2019 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)

Performance comparison of GPU-accelerated fast motion estimation method

1st Pengcheng Chen
School of Computer Science,
Southwest Petroleum University
Chengdu, China
chenpengcheng66@outlook.com

2nd Bo Peng
School of Computer Science,
Southwest Petroleum University
Chengdu, China
bopeng@swpu.edu.cn

3rd Anxin Zou
Electric Power Research Institute,
Chongqing Electric Power Company
Chongqing, China
331501253@qq.com

4th Luwen Xu
Electric Power Research Institute,
Chongqing Electric Power Company
Chongqing, China
326810270@qq.com

***Abstract*—Motion estimation is an important part of ultrasound strain elastography. The sum-table based motion estimation method, which can effectively reduce computation time and redundant calculation, is a fast motion estimation approach. Except for NCC, it can also be extended to other motion estimators.**

The main work and contribution of this paper is to implement different motion estimators (CC, NCC, SSD, SAD) on GPU platform based on the sum-table method and to compare the performance of these motion estimators with the classical method on GPU. The difference of computational cost of each estimator based on the sum-table method mainly lies in the construction of the sum-table. The computation burden per thread is small, so data addressing is the dominant work per thread. It has been reported that the sum-table method does not have particular advantage over the classical method with a large search range on GPU platform. However, it possesses good performance with a huge tracking kernel window.

***Keywords*—motion estimation, ultrasound elastography, graphics processing unit, sum table**

I. Introduction

Ultrasonic strain elastography (USE) is a method for obtaining tissue strain to continue further clinical diagnosis [1]. Motion estimation is an essential part of USE. It is used to calculate the displacement of corresponding sampling points by comparing the similarity/correlation of the ultrasound echo radio frequency (RF) signal data blocks before and after tissue compression.

Distinctions among different motion estimator based on the similarity/correlation evaluation method, which will affect the accuracy and efficiency of the calculation. Normalized cross-correlation (NCC) represents a “gold-standard,” because it provides adequate precision, is easy to implement, and is the most commonly used motion estimation method [2]. Except for NCC, sum absolute error (SAD), sum square error and (SSD) and non-normalized cross-correlation (CC) have also been frequently used [3]. However, the algorithms will take plenty of time to calculate the similarity/correlation. When similarity/correlation evaluation is calculated by block matching, the windows of each sample will overlap, which results in repeated calculation process and high computational cost.

A fast NCC calculation was developed based on the sum-table method to solve high window overlap. It has been verified that the sum-table method can avoid redundant calculation in NCC and significantly reduce computation time [4]. The sum-table method proposed by Lewis employed the sum-table method in the denominator and fast Fourier Transform in the numerator [5]. Luo and Konofagou changed the numerator calculation to the sum-table method, and further reduced computational cost. In addition, the sum-table method was also extended to other motion estimation method, and it also showed high performance [6].

GPU platform has been demonstrated to have computing advantages in various ultrasound elastography applications, such as USE, shear wave elastography and color Doppler imaging [7]–[9]. By using GPU, the computational cost of the classical NCC method can be reduced substantially [10]. The sum-table NCC method was also implemented and analyzed on the computing performance of GPU [7]. However, no relevant work has been done to compare the performance of other common motion estimators by the sum-table method, including SAD, SSD, and CC on GPU platform.

The main work of this study is to compare the performance of different common motion estimators based on the sum-table method on GPU platform. The parallel strategies of classical and sum-table method for motion estimators are described. The computational efficiency of motion estimators based on the sum-table method is evaluated on GPU platform in 2-D estimation cases.

The project is funded in part by research grants from Projects of International Cooperation and Exchanges of Chengdu City (2019-GH02-00040-HZ), Nanchong Scientific Council (Strategic Cooperation Program between University and City: NC17SY4020).

II. Method

The entire computational process of motion estimation is divided into three steps: (1) Preparing the data that needs to be processed, and predetermining a search range and a tracking kernel window; (2) Calculating a similarity/correlation value between the pair of selected reference and target echo signals, generating a similarity/correlation value map for a given search range, and finding a maximum or minimum from the similarity/correlation value map; (3) Quadratic polynomial fitting. The main task of this step is to improve the quality of displacement image. Finally the displacement data will be calculated. Depending on the different motion estimator, the method of calculating the similarity/correlation will be different, but the whole process of calculating the displacement is the same. While the sum-table method is adopted, there is a step added to construct the sum-table.

A. Classical motion estimation algorithms

Each estimator has relative superiority and deficiency. NCC provides enough precision, and is easy to implement, but high computational cost. SSD can guarantee adequate accuracy with a relatively small computation. SAD has no obvious performance defects, nor does it have distinct advantages. CC is extremely dependent upon variations in mean and standard deviation of the RF signal data. It is proved that the computational costs of SSD, SAD and CC is lower than NCC [3].

Equation (1) is the calculation method of NCC. The reference and comparison ultrasound echo RF signal data are referred to as $f(m,n)$ and $g(m,n)$. W is the tracking kernel window size, τ is the shift between the reference and comparison windows, τ_x is the search range, which is determined by the range of the physiologic displacement corresponding to the X-axis, and τ_y corresponds to the Y-axis.

$$R_{\text{NCC}}(u,v,\tau_x,\tau_y) = \frac{\sum_{m=u}^{u+W_x-1}\sum_{n=v}^{v+W_y-1} f(m,n)\cdot g(m+\tau_x,n+\tau_y)}{\sqrt{\sum_{m=u}^{u+W_x-1}\sum_{n=v}^{v+W_y-1} f^2(m,n)\cdot \sum_{m=u}^{u+W_x-1}\sum_{n=v}^{v+W_y-1} g^2(m+\tau_x,n+\tau_y)}} \tag{1}$$

The numerator of NCC calculates the cross-correlation between the signal in the reference window and the signal in the comparison window, which is $\sum_{m=u}^{u+W_x-1}\sum_{n=v}^{v+W_y-1} f(m,n)\cdot g(m+\tau_x,n+\tau_y)$.The denominator of NCC is divided into two parts. The first is $\sum_{m=u}^{u+W_x-1}\sum_{n=v}^{v+W_y-1} f^2(m,n)$, which represents the total energy in the reference window. The second is $\sum_{m=u}^{u+W_x-1}\sum_{n=v}^{v+W_y-1} g^2(m,n)$, which represents the total energy in the comparison window.

CC does not need to be normalization by the energy of the reference and comparison window, and can be calculated by the cross-correlation between the signal in the reference window and the signal in the comparison window alone.

The SAD sums the absolute of the difference between the reference and comparison signals. The method is express by

$$R_{\text{SAD}}(u,v,\tau_x,\tau_y) = \sum_{m=u}^{u+W_x-1}\sum_{n=v}^{v+W_y-1} |f(m,n) - g(m+\tau_x,n+\tau_y)|, \tag{2}$$

The SSD sums the square of the difference between the reference and comparison signals. This method is defined as

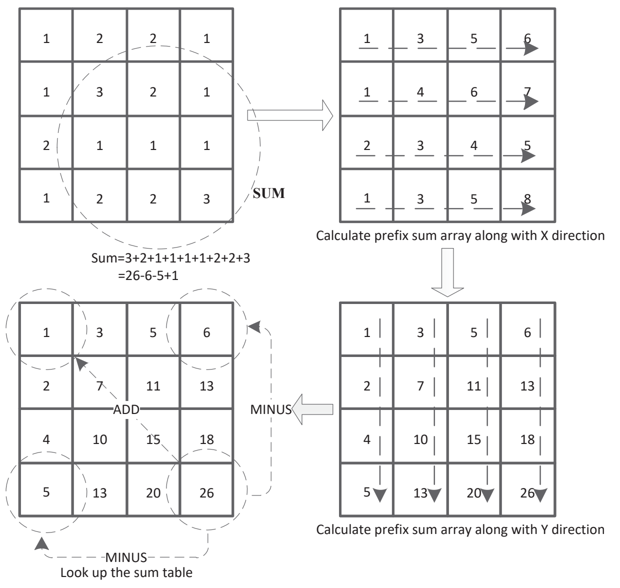
$$R_{\text{SSD}}(u,v,\tau_x,\tau_y) = \sum_{m=u}^{u+W_x-1}\sum_{n=v}^{v+W_y-1} [f(m,n) - g(m+\tau_x,n+\tau_y)]^2, \tag{3}$$

The calculation of variance of squares can be expanded in mathematical calculation, so the formula can be expanded as follows

$$\begin{aligned} R_{\text{SSD}}(u,v,\tau_x,\tau_y) = &\sum_{m=u}^{u+W_x-1}\sum_{n=v}^{v+W_y-1} f^2(m,n) \\ &+ \sum_{m=u}^{u+W_x-1}\sum_{n=v}^{v+W_y-1} g^2(m+\tau_x,n+\tau_y) \\ &- 2\sum_{m=u}^{u+W_x-1}\sum_{n=v}^{v+W_y-1} f(m,n)g(m+\tau_x,n+\tau_y), \end{aligned} \tag{4}$$

B. Motion estimations algorithms using the sum-table method

The most important step in the sum-table method is to preprocess the ultrasound echo RF signal data. The purpose of preprocessing is to construct a sum table. Fig. 1 illustrates the process of constructing a sum-table. The process can be divided into two steps in two-dimensional case: (1) Calculating



SUM
Calculate prefix sum array along with X direction
Sum=3+2+1+1+1+1+2+2+3
=26-6-5+1
ADD
MINUS
MINUS
Look up the sum table
Calculate prefix sum array along with Y direction

Fig. 1. Diagram of setting up a sum-table.

prefix sum array along with X-direction; (2) Calculating prefix sum array along with Y-direction based on the result of (1). Fig. 1 also outlines the calculation of the similarity/correlation by sum-table. The classical method must to calculate the sum of all values in whole window, nevertheless the sum-table method with no need for the same process.

The sizes of the window are assumed to be W_x and W_y, corresponding to the X and Y axes respectively. Suppose that the coordinate point (u, v) is in the center of the window. Nevertheless, the sum-table method only evaluates three times. The value of coordinate point $(u + W_x - 1, v + W_y - 1)$ minus the value of coordinate point $(u + W_x - 1, v - 1)$ and $(u-1, v+W_y-1)$. The value of coordinate point $(u-1, v-1)$ has been subtracted twice. Thus, the value of this point needs to be added again.

As mentioned above, direct computation of similarity/correlation can be converted into a "lookup-table".

The calculation of NCC can be divided into three parts, which can be constructed as

$$\begin{aligned} s_{f,g}(u, v, \tau_x, \tau_y) =& f(u, v) \cdot g(u, v, \tau_x, \tau_y) \\ &+ s_{f,g}(u-1, v, \tau_x, \tau_y) \\ &+ s_{f,g}(u, v-1, \tau_x, \tau_y) \end{aligned} \tag{5}$$

$$\begin{aligned} s_g^2(u, v, \tau_x, \tau_y) =& g^2(u, v, \tau_x, \tau_y) \\ &+ s_g^2(u-1, v, \tau_x, \tau_y) \\ &+ s_g^2(u, v-1, \tau_x, \tau_y) \end{aligned} \tag{6}$$

$$s_f^2(u, v) = f^2(u, v) + s_f^2(u-1, v) + s_f^2(u, v-1) \tag{7}$$

The numerator in the NCC can be directly calculated as

$$\begin{aligned} \sum_{m=u}^{u+W_x-1} \sum_{n=v}^{v+W_y-1} & f(u, v) \cdot g(m+\tau_x, n+\tau_y) = \\ & s_{f,g}(u+W_x-1, v+W_y-1, \tau_x, \tau_y) \\ & -s_{f,g}(u-1, v+W_y-1, \tau_x, \tau_y) \\ & -s_{f,g}(u+W_x-1, v-1, \tau_x, \tau_y) \\ & +s_{f,g}(u-1, v-1, \tau_x, \tau_y) \end{aligned} \tag{8}$$

Certainly, this approach can be applied to CC method. CC can be easily calculated from (8) with the sum-table method.

The energy of the reference window can be calculated through

$$\begin{aligned} \sum_{m=u}^{u+W_x-1} \sum_{n=v}^{v+W_y-1} & f^2(u, v) = s_f^2(u+W_x-1, v+W_y-1) \\ & -s_f^2(u-1, v+W_y-1) - s_f^2(u+W_x-1, v-1) \\ & +s_f^2(u-1, v-1) \end{aligned} \tag{9}$$

The energy of the comparison window can be calculated as

$$\begin{aligned} \sum_{m=u}^{u+W_x-1} \sum_{n=v}^{v+W_y-1} & g^2(m+\tau_x, n+\tau_y) = \\ & s_g^2(u+W_x-1, , v+W_y-1, \tau_x, \tau_y) \\ & -s_g^2(u-1, v+W_y-1, \tau_x, \tau_y) \\ & -s_g^2(u+W_x-1, v-1, \tau_x, \tau_y) \\ & +s_g^2(u-1, v-1, \tau_x, \tau_y) \end{aligned} \tag{10}$$

The SAD method of set up the sum-table can be express by

$$\begin{aligned} s_{\text{SAD}}(u, v, \tau_x, \tau_y) =& |f(u, v) - g(u+\tau_x, v+\tau_y)| \\ &+ s_{\text{SAD}}(u-1, v, \tau_x, \tau_y) \\ &+ s_{\text{SAD}}(u, v-1, \tau_x, \tau_y) \end{aligned} \tag{11}$$

The SAD can be calculated by the sum-table method as

$$\begin{aligned} R_{\text{SAD}}(u, v, \tau_x, \tau_y) =& s_{\text{SAD}}(u+W_x-1, v+W_y-1, \tau_x, \tau_y) \\ &- s_{\text{SAD}}(u-1, v+W_y-1, \tau_x, \tau_y) \\ &- s_{\text{SAD}}(u+W_x-1, v-1, \tau_x, \tau_y) \\ &+ s_{\text{SAD}}(u-1, v-1, \tau_x, \tau_y) \end{aligned} \tag{12}$$

The calculation method of SAD is described in formula (4), which can be used to calculate $f^2(m, n)$, $g^2(m+\tau_x, n+\tau_y)$ and $f(m, n)g(m+\tau_x, n+\tau_y)$ separately. Their sum-table can refer to NCC.

The SSD method's sum-table can be constructed as

$$\begin{aligned} s_{\text{SSD}}(u, v, \tau_x, \tau_y) =& [f(u, v) - g(u+\tau_x, v+\tau_y)]^2 \\ &+ s_{\text{SSD}}(u-1, v, \tau_x, \tau_y) \\ &+ s_{\text{SSD}}(u, v-1, \tau_x, \tau_y) \end{aligned} \tag{13}$$

The SSD can be calculated by the sum-table method as

$$\begin{aligned} R_{\text{SSD}}(u, v, \tau_x, \tau_y) =& s_{\text{SSD}}(u+W_x-1, v+W_y-1, \tau_x, \tau_y) \\ &- s_{\text{SSD}}(u-1, v+W_y-1, \tau_x, \tau_y) \\ &- s_{\text{SSD}}(u+W_x-1, v-1, \tau_x, \tau_y) \\ &+ s_{\text{SSD}}(u-1, v-1, \tau_x, \tau_y) \end{aligned} \tag{14}$$

III. Implementation

When calculating the correlation coefficients using classical motion estimation methods, the axial displacement of the point number and the lateral displacement of the point number denoted by M and N. In brief, these two parameters determine the total displacement estimation point (M×N) where M is the number of axial sampling points and N is the number of lateral ultrasound scan lines.

Each correlation coefficient is calculated without data dependence. Therefore, a parallel strategy can be adopted to optimize its calculation. A and B are the size of the tracking kernel window. (M×N)×(A×B) threads can be used to calculate all the correlation coefficients and then (M×N) threads are allotted to find a maximum or minimum on the correlation coefficient map and to accomplish quadratic polynomial fitting. This parallel strategy is applicable to all classical motion

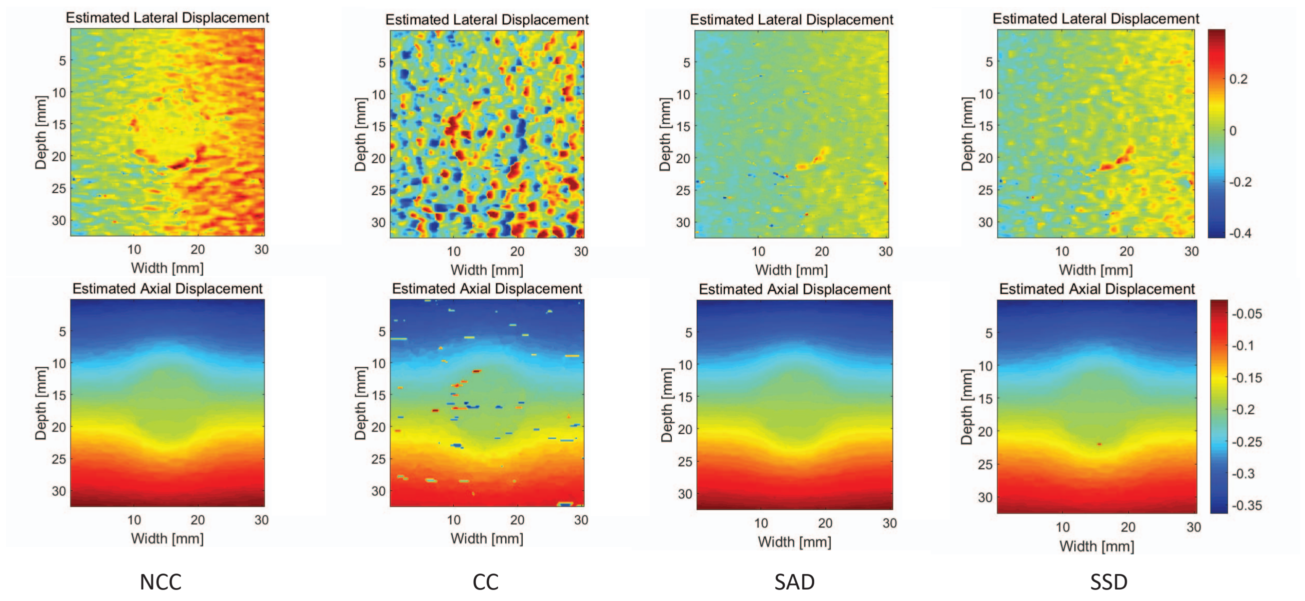


Fig. 2. Displacement image with different displacement estimators.

estimation methods mentioned in this paper (NCC, CC, SAD, SSD).

The parallel strategy is similar to the above method, which can be used in motion displacement estimation based on the sum-table method. (M×N)×(A×B) threads are still used to lookup-table to calculate all correlation coefficients. In the case of two-dimensional data, it is not necessary to construct the sum-table on GPU since CPU can also perform well. Constructing a sum-table is considered data preprocessing.

In addition, several strategies are utilized for memory reading. Firstly, constant memory is used to store parameters that are important, and used repeatedly. Secondly, in order to ensure data alignment and merge memory access, a one-dimensional method is adopted to import data into the device. Finally, off-chip memory (such as global memory, texture memory) has a higher access delay than on-chip memory (such as register, shared memory, constant memory), so on-chip memory should be used to improve access efficiency as much as possible.

A. *Experimental design*

The emphasis of this study is to compare the computational efficiency of different motion estimation methods. We use a computer to simulate the ultrasound echo RF signal in order to adjust the test data more flexibly. First of all, a numerical phantom is designed to simulate the human environment. The numerical phantom size is (40mm × 40mm). The center of the numerical phantom contains a circular inclusion with a radius of 5 mm that is 5 times harder than the background. Then in order to simulate the displacement process in ultrasound elastography, the numerical phantom is compressed by 1% in the axial direction by the finite element software ANSYS (ANSYS, Inc., PA, USA). Finally, ultrasound echo RF signals can be generated by a software package FIELD II [11].

For evaluating the performance of sum-table based motion estimation on GPU, the GPU version classical and sum-table based motion estimation methods are implemented together in this study. The computational efficiency of the classical and sum-table motion estimation method, affected by the parameters on GPU, is discussed by modifying speckle tracking parameters.

The GPU hardware used for algorithms implementation is Nvidia GeForce GTX TITAN V (Nvidia Corp., Santa Clara, CA, USA). The GPU card comes along with 80 Stream Multiprocessors, with a total of 5120 CUDA computing cores and 12 GB of Memory. The CPU platform is Intel(R) Core(TM) i7-8700 CPU 3.20GHz, and 16 GB of host memory. The ANSI C and CUDA 10.0 were adopted for implementing all the algorithms.

IV. Results

Fig. 2 shows displacement image of different displacement estimators. CC, NCC, SAD and SSD are implemented on GPU in this study. Moreover, the classical and the sum-table methods of each motion estimator are completed for performance comparison. The accuracy comparison of different motion estimators is beyond the scope of this paper.

A. *Comparison time cost*

Table I shows the computational cost of both classical and the sum-table method under different motion estimators on GPU. The size of the displacement image is 170×196, the tracking kernel window is 61×11, and search range is 11×7. The size of the data does not change significantly after constructing a sum-table, so this study ignores the time

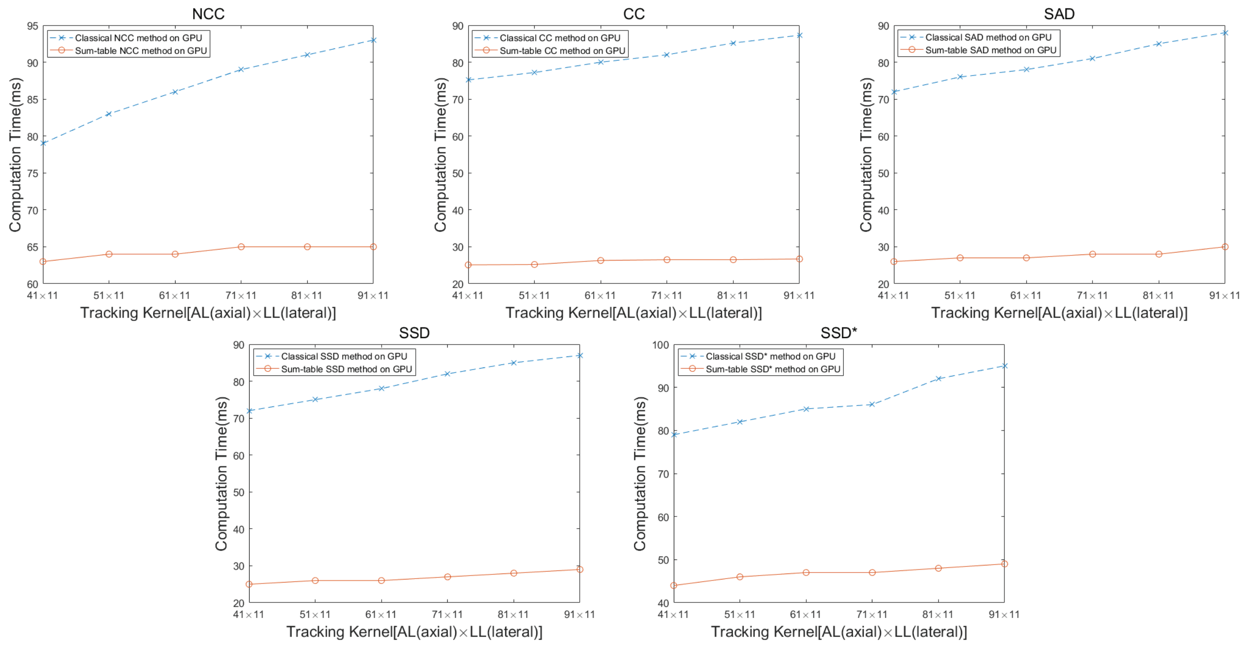


Fig. 3. Computing time when the tracking kernel changed with different motion estimation methods.

caused by copying the data to the device. In the case of two-dimensional data, it is not necessary to complete the sum-table construction process on GPU. Constructing sum-table as a data preprocessing is implemented on CPU.

In contrast, it can be seen that the sum-table method can reduce the time of computation. In general, CC, SAD and SSD have improved computational efficiency by using sum-table method. Compared to the motion estimators described above, the performance improvement of NCC and SSD* are not obvious. It takes a plenty of time to construct the sum-table and lookup-table so that they are less efficient than CC, SAD and SSD.

TABLE I
COMPUTATIONAL COST (ms) OF DIFFERENT MOTION ESTIMATORS BASED ON SUM-TABLE

	GPU		**CPU**
	Classical Method	***Lookup-table***	***Construct sum-table***
CC	80	15	11
NCC	86	37	27
SAD	78	14	13
SSD	78	14	12
SSD*	85	27	20

* calculted by Eqs.(4).

B. Computational efficiency

As shown in Fig. 3, only the tracking kernel is changed to observe the effect of this parameter on the calculation efficiency. The calculation time of motion estimators based on sum-table is almost independent of the size of the tracking kernel window, while the calculation time of classical method is greatly affected by the size of the tracking kernel window. The larger the window, the longer the computation time will be needed in classical method. Therefore, in terms of computational performance, the sum-table method outperforms the classical methods when large tracking kernel window is adopted.

As shown in Fig. 4, the computational efficiency of the sum-table method does not improve significantly with the increase of the search range. After changing the size of the search range, more sum-tables need to be construct to provide the calculations. Since SAD, SSD and CC method only construct a sum-table once, these methods still provide good performance. The CPU has sufficient computing advantages when calculating prefixes on a two-dimensional data scale. As NCC and SAD need to construct the sum-table multiple times, these estimators have a non-significant performance improvement.

V. Discussion

The main work and contribution of this paper is to implement different motion estimators (CC, NCC, SSD, SAD) on GPU platform based on the sum-table method and to compare the performance of motion estimators by using classical methods on GPU. From this performance comparison, it can be seen that the computational cost in lookup-table is independent of the size of the tracking kernel window. With the increasement of the search range, more sum-tables need to be constructed, which reduces the computational efficiency.

The differences of the computational cost of each estimator based on sum-table method mainly lies in the construction of sum-table. The computation burden per thread is small, so data addressing accounts for most of the computational cost per thread. It is worth noting that the data pre-processing

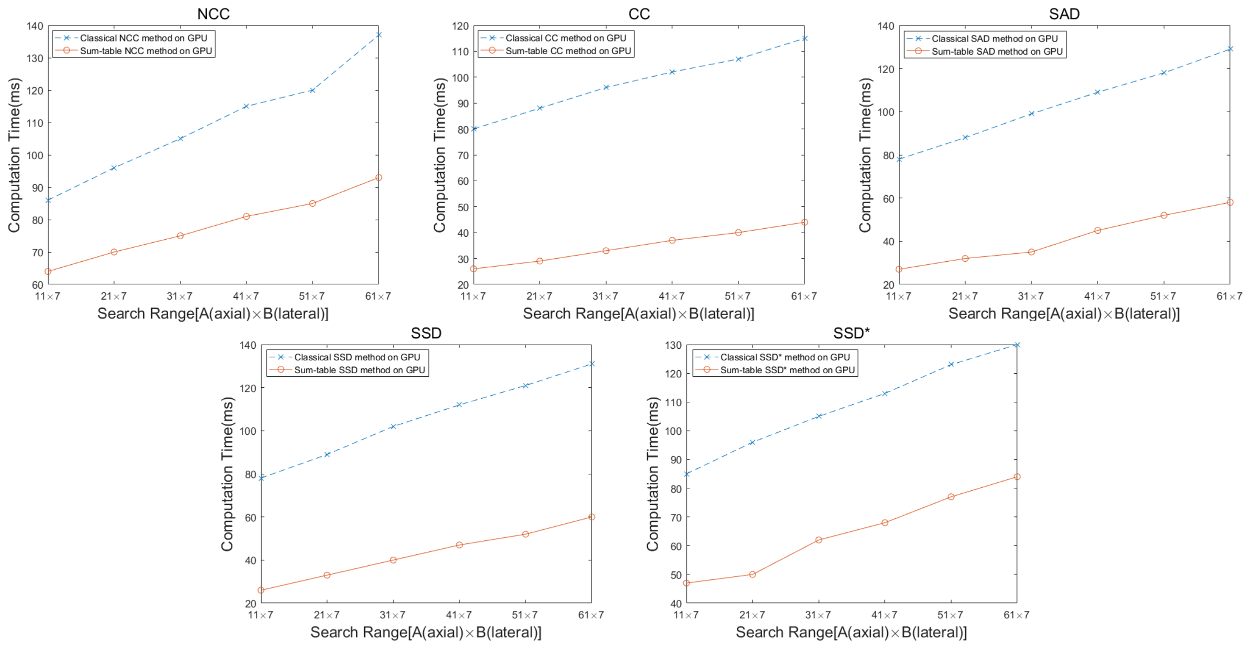


Fig. 4. Computing time when the search range changed with different motion estimation methods.

process of sum-table construction is implemented on CPU. In the case of two-dimensional data, it is not necessary to complete the sum-table construction on GPU, but multiple sum-table construction will still consume a certain amount of time.

In a large tracking kernel window, the sum-table method possesses good performance. In a huge search range, the sum-table method does not show advantage over the classical method on GPU platform. As the total amount of sum-table construction increases, the computational efficiency will decrease.

In short, on GPU platform, the sum-table method is very suitable for displacement tracking applications with a large tracking kernel window and a small search range. This conclusion is applicable to all motion estimators mentioned in this paper.

Acknowledgment

The author would like to thank Prof. Jianwen Luo from Tsinghua University for his guidance on the preliminary work of this paper.

References

[1] T. Shiina, K.R. Nightingale, M.L. Palmeri, T.J. Hall, J.C. Bamber, R.G. Barr, et al. “WFUMB Guidelines and Recommendations for Clinical Use of Ultrasound Elastography: Part 1: Basic Principles and Terminology.” Ultrasound Med. Biol. vol. 41, no. 5, pp. 1347–1357, May 2015.

[2] H. Rivaz, E. Boctor, P. Foroughi, R. Zellars, G. Fichtinger, and G. Hager, “Ultrasound Elastography: A Dynamic Programming Approach,” IEEE Trans. Med. Imag., vol. 27, no. 10, pp. 1373–1377, June 2008.

[3] F. Viola and W. F. Walker, “A Comparison of the Performance of Time-Delay Estimators in Medical Ultrasound,” IEEE Trans. Ultrason.Ferroelectr.Freq. Control, vol. 50, no. 4, pp. 392–401, Apr. 2003.

[4] J. Luo, and E.E. Konofagou, “A fast normalized cross-correlation calculation method for motion estimation,” IEEE Trans. Ultrason. Ferroelectr. Freq. Control, vol. 57, no. 6, pp. 1347–1357, June 2010.

[5] J. P. Lewis, “Fast template matching,” Vision Interface, pp. 120C123, May 1995.

[6] J. Luo, and E.E. Konofagou, “A fast motion and strain estimation method,” in Proc. IEEE Int. Ultrason. Symp., pp. 1608-1611, Oct. 2010.

[7] P. Bo, L. Shasha, and X. Zhengqiu, “Accelerating 3-D GPU-based Motion Tracking for Ultrasound Strain Elastography Using Sum-Tables: Analysis and Initial Results,” APPL SCI-BASEL, vol. 9, no. 10, pp. 1991, May 2019.

[8] S. Rosenzweig, M. Palmeri, and K. Nightingale, “GPU-based real-time small displacement estimation with ultrasound,” IEEE Trans. Ultrason. Ferroelectr. Freq. Control, vol. 58, no. 2, pp. 399C-405, Feb. 2011.

[9] L. Chang, K. Hsu, and P. Li, “GPU-based color Doppler ultrasound processing,”in Proc. IEEE Int. Ultrason. Symp., pp. 1836-1839, Sep. 2009.

[10] T. Idzenga, E. Gaburov, W. Vermin, J. Menssen, and C. L. de Korte, “Fast 2-D ultrasound strain imaging: the benefits of using a GPU,” IEEE Trans. Ultrason.Ferroelectr.Freq. Control, vol. 61, no. 1, pp. 207–213, .

[11] J. A. Jensen, “Field: A program for simulating ultrasound systems,” Med. Biol. Eng. Comp., Vol. 4, Supplement 1, pp. 351–353, 1996b [10th Nordic-Baltic Conference on Biomedical Imaging, 1996].