OPERATING PLAN FOR ENGINEERING DESIGN PROJECT

WILL KENT, JACKSON DOUGHERTY, KEIR ADAMS

Project Schedule

There are several principle steps to the project. We first need to finish work on image processing, thresholding, and contouring, so that we have the data to begin machine learning development and analysis. From there we will collect statistical information on all of the images available to the project, as well as research and develop machine learning methods to analyze the spray images. After developing the machine learning code, we will test it on the images while, concurrently, researching physical information that can be determined from the results of our machine learning analysis. This work should all be completed two to three weeks before the end of the project. Finally, we will use the machine learning methods developed to learn as much about the spray images available to the project and try to provide both quantitative and qualitative analysis on the performance of the spray nozzles. By this time we also will have packaged the code such that it will be easy for those who do not have programming experience to use and documented sufficiently for future software developers to modify and augment the project. Figure 1 illustrates this process and how all of the project nodes interact with on another.

Below you will find a week by week schedule on how the project will proceed. It outlines week by week what the team will accomplish, as well as deadlines for key materials like presentations and reports. In addition to the weekly tasks mentioned in the below schedule, the design team will meet with Xiaoying Liu one to two times a week, and meet with external mentors at a frequency to be determined.

Autumn

Week 4

- Deadline: Draft of SOW and OP
- Finish thresholding work
- Get details on cloud file storage
- Crop all images to prevent background noise
- Meet with Doug MacAuley on Monday
- Contact external mentors
- Deadline: Finalized SOW and OP, presentation of OP to Xiaoying (5min)
- Revise SOW and OP based upon feedback from mentors and Xiaoying

Week 5

Date: October 20, 2019.

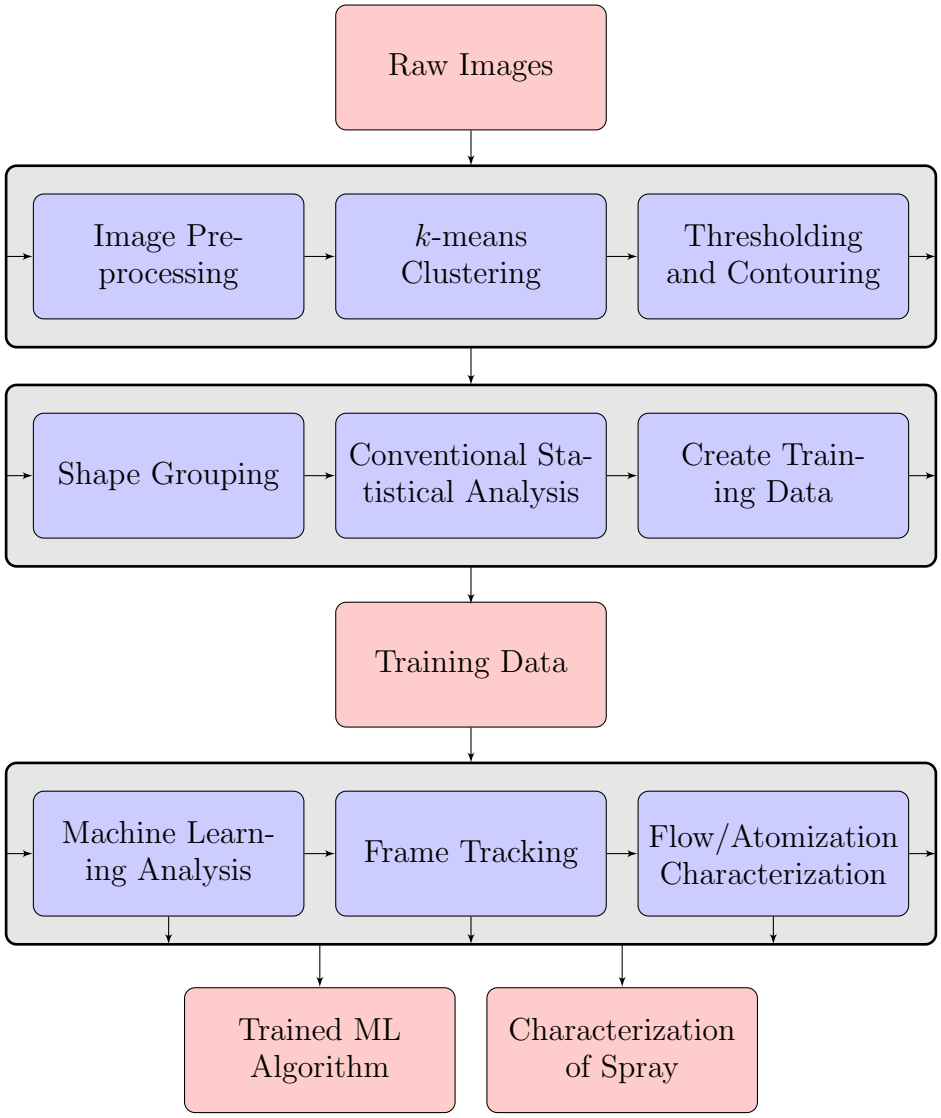

Figure 1. Project flow: blue nodes are processes or functions, red nodes are outputs and/or inputs

- Investigate *openCV* methods for collecting information on shape, size, density (color), position, and time distribution of droplets
- Generate statistical information, namely distributions of information mentioned in previous bullitt point.

Week 6

- Complete droplet cataloging and analysis from week 5
- Commence research on machine learning methods
  - Keir: neural networks, ridge regression
  - Will: Random forest walk and other unsupervised methods
  - Jackson: Support vector machines, genetic algorithms

Week 7

- Decide on most promising machine learning methods to pursue
- Create python frameworks for each method (see Figure 1 **Machine Learning**)
- Start researching frame tracking and temporal data collection
- Start researching physical information to track

Week 8

- Hyper-parameter optimization
- Continue research into physical information for tracking
- Continue researching frame tracking and temporal data collection

Week 9

- Continue hyper-parameter optimization
- Continue research into physical information for tracking
- Develop and code frame tracking and temporal data collection
- Begin working on mid-project presentation and progress report

Week 10

- Finish hyper-parameter optimization
- Continue research into physical information for tracking
- Develop and code frame tracking and temporal data collection
- Present mid-project presentation
- Finish progress report

Week 11

- Revise mid-project presentation based on class feedback
- Present progress report and mid-project presentation to external mentor

Winter

Week1

- Continue research into physical information for tracking
- Continue development and coding of frame tracking and temporal data collection

Week 2

- Continue research into physical information for tracking
- Continue development and coding of frame tracking and temporal data collection
- Augment machine learning methods to accommodate frame tracking and physical information

Week 3

- Present progress report to external mentor
- Continue research into physical information for tracking
- Finish development and coding of frame tracking and temporal data collection
- Test code for frame tracking and temporal data collection

- Continue augmenting machine learning methods to accommodate frame tracking and physical information

Week 4

- Finish research into physical information for tracking
- Improve and continue to test frame tracking and temporal data collection code
- Continue augmenting machine learning methods to accommodate frame tracking and physical information
- Prepare mid-quarter presentation to the mentors

Week 5

- Continue improving and testing code for frame tracking and temporal data
- Test physical information tracking code
- Present mid-quarter presentation to the mentors

Week 6

- Continue improving and testing code for frame tracking and temporal data
- Improve and continue to test physical information tracking code

Week 7

- Finalize code for frame tracking and temporal data
- Finalize physical information tracking code
- Begin packaging project for ease-of-use and further development
- Begin final analysis of provided spray data

Week 8

- Finish final analysis of provided spray data
- Finish packaging project for ease-of-use and further development
- Begin final report for design project

Week 9

- Continue work on final report for design project
- Prepare final presentation to the class

Week 10:

- Complete design project
- Finish final report
- Deadline: present final presentation to the class

Week 11:

- Deadline: final report and presentations to external mentors

Organization of Project Materials

Both for the project to proceed efficiently and for the project to continue after the allotted two quarter timespan, the code and documents need to be organized effectively and the project materials must be documented effectively. There are three main structures that we have mapped: what the final project should do, what files will be included and how they will interact with one another, and how the files will be stored. Previous sections have outline what the project will accomplish and how the different portions of the project will interact with each other (see Figure 1 and ...). Now we will examine *how* the different portions of the project will be actualized and organized.

As should be clear from the discussions of image thresholding, clustering, and machine learning algorithms, there are many different moving parts to this project. Multiple programming libraries will be used to generate plots, statistical information, machine learning settings, as well as many outputs. The heterogeneity of the different sections of the project and the fact that three programmers will be working on the same code at a given time requires a highly coordinated effort. If the group is not coordinated, it will be difficult to test code, combine work from multiple individuals, and finish the project in a manner that allows for further development. To prevent these undesirable outcomes, we have outlined two structures: what kinds of program files will be present and how they will interact with one another, and where files will be stored and how they will be organized.

It was decided amongst the group to code the project entirely in Python because of the access to machine learning libraries that it proves as well as the common basis everyone in the team shares. Figure 2 shows the different kinds of files that will be used and how they will interact with one another. The ultimate goal of the project is to develop the code rigorously enough to the point where a user will only need to specify the address of files in an input/parameters file (colored yellow in Figure 2) and execute a python file (*run.py*) to run analysis on image files. This run file will read information from the input file, find the images, create directories for output, and then pass all of the necessary information to another file (*run_analysis*) that will run analysis on the images. We will call these files *execution files* (see (2) in Figure 2).

The analysis file will call upon functions from several *main files*. These files include the programs for k-means cluster analysis and thresholding, documentation of results and settings, image processing, and machine learning (see (3) in Figure 2). The programs within the *main files* will call upon functions written in *auxiliary files*. These auxiliary files will include process specific auxiliary files, such as *thresholding_aux.py* or *ML1_aux.py*, but also contain files with more general functions, such as reading and writing images into files, plotting images, and documenting results. This structure is important because using *auxiliary files* will allow the *main files* to be coded in fewer lines and begin at the top of the files. This will make the code easier to debug, documents, and significantly easier to follow both for team members and any future developers. The *auxiliary files* will also include python files for every single instance of plots, so that plots from every part of the project can be automatically regenerated if on node in the project is changed significantly. Finally, personal development files for each developer will be within the auxiliary files so that developers can

test code before adding it to the project functions and potentially damaging other work on the project.

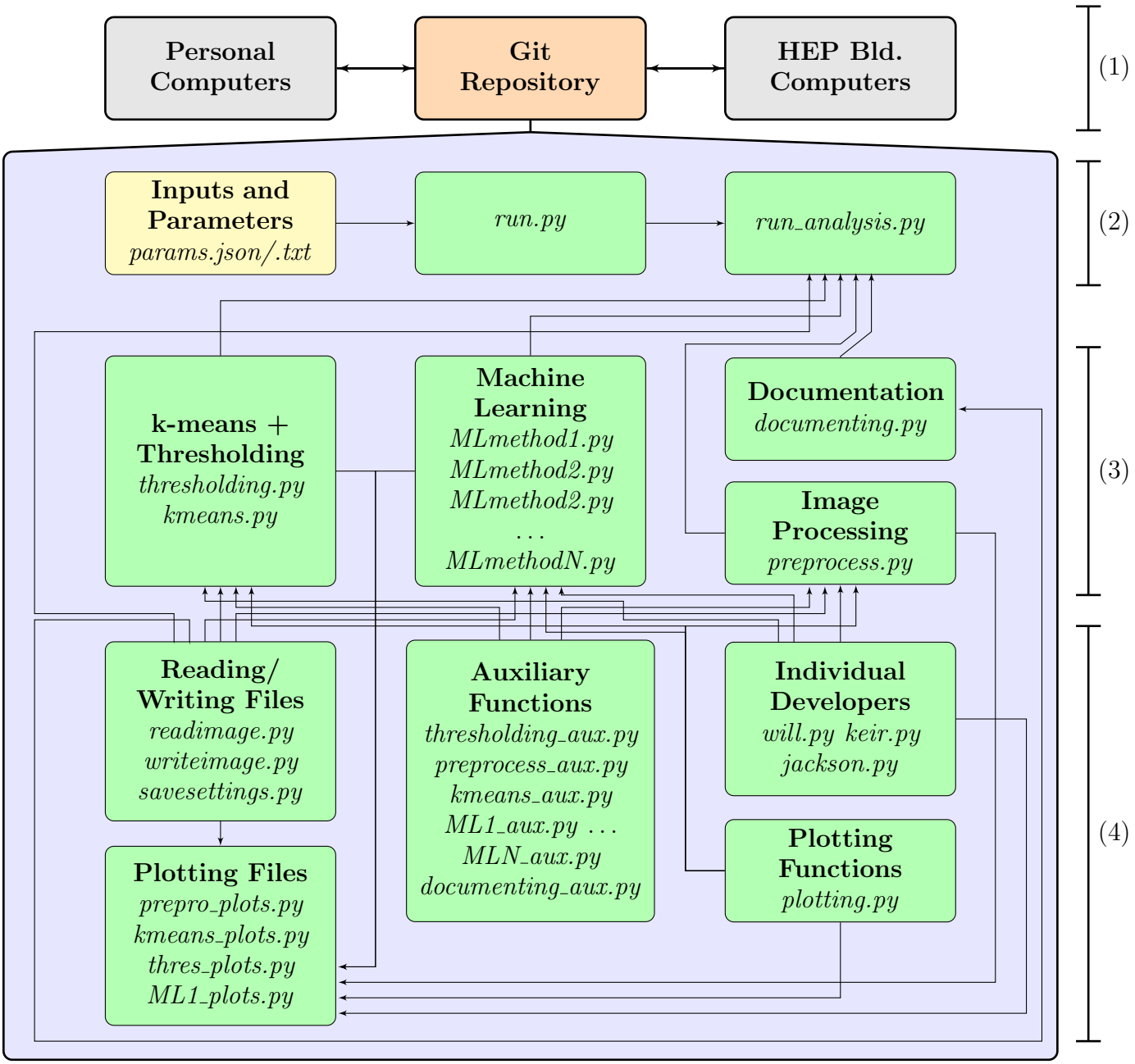

Figure 2. Code Structure: (1) Computers/storage (2) Execution Files (3) Main Files (4) Auxiliary Files. Each green node is a python file or a group of python files.

In order to facilitate working on the project with multiple software developers, we will use git, an open source distributed version control system, and host the project code on GitHub, a cloud-based git repository. The repository is called `MENGEDML` and can be checked out at the address `https://github.com/wkent2020/MENGEDML.git`. Using git will allow

the team to share work and keep detailed track of each change to the master code (see Figure 3). In addition to keeping track of all changes to the code, git allows for all of the team to develop code on their personal computers and test it on other devices.

```
commit 98c0ea01a3b41bfcd4476183e06813d02e1b9e95 (HEAD -> master, origin/master, origin/HEAD)
Merge: 6f63bf0 2f9b29e
Author: wkent2020 <wkent@uchicago.edu>
Date:   Thu Oct 17 21:35:12 2019 -0500

    Merging croptop with jackson's work

commit 6f63bf042c919fc507b57085edd39763f0c5be14
Author: wkent2020 <wkent@uchicago.edu>
Date:   Thu Oct 17 21:28:33 2019 -0500

    Added varying width segmentation
```

Figure 3. Example git log: git keeps track of every change made to the code, who made the change, and allows users to revert to past version.

One problem facing the project is how files will be *stored* and *where* they will be stored. The files within the git repository will be organized as follows:

Git Repository
- *kmeans_thesh.py*
- *preprocess.py*
- *documenting.py*
- *plotting.py*
- *run.py*
- *run_analysis.py*
- *params.json/.txt*
- ⋮
- *mainN.py*

⇒ Auxiliary Functions [Auxiliary Functions Files]
- *thesholding_aux.py*
- *kmeans_aux.py*
- *preprocess_aux.py*
- ⋮

⇒ Plotting [Plotting files and Data]
- *plotting.py*
- *kmeans_plotting.py*
- *preprocess_plotting.py*
- ⋮

⇒ Plots
- ⇒ Thresholding Plots
- ⇒ Pre Processed Images
- ⇒ ML Method 1 Plots

⇒ Documentation
- ⇒ K-means Docs
- ⇒ ML Docs
- ⇒ Papers
- ⇒ Plotting Docs

⇒ Individual
- *will.py*
- *keir.py*
- *jackson.py*

Because of the available images files are so numerous, we cannot store the team cannot store them on personal computers. Thus, we will store image files on two desktops located in the High Energy Physics Building which can be accessed remotely.