Reconfigurable communications for image processing applications

André B. Soares[1], Luigi Carro, Altamiro A. Susin
Instituto de Informática
Universidade Federal do Rio Grande do Sul - UFRGS,
Porto Alegre, Brazil
{borin,carro,susin}@inf.ufrgs.br

Abstract: This work tries to reuse programmable communication resources like a Network-on-Chip (NoC) in the acceleration of image applications. We show a mathematical model for the computation and communication pattern of two distributed motion estimation algorithms, Full Search Block Matching Algorithm and Multi-Resolution Block Matching Algorithm. Experimental results show that the use of the Multi-Resolution method reduces not only the computation time but also the traffic of messages on the NoC. This leads to a lower power consumption in the NoC during the processing time of each image. The studied examples show the importance of the link between algorithms and their mapping onto a programmable fabric, not only regarding computation, but facing communication as well.

1. Introduction

Image and video processing operations already have great importance nowadays, being used in a wide range of products, from industrial applications to mobile devices, like portable video cameras and portable DVD players. There are several critical issues in the design of these devices, like performance and, particularly in this last category, energy consumption.

Many efforts are made in the hardware domain as well as in the algorithmic domain in order to surpass the performance and energy limitations. One example is the motion estimation and compensation method, which is very studied because it's an important part of the most used video compression standards like MPEG2 and MPEG4 and in the emerging h.264. Basically, it consist in predict the content of a frame based on the motion of elements from a reference frame. The compensation of the motion of image blocks between successive frames eliminates most of the temporal redundancy of the video, reducing the necessary bandwidth to its transmission. The evaluation of the motion of the blocks basically consist in realize a search, comparing each block of the image to be predicted in a specific area in the reference frame and taking the best match position, by using an assessment criterion, as the Sum of Absolute Differences (SAD).

This search can be accelerated by the use of the parallelism, where more than one block are verified in the reference image. To do this, the image is divided in different regions and several processing elements can realize this search in parallel.

In order to realize this search in a more efficient way, optimized methods were created[1][2][3][4] and the developped architectures are, by a large majority, dedicated architectures. In this work we show that it's possible to use a very flexible communication infrastructure to replace dedicated communication architecture while mantaining a good performance. Also, this architecture can be reused to perform different algorithms. Two solutions will be considered: the Full Search Block Matching Algorithm (FSBMA), with a regular processing and an optimal solution, and the Multi-Resolution Block Matching Algorithm (MRBMA)[4], with smaller computational demand and with a most irregular processing. Both solutions will be mapped to a Network-on-Chip (NoC) with the objective of test the use of a multi-resolution method which use a layered image structure (pyramid) in a regular array of processing elements and to assess the behavior of the number of messages according to the algorithm. Both algorithms can run on the same platform by using the same processing elements when using a reconfigurable communication infrastructure. Even running algorithms with different communication patterns, the network permits the platform to be more flexible than using dedicated communications. Also, it is shown that changes in the algorithm reduce the computational demand, but also permit to exploit best the communication infrastructure. This brings energy savings with the reduction of the volume of the messages.

The use of NoCs to interconnect hardware elements in Image Processing architectures is not new. Some examples are platforms like X-Pipes[7] and aSoC[8]. However, in this work we try to explore link between the algorithms and the hardware to enhance the performance of the system.

2. First considerations:

The dimensions in pixels of each image will be indicated by T_{ix} and T_{iy}. The image will be divided in equal size regions. Each region will have dimensions of T_{rx} by T_{ry} pixels. The number of regions will be equal to the number of processing elements by image, and will be equal to $P_x \bullet P_y$, where $P_x = T_{ix}/T_{rx}$ and $Py = T_{iy}/T_{ry}$. Each

[1] The author is supported by CNPq(Conselho Nacional de Pesquisa).

block will have dimensions of 4 by 4 pixels. The number of blocks in a region will be equal to m•n, where $m=T_x/4$ and $n=T_y/4$. This is shown on Figure 1.

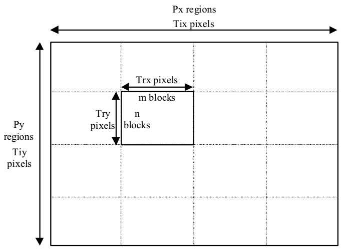

Figure 1-Definitions of the model used in the problem.

We assume that the blocks of an image (current image) will be located in a second image (reference image). In order to perform the search through the frontier between two regions, the left block column of each of each region of the reference image will be transmitted to the region on its left and the upper block line to the region above this. Figure 2 shows the communications of each processing element. The direction of each communication are indicated in the figure by the letters from A to P. The gray region corresponds to the column, line and block from the reference image and which will be received by the region shown.

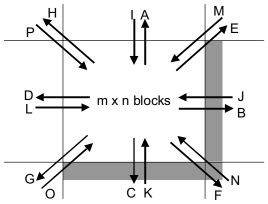

Figure 2 – Region of the image associated to a processing element and associated communications

Each processing element will store a region of the current image and from the reference image. These images are loaded before the processing. As the motion estimation has a very high computational cost, the storing of the image is justified. The generation of the additional images in lower resolutions will follow the same method and numbering presented in [4], in which each pixel in the inferior level (image with lower resolution) is obtained from the mean value of the corresponding four pixels in the superior level.

In order to realize the search of a 4 by 4 block in a region of the reference image with a distance from the original point equal to d, it´s necessary to sweep a region with the number of positions equal to:

$$N_p=(2d+1)^2 \qquad (1)$$

where Np is the number of pixels, and d is the distance.

Assuming that

- we have a datapath which can calculate one SAD per cycle between the 4 by 4 block from the current image and the 4 by 4 from the reference image in each processing element
- one 4 by 4 block needs 16 cycles to be loaded in the start of the process
- each change in the position after the initial loading needs 4 cycles

then the number of cycles N_{cb} necessary to realize this search is equal to

$$N_{cb}= 16+4N_p = 16d^2+16d+20 \text{ cycles} \qquad (2)$$

Once that one block of the current image is in one adjacent region of the reference image (because the distance to the frontier is smaller than the total search distance), this block must be sent to the neighbour region for the neighbour processor element to continue the search.

According to the position over the image, there will be different processors with different neighbourhoods (9 types). In Table 1 are shown the possible communications made by one processor. Due to the position of some processors in the network (like the ones in the border of the image, for example), some of the communications shown in Figure 2 will not exist. Here, we consider that

$$D=\left\lfloor \frac{d}{4} \right\rfloor+1 \qquad (3)$$

which is the distance d in pixels converted to a distance in block units, or multiple of 4 pixels. Blocks which are located in a distance D from the frontier must be sent to the neighbour processing element to complete the search process.

3. Full Search

To solve the motion estimation problem by using the FSBMA, one full search is done to each block of the current image in the reference image inside of a distance d from the original position of the block. This procedure is time consuming as the number of cycles grows with d^2.

4. Multi-Resolution Search

The use of a search in several images with different scales will reduce the computational demand when comparing to the full search algorithm. Results of a dedicated implementation can be seen in [4]. In this method, different copies of the image with lower resolutions are produced, following the powers of 2. After this, successive searches are performed starting with the images with lower resolutions until reach the image with the original resolution. The motion vectors found at coarser resolutions are used in finer resolutions as a start point. In this process, each search will be performed over a smaller distance d_m. In this case, the time necessary to the computation will be proportional to

d_m^2 and not d^2, where d_m is smaller than d.

Table 1-Communications for one processor in the central region of an image.

Comm.	Number of blocs from the current image	Number of blocks from reference image	Number of motion vectors
A	D•D•m	m	
B	D•(D-1)•n		
C	D•(D-1)•m		
D	D•D•n	n	
E	(D•D)•(D•D-1)		
F	(D•D-1)•(D•D-1)		
G	(D•D)•(D•D-1)		
H	(D•D)•(D•D)	1	
I			D•D•m
J			D•(D-1)•n
K			D•(D-1)•m
L			D•D•n
M			(D•D)•(D•D-1)
N			(D•D-1)•(D•D-1)
O			(D•D)•(D•D-1)
P			(D•D)•(D•D)
Total	$(2\bullet D^2-D)\bullet m+(2\bullet D^2-D)\bullet n+4\bullet D^4-4\bullet D^2+1$	m+n+1	$(2\bullet D^2-D)\bullet m+(2\bullet D^2-D)\bullet n+4\bullet D^4-4\bullet D^2+1$

The problem can be modeled in a way that the computation time in the multi-resolution search will be approximately equal to each plane. In this case, each plane will have the same kind of processing elements which will act over regions of the same size. Starting from a low resolution image over which one processing element will act, in a level L the number of processors will be equal to 4^L. In level 0 the computation time is be given by eq. (2) and there is no communications during the search process. In level 1 the computation time will be almost the same, but now there is the communication time between four processors and the loading time of the regions received by each processors. In level two there are 16 processing elements and messages of all types, and so on.

As the distance d_m will be much smaller than d, the number of blocks to be transferred between the processing elements in the several levels will be smaller with the utilization of the MRBMA. Part of the search is realized in the local processing element and part in its neighbour. The total computation time will be the same because it will also receive blocks from its neighbour to search in its region. In this case the number of cycles (16 in the used model) to load these blocks in the datapath must be added, multiplied by the number of received blocks. In the presented model they are omitted because the main focus of the work is the communication model.

In Table 2 we can observe the total number of blocks transmitted inside the same level over a distance D from the original position of the block. It's expected that the number of blocks transmitted in the MRBMA will be much lower than the number of blocks transmitted in the FSBMA in regions without significant motion because the smaller search range in each level. To regions where there is significant motion, the intra-level communications in the multi-resolution approach will be modified, because inferior levels will provide a starting point different from the original position. This new starting point can produce an increasing number of messages, as the blocks can have the beginning of its search outside of the current region or a decrease when more blocks from the border of the region start its search in the same region, but nearer the center of the region.

Table 2-Total number of blocks transferred in the same level

Level	No of P.E.s	Transferred blocks, by level (current image)	Transferred blocks, by level (reference image)	Motion vectors
0	1	0	0	0
1	4	$(4\bullet D^2-2\bullet D)\bullet m+(4\bullet D^2-2\bullet D)\bullet n+4\bullet D^4-4\bullet D^2+1$	2•m+2•n+1	$(4\bullet D^2-2\bullet D)\bullet m+(4\bullet D^2-2\bullet D)\bullet n+4\bullet D^4-4\bullet D^2+1$
2	16	$(24\bullet D^2-12\bullet D)\bullet m+(24\bullet D^2-12\bullet D)\bullet n+36\bullet D^4-36\bullet D^2+9$	12•m+12•n+9	$(24\bullet D^2-12\bullet D)\bullet m+(24\bullet D^2-12\bullet D)\bullet n+36\bullet D^4-36\bullet D^2+9$
N	Px•Py	$((2\bullet Px\bullet Py-2\bullet Px)\bullet D^2-(Px\bullet Py-Px)\bullet D)\bullet m+((2\bullet Px\bullet Py-2\bullet Py)\bullet D^2-(Px\bullet Py-Py)\bullet D)\bullet n+(4\bullet Px\bullet Py-4\bullet Px-4\bullet Py+4)\bullet D^4-(4\bullet Px\bullet Py-4\bullet Px-4\bullet Py+4)\bullet D^2+(Px\bullet Py-Px-Py+1)$	(Px•Py-Px)•m+(Px•Py-Py)•n+(Px•Py-Px-Py+1)	$((2\bullet Px\bullet Py-2\bullet Px)\bullet D^2-(Px\bullet Py-Px)\bullet D)\bullet m+((2\bullet Px\bullet Py-2\bullet Py)\bullet D^2-(Px\bullet Py-Py)\bullet D)\bullet n+(4\bullet Px\bullet Py-4\bullet Px-4\bullet Py+4)\bullet D^4-(4\bullet Px\bullet Py-4\bullet Px-4\bullet Py+4)\bullet D^2+(Px\bullet Py-Px-Py+1)$

Standard test sequences were used to evaluate the effect of the motion over the number of messages. In Table 3 we can see the results of these tests, considering video sequences of resolution 352x288, and three levels of processing (1, 2x2 and 4x4 processing elements). The number of blocks which need to be searched in adjacent regions was counted frame by frame by a test software. According to the mathematical model, the number of messages in the level 2 will be equal to 978 and the total number of messages considering all three levels will be 1140. The number of messages considering the full search algorithm in this case is 2588658. In Table 4 one can see the equations relating the number of motion vectors transmitted between two levels, as a function of the size of each region (in blocks) and the number of processing elements. A motion vector that comes from a lower level is replicated in the four corresponding blocks of the upper level. Because each P.E. communicates with its four neighbours and with neighbours in other planes, the ideal architecture is a pyramid. Nevertheless, in general this architecture does not have a very good layout when mapped to 2 dimensions. One solution is to map the planes to a more regular interconnection infrastructure which is not very well suited in the architectural point of view, but it's adequate to the mapping to the silicium. One possible placement can be seen in Figure 3, where the groups of processors which act in each planes are placed over distinct groups over the system's area. Point-to-point interconnections are not the best solution once that there is also the communication between the different levels, in which one message can run over several processors on the bidimensional array.

Table 3-Number of messages in some test sequences

Sequence	Messages (total of all levels)	Increase
Foreman	1195	4,82%
Football	1215	6,57%
Garden	1146	0,5%

Table 4-Motion vectors to be transmitted between two levels.

Levels	No of motion vectors
0,1	$Px \bullet Py/4^{(N-1)} \bullet m \bullet n$
1,2	$Px \bullet Py \bullet /4^{(N-2)} \bullet m \bullet n$
2,3	$Px \bullet Py \bullet /4^{(N-3)} \bullet m \bullet n$
(N-1),N	$Px \bullet Py \bullet m \bullet n$

The solution to this problem is the use of a NoC of the grid type, which has their tiles placed in a regular fashion. The network can transmit the motion vectors from one plane to another thanks to its routers, without the intervention of the processors. The communication overhead of the network can be very small when grouping messages to a specific processor. The contention will be also very small, once that most of the communications are over short distances (during the search), So, instead of a fixed and dedicated interconnection network, it's much better to use a communication infrastructure which supports some changes in the algorithm.

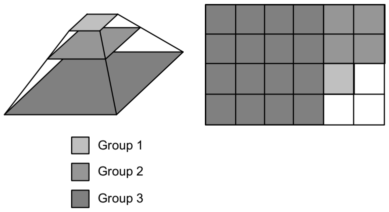

Figure 3-Three groups of processors which act over different planes of the images and a possible placement in a NoC of the grid type.

5. Results

Assuming that both methods of motion estimation will use the same type of P.E., now it's possible to make an assessment of the number of messages, which will have direct influence in the power consumption, once that it increases the execution time of the application and the switching activity over the network.

In Table 5 on can see the performance of each method when mapped to the NoC with different parameters. When the developed model is applied to images with typical resolutions that are used in known video standards[5][6], one can observe that the MRBMA is better not only from the computation point of view but also from the communication side, which are responsible for the power consumption in the network.

Table 5-Comparison between both search methods for different image resolutions

	Resolution	640x 480	720x 480	1600x 1152	1920x 1080
FS	No. of msgs	10.7M	10.7M	11.9M	12.1M
	Computation (Mcycles)	17.5	19.3	105.4	119.5
	P.E.s	64	64	64	64
	D	60	60	60	60
MRS	No. of msgs	5018	5298	12158	13278
	Computation (Kcycles)	408.0	448.8	2448.0	2774.4
	Levels	4	4	4	4
	D_{level}	4	4	4	4
	D	60	60	60	60
	P.E.s	85	85	85	85

6. Conclusions

A mathematical model of the interchange of messages from two distributed motion estimation applications mapped onto a NoC was presented. Results showed that the use of a multi-resolution method to motion estimation not only reduced the computational demand as expected, but also the communication volume between the P.E.s in the NoC. This will lead to a smaller time in the communication time. As the P.E.s and the network were of the same type in both cases, it's also expected that the power consumption will be lower in the second case also, not only in the computational part but also in the communication infrastructure. This also makes the method more adequate to the use on embedded systems.

7. References

[1] Byung Cheol Song; Nak Hoon Kim; Dong Keun Lim; Tae Hee Kim; Jun Hyuk Ko; Kang Wook Chun; Fast multi-resolution motion estimation algorithm and its VLSI architecture; Consumer Electronics, 2005. ICCE. 2005 Digest of Technical Papers. International Conference on 8-12 Jan. 2005 Page(s):71 – 72.

[2] Sarma, M.; Samanta, D.; Dhar, A.S.;Motion estimation using multi-resolution based on Haar wavelet transformation;TENCON 2004. 2004 IEEE Region 10 ConferenceVolume A, 21-24 Nov. 2004 Page(s):247 - 250 Vol. 1

[3] Jae Hun Lee; Kyoung Won Lim; Byung Cheol Song; Jong Beom Ra; A fast multi-resolution block matching algorithm and its LSI architecture for low bit-rate video coding; Circuits and Systems for Video Technology, IEEE Transactions on; Volume 11, Issue 12, Dec. 2001 Page(s):1289 – 1301

[4] Byung Cheol Song; Jong Beom Ra; A fast motion estimation algorithm based on multi-resolution frame structure; Acoustics, Speech, and Signal Processing, 1999. ICASSP '99. Proceedings., 1999 IEEE International Conference on;Volume 6, 15-19 March 1999 Page(s):3361 - 3364 vol.6

[5] Mitchell, Joan L.; Mpeg video compression standard. New York: Chapmann & Hall, c1997. 470 p. : il.

[6] Kuhn, Peter; Algorithms, complexity analysis and VLSI architectures for mpeg-4 motion estimation. Dordrecht: Kluwer, c1999. 235 p. : il.

[7] Bertozzi, D.; Benini, L.;Xpipes: a network-on-chip architecture for gigascale systems-on-chip; Circuits and Systems Magazine, IEEE; Volume 4, Issue 2, 2004 Page(s):18 – 31

[8] Laffely, A.; Jian Liang; Jain, P.; Burleson, W.; Tessier, R.; Adaptive systems on a chip (aSoC) for low-power signal processing; Signals, Systems and Computers, 2001. Conference Record of the Thirty-Fifth Asilomar Conference on;Volume 2, 4-7 Nov. 2001 Page(s):1217 - 1221 vol.2