

# YeastPed: An Automated Pipeline for Genealogy Tracking of Budding Yeast from Time-lapse Microscopy Images

## Version 1 Summary, Documentation, and User Guide

Berk Yalcinkaya | Forsyth Country Day School, North Carolina School of Science and Math

### Overview

This program was designed to aid in cell image analysis of budding yeast. The program incorporates four important functionalities, which have been optimized for budding yeast, into an automated workflow:

- 1) Segmentation
- 2) Artefact Removal
- 3) Frame-to-Frame Tracking
- 4) Genealogy reconstruction

This pipeline incorporates both previously developed algorithms and a novel genealogy reconstruction method. A convolutional neural network (CNN) trained on segmented images of budding yeast (Dietler et al., 2020) performs the segmentation, and our artefact removal method utilizes a minimum life span and minimum cell area criteria. Tracking is based upon Lineage Mapper – a Hungarian algorithm-based tool optimized for cell tracking (Chalfoun et al., 2016). The final step, genealogy reconstruction, is based upon assumptions of mother-daughter proximity and the observation that the major axis of elliptical yeast daughter cells aligns with possible mother cells.

Based on benchmark of the genealogy' accuracy on three 200 image datasets, this novel algorithm correctly identified mother-daughter pairs 88.3% of the time.

### How it Works

Inputted images are first segmented via YeaZ's CNN and watershed method (Dietler et al., 2020). This step is optional, however, and we caution any users who rely on this user-out-of-the-loop segmentation method, as segmentation often requires user intervention to ensure accuracy. The automated nature of this software does not allow for user segmentation corrections and thus erroneous segmentation – which in turn reduces tracking and genealogy reconstruction accuracy – can occur. A recommended approach to use a GUI-based tool to segment and make corrections before inputting the segmented masks into the YeastPed pipeline.

Even though user oversight is recommended, YeastPed has implemented an artefact removal feature to make segmentation cleanup less tedious. Cells below a minimum, user specified threshold are removed and after tracking, cells with lifespans less than a user-specified threshold are removed as well. **Keeping this feature in mind, small and/or short-lived artefacts that**

**occur due to erroneous segmentation do not have to be cleaned up from the segmentation masks, reducing the user's time spent on fixing false positives during segmentation.** Instead, the user should focus on correcting incorrectly merged cells or false negatives to achieve maximum accuracy with minimum manual labor.

### The Genealogy Analysis Method

The cell tracking functionality is proposed as matrix assignment problem, and Hungarian optimization of a cost function is used to assign cells across frames. YeastPed utilizes NIST's Lineage Mapper (LM) algorithm (Chalfoun et al., 2016) due to its use of biologically relevant cost function parameters and user configurability (See 'Tracking Parameters' below).

Genealogy reconstruction begins by identifying a cell birth. At the cell's birth frame and the following two frames, this new cell is dilated using a 15x15 kernel and the overlap between the dilated cell area and the surrounding cells is calculated. If there is only one cell that overlaps with the dilated daughter cell, that one cell is determined to be its mother.

If there are multiple potential mother cells that overlap with this daughter; then at five, ten, fifteen frames after the daughter cell's birth, an ellipse is fit to the contour of the daughter cell and a line is plotted over the extended major axis of the ellipse. Surrounding cells that are consistently intersected by this major axis line in all three of the frames where the axis is plotted are considered potential mothers. The potential mother from this list with the greatest overlap with the dilated daughter cell is determined to be the mother.

If no cell is consistently detected by this major axis line, then maximum overlap with the dilated daughter cell mask is used as the determining criteria for motherhood.

### Downloading YeastPed

1. Ensure you have a MATLAB license and [version of MATLAB compatible with python 3.6.8](#) installed on your computer.
2. Clone this repository ("git clone <https://github.com/berkylcinkaya/YeastPed>")
3. Download the neural network weights (CNN trained by Dietler et al) for segmenting phase contrast images from: [https://drive.google.com/file/d/1UTivmx\\_aEMpeGdOkCZO1CS9mcdJ3zmw2/view?usp=sharing](https://drive.google.com/file/d/1UTivmx_aEMpeGdOkCZO1CS9mcdJ3zmw2/view?usp=sharing). Put the file in the folder /unet. **Note:** this step is not necessary if you do not plan to use YeastPed to segment automatically
4. Download the parameters for segmenting bright-field images (CNN trained by Dietler et al) from: [https://drive.google.com/file/d/1VYBzUtgLQcS-w6S9XpjGcSBackYltYJ\\_/view?usp=sharing](https://drive.google.com/file/d/1VYBzUtgLQcS-w6S9XpjGcSBackYltYJ_/view?usp=sharing). Put the file in the folder /unet. **Note:** this step is not necessary if you do not plan to use YeastPed to segment automatically
5. If you don't have conda or miniconda installed, download it from <https://docs.conda.io/en/latest/miniconda.html>.
6. In the command line, create a virtual environment with python 3.6.8 with the command `conda create -n YeastPed python=3.6.8`.
7. Activate that environment using `conda activate YeastPed`
8. Install the necessary packages using `pip install -r requirements.txt`.

## Usage

### Parameters

The pipeline is written in Python and MATLAB and has been configured to be run via a command line interface using the Python module argparse. The parameters can be displayed by typing either of the following commands in a command prompt window:

```
python Main.py --help or python Main.py -h
```

Upon running this command, the general parameters, tracking parameters, and segmentation parameters are displayed.

#### General Parameters:

Parameter	Description	Default	Example
-image_path	Path to raw images. <b>Note:</b> Even if segmentation of raw microscopy images is not desired, raw images will be used when creating color mapped visualizations of the processed images	REQUIRED	<code>'python Main.py -image_path /Users/berk/Images/Agar7.tif'</code>
-segmented_image_path	Path to segmented image masks. Providing an argument for this parameter will bypass segmentation.	If no segmented image path is given, the raw images provided by next to -image_path will be segmented and processed	<code>'python Main.py -image_path /Users/berk/Images/Agar7.tif -segmented_image_path Images/image.tif'</code>
-output_dir	The path following -output_dir is the folder in which all output files will be located. If the path after '-output_dir' does not yet exist, it will be created for you.	If [-output_dir /path/to/dir] is not included, the default output directory becomes the directory passed with -image_path	<code>'python Main.py -image_path /Users/berk/Images/Agar7.tif -output_dir some/file'</code>

-output_filename	The name prefix given to all the output images/file	Default is 'output'. Examples: output_masks.tif, output_values.csv	'python Main.py -image_path /Users/berk/Images/Agar7.tif -output_dir some/file -segment -output_filename Agar7'
-start_frame	Frame on which segmentation and/or tracking will begin	0	'python Main.py -image_path /Users/berk/Images/Agar7.tif -segment -start_frame 1 -end_frame 10'
-end_frame	Frame on which segmentation and/or tracking will end	Number of frames inputted.	'python Main.py -image_path /Users/berk/Images/Agar7.tif -segment -start_frame 1 -end_frame 10'
-get_flou	Supplying the '-get_flou' argument extracts cell statistics and fluorescence values from inputted images	False. In absence of '-get_flou', no fluorescence or cell statistics will be extracted	'python Main.py -image_path /Users/berk/Images/Agar7.tif -output_dir some/file -segment -output_filename Agar7 -get_flou'
-flou_image_path	Path to fluorescence images. Only needed if '-get_flou' is used.	If '-get_flou' is passed without supplying a filepath via [-flou_image_path some/path], then the images via the -image_path arguments will be used as a fluorescence channel	python Main.py -image_path /Users/berk/Images/Agar7.tif -output_dir some/file -segment -output_filename Agar7 -get_flou -flou_image_path some/path.tif

### Segmentation Parameters

Note that segmentation is an optional feature of this pipeline. Running the built-in segmentation algorithm can often result in erroneous segmentation due to the absence of user intervention opportunity. It is thus recommended that segmentation accuracy be ensured via a GUI based segmentation algorithm and segmented images are input into the program via the -segmented\_image\_path parameter (see above for General Parameters).

Parameter	Description	Default	Example
-threshold	Probability threshold used to	0.50	'python Main.py -image_path /Users/berk/Images/Agar7.tif -segment

	determine cell bodies from YeaZ's neural network prediction		-threshold 0.80'
-seed_distance	Minimum distance between seeds during watershed method	10	'python Main.py -image_path /Users/berk/Images/Agar7.tif -segment -threshold 0.80 -seed distance 10'

### Tracking and Artefact Removal Parameters:

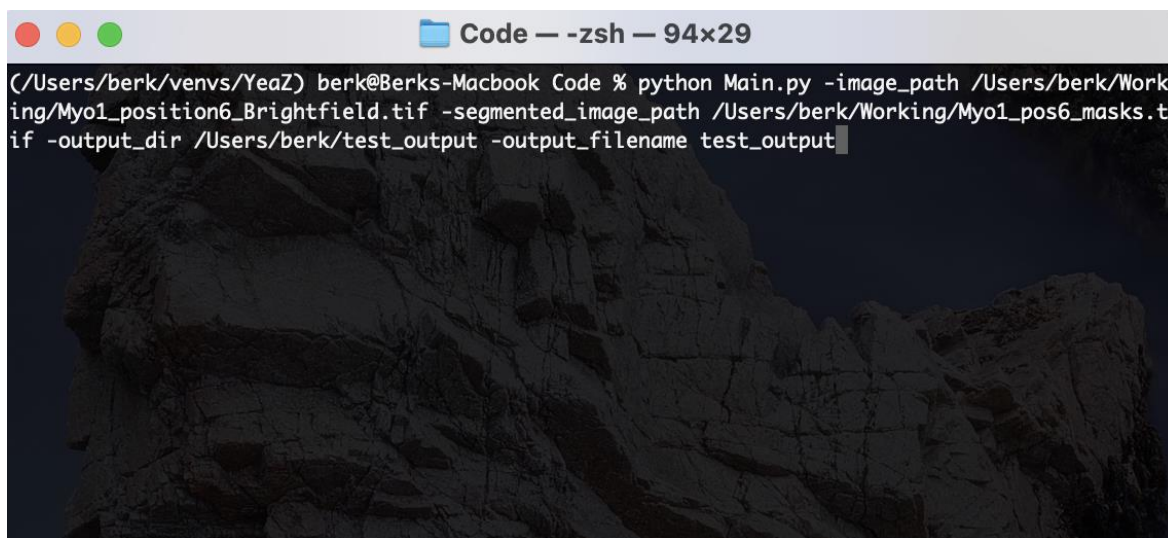
As mentioned previously, LineageMapper's cell tracking algorithm was optimized for budding yeast. The cell tracking parameters below can be configured, but based on observation, they **do not need to be changed under normal circumstances**. If tracking errors seem to be occurring, modifying the cost function parameters – which determines the features used to assign cells from one frame to the next – is the best way of tailoring the algorithm to your images. Please refer to the [LineageMapper user guide](#) for more information about the tracking parameters used in this pipeline.

Parameter	Default Value	Description
-weight_overlap	95	Cost function: weight assigned to the overlap of cells between frames (should be high for immotile yeast)
-weight_centroids	50	Cost function: weight assigned to the distance between cell centroids from one frame to next
-weight_size	10	Cost function: weight assigned to the size of a cell from frame to frame
-max_centroid_distance	75.0	Cost function: The radius from a cell centroid to the max centroid distance represents the area of a possible cell migration.
-cell_size_threshold	100 pixels	Minimum size for a segmented 'blob' to be considered a cell. Also used by LM's cell fusion detection (see user guide)
-enable_cell_fusion	False	Cell Fusion: enables cell fusion (should not be set to true for yeast)

-Fusion_overlap_threshold	50%	Cell Fusion: minimum overlap between frames for two cells to be considered incorrectly joined (see user guide)
---------------------------	-----	--

## Usage Examples/Instructions

The following screen shots display several examples of program usage.

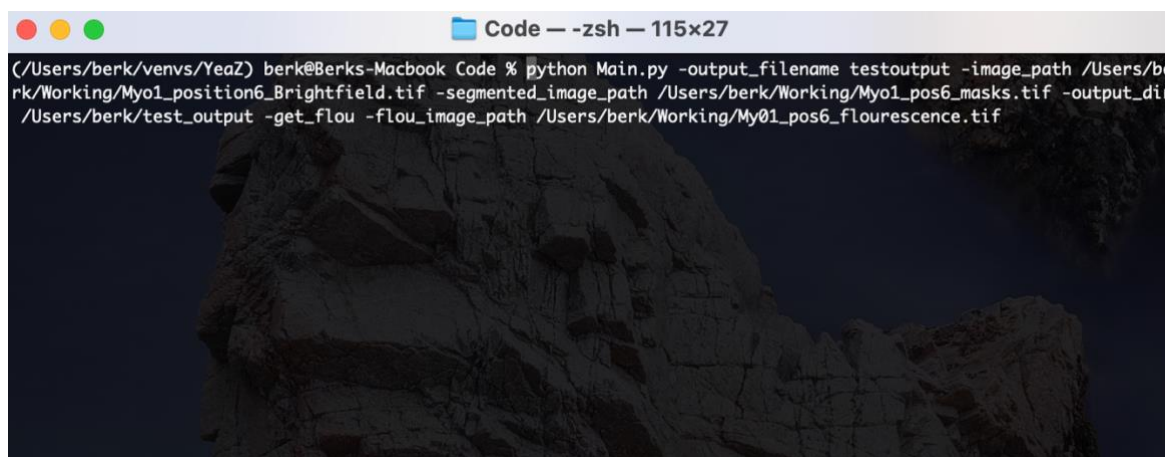


```
Code — -zsh — 94x29
(/Users/berk/venvs/YeaZ) berk@Berks-Macbook Code % python Main.py -image_path /Users/berk/Working/Myo1_position6_Brightfield.tif -segmented_image_path /Users/berk/Working/Myo1_pos6_masks.tif -output_dir /Users/berk/test_output -output_filename test_output
```

The above command will run the pipeline without segmentation. All output will end up in /Users/berk/test\_output and all files that are output by the program will have test\_output as a prefix.

A few notes and tips on using the command line:

- 1) The arguments can come in any order.
- 2) All arguments must be preceded by dashes
- 3) No commas or quotation marks should be included. The arguments are parsed by space separation



```
Code — -zsh — 115x27
(/Users/berk/venvs/YeaZ) berk@Berks-Macbook Code % python Main.py -output_filename testoutput -image_path /Users/berk/Working/Myo1_position6_Brightfield.tif -segmented_image_path /Users/berk/Working/Myo1_pos6_masks.tif -output_dir /Users/berk/test_output -get_flo -flo_image_path /Users/berk/Working/Myo1_pos6_florescence.tif
```

In the above example, we have added two additional parameters ‘-get\_flou’ and ‘-flou\_image\_path’. Notice that ‘-get\_flou’ does not have a corresponding variable adjacent to it like the other parameters, its presence determine whether or not fluorescence is extracted.

## Output

### Directories

This program outputs several directories and files. Two directories are automatically created in the output\_dir directory and are used by Lineage Mapper. They are called ‘segmented’ and ‘tracked’. In the ‘segmented’ folder are the separated frames of the segmented images that are either produced by the built-in segmentation or inputted by the user. The ‘tracked’ folder holds the results of the LineageMapper tracking.

Note that in the ‘tracked’ directory, there exists a .mat file which can be input into the LineageMapper GUI to reproduce the tracking results. See the [LineageMapper user guide](#) for details about loading a session from a .mat file.

### Lineage

This program outputs a csv file called lineage.csv which contains each cell, its birth frame and death frame, its proposed mother cell, and the cells that were detected by the major axis line method, if it was used. If the major axis method was not run, the column “Major Axis Overlap” will be N/A. If no cells were detected on the major axis overlap, then the column “Major Axis Overlap” will be \*.

Keep in mind that cells born near the end of the frame will also display an asterisk in the column “Major Axis Overlap” because there were not enough frames to compute the major axis and thus no cells could be detected along their major axes.

The column titled correct is present as an option for user correction. By looking at the annotated and color mapped image stack produced by this program (see “Images” subsection below), the user can determine which mother-daughter pairs were correctly identified and can make corrections.

### Fluorescence

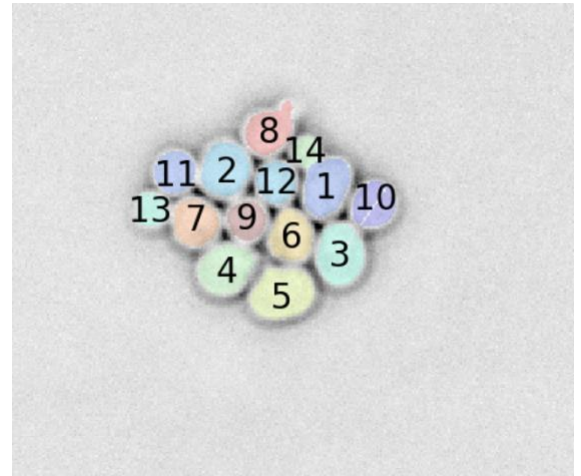
If fluorescence is computed, the fluorescence along with cell morphology values will be output in a csv file that end with \_flou.csv

## Images

The final processed and tracked masks will be output in a stack tif file that ends with `_masks.tif`.

Most importantly, a color mapped and annotated image of the masks overlaying the raw images will be produced. This color mapped visualization displays cell numbers and allows the user to easily visualize the tracking and segmentation results. This ease of visualization is especially important for correcting lineage data.

By scrutinizing this color mapped image, you can see the faint lines that represent the major axis plots used to determine lineages.





### Works Cited

Chalfoun, J., Majurski, M., Dima, A., Halter, M., Bhadriraju, K., & Brady, M. (2016). Lineage mapper: A versatile cell and particle tracker. *Scientific Reports*, 6(1), 36984.

<https://doi.org/10.1038/srep36984>

Dietler, N., Minder, M., Gligorovski, V., Economou, A. M., Joly, D. A. H. L., Sadeghi, A., Chan, C.

H. M., Koziński, M., Weigert, M., Bitbol, A.-F., & Rahi, S. J. (2020). A convolutional neural network segments yeast microscopy images with high accuracy. *Nature*

*Communications*, 11(1), 5723. <https://doi.org/10.1038/s41467-020-19557-4>