Hardware Accelerating Design of Image Matching With FNCC Similarity Measure Algorithm

Xiong-Bo Zhao, Song-Ling Wu, Liang-Liang Liu
National Key Laboratory of Science and Technology on Aerospace Intelligent Control, Beijing 100854, China
Beijing Aerospace Automatism Control Institute, Beijing 100854, China
E-mail:bbzime@163.com

***Abstract*-Similarity measure is a usual image processing algorithm, which is decisive for the quality of image registration. And Fast Normalized Cross Correlation(FNCC) is one of the most common similarity matching algorithm. However, it is computationally intensive and very time-consuming which limits its application. This paper presents a hardware accelerating design of similarity measure algorithm using FNCC based on FPGA which is highly faster than DSP solution. The FPGA implementation is performed effectively according to formula deformation, optimized pipelining and parallel processing. And a new memory controller structure which supports mis-aligned access in different bit-width is designed. Compared to the implement on DSP which running at 300MHz, implement on FPGA which works at 100MHz improves the process speed more than an order of magnitude.**

Keywords-component; similarity measure; FNCC; image matching; memory controller structure

I. INTRODUCTION

Image registration is often necessary for integrating information taken from different sensors, finding identical geometries in images taken at different times or under different conditions. It is used to remove or suppress geometric distortions between the reference and sensed images, which are introduced due to different imaging conditions, and thus to bring images into geometric alignment [1].

Similarity measure is one of the most important steps for image registration and it will decide how to determine registration transformation. It is commonly used in image matching by evaluating the degree of similarity (or dissimilarity) between two compared images. The algorithm firstly locate print region after coarse positioning, then match the gray-scale character image with template image one by one. Experiment verify that the algorithm have a highly recognition rate.

Similarity measure can be decisive for the quality of registration. And there are a lot of different, well-known similarity measure algorithms and FNCC is a reasonable choice [2, 3]. FNCC has been used extensively for many signal processing applications such as object recognition, face detection, motion analysis and industrial inspections. However, it is computationally intensive and very time-consuming. The usual solution is implemented by programming on high-performance DSP. As the increased computational operand and real-time demand in some time-critical applications [4], DSP solution can not satisfy the real-time application requirements. Hard software is a valid solution to improve its processing speed. This paper describes the hardware design of similarity measure algorithm using FNCC and implementation based on FPGA (Virtex5-XC5VSX240T). It improves the operating speed greatly by pipelining and parallel processing.

The rest of this paper is organized as follows. In section 2, FNCC algorithm and deformation is introduced. In section 3, the algorithm hardware design is described in detail. In section 4, a memory controller structure is presented emphatically. Section 5 is a test experiment of algorithm working, and summary is given in section 6.

II. FNCC ALGORITHM AND DEFORMATION

FNCC refers two images which are template image and sensed image as shown in Figure 1. Sensed image is either bigger or smaller then template image. The smaller image is compared with the bigger image in a pixel-by-pixel basis.

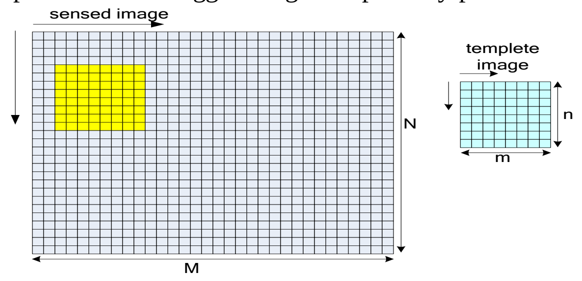

Figure 1. Image matching of FNCC.

FNCC algorithm can be shown in Eq (1), S denotes the sensed image of size M×N, and T represents the template image of size m×n. FNCC gives a measure of the degree of similarity, which is correlation coefficient ρ between template image and sensed image by normalizing the image and feature vectors to unit length [5].

$$\rho(p,q)=\frac{\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}[S(x+p,y+q)-\bar{S}_{p,q}][T(x,y)-\bar{T}]}{\{\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}[S(x+p,y+q)-\bar{S}_{p,q}]^2\}^{1/2}\{\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}[T(x,y)-\bar{T}]^2\}^{1/2}} \tag{1}$$

T(x,y) stands for the intensity value of template image of the size m×n at the point (x,y), xϵ, {0,1,…,m-1}, yϵ, {0,1,…,n-1}. And S(x+p,y+q) stands for the intensity value of sensed image of size M×N at the point (x+p,y+q), xϵ, {0,1,…,m-1}, yϵ, {0,1,…,n-1}, pϵ, {0,1,…,M-m}, qϵ, {0,1,…,N-n}. $\bar{T}$ is the average of template image, and $\bar{S}_{p,q}$ is the average of sensed image at the integer pixel offset (p,q) direction in the region under the template image.

ρ(p,q) is the calculated correlation coefficient between the template image and the searched regions of sensed image at the integer pixel position (p,q). Thus, FNCC algorithm will yield (M-m+1)×(N-n+1) ρ. The values of ρ ranged between −1 (when the matching entities are inverses of each other) and 1 (when the matching entities are exactly the same). The biggest absolute value of ρ represents the best match, and (p,q) is the match point.

FNCC formula is deformed to adapt for hardware design. Firstly, through unwinding and simplifying the Eq (1), we can get Eq (2).

$$\rho(p,q)=\frac{\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}S_{p,q}(x+p,y+q)\bullet T(x,y)-\bar{S}_{p,q}\bullet\bar{T}\bullet m\bullet n}{\sqrt{\left(\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}S_{p,q}(x+p,y+q)^2-\bar{S}_{p,q}^{\,2}\bullet m\bullet n\right)\bullet\left(\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}T(x,y)^2-\bar{T}^2\bullet m\bullet n\right)}} \tag{2}$$

There are large amount of calculations in this equation, especially the division and square root operation, which both are very time-consuming and resource-consuming. Because the template image doesn't change during the whole FNCC calculation process, arithmetic about T calculate only once. To reduce the operands, we try to displace the division and square root operation from S operation to T operation. Thus, we can get the Eq (3).

$$\rho(p,q)=\frac{\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}S_{p,q}(x+p,y+q)\bullet T(x,y)-\bar{T}\bullet\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}S_{p,q}(x+p,y+q)}{\sqrt{\left(\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}S_{p,q}(x+p,y+q)^2\right)\bullet m\bullet n-\left(\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}S_{p,q}(x+p,y+q)\right)^2}\bullet\sqrt{\frac{\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}T(x,y)^2}{m\bullet n}-\bar{T}^2}} \tag{3}$$

III. ALGORITHM HARDWARE DESIGN

The design structure of similarity measure algorithm using FNCC is shown in Figure 2. When getting the request signal, FNCC algorithm gets started. In term of Eq (3), (M-m+1)×(N-n+1) ρ is calculated as the ergodic matching between template image and sensed image. And position of biggest absolute value ρ is the orientation of the best matching image.

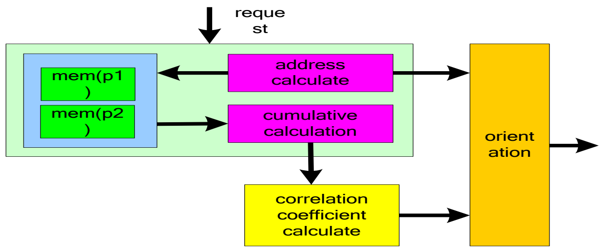

Figure 2. Design structure of FNCC algorithm.

From Eq (3), firstly we have to get the cumulative values of $\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}F(x,y)$, $\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}S_{p,q}(x+p,y+q)$, $\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}S_{p,q}(x+p,y+q)^2$, $\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}F(x,y)^2$, and $\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}S_{p,q}(x+p,y+q)\bullet F(x,y)$. Cumulative calculation adopts a 3-stage pipelining which is address computation, data read, and cumulative calculation, as shown in Figure 3. Each matching process has to read m*n grayscale from template image and sensed image memories respectively and synchronously. We present an optimized memory controller structure which is introduced concretely below. With the help of this memory controller, we can read 4 mis-aligned grayscales in one clk. It improves the operation speed nearly 4 times.

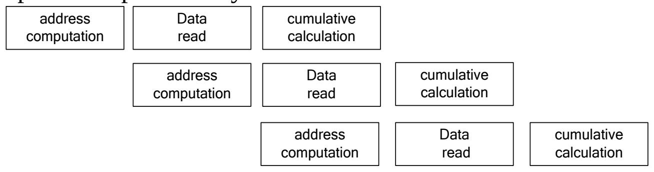

Figure 3. Pipelining design of cumulative calculation.

After getting the accumulated values, ρ is calculated in the light of Eq (3). At the same time, cumulative pipelining of next matching starts up. Thus, we can get the total pipelining in Figure 4.

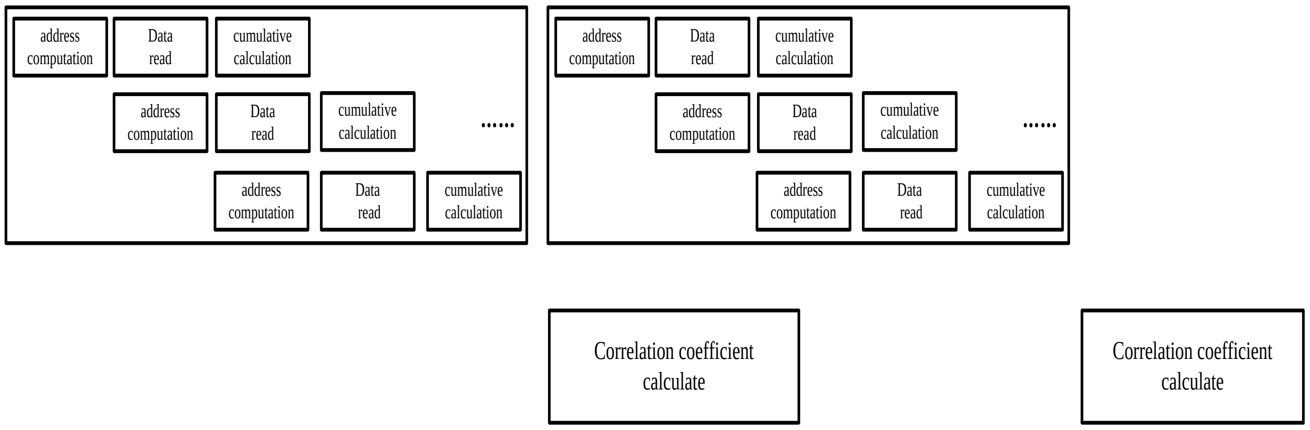

Figure 4. Total pipelining of FNCC.

The processing speed of measure algorithm is critical to us. And we use two dividers and two sqrt(square root) extractors. An optimized state machine which is shown in Figure 5 is designed to execute the ρ calculation. There are six states in our FSM, and gray code is adopted for state coding. The work of each state is described below.

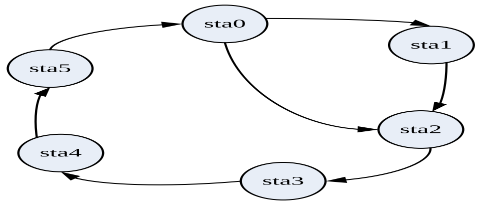

Figure 5. SM design.

sta0("000"): Backup cumulative values; prepare the divisors and dividends of $\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}F(x,y)\Big/_{m*n}$, $\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}F(x,y)^2\Big/_{m*n}$ and enable the dividers if the first matching of FNCC. Jump to sta1 if the first matching, to sta2 if not.

sta1("001"): Calculate $\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}F(x,y)\Big/_{m*n}$ and $\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}F(x,y)^2\Big/_{m*n}$, jump to sta2 when finished.

sta2("010"): Calculate the intermediate variables to prepare the radicands. Jump to sta3.

sta3("011"): Calculate the dividend of Eq (3); calculate the S image radicand and enable the sqrt extractor; calculate the T image radicand and enable the sqrt extractor if the first matching. Jump to sta4 when sqrt arithmetic of the S image is finished.

sta4("111"): Multiply the sqrt results of S image and T image and enable the divider. Jump to sta5.

sta5("110"): Calculate the ρ according the division arithmetic, and jump to sta0 to get ready for next operation when finished.

IV. MEMORY CONTROLLER DESIGN

In order to improve the operation rate, a new memory controller is designed. It supports mis-aligned access in different bit-width. This memory controller helps us read memory data conveniently without care about data alignment. The memory controller structure is shown in Figure 6.

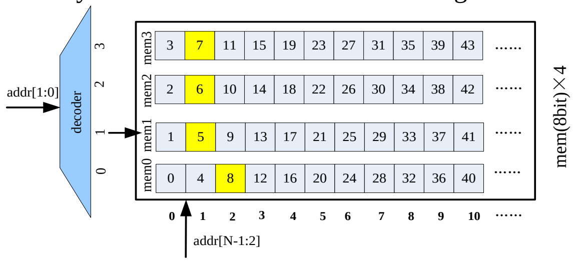

Figure 6. Memory controller structure.

It puts 4 8-bit memories together to composite control, which are mem0, mem1, mem2, mem3. Address is coding started from mem0, mem1, mem2 to mem3. As shown from figure 7, address 0 of mem0 is address 0 of our memory, and address 0 of mem1 is address 1 of our memory, and so on. Data is written into the memory in this address mode. When reading, address decoding is accomplished compositely. Addr[1:0] is decoded to choose the 8-bit memory, and at the same time addr[N-1:2] points to the address of 8-bit memory. The address of the chosen 8-bit memory is the starting address, and the required data after it will be read. This memory controller helps to access 4 data at one cycle without think about data alignment. 4 channels of data process concurrently.

V. EXPERIMENT TEST AND RESULT ANALYSIS

We take two images to test our hardware accelerating design of FNCC similarity measure algorithm. As shown in Figure 7, the size of sensed image is 80×80, and template image is intercepted from sensed image in coordinate (30,30), and its size is 40×40.

Figure 7. Test image.

Template image is compared with the sensed image in pixel-by-pixel basis, and the simulation waveform of similarity measure algorithm using FNCC is shown from Figure 8. The waveform displays the input signals such as image size, request signal, output signals such as correlation coefficient and its valid signal, matching point coordinate, finish signal, and some interactive signals with memory, some intermediate signals. The template image matches with the sensed image from coordinate (0,0) to coordinate (40,40). The peak point is (30,30) which also can be got from Figure 9. Its calculated correlation coefficient is nearly 1. It conforms to the interception position.

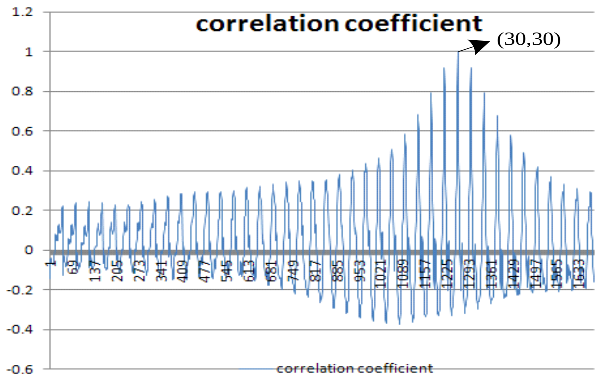

Figure 9. Correlation coefficient.

The hardware design is implemented on FPGA, and Virtex5-XC5VSX240T is chosen as the program device. The frequency constrain is set at 100M. After timing optimization and implement successfully, we get the device utilization which is shown in table 1.

TABLE I. FPGA DEVICE UTILIZATION

Slice Logic Utilization	Used	Available	Utilization
Number of Slice Registers	1,677	149,760	1%
Number used as Flip Flops	1,615		
Number used as Latches	1		
Number used as Latch-thrus	61		
Number of Slice LUTs	4,845	149,760	3%
Number used as logic	4,824	149,760	3%
Number using O6 output only	3,759		
Number using O5 output only	234		
Number using O5 and O6	831		
Number used as exclusive route-thru	21		
Number of route-thrus	283		
Number using O6 output only	255		
Number using O5 output only	28		
Number of occupied Slices	1,853	37,440	4%
Number of LUT Flip Flop pairs used	5,308		
Number with an unused Flip Flop	3,631	5,308	68%
Number with an unused LUT	463	5,308	8%
Number of fully used LUT-FF pairs	1,214	5,308	22%
Number of unique control sets	48		
Number of slice register sites lost to control set restrictions	76	149,760	1%
Number of BUFG/BUFGCTRLs	1	32	3%
Number used as BUFGs	1		
Number of DSP48Es	22	1,056	2%

Our accelerating hardware design of similarity measure algorithm using FNCC takes about 6.7ms to finish the image similarity matching of Figure 7. We run the same similarity measure on the DSP(TI-TMS320C6416), which is working at 300M. And it costs about 71ms. Therefore, implement of our hardware design on FPGA improves the processing speed more than an order of magnitude compared to DSP.

VI. SUMMARY

Similarity measure algorithm can be decisive for the quality of registration. Due to its increased operand and real-time demand, hard software of the algorithm is very requisite. This paper presents a hardware accelerating design process of similarity measure algorithm using Fast Normalized Cross Correlation (FNCC). A new memory controller structure which supports mis-aligned access in different bit-width is designed to allow reading 4 data at cycle. And according to formula deformation and optimized pipelining and FSM design, similarity measure algorithm is performed effectively, and it works at 100M at Virtex5-XC5VSX240T. It improves the processing speed more than an order of magnitude compared to the implement on TMS320C6416, which is running at 300M.

REFERENCES

[1] L.G.Brown, A Survey of Image Registration Techniques, ACM Computing Surveys, vol. 24(4), pp. 325 – 376, 1992.

[2] J.P.Lewis, Fast Normalized Cross-Correlation, Industrial Light and Magic, 1995

[3] D. Tsai, C. Lin, Fast normalized cross correlation for defect detection. Pattern Recognit. Lett. 24(15), 2625–2631 (2003)

[4] Jae-Chern Yoo, Tae Hee Han, Fast Normalized Cross-Correlation, Circuits, Systems and Signal Processing, December 2009, 28:819

[5] K.Briechle, U.D.Hanebeck, Template matching using fast normalized cross-correlation, Optical Pattern Recognition XII, 2001.