ImageJ method: Code and directions

Non-annotated code

Below is the code used to measure images of baboons. See next section for the annotated version that explains the function of each line.

```
imageName = getTitle();
//setTool("rectangle");
makeRectangle(1662, 924, 1620, 540);
run("Crop");
run("Set Measurements...", "area centroid bounding display
invert redirect=None decimal=3");
run("Split Channels");
imageCalculator("Subtract create", imageName + " (green)",
imageName + " (red)")
setAutoThreshold("Default dark");
//run("Threshold...");
run("Analyze Particles...", "display exclude summarize add");
```

Annotated code

Below is the annotated code. Each line is followed by a description of what that line does and any adjustments you might need to make when adapting for your study system. These annotations are marked by a pound sign and are in italics.

```
imageName = getTitle();
#Centers the script on the image that it's about to process

//setTool("rectangle");
#Selects the rectangle tool.
#You could perform this auto-crop with other tools, eg oval tool
#If using hand-cropped images, delete this line

makeRectangle(1662, 924, 1620, 540);
#Creates a rectangle that will encompass the laser spots
#You can make this rectangle smaller (see Recording macro below)
#If using hand-cropped images, delete this line

run("Crop");
#Crops the image to the coordinates indicated by previous line
#If using hand-cropped images, delete this line

run("Set Measurements...", "area centroid bounding display
invert redirect=None decimal=3");
#This was not necessary when processing baboon images, but
  #was needed when processing hand-cropped gorilla images.
#Under some conditions, ImageJ will flip the y-axis in the final
  #coordinates unless this line is included
```

```
#If the y-axis is not flipping, including this line is
  #unnecessary but it will not change your measurements if you
  #include it

run("Split Channels");
#Splits the image into red, green, and blue channels

imageCalculator("Subtract create", imageName + " (green)",
imageName + " (red)")
#Subtracts the red channel from the green channel

setAutoThreshold("Default dark");

//run("Threshold...");
#Performs an automatic light threshold, separating pixels into
  #white or black.
#If the script is working, you will see white or red dots where
  #the laser points were.
#We use following inputs and buttons:
  #Default & Red
  #Dark background checked
  #Click Auto

run("Analyze Particles...", "display exclude summarize add");
#Record the coordinates of the dots.
#We use the pre-set inputs:
  #Size (pixel^2): 0-Infinity; Circularity: 0.00-1.00
  #Show: Nothing, though this is flexible.
  #Display results, Summarize, Add to Manager, Exclude on edges
```

Directions for implementing and re-recording code in ImageJ software

Setting up

1. Make sure the photos you want to measure are all in one folder [*Input folder*].
2. After the batch processing is finished, ImageJ will save an output image for each input image. Create a separate *Output folder* for those images.

Running the code

3. Open ImageJ
4. Select *Process* → *Batch* → *Macro*
5. Select the correct *Input* and *Output* folders
6. Paste the code into the blank box. Note that you can save code for later use by selecting *Save*, and retrieve it by selecting *Open*.
7. Select *Process*
8. ImageJ will now perform the batch process on your chosen set of input files and display two windows:

    a. Summary window: Records the number of laser spots identified for each image (see Count column)
    b. Results window: Records the locations of each identified laser spot (see X and Y columns); will have one row for each laser spot identified
9. If you would like to look at the photos after they have been modified by ImageJ, you can view them in the output folder.

<u>Troubleshooting: Incorrect number of laser spots identified</u>

1. Move images that identified the incorrect number of last spots into a new folder.
2. Hand-crop these images to include only the laser spots and intermediate space. We recommend using a simple image viewing software such as Preview for this; use keyboard shortcuts to go through photos very quickly. Save the file.
3. Run these hand-cropped images through the script again, but delete the lines that perform the auto-crop (see Annotated code above).
4. Any images that still didn't work must be hand-measured.

<u>Recording macro</u>

An advantage of ImageJ is that you can record a macro while testing out image processing steps. The recording contains the code for each step, so you can easily test out a whole new set of code, or test our small portions and copy the code for the steps that work well into your main code. For example, this is a good way to adjust the size of the auto-crop.

1. Select *Plugins* → *Macros* → *Record*
2. You can now perform any image processing steps you would like to try. ImageJ will automatically record the code for that specific image in the Recorder window, which you can later copy and paste into a text file.

   If you are making slight adjustments (e.g., changing the size of the auto-crop), you can copy those specific lines of code into your text file. If you are re-recording the whole code, the directions below will enable you to turn the code from the Recorder window into new code for batch image processing:

3. The code generated is specific for the one specific image you opened and performed the actions on, so it needs to be slightly modified in order to be applied to an entire folder of images:
    a. If it exists, delete the first line generated by the Recorder that says "`open(`…"
    b. Now add this line:
    ```
    imageName = getTitle()
    ```
    Now that you have added this line, every time you want the code to loop through the entire set of images, you can use `imageName` in place of an individual image's name. Note that image names will be in quotation marks in the Recorder window; `imageName` should <u>not</u> be in quotation marks.
    c. Replace every mention of the specific image name with `imageName`

d. In places where there is something additional to the image title, for example:

```
selectWindow("P7150089.JPG (green)");
```

Adjust like this:

```
selectWindow(imageName + " (green)");
```

e. Unless changed, when this code is applied to the batch, the measurements are updated for every new picture and the program doesn't collect them all together. So if the last line looks like this:

```
run("Analyze Particles...", "display clear
summarize add");
```

Delete the `clear` and replace with `exclude`, so your code should now look like this:

```
run("Analyze Particles...", "display exclude
summarize     add");
```