

Seed Color ImageJ Manual (unfinished version)

Ian McNish
University of Minnesota
last modified 15-Feb-2017

Contents

1	Introduction	1
2	Using ImageJ	1
2.1	Downloading Fiji	1
2.2	Downloading tutorial materials from GitHub	1
2.3	Downloading Ellipse Split	3
2.4	Open the script	4
2.5	Setting parameters in ImageJ	5
2.5.1	Choosing the file pathway	6
2.5.2	Enter control patch coordinates	6
2.5.3	Enter seed coordinates	7
2.5.4	Enter control patch reflectance data	7
2.5.5	Changing initial photo file type	8
3	Running the script in Fiji	8
4	Using R	9
5	Tips for taking photos	9
6	ImageJ process walkthrough	9
6.1	Section 3: importing the photo and splitting channels	10
6.2	Section 4: radiometric calibration	11
6.3	Section 6: background masking	12
6.4	Section 7: using Ellipse Split to create seed ROIs	12
7	FAQ	13
8	Bibliography	13
Index		14

1 Introduction

This document is meant to serve as a manual for the ImageJ[1] script seed_color_imagej and R[2] script seed_color_r created by Ian McNish. This script was developed to collect color and size data from oat seed samples. This script was originally developed on an Apple MacOS version 10.11 system using Fiji version 2.0.0. Testing was also completed on a Windows system. In this manual I will explain how this script can be used to analyze seed photos using example photos available on . I will also attempt to explain how the script works so that other researchers can modify my code to fit their own needs.

2 Using ImageJ

2.1 Downloading Fiji

I recommend using Fiji to run this script. Fiji can be found at this website <https://fiji.sc>. It is possible to use the script on a vanilla version of ImageJ, but it does require you to separately download the ImageJ macro 'Ellipse Split'. This is easily accomplished using the 'Update' feature in Fiji. Instructions for using using 'Ellipse Split' in ImageJ can be found at http://imagej.net/Ellipse_split.

2.2 Downloading tutorial materials from GitHub

All the materials used in this tutorial can be found at https://github.com/mcnis003/image-analysis/tree/master/seed_color_tutorial. I will likely develop new versions of this script in the future. The newest version of these scripts can be found at <https://github.com/mcnis003/image-analysis.git>.

The materials can be downloaded directly from the website. First, navigate to the webpage provided above. Then, select the 'Clone or Download' button on the left side of the page.

The screenshot shows a GitHub repository page for the user 'mcnis003' with the repository name 'image-analysis'. At the top, there are buttons for 'Watch' (0), 'Star' (0), and 'Fork' (1). Below the header, there are tabs for 'Code' (selected), 'Issues' (0), 'Pull requests' (0), 'Projects' (0), 'Wiki', 'Pulse', 'Graphs', and 'Settings'. A description below the tabs reads 'Scripts for image analysis using Stereo Pipeline, ImageJ, SQL, and R languages.' There are buttons for 'Edit', 'imagej', 'r', 'phenotyping', 'sql', and 'Manage topics'. Below this, there are statistics: 18 commits, 1 branch, 0 releases, 1 contributor, and MIT license. A 'Clone or download' button is prominently displayed. The main area lists files and commits:

File/Commit	Date	Time Ago
mcnis003 pull 13-feb-2017 Merge branch 'master' of https://github.com/mcnis003...	13-feb-2017	Latest commit 754b9ab 22 hours ago
old_versions	13-feb-2017	additions 22 hours ago
seed_color_tutorial	13-feb-2017	additions 22 hours ago
LICENSE		Create LICENSE 6 days ago
README.md		readme.md version 1 7-feb-2017 7 days ago
headrow_color_v1.txt		version 1 15 days ago
seed_color_imagej_v09.txt	13-feb-2017	additions 22 hours ago

Then select the 'Download ZIP' button.

mcnis003 / image-analysis

Code Issues Pull requests Projects Wiki Pulse Graphs Settings

Scripts for image analysis using Stereo Pipeline, ImageJ, SQL, and R languages.

imagej r phenotyping sql Manage topics

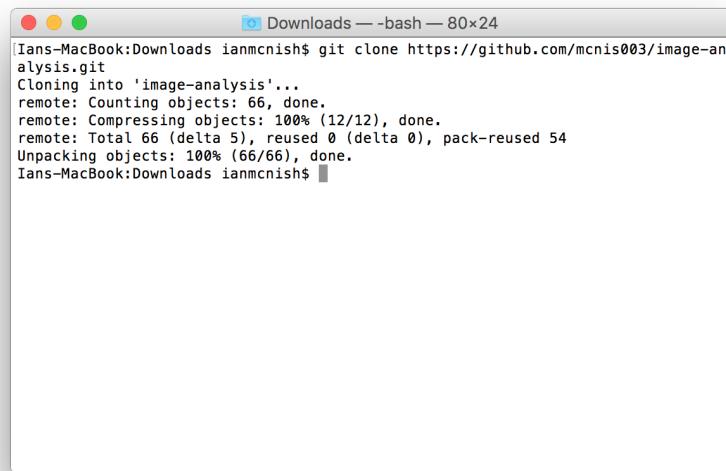
18 commits 1 branch 0 releases 1 contributor MIT

Branch: master New pull request Create new file Upload files Find file Clone or download

old_versions	13-feb-2017 additions	Clone with HTTPS Use SSH	
seed_color_tutorial	13-feb-2017 additions	https://github.com/mcnis003/image-anal...	
LICENSE	Create LICENSE	Open in Desktop	Download ZIP
README.md	readme.md version 1 7-feb-2017	7 days ago	
headrow_color_v1.txt	version 1	15 days ago	
seed_color_imagej_v09.txt	13-feb-2017 additions	22 hours ago	

Alternatively you can also download the files using Terminal commands. A similar system is available on Windows, but I will not supply instructions here. First open . I normally do this using the 'Spotlight' feature in MacOS 10.11. You may need to refer to other sources to fully understand Terminal. I will only provide instructions on how to download materials from in this manual. While in an appropriate directory use the following commands:

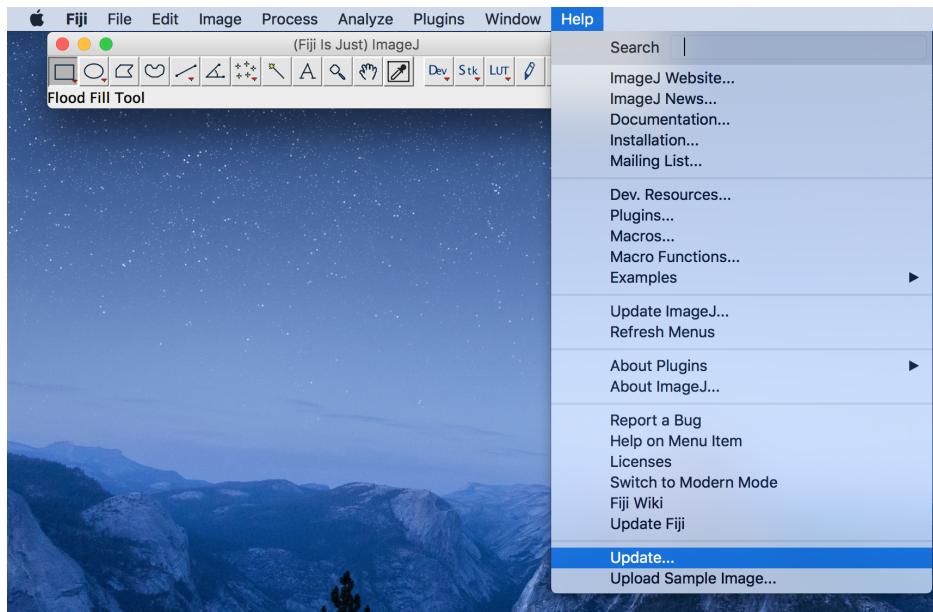
```
git clone https://github.com/mcnis003/image-analysis.git
```



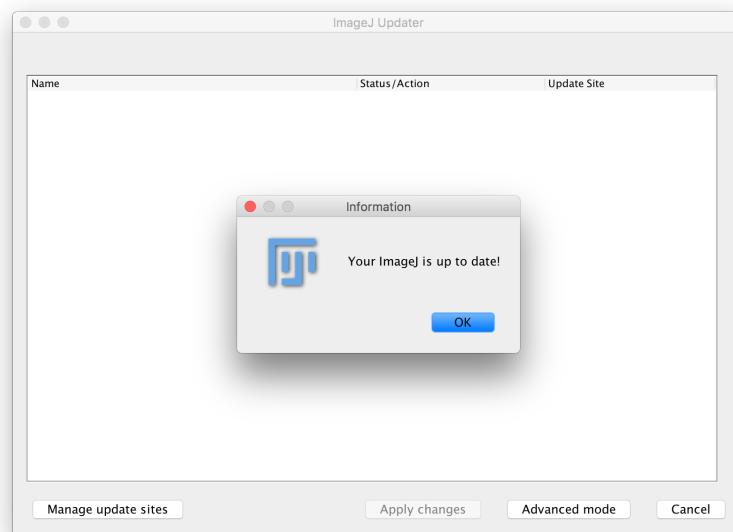
2.3 Downloading Ellipse Split

Downloading ‘’ is easiest when using Fiji. ‘Ellipse Split’ works in vanilla ImageJ but the installation process is slightly more calculated. I will not provide instructions for downloading ‘Ellipse Split’ for use in vanilla ImageJ.

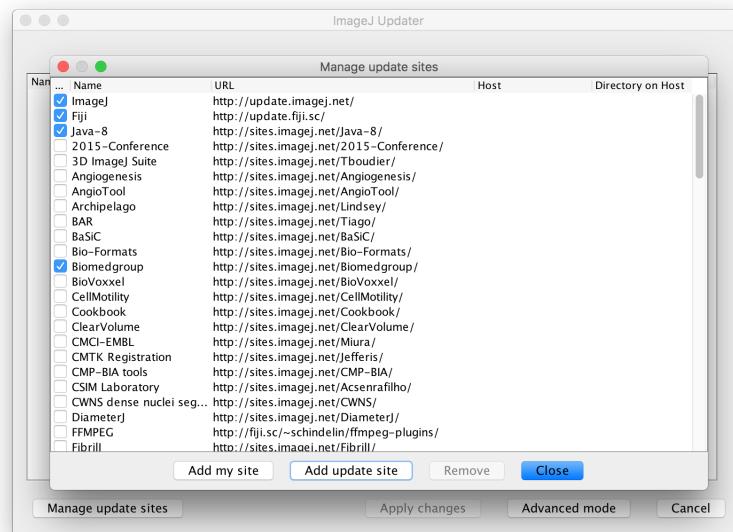
1. To download ‘Ellipse Split’, navigate the the Fiji ‘Help’ menu and select ‘Update...’



2. After Fiji has completed updating press ‘OK’. Then select ‘Manage update sites’.



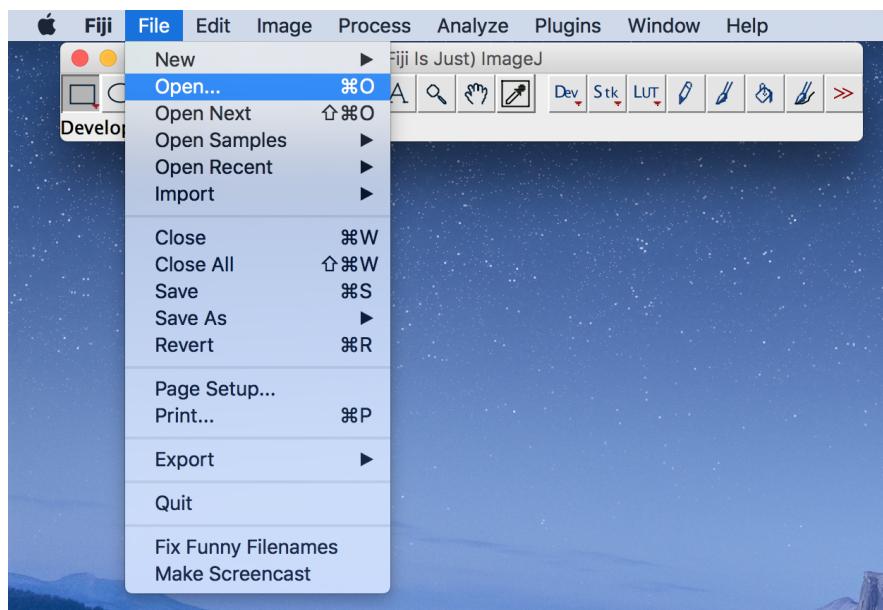
3. Select ‘’ from the list and select ‘Add update site’. ‘’ should then be available in you ‘Macro’ menu.



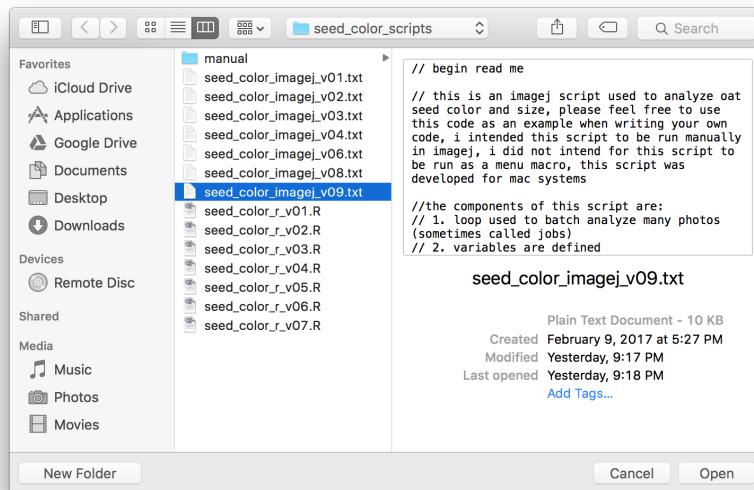
2.4 Open the script

Now that we have Fiji, Ellipse Split, and the materials from downloaded we are ready to open the script in . Do this through the open command in Fiji.

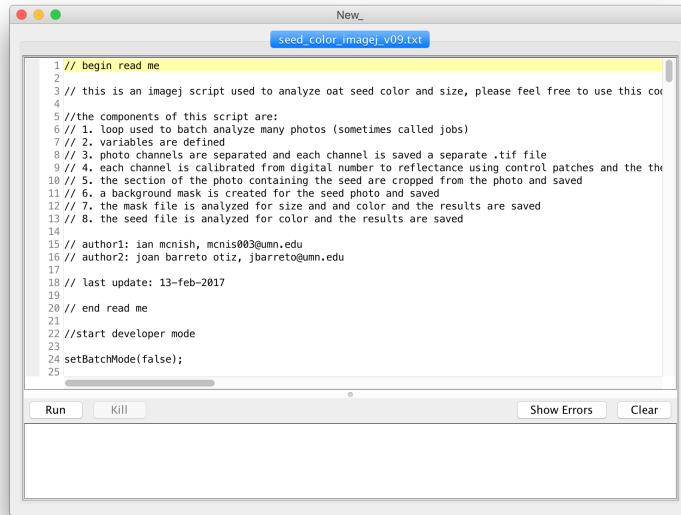
1. Select ‘Open’ in the ‘File’ menu.



- Select the script you wish to open.



- The script will then open in 's editing environment. The entire script may be run using the 'Run' button or by pressing 'command'+‘R’ on Mac. To run a portion of the code select the lines you wish to run a press ‘command’+‘shift’+‘R’.



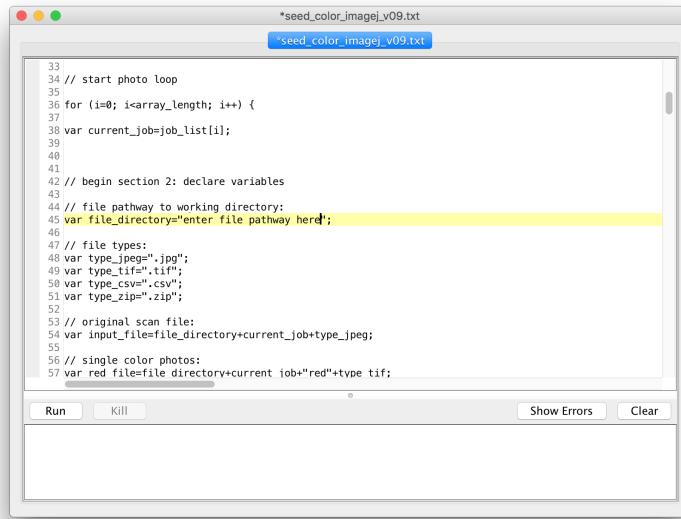
2.5 Setting parameters in ImageJ

Parameters other than the file directory should not need to be changed to run the Tutorial photos available on . However, you will definitely need to change the parameters for photos you collect. The

script works by dividing the image into separate zones. Each zone contains elements that are used for different processes. The position of the zones will depend on how you arrange your items in the photo, the equipment you use to take the photo, and the settings of your equipment. I suggest you test a couple different arrangements, and then strictly stick to the arrangement you think is most optimal.

2.5.1 Choosing the file pathway

You will need to change the directory defined on line 45 of the script to the directory containing your photos. The result files will also be written to this directory. The behavior of this part of the script is similar to the setwd() command in R.



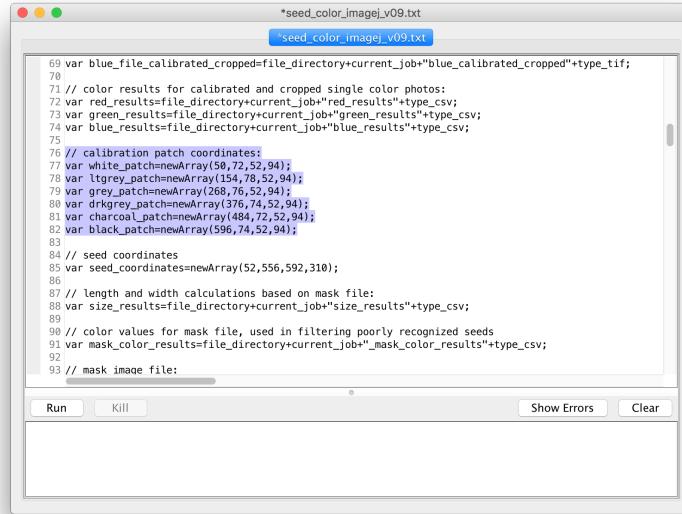
```

33
34 // start photo loop
35
36 for (i=0; i<array_length; i++) {
37
38 var current_job=job_list[i];
39
40
41
42 // begin section 2: declare variables
43
44 // file pathway to working directory:
45 var file_directory="enter file pathway here!";
46
47 // file types:
48 var type_jpeg=".jpg";
49 var type_tif=".tif";
50 var type_csv=".csv";
51 var type_zip=".zip";
52
53 // original scan file:
54 var input_file_directory+current_job+type_jpeg;
55
56 // single color photos:
57 var red_file=file_directory+current_job+"red"+type_tif;

```

2.5.2 Enter control patch coordinates

The coordinate of you control patches must be entered in this part of the script. These must be entered as an array because each point in the array is specifically referenced later in the script. If you use more or less control patches you will need to modify this section and the calibration sections later in the script. You will need to determine if your application requires .



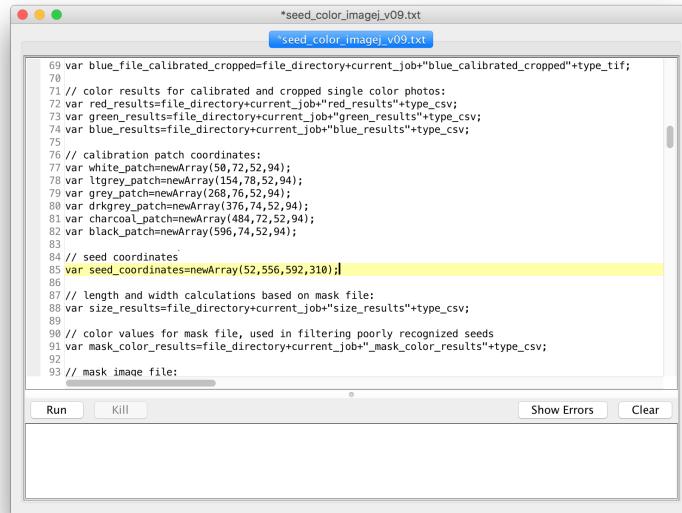
```

69 var blue_file_calibrated_cropped=file_directory+current_job+"blue_calibrated_cropped"+type_tif;
70
71 // color results for calibrated and cropped single color photos:
72 var red_results=file_directory+current_job+"red_results"+type_csv;
73 var green_results=file_directory+current_job+"green_results"+type_csv;
74 var blue_results=file_directory+current_job+"blue_results"+type_csv;
75
76 // calibration patch coordinates:
77 var white_patch=newArray(50,72,52,94);
78 var ltgrey_patch=newArray(154,78,52,94);
79 var grey_patch=newArray(268,76,52,94);
80 var drkgrey_patch=newArray(376,74,52,94);
81 var charcoal_patch=newArray(484,72,52,94);
82 var black_patch=newArray(596,74,52,94);
83
84 // seed coordinates
85 var seed_coordinates=newArray(52,556,592,310);
86
87 // length and width calculations based on mask file:
88 var size_results=file_directory+current_job+"size_results"+type_csv;
89
90 // color values for mask file, used in filtering poorly recognized seeds
91 var mask_color_results=file_directory+current_job+"_mask_color_results"+type_csv;
92
93 // mask image file:

```

2.5.3 Enter seed coordinates

Enter the coordinates of the seeds here. Seeds that are on the edge of the region you draw will not be included in the analysis if you use ‘‘ with the default settings from my script.



```

69 var blue_file_calibrated_cropped=file_directory+current_job+"blue_calibrated_cropped"+type_tif;
70
71 // color results for calibrated and cropped single color photos:
72 var red_results=file_directory+current_job+"red_results"+type_csv;
73 var green_results=file_directory+current_job+"green_results"+type_csv;
74 var blue_results=file_directory+current_job+"blue_results"+type_csv;
75
76 // calibration patch coordinates:
77 var white_patch=newArray(50,72,52,94);
78 var ltgrey_patch=newArray(154,78,52,94);
79 var grey_patch=newArray(268,76,52,94);
80 var drkgrey_patch=newArray(376,74,52,94);
81 var charcoal_patch=newArray(484,72,52,94);
82 var black_patch=newArray(596,74,52,94);
83
84 // seed coordinates
85 var seed_coordinates=newArray(52,556,592,310);|
86
87 // length and width calculations based on mask file:
88 var size_results=file_directory+current_job+"size_results"+type_csv;
89
90 // color values for mask file, used in filtering poorly recognized seeds
91 var mask_color_results=file_directory+current_job+"_mask_color_results"+type_csv;
92
93 // mask image file:

```

2.5.4 Enter control patch reflectance data

Enter the control patch expected reflectance data here. You will need to do this three times (lines 152, 193, and 227).

```

149 run("Measure");
150
151 makeRectangle(charcoal_patch[0],charcoal_patch[1],charcoal_patch[2],charcoal_patch[3]);
152 run("Measure");
153
154 makeRectangle(black_patch[0],black_patch[1],black_patch[2],black_patch[3]);
155 run("Measure");
156
157 // calibrate and save the calibrated red image
158
159 run("Calibrate...", "function=[Straight Line] unit=[Optical Density] text2=[.95 .78 .63 .47 .32 .12] show");
160 close();
161
162 saveAs("Tiff",red_file_calibrated);
163 close();
164
165 run("Clear Results");
166
167 open(green_file);
168
169 // extract average Dn values for each control patch
170
171 setTool("rectangle");
172
173 makeRectangle(white_patch[0],white_patch[1],white_patch[2],white_patch[3]);

```

2.5.5 Changing initial photo file type

The Tutorial variable uses the .jpg file type to match the photos available on GitHub. The script has also been tested with TIFF format photos. You will need to change the extension here to match the type of photos you use here.

```

41
42 // begin section 2: declare variables
43
44 // file pathway to working directory:
45 var file_directory="enter file pathway here";
46
47 // file types:
48 var type_jpeg=".jpg";
49 var type_tif=".tif";
50 var type_csv=".csv";
51 var type_zip=".zip";
52
53 // original scan file:
54 var input_file=file_directory+current_job+type_jpeg;
55
56 // single color photos:
57 var red_file=file_directory+current_job+"red"+type_tif;
58 var green_file=file_directory+current_job+"green"+type_tif;
59 var blue_file=file_directory+current_job+"blue"+type_tif;
60
61 // calibrated single color photos:
62 var red_file_calibrated=file_directory+current_job+"red_calibrated"+type_tif;
63 var green_file_calibrated=file_directory+current_job+"green_calibrated"+type_tif;
64 var blue_file_calibrated=file_directory+current_job+"blue_calibrated"+type_tif;
65

```

3 Running the script in Fiji

Once all your parameters are set you can run the script by pressing the ‘Run’ button in the development window or by pressing ‘command’+’R’ on Mac.

```

302 close();
303 run("Clear Results");
304
305 open(green_file_calibrated_cropped);
306 roiManager("open", roi_file);
307 roiManager("measure")
308 saveAs("Results",green_results);
309 roiManager("reset")
310 close();
311 run("Clear Results");
312
313 open(blue_file_calibrated_cropped);
314 roiManager("open", roi_file);
315 roiManager("measure")
316 saveAs("Results",blue_results);
317 roiManager("reset")
318 close();
319 run("Clear Results");
320
321 } // end photo loop
322
323 // end developer mode
324
325 setBatchMode(false);
326
327

```

Run Kill Show Errors Clear

4 Using R

```

18 #author: Ian McNish, mcnis003@umn.edu
19 #institution: University of Minnesota
20 #last modified: 13-feb-2017
21
22 #end readme
23
24 #1. set working directory, load libraries, import data files ----
25 setwd("enter working directory here")
26 library(lattice)
27 red_result_files = list.files(pattern="*red_results.csv")
28 green_result_files = list.files(pattern="*green_results.csv")
29
30 #2. process each file
31 for (i in 1:length(red_result_files)) {
32   red_file = red_result_files[i]
33   green_file = green_result_files[i]
34
35   #3. read in the data
36   red_data = read.csv(red_file)
37   green_data = read.csv(green_file)
38
39   #4. calculate seed size
40   red_size = red_data$size
41   green_size = green_data$size
42
43   #5. plot the data
44   plot(red_size, green_size, xlab="Red Seed Size", ylab="Green Seed Size", main="Red vs Green Seed Size")
45
46   #6. add a regression line
47   abline(lm(green_size ~ red_size))
48
49   #7. calculate correlation coefficient
50   cor_coeff = cor(red_size, green_size)
51
52   #8. print the correlation coefficient
53   print(paste("Correlation coefficient: ", cor_coeff))
54 }

```

5 Tips for taking photos

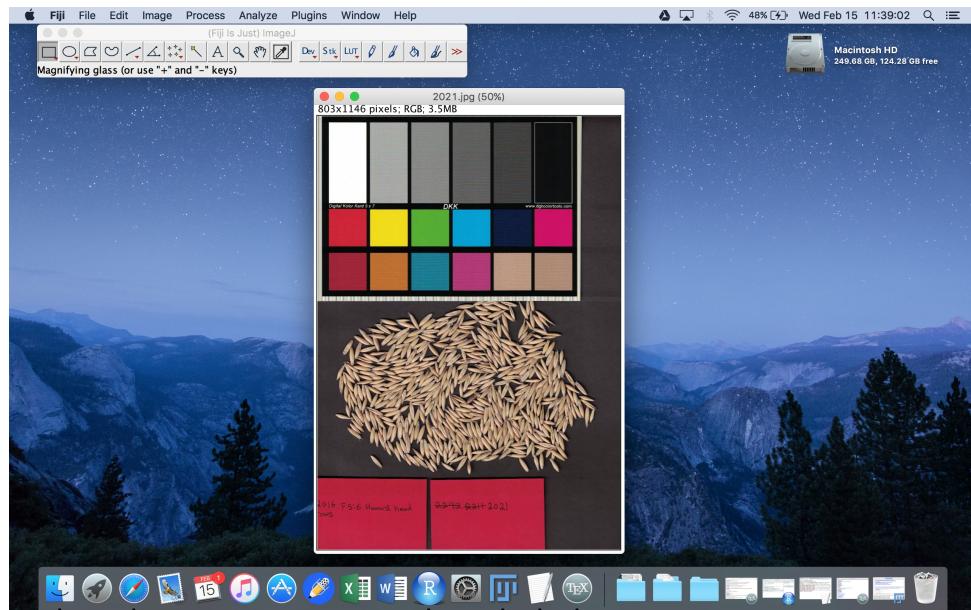
enter information about using this script

6 ImageJ process walkthrough

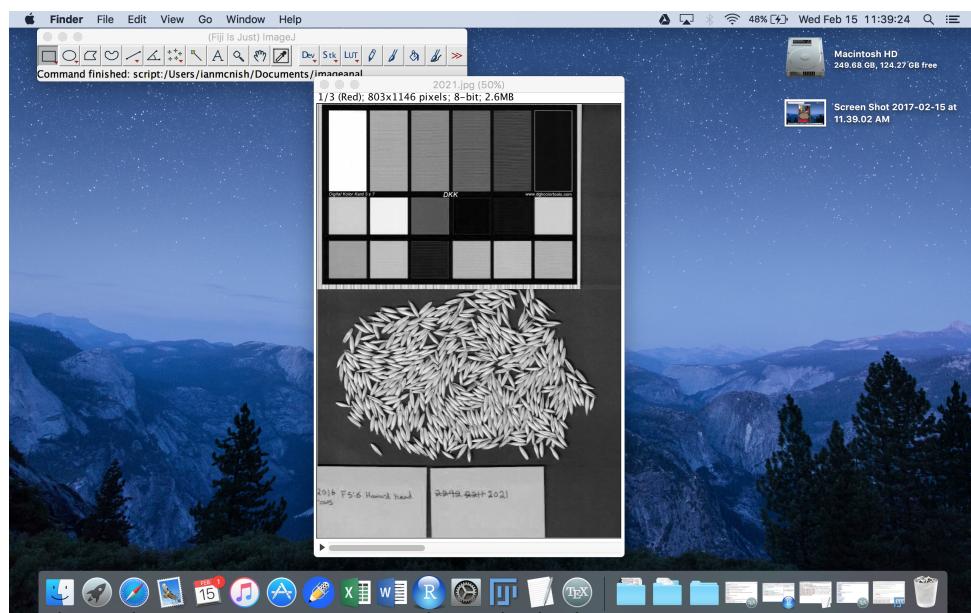
enter information about using this script

6.1 Section 3: importing the photo and splitting channels

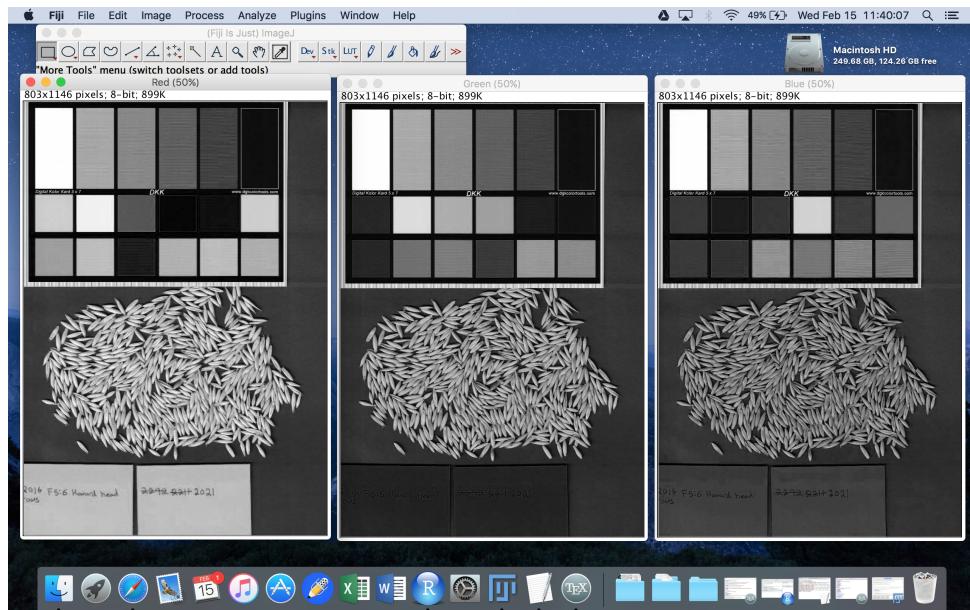
1. The full color photo is opened.



2. The photo is then split into R, G, and B channels in a stack.

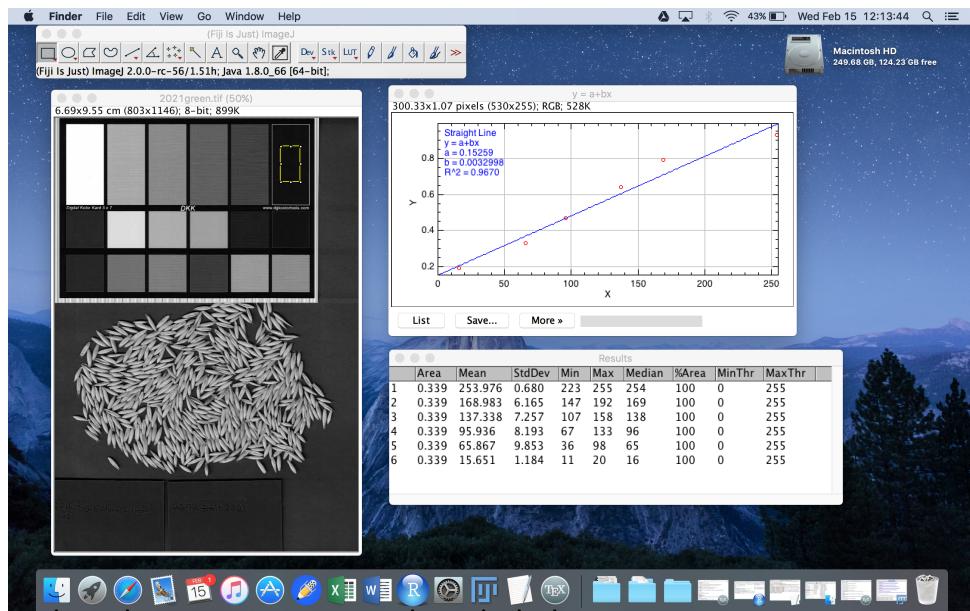


3. The stack of channels is then split into separate photos and each photo is saved. All photos are saved in TIFF format from this point onwards.



6.2 Section 4: radiometric calibration

Radiometric calibration is accomplished using a standard photo card placed in each photo combined with data collected with sensitive laboratory equipment. A detailed explanation of this method is provided by <http://ieeexplore.ieee.org/document/7098353/?reload=true>. Ultimately you will need to decide if you need to calibrate your photos. Radiometric calibration should be less important in circumstances with very controlled image acquisition. I provided the code in this script so that other researchers could use it if it is important in their process.



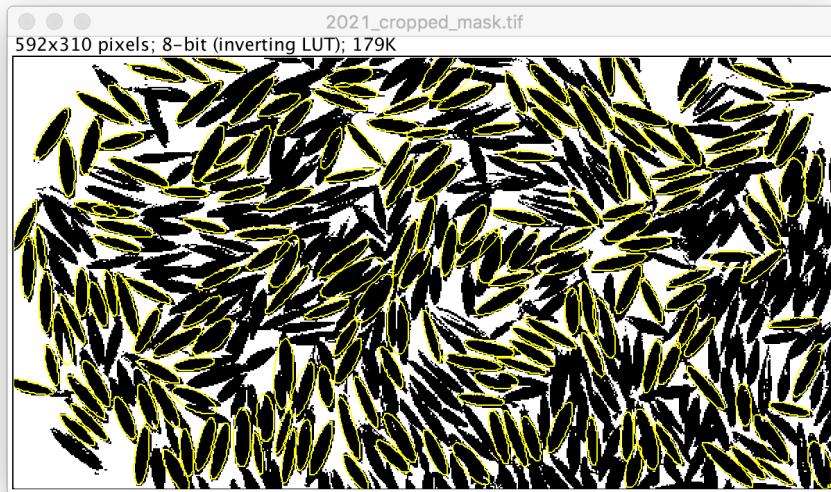
6.3 Section 6: background masking

For our application, background masking was not required. However, the mask is used by ‘Ellipse Split’ to produce the seed ROIs. I also use color data collected from the mask file to find instances where a seed ROI contains multiple seeds or large amounts of background. Masking may be necessary if your background is irregular.



6.4 Section 7: using Ellipse Split to create seed ROIs

Because our seeds are ellipse shaped, I used the ImageJ macro “`Elliptical Selection`” to find ROIs. If you are analyzing objects of another shape this method of locating the object will probably not work. For other shapes of object I recommend using the “`Find Maxima`” command.



7 FAQ

1. Q: how long does it take to analyze a photo with this program?
A: Our application uses a 30mb photo with 100 objects and it takes 4 seconds per photo. The program is much faster if batch mode is enabled, but unfortunately I have been having problems with batch mode lately.
2. Q: How accurate is the script at finding seeds? A: In our experience, if items in the photo are well arranged the script will accurately find 85% of the seeds, 5% of the seeds will be misidentified in some way, and 10% of the seeds will not be identified.

8 Bibliography

- [1] SCHNEIDER, C. A., RASBAND, W. S., AND W, E. K. Nih image to imagej: 25 years of image analysis. *Nature Methods* 9, 7 (2012), 671–675.
- [2] TEAM, R. D. C. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2008. ISBN 3-900051-07-0.

Index

Analyze Particles, 12
Biomedgroup, 4
Ellipse Split, 3, 4, 7, 12
Fiji, 4, 5
GitHub, 1, 2, 4, 5
radiometric calibration, 6
Terminal, 2