The Virtual Microscope [*]

Renato Ferreira[†], Bongki Moon, Ph.D.[†], Jim Humphries[†], Alan Sussman, Ph.D.[†],
Joel Saltz, M.D., Ph.D.[† ‡], Robert Miller, M.D.[‡], Angelo Demarzo, M.D.[‡]

[†]UMIACS and Dept. of Computer Science
University of Maryland
College Park, MD 20742

[‡]Department of Pathology
Johns Hopkins Medical Institutions
Baltimore, MD 21287

{renato,bkmoon,humphrie,als,saltz}@cs.umd.edu
{rmiller,ademarz}@pds.path.jhu.edu

We present the design of the Virtual Microscope, a software system employing a client/server architecture to provide a realistic emulation of a high power light microscope. We discuss several technical challenges related to providing the performance necessary to achieve rapid response time, mainly in dealing with the enormous amounts of data (tens to hundreds of gigabytes per slide) that must be retrieved from secondary storage and processed. To effectively implement the data server, the system design relies on the computational power and high I/O throughput available from an appropriately configured parallel computer.

Introduction

The *Virtual Microscope* is a software system that provides a realistic digital emulation of a high power light microscope. The raw data for such a system can be captured by digitally scanning collections of full microscope slides under high power. An example of a portion of a digitally captured composite slide, stitched together by hand from multiple individually captured images, is shown in Figure 1. While the hardware for performing the data capture more effectively is rapidly becoming commercially available, the software support required to provide interactive response times for the standard behavior of a physical microscope has not been developed. These behaviors include continuously moving the stage and changing magnification and focus. In addition, a software solution can enable new modes of behavior that cannot be achieved with a physical microscope, such as simultaneous viewing and manipulation of one slide by multiple users. This paper describes the design of a complete software system for implementing the functionality of the Virtual Microscope, through a client/server architecture. The client software runs on a user's PC or workstation, while the database software for storing, retrieving and processing the microscope image data runs on a high performance parallel computer at a potentially remote site.

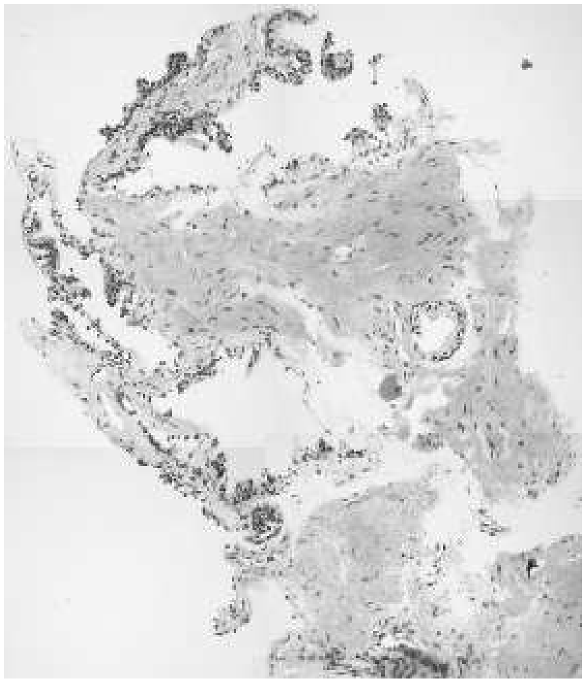

Figure 1: A 1700x2100 digitally captured image of a prostate cancer at 200X magnification

One obvious application of the Virtual Microscope software system is to simply emulate the usual behavior of a physical microscope, replacing cabinets full of slides with a digital storage subsystem. Retrieving a slide then becomes a matter of accessing the slide database, without

[*]This research was supported by the National Science Foundation under Grant #ASC 9318183, and ARPA under Grants #DABT 63-94-C-0049 (Caltech Subcontract #9503) and #F19628-94-C-0057. The Maryland IBM SP2 hosting the system was provided by NSF CISE Institutional Infrastructure Award #CDA9401151 and a grant from IBM.

requiring physical access to the slide. However, the Virtual Microscope can provide functionality that a physical microscope can never achieve. One example of additional capability is in a teaching environment, where an entire class of students can access and individually manipulate the same slide at the same time, searching for a particular feature in the slide. Another example is remote consultation, where a physician in a different part of the country can view a slide via the Internet, once the slide has been scanned and placed in an accessible database. Additional applications include coupling the Virtual Microscope to computation modules that can perform various types of processing, including

- three dimensional image reconstruction from data found in multiple focal planes and on multiple microscope slides,
- image registration and compositing that takes into account data obtained using various special stains that reveal the presence or absence of biochemical markers,
- image segmentation and pattern recognition to better characterize known malignancies, and
- applications that aid the pathologist in screening for possible malignancy.

The main difficulty in providing the functionality of the Virtual Microscope is dealing with the extremely large quantities of data required to represent a large collection of slides. For example, using the digitizing microscope currently available at Johns Hopkins, a single spot at a magnification of 200X produces a grid of 1000x1000 pixels. We estimate that an array of 50x50 spots is required to cover an entire slide, and each pixel is a three byte RGB color value. Under this scenario, one slide image requires about 7 GBytes. However, such an image captures only a single focal plane, and many specimens will require capture of from five to thirty focal planes. Clearly there is an enormous storage requirement, and there are also the attendant difficulties in achieving rapid response time for various types of inquiries into the slide image database.

Recent work related to some of the problems we are addressing with the Virtual Microscope come from telepathology. Pathology specimens are typically directly examined using a light microscope. Over the past 10 years, there has been increasing interest in technologies that make it possible to examine specimens at a distance. There are currently two forms of telepathology imaging: static and dynamic [1]. In the dynamic mode, live images of microscope slides are transmitted and visualized in real time. The remote pathology consultant is able to control the microscope stage and to select the image to be viewed. In static-image telepathology, the referring pathologist captures a small set of digital images that are transmitted to the consultant. The consulting pathologist relies on the referring pathologist to select tissue fields. A variety of telepathology projects have demonstrated the use of static, dynamic and hybrid forms of telepathology [2, 3]. In these terms, the Virtual Microscope can best be described as a form of completely digital dynamic telepathology.

The rest of this paper concentrates on the technical challenges that must be addressed to effectively support the functionality of the Virtual Microscope, and describes the design of a software system based on a client/server architecture to implement the Virtual Microscope. The client user interface allows the user to select slides to view, and provides the usual microscope functionality for viewing different portions of a slide at varying magnifications, and changing the current focal plane. The client is a Java [4] program that can be run on any PC or workstation that has the Java virtual machine available. The client connects, across a local network or the Internet, to a Java network server program running as the front end to a parallel C/C++ data server that accesses the slide image data stored across disks located on all the nodes of the parallel server machine. This client/server approach mirrors the design of the Maryland Titan system [5] for storage and retrieval of vast quantities of remote sensing satellite imagery, providing strong evidence that the design will provide the required performance. A prototype of the entire system is currently operational, and we end by describing our current status and future plans.

Technical challenges

Data Placement

As described in the previous section, managing extremely large quantities of data is the major problem in the design and implementation of the Virtual Microscope. Since the ultimate goal of the Virtual Microscope is to provide users with the illusion that they are using a physical microscope, we must be able to support the standard functions of a physical microscope in software with the same level of responsiveness and ease of use.

These requirements present technical challenges in the design and implementation of a Virtual Microscope. The image database must provide low latency retrieval of large volumes of two dimensional image data (representing a portion of a focal plane of a given slide) from secondary storage as well as efficient directory management for a large collection of slides. In this paper, we only address the first problem, but in future work we will investigate the use of relational database technology for cataloging collections of slides in the image database. The large volume of image data requires effective use of a large number of disk units, which in turn requires effective placement of multidimensional data sets (each slide consisting of multiple 2D focal planes) onto a large disk farm to maximize disk access parallelism and minimize disk access latency. Moon et al. have designed and evaluated many algorithms for effectively declustering multidimensional data sets to maximize disk parallelism [6], and also for clustering data on each disk to optimize single disk performance (minimizing disk seeks) [7].

A small number of operations must be supported by the Virtual Microscope system to implement the required functionality:

1. fast browsing through the slide to locate an area of interest,
2. local browsing to observe the region surrounding the current view,
3. changing magnification, and
4. changing the focal plane.

The digitized image from a slide is essentially a three dimensional data set, because each slide usually consists of multiple focal planes. In other words, each digitized slide consists of several two-dimensional images stacked on top of one another. However, the portion of the entire image that must be retrieved to provide a view into the slide for any given set of microscope parameters (area of interest, magnification and focal plane) is two dimensional. Therefore, to optimize performance, each two-dimensional image (a focal plane) should be an independent unit for the declustering algorithm to maximize disk parallelism, whereas the entire three-dimensional data set (a set of focal planes) should be considered together by the clustering algorithm to improve data locality on each disk (changing focal planes does not change the area of interest within a plane).

Another issue to consider to both maximize parallelism and data locality is support for multiple users in a single image server. If we call sending a set of microscope parameters to the server, so that the server can return the relevant portion of a slide image, a *query*, then both *intra-query* and *inter-query* parallelism should be supported to optimize performance. Intra-query parallelism minimizes the response time of a single query, and can be achieved in a parallel data server via overlap of disk accesses, computation and interprocessor communication in each processor [5]. Inter-query parallelism maximizes the overall throughput of the parallel data server, by scheduling queries from multiple users to optimize overall resource utilization. We discuss how the system design of the Virtual Microscope addresses these issues later in the paper.

Data Compression

Another technical issue that must be addressed is compression of the stored slide data. As was discussed in the introduction, a single slide can have from five to thirty focal planes, and each focal plane is about 7GB of uncompressed RGB data, for a total of 35-210GB of raw data per slide. Since we expect that a real image database would contain hundreds to thousands of slides, compression must play a very important role in the Virtual Microscope system to reduce data storage requirements. Compression will also reduce the amount of I/O required for the server to respond to a query, but will increase the amount of computation required to produce an image to return to the user.

Many different compression algorithms have been proposed in the literature (e.g. JPEG [8]), but the Virtual Microscope requires both a high degree of compression and a low loss of information. This reduces the search for compression techniques to a small number of methods, and we are using a wavelet compression technique [9].

Besides reducing storage and I/O requirements, the wavelet compression technique has other useful performance characteristics. First, wavelet compression preserves locality in the data, meaning that if a user requests a small contiguous portion of the whole image, the system can compute the corresponding regions in the stored compressed image, and there will only be a small number of contiguous stored regions to retrieve. This property has good implications for the performance of the Virtual Microscope system, since it means that the number of data blocks that must be retrieved from disk is always only proportional to the size of the output image that will be displayed to the user, and not to the size of the entire stored image.

Another significant characteristic of wavelet compression is that it is a multi-resolution compression technique. This means that a compressed image is stored as a sequence of images of varying resolutions, corresponding to different magnifications for the Virtual Microscope. Therefore, the system will be able to directly reconstruct an image at any magnification that is lower than the magnification of the stored image (initially only by powers of two). In addition, the amount of data that must be read from disk is proportional to the magnification of the requested image. For example, reconstructing an entire image at a magnification of 25X from a compressed image stored at a magnification of 200X only requires retrieving $1/64^{th}$ of the stored data (a factor of eight in each of the two dimensions).

System design

The Virtual Microscope is a distributed system consisting of 3 parts: a client/user interface, which allows a

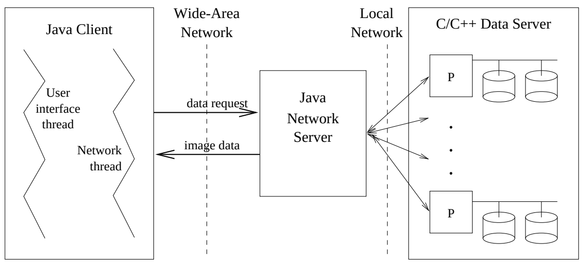

Figure 2: Client/Server Architecture

user to view slides over a network, a parallel data server, which retrieves the slide image data, and a network server, which allows the other two parts to communicate. The modularity of the system allows us to develop and optimize each part somewhat independently and focus on the unique requirements of each piece. So, while the client interface must be user-friendly, web-friendly, and portable, the parallel data server can be finely tuned for a specific parallel machine configuration. A diagram of the system architecture is shown in Figure 2.

Client/User Interface

The Internet-downloadable Java client program shown in Figure 3 provides a graphical user interface so that users can control browsing through slides by dragging and clicking the mouse. The display window for the client process contains three components:

1. the **directory panel** allows users to choose from a collection of different slides (upper right in Figure 3),
2. the **display panel** shows the selected portion of a slide at a selected magnification (left side of Figure 3), and
3. the **control panel** provides the standard operations supported by the Virtual Microscope as described previously (lower right in the Figure 3). The control panel has three sub-components:
   - a *magnification* selection button,
   - a *list* of available focal planes for a selected slide, and
   - a *thumb-nail window* with four directional buttons.

The thumb-nail window in the control panel presents a small, low magnification version of the entire slide (not shown in this image) and provides users with two types of browsing operations. First, users can move the microscope stage by small increments in one of the four directions (i.e., up, down, left, right) by clicking the corresponding directional button. Second, users can locate the interesting portion of a slide rapidly by dragging the mouse on the small box inside the thumb-nail window. The small box

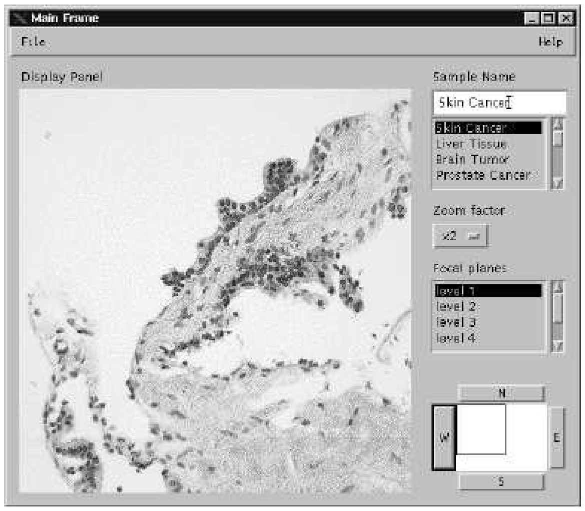

Figure 3: Graphical User Interface for the Client Program

inside the window indicates the current portion of the image shown in the display panel.

Network Server

The network server's primary responsibility is to connect the client, running on a PC or workstation, to the parallel data server, running on the parallel machine. The network server runs on a node or front end of the parallel machine and can communicate directly with the parallel data server. The client must communicate with the network server via a standard network protocol. Currently, the network server is also responsible for stitching together the pieces of the output image that are returned by each of the nodes in the parallel data server, into a single composite image that is returned to the client. For performance reasons, that functionality may eventually migrate into the client so that the client can receive parts of the output image as they are produced by the data server nodes, without waiting for the entire image to be produced by all data server nodes.

Parallel Data Server

The parallel data server is the program responsible for efficiently serving image data. In order to produce an image, the data has to be read from disk and an image of the specified magnification must be reconstructed. In order to serve this image data effectively the server must also perform well, making good use of parallel disk access, as well as caching, prefetching, and compression techniques.

In order to get good performance, data must be read from multiple disks in parallel. The server itself is a distributed program running on each of the nodes of a parallel machine, with each node having several disks. The images stored on the machine are split into blocks, compressed, and distributed across all the disks, so that each node has multiple rectangular portions of the full image. In processing a user request each node can determine which of these portions, or image blocks, are on its local disks. If the system has not cached or prefetched the required blocks, the data is retrieved from disk in parallel.

Queries from a client will frequently involve either image data that has been accessed before or image data that is adjacent to previously accessed image data. Since that is the expected access pattern, and given the performance penalty of retrieving data from disk relative to retrieval from main memory, the Virtual Microscope system must take advantage of prefetching and caching techniques. For the parallel data server this means that every query does not necessarily require a full retrieval from disk. For a single query, prefetching allows the server to retrieve portions of the image surrounding the query before they are requested by the client, under the assumption that nearby parts of the image are likely to be requested in the near future. For multiple queries, caching allows the server to reuse data retrieved from disk to service multiple clients, without multiple disk accesses. Several caching and prefetching strategies are being studied, to determine performance effects under various user scenarios.

As data from disks becomes available in memory, further processing is required to produce an image at the magnification level required by the client. Since image blocks are stored in a a compressed format, the processing mostly consists of decompression. The wavelet transformation technique used to compress the blocks allows direct reconstruction of the images at the required magnification in the decompression step, eliminating the need to store or compute the image at multiple magnifications. While decompression is the most expensive computational task for the parallel data server, the the server is able to take advantage of asynchronous I/O requests and overlap the computation (block decompression) with I/O for other blocks.

Current status and future plans

We currently have a prototype system running. The prototype uses separate client and network server processes, and a single data server process, all running on a single workstation processor. Extending the prototype to multiple data server processes, each running on separates nodes in a parallel computer, is straightforward, but will require significant tuning for high performance. However, we are confident that the parallel data server will provide the required performance, because of our previous experience with the Titan satellite image database system. Titan also employs a parallel data server, and proved capable of providing high performance. For example, a global ten-day composite query into a 24GB sensor database can be performed by Titan in under 90 seconds, moving about 1.7GB of data from secondary storage. The main bottleneck in current Titan performance is the computational part of the system, to generate the composite image from the retrieved raw sensor data. The computational part of the Virtual Microscope system occurs mainly in decompressing the stored image blocks, and should be significantly less expensive than the computation in Titan.

Future projects that use the Virtual Microscope infrastructure will involve integration of additional sensor modalities along with associative retrieval of sensor data obtained from related cases. The sensor modalities that would be involved in a sensor data fusion effort include

- different radiological imaging modalities such as CT, MRI and PET,
- electron microscopy,
- confocal microscopy, and
- conventional high power light microscopy.

References

[1] Ronald S. Weinstein, A.K. Bhattacharyya, Anna. R. Graham, and John R. Davis. Telepathology: A ten-year progress report. *Human Pathology*, 28(1):1–7, January 1997.

[2] Michael Weinstein and Jonathan I. Epstein. Telepathology diagnosis of prostate needle biopsies. *Human Pathology*, 28(1):22–29, January 1997.

[3] S. Olsson and C. Busch. A national telepathology trial in Sweden: Feasibility and assessment. *Arch Anat Cytol Pathol*, 43:234–241, 1995.

[4] Ken Arnold and James Gosling. *The Java Programming Language*. The Java Series. Addison-Wesley, 1996.

[5] Chialin Chang, Bongki Moon, Anurag Acharya, Carter Shock, Alan Sussman, and Joel Saltz. Titan: A high performance remote-sensing database. In *Proceedings of the 1997 International Conference on Data Engineering*. IEEE Computer Society Press, April 1997. To appear.

[6] Bongki Moon and Joel H. Saltz. Scalability analysis of declustering methods for multidimensional range queries. *IEEE Transactions on Knowledge and Data Engineering*, 1997. To appear.

[7] Bongki Moon, H.V. Jagadish, Christos Faloutsos, and Joel H. Saltz. Analysis of the clustering properties of Hilbert space-filling curve. Technical Report CS-TR-3611 and UMIACS-TR-96-20, University of Maryland, Department of Computer Science and UMIACS, March 1996. Submitted to IEEE Transactions on Knowledge and Data Engineering, March 1996.

[8] William B. Pennebaker and Joan L. Mitchell. *JPEG Still Data Compression Standard*. Van Nostrand Reinhold, 1993.

[9] John J. Benedetto and Michael W. Frazier. *Wavelets: Mathematics and Applications*. Studies in Advanced Mathematics. CRC Press, 1994.