10-31-1996

Novel high-speed architecture for machine vision applications

Bassam S. Farroha

Raghvendra G. Deshmukh

PROCEEDINGS OF SPIE

SPIEDigitalLibrary.org/conference-proceedings-of-spie

Novel high-speed architecture for machine vision applications

Bassam S. Farroha
R. G. Deshmukh

SPIE.

A novel high-speed architecture for machine vision applications

B. S. Farroha

Astronaut Memorial Space
Science Center, BCC
P.O. Box 1274
Melbourne, FL 32902

R. G. Deshmukh

Electrical and Computer Engineering Dept.
Florida Institute of Technology
150 W. University Blvd.
Melbourne, FL 32901

ABSTRACT

This paper focuses on producing a state-of-the-art technique for designing an image recognition system for machine vision applications. The motivation behind the new system design is to provide a unique methodology, using strategic design techniques, to implement a system that addresses real-world image recognition applications. The introduction of application-specific, massively parallel array of processors, where low-level processing is accomplished on reconfigurable hardware structures, highlights the scheme. The system was built and simulated on a VLSI chip and results were verified using Electric Rules Check (ERC) and Harris Timing Analysis (HTA) examination tools. The system is composed of three functional layers and a main control unit. The top two layers are used for image loading and manipulation and the third layer is used for processing the pixel values. Each layer has a local control unit, while the main control unit oversees the operations of the whole system and synchronizes the processes.

A CAD designer tool was implemented to facilitate the design and reconfiguration of the low-level processing elements (PE). This tool has a modular library of processing elements and a logic verification algorithm. The current architecture of the chip was built to accommodate a 64 X 64 array with unlimited stackable characteristics to handle larger images.

This paper will present the top level design layers, the distributed control, and demonstrate that the proposed system is image complexity invariant. The massively parallel approach of transferring data and control signals and the processing of image data is presented. The system takes into consideration cost, size, speed, and reliability needs of today's applications.

Key Words: Real-time Vision, Robotic Vision, Machine Vision, Massive Parallel Processing.

2. BACKGROUND

For more than a decade, there have basically been two approaches to satisfying machine vision requirements: general purpose processing (GPP) and application specific processing (ASP). The majority of mainstream image-recognition systems concentrate on flexibility using GPP or high speed using ASP. However, a combination of these valuable characteristics have not been adequately addressed in one system. Machine vision systems using GPP typically are very large in size and require significant amounts of time, power, and memory to process images. Issues such as cache, memory bandwidth, and address modes have to be accounted for when implementing a GPP vision system. ASP vision systems are compact and use dedicated hardware to optimally process the image data. Dedicated hardware vision systems are far more efficient and deterministic. Speed and throughput can be easily calculated because the system was designed for a specific task and each component of the system is known, therefore, quantifiable. ASP offers inherent addressing, high speed with small latency, and the ability to expand and augment processing power without much additional effort. However, ASP can be cost prohibitive because the system can only be used for one task, it does not have the flexibility needed to migrate it for multiple uses. [1]

This research has taken into account both the advantages and disadvantages of GPP and ASP type systems and has produced a novel approach for designing an image recognition system that provides limited flexibility with

real-time responses. The system was implemented using massively parallel design techniques, ASP, and VLSI. The overall scheme divides the system into three distinct layers each with its own local control and one main control unit for the entire system. The system implements massively parallel communication links between the layers and the main control unit to provide for the real-time response. The first layer handles image loading and the second layer provides for data manipulation composed of reconfigurable arrays, which allows for the needed flexibility. The third layer implements a dedicated hardware architecture to process the pixel values. This third layer is unique to the task, so to account for the effort required for redesign to use this system for multiple tasks, a CAD Designer tool was developed. The CAD Designer contains a library of PEs for simple placement, verification and check-out of a new design.

3. THE NEW DESIGN APPROACH

The new system is implemented primarily in hardware, especially at the low level of processing. The speed and size advantage over typical image recognition systems is accomplished through the use of VLSI and ASIC technologies. The major components are optimized to function within the boundaries specified by the designer. As in any other imaging system, more than one pass through the data is required to recognize the predefined object. The idea behind this system design is the concentration on the parallel orientation of processing. Interprocessor communication is minimized in order to reduce the bottlenecks that exist in current systems. The object's image is segmented at the design stage into its major components, which are further divided into the basic shapes of which the image is comprised. The processing elements inputs are placed to take advantage of this open architecture, where the communication between the PEs is not required at the lower levels. This approach enabled the new design to be a truly parallel system. To produce a hardware processing system for image recognition, it is important to define the simplest elements of the image. These elements must be standard in shape, but can vary in size and orientation. The basic requirements of robotics and industrial imaging is mostly limited to predefined shapes composed of straight and curved lines. These lines, when combined in different ways, can produce any object's image. [2,3]

3.1 Architecture

The new system is composed of three functional layers that execute in parallel. The New Image (NI) Layer is a systolic array of the new binary image waiting to be processed. The Local Communicator (LC) Layer is also a systolic array that contains the local memory and local control. [4] The Processing Elements (PE) Layer performs all logical operations on the pixels. The Main Control Unit (MCU) provides uniform control throughout the layers of the system. The current design is implemented for binary imaging.

3.1.1 NI layer

The NI layer is the first layer to receive the digitized image data and transfers it to the other layers when the frame is full and when the data is requested. The new image array has a common design for this type of a system, with a selectable number of elements, depending on the size of the image to be analyzed. This layer is composed of two main components: the new image array and the array full detector, which serves as the local control. The elements of the new image array are designed with D flip-flops and arranged in a serial in / parallel out format for each row. The array starts loading from left to right, with all the rows loading concurrently, until the array is full.

When the new image array is full and is ready to transfer its contents to the next layer, with one clock pulse, the whole image array is transferred. After this transfer the new image layer starts loading the next image. Shifting of the data into the new image array is disabled when the array is full and the contents are held in place until the next layer is ready to receive them.

The only signals this layer needs from the Main Control Unit (MCU) is the clock and the signal to start loading a new image. The NI then sends a signal to the MCU indicating that the array is full and the new image is ready. This separation has been designed to allow the system to process the current image with the minimum overhead and maximum multiple processing environment.

3.1.2. LC layer

The LC layer is the second layer of the system, which interfaces with the NI layer and the PE layer. It receives input in total parallelism from the NI layer, on the command of the MCU. The LC array is a linear systolic array and is the same size as the NI array. The LC layer provides data to the PEs and serves as a data manipulation device which is used to transform the image data into different positions. Shifting of the data in different directions is one of this layer's main functions, but it can also be used for zooming and image rotation. There are separate data and command paths between the modules, and the buses are dedicated and directional. Each LC has four data lines within this layer and one connection to each of the other layers. Communication bottlenecks are not possible on this layer, because the architecture prohibits such cases. Also, the MCU contains a prioritization scheme that will schedule the sequence of events. Parallelism is widely implemented throughout this layer, except where data integrity can be compromised. In a compromised situation, operations are split into a sequence of steps each having compatible parallel functions. Each LC is made up of D flip-flops and combinational logic to interface to the adjacent left, right, up, and down pixels or A, B, C, D, respectively.

The LCs can receive input data internally, for data manipulations, or externally, from the NI layer. Figure 1 shows the block diagram of the LC layer, the control buses, and local circuitry that enables the data manipulation. The r/c signal is a global signal to the whole layer indicating either a row or column shift that translates to the S0 control bit. Row#, Col# carry the signal indicating the direction of the shift, which is the S1 signal. Because only row or column shifts are allowed upon the occurrence of each clock, these signals are ORed with no potential conflict. S2 represents an external shift from the NI layer and it overrides the two previous signals when set. Finally, the clock to the D flip-flop is ANDed with the enabled signal to generate local control on each of the elements and provide more freedom in shifting parts of the whole image.

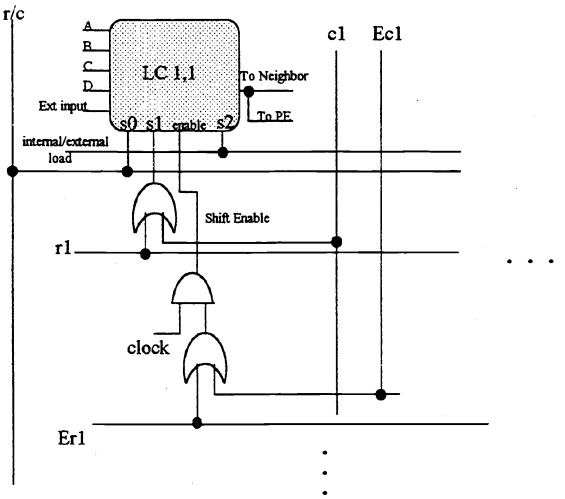

Figure 1 Block Diagram of LC Component

3.1.3 PE layer

The PE layer is the low level detection layer, which analyzes the values of the pixels and determines if the segment of the image expected exists or not. When designing this layer two major design points must be considered. First is the components and their architecture: their design is dependent on noise tolerance, speed of the processing, and size of the overall image. Because this is the pixel-level processing layer, its speed and the amount of parallelism in it will have the most influence on the performance of the system. The design is a combinational logic with logic components combined so that their binary outputs determine the existence of the segment. The second consideration is to the placement of the inputs of the PEs, which are assigned at the design phase to receive data from the local communicators. These inputs are labeled in the array terms to signify the row and column numbers from the LC components, or more precisely their (i,j) positional relationship to the image array. It is important to note that this massively parallel approach does not promote assigning a PE for each pixel

of the image array. Figure 2 shows the PEs to detect lines of 1s and lines of 0s, respectively. These PEs have been designed with combinational logic to detect the corresponding line segment and to filter-out low-level noise.

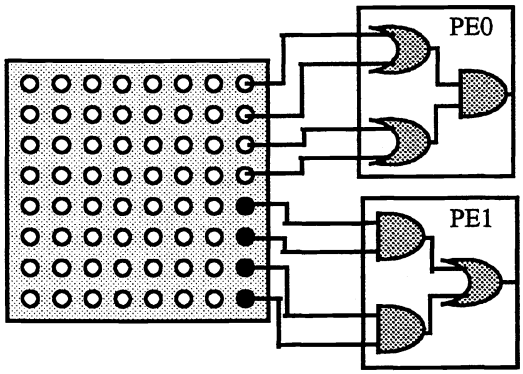

Figure 2 PEs to Detect 1s and 0s

The Processing Elements inputs are assigned with the same labeling system that is used in the LC layer. Each input of the PE is labeled in a consistent array (row,column) format that refers to its pixel location with respect to the whole image array. Because this design is tedious and time-consuming, the CAD designer algorithm is presented to do the mathematical calculations of positioning each PE input.

The design software is the CAD designer tool that is being implemented to produce and place the Processing Elements. This software enables the design of the PEs to be a user-friendly procedure. The CAD will check for logical errors and produce suggested architectures for better performance and higher reliability.

3.1.4 Combiners

The purpose of the combiners are to consolidate the outputs of the PEs to form the segment of the object detected or located. Each PE, or set of PEs, is responsible for a segment of the desired image. Combining all segments will produce the entire image. However, each segment has its own combiner whose output is used by the main control unit. The outputs of these PEs are sent to another combiner to be brought together to form the required image segment. One responsibility of the combiner is to serve as a counter for the PE outputs. The combiner consists of a parallel counting circuitry that accepts the outputs of the PEs and produces a binary count of the passing PE outputs. The combiner is composed of full adders (FA) that are organized to assign the same priority level to each of the PEs. The block diagram in Figure 3 represents the combiner of a line that is represented by a maximum of nine PEs.

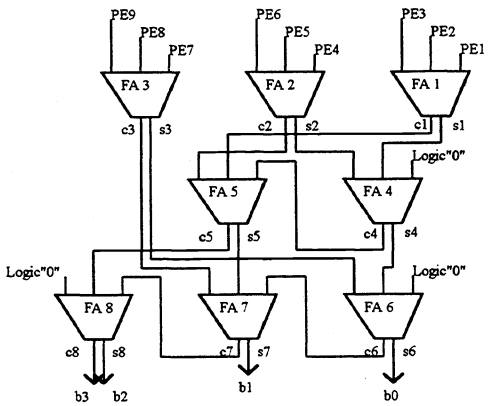

Figure 3 Combiner Block Diagram

Figure 4 shows the flow of data within the PE Layer.

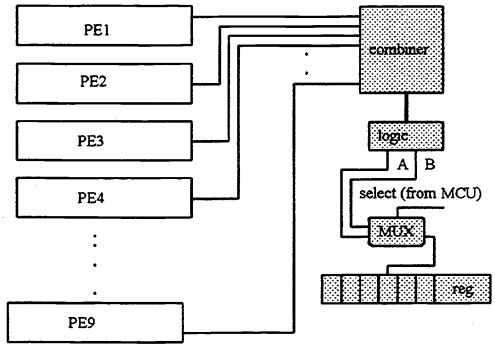

Figure 4 PE Layer Block Diagram

3.1.5 Main Control Unit (MCU)

This sub-system is designed to continuously monitor the progress of the events in all the components. The MCU serves as a sequencer and scheduler to the different layers, because the layers are concurrently processing at different levels. The MCU separates the control signals of the system into layers and stages to allow the different layers to continuously process data at different stages. This design of the MCU produces the effect of a massively parallel pipelined system, with each stage of the pipe being a massively parallel architecture.

The sequence of events, the direction, and duration of the signals are crucial to generating the correct result. The MCU will not enable the transfer of data to the LC layer until it receives a data-ready signal from the NI layer. After the new data has been received at the local communicator layer, the image data is made available to the PEs. The output of the combiners is loaded into the found image registers after receiving the signal from the MCU. Then the MCU signals to load the output of comparing the available image with the output of the expected image into the comparison register. At this time the shift detectors starts analyzing this output and provide a code that is loaded into the shift-detected registers. If this output contains any positive results, the system will go into an internal loop to correct the shift and bring back the image array to the expected position and re-analyze the available data.

3.2 Synchronization

Designing the system with three functional layers that are concurrent required the system to be synchronous in the data exchange mechanism to avoid the loss of information. This synchronization mechanism is done in the MCU. This emphasizes the need for a correct delay count in each of the layer components. A partial clocking of the functionality of the system is analyzed: after the compared image has been loaded, the detection mechanism of defining whether or not a shift left, right, up, or down has occurred, will be performed. If one of these conditions has occurred then the sequence of correcting will start. The image data in the LC layer will be shifted to its expected position, or if the function was to find the position, or correct mechanically, an appropriate action will be taken. Finally, an analysis of the PE layer and how it is synchronized by the MCU is presented. This is done by prompting the combiners to load, in parallel, the outputs of the PEs and determine the number of PEs that pass the requirements of detecting their prospective segments. The combiner outputs will not be read until the MCU sends a signal to the registers to load the combiner outputs.

4. THE CAD DESIGNER

This CAD package was designed and developed for the purpose of producing a functional and core description of the PE layer. The output is presented in three different formats: graphical, logical, and remodifiable code. The goal was to emphasize the modularity and reusability of the sub-features. This, in turn, promises to

reduce the design time as the library of features develop. Because the developed system is aimed at robotics and industrial applications, the PEs are designed to detect known graphical patterns. The system output will be a link in the chain of a closed-loop control system. This output is to be directed to a motion control system that will react to the specific findings of the results from the image recognition system.

The CAD designer presents its output in a format that can be adjusted and can be input to different wafer-layout systems. The CAD tool is currently implemented in the windows environment using the Visual C++ compiler. The program prompts the user for the different lines to be specified, type of edges, and boundaries to be detected. The maximum amount of mismatches, while considering the line acceptable, must be determined by the designer. This is an element of the predefined system tolerance. The CAD Designer currently produces the logic design file and an array file. This array file is produced for the user to visually inspect the layout of the object view and make changes if it does not meet the specifications.

The tool also provides the user the flexibility to enter the preference of the PE type, PE size, and the width of boundary detection. The current user interface is in the GUI environment of the Windows 3.1 application. The tool will prompt the designer to input the image array size, as shown in Figure 5. The window of Figure 6 shows the design entry mode of an object feature.

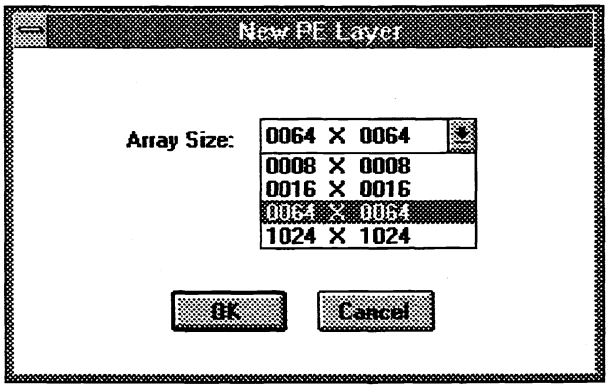

Figure 5 Image Array Size Selection

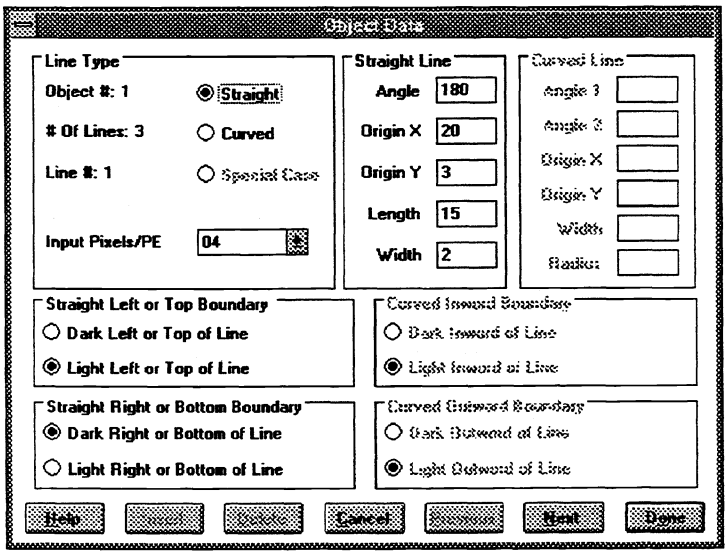

Figure 6 CAD Segment Description Entry Format

5. SUMMARY

This research targeted the areas of real-time image recognition systems for use in the robotics and industrial fields, and also for automatic target recognition and tracking systems. The requirements were to produce a system that is reliable, fast, mobile, and cost effective for actual machine vision applications.

The solution was found by combining the areas of ASP, VLSI, MPP, and current image processing techniques. The first two subjects were implemented to give the mobility aspect, while the third provided the real-time performance. This novel approach required a complete change in the design philosophy of existing image recognition systems. This idea has been proven to be plausible in many areas of image recognition. Moreover, this new approach takes advantage of modularity, strategic up-front design techniques, distributed processing, and system-wide parallelism. Thus, this unique, original, practical design is an ideal alternative for implementing systems in the fields of industrial automation, robotic vision, and inspection/monitoring. The implementation of this design addressed modularity and adjustable tolerances to accommodate real-world applications. Consequently, the resulting comprehensive architecture is unmatched in speed and size.

The final system was implemented with the CADENCE package and found to require about 250,000 gates on a chip of 64 X 64 resolution. Rigorous testing and benchmarking of the final results showed the performance of the system to be at least 15 times the throughput of the current systems.[5,6,7] The chips are cascadable to generate a system of higher resolution. The image complexity did not affect the processing time due to the inherent parallelism in the system. Overall, this new design approach allows a smooth progression into the next generation of image recognition systems.

6. REFERENCES

1. Child, J., "Image Processing", Computer Design, Vol. 35, No. 8, July 1996.
2. Farroha, B. and Deshmukh, R.G., "Real-time VLSI techniques for processing of computer images", SPIE Proceedings Society of Photo-Optical Instrumentation Engineers, 1996 International Symposium on Optical Science, Engineering, and Instrumentation
3. Farroha, B and Deshmukh, R.G., "An Alternative Method For Designing Parallel Image Recognition Systems: A Feasibility Study." IEEE Southeastcon '95 Proceedings, Raleigh, NC.
4. K. Johnson, B. Shirazi, "General Purpose Systolic Arrays," Computer Magazine, November 1993.
5. A. Boubekeur, G. Saucier, "A Silicon Compiler for Massively Parallel Image Processing ASICs," IEEE Proceedings The Third Symposium on the Frontiers of Massively Parallel Computation, 1990.
6. R. Sastry, N. Ranganathan, R. Jain, "VLSI Architectures for High-Speed Range Estimation," IEEE Transactions PAMI, Vol 17, No. 9, September 1995.
7. C. Brown, "MIT Project Sees Machine Vision for the Masses," Electronic Engineering Times, 23 October 1995.