Three-dimensional Facial Landmark Detection in 3D Photos

Master's Thesis in Computing Science (Specialisation: Data Science)

Luca Carotenuto
s1047465

February 27, 2022

Daily Affiliated Supervisor:
dr. Guido de Jong*

Internal Supervisor:
prof. dr. Tom Heskes**

Second reader:
prof. dr. Thomas Maal*

*
3D Lab, Radboudumc, Nijmegen, The Netherlands
Deparment of Oral and Maxillofacial Surgery, Radboudumc, Nijmegen, the Netherlands

**
Institute for Computing and Information Sciences, Radboud University, Nijmegen, The Netherlands

Radboudumc Radboud University

Abstract

Three-dimensional (3D) landmarks are used in various fields within medicine. Oral and maxillofacial surgery involves reconstructive operations on the head, face and jaw as well as facial cosmetic surgery. Landmarks are being used during the planning, follow-up and diagnostics of surgical interventions. However, placing 3D landmarks manually can be tedious and inconsistent. AI-assisted landmark detection can help to automate this process by making use of recent developments in the field of 3D deep learning. This work leverages DiffusionNet for surface-learning on point clouds. DiffusionNet is a representation-independent and sampling robust network structure based on heat diffusion. This work presents a point-wise regression method that predicts regions around landmarks with increasing activation closer to the landmark point. The initial network predicts rough landmark positions based on the raw coordinates or the heat kernel signature. A refinement network is subsequently applied to more accurately locate the landmark based on its neighbourhood sampled in high resolution. The refinement network appears effective as it improves the initial network's detection accuracy of 2.78mm to 2.22mm. Raw coordinate input shows better detection accuracy on faces and craniums that are consistently oriented in space. It was found that the Heat Kernel Signature is a more suitable shape descriptor for faces "in the wild", where such assumptions cannot be made.

1 Introduction

Three-dimensional (3D) landmarks find application in various fields within medicine, such as cephalometry, the study and measurement of the head. Jaw surgery, also known as orthognathic surgery, deals with correcting irregularities of the jaw bones. Orthognathic surgery and orthodontics often aim at creating facial balance or harmony [1]. Landmarks help the surgeon during the diagnosis, planning and documentation of surgical interventions. Moreover, they are used to retrospectively judge whether the interventions have been successful.

Conventionally, landmarks are placed by the surgeon manually. This process is tedious and introduces human error. It can suffer from a high variability caused by how the same surgeon places landmarks due to human error (intraobserver variability) and how different surgeons place landmarks due to different landmarking habits (interobserver variability). AI-assisted landmarking can automate the landmarking procedure by employing recent advances in the field of deep learning.

In machine learning, a distinction is made between object localization and object detection. The former only locates the presence of an object, whereas the latter also assigns a class label to the object. In this work, the problem is tackles as a landmark detection problem, to distinguish between different landmark types.

Different modalities can be used for landmarking, such as 3D photos, textured 3D photos, bony-tissue CT scans and soft-tissue CT scans. 3D photographs only capture soft tissue. Nonetheless, [1] showed high reproducibility for most soft tissue landmarks, suggesting that no hard tissue data, i.e. bony structure, is needed to perform accurate soft tissue analysis. Acquiring hard tissue data requires radiation-based capturing devices such as X-ray or CT, whereas soft tissue images can be recorded by 3D stereophotogrammetry (see Figure 1). Stereophotogrammetry is a three-dimensional registration method for quantifying facial morphology and detecting changes in facial morphology during growth and development [2]. The images come from the Headspace dataset [3], a public dataset that comprises 3D images of the human head for 1519 subjects. The majority of the images come with landmark annotations. However, the annotations from the Headspace dataset have been identified automatically. Specifically, they were determined by the Zhu-Ramanan mixture-of-trees algorithm [4] applied to the texture of each mesh. Dai et al. project the 2D points to 3D using the texture coordinates, which adds inaccuracies to the resulting landmarks. Thus, we manually annotate 3D landmarks for 307 3D photos to ensure accurate testing scores and improve training by using more accurate annotations.

Deep learning is becoming an increasingly powerful tool for data processing in computer vision tasks. Especially 2D computer vision tasks can be solved with high accuracy. Convolutional neural networks (CNNs) have delivered excellent results in computer vision tasks such as classification, object detection and segmentation. However, 3D deep learning faces several challenges. 3D data can be represented in different formats, including depth images, multi-view images, volumetric grids, point clouds or meshes. Which data representation should be used depends heavily on the application and the data acquisition device. Point clouds and meshes do not suffer from discretization or projection loss and are therefore the preferred method for surface-based learning [5]. Point clouds and meshes are intrinsically non-euclidean data representations. Due to the irregular distribution of the points in space, conventional techniques such as convolution are not directly transferable to 3D.

Moreover, there is a lack of big data sets. There exist several big data sets for 2D facial landmarking, such as the Annotated Facial Landmarks in the Wild (AFLW) [6] collection that comprises 25,993 faces. However, even with larger 3D data sets, training would remain difficult due to high computation costs and memory footprints.Despite these challenges, in recent years, the field of 3D deep learning comes up with increasingly powerful techniques to tackle surface learning problems. This work leverages DiffusionNet [7], a discretization agnostic network by Nicholas Sharp et al., to extract features for the prediction of heatmaps around landmarks.

The paper starts with related works of facial landmarking and important methods for 3D deep learning in Section 2. Then, the data, pre-processing and networks are described in section 3. Section 4 shortly presents the experimental setup and section 5 the quantitative results on the Headspace dataset and qualitative results on data from Radboudumc. In Section 6, limitations and future work are discussed. Section 7 finally concludes the paper.

2 Related Work

2.1 Facial landmarking

There is extensive research about 2D facial landmarking. Many non-medical tasks such as person identification, expression transfer or emotion recognition require automatic landmarking as a necessary step [9]. Existing methods for facial landmark detection can be classified into two categories: generative and discriminative. Generative methods model the facial shape as a probabilistic distribution. This category includes part-based generative models such as ASM and holistic generative models such as AAM, that capture variations in the shape or texture by Principal Component Analysis (PCA), or Gauss- Newton Deformable Part Models (GN-DPM) [10]. Discriminative models take a different approach and directly look for relevant features which can be used to localize the landmarks given the input. Discriminative methods include Cascaded Regression models but also neural networks. With the

Table 1: **Landmarks that are considered in this work.** Definitions from [8] and [1]. Left/right is given from the perspective of the subject.

Landmark	Abbreviation	Definition
Pogonion	pg	The most anterior midpoint of the chin
Nasion	n	The Point in the midline of both the nasal root and nasofrontal suture
Pronasale	prn	The most anterior midpoint of the nasal tip
Subnasale	sn	The midpoint on the nasolabial soft tissue contour between the columella crest and the upper lip
Alar curvature (right)	ac-r	The point located at the facial insertion of the alar base (right)
Alar curvature (left)	ac-l	The point located at the facial insertion of the alar base (left)
Exocanthion (right)	ex-r	The soft tissue point located at the outer commissure of the right eye fissure
Endocanthion (right)	en-r	The soft tissue point located at the inner commissure of the right eye fissure
Endocanthion (left)	en-l	The soft tissue point located at the inner commissure of the left eye fissure
Exocanthion (left)	ex-l	The soft tissue point located at the outer commissure of the left eye fissure
Cheilion (right)	ch-r	The point located at the right labial commissure
Cheilion (left)	ch-l	The point located at the left labial commissure

emergence of Convolutional Neural Networks (CNNs), many traditional methods have been outperformed by neural networks. Most research on facial landmarking focuses on 2D. This research is focused on 3D facial landmarking with deep neural networks. As there is little research about deep learning for the case of 3D facial landmark detection, the chapter is focused on more general work on 3D deep learning.

2.2 3D Deep Learning

In the past years, point cloud understanding is receiving increasing attention from the research community as practical applications such as autonomous driving and robotics emerge. These applications require more information than flat images can provide to obtain a better sense of the environment. The 3D data is captured by cameras with depth sensor, e.g., lidar or RGB-D cameras.

Most authors that develop novel network architectures report their results for the more common 3D tasks classification, segmentation, object detection or shape correspondence. Since results for keypoint detection are less common, the performance results of the networks can only be regarded as a rough reference for their potential feature extraction.

Bello et al. [11] identify three main challenges of deep learning with point clouds: (1) Irregularity, meaning points in the point cloud are unevenly or irregularly sampled across different regions, leaving some regions that are more sparsely, and some that are more densely occupied. (2) Unstructuredness, meaning the distance between points is not fixed, and the points can not be placed on a grid, as can be done for 2-dimensional images. (3) Unorderedness, which expresses, that a point set can be shuffled arbitrarily yielding the same point cloud, making the point set invariant to permutation.

For years, especially the unorderedness and unstructuredness of surface data hindered the performance of machine learning approaches. Early methods solve these problems by converting the point cloud into a structured grid. These methods generally fall into Voxel-based and Multi-view-based approaches.

Voxel-based approaches convert the point cloud into a volumetric occupancy grid, with the smallest unit being a fixed-size voxel. Then, 3D convolutions combined with pooling can be applied to extract features. Pioneering work that falls into this category is VoxNet [12], which introduced a volumetric occupancy network for 3D object recognition. Wu et al. showed with ShapeNets [13] that by representing 3D shapes as probability distributions of binary variables on 3D voxels and a convolutional deep belief network, encouraging performance can be achieved. However, 3D CNNs often scale poorly, as the computation and memory footprint grow cubically [5]. More recent work such as the Minkovski Engine [14] by Nvidia make use of the sparsity of voxels in space. Special operations are used for spatially sparse tensors. Conventional sparse CNNs compress networks by pruning weights to create sparse parameters but still operate on dense tensors. In contrast, the Minkovski Engine processes sparse tensors at every stage of the network including the convolutional layers.

Multi-view based methods benefit from the extensive research on 2D CNNs. Similarly to how the human eye perceives depth, a 3D image can be constructed by combining views from multiple angles. In the original work, MVCNN [15], the image is rendered from different views, a CNN is applied to each of the 2D images, and a view pooling operation combines the features. A significant drawback of this method is that the max-

pooling layer only preserves the maximum value from one view. Although later methods solve this by more sophisticated view pooling, multi-view based methods still suffer from illumination differences, object occlusion and information loss during the reconstruction of the objects from different views. Furthermore, they fail to capture the underlying structure and geometry of the data.

Deep learning on point clouds has received increasing potential since Qi et al. presented PointNet [16] in 2017. Unlike the Euclidean case, there is no universal concept for convolutions in 3D due to the unstructuredness of the data. Different types of approaches have been developed to address this problem. PointNet is one of the first methods to apply deep learning to unstructured and unordered inputs. PointNet learns per-point features with a Multi-Layer-Perceptron (MLP) with shared parameters. Global features are determined by symmetrical pooling functions. Symmetric functions output the same numbers independent of the input order, making them a perfect fit for dealing with unordered point sets. Since PointNet does not consider local dependency among points, the local structure cannot be captured. Therefore, PointNet++ [17] was published that manages to improve classification and segmentation performance by modelling local regions through sampling and grouping. PointCNN [18] goes a step further and also takes into account the correlation between points in the local regions.

An approach that operates directly on triangular mesh data is MeshCNN [19], which applies convolutions on edges and the four edges of their incident triangles, and pooling is applied via an edge collapse operation. Graph-based methods generate a graph with vertices and directed edges. Feature learning can be performed either in the spatial or spectral domain. Approaches that compute the Laplacian for representing the graph operate in the spectral domain. An advantage of graph-based methods is their generality. They apply to many fields other than triangular meshes, such as for social networks, sensor networks or genetic data.

This work uses DiffusionNet [7] to learn features on surfaces. DiffusionNet borrows concepts from point-based and graph-based methods by combining point-wise MLPs with a learned diffusion operation. Although the Laplacian is used to define the diffusion operator, the network itself operates in the spatial domain, as the spectral operations are merely used in the spectral expansion step to accelerate the evaluation of diffusion.

3 Methods

3.1 Data set

The models are trained predominantly on the Headspace dataset [3], a set of 3D images of the human head that is available for university-based non-commercial research. The collection consists of 1519 subjects, each wearing tight-fitting latex caps to avoid holes in the mesh on the scalp of the patient. The photos are captured by a 5-camera setup around the head of the person (Fig. 1).

Figure 1: **Stereophotogrammetry.** The patient is being photographed from 5 different angles around the head. Subsequently, the mesh is created by combining the different views into a single 3D mesh. Photo from Headspace dataset[3].

The images have a high quality, consistent illumination and are pose normalized. 1200 samples include annotations, but as they were automatically generated, the quality of the labels differs for each landmark type and each sample. Nonetheless, due to the sheer number of annotated samples, the Headspace data is used as training data for the first network that extracts rough landmark locations. Manual inspection shows that the Zhu-Ramanan mixture of trees algorithm in subsequent 2D to 3D projection similar mistakes for most samples, e.g. the landmarks are placed a few millimetres too low. In total, the Headspace data set comes with 68 landmarks per image. However, many of them are non-surgical and ill-defined and can be discarded for the purpose of finding landmarks to assist oral and maxillofacial surgery. Out of the 68 landmarks, we keep 12 medically relevant, anatomical landmarks (see table 1). to properly evaluate the results and to train the refinement network, we manually annotate the 12 landmark positions for around 307 Headspace samples. The landmarking is performed in 3DMedX[1][20]. Although not placed by a specialist, these labels are considerably more accurate, as they are human-annotated instead of to machine-annotated (see appendix C for details).

Furthermore, 3D photos from Radboudumc are used to validate the method further. The in-house meshes have lower quality, and the illumination varies greatly. The 3D photos are not pose normalized oriented differently in space than the Headspace photos. As the

1. 3DMedX is software from Radboudumc that allows 3D reconstruction using DICOM files from (CB)CT-scans or MRI scans and offers tools for the evaluation of orthognathic surgery. It also supports the creation of custom workflows, e.g. for registering landmarks.

Figure 2: **Data comparison.** Left: Two subjects from the Headspace data set. Right: Two subjects from the Radboudumc data. The meshes on the right contain more artefacts because a 2-camera setup captured the photos instead of a 5-camera setup. Additionally, the subjects do not wear latex caps which makes the reconstruction of the hair more difficult. The meshes on the left and the right are differently oriented in space.

patients do not wear any latex caps, holes and artefacts in the region of the hair are common. Also, most 3D photos are captured by a 2-camera-setup, namely one from the front-left and one from the front-right. Consequently, meshes reconstructed from such photos have big holes in the back of the cranium as they only capture the frontal view.

See figure 2 for a visualization of the photos from the Headspace data set and the Radboudumc data set.

3.2 Data Preparation

The 3D meshes are available as wavefront object'files (OBJ.). This file format contains information about vertices, edges, faces, normal vectors and texture. Vertices are points in the Cartesian coordinate system defined by x, y and z. Meshes also contain surface data in edges and faces that define the interconnectivity between vertices. Normals define vectors that are perpendicular the to face. Textures add color information by storing uv-coordinates per vertex. The uv-coordinates map the vertex to a bitmap file (.BMP) to refer to the corresponding color [21]. To simplify the problem, the networks process point clouds instead of meshes. Using PyMeshLab, a Python interface for the 3D mesh processing software Meshlab [22], filters are applied to prepare the point cloud data. First, a point sampling is applied, optionally with corresponding normals and colors. This process preserves the vertices of the mesh while discarding edge and face information. In literature, irregular sampling was identified as a factor that can potentially hamper certain machine learning methods. Thus, we use a sampling technique ('Texel Sampling') that was observed to produce evenly distributed points on the surface. Depending on the network the data is prepared for, the point cloud can be simplified to reduce the resolution. The refinement network takes as input the full resolution point cloud with an average vertex-to-vertex distance of 1.0mm. In contrast, the initial network is applied on low-resolution data with an average vertex-to-vertex distance of 2.2mm. Around 30,000 vertices are sampled for the low-resolution point cloud.

The 68 landmarks in the Headspace data are given by a reference to the vertex index in the mesh. The manually annotated landmarks in 3DMedX are saved as coordinates in Comma-separated values files. Because the manually identified landmarks are stored as an arbitrary coordinate in space, it usually does not refer to a point. As we employ a point-wise network, the targets also need to be saved point-wise. To this end, the closest point to the landmark coordinate is identified. For the Headspace landmarks, landmarks are already given in a point-wise fashion. However, as the initial network requires a downsampled point cloud, the landmark might not point to a vertex in the downsampled point cloud anymore and still needs to be recalculated. Currently, the calculation of the closest point to the landmark is done in a naive way by iterating over each vertex in the mesh. This step can be sped up significantly by applying a more sophisticated algorithm. Constructing the targets as simple point landmarks can have certain disadvantages. For example, the neural network is more difficult on sparse positive cases. Therefore, we construct point clusters around the landmark point to create regions that the network can learn more easily. The point closest to the landmark has the highest activation (1.0), points in the neighborhood are assigned decreasing activations with decreasing proximity and points outside the landmark regions are considered background points with activation of zero. The exact activation scheme differs depending on which network the targets are prepared (see table 2 for details). An alternative, perhaps more balanced solution would be to assign activations based on a continuous distribution such as the Gaussian distribution. The approach of generating point clusters around landmarks increases the proportion of points with an activation higher than zero and improves the class imbalance problem.

Activation	Distance range	
	Initial network	Refinement network
1.0	closest point	closest point
0.75	up to 3.0mm	up to 1.5mm
0.5	3.0 to 4.5mm	1.5 to 3.0mm
0.25	4.5 to 6.0mm	3.0 to 4.5mm
0.0	higher than 6.0mm	higher than 4.5mm

Table 2: **Target activation scheme.** The points are assigned an activation depending on the proximity to the respective landmark.

To build a landmark detector that can discriminate between landmark types and to allow for overlapping activation clusters (meaning one point can be part of the neighborhood of multiple landmarks), each landmark cluster is stored a channel.

However, storing the activation for each point is very memory-consuming. To compress the label files, only the vertex indices and activation for points with an activation higher than zero are explicitly stored. All other points are assumed to have an activation equal to zero. This reduces the necessary information to store from *total vertices*×*landmarks* for the sparse matrix representation to *vertices in landmark regions* × *2* × *landmarks* where the factor 2 arises as the compressed representation not only needs to save the activation but also additional vertex index information. The sparse labels are saved as serialized Python object structures, so-called pickle objects.

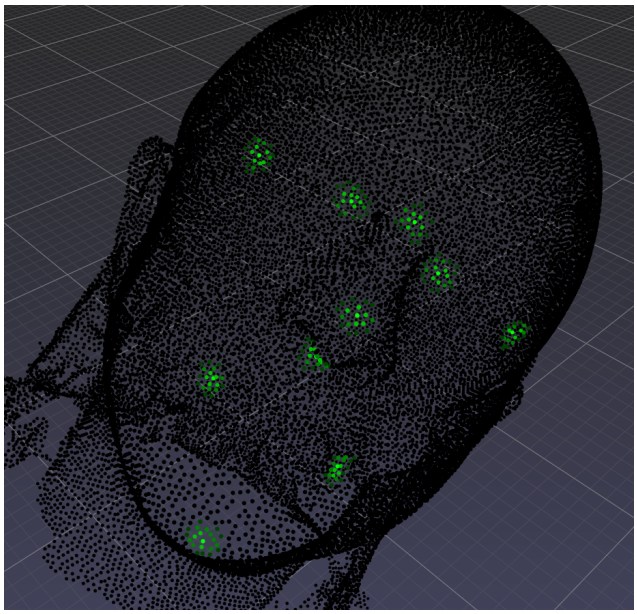


Figure 3: **Prepared sample that serves as the ground truth input for the initial network.** The intensity of the green points represents the activation of each point. Activations are either 0 for background and 0.25, 0.5, 0.75 or 1 for points in landmark regions.

3.3 DiffusionNet

In this work, DiffusionNet [7] is used to extract surface features from faces. With a diffusion operation, surface information is propagated through space. Diffusion is defined by the heat equation

$$\frac{d}{dt}u_t = \Delta u_t, \tag{1}$$

where Δ is the Laplacian and u_t the diffused distribution. The action of diffusion is defined as

$$H_t(u_0) = e^{-t\Delta}u_0, \tag{2}$$

where the heat operator H_t creates a diffused distribution based on the initial distribution u_0. With increasing time, diffusion becomes a global smoothing process, thus approaching the average for $t \to \infty$. An advantage of this technique is that diffusion is fairly robust to how the surface is sampled. In practice, diffusion needs to be discretized by replacing the distributional Laplacian with the weak Laplace matrix L, which, together with the mass matrix M (diagonal matrix of areas associated with each vertex) yields the rate of diffusion $M^{-1}Lu$.

A diffusion layer $h_t : \mathbb{R}^V \to \mathbb{R}^V$ creates the diffused matrix for V vertices and the feature channel u. The time $t \in \mathbb{R}_{\geq 0}$ is a learned parameter per feature channel. The time parameter represents the spatial support and is analogous to the receptive field in CNNs, dependent on the filter size and pooling. Since diffusion time is a learned parameter per feature channel, the spatial support is more flexible and does not need to be manually chosen. Interestingly, Sharp et al. give theoretical proof that shows that replacing convolution by diffusion maintains the expressive power of the network.

The evaluation of the diffusion layer $h_t(u)$ needs to be efficient and differentiable to allow network training. Sharp et al. propose to use a direct implicit timestep or spectral expansion. This work uses spectral expansion as the diffusion can be more efficiently computed at evaluation time, with the trade-off that the eigenbases need to be precomputed. Diffusion is truncated to a lower frequency basis k, which leads to some approximation error, but no noticeable effect on network performance.

Since diffusion only allows for radially symmetric filters, in addition, spatial gradients are used to enable directional filters. In the case of point cloud input without normals, the normal vectors per point are approximated by the m-nearest neighbors (we use $m = 30$). A least squares approximation then calculates gradient values in the tangent plane which yields the sparse gradient matrix $G \in \mathbb{C}^V$. Applying the feature vector u to the sparse matrix yields the gradient tangent vector per point.

3.4 Implementation

We define a point set $X = \{X_i \in \mathbb{R}^F, \quad i = 1, 2, ..., V\}$ as the input of the model, where V defines the number

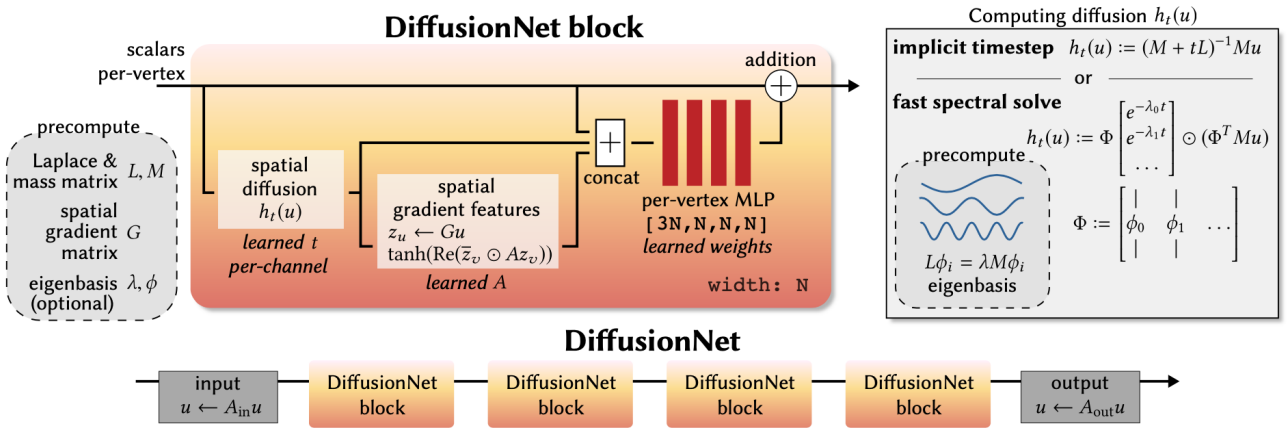


Figure 4: **DiffusionNet architecture.** After the precomputation of necessary operators, the input is applied to several blocks. In each block, spatial gradients and diffused features are fed into a point-wise MLP with shared weights, and residual connections facilitate training. In this work, diffusion is computed by spectral acceleration.

of vertices in the point cloud, D the dimension, and x_i is the 3D coordinate of each point in the Cartesian reference system. Note that even in a 3-dimensional reference system, D is not restricted to 3 as we can use other point-based features such as the color or the normal vector.

Most machine learning algorithms require a fixed input size. DiffusionNet can deal with a flexible input size, making sampling or simplification for standardizing the number of vertices unnecessary.

The problem is approached as a point-wise regression problem as opposed to directly regressing coordinates (see appendix B.2 for details).

The network and data loading is implemented in PyTorch [23], a popular deep learning library that supports GPU acceleration.

DiffusionNet requires precomputing the Laplacian, mass matrix and gradients for spectral acceleration and saving them in a cache to quickly access them during data loading to avoid repeating the computationally expensive operations. Together with these operators we additionally cache the labels. Since the labels were previously compressed to reduce disk space, they are decompressed as the sparse representation is needed for training.

As DiffusionNet is originally built for classification, segmentation and functional correspondence tasks. We adapt the network architecture to output per-point scores for each landmark channel by changing the size of the last layer to *vertices* × *landmarks*.

Given the per-point scores for each landmark channel, the point landmarks can be extracted very easily. A simple argmax operation can find the point with the maximum activation which is regarded as the best estimate for the landmark prediction.

3.5 Training

The networks are trained with the weighted mean squared error (WMSE) metric. Mean squared error (MSE) guarantees the error to be strictly positive the square punishes very large differences:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{3}$$

Even with modelling landmark regions, activations > 0 are rare due to an imbalance between background and landmark regions. We introduce weights to counteract the class imbalance problem. Points with higher activations are assigned higher weights and background points are assigned a weight of one. The exact weighting scheme differs for the initial network and the refinement network, because they exhibit different imbalances between background and landmark regions. The weighting schemes ensure that the network pays sufficient attention to finding points in landmark regions while not overemphasizing on them, which can lead to exceedingly large and blurry clusters. Leaving out the weight leads to the network not learning anything or very small activations. In contrast, too high weights lead to exceedingly high activations making it hard to extract the correct point landmark with the maximum activation as the activations are too close to each other.

3.6 Initial Network

The initial network takes as input the full point cloud which typically involves the cranium, neck and possibly some artefacts and other body parts of the upper body depending on the photo and mesh reconstruction. The original Headspace meshes can have 100,000 to 200,000 vertices. Processing the photos in full resolution would lead to excessively high memory footprints and very long times for the precomputation of the operators. Thus, around 30,000 vertices are sampled and used as input for the initial network (see figure 3 for a visualialization of the input and figures 5 for a visualization of the output). Due to the big class imbalance, the weighting scheme is set quite aggressively (see table 3)

Activation	Weight	
	Initial network	Refinement network
0.0	1	1
0.25	25	5
0.5	50	10
0.75	75	15
1.0	100	20

Table 3: **Weighting scheme.** The initial network uses more aggressive weighting for points with higher activation to combat class imbalance. For the refined network, smaller weights are sufficient because of less prominent class imbalance.

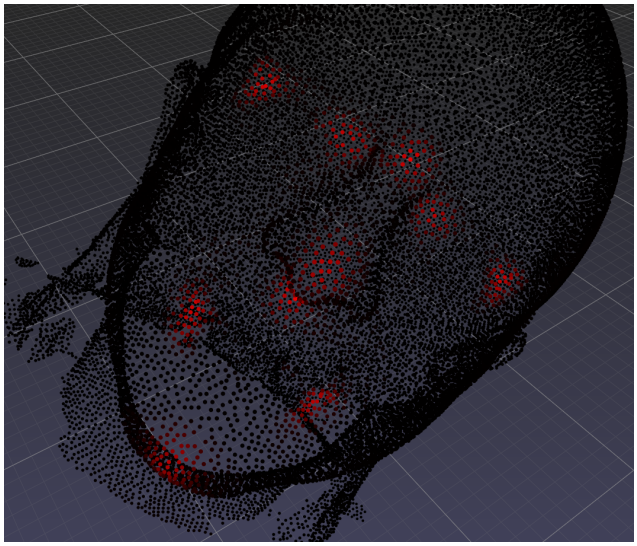


Figure 5: **Prediction example.** Sample from Headspace data set. The intensity of the red points represents the predicted score. Note that although it is not visible in this depiction, each cluster is part of a different channel to ensure a clear distinction between landmarks.

3.7 Refinement Network

The refinement network is applied to regions around landmarks. The regions are sampled in full resolution to enable a higher potential accuracy. For training, a sphere around the ground truth landmark with a radius r cut out. To prevent the model from overfitting by consistently selecting the center point of the sphere as the mid point of the predicted cluster, we specify a jittering j of the cluster center. It is implemented by a random translation of up to j millimeters in each direction. For each landmark region in a photo, we take three jittering samples to augment the training set. This effectively triples the data set and makes better use of the training samples available.

To emphasize the network's focus on the local neighborhood we adapt a more local activation scheme for the targets.

The regional point clouds only comprise around 1000 to 2000 points for the Headspace data, depending on the landmark type. For the Radboudumc data, this number is even lower since the meshes have a lower resolution. This, combined with the chosen activation scheme, leads to a less prominent class imbalance, reflected in the chosen weights.

Similarly, during prediction, a sphere of the same size is cut around the predicted landmark and sampled in high-resolution (see figure 6).

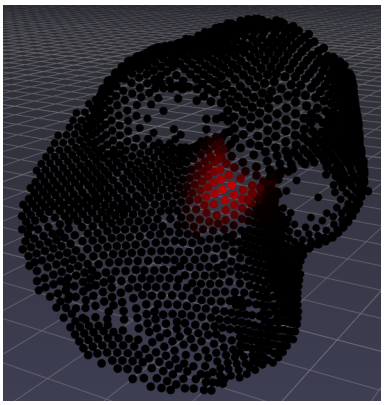


Figure 6: **Refined example.** The image shows the subnasale landmark. Previously, the rough landmark location is inferred by the initial network. Here, the refined network infers the location from the high-resolution point cloud sampled in the local region. The intensity of the red points represents the predicted score of the refinement network.

3.8 Shape Variants

3D shapes can come in different variants that the network should be invariant to, such as different orientations or different discretization. Varying camera setups or differing preprocessing can lead to different head orientations in space. The network should give the same result regardless of head rotations. The perhaps most straightforward approach is to perform data augmentation. While it can make the network more robust to the presented augmented variants, data augmentation does not scale well as it is not feasible to sample all variations. Additionally, including slightly varied samples in training quickly increases training times. The preferred approach to dealing with shape variants is to design an inherently invariant network to rotations. There is still ongoing research on how to design such invariant networks most efficiently.

Another way is to use input features that are intrinsically invariant to isometric transformations, thus also to rigid deformations such as rotation. To this end, we include the Heat Kernel Signature (HKS) in our experiments and compare HKS features to raw XYZ coordinate features.

Simple preprocessing steps can already create robustness to scaling and translation. Thus, we center and unit scale the objects. Centering mitigates translations in the x-, y- or z-axis by centering around the average of the point positions. Unit scaling can avoid significant variances in head sizes. However, it should

be noted that both centering and unit scaling are applied to the entire point cloud. Because the photos vary a lot in artefacts and different body parts captured, the scaling and centering of the face itself are noisy.

While centering the object is likely unnecessary for the network with HKS features as input, as the features are already invariant to rigid deformations, it can improve robustness for networks with raw coordinate input which is sensitive to translation.

3.9 Augmentation

We implement mirroring or horizontal flipping as a data augmentation technique. This effectively doubles the training samples by making use of the symmetry of faces.

To create a mirrored data set, we use a general method that finds the rotation matrix R that aligns unit vector a onto unit vector b. With the cross product $v = a \times b$, the sine of the angle $s = \|v\|$ and the cosine of the angle $c = a \cdot b$ the rotation matrix is given by

$$R = I + [v]_\times + [v]_\times^2 \frac{1-c}{s^2} \tag{4}$$

where $[v]_x$ is the skew-symmetric cross-product matrix of v:

$$[v]_\times = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix} \tag{5}$$

We calculate the rotation matrix R that aligns the vector $\overrightarrow{ex\text{-}r\ ex\text{-}l}$ onto the x-axis. To apply the transformation, i.e. transforming the point cloud to be x-axis aligned, the dot product of the point cloud and the rotation matrix is calculated. Then, multiplying the x-values of the resulting matrix with -1 yields the mirrored point cloud. Although it is likely not the most efficient method for mirroring a point cloud, the technique can easily be reused for other applications that involve vector alignment, e.g. aligning faces completely by horizontal and vertical axis alignment.

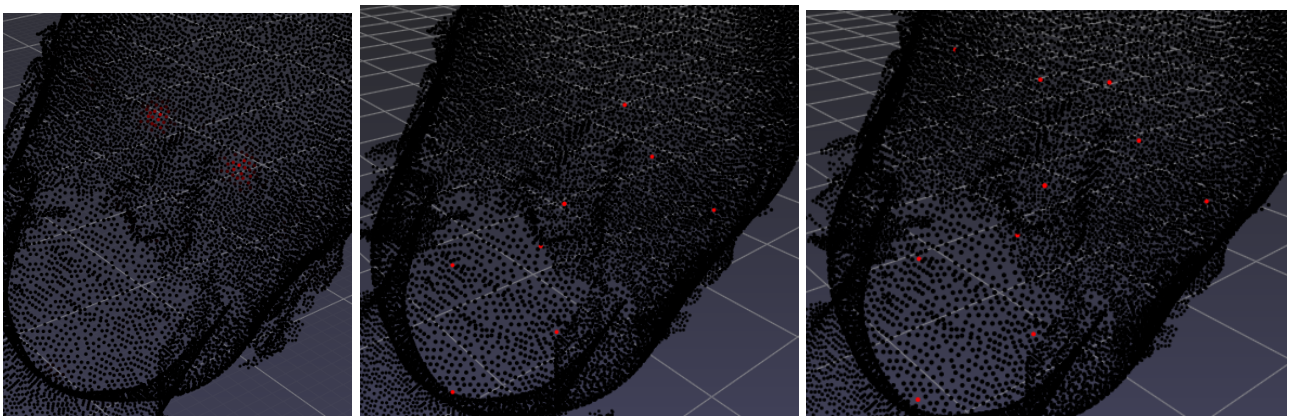
Note that the targets are defined point-wise, and a rotation does not alter the order of the point set. Therefore, the labels of the original sample can be reused for the horizontal flip augmentation.

3.10 Bilateral landmark issue

As explained, using HKS features brings the valuable consequence of rotation-invariance. Unfortunately, we observe an issue that makes the network unable to differentiate between bilateral landmarks. With this issue, the channels of bilateral landmarks such as the left and right exocanthion show identical predicted activations (see figure 7a). We only consider activations from the 'correct' half of the face for the point landmark extraction to fix the issue. To this end, a mid-face plane is constructed through the nasion, pronasale and subnasale. Generally, a plane can be uniquely identified given three points if they are non-collinear, meaning they do not lie on the same straight line. Plugging the point into the plane equation

$$ax + by + cz + d \tag{6}$$

and comparing the sign gives insight into where the point lies. We use the geometry module for SymPy [24], a Python library for symbolic mathematics to compute the plane equation. If the equation yields a positive value, the point is on the same side as the direction of the normal vector of the plane, which in this case is the left side of the face. Conversely, if the equation yields a negative value, the point lies on the opposite side, here the right side of the face. Plugging each point into the plane equation would result in unnecessarily long processing times. The points are arranged in descending order for each channel to keep the postprocessing times short. The point with the highest activation that lies on the correct side of the face is assigned the activation of 1.0 and thus considered the best point prediction for the landmark (see figure 7b for before and 7c after the fix.



(a) (b) (c)

Figure 7: **Bilateral landmarks issue.** a) Depiction of the predicted channel for the right endocanthion. Due to the bilateral landmark problem introduced with HKS features, the network falsely detects both the left and right endocanthion in the same channel. b) Point landmarks before the fix. c) Point landmarks after the fix.

Table 4: **Model configuration.** Models are trained on an Nvidia Titan V. *Diffusion is truncated to an Eigenbasis of k = 128 for spectral acceleration. **HKS is sampled at $F = 16$ values of t logarithmically spaced on [0.01, 1]

	Initial Network XYZ	Initial Network HKS	Refinement Networks
N Blocks	4	8	4
Block width D	256	356	128
F Input features	XYZ (+ RGB)	HKS**	XYZ
Dropout	0.1	0.1	0.1
Epochs	200	550	200
Initial LR	1e-3	1e-3	1e-3
LR decay factor	0.5 per 80	0.5 per 200	0.5 per 80
Optimizer	Adam	Adam	Adam
Size of eigenbasis k*	128	128	128
Batch size	1	1	1

4 Experimental setup

Multiple experiments are performed to assess the models for facial landmark detection. The experiments involve testing which input is the most effective (point cloud or mesh input), evaluating the refinement model's effectiveness in different configurations, and assessing which input features (XYZ or HKS) are more suitable for the problem. Finally, the data augmentation technique horizontal flipping and color features are added to the experiments. The exact model configurations can be found in table 4.

We train the refinement model in two different configurations: The first has lower jittering ($j = 3mm$) and a radius of the sphere of $r = 2.5cm$. For the second network, we increase jittering to $j = 6mm$ and slightly increase the sphere radius to $r = 3cm$ to ensure that the true landmark is still within that radius.

For each experiment, the model with the best accuracy is taken. During training, the models are evaluated on a validation set. The final evaluation in the next section is performed on a separate test set to ensure a reasonable estimate of a generalization error.

5 Results and Analysis

During training, the error metric used is the mean squared error between the actual and the predicted activation. While this is a valuable and differentiable metric for training the point-wise regression model, we are ultimately interested in the point landmarks. As described, the point landmarks are extracted by taking the point per channel with the maximum activation. Then, the results can easily be interpreted by calculating the difference between the actual and predicted landmarks. Because coordinates are given in millimeters, the difference is equivalent to the distance between the actual landmark coordinate and the predicted landmark coordinate in millimeters. We compute the mean per landmark and the total mean for presenting results. Figure 8 shows the visualization of an example with a relatively poor mean error of 4.40mm. The green dots are the actual, manually annotated landmarks and the red dots are the predicted landmarks.

The first experiments are performed on meshes and point clouds as input to compare the performance (see table 5). When using meshes, additionally to the vertices, also the face information is used by DiffusionNet to compute the operators. For the gradient computation, the 1-ring-neighbors are used instead of nearest neighbors for point clouds. Oddly, we observe a lower performance on meshes than point cloud input. The performance drop could be due to a faulty implementation but is not investigated further. Therefore, the remaining experiments are performed on the point cloud format. The training and evaluation are performed on the original noisy Headspace labels.

Table 5: **Point cloud versus mesh input.** The initial networks are trained on 1200 Headspace point clouds with the Headspace labels and XYZ features. Evaluating on Headspace labels instead of manual labels unsurprisingly yields better results because the network presented more similar targets during training and evaluation. In the experiments, training on meshes instead of point clouds led to higher errors, which is why the remaining experiments were performed on the point cloud format.

	Mean error in mm	
Landmark	Initial network (Point cloud)	Initial Network (Mesh)
Pogonion	4.08	5.35
Nasion	2.00	2.78
Pronasale	2.96	3.20
Subnasale	2.81	1.98
Exocanthion (right)	2.71	3.89
Endocanthion (right)	2.51	2.92
Endocanthion (left)	2.30	2.15
Exocanthion (left)	3.07	3.50
Cheilion (right)	2.71	3.46
Cheilion (left)	3.12	3.79
total	**2.825**	**3.302**

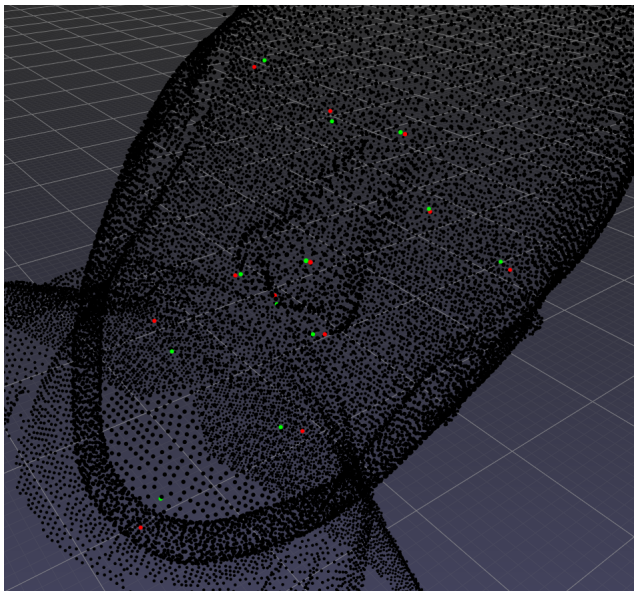


Figure 8: **Result example.** Initial network predictions (red) against ground truth (green). The ground truth is a manual annotation of a Headspace sample. Sample mean error: 4.40mm. Landmark errors: pg = 12.8mm, n = 1.3mm, prn = 1.1mm, ac-r = 1.8mm, sn = 3.4mm, ac-l = 3.3mm, ex-r = 3.8mm, en-r = 3.7mm, en-l = 0.9mm, ex-l = 4.0mm, ch-r = 10.2mm, ch-l = 6.4mm

The second round of experiments (see table 6) are performed to assess the refinement network and see if an improvement to the predictions of the initial network can be observed. The refinement network indeed manages to improve upon the initial network's accuracy by over 0.6mm.

Note that the training has been performed on the noisy Headspace labels, whereas the models were evaluated on the more accurate manual annotations, which explains why a drop of accuracy from 2.825mm to 3.731mm can be observed. This gives reason to suspect that the 'noise' in the Headspace labels follows a pattern that the previous model learns.

As a part of this experimental setup, we also test a model trained on the rotation-invariant HKS features. The model performs significantly worse with a mean error of 5.386mm. The model especially fails to detect the Pogonion well, with a mean error of 13.9mm. Table 8 reveals that in some cases, the prediction entirely fails with a maximum error of 11.0cm. This experiment is performed without alar curvature landmarks, thus the missing values in the table. This model, however, was not trained to its full potential because at 550 epochs, the training was cancelled due to long training times. As shown in the model configuration in table 4, a significantly more complex model is chosen with double the diffusion blocks and a higher block width. Still, likely, the model has not fully converged yet, and further training is necessary to judge its full potential. Another reason for the drop in performance might be the arising bilateral landmark issue, at least for the affected landmarks with a bilateral counterpart.

For the refinement model, taking three samples of random translation of the center point proves effective as a slight increase of mean accuracy from 3.164 to 3.089 was observed.

In the last round of experiments, the models are trained solely on the more accurate manual annotations. For the initial network, the data is augmented by horizontal flipping. Additionally, the input dimension F increases from 3 to 6 as RGB features are added to the input. With this setup, the initial network achieves a lower mean error of 2.779mm (see table 7. Refinement model (1) with lower jittering improves the accuracy to 2.548mm. Refinement model (2) with higher jittering achieves the best results with a mean error of 2.217mm.

Remarkably, we use the original landmark coordinates as the ground truth for calculating the results,

Table 6: **Refinement network assessment and HKS features.** The initial networks are trained on Headspace meshes (110 manually labelled + 1100 Headspace labels) with XYZ and HKS features. The refinement network is trained on 110 manually labelled 110 meshes with XYZ features. All networks are evaluated on 30 manually labelled Headspace meshes. The refinement network improves upon the initial network. The initial network trained on HKS clearly performs worse on pose normalized images. Maximum errors are listed in appendix D, table 8.

	Mean error ± std in mm		
Landmark	Initial Network (XYZ)	Refinement Network (1)	Initial Network (HKS)
Pogonion	7.70 ± 3.44	4.78 ± 2.32	13.90 ± 18.26
Nasion	2.76 ± 1.14	1.86 ± 0.62	3.31 ± 1.54
Pronasale	2.44 ± 1.16	1.86 ± 1.12	3.31 ± 1.65
Alar curvature (right)	2.54 ± 1.41	2.08 ± 1.30	-
Subnasale	2.87 ± 1.37	2.47 ± 1.23	3.28 ± 1.80
Alar curvature (left)	3.03 ± 1.35	2.66 ± 2.13	-
Exocanthion (right)	5.13 ± 2.04	5.01 ± 2.22	5.38 ± 2.50
Endocanthion (right)	3.34 ± 1.56	2.38 ± 1.44	4.25 ± 1.97
Endocanthion (left)	2.35 ± 1.63	2.63 ± 2.13	4.08 ± 1.73
Exocanthion (left)	4.55 ± 2.25	4.04 ± 2.28	5.72 ± 2.82
Cheilion (right)	4.80 ± 2.18	4.18 ± 2.56	5.71 ± 7.62
Cheilion (left)	3.22 ± 2.14	2.96 ± 2.11	4.73 ± 3.44
total	**3.731 ± 2.43**	**3.089 ± 2.18**	**5.386 ± 7.22**

Table 7: **Horizontal flipping and color features.** Both, the initial and refinement network are trained on 277 manually labelled Headspace point clouds. As before, the evaluation is done on 30 manually labelled Headspace samples. The initial network is trained on XYZ features enrichened with RGB information and horizontal flipping is performed during training to double the training set. These measures and more training samples with accurate labels achieve the highest accuracy in our experiments. The refined network slightly improves upon that accuracy. Maximum errors are listed in appendix D, table 9.

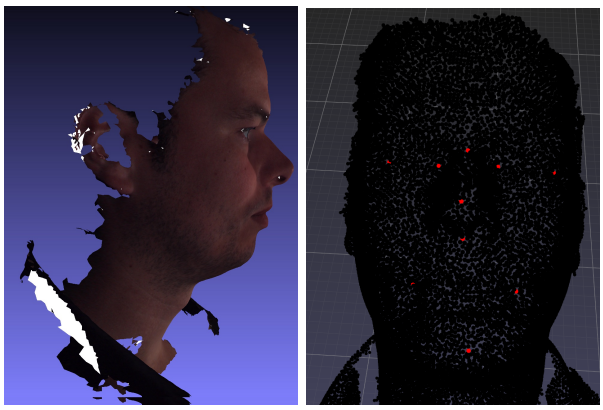
	Mean error ± std in mm		
Landmark	Initial network	Refinement network (1)	Refinement network (2)
Pogonion	3.33 ± 2.00	3.23 ± 1.99	2.68 ± 1.73
Nasion	2.24 ± 1.32	2.47 ± 1.91	2.50 ± 2.00
Pronasale	2.35 ± 1.27	1.50 ± 0.84	1.33 ± 0.84
Alar curvature (right)	1.99 ± 1.28	2.48 ± 1.87	2.29 ± 1.98
Subnasale	1.86 ± 0.84	1.53 ± 0.97	1.29 ± 0.89
Alar curvature (left)	2.05 ± 1.20	1.74 ± 0.98	1.70 ± 1.03
Exocanthion (right)	2.44 ± 1.76	2.28 ± 1.46	2.05 ± 1.30
Endocanthion (right)	2.54 ± 1.91	2.66 ± 1.77	1.92 ± 1.31
Endocanthion (left)	3.23 ± 1.74	3.26 ± 2.16	2.39 ± 1.70
Exocanthion (left)	2.69 ± 1.32	3.34 ± 2.12	2.95 ± 2.06
Cheilion (right)	3.61 ± 3.01	2.76 ± 1.72	2.70 ± 1.82
Cheilion (left)	5.00 ± 3.85	3.32 ± 2.36	2.83 ± 1.58
total	**2.779 ± 2.16**	**2.548 ± 1.87**	**2.217 ± 1.67**

not the recalculated point-wise targets that the networks are trained on. As section 3.2 describes, the targets are recalculated by picking the closest point to the original landmark. Consequently, even with a point-wise prediction an error of zero millimeters is practically impossible because the true landmark point is not sampled as an input point. The reason for that is twofold: Firstly, the landmarking in 3DMedX saves arbitrary coordinates and not pointers to vertices. Secondly, the initial network uses the downsampled mesh. Here, the targets are recalculated for the low-resolution point cloud. Only for the high-resolution point clouds with Headspace labels, the original landmarks are equivalent to the targets used as input to the networks, allowing predictions of zero millimeters to be recorded (but these experiments are not listed in the results in this work). Experiments show that the difference between the ground truth used for the final evaluation of the initial network and the low-resolution network targets cause an overestimation of the error of around 0.2mm. Thus, if the initial network would predict every landmark perfectly, the final evaluation would still yield an error of around 0.2mm due to the low-resolution approximation. Nonetheless, we consciously decide to evaluate the network in that manner to allow a fair comparison between the initial network and the refinement network. The initial network's performance is thus slightly better than the results might convey.

5.1 Radboudumc data

Despite the poor results of HKS compared to XYZ features on the Headspace data, we find that models trained on HKS features are more widely applicable to 3D meshes "in the wild". Such meshes are typically not consistently oriented. The initial network trained on XYZ features fails to learn meaningful features in such cases. Augmentation by random rotation helps prepare the network for inconsistently oriented faces, but the performance is still inferior. These findings indicate that the space of possible rotation in 3D is too large to be simulated with data augmentation. Therefore, we resort to training the networks on HKS features, making the network more difficult to train, but showing acceptable detection performance on the Radboudumc data. Unfortunately, no quantitative results are available because the data does not come with landmark annotations.

Figure 9b shows an example prediction of the initial network trained on HKS features.



(a) (b)

Figure 9: **Radboudumc prediction example.** (a) Side profile of the original mesh. The photo was captured by two cameras, capturing only the face. The head is oriented differently to the pose normalized samples in the training set. The lower mesh quality is also visible in the texture, demonstrated as white spots for faulty textures. (b) Frontal profile of the same subject showing predictions. At first glance, the predictions look seemingly accurate. Quantitative results on Headspace annotations indicate however, that the model trained on HKS is more inaccurate with a mean error over 5mm.

6 Discussion

The approach to regress point-wise scores demonstrates to be a suitable approach for the means of identifying point landmarks. The networks effectively learn regions around landmarks. The model generally recognizes the pattern of high activations in the center and lower activations with increasing distance. An advantage of the approach is that the intensity of the prediction can be interpreted as the model's confidence in the prediction. We see that the prediction cluster around the pogonion often has a lighter intensity reflecting the model's lower confidence in that landmark type (see figure 5). Especially in medical imaging, it can be helpful for the medical practitioner to be informed of how confident the model is in the prediction, hinting at how reliable the prediction is. Some other approaches, e.g. regressing landmark positions directly, do not allow such an assessment.

It is found that the initial network benefits from horizontal flipping and texture information. Unsurprisingly, it is preferred to train the network with accurate labels. Exchanging more samples with less accurate labels for less samples with more accurate labels does not appear worthwhile. The refinement model's accuracy can be increased by sampling various random jitterings, although at the cost of training time. Higher jittering with a slightly higher radius for the local region is preferred as a better refinement performance is achieved.

Introducing higher weights for points in landmark regions appears to be sufficient to combat class imbalance. Nevertheless, carefully tuning the values is expected to improve results. The activation scheme for the ground truth effectively models neighborhoods around landmarks. Similarly, choosing the radius more carefully can potentially facilitate network learning.

The refinement networks improve accuracy while only adding little time for inference. The initial model is slow but essential to give rough estimates of land-

mark regions. However, for scenarios where real-time inference is needed, it might be worthwhile to reduce the input resolution for the initial network to accelerate the rough predictions. Then, the fast refinement network might deliver sufficient detection accuracies for the task.

6.1 Limitations and future work

Significant variations in the input are a major challenge for the model. The network trained on craniums does not generalize very well to faces. Different head poses can be tackled by suitable input features such as HKS features, but the performance lacks behind pose normalized images trained on raw coordinates.

DiffusionNet requires precomputed operations which are rather time-intensive depending on the resolution. Especially during inference, this can be an issue because long waiting times hinder the practical usability of a landmarking tool for practitioners. The precomputation of operators involves different components. In this work, the gradient building component is rewritten to support CUDA-acceleration. Still, other components are computed on the CPU, which greatly slow down the precomputation.

The choice of DiffusionNet as a network architecture has several advantages. Firstly, the network is robust to a different sampling of the points. Sampling robustness is a critical component as it avoids unnecessary preprocessing. However, it is noteworthy to mention that we do not observe true sampling invariance, which Sharp et al. speak of in their paper. The initial network trained on low-resolution but predicting in high-resolution performs slightly worse caused by the sampling differences. Secondly, the network can deal with different geometrical representations. Although the 3D photos are represented as point clouds in this work, it is conceivable that a mesh representation has many benefits as the connectivity information can be used, e.g. for more accurate or more efficient computation of the gradient features.

Furthermore, the network run on HKS features produces the bilateral landmark issue. In this work, we avoid the problem by postprocessing, but the issue must be addressed on the network level to deal with the facial asymmetries of the subject properly. In the DiffusionNet paper, the authors describe how gradient features behave differently for segmenting bilaterally symmetric models. For consistently oriented normals, the model manages to disambiguate bilateral symmetry. However, for inconsistently oriented normals, the model fails to segment the left and right sides correctly. Point clouds are examples of a representation where normals are inconsistently oriented. According to the authors, a learned linear transformation A is used to learn pairwise products of feature gradients. Inner products are invariant to rotations. However, the inner product is scaled by the matrix A with dimension $D \times D$. Choosing A as a complex matrix allows rotating and scaling gradients, creating sensitivity to the orientation. Real A only allows scaling. A should be restricted to real values for point clouds due to inconsistently oriented normals. These findings indicate that models trained on meshes and HKS features might intrinsically succeed in disambiguating bilateral landmarks.

Unfortunately, due to a lack of time, no proper ablation study for the different network components are conducted in this work. It is desirable to analyze the effect on performance by adding components, such as examining the effect of added RGB features or added horizontal flip in isolation. The results presented here lack fair comparability caused by different training setups with a varying number of samples and different annotation qualities.

Regarding the annotations, it could be helpful to apply transfer learning by pretraining the model on many samples with noisy annotations and then tuning or retraining the model by training fewer samples with accurate annotations.

It would be helpful to study networks that support rigid invariance. When labels for the Radboudumc data are available, the model trained on HKS features can be evaluated quantitatively on craniums and faces without consistent orientation. With the rough landmarks, the subject can be aligned for a more accurate detection with the refinement model.

Lastly, the approach of point-wise regression may be suboptimal for solving facial landmark detection. Perhaps it is more effective for the initial model to segment regions around landmarks and then predict scores for the most likely landmark candidate. It is also imaginable that vastly different architectures produce superior results.

7 Conclusion

In this work, a facial landmark detection network is developed based on DiffusionNet. An initial network predicts rough landmark positions in low resolution, which can be tweaked by a refinement model that operates in full resolution. With a mean error of 2.22mm, the landmarks detector shows promising results and makes only slightly more inaccurate predictions than a human annotator does. However, the model is limited to consistent head orientations. If that requirement is not met, a model trained on HKS features can enable rotation invariance but shows inferior detection accuracies.

It is recommended to further research methods to promote rigid invariance as it has high clinical relevance. 3D deep learning is a challenging field that only recently is receiving increasing attention from the research community. Current methods have promising potential and better prevent overfitting on the mesh sampling and data representation.

References

[1] J. M. Plooij et al. "Evaluation of reproducibility and reliability of 3D soft tissue analysis using 3D stereophotogrammetry". In: *International Journal of Oral and Maxillofacial Surgery* 38 (3 2009). ISSN: 09015027. DOI: 10.1016/j.ijom.2008.12.009.

[2] F. Ras et al. "Quantification of facial morphology using stereophotogrammetry - Demonstration of a new concept". In: *Journal of Dentistry* 24 (5 1996). ISSN: 03005712. DOI: 10.1016/0300-5712(95)00081-X.

[3] Hang Dai et al. "Statistical Modeling of Craniofacial Shape and Texture". In: *International Journal of Computer Vision* 128.2 (Nov. 2019), pp. 547–571. ISSN: 1573-1405. DOI: 10.1007/s11263-019-01260-7. URL: https://doi.org/10.1007/s11263-019-01260-7.

[4] Xiangxin Zhu and Deva Ramanan. "Face detection, pose estimation, and landmark localization in the wild". In: 2012. DOI: 10.1109/CVPR.2012.6248014.

[5] Yulan Guo et al. *Deep Learning for 3D Point Clouds: A Survey*. 2021. DOI: 10.1109/TPAMI.2020.3005434.

[6] Martin Köstinger et al. "Annotated facial landmarks in the wild: A large-scale, real-world database for facial landmark localization". In: 2011. DOI: 10.1109/ICCVW.2011.6130513.

[7] Nicholas Sharp et al. "DiffusionNet: Discretization Agnostic Learning on Surfaces". In: *ACM Trans. Graph.* 01.1 (2022).

[8] Avinash S. Bidra et al. "The relationship of facial anatomic landmarks with midlines of the face and mouth". In: *Journal of Prosthetic Dentistry* 102 (2 2009). ISSN: 00223913. DOI: 10.1016/S0022-3913(09)60117-7.

[9] Romuald Perrot, Pascal Bourdon, and David Helbert. "Implementing Cascade of Regression-based Face Landmarking: an in-Depth Overview". In: *Image and Vision Computing* 102 (Oct. 2020), p. 103976. DOI: 10.1016/j.imavis.2020.103976. URL: https://hal.archives-ouvertes.fr/hal-02884592.

[10] Yongzhe Yan et al. "A survey of deep facial landmark detection". In: *RFIAP*. Paris, France, June 2018. URL: https://hal.archives-ouvertes.fr/hal-02892002.

[11] Saifullahi Aminu Bello et al. *Review: Deep learning on 3D point clouds*. 2020. DOI: 10.3390/rs12111729. arXiv: 2001.06280.

[12] Daniel Maturana and Sebastian Scherer. "VoxNet: A 3D Convolutional Neural Network for real-time object recognition". In: vol. 2015-December. 2015. DOI: 10.1109/IROS.2015.7353481.

[13] Zhirong Wu et al. "3D ShapeNets: A deep representation for volumetric shapes". In: vol. 07-12-June-2015. 2015. DOI: 10.1109/CVPR.2015.7298801.

[14] Christopher B. Choy, JunYoung Gwak, and Silvio Savarese. "4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks". In: *CoRR* abs/1904.08755 (2019). arXiv: 1904.08755. URL: http://arxiv.org/abs/1904.08755.

[15] Hang Su et al. "Multi-view Convolutional Neural Networks for 3D Shape Recognition". In: *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 945–953. DOI: 10.1109/ICCV.2015.114.

[16] Charles R. Qi et al. "PointNet: Deep learning on point sets for 3D classification and segmentation". In: *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*. 2017. ISBN: 9781538604571. DOI: 10.1109/CVPR.2017.16. arXiv: 1612.00593.

[17] Charles R. Qi et al. "PointNet++: Deep hierarchical feature learning on point sets in a metric space". In: *Advances in Neural Information Processing Systems*. 2017. arXiv: 1706.02413.

[18] Yangyan Li et al. "PointCNN: Convolution on X-transformed points". In: *Advances in Neural Information Processing Systems*. 2018.

[19] Rana Hanocka et al. "MeshCNN". In: *ACM Transactions on Graphics* (2019). ISSN: 0730-0301. DOI: 10.1145/3306346.3322959.

[20] 3D Lab Radboudumc Nijmegen. *3DMedX® (v1.2.23.0), The all-in-one solution for 3D research and the home of the OrthoGnathicAnalyser*. URL: https://www.3dmedx.nl (visited on 04/02/2022).

[21] Guido de Jong. *Lecture notes: 3D Mesh Processing and Analysis, 3D Computer vision for Medical Applications*. Oct. 2021.

[22] Paolo Cignoni et al. "MeshLab: an Open-Source Mesh Processing Tool". In: *Eurographics Italian Chapter Conference*. Ed. by Vittorio Scarano, Rosario De Chiara, and Ugo Erra. The Eurographics Association, 2008. ISBN: 978-3-905673-68-5. DOI: 10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136.

[23] Adam Paszke et al. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[24] Aaron Meurer et al. “SymPy: symbolic computing in Python”. In: *PeerJ Computer Science* 3 (Jan. 2017), e103. ISSN: 2376-5992. DOI: 10.7717/peerj-cs.103. URL: https://doi.org/10.7717/peerj-cs.103.

A Explanation of important concepts

Rigid deformation Deformation describes the transformation of an object from an initial to some final geometry. In contrast to non-rigid deformation, a rigid deformation does not change the position and orientation of the object relative to the internal reference frame. Rotation around an axis is an example of a rigid operation that changes the configuration of the points relative to the external but not to the internal reference frame. Translation is another rigid transformation because it only affects the external reference frame, as the points within the object are all moved along parallel paths to the axis.

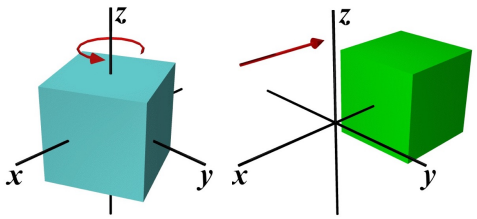


Figure 10: **Rigid transformations.** Rotation (left), translation (right). Images from [1].

Non-rigid deformation Non-rigid deformations can affect points within the object relative to both the internal and external reference frame. Distortion is an example of a non-rigid operation that changes the spacing of points within the object and consequently changes the object's overall shape. Dilation or scaling is another non-rigid operation that changes the object's volume, but differently from distortion, retains the same shape.

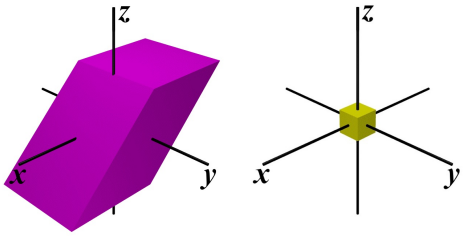


Figure 11: **Non-rigid transformations.** Distortion (left), dilation (right). Images from [1].

Shape descriptor A shape descriptor characterizes the local geometry of the surface. In other words, they describe a point's role on the surface. Examples for shape descriptors are the Gaussian curvature $K(x) = \kappa_1(x)\kappa_2(x)$ and the mean curvature $H(x) = \kappa_1(x) + \kappa_2(x)$. Good shape descriptors are robust to noise in the triangulation and against small deformations. They should also be invariant under rigid transformation and other isometries. [2]

Heat Kernel Signature The Heat Kernel Signature (HKS) is a popular shape descriptor derived from the Laplacian.

For a fixed time t, it is defined as

$$HKS(x) = k_t(x, x) = \sum_i e^{-\lambda_i t} \phi_i(x)^2. \tag{7}$$

The time describes, how locally or globally the shape is captured from a given point (see figure 12).

There are different shape descriptors, such as the Wave Kernel Signature (WKS) similar to HKS but based on the Schrödinger wave equation. Each has its advantages depending on the application, but both are invariant to isometric transformations invariant and can be computed efficiently. Furthermore, HKS is also stable under small perturbations to the shape.

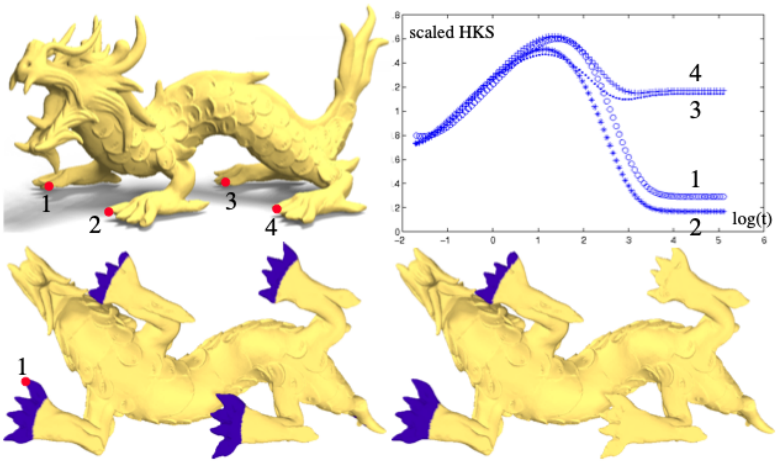


Figure 12: **Heat Kernel Signature.** At four different points on a triangulated surface. For small values of t ($t < 1$), the heat kernel signature $k_t(x, x)$ is almost the same. This is, because the local geometry, i.e. the tips of the dragon's feet, is approximately equivalent. As t increases ($t > 1$), more global information is considered, and the heat kernel signatures diverge. The two points at the front and back feet still capture more common surface information about the shape than a point at the front and at the back. Image from [3].

Manifold meshes Generally speaking, a manifold is a topology that locally resembles Euclidean space. In computer graphics, often, triangular meshes are used as they are the simplest form of polygon meshes and require fewer computations for computer graphical operations. Many applications require the reconstructed triangular mesh surface to be a watertight manifold surface with correct topology. Watertightness is fulfilled if the mesh does not have holes or missing triangles. Formally, a mesh is manifold if every edge in the mesh is either a boundary edge or a manifold edge, whereas a boundary edge is part of exactly one face and a manifold edge of two faces. Moreover, it does not include non-manifold vertices, i.e. vertices where the corresponding star (= union of all incident faces) is not connected when removing the vertex. See figure 13 for an illustration of a non-manifold vertex and non-manifold edge. Fixing the mesh to be manifold or watertight is called mesh healing or mesh repairing.

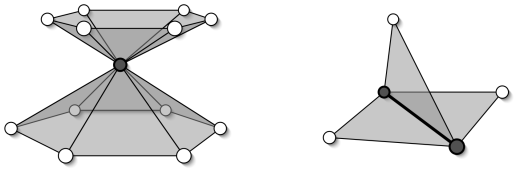


Figure 13: **Non-manifold geometry.** Non-manifold vertex (left), non-manifold vertex (right). Image from [4].

B Background for choice of the approach

B.1 Network

The project started with exploring different networks that can tackle the problem of 3D landmark detection. This phase also led to insights regarding networks that do not work well for the problem. PointNet is one of the earliest and more straightforward model architecture that operates on point clouds was a straightforward choice. The first attempts involved the Pytorch implementation of the extension of PointNet, called PointNet++. However, experiments on this network were difficult because of the very high memory consumption of the network architecture, indicating poor scalability. Even for decimated meshes, the input images were still too big for the approach of predicting point-wise activations per landmark channel.

The next attempt to tackle the problem was MeshCNN, which showed better scaling properties and lower memory consumption in our experiments. The motivation to use a network that processes meshes directly

instead of point clouds is that meshes include additional surface information that can better capture the surface structure. Specifically, MeshCNN combines specialized convolution and pooling layers that operate on the edges of the mesh. While the first results of a network that performs direct coordinate regression were promising, one big drawback prevents using MeshCNN in practice: the network requires mesh watertightness or manifoldness. Repairing the meshes is usually a manual task in a 3D computer graphics software such as Blender [5] or Meshlab [6] that cannot easily be automized inside a preprocessing pipeline. Therefore, it is not feasible to apply MeshCNN on unseen meshes, e.g. the meshes from the control samples from the Radboudumc data set.

B.2 Direct coordinate regression vs point-wise regression

A straightforward way to learn the XYZ coordinates of landmarks is to build a regression network that predicts the coordinates directly. However, experiments performed on the MeshCNN network suggested that this approach tends to overfit the average face structure observed in the training set, as opposed to learning surface-based features (see figure 14). Notably, the set of possible predictions also includes all points in space, not restricted to points on the mesh surface. A point-wise approach circumvents this problem by predicting a score for each point, thus only allowing landmark predictions located on the surface of the mesh. Besides, a point-wise approach enables the learning of point clusters that represent a region in proximity of a landmark.

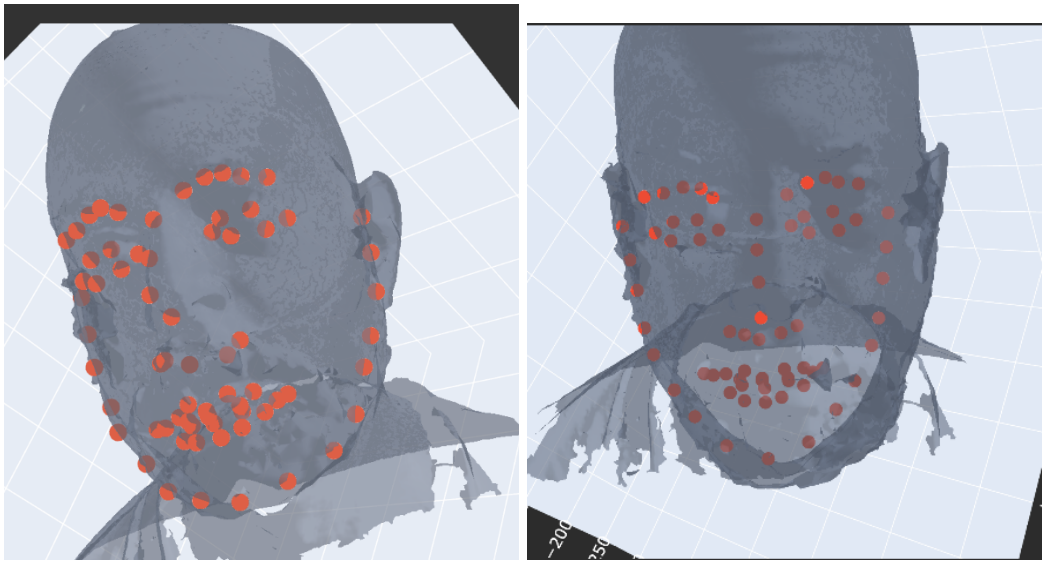


Figure 14: **Direct coordinate regression with MeshCNN.** Left: ground truth. Right: MeshCNN predictions. The overall facial shape is decently captured, but the predictions are very far off. MeshCNN directly operates on the meshes, making it unnecessary perform point sampling. Nonetheless, considerable mesh simplification is required to maintain a manageable memory footprint. Note, that in an earlier version 68 landmarks were processed. However, because most of them are non-anatomical and ill-defined, they were later discarded.

C Manual annotation

We quickly reach a threshold where not the approach and network architecture but the poor annotation quality limit detection performance. Thus, 307 Headspace samples are manually annotated in 3DMedX. The samples are labelled by two annotators, each annotating roughly half of them. Therefore, it is to be expected that there is a small interobserver variability among the annotations. Specific samples are left out if they meet one of the criteria: Firstly, the photo is left out if the subject has a moderate or heavy beard because it is hard to define the correct landmark positions of pogonion, left cheilion and right cheilion for such subjects. Secondly, the photo is left out if no texture information is available for the corresponding mesh. The third criterion is if artefacts in the mesh prevent the annotator from properly labelling one of the landmarks. This can happen in rare cases where the mesh has a prominent chin hole. After the annotation process, the average variability between the Headspace and manual labels is measured at 3.81mm. This number suggests that there is indeed an inaccuracy in the Headspace labels that can prevent the model from learning the correct facial features.

D Results details

Table 8: **Maximum errors for table 6: Refinement network assessment and HKS features.**

	Maximum error in mm		
Landmark	Initial Network (XYZ)	Refinement Network (1)	Initial Network (HKS)
Pogonion	12.78	8.82	110.15
Nasion	5.02	3.16	7.68
Pronasale	5.78	5.07	7.74
Alar curvature (right)	4.89	6.58	-
Subnasale	6.94	5.97	7.53
Alar curvature (left)	6.13	11.05	-
Exocanthion (right)	8.94	8.71	10.27
Endocanthion (right)	6.56	6.57	8.84
Endocanthion (left)	7.73	9.39	7.73
Exocanthion (left)	9.26	9.20	11.36
Cheilion (right)	10.20	9.84	44.81
Cheilion (left)	8.06	8.37	16.82
total	**12.78**	**11.05**	**110.15**

Table 9: **Maximum errors for table 7: Horizontal flipping and color features.**

	Maximum error in mm		
Landmark	Initial network	Refinement network (1)	Refinement network (2)
Pogonion	9.31	8.21	8.21
Nasion	6.06	7.87	10.12
Pronasale	5.38	4.04	4.03
Alar curvature (right)	5.42	7.58	8.60
Subnasale	4.06	5.36	4.28
Alar curvature (left)	5.00	3.89	5.27
Exocanthion (right)	6.96	6.10	5.01
Endocanthion (right)	8.57	7.15	5.58
Endocanthion (left)	8.06	7.93	6.47
Exocanthion (left)	6.48	8.65	9.74
Cheilion (right)	15.26	7.25	7.25
Cheilion (left)	15.48	11.20	6.42
total	**15.48**	**11.20**	**10.12**

References

[1] San Diego State University. , *Visualizing Strain*. URL: http://www.sci.sdsu.edu/visualstructure/vss/htm_hlp/def_d.htm (visited on 04/02/2022).

[2] Stanford University Justin Solomon. *CS 468 Lecture 16:, Isometry Invariance and Spectral Techniques*. URL: https://graphics.stanford.edu/courses/cs468-13-spring/assets/lecture16-gawlik.pdf (visited on 04/02/2022).

[3] Jian Sun, Maks Ovsjanikov, and Leonidas Guibas. "A concise and provably informative multi-scale signature based on heat diffusion". In: vol. 28. 2009.

[4] Mario Botsch et al. *Polygon Mesh Processing*. 2010. DOI: 10.1201/b10688.

[5] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation. Stichting Blender Foundation, Amsterdam, 2018. URL: http://www.blender.org.

[6] Paolo Cignoni et al. "MeshLab: an Open-Source Mesh Processing Tool". In: *Eurographics Italian Chapter Conference*. Ed. by Vittorio Scarano, Rosario De Chiara, and Ugo Erra. The Eurographics Association, 2008. ISBN: 978-3-905673-68-5. DOI: 10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136.