**Pupil Analysis Code Readme**

Gunvant Chaudhari, July 2019

**Overall flow-through**

Raw Video from PI (.h264)

**↓** 1: *convertMP4.bat*

Video (.mp4)

**↓** 2: *appropriate* *pupilAnalysis .m file*

Pupil Sizes (pixels)

**↓** 3: *filterPupilData.m*

Filtered Pupil Sizes (pixels)

**↓** 4: *processNormalDiams.m*

Pupil Sizes Aligned to trials (pixels) \*not always possible\*

**↓** 5: *copy-paste, etc.*

Export to Excel, Various Analysis (pixels)

**↓** 6: *convert to mm*, *see conversion values below*

Pupil Sizes, Analyzed, in mm

**Step-by-step instructions**

Step 1:

Video (.h264)

**↓** 1: *convertMP4.bat*

Video (.mp4)

* convertMP4.bat is a simple command prompt script found in (C:\Users\PorteraAdmin\Documents\MATLAB\Pupil Analysis) that converts all .h264 raw files in the folder it is in to mp4. You can simply double click and it should run.\
* So, move all the .h264 files you want to analyze to one folder, copy convertMP4.bat to that folder, double click it, and let it run.

Step 2:

Video (.mp4)

**↓** 2: *appropriate* *pupilAnalysis .m file*

Pupil Sizes (pixels)

* Possible options for analysis code files (in C:\Users\PorteraAdmin\Documents\MATLAB\Pupil Analysis):
  + SerialPupilAnalysisTester.m => for processing only 1 file from NRB rig
  + SerialPupilAnalysisStart.m => for processing many files one after another, automatically from NRB rig
  + SerialPupilAnalysisTesterCaImaging.m => for processing only 1 file from Adrianscope rig
* They all are pretty similar in structure and should be decently commentated
* For the “Tester” codes
  + You basically need to find the right threshold for each video using this code (hence the “tester” name) by testing it on small subsets of the frames. And then you exit setup mode by changing one value and run the code on all the frames.
  + Almost all the settings you need to change for each video (except two: SETUP and i) are at the top and in all CAPS
    - Only set A\_FILTER to 1 if the video has background noise (e.g. vertical bars moving); if not, set to 0
    - A\_DISCREP\_THRESH should be around 1.3 to 1.8. Changing it to the higher end might be useful when you see pupils not being recognized as circles often
    - A\_FILTER\_ORDER shouldn’t really change. 3 is default
    - A\_OFFSET\_THRESH needs to be played around with until the first couple of thresholds recognize the correct pupil most of the time and separate pupil from the surrounding darkness/shadows in the eye
      * For NRB rig videos: decrease this value to make pupil more recognizable (makes the cutoff closer to pure black)
      * For adrianscope videos: decrease value to make pupil more recognizable (makes cutoff closer to pure white)
      * Sometimes you just need to play around with a few values before you get one that good enough
      * Sometimes the pupil video is just too noisy or bad angle and shouldn’t be used because it will give bad data
    - RD\_DIR: Read-directory
    - A\_VID\_NAME: video name, without .mp4 extension
  + Start with SETUP variable to 1. This gives you visual feedback that allows you to adjust the thresholds. You turn this to 0 when you are ready to run the full analysis.
  + The “i” variable denotes what frame the code starts on. There is usually some initial brightness which cannot be processed, so the code should start on **frame 100** at runtime. However, during setup, **change this to different frame #s to see how the code runs on different parts of the video.** You can Ctrl+c to stop runtime repeatedly after you are satisfied that your subset of frames is responding well to threshold or if you want to change some setting and retry.
  + \* Please make sure SETUP is 0 and i starts at 100 when you are ready to start full analysis\*
  + Files will output to C:\Users\PorteraAdmin\Documents\MATLAB\Pupil Analysis\Data
  + *Note all the output variables from here on out say pupil diameter, but it is actually pupil radius that is being recorded. That was my bad.*
* For the “SerialPupilAnalysisStart” code (only for NRB rig)
  + Use “SerialPupilAnalysisTester” to find the right values for the top 8 variables. Note that eye-center and crop-rect are commented out in this file. This is intentional because they need to be calculated from your input during runtime of the code.
  + When you are satisfied with the values for one video, copy paste the 8 variables from the top of the tester code to the beginning of the start code. Also copy paste the eye-center and crop-rect variable VALUES from the workspace to the commented out sections of the other stuff you copied. And then you can uncomment this section.
  + Copy paste an “errorxx = serialpupilanalys….”
    - This command starts a function that analyzes based on the parameters you fed it.
    - You only need to change the “xx” for each video, as it records if there was any error in the serial pupil analysis for each video. I’ve never had an error, but this is just in case.
  + Repeat for each video.

Step 3:

Pupil Sizes (pixels)

**↓** 3: *filterPupilData.m*

Filtered Pupil Sizes (pixels)

* Code is pretty self-explanatory; excludes some pupil sizes due to bad quality (determined from a number of measures)
* Also scales down pupil measurements to 1 measurement per 0.1 s.
* Just change read directory (rd\_dir) and save directory (sv\_dir). It will process all files in the rd\_dir, so you may need to move your raw files to a separate directory

Step 4:

Filtered Pupil Sizes (pixels)

**↓** 4: *processNormalDiams.m*

Pupil Sizes Aligned to trials (pixels) \*not always possible\*

* Code is also well commented, so pretty self-explanatory
* Note that the trial start frames need to be manually determined for the NRB rig videos
* Adrianscope videos
  + For adrianscope videos, it is a little more complicated, as you can find the true frame start time from the cpu clock time stamps. However, the PI and computer cpu clocks run at a slightly different rate, so there is some calibration values taken during behavior, stored in META\_raspicalib
  + The PI timestamps for each frame are stored in a text file with the same name as the video. Not every frame will have a timestamp. Some frames will have two timestamps.
  + To parse out frame stamps from the text file, use parseFrameStamps.m (located in the same folder as all the other code)
    - The code is commented, so it should be pretty easy to use. However, it is prone to error sometimes if the text file has a weird ending

Step 5:

Pupil Sizes Aligned to trials (pixels) \*not always possible\*

**↓** 5: *copy-paste, etc.*

Export to Excel, Various Analysis (pixels)

* Analyze as you wish. All the variables within the output mat files should be well-labeled, so what they contain should be easy to determine.

Step 6:

Export to Excel, Various Analysis (pixels)

**↓** 6: *Convert to mm*, *see conversion values below*

Pupil Sizes, Analyzed, in mm

* Conversion values:
  + 1280x720 video (not scaled down) from NRB Rig
    - From analyzed “pupil diameters” (actually pupil radii, see note above in step 2)
    - Factor for Pupil radii in pixels to pupil radii in mm: **0.026458**
    - SO from your values to pupil diameters, conversion is **0.026458\*2**
  + 640x480 video (not scaled down) from adrianscope Rig
    - Similarly pupil radii have been mislabeled as diameters
    - Factor for Pupil radii in pixels to pupil radii in mm: **0.029695**
    - SO from your values to pupil diameters, conversion is **0.029695\*2**
  + Note when you are combing videos from both rigs, be sure to convert to mm before running any stats as they have different conversions