AtlasNet: 3D Shape Reconstruction from RGB Images and Point Clouds

Chang Luo
chang.luo@tum.de

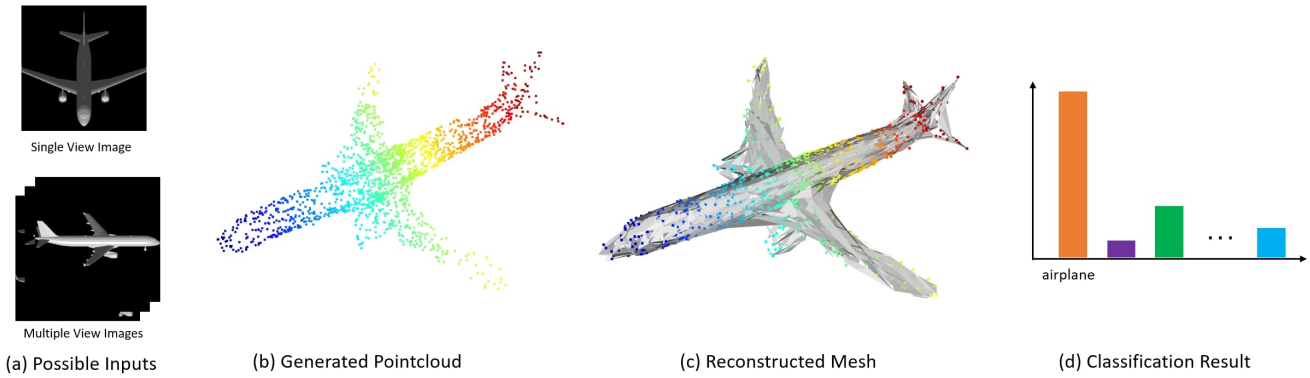

Figure 1. Using one of the possible inputs (a), we deform a uniformly sampled point set into a point cloud (b), recovering the edge connections we reconstruct a mesh (c). The input image is also used to get classification results (d).

Abstract

Even though there is significant progress in learning 3D shape representations it is still an open challenge. One of the biggest hurdles in learning high resolution 3D shapes is the parameterization of the surface which is required by many applications such as texture mapping or re-meshing. By using AtlasNet as a basis we improve the learning process via using an improved encoder and introducing a shape classification functionality. We also provide our results after training the program with another dataset.

1. Introduction

There are different ways to represent high resolution 3D shapes. Point clouds, voxels or meshes can be used but each technique has its advantages and disadvantages. In this paper we are focusing on meshes. A mesh consists of a set of vertices, edges and faces and usually contains additional information such as 2D embedding to a plane. This 2D information is called UV parameterization and it is useful for illustrating additional surface details, normals or textures.

AtlasNet [3] is an approach to generate 3D meshes from an input of either a 2D image or a 3D point cloud via learning the surface representation directly. AtlasNet [3] approximates the target surface by mapping a set of squares to the surface of the 3D shape and uses these squares to learn topographic features of the surface. With an input of a sampled point on a square, a 3D point on a surface can be generated and by repeating this approach the 3D shape can be represented as an union of these generated surfaces.

In our approach we used another dataset to train AtlasNet [3] which is originally trained using the second version of the ShapeNetCore [1] dataset and afterwards extended it with shape classification and an improved encoder.

Shape classification means recognizing, understanding and grouping shapes into preset categories. By using a precategorized training dataset a machine learning program can classify future inputs in the respective and relevant categories. In our example we trained our program with a categorized set and after training expected it to recognize which of the learned categories a new input belongs to.

We show that by extending the encoder to accept more images per object more features can be extracted thus the program can learn faster and therefore requires less epochs to train.

2. Related work

Surface parameterization is an important aspect of geometry processing. It is necessary in re-meshing, texture mapping and shape correlation [5]. There are various proposed techniques to solve the parameterization problem such as

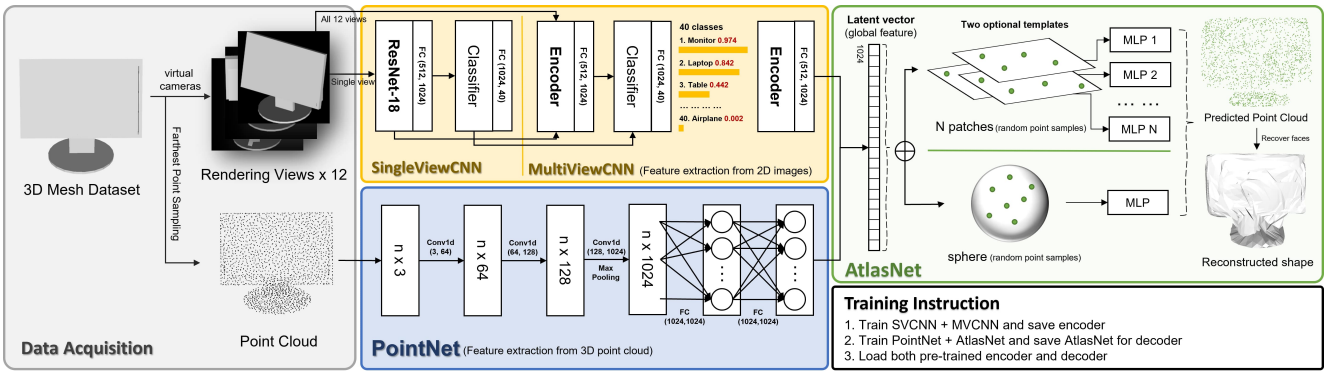

Figure 2. Overall Structure of our AtlasNet implementation.

local polar coordinates or global seamless maps [6]. For global surface parameterization and meshing, we followed the theory given in AtlasNet [3].

A series of charts, which can cover a 2D manifold, can be called an atlas of 2D manifolds. Each chart corresponds to a region of the 2D manifold. With this definition in mind, it is possible to summarize our task as: Find a parameterized function $\psi_\theta(x)$ that maps the uniform area $(0, 1)^2$ around a point p into an approximation of a 2D manifold. In turn, the function $\psi_\theta(x)$ needs to satisfy two properties: (1) it is able to generate two-dimensional manifolds; and (2) it is able to approximate the two-dimensional manifold very well. It is also shown in the paper [3] that networks built from MLPs and ReLUs have almost these properties, and this is used for model building.

3. Dataset

In the original paper [3] a subset of the ShapeNetCore dataset, split into 80% training data and 20% test data, is used. This subset contains 13 object categories and is often utilized as a computer vision 3D benchmark dataset [1].

For our approach we trained the network with ModelNet40 [10]. This dataset contains 12,311 pre-aligned shapes from 40 categories, which are also split into 80% training data and 20% test data. The motivation on using another dataset was to see the networks' performance on other data from new categories and to be able to compare this performance with the original one.

3.1. Data Pre-Processing

ModelNet40 [10] provides 3D models in .off format. AtlasNet takes point clouds and/or images as input. Thus, some pre-processing of the data was necessary, in order to train the network with it.

First we set up a system with 12 virtual cameras rotating around the object every 30 degrees resulting in (227×227) RGB images rendered from the respective camera views.

Afterwards we used the farthest point sampling (FPS) algorithm to generate point clouds from the 3D models of ModelNet40. FPS is a greedy technique used to sample a point cloud efficiently [7].

3.2. Network Architecture

In this section, we will introduce our network architecture, namely the Autoencoder structure. The encoder which extracts the feature from a single view image or a point cloud to form a latent representation, while a decoder deforms random sampled points and learns to generate the point cloud given the shape encoding. Unlike the original paper, we changed the encoder part from the naive encoder to a pre-trained encoder on a classification network, for the sake of importing a classification ability. The performance of the different encoders is further discussed below.

PointNet Encoder: For point cloud inputs, we followed the original implementation to utilize the PointNet [8] network for extracting features from a 3D point cloud. The PointNet network takes N (in our case, 2000) points as input, the point features are aggregated by max pooling. We also included the final classifier which reads the latent vector to perform a point cloud based classification.

ResNet-18 Encoder: For image input, the original paper considers ResNet-18 [4] as the feature extractor. Similarly, the ResNet-18 takes an image with spatial size $(224 \times 224 \times 3)$, and outputs the score for 1000 classes, the original paper changed this number to 1024 and took this output as the encoding for the input image.

SVCNN and MVCNN Encoder: MVCNN [9] is a classification network which proposed a new representation format for 3D objects. The mesh is rendered from multiple views to get the 2D image from these view angles. The view renderings then are further passed into Single-View CNNs (SVCNN) to extract view based features, which are then max-pooled for all views and passed into Multiple-View CNN (MVCNN) to obtain a compact encoding of the shape. We changed the CNN feature extractor from VGG-11 [9] to

ResNet-18 to avoid network structure changes.

AtlasNet Decoder: The task of the decoder is, when a latent feature descriptor γ is given, to take the advantage of the low dimensional feature and generate the point cloud from this feature. As the original paper already proved, the combination of multiple layer perceptrons and ReLUs can locally generate shapes by learning a 2D to 3D mapping function. In practice, the shape is considered to be divided in N parts, and each part represents a learnable parameterzation θ_N of the generated shape that corresponds to the MLP which is learned. A point set with the same point number as the input point cloud is uniformly sampled from a unit square area $[0,1]^2$, denoted as P_s. The sampled points are concatenated with the given encoding γ and passed as the input $(p_s;\gamma)$ for MLP. During training, the MLP learns the weights that deform P_s into the generated point set P_d.

$$p_d = \theta_i(p_s;\gamma) \qquad (p_d \in P_d,\, p_s \in P_s) \tag{1}$$

In order to measure the distance between the generated point set P_d and the ground truth input P_i, the Chamfer distance is used.

$$L(P_d,P_i) = \sum_{p_d \in P_d} \min_{p_i \in P_i} ||p_d - p_i||_2^2 + \sum_{p_i \in P_i} \min_{p_d \in P_d} ||p_d - p_i||_2^2 \tag{2}$$

The Chamfer distance sums up the squared distance between nearest neighbor correspondences of two point clouds, hence it can be used to measure the similarity of two point clouds. We minimized the Chamfer distance between the deformed point set P_d and the input set P_i to train our AtlasNet decoder.

3.3. Training Pipeline

Following the instruction from the original paper [3], the combination of PointNet and AtlasNet was trained at first, and the decoder was saved for later training. Subsequently, we trained the MVCNN network, and saved the weights for both SVCNN and MVCNN. At the final stage, according to the different types of test input, the corresponding encoder (PointNet, SVCNN or MVCNN) is loaded.The pre-trained decoder is also loaded for fine tuning. In this way we were able to get an acceptable model in only a few epochs.

3.4. Mesh Generation

The deformed point cloud can be obtained effortlessly, but the goal was to reconstruct the mesh representation. In order to recover the connection information of vertices, we used the same method as AtlasNet [3], but implemented some changes due to the different point number for point clouds. There are two types of templates that can be selected while testing, square or sphere. The square template is nearly identical to a 2D unit square, but we transferred it into 3D space, and set the value of the z-axis to zero.

Classifier	Accuracy
PointNet	88.89%
SVCNN	90.63%
MVCNN	93.67%

Table 1. Classifier accuracy between different encoders

For the sphere template, the method involving the process of generating the icosphere was discarded, since it can only generate a specific number of vertices on the sphere surface. We changed the sampling way to Fibonacci sphere [2], aiming to get a specific number of samples on a sphere. After the vertices are prepared, we computed the convex hull of all vertices, to address the faces information. As long as the vertices are deformed while learning, the faces data will never change, hence the mesh structure can be easily recovered.

4. Results

4.1. Comparison

As mentioned in the previous section, we have experimented with both ResNet-18 as image encoder, same as the original paper [3] as well as with a pre-trained MVCNN [9] which uses multiple ResNet-18 for feature extraction. During our experiments, we observed that both the training loss and the validation loss converge a lot faster with MVCNN when compared to using a single ResNet-18 as the encoder. In addition to that, the F-Score also converges faster and has a slightly higher score compared to the ResNet-18.

The reason for that is, unlike using ResNet-18 as the encoder, by using MVCNN, we are able to utilize the multi-view information, which in turn increases the overall performance of the network.

4.2. Classification

Another benefit of using MVCNN [9] instead of ResNet-18 is the ability to do accurate classification. By using MVCNN, it is possible to use the multi-view information for an object, which in turn increases the classification accuracy. As seen from Table 1, we managed to acquire 93% accuracy on the classification task over 2468 models, which is the validation set, with 40 different classes by using MVCNN. The PointNet [8] performed worse out of these three for the classification task, however, if we consider that it only used point clouds as input, it is still an impressive result.

4.3. Reconstruction Task

For the single-view reconstruction task, we have used two different models, a model with SVCNN as the encoder, and a model with MVCNN [9] as the encoder. From our

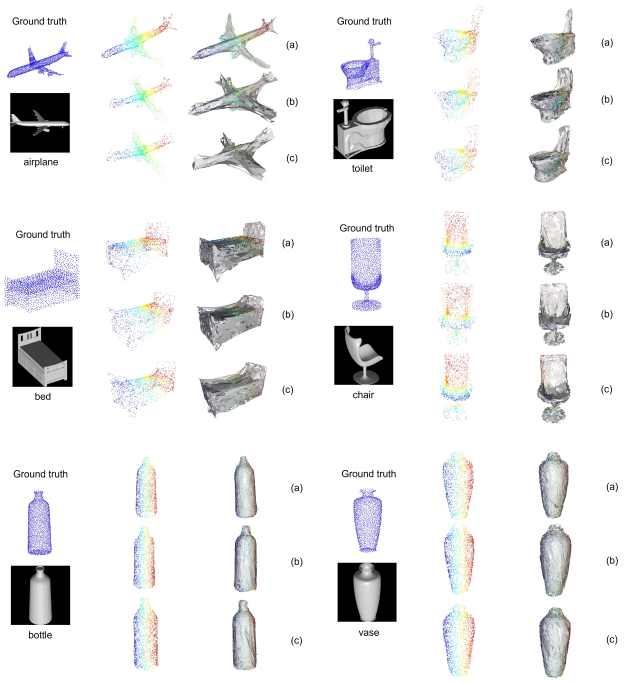

Figure 3. Reconstruction result for three networks (a) MVCNN - Multiple views image input, (b) SVCNN - Single view image input, (c) MVCNN - Single view image input, (d) Ground truth point cloud

observations, as well as the F-scores of both models, we have determined that the model with MVCNN outperforms the other model. The main reason for that is, MVCNN was pre-trained with multiple views but fine-tuned with single view images similar to ResNet-18 [4]. As such, it was able to learn features better than the ResNet-18.

For multi-view reconstruction, we have used MVCNN with multiple images as input. This model performed best out of the three (single-view reconstruction with SVCNN, single-view reconstruction with MVCNN, multi-view reconstruction with MVCNN). This was an expected result since the model was able to utilize the multi-view information during the fine-tuning as well as the testing phase. Due to that, the provided results will be from the better performing model.

The network provided satisfactory results for this task, however, the results on the original paper are better due to the difference in the training data, as ModelNet40 [10] is around 5 times smaller than ShapeNet [1]. The network performed quite well on simple shapes, such as bottle and cup, as well as classes with a lot of training data. For instance, in Figure 3 the reconstructed chair is quite good given that the view of the image used for reconstruction is quite bad.

The main limitation for this task were the classes with scarce training data. This problem was caused by the structure of ModelNet40 [10] as it is not a balanced dataset. Due

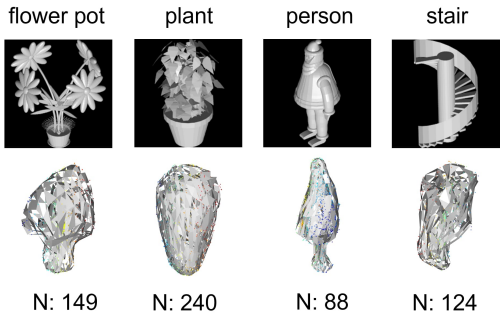

Figure 4. Bad reconstruction due to the insufficient training data

to that, some of the classes, such as person or stairs do not contain sufficient data for good results. The network also performed poorly on complex shapes such as the flower part of the flower pot (see Figure 4. Even though it was able to construct the overall shape correctly in those cases, it was unable to construct the details properly. A solution to this problem would be adding more training data with complex shapes, and possibly increasing the depth of the decoder while doing so.

We observed that given sufficient training data, a class with mediocre complexity was also reconstructed quite successfully. For instance, on Figure 3, the toilet was reconstructed quite successfully even though it's shape is not as simple as a bottle. This further proves that if sufficient training data for complex classes is provided, it would also possible to reconstruct them successfully.

5. Conclusion

In this project, we have trained the AtlasNet [3] with the ModelNet40 [10] dataset. In addition to that, we have introduced MVCNN [9] as an encoder to the pipeline when we have multiple views as input, which improved the performance. We have also observed that the network did pretty well on simple shapes as well as classes with sufficient training data, however, it performs poorly on complex shapes and some classes with low amount of training data.

There are several possible improvements to the overall project. The biggest improvement would be training it on a bigger, more balanced dataset. It is also possible to integrate shape part segmentation or shape recognition to the network. After these improvements, if the results are satisfactory, an automated annotation with a human only doing verification for the new data can be set up. This would provide a bigger dataset, which should in turn improve the performance of the network.

References

[1] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Mano-

lis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository, 2015. cite arxiv:1512.03012. 1, 2, 4

[2] Álvaro González. Measurement of areas on a sphere using fibonacci and latitude–longitude lattices. *Mathematical Geosciences*, 42(1):49–64, nov 2009. 3

[3] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan C. Russell, and Mathieu Aubry. Atlasnet: A papier-mch approach to learning 3d surface generation, 2018. 1, 2, 3, 4

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 2, 4

[5] Kai Hormann, Konrad Polthier, and Alia Sheffer. Mesh parameterization: Theory and practice. In *ACM SIGGRAPH ASIA 2008 Courses*, SIGGRAPH Asia '08, New York, NY, USA, 2008. Association for Computing Machinery. 1

[6] Jonathan Masci, D. Boscaini, Michael M. Bronstein, and Pierre Vandergheynst. Geodesic convolutional neural networks on riemannian manifolds. *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)*, pages 832–840, 2015. 2

[7] Carsten Moenning and Neil Dodgson. Fast marching farthest point sampling for point clouds and implicit surfaces. 06 2003. 2

[8] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation, 2016. 2, 3

[9] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition, 2015. 2, 3, 4

[10] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1912–1920, 2015. 2, 4