Using Complex-Valued Levenberg-Marquardt Algorithm for Learning and Recognizing Various Hand Gestures

Abdul Rahman Hafiz*, Md Faijul Amin* and Kazuyuki Murase*
*Department of Human and Artificial Intelligence System, Graduate School of Engineering, University of Fukui, JAPAN
Email: murase@synapse.his.u-fukui.ac.jp

Abstract—With the advancement in technology, we see that complex-valued data arise in many practical applications, specially in signal and image processing. In this paper, we introduce a new application by generating complex-valued dataset that represents various hand gestures in complex domain. The system consists of three components: real time hand tracking, hand-skeleton construction, and hand gesture recognition. A complex-valued neural network (CVNN) having one hidden layer and trained with Complex Levenberg-Marquardt (CLM) algorithm has been used to recognize 26 different gestures that represents English Alphabet. The result shows that the CLM provides reasonable recognition performance. In addition to that, a comparison among different activation functions have been presented.

I. INTRODUCTION

Recent technological advancements in the human-computer interaction field have shown that conventional tools, such as keyboard, mouse, and light pen, do not provide a natural form of interaction. Even though those tools were the standard forms of input for many decades, the ubiquity of digital systems revealed the urgent requirement for a more reachable interaction method that can be used by anyone regardless of his educational background. Since the hand has always been the natural interaction methods among humans, a recent resurgence in developing new hand modeling techniques has been observed. Regardless of the technique used, the main goals have always been: using descriptive gestures while keeping the computer processing and modeling as simple as possible.

The system can be applied to various background, changeable lighting of the environment and different kinds of human colors. To achieve that, we construct simple representation of human hand (hand-skeleton) after applying edge detection algorithm (Fig.1) to the input image, we then define gestures for each English characters [1].

Recently, complex-valued data are used in many applications, such as array signal processing [2], radar and magnetic resonance data processing [3], communication systems [5], signal representation in complex baseband [6], and processing data in the frequency domain [3].

In our approach, complex-valued data that represent hand gesture can be obtained after applying sequence of filters to the image captured using Kinect camera [7]. We used three layer complex-valued neural network (CVNN), and Complex Levenberg-Marquardt (CLM) algorithm [8] for the training, due to the nature of the data that we can collect from the generated hand-skeleton representation. We investigate the recognition performance with respect to various activation functions in the hidden layer. The output layer, however, uses a recently proposed activation function [11], that helps an output neuron behaving like a discriminative function.

The remainder of the paper is organized as follows. Section II, discusses the procedures to generate the hand-skeleton structure. Section III introduces CLM algorithm and various activation functions. Computer simulation results are discussed in Section IV. Finally, concluding remarks are given in Section V.

II. PROCEDURES

In this research, we utilized the camera of a Microsoft Kinect motion-sensing input device [7], accompanied by OpenCV platform which have the computational capabilities required for real-time image acquisition and handling. Fig.
2 represents a modular view of the final system. The image acquired by the Video Input Module is passed to both the Hand Location Module and the Image Processing Module. While the Hand Location Module is responsible for detecting the location of the hand within the image, the Image Processing Module processes the area that has been previously detected by the Hand Location Module. The output of the Image Processing Module is then passed to the Hand-Skeleton Construction Module which is responsible for creating a skeleton module from the image of the hand. This module has two outputs: skeleton model of the hand, and supplementary data passed to the Hand Location Module to increase the location detection accuracy. The skeletal model is then passed to the CLM where the actual recognition takes place. The following subsections describe in detail each stage of the system:

A. Tracking and Detection

The first step involves separating the image of the hand from the rest of the image. To do that, we used Kinect’s depth map to wipe the background of the image. As we can see in Fig. 3, only the silhouette of the human body was extracted from the image, disposing any other unneeded objects.

Next, we removed the regions not having the color of human skin. The resulting image contains the location of the human hand and face. Then we used HSL representation of color to identify the color of human skin. It is known that HSL representation identifies the color of human skin more accurately than RGB representation[12]. The last step involved separating the image of the hand from image of the face (Fig. 4).

Although the hues of the hand’s color and the face’s color are different, the difference is too small to be considered reliable by itself. Accordingly, we had to support that by another source of information. For instance, when the system is initialized, it depends on the motion of the hand to distinguish it from the face, and then the system would realize the location of the hand using the feedback of the hand-skeleton construction part of the system.

From Fig. 4, we can notice that when the fingers are close to each other, we might lose some information about each finger state. To compensate for that problem, we used a sequence of image processing algorithms to aid the correct recognition of the finger’s state, as described in the next section.

B. Image Processing

After locating the human hand in the image, the system filters the region where the hand is located as shown in Fig. 5.

First, the system applies Sobel Edge Detection Algorithm [10] to get a contour of the hand. This filter scans the image for sharp contrast differences, and assigns a white color shade equivalent to the contrast in that region.
Next, by restricting the whiteness to a specific threshold, the system deletes any noisy edges effectively creating a sharper edge representation. However, that step would produce disconnected regions in the edges of the hand, affecting the outcome of thinning algorithm. To avoid that drawback, we used a dilation algorithm resulting in a fully connection figure.

The dilated image is then passed to a thinning algorithm [8]. This algorithm generates one line of pixels representing the branching of the structure. The outcome of that step is: a representing the hand as interconnected lines meeting at multiple nodes.

The final step involves reading that representation. The system creates pairs of data for each branching by traces the line that connects the nodes, calculating the length and the angle of these lines. The calculation method for the length of the line which connects two nodes and its relative angle is shown in Fig. 6.

Fig. 6 shows the method used for converting a hand-skeleton model to a set of amplitude-phase pairs that can be processed by the CVNN. The conversion process involves the recursive measurement of the branches relative to the root branches. The complete conversion process is made of the following steps:

1) Locate the lower-most root branch.
2) Measure the angle between branches from the node at the end of the root branch and the extension of the root branch ($\theta_1$) and the length of that branch ($r_1$).
3) Repeat step 2 until you reach branches that don’t branch at their terminal nodes.
4) Input the parameters of the non-branching branches (terminal nodes) to the CVNN.
5) In the case when the number of terminal branches is not equal to 5 (that is, less than the number of human fingers), other terminal branches are assigned with zeroes for both phase and amplitude. For simplicity, the assumed pictures will follow a predefined pattern, as can be seen in Fig. 6.

### III. Complex-Valued Levenberg-Marquardt (CLM) Algorithm for Hand Gesture Recognition

The image processing steps discussed above produces a complex vector $x = [x_1, x_2, \ldots, x_8]^T$ consisting of eight elements. Each element $x_i$, $1 \leq i \leq 8$, is a Cartesian representation of each segment of the hand skeleton, i.e., $x_i = r_i e^{\theta_i} = r_i \cos(\theta_i) + j r_i \sin(\theta_i)$, where $j = \sqrt{-1}$. To classify the patterns represented by complex-valued feature vectors, we apply a feedforward complex-valued neural network (CVNN) with one hidden layer. The output layer uses an activation function proposed in [11] which can act as a discriminating function giving a discriminating score. We call the function here as $\text{discrim}$. The function has the following form:

$$f_{C\to R}(z) = (f_R(u) - f_R(v))^2$$

where $z = u + j v$ denotes the weighted sum of input signals along with the bias and called net-input and $f_R()$ is a real-valued log-sigmoid function. The hidden layer, however, may take any activation function found in the literature of the CVNNs. Recently CLM has been proposed by [8] as a fast learning algorithm for the feedforward complex-valued neural networks (CVNN). The CVNN in this study is trained with the CLM algorithm because of its faster convergence. Since we have a total of 26 different gestures, the output layer has 26 neurons, each representing one gesture.

In order to see the effect of hidden layer activation functions in the hand gesture recognition problem, we investigate a number of complex activation functions listed below.

splitTanh [14] : $f(z) = \tanh u + j \tanh v$ \hspace{1cm} (2)

splitSigm [14] : $f(z) = \frac{1}{1 + e^{-u}} + j \frac{1}{1 + e^{-v}}$ \hspace{1cm} (3)

linear : $f(z) = z$ \hspace{1cm} (4)

George [11] : $f(z) = z/(c + |z|/r)$, $c, r$, constants \hspace{1cm} (5)

tan [15] : $f(z) = \frac{e^{iz} - e^{-iz}}{i(e^{iz} + e^{-iz})}$ \hspace{1cm} (6)

sin [15] : $f(z) = \frac{e^{iz} - e^{-iz}}{2i}$ \hspace{1cm} (7)

atan [15] : $f(z) = \arctan z = \int_0^z \frac{dt}{1 + t^2}$ \hspace{1cm} (8)

asin [15] : $f(z) = \arcsin z = \int_0^z \frac{dt}{(1 - t^2)^{1/2}}$ \hspace{1cm} (9)

acos [15] : $f(z) = \arccos z = \int_0^z \frac{dt}{(1 - t^2)^{1/2}}$ \hspace{1cm} (10)

sinh [15] : $f(z) = \frac{e^z - e^{-z}}{2}$ \hspace{1cm} (11)

tanh [15] : $f(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$ \hspace{1cm} (12)

atanh [15] : $f(z) = \arctanh z = \int_0^z \frac{dt}{(1 + t^2)^{1/2}}$ \hspace{1cm} (13)

asinh [15] : $f(z) = \arcsinh z = \int_0^z \frac{dt}{(1 - t^2)^{1/2}}$ \hspace{1cm} (14)

### IV. Results

A pattern set with 26 hand gestures were collected. The data set comprises 520 patterns and was divided into a training set (50%), a validation (25%), and a testing set (25%).
We test the system robustness in simple data collection and noise deletion tasks, the system could work in 10 frames per second (fps) and could read the hand state for 90% of the time [1].

For recognizing the English character, we defined distinguishable gestures of the hand to represent each character. These gestures have been chosen so that it will be easier for the system to recognize them. Consideration was also taken on human’s natural skills to move from one gesture to other. Our algorithm detects the edges between the fingers even though the fingers are stuck together. This allowed us to design a simple representation for each character as shown in Fig. 7.

![Hand gesture for each English character](image)

Fig. 7. Hand gesture for each English character, we can notice that the gestures are differing in the number of fingers and the angles they make with each other.

We presented all the input data to the CLM and computed the outputs and the validation error. From Fig.8 we can notice that the optimal number for the neurons in the hidden layer was 4.

![Validation error for different activations function and number of neurons in the hidden layer](image)

Fig. 8. Validation error for different activations function and number of neurons in the hidden layer

The learning process was terminated if some stopping criteria were met, such as, validation error increases rather than a decrease.

We test the system robustness in simple data collection and noise deletion tasks, the system could work in 10 frames per second (fps) and could read the hand state for 90% of the time [1].

For recognizing the English character, we defined distinguishable gestures of the hand to represent each character. These gestures have been chosen so that it will be easier for the system to recognize them. Consideration was also taken on human’s natural skills to move from one gesture to other. Our algorithm detects the edges between the fingers even though the fingers are stuck together. This allowed us to design a simple representation for each character as shown in Fig. 7.

![Hand gesture for each English character](image)

Fig. 7. Hand gesture for each English character, we can notice that the gestures are differing in the number of fingers and the angles they make with each other.

We presented all the input data to the CLM and computed the outputs and the validation error. From Fig.8 we can notice that the optimal number for the neurons in the hidden layer was 4.

![Validation error for different activations function and number of neurons in the hidden layer](image)

Fig. 8. Validation error for different activations function and number of neurons in the hidden layer

The learning process was terminated if some stopping criteria were met, such as, validation error increases rather than a decrease.

We test the system robustness in simple data collection and noise deletion tasks, the system could work in 10 frames per second (fps) and could read the hand state for 90% of the time [1].

For recognizing the English character, we defined distinguishable gestures of the hand to represent each character. These gestures have been chosen so that it will be easier for the system to recognize them. Consideration was also taken on human’s natural skills to move from one gesture to other. Our algorithm detects the edges between the fingers even though the fingers are stuck together. This allowed us to design a simple representation for each character as shown in Fig. 7.

![Hand gesture for each English character](image)

Fig. 7. Hand gesture for each English character, we can notice that the gestures are differing in the number of fingers and the angles they make with each other.

We presented all the input data to the CLM and computed the outputs and the validation error. From Fig.8 we can notice that the optimal number for the neurons in the hidden layer was 4.

![Validation error for different activations function and number of neurons in the hidden layer](image)

Fig. 8. Validation error for different activations function and number of neurons in the hidden layer

The learning process was terminated if some stopping criteria were met, such as, validation error increases rather than a decrease.

We test the system robustness in simple data collection and noise deletion tasks, the system could work in 10 frames per second (fps) and could read the hand state for 90% of the time [1].

For recognizing the English character, we defined distinguishable gestures of the hand to represent each character. These gestures have been chosen so that it will be easier for the system to recognize them. Consideration was also taken on human’s natural skills to move from one gesture to other. Our algorithm detects the edges between the fingers even though the fingers are stuck together. This allowed us to design a simple representation for each character as shown in Fig. 7.

![Hand gesture for each English character](image)

Fig. 7. Hand gesture for each English character, we can notice that the gestures are differing in the number of fingers and the angles they make with each other.

We presented all the input data to the CLM and computed the outputs and the validation error. From Fig.8 we can notice that the optimal number for the neurons in the hidden layer was 4.

![Validation error for different activations function and number of neurons in the hidden layer](image)

Fig. 8. Validation error for different activations function and number of neurons in the hidden layer

The learning process was terminated if some stopping criteria were met, such as, validation error increases rather than a decrease.

We test the system robustness in simple data collection and noise deletion tasks, the system could work in 10 frames per second (fps) and could read the hand state for 90% of the time [1].

For recognizing the English character, we defined distinguishable gestures of the hand to represent each character. These gestures have been chosen so that it will be easier for the system to recognize them. Consideration was also taken on human’s natural skills to move from one gesture to other. Our algorithm detects the edges between the fingers even though the fingers are stuck together. This allowed us to design a simple representation for each character as shown in Fig. 7.

![Hand gesture for each English character](image)

Fig. 7. Hand gesture for each English character, we can notice that the gestures are differing in the number of fingers and the angles they make with each other.

We presented all the input data to the CLM and computed the outputs and the validation error. From Fig.8 we can notice that the optimal number for the neurons in the hidden layer was 4.

![Validation error for different activations function and number of neurons in the hidden layer](image)

Fig. 8. Validation error for different activations function and number of neurons in the hidden layer

The learning process was terminated if some stopping criteria were met, such as, validation error increases rather than a decrease.

We test the system robustness in simple data collection and noise deletion tasks, the system could work in 10 frames per second (fps) and could read the hand state for 90% of the time [1].

For recognizing the English character, we defined distinguishable gestures of the hand to represent each character. These gestures have been chosen so that it will be easier for the system to recognize them. Consideration was also taken on human’s natural skills to move from one gesture to other. Our algorithm detects the edges between the fingers even though the fingers are stuck together. This allowed us to design a simple representation for each character as shown in Fig. 7.

![Hand gesture for each English character](image)

Fig. 7. Hand gesture for each English character, we can notice that the gestures are differing in the number of fingers and the angles they make with each other.

We presented all the input data to the CLM and computed the outputs and the validation error. From Fig.8 we can notice that the optimal number for the neurons in the hidden layer was 4.

![Validation error for different activations function and number of neurons in the hidden layer](image)

Fig. 8. Validation error for different activations function and number of neurons in the hidden layer

The learning process was terminated if some stopping criteria were met, such as, validation error increases rather than a decrease.

We test the system robustness in simple data collection and noise deletion tasks, the system could work in 10 frames per second (fps) and could read the hand state for 90% of the time [1].

For recognizing the English character, we defined distinguishable gestures of the hand to represent each character. These gestures have been chosen so that it will be easier for the system to recognize them. Consideration was also taken on human’s natural skills to move from one gesture to other. Our algorithm detects the edges between the fingers even though the fingers are stuck together. This allowed us to design a simple representation for each character as shown in Fig. 7.

![Hand gesture for each English character](image)

Fig. 7. Hand gesture for each English character, we can notice that the gestures are differing in the number of fingers and the angles they make with each other.

We presented all the input data to the CLM and computed the outputs and the validation error. From Fig.8 we can notice that the optimal number for the neurons in the hidden layer was 4.

![Validation error for different activations function and number of neurons in the hidden layer](image)

Fig. 8. Validation error for different activations function and number of neurons in the hidden layer

The learning process was terminated if some stopping criteria were met, such as, validation error increases rather than a decrease.

We test the system robustness in simple data collection and noise deletion tasks, the system could work in 10 frames per second (fps) and could read the hand state for 90% of the time [1].

For recognizing the English character, we defined distinguishable gestures of the hand to represent each character. These gestures have been chosen so that it will be easier for the system to recognize them. Consideration was also taken on human’s natural skills to move from one gesture to other. Our algorithm detects the edges between the fingers even though the fingers are stuck together. This allowed us to design a simple representation for each character as shown in Fig. 7.

Table 1, shows the classification error for different activation functions sorted by the smallest value. From the table we can notice that the split type activation functions performed better for this problem.

<table>
<thead>
<tr>
<th>Activation Functions</th>
<th>Classification Error(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>splitTanh[14]</td>
<td>08.46</td>
</tr>
<tr>
<td>splitSigm[14]</td>
<td>11.29</td>
</tr>
<tr>
<td>linear[15]</td>
<td>13.59</td>
</tr>
<tr>
<td>asin[15]</td>
<td>15.65</td>
</tr>
<tr>
<td>tan[15]</td>
<td>16.41</td>
</tr>
<tr>
<td>discrim[11]</td>
<td>17.18</td>
</tr>
<tr>
<td>acos[15]</td>
<td>17.69</td>
</tr>
<tr>
<td>tanh[15]</td>
<td>17.95</td>
</tr>
<tr>
<td>sin[15]</td>
<td>20.00</td>
</tr>
<tr>
<td>asinh[15]</td>
<td>20.35</td>
</tr>
<tr>
<td>sinh[15]</td>
<td>21.54</td>
</tr>
<tr>
<td>atan[15]</td>
<td>23.85</td>
</tr>
</tbody>
</table>

V. CONCLUSION

In this paper, the CLM algorithm has been used in a hand gesture recognition system to distinguish 26 differed gestures (English Alphabet). By using Kinect depth map and the human skin color we could isolate human hand from the rest of the image, then we used a sequence of image filters to generate a descriptive representation of human hand. We call it ”Hand-Skeleton”. This representation allows us to use the CLM algorithm for learning and recognition stage. The results shows that the CLM algorithm with the split type activation functions achieves the highest recognition performance.

ACKNOWLEDGMENT

This study was supported by grants to K.M. from Japanese Society for promotion of Sciences and Technology, and the University of Fukui.
REFERENCES


