Seed Counting App User Manual

# Setup

NB: Jupyter is assumed to be installed and working, together with the Python3 libraries numpy, matplotlib, pandas and the OpenCV cv2 library. Installing these and making sure that they all work together properly is not straightforward, and is often error-prone. *A much more user-friendly version of this seed counting app has been recently ported to R.*

Load the Jupyter notebook found at /notebooks/analysis.ipynb.

## Loading the images

By default, images are loaded from the /images/ folder. This can be changed if needed by modifying cell 8:



The images should have a circular reference shape at the top, which has a known area that can be used to derive the areas of the seeds via shape similarity.

While the seeds do not strictly need to be distributed in rows and columns, do note that the script implicitly orders them based on their y-axis height on the page, so the output will list the seeds accordingly. E.g. the seeds will either be listed in order of appearance going up the image or going down the image (depending on how the underlying OpenCV algorithm decides to identify the contours).

## Tuning the parameters

The main parameter that can be tuned a priori is the area of the reference object, which can be modified in cell 5 of the Jupyter notebook:



e.g. here the area of the reference circle is measured (in the real-world) to be approximately 706.86 square millimetres. The script works by counting the pixels in the reference object and the seeds. If we know the ratio of the pixels to the area in the reference object, then since the seeds are (roughly) circular, we can exploit similarity to calculate the real-world area of the seeds in square millimetres.

# Running the script

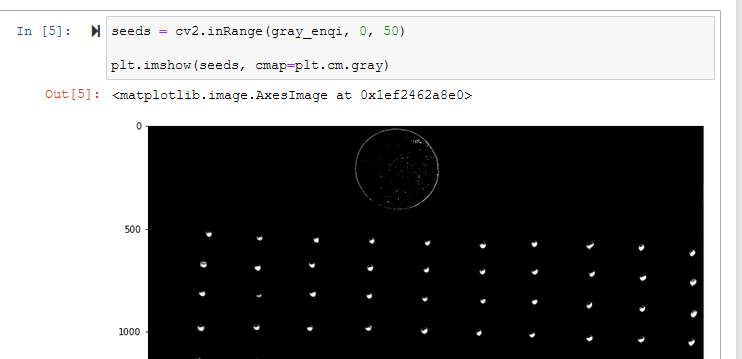
The script requires ad hoc management of the black/white thresholds to ensure optimal detection of the seeds (depending on ambient light, shadows, and so on). The Jupyter notebook provides visual feedback of each of the steps in the process for a test image, so that thresholds can be calibrated.

## Adjust thresholds to isolate seeds

Run cells 1 and 2 of the Jupyter notebook to confirm that the images are loaded appropriately. You should see your designated test image appear in the output view. (Here the image is /images/test.jpg but you can change this to any image you like.)



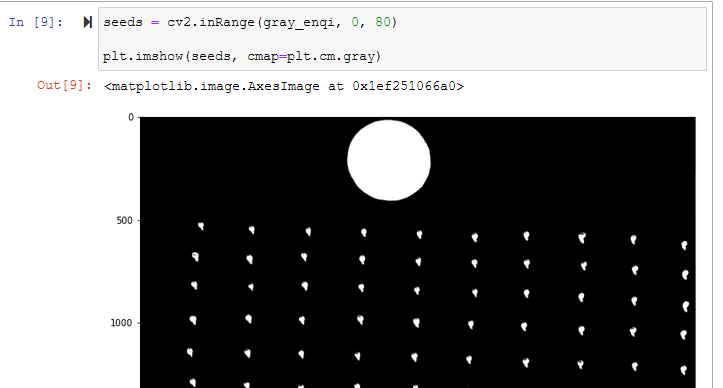
In cell 3 we adjust the thresholds based on the B/W pixel values. For instance, here we find that [0, 50] does not adequately capture the reference object or the seeds:



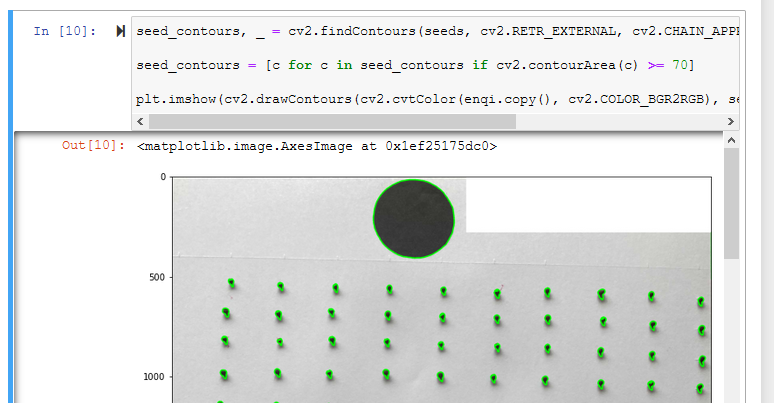
On the other hand, [0, 140] perfectly captures the reference object, but we also pick up shadows on all of the seeds:



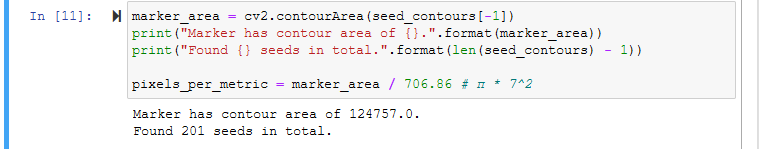
After a play, we find that for this image, [0, 80] provides a reasonable (if imperfect) balance.



Cell 4 draws the contours that the script will use to delineate the pixels used for the seed area computations.



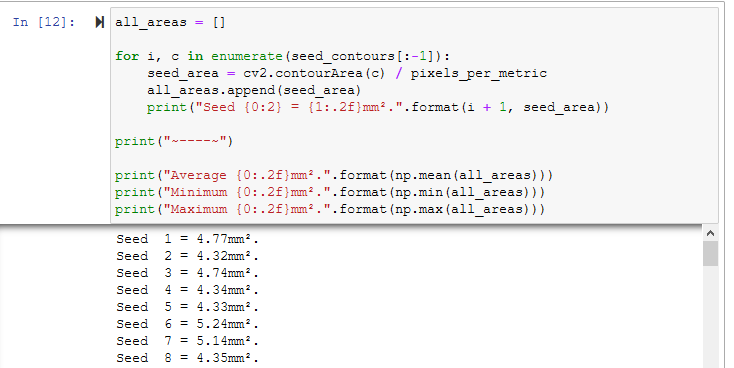
Run cell 5 to confirm that all of the seeds (including the reference object) have been counted.



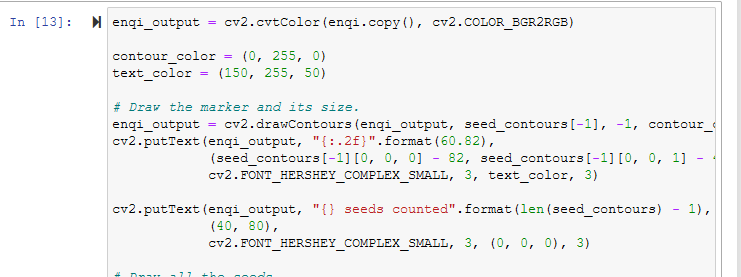
Here we find that the reference marker has 124,757 pixels, and there are 200 seeds (plus 1 for the reference marker object).

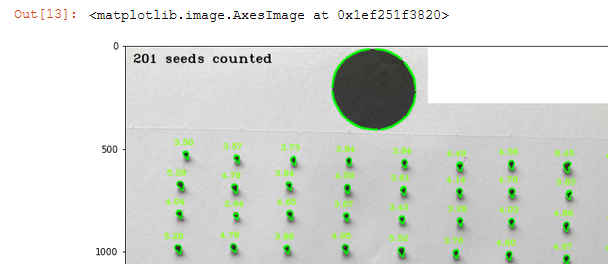
(If this cell reports more seeds than you expect, it is possible that one of the seeds is being double-counted; eg. If the threshold is wrong, and the script identifies two half-seeds instead of one whole seed. In which case you should readjust the threshold accordingly.)

Running cell 6 then reports the measured (real-world) areas for each of the seeds in the test image. You should check that these values are what you expect; if the values seem off, double-check the thresholds and the area that you specified for the reference object.



Similarly, running cell 7 previews the annotated output for this test image.



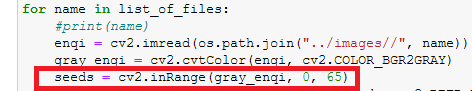


Once you are happy with the test image, you can use the same threshold to bulk-process all of the images and automatically generate the annotated image output and spreadsheet of counted seed areas.

## Count seeds in all images

Cell 8 (the last cell in the Jupyter notebook) processes every image in the /images/ folder, counting the seeds and generating a spreadsheet of the counts and annotated images.

The threshold that you found to work best for the test image should be changed in the line here (e.g. here we use [0, 65] for the threshold across all images):



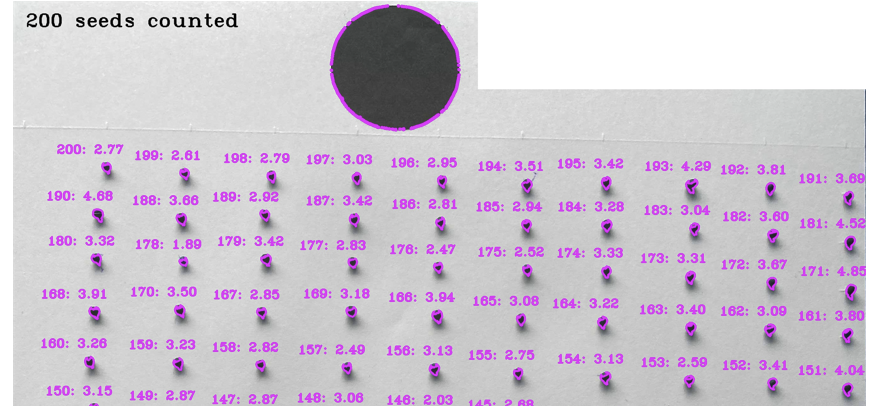
Also double-check that the reference area (in real-world units) is as specified. Note that the reference area is assumed to be the same across all images. (e.g. here the reference circle has 706.86mm2 area).



When ready, run cell 8 to generate all of the output for every image in the /images/ directory all at once.

# Collecting the output

The output appears in the /output/ folder. If the input image is image.jpg, then a corresponding image.jpg i.e. with the same filename) appears in the output folder, annotated with the seed labels and computed seed areas:



And a corresponding spreadsheet image.csv (with the same name as the input image file) is generated, tabulating the seeds and their reported area sizes.

