Research on Biomedical Engineering
rbejournal.org
Volume 32, Number 4, p. 347-357, 2016

Original Article

DOI: http://dx.doi.org/10.1590/2446-4740.05915

Efficient medical image access in diagnostic environments with limited resources

José Eduardo Venson*, Fernando Bevilacqua, Fabio Onuki, Marcos Cordeiro d'Ornellas, Anderson Maciel

Abstract **Introduction:** A medical application running outside the workstation environment has to deal with several constraints, such as reduced available memory and low network bandwidth. The aim of this paper is to present an approach to optimize the data flow for fast image transfer and visualization on mobile devices and remote stationary devices. **Methods:** We use a combination of client- and server-side procedures to reduce the amount of information transferred by the application. Our approach was implemented on top of a commercial PACS and evaluated through user experiments with specialists in typical diagnosis tasks. The quality of the system outcome was measured in relation to the accumulated amount of network data transference and the amount of memory used in the host device. Besides, the system's quality of use (usability) was measured through participants' feedback. **Results:** Contrarily to previous approaches, ours keeps the application within the memory constraints, minimizing data transferring whenever possible, allowing the application to run on a variety of devices. Moreover, it does that without sacrificing the user experience. Experimental data point that over 90% of the users did not notice any delays or degraded image quality, and when they did, they did not impact on the clinical decisions. **Conclusion:** The combined activities and orchestration of our methods allow the image viewer to run on resource-constrained environments, such as those with low network bandwidth or little available memory. These results demonstrate the ability to explore the use of mobile devices as a support tool in the medical workflow.

Keywords: mHealth, Teleradiology, Radiology, Mobile viewer.

Introduction

Mobile technology has contributed to the management of chronic diseases, alerting people on medication schedule and improving the efficiency of health systems (West, 2012). A wider access through mobile devices created new possibilities for diagnosis, meaning less time between the exam execution and the medical report, which improves the diagnosis workflow. In this paper, we address two important problems that still prevent the global dissemination of mobile tools for diagnosis based on radiologic images. The first is the limited computational resources of mobile devices (e.g. network communication, computational power), and the second is the low friendliness of the user interface due to limited peripherals and underdeveloped specific design methodologies.

The main contribution of this paper, thus, is the introduction of an approach that uses a combination of client- and server-side procedures to dynamically optimize the data flow for fast image transfer and visualization on mobile devices. The main advantage of our approach is to minimize the amount of data transferred to and used in the host device without sacrificing the user experience. Despite resource constraints in the hosting device, e.g. limited memory/processing power in smartphones, the user should still be able to perform computationally complex actions, such as image windowing.

Another contribution is the implementation of our approach in a real PACS system, the mobile application Animati Viewer (part of the Animati PACS - a system for distribution and image diagnostics developed by Animati - *animati.com.br*), and the conduction of an experimental evaluation with real users to assess system performance and user experience. A native viewer would require a different system to be implemented to each platform and would require the user to download and install different apps. We chose to focus on web-based PACS to provide rapid and reliable access to medical data according to Web DICOM standard (Koutelakis and Lymperopoulos, 2006). The remainder of this paper briefly reviews the previous art, defines requirements for diagnosis tools, describes our design assumptions and algorithms, characterizes the experimental setups and extensively discusses the results. Interesting findings regarding the typical behavioral pattern of radiologists when navigating through the image datasets are also presented in this paper for the first time.

*e-mail: jevenson@inf.ufrgs.br
Received: 31 December 2015 / Accepted: 19 December 2016

Related work

There are several initiatives to make the diagnostic procedures available beyond the workstation environment. In order to identify such initiatives, we conducted a systematic review using the strings: "DICOM viewer", "Mobile radiology" and "Mobile diagnostic" in the Google Scholar search engine, and then selecting works with viewable results on desktop and/or mobile devices.

A more complete and widely used mobile application is the OsiriX mobile, whose implementation is described by Choudhri and Radvany (2011). The application has tools for viewing and processing DICOM images, allowing users to perform several operations such as zoom and rotation. All images are downloaded to the application as uncompressed DICOM files, and then processed locally for all subsequent actions. This architecture affects the access time on networks with low bandwidth. As a result, the application does not allow the transfer of images larger than 1024x1024. In addition, the application is only available for iOS, limiting the range of access devices. Correa et al. (2008) presents an approach to extend the access of medical images on mobile devices. They developed a distributed system composed of a web server, responsible for processing computer aided diagnostics (CAD) algorithms on the images, and a mobile device client, which is able to visualize the results of such process. The combination of both ends provides physicians with a solution containing extra information regarding the diagnosis. The system and user performance requirements for CAD, however, are not the same as for a visualization tool. Similarly, Pasha et al. (2012) explore the use of mobile and stationary devices as a platform for collaborative discussion of medical images among hospital personnel. The patient images are uploaded from the workstation to a web server, which makes them available to mobile devices in the network. Using mobile applications that access the web server, the doctor can view DICOM images and collaboratively discuss the diagnosis with colleagues. The problem of limited memory available in mobile devices is mitigated by compressing large DICOM images to JPEG before sending them to the portable devices. Usability is not discussed. Kaserer (2013) presents a DICOM web viewer able to display uncompressed DICOM files. Even though the viewer is able to run on desktop and on mobile platforms, it has some limitations, such as no integration with PACS systems and the limitation to display only uncompressed DICOM images. There is no performance evaluation.

Our approach, instead, focus on delivering a visualization solution that works outside the workstation environment and is consistent on a variety of platforms, such as desktops and low- to high-end mobile devices. Our approach constantly uses compressed images in order to minimize data transferring, as opposed to some of the previous works. We also introduce a novel approach that uses a buffer to download a set of images on-demand, favoring JPEG compressed files instead of uncompressed DICOM images whenever possible to overcome network bandwidth limitations.

Mobile application for medical diagnosis

Several technical requirements have to be met in order to make a mobile application for assisted diagnosis and for interpretation of medical images. Compared to the environment of a workstation, a mobile application must deal with several constraints such as limited processing power, fewer memory and lower network bandwidth.

Despite such adversities, mobile diagnostics is a valid and trending practice. As pointed out by Chandratilleke and Honeybul (2013), the use of mobile tools can reduce the time gap between the image acquisition and the moment it is viewed by the medical team, decreasing the time a patient has to wait for a diagnosis. De Maio et al. (2014) studied the accuracy of mobile diagnostics related to intra-articular knee pathology, concluding that an iPhone DICOM Viewer can be used and it is similar to that of a conventional radiology workstation. Bhatia et al. (2013) also presented that mobile devices such as the Apple iPad can display adequate resolution of CT (computed tomography) and MRI (magnetic resonance imaging) sequences to accurately diagnose acute central nervous system injuries and other non-acute pathologies. John et al. (2012) suggest that the emergency conditions commonly encountered in CT and MRI can be diagnosed using a portable device DICOM viewer with good concordance with the workstation evaluation. Choudhri et al. (2012) present similar results regarding the analysis of acute appendicitis on abdominal CT studies using a mobile device. The study made by Zwart et al. (2015) has explored the technical requirements on a mobile device for faster access to patient data than on conventional PACS. After comparing 565 image-viewing events, results showed identical ease of use and similar diagnosis confidence to desktop devices. However, the mean time to first image has been significantly shorter on mobile viewer. In order to create a tool that allows the interpretation of medical images on the mobile devices, one has to structure the application and its workflow in such a way to minimize the environment constraints. Compressed images can be used to reduce memory consumption and the download

time. The hardware differences among mobile devices and the limited bandwidth, e.g. poor connection in a mobile network, require the application to adapt to the available computing resources, balancing tasks and strategies in favour of usability and/or performance. The next section presents our approach to make a mobile diagnostic application able to overcome the previously mentioned adversities, which makes it capable of running on different scenarios and devices.

Methods

System design and implementation

Our approach to creating a mobile DICOM viewer aims to reduce to a minimum the amount of information downloaded and processed by the application. This goal is achieved by orchestrating two modules: the on-demand download manager (ODM) and the dynamic windowing module (DWM). Both modules use compression and cache strategies, as well as a combination of client- and server-side processing to reduce the amount of transferred data. This study was approved by the Institutional Review Board of the university (1.782.728).

The ODM is the means to efficiently visualize the images in the series. It is controlled by the image selector, a slider widget represented by the red circle at the bottom of Figures 1a and b. As the user slides the image selector over the sliding area, a specific image of the series is selected. Assuming the image selector is at position P and the series has 100 images, for instance, P can vary from 0 (far left side of the sliding area), which displays the first image of the series, to 99 (far right side), which displays the last image of the series. This approach allows the user to navigate back and forth through the images of a series. The currently active position/image P is called the pivot and it is used by the ODM to control the flow to fetch data from the server, deciding how many and which images should be downloaded. The DWM, on the other hand, allows the user to perform windowing operations on the currently selected image. The DWM decides if this operation is best executed locally (client-side) or remotely (server-side). The following subsections describe in detail how these two modules work.

On-demand download manager

The on-demand download manager (ODM) acts as a new level in the memory hierarchy. When the user opens a series, ODM fetches the corresponding images of that series from the server. In order to save on data transferring and comply with any memory constraints, the ODM module will fetch only a subset of the images from the server, storing them in a local buffer. As soon as the first image is downloaded by the ODM, the user can navigate the series by sliding the image selector (pivot), as previously explained. The subset of images downloaded into the ODM buffer will always be centered at the position of the pivot. For instance, assuming the device is able to store 5 images in the ODM buffer and the pivot P is at position 5 (P=5), the buffer will contain the images {3, 4, 5, 6, 7}, in that order. For every change in the position of the pivot, the ODM will calculate and decide new fetches. Using the previous example, if the pivot suddenly moves to position 7 (P=7), the ODM will re-centralize the buffer around the pivot, fetching more images from the server (i.e., images 8 and 9), placing them to the right of the pivot; the ODM will also discard the images to the left (i.e., 3 and 4) to make room for the upcoming images

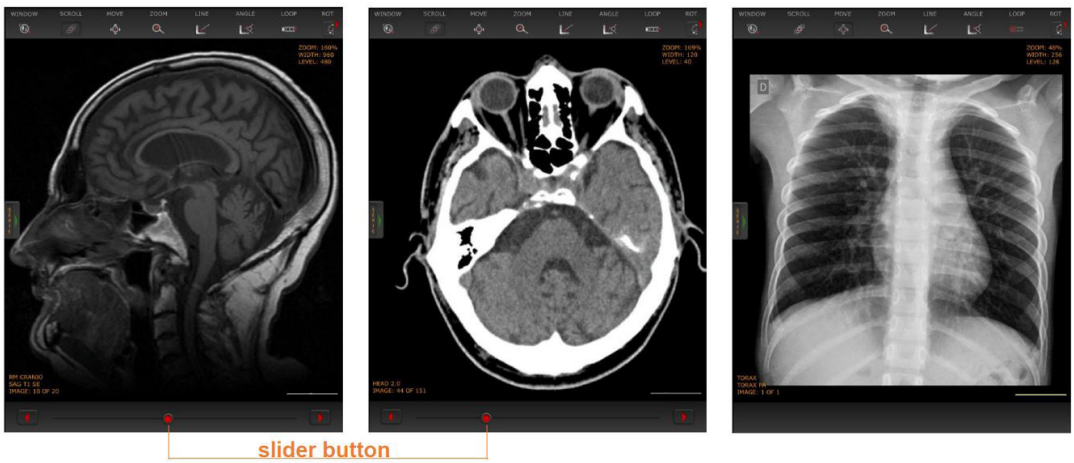

Figure 1. Examples of the application interface for 3 exams: MRI (left), CT (center) and conventional radiography (right).

being downloaded. The resulting buffer will contain the images {5, 6, 7, 8, 9}. The ODM will always try to make the pivot the central image in the buffer, which means it will fetch images to the left and to the right of the pivot until the buffer is full. As a consequence, the user will be able to navigate back and forth with the image selector while the ODM controls the download process in the background. The centralization approach of the ODM was designed based on a set of empirical viewing cases provided by medical specialists, presented by Venson et al. (2015). The size of the ODM buffer defines how many images are stored in the device at any given time. Such size depends on the memory constraints of the hosting device. According to Wang et al. (2011) the size of a web-browser cache is about 300MB for desktops. This is several times larger than the 6MB available in the Android Gingerbread browser cache, for instance. For that reason, the ODM automatically adjusts its buffer size to meet the memory constraints of the host system. As an additional optimization, all images downloaded by the ODM are JPEG files compressed with quality 75, which minimizes the amount of memory space required by the ODM to operate. The impact of this decision was measured through the user's perception of the images quality. This optimization was used to ensure that the viewer application would meet the memory constraints of the system it is running on. A CT exam, for instance, typically contains 500 images in a study, which accounts for 250MB of data (Pianykh, 2009). Without the ODM and its subset/centralization strategy, the viewer would have to download all images of the study, which could be impractical on mobile devices, or download each image on demand, which will cause unbearable waiting times. The imposed size limit of the ODM buffer, however, allows an estimation of the amount of memory that will be consumed by the application. It can be calculated by averaging the size of each image in a study and multiplying that value by the size of the ODM subset. In an empirical test, we measured the memory space required to analyze a CT study containing 3760 slices/images (JPEG compressed). If all images were downloaded and stored in memory, a maximum amount of 134.5MB would be used in the host device. Using the ODM and its strategy of downloading a subset of images at any given time, a maximum amount of 7.5MB of memory was used instead. The amount, in this case, is about 5% of the total size of the study, which can be estimated and adjusted according to the available resources.

Dynamic windowing module

A windowing action can be seen as a contrast adjustment performed on the images of a study being analyzed. Actually, as medical images typically present a very high dynamic range, windowing is necessary to optimize visual detail for a specific tissue in the images, e.g. bones or muscles (Bourne, 2010). This is done by mapping a range of pixel values to a grayscale ramp. The dynamic windowing module (DWM) conveys a new windowing approach for more efficient use of this functionality. The module was designed to intermediate windowing operations, aiming at minimizing the amount of information downloaded, without sacrificing the user experience. The module will work alongside the ODM module, performing its actions based on three main variables: waiting time between windowing actions, size of the uncompressed version of the images and available memory in the device. The DWM allows the user to input windowing parameters: window level (wl) and window width (ww), which are performed and previewed in the currently active image of the series on the screen. The user is required to apply or discard any ongoing windowing input before switching to another image in the series (image selector sliding area is disabled when the windowing panel is active). As the user inputs windowing values, the DWM will constantly update the currently active image preview to present the new window selected by the user, so the user can input values until the desired windowing configuration is achieved. The update process can be performed locally (on the client's browser) or remotely (on the server). The DWM will choose between the local and the remote windowing approach based on the size of the uncompressed (raw) version of the image being windowed. If the raw image fits the memory constraints, the DWM will choose a local approach, using the raw image to calculate the adjustments, which allows the application to instantly render all updates as they are input by the user. In that case, after the raw image is downloaded, all windowing computation is performed locally and the result is instantly rendered. A CT study, for instance, has an average size of 512kB for each raw image (Pianykh, 2009), so the DWM will perform all windowing adjustments locally for CT studies. Even though the user has to wait for the download of the raw image before inputting windowing values, any adjustment made after the raw file is downloaded will be instantly rendered by the DWM. It avoids network traffic while the windowing is in place and allows the user to slide a finger (or the mouse cursor when available) to adjust window dynamically, performing hundreds of windowing operations with no network cost. If the size

of the raw version does not fit the memory constraints, the DWM will choose the remote windowing. In that case, the DWM transmits to the server every windowing value the user inputs. All windowing operations are performed on the server-side. When the server finishes the current operation, the DWM downloads the result as a JPEG compressed image and presents it to the user. The process is repeated while the user modifies the windowing parameters. Typically, the DWM uses remote windowing for mammographies, whose size of a single raw image is about 58MB (Pianykh, 2009), and conventional radiography, which are also large. After the windowing process is concluded, the user is allowed to navigate with the image selector and use any other visualization tools again. From that moment on, the new windowing configuration is stored and applied to all subsequent images fetched from the server. The user is then able to analyze the study using the selected window. Every time a windowing parameter changes, the DWM has to inform that to the ODM, causing it to purge its buffer and re-download new JPEG compressed images to match the newly defined windowing configuration. The user may need to input several windowing values before the desired configuration is achieved. So, the DWM was designed to trigger the ODM re-download only after the user is done inputting values, not after every interaction. Since the user is not able to interact with the rest of the application when the windowing panel is in place, the DWM can precisely decide when the windowing operation is over, triggering the ODM re-fetch only when it is useful. The re-fetch is expensive because the ODM will remove from its buffer all previously downloaded images with different windowing configuration from the one the user just input and re-fetch them using the newly input windowing parameters.

We conducted a simple experiment to compare the accumulated amount of transferred data between the local and the remote windowing approaches during the input and preview of windowing parameters. Our test assumes a study with 512x512 images with 512kB average size, common in CT and MRI, where every windowing input (changing wl and/or ww) is followed by a preview of the currently defined configuration. As illustrated in Figure 2, the accumulated amount of data transferred by the local windowing approach is constant after the raw image is downloaded. The DWM downloads such raw file (about 512kB) once, and uses it to locally perform further inputs and previews; no network traffic is required after the raw image is downloaded. The remote approach, on the other hand, requires the DWM to download a new (JPEG compressed) image for each preview of a windowing input. Size of the images in the remote approach varies, but it is generally smaller than the full-uncompressed image (5% of the full image size in average).

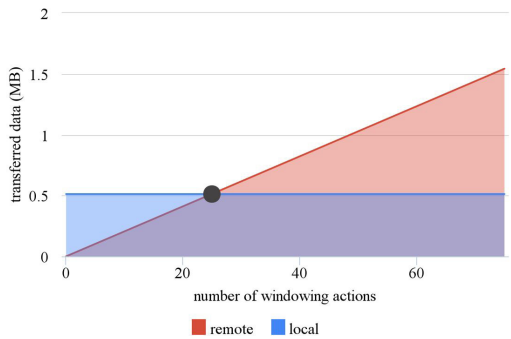

Figure 2. Accumulated amount of data transferred during successive windowing operations with a single CT slice. Notice that the amount of data transferred using the local approach is constant, while using the remote approach it grows linear with the number of inputs. Interactions requiring more than 24 windowing inputs are more efficient when using the local approach.

After 72 inputs/previews of windowing parameters, the remote (server-side) windowing approach caused the application to download an amount of 1.5MB. As previously mentioned, the remote approach requires the download of a new JPEG image from the server for each windowing preview, which produces a linear growth in the amount of transferred data. The local approach (client-side), however, requires the download of a single file, which is the uncompressed version of the image being windowed.

As illustrated in Figure 2, after the download of the uncompressed DICOM file, the accumulated amount of data transferred remains the same for all subsequent windowing inputs. If up to 24 windowing inputs/previews are performed, as indicated by intersection black point in Figure 2, the remote approach transfers less data compared to the local approach. After 24 inputs/preview, the savings achieved by the local approach becomes evident as the accumulated amount of data transferred remains constants regardless of the number of further inputs/previews.

Experimental setup

We designed and carried out experiments with real users to evaluate the performance of the proposed techniques. The main goal was to measure the effectiveness of our solution as a support tool for medical workflow. Part of this research was conducted at the premises of a partner teleradiology company. The company receives exams from dozens of clinics located in different towns spread on a radius over 500 km, causing large variance in the equipment and protocols used for image acquisition. This indicates

that any results we shall obtain derive from real world data, which accounts for their validity in a general scenario. Seven radiologists participated as volunteer users in the experiment (57.1% aged 31 to 35 years, 28% aged 36 to 40 and 14.3% aged 41 to 50). Six of them declared to be experienced radiologists. They have several subspecialties, e.g. neuroradiology, musculoskeletal, abdominal and thoracic. They are active professionals with appointments also in other clinics besides our partner's and hospitals, having large experience with PACS. For our experiment, these doctors performed a total of 40 image analysis events without clinical purposes. Firstly, all participants signed a legal term of consent informing about the compliance with ethical precepts. Afterwards, they filled a characterization form. Then, they were invited to perform a diagnosis task using our software application installed on a tablet device. After the task conclusion, they should fill a user experience questionnaire.

During the experimental task, we captured the application log to evaluate ODM and DWM performance, as well as the user behavior while they perform the exams analysis. In addition, we evaluated the user's perception of delay on the windowing actions, using a 5-point Likert scale from 1 (no impact) to 5 (strong impact). The user's perception about the quality of the images was also evaluated in a similar way, by using a Likert scale ranging from 1 (poor for diagnosis) to 5 (very good for diagnosis).

Data from CT, MRI and radiography were used, as they represent, together, all expected visualization behaviors. Table 1 summarizes the selected modalities and specialties. This selection also considered the frequency of occurrence of these cases in emergency. This is important because a mobile viewer is particularly useful in emergency, as it allows for rapid access to images through the network, which means early assessment and treatment for the patient.

The studied events have been selected from the partner clinic's database, anonymized and distributed according to the participant's specialty. They were stored in a PACS server located 300km away from the user, which accurately simulates the average real case of mobile access in which the device is connected to an external network to the clinic. During the image analysis by the radiologists, the system captured a log of all operations performed on the interface, such as zoom, windowing and slice navigation. The tablet used in the experiment was an Apple iPad 3 mini with 1 GB of RAM and connected to a wifi network 10Mbps shared by multiple users.

Table 1. Image modality distribution along body parts for the 40 events studied. Notice that we focused on cases that are especially common in emergency.

Body region	Radiography	CT	MRI
Head and neck	2	10	5
Abdomen and thorax	5	6	--
Bones and articulations	8	2	2

Results

The following subsections present the results obtained with our experiment. These results are categorized and discussed according to their connection to the observed user behavior while interacting, the dynamic windowing module and the on-demand download manager.

Behavioral pattern for user interaction

We observed two main usage patterns among the users during the analysis of the exams. The first one is that windowing actions are not needed for all situations. The second one is that the navigation among images of a series (performed by sliding left/right the image selector) is continuous and linear. As a consequence, if the current image being displayed is image 5, for instance, the next one more likely to be viewed is image 4 (previous) or 6 (next). Regarding windowing actions, in 17 out of 40 tests (42.5%) the radiologist performed a windowing action, while in 23 tests (57.5%) no windowing action was performed at all. When this information is grouped by exam type, in the 18 CT exams performed, 9 of them (50%) featured a windowing action; in the 7 MRI exams, 4 of them (57%) presented a windowing action, 3 (43%) did not present any; in the 15 radiographies exams, 4 of them (26%) featured windowing actions, 11 (74%) had no windowing actions. In summary, less than half of the exam analyses required windowing, and when radiography is used, only about 25% of the exams underwent windowing. The main reason for a windowing action not be required in the majority of exams is that the image acquisition protocol is often set to the average ideal window for each specific tissue and/or disease. Besides, depending on the exam, more than one protocol is used in such a way that an image is already available in different versions with varied windowing parameters. This is usually enough for the radiologist to make the analysis regardless of any extra windowing action. Regarding the behavior when visualizing a set of images/slices, Figure 3 illustrates the navigation pattern using the image selector during the analysis of four different exams. The y-axis represents the image that is currently being displayed (pivot), while the x-axis represents the event of exhibition of one image regardless of the time spent with each image (this equalization of the Δt was necessary for better visualization of the chart). An ascending line

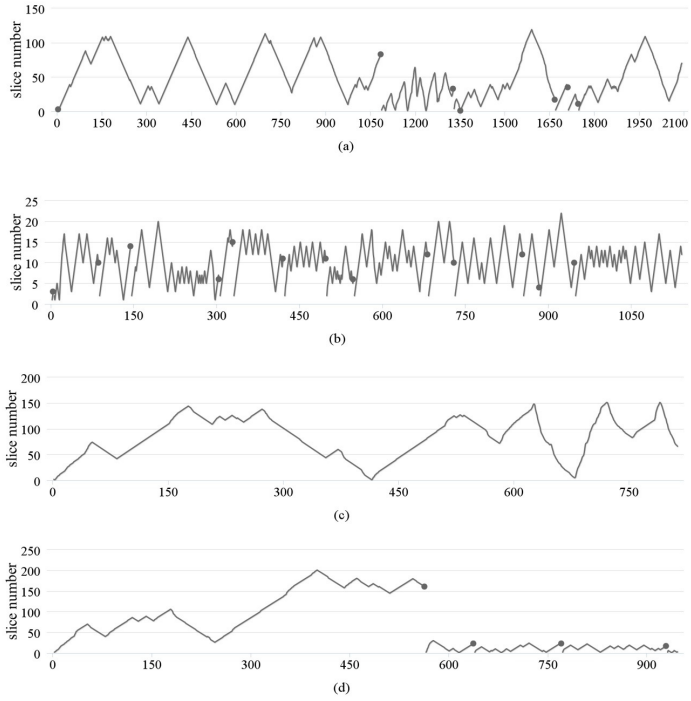

Figure 3. Slice change events during navigation with time equalized. An ascending line indicates the image selector is being slided to the right; a descending line means a movement to the left. The circles indicate a change to a new series of the exam. (a) CT of the head with 9-324 (series-images); (b) MRI of bones and joints with 8-129; (c) CT of the head with 1-125; (d) CT of bones and joints with 11-582.

in Figure 3 means the image selector is being slided to the right; a descending line means a slide to the left. The circles indicate when the user changes to a new series of the exam. We selected the four cases presented in Figure 3 as each of them represent a distinct group of exams.

The design of our ODM was based on empirical information that the user visualizes an image/slice more than once. The observed results confirm such behavior with statistical significance for exams with several images, such as CT and MRI, as illustrated in Figure 3. The linear navigation pattern is explicit, as the user navigates throughout a series moving back and forth (sliding the image selector left and right), especially in specific areas of interest for the diagnosis process. Such navigation pattern is not connected to the number of images/slices in a series. Figure 3c, for instance, presents an exam with a single series where the back and forth navigation movement is also recognizable. The same pattern is also recognizable in each one of the four series presented in Figure 3d, particularly in the first one, which is larger than the other three smaller series. The left/right changes in the navigation direction of the image selector are not equivalent. The radiologists tended to slide the image selector to the right more often (53.7%) than to the left (46.3%), i.e., they are more likely to move the image selector towards the end of the series than towards its beginning. We performed a Student's t-test, which demonstrated that this effect occurred with statistical significance ($p=0.0013$). Our implementation was based on the centralization of the ODM buffer according to the position of the pivot. The collected results, however, indicate that the ODM strategy would probably be more efficient if its buffer was not centralized, but instead, organized according to the previously mentioned proportions.

Dynamic windowing module

In this section, we present the results for the DWM and the impact of this module on the perceived user satisfaction when either local or remote windowing actions were performed. The results are based on the 17 analyses (out of 40) where users performed at least one windowing action. After performing the analysis, the users were questioned if they perceived any delays during the windowing actions. When the DWM selected a local (client-side) windowing approach, 70% of the users perceived no delays, while 30% reported delays in the operation. The users that perceived delays also answered a Likert scale question to measure to which extent such delay impacted their work. The average answer was 1.6 in a scale varying from 1 (no impact) to 5 (strong impact). When the DWM selected a remote

(server-side) windowing approach, 75% of the users perceived delays in the windowing actions, with a negative average impact of 2 in the same previously mentioned Likert scale. The perceived delay is then lower in the local windowing approach compared to the remote one. Results in Figure 4 were computed only for those images/slices on which windowing was applied at least once. The average number of visualization instances in those cases is far greater when the local approach is used. Actually, with the remote approach, only one image is generated per interaction (number of interactions = number of images rendered). With the local windowing, many new images are (locally) generated and displayed between the start and the end of an interaction. Generating many previews while interacting means immediate feedback to users, which improves the perceived user experience. Then, these results justify our efforts in making a local windowing approach available. It should be used whenever possible, i.e., when the raw image is not too large to be downloaded at the beginning of every new image windowing.

As previously explained, during our design phase, an empirical test indicated that the local windowing approach would transfer less data through the network compared to the remote approach after 24 inputs/previews of windowing parameters. As illustrated in Figure 4, the average number of windowing inputs/previews is 143.4, which is significantly greater than 24. This confirms our expectations that the local windowing approach is able to outperform its remote equivalent in the vast majority of cases. The local approach provides the users with more previews during the windowing operations, which improves the overall experience as well as minimizes the amount of data transferred over the network.

On-demand download manager

We analyzed the ODM behavior for events with MRI and CT scans. Radiography was excluded from this analysis as it does not contain multiple slices. Each column of Figure 5 shows an event where the blue part stands for the successful hits and the red corresponds to the access errors in the ODM buffer. The value at the top of the columns is the user satisfaction regarding the waiting time on the slices navigation task, in a 5-point Likert scale where 1 = "very slow" and 5 = "adequate". The 60% of users rated the task with the maximum score (5), 35% with score 4 and 5% with score 3. Even those cases with errors in the cache received a good evaluation due to the small waiting time for image transfer. Notice that some exams include more than one body part, e.g. head and spine. They are marked with * in Figure 5. It is also noticeable that the number of views is greater than the number of images of the study, showing that the user sees the same region more than once, as discussed above.

The average of hits for all events was 95.26% (SD=5.36). We noticed a correlation between the ODM errors and both the number of images/series and the windowing actions. The dashed line in Figure 6a represents the end of a windowing action, which removes all images with the old window from the buffer, causing the transfer of images with the new *ww* and *wl* values. Notice that just after the windowing action there is a region in red indicating buffer errors (misses). In Figure 6b, the buffer error (miss) is caused by another reason. Here, the user advanced the image-selector at a speed greater than the refresh rate of the ODM in a series with many slices.

It is noticeable that changes in the navigation direction usually occurred within the threshold of the

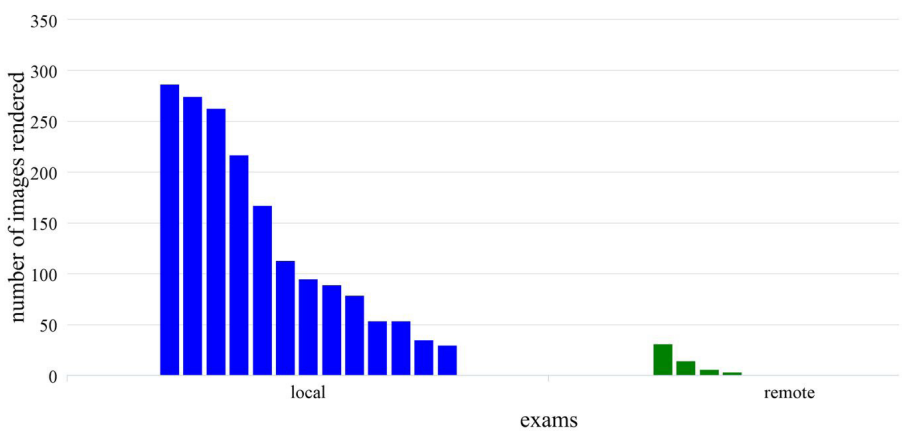

Figure 4. Average number of windowing previews per slice for the 17 individual exams. Only slices/images that have undergone at least one windowing operation are accounted. Results for each exam are ordered by the number of previews and grouped by the windowing approach.

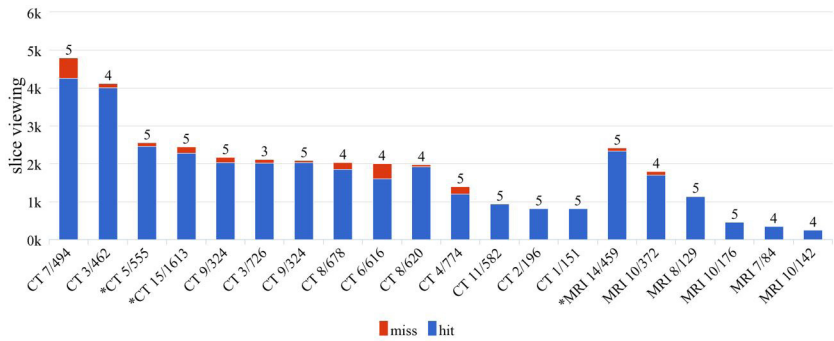

Figure 5. Number of slice visualization events with proportion of cache errors and user satisfaction. For each exam, the blue part stands for the successful hits and the red corresponds to the access errors in the ODM buffer. The value at the top of the columns is the user satisfaction regarding the waiting time on the slices navigation task, where 5 is the best.

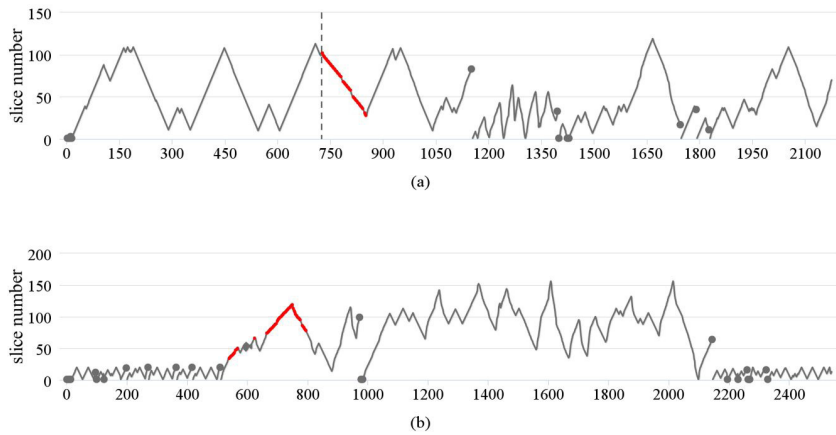

Figure 6. Slice change events during navigation with time equalized. The circles indicate a change to a new series of the exam. (a) CT of the head with 9-324 (series-images); (b) MRI of the head and angiography with 14-459 (series-images). The regions in red indicate buffer errors and the dashed line in (a) represents the end of a windowing action.

buffer update. This means visualization without waiting for the user. We also observed that exams with fewer images per series (in average), which are common in emergency cases, presented the best results in terms of cache errors. This also impacted on the perceived user experience. As shown in Figure 5, exams with larger series tend to present more cache errors and lower user satisfaction regarding the waiting time. In addition, the buffer errors are also sensitive to the type of exam (CT or MRI) and network performance during the experiment (number of connected users, peak hours, etc.). Finally, images with more information (less black areas) generate larger JPEGs, which may affect the general system performance.

To measure the impact of using compressed images in our approach, we evaluated the radiologists' perception of images quality. The results showed that 72.1% of the users answered 5 (very good for diagnosis) and 23.3% answered 4, which means the image quality is considered suitable for diagnosis. Grades from 1 to 3 (4.6%) were given only when a significant zoom factor was used during the image visualization or when the image was captured by a low-quality acquisition equipment.

Discussion

This paper presented a new approach that provides dynamic optimization for fast medical image transfer and visualization suitable for mobile and stationary devices. Our approach was implemented and validated using a real use case, the application Animati Viewer, which is a web viewer for diagnostic images, part of the Animati PACS.

Our results show that a hybrid approach using server- and client-side procedures is able to optimize the data flow on radiology image analysis. The two new approaches proposed in our design, the on-demand download manager (ODM) and the dynamic windowing module (DWM), work together to keep the application within the memory constraints, minimizing data transferring whenever possible. This combination has

demonstrated to be useful, allowing the application to run on a variety of devices (including tablets and smartphones) without sacrificing the user experience.

The ODM plays an important part in the process of controlling how much memory is used by the application. Its internal buffer dictates how many images will be stored in memory, which allows the application to estimate and adjust its memory consumption based on the available resources. As presented in our results, the amount of memory used by our implementation is constant, which is a key aspect to make the approach suitable to run on resource-constrained devices. Some previous researches work only with uncompressed DICOM files, affecting the access time on networks with low bandwidth (Choudhri and Radvany, 2011; Kaserer, 2013). Our approach, instead, uses compressed JPEG images in the whole process, which saves on network transfer, making the application capable of being used in situations where the connectivity is not ideal, e.g. mobile networks. Even though the compression process might interfere with the original image data, Kim et al. (2011) show that compressed medical images have already been used with good results in a system for rapid emergency care via mobile networks using the JPEG2000 algorithm. We confirmed that, during our tests, by asking the users to evaluate the quality of the images regarding the diagnosis process. In a Likert scale ranging from 1 (poor for diagnosis) to 5 (very good for diagnosis), 72.1% of the users answered 5 and 23.3% answered 4, which means the image quality is considered suitable for diagnosis. The results are encouraging to support the idea of using compressed images to minimize data transferring without sacrificing medical judgment. Nevertheless, the user can view the uncompressed version of an image, at any time, by performing a windowing action. When it happens, our approach relies on the DWM, which analyzes the size of the uncompressed image and decides the most appropriate way to apply such windowing. It ensures the user is always able to perform windowing actions, even when the network bandwidth is not suitable. As presented in the results, the accumulated amount of data transferred by the DWM can be adapted based on the circumstances. Under adverse conditions (e.g. poor connectivity) the DWM will perform all windowing actions in the server, downloading the result as a compressed JPEG. When conditions are ideal, the DWM will download the uncompressed version of the image and perform the windowing action locally. The results regarding the DWM confirmed our design decisions of trying to use a local windowing approach whenever possible. As previously mentioned, it allows the processing of a virtually unlimited number of windowing actions without network transfers, except the download of the uncompressed image used by the DWM to start the process. The majority of the users reported no perceived delay when the local approach was used. That approach is also able to minimize network traffic, which is important when the device is connected to a network with low bandwidth, for instance. Additionally, in the exams with higher resolution images, e.g. radiographies, where the remote windowing approach is required, users did report a perceived delay. However, this did not strongly impact the radiologist analysis.

Our experimental protocol with radiologists and 40 different datasets demonstrated that the Animati Viewer, after being equipped with our solution, was able to minimize the amount of information downloaded through the network, without sacrificing the user experience. As future work, further tests should be planned regarding the still incipient use of JPEG-compressed images in diagnostic tools. The use of compressed images is widespread in other areas and is a key aspect to keeping the download rate acceptable in environments with limited networking. One of the limitations of our work is the small number of professionals who evaluated the application, another suggestion is a field test including more hospital personnel to evaluate how our approach performs in a clinical context. We are planning face validation experiments in a clinical environment, especially focused on the ability to perform accurate diagnosis using mobile tools. Comparative experiments with standard workstation environments and other widespread mobile platforms, e.g. OsiriX mobile, should be devised.

Acknowledgements

Thanks are due to Medvia and the professionals that volunteered for the experiments. The study and applications of this research are part of the project Appification of Medical Reports funded through the MCTI/SETEC/CNPq No. 17/2012 - RHAE. Authors are also partially supported by CNPq grant 305071/2012-2, and FAPERGS project 2283-2551/14-8.

References

Bhatia A, Patel S, Pantol G, Wu Y, Plitnikas M, Hancock C. Intra and inter-observer reliability of mobile tablet PACS viewer system vs. standard PACS viewing station-diagnosis of acute central nervous system events. Open Journal of Radiology. 2013; 3(02):91-8. http://dx.doi.org/10.4236/ojrad.2013.32014.

Bourne R. Fundamentals of digital imaging in medicine. New York: Springer Science & Business Media; 2010.

Chandratilleke M, Honeybul S. Modifying clinicians use of PACS imaging. Journal of Digital Imaging. 2013; 26(6):1008-12. PMid:23670588. http://dx.doi.org/10.1007/s10278-013-9608-5.

Choudhri AF, Carr TM 3rd, Ho CP, Stone JR, Gay SB, Lambert DL. Handheld device review of abdominal CT for the evaluation of acute appendicitis. Journal of Digital Imaging. 2012; 25(4):492-6. PMid:22146833. http://dx.doi.org/10.1007/s10278-011-9431-9.

Choudhri AF, Radvany MG. Initial experience with a handheld device digital imaging and communications in medicine viewer: OsiriX mobile on the iPhone. Journal of Digital Imaging. 2011; 24(2):184-9. PMid:20567992. http://dx.doi.org/10.1007/s10278-010-9312-7.

Correa B, Ishikawa E, Ziviani A, Faria M. Medical image analysis using mobile devices. In Proceedings of the 2008 ACM symposium on Applied computing. In: SAC '08 Proceedings of the 2008 ACM Symposium on Applied Computing; 2008; Fortaleza, Ceará. New York: ACM; 2008. p. 1380-4. http://dx.doi.org/10.1145/1363686.1364005.

De Maio P, White LM, Bleakney R, Menezes RJ, Theodoropoulos J. Diagnostic accuracy of an iPhone DICOM viewer for the interpretation of magnetic resonance imaging of the knee. Clinical Journal of Sport Medicine. 2014; 24(4):308-14. PMid:24284945. http://dx.doi.org/10.1097/JSM.0000000000000005.

John S, Poh AC, Lim TC, Chan EH, Chong R. The iPad tablet computer for mobile on-call radiology diagnosis? Auditing discrepancy in CT and MRI reporting. Journal of Digital Imaging. 2012; 25(5):628-34. PMid:22562174. http://dx.doi.org/10.1007/s10278-012-9485-3.

Kaserer M. DICOM Web Viewer [thesis]. Wien: Technische Universität Wien; 2013.

Kim DK, Kim EY, Yang KH, Lee CK, Yoo SK. A mobile tele-radiology imaging system with JPEG2000 for an emergency care. Journal of Digital Imaging. 2011; 24(4):709-18. PMid:20824300. http://dx.doi.org/10.1007/s10278-010-9335-0.

Koutelakis GV, Lymperopoulos DK. PACS through web compatible with DICOM standard and WADO service: advantages and implementation. In: Proceedings of the 28th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS'06); 2006; New York, NY. USA: IEEE; 2006. p. 2601-5. PMid: 17945727. http://dx.doi.org/10.1109/IEMBS.2006.260761.

Pasha MF, Supramaniam S, Liang KK, Amran MA, Chandra BA, Rajeswari M. An android-based mobile medical image viewer and collaborative annotation: development issues and challenges. International Journal of Digital Content Technology and its Applications. 2012; 6(1):208-17. http://dx.doi.org/10.4156/jdcta.vol6.issue1.26.

Pianykh OS. Digital imaging and communications in medicine (DICOM): a practical introduction and survival guide. New York: Springer Science & Business Media; 2009.

Venson JE, Bevilacqua F, Maia CES, d'Ornellas MC. Dynamic optimization for fast image transfer and visualization for mobile and stationary devices: a performance evaluation using Animati Viewer. In: Workshop de Informática Médica (WIM); 2015; Recife, PE. 2015. Available from: http://www.lbd.dcc.ufmg.br/colecoes/wim/2015/027.pdf

Wang Z, Lin FX, Zhong L, Chishtie M. How effective is mobile browser cache? In: Proceedings of the 3rd ACM Workshop on Wireless of the Students, by the Students, for the Students; 2011; Las Vegas, NV. New York: ACM; 2011. p. 17-20.

West D. How mobile devices are transforming healthcare. Issues in Technology Innovation. 2012; 18(1):1-14.

Zwart CM, He M, Wu T, Demaerschalk BM, Mitchell JR, Hara AK. Selection and pilot implementation of a mobile image viewer: a case study. JMIR mHealth and uHealth. 2015; 3(2):e45. PMid:26018587. http://dx.doi.org/10.2196/mhealth.4271.

Authors

José Eduardo Venson[1,2]*, Fernando Bevilacqua[3,4], Fabio Onuki[5], Marcos Cordeiro d'Ornellas[6], Anderson Maciel[1]

[1] Institute of Informatics – INF, Universidade Federal do Rio Grande do Sul – UFRGS, Av. Bento Gonçalves, 9500, CEP 90040-060, Porto Alegre, RS, Brazil.

[2] Animati - Computação aplicada à Saúde, Santa Maria, RS, Brazil.

[3] The Informatics Research Centre (Interaction Lab), University of Skövde, Sweden.

[4] Universidade Federal da Fronteira Sul – UFFS, Chapecó, SC, Brazil.

[5] Medvia Diagnósticos, Porto Alegre, RS, Brazil.

[6] Laboratory for Applied Computing – LaCA, Universidade Federal de Santa Maria – UFSM, Santa Maria, RS, Brazil.