User Interface Design for a Web-based Image Processing and Analysis System

Aleksey Bragin
Bauman Moscow State Technical University
5, Baumanskaya 2-ya, Moscow 105005
aleksey@reactos.org

Alexander Dubanov
Bauman Moscow State Technical University
5, Baumanskaya 2-ya, Moscow 105005
qrcs@mail.ru

Alexander Rechitskiy
North-Caucasus Federal University
1, Pushkin Street, Stavropol 355009
art1st.tm@gmail.com

Abstract

Collaborative web-based technologies, such as Google Docs, Sheets and Slides quickly became popular amongst both individuals and businesses thanks to great increase in Internet accessibility recently. Increase in computational power of personal computers and mobile devices provides perfect opportunity to move more sophisticated applications to web. The results of creating a new web-based image processing and analysis system developed for processing and analysis of images in microscopy are presented in this article.

1 Introduction

It is quite common for vendors of various image producing equipment (e.g. biological, metallographic and other types of microscopes equipped with a digital image acquiring device) to bundle a specific image processing software[AMR04] with it. However, there are some drawbacks in that approach mostly related to either too simple feature set which does not cover basic needs (and an advanced version is not available as a free upgrade) or vice versa, too complex feature set resulting in a steep learning curve for anyone working with the software. Also sometimes further upgrades of vendor image processing software are not free and hence maintenance stops when the support contract runs out.

That was the reason to develop vendor independent image processing software, of which ImageJ deserves much attention. ImageJ was developed by Wayne Rasband of the Research Services Branch, National Institute of Health, in Java programming language. It gained sufficient popularity, as much as 24,000 downloads per month are reported[AMR04]. This software will celebrate 20 years of development in September of this year [Collins]. ImageJ has a number of benefits, such as being free and independent of any hardware vendor, and independent of operating system (Java Runtime Engine, further referred as JRE, which is needed to execute software written in Java programming language, is available for majority of operating systems and computer architectures). It is able to load nearly all existing image formats used in microscopy. A detailed table with the list of supported

In: S. Hölldobler, A. Malikov, C. Wernhard (eds.): *YSIP2 – Proceedings of the Second Young Scientist's International Workshop on Trends in Information Processing, Dombai, Russian Federation, May 16–20, 2017*, published at http://ceur-ws.org.

formats could be found in the paper[Col07]. Additionally, ImageJ provides a plugin mechanism for extensibility, and it is accounted for the popularity which ImageJ got amongst scientists.

However, being developed roughly twenty years ago, ImageJ is not free from drawbacks. Most importantly is its user interface. It is not uncommon for people to spend considerable amount of time trying to find the action they want to perform on an image. Also, ImageJ needs to be installed on a computer, and if there is no JRE pre-installed – install it too. Additionally, one needs to download and install all the plugins he or she wants to work with. It is obvious that collecting and maintaining the add-on files that could benefit a given research program would be prohibitively time-consuming and arduous[Col07].

The popularity of collaborative web-based technologies[HR10] reveals the users' expectations and demand for the Web applications requiring only the browser to run them.

The aim of this work was the creation of a light-weight but sufficiently powerful web application for processing images from an optical microscope. In our opinion, such application should be suitable both for research and education purposes. To fulfill these goals it should be easy-to-use both for experienced and inexperienced users. Therefore, a special attention is paid to the UI design.

2 Web-Based Image Processing

With the rise of available computing power during these 20 years, and with the rise of Internet availability during same time frame, the next logical step in evolution of image processing system would be development of a new system which would be entirely web based, would not require any installation (except for a web browser of users choice) and be accessible from a variety of computing devices, such as personal computers, laptops, smartphones, tablets and other mobile devices. It should also feature a modern, easy to use user interface which does not require steep learning curve.

In this article an emphasis is placed on the user interface considerations, as it is the core of such image processing and analysis system development. It was decided to implement the basic image processing functionality first which is required by all scientific microscopy fields and then work out a way to build extensibility modules. The first analysis module to be implemented is for particle analysis.

Now, the most widespread web browsers computers running them reached such performance level that is sufficient for the image processing to be performed directly inside the web page script code, at least as applied to low-complexity algorithms. That kind of algorithms are easily implementable using client side JavaScript. These circumstances make it possible to perform the image processing mostly at the client side, avoiding server overload.

3 User Interface

Microscopy image processing and analysis software can be classified as special type of software – bitmap image editors. Microscopy image processing is a special case of image processing and hence the required toolset for both of those is quite similar.

For example, people got used to the user interface (later referred to as UI) of image editing software, and they would feel comfortable if microscopy image processing software would have similar tool set and feature set.

The desire for the UI of a web-based image processing software is to resemble the UI of a native general purpose image editing software.

The most successful attempt at creating a web-based general purpose image editing software belongs to Autodesk, according to opinion of authors. The image editor called Pixlr, which this company offers, is the most popular editor (according to Google web search results). Its distinctive feature is that it is available in two version: Pixlr and Pixlr Express. These two versions differ in the tools they provide, and obviously their UIs also differ (see Figure 1).

In our opinion, these applications are the examples of two types of user interfaces, and these types are completely different. Let us survey them in the details.

3.1 Type I User Interface

User interface of the first type (Adobe Photoshop, GIMP, Pixlr) is characteristic for general purpose or universal image editing software which includes many different image processing and image analysis tools and functions, for applying various effects on images, file format conversions, batch processing, extensibility support (macros, plugins, etc).

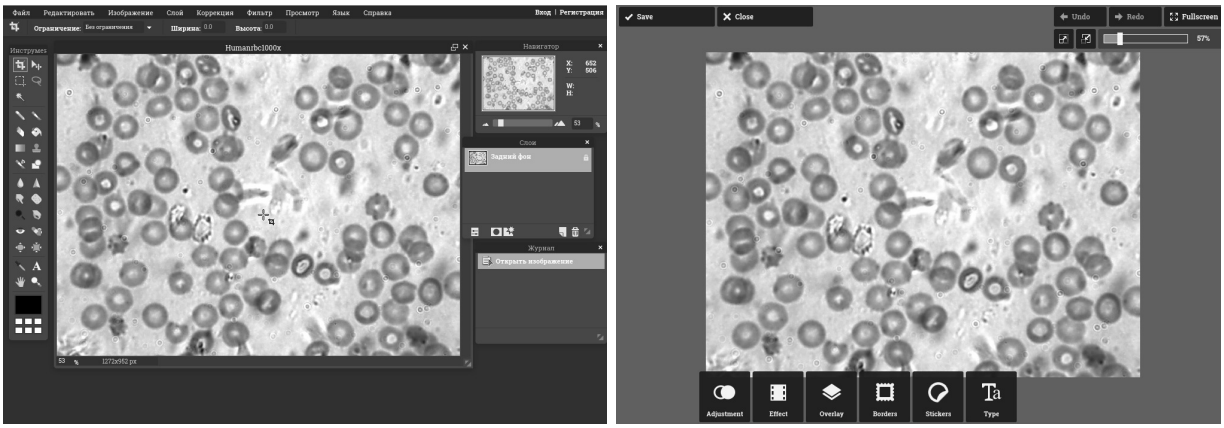

Figure 1: User interface elements location for Adobe Photoshop-GIMP-Pixlr (left) and Pixlr Express (right). The example (*https://commons.wikimedia.org/wiki/File:Humanrbc1000x.jpg*) is loaded.

Elements (windows) can either be independently placed on any part of a desktop or could be placed on special areas (frames). It is also possible to switch between these two modes in some image editing software. In native applications, the toolbox's contents and its location can be customized. This type of user interface assumes use of a personal computer or a laptop, as it assumes precise mouse cursor positioning, large screen, use of both keyboard and mouse. Such image editor is designed for a rather big amount of operations being performed in random order on the same or different images. That provides the flexibility, however, it requires more time to learn the UI and get used to it. Tools are usually grouped by type (for example, drawing, scaling, color correction, etc) in this type of UI, and many of tools have numerous additional parameters offered for users customization.

Downsides of this interface type derives from its advantage: too many tools make it hard for the user to navigate through them and find the right one; too many tools require users theoretical background; tools grouping leads to many additional actions to reach the target tool and finally too many toolboxes just fill up space in a working area.

Web browsers are designed as a document-oriented system. Applying that interface type to a web-based image editor leads to incompatibility problem, as it would lead to a new abstraction layer to represent many windows with images within one web browser tab.

Basing on the reasons above, we considered this type of interface is not suitable for our application.

3.2 Type II User Interface

The user interface of the second type (Pixlr Express) is designed for quick photo processing. The main advantage of that user interface type is the fact that it completely uses working areas window (and screen) to show an image that is good to see the details of a large area on the monitors as well on small screens of netbooks and gadgets. It is a pretty minimalistic interface as is not overloaded with information and tools.

An image editor having such user interface type is suitable for use by a novice user. It could be adapted for touchscreen usage when necessary.

Moreover, it is convenient for development, extension, and support since it does not require an addition layer of abstraction and may be easily implemented using browser API within the document model of a webpage. That is why this type of user interface was considered to be suitable for the application to be developed.

3.3 User Interface For Microscopy Images Processing

It is quite common in microscopy to scroll through various parts of an image at higher magnification, that is, when the full image does not fit the screen. For example, that is typical for finding and counting particles by a chemist or a technician, specific cells by a biologist or lab staff. When microscopes ocular is viewed by human eye such operation is done by actually moving a sample in the microscope, quite often manually.

Obviously, using scrollbars for that purpose is not convenient as it requires additional actions which distract users view. So some other solution should be used instead.

Similar problem was successfully resolved in web based geomap viewing applications, like Google Maps and Yandex Maps. Those also have toolbox icons placed above the image, the image can be scrolled just by simply moving it around with a mouse, and discrete zooming is implemented either by dedicated buttons or by rotating the mouse wheel. These applications are familiar to anyone so it is quite logical to use same approach to view microscopic images.

Besides of that, such approach also better fits the hypertext document style and makes the implementation simpler in terms of HTML and web browser API usage.

After reviewing both types of user interface design and taking in account additional considerations explained above it was concluded that the second type of user interface design enhanced with mouse-based scrolling and zooming fits the goals of developing a web based image processing and analysis (see Figure 2).

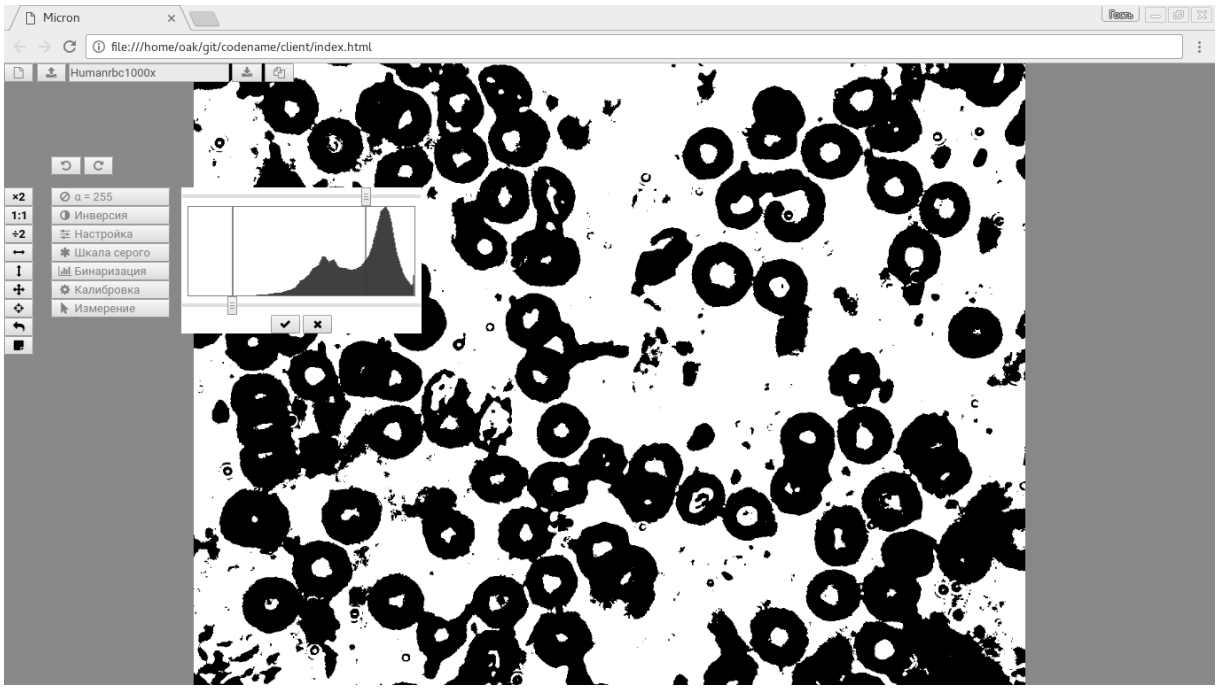

Figure 2: User interface of the developed web based image processing and analysis tool "Micron". The example (*https://commons.wikimedia.org/wiki/File:Humanrbc1000x.jpg*) is loaded, zoomed and preliminary binarized; the binarization tool is open. See the text for details.

Buttons and toolboxes are grouped into three categories: file operations (top left corner), viewing options (left side) and actual image processing tools (left side too). The undo and redo buttons are located above the toolbox. Mutual location of these UI elements is chosen this way to reduce mouse cursor movement required to perform actions. When additional user input is required (e.g. calibration data, brightness, and contrast, etc) a new dialogue window appears to the right of the respective tool icon. File panels and other panels become blocked in this case. However as the user may want to look at fine details of the image, or vice versa, have an overview of the complete image, the scrolling and zooming functions are still available during that operation. Additional information (e.g. calibration ruler) is displayed in the corner of the image and occupies as little space as possible and can be turned on or off.

We asked some specialists using the microscopy and related software in their routine work, to test out the usability of this interface. By their undivided consideration, the interface is really practical and easy-to-use.

3.4 Implementation Features

The application processes raster (bitmap) images. The web API of currently used browsers provides friendly and effective tools to work with images: a canvas HTML element to render the image, a 2D rendering context interface to modify it and an ImageData object for reading and writing raw pixel data[W3C15]. Thus, it was evident to use them in the application as they seemed to be suitable both for the rendering and the location of

an image data undergoing the processing.

Resolution of cameras currently used in light microscopy is between 2 and 10 megapixels (Mpx). Testing Micron application using images with resolution up to 15 Mpx revealed satisfactory performance of the application. For example, applying pixel-by-pixel color inversion to a 5 Mpx image takes 0.8-0.9 seconds in Google Chrome 54.0 and 0.1-0.3 seconds in Mozilla Firefox 50.0 (Intel Core i5 CPU 2.7GHz, 4Gb RAM, GeForce 210 video, OS OpenSUSE Leap 42.1 x86_64).

Some interactive operations like calibration and distance measurement require markers, labels and other objects arising atop of an image in a response to a users actions. In some cases, it is useful to have a scale marker or text labels over an image and to be able to export the image together with them. As user can move, zoom in or zoom out an image to see details and select areas of interest, the images visible resolution may change. Whereas pixels of a raster image may become squares or mix one to another, the drawings over the image should stay clear when zooming.

In Micron application, this behavior is provided by the "sandwich" of two HTML canvases layered on top of each other. The bottom layer contains the raster image and the top layer renders markers, labels and other drawings. The top canvas also receives events (mostly mouse clicks and movements) addressed to both of them. The canvases have identical sizes in the Cascading Style Sheets (CSS)[Col15]. Responding to user's manipulations, the canvases move and resize simultaneously as their CSS size and position attributes change together. However, real size in pixels of the canvas containing the image stays constant, whereas size in pixels of the top canvas is a multiple of the zoom factor. Markers and labels are stored as objects and are redrawn on the top canvas on the UI request. The application logic integrates the canvases into a single entity. Processing methods can work only with one of them or with the both.

In general, the application design corresponds to the Model-View-Controller software architectural pattern[KP98] that seems to be typical for a single-page application (SPA)[MP13].

4 Conclusion

In this article, the key aspects of creating a lightweight and easy-to-use microscope image processing application were analyzed with a particular emphasis on the user interface design. As a first step, such application was implemented as a "pure" web application that does not require a server to run. An appropriate user interface was designed and implemented. The application can already be used for research, technological, and educational purposes. Since its UI can be adapted to touchscreens, it also can be used as a component of other software and documents, for example, for publishing microscopic images collections on the Internet. Authors are unaware of any previous attempts to implement microscopy image processing and analysis software as a web application.

References

[AMR04] Abràmoff, M. D., Magalhães, P. J., and Ram, S. J. Image processing with ImageJ. *Biophotonics international*, 11(7):36–42, 2004.

[Col07] T. J. Collins. ImageJ for microscopy. *2007*, 43(S1):S25–S30, 2007.

[Col15] M. J. Collins. *Pro HTML5 with Visual Studio 2015. Cascading Style Sheets.* Springer, 2015.

[HR10] N. Herr and M. Rivas. The use of collaborative web-based documents and websites to build scientific research communities in science classrooms. In *Proceedings of the 8th Annual Hawaii International Conference on Education*, 2010. URL: *https://www.csun.edu/science/books/articles/Hawaii-international/web-based-docs-general.pdf.*

[KP98] G. E. Krasner and S. T. Pope. A cookbook for using the model-view controller user interface paradigm in smalltalk-80. *J. Object Oriented Program*, 1(3):26–49, 1998. URL: *http://dl.acm.org/citation.cfm?id=50757.50759.*

[MP13] M. S. Mikowski and J. C. Powell. *Single page web applications.* Manning Publications, 2013. URL: *https://www.manning.com/books/single-page-web-applications.*

[W3C15] W3C. *HTML Canvas 2D Context*, 2015. URL: *https://www.w3.org/TR/2dcontext/.*