Proceedings of the 28th IEEE
EMBS Annual International Conference
New York City, USA, Aug 30-Sept 3, 2006

Hardware Implementation of Hierarchical Volume Subdivision-based Elastic Registration

Omkar Dandekar, *Student Member, IEEE,* Vivek Walimbe, *Student Member, IEEE,* and Raj Shekhar, *Member, IEEE*

Abstract— **Real-time, elastic and fully automated 3D image registration is critical to the efficiency and effectiveness of many image-guided diagnostic and treatment procedures relying on multimodality image fusion or serial image comparison. True, real-time performance will make many 3D image registration-based techniques clinically viable. Hierarchical volume subdivision-based image registration techniques are inherently faster than most elastic registration techniques, e.g. free-form deformation (FFD)-based techniques, and are more amenable for achieving real-time performance through hardware acceleration. Our group has previously reported an FPGA-based architecture for accelerating FFD-based image registration. In this article we show how our existing architecture can be adapted to support hierarchical volume subdivision-based image registration. A proof-of-concept implementation of the architecture achieved speedups of 100 for elastic registration against an optimized software implementation on a 3.2 GHz Pentium III Xeon workstation. Due to inherent parallel nature of the hierarchical volume subdivision-based image registration techniques further speedup can be achieved by using several computing modules in parallel.**

I. INTRODUCTION

Medical image registration is the process of achieving an optimal alignment between images in order to track changes in anatomy or to combine complementary structural and functional information. Elastic image registration can achieve this alignment using a nonlinear, continuous transformation and hence, is better suited for recovering realistic tissue deformations. Mutual information (MI)-based elastic registration has been shown to be effective in multi-modality image registration due to the robustness of the similarity measure [1].

Our group has reported an MI-based elastic registration algorithm that utilizes volume subdivision [2,3]. This algorithm has been used and validated in the context of whole body PET-CT registration [4] and ultra low-dose CT-guided interventions [5]. MI-based 3D elastic image registration is an automatic but computationally intensive task with typical execution time on the order of several hours on modern desktops. This aspect limits the use of elastic registration in clinical applications requiring real-time performance, such as image guided interventions. Volume subdivision-based image registration algorithms are inherently faster and are better suited for parallelization and can yield real-time performance. Our group has also reported an FPGA-based architecture optimized for accelerating FFD-based image registration [6]. In the current work, we will show how our architecture can be adapted for accelerating volume subdivision-based elastic registration.

O. Dandekar is with the Department of Electrical and Computer Engineering, University of Maryland, College Park, MD 20742 USA (dandekar@ieee.org)

V. Walimbe is with the Biomedical Engineering Department, The Ohio State University, Columbus, OH 43210 USA (walimbe.2@osu.edu)

R. Shekhar is with the Department of Diagnostic Radiology, University of Maryland, Baltimore, MD 21201 USA (phone: 410-706-8714, fax: 410-706-8724, rshekhar@umm.edu)

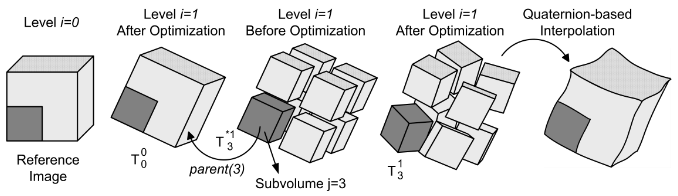
Level i=0
Level i=1 After Optimization
Level i=1 Before Optimization
Level i=1 After Optimization
Quaternion-based Interpolation
Reference Image
T_0^0
T_3^{*1}
parent(3)
Subvolume j=3
T_3^1

Fig. 1. Hierarchical volume subdivision-based image registration

II. REGISTRATION ALGORITHM

3D image registration using volume subdivision has been proposed earlier, but the implementations were limited to local translation-based model. Our group has enhanced this model by incorporating local rotations. We have also reported a quaternion-based scheme for interpolating multiple 3D rigid-body transformations for elastic registration using the volume subdivision approach [2,3]. For a pair of images, one treated as reference and the other floating, this algorithm performs elastic registration using a series of hierarchical, locally rigid registrations. The six-parameter rigid registration at each level is optimized by maximizing the normalized mutual information (NMI) between the reference and floating images. This hierarchical registration scheme is shown in Fig. 1.

The initial optimal rigid alignment (at the root level) between these images can be represented using a transformation matrix T_0^0 (where T_j^i represents the cumulative optimal transformation at level i for subvolume j). Next, the algorithm uses a hierarchical octree-based subdivision scheme. At each subdivision level i, the reference image is divided into 8^i subvolumes, numbered from 0 to 8^i–1. Each of these subvolumes is then individually registered with the floating image, under transformation range constraints derived from the transformation of its parent subvolume at the earlier level $T_{parent(j)}^{i-1}$. The notation $parent(j)$ refers to the subvolume number at the previous subdivision level i−1, which contains the current subvolume

j (for example : at the root level ($i=0$), there is a single subvolume (entire image) numbered $j=0$. After one level of subdivision ($i=1$), there will be eight subvolumes numbered from $j=0$ to $j=7$. At level $i=1$, $parent(3)$ refers to subvolume number 0 at level $i=0$, as it contains subvolume $j=3$ at the current level ($i=1$) of subdivision, See Fig. 1). Again, the optimal alignment of the subvolume j within the floating image, is determined by maximizing NMI under a six-parameter rigid-body transformation model.

Volume subdivision and subvolume registration continue until the voxel count for an individual subvolume remains above a predefined limit (usually 16^3) to yeild a statistically significant similarity measure. Thus, this algorithm achieves hierarchical refinement of the localized matching between the reference and the floating images. The overall elastic alignment between the image pairs is computed by quaternion-based direct interpolation of the individual subvolume transformations at the final subdivision level.

A. Calculating NMI for a subvolume

Registration of a subvolume during the hierarchical refinement process is based on maximization of the NMI, which is a statistical measure. With progressive subdivision, the subvolumes at every level get increasingly smaller. The mutual histogram (MH) corresponding to an individual subvolume becomes sparse and thus rendering NMI unreliable. The said algorithm addresses this issue by using the MH of the entire image (all the subvolumes) to calculate NMI during the registration of a subvolume. The contribution of the current subvolume k at level i to the MH is computed under the current candidate transformation T_k^{*i}. The contribution to the MH from the rest of the subvolumes remains constant during this registration process and is derived from their parent subvolumes. Thus, NMI is computed over the entire image with local variations corresponding to the subvolume under optimization. Equations (1a-1c) summarize this process. The function " $\mathrm{Accumulate}(T)$ ", contributes to the mutual histogram using the voxels in a given subvolume using the mapping provided by the given transformation T. T^* denotes a candidate transformation during the optimization process. Further details on this operation are provided in [2,3].

$$MH^i_{Total_k} = MH^i_{Subvolume_k} + MH^i_{Rest_k} \tag{1a}$$

$$MH^i_{Subvolume_k} = \underset{j=k}{\mathrm{Accumulate}}\left(T_j^{*i}\right) \tag{1b}$$

$$MH^i_{Rest_k} = \underset{\forall j, j\neq k}{\mathrm{Accumulate}}\left(T^{i-1}_{parent(j)}\right) \tag{1c}$$

III. HARDWARE IMPLEMENTATION

Constructing the mutual histogram for a given transformation is a critical step in all MI/NMI-based image registration algorithms. It has been shown [7] that, for large images, accumulating the mutual histogram can take 99.9% of the total mutual information calculation time in software. Our efforts for acceleration of image registration, therefore, are targeted towards optimized and pipelined implementation of mutual information calculation.

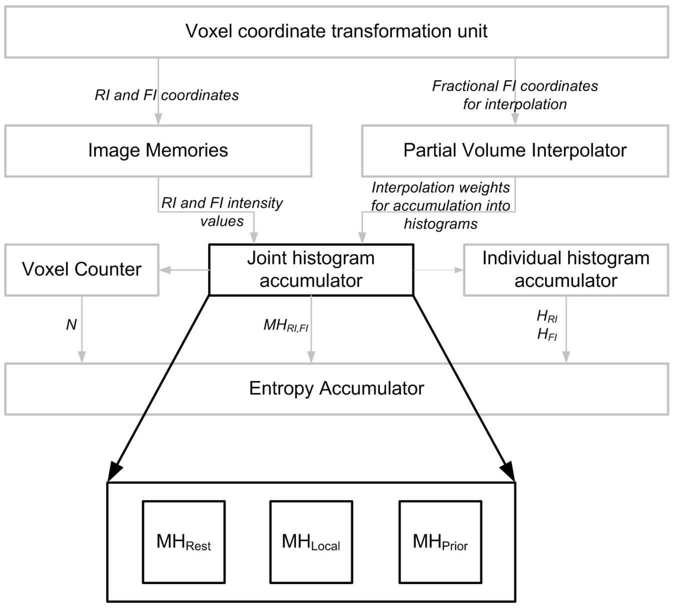
Voxel coordinate transformation unit
RI and FI coordinates
Fractional FI coordinates for interpolation
Image Memories
Partial Volume Interpolator
RI and FI intensity values
Interpolation weights for accumulation into histograms
Voxel Counter
Joint histogram accumulator
Individual histogram accumulator
N
$MH_{RI,FI}$
H_{RI}
H_{FI}
Entropy Accumulator
MH_{Rest}
MH_{Local}
MH_{Prior}

Fig. 2. FAIR-II architecture reported in [6]. The proposed design updates "Joint Histogram Accumulator" with computation of MH_{Rest}, MH_{Local}, and MH_{Prior} to support volume subdivision-based image registration.

We have earlier reported an FPGA-based architecture (FAIR-II) to accelerate the calculation of mutual information [6,7]. Although, this architecture can achieve a speedup of 100 for elastic registration, it is not suitable for volume subdivision-based image registration in its current form. The partitioning scheme employed in FAIR-II will result into prohibitively high mutual histogram memory requirement especially since MH_{Rest} needs to be calculated for every subvolume. The proposed design is primarily based on the FAIR-II architecture and has been extended for the optimized calculation for MH_{Rest}. Fig. 2 shows the important changes in the architecture to support volume subdivision-based image registration natively.

In the following subsection we provide an overview of the important hardware modules in the proposed design. Since these functional blocks are based on the FAIR-II architecture, detailed description of their design and operation can be found in [6,7]. Calculating MH_{Rest} is integral and unique to volume subdivision-based image registration. The primary focus of this work is to extend the architecture to compute MH_{Rest} in hardware. This implementation strategy is outlined in subsection B. Moreover, this architecture is ideally suited for parallelized implementation using multiple computing units. This is briefly discussed in subsection C.

A. Architecture

Initial step in mutual information calculation is to apply a candidate transformation T_j^{*i} to a voxel $\vec{v}_r$ in current subvolume j (in reference image) and map it to floating image space ($\vec{v}_f$) as shown in (2). Since the algorithm is

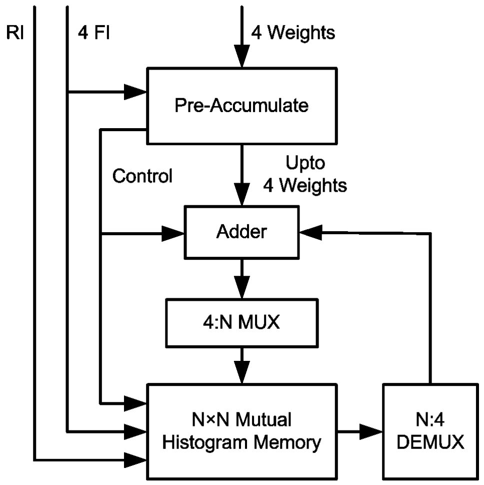

Fig. 3. Updating the mutual histogram

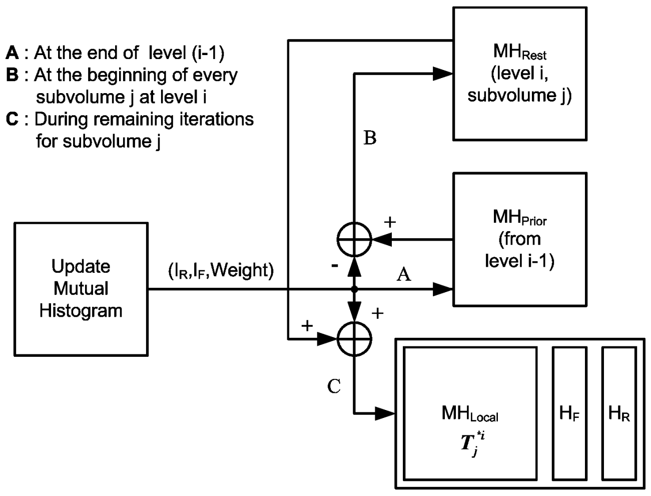
A : At the end of level (i-1)
B : At the beginning of every subvolume j at level i
C : During remaining iterations for subvolume j
MHRest (level i, subvolume j)
B
MHPrior (from level i-1)
Update Mutual Histogram
(IR,IF,Weight)
A
C
MHLocal
Hf
HR

Fig. 4. A scheme to calculate MH_{Rest}

linear at every subvolume, this is implemented using the six parameter rigid transformation model.

$$\vec{v}_f = T_j^{*i} \cdot \vec{v}_r \tag{2}$$

The mapped coordinates ($\vec{v}_f$), will have both fractional and integer components and, in general, will land within a floating image neighborhood of size $2 \times 2 \times 2$. Using the partial volume (PV) interpolation introduced by Maes et al. [1], the interpolation weights are calculated using the fractional components of $\vec{v}_f$. The corresponding floating voxel intensities are fetched in parallel using the integer component of $\vec{v}_f$. Simultaneously, the reference image controller will fetch the voxel corresponding to $\vec{v}_r$. The mutual histogram will then be updated for each pair of reference and floating voxel intensities (8 in all), using the corresponding weights computed by the PV interpolator.

1) Updating the mutual histogram: For a given reference image voxel RI, there are eight intensity pairs $(RI, FI_0 \colon FI_7)$ and corresponding interpolation weights. Since the MH needs to be updated (read-modify-write) at corresponding eight locations, this amounts to 16 accesses to mutual histogram memory for each reference image voxel. This problem is alleviated using the high speed, dual ported SRAMs provided within the FPGA. Because this memory can operate at twice the frequency of the main pipeline and can be read and written simultaneously, the mutual histogram access requirement is reduced to four accesses per clock cycle. Fig. 3 illustrates the modification to the FAIR-II architecture [6] to implement this. The pre-accumulate block accumulates the weights for floating image voxels with the same intensity. Since the mutual histogram is organized as $N \times N$ (typically $N = 128$) block of memory, it is possible to use byte-enable feature or a simple multiplexer to update these weights in one clock cycle.

While the mutual histogram is being computed, the individual histogram accumulator unit computes the histograms for the floating and reference images. This is done using a structure similar to that of mutual histogram but with additional pre-accumulate operations. The voxel counter module keeps track of the number of valid voxels accumulated in the mutual histogram and calculates its reciprocal value. The resulting value is then used for calculating the individual and joint probabilities.

2) Entropy Accumulation: The second step of mutual information calculation is to compute the entropy using the joint and individual probabilities. To calculate the mutual information it is necessary to apply the function $f(p) = p \cdot \ln(p)$ to these probabilities. This function is implemented in hardware using the multiple lookup table-based piecewise polynomial approximation. This approach preserves the shape of the MI curve and the location of the extrema and thus does not affect the optimization process. This method is explained in detail in [6].

B. Calculating MH_{Rest}

During the optimization process of finding the best alignment for a given subvolume k at level i, MH_{Total_k} is computed as shown in (1a-1c). It is therefore necessary to compute the contribution of remaining subvolumes to the MH_{Total_k} using the registration information at the previous level of subdivision ($T_{parent(j)}^{i-1}, \forall j \neq k$). This process needs to be repeated for every subvolume at the current level i, and the contents of the MH_{Rest_k} will be different every time depending on the subvolume under consideration. Computing MH_{Rest_k}, from scratch, for every iteration of each subvolume will not be efficient. In order to address this issue, we introduce mutual histogram buffers MH_{Prior}, MH_{Rest_k}, and MH_{Local} which store the previous level MH and partial MH during various stages of the algorithm. A flow diagram depicting the activity of these MH buffers during various steps of the algorithm is shown in Fig. 4 and the detailed description is provided below.

At a given level i, MH_{Prior} contains the mutual histogram for the entire image, which is computed using all the subvolumes at the earlier level $i-1$ and corresponding transformations T_j^{i-1}. At the beginning of the registration of each subvolume k at the current level i, the transformation of its parent from the previous level, $T^{i-1}_{parent(k)}$, is applied to the current subvolume and the resultant mutual histogram is subtracted from MH_{Prior}. This step is mathematically equivalent to computing MH_{Rest_k} (as in (1c)) for the subvolume k and the resultant mutual histogram is stored in the buffer MH_{Rest_k}. During the optimization of this subvolume, a candidate transformation T^{*i}_k is applied only to the current subvolume k, and the resultant histogram (contribution of this subvolume) is added to MH_{Rest_k}, to form the mutual histogram for the entire image (MH_{Total_k} in (1a)) for the current optimization step. This final histogram is stored in the buffer MH_{Local} and further used for computing the image similarity measure (NMI) corresponding to the current transformation T^{*i}_k. This process is repeated for all the subvolumes at the current level. At the end of each level (after optimizing all the subvolumes at that level), the mutual histogram for the entire image is computed using the updated transformations $T^i_k\,(\forall k)$ and is stored in MH_{Prior}, which will subsequently be used at the next level of subdivision $i+1$.

C. Parallelization

For every subsequent level of subdivision after the initial global rigid registration, the algorithm optimizes the individual subvolumes at the current level independent of each other. Using multiple computing modules in parallel it is possible to optimize these subvolumes simultaneously. Equation (3) shows the speedup that can be achieved using N computing modules, for L levels of subdivisions.

$$Speedup = \frac{L}{1 + \dfrac{(L-1)}{N}} \tag{3}$$

Further speedup can be achieved by distributing non-overlapping parts of reference image across all computing modules for the purpose of MH accumulation, during the global rigid registration. But this strategy will have an additional overhead of accumulating partial mutual histograms.

IV. IMPLEMENTATION AND RESULTS

The proposed design was implemented, as a proof of concept, using an Altera Stratix EP1S40 FPGA in a PCI prototyping board manufactured by SBS, Inc. The design achieved a maximum internal frequency of 200 MHz, with a 50 MHz reference image processing rate. Reference and floating images were stored using two separate standard PC100 SDRAM modules. Entropy calculation was implemented using the 4-LUT, first-order polynomial configuration. Mutual information was calculated using 32-bit fixed point numbers. Since the design of the entropy calculator is a variant of the corresponding desing in the FAIR-II architecture, further details could be found in [6,7].

With wider reference image RAM bus or DDR SDRAMS, the projected processing speed of the system is 100 million reference image voxels per second. This processing speed translates to image registration time of approximately 1 minute, for image size of $256 \times 256 \times 256$, with 5 levels of subdivision and 50 optimization iterations at each level. Table 1 compares the execution time of the aforementioned algorithm using the proposed hardware against an optimized software implementation. The number of optimization steps in both the cases were identical with the lowest subvolume size of 16^3. Our architecture can achieve a speedup of over 100 for elastic registration against the corresponding software implementation on a 3.2-GHz PIII Xeon workstation with 1.5GB of 266 MHz RAM. With parallel implementation as described earlier, further speedup can be achieved. This will enable 3D image registration in a few seconds.

TABLE I
COMPARISON OF ELASTIC REGISTRATION TIME (SECONDS)

Image Size	Software Implementation	Hardware Implementation	Speedup Achieved
64^3	57	0.5	114
128^3	429	3.5	122
256^3	5231	35.55	142

V. CONCLUSIONS

Acceleration of elastic image registration is critical for its efficient and effective use in clinical procedures, especially in image-guided interventions. Our architecture features a fast implementation of volume subdivision-based elastic registration. Using this architecture the execution time of elastic image registration can be reduced from more than an hour to a few seconds (for image size of $256 \times 256 \times 256$, with 5 levels of subdivision and 50 optimization iterations per level). Compact and scalable nature of this architecture makes it suitable for clinical deployment.

REFERENCES

[1] F. Maes, A. Collignon, D. Vandermeulen, G. Marchal, and P. Suetens, "Multimodality image registration by maximization of mutual information," *IEEE Trans. Medical Imaging*, vol. 22, pp. 986–1104, 2003.

[2] V. Walimbe, V. Zagrodsky, S. Raja, B. Bybel, M. Kanvinde, and R. Shekhar, "Elastic registration of three-dimensional whole body CT and PET images by quaternion based interpolation of multiple piecewise linear rigid-body registrations," in *proc SPIE*, 2004, pp. 1191–1228.

[3] V. Walimbe and R. Shekhar, "Automatic elastic image registration by interpolation of 3D rotations and translations from discrete rigid-body transformations," in *Medical Image Analysis*, Acccepted for publication.

[4] R. Shekhar, V. Walimbe, S. Raja, V. Zagrodsky, M. Kanvinde, G. Wu, and B. Bybel, "Automated 3-dimensional elastic registration of whole-body PET and CT from separate or combined scanners," *Journal of Nuclear Medicine*, vol. 46, pp. 1488–1496, 2005.

[5] O. Dandekar, K. Siddiqui, V. Walimbe, and R. Shekhar, "Image registration accuracy with low-dose CT: How low can we go?," in *proc IEEE International Symposium on Biomedical Imaging*, 2006.

[6] 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.

[7] C.R. Castro Pareja, J.M. Jagadeesh, and R. Shekhar, "FAIR : A hardware architecture for real-time 3D image registration," *IEEE Trans. Information Technology in Biomedicine*, vol. 7, pp. 426–434, 2005.