HAAG Week 8 Report -Lizard Jaw Segmentation

Shuyu Tian

Time-Log

What did I do this week?

- I began integration of gaussian mixture model used in previous exploration in the registration code provided by Philip Woolley from last semester.
- As the webpage manager and now meeting leader of my group, I updated the webpage of the Stroud group with a subpage for the Lizard Jaw Segmentation group's weekly reports and group meeting recording link.

- What I will do next week
  - I will continue to conduct data collection work on 3D Slicer for additional labelled lizard image sets
  - I will follow the suggestion by the computational advisor to conduct coarse point matching with registration using the RANSAC approach to find an average distance or another accuracy metric to see the likelihood of matching onto an unlabeled point clouds
  - I will provide findings and feedback on the that usage when we meet with the computational advisor again next week
- Blockers, things I want to flag, problems, etc.
  - No major blockers right now to note, will report back after the computational advisor meeting

Abstract:

Lately, there has been an emphasis on the importance of studying inter-individual variation in animal behaviour and cognition and understanding its underlying mechanisms. What was once considered mere noise around population mean can be explained by individual characteristics such as brain morphology and functionality. However, logistical limitations can be faced when studying the brain, especially for research involving wild animals, such as dealing with small sample sizes and time-consuming methods. Here, we combined an efficient and accurate method using X-ray micro-tomography and deep-learning (DL) segmentation to estimate the volume of six main brain areas of wild lizards, *Podarcis bocagei*: olfactory bulbs, telencephalon, diencephalon, midbrain, cerebellum and brain stem. Through quantitative comparison, we show that a sufficient deep-learning neural network can be trained with as few as five data sets. From this, we applied the trained deep-learning algorithm to obtain volume data of the six brain regions from 29 brains of *Podarcis bocagei*. We provide a detailed protocol for our methods, including sample preparation, X-ray tomography, and 3D volumetric segmentation. Our work is open-access and freely available, with the potential to benefit researchers in various fields, such as animal physiology, biomedical studies, and computer sciences.

Link: https://www.biorxiv.org/content/10.1101/2024.07.05.602071v1

General summary: The study titled "Brain Virtual Histology of a Lizard Species (Podarcis bocagei) Using X-ray MicroCT" presents a novel approach to examining lizard brain morphology. Utilizing X-ray micro-computed tomography (microCT), the researchers generated high-resolution, three-dimensional images of the lizard's brain, enabling detailed visualization of its internal structures without the need for traditional histological sectioning. This method offers a non-destructive alternative to conventional techniques, preserving the specimen for future analyses. The findings demonstrate the potential of microCT imaging in advancing our understanding of reptilian neuroanatomy and its variability among individuals.

What did you do and prove it

I refactored code used previously to explore gaussian mixture model (GMM) for label prediction through 3 changes:

1. The original registration.py now has extracted source and target point clouds being clustered using GMM
2. The most frequent cluster label is selected now to hopefully reduce noise and focus registration on relevant regions.
3. The prepare_dataset function integrates GMM-enhanced point clouds now and are converted to Open3D format and then fed into the preprocess_point_cloud function for feature extraction

Some relevant code segment to produce the results are below:

```
def process(max_iters=5):
    iternum = 0
    reg_icp = None
    result_ransac = None
    stop = False
    while(not stop):
        #do preprocess (downsampled)
        voxel_size = 15
        source, target, source_down, target_down, source_fpfh, target_fpfh = prepare_dataset(
                        voxel_size, source_file, target_file)
        #do global registration
        result_ransac = execute_global_registration(source_down, target_down,
                                                    source_fpfh, target_fpfh,
                                                    voxel_size)
        st = source_down.transform(result_ransac.transformation)
        #draw_registration_result(st, target_down, np.eye(4))
        print(result_ransac.fitness, result_ransac.inlier_rmse)
        #do crop
        bbox = st.get_minimal_oriented_bounding_box()
        bbox.extent = bbox.extent * 2
        tt = target_down.crop(bbox)
        #o3d.visualization.draw_geometries([st, bbox, tt])
        #do preprocess
        source, target, source_down, target_down, source_fpfh, target_fpfh = prepare_dataset(
                        3, source_file, target_file)
        st = source_down.transform(result_ransac.transformation)
        tt = target_down.crop(bbox)
        #do local registration
        threshold=1000
        trans_init = np.eye(4)  # initial transformation
        reg_icp = o3d.pipelines.registration.registration_icp(
                        st, tt, threshold, np.eye(4),
                        o3d.pipelines.registration.TransformationEstimationPointToPoint(with_scaling=True),
                        o3d.pipelines.registration.ICPConvergenceCriteria(max_iteration=1000, relative_fitness=.00000000000001, relative_rmse=.00000000
                        )
        print(reg_icp.inlier_rmse)
        st = st.transform(reg_icp.transformation)
        #draw_registration_result(st, target_down, np.eye(4))
        #do crop
        bbox2 = st.get_minimal_oriented_bounding_box()
        bbox2.extent = bbox2.extent * 1.15
        tt = tt.crop(bbox2)
        #o3d.visualization.draw_geometries([st, bbox2, tt])
        if(reg_icp.inlier_rmse < 6):    #if rmse of ICP registration <5, good fit achieved
            stop = True
            print('Sample met stopping condition for minimizing error')
            #draw_registration_result(st, tt, np.eye(4))
        if(iternum >= max_iters):       #if maximum number of retries occurs with no good fit, notfiy user to process manually
            stop = True
            print('Reached max iterations, alignment of this sample should be adjusted manually')
        iternum = iternum + 1
    return result_ransac, reg_icp
```

```python
import numpy as np
import nrrd as pynrrd
import open3d as o3d
import copy
from sklearn.mixture import GaussianMixture

def preprocess_point_cloud(pcd, voxel_size):
    pcd_down = pcd.voxel_down_sample(voxel_size)

    radius_normal = voxel_size * 2
    pcd_down.estimate_normals(
        o3d.geometry.KDTreeSearchParamHybrid(radius=radius_normal, max_nn=30))

    radius_feature = voxel_size * 5
    pcd_fpfh = o3d.pipelines.registration.compute_fpfh_feature(
        pcd_down,
        o3d.geometry.KDTreeSearchParamHybrid(radius=radius_feature, max_nn=100))
    return pcd_down, pcd_fpfh

def volume_to_point_cloud(volume, threshold=38000):
    temp = np.asarray((volume > threshold))
    z, y, x = temp.nonzero()
    points = np.vstack((x, y, z)).T  # Transpose to get points in (N, 3) format
    return points

def run_gmm_clustering(points, num_clusters=3):
    points_mean = np.mean(points, axis=0)
    points_std = np.std(points, axis=0)
    points_norm = (points - points_mean) / points_std

    gmm = GaussianMixture(n_components=num_clusters, covariance_type='full', random_state=42)
    labels = gmm.fit_predict(points_norm)  # Cluster assignments

    return labels, gmm

def prepare_dataset(voxel_size, source_file=None, target_file=None, num_clusters=3):
    source_data, _ = pynrrd.read(source_file)
    target_data, _ = pynrrd.read(target_file)

    source_points = volume_to_point_cloud(source_data)
    target_points = volume_to_point_cloud(target_data)

    # Apply GMM clustering
    source_labels, source_gmm = run_gmm_clustering(source_points, num_clusters)
    target_labels, target_gmm = run_gmm_clustering(target_points, num_clusters)

    # Filter to retain only the dominant cluster (assuming the largest represents the structure of interest)
    dominant_source_cluster = np.argmax(np.bincount(source_labels))
    dominant_target_cluster = np.argmax(np.bincount(target_labels))

    filtered_source_points = source_points[source_labels == dominant_source_cluster]
    filtered_target_points = target_points[target_labels == dominant_target_cluster]

    # Convert to Open3D point cloud format
    source_pcd = o3d.geometry.PointCloud()
    source_pcd.points = o3d.utility.Vector3dVector(filtered_source_points)

    target_pcd = o3d.geometry.PointCloud()
    target_pcd.points = o3d.utility.Vector3dVector(filtered_target_points)

    return preprocess_point_cloud(source_pcd, voxel_size), preprocess_point_cloud(target_pcd, voxel_size)
```

Additionally, I updated the Stroud lab webpage with my group's relevant information up to week 7 of this semester (see images below).

Home / Lizard Jaw Segmentation /

Lizard Jaw Segmentation Group Meetings and Recordings

Updated On February 28, 2025

Spring 2025 Week 8 Meeting Recording

Weekly Lizard Jaw Meeting-20250226_170110-Meeting Recording.mp4

Spring 2025 Week 7 Computational Advisor/Group Meeting Recording

Weekly Lizard Jaw Meeting-20250219_170647-Meeting Recording.mp4

Spring 2025 Week 6 Meeting Recording

Weekly Lizard Jaw Meeting-20250212_170746-Meeting Recording.mp4

Spring 2025 Week 5 Computational Advisor Meeting Recording

Lizard Jaw Segmentation_ Student Researcher _ Computational Advisor Meet-20250206_171303-Meeting Recording.mp4

Spring 2025 Week 4 Meeting Recording

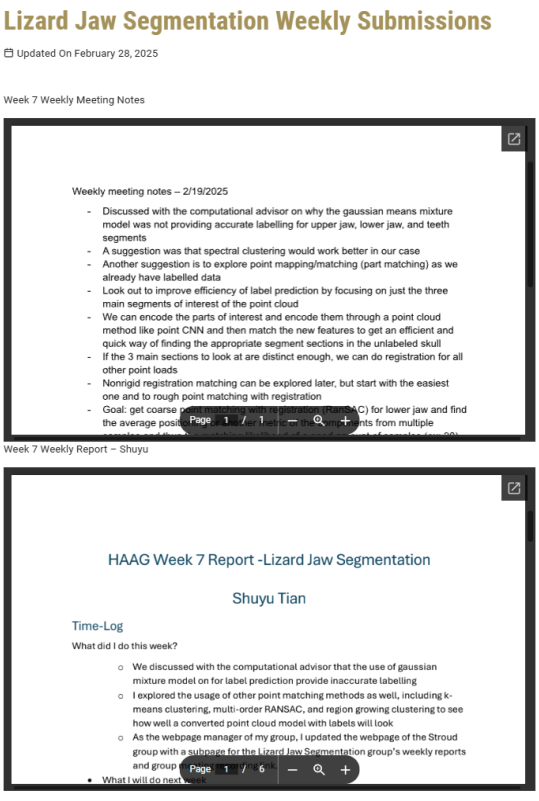
Weekly Lizard Jaw Meeting-20250129_150624-Meeting Recording.mp4

Spring 2025 Week 3 Meeting Recording

Weekly Lizard Jaw Meeting-20250122_150152-Meeting Recording.mp4

Spring 2025 Week 3 Computational Advisor Meeting Recording

Lizard Jaw Segmentation_ Student Reasearcher _ Computational Advisor Meet-20250120_150433-Meeting Recording.mp4



Lizard Jaw Segmentation Weekly Submissions

Updated On February 28, 2025

Week 7 Weekly Meeting Notes

Weekly meeting notes – 2/19/2025

- Discussed with the computational advisor on why the gaussian means mixture model was not providing accurate labelling for upper jaw, lower jaw, and teeth segments
- A suggestion was that spectral clustering would work better in our case
- Another suggestion is to explore point mapping/matching (part matching) as we already have labelled data
- Look out to improve efficiency of label prediction by focusing on just the three main segments of interest of the point cloud
- We can encode the parts of interest and encode them through a point cloud method like point CNN and then match the new features to get an efficient and quick way of finding the appropriate segment sections in the unlabeled skull
- If the 3 main sections to look at are distinct enough, we can do registration for all other point loads
- Nonrigid registration matching can be explored later, but start with the easiest one and to rough point matching with registration
- Goal: get coarse p[illegible] matching with registration (RanSAC) for lower jaw and find the average positi[illegible] the [illegible]ints from multiple

Page 1 / 1

Week 7 Weekly Report – Shuyu

HAAG Week 7 Report -Lizard Jaw Segmentation

Shuyu Tian

Time-Log

What did I do this week?

- We discussed with the computational advisor that the use of gaussian mixture model on for label prediction provide inaccurate labelling
- I explored the usage of other point matching methods as well, including k-means clustering, multi-order RANSAC, and region growing clustering to see how well a converted point cloud model with labels will look
- As the webpage manager of my group, I updated the webpage of the Stroud group with a subpage for the Lizard Jaw Segmentation group's weekly reports and group [illegible]

Page 1 / 6

- What I will do next [illegible]