Web-Based Manipulation of Multiresolution Micro-CT Images

Lasse Wollatz, Simon Cox and Steven Johnston
Faculty of Engineering and the Environment
University of Southampton
United Kingdom
Email: L.Wollatz@soton.ac.uk

***Abstract*—Micro Computed-Tomography (µCT) scanning is opening a new world for medical researchers. Scientific data of several tens of gigabytes per image is created and usually requires storage on a common server such as Picture Archiving and Communication Systems (PACS). Previewing this data online in a meaningful way is an essential part of these systems. Radiologists who have been working with CT data for a long time are commonly looking at two-dimensional slices of 3D image stacks. Conventional web-viewers such as Google Maps and Deep Zoom use tiled multiresolution-images for faster display of large 2D data. In the medical area this approach is being adapted for high resolution 2D images. Solutions that include basic image processing still rely on browser external solutions and high-performance client-machines. In this paper we optimized and modified Brain Maps API to create an interactive orthogonal-sectioning image-viewer for medical µCT scans, based on JavaScript and HTML5. We show that tiling of images reduces the processing time by a factor of two. Different file formats are compared regarding their quality and time to display. As well a sample end-to-end application demonstrates the feasibility of this solution for custom made image acquisition systems.**

I. Introduction

Images have always been important in medicine for diagnostics. Medical instruments such as microscopes and µCT scanners produce images that are several tens of gigabytes in size. The increasing amount of data obtained provides new opportunities for medical researchers but also poses challenges in terms of data storage and retrieval.

In histology sections of *in-vitro* tissue-samples are stained and analyzed under a microscope. The results coming from such analysis are limited to one 2D sample of the extracted tissue. Creating 3D data of the whole tissue using non-destructive µCT prior to the sectioning allows to create a more complete picture of diseases included in the tissue.

µCT data and resulting processed data need to be available to medical researchers in an accessible and meaningful way. One of the critical elements identified was fast previewing of images, in order to allow researchers to decide which images are worth analyzing before retrieving the full image data from a remote storage onto their system for processing. As mobile devices are of increasing importance in today's world and also within medicine, compatibility with mobile devices becomes a key concern [1].

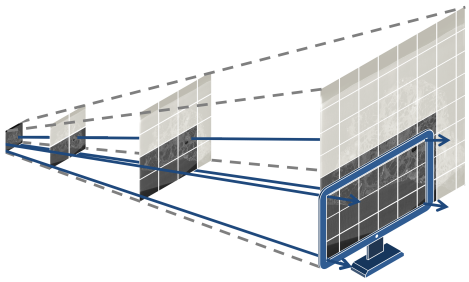

Fig. 1. Multiresolution tiled image - Each resolution level is four times the size of the previous. Tiles are loaded from server for the current view only.

Sending full scale image over the web can take a long time. It is therefore of advantage to save the image as small tiles and only send those parts of the image which are required (compare Fig. 1). It is important to balance the tile size between amount of data to load from the server and amount of server requests made [2].

If the user zooms out, the whole image is displayed on the screen. As this means all the tiles need to be loaded, the image is scaled to several different resolutions and tiled at each of these resolutions. The result is a multiresolution image as shown in Fig. 1. For images where the overhead from multiple tile requests is too large, loading multiple resolutions will still provide a faster response experience. The major idea of the viewer is not to speed up the loading, but to limit the amount of work which needs to be done. For non-medical images, the display of very large images over the web by the use of multiresolution tiled images is a well-established method. Examples are Google Maps, where a satellite image of the entire earth is displayed over various zoom levels with individual tiles being loaded on demand, Deep Zoom, which is based on Silverlight Plug-In, and Grid DataBlade by BCS, which runs in Java [3], [4].

In the medical field the implementation of tiled image viewers is also being established [5], [6]. These solutions are limited to displaying images and do not support image filters.

The Multiresolution Image Viewer (MIV) [5], later renamed to Brain Maps API, is an open source code for displaying high resolution brain images. These images are taken with microscopes, which have a very high resolution

compared to µCT. Even though microscopes only allow to view a small part of the sample at high magnification, methods exist to stitch these images back together to form a complete image of the sample [7]. After being stitched together, the image is converted into a multiresolution, tiled image and stored in a predefined folder structure. Brain Maps API loads and places the image tiles needed for the current view and was extended by StackVis [8], which is browser external, to enable the display of coarsely spaced 3D image stacks and 2D images with a birds-view perspective.

In this paper we describe an implementation of a multiresolution tiled image viewer based on the Brain Maps API [5] that makes it possible to view images and process them pixel based on low-performance devices within web-browser. This improves the capabilities of existing Web-PACS and the practical usability of remote data storage for e-Health. Using Portable Network Graphics (PNG) image tiles instead of commonly used Joint Photographic Experts Group (JPEG or JPG) tiles, better results are obtained as shown in section II. Requirements for speed improvement are discussed in section III and an integration into an image server as well as performance of the code are presented in section IV.

II. Image Format Consideration

Raw data can be stored in various ways, such as uncompressed bits or in a file format like Tagged Image File Format (TIFF). TIFF allows storing 16-bit or 32-bit data as well as being able to store multiple images in one file. Compared to raw data files, it also presents a standard for storing keywords related to the image. As standard TIFF uses 4-byte pointers it can only store file-sizes up to 4GB. Images larger than that are stored as image stacks of several separate 2D TIFFs.

The tiles for the image viewer need to be stored in a web-compliant format. Modern web-browsers support only standard 8-bit images natively commonly JPG, Graphics Interchange Format (GIF) and PNG but also Windows Bitmap (BMP). GIF, which uses an 8-bit indexed color map, has been replaced by PNG due to licensing restrictions [9]. PNG provided 5% better lossless compression than JPEG for small images. JPG can be seen as a standard file format for tiled image viewers [2], [8], [10], [11] but server side solutions also use their own multiresolution file formats or compression methods [11], [12]. JPEG2000, which is increasingly popular in medicine, applies partial lossy compression [13]. This paper compares lossless compressed PNG and lossy compressed JPG and as an uncompressed format BMP. Comparison is made in terms of quality loss due to conversion and quality loss due to compression as well as file-size.

Radiologists can discern 800 to 1000 Just Noticeable Differences (JNDs) within one scene, corresponding to 800 to 1000 individual gray-shades. This means that a 10-bit image format with 1024 gray shades is a minimum requirement for accurate display [14], [15]. A visible loss

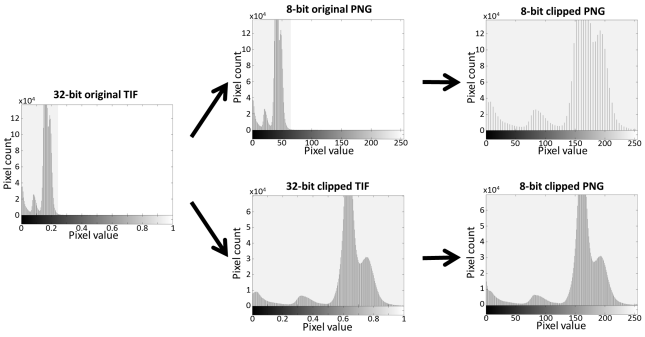

Fig. 2. Quality of image for two possible conversions to 8-bit are compared: The upper row shows the case of the histogram first converted to 8-bit and then stretched, while the lower row shows the histogram first stretched and then converted. The number of gray shades visibly reduces in the first case.

in intensity-resolution is therefore given independent of the choice of file format for the tiling. The original images from the µCT come at 32-bit but are reduced to 16-bit. This is sufficient quality for medical purposes [16]. TIFF allows for high resolution but standard image formats are based on 8-bit integers for gray-scale ranging from 0 to 255. The difference becomes noticeable if the displayed density range is changed and the histogram stretched as illustrated in the top of Fig. 2. A small improvement can be achieved, by partial clipping of the original histogram to make better use of the limited 8-bit palette if the original image uses only a small range of the available gray-scales as shown at the bottom of Fig. 2.

Besides the quality loss due to conversion to 8-bit, the compression-methods of the different image-formats play a role. The different image resolutions vary by a scaling factor of two, meaning that each zoom level contains only a quarter of the pixels of the previous one. This means that in total the tiled image will contain 4/3 the amount of pixels. The overhead of having several files instead of a single one adds to this. JPEG uses a 2D frequency transform to convert the image into the frequency domain and only stores the most important frequencies. This allows for much smaller file sizes but less quality. PNG uses lossless compression and BMP uses none or run-length encoding (RLE) compression only. Fig. 3 shows the difference in terms of image quality. An increased amount of artifacts for JPEG is visible. Lossless formats do not show major differences to the original image, even for large image scaling. The Root Mean Square (RMS) of the difference between the different images and the original TIF were computed for the selected tile. Both BMP and PNG resulted in an RMS of 0.005 while converting the image to JPEG and then tiling it gave a difference RMS of 0.0144 and tiling the image prior to the conversion one of 0.0134. Lossy JPEG compression reduced the image size of the multiresolution image shown in Table I by a factor of 15 which is much less than reported for normal images. This is mainly due to tiling and the increased amount of pixels of the multiresolution image [17]. PNG achieved the expected

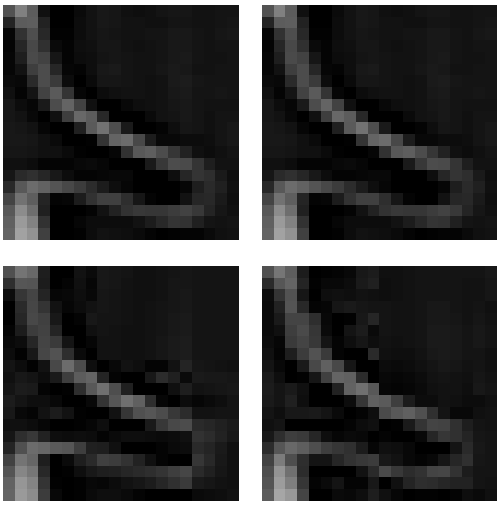

Fig. 3. Quality of JPEG compression compared to PNG based on a 20 by 20 pixel tile. Images on the left side result from the TIF being tiled and then converted to the new file format. Images on the right have first been converted and then cut into tiles. The JPEG images at the bottom show a lot of errors.

TABLE I
SIZE OF TILED IMAGES COMPARED TO THE ORIGINAL UNTILED TIF.

Image	Orig. TIF	Tiled JPG	Tiled PNG	Tiled BMP
Sample 1 (32-bit)	8.13GB	138MB	855MB	2.89GB
Sample 2 (16-bit)	7.09GB	265MB	1.79GB	5.01GB
Sample 3 (16-bit)	60.8GB	2.1GB	9.53GB	21.1GB

compression ratio of a factor of 3 [18].

If the increased amount of data transfer results in an unacceptable delay of page display, then PNG cannot be used for the tiled images. Otherwise it is preferred, as the quality is much greater. If the original data is several tens of gigabytes in size the larger overall data size of PNG compared to JPG-tiles has no major significance.

III. SPEED CONSIDERATION

The speed of web-sites is a major issue [19]. Server, network and coding related improvements were explored in order to ensure short website response times.

The processing/ rendering of the images occurs on the client side. For comparison an AJAX based loading of the images was implemented allowing for server side image processing but was much slower than the client-side processing through pure JavaScript. This agrees with the observations regarding Deep Zoom that server side image generation is slow as it increases the load on the server [20].

In order to decrease the page loading time, the JavaScript routines called on start-up were reduced to a minimum. The image loads on a comparatively small zoom level to keep the number of image files transferred on page load low. For further speed-up the code was modified to reduce the amounts of time tiles were recomputed and loaded. A differentiation was made between the case of tiles needing to be updated and the case where the tiles visible might have changed. In the first case all the tiles need to be recomputed. This occurs if the user changes the image manipulation parameters. The second case requires checking which tiles are within the field of view and loading those, while removing the ones that moved out of the displaying area. This event needs to be triggered whenever the zoom-level is changed, or the image is panned.

A low-resolution thumbnail was placed behind the tiles to allow quick orientation while higher resolution images are loading. To create a better feeling of responsiveness, the thumbnail is always loaded before the tiles and placed in the background. Tiles will then appear on top of the low resolution image, as they are processed. In case of quick scrolling through the dataset, the low resolution image loads much quicker than the tiles and improves the user orientation in the 3D stack.

Another change was made in the amount of slices the user goes through: it was made dependent on the zoom level and the angle of mouse-wheel movement. At higher magnification the user moves only a single slice at a time, while at low resolution he can move 20 images at once. This way it gives the feeling of moving through the image more quickly, while reducing the number of that need to be loaded.

IV. RESULTS

A sample website was built using Webmatrix 3 and PL-upload. If the server receives new images, a message is sent to a queue hosted on Azure which defines the image path. A Python script on the server checks the queue frequently for outstanding messages and creates the image tiles in a subdirectory of the original image. It also creates a header file for the image viewer that contains information about the overall 3D image. The viewer requests that file using AJAX. The pixel values and dimensions are mapped back to the density of an object using the header file. The user can select a Hounsfield unit (HU) range to display within the boundaries of the scanned HU values. By adding the queue request into the uploading process and linking the content request of the Image viewer to the systems database, this viewer can be added to an existing web image server.

For speed evaluation, the time to respond (TTR) and the time to display (TTD) were recorded. The average times are presented in Table II. The TTR defines the time it took to display a preview after a user input, while the TTD describes the amount of time it took till the full resolution image was displayed. In general TTD was 3 to 4% faster for JPG compared to PNG but 25% to 60% faster than for the untiled image. For low zoom levels, the TTD new image slices averaged at 90fps for panning only required loading a selection of new tiles and was even faster (around 6ms). On mobile devices, the advantage of JPG over PNG is much clearer with 16 to 45% speed improvement. Loading and editing the full image on a mobile device took 16 seconds on average but was decreased to 4 seconds by tiling. Due to the multiresolution loading the TTR is much lower at 63ms for a workstation and

TABLE II
TTR AND TTD USING DIFFERENT FILE FORMATS. TTD THE UNTILED PNG ON THE TABLET WAS NOT RECORDED, AS THE BROWSER APPLICATION CRASHED BEFORE COMPLETING THE REQUEST.

Device	Browser	Average	Average TTD [ms]			
		TTR [ms]	JPG	PNG	BMP	Untiled PNG
PC	Opera	32	1401	1456	1530	3514
PC	IE	20	1613	1633	1643	5013
PC	Firefox	94	2962	3056	3025	3975
Mobile	Opera	354	2478	3060	3734	20436
Mobile	Firefox	239	5855	6988	8611	11429
Tablet	Safari	465	6835	7481	8194	—

a fifth of a second for the mobile phone. The average speed of displaying different size images did not differ as expected. The reduction to equal sized tiles allows massive scalability not only in two but also in three dimensions. Results are shown for Opera, Firefox (FF) and Internet Explorer (IE).

Comparing PNG and JPG, differences in compression and speed are negligible. For the given medical applications, PNG was the better choice due to the lower quality of JPG.

For the test, the zoom level was set to ensure a maximum number of images were loaded. Speed was measured from the point of user input to the point of image displayed. It was ensured that images were not cached. These were the worst conditions possible and this is reflected in the results.

The server was hosted with IIS on Windows 8 and the times recorded on a Windows 7 (both 16GB RAM) with Opera 28, the results were compared to FF 35 and IE 11. Each request required 38 tiles to be loaded. Mobile speeds were recorded on a Samsung Galaxy Duos (Android 4, 645MB RAM, FF & Opera, 8 Tiles to load) and iPad (iOS 5, Safari, 20 Tiles to load) using Wi-Fi with VPN. The image loading time fluctuated massively, especially on slower devices. As this was not tested in an isolated environment, the workload on both - server and client - was not constant.

V. CONCLUSION

Advances in the application of µCT in e-Health lead to growing amounts of data requiring organization and remote viewing. Viewing should not be dependent on browser-external solutions, as this limits the compatibility. Multiresolution images can avoid these limitations. We created an image viewer and manipulator for 3D CT data and showed that it can be implemented into an existing system. We demonstrate the feasibility of this method for medical applications in several aspects: a) Using lossless compressed image formats does improve the overall speed of the viewer compared to untiled approaches; b) Basic requirements including pixel based image processing of medical image viewers can be implemented into a web-viewer; c) These features do not restrict the usability of the system for mobile or low-performance devices; d) Using JavaScript and HTML5 these functionalities can be implemented for all main browsers.

In the future we plan to extend the support of input formats and reformatting options to create an image server for practical use. For that we would like to explore direct integration of DICOM as well as security aspects of the transmission of medical images over the web.

REFERENCES

[1] P. T. Johnson, S. L. Zimmerman, D. Heath, J. Eng, K. M. Horton, W. W. Scott, and E. K. Fishman, "The iPad as a mobile device for CT display and interpretation: diagnostic accuracy for identification of pulmonary embolism," *Emerg. Radiol.*, vol. 19, no. 4, pp. 323–327, Aug. 2012.

[2] L. Gerhard, A. Szoforan, and R. Nickolov, "Scalable mutable tiled multi-resolution texture atlases," U.S. Grant 8 385 669, Feb. 26, 2013.

[3] C. Crichton, J. Davies, J. Gibbons, A. Tsui, J. Brenton, C. Caldas, and L. Morris, "Deep Zoom and touch screen for tissue microarray image scoring," in *Proc. of ICSE*, vol. 70, no. 2003. ACM Press, 2008, pp. 217–243.

[4] C. Pendleton, "The world according to Bing," *IEEE Comput. Graph.*, vol. 30, no. 4, pp. 15–17, Jul./Aug. 2010.

[5] S. Mikula, I. Trotts, J. M. Stone, and E. G. Jones, "Internet-enabled high-resolution brain mappingand virtual microscopy," *Neuroimage*, vol. 35, no. 1, pp. 9–15, 2007.

[6] E. J. C. Arguiñarena, J. E. Macchi, P. P. Escobar, M. del Fresno, J. M. Massa, and M. A. Santiago, "Dcm-Ar: a fast Flash-based web-PACS viewer for displaying large DICOM images," in *Proc. of EMBC*. Buenos Aires: IEEE, 2010, pp. 3463–3466.

[7] B. Appleton, A. P. Bradley, and M. Wildermoth, "Towards optimal image stitching for virtual microscopy," in *Proc. of DICTA'05*. Queensland, Australia: IEEE, Dec. 2005, pp. 44–51.

[8] I. Trotts, S. Mikula, and E. G. Jones, "Interactive visualization of multiresolution image stacks in 3D," *Neuroimage*, vol. 35, no. 3, pp. 1038–1043, Apr. 2007.

[9] J. Miano, *Compressed Image File Formats: JPEG, PNG, GIF, XBM, BMP*. Addison-Wesley Professional, 1999.

[10] C. K. Chui and H. Wang, "System and method for tiled multiresolution encoding/decoding and communication with lossless selective regions of interest via data reuse," US Grant 09/999,760, Jun. 7, 2005.

[11] W.-K. Jeong, J. Schneider, S. G. Turney, B. E. Faulkner-Jones, D. Meyer, R. Westermann, R. C. Reid, J. Lichtman, and H. Pfister, "Interactive histology and of large-scale and biomedical image and stacks," in *IEEE T. Vis. Comput. Gr.*, vol. 16, no. 6. IEEE, 2010, pp. 1386–1395.

[12] M. J. Gormish, E. L. Schwartz, A. Keith, M. Boliek, and A. Zandi, "Lossless and nearly lossless compression for high-quality images," in *Proc. of Very High Resolution and Quality Imaging II*, vol. 3025. SPIE, Apr. 1997, pp. 62–70.

[13] D. A. Clunie, "Lossless compression of grayscale medical images: effectiveness of traditional and state-of-the-art approaches," in *Proc. of Medical Imaging*, G. J. Blaine and E. L. Siegel, Eds., vol. 3980. San Diego, CA: SPIE, Feb. 2000.

[14] H. R. Blume and E. Muka, "Hard copies for digital medical images: an overview," in *Proc. of Color Hard Copy and Graphic Arts IV*, vol. 2413. San Jose, CA: SPIE, Feb. 1995.

[15] R. A. Robb, *Biomedical Imaging, Visualization, and Analysis*. John Wiley & Sons, 2000.

[16] A. E. Scott, D. M. Vasilescu, K. A. D. Seal, S. D. Keyes, N. M. Mark, J. C. Hogg, I. Sinclair, J. A. Warner, T.-L. Hackett, and P. M. Lackie, "Three dimensional imaging of paraffin embedded human lung tissue samples by micro-computed tomography," *PLOS ONE*, vol. 10, no. 6, Jun. 2015.

[17] R. Norcen, M. Podesser, A. Pommer, H.-P. Schmidt, and A. Uhla, "Confidential storage and transmission of medical image data," *Comput. Biol. Med.*, vol. 33, no. 3, pp. 277–292, May 2003.

[18] P. Cosman, R. Gray, and R. Olshen, "Evaluating quality of compressed medical images: Snr, subjective rating, and diagnostic accuracy," in *Proc. of the IEEE*, vol. 82, no. 6. IEEE, Jun. 1994, pp. 919–932.

[19] D. Gehrke and E. Turban, "Determinants of successful website design: relative importance and recommendations for effectiveness," in *Proc. of the HICSS-32*, vol. 5. Maui, HI, USA: IEEE, Jan. 1999, p. 8.

[20] J. Prosise, "Wicked code: Taking Silverlight Deep Zoom to the next level," *MSDN Magazine*, vol. 24, no. 7, p. 90, Jul. 2009.