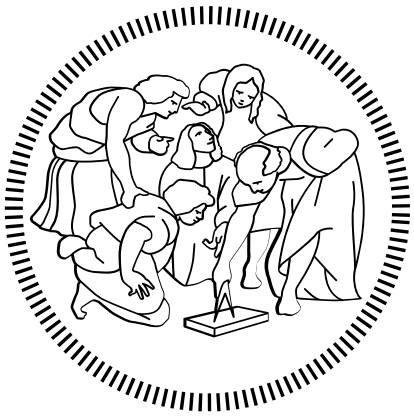


POLITECNICO
MILANO 1863

School of Industrial and Information Engineering
Master of Science in Biomedical Engineering

Deep Learning-Based 3D Facial Landmark Localization: A Two-Step Approach with CNN and T-Net

Supervisor: Prof. Giuseppe Baselli
Dott.ssa. Annalisa Cappella

Co-supervisor: Ing. Benedetta Baldini

Master Thesis of: Ali Shadman Yazdi (Matr. 991655)

Academic year 2023/2024

Acknowledgement

I would like to thank prof. Baselli for giving me the opportunity to work on this project and guiding me through the whole process. His immense experience was an irreplaceable asset, you could tell even the slightest suggestion was sourced from a wealth of knowledge.

Deep appreciation to prof. C. Sforza and Dott.ssa. Annalisa Cappella for providing me the dataset and providing continous feed back along the way. Most of my work seems to have turned into a tribute to their expertise, as I find myself citing their contributions at every turn. Additionally, my gratitude towards Ricardo, which I have not met many times, but without him manually annotating every single landmark by hand, this project would not have been possible.

This thesis would not be what it is without Benedeta, I could not thank her enough, she was truly a guiding light while putting this together, a real blessing.

Genuine thanks to Micheal who was basically my human Grammarly throughout the writing process of my thesis. Big shout-out to Sia, who I would have in depth conversations about technicalities of our theses and give insights on how to make it better and advanced.

Special thanks to my beloved ragazza, Miss Jiaying, whose annoying persistence (I mean, gentle motivation) propelled me through the tough times of this thesis, pushing me to be the best version of myself. You're annoyingly amazing!

Last but certainly not least, I am beyond grateful for my parents. Their unwavering support in every aspect of my life has been an invaluable asset. Words cannot express the depth of my gratitude for their love, guidance, and belief in me.

Table of Contents

List of Figures

List of Tables

List of Appendices

Abstract

This thesis introduces an automated 3D facial landmark annotation system using deep learning, specifically Convolutional Neural Networks (CNNs). Trained on a dataset of stereophotogrammetry-acquired facial models, the model showcases potential applications in orthodontics, maxillofacial, and aesthetic surgery. By reducing manual annotation time, it enhances efficiency and precision in 3D anthropometric analysis, marking a significant stride in automating critical anatomical landmark identification. Furthermore, this study delves into stereophotogrammetry technology, highlighting its relevance in diverse fields. Despite requiring meticulous calibration, its advantages include minimal acquisition time and high accuracy. The conversion of 3D facial models into point cloud data facilitates versatile operations, with further exploration of point cloud processing methods, including PointNet. Emphasizing the crucial role of data pre-processing, ensuring a standardized and well-prepared dataset for effective model training. A two-stage deep learning approach, incorporating CNNs and T-Net architecture, achieves millimeter precision in facial landmark localization. The models undergo thorough analysis, considering variations in preprocessing techniques and their impact on performance. The Refined EsTi model emerges as robust and consistent, showcasing superior performance with adjusted metrics. These findings provide valuable insights for future developments in automated 3D facial landmark localization. The proposed methodology's versatility allows for comprehensive evaluations on diverse facial databases and landmarks, offering potential applications beyond the initial scope. This research represents a significant contribution to the field, demonstrating the potential of deep learning in automating critical anatomical landmark identification for medical and research purposes.

1. Introduction

1.1. Aim of the thesis

The objective of this thesis is to provide orthodontists, maxillofacial and aesthetic surgeons with a completely automated tool for annotating landmarks on three-dimensional (3D) facial models. Specifically, the project involves constructing a deep learning (DL) model designed to accurately localize 50 pre-defined anatomical points on facial 3D models acquired through stereophotogrammetry imaging.

Facial landmark annotation is a task that consists of identifying and localizing reference points or landmarks on 2D, or 3D face utilizing them for applications in different domains. In the medical field, facial landmarking is essential for 3D anthropometric analysis, that refers to the examination and measurement of human face dimensions and proportions. Accurate 3D anthropometric analysis of the face plays a crucial role in biological and clinical applications, including the identification of age, sex, and ethnic group [1], the study of facial changes resulting from various treatments and surgeries such as orthognathic surgery [2], [3], plastic surgery planning and evaluation [4]. Furthermore, it proves invaluable insights in early diagnoses of different pathologies, such as Marfan and Aicardi syndrome [5], [6] in which traditional diagnostic methods often necessitate patients to reach adulthood before accurate diagnosis, leading to delays in treatment [6]. Anthropometric studies of the face have observed characteristic features and their change over time for number of dysmorphic syndromes, including Down syndrome [7], Rubinstein–Taybi syndrome [8], and Sotos syndrome [9]. Moreover, facial landmarks can enhance the precision of superimposing two faces for various tasks, such as comparing changes before and after a treatment [10]. Facial landmarks can also be employed to investigate facial asymmetry[11], and normal or abnormal growth patterns [12].

Traditionally, landmarks annotation for anthropometric analysis was performed by direct annotation of points on the patient's face and use of instruments like calipers and protractors to measure distances and angles between them. In modern applications, the advent of 3D surface imaging systems, like stereophotogrammetry, enables the creation of accurate models of facial soft tissues, on which landmarks are digitalized by experienced operators. These noninvasive techniques help to overcome certain limitations of direct anthropometry, namely the risk of motion artifact due to patient movement and the risk of soft-tissue deformation, due to contact between instrument and the skin. Despite these advancements, the landmarking procedure remains tedious and time consuming, and dependent on the operator expertise [13]. Therefore, automating the landmark annotation process in anthropometric analysis could offer a broad spectrum of benefits that collectively enhance the efficiency, precision, and applicability of the analysis in research and clinical settings.

In this perspective, a dataset comprising of 200 3D facial models acquired through stereophotogrammetry was used. Each 3D facial image was manually annotated by an expert operator following a protocol developed by Ferrario et al. [14] to describe the entire face and its portions. This protocol includes 50 anatomical landmarks, positioned on the forehead, eyes, nose,

lips, mouth, chin, ears, and lateral facial surface. Specifically, 12 landmarks are on the midline, and 19 landmarks are on each side of the face. Then, landmarks are exported as 3D coordinates for successive data elaboration. The annotated dataset is used to train a deep learning model, specifically designed to localize the 50 landmarks on 3D facial models, achieving identification accuracy comparable to skilled practitioners and within clinically acceptable range.

The deep learning method employed in this work is the Pointwise Convolutional Neural Networks (CNNs), which have demonstrated efficiency in tasks such as segmentation, object recognition, and classification [15] [16]. Pointwise CNNs, while simpler compared to other 3D processing techniques like PointNet [17] which implements point feature leaning by fully connected layers [15], have yielded competitive results when compared to these methods. Notably, no study has been conducted on localizing landmarks on 3D facial data using Pointwise CNNs. This study aims to explore the application of Pointwise CNNs to 3D facial data for automatic landmark localization and to compare its results with the state-of-the-art techniques.

In the upcoming chapters, the content will begin with an introductory overview of stereophotogrammetry imaging technology. This will be followed by a comprehensive discussion on deep learning, both in a general sense and within the specific context of the study. The subsequent sections will delve into a detailed description of the architecture developed for the model. To assess its effectiveness, the model's performance will be thoroughly evaluated, with a focus on the identification error between manually and automatically identified landmarks. This evaluation aims to provide insights into the accuracy and reliability of the developed model in landmark localization.

In conclusion, this work represents a comprehensive approach to automatically identifying clinically relevant anatomical landmarks on 3D facial images obtained through stereophotogrammetry. The tool developed in this work could support clinicians in the landmark's annotation task, providing numerous advantages. Firstly, it enhances efficiency by significantly reducing the time required for annotation compared to manual methods. Then, automation ensures a consistent and standardized approach, minimizing dependence on operator expertise. Finally, the developed model has the potential to be adapted and integrated with other technologies, allowing for more sophisticated and advanced analysis.

1.2. Stereophotogrammetry

Stereophotogrammetry image acquisition is a technique used to capture 3D information about objects or scenes through analysis of multiple bidimensional images. It mimics the human eye's depth perception by capturing two or more images of the same subject from different viewpoints. The captured images then will go through the triangulation process to extract spatial information and create a 3D representation of the scene [18].

Currently, stereophotogrammetry stands out as a promising noninvasive method for facial anthropometric analysis, leveraging detailed 3D reconstruction of facial image and overcoming limitations of direct anthropometry for soft tissue analysis [19]. By employing multiple cameras, stereophotogrammetry can swiftly capture a subject's 3D image, mitigating the impact of subject

movement and avoiding direct contact with the face, which could otherwise causes surface deformations and affects measurement outcomes [20] [1]. The images obtained through stereophotogrammetry has the reported accuracies ranging between 0.5 and 1mm [21]. Stereophotogrammetry is considered the gold standard among the methods for reconstructing the 3D surface of the face. This technology holds significant potential for enhancing precision and reliability in facial assessments within clinical settings. Primary issues include the instrument's high costs and limited portability of the instrument, which could be addressed by employing portable instruments already validated for similar accuracy levels. Moreover, newer and simpler techniques are under investigation.

The complexities arising from the transparency, reflectivity, and intricate nature of hair, as well as the specular reflections and distortions caused by wet surfaces can lead to potential artifacts in the mesh and the final 3D image outcome [22]. In many cases, pins and hairbands are employed to move any stray hairs away from the face, creating a finer mesh around it. However, this process must be executed carefully to avoid skin stretching, which could alter the facial surface [23].

The instrument used in this work is the Vectra M3 (Canfield Scientific Inc., Fairfield, NJ,USA) 3D imaging system, a modular 3D image acquisition system designed for stereo imaging. To maintain precision, the system requires daily calibration and or calibration in case of camera displacement [1]. The Vectra 3D imaging system is comprised of three groups of cameras, two black and white and one-color cameras, and a projector. A safe laser light is projected onto the face to accentuate facial features and depth differences. Each camera captures a 2D image of the subject within a rapid 0.75-millisecond timeframe [19]. Through dedicated software and the triangulation process, the system reconstructs a 3D model of the subject, facilitating various anthropometric analyses.

1.3. Deep learning in stereophotogrammetry

Recently, the study of G. de Jong et al. [24], aimed to combine 3D stereophotogrammetry with DL algorithms to investigate the capability of accurately classifying head shape of infants on 3D stereophotographs. This classification includes distinguishing between healthy infants and those with craniosynostosis, as well as identifying the specific subtype, such as scaphocephaly, trigonocephaly, or anterior plagiocephaly. Craniosynostosis is characterized by the premature fusion of one or more cranial structures, resulting in cranial deformation and leading to facial asymmetry [25]. For this study a conventional feed forward neural network was employed, compromising hidden layers with 192, 128, 64 and 32 nodes respectively. The activation function used within the hidden layers was Leaky Rectified Linear Unit (Leaky ReLU). Regularization techniques included Dropout with a rate of 0.5, batch normalization, and Gaussian noise applied to the training set. The output layer featured a Softmax activation function with 4 nodes corresponding to the 4 classes. Training was conducted using the Adam optimizer with a learning rate of $1 * 10^{-3}$ and a gradient normalization of 0.001. A batch size of 256 samples was utilized, and training extended over 1000 epochs. Categorical cross-entropy served as the evaluation metric and early stopping criterion. To partition the data into training and test sets, stratified tenfold cross-validation was employed. Each subject's original 3D stereophotograph and its mirrored image were retained within the same set. In this study, 196 subjects were included, and 195 of them were correctly classified, resulting in an overall accuracy of 99.5%. While it is known that deep learning

models generally perform better with larger datasets [26], the rarity of the disease (3.14 to 6 per 10,000 live births) [27] posed challenges in obtaining a larger dataset. Working with a small dataset can lead to overfitting, and the results achieved with limited data such as in this study may be considered as suboptimal results.

1.4. State of the art on automatic landmark localization

Various methods and algorithms have been employed for landmark localization on 2D facial images for different purposes [28] [29] [30]. However, due to limitations in the availability of 3D facial images and overall 3D data, fewer methods have been explored in this domain. Several approaches have been investigated for 3D facial landmark localization, encompassing machine learning (ML) methods where features are manually extracted from the data, and different models are trained to predict the final landmarks. Deep learning (DL) techniques have also been extensively explored. For instance, R. R. Paulsen et al. [31] processed 3D data into 2D images and applied 2D deep learning methods, localizing landmarks by predicting on 2D images from various views. Another approach involves converting 3D data into volumetric data and processing it, although this method faces limitations in terms of substantial memory requirements [15] [32]. Various 3D processing techniques, such as the use of PointNet [17], and Convolutional Pose Machines [33], have been applied to 3D data for landmark localization.

In the study of C. Creusot et al. [34], a ML technique is employed to annotate 14 facial landmarks on 3D facial models (meshes). In this process, manual feature extraction is conducted, wherein scalar-valued and vector-valued features are extracted from the meshes. These features include vector normal, information extracted from neighboring points on the mesh withing a predefined threshold, local volume, principal curvatures, and other features derived from the principal curvatures such as Gaussian curvature, mean curvature, curvedness, and other similar features. The model has an offline training process and online landmark detection. During the offline training, the position of the landmark is known, and they are used to learn the statistical distribution of the landmarks. In the online process, the mentioned features are extracted from the meshes and with using the statistical distribution model made from the offline training, a score map is generated for each vertex. The strongest local maxima over a certain threshold in the score map indicate a predicted landmark. The prediction error varies between 2.5 to 10 mm, measuring the difference between the predicted landmark and the ground truth.

In the study of R. R. Paulsen et al. [31], a multi-view approach is proposed to identify feature points on facial surfaces. It involves rendering the 3D facial model from multiple views, generating multiple 2D images of the face. The 3D facial model is placed at the origin of the coordinate system, and the camera is placed in 100 random positions around the face, with their focal point at the origin. For each of the rendered 2D images, the 2D coordinated of the projected ground-truth 3D landmarks are stored. The 2D ground truths are recorded as one heatmap per landmark, with a gaussian peak located at the 2D position of the landmark. The 2D images are processed by an hourglass CNN, as detailed in [35]. The model's input has a flexible number of layers, with three layers added for RGB texture rendering. If depth rendering, geometry rendering, and/or curvature rendering are employed, each adds one layer. The input images are 256x256 pixels, and the model's output consists of one heatmap per landmark, each with a size of 256x256 pixels, same as the size

of the input image. During training, the model used an initial learning rate of 0.00025, with a decay rate of 0.96. The batch size ranged from 4 to 8, and dropout with a rate of 0.02 was employed as a regularization technique. Convergence was typically achieved around 60-100 epochs, depending on the specific rendering input utilized. After the prediction is performed on the 2D rendered images, 3D view rays are generated using the 2D predictions. For each landmark, there are 100 rays in 3D space, corresponding to the 100 2D predictions from different views for that landmark. The point where all these rays intersect in space represents the 3D coordinates of the predicted landmark. However, due to errors in the 2D predictions, there may be some rays that never converge to a single point. To address this issue, least squares fit combined with RANSAC selection was employed, as described in [36]. The evaluation criteria employed is the Euclidean distance between the actual landmark's 3D coordinates and its predicted counterpart. The model underwent training and evaluation on the BU-3DFE database, comprising 83 facial landmarks. Validation was conducted on 500 images, resulting in a mean Euclidean distance error of 2.42mm. This study shows the feasibility of achieving good results in facial landmark localization through using multi-view rendering techniques.

E. O' Sullivan and S. Zafeiriou [33] proposed extending 'Convolutional Pose Machines' (CPMs) for facial landmark localization, leveraging the architecture known for achieving state-of-the-art results in face and body landmark localization in 2D images [37] [38]. CPMs inherit the pose machine architecture described in [39], localizing landmarks using a sequence of convolutional networks to sequentially refine a series of heatmaps. Refinement is performed at each stage to achieve more precise localization and the heatmap produced at each stage becomes the input for the subsequent stage. The initial heatmaps are generated by applying a Gaussian peak at the ground truth, with the peak representing the landmark position. This results in having a 1D vector heatmap for each of the landmarks, where each element of the vector is associated with one of the vertices in the mesh. To extend the use of CPM to 3D facial models, PointNet++ architecture [17] is used as the feature extraction block. Three stages were used for the heatmap refinement. After the refinement stages on the heatmaps, the final landmark prediction is calculated by selecting the three highest (hottest) points in each heatmap. The barycenter of these three vertices is then determined as the final position of the landmark. This three-point strategy is implemented to mitigate outliers and ensure a more robust result. The loss function for training involves minimizing the sum of the mean squared error of the output heatmaps at each of the three refinement stages. The model is trained using the Pytorch library with a batch size of 8. Training is done using Adam optimization with an initial learning rate of 0.001. As a form of data augmentation, all meshes are normalized and randomly rotated around the XY and Z axes, with their corresponding landmarks. The training and evaluation were conducted on the BU-3DFE database, with 80 out of 100 subjects randomly assigned to the training set and the remaining 20 to the test set. The training and evaluation process was repeated five times, each time with a different test set to ensure that each subject was assigned to the test set at least once. The achieved results were compared with those from [40] [41] [42]. This study demonstrates the feasibility of extending 2D methods for facial and body landmark localization to 3D facial landmark localization.

2. Methods

In this chapter, the underlying physical principles of stereophotogrammetry imaging technology are introduced. The subsequent section outlines the primary characteristics of the 3D model and the methodological steps for preprocessing and analysis. Namely, it is stressed the nature of stereo scans, and point cloud sampling the face surface in the 3D space, which presents specific representation and processing requirements, different from 2D or 3D images sampled on regular grids.

Next, methods involving artificial intelligence (AI) and neural networks (NN) are presented. Basic principles are recalled and finalized to the processing of point clouds. Namely, ICP algorithm for aligning the point clouds in a common space, resampling rates, and various methods utilized for indexing the points of the point clouds, for further analysis.

2.1. Stereophotogrammetry Principle

Objective assessment of facial morphology is important for research and clinical application in various fields involving head and face like orthodontics, genetics, aesthetical surgery, dysmorphology, but also forensic sciences and industrial fields [43]. Currently, classic direct anthropometry is being replaced by modern digital techniques for soft-tissue facial imaging and analysis. There are two main methods of computerized, non-invasive, soft-tissue 3D facial anthropometry: optical non-contact instruments such as stereophotogrammetry and contact instruments such as ultrasounds probes [21]. The latter method involves digitizing specific facial landmarks for 3D reconstruction of fetal faces. Concerning non-contact methods, optical technique such as laser scanners and stereophotogrammetric systems are employed [13] Laser scanners use a laser light source to illuminate the face, digital cameras to capture the reflected lights, and triangulation geometry algorithms to reconstruct the 3D depth information. The image is acquired within 10 seconds, with reported accuracy ranging between 0.5 and 1mm. The ears, nostrils and the chin are the critical parts of the face. Shadows and facial characteristics such as hairs and moles can introduce additional complexity to the analysis. Moreover, high acquisition times could result in motion artifacts. This is particularly crucial when the image acquisition is done for children and disabled persons.

Stereophotogrammetry image acquisition represents the most promising approach for soft tissue analysis. It utilizes safe lights to illuminate the face, two or more coordinated digital cameras to capture the reflected light from different points of view and exploits the stereoscopic principle to reconstruct the 3D image [44] [45] [21]. The stereoscopic principle is the ability of a visual system to perceive depth and 3D information by capturing slightly different images, with the use of multiple cameras or image sensors. This technique simulates human binocular vision, which is the slight difference in the perspective of the same scene when viewed from each eye.

The time required for stereophotogrammetry image acquisition is insignificant, in the order of 2 ms, minimizing the likelihood of motion artifacts in the acquired images. Using stereophotogrammetry it becomes feasible to capture facial texture and combine it with the stereoscopic reconstruction to have an accurate reproduction of all facial characteristics with an

accuracy between 0.5 and 1mm [21]. However, a drawback with stereophotogrammetry image acquisition is the need for instrument calibration with an object with known geometric characteristics, which must be performed each time the instrument is switched on. Moreover, depending on the computer's specifications, the time required for the post-processing step could be considerable. This is because each facial image acquired with the digital cameras must undergo mathematical combination to obtain the 3D surface [21]. Another limitation associated with stereophotogrammetry image acquisition is the cost of the instrumentation and the fixed setup, making it challenging to meet certain conditions for patients in different locations. To address this limitation, portable stereophotogrammetric instruments have been developed and are utilized outside the laboratory [18], their reported accuracy is around 1mm and considered sufficient for basic and clinical studies, even if motion artifacts may potentially limit their use [46].

Figure 2.1. represents three modalities of the reconstruction process:

Figure 2.1. *Three Stages of Reconstructing 3D Facial Stereophotographs with Facial Texture. A) Mesh Reconstruction, B) Surface Rendering, C) Final Facial Reconstruction*

(A) (B) (C)

(A) Facial reconstruction with mesh rendering, where the quality of the reconstruction improves with finer mesh. Some artifacts, particularly noticeable on the eyebrows and eyes, arise from low-resolution scans, as previously discussed, these artifacts result from inadequate reflection of light due to facial hair or sweat.

(B) Surface rendering of a 3D facial reconstruction made by stereophotogrammetry.

(C) Final facial reconstruction with the relevant facial texture, providing an accurate reproduction of all facial characteristics.

2.2. Point Clouds

A point cloud is a collection of data points in a 3D coordinate system that represents the external surface of a physical object, typically obtained from 3D scanners or photogrammetry software. Each point denotes a specific position in space and is defined by its set of Cartesian coordinates (X, Y, Z). Additionally, other information such as color, intensity, and vector normals can be associated with each point, which will be discussed in later sections. Point clouds are utilized in

various computer vision tasks such as robotics, geomatics, and 3D modeling, providing detailed representations of object geometry. The 3D facial models acquired from stereophotogrammetry can be converted into point cloud data by extracting the vertices of the mesh without any loss of information. This conversion allows for the application of various operations and preprocessing algorithms, leveraging the simplicity of point clouds. Figure 2.2. demonstrated the same 3D facial model as figure 2.1. converted to point cloud with 202,395 vertices. The point cloud representation also includes the artifacts caused during the mesh construction, due to the presence of facial hair such as eyebrows and eye lashes which cannot reflect light, due to their translucent nature.

***Figure 2.2**. 3D Facial Model Represented as Point Cloud.*

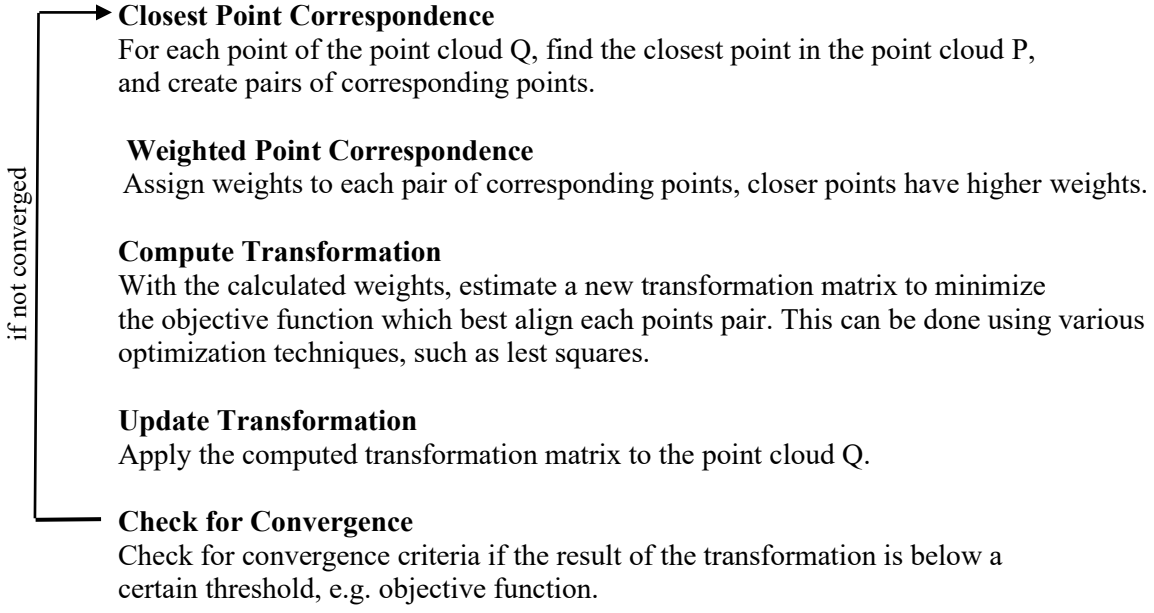
2.2.1. Iterative Closest Point (ICP) Registration

Iterative Closest Point (ICP) is an algorithm utilized in various computer vision applications to align two or more 2D or 3D point clouds. This iterative optimization technique seeks to determine the optimal rigid transformation (transformation matrix) that minimizes the disparity between corresponding points in the point cloud [47]. The optimization process involves minimizing an error metric, the objective function, with Euclidean distance or sum of squared distances commonly used for this purpose. The step-by-step process of the ICP registration algorithm is outlined below:

Given two-point clouds, the target point cloud P and the source point cloud Q, with corresponding points $p_i \in P$, $q_i \in Q$:

Initial guess for the transformation
Typically, the identity matrix for rotation matrix (R) and a zero vector for translation (t)

Apply the transformation to the source point cloud Q.
Multiply the translation matrix with the points qi of the point cloud Q.



Final transformation
The final output transformation matrix T_{final}, is the product of all the transformation matrixes applied during the iterations. $T_{final} = \prod_{i=1}^{N}(R_i, t_i)$

ICP registration is a straightforward algorithm employed for aligning point clouds. While its implementation is not inherently complex, the algorithm's performance is highly dependent on a good initial alignment. If the point clouds are initially positioned far apart and are not well-aligned, the algorithm may converge to local minima. Additionally, computational demands can be significant, especially for point clouds with a high number of points, as the algorithm calculates distances point by point. Sensitivity to outliers is another consideration, as outliers lacking a true corresponding point in the other point cloud can significantly impact alignment results. This sensitivity may affect the objective function and lead to an incorrect transformation. The algorithm may converge to suboptimal transformations in cases where there are partial overlaps, meaning not all points in the source point cloud have corresponding points in the target point cloud. ICP registration is a rigid registration method, implying that the transformation involves only rotation and translation of the source point cloud to align it with the target point cloud without introducing any deformation. Despite its simplicity, ICP registration has proven useful in numerous rigid registration tasks, and many variations of ICP registration has been introduced to tackle the mentioned limitations [48] [49].

2.2.2. Morton Code

The Morton code, also known as a Z-order curve or Z-order indexing, is a technique applied to multi-dimensional data to transform it into a linear, one-dimensional representation. This method is commonly used in computer graphics and tasks involving point cloud processing. In the context of point clouds, where each point is represented by its coordinates (X, Y, Z), Morton code interleaves these coordinates into a single value, creating a one-dimensional representation for each point. When applied to coordinates, the Morton code defines a space-filling curve that takes

on a Z-shaped pattern, hence the names Z-order or Z-curve. Morton code achieves this by converting each coordinate into binary representation and interleaving their bits. The following provides a more detailed, step-by-step description and an example of Morton code encoding for a 3D coordinate [50] [51]:

Let us consider a point with coordinates (X=4, Y=2, Z=6)

1. **Convert to Binary:**
   (X= 100, Y=010, Z=110)

2. **Interleave Bits:**
   This is usually done by taking alternate bits from each coordinate.
   Interleaved bits = 101110000

3. **Convert to Decimal:**
   Morton code = 368

The Morton code process is reversible, allowing the retrieval of the original spatial coordinates from the Morton code of a point in a point cloud. This reversibility is achieved by converting the Morton code from decimal to binary, deinterleaving the bits back into their original coordinates (X, Y, Z), and finally converting the binary representation of X, Y, and Z back to decimal. Through these steps, the original spatial coordinates of the point in the point cloud can be accurately reconstructed from its Morton code.

Morton code sorting is a technique used to efficiently sort data points based on their Morton codes. Encoding point cloud data points using Morton codes has the advantage of preserving spatial locality in the one-dimensional space. This means that points that are numerically close to each other in their Morton code are also close to each other in their original multidimensional space. This feature proves advantageous for various tasks, including nearest neighbor searches, spatial indexing, and various computer vision applications [15]. To apply Morton code sorting to a point cloud, all the points in the cloud are transformed into Morton codes, and these codes are then sorted based on their values. Subsequently, the sorted Morton codes are transformed back into their original 3D coordinates, resulting in a point cloud with the same points but a different indexed order. Resulting in, points that are closer to each other in space are indexed close to each other after the sorting process.

2.2.3. Sorting Based on a Reference Point

Another proposed method for indexing point clouds involves sorting them based on their distance to a fixed reference point. This is achieved by calculating the distance of each point in the point cloud from the reference point and then sorting the points based on their numerical distance. The resulting indexed array arranges points in ascending order of their distances from the fixed reference point, with the point closest to the reference point appearing first and those farther away

following subsequently. Similar to Morton code sorting, this method proves advantageous for various computer vision tasks, including point cloud processing [15].

2.3. Artificial Neural Networks

In the realm of Artificial Intelligence (AI) and computer science, Artificial Neural Networks (ANN) are computational models inspired by the functioning of the human brain [52]. They represent a subset of ML involving the training of models with a large amount of data to perform tasks without explicit programming. More specifically, ANNs constitute a subgroup of DL, an advanced category within ML characterized by employing multiple layers of processing steps. One powerful capability of ANNs is their ability to automatically perform feature extraction. During this process, the model learns to identify and extract features from raw input data, such as images, text, time series, 3D volumes, and, in our case, 3D stereophotographs. ANNs consist of layers of interconnected nodes, also known as neurons. These layers are organized into three main types: the Input Layer, Hidden Layers, and the Output Layer, as shown in figure 2.3. [52]. The input layer receives the initial raw data, while the layers between the input and output layers are called the hidden layers. Each node in the hidden layer performs a mathematical operation and then passes the result to the next layer. The output layer produces the final output of the neural network based on the computations performed in the hidden layers. The number of nodes in the output layer depends on the task for which the network is designed (e.g., classification, regression). Figure 2.3. demonstrates a simple neural network with an input layer with 3 nodes, a hidden layer with 2 nodes, and an output layer with 1 node [52].

In artificial neural networks, neurons from one layer to another layer are connected using associated weights, which represent the strength of the connections between the neurons. During training, the values of these weights are adjusted so that the difference between predicted and actual outputs is minimized. There are two different approaches to training models:

- In supervised learning, the model is trained on a labeled dataset, where each input is associated with a corresponding desired output;
- In unsupervised learning, the model is trained with unlabeled dataset. The main goal of unsupervised learning is to extract patterns, structure, or meaningful representations from the data without the aid of labels.

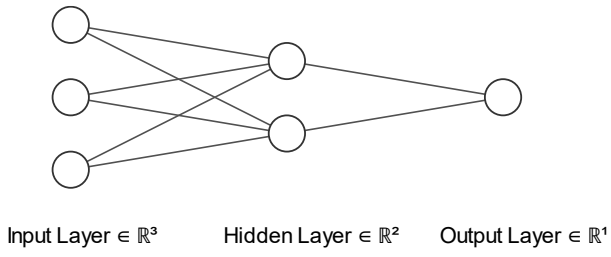


***Figure 2.3**. A simple neural network architecture*

The output of a neuron is calculated using the formula:

Equation 2.1.

$$Output = Activation\left(\sum_{i=1}^{n} Weight_i \times Input_i + Bias\right)$$

- $Output$ is the output of the neuron.
- $Activation$ is the activation function applied to the weighted sum.
- n is the number of inputs to the neuron.
- $Weight_i$ is the weight associated with the $i-th$ input.
- $Input_i$ is the $i-th$ input to the neuron.
- $Bias$ is the bias term.

The bias term is an additional parameter used to shift the output. It acts as an offset, allowing neurons to fire even when the sum of the products of inputs and their corresponding weights is not sufficient on its own. Similar to neuron weights, biases are adjusted during the training of neural networks, and mathematically, they are added before applying the activation function. The bias allows the network to model situations where an input feature may have small influence, but the neuron should still activate.

2.3.1. Activation function

Activation functions are crucial components of ANNs; they introduce non-linearity into the network, enabling it to learn complex patterns and relationships in the data. Rectified Linear Unit (ReLU) is one of the most commonly used activation functions in hidden layers. The formula for the ReLU activation function can be seen in Equation 2.2, where the output of the function is equal to the input if it's positive; otherwise, the function outputs equals zero [53]. Other activation functions include the Sigmoid Activation, which outputs values between 0 and 1; Hyperbolic Tangent (tanh) Activation, whose output ranges from -1 to 1; and Leaky Rectified Linear Unit (Leaky ReLU), designed to address the 'dying ReLU' problem by introducing a small, non-zero slope for negative inputs [54].

Equation 2.2.

$$ReLU(x) = \max(0, x)$$

2.3.2. Loss function

The loss function measures the performance of the neural network model by numerically assessing how well the model's predictions match the actual (ground truth) values, in case of supervised learning. In unsupervised learning, however, the loss function quantifies how well the model is achieving its objectives. During the training of neural networks, the primary goal is to find model parameters (weights and biases) that minimize the loss function to the smallest possible value. This is achieved using optimization algorithms such as gradient descent, which iteratively adjusts the model parameters to decrease the loss. One of the most used loss functions for regression tasks is Mean Squared Error (MSE), as demonstrated by Equation 2.3. The MSE calculates the average squared difference between predicted ($\hat{y}_i$) and actual (y_i) values, where n is the number of samples.

Equation 2.3.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(yi - \hat{y}_i)^2$$

2.3.3. Weight update (Backpropagation)

The weights and biases of the model are adjusted to improve its performance using the backpropagation algorithm. This process involves calculating the gradients of the loss function with respect to the weights and biases and adjusting them to minimize the loss. Backpropagation relies on the chain rule of calculus; the error is propagated backward through the layers, and the chain rule is applied at each step to calculate the gradients. These gradients represent the rate of change of the loss function with respect to each parameter and guide the updates to the parameters during training. The learning rate determines the size of the steps taken during parameter updates. A limitation of backpropagation is the potential for the vanishing and exploding gradient problem. In the vanishing gradients scenario, gradients become very small during backpropagation through many layers. Conversely, in the exploding gradient scenario, gradients during backpropagation become excessively large, leading to numerical instability and difficulties in training the neural network. Techniques such as weight initialization and using activation functions that mitigate vanishing gradients, such as Leaky ReLU, are employed. The gradient descent rule for a weight W is demonstrated in Equation 2.4, where η is the learning rate [54].

Equation 2.4.

$$W_{new} = W_{old} - \eta \times \frac{\partial Loss}{\partial W}$$

2.3.4. Regularization techniques

To prevent overfitting and enhance the generalizability of neural networks on unseen data, dropout is a widely used regularization technique. Dropout operates by randomly deactivating a fraction of neurons in the network during each iteration of the training process. It mimics the idea of training an ensemble of multiple models, as each iteration sees a different set of neurons contributing to the network's performance. The dropout probability, usually set between 0.2 and 0.5, determines the likelihood of a neuron being dropped out. This technique makes the model more robust by preventing the network from solely relying on specific neurons, enabling the model to learn diverse features and improve its capability to generalize well to unseen data.

Another technique to enhance the generalizability of neural networks on unseen data and prevent overfitting is early stopping during training. This technique involves continuously monitoring the model's performance on a separate dataset, the validation set, throughout the training process. The key idea is to stop the training when the model's performance on the validation set ceases to improve or starts to decrease after a prespecified iteration. During training, the model's performance on the validation set is routinely assessed, typically after each epoch. The early stopping criterion is often based on the loss function or other metrics such as accuracy. Training is terminated if there is no improvement in the performance on the validation set for a specified

number of evaluations. The model parameters associated with the best performance on the validation set are then retained as the final version of the model. This technique prevents the model from becoming overly specific to the training set data and ensures the capture of more robust patterns capable of generalizing to unseen data.

Data augmentation can be introduced within the training data to reduce overfitting and improve generalizability. This process involves artificially increasing the training data by applying various transformations such as rotation, scaling, flipping, and cropping. By presenting the model with diverse examples of the same data, it becomes more capable of recognizing finer patterns and becomes less sensitive to small variations. Data augmentation serves to reduce overfitting by providing the model with a more extensive and varied training dataset. This helps prevent the model from memorizing specific patterns, encouraging it to focus more on essential features across multiple variations. Consequently, this leads to a more generalized representation and improved performance on unseen data.

2.3.5. Weight Initialization

Weight initialization is the process of assigning initial values to the weights of the network before training. Proper weight initialization is a crucial step as it influences the performance and capabilities of the ANN and can significantly impact the training process. If the initial weights are too small, the vanishing gradient problem may occur during backpropagation, leading to slow and halted training. On the other hand, if the weights are too large, the gradients may explode, causing the model to fail to converge. Symmetric weight initialization across the network can result in the model learning the same features during training, reducing its capacity to perform well on unseen data. Therefore, proper weight initialization should involve asymmetry, allowing neurons to learn different features.

Various techniques have been introduced as a principled way to set the initial weights of a ANN, such as Xavier/Glorot initialization [55] and He initialization [56], to address the mentioned issues by considering the number of input and output units in a layer. GlorotNormal initialization, also known as Xavier Normal initialization, sets the initial weights in such a way that the variance of the activations remains roughly the same across different layers. For a layer with n_{in} input units and n_{out} output units, GlorotNormal initialization sets the initial weights by sampling from a normal distribution with a mean of 0 and a standard deviation of σ, as shown in equation 2.4. GlorotNormal initialization, along with other similar methods like He initialization, which follows the same process with a different distribution, allows the model to mitigate the issues of vanishing and exploding gradients, making the model more robust and efficient.

Equation 2.5.

$$\sigma = \sqrt{\frac{2}{n_{in} + n_{out}}}$$

2.4. Convolutional Neural Networks

CNNs, short for Convolutional Neural Networks, are a type of ANN specially designed for processing and analyzing visual data. They have proven highly effective in various computer vision tasks such as classification, object detection, and recognition [57], [58], [59]. Inspired by the visual processing of the human brain, CNNs can automatically and adaptively learn spatial hierarchies of features from input data. CNNs typically consist of three main types of layers: the convolutional layer, pooling layer, and fully connected layer [60].

2.4.1. Convolutional operation

The convolutional layer functions by sliding a filter (also known as a kernel) across the width and height of the input data, calculating the dot product between the filter and the local region at each position in the input. However, in the context of the pointwise convolution layer designed for point cloud data, it operates on individual points within the point cloud rather than structured grids. Equation 2.5. presents the general formula for a pointwise convolutional layer. The summation over J encompasses all input points, indicating that each output point is influenced by every point in the input [15]. This characteristic of pointwise convolution ensures that each output point is directly linked to all input points, and their convolution represents a weighted sum of the input points with learnable weights and biases. The weight W_{ij} indicates the significance of the connection between the $i-th$ output and the $j-th$ input point, enabling the network to discern meaningful patterns and relationships within the point cloud. Throughout training, these weights and biases are iteratively adjusted using algorithms like gradient descent to minimize the loss function. Pointwise convolutional layer often uses activation functions to introduce non-linearity into the model [54]. ReLU, as introduced in Section 2.2.2, is a common activation function used in convolutional layers, including pointwise convolutions.

Equation 2.5.

$$Y_i = ReLu(\sum_{j=1}^{N} W_{ij} \cdot X_j + b_i)$$

In this formula:

- Yi is the output value at the $i-th$ point in the output feature map.
- N is the number of input points in the point cloud.
- X_j represents the input value at the $j-th$ point in the input feature map.
- W_{ij} denotes the weight associated with the connection between the $i-th$ output point and the $j-th$ input point.
- b_i is the bias term associated with the $i-th$ output point.

In the context of a 2D pointwise convolutional layer, the operation is analogous to the 1D pointwise convolution but is applied to 2D data, such as images or grids of points. The mathematical formula for 2D pointwise convolution can be observed in Equation 2.6. This convolution operation is

executed at every position in the 2D feature map, allowing it to capture and extract local patterns and spatial relationships within the data. The weights W and biases b are iteratively adjusted during training to minimize the loss function, ensuring that the layer effectively learns meaningful features from the input.

Equation 2.6.

$$Y_{i,j} = f\left(\sum_{p=1}^{H}\sum_{q=1}^{W} W_{i,j,p,q}.X_{p,q} + b_{i,j}\right)$$

Where:

- $Y_{i,j}$is the output at position (i,j) in the feature map.
- H is the height of the input feature map.
- W is the width of the input feature map.
- $X_{p,q}$ represents the input value at position (p,q) in the input feature map.
- $W_{i,j,p,q}$ denotes the weight associated with the connection between the (i,j) −th output position and the (p,q) −th input position.
- $b_{i,j}$ is the bias term associated with the (i,j) −th output position.
- f is the activation function.

2.4.2. Max pooling

Max pooling is a common operation employed in CNNs to act as a down sampling method, effectively reducing the spatial dimensions of the input feature maps and minimizing computation. When dealing with point cloud data, pooling is applied individually to each point. The max pooling operation, as illustrated in Equation 2.7, partitions the input into non-overlapping regions and selects the maximum value from each region. Importantly, the max pooling operation lacks any learnable parameters; it simply identifies the maximum value within each predefined pooling region. This operation plays a crucial role in decreasing the spatial resolution of feature maps while capturing the most essential features from the input [54].

Equation 2.7.

$$Y_i = max_{j \in pooling\ region}(X_j)$$

- Y_i is the output value at the $i-th$ point in the pooled feature map.
- $X_j$represents the input value at the $j-th$ point in the input feature map.
- The pooling region consists of points that are grouped together for the pooling operation.

2.4.3. Fully connected (Dense) layers

The fully connected layer plays a crucial role in transforming the high-dimensional features extracted by the convolutional layers into the desired final output, essentially serving as a global operation. While convolutional layers concentrate on local patterns and spatial relationships, fully

connected layers establish connections between every neuron in the input to each neuron in the output. In the context of landmark localization, where the model aims to localize the landmarks in the point cloud, the fully connected layer would output triplets of values corresponding to the x, y, and z coordinates for each of the predicted points.

The fully connected layer operates through a weighted sum of input values, followed by an activation function as shown in Equation 2.8, where f represents the activation function, such as ReLU, and M is the number of neurons in the previous layer. The crucial distinction in the formula between the pointwise convolution layer and the fully connected layer lies in the fact that, in the pointwise convolution layer, the operation is applied to individual points within the point cloud, focusing on specific points. Conversely, in the fully connected layer, each neuron is connected to every neuron in the previous layer, allowing it to capture complex relationships across the entire input of the fully connected layer. This broader connectivity enables the fully connected layer to understand intricate patterns and dependencies present in the extracted features from the previous layers.

Equation 2.8.

$$Y_i = f\left(\sum_{j=1}^{M} W_{ij} \cdot X_j + b_i\right)$$

2.5. PointNet

PointNet is a deep learning architecture specifically designed for processing point cloud data, showcasing its versatility in handling unordered point sets. This characteristic makes it applicable to various 3D computer vision tasks. Notably, PointNet excels in processing point clouds with invariant permutations, rotations, and translations. It achieves invariant permutation, meaning it processes point clouds without considering the order of the points, and the arrangement of points does not impact the final result. Additionally, PointNet is invariant to rotation and translation, ensuring that varying orientations and positions of point clouds do not affect the model's output; they are processed consistently regardless of their transformations in 3D space. PointNet’ s ability to capture features from neighboring points for all points in the point cloud enhances its performance in segmentation tasks. The architecture has demonstrated promising results across various point cloud applications, including classification and segmentation [61] [62].

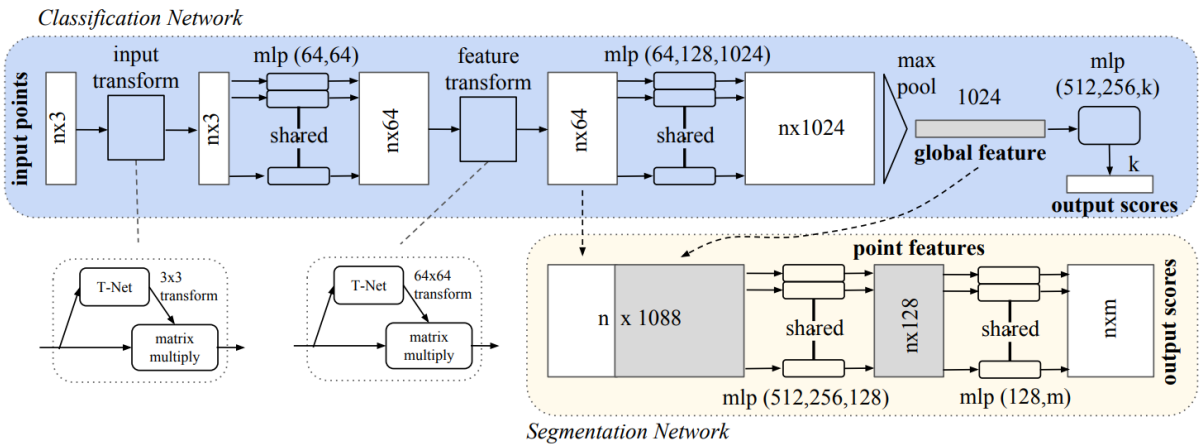


***Figure 2.4**. PointNet Architecture [61]*

The overall architecture of PointNet is depicted in figure 2.4. [61]. The input shape of the model is Nx3, with N representing the number of points in the point cloud, and 3 corresponding to X, Y, and Z coordinates. To transform the point cloud data to its canonical representation—where the data is represented in the canonical coordinate system, not accounting for translation and rotation—T-Net is employed in the input transform block. T-Net draws inspiration from Spatial Transformer Networks (STNs) [63], which offer pose normalization for 2D images through rotation and translation. Spatial transformers, applied in certain computer vision tasks, can reduce the need for extensive data augmentation [63].

Figure 2.5. illustrates the diagram of the Spatial Transformer Network architecture. Given an input U, the construction of the output V involves the use of θ, a grid generator, and a sampler. The localization network, typically composed of fully connected or convolutional layers, is responsible for predicting the transformation parameters θ. θ can be viewed as a transformation matrix containing the adjustments needed to align the input data with a canonical representation. The grid generator generates a sampling grid based on θ, creating a set of coordinates that dictates how the input should be sampled using the sampler and how that sample must be transformed to produce the output image. The sampler executes the sampling operation, often employing bilinear interpolation, to calculate the values of the transformed images based on the grid. This process enables the Spatial Transformer Network to effectively align and transform input data to produce the desired output.

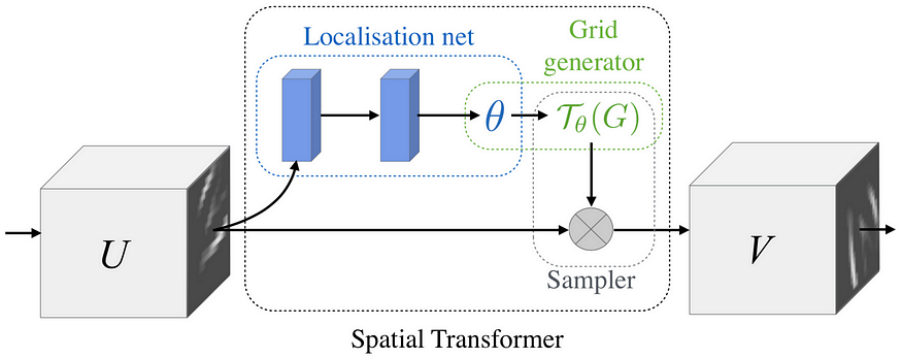


***Figure 2.5**. Diagram of the Spatial Transformer [63]*

In the case of T-Net, within the PointNet architecture, a similar operation occurs. An initial Nx3 point cloud is input with its coordinates in a local reference system subject to arbitrary rotation and/or translation. T-Net outputs the same Nx3 point cloud in its canonical presentation. T-Net functions as a regression network, producing a 3x3 affine or rigid transformation matrix. When this matrix is multiplied with each point in the point cloud, it transforms the entire point cloud to its canonical representation. This process reduces the need for a sampler compared to Spatial Transformer Networks (STN).

After obtaining the point cloud in its canonical representation, the point cloud feature extraction is conducted by passing it through 1x1 convolutional layers to capture local spatial features, resulting in higher-dimensional features of Nx64. The data is then passed through T-Net for pose normalization, producing a transformation matrix of shape 64x64. Following normalization, the point cloud data undergoes another round of feature extraction to obtain features in higher dimensions of 128 and 1024. At this stage, every point in the point cloud contains 1024 values of information, denoted as Nx1024. To reduce complexity and noise, a max pooling operation is applied, and the information is stored as a global feature vector, which can be passed to the classification or segmentation head, depending on the task at hand.

As discussed previously, PointNet is capable of processing point clouds where the points of the point cloud are order arbitrary. It achieves this by employing weight sharing across its architecture. In traditional deep learning networks, each input has its own set of weights and biases that are adjusted independently during training. However, in PointNet, a shared neural network is used for each point of the point cloud, meaning the same set of weights and biases are utilized for processing each point. This approach enables the model to produce consistent results regardless of the order of the input points.

For the task of classification, the global feature vector is fed through three fully connected layers: the first layer with 512 neurons, the second layer with 256, and the final output layer with K neurons. Here, K represents the number of classes to be predicted. Regarding the segmentation head, the global feature vector is concatenated with the output of the 64x64 T-Net, producing an Nx1088 data vector. This data vector passes through three fully connected layers with 512, 256, and 128 neurons respectively, resulting in an Nx128 data vector. The generated data is then forwarded through the output layer with M neurons, yielding an NxM vector, where each point is

classified into one of the segmentation classes (M). PointNet architecture can be utilized for the task of facial landmark localization by taking advantage of its feature extraction capabilities to capture spatial information, followed by a custom fully connected layer to map the features to predict the coordinates of the facial landmarks.

3. Experimental Protocols

This chapter provides a description of the approach used in this study to automated landmarks identification task. After introducing the experimental set-up and dataset, implementation details of pre-processing steps and models' description are detailed. Finally, the metrics employed for assessing network performances are presented.

3.1. Materials

The present study used retrospective data retrieved from the dataset of the LAFAS (Laboratory of Functional Anatomy of Stomatognathic system of Dipartimento di Scienze Biomediche per la Salute, Università degli Studi di Milano), which contains a collection of 3D facial scans acquired in the last 20 years. The selected dataset comprises 200 3D facial models, relative to heathy subjects aged 18 to 49 years old of both sexes. Excluded from this study were subjects with a previous history of craniofacial traumas, congenital anomalies, or craniofacial surgery. The 3D facial models were acquired using VECTRA M3, a fixed stereophotogrammetric device, (Canfield Scientific Inc., Fairfield, NJ, USA), and VECTRA H1, a portable one (Canfield Scientific Inc., Fairfield, NJ, USA). These instruments are employed routinely in the LAFAS and have demonstrated comparability and equivalence [64] [65]. For each 3D scan, 50 facial landmarks were annotated following a protocol developed by Ferrario et al. [66] [19]. These landmarks, shown in figure 3.1. and listed in table 3.1., were manually annotated using a point-and-click interface by the skilled personnel of the LAFAS. For improved precision in manual landmark localization, specific landmarks were labeled directly with an eyeliner on the face before data acquisition. These labeled landmarks, visible in the face texture, facilitated more straightforward and accurate manual localization. Each 3D face model is organized within a dedicated folder. Depending on the device used for acquisition, each folder contains 2 or 3 JPEG images, a .obj file, an MTL file, and a .txt file that includes 50 3D XYZ coordinates representing the 50 landmarks.

Table 3.1. *The set of 50 landmarks used in this study. From C. Sforza et al. [21].*

Region of the face		Name of the landmark	Midline	Paired
Forehead	tr	Trichion	✓	
	g	Glabella	✓	
Eyes	ex	Exocanthion		✓
	en	Endocanthion		✓
	os	Orbitale superius		✓
	or	Orbitale		✓
Nose	n	Nasion	✓	
	prn	Pronasale	✓	
	c'	Columella	✓	
	sn	Subnasale	✓	
	al	Alare		✓

	ac	Nasal alar crest		✓
	itn	Inferior point of the nostril axis		✓
	stn	Superior point of the nostril axis		✓
Mouth and lips	ls	Labiale superius	✓	
	sto	Stomion	✓	
	li	Labiale inferius	✓	
	sl	Sublabiale	✓	
	chp	Crista philtri		✓
	ch	Cheilion		✓
Chin	pg	Pogonion	✓	
	me	Menton	✓	
Lateral part of the face	ft	Frontotemporale		✓
	chk	Cheek		✓
	zy	Zygion		✓
	go	Gonion		✓
Ears	t	Tragion		✓
	pra	Preaurale		✓
	sa	Superaurale		✓
	pa	Postaurale		✓
	sba	Subaurale		✓

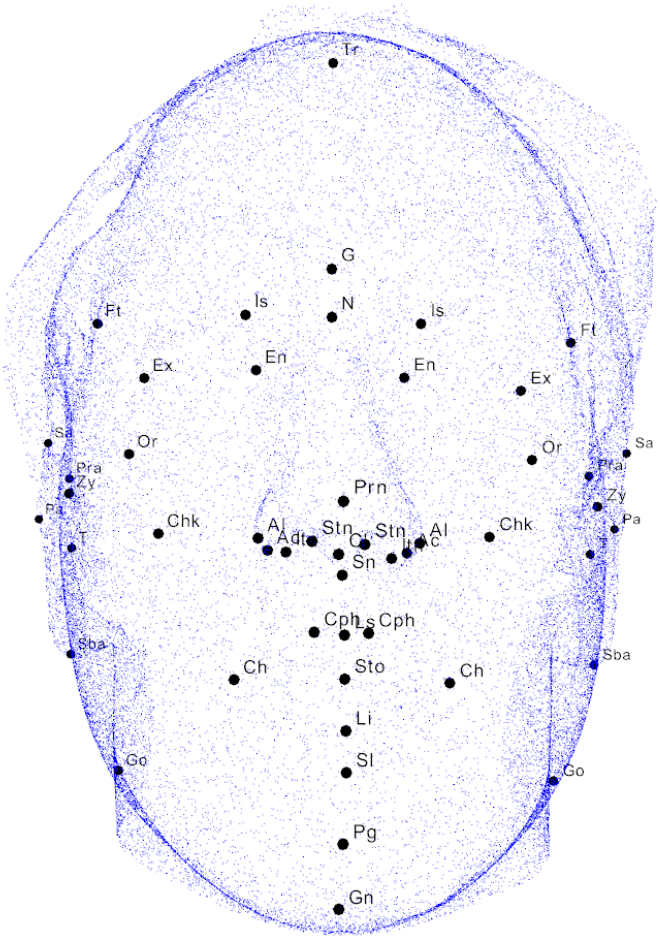


Figure 3.1. *3D Facial Model Represented as a Point Cloud with Identified Anatomical Facial Landmarks from C. Sforza et al. [21]*

Out of the 200 cases, 65 were acquired by VECTRA M3, which captured 3 JPEG images from left, right, and front perspectives of the face. Additionally, the remaining 135 cases were acquired by VECTRA H1, which captured 2 images, but were allocated to a single JPEG image, where half of the image included the left side, and the other half included the right side of the face. These images serve as texture maps for the 3D facial model, containing information such as skin color, facial features, and other textures that contribute to the realistic appearance of the 3D model.

The 3D facial models are stored as .obj files, which are plain text files representing 3D geometry. These files contain information such as the coordinates of the vertices (v-), their normals (vn-), texture coordinates, and other details regarding the 3D object. It's noteworthy that in our dataset, the number of vertices varies across cases. This variability can be attributed to factors such as facial hair, different methods of image acquisition, ambient lighting, movement, location, and placement, among other factors [43]. The distribution of the number of vertices per case is illustrated in figure 3.2.

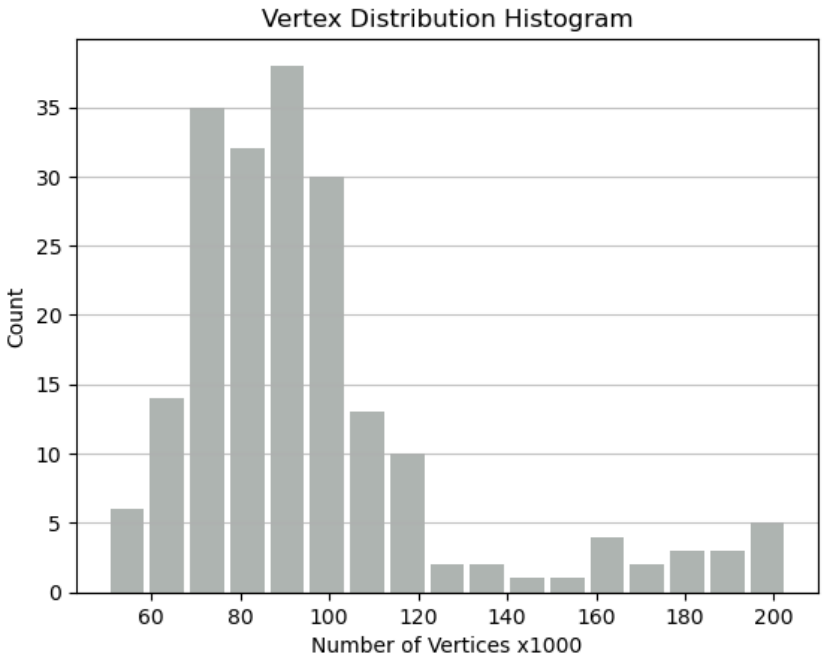


Figure 3.2. *Histogram of the Number of Vertices per Case*

3.2. Data Pre-Processing

To conduct this study, Python 3.10 was employed along with relevant libraries for preprocessing. The Python library Trimesh [67] facilitated the import of facial models into the Python environment. Utilizing the Trimesh library, the facial 3D models were sampled to consist of 100,000 vertices using the Triangle Point Picking method [68]. Maintaining a consistent number of vertices in the facial models is crucial for subsequent stages of development, lowering computational power and complexity. The choice of 100,000 samples was made based on its peak occurrence in the distribution illustrated in figure 3.2. The resulting 200 facial models, each having 100,000 vertices, were stored in a multidimensional array with the shape of (200, 100000, 3). Additionally, for each facial model, their corresponding landmarks were imported into the environment using a simple loop that iterates through each folder, extracts the coordinates of the landmarks, and stores them in a multidimensional array with the shape of (200, 50, 3). Figure 3.4(a) presents a visualization of one of the facial models after the resampling process. As can be seen from the figure, after the resampling process the artifacts have been adjusted thanks to the Point Picking sampling method.

The overall preproccesing pipeline of the point cloud data can be seen in figure 3.3. Data augmentation and adding vector normals of the points of the point cloud, as detailed in the following sections, can be integrated into the pipeline to enhance the model's generalizability.

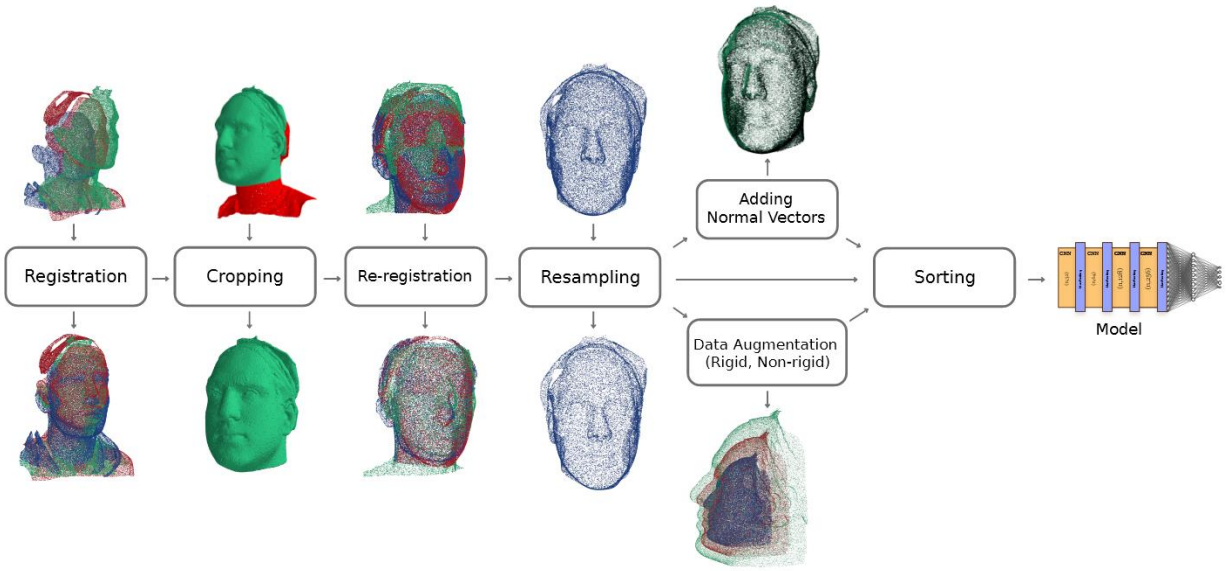
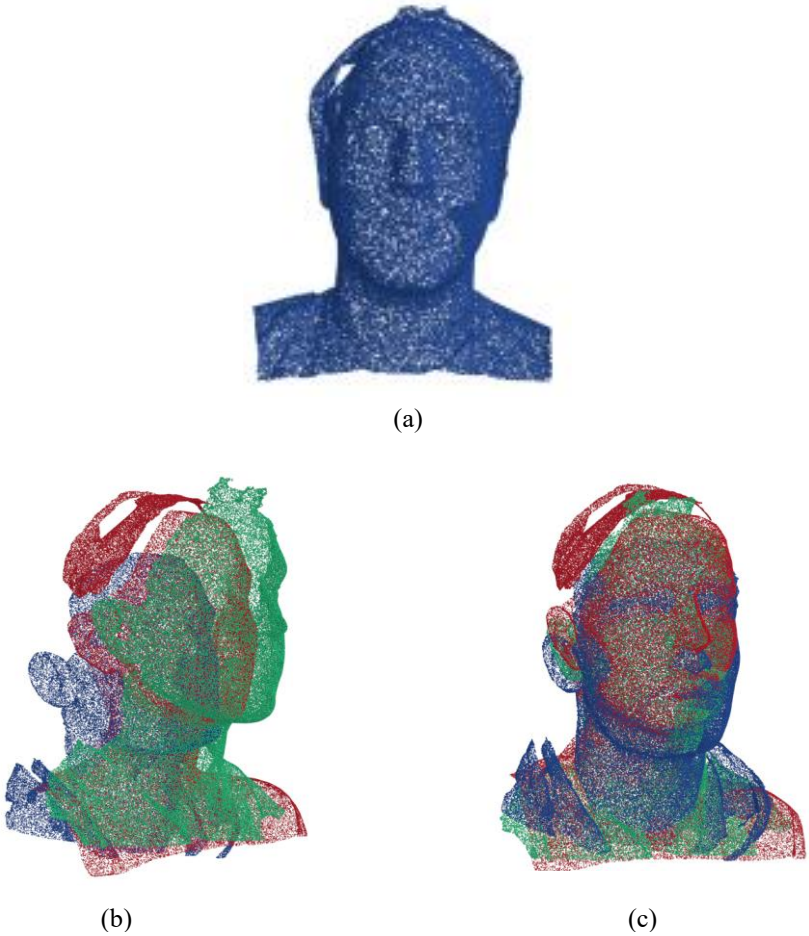


Figure 3.3. *Preprocessing Pipeline*

The 3D facial models exhibit consistent rotation and translation in space, as depicted in figure 3.4(b), where each point cloud in red, green, and blue represents a different subject in space. As a preliminary preprocessing step, all the facial models along with their corresponding landmarks were aligned using the Iterative Closest Point (ICP) registration algorithm. In this process, the first facial model in the array served as the arbitrary target point cloud, onto which all other facial models were aligned. The mean Euclidean distance between corresponding points served as the objective function for the algorithm, which aimed to minimize this distance by adjusting the transformation matrix, aligning the faces rigidly without causing any deformation. The iteration for adjusting the transformation matrix terminated either when the mean Euclidean distance of corresponding points fell below 1e-6 or when the number of iterations exceeded 100. Figure 3.4(b) and (c) illustrate facial models before and after the registration process.



(a)

(b) (c)

Figure 3.4. *. Represents the 3D Facial Models (a) Before (b) and After (c) the ICP Registration Process*

While acquiring stereophotogrammetry images of the subjects' faces, the upper chest of the subject is often included in their 3D image, as depicted in red in Figure 3.5. Some male subjects wearing shirts with larger collars or female subjects with long hair and hairbands can be observed as outliers, introducing inconsistency within the point clouds, and potentially affecting the final results. To address this issue, considering the restricted focus on the facial area excluding the neck, cropping was applied to facial models to remove any extraneous data and retain only the region of interest. Leveraging the previous preprocessing step that aligned all the faces together, cropping was achieved by simply removing any points in the point cloud that fell outside the predefined region of the X, Y, and Z axes. Figure 3.5 illustrates the facial models before and after the cropping process, the points in red are the parts of the point cloud which are removed.

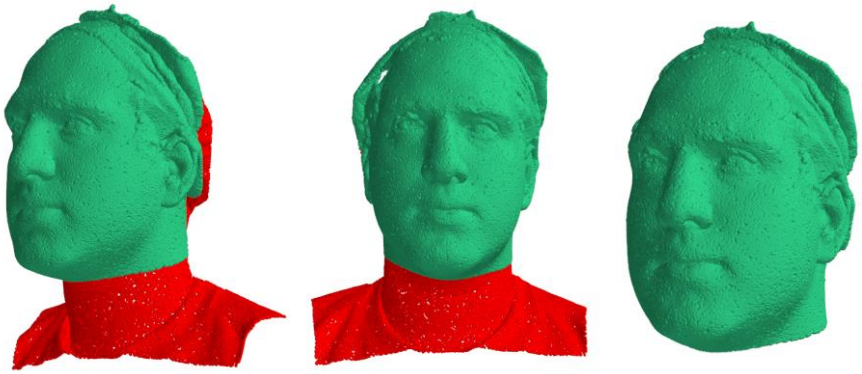


Figure 3.5. *Representing the Facial Model Before and After the Cropping Process.*

After the cropping step, the facial models did not have a consistent number of vertices. Similar to the first preprocessing step, using the Trimesh library, the facial models were resampled to have a consistent number of samples. Different resampling techniques were utilized, using varying numbers of samples, such as techniques to have evenly spaced-out points across the point clouds. The final results for the resampling techniques will be showcased and described in the results section of the study. However, a default value of 41,000 was chosen to resample the points of the point cloud, following the same reasoning of selecting the peak value of the vertex distribution histogram of the cropped point clouds. For a final refinement and alignment of the faces, once again, the ICP registration algorithm was utilized on the facial model with their corresponding landmarks.

The point clouds are indexed in an arbitrary manner, the arbitrary indexing of points means that spatially close points are not necessarily indexed sequentially and may cause ambiguities to model to capture complex patterns. As well as the order of the points of the point clouds determine the order of features in the fully connected layers [15], having arbitrary order can introduce instability and inconsistency in the model's learning process resulting difficulties in convergence and models inability to generalize. To mitigate this issue, as discussed in section 3.2, Morton code sorting, and sorting from a reference point is employed to address the mentioned issues and sort the points based on their spatial proximity to each other. Sorting the points of the point cloud based on Morton code, or a reference point, can enable the model to be more robust and to produce consistent results.

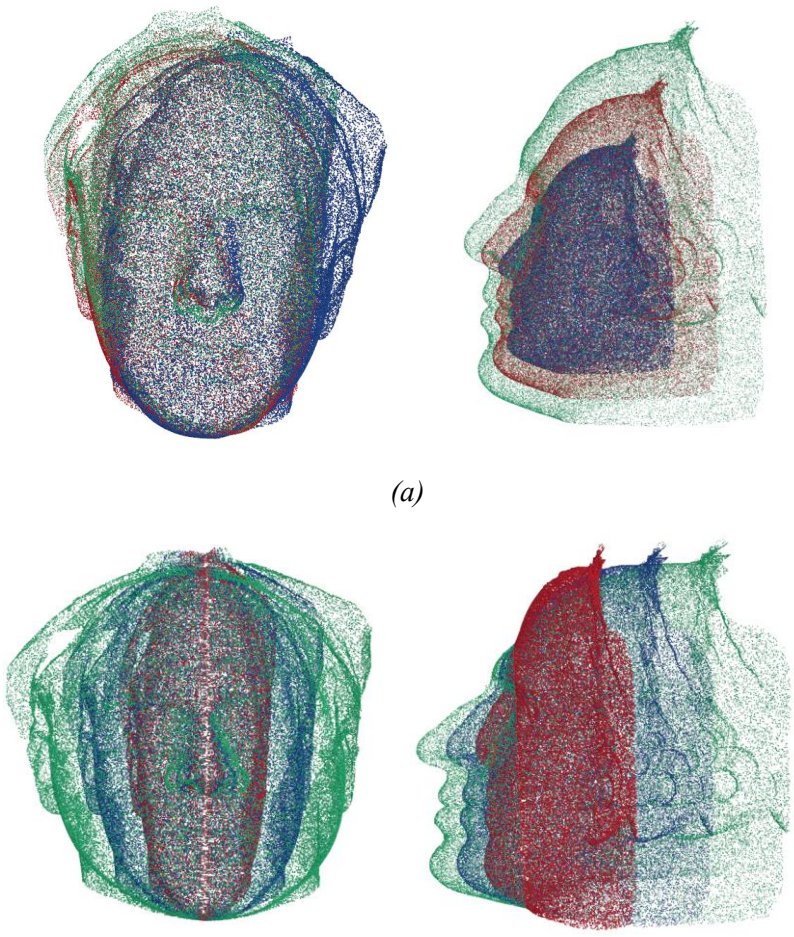
Furthermore, data splitting was applied to divide the dataset into a training set and testing set. The training set included 80 percent of the data (157 subjects) while the validation set included 20 percent of the data (40 subjects). The dataset was only divided into training and testing, and used the test set for validation as well, due to the limited amount of data available. Three subjects were removed from the dataset, due to failing the registration procedure or containing outliers that can impact the training and evaluation process. However, a more robust registration process can be used, which will be discussed in the later sections.

3.2.1. Data Augmentation

As presented in figure 3.3, after cropping, registration and resampling, data augmentation was applied in both rigid and non-rigid ways. In the case of rigid augmentation, rotation and scaling

operations were applied to the facial models along with their corresponding landmarks. The rotation augmentation rotates the point clouds around a specific axis by a given angle. This process involves creating a rotation vector based on the specified axis and angle and using it to construct a rotation matrix. The rotation matrix is then multiplied with the point cloud, resulting in the rotated point cloud. Additionally, data can be scaled based on a specified scale factor. This process involves multiplying each XYZ coordinate of every point by the given scale factor, ensuring that the entire point cloud is uniformly expanded or contracted based on the provided scaling factor. A scale factor of 0.5 would shrink the size of the point cloud by half, while a scale factor of 2.0 would double its size. Different rotation angles and scaling factors were utilized and tested to improve landmark localization, and these results will be discussed further in the results section of the study. Figure 3.6(a) demonstrates an example of a rigid data augmentation, with the red point cloud being the original non-augmented point cloud. The figure on the left shows' rotation on the point clouds, and the figure on the right shows scaling with factors 0.75, 1, and 1,25 for the point clouds in blue, red and green respectively.

Non-rigid data augmentation was applied to the facial models to enhance generalizability and improve the model's ability to make accurate predictions, especially when faced with variations in facial shapes and forms. Training the model on a broader range of facial variations results in a more robust and adaptable model. Non-rigid augmentation is implemented by squeezing or stretching the faces using scaling factors for each of the X, Y, or Z axes. The process involves creating a 4x4 identity matrix and placing the X, Y, and Z scaling factors diagonally on the matrix. This matrix is then multiplied by every point in the point cloud to produce the augmented point cloud. A default scale value of 1 result in no change, while values greater than 1 introduce a stretching effect, and values smaller than 1 introduce a squeezing effect along the corresponding axis. This non-rigid scaling approach allows for flexible deformation along specific axes while preserving the overall structure of the data. Various rotation angles, scaling factors, and stretching scales were applied exclusively to the training set for the purpose of enhancing landmark localization. The evaluation of these augmentations was conducted on the original, non-augmented data to minimize biases and ensure a fair assessment, as detailed in the results section. Figure 3.6(b) demonstrates an example of a non-rigid data augmentation, with the point cloud in blue representing the original non-augmented point cloud. Squeezing and stretching factors of 0.6, 1.0, and 1.4 were applied to the point clouds in red, blue, and green, respectively, on the X-axis for the image on the left, and on the Z-axis for the image on the right.



(a)

(b)

Figure 3.6. *Example of Rigid Data Augmentation, Rotation, and Scaling(a) and a Non-Rigid Data Augmentation on the X and Z Axis(b)*

3.2.2. Vector Normals

The vector normals of the point clouds represent perpendicular vectors to the surfaces of the points within the cloud. These vectors offer crucial insights into the orientation and direction of the surfaces at each point, serving purposes like surface analysis, lighting and shading effects, and feature extraction for various computer vision and geometry processing tasks. In our study, the Open3D library [69] was employed within the Python environment to compute the vector normals for each point in the point clouds.

The Open3D library employs a neighborhood search combined with covariance matrix analysis to estimate the normals at each point. For a given point in the point cloud, a local neighborhood is defined with a predefined radius. The covariance matrix is then calculated for each of the points within the local neighborhood, which its eigenvectors correspond to the directions of maximum variance in the local distribution. The eigenvector corresponding to the smallest eigenvalue, representing the direction of the least variation in the local neighborhood, is considered the normal vector of the point, providing the orientation of the local surface. This process is done for all the

points within the point cloud to produce the vector normals of the point cloud. Furthermore, the length of the normals is commonly normalized to ensure unit length. The resulting vector normal for a point consists of three values, represented as (nx, ny, nz), where each value indicates the direction of the normal vector along its axis.

To prepare the point clouds and their corresponding normal vectors for input to the model, normalization is applied to ensure they have the same range of values. Subsequently, they are stacked together. The resulting data, comprising point clouds with their associated vector normals, is stored in a multidimensional array with a shape of (200, n, 6). Here, 'n', which in the default case is 41000, denotes the number of points in the point cloud, with three values representing the coordinates of each point and three values representing the vector normal along the X, Y, and Z axes for each point. The influence of normal vector as feature for facial landmark localization will be discussed in the results section.

3.3. Model Training

In this study, a two-stage DL model was employed for facial landmark localization. The training was done in a supervised way, using as target output the manual annotations provided by LAFAS. Initially, an Estimation stage was trained using all the points of the point cloud, aiming to provide approximate landmarks' locations. Subsequently, a Refinement stage was trained on points within a predefined radius from the landmarks' positions predicted by the Estimation model, focusing on refining the landmark localization for greater accuracy. Furthermore, a third model referred to as EsTi model, where the T-Net architecture was incorporated with the Estimation model. This additional model, like the Estimation model, is subject to further refinement to enhance the precision of landmark localization.

The input of the models is a point cloud represented as a multidimensional array with the shape of (m, n, c), where m represents the number of subjects (in our case, 157 for training), n represents the number of points in the point cloud (default value of 41,000), and c represents the number of features/channels for each point in the point cloud. In the case of no added extra features, it will have only 3 features representing the X, Y, and Z coordinates. In the case of utilizing other features such as vector normals and RGB data, each can add 3 more channels, resulting in either 6 or 9 channels.

The output of the model is another multidimensional array with the shape of (m, 50, 3), where, as previously mentioned, m represents the number of subjects. For each subject, there are 50 landmarks, and each landmark is represented by a 3D coordinate of X, Y, and Z. During model training, the weights of the models are adjusted to minimize the difference between the predicted coordinates and the actual ground truth coordinates. The weights of the model are updated iteratively after processing each batch of training data during the training process. Batches are subsets of the entire training set, and the batch size is the number of training examples in each batch. Commonly, batch sizes are powers of 2 due to memory structures, resulting in optimal memory allocation. Instead of adjusting the model's weight after each individual training example or using the entire dataset, weight updates are done on batches to increase computational and memory efficiency while having a faster convergence time.

3.3.1. Estimation Stage

The Estimation stage is composed of pointwise Convolutional Neural Networks (CNN) designed for feature extraction, followed by a fully connected layer to capture global features and facilitate predictions. The CNN is composed of four convolutional blocks, in which two pointwise 1-dimensional convolutional layers are employed followed by a MaxPooling layer. Convolutional operation is iterated twice in each block, utilizing 32, 64, 128, and 256 filters, respectively to capture local patterns and relationships among neighboring points. Each convolutional layer is succeeded by a Rectified Linear Unit (ReLU) activation function and is padded to maintain the input's original length. MaxPooling layers are introduced after each convolutional block to down-sample the data, allowing the model to concentrate on crucial features and to reduce computational complexity. In the default scenario, where each point cloud encompasses 41,000 points, MaxPooling is applied twice with a window size of 4 and twice with a window size of 5, resulting in 102 points, each possessing 256 features/channels (102, 256). It is remarked that, depending on the number of points in the point cloud, a varying number of MaxPooling layers should be employed to adapt the model, to find a balance between down-sampling and preserving important features.

Additionally, the extracted features are flattened to create a one-dimensional vector of features, which is then passed through fully connected (dense) layers to enable the model to capture features from different parts of the point cloud. A fully connected layer with 1024 units is employed, followed by a ReLU activation function. To prevent overfitting and increase the generalizability of the model, a Dropout layer was introduced after the dense layer with a dropout probability factor of 0.1, chosen based on experimental analysis. This layer randomly drops a fraction of the connections during training to reduce overfitting to the training set. The output layer consists of a densely connected layer with 150 units utilizing a linear activation function, serving as a regression layer to compute continuous numerical values as the final prediction. The produced 150 values are then reshaped into a multidimensional array of (50, 3), representing the 50 landmarks and their 3D X, Y, and Z coordinates. Figure 3.7. demonstrates the architecture of the Estimation model, consisting of four convolutional blocks as the feature extractor, and the fully connected layers to produce the prediction. In each CNN block the convolution operation is applied twice.

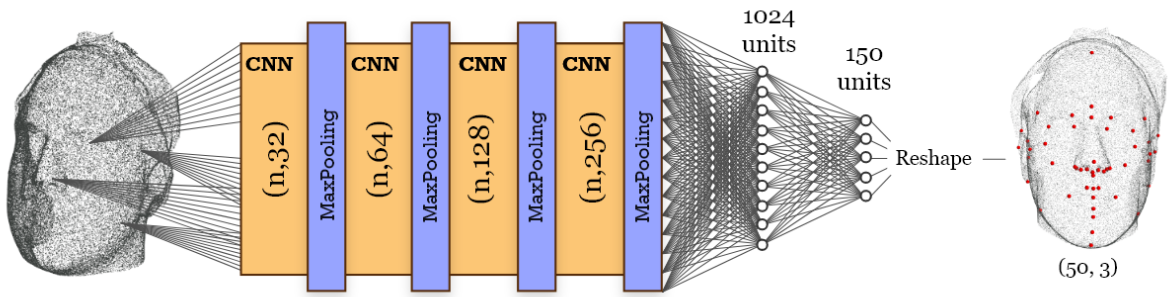


Figure 3.7. *The Architecture of the Estimation model*

The loss function utilized during the training of the model is the mean absolute error (MAE) between the predicted values and the actual ground truth values. The weights of the model were initialized using the GlorotNormal method. Adam optimizer, a variation of stochastic gradient descent (SGD), was used as the optimization algorithm to update and adjust the weights of the model. The model was trained with a batch size of 4, using 85 epochs, with an initial learning rate of 0.0010. An epoch is one complete pass through the entire training dataset, and between each epoch, the model's performance was evaluated and monitored on the test dataset. In the case of no improvement of the model on the test set after 10 epochs, the model's learning rate would be decreased by a factor of 0.3, and in the case of no improvement after 15 epochs, early stopping would be applied and the training process would be terminated, to avoid overfitting. The model weights that correspond to the best performing epoch would be loaded as the final parameters for the trained model. The lowest possible learning rate observed during the training was 1.e-06.

3.3.2. Refinement model

To enhance the localization accuracy of facial landmarks, a refined process of additional processing and feature extraction was implemented. This involves leveraging the predicted coordinates from the Estimation stage to train subsequent model stage, aiming to obtain superior performance. The procedure entails isolating only the points that fall within a predefined radius from the predicted landmarks and removing the rest. This process is executed iteratively by examining each predicted landmark of the Estimation model, and storing every point of the point cloud that is within a predefined radius. For this study, the chosen radius was determined by the mean error distance of the landmark exhibiting the highest discrepancy between its predicted value from the Estimation model and the actual ground truth, with addition of 3mm to ensure inclusion of the ground truth landmarks, resulting in a radius of 12.5 mm. The maximum error distance of the landmark was not chosen as the radius, as a precautionary measure to account for potential outliers and errors that may exist. Moreover, such anomalies can impact the outcomes of the refinement model. Figure 3.8. illustrates the point cloud in blue, with points within the chosen radius in yellow. The green points represent the ground truth coordinates of the landmarks, while the red points depict the prediction of the Estimation model. However, in some cases, some of the actual ground truth landmarks may fall outside the predefined radius, as showcased in Figure 3.8 for the landmarks located on the forehead and cheek.

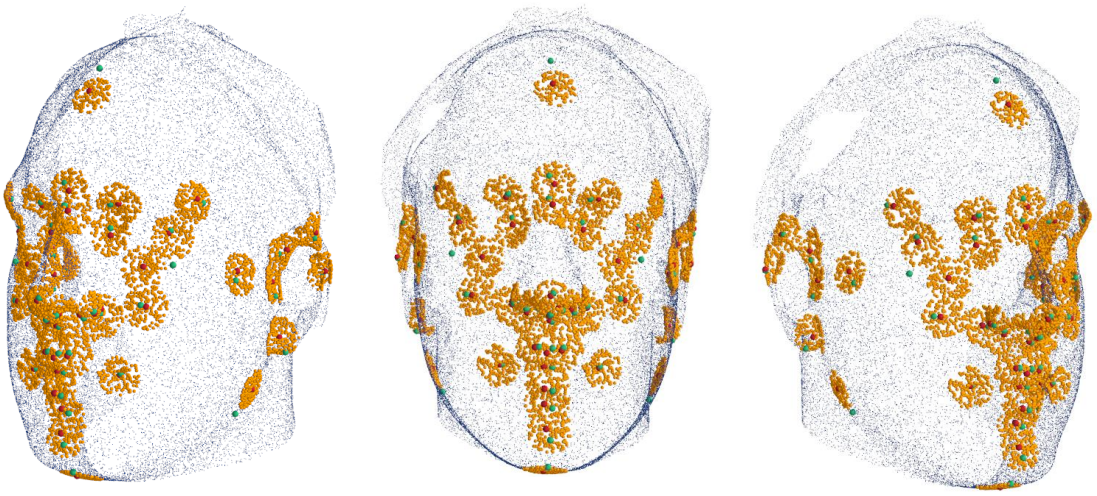


Figure 3.8. *Points in Orange Represent Those Within the Radius of the Estimation model's Prediction.*

For each predicted landmark of the Estimation stage, the points of the selected regions of interest were stored in an array, resulting in 50 sub-point clouds. Each sub-point cloud, being the point in orange, is then re-sampled to have 500 points, resulting in a new multidimensional array of shape (m, 50, 500, 3), where m is the total number of subjects, each subject has 50 sub-point clouds, and each sub-point cloud contains 500 vertices with 3 channels representing the X, Y, and Z coordinates. Optionally, additional features such as vector normals and RGB data can be included for each point. The resampling value of 500 was chosen through experimental analysis, which demonstrated that sampling with 500 vertices leads to more accurate localization of the landmarks. In cases where only the 3D coordinates of each point are retained, resulting in a multidimensional array of shape (m, 50, 500, 3), the data can be viewed as a 2D image for each subject. Each subject's data forms an image with 50 pixels in height and 500 pixels in width, with 3 channels, each channel representing a color intensity for a pixel. This representation allows the utilization of 2D convolutional neural networks in the training of the Refinement model.

Figure 3.9 illustrates an example of the 2D image feature obtained from the multidimensional array. The image comprises 50 columns and 500 rows of pixels, where each row represents one of the 50 sub-point clouds, and each pixel corresponds to a point in that sub-point cloud. The pixel values represent the normalized 3D coordinates of X, Y, and Z, mapped to the RGB (red, green, blue) channels, respectively. It's important to note that the RGB normalization is applied solely for the purpose of visualizing the data as a 2D image. During the training of the Refinement model, no such normalization is applied.

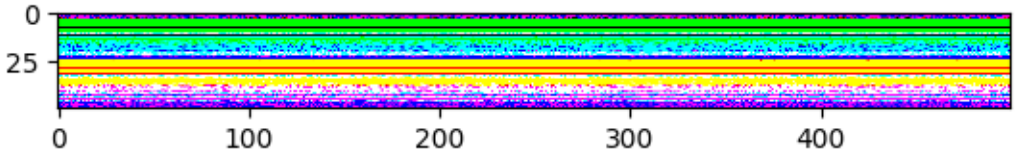


Figure 3.9. *The 2D Image Formed by Stacking the Sub-Clouds Together*

To facilitate the model's understanding and capture of spatial localization, as well as to provide ordered features, point sorting is applied to each of the sub-point clouds as a preprocessing step before training the Refinement stage. The same training and test dataset is used for training the Estimation stage to ensure consistent results and avoid introducing biases to the model. The Refinement stage is composed of three convolutional blocks for feature extraction followed by fully connected layers for producing predictions. Similar to the Estimation stage, within each block, there exist two pointwise convolutional layers where 2D convolution is applied. In each block, the convolution operation is applied with 32, 64, and 128 filters, utilizing ReLU activation function, followed by a MaxPooling layer.

In the refinement model stage, the MaxPooling operation is selectively applied to our data to prevent the loss of essential features. The feature array, as mentioned previously, has a shape of (50, 500, 3) for each subject. To not lose valuable information, MaxPooling is only applied to the 2nd axis of the feature vector, reducing the number of points for each sub-point cloud from 500. From a 2D image point of view, MaxPooling is only applied to the width, reducing columns of pixels from the image. If we were to reduce the dimensionality from the height of the image (1st axis), removing rows of pixels, we would be reducing the number of sub-point clouds from 50, losing valuable information such as the predictions from the Estimation model, their proximity points, and their corresponding ground truth landmarks. MaxPooling is applied twice with a window size of 5 and once with a window size of 4, on the 2nd axis resulting in features with a shape of (50, 5, 128). At this point, each of the sub-point clouds is transformed to contain 5 points where each point contains 128 channels of information only capturing, thanks to CNN.

The extracted features are flattened to produce a single-dimensional vector of features, which is subsequently passed through a fully connected layer to map the features to prediction coordinates. Similar to the Estimation model, the fully connected layer of the Refinement model consists of dense layers with 1024 units, utilizing the ReLU activation function. A Dropout layer is employed after the dense layer to prevent overfitting. Furthermore, the output layer consists of a linear activation function with 150 units, which are then reshaped to a multi-dimensional array of (50, 3), representing the 50 landmarks and their 3D X, Y, and Z coordinates. Figure 3.10. demonstrates the Refinement model's architecture.

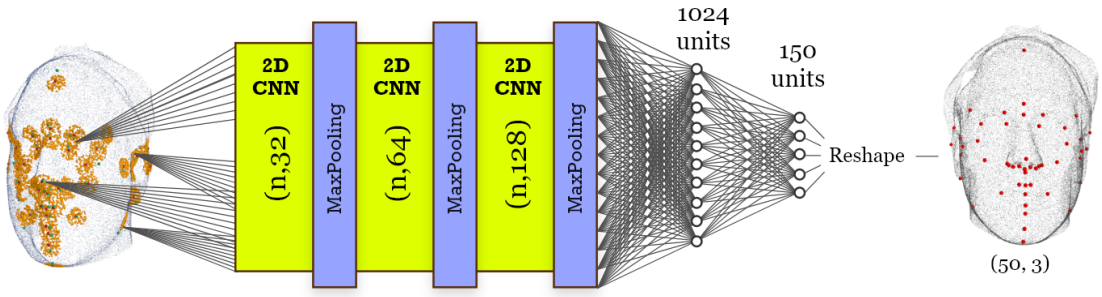


Figure 3.10. *The Architecture of the Refinement model*

The model training involved adjusting the weights to minimize the Mean Square Error (MSE) loss function, which computes the squared differences between the predicted values and the ground

truth values. The training procedure followed the same protocol as the Estimation stage, utilizing GlorotNormal initialization, Adam optimizer with an initial learning rate of 0.0010. Training was stopped after 59 epochs with a batch size of 4, incorporating early stopping and learning rate reduction in the case of no improvement on the test set.

3.3.3. EsTi model

Drawing inspiration from the PointNet model, the T-Net architecture can be added to the beginning of the estimation stage, enabling the model to consider the rotational and translational variations of the faces in space. As discussed in Chapter 2, the T-Net plays a role in transforming the point clouds to a canonical presentation by applying a transformation matrix, adjusted through the model's weights.

The architecture of the T-Net consists of three convolutional layers with an increasing number of filters: 64, 128, and 1024. Following the convolutional operations, a global max-pooling layer is applied to obtain a global feature representation. Three dense layers are utilized within the network, with the first two layers having 512 and 256 units, respectively. The number of units in the third layer depends on the number of features in use. For the case where only the 3D coordinates of each point are used as features, the third layer will have 9 units, resulting in a transformation matrix of size 3x3. The weight initialization of the third layer is set to zeros, representing an identity matrix, and Orthogonal regularization [70] is employed as the regularization technique for the dense layer. The role of the Orthogonal regularization is to encourage the weight matrix to be close to an orthogonal matrix, penalizing deviations from orthogonality during training. In this study, orthogonality of the matrix is crucial for preserving geometric relationships in the point cloud, ensuring that the learned transformation matrices are meaningful and do not introduce randomness within the features. Finally, the transformation matrix generated by the third dense layer is dot-multiplied with the input features, transforming the point cloud to its canonical presentation, which is then fed to the Estimation stage. All filter and unit values used in this study are derived from the original study of Qi et al. [61]. Figure 3.11 demonstrates the architecture of the EsTi model, where m represents the number of features for each point of the input point cloud.

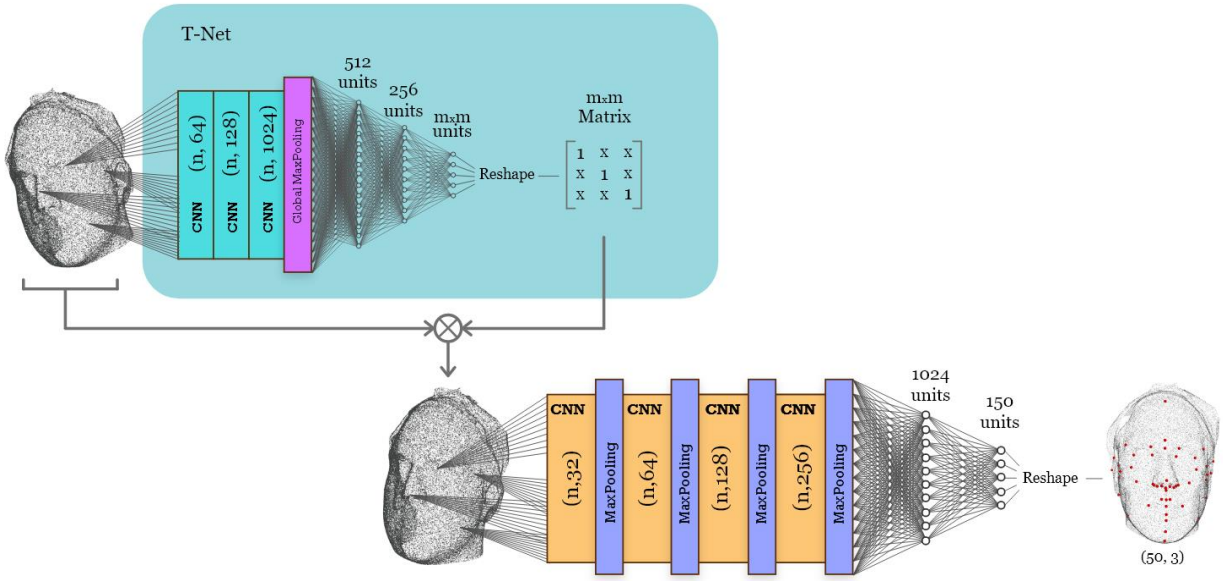


Figure 3.11. *The Architecture of the EsTi model*

Training of the EsTi model followed the same steps as the Estimation stage, utilizing MAE as the loss function, using Adam optimization, and the same weight initialization, except for the third dense layer of the T-Net where zeros were used as weight initialization. The predictions obtained from the EsTi model can be further processed to be fed into the Refinement stage to achieve an even more accurate landmark localization. This can be done by following the same procedures done for the output of the Estimation stage, where regions of interest within a predefined radius from the predicted landmarks were considered. This procedure enables the Refinement stage to capture the error patterns derived from either the Estimation or the EsTi model and compensate for them. The achieved results of this process will be discussed in the result section.

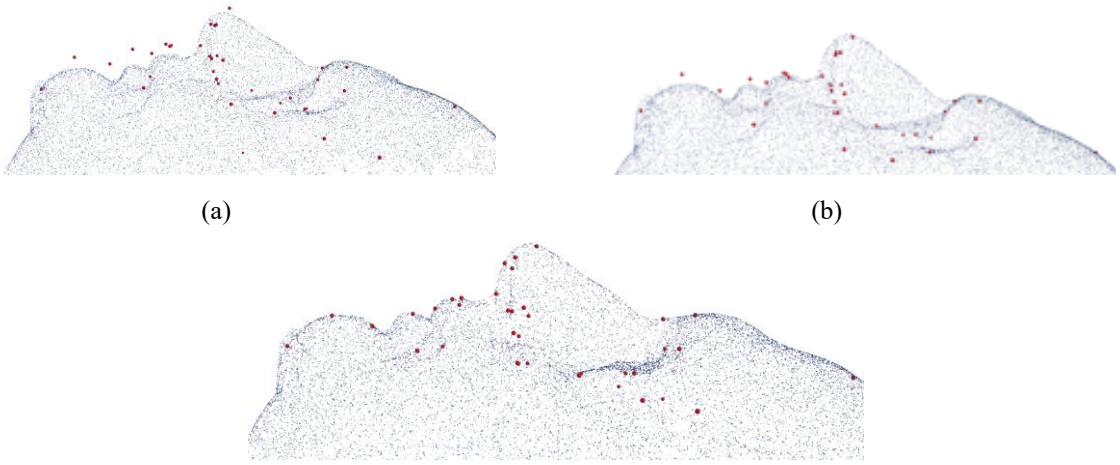
3.4. Post Processing

In practical scenarios, the predicted coordinates of landmarks may not align precisely with the surface of the 3D facial model due to inherent model complexity. Figure 3.11(a) illustrates an example where a predicted landmark in red does not align with the facial surface. To address this issue, two potential post-processing steps were introduced.

One approach involves finding the closest point in the point cloud to the predicted coordinate and assigning that point as the predicted landmark. However, the effectiveness of this step depends on the point cloud's vertex sampling rate, and it may not be optimal for noisy or sparse point clouds. The distance between the closest pair of points in the 3D scan can range from 0.5 mm to 1 mm, making this method suitable for high-density point clouds and less optimal for sparse ones.

To tackle this problem, an alternative solution is proposed is finding the three closest points in the sparse point cloud to the predicted coordinate and computing their centroid as the predicted landmark. Figure 3.12(a) demonstrates the Estimation model's prediction error from the facial

surface, Figure 3.12(b) illustrates the Refinement model's performance, and Figure 3.12(c) showcases the results after the proposed post-processing step.



(a) (b)

(c)

Figure 3.12. *Prediction Demonstration for the Estimation (a) and the Refinement (b) Model After Post-Processing (c)*

3.5. Evaluation criteria

The loss functions minimized during the training of the estimation and Refinement models are MAE and MSE, respectively. However, these metrics only represent the deviation of each axis of the predicted 3D coordinate from the actual ground truth. While suitable for achieving accurate predictions, they may not fully capture the details of facial landmark localization in the physical world, where precise measurements in millimeters (mm) are crucial.

To evaluate the models' performance in this task, the distance between the predicted landmark and the actual ground truth must be assessed in mm for the cases included in the test set. In the study by De Stefani et al. [44], the validation of the Vectra 3D imaging system was verified. This implies that the reference system in which the point clouds are represented in the 3D Cartesian coordinates with millimeter precision. In order to obtain the deviation of the predicted landmarks from their actual ground truth positions in mm, the Euclidean distance between them is calculated. Equation 3.1 demonstrates the formula for calculating the Euclidean distance (d) from the predicted $(pred)$ and the ground truth (GT) 3D coordinates.

Equation 3.1.

$$d = \sqrt{(x_{Pred} - x_{GT})^2 + (y_{Predicted} - y_{GT})^2 + (z_{Predicted} - z_{GT})^2}$$

4. Model Optimization

In this chapter, the focus is on evaluating the performance of the 3D facial landmark localization models demonstrated in the previous chapter, through a comprehensive analysis of various factors. The chapter begins with an in-depth analysis of the training behavior exhibited by the three utilized models, providing insight into their learning curves. subsequently, chapter proceeds with a thorough evaluation of the models based on the set of 50 anatomical facial landmarks, employing various preprocessing pipelines such as Morton code sorting, reference point-based sorting, sampling rates, and the inclusion of normal vectors as features. Additionally, the effectiveness of rigid and non-rigid data augmentation techniques is examined. The analysis encompasses both training and testing sets, guiding the selection of optimal configurations to enhance accuracy in facial landmark localization.

4.1. Model Behavior

In this section, the training behavior of the three models utilized in this study will be examined by analyzing their learning curves throughout each epoch. The learning curves demonstrated in this section were trained on point clouds, which underwent cropping process and resampling to 41,000 points, and Morton code sorting was applied to sort the indexes. The learning curve demonstrates the Mean Absolute Error of every epoch through training, which can be examined as an indicator of the models' performance and convergence.

Figure 4.1 illustrates the learning curve of the Estimation stage based on the loss function Mean Absolute Error (MAE) at each epoch for both the training set and the test set, indicating with a dashed line the epoch at which the model achieved its optimal performance. The X-axis represents the number of epochs, while the Y-axis represents the absolute value of deviation between the predicted value and the ground truth, considering mean value across all considered landmarks. The blue line represents the MAE for each epoch on the training set, while the orange line represents the MAE for each epoch on the test set. The best epoch corresponds to the point at which the lowest Mean Absolute Error (MAE) was attained. In our case, this occurs 15 epochs before the final epoch, as determined by the predefined number of epochs set for the early stopping criteria.

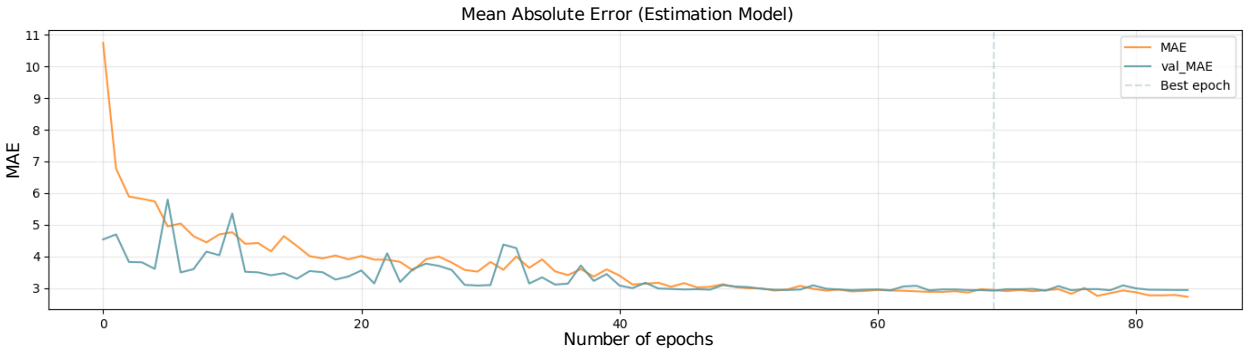


Figure 4.1. Learning curve of the Estimation model for Mean Absolute Error (MAE)

Figure 4.2. illustrates the Mean Absolute Error (MAE) of the Refinement stage for both the training and test sets across epochs, addressing the best performing epoch as well. However, the loss function utilized for the Refinement stage was MSE, but the MAE was monitored throughout the training process. Notably, due to the simplicity of the Refinement model, the training curve exhibits a shorter tail compared to the Estimation model.

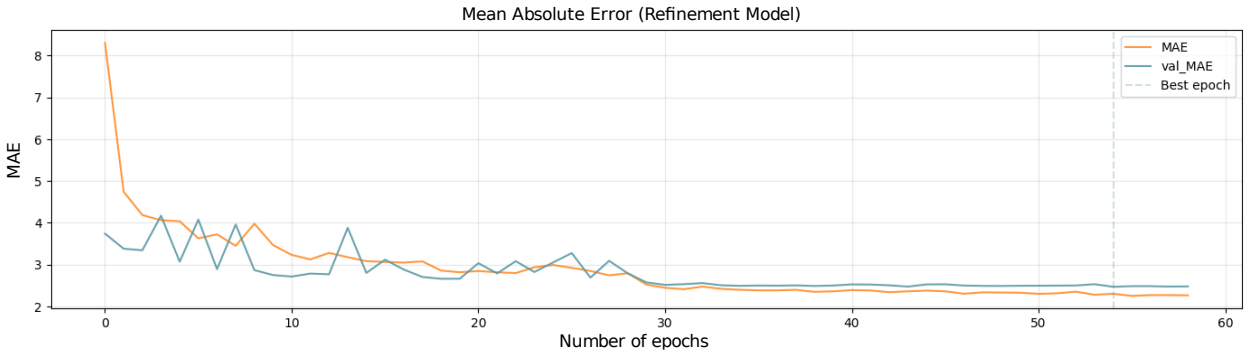


Figure 4.2. *Learning curve of the Refinement model for Mean Absolute Error (MAE).*

In this study, the PointNet architecture was employed for the task of 3D facial landmark localization. Both the classification and segmentation heads were utilized, separately, as feature extraction methods, followed by fully connected layers similar to those used in the Estimation and Refinement stages, to train and evaluate the architecture. However, the results obtained were not comparable with those achieved from the Estimation and Refinement models. The training process was lengthy and unstable, requiring significant computational resources. This is attributed to the fact that the PointNet architecture was originally developed for classification and segmentation tasks, where the number of points in the point cloud data is not as extensive as in the case of landmark localization using point clouds derived from stereophotogrammetry images. To leverage the novelty of the PointNet architecture for the task of facial landmark localization, the T-Net architecture was integrated within a distinct framework, which is this study is referred to the EsTi model.

The learning curve of the EsTi model is demonstrated in figure 4.3. Similar to the Estimation stage, the training was terminated after 79 epochs by utilizing the early stopping regularization, but the learning process was more computationally expensive, resulting in longer training time. Another attribute that can be observed from the plot is the model's unstable learning process at the initial stages of the training. The epoch that represents the sudden decrease of error in the plot corresponds to when the learning rate was decreased, showing improvement in the model's convergence and accuracy. Although the facial models were superimposed using the ICP registration algorithm, the T-Net model was able to account for the minor variations within the faces, resulting in superior performance compared to the Estimation model, which are further discussed in the next section.

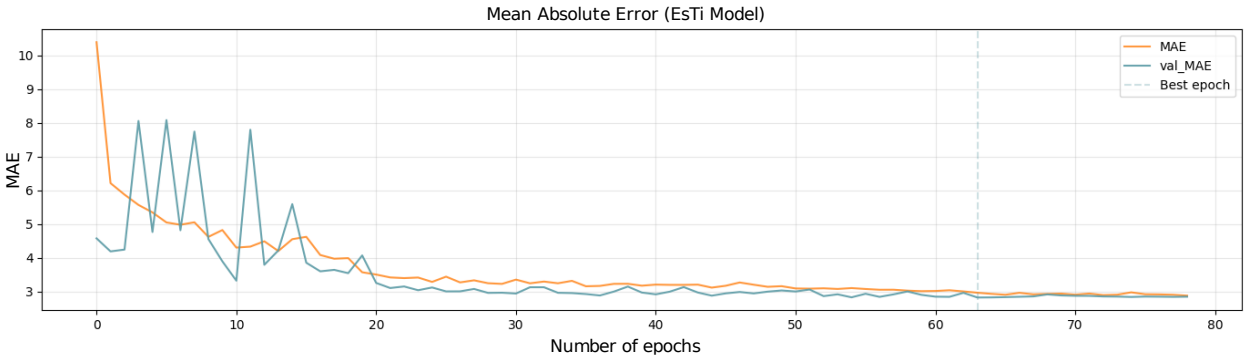


Figure 4.3. *Learning curve of the EsTi model for Mean Absolute Error (MAE)*

4.2. Model Evaluation

The facial landmarks utilized in this study consist of 50 anatomical facial landmarks provided in Table 3.1, where each landmark is categorized into different regions of the face. The landmarks are classified as either midline or paired, and each landmark is associated with a specific anatomical feature. Each of the 3D facial models was annotated with 50 3D coordinates corresponding to each of the landmarks. The dataset of the facial landmarks was split into an 80%-20% train and test set, where the proposed models in Chapter 3 were trained and evaluated based on their capabilities to localize the mentioned landmarks. Figure 3.1 showcases a 3D facial model represented as a point cloud with the anatomical facial landmarks identified.

The models were evaluated based on the disparity between their predicted coordinates and the manual annotation ground truth coordinates, that is calculated using the Euclidean distance metric. As discussed in section 3.5, the resulting Euclidean distance represents the disparity between the predictions and the ground truth in millimeters, providing an accurate evaluation criterion in the physical world as well. To ensure consistent and reproducible results throughout the training process, the GlorotNormal initialization method was utilized with a fixed seed, ensuring reproducibility of the results. By initializing the model weights with a fixed seed, randomness during training is eliminated, ensuring that the observed results are solely influenced by variations in the input data. This becomes particularly crucial when comparing model evaluations, such as the impact of incorporating vector normals data augmentation and comparing different sampling rates. In this section the models are evaluated on different preprocessing pipelines where in each section a different approach is adopted. At each section the best performing results are underlined.

4.2.1. Baseline (Morton Code)

As a baseline, considered as the default case in the previous section, the models were trained and evaluated on point cloud data, where each point only contained information regarding its coordinate in 3D space. For the baseline model, each point cloud, after the cropping process, was resampled to have 41,000 points and was indexed using the Morton code sorting method. Subsequently, without further processing, the sorted point clouds were used to train and evaluate the models, where each point cloud had the shape (41000, 3).

Table 4.1. showcases the results achieved by training and evaluating the models with the baseline preprocessing. Initially, the data was utilized to train and evaluate the Estimation model, and its predictions were used to train and evaluate the Refinement model. Furthermore, as a second approach, the baseline data was used to train and evaluate the EsTi model, which was then refined using the Refinement model. The models are evaluated based on their Mean Euclidean Distance (MED), which is calculated by averaging the mean displacement error between each of the predicted landmarks and their corresponding ground truth for the subjects in the training and the test set.

Table 4.1. *Mean Euclidean Distance Error for each of the Models for the Baseline Preproccesing Indexed with Morton Code Sorting*

	Model	***MED-train***	***MED-test***
Baseline *(Morton code sorting)*	Estimation:	4.0mm	5.2mm
	Refinement:	3.7mm	4.5mm
	EsTi:	4.3mm	5.0mm
	Refinement:	3.1mm	4.1mm

The Refinement model can improve the Estimation and the EsTi models, depending on whether the preprocessing steps align with the predictions from either the Estimation model or the EsTi model. Meaning, if the Refinement model is trained based on the Estimation model's prediction, it may not have an optimal result when tested on the data coming from the EsTi model's prediction. This shows that the Refinement model is able to capture the error patterns that come within each of the models' predictions and is able to compensate for the error by producing a more accurate prediction, hence producing a non-optimal result when a different model's prediction with different error patterns is used.

Considering the MED-test results shown in Table 4.1, EsTi model followed by Refinement model achieved a better performance in terms of Euclidean error with respect to the Estimation and Refinement. Figure 4.4. demonstrates the capability of the EsTi model and its Refinement model in localizing each of the 50 landmarks. The X-axis of the plot shows the name of the landmark, and the Y-axis represents the error in mm. The height of each bar represents the mean Euclidean error between the predicted landmark and its ground truth position across all subject in the test set. The red bars in the bar plot demonstrate the prediction error for the Estimation model, and the green bars demonstrate the prediction errors for the Refinement model.

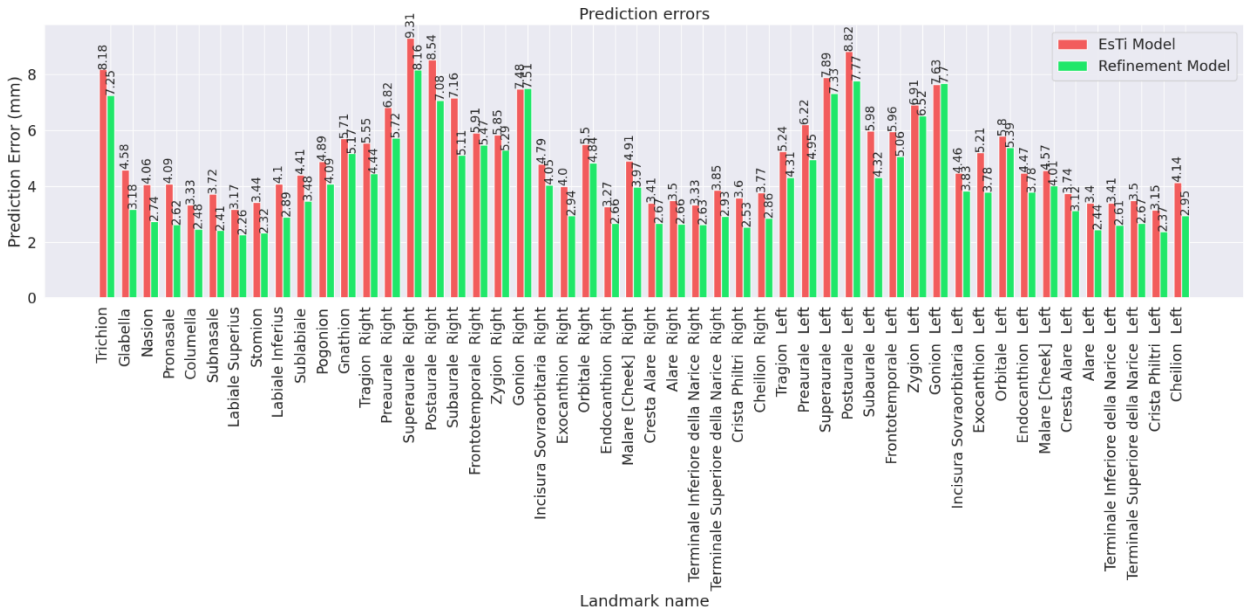


Figure 4.4. *Comparison of Prediction Errors for EsTi models and its Refinement Models across Landmarks*

As evident from the plot, both the EsTi model and its Refinement models exhibit a similar distribution of errors across landmarks, indicating that they capture the same underlying patterns. However, the Refinement model consistently outperforms the Estimation model, achieving lower prediction errors for all landmarks. This observation underscores the enhanced capacity of the Refinement model to capture finer and more detailed patterns.

4.2.2. Baseline (Sorting Based on a Reference Point)

In this section, the same preprocessing steps are taken as in the previous step, with the exception of the method used for indexing the points of the point cloud. In this preprocessing pipeline, after the cropping process and the resampling, the point clouds were sorted based on their distance from the origin of the reference system. The origin of the reference system was chosen as a reference point in space, which is situated at a slight distance from the tip of the nose, occupying the central position. This choice allows for a standardized reference point that aligns with the symmetry of the face. During experimental analysis, different points in space were chosen as a reference point, such as a point positioned above the forehead or below the chin, but the produced results did not show improvements compared to the reference point being the origin of the reference system. In another experimental analysis, the barycenter of the point cloud under examination was chosen as the reference point to index the point clouds, but the obtained results had reduced accuracy. This reduced accuracy can result from the variations in the distribution of the point clouds, where a point is indexed differently in different point clouds due to the nature of different head and face sizes. Notably, the indexing method is not only used after the resampling method; it is also utilized during the Refinement stage while preprocessing each of the sub-point clouds. Table 4.2. showcases the results achieved by training and evaluating the models with the baseline preprocessing with point of the point cloud being sorted based on a reference point.

Table 4.2. *Mean Euclidean Distance Error for each of the Models for the Baseline Preproccesing Indexed Based on a Reference Point*

	Model	***MED-train***	***MED-test***
Baseline *(Sorting based on the origin of the reference system)*	Estimation:	4.4mm	5.1mm
	Refinement:	3.4mm	4.2mm
	EsTi:	4.6mm	4.7mm
	Refinement:	2.8mm	3.9mm
Baseline *(Sorting based on a point above the forehead)*	Estimation:	4.2mm	4.9mm
	Refinement:	3.0mm	4.1mm
	EsTi:	4.8mm	4.8mm
	Refinement:	3.1mm	4.0mm

As demonstrated in table 4.2, indexing the points of the point clouds based on their distance from a reference point yields improved accuracy compared to Morton code sorting. An attribute that can be observed from the table is that predictions obtained from the Estimation stage for the point clouds that are indexed based on the forehead yield slightly more accurate localization compared to the prediction from the Estimation stage of the point clouds which are indexed according to the origin of the reference system. However, when the EsTi model is used and then refined, the point clouds that are indexed based on the origin of the reference system produce more accurate localization compared to the other indexing method. However, the variation in results in both preprocessing steps is small, but it can be explained that at each stage, the preprocessing method applied can enable the model to produce a more accurate localization.

4.2.3. Using Normal Vectors as a Feature

In this section, the models are trained and evaluated on the point clouds, where each point of the point cloud includes features from both its position in space and its vector normal. The vector normal of each point is computed using the method described in section 3.2.2, resulting in three values representing the vector normals for each axis. The same processing pipeline as discussed in the previous section is applied except for the additional process of computing the vector normals. After the cropping process, each point cloud is sampled to 41,000 points, then for each point of the point cloud, the vector normals are computed. Subsequently, the points of the point cloud with their corresponding vector normals are indexed based on their distance from the origin. As a final step, both the 3D coordinates and the vector normals are normalized to be within the same range of values and are then fed to the network. The resulting point clouds contain the sorted 3D coordinates and their corresponding vector normals in a multi-dimensional array with the shape (41,000,6).

A different training method was employed for the EsTi model when using point cloud data that included vector normals. As previously discussed, the T-Net architecture used in PointNet is

optimal for transforming point clouds to a canonical form, but it performs best when point cloud data containing only their 3D coordinates (41000,3) are inputted. In our case, the point cloud with vector normals (41000,6) was used, and when trained and evaluated on the EsTi model, the results achieved were suboptimal. To address this issue, a modification was made to the architecture of the EsTi model. The input point cloud (41000,6) was decomposed into two parts: 3D coordinates and vector normals, each with a shape of (41000,3). The part containing the 3D coordinates was fed into the T-Net architecture, which transformed them into a canonical presentation. The vector normals were then concatenated with the output of the T-Net architecture, resulting in a reconstructed canonical point cloud with their corresponding vector normals. This reconstructed canonical point cloud, along with the vector normals, followed the same path as the original Estimation model for further processing. Table 4.2 demonstrates the mean Euclidean distance error for the models that are trained and evaluated using the 3D coordinates and vector normals of each point of the point cloud.

***Table 4.3**. Mean Euclidean Distance Error for each of the Models for Point Clouds Incorporating Vector Normals*

	Model	***MED-train***	***MED-test***
Vector Normals	Estimation:	3.9mm	4.6mm
	Refinement:	3.0mm	4.1mm
	EsTi:	3.6mm	4.6mm
	Refinement:	3.3mm	4.3mm

From the obtained results in table 4.3, it can be observed that, unlike the previous preprocessing steps, the EsTi model does not outperform the Estimation model. Before the Refinement stage, both the Estimation and the EsTi models exhibit equal localization accuracy. However, during the refinement process, the Refinement stage of the Estimation model shows superior performance compared to the Refinement stage of the EsTi model. This difference in performance could be attributed to the complex behavior of the EsTi model incorporating vector normals, causing the preceding refinement stage to struggle in capturing and compensating for the error patterns originating from the EsTi model. While vector normals enhanced the Estimation model's performance compared to both baseline preprocessing steps (tables 4.1 and 4.2), their value as a feature for landmark localization appears to be diminished in models like the EsTi model when dealing with finer and more complex data.

4.2.4. Data Augmentation Evaluation

In this section, we apply both rigid and non-rigid data augmentation to the point clouds in the training dataset, following the methods described in Section 3.2.1. The aim is to enhance the localization accuracy of the models by improving their generalizability. In this section the point clouds are sorted based on their distance from the origin of the reference system, due to their superior localization accuracy compared to Morton code sorting. The models undergo separate training and evaluation processes for rigid and non-rigid data augmentation to understand which

type of data augmentation improves model performance. For rigid data augmentation, scaling is applied to uniformly change the size of the point clouds, using scaling factors ranging from 0.8 to 1.2. This results in a larger training set. In the case of non-rigid data augmentation, the same squeezing and stretching factors are applied to the X, Y, and Z axes of the point clouds randomly. However, to prevent significant deformation, squeezing and stretching are applied only to one axis at a time, meaning, if augmentation is applied to the X-axis, the Y and Z axes remain untouched. Additionally, to prevent significant deformation, small augmentation factors are used in both rigid and non-rigid data augmentation. The resulting training sets for both rigid and non-rigid data augmentation consist of 471 cases each, with the original testing set comprising 40 cases for both rigid and non-rigid data augmentation. Table 4.4. showcases the results achieved by training the point clouds with rigid and non-rigid data augmentation.

Table 4.4. *Mean Euclidean Distance Error for each of the Models Trained on Augmented Point Clouds*

	Model	***MED-train***	***MED-test***
Data Augmentation *Rigid*	Estimation:	3.6mm	4.7mm
	Refinement:	4.1mm	4.2mm
	EsTi:	3.4mm	4.9mm
	Refinement:	4.0mm	4.2mm
Data Augmentation *Non-rigid*	Estimation:	3.7mm	4.6mm
	Refinement:	2.8mm	3.9mm
	EsTi:	4.0mm	4.8mm
	Refinement:	2.9mm	4.0mm

From the results obtained in Table 4.4, it is evident that, for both cases of data augmentation, the EsTi model and its Refinement stage did not produce finer results compared to the Estimation and its Refinement stage, similar to the preprocessing with vector normals. This highlights the sensitivity of the T-Net architecture to the presence of diverse data in the training dataset. As discussed previously, the difference between the Estimation and the EsTi model is the presence of the T-Net architecture in the Estimation model. If the results obtained from the EsTi model are less accurate than the Estimation model, it is due to the effect of the T-Net architecture. In the case of non-rigid data augmentation, the model exhibits the lowest localization error for the Estimation and Refinement stage compared to the previous preprocessing steps, such as incorporating vector normals and different sorting methods. The obtained localization error with non-rigid data augmentation using the Estimation and its Refinement model, has the same localization error as using the Esti and its Refinement model without data augmentation as demonstrated in table 4.2. However, the results achieved are the same but in one of the cases a more complex model was utilized with less training data, while in the other case, a simpler model was utilized with a higher number of training sets. This demonstrates the effectiveness of data augmentation on point cloud data for the task of facial landmark localization, managing the tradeoff between model complexity and the volume of training data.

4.2.5. Re-Sampling Evaluation

In this section, different sampling rates are applied to resample the point clouds to reduce computational complexity and observe the model's ability to localize landmarks on sparser point clouds. Up to this point, all point clouds after cropping process were sampled to have 41,000 points. However, in this section, various values are used as sampling rates to understand the models' behavior on differently dense point clouds and find an optimal balance between accuracy and computational efficiency.

As discussed previously, depending on the number of points in the point cloud, modifications need to be made to the number of MaxPooling layers of the model to ensure optimal feature extraction. For instance, when each point cloud is sampled to have 41,000 points, four MaxPooling layers with window sizes of 4 and 5 are utilized, resulting in 102 points at the end of the feature extraction process. However, if the same number of MaxPooling layers is used when the point clouds are sampled to have 10,000 points, the output of the feature extraction process would be too small. Thus, to achieve optimal feature extraction, the MaxPooling layers and their window sizes must be adjusted according to the chosen sampling rate. Voxel sampling technique is employed in this study to sample the point clouds. This technique divides the point cloud into N 3D voxels, where N is the desired number of points in our point cloud. Voxel sampling is effective in our case because the resulting sampled point cloud tends to preserve both global and local features, as voxelization considers the distribution of points in the entire space. Table 4.5. presents the model evaluation on the training and test datasets for different sampling values.

Table 4.5. *Mean Euclidean Distance Error for the Models Trained on Point Clouds with Different Sampling Rate*

	Model	***MED-train***	***MED-test***
10,000 *Vertices*	Estimation:	4.4mm	5.5mm
	Refinement:	3.7mm	4.8mm
	EsTi:	4.4mm	5.1mm
	Refinement:	3.4mm	4.6mm
20,000 *Vertices*	Estimation:	4.3mm	5.2mm
	Refinement:	3.2mm	4.3mm
	EsTi:	3.9mm	4.8mm
	Refinement:	3.1mm	4.3mm
30,000 *Vertices*	Estimation:	4.1mm	5.0mm
	Refinement:	3.4mm	4.4mm
	EsTi:	3.6mm	4.7mm

	Refinement:	3.2mm	4.2mm
41,000 *Vertices*	Estimation:	4.4mm	5.1mm
	Refinement:	3.4mm	4.2mm
	EsTi:	4.6mm	4.7mm
	Refinement:	2.8mm	3.9mm

As evident from Table 4.5, the number of points in the point cloud directly influences the localization accuracy of the models. The table demonstrates the change in localization accuracy for every additional 10,000 points added. In almost all cases, the model exhibits improved localization accuracy with an increased number of vertices. However, these improvements are relatively small when considering the additional computational power required for automatic landmark localization on point clouds with more vertices. This correlation holds true for all the models, and it can be leveraged for further development of more complex models. Such as, during experimental analysis, point clouds with a low number of vertices can be used to optimize the model. Once an ideal model has been produced, it can be trained on point clouds with denser points to achieve the most optimal localization accuracy. It's worth noting that, depending on the hardware specifications available on the system where the models will be used, a less computationally expensive approach may be considered, with a relatively higher localization error.

5. Optimized Models & Discussion

In this chapter, an in-depth analysis of the two top-performing models is investigated, providing insight into their differences in terms of error distribution, residual standard deviation, and overall performance. The models exhibit a trade-off between complexity and dataset volume, with one model being more complex but trained on a smaller dataset, while another model being simpler but trained on a larger dataset. Despite both models achieving similar mean localization accuracy, an in-depth analysis reveals distinctions in prediction errors, robustness, and error distribution.

5.1. Best Model(s)

From the models trained and evaluated using various preprocessing methods, the two models with the best localization accuracy are reported in Table 5.1. The first model with the best localization accuracy is the Refined EsTi model trained on non-augmented point clouds, which were indexed based on their distance to the origin of the reference system. The second model with the best localization accuracy is the Refined Estimation model trained on non-rigid augmented point clouds, where each point is indexed based on its distance from the reference system. In the first case, a more complex model was utilized with a smaller volume of data, while in the second case, a much simpler model was used with a larger volume of data. This comparison highlights the importance of considering both model complexity and dataset volume in achieving optimal performance. The approaches utilized in the two models demonstrate the trade-off between model complexity and data volume, providing insights for further development.

Table 5.1. *Mean Euclidean Distance Error for the Models with the Best Localization Accuracy*

Preprocessing - Model	***MED-train***	***MED-test***
Baseline – sorting based on origin (EsTi Model-Refined)	2.89mm	3.98mm
Non-rigid Data Augmentation (Estimation Model-Refined)	2.87mm	3.98mm

Although both models have the same mean localization accuracy, to have a deeper understanding of how each of the models perform, further analysis is required. Figure 5.1. demonstrates the capability of the Refined EsTi model trained on non-augmented data, and the Refined Estimation model trained on augmented data in localizing each of the 50 landmarks from table 3.1. The X-axis of the plot shows the name of the landmark, and the Y-axis represents the error in mm. The height of each bar represents the disparity between the predicted landmarks. The bars in blue represent the localization accuracy for the Refined EsTi model with baseline preprocessing, to be termed the “EsTi model” for the remainder of this chapter; and the bars in green represent the Refined Estimation model trained on non-rigid data augmentation, to be denoted as the “Estimation model” for the remainder of this chapter.

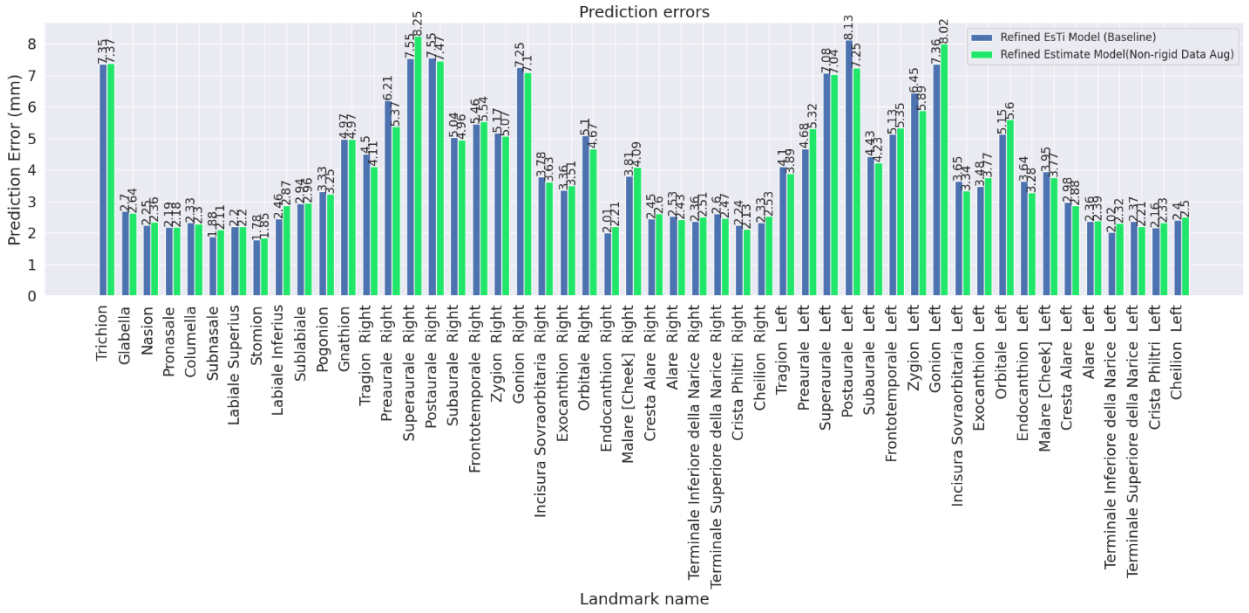


Figure 5.1. *Comparison of Prediction Errors for Two Models with the Best Localization Accuracy*

As evident from the plot, both models inherit the same error distribution across the landmarks. However, with a closer examination, for some landmarks such as ‘Postaurale’ left and right, the Estimation model outperforms the EsTi model, and vice versa for other cases, such as 'Gonion' and ‘Superaurale’. However, these variations in localization errors between the models are more noticeable for landmarks that inherit a higher degree of error, while in most cases, both models exhibit similar localization accuracy. Another observation is that both models have the lowest localization error for the landmark ‘Stomion’ with 1.78mm for the EsTi model and 1.85mm for the Estimation model. However, the landmark with the highest localization error for each model is different, with 8.13mm for the landmark ‘Postaurale left’ for the EsTi model and 8.25mm for the landmark ‘Superaurale right’ for the Estimation model. Nevertheless, the EsTi model resulted in lower values for both the highest and lowest localization errors in predicted landmarks compared to the Estimation model.

To further analyze the two models, figure 5.2. illustrates the histogram representing the distribution of errors in millimeters. Each bin in the histogram corresponds to a different range of errors, where the height of the bars represents the frequency or count of errors falling within that specific range across the cases in the testing set. The blue bars represent the frequency of errors for the EsTi model, and the green bars represent the error frequency for the Estimation model. The demonstrated histogram shows both models follow a similar normal distribution; although the EsTi model has a slight shift to the left with a longer tail to the right, while the Estimation model is shifted to the right with a longer tail to the left. Additionally, the EsTi model's histogram has a sharper and elevated bell shape characteristic of a normal distribution, highlighting its robustness compared to the Estimation model. Another observation from the plot is that the EsTi model tends to make frequent small errors, while the Estimation model makes less frequent larger errors, emphasizing the robustness of the EsTi model.

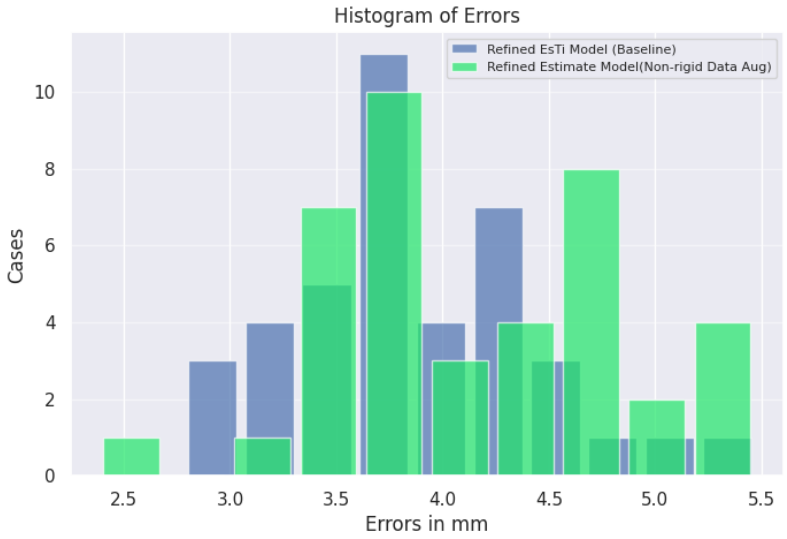


Figure 5.2. *Histogram of Prediction Errors for Two Models with the Best Localization Accuracy*

Another evaluation criterion that can be used for a more in-depth analysis of the models is to calculate the standard deviation (SD) of the localization error for each predicted landmark. Standard deviation measures the amount of variation in a given set of values from its mean and is expressed in the same units as the original data. In our case, standard deviation can be applied to the residuals of each predicted landmark from the test set, with its values serving as a criterion for assessing robustness. A low residual standard deviation for a landmark indicates that the model consistently predicts the positions of landmarks with small variations, reflecting a more robust localization accuracy. However, a higher standard deviation suggests that the model is more variable and less consistent in making accurate predictions, indicating a less robust performance.

The mean residual standard deviation is calculated by averaging the standard deviation of the errors made by model for each of the 50 landmarks. The localization error of each of the landmarks are calculated in mm for every case in the testing set, subsequently the standard deviation is calculated for every landmark's error set. Figure 5.3. demonstrates the residual standard deviation for each of the landmarks predicted by the EsTi and the Estimation model, and Table 5.2. shows the mean residual standard deviation for each of the models.

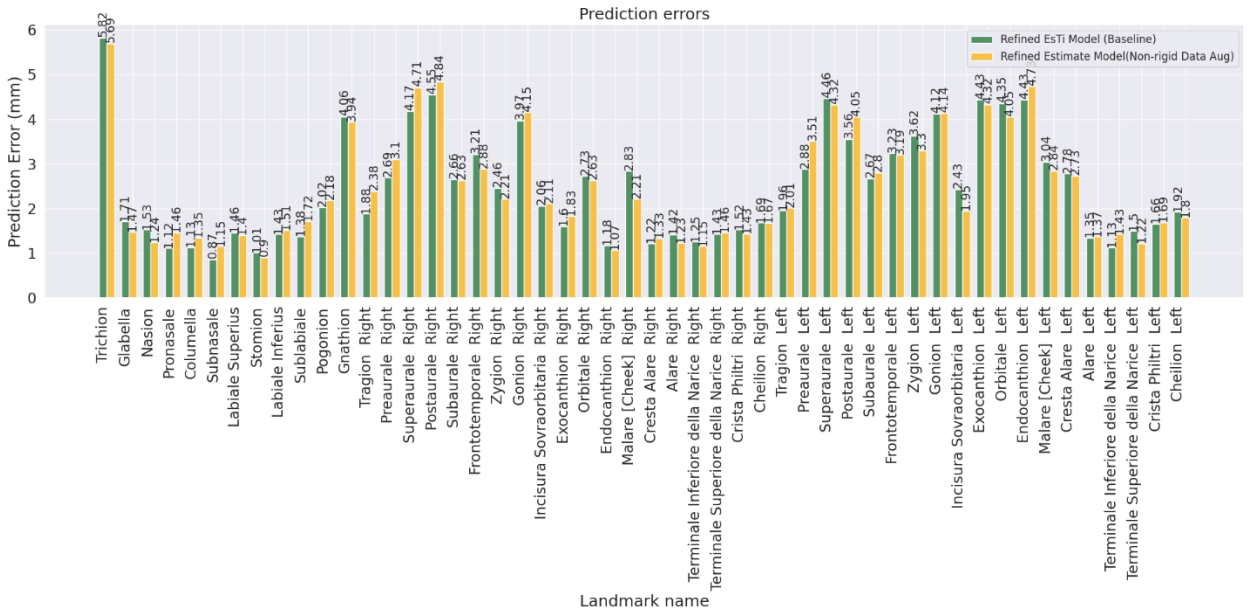


Figure 5.3. *Comparison of Residual Standard Deviation for Two Models with the Best Localization Accuracy*

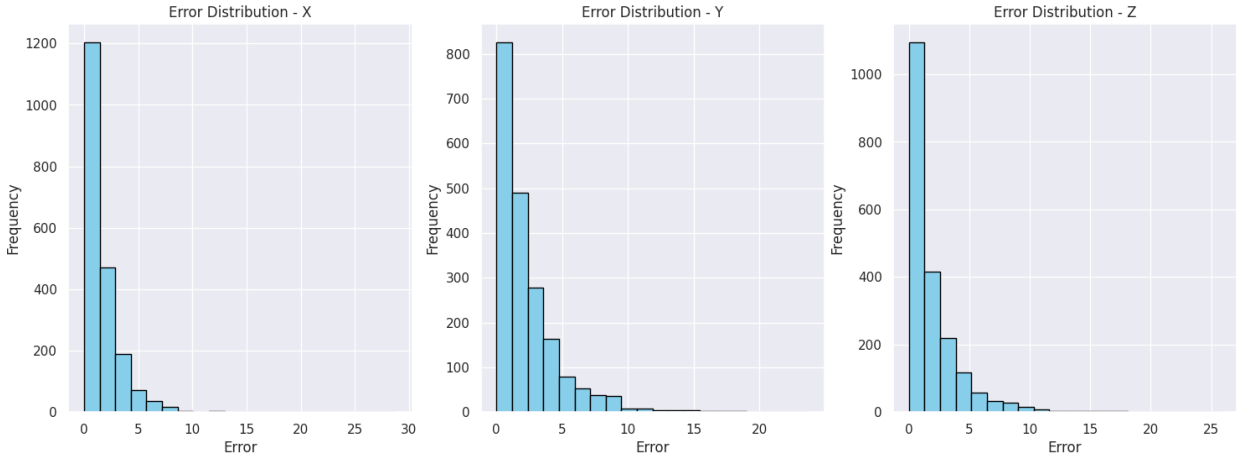
Table 5.2. *Residual Standard Deviation of the Models with the Best Localization Accuracy*

Preprocessing - Model	***Mean Residual SD-test***
Baseline – sorting based on origin (EsTi Model-Refined)	2.47mm
Non-rigid Data Augmentation (Estimation Model-Refined)	2.49mm

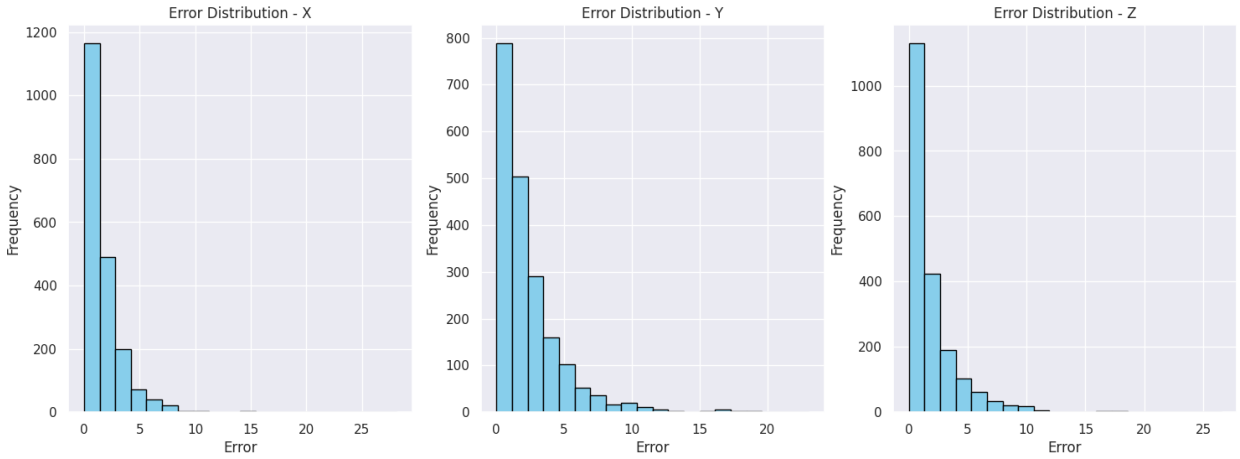
As depicted in Figure 5.3, both the EsTi and Estimation models showcase a similar distribution of residual standard deviation across the landmarks. Additionally, they share the same distribution as their mean Euclidean distance, as shown in Figure 5.1. However, notable finding from the provided graphs is the EsTi model's ability to achieve a lower residual standard deviation for certain landmarks, such as 'Subaurale' and 'Postaurale', despite having a higher mean Euclidean distance compared to the Estimation model. This underscores the robustness of the EsTi model compared to the Estimation model, as illustrated in Table 5.2. This characteristic of the models was also highlighted previously in Figure 5.2, where the EsTi model tends to make frequent small errors, resulting in a more normal distribution in histogram of errors, and a lower mean residual standard deviation, thus making it a preferable model over the Estimation model.

As discussed previously, the model aims to minimize the loss function (MAE or MSE) by adjusting its weights. The loss function is calculated based on the disparity between the predicted landmarks and their corresponding ground truth coordinates. Figure 5.4 demonstrates the histogram of the numerical differences between the predicted coordinates axes and their corresponding ground truth axes on the testing set. Figure 5.4(a) demonstrates the EsTi model and figure 5.4(b) demonstrates the Estimation model. At first glance, the errors in each axis have the same distribution, peaking at 0 value and having a tail on the right, which indicates the model's effectiveness in making

accurate predictions. However, for the histogram of the Y-axis, it has a shorter peak and incorporates a longer tail compared to the other distributions. This attribute was consistently present in all cases where the models were evaluated. It highlights that the model has more difficulty predicting an accurate value for the Y-axis. In our study, the X-axis represents the horizontal dimension of the face, Y-axis represents the vertical dimension of the face, and the Z-axis represents the depth dimension of the face, where the points on the Y-axis are distributed in a longer range, resulting in a higher error for localization compared to the other axes. This observation can be taken into consideration for further development of the model. Additionally, the distribution for the EsTi model has a higher peak around 0 for the X and the Y axis, compared to the Estimation mode. Although both models have the same localization accuracy, this attribute demonstrates the tendency of the Estimation model to have a broader spread of errors, as demonstrated in the residual standard deviation and in the histogram of errors in figure 5.2 and 5.3.



(a)



(b)

Figure 5.4. *Histogram of Prediction Errors for each of the X,Y, and Z axis*

While both models demonstrate similar localization accuracy on our testing set, a closer examination, including the computation of residual standard deviation and the analysis of the error histogram, reveals a slight difference. The EsTi model exhibits a more robust prediction,

characterized by a lower mean residual standard deviation and a more pronounced bell curve in the histogram of errors, as a result, the EsTi model achieves best performing model in this study. Figure 5.5 illustrates the point cloud of a 3D facial model, which was not present in the training not the testing set, displaying its annotated ground truth landmarks depicted in green, alongside the predicted landmarks by the EsTi model highlighted in red, providing a visual comparison between the actual and predicted positions.

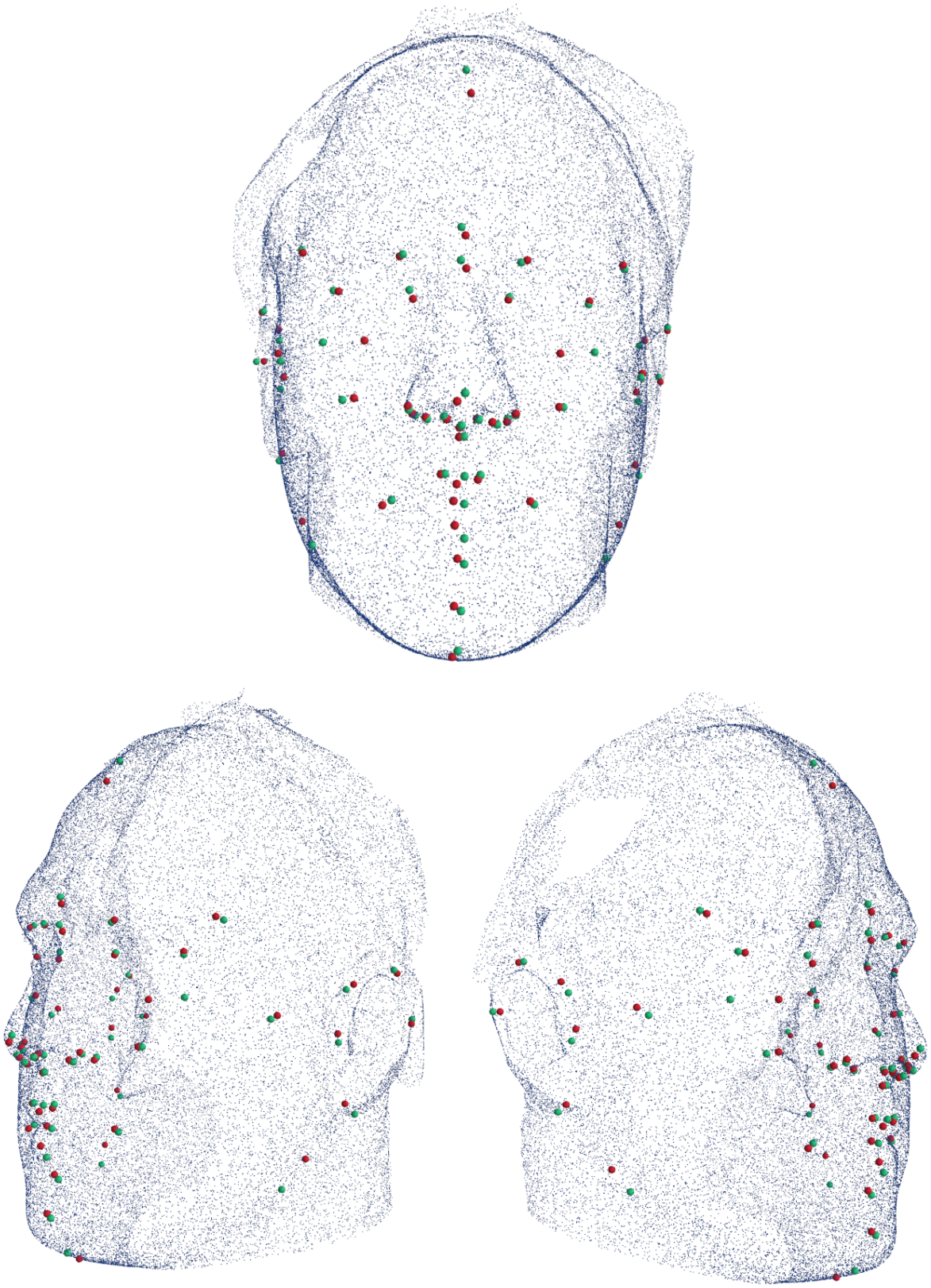


Figure 5.5. 3D Facial Model with Predicted Landmarks (red) and Annotated Ground Truth Landmarks (green)

5.2. Removal of Inconsistent Landmarks

This study aimed to develop a model for localizing 3D facial landmarks as accurately as possible. However, in the given dataset, which included 50 facial landmarks as shown in Figure 3.2, some of the landmarks are not located on the face but around the ears. As discussed previously, one of the limitations of stereophotogrammetry is its ineffectiveness in acquiring a fine mesh in places where hair is present. This limitation causes the points in the point cloud in the region around the ear to be hindered. As reported from LAFAS lab, the professional technician encounters challenges in localizing those landmarks manually, often due to missing or distorted points of the point cloud, resulting in inconsistent localization of those specific landmarks. The paired landmarks reported are, 'Preaurale', 'Superaurale', 'Postaurale', and 'Subaurale', which do not have a consistent localization. As shown in Figure 4.5, those landmarks correspond to the landmarks that inherit some of the highest degrees of error compared to the other landmarks. However, since these landmarks are being used clinically, they were not excluded from the training phase. Nevertheless, for a more robust technical evaluation of the models, these landmarks can be excluded from the evaluation criteria, resulting in a new adjusted evaluation metric that specifically assesses the performance of the models on a set of landmarks excluding those with inconsistent measurement. Table 4.8. shows the adjusted mean Euclidean distance, where the inconsistent landmarks were removed from the evaluation criteria.

Table 5.3. *Adjusted Mean Euclidean Distance Error for the Models with the Best Localization Accuracy*

Preprocessing - Model	***Adjusted MED-test***
Baseline – sorting based on origin (EsTi Model-Refined)	3.54mm
Non-rigid Data Augmentation (Estimation Model-Refined)	3.55mm

6. Conclusion and Future Work

6.1. Conclusion

This thesis aims to automate the annotation of 3D facial landmarks leveraging deep learning methods such as CNNs. The developed model, trained on a dataset of 200 stereophotogrammetry-acquired facial models and manually annotated by experts as the ground truth reference, demonstrates potential applications in orthodontics, maxillofacial, and aesthetic surgery. A set of 50 landmarks commonly used for facial anthropometric analysis was considered in this study. By reducing manual annotation time and ensuring a standardized approach, the tool enhances efficiency and precision in 3D anthropometric analysis. The work represents a significant step towards automating critical anatomical landmark identification, offering benefits in clinical and research settings. Previous studies on automatic landmark localization methods were investigated demonstrating effectiveness in predicting facial landmarks on 3D models. The progress in previous studies underscore the potential for adapting 2D methodologies to enhance precision in 3D facial landmark localization, opening different approaches for improved medical diagnostics and analysis.

Stereophotogrammetry, utilizing coordinated digital cameras and the stereoscopic principle, emerges as a promising approach for soft tissue analysis, offering advantages such as minimal acquisition time and high accuracy. This method facilitates the transformation of 3D facial models into point cloud data, representing object surfaces through 3D data points. This enables versatile operations and preprocessing techniques like the ICP algorithm and Morton code sorting, enhancing analysis capabilities and accuracy in facial feature localization.

ANNs and CNNs have been introduced, highlighting their significance in visual data processing. Key components such as convolutional and fully connected layers were discussed. PointNet was also introduced, showcasing its versatility in managing unordered point sets and maintaining invariance to permutations, making it a potent solution for point cloud processing. The incorporation of T-Net for pose normalization adds to its efficiency and robustness.

To feed the point cloud data to the models, data pre-processing was required in order to achieve optimal results. The data pre-processing stage of this study plays a crucial role in preparing the 3D facial models for subsequent analysis. The pipeline involves alignment, cropping, and resampling, addressing spatial and indexing challenges. Moreover, data augmentation, vector normals, and data splitting enhance model versatility and robustness. These pre-processing methods ensure a standardized, and well-prepared dataset for effective model training and evaluation. The model training employed a two-stage DL approach for precise facial landmark localization. The Estimation stage, utilizing CNNs, provided initial predictions, followed by the Refinement stage to enhance accuracy within a predefined radius of the Estimation stage's predictions. An additional EsTi model, incorporating the T-Net architecture was introduced, where further improved the Estimation stage. The models processed point clouds as multidimensional arrays, achieving predictions in the order of magnitude of millimeters. Evaluation was based on the Euclidean distance between predicted and ground truth landmarks, showcasing an effective facial landmark

localization method that combined advanced architectures and evaluation metrics for accurate results.

For each of the developed models an analysis of their training behavior was done, examining their learning curve through different epochs. Furthermore, assessment on the performance of facial landmark localization, with a focus on preprocessing variations were employed, utilizing 50 anatomical landmarks on 3D facial models. The evaluation includes baseline (Morton Code and reference point-based sorting), vector normals as features, data augmentation (rigid and non-rigid), and resampling techniques. The models, Estimation, Refinement, and EsTi, are measured based on Mean Euclidean Distance (MED) in both training and test sets. While certain methods enhance accuracy, the tradeoff with computational complexity is considered, emphasizing the need for a balanced approach in landmark localization models.

Finally, among the evaluated models for 3D facial landmark localization, two models, Esti and the Estimation model, stood out with similar mean localization accuracy. However, with closer examination revealed distinctions in robustness. The EsTi model exhibited a more normal distribution of errors, frequent small errors, and lower residual standard deviation, indicating superior consistency and reliability. Adjusted metrics, excluding inconsistent landmarks, further emphasized the EsTi model's performance. The insights presented in this thesis offer valuable guidance for future developments in the field of 3D facial landmark localization.

In the study by R. R. Paulsen et al [31], which adopted a multi-view approach for facial landmark identification, a localization error of 2.42mm was achieved on the BU-3DFE database [71]. While their study obtained a lower localization error compared to ours, with an adjusted accuracy of 3.54mm, it employed a more complex approach demanding significantly higher computational power. However, our study employed a simpler approach with lower computational requirements yet achieved comparable localization accuracy. It's noteworthy that in the study by R. R. Paulsen, the BU-3DFE database with its corresponding facial landmarks was used for training and evaluation, with all facial models cropped to contain only the face region. In contrast, our study utilized raw stereophotographs, introducing some limitations, which will be discussed in the next section.

6.2. Limitations and Future Work

Although the results achieved from this study are preliminary evaluations of a newly developed 3D automated landmarks identification approach for facial models, there is potential for further enhancement to meet the standards required for clinical application, such as a localization accuracy of less than 1mm. However, the methods and the techniques used in this study can be seen as a preliminary study for much broader research. The study was limited to only 200 facial models, while robust and accurate deep learning models typically require a larger dataset in the range of thousands, posing a constraint on the study. The absence of a dedicated validation set and reliance on the testing set for validation further emphasizes the need for expanded research in this domain.

Furthermore, a more robust registration algorithm can be introduced for aligning the point clouds. While ICP is effective in most cases, as discussed previously it has some limitations which can impact the final result significantly. The ICP algorithm is sensitive to initial guess and requires a good initial positioning of the point clouds, otherwise the algorithm will converge in local minima. Due to the nature of our study, a good initial positioning of the point clouds cannot be guaranteed, resulting in some point clouds not aligning optimally. Another constraint of the ICP registration algorithm is its sensitivity to outliers, which highlights another limitation of this work, that is the cropping method utilized. In this study, the cropping is applied by using a frame. If the point of the point clouds fell within the frame they were kept while otherwise they would be removed. Although this method proved beneficial in this study, it is not an optimal method. The limitations associated with the ICP algorithm, and the cropping process are interconnected, meaning if the first alignment is not optimized, it can adversely affect subsequent cropping, and similarly, inefficient cropping can compromise the effectiveness of the second registration process. Figure 6.1. represents a case where was removed from the training dataset due to having an outlier and failing in the registration process. Ideally the two faces should be superimposed on each other, but due to existence of outliers in the orange point cloud, both the ICP algorithm and the cropping process failed to operate optimally.

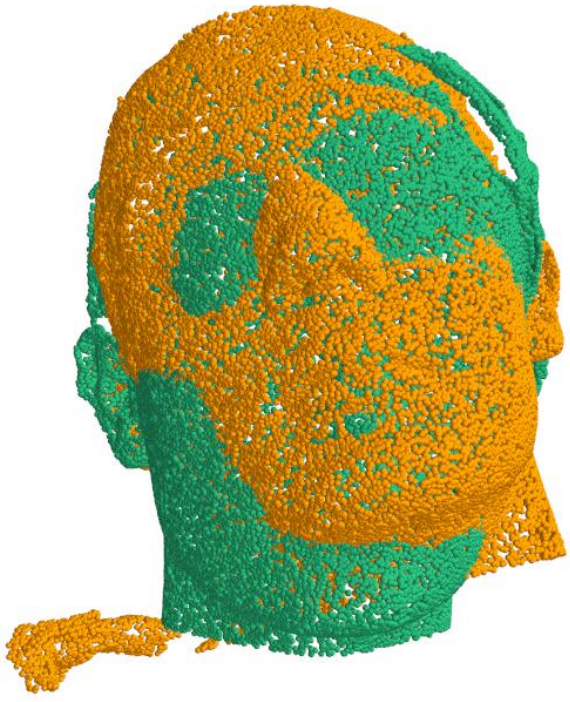


Figure 6.1. *An Example of a Failed Registration Process, Due to Existence of Outliers*

This demonstrated the sensitivity of the developed model to outliers. However, to mitigate this issue, various registration algorithms [72] [73] [74] and cropping techniques, such as segmentation models, can be utilized. In the study of Bai et al. [74] a robust point cloud registration algorithm is developed using deep spatial consistency featuring outlier rejection. Such models can be

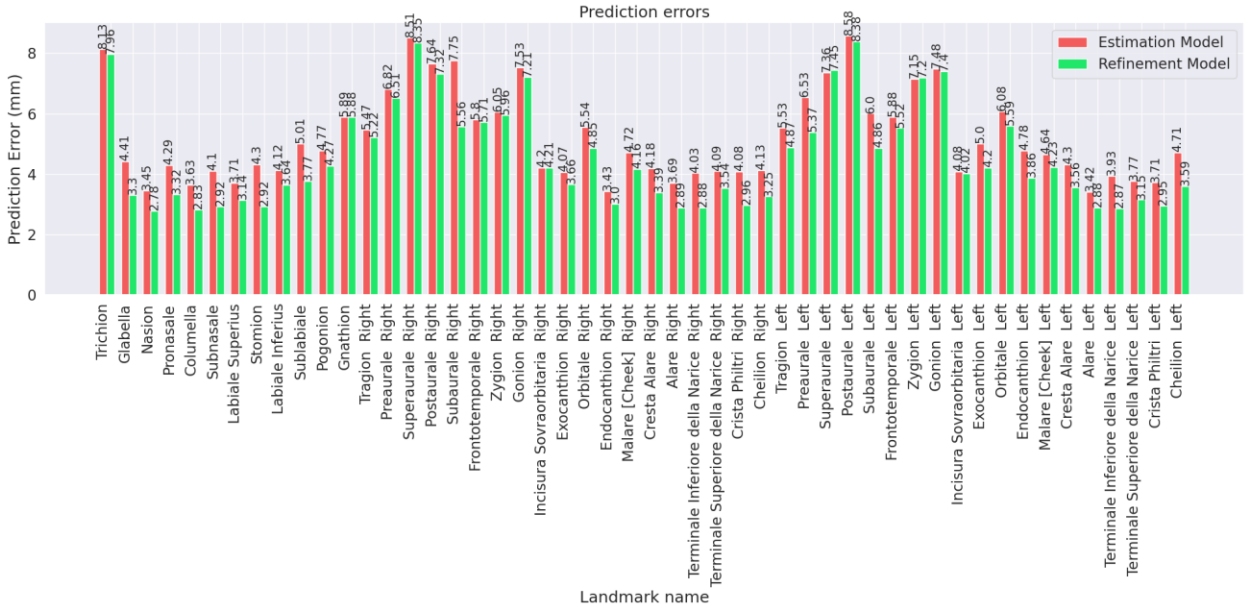
introduced in the pipeline, replacing the ICP algorithm for a more robust outcome. Many studies have been done on 2D face segmentation [75] [76] [77], however, the literature on 3D face segmentation remains relatively limited, with only a few studies conducted[78], while custom models and networks, such as PointNet [61] and PointNet ++ [17], can be extended for 3D face segmentation as a cropping methods, this highlights the need for further exploration and investigation in the realm of 3D face segmentation.

While this study did not consider the texture of the face in the point clouds, future research could incorporate RGB features based on the position of each point. This addition would assign an RGB value corresponding to the skin color at each point's location within the point clouds. Incorporating color information in this manner could enhance the model's ability to capture and process both geometric and chromatic aspects of facial surfaces, potentially improving the accuracy and robustness of facial landmark localization models.

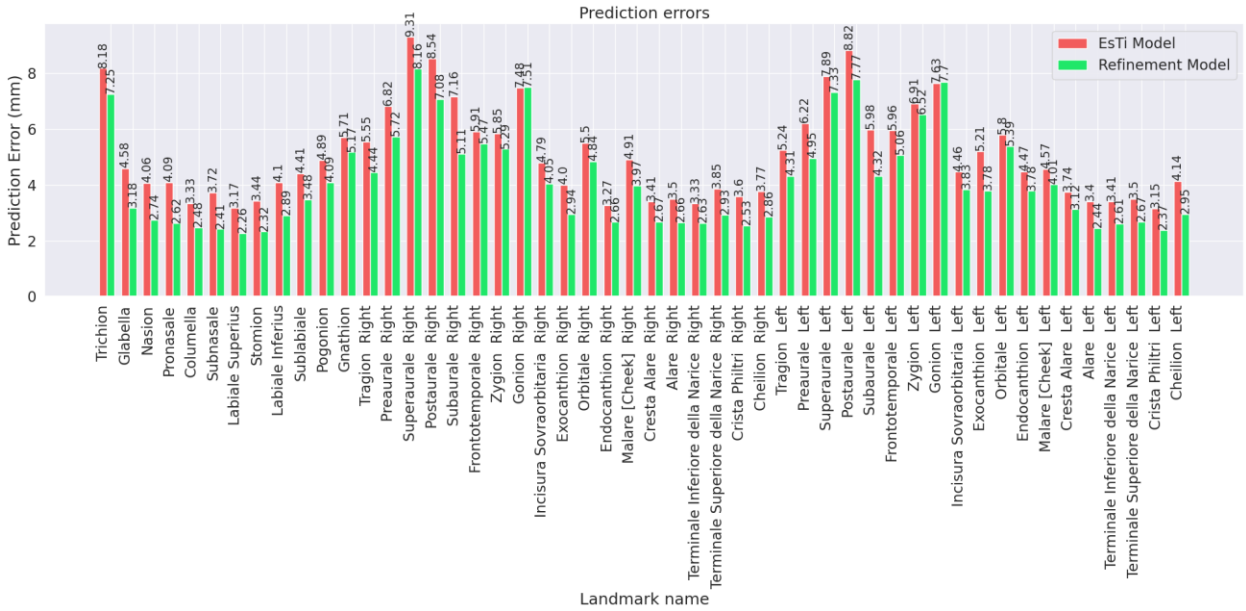
Additionally, further improvements can be made on the task of automatic 3D facial landmark localization by utilizing Graph Convolutional Neural Networks (GCNNs), which have been widely used in various point cloud processing tasks [79] [80] [81] [82]. GCNNs operate by representing the points within a point cloud as nodes in a graph. Where each node encapsulates information like 3D coordinate, vector normals, RGB features, and other features. The model then employs graph convolution operations to extract features, capturing both local and global relationships among the points. As discussed previously in some of the cases the predicted landmarks were not localized on the surface of the face, and further post-processing was required. This method can be avoided by using GCNNs, where the prediction of the model would be a node of the graph instead of continuous values. Using GCNNs can reduce complexity of the models and potentially enhance the localization accuracy.

Finally, to enable a comprehensive comparison with other studies, the proposed methodology can be evaluated on diverse facial databases and landmarks, including the BU-3DFE database. Furthermore, the feature extraction blocks of the model hold the potential to be trained on various 3D facial databases [71] [31] [83], allowing them to capture facial features for different tasks. This versatility enables the model to perform tasks like facial detection and anatomical or non-anatomical facial landmark localization without the need for extensive training data.

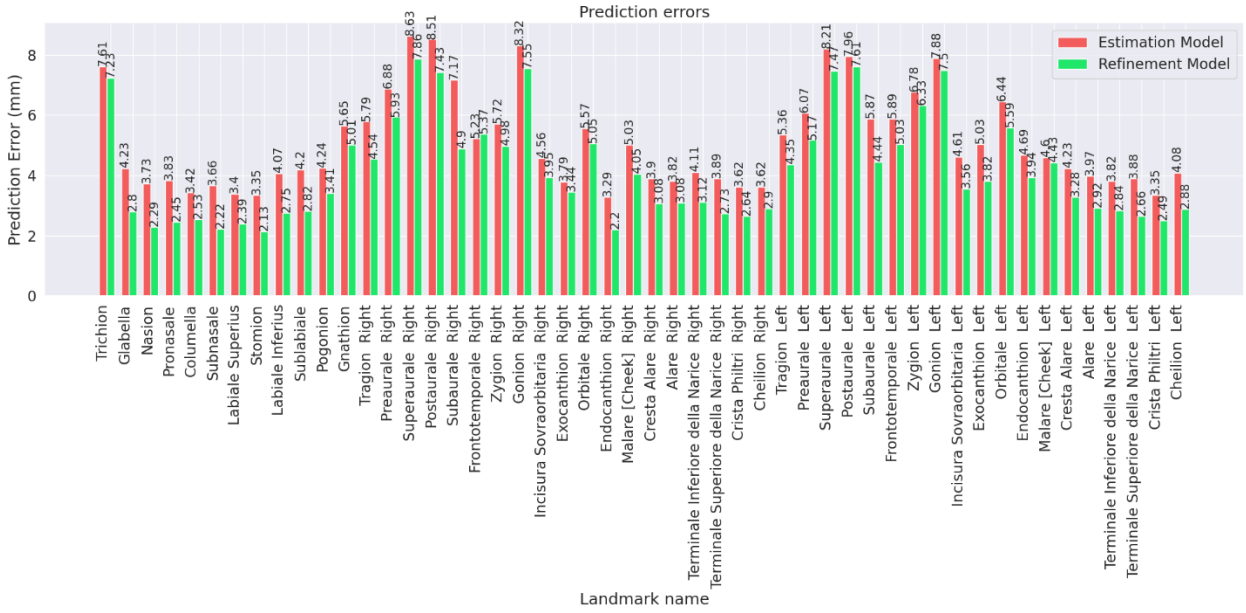
Appendix



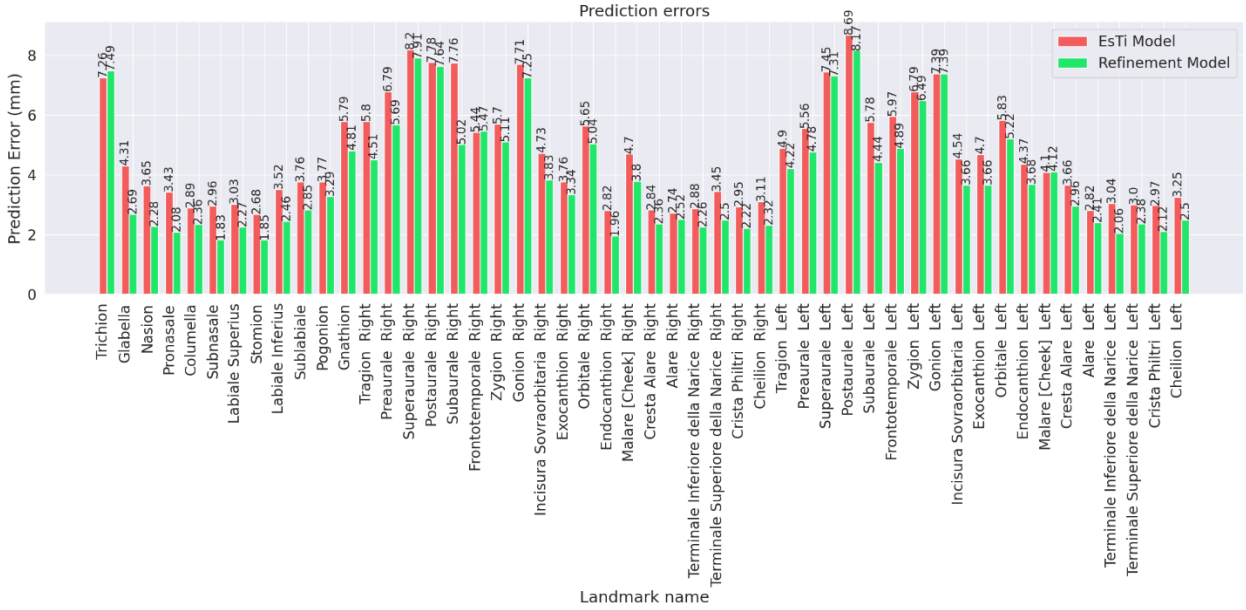
Appendix 1. *Comparison of Prediction Errors for Estimation Models and its Refinement Models for the baseline with Morton code indexing.*



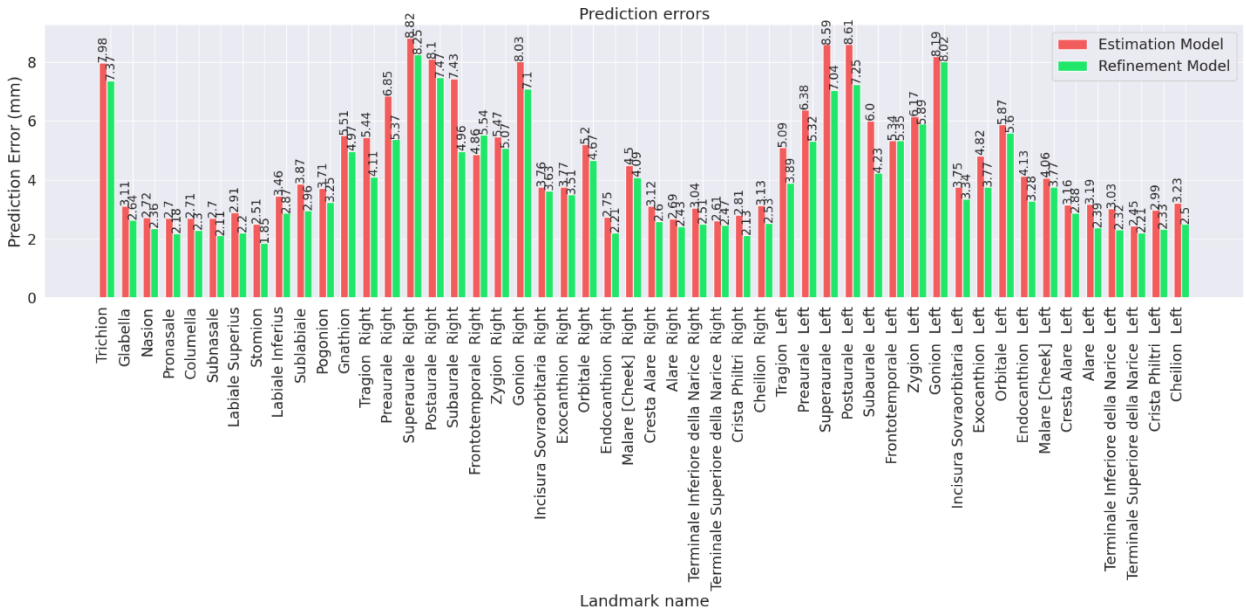
Appendix 2. *Comparison of Prediction Errors for EsTi models and its Refinement Models for the baseline with Morton code indexing.*



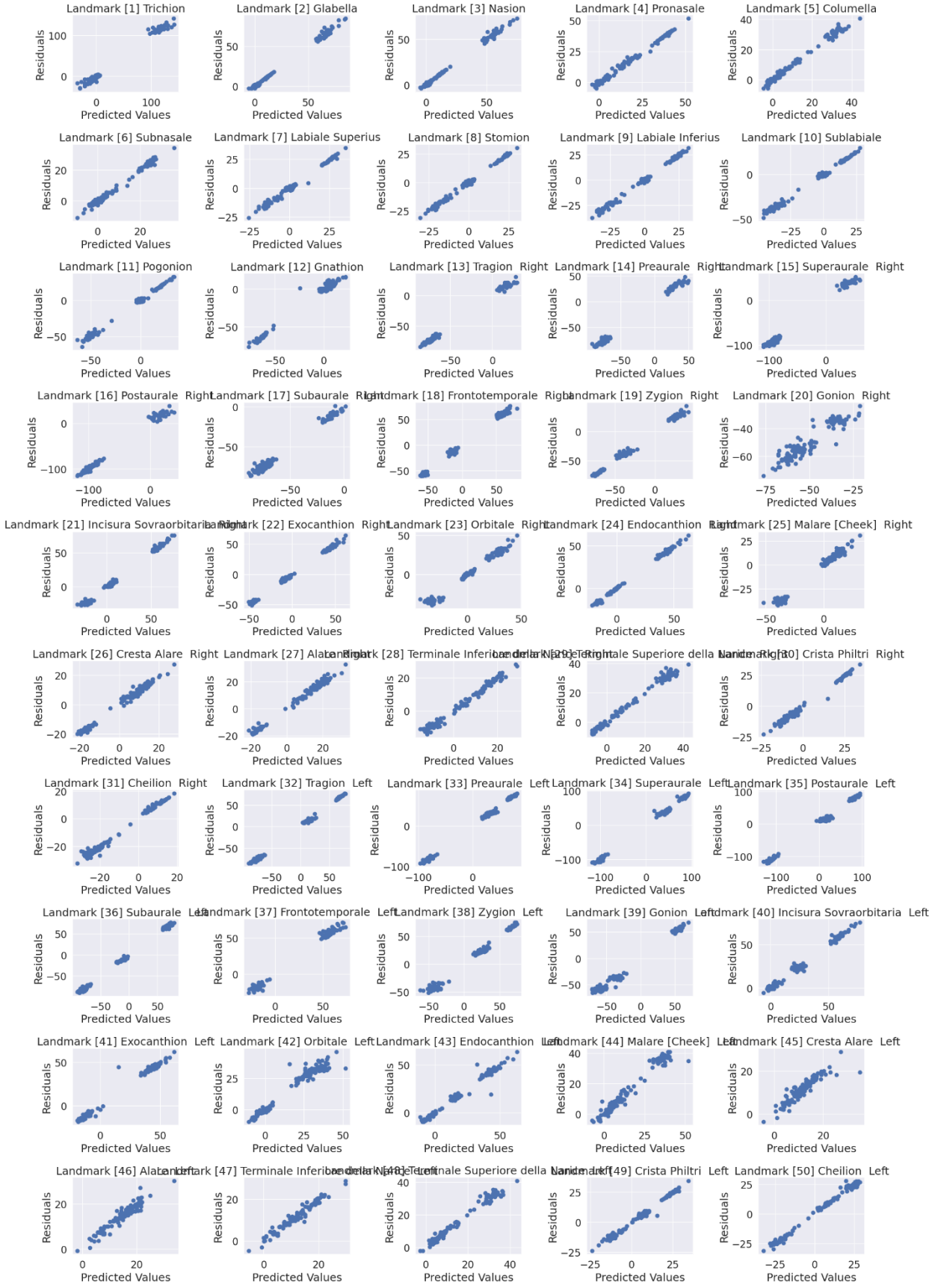
Appendix 3. *Comparison of Prediction Errors for Estimation Models and its Refinement Models for the baseline with indexing based on the origin of the reference system.*



Appendix 4. *Comparison of Prediction Errors for EsTi models and its Refinement Models for the baseline with indexing based on the origin of the reference system.*



Appendix 5. *Comparison of Prediction Errors for Estimation Models and its Refinement Models for data with non-rigid data augmentation.*


Landmark [1] Trichion
Landmark [2] Glabella
Landmark [3] Nasion
Landmark [4] Pronasale
Landmark [5] Columella
Landmark [6] Subnasale
Landmark [7] Labiale Superius
Landmark [8] Stomion
Landmark [9] Labiale Inferius
Landmark [10] Sublabiale
Landmark [11] Pogonion
Landmark [12] Gnathion
Landmark [13] Tragion Right
Landmark [14] Preaurale Right
Landmark [15] Superaurale Right
Landmark [16] Postaurale Right
Landmark [17] Subaurale Right
Landmark [18] Frontotemporale Right
Landmark [19] Zygion Right
Landmark [20] Gonion Right
Landmark [21] Incisura Sovraorbitaria Right
Landmark [22] Exocanthion Right
Landmark [23] Orbitale Right
Landmark [24] Endocanthion Right
Landmark [25] Malare [Cheek] Right
Landmark [26] Cresta Alare Right
Landmark [27] Alare Right
Landmark [28] Terminale Inferiore della Narice Right
Landmark [29] Terminale Superiore della Narice Right
Landmark [30] Crista Philtri Right
Landmark [31] Cheilion Right
Landmark [32] Tragion Left
Landmark [33] Preaurale Left
Landmark [34] Superaurale Left
Landmark [35] Postaurale Left
Landmark [36] Subaurale Left
Landmark [37] Frontotemporale Left
Landmark [38] Zygion Left
Landmark [39] Gonion Left
Landmark [40] Incisura Sovraorbitaria Left
Landmark [41] Exocanthion Left
Landmark [42] Orbitale Left
Landmark [43] Endocanthion Left
Landmark [44] Malare [Cheek] Left
Landmark [45] Cresta Alare Left
Landmark [46] Alare Left
Landmark [47] Terminale Inferiore della Narice Left
Landmark [48] Terminale Superiore della Narice Left
Landmark [49] Crista Philtri Left
Landmark [50] Cheilion Left
Residuals
Predicted Values

Appendix 6. *Residual plot of the best model's prediction.*

The X-axis of the plot presents the predicted value for the X,Y and Z axis of the point clouds, and the Y-axis represents their actual ground truth value. The distribution of the data is closer to a identity line, the more accurate the localization is, and the more the data points are scattered in space the less accurate it is. As can be seen the landmark 'Gonion Right' has more of a scattered distribution compared to other landmarks, and subsequently in figure 4.5 it corresponds to one of the landmarks with the highest localization error.

Bibliography

[1] M. S. See, C. Roberts, and C. Nduka, “Age- and Gravity-Related Changes in Facial Morphology: 3-Dimensional Analysis of Facial Morphology in Mother-Daughter Pairs,” *J. Oral Maxillofac. Surg.*, vol. 66, no. 7, pp. 1410–1416, Jul. 2008, doi: 10.1016/j.joms.2007.12.041.

[2] C. Baserga *et al.*, “Efficacy of Autologous Fat Grafting in Restoring Facial Symmetry in Linear Morphea-Associated Lesions,” *Symmetry*, vol. 12, no. 12, p. 2098, 2020.

[3] V. F. Ferrario, C. Sforza, J. H. Schmitz, and F. Santoro, “Three-dimensional facial morphometric assessment of soft tissue changes after orthognathic surgery,” *Oral Surg. Oral Med. Oral Pathol. Oral Radiol. Endodontology*, vol. 88, no. 5, pp. 549–556, 1999.

[4] J. B. Chang, K. H. Small, M. Choi, and N. S. Karp, “Three-Dimensional Surface Imaging in Plastic Surgery: Foundation, Practical Applications, and Beyond,” *Plast. Reconstr. Surg.*, vol. 135, no. 5, p. 1295, May 2015, doi: 10.1097/PRS.0000000000001221.

[5] S. Masnada *et al.*, “3D facial morphometry in Italian patients affected by Aicardi syndrome,” *Am. J. Med. Genet. A.*, vol. 182, no. 10, pp. 2325–2332, 2020.

[6] C. Dolci *et al.*, “The face in marfan syndrome: A 3D quantitative approach for a better definition of dysmorphic features,” *Clin. Anat.*, vol. 31, no. 3, pp. 380–386, 2018.

[7] J. Allanson, P. O’Hara, L. Farkas, and R. Nair, “Anthropometric craniofacial pattern profiles in Down syndrome,” *Am. J. Med. Genet.*, vol. 47, no. 5, pp. 748–752, 1993.

[8] J. E. Allanson and R. C. M. Hennekam, “Rubinstein-Taybi syndrome: Objective evaluation of craniofacial structure,” *Am. J. Med. Genet.*, vol. 71, no. 4, pp. 414–419, Sep. 1997, doi: 10.1002/(SICI)1096-8628(19970905)71:4<414::AID-AJMG8>3.0.CO;2-T.

[9] J. E. Allanson and T. R. P. Cole, “Sotos syndrome: Evolution of facial phenotype subjective and objective assessment,” *Am. J. Med. Genet.*, vol. 65, no. 1, pp. 13–20, Oct. 1996, doi: 10.1002/(SICI)1096-8628(19961002)65:1<13::AID-AJMG2>3.0.CO;2-Z.

[10] X. Dong, S.-I. Yu, X. Weng, S.-E. Wei, Y. Yang, and Y. Sheikh, “Supervision-by-registration: An unsupervised approach to improve the precision of facial landmark detectors,” presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 360–368.

[11] A. Cappella *et al.*, “Facial Asymmetry of Italian Children: A Cross-Sectional Analysis of Three-Dimensional Stereophotogrammetric Reference Values,” *Symmetry*, vol. 15, no. 4, p. 792, 2023.

[12] K. Kočandrlová, J. Dupej, E. Hoffmannová, and J. Velemínská, “Three-dimensional mixed longitudinal study of facial growth changes and variability of facial form in preschool children using stereophotogrammetry,” *Orthod. Craniofac. Res.*, vol. 24, no. 4, pp. 511–519, 2021, doi: 10.1111/ocr.12461.

[13] D. Gibelli, C. Dolci, A. Cappella, and C. Sforza, “Reliability of optical devices for three-dimensional facial anatomy description: a systematic review and meta-analysis,” *Int. J. Oral Maxillofac. Surg.*, vol. 49, no. 8, pp. 1092–1106, Aug. 2020, doi: 10.1016/j.ijom.2019.10.019.

[14] V. F. Ferrario, C. Sforza, G. Serrao, V. Ciusa, and C. Dellavia, “Growth and aging of facial soft tissues: A computerized three-dimensional mesh diagram analysis,” *Clin. Anat.*, vol. 16, no. 5, pp. 420–433, 2003, doi: 10.1002/ca.10154.

[15] B.-S. Hua, M.-K. Tran, and S.-K. Yeung, “Pointwise Convolutional Neural Networks,” presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 984–993. Accessed: Jan. 17, 2024. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2018/html/Hua_Pointwise_Convolutional_Neural_CVPR_2018_paper.html

[16] A. Boulch, “ConvPoint: Continuous convolutions for point cloud processing,” *Comput. Graph.*, vol. 88, pp. 24–34, May 2020, doi: 10.1016/j.cag.2020.02.005.

[17] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, “PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space,” in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2017. Accessed: Jan. 16, 2024. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/hash/d8bf84be3800d12f74d8b05e9b89836f-Abstract.html

[18] K. Schwenzer-Zimmerer *et al.*, “Quantitative 3D soft tissue analysis of symmetry prior to and after unilateral cleft lip repair compared with non-cleft persons (performed in Cambodia),” *J. Cranio-Maxillofac. Surg.*, vol. 36, no. 8, pp. 431–438, Dec. 2008, doi: 10.1016/j.jcms.2008.05.003.

[19] M. de Menezes, R. Rosati, V. F. Ferrario, and C. Sforza, “Accuracy and Reproducibility of a 3-Dimensional Stereophotogrammetric Imaging System,” *J. Oral Maxillofac. Surg.*, vol. 68, no. 9, pp. 2129–2135, Sep. 2010, doi: 10.1016/j.joms.2009.09.036.

[20] S. M. Weinberg, N. M. Scott, K. Neiswanger, C. A. Brandon, and M. L. Marazita, “Digital Three-Dimensional Photogrammetry: Evaluation of Anthropometric Precision and Accuracy Using a Genex 3D Camera System,” *Cleft Palate Craniofacial J.*, vol. 41, no. 5, pp. 507–518, Sep. 2004, doi: 10.1597/03-066.1.

[21] C. Sforza, C. Dellavia, M. De Menezes, R. Rosati, and V. F. Ferrario, “Three-dimensional facial morphometry: from anthropometry to digital morphology,” in *Handbook of Anthropometry: Physical Measures of Human Form in Health and Disease*, Springer, 2012, pp. 611–624.

[22] C. Lane and W. Harrell, “Completing the 3-dimensional picture,” *Am. J. Orthod. Dentofacial Orthop.*, vol. 133, no. 4, pp. 612–620, Apr. 2008, doi: 10.1016/j.ajodo.2007.03.023.

[23] C. L. Heike, M. L. Cunningham, A. V. Hing, E. Stuhaug, and J. R. Starr, “Picture Perfect? Reliability of Craniofacial Anthropometry Using Three-Dimensional Digital Stereophotogrammetry,” *Plast. Reconstr. Surg.*, vol. 124, no. 4, p. 1261, Oct. 2009, doi: 10.1097/PRS.0b013e3181b454bd.

[24] G. de Jong *et al.*, “Combining deep learning with 3D stereophotogrammetry for craniosynostosis diagnosis,” *Sci. Rep.*, vol. 10, no. 1, Art. no. 1, Sep. 2020, doi: 10.1038/s41598-020-72143-y.

[25] R. E. Bristol, G. P. Lekovic, and H. L. Rekate, “The effects of craniosynostosis on the brain with respect to intracranial pressure,” *Semin. Pediatr. Neurol.*, vol. 11, no. 4, pp. 262–267, Dec. 2004, doi: 10.1016/j.spen.2004.11.001.

[26] A. Esteva *et al.*, "Dermatologist-level classification of skin cancer with deep neural networks," *Nature*, vol. 542, no. 7639, Art. no. 7639, Feb. 2017, doi: 10.1038/nature21056.

[27] A. Shuper, P. Merlob, M. Grunebaum, and S. H. Reisner, "The Incidence of Isolated Craniosynostosis in the Newborn Infant," *Am. J. Dis. Child.*, vol. 139, no. 1, pp. 85–86, Jan. 1985, doi: 10.1001/archpedi.1985.02140030091038.

[28] Y. Wu and Q. Ji, "Facial Landmark Detection: A Literature Survey," *Int. J. Comput. Vis.*, vol. 127, no. 2, pp. 115–142, Feb. 2019, doi: 10.1007/s11263-018-1097-z.

[29] B. Johnston and P. de Chazal, "A review of image-based automatic facial landmark identification techniques," *EURASIP J. Image Video Process.*, vol. 2018, no. 1, p. 86, Sep. 2018, doi: 10.1186/s13640-018-0324-4.

[30] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, "Facial Landmark Detection by Deep Multi-task Learning," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2014, pp. 94–108. doi: 10.1007/978-3-319-10599-4_7.

[31] R. R. Paulsen, K. A. Juhl, T. M. Haspang, T. Hansen, M. Ganz, and G. Einarsson, "Multi-view Consensus CNN for 3D Facial Landmark Placement," in *Computer Vision – ACCV 2018*, C. V. Jawahar, H. Li, G. Mori, and K. Schindler, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, pp. 706–719. doi: 10.1007/978-3-030-20887-5_44.

[32] D. Maturana and S. Scherer, "VoxNet: A 3D Convolutional Neural Network for real-time object recognition," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2015, pp. 922–928. doi: 10.1109/IROS.2015.7353481.

[33] E. O' Sullivan, "Extending Convolutional Pose Machines for Facial Landmark Localization in 3D Point Clouds," presented at the Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops, 2019, pp. 0–0. Accessed: Jan. 16, 2024. [Online]. Available: https://openaccess.thecvf.com/content_ICCVW_2019/html/PreReg/Sullivan_Extending_Convolutional_Pose_Machines_for_Facial_Landmark_Localization_in_3D_ICCVW_2019_paper.html

[34] C. Creusot, N. Pears, and J. Austin, "A Machine-Learning Approach to Keypoint Detection and Landmarking on 3D Meshes," *Int. J. Comput. Vis.*, vol. 102, no. 1, pp. 146–179, Mar. 2013, doi: 10.1007/s11263-012-0605-9.

[35] A. Newell, K. Yang, and J. Deng, "Stacked Hourglass Networks for Human Pose Estimation," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2016, pp. 483–499. doi: 10.1007/978-3-319-46484-8_29.

[36] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981, doi: 10.1145/358669.358692.

[37] X. Dong, Y. Yan, W. Ouyang, and Y. Yang, "Style Aggregated Network for Facial Landmark Detection," presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 379–388. Accessed: Jan. 16, 2024. [Online]. Available:

https://openaccess.thecvf.com/content_cvpr_2018/html/Dong_Style_Aggregated_Network_CVPR_2018_paper.html

[38] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional Pose Machines," presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 4724–4732. Accessed: Jan. 16, 2024. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2016/html/Wei_Convolutional_Pose_Machines_CVPR_2016_paper.html

[39] V. Ramakrishna, D. Munoz, M. Hebert, J. Andrew Bagnell, and Y. Sheikh, "Pose Machines: Articulated Pose Estimation via Inference Machines," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2014, pp. 33–47. doi: 10.1007/978-3-319-10605-2_3.

[40] S. Canavan, P. Liu, X. Zhang, and L. Yin, "Landmark localization on 3D/4D range data using a shape index-based statistical shape model with global and local constraints," *Comput. Vis. Image Underst.*, vol. 139, pp. 136–148, Oct. 2015, doi: 10.1016/j.cviu.2015.06.006.

[41] G. Fanelli, M. Dantone, and L. Van Gool, "Real time 3D face alignment with Random Forests-based Active Appearance Models," in *2013 10th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, Apr. 2013, pp. 1–8. doi: 10.1109/FG.2013.6553713.

[42] J. Sun, D. Huang, Y. Wang, and L. Chen, "Expression Robust 3D Facial Landmarking via Progressive Coarse-to-Fine Tuning," *ACM Trans. Multimed. Comput. Commun. Appl.*, vol. 15, no. 1, p. 21:1-21:23, Feb. 2019, doi: 10.1145/3282833.

[43] C. L. Heike, K. Upson, E. Stuhaug, and S. M. Weinberg, "3D digital stereophotogrammetry: a practical guide to facial image acquisition," *Head Face Med.*, vol. 6, no. 1, p. 18, Jul. 2010, doi: 10.1186/1746-160X-6-18.

[44] A. De Stefani *et al.*, "Validation of Vectra 3D Imaging Systems: A Review," *Int. J. Environ. Res. Public. Health*, vol. 19, no. 14, p. 8820, Jul. 2022, doi: 10.3390/ijerph19148820.

[45] Z. Majid, A. K. Chong, A. Ahmad, H. Setan, and A. R. Samsudin, "Photogrammetry and 3D laser scanning as spatial data capture techniques for a national craniofacial database," *Photogramm. Rec.*, vol. 20, no. 109, pp. 48–68, 2005, doi: 10.1111/j.1477-9730.2005.00304.x.

[46] R. J. Hennessy, S. McLearie, A. Kinsella, and J. L. Waddington, "Facial Shape and Asymmetry by Three-Dimensional Laser Surface Scanning Covary With Cognition in a Sexually Dimorphic Manner," *J. Neuropsychiatry Clin. Neurosci.*, vol. 18, no. 1, pp. 73–80, Feb. 2006, doi: 10.1176/jnp.18.1.73.

[47] P. Li, R. Wang, Y. Wang, and W. Tao, "Evaluation of the ICP Algorithm in 3D Point Cloud Registration," *IEEE Access*, vol. 8, pp. 68030–68048, 2020, doi: 10.1109/ACCESS.2020.2986470.

[48] J. Zhang, Y. Yao, and B. Deng, "Fast and Robust Iterative Closest Point," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 7, pp. 3450–3466, Jul. 2022, doi: 10.1109/TPAMI.2021.3054619.

[49] G. C. Sharp, S. W. Lee, and D. K. Wehe, "ICP registration using invariant features," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 1, pp. 90–102, Jan. 2002, doi: 10.1109/34.982886.

[50] J. Baert, “Morton encoding/decoding through bit interleaving: Implementations,” Jeroen Baert’s Blog. Accessed: Jan. 31, 2024. [Online]. Available: https://www.forceflow.be/2013/10/07/morton-encodingdecoding-through-bit-interleaving-implementations/

[51] G. M. Morton, “A computer oriented geodetic data base and a new technique in file sequencing,” 1966.

[52] K. O’Shea and R. Nash, “An Introduction to Convolutional Neural Networks.” arXiv, Dec. 02, 2015. Accessed: Jan. 18, 2024. [Online]. Available: http://arxiv.org/abs/1511.08458

[53] J. Wu, “Introduction to Convolutional Neural Networks,” 2017.

[54] Ian Goodfellow and Yoshua Bengio and Aaron Courville, *Deep Learning*. MIT Press.

[55] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, JMLR Workshop and Conference Proceedings, Mar. 2010, pp. 249–256. Accessed: Feb. 28, 2024. [Online]. Available: https://proceedings.mlr.press/v9/glorot10a.html

[56] K. He, X. Zhang, S. Ren, and J. Sun, “Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification.” arXiv, Feb. 06, 2015. doi: 10.48550/arXiv.1502.01852.

[57] A. Dhillon and G. K. Verma, “Convolutional neural network: a review of models, methodologies and applications to object detection,” *Prog. Artif. Intell.*, vol. 9, no. 2, pp. 85–112, Jun. 2020, doi: 10.1007/s13748-019-00203-0.

[58] P. Simard, D. Steinkraus, and J. Platt, “Best Practices for Convolutional Neural Networks,” Sep. 2003.

[59] W. Zhiqiang and L. Jun, “A review of object detection based on convolutional neural network,” in *2017 36th Chinese Control Conference (CCC)*, Jul. 2017, pp. 11104–11109. doi: 10.23919/ChiCC.2017.8029130.

[60] S. S. Basha, S. R. Dubey, V. Pulabaigari, and S. Mukherjee, “Impact of fully connected layers on performance of convolutional neural networks for image classification,” *Neurocomputing*, vol. 378, pp. 112–119, 2020.

[61] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation.” arXiv, Apr. 10, 2017. doi: 10.48550/arXiv.1612.00593.

[62] F. Zhang, J. Fang, B. Wah, and P. Torr, “Deep FusionNet for Point Cloud Semantic Segmentation,” in *Computer Vision – ECCV 2020*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 644–663. doi: 10.1007/978-3-030-58586-0_38.

[63] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, “Spatial Transformer Networks.” arXiv, Feb. 04, 2016. doi: 10.48550/arXiv.1506.02025.

[64] D. Gibelli, V. Pucciarelli, A. Cappella, C. Dolci, and C. Sforza, “Are Portable Stereophotogrammetric Devices Reliable in Facial Imaging? A Validation Study of VECTRA H1 Device,” *J. Oral Maxillofac. Surg. Off. J. Am. Assoc. Oral Maxillofac. Surg.*, vol. 76, no. 8, pp. 1772–1784, Aug. 2018, doi: 10.1016/j.joms.2018.01.021.

[65] C. Dolci *et al.*, “Robustness of Distinctive Facial Features in Prader-Willi Syndrome: A Stereophotogrammetric Analysis and Association with Clinical and Biochemical Markers in Adult Individuals,” *Biology*, vol. 11, no. 8, p. 1148, Jul. 2022, doi: 10.3390/biology11081148.

[66] V. F. Ferrario, C. Sforza, C. E. Poggio, M. Cova, and G. Tartaglia, “Preliminary evaluation of an electromagnetic three-dimensional digitizer in facial anthropometry,” *Cleft Palate-Craniofacial J. Off. Publ. Am. Cleft Palate-Craniofacial Assoc.*, vol. 35, no. 1, pp. 9–15, Jan. 1998, doi: 10.1597/1545-1569_1998_035_0009_peoaet_2.3.co_2.

[67] “trimesh 4.1.3 documentation.” Accessed: Feb. 05, 2024. [Online]. Available: https://trimesh.org/index.html

[68] E. W. Weisstein, “Triangle Point Picking.” Accessed: Feb. 05, 2024. [Online]. Available: https://mathworld.wolfram.com/

[69] Q.-Y. Zhou, J. Park, and V. Koltun, “Open3D: A Modern Library for 3D Data Processing.” arXiv, Jan. 29, 2018. doi: 10.48550/arXiv.1801.09847.

[70] C. Wu, S. Zhang, F. Long, Z. Yin, and T. Leng, “Towards Better Orthogonality Regularization with Disentangled Norm in Training Deep CNNs.” arXiv, Jun. 16, 2023. doi: 10.48550/arXiv.2306.09939.

[71] L. Yin, X. Wei, Y. Sun, J. Wang, and M. J. Rosato, *A 3D Facial Expression Database For Facial Behavior Research*, vol. 2006. 2006, p. 216. doi: 10.1109/FGR.2006.6.

[72] X. Huang, G. Mei, J. Zhang, and R. Abbas, “A comprehensive survey on point cloud registration.” arXiv, Mar. 05, 2021. doi: 10.48550/arXiv.2103.02690.

[73] X. Huang *et al.*, “Robust real-world point cloud registration by inlier detection,” *Comput. Vis. Image Underst.*, vol. 224, p. 103556, Nov. 2022, doi: 10.1016/j.cviu.2022.103556.

[74] X. Bai *et al.*, “PointDSC: Robust Point Cloud Registration Using Deep Spatial Consistency,” presented at the Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 15859–15869. Accessed: Mar. 11, 2024. [Online]. Available: https://openaccess.thecvf.com/content/CVPR2021/html/Bai_PointDSC_Robust_Point_Cloud_Registration_Using_Deep_Spatial_Consistency_CVPR_2021_paper.html

[75] D. Chai and K. N. Ngan, “Face segmentation using skin-color map in videophone applications,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 4, pp. 551–564, Jun. 1999, doi: 10.1109/76.767122.

[76] S. Benini, K. Khan, R. Leonardi, M. Mauro, and P. Migliorati, “Face analysis through semantic face segmentation,” *Signal Process. Image Commun.*, vol. 74, pp. 21–31, May 2019, doi: 10.1016/j.image.2019.01.005.

[77] S. A. Sirohey, “Human Face Segmentation and Identification,” Oct. 1998, Accessed: Mar. 11, 2024. [Online]. Available: http://hdl.handle.net/1903/400

[78] M. P. Segundo, C. Queirolo, O. R. P. Bellon, and L. Silva, “Automatic 3D facial segmentation and landmark detection,” in *14th International Conference on Image Analysis and Processing (ICIAP 2007)*, Sep. 2007, pp. 431–436. doi: 10.1109/ICIAP.2007.4362816.

[79] C. Yue, Y. Wang, X. Tang, and Q. Chen, “DRGCNN: Dynamic region graph convolutional neural network for point clouds,” *Expert Syst. Appl.*, vol. 205, p. 117663, Nov. 2022, doi: 10.1016/j.eswa.2022.117663.

[80] Z. Zhai, X. Zhang, and L. Yao, “Multi-Scale Dynamic Graph Convolution Network for Point Clouds Classification,” *IEEE Access*, vol. 8, pp. 65591–65598, 2020, doi: 10.1109/ACCESS.2020.2985279.

[81] R. Li *et al.*, “PointVGG: Graph convolutional network with progressive aggregating features on point clouds,” *Neurocomputing*, vol. 429, pp. 187–198, Mar. 2021, doi: 10.1016/j.neucom.2020.10.086.

[82] Y. Cui, X. Liu, H. Liu, J. Zhang, A. Zare, and B. Fan, “Geometric attentional dynamic graph convolutional neural networks for point cloud analysis,” *Neurocomputing*, vol. 432, pp. 300–310, Apr. 2021, doi: 10.1016/j.neucom.2020.12.067.

[83] A. Savran *et al.*, “Bosphorus Database for 3D Face Analysis,” in *Biometrics and Identity Management*, B. Schouten, N. C. Juul, A. Drygajlo, and M. Tistarelli, Eds., in Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2008, pp. 47–56. doi: 10.1007/978-3-540-89991-4_6.