Title

Index

Abstract:

This outlines the installation and usage of the collection of drivers and data processes which are designed to work independently to collect and store data autonomously, and to render the data into human readable formats. There exist 3 peripheral devices: a wide-angle camera, an Infrared camera, and a spectrometer.

Construction:

The Infrared camera and the spectrometer are connected in parallel to a common 3.3V output pin and a common ground (GPIO 1, and GPIO 9 respectively). Both devices communicate with the Raspberry Pi 4 via I2C protocol, and share both the serial data line, and serial clock line (GPIO 2, and GPIO5 respectively). Keep in mind the Raspberry Pi 4 has built in pullup resistors on the I2C pins. The MLX90640 uses address 0x33 and AS726X uses address 0x49. Below is the schematic for the setup.

Insert Diagram here

```
pi@raspberrypi:~/Infrared_camera/driver $ i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- 33 -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- 49 -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70:
```

The ELP-USBFHD01M-L180 is attached to the raspberry pi 4 via USB3. All three devices are taped together, and they're estimated to be the same orientation with the naked eye.

Installation:

MLX90640 Infrared Camera:

To use this camera with the raspberry pi and our collection of scripts there a several dependencies that are required to be installed

1) Adafruit_mlx90640
    a. In command line run *pip install adafruit-circuitpython-mlx90640*
2) board
    a. In command line *run pip install --upgrade adafruit-python-shell*
3) Busio
    a. Will be installed from above.

AS726X Spectral Sensor

The dependencies for this camera should be installed with the MLX90640 IR camera except for the as726x itself:

- Run pip3 install *adafruit-circuitpython-as726x*

ELP-USBFHD01M-L180

This is an ultra-wide angle USB camera, and the program that operates this camera depends on ***OpenCV*** to operate and can be installed with

- *sudo apt-get install python-opencv*

Configuring Drivers

Configuration is set inside JSON files for each peripheral device. Located in the driver/config directory.

MLX90640 configuration

- Size: Contains the configuration to determine the width and height of the frame, and channels determines the color complexity of the color map, 1 indicates a single linear channel.
- Storage: The infrared camera stores data as a CSV which can later be transformed into a heatmap.
    - Precision indicates the decimals of precision that will be stored, increasing this will incur an increase in CPU load
    - Location determines where the csv file will be stored. By default, it will create a new csv file based on the date, if one exists already, data gets appended
    - Path is the relative location to store the csv
- Recording
    - Frequency dictates how many frames per second will be captured. Incurs penalties to CPU load
    - Duration specifies how long the IR will run for, taking frames at the specified frequency above.
- Triggers
    - At this stage, it does nothing, but future goal is to specify what events are worth responding to.

AS726X configuration

Stored in: *driver/config/spectro.json.*

- Color_channels set what color channels will be stored in a CSV for later processing. Reducing the amount set to true will improve CPU load.
- Storage: data is stored in a CSV.
    - Precision indicates the decimals of precision that will be stored, increasing this will incur an increase in CPU load
    - Location determines where the csv file will be stored. By default, it will create a new csv file based on the date, if one exists already, data gets appended
    - Path is the relative location to store the csv
- Recording
    - Frequency dictates how many frames per second will be captured. Incurs penalties to CPU load
    - Duration specifies how long the spectrometer will run for, taking readings at the specified frequency above.
- Mode determines what color channels it will read.
    - **0:** Continuously gather samples of violet, blue, green and yellow. Orange and red are skipped and read zero.
    - **1:** Continuously gather samples of green, yellow, orange and red. Violet and blue are skipped and read zero.
    - **2:** Continuously gather samples of all colors
- Triggers:
    - At this stage, it does nothing, but future goal is to specify what events are worth responding to.

ELP-USBFHD01M-L180 configuration

Stored in: *driver/config/mp.json.*

- Size: Contains the configuration to determine the width and height of the frame, and channels determines the color complexity of the color map, 3 represents 3 distinct color channels
- Storage: Due to the size of the image (3 X 640 X 480) its stored in a directory based on the date, as PNG's
    - Precision is redundant for this image
    - Location determines where the PNG file will be stored. It will use the current time and be stored in the directory that indicates the date.
    - Path is the relative location to store the directory storing the PNG's
- Recording

    - Frequency dictates how many frames per second will be captured. Incurs heavy penalties to CPU load.
    - Duration specifies how long the camera will run for, taking readings at the specified frequency above.
- Triggers
    - At this stage, it does nothing, but future goal is to specify what events are worth responding to.

NOTE: Setting frequency of the ELP-USBFHD01M-L180 too high will incur a large CPU load and Raspberry Pi 4 will become unresponsive.

Cumulative configuration

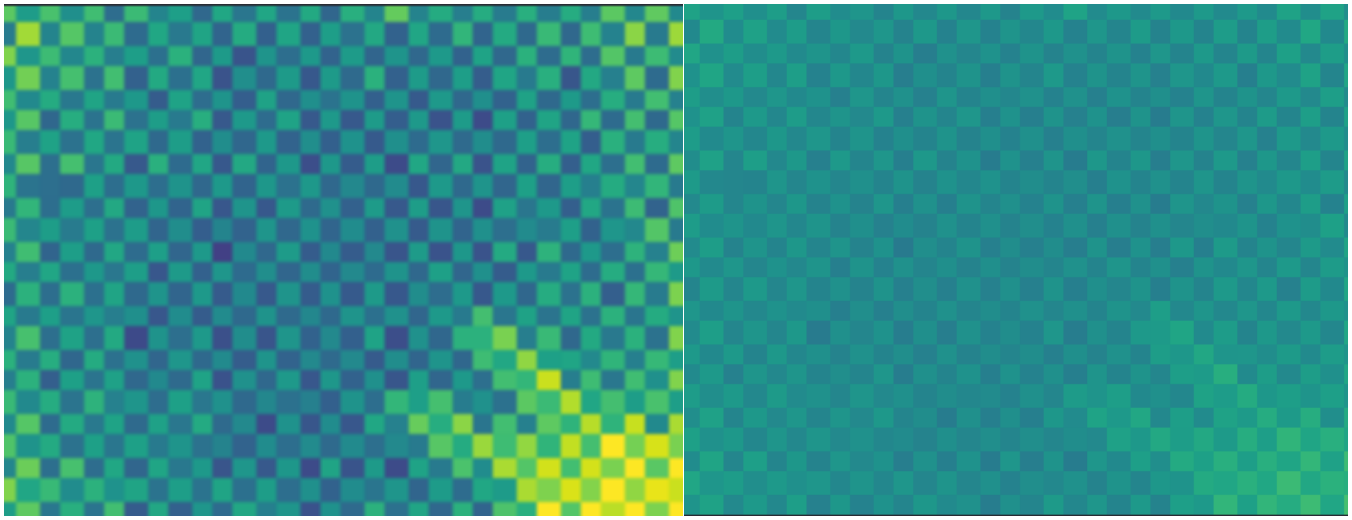
Stored in: *driver/config/once.json.*

- Is the summation of the above json configurations and is used exclusively by **run_once.py**

visualize.json and **compose.json:**

stored in *driver/config/…*

- Storage: specifies where the output for converting infrared CSV data to an image can be located
- standard_deviations:
    - Determines how many standard deviations will be included in the scan. Values outside the number of standard deviations are left out.
    - A higher value for standard deviations results in less contrast in the heat map.
    - A lower value for standard deviations results in more noise in the heat map.



3 standard deviations. 10 standard deviations.

Software Components:

run_mlx.py:

Operates the MLX90640 infrared camera using the configuration specified by the *config/mlx.json* file. Captured data is also stored based on the configuration file. In the case of a CSV not existing for the current date, one will be created automatically, otherwise data is appended to the appropriate CSV file. By default, the data is stored and can be identified based on the date with *infrared_YYYY_MM_DD.csv* format

Each row of the CSV corresponds to a singular frame. The very first entry of that row identifies the time of the frame and is designed to correlate data from different cameras together.

run_mp.py:

Operates the ELP-USBFHD01M-L180 wide angle camera using the configuration specified by the *config/mp.json* file. Captured frames is also stored based on the configuration file. Frames are stored as a PNG, as the large amount of

data captured by the camera is large enough to make the Raspberry Pi unresponsive when processing the data. By default, frames are stored and can be identified with *MP_YYYY_MM_DD.png* format.

run_spectro.py:

Operates the AS726X infrared spectrometer using the configuration specified by the *config/spectro.json* file. Captured data is also stored based on the configuration file. In the case of a CSV not existing for the current date, one will be created automatically, otherwise data is appended to the appropriate CSV file. By default, data is stored and can be identified based on the date with *Spectrometer_YYYY_MM_DD.csv* format.

Each row corresponds of the CSV corresponds to a singular frame. The very first entry of that row identifies the time of the frame and is designed to correlate data from different cameras together. Subsequent entries are the levels in the following order;

1) Raw violet
2) Raw blue
3) Raw green
4) Raw yellow
5) Raw orange
6) Raw red
7) Violet
8) Blue
9) Green
10) Yellow
11) Orange
12) Red
13) temperature

run_once.py:

Operates the MLX90640 infrared camera, ELP-USBFHD01M-L180 wide angle camera, and the AS726X infrared spectrometer devices using their respective json configuration file specified in **Configuration Files**. This script captures a frame from each of the peripheral devices and by default, those frames are stored in the same directory, identified by the data and time in the format *Once/YYYY_MM_DD/HH_MM_SS.* The AS726X file is a CSV named *infra_spectro.csv,* the MLX90640 frame is *infrared.csv*, and the ELP-USBFHD01M-L180 frame is *mp.png.*

viz_infrared_for_analysis.py:

Converts CSV rows into PNG images exclusively for the MLX90640 camera. Designed to be run from a workstation, and not while the device is operating with limited processing power. This script yields an image that also contains scale information, and axis values which is different from: different than *raw_infrared_for_composition.py.* Takes 1 parameter:

1) Path to the csv file.
    a. Must exist in the current directory. Script can't traverse to parent directory.

raw_infrared_for_composition.py:

Converts CSV rows into PNG images exclusively for the MLX90640 camera. Designed to be run from a workstation, and not while the device is operating with limited processing power. Unlike ***viz_infrared_for_analysis.py,*** this script yields only the heatmap, such that it can be composed onto a raw PNG file. Takes 1 parameter:

1) Path to the csv file.
    a. Must exist in the current directory. Script can't traverse to parent directory.

Utils/constants.py

Declares and initializes the constants used across the driving programs for the devices and for the data processing software that converts captured data into human readable formats.

Utils/csv_handling.py:

Is responsible for handling the conversion of data received from the AS726X, and the MLX90640 into a CSV file.

Utils/image_handling.py:

Contains the function definition for saving the image captured by the ELP-USBFHD01M-L180 wide angle camera.

Add:

Testing

Construction