Guide to My Work

Things to copy paste for ease of file transfer:

- source /data/$USER/conda/etc/profile.d/conda.sh && source /data/$USER/conda/etc/profile.d/mamba.sh (grabs the source for your own mamba)
- TO PUSH A FILE TO BIOWULF: pscp [file name of file from personal computer] [your nih id]@helix.nih.gov:[the path to the folder you want to put it in in biowulf]
    - Ex. pscp BatchAnalyzerNOGUI.py tranne@helix.nih.gov:/data/tranne/hitips-nicole/HiTIPS
- TO GET A FILE FROM BIOWULF: pscp -r [your nih id]@helix.nih.gov:[path to the file in biowulf] .
    - The dot at the end is very crucial 🙂
- Also make sure you're running HiTIPS from inside the HiTIPS folder

Open a terminal window and request a GPU node:

FASTEST GPU: Volta 100, 16GB VRAM.
Biowulf ONLY has 8 nodes with v100 chips and THEY ARE SELDOM AVAILABLE DURING PEAK WEEKDAYS.

sinteractive --constraint=gpuv100 --gres=gpu:v100:4,lscratch:500 --cpus-per-task=32 --mem=120g -t 1-12n

FAST GPU: Pascal 100, 16GB VRAM

sinteractive --constraint=gpup100 --gres=gpu:p100:4,lscratch:400 --cpus-per-task=32 --mem=120g -t 1-12

Good GPU, Kepler 80, 12GB VRAM ---
sinteractive --constraint=gpuk80 --gres=gpu:k80:4,lscratch:700 --cpus-per-task=32 --mem=244g -t 1-12

Use this command if you ARE NOT doing deeping learning
sinteractive --gres=lscratch:700 --cpus-per-task=32 --mem=244g -t 1-12

MUST CHANGE SEVERAL THINGS ABOUT MAIN BATCH ANALYZER:

- Overwritten parts (places where he accidentally overwrites variables, pretty sure I marked them in my code (?) (actually this is in Analysis, not batchanalyzer) but look for #CHANGED around where the laplacian of gaussian is
- Check for that place where the folder wasn't initialized (forget where it was, but there was one place it was saying directory does not exist

    - well_spots_output_folder = os.path.join(self.output_folder, 'well_spots_locations')
    - Line 280 ish in BatchAnalyzer, there isn't a place initializing it
- The location of the bayesian track file is wrong -> changed it
    - Marked with #CHANGED, now it looks for something named btrack etc in the folder rather than a hardcoded pathway

Main things I worked on:

- Headless version
    - Based on the arguments in terminal, it will call a different version of the code
    - If there is only one argument (ex. python HiTIPSNOGUI.py), it calls the GUI version of the code
    - If there are two arguments (ex. python HiTIPSNOGUI.py ./input_path/input_file) it calls the headless version of the code on one plate
    - If there are three arguments (ex. python HiTIPSNOGUI.py ./input_path/input_files multi) then it will call the headless version of the file on each of the input files in the csv
        - The path here leads to a csv with the paths to the different input files listed next to the experiment names
        - All the information needed to run the program is contained in an input file (metadata file name + path, output dir, params, etc)
    - Made three new NOGUI files (analysisNOGUI, HiTIPSNOGUI, BatchAnalyzerNOGUI) and a couple of new classes (parameters_class, config_to_input, metadata_trimmer)
    - All the parameters are now pulled from parameters_class, which initializes differently based on whether the GUI or headless version is called
    - Some big things of note are where the BatchAnalyzer checks for things like the nuclear methods, because without the GUI widget format, I had to check if they were equal to strings rather than indices. It's noted in the comments.
    - To look for changed places, I usually have either a note at the beginning of the function, or something like #edited or #CHANGED in front of the line.
- Progress-safe counters
    - Implemented a couple progress-safe counters, one for nuclear masks and one for spot distances
    - Can probably (?) make the same thing for other parts
    - Class for the counters is at the top of BatchAnalyzerNOGUI
- Prepopulation of input file
    - A standalone file (config_to_input) run straight from terminal
    - Format: python config_to_input.py [path to config file] [experiment name]
    - Will prepopulate an input file with the config file and then save it to a file called [experiment name].py
    - Make sure you run it in the same folder as the original input_template.csv, the code looks for a file by that name to prepopulate

- Metadata file trimmer (as a part of the resume from interrupt feature)
    - Class is defined in a separate file called metadata_trimmer
    - Uses minidom to check the attributes of each line(node) and if they match up to the row/col/time/fov of the image that just ran, they are deleted
    - After deletion, saves the edited xml file to the same file it read from
    - Implemented in a process safe way in BatchAnalyzer using the built in locks of the multiprocessing package
        - Currently only implemented for nuclear masks
    - Not sure if the rest of the code will work after the nuclear masks are finished running if using a trimmed metadata file
    - Further work might include making it so that the nuclear mask generation can be skipped
    - Also, in order to implement the process safe locks, edits had to be made to the process initialization part of the code, but they're behind if statements checking for if the GUI or headless version is being run.
- Multi plate analysis
    - Explained in the first part of the headless version, but it basically runs in a for loop in HiTIPSNOGUI.py for each of the input file pathways listed in the csv
- Input files
    - Lots of stuff in these
    - Make sure you include the input template in the folder that you're running config_to_input in
    - Contain all the stuff from the GUI plus output dir, metadata name/path, etc
    - There are two that I include, one excel and one csv. CSV is for use in config_to_input, and excel is for users to change around and then save the final file as a csv before using it in HiTIPSNOGUI
    - HiTIPSNOGUI only takes csv input files

Thoughts on the final integration:

- I think overall it should be pretty okay, except for how I don't actually know how the GUI version works with how different widgets and windows open
- But all the parameters are now stored in the parameters_class file, and more can be added as needed
- After all, even now in my HiTIPSNOGUI class, the Analysis and BatchAnalyzer classes I instantiate are the NOGUI classes and the GUI still opens from the HiTIPSNOGUI class (to find where they instantiate, look for #CHANGED)
- I think the final integration will be just tweaking the NOGUI files

Questions:

- Where is the spot tracker?
- Didn't get to test a lot with the deepcell tracker