computers

MDPI

Article

Elderly Care Based on Hand Gestures Using Kinect Sensor

Munir Oudah [1], Ali Al-Naji [1,2,*] and Javaan Chahl [2]

[1] Electrical Engineering Technical College, Middle Technical University, Baghdad 10022, Iraq; Munir_aliraqi@yahoo.com
[2] School of Engineering, University of South Australia, Mawson Lakes, SA 5095, Australia; Javaan.Chahl@unisa.edu.au
* Correspondence: ali_al_naji@mtu.edu.iq; Tel.: +964-7710304768

Abstract: Technological advances have allowed hand gestures to become an important research field especially in applications such as health care and assisting applications for elderly people, providing a natural interaction with the assisting system through a camera by making specific gestures. In this study, we proposed three different scenarios using a Microsoft Kinect V2 depth sensor then evaluated the effectiveness of the outcomes. The first scenario used joint tracking combined with a depth threshold to enhance hand segmentation and efficiently recognise the number of fingers extended. The second scenario utilised the metadata parameters provided by the Kinect V2 depth sensor, which provided 11 parameters related to the tracked body and gave information about three gestures for each hand. The third scenario used a simple convolutional neural network with joint tracking by depth metadata to recognise and classify five hand gesture categories. In this study, deaf-mute elderly people performed five different hand gestures, each related to a specific request, such as needing water, meal, toilet, help and medicine. Next, the request was sent via the global system for mobile communication (GSM) as a text message to the care provider's smartphone because the elderly subjects could not execute any activity independently.

Keywords: elderly care; hand gesture; embedded system; Kinect V2 depth sensor; simple convolutional neural network (SCNN); depth sensor

Citation: Oudah, M.; Al-Naji, A.; Chahl, J. Elderly Care Based on Hand Gestures Using Kinect Sensor. *Computers* **2021**, *10*, 5. https://doi.org/10.3390/computers10010005

Received: 23 November 2020
Accepted: 21 December 2020
Published: 26 December 2020

1. Introduction

The aged population in the world is increasing by nine million per year and is expected to reach more than 800 million by 2025 [1]. Therefore, an increase in the demands of the various sponsorship programs is expected. In addition, home care is cost-effective, especially for long-term care provided inside specialised facilities. Additionally, it has a positive effect on elderly people when provided care service in their own homes. This paper proposes a remote natural interaction system for elderly disabled people who are speechless due to sudden stroke, medical accident or who are already deaf-mute, who have difficulty communicating with other family members at home, especially for providing daily routine needs.

Previously, human-computer interaction (HCI) based on camera imaging systems used a variety of techniques and provided natural interaction using hand gestures by making particular gestures in front of a camera. Where this technique has some challenges, such as complex background [2], lighting conditions [3], occlusions [4], detection distance [5] and in cases using RGB cameras the system cannot work in dim or dark environments regardless of algorithms.

Many research systems have proposed different hand gestures with regard to computer vision techniques for different applications which have shown some drawbacks, as mentioned in the previous paragraph that effect recognition rate. However, the Kinect sensor offers a sensor modality that helps to overcome some challenges with a depth sensor that gives 3D x, y, z coordinates of an object by analysing data returned by the depth

sensor based on an infrared projector, that effectively overcomes lighting and background limitations.

This study proposes a non-contact natural interaction system for assisting elderly people by performing specific gestures in front of a camera in any light conditions, where these gestures are translated as a request and sent via SMS to the care provider or family member's smartphone. In addition, the study provides a comparison between three different techniques using the Kinect V2 sensor in order to validate the system.

The rest of this paper is arranged as follows: Section 2 presents the related works and mentions the weaknesses of former works. Section 3 describes the materials and methods, including the participants and experimental setup, hardware design and hand gesture scenarios. Section 4 shows the experimental results and discusses the obtained results. Finally, conclusion and future research directions are provided in Section 5.

2. Related Works

In the last decade, many papers with regard to processing hand gestures were published and have become an interesting topic for researchers. Where some of these studies have considered a range of different applications. However, the hand gesture interaction systems depend on recognition rate which is affected by some factors, including the type of camera used and its resolution, the technique utilised for hand segmentation and the recognition algorithm used. This section summarises some key papers with respect to the use of the Microsoft Kinect depth sensor for hand gesture recognition techniques, as shown in Table 1.

Table 1. A set of research papers that used Kinect depth sensor for hand gestures.

Author	Type of Camera	Resolution	Techniques/Methods for Segmentation	Feature Extract Type	Classify Algorithm	Recognition Rate	No. of Gestures	Application Area	Invariant Factor	Distance from Camera
Ren et al. [6] 2011	Kinect V1	640 × 480 320 × 240	depth map & colour image	finger	Near-convex Decomposition & Finger-Earth Mover's Distance (FEMD)	93.9%	10-gesture	HCI applications	No	No
Wen et al. [7] 2012	Kinect V1	Depth– 640 × 480	skin colour segmentation and depth joint	fingertip	K-means clustering & convex hull	No	fingertip gesture	human-computer interaction (HCI)	No	No
Li et al. [3] 2012	Kinect V1	640 × 480 320 × 240	Depth thresholds	fingertip	K-means clustering algorithm convex hulls	90%	9-gesture	real-time communication such as chatting with speech	the difficulty of recognising one of the nine gesture	0.5 to 0.8 m
Lee et al. [8] 2013	Kinect V1	640 × 480 320 × 240	3D depth sensor	fingertips	shape bases matching	91%	6-gesture	finger painting and mouse controlling	low accuracy in rough conditions	0.5 to 0.8 m
Ma et al. [9] 2013	Kinect V1	640 × 480 320 × 240	depth threshold	fingertip	k-curvature algorithm	No	5-gesture	human-robot interactions	No	No
Marin et al. [10] 2014	Kinect V1	depth– 640 × 480	depth and colour data & leap motion	Position of the fingertips	multi-class SVM classifier	91:3%	10-gesture	a subset of the American Manual Alphabet	Leap motion is limit while Kinect provides the full depth map.	No
Bakar et al. [11] 2014	Kinect V1	depth– 640 × 480	threshold range	hand gesture	No	No	hand gesture	hand rehabilitation system	No	0.4–1.5 m
Bakar et al. [12] 2015	Kinect V1	depth– 640 × 480	depth threshold and K-curvature	finger counting	depth threshold and K-curvature	73.7%	5 gestures	picture selection application	detection fingertips should though the hand was moving or rotating	No
Karbasi et al. [13] 2015	Kinect V1	depth– 640 × 480	distance method	hand gesture	No	No	hand gesture	human-computer interaction (HCI)	No	No

Table 1. *Cont.*

Author	Type of Camera	Resolution	Techniques/Methods for Segmentation	Feature Extract Type	Classify Algorithm	Recognition Rate	No. of Gestures	Application Area	Invariant Factor	Distance from Camera
Kim et al. [14] 2016	Kinect V2	depth– 512 × 424	operation of depth and infrared images	finger counting & hand gesture	number of separate areas	No	finger count & two hand gestures	mouse-movement controlling	No	<0.5 m
Pal et al. [15] 2016	Kinect V1	640 × 480 320 × 240	skin and motion detection & hu moments	dynamic hand gesture	Discrete Hidden Markov Model	Table	single handed postures combination of position, orientation & 10-gesture	controlling DC servo motor action	backward movement gesture effect recognition rate	No
Desai et al. [16] 2017	Kinect V1	depth– 640 × 480	range of depth image	hand gestures 1–5	kNN classifier & Euclidian distance	88%	5 gestures	electronic home appliances	No	0.25–0.65 m
Desai et al. [17] 2017	Kinect V2	RGB– 1920 × 1080 depth– 512 × 424	Otsu's global threshold	finger gesture	kNN classifier & Euclidian distance	90%	finger count	Human computer interaction (HCI)	hand not identified if it's not connected with boundary	0.25–0.65 m
Xi et al. [18] 2018	Kinect V2	1920 × 1080 512 × 424	threshold and recursive connected component analysis	hand skeleton & fingertip	Euclidean distance and geodesic distance	No	hand motion	controlling actions and interactions.	occlusions may have side effects on the depth data.	No
Li et al. [19] 2018	Kinect V2	RGB– 1920 × 1080 depth– 512 × 424	double-threshold segmentation and skeletal data	fingertip	fingertip angle characteristics & SIFT key points	No	10-gesture	(HCI)	some constraints are set in hand segmentation	No
Ma et al. [5] 2018	Kinect V2	1920 × 1080 512 × 424	threshold segmentation & local neighbour method	fingertip	convex hull detection algorithm	96 %	6-gesture	natural human-robot interaction.	Some small noise spots around the hands will reduce the detection performance of fingertips	0.5 to 2.0 m
Bamwenda et al. [20] 2019	Kinect V2	depth– 512 × 424	skeletal data stream & depth & colour data streams	hand gesture	support vector machine (SVM) & artificial neural networks (ANN)	93.4% for SVM 98.2% for ANN	24 alphabets hand gesture	American Sign Language	No	0.5–0.8 m

A study by Ren et al. [6] proposed a new method based on the finger earth mover distance (FEMD) approach that was evaluated in terms of speed and precision and then compared with a shape-matching algorithm using the depth map and colour image acquired by a Kinect camera. Wen et al. [7] proposed a gesture recognition system in order to segment the hand based on skin colour and used K-means clustering and convex hull to identify hand contour and finally detect fingertips. In another study by Li et al. [3], where a depth threshold was used to segment the hand and then a K-mean algorithm was applied to obtain pixels from both of the user's hands. Next, Lee et al. [8] presented a developed algorithm that used an RGB colour frame and converted it to a binary frame using Otsu's global threshold. After that, a depth range was selected for hand segmentation, and then the two methods were aligned. Finally, the k nearest neighbour (kNN) algorithm was used with Euclidian distance for finger classification. Another study by Ma et al. [9] proposed a wireless interaction system for a robot through translating hand gesture information into commands, where a slot algorithm was utilised to identify finger gestures. Marin et al. [10] used two techniques together to detect finger regions such as leap motion and Kinect devices to extract different feature sets. The system accuracy was increased by combining the two device features, where the leap motion provides high-level data information but lower reliability than the Kinect sensor, which provides a full depth map. In a study by Bakar et al. [11], the segmentation used 3D depth data selected based on a threshold range. Bakar et al. [12] used fingertips selected using depth threshold and the K-curvature algorithm based on depth data. In Karbasi et al. [13], the hand was segmented based on depth information using a distance method and background subtraction method. Iterative techniques were applied to remove the depth image shadow and decrease noise. A study by Kim et al. [14] proposed a new method based on a near depth range of fewer than 0.5 m where skeletal data was not provided by the Kinect. This method was implemented using two image frames: depth and infrared. Next, Graham's scan algorithm was used to detect the convex hulls of the hand in order to merge with the result of the contour tracing algorithm to detect the fingertips. In a study by Pal et al. [15], the skin–motion detection technique was used to detect the hand, and then Hu moments were applied for feature extraction, after which HMM was used for gesture recognition. Another study by Desai et al. [16] proposed a home automation system for facility control by senior citizens who face disabilities, using a computer vision system based on a Kinect sensor. Desai et al. [17] introduced an algorithm based on an RGB colour and Otsu's global threshold. After that, a depth range was selected for hand segmentation, and then the two methods were aligned. Finally, the kNN algorithm was used with Euclidian distance for finger classification. Another study by Xi et al. [18] used a skeleton tracking method to capture the hand and locate fingertips, where a Kalman filter was used to record the motion of the tracked joint. The cascade extraction technique was used with a novel recursive connected component algorithm. Another study by Li et al. [19] presented a developed system to combine depth information and skeletal data, facing the challenge of complex background and illumination variation, rotation invariance, in which some constraints were set in hand segmentation. Another study by Ma et al. [5] improved depth threshold segmentation by combining depth and colour information using the hierarchical scan method, and then hand segmentation was used based on the local neighbour method. This approach gave results over a range of up to 2 m. Bamwenda et al. [20] used depth information with skeletal and colour data to detect the hand. The segmented hand was then matched with the dataset using a support vector machine (SVM) and artificial neural networks (ANN) for recognition. The authors concluded that ANN was more accurate than SVM. Extensive review on this subject can be found in [21].

3. Materials and Methods

3.1. Participants and Experimental Setup

This study was investigated with three different experiments, where each experiment evaluated with the same group of elderly participants, including two males and one female

with different ages 65 to 75 with one adult aged 35 years. This study adhered to the Declaration of Helsinki ethical principles (Finland 1964) where written informed consent forms were obtained from all participants after a full explanation of the experimental procedures. All participants trained individually according to every proposed scenario. All scenarios were tested indoors, and it took a half-hour for every participant where the Kinect sensor set up at a fixed distance on each experiment from (0.5–4.5 m). Figure 1 shows the proposed system experimental setup.

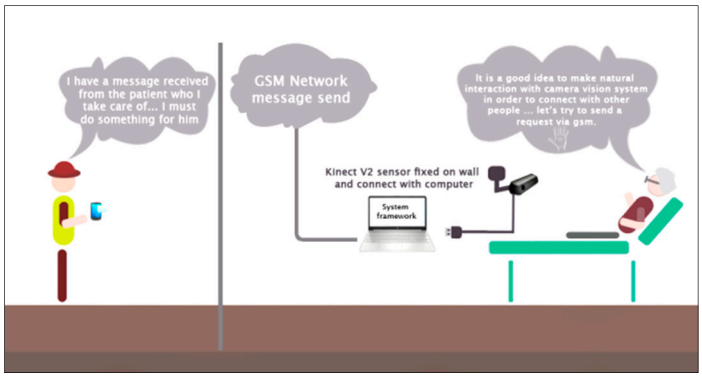


Figure 1. Experimental setup of the proposed method.

3.2. Hardware

Figure 2 represents the design of the practical circuit that utilised in each experiment, which consisted of a Kinect V2 depth sensor, DC-DC chopper (buck), Arduino microcontroller type-Nano and GSM module Sim800L.

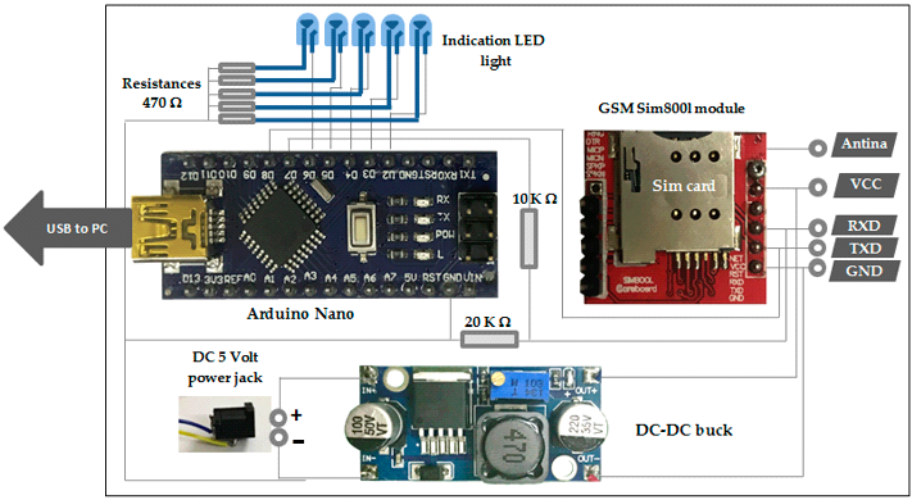


Figure 2. Practical circuit of the system.

3.2.1. Microsoft Kinect Sensor

The Kinect V2 sensor, shown in Figure 3, was released by Microsoft in 2014. It is considered an enhanced version of the Kinect V1 model. In this study, the Kinect V2 sensor was utilised because it offers high-resolution image capture for RGB and depth to provide body joints information. Moreover, it has enhanced specifications compared with the older version. The most important features of the Kinect sensor V2 are listed in Table 2. More detail can be found in [22–28].

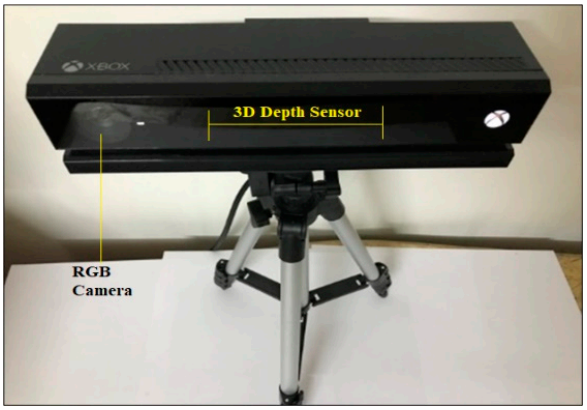


Figure 3. Microsoft Kinect sensor v2 for Xbox.

Table 2. A list of Kinect sensor V2 specifications.

Feature	Description
Body tracking	Up to 6 persons
Joint detection	Up to 25 joint per person
Depth sensing	512 × 424 resolution 30 Hz
Active infrared	3 IR emitter
Colour camera	1920 × 1080 resolution 30 Hz
Depth range	0.5 m to 4.5 m
Field of view	70-horizontal and 60-vertical
Microphone array	Four microphone sensors linearly aligned

3.2.2. Arduino Nano Microcontroller

An Arduino-Nano type microcontroller was the heart of the proposed system, where it received a command from computer via serial port and controlled the GSM module. It had suitable specifications such as small size with a clock frequency 16 MHz [29]. The Nano connected with a GSM-module via a transmitter and receiver through two digital pins and with the computer via a mini-B USB cable. The microcontroller task was to receive data from MATLAB 2019 and control on the GSM-module to send proper messages according to the type of hand gesture performed by participants.

3.2.3. GSM Module Sim800L

The GSM-module Sim800L was utilised in the practical circuit of the proposed system because it has a small size and can be used for making calls, sending messages and give GPRS data. The module transmitter and receiver pins connect with microcontroller via two digital pins. The module feed with suitable voltage level (3.7 Volt) through connecting Vcc and GND with a DC-buck chopper LM2596 [30,31] because the Arduino digital pin provides 40 mAmp which is not sufficient for GSM proper function [30].

3.2.4. DC-DC Chopper (Buck)

A DC-to-DC step-down converter was used. The simplest way to reduce the voltage of a DC supply is to use a linear regulator (such as a 7805) yet linear regulators waste energy as they operate by dissipating excess power as heat. Buck converters, on the other hand, can be remarkably efficient (95% or higher for integrated circuits). It utilises a MOSFET switch (IRFP250N), a diode, an inductor and a capacitor. Some resistors are also used in the circuit for the protection of the main components. When the MOSFET switch is "ON" current rises through inductor, capacitor and load. The inductor is used to store energy. When the switch is "OFF", the energy in the inductor circulates current through the inductor, capacitor freewheeling diode and load. The output voltage will be less than or equal to the input voltage. In this study, an LM2596 dc-dc buck converter step-down

power module with a high-precision potentiometer for adjusting output voltage was used that is capable of driving a load up to 3A with high efficiency.

3.3. Software

In this study, the following software and tools have been used:

1. MATLAB R2019a (Image processing toolbox, Computer vision toolbox, Deep learning toolbox).
2. Microsoft standard development kit (SDK) for Kinect V2.
3. Kinect for Windows Runtime V2.
4. Arduino program (IDE).

3.4. Methods

3.4.1. The First Scenario: Hand Detection Using Depth Threshold and Depth Metadata

The Kinect V2 sensor provides depth information and skeleton data for up to six human bodies at once. A threshold-based segmentation to the depth frame using the z-axis was adopted in order to extract the hand mask. The resulting image was then smoothed by using a median filter [20]. The filtered image was combined with the cropped hand based on joint tracking to improve the result of hand segmentation. The diagram that describes the process for the first scenario is shown in Figure 4.

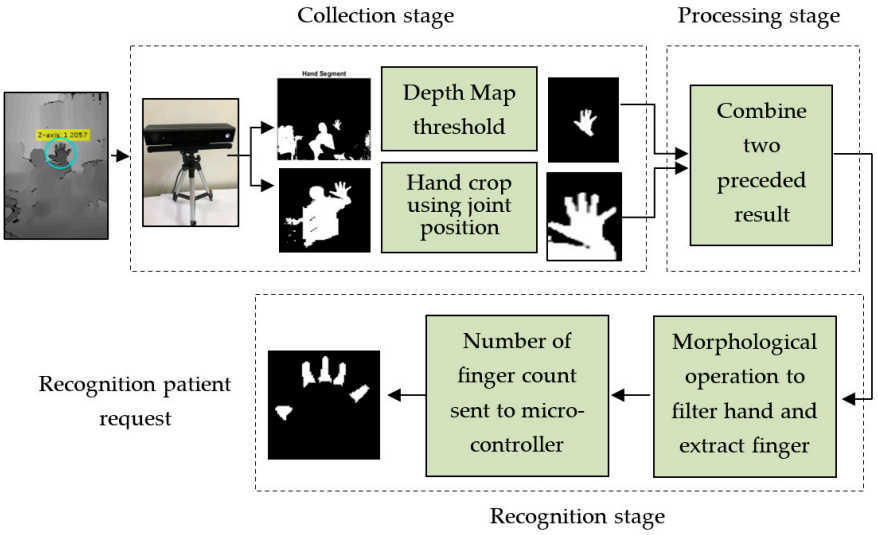


Figure 4. The proposed method based on depth threshold and depth metadata.

The steps illustrated in Figure 4 can be summarised as follows:

- After acquiring the depth frame from the Kinect depth sensor, it can be easy to locate the centre of the palm of the hand from depth metadata using the joint position property. This point is mapped onto the depth map, and their depth values are saved for the next step.
- As every skeleton point in 3D space is associated with a position and an orientation, we can obtain the position of the central palm in real-time.
- The depth metadata returned by the depth sensor gave body tracking data so that the body index frame property enabled segmentation of the full human body into six bodies.
- After segmenting the body, a rectangular region was selected (for example, with size 200 × 200) around the central point of the hand/palm in the depth images. Initial segmentation was conducted based on the hand crop using the tracking point of the central palm. Because the right hand conforms more to the habit of human-computer interaction, we chose the right hand as the identification target.

- The depth threshold was provided for the depth map and the hand segment using a *z*-axis threshold.
- The hand cropped result was combined with the depth threshold result to improve the outcome.
- The binary image was smoothed using a median filter, and we set 5 as the linear aperture size.
- Using some morphological operations, such as erosion and dilation and image subtraction to extract the palm by drawing a circle covering the whole area of the palm using a tracked joint of the central palm. The fingers were then segmented, where the number of fingers counted appear as a white area and were then connected with a specific request.
- Finally, five fingers carried out five requests according to finger count that was sent by the microcontroller as a numeric value via the serial port to control the GSM module.

3.4.2. The Second Scenario: Hand Detection and Tracking Using Kinect V2 Embedded System

The Kinect V2 depth sensor has one specific property associated with body tracking, where the depth sensor collects body metadata by turning on the body tracking property, while the metadata provides the parameters of the body data as listed in Table 3.

Table 3. The metadata fields related to tracking the bodies.

No.	Parameters of the Body Data Obtained by the Depth Sensor	Struct Array
1	IsBodyTracked	[1 × 6 logical]
2	BodyTrackingID	[1 × 6 double]
3	BodyIndexFrame	[424 × 512 double]
4	ColorJointIndices	[25 × 2 × 6 double]
5	DepthJointIndices	[25 × 2 × 6 double]
6	HandLeftState	[1 × 6 double]
7	HandRightState	[1 × 6 double]
8	HandLeftConfidence	[1 × 6 double]
9	HandRightConfidence	[1 × 6 double]
10	JointTrackingStates	[25 × 6 double]
11	JointPositions	[25 × 3 × 6 double]

Using the "get data" property provided by depth sensor, we can easily access to body tracking data as metadata on the depth stream. The function returns frames of size 512 × 424 in mono 13 formats and uint16 data type. We look at the metadata to see the parameters in the body data which bring eleven different properties; these metadata fields are related to tracking the bodies as listed in Table 3.

The Kinect depth sensor provides metadata parameters such as the left-hand state and right-hand state which is a 1 × 6 double array that identifies possible states for both the left and right hands of the tracked bodies. Where the values returned by the depth sensor include information on the body hands state as the following:

0 = unknown (indicate the body not tracked)
1 = not tracked (indicate the detected body but not tracked)
2 = open (indicate the hand fingers extended all)
3 = closed (indicate the hand fingers collapsed all)
4 = lasso (indicate the hand index finger is extended)

In this scenario, the metadata parameters were encoded for three different gestures performed by the right hands and two gestures performed by the left hands in order to

represent five different requests and sent via GSM. The requests represented by the right hand are open hand, closed hand and lasso gestures, which indicate "Water", "Meal", "Toilet", respectively. Whereas the remaining two requests represented by the left hand using (open hand and closed hand) that indicate "Help" and "Medicine", respectively. This experiment used both hands to implement five different gestures, where every gesture indicates a specific request as a reverse of the first experiment that used only one hand to perform these five requests.

3.4.3. The Third Scenario: Hand Gestures Based on SCNN and Depth Metadata

In this scenario, the experiment was conducted using a deep learning classifier based on a simple convolutional neural network (SCNN). CNN is a suitable tool for building an image recognition system.

The hand image samples were captured by an automatic program created by the author, where the image data was resized and stored in one folder to separate into different categories related to five gestures manually. These categories were named image data-store. The image data-store in this folder category was labelled based on folders' names with storage of the image as an object. The images data-store can store a large amount of image data and efficiently read a batch of images while training the CNN.

The data store includes 1000 images for every category of hand gestures from 1–5 and a total of 5000 images for all categories. The number of classes was specified at the last fully connected layer in the output of the network. Additionally, the input image size was specified at the input layer. Each image must be stored as 28-by-28-by-1 pixels. Figure 5 shows five hand gestures used in this experiment, where the dataset categories were created by the authors using the Kinect depth sensor.

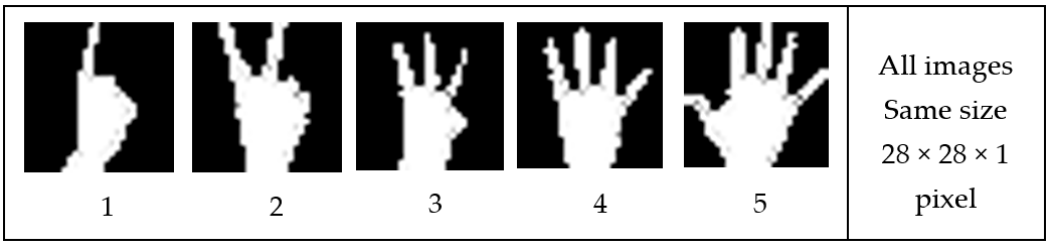


Figure 5. Five gestures created and stored for training and testing.

The image dataset was separated into training and validation data-sets, where the training-set includes 70 images and the remaining images for a validation-set. Each label splits the data store into two new data stores, training hand gestures data and validation hand gestures data.

- Specify Training and Validation Sets

The image dataset is separated into training and validation data-sets, where the training set includes 700 images and the remaining images for the validation set. Each label splits the data store hand gestures data into two new data stores; train hand gestures data and validation hand gestures data.

- Define Network Architecture

The architecture of CNN can be defined as follows:

1. Input Layer Image

At the first layer of the network, the size of the input image was specified by 28-by-28-by-1, which indicates the height, width and channel size, respectively. The channel size is 1 related to the binary image processed. Moreover, the trained network shuffles the image data at the beginning of the training process and for every epoch while it trains.

2. Convolutional Layer

At the convolutional layer, the filter was used to make a scan along with the image at the training function to extract features. In this experiment, the filter size was specified to be 3-by-3 high and wide, respectively which can specify different sizes for the filter used. The number of filters indicated the number of neurons that have the same connection point at the input. The number and size of the filter play an important role in determining the number of feature map extracted.

3. Batch Normalisation Layer

Batch normalisation layers enhance the activations and gradients propagating in the network, where the network is easy to train. To increase the speed of network training, the Batch normalization layers were used between convolutional layers and ReLU layers.

4. ReLU Layer

The nonlinear activation function is located after the batch normalisation layer. The most common activation function was used which is the rectified linear unit (ReLU).

5. Max Pooling Layer

The function of the max-pooling-layer was used for downsampling operation which was used to decrease the spatial size of the feature map and also eliminate the redundant-spatial-information. The benefits of downsampling are to increase the number of filters in the deeper layers of the convolutional network while maintaining computation per layer. The max-pooling layer is often placed after convolutional-layers and gives the max value of the rectangular region of the input. In this experiment, the rectangular region size was [2, 2].

6. Fully Connected Layer

The fully connected layer is preceded by the convolution layer and down-sampling layer. It is fully connected with all neurons in the preceded layers and works to merge all the learned-features by the preceded layers into the image to introduce the biggest pattern. In the last fully connected layer, all features are merged to classify the images. The network output size is equal to the number of classes, where the output size is 5 with regard to five classes.

7. Softmax Layer

The softmax activation function is responsible for printing the output of the fully connected layer which preceded it. Where the softmax-layer includes positive-numbers in which the sum of these numbers is equal to one. This number is used for classification probability.

8. Classification Layer

The last network layer is the classification layer. Its output value takes the softmax activation function for each input to match the input with one of the matching classes and compute the error.

- Specify Training Options

To specify the training based on a CNN structure build, this step needs to determine the training parameters, where the network trained using stochastic gradient descent with momentum (SGDM) with a learning rate initially of 0.01 and max-epoch number 4. The epoch is the full training cycle for the input training dataset.

- Train Network Using Training Data

The network was trained using the GPU by default. Otherwise, it would use only the CPU. Figure 6 shows the deep-learning-training-progress and plots the mini-batch-loss (cross-entropy loss), the validation loss and accuracy (percentage of images classified by the network correctly).

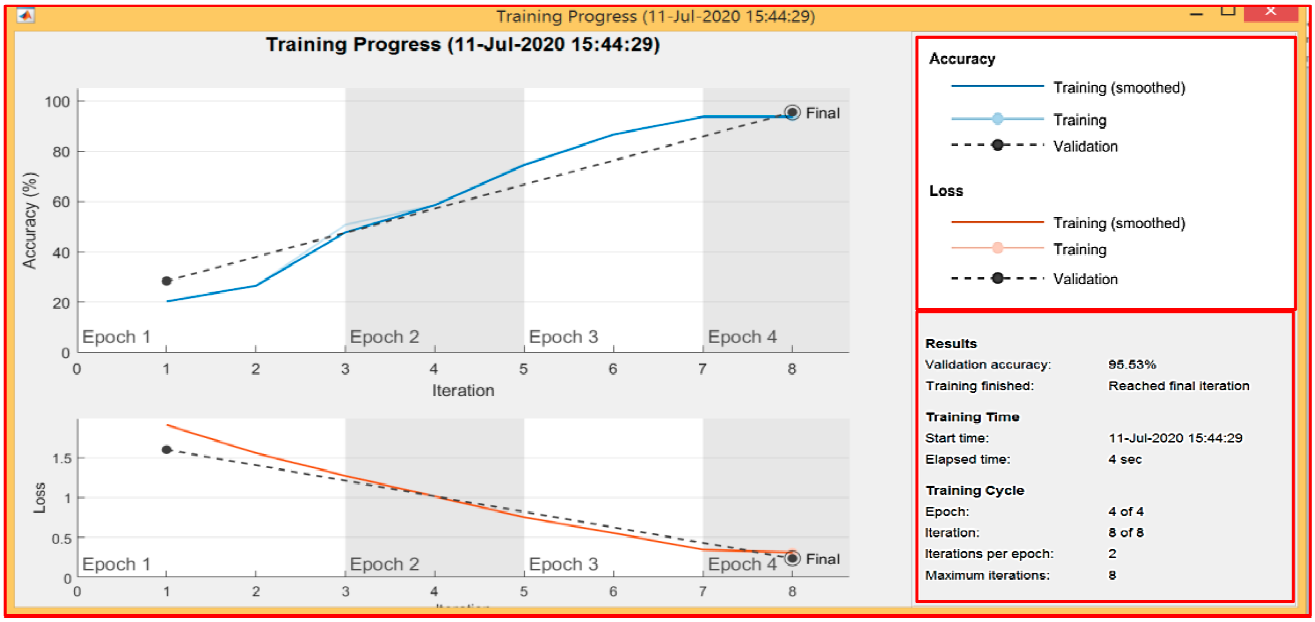


Figure 6. Training progress of neural network for 4 epochs.

4. Experimental Results and Discussion

4.1. Experimental Results

For the 1st scenario, the hand detection method based on depth threshold and depth metadata was used. The experimental results for the first scenario are shown in Figure 7 at which shows five different gestures based on finger counting.

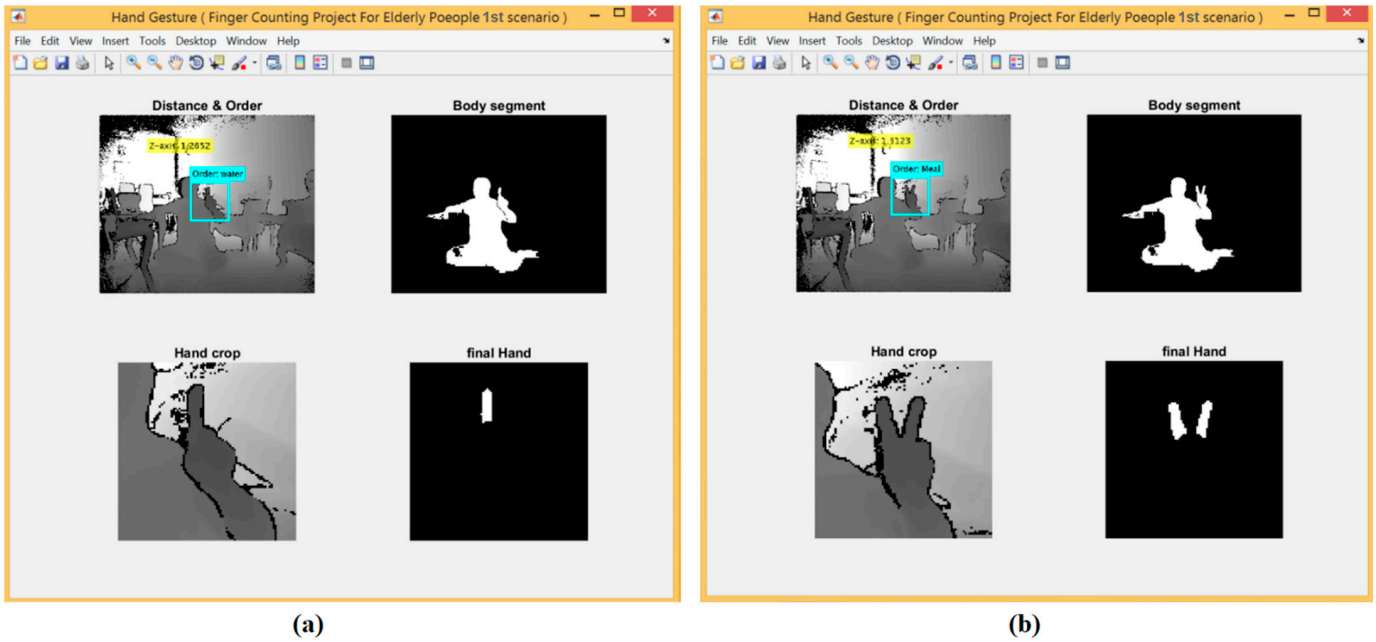


Figure 7. *Cont.*

Figure 7. Finger count interpreted as patient requests (**a**–**e**).

Table 4 shows the experimental results for all participants with every single gesture. The results were recorded for all participants and we took the mean of these recorded results. The recognition rate for the overall gestures was 83.07% at detection distance between 1.2–1.5 m.

Table 4. The results analysis for the total number of tested gestures for each participant (First scenario).

Hand Gesture Type	Total Number of the Sample per Tested Gesture	Number of Recognised Gestures	Number of Unrecognised Gestures	Percentage of Correct Recognition for Total Number of Sample Gesture %	Percentage of Fault Recognition for Total Number of Sample Gesture %
0	65	65	0	100.00	0
1	65	55	10	84.62	15.38
2	65	52	13	80.00	20.00
3	65	50	15	76.92	23.08
4	65	49	16	75.38	24.62
5	65	53	12	81.54	18.6
Total	390	324	66	83.07%	16.94%

The confusion matrix was adopted to analyse the results of Table 4, which provide predicted and actual results for all tested gestures. Figure 8 shows the results of the confusion matrix and summaries of the predicted results and actual results in the form of row and column.

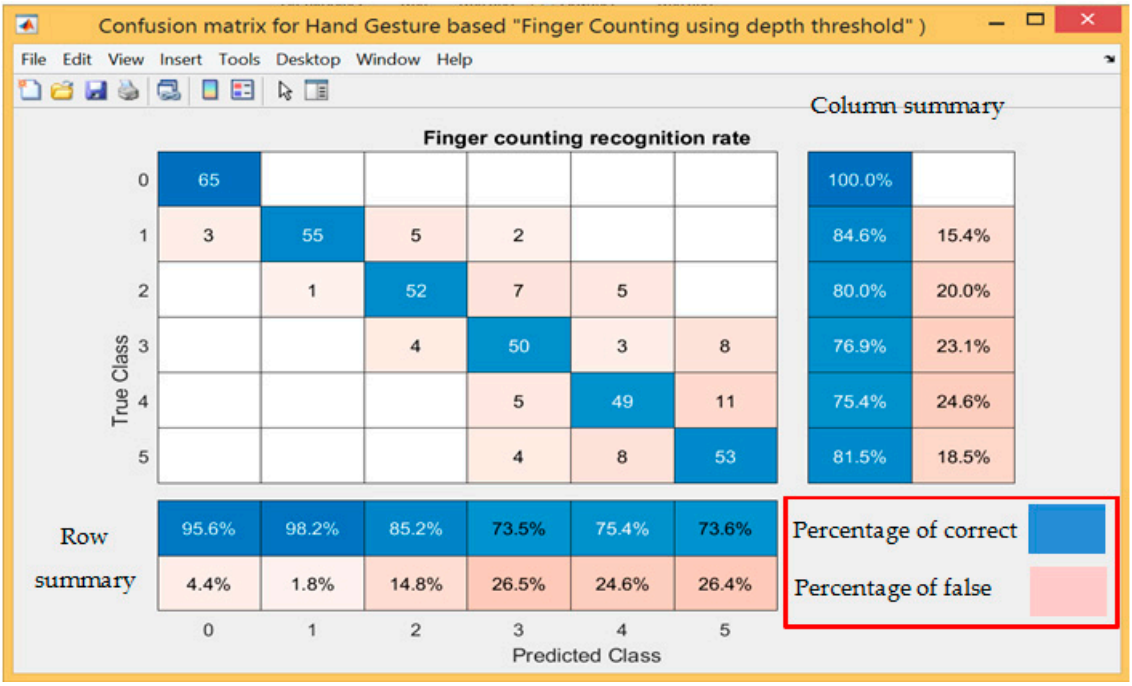
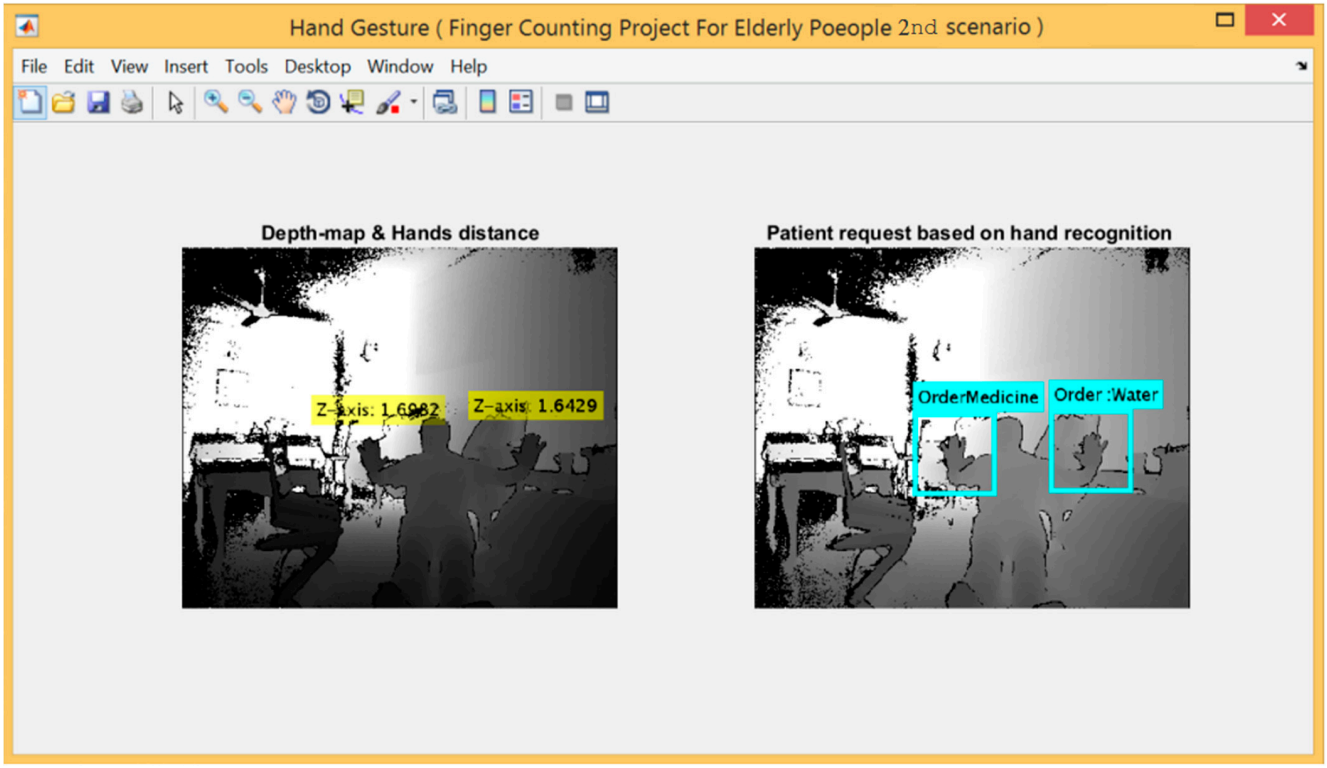
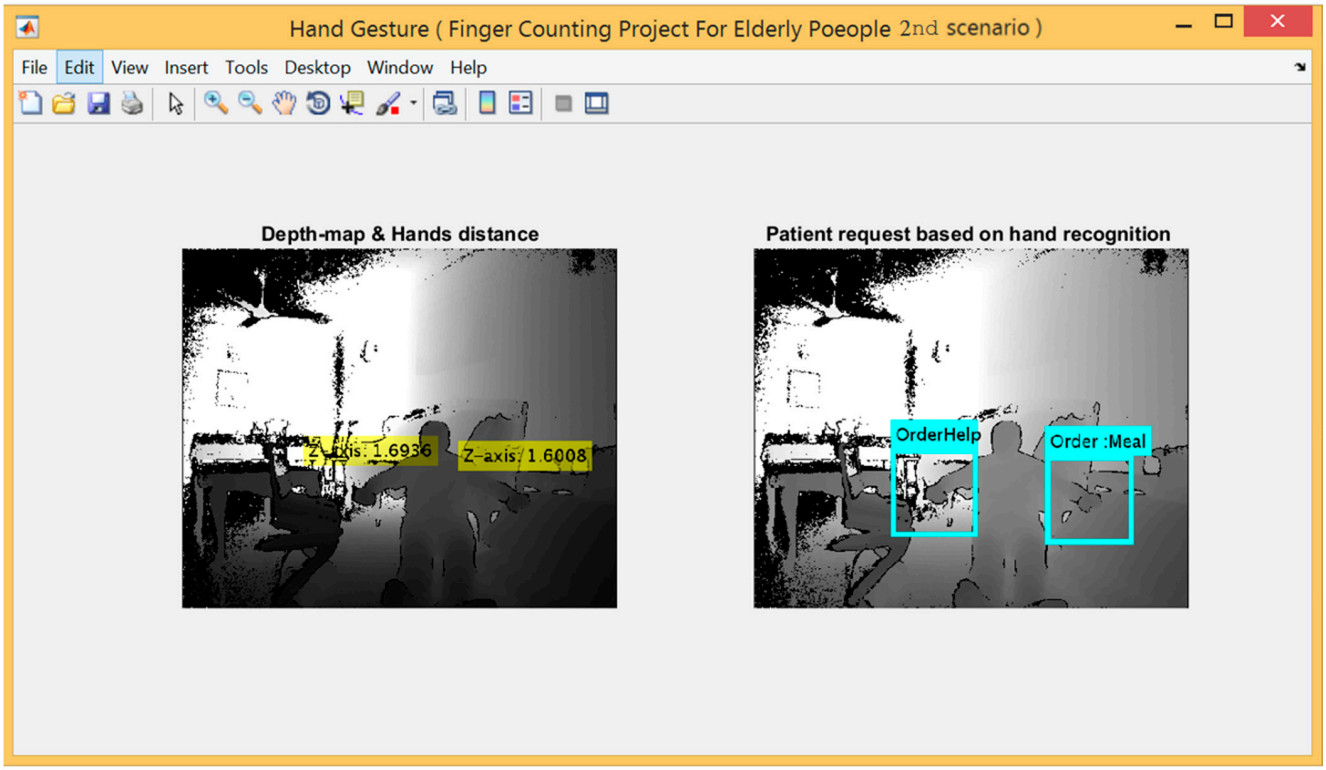


Figure 8. Confusion matrixes for the first scenario.

For the 2nd scenario, the hand detection method using Kinect V2 embedded system was used. Figure 9 shows five gestures provided by the left and right hands, whereas Figure 10 shows the detection range between 0.5~4.5 m for applying this scenario.

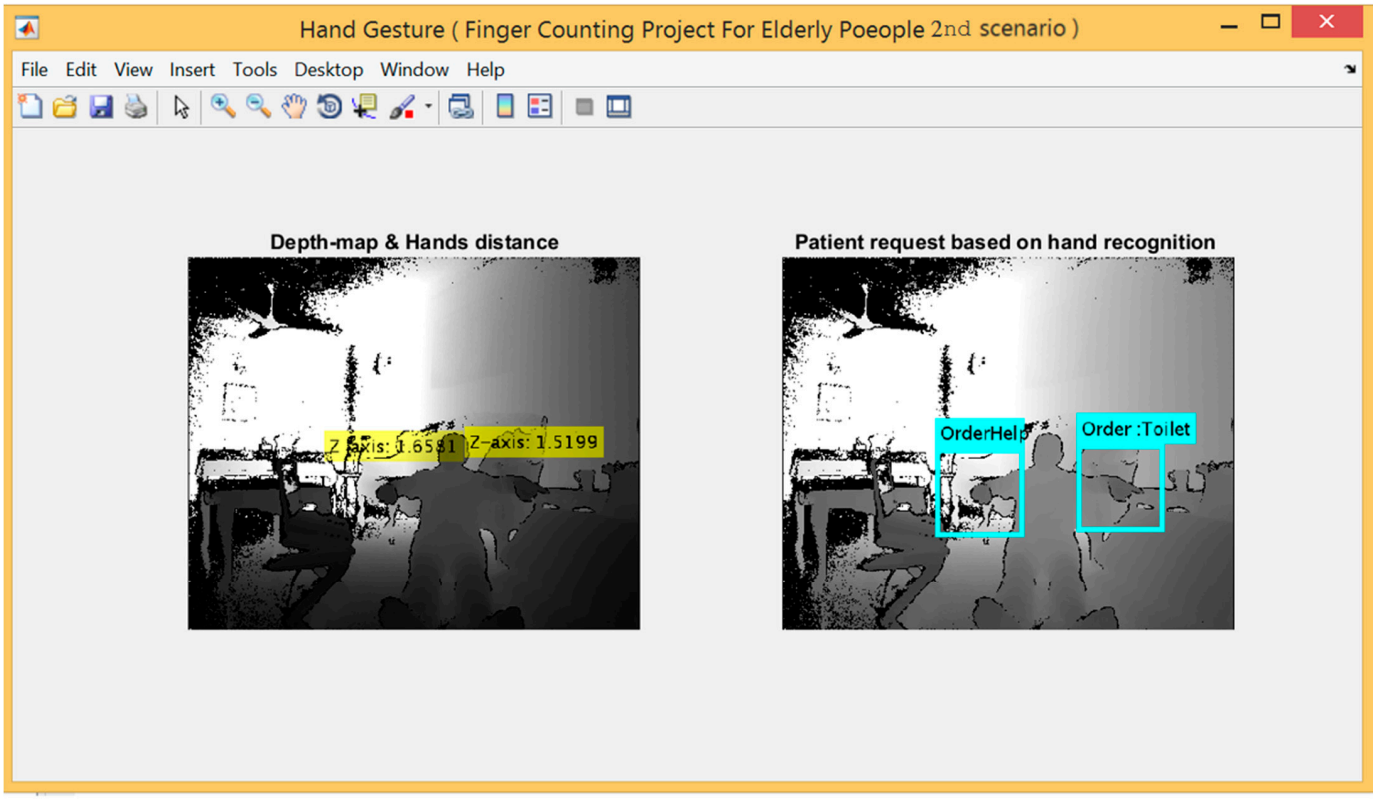


(a)



(b)

Figure 9. *Cont.*



(c)

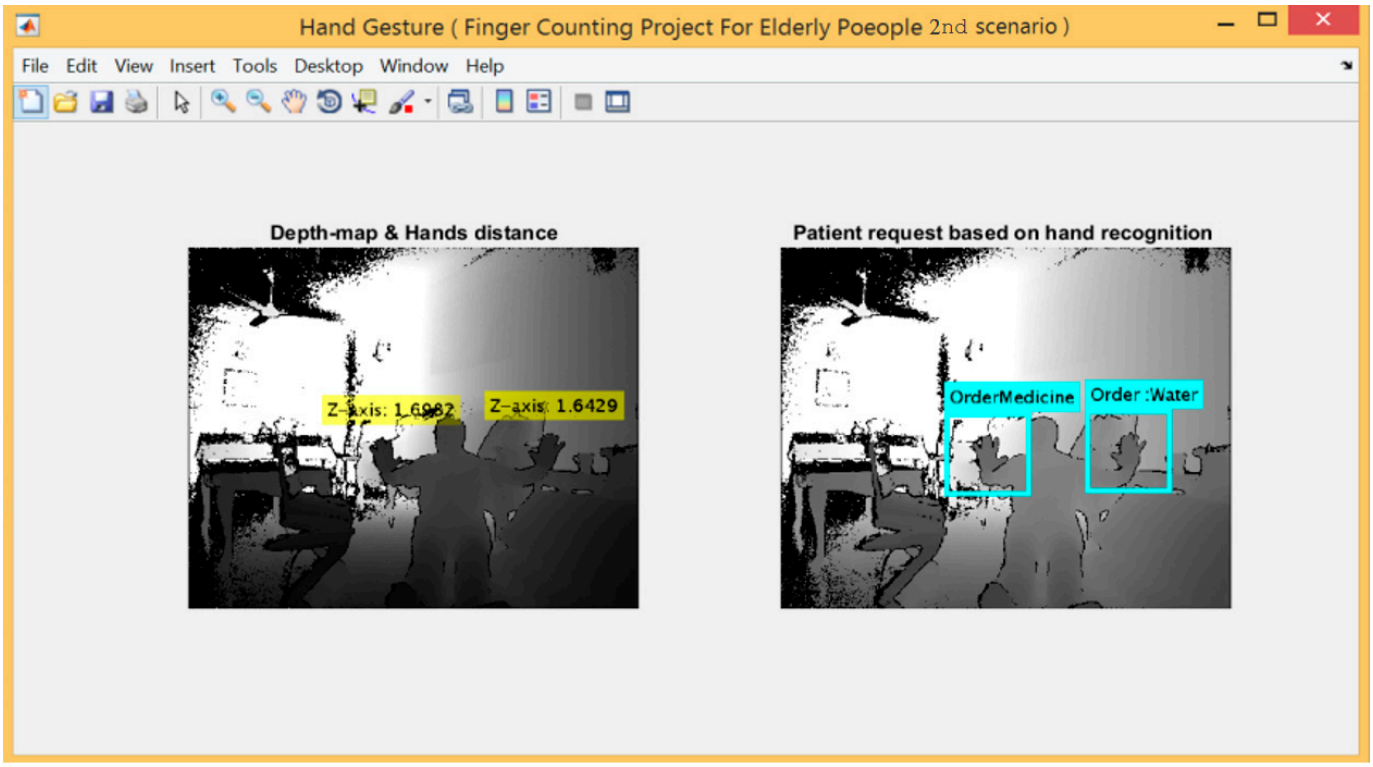
Figure 9. The results of the proposed method for the second scenario for both hands (**a**–**c**).

Table 5 shows the experimental results for all participants regarding every single gesture performed by both hands together. The recognition rate for the overall gestures in this scenario was 95.2% at flexible detection distance between 0.5~4.5 m.

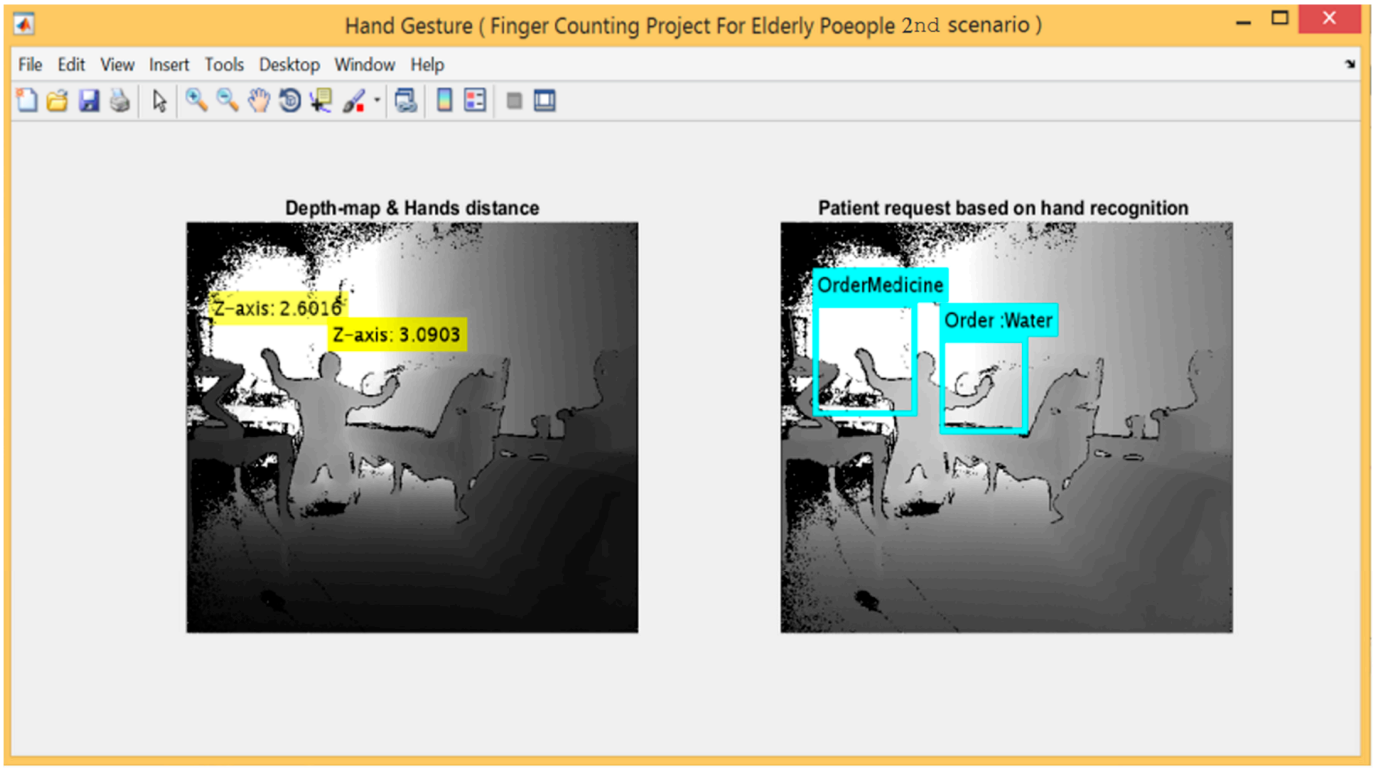
Table 5. The results analysis for the total number of tested gestures for each participant (second scenario).

Hand Gesture Type	Total Number of Sample per Tested Gesture	Number of Recognised Gestures	Number of Unrecognised Gestures	Percentage of Correct Recognition for Total Number of Sample Gesture %	Percentage of Fault Recognition for Total Number of Sample Gesture %
1	50	47	3	94.00	6.00
2	50	48	2	96.00	4.00
3	50	47	3	94.00	6.00
4	50	48	2	96.00	4.00
5	50	48	2	96.00	4.00
Total	250	238	12	95.2%	4.8%

The confusion matrix was adopted so as to analyse results of Table 5, which provides the predicted and actual results for all tested gestures. Figure 11 shows the result of the confusion matrix and summarises the predicted and actual results in the form of row and column.

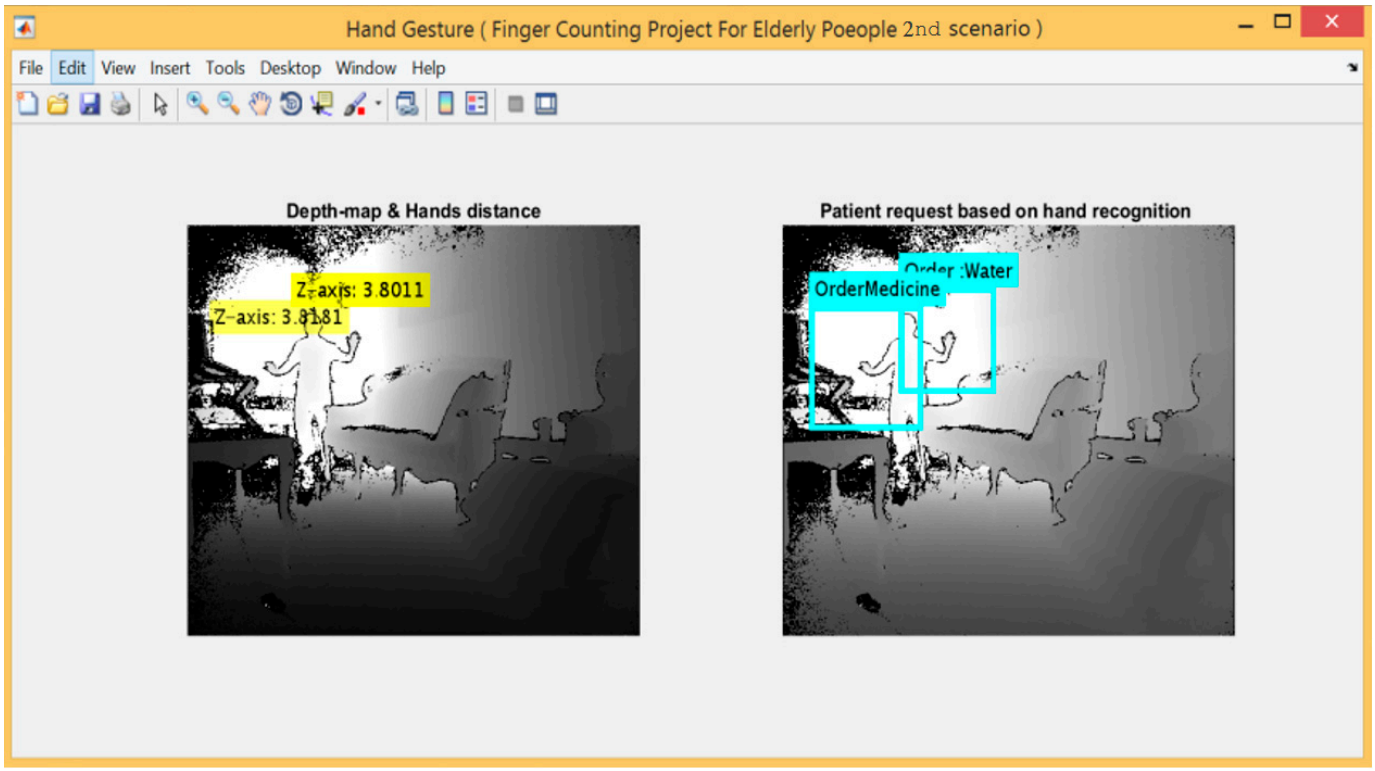


(a)

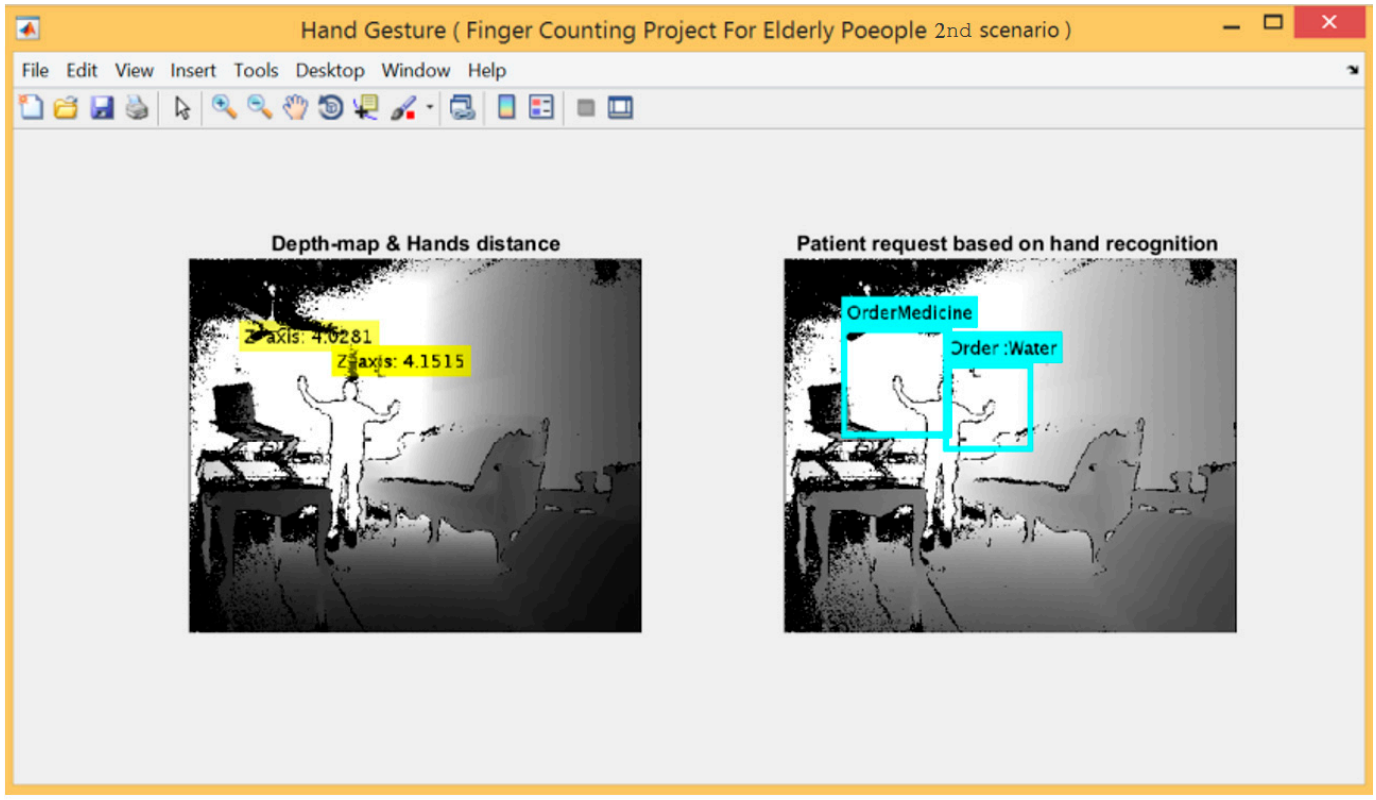


(b)

Figure 10. *Cont.*



(c)



(d)

Figure 10. The test of the detection range for the second scenario (**a**–**d**).

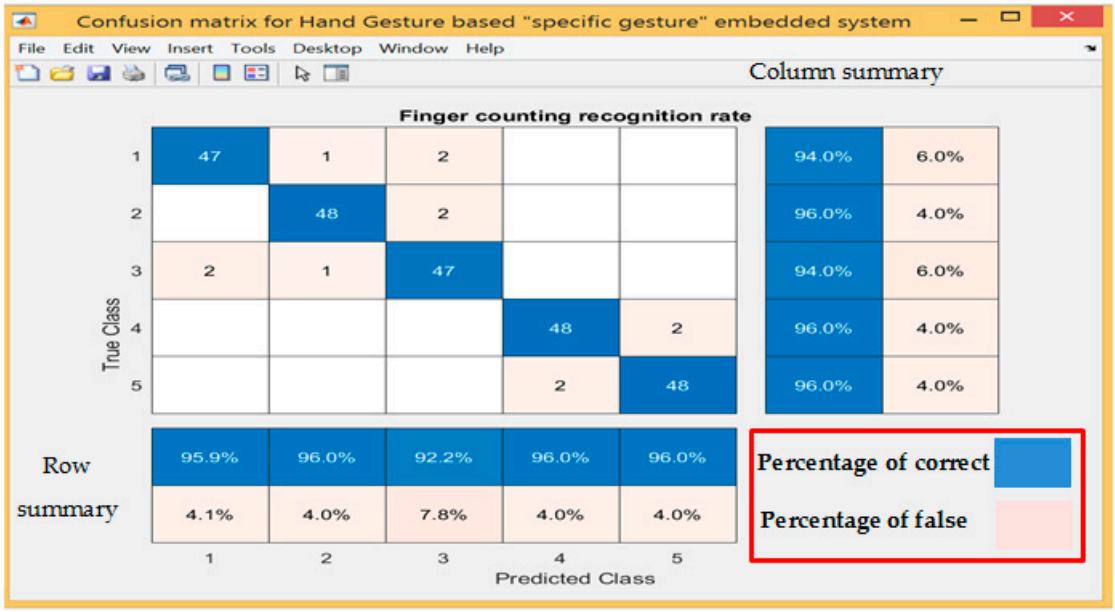


Figure 11. Confusion matrixes for the second scenario.

For the 3rd scenario, the hand detection method based on SCNN and depth metadata was used. Figure 12 shows five gestures provided by the left and right hands.

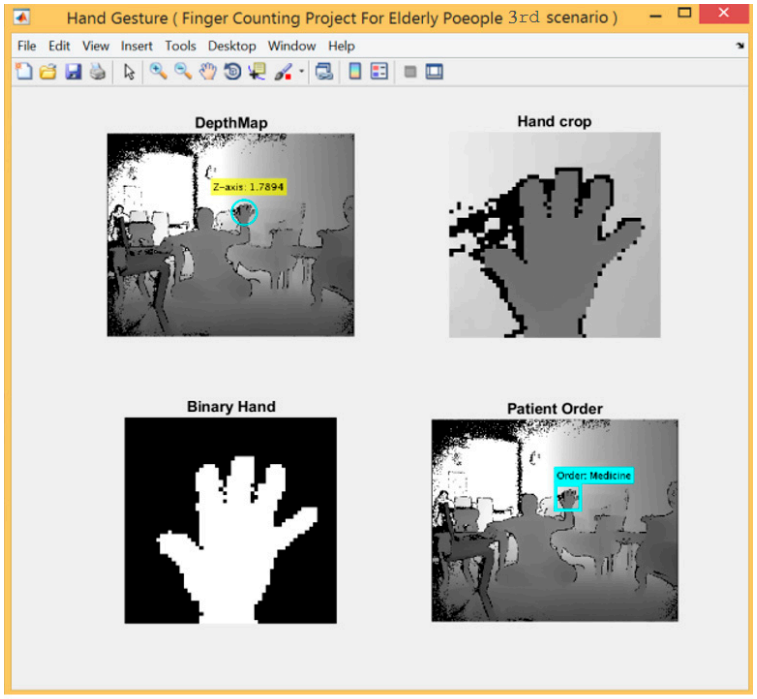
Table 6 shows the experimental results for all participants regarding every single gesture performed by both hands together. The recognition rate for the overall gestures in this scenario was 95.53 % at detection distance between 1.5~1.7 m.

Table 6. The results analysis for the total number of tested gestures for each participant (third scenario).

Hand Gesture Type	Total Number of Sample per Tested Gesture	Number of Recognised Gestures	Number of Unrecognised Gestures	Percentage of Correct Recognition for Total Number of Sample Gesture %	Percentage of Fault Recognition for Total Number of Sample Gesture %
1	65	64	1	98.46	1.54
2	73	73	0	100.00	0.00
3	49	49	0	100.00	0.00
4	65	53	12	81.54	18.46
5	61	60	1	98.36	1.6
Total	313	299	14	95.53%	4.47%

The confusion matrix was adopted so as to analyse the results of Table 6, which gives the predicted and actual results for all tested gestures. Figure 13 shows the result of the confusion matrix and summarises the predicted and actual results in the form of rows and columns.

Figure 12. *Cont.*



(e)

Figure 12. Fingers count interpreted at different participant requests using a deep learning method (**a**–**e**).

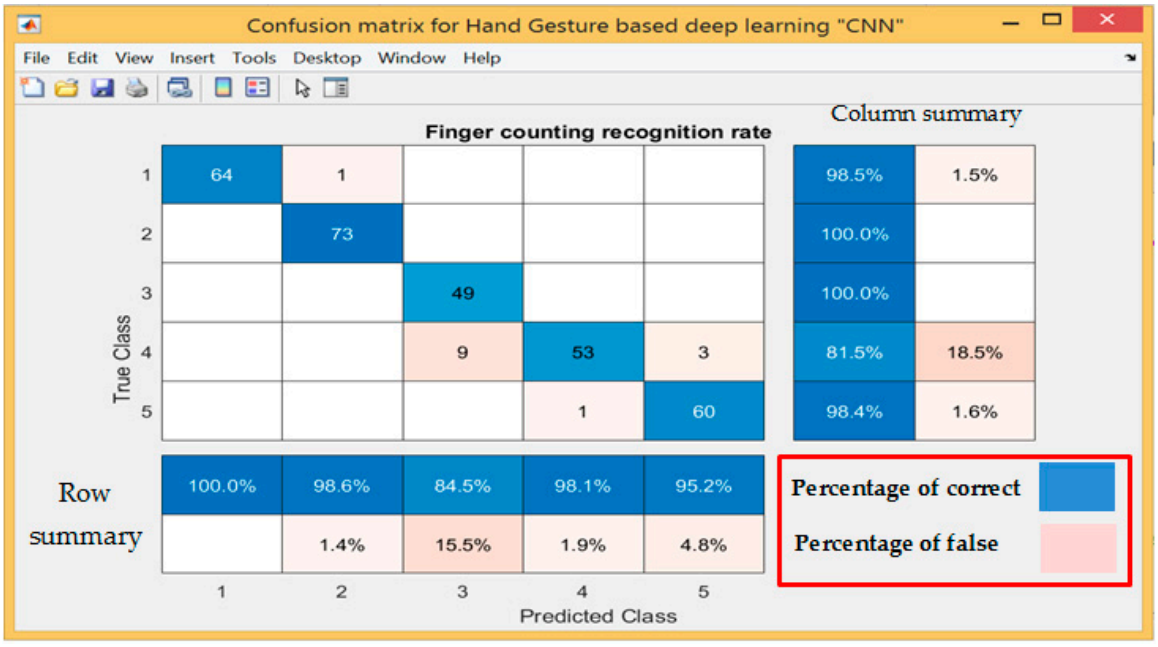


Figure 13. Confusion matrixes for the third scenario.

4.2. Discussion

A comparison of three scenarios results were discussed in this section. The three different hand gestures recognition scenarios were conducted using the Microsoft Kinect V2 sensor. These scenarios can be categorised into three main approaches: Finger counting, the embedded system provided by Kinect V2 and deep learning based on a simple CNN.

In this section, the key points for these three categories are compared and summarised in Table 7.

Table 7. The key points of every approach for three scenarios and their performance.

Method	Type of Gesture	Principle	Classification	Image Pixel	Recognition Rate	Distance from the Camera
Scenario 1	Finger count (0–5) Single hand	Depth threshold and skeleton joint tracking using metadata information	Appearance of white area	512 × 424	83.07%	1.2~1.5 m
Scenario 2	Specific gesture both hand	Metadata parameter	Hand left state, hand right state parameter by Kinect depth	512 × 424	95.2%	0.5~4.5 m
Scenario 3	Finger count image features (1–5) Single hand	SCNN Depth metadata	CNN	Dataset 28 × 28 × 1	95.53%	1.5~1.7 m

From Table 7, it can easily be observed which is the best approach with regard to recognition rate, distance from the camera and ease to perform hand gestures.

However, taking consideration of some challenges facing every category can be illustrated as follows:

- The first scenario offers acceptable results, but has limitations in regard to classification, where the number of fingers recognised is based on the apparent white area and results are affected by any white speckle.
- The second scenario provides a high recognition rate because it offers better flexibility in regard to distance during capturing the gestures in real-time if compared with other categories. However, the only type of gestures that can be read are three active gestures for every hand (from the default of the embedded system provided by the Kinect) and five hand gestures must be performed by both hands using three gestures for each hand, respectively.
- The third scenario provides a good recognition rate but suffered due to the distance limitation related to the range sensor used when the dataset was created.

4.3. Comparison Result with Related Work

The main goal of this paper was to investigate the natural interaction system performed by hand gestures with the use of camera imaging-based technologies at real-time interaction to control messages sent via the GSM module. The goal was motivated by the challenges associated with current monitoring systems under different assumptions, including the distance from the camera, recognition rate, and real-time interaction. Table 8 summarises and compares the research results with the closest related work.

Table 8. Comparison research results with the related work.

Ref	Camera	Number of Gesture	Principle	Classification Algorithm	Issues	Recognition Rate	Distance from the Camera
Ganokr-atanaa et al. [32] 2017	RGB camera	6 gestures	optical flow and blob analysis	blob analysis technique	error pre-processing stage because shadow under the hand	good results	Not mentioned
Norah et al. [33] 2019	Mobile Camera	7 gestures	CNN	CNN	backgrounds, illumination	accuracy is 99%	short distances
This paper	Depth camera	5 gestures	Depth threshold	Connected component	White speckle	83.07%	1.2~1.5 m
			embedded	- -	Depth range	95.2%	0.5~4.5 m
			CNN	CNN	Depth range	95.53%	1.5~1.7 m

The comparison results can be summarised as follow:

- The two proposed methods presented in the first and second row by [112, 88] cannot use a dim environment because they use RGB and mobile cameras and effected by lightning conditions while this paper proposed three methods that can be used in a dim environment.
- The two proposed methods presented in the first and second row by [112, 88] can be used only at the short distance while this paper proposed three different methods with flexible distance.
- The two proposed methods presented in the first and second row by [112, 88] carried out only hand gesture recognition while this thesis proposed three hand gestures recognition methods with a practical circuit that send text message according to these gestures.

5. Conclusions

In conclusion, this study explored the feasibility of extracting hand gestures in real-time using the Microsoft Kinect V2 sensor under three scenarios: finger counting, the embedded system provided by the Kinect itself, and deep learning based on CNN. The proposed methods used the same practical circuit for each scenario, which reports that the correct SMS message sent to the care provider smartphone correlated directly with the results and accuracy of the recognition system. The experimental evaluation of the proposed methods has been conducted in real-time for all participants under three different scenarios. The experimental results were recorded and analysed using a confusion matrix which gave acceptable outcomes making this study a promising method for future home assisting care applications.

Author Contributions: Conceptualization, A.A.-N. and M.O.; methodology, M.O. and A.A.-N.; investigation, M.O. and A.A.-N.; data curation, M.O.; project administration, A.A.-N. and J.C.; resources, M.O.; software, M.O. and A.A.-N.; supervision, A.A.-N. and J.C.; validation, M.O.; funding acquisition, A.A.-N. and J.C.; writing—original draft preparation, M.O.; writing—review and editing, A.A.-N. and J.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors of this manuscript have no conflict of interest relevant to this work.

References

1. Truelsen, T.; Bonita, R.; Jamrozik, K. Surveillance of stroke: A global perspective. *Int. J. Epidemiol.* **2001**, *30*, S11–S16. [CrossRef] [PubMed]
2. Pansare, J.R.; Gawande, S.H.; Ingle, M. Real-Time Static Hand Gesture Recognition for American Sign Language (ASL) in Complex Background. *J. Signal Inf. Process.* **2012**, *3*, 364–367. [CrossRef]
3. Li, Y. Hand gesture recognition using Kinect. In Proceedings of the 2012 IEEE International Conference on Computer Science and Automation Engineering, Beijing, China, 22–24 June 2012; pp. 196–199.
4. Poupyrev, I. Occluded Gesture Recognition. US Patent 9,778,749B2, 3 October 2017.
5. Ma, X.; Peng, J. Kinect Sensor-Based Long-Distance Hand Gesture Recognition and Fingertip Detection with Depth Information. *J. Sens.* **2018**, *2018*, 1–9. [CrossRef]
6. Ren, Z.; Meng, J.; Yuan, J. Depth camera based hand gesture recognition and its applications in Human-Computer-Interaction. In Proceedings of the 2011 8th International Conference on Information, Communications & Signal Processing, Singapore, 13–16 December 2011; pp. 1–5.
7. Wen, Y.; Hu, C.; Yu, G.; Wang, C. A robust method of detecting hand gestures using depth sensors. In Proceedings of the 2012 IEEE International Workshop on Haptic Audio Visual Environments and Games (HAVE 2012) Proceedings, Munich, Germany, 8–9 October 2012; pp. 72–77.
8. Lee, U.; Tanaka, J. Finger identification and hand gesture recognition techniques for natural user interface. In Proceedings of the 11th Asia Pacific Conference on Computer Human Interaction, Bangalore, India, 24–27 September 2013; pp. 274–279.
9. Ma, B.; Xu, W.; Wang, S. A Robot Control System Based on Gesture Recognition Using Kinect. *TELKOMNIKA Indones. J. Electr. Eng.* **2013**, *11*, 2605–2611. [CrossRef]
10. Marin, G.; Dominio, F.; Zanuttigh, P. Hand gesture recognition with leap motion and kinect devices. In Proceedings of the 2014 IEEE International Conference on Image Processing (ICIP), Paris, France, 27–30 October 2014; pp. 1565–1569.
11. Abu Bakar, M.Z.; Samad, R.; Pebrianti, D.; Aan, N.L.Y. Real-time rotation invariant hand tracking using 3D data. In Proceedings of the 2014 IEEE International Conference on Control System, Computing and Engineering (ICCSCE 2014), Ferringhi, Malaysia, 28–30 November 2014; pp. 490–495.
12. Abu Bakar, M.Z.; Samad, R.; Pebrianti, D.; Mustafa, M.; Abdullah, N.R.H. Finger application using K-Curvature method and Kinect sensor in real-time. In Proceedings of the 2015 International Symposium on Technology Management and Emerging Technologies (ISTMET), Langkawai Island, Malaysia, 25–27 August 2015; pp. 218–222.
13. Karbasi, M.; Bhatti, Z.; Nooralishahi, P.; Shah, A.; Mazloomnezhad, S.M.R. Real-time hands detection in depth image by using distance with Kinect camera. *Int. J. Internet Things* **2015**, *4*, 1–6.
14. Kim, M.-S.; Lee, C.H. Hand Gesture Recognition for Kinect v2 Sensor in the Near Distance Where Depth Data Are Not Provided. *Int. J. Softw. Eng. Its Appl.* **2016**, *10*, 407–418. [CrossRef]
15. Pal, D.H.; Kakade, S.M. Dynamic hand gesture recognition using kinect sensor. In Proceedings of the 2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC), Jalgaon, India, 22–24 December 2016; pp. 448–453.
16. Desai, S.; Desai, A.A. Human Computer Interaction Through Hand Gestures for Home Automation Using Microsoft Kinect. In *Proceedings of International Conference on Communication and Networks. Advances in Intelligent Systems and Computing*; Springer Science and Business Media LLC: Berlin/Heidelberg, Germany, 2017; Volume 508, pp. 19–29.
17. Desai, S. Segmentation and Recognition of Fingers Using Microsoft Kinect. In *Proceedings of International Conference on Communication and Networks. Advances in Intelligent Systems and Computing*; Springer: Singapore; Berlin/Heidelberg, Germany, 2017; Volume 508, pp. 45–53.
18. Xi, C.; Chen, J.; Zhao, C.; Pei, Q.; Liu, L. Real-time Hand Tracking Using Kinect. In Proceedings of the 2nd International Conference on Digital Signal Processing ICDSP, Tokyo Japan, 25–27 February 2018; pp. 37–42. [CrossRef]
19. Li, J.; Wang, J.; Ju, Z. A Novel Hand Gesture Recognition Based on High-Level Features. *Int. J. Humanoid Robot.* **2018**, *15*, 1750022. [CrossRef]
20. Özerdem, M.S.; Bamwenda, J. Recognition of static hand gesture with using ANN and SVM. *DÜMF Mühendislik Dergisi* **2019**, *10*, 561–568. [CrossRef]
21. Oudah, M.; Al-Naji, A.; Chahl, J.S. Hand Gesture Recognition Based on Computer Vision: A Review of Techniques. *J. Imaging* **2020**, *6*, 73. [CrossRef]
22. Samir, M.; Golkar, E.; Rahni, A.A.A. Comparison between the KinectTM V1 and KinectTM V2 for respiratory motion tracking. In Proceedings of the 2015 IEEE International Conference on Signal and Image Processing Applications (ICSIPA), Kuala Lumpur, Malaysia, 19–21 October 2015; pp. 150–155.
23. Kim, C.; Yun, S.; Jung, S.-W.; Won, C.S. Color and Depth Image Correspondence for Kinect v2. In *Lecture Notes in Electrical Engineering*; Springer Science and Business Media LLC: Berlin/Heidelberg, Germany, 2015; pp. 111–116.
24. Yang, L.; Zhang, L.; Dong, H.; Alelaiwi, A.; El Saddik, A. Evaluating and Improving the Depth Accuracy of Kinect for Windows v2. *IEEE Sensors J.* **2015**, *15*, 4275–4285. [CrossRef]
25. Sarbolandi, H.; Lefloch, D.; Kolb, A. Kinect range sensing: Structured-light versus Time-of-Flight Kinect. *Comput. Vis. Image Underst.* **2015**, *139*, 1–20. [CrossRef]

26. Mutto, C.D.; Zanuttigh, P.; Cortelazzo, G.M. *Time-Of-Flight Cameras and Microsoft Kinect (TM)*; Springer Publishing Company Incorporated: Berlin/Heidelberg, Germany, 2012.
27. Al-Naji, A.; Gibson, K.; Lee, S.-H.; Chahl, J.S. Real Time Apnoea Monitoring of Children Using the Microsoft Kinect Sensor: A Pilot Study. *Sensors* **2017**, *17*, 286. [CrossRef] [PubMed]
28. Al-Naji, A.; Chahl, J.S. Detection of Cardiopulmonary Activity and Related Abnormal Events Using Microsoft Kinect Sensor. *Sensors* **2018**, *18*, 920. [CrossRef] [PubMed]
29. Sudhan, R.; Kumar, M.; Prakash, A.; Devi, S.R.; Sathiya, P. Arduino ATMEGA-328 microcontroller. *IJIREEICE* **2015**, *3*, 27–29. [CrossRef]
30. Mluyati, S.; Sadi, S. Internet of Things (IoT) Pada Prototipe Pendeteksi Kebocoran Gas Berbasis MQ-2 Dan SIM800L. *J. Tek.* **2019**, *7*, 2.
31. Oudah, M.; Al-Naji, A.; Chahl, J. Hand Gestures for Elderly Care Using a Microsoft Kinect. *Nano Biomed. Eng.* **2020**, *12*, 3.
32. Ganokratanaa, T.; Pumrin, S. The vision-based hand gesture recognition using blob analysis. In Proceedings of the 2017 International Conference on Digital Arts, Media and Technology (ICDAMT), Chiang Mai, Thailand, 1–4 March 2017; pp. 336–341.
33. Alnaim, N.; Abbod, M.; Albar, A. Hand Gesture Recognition Using Convolutional Neural Network for People Who Have Experienced A Stroke. In Proceedings of the 2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), Ankara, Turkey, 10–11 December 2019; pp. 1–6.