Parallel Extraction Architecture for Image Moments of Numerous Objects

Yoshihiro Watanabe, Takashi Komuro, Shingo Kagami and Masatoshi Ishikawa
Department of Information Physics and Computing,
University of Tokyo,
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan

***Abstract*— In this paper, we propose a new architecture that can extract information of numerous objects in an image at high-speed. Various characteristics can be obtained from the image moments. The proposed architecture simultaneously extracts the moments of multiple objects in parallel. This parallel extraction enables a significant reduction in the amount of calculation required. In addition, asynchronous operation allows fast processing. We believe that our architecture can obtain more information in real-time even at high frame rates, providing advantages in a wide range of applications, mainly for image measurement. This paper describes our proposed architecture and some results on its implementation in FPGA.**

I. INTRODUCTION

Image measurement is an important technique used in various applications; its advantages include contactless measurement and highly flexible operation. In particular, we focus on the development of applications such as robotics, multimedia, and inspection. In these applications, real-time operation is required; that is, measurement data must be obtained from input images within one frame period, and the system must quickly respond to changes in the environment.

In addition to data acquisition in real-time, it is necessary to improve the frame rate in order to quickly respond to changes and to observe high-speed phenomena. This can be achieved by introducing a system called high-speed vision [1]–[3]. High-speed vision achieves high-speed imaging at frame rates considerably higher than the conventional video frame rate; in the NTSC system, for example, the frame period is 33 ms.

The improvement of the frame rate by the high-speed vision system and real-time data acquisition are powerful elements for developing new applications. However, the amount of calculation in most visual processing applications is huge. Thus, it is very difficult to achieve real-time operation even at conventional video-signal frame rates. This is a serious constraint on the performance of applications. To overcome this limitation, we consider the introduction of a new architecture in order to ensure data acquisition in real-time even at high frame rates.

In this paper, we focus on image moments to be measured. From the image moments, we can obtain various characteristics, such as the size, centroids, posture, and so on of individual regions. Image moments are very effective for image analysis and pattern recognition, as has previously been described [4]. The architecture that we propose enables the extraction of information of numerous objects, even at high frame rates. Such processing is expected to be practical in a wide range of applications. Fig. 1 shows the concept overview of our study.

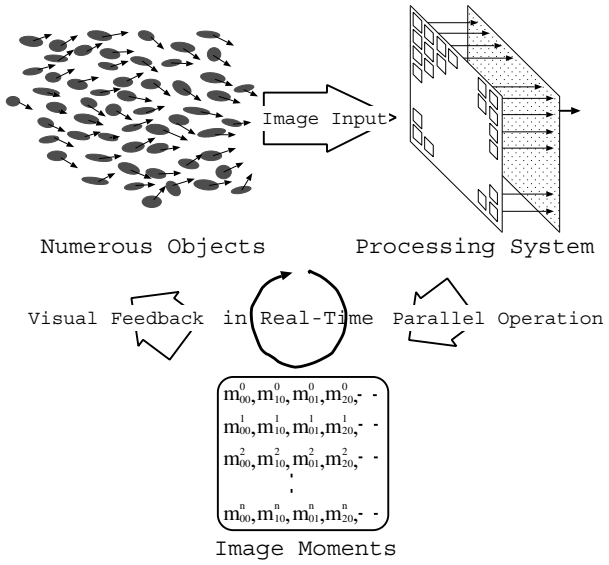

Fig. 1. Concept Overview

As one such application, we consider particle image velocimetry (PIV) [5] in real-time. PIV is an image measurement technique in which a flow field is observed by analyzing information of numerous particles. PIV is conventionally carried out as batch processing. In this paper, we use particle information obtained from moments for the analysis. Real-time PIV is expected to realize new applications based on fluid and motion measurement. In addition, we consider other applications, such as, for example, precise microscope observation of multiple protozoa moving at high-speed, image recognition for robot control, and inspection.

We propose a new architecture that can be a platform of such applications. In our proposed architecture, the moments of multiple regions are simultaneously extracted based on parallel operation. This parallel extraction enables a significant reduction in the amount of calculation required. In addition, this architecture increases the processing speed by means of asynchronous operation. In the remainder of this paper, we

Proceedings of the Seventh International Workshop on Computer Architecture for Machine Perception (CAMP'05)
0-7695-2255-6/05 $20.00 © 2005 IEEE

first describe the architecture and its performance, followed by some results from its implementation in FPGA.

II. Background

The $(i+j)$th moments of an image $\boldsymbol{I}$ are calculated from equation (1) below.

$$m_{ij} = \sum_x \sum_y x^i y^j I(x,y) \tag{1}$$

Here, $I(x,y)$ is the value at pixel (x,y). From the image moments, we can obtain various features, such as the size, centroid, orientation, rough shape information, and so on. However, as shown in equation (1), moment extraction needs a large amount of calculation because it uses information of the entire image. In order to carry out moment extraction at high speed, a suitable new processing system is required.

The problem of the large amount of calculation for moment extraction can be solved by using parallel operation. A drastic reduction in calculation time can be achieved by introducing parallel processing and circuits for summation [6]. A device called a programmable digital vision chip [7] adopts this moment extraction architecture and can obtain moments from an input image even at frame rates as high as 1 kHz. This device has been successfully applied to robot control, microscope observation, inspection, and so on.

In robot control, in order to realize various tasks, it is important for a system to quickly respond to rapid changes in the environment. This can be achieved by analyzing the environment at high frame rates. There have been applications such as catching [8] and batting [9] developed by using high-speed moment extraction.

In applications using a microscope, it is difficult to observe objects with high precision because protozoa such as paramecia move very quickly. It is expected that this problem can be overcome by introducing a high-speed vision system such as a vision chip, which enables high-frame-rate imaging and high-speed processing. A tracking system for a single paramecium under a microscope has been developed [10]. Also, there has been an application using this system [11].

In inspection such as industrial products and crops, the required time can be reduced considerably by using high-speed measurement.

The results of those applications show that moment extraction based on parallel operation is an effective method for high-speed processing. High-speed moment extraction has other important advantages. However, until now, these applications have been based on a single target observation; the vision-chip architecture deals with an image including only one target region. Therefore, when an input image has multiple regions, it is necessary to divide the image into separate images each including one target. After this operation, the moment extraction is applied to each image one by one, as depicted in Fig. 2. This causes an increase in the amount of calculation proportional to the number of targets. This is a critical problem in terms of performance when the image has numerous objects.

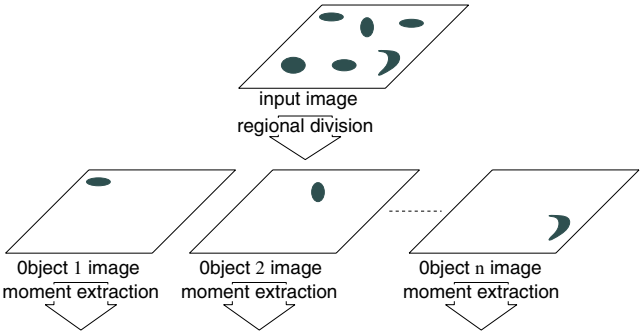

Fig. 2. Conventional Method of Separating Image with Multiple Objects

III. Parallel Extraction Architecture

We focus on moment extraction of numerous objects in real-time even at high frame rates. As we mentioned above, there has been a novel method for moment extraction of a single object based on parallel operation. However, it occurs a crucial problem to numerous objects.

On the other hand, a processing system based on parallel operation must be able to extract information about multiple objects simultaneously by separating the processing in the array according to an input pattern. In this paper, we call this operation parallel extraction. We can reduce the amount of calculation significantly by using this parallel extraction.

Thus, our proposed architecture enables extraction of information of numerous objects by integrating two approaches, namely, moment calculation based on the conventional method and parallel extraction. In this section, we describe the details of this architecture.

A. Basic Concept

First, we briefly describe conventional moment extraction based on a parallel system. In a parallel processor array, the moment of each column is calculated bit sequentially from the low bit to the high bit. Outside of the array, the processing elements receive the result of each column calculated in parallel and output the moment of the entire image. This architecture can also obtain the high-order moments only by summation because the calculation is carried out bit sequentially, while selecting the proper pixel data.

The parallel extraction enables simultaneous moment extraction of multiple objects by using the pixel parallel operation. Fig. 3 shows the basic concept of the parallel extraction. In this figure, the calculation of each column is performed in parallel within the array. As shown in this figure, if the multiple objects do not overlap in the column direction, the calculation of those objects is carried out correctly. Therefore, the moment extraction of the objects at such locations can be executed simultaneously.

The calculation results from the array can be divided into a result for each object by using the space between the objects. Also, while the calculation within the array is carried out bit sequentially, the calculation outside the array has

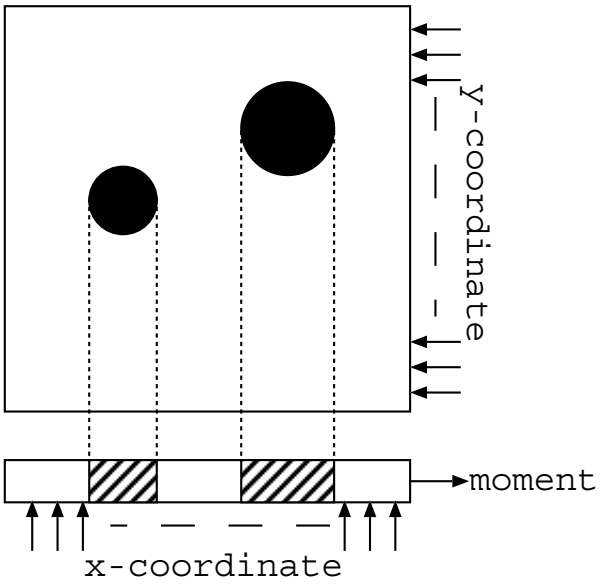

Fig. 3. Parallel Extraction

no constraints, which allows the external calculation to be executed at high speed. Thus, we assume that the required time outside the array is negligible compared to the time within the array. This means that the total calculation time for moments depends on the time within the array using parallel extraction.

The parallel extraction enables high-speed calculation of the moments based on the conventional method and enables a significant reduction in the amount of calculation, which was previously proportional to the number of objects. However, the parallel extraction needs preprocessing to select, from any pattern, targets that do not overlap in the calculation direction. The selection algorithm is described in the next section. Also, in section III-C, we show some results, through simulations, on the actual amount of calculation reduction achieved by the parallel extraction.

B. Selection of Targets for Parallel Extraction

In this section, we describe the selection algorithm for the parallel extraction. In order to achieve selection, we need to know the locations of all objects in an image. This operation takes a long time because it requires a scan of the entire image. However, the selection process must be finished in about the same time as the moment extraction to maintain the speed advantage of the parallel extraction. In order to overcome this problem, the processing can be executed based on a pixel parallel algorithm. Also, we consider making the processing faster by asynchronous operation.

The selection algorithm in this paper consists of iterative operations, which ensures convergence of the results after a certain number of iterations. The algorithm described by such iterative operations can be executed without clock synchronization: communication between pixels is carried out asynchronously. Therefore, the processing speed is determined only by the average delay of signal transmission. This enables the processing to be executed at high-speed, even the target region covers a wide area.

At first, we explain the assumptions used in the selection algorithm. Input images for this algorithm are assumed to be binary. Also, any nonzero 4-neighbor pixels, that is, up, down, left and right pixels, are assumed to be within the same object. In addition, the algorithm includes a definition about shape. In the algorithm, all objects are divided into two groups, concave objects (in the calculation direction) and other objects. The selection algorithm includes a step standardizing the shape of all objects by making concave objects convex. This step avoids complex processing caused by differences in shape, except in the case where the algorithm sees a concave object and an object overlapping in its concave area as a single object.

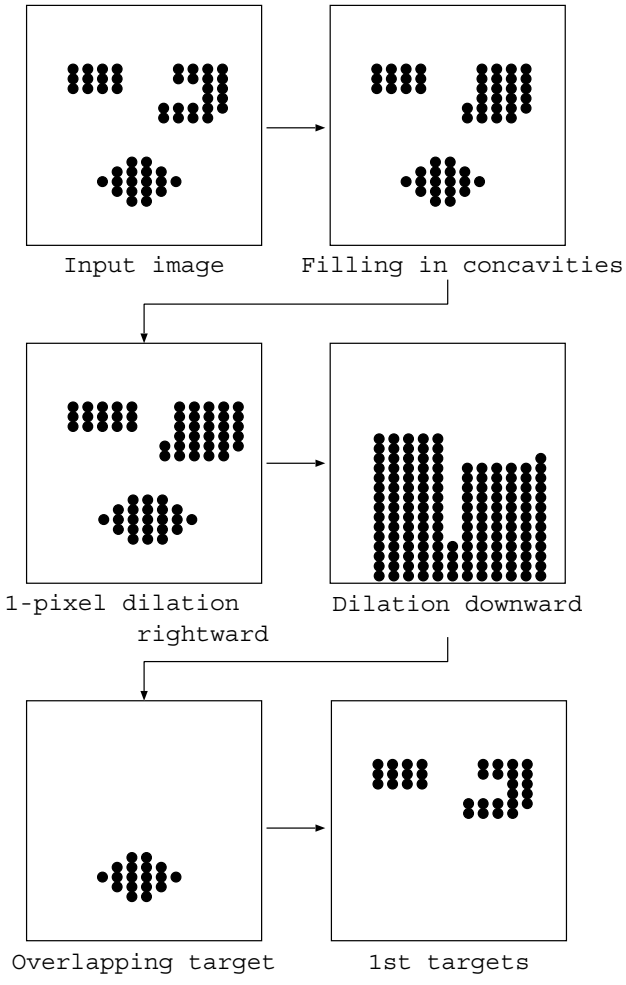

Fig. 4. Target Selection for Parallel Extraction

Fig. 4 shows the flow of the selection for the parallel extraction. The algorithm steps are described in detail below. In the equations, f is an input image. Terms in the equations

including the suffixes k and n indicate iterative operations. The suffix k shows iteration in a single equation and n shows iteration over a number of equations. Values without suffixes are the convergence results. In addition, when x and y are negative, values including arguments are zero.

Step 1. Filling in concavities

$$\begin{cases} g^0(x,y) = f(x,y) \\ g^{n+1}(x,y) = g^n(\underline{x,y}) \cup i^n(x,y) \\ \quad \begin{cases} h^n_{k+1}(x,y) = \overline{g^n(x,y)} \cap \{g^n(x,y-1) \cup h^n_k(x,y-1)\} \\ \qquad\qquad \cap\{g^n(x-1,y) \cup g^n(x+1,y)\} \\ i^n_{k+1}(x,y) = h^n(x,y) \cap \{i^n_k(x,y+1) \cup g^n(x,y+1)\} \end{cases} \end{cases}$$

Step 2. 1-pixel dilation rightward

$$d(x,y) = g(x,y) \cup g(x-1,y)$$

Step 3. Dilation downward

$$e_{k+1}(x,y) = e_k(x,y-1) \cup \{d(x,y-1) \cap \overline{d(x,y)}\}$$

Step 4. Extraction of overlapping targets

$$\begin{cases} j_0(x,y) = e(x,y) \cap d(x,y) \\ j_1(x,y) = f(x,y) \cap \{j_0(x,y) \cup j_0(x+1,y) \\ \qquad\qquad \cup j_0(x,y-1) \cup j_0(x,y+1)\} \\ j_{k+1}(x,y) = f(x,y) \cap \{j_k(x,y) \cup j_k(x-1,y) \\ \qquad \cup j_k(x+1,y) \cup j_k(x,y-1) \cup j_k(x,y+1)\} \end{cases}$$

Step 5. Selection of targets for parallel extraction

$$k(x,y) = f(x,y) \cap \overline{j(x,y)}$$

Step 6. Parallel extraction

–Apply parallel extraction to an image $\boldsymbol{k}$.

In step 1, the concavities of the concave objects in the column direction are filled in. In steps 2 to 4, the targets overlapping in the column direction are detected. In step 5, the algorithm selects targets for parallel extraction.

C. Calculation Reduction by Parallel Extraction

In this section, we show, through simulations, the reduction effect of the amount of calculation by the parallel extraction. In the simulations, we count the number of moment extractions for all objects in an image. The moment extractions in all dimensions are defined as one set. Therefore, this number corresponds to the number of times targets are selected for the parallel extraction. The image size is 256×256 pixel^2. The image patterns are generated by randomly setting the size and positions of circles within a radius of 6 pixels.

Fig. 5 shows an example of the parallel extraction. In this example, the pattern has 172 objects. In Fig. 5, black objects are the targets for parallel extraction. We can see that all operations for moment extraction are finished after only 12 iterations by parallel extraction, compared to 172 iterations conventionally.

Fig. 6 shows the relationship between the number of objects and the number of extractions. The plotted points are simulation results and the curve is an approximated line to the plotted points. This curve is assumed to be an exponential function passing through the points $(0,0)$ and $(1,1)$. Also, the

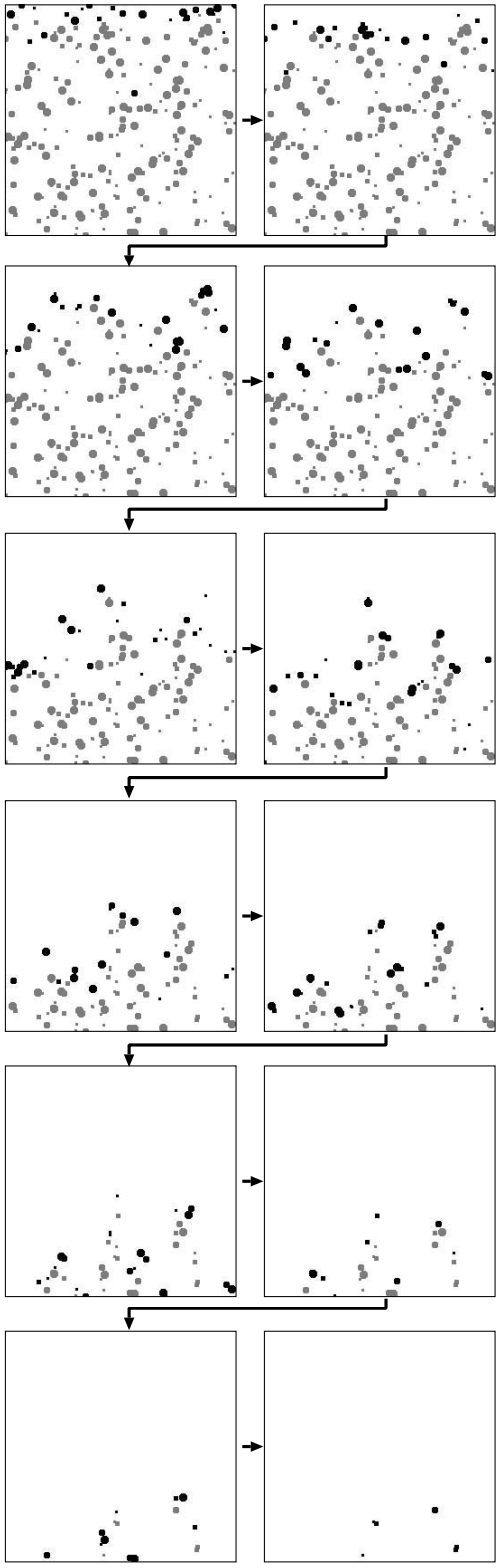

Fig. 5. Example of Parallel Extraction

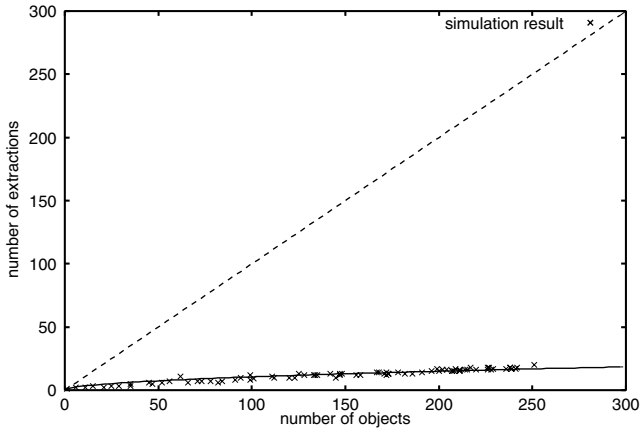

Fig. 6. Simulation Result

straight line shows the conventional case. When we define a and b as the number of targets and the number of extractions respectively, the curve is $b = a^{0.511}$ and the straight line is $b = a$. These results show that the parallel extraction enables a significant reduction in the amount of calculations.

IV. Implementation in FPGA

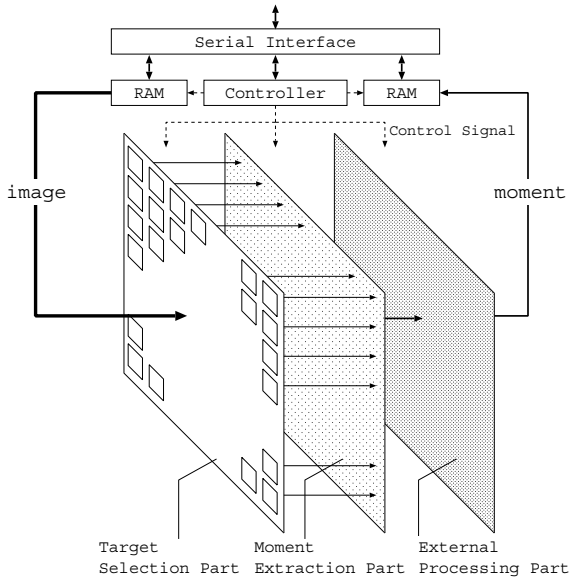

Fig. 7. Structure of Overall System

We implemented the proposed architecture in FPGA by using a hardware description language, Verilog-HDL. Fig. 7 shows the block diagram of the overall system. The system had three processing parts, the target selection part, the moment extraction part and the external processing part. The target selection part and the moment extraction part were implemented as the parallel processor array. In this implementation, the array could deal with 32 $\times$ 32 pixel data in parallel. Also, external processing part was implemented outside the array.

An target image for parallel extraction was generated thorough the target selection part from an input image. The moment extraction part received the target image and outputted the moments of each column. In the external processing part, the moments of the entire image were calculated. Data such as input images and results passed through a serial port between the FPGA and a computer. The each processing part is described in detail below.

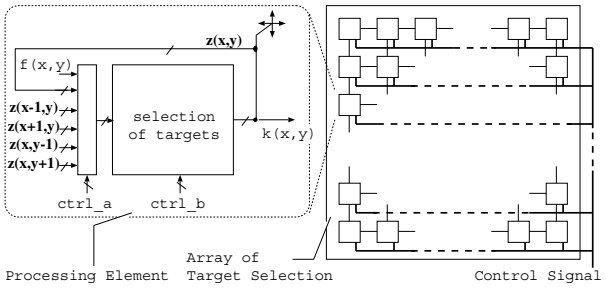

Fig. 8. Structure of Target Selection Part

Fig. 8 shows the array of target selection and its processing element. We implemented the processing element through the logic synthesis based on the algorithm in section III-B. In the implemented circuits, each pixel element communicated asynchronously with its neighborhood, which enabled high-speed processing.

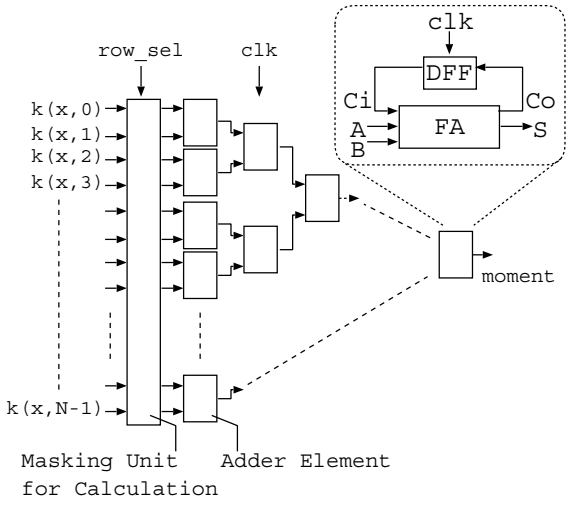

Fig. 9. Structure of Moment Extraction Part

Fig. 9 shows the moment extraction part of each column. We configure the array connecting elements as a binary tree. Each element consisted of a one-bit full adder and a flip-flop. The circuit received data from the preprocessing elements and outputted the column moment, while selecting the proper pixel

data through the masking unit. We also implemented the circuit for moment extraction outside the array as the same structure for the sake of simplicity, though this configuration had the room of improvement on the required performance. We could obtain multiple results simultaneously by providing several moment extraction parts and a circuit dividing the column data into the result of each target region. In this implementation, we provided the four parts. These circuits received the data from the column summation within the array and outputted the moments of targets.

This implemented system outputted the 0th and 1st moments. As shown in Fig. 3, the moments that need multiplication by the x coordinate can be calculated from the results of each column outside the array. Therefore, the required calculation within the array is only for the y coordinate. When we obtain the moments from 0th to 1st order, m_{00} and m_{01} are calculated within the array. m_{10} is calculated from m_{00}.

The FPGA device used was the Virtex-II Pro series XC2VP50 made by Xilinx. XC2VP50 contained 53,169 logic cells and 232 RAM blocks containing a total of 4176 kbit. The speed grade was -7 in this implementation. As a result, it took 38 % of all resources (slices) in XC2VP50 to implement the 32×32 array and the processing circuits outside the array.

The clock frequency provided to the circuits was set to 100 MHz. This clock controlled the summation operations. The clock period was determined by the time required for extracting the moment within the array. The clock period was nt in this configuration, where the image size was $2^n \times 2^n$ and the summation delay per element was t. We set n and t as 5 and 2 ns, respectively. For example, it required $(n+1)$ clock cycles to extract the 0th moment from the array. Therefore, it took $nt(n+1)$, assuming that we implemented the external processing part so that the processing time outside the array was negligible as mentioned in section III-A. Also, the total time required to generate a target image from an input image was shorter than the time to extract 0th and 1st moments in this implementation.

We confirmed that the implemented circuits selected the proper targets for parallel extraction and outputted the correct moments for any pattern. The circuit area can be reduced by changing the configuration for selection processing. Also, we can improve the clock frequency up to about 200 MHz ($t = 1$ ns)by optimizing the configuration.

There is room of improvement in the configuration of the architecture, considering the processing speed and the circuit area. However, the results from the experimental implementation in FPGA show that we can realize the proposed architecture as an expected platform of real-time image measurement.

V. Conclusion

We have proposed and designed a parallel extraction architecture for extracting information of numerous objects. We have also implemented it in FPGA and confirmed that it functions correctly. This architecture is capable of extracting a sufficient amount of information from images even when real-time processing at high-frame rates is required. Such processing is very useful in various applications.

As a next step, we will develop a system that connects a processor based on our architecture with a high-speed vision system. Also, we plan to demonstrate new applications using this real-time system, such as fluid and motion measurement based on PIV, precise microscope observation of numerous protozoa, image recognition for robot control, inspection, industrial applications, and so on.

References

[1] http://www.micron.com/products/imaging/
[2] http://www.dalsa.com
[3] http://www.fillfactory.com/
[4] M. K. Hu: "Visual Pattern Recognition by Moment Invariants," IRA Transactions on Information Theory, Vol. IT-8, pp. 179-187, 1962.
[5] M. Raffel, C. E. Willert, and J. Kompenhans: "Particle Image Velocimetry: A Practical Guide (Experimental Fluid Mechanics)," Springer Verlag, 1998.
[6] I. Ishii, T. Komuro, and M. Ishikawa: "Method of Moment Calculation for a Digital Vision Chip System," Proceedings of IEEE International Workshop on Computer Architecture for Machine Perception, pp.41-48, 2000.
[7] T. Komuro, S. Kagami, and M. Ishikawa: "A New Architecture of Programmable Digital Vision Chip," Proceedings of 2002 Symposium on VLSI Circuits, pp. 266-269, 2002.
[8] Y. Imai, A. Namiki, K. Hashimoto, and M. Ishikawa: "Dynamic Active Catching Using a High-speed Multifingered Hand and a High-speed Vision System," Proceedings of 2004 IEEE International Conference on Robotics and Automation, pp.1849-1854, 2004.
[9] T. Senoo, A. Namiki, and M. Ishikawa: "High-Speed Batting Using a Multi-Jointed Manipulator," Proceedings of 2004 IEEE International Conference on Robotics and Automation, pp.1191-1196, 2004.
[10] H. Oku, I. Ishii, and M. Ishikawa: "Tracking a Protozoon Using High-Speed Visual Feedback," Proceedings of 1st Annual International IEEE-EMBS Special Topic Conference on Microtechnologies in Medicine & Biology, pp.156-159, 2000.
[11] N. Ogawa, H. Oku, K. Hashimoto, and M. Ishikawa: "Motile Cell Galvanotaxis Control using High-Speed Tracking System," Proceedings of 2004 IEEE International Conference on Robotics and Automation, pp. 1646-1651, 2004.
[12] S. Kleinfelder, S. Lim, X. Liu and A. E. Gamal: "A 10000 Frames/s CMOS Digital Pixel Sensor," IEEE Journal of Solid-State Circuits, Vol.36, No.12, pp.2049-2058, 2001.