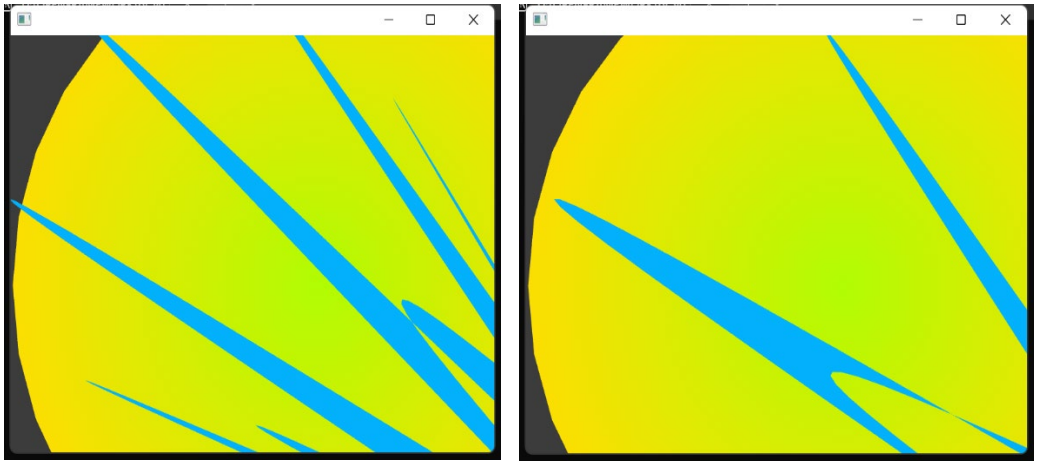
Coding2 Lab work Readme

In this Lab work submission, I have 2 project written by open framework and 2 project using Python.

1. Lab work1 Open framework +Addon (audio and visual effect)

The Labwork1 is a project using addon in Open framework. It is an audio + visual artwork. Sound and Image is linked. Mouse can control the shape gernerated ass well. One of the addons used is ofxMaxiliam. It generate the sound in this work. I downloaded addon from: https://github.com/micknoise/Maximilian. Another addon is ofxVectorGraphics. I found it in example of open framework. It controls the change of visual part.



Panopto link:

https://ual.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=6d3d45db-8276-44f9-a32f-afc500ee7efc

2. Lab work2 Open framework communicate with Arduino

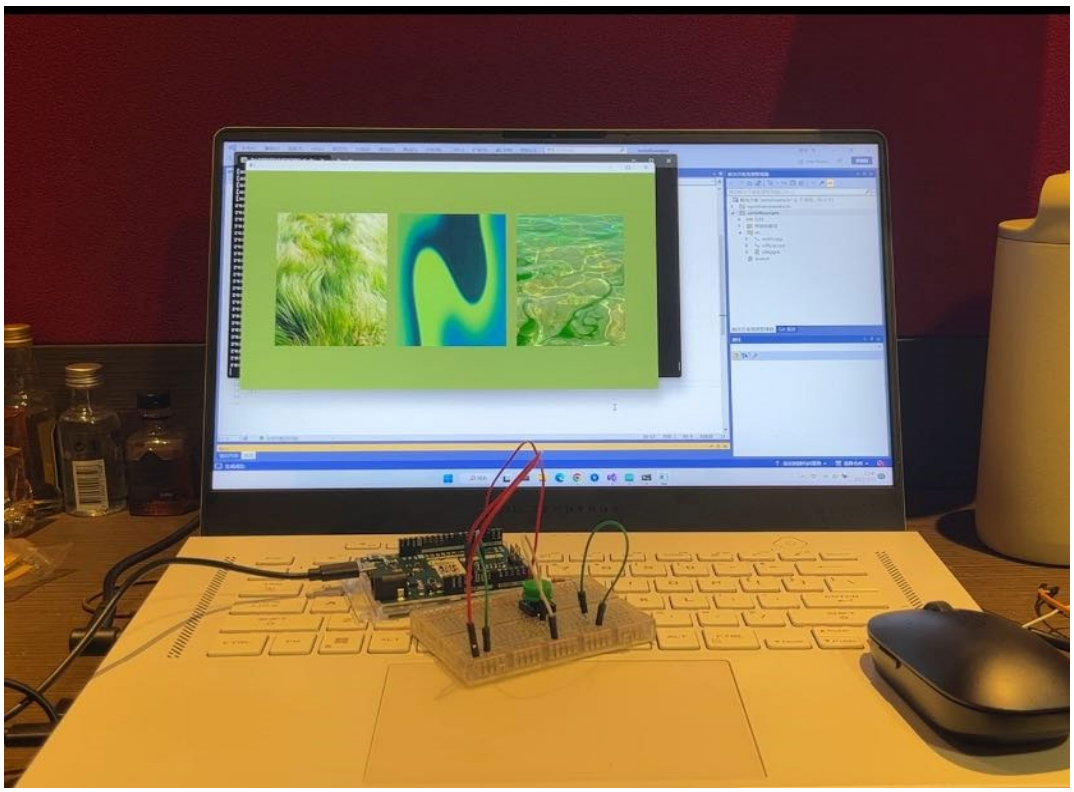
Labwork2 is a project based on communication of Arduino and open framework.

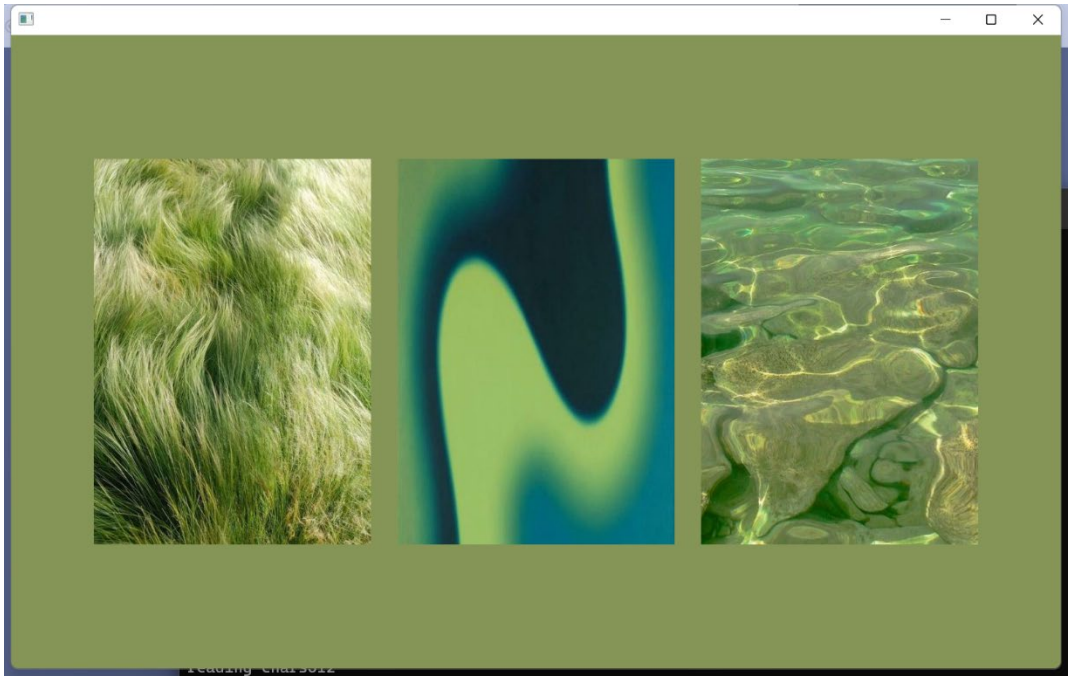
The setup function sets the background color, loads fonts, and sets up serial communication with a device at a specific port and baud rate. The communication between Arduino and open framework is achieved by serial communication and setting both Arduino and open framework with 9600 baud so that they can send and receive message. When button is pressed, Arduino will send a “@” to open framework, I use “if” here, and if open framework received byte number bigger than 0, pictures and

sound will be load and showed.

This is the basic process. The most important part in this project is setting right baud rate for serial communication, which will use **int baud = 9600.**

This project can be more improved that using more addon to generate motion images on screen and changeable sound because now the pictures and sound is made, and can not be changed by Arduino sensor.



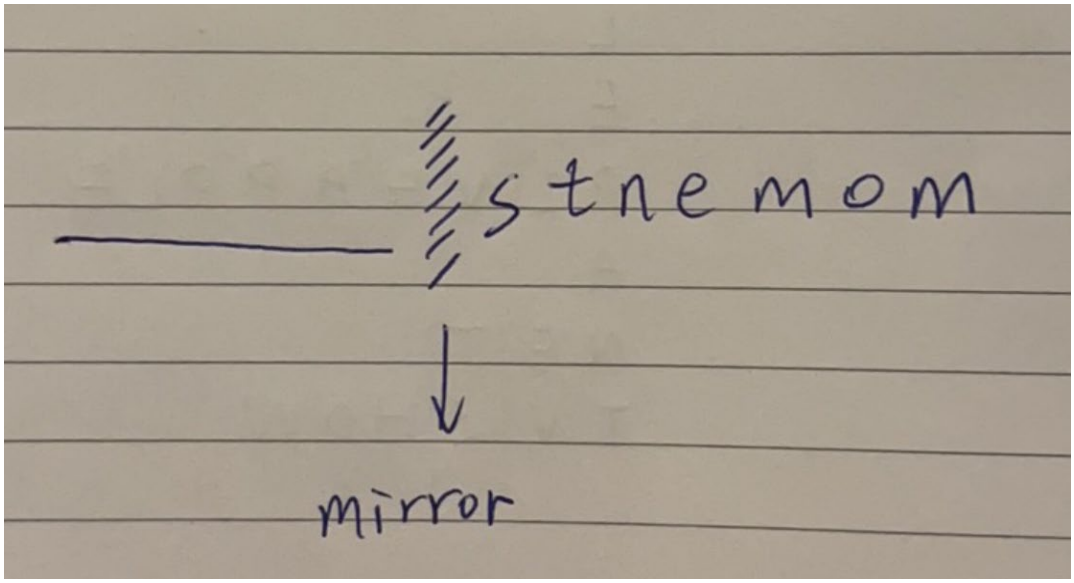


Panopto link:

https://ual.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=9def4611-0943-4c87-919c-afc500ee9a86

https://ual.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=a11193dd-5113-460e-a040-afc500ee9c64

3. Python challenge (python puzzles)



Python challenge is puzzle needed python to solve. In this homework, I designed four puzzles needed to be solved by python. The puzzles here require the use of sorting in Python, searching in order, printing the results, finding information in specified rows and columns, converting to ASCII using Python, and operating on stacks. For complete Python challenge, you can check Jupyter notebook.

4. Python image dataset exercise (python average pics)

In this python image exercise, I tried download 60 pictures with same theme--flowers with python, saved them into a new directory. Then average them with different numpy functions. I tried different numbers of source images with same function and I also added some decimal numbers to the original operations, and added the results obtained from different numpy functions together to generate a new image and create different effects. The following are two of them, you can open my Jupyternotebook for more effects

The first difficult I met in this process is choose suitable dataset. The ideal dataset contains pictures with same size. Then create use **os.mkdir('dirctory name') to build a new directory for storing pictures.**

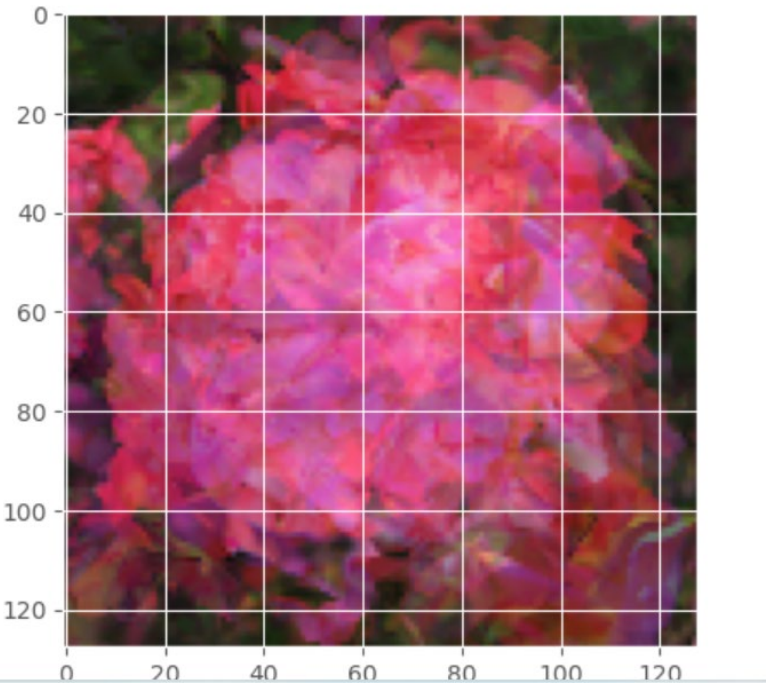
```
    # create a string using the current loop counter
    # 3 means how many digits will the printed number have; d means decimal integer
    f = '00_%03d.png' % img_i

    # and get the url with that string appended the end
    url = 'https://raw.githubusercontent.com/ZIqinGX/Herb_leaf_for_image_study/main/flower60/' + f
```

The step here is to download pictures in dataset by url.

```
In [13]: imgs[0].shape

Out[13]: (128, 128, 4)
```

With img[ ].shape, I can know width and height of pictures and color channels. Then using Numpy library to average pictures.

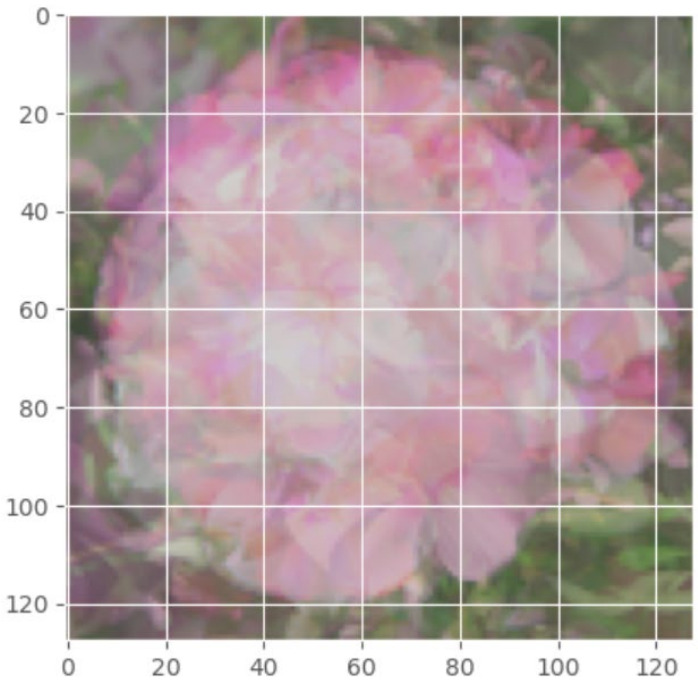
```
In [19]: nan_img2 = np.nanmedian(imgs2, axis=0)
         plt.imshow(nan_img2)

Out[19]: <matplotlib.image.AxesImage at 0x27f42b41cd0>
```



In [24]:

```
imgs6 = [plt.imread(files[file_i])
        for file_i in range(6,14)]
quant_img6 = 0.8*np.quantile(imgs6,0.9, axis=0)+ 0.01*np.average(imgs6, axis=0)
plt.imshow(quant_img6)
```

Out[24]: <matplotlib.image.AxesImage at 0x27f42eb6550>



I think the improvement can be made here is try more pictures.