Image Alignment: Registration of one or more moving histology images to a reference image

# Overview

In a broad sense, we want to simply align images. Brightfield or IF, affine or deformable, in any sense we just want to spatially link two images, ideally merging them together. There are a series of “modules” that can span across multiple software platforms, depending on how in-depth the alignment has to get. For the bulk of projects, we’ll be using <https://github.com/MarkZaidi/QuPath-Image-Alignment>. As such, we’ll just reference the documentation in that, rather than mentioning in the SOP. Deformable alignment is something that I’m working on, and will be done in MATLAB.

# Prerequisites

A paired set of two or more images to align. Files should be names in a manner where images from the same block of samples all have some kind of identifiable name in common, and have a unique identifier that tells them which stain or section they came from. Both brightfield and IF images can be integrated in this workflow. Any image type supported by QuPath is acceptable, with preference to the .tiff or .svs formats. Make sure the images are rotated to be roughly in the same orientation (<90 degrees apart), which can often be done in the parent image itself. DO NOT ROTATE IMAGES WITHIN QUPATH. It breaks the alignment script

You’ll need QuPath (0.2.3 or later), optionally some kind of IDE (like IntelliJ), a fair bit of RAM to support loading in the images (16 Gb or more should suffice), and MATLAB with the image processing library installed (if planning to perform deformable image alignment)

# Affine based alignment (QuPath-Image-Alignment)

An affine matrix is a 3x3 array, with each of the 9 numbers corresponding to some aspect of a geometric transformation that preserves lines and parallelism. My script operates in two discrete steps: calculating the affine matrix, and applying the matrix to generate new images.

1. Load all images into QuPath
2. Follow the steps outlined in <https://github.com/MarkZaidi/QuPath-Image-Alignment>

# Deformable image alignment

Affine transformations don’t consider local deformations within the tissue. This can be caused by folds, tears, or changes in morphology in sections distal to the reference image. In most cases, deformable image alignment should be avoided, as it can lead to issues downstream (changes in nuclear morphology affecting segmentation accuracy, changes in signal intensity). Only in situations where cutting and staining new samples is not feasible is when deformable image alignment should be used.

1. Load `demons\_registration.m` in MATLAB 2020a or later
2. Set `imgname` to be the path of the 6-channel aligned RGB image produced in the QuPath alignment
   1. Currently configured for the PIMO-ST(H&E) alignment, where the H&E from spatial transcriptomics is the reference image, and the HDAB PIMO is the moving image. Channels are as follows: 1-H&E red, 2-H&E green, 3-H&E blue, 4-HDAB red, 5- HDAB green, 6- HDAB blue
3. Create an empty excel file called `log.xlsx`. Set `excelpath` to point to the location of this log. Any time an alignment is calculated, the parameters, accuracy, and time of alignment are appended to this document
   1. Might update program to automatically create this file if not already present
4. In ` moving(moving==0)=243`, set the value to be whatever the background value of the moving tissue section is
   1. Probably will automatically calculate this from the mode or something
5. Begin optimization
   1. Iteratively calculate deformable image registration by trying out different combinations of the 3 name pair arguments: pyramids, accumulated field smoothing, and number of iterations. See <https://www.mathworks.com/help/images/ref/imregdemons.html> for details
   2. set `iterations`, `pyramids`, and `accumulatedfieldsmoothing` to a range of values you want to optimize across. Generally, 1-1000 iterations, 1-10 pyramid levels, and 0.01-100 field smoothing might be a good place to start
   3. disable writing of the images, during optimization
   4. run `registration` portion of code block
6. Preview best transform
   1. Open `log.xlsx` and find the entry with the lowest MSE and/or highest PSNR, as that’s the best alignment
   2. Run the `registration` block, using only the parameters from that run (should say “processing: 1 of 1”
   3. Run the `apply to downsampled image` block and preview output
7. Apply best transform to whole image (TBD)
   1. **Option 1: apply low res dmap to low res image, scale up image**
      1. **Easy, but loss in img quality**
      2. **Guess its fine for rough annotations and mean intensity…**
      3. **This is currently what I’m using for the PIMO-ST alignment**
   2. Option 2: use parameters generated from the alignment optimization for WSI alignment
      1. Can’t get it to work, out of memory
      2. Troubleshooting with MathWorks support staff. More RAM may be a solution
   3. Option 3: somehow use block proc to apply the low res dmap to the original PIMO section
      1. I’m able to resize the dmap to the original size. Was able to save it as a 11gb .mat file
      2. Data can either exist on disk, in memory, or a mix of both. One way would be to load both the image and dmap, use blockproc to apply imwarp, and write the output directly to a .tiff
         1. Would be even better if I could keep both inputs on the disk, but I don’t think .mat files can be accessed in a block-like manner. Theoretically could write out as two separate grayscale double tiffs, but makes things more complicated