Reference document: Tracking Cells and Analyzing images

AMOLF [*]

July 29, 2015

1 Installation

All the guidelines here apply to Windows 7 system, but similar procedures could be followed in other operational systems. To run the tracking software described in [1], a PC with a GPU with CUDA capabilities[1] is required. If you have this, install the CUDA toolkit, available in `https://developer.nvidia.com/cuda-downloads`.

1.1 Download the C program

1. Download the ZIP file in `http://www.janelia.org/sites/default/files/Labs/Keller%20Lab/TGMM_Supplementary_Software_1_0.zip`
2. Unzip the file into a desired folder
3. Copy the file `$C_FOLDER$/data/TGMM_configFile.txt` to any folder you want, so that you can edit it while keeping the original[2]
4. It is highly recommended that you read the `$C_FOLDER$/README.txt` file

1.2 Download GitHub

If you do not have GitHub installed, download it from `https://windows.github.com/`. Be familiar with the simple git functions, it is indeed very simple and useful. The task here is to clone a repository (where the analysis code is) to your machine, in a folder that we will refer as `$GIT_FOLDER$`. Is you want to do it as fast as possible, follow the simple instructions (up to step 3) in `https://help.github.com/articles/getting-started-with-github-for-windows/`. The URL of the repository is `https://github.com/epolimpio/image_extraction.git`.

If you want to start to change the code from this version, just fork the project to your Desktop. This option is the best if you want to keep developing the code using the functionalities of GitHub. The instructions to do it are found in `https://guides.github.com/activities/forking/`.

[*]1st version by Eduardo Olimpio

[1]More information found in `http://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html#axzz3ejDLnsIT`

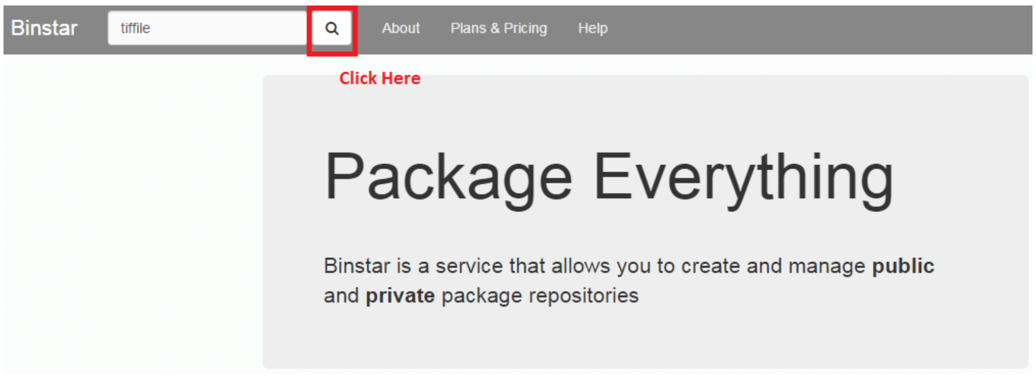
[2]We refer to `$C_FOLDER$` as the folder where you extracted the C software

1.3 Install Python

All the analysis are done in Python, so if you do not have Python installed, it is time to do so. For Windows I recommend the Anaconda distribution, but feel free to install any distribution you want. We will need some extra packages, which I explain below. The easiest way install the packages is by using the command line[3] and type the command:

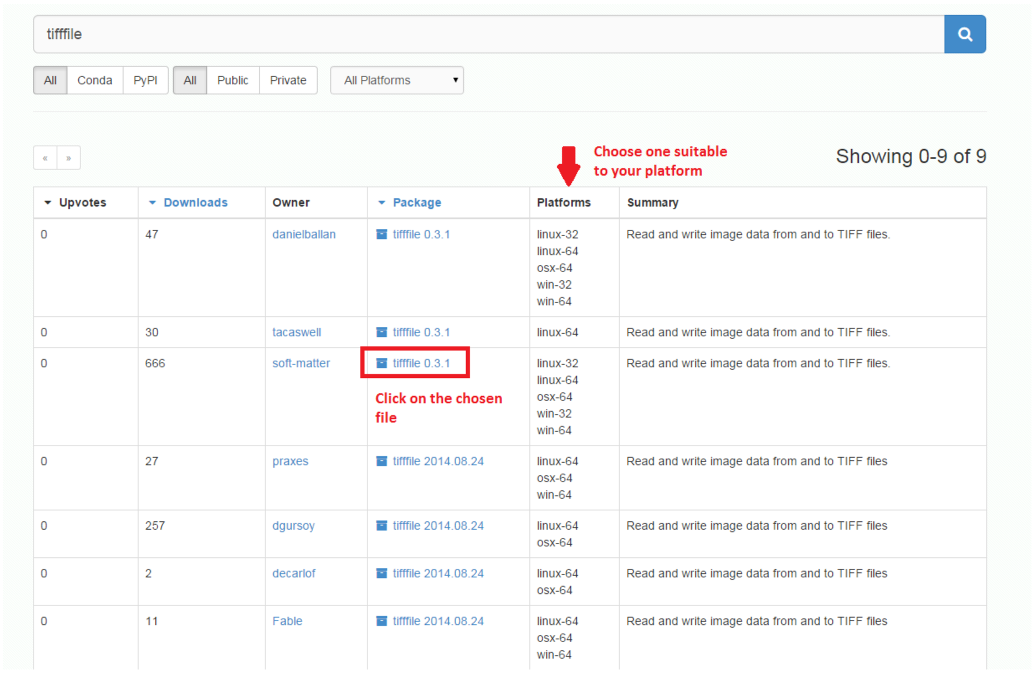
```
conda install <package_name>
```

If the package is not available, the easiest way to get it is going to `https://binstar.org/` and search for the package in the top left of the page. In the example below we show how to do it to install the *tifffile* package.



Package Everything

Binstar is a service that allows you to create and manage **public** and **private** package repositories



Showing 0-9 of 9

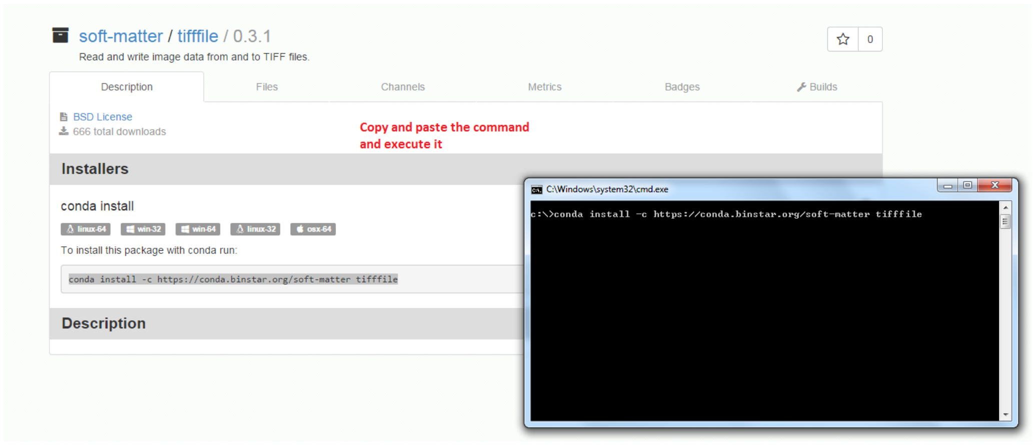
Upvotes	Downloads	Owner	Package	Platforms	Summary
0	47	danielballan	tifffile 0.3.1	linux-32 linux-64 osx-64 win-32 win-64	Read and write image data from and to TIFF files.
0	30	tacaswell	tifffile 0.3.1	linux-64	Read and write image data from and to TIFF files.
0	666	soft-matter	tifffile 0.3.1	linux-32 linux-64 osx-64 win-32 win-64	Read and write image data from and to TIFF files.
0	27	praxes	tifffile 2014.08.24	linux-64 osx-64 win-64	Read and write image data from and to TIFF files
0	257	dgursoy	tifffile 2014.08.24	linux-64 osx-64	Read and write image data from and to TIFF files
0	2	decarlof	tifffile 2014.08.24	linux-64 osx-64	Read and write image data from and to TIFF files
0	11	Fable	tifffile 2014.08.24	linux-64 osx-64 win-64	Read and write image data from and to TIFF files

IMPORTANT NOTE: All the development was made in Python 3 and there could be some problems running in Python 2. If you want to avoid these problems, you can create a Anaconda environment and run in Python 3 without affecting the Python 2 installation.

1.3.1 Necessary Packages

There are several extra packages needed to run the analysis. The packages marked with * come with the basic Anaconda installation.

[3]Accessible in Windows by running `cmd.exe` in the run prompt (Windows+R in the keyboard)



Read and write image data from and to TIFF files.

BSD License
666 total downloads

Copy and paste the command and execute it

Installers

conda install

To install this package with conda run:

```
conda install -c https://conda.binstar.org/soft-matter tifffile
```

Description

```
c:\>conda install -c https://conda.binstar.org/soft-matter tifffile
```

1. tifffile
2. lxml
3. numpy*
4. matplotlib*

1.3.2 If not using Anaconda

I found that for some things (e.g. compiled code) Anaconda does not behave properly. If you switch back to the normal Python, you will need to get the binaries of the packages in `http://www.lfd.uci.edu/~gohlke/pythonlibs/`. Then use the command line and go to the forder where the binaries are and type:

```
$PYTHON_PATH$\Scripts\pip.exe install <name_of_the_binary>
```

where we assume that the installation of Python is in the folder `$PYTHON_PATH$`

1.4 Running a test of the C software

To figure out if the tracking software is properly working, you need to change the configuration file and run an example (already provided with the software) and check if there are results coming out from it.

1.4.1 Changing the configuration file

The configuration file contains many parameters used to run the tracking software. It is important to be familiar with how to use this, as explainend in `https://www.janelia.org/sites/default/files/Labs/Keller%20Lab/TGMM_UserGuide.v2.pdf`. Here we will teach the basics on how to run the tracking program.

1. Open the copy of the configuration file you made during installation of the tracking software, in the path that we call `$CONFIG_PATH$`
2. Change the line containing imgFilePattern to imgFilePattern=`$C_FOLDER$/data/data/TM?????_timeFused_blending/SPC0_CM0_CM1_CHN00_CHN01.fusedStack_?????`
3. Change the line containing debugPathPrefix to debugPathPrefix=`$RESULTS_FOLDER$`

Here, replace `$C_FOLDER$` by the folder where you extracted the tracking program and `$RESULTS_FOLDER$` by a folder of your choice created to handle the results of the tracking program. Each run of the program will generate a new folder inside the chosen one. Remember that the folder of the variable imgFilePattern MUST use slash and for debugPathPrefix you MUST use backslash.

1.4.2 Changing the batch file

The batch files are used to make it easier to run the tracking software. There are three different batch files in the folder `$GIT_FOLDER$`:

1. `segmentation.bat`: Used to run the pre-Track program, required to run the main program.

2. `bayesian_pos.bat`: Runs the main program

3. `seg_analysis.bat`: Used to run the segmentation analysis in debug (with report) if the segmentation program fails.

We need to change these files to provide them with the right path. To edit these files, right click on them and select “Edit”. The changes you must provide in all of them are the same. You need to change the line with `c_folder` to include the path you extracted the tracking software (referred here as `$C_FOLDER$`) and the line containing the set of `config_path` should be changed to `$CONFIG_PATH$`.

1.4.3 Running the software

To run the program now, just execute the file `segmentation.bat` (double/click or use command line) and use initial frame 0 and final frame 30. If the program has sucessfully run, you will get the message *“Hierarchical segmentation ran successfully”*. Then you can run the file `bayesian_pos.bat` and use again initial frame 0 and final frame 30. If everything runs OK, you will find a new folder inside the `$RESULTS_FOLDER$` in a format similar to `GMEMtracking3D_yyyy_mm_dd_hh_mm_ss` and inside it you will have two folders with the XML and the binary (`.svb`) files, and a `.txt` with the log of your run.

2 Running the Tracking Software and the Analysis

In order to analyze the data, you need to follow the steps:

1. Cut the area of interest and make the data compatible to that required by the tracking software

2. Run the tracking algorithm

3. Run the eye check procedure to generate the analysis image

4. Curate the data using the manual track

2.1 Before running the track analysis

Before running the analysis make sure that the TIFF images are 16-bit. The program runs only for 8-bits or 16-bits one color images. Usually the images come in different folders (one for each time step) and with several stacks per time frame (for 3D compositions). Therefore, the path pattern resembles something like `../T?????/T?????Z@@@` where the symbols `?` and `@` are replaced by the corresponding time and stack integers. As an example, for the second frame (Time = 2) and the tenth stack (Z = 10) we have `../T?????/T?????Z@@@`.

To generate the stacked TIFF image, you need to run the script `gen_stacks_for_tracking.py`, following the steps:

1. Copy the file `config_stacked_images.conf` to the image folder (we will call it `$IM_FOLDER$`). The path of the copied configuration file is `$IM_CONFIG_PATH$`.
2. Check the images, contained in the path (example) `$IM_FOLDER$/T?????/T?????Z@@@` and see how you want to cut the images (if needed). To check the cutting parameters, use a nice image visualizing software. In Figure 1 we use IrfanView as an example.
3. Edit the configuration file (in `$IM_CONFIG_PATH$`) and include all the parameters required (section 2.1.1)
4. Open the command window, go to the `$GIT_FOLDER$` and execute `py gen_stacks_for_tracking.py <$IM_CONFIG_PATH$>`

2.1.1 Writing the configuration file

The program generates the stacked TIFF file cut in the region you chose and output it according to the pattern you choose. All the required parameters are saved in the file `$IM_CONFIG_PATH$`, called together with the function. An example file is below:

```
# ====================================================
# number of time frames
n_time = 129

# initial time
t_ini = 1

# number of z-components per time frame
n_z = 29

# Cutting parameters, used only if cut = True
cut = True
cut_x = 600
cut_y = 500
pos_x = 300
pos_y = 200

# In - out file path
path_in = "D:\\data\\25 tif\\T?????C01Z@@@.tif"
path_out = "D:\\data\\25 tif\\T?????C01.tif"
# ===================================================
```

1. **n_time:** Number of frames to be included in the stacked file

2. **t_ini:** Starting frame number. In the present example it starts at $T = 1$ and goes until $T = 129$
3. **n_z:** Number of stacks to be included (the first is assumed to be 1)
4. **cut:** True is you want to cut, False if not
5. **pos_x, pos_y:** Define the corner position of the region to be cut
6. **cut_x, cut_y:** Size of the region to be cut
7. **path_in:** Input pattern, use `?` for the time and `@` for Z
8. **path_out:** Output pattern of the stacked image, use `?` for the time

To illustrate how to choose the cut parameters, we used IrfanView to select the desired region and show how to determine the parameters in Figure 1.

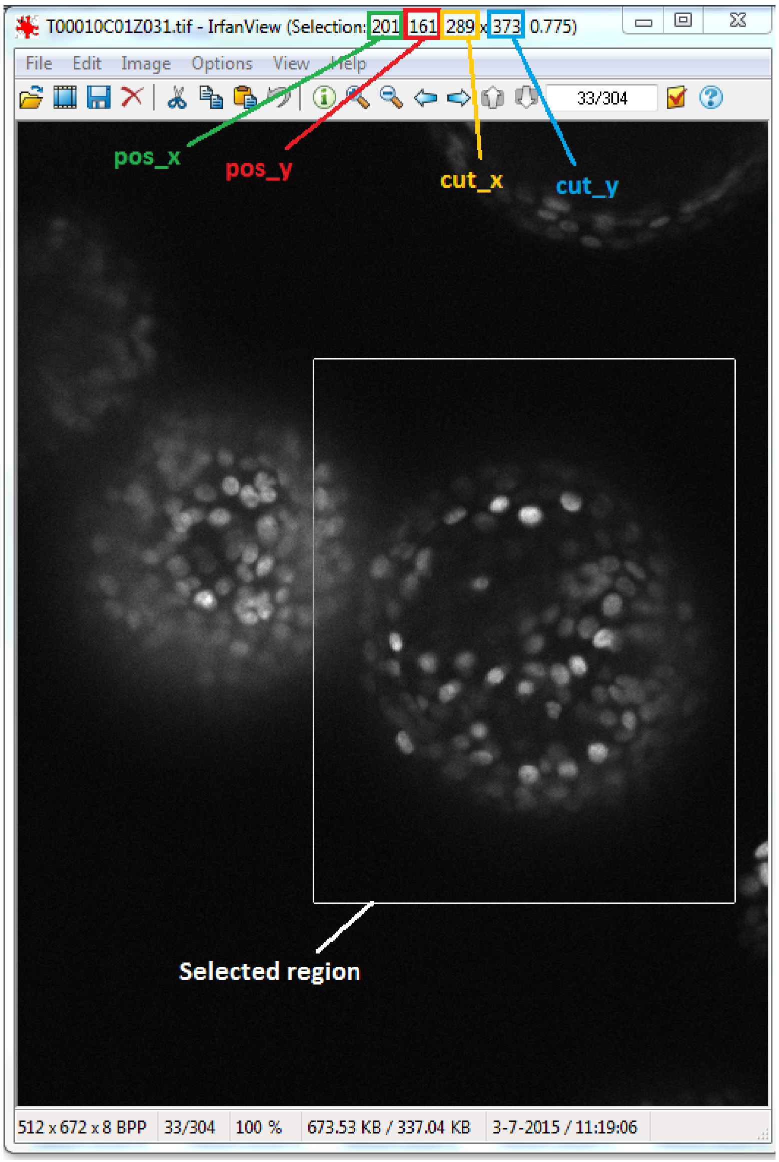

T00010C01Z031.tif - IrfanView (Selection: 201 161 289 x 373 0.775)
File Edit Image Options View Help
33/304
pos_x
pos_y
cut_x
cut_y
Selected region
512 x 672 x 8 BPP 33/304 100 % 673.53 KB / 337.04 KB 3-7-2015 / 11:19:06

Figure 1: Cutting region determination

References

[1] Fernando Amat, William Lemon, Daniel P Mossing, Katie McDole, Yinan Wan, Kristin Branson, Eugene W Myers, and Philipp J Keller. Fast, accurate reconstruction of cell lineages from large-scale fluorescence microscopy data. *Nature methods*, 11(July), 2014.