Pathologist view pattern visualization

Data Source

In traditional tumor diagnosis, pathologist reviewed glass slide with conventional light microscopy, they often used a marker pen to draw the tumor area on each slide for diagnosis, and change the objective lens for different resolutions, then finally has a surgical pathology report for each case.

Digital pathology is an emerging technology that provides an image-based environment for managing and interpreting information generated from a digitized glass slide with lots of advantages against the traditional one. However, there can be 50 slides per patient, each of which is 10+ giga-pixels when digitized at 40X magnification taking the typical prostate case as example. Imagine having to go through a massive images, and having to be responsible for every pixel. Needless to say, this is a lot of data to cover, and often time and space are limited.

We need to reduce information for clinical review. Many researchers focus their work on assisting pathologists with deep learning based on specific topology of the tumor. But the size is still large to feed into the model. We consider from another aspect, if we can learn from pathologist' behavior, to create a multi-resolution panorama with interesting spots in the first place, like the microscopy step. We just need to save those interesting spots with high resolution for pathologist next step review, which is called smart microscopy.

In order to do that, a framework to achieve model observers in digital pathology needs to be developed, to be able to monitor and measure users as they interact with large images when performing specific tasks. To achieve those goal, we built a web-based viewer on top of Openseadragon can be used on any type of multi-scale image. The basic idea for the multi-resolution viewer with the pyramidal format is like the google map, when you zoom out, looking at the whole image, you don't need the details but just the brief perspective, so the low resolution image at the top of the pyramid will be presented to save time and space. When you zoom in to see the image for details, it will provide you only with the corresponding piece of image in a high resolution, like the bottom of the pyramid. It's like you only view a specific part of the whole image for high resolution. The viewer will record the viewing area(the red rectangle called navigator window at top right), zoom level, mouse coordinates and dwell time for each action at the same time, and also annotation with a red pen here, and make a screenshot. The log files will record all of the information for each slides, with the precision to millisecond shown in **figure 1**.

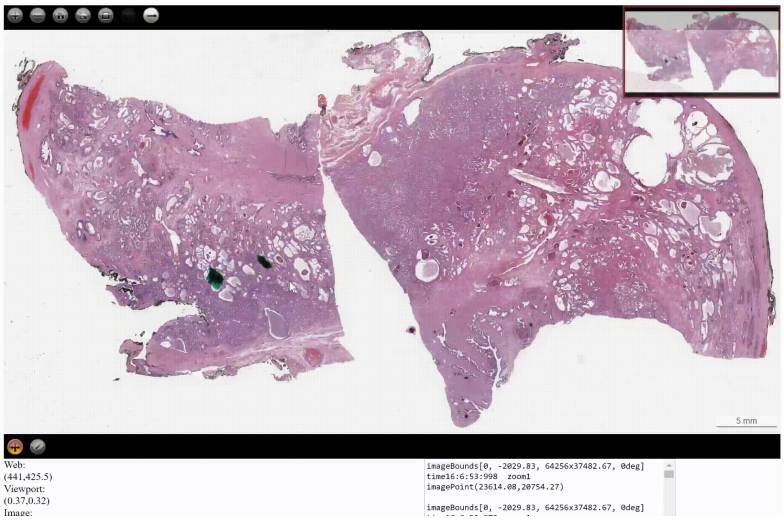

Figure 1: Example of the web-page viewer

To better understand the view pattern for further analysis and feature extraction, we need to find a way to visualize the viewing pattern— using the d3 to reproduce the whole process.

Pre-processing

Since the original images are around 3-4 GB, containing more than 10+ giga-pixels, and we don't need the full resolution to see through every single nuclei during the reproducing, which can also largely slow the processing time. The general viewing method is what we want. The 1AAT.jpg is the downsampled version of the original full resolution image as an example, it downsized 63.28 to the original image, and the ratio will be used to adjust the coordinates in the log files.

To format the log files into the csv we can directly apply with d3, the **preprocess_txt.m** transfers the log files(1AA.txt) into 1AA.csv with following works:

1. Divide all coordinates by the downsample ratio to round them.
2. Convert hh:mm:ss.mmm time stamp into millisecond
3. Calculate the time interval between each action(zoom and pan)
4. Add another column on millisecond level
5. Add index column on each action/row
6. Find the middle point coordinates of the viewport
7. Form all 12 columns information into a matrix
8. Write to suitable csv file

Visualization

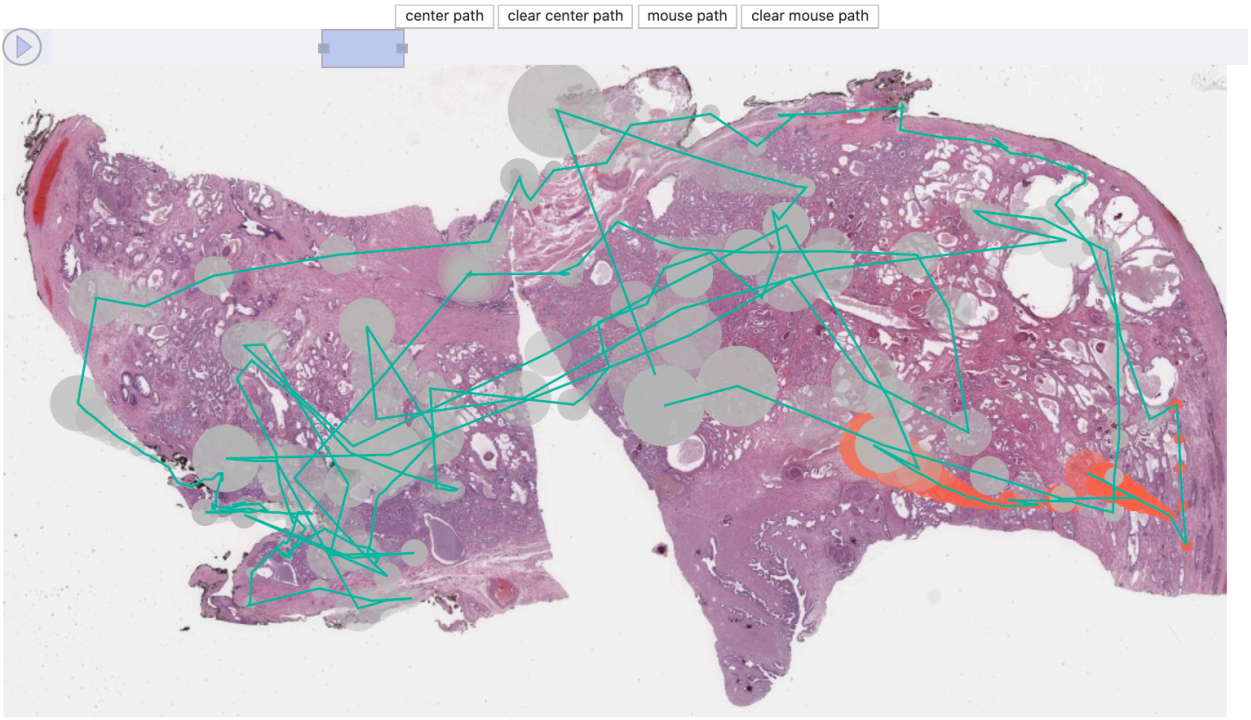

Figure 2: Example of the process reproducing visualization

The visualization is implemented by d3.v5 in **index.html**, achieving the following features:

1. Add the compressed image as canvas.
2. Center path button: click to draw the center of the viewport in sequence(time series), for each point in the path using Monotone Cubic Interpolation with green line.
3. Clear center path button: click to clear the center path or interrupt the unfinished center path drawing.

4. Mouse path button: click to draw the mouse movement during the process in sequence(time series), for each point in the path using Monotone Cubic Interpolation with blue line.
5. Clear mouth path button: click to clear the mouse path or interrupt the unfinished mouse path drawing.
6. Grey circle: encode each viewport during the entire viewing process

(1) the size of the circle represents the the size of the viewport/zoom level, the large of the circle, the larger the viewport, the more zoom out on the action, vice versa. But not representing the actual viewport size, since the viewport size might be so big(the entire image), linear scale the original size to a specified range([0,0.06] in this case).

(2) the opacity of the circle represents the dwell time of the viewport, the longer the dwell/linger time in this viewport, the more opaque/less transparency of the circle, also linear scale to a range [0,1]

7. Slider bar: the slider bar on top of the image canvas representing the total time

(1) the red circle represents the viewport in that selected time interval, the size and the opacity are the same with the grey cirlces.

(2) the range of the selected time interval can be adjusted by changing the left or right brushed area on the slide bar, they can also be dragged to different spot on the slider bar, the red circles will change along with the time interval.

(3) the start and stop button on the left of the slider bar, can automatically move the selected time interval every one second in this case. Hit the start, the slider will go right; hit the stop, the slider will stop.

References

https://bl.ocks.org/alexmacy/eb284831aff6f9d0119b
https://bl.ocks.org/tomshanley/3c49d036610853d380e3fcaf8d3f0b89
https://github.com/RasmusFonseca/d3RangeSlider
http://bl.ocks.org/DStruths/9c042e3a6b66048b5bd4
https://bl.ocks.org/basilesimon/f164aec5758d16d51d248e41af5428e4