Report on Naamii Bone Segmentation Task

Task 1.1 – Initial Bone Segmentation

Goal

Segment **femur** and **tibia** bones from the input CT scan using image processing techniques.

Approach

- Initial intuition: I have visualized 3d volumes using ITK-SNAP. Seems like binary segmentation task, there are no complex structures so Threshold might work

1. **HU Thresholding**:
    - CT Hounsfield Unit (HU) values for cortical bone typically range from **250 to 3000**.
    - Used SimpleITK.BinaryThreshold to isolate high-density bone regions.
2. **Connected Component Labeling**:
    - Applied ConnectedComponentImageFilter to label all bone blobs.
    - Ranked labels by size using LabelShapeStatisticsImageFilter.
3. **Label Assignment**:
    - Assigned:
        - **Largest component** as **femur (label = 1)**
        - **Second largest** as **tibia (label = 2)**
4. **Post-processing**:
    - Used BinaryFillhole to fill internal holes and clean up bone masks.
    - But no significant improvement on applying morphological hole filling.

Task 1.2 – Morphological Expansion

Goal

Create new masks by expanding each structure's contour by:

- **2 mm**
- **4 mm**

Approach

1. **Isolate Femur and Tibia Separately**
2. **Apply BinaryDilate**:
    - Used BinaryDilate with a radius vox to approximate 2 mm and 4 mm.
3. **Merge Expanded Masks**
    - Combined femur and tibia back into a single labeled mask.

Task 1.3 – Randomized Contour Adjustment

Goal: Create two randomized tibia/femur masks where:

- Shape is **between** original and 2 mm expanded masks.
- Simulates slight anatomical variability.

Approach

- Defined function get_randomized_label(original, expanded, label, ratio)
- Identified voxels in **expanded but not in original** (i.e., the 2 mm ring).
- Randomly selected a subset of those voxels (ratio=0.5) to simulate irregular growth.
- Added them back to original mask to get new label shapes.

Task 1.4 – Tibia Landmark Detection

Goal: Extract two key anatomical points from **tibia (label 2)**:

- **Medial lowest point**
- **Lateral lowest point**

Approach

1. Loaded mask and extracted all tibia voxels.
2. Found **lowest axial slice** (maximum z index).
3. On that slice:
    - **Medial point** = voxel with **minimum x**
    - **Lateral point** = voxel with **maximum x**
4. Converted voxel coordinates to **physical coordinates** using image origin, spacing, and direction.

Applied To:

- Original mask
- 2 mm expanded mask
- 4 mm expanded mask
- Randomized mask 1
- Randomized mask 2

Output

A list of (x, y, z) physical coordinates for both medial and lateral points per mask.

Original:
Medial = (np.float64(9.1552734375e-05), np.float64(127.76381599903107), np.float64(-470.5))
Lateral = (np.float64(197.29509460926056), np.float64(88.65247178077698), np.float64(-470.5))

Expanded 2mm:
Medial = (np.float64(-2.607331395149231), np.float64(126.8946750164032), np.float64(-470.5))
Lateral = (np.float64(199.90251755714417), np.float64(87.78333079814911), np.float64(-470.5))

Expanded 4mm:
Medial = (np.float64(-2.607331395149231), np.float64(126.8946750164032), np.float64(-470.5))
Lateral = (np.float64(199.90251755714417), np.float64(87.78333079814911), np.float64(-470.5))

Randomized 1:
Medial = (np.float64(-2.607331395149231), np.float64(126.8946750164032), np.float64(-470.5))
Lateral = (np.float64(199.90251755714417), np.float64(87.78333079814911), np.float64(-470.5))

Randomized 2:
Medial = (np.float64(-2.607331395149231), np.float64(127.76381599903107), np.float64(-470.5))
Lateral = (np.float64(199.0333765745163), np.float64(90.39075374603271), np.float64(-470.5))