Research Article

Detecting Direction of Pepper Stem by Using CUDA-Based Accelerated Hybrid Intuitionistic Fuzzy Edge Detection and ANN

Mahit Gunes¹ and Hasan Badem²

¹Department of Electrical and Electronical Engineering, Kahramanmaras Sutcu Imam University, Kahramanmaras, Turkey
²Department of Computer Engineering, Erciyes University, Kayseri, Turkey

Correspondence should be addressed to Mahit Gunes; mgunes@ksu.edu.tr

Received 1 June 2016; Accepted 8 September 2016

1. Introduction

Image processing, changing the current images and graphics, and separating, improving, or defining an object in an image have found many application areas in industry [1, 2]. Since 1960, these are used almost in many areas such as space research, medical, military, education, agriculture, industry, geography, archeology, physics, biology, character recognition, fingerprint recognition, and so forth [3]. In this study, it is aimed to develop image processing algorithms for destemming pepper which is used in the production of red pepper flakes or chili powder from the red pepper that is peculiar to Kahramanmaras and Gaziantep region [4]. Red pepper in form of chili powder or red pepper flakes are broadly used in foods for its flavor, taste, aroma, bitterness, and colors all around the world [4].

In the process of producing red pepper flakes or chili powder from red pepper, two methods are used: drying in the sun and oven-drying [5, 6]. Since destemming the pepper is a complex process, there are few scientific studies on destemming pepper in the literature. The study with patent number US2008/0289515A1 includes a mechanic system which is developed for destemming the pepper [7]. In this study, machine vision algorithms using laser sensors have been used. In another study from M-Tec lab in New Mexico State University, they are attempted in a mechanical system by developing a computer vision system to destem pepper by using distance from edge image method with the images from X-ray cameras [8].

Destemming the pepper processing is still carried out by human power within the exceptions of this product. The developed system includes 2 phases of image processing and pattern recognition.

In the phase of image processing, a new hybrid intuitionistic fuzzy edge detection (HIFED) algorithm presents very successful results in linear structures since it gives results according to the rules determined by the expert designers. In this algorithm, more stable results are obtained since it includes hesitated calculation that minimizes the expert errors, which is not available in classical fuzzy logic. In developed system, HIFED has been used for extraction of pepper edges on the images captured from camera. In pure intuitionistic fuzzy logic (IFED) algorithm for edge extraction, threshold process is carried out on a static value determined by the system designer. Threshold process applied over a static value is a weakness for the algorithm.
2. System Structure

HIFED algorithm has been used for image processing system. In pure IFED algorithm, threshold is carried out on a static value. Otsu method which is a method of automatic threshold value calculation with the numeric values of the image is used for the first time in this study in order to eliminate this disadvantage in industrial application, and more apparent patterns are obtained. The directions of the pepper are determined by using ANN which classifies the images of edges extracted. The output from this classification process is conveyed to pepper inversion unit through the drive circuit. The general block diagram of the developed system is shown in Figure 1 and the general algorithm of the system is shown in Figure 2. The developed system includes subunits such as computer vision system, image capturing, image processing algorithm, and pattern recognition algorithm.

2.1. Computer Vision System. Computer vision system includes all the phases of the process from capturing the images of camera with frame grabber card to conveying them to computer. While developing the system, the most optimal resolution in per unit of time is determined as 200 x 400, considering the quality of the captured image and the process speed of the numbers of peppers analyzed in per unit of time.

2.2. Image Capturing. The image captured from digital camera is used in OpenCV library for processing at both CPU-based and GPU-based (CUDA) software. It is the most ideal library for capturing image, reaching all the feature of the image, working at CUDA platform, and supporting C programming language for developing software.

Capturing images from a camera connected to the system provides opening empty window and storing the images from camera there. It gives the opportunity of processing with numeric values of the image in processing functions of both CPU- and GPU-based applications on the stored frame object.

2.3. Image Processing Algorithm. In the developed system, firstly IFED algorithm is used on the image to determine the direction of pepper’s arrival to grinding machine by processing the pepper images. In this algorithm, process of threshold is carried out over a static threshold value by the method of trial and error. To eliminate this disadvantage, Otsu method is used. Therefore, HIFED algorithm is obtained. These algorithms are used at both CPU-based and GPU-based applications.

2.3.1. Basic Properties of CUDA and GTX 480. CUDA is a recently developing hardware and programming API (Application Programming Interface) for GPU programming that is released by NVIDIA Company in 2006 and it can be used with slight additions to C language. It is used by many users very easily and effectively because of its GPU structure.

CUDA programs consist of two parts as serial (CPU) and parallel (GPU). Parallel parts are called kernel. The required developing application cannot be worked on completely by CUDA because NVCC (NVIDIA C Compiler) functions by separating the source codes as CPU and GPU codes.

The core structure of the NVIDIA GTX 480 graphic card is used in this study; CUDA cores are collected in groups of 32 and called Stream Multiprocessor (SM). There are 16 SMs in total. These SMs take the threads in groups of 32, and block of each thread is called warp. All threads in a warp carry out the same code. Device has 3 different memories: global, constant, and texture memory. Global memory is the part where data and processes in GPU are stored. The data is stored till end of the application. CPU and threads can communicate. GTX 480 graphic card has 1.5 GB DDR5 capacity. Constant memory is written by CPU and read by threads. The data is kept till end of the process. It is fast since it is cached and its size is 64 KB. Texture memory is for filtering, image processing, and 3 dimensional area calculations. Data writing process is carried out by CPU, and reading process is carried out by threads. Texture memory size might change between 6 and 8 KB. These 3 types of memories can communicate with host memory. Each multiprocessor basically has 3 types of memory models. These are local memory and register and shared memory. Local memory is memory of threads. It is closed to the user access. Shared memory is where data of each thread block is stored. Shared memory is the fastest memory model since threads can directly reach it. Memory size of shared memory in each thread block is limited to 48 KB.
2.3.2. Intuitionistic Fuzzy Sets. It is stated that, after Zadeh's classical fuzzy logic theory, Atanassov claimed that the definition is correct though it might not respond correctly in real life. In addition to Atanassov's claim, it is stated that any value determined in real life experience cannot be correctly determined within real membership value [14]. For the solution of this problem, third parameter which is named hesitation value is noted. It should be added to (1) that is Zadeh's classical fuzzy logic theory:

\[ A = \{ x, \mu_A(x), \nu_A(x) \mid x \in X \} . \]  
\[ (1) \]

Here \( \mu_A(x) : X \rightarrow [0, 1] \) is a membership degree of each \( x \) component defined in \( X \) set. According to Zadeh's theory [13], nonmembership value signed with \( \nu_A(x) \) is taken from

\[ \nu_A = 1 - \mu_A(x) . \]  
\[ (2) \]

Mathematical representation of this situation is shown at

\[ A = \{ (x, \mu_A(x), 1 - \mu_A(x)) \mid x \in X \} . \]  
\[ (3) \]

Atanassov suggested that an index of intuitionistic fuzzy logic or a hesitation value statement should be added to Zadeh's theory of fuzzy logic set theory shown in (3) for minimizing the user mistakes. In this new situation, there is fuzzy logic statement with hesitation value. It is known that the total of the membership value and nonmembership value in fuzzy logic set is 1. Values of total of intuitionistic fuzzy logic set theory by adding the intuitionistic degree are shown at

\[ \mu_A(x) + \nu_A(x) + \pi_A(x) = 1 . \]  
\[ (4) \]

Parameters in (4) are as follows:

(i) \( \mu_A(x) \) is membership value of used graphic set.
(ii) \( \nu_A(x) \) is nonmembership value of intuitionistic fuzzy logic. This value is obtained from (5).
(iii) \( \pi_A(x) \) is hesitation value of intuitionistic fuzzy logic. This value is obtained from (6), in which \( C \) is a constant hesitation which is shown as \( c_i \)

\[ \nu_A(x) = 1 - \mu_A(x) - \pi_A(x) , \]  
\[ (5) \]

\[ \pi_A(x) = C (1 - \mu_A(x)) . \]  
\[ (6) \]

2.3.3. Intuitionistic Fuzzy Edge Detection (IFED). In the developed system, firstly IFED algorithm is used on the image to determine the direction of peppers. By Chaira and Ray, Atanassov’s intuitionistic fuzzy set theory is put forward for
edge detection application [15, 16]. IFED algorithm is based on intuitionistic fuzzy divergence calculation. Intuitionistic fuzzy divergence calculation is the probability calculation of hesitation value in intuitionistic fuzzy set. Probability values of each component in a set defined with

\[ H ((p_1, p_2, \ldots, p_n)) = -\sum_{i=1}^{n} [p_i \log_2 (p_i)]. \]  

IFED algorithm is based on intuitionistic fuzzy divergence calculation. The probability calculation of hesitation value in intuitionistic fuzzy set is obtained from

\[ IFD (A, B) = \sum_{i=1}^{n} \left( \mu_A (x_i) - \mu_B (x_i) \right)^2 + \left( \nu_A (x_i) - \nu_B (x_i) \right)^2 + \left( \pi_A (x_i) - \pi_B (x_i) \right)^2, \]  

\[ HIFS (A, B) = \sum_{i=1}^{n} [\mu_A (x_i) - \mu_B (x_i)] + [\nu_A (x_i) - \nu_B (x_i)] + [\pi_A (x_i) - \pi_B (x_i)]. \]  

Membership values of each component in two sets defined with \( P = \{p_1, p_2, \ldots, p_n\} \) and \( Q = \{q_1, q_2, \ldots, q_n\} \) are shown at [17]

\[ D (P, Q) = \sum_{i=1}^{n} p_i \log_2 \frac{p_i}{q_i}. \]  

Szmidt and Kacprzyk define the calculation of membership values between two sets, as the pixels of two images in this study, by using Hamming and Euclidean distance calculation methods that are represented at (9) and (10), respectively [18]:

\[ D_2 (A, B) + D_2 (B, A) = \sum_{i=1}^{n} (2 - m - n), \]

\[ m = (1 - (\mu_A (a_i) - \mu_B (b_i))) \]

\[ + (\pi_A (a_i) - \pi_B (b_i)) \]

\[ \cdot e^{(\mu_A (a_i) - \mu_B (b_i))}, \]

\[ n = (1 - (\pi_A (a_i) - \pi_B (b_i))) \]

\[ + (\mu_A (a_i) - \mu_B (b_i)) \]

\[ \cdot e^{(\pi_A (a_i) - \pi_B (b_i))}. \]

Intuitionistic fuzzy divergence (IFD) is obtained from

\[ IFE (A) = \sum_{i=1}^{n} \mu_A (x_i) e^{[1 - \mu_A (x_i)]}. \]

According to Chira and Ray, intuitionistic fuzzy divergence calculation can be obtained by using the distance calculation between two pictures [15, 16]. Hence, if \( A \) and \( B \) are two images, according to intuitionistic fuzzy entropy (IFE) value between \( A \) and \( B \) images \((A \rightarrow B)\) is obtained from (12) by using (5) and (11):

\[ D_1 (A, B) = \sum_{i=1}^{n} \sum_{j=1}^{n} (1 - (1 - \mu_A (a_i))) e^{[\mu_A (a_i) - \mu_B (b_j)]} \]

\[ - (\mu_A (a_i)) e^{[\mu_A (a_i) - \mu_B (b_j)]} \].

Similarly, according to IFE divergence value between \( B \) and \( A \) images \((B \rightarrow A)\) is obtained from

\[ D_1 (B, A) = \sum_{i=1}^{n} \sum_{j=1}^{n} (1 - (1 - \mu_B (a_i))) e^{[\mu_B (a_i) - \mu_A (b_j)]} \]

\[ - (\mu_B (a_i)) e^{[\mu_B (a_i) - \mu_A (b_j)]} \].

According to IFE, total divergence value is obtained from

\[ D_1 (A, B) + D_1 (B, A) = \sum_{i=1}^{n} \sum_{j=1}^{n} (2 - x - y), \]

\[ x = (1 - \mu_A (a_i) + \mu_B (b_j)) e^{[\mu_A (a_i) - \mu_B (b_j)]}, \]

\[ y = (1 - \mu_B (b_j) + \mu_A (a_i)) e^{[\mu_B (b_j) - \mu_A (a_i)]}. \]
\[
n = (1 - (\pi_B(b_{ij}) - \pi_A(a_{ij})) + (\mu_A(a_{ij}) - \mu_B(b_{ij})))
\cdot e^{(\pi_B(b_{ij}) - \pi_B(a_{ij}) - (\mu_A(a_{ij}) - \mu_B(b_{ij})))}.
\]

(17)

If \( A = \{x, \mu_A(x), \nu_A(x) \mid x \in X\} \) is intuitionistic fuzzy logic set calculated from pixel values of image and \( B = \{x, \mu_B(x), \nu_B(x) \mid x \in X\} \) is intuitionistic fuzzy logic set used from the fuzzy extraction sets and same sized image pictures.

\[
\text{IFD}(i, j) = \max_N \left[ \min_r \left( \text{IFD}(a_{ij}, b_{ij}) \right) \right].
\]

(19)

In (19), value of IFD\((a_{ij}, b_{ij})\) is between fuzzy extraction \((b_{ij})\) and their image pictures \((a_{ij})\). \( N \) is the number of fuzzy extraction sets and \( r \) is the component number of fuzzy extraction sets' squares. IFD\((i, j)\) is attained after processing the complete pixel positions of the image. Finally, IFD\((i, j)\) within the same size as original image is attained in each form of matrix. This IFD matrix image is attained by thresholding and thinning the edges. These thresholding and thinning values are set up by trials and error methods.

2.3.4. Otsu Method. Otsu method, which was proposed by Nobuyuki Otsu, is a method of calculating automatic threshold value over the numeric values of images. In this method, it is assumed that image is composed of two different color classes as rear plan and frontal plan of image. The most suitable threshold value that differentiates these two classes is calculated to minimize variance. Otsu method is calculation of threshold value on variance calculation over histogram [20].

Variance is a measure that makes distribution of a series of numbers on its arithmetic average analysis. In case of higher variance, it is shown that values in series of numbers are dispersed. In case of lower variance, it is shown that values in series of numbers are not dispersed [20]. Variance value of a series of number is obtained from

\[
\sigma^2 = \frac{1}{N} \sum_{i=1}^{N} (x_i - \overline{x})^2 \Pr(x_i),
\]

where \( x_i \) represents the expected value, \( \Pr(x_i) \) represents the probability value of expected value, and \( \overline{x} \) represents the weighted average. In Otsu method, threshold value is obtained from (21). \( \omega_1 \) represents the weight class (1st and 2nd class) and \( t \) represents threshold value in

\[
\sigma^2_w(t) = \omega_1(t) \sigma^2_1(t) + \omega_2(t) \sigma^2_2(t).
\]

(21)

as template intuitionistic fuzzy logic set, 16 fuzzy extraction sets of \( 3 \times 3 \) were seen (16 extraction sets of IFED (18)) and are formed for IFED algorithm [15]. Fuzzy extraction sets are important since they determine the type and direction of edge. These sets represent edge examples "a," "b," and "d" in edge extraction sets which represent pixel equivalence of edge examples. "a" and "b" values are found completely with trial and error method. However, in literature survey, it is seen that the most suitable value is predicted as \( a = 0.3 \) and \( b = 0.8 \) [15]:

IFD is measured for each \((i, j)\) pixel position in image. IFD\((i, j)\) value is obtained from Max–Min relation in (19) from the fuzzy extraction sets and same sized image pictures.

Otsu showed that minimum value of variance in classes and maximum value of variance interclasses are the same. \( \mu_i \) represents class average and \( \omega_i \) represents the class probability in

\[
\sigma^2_b(t) = \sigma^2 - \sigma^2_w(t) = \omega_1(t) \omega_2(t) (\mu_1(t) + \mu_2(t))^2,
\]

(22)

where \( \omega_i(t) \) as class probability and \( t \) as threshold value are obtained from the histogram in

\[
\omega_i(t) = \frac{1}{\sum_{i=0}^{t} p(i)}.
\]

(23)

\( \mu_i(t) \) as class average is obtained from

\[
\mu_i(t) = \frac{1}{\sum_{i=0}^{t} p(i)} \chi(i),
\]

(24)

where \( \chi(i) \) in (24) is the central value of histogram value. Similarly, \( \omega_3(t) \) value is calculated for higher values than \( t \) threshold values on histogram.

In the developed system, Otsu method is used for automatic thresholding process of hybrid intuitionistic fuzzy logic edge detection algorithm.

2.3.5. Implementation of CUDA-Based Hybrid Intuitionistic Fuzzy Edge Detection (HIFED) Algorithm. CUDA-based HIFED algorithm is divided into 3 kernels for implementing GPU-based application. First kernel is used to find Otsu thresholding value. Second kernel is run to process IFD application to image for all pixels. Third kernel is used for defuzzification of the image based Otsu thresholding. First kernel in Algorithm 1 and second and third kernel in Algorithm 2 are shown. All kernels are used on global memory in CUDA graphic card. Flow chart of the HIFED algorithm is shown in Figure 3 [10].

2.3.6. Artificial Neural Network (ANN). Neural networks are used as a direct substitute for multivariable regression, auto-correlation, linear regression, and trigonometric and other statistical analyses and techniques [21]. Neural networks, which have exceptional ability to develop a common solution
for complex or inexact data, might be used to extract patterns and identify trends which look very complicated to be noticed by either human or any computer techniques. A neural network with training may be considered as an "expert" in the group of knowledge which is given for analyzing. This expert can be utilized for offering projections according to the new situations of importance and giving answers to "what if" questions. As a data stream is analyzed by using a neural network, it is possible to find significant predictive patterns that are not formerly clear to a nonexpert. Accordingly, the neural network can act as an expert. The specific network can be defined by three important components: transfer function, network architecture, and learning rules [22]. It is important to define these components to find an effective solution for the problem. Neural networks contain a comprehensive class of different architectures. One of the commonly used artificial neural network architectures in literature for classification problems is multilayer perceptron (MLP) and it is very effective in pattern classification problems [23, 24].

As Greenberg and colleagues point out that 4–50% of data on training process of artificial neural network should be separated for training and the rest for test process, 44% of the patterns of the study are separated for training and the rest for test process [25].

In this study, architecture of ANN (MLP) is used for estimating the position of red pepper destemming machine. All data include differently 118 samples which are divided into two data sets such as training (52 samples of all data) and test (66 samples of all data). In this study, developed software is used in neural network analyses which has three layers of network with one input layer (3200 neurons), one hidden layer (70 neurons), and one output layer (2 neurons). Designed artificial neural network is shown in Figure 4.

In a model of artificial neural network, if the input values are defined as $X_p = (x_{p1}, \ldots, x_{pn})$, output values are defined as $O_p = (O_{p1}, \ldots, O_{pk})$, and expected values are defined as $d_p = (d_{p1}, \ldots, d_{pk})$, the error values are obtained from

$$\epsilon_{p,j} = o_{p,j} - d_{p,j}. \quad (25)$$

When calculated error values are applied, squares of total errors for each output are obtained from

$$\sum_{p=1}^{P} \sum_{j=1}^{K} (\epsilon_{p,j})^2. \quad (26)$$
Each of the extraction sets elements and calculate the matrix elements of divergence between the intuitionistic fuzzy logic sets. Create 16 sets with values of \( a \) and \( b \). Create a 3×3 image matrix for \((i,j)\) pixel. Apply image matrix to all extraction sets. Get minimum IFD from applying each extraction set. Get maximum IFD from getting minimum IFD. Save max calculated IFD for \((i,j)\). Pixel on image matrix as a size of original image and \( (j < \text{height of image}) \) and \( (i < \text{width of image}) \). Yes \( (i < \text{width of image}) \) and \( (j < \text{height of image}) \). No

Values of \( O_p = (O_{p,1}, \ldots, O_{p,k}) \) in output layer are obtained from

\[
\alpha_k = S(\text{net}_k^{(2)}) = S\left( \sum_j w_{kj}^{(2,1)} x_j^{(1)} \right). \tag{28}
\]

Calculation of output value is repeated till reaching negligibility level of total of errors' squares from (26) [26].

### 3. Experimental Results

GPU- and CPU-based software is used for testing the developed system by using programming languages of CUDA C, C++, and C# as shown in Figure 5. By this software, test process is done on 66 different patterns of pepper. Otsu algorithm, artificial neural network, and performances of HIFED algorithm are analyzed in test processes.

Different patterns are tested on both CPU-based and GPU-based applications to analyze the performance in the developed system. Performance of CPU-based and GPU-based applications is analyzed by computational time over some patterns.

---

Figure 3: Flow chart of HIFED algorithm.

Figure 4: Developed network diagram of ANN.
3.1. Experiment #1: Response of the Developed System to Pepper Coming Backwards. The image obtained after processing the image captured from camera in Figure 6(a) on CPU-based application is presented in Figure 6(b) and on GPU-based application is presented in Figure 6(c). Before applying input of ANN, there must be extracted object and resized extracted image from the edged images. In CPU-based application, extracted and resized image are presented in Figures 6(d) and 6(f). Similarly, in GPU-based application, extracted object image and resized image are shown in Figures 6(e) and 6(g).

Analyzing the data on the pepper pattern seen at Table 1, HIFED algorithm on CPU-based applications lasts 6742 ms that is almost 7 seconds and on GPU-based application lasts almost 17 ms that is 0.017 seconds. This algorithm is accelerated 383 times with CUDA technology. Otsu algorithm on CPU-based applications lasts 0.55 ms and on GPU-based application lasts 0.26 ms. This algorithm is accelerated twice with CUDA technology. Threshold value from the numeric values of the pattern on each application is 133. ANN algorithm on CPU-based applications lasts 75 ms and on GPU-based application lasts almost 76 ms. Response duration of CPU-based application is almost 6818 ms that is likely 7 seconds while response duration of GPU-based application is almost 93 ms that is likely 0.09 seconds. The system is accelerated 72 times with CUDA technology.

<table>
<thead>
<tr>
<th>Table 1: Collected data on phases of system during processing of Experiment #1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Values of phases of systems</td>
</tr>
<tr>
<td>Processing time of Otsu method (ms)</td>
</tr>
<tr>
<td>Thresholding values</td>
</tr>
<tr>
<td>Processing time of HIFED (ms)</td>
</tr>
<tr>
<td>ANN process time</td>
</tr>
<tr>
<td>Total response duration (ms)</td>
</tr>
<tr>
<td>Direction of pepper</td>
</tr>
<tr>
<td>Otsu method proportion (CPU/GPU)</td>
</tr>
<tr>
<td>HIFED proportion (CPU/GPU)</td>
</tr>
<tr>
<td>Response of system proportion (CPU/GPU)</td>
</tr>
</tbody>
</table>
It is clearly seen in Table 1 that one frame is processed within 7 seconds in CPU-based application while 11 are processed within one second in GPU-based application. For this reason, GPU-based application is suitable for real-time application. This duration of process leads to successful results.

3.2. Experiment #2: Response of the Developed System to Pepper Coming Upwards. The images obtained after processing the image captured from camera in Figure 7(a) on CPU-based application are presented in Figure 7(b) and on GPU-based application are presented in Figure 7(c). Extracted and resized images obtained from CPU-based application for applying input of ANN have been shown in Figures 7(d) and 7(f). Similarly, the GPU-based extracted images were also applied as inputs to the ANN shown in Figures 7(e) and 7(g).

By analyzing the data on the pepper pattern seen in Table 2, HIFED algorithm on CPU-based applications lasts 6627.88 ms that is almost 7 seconds and on GPU-based application lasts almost 17 ms that is 0.017 seconds. This algorithm is accelerated 376 times with CUDA technology. Otsu algorithm on CPU-based applications lasts 0.55 ms and on GPU-based application lasts almost 0.25 ms. This algorithm is accelerated almost twice with CUDA technology. Threshold value from the numeric values of the pattern on each application is 123. ANN algorithm on CPU-based applications lasts 79 ms and on GPU-based application lasts almost 79 ms. Response duration of CPU-based application is almost 6707 ms that is likely 7 seconds while response duration of GPU-based application is almost 96 ms that is likely 0.09 seconds. The system is accelerated 69 times with CUDA technology.

As the image captured from camera which is used in the system is 50 fps, it is impossible applying it to real-time application since 1 frame is processed within 7 seconds on CPU-based applications. However, on GPU-based application, it is possible to apply it to real-time application since II frames are processed within a second. This duration of process leads to successful results.

3.3. Accuracy Rate of ANN on CPU-Based System. Distribution of the test pattern results that is obtained from the artificial neural network used in CPU-based application is seen in Table 3. Highly successful result that is 91.30% correct is obtained from the patterns. In literature survey, Lee and colleagues as well as Do and colleagues state that obtaining 80% correct result from artificial neural network is a successful result [27, 28].

3.4. Accuracy Rate of ANN on GPU-Based System. Distribution of the test pattern results that is obtained from the artificial neural network used in GPU-based application is seen in Table 4. Highly successful result that is 85.29% correct is obtained from the patterns.
Table 5: Data on the results of patterns on CUDA technology.

<table>
<thead>
<tr>
<th>Pattern Type</th>
<th>Speed (times faster)</th>
</tr>
</thead>
<tbody>
<tr>
<td>The highest speed from Otsu algorithm</td>
<td>3.12</td>
</tr>
<tr>
<td>The lowest speed from Otsu algorithm</td>
<td>1.61</td>
</tr>
<tr>
<td>The highest speed from HIFED algorithm</td>
<td>442.20</td>
</tr>
<tr>
<td>The lowest speed from hybrid HIFED algorithm</td>
<td>364.71</td>
</tr>
<tr>
<td>The highest speed from ANN algorithm</td>
<td>1.74</td>
</tr>
<tr>
<td>The lowest speed from ANN algorithm</td>
<td>0.75</td>
</tr>
<tr>
<td>The highest speed from the general of the system</td>
<td>79.99</td>
</tr>
<tr>
<td>The lowest speed from the general of the system</td>
<td>54.07</td>
</tr>
</tbody>
</table>

3.5. Contribution of CUDA Technology to the System. In the developed system, it is aimed to get maximum performance from the system by developing both CPU-based and GPU-based applications. As seen in Table 5 that is data of processing the patterns in the system, the developed system is accelerated 3 times faster than Otsu algorithm at most, 442 times faster than hybrid intuitionistic fuzzy logic detection algorithm at most, and 79 times faster than total response duration of the system at most. In literature survey, it is seen that these acceleration rates are highly successful.

4. Conclusion

Since pepper extracting process is still carried out by human power within the exceptions of this product, it is the highest expenses necessity for producers at sides of both process duration and employer cost. As computer vision systems are commonly used in every area of industry, it is aimed in this study to determine the direction of the pepper by defining the patterns of pepper from the images captured from the camera. Within this aim, HIFED algorithm is used as image processing algorithms. ANN is used for the classification of patterns of pepper in processed images. All the image processing algorithms are developed with CUDA technology by utilizing GPU technology to improve system performance.

Edge detection is done in the images captured from camera by using IFED algorithm, which is developed by Chaïra and Ray. As a static threshold value is used on the process of IFED, it leads to unacceptable error level due to image values. Yalcın et al. propose HIFED algorithm with Otsu method whose automatically thresholding value over numeric values of the images is used as an edge extraction algorithm to eliminate the problem.

ANN is used for the classification of patterns of pepper for the edge extractions of the pepper. For ANN model, 52 of 118 red pepper patterns are separated for training and 66 are separated for testing. Result from the HIFED is downsized to 40 × 80 resolution and applied as the entry of ANN, and accuracy of GPU and CPU is obtained by 91.30% and 85.29%, respectively, which is an exceedingly high result.

Graphics card supported with CUDA NVIDIA GTX 480 GPU is used to utilize the performance of the developed system. While developing the system, two applications are developed to process on both CPU-based and GPU-based. By using GPU technology, the developed system is accelerated 3 times faster than Otsu algorithm at most, 442 times faster than HIFED at most, and 79 times faster than total response duration of the system at most. It is seen that these acceleration rates are highly successful.

Competing Interests

The authors declare that they have no competing interests.

References


