Review on Instrumentation codes

All the codes are in <https://github.com/unwatcher/sutter-fella-group>

1. PID-controlled heater with pyrometer feedback
   1. The program was implemented in python, GUI developed with the python package pyqt5. PID was realized with simple-pid package [1]
   2. temperature measurements were done with pyrometer with 0-10 voltage output, which is recorded by a KeysightTM 34972A data logger. It is converted based on Temperature = Voltage \* 20, and fed to pid loop.
   3. The data logger is hooked to the python program with NI visa and python package pyvisa. To use it, the IO suite must be installed first, which can be found here: <http://www.agilent.com/find/iosuite>. The visa location is static and can be found in line 171.
   4. The temperature is fed to the PID loop which updates every 0.5s. Temperature also output to csv row every 2s. (line 83- 88).
   5. Plotting preferences for the app can be found in canvas settings (line 42-47)
2. Syringe Pump control
   1. The program was implemented in python, GUI developed with python package pyqt5, with which two NE-1000 syringe pumps were controlled via COM serial interface. The commands can be found in manual.
   2. The purging mode is set by deleting all phases and setting the only phase to be dispensing at highest speed allowed for the syringe. The dispensing mode is set that the first phase dispenses a set amount, and second phase is STOP. To see the details of commands and make change see the user manual.
   3. The pumps themselves has a speed limit depending on the diameter you set. Check “Syringe Diameter Chart.pdf” in resources.
3. Lepton camera acquisition
   1. The program for PureThermal 2 board with a FLIR 3 is impleted using the custom python library fork of libuvc provided by purethermal [2]. The hardware hook was modified upon the original code for purethermal 1, for radiometry mode [3]. The modified code has an additional zoomed in view of the substrate, so that it is possible to closely monitor the surface emissivity.
   2. The color profile for the raw image output is in Y16, which has one value per pixel. The values are in range 0-216, and temperature (in Kelvin) = (value)/100
   3. Controls: ‘D’ for record, press again to stop recording. (**double check is this right?)** ‘C’ for exit. The program saves both a colored image and a raw image.
4. Lepton camera image processing
   1. (Lepton\Make Thermal Videos\Make\_thermal\_video.py) The thermal video is made via python and matplotlib. First a colorbar image is made with the desired range of temperature (static, line 13). Then the raw data in selected folder is converted to pseudo colored pictures by shrinking to 8-bit and applying colormap (cv2.colormap). The output image is saved to a subfolder “new” in the selected folder. The output images are combined into a video in the “new” folder.
5. XRD contour plotting and analysis with MATLABTM
   1. (ALS\_analysis/ALS\_plot\_horizontal.m) The data is first calibrated (with alumina? Not sure, but there was one time we just used 14.2 perovskite peak as a reference) and integrated with XMAS software, either along the chi or 2theta. The output “.dat” file is fed into the program, converted to an intensity matrix with x-length corresponding to the 2theta points and y-length corresponding to time frame. The plotting is done by using contourf in MATLAB, and the individual rows in the data matrix can be taken out and fitted in loops, yielding features like FWHM or integration.
   2. The peaks in each time frame are identified and fwhm is extracted for each one. In the beginning of the code a minimum height can be changed to adjust for the background noises. ”Caxis” attribute of the contour can be changed so that more (or less) details are shown in the plot.
   3. The chi data yielded by integrating along chi in XMAS can also be plotted in a similar way. (ALS\_analysis/chi\_plot.m)
6. PL signal captured with Oceanview and exported into csv format.
   1. Ocean Optics QE Pro control software OceanView captured full-spectrum 185 - 1100 nm data at 1 second intervals while custom python code read the output data files and updates a false color contour plot of intensity to wavelength. The plot was implemented with real-time animated plot and updates sufficiently close to real time so that operator could adjust antisolvent delivery timing and heating parameters while directly witnessing the effects.
   2. A contour plot code is written in MATLAB for this specific format (PL\_heatmap.m) under the PL/PL plot folder. It allows for plotting in wavelength, eV, or both. There might be some issues with MATLAB earlier than 2018.
7. PL contour plotting and real-time PL preview
   1. PL/Live\_heatmap.py: Real time preview of PL signal was implemented with animated matplotlib.pyplot object. It scans the target folder for files and updates the plot. Currently it is not adaptive in color range, which should be added in the future.

Time sync was generally done with syncing the computer’s system time with the LBL NTP server (ntp1.lbl.gov) every time a program is launched.

[1] <https://pypi.org/project/simple-pid/>

[2] <https://github.com/groupgets/libuvc>

[3] <https://github.com/groupgets/purethermal1-uvc-capture/blob/master/python/uvc-radiometry.py>