Acceleration of Medical Image Registration using Graphics Process Units in Computing Normalized Mutual Information

Wei-Hung Cheng and Cheng-Chang Lu
Department of Computer Science
Kent State University
Kent, OH 44242

Abstract

This paper presents a computational performance analysis of an accelerated medical image registration using Graphics Processing Units (GPUs). In our previous work, a multi-resolution approach using normalized mutual information (NMI) has proven to be useful in medical image registration. In this paper, we propose an acceleration of the NMI procedure using GPU implementation because of the parallel processing capabilities. Registration algorithms were implemented on NVIDIA's GeForece 9600 GT graphic processor with the Compute Unified Device Architecture (CUDA) programming environment. Experimental results showed that the GPU implementation improves the registration computational performance with a speedup factor of 38. In addition, the maximum speedup can be achieved with diligent data profiling.

1 Introduction

Image registration is the procedure of aligning images as their features can be corresponding between two images.Image registration is important in medical image applications and can be applied to medical images with different modalities. Various approaches have been developed to improve the performance of the medical image registration in terms of speed optimization, capture range, and robustness of the method[4]. In the past, most medical registration algorithms were developed and implemented on CPU platforms. The growing popularity of using GPU platforms for accelerated computing in the scientific research demonstrated the potential improvement of the computation performance in the medical image registration[6]. In this work, we propose to use the CUDA programming model with a GPU platform for medical image registration. We first describe an approach of multi-resolution registration using normalized mutual information. Section IV illustrates the CUDA programming model on GPU platform used for parallel implementation of the registration approach. Section V shows the simulation result and the conclusion is drawn in Section VI.

2 Multi-resolution Approach Registration

Registration based on maximization of mutual information uses an iterative approach in which an initial estimate of the transformation is gradually refined. One of the difficulties with this approach is that it can converge to a local optimum instead of a global optimum. Using multi-resolution in conjunction with maximization of mutual information has been proven to be very helpful with this problem. Multi-resolution is supplemented with binarization giving improved accuracy with no loss of speed. Two levels of registration are used. Before the registration process images are first segmented into background and foreground by region growing[7][2]. There are two criteria for a voxel to be annexed to the background: (1) The absolute intensity difference between any voxel and the seed must be less than a threshold. The threshold for the region is determined by the histogram. (2) To be included in the background, a voxel must be adjacent to at least one voxel in the background. Adjacency includes the 8 directions: N, S, E, W, NE, NW, SE, and SW. The background voxels are then put into one bin, and the foreground into the another bin. Since the background is large the initial seed is automatically chosen to be a voxel near a corner. The background/foreground segmentation is fully automatic and takes about 2% of the total registration time. The binarized, 2-bin images are down-sampled by a factor of 2 along the x, y and z axis directions. This down-sampled image is then used as the input to the first level of the multi-resolution registration. The result of the first level provides the initial estimate for the second level. The second level performs registration of the original images.

3 Normalized Mutual Information

Mutual information (MI) can be thought of as a measure of how well one image explains the other and when maximized indicates optimal alignment.

Mutual information $I(A, B)$ was proposed for intermodality medical image registration by several researchers[1][3][10]. The formula to compute mutual information is:

$$I(A,B) \quad = \quad \sum_a \sum_b P_{AB}(a,b) \log \frac{P_{AB}(a,b)}{P_A(a).P_B(b)}$$

where $P_A(a)$, $P_B(b)$ denote the marginal probability distributions of the two random variables, A and B, and $P_{AB}(a, b)$ is their joint probability.

In this study, we are using the formulation of normalized mutual information (NMI) as described by Studholme:[9]

$$NMI(A,B) = \frac{H(A) + H(B)}{H(A,B)}$$

where

$$H(A) = -\sum_a P_A(a) \log P_A(a)$$

and

$$H(A,B) = -\sum_a \sum_b P_{AB}(a,b) \log P_{AB}(a,b)$$

$H(A)$ and $H(B)$ are the entropies of A and B, and $H(A, B)$ is their joint entropy. NMI, devised to overcome the sensitivity of MI to change in image overlap, has been shown to be more robust than standard mutual information[9]. A general flowchart of the implementation of NMI registration is given in Figure 1.

When the registration parameters (3 translations and 3 rotations) are applied to the floating image (Fig. 1), the algorithm iteratively transforms the floating image with respect to the reference image while optimizing the NMI measure calculated from the voxel intensities. While all samples are taken at grid points of the floating image, their transformed position will in general not coincide with a grid point of the reference image and interpolation of the reference image is needed to obtain the image intensity value at this point.

The NMI registration criterion states that the images are geometrically aligned by the transformation T_λ for which $NMI(A, B)$ is maximal. (The registration parameters are denoted by λ and T_λ which is the transformation based on the six registration parameters given above.) To decide if NMI is maximal, some optimization algorithms are applied. If it is not, a new set of parameters will be evaluated. In this paper, we take the Downhill simplex as the optimization approach. The method, as shown by Nelder and Mead, only requires evaluation of the cost function, the derivative computation need not be redone[8].

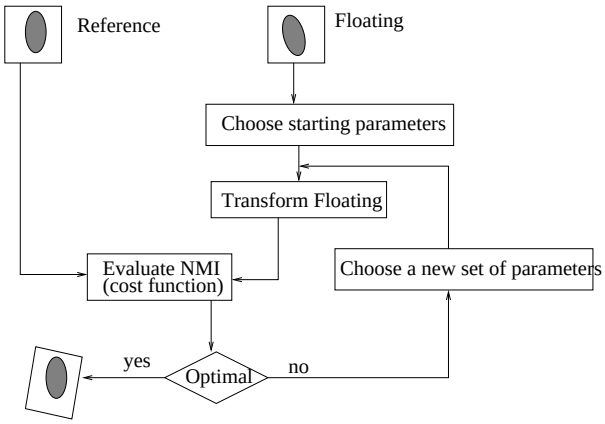

Figure 1. Flow chart of image registration using the maximization of normalized mutual information (NMI).

4 GPUs and CUDA Programming Model

The GPU platform is gaining popularity in performing computation intensive tasks and is well-suited for the data-parallel computation applications.

The CUDA (Compute Unified Device Architecture) programming, developed by NVIDIA, is a parallel programming model and software environment designed to handle parallel computing tasks. Its major abstractions are a hierarchy of thread groups, shared memories, and barrier synchronization. These abstractions provide a programming model for data parallelism, thread parallelism, and task parallelism. The CUDA presents a structure to the GPU programmer that includes a collection of threads to run in parallel. This structure is similar to the traditional single instruction, multiple data (SIMD) parallel model. The computation paradigm of the CUDA[5] is described as follows:

- A program is divided into *blocks* that can run in parallel. A block is a group of threads mapped to a single multiprocessor by the programmer to share the memory. All available resources are divided equally amongst all threads of concurrent blocks on a single multi-processor. Moreover, the data is also divided amongst all threads in a SIMD fashion by the programmer.

- All threads are organized into *warps*. Each warp is a group of 32 parallel scale threads, which can run concurrently on the multi-processors. Moreover, each

warp executes one kernel instruction at a time. Collections of *warps* are known as *thread block* and these threads run on the same multiprocessor and use a part of the local storage. Therefore, the programmer can freely design the number of threads to be executed.

- The GPU processes many threads in parallel using the *kernel* on each one of them. A *kernel* is the actual code to be executed by threads; the executable is shared among all of the threads in the system. In order to identify the conditional execution of operations on multi-processors, the programmer can process each on a unique thread ID.

- Memory hierarchy in the CUDA programming environment is in the form of registers, constant memory, global memory, and textures. The register file is the fastest level in hierarchy but only has a limited amount of space. Constant memory is a subset of device memory which is shared by the processors and cannot be modified at run-time by a device. Global memory permits read and write operation from all threads, but is uncached and has long latencies. Texture memory is a subset of the device memory; it is read-only on the device, provides faster cached reads, and allows addressing through a specialized texture unit.

5 Implementation and Experiments

This study involved the data sets of 7 patients, each consisting of Computed Tomography (CT) and six Magnetic-Resonance (MR) volumes (spin-echo T1, PD, T2 and the rectified version of each of these three). The MR-T1-rectified image was not available for patient 6. Thus, 41 image pairs were available to be registered. These images were provided by *The Retrospective Registration Evaluation Project* database maintained by J. Michael Fitzpatrick from Vanderbilt University, Nashville, TN, USA. The characteristics of these images are summarized in Table 1. For each patient data set, all CT images were registered to the MR image using MR as the reference image. All image pairs for image registration processes were using a mutual information method on a PC, having a 2.40 GHz Intel(R) Core(TM) 2 Quad CPUs, and 4 GB DDR2 memory with NVIDIA's GeForce 9600 GT graphic card. Image registration MI algorithm were programming in C for the CPU platform, and in CUDA for the GPU platform for performance comparison.

Table 1. Image Characteristics

Image	Size	Voxels(mm)
CT	512^2 x (28-34)	0.653595^2 x 4.0
MR PD	256^2 x (20-26)	1.25^2 x 4.0
MR PD Re.	256^2 x (20-26)	$(1.25-1.26)^2$ x (4.04-4.11)
MR T1	256^2 x (20-26)	1.25^2 x 4.0
MR T1 Re.	256^2 x (20-26)	$(1.25-1.26)^2$ x (4.04-4.12)
MR T2	256^2 x (20-26)	1.25^2 x 4.0
MR T2 Re.	256^2 x (20-26)	$(1.25-1.27)^2$ x (4.04-4.16)

5.1 CUDA Algorithm

It is observed that the registration procedure spent 90-95% of its run-time on mutual information computation. Medical image registration has a high level of data parallelism and image data can be mapped onto the GPU platform. In order to implement image registration algorithms with the CUDA programming environment, it is necessary to identify the tasks for data-parallel as CUDA kernels. Based on our normalized mutual information method, the tasks are classified into four CUDA kernels as follows:

1. Transformation: This group performs coordinate transform, affine transform, and mapping matrix to establish spatial correspondence between two images.

2. Interpolation: This group involves iteratively transforming image A with respect to image B while optimizing the MI measure which is calculated from corresponding voxel values. While all samples are taken at grid points of the floating image A, their transformed position will in general not coincide with a grid point of the reference image B so interpolation of the reference image is needed to obtain the image intensity value at this point. Three commonly used interpolation methods are implemented in our experiments such as Nearest Neighbor Interpolation, Trilinear Interpolation, and Trilinear Partial Volume Interpolation.

3. Histogram: This group computes a joint histogram of the pairs of images to evaluate the mutual information.

4. Optimization: This group detects optimization of estimate transformation to evaluate its similarity, and repeats the process until the algorithm converges to an optimal value.

The NMI registration criterion states that the images are geometrically aligned by the transformation T_λ for which $NMI(A, B)$ is maximal. The registration parameters are denoted by λ and T_λ which is the transformation based on the six registration parameters. If NMI is not optimal, a new set of transformation parameters will be chosen and this process repeats. We designed the pipeline sequences to map with CUDA kernels to speedup the performance of NMI registration.

5.2 Performance

In the simulation, the comparison of NMI computation using the CUDA implementation and CPUs implementation with the data set of 41 pairs of CT-MR (7 patients) is conducted. Experimental results showed that the GPU implementation improves the registration computational performance with a speedup factor of 38. In addition, a better performance can be achieved by reducing memory accesses from kernels and kernels call from functional computation.

6 Conclusion

In this paper, medical image registration using the NMI criterion with CUDA programming environment is proposed. We focused on implementing NMI algorithms with functional data-parallelism to speedup the medical image registration process. Experimental results showed that the GPU implementation improves the registration computational performance with a speedup factor of 38. In addition, the maximum speedup can be achieved with diligent data profiling.

References

[1] A. Collignon, F. Maes, D. Delaere, D. Vandermeulen, P. Suetens, and G. Marchal. Automated multi-modality image registration based on information theory. *Information Process. Med. Imaging*, pages 263–274, 1995.

[2] R. Gonzalez and R. Woods. *Digital Image Procesing*. Prentice Hall Press, 2 edition, 2002.

[3] F. Maes, A. Collignon, D. Vandermeulen, G. Marchal, and P. Suetens. Multimodality image registration by maximization of mutual information. *IEEE Trans. Med. Imaging.*, 16(2):187–198, 1997.

[4] J. B. A. Maintz and M. A. Viergever. A survey of medical image registration. *Medical Image Analysis.*, 2(1):1–36, 1998.

[5] NVIDIA. Nvidia cuda compute unified device architecture. *Programming Guide*, Version 2.0. NVIDIA, 2008.

[6] J. D. Owens, M. Houston, D. Luebke, S. Green, J. E. Stone, and J. C. Phillips. Cpu computing. *Proceedings of the IEEE*, 96(5):879–899, 2008.

[7] D. L. Pham, C. Xu, and J. L. Prince. Current methods in medical images segementation. *Annual Review of Biomedical Engineering*, 2, 2000.

[8] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. Numerical recipes in c: The art of scientific computing. 2nded. *Cambridge, U.K.:Cambridge University Press*, pages 394–455, 1999.

[9] C. Studholme, D. L. G. Hill, and D. J. Hawkes. An overlap invariant entropy measure of 3d medical image alignment. *Pattern Recognition.*, 32(1):71–86, 1999.

[10] W. M. WellIII, P. Viola, H. Atsumi, S. Nakajima, and R. Kikinis. Multi-modal volume registration by maximization of mutual information. *Medical Image Analysis.*, 1(1):35–51, 1996.