Image2Mesh: A Learning Framework for Single Image 3D Reconstruction

Jhony K. Pontes[1(✉)], Chen Kong[2], Sridha Sridharan[1], Simon Lucey[2], Anders Eriksson[1], and Clinton Fookes[1]

[1] Queensland University of Technology, Brisbane, Australia
jhonykaesemodel@gmail.com
[2] Carnegie Mellon University, Pittsburgh, USA

Abstract. A challenge that remains open in 3D deep learning is how to efficiently represent 3D data to feed deep neural networks. Recent works have been relying on volumetric or point cloud representations, but such approaches suffer from a number of issues such as computational complexity, unordered data, and lack of finer geometry. An efficient way to represent a 3D shape is through a polygon mesh as it encodes both shape's geometric and topological information. However, the mesh's data structure is an irregular graph (*i.e.* collection of vertices connected by edges to form polygonal faces) and it is not straightforward to integrate it into learning frameworks since every mesh is likely to have a different structure. Here we address this drawback by efficiently converting an unstructured 3D mesh into a regular and compact shape parametrization that is ready for machine learning applications. We developed a simple and lightweight learning framework able to reconstruct high-quality 3D meshes from a single image by using a compact representation that encodes a mesh using free-form deformation and sparse linear combination in a small dictionary of 3D models. In contrast to prior work, we do not rely on classical silhouette and landmark registration techniques to perform the 3D reconstruction. We extensively evaluated our method on synthetic and real-world datasets and found that it can efficiently and compactly reconstruct 3D objects while preserving its important geometrical aspects.

1 Introduction

Most of us take for granted the ability to effortlessly perceive our surrounding world and its objects in three dimensions with a rich geometry. In general, we have good understanding of the 3D structure only by looking at a single 2D image of an object even when there are many possible shapes that could have produced the same image. We simply rely on assumptions and prior knowledge

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-030-20887-5_23) contains supplementary material, which is available to authorized users.

C. V. Jawahar et al. (Eds.): ACCV 2018, LNCS 11361, pp. 365–381, 2019.
https://doi.org/10.1007/978-3-030-20887-5_23

acquired throughout our lives for the inference. It is one of the fundamental goals of computer vision to give machines the ability to perceive its surroundings as we do, for the purpose of providing solutions to tasks such as self-driving cars, virtual and augmented reality, robotic surgery, to name a few.

A specific problem that is of particular interest towards achieving this ambition of human-like machine perception is that of recovering 3D information from a single image. This exceedingly difficult and highly ambiguous problem is typically addressed by incorporating prior knowledge about the scene such as shape or scene priors [1–10]. This body of work has provided a valuable foundation for this task and it has in particular indicated that the use of shape priors is highly beneficial. With the online availability of millions of 3D CAD models across different categories, the use of 3D shape prior becomes even more attractive and motivating. This is a realisation we propose to exploit in this work.

With the recent arrival of deep learning many interesting work have been done to tackle 3D inference from 2D imagery by exploring the abundance of 3D models available online [11–14]. Most of them rely on volumetric shape representation, an approach arguably motivated by the ease of which convolutions can be generalized from 2D to 3D. A significant drawback these methods have is that the computational and memory costs scale cubically with the resolution. Octrees [13] and point cloud [15] representations have been proposed to make the learning more efficient. However, despite of its improved performance, such representations still fail to capture fine-grained geometry as a dense 3D mesh representation would might capture.

The aim of this work is to exploit a compact mesh representation to better unlock fine-grained 3D geometry reconstruction from a single image. We propose a novel learning framework based on a graph that embeds 3D meshes in a low-dimensional space and still allow us to infer compelling reconstructions with high-level details. We draw inspiration by the works in [10,16], where a graph embedding to compactly model the intrinsic variation across classes of 3D models has been proposed. Essentially, any 3D mesh can be parametrized in terms of free-form deformation (FFD) [17] and sparse linear combination in a dictionary. FFD allow us to embed a 3D mesh in a grid space where deformations can be performed by repositioning a smaller number of control points. Although the FFD conserves the objects' topology, the sparse linear combination step allows it to be modified to better generalise the 3D reconstruction to unseen data.

Our method first classifies the latent space of an image to retrieve a coarse 3D model from a graph of 3D meshes as initialization. Then, the compact shape parameters are estimated from a feedforward neural network which maps the image features to the shape parameters space - FFD and sparse linear combination parameters. The dense 3D mesh model is then recovered by applying the estimated deformations to the 3D model selected. An overview of the proposed framework is illustrated in Fig. 1. In contrast to [10,16], our proposed method neither rely on landmark and silhouette registration techniques nor manually annotated 2D semantic landmarks which would limit its applicability.

The main contributions of this paper are:

- We propose a simple and lightweight learning framework to infer a high-quality 3D mesh model from a single image through a low-dimensional shape embedding space;
- To the best of our knowledge, the proposed method is the first to estimate visual compelling 3D mesh models with fine-grained geometry from a single image neither relying on classical landmark and silhouette registration techniques nor class-specific 2D landmarks;
- We demonstrate the performance of our method and its generalization capacity through extensive experiments on both synthetic and real data.

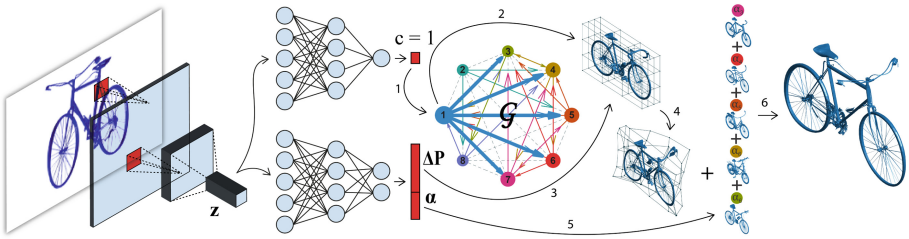

Fig. 1. Given a single image, our framework employs a convolutional autoencoder to extract the image's latent space $\mathbf{z}$ to be classified into an index c and regressed to a shape parametrization ($\mathbf{\Delta P}$, $\boldsymbol{\alpha}$). We use a graph embedding $\mathcal{G}$ to compactly represent 3D meshes. The estimated index c selects from $\mathcal{G}$ the closest 3D model to the image. The selected model is then deformed with the estimated parameters - FFD displacements $\mathbf{\Delta P}$ and sparse linear combination weights $\boldsymbol{\alpha}$. For instance, model 1 is selected (arrows 1 and 2), FFD is then applied (arrows 3 and 4), and finally the linear combination with the nodes 3, 4, 5, 6, and 7 (blue arrows on the graph that indicates the models in dense correspondence with node 1) are performed (arrow 5) to reconstruct the final 3D mesh (arrow 6). (Color figure online)

2 Related Work

Recent advances in neural networks and the online availability of 3D models (such as ShapeNet [18]) has sparked a considerable interest in methods using deep learning to solve tasks related to geometry and 3D reconstruction. Several papers have proposed a 3D volumetric representation [11,12,19–27] so they can feed deep neural networks applying 3D convolutions, pooling, and other techniques that have been successfully applied to 2D images for the learning process. Volumetric autoencoders [12,28,29] and generative adversarial networks (GANs) have been proposed [30–32] to learn of the probabilistic latent space of object shapes for object completion, classification and reconstruction. Despite of all the great work, volumetric representation has a great drawback. The memory and the computational costs grow cubically as the voxel resolution increases which limit such works to low-resolution 3D reconstructions.

Octree-based convolutional neural networks have been presented to manage these limitations [13,14,33,34]. It splits the voxel grid by recursively subdividing it into octants thus reducing the computational complexity of the 3D convolution. The computations are then focused on regions where most of the information about the object's geometry is contained - generally on its surface. Although it allows for higher resolution outputs, around 64^3 voxels, and a more efficient training, the volumetric models still lacks fine-scaled geometry. In trying to provide some answers to these shortcomings, a more efficient input representation for 3D geometry using point clouds have been recently entertained [15,35–39]. In [36] it was proposed a generative neural network to output a set of unordered 3D points used for the 3D reconstruction from single image and shape completion tasks. These architectures have been demonstrated for the generation of low-resolution 3D models and to scale them to higher resolution is yet to be investigated. Moreover, generating 3D surfaces for point clouds is a challenging problem, specially in the case of incomplete, noisy and sparse data [40].

3D shapes can be efficiently represented by polygon meshes to encode both geometrical and topological information [41]. However, the data structure of a mesh is an irregular graph (*i.e.* set of vertices connected by edges to form polygonal faces) and it is not straightforward to integrate it into learning frameworks since every mesh is likely to have a different structure. A deep residual network to generate 3D meshes has been proposed in [42]. The authors used a regular data structure to encode an irregular mesh by employing the geometry image representation. Geometry images however can only manage simple surfaces (*i.e.* genus-0 surface). FFD has also been explored for 3D mesh representation where one can represent an object by a set of polynomial basis and a fixed number of control points used to deform the mesh. A 3D shape editing tool has been presented in [43] and it uses a volumetric network to infer per-voxel deformation flows using FFD. Their method takes a volumetric representation of a 3D mesh as input and a deformation intention label (*e.g.* sporty car) to learn the FFD displacements to be applied to the original mesh. A novel graph embedding based on the local dense correspondences between 3D meshes has been proposed in [10,16]. The method is able to reconstruct the finer geometry of a single image based on a low-dimensional 3D mesh parametrization. Although they showed impressive results for high-quality 3D mesh reconstruction, it relies on classical landmark and silhouette registration techniques that depend on manually annotated and class-specific 2D/3D landmarks which considerably limit its applicability.

To step further, our work efficiently converts a given unstructured 3D mesh into a regular and compact parametrization that is ready for machine learning applications. Nevertheless, our proposed learning framework is able to reconstruct high-quality 3D meshes from a single image.

3 Proposed Learning Framework

Given a single image from a specific category (*e.g.* bicycle, chair, etc.), our aim is to learn a model to infer a high-quality 3D mesh. Our proposed framework does

not build on classical landmark/silhouette registration techniques which might constrain its applicability due to the need for image annotations. To efficiently represent the 3D mesh data to feed neural networks we employed a compact mesh representation [10,16] that embeds a 3D mesh in a low-dimensional space composed of a 3D model index c, FFD displacements $\mathbf{\Delta P}$ (*e.g.* 32 control points), and sparse linear combination weights $\boldsymbol{\alpha}$ depending on the size of the graph $\mathcal{G}$.

To estimate the 3D shape parameters, we train a multi-label classifier to infer the index and a feedforward neural network to regress the FFD displacements and the sparse linear combination weights from the image latent space learnt from a convolutional autoencoder (CAE). A model is then selected from the graph using the estimated index and the FFD parameters are applied to initially deform the model. Once a satisfactory model candidate is selected and deformed, we have, from the graph, the information about what models are possible to establish dense correspondences. Finally, we apply the sparse linear combination parameters to refine the 3D reconstruction. The framework is shown in Fig. 1.

3.1 3D Mesh Embedding

We parametrize a 3D mesh model using the compact shape representation method presented in [10,16]. It uses a graph $\mathcal{G}$ with 3D mesh models from the same class as nodes and its edges indicate whether we can establish dense correspondences or not (see an example in Fig. 1). Note that the graph is not fully connected but sparse. So in every subgraph we can perform sparse linear combinations to deform a 3D model (*i.e.* a union of subspaces). Mathematically, consider Ω as an index set of the nodes in a certain subgraph with $\mathcal{S}_c(\mathbf{V}_c, \mathbf{F}_c)$ as the central shape node. Here $\mathbf{V}$ and $\mathbf{F}$ stand for the shape vertices and faces. Dense correspondences will always exist for all $i \in \Omega$ allowing us to deform the model $\mathcal{S}(\mathbf{V}, \mathbf{F})$ by linear combination,

$$\mathbf{V} = \alpha_c \mathbf{V}_c + \sum_{i \in \Omega} \alpha_i \mathbf{V}_c^i, \quad \mathbf{F} = \mathbf{F}_c, \tag{1}$$

where α's are the weights. A two-step process is then performed, first we need to find a candidate model (*i.e.* a node) from $\mathcal{G}$. Second, knowing the index that indicates the central node we can deform the model by linearly interpolating it with its dense correspondences from the subgraph. To select a good candidate in the first step, FFD is used to deform a model and pick the one which best fit the image. A 3D mesh can be represented in terms of FFD as

$$\mathbf{S}_{ffd} = \mathbf{B}\mathbf{\Phi}(\mathbf{P} + \mathbf{\Delta P}), \tag{2}$$

where $\mathbf{S}_{ffd} \in \mathbb{R}^{N \times 3}$ are the vertices of the 3D mesh, $\mathbf{B} \in \mathbb{R}^{N \times M}$ is the deformation matrix, $\mathbf{P} \in \mathbb{R}^{M \times 3}$ are the control point coordinates, N and M are the number of vertices and control points respectively, $\mathbf{\Delta P}$ are the control point displacements, and $\mathbf{\Phi} \in \mathbb{R}^{3M \times 3M}$ is a matrix to impose symmetry in the FFD grid as in [16]. The deformation matrix $\mathbf{B}$ is a set of Bernstein polynomial basis[1].

[1] Refer to [10,16] for more details about the graph creation and deformation process.

Once we have the graph, we can embed a 3D mesh in a low-dimensional space. We only need an index that indicates what model we should pick up from the graph, the symmetric FFD displacements $\mathbf{\Delta P}$ to apply the initial deformation to the selected model, and the sparse linear combination weights $\boldsymbol{\alpha}$ to refine the final 3D model. In this work however we tackle the following question: Could such a low-dimensional parametrization be learned to infer high-quality 3D meshes from a single image?

3.2 Collecting Synthetic Data

Since our goal is to infer a 3D mesh parametrization from a single image, we need to synthetically generate data containing 3D meshes, its compact parametrization (c, $\mathbf{\Delta P}$ and $\boldsymbol{\alpha}$ parameters) and rendered images. One may question why we do not simply parametrize the whole ShapeNet, for example, using the shape embedding proposed in [16]. One of the drawbacks of such an approach would be that it relies on 3D semantic landmarks that were manually annotated in some CAD models from ShapeNet to obtain a compact shape embedding. For this reason we decided to use the graph to generate data for each object class instead of manually annotating 3D anchors on several models which is laborious.

To generate the data, we randomly choose an index and then we deform the selected model by applying $\mathbf{\Delta P}$ and $\boldsymbol{\alpha}$ from a learned probability density function (PDF). To learn a PDF for the displacements $\mathbf{\Delta P}$ we use a Gaussian Mixture Model (GMM) to capture information about the nature of the desired deformations. Since we have from the graph creation process the FFD parameters for every pair of 3D models in the graph (obtained during the deformation process to find dense correspondences in [16]), we can learn a GMM from such prior information. For the sparse linear coefficients' PDF we simply fit a normal distribution to a set of α's from 3D reconstructions on the PASCAL3D+ dataset [44] from [16]. Armed with this, we can synthetically generate deformed 3D meshes and images with their respective low-dimensional shape parametrizations.

3.3 Learning the Image Latent Space

To obtain a lower-dimensional representation for the images, a convolutional autoencoder is proposed to extract useful features in an unsupervised manner. The encoder network consists of three fully convolution layers with numbers of kernels $\{8, 16, 32\}$, kernel sizes $\{5, 3, 3\}$, and strides $\{3, 3, 3\}$. The decoder network has three fully transposed convolutional layers with numbers of kernels $\{16, 8, 1\}$, kernel sizes $\{3, 3, 5\}$, and strides $\{3, 3, 3\}$. The input takes grayscale images of size 220×220. All layers use ReLU as activation functions except the last one that uses tanh. This gives us a network flow of size $220^2 \rightarrow 72^2 \rightarrow 24^2 \rightarrow 8^2 \rightarrow 24^2 \rightarrow 72^2 \rightarrow 220^2$, respectively. We take the image latent space $\mathbf{z} \in \mathbb{R}^{2,048}$ from the last layer of the encoder as feature representation.

3.4 Learning the Index to Select a Model

Firstly we need to retrieve a 3D model from the graph (a graph is explained in Subsect. 3.1). We treat it as a multi-label classification problem. For example, if a graph of the class 'car' would have 30 different cars/nodes it would have 30 indices/labels (1 to 30) that an image of a car could be classified as. In this way, the "closest" car to the image can be selected from the graph according to the estimated index. For this purpose, we propose a simple yet effective multi-label classifier to estimate graph indices. The input is the image latent space from the convolutional autoencoder of size 2,048 ($8 \times 3 \times 3$). The output is a one-hot encoded vector that represents the ground truth indices. The network has one hidden layer of size 1,050 and ReLU is used as activation function.

3.5 Learning the Shape Parameters

We wish to learn a mapping from the image feature representation $\mathbf{z}$ of size 2,048 to its corresponding 3D shape parameters $\mathbf{\Delta P}$ and $\boldsymbol{\alpha}$, *i.e.* $f : \mathbf{z} \rightarrow \{\mathbf{\Delta P}, \boldsymbol{\alpha}\}$. Given the training set of image features and the ground truth shape parameters, we learn this mapping using a feedforward neural network. The input is the image latent space of size 2,048. The output is a vector containing the shape parameters $\boldsymbol{\kappa} \in \mathbb{R}^{MN}$ where M and N are the number of FFD and sparse linear combination parameters $\boldsymbol{\alpha}$ respectively. For instance, $\boldsymbol{\kappa} \in \mathbb{R}^{126}$, where 96 values would be the FFD parameters (32×3) and the remaining 30 values would be the $\boldsymbol{\alpha}$ parameters for the sparse linear combination of models in the graph. To handle different number of α coefficients that might differ according to the subgraph selected, we consider a fixed-size vector according to the size of $\mathcal{G}$ and then we pick only the estimated α's corresponding to the subgraph (*i.e.* we ignore the other α's). The network has one hidden layer of size 1,500 and it uses ReLU as activation function.

4 Experiments

Dataset. To train our framework we take the approach of synthesizing 3D mesh models using the strategy discussed in the Subsect. 3.2. We generated 5,000 deformed 3D models of eight object categories (car, bicycle, motorbike, aeroplane, bus, chair, dining table, and sofa) using the graphs from [16]. Every graph has 30 CAD models sampled from ShapeNet [18] except for the bicycle and motorbike graphs that have 21 and 27 CAD models, respectively. We rendered for every 3D synthesised model a 2D view of size 256×192 using different viewpoints with a white background. We also produced uniform lighting across the surfaces of the object. With the images and the ground truth 3D meshes, indices and shape parameters, we can train our framework and evaluate its performance. The data was split in 70% for training and 30% for testing.

Evaluation Metrics. To quantify the quality of the classification step we employed the accuracy, precision and recall metrics. To evaluate the estimated

shape parameters we use the mean squared error (MSE). For the 3D shape reconstruction measure we use the symmetric surface distance s_{dist} to the ground truth. s_{dist} is computed by densely sampling points on the faces and using normalized points distance to estimate the model similarity and it is defined as

$$dist_{3D} = \frac{1}{|\hat{\mathbf{V}}|} \sum_{\mathbf{v_i} \in \hat{\mathbf{V}}} dist(\mathbf{v_i}, \mathcal{S}) + \frac{1}{|\mathbf{V}|} \sum_{\mathbf{v_i} \in \mathbf{V}} dist(\mathbf{v_i}, \hat{\mathcal{S}}), \tag{3}$$

where $\hat{\mathbf{V}}$, $\hat{\mathcal{S}}$, $\mathbf{V}$, $\mathcal{S}$ are the estimated vertices and surfaces, and the ground truth vertices and surfaces, respectively.

Moreover, we use the intersection over union (IoU) as a metric to compare different voxel models as in [11] defined as $(\hat{\mathcal{V}} \cap \mathcal{V})/(\hat{\mathcal{V}} \cup \mathcal{V})$, where $\hat{\mathcal{V}}$ and $\mathcal{V}$ are the voxel models of the estimated and ground truth models respectively.

Table 1. Evaluation of our method on synthetic data. We show the performance of the convolutional autoencoder (CAE), the multi-label classification for the 3D model selection, the feedforward network for the parameters estimation, and the 3D reconstruction from single image. *Acc*, *Prec*, *Rec*, and t stand for the accuracy, precision, recall, and the training time, respectively.

	CAE		3D model selection				Params estimation		3D reconstruction	
	MSE	$\sim t$ (min)	$Acc(\%)$	$Prec(\%)$	$Rec(\%)$	$\sim t$ (min)	MSE	$\sim t$ (min)	$dist_{3D}$	IoU
Car	**0.0012**	30	92.13	92.11	93.33	76	0.0176	197	0.006	0.664
Bicycle	0.0068	25	92.07	92.00	92.18	73	0.0102	208	0.025	0.795
motorbike	0.0045	21	**97.80**	**97.87**	**97.72**	74	0.0150	322	0.007	0.679
Aeroplane	0.0017	24	77.33	76.90	78.09	96	0.0108	209	0.023	0.551
Bus	0.0013	35	90.20	90.05	92.14	74	0.0329	156	**0.003**	**0.776**
Chair	0.0022	21	87.33	86.97	88.74	75	**0.0090**	158	0.034	0.403
Dining table	0.0020	22	73.60	73.60	74.98	75	0.0119	158	0.165	0.332
Sofa	0.0013	21	76.20	76.47	77.69	74	0.0119	156	0.063	0.402
Mean	0.0024	25	85.86	85.75	86.86	77	0.0144	195	0.041	0.575

4.1 Estimating the Image Latent Space

The first set of experiments were performed on the CAE to learn a latent representation from an image. We found the architecture described in the Subsect. 3.3 to have the better performance. The image feature representation is discriminative and performed well on the classifier and on the feedforward network to estimate the shape parameters. Table 1 shows the MSE on the test set and the time spent training the network for every class used. We used the MSE evaluated on the test set as a measure of how close the input image (ground truth) is from the image generated by the decoder. It does not mean the latent space is representative as image descriptors but we further validated it in the classification and regression steps by achieving high accuracies. This means that the descriptors learnt are indeed discriminative.

4.2 Selecting a 3D Mesh

The first stage of our learning framework after having the image latent space is the selection of a 3D mesh from the given graph. The performance of the proposed multi-label classifier is shown in Table 1. One can note that the overall performance on the test set was satisfactory in terms of accuracy (85.68%), precision (85.75%) and recall (86.86%). The best performance was achieved on the motorbike category (27 labels) with an accuracy of 97.80%. The category has very different motorbikes from each other which explains the great performance. Besides, the dining table category (30 labels) had the lowest performance, with an accuracy of 73.60%. This is presumably due to the high degree of similarity between the synthesised images as there are not many unique tables in this object class. Moreover, we fixed the network architecture for all classes. Fine tuning the classifier for specific classes would more than likely improve performance.

4.3 Estimating the Shape Parameters

The last stage of our framework before the final 3D reconstruction is the estimation of the shape parameters. The results are also shown in Table 1. The overall MSE (0.0144) on the testing set shows that the network is indeed learning a mapping function to estimate the FFD and the $\boldsymbol{\alpha}$ parameters from the image latent space. The graphs used have about 30 mesh models each which means that once we have selected a model we can establish dense correspondences with up to 29 models (29 α values). The resolution of the FFD grid is of 4^3 that gives us 64 control points to free-deform the model. Since the majority of man-made object are symmetric, we impose a symmetry on the FFD grid so that the deformations are forced to be symmetric and more realistic. Therefore, we have to estimate only half of the FFD parameters. The feedforward network then maps the image latent space to 32 displacements of the control points in the 3D space, $\boldsymbol{\Delta P} \in \mathbb{R}^{32\times 3}$, and to 30 sparse linear combinations parameters, $\boldsymbol{\alpha} \in \mathbb{R}^{30}$ (29 + the model selected). The estimated parameters, in this case, is of size $\boldsymbol{\kappa} \in \mathbb{R}^{126}$. One can note that it is a very low-dimensional parametrization that is efficiently learned through a simple and lightweight network architecture.

4.4 3D Reconstruction from a Single Image

Given a single image we can forward pass it to our learned framework to estimate an index c and the shape parameters $\boldsymbol{\kappa}$. A 3D mesh is initially selected from the class-specific graph by the estimated index c. Afterwards, the FFD displacements $\boldsymbol{\Delta P}$ is applied to free-deform the model using Eq. 2, $\mathbf{S}_{ffd} = \mathbf{B\Phi}(\mathbf{P} + \boldsymbol{\Delta P})$. Note that we only need to add the estimated displacements to the initial grid of control points $\mathbf{P}$. Finally, we can apply the linear combination parameters $\boldsymbol{\alpha}$ to deform the model through Eq. 1, $\mathbf{V} = \alpha_c \mathbf{V}_c + \sum_{i\in\Omega} \alpha_i \mathbf{V}_c^i$.

3D Reconstruction from Synthetic Images. The initial experiments were performed on synthetic images from the 8 classes where we have the ground

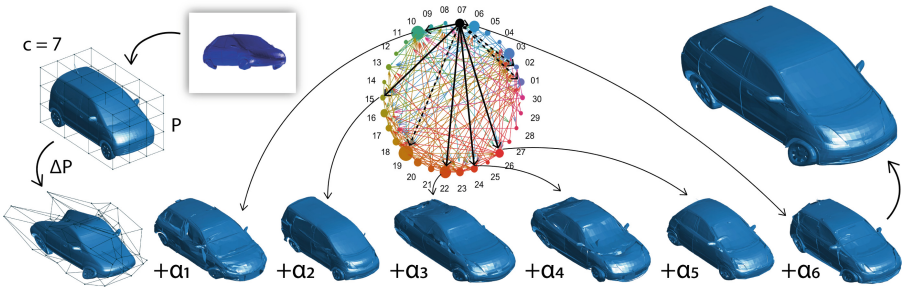

Fig. 2. Given an input image of a car and a graph $\mathcal{G}$ with 30 models as nodes, our method first selected the model 7 from $\mathcal{G}$. Then it is deformed by the FFD displacements $\mathbf{\Delta P}$ on the initial grid $\mathbf{P}$. Afterwards, the linear combination is performed with the estimated $\boldsymbol{\alpha}$ to reconstruct the final model. The black arrows on $\mathcal{G}$ show what models are possible to perform linear combination with the selected model (*i.e.* models in dense correspondences). Note that not all the models were selected, but 6 out of 9 models in this example. For illustration, the node size in $\mathcal{G}$ is proportional to the number of edges starting from the node.

truth 3D meshes and also the shape embedding parameters. Figure 2 shows a real example of our framework flow. Table 1 summarizes the quantitative results of our 3D reconstruction on the synthetic test set. We measured the quality of the 3D reconstruction through the surface distance metric $dist_{3D}$ and the IoU between the reconstructed and the ground truth model. Our framework clearly performed well on the synthetic dataset according to the surface distance metric. The IoU for the classes aeroplane, chair, dining table, and sofa had the lowest values which means that a good voxel intersection between the reconstructed model and the ground truth was not possible. In fact, IoU between thin structures (*e.g.* chair's legs, aeroplane's wings, etc.) are low if they are not well aligned.

Qualitative results are shown in Fig. 3. Our proposed learning framework performed well at selecting a proper model to start the deformation process, and also at estimating the shape parameters to obtain the final deformed mesh. In the successful cases, one can see the final models are similar to the ground truth with slight differences that can be hard to point them out. An interesting example to show the expressiveness of our proposed method is the chair instance. One can note that the selected chair has long legs, but the estimated FFD parameters managed to deform the chair to get shorter legs before applying the linear combination parameters to get the final model. A failure case is shown on the last row in red where an "incorrect" model was selected from the graph, in this case a fighter jet instead of a commercial airplane. This can in part be explained by the challenging image perspective of this instance. Even for a human it is difficult to correctly classify such an image, in this case a fighter jet is in fact a highly plausible choice of model[2].

[2] More results, failure cases, and videos can be found in the supplementary material.

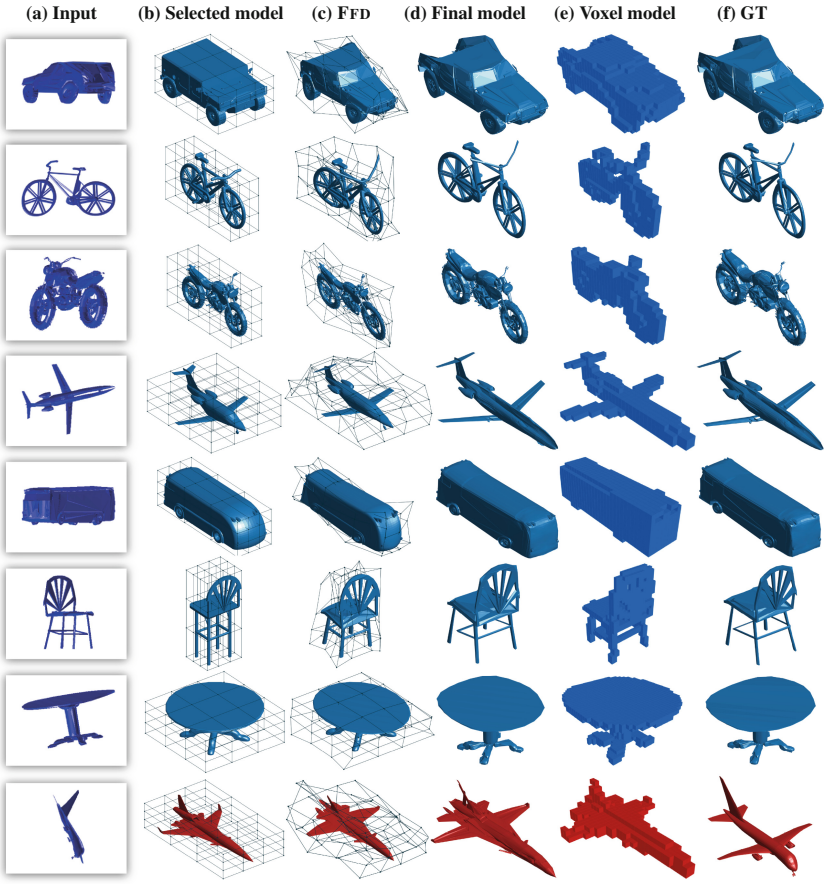

Fig. 3. Visual results on synthetic data. (a) shows the input image; (b) the selected model from the graph; (c) the selected model deformed by FFD. The final 3D model deformed by linear combination is shown in (d). The voxelized final model is shown in (e) and the ground truth in (f). In the success cases (blues), one can note the final models are similar to the ground truth with slight differences that can be hard to point it out. A failure case is shown on the last row in red where a "wrong" model was selected from the graph. (Color figure online)

3D Reconstruction from Real World Images. To verify our framework's generalization capacity we test it on a dataset with real world images. We evaluated the performance of the proposed method on the PASCAL3D+ dataset and we compared to the results presented in [16]. We found that it is a fair comparison since we are not playing with volumetric or point cloud representations but with dense polygonal meshes. In order to forward pass the real world images to our learning framework, we used the image silhouettes provided by the PASCAL3D+ dataset since our framework was trained on images with uniform

Table 2. Evaluation of our method on the PASCAL3D+ dataset and comparison with the method presented in [16].

	[16]		Ours	
	$dist_{3D}$	IoU	$dist_{3D}$	IoU
Car	**0.174**	**0.382**	0.179	0.371
Bicycle	0.290	**0.419**	**0.282**	0.402
Motorbike	**0.084**	**0.384**	0.186	0.309
Aeroplane	0.262	**0.442**	**0.153**	0.366
Bus	0.091	**0.376**	**0.058**	0.280
Chair	**0.309**	**0.261**	0.461	0.236
Dining table	**0.353**	**0.256**	0.695	0.223
Sofa	**0.346**	**0.241**	0.573	0.207
Mean	**0.239**	**0.345**	0.323	0.299

background. Table 2 summarizes the results of our method and the results presented in [16] in terms of the surface distance $dist_{3D}$ and the IoU. Our method did not outperform the method proposed in [16], except for some classes. However, we achieved a similar performance on the real world dataset neither relying on silhouette and landmark registration algorithms nor using class-specific landmarks. Moreover, since the ground truth models in the PASCAL3D+ dataset were aligned to images by humans, the comparison metrics are not robust. As stated in [16], most of the 3D reconstructions look closer to the images than the

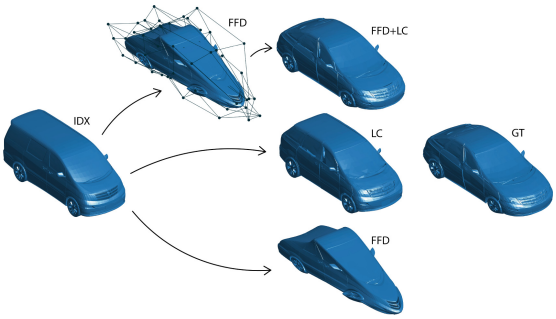

Fig. 4. Qualitative example showing the importance of every step in our method. The upper part shows the FFD and the linear combination (LC) being applied to the selected model (IDX). It shows that even with a strong FFD deformation the LC step managed to deform the mesh to look similar to the GT model. The bottom part shows the 3D reconstruction when omitting the FFD step. The LC step managed to deform the van into a compact car (perhaps SUV?) but it is still different from the GT. If we omit the LC step one can notice that the FFD model looks very different from the GT.

ground truth models themselves. This explains the high values for the surface distance and the low values for the IoU.

Qualitative results are shown in Fig. 5. One can see that our proposed method performed well on a real-world dataset. In the motorbike example it is clear when looking at the image that the motorbike does not have a backrest. The selected model was a good choice, since it shares topological similarities and although it has a backrest device, the linear combination step managed to diminish it. Another interesting example is the airplane where the selected model has a different type of wings, but the deforming process made it appear much more similar to the input image.

Ablation Study. We performed an extensive ablation experiment where the 3D reconstruction error is evaluated at each step, model selection, FFD and linear combination, to shows how sensitive the model is to the noise of each step. A qualitative example showing the importance of every step is shown in Fig. 4. Please refer to the supplementary material for the quantitative analysis.

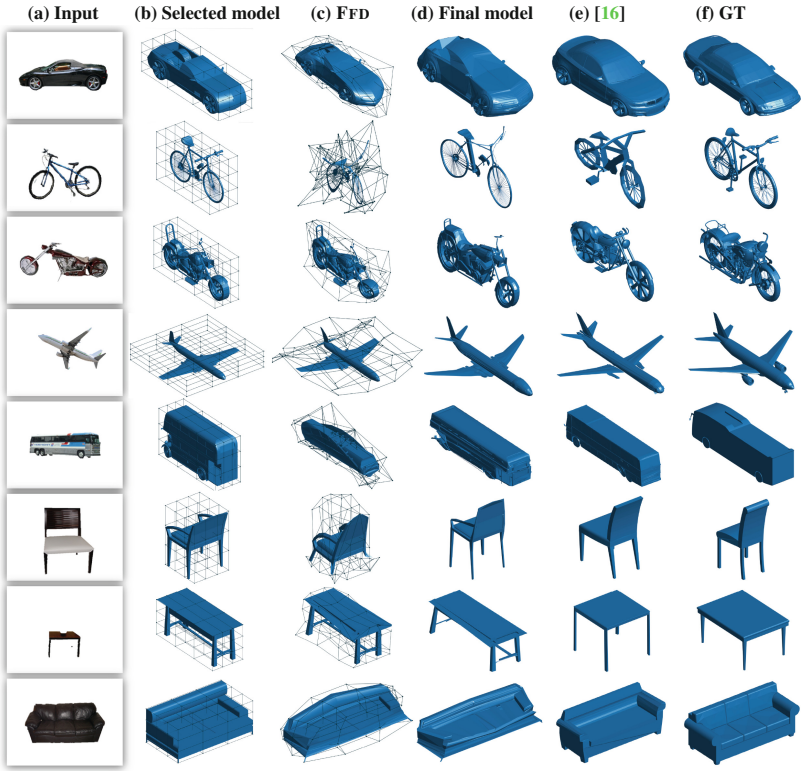

Fig. 5. Visual results on real-world data. (a) shows the input image; (b) the selected model; (c) the selected model deformed by FFD. The final 3D model reconstructed by linear combination is shown in (d). We compare with [16] in (e) and the ground truth is shown in (f).

4.5 Implementation Details and Limitations

We designed our networks using PyTorch [45] on a GPU Nvidia Tesla M40. We trained the convolutional autoencoder using a MSE loss function and the Adam optimizer [46] with a learning rate of $1e^{-3}$ and a weight decay of $1e^{-5}$ during 100 epochs. To train the multi-label classifier we used a multi-label soft margin loss function and for the feedforward neural network we trained it using a MSE loss function. For both models we used the Adam optimizer with a learning rate of $1e^{-3}$ during 1,000 epochs.

One of the main limitations, which our method inherits from [16], is the need for a good embedding graph. One can see in Fig. 2 that some models for the linear combination step have some crinkles on its surfaces. This happens during the graph construction when searching for dense correspondences between the models. This is especially important for achieving high-quality 3D reconstructions. However, finding dense correspondences between two different models that do not share the same number of vertices is still an open problem [16]. Another limitation is that we synthesise the images with white background so real images must be segmented beforehand for the 3D reconstruction. GANs can fit in this context to generate more realistic images.

5 Discussion

The proposed method is category-specific, meaning that we need one model for every class. Although [36] is able to better generalize to multiple object categories with a single model trained on 2k classes, we believe our paper makes a step in 3D reconstruction in the mesh domain which has largely been unexplored. Moreover, we would like to reinforce the generalization capacity of our method to specific classes, where a single graph embedding model is able to generalize to unseen 3D meshes by using FFD and linear combination in a small dictionary. We successfully validated this using the PASCAL3D+ dataset since its shape distribution differs from the trained models on synthetic data.

We would like to reiterate our argument that even though [16] performed better in some categories, our method is able to achieve similar fine-grained mesh reconstruction without the need to rely on any class-specific landmarks or any silhouette registration whatsoever. This is a significant improvement in the utility of these approaches with the removal of these limiting constraints.

A comparison with [36] would not be entirely straightforward or terribly informative as we propose a mesh representation with fine-grained geometry whereas [36] proposes a coarse point cloud representation. Moreover, IoU of coarse volumetric models is not a robust metric to capture fine-grained geometry contained on the surface of mesh models. From visual inspection, one can observe that our deformation of mesh models and their detailed geometry clearly outperforms [36] and [34]. We considered a comparison with [34], however the authors have not yet shared their code and the information found in the paper is not enough to reproduce their results. For this reason we decided to restrict our comparisons to competing methods as [16] (and implicitly with [10]).

6 Conclusion

We have proposed a simple yet effective learning framework to infer 3D meshes from a single image using a compact mesh representation that does not rely on class-specific object landmarks which would limit its applicability. A 3D mesh is embedded in a low-dimensional space that allows one to perform deformations by FFD and sparse linear combination. Experiments on synthetic and real-world datasets show that our method convincingly reconstructs 3D meshes from a single image with fine-scaled geometry not yet achieved in previous works that rely on volumetric and point cloud representations. Although our method relies on background segmentation, we do believe the field is mature to provide off-the-shelf segmentation techniques for a practical application. Such high quality 3D representation and reconstruction as proposed in our work is extremely important, especially to unlock virtual and augmented reality applications. Finally, we believe that this work is a great first step towards more effective mesh representations for 3D geometric learning purposes.

Acknowledgements. This research was supported by the grants ARC DP170100632, ARC FT170100072 and NSF 1526033.

References

1. Wang, C., Wang, Y., Lin, Z., Yuille, A.L., Gao, W.: Robust estimation of 3D human poses from a single image. In: CVPR (2014)
2. Zhou, X., Leonardos, S., Hu, X., Daniilidis, K.: 3D shape estimation from 2D landmarks: a convex relaxation approach. In: CVPR (2015)
3. Kar, A., Tulsiani, S., Carreira, J., Malik, J.: Category-specific object reconstruction from a single image. In: CVPR (2015)
4. Rock, J., Gupta, T., Thorsen, J., Gwak, J., Shin, D., Hoiem, D.: Completing 3D object shape from one depth image. In: CVPR (2015)
5. Zhou, X., Zhu, M., Leonardos, S., Derpanis, K.G., Daniilidis, K.: Sparseness meets deepness: 3D human pose estimation from monocular video. In: CVPR (2016)
6. Wu, J., et al.: Single image 3D interpreter network. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9910, pp. 365–382. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46466-4_22
7. Kong, C., Zhu, R., Kiani, H., Lucey, S.: Structure from category: a generic and prior-less approach. In: 3DV (2016)
8. Bansal, A., Russell, B., Gupta, A.: Marr revisited: 2D–3D model alignment via surface normal prediction. In: CVPR (2016)
9. Han, K., Wong, K.Y.K., Tan, X.: Single view 3D reconstruction under an uncalibrated camera and an unknown mirror sphere. In: 3DV (2016)
10. Kong, C., Lin, C.H., Lucey, S.: Using locally corresponding CAD models for dense 3D reconstructions from a single image. In: CVPR (2017)
11. Choy, C.B., Xu, D., Gwak, J.Y., Chen, K., Savarese, S.: 3D-R2N2: a unified approach for single and multi-view 3D object reconstruction. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9912, pp. 628–644. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46484-8_38

12. Sharma, A., Grau, O., Fritz, M.: VConv-DAE: deep volumetric shape learning without object labels. In: Hua, G., Jégou, H. (eds.) ECCV 2016. LNCS, vol. 9915, pp. 236–250. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-49409-8_20
13. Tatarchenko, M., Dosovitskiy, A., Brox, T.: Octree generating networks: efficient convolutional architectures for high-resolution 3D outputs. In: ICCV (2017)
14. Riegler, G., Ulusoy, A.O., Geiger, A.: OctNet: learning deep 3D representations at high resolutions. In: CVPR (2017)
15. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: PointNet++: deep hierarchical feature learning on point sets in a metric space. In: NIPS (2017)
16. Pontes, J.K., Kong, C., Eriksson, A., Fookes, C., Lucey, S.: Compact model representation for 3D reconstruction. In: 3DV (2017)
17. Sederberg, T., Parry, S.: Free-form deformation of solid geometric models. In: SIGGRAPH (1986)
18. Chang, A.X., et al.: ShapeNet: an information-rich 3D model repository. Technical report arXiv:1512.03012 [cs.GR] (2015)
19. Wu, Z., Song, S., Khosla, A., Tang, X., Xiao, J.: 3D ShapeNets: a deep representation for volumetric shapes. In: CVPR (2015)
20. Ulusoy, A.O., Geiger, A., Black, M.J.: Towards probabilistic volumetric reconstruction using ray potential. In: 3DV (2015)
21. Cherabier, I., Häne, C., Oswald, M.R., Pollefeys, M.: Multi-label semantic 3D reconstruction using voxel blocks. In: 3DV (2016)
22. Rezende, D.J., Eslami, S.M.A., Mohamed, S., Battaglia, P., Jaderberg, M., Heess, N.: Unsupervised learning of 3D structure from images. In: NIPS (2016)
23. Yan, X., Yang, J., Yumer, E., Guo, Y., Lee, H.: Perspective transformer nets: learning single-view 3D object reconstruction without 3D supervision. In: NIPS (2016)
24. Qi, C.R., Su, H., Nießner, M., Dai, A., Yan, M., Guibas, L.J.: Volumetric and multi-view CNNs for object classification on 3D data. In: CVPR (2016)
25. Kar, A., Häne, C., Malik, J.: Learning a multi-view stereo machine. In: NIPS (2017)
26. Zhu, R., Galoogahi, H.K., Wang, C., Lucey, S.: Rethinking reprojection: closing the loop for pose-aware shape reconstruction from a single image. In: NIPS (2017)
27. Wu, J., Wang, Y., Xue, T., Sun, X., Freeman, W.T., Tenenbaum, J.B.: MarrNet: 3D shape reconstruction via 2.5D sketches. In: NIPS (2017)
28. Liao, Y., Donné, S., Geiger, A.: Deep marching cubes: learning explicit surface representations. In: CVPR (2018)
29. Girdhar, R., Fouhey, D.F., Rodriguez, M., Gupta, A.: Learning a predictable and generative vector representation for objects. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9910, pp. 484–499. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46466-4_29
30. Wu, J., Zhang, C., Xue, T., Freeman, W.T., Tenenbaum, J.B.: Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. In: NIPS (2016)
31. Liu, J., Yu, F., Funkhouser, T.A.: Interactive 3D modeling with a generative adversarial network. In: 3DV (2017)
32. Gwak, J., Choy, C.B., Garg, A., Chandraker, M., Savarese, S.: Weakly supervised generative adversarial networks for 3D reconstruction. In: 3DV (2017)
33. Wang, P.S., Liu, Y., Guo, Y.X., Sun, C.Y., Tong, X.: O-CNN: octree-based convolutional neural networks for 3D shape analysis. In: SIGGRAPH (2017)
34. Häne, C., Tulsiani, S., Malik, J.: Hierarchical surface prediction for 3D object reconstruction. In: 3DV (2017)

35. Li, J., Chen, B.M., Lee, G.H.: SO-Net: self-organizing network for point cloud analysis. In: CVPR (2018)
36. Fan, H., Su, H., Guibas, L.J.: A point set generation network for 3D object reconstruction from a single image. In: CVPR (2017)
37. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: PointNet: deep learning on point sets for 3D classification and segmentation. In: CVPR (2017)
38. Lin, C.H., Kong, C., Lucey, S.: Learning efficient point cloud generation for dense 3D object reconstruction. In: AAAI (2018)
39. Kurenkov, A., et al.: DeformNet: free-form deformation network for 3D shape reconstruction from a single image. In: WACV (2018)
40. Nan, L., Wonka, P.: PolyFit: polygonal surface reconstruction from point clouds. In: ICCV (2017)
41. Shin, D., Fowlkes, C.C., Hoiem, D.: Pixels, voxels, and views: a study of shape representations for single view 3d object shape prediction. In: CVPR (2018)
42. Sinha, A., Unmesh, A., Huang, Q., Ramani, K.: SurfNet: generating 3D shape surfaces using deep residual network. In: CVPR (2017)
43. Yumer, M.E., Mitra, N.J.: Learning semantic deformation flows with 3D convolutional networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9910, pp. 294–311. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46466-4_18
44. Xiang, Y., Mottaghi, R., Savarese, S.: Beyond PASCAL: a benchmark for 3D object detection in the wild. In: WACV (2014)
45. Paszke, A., et al.: Automatic differentiation in PyTorch. In: NIPS-W (2017)
46. Diederik, K., Jimmy, B.: Adam: a method for stochastic optimization. In: ICLR (2014)