# Reg-PointNet++: A CNN Network Based on PointNet++ Architecture for 3D Reconstruction of 3D Objects Modeled by Supershapes

Hassnae Remmach[1,*], Raja Mouachi [1], Mohammed Sadgal [2], and Aziz El Fazziki [2]
[1] LAMIGEP, EMSI Moroccan School of Engineering, Marrakesh, Morocco; Email: r.mouachi@emsi.ma (R.M.)
[2] Department of Computer Science, Computer Systems Engineering Laboratory, Cadi Ayyad University, Marrakesh, Morocco; Email: sadgal@hotmail.com (M.S.), elfazziki@uca.ma (A.E.F.)
*Correspondence: h.remmach@emsi.ma (H.R.)

***Abstract*—The use of 3D reconstruction in computer vision applications has opened up new avenues for research and development. It has a significant impact on a range of industries, from healthcare to robotics, by improving the performance and abilities of computer vision systems. In this paper we aim to improve 3D reconstruction quality and accuracy. The objective is to develop a model that can learn to extract features, estimate a Supershape parameters and reconstruct 3D directly from input points cloud. In this regard, we present a continuity of our latest works, using a CNN-based Multi-Output and Multi-Task Regressor, for 3D reconstruction from 3D point cloud. We propose another new approach in order to refine our previous methodology and expand our findings. It is about "Reg-PointNet++", which is mainly based on a PointNet++ architecture adapted for multi-task regression, with the goal of reconstructing a 3D object modeled by Supershapes from 3D point cloud. Given the difficulties encountered in applying convolution to point clouds, our approach is based on the PointNet ++ architecture. It is used to extract features from the 3D point cloud, which are then fed into a Multi-task Regressor for predicting the Supershape parameters needed to reconstruct the shape. The approach has shown promising results in reconstructing 3D objects modeled by Supershapes, demonstrating improved accuracy and robustness to noise and outperforming existing techniques. Visually, the predicted shapes have a high likelihood with the real shapes, as well as a high accuracy rate in a very reasonable number of iterations. Overall, the approach presented in the paper has the potential to significantly improve the accuracy and efficiency of 3D reconstruction, enabling its use in a wider range of applications.**

***Keywords*—3D reconstruction, Convolution Neural Network (CNNs), multi-output regressor, multi-task regressor, 3D point cloud, Supershapes, PointNet, PointNet++, deep learning**

## I. Introduction

The task of 3D reconstruction is a crucial component of computer vision, as it facilitates the creation of precise 3D models of various objects and scenes using different types of data. The role of 3D reconstruction is crucial in various fields such as medical imaging, facial recognition [1], robotics, computer graphics, and many more. In medicine, 3D reconstruction plays a vital role in diagnosis and surgical planning, enabling doctors to examine and analyze the internal structures of a patient's body. In robotics, 3D reconstruction helps robots to perceive their environment, enabling them to navigate and perform tasks autonomously. In computer graphics, 3D reconstruction is used to create lifelike virtual environments and characters for movies, games, and simulations. Overall, 3D reconstruction plays an important role in advancing our understanding and application of various fields.

3D reconstruction can also be used to enhance computer vision systems, enabling them to better perceive and understand the 3D structure of objects and scenes.

3D reconstruction and computer vision are interdependent fields that work together to advance our understanding and application of visual information. However, 3D reconstruction is a complex and challenging task that requires sophisticated algorithms and specialized techniques to achieve accurate results. One of the main difficulties in 3D reconstruction is dealing with noisy and incomplete data, which can lead to inaccuracies in the reconstructed 3D models. Additionally, the process of 3D reconstruction often involves integrating data from multiple sources, such as images, or point clouds, or even sketches [2], which requires careful calibration and alignment to ensure consistency and accuracy in the final 3D model. Despite these challenges, advances in computer vision and machine learning have enabled significant progress in the field of 3D reconstruction, with new methods and techniques continually emerging to enhance the accuracy and efficiency of the process.

In recent years, there has been a growing interest in using deep learning approaches for 3D reconstruction. This is due to the impressive capabilities of Artificial Intelligence (AI) algorithms, such as Convolutional Neural Networks (CNNs) [3] and Recurrent Neural Networks (RNNs) [4], in handling complex data and learning high-level features that are useful for 3D reconstruction. These

Manuscript received May 31, 2023; revised June 12, 2023; accepted July 11, 2023.

deep learning approaches have shown significant improvements in the accuracy and efficiency of 3D reconstruction, as they can learn to extract relevant features from images, point clouds, or other types of data, and use them to construct 3D models. Additionally, deep learning models can handle noisy and incomplete data more robustly than traditional methods, making them well-suited for real-world applications. However, deep learning approaches for 3D reconstruction still face many challenges, such as the need for large amounts of high-quality training data, as well as the potential for overfitting and other issues. Nonetheless, the promise of deep learning in 3D reconstruction is driving significant research and development efforts in the field.

Our work is a contribution to the ongoing research in using deep learning for 3D reconstruction. Specifically, we focus on modeling 3D objects using a parametrized surface, known as Supershapes [5]. Supershapes are a class of geometric shapes that can be defined by a small set of parameters, making them well-suited for use in deep learning models. By using Supershapes, we aim to improve the accuracy and efficiency of the 3D reconstruction process.

Our contribution involves building upon our previous research, which presented a CNN-based multi-output regressor and a CNN-based multi-task regressor [6], using PointNet [7] as base architecture. This time around, we propose to apply a multi-task regressor based on the base architecture of PointNet++ [8]. By doing so, we aim to further enhance the efficiency and accuracy of our 3D reconstruction approach.

In our previous research, we demonstrated the effectiveness of a CNN-based multi-output/multi-task regressor for estimating the Supershape parameters required for 3D reconstruction. This approach allowed us to estimate multiple parameters simultaneously, improving the efficiency and accuracy of the reconstruction process. However, the use of a CNN-based approach also presented some limitations in terms of computational complexity.

To address this limitation, we propose to adapt the base architecture of PointNet++ to perform multi-task regression. By doing so, we can leverage the power of PointNet++ to perform feature extraction from 3D point clouds and use it in conjunction with a multi-task regressor to estimate Supershape parameters. This approach is expected to improve the efficiency and accuracy of the 3D reconstruction process by reducing computational complexity. And to address the limitations of the PointNet architecture, we have focused our attention on the PointNet++ architecture. PointNet++ is an improved version of the PointNet architecture that introduces a hierarchical feature learning approach to 3D point clouds. This approach allows for the efficient processing of large-scale point clouds and improves the accuracy of feature extraction. By leveraging the advantages of PointNet++, we aim to improve the accuracy and efficiency of our 3D reconstruction approach.

We will adapt and adjust the PointNet++ architecture to perform multi-task regression, similar to our previous approach with PointNet. This will involve modifying the classification network in PointNet++ to a multi-task regressor that can estimate Supershape parameters.

By combining the advantages of PointNet++ with multi-task regression, we believe that we can significantly improve the efficiency and accuracy of our 3D reconstruction approach. This is a crucial step towards the development of more advanced and efficient 3D reconstruction techniques for a range of applications.

The article has four main parts. First is the literature review, summarizing the latest in CNNs with a focus on PointNet and PointNet++ and comparing their strengths and weaknesses. The second part, methodology, explains the study components, introduces Supershapes, and details PointNet++ architecture, including multitask learning. The third part shares experimental results, discussing datasets, metrics, and comparing with previous work. The last part concludes findings, highlights contributions, and suggests future research directions.

## II. Literature Review

The field of 3D reconstruction has experienced a revolutionary transformation thanks to the advancements in deep learning techniques. These techniques have brought about significant improvements in the accuracy and efficiency of reconstructing complex objects and scenes. One pivotal milestone in the evolution of deep learning for 3D reconstruction can be attributed to the use of Convolutional Neural Networks (CNNs) for tasks such as 3D shape classification and segmentation.

The ability of CNNs to process point clouds has opened up new avenues for advancing the field of 3D reconstruction and enhancing its practical applications. By effectively analyzing and interpreting the spatial relationships between points, CNNs can extract meaningful features from point clouds, enabling accurate and efficient processing of 3D shape data.

Consequently, CNNs have been extensively applied in various domains, including object detection, recognition, and pose estimation, by leveraging the information contained within point cloud data.

### *A. State of Art: CNNs*

In the realm of 3D reconstruction, two notable advancements in deep learning techniques are PointNet and PointNet++. These architectures have made significant contributions to the processing and analysis of point cloud data.

PointNet [7] was one of the pioneering architectures that introduced the concept of directly processing unordered point cloud data with a CNN. It treats each point as an individual input and uses shared MLPs to extract local features. PointNet then aggregates these features to obtain a global feature representation of the entire point cloud. PointNet has been applied to tasks like object classification, semantic segmentation, and shape retrieval.

As an extension of PointNet, PointNet++ [8] introduced a hierarchical neural network architecture for point cloud analysis. It aims to capture both local and global contextual information by employing a series of Set Abstraction (SA) and Feature Propagation (FP) layers. SA layers downsample the input point cloud by selecting representative points and extracting local features. FP layers then upsample and propagate refined features to higher-resolution layers. PointNet++ has demonstrated improved performance in tasks such as semantic segmentation, object part segmentation, and shape classification.

Dynamic Graph CNN (DGCNN) [9] is also one of the notable CNN-based architectures specifically designed for point cloud processing. It leverages the concept of graphs to process point clouds. It constructs a k-nearest neighbor graph for each point in the point cloud and employs EdgeConv layers to aggregate information from neighboring points. The EdgeConv layers operate on the graph structure and update the features of each point based on its neighbors, capturing local geometric relationships. DGCNN has been successful in tasks like object classification, part segmentation, and scene classification.

PointCNN [10] introduces a convolution operation for point cloud data, that is independent of local order.

It defines a permutation-invariant kernel function that operates on local neighborhoods of points. PointCNN leverages this kernel function to perform convolutions on the point cloud, capturing local patterns and preserving permutation invariance. This architecture has been applied to tasks like object classification, semantic segmentation, and part segmentation.

And, the Kernel Point Convolution (KPConv) [11] combines the concept of point clouds with kernel convolutions. It represents each point as a kernel with learnable weights and defines convolutions by measuring the influence of each kernel on its neighboring points. KPConv enables efficient and flexible convolutions on point cloud data and has shown promising results in tasks like object segmentation, semantic segmentation, and 3D shape completion.

### *B. Comparative Analysis*

The performance of CNN-based architectures for point cloud processing can vary depending on the specific task, dataset, and evaluation metrics. It's important to note that the performance of these architectures is often evaluated based on different benchmarks and datasets, making direct comparisons challenging. However, here are a few considerations regarding the performance of some popular architectures:

PointNet++ [8] has demonstrated improved performance compared to its predecessor, PointNet, in various tasks such as semantic segmentation, object part segmentation, and shape classification. Its hierarchical architecture allows it to capture both local and global contextual information, leading to enhanced performance in capturing fine-grained details and understanding complex 3D structures.

KPConv [11] has shown impressive performance in tasks like object segmentation, semantic segmentation, and 3D shape completion. Its ability to efficiently perform kernel convolutions on point cloud data and adapt to varying point densities contributes to its effectiveness. KPConv has achieved state-of-the-art results on several benchmarks, showcasing its performance in handling point cloud data.

DGCNN [9] has been successful in tasks like object classification, part segmentation, and scene classification. By incorporating graph structures and utilizing EdgeConv layers to capture local geometric relationships, DGCNN has achieved competitive performance in various point cloud processing tasks.

It's worth mentioning that the performance of these architectures can also depend on factors such as dataset size, complexity, and the availability of labeled data for training. Furthermore, advancements in these architectures and the introduction of newer models can influence performance. It is recommended to consult the latest research and benchmark evaluations to obtain more specific and up-to-date information on the performance of CNN-based architectures for point cloud processing.

### *C. PointNet Vs PointNet++*

PointNet [7] is a pioneering deep learning architecture that operates directly on unordered point clouds. Unlike traditional methods that require structured inputs, PointNet can take in raw point cloud data without any predefined ordering or connectivity. It leverages a symmetric function to process individual points independently and then aggregates their features to obtain a global representation of the entire point cloud. This holistic approach enables PointNet to learn powerful features and effectively perform tasks such as object recognition, segmentation, and classification on point cloud data.

Building upon PointNet's success, PointNet++ [8] further extends the capabilities of deep learning networks for point cloud analysis. PointNet++ addresses one of the limitations of PointNet, which is the lack of local contextual information. PointNet++ introduces a hierarchical neural network architecture that gradually builds a more detailed and contextual understanding of the point cloud data. It achieves this by employing a set of PointNet modules at different scales, where each module processes a subset of points and captures local structures. These hierarchical modules are designed to capture both local and global information, allowing PointNet++ to achieve enhanced performance in tasks such as segmentation, object part classification, and scene understanding.

Both PointNet and PointNet++ have significantly advanced the field of 3D reconstruction by enabling direct processing of unordered point cloud data. These architectures have proven effective in capturing essential features and contextual information from point clouds, leading to improved accuracy and efficiency in various applications. With ongoing research and development, it is likely that these architectures will continue to evolve, providing even more powerful tools for analyzing and reconstructing 3D data.

PointNet and PointNet++ have demonstrated robustness to variations in point cloud inputs, such as different point densities, rotations, and translations. This adaptability allows them to handle diverse and real-world point cloud data, making them suitable for applications where the quality and characteristics of the input data may vary. Both architectures exhibit scalability, enabling them to handle point clouds with varying numbers of points. This flexibility is essential when dealing with complex scenes or large-scale datasets, as it ensures efficient processing and analysis without sacrificing performance.

PointNet and PointNet++ have showcased generalization capabilities, meaning they can learn representations that are applicable across different datasets and tasks. This generalizability allows these architectures to be utilized in a wide range of applications and domains, providing flexibility and versatility in various 3D reconstruction scenarios. They have been particularly effective in semantic segmentation tasks, where the objective is to assign semantic labels to individual points in a point cloud. By considering local and global features, these architectures can accurately classify and segment points, enabling more detailed and comprehensive understanding of 3D scenes.

PointNet and PointNet++ have also shown promise in object detection and pose estimation tasks. By leveraging their ability to capture global features and local contextual information, these architectures can effectively identify and localize objects within a point cloud, contributing to applications such as robotics, augmented reality, and autonomous driving. While initially designed for point cloud data, the concepts and principles underlying PointNet and PointNet++ have been extended to other modalities, such as 3D meshes and volumetric data. This adaptability highlights the potential for these architectures to contribute to the analysis and reconstruction of 3D data in various forms.

Overall, PointNet and PointNet++ have not only introduced groundbreaking architectures for point cloud analysis but have also inspired further exploration and innovation in the field of 3D reconstruction. Their robustness, scalability, generalizability, and applicability to various tasks have solidified their significance and paved the way for advancements in understanding and utilizing 3D data.

## III. Methodology

### A. The Parametric Surfaces: Supershapes

In the realm of 3D modeling, Supershapes which is introduced by the mathematician Johan Gielis [5]; enable the creation of parametrized surfaces and objects with complex and fascinating geometries.

The concept behind Supershapes lies in defining a set of adjustable parameters that can modify the shape of an object or surface. *A* Supershape $S$ can be expressed as follow [5]:

$$S = [a, b, m, M, n_1, n_2, n_3, N_1, N_2, N_3] \quad (1)$$

$m$: Symmetry parameter, this parameter controls the number of radial arms in the Supershape.

$M$: Symmetry parameter, this parameter controls the number of repeating segments around the Supershape.

$a$, $b$: The scaling parameters that control the overall size of the shape.

$n_1$, $n_2$, $n_3$, $N_1$, $N_2$, $N_3$: The shape parameters coefficients.

By manipulating these parameters, e.g., Table I, one can achieve a wide range of visually appealing and diverse shapes, e.g., Fig. 1. The Supershape formula serves as a mathematical framework to generate these shapes and surfaces by assigning appropriate values to the parameters.

TABLE I. Supershapes Parameters Represented in Fig. 1

| *S* | m | $n_1$ | $n_2$ | $n_3$ | M | $N_1$ | $N_2$ | $N_3$ |
|---|---|---|---|---|---|---|---|---|
| **(a)** | 4 | 10 | 10 | 10 | 4 | 10 | 10 | 10 |
| **(b)** | 8.52 | 33 | 21.39 | −33.40 | 2.57 | 0.10 | −5.13 | 1.71 |
| **(c)** | 7.50 | 1.80 | 1.50 | 0.90 | 16.20 | 1 | 1.60 | 1.10 |
| **(d)** | 6 | 3 | 7 | 6 | 2 | 6 | 1 | 3 |
| **(e)** | 8 | 8 | 10 | 14 | 9 | 14 | 6 | 14 |
| **(f)** | 1 | 7 | 2 | 2 | 4 | 2 | 13 | 6 |
| **(g)** | 0 | 1.07 | 0 | 5 | 10 | 6 | 0 | 11 |
| **(h)** | 12.80 | 16.80 | 9.20 | 10 | 17.70 | 7.20 | 11.10 | 9.20 |

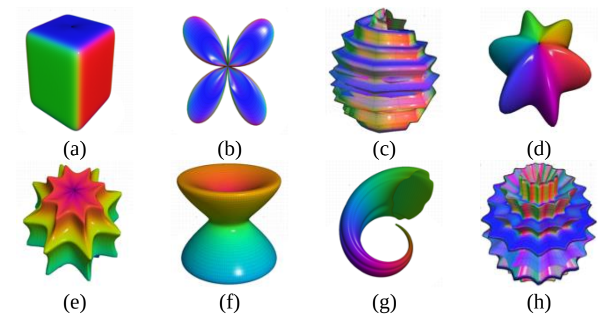


Figure 1. Examples of Supershapes.

### B. PointNet++ Architecture

PointNet++ [8] is an advanced architecture designed for deep learning on point clouds, enabling efficient analysis and processing of 3D data. It improves upon the original PointNet model by introducing hierarchical feature learning, allowing the capture of both local and global information. The architecture, e.g., Fig. 2, consists of four key components: input transformation, PointNet Set Abstraction (SA), PointNet Feature Propagation (FP), and global feature extraction.

In the first step, the input transformation network is applied to align the points in the point cloud, ensuring invariance to input permutations. This network learns a linear transformation for each point, normalizing the input across different point clouds. This process enables the model to handle variations in point order or orientation.

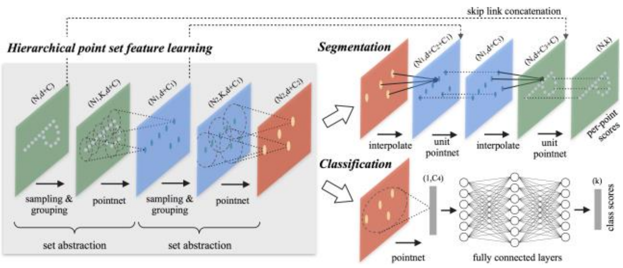


Figure 2. PointNet++ architecture [8].

The SA module is the heart of PointNet++. It performs hierarchical feature learning by iteratively down-sampling the point cloud and extracting local features. It starts by sampling a set of representative points called centroids.

These centroids act as seed points for defining local regions. The points within the vicinity of each centroid are then grouped together using a spatial search algorithm. Within each local region, a PointNet module is applied to capture local features, utilizing shared Multi-layer Perceptron (MLPs) to process each point individually.

The FP module works in conjunction with the SA module and is responsible for propagating the learned local features to higher-resolution levels. It begins by interpolating the local features to align with the centroids in the previous level, ensuring spatial consistency. The interpolated features are then further processed by a PointNet module to refine the propagated information.

To capture global information, a global feature extraction module is applied. It aggregates the features from all points in the point cloud, typically using max pooling or a similar operation to obtain a fixed-length global feature vector. This global feature vector is then passed through fully connected layers for tasks such as classification or segmentation.

Overall, PointNet++ leverages hierarchical feature learning to capture both local and global information from point clouds. By incorporating SA and FP modules, it enables efficient processing of 3D data and has shown superior performance in tasks like object recognition, semantic segmentation, and point cloud completion. Its ability to handle unstructured point cloud data makes it a powerful tool in various applications related to 3D analysis and understanding.

### C. Multitask Learning

The multi-task learning is a machine learning paradigm where a model is trained to perform multiple related tasks simultaneously. Instead of training separate models for each task, multi-task learning aims to leverage the shared information between tasks to improve overall performance. In multi-task learning, e.g., Fig. 3, the model is designed to have shared hidden layers that capture common patterns and features across different tasks. These shared layers allow the model to learn representations that are beneficial for multiple tasks. Additionally, each task may have its own task-specific layers that specialize in capturing task-specific patterns.

The specific number of hidden layers and their sizes can be determined based on the complexity of the tasks and the available data. It is common to experiment with different architectures, layer sizes, and activation functions to find the optimal configuration for the multi-task regressor. Regularization techniques, such as dropout or batch normalization, can also be applied to prevent overfitting and improve generalization.

The benefits of multi-task feature learning include improved generalization, better resource utilization, and the ability to learn from limited labeled data. By jointly learning multiple tasks, the model can learn more robust representations and better handle variations and uncertainties in the data.

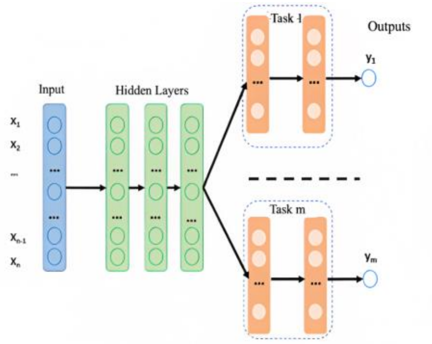


Figure 3. Multi-task regression.

### D. The Proposed Approach

This paper presents a novel approach that uses deep learning to retrieve the individual parameters of Supershape models from point clouds. Specifically, we tackle the task of recovering Supershape model parameters by formulating it as a prediction problem. The objective is to estimate the parameters of a given surface model, leveraging the power of deep learning techniques:

$$\hat{y} = [m, M, n_1, n_2, n_3, N_1, N_2, N_3] \in R^{1\times 10} \quad (2)$$

We aim to achieve an estimation $\hat{y} = f(x, W)$, where $f$ represents the Reg-PointNet++ model with weights $W$, for the input $x$ (the 3D object), as close as possible to $y$. To accomplish this, the network needs to be trained on a large number of 3D objects in point cloud format. The training of Reg-PointNet++ utilizes the stochastic gradient descent optimization algorithm ADAM minibatch (an improved version of SGD mini-batch) that efficiently minimizes the loss function on the available training dataset.

The objective is to optimize the model's weights $W$ to minimize the discrepancy between the predicted output $\hat{y}$ and the true output y for each input $x$. The loss function measures the dissimilarity between the predicted and true values, capturing the model's performance. By employing the ADAM minibatch algorithm, the optimization process efficiently updates the weights based on gradients computed from subsets of the training dataset, known as mini-batches.

The following Fig. 4 illustrates the architecture of our Reg-Pointnet++ network:

Reg-Pointnet++ is a network with PointNet++ as the base architecture, extended with a multitask regressor to estimate the parameters of a Supershape. The input to the network would be a 3D point cloud representing the object we aim to reconstruct.

The PointNet++ network would be used to extract meaningful features from the point cloud. This may involve transformation and hierarchical grouping operations to capture the spatial relationships between the points.

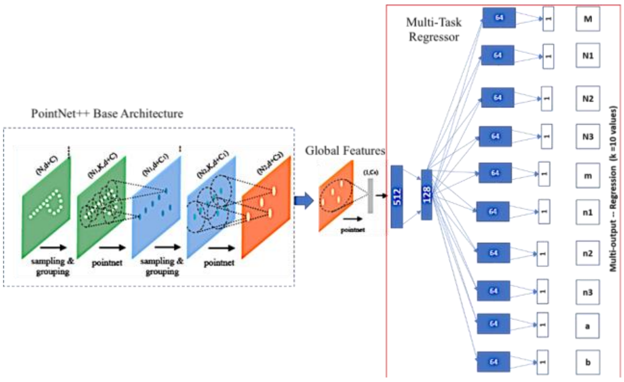


Figure 4. Architecture of Reg-PointNet++.

The features extracted by PointNet++ would then be used to represent the object in a compact and informative manner.

The extension with Multitask Regressor will perform the prediction of the Supershape parameters. It takes the features extracted by PointNet++ as input.

This additional branch consists of three hidden layers that have respective sizes of 512, 128, and 64, and are designed to extract and learn complex features from the input data. In conclusion, the network incorporates a multiple output layer that utilizes a straightforward linear regression to compute the surface parameters. This allows the network to achieve precise predictions of the surface parameters for the given input object.

The multitask regressor will be trained by minimizing an appropriate loss function, which compares the network's predictions to the ground truth values of the Supershape parameters.

To assess the performance of our Regressor model, we employ the L2 norm-based distance metric between the predicted Supershape parameters $\hat{y}$ and the pre-defined parameters $y$. The objective function is defined as:

$$L_2\,(y, \hat{y}) = \|y - \hat{y}\|^2 \tag{3}$$

By minimizing the $L_2$ distance between the predicted and target Supershape parameters during the training process, our goal is to ensure that our model acquires the ability to make accurate predictions and reduce the error in our estimations. This approach allows us to quantitatively evaluate the accuracy of our model and ascertain its generalization capabilities when faced with new, unseen data.

The $L_2$ norm computes the square root of the sum of squared differences between the corresponding elements of $y$ and $\hat{y}$. This distance metric provides a measure of the overall dissimilarity between the predicted and target values of the Supershape parameters. The objective function guides the learning process, driving the model to minimize the discrepancy between its predictions and the ground truth.

To ensure accurate predictions of the Supershape parameters for a given 3D object $x$, our Reg-PointNet++ model undergoes training on a substantial dataset comprising 3D objects represented as point clouds. When presented with a 3D object $x$ as input, the model utilizes its learned weights $W$ to generate a prediction of the Supershape parameters $\hat{y}$.

During training, the model is exposed to a sequence of input-output pairs $(x, y)$, where $x$ represents a 3D object in the form of a point cloud, and $y$ corresponds to the pre-defined set of Supershape parameters. By processing the input $x$, the model predicts the Supershape parameters $\hat{y}$. The subsequent step involves calculating the loss between the predicted parameters $\hat{y}$ and the ground truth parameters $y$, utilizing the $L_2$ distance metric. This metric quantifies the dissimilarity between the predicted and target values of the Supershape parameters.

To update the model's weights $W$ and refine its predictive capabilities, the ADAM minibatch algorithm is employed. This optimization algorithm iteratively adjusts the weights based on gradients computed from mini-batches, which are subsets of the training dataset. By minimizing the loss function, the model incrementally improves its predictions, striving to accurately estimate the Supershape parameters.

The training process involves repeating the presentation of input-output pairs, the calculation of loss, and the subsequent weight updates for numerous iterations. This iterative procedure continues until the model converges to an optimal set of weights that enable it to produce accurate predictions of the Supershape parameters for new, unseen 3D objects.

Through this training methodology, our Reg-PointNet++ model acquires the ability to generalize and make accurate predictions of the Supershape parameters for various 3D objects. By leveraging the power of the ADAM minibatch algorithm and the informative point cloud representations, the model optimizes its performance, enabling it to handle diverse and complex 3D shapes effectively.

The entire network would be trained iteratively by adjusting the weights of the different branches based on the total loss calculated from the network's predictions for all tasks. The goal is to optimize the network to accurately estimate the parameters of the Supershape from the input point cloud. This combined architecture of PointNet++ and a multitask regressor would leverage the 3D representation capabilities of PointNet++ while performing precise parameter estimations of the Supershape.

## IV. Experimental Results

### *A. The Dataset*

In order to train and evaluate our Reg-Pointnet++, we create a synthetic DataSet of 3D shapes using a specific algorithm. This algorithm takes into account three steps.

The first step is the generation of 3D objects. We generate 3D shapes by randomly varying the Supershape parameters within predefined bounds. This allows us to create a diverse set of shapes for training and evaluation.
The second step is the resampling and dataSet organization. To ensure consistency in the input data, we resample each point cloud by reducing the number of points to 512 per object. After resampling, we organize the point clouds into a DataSet, with 70% of the objects allocated for training

(9300 objects) and the remaining objects reserved for testing (2300 objects).

The last step is the data representation. The data used for training and evaluation is represented in the form of coordinates of the points in the ply format. Each object is described by an ASCII file containing the (x, y, z) coordinates of all the points.

By following this process, we create a synthetic DataSet that enables us to effectively train and evaluate our Reg-Pointnet++.

Here is an example of generated objects with 512 points under different views (Fig. 5):

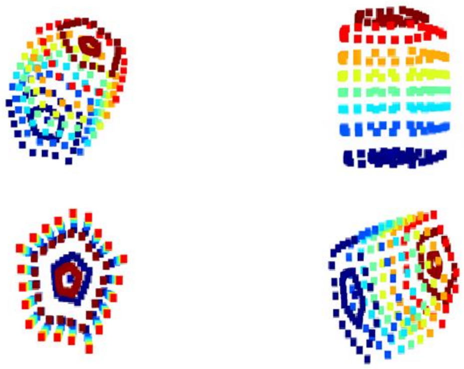

Figure 5. DataSet object in point cloud under different views.

## B. Performance Metrics

To assess the effectiveness of our CNN model in predicting the Supershape parameters, we utilize the Mean Squared Error (MSE) as our evaluation metric for each parameter $p \in \{m, n_1, n_2, n_3, M, N_1, N_2, N_3, a, b\}$ during experimentation. The MSE is calculated as follows:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \quad (4)$$

The MSE is a widely adopted metric for evaluating the accuracy of regression models, including our CNN based multitask regressor. It quantifies the average squared difference between the predicted values ($\hat{y}_1$) and the true values ($y_i$) of the Supershape parameters across a given set of test data.

For each parameter $p$, the MSE is computed by averaging the squared differences between the predicted and true values over all test samples. A lower MSE value indicates a higher level of accuracy in predicting the Supershape parameters. This metric is commonly employed in machine learning to evaluate the performance of regression models.

By employing MSE as our evaluation metric for the Reg-Pointnet++, we can measure the accuracy of the model in predicting the Supershape parameters and compare its performance against other regression models or different iterations of our own model. This facilitates iterative refinement of the model's architecture and hyperparameters to achieve enhanced performance in predicting the Supershape parameters.

To optimize the performance of our Reg-Pointnet++ model, we employ the ADAM, which is known for its effectiveness in minimizing the loss function and converging to an optimal set of weights. It combines the advantages of the AdaGrad [12] and RMSProp [13] techniques by utilizing adaptive learning rates for each parameter, leading to faster convergence and better generalization. During training, ADAM updates the model weights using both first-order gradients (momentum) and second-order gradients (RMSProp), allowing for adaptive adjustment of the learning rate based on the model's current state. By using ADAM with minibatch training, we efficiently update the weights of our Reg-Pointnet++ model using gradients calculated on small subsets of the training data, enabling effective handling of large datasets and accelerated convergence. Overall, the utilization of the ADAM optimization algorithm in conjunction with our CNN-based Multi-Output Regressor model plays a crucial role in achieving accurate predictions of the Supershape parameters and enhancing the overall performance of our model.

## C. Training and Testing

The results over 400 iterations of the evolution of the global precision (Accuracy for the training and the test) are represented by Figs. 6–8.

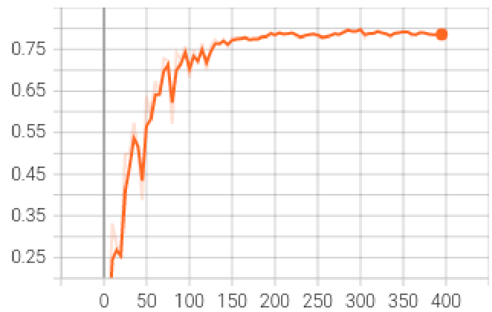


Figure 6. Evolution of overall precision (Accuracy, Training).

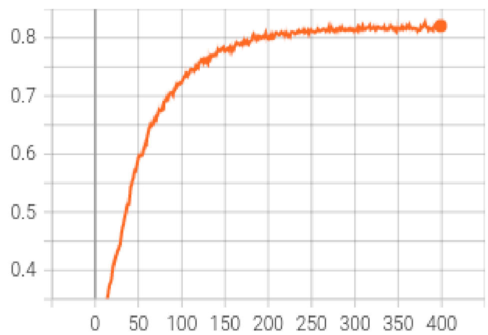


Figure 7. Evolution of overall accuracy (Accuracy, Test).

## D. Results on Test Objects

Input: object in cloud points

Output: Supershapes parameters (10 base values)

The following figure represents the results obtained for four different objects provided as input under two different views. Objects in red represent input objects, and those in green represent result objects.

There are no obvious imperfections. All prediction reconstructions perfectly match the actual shapes.

The results shown in Table II support the visual results. It can be seen that the predicted parameters do not represent a great difference with the parameters of the input form. This confirms the quality of the obtained reconstruction.

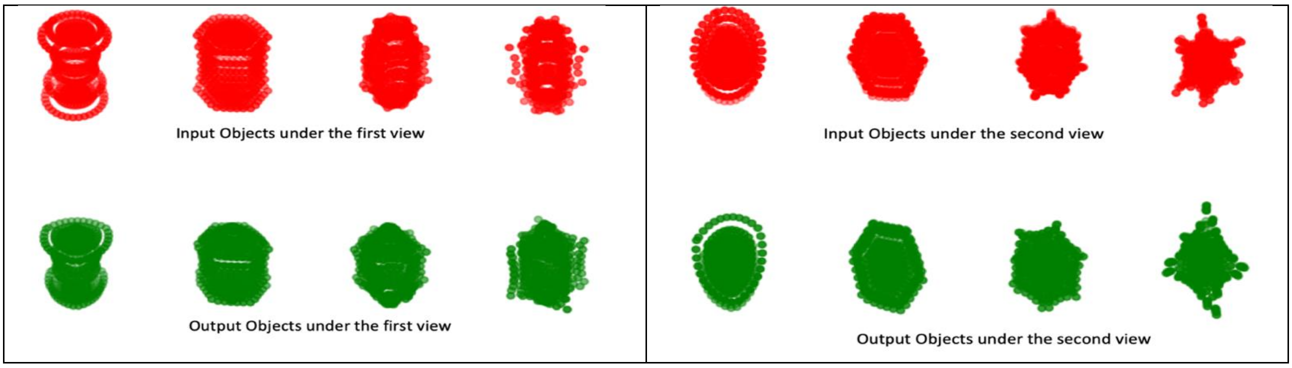


Figure 8. Reconstruction of 4 different objects provided as input to Reg-PointNet++ under two different views.

TABLE II. TEST RESULTS WITH MULTI-TASKING ARCHITECTURE ON 4 DIFFERENT OBJECTS

| ***Supershapes param.*** | | m | $n_1$ | $n_2$ | $n_3$ | M | $N_1$ | $N_2$ | $N_3$ | a | b |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Object 1 | *Input* | 1 | 70 | 31 | 27 | 5 | 19 | 83 | 30 | 1 | 1 |
| | *Output* | 2.9972 | 71.0867 | 29.8220 | 26.6507 | 5.4638 | 23.4425 | 81.1402 | 28.0925 | 0.9954 | 0.9954 |
| Object 2 | *Input* | 6 | 99 | 54 | 33 | 5 | 81 | 95 | 78 | 1 | 1 |
| | *Output* | 5.7735 | 97.3337 | 54.2550 | 33.9757 | 5.8488 | 82.8021 | 91.6819 | 74.9392 | 1.0009 | 1.0009 |
| Object 3 | *Input* | 7 | 109 | 87 | 97 | 9 | 90 | 74 | 68 | 1 | 1 |
| | *Output* | 6.4273 | 106.4220 | 85.8470 | 97.9915 | 6.8821 | 86.3549 | 74.7567 | 66.7051 | 1.0041 | 1.0041 |
| Object 4 | *Input* | 6 | 19 | 70 | 28 | 9 | 55 | 32 | 83 | 1 | 1 |
| | *Output* | 4.4713 | 19.6545 | 67.1488 | 29.7228 | 6.1809 | 52.2985 | 32.8960 | 82.5247 | 0.9962 | 0.9962 |

*E. Evalution and Discussion*

The Table III summarizes the accuracy values of two applications of the regressors, in our previous works: CNN-based multi-output regressor [6] and CNN-based multi-task regressor, with the proposed approach Reg-Pointnet++. It can be observed that Reg-PointNet++ significantly improves accuracy with fewer iterations. However, the architecture of the latter is quite heavy and requires much more computation compared to the others during training. It is worth noting that all experiments were conducted on a single CPU (Intel Core i7, 8 cores, 16GB RAM) without utilizing CUDA (GPU) acceleration. The execution (training, validation, and testing) was halted after approximately three days. This resulted in 3000 iterations for CNN-based multi-output regressor, 2000 iterations for the Multi-Task Regressor, and only 400 iterations for Reg-PointNet++ with superior performance. This indicates that with more iterations, the Reg-PointNet++ version holds the promise of further improvements in performance but demands additional resources (such as NVIDIA GPUs with CUDA support).

Two limitations can be spotted in our approach. The first one, is the network's dependence on uniformly sampled point clouds. The network requires an even distribution of points within these regions to operate effectively. However, in real-world scenarios, point clouds may exhibit non-uniform sampling density, with varying point densities in different regions. Addressing this limitation and developing techniques to handle non-uniformly sampled point clouds is an active area of research in the field of 3D deep learning. The second one, is the creation of the dataset. Since Supershapes are recent shapes, there is no dataset available for use. Creating our own dataset can be very time consuming.

TABLE III. COMPARATIVE TABLE

| **Approach** | **Accuracy** | |
|---|---|---|
| | **(%) Train** | **(%) Test** |
| CNN-based Multi-Output Regressor (3000 iterations) | 76.8 | 74.0 |
| CNN-based Multi-Task Regressor (2000 iterations) | 80.0 | 72.8 |
| Reg-PointNet++ (400 iterations) | **82.5** | **79.2** |

However, given the encouraging results, we can extend our research to real-world objects, specifically focusing on composite objects. It is extremely challenging to model any shape of objects as a closed Supershape surface. To understand the problem, we examine industrial 3D objects that are composed of simple primitive objects such as spheres, cylinders, cones, tori, etc. Constructive Solid Geometry (CSG), e.g., Fig. 9 which is a branch of solid modeling in computer graphics, can be used to address this challenge [14].

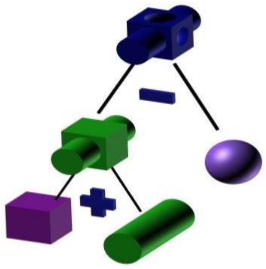

Figure 9. Example of an object represented by a CSG tree.

The Reg-Pointnet++ enables easy retrieval of Supershape forms for the primitives used in industrial object compositions. An industrial object, provided as a point cloud (e.g., obtained from scanning), can have its CSG determined, thereby identifying its primitives. These primitives can then be represented easily using the Supershape forms predicted by our model.

## V. Conclusion and Future Work

In this paper, we have introduced a novel approach to recover Supershape from points cloud using Deep Learning. We have demonstrated that extending the PointNet++ model with regressors is capable of achieving satisfactory recovery accuracy (80%) in parameter prediction, while also reducing the computational time. The visual results are highly satisfying and optimal, demonstrating the effectiveness of our proposed approach. To the best of our knowledge, this work is the first to introduce a 3D reconstruction model for Supershapes based on regressor models, highlighting the significant potential of this research direction.

While our approach has shown promising results, there are several avenues for future research and improvement. Among the potential directions that we can explore, the improvement of the parameter prediction accuracy for Supershapes. This can involve studying advanced neural network architectures, or investigating novel loss functions. We can also investigate methods to handle noisy or incomplete point cloud data which is crucial for real-world applications. The proposed approach can be validated on larger and more diverse datasets, this would provide a comprehensive assessment of its performance and generalization capabilities. By constructing domain-specific datasets for Supershape reconstruction, we would facilitate fair comparisons with existing methods and promote reproducibility.

By addressing these future research directions, we can continue to advance the field of 3D reconstruction for Supershapes using deep learning, paving the way for practical applications in computer graphics, virtual reality, and shape analysis.

## Conflict of Interest

The authors declare no conflict of interest.

## Author Contributions

Hassnae Remmach and Mohamed Sagdal conducted the research on the subject. They developed the idea with Aziz El Fazziki, who presented suggestions to improve the quality of the work rendered. Hassnae Remmach and Mohamed Sagdal worked on the realization of the approach and the experimental results. Aziz El Fazziki also presented his support for this part. Raja Mouachi contributed in the literature review and writing of the paper, as well as the analysis of the data to make the necessary adjustments. All the authors mentioned areas of application of the proposed approach as future work. The article was mainly written by Hassnae Remmach, and the final version was revised and approved by all authors.

## References


[1] J. A. C. Moreano and N. B. L. S. Palomino, "Global facial recognition using Gabor wavelet, support vector machines and 3D face models," *Journal of Advances in Information Technology*, vol. 11, no. 3, pp. 143–148, August 2020. doi: 10.12720/jait.11.3.143-148

[2] M. Tanaka and T. Kaneeda, "A method of reconstructing 3D models from sketches by extracting features," vol. 5, no. 3, pp. 74–78, August, 2014. doi:10.4304/jait.5.3.74-78

[3] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proc. the IEEE*, 1998, pp. 2278–2324.

[4] J. L. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, no. 2, pp.179–211, 1990.

[5] H. Remmach, M. Sadgal, and A. EL Fazziki, "Swarm optimization for tridimensional point cloud reconstruction using Supershapes," *International Journal of Advanced Computer Science and Engineering*, 2020.

[6] H. Remmach, R. Mouachi, and M. Sadgal, "CNN-based multi-output regressor: A deep learning approach for three-dimensional approach for three-dimensional reconstruction from 3D points cloud," in *Proc. International Workshop on Smart Systems and Artificial Intelligence*, 2023.

[7] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. CVPR*, 2017.

[8] C. R. Qi *et al.*, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. the 31st Conference on Neural Information Processing Systems*, *NIPS 2017*, Long Beach, CA, USA, December 4–9, 2017.

[9] Y. Wang *et al.*, "Dynamic graph CNN for learning on point clouds," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 5, pp. 1–12, 2019.

[10] Y. Li *et al.*, "PointCNN: Convolution on x-transformed points," in *Proc. the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[11] H. Thomas *et al.*, "KPConv: Flexible and deformable convolution for point clouds," in *Proc. the IEEE International Conference on Computer Vision*, 2019.

[12] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," arXiv preprint, arXiv:1412.6980, 2014.

[13] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. Jul., pp. 2121–2159, 2011.

[14] T. Tieleman and G. Hinton, "Lecture 6.5—RMSProp: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural Networks for Machine Learning*, vol. 4, no. 2, pp. 26–31, 2012.

[15] Q. Wu, K. Xu, and J. Wang, "Constructing 3D CSG models from 3D raw point clouds," *Computer Science, Comput. Graph. Forum*, 2018.