A Grid Service for the Interactive Use of a Parallel Non-Rigid Registration Algorithm

Radu Stefanescu, Xavier Pennec, Nicholas Ayache
{Radu.Stefanescu,Xavier.Pennec,Nicholas.Ayache}@sophia.inria.fr
INRIA Sophia, Epidaure, 2004 Rte des Lucioles, BP93, F-06902 Sophia-Antipolis Cedex, France

Summary

In order to improve the usability of a non-rigid registration software running in parallel on a cluster of workstations, we have build a registration grid service. The system is composed of a graphical user interface on the user side that interacts in a complex and fluid manner with the registration software running on a remote parallel computer. Although the transmission of images back and forth between the computer running the user interface and the cluster running the registration service adds to the total registration time, it provides a simple way of using the registration software without heavy infrastructure investments in hospitals. The system exhibits good performances even if the user is connected to the grid service through a low throughput network such as wireless network interface or ADSL.

Keywords: grid service, non-rigid registration, cluster, medical imaging.

1. Introduction

Non-rigid registration recovers a point-to-point correspondences between the voxels of two images. The procedure is used in many domains of medical imaging, ranging from building and usage of medical atlases [4] to pre-operative planning [5].

In order for registration to be really used in clinical environments, physicians need software tools that are precise and robust. Furthermore, care must be taken regarding the usability of such tools. One primary issue is the execution time, which must not be larger than a few minutes. However, since healthcare organizations are usually more willing to invest in common medical equipment than in high-performance computing, the software should not require hardware that is too expensive to be acquired by a hospital (more expensive than common medical equipment), or that is too hard to administrate.

In previous work, we developed a non-rigid registration algorithm able to reliably recover large deformations [1] and that admits an efficient parallel implementation on a cluster of personal computers. The algorithm solves a two-part PDE: If we denote by $U=(u_1,u_2,u_3)$ the deformation that links the source image J to the target image I , then the first term minimizes the square distance $\|I(p)-J(p+U(p))\|^2$ between the target image and the deformed source image. The second term is based on two assumptions: the deformation is supposed to be regular, and the degree of regularity spatially vary based on the capability of the materials to deform. Hence, the *regularity term* has the shape of a diffusion term, where the conductivity represents the local ability to deform. The two terms are weighted by a confidence factor k that estimates at each point the reliability of the matching process:

$$\frac{\partial u_i}{\partial t}(p) = k(p)\nabla \sum_{p\in\Omega(images)} \|I(p)-J(p+U(p))\|^2 + (1-k(p))\nabla(d\nabla u_i)$$

We also provided in [1] a parallel implementation of this algorithm, using a cluster of personal computers, which lowers the registration process time to less than five minutes.

Though precise, robust and fast, our registration software suffers from several issues regarding its usability: Firstly, it is command line based, which means that once the algorithm is started on the two input images, there is no mean to *interact* with the algorithm until it ends. Desirable interactions are: stopping a registration process either because it is about to diverge or because the level of convergence is sufficient for the user's needs; modifying parameters, visualizing the intermediate result. In the near future, we would also like to enable the user to add some medical expertise in order to guide the registration process. All this requires the creation of a *graphics user interface* (GUI) that allows the user to master the registration software.

We conceive our system as being composed of two interacting modules: the registration software and the graphics user interface. The registration module performs the computation. It is run on a parallel computer that usually needs special conditions to operate, such as air-conditioned rooms. Therefore, such a machine cannot lie next to the user and is more likely to find a suitable place as a shared resource in a data center than in an operating or pre-operative planning room. The GUI provides the interaction between the registration module and the user. It also deals with the visualization of the three dimensional images involved. For visualization performance reasons, the GUI must be executed on the user's computer. The computers running the two components of our system must be connected through a network. Due to the different conditions in which these two computers must operate, such a network may be a long distance one. Moreover, since in intra-operative registration it is important that additional wires not to be installed in the operating room, we want to be able to provide our service through a wireless, hence low bandwidth, network interface.

The purpose of our work is to implement the registration as a *grid service* able to connect the clinically useful user interface to a computational center. This interaction must be sufficiently refined to allow the user to manipulate every intimate aspect of the registration software.

A previous work on a non-rigid registration service has been done by Ino *et al.* [2], which used a high-end cluster connected to the visualization workstation through a high-bandwidth Wide Area Network (WAN). We believe that the infrastructure the authors used is not representative of the one available in most hospitals. Lower end to average clusters and WAN's are in our opinion more realistic hardware platforms for any grid service for medical use. Moreover, in this paper we bring more detailed information about the mechanisms used to connect our user interface to the grid service and the security and usability concerns that are raised by such a system.

2. Objectives

To summarize, we want to build a registration grid service implemented on a cluster of PC's on one side, and a graphical interface able to control it on the user side. In order to connect them, we need a communication library that fulfills several conditions:

1) The communication library has to be as flexible as possible. If the users makes two requests A and B to the GUI, and request A is transmitted to the grid service before request B, it is important that no constraint be imposed to the responses of the two request. The response to B may arrive before or after the response to A, and one or both responses may be never sent at all. This represents an argument against the simple-to-use *client-server model* used by

protocols such as CORBA or RPC. Therefore, we prefer a *message-passing model* for our communication library.

2) Both the requests and their responses must be able to pass through firewalls. If sockets are to be used, gates must be open in the firewall for each port number in use. Hence, port numbers may not be chosen randomly, as do many MPI implementations. We prefer to limit the number of ports in use.

3) The access to the grid service must be secured. A public key authentication such as RSA fulfills our needs regarding access permissions to the grid service.

4) Communications should be secured. Since confidential medical data is transmitted through a WAN that is not always under the hospital's full control, this data should be encrypted.

5) The whole system should preserve the anonymity constraints imposed be current regulations [2]. Image files usually contain information about the subject's identity. This information should never make its way to the non-controlled WAN. This excludes the transmission of the input images as such through a file transfer protocol. The solution is to send only the data necessary for the registration: the image sizes, their voxel sizes and the image voxel data.

6) In order to maximize the accessibility to the grid service, the communication library should spare bandwidth. Therefore, each message transmitted through the WAN should be compressed before sending.

3. Methods

We designed a message passing library that is able to transmit messages back and forth between the GUI and each of the nodes running the grid service. Thanks to compression and encryption, the transmission of messages is fast and secure.

3.1. Security issues

Both the GUI and the grid service are isolated from the rest of the world by firewalls. Two security issues are important:

- Communications must be able to pass through the two firewalls.
- The user must be authenticated before being allowed to use the service.
- Communication should be encrypted.

We use *ssh tunneling* in order to fulfill these three requirements. If a ssh tunnel is not created the registration service is not visible by the user. In order to create the ssh tunnel, the user must authenticate using a user name and a password delivered by the service administrator. The tunnel also provides the encryption of the data flow between the GUI and the service.

Despite all these security mechanisms, a possible anonymity breach can come from the facial reconstruction that can be done on some MRI images. The solution would be to modify the images before transmission through the network in a way that does not affect registration but makes facial reconstruction impossible. We have not tackled this problem.

3.2. Message passing

When *the grid service sends a message to the GUI*, the communication is performed by the master node. Once a message is created by the former, it is packed into a communication buffer. This procedure eliminates the difference in endianism between the two communicating processors. The message buffer is compressed, encrypted and written into the socket that connects the master node of the grid service to the GUI. The receiving processor decrypts, uncompresses and unpacks the message before processing.

A message that is *sent by the GUI to the parallel computer* running the grid service must be dispatched to each node of the cluster in order to be processed. The message is transmitted from the GUI to the master node of the grid service using the procedure described above. When the message is received by the master node, it is decrypted, decompressed, unpacked and then repacked using MPI's message packing module. It is afterwards broadcasted to the nodes of the parallel computer running the grid service (figure 1).

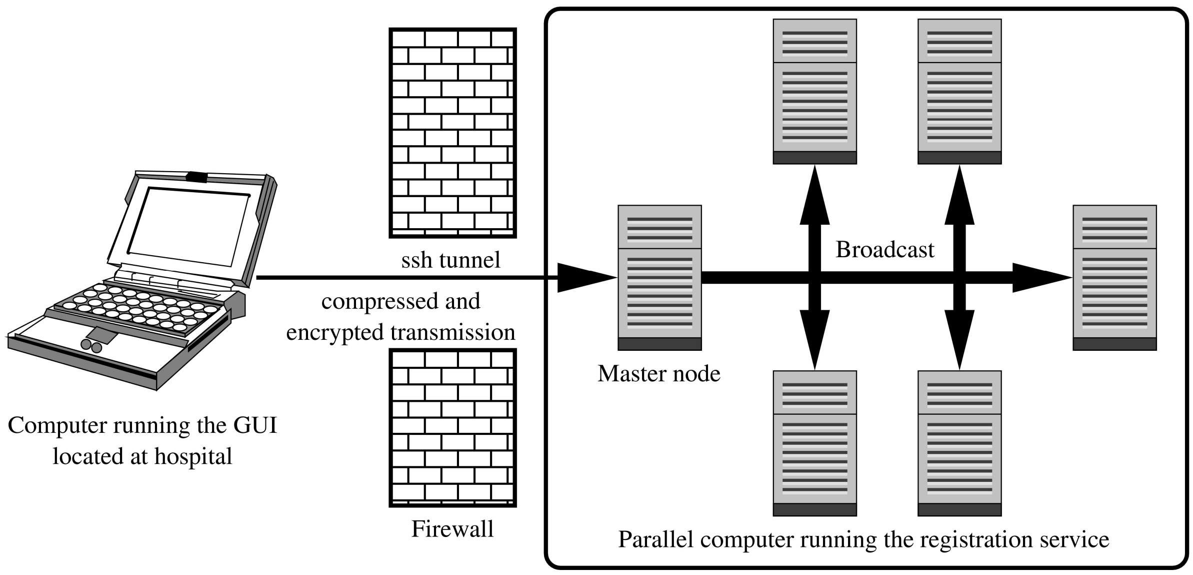

Figure 1 A message sent by the GUI is compressed, encrypted and sent through the ssh tunnel to the master node, which broadcasts it to all nodes of the parallel computer running the registration service.

4. Results

The final system is the following: After user authentication, the GUI allows the loading and comparative visualization of the images to register. A user interface button enables the user to contact the grid service and start the registration. During the process, the user is informed in real time about the status of the algorithm (small size messages). An intermediate result image (several megabytes of data) is regularly sent to the GUI which displays it. At any time the user has the option to abort the registration. Upon termination of the algorithm, the registration result is sent back to the GUI.

We have tested our system by running the GUI on a 600MHz Pentium III laptop equipped with a 11MB/s wireless network interface. The grid service runs on 15 2GHz Pentium IV PC's linked through a 1GB/s Ethernet network and protected by a firewall running on a 2GHz Pentium IV PC. All systems run the Linux operating system. The total size of the input images is 47.2MB. After compression, they reach a size of only 3.3MB. The transmission of the input images takes 39 seconds. For comparison, if compression is not used, the transmission takes 2 minutes and 26 seconds. During the execution of the algorithm, one update of the intermediate result takes 18 seconds. The extra time took by the transmission of images through the network is of about 2 minutes, thus bringing the total registration time to about 7 minutes. However, thanks to the user interface and the grid service, the registration software becomes usable in the clinical environment. Thus, we believe that the two extra minutes in computation time are tolerable.

We also tested our system in a worst case scenario by using a home Asymmetric DSL connection between the user interface and the grid service. The maximum transfer rate is 16kB/s from the GUI to the service and 64kB/s from the service to the GUI. The image upload time is about 3min30, and the result update time is about 1min30. In this extreme case, the total registration time reaches ten minutes. However, this proves that our non-rigid registration service can be used from almost everywhere with a very modest equipment.

5. Conclusion

We have presented a system that enables a clinician to transparently use a powerful but computationally intensive non-rigid registration algorithm run by a parallel computer physically located at a large distance. The system combines the speed and precision provided by the parallel computer to the ease of use of the physician’s workstation. A graphics interface enables the user to supervise the execution of the algorithm in real time. In the future, we will modify the system so that physicians will be able to use their clinical expertise to guide the registration. We will add an integrated authentication, the automatic creation of the ssh tunnel and the remote execution of the service by the user. A test in a clinical setup is also in preparation.

Our non-rigid registration service opens up to the clinical practice numerous medical image analysis applications such as the usage of brain atlases or image guided therapy. While medical imaging algorithmic issues has mostly been dealt with, grid management problems still have to be addressed. Standardization of the communication library, a discovery service and a payment system are necessary to transform this prototype into a commercial system.

References

[1] R. Stefanescu, X. Pennec, and N. Ayache. **Grid Enabled Non-Rigid Registration with a Dense Transformation and A Priori Information**. In *Proc. of , Medical Image Computing and Computer-Assisted Intervention(MICCAI'03)*, LNCS, November 2003. Springer Verlag.

[2] JAM. Herveg, and Y. Poullet. **Directive 95/46 and the Use of Grid Technologies in the Healthcare Sector: Selected Legal Issues.** In Sofie Norager, editor, *Proc. of HealthGrid'03*, Lyon, January 2003. European Commission, DG Information Society.

[3] F. Ino, K. Ooyama, Y. Kawasaki, et al. **A High Performance Computing Service over the Internet for Nonrigid Image Registration**. In *Proceedings of Computer Assisted Radiology and Surgery 17th International Congress and Exhibition (CARS 2003)*, International Congress Series 1256, Elsevier Science, pp. 193--199, London, UK, June 2003.

[4] DLG. Hill, JV. Hajnal, D. Rueckert, SM. Smith, T. Hartkens, and K. McKleish. **A Dynamic Brain Atlas.** In Takeyoshi Dohi and Ron Kikinis, editors, *Medical Image Computing and Computer-Assisted Intervention (MICCAI'02)*, volume 2489 of *LNCS*, Tokyo, pp. 532--539, September 2002. Springer.

[5] S.K. Warfield, F. Talos, A. Tei, et al. **Real-Time Registration of Volumetric Brain MRI by Biomechanical Simulation of Deformation during Image Guided Neurosurgery**. Computing and Visualization in Science, Springer, 5:3-11, 2002.