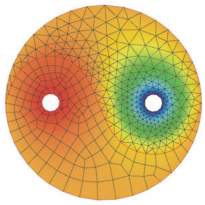
Publishing House
AKAPIT

COMPUTER METHODS IN MATERIALS SCIENCE

Informatyka w Technologii Materiałów

Vol. 15, 2015, No. 3



VISUALIZING THE HIGH-RESOLUTION MODELS AND MICROSCOPIC IMAGES USING WEB-BASED SYSTEMS

KAMIL J. DUDEK[1*], OSKAR BOŻEK[2]

[1] Akademia Górniczo-Hutnicza im. Stanisława Staszica, al. A. Mickiewicza 30, 30-059 Kraków, Poland

[2] Studenckie Koło Naukowe przy Katedrze i Zakładzie Patomorfologii Wydziału Lekarskiego Śląski Uniwersytet Medyczny, Katowice

**Corresponding author: kamdud@agh.edu.pl*

Abstract

Despite the ongoing automation of image processing and analysis, the need of manual browsing, i. e. visual examination has not diminished. The following article presents a dynamic and highly efficient method of presenting and streaming the large resolution images, using the Linux web-based front-end, built on top of the Internet Image Protocol (IIP) Image library and tools. This kind of approach not only solves the problem of local storage and duplication of results, but also allows a wider group of users to browse and annotate the images. The central image repository, dedicated for managing significant amount of data, has a variety of potential applications, both in research and in teaching. Taking a step sideways, this paper is meant to walk through the initial work and plans related to applying IIP Image in metallurgical imaging, current applications on other fields and suggested teaching scenarios. The system description is followed by a brief performance analysis.

Key words: imaging, visualization, iipsrv, modeling

1. INTRODUCTION

To state the obvious, multiple sources as well as types of the images used in digital metallurgy are known to exist. Those include the microscopic input, cellular automata-generated microstructures, intermediate computation steps and many more. Most software both commercially available and custom-made for the purpose of research are created solely with the local data storage in mind. It means that the entire image database is kept and processed locally. The burden of image-manipulation computing overhead is borne by the "local" workstation. As far as the local storage is concerned, it obviously cannot be eliminated entirely due to practical reasons. However, dealing with a large database, especially if used mostly for browsing, creates serious storage space demand and significantly reduces the potential of workplace portability. The in-office work model is becoming less and less popular, being replaced by remoting, mobile devices etc., leading to frequent switching of workplace and devices. Moreover, developing collaboration requires establishing a well-accessible and responsive repository of data, code and images. All those reasons speak against the locally-centered software design and raise a requirement of other solution to the problem of storage and retrieving pictures.

The most common point between the aforementioned technologies is the web browser. Currently, the flexibility and power of web-based applications allow the creation of solutions with feature sets comparable to standalone desktop software, while being much more portable and usually more accessible. Feature-rich websites offer almost limitless po-

ISSN 1641-8581

tential, due to constantly developing back-end libraries and powerful HTML5 standard. Versality of this technology makes it suitable also for visualizing.

2. PREREQUISITES

Creating a visualization utility, using the aforementioned technology requires an agreement on multiple issues, most notably the file format. Input formats vary, depending on the utility used and in many cases, the choice of format has not been discussed. Sometimes the format is predefined: there are devices that offer only reduced-quality JPEG. Learning materials frequently rely on scanned monochrome images taken from the undigitized literature pieces. Custom-made software simply picks the easily-available API. This approach often means using the uncompressed, heavy Windows BMP bitmaps, just because the *System.Drawing.Bitmap* namespace was simple enough and readilyavailable. Such decisions cause an cascade of problems highlighted in the Introduction.

Those formats have little in common. This issue creates the need for a unification layer, that makes the input differences invisible to the end user. Images stored in the database of offered slides can be contained in any format, as long as it is supported by the visualizing software. Thanks to that, the dataset creator can focus on truly important matters, which are:

- Image Quality (or losslessness)
- Fidelity/Accuracy (size in pixels)

Surprisingly, high-definition images are not always available. The author has also came across multiple cases where the big images were indeed prepared, but stored in the lossy JPEG format with the quality set to fixed 75%. All these issues prove that there is a strong requirement for reorganization of stored images as quite often there are no rules related to them in the entire scientific process. Highlighting the importance of imaging has proved to be a demanding and somewhat dull task, yet the potential advantages might outweigh the difficulties.

Solving this discrepancy is not exactly a purely technical task: using low-resolution or paperonly images is an old habit that can not be overcome by rules or guidelines. It means that common digitization of imaged resources has to be a teamwork.

3. SUGGESTED SOLUTION

Assuming that the image set is prepared, the browsing and post-processing utility can be created. Hardware-wise, it requires a web server to serve the interface, using the previously outlined technologies.

3.1. Apache HTTP Server

The reference system is being built on Linux-based operating systems, namely Fedora and Debian Testing. They host an Apache2 web server with FastCGI support and PHP Pear. Selection of Apache is a technicality, an equal option in a variety of few others, but it proved itself to be a robust and efficient platform. The imaging server itself is server-independent and works on any HTTP server as long as the library dependencies and Common Gateway Interface (CGI) support is supplied.

3.2. IIP Image Server

What is actually essential for the server is the IIP Image software (Pitzalis et al., 2006). The iipsrv layer works on top of the HTTP Server as the FastCGI application. The most distinguished qualities of iipsrv include exemplary scalability and efficiency, especially when compared with its modest computing power requirements. Thanks to that, it can be easily used on non-specialized hardware, like business grade laptops, before being deployed to the production environment.

Areas where iipsrv is currently used most heavily include medical and biomedical microscopy, but the proceedings in astronomy have also been made, as noted by Bertin et al. (2015). Those fields quite often deal with highly demanding large images of top and/or lossless quality (also without the agreement on one single image processing format). The investigated use case includes the initial set of basic, scanned images that were not designed to be digitized. Therefore, 600 DPI scan does not improve quality, as it enhances the printout dots, instead of the actual microstructure details. This disappointing outcome is a direct result of the long-lasting assumption that those images are "good enough". Strong dedication to paper media is therefore a significant piece of computer methods science.

Scanned images were saved as an (at least) 600 DPI lossless TIFF files. Such an overkill approach definitely excludes the possibility of common smooth image browsing in realtime, considering how computationally demanding the processes of zooming and scrolling are, especially in case of the created database. This is where the true power of IIP Server comes in: during the image database post-

processing it creates a set of "sub-images", in a manner of a pyramid. Each big image is divided into few levels. Every level consists of more "sub-images", and on each level they sum to the initial image (required navigation controls are shown on figure 1). This approach makes it possible to receive few sets of tiles, for example 16x16, 8x8, 4x4, 2x2 and, of course, 1x1 (original). So instead of serving the entire image, only the requested area can be delivered, and if the user decides to zoom the image, another level of tiles is picked, and the corresponding area, consisting of smaller tiles, is served. That feature offers an array of potential applications. First of all, it enables the possibility of smooth image exploring. The "grab and move" method does not require repetitive recomputation of the image due to the smaller, more easily processed images. Zooming is much easier as well. Those features greatly facilitate the visual image analysis and assessment.

What is not covered by the current IIP Image software includes capabilities like processing the proprietary microscoping image formats and, obviously, raw datafile-to-image conversion which would be a significantly promising way of development (Murray, 2015). This issue will be highlighted in the later paragraphs as it is considered a "missing link" in the process of data interchange.

3.3. Storage

In order to offer pyramid images there is a need for a considerable amount of storage server system that conforms to multiple criteria, most notably:

- Store space, a varying increment of the initial size of the image, multiplied by the amount of requested images
- Storage device random access speed that mitigates the risk of "bottleneck". Currently the element of a computer system that degrades performance in the most severe way is the storage. The issue is being resolved by gradual migration to solid state storage, which in effect gravely reduces the storage space (much higher cost of gigabyte compared to disk-based media).
- Network infrastructure responsiveness. A slow, unreliable or unstable network connection will defy the purpose of dynamic image reprocessing, as the time saved on skipping the redrawing will be lost while waiting for the data to pass the network band.

The format picked for the internal initial image set is a modified version of TIFF (Khelou, 2008; Pitzalis et al., 2006). It is still compatible, but includes additional overlay data used to compose a pyramid. IIP offers support for other formats, but the accompanying libraries do not provide acceptable efficiency due to lack of multithreading, 32-bit precision (unlike TIFF) and proper development track. Fixing, reimplementing or extending them would require an unacceptably high amount of time and would grow to an unwanted side project, stalling the initiative. However, the open source imaging community awaits those missing features and/or libraries. In conclusion, the TIFF format support is considered superior. All the input is pre-processed into TIFF, not the other way around, where IIP is enriched with support of a variety of, often obscure, formats.

3.4. Pre-processing

The frustrating variety of the discussed input feeds requires introducing a mandatory conversion step in the process of creating the system. Two image processing frameworks wew selected out of numerous other available: ImageMagick and VIPS, both considered the most reliable, with the long-lasting development and support. VIPS is a newer project than the former, with more promising improvements, compared to the somewhat stagnant ImageMagick. Therefore, it was selected as an initial, non-interactive pre-processor for the data input.

The crucial point of pre-processing is introducing the redundancy into the images. It comes from the aforementioned "pyramid" stack offered by the TIFF format. That's why the image set is expected to be initially converted into pyramid TIFFs. The first step consists of remeshing the entire picture collection, is performed by an unattended shell script. In order to prepare the image, VIPS is invoked, using the following server command: vips im_vips2tiff *img-src* *img-out*.TIFF:*compr-level*, tile:*tilesize*,pyramid . The size of the tile and the compression level provided within the compr-level and tilesize variables, must be chosen before the automated processing. The required collection integrity (and the results of the stress test that follows) strongly enforces the same tile size for all the pieces and the highest achieveable quality. Setting the quality is done by manipulating the compr-level variable. It can either be a "deflate" compression or a lossy format such as JPEG. In the latter case, the fidelity level must also be provided. That would suggest the expected fidelity to be 100%, but the highest JPEG

quality can never outperform the deflate compression. Securing the proper amount of disk storage works in favor of the deflate algorithm as well. Creating the TIFF images using such kind of "cautious overhead" can easily lead, in some cases at least, to generating really massive output files measured in gigabytes. Hitting the 4GB size boundary might be a considerable risk which could make the image formally incompatible with the TIFF format itself. The new builds of the libtiff library remove this limitation (and display a firm warning) and when this size is generated, they parse the file correctly. That might not be the case with some external third-party utilities, though. In fact the pyramid images can be also generated by the ImageMagick suite. The process is not multithreaded and was significantly longer than the processing performed using VIPS. Curiously, with the same initial parameters, the files generated with both suites were not identical. The reasons behind that difference have not been investigated, yet they pose an interesting technical question. Converting a single image into the desired form usually takes 5 to 10 seconds on a testing workstation, never longer. It should be noted that the time of image preprocessing does not impact the responsiveness of the entire system, as it is only performed once.

3.5. Working with the set

Assuming that the image set preparation is complete, the images can now be retrieved with the application of the interface provided by iipsrv. The server accepts the request in a usual manner of a URL with commands and parameters. On the test machine it is exposed by the *http://chinchou.localdomain/a2/iipsrv.fcgi* access point. The instructions are provided with the URL after the question mark, and are separated with the ampersand (a well-known and entirely expected syntax). The following list outlines the most important parameters accepted in a iipsrv request (revisited and extended by Husz, 2008).

- CVT returns the image
- CNT sets the contrast factor
- QLT overrides the output quality
- FIF contains the image URI path
- INV inverts the image
- GAM applies the gamma correction filter

There are obviously multiple other iipsrv directives, all of which are thoroughly described in a comprehensive documentation.

3.6. End Point

A staggering variety of features provided by iipsrv is not directly visible to the end user without connecting it with the interface. There are few utilities created to browse high-resolution images via the web browser, one of them is OpenSeaDragon, written in JavaScript. This simple tool takes pyramid tiled images and serves them in a manner similar to the one known from the "online maps". This allows the real-time zooming and smooth scrolling, thanks to data redundancy. It does not support TIFF as an input, though - one of the problems that were approached by Rollus (2014). It expects the image to be in a DeepZoomImage (DZI) format, invoked by a *?DeepZoom* = command. VIPS indeed does support converting to DZI directly, but this feature comes with an array of shortcomings. The one that is visible almost immediately is the speed of preprocessing. Creating a standalone set of ready-made DZI tiles takes significantly more time than creating a pyramid TIFF.

The DZI-type image is acutally an XML-like description file followed by the independent sets of JPG images, where each set consists of tiles which divide the image using different granularity. Tiles are stored in generic JPEG files.

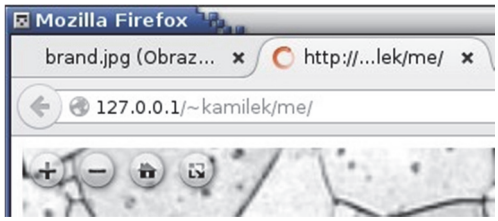


Fig. 1. *OpenSeaDragon Controls*

When a pyramid TIFF file is prepared the iipsrv can deliver it in a deep zoom format on the fly. It eliminates the need of having a big set of little separate files for each images. The advantages of this will be discussed at the performance analysis. Dynamic recreation of the image makes the iipsrv + openseadragon a considerably responsive, yet clean, interface for visualizing and browsing of the high-resolution images.

3.7. Copy protection and security

The problem of security concerns both the server and the actual data. As far as the server is concerned, there are no reasons to apply any complex security precautions that extend over the web development

best practices, like securing the path strings, keeping permissions and sanitizing input.

Although, the data served by iipsrv are a different issue, as they might include the copyrighted material. Formally, the content served by the constructed system is a property of the Faculty, and the availability is limited to the staff and students. That should imply the proper scope of "allowed usage" but does not deal with corner cases.It means that the access boundaries do not necessarily provide the requested protection level as the delivered content might still be freely copied. The server provides the watermarking functionality that can be used to uniquely tag the images, but this feature is mostly used for specially-prepared materials bound with explicitly strict copyright licence. In case of internal usage, this kind of precaution is not specifically requested, mostly because of public domain or non-specified copyrights. Sharing the access to the database with the faculty beneficients and assuming its fair use sets the same agreement on content delivery as that applied in other sources.

When the prepared iipsrv system is used during outside events such as The Faculty Open Day (an outreach promotional public event) and is considered to be a promotional material, then the watermarking is an adequate and suitable configuration. The reasons behind it, though, are entirely non-technical and are not related to the direct security concerns.

4. PERFORMANCE STUDY

The initial work was to create a proof-of-concept tool that offers 25 photographic high resolution pictures and 100 generated outputs. For the reasons outlined before, it was not possible to quickly retrieve the photos in a form that would allow to populate the system with it. Creating and prioritizing the image preservation and circulation rules is in progress.

For the network latency and system responsiveness, two images were selected:

- 2560x1920 sRGB 8-bit JPEG image of 99% quality from One Eighty Degree Material Testing Facility; sized 3.785MB (figure 3),
- 4032x3104 gray scale (256 colors) 8-bit JPEG image of 100% quality, showing a microstructure of translucent ceramic Al_2O_3, thermally etched with purity 99.9%, in reflected light; sized,
- 3.962 MB (figure 2).

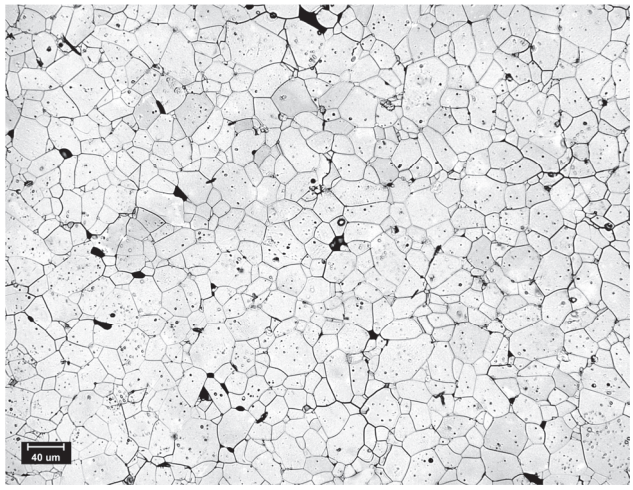


Fig. 2. *"Etch" - one of the images in the database, a high-resolution structure of pure Al2O3 ceramic.*

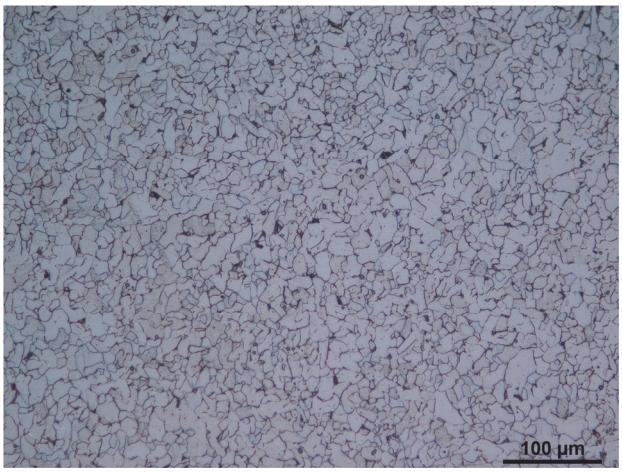


Fig. 3. *"Brand" - another image from the database, a sample steel from the One Eighty Degrees material testing corporations.*

The following numbers will be mostly based on tests performed on those reference images. Due to the dynamic nature of the data exchange in the project, initial performance investigation was performed on a workstation computer, with Core i5 i5-2520M with the clock at 2.5 GHz, 16GB RAM and a Plextor PX256-M6 SSD Drive. The potential performance degradation, related to the Solid State Drive cell life cycle was not included in the analysis. Nonetheless, the problems of cell wear level, inevitable degradation of the medium (alongside with the methods of managing it) and potential write amplification should be examined before deploying a large scale IIP solution based on SSD. There is no direct method of assessing whether the solid state drives are a better choice in the long-term, largescale amd heavy duty applications, but the issue is a matter of active scientific debate. A thorough and promising analysis has been published by Zhao et al. (2015).

4.1. Pre-processing analysis

The input of a pre-processor is a JPEG file stored on physical media to be converted by VIPS into the pyramid TIFF. The command *time vips im_vips2tiff brand.jpg brand.tif: deflate, tile:64x64, pyramid*, on a middle-end Core i5 device returned 3.184 seconds. The process of conversion is not multithreaded. All cores of the machine processor are used thanks to the automated bulk conversion. There are no I/O lock-ups during the process, as it is not disk-intensive (single read, single write per file). Afterwards, the size of the files derived from the reference images was compared with the originals, the most notable examples are presented in table 1. There is no clear correlation between the input JPEG and the produced TIFF file.

Table 1. *Reference file sizes in bytes and percentage of the original size after the pre-processing*

Filename	Original	Processed	% Orig.
brand.jpeg	3785107	14316518	378.23%
etch.jpeg	3961718	11770132	297.1%
colloid.jpeg	2724570	15490128	568.5%

In comparison, the static DZI files made instead of the TIFF pyramid, are *smaller*, but it does not provide higher responsiveness. When the built-in Mozilla profiler is used, it can be observed that due to the seek times, responsiveness of a viewer with pre-defined DZIs does not always outperform a viewer with a backend provided by iipsrv and pyramid TIFFs. It became more visible after heavier loads, but the matching results are not easily-reproducible. Storing multiple smaller files is also more problematic than bigger "blobs", for example in terms of media copying and backup.

Moreover, using the iipsrv backend opens a possibility of connecting with the memcached project. Memcached caches the frequently-used objects in RAM, significantly improving the responsiveness of an IIP system especially when the requests are similar and/or database is relatively small leading to repeated, similar requests. This is precisely the scenario for the didactics use case.

4.2. Request handling

Analyzing the raw network traffic, without any server load and not counting the communication delay, the raw number present themselves as follows:

- The file brand.jpg consists of 3785107 bytes.
- The request for brand.jpg rerouted through the HTTP web server, consists of 5046804 bytes (133% of the original image size). This is called "HTTP overhead" and occurs naturally. The file saved from such a connection will have the original (100%) size.
- The pyramid file brand.tif has a size of 14316518 bytes, which is 378% of the original image(!)
- The same file, delivered via iipsrv as an JPEG image, converted on the fly (with the default settings), and *saved* locally (HTTP overhead not counted) has 1355802 bytes which is 9.47% of the TIFF image
- If the delivery is set to be DeepZoomImage and generated on the fly from the TIFF, the network traffic equals 1161718 bytes which is 8% of the TIFF image and, more importantly, 30% of the size of the original, unprocessed brand.jpg

The last query, made with dynamic tiles, takes 127 HTTP requests and actually takes 7 seconds to complete, while the direct image retrieving takes (no local cache) 306 ms and even the iipsrv TIFF-to-JPEG 790 ms, which is still orders of magnitude lower (and, by the way, shows how optimized the iipsrv CGI is to perform this kind of conversion in such a short time). However, while the dynamic tiled image is loading, its map is browseable almost im-

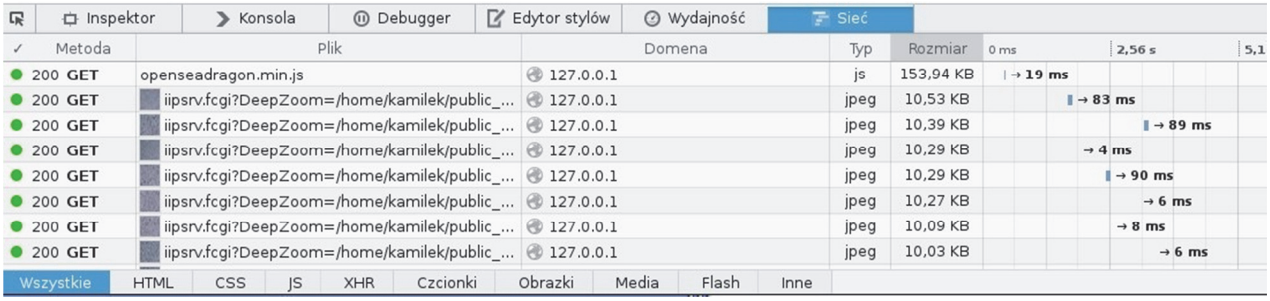
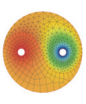


Fig. 4. *HTTP traffic profiler.*



mediately, but with lower quality. It also allows instant zoom,and the missing parts are prioritized to be downloaded first.

This juxtaposition can be seen in a different light when the network latency is taken into consideration. While the previous response tests were concentrating on the size of the request and were performed locally the network-oriented tests show the advantages of the dynamic tile approach.

When the server is busy with handling multiple requests, obviously the speed of delivery decreases. Even a small-scale stress test was reproducing this obvious statement. Performance degradation means that the "ready-made" image might only partially load, in a manner where, for example, only the top 20% is loaded with a high quality and the rest is a blank space. On the other hand, having a part of the image already available, the "tile" endpoints are allowed to browse and zoom the general form of the image and avoid the blank space issue. This is because the tiles do not load linearly but gain the quality over time, so the image is always complete with a varying fidelity on its area, though. The aforementioned observation leads to the statement that the tiled approach is ideal for the classroom visualization and collaborative visual image analysis.

5. CONCLUSIONS

During the creation of a proof-of-concept and brief use-case testing The II Image Server has proved itself to be a valuable tool in networked image visualization, reducing the bottleneck issue that occurs at bulk requests.

Recommendation of an intermediate step in the work with images during the research, namely participating in the creation of an image database, allows wider availability of the images. Common format facilitates the data interchange and builds a collection of redundant images that can be queried for specific quality and/or details via the iipsrv. The TIFF format, along with the VIPS converter make an excellent combination of image manipulating utilities. While the VIPS toolkit is far from perfect (threaded processing), the possibilities it offers are significant. Image pre-processing and conversion to the pyramid format can be quickly automated, thus eliminating the need of additional special image treatment by the researcher.

The iipsrv used in charge of the image database lets the users quickly pick the desired quality of an image, along with the non-standard resolution, gamma correction, contrast, rotation and region. An obtained image can be redirected to further processing without any additional layer, except pure HTTP. Combined with the in-house image processing web service based on ImageMagick and the remote "private cloud" data storage, IIP changes the neglected part of research process - technical details of image storage and processing.

Tile-based redundant image format provides not only the "master images" that can be the source for the case-specific, on-demand structures, but also the data set for a very responsive, accessible browser suitable for the visual analysis. Such a tool is useful both for the researches and the students - who get a dynamic tool for browsing the otherwise static, low-quality or simply unobtainable images. Collecting the initial picture set highlighted the absolute lack of any standardized image processing guidelines and formats. Moreover, the problem awareness was seldom sufficient for applying any changes. The guidelines are being created, but this task is not straightforward, as "facilitating" the image management cannot interfere with the scientific process itself, as it would then prioritize the technicality over the general purpose.

The solution that has been set up provides a nice sample interface for public demonstrations of the scope of research conducted at the Faculty. Despite the pleasant outcome, there is still room for improvement.

6. FURTHER WORK

Over the course of the work on the discussed system, multiple areas for future development have occurred.

The most obvious idea is abstracting out the entire visualization subsystem if it is created for the need of a single project. This is quite frequent with the custom-made cellular automata software. Raw output from the calculations should ideally be used to build the image directly on the server, which would enable the browsing of the result in a quick form, with a broader audience and conserving the storage space on the computing device.

Second field, already started by multiple stagnant and abandoned project, includes extending the iipsrv support for additional formats and conversions. Primarily, fixing the support for OpenSlide, which would extend the processing capabilities for more rare formats, typically used by specialized devices (Rollus, 2014).

Last but not least, changing the attitude towards the image storage and processing. While it is not the main objective of the research in metallurgy and is considered (not without a justification) a technicality, it might become a negligence at the time of developing collaboration and interchange, as well as raising standards in teaching.

REFERENCES

Bertin, E., Pillay, Ruven, Marmo, C., 2015, Web-Based Visualization of Very Large Scientific Astronomy Imagery, *Astronomy & Computing*, 10, 43-53.

Husz, Z., 2008, IIP extension for Woolz sectioning, *J. Applied Mechanics*, 50, 921-934.

IIP Image Server Introduction, 2013, available online at: *http://iipimage.sourceforge.net/documentation/server/*, accessed: 14.12.2014.

Khelou, A, 2008, DOW Partbooks Reader, *MUMT-502 Project Report*, 1.

Murray, P., 2015, JPEG2000 to Zoomify Shim — Creating JPEG tiles from JPEG2000 images, available online at: *http://dltj.org/article/introducing-j2ktilerenderer/*, accessed: 2.03.2015.

Pitzalis, D., Pillay, Ruven, Lahanier, C., 2006, A New Concept in High Resolution Internet Image Browsing, *Digital Spectrum: Integrating Technology and Culture*, 1, 291-298.

Rollus, L., 2014, Cytomine Version Control Repository at Github, available online at: *https://github.com/cytomine*, accessed: 20.12.2014.

Virtual slide formats understood by OpenSlide, 2011, available online at: *http://openslide.org/formats/*, accessed: 7.02.2015.

Zhao, D., Qiao, K., Raicu, I., 2015, Towards Cost-Effective and High-Performance Caching Middleware for Distributed Systems, *International Journal of Big Data Intelligence*, Special Issue on High Performance Data Intensive Computing (1).

WIZUALIZACJA MODELI I OBRAZÓW MIKROSKOPOWYCH W WYSOKIEJ ROZDZIELCZOŚCI Z WYKORZYSTANIEM SYSTEMÓW OPARTYCH O SIEĆ WEB

Streszczenie

Mimo szybko postępującego rozwoju automatyzacji przetwarzania obrazów, w dalszym ciągu zachodzi potrzeba analizy wizualnej. Praca z obrazami oznacza jednakże przetwarzanie danych o znacznej objętości. Metody generowania, zapisu i przechowywania obrazów są uznawane za detal techniczny, co prowadzi do wykorzystywania nieoptymalnych i nieprzenośnych narzędzi, utrudniających wymianę i prezentowanie informacji w formie obrazów. Poprawa tej sytuacji umożliwiłaby łatwiejszą współpracę (między zespołami i oprogramowaniem), otwierając przy okazji szereg dodatkowych możliwości. Podejmując temat związany´ z technicznym zapleczem działalności badawczej, niniejsza praca przedstawia sugerowane rozwiązanie tej zaniedbanej kwestii, oferując system oparty na serwerze obrazów IIP Image, konwerterze VIPS i platformie Apache w środowisku Linux. Opisowi systemu towarzyszy dyskusja nad responsywnością oraz przydatnością w dydaktyce.

Received: April 23, 2015
Received in a revised form: August 15, 2015
Accepted: September 12, 2015