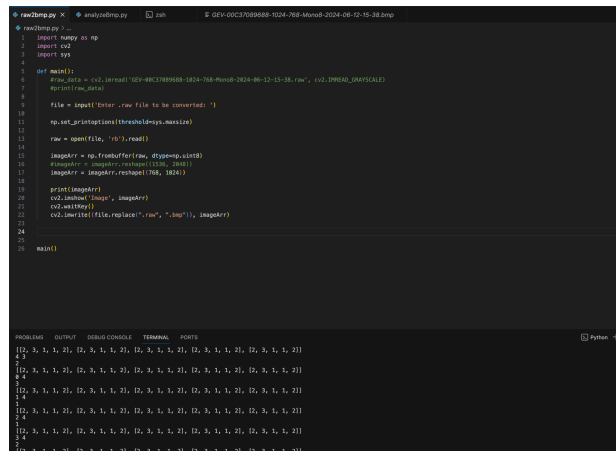


Week of 6-10-24

This week was a bit more of the same; Dr. Yorita provided me with a software package from ~2014 that could be used to perform the analysis on the .bmp files we're interested in. I had to throw together a script (see below) to convert the files we obtained experimentally from .raw to .bmp so they could be used in the program. I got some hands-on experience this past Wednesday with adjusting the focus of a lens used by a remote-controlled camera that was collecting the files, as well as with taking apart a portion of the beamline in order to examine possible sources of contamination (since the collected images were experiencing some severe aberrations due to, presumably, some ions coming in at directions not parallel to the beamline).



```
1 # raw2bmp.py
2 import numpy as np
3 import cv2
4 import sys
5
6 def main():
7     raw_path = cv2.imread('DEV-003708888-1024-768-Mondb-2024-06-12-15-36.raw', cv2.IMREAD_ANYDEPTH)
8     print(raw_path)
9
10    file = input('Enter .raw file to be converted: ')
11
12    np.set_printoptions(threshold=sys.maxsize)
13
14    raw = open(file, 'rb').read()
15
16    imgdata = np.frombuffer(raw, dtype=np.uint8)
17    imgdata = imgdata.reshape(1024, 768)
18    imgdata = imgdata.reshape(768, 1024)
19
20    print(imgdata)
21    cv2.imshow('Image', imgdata)
22    cv2.waitKey()
23    cv2.imwrite('file.replace(".raw", ".bmp"), imgdata)
24
25 main()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PLOTS

[0] 3, 1, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2]

[1] 3, 1, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2]

[2] 3, 1, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2]

[3] 3, 1, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2]

[4] 3, 1, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2]

[5] 3, 1, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2]

[6] 3, 1, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2]

[7] 3, 1, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2]

[8] 3, 1, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2]

[9] 3, 1, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2], [0, 3, 1, 2]

Fig. 2.1: Script used for aforementioned file conversion.

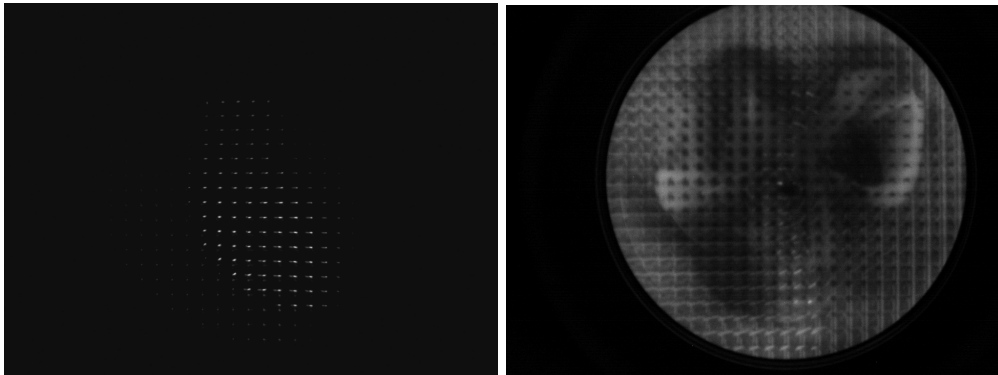


Fig. 2.2: Data from 2014 (left) versus data from this past week (right); this data is fine for calibration of systems, but new shielding for the beamline is currently being fabricated to try to reduce these abnormalities during actual runs.

The Opal simulation is running, but it seems to be generating data inconsistent with what one might expect compared to the actual beamline of interest. Investigation into this is ongoing, and Dr. Yorita has a copy my input file to look over for sources of abnormalities (since this is my first time working with the software).

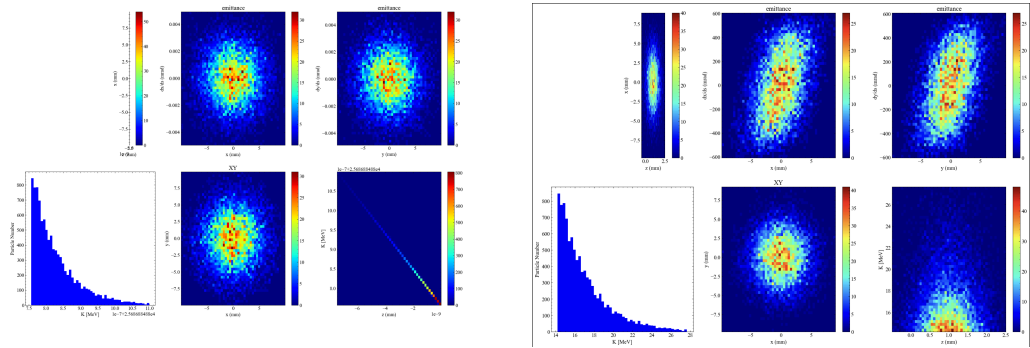


Fig. 2.3: Plots of the phase space of beamlines; the left images are from a test program (“normal” or expected distributions) and the right images are from my Opal input file.

Lastly, I’ve begun work on my own analysis program for use in the experiment in the stead of the C++ program written roughly a decade before. In general, I and Dr. Yorita had sufficient trouble with determining the purposes of all the header files, especially that which was used for plotting data (written using a fairly old software that I’ve never seen before and whose name I don’t recall) and for this reason decided that it would make sense for me to make my own analysis program from scratch not only to improve my own workflow but so that I can provide documentation for future researchers who might decide to use it in the future. The program is in very early stages, but I expect a large portion of its functionality should be completed in the following week (importing and formatting of .bmp and .raw data has already been done, so all that really needs done is mathematical calculation and plotting). Some optimizations might need to be done on the Python code so it can run optimally in real time (this is the ultimate goal) but I expect that this is not impossible.