In Proceedings of the IEEE-NIH LIfe Science Systems and Applications Workshop, pages 215-218, Bethesda, Maryland, November 2007.

A Taxonomy for Medical Image Registration Acceleration Techniques

William Plishker[1,2], Omkar Dandekar[1,2], Shuvra Bhattacharyya[2], and Raj Shekhar[1,2]
[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. Fast, accurate image registration enhances diagnoses of patients, accounts for changes in morphology of structures over time, and even combines images from complementary modalities (e.g. computed tomography and positron emission tomography). The ultimate goal of medical image registration research is to create a robust, real time, elastic registration solution that may be used for multiple modalities such that new image-guided intervention techniques can be pioneered. With such a computationally intensive and multifaceted problem, researchers have applied a variety of innovative approaches at different levels of the problem to improve the performance of this solution. This diversity in work presents many opportunities for synergies between complementary techniques, but to date has mostly gone unrealized. This paper samples existing literature on image registration acceleration techniques and proposes a classification of "levels" into which they can be binned. We show experimentally that combining approaches from different levels can lead to over 100x speedup on an eight node heterogeneous cluster.**

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

I. Introduction

Advances in medical imaging technologies have enabled medical diagnoses and procedures simply not possible a decade ago. The improving speed of acquisition and the increasing resolution of images have given doctors more information, less invasively about their patients. However, because of the many imaging modalities and the sheer volume of data being acquired, utilizing this new data effectively has become problematic.

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 alignment and shifting (called *rigid registration*), a robust image registration algorithm can register three dimensional (3D) images between different modalities (e.g. computed tomography (CT) & positron emission tomography (PET) or CT & magnetic resonance imaging (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*. Unfortunately, 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 robust image registration techniques into the clinical setting. While some solutions exist for simple registration, there is no widely used robust elastic registration software solution in clinical practice due in large part to the long execution times of existing implementations.

This work samples the existing literature on medical image registration acceleration techniques to derive a taxonomy to facilitate synergies between complementary works. While more general purpose taxonomies exist for describing acceleration techniques, a domain specific taxonomy is able to capture those features which will give significant performance improvements to this computationally intensive problem. We believe this is the first step towards an image registration specific framework that could be ultimately formalized into an interoperability framework such that complementary research efforts could be combined. The remainder of this paper will cover background related to acceleration in image registration and describe our taxonomy using representative works for illustration as this is not an exhaustive survey.

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 only pixel intensity correlation. 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 (as with rigid registration) or from a set of "control points" which capture the non-uniformity of elastic registration. The final transformation deforms 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. Our Taxonomy

While acceleration techniques in general may modify functionality, we focus on the categorization of techniques which rely on parallelism for performance improvements. As with classical general purpose categorizations, we classify acceleration techniques into "levels," which are depicted in Fig. 1. Like the classical levels of parallelism (bit, data, instruction, task/thread, and process level), higher levels are specializations (or restricted forms) of lower levels. Conversely, it tends to be easier to reap the rewards of higher level parallelism than lower. Many architectural and application factors affect this tendency both positively and negatively (e.g. communication patterns, memory sizes, topology, compiler performance), but for typical applications, the higher the level of parallelism expressed, the more readily applications can be accelerated. The ongoing push to multicore architectures is one instance of this observation. For example, task level parallelism is better implemented on a multiprocessor with simple processing elements as it avoids the architectural overhead of instruction level parallelism.

A. Optimization level parallelism

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. Ino, et al. [1] use this idea (which they call "Speculative parallelism") to promote faster convergence in their time critical registration application. Since the best optimization parameters are difficult to know *a priori*, multiple instances of the same algorithm are launched with different parameters defined then the best solution is taken after a specified time period. The multiple optimization instances require minimal communication and coordination and therefore are easy to execute in parallel. Butz and Thiran [2] perform registration by utilizing a genetic algorithm in which fitness (or the metric of survival) is determined by how well the transformed image matches the reference. They implement this approach with an existing genetic solver parallelized using the Message Passing Interface (MPI) [3] on a 10 node cluster. This optimization level parallelism is naturally utilized because evaluating the population of solutions is an inherently independent act. With this class of parallelism, utilizing it is straightforward and can be efficiently implemented, but an individual optimization instance is not accelerated. For this, application designers must tap into opportunities at lower levels of parallelism.

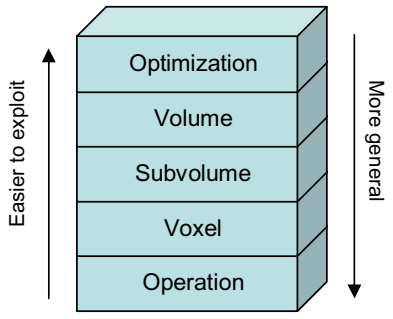

Fig. 1. Levels of image registration parallelism

B. Volume level parallelism

Volume level parallelism is a generalization of optimization level parallelism where the computational units operate on entire volumes. For example, an optimization iteration could be pipelined (applying one trial transform to the floating image while generating another candidate transform). Ino, et al. [1] discuss the potential of "Task parallelism" in accelerating the gradient computation of a rigid registration algorithm, since independent finite difference calculations are done using the entire volume. While volume level parallelism is simple to capture, its use is limited. For many algorithms, the number of independent, entire volume calculations is small. Furthermore, distributing volumes to processing elements can suffer from high communication overhead. Lower levels of parallelism have tended to offer more opportunities for acceleration.

C. Subvolume level parallelism

In medical image registration subvolume level parallelism is perhaps the most popular in which computation is performed on subvolumes of image. Often designers can divide volume level work into smaller subvolumes that are later recombined to produce the final solution. While this creates many opportunities for parallelism, it comes at the price of additional overhead such as coordinating how volumes will be split, managing overlap regions, and consolidating results. Rohlfing and Maurer [4] employ subvolume parallelism for accelerating the similarity calculation. The volume is broken into equally sized sections such that a thread computes its local mutual histogram for MI and then merges its result into the global one. Ourselin et al.[5] use a block matching approach to find the deformation field. Inspired by video compression, the block matching technique compares "blocks" of one image against blocks of the other. These calculations are distributed across processors using MPI. In the same implementation, the authors accelerate image resampling with OpenMP [6]. By distributing computation on individual multiprocessor machines, processes can share image memory and reduce the communication overhead incurred by transmitting images. They simultaneously utilize two programming paradigms to improve performance results.

Ino, et al. [1] use "Data parallelism" by distributing "small

parts" of the image to subtasks which are assigned to different processors. Ino, et al. leverage this same level of parallelism in [7] by distributing the gradient computation of control points across a distributed memory system. Such a distribution not only load balances the computation, but also reduces the memory requirements on an individual node. Stefanescu et al. [8] parallelizes Demons algorithm [9] which is based on optical flow onto a 15 node cluster. The authors split the image into subvolumes to perform matching and filtering. Stenfanescu, et al. [10] perform similar parallelization using a different registration technique. Subvolumes are assigned to different processors and communication is regularized over them. Hardware based approaches can also utilize subvolume level parallelism. Dandekar et al. [11] create an architecture in a field programmable gate array (FPGA) that solves the registration problem recursively on subvolumes. Since each subvolume is an independent local registration problem, datapaths can be replicated for additional performance.

Greater effort has been applied to this level of parallelism to achieve speedups in medical image registration because it is the most general form of parallelism readily exploited by the most commonly used parallel platform: clusters. While clusters are not optimally suited to lower levels of parallelism, researchers have been finding opportunities for parallelism at lower levels using different platforms.

D. *Voxel level parallelism*

Voxel level parallelism describes parallelism in terms of single voxels. In this case, the regional benefit of using subvolumes is not present, so application designers find parallelism in independent voxel computations. R. Strzodka, et al. [12] implement a gradient flow algorithm optimized for a graphics processor (GPU). This algorithm maps well to a GPU as images are put into texture memory and the operations used are supported by the GPU hardware. Warfield et al. [13] utilized a "workpile" of threads to process a voxel independent classification method. They used threads on a shared memory platform to accelerate the task. Voxel level parallelism that cannot be modeled as subvolume parallelism turns out to be rare. But with the rise in popularity of GPUs, efforts that utilize this parallelism are likely to increase. The last level explored by designers is the parallelism present in processing an individual voxel.

E. *Operation level parallelism*

Operational level parallelism is the lowest, most general form of parallelism. At this level, parallelism can be explored in many ways as the basic computational unit is no longer defined. Image registration application designers have found parallel activities to accelerate in the course of processing a single voxel. Castro-Pareja, et al. [14, 15] and Shekhar, et al. [16] construct the "Fast Automatic Image Registration" architecture which parallelizes the computation of transforming, interpolating, and computing the MI of voxels in an image in milliseconds. Designed in a hardware description language (HDL) [17], it can perform MI-based rigid registration in ~1 min. Beyond taking advantage of the 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.

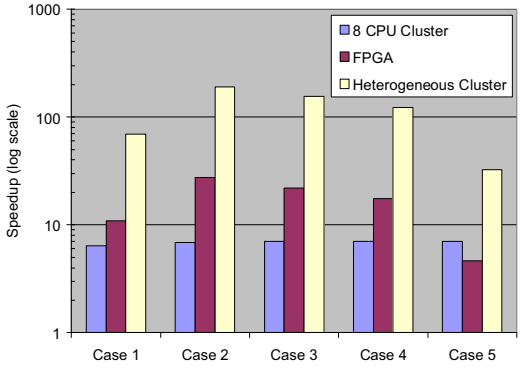

Fig. 2. Speedups from acceleration techniques.

IV. Experiment Results

To show the potential synergy in utilizing this taxonomy to combine techniques from different levels of parallelism, we selected techniques based on a widely used elastic image registration algorithm (a gradient descent method described by Reuckert [18]) and targeted platforms that can readily utilize this parallelism: FPGAs, multiple CPUs, and GPUs. To simulate realizable performance potential, we incorporated each of the acceleration techniques into the same code base. The base algorithm used MI enabling multimodal registration. We used a *subvolume level* technique to distribute the gradient computation equally across nodes in a small cluster similar to [1] and we used *operation level* technique on an FPGA for the transformation and similarity calculation of a voxel [14]. At compile time the software could employ the specific acceleration targeting the corresponding platform. The PC cluster and the uniprocessor results were generated on a 3GHz Intel Xeon. The FPGA accelerator board featured an Altera Stratix EP1S40 FPGA.

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 128x128x128 such that the voxels were roughly isotropic. First a rigid registration was done followed by elastic registration with a deformation field represented by a 9x9x9 control point grid. The gradient step size was based on linear feedback and the algorithm stopped when no improvement could be found at a minimum step size. The results of each registration were visually inspected and in each case improvement was seen.

The speedups based on the time to reach the same level of similarity are shown in the log graph in Fig. 2. While the FPGA and cluster speedups were determined experimentally, we analytically derived speedups for a heterogeneous compute cluster with FPGAs in each node that could take advantage of

both of these kinds of parallelism simultaneously. This multiplicative speedup potential is displayed based on the experimental values. Note that while the software speedup is relatively constant, the FPGA's datapath precision puts it on a different path through the optimization space, leading to varying speedups. Each iterative step is processed faster than software, but certain cases require more steps.

We utilized a *voxel level* acceleration technique on a GPU for the rigid registration transformation step, producing a 3x rigid registration speedup over a uniprocessor implementation. To parallelize the transformation step at the voxel level, we used a GPU, which is an array of processing elements customized for pixel processing. We used Brook [19] 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. Since our GPU implementation only accelerates rigid registration currently, its effect on the total registration time is limited. With a more powerful GPU and elastic registration integration, we believe it will be another valuable target for acceleration techniques in the taxonomy.

V. Conclusion

In this work we have shown the potential power of categorizing and then combining acceleration approaches that exploit different kinds of parallelism in medical image registration, which is an important and computationally intensive application. Our experiments indicated that an eight node, heterogeneous platform could produce up to a 190x speedup over a high performance, general purpose uniprocessor. Such speedups for this size cluster mean that clinicians might have robust registration that normally takes 7 hours to require only a couple minutes to complete. The proposed taxonomy should enable other such synergies. To this end, we plan to construct a prototype heterogeneous cluster as a test platform for further experimentation of other combinations of techniques.

As more complex acceleration techniques are combined, a more rigorous system of capturing the parallelism of the application will be needed. For optimal system performance, programmers must be able to express and to reason about application parallelism in an implementation-independent way. Such a framework would give programmers a more natural way of expressing each type of parallelism without having to dive into low-level, idiosyncratic 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.

VI. 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] 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.

[2] 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.

[3] 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.

[4] 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.

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

[6] L. Dagum and R. Menon, "OpenMP: an industry standard API for shared-memory programming," *Computational Science and Engineering, IEEE*, vol. 5, pp. 46-55, 1998.

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

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

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] C. R. Castro-Pareja and R. Shekhar, "Hardware acceleration of mutual information-based 3D image registration," *Journal of Imaging Science and Technology*, vol. 49, pp. 105-113, 2005.

[16] 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.

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

[18] 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.

[19] 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.