Memory Management in Interactive Medical Teleconsultation Systems

Filip Malawski
AGH University of Science and Technology
Faculty of Computer Science, Electronics and Telecommunication
Department of Computer Science
al. Mickiewicza 30, 30-059 Krakow, Poland
e-mail: fmal@agh.edu.pl

Medical teleconsultation systems are an important tool in modern medicine, as they enable convenient, real-time collaboration of doctors from remote hospitals, without the need to travel. Due to the large size, efficient handling of medical imaging data in real-time teleconsultations is challenging. Memory management provided by operating systems is based on the virtual memory, which introduces considerable delays. In this paper we propose dedicated methods for memory management, including both memory monitoring and pre-loading of images. Results of the conducted experiments indicate, that the proposed methods can significantly improve the user experience.

Keywords: telemedicine, teleconsultation, medical imaging, DICOM, memory management

Introduction

Medical imaging is one of the most important diagnostic tools in modern medicine, as it allows for non-invasive examination of the interior of the patient's body. Equipment for performing ultrasonography (US), computed tomography (CT) and magnetic resonance imaging (MRI) is becoming more and more common in hospitals. On the other hand, there is a shortage of radiologists and doctors who are able to analyze difficult cases of medical images and properly diagnose the patient. Lack of medical specialists is particularly problematic in remote hospitals in small cities. These hospitals often need to relay on cooperation with larger medical facilities. Dedicated telemedical systems allow for remote consultations in order to facilitate this cooperation, despite the distance between the hospitals.

Medical teleconsultations can be performed in two scenarios – asynchronous and synchronous. In the asynchronous scenario a doctor sends examination data to the consulting hospital, where it awaits to be handled by one of the experts [1], [2]. Depending on the availability of the experts and the severity of the case, required time varies usually between a few hours and a few days. This model of teleconsultations is also called store-and-forward [3]. In the synchronous scenario, both the requesting and the consulting doctor meet online in order to discuss the given case [4]. Often not only two, but a few doctors collaborate simultaneously. Tools for videoconferencing and shared views are useful in this scenario. Advanced telemedical systems are interactive, thus allowing not only to synchronize views between remote machines, but also to synchronize actions [5], [6]. Each participant can apply transformations to the imaging data or make measurements, and all of these actions are propagated to all other participants in real-time. Such systems are useful: 1) in critical cases, when quick consultation is required, 2) in difficult cases, when a group of doctors discuss possible treatment options, 3) for learning, allowing a group of students to interact together with a teacher. Proper implementation of interactive telemedical systems poses a number of technical challenges [7], one of them being management of the medical data.

Medical data management

Medical imaging data often requires vast amounts of space. A single CT or MRI examination can occupy hundreds of megabytes. During an interactive teleconsultation it is often the case, that multiple examinations of the discussed patient are provided and all of them need to be analyzed. Such a huge amount of data is problematic in terms of transfer, processing and displaying.

Transfer and processing

In hospitals bandwidth available for teleconsultation systems is often very limited and not sufficient for real-time transfer of medical images. Therefore, the data are usually transferred prior to the consultation start, so that all of the participants have all the data available during the consultation. The data is often transferred in a compressed format in order to reduce the transfer time.

When analyzing a medical image, doctors need to apply various transformations, such as zooming or changing the contrast or the brightness. In the case of large images, these operations may be time-consuming, which in turn can negatively affect the real-time communication. In order to address this issue, image transformations are often performed using the GPU rather than the CPU, as it allows for efficient parallel processing of the images.

Displaying

Although displaying the images may seem to be an easy enough task, when huge amounts of large images are considered, it becomes a complicated issue. CT and MRI data represent usually 3D images, which are stored as sequences of 2D images. US data often include a sequence of images taken in consecutive time moments. Therefore, when a doctor loads an examination to be displayed, it requires loading multiple (often around a hundred or more) images, which will be visible as a sequence. Such sequence can occupy as much as a few hundred megabytes of memory. Thus, loading an image sequence from the hard drive often takes a considerable amount of time. Depending on the hardware, available resources and the sequence size, the user may need to wait even more than 30 seconds (see also the Results section). A convenient solution would be to load all of the imaging data to the internal memory in the beginning of the consultation. However, available internal memory is usually not sufficient to store all the data.

Loading too much data to the internal memory can actually result in a significant increase of time required to display an image sequence. Once the operating system (OS) notices that there is not enough internal memory, it starts to employ the virtual memory, which is seen as the internal memory, but actually the data are stored on the hard drive. Moving the data from the internal memory to the hard drive is very slow, therefore it introduces significant delays. Things can become even worse, when we consider the following scenario: the requested image sequence is already moved to the virtual memory; it needs to be moved to the internal memory in order to be displayed, but there is not enough free internal memory; the OS needs to move some other data from the internal memory to the hard drive and only then can it move the requested image sequence from the hard drive to the internal memory. In such cases, delay introduced by writing to and reading from the hard drive is often unacceptable for interactive teleconsultations. It is worth mentioning, that the quality of collaboration depends on the machines of all of the participants, since all the participants need to wait for the slowest machine to load the image sequence locally, before the discussion may begin.

In order to address these issues we propose a dedicated memory management scheme, which allows to minimize the delay when displaying the images during the consultations.

Methods

In this section we first discuss methods for storing the medical images and associated metadata. Then, we propose mechanisms for effective memory management.

Medical images and annotations

Medical imaging data are usually saved in the DICOM [8] format. It can store not only the image data but also metadata including: 1) patient and examination details 2) annotations, such as regions of interest or measurements. Usually, the DICOM files are compressed, which increases time needed to access both the imaging data and the metadata. In order to present the user with the list of available medical images, we need to load the metadata and a thumbnail of at least one frame, for all the image sequences in the dataset, at the very beginning of the consultation. Therefore, for the sake of efficiency, we store both metadata and thumbnails in a separate file, which is created prior to the consultation start.

The imaging data may also be stored in an uncompressed format in a separate file, in order to reduce loading time by avoiding the decompression process, which may be time-consuming, particularly in the case of large files. On the other hand, this may significantly increase the space required on the hard drive, therefore it is not always desired. Another approach is to store the imaging data in a separate file and employ dedicated compression algorithms, which allow for real-time decompression on the GPU [9]. In such a scenario the image is kept in a compressed format on the hard drive and in the internal memory, and decompressed only when being displayed. This approach is commonly used for handling textures in video games. Transforming the image data extracted from the DICOM file to the custom compressed format and saving it in a separate file can be performed offline, before the consultation starts. Fig. 1 presents how image data, metadata and thumbnails are stored.

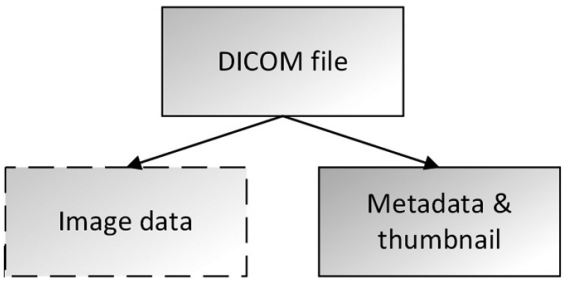

Figure 1. Medical imaging data storage. A DICOM file stores both the image data and the metadata. The metadata is saved in a separate file together with a thumbnail. The image data can be read directly from the DICOM file or optionally stored in a separate file

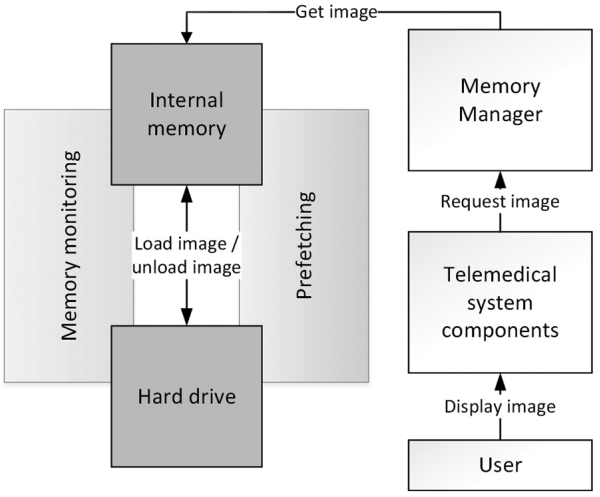

Figure 2. Architecture of the proposed memory management system. When the user wants to display an image the proper component of the telemedical system requests the image from the Memory Manager, which controls image loading process by employing the memory monitoring and prefetching services

Memory Manager

In order to provide proper user experience in the teleconsultations, we need to minimize the delays introduced when loading the medical images. Pre-loading of images and avoiding using virtual memory are the key factors in this case. Memory management is a very dynamic process, therefore, effective algorithms are needed.

In order to facilitate the memory management, in our system the access to the imaging data is allowed only through a dedicated interface, namely the Memory Manager. This solution provides control over the process of loading and unloading the images, and enables employing mechanisms such as prefetching and memory monitoring, which are described in the next subsections.

It also provides transparency for other components of the system, which need only to request the imaging data, not knowing exactly how these data will be loaded. The components requiring the imaging data are to assume that the requested data will be delivered in the shortest possible time. Architecture of the proposed memory management system is presented in Fig. 2.

Memory monitoring

In order to avoid using the virtual memory, it is crucial to always ensure enough free space in the internal memory, before loading an image from the hard drive. We meet this requirement by employing a memory monitoring service, with a dedicated thread, which constantly monitors the internal memory and applies both proactive and reactive actions. Proactive actions are taken by unloading some of the images, when the amount of the free memory falls below a given threshold. Only the images which are not currently displayed can be removed from the memory. The unloading algorithm sorts the images according to the time of the last access and starts with the oldest ones. It unloads images as long as the free memory is below the threshold, then it returns to the monitoring. Reactive actions are taken by unloading images, when there is not enough space in the internal memory to load the currently requested image. Selection of the images that should be unloaded is the same as in the case of proactive actions. Both proactive and reactive unloading actions are complementary – the proactive one tries to keep enough free memory at all time and the reactive one is used when the proactive actions are not sufficient.

It is also worth mentioning, that unloading the images is done by marking them as a weak reference. Object stored as a weak reference remains available as long as it is not removed by the Garbage Collector (GC). Therefore, should the user decide to load again one of the previously displayed images, it may be still available and no loading would be required. On the other hand, when a new image is loaded, the GC can quickly provide required space.

Prefetching

Although memory monitoring provides efficient loading of images by avoiding using the virtual memory, large image sequences still need a considerable time to load. In order to address this issue, we propose the prefetching service. It predicts which image will be displayed next and loads it to the internal memory, before it is requested. If the predictions are correct, the user will notice practically no delay when displaying the images. The predictions are based on the order of images in the consultation dataset. Since the doctors preparing the consultation datasets put the images in the order that they want to discuss them, this is quite an effective strategy. Naturally, in some cases, predictions will not be correct, e.g. should the doctor request again a previously displayed image. However, such situations are rather rare and should be handled by the memory monitoring service.

Results

In order to verify the proposed methods, we performed the following experiments. Using publicly available medical imaging data [10] we created a dataset consisting of eight image sequences (six CT and two MR examinations), which were loaded in a given order. Computer with 7200 RPM HDD, 3 GB RAM and 2 GHz CPU was employed. Software components from an existing medical teleconsultation system [5] were used for DICOM image loading. We assumed that at least 20 seconds are needed for a doctor to analyze and/or comment on a loaded image sequence during a teleconsultation, therefore the next image sequence was always loaded 20 seconds after the previous one has finished loading. The sizes of the eight image sequences

in the dataset are given in Table 1. Total size of the dataset was 1287 MB.

In the experiments we considered 4 scenarios:

1. Available internal memory is about 2 GB, which is sufficient to load whole dataset without unloading any images or moving them to the virtual memory.
2. Available internal memory is about 0,7 GB, which is sufficient for little more than half of the dataset. In this scenario memory management service was inactive, therefore the OS was expected to use the virtual memory.
3. Available internal memory is about 0,7 GB. In this scenario memory monitoring was active and prefetching was inactive.
4. Available internal memory is about 0,7 GB. In this scenario both memory monitoring and prefetching were active.

Each scenario was run three times. Results of the experiments for scenarios 1, 2 and 3 are presented in Figure 3. Results for scenario 4 are not presented graphically, but discussed further in this section.

In Fig. 3 we can observe, that in the case of scenario 1 (blue/left bars) loading times are similar in all three repetitions. In the case of scenario 2 (red/middle bars) the first two image sequences are loaded in a similar time as in the previous scenario, but starting from the third image sequence we can observe peaks, which indicate that the OS needed to move some data to the virtual memory in order to clear the internal memory. This can result in over two times longer loading, as in the case of image sequences number 3 and 6. In scenario 3 (green/right bars) there are no peaks, therefore we can conclude that the memory monitoring service successfully avoids using the virtual memory.

Table 1. Sizes of image sequences used in the experiments

Image seq. nb.	1	2	3	4	5	6	7	8
Size [MB]	217	173	245	56	122	197	180	56

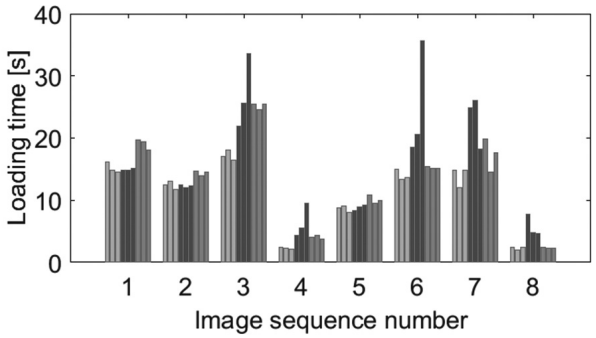

Figure 3. Loading times of the eight image sequences from the dataset used in the experiments. Blue (left) bars: scenario 1, red (middle) bars: scenario 2, green (right) bars: scenario 3

On the other hand, loading time is often slightly longer as compared to the scenario 1, due to the memory monitoring service removing old images from the memory. Nevertheless, such overhead is preferred to the peaks occurring in scenario 2.

In scenario 4 all loading times were close to zero (therefore it is not presented in Fig. 3), as the images were preloaded by the prefetching service. The service was set to load the first two images before the start of the experiment and then each time when an image was displayed, the service started pre-loading the next image. Due to the memory monitoring service, old images were removed and the virtual memory was not used.

Summarizing, using both the prefetching and memory monitoring services can reduce the loading times almost to zero. Even though in some rare cases, when the images are requested in a different order, the loading times may be higher, the proposed methods can greatly improve an overall user-experience in medical teleconsultations.

Conclusions

In this paper we proposed methods for memory management in medical teleconsultations. We analyzed issues related to loading extensive medical data in real-time and remote collaboration of doctors. We designed and implemented a dedicated memory management system, which includes the memory monitoring and prefetching services. The memory monitoring service provides sufficient free internal memory for loading images by employing both proactive and reactive actions. The prefetching service preloads images based on their order in the consultation dataset. Both services are complementary and together can provide efficient memory management. Based on the conducted experiments, the proposed memory management system can provide the participants of the interactive medical teleconsultations with good user experience regarding the loading of the imaging data.

Acknowledgement

This work was supported by AGH University of Science and Technology, Faculty of Informatics, Electronics and Telecommunications statutory project. We would like to express our thanks to Łukasz Czekierda, PhD, for his valuable feedback.

Bibliography

[1] J. H. Gennari, C. Weng, J. Benedetti, and D. W. McDonald, "Asynchronous communication among clinical researchers: a study for systems design.," *Int. J. Med. Inform.*, vol. 74, no. 10, pp. 797–807, Oct. 2005.

[2] G. Luccichenti, F. Cademartiri, a Pichiecchio, E. Bontempi, U. Sabatini, and S. Bastianello, "User interface of a teleradiology system for the MR assessment of multiple sclerosis.," *J. Digit. Imaging*, vol. 23, no. 5, pp. 632–8, Oct. 2010.

[3] N. Lasierra, a Alesanco, Y. Gilaberte, R. Magallón, and J. García, "Lessons learned after a three-year store and forward teledermatology experience using internet: Strengths and limitations.," *Int. J. Med. Inform.*, vol. 81, no. 5, pp. 332–43, May 2012.
[4] Y.-C. Hsu, P.-Y. Lin, S. J. Hsu, and C.-T. Chan, "Design and Implementation of Teleconsultation System for Instant Treatment," *2008 2nd Int. Conf. Bioinforma. Biomed. Eng.*, pp. 1355–1358, May 2008.
[5] Ł. Czekierda, F. Malawski, and P. Wyszkowski, "Holistic approach to design and implementation of a medical teleconsultation workspace," *J. Biomed. Inform.*, vol. 57, pp. 225–244, 2015.
[6] J.-S. Lee, C.-T. Tsai, C.-H. Pen, and H.-C. Lu, "A real time collaboration system for teleradiology consultation," *Int. J. Med. Inform.*, vol. 72, no. 1–3, pp. 73–79, Dec. 2003.
[7] Ł. Czekierda, T. Masternak, and K. Zieliński, "Evolutionary approach to development of collaborative teleconsultation system for imaging medicine.," *IEEE Trans. Inf. Technol. Biomed.*, vol. 16, no. 4, pp. 550–60, Jul. 2012.
[8] NEMA, "DICOM standard." [Online]. Available: http://dicom.nema.org/standard.html. [Accessed: 09-May-2017].
[9] F. Malawski and L. Czekierda, "Compression of Image Sequences in Interactive Medical Teleconsultations," *Comput. Sci.*, vol. 18, no. 1, pp. 95–114, 2017.
[10] OsiriX, "DICOM sample image sets." [Online]. Available: http://www.osirix-viewer.com/datasets/.

Author: MSc. Eng. Filip Malawski *- is a PhD candidate at AGH University of Science and Technology in the Department of Computer Science. He received his MSc degree in computer science at AGH University in 2012. His research interests include telemedicine, computer vision, image processing and human-computer interaction.*