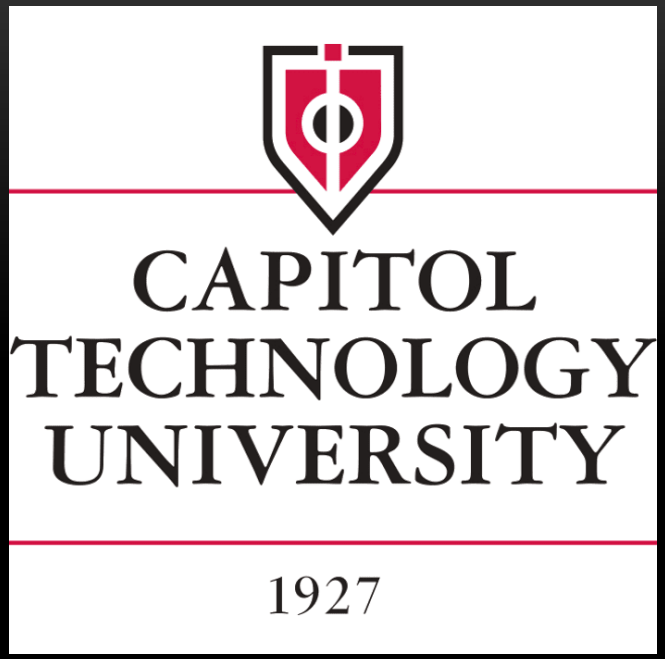
MDSGC SUMMER EXCHANGE PROGRAM: ROCKET PAYLOAD DEVELOPMENT

Sam Lawson

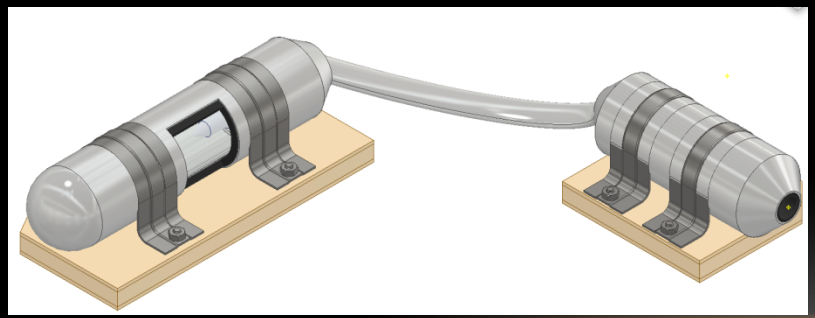
ABOUT ME

- School: Capitol Technology University
- Major: Astronautical Engineering
- Year: Junior, graduating May 2020
- Interests: Software Design/Programming, 3D Modeling, Spacecraft Design



PROJECT SUMMARY

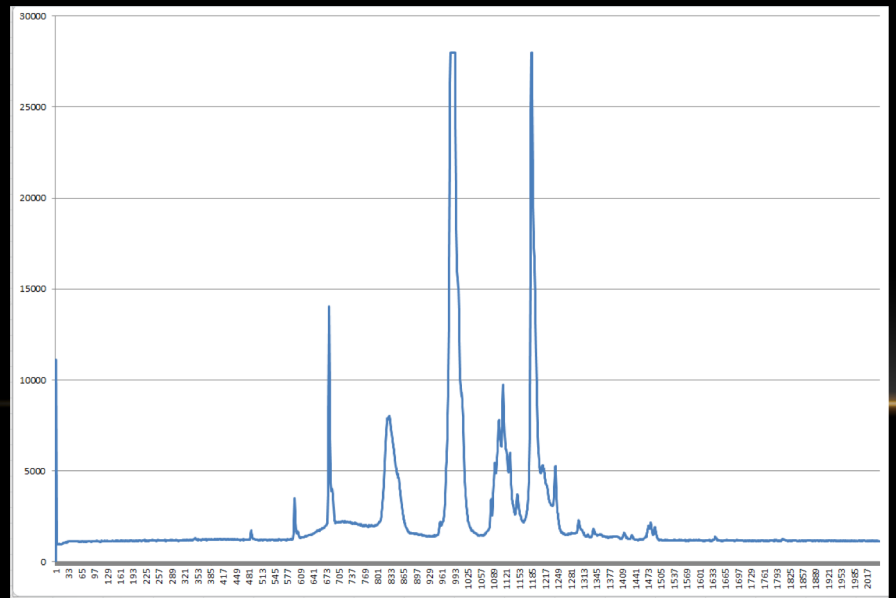
- Project Title: Air Breakdown and Plasma Spectroscopy at Low Pressures and High Flow Rates
- Mentor: Dr. Romero-Talamás
- Purpose: Design, build, and test a spectroscopy experiment for a sounding rocket payload



THE EXPERIMENT

- Purpose: Seeking sodium in the upper atmosphere

- Transformer steps 7.4V up to ~40000V, allowed to arc through air
- Arc ionizes air, causing various wavelengths of light to be released
- Spectrometer gathers light and reports relative intensity of each wavelength
- Spectral lines then analyzed to determine what elements were present in the arc



Example spectrum of a standard fluorescent light

MY OBJECTIVES

- Project broken into three parts for three interns. My objectives were as follows:
- Develop a software interface to allow a Raspberry Pi to control and retrieve data from the Flame Spectrometer
- Verify that the spectrometer can function in high vacuum (~5e-5 Torr)
- Test with the spectrometer and spark gap to identify the optimal integration (stare) time to get our data
- Get a final calibration for the spectrometer before flight
- Assist with other objectives as needed

INTERFACING WITH THE SPECTROMETER

- Used an API called SeaBreeze to create Linux based programs for the spectrometer
  - Code was written in a combination of C and C++
- Created a program that retrieves a new spectrum every 3.5 seconds and stores that data in a unique text file where it can later be accessed and transmitted

```
    time(&rawtime); //Updates rawtime
    timeStamper = localtime(&rawtime); //Converts rawtime to local format and loads into timeStamper
    snprintf(timeStamp, sizeof(timeStamp), asctime(timeStamper)); //Puts data of timeStamper into a char array time

    system(turnOff); //Call to python script which sets spark gap trigger pin to 0

    // strip synchronization byte if present (not all devices have one)
    if (raw_length % 2) {
        raw_length--;  // ignore last byte hereafter
    }

    int pixels = raw_length / 2; //The spectrum returns 2 bytes per pixel

    saveToFile(dataNum, pixels, bSpectrum, timeStamp); //Send spectrum to function to be converted and written to f

    free((void*) bSpectrum); //Un-allocate memory for bSpectrum

    return 0;
}

int main(int argc, char **argv){ //Applies settings to spectrometer and repeatedly calls get_unformatted_spectrum_d
    //Parses data into char arrays for OS system() calls
    snprintf(turnOn, sizeof(turnOn), "%s %d", turnOnCode, triggerPin);
    snprintf(turnOff, sizeof(turnOff), "%s %d", turnOffCode, triggerPin);

    system(turnOff); //Call to python script which sets spark gap trigger pin to 0

    int raw_length, i;
    i = 0;

    seabreeze_open_spectrometer(0, &error); //Opens the spectrometer

    seabreeze_set_integration_time_microsec(0, &error, integration_time); //Sets the integration time (see program
    check_error(0, &error, "seabreeze_set_integration_time_microsec");

    seabreeze_set_trigger_mode(0, &error, trigger_mode); //Sets trigger mode (see program settings above)
    check_error(0, &error, "seabreeze_set_trigger_mode");

    raw_length = seabreeze_get_unformatted_spectrum_length(0, &error); //Gets value for raw_length (number of bytes
    check_error(0, &error, "seabreeze_get_unformatted_spectrum_length");

    while(1){ //Repeatedly call get_unformatted_spectrum_doubles()
        getASpectrum(i, 0, raw_length);
        if (check_error(0, &error, "seabreeze_get_spectrum_doubles")){
```

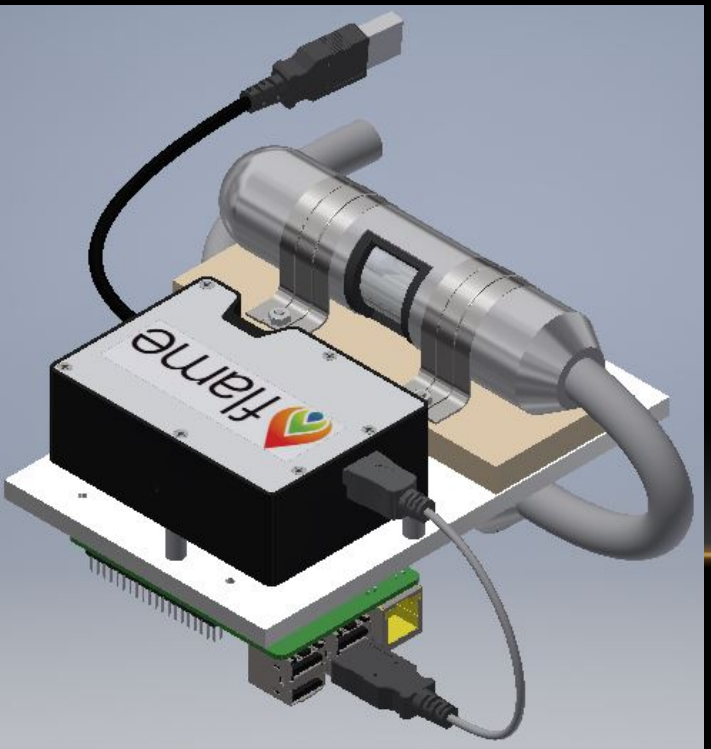
<< Section From Code

Example Data File >>

```
Mon Jul 16 10:47:01 2018
Integration Time: 5000000 microseconds
PLASMA
P:    0:        7
P:    1:    11133
P:    2:     1094
P:    3:     1061
P:    4:     1343
P:    5:     1180
P:    6:     1206
P:    7:     1196
P:    8:     1248
P:    9:     1329
P:   10:     1245
P:   11:     1234
P:   12:     1251
P:   13:     1245
P:   14:     1318
P:   15:     1206
P:   16:     1248
P:   17:     1277
P:   18:     1354
P:   19:     1248
P:   20:     1276
P:   21:     1361
P:   22:     1294
P:   23:     1284
P:   24:     1303
P:   25:     1301
P:   26:     1324
P:   27:     1328
P:   28:     1297
P:   29:     1377
P:   30:     1302
```

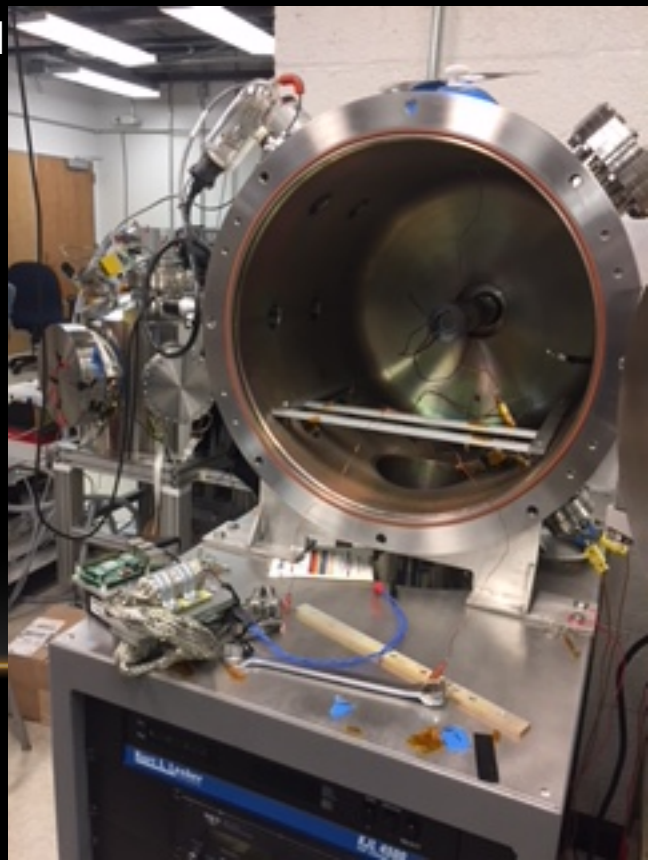
TESTING THE SPECTROMETER

- Built a test rig to go in a vacuum chamber and be pumped down to 5e-5 torr
  - Proved that the spectrometer can function in high vacuum
- Created program to average and noise-subtract many data sets at once
  - Analysis of data showed an integration time of 3.5s ideal

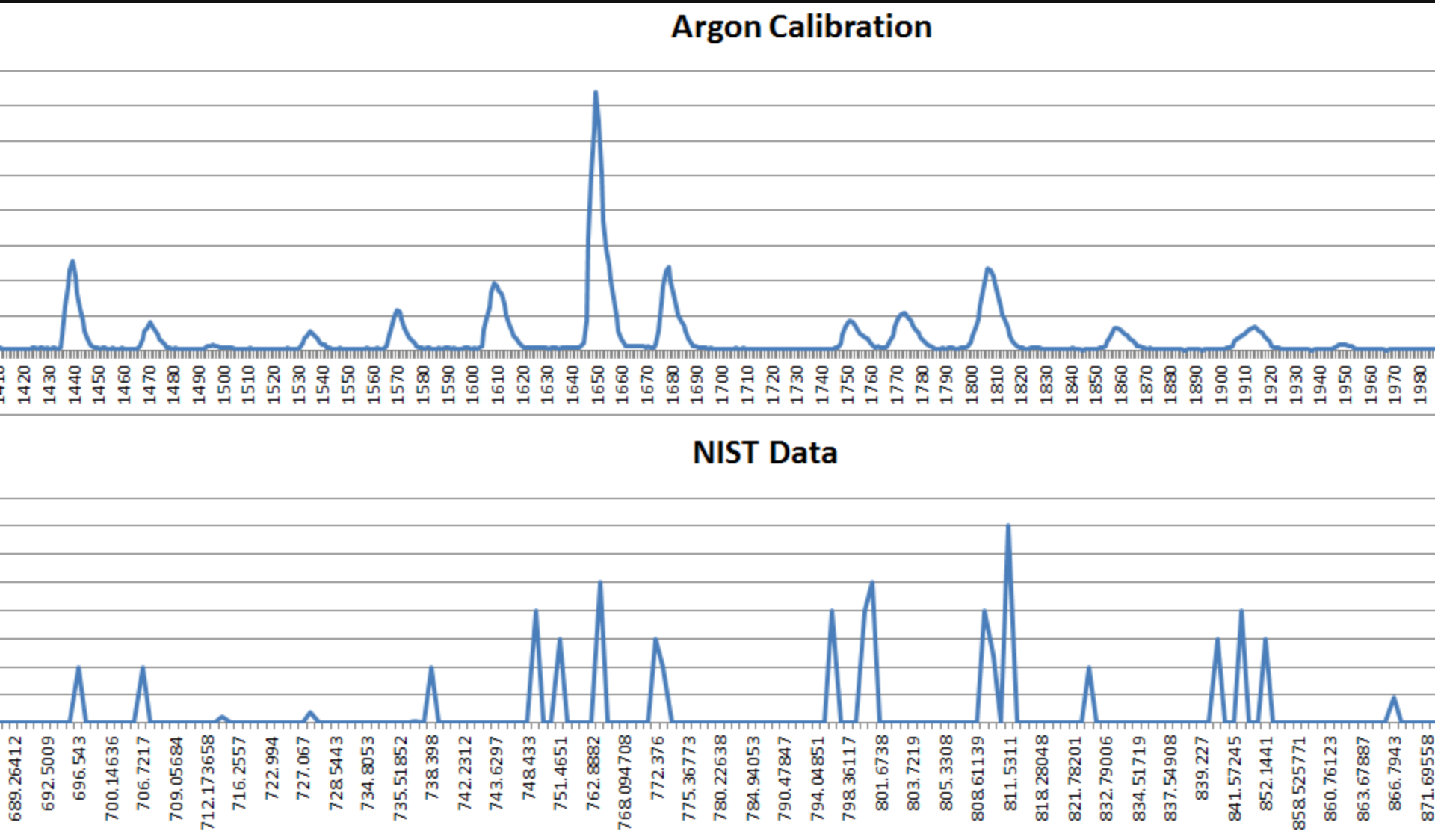


<< CAD of the test rig

Test rig about to go in the vacuum chamber >>



CALIBRATING THE SPECTROMETER



CHALLENGES

- Interfacing with other peoples' uncommented & poorly styled code
- Waiting on supplies we didn't think we needed to ship
- Locating a new vacuum chamber when the one we planned on using broke

WHAT I'VE LEARNED

- How to write software that interfaces with something other than a console user
- How to read, understand, and use spectrographic data
- The more important it is that something succeed on its first try, the less likely it is to succeed on its first try
- Regardless of how loose it seems, giving a nut an extra half-turn can have disastrous consequences

MOVING FORWARD

- Finalize the Spectrometer's Calibration
- Prepare Software to Analyze the Flight Data When it is Received
- Document Everything I Have Done for Future Reference

ACKNOWLEDGEMENTS

A special thank you to the Maryland Space Grant Consortium for providing this opportunity

Thank You to the following who have all contributed to making this internship both productive and enjoyable:

- Dr. Romero-Talamás
- Dr. Jared Young
- Angela Walters
- Hank Mink
- Will Rivera
- Jackson Stefancik
- Mike Schwab
- Marcus Bailey

QUESTIONS?