Design and Implementation of Parallel Nonrigid Image Registration Using Off-the-Shelf Supercomputers

Fumihiko Ino[1], Kanrou Ooyama[2], Akira Takeuchi[2], and Kenichi Hagihara[1]

[1] Graduate School of Information Science and Technology, Osaka University
[2] Graduate School of Engineering Science, Osaka University
1-3 Machikaneyama, Toyonaka, Osaka 560-8531, Japan
{ino,hagihara}@ist.osaka-u.ac.jp

Abstract. This paper presents a new parallel algorithm for nonrigid image registration using off-the-shelf supercomputers, or clusters of PCs. Our algorithm realizes scalable registration for high resolution three-dimensional (3-D) images by employing three techniques: (1) data distribution; (2) data-parallel processing; and (3) dynamic load balancing. The experimental results show that our parallel implementation on a cluster of 64 off-the-shelf PCs (with 128 processors) registers liver CT images of $512\times512\times159$ voxels within 8 minutes while a sequential implementation takes 12 hours. Furthermore, our implementation allows processors to use less memory, and thereby enables us to align $1024\times1024\times590$ voxel images, which is not easy for single processor systems due to the restrictions on the memory space and the processing time.

1 Introduction

Nonrigid image registration, which defines a geometric relationship between each point in the nonrigid images, is increasing its role in medical diagnosis. One problem for this technique is a large amount of computation due to many degrees of freedom (DOF). Therefore, to achieve intraoperative nonrigid registration, one major challenging problem is how to reduce its computational cost without dropping the accuracy of alignment.

To realize fast nonrigid registration, high performance computing approaches have been employed in recent works [1,2,3]. In [1], a shared memory multiprocessors reduced the registration time for brain MR images of $256\times256\times100$ voxels from one hour on one processor to approximately a hundred seconds on 64 processors. On the other hand, [2,3] employed clusters [4], or a distributed memory multiprocessors, which has advantages of cost effectiveness and extensibility compared to shared memory multiprocessors. In [2], a cluster of workstations realized intraoperative registration of segmented brain images within 10 seconds. However, its speedup was below 8 on 16 processors due to the imbalance of processor workloads. A cluster of off-the-shelf PCs also demonstrated affine registration in [3], yielding a speedup of 6 on 10 processors.

Thus, a number of parallel registration algorithms have demonstrated the time benefits of high performance computing. However, existing algorithms assume that every processor holds the entire 3-D images. Therefore, the image size available on a parallel system is strictly bounded by the memory space of one processor in the system. This strict restriction is undesirable for high resolution image registration.

R.E. Ellis and T.M. Peters (Eds.): MICCAI 2003, LNCS 2878, pp. 327–334, 2003.

In this paper, to demonstrate the space benefits of high performance computing, we propose a parallel nonrigid registration algorithm for high resolution 3-D images. To achieve this goal, we have parallelized a sequential algorithm [5,6] using a voxel-based similarity measure based on information theory, which realizes robust multimodality registration with no user interaction and preprocessing. Furthermore, we also have employed the following three techniques. First, data distribution enables us to increase the data size available on a parallel system by dividing the entire 3-D image into partially overlapped blocks and distributing them to processors. Second, data-parallel processing realizes high performance registration since it allows processors to independently process the divided blocks in parallel. At last, dynamic load balancing improves parallel efficiency by balancing processor workloads imbalanced due to the difference of computational costs associated with each block.

2 Methods

To register the floating image F and the reference image R, our sequential base of the registration algorithm [5, 6] optimizes a cost function associated with a similarity measure between F and R in a coarse-to-fine manner. The cost function is given by

$$\mathcal{C}(\Phi) = -\mathcal{C}_{\mathrm{similarity}}(R, \mathbf{T}(F)), \tag{1}$$

where $\mathbf{T}$ denotes a nonrigid transformation defined by hierarchical B-splines, $\mathcal{C}_{\mathrm{similarity}}$ denotes a similarity measure based on normalized mutual information, and Φ denotes a mesh of control points $\phi_{i,j,k}$ with uniform spacing δ in the image domain Ω. Any control point $\phi_{i,j,k}$ has a status $\mathcal{S}(\phi_{i,j,k})$ determined by

$$\mathcal{S}(\phi_{i,j,k}) = \begin{cases} \text{active}, & \text{if } \mathcal{M}(\omega_{i,j,k}) > \alpha \\ \text{passive}, & \text{otherwise}, \end{cases} \tag{2}$$

where α is a threshold and $\mathcal{M}$ is a statistical measure determined from $\phi_{i,j,k}$'s $4\delta \times 4\delta \times 4\delta$ neighborhood domain $\omega_{i,j,k}$ (described later in Section 3). During registration process, active control points are allowed to move while passive control points stay fixed. That is, the sequential algorithm optimizes $\mathcal{C}(\Phi^+)$, where Φ^+ represents a set of active control points, in order to simulate adaptive mesh refinement.

To find performance bottlenecks of this algorithm, we analyzed its behavior before parallelizing it. We then found two bottlenecks: one is *the gradient calculation* required for the optimization of $\mathcal{C}(\Phi^+)$ and the other is *the similarity calculation* required for the evaluation of $\mathcal{C}(\Phi^+)$, accounting for 92% and 7% of total execution time, respectively. In the gradient calculation, an active control point $\phi_{i,j,k}$ requires a local gradient $\partial\mathcal{C}/\partial\phi_{i,j,k}$ calculated from small local domain $\omega_{i,j,k}$. However, the gradient calculation requires a large amount of computations since fine resolution meshes include a large number of control points. Therefore, the gradient calculation is the key factor for high performance registration. On the other hand, the similarity calculation mainly consists of the construction of joint histogram $h(R, \mathbf{T}(F))$. Although the similarity calculation seems to be a small bottleneck, it also has to be parallelized to scale the parallel speedup. Otherwise, as Amdahl's law [7] says, the speedup for N processors is bounded by a factor

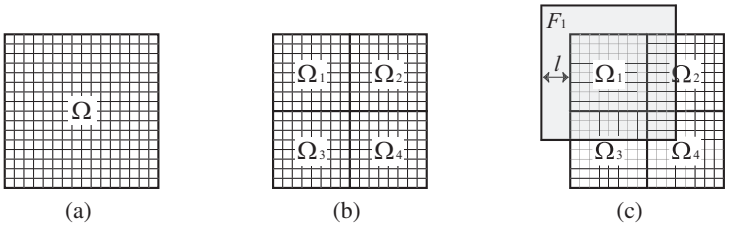

Fig. 1. Image distribution. (a) No distribution, (b) Non-overlapped block distribution, and (c) Partially overlapped block distribution with margin length l. For all $1 \leq s \leq N$, processor P_s holds subimage F_s and takes responsibility for calculation within subdomain Φ_s.

of $\lim_{N\to\infty}(100/(7+92/N)) < 15$. That is, even on a large number of processors, the speedup never reaches a factor of 15, if we avoid parallelizing the similarity calculation. Therefore, to realize scalable registration, we as well as [1] have decided to parallelize both the gradient calculation and the similarity calculation.

2.1 Data Distribution

To investigate which data fragments should be distributed to processors, we next analyzed the memory usage of the sequential algorithm and then found that 3-D images take the most, compared to other data such as histogram $h(R, \mathbf{T}(F))$ and parameters Φ. Hence, we have decided to divide the floating image F and the reference image R into blocks and distribute them only at fine resolution levels. At coarse resolution levels where the image size is small enough, we avoid data distribution since it can cause some additional communications between processors, which drops program performance. In the following, we assume that for all $1 \leq s \leq N$, processor P_s holds subimages F_s and R_s, or a portion of F and that of R, respectively.

We now describe how our algorithm divides the images into blocks. To determine the block size, we have to consider the tradeoff between data distribution and program performance. While the maximum block size corresponds to no data distribution (Fig. 1(a)), which brings the best performance but requires a larger amount of memory, the minimum block size corresponds to non-overlapped block distribution (Fig. 1(b)), which brings the minimum usage of memory but causes additional communication overheads. Our approach for this issue is to divide the images into partially overlapped blocks with an appropriate margin to avoid communications during data-parallel processing of the similarity calculation and the gradient calculation (Fig. 1(c)). Here, assume that for all $1 \leq s \leq N$, processor P_s takes responsibility for calculation within Ω_s, where Ω_s denotes a non-overlapped subdomain of Ω. Then, our approach satisfies the following two conditions C1 and C2, for all $1 \leq s \leq N$.

- C1: For any active control point $\phi_{i,j,k}$ in subdomain Ω_s, F_s contains neighborhood domain $\omega_{i,j,k}$ and R_s contains transformed neighborhood domain $\mathbf{T}(\omega_{i,j,k})$.
- C2: For any point (x, y, z) in subdomain Ω_s, F_s contains (x, y, z) and R_s contains transformed point $\mathbf{T}(x, y, z)$.

Both conditions C1 and C2 allow every processor to perform data-parallel processing with no communication. That is, while condition C1 allows P_s to locally calculate every gradient $\partial \mathcal{C}/\partial \phi_{i,j,k}$ within Ω_s, condition C2 also allows P_s to create a local joint histogram for Ω_s, for all $1 \leq s \leq N$.

Given the maximum length of valid deformations in v mm, the margin length l for satisfying conditions C1 and C2 is determined by

$$l = d + 2 \cdot \delta \ \text{ such that } \ d \geq v, \tag{3}$$

where d is the maximum permissible length for deformations in mm. (3) satisfies condition C1 since $l \geq v + 2 \cdot \delta$. It also satisfies condition C2 since $l > v$. Although determining the accurate value for l seems difficult due to the difficulty of detecting v before registration, we think that rough estimation based on users' experiences can be applied to practical use.

2.2 Dynamic Load Balancing for Gradient Calculation

Since adaptive mesh refinement causes active control points in a non-uniform distribution, the workloads associated with each subdomain Ω_s become imbalanced in the compute-intensive gradient calculation. To address this performance issue, we analyzed the control point distribution on 128 processors with some clinical images. In most cases, approximately a quarter of 128 processors are responsible for more than 80% of active control points. Therefore, the key issue for high performance registration is how to scatter the 80% of active control points to the remaining three quarters.

To realize this, our algorithm employs two load-balancing strategies. One is for coarse resolution levels where data distribution is unnecessary and the other is for fine resolution levels where data distribution is necessary. For no data distribution scheme, since every processor holds the entire images and knows which control points are active, we dynamically assign active control points to processors in a round-robin manner. For data distribution scheme, we apply a list scheduling algorithm [8] to place processors into groups, in which the active control points are scattered for load balancing.

Let Φ_s^+ be a set of active control points in subdomain Ω_s, where $1 \leq s \leq N$. Then, our algorithm places processors into groups by two steps:

1. Group Initialization:
   Set the number of groups, M, as that of processors with more active control points than the average: $\overline{W} \equiv \sum_{s=1}^{N} |\Phi_s^+|/N$. Placing each of these high-loaded M processors into groups gives group $G_t = \{P_{t(1)}\}$, where $1 \leq t \leq M$ and $P_{t(1)}$ denotes one of the M processors.
2. Group Construction:
   Create a list, L, in which the remaining $N - M$ processors are sorted by $|\Phi_s^+|$ in an ascending order. Then, from head to tail of L, any processor $P_s \in L$ are added to G_t such that $\overline{W}(G_t)$ is the maximum, where $\overline{W}(G_t)$ represents the average number of active control points in G_t.

As a result, for all $1 \leq t \leq M$, G_t consists of one high-loaded processor $P_{t(1)}$ and some low-loaded processors $P_{t(2)}, \ldots, P_{t(|G_t|)}$, because a small portion of processors have a large portion of active control points, as we mentioned before.

Given group G_t, where $1 \leq t \leq M$, to balance workloads, subimages with many active control points, $F_{t(1)}$ and $R_{t(1)}$, and information on active control points $\Phi^+_{t(1)}$ are broadcasted to every processor $P_s \in G_t$. This broadcast enables low-loaded processors to calculate $\partial \mathcal{C}/\partial \phi_{i,j,k}$, for any $\phi_{i,j,k} \in \Phi^+_{t(1)}$. Thus, in addition to steps 1. and 2., our algorithm parallelizes the gradient calculation by following steps:

3. Image Transmission:
   For all $1 \leq t \leq M$, the high-loaded processor $P_{t(1)}$ broadcasts $F_{t(1)}$ and $R_{t(1)}$, and information on active control points $\Phi^+_{t(1)}$ to every processor $P_s \in G_t$.
4. Workload Distribution:
   For all $1 \leq t \leq M$, $\Phi^+_{t(1)}$ is splitted into $\lambda_{t(1)}, \lambda_{t(2)}, \ldots, \lambda_{t(|G_t|)}$ such that $|\lambda_{t(1)}| = |\lambda_{t(u)} \cup \Phi^+_{t(u)}| = \overline{W}(G_t)$, where $2 \leq u \leq |G_t|$.
5. Gradient Calculation:
   For all $1 \leq t \leq M$, processors $P_{t(1)}$ and $P_{t(u)}$, where $2 \leq u \leq |G_t|$, independently calculate gradients for λ_1 and $\lambda_u \cup \Phi^+_{t(u)}$, respectively.
6. Result Distribution:
   Processor P_1 gathers the calculation results and broadcasts them to all processors $P_1, P_2, \ldots, P_N$.

2.3 Data-Parallel Processing for Similarity Calculation

The similarity calculation is parallelized by data-parallel processing and the binary-swap (BS) method [9], or a scalable method for parallel image compositing. The original purpose of BS is to merge N local images into the final image, and this calculation is similar to the similarity calculation, where processors independently construct N local histograms by data-parallel processing and merge them into the global histogram $h(R, \mathbf{T}(F))$. One advantage of BS is high scalability which comes from parallel processing and tree structured merging. On N processors, BS completes the data merging in $\log N$ stages, and every processor participates in all stages of the merging process.

Our parallel similarity calculation consists of three steps. In the first step, for all $1 \leq s \leq N$, P_s independently creates a local joint histogram h_s for the subdomain Ω_s. In the second step, all created local histograms $h_1, h_2, \ldots, h_N$ are merged into the global histogram $h(R, \mathbf{T}(F))$ by BS method. To do this, every processor is paired up. Every pair of processors splits its local histogram into two pieces, and each processor takes responsibility for one of the two pieces. Repeating this splitting and exchanging with different pairs for $\log N$ stages generates the global histogram $h(R, \mathbf{T}(F))$ in a distributed manner. At last, $\mathcal{C}_{\text{similarity}}(R, \mathbf{T}(F))$ is calculated from $h(R, \mathbf{T}(F))$.

3 Results

To evaluate our parallel algorithm, we have implemented our algorithm on a cluster of PCs by using the C++ language and MPICH-SCore library [10], which is a fast implementation of the Message Passing Interface (MPI) standard [11]. Our cluster consists of 64 symmetric multiprocessor (SMP) nodes. Each node has two Pentium III 1GHz processors and connects to a Myrinet switch [12] which provides bandwidth of 2Gb/s.

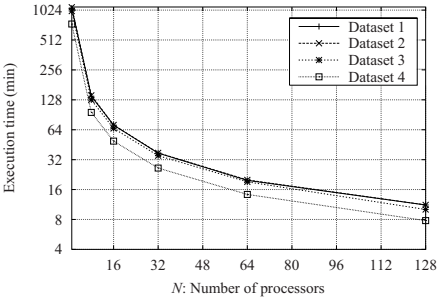

Fig. 2. Execution time.

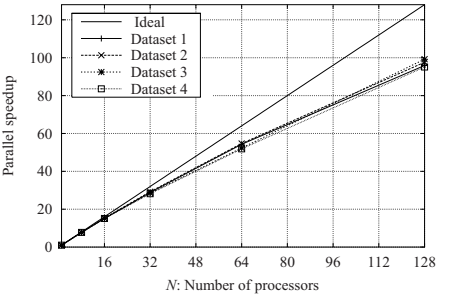

Fig. 3. Parallel speedup.

We applied our algorithm to four datasets of contrast-enhanced liver CT images, which have a size of $512 \times 512 \times 159$ voxels with spatial resolution of $0.67 \times 0.67 \times 1.25$ mm. We refined Φ by three levels with spacing δ from 42.88 to 10.72 via 21.44 mm. The original image was resampled to $\gamma = 2.68$, 1.34, and 0.67 mm at each of deformation levels. As we mentioned in Section 2.1, this image resampling reduces the data size, so that the images at levels Φ^1 and Φ^2 were small enough to avoid data distribution. Therefore, to yield better performance, we distributed the images only at level Φ^3.

From some experiments, we decided to employ two measures for determining control point statuses $\mathcal{S}(\phi_{i,j,k})$: $H(R, \omega_{i,j,k})$, or the local image entropy for the subdomain $\omega_{i,j,k}$ as a reference image measure; and $||\partial \mathcal{C}/\partial \phi_{i,j,k}||$, or the norm of the local gradient of the cost function as a joint image pair measure. Since $H(R, \omega_{i,j,k})$ locally characterizes the reference image R and $||\partial \mathcal{C}/\partial \phi_{i,j,k}||$ describes the degree of image alignment, we initialized $\mathcal{S}$ by using $\alpha = 0.65 \cdot H(R)$ at the beginning of each deformation level and updated it by using $\alpha = 0.005 \cdot ||\partial \mathcal{C}/\partial \Phi^+||$ after the gradient calculation.

3.1 Registration Performance

Figures 2 and 3 show the total execution time and parallel speedup of our method, respectively. On 128 processors, it reduces the registration time from 741 to approximately 8 minutes with high speedups ranging from a factor of 95 to 99. At this time, our method accelerates the gradient calculation and the similarity calculation by a factor of 103 and 92, respectively. Both these high speedups contribute to the total speedup of 96. If we avoid parallelizing the similarity calculation, the execution time on 128 processors can be estimated as 84 minutes, resulting in a low speedup of 13. Thus, our implementation realizes scalable registration. Next, the execution time for other processing, such as image initialization and its resampling, reduces from 7.4 to 0.8 minutes with a speedup of 9. This low speedup mainly comes from disk access required for image resampling which takes about 25 seconds. Therefore, on a large number of processors, file input/output can appear as a performance bottleneck.

To make clear the effect of our dynamic load balancing, we now compare the following three methods: (1) (proposed) PDL, (2) PD, and (3) PL, where P represents parallel, D represents data distribution, and L represents load balancing. Fig. 4 shows speedups for the optimization at Φ^3. On 128 processors, our PDL improves the speedup of PD

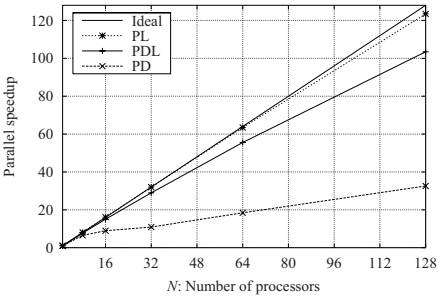

Fig. 4. Comparison of methods.

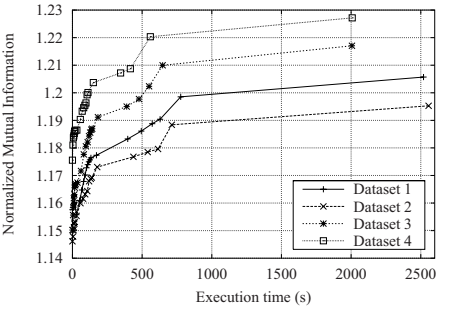

Fig. 5. Optimization progress (PDL).

from a factor of 33 to 103. However, its speedup is less than a factor of 123 obtained by PL. Thus, our method enables data distribution with less performance decreases.

In PD, Φ_7^+ has the maximum of 185 active control points, so that processor P_7 takes that of 436.3 seconds for the gradient calculation. On the other hand, the execution time in our PDL ranges from 56.4 to 94.7 seconds. Here, our dynamic load-balancing mechanism allows the bottleneck processor P_7 to reduce its execution time by distributing its workload to six low-loaded processors: $P_1, P_{29}, P_{35}, P_{49}, P_{53}$ and P_{57}. Besides, the standard deviations of the execution time in PDL, PD, and PL are 9.28, 107.4 and 0.37, respectively, so that our PDL achieves better load balancing compared to PD.

Although PDL requires data broadcasting from high-loaded to low-loaded processors, it takes only 1.1 seconds on our cluster. For all datasets, the amount of broadcasted data is 28.8MB composed mainly of subimages $F_{t(1)}$ and $R_{t(1)}$ with many active control points, where $1 \leq t \leq M$. Although $|G_t|$ ranges from 2 to 7, the communication imbalance associated with this broadcasting has little effect on the overall performance, because the communication takes shorter time than the gradient calculation. However, it can be a performance bottleneck on low-speed networks such as 100Mb/s Ethernet.

3.2 High Resolution Image Registration

Data distribution enables us to register images in a fine resolution level Φ^4, where $\delta = 5.36$ and $\gamma = 0.335$ mm. Without data distribution, the optimization at Φ^4 requires more than 2GB memory, so that out-of-core computation occurs on our cluster. Since this computation takes significant execution time, PL is no realistic method for high resolution image registration. On the other hand, PD can perform on-memory computation, however, it takes 105 minutes on 128 processors while PDL takes 47 minutes. Thus, in our PDL method, data distribution increases the image resolution available on a system and dynamic load balancing reduces the execution time for image registration.

Fig. 5 shows the optimization progress on 128 processors. We obtain more accurate results as the deformation level increases. At the finest level Φ^4, NMI raises its value by 0.015, where the total improvement is 0.067. Since Φ^4 takes much longer time than the former levels, deciding when the registration should be terminated is important for clinical use, where a series of images can be produced in rapid succession.

4 Conclusion

We have presented a parallel algorithm for nonrigid image registration on distributed memory multiprocessors. To realize robust and scalable nonrigid registration for high resolution 3-D images, our algorithm employs hierarchical FFDs described in [5,6] with three techniques: (1) data distribution, (2) data-parallel processing, and (3) dynamic load balancing. The experimental results show that our method performs nonrigid registration of liver CT images of $512 \times 512 \times 159$ voxels in less than 8 minutes on 128 processors, which takes approximately 12 hours on a single processor. Moreover, our method reduces the amount of memory usage to less than 25%, enabling us to increase the image resolution available on our system.

One future work includes to develop a methodology for detecting the margin length during registration.

Acknowledgments. This work was partly supported by JSPS Grant-in-Aid for Scientific Research (C)(2) (14580374) and JSPS Research for the Future Program JSPS-RFTF99I00903. The authors would like to thank Dr. J. Masumoto, Dr. Y. Sato and Dr. S. Tamura of the Graduate School of Medicine, Osaka University for useful discussions regarding this work.

References

1. Rohlfing, T., Maurer, C.R.: Nonrigid image registration in shared-memory multiprocessor environments with application to brains, breasts, and bees. IEEE Trans. Inform. Technol. Biomed. **7** (2003) 16–25
2. Warfield, S.K., Ferrant, M., et al.: Real-time biomechanical simulation of volumetric brain deformation for image guided neurosurgery. In: Proc. SC2000. (2000) 1–16
3. Ourselin, S., Stefanescu, R., Pennec, X.: Robust registration of multi-modal images: Towards real-time clinical applications. In: Proc. MICCAI'02. (2002) 140–147
4. Buyya, R., ed.: High Performance Cluster Computing. Prentice Hall PTR (1999)
5. Rueckert, D., Sonoda, L.I., et al.: Nonrigid registration using free-form deformations: Application to breast MR images. IEEE Trans. Med. Imag. **18** (1999) 712–721
6. Schnabel, J.A., Rueckert, D., et al.: A generic framework for non-rigid registration based on non-uniform multi-level free-form deformations. In: Proc. MICCAI'01. (2001) 573–581
7. Amdahl, G.: Validity of the single processor approach to achieving large-scale computing capabilities. In: Proc. AFIPS Conf. Volume 30. (1967) 483–485
8. Graham, R.L.: Bounds on multiprocessing timing anomalies. SIAM J. Appl. Math. **17** (1969) 416–429
9. Ma, K.L., Painter, J.S., Hansen, C.D., Krogh, M.F.: Parallel volume rendering using binary-swap compositing. IEEE Comput. Graph. Appl. **14** (1994) 59–68
10. O'Carroll, F., Tezuka, H., Hori, A., Ishikawa, Y.: The design and implementation of zero copy MPI using commodity hardware with a high performance network. In: Proc. 12th ACM Int'l Conf. on Supercomputing (ICS'98). (1998) 243–250
11. Message Passing Interface Forum: MPI: A message-passing interface standard. Int'l J. of Supercomputer Applications and High Performance Computing **8** (1994) 159–416
12. Boden, N.J., Cohen, D., et al.: Myrinet: A gigabit-per-second local-area network. IEEE Micro **15** (1995) 29–36