

Orbis v1.0

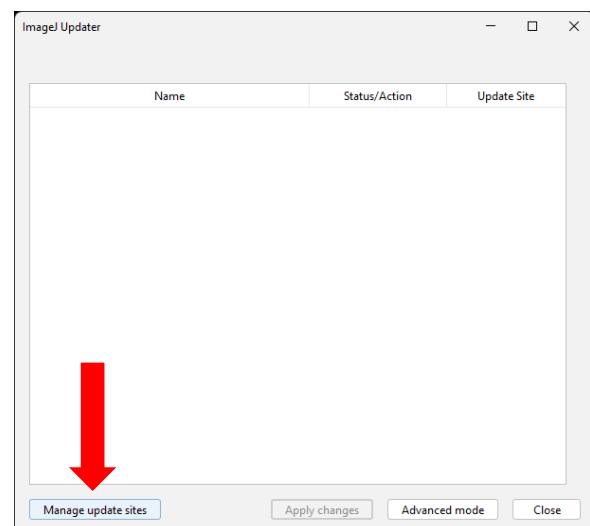
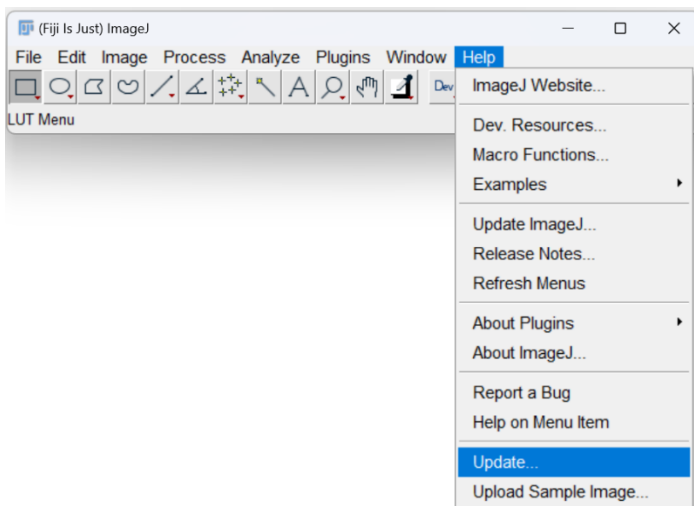
A Fiji/ImageJ macro to automatically measure circle areas

1. Introduction

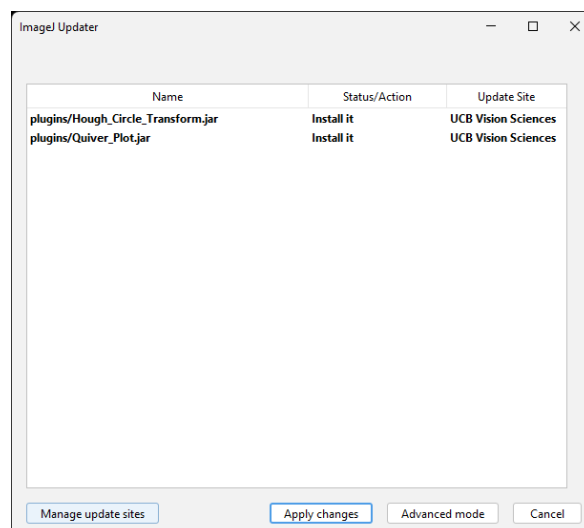
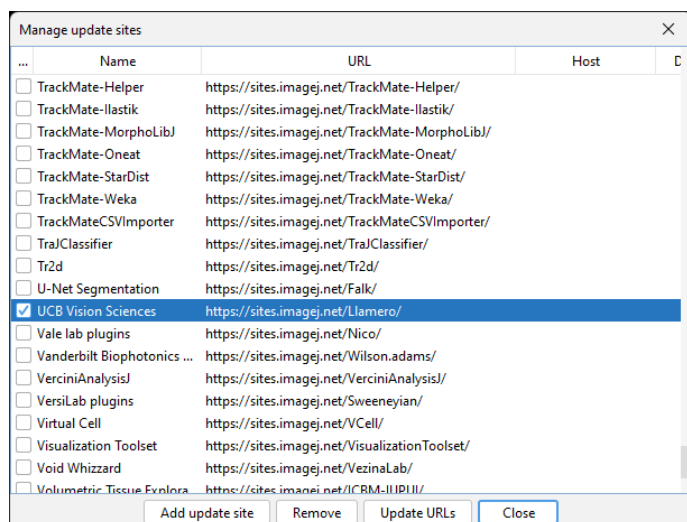
Orbis is a macro script written in ImageJ Macro Language that works to automatize the measurement of circle areas in Fiji/ImageJ. It was made initially to measure fungal colony areas, to determine how different conditions affect growth in agar plates, since determining the colony area is a good method to assess growth rate.

2. Installing Fiji and the necessary plugins

Fiji is an extended version of ImageJ, which is a free program for image treatment and analysis. [Install Fiji](#). Orbis works by first treating the images using ImageJ's build-in features, and then detecting the circles using the Hough Circle Transform plugin from the UCB Vision Sciences package. To install this package, go to "Help" -> "Update" and click on "Manage update sites".



Next, locate "UCB Vision Sciences" and toggle it on. Click "Close". Then click "Apply changes" and the plugins will be installed.

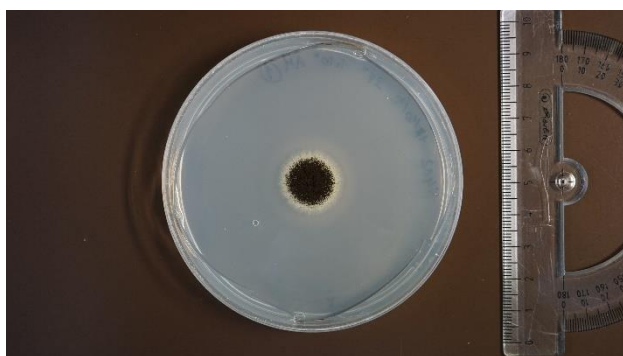


3. Running Orbis

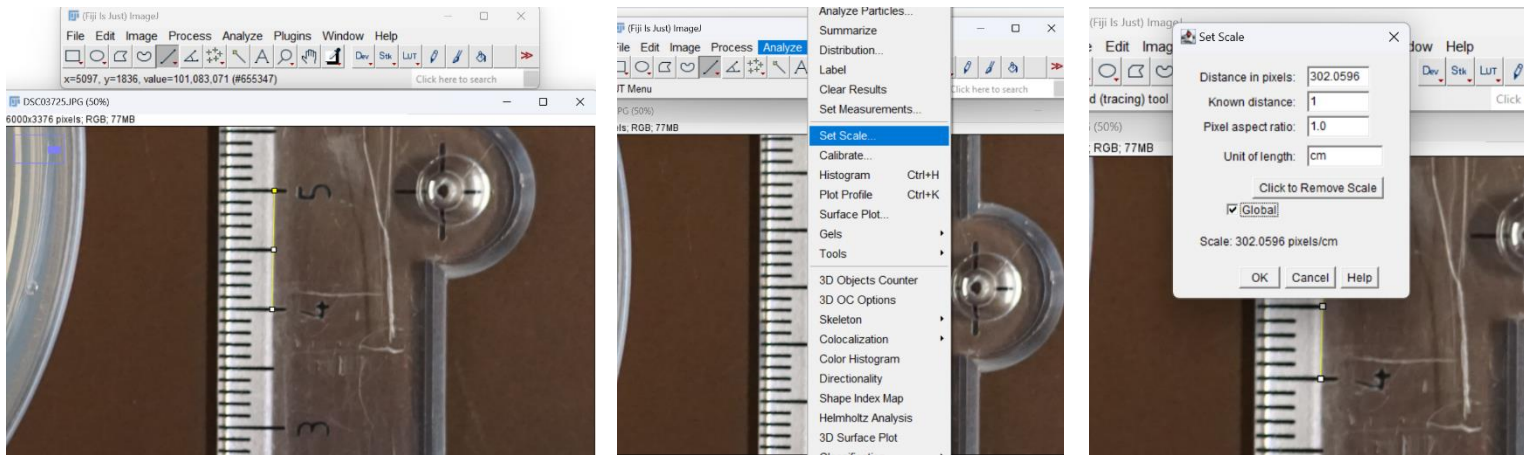
3.1. Prepare your files and set the scale

Have all the images you want to analyze together in one folder. Although it is not necessary, it is recommended to set the correct scale before analyzing the images. If this is not done, Orbis will output the circle areas in pixels, which is ImageJ's default unit.

To set the scale, all photos should be taken at the same distance, and at least one of them should have a ruler, like shown below (left), or a separate photo can be taken of the ruler (right). The ruler should be as straight as possible to facilitate measurement.

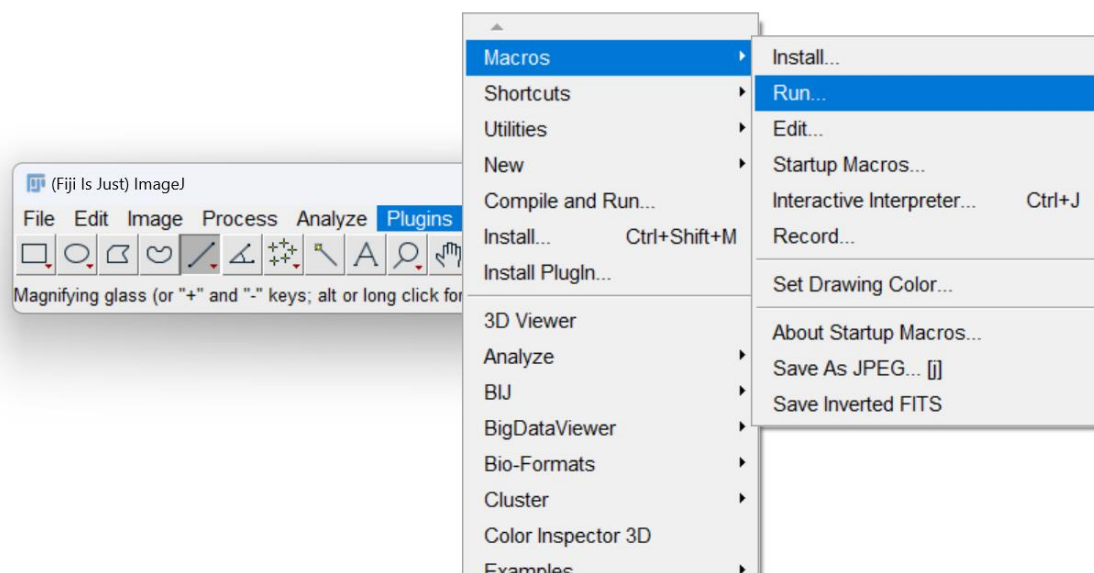


Open an image with the ruler with Fiji. Zoom in to the ruler you placed in the frame of the photo, next to the Petri dish. Draw a straight line spanning exactly 1 unit (e.g. cm) in the ruler. Then go to "Analyze" -> "Set Scale", substitute the "Known distance" by 1, and the "Unit of length" by the ruler's unit, for example "cm". **Click on "Global"**. Click "Ok" and then close the image used to set the scale.



3.2. Run the macro

Go to “Plugins” -> “Macros” -> “Run” and find your Orbis macro.

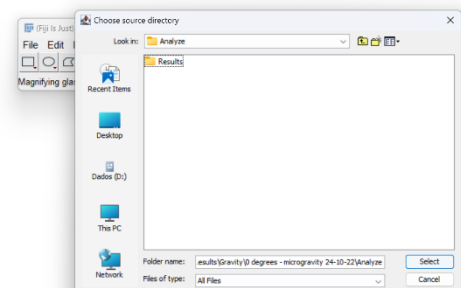


3.2.1. Choosing variables

Orbis is flexible enough to allow for the change of several of its properties to optimize the analysis of different types of images, and depending on the user’s preferences.

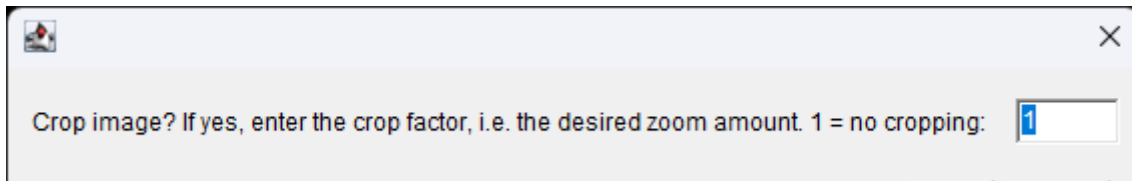
Firstly, you will be asked to choose the source directory, i.e., the directory where your images are located, as seen on the right.

Next, you will be asked to input other variables, some of which related to the image treatment, and others to the properties of the Hough Circle Transform.



Experimenting with different values is important to ensure the best possible measurements.

Image Treatment Variables



- **Cropping/zoom factor** – images are cropped by default into a 1:1 aspect ratio (square) to remove unnecessary pixels that slow down processing time. So, if your image is 1920x1080 pixels, it will automatically trim the sides and become 1080x1080. Additionally, if you want to zoom in, you can increase the factor. If you set it to 2, the image will be additionally cropped to 540x540, zooming in the center. An example of no zoom as well as zoom 2 are shown below:

Original: 6000x3376

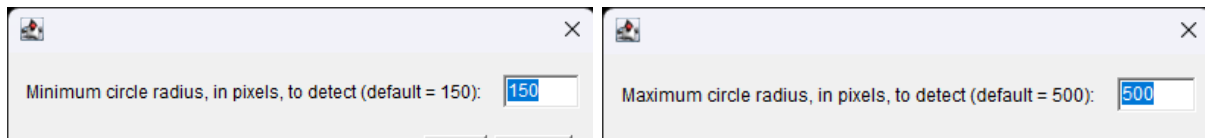


Crop/zoom factor = 1: 3376x3376

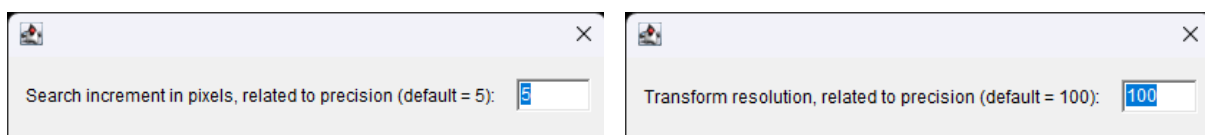
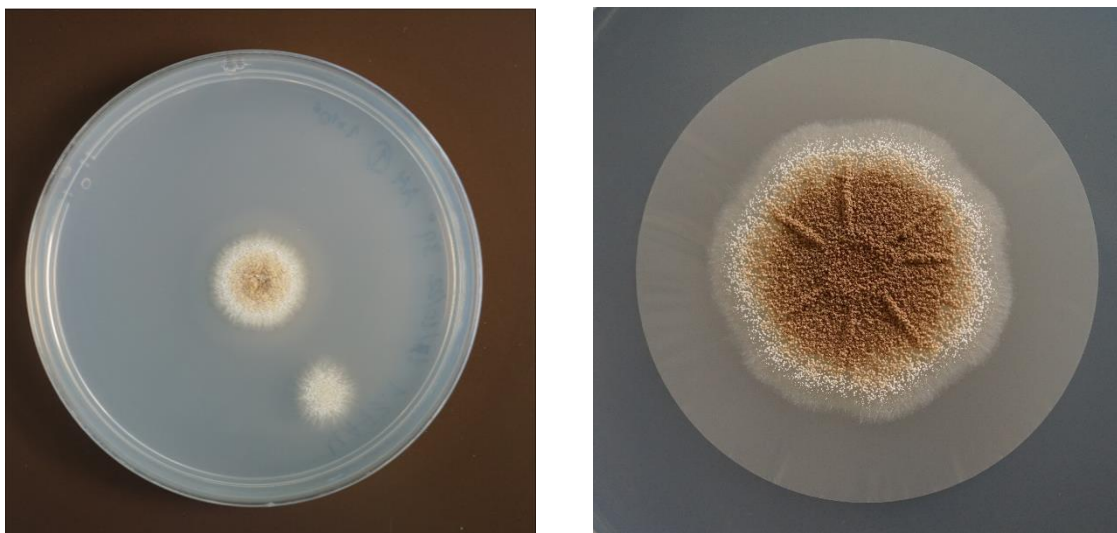


Crop/zoom factor = 2: 1688x1688





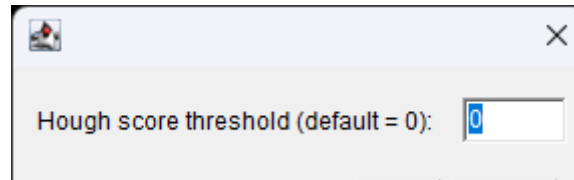
- **Minimum/maximum circle radius** – “The minimum and maximum search radii are the lower and upper cutoff for the radii you expect to find in the image. Ideally, you want to make the Hough search space as specific as possible, so be sure to set these values to specifically the range of radii you expect to find in your data.” (ImageJ wiki). To get these values, draw a straight line from the center of the circle until the edge and check its length. This does not have to be precise, it should just give you a ballpark of the size of the circles you’re looking to measure. In one of the images below (left) the fungal colony radius is indeed between the default values of 150 and 500 pixels. But, in the other image (right), the colony is larger, and the photo was also taken from closer, and therefore higher values are necessary for these limits (e.g. 500 – 1000 pixels).



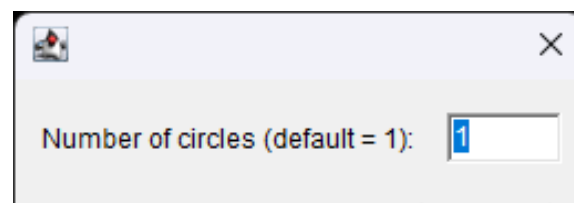
- **Radius search increment and Hough transform resolution** – these variables are required by the Hough Circle Transform, and they’re related to the precision of the detection and how fast it runs. The radius search increment “determines the radius step size to use when creating the 3D Hough space from the minimum radius to the maximum radius. This allows a trade-off between speed and resolution, where larger steps will give a linear increase in speed, but also decrease the precision of the

measured radii.” (ImageJ wiki). The Hough transform resolution “sets the number of steps in each circle transform”. A higher resolution may produce a more precise measurement, but at a much higher performance cost.

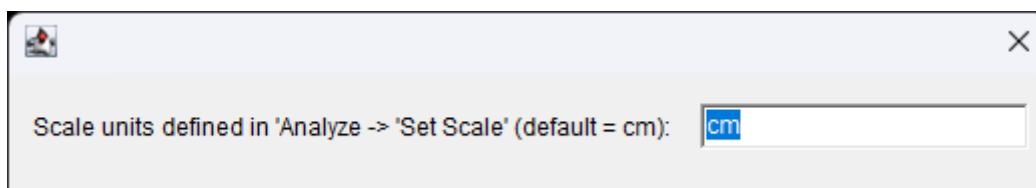
Note: it is recommended that you modify the radius search increment first to try to get adequate results; and only if this doesn’t work change the transform resolution.



- **Hough score threshold** – “This option sets the minimum cutoff for the Hough score (i.e. ratio of votes) that a circle can have to count as a valid object.” (ImageJ wiki). Essentially, it sets the bar for how clear a circle has to be for it to be detected by the plugin. It is highly recommended to leave this option at the default of 0.



- **Number of circles** – the maximum number of circles to detect. It is recommended that you input the expected number of circles.

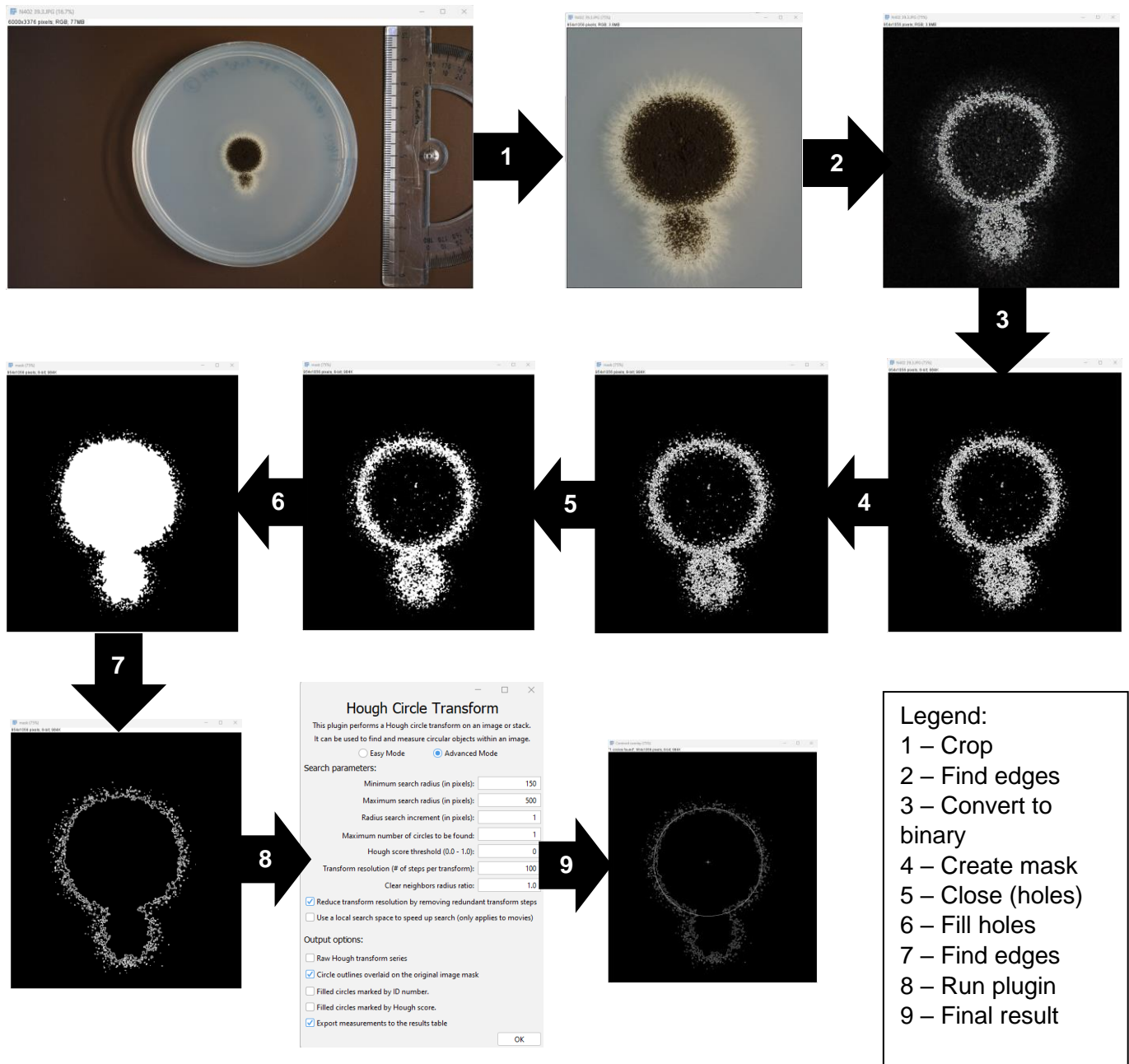


- **Scale units** – the units you defined when you set the scale. If you did not do this, write “pixels”. If the value entered here is not equal to what is set in the scale, the program will fail and print the following error: “Incorrect scale unit. Go to 'Analyze' -> 'Set Scale' to verify the correct unit, or to change it.”

Check the [ImageJ wiki](#) for more information on the Hough Circle Transform and its variables.

3.2.2. Image treatment pipeline

This section details the image treatment steps taken by Orbis to highlight the circles present in the picture, allowing the Hough Circle Transform plugin to detect them accurately.



3.2.3. Output, modifying variables and performance analysis


Before each run, the script creates a folder containing the current date and time, to differentiate between each run. In the following example, we can see two folders created from two different trials during the same day, but a few minutes apart:


<< Analyze > Results					Search Results	
Name	Status	Date modified	Type	Size		
17-11-2022_12-6-18	✓	17-Nov-22 12:27	File folder			
17-11-2022_12-44-27	✓	17-Nov-22 12:45	File folder			

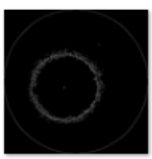
Inside each folder you can find the processed images and the detected circles, so you can evaluate whether the measurements were done correctly. You'll also find a .csv file containing the measured circle radii values for each image and the corresponding area of the circle.

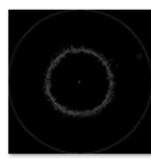
<< Analyze > Results > 17-11-2022_12-44-27

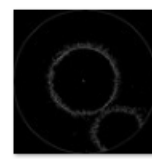
Search 17-11-2022_12-44-27

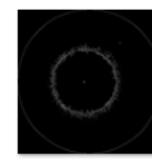

✓ colony_areas.csv

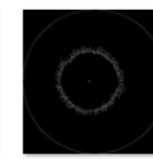

✓ log.txt



✓ MA93.1 39.1.png

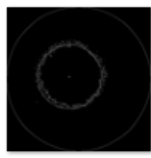

✓ MA93.1 39.2.png

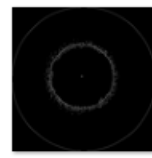

✓ MA93.1 39.3.png

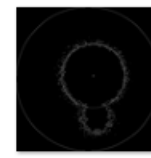

✓ MA93.1 C.1.png

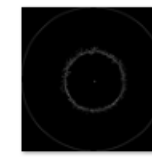

✓ MA93.1 C.2.png

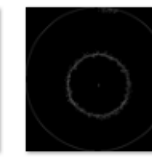

✓ MA93.1 C.3.png

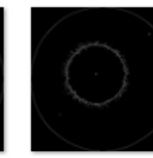

✓ N402 39.1.png


✓ N402 39.2.png


✓ N402 39.3.png


✓ N402 C.1.png


✓ N402 C.2.png


✓ N402 C.3.png

A log file is also created in the results directory, containing information about the variables selected during this run. Additionally, the last line contains data on the performance of the run during this run, in the form of the average time taken per image.


```
log.txt - Notepad
File Edit View

Date - 17-11-2022_12-44-27
Crop/zoom = 3
Min. radius = 150
maxRadius = 500
Increment = 10
Resolution = 100
Threshold = 0
Circle number = 1
Scale unit = cm
Performance = 3 seconds per image (average)
```

Note that the performance depends not only on the variables chosen, but also on the analyzed images' resolution. Orbis' processing speed bottleneck is always the Hough Circle Transform calculations, which may take as little as a few seconds per image and as much as a few minutes per image.

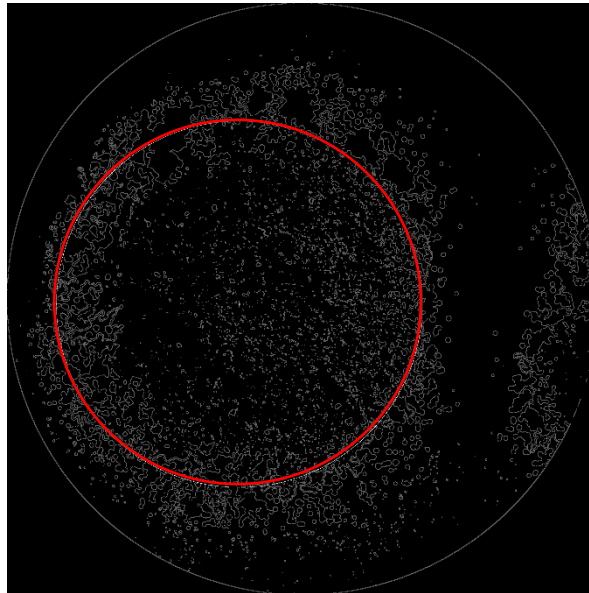
Users should keep in mind that experimenting with the program is important to yield the best results possible. Using a higher precision will not always get you better results, depending on the circumstances. Let's analyze an example case.

Example:

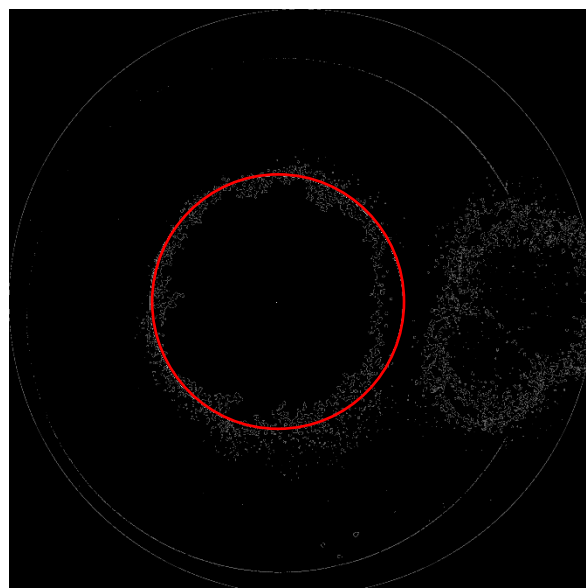
As already explained, the crop factor variable allows you to remove irrelevant pixels of the picture, thus increasing processing speed, and focusing more on the circle. However, in some cases, zooming in too much on the circle may worsen the detection. Therefore, more zoom/crop isn't always better. In the following example, by keeping the crop factor at the default of 1, that is, not zooming into the image, Orbis gets a much more precise detection of the circle, due to how this impacts the image treatment results described in 3.2.2. Here's the original image used in this example:



If at first we try to run Orbis with a crop factor of 1.7, the colony at the center is highlighted clearly, and the second colony (which we do not want to measure) is essentially excluded. This, in principle, is good practice, but, since the image was taken from close, the colony now takes up most of the cropped photo, and the image treatment does not totally highlight the edges sufficiently. Using an increment value of 5, we get the following measurement (traced in red for clarity), which took about 30 seconds to complete:



Clearly, the detected circle is smaller than the real size of the colony. Now, if we keep the crop factor at 1 (no zoom), even if we decrease the precision slightly by increasing the increment to 10, we still get a much more accurate result:



Additionally, the processing time for this second implementation was about 22 seconds, which is lower than the previous run, even with a larger image with more pixels to analyze. This was possible due to the increase in the increment value.