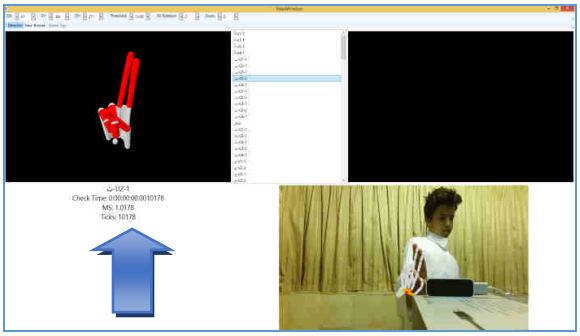
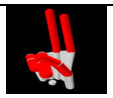


*Fig. 12. A snapshot from the classification phase.*

Table I presents samples of the same letters "ت " and " ج" signed by four participants. In the training phase, we repeated some gestures more than one time because some participants felt tired and the device became too slow to detect due to heating.

*TABLE I. SAMPLE OF "ت" AND "ج" LETTERS SIGNED BY THREE PARTICIPANTS*

| | Gesture of sign ت | Gesture of sign ج |
|---|---|---|
| User1 |  |  |
| User2 |  |  |
| User3 |  |  |

1.5. The algorithm of gesture classification

The bone direction was used to detect the gesture meaning using a skeletal algorithm that the two devices feature. To check the similarity of the letters, the comparison step between each of the two bones that had the same index in both hands is the core of our algorithm. The algorithm with four steps of the proposed model is:

1-Initializing values:

- Threshold= threshold numbers (angle);
- List [Sign details] = XML file.
- Local Info= Coordinates value (X,Y,Z)
- IsSimilar (Local Info Value1, Local Info Value2, Threshold)

Get RGB from Kinect; then, draw a hand by vectors between each joint, line (from previous joint to next joint), and points based on LMC coordinates

- Calculate the direction value for each bone (pervious joint, next joint)

2- Get angles:

- X= Angle between the vector of point(i) around X-axis
- Y= Angle between the vector of point(i) around Y-axis
- Z= Angle between the vector of point(i) around Z-axis

3- IsSimilar (Local Info Value1(x,y,z), Local Info Value2(x,y,z), Threshold) \\ If (Same hand index and same finger index)

- Loop all points to check the difference between value 1 and value 2;
- If point value < threshold → matched
- Else not matched and exist
- Calculate the AVR for (X, Y , Z).
- Calculate the average of all points (all hand/ finger joints).

4- If the result contains more than one hand, then select the hand with minimum AVR.

For example, the measurement of the similarity bones' direction between the unknown gesture skeleton points and one of the gestures in the dataset (e.g., sign of the number 2). The following example provides the calculation process; "Fig. 13" shows the values of each point in the hand skeleton, and Table II contains example of the full point values obtained by LMC for thumb and index where we had the same for each bone in the hand.

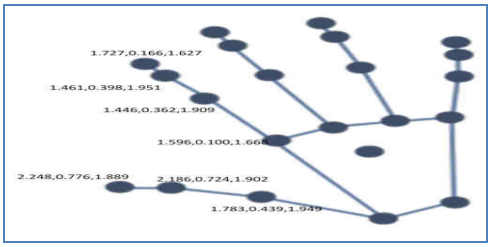



*Fig. 13: Values of some hand skeleton point.*

*TABLE II. VALUES OBTAINED BY LMC FOR EACH BONE IN THE HAND*

| | | | **Unknown Sign** | | | **Sign of number 2** | | |
|---|---|---|---|---|---|---|---|---|
| | | | ***Live Sign angle values between point and one coordinate (Radian)*** | | | ***Stored Sign angle values between point and one coordinate (Radian)*** | | |
| | ***Finger name*** | ***Bone*** | ***X1*** | ***Y1*** | ***Z1*** | ***X2*** | ***Y2*** | ***Z2*** |
| 1 | 0-thumb | 1 | 1.78 | 0.44 | 1.95 | 0.92 | 0.66 | 1.51 |
| 2 | 0-thumb | 2 | 2.19 | 0.72 | 1.90 | 1.46 | 0.36 | 1.91 |
| 3 | 0-thumb | 3 | 2.25 | 0.78 | 1.89 | 2.36 | 1.24 | 2.24 |
| 4 | 1-index | 0 | 1.6 | 0.1 | 1.6 | 1.4 | 0.2 | 1.3 |

| | | | **Unknown Sign** | | | **Sign of number 2** | | |
|---|---|---|---|---|---|---|---|---|
| | | | ***Live Sign angle values between point and one coordinate (Radian)*** | | | ***Stored Sign angle values between point and one coordinate (Radian)*** | | |
| | ***Finger name*** | ***Bone*** | ***X1*** | ***Y1*** | ***Z1*** | ***X2*** | ***Y2*** | ***Z2*** |
| | | | 0 | 0 | 7 | 9 | 1 | 8 |
| 5 | 1-index | 1 | 1.43 | 0.32 | 1.85 | 1.15 | 1.00 | 2.40 |
| 6 | 1-index | 2 | 1.45 | 0.36 | 1.91 | 1.44 | 2.33 | 2.37 |
| 7 | 1-index | 3 | 1.46 | 0.40 | 1.95 | 1.72 | 2.91 | 1.74 |

Thus, the same calculations will apply, and the above table values become "Table III". The last column in Table III tests the matching of an unknown letter with the sign of the number two, based on the threshold, where there are four values over 15, which means the unknown sign was not matched with the sign of the number two. Each letter passes a comparison process, which is:

- Same Hand (Left, Right)=2
- Same Finger with the same index and direction (0=Thumb, 1=index, 2=Middle, 3=Ring, 4=little)=5
- Same Bone with the same index and direction (0=metacarpal, 1=Proximal, 2=intermediate, 3=distal), where the thumb has just three bones = 3 + 4 (4) = 19

Thus, if the letter received 19 yes's, that means it was a match; otherwise, it was not. If it was matched with more than one sign, the stored sign with the minimum average will be taken. The bar chart of "fig. 14" illustrates the comparison of each stored letter with letter "ت" pronounced "ta" bones. It was measured by counting the number of "yes" and "no" matches "fig. 15".

- "Yes" means all of the bone directions matched, and "no" means none were a match.
- The y-axis is the number of "yes" and "no" matches (it will be between 0 and 19).
- The x-axis is the sign of users for all stored letters (u1-1-أ), meaning the first participant in the first attempt for letter "أ".

Overall, it can be seen that the letter "ت" matches correctly with participant 2 in his first attempt and participant 3 with his second attempt where the comparison count was equal to 19. The line graph compares the average of letter "ت" with all of the stored letters, and it shows that the minimum average matched with the same letter for participant 2 in his first attempt, so it will be the matched result.

Some of the results were incorrect, such as those for the letter "هـ" which the device cannot get correctly. It could be because many fingers need to cross over one another, or since it is the final letter in the alphabet, the participants could have been fatigued, or the device may have heated up. Even if the participant made the same sign gesture several times, the devices were not able to capture it properly, so one extra camera (maybe another LMC) might have been needed to capture the sign from a different angle. Letters with multiple frames need a special algorithm to take certain frames from each sign. Thus, comparing becomes more efficient.

In spite of these few cases, the recognition of the other letters was correct 100%, especially for the letters that had clear shapes, such as "ض, م, ي, ش, ث, ت, ب, س, ن , ل, د, ص".

## VI. DISCUSSION AND CONCLUSION

This research examined 224 matching cases of 28 letters. We found the class name (correct letter) with high accuracy for more than 20 letters because the matched letter value was less than 15 degrees over the threshold. Meanwhile, the other letters' values were not only more than the threshold but also very far away from the same letter because the device obtained data from one direction. Some critical cases included difficulty in counting the joints, or the palm might have been covered by the fingers. Thus, letters with overlapping fingers or letters with multiple frames need more processing to be accurate.

The bone direction feature that was used in our algorithm gave very high accuracy and did not care about the user's hand position. This means that if the participants stood in a position and made gestures, there was no need in the next iteration to stand in the same position. The hand did not need to be in the same position, either. This was true even in the dynamic classification of letters (once the hand changes, the letter class changes directly). However, the limitation of using both LMC and Kinect is that each depends on one direction. For the accurate recognition of hand gestures in LMC, only one person should stay in the visible range, which is only 57 degrees in front of the device and 50 cm away from the device (users' movement is restricted).

Using Kinect could possibly overcome a crucial problem for Arab hearing-impaired people that manifests in the lack of ArSL interpreters and the excessive variation of ArSL dialects. Hearing-impaired people who use such a platform (Kinect and Leap) as an interpreter could actually be offered the chance to be more interactive if the device is fed with enough data about these various dialects.

*TABLE III. VALUES AFTER APPLYING THE CALCULATIONS*

| | Finger name | Bone | Unknown Sign | | | Sign of number 2 | | | Average of similarity | Less than Thresh old (15) |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Live Sign angle values between point and one coordinate (Radian) | | | Similarity percentage (value/360) | | | | |
| | | | X | Y | Z | X/360 | Y/360 | Z/360 | | |
| 1 | 0-thumb | 1 | 33.623 | 12.494 | 7.580 | 9.34% | 3.47% | 2.11% | 4.97% | Yes |
| 2 | 0-thumb | 2 | 20.807 | 3.700 | 22.642 | 5.78% | 1.03% | 6.29% | 4.37% | Yes |
| 3 | 0-thumb | 3 | 8.033 | 24.307 | 17.238 | 2.23% | 6.75% | 4.79% | 4.59% | Yes |
| 4 | 1-index | 0 | 3.023 | 7.658 | 7.740 | 0.84% | 2.13% | 2.15% | 1.71% | Yes |
| 5 | 1-index | 1 | 3.272 | 62.643 | 63.421 | 0.91% | 17.40% | 17.62% | 11.98% | Yes |
| 6 | 1-index | 2 | 6.997 | 103.946 | 35.900 | 1.94% | 28.87% | 9.97% | 13.60% | Yes |
| 7 | 1-index | 3 | 8.744 | 123.427 | 11.890 | 2.43% | 34.29% | 3.30% | 13.34% | Yes |
| 8 | 2-Middle | 0 | 2.691 | 5.578 | 10.466 | 0.75% | 1.55% | 2.91% | 1.73% | Yes |
| 9 | 2-Middle | 1 | 19.019 | 15.298 | 7.568 | 5.28% | 4.25% | 2.10% | 3.88% | Yes |
| 10 | 2-Middle | 2 | 21.267 | 9.027 | 2.349 | 5.91% | 2.51% | 0.65% | 3.02% | Yes |
| 11 | 2-Middle | 3 | 22.976 | 3.749 | 2.229 | 6.38% | 1.04% | 0.62% | 2.68% | Yes |
| 12 | 3-Ring | 0 | 2.376 | 3.366 | 13.235 | 0.66% | 0.93% | 3.68% | 1.76% | Yes |
| 13 | 3-Ring | 1 | 5.441 | 53.755 | 89.312 | 1.51% | 14.93% | 24.81% | 13.75% | Yes |
| 14 | 3-Ring | 2 | 15.621 | 91.278 | 82.127 | 4.34% | 25.35% | 22.81% | 17.50% | No |
| 15 | 3-Ring | 3 | 19.905 | 110.122 | 61.538 | 5.53% | 30.59% | 17.09% | 17.74% | No |
| 16 | 4-Little | 0 | 1.180 | 1.591 | 15.439 | 0.33% | 0.44% | 4.29% | 1.69% | Yes |
| 17 | 4-Little | 1 | 28.895 | 35.547 | 94.157 | 8.03% | 9.87% | 26.15% | 14.69% | Yes |
| 18 | 4-Little | 2 | 46.680 | 69.417 | 85.296 | 12.97% | 19.28% | 23.69% | 18.65% | No |
| 19 | 4-Little | 3 | 53.066 | 84.655 | 67.394 | 14.74% | 23.52% | 18.72% | 18.99% | No |

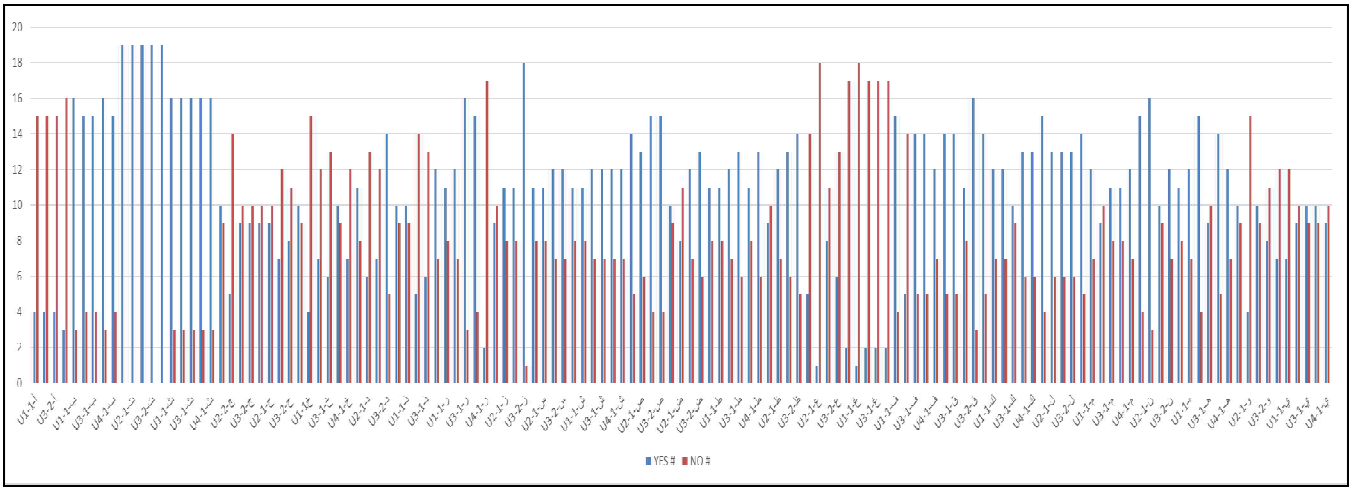


*Fig. 14. Bar chart of evaluation of the "ت" letter.*

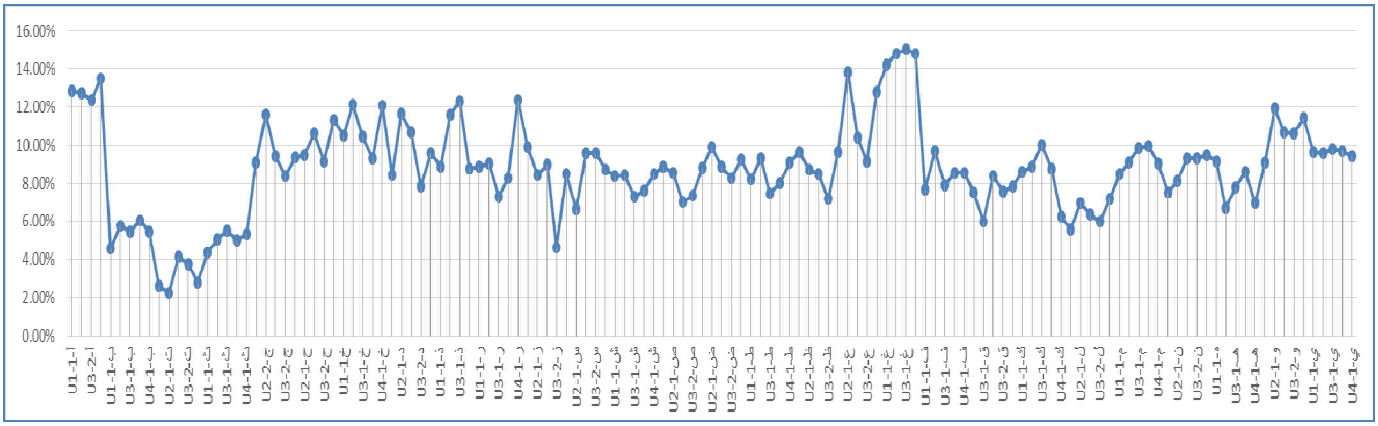



*Fig. 15. Line chart of evaluation of the "ت" letter.*

## REFERENCES

[1] Sarkat, A., Sanyal, G., & Majumder S. (2013). Hand gesture recognition system: A survey. International Journal of Computer Applications, 71(15), 0975–8887.

[2] Ballan, L., Taneja, A., Gall, J., Gool, L. V., & Pollefeys, M. (2012). Motion capture of hands in action using discriminative salient points. Paper presented at European Conference on Computer Vision (ECCV), Florence, Italy. Springer: Springer Berlin Heidelberg.

[3] Oikonomidis, I. Kyriazis, N., & Argyros, A. A. (2011). Efficient model-based 3D tracking of hand articulations using Kinect. Paper presented at British Machine Vision Conference (BMVC), UK. Place of Publication: University of Dundee.

[4] Wang, R. Y., & Popović, J. (2009). Real-time hand-tracking with a color glove. ACM Transactions on Graphics (TOG), 28(3), 63.

[5] Hui, L., & Junsong, Y. (2014). Hand parsing and gesture recognition with a commodity depth camera. In L. S. Ling, J. Han, P. Kohli, & Z. Zhang (Eds.), Advances in computer vision and pattern recognition (pp. 239–265). Switzerland: Spring International Publishing.

[6] Liang, H., Yuan, J., Thalmann, D., & Zhang, Z. (2013). Model-based hand pose estimation via spatial-temporal hand parsing and 3D fingertip localization. The Visual Computer International Journal of Computer Graphics, 29(6-8), 837-848.

[7] Lingchen, C., Feng, W., Hui, D., & Kaifan, J. (2013). A survey on hand gesture recognition. Proceedings of the International Conference on Computer Sciences and Applications, Kunming, China. Place of publication: IEEE Xplore Digital Library.

[8] Suarez, J., & Murphy, R. (2012). Hand gesture recognition with depth images: A review. Paper presented at the 21st IEEE International Symposium on Robot and Human Interactive Communication, Paris, France. Place of Publication: IEEE Xplore Digital Library.

[9] Han, J., Shao, L., Xu, D., & Shotton, J. (2013). Enhanced computer vision with Microsoft Kinect Sensor: A review. IEEE Trans Cybern, 43(5), 2168–2267.

[10] Chen, X., Li, H., & Tansley, S. (2013). Kinect Sign Language Translator expands communication possibilities. Retrieved from http://research.microsoft.com/en-us/collaboration/stories/kinectforsignlanguage_cs.pdf

[11] Leap Motion – Best practices guidelines. (2015) Available from https://developer.leapmotion.com/documentation/

[12] Marin, G., Dominio, F., & Zanuttigh, P. (2015,7). Hand gesture recognition with jointly calibrated Leap Motion and depth sensor. Multimedia Tools and Applications journal, 1380-7501, 1-25.

[13] Murthy, G., & Jadon, R. (2009). Review of vision based hand gestures recognition. International Journal of Information Technology and Knowledge Management, India, 2(2), 105–410.

[14] Ali, E., Mircea, N., Richard, B., & Xander, T. (2007). Vision-based hand pose estimation: A review. Computer Vision and Image Understanding, 108(1), 52–73.

[15] Hsiang, Y., & Han, J. (2014). Real-time dynamic hand gesture recognition. Proceedings of the International Symposium on Computer, Consumer and Control, Taichung. IEEE Xplore Digital Library: IEEE.

[16] Li, Y. (2012, June). Hand gesture recognition using Kinect. In Software Engineering and Service Science (ICSESS), 2012 IEEE 3rd International Conference on (pp. 196-199). IEEE.

[17] SignSpeak. (2009). Retrieved from http://www.signspeak.eu/en/index.html.

[18] ClassAct. (n.d.). Retrieved from https://www.deaftec.org/classact

[19] tawasoul4arsl. (2012). Retrieved from http://tawasoul4arsl.ksu.edu.sa/

[20] Senghas, R. J., & Monaghan, L. (2002). Signs of their times: Deaf communities and the culture of language. Annual Reviews of Anthropology from Journal Storage (JSTOR), 31(2002), 69–97.

[21] Mohandes, M., Liu, J., & Deriche, M. (2014, February). A survey of image-based arabic sign language recognition. In Multi-Conference on Systems, Signals & Devices (SSD), Barcelona, 2014 11th International (pp. 1-4). IEEE.

[22] Hui, L., Junsong, Y. Y., & Daniel, T. (2015). Resolving ambiguous hand pose predictions by exploiting part correlations. IEEE, 25(7), 1125–1139.

[23] Youssif, A., Aboutabl, A., & Ali, H. (2011). Arabic Sign Language (ArSL) recognition system using HMM. International Journal of Advanced Computer Science and Applications, 2(11), 45–51.

[24] EI-Bendary, N., Zawbaa, H. M., Daoud, M. S., Hassanien, A. E., & Nakamatsu, K. (2010). ArSLAT: Arabic Sign Language Alphabets Translator. Paper presented at the 2010 International Conference on Computer Information Systems and Industrial Management Applications (CISIM), Krackow. IEEE Xplore Digital Library: IEEE.

[25] Hemayed, E. E., & Hassanien, A. S. (2010). Edge-based recognizer for Arabic sign language alphabet (ArS2VArabic sign to voice). Paper presented at the Computer Engineering Conference (ICENCO), 2010 International, Giza. IEEE Xplore Digital Library: IEEE.

[26] Mohandes, M., Aliyu, S., & Deriche, M. (2014). Arabic sign language recognition using the Leap Motion Controller. Paper presented at King Fahd University of Petroleum & Minerals Dhahran, 31261, Saudi Arabia. Place of Publication: IEEE Xplore Digital Library.

[27] Samir, A., & Aboul-Ela, M. (2012). Error detection and correction approach for Arabic sign language recognition. Paper presented at the 2012 Seventh International Conference on Computer Engineering Systems (ICCES), Cairo. Place of Publication: IEEE Xplore Digital Library.

[28] Shanableh, T., & Assaleh, K. (2007). Video-based feature extraction techniques for isolated Arabic sign language recognition. Paper presented at the 9th International Symposium on Signal Processing and Its Applications, Sharjah. Place of Publication: IEEE Xplore Digital Library.

[29] Mohandes, M., Deriche, M., Johar, U., & Ilyas, S. (2012). A signer-independent Arabic Sign Language recognition system using face detection, geometric features, and a Hidden Markov Model. Comput. Electr. Eng., 38(2), 422–433.

[30] Assaleh, K. (2010). Continuous Arabic sign language recognition in user dependent mode. J Intel. Learn. Syst. App!, 2(1), 19–27.

[31] Tolba, M. F., Samir, A., & Abul-Ela, M. (2012). A proposed graph matching technique for Arabic sign language continuous sentences recognition. Paper presented at the 2012 8th International Conference on Informatics and Systems (INFOS), Cairo. Place of Publication: IEEE Xplore Digital Library.

[32] Spiegelmock, M. (2015). Leap Motion Development Essentials packet. Retrieved from www.packtpub.com/leap-motion-development-essentials/book

[33] Jana, A. (2012). Kinect for Windows SDK programming guide. Birmingham B3 2PB, UK: Packt Publishing Ltd.

[34] Chen, X., Li, H., Pan, T., Tansley, S., & Zhou, M. Kinect Sign Language Translator expands communication possibilities. Retrieved from http://research.microsoft.com/en-us/collaboration/stories/kinect-sign-language-translator.aspx.