Run3dmorph (Ver.2017-06) Manual

Allison Hsiang

2016-09-06

I Introduction

Run3dmorph is the 3D-data extraction module of the *AutoMorph* software package developed by Pincelli Hull and team [1]. *AutoMorph* is available for download on GitHub. The *run3dmorph* module is described briefly in Hsiang *et al.* [2] and in detail in Hsiang *et al.* [3].

Run3dmorph is designed to work directly with the output of *segment* and *focus*, the image segmentation and batch focus stacking modules of *AutoMorph*. The basic pipeline of *run3dmorph* (Fig. 1) is as follows: using z-stacks of individual objects extracted by *segment*, *run3dmorph* writes a series of ImageJ [4] macros that call the Stack Focuser plugin [5] and generates height maps for each object (Fig. 1a). The height maps code relative height using greyscale pixel values ranging from 0 to 255. To remove background noise, the 2D outline of the object is extracted using the EDFs generated by *focus*; the background is then deleted via element-wise multiplication (Fig. 1b). The cleaned-up height map (Fig. 1c) is then converted to a 3D mesh: by correlating the height map greyscale values with the height of each focus plane in the original z-stack using the z-step distance between plane, the real-world height of each pixel (as limited by z-step distance resolution) can be determined (Fig. 1d). Residual noise in the mesh is removed using a custom sliding neighborhood filter with a user-defined $n \times n$ kernel (where n is a positive odd integer).

The extracted mesh is outputted as OBJ and OFF files, and the surface area and volume are calculated. The surface area and volume of the hidden backs of the objects are also calculated using a series of three idealized shapes (Fig. 2). For more details on these steps and algorithms, see Hsiang *et al.* [3].

II Technical Specification

Run3dmorph, like all *AutoMorph* modules, is run using the command line. On Mac OSX, you must install the GNU Coreutils command line tools (more in-

formation here) in order to use the essential UNIX commands (*e.g.*, 'ls' and cat') and run the *run3dmorph* 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!

Run3dmorph 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, *run3dmorph* will output the x,y,z-coordinates of extracted meshes as CSV, OBJ, and OFF files. It will also output the calculated volume and surface area of each extracted mesh, along with total volume and surface area estimates using each of the three idealized back shapes (Fig. 2).

Optionally, the user can have *run3dmorph* output 3D PDFs, which contain: The metadata label for each object; the extracted length and height of each object; the operative grid size and base unit used for mesh extraction; the 3D mesh embedded as a rotatable, zoomable object (viewable using Adobe Acrobat/Reader; Adobe Flash is required); and the original 2D EDF image of each object. The 3D PDF allows for easy checking of the fidelity of the 3D mesh extraction, but requires the non-trivial installation of many dependencies (see section III.1). If the user has difficulty getting the 3D PDF to build, we recommend checking the extracted 3D meshes in a program such as MeshLab.

III Installation

III.1 Prerequisites

III.1.1 Dependencies

Run3dmorph has the following dependencies:

- Python 2.7
- FIJI
- ImageMagick
- LaTeX (optional)

III.1.2 Python

Run3dmorph is written in Python and currently is only compatible with Python 2.7. Compatibility with Python 3.x is under development for future release. *Run3dmorph* requires the following Python modules:

- numpy
- scipy
- pandas
- scikit-image
- scikit-learn
- pillow
- matplotlib
- opencv
- shapely

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 and shapely modules:

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

```
$ conda install shapely
```

Note that the -c conda-forge flag is necessary for opencv to install correctly.

III.1.3 FIJI

Run3dmorph also requires an installation FIJI [4] that is callable from the command line via the one of the following commands:

- Mac: 'ImageJ-macosx'
- Linux: 'ImageJ-linux32' or 'ImageJ-linux64'
- PC: Not yet supported (under development)

On Mac OSX, this can be achieved by creating a symbolic link to the FIJI executable in the /usr/bin/local folder. Assuming the user has installed FIJI, this can be done by opening a Terminal window and entering:

```
$ ln -s /IPATH/Fiji.app/Contents/MacOS/ImageJ-macosx /usr/local/bin
```

where IPATH is the installation location of FIJI, for example /Applications. You will now be able to use FIJI from anywhere in your system, so *run3dmorph* will be able to call FIJI as necessary. Note that you should use absolute paths when creating symbolic links, not relative ones.

The Stack Focuser plugin [5] must also be installed. The class file can be downloaded here. Once downloaded, the plugin can be installed by opening FIJI and selecting the menu option `Plugins -> Install PlugIn...` and then selecting the class file.

If the user wishes to use 'macro' mode (which is necessary when the input images are taken with a camera rather than a microscope), then they must also install the FIJI plugins StackReg [6] and TurboReg [6]. Once downloaded, they can be installed following the same procedure as for the Stack Focuser plugin (but note that only the .jar file should be installed, and *not* the .java file).

III.1.4 ImageMagick

ImageMagick can be downloaded and installed using the installers and directions available here. Alternatively, if you have MacPorts or Homebrew on your system, ImageMagick can be installed using the following commands:

```
$ sudo port install ImageMagick
```

or

```
$ brew install ImageMagick
```

On newer versions of Mac OS X, you may need to return ownership of /usr/local bak to your user account with the following command:

```
$ sudo chown -R $(whoami) /usr/local
```

III.1.5 LaTeX

If the user is intending to use the 3D PDF functionality of *run3dmorph*, an installation of LaTeX is required. LaTeX must be callable from the command line via the command 'pdflatex'. This should be the default behavior when installing LaTeX with the default settings. The required STY file for the required LaTeX package, media9, is included in /AutoMorph/run3dmorph/lib/media9.

III.2 Setup

Once you have downloaded the *AutoMorph* software package, you will find the *run3dmorph* executable in the 'run3dmorph' folder. Users will also find a clean control file, which is necessary for setting *run3dmorph*'s various parameters.

We recommend adding the *run3dmorph* executable to your path, so that *run3dmorph* 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/run3dmorph/run3dmorph /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/run3dmorph/run3dmorph /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 *run3dmorph* from any location on your computer.

IV Quick Run

Once *run3dmorph* is installed, it can be run via the command line using the following command (assuming the *run3dmorph* executable is in your path):

```
$ run3dmorph <path to control file>
```

Run3dmorph can also take an optional argument, `--reset`, which will delete the output 'morph3d' folder if it exists in the currently specified output directory and rerun the entire pipeline. This is useful if a previous run was aborted or poorly executed due to inappropriate parameter values. This is activated via the following command:

```
$ run3dmorph <path to control file> --reset
```

While running, *run3dmorph* will output status messages on its progress.

Depending on the number of objects being processed, *run3dmorph* can take a long time to run (for reference, our in-house microfossil images take 3-5 minutes to process per object). If *run3dmorph* is stopped before it is compete, it will restart from where it left off as long as the morph3d output folder has been untouched. To restart *run3dmorph*, simply run *run3dmorph* with the original control file again.

V Control File and Parameters

The control file serves as the means by which the user supplies the necessary parameters for *run3dmorph* to operate. A clean version of the control file can be found in AutoMorph/run3dmorph. Some parameters require user input while others can left blank to allow *run3dmorph* to use default values. It is strongly recommended that the user read through this section and understand what all the parameters do, and how to set them, before using *run3dmorph*.

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 *run3dmorph* 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 *run3dmorph* follows, with required parameters marked with an *:

V.1 Global Parameters

*** in_directory**: The full path to the input directory, which is equivalent to the path to the 'final' directory (containing the 'focused', 'focused_unlabeled', and 'stripped' directories) that *focus* outputs. This cannot be a relative path.

out_directory: The full path of the folder to which *run3dmorph*'s output will be saved. If the folder does not exist, *run3dmorph* will create it. If this parameter is left blank, *run3dmorph* will generate a folder named 'morph3d' inside the directory specified under 'in_directory'. This cannot be a relative path.

*** sampleID**: A name designating the identity of the current run, which will be appended before all files that *run3dmorph* outputs.

*** stack_image_ext**: The file extension of the z-stack images to be processed by *run3dmorph*. This should be formatted with the dot - so `.tif` is valid, but `tif` is not.

macro: A boolean (*i.e.*, True or False) specifying whether macro-object mode should be activated. This should be set to True when the input images were captured using a camera (*e.g.*, a DSLR camera) and False when the input images were captured using a microscope. When set to True, FIJI will run the StackReg plugin using the 'scaled rotation' method to align z-slices before generating the height map, and then invert the greyscale values in the height map (a necessary step based on in-house experience dealing with camera-generated images).

latex: A boolean specifying whether *run3dmorph* should run LaTeX to generate 3D PDFs. Default value is True.

V.2 Z-Stack Parameters

*** unit**: The base unit used for pixel conversion, *e.g.*, microns.

pixel_size_x: A number specifying a conversion factor for pixel size in the width of the input image. *Run3dmorph* will scale the width of all images by this factor before extracting meshes. 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.

*** slices**: The number of slices in each individual image z-stack.

*** zstep**: The distance between each z-slice in the input z-stacks, in base units (as specified under **unit**).

V.3 Mesh Extraction Variables

kernel_heightmap: An positive odd integer n that sets a $n \times n$ kernel that is used by Stack Focuser to generate a heightmap using a sliding neighborhood filter. Larger kernel sizes correspond to larger patches over which pixel values are averaged, *i.e.*, lower detail. Images with large amounts of noise should use larger kernel values. The default value is 11.

kernel_outlierfilter: A positive odd integer designating the neighborhood size (in pixels) for the outlier filter that cleans noise the 3D mesh (see Fig. 1d). Use a larger kernel size for larger patches of noise (though note that processing time increases significantly as kernel size increases). Default value is 45.

grid_size: A positive integer designating the x,y-grid size (in pixels) to be used if the user wishes to downsample the 3D mesh coordinates. The default value is 1, which corresponds to a coordinate being written for every pixel in the object.

V.4 Run2dmorph Parameters

These parameters are used when using *run2dmorph* to extract 2D outlines to delete background noise (see Fig. 1b). Users are advised to determine the optimal values for these parameters using *run2dmorph* before running *run3dmorph*.

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.

VI Hands-On Example Run

An example 'final' output directory created by *focus* that is ready to be processed using *run3dmorph* can be downloaded on Zenodo here. A mini dataset containing only two z-stacks is also available in AutoMorph/example_datasets. For the following tutorial, we assume the user is using the larger dataset from Zenodo.

A control file that can be used with these images is located at AutoMorph/run3dmorph/examples/run3dmorph_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 *run3dmorph* to their system's path, that *AutoMorph* is installed in /Applications, and that the 'final' directory is located in /Downloads – follow:

1. Open run3dmorph_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/final
   ```

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

   ```
   $ cd /Applications/AutoMorph/run3dmorph/examples
   ```

3. Start *run3dmorph* by entering the following command:

```
$ run3dmorph run3dmorph_control_4sq_example.txt
```

Run3dmorph will proceed, beginning with height map generation. Status update messages will be printed to the screen as *run3dmorph* runs. When extraction is complete, the command line prompt will reappear. The output files will be located in a folder named 'morph3d' nested inside the directory specified under **in_directory**.

VII Troubleshooting

At the moment, *run3dmorph* does not support verbose error handling in many instances (for instance, if heightmap generation in FIJI fails, mesh extraction will fail without indicating that the issue is a missing heightmap). This is under development and will be improved for subsequent releases. In the meantime, users should check the output folders to determine if files were created properly.

Issues may arise due to invalid control file specifications and/or inappropriate image processing parameter values (as in *run2dmorph*). In general, if *run3dmorph* 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 paths specified under **Global Parameters** are correct.
- Check that the parameters are formatted correctly.

If *run3dmorph* runs but seems to be removing background noise poorly, the problem likely lies in the *run2dmorph* image processing parameters; in this case, the user is recommended to first optimize these parameters by running *run2dmorph* separately on 5-10 test images before returning to run *run3dmorph* on the full sample.

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.*

[3] Hsiang AY, Elder LE, Hull PM. (2016) Toward a morphological metric of assemblage dynamics in the fossil record: a test case using planktonic foraminifera. *Philosophical Transactions of the Royal Society B.* **371**:20150227 (doi:10.1098/rstb.2015.0227)

[4] Schindelin J, Arganda-Carreras I, Frise E *et al.* (2012) Fiji: an open-source platform for biological-image analysis. *Nature Methods.* **9**(7):676-682.

[5] Umorin, M. (2002) Stack Focuser (https://imagej.nih.gov/ij/plugins/stack-focuser.html)

[6] Thvenaz, P., Ruttimann U.E., Unser M. (1998) A pyramid approach to subpixel registration based on intensity. IEEE Transactions on Image Processing, vol. 7, no. 1, pp. 27-41.

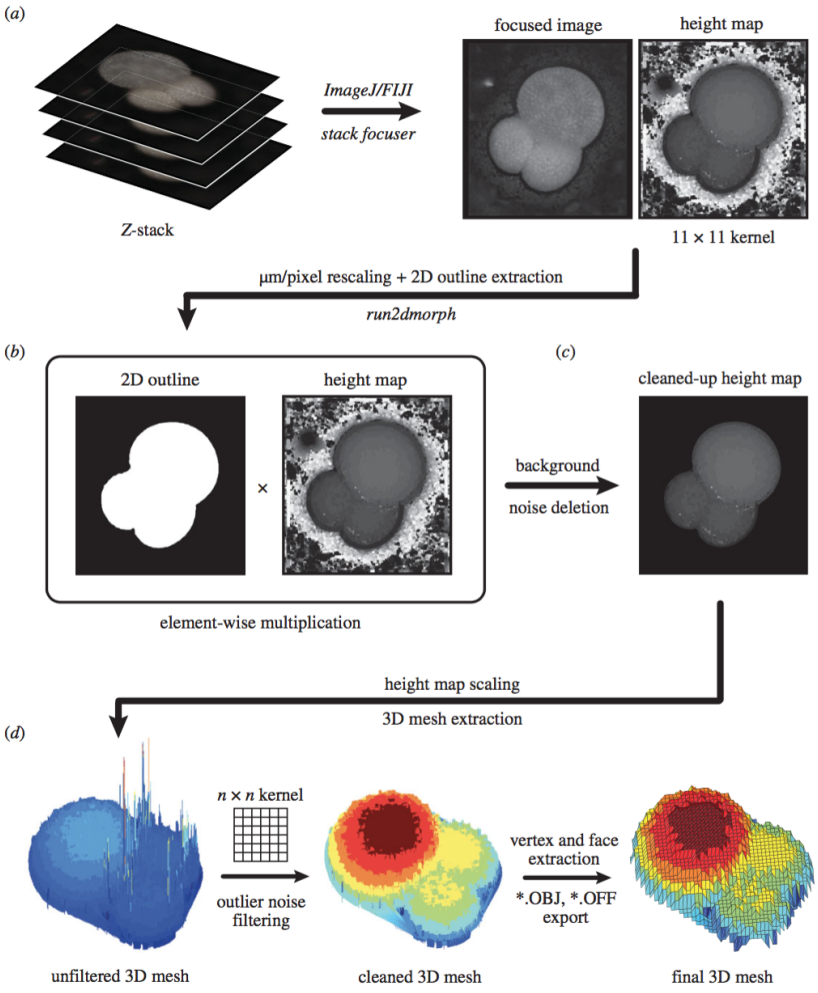

Figure 1: Visual pipeline of 3D mesh extraction using *run3dmorph*. From Hsiang *et al.* [3].

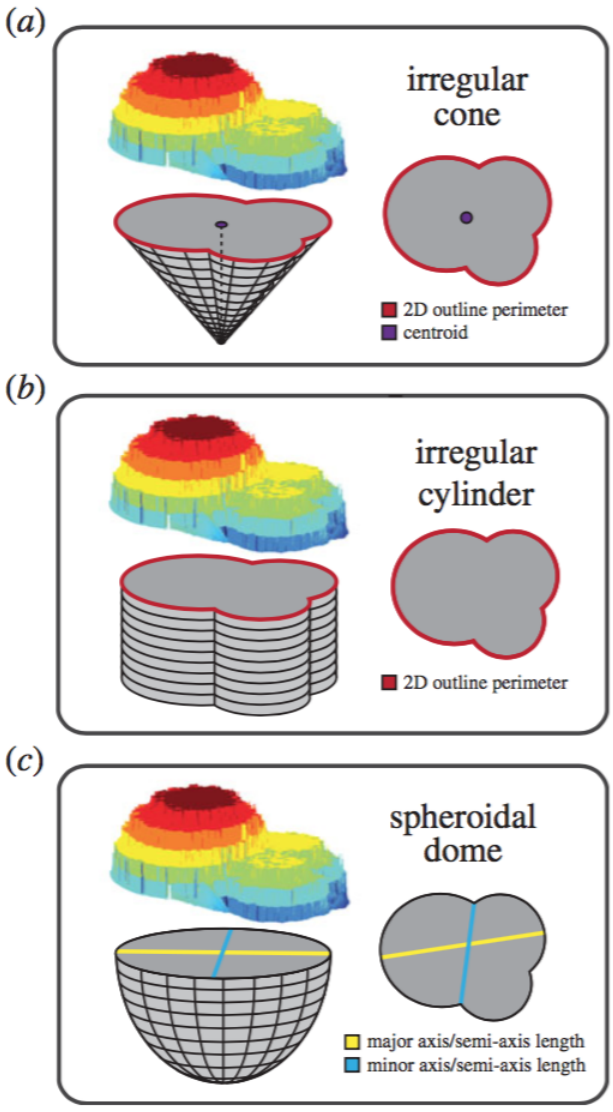

Figure 2: Idealized back shapes, used by *run3dmorph* to estimate surface area and volume of the entire object. From Hsiang *et al.* [3].