Created by Nicolas Mei on Wed 2015.08.05

Updated by Jamie Catalano and Kavin Nuñez on 2019.03.04

First steps:

# **fly-group-activity-monitor (flyGrAM)**

Set of functions and scripts to do real time fly activity quantitation with the added option to control optogenetic stimulation

## **Overview:**

The flyGrAM allows for quantitation of fly motion (i.e. the number of flies moving in a given frame) during various experimental paradigms. Importantly, this movement quantitation is done in real-time (or near real-time).

## **Features:**

1. Can communicate with an arduino to control the onset time, duration, frequency, and pulse width of LED diodes for optogenetic experiments.
2. Can directly write out 'raw' (corrected for camera lens barrel distortions) collected video during an experiment to '.avi' format for subsequent analysis with other more sophisticated tracking software packages (i.e. C-Trax, Jabba, etc...).
3. Can produce real time activity plots for 4 differents regions of interest (ROI).

## **Detailed install instructions**

(link detailed instructions below - or keep all as one file)

## **Required software:**

The FlyGrAM was developed and tested in the Windows OS environment. In theory, the software should be platform agnostic but has not been tested or and is not guaranteed to work with Mac or \*nix systems.

1. Python 2.7 64-bit (Anaconda distribution is highly highly recommended: <https://store.continuum.io/cshop/anaconda/>)
2. OpenCV (version 3.1) with python bindings RC 3.0+ (<http://opencv.org/downloads.html>)
3. FFMPEG 64-bit Zeranoe build (version: 2015-06-05 git-7be0f48) (<http://ffmpeg.zeranoe.com/builds/>, see also: <http://www.wikihow.com/Install-FFmpeg-on-Windows>)

## **Setup particulars:**

1. Flies need to be IR backlit and cameras must have an IR filter fitted for tracking and all components of the experimental setup to work properly.
2. Arduino's need to be loaded with the 'Opto-blink and Solenoids' sketch

## **Required Computer hardware:**

Minimum (untested but should work in theory...):

* 2-core processor is needed (higher clock rate is better)
* 8-16 GB RAM
* USB camera that is supported by OpenCV
* Arduino (Uno: <http://store.arduino.cc/product/A000066>)

Recommended:

* 4-core processor (higher clock rate is better) - will ensure fast FFMPEG encoding
* 16-32 GB RAM
* High quality USB camera that supports at least 30 fps that is supported by OpenCV
* Arduino (Uno: <http://store.arduino.cc/product/A000066>)

FlyGrAM Detailed Installation Instructions

1. Download the latest flyGrAM software from Github (<https://github.com/kaunlab/FlyGrAM-fly_group_activity_monitor>)
2. fly\_activity\_experiment\_manager.py
3. fly\_group\_activity\_monitor\_gui.py
4. roi.py

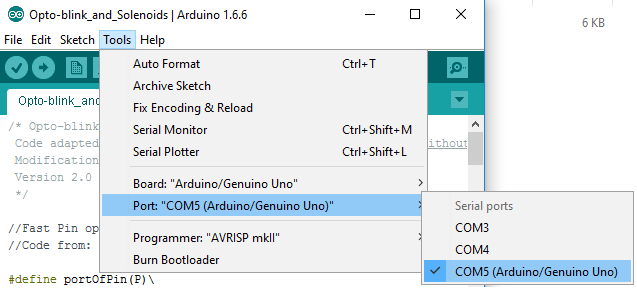
1. Unzip flyGrAM zipped folder to a desired location

Setting up the Arduino:

1. Download the free and open-source Arduino IDE (integrated development environment) ([https://www.arduino.cc/en/ Main/Software](https://www.arduino.cc/en/Main/Software))
2. Plug in Arduino
3. Using the Arduino IDE open up the .ino file located in the unzipped flyGrAM software folder. The folder path should look something like:

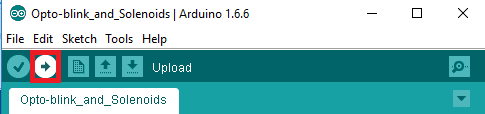
“C:\fly\_gram\_unzipped\_location\fly-group-activity-monitor-master\arduino\Opto-blink\_and\_Solenoids\”

1. After opening up the .ino file click the “Tools” submenu and make sure that the Arduino is detected by the computer. This can be determined be looking at the “Port” and determining if “(Arduino/Genuino Uno)” immediately follows a COM port designation. Be sure to remember the COM port number! It will be used later!

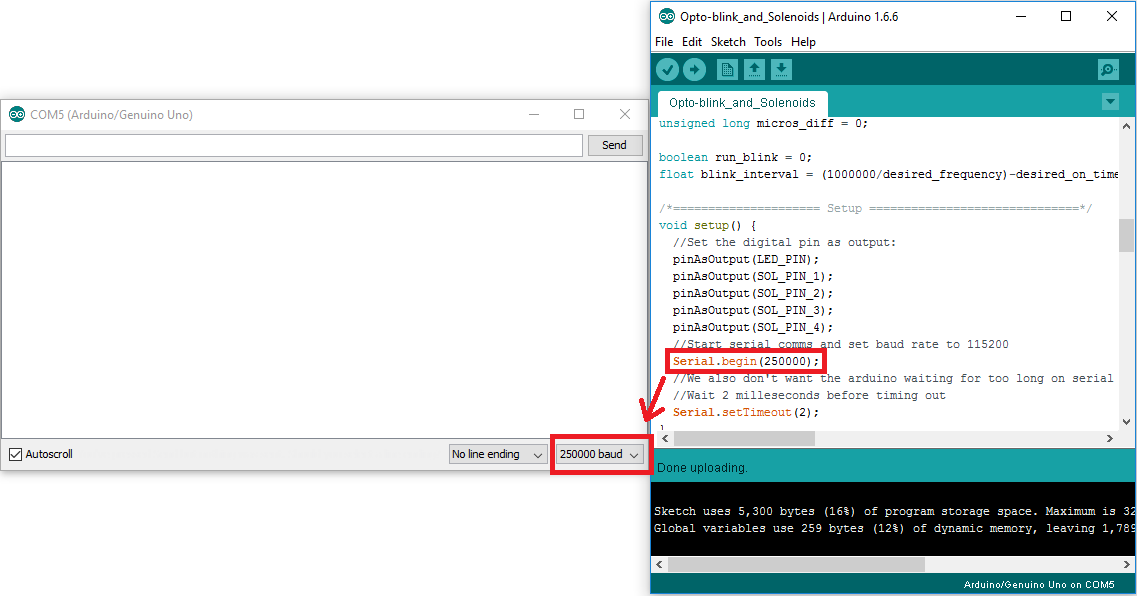


Note: Depending on how many devices are connected to the computer, the COM port which the Arduino will be detected on could vary. Furthermore, the COM port of the arduino may also change after plugging in other devices.

1. Upload the .ino code to the Arduino by clicking the upload button. If the code compiles and upload correctly you should see the message “Done Uploading” near the bottom of the IDE screen.



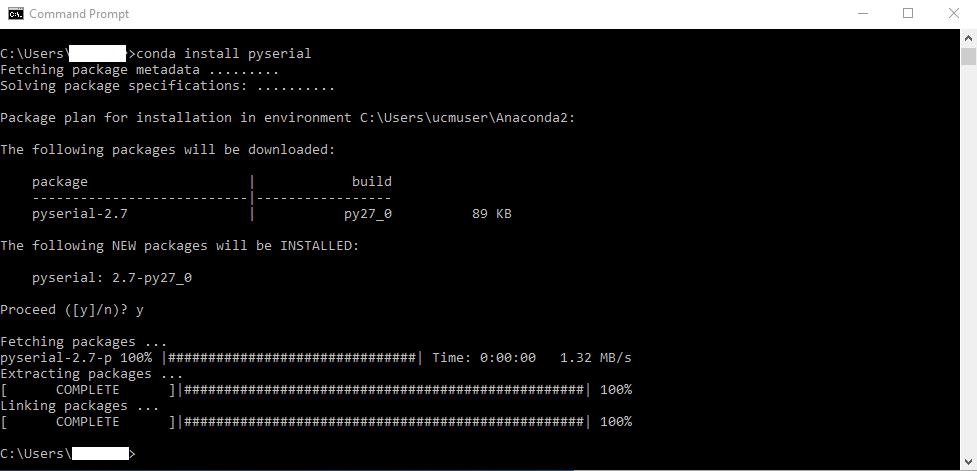
1. To further test that the upload process was successful. Open up a “Serial Monitor” instance which can be found under the “Tools” submenu.
2. In the new “Serial Monitor” window. First ensure that the baud rate is set equal to the “Serial.begin(baud\_rate);” found in the Arduino code.



1. In the serial monitor menu type in “[5,50]” and then press “Enter” to start a 5 Hz 50 ms pulse width LED blink. Other frequencies and intensities of blinking can be used by typing in “[desired\_frequency, desired\_led\_on\_pulse\_width]”. (Note: the Arduino does not check input so it is possible to enter bogus parameters like: “[50,25]” which will just result in the LEDs being always on)

Setting up Python and OpenCV support:

1. Download and install Python Anaconda 2.7 64-bit (<https://www.continuum.io/downloads>)
   1. Now open a command prompt and type: “conda update conda”. Say yes to the command prompt asking if you would like to update the conda environment.
2. Now type “conda update anaconda”. Say yes to the command prompt asking if you would like to update.
3. Finally type in “conda install pyserial” and okay the prompt asking if you would like to install the “NEW package”. You should see the following:



1. Download OpenCV for windows (version 3.1) (<http://opencv.org/downloads.html>)
2. Running the downloaded OpenCV file will open up an unzip location prompt. Extract the OpenCV folder to a desired location (the location doesn’t really matter. Desktop is recommended since it is easy to access)
3. In the unzipped OpenCV folder go to: \opencv\build\python\2.7\x64 (if you extracted the OpenCV folder to the desktop the path should look something like: “C:\Users\user\Desktop\opencv\build\python\2.7\x64”). This folder should contain a file called “cv2.pyd”. “Ctrl+C” this file.
4. Copy the “cv2.pyd” file to the “site-packages” folder of your Anaconda Python distribution.
   1. If you installed Anaconda Python for all users then the path to site-packages should look something like: “C:\Anaconda2\Lib\site-packages”
   2. If you installed Anaconda Python only for yourself then the path to site-packages should look something like: “C:\Users\your\_user\_name\Anaconda2\Lib\site-packages”

Connecting Arduino ‘COM’ to Python

1. Start up Spyder. If using Windows, make sure to start the program as an administrator! (an IDE that is included in the Anaconda Python install package and that should have already been installed)
2. Once Spyder has started up, use it to open “fly\_group\_activity\_monitor\_master.py” located in the fly-group-activity-monitor-master folder that was downloaded and extracted. The folder path to the .py file should look something like: “\fly-group-activity-monitor-master\fly\_group\_activity\_monitor\”
3. Change Line 131 “arduino.port = ‘COM’” to include the port number where Arduino is connected (see step 6).

(e.g. “arduino.port = ‘COM5’”)

1. Save changes to “fly\_group\_activity\_monitor\_master.py” and close out of Spyder

Setting up FFMPEG:

1. Download ffmpeg 64-bit static (version: 2015-06-05 git-7be0f48) for the relevant OS (<https://ffmpeg.org/download.html#build-linux>)

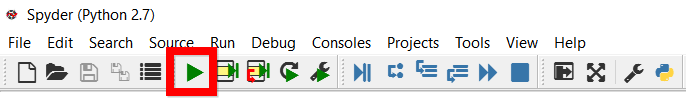
(<https://ffmpeg.org/download.html#build-windows>)

(<https://ffmpeg.org/download.html#build-mac>)

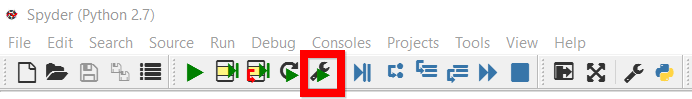
1. Extract the downloaded compressed ffmpeg static file to the Desktop. Rename the extracted folder to “FFMPEG”. Finally, move this renamed folder to the “C:” drive. The path to the ffmpeg executable (as well as ffplay and ffprobe) should be found in “C:\FFMPEG\bin\” if the above steps have been done correctly.

Camera calibration and testing that everything works:

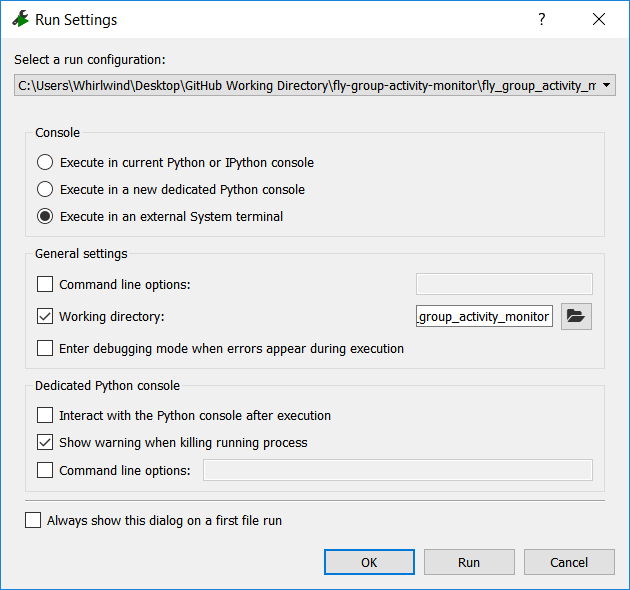
1. Start up Spyder. If using Windows, make sure to start the program as an administrator!
2. Once Spyder has started up, use it to open “fly\_group\_activity\_monitor\_gui.py” located in the fly-group-activity-monitor-master folder that was downloaded and extracted. The folder path to the .py file should look something like: “\fly-group-activity-monitor-master\fly\_group\_activity\_monitor\”
3. Once the “fly\_group\_activity\_monitor\_gui.py” file has been opened press the “Play” button located in the top menu.



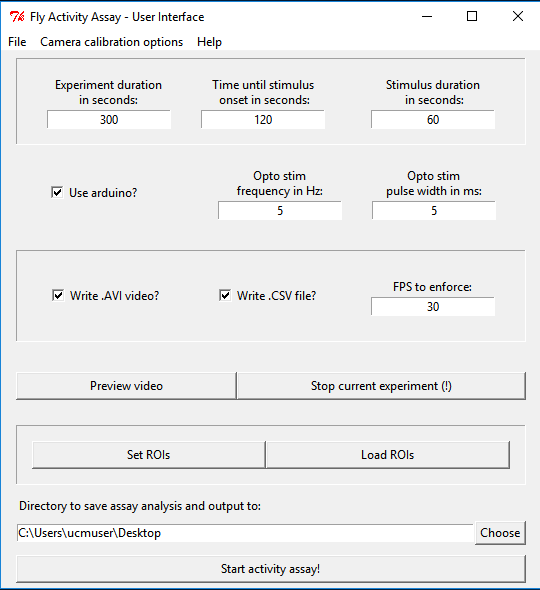
* 1. If this is is not your first time running a python file then you’ll need to click the “Run Settings” button which looks like:



* 1. If this is the first time running a python file you should see the following menu and run the file and you should see a user interface pop up



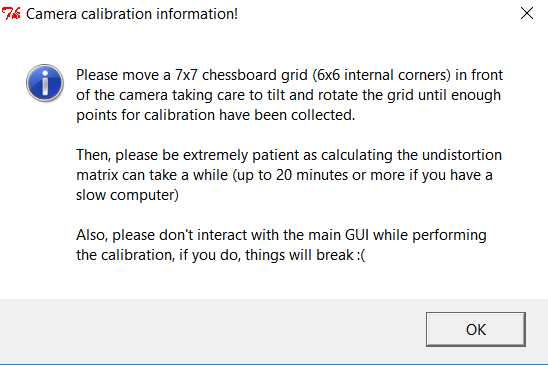
1. Make sure to remember what the current settings are as you may want to revert your run settings for other python programs.
2. The settings for flyGrAM should use “**Execute in an external System terminal**” found under the “Console” subheading.
3. If successfully run, you should see the following user interface pop up:



1. First, check that the camera is operational. To do so, click the “Preview video” button on the interface. (Note: the video window may take some time to show up. DO NOT click the “Preview video” button multiple times)
   1. If the video that shows up is from a laptop webcam or another camera. You will need to go into your “device manager” and “disable” the camera. You can always go back and re-enable it later via the “device manager”
2. Note that the preview video that is initially taken will feature some distortion (straight edges in the video will look curved). To correct for this distortion a camera calibration will need to be run.
3. Print off the 10mm checkerboard found in “\fly-group-activity-monitor\camera\_calibration” and make sure to mount it on a solid flat surface. Make sure there are no wrinkles or distortions of the chessboard!

IMPORTANT: Before running the camera calibration, please make sure that any infrared (IR) filter on the camera lens has been taken off and any IR back-illumination has been turned off. It is also critical that the position of the camera as well as its focus has been optimized. If the distance of the camera from the arenas or the lens focus is changed, then the calibration will need to be performed again.

1. To run the calibration, select the “calibrate camera” option in the menu of the flyGrAM user interface. Read the directions that pop up and make sure to save the resulting calibration file. (Note: If the directions are not informative enough see: <https://www.youtube.com/watch?v=kizO_s-YUDU> for a video example of a similar calibration process)



IMPORTANT: To get a good calibration, it is important to move the checkerboard at a slow but steady rate, taking care to rotate and translate the board so that it ends up at many different positions and angles. Calibration points will show as a rainbow colored array overlaid on the checkerboard. However, if the camera can’t detect all vertices of the checkerboard then only the previous good image will be shown.

NOTE: After collecting enough calibration points, the main “Fly Activity Assay - User Interface” window will likely show “(Not Responding)”. This is normal. Wait until the Python console shows “RMS, camera\_matrix, distortion coefficients, mean error, and reprojection error” before resuming interaction with the user interface.