Localization of RNAscope dots to specific nucleus morphology, Tracked through a Stack

CALM Facility –Todd Fallesen

November 14, 2019

Abstract

This project was used for **Neuronal programming by microbiota enables environmental regulation of intestinal physiology** by *Yuuki Obata, Alvaro Castano Medina, Stefan Boeing, Ana Bon-Frauches, Werend Boesmans, Candice Fung, Mercedes Gomez de Aguero, Bahtiyar Yilmaz, Rita Lopes, Almaz Huseynova, Todd Fallesen, Stuart Horswell, Muralidhara Rao Maradana, Pieter Vanden Berghe, Andrew Murray, Brigitta Stockinger, Andrew Macpherson, and Vassilis Pachnis.*

This is a CellProfiler Pipeline used to detect RNAscope dots localized into nuclei, tracked through an image stack. The cells are stained with DAPI and the RNAScope spots are in GFP. Images were first channel separated using BioFormats and a quick plugin for FIJI. For this CellProfiler pipeline, Images **MUST** have their filenames changed so that the Z stack variable is two digits (i.e. instead of *Z=1,Z=2,Z=3..Z=10*, we have *Z=01, Z=02, Z=03...Z=10)*. If this is not done, the tracking between slices will not occur correctly. This renaming can be done using a python scrip with regex.

Method for publication: RNAScope signal intensity was co-localized to nuclei using CellProfiler. Briefly, Images were split into two channels, RNAScope signal and "Nueron-stain". Cell nuclei in the 'nueron-stain' channel are segmented and tracked through an entire image stack, to eliminate double counting. The RNAScope signal intensity inside each nucleus is then integrated through the image stack in slices where the nucleus was identified, to eliminate counting RNAScope signal that was present in the same (x,y) position as the nucleus, but either above or below the nucleus in the Z-stack.

1 Image Preparation

Olympus **.oir* files can not be opened by CellProfiler. As such, we need to process the images before we can load them into CellProfiler. We can do this easily using FIJI.

- Install the macro *save_all_windows.ijm*
- Drag **.oir* file into FIJI
- Check the boxes under **Split into separate windows** for *Split Channels* and *Split focal planes*.
- Once all the windows have opened, run the macro *save_all_windows.ijm*.
- At the prompt, enter a folder name for the files to be saved in, and click *ok*

1.1 Python renaming

Once the files are split, open up the *python_renamer.py* file in a text editor, and change the *sourcedir* variable to the directory that the split up files are in, and change the *prefix* variable to the first part of the filename up until and including *Z=*. The file can be then run, and should rename the files. The program doesn't always run on Mac computers, so do check to make sure the files are renamed.

When the images are saved into a directory and renamed, they are ready to be used in the CellProfiler pipeline.

2 CellProfiler overview

CellProfiler is an image analysis program that steps through a series of modules in a pipeline to bring in an image, process it, and extract data from it. Modules that perform an operation (like smoothing, or measurement) need an input, which can either be an image or an **Object**. An **object** is a feature that is detected in an image, such as a cell nucleus, a bead, etc. Many modules also have an output, which can be another image or a series of objects detected. Other modules only perform measurements, or comparisons between objects. At the end of the pipeline, the results are exported to a spreadsheet. Modules in the pipeline can toggle between showing output to the screen or not by clicking on the icon of a human eye next to the module name. Modules can be disable or enabled by clicking on the box next to their name (see figure 1).

3 Pipeline Overview

The Following Modules are used in this order in the pipeline:

- Images
- MetaData
- NamesAndTypes
- Groups
- Erosion
- IdentifyPrimaryObjects
- IdentifyPrimaryObjects
- MeasureObjectIntensity
- MeasureObjectSizeShape
- RelateObjects
- SplitOrMergeObjects
- TrackObjects
- OverlayObjects
- MeasureObjectIntensity
- OverlayObjects
- RescaleIntensity
- ExportToSpreadsheet
- DisplayDataOnImage
- SaveImages
- SaveImages
- SaveImages

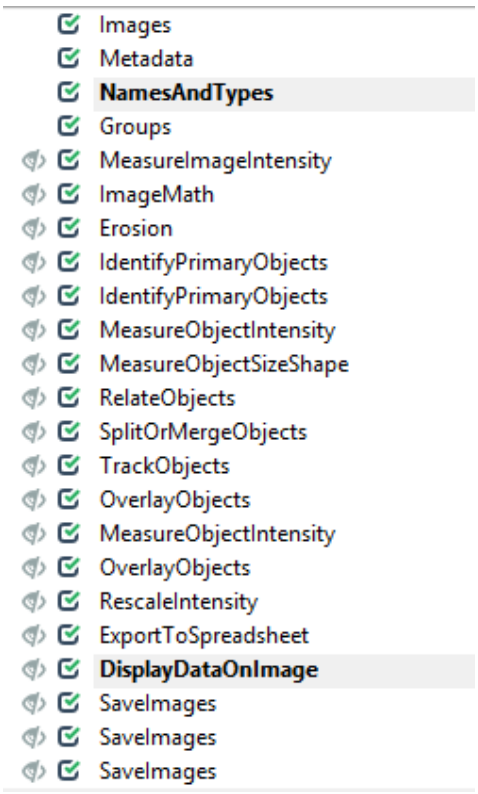

Figure 1: Overview of modules as seen in CellProfiler

4 File processing modules

The **Images, Metadata, NamesAndTypes** and **Groups** modules are involved in inputting images and files.

In the **Images** module, drag and drop the directory with the split files into the Window marked **Drop files and folders here**. See figure (2)

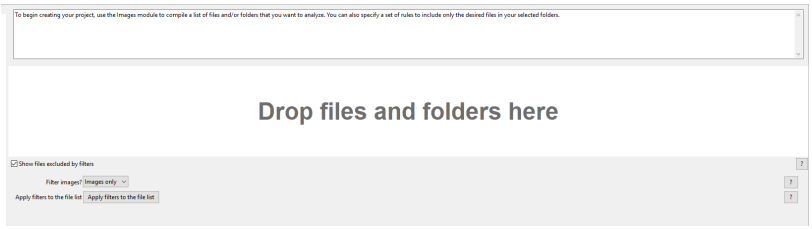

Figure 2: Example of Drag and Drop Screen

In the **Metadata** module, check *No* on EXTRACT METADATA? question. The files imported from the directory don't have any usable metadata for this particular problem.
This pipeline uses the **NamesAndTypes** module to determine which images are DAPI and which images are RNAScope spots (see figure (3)). To do this we set:

Match All of the following rules:

File does contain **C=0** (User sets **C=0** here)

Name to assign these images **Cells** (user sets **Cells** here)

After **C=0** is set up, click *Add another image* and set up the new rules in the same manner with **C=1** for *Spots*, which will bring in the image from channel 1 that are the RNAScope marked dots.

In both cases, the title of the file contains the string $C{=}n$ where $n = 0$ or 1 referencing the original channel from the Olympus microscope. See figure (3). **MAKE SURE THAT THE FILES HAVE Z=00, Z=01, Z=02 AND NOT Z=0, Z=1, Z=2**. If the Z slices are ordered by single integers, the tracking **WILL FAIL.**

If there is a need to bring in further channels at a later date, the same procedure can be used to bring in additional channels and label them.

Under the **Groups** module, click *No*. There isn't a need to group the images here.

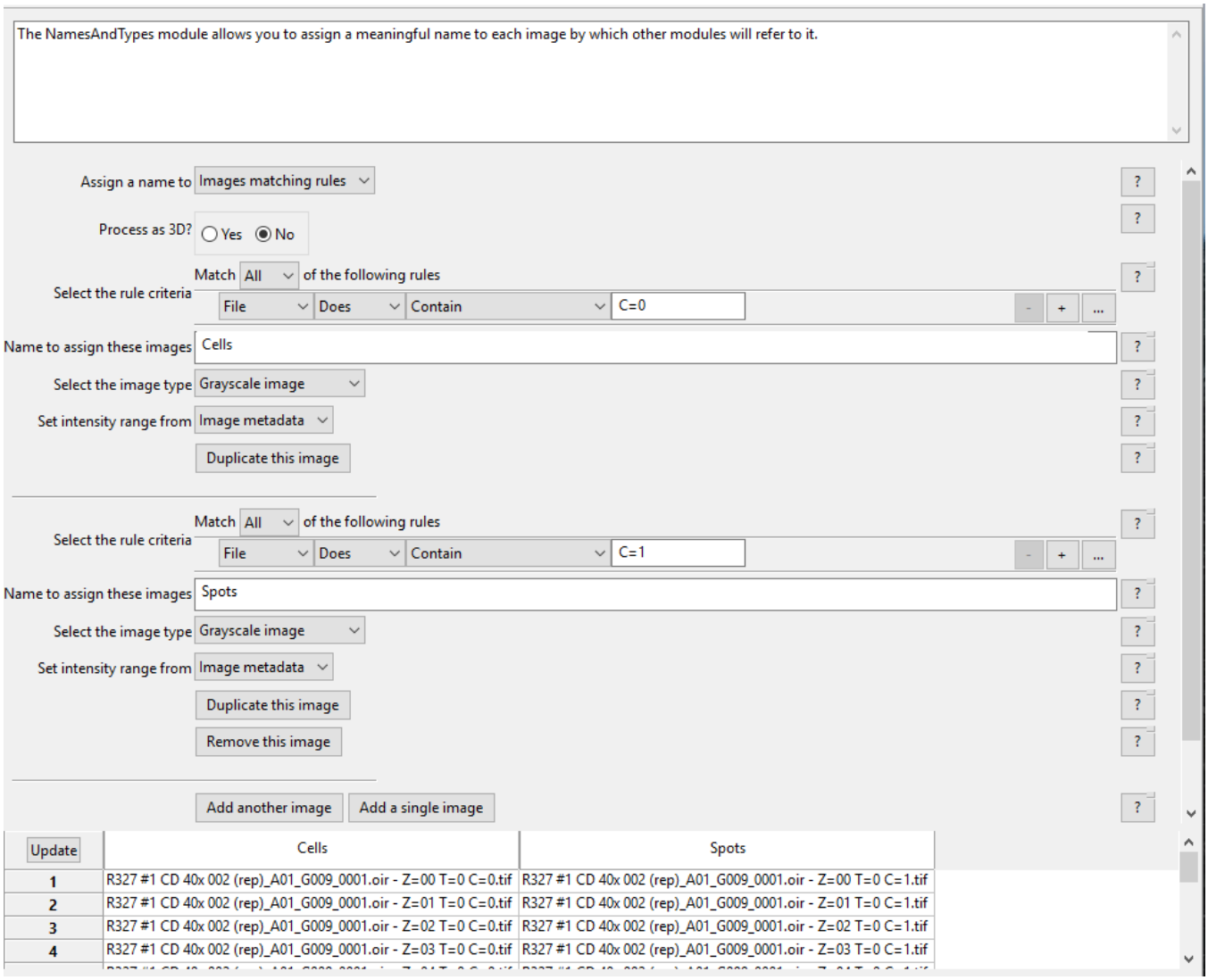

Figure 3: NamesAndTypes parameter screen with values entered to bring in necessary data. Note that the Z=00, Z=01 in the file names

5 Image Preparation Modules

To better detect individual cells, we need to perform background subtraction on each slice, which removes some of the bleed through and out of focus light. To do this we use two modules: **MeasureImageIntensity** and **ImageMath**. The general idea is to subtract 10% of the intensity off of the image.

MeasureImageIntensity

INPUT is *Cells*. OUTPUT is *Measurements on Cells image*. Select NO on *Measure the intensity only from areas enclosed by objects?*. This module will measure the intensity of the image, so that we can use that information in the next module, **ImageMath**.

ImageMath

There are two INPUTS for this module, the *Cells* image and the *MaxIntensity* from the *Cells Measurement* from the **MeasureImageIntensity** module. The OPERATION is *Subtract*. The OUTPUT is *Cells_background_subtract*. To use the module, set the first *Image or Measurement* to *Cells* and *Multiply the first image by* 1.0. For the second *Image or Measurement* select *Measurement* and under select *Category: Intensity, Measurement: MaxIntensity, Image:Cells. Multiply the second image by* ***0.1***. See figure 4.

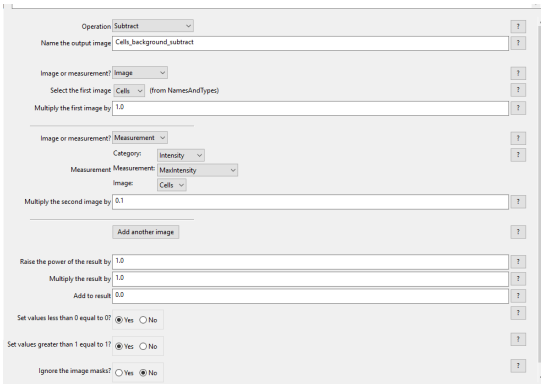

Figure 4: ImageMath parameter screen with values entered to do background subtraction

Erosion

INPUT is *Spots*. OUTPUT is *Spots_eroded*. The **Erosion** modules are used to prepare the images for further analysis. The **Erosion** module erodes away the bright parts of the image, making it easier to differentiate objects. The value set was a square pattern at a pixel width of 2.

6 Identification and classification modules

The following modules of the pipeline are the real engine of the analysis.

IdentifyPrimaryObjects

INPUT is *Cells_background_subtract*. OUTPUT is *Cells_Primary_Objects*. This module detects objects in the image. In this instance, the module is running on the background subtracted cells image, so detecting, without bias on shape, the cells in the image. The first thing that is set is the typical DIAMETER of the objects, which is set for *15 to 120 pixel units*. Objects out of this range are discarded, as the object is often two cells overlapping, which can not be accurately counted. Objects touching the border of the image are discarded as it is impossible

to know their actual size

Thresholding must be done in order to detect objects. In thresholding, all pixels above a certain THRESHOLD brightness will be given the maximum white value, while pixels with a brightness below threshold will be given the value of zero. Under advanced settings, we are using the Otsu method for Thresholding with autodetermination of the THRESHOLD SMOOTHING SCALE, with *Three Classes* (i.e. low intensity, medium intensity, and high intensity pixels). The medium intensity pixels are often out of focus or out of plane cells, so they are set to BACKGROUND. The THRESHOLD CORRECTION FACTOR, set to 1.2 is used to multiply the autodetermined threshold by a value, to tweak the value empirically. Select *No* under SPEED UP BY USING LOWER-RESOLUTION IMAGE TO FIND LOCAL MAXIMA

Since the image detected has many round cells that are the objects of interest, the METHOD TO DISTINGUISH CLUMPED OBJECTS and the METHOD TO DRAW DIVIDING LINES BETWEEN CLUMPED OBJECTS is set to *Intensity* as that tells the computer to base the segementation on regions where the intensity diminishes, as where cell nuclei transition into cytoplasm. This can be switched to *shape* if later experiments have a large amount of cells that are closely grouped. Segmenting by shape segments the cells by looking for 'dimples' between the objects. In preliminary tests, there was less than a 1% difference by segmenting by shape or intensity.

IdentifyPrimaryObjects

INPUT is *Spots_eroded*. OUTPUT is *Spots_Primary_Objectss*. This module detects objects in the image. In this instance, the module is running on the eroded spots image, so detecting, without bias on shape, the spots in the image. The first thing that is set is the typical DIAMETER of the objects, which is set for *2 to 30 pixel units*. Objects out of this range are discarded. Objects touching the border of the image are discarded as it is impossible to know their actual size. Advanced settings are not used in this instance of the module.

MeasureObjectIntensity

INPUTS are *Spots_eroded and Spots_Primary_Objects*. This module measures the intensity of the *Spots_Primary_Objects* in the *Spots_eroded* image. These values will be later used to determine the total intensity of the spots in each individual cell, in the **RelateObjects** module.

MeasureObjectSizeShape

INPUT is *Cells_Primary_Objects*. Do not calculate *Zernike features*. This module measures a variety of parameters of the detected objects, including area, radius, form factor and eccentricity. There is no output from this module, but the measured parameters will be used later to label the cells.

7 Relation Modules

The follow modules are thr crux of the pipeline. They find all the spots that are in a particular cell and merge them to find the total intensity inside that cell. The cells are also tracked so that the intensity present in an individual cell in each slice can be totaled over all slices

RelateObjects

PARENT OBJECTS are *Cells_Primary_Objects* (the detected cells per slice) and the CHILD OBJECTS are **Spots_Primary_Objects**. The OUTPUT OBJECTS are *Spots_inside_cells*. This module simply finds all child objects (spots) that are contained within the border of the parent objects (the cell).

SplitOrMergeObjects

INPUT is *Spots_Primary_Objects*. OUTPUT is *SpotsInCells_merged*. The OPERATION of this module is to *merge* the spots together that are all contained within the same cell. The MERGING METHOD is *Per-parent* which in this case, the parent object is the *Cells_Primary_objects*. The OUTPUT TYPE is *disconnected* so that the computer doesn't create polygon around the perimeter of the linked spots, rather, just treats them all as one object, even though they are not necessarily touching.

TrackObjects

INPUT is *Cells_Primary_Objects*. OUTPUT IMAGE is *TrackedCells*. This module keeps track of the cells between slices. The OBJECTS TO TRACK are *Cells_Primary_Objects*. The TRACKING METHOD is *overlap* and the MAXIMUM DISTANCE TO CONSIDER MATCHES is set at *50 pixels*. The maximum distance parameter can be changed readily by the user, as can the tracking method, which may enhance tracking.

The module is set to SAVE A COLOR CODED IMAGE *TrackedCells*, and the DISPLAY OPTIONS are *Color and Number* which will print both a different color and the Tracking ID to the saved picture.

OverlayObjects

INPUT is *Cells*. The *Output Image* is *SpotsInCells_merged_overlay*. The OBJECTS that are overlaid on the image are are *SpotsInCells_merged*. The OPACITY is set to *0.02* so that the text is visible on the image. This image is just a sanity check so that we can see where the detected merged spots are in relation to the cells.

MeasureObjectIntensity

INPUTS are *Spots_eroded and SpotsInCells_merged*. This module measures the intensity of the merged spot objects in the *Spots_eroded* image. These values will be later used to determine the total intensity of the spots in each individual cell, in the **RelateObjects** module.

OverlayObjects

INPUT is *Cells*. The *Output Image* is *Overlay_Image_Spots_Inside_Cells*. The OBJECTS that are overlaid on the image are are *Spots_inside_cells*. The OPACITY is set to *0.02* so that the text is visible on the image. This image is just a sanity check so that we can see where the detected **un-merged** spots are in relation to the cells.

RescaleIntensity

INPUT is *SpotsInCells_merged_overlay*. The OUTPUT IMAGE is *Overlay_rescaled*. For both the METHOD TO CALCULATE THE MINIMUM INTENSITY and METHOD TO CALCULATE THE MAXIMUM INTENSITY a CUSTOM method is used, where the range is 0 TO 0.08. This intensity is rescaled so that the saved image is more readily viewable on a normal computer screen.

8 Modules for Data export and saving

ExportToSpreadsheet

Under SELECT THE COLUMN DELIMITER select *Comma* for comma separated variable spreadsheets. Choose a folder to export the data to, which should be specific to the experiment the user is running. **Do not change the filename prefix**. MATLAB reads the output from this module to make a final spreadsheet, and is looking for files with the prefix *MyExpt_*. The module is set to overwrite existing files without warning, so **be sure to keep results in separate folders**. The module is also set to *Yes* under PRESS BUTTON TO SELECT

MEASUREMENTS. The measurements that are selected are numerous, and the user shouldn't change these without understanding the results. Changing which variables are exported might cause the MATLAB script used to group data to fail. The export window is shown in figure 5The Export tree is given in figure 6

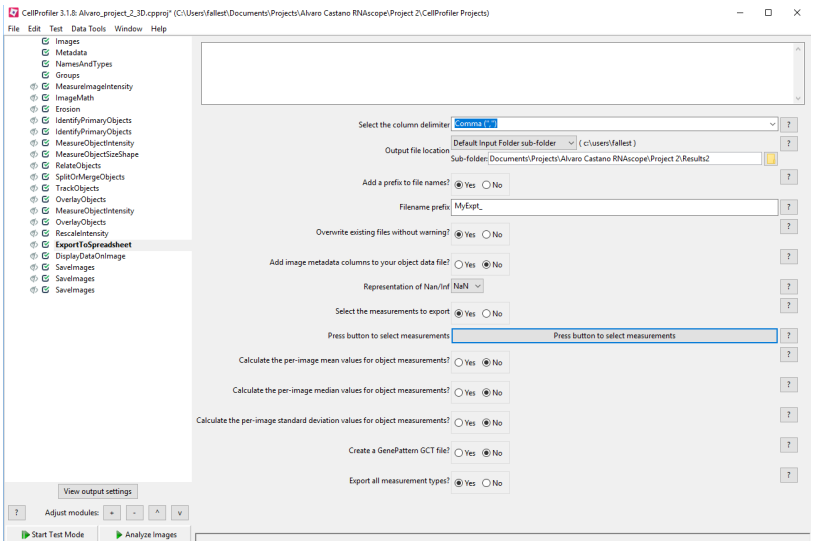

Figure 5: Export window

DisplayDataOnImage

INPUT is *Cells_Primary_objects*. SELECT THE IMAGE ON WHICH TO DIPSLAY THE MEASUREMENTS is *Cells*. The OUTPUT (where image that has the measurements displayed) is *Cells_labelled_number*. This module will label the cells with their Primary Object ID from the **IdentifyPrimaryObjects** module which found the cells. This module will be a sanity check to the user can look at the cells and make sure that the cell ID's label the cells they think they should.

This module is very testy. If the pipeline is loaded and this module is red-exed out, it is because there is a character error somewhere in there, and the screen looks like figure 7 then the pipeline will not work. To change the variables, click on where it says *object* and click down a menu item, and more things will pop up. Continue doing this until you can set the module to have the same parameters as figure 8.

SaveImages

SELECT IMAGE TO SAVE is *Overlay_rescaled*. TEXT TO APPEND TO THE IMAGE NAME is *Merged_Overlay_*. This module saves an 8-bit tiff file to a folder of the users choice. The IMAGE NAME FOR THE FILE PREFIX is *Cell*. This module will OVERWRITE EXISTING FILES WITHOUT WARNING, and SAVES EVERY CYCLE. **This module saves an image of the merged spots overlaid onto the image of the cells**

SaveImages

SELECT IMAGE TO SAVE is *Cells_labelled_number*. TEXT TO APPEND TO THE IMAGE NAME is *Object_number_*. This module saves an 8-bit tiff file to a folder of the users choice. The IMAGE NAME FOR THE FILE PREFIX is *Cell*. This module will OVERWRITE EXISTING FILES WITHOUT WARNING, and SAVES EVERY CYCLE.**This module saves an image of the cell ID number per slice overlaid onto the image of the cells**

SaveImages

SELECT IMAGE TO SAVE is *TrackedCells*. TEXT TO APPEND TO THE IMAGE NAME is *_Tracked*. This module

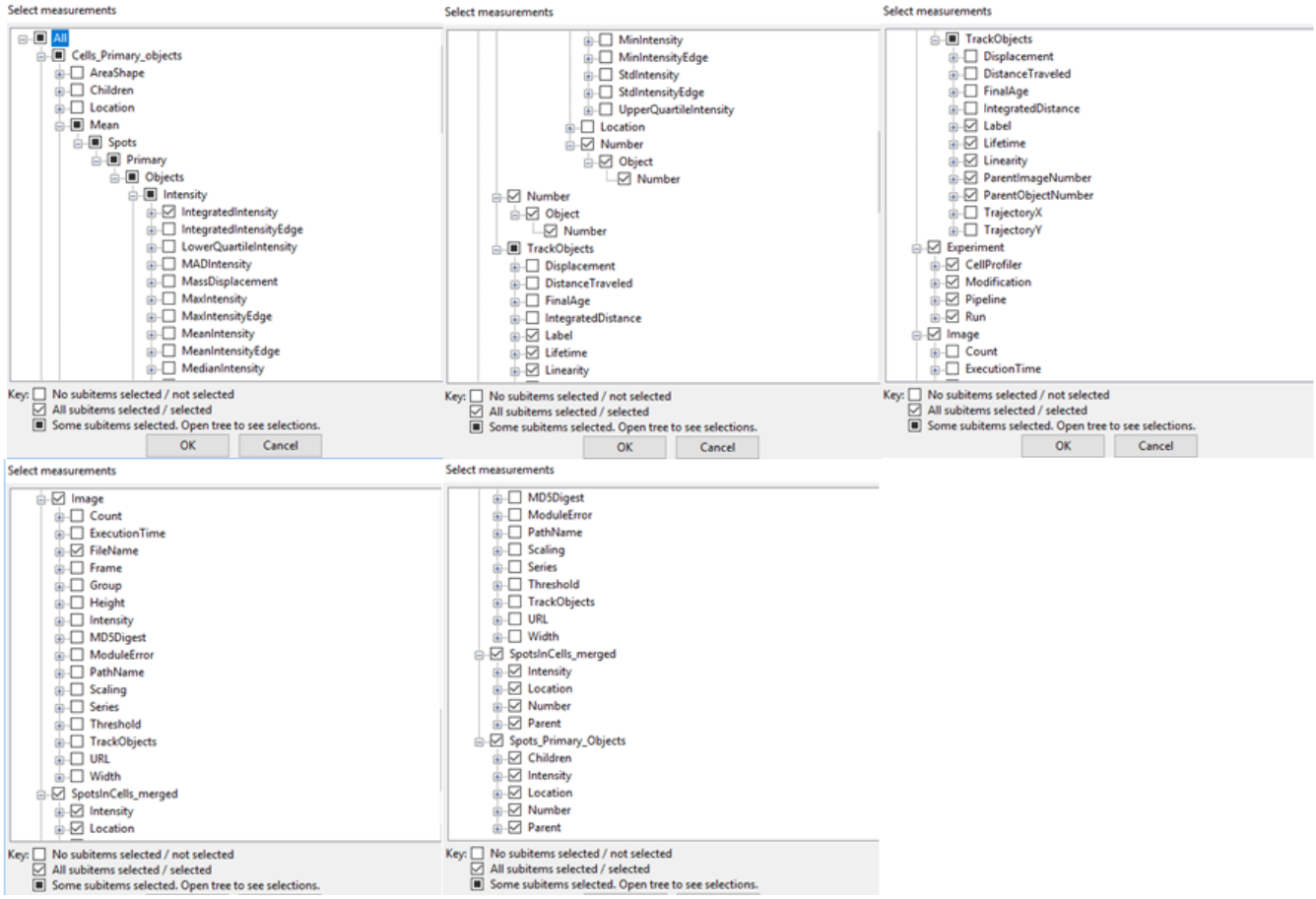

Figure 6: Tree of variables that are exported

saves an 8-bit tiff file to a folder of the users choice. The IMAGE NAME FOR THE FILE PREFIX is *Cell*. This module will OVERWRITE EXISTING FILES WITHOUT WARNING, and SAVES EVERY CYCLE. **This module saves an image of the cells with their tracking ID overlaid onto the image of the cells**

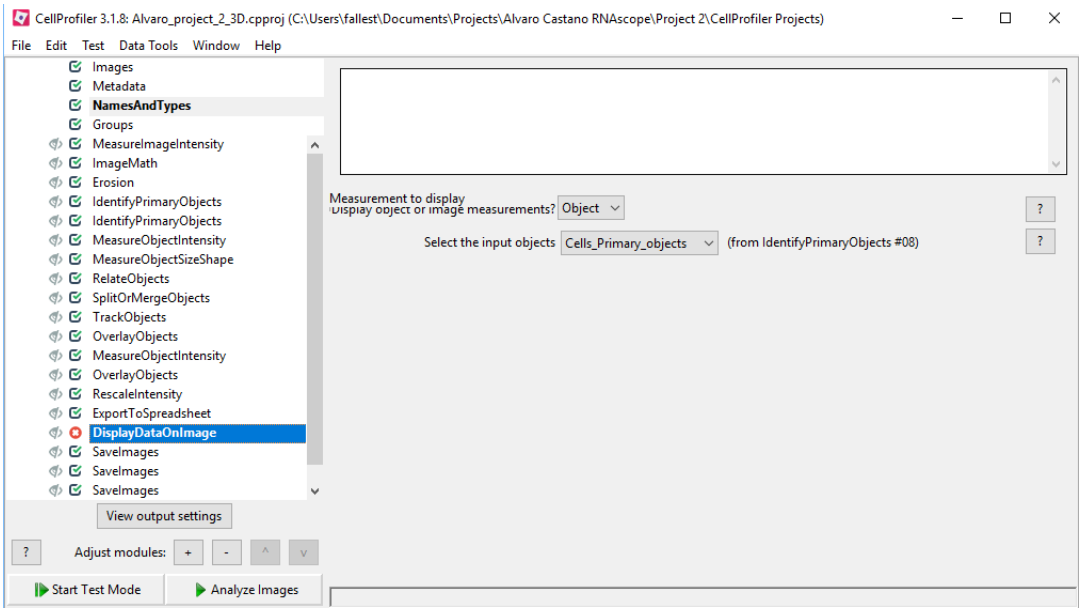

Figure 7: Example of corrupt DisplayDataOnImage module

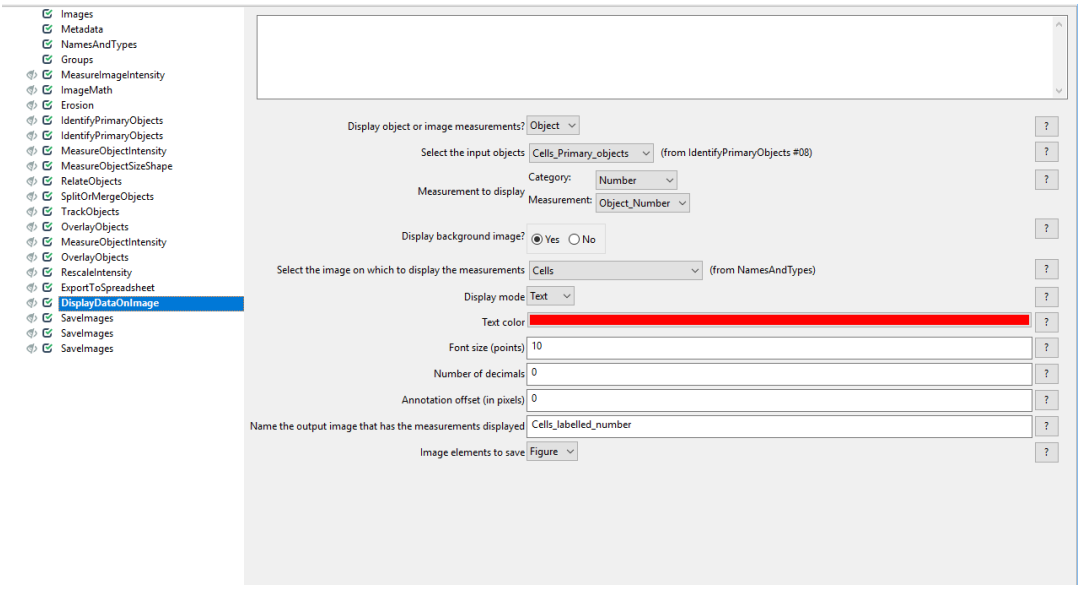

Figure 8: Export window

9 MATLAB Post Processing

The script *Cell_Profiler_3D_result_analysis.m* was written to take the data exported from CellProfiler in multiple csv files and group them into something usable. The output of CellProfiler gives us the total intensity of the merged spots co-localized to a cell per slice, and also the TrackingID of each cell as the computer moves through the stack. For any given cell, the Tracking ID will not change throughout the stack, though the Cell ID will. The MATLAB script matches the Tracking ID to a specific cell in a specific slice, and then looks at the child object of that cell, which will be the merged spots present in that cell, in that slice. The MATLAB script then saves the integrated intensity for each cell in each slice, linked to the Tracking ID for that cell, so that the intensities of a given cell are totalled up over all slices. The MATLAB script then exports an Excel spreadsheet with two columns, **ID**, which is the Tracked Cell ID, and **Total_intensity**, which is the total intensity of all merged spots for that cell, over all slices.

The code for the script is well commented and should be easily understood, but the user hopefully doesn't have to edit the code.

9.1 Running the MATLAB script

To run the MATLAB script, open MATLAB and choose *Open* to open a script. Open *Cell_Profiler_3D_result_analysis.m* and click on the Green *Run* icon. This will open a dialog box where you must choose the folder with the *MyExpt_* csv files that were produced by CellProfiler are.

After clicking ok, the script will run and process the data, before asking you to select a folder to save the Excel output in. By default, the folder should be the same as the CellProfiler results were, and the default filename will be the filename of the first image file in the stack, without the channel, timepoint and z information. If this is ok, click *Save*. At the same time, a *.mat* file will be saved with the same name as the Excel output. This **.mat* file will the entire data structure with all the linked data from the experiment, and may be useful for further analysis.