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

Towards Ubiquitous Access of Computer-Assisted Surgery Systems

Hui Liu[†‡], Hanping Lufei[‡], Weisong Shi[‡], and Vipin Chaudhary[‡]
[†] Xidian University [‡] Wayne State University

Abstract— **Traditional stand-alone computer-assisted surgery (CAS) systems impede the ubiquitous and simultaneous access by multiple users. With advances in computing and networking technologies, ubiquitous access to CAS systems becomes possible and promising. Based on our preliminary work, CASMIL, a stand-alone CAS server developed at Wayne State University, we propose a novel mobile CAS system, UbiCAS, which allows surgeons to retrieve, review and interpret multimodal medical images, and to perform some critical neurosurgical procedures on heterogeneous devices from anywhere at anytime. Furthermore, various optimization techniques, including caching, prefetching, pseudo-streaming-model, and compression, are used to guarantee the QoS of the UbiCAS system. UbiCAS enables doctors at remote locations to actively participate in remote surgeries, share patient information in real time before, during, and after the surgery.**

I. INTRODUCTION

Computer-Assisted Surgery (CAS) has broad applicability to human health. Several CAS systems have been designed and developed such as [2], [3], [4], [16]. All of these CAS systems are isolated solutions located in the operating room. Therefore, all the surgery data preparation, registration, segmentation, planning, and related operations are restricted to physically fixed machine which reduces the potential for telementoring, telepresence, and telesurgery in CAS systems.

In practical clinical use, although abundant research work has been done on mobile hospital applications [1], [6], [10], [15], few attempts have so far been made to support mobile CAS applications on heterogeneous devices in distributed environment.

This work is supported by Michigan Life Science Corridor. The overall objective of this project is to develop infrastructure and critical mass of multidisciplinary investigators that will advance research in computer-assisted surgery (CAS), leading to the adoption of new and innovative surgical approaches in Michigan and elsewhere. In the last three years, the researchers at Wayne State University have successfully developed a CAS system CASMIL [9]. CASMIL is a comprehensive Image-guided Neurosurgery System with extensive novel features. It is an integration of various modules like rigid and non-rigid body co-registration (image-image, image-atlas, and image-patient), automated 3D segmentation, brain shift predictor, knowledge based query tools, intelligent planning, and augmented reality. However, CASMIL is also a stand-alone CAS system. In order to allow the surgeons access to CASMIL from anywhere at anytime, there is a need to augment the stand-alone CASMIL with a mobile CAS system.

Based on these observations and preliminary work we have done in CASMIL, the current research goal of our project is to design and develop a mobile CAS system, *UbiCAS*, that enables doctors at remote locations to access and plan the surgery, to actively participate in remote surgeries, and to share patient information and exchange opinions in real time before, during, and after the surgery. Besides supporting the heterogeneous user devices and a full set of surgery related functionalities, QoS is a key requirement for a practical system. We adopt various optimization mechanisms in UbiCAS to improve system performance metrics in terms of user-perceived latency, scalability, and security.

There are several challenges in the UbiCAS system. Surgery-related function implementation and conciliation on heterogeneous devices, especially on resource-constraint devices like PDAs and smartphones are challenging. In a distributed environment, the diverse networking technologies, huge medical image transmission, security and privacy are also key factors in this system. We use caching, prefetching, pseudo-streaming-model, compression, and concurrent technologies to guarantee the QoS of UbiCAS. In this paper, we discuss and report the design and implementation of UbiCAS.

II. SYSTEM OVERVIEW

Different from traditional stand-alone CAS systems, UbiCAS addresses a distributed CAS application scenario as shown in Figure 1. Distributed users connect to UbiCAS from different places using various devices. Each user has his own device and network connections. For example, user 1 uses wireless network to connect his Pocket PC to the system. User 2 uses dialup network to connect his laptop to the system. The UbiCAS server is set in the operating room and proxies may be deployed at different medical buildings, campuses and hospitals. When a surgeon moves from the operating room to a far away medical campus, he can continue accessing the UbiCAS server through the nearest proxy to him with his handheld device.

Proxies play important role in the system. At each stage, from the pre-surgery to the post-surgery, lots of computation need to be done in a prompt and secure way. It is impractical for CAS server to perform the entire workload by itself.

Hui Liu is with the Software Engineering Institute, Xidian University, Xian 710071, China liuhui@xidian.edu.cn. The work was done during her visit at Wayne State University.

Hanping Lufei, Weisong Shi and Vipin Chaudhary are with the Department of Computer Science and Computer Assisted Surgery Lab, Wayne State University, Detroit, MI 48202, USA [hlufei, weisong, vipin]@wayne.edu.

Proxies act as offload processing assistants to share the workload of requested tasks, such as secure and real time communication between CAS server and remote users. With the support from adjacent proxies, remote users can focus on the surgery related computing as if the information from the remote CAS server is provided locally.

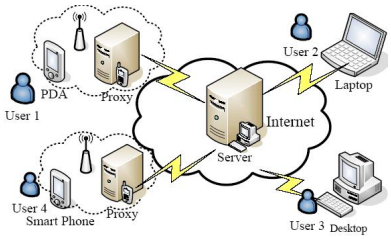

Fig. 1. The network deployment structure of UbiCAS.

UbiCAS includes three components, namely, server, client and proxy. The UbiCAS server is an extension of CASMIL and provides traditional CAS functionalities. UbiCAS clients provide surgery-related GUI interfaces on various user devices. The UbiCAS proxy is an optional component used to improve system performance. The design of each component and the security issues are discussed in the following sections.

III. System Design

A. UbiCAS Server

Based on our preliminary work, the stand-alone CASMIL, we implement UbiCAS Server on Windows platform, using C++ languages, ITK [8], OpenGL [14], VisSDK, CMake and Trolltech Qt framework [17], as shown in Figure 2.

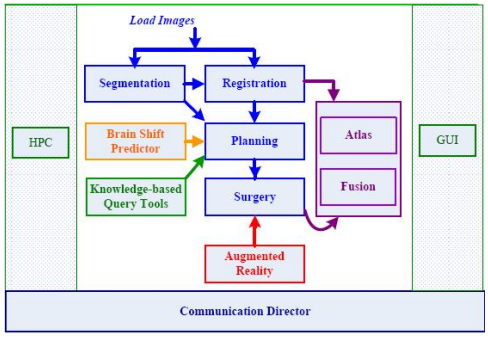

Fig. 2. The UbiCAS server structure.

Surgery-related Functionalities: The UbiCAS server integrates human brain data modeling, information extraction, management and retrieval together for automatic identification of structures. It also includes automated atlas integration, information analysis and data mining for intelligent surgery planning. Other vital modules include three-dimensional segmentation and augmented reality. High Performance Computing (HPC) is used to speed up computationally intensive tasks involved in CAS, thus providing near real time results while performing surgery. More details of the CAS engine (CAMIL) can be found in [9].

Extensions of CASMIL: The communication directory module is added to CASMIL so as to make it accessible to other computers. Note that we use asynchronous socket to ensure better scalability. Several compression algorithms have been implemented on the server-side. Based on the types of client devices, the server can adapt different compression algorithm to provide the best user-perceived latency [11].

B. UbiCAS Client

User-friendly User Interface: Various functionalities and devices demand particular user interfaces (UI). For example, PocketPCs and smart phones are both limited by memory/computing/power capacity. So, we should put only simple display and user-input tasks on those devices. The complicated backend processing should be supported by the central server or proxies. For PocketPCs and smartphones, the display screen is small; whereas the desktop and laptop have large screens. Thus, we should provide different UI for different devices. The CAS server in Operating Room has a large screen, the display screen is divided into four parts, and each part provides a different angle of the patient's brain, 2D or 3D, as shown in Figure 3. On the contrary, the screen of PocetPCs and smartphones is much smaller; we need to do content-adaptation to these devices by shrinking or filtering the content, as exemplified in Figure 4.

Functionalities on Handhelds Devices: Currently, the following functionalities are provided on handheld devices: (1) remote DICOM images download; (2) local DICOM image validation, reading and displaying; (3) advanced image processing functions, i.e., Zoom in, Zoom out, rotating, and brightness adjustment, etc.; (4) GUI interfaces to execute the complicated neurosurgical procedures (i.e., segmentation, registration and planning) on the UbiCAS Server. To the best of our knowledge, we are the first to provide such neurosurgical procedures on handheld devices. Figure 4 shows the scenario where a surgeon executes a segmentation operation on an HP iPAQ 4300 PocketPC to find out the probably scope of a tumor.

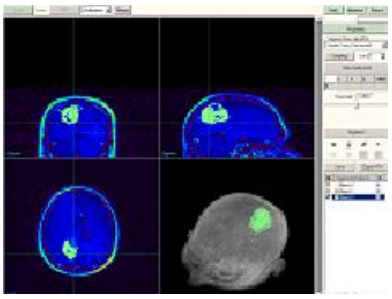

Fig. 3. UbiCAS Server.

Cross-Platform Compatibility: The reuse of the same client software on various devices can greatly reduce system development and maintenance cost. Java technology is used on client-side for its platform-independent nature. By selecting the corresponding Java Virtual Machine, a UbiCAS

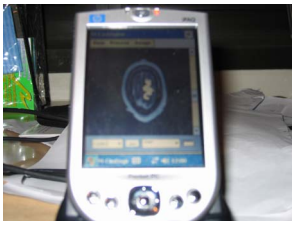

Fig. 4. Snapshots of segmentation operation on a PocketPC.

Client can run on various devices. For example, Jeode Java Virtual Machine [7] is used for PocketPC.

Pseudo-Streaming Model: Inspired by the streaming technology, we design and implement a pseudo-streaming model on the client-side, which is implemented using two parallel threads. That is, one thread is in charge of the GUI part, whereas the other thread is in charge of the lower lever communication. Therefore the surgeon can instantly observe the continuous slices one by one without being aware of the downloading action executed simultaneously at the back-end.

C. UbiCAS Proxy

The UbiCAS proxy is the component used to improve latency and scalability of UbiCAS. Three optimization mechanisms are adopted as shown in Figure 5.

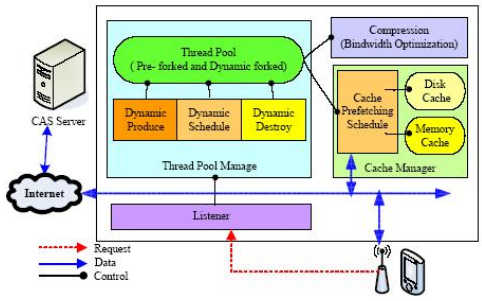

Fig. 5. The UbiCAS Proxy structure.

Compression and Deployment: A comprehensive patient's brain image set may include 100 DICOM [5] slices. So, a good compression algorithm is needed to reduce the huge image transmission overhead. In heterogeneous environment, the final user latency is decided by several factors in terms of network bandwidth, compression ratio, compressing and decompressing time on various devices with diverse CPU capability. So the deployment of compression functions is also a key issue to final user latency. We deploy Bitmap-Diff, RAR, ZIP and 7Z compression algorithms on actual physical hardware and networks to find the best compression algorithm and deployment strategy.

Caching and Prefetching: Least Recently Used (LRU) cache and prefetching mechanisms are used to reduce user access latency. Prefetching refers to fetching data in advance of user's request. A user-transparent prefetching mechanism is implemented based on the regular filename naming mechanism in UbiCAS.

Concurrent Support: In order to cope with the heavy workload during concurrent access by multiple surgeons, pre-connection, pre-spawn, and dynamic scheduling techniques are implemented in UbiCAS. Pre-connection and pre-spawn can reduce the overhead of creating new thread and the overhead of establishing connections. Dynamic scheduling can create and destroy service threads according to current system workload so as to cope with the peak time and save system resources.

D. Security and Privacy

Security and privacy are two important issues of the system. From the patient privacy's point of view, access to the patient information should be granted in a controlled way. From the data integrity's perspective, information should not be altered during transmission.

In terms of privacy, we use access control model to guarantee different doctors have different views of the same data set. Currently, role-based access control model is used in our system. Permissions are assigned to roles rather than to individual users. Roles are created for various job functions, and users are assigned to roles based on their qualifications and responsibilities. For example, the radiologist may only take care of the diagnosis of patient's X-ray images instead of patient's profile. More details can be found in M-CASEngine [12].

In terms of data integrity and encryption, we have to take care of the malicious proxies. We consider the following scenarios: (1) data should be encrypted to preserve confidentiality and should be decrypted only by those with appropriate read permissions. (2) a malicious proxy should not be able to break data integrity by modifying content without a client detecting the change. For example, when a client requests some patient's DICOM files from remote server through a proxy, we use the opaque tokens generated by the server to handle these security issues. The tokens serve as a shared secret (between the client and proxy) with which to derive symmetric cryptographic keys for transmitting data from the proxy to client. The client can verify the integrity of retrieved data, as the token acts to bind the DICOM file content to a specific verifiable value.

IV. Results and Evaluations

UbiCAS is a prototype system under developing. Currently, we have implemented multiple surgery-related functionalities on various user devices in UbiCAS, such as segmentation, registration, and planning. The surgeon can retrieve single DICOM image or browse the continuous multiple DICOM slices of one patient in real time. UbiCAS also provides some advanced image processing functions on user device.

We tested UbiCAS for functionality and performance. In terms of system functionality, we focus on the correctness of each functionality and UI user-friendliness. We have deployed and tested UbiCAS on PocketPC, laptop, desktop these devices and verified its correctness.

For evaluating system performance, we built an experimental test-bed to simulate the real network environment to test the latency and scalability of UbiCAS, as shown in Figure 6. For the space limit, here we only describe our test-bed and present the segmentation latency in a slow network.

Experimental Platform: The experimental platform consists of five computers, as shown in Figure 6. The laptop and PocketPC are used as two kinds of client devices, the three desktop are used as proxy, network emulator, and server, respectively. This testbed simulates the application scenarios that if the user is far from the server, he can observe prompt response by accessing the server through the nearest proxy to him with his PDA. The transmitting workload in our experiment is a full set of DICOM images with 100 slices, the size of each DICOM slice is 134KB. NISTNet 2.0.12 [13], a network emulator, is used to simulate the network environment.

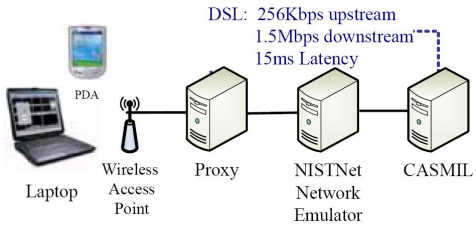

Fig. 6. Experiment testbed.

The Latency of Segmentation: Segmentation is a time-consuming neurosurgical functionalitiy executed on UbiCAS Server. In order to reduce the segmentation response time for the handheld users, what we can do is to try to reduce the transmission cost. However, there is a tradeoff between the transmission cost and the computing cost on server and the handhelds. So we experiment to judge whether or not compressing the segmentation result before transmission is useful. We measure the segmentation response time in two kinds of network connections, DSL and WLAN, on a PocketPC.

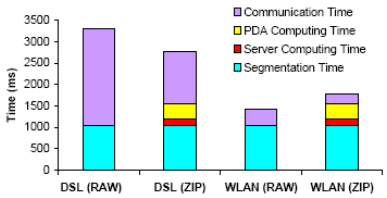

Fig. 7. Segmentation time.

Figure 7 shows that in slow DSL network connection, where the transmitting bytes determine the total time, compressing the segmentation result is a good choice. Opposite conclusion is got in WLAN. However, we can see that in the same network connection, there is no obvious latency gap (less than 0.5s) between the two corresponding bars, with or without compression. Compared with human reaction, the 0.5s latency gap may be negligible.

V. Conclusions and Future Work

In this paper, we propose a novel mobile CAS system, UbiCAS, which is part of the CAS project [9].

This system is being developed to provide the convenience, safety and security so that doctors at remote locations can access and plan the surgery, and actively participate in remote surgeries, share patient information in real time before, during and after the surgery. Besides providing a full set of surgery-related functionalities on heterogeneous devices, UbiCAS also gurantees security and QoS by adopting various optimization mechanisms.

So far, we have developed and tested UbiCAS on PocketPCs, laptops and desktops in a controlled environment. Our future work includes: (1) deployment of entire functionality of CASMIL and testing the correctness of each function and UI user-friendliness on other devices; and (2) deployment and testing of the system in real scenarios.

VI. Acknowledgments

The authors gratefully acknowledge Michigan Life Science Corridor (MLSC) for their support. We also want to thank other group members who help us understand and augment the existing CASMIL server.

References

[1] M. Ancona, G. Dodero, V. Gianuzzi, et al. Mobile Computing in A Hospital: The Ward-In-Hand Project. *Proc. of the 2000 ACM Symposium on Applied Computing*, Mar. 2000.

[2] Brainlab, `http://www.brainlab.com/scripts/website_english.asp`.

[3] K. Cleary, L. Ibez, S. Ranjan, and M. B. Blake. IGSTK: A Software Toolkit for Image-guided Surgery Applications. *Proceedings of 2004 Computer Assisted Radiology and Surgery 18th International Congress and Exhibition*, 2004.

[4] D. Dey, D. G. Gobbi, P. J. Slomka, et al. Automatic fusion of free-hand endoscopic brain images to three-dimensional surfaces: Creating stereoscopic panoramas. *IEEE Trans Med Imaging* 21(1), January 2002.

[5] Dicom standard, `http://medical.nema.org`.

[6] J. Favela, M. Rodrguez, A. Preciado, et al. Integrating Context-Aware Public Displays Into a Mobile Hospital Information System. *IEEE TRANSACTIONS ON INFORMATION TECHNOLOGY IN BIOMEDICINE* 8(3), September 2004.

[7] Insignia Jeode Runtime Environment, `http://www.insignia.com/content/products/jvmProducts.shtml`.

[8] Itk, `http://www.itk.org/`.

[9] G. Kaur et al. CASMIL: A Comprehensive Software/Toolkit for Image-guided Neurosurgeries. *International Journal of Medical Robotics and Computer Assisted Surgery (IJMRCAS)*, May/June 2006.

[10] Y. Lin, I. Jan, P. Ko, et al. A Wireless PDA-Based Physiological Monitoring System for Patient Transport. *IEEE TRANSACTIONS ON INFORMATION TECHNOLOGY IN BIOMEDICINE* 8(4), December 2004.

[11] H. Lufei and W. Shi. Fractal: A mobile code based framework for dynamic application protocol adaptation in pervasive computing. *Proceedings of IPDPS'2005*, Apr. 2005.

[12] H. Lufei, W. Shi, and V. Chaudhary. M-casengine: A collaborative environment for computer-assisted surgery. *Proceedings of 2006 Computer Assisted Radiology and Surgery 20th International Congress and Exhibition*, July 2006.

[13] NISTNet, `http://snad.ncsl.nist.gov/itg/nistnet/`.

[14] OpenGL, `http://www.opengl.org/`.

[15] A. Rovetta. Computer Assisted Surgery with 3D Robot Models and Visualisation of the Telesurgical Action. *Studies in Health Technology and Informatics* 70, 2000.

[16] Stryker, `http://www.stryker.com/navigation/neuro/index.htm`.

[17] Trolltech Qt, `http://www.trolltech.com/`.