

Cluster of Workstation based Nonrigid Image Registration Using Free-Form Deformation

Xiaofen Zheng, Jayaram K. Udupa, and Xinjian Chen

Medical Image Processing Group, Department of Radiology
423 Guardian Drive, Blockley Hall, 4th Floor
University of Pennsylvania, Philadelphia, U.S.A.

ABSTRACT

Nonrigid image registration plays an important role in medical application fields. Owing to its complex computations, it incurs high computational cost. In this paper, a parallel algorithm schema for nonrigid image registration methods that use B-splines for deformation and mutual information as a similarity measure is proposed. It involves a complex interplay of various steps which are analyzed in considerable detail from the view point of parallelizing registration. The algorithms are implemented on a cluster of workstations. We present some results on a 10 processor cluster of PCs and compare them with a sequential implementation. The results show that a speed up of $n/2$ for n processors in registering large images. The method is fully portable and seamlessly expandable.

Keywords: Nonrigid registration, cluster of workstations, free-form deformation, cubic B-spline, mutual information, image pyramid, multiresolution, optimization, parallel algorithm

1. INTRODUCTION

Many medical applications require the comparison of images taken over time, longitudinal or immediate time. Image registration techniques allow such comparisons and further analysis. To address the deformable nature of organs and tissues, a variety of nonrigid registration methods have been developed. However, owing to their computational complexity, they usually require a large computation time, which limits their clinical usability on large data sets. To address this problem, especially on large data sets, we present a methodology for parallelizing such algorithms.

A parallel nonrigid registration algorithm on a shared memory multiprocessor system (MPS) has been previously reported.¹ Using multithreaded programming, it partitions data and tasks depending on the computational subproblems to take advantage of the shared-memory multiprocessor computer. The reported registration time for 3-D images of size $256 \times 256 \times 100$ is 100 seconds on 64-CPU's. The shared memory allows converting the existing sequential algorithms to parallel execution, but the cost for a large scale MPS is usually high. To increase computation power at a low cost, a cluster of workstations (COW) offers certain distinct advantages over MPS for intensive image processing operations,² such as image registration. Besides the computing power and economical viability, the memory usage per cluster node is also reduced. Therefore, from the considerations of cost, portability, extensibility, and generality, we chose the COW for parallelizing registration.

There are some published nonrigid registration approaches based on COW. A method for multi-subject non-rigid registration that uses a priori information about the nature of imaged objects in order to adapt the regularization of the deformations has been demonstrated.³ The reported registration time for 3-D images of size $256 \times 256 \times 124$ is 5 minutes on a cluster of 15 standard PCs. A distributed memory parallelization method for a nonrigid image registration algorithm using a block decomposition of the displacement field has also been implemented.⁴ In this method the registration of MRI images of size $256 \times 256 \times 120$ takes about 3.5 minutes by using 15 processors. A data distributed parallel algorithm that can register large-scale 3-D images of deformable

Further author information: (Send correspondence to Xiaofen Zheng or Jayaram K. Udupa)
Xiaofen Zheng: E-mail: xiaofenz@mail.med.upenn.edu, Telephone: 1 215 662 6780
Jayaram K. Udupa: E-mail: jay@mail.med.upenn.edu, Telephone: 1 215 662 6780

objects has been proposed.⁵ This method registering liver CT images of size $512 \times 512 \times 295$ takes about 10 minutes using 128 processors.

In this paper, we propose a parallel algorithm schema for nonrigid image registration using the popular B-spline formulation based on a COW. We describe in detail the strategies employed to parallelize the multiresolution registration using mutual information as the similarity measure. To closely analyze the benefits of parallelization, the experimental results from a sequential algorithm and a parallel scheme on a cluster of 10 nodes are compared at each level step by step.

2. NONRIGID REGISTRATION METHOD AND PARALLELIZATION

Our nonrigid registration algorithm uses cubic B-splines to describe the free-form deformation and mutual information for a similarity measure, although other similarity measures can also be used. Cubic B-splines are used in image representation because they are especially suited for operations such as differentiation, integration, searching for extrema, etc.⁶ Since both the deformation model and the similarity measurement are differentiable, an analytic method⁷ of computing the gradient of mutual information can be applied. The nonrigid deformation parameters are optimized by maximizing mutual information with a stochastic gradient descent technique.⁸

In order to achieve model flexibility, reduce time cost, and avoid local minima, a hierarchical multiresolution optimization method is implemented. To register two images, this algorithm schema consists of the following main steps:

1. Calculating the image pyramids for the two images;
2. In each image level:
 - (a) Representing image in the continuous space via the expansion B-spline coefficients;
 - (b) Optimizing deformation fields by maximizing mutual information;
3. Computing an output deformed image using optimized parameters.

These steps are sequentially executed, but parallelization within each step can reduce computation time dramatically. Further, the images, the B-spline representation of the images, and the necessary auxiliary data structures call for extensive storage space especially when image data are large. In the parallel schema, the data are divided into chunks² which drastically reduces the memory requirements for the individual cluster nodes. Below we describe how these chunks are determined and how parallelization is effected in each step.

2.1 Cubic B-spline Pyramid

Taking advantage of the B-spline's separable property, a 3-D image pyramid is obtained by successive 1-D filtering and decimation along rows, columns, and slices of the image. In our implementation, in one dimension, the image signal is represented by the 3rd order spline which is a continuously-defined function.⁹ This spline model⁹ is unambiguously defined by the sample values and specifies the reduction algorithm which is optimal in the least squares sense. Then 1-D image signal is downsampled by a reduction filter.¹⁰ The standard pyramid¹⁰ where the coarser grid points are at the even integers is implemented.

Given a 3-D image, we parallelize this step by first breaking the image into equally sized partitions in the Y direction for filtering and reduction along the Z direction. This partitioning in Y direction allows each cluster node to have a continuous range of slices along the Z direction. Each cluster node is assigned one of these partitions and it filters and decimates the image signal along Z direction. Then the results are equally partitioned in the Z direction for filtering and reduction along X first and then Y on each cluster node without calling for communication among the processors.

2.2 B-spline Coefficients

Following the notations of Thevenaz and Unser,⁷ let $f_T(\mathbf{x})$ be a test image that we want to align with a reference image $f_R(\mathbf{x})$.

In each level of the image pyramid, we employ Thevenaz and Unser's image model⁷ via B-spline functions of degree 3

$$f(\mathbf{x}) = \sum_{\mathbf{x}_i \in \mathbf{V}} c(\mathbf{x}_i) \beta^{(3)}(\mathbf{x} - \mathbf{x}_i)$$

where $f(\mathbf{x})$ is the continuous image, cubic B-spline $\beta^{(3)}$ is the model kernel, \mathbf{x}_i stand for points of a continuous 3-D domain in which images are defined, \mathbf{V} is a discrete set of points in this domain, and $c(\mathbf{x}_i)$ are the expansion B-spline coefficients.

We use Unser et al.'s method¹¹ for the determination of the B-spline coefficients $c(\mathbf{x}_i)$. They are computed by recursive digital filterings successively along each image dimension. For the discrete signal $f(k)$ in each dimension, the recursive filter equations¹¹ are

$$\begin{cases} c^+(k) = f(k) + b_1 c^+(k-1), & (k = 2, \dots, K) \\ c^-(k) = f(k) + b_1 c^-(k+1), & (k = K-1, \dots, 1) \\ c(k) = b_0 (c^+(k) + c^-(k) - f(k)) \end{cases}$$

where $b_0 = -6\alpha/(1-\alpha^2)$ and $b_1 = \alpha = -0.2679$. Filter $c^+(k)$ is causal running from left to right, and filter $c^-(k)$ is anti-causal running right to left. The boundary conditions that the signal is extended by its mirror image are imposed.¹¹ The initial values¹¹ are

$$\begin{cases} c^+(1) = \sum_{k=1}^{k_0} \alpha^{|k-1|} f(k) \\ c^-(K) = c^+(K) \end{cases}$$

where k_0 is such that α^{k_0} is below some prescribed level of precision.

For a 3-D image, the coefficient computing is separately performed along each dimension. Along each dimension, the coefficients computing are independently performed on each line. Therefore this step is parallelized by first breaking the image into equally sized partitions in the Y direction for filtering along the Z direction. This partitioning in the Y direction allows each node to have a continuous range of slices along the Z direction. Each cluster node is assigned one of these partitions and each computes the coefficients along the Z direction. After finishing this part of the computation, master machine collects the result image, then equally partitions the results in the Z direction for filtering along X and Y without calling for communication between processors as each node has a continuous range along both X and Y directions. The filtered chunks of the test image are kept in each cluster node for the next step.

2.3 Stochastic Gradient Descent Optimization

The goal of optimization is to find the optimal deformation which maximizes the similarity between the test and the reference image. We choose mutual information as the similarity measure. Using gradient descent optimization, the transformation parameters $\boldsymbol{\mu}$ are optimized by calculating the gradient of mutual information $\nabla \mathbf{S}$. Based on the comparison results⁸ of some popular techniques, a randomly selected subset of 2048 voxels is used in every iteration of the optimization process. A bias in the approximation error is avoided in this stochastic subsampling while time cost is greatly reduced for optimization. The stochastic gradient descent optimization is defined as

$$\boldsymbol{\mu}_{k+1} = \boldsymbol{\mu}_k - a_k \widetilde{\nabla \mathbf{S}}$$

where a_k is the gain factor and $\widetilde{\nabla \mathbf{S}}$ is the approximation of $\nabla \mathbf{S}$. The gradient $\nabla \mathbf{S}$ is defined as

$$\nabla \mathbf{S} = \left[\frac{\partial S}{\partial \mu_1}, \frac{\partial S}{\partial \mu_2}, \dots \right]$$

With Thevenaz and Unser’s image model, both the deformation and mutual information are differentiable. This facilitates calculating the gradient of the mutual information, from which the deformation field can be optimized. Thevenaz and Unser developed an analytic method⁷ of computing the gradient of mutual information from the joint probability distribution of the test and the reference image, the gradient of the joint probability distribution, and the marginal probability distribution of the test image:

$$\frac{\partial S}{\partial \mu} = - \sum_{l \in L_T} \sum_{k \in L_R} \frac{\partial p(l, k; \boldsymbol{\mu})}{\partial \mu} \log_2 \left(\frac{p(l, k; \boldsymbol{\mu})}{p_T(l; \boldsymbol{\mu})} \right)$$

where L_T and L_R are the discrete sets of intensities associated with the test and the reference image, l and k are the discrete bins, p_T is the marginal discrete probability of the test image, p is the joint probability distribution of the test and the reference image. The partial derivative of joint probability distribution is given by

$$\frac{\partial p(l, k; \boldsymbol{\mu})}{\partial \mu} = \frac{1}{\Delta b_T v} \sum_{\mathbf{x}_i \in \mathbf{V}} \beta^0 \left(k - \frac{f_R(\mathbf{x}_i) - f_R^\circ}{\Delta b_R} \right) \frac{\partial \beta^{(3)}(\xi)}{\partial \xi} \Big|_{\xi = l - \frac{f_T(\mathbf{g}(\mathbf{x}_i; \boldsymbol{\mu})) - f_T^\circ}{\Delta b_T}} \left(\frac{-df_T(\mathbf{t})}{d\mathbf{t}} \Big|_{\mathbf{t}=\mathbf{g}(\mathbf{x}_i; \boldsymbol{\mu})} \right)^T \frac{\partial \mathbf{g}(\mathbf{x}_i; \boldsymbol{\mu})}{\partial \mu}$$

where v is the number of voxels in \mathbf{V} , \mathbf{g} is the deformation, f_T° and f_R° are the minimum intensity value, Δb_T and Δb_R are the intensity range of each bin. The smoothed joint probability¹² is given by

$$p(l, k; \boldsymbol{\mu}) = \alpha \sum_{\mathbf{x}_i \in \mathbf{V}} \beta^0 \left(k - \frac{f_R(\mathbf{x}_i) - f_R^\circ}{\Delta b_R} \right) \beta^{(3)} \left(l - \frac{f_T(\mathbf{g}(\mathbf{x}_i; \boldsymbol{\mu})) - f_T^\circ}{\Delta b_T} \right)$$

where α is a normalization factor that ensures $\sum p(l, k) = 1$. Therefore the marginal smoothed probability of the test image can be computed from the joint probability

$$p_T(l; \boldsymbol{\mu}) = \sum_{k \in L_R} p(l, k; \boldsymbol{\mu})$$

Our free-form deformation model is based on cubic B-splines. B-spline deformation are determined by control points. A higher resolution of the control point mesh defines a larger number of degrees of freedom, meanwhile the computation is more complex and time consuming. In the multiresolution application, B-spline deformation can be refined by dividing the control point spacing in every dimension by two. Starting from the coarsest level, control point parameters are optimized in each level and refined for the next finer level. The method in¹³ is used for the refinement of control points between two levels.

In our parallel algorithm, the reference image is equally partitioned along the Z direction and each partition is sent to the appropriate machine. For the test image, as shown in Figure 1, each partition consists of two parts: main part and an overlapping part. The main part is the equally partitioned part. The size of the extra overlapping part is determined by calculating the maximum deformation of the voxels in the partition’s border. We assume that the largest possible movement of the control points is the spacing between control points. This parameter is adjustable depending on need. For deformation above this limit, the cluster nodes can always get the necessary pieces from the main parts which are evenly distributed among workstations. Then each machine in the cluster has a portion of the continuous image model which can be used to compute the partial of image joint probability and its gradient needed in optimization. The master machine generates 2048 random numbers corresponding to the 2048 voxels in the entire image, so each machine needs to calculate only on the subset of voxels which fall within its main part. From the equation of joint probability and its gradient, they are the summation over the sample set; therefore, in the parallel algorithm, each node can compute a part of the joint probability on the chunk of sample. The master machine collects all the results, computes the joint probability, and then sends it to each computer. Only those 64 control points nearest \mathbf{x} contribute to the sums, so each node computes the part of $\nabla \mathbf{S}$, then the master machine collects these results and updates the control points. After one iteration of optimization, the optimized control parameters are distributed to the individual machines for the next iteration.

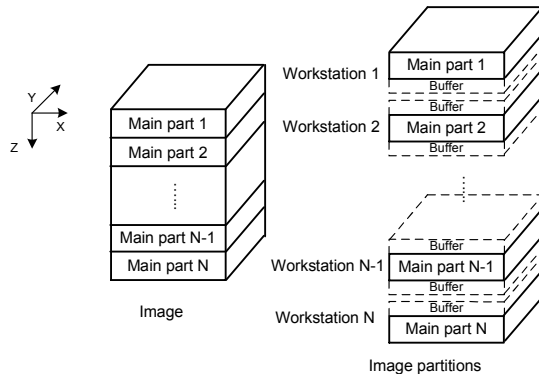


Figure 1. Image partitioning of main part and extra part for parallel algorithms with N workstations in the cluster.

2.4 Cubic B-spline Deformation and Interpolation

Using the optimized deformation parameters, the deformation $\mathbf{g}(\mathbf{x}; \boldsymbol{\mu})$ is computed. A cubic B-spline convolution kernel is used in interpolation and deformation. The deformation¹² at any point $\mathbf{x}(x, y, z)$ in the image is

$$\mathbf{g}(\mathbf{x}; \boldsymbol{\mu}) = \sum_j \mu_j \beta^{(3)} \left(\frac{\mathbf{x} - \lambda_j}{\Delta\rho} \right)$$

where $\Delta\rho$ stands for the regular spacing of control point grid, λ_j is the control point grid coordinates and $\beta^{(3)}(\mathbf{x}) = \beta^{(3)}(x)\beta^{(3)}(y)\beta^{(3)}(z)$ is a separable cubic B-spline convolution kernel. Only its nearest $4 \times 4 \times 4$ control points contribute to the deformation of a point. Then the gradient interpolation is computed over Thevenaz and Unser's image model that is discussed in section 2.2.

To compute the output image, parallelization is achieved easily by computing on the previously distributed test image chunk on each machine. Each machine computes only the voxels in the main part of the image chunks, and the extra part of the image chunk is used for the deformation and interpolation.

3. EXPERIMENTS AND RESULTS

We tested the sequential and the parallel algorithm on a cluster with 10 workstations connected through 1 GB/s switch. Each workstation has a Pentium D 3.4 GHz CPU and 4 GB of main memory. The reference image is the original image and the test image in each experiment was created by deforming the reference image by a known deformation field.

In the experiments, a large CT image data set of size $512 \times 512 \times 459$ with voxel size $0.68 \times 0.68 \times 1.5 \text{ mm}^3$ was used. B-spline control point spacing is set to 13.6 mm at the finest level. This yields a $27 \times 27 \times 52$ mesh of control points to parameterize the deformation field, i.e., 113,724 parameters to optimize on the finest level. Mutual information is computed on 32 histogram bins for the test and the reference image. The registration process is implemented over a 7 level pyramid from coarse to fine, with level 1 considered to have the finest resolution. In level 7, the computation time is less than 1 second in every step, the time cost discussion is not included. For comparison using 6 levels, each of the 10 workstations can get at least a slice of image in the finest level.

A run time analysis in each step for the multiple levels is provided in the following sections. The listed time is based on 6 trials of experiments.

3.1 Cubic B-spline Pyramid

Fist step is to build the image pyramid starting from the finest level. The time cost for computing image pyramid sequentially and parallelly on 10 workstations is shown in Table 1. Since sequential computing at level 4 and level 5 takes less than 1 second, the time cost comparison at these levels is not listed. At each level the operations

are on the test and the reference image. The total time costs of sequentially and parallelly computing B-spline pyramid for test image and reference image over the entire 6 levels are 221 seconds and 31 seconds. The speed up by using 10 clusters in this step is 7.1.

Table 1. Time cost of sequential and parallel algorithms for computing image pyramid in a multiresolution setup. Time units are seconds.

Image level and size	Sequential	Parallel (10)
Level 3 ($128 \times 128 \times 114$)	2.8	0.4
Level 2 ($256 \times 256 \times 229$)	21.2	4.0
Level 1 ($512 \times 512 \times 459$)	176.4	26.6

3.2 B-spline Coefficients

At each level, B-spline coefficients are computed. The time cost of sequential and parallel algorithms in this step is shown in Table 2. Since the sequential computing from level 4 to level 6 takes less than 1 second, the time cost at these levels is not listed in the figure. The overall time of sequential and parallel computing in this step is 296 seconds and 60 seconds. The speed up by using 10 clusters in this step is 5.0.

Table 2. Time cost of sequential and parallel algorithms for computing B-spline coefficients in a multiresolution setup. Time units are seconds.

Image level and size	Sequential	Parallel (10)
Level 3 ($128 \times 128 \times 114$)	3.4	0.7
Level 2 ($256 \times 256 \times 229$)	28.8	4.1
Level 1 ($512 \times 512 \times 459$)	263.7	54.9

3.3 Optimization

The optimization time cost largely depends on the number of iterations and the resolution of the control points. Every iteration must be done sequentially because the previously optimized deformation parameters are used for deformation in the next iteration. The communication time portion would therefore increase in the parallel method if the number of iterations is large and the control point resolution is high.

The optimization times are tested on 100 iterations in each level. Table 3 shows the time cost of sequential and parallel optimization at each level. From level 2 to level 6, the parallel optimization is accelerated. At level 1, sequential optimization uses less time. From our experiments, for large data, sequential optimization might be faster, while for a regular image size such as the size similar to the image at level 2 and smaller, parallel optimization takes less time. This is due to the large data communication in each iteration if the image data are large. The sequential optimization at 6 levels takes 284 seconds, and the parallel optimization takes 352 seconds. If we mix sequential and parallel methods, i.e., computing parallelly from level 6 to level 2 and sequentially on level 1, the time can be reduced to 241 seconds.

3.4 Cubic B-spline Deformation and Interpolation

The last step in our registration scheme is to use B-spline deformation and interpolation for computing the output image. The computation is at the finest level over the test image using the optimized control point parameters. The speed up obtained for this step is 7.3 for 10 workstations.

Table 3. Time cost of sequential and parallel algorithms for optimizing in a multiresolution setup. The number of iterations is 100 in each level. Time units are seconds.

Image level and size	Control mesh	Sequential	Parallel (10)
Level 6 ($16 \times 16 \times 14$)	$2 \times 2 \times 3$	6.2	0.8
Level 5 ($32 \times 32 \times 28$)	$3 \times 3 \times 4$	10.2	1.3
Level 4 ($64 \times 64 \times 57$)	$5 \times 5 \times 8$	14.7	2.6
Level 3 ($128 \times 128 \times 114$)	$8 \times 8 \times 14$	18.7	6.6
Level 2 ($256 \times 256 \times 229$)	$14 \times 14 \times 27$	37.0	33.0
Level 1 ($512 \times 512 \times 459$)	$27 \times 27 \times 52$	196.7	307.8

Table 4. Time cost of sequential and parallel algorithms for computing output image with B-spline deformation and interpolation. Time units are seconds.

Image level and size	Sequential	Parallel (10)
Level 1 ($512 \times 512 \times 459$)	2255.7	310.8

3.5 Cumulative Result

In total, one sequential registration takes 3057 seconds, and one parallel registration takes 754 seconds on 10 workstations. If a threshold is used in each step to decide on using the parallel or the sequential optimization, the best time cost can be reduced to 642 seconds.

Figure 2 illustrates the scaled time comparison in each step at each level. The darker color bars stand for the time of sequential processing, and the light color bars show the time for parallel processing. From left to right, progression is from coarse level to fine level. The parallel computing time is far less than the sequential in all processes except in optimization in the finest level. If the computer has enough memory to deal with the sequential optimization, using sequential optimization in this step is preferred. But parallel methods can be useful when the image data are so large as not to fit into the RAM of the single machine in the sequential case.

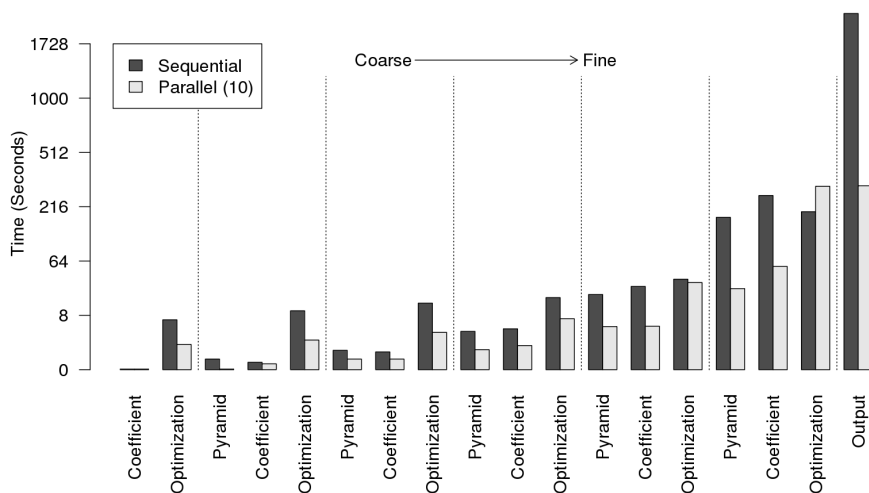


Figure 2. Sequential and parallel time comparison for each step at each level. The time axis is scaled nonlinearly to show the differences at all levels.

Figure 3 shows the parallel cumulative time cost relative to the sequential cumulative time cost in the whole registration process. The first step is computing image pyramids, which starts from the finest and proceeds to the coarsest, level by level. Then from coarsest level to the finest level, image is interpolated and the parameters are optimized. The last step is to compute the output image. In each process at every level, the accumulative sequential time is denoted as 1 and the accumulated parallel time is illustrated in the figure. We can also choose sequential optimization at the finest level and parallel for all the rest of the processing. The time cost of this combined method is shown in the figure as well.

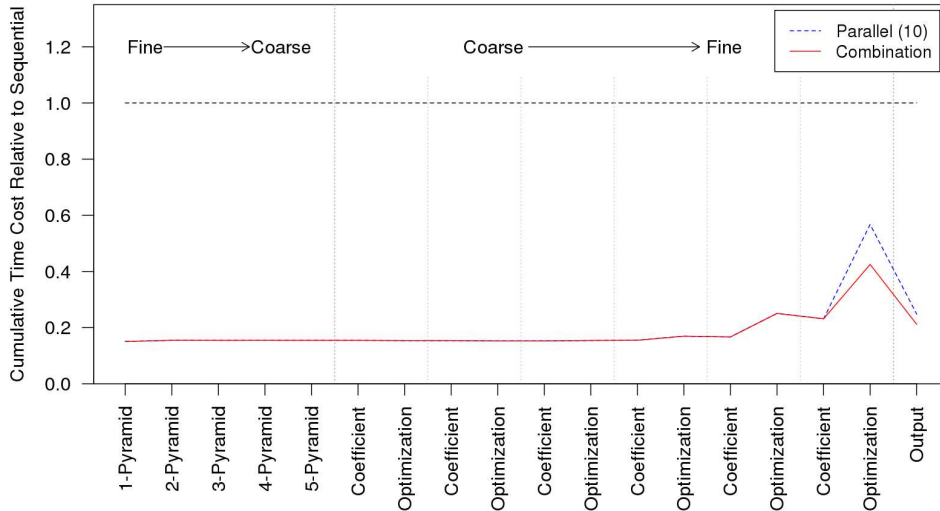


Figure 3. Cumulative parallel time cost relative to cumulative sequential time cost. Cumulative sequential time is denoted as 1.

We tested the method on second smaller data set, a brain MRI image of size $256 \times 256 \times 46$ and voxel size $0.98 \times 0.98 \times 3 \text{ mm}^3$. The B-spline control spacing was set to 9.8 mm on the finest level which yields a $27 \times 27 \times 15$ mesh of control points, i.e., 10,935 parameters to optimize on the finest level. With 3 levels, using 100 iterations of optimization on each level, one registration takes about 115 seconds by parallelizing on 10 computers. The sequential version takes 195 seconds.

4. CONCLUSION

This paper has carefully examined the orchestration of a general parallel strategy for nonrigid image registration by utilizing the various efficient techniques that have been reported in the literature for the component steps. The complex interplay of the various steps involved in non-rigid image registration is analyzed from the point view of parallelizing registration.

Our experiments over 10 cluster machines indicated that a speed up of $0.5 \times N$, where N is the number of computers in the COW, is feasible. Because of the partitioning operations, very large data sets that may not fit in the RAM of a single machine can be handled by the method. This general parallel schema is based on portable distributed computing on a cluster of PCs, and its implementation in the CAVASS software² via distributed computing on a COW is portable.

ACKNOWLEDGMENTS

This work is supported by NIH grant EB004395. The authors wish to thank Dewey Odhner and George Grevera in Medical Image Processing Group and Scott Sherrill-Mix at University of Pennsylvania for their comments and assistance.

REFERENCES

- [1] Rohlfing, T. and Maurer, C. R., “Nonrigid image registration in shared-memory multiprocessor environments with application to brains, breasts, and bees,” *IEEE Transactions on Information Technology in Biomedicine* **7**(1), 16–25 (2003).
- [2] Udupa, J. K., Grevera, G., Odhner, D., Zhuge, Y., Souza, A., Mishra, S., and Iwanaga, T., “CAVASS: a computer-assisted visualization and analysis software system - image processing aspects,” in [*Progress in biomedical optics and imaging*], *Proc. SPIE* **6509**, 65091V (2007).
- [3] Stefanescu, R., Pennec, X., and Ayache, N., “Grid powered nonlinear image registration with locally adaptive regularization,” *Medical Image Analysis* **8**(3), 325–342 (2004).
- [4] Stefanescu, R., Pennec, X., and Ayache, N., “Parallel non-rigid registration on a cluster of workstations,” in [*HealthGrid*], *Proc.* (2003).
- [5] Ino, F., Ooyama, K., and Hagihara, K., “A data distributed parallel algorithm for nonrigid image registration,” *Parallel Computing* **31**, 19–41 (2005).
- [6] de Boor, C., [*A Practical Guide to Splines*], Applied Mathematical Sciences, Springer, New York (1978).
- [7] Thvenaz, P. and Unser, M., “Optimization of mutual information for multiresolution image registration,” *IEEE Transactions on Image Processing* **9**(12), 2083–2099 (2000).
- [8] Klein, S., Staring, M., and Pluim, J. P., “Evaluation of optimization methods for nonrigid medical image registration using mutual information and b-splines,” *IEEE Transactions on Image Processing* **16**, 2879–2890 (2007).
- [9] Unser, M., “Splines: A perfect fit for signal and image processing,” *IEEE Signal Processing Magazine* **16**(6), 22–38 (1999).
- [10] Brigger, P., Mller, F., Illgner, K., and Unser, M., “Centered pyramids,” *IEEE Transactions on Image Processing* **8**(9), 1254–1264 (1999).
- [11] Unser, M., Aldroubi, A., and Eden, M., “Fast b-spline transforms for continuous image representation and interpolation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence* **13**(3), 277–285 (1991).
- [12] Mattes, D., Haynor, D. R., Vesselle, H., Lewellen, T. K., and Eubank, W., “PET-CT image registration in the chest using free-form deformations,” *IEEE Transactions on Medical Imaging* **22**(1), 120–128 (2003).
- [13] Rohlfing, T., Maurer, C. R., ODell, W. G., and Zhong, J., “Modeling liver motion and deformation during the respiratory cycle using intensity-based free-form registration of gated MR images,” in [*Medical Imaging*], *Proc. SPIE* **4319**, 337–348 (2001).