International Journal of Control and Automation
Vol.7, No.7 (2014), pp.85-96
http://dx.doi.org/10.14257/ijca.2014.7.7.07

Implementaion of Real-time Pedestrian Detection Based on Parallel Processing with a Dual Core Processor

Kwang-Yeob Lee

Seokyeong University
Seoul, Korea
kylee@skuniv.ac.kr

Abstract

This paper proposes a method for optimizing and parallelizing a pedestrian detection algorithm based on CENTRIST in the embedded environment. Pedestrian detection is applied to various areas, but there is a difficulty in the real-time processing of pedestrian detection in the embedded environment. In this paper, a pedestrian detection algorithm that has improved performance by reducing unnecessary memory access due to duplicate computations for real-time processing. The proposed algorithm was implemented in the ALDEBARAN embedded processor(300MHz) which can perform parallel processing. For efficient parallel processing, images were divided into halves which were then processed separately in two CPU cores and the volume of computations was balanced between the two CPU cores. For input images, 512x360 sized images were used. The single core showed the performance of 3.1 frames per second. In the dual core for which the wait time in the process of parallelizing was reduced, the performance improved by about 55% to 4.8 frames per second.

Keywords: *Pedestrian Detection, CENTRIST, Parallel Processing, Sobel, Dual Core*

1. Introduction

Pedestrian detection in image recognition technology is applied to various areas including automatic driving systems, driving support systems, security systems, and traffic signal systems. Pedestrian detection is a technology for detecting standing or walking people in inputted images and is being actively researched in many applications. Pedestrian detection technology is important because it can reduce traffic accidents that may injure people. However, it is a very difficult problem to detect pedestrians in images where there are various postures of people, different lighting conditions, and various backgrounds and to maintain a high detection rate. The main cause of these problems is that visual features for finding pedestrians in various postures and backgrounds are insufficient. Sobel and CENTRIST (CENsus Transform hISTogram) [1] were used in this paper to address this problem and implement a pedestrian detection technology that can find the features of pedestrians. Furthermore, a pedestrian detection algorithm was implemented in the embedded environment and the performance was measured. However, the real-time processing of pedestrian detection in the embedded environment is difficult because a large amount of computations is required. In this paper, the pedestrian detection algorithm [2] using Sobel and CENTRIST was optimized to embedded environment. Furthermore, a parallel pedestrian detection algorithm for real-time processing in the embedded environment was implemented and the performance was improved by balancing the pedestrian detection algorithm that is performed in parallel in two CPU cores.

ISSN: 2005-4297 IJCA

2. Related Work

The pedestrian detection technologies that can be applied to various areas have been studied for many years. Haar Feature-based Cascade Classifier [3] is a classic method of pedestrian detection which finds objects by combining very simple features. This method uses multiple detectors rather than one detector and applies a complex detector through simple detectors for recognition. Furthermore, pedestrian detection using the basic method HOG (Histogram of Oriented Gradient) [4] extracts the distribution characteristics of gradients in the image. However, this method has the problem of slow detection speed. As a way to improve the slow processing speed, the cascade technique was applied to HOG, which is called Cascade HOG [5]. The Cascade HOG method detects pedestrians using HOGs in various sizes of blocks that exist in various locations. Furthermore, unlike conventional methods that detect pedestrians using one-channel image data, ChnFtrs [6] uses multiple channels to detect pedestrians. The ChnFtrs method using multiple channels captures the features of images by using a variety of features about the target area including gradient, color, and brightness. The CENTRIST-based pedestrian detection algorithm is much better than other methods in terms of speed. In this paper, a CENTRIST-based pedestrian detection algorithm that is excellent in speed was applied for the implementation of pedestrian detection in the embedded environment.

3. Proposed Algorithm

3.1. Sobel

Sobel [7] is the most widely used primary differential operator for edge detection. Primary differential is the method for measuring brightness changes, that is, the inclination of the graph. In the image, such inclination is called gradient. Edges can be identified by calculating the gradients. Although mathematical differential cannot be performed, values close to primary differential can be calculated simply using the differences between adjacent pixels. Formula (1) represents horizontal differential, and Formula (2) represents vertical differential.

$$\mathrm{G(x)} = f(x-1,y) - f(x+1,y) \quad \textbf{(1)}$$

$$\mathrm{(x)} = f(x,y-1) - f(x,y+1) \quad \textbf{(2)}$$

A Sobel mask is capable of deriving edges in all directions and equalizing pixel values within the image, which makes it highly resistant against noises. Another characteristic of this mask is that it is more sensitive to diagonal edges than horizontal or vertical edges. Figure 1 shows a 3x3 Sobel mask.

1	0	-1
2	0	-2
1	0	-1

-1	-2	-1
0	0	0
1	2	1

Figure 1. Vertical Mask (left) and Horizontal Mask (right)

In this paper, the edges of images were obtained by using a 3x3-sized Sobel mask as shown in Figure 1. Using Census Transform after obtaining the edges through Sobel can provide much better accuracy than obtaining the feature vectors of original images by only using Census Transform.

3.2. Integral Image

Integral Image [8] refers to image where the value of the preceding pixel is added to the next pixel, as shown in Formula (3) below.

$$IntegImage(\mathrm{x},\mathrm{y}) = \sum_{y'=0}^{y} \sum_{x'=0}^{x} OrgImage(x', y') \quad \textbf{(3)}$$

Where IntegImage (x, y) is the integral image and orgImage (x', y') is the original image. Using integral image, the total sum of pixel values in certain areas can be calculated with ease.

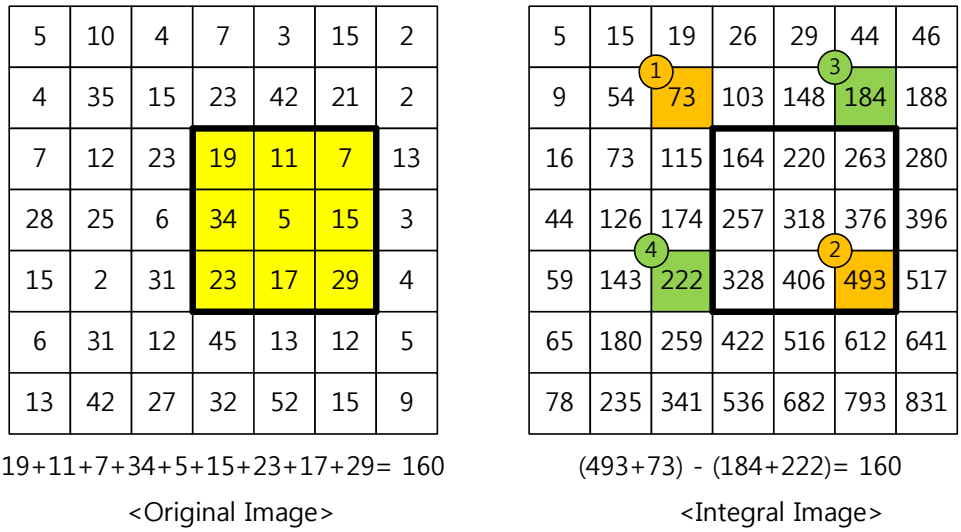

Figure 2. A Random 7x7 One-channel Image

As shown in Figure 2, the sum of pixel values in the colored areas in the original image is equal to the sum of the values of pixels 1 and 2 minus the sum of the values of pixels 3 and 4, which can be described as Formula (4) below.

$$Sum_{block}(x_1, y_1, x_2, y_2) = \sum_{y=y_1}^{y_2} \sum_{x=x_1}^{x_2} (x, y) \quad \textbf{(4)}$$

$$= IntegImage(x_2, y_2) - IntegImage(x_1, y_2)$$

$$-IntegImage(x_2, y_1) + IntegImage(x_1, y_1)$$

The pedestrian detection algorithm proposed in this paper uses a detect window from all image data. When a detect window that is similar to the pedestrian is found by

comparing the sum of pixels in the detect window and the training data, many computations can be reduced by easily obtaining the sum of pixels in the detect window through the integral image.

3.3. CENTRIST (Census Transform)

The Census Transform algorithm in CENTRIST [9] converts images by comparing the intensity with surrounding pixels and extracting stereo matching and feature points. Census Transform creates a Census Transform Window of 3x3 size on the image. Furthermore, the pixels neighboring with the center pixel in the 3x3 Census Transform Window are listed as a bit string. The following is the formula for converting the pixels neighboring the center pixel into a bit string.

$$p_{xy} = \begin{cases} 0 & if \quad p_{center} > p_{xy} \\ 1 & else\ if \quad p_{center} \le p_{xy} \end{cases} \tag{5}$$

The bit string of the Census Transform Window which was converted with Formula (5) is shown in Figure 3.

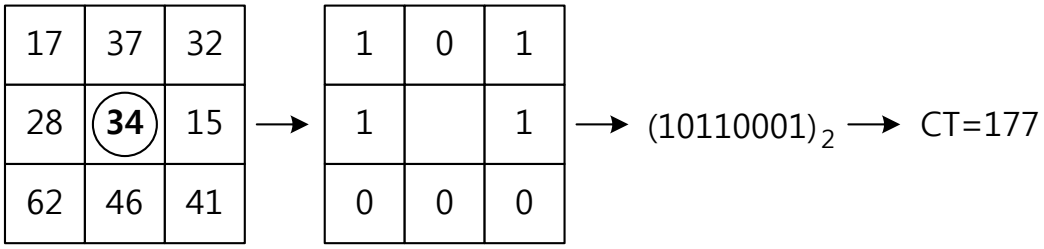

Figure 3. Bit String Conversion Process of Census Transform

The Census Transform algorithm for image conversion can provide the result with simpler and fewer computations than other feature point extraction algorithms. The amount of computations can be reduced and the memory space can be saved because one-channel images are used. In this paper, the Transform algorithm was used to obtain the CT values for the training images of pedestrians and the CT values for the corresponding pixel positions in the input images. The CT values for the training images of pedestrians are used to generate training data and the CT values in the input images are used to detect pedestrians.

3.4. CENTRIST (Histogram)

Histogram [10] in CENTRIST is a bar graph representation of the image data to show the features of the given image data at one glance. The images can be analyzed by using the distribution of brightness values comprising the images. Figure 4 shows the actual histogram in a specific area of the image.

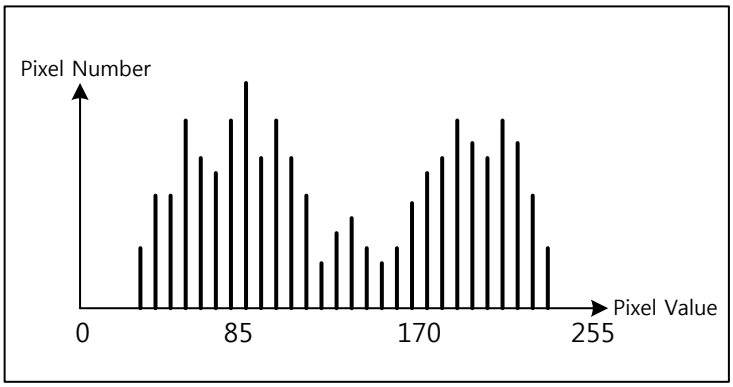

Figure 4. Actual Histogram in a Specific Area

The horizontal axis which is the brightness values of pixels in one-channel images has a range of 0-255. The vertical axis represents the number of pixels corresponding to the pixel brightness value. An analysis of an image's histogram gives information about the brightness distribution and contrast of the image. This information can be used to improve image quality or detect specific objects. In this paper, the histogram of CT indices that were calculated from the training image of pedestrians and the histogram of the CT indices calculated from the pixel positions corresponding to the input images were obtained.

4. Implementation and Results

4.1. CENTRIST-based Pedestrian Detection

In this paper, a CENTRIST-based pedestrian detection algorithm was implemented. The CENTRIST-based pedestrian detection algorithm obtains edges through the Sobel algorithm from the input image data and extracts strong outlines using the Census Transform algorithm. Figure 5 shows the input image, and Figure 6 shows the image data resulting from the processing of the input image with the Sobel and Census Transform algorithms.

Figure 5. Input Image

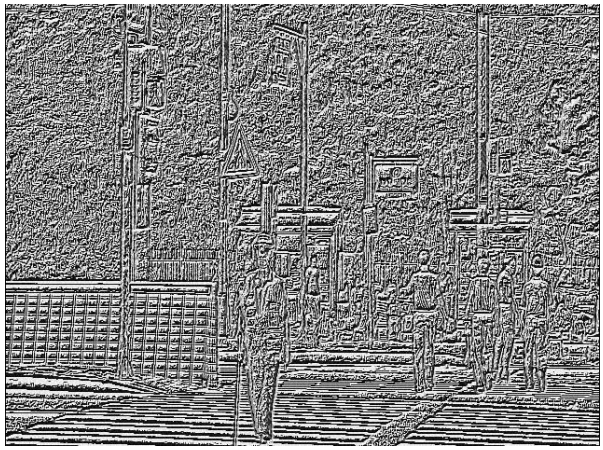

Figure 6. Image Processed with the Sobel and Census Transform Algorithms

Processing the input image in Figure 5 with the Sobel and Census Transform algorithms produces the outlines of pedestrians as shown in Figure 6. The first score for the detect window is generated with the image data and training data which contain the outline information, and integral data to find the detect window that has similar features as the pedestrian. Then the second score is generated with a histogram and the detect window that is detected as pedestrian is found.

4.2. Optimization of Pedestrian Detection Algorithm

In this paper, the part of the conventional CENTRIST-based pedestrian detection algorithm that has the largest amount of computations was optimized to the ALDEBARAN board which consists of two embedded processors (dual cores). The conventional CENTRIST-based pedestrian detection algorithm generates the largest amount of computations when generating a score for comparing with the image data of the image to be compared in real time using training data. The reason for many computations is that to generate a score for comparison, the total image is moved in one pixel units using the detect window. Many duplicate computations are used when the image is moved in one pixel units. In this paper, performance was improved by reducing unnecessary memory accesses due to duplicate computations when generating the score for comparison with training data. In this paper, an improved CENTRIST-based pedestrian detection algorithm in the embedded environment was proposed and was implemented with the ALDEBARAN board (300MHz) which supports an embedded environment. Furthermore, input images with the size of 512x360 pixels were used. For pedestrian detection dataset, the INRIA Person Dataset was used to generate the training data required for pedestrian detection. To implement a pedestrian detection algorithm that is appropriate for the ALDEBARAN board, double-type training data were converted to Int-type training data. The improved CENTRIST-based pedestrian detection algorithm implemented on the ALDEBARAN board showed a performance of 3.1 frames per second. Figure 7 shows the resulting screen of the pedestrian detection algorithm implemented on the ALDEBARAN board.

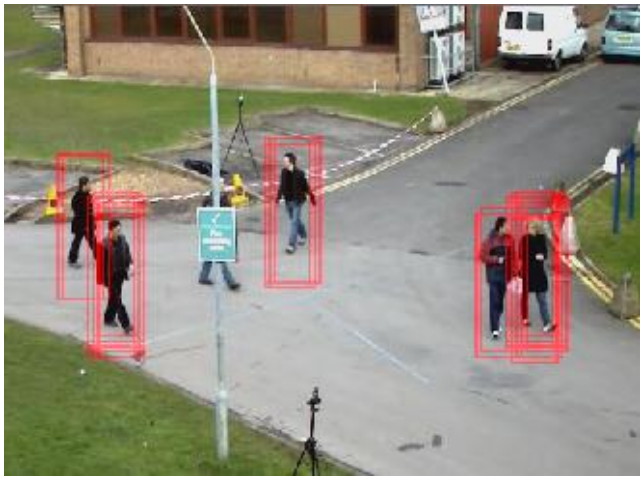

Figure 7. Implementation of the Pedestrian Detection Algorithm on the ALDEBARAN Board

Table 1 compares the pedestrian detection performance among the proposed method, the HOG-cascade [5] method, and the CENTRIST [2] method.

Table 1. Pedestrian Detection Performance

	Image Size	Frame Per Second
HOG cascade[5]	512x384	0.83 fps
CENTRIST[2]	512x360	0.7 fps
This Paper	512x360	3.1 fps

4.3. Parallelization of Pedestrian Detection Algorithm

In this paper, a CENTRIST-based pedestrian detection algorithm that is appropriate for embedded environment was implemented. However, real-time processing is difficult due to a large amount of computations. Thus, for real-time processing in the embedded environment, the pedestrian detection algorithm using CENTRIST was implemented with dual core through parallelization on the ALDEBARAN board which allows parallel processing. Instead of the OpenMP method which divides the repeated parts by the number of core and then allocates them, a parallel processing method that is appropriate for the ALDEBARAN board was used. The pedestrian detection algorithm in the single core performs computations by using the data of all images as shown in Figure 8.

Figure 8. Image Computation in the Single Core

To process in dual core by parallelizing the pedestrian detection algorithm, as shown in Figure 9, one image is divided into halves and the upper image is processed in Core_0 and the lower image is processed in Core_1. Each core processes a half regardless of the other core instead of processing only the repeated parts.

Figure 9. Image Computed in Dual Core

For parallelization appropriate for the ALDEBARAN board, the entire pedestrian detection algorithm is divided into halves for separate processing. Because each core independently performs computation, a shared memory was used as shown in Figure 10. In this shared memory, the wait signals that allow processing of the same frame by different core and the positions of feature points of the detected pedestrian are saved.

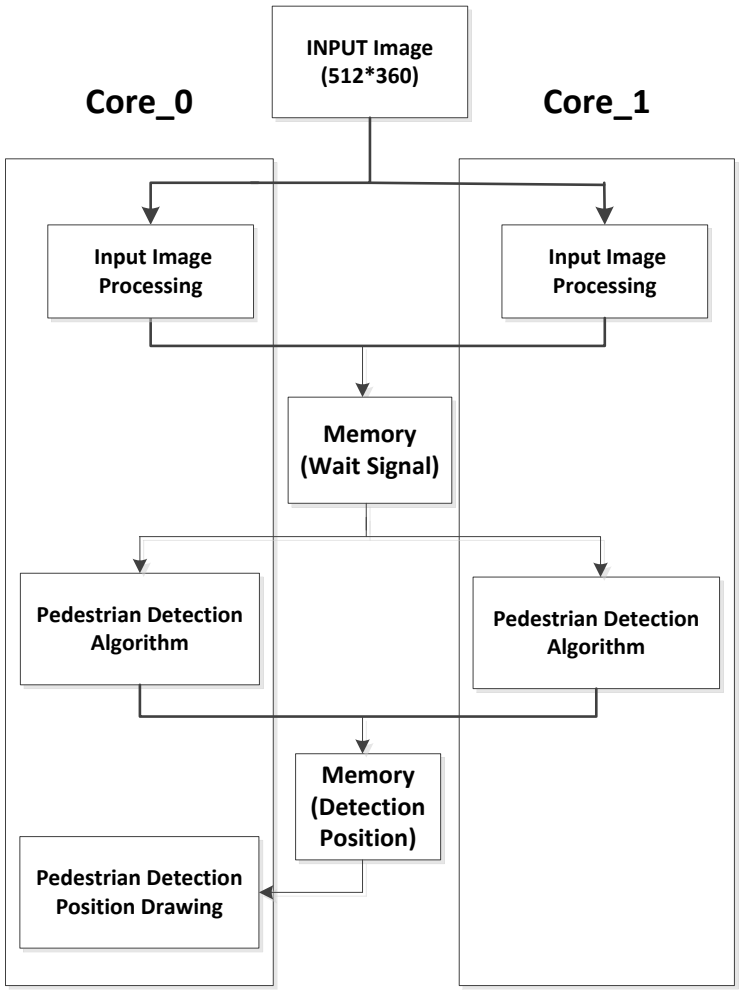

Figure 10. Shared Memory for Parallel Processing

In addition, as shown in Figure 11, each core reads the image and processes one half of the image. The feature points detected as pedestrian in Core_1 are saved in the shared memory between the cores. The feature points of the part detected as pedestrian in Core_0 and the feature points detected in Core_1 in the shared memory are added up to detect a pedestrian in the image. Then a wait signal is sent to the shared memory so that Core_0 and Core_1 can process the same frame.

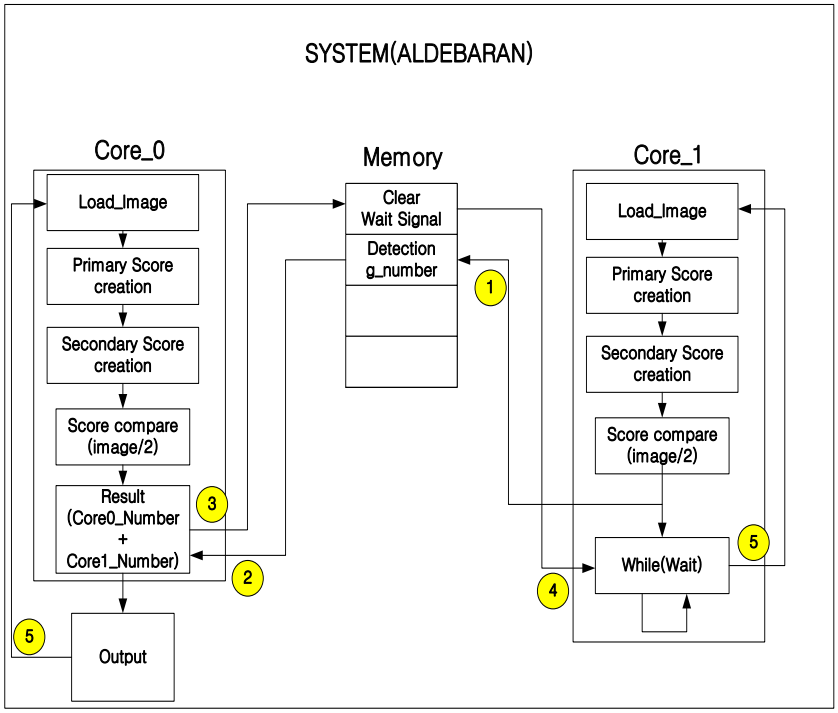

Figure 11. Parallel Processing on the ALDEBARAN Board

Figure 12 shows the resulting screen of the pedestrian detection algorithm implemented with dual core.

Figure 12. Resulting Screen of Pedestrian Detection Implemented with Dual Core

Table 2 shows the improved CENTRIST-based pedestrian detection algorithm implemented on the ALDEBARAN in single core and in dual core. The dual core computation showed about 40% improved performance.

Table 2. Comparison of Single Core and Dual Core

	Image Size	Frame Per Second
Single core	512x360	3.1 fps
Dual core	512x360	4.3 fps

When the improved CENTRIST-based pedestrian detection algorithm was implemented in dual core with a parallelization method appropriate for the ALDEBARAN board, the performance improved 1.4 times rather than 2 times. Because the pedestrian detection algorithm is processed in halves by two cores, the algorithm computation part is processed in parallel. However, the reading and outputting computations of the image data are processed only in Core_0, not in parallel. Furthermore, there is a wait signal to process the same frame because Core_0 and Core_1 work simultaneously. Due to this wait signal, wait time is generated in which Core_0 waits for the end of the computation of Core_1 and vice versa. This wait time is longer in Core_1 which has fewer computations than Core_0. To address this problem, the pedestrian detection algorithm in Core_1 is divided and separately computed. The reason for dividing the pedestrian detection algorithm is to reduce the wait time of Core_1 by processing the divided pedestrian detection algorithm during the wait time of Core_1. Furthermore, wait signals are added between the divided pedestrian detection algorithm so that Core_1 and Core_0 can process the same frame. Table 3 shows the result of improving the parallel processing by adjusting the wait signals of Core_0 and Core_1 and dividing the pedestrian detection algorithm. The performance improved by about 11% when the wait time in Core_0 and Core_1 was reduced for parallel processing. The dual core showed about 55% improved performance than the single core.

Table 3. Improved Performance of Dual Core

	Image Size	Frame Per Second
Dual core	512x360	4.3 fps
Improved dual core	512x360	4.8 fps

5. Conclusion

In this paper, to enable real-time processing of pedestrian detection in the embedded environment, an improved CENTRIST-based pedestrian detection algorithm that is more appropriate for the embedded environment compared to conventional pedestrian detection algorithms was implemented on an embedded board. The proposed algorithm was implemented in the ALDEBARAN board (300MHz) using input images with a size of 512x360 pixels. The improved CENTRIST-based pedestrian detection algorithm showed a performance of 3.1 frames per second. Furthermore, for real-time processing in the embedded environment, the CENTRIST-based pedestrian detection algorithm was parallelized and implemented in dual core. As a result, the dual core showed a performance of 4.3 frames per second which has improved by about 40% over the single core. Furthermore, the wait time in Core_0 and Core_1 was reduced to improve the performance of dual core. As a result, the proposed system showed a performance of 4.3 frames per second which has improved by

11%. Most pedestrian detection algorithms that are applied to various areas are implemented in PCs due to a large amount of computations. However, as the pedestrian detection algorithm is being applied to more and more areas, the need for pedestrian detection technology in the embedded environment is increasing. The pedestrian detection algorithm that can perform real-time processing and is appropriate for the embedded environment is expected to be applicable to a wide range of areas including smart cars, pedestrian auto licensing signal lights, security systems, and pedestrian accident prevention systems.

Acknowledgment

This research was supported by Seokyeong University in 2012.

References

[1] J. Wu and J. M.Rehg, "CENTRIST: A Visual Descriptor for Scene Categorization. IEEE Transactions on Pattern Analysis and Machine Intelligence, TPAMI, vol. 33, (2011), pp. 1489-1501.
[2] Y.-S. Hwang, J.-C. Kwak, K.-Y. Lee, "Implementation of a Pedestrian Detection Device based on CENTRIST for an Embedded Environment", Advanced Science and Technology Letters Embedded Ubiquitous , SERSC, vol. 46, **(2014)**, pp. 123-127
[3] P. Viola, Michael Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, **(2001)**, pp. 511-518.
[4] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection. IEEE Computer Society Conference on Computer Vision and Pattern Recognition", vol. 1, **(2005)**, pp. 886-893.
[5] Q. Zhu, S. Avidan, M.-C. Yeh and K.-T, "Cheng. Fast Human Detection Using a Cascade of Histograms of Oriented Gradients", IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, **(2006)**, pp. 1491-1498.
[6] P. Dollar, Z. Tu, P. Perona and S. Belongie, "Integral Channel Features", British Machine Vision Conference, **(2009)**.
[7] N. Kazakov, M Margala and, N.G Durdle, "Sobel Edge Detection Processor for a Real-time Volume Rendering System, Proceedings of the International Symposium on Circuits and Systems, vol. 2, **(2004)**, pp. 913-916.
[8] C. Messom and A. Barczak, "Stream Processing of Integral Images for Real-Time Object Detection", Ninth International Conference on Parallel and Distributed Computing, Applications and Technologies, **(2008)**, pp. 405-412.
[9] R. Zabih and J. Woodfill, "Non-parametric Local Transforms for Computing Visual Correspondence", Proceedings of the European Conference on Computer Vision, **(1994)**, pp. 151-158.
[10] A. Satpathy, X. Jiang and H.-L. Eng, "Extended Histogram of Gradients Feature for Human Detection", IEEE International Conference on Image Processing (ICIP), **(2010)**, pp. 3473-3476
[11] P. Dollar, C. Wojek, B. Schiele and P. Perona, "Pedestrian Detection: A Benchmark", IEEE Conference on Computer Vision and Pattern Recognition, **(2009)**, pp. 304-311.

Author

Kwang-Yeob Lee
1985. 8 Seogang University, Dept. of Electronics Engineering (BS)
1987. 8 Yonsei University, Dept. of Electronics Engineering (MS)
1994. 2 Yonsei University, Dept. of Electronics Engineering(Ph.D)
1989~1995. 2 Hyundai Electronics Inc., Senior Researcher
1995. 3~ Seokyeong University, Dept. of Computer Engineering, Professor
<Research interests>
Microprocessor, Embedded System, 3D Graphics System

