In Proceedings of the IEEE Biomedical Circuits and Systems Conference,
Montreal, Canada, November 2007.

Towards a Heterogeneous Medical Image Registration Acceleration Platform

William Plishker[1,2], Omkar Dandekar[1,2], Shuvra Bhattacharyya[2], and Raj Shekhar[1]
[1] University of Maryland, Baltimore
[2] University of Maryland, College Park
{plishker, omkar, ssb}@umd.edu, rshekhar@umm.edu

Abstract— **For the past decade, improving the performance and accuracy of medical image registration has been a driving force of innovation in medical imaging. Accurate image registration enhances diagnoses of patients, accounts for changes in morphology of structures over time, and even combines images from different modalities. The ultimate goal of medical image registration research is to create a robust, real time, elastic registration solution that may be used on many modalities. With such a computationally intensive and multifaceted problem, researchers have exploited parallelism at different levels to improve the performance of this application, but there has yet to be a solution fast enough and effective enough to gain widespread clinical use. To achieve real time elastic registration, an implementation must simultaneously exploit multiple types of parallelism in the application by targeting a heterogeneous platform whose computational components (e.g. multiprocessors, graphics processors, field programmable gate arrays) match these types of parallelism. Our initial experiments indicate that an 8 node heterogeneous cluster can realize over 100x speedup compared to a high performance uniprocessor system. By creating a platform based on modern hardware, we believe that a heterogeneous compute platform customized for image registration can provide robust, scalable, cost effective sub-minute medical image registration capabilities.**

Index Terms—**Image Registration, Parallelism, Medicine.**

I. Introduction

Advances in medical imaging technologies have enabled diagnoses and procedures simply not possible a decade ago. The increasing acquisition speed and the improving resolution of images have given doctors more information, less invasively about their patients. However, because of the multitude of imaging modalities (e.g. computed tomography (CT), positron emission tomography (PET), magnetic resonance (MRI), ultrasound (US), etc.) and the sheer volume of data being acquired, utilizing this new data effectively has become problematic. To tap into the potential of this raw data, these images can be merged into one integrated view through a procedure called *image registration.*

Image registration is the process of fusing two images such that the features in one image are aligned with the features in another. Beyond simply correcting for whole image alignment and shifting (called rigid registration), a robust algorithm can register three dimensional (3D) images between different modalities (e.g. CT with PET or CT with MRI), account for changes in morphology over time, and also adjust for internal tissue motion (e.g. liver deformation due to breathing), called elastic registration. Fully automatic elastic registration is computationally intensive, requiring hours or even days to solve typical sized problems using high-end processors. This has fueled a significant body of research into image registration acceleration in an effort to bring more accurate and more robust image registration techniques into the clinical setting. While many of these works have shown important performance enhancing techniques, to our knowledge no work in medical image registration has done an integrated comparison to consider the synergistic effects of combining them on a customized heterogeneous computational platform.

Since no single technique has accelerated the problem sufficiently for all clinical applications, we believe true real time image registration will require the utilization of multiple levels of parallelism. The image registration description should expose this parallelism in the application and be implemented on a heterogeneous computational platform made up of building blocks which are suited to the expressed parallelism. The remainder of this paper will cover background related to image registration acceleration, present our experiments utilizing different levels of parallelism, and discuss what the possible performance benefits of a heterogeneous image registration platform would be.

II. Medical Image Registration

Medical image registration approaches have appeared in a variety of contexts often specialized for a particular modality or anatomy. Despite the variation, there are commonalities between these approaches. The objective of image registration is to find a transformation to apply to a *floating image* so that it best aligns to a *reference image*. First a transformation is generated through analytical, heuristic, or manual means. While transformations can be based on feature detection, we focus on "area-based" methods, which rely on pixel intensity correlation without using assumptions about the underlying anatomy. Area-based methods tend to be more robust and computationally intensive and consequentially are a natural fit for pure acceleration techniques.

A transformation is often described as a deformation field, in which all parts of the floating image space have a specific deformation. Construction of the deformation field can start from a just few parameters in the case of rigid registration or from a set of "control points" which capture the non-uniformity of elastic registration. The final transformation contains the information necessary to deform all of the voxels (3D pixels) in the floating image.

Once a transformation is constructed, it is applied to the floating image. This transformed image can be compared to the reference image using a variety of *similarity* metrics, such as the sum of intensity differences or mutual information. For iterative approaches, the similarity value is returned to the procedure which created the transformation so that it may guide it towards successively better solutions. Problem parameters may strategically change while running to improve run time and accuracy (e.g. rigid registration may be run before elastic registration or the control point grid resolution may change such that a coarse elastic registration occurs before a fine grained one).

III. Exploiting Multiple Forms of Parallelism

While image registration is a computationally intensive problem, it can be readily accelerated by exploiting parallelism. We chose to implement a few of the more popular techniques for a well known gradient descent algorithm [1] on different architectures that could be used in a future heterogeneous platform.

For the highest level of parallelism, we used the Message Passing Interface (MPI) [2], which is a popular standard for explicitly parallelizing code on multiprocessor systems. Threads have local memory that can be readily implemented on distributed memory platforms such as clusters. Using MPI, we distributed the gradient computation equally across nodes in a small cluster similar to [3]. This distribution is possible by virtue of the fact that each finite difference calculation for each control point is independent and requires only the neighboring voxels and control points to calculate.

To parallelize the transformation step at the voxel level, we used a graphics processor (GPU), which is an array of processing elements customized for pixel processing. The increasing programmability of GPUs have opened them up for many other applications including image registration [4]. High level languages are emerging to aid the task of programming GPUs and in this work we utilized Brook [5]. We used Brook to describe the rigid transformation of the floating image space into the reference image space as a streaming operation to be applied to each voxel. As an independent streaming operation, the task could be efficiently distributed across the processing elements of the GPU.

For the lowest level of parallelism, we employed a hardware description language (HDL) [6]. With HDLs the final implementation is not destined for a processor, so designers layout their application structurally, exposing interfaces and cycle-by-cycle control. We utilize a field programmable gate array (FPGA) to accelerate the voxel processing of the elastic transformation and the similarity calculation [7]. The independence of tasks allows for many memory accesses, operations, and IO to be performed in the same clock cycle.

Since these acceleration techniques are independent of each other and implemented on different types of platforms, a heterogeneous computational platform that supported all of these approaches would create a powerful new image registration engine. Orthogonal acceleration techniques such as the FPGA and MPI techniques we employ, can provide multiplicative speedup effects. In the following section we attempt to quantify what speedups can be expected.

IV. Experiment Results

To support our claims on the potential synergy in combining levels of parallelism, we have accelerated a widely used image registration algorithm on computational building blocks likely to make up a future heterogeneous image registration platform: FPGAs, multiple CPUs, and GPUs. While it is possible to extract performance numbers from previous authors' works, we come as close a possible to simulating realizable performance potential by incorporating each of the acceleration techniques in the same code base.

The base algorithm was the gradient descent method described by Reuckert et al. [1] using mutual information, which is effective for multimodal registration. Using this single code base, acceleration techniques were incorporated that were wrapped by preprocessing directives. Therefore, at compile time the software could be targeted for a specific parallel platform: an FPGA accelerated PC, a cluster of PCs using varying numbers of nodes, or a GPU accelerated PC. The PC cluster and the uniprocessor results were generated on a 3GHz Intel Xeon. The FPGA accelerator was a PCI board (Tsunami, SBS Technologies, Albuquerque, NM) with an Altera Stratix EP1S40 FPGA and two on-board 512 MB SDRAMs for storing the reference and floating images, which were accessed via a 32-bit bus. The GPU tested was the NVIDIA Quadro FX 1400.

The implementations were tested with five pairs of clinically gathered PET and CT images of the torso. Each of the five cases was visibly misaligned and some had non-overlapping regions. Each image was resampled to be 128×128×128 such that the voxels were roughly isotropic. First a rigid registration of the two images was done followed by elastic registration performed with a deformation field represented by a 9×9×9 control point grid. The gradient step size was based on linear feedback and the algorithm stopped when no improvement could be found at a specified minimum step size. The results of each registration were visually inspected as well and in each case improvement was seen.

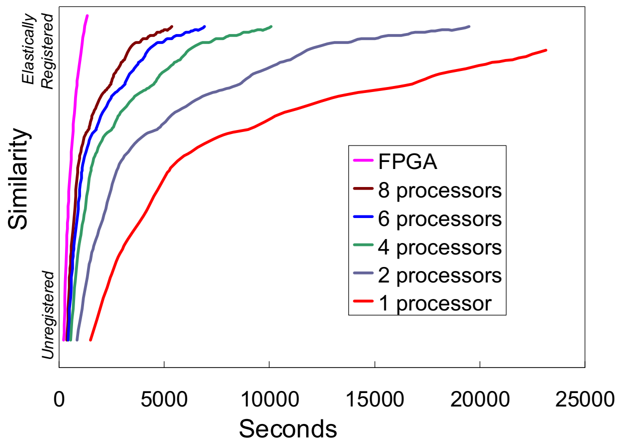

Fig. 1. Optimization traces of one case of registration.

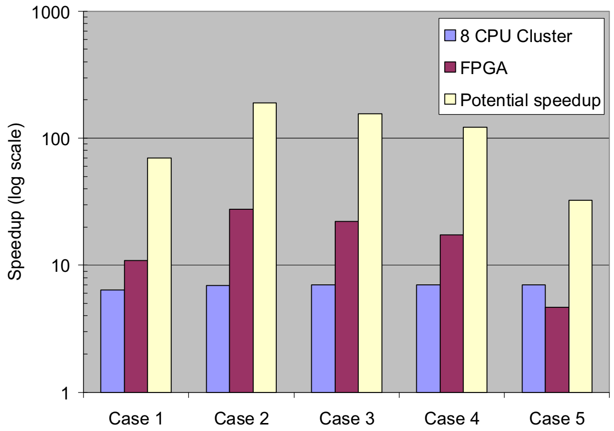

Fig. 2. Final speedups from acceleration techniques.

Each platform was run with these identical set of parameters, which produced optimization traces such as the example shown in Fig. 1. Each line represents the execution of a different implementation of the image registration algorithm, and the right most line represents the base case of a uniprocessor implementation. Points on the line represent the similarity between the two images with the best known transformation at that time. As expected, the speedups of the cluster implementations were near-linear with respect to the number of processors. The FPGA arrives at the result the fastest, but because of a difference in the precision of the datapath instantiated, it takes a slightly different path to the final transformation.

The final speedups are shown in the log graph in Fig. 2. By imagining a heterogeneous compute cluster with an implementation that takes advantage of both of these kinds of parallelism (see Fig. 3), we have also extrapolated the potential speedup of combining these two approaches. Since the two types of parallelism being exploited are on separate levels, this is a multiplicative speedup. Note that while the software speedup is relatively constant, the FPGA's different path through the optimization space leads to varying degrees of acceleration. Each iteration step is consistently processed faster than software, but for some cases it requires more steps to arrive at the final solution.

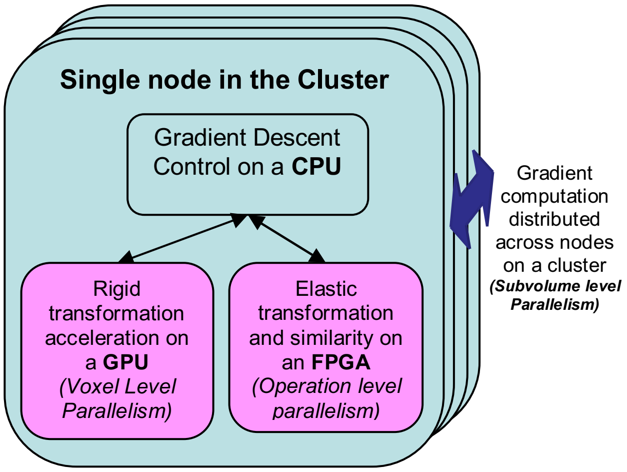
Single node in the Cluster
Gradient Descent Control on a CPU
Rigid transformation acceleration on a GPU (Voxel Level Parallelism)
Elastic transformation and similarity on an FPGA (Operation level parallelism)
Gradient computation distributed across nodes on a cluster (Subvolume level Parallelism)

Fig. 3. An image registration implementation on a heterogeneous computational platform.

We utilized a GPU for the rigid registration transformation step, producing a 3x rigid registration speedup over a uniprocessor only implementation. Since our GPU implementation currently accelerates rigid registration only, its effect on the overall elastic registration is limited, but with a more powerful GPU and integration into elastic registration, we believe it will be an important addition to a future platform. While it utilizes some of the same parallelism as an FPGA, kernels offloaded to it would make more room for other tasks on the FPGA, thus permitting synergistic effects between the two.

V. Related Work

Utilization of parallelism is the most common form of general application performance improvement. As there are variety of parallel platforms offering acceleration opportunities, researchers have identified and exploited different kinds of parallelism in existing algorithms. We discuss some of these approaches by using our own categorization of parallelism specifically designed for image registration.

Optimization level parallelism represents those parts of an algorithm that can run in parallel given the basic unit is an iteration of the image registration routine such as gradient decent invocations from different starting points [3] or with a distributed genetic algorithm [8]. *Volume level parallelism* is a generalization of optimization level parallelism where the computational units operate on entire volumes like computing a multidimensional gradient on whole image volumes [3]. In medical image registration *subvolume level parallelism* is perhaps the most popular kind of parallelism to exploit, where an application is parallelized by dividing the volume level work into smaller subvolumes that are later recombined to produce the final solution. [3, 9-16]. *Voxel level parallelism* describes parallelism in terms of single voxels, which is a good match to GPUs [4] and with threads using a workpile [17]

Operational level parallelism is the lowest, most general form of parallelism. Beyond taking advantage of instruction level parallelism transparently on a modern processor, operation level parallelism is the most difficult to utilize. Only hardware platforms are suitable to effectively exploit this level of parallelism [7, 18, 19]. But if an application designer has the expertise to utilize it, significant performance gains can be realized. Each of these works has the potential to be combined for additive or multiplicative performance benefits on a heterogeneous platform.

VI. Conclusion and Future Work

To achieve our ultimate goal of real time robust accurate image registration with medical images that keep increasing in resolution, we will need more than Moore's Law. Combining optimization techniques and developing new heterogeneous platforms can provide the technology that is capable of this clinical breakthrough. In this work we have shown the potential of combining approaches that exploit different kinds of parallelism in medical image registration. Our experiments indicated that using just an 8 node heterogeneous platform could produce up to a 190x speedup over a high performance, general purpose uniprocessor. Such unprecedented speedups for this size cluster mean that clinicians might have robust elastic registration that normally takes 7 hours to only require a couple minutes.

These projected speedups could be thought to be too optimistic as an actual implementation utilizing them both will incur some additional overhead and might have system bottlenecks not captured here. Conversely, these numbers may be conservative as the technology used is a generation behind the baseline uniprocessor it is compared to. Furthermore, the MPI and GPU acceleration techniques themselves were relatively simple. While in line with previously published speedup factors, new innovative techniques tailored to the platform could improve upon those factors. We plan to construct a prototype heterogeneous platform to further examine these issues.

As more complex acceleration techniques are combined, a more rigorous system of capturing the parallelism of the application will be needed. Such a framework would give programmers a more natural way of expressing each type of parallelism without having to dive into the idiosyncrasies of GPU languages or HDLs, for example. Moreover, designers should be able to weigh their decisions on parallelism in a flexible way, allowing kernels to migrate or to be distributed on different parts of the platform. This effort will require a robust and flexible language capable of capturing and reasoning about parallelism at many levels.

VII. Acknowledgements

This work was made possible by the Department of Defense grant DAMD17-99-1-9034 and NSF award CISE CNS 0403313. We would like to express our thanks for the assistance of the UMIACS staff, the Imaging Technologies Laboratory, and the DSPCAD group.

References

[1] D. Rueckert, L. I. Sonoda, C. Hayes, D. L. G. Hill, M. O. Leach, and D. J. Hawkes, "Nonrigid registration using free-form deformations: application tobreast MR images," *Medical Imaging, IEEE Transactions on*, vol. 18, pp. 712-721, 1999.

[2] W. Gropp, E. Lusk, N. Doss, and A. Skjellum, "A High-Performance, Portable Implementation of the MPI Message Passing Interface Standard," *Parallel Computing*, vol. 22, pp. 789-828, 1996.

[3] F. Ino, Y. Kawasaki, T. Tashiro, Y. Nakajima, Y. Sato, S. Tamura, and K. Hagihara, "A Parallel Implementation of 2-D/3-D Image Registration for Computer-Assisted Surgery," in *Proceedings of the 11th International Conference on Parallel and Distributed Systems - Workshops (ICPADS'05) - Volume 02*: IEEE Computer Society, 2005.

[4] R. Strzodka, M. Droske, and M. Rumpf, "Fast Image Registration in DX9 graphics Hardware," *Journal of Medical Informatics and Technologies*, pp. 6:43-49, 2003.

[5] I. Buck, T. Foley, D. R. Horn, J. Sugerman, K. Fatahalian, M. Houston, P. Hanrahan, and, "Brook for GPUs: Stream Computing on Graphics Hardware," *ACM Transactions on Graphics (Proc. SIGGRAPH 2004)*, 2004.

[6] "IEEE standard Verilog hardware description language," 2001.

[7] C. R. Castro-Pareja, J. M. Jagadeesh, and R. Shekhar, "FAIR: a hardware architecture for real-time 3-D image registration," *Information Technology in Biomedicine, IEEE Transactions on*, vol. 7, pp. 426, 2003.

[8] T. Butz and J.-P. Thiran, "Affine registration with feature space mutual information," in *Medical Image Computing and Computer-Assisted Intervention*, vol. 2208, *Lecture Notes in Computer Science*, W. J. Niessen and M. A. Viergever, Eds. Berlin, Germany: Springer-Verlag, 2001, pp. 549–556.

[9] T. Rohlfing and C. R. Maurer, "Nonrigid image registration in shared-memory multiprocessor environments with application to brains, breasts, and bees.," *IEEE Transactions on Information Technology in Biomedicine*, vol. 7, pp., 2003.

[10] S. Ourselin, R. Stefanescu, and X. Pennec, "Robust registration of multimodal images: towards real-time clinical applications," *Medical ser. LNCS*, 2002.

[11] F. Ino, K. Ooyama, and K. Hagihara, "A data distributed parallel algorithm for nonrigid image registration," *Parallel Computing*, vol. 31, pp. 19-43, 2005.

[12] J A. Schnabel and e. a. "D. Rueckert, "A Generic Framework for Non-rigid Registration Based on Non-uniform Multi-level Free-Form Deformations," in *Proceedings of the 4th International Conference on Medical Image Computing and Computer-Assisted Intervention*: Springer-Verlag, 2001.

[13] O. Dandekar, V. Walimbe, and R. Shekhar, "Hardware Implementation of Hierarchical Volume Subdivision-based Elastic Registration," presented at IEEE EMBS, 2006.

[14] Radu Stefanescu, Xavier Pennec, and N. Ayache, "Parallel non-rigid registration on a cluster of workstations," *Proc. of HealthGrid'03*, 2003.

[15] J. P. Thirion, "Non-rigid matching using demons," presented at IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Proceedings CVPR '96, San Francisco, CA, USA, 1996.

[16] R. Stefanescu, X. Pennec, and N. Ayache, "Grid powered nonlinear image registration with locally adaptive regularization," *Med Image Anal.*, vol. 8, pp. 325-42, 2004.

[17] K. Warfield Simon, A. Jolesz Ferenc, and R. Kikinis, "A high performance computing approach to the registration of medical imaging data," *Parallel Comput.*, vol. 24, pp. 1345-1368, 1998.

[18] R. Shekhar, V. Zagrodsky, C. R. Castro-Pareja, V. Walimbe, and J. M. Jagadeesh, "High-speed registration of three- and four-dimensional medical images by using voxel similarity," *Radiographics*, vol. 23, pp. 1673-81, 2003.

[19] O. Dandekar and R. Shekhar, "FPGA-accelerated Deformable Registration for Improved Target-delineation During CT-guided Interventions," *IEEE Transactions on Biomedical Circuits and Systems*, 2007.