J Digit Imaging (2011) 24:833–843
DOI 10.1007/s10278-010-9343-0

Using JPEG 2000 Interactive Protocol to Stream a Large Image or a Large Image Set

Rita Noumeir · Jean-François Pambrun

Published online: 27 October 2010

Abstract The electronic health record (EHR) is expected to improve the quality of care by enabling access to relevant information at the diagnostic decision moment. During deployment efforts for including images in the EHR, a main challenge has come up from the need to compare old images with current ones. When old images reside in a different system, they need to be imported for visualization which leads to a problem related to persistency management and information consistency. A solution consisting in avoiding image import is achievable with image streaming. In this paper we present, evaluate, and discuss two medical-specific streaming use cases: displaying a large image such as a digital mammography image and displaying a large set of relatively small images such as a large CT series.

Keywords Medical imaging · Electronic health record · Image communication · Image streaming · JPIP · JPEG 2000

Introduction

The deployment of the cross enterprise document sharing for imaging (XDS-I) [1] as the framework for sharing images within the electronic health record (EHR) is taking place in many countries, including Canada, USA, Japan, and several European countries. As part of those deployment efforts, several difficulties have emerged. The main origin of these difficulties resides in the need to compare old images with current ones, as part of the diagnostic interpretation process of current images. In fact, to conduct the interpretation, the radiologist usually compares the current images with prior ones that may have been acquired in a different enterprise. With the EHR, the radiologist knows about the existence of those priors and can access them; however, comparison is conducted within a single software application that offers specific operations for medical imaging interpretation, such as a synchronized navigation between two different image sets. This application is thus required to have access to both image sets.

Presently, most medical imaging applications assume images are under their complete control. In other words, they assume all images are identified and managed in a single consistent way. This assumption does not hold when foreign images need to be imported into the system, as identification schemes are different between several enterprises and may result in identification that is not unique. Patient and order identifications are such examples. A work-around for this problem exists. It consists in coercing identifications in the images before importing them from the EHR into the local application; however, importing foreign images into a local application creates a new major problem related to persistency management and information consistency.

Image import is basically image duplication. Present medical imaging applications, as part of the broader picture archiving and communication system (PACS), visualize and process images that they manage and archive. Then, PACS need to import the foreign image for visualization, but, how can foreign images be identified as such so they can be deleted or discarded at the end of the process? Moreover, information in the original image (such as the one present in its medical header) can be corrected; how can that correction be propagated to the duplicated instance?

R. Noumeir (✉) · J.-F. Pambrun
Department of Electrical Engineering, University of Quebec,
École de technologie supérieure,
Montreal, Canada
e-mail: Rita.noumeir@etsmtl.ca

One possible solution for all the previously stated problems consists in avoiding image import. This is achievable with image streaming. JPEG 2000 Interactive Protocol (JPIP) [2, 3] is a standard that can be combined with XDS-I to enable streaming of medical images directly from EHR connected imaging sources to image processing workstations. This can be achieved with the use of the Digital Imaging and Communications in Medicine (DICOM) Web access to persistent objects [4] along with the DICOM JPIP referenced transfer syntax [5].

In this paper, we implement two medical-specific streaming use cases: displaying a large image such as a digital mammography image and displaying a large set of relatively small images such as a computed tomography (CT) series. We also present measurements of bandwidth efficiency and improvements.

Method

JPIP

JPIP is a client/server standard image streaming protocol. It allows a client application to request only portions of a JPEG 2000 image that are necessary to fulfill the client's viewing needs.

JPIP streaming relieves the client application from importing the image into its environment eliminating thus the problems of persistency, consistency, and reconciliation. It also results in an improvement in bandwidth efficiency because the complete image is not needed at the client application; only the necessary needed information is streamed to fulfill the viewing requirements. This improvement is very important in medical imaging as medical images are either large images or very large image sets.

JPIP is based on JPEG 2000 standard. This latter is a standard for compressing images. It allows an image to be compressed with various parameters, including transformation levels and precincts. An introduction to JPEG 2000 compression standard is provided in the Appendix. Understanding JPEG 2000 parameters is important for implementing JPIP effectively. Readers interested in more details about JPEG 2000 are invited to read [6] while others interested in implementing the results of this study can apply the recommended compression parameters depending on the image type.

JPIP is a standard way to express requests from the client's application to the server. The server application has access to the JPEG 2000 image. To fulfill a client's request, the server fetches data from the JPEG 2000 image and sends it to the client that decompresses the JPEG 2000 stream into an image to display (Fig. 1).

JPIP allows the server or the client to keep, in its cache, information about what data is already available to the client (Fig. 1). If the client uses a cache, then the client would include, in its request, information about what is already available, e.g., the client would say "I have this" and "I need to view that." If the server uses a cache, then the client does not need to include anything about what is already available relying on the server cache, e.g., the client would say "I need to view that" and the server would know what additional information is necessary. The server cache implementation requires managing several caches, one per client; the client cache implementation is easier to implement while requiring more bandwidth for the client's request.

Use Cases

The JPEG 2000 data available to the server has a direct impact on the streamed data to the client. In other words, one needs to know how to compress the JPEG 2000 image that is available to the server, as changing the compression parameters has a major impact on the bytes that are streamed. Therefore, to decide how to compress the image using JPEG 2000, one needs to know how the image will be requested using JPIP or, in other words, how the user would manipulate the image. Of course, image manipulation is not unique. It depends mainly on the image modality. In this study, we considered two different use cases: viewing a large image and viewing a large image set. The first case is encountered when viewing a mammography or a large X-ray image; the second case is encountered when viewing a large CT or magnetic resonance (MR) image set which is composed of a big number of relatively small images.

Stack Navigation of a Large Image Set

We assume that the CT or MR image is smaller than the screen size; therefore, the image will not be downsized to fit the screen and the entire image will be commonly visualized. On the other hand, we consider that the user may want to rapidly scroll through the image set; therefore, a progressive transfer of the image would be of interest where the image is first downloaded with a lower resolution followed by additional information, automatically requested, up to a full resolution image.

Large Single-Image Navigation

We assume that a large image has a dimension bigger than the screen size on which it is displayed; therefore, it is not necessary to fetch the complete resolution of that image as it will be reduced to fit the screen. Consequently, when viewing a large single image on the screen, only the portion

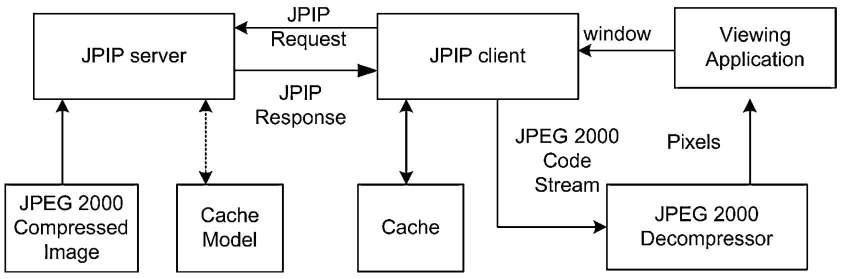


Fig. 1 JPIP Client/Server main components

of that image that is displayed needs to be fetched up to a resolution that fits the screen resolution. Then, if the user changes the viewing window by zooming and panning through the image, a new request is sent to the JPIP server specifying the new window. Additional information for that specific portion of the image is sent to the client application.

This use case can also be implemented with a progressive data transfer. The main difference with the large image set use case is the relevance here of regions of interest.

Test System Implementation

We have implemented a JPIP test system to quantify the amount of data transfer involved in streaming medical images from an image archive to an imaging visualization workstation. The test system is based on a client/server architecture where the server is capable of delivering DICOM and JPEG 2000 files to a client. Medical images, encoded using the DICOM standard, are compressed according to JPEG 2000. They are made available to the server. The server parses a JPIP request and streams required data blocks from the JPEG 2000 compressed files.

The server is also capable of gathering information and statistics about time and data transfer to help in the analysis and performance assessment. These statistics can be saved for further evaluation. They include the amount of received bytes, the amount of sent bytes, the processing time, the request received by the server, the request header as well as the response header.

We have used the system to conduct experiments in order to gather measurements about data transfer and to provide recommendations with regard to JPEG 2000 compression parameters specific to the use cases considered. The test cases were implemented by simulating viewer interactions. This was achieved by implementing specific JPIP query sequences that are hard-coded within the software of the client.

Results

The JPIP server must have access to a JPEG 2000 compressed image in order to stream it. Simply stated, a JPEG 2000 image can be considered as a stream of coefficients. When part of this stream is discarded, the image is lossy-compressed. JPIP consists of transferring part of the available stream to fulfill the client's need. The JPIP server can transfer the available coefficients progressively until there are no more coefficients to send. If the available coefficients are those of a lossy-compressed image, then the JPIP server would be serving a lossy-compressed image. If the available coefficients are those of a lossless-compressed image then the server would be serving a lossless-compressed image. The structure of the encoded stream, available to the JPIP server is the same regardless whether the image was lossy- or lossless-compressed. Therefore, lossy compression image according to [7] can be used with JPIP. Although, the results presented in this paper are obtained with lossless compression, the same compression parameters and JPIP requests could be used with lossy-compressed images. When lossy-compressed images are used, the stream is shorter leading to a shorter streaming time. Of course, the best resolution would be the one of the lossy-compressed image because the information that is lost during the lossy compression stage cannot be retrieved.

Several parameters affect how JPEG 2000 compression is achieved. The most common parameter is the compression ratio which dictates the ratio of the lossy compression. This parameter has no impact on JPIP; in this work we will consider lossless compression without any loss of generality. Other parameters such as the number of decomposition levels, the precinct sizes, and the use of quality layers have direct impact on the JPIP performance.

A precinct can be considered as a group of coefficients that belong to a specific region of interest. If a JPIP interaction involves region of interests, then the precinct's size has a direct impact on JPIP performance. This is the case when streaming a large image. When JPIP interactions do not involve transferring region of interests, precincts are not relevant. This is the case for small images such as MR or CT.

The rest of this section is devoted to evaluate the bandwidth gain from using JPIP. Our goals are twofold: (1) evaluate JPIP performance and quantify the gain of using it. (2) Derive recommendations with regards to the optimal JPEG 2000 compression parameters, specific to each image type. Optimal, in this context, is defined as providing the best JPIP performance: the minimum amount of bytes transferred to fulfill the user's needs.

Displaying a Large Series of Relatively Small Images

In this section, we consider the case of a series containing a large number of images, each image having a size smaller than the screen size. This is a common case encountered with a series of CT (or MR) images when they are composed of a large number of images, each having a size of 512×512 pixels. Because the image size is smaller than the screen size, the complete image is displayed at any time. Consequently, precincts are not relevant here, as precincts enable the transfer of full resolution region of interest; however, the number of images in a series may be very large (i.e., 2000 images). Therefore, displaying the image at a lower resolution first, then improving the quality until a full resolution image is obtained, enables quick navigation through the stack.

Progressive image transfer can be obtained either by resolution or by quality layers. Progression by resolution implies transferring lower resolution first followed by higher resolution subbands. LL2 (see Appendix for the definition of LL2) is 16 times smaller than the initial image. It can be interpolated to fill the size of the initial image. Likewise, LL1 is four times smaller than the initial image and can be interpolated to the initial image size. On the other hand, progression by quality implies compressing the image with various quality layers and transferring them progressively. Each quality layer adds information and therefore quality, to the previously transferred layers. When all layers are used to reconstruct an image, the full resolution image is obtained. The criteria on which to base layer definition are defined at the JPEG 2000 encoder. Example include bit rate, signal to noise ratio, or distortion ratio. The number of layers to use only dictates the number of possible refinements. In our experiment, we have considered ten quality layers. Besides the quality layers, we considered three decomposition levels for use with an image size of 512×512 pixels; also, we used large precincts which is equivalent to considering each subband as a single precinct.

Two experiments were conducted. In the first experiment, we generated JPIP requests to request progressive quality enhancements. The same image reconstructed with different quality layers is shown in Fig. 2. Root mean square error, normalized Root mean square error as well as peak signal to noise ratio (PSNR) are calculated for each reconstructed image and are shown in Table 1.

$$\mathrm{RMSE}\left(I,\hat{I}\right)=\sqrt{\sum_{x}\sum_{y}\left(\left(I_{(x,y)}-\hat{I}_{(x,y)}\right)^{2}\right)} \tag{1}$$

$$\mathrm{NRMSE}\left(I,\hat{I}\right)=\frac{\mathrm{RMSE}\left(I,\hat{I}\right)}{I_{\max}-I_{\min}} \tag{2}$$

$$\mathrm{PSNR}\left(I,\hat{I}\right)=20\log_{10}\left(\frac{I_{\max}-I_{\min}}{\mathrm{RMSE}\left(I,\hat{I}\right)}\right) \tag{3}$$

Table 1 also shows the additional bytes required to transfer each quality layer, an image at the lowest quality layer requires 5,442 bytes; each additional quality layer improves the quality of the image and requires additional bytes to be transferred. The full resolution image requires 173,243 bytes as compared to its uncompressed size of 528,150 bytes. The quality of the image improves with the layers, evidently.

In the second experiment, a resolution-based progression scheme is tested; images are initially downloaded at a resolution of 64×64 pixels then resolution is incremented progressively until 512×512 pixels, the original image size is reached. The amounts of data downloaded are given in Table 2.

The image with a resolution of 64×64 pixels scaled to 512×512 is depicted in Fig. 3. Visually, this image seems to be of inferior quality when compared to the image reconstructed with a single quality layer only (see Fig. 2). In both cases, the transferred size is about 5 kilobytes.

Displaying a Large Image

A large mammography image is used (Fig. 4). It has a width of 3,540 pixels and a height of 4,740 pixels, its size is 33,562,298 bytes. The image is compressed with five decomposition levels. Precincts are used to achieve full resolution regions of interest. Even though tiles could have been used to achieve the transmission of full resolution regions of interest, they have not been explored in this work because of the blocking artifact that occurs at the edges of the tiles at low resolution. The precinct size of subbands HL2, LH2, and HH2 is considered equal to 128×128 pixels. The size of all other precincts is considered equal to 256×256 pixels. The image is supposed to be visualized on a screen whose width is 1,920 pixels and whose height is 1,080 pixels. This is the screen size of a common computer. Evidently, this size is different from the common radiology dedicated workstation screen sizes that are in use nowadays; however, screen sizes and images sizes are continuously increasing, but the discussion here will always be valid as far as the screen size is smaller than the image size.

Clearly, the screen size is smaller than the image size; therefore information from low-resolution subbands up to LL2 is enough. The amount of bytes needed to have a preview that best fits the screen size is 463,599 bytes. This represents the size of LL2 that is used to view the image at the screen resolution. Therefore, precincts of subbands at a resolution lower than LL2 have not been used and are not relevant here.

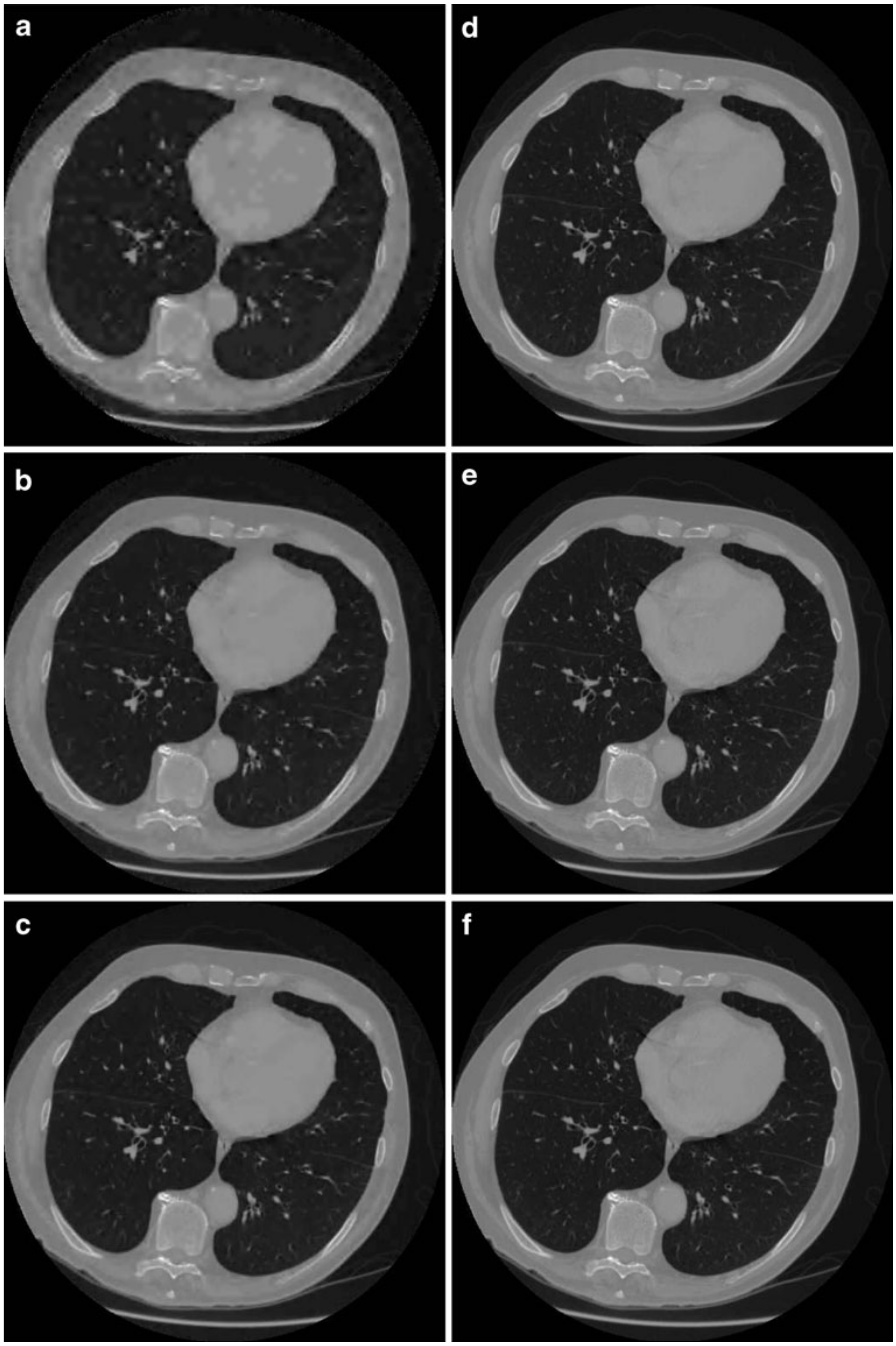


Fig. 2 Same image reconstructed with different quality layers. **a** One quality layer. **b** Two quality layers. **c** Three quality layers. **d** Four quality layers. **e** Seven quality layers. **f** Ten of ten quality layers

JPIP requests have been generated to simulate a lens tool that is used to visit the image completely, according to a navigation scheme that goes top down, from left to right. The regions of interest are shown in Fig. 4 as grid lines superimposed on the image. The region of interest is considered of size 256×256 pixels. The additional bytes needed to display full resolution regions of interest are recorded and are shown in Table 3.

The total amount of bytes to view the complete image at full resolution is 6,966,349. This is achieved after visiting all regions of interest. It is slightly bigger than the initial image size, but this is not important here. Of importance is the amount of bytes required to visualize the image at the best screen resolution which is 463,599 compared to the full resolution size of 6,966,349, leading thus to a compression ratio of about 16:1. Also, of importance is

Table 1 Bytes transferred and error measurements for ten quality layers

Quality layers	Bytes transferred	RMSE	NRMSE (%)	PSNR (dB)
1	5,442	58.53	2.34	32.61
2	5,289	33.26	1.33	37.52
3	4,686	23.19	0.93	40.65
4	11,457	13.31	0.53	45.48
5	26,026	6.25	0.25	52.04
6	11,492	4.90	0.20	54.15
7	24,533	2.74	0.11	59.19
8	22,258	1.71	0.07	63.31
9	62,060	0		Infinity
10	The image was fully transferred from step 9. Images 9 and 10 are identical			
Total	173,243			

RMSE root mean square error, *NRMSE* normalized root mean square error

the additional amount of bytes required to visualize a single region of interest which is about 40 kilobytes. This means that while the user is either panning within a zoomed image or using a lens tool to examine a region of interest, only about 40 kilobytes are requested for each region. Moreover, one can note that many regions do not contain information of diagnostic value. These regions correspond to the background and occupy, in the case of this mammography image, about 60% of the whole image. These regions are normally not examined at full resolution, therefore the additional bytes to visualize these background regions at full resolution may not be requested.

This experiment has been repeated with different JPIP requests in order to progressively download the first image that best fits the screen.

In order to achieve progressive rendering, the image has been compressed with compression parameters as before with the additional use of quality layers. The compression parameters consist of five decomposition levels, precinct size of subbands HL2, LH2, and HH2 equal to 128×128 pixels, other precinct size equal to 256×256 pixels, and ten quality layers.

Quality layers are requested to be downloaded progressively, the lowest quality layer followed by a better quality layer, until all quality layers are requested. This enables a low quality initial image to be displayed very quickly while subsequently refined until best screen resolution is attained. PSNR is calculated for each reconstructed image, it is shown in Table 4 along with the additional bytes required to transfer each quality layer.

An image at the lowest quality layer requires 57,848 bytes only, compared to the full resolution size of the image that is 6,966,349 bytes. Each additional quality layer improves the quality of the image and requires additional bytes to be transferred; the total is the amount of bytes needed to display the image at the best resolution of the screen. Compared to the full resolution of the image, a compression ratio over 15:1 is achieved. Visualization of regions of interest can also be achieved progressively [8].

Table 2 Bytes downloaded per resolution level

Resolution level	Bytes downloaded
64×64	5,032
128×128	11,856
256×256	40,061
512×512	116,283
	173,224

Conclusion

JPIP brings two major advantages when viewing medical images in a distributed environment. The first advantage comes from the streaming capability which eliminates the need for importing foreign images into a medical image archive, avoiding thus the problems related to information consistency and persistency management of duplicated

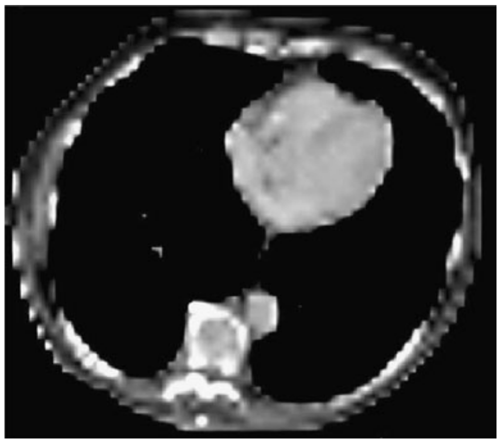


Fig. 3 A 64×64-pixel image scaled to its original size of 512×512 pixels

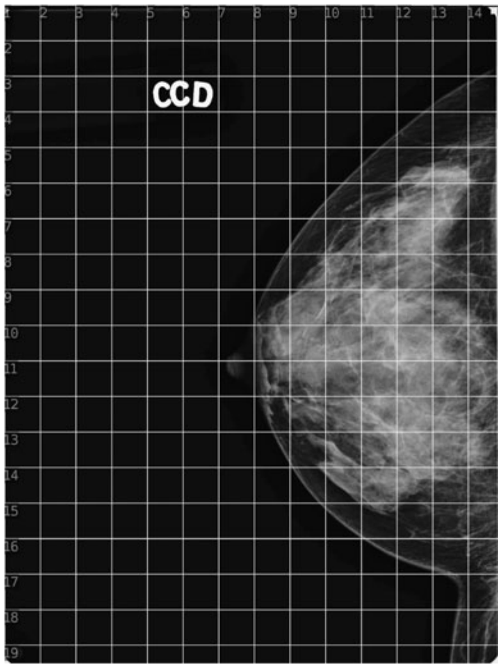


Fig. 4 The large image over which a grid of 256×256 pixels is drawn

images. The second advantage comes from the significant improvement in bandwidth efficiency when viewing medical images which usually are either large images or very large image sets.

JPIP enables the client application to visualize a large image at the best screen resolution with much less data than required when visualizing the same image lossless-compressed. This additional "compression" depends on the ratio of image size to screen size. Moreover, JPIP enables the display of full resolution regions with very little additional data transfer. We found that only additional 40 kilobytes enabled the rendering of a full resolution region of size 256× 256 pixels. Moreover, background regions will not require any additional transfer as these regions do not need to be examined at full resolution, normally. This may represent as much as 60% of a mammography image.

Also, JPIP enables the client application to visualize an image very quickly with a low quality while enabling quality progressive refinement at a subsequent moment. Fast preview of images allow the user to quickly navigate through a stack composed of a large number of images without waiting for full fidelity images to be transferred. Rendering of the initial image can be attained 30 times faster of what can be achieved with lossless-compressed images. Progressive refinements allow the user to focus on full resolution images selectively and subsequently. To achieve this use case, progression based on quality layers provides better results than progression based on resolution level. The number of quality layers defines the number of possible refinements. If this number is high, many refinements are needed in order to attain full resolution while each refinement would require a small amount of bytes. Progressive refinement of the preview image could also be used when viewing a large image.

Discussion

While using JPIP to deliver medical images from the EHR to the radiologist's workstation appears very promising, many challenges still exist and will be discussed hereafter.

Trade-offs for Considering JPIP Technology JPIP offers gains in image transfer performance and in persistency management at the cost of an additional architecture complexity. This complexity comes from a need of an additional server, the JPIP server, which must have access to the images in a JPEG 2000-compressed format.

Lossless vs Lossy Compression JPIP can stream any images encoded in a JPEG 2000 format regardless of whether the images have been lossless- or lossy-compressed. Evidently, with lossy compression some information is definitely lost during the compression step and is not streamed with JPIP, but the good news is that images can be lossy-compressed in accordance to the recommendations of [7] and can still be streamed with JPIP.

Compression Parameters The JPEG 2000 compression parameters impact the JPIP performance. Therefore, we have proposed two different sets of parameters to be used while compressing large images such the one encountered in radiography image modalities (CR, DX, MG, etc.) or small images from large stacks such as the one encountered in MR or CT. Therefore, the JPIP implementation should enable a system administrator to specify the compression parameters per modality type.

Access to Images The JPIP server needs to have access to the images in a JPEG 2000-compressed format. If they are not available from the image archive in this specific format or in a related one such as DICOM JPEG 2000 format, the images would then need to be transformed before streaming. Transforming the images into a JPEG 2000-compressed format requires processing time. The output requires space storage; it also requires special persistence management as these compressed images are duplications of the archived ones, therefore are not intended for long-term storage.

PACS installations have workflow requirements that involve the usage of short-term duplicated images; they normally manage cached versions of images for enhanced performance. They usually rely on order messages to prepare prior images in advance and automatically purge

Table 3 Additional transfer size for viewing regions of interest

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	15,629	16,518	16,286	16,236	16,126	16,127	15,832	15,956	16,208	16,048	16,051	15,780	15,742	12,946
2	13,870	13,641	14,345	14,359	13,010	13,852	14,112	13,856	14,005	13,947	14,095	14,212	13,744	12,846
3	19,805	20,195	20,144	17,207	24,671	24,070	13,111	13,771	13,875	14,150	14,298	14,197	27,581	25,688
4	17,655	17,168	16,412	15,129	12,937	12,705	13,829	14,093	14,045	13,781	16,103	33,564	38,707	39,044
5	14,050	13,950	13,742	13,916	13,996	13,813	13,747	13,986	13,747	17,301	38,414	46,938	53,616	43,885
6	14,318	13,866	13,983	13,936	13,853	14,013	13,657	13,465	16,327	42,276	51,499	56,218	55,817	42,434
7	14,614	13,887	14,098	14,095	14,357	13,904	13,985	13,594	38,981	52,284	56,465	56,136	53,365	42,285
8	14,746	13,942	14,388	14,321	14,158	14,144	13,641	25,300	48,271	56,469	56,615	56,752	55,250	43,829
9	14,665	14,459	14,430	14,573	14,291	14,586	12,973	44,701	56,293	56,756	55,491	56,419	55,835	44,771
10	15,054	14,316	14,213	14,346	14,347	14,324	18,056	54,431	56,531	56,507	56,280	55,907	56,461	46,095
11	14,987	14,575	14,366	14,422	14,231	14,121	20,109	51,925	56,115	56,671	56,681	56,470	56,755	46,891
12	15,328	14,964	14,594	14,716	14,627	14,480	13,421	36,650	54,533	56,506	56,240	56,329	56,840	46,987
13	14,989	14,595	14,496	14,710	14,427	14,490	14,110	20,410	47,298	55,486	56,781	56,680	56,885	46,847
14	15,264	14,611	14,724	14,850	14,625	14,356	14,691	13,765	32,007	49,930	56,717	55,096	54,700	45,709
15	15,011	14,700	14,660	14,773	14,418	14,545	14,295	14,647	14,085	32,647	43,413	45,419	50,888	44,782
16	15,049	14,743	14,676	14,696	14,790	14,400	14,583	14,541	14,516	13,734	22,531	35,985	44,410	42,450
17	15,140	14,821	14,620	14,771	14,777	14,874	14,913	14,803	14,680	14,698	14,430	13,609	23,665	37,761
18	14,746	14,563	14,847	14,687	14,929	14,772	14,683	14,899	14,627	14,653	14,368	14,688	26,246	39,868
19	8,506	7,609	7,530	7,580	7,654	7,584	7,499	7,574	7,571	7,592	7,742	7,473	14,831	3,633

Table 4 Bytes transferred and PSNR measurements for ten quality layers

Quality layer	PSNR (dB)	Bytes downloaded
1	44.67	57,848
2	51.81	64,580
3	57.95	63,149
4	62.01	79,832
5	63.04	31,890
6	66.73	108,365
8	68.81	34,392
8	103.83	33,811
9	Infinity	249
	Total	474,353

cache of duplicated images based on the image's age or the frequency of access. Pre-preparation of images and cache management can be set up in a similar way; however, on-the-fly compression would still be needed for images that are requested and are not ready for immediate streaming. Pre-preparation and on-the-fly compression can be combined in the same implementation to address all possible cases.

Management and Operational Challenges In order to have access to archived images that are managed by the image manager/archive, the JPIP server needs to be integrated with an image manager/archive. This integration can be either tight or loose. In a tight integration, the JPIP server is part of the archive system; it is provided by the image archive vendor and integrates with the archive in a proprietary manner, such as with direct access to the file system or database where the images are stored. Evidently, JPIP is a new technology, and most archive systems do not provide JPIP capabilities. In a loose integration, the JPIP server is provided by a system that is different from the image archive by a third party possibly. The JPIP server can integrate with the archive system using a standard communication protocol such as DICOM. This type of integration adds a communication overhead whose impact is mitigated if images are pre-prepared; however, when images are not ready in the short-term cache of the JPIP server, on-the-fly access to the image archive increases the response time. This proxy-based solution can be deployed with any archive that does not provide JPIP capabilities.

Integration with Visualization Workstations JPIP is based on a client/server architecture. While deploying JPIP, a server should enable and encourage the deployment of JPIP clients; without the successful deployment of JPIP clients, the JPIP servers are useless. The JPIP client needs to be integrated with the visualization workstation. This integration cannot be loose. It has to be a tight integration because the user interaction with the system at the graphical user interface level directly dictates JPIP interactions. Examples of such direct links can be grasped by thinking of the user zooming and panning through the image; the user's action requires specific JPIP requests. The tight integration of the JPIP client with the visualization workstation is the most important challenge to JPIP deployment.

Acknowledgment This work was supported by a grant from Canada Health Infoway and the Government of Quebec.

Appendix: Introduction to JPEG 2000

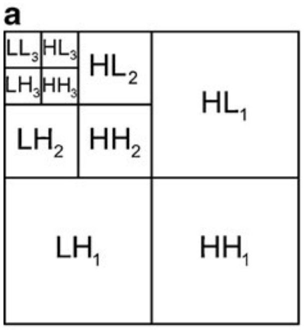
JPEG 2000 is an image compression method based on the discrete wavelet transform (DWT). The DWT is specified by a pair of two linear filters, a high-pass filter and a low-pass filter. In JPEG 2000, two types of filters are used, the Daubechies 9/7 and Le Gall 5/3 filter; the latter is used for lossless compression because it generates integer values. When those filters are applied on an image, four subbands are generated; they contain the output of the filters, the wavelet coefficients:

- Applying the low-pass filter on both the rows and columns generates a lower resolution image (LL).
- Applying the low-pass filter on one direction and the high-pass filter on the other direction generates a subband that contains coefficient for low frequency in one direction and high frequency in the other direction respectively (HL and LH).
- Applying the high-pass filter on both directions generates a high frequency subband (HH).

The subbands are half the size of the original image because the decomposition includes a subsampling by two. The low-pass subband represents a smaller low-resolution version of the original image. The high-pass subbands represent high frequency information that is needed along with the low-pass subband for a perfect reconstruction of the original image.

The decomposition of the low frequency subband can be repeated (Fig. 5), and there is no restriction on the number of decomposition other than the size of the image itself.

JPEG 2000 allows progressive decompression. One straightforward scheme is to start with the low-pass image at the lower decomposition level, add all subbands from the upper decomposition level to reconstruct a



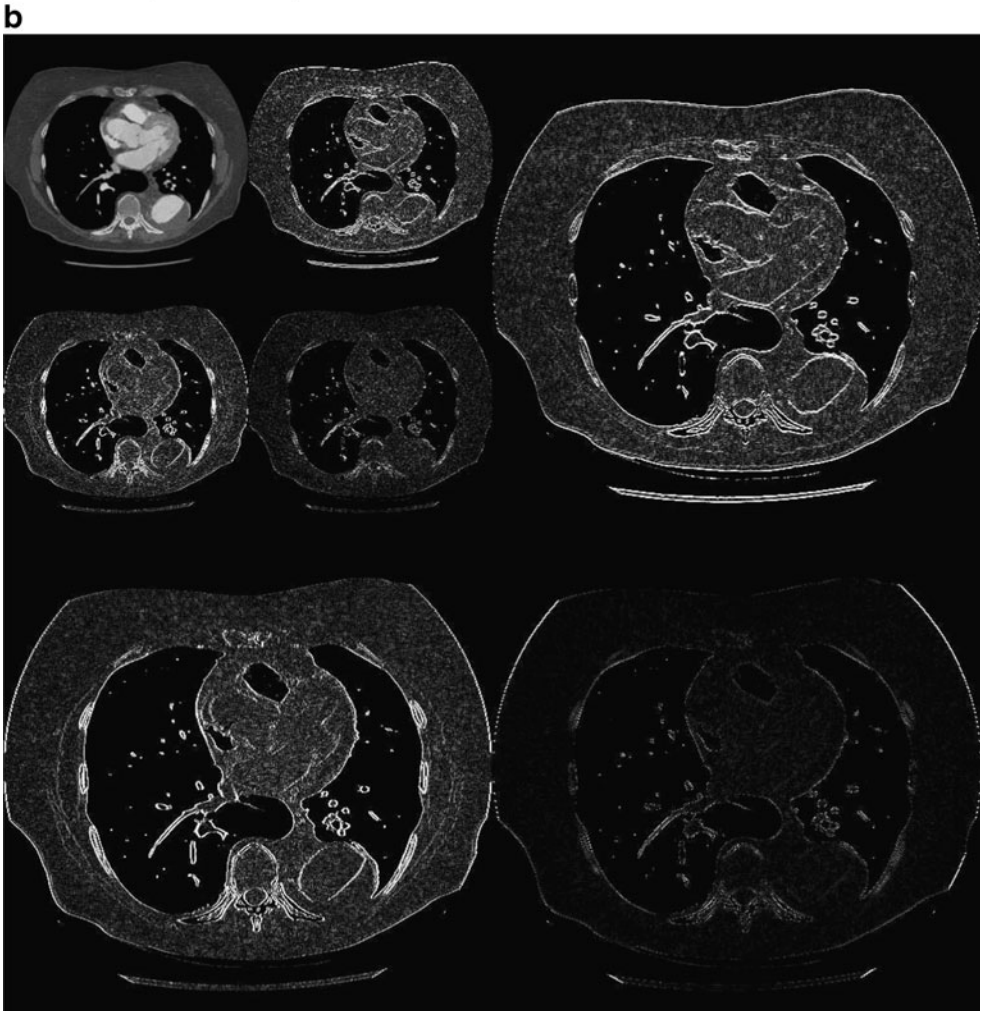


Fig. 5 A wavelet decomposition with three levels. **a** Wavelets subbands. **b** Wavelets decomposition on a CT image

new low pass and so on until the complete image is reconstructed.

Full Resolution Region of Interest and Space–Frequency Localization

The wavelet transform has an important property known as the space–frequency localization; this comes from the fact that contiguous pixels in the original image are transformed into contiguous wavelet coefficients in each of the four subbands (Fig. 6).

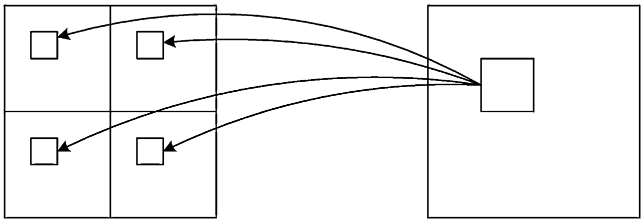


Fig. 6 Pixels that are neighbors in the original images are transformed into coefficients that are neighbors

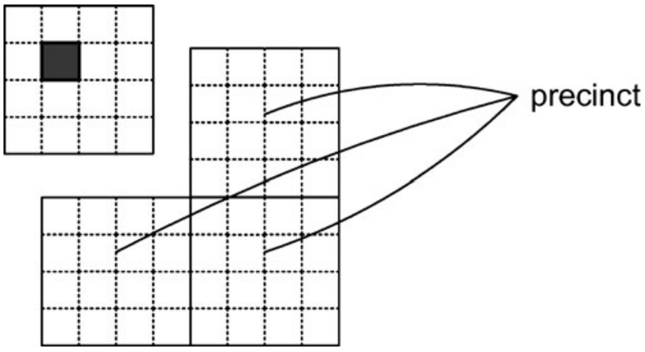


Fig. 7 Precincts

Precincts and Progressive Reconstruction of a Region of Interest

In order to achieve a progressive reconstruction of a specific region, JPEG 2000 allows a subband to be subdivided into rectangular regions. For each rectangular region in the LL subband, there are three rectangular regions, one in each of the higher frequency subband. This set of three regions is called a precinct (Fig. 7).

Precincts from all decomposition levels that form a specific spatial region are used to reconstruct a full resolution region. Precincts are important for interactive navigation. In fact, if we are interested in panning into a large image, only the precincts that correspond to the viewing window need to be transferred. In case where the window overlaps multiple precincts, all of the overlapped precincts would be needed.

References

1. IHE Technical Framework and Supplements. [Online] Available http://www.ihe.net/Technical_Framework/index.cfm. Accessed 2 March 2010.
2. Taubman D, Prandolini R: "Architecture, philosophy, and performance of JPIP: internet protocol standard for JPEG2000." In: Proc. International Symposium on Visual Communications and Image Processing (VCIP '03), vol. 5150 of Proceedings of SPIE, Lugano, Switzerland, July 2003, pp 791–805.
3. Lima L, Taubman D, Leonardi R: JPIP proxy server for remote browsing of JPEG2000 images, IEEE 10th Workshop on Multimedia Signal Processing (MMSP), 2008, pp 844–849.
4. The Digital Imaging and Communications in Medicine (DICOM) Standard. [Online] Available http://medical.nema.org. Accessed 2 March 2010.
5. Noumeir R, Pambrun JF: Images within the Electronic Health Record, IEEE International Conference on Image Processing, 2009, pp1761–1764.
6. Taubman D, Marcellin MW: JPEG 2000: Image compression fundamentals, standards and practice. Kluwer Academic Publishers, Norwell, MA, 2001
7. Koff D, Bak P, Brownrigg P, Hosseinzadeh D, Khademi A, Kiss A, Lepanto L, Michalak T, Shulman H, Volkening A: Pan-Canadian evaluation of irreversible compression ratios ("lossy" compression) for development of national guidelines. Journal of Digital Imaging 22(6):569–578, 2009
8. Noumeir R, Pambrun JF: Streaming of medical images using JPEG 2000 interactive protcol, IWSSIP 2010 17th international conference on systems, signals and image processing, to appear, 2010.