Open SEM

Release 0.1

Snow Wang

Sep 19, 2022

MICROSCOPY

Open SEM is a site providing free tutorials for serial electron microscopy (SEM), EM datasets, and open code for image processing and data analysis.

This repository provides computational tools for practicing biologists to analyze volumetric image data at large-scale. These tools focus on in-depth analyses **after** automatic/semi-automatic segmentation. We use examples of 3D data acquired from serial electron microscopy for demonstrations, but all the codes and scripts should be able to deal with other types of 3D images such as fluorescence microscopy image stacks.

The intended audience is the *practicing biologists/neuroscientists* - e.g., the students, researchers, and clinicians collecting volumeric image data in the hospital or laboratory. Some sections of this material can get pretty math-heavy, but we have tried to outline the main concepts as directly as possible, with hands-on implementations of all concepts.

Note: This project is under active development.

CHAPTER

ONE

CONTENTS

This site provides information and resources for dealing with images acquired using Serial Electron Microscopy. We are presnting the entire computational pipeline here.

CHAPTER

TWO

CONTRIBUTORS

This repository is created by Snow Wang. Core developers and their contributions are listed below:

- Website Building & Contents Creation: Snow, Zhirui (Ray) Shen
- Microscopy Tutorials: Snow, Yulan (Mumu) Fang
- EM Tutorials: Snow
- Image Processing Tutorials: Snow
- Data Analysis: Snow, Yutian (Tim) Fan, Wenjie (Kelly) Yin
- Data Visualization: Bowen Gong, Kelly, Snow

Former Contributors: Bowen (Nola) Zheng, Ruilin Cai

CHAPTER

THREE

SUPPORTED IMAGE DATA

All the tools presented in this repository are designed to process or visualize annotated 3D datasets exhibiting the following features:

- labeled/segmented serial cross-sectional images
- segmentation files are sequence of 8-bit, 16-bit, or 24-bit grayscale images
- .PNG files are preferred

CHAPTER

FOUR

SUPPORTED LANGUAGES & SOFTWARE

- Python
- MATLAB
- VAST
- 3ds Max

CHAPTER

FIVE

RELATED PROJECTS

- 3D reconstruction of mitochondria
- JWR15 Project

5.1 Digital Image

Intro to digital image

5.2 Electron Microscopy (EM)

To retrieve a list of random ingredients, you can use the `lumache.get_random_ingredients()` function:

The `kind` parameter should be either `"meat"`, `"fish"`, or `"veggies"`. Otherwise, `lumache.get_random_ingredients()` will raise an exception.

5.3 Serial EM

Intro to SEM

5.4 How Cells Appear in EM

Intro to cellular ultrastructure

5.5 Cellular Organelles in EM

Show various Organelles images

5.6 Neurons in EM

Show various neurons images

5.7 Image Registration

5.7.1 Image Stitching

2D

5.7.2 Image Alignment

3D

5.8 Image Segmentation

5.8.1 Semantic Segmentation

examples

5.8.2 Instance Segmentation

examples

5.9 Feature Detection

Link several algorthms

5.10 Data Annotation

VAST intro

5.11 Data Proofreading

Contributor: Haishan Zhang

Here we are sharing our proofreading pipeline to show how this procedure can be standardized.

5.11.1 Preparations & File Management

- **Always work on your own file**
  - Do not make changes on the original .vsseg file or other people's versions
  - Before working on a new cell, make sure to duplicate the source file and store it in your designated folder (preferably a local folder)
- **Naming convention:**
  - Your folder's name: Yourname
  - Cell's folder's name: pyr#_Yourname
  - Version's name: pyr#_ddmmyy_Yourname
- Save your work regularly (e.g. every 2-3 hours). Save your segmentation as a new version at the beginning of each day and make sure to save it at the end of the day.
- So, a good working folder will look like this:

5.11.2 VAST Tutorial

1. **Open the files you need:**
   - .vsvi: the image volume
   - .vsvi that says "seg" or something similar": the auto segmentation
   - .vsseg: the manual segmentation you will be working on
   - Open these files by dragging the file to the software or click file->open (recent) image volume/open (recent) segmentation
2. Make sure that the auto segmentation is not visible. Toggle the alpha value to adjust the transparency of the segmentation (the teal part in Picture 1) to your preference. The recommended alpha value is between 10 - 40 so that your segmentation layer is visible, but not blurring details such as cell membrane or vesicles.
3. Select the "draw segment mode" (the pencil icon). If you want to navigate under this mode, press ctrl while moving your cursor.
4. Turn on z-scrolling. This allows you to draw continuously through multiple pages as you set your pen on one point and flip forward or backward.
5. Enable masking. Choose your auto segmentation file as the source. Choose "auto pick source color" as the method.
6. **Go to the starting point by right clicking your "segment color" (Usually it's the color other than the background, with the name being your cell#). Select "go to anchor point".**
   - Note that when your mouse stays at these floating windows (i.e. the windows mentioned in D, E, F), any changes such as moving cursors or pressing keys will apply to contents in the windows. If the software is not functioning as you wish, come back to check whether any change is accidentally made. For instance, if you pressed up/down while the cursor is on the "segment colors" window, you might now be on the background layer, where you cannot edit anything.
   - If no anchor point is set or it was set at a wrong place, find the cell body and click "set anchor point". Regarding how to find the cell body, ask someone to help if you cannot ;)
7. Press a or up to flip forward, in which case the z coordinate becomes smaller. Press z or down to flip backward, in which case the z coordinate becomes bigger.

Note: Note that forward/backward, smaller/bigger are just a relative way to talk about how we navigate through the pages. To get a better understanding of how you are traveling through a cell, you will need to open the neuroglancer and explore what coordinates mean in the 3d model.

8. Use your mouse scroll to zoom in or out. **Always work on mip level 1 or 0**.

5.11.3 WACOM Tutorial

- Press the lower button for "erase" mode
- Press the upper button to adjust the pen diam
- If you need to adjust the pen's tip feel, mapping, and other properties, go to wacom's app (just search "wacom").

5.11.4 Tracing

Our main goals are:

- Sometimes other axons or dendrites are identified and auto-segmented as part of the target dendrite due to membrane break. **Erase** them whenever you see them.
- If there are vast numbers of pages missing segmentation or the segmentation is too coarse, having pixel-like edges, **report** them to your supervisor.
- **Find** out the missing spines. See more below.

Though the three elements in picture 3 are ubiquitous to all synapses, the shape of each spine (including head and neck) looks very different on EM images depending on on which panel you are observing it. You will get better at identifying spines and painting them across pages with some practice.

Some pro tips:

- **When first working with EM images, take some time to navigate back and forth and visualize:**
  - Given a certain page, can you project the segmentation onto the 3d structure? Illustrations that may help you to conceive:
  - When navigating in one direction, can you relate the 2D-shape change to the 3D-structure change?
- Always note down the coordination where you have questions. **When in doubt, ask!**
- Remember that in most cases, protrusions mean active spines. Do not give up until you see the synapse. However, there are cases where protrusions end without forming a head and a synapse. If you have just started to proofread, ask your supervisor if a questionable tail really ends without the head (chances are that you just overlooked the head ;)
- When you encounter a longitudinal section, the segmentation (i.e., the colored shape) can be very long and big. In order to keep track of everything, remember to first zoom out to get a rough idea about how the dendrite is going (in both a&z directions). Then split the whole shape into several parts and track each part individually.
- If in your image, there are too many synapses popping up and dwindling down, do not forget to go back and forth to check 1-2 times after you think you've finished everything. You never really finish all the spines!
- Common mistake for newbies: on 2D images, a synapse will disappear like dissolving in the darkness. Sometimes a similar-sized oval will appear immediately, this is someone else. Do not fill this in as it may lead you to merge your dendrite with a random passer-by.
- Once again, **when in doubt, ask!**

5.11.5 Note Taking

If you look at the pyramidal cells (and other types of neurons), you will find that neurites always branch into two at any branch point.

On our images, this feature looks like the following images:

If the branches go in x/y direction, you will find (on 2D image) one shape split into two smaller shapes when going in either a/z direction.

However, if the branching happens in z direction (i.e. the direction along which you are traveling through the neuron), you will find one sub-branch goes in the original a/z direction, and the other in the opposite z/a direction. Under this circumstance, you will never see both sub-branches on the same page.

With that said, our note taking method is:

- Mark the beginning of a selected dendrite as integrals (For instance, 1)
- **Whenever at a branching point:**
  - Mark down the page where it is about to branch
  - Create new label as, for instance, 1.1 and 1.2
  - Mark down the starting point of 1.1 and 1.2 immediately
- **Repeat until the dendrite ends**
  - In most cases, dendrites end in two ways: it reaches page 0/the last page (i.e. the z border); it goes out of the image (i.e. the x/y border).
  - Very very rarely, a dendrite might end in the middle for some reason. Always consult your supervisor before marking it as a special ending.

Note: Remember to copy and paste coordinates after you place the target segmentation in the middle. In other words, make sure the crosshair is amid your segmentation. Since there are usually multiple segmentation shapes on one page (such as in picture 1, 8, and 10), this allows you to quickly pinpoint what your notes refer to.

Note: There's no requirement on how detailed your notes should be. The bottom line is you take down every branching point (recall: the starting point + the beginning points of the two sub-branches). However, it is suggested to:

- Leave a mark whenever a dendrite branch ends. Since you have a beginning point on record, chances are that you take it as a finished branch and skip it. An ending mark is thus really helpful.
- Take down some midway coordinates. Sometimes dendrites grow spines rampantly but do not branch for a long time. This can drive you nuts. Take down some midway coordinates when you (and the dendrite!) are sober. This helps you to step back and begin again when you get lost in the jungle of spines.
- Sometimes the dendrite zigzags in the z direction, which means you need to go back and forth frequently. Make a note at the turning point and label a/z, this helps you to step back and begin again whenever needed.

Without saying, it is also important to take notes of any interesting findings. After you finish proofreading a neuron, please store the note in the corresponding folder.

Voila! Now you can open VAST and try proofreading :)

5.12 Exporting Data from VAST

code

5.13 3D Rendering Using Neuroglancer (ng)

Contributors: Kelly Yin and Snow Wang

Neuroglancer is a data visualization tool developed by Google. To set up Neuroglancer locally on your PC, please follow the tutorial below.

Note: Currently Neuroglancer is only supported on Windows Machines.

5.13.1 Prepare for ng installation

To prepare your Windows computer for ng installation, first install these applications:

- Install Visual Studio - (download the free community version)
- Install Anaconda
- Install Git (call Anaconda Prompt or Command Prompt in your window search option)

```
>>> conda install -c anaconda git
```

- Install chocolatey
  - Open Windows PowerShell and run as Administrator
  - In **PowerShell**, run:

```
>>> Get-ExecutionPolicy
```

  - If it returns `Restricted`, then run `Set-ExecutionPolicy AllSigned` or `Set-ExecutionPolicy Bypass -Scope Process`.
  - In **PowerShell**, run:

```
>>> Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.
↪ServicePointManager]::SecurityProtocol = [System.Net.
↪ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object System.
↪Net.WebClient).DownloadString('https://community.chocolatey.org/install.
↪ps1'))
```

  - Now check if choco is installed. In **PowerShell**, type:

```
>>> choco
```

  - If it returns the message below, you are ready to go.
- Install nodejs
  - In **PowerShell**, run:

```
>>> choco install -y --force nodejs
```

- If returns the message below, you are good to move on.
- Test nodejs. In Command Prompt, run:

```
>>> node -v
```

- If it returns the message below, you are good to proceed.

5.13.2 Install Neuroglancer

To install neuroglancer and the related components, we want to first create a new enviroment (let's call it `ng_torch`) and install PyTorch for it:

- Open **Anaconda Prompt** or **Command Prompt**, run:

```
>>> conda create -n ng_torch python=3.8
>>> activate ng_torch
>>> conda install pytorch torchvision cudatoolkit=11.0 -c pytorch
```

- Now check your PyTorch version (we need the version to be >1.80):

```
>>> pip3 show torch
```

- If you need to update `pip` in the virtual enviroment, run

```
>>> python -m pip install --user --upgrade pip
```

- Now use `pip` to install ng and other packages you need:

```
>>> pip install neuroglancer
>>> pip install jupyter  # (optional) jupyter/ipykernel installation
>>> pip install numpy Pillow requests tornado sockjs-tornado six google-apitools␣
↪selenium imageio h5py cloud-volume
>>> python -m pip install -U scikit-image
```

- Make a folder for ng:

```
>>> mkdir project
>>> cd project
>>> git clone https://github.com/google/neuroglancer.git
>>> cd neuroglancer
>>> curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.0/install.sh | bash
export NVM_DIR="$([ -z "${XDG_CONFIG_HOME-}" ] && printf %s "${HOME}/.nvm" || printf %s "
↪${XDG_CONFIG_HOME}/nvm")" \
    [ -s "$NVM_DIR/nvm.sh" ] && \. "$NVM_DIR/nvm.sh"    # This loads nvm
```

- Install npm. In **Anaconda Prompt**, run:

```
>>> pip install npm
```

- After installation, run:

```
>>> npm i
```

- Finally, in **Anaconda Prompt**, run:

```
>>> python setup.py install
```

Note: If you encounter error messages in the last step, run this code before re-run the last line:

```
>>> npm run build-python-min
```

5.13.3 Use Jupyter Notebook to set up your ng viewer

Open **Anaconda**, locate your ng enviroment and start a `Jupyter Notebook`:

- In the notebook, run the code blocks in sequence:

```
import neuroglancer
import numpy as np
from skimage.io import imread
import h5py
import os
```

- Set up the local server:

```
ip = 'localhost'  # or public IP of the machine for sharable display
port = 9999       # change to an unused port number
neuroglancer.set_server_bind_address(bind_address=ip, bind_port=port)
viewer=neuroglancer.Viewer()
```

- If your reconstruction has been exported as an image stack, this code loads your entire image folder. In this case, we are loading a folder named *jwr_pyr87* containing 773 image sections:

```
script_dir = os.path.abspath('') # locate the folder where the current script is being␣
↪run
sample_name = 'jwr_pyr87' # put your image folder in the script path and specify the␣
↪name of the folder
img_dir = os.path.join(script_dir, sample_name)
img_idx = sorted(next(os.walk(img_dir))[2])
num_of_img = len(img_idx)
sample_height = 832 # specify the exported image size in x
sample_length = 832 # specify the exported image size in y
img_shape = (sample_height, sample_length)
img_stack = np.zeros((len(img_idx),) + img_shape, dtype=np.int64) # allocate memory
print(img_stack.shape)

i = 0
for i in range(num_of_img):

    img_stack[i] = imread(img_dir + "/" + img_idx[i])
    i += 1
```

(continues on next page)

(continued from previous page)

```
print(img_stack.shape) # read all the images exported from VAST into a single image stack
```

- If your reconstruction file is in .h5 format, use the code below to load your image stack:

```
with h5py.File('C:/Users/Lichtman Lab/Desktop/h5_data/jwr_pyr87.h5', 'r') as fl:

    img_stack = np.array(fl['images'])
```

- Set the x,y,z resolutions for the ng viewer:

```
res = neuroglancer.CoordinateSpace(
    names=['z', 'y', 'x'],
    units=['nm', 'nm', 'nm'],
    scales=[120, 256, 128]) # set the x,y,z resolutions for neuroglacer
```

- Add a layer in ng viewer to show the segmentation/reconstruction:

```
def ngLayer(data, res, oo=[0,0,0], tt='segmentation'):

    return neuroglancer.LocalVolume(data, dimensions=res, volume_type=tt, voxel_
↪offset=oo)
```

- Configure the ng layers: (in this case, we are loading a precomputed EM volume)

```
with viewer.txn() as s:
    s.layers['em'] = neuroglancer.ImageLayer(source='precomputed://https://rhoana.rc.fas.
↪harvard.edu/ng/jwr15-120_im')
    s.layers.append(name='seg', layer=ngLayer(img_stack.astype(np.uint8), res, tt=
↪'segmentation'))
```

- Generate a link for your ng viewer:

```
print(viewer)
```

- Obtain the complete segment list for the segmentation layer:

```
np.unique(img_stack)
```

- Please feel free to download the sample Jupyter [notebook] or Colab notebook whichever is convenient for you.

5.14 3D Rendering Using 3ds Max

Contributors: Bowen Gong and Kelly Yin

5.14.1 Animation

In this section, we are going to create a rotation animation using the software 3ds Max. When complete, this animation will allow you to demonstrate your object in 360 degrees with controllable speed, framerate, and resolution. For instance, the short gif below demonstrates 4 parts of a block of brain cells of rat, each rotates 360 degrees and lasts for 10 seconds.

Nonetheless, this tutorial will allow you to create a similar animation above, demonstrating anything you want, and you can expect a higher resolution and frame rate for your work since this file is compressed for convenience.

5.14.2 Overview & Introduction

There is something important we need to mention before we start the actual procedure.

- The software, 3ds Max is a **Windows-only** software, so please get a pc. More importantly, we do recommend that you use a computer with a discrete graphics card and better have **RAM larger or equal to 16G**.
- You don't need any background knowledge to fluently complete this section.
- You can get a **free license** of the software if you work for an academic institute or are a student, you can get the free license from the official website. But if you are neither of the 2 above, you can only use the software for 30 days for free and you will be charged later.
- The tutorial cost you about **40 minutes to go through**, however, we also need to wait for our computers to complete the rendering of the animation. So, depending on your specs, extra hours may be needed to complete the rendering. Feel free to take a nap.

5.14.3 Related features we applied

3ds Max is one of the most popular computer graphics programs for model design, animations, and digital images. Despite its other fancy and marvelous features that create dazzling effects, the functions we are applying are simple and efficient.

- **Max script**, the built-in scripting language for 3ds Max, is designed to complement 3ds Max, and the syntax is simple enough for non-programmers to use.
- **Auto Key Mode animation**, allows the user to animate the position, rotation, and scale of an object using auto-created keyframes.
- **Set Key mode animation**, allows you to create keys for selected objects individual tracks using a combination of the Set Keys button, which has more flexibility than Auto Key Mode.

Some basic modeling functions are also used to create a frame around the object to make the boundaries more obvious, and it is up to you to decide whether to have it or not. Please don't be confused by all the definitions above, all of them will be explained in detail in the following parts. If you are still confused after go through the tutorial or, if something you want to achieve is not included, please go to the Autodesk Knowledge Network for more tutorials and information.