Introduction

Markerless registration is the process of aligning 3D data **without using physical tracking markers or fiducials**.A markerless registration pipeline is developed for aligning **RGB-D data from an Intel RealSense camera** with **CT-derived anatomical structures**, specifically focusing on the **spine**. The goal is to extract accurate 3D surfaces from both modalities and register them using only intrinsic features and geometry,without any external markers or trackers.

Dataset Collection and Annotation

Image Acquisition

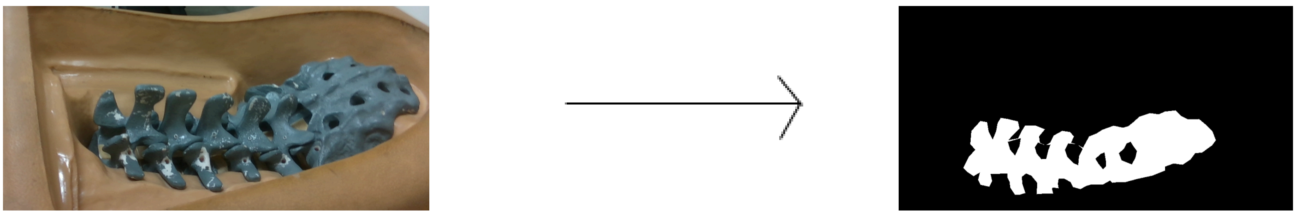
- **RGB Images:** 53 images captured using **Intel RealSense**
- **Testing Data:** 5 images captured using **Intel RealSense**
- **Depth Data:** Depth available for 5 images (stored as `.npy` arrays)

Annotation

- **Tool Used:** Labelme
- **Annotated Images:** 53 (trained images) + 5 (testing images)
- **Classes:**
    - `0`: Background
    - 1: Bone

Mask Generation

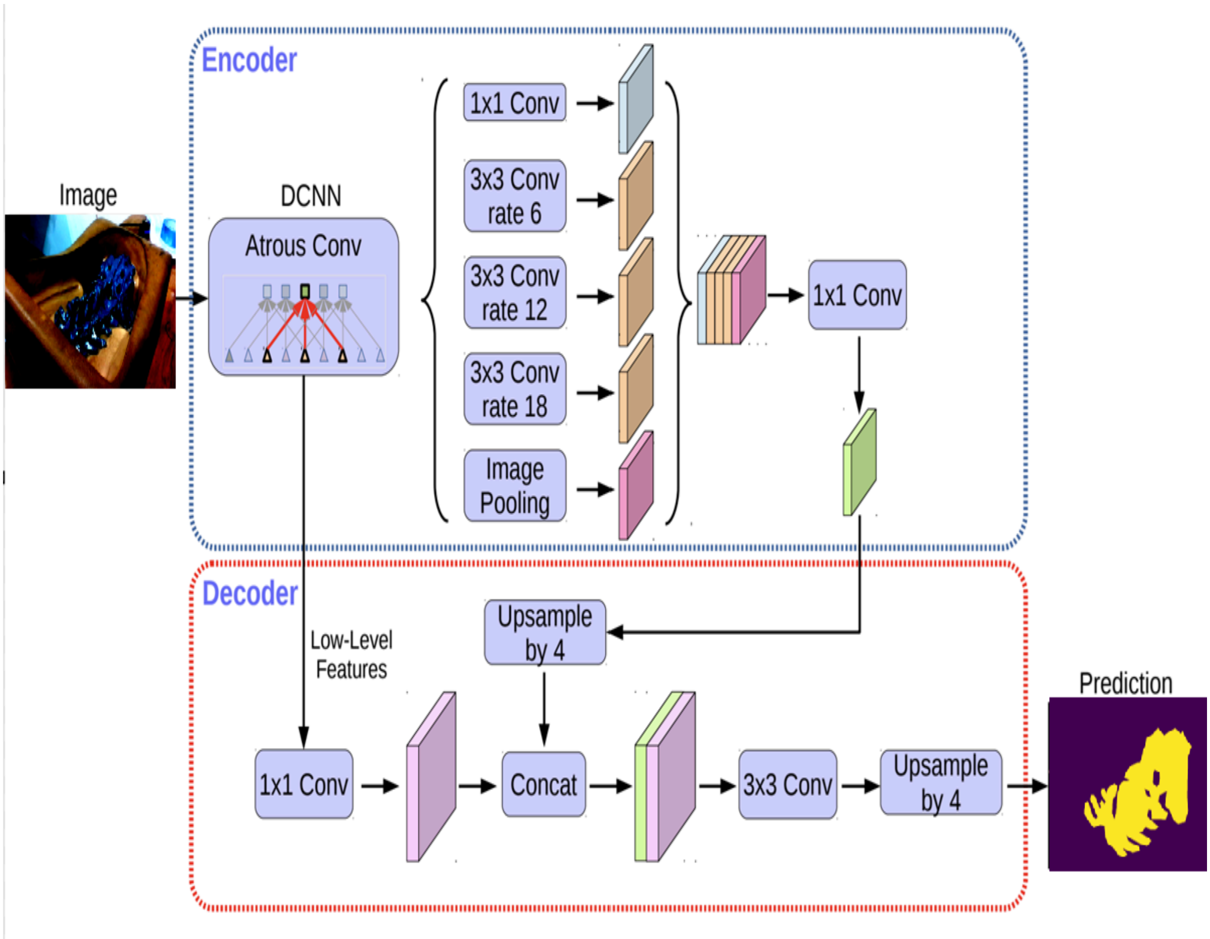
- Polygon annotations from Labelme were converted into binary masks.
- Workflow:
  Input RGB → Manual Polygon Annotation → Binary Mask Generation



Model Training – DeepLabV3+

Architecture

- **Model:** DeepLabV3+
- **Backbone:** ResNet-50
- **Input Resolution:** 1280 x 720
- **Output:** Binary segmentation mask (bone vs background)



Why DeepLabV3+?

I compared multiple architectures during my research:

Architecture	Notes
Gated CNN	Suitable for time series (gated units help with temporal data), but **not ideal for static 2D anatomical structures** like bones
W13 Net	A dual-UNet approach which produced good results, but consumed **~9,000 MB GPU memory**, making it impractical on limited hardware
DeepLabV3+	Chosen for its **Atrous Spatial Pyramid Pooling (ASPP)** module, giving excellent multi-scale feature extraction and boundary precision. Achieved **high IoU and pixel accuracy** during training

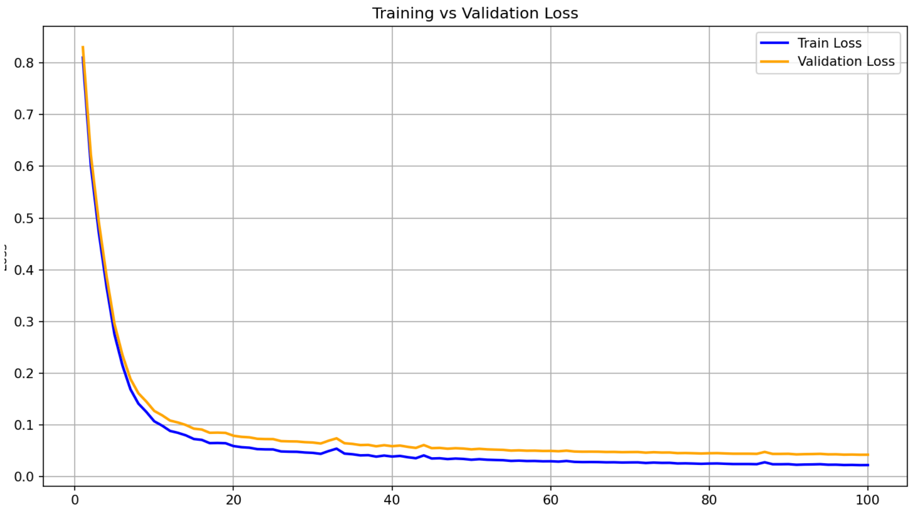
Training Setup

- **Framework:** PyTorch
- **Loss Function:** CrossEntropyLoss
- **Optimizer:** Adam
- **Epochs:** 100
- **Dataset Size:** 53 images

Evaluation Metrics

- **IoU (Intersection over Union):** 86.5%

- **Pixel Accuracy:** 96%



These metrics confirm the model's effectiveness at segmenting the bone structures accurately.

RGB-D Based 3D Surface Extraction

Input

- RGB image (with image size 1280x720)+ corresponding `.npy` depth image from RealSense

Segmentation & Backprojection

- Trained **DeepLabV3+** was used to infer the mask from RGB image.
- The masked region was used to filter valid depth pixels.

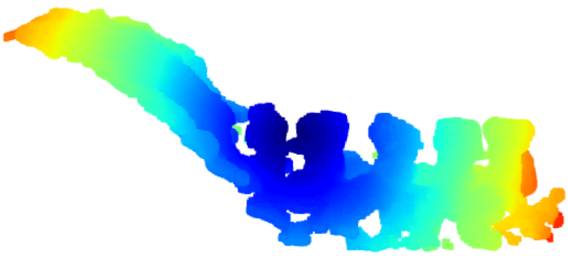
Each valid `(u, v, z)` point was backprojected into 3D using camera intrinsics:

```
X = (u - cx) * z / fx
Y = (v - cy) * z / fy
Z = z
```

Point Cloud Generation

- The filtered 3D points were assembled into a **point cloud** using `Open3D`.

- Saved as `.ply` files for visualization and further processing.



3D Point Extraction from CT Data

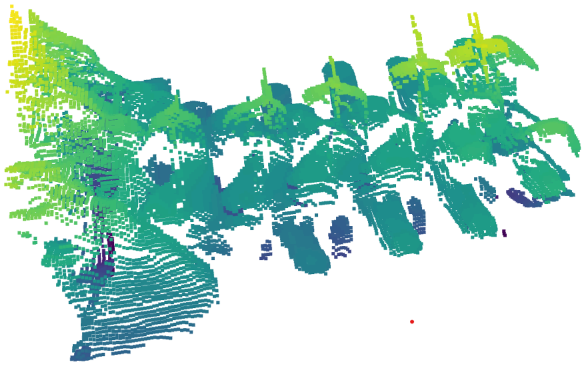
- **Input:** OpenSawbone CT in **DICOM** format

- Converted to `.vtk` format for use with VTK and `Open3D`

- Encountered **scaling issues** due to voxel spacing differences

Resolved by applying:
`volume_actor.SetScale(spacing)`

- to ensure proper proportions along all axes
- Used a scalar threshold (i.e, `-320 to 395`) to isolate bone density and extract the surface as a **point cloud**



3D Point Extraction Using RealSense

- Captured **RGB + Depth** image from RealSense
- Applied the **trained DeepLabV3+** model on the RGB image
- Used the mask to filter valid depth pixels
- Extracted **surface-only 3D point cloud** from RealSense depth and saved as `.ply`

Markerless Registration using Geometry

- Both point clouds (RealSense and CT) were aligned using a two-step markerless registration approach:
  - RANSAC with FPFH features for rough global alignment
  - ICP (Iterative Closest Point) for precise local refinement
- No physical markers or tracking devices were used