Run2dmorph (Ver.2017-06) Manual

Allison Hsiang

2017-09-18

I Introduction

Run2dmorph is the 2D-data extraction module of the *AutoMorph* software package developed by Pincelli Hull and team [1]. *AutoMorph* is available for download on GitHub. The *run2dmorph* module is described in detail in Hsiang *et al.* [2].

The basic pipeline of *run2dmorph* is as follows: the user supplies the program with a directory of RGB images containing individual objects (one object per image; all potentially confounding information, such as labels, should be excluded). *Run2dmorph* expects light objects on a dark background. If the user is using *run2dmorph*'s companion *AutoMorph* modules *segment* and *focus*, the 'focused_unlabeled' folder outputted by *focus* should be used as the input directory of images for *run2dmorph*. The images are run through a series of filters (Fig. 1) that converts the original color image into black and white, with black indicating background and white indicating the object. The 2D outline is then extracted from this black and white image, along with measures of enclosed area, eccentricity, major and minor axis length, perimeter length, rugosity, and aspect ratio.

II Technical Specification

Run2dmorph, like all *AutoMorph* modules, is run using the command line. On Mac OSX, you must install the GNU Coreutils command line tools (more information here) in order to use the essential UNIX commands (*e.g.*, 'ls' and 'cat') and run the *run2dmorph* binary executable. If you are unfamiliar with the command line, we recommend searching for introductory tutorials online and familiarizing yourself before diving in (here are some suggestions). A good golden rule when dealing with the command line as a beginner is: never input a command if you don't know exactly what it will do!

Run2dmorph is run using a plain-text input file, referred to here as a 'control file'. Explanations for the user-controlled variables in the control file, and how to set them, are presented in section V. By default, *run2dmorph* will output greyscale versions of the original images with the extracted outline overlaid for checking outline extraction fidelity, a text file containing the user input parameters, and a CSV of all 2D shape measurements. Optional outputs include x,y-coordinates for the extracted outlines, an illustration of the minimum bounding box used to calculate aspect ratio, and intermediate images of the output of each image processing filter step.

If *run2dmorph* fails to extract an object, it is usually due to the input parameters being poorly suited to the object image. In this case, *run2dmorph* will skip the object and move on to the next. If any objects are skipped in this way, *run2dmorph* will also output a TXT file listing all skipped objects, and generate a folder nested within the output directory named 'no_outline_extracted' that contains a copy of all skipped objects to facilitate easy re-processing.

III Installation

III.1 Prerequisites

Run2dmorph is written in Python and currently is only compatible with Python 2.7. Compatibility with Python 3.x is under development for future release. *Run2dmorph* requires the following modules:

- numpy
- scipy
- pandas
- opencv
- scikit-image (v0.9.0+)
- pillow
- matplotlib

We recommend that users install Anaconda, as this is the easiest way to install these Python modules. Once Anaconda is installed, users need only manually install the opencv module:

```
$ conda install opencv -c conda-forge
```

Note that the -c conda-forge flag is necessary for opencv to install correctly. If you run into problems installing/importing opencv, consult the Troubleshooting section below.

If you choose to install the required Python modules manually, note that scikit-image is named skimage in Python repositories.

III.2 Setup

You should first download *AutoMorph* from the GitHub repository by clicking the green "Clone or download" button on the right side of the screen and selecting the "Download ZIP" option. Once you have downloaded the *AutoMorph* software package, you will find the *run2dmorph* executable in the 'run2dmorph' folder. We recommend adding the *run2dmorph* executable to your path, so that *run2dmorph* can be called from anywhere in your system. To do this on Mac OSX, open the Terminal program (located at /Applications/Utilities/Terminal), and type the following command at the prompt:

```
$ ln -s AMPATH/AutoMorph/run2dmorph/run2dmorph /usr/local/bin
```

where AMPATH is the location of your *AutoMorph* installation. For example, if *AutoMorph* is located in /Applications, the full command would be:

```
$ ln -s /Applications/AutoMorph/run2dmorph/run2dmorph /usr/local/bin
```

If you are using Mac OS X, you may encounter a 'Permission denied' error when attempting to create the symbolic link. If this happens, you should use the 'sudo' command like so:

```
$ sudo ln -s /Applications/AutoMorph/segment/segment /usr/local/bin
```

You will be prompted to enter your password; once you have done so, the symbolic link will be created. Note that you should not use 'sudo' unless you know exactly what you are doing!

You can now use *run2dmorph* from any location on your computer. Note that when creating symbolic links you should always use absolute paths and not relative ones.

IV Quick Run

Once *run2dmorph* is installed, it can be run via the command line using the following command (assuming the *run2dmorph* executable in your path):

```
$ run2dmorph <path to control file>
```

You can use either absolute or relative paths when calling *run2dmorph*. For instance, if my control file is named run2dmorph_control.txt and located in my current working directory, the full command would be:

```
$ run2dmorph run2dmorph_control.txt
```

While running, *run2dmorph* will output status messages on its progress.

V Control File and Parameters

The control file serves as the means by which the user supplies the necessary parameters for *run2dmorph* to operate. A clean version of the control file can be found in AutoMorph/run2dmorph. Some parameters require user input while others can left blank to allow *run2dmorph* to use default values. It is recommended that the user read through this section and understand what all the parameters do, and how to set them, before using *run2dmorph*.

The control file should be edited using a plain text editor (*e.g.*, BBEdit for Mac OSX) to avoid interpretation issues and must use Unix (LF) encoding.

For parameters that are not required, the default value can be used by leaving the parameter blank, like so:

```
out_directory =
```

The default values of *run2dmorph* tend to work well with the images of foraminifera produced by the Hull lab (see example files in AutoMorph/example_datasets); users are encouraged to use these as a baseline against which to test their own images and parameter values.

A listing of the parameters for *run2dmorph* follows, with required parameters marked with an *:

*** in_directory**: The full path to the input directory containing the images to be processed by *run2dmorph*. This cannot be a relative path.

*** input_ext**: The file extension of the images to be processed by *run2dmorph*. For instance, `.tif` or `tif`.

*** sampleID**: A name designating the identity of the current run, which will be appended before all files that *run2dmorph* outputs.

out_directory: The full path of the folder to which *run2dmorph*'s output will be saved. If the folder does not exist, *run2dmorph* will create it. If this parameter is left blank, *run2dmorph* will generate a folder named 'morph2d' inside the directory specified under 'in_directory'. This cannot be a relative path.

pixel_size_x: A number specifying a conversion factor for pixel size in the width of the input image. *Run2dmorph* will scale the width of all images by this factor before extracting outlines and shape measures. The default value is 1 (*i.e.,*, no conversion).

pixel_size_y: As above, but for the height of the image. The default value is 1.

get_coordinates: A boolean (*i.e.*, True or False) specifying whether *run2dmorph* will output x,y-coordinates for the outlines extracted from

each image. The default value is True.

draw_aspect_ratio: A boolean specifying whether *run2dmorph* will output JPGs and PDFs illustrating the minimum bounding box around each extracted outline, which is used to calculate the aspect ratio of the object. The default value is True.

save_intermediates: A boolean that determines whether *run2dmorph* will output intermediate images of the output of every image processing filtering step (*i.e.*, the images shown in Fig. 1). The default value is False; we suggest turning this option on when users are conducting initial explorations of the most appropriate filter parameter values and when troubleshooting poor outline extraction.

disk_size_opening: An integer specifying the diameter of the disk used to to perform morphological opening (erosion followed by dilation; first step in Fig. 1). In general, turning this value up will remove increasing larger patches of background noise, but will also remove morphological detail from the object itself. The default value is 10.

contrast_adjustment: An integer specifying the degree of contrast enhancement to perform in the Contrast filter (step 3 in Fig. 1). The higher the value, the higher the contrast. The default value is 3.

threshold_adjustment: A number that specifies an adjustment value that will be added to the threshold value that is automatically determined *run2dmorph* during conversion of the image from greyscale to black and white. A higher threshold adjustment value corresponds to higher tolerance during the conversion (*i.e.*, a broader, lighter range of grey will be considered white in the final black and white image). Note that while this adjustment value can be negative, the *total* threshold adjustment value cannot be negative. *Run2dmorph* will thus ignore this adjustment value and use the automatically determined threshold value if the sum of the two is negative. The default value is 0.

disk_size_smoothing: An integer specifying the diameter of the disk used to smooth the edge of the extracted outline (via binary opening). The higher the value, the more smoothing is applied (*i.e.*, the more detail is removed from the outline). The default value is 20.

downsample: A boolean specifying whether the x,y-coordinates extracted by *run2dmorph* should be downsampled from the default number of points (equal to the number of pixels in the outline). The default value is True. This parameter is only applicable when the 'get_coordinates' parameter is True.

num_points: An integer specifying the number of x,y-coordinates to be outputted if the 'downsample' parameter is set to TRUE. For example, setting this value to 100 will result in 100 outline x,y-coordinates outputted for every object processed. The default value is 100.

VI Hands-On Example Run

A set of example images for testing *run2dmorph* can be dowloaded on Zenodo here. Users can also use the mini dataset provided on GitHub, which is considering smaller in size. The following tutorial assumes the user is using the larger dataset provided on Zenodo.

A control file that can be used with these images is located at AutoMorph/run2dmorph/examples/run2dmorph_control_4sq_example.txt. The user need only fill in the **in_directory** parameter to run the example. Step-by-step instructions – which assume that the user has added *run2dmorph* to their system's path, that *AutoMorph* is installed in /Applications, and that the example images are located in /Downloads – follow:

1. Open run2dmorph_control_4sq_example.txt in a plain-text editor and change the **in_directory** to the appropriate path, and then save the file:

   ```
   in_directory = /Downloads/run2dmorph_4sq_example_EDFs
   ```

2. At the command line prompt, navigate to the folder containing the control file:

   ```
   $ cd /Applications/AutoMorph/run2dmorph/examples
   ```

3. Start *run2dmorph* by entering the following command:

   ```
   $ run2dmorph run2dmorph_control_4sq_example.txt
   ```

Run2dmorph will proceed to extract outlines from the example objects; when extraction is complete, the command line prompt will reappear. The output files will be located in a folder named 'morph2d' inside the folder specified under **in_directory**.

VII Troubleshooting

Based on in-house usage of *run2dmorph*, the most common issues that arise concern incorrect specification of parameters in the control file. In general, if *run2dmorph* encounters an error before outline extraction begins, something is likely wrong with the control file formatting or encoding. In this case, the user is advised to:

- Check that the encoding of the control file is set to Unix (LF) and that the control file is in plain text format.
- Check that the correct input directory containing the images is specified.
- Check that the parameters are formatted correctly.

If *run2dmorph* runs but extracts poor outlines, the problem likely lies in the image filter parameters. In this case, the user is recommended to make a test folder containing 5-10 images from their full sample, turn on the **save_intermediates** parameter, and adjust parameter values until acceptable outlines are extracted for the test objects before running *run2dmorph* on the entire sample.

VII.1 Issues with Installing Opencv

You may run into issues installing opencv with Anaconda, particularly if your system has previous installations of Python and/or other package managers such as Homebrew. The following are issues and solutions that we have come across in-house when installing opencv on Mac OS X.

VII.1.1 Homebrew

If you have previously installed Homebrew, you can install opencv by tapping the homebrew/science repository, like so:

```
$ brew tap homebrew/science
$ brew install opencv
```

VII.1.2 ImportError

If you install opencv with Anaconda, you may encounter an ImportError when trying to import the cv2 library. This may be the result of certain libraries being missing, and you will see an error message similar to the following:

```
Traceback (most recent call last):
  File "/usr/local/bin/segment", line 4, in <module>
    import images
  File "/Automorph/AutoMorph-master/segment/images.py", line 9, in <module>
    import cv2
ImportError: [...] : Library not loaded: @rpath/libopenblasp-r0.2.19.dylib
    Referenced from: somepath/libopencv_hdf.3.2.0.dylib
    Reason: image not found
```

This issue occurs because the dlib library is missing. You can install this library via the following command:

```
$ conda install dlib -c conda-forge
```

You may then need to update other packages (*e.g.*, numpy). Update all your packages like so:

```
$ conda update --all -c conda-forge
```

Alternatively, you may encounter the following error message:

```
ImportError: libopenblas.so.0: cannot open shared object file: No such file
or directory
```

This error can be solved by installing the Openblas library:

```
$ conda install openblas
```

You should now be able to use opencv without issue.

References

[1] *AutoMorph* (https://github.com/HullLab/AutoMorph)

[2] Hsiang AY, Nelson K, Elder LE, Sibert EC, Kahanamoku SS, Burke JE, Kelly A, Liu Y, Hull PM. AutoMorph: Accelerating community morphometrics with 2D and 3D image processing and shape extraction. **Methods in Ecology and Evolution**. *In revision.*

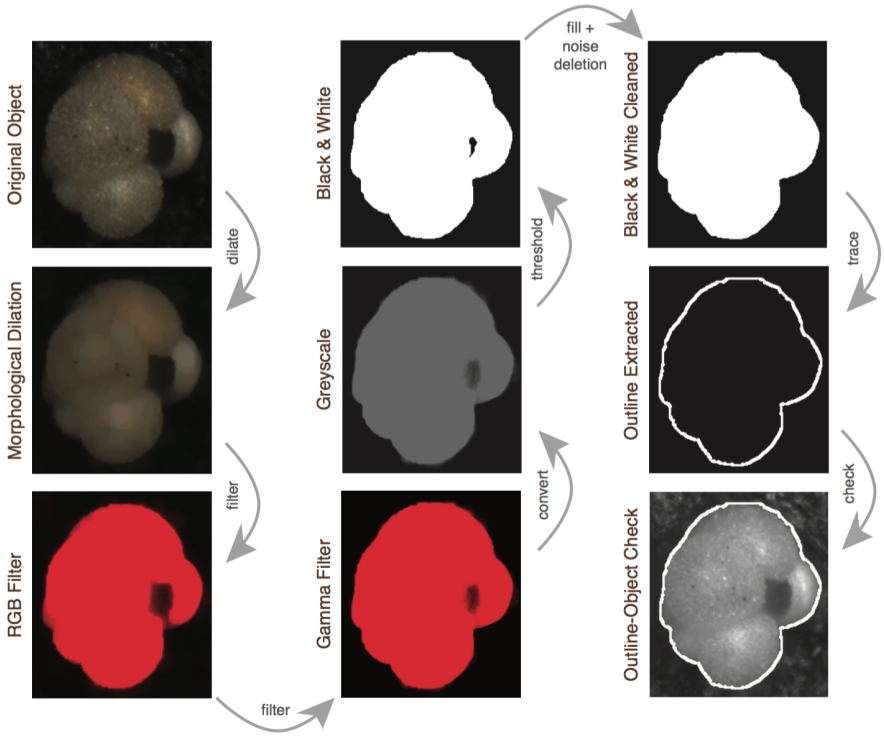

Figure 1: Visual pipeline of the image processing filters used in *run2dmorph*.