Construction of a Control System for a Spatial Spectral Double-Fourier Transform Interferometer

Braden B. Gail[1], Chris S. Benson, Jeremy P. Scott, Dinula D. Silva, Locke D. Spencer
[1]braden.gail@uleth.ca
Institute for Space Imaging Science, Department of Physics & Astronomy, University of Lethbridge, 4401 University Drive, Lethbridge, Alberta, T1K 3M4, Canada

Abstract

This projects focus is on the development of a software interface for a 25 pixel cryogenic Transition-edge-sensor (TES) detector array, two linear translation stages, and other lab monitoring equipment used in the construction of a spatial spectral double-Fourier interferometer. The primary purpose of the control system is to assist in the design, calibration, and data acquisition for the system. This is accomplished by allowing users to quickly and easily interact with individual parameters or components of the apparatus and allows for the interaction of multiple subsystems at once.

Linear Translation Stages

The construction of the spatial spectral double-Fourier interferometer involved the acquisition of two Aerotech ALS20045 linear translation stages which are controlled via ASCII commands. This enables two way communication to occur over a network between the Ensemble ML stage controllers and the control interface. The stage positions in relation to the apparatus can be seen in Figure 1 below.

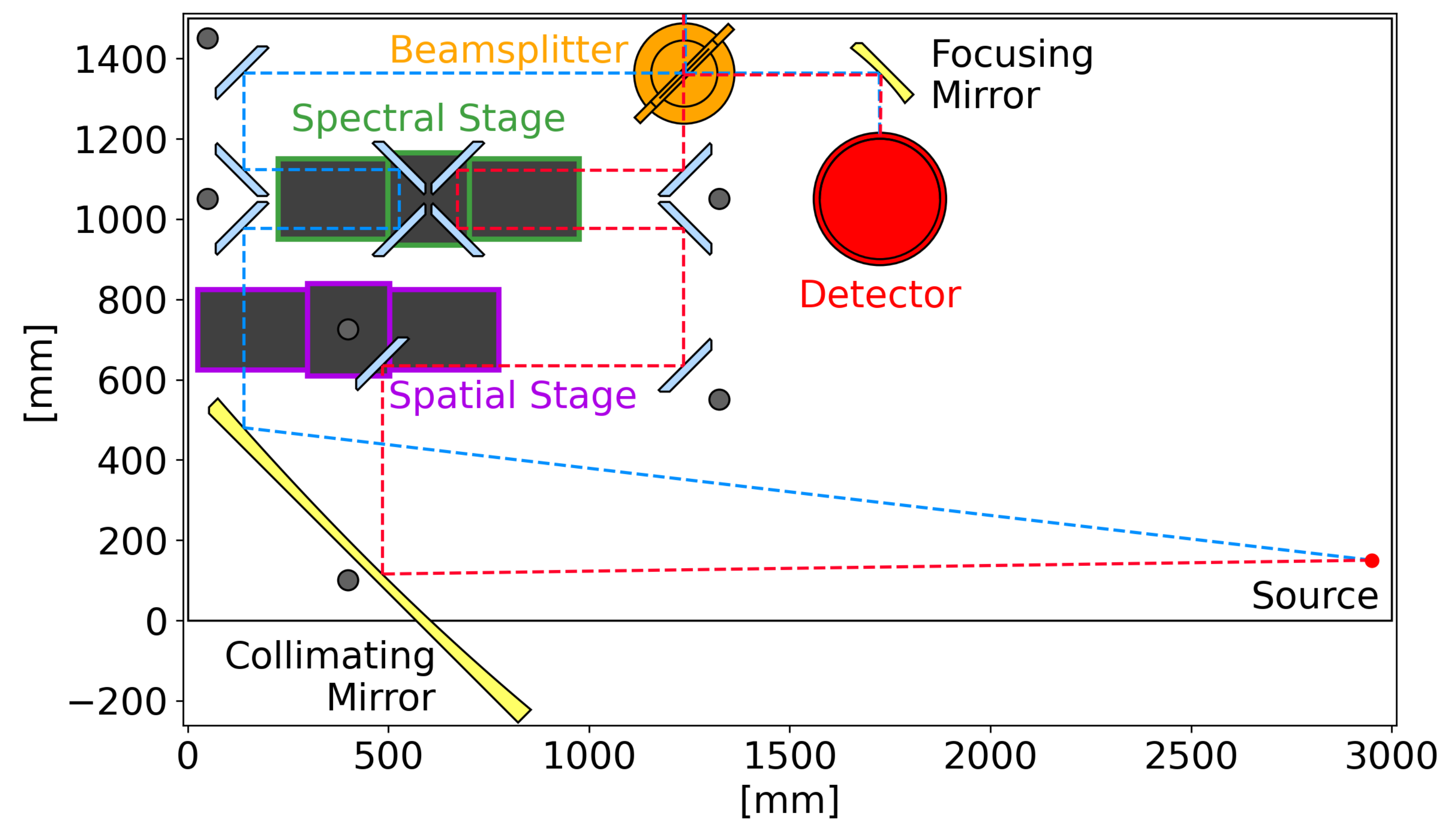


Figure 1: General double-fourier transform interferometer setup including the two linear translation stages and the detector system. A few rays from a source are also traced through the system as well.

Detector System

The detector system is comprised of 25 Transition Edge Sensors (TES) that are all cryogenically cooled to around 4K (Figure 2). At this temperature, individual PID control loops maintain each TES at their critical superconducting transition temperature by varying the voltage bias across each sensor. Each of these PID control systems are maintained by a National Instruments Field Programmable Gate Array (FPGA) and can be individually tuned.

Although the FPGA was originally controlled using LabView software, the National Instruments FPGA (NIFPGA) python API allowed for the interaction between the FPGA and my interface via the python programming language.

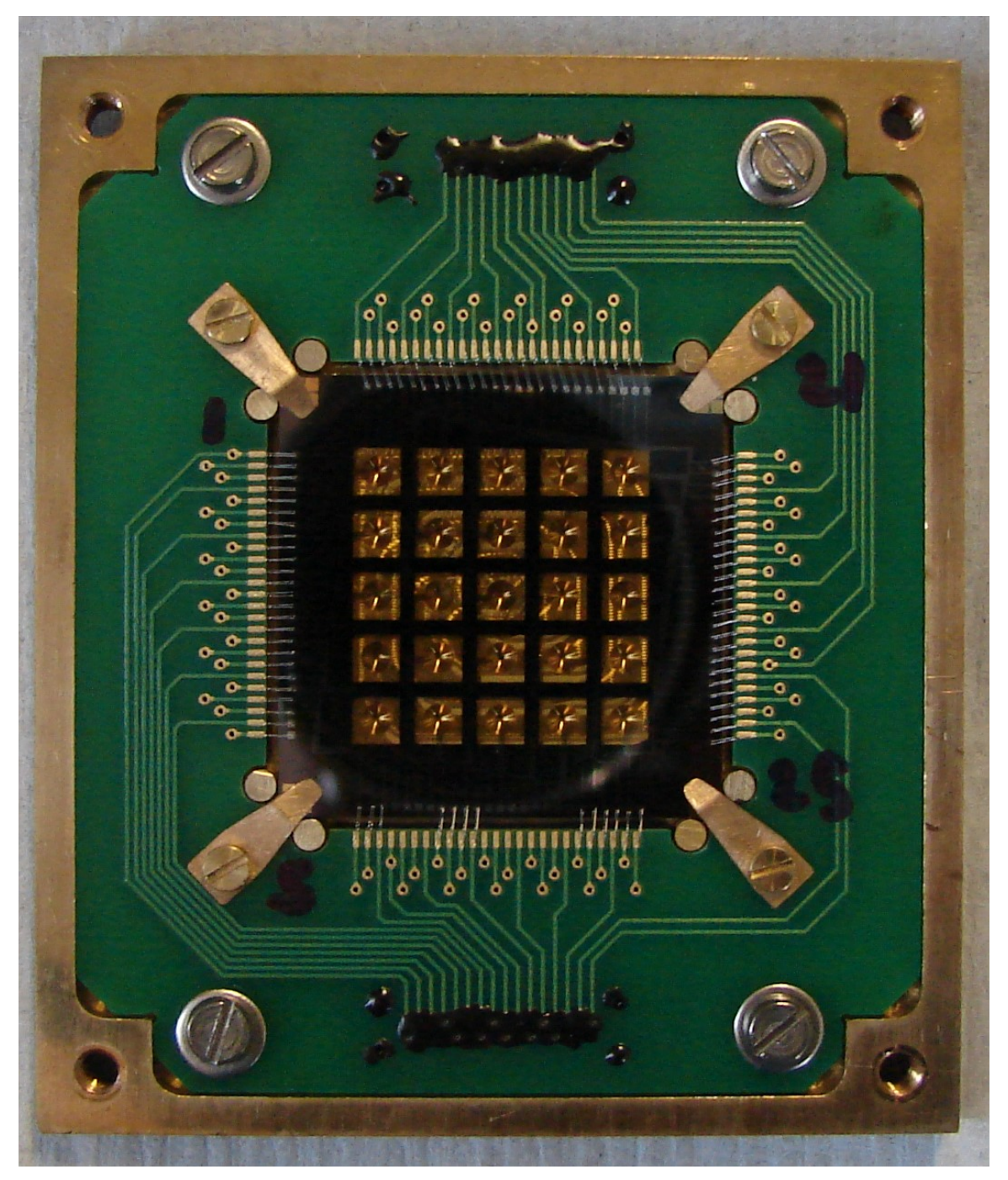
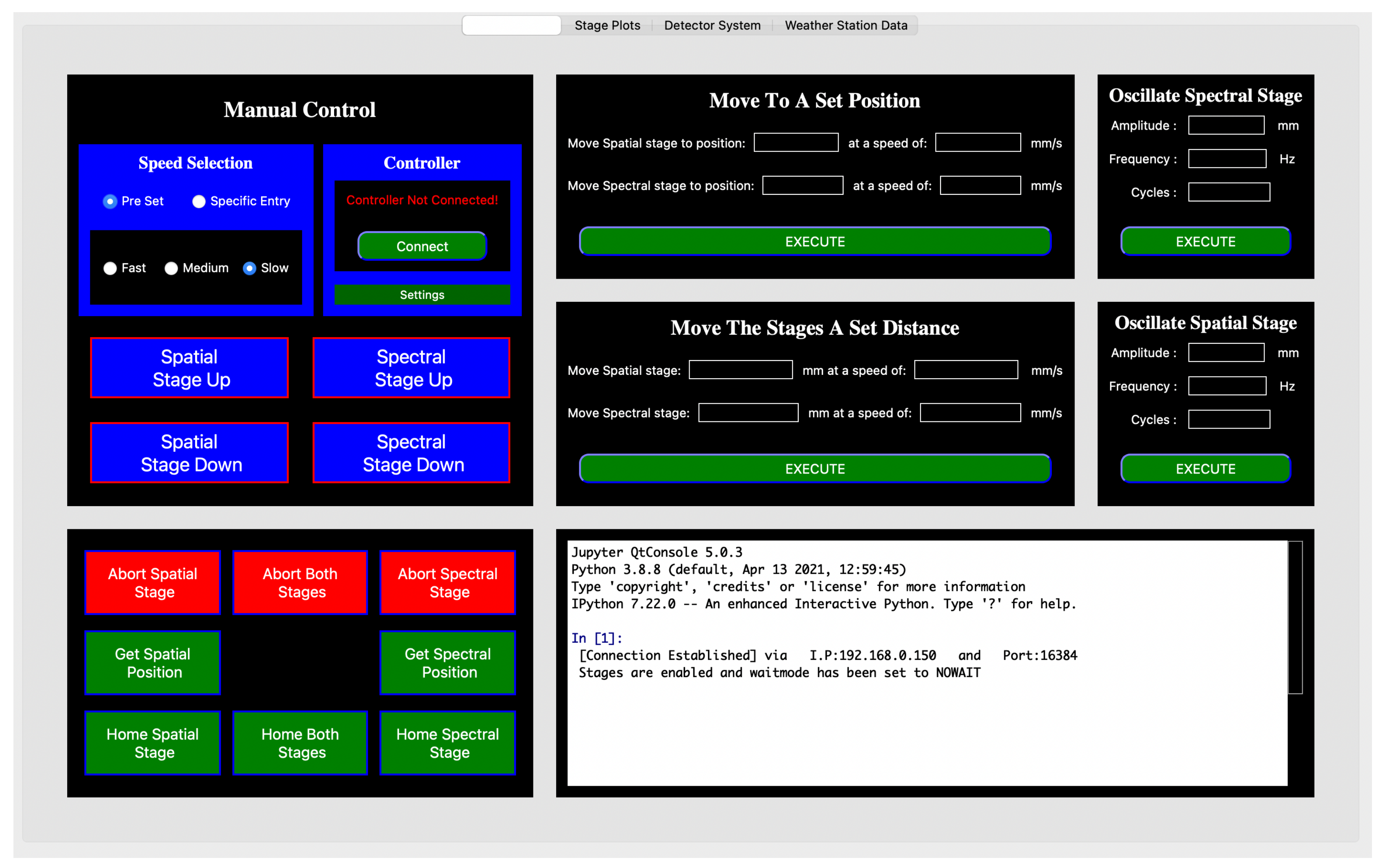


Figure 2: 25 pixel transition-edge-sensor array used as the interferometers detector system.

Construction of the Control Interface

The entire software interface was constructed using python's PyQt library which allowed for the development of a graphical control system. Due to the large quantity of features required to manage both stage and detector control, a tabular organization scheme was employed to avoid an oversaturation of graphical elements. Each tab also had a settings button which gave users access to additional features of the program (Examples of tabs and settings can be seen in Figure 3).

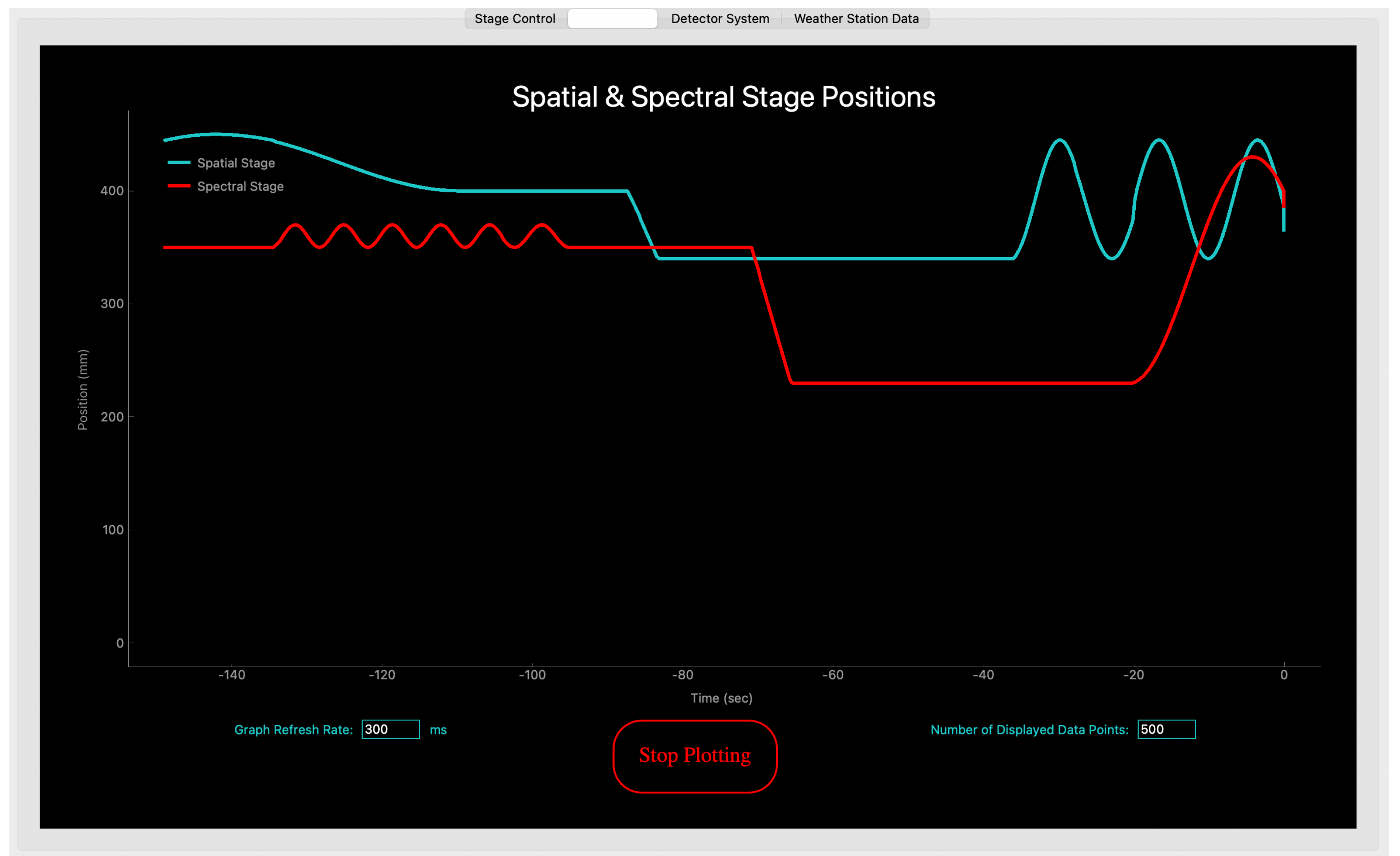


```
Jupyter QtConsole 5.0.3
Python 3.8.8 (default, Apr 13 2021, 12:59:45)
Type 'copyright', 'credits' or 'license' for more information
IPython 7.22.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]:
 [Connection Established] via   I.P:192.168.0.150   and   Port:16384
 Stages are enabled and waitmode has been set to NOWAIT
```

Figure 3: Control interface for the two spatial/spectral linear translation stages. You can see the various tabs at the top of the program which allow the user to switch between subsystem control screens.

The extensive number of tasks the interface had to deal with also caused issues early on with the program freezing or lagging. This issue was fixed however through the use of multi-threading which works by putting each individual task into its own separate thread which could then be run concurrently. The user can also adjust the refresh rates of the real time plots depending on their needs, or turn the active plotting off which would assist in the programs efficiency. Multi-threading was also used to send the ASCII commands to the stage controllers which allowed multiple commands to be sent over the network without having to wait for a response from the stages.



Graph Refresh Rate: 300 ms
Number of Displayed Data Points: 500

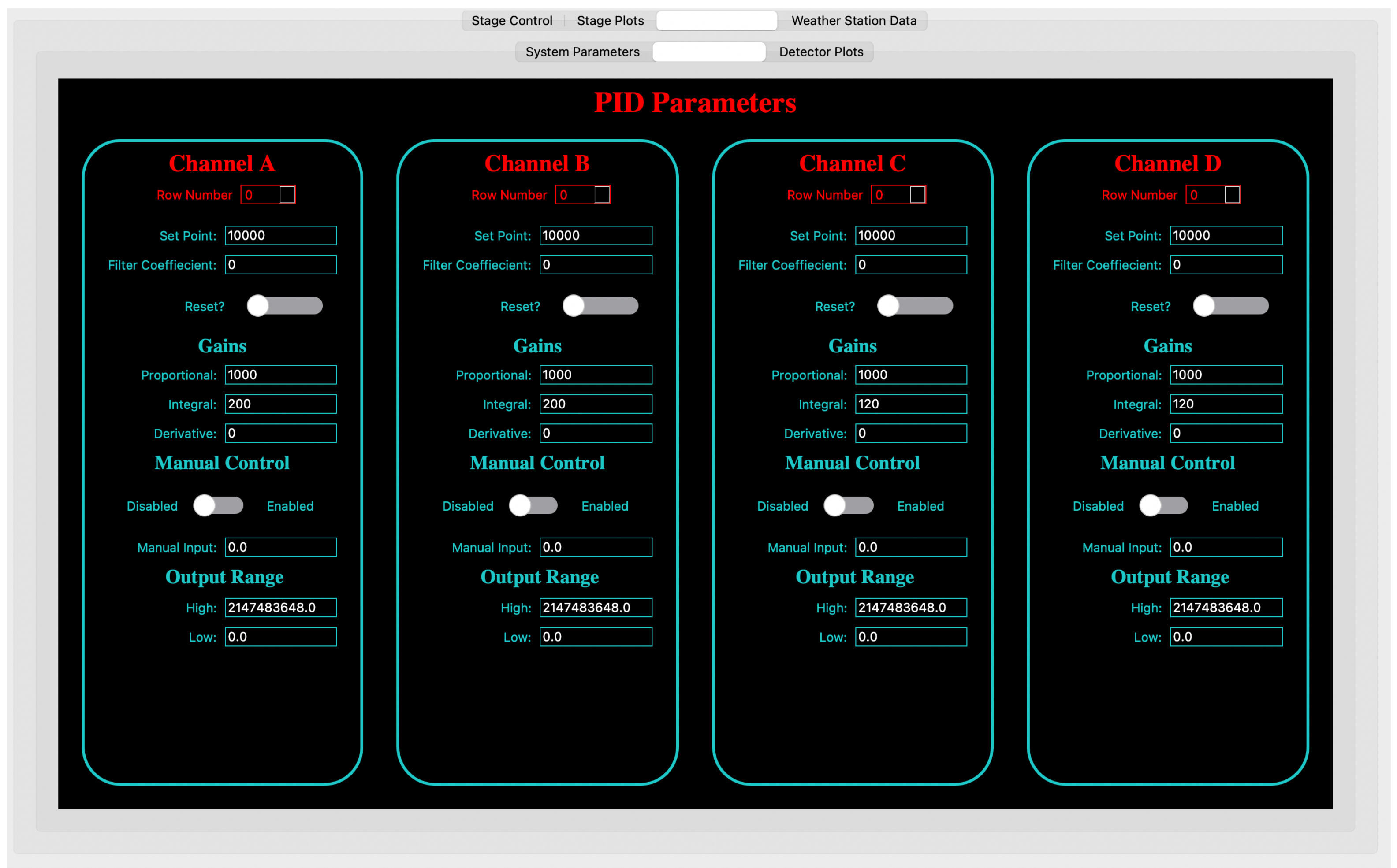
Figure 4: This section of the interface was dedicated to tracking stage movement. The refresh rate and number of data points could be changed by the user depending on their needs. Red line shows the position of the spectral stage while the cyan line shows the spatial stage.

Features of the Interface

Brief overview of some of the features contained in the control system:

- Can manually control both linear stages through the use of an external game pad
- Can set the location, speed, or oscillation path of either stage
- Active warnings when the user attempts to move the stages outside of their physical limit
- Real time plotting of both stage positions
- Users can update individual PID parameters for a specific TES pixel in real time
- Start/Stop detector system recording
- Can Specify the allotted direct memory access for the FPGA
- Can turn on/off detector multiplexing
- Program checks all user entries before sending them to the FPGA.
- Individual plotting of each pixels voltage output
- Entire control system is multi-threaded to speed up the performance and to minimize the interfaces lag time

The control systems most important feature is its simplicity and ease of use. For example, all of the PID parameters for each individual pixel can be accessed and changed by navigating to the PID tab shown in Figure 5. From here, the user has access to a multitude of settings and can modify any of them. The program also checks any new entries to ensure that the new parameter value is within a specific bound and of the right type (float, string, integer, etc).



	Channel A	Channel B	Channel C	Channel D
Row Number	0	0	0	0
Set Point:	10000	10000	10000	10000
Filter Coeffecient:	0	0	0	0
Gains				
Proportional:	1000	1000	1000	1000
Integral:	200	200	120	120
Derivative:	0	0	0	0
Manual Control				
Manual Input:	0.0	0.0	0.0	0.0
Output Range				
High:	2147483648.0	2147483648.0	2147483648.0	2147483648.0
Low:	0.0	0.0	0.0	0.0

Figure 5: This tab contains all of the parameters associated with the PID control loop for each TES pixel.

References

[1] P. Mauskopf. Transition edge sensors and kinetic inductance detectors in astronomical instruments. *Publications of the Astronomical Society of the Pacific*, 130(990):082001, 2018.
[2] National Instruments. National instruments FPGA interface python API. `https://nifpga-python.readthedocs.io/en/latest/`, 2017. Maintained by Michael Strain and Mose Gumble.