Focus **Manual**

Allison Hsiang

2016-11-22

I Introduction

The *focus* module of the *AutoMorph* software package (developed by Pincelli Hull and team [1]) is used to generate extended depth of field (EDF) images from a series of z-stack images via focus stacking, using either Zerene Stacker [2] or ImageJ/FIJI [3]. For the ImageJ/FIJI option, the StackFocuser plugin [4] is used. Zerene Stacker is commercially available, and produces the highest quality EDFs in our experience; however, the non-proprietary ImageJ/FIJI option is provided for users unwilling or unable to obtain a Zerene Stacker license. Zerene Stacker does allow users to download a 30-day free license, which allows users to test out the performance of the software before committing.

Focus is designed to work with the output generated by *segment*, the *AutoMorph* module for image segmentation. It takes as input a series of folders, each containing z-stack images of an individual light-colored object on a dark-colored background. As output, *focus* generates three folders: a folder named 'focused' that contains all individual object EDFs, with the metadata labels created by *segment*; a folder named 'focused_unlabeled' that contains all individual object EDFs, without the metadata labels created by *segment*; and a folder named 'stripped' containing the z-stack images for all identified objects without the metadata labels (used for downstream processing, namely as input for the *run3dmorph* module of *AutoMorph*). In addition, *focus* will compress and archive the original *segment* output z-stack images to conserve space. If the user is using the Zerene Stacker version of *focus*, a LOG file and XML file detailing Zerene Stacker's parameter settings will also be written to file. Figure 1 shows a overview of the *segment* and *focus* pipeline.

Focus, 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 *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!

II Installation

II.1 Prerequisites

Focus is written in Python and requires the following prerequisites:

- Python v.2.7
- Zerene Stacker (recommended)
- ImageJ or FIJI (alternative)

Python must be in your system's path and callable via the command 'python'. This should be the default behavior with a standard installation of Python on a Unix-like system. Python comes preinstalled on Mac OSX; this can be verified (and the version number checked) by typing 'python' into the command line in Terminal.

In order to run *focus* using ImageJ/FIJI, the user must have an installation of ImageJ/FIJI that is accessible from the path via one of the following commands:

- Mac: 'ImageJ-macosx'
- Linux: 'ImageJ-linux32' OR 'ImageJ-linux64'

On Mac OSX, this can be achieved by creating a symbolic link to the ImageJ/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/ImageJ-macosx
```

where IPATH is the installation location of FIJI, for example /Applications. After entering your system password when prompted, you will now be able to use FIJI from anywhere in your system, so *focus* will be able to call FIJI as necessary. If you elect to install ImageJ instead of FIJI, note that the ImageJ executable is found at ImageJ.app/Contents/MacOS/JavaApplicationStub. This should still be linked symbolically to /usr/local/bin/ImageJ-macosx as above; *focus* does not recognize the JavaApplicationStub command when running ImageJ. For this reason we recommend the user install FIJI instead of ImageJ to use with *focus*.

If using ImageJ/FIJI for focusing, the Stack Focuser plugin [4] must also be installed. The class file can be downloaded here; this file should be placed in the plugins folder in the user's ImageJ or FIJI installation (for FIJI on Mac

OSX, this folder can be accessed by right-clicking the FIJI application icon and clicking 'Show Package Contents').

II.2 Setup

Once you have downloaded the *AutoMorph* software package, you will find *focus* in the 'focus' folder. *focus* is run using the 'focus' executable located in this folder. Within this folder you will also find a configuration file (focus.cfg) that must be changed to match the user's system and preferences; details on this can be found in section IV.

We recommend adding the *focus* folder to your path, so that *focus* 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:

```
nano ~/.bash_profile
```

The nano text editor will open the .bash_profile, and you will likely see paths set by other programs (*e.g.*, Python). Scroll to the bottom of the of your .bash_profile (do not change anything that's already there unless you know what you're doing!) and then type:

```
#Setting PATH for focus
PATH="FPATH:${PATH}"
export PATH
```

where FPATH is the full path to your installation of *focus*. For instance, if *focus* is located at /Applications/AutoMorph/focus, the full entry would be:

```
#Setting PATH for focus
PATH="/Applications/AutoMorph/focus:${PATH}"
export PATH
```

Once this is done, exit nano (control + X), hit 'Y' when prompted to save, and then hit 'enter' to accept the original file name to write. You can now run *focus* from anywhere in your system (note that you may need to restart your shell for this to take effect).

III Quick Run

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

```
focus <path to directory containing z-stacks>
```

Unlike *segment*, *focus* does not use a control file; rather, the user supplies the path to the directory that contains the individual object z-stacks (*i.e.*, the path to the 'final' folder generated by *segment.* Note, however, that the *focus* configuration file must be properly set before running *focus* (see section IV).

Focus can take some optional arguments, namely:

- -v, --verbose: turns on verbose mode
- -i, --interactive: runs focusing software in interactive mode (note that this greatly slows down the performance speed of *focus*)
- --reset: reverts the input directory to pre-focused state
- --clean: removes the z.stack directory if a tar.gz archive version of the directory exists

In general, the average user will not need these optional arguments, and they are included here merely for documentation completion's sake.

IV Configuration File

The configuration file tells *focus* what program to use to generate the EDF and sets the necessary parameters based on this choice. The default configuration file is located at /AutoMorph/focus/focus.cfg. If the user wishes to use different configuration settings for *focus*, they may either place a copy of focus.cfg with the changed settings in the directory on which *focus* is being called, or edit the default focus.cfg file. *focus* will look for focus.cfg in the directory on which *focus* is being called first; if it cannot find focus.cfg there, it will default to using the focus.cfg file in the installation directory. Thus, if the user needs to use customized settings for a single sample, they can simply make a new focus.cfg file without tweaking the entire installation of *focus.* focus.cfg should be edited using a plain-text editor such as TextWrangler for Mac OSX. Example configuration files can be found in /AutoMorph/focus/example_cfg.

A list of the parameters in the configuration file follows:

IV.1 Focus Parameters

Software: the software to be used for focus stacking and generating the EDF. The two options are **zerene** for Zerene Stacker and **imagej** for ImageJ/FIJI. The default software is Zerene Stacker.

IV.2 ImageJ Parameters

These parameters only need to be set if the **Software** parameter is set to **imagej**.

Kernel Size: an odd integer n that sets a $n \times n$ kernel that is used by Stack Focuser to generate a heightmap. This parameter is necessary to set in order for Stack Focuser to run but is irrelevant for the EDF output, and thus can be ignored.

FIJI Architecture: can be set to either 32 or 64, corresponding to the 32-/64-bit versions of ImageJ/FIJI. This is only relevant for Linux systems and should be set as 'None' on Mac OSX systems, like so:

```
fiji_architecture = None
```

IV.3 Zerene Stacker Parameters

These parameters only need to be set if the **Software** parameter is set to **zerene**.

Zerene Directory: the full path to the location of the user's Zerene Stacker installation. On Mac OSX, the default location is /Applications/ZereneStacker.app.

System Memory in MB: the amount of system memory in MB available to allocate to Zerene Stacker's operation. A minimum of 4000 MB is suggested.

Temp Directory: the path to the temp directory for Zerene Stacker; on most systems, this should be /tmp.

Headless Mode: the command to use for running Zerene Stacker in headless mode (*i.e.*, without a graphical user interface (GUI). On Mac OSX, this should be set to **Xvfb**; on Linux systems, this should be set to **xvfb-run –auto-servernum –server-num=1** (or other appropriate server number depending on your setup). To disable headless mode, set this parameter as nothing.

V Hands-On Example Run

Included in the *focus* package is a set of z-stack images from four example objects that the user can use to test *focus*. The example files can be downloaded on Zenodo here. If the user completed the hands-on example run from the *segment* manual, they can also use the output from that example for this tutorial.

This tutorial assumes that the user is using a Mac OSX system with at least 8 GB of memory and Zerene Stacker installed at /Applications/ZereneStacker.app/. If the user meets these requirements, they do not need to change the configuration file focus.cfg at all; otherwise, the user must adjust the configuration file as necessary (see section IV). This tutorial also assumes that *AutoMorph* is installed at /Applications/AutoMorph and that the user has added the focus folder to the path (see section II.2).

1. Open the Terminal (/Applications/Utilities/Terminal.app).
2. At the command line prompt, type:

```
focus /Applications/AutoMorph/focus/examples/4sq
```

That's it! While running, *focus* will output messages updating its progress. When complete, the command line prompt will reappear. The output of *focus* will be in /Applications/AutoMorph/focus/examples/4sq.

VI Troubleshooting

If the user is using *focus* directory on successful output from *segment*, they are unlikely to run into any problems regarding input files. The most likely source of error while running *focus* is Zerene Stacker and appropriate configuration file settings. Zerene Stacker was not built to run via the command line, and our in-house experience suggests that it is prone to error, especially if multiple instances of it are being run at the same time. If issues arise, we recommend considering the following:

- Check that Zerene Stacker is up to date (the 'update me' dialog box can cause Zerene Stacker to hang even when headless mode is engaged)
- If using the 30-day free trial version of Zerene Stacker, check that this trial has not expired
- check that the configuration file is properly set for the focus stacking software being used
- If using *focus* in Zerene Stacker mode on a multi-user system, make sure that the Zerene Stacker license is accessible to all users who need it

References

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

[2] *Zerene Stacker*, Zerene Stacker LLC (http://zerenesystems.com/cms/stacker)

[3] 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.

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

[5] Hsiang AY, Nelson K, Dobbins B, Elder LE, Liu Y, Hull PM. AutoMorph: Accelerating community morphometrics with 2D and 3D image processing and shape extraction. *Methods in Ecology and Evolution. In prep.*

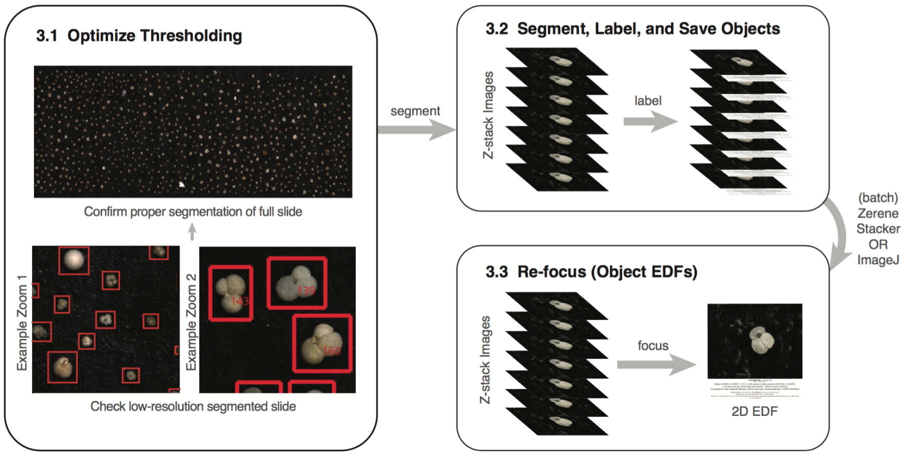
3.1 Optimize Thresholding
Confirm proper segmentation of full slide
Example Zoom 1
Example Zoom 2
Check low-resolution segmented slide
segment
3.2 Segment, Label, and Save Objects
Z-stack Images
label
(batch)
Zerene
Stacker
OR
ImageJ
3.3 Re-focus (Object EDFs)
Z-stack Images
focus
2D EDF

Figure 1: Overview of the image processing pipeline of the *segment* and *focus* modules of *AutoMorph*.