This guide serves as a reference to calibrate the UVAMO Spinning Waveplate Polarimeter. Each constructed polarimeter must be calibrated with multiple parameters, which is done by running a series of Python scripts. The construction of our Polarimeter has a Raspberry Pi 3 with an MCC118 DAQ HAT, and the optimal settings from our setup will be outlined below. All scripts are found within the UVAMO GitHub. It is recommended to perform the calibration in the same room or area as the experiment. As with any lab procedure, it is advisable to read the entire document before beginning.

Initial setup after installation

Depending on the model of DAQ HAT and Raspberry Pi, your setup may have a different capacity for data acquisition, and initial sampling settings must be set per version. To access this sampling file, go to `settings > daqsettings.json`, and change `samples_per_channel` and `scan_rate` as follows. Set `samples_per_channel` to an integer between 1000 and 10000. This will designate the volume of samples per channel; at higher volumes, the averaging improves, and at lower volumes, the more accurate the real-time representation. The MCC118 code and DAQ HAT model determine the maximum `scan_rate`. This code will give an error for values above 100000. The `scan_rate` value must accommodate for all the channels. In our case, we chose 25000, which gives a sampling rate of 50000 across two channels. This provides the resolution for the sampling. The list of channels and the timeout value can also be changed within this file, but this is not critical.

Calibration

Requirements for calibration

Calibration can only be accomplished with a coherent light source of a known monochromatic wavelength. The laser light must be directed at normal incidence to the photodiode. Other optical apparatus includes a half-waveplate to obtain horizontally polarized light. Turn on the motor, and proceed with the following procedures.

Calibrating the trigger delay

The Hall sensor and spinning waveplate are not simultaneous in rotation, therefore the orientation and triggers must be precisely calibrated for the reconstruction of polarization states. The calibrating light source must be used to complete this step, and one must ensure that the light is highly circularly polarized. Go to `calibrate_trigger_delay.py` and run the script. The script automatically gauges the pulse of the Hall sensor and determines the delay ($\Delta\theta$) such that any component the component of the signal that can be represented as a $\cos 2(\omega t + \Delta\theta)$ term is minimized (leaving only the $\sin 2(\omega t + \Delta\theta)$ component at frequency 2ω) allowing a zero point to be set. The script then writes the trigger delay to the `swpsettings.json` file so that it may be accessed by the main function. It is recommended to run twice or more to verify the value of the trigger delay. It is also advised to run the script again every time the polarimeter is partially or fully disassembled, regardless of whether the polarizing film or hall sensor are removed/replaced.

Adjusting for background light sources

The photodiode utilized in the polarimeter is sensitive to background light which must be subtracted from the input signal. This step is critical and should be repeated whenever the environment changes. Go to `calibrate_background.py` and run the script. Block your input beam at the source to ensure the background value remains consistent. The script will verify with the user if the coherent light source is fully blocked before running, and then automatically save the parameters of the background light. You can now unblock your beam.

Adjusting the spinning waveplate retardance

The use of a polarizing filter is required for this portion of the calibration, as the sensing script will now require pure, ”horizontally” polarized light. The previous scripts, `calibrate_background.py` and `calibrate_trigger_delay.py`, must be completed before this step. Ensure that your input light is linearly polarized along the transmission axis of the polarizer (referred to a horizontally polarized as convention with this device). This can be achieved by rotating a polarizer in front of the polarimeter as it is scanning, observing the raw voltage race output from the device, and finding the polarization direction for which the upper peaks (V_max) of the signal are at a maximum. Then, go to `set_waveplate.py` and run the script. This script will adjust the spinning waveplate trace retardance for the selected wavelength. It is advised to run the any time there is a change in wavelength used or significant change to input angle for the polarimeter. This step concludes the calibration of the polarimeter.

Verification of drive motor setup

There has been a supplementary script added to help assess the efficacy of the chosen drive motor, and whether or not the motor has any rotational inconsistency. Go to `estimate_motor_jitter.py` and run the script. This function measures what the mean period of rotation is, as well as the maximum, minimum, and relative error of the mean. It would be meaningful to run this script more than once per setup, or whenever the motor speed has changed. For our polarimeter, we noted that these values did not change more than one discretized value per cycle. If a change of more than one discritized period is observed consistently, then an alternative motor should be used.