A Moving Object Counting Algorithm Implemented on Analog CNN and DSP Microprocessors

Fethullah Karabiber
Department of Computer Engineering, Istanbul University, 34320 Istanbul, Turkey
fetullah@istanbul.edu.tr

Serhat Pabuccuoglu
Istanbul University, Faculty of Veterinary Medicine, Department of Reproduction and Artificial Insemination, 34320 Istanbul – Turkey
serpab@istanbul.edu.tr

Sabri Arik
Department of Computer Engineering, Istanbul University, 34320 Istanbul, Turkey
ariks@istanbul.edu.tr

Abstract

In this study, we have implemented a moving Object Counting algorithm using Bi-i Cellular Vision System built mainly on Cellular Neural /Nonlinear Networks (CNNs) type named as ACE16k and Digital Signal Processing (DSP) type microprocessors. In the implemented algorithm, we have used the Adaptive Morphology to segment the video sequences. Once detecting the moving objects, we have recognized and counted the object using some certain features of the objects such as area, major axis, minor axis and eccentricity. The performance of the algorithm is evaluated using visual inspection and timing analysis. The experimental results have shown that Bi-i Cellular Vision System proved to be qualified to implement image processing algorithms in real time.

1. Introduction

Cellular Neural/Nonlinear Networks (CNNs) theory proposed in [1] have advanced properties for image processing applications. It is very useful for developing image processing applications by exploiting the advanced computational capabilities offered by the CNN Universal Machine (CNN-UM) [2]-[3]. However, it is possible to implement image processing algorithms on Bi-i standalone ultra high speed vision system, which can operate in real time and at high speeds. Bi-i Cellular Vision have high resolution sensors, Cellular Neural Network type (ACE16k) and Digital Signal Processors (DSP) type microprocessors and communication processors [4].

Analysis of cell movement plays an important role in some fields of biology and medicine. A typical example of dealing with this problem is provided by studies on sperm motility [5]. Since Sperm motility is used to assess potential fertility of male (human or animals); the measurement of sperm motility has been a major focus of basic and clinical sperm evaluation for over 25 years. Computer-assisted sperm analysis (CASA) systems have been commercially available since the mid-1980s. The goal of CASA systems is to obtain objective data on sperm motility that can be used in research, human fertility clinics, and animal breeding programs. Many researches were published about CASA systems [6].

In this study, we will give the first results of the Moving Object Counting System (MOCS) developed by using the capability of Bi-i Cellular Vision System. Specifically, we try to count the number of the moving animal sperms. In this application, we try to calculate the number of the moving sperms using Bi-i Cellular Vision System.

This paper is organized as follows: general concepts of CNN and Bi-i Cellular Vision System are given in Section 2. Moving Object Counting algorithm is given in Section 3. In Section 4, the experimental results are reported. Concluding remarks are given in Section 5.

2. CNN basics and Bi-i cellular vision system

In this section, we will give the basics of CNN and Bi-i Cellular Vision System. Firstly, the basic concepts of the CNN theory are presented in Subsection 2.1. Then, Subsection 2.2 gives the general information about the CNN Universal Machine. Then, hardware structure of Bi-i Cellular Vision System is described in Subsection 2.3. Finally, in Subsection 2.4, we will show how an algorithm can be implemented and run on the Bi-i Cellular Vision System.

2.1 Cellular neural networks

Cellular Neural Networks (CNNs) derived from Hopfield Neural network is introduced in [1]. The two most fundamental components of the

CNN paradigm are: the use of analog processing cells with continuous signal values, and local interaction within a finite radius.

The structural design of CNN is formed with basic circuits called cell. Each cell containing a linear capacitor, a non-linear voltage controlled current source, and a few resistive linear circuit elements is connected to its neighboring cells; therefore direct interactions take place only among adjacent cells. Mathematical expression of the standard CNN model is described by the following set of linear and nonlinear differential equations associated with the cells in the circuit.

$$C\frac{d}{dt}x_{ij}(t) = -\frac{1}{R}x_{ij}(t) + \sum_{C(k,l)\in S_r(i,j)} A(i,j;k,l)y_{kl}(t) + \sum_{C(k,l)\in S_r(i,j)} B(i,j;k,l)u_{kl}(t) + z_{ij} \quad (1)$$

$$y_{ij} = f(x_{ij}) = \frac{1}{2}\left|x_{ij}+1\right| - \frac{1}{2}\left|x_{ij}-1\right|$$

where x_{ij} , u_{ij} and y_{ij} are the state, input and output voltage of the specified CNN cell, respectively. The notation *ij* refers to a grid point associated with a cell on the 2D MxN grid, and $C(k,l)\in S_r(i,j)$ is a grid point in the neighborhood within the radius r of the cell *ij*. $A(i,j;k,l)$ and $B(i,j;k,l)$ represent the linear feedback and the linear control, respectively. The constant z_{ij} is the cell current, which could also be interpreted as a space-varying threshold. The output characteristic *f* which is only nonlinear element in each cell is a sigmoid-type (e.g. piecewise-linear) function. Without loss of generality, linear resistor (R) and linear capacitor (C) values can be set to 1. The block diagram of a cell C(i, j) is shown in the Figure 1.

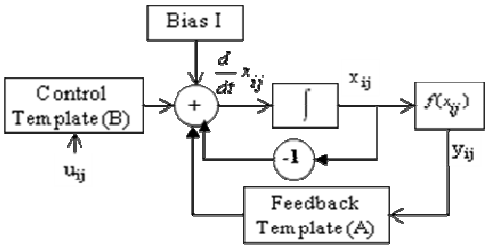


Figure 1. The block diagram of a cell C(i, j)

In this way CNNs have provided an ideal framework for programmable analog array computing. It means that the CNN can be used as a programmable device where the instructions are represented by the templates which define the connections between a cell and its neighboring cells. In general, the CNN templates consists of the feedback template (B), control template (A) and bias value. Basically, three different images can describe a CNN layer, that is, the input U, the state X and the output Y.

2.2. CNN universal machine

The hardware implementation of the CNN is easier because of the cell structure and local interactions in comparison to classical Neural Networks. Analogic Cellular Engines (ACE4k and ACE16k [10]) are designed based on the CNN Universal Machine (CNN-UM) architecture. The architecture of the CNN-UM is able to combine analog array operations with logic operations, therefore named as analogic computing by Roska and Chua [2]. A global programming unit is added to the array, and to make it possible an efficient reuse of intermediate results. In order to avoid input connections with external word on CNN-UM, the integration of an array of sensors on the system was proposed. Furthermore each computing cell was extended by local memories. It also has Global Analogic Program Unit (GAPU) to execute the program efficiently and control the Units effectively.

Referring to the ACE16k (Focal Plain Array Analog Processor), we point out that a full description can be found in [10]. Herein, we only recall that it contains a low resolution (128*128) CMOS grayscale image sensor and an analog processor array. All processor units are connected to the neighboring processors with specified weight. These weights represent a general analog program (template). This processor array is more useful for several image processing operations, than the traditional processors, because it processes the whole image in parallel. ACE16k chip is capable of capturing and processing 128x128-pixel images at extremely high speeds, or can act as a coprocessor.

2.3. Bi-i cellular vision system

Recently, a Bio-inspired (Bi-i) Vision System Architecture has been introduced, which combines ACE16k and DSP type microprocessors [4]. Its algorithmic framework contains several feedback and automatic control mechanisms among the different processing stages. In particular, this paper exploits the Bi-i Version 2 (V2), which has been described in detail in reference [4].

Bi-i Cellular Vision System has a color (1280*1024) CMOS sensor array (IBIS 5-C), two high-end digital signal processors (Embedded Texas Instruments C6415 fixed point DSP running at 600 MHz, Embedded Texas Instruments C6711 floating-point DSP running at 150 MHz) and

Embedded Etrax communication processor which is capable of a 100 Mbit/sec information exchange over TCP/IP with some external interfaces (USB, FireWire and a general digital I/O, in addition to the Ethernet and RS232). With this features, Bi-i is a family of high speed, compact, standalone and industrial vision systems.

2.4. The Bi-i programming

In CNN-UM, there are different ways of programming. One is AMC (Analogic Macro Code) language. This is the conventional way of Bi-i programming suitable for small applications. The other way is The Bi-i Software Development Kit (SDK). This is a collection of software libraries to be used when programming the Bi-i in C++. The SDK can be used, if one needs a large, complex, thoroughly optimized Bi-i application, which is beyond the scope of AMC programming.

Bi-i Cellular Vision System also contains Digital Signal Processor (DSP) unit. The InstantVision libraries can be used with DSP from Texas Instruments. InstantVision libraries are portable to other platforms as well. The software library contains Basedata Structures which describes the data structures, Interface for the Input/Output, The Signal and Image Processing Library, Feature Classification Library, Multi target tracking library.

The Code Composer Studio application suite from Texas Instruments is used to implement applications in hardware. General C++ programming skills are required. We use Bi-i SDK and InstantVision libraries to implement the algorithms on the Bi-i Cellular Vision System. IVRun which is a graphical environment for running InstantVision based applications displays the pictures and text messages in separate windows [9].

3. Moving object counting algorithm

We have developed an image processing based moving object counting system using Bi-i Cellular Vision System. The block diagram of the algorithm is given in Figure 2.

In the implementation process of the algorithm, first, the input image is loaded from the video sequences obtained by using video captured from a variety of microscopes and video cameras. In order to remove the noises, a Low Pass Filter is used. The low pass filtered image (Y_i^{LP}) is given in Figure 3a.

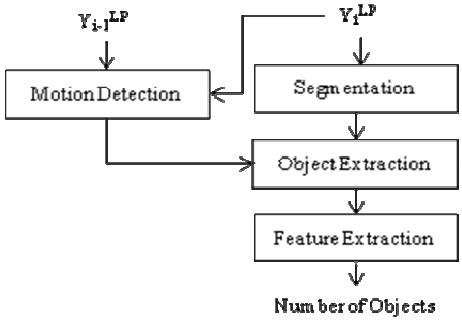
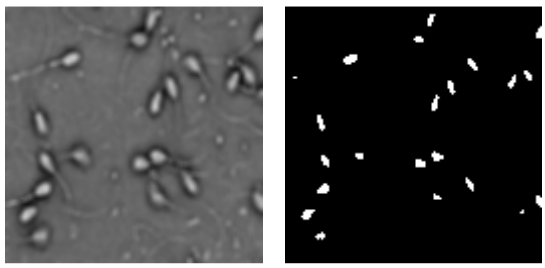


Figure 2. Block Diagram of Moving Cell Counting Algorithm

3.1. Segmentation

Segmentation is the most crucial part of moving object counting algorithm. Segmentation is the process of representing a digital image into multiple meaningful regions, which would make it easy to analyze the images to be processed. The development of segmentation algorithm is necessary in wide range of applications such as target tracking, object classification, pattern recognition, MPEG coding. The segmentation algorithms can be divided basically into two categories – edge based and region growing [7-8]. Edge based segmentation algorithms are based on discontinuity property of gray levels. Similarity is the main property exploited by region growing segmentation algorithms. In addition, using threshold techniques we can segment the image in simply way.

Here, we have performed a segmentation algorithm based on the threshold and region growing. In the first step of the proposed segmentation algorithm, the threshold operation is applied to obtain the binary image using *ConvLAMtoLLM* function on the ACE16K chip. This function (included in the SDK) converts a grey-level image stored in the Local Analog Memory (LAM) into a binary image stored in the Local Logic Memory (LLM) [9]. The output of the threshold operation is given in Figure 3b.



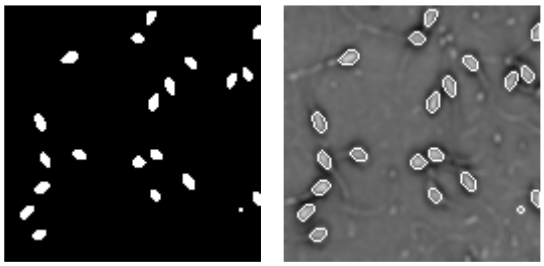
(a) (b)

Figure 3. (a) Low pass filtered image (b) Threshold

As it can be seen in Figure 3b, due to the low image quality and selected threshold level; some

objects cannot be extracted in full. In order to obtain the objects precisely, we apply the adaptive morphology function defined in the InstantVision library.

The adaptive morphology is a special type of dilation controlled by a grayscale picture. It is like the water flow, which has a source and is controlled by the landscape. In this context the 'source' is the input binary picture and the 'landscape' is the grayscale control picture. The output is either the state of the water flow at a moment or the final state of the flow when the water cannot flow any further because everywhere is bordered by hills and mountains and higher landscape. This adaptive morphology can be used effectively in image segmentation and edge detection processes for a large class of image models [9]. The result of the Adaptive Morphology function using Figure 3b as the input image and Figure 3a as the control image is depicted in Figure 4a. Finally, we have obtained Figure 4b to show the accuracy of the segmentation algorithm.



(a) (b)

Figure 4. (a) Segmented objects (b) Final result of segmentation

3.2. Motion detection

The block diagram of the implemented motion detection algorithm is given in Figure 5.

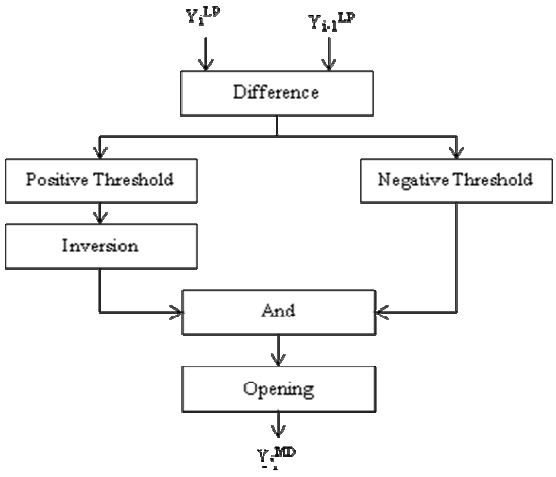
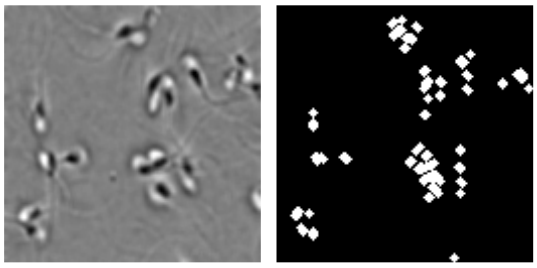


Figure 5. Block diagram of the motion detection

The first step consists of computing the difference between the current frame (Y_i^{LP}) and the preceding frame (Y_{i-1}^{LP}). Difference image is given in Figure 6a. Then, the positive and negative threshold operations are applied to the difference image via the *ConvLAMtoLLM* function. Since the zero (0) value in the ACE 16k chip corresponds to 127, the values of threshold levels for the positive and negative threshold operations are selected as 132 and 122, respectively. Successively, the logic AND operation is applied between the output of the positive threshold and the inverted output of the negative threshold. The resulting image includes all the changed pixels.

Finally, the *Opening* function (running on the ACE16K) is used for deleting small objects. Note that Opening function uses *erosion* and *dilation* functions to delete small objects and to smooth the image. The output of the algorithm is the motion detection (*MD*) mask Y_i^{MD}, which entirely preserves the moving objects. Figure 6b shows the motion detection mask.

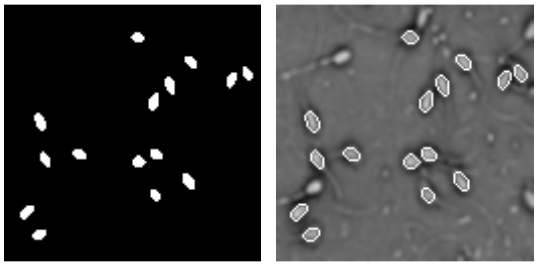


(a) (b)

Figure 6. (a) Difference of successive video sequences (b) Motion Detection Mask

3.3. Object extraction

After obtaining the motion detection mask we need to extract the moving objects. In order to find only moving objects we use the *recall* function defined in Bi-i SDK for ACE16k using the segmentation results as input and motion detection mask as mask image. Figure 7a shows only the moving objects. Figure 7b is obtained to represent the extracted moving objects.



(a) (b)

Figure 7. (a) Moving Objects (b) Extracted Moving Objects

3.4. Feature extraction

Feature extraction is a conversion of the visual information of a binary picture to numeric format. It is based on indexing the connected sets of active pixels ("objects") on the picture, and then calculate some features (like area, length, eccentricity etc.) of each one [9].

In order to extract the features of detected moving objects, we use the *FeatureExtraction* routines of the InstantVision Signal and Image Processing Library [9]. The functions find the number of objects and extract various features from input binary images. Here, we use the following features to recognize the objects.

- **Area:** The number of pixels belonging to the object
- **Major Axis length:**The length of the major axis of the ellipse
- **Minor Axis length:** The length of the minor axis of the ellipse
- **Eccentricity:** Ratio of the focus distance and the major axis length of the ellipse. A circle has an eccentricity of 0, a line 1

4. Experimental results

Minimum, maximum and average of these features are obtained using *FeatureStatistics* function defined in InstantVision to decide the range of the features. The results are given in Table 1. Using the statistical properties of the data as the control parameters to recognize the objects, we have counted the number of moving objects.

Table 1. Some features of the objects

Features	Minimum	Maximum	Average
Area	23	39	29
Major Axis	6.53	9.58	8.04
Minor Axis	4.19	5.90	4.86
Eccentricity	0.53	0.86	0.78

The main objective of this study is to count moving objects. To guarantee the accuracy of the results, we have obtained the average number of the moving objects in 10 frames. In this video sequences, average of the segmented and moving objects is 21 and 14, respectively.

The execution times of the algorithm are given in Table 2. Firstly, we have implemented the overall algorithm using only DSP. In this case, execution time is 65818 µs. In order to improve the performance of the algorithm, we have implemented the some possible parts of the algorithm on ACE16k. If we implement Motion Detection and Object Extraction parts on ACE16k, execution time of them decrease approximately 20 times. The execution time of the overall operation using both DSP and ACE16k is 5540 µs faster than only DSP.

Table 2. Execution time of the algorithm

Operation	DSP	DSP+ACE16k
Segmentation	58207 µs	58207 µs (DSP)
Motion Detection	4921 µs	469 µs (ACE16k)
Object Extraction	1140 µs	52 µs (ACE16k)
Feature Extraction	1550 µs	1550 µs (DSP)
Total	**65818 µs**	**60278 µs**

5. Conclusion

We have implemented a Moving Object Counting System (MOCS) using Bi-i Cellular Vision System. Proposed segmentation algorithm based on threshold and adaptive morphology is implemented on DSP. Then, Motion Detection and Object Extraction parts have been implemented on both DSP and ACE16k chip. We have counted the number of the moving objects extracting the some features of the objects. Experimental results show the efficiency of the algorithm. The execution times prove the high computational capability of the ACE16k chip. Furthermore, Bi-i Cellular Vision System is rather qualified to implement image processing algorithms in real time. In the future, we will tackle to the problem of tracking the objects and analyzing their trajectories.

6. References

[1] L. O. Chua and L. Yang, " Cellular neural networks: Theory and applications", *IEEE Trans. on CAS*, vol. 35 no. 10, 1988, pp.1257–1290

[2] T. Roska and L. O. Chua, "The CNN universal machine: an analogic array computer", *IEEE Trans. on CAS-I*, vol. 40 no.3, 1993, pp. 163–173

[3] T. Roska and A. Rodriguez-Vazquez, "Towards visual microprocessors", *Proceedings of the IEEE*, vol. 90 no.7, 2002, pp. 1244–1257

[4] A. Zarandy and C. Rekeczky, "Bi-i: a standalone ultra high speed cellular vision system.", *IEEE Circuit and Systems Magazine,* vol. 5, no.2, 2005, pp. 36–45

[5] W. Warchol, J. B. Warchol, K. Filipiak, Z. Karas and F. Jaroszyk, "Analysis of spermatozoa movement using a video imaging technique" , *Histochemistry and Cell Biology,* vol.106, No. 5, 1996

[6] R. P. Amann, and D. F. Katz, "Reflections on CASA after 25 years," *Journal of Andrology*, vol. 25, 2004

[7] C. Gonzales and R.E. Woods, "*Digital Image Processing*", Prentice Hall, New Jersey, 2002

[8] T. Acharya and A.K. Ray, "*Image Processing: Principles and Applications*", Wiley and Sons, 2005

[9] http://www.analogic-computers.com /Support

[10] A.R. Vazquez, G. L. Cembrano, L. Carranza, E.R.Moreno, R.C. Galan, F.J. Garrido , R.D. Castro and S. E. Meana, "ACE16k: the third generation of mixed-signal SIMDCNN ACE chips toward VSoCs", *IEEE Trans. CAS-I*, vol.51,no.5, 2004, pp. 851– 863