



A systolic parallel Laplacian pyramid image compression architecture
by Diane Wendy Mathews

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in
Electrical Engineering
Montana State University
© Copyright by Diane Wendy Mathews (1992)

Abstract:

The Laplacian pyramid image compression (LPIC) algorithm can be used to preprocess an image for efficient compression, particularly for extraterrestrial exploration projects. LPIC involves a two-dimensional convolution which lends itself to a parallel architecture implementation. This thesis develops the design of a systolic parallel architecture for the algorithm.

The LPIC algorithm is implemented in a special purpose integrated chip design. The architecture for the system is discussed, as well as the individual registers used for the multiplication. The registers are constructed with a single-clock multiplier/accumulator circuit. Simulations show that the bit level circuit could operate at 50 MHz and is logically correct. A 16-bit register has been completely designed and submitted for fabrication. The modules for the convolution have also been designed.

**A SYSTOLIC PARALLEL LAPLACIAN PYRAMID
IMAGE COMPRESSION ARCHITECTURE**

by

Diane Wendy Mathews

**A thesis submitted in partial fulfillment
of the requirements for the degree**

of

Master of Science

in

Electrical Engineering

**MONTANA STATE UNIVERSITY
Bozeman, Montana**

January 1992

N378
M422

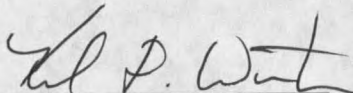
APPROVAL

of a thesis submitted by

Diane Wendy Mathews

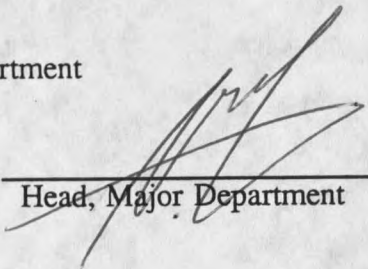
This thesis has been read by each member of the thesis committee and has been found to be satisfactory regarding content, English usage, format, citations, bibliographic style, and consistency, and is ready for submission to the College of Graduate Studies.

1-21-92
Date


Chairperson, Graduate Committee

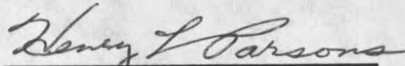
Approved for the Major Department

1-31-92
Date


Head, Major Department

Approved for the College of Graduate Studies

2/3/92
Date


Graduate Dean

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a master's degree at Montana State University, I agree that the Library shall make it available to borrowers under rules of the Library. Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of source is made.

Permission for extensive quotation from or reproduction of this thesis may be granted by my major professor, or in his/her absence, by the Dean of Libraries when, in the opinion of either, the proposed use of the material is for scholarly purposes. Any copying or use of the material in this thesis for financial gain shall not be allowed without my written permission.

Signature

Diane Walker

Date

1/29/92

To the Girl in the Hat

When, often, I see you smile and laugh
And hear the joy, the mirth, you spread abroad
I'd like to keep you, ever; so with a nod
My wistful longing you could quaff.

Bit by bit, and word by word,
We've come to know each other's soul.
To store your memory, relight the coal,
To write each sound that's e'er been heard.

What justice, then, can it do
Listing each sound, writing each spot?
'Til all that's left is mark and dot,
And this records what lives in you?

Yet so we work and live our days
To lose you not in the silicon maze.

- Diane Mathews

"So long as men can breathe or eyes can see
So long lives this, and this gives life to thee."

- 18th Sonnet, William Shakespeare

ACKNOWLEDGMENTS

I would like to gratefully acknowledge the help of my fellow graduate students, Bob Wall and Tim Thompson, for their invaluable help and support in the research and writing of this thesis. I would also like to express my thanks to Professors Fred Cady and Dan March for their time and energy. Thanks also to professor Kel Winters for his time and guidance on the project. Finally, thanks to Tim Henning, whose help, encouragement, and love made this thesis possible.

TABLE OF CONTENTS

	Page
1. INTRODUCTION	1
General Background	1
The Mars Rover Project	2
Scope of Thesis :	6
2. LAPLACIAN PYRAMID IMAGE COMPRESSION	7
3. DESIGN PROCEDURE	11
Design Goals	11
LPIC Algorithm	12
Reduction	13
Expansion	15
Multiplier/Accumulator	18
Single Clock Dynamic Logic	20
Bit Level Design	21
Logic	21
Timing	26
Layout	27
Summary of Proposed Design	27
4. FINAL DESIGN	29
Modifications on the Design	29
Layout	33
Bit Level	33
Register Level	35
Module Level	35
Simulations and Performance	35
5. CONCLUSION	37
Existing Results	37
Future Work	37
REFERENCES CITED	39
APPENDICES	41
APPENDIX A - Schematic Logic Simulations	42
APPENDIX B - TSPICE Simulations	49
APPENDIX C - Layouts	62

LIST OF FIGURES

Figure	Page
1. LPIC data flow diagram	8
2. Reduction architecture for one column module	13
3. Connection of reduction column modules	13
4. Expansion architecture	16
5. Expansion column modules	17
6. Chu multiplier/accumulator	19
7. Chu m/a logic schematic	20
8. Product logic table and schematic	22
9. Carry latch logic table and schematic	22
10. PXOR logic table and schematic	23
11. Carry logic table and schematic	24
12. NXOR logic table and schematic	25
13. Flipflop transition table and schematic	26
14. Systolic Chu m/a schematic, with multiplier pipeline	30
15. Operands in double systolic Chu m/a	31
16. Chu m/a n-bit register	32
17. Layout of the double systolic Chu m/a	34
18. Product logic simulation	43
19. Carry latch simulation	44
20. Carry generating and systolic NXOR logic simulation	44
21. PXOR simulation	45
22. Chu NXOR and carry generating simulation	46
23. Systolic Chu TSPICE simulation file	50
24. 50 MHz clock model	51
25. Systolic Chu circuit model	51
26. VTI_SPICE_SLOW.MOD (Transistor parameter model)	55
27. Carry latch simulation with TSPICE	56
28. Product simulation with TSPICE	57
29. Flipflop simulation with TSPICE	58
30. P-logic XOR simulation with TSPICE	59
31. N-logic XOR simulation with TSPICE	60
32. Carry logic simulation with TSPICE	61
33. Single systolic Chu m/a	63
34. Double systolic Chu m/a with multiplier pipeline	64
35. 16-bit register of double systolic Chu m/a and switch cell	65
36. 16-bit register core for fabrication submission	66
37. 16-bit register with pad-frame and routing	67
38. Reduction	68
39. Expansion	69

ABSTRACT

The Laplacian pyramid image compression (LPIC) algorithm can be used to preprocess an image for efficient compression, particularly for extraterrestrial exploration projects. LPIC involves a two-dimensional convolution which lends itself to a parallel architecture implementation. This thesis develops the design of a systolic parallel architecture for the algorithm.

The LPIC algorithm is implemented in a special purpose integrated chip design. The architecture for the system is discussed, as well as the individual registers used for the multiplication. The registers are constructed with a single-clock multiplier/accumulator circuit. Simulations show that the bit level circuit could operate at 50 MHz and is logically correct. A 16-bit register has been completely designed and submitted for fabrication. The modules for the convolution have also been designed.

CHAPTER 1

INTRODUCTION

General Background

Digital image processing is used in a variety of communication systems. For example, images for the media are often digitally processed, then transmitted via satellite, before appearing in newspapers or on live broadcasts; and satellite weather maps are produced and displayed with digital processing systems. The reason that images are often used, instead of verbal descriptions, is that an image can be more readily understood by people, more quickly, than verbal descriptions; thus fulfilling the old adage "a picture is worth a thousand words." Also, in satellite weather systems, there are often few other forms for the information to be transmitted. For example, there is no person, or equivalent intelligence system, on the satellite which could relate a verbal description of nebular movements; thus leaving images as the most easily assimilable form of information available for such descriptions.

If scientists, journalists, intelligence agencies, and anyone else with enough capital, can put a camera in orbit and receive images, then why not do the same for images of outer space? Digital images are, perhaps, the key to modern unmanned exploratory missions: "...to boldly go where no one has gone before." That is indeed what the Voyagers and Magellan have done for us. If people are to know what lies beyond the moon then a camera, and other scientific instruments, must be sent in people's place. A proposal has been made to put a roving vehicle on the surface of Mars. This vehicle would contain several systems: enough artificial intelligence to keep itself from rolling off of a cliff or into a cliff; a minimal laboratory with which to determine possible composition of materials it encounters; sensors for temperature, wind speed, magnetic influences, etcetera; and an image processing and transmission system.

A brief discussion of digital image processing systems is helpful before discussing the Mars rover project in more detail. Digital image processing systems include: sensor; digital to analog converter; preprocessor/encoder; transmitter; postprocessor/decoder; and receiving system. The preprocessor and postprocessor are usually for the purpose of protecting or enhancing the image. That is, if the receiver expects errors of a certain type then corrections can be affected via the pre- or postprocessors, or both. Image enhancement means that if a particular aspect of the image is considered more important than other characteristics then the enhancing subsystem may exaggerate that aspect. An example of enhancement might be that of contrast. That is, if an image is important because of the difference between light and dark areas then the enhancement subsystem would increase that contrast as much as could be tolerated within the limits of the system.

An ideal system will process an image very quickly, without losing any of the information in the image. The reason for speed is that those at the receiving end usually want as many images as are possible in as little time as possible, especially in the case of unmanned exploratory missions such as the Mars project. Image data is usually memory and/or transmission intensive. That means that it occupies a great deal of storage space and requires a lot of time and/or bandwidth for transmission. The image, then, must be compressed so that it requires less memory or bandwidth, but it must be compressed without any losses of the image content. The topic of this thesis is a preprocessor which operates, at very high speeds, on the digital image in order to help reduce the amount of data which is stored and transmitted, and it does so without losing any of the image information.

The Mars Rover Project

Many people have an interest in the closest planet to Earth. Scientists hope that an exploration of Mars will indeed put people further out into space than they have been to date. The tremendous expense of sending humans to Mars is prohibitive. An

unmanned mission does not require any of the environmental precautions or comforts. The machine is the only part of the exploration attempt that needs to travel, and also the only part that needs to "survive."

The proposed exploration of Mars does begin with an unmanned rover, which is to explore the surface of the red planet. As mentioned before, it will carry several types of scientific measuring devices, including an advanced imaging system. The idea behind the project is that it would provide another opportunity to examine the needs of scientific equipment in space or on another planet. The project, dubbed "Pathfinder" [NASA 1989], will consist of a rover which will collect data to be transmitted back to Earth. Data taken will include images, wind speed, air content, soil content, examinations of rocks, electric and magnetic field measurements.

The imaging system will be required to recognize types of terrain and objects, ranging from very small in size to huge. The importance of the imaging system is primarily for the rover's survival, and also for generating public interest for the project. The importance of the survival of the rover is mandatory in order for data to be transmitted back to Earth. There are educational bonuses to be achieved from having the images made available to the public. More importantly, for those interested in extraterrestrial exploration, public interest could generate support, political and financial, for more space projects.

The design goals for the imaging system, as mentioned before, are resolution, speed, and accuracy. Resolution is not an issue in this paper because by the time the preprocessor receives the image the image has been completely acquired. That is, the preprocessor developed in this thesis receives image data that has been acquired and stored in a buffer, so that resolution cannot be affected by the preprocessor. The topic of this thesis, however, does not contain a sensing system and, as mentioned earlier, does not begin its functions until the entire image has been acquired and stored.

Speed is important to the rover's imaging system mostly because the more images taken and stored, the more information is available. Information is the primary goal of the Pathfinder project. The speed of the rover's imaging system is nearly irrelevant at the receiving end. That is, there is a finite amount of information that can be transmitted from Mars to Earth. As long as the available bandwidth and time for transmission is entirely used in sending highly compressed data then any more information simply will not be sent to Earth. The rover, however, can use that information to determine whether or not to roam off of a cliff, or examine a particular rock.

One of the obstacles to constant transmissions to Earth is that, at present, there is not a sufficient satellite system to support it. At least one satellite would need to have an unobstructed transmission path to Earth in order to maintain constant transmissions. That means that Earth-based systems will only be able to receive a specific amount of information in that minimal time period per day, so from Earth's viewing perspective, the processor only has to be fast enough to fill up the transmittal time and bandwidth.

The rover does require image data that is processed at high rates of speed. The model for optimum speed, by the way, is that of having a human on the surface using his/her judgment to deem what is important or not, how to avoid obstacles, or visually analyze materials. A human can do many of these tasks almost simultaneously, or perhaps in parallel. Designing a machine to do this requires a great deal of planning. To date, one of the priorities in the imaging system is that of determining edges for the sake of its own survival, and secondarily in order to recognize distinct objects. The advantage of high rates of speed is that the rover could then accomplish more in less time, and it might be possible for the long-distance observers to have the information a little bit sooner. In other words, the imaging system should not be a bottleneck to the rover's mission. If it must be a bottleneck, then at least the data should get through it as quickly as it can.

The analysis of images by the rover's artificial intelligence (a.i.) will allow for limited modifications on the imaging system. Such modifications might include depth of field changes, scope of image changes, and even changes to the internal processing and encoding values. These changes made by the rover's a.i. systems will be real-time changes; changes made by Earth-based systems can occur as well, but will not be real-time. Suppose, for example, that the rover takes a panoramic view of an environment to which it has just moved, and then slowly zooms in to its immediate environment. The rover can be evaluating the atmosphere in its immediate environment, with its imaging system and its other instruments, making its decisions based on the images and data that it takes in. When Earth-based systems have the opportunity to view the panoramic, and some of the nearer views, they can make a decision as the rover's future direction of travel and perhaps modify some of the values used in internal image processing. Real-time operations, from Earth-based systems, are impractical due to the amount of time it takes for round-trip communication to and from Mars [NASA89].

The imaging system must be able to handle alterations in desired image characteristics, regardless of whether the decision about the characteristics is made from Earth or by the rover. The question of which part, or combination of parts, of the imaging system is to be used to handle the recognition of need for alterations, and the alterations themselves, has already been addressed. A part of the solution is to use the Laplacian pyramid image compression technique developed by Peter Burt and Edward Adelson [BuAd83]. A discussion of the technique is given in the second chapter; for the purpose of this paper, however, the only real consideration is that of the ability to change the internal values of the preprocessor. Other considerations include real-time correlation of image characteristics with stored characteristics, classification of the image according to spatial structure, differentiating between areas of varying reflectance[Huck89].

The final issue, that of the accuracy of the image, will depend on the system's physical characteristics, the algorithms used within the processors, and the implementation of those algorithms. Physical characteristics are in reference to the optical properties of the lenses used and the capturing abilities of the sensing devices. The algorithms used refers to the preprocessing and encoding of the image. The implementation of the algorithm differs from the algorithm itself in that the physical characteristics and requirements of implementations often detract from the theoretical performance of the algorithm. The accuracy of the image received from the acquisition system will be preserved up to the encoding stage if the implementation is perfect, and of course, if the algorithm is lossless.

The preprocessor is the hardware implementation of a Laplacian pyramid image compression (LPIC) algorithm. It receives digital image data, and reorganizes the data so that it can be efficiently encoded using a Huffman code. The next section discusses the LPIC algorithm in detail.

Scope of Thesis

This paper discusses the development of a systolic architecture for the Laplacian pyramid image compression algorithm. First, the operation of the LPIC algorithm is discussed in more detail. This is followed by a discussion of the top-down design, from module architecture to bit-level design. Finally, the results of simulations on the design, and figures representing the final design are presented.

CHAPTER 2

LAPLACIAN PYRAMID IMAGE COMPRESSION

Laplacian pyramid image compression (LPIC), developed by Burt and Adelson [BuAd83], is a method of compressing digital image data. The basic idea behind the algorithm is to extract spatial bandpass information from the image, and to do so for several spatial bandpass frequencies. In other words: the sharp edges are extracted and that information stored; and then lines are extracted, and the line information stored; broad line information is then extracted and stored; and so on until the system end has been reached. The spatial bandpass information is extracted by first filtering the input image, and then subtracting the filtered image from the input image. The filtered image is acquired by convolving the input image with a predetermined 5x5 matrix.

The reason for extracting the spatial bandpasses is that images often have a great deal of redundancy in them. For example, the "Girl in the Hat" image, which is often used in image processing research, contains background information that is not very interesting. If the background is primarily monochrome then a way to compress would be to describe the shade and shape of the background. This would result in far fewer bytes used to describe the background than if each individual pixel was recorded. A compression technique that could describe the background as monochrome might also describe any other large areas of uniform shade. Extracting spatial bandpasses of an image using the LPIC technique would yield the outline of the "Girl in the Hat", followed by stripes of monochromatic areas, and so on until the filtered image contained only the large areas of single shade, such as a background of the forementioned photograph.

The LPIC technique consists of a filter, reduction, expansion and subtraction, with storage capacity for the input image. The algorithm is, perhaps, best described with the help of Figure 1. In general, the input image is both filtered, and reduced in size, in the

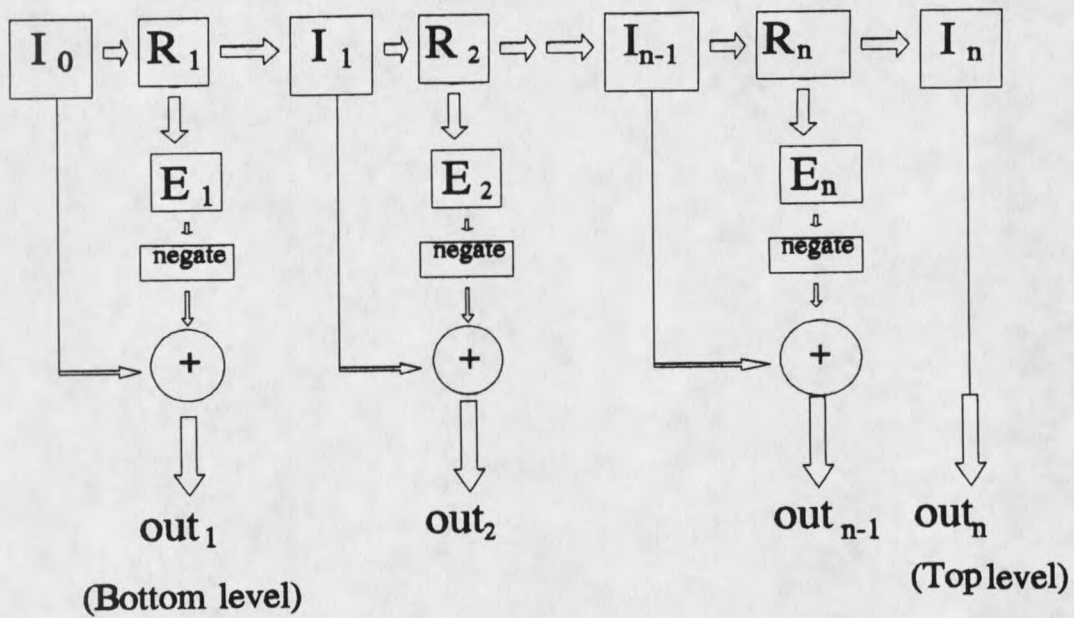


Figure 1. LPIC data flow diagram

block marked R. The reduced and filtered image is the input to the next stage, the next stage being identical to the first. The reduced and filtered image is then expanded to the size of the original, and subtracted from the original (in the block marked E). The result of the subtraction is the data which is retained for encryption and transmission.

It may be noted here that LPIC is not really a compression technique. The subtraction from the first step contains as much pixel data as did the original image. The results from each iteration of the algorithm produce image data with the same amount of pixel data as the input image had. After the first two iterations, for instance, the two output images have one-fourth again as much digital image data as the first image had. If the process was infinite then there would be 1.5 times as much "compressed" data as compared to the original input image. The compression occurs at the encoding step, which uses a Huffman-like, or block-length, code. Remember that the first filtered image

basically only has sharp edges and the rest of it is uniform. Most of the first filtered image could then be encoded with only one bit; that is, one bit for each pixel which is not part of an edge. The compression ratio for LPIC is entirely dependent on the code used to encode the processed image information. The relationship between codes and convolution mask values has been studied and is not part of the study for this thesis.

The reduction step of LPIC consists of a two-dimensional convolution in which only one-fourth of the convolved values are kept. These values comprise the reduced image. The expansion is a two-dimensional convolution as well, one which interpolates the missing values. The interpolation is a kind of averaging, so that the expanded image is blurred with respect to the original image. Thus, when the expanded image is subtracted from the original, the result is an image with edges only.

The convolution mask used for both the reduction and the expansion step is shown below, along with the relationships between the mask elements. The mapping of the input image with the convolution mask elements is also shown.

LPIC uses a two-dimensional convolution to filter the input image. The convolution mask lends itself to parallel input. If the image is processed in a parallel fashion then the amount of time required to process the image depends only on the amount of data in one of the dimensions. Most systems currently available process images one row at a time, a pixel at a time. The processing time is then dependent on the number of columns by the number of rows. In parallel system an entire column of image data enters the system at the same time. The processing time then depends only on the number of rows of image data.

The reduction is the filtering step, and the expansion interpolates the missing data points. The reduction step convolves the image with the mask, and retains only one fourth of the convolution. The output from the reduction step becomes the input image for the next iteration of the algorithm. The high pass information has been lost from the original

image, and must be found again. The filtered image is subtracted from the input image in order to regain the high pass information. The filtered image, however, has one fourth as many data points, so that it cannot be subtracted, point by point, from the input image. The expansion step interpolates, from the output of the reduction step, the missing values. The output, from the expansion step, has the same number of data points as the input image; it is negated and then added to the input image.

The subtractions, from each iteration, contain the information that is to be transmitted. The first iteration yields the highest pass information from the system. This is, perhaps, information that could be used to detect lines which might be cliff edges. If the first set of data were referred to as lines, then the second iteration would yield stripes, and the one following would yield bands.

The first iteration of the algorithm yields the bottom level of the pyramid. It is referred to as the bottom level of the pyramid because it is transmitted last. If the progressively extracted data were stacked on top of each other, drawn in scale to their size, the figure would resemble a pyramid. The top level of the pyramid is transmitted first, expanded, and the next layer transmitted and added in to the first level. The algorithm provides a lossless method of image reconstruction. An alleged advantage to this system is that the image reconstruction can be terminated before the final layer is received. That is, if the viewer deems the image information as unimportant, then (s)he can cancel the reception/transmission of the rest of the image and start over with an entirely new one.

CHAPTER 3

DESIGN PROCEDURE

Design Goals

The primary goal of the system is very fast throughput. A fully parallel and systolic system is one way to accomplish a high rate of throughput. A system is systolic when it is devoted to the single task of processing the image. The system will consist of processing elements which perform a simple task, and the processing elements will be connected locally [Ji-Ren91]. The hardware implementation of the system should "look like" the algorithm, which is one of the features of a systolic architecture. In order to make the system parallel, data enters the system a column at a time, instead of pixel by pixel. In terms of speed, this parallel data processing feature will mean that the time required to process the entire image is dependent not on the total number of pixels (columns by lines) in the image but only on the number of columns in the image.

Another way to reduce the amount of processing time is to reduce the number of mathematical operations that the system requires. Convolutions are highly multiplication and addition intensive. A minimal number of multiplications and additions, then, is a consideration in the design. The minimum number of multiplications, performed on an incoming pixel, by each multiplicand is one or zero. In the general convolution case the minimum number of multiplications is, obviously, one. The LPIC algorithm, however, does not multiply each pixel by each mask element. Some of the incoming data is multiplied more than once by a few mask elements. The repetition was used to preserve local connectivity. With the configurations in the next section, each data element is multiplied 3 times for the reduction, and 6 times for the expansion/interpolation step.

LPIC Algorithm

The LPIC algorithm filters the input image in three steps, which are iterated until the iterated image is small. The first step is the reduction, the second is the expansion, and the third is the subtraction. The reduction step both filters and reduces the input image. The reduced image becomes both the input image for the next iteration and the input to the expansion step. The expansion step interpolates, from the reduced image, in order to fill in the missing steps and also finishes the filtering of the image. The subtraction step subtracts the result of the expansion step from the input image of the current iteration.

Filtering occurs in a two-dimensional convolution of the images. The convolution mask for the algorithm is shown below. It is a symmetric mask, and the relationship between the mask values are also shown below. The system designed does not define what value is to be used for 'a.' The choice is left up to the user of the system.

A B C B A
 B D E D B
 C E F E C
 B D E D B
 A B C B A

$F = a$; often $a < 1$
 $E = 1/4$
 $D = 1/4 - a/2$
 $C = F * D$
 $B = E * D$
 $A = D * D$

The convolution mask for both the reduction and the expansion is the same. The reduction step does not retain a convolution for each point of the input image. The expansion step splits the incoming data lines in order to fill in missing data points. The data flow for both the expansion and the reduction are discussed in more detail below.

The data flow will be fully parallel in that the data enters on the left side by column, and exits on the right side, by column. For the reduction it enters the system two columns at a time, and exits one column at a time. For the expansion the data enters one column at a time and exits two columns at a time.

Reduction

The reduction algorithm is implemented as in Figures 2 and 3, where the letters

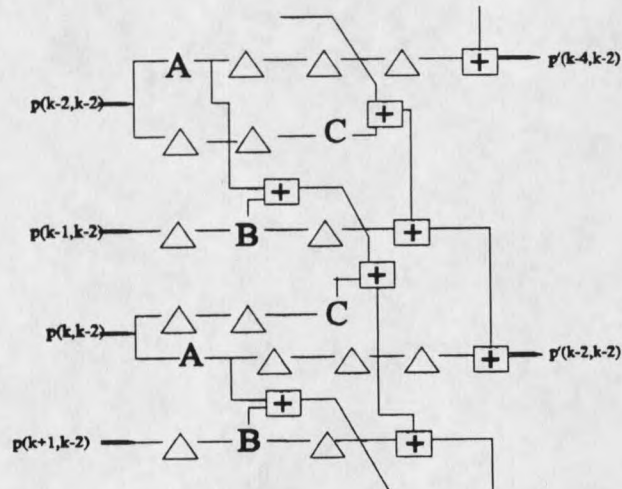


Figure 2. Reduction architecture for one column module

are the convolution mask values, the deltas are n clock cycle delays, the sum boxes are adding registers, and the p 's denote pixels from different rows. Note that each pixel of

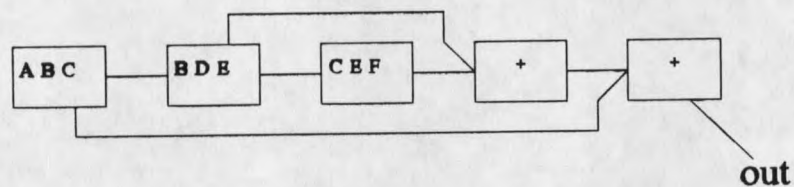


Figure 3. Connection of reduction column modules

the incoming column data is not necessarily multiplied by every element of the convolution mask. The odd column data, for instance, is only multiplied by the odd columns of the convolution mask; and the even column data is only multiplied by the even columns of the mask. Similarly, only the odd rows of the incoming data are multiplied by the odd rows of the convolution mask, and the even row data is only multiplied by the even rows of the convolution mask.

The fact that the even and odd columns are never multiplied by the same mask elements can contribute to speed up of the system. If even and odd columns are multiplied by entirely different elements then the system can process two columns at a time. Also, it does not multiply each incoming data element by each multiplicand.

Suppose that a pixel, $p(i,k)$, enters the reduction architecture in Figure 2 at the top row. During the first n clock cycles p is multiplied by A in the upper line, and shifted through a register in the lower line. The result from the multiplication in the upper line is then added to the result of $p(2,k) * B$, and $p(2,k)$ in the lower line is shifted again. During the third n clock cycles the result of $(p(1,k) * A) + (p(2,k) * B)$ is then added to $p(3,k) * C$. The accumulated result is then added in with $p(4,k) * B$ during the 4th n clock cycles. Not shown in the diagram is a module exactly like the diagram itself. If it were repeated so that the outgoing lines on the bottom were connected with the incoming lines of an identical module, then the accumulated result would then be added in with $p(5,k) * A$. This would yield one column's convolution with part of the convolution mask. An equation for a partially convolved pixel, p' , would be: $p'(k,k) = A*p(k-2,k-2) + B*p(k-1,k-2) + C*p(k,k-2) + B*p(k+1,k-2) + B*p(k+2,k-2)$. While column $k-2$ is being processed, column $k-1$ is being processed in the second module. The second module, in Figure 3, is identical in structure to the first. The multiplier values are changed from A, B and C to B, D and E , respectively. Column k is being concurrently processed by the third module. Again, it is identical in structure to the module in Figure

2, but with multipliers C, E, and F replacing A, B, and C, respectively. The results from the first two modules are forwarded to the modules represented by the adder boxes. This effectively produces the column convolutions for columns $k+1$ and $k+2$. An entire column of filtered data exits at the output.

There are two methods implemented in the scheme in order to reduce the data so that only one in four filtered results is obtained. The first is that only the convolution from every other row of the system is retained. The second is that the data are shifted in two columns in n clock cycles, while only one column exits in n clock cycles. The convolution is then computed for each n clock cycles, where n is the number of clock cycles required for one module to complete operations on a column of data. The output the adder becomes the addend for the next module. The next module has the same structure, but with different multiplicands.

Note that the reduction scheme implemented in Figures 2 and 3 has a highly regular structure. This means that only one module need be constructed. This module can then be repeated both horizontally and vertically to construct the entire reduction scheme. The module is repeated horizontally to effect the full convolution. Figure 2 shows a single row's worth of the two dimensional convolution, and Figure 3 shows the module level architecture for four incoming rows of data.

Each letter in the figure represents a register which requires n clock cycles to multiply the incoming data. The figure only shows the process for four incoming rows of data. It is repeated vertically in order to accommodate more rows.

Expansion

The expansion algorithm multiplies each data element by all of the multiplicands in the convolution mask. The data lines are split into two parts. The upper data line is multiplied by odd column elements of the convolution mask and the lower line is multiplied by even column elements.

