Segment (Ver.2016-7-12) Manual

Allison Hsiang

2016-11-22

I Introduction

Segment is the image segmentation module of the *AutoMorph* software package, which was developed by Pincelli Hull and team [1]. *AutoMorph* is available for download on GitHub. The *segment* module was originally written by Brian Dobbins in MATLAB and then translated to Python by Kaylea Nelson; it is described in detail in Hsiang *et al.* [2].

Segment takes as input a series of z-stack images containing light objects on a dark background and chops out each individual object (contingent on user-provided threshold and min/max size values) into its own stack of images. The z-stack images must be numbered in order of their height (*e.g.*, image001 is at height 0, image002 is at height 1, image003 is at height 2, image004 is at height 3, etc.). The user is able to test various threshold values and output low-resolution JPGs of the entire input field of view with segmented objects indicated by a red box (Fig. 1) and an automatically assigned object number (Fig. 2) in **sample** mode.

Once the thresholding parameter has been optimized in **sample** mode, the user can then run *segment* in **final** mode, which will chop out the objects identified using the specified threshold value, saving each z-stack to its individual folder labeled by object number (as determined by *segment*'s algorithm). *Segment* also generates a label (Fig. 3) containing metadata for each object that is appended to every z-stack image for a given object. In addition, under **final** mode, *segment* will output a text file listing the parameters used for the run and the low-resolution JPG overview image described above.

Segment, like all *AutoMorph* modules, is run using the command line. 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!

Segment 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 IV.

II Installation

II.1 Prerequisites

Segment is written in Python and requires the following prerequisites/modules:

- Python v.2.7
- scikit-learn
- scipy
- numpy
- pillow
- tifffile

We recommend installing the Anaconda distribution of Python, which features easy package managing/installation. For example, once Anaconda is installed on your system, the scipy package can be easily installed by typing the following command into the command line:

```
conda install scipy
```

II.2 Setup

Once you have downloaded the *AutoMorph* software package, you will find *segment* in the 'segment' folder. *Segment* is run using the 'segment' executable located in this folder. Within this folder you will also find an example control file (example_settings.txt).

We recommend adding the *segment* folder to your path, so that *segment* 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 segment
PATH="SPATH:${PATH}"
export PATH
```

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

```
#Setting PATH for segment
PATH="/Applications/AutoMorph/segment:${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 *segment* from anywhere in your system (note that you may need to restart your shell for this to take effect).

III Quick Run

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

```
segment <path to control file>
```

While running, *segment* will output status messages, including what mode it is running under (**sample** or **final**) and the number of objects found under the current parameters.

IV Control File and Parameters

The control file serves as the means by which the user supplies the necessary parameters for *segment* to operate. An example version of the control file can be found in AutoMorph/segment. Within the control file, each parameter is briefly described. It is recommended that the user read through the entire control file before running *segment*.

A listing of the parameters for *segment* follows:

Input Directory: the full path containing the z-stack images to be processed. This path cannot be a relative path. For example, if the images to be processed are located in /Applications/AutoMorph/segment/images, **/images** would not be acceptable as input.

Output Directory: the full path to where *segment*'s output should be saved. A subfolder (either named 'sample' or 'final' depending on if *segment* is run in **sample** or **final** mode, respectively) will be created in the specified directory.

Threshold: the threshold value used to convert the input images into black and white and detect objects. When running *segment* in **final** mode, this should be a single value, *e.g.*:

```
threshold = 0.16
```

When running *segment* in **sample** mode, a range of threshold values can be selected in the following format:

```
threshold = L - H by i
```

where *L* and *H* are the low and high end of the range of desired threshold values, respectively, and *i* is the increment value between each threshold value to be tested. For example, if the user wishes to test threshold values between 0.16 and 0.20 with an increment value of 0.01, they would enter:

```
threshold = 0.16 - 0.20 by 0.01
```

which would test threshold values of 0.16, 0.17, 0.18, 0.19, and 0.20.

Pixel Size X, Pixel Size Y: the number of units (as specified by the user under **Unit** in the **Labeling Parameters** (IV.2) per pixel in the input images, usually determined by the equipment and settings used to capture the original images. For instance, if one pixel in the input images is equal to 0.975 microns, and the pixels are square, then these parameters would be set as:

```
pixel_size_x = 0.975
pixel_size_y = 0.975
```

Minimum Size: the minimum valid width of an object identified using the current threshold setting in the user-defined **Unit** (*e.g.*, microns). This parameter is used to filter out background light scatter by excluding identified objects with widths smaller than the specified size. Like the **threshold** parameter, this parameter can be set as a range of values, which will result in segmenting results by each combination of threshold/minimum size values.

Maximum Size: as above, but the maximum valid width. This parameter is used to filter out border effects (e.g., rims, occluded objects, etc.) and other sources of large-scale noise.

Mode: set as either **sample** or **final**. **Sample** mode is used to optimize threshold and min-max filtering sizes and outputs a low-resolution JPG of all objects identified by *segment* given the test settings. **Final** mode is when the parameters are optimized and finalized and will output all segmented objects in individual folders in addition to the low-resolution JPG.

Skip Last Plane: a boolean (*i.e.*, True or False) specifying whether the last plane in the input z-stack should be excluded in the segmentation process or not. This is used when the last plane is not a true focal plane (for instance, an extended depth of field (EDF) image generated automatically by Surveyor software). By default, this parameter is set to True.

Box Thickness: a number that sets the thickness (in pixels) of the lines in the red boxes drawn around identified objects in the low-resolution JPG overview images. The default value is 20; the user may change this depending on the images to optimize visibility/clarity of the boxes.

IV.1 File Format Parameters

Input Extension: the file extension for the input images to *segment*; the default value is tif.

Output Extension: the file extension for the output object images (*i.e.*, the individual object z-stack images); the default value is tif.

IV.2 Labeling Parameters

Unit: the base unit of measure used in the pixel size conversion; the default value is microns.

Scale Bar Length: the length of the scale bar (in the unit specifed under the **Unit** parameter) that is outputted in the metadata labels generated by *segment*. The default value is 100; a smaller scale bar that is 1/4 this size is also generated.

ID: a name that serves as an identifier for the sample being processed. This name is used as a prefix in the names of output files and labels generated by *segment* (in combination with the **Catalog Prefix** parameter. If this parameter is left empty, *segment* will set this parameter by reading the input directory basename (*e.g.*, the last part of the path) and using everything preceding the first underscore (*e.g.*, if the input directory is /Applications/AutoMorph/segment/IP.307717_images, the **ID** will be set as **IP.307717**.

Age: a tag used for reporting the geological age estimation of the sample being processed; this will be printed to the metadata labels generated by *segment*.

Source: a tag used to provide a reference or indicate the source of the age estimation data specified under **Age**; this will be printed to the metadata labels generated by *segment*.

Location: a tag indicating the processing location of the images (*e.g.*, Yale Peabody Museum); this will be printed to the metadata labels generated by *segment*.

Author: a tag indicating the group or institution who performed the image processing; this will be printed to the metadata labels generated by *segment*.

Catalog Prefix: a tag indicating the library/museum catalog number prefix (if applicable) for the images being processed; this will be printed to the metadata labels generated by *segment* in combination with the specified **ID**. For example, if the catalog prefix is 'YPM IP', and the ID number is '5067', then the label will identify the sample as 'YPM IP 5067'.

V Hands-On Example Run

A set of example images (and associated XML files generated by the Leica DFC450 camera used to generate the images) that can be used to test *segment* can be downloaded on Zenodo here. A control file to use with these test images is located at /AutoMorph/segment/examples/segment_control_4sq_example.txt; the user need only change the **Input Directory** and **Directory Output** parameters to run the example. Step-by-step instructions – which assume that the user's installation of *AutoMorph* is in /Applications, and that the user has added the *segment* folder to the path – follow:

1. Unzip segment_4s_example.zip and place the unzipped contents in /Applications/AutoMorph/segment/examples/4sq.

2. Open segment_control_4sq_example.txt in a plain-text editor and change the **Input Directory** and **Output Directory** parameters to the appropriate paths:

```
# input directory:
directory = /Applications/AutoMorph/segment/examples/4sq

# output directory:
directory = /Applications/AutoMorph/segment/examples/4sq
```

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

```
cd /Applications/AutoMorph/segment/examples
```

4. Start *segment* by entering the following command:

```
segment segment_control_4sq_example.txt
```

The provided control file is set to run in **final** mode; as such, upon running, *segment* will proceed to identify the individual objects in the input slide and chop them out into individual object z-stacks. When *segment* is done running, the command line prompt will reappear. The output files will be located in: /Applications/AutoMorph/segment/examples/4sq/final. At a threshold value of 0.16, *segment* should have recognized and extracted a total of 28 objects.

VI Troubleshooting

The most common errors that arise during usage of *segment* occur due to inappropriate control file settings. In particular, *segment* will often quit because it is unable to find any input images; this will be apparent from the status messages outputted by *segment* (*i.e.*, *segment* will say that 0 images were found). If this occurs, the user is advised to double-check their **Input Directory** path and **Input/Output Extensions** and ensure they are correct.

If *segment* finds the input images but does not extract any objects, the problem likely lies with the **Threshold** parameter, **Minimum and Maximum Size** parameters, or both. In this case, the user is advised to try out different values for these parameters and determine the optimal values via trial and error.

References

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

[2] 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.*

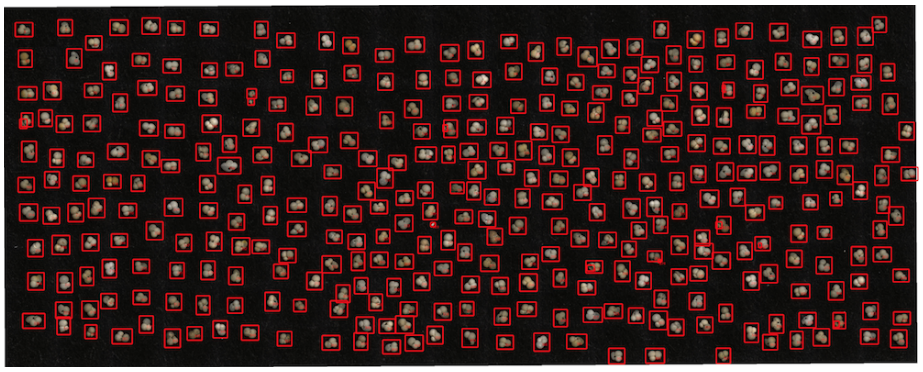


Figure 1: Example of *segment*'s low-resolution JPG output, showing the complete field of view from images inputted into *segment*, with segmented objects indicated by red boxes and numbering (numbering not visible at this scale; see Fig. 2).

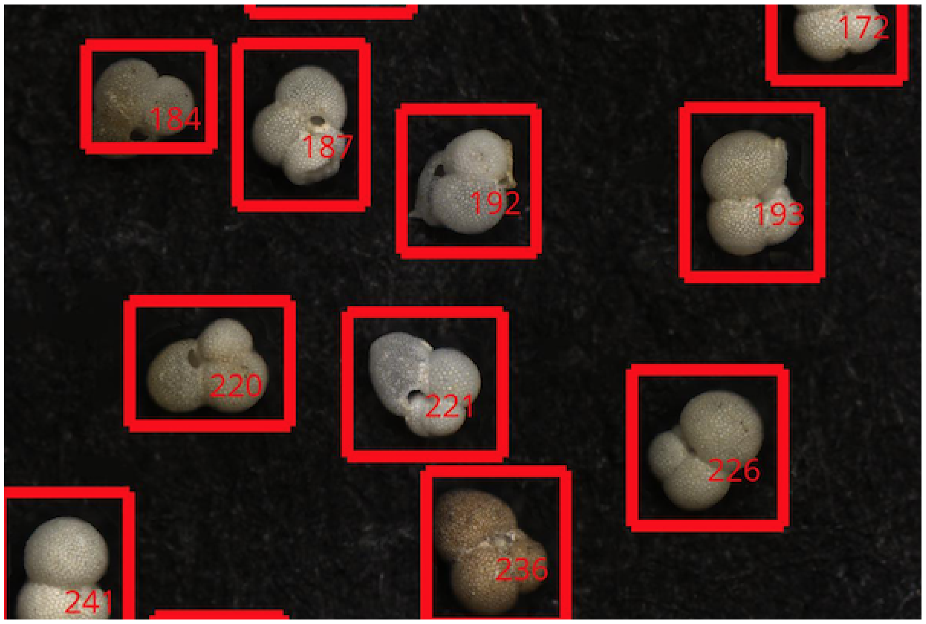
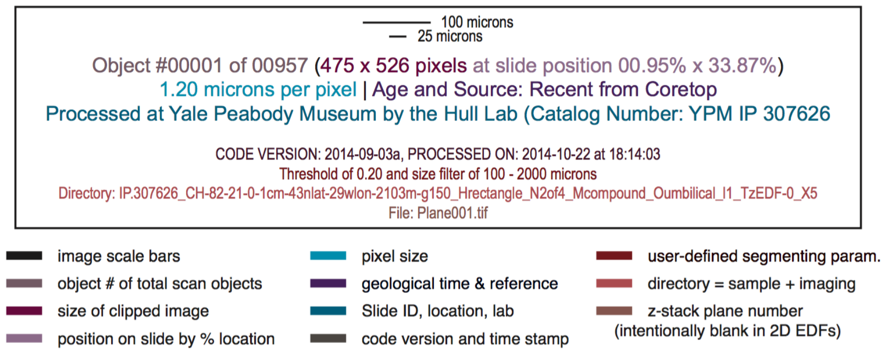


Figure 2: Close-up of *segment*'s output (Fig. 1), showing boxes around individual objects isolated by *segment* and automatic object numbering.



100 microns
25 microns

Object #00001 of 00957 (475 x 526 pixels at slide position 00.95% x 33.87%)
1.20 microns per pixel | Age and Source: Recent from Coretop
Processed at Yale Peabody Museum by the Hull Lab (Catalog Number: YPM IP 307626

CODE VERSION: 2014-09-03a, PROCESSED ON: 2014-10-22 at 18:14:03
Threshold of 0.20 and size filter of 100 - 2000 microns
Directory: IP.307626_CH-82-21-0-1cm-43nlat-29wlon-2103m-g150_Hrectangle_N2of4_Mcompound_Oumbilical_I1_TzEDF-0_X5
File: Plane001.tif

■ image scale bars	■ pixel size	■ user-defined segmenting param.
■ object # of total scan objects	■ geological time & reference	■ directory = sample + imaging
■ size of clipped image	■ Slide ID, location, lab	■ z-stack plane number (intentionally blank in 2D EDFs)
■ position on slide by % location	■ code version and time stamp	

Figure 3: Example metadata label generated by *segment*.