Multithreaded Approach for Registration of Medical Images using Mutual Information in Multicore Environment and its Applications in Medical Imaging

Sanjay Saxena
School of Biomedical Engineering
Indian Institute of Technology

Neeraj Sharma, Ph.D.
School of Biomedical Engineering
Indian Institute of Technology

Shiru Sharma, Ph.D.
School of Biomedical Engineering
Indian Institute of Technology

ABSTRACT
Image Registration plays very crucial role in case of medical imaging to register different modalities of images like CT (Computed Tomography) and PET (Positron Emission Tomography) registration. CT is essential for structural information of anatomic and PET (Positron Emission Tomography) is for functional information. Basically it is the procedure of transforming dissimilar sets of data into one coordinate system. These sets of data can be acquired from multiple image modalities, different viewpoints, similar or dissimilar sensors. MI based image registration has been found to be reasonably useful methods of image registration. However, it is found to be quite computationally intensive and time consuming process for enormous size images and for different data sets of images. It involves steps for computation of joint histogram, marginal entropies, calculation and probability distribution. Main motive of this paper is to provide an intelligent method for image registration based on Mutual Information using multi core environment with maintaining the synchronization between different activated cores and processors. Proposed Method has been able to execute with different number of threads to achieve all the remuneration of the processors and gives significant speedup working with verity of images like gray scale, RGB and Dicom images with different size. Finally the designed algorithm has been used to register medical images of different modalities.

Keywords
Image registration, Parallel computing, Mutual Information, Medical Images, Multithreading, CT, PET.

1. INTRODUCTION
Image registration (IR) is extremely essential task of Image Processing as it is the procedure of aligning two images so that the point in one image corresponds to the same anatomical position in the other. It is a key part in the medical imaging analysis. Medical images are often taken at different time and places, resulting in varying frame of references for the same part of the human body in the images [1]. Image registration plays vital role in the registration of different modalities of medical images like CT – PET registration and many more. MI based image registration is very valuable as it is based on the statistical relationship between two images and most important thing is that segmentation is not necessary prior to MI based image registration. Apart from the importance of MI based image registration it is quite computationally expensive [2]. Parallel computing seems to be the only solution for fast and efficient computing as by using this we can use non local resources very efficiently with suitable multithreading. Importantly it removes the limits of serial computing [3]. Now day’s multicore processors include several processing elements within an integrated circuit and this can be considered as parallel solution which removes the limitation of sequential computing [4-7].

For implementing MI based Image Registration we need to consider following basic concepts

1.1 Entropy
It was introduced by Shannon in 1948 [8], and is defined as

$$H(X) = \sum_{x \in X} p(x) \log \frac{1}{p(x)}$$  (1)

Here p represents the function of probability mass of the random variable X. Shannon entropy measures the degree of uncertainty of a random variable by scoring less likely outcomes higher than the more likely ones. To define a Joint Probability distribution for both images to have to generate a 2D histogram where each axis represents the number of possible gray scale values in each image and each histogram cell is incremented each time occurs in the pair of images. If the images are perfectly aligned then the histogram is highly focused if it is not than the dispersion grows.

1.2 Mutual Information
Mutual Information is the amount that the uncertainty in N(or M) is minimized when M (or N) is known and is defined as [9-16]

$$I(M;N) = H(N) - H(N|M) = H(M) - H(M|N)$$  (2)

MI based image registration is very much important as it consists individual entropy and works superior than simply joint entropy in regions of image background (low contrast) where there will be stumply joint entropy but this is offset by low individual entropies as well so the overall mutual information will be low.

To make best use of the mutual information we have to minimize the joint entropy so the equation is

$$I(M;N) = H(M) + H(N) - H(M,N)$$  (3)

One more way of representing MI is by the following equation

$$I(M,N) = \sum_{a,b} p(m,n) \cdot \log \left( \frac{p(m,n)}{p(m)p(n)} \right)$$  (4)

This definition is correlated to the Kullback-Leibler distance between two distributions. It measures the dependence of the two distributions. In image registration I (M,N) will be maximized when the images are aligned.
It is having some properties that are given below
a. MI is symmetric: \( I(M;N) = I(N;M) \)
b. \( I(M;M) = H(M) \)
c. \( I(M;N) \leq H(M), I(M;N) \leq H(N) : \) info every image contains concerning the other cannot be larger than the info they themselves include.
d. \( I(M;N) \geq 0 : \) Cannot amplify uncertainty in M by knowing N
e. If M,N are independent then \( I(M;N) = 0 \)
f. If M,N are Gaussian then:
\[
I(M, N) = -\frac{1}{2} \log(1 - \rho^2)
\] (5)
Following figures describes the basic steps of Image Registration and MI based Image Registration describe by [11].

![Figure 1: Basic Steps of MI Based Image Registration][11]

Above figure describes about the basic steps of MI Based Image Registration we have parallelized this method as per suitability in our research to gain best result.

There are several techniques developed using parallel computing [17-21] for fastest Image Registration has been developed. Some of them are “A Parallel GPU Algorithm for Mutual Information based 3D Non rigid Image Registration” [22], “A Parallel Mutual Information Based Image Registration Algorithm for Applications in Remote Sensing” [23], “GPU Accelerated Medical Image Registration Techniques” [24], “Acceleration of Genetic Algorithm with Parallel Processing with Application in Medical Image Registration” [25], “Parallel Image Registration Using Bio-Inspired Computing” [26], “Multimodal Image Registration Using GPU Parallel Computing Technology” [27], “Fast Parallel Image Registration On CPU And GPU For Diagnostic Classification Of Alzheimer’s Disease” [28], “Histogram-Based Image Registration Using Parallel Computing” [29]. There are some limitations of these algorithms like some of them approaches need GPU and good knowledge of CUDA programming models. Some of them need to be implemented on Medical Dicom images some approaches need to be tested on variety of images and so on. One of our papers based on review of existing image registration techniques using parallel computing has been accepted for publications and is in press[30]. There is the comparison of different techniques, limitations and benefits. This work is basically implemented to overcome the limitations of existing technique.

For this we organize this research paper into seven sections. Next section II will provide an overview of the concepts of parallel computing, parallel programming models and environments available for parallel computing. Section III describes about the proposed method for MI based image registration using parallel computing. Evaluation of experiment with different image data sets and results obtained is discussed section IV. Section V contains the discussion of this method and results obtained. Applications in medical imaging is described in section VI followed by conclusion in Section VII.

Now we are going to discuss briefly about the basic concepts of parallel computing and the parallel programming models which we have implemented in our work.

2. PARALLEL COMPUTING, ENVIRONMENT AND PROGRAMMING MODELS AVAILABLE FOR PARALLEL COMPUTING

It is a variety of computation in which various calculations are carried out concomitantly as it is having the principle that a large problem or a multifarious problem can be separated into smaller one. There can be numerous kind of parallel machine like a cluster of computers that contains multiple PCs combined together with a elevated speed network, a shared memory multiprocessor by connecting multiple processors to a single memory system, a Chip Multi-Processor (CMP) contains multiple processors/cores on a solitary chip [31 - 39]. Major fundamentals of Parallel computing are given below. To find maximum parallelism there is

Amdahl’s Law: It states that if s be the fraction of work done sequentially, so \((1-s)\) is fraction parallelizable [40]

\[
P = \frac{\text{Speedup (P)}}{\text{Time (1)/Time(P)}}
\]
\[
= \frac{1/(s + (1-s)/P)}{1/s}
\]

There are some keys of parallelism which we have considered at the time of parallelism of this research.

Granularity: It is defined as the numeral of basic units. Sometime it is coarse grained (Few Tasks of more powerful computing) and some time it is Fine Grain (Large Number of Small parts and less powerful computing)[41].

Implicit and Explicit Parallelism: We can defined parallel processing as two type First is Explicit in which algorithms includes instructions to specify which processes are built and executed in parallel way and Second is Implicit in which compiler has the task of inserting the necessary instructions to run the program on a parallel computer[41].

Synchronization is also necessary for making an algorithm from sequential to parallel as it prevents from the overlapping of two or more processors [41].

Latency: it is defined as the time conversion of information from request to accept [41].

Scalability: If we talk about scalability than It is defined as the capability of an algorithm to preserve its effectiveness by escalating the number of processors and the size of the problem in the same percentage [42]. Initially processors were developed with using only one core. As per the requirement multicore processors were developed later. Multi core processors may contain 2 cores as present in dual core Central processing units, for example Intel Core Duo and AMD Phenom II X2 , four cores are present as in quad core CPUs,
e.g. Intel i5 and i7 processors and AMD Phenom II X4, six cores in hexa core CPUs, e.g. AMD Phenom II X6 and Intel Core i7 Extreme version 980X, eight cores are there in octa core processors, like Intel Xeon E7 2820 and AMD FX 8350 and ten cores e.g. Intel Xeon E7 2850, or more[43].

Performance of the multi core processors is basically dependent on the designed algorithm and their implementation. As per Amdahl’s law possible gains are restricted by the portion of the software which can be executed in parallel concurrently on various cores. Basic architecture of dual core system is given below.

![General Dual Core Structure](image)

For developing environment of parallel computing there are several tools available as HPF, Message Passing, Active Messages, Fortran, Parallel C, Parallel C++, Data Structure Libraries (PETSc 2.0 for MPI, Multipol, LPARX), Numeric Libraries (LAPACK ) and Many More. MATLAB plays a very vital role for developing parallel programming model it is extensively used for prototyping an algorithm. It is having parallel computing tool box which can be effectively used in many applications. Teng-Yi Huang*, Yu-Wei Tang and Shiun-Ying Ju uses MATLAB through C for parallel computation [45]. NVIDIA finds near about 15X speed up using MATLAB [46]. Simi V.R. Justin Joseph and Praveer Sihota analyzed performance measurement by MATLAB[47]. So we can see that there are so many authors developed their tool by using MATLAB.

Java is also very important tool for multithreading with the following reason we have used this as the major tool with the following reason. Java uses better resources utilization. Designing of the programs are very simple. Performance on multiple processors is very good. Programs are very responsive in this case.

Apart from this there are many disadvantage of multithreading in java but by synchronization it with MATLAB we can overcome all the drawbacks of this as we have used this in our research.

### 3. PROPOSED METHOD: IMAGE REGISTRATION USING MUTUAL INFORMATION IN PARALLEL COMPUTING ENVIRONMENT

Followings are the description of our proposed method for MI based parallel image registration. Here we are going to discuss developed framework, Computation of Mutual Information on Individual core, Joint Histogram Calculation.
Framework for Parallel Image Registration Using Mutual Information (MI) Method in Multicore Architecture

Figure 3: Framework of the Proposed Method

*Steps for Parallel Computation of MI (Mutual Information)*

**STEP 1:** Reading of two images on different cores.

**STEP 2:** Computation of Joint Histogram

**STEP 3:** Normalization of the joint histogram

**STEP 4:** Find Sum of the rows and columns of joint histogram

**STEP 5:** Partition the array of rows and columns into the number of logical cores available.

**STEP 6:** Distribute the part of the images on the logical cores available.

**STEP 7:** Calculate marginal entropies of the individual part of image by maintaining synchronization between activated cores.

**STEP 8:** Accumulate the data of marginal entropy from activated individual logical cores.

**STEP 9:** Calculation of the joint entropy and Mutual Information (MI) on client.
Steps for Calculation of Joint Probability Distribution on logical core

STEP 1: Start

STEP 2: Calculate R = No of rows of Image1, C = No of Columns of Image2

STEP 3: N = R.

STEP 4: Zero Matrix H of Size N X N.

STEP 5: For I = 1: R

STEP 6: For J = 1: C

STEP 7: H(Image1(I,J) + 1, Image2(I,J) + 1) = H(Image1(I,J) + 1, Image2(I,J) + 1) + 1.

STEP 8: End

STEP 9: End

STEP 10: End

4. EXPERIMENTAL SETUP AND RESULTS OBTAINED

This Section describes about the Experimental Setup developed to implement the proposed intelligent method. We have implemented 60 Experiments with the following experimental setup, plot of the result obtained, registration of images and working of core i7 processor.

Setup for Experiments:

Hardware Configuration:
Processor: Intel(R) Core(TM) i7-3770 CPU @3.40 GHz
RAM (Random Access Memory) : 4 GB
Hard Disk Drive: 320 GB

Software Environment:
System Type: 64 Bit operating System, x64-based processor.
Development Tools: MATLAB, Intel Compiler 9.1, jdk–6–windows–i586

Parallelization Scheme in MATLAB [20]
Multithreaded Parallelism and Explicit Parallelism

Following Table illustrates the Comparative study of execution time of sequential and parallel implementation. We have mark out the speedup with significant result

4.1 Image Set: Twenty Gray Scale Images (Different Size – 256 X 256, 512 X 512, 1024 X 1024, 2048 X 2048)

<table>
<thead>
<tr>
<th>Images</th>
<th>Size</th>
<th>Sequential Execution with Implicit Multithreading</th>
<th>Parallel Execution with Explicit Multithreading</th>
<th>Speedup (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image 1</td>
<td>256 X 256</td>
<td>12.1</td>
<td>6.7</td>
<td>180.59</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Images</th>
<th>Size</th>
<th>Sequential Execution with Implicit Multithreading</th>
<th>Parallel Execution with Explicit Multithreading</th>
<th>Speedup (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image 5</td>
<td>2048 X 2048</td>
<td>6772.2</td>
<td>190.30</td>
<td></td>
</tr>
</tbody>
</table>

Now we are going to show the plot of best time comparison and maximum speed up from different images and speed up plot for this

Figure 4: Plot of Comparison of Time
4.2 Image Set: Twenty RGB Images (Different Size – 256 X 256, 512 X 512, 1024 X 1024, 2048 X 2048)

Table 2: Comparison of sequential and parallel execution of MI based IR for RGB images

<table>
<thead>
<tr>
<th>Images</th>
<th>Size</th>
<th>Sequentially Execution with Implicit Multithreading</th>
<th>Parallely Execution with Explicit Multithreading</th>
<th>Speedup (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image 1</td>
<td>256 X 256</td>
<td>14.5</td>
<td>7.8</td>
<td>185.89</td>
</tr>
<tr>
<td>Image 1</td>
<td>512 X 512</td>
<td>235.2</td>
<td>120.62</td>
<td>194.99</td>
</tr>
<tr>
<td>Image 1</td>
<td>1024 X 1024</td>
<td>5922.4</td>
<td>3370.8</td>
<td>175.69</td>
</tr>
<tr>
<td>Image 1</td>
<td>2048 X 2048</td>
<td>13126.1</td>
<td>7344.5</td>
<td>178.72</td>
</tr>
<tr>
<td>Image 2</td>
<td>256 X 256</td>
<td>13.7</td>
<td>7.8</td>
<td>175.64</td>
</tr>
</tbody>
</table>

Execution of MI based IR for RGB Images

Figure 7: Plot of Comparison of Time

Figure 8: Speed UP Plot for RGB Images
### 4.3 Image Set: Twenty different Brain Dicom Images of size 256 X 256

**Table 3: Comparison of sequential and parallel execution of MI based IR for Dicom images**

<table>
<thead>
<tr>
<th>Images</th>
<th>Size</th>
<th>Sequentially with Implicit Multithreading</th>
<th>Parallelly with Explicit Multithreading</th>
<th>Speedup (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image 1</td>
<td>256 X 256</td>
<td>13.55</td>
<td>5.01</td>
<td>270.45</td>
</tr>
<tr>
<td>Image 2</td>
<td>256 X 256</td>
<td>13.98</td>
<td>4.98</td>
<td>280.72</td>
</tr>
<tr>
<td>Image 3</td>
<td>256 X 256</td>
<td>12.60</td>
<td>5.22</td>
<td>241.37</td>
</tr>
<tr>
<td>Image 4</td>
<td>256 X 256</td>
<td>14.2</td>
<td>5.17</td>
<td>274.66</td>
</tr>
<tr>
<td>Image 5</td>
<td>256 X 256</td>
<td>13.56</td>
<td>5.32</td>
<td>254.88</td>
</tr>
<tr>
<td>Image 6</td>
<td>256 X 256</td>
<td>14.02</td>
<td>4.77</td>
<td>293.92</td>
</tr>
<tr>
<td>Image 7</td>
<td>256 X 256</td>
<td>13.24</td>
<td>4.23</td>
<td>313.01</td>
</tr>
<tr>
<td>Image 8</td>
<td>256 X 256</td>
<td>14.16</td>
<td>5.19</td>
<td>272.83</td>
</tr>
<tr>
<td>Image 9</td>
<td>256 X 256</td>
<td>13.43</td>
<td>4.76</td>
<td>282.14</td>
</tr>
<tr>
<td>Image 10</td>
<td>256 X 256</td>
<td>12.80</td>
<td>4.38</td>
<td>292.23</td>
</tr>
<tr>
<td>Image 11</td>
<td>256 X 256</td>
<td>14.03</td>
<td>5.98</td>
<td>234.61</td>
</tr>
<tr>
<td>Image 12</td>
<td>256 X 256</td>
<td>14.98</td>
<td>6.01</td>
<td>249.25</td>
</tr>
<tr>
<td>Image 13</td>
<td>256 X 256</td>
<td>13.78</td>
<td>5.34</td>
<td>258.05</td>
</tr>
<tr>
<td>Image 14</td>
<td>256 X 256</td>
<td>13.93</td>
<td>4.68</td>
<td>297.64</td>
</tr>
<tr>
<td>Image 15</td>
<td>256 X 256</td>
<td>14.45</td>
<td>5.38</td>
<td>268.58</td>
</tr>
<tr>
<td>Image 16</td>
<td>256 X 256</td>
<td>15.01</td>
<td>6.10</td>
<td>246.06</td>
</tr>
</tbody>
</table>

**Figure 10:** Plot of Comparison of Time

**Figure 11:** Speed UP Plot for Dicom Images

**Figure 12:** Average Result of Image Registration

Following figure represents copious consumption of the cores in i7 processor.
work can be easily implemented on multi core processor which are easily available in the market and produce tremendous result. Image Registration based on Mutual Information is time consuming processes as it is having several task to be executed. In this paper, we have assigned individual task to each activated cores after maintaining the synchronization between them. After execution of the particular task assigned core/processor is free to execute another existing task in queue. Main advantage of this work is that we have parallelized all the time consuming steps of MI based image registration as a result we can see that the performance in terms of speed up of the developed approach is extensively excellent. Another major advantage of this approach is that this is able to work on multi core processor having no GPUs. It is tested on the variety of Gray Scale, RGB and Dicom images as discussed above and able to give result in minimum execution time compared with existing algorithm of parallel image registration. For the future work’s point of view we can test this work on cluster of computers and the computer having GPU. Apart from this we can add the concept of memory sharing between cores/processors/computers in this work.

8. REFERENCES


[31] Help, MATLAB Software.


[36] Slabaugh, Richard Boyes, om Xiaoyun Yang, “Multicore Image Processing with OpenMP”.


[38] Concepts of Parallel Computing by NVIDIA


