3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction

Christopher B. Choy Danfei Xu* JunYoung Gwak*
Kevin Chen Silvio Savarese

Stanford University
{chrischoy, danfei, jgwak, kchen92, ssilvio}@stanford.edu

Abstract. Inspired by the recent success of methods that employ shape priors to achieve robust 3D reconstructions, we propose a novel recurrent neural network architecture that we call the 3D Recurrent Reconstruction Neural Network (3D-R2N2). The network learns a mapping from images of objects to their underlying 3D shapes from a large collection of synthetic data [1]. Our network takes in one or more images of an object instance from arbitrary viewpoints and outputs a reconstruction of the object in the form of a 3D occupancy grid. Unlike most of the previous works, our network does not require any image annotations or object class labels for training or testing. Our extensive experimental analysis shows that our reconstruction framework i) outperforms the state-of-the-art methods for single view reconstruction, and ii) enables the 3D reconstruction of objects in situations when traditional SFM/SLAM methods fail (because of lack of texture and/or wide baseline).

Keywords: multi-view, reconstruction, recurrent neural network

1 Introduction

Rapid and automatic 3D object prototyping has become a game-changing innovation in many applications related to e-commerce, visualization, and architecture, to name a few. This trend has been boosted now that 3D printing is a democratized technology and 3D acquisition methods are accurate and efficient [2]. Moreover, the trend is also coupled with the diffusion of large scale repositories of 3D object models such as ShapeNet [1].

Most of the state-of-the-art methods for 3D object reconstruction, however, are subject to a number of restrictions. Some restrictions are that: i) objects must be observed from a dense number of views; or equivalently, views must have a relatively small baseline. This is an issue when users wish to reconstruct the object from just a handful of views or ideally just one view (see Fig. 1(a)); ii) objects' appearances (or their reflectance functions) are expected to be Lambertian (i.e. non-reflective) and the albedos are supposed be non-uniform (i.e., rich of non-homogeneous textures).

* indicates equal contribution.

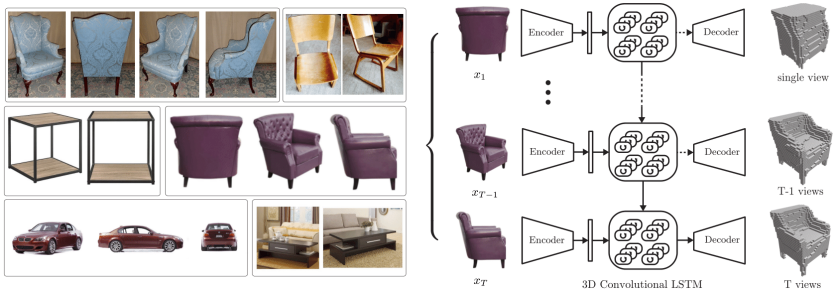
These restrictions stem from a number of key technical assumptions. One typical assumption is that features can be matched across views [3,4,5,6] as hypothesized by the majority of the methods based on SFM or SLAM [7,8]. It has been demonstrated (for instance see [9]) that if the viewpoints are separated by a large baseline, establishing (traditional) feature correspondences is extremely problematic due to local appearance changes or self-occlusions. Moreover, lack of texture on objects and specular reflections also make the feature matching problem very difficult [10,11].

In order to circumvent issues related to large baselines or non-Lambertian surfaces, 3D volumetric reconstruction methods such as space carving [12,13,14,15] and their probabilistic extensions [16] have become popular. These methods, however, assume that the objects are accurately segmented from the background or that the cameras are calibrated, which is not the case in many applications.

A different philosophy is to assume that prior knowledge about the object appearance and shape is available. The benefit of using priors is that the ensuing reconstruction method is less reliant on finding accurate feature correspondences across views. Thus, shape prior-based methods can work with fewer images and with fewer assumptions on the object reflectance function as shown in [17,18]. The shape priors are typically encoded in the form of simple 3D primitives as demonstrated by early pioneering works [19,20] or learned from rich repositories of 3D CAD models [21,22,23], whereby the concept of fitting 3D models to images of faces was explored to a much larger extent [24,25,26]. Sophisticated mathematical formulations have also been introduced to adapt 3D shape models to observations with different degrees of supervision [27] and different regularization strategies [28].

This paper is in the same spirit as the methods discussed above, but with a key difference. Instead of trying to match a suitable 3D shape prior to the observation of the object and possibly adapt to it, we use deep convolutional neural networks to learn a mapping from observations to their underlying 3D shapes of objects from a large collection of training data. Inspired by early works that used machine learning to learn a 2D-to-3D mapping for scene understanding [29,30], data driven approaches have been recently proposed to solve the daunting problem of recovering the shape of an object from just a single image [31,32] for a given number of object categories. In our approach, however, we leverage for the first time the ability of deep neural networks to automatically learn, in a mere end-to-end fashion, the appropriate intermediate representations from data to recover approximated 3D object reconstructions from as few as a single image with minimal supervision.

Inspired by the success of Long Short-Term Memory (LSTM) [33] networks [34,35] as well as recent progress in single-view 3D reconstruction using Convolutional Neural Networks [36,37], we propose a novel architecture that we call the 3D Recurrent Reconstruction Neural Network (3D-R2N2). The network takes in one or more images of an object instance from different viewpoints and outputs a reconstruction of the object in the form of a 3D occupancy grid, as illustrated in Fig. 1(b). Note that in both training and testing, our network does not require



(a) Images of objects we wish to reconstruct (b) Overview of the network

Fig. 1. (a) Some sample images of the objects we wish to reconstruct - notice that views are separated by a large baseline and objects' appearance shows little texture and/or are non-lambertian. (b) An overview of our proposed **3D-R2N2**: The network takes a sequence of images (or just one image) from arbitrary (uncalibrated) viewpoints as input (in this example, 3 views of the armchair) and generates voxelized 3D reconstruction as an output. The reconstruction is incrementally refined as the network sees more views of the object.

any object class labels or image annotations (i.e., no segmentations, keypoints, viewpoint labels, or class labels are needed).

One of the key attributes of the 3D-R2N2 is that it can selectively update hidden representations by controlling *input* gates and *forget* gates. In training, this mechanism allows the network to adaptively and consistently learn a suitable 3D representation of an object as (potentially conflicting) information from different viewpoints becomes available (see Fig. 1).

The main contributions of this paper are summarized as follows:

- We propose an extension of the standard LSTM framework that we call the 3D Recurrent Reconstruction Neural Network which is suitable for accommodating multi-view image feeds in a principled manner.
- We unify single- and multi-view 3D reconstruction in a single framework.
- Our approach requires minimal supervision in training and testing (just bounding boxes, but no segmentation, keypoints, viewpoint labels, camera calibration, or class labels are needed).
- Our extensive experimental analysis shows that our reconstruction framework outperforms the state-of-the-art method for single-view reconstruction [32].
- Our network enables the 3D reconstruction of objects in situations when traditional SFM/SLAM methods fail (because of lack of texture or wide baselines).

An overview of our reconstruction network is shown in Fig. 1(b). The rest of this paper is organized as follows. In Section 2, we give a brief overview of LSTM and GRU networks. In Section 3, we introduce the 3D Recurrent Reconstruction Neural Network architecture. In Section 4, we discuss how we generate training data and give details of the training process. Finally, we present test results of our approach on various datasets including PASCAL 3D and ShapeNet in Section 5.

2 Recurrent Neural Network

In this section we provide a brief overview of Long Short-Term Memory (LSTM) networks and a variation of the LSTM called Gated Recurrent Units (GRU).

Long Short-Term Memory Unit. One of the most successful implementations of the hidden states of an RNN is the Long Short Term Memory (LSTM) unit [33]. An LSTM unit explicitly controls the flow from input to output, allowing the network to overcome the vanishing gradient problem [33,38]. Specifically, an LSTM unit consists of four components: memory units (a memory cell and a hidden state), and three gates which control the flow of information from the input to the hidden state (*input gate*), from the hidden state to the output (*output gate*), and from the previous hidden state to the current hidden state (*forget gate*). More formally, at time step t when a new input x_t is received, the operation of an LSTM unit can be expressed as:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{1}$$
$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{2}$$
$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \tag{3}$$
$$s_t = f_t \odot s_{t-1} + i_t \odot \tanh(W_s x_t + U_s h_{t-1} + b_s) \tag{4}$$
$$h_t = o_t \odot \tanh(s_t) \tag{5}$$

i_t, f_t, o_t refer to the input gate, the output gate, and the forget gate, respectively. s_t and h_t refer to the memory cell and the hidden state, respectively. We use $\odot$ to denote element-wise multiplication and the subscript t to refer to an activation at time t. $W_{(\cdot)}, U_{(\cdot)}$ are matrices that transform the current input x_t and the previous hidden state h_{t-1}, respectively, and $b_{(\cdot)}$ represents the biases.

Gated Recurrent Unit. A variation of the LSTM unit is the Gated Recurrent Unit (GRU) proposed by Cho et al. [39]. An advantage of the GRU is that there are fewer computations compared to the standard LSTM. In a GRU, an update gate controls both the input and forget gates. Another difference is that a *reset gate* is applied before the nonlinear transformation. More formally,

$$u_t = \sigma(W_u \mathcal{T} x_t + U_u * h_{t-1} + b_f) \tag{6}$$
$$r_t = \sigma(W_i \mathcal{T} x_t + U_i * h_{t-1} + b_i) \tag{7}$$
$$h_t = (1 - u_t) \odot h_{t-1} + u_t \odot \tanh(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h) \tag{8}$$

u_t, r_t, h_t represent the update gate, the reset gate, and the hidden state respectively. We follow the same notations as LSTM for matrices and biases.

3 3D Recurrent Reconstruction Neural Network

In this section, we introduce a novel architecture named the 3D Recurrent Reconstruction Network (3D-R2N2), which builds upon the standard LSTM and GRU. The goal of the network is to perform both single- and multi-view 3D

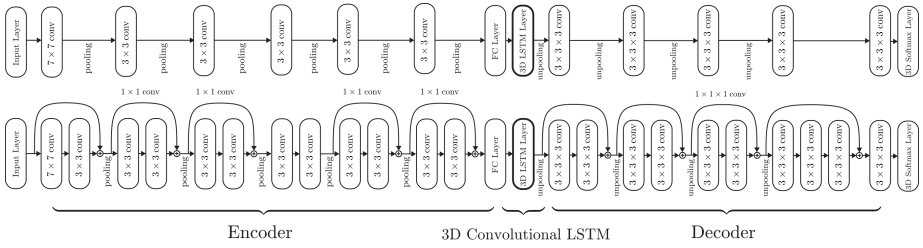


Fig. 2. Network architecture: Each 3D-R2N2 consists of an encoder, a recurrence unit and a decoder. After every convolution layer, we place a LeakyReLU nonlinearity. The encoder converts a 127×127 RGB image into a low-dimensional feature which is then fed into the 3D-LSTM. The decoder then takes the 3D-LSTM hidden states and transforms them to a final voxel occupancy map. After each convolution layer is a LeakyReLU. We use two versions of 3D-R2N2: (top) a shallow network and (bottom) a deep residual network [40].

reconstructions. The main idea is to leverage the power of LSTM to retain previous observations and incrementally refine the output reconstruction as more observations become available.

The network is made up of three components: a 2D Convolutional Neural Network (2D-CNN), a novel architecture named 3D Convolutional LSTM (3D-LSTM), and a 3D Deconvolutional Neural Network (3D-DCNN) (see Fig. 2). Given one or more images of an object from arbitrary viewpoints, the 2D-CNN first encodes each input image x into low dimensional features $\mathcal{T}(x)$ (Section 3.1). Then, given the encoded input, a set of newly proposed 3D Convolutional LSTM (3D-LSTM) units (Section 3.2) either selectively update their cell states or retain the states by closing the input gate. Finally, the 3D-DCNN decodes the hidden states of the LSTM units and generates a 3D probabilistic voxel reconstruction (Section 3.3).

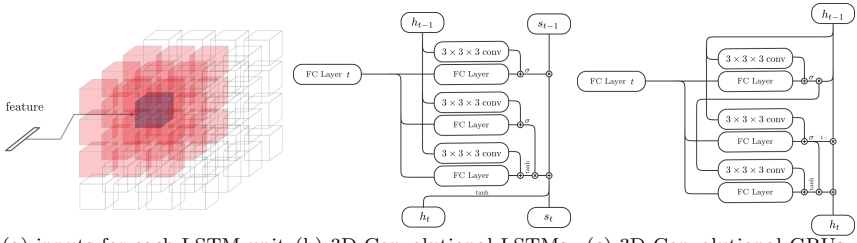
The main advantage of using an LSTM-based network comes from its ability to effectively handle object self-occlusions when multiple views are fed to the network. The network selectively updates the memory cells that correspond to the visible parts of the object. If a subsequent view shows parts that were previously self-occluded and mismatch the prediction, the network would update the LSTM states for the previously occluded sections but retain the states of the other parts (Fig. 2).

3.1 Encoder: 2D-CNN

We use CNNs to encode images into features. We designed two different 2D-CNN encoders as shown in Fig. 2: A standard feed-forward CNN and a deep residual variation of it. The first network consists of standard convolution layers, pooling layers, and leaky rectified linear units followed by a fully-connected layer. Motivated by a recent study [40], we also created a deep residual variation of the first network and report the performance of this variation in Section 5.2. According to the study, adding residual connections between standard convolu-

tion layers effectively improves and speeds up the optimization process for very deep networks. The deep residual variation of the encoder network has identity mapping connections after every 2 convolution layers except for the 4th pair. To match the number of channels after convolutions, we use a 1×1 convolution for residual connections. The encoder output is then flattened and passed to a fully connected layer which compresses the output into a 1024 dimensional feature vector.

3.2 Recurrence: 3D Convolutional LSTM



(a) inputs for each LSTM unit (b) 3D Convolutional LSTMs (c) 3D Convolutional GRUs

Fig. 3. (a) At each time step, each unit (purple) in the 3D-LSTM receives the same feature vector from the encoder as well as the hidden states from its neighbors (red) by a $3 \times 3 \times 3$ convolution ($W_s * h_{t-1}$) as inputs. We propose two versions of 3D-LSTMs: (b) 3D-LSTMs without output gates and (c) 3D Gated Recurrent Units (GRUs).

The core part of our 3D-R2N2 is a recurrence module that allows the network to retain what it has seen and to update the memory when it sees a new image. A naive approach would be to use a vanilla LSTM network. However, predicting such a large output space ($32 \times 32 \times 32$) would be a very difficult task without any regularization. We propose a new architecture that we call 3D-Convolutional LSTM (3D-LSTM). The network is made up of a set of structured LSTM units with restricted connections. The 3D-LSTM units are spatially distributed in a 3D grid structure, with each unit responsible for reconstructing a particular part of the final output (see Fig. 3(a)). Inside the 3D grid, there are $N \times N \times N$ 3D-LSTM units where N is the spatial resolution of the 3D-LSTM grid. Each 3D-LSTM unit, indexed (i, j, k), has an independent hidden state $h_{t,(i,j,k)} \in \mathbb{R}^{N_h}$. Following the same notation as in Section 2 but with f_t, i_t, s_t, h_t as 4D tensors ($N \times N \times N$ vectors of size N_h), the equations governing the 3D-LSTM grid are

$$f_t = \sigma(W_f \mathcal{T}(x_t) + U_f * h_{t-1} + b_f) \tag{9}$$

$$i_t = \sigma(W_i \mathcal{T}(x_t) + U_i * h_{t-1} + b_i) \tag{10}$$

$$s_t = f_t \odot s_{t-1} + i_t \odot \tanh(W_s \mathcal{T}(x_t) + U_s * h_{t-1} + b_s) \tag{11}$$

$$h_t = \tanh(s_t) \tag{12}$$

We denote the convolution operation as $*$. In our implementation, we use $N = 4$. Unlike a standard LSTM, we do not have output gates since we only extract the output at the end. By removing redundant output gates, we can reduce the number of parameters.

Intuitively, this configuration forces a 3D-LSTM unit to handle the mismatch between a particular region of the predicted reconstruction and the ground truth model such that each unit learns to reconstruct one part of the voxel space instead of contributing to the reconstruction of the entire space. This configuration also endows the network with a sense of locality so that it can selectively update its prediction about the previously occluded part of the object. We visualize such behavior in the appendix.

Moreover, a 3D Convolutional LSTM unit restricts the connections of its hidden state to its spatial neighbors. For vanilla LSTMs, all elements in the hidden layer h_{t-1} affect the current hidden state h_t, whereas a spatially structured 3D Convolutional LSTM only allows its hidden states $h_{t,(i,j,k)}$ to be affected by its neighboring 3D-LSTM units for all $i, j,$ and k. More specifically, the neighboring connections are defined by the convolution kernel size. For instance, if we use a $3 \times 3 \times 3$ kernel, an LSTM unit is only affected by its immediate neighbors. This way, the units can share weights and the network can be further regularized.

In Section 2, we also described the Gated Recurrent Unit (GRU) as a variation of the LSTM unit. We created a variation of the 3D-Convolutional LSTM using Gated Recurrent Unit (GRU). More formally, a GRU-based recurrence module can be expressed as

$$u_t = \sigma(W_{fx}\mathcal{T}(x_t) + U_f * h_{t-1} + b_f) \tag{13}$$

$$r_t = \sigma(W_{ix}\mathcal{T}(x_t) + U_i * h_{t-1} + b_i) \tag{14}$$

$$h_t = (1 - u_t) \odot h_{t-1} + u_t \odot \tanh(W_h\mathcal{T}(x_t) + U_h * (r_t \odot h_{t-1}) + b_h) \tag{15}$$

3.3 Decoder: 3D Deconvolutional Neural Network

After receiving an input image sequence $x_1, x_2, \cdots, x_T$, the 3D-LSTM passes the hidden state h_T to a decoder, which increases the hidden state resolution by applying 3D convolutions, non-linearities, and 3D unpooling [41] until it reaches the target output resolution.

As with the encoders, we propose a simple decoder network with 5 convolutions and a deep residual version with 4 residual connections followed by a final convolution. After the last layer where the activation reaches the target output resolution, we convert the final activation $\mathcal{V} \in \mathbb{R}^{N_{vox} \times N_{vox} \times N_{vox} \times 2}$ to the occupancy probability $p_{(i,j,k)}$ of the voxel cell at (i, j, k) using voxel-wise softmax.

3.4 Loss: 3D Voxel-wise Softmax

The loss function of the network is defined as the sum of voxel-wise cross-entropy. Let the final output at each voxel (i, j, k) be Bernoulli distributions $[1 - p_{(i,j,k)}, p_{(i,j,k)}]$, where the dependency on input $\mathcal{X} = \{x_t\}_{t \in \{1,\ldots,T\}}$ is omitted, and let the corresponding ground truth occupancy be $y_{(i,j,k)} \in \{0, 1\}$, then

$$L(\mathcal{X}, y) = \sum_{i,j,k} y_{(i,j,k)} \log(p_{(i,j,k)}) + (1 - y_{(i,j,k)}) \log(1 - p_{(i,j,k)}) \tag{16}$$

4 Implementation

Data augmentation: In training, we used 3D CAD models for generating input images and ground truth voxel occupancy maps. We first rendered the CAD models with a transparent background and then augmented the input images with random crops from the PASCAL VOC 2012 dataset [42]. Also, we tinted the color of the models and randomly translated the images. Note that all viewpoints were sampled randomly.
Training: In training the network, we used variable length inputs ranging from one image to an arbitrary number of images. More specifically, the input length (number of views) for each training example within a single mini-batch was kept constant, but the input length of training examples across *different* mini-batches varied randomly. This enabled the network to perform both single- and multi-view reconstruction. During training, we computed the loss only at the end of an input sequence in order to save both computational power and memory. On the other hand, during test time we could access the intermediate reconstructions at each time step by extracting the hidden states of the LSTM units.
Network: The input image size was set to 127×127. The output voxelized reconstruction was of size $32 \times 32 \times 32$. The networks used in the experiments were trained for $60,000$ iterations with a batch size of 36 except for [Res3D-GRU-3] (See Table 1), which needed a batch size of 24 to fit in an NVIDIA Titan X GPU. For the LeakyReLU layers, the slope of the leak was set to 0.1 throughout the network. For deconvolution, we followed the unpooling scheme presented in [41]. We used Theano [43] to implement our network and used Adam [44] for the SGD update rule.

5 Experiments

In this section, we validate and demonstrate the capability of our approach with several experiments using the datasets described in Section 5.1. First, we show the results of different variations of the 3D-R2N2 (Section 5.2). Next, we compare the performance of our network on the PASCAL 3D [45] dataset with that of a state-of-the-art method by Kar et al. [32] for single-view real-world image reconstruction (Section 5.3). Then we show the network's ability to perform multi-view reconstruction on the ShapeNet dataset [1] and the Online Products dataset [46] (Section 5.4, Section 5.5). Finally, we compare our approach with a Multi View Stereo method on reconstructing objects with various texture levels and viewpoint sparsity (Section 5.6).

5.1 Dataset

ShapeNet: The ShapeNet dataset is a collection of 3D CAD models that are organized according to the WordNet hierarchy. We used a subset of the ShapeNet dataset which consists of 50,000 models and 13 major categories (see Table 5(c) for a complete list). We split the dataset into training and testing sets, with 4/5

for training and the remaining 1/5 for testing. We refer to these two datasets as the ShapeNet training set and testing set throughout the experiments section.
PASCAL 3D: The PASCAL 3D dataset is composed of PASCAL 2012 detection images augmented with 3D CAD model alignment [45].
Online Products: The dataset [46] contains images of 23,000 items sold online. MVS and SFM methods fail on these images due to ultra-wide baselines. Since the dataset does not have the ground-truth 3D CAD models, we only used the dataset for qualitative evaluation.
MVS CAD Models: To compare our method with a Multi View Stereo method [47], we collected 4 different categories of high-quality CAD models. All CAD models have texture-rich surfaces and were placed on top of a texture-rich paper to aid the camera localization of the MVS method.

Metrics: We used two metrics in evaluating the reconstruction quality. The primary metric was the voxel Intersection-over-Union (IoU) between a 3D voxel reconstruction and its ground truth voxelized model. More formally,

$$IoU = \sum_{i,j,k} \left[I(p_{(i,j,k)} > t) I(y_{(i,j,k)}) \right] / \sum_{i,j,k} \left[I\left(I(p_{(i,j,k)} > t) + I(y_{(i,j,k)}) \right) \right] \quad (17)$$

where variables are defined in Section 3.4. $I(\cdot)$ is an indicator function and t is a voxelization threshold. Higher IoU values indicate better reconstructions. We also report the cross-entropy loss (Section 3.4) as a secondary metric. Lower loss values indicate higher confidence reconstructions.

5.2 Network Structures Comparison

We tested 5 variations of our 3D-R2N2 as described in Section 3. The first four networks are based on the standard feed-forward CNN (top Fig. 2) and the fifth network is the residual network (bottom Fig. 2). For the first four networks, we used either GRU or LSTM units and and varied the convolution kernel to be either $1 \times 1 \times 1$ [3D-LSTM/GRU-3] or $3 \times 3 \times 3$ [3D-LSTM/GRU-3]. The residual network used GRU units and $3 \times 3 \times 3$ convolutions [Res3D-GRU-3]. These networks were trained on the ShapeNet training set and tested on the ShapeNet testing set. We used 5 views in the experiment. Table 1 shows the results. We observe that 1) the GRU-based networks outperform the LSTM-based networks, 2) that the networks with neighboring recurrent unit connections ($3 \times 3 \times 3$ convolutions) outperform the networks that have no neighboring recurrent unit connection ($1 \times 1 \times 1$ convolutions), and 3) that the deep residual network variation further boosts the reconstruction performance.

Table 1. Reconstruction performance of 3D-LSTM variations according to cross-entropy loss and IoU using 5 views.

	Encoder	Recurrence	Decoder	Loss	IoU
3D-LSTM-1	simple	LSTM	simple	0.116	0.499
3D-GRU-1	simple	GRU	simple	0.105	0.540
3D-LSTM-3	simple	LSTM	simple	0.106	0.539
3D-GRU-3	simple	GRU	simple	0.091	0.592
Res3D-GRU-3	residual	GRU	residual	**0.080**	**0.634**

5.3 Single Real-World Image Reconstruction

We evaluated the performance of our network in single-view reconstruction using real-world images, comparing the performance with that of a recent method by Kar et al. [32]. To make a quantitative comparison, we used images from the PASCAL VOC 2012 dataset [42] and its corresponding 3D models from the PASCAL 3D+ dataset [45]. We ran the experiments with the same configuration as Kar et al. except that we allow the Kar et al. method to have ground-truth object segmentation masks and keypoint labels as additional inputs for both training and testing.

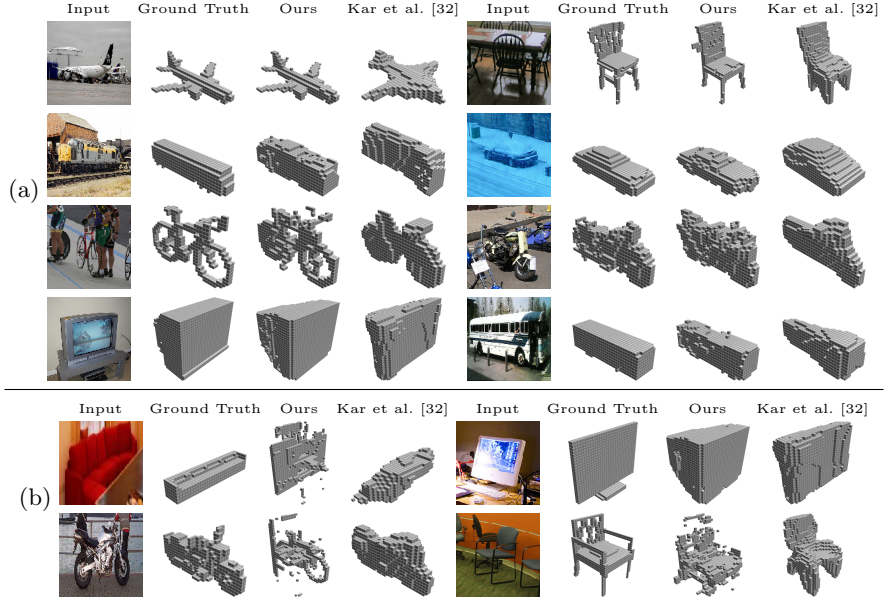


Fig. 4. (a) Reconstruction samples of PASCAL VOC dataset. (b) Failed reconstructions on the PASCAL VOC dataset. Note that Kar et al. [32] is trained/tested per category and takes ground-truth object segmentation masks and keypoint labels as additional input.

Training. We fine-tuned a network trained on the ShapeNet dataset with PASCAL 3D+. We used the PASCAL 3D+ validation set to find hyperparameters such as the number of fine-tuning iterations and the voxelization threshold.

Results. As shown in Table 2, our approach outperforms the method of Kar et al. [32] in every category. However, we observe that our network has some difficulties reconstructing thin legs of chairs. Moreover, the network often confuses thin flat panels with thick CRT screens when given a frontal view of the monitor. Yet, our approach demonstrates a competitive quantitative performance. For the qualitative results and comparisons, please see Fig. 4.

Aside from better performance, our network has several advantages over Kar et al. [32]. First, we do not need to train and test per-category. Our network

trains and reconstructs without knowing the object category. Second, our network does not require object segmentation masks and keypoint labels as additional inputs. Kar et al. does demonstrate the possibility of testing on a wild unlabeled image by estimating the segmentation and keypoints. However, our network outperforms their method tested with ground truth labels.

Table 2. Per-category reconstruction of PASCAL VOC compared using voxel Intersection-over-Union (IoU). Note that the experiments were ran with the same configuration except that the method of Kar et al. [32] took ground-truth object segmentation masks and keypoint labels as additional inputs for both training and testing.

	aero	**bike**	**boat**	**bus**	**car**	**chair**	**mbike**	**sofa**	**train**	**tv**	**mean**
Kar et al. [32]	0.298	0.144	0.188	0.501	0.472	0.234	0.361	0.149	0.249	0.492	0.318
ours [LSTM-1]	0.472	0.330	0.466	0.677	0.579	0.203	0.474	0.251	0.518	0.438	0.456
ours [Res3D-GRU-3]	**0.544**	**0.499**	**0.560**	**0.816**	**0.699**	**0.280**	**0.649**	**0.332**	**0.672**	**0.574**	**0.571**

5.4 Multi-view Reconstruction Evaluation

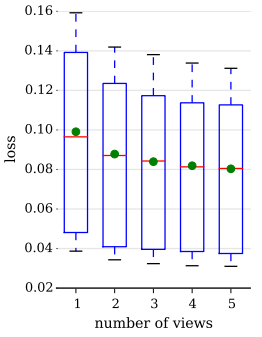
In this section, we report a quantitative evaluation of our network's performance in multi-view reconstruction on the ShapeNet testing set.

Experiment setup. We used the [Res3D-GRU-3] network in this experiment. We evaluated the network with the ShapeNet testing set. The testing set consisted of 8725 models in 13 major categories. We rendered five random views for each model, and we applied a uniform colored background to the image. We report both softmax loss and intersection over union(IoU) with a voxelization threshold of 0.4 between the predicted and the ground truth voxel models.

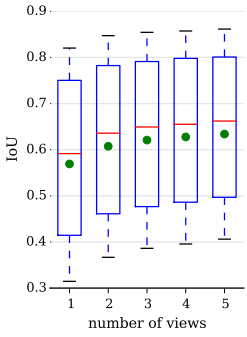
Overall results. We first investigate the quality of the reconstructed models under different numbers of views. Fig. 5(a) and (b) show that reconstruction quality improves as the number of views increases. The fact that the marginal gain decreases accords with our assumption that each additional view provides less information since two random views are very likely to have partial overlap.

Per-category results. We also report the reconstruction IoUs on each of the 13 major categories in the testing set in Table 5. We observed that the reconstruction quality improved for every category as the number of views increased, but the quality varied depending on the category. Cabinets, cars, and speakers had the highest reconstruction performance since the objects are bulky-shaped and have less (shape) variance compared to other classes. The network performed worse on the lamp, bench, and table categories. These classes have much higher shape variation than the other classes. For example, a lamp can have a slim arm or a large lampshade which may move around, and chairs and tables have various types of supporting structures.

Qualitative results. Fig. 6(a) shows some sample reconstructions from the ShapeNet testing set. One exemplary instance is the truck shown in row 2. In the initial view, only the front part of the truck is visible. The network took the safest guess that the object is a sedan, which is the most common shape in the car category. Then the network produced a more accurate reconstruction of the truck after seeing more views. All other instances show similar improvements as the network sees more views of the objects. Fig. 6(b) shows two failure cases.



(a) Cross entropy loss



(b) Voxel IoU

# views	1	2	3	4	5
plane	0.513	0.536	0.549	0.556	0.561
bench	0.421	0.484	0.502	0.516	0.527
cabinet	0.716	0.746	0.763	0.767	0.772
car	0.798	0.821	0.829	0.833	0.836
chair	0.466	0.515	0.533	0.541	0.550
monitor	0.468	0.527	0.545	0.558	0.565
lamp	0.381	0.406	0.415	0.416	0.421
speaker	0.662	0.696	0.708	0.714	0.717
firearm	0.544	0.582	0.593	0.595	0.600
couch	0.628	0.677	0.690	0.698	0.706
table	0.513	0.550	0.564	0.573	0.580
cellphone	0.661	0.717	0.732	0.738	0.754
watercraft	0.513	0.576	0.596	0.604	0.610

(c) Per-category IoU

Fig. 5. (a), (b): Multi-view reconstruction using our model on the ShapeNet dataset. The performance is reported in median (red line) and mean (green dot) cross-entropy loss and intersection over union (IoU) values. The box plot shows 25% and 75%, with caps showing 15% and 85%. (c): Per-category reconstruction of the ShapeNet dataset using our model. The values are average IoU.

5.5 Reconstructing Real World Images

In this experiment, we tested our network on the Online Products dataset for qualitative evaluation. Images that were not square-shaped were padded with white pixels.

Fig. 6(c) shows some sample reconstructions. The result shows that the network is capable of reconstructing real world objects using only synthetic data as training samples. It also demonstrates that the network improves the reconstructions after seeing additional views of the objects. One exemplary instance is the reconstruction of couch as shown in row 1. The initial side view of the couch led the network to believe that it was a one-seater sofa, but after seeing the front of the couch, the network immediately refined its reconstruction to reflect the observation. Similar behaviors are also shown in other samples. Some failure cases are shown in Fig.6(d).

5.6 Multi View Stereo(MVS) vs. 3D-R2N2

In this experiment, we compare our approach with a MVS method on reconstructing objects that are of various texture levels with different number of views. MVS methods are limited by the accuracy of feature correspondences across different views. Therefore, they tend to fail reconstructing textureless objects or images from sparsely positioned camera viewpoints. In contrast, our method does not require accurate feature correspondences or adjacent camera viewpoints.

Dataset: We used high-quality CAD models of 4 different categories and augmented their texture strengths to low, medium and high by manually editing their textures. We then rendered the models from viewpoints with uniformly sampled azimuth angles. Please refer to Fig. 7 for some samples of rendered

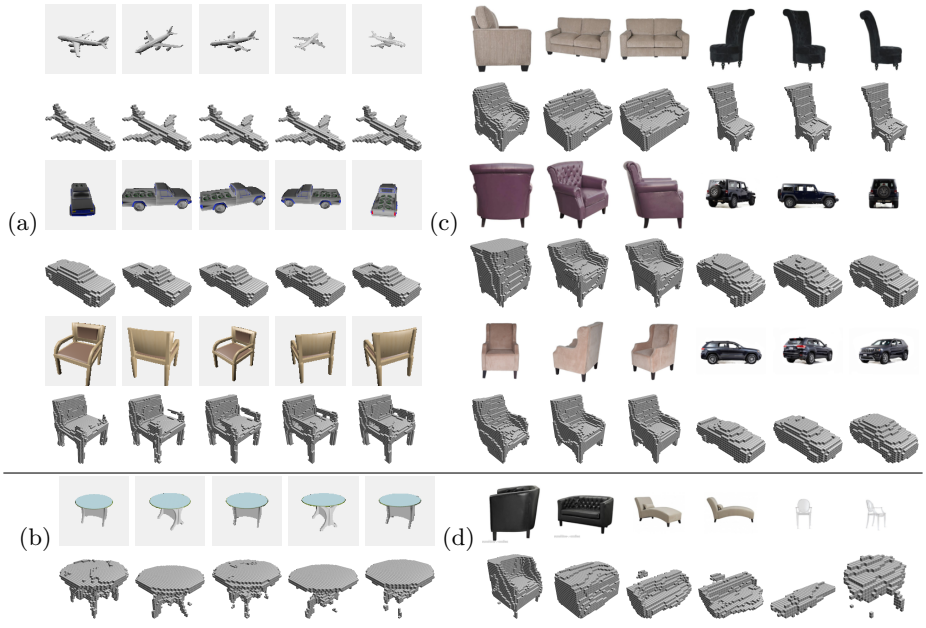


Fig. 6. Sample reconstructions on (a) the ShapeNet [1] testing set and (c) the Online Products dataset [46]. Top rows are input image sequences (from left to right). Bottom rows are the reconstructions at each time step. (b), (d): Failure cases on each dataset.

models across different viewpoints and texture strengths. For each texture level and number of views configuration, both the MVS method and our network took identical sets of images as inputs.

Experiment setup. We used a Patch-Match [48]-based off-the-shelf implementation [49] as the MVS method. The MVS method takes images along with their camera positions estimated by Global SFM [50] and outputs the reconstructed model. For our network, we used the [Res3D-GRU-3] network trained with at most 5 views. In order to cope with more views, we fine-tuned our network with samples that have a maximum of 24 views for 5000 iterations using the ShapeNet training set. We quantified the quality of the reconstructions using IoU of the voxels. The network was voxelized with the occupancy probability threshold set to 0.1. The mesh reconstructed from the MVS method was voxelized into a $32 \times 32 \times 32$ grid for comparison.

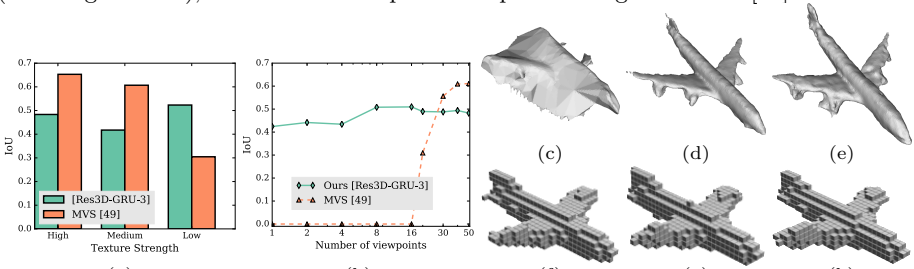
Results. The results are shown in Fig. 8 (a) and (b). We observed 1) that our model worked with as few as one view, whereas the MVS method failed completely when the number of views was less than 20 (IoU=0), and 2) that our model worked regardless of the objects' texture level, whereas the MVS method frequently failed to reconstruct objects that had low texture level even when a large number of views were provided. This shows that our method works in situations where MVS methods would perform poorly or completely fail. Note that the reconstruction performance of our method decreased after the number

of views passed 24. This is because we only fine-tuned our network on samples with a maximum of 24 views.

We also discovered some limitations of our method. First, our method could not reconstruct as many details as the MVS method did when given more than 30 different views of the model. Second, our method performed worse in reconstructing objects with high texture levels. This is largely because most models in the ShapeNet training set have low texture level.

(a) (b) (c) (d) (e)

Fig. 7. Rendered images with various viewpoints (a,b,c) and texture levels (c, d, e) (from high to low), used for the comparison experiment against MVS [49].



(a) (b) (c) (d) (e) (f) (g) (h)

Fig. 8. Reconstruction performance of MVS [49] compared with that of our network. (a) shows how texture strengths affect the reconstructions of MVS and our network, averaged over 20, 30, 40, and 50 input views of all classes. (b) compares the quality of the reconstruction across the number of input images, averaged over all texture levels of all classes. (c-e) show the reconstruction result of MVS and (f-h) show the reconstruction results from our method [Res3D-GRU-3] on a high-texture airplane model with 20, 30, and 40 input views respectively.

6 Conclusion

In this work, we proposed a novel architecture that unifies single- and multi-view 3D reconstruction into a single framework. Even though our network can take variable length inputs, we demonstrated that it outperforms the method of Kar et al. [32] in single-view reconstruction using real-world images. We further tested the network's ability to perform multi-view reconstruction on the ShapeNet dataset [1] and the Online Products dataset [46], which showed that the network is able to incrementally improve its reconstructions as it sees more views of an object. Lastly, we analyzed the network's performance on multi-view reconstruction, finding that our method can produce accurate reconstructions when techniques such as MVS fail. In summary, our network does not require a minimum number of input images in order to produce a plausible reconstruction and is able to overcome past challenges of dealing with images which have insufficient texture or wide baseline viewpoints.

7 Acknowledgements

We acknowledge the support of NSF CAREER grant N.1054127 and Toyota Award #122282. We also thank the Korea Foundation for Advanced Studies and NSF GRFP for their support.

References

1. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., Yu, F.: ShapeNet: An Information-Rich 3D Model Repository. Technical report, Stanford University — Princeton University — Toyota Technological Institute at Chicago (2015)
2. Choi, S., Zhou, Q.Y., Miller, S., Koltun, V.: A large dataset of object scans. arXiv preprint arXiv:1602.02481 (2016)
3. Fitzgibbon, A., Zisserman, A.: Automatic 3d model acquisition and generation of new images from video sequences. In: Signal Processing Conference (EUSIPCO 1998), 9th European, IEEE (1998) 1–8
4. Lhuillier, M., Quan, L.: A quasi-dense approach to surface reconstruction from uncalibrated images. Pattern Analysis and Machine Intelligence, IEEE Transactions on **27**(3) (2005) 418–433
5. Agarwal, S., Snavely, N., Simon, I., Seitz, S.M., Szeliski, R.: Building rome in a day. In: Computer Vision, 2009 IEEE 12th International Conference on, IEEE (2009) 72–79
6. Engel, J., Schöps, T., Cremers, D.: Lsd-slam: Large-scale direct monocular slam. In: Computer Vision–ECCV 2014. Springer (2014) 834–849
7. Häming, K., Peters, G.: The structure-from-motion reconstruction pipeline–a survey with focus on short image sequences. Kybernetika **46**(5) (2010) 926–937
8. Fuentes-Pacheco, J., Ruiz-Ascencio, J., Rendón-Mancha, J.M.: Visual simultaneous localization and mapping: a survey. Artificial Intelligence Review **43**(1) (2015) 55–81
9. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International journal of computer vision **60**(2) (2004) 91–110
10. Bhat, D.N., Nayar, S.K.: Ordinal measures for image correspondence. Pattern Analysis and Machine Intelligence, IEEE Transactions on **20**(4) (1998) 415–423
11. Saponaro, P., Sorensen, S., Rhein, S., Mahoney, A.R., Kambhamettu, C.: Reconstruction of textureless regions using structure from motion and image-based interpolation. In: Image Processing (ICIP), 2014 IEEE International Conference on, IEEE (2014) 1847–1851
12. Seitz, S.M., Dyer, C.R.: Photorealistic scene reconstruction by voxel coloring. International Journal of Computer Vision **35**(2) (1999) 151–173
13. Kutulakos, K.N., Seitz, S.M.: A theory of shape by space carving. International Journal of Computer Vision **38**(3) (2000) 199–218
14. Gregory G Slabaugh, W Bruce Culbertson, T.M., Stevens, M.R., Schafer, R.W.: Methods for volumetric reconstruction of visual scenes. International Journal of Computer Vision **57**(3) (2004) 179–199
15. Anwar, Z., Ferrie, F.: Towards robust voxel-coloring: Handling camera calibration errors and partial emptiness of surface voxels. In: Proceedings of the 18th International Conference on Pattern Recognition - Volume 01. ICPR '06, Washington, DC, USA, IEEE Computer Society (2006) 98–102

16. Broadhurst, A., Drummond, T.W., Cipolla, R.: A probabilistic framework for space carving. In: Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on. Volume 1., IEEE (2001) 388–393
17. Dame, A., Prisacariu, V.A., Ren, C.Y., Reid, I.: Dense reconstruction using 3d object shape priors. In: Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on. (2013)
18. Bao, Y., chandraker, M., Lin, Y., Savarese, S.: Dense object reconstruction using semantic priors. In: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition. (2013)
19. Lawrence, G.R.: Machine perception of three-dimensional solids. Ph. D. Thesis (1963)
20. Nevatia, R., Binford, T.O.: Description and recognition of curved objects. Artificial Intelligence **8**(1) (1977) 77–98
21. Zia, M.Z., Stark, M., Schiele, B., Schindler, K.: Detailed 3d representations for object modeling and recognition, TPAMI (2013)
22. Rock, J., Gupta, T., Thorsen, J., Gwak, J., Shin, D., Hoiem, D.: Completing 3d object shape from one depth image. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2015) 2484–2493
23. Bongsoo Choy, C., Stark, M., Corbett-Davies, S., Savarese, S.: Enriching object detection with 2d-3d registration and continuous viewpoint estimation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (June 2015)
24. Blanz, V., Vetter, T.: Face recognition based on fitting a 3d morphable model. Pattern Analysis and Machine Intelligence, IEEE Transactions on **25**(9) (2003) 1063–1074
25. Matthews, I., Xiao, J., Baker, S.: 2d vs. 3d deformable face models: Representational power, construction, and real-time fitting. International journal of computer vision **75**(1) (2007) 93–113
26. Kemelmacher-Shlizerman, I., Basri, R.: 3d face reconstruction from a single image using a single reference face shape. Pattern Analysis and Machine Intelligence, IEEE Transactions on **33**(2) (2011) 394–405
27. Prisacariu, V.A., Segal, A.V., Reid, I.: Simultaneous monocular 2d segmentation, 3d pose recovery and 3d reconstruction. In: Computer Vision–ACCV 2012. Springer (2012) 593–606
28. Sandhu, R., Dambreville, S., Yezzi, A., Tannenbaum, A.: A nonrigid kernel-based framework for 2d-3d pose estimation and 2d image segmentation. Pattern Analysis and Machine Intelligence, IEEE Transactions on **33**(6) (2011) 1098–1115
29. Saxena, A., Sun, M., Ng, A.Y.: Make3d: Learning 3d scene structure from a single still image. IEEE Trans. Pattern Anal. Mach. Intell. **31**(5) (may 2009) 824–840
30. Hoiem, D., Efros, A.A., Hebert, M.: Automatic photo pop-up. ACM transactions on graphics (TOG) **24**(3) (2005) 577–584
31. Vicente, S., Carreira, J., Agapito, L., Batista, J.: Reconstructing pascal voc. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2014)
32. Kar, A., Tulsiani, S., Carreira, J., Malik, J.: Category-specific object reconstruction from a single image. In: Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on, IEEE (2015) 1966–1974
33. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8) (November 1997) 1735–1780
34. Sundermeyer, M., Schlüter, R., Ney, H.: Lstm neural networks for language modeling. In: INTERSPEECH. (2012) 194–197

35. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Advances in neural information processing systems. (2014) 3104–3112
36. Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. In: Advances in Neural Information Processing Systems 27. (2014)
37. Liu, F., Shen, C., Lin, G.: Deep convolutional neural fields for depth estimation from a single image. In: Proc. IEEE Conf. Computer Vision and Pattern Recognition. (2015)
38. Bengio, Y., Simard, P., Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks **5**(2) (Mar 1994) 157–166
39. Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. ArXiv e-prints (2014)
40. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. ArXiv e-prints (December 2015)
41. A.Dosovitskiy, J.T.Springenberg, T.Brox: Learning to generate chairs with convolutional neural networks. In: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR). (2015)
42. Everingham, M., Van Gool, L., Williams, C., Winn, J., Zisserman, A.: The pascal visual object classes challenge 2012 (2011)
43. Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., Warde-Farley, D., Bengio, Y.: Theano: a CPU and GPU math expression compiler. In: Proceedings of the Python for Scientific Computing Conference (SciPy). (June 2010)
44. Kingma, D., Ba, J.: Adam: A Method for Stochastic Optimization. ArXiv e-prints (2014)
45. Xiang, Y., Mottaghi, R., Savarese, S.: Beyond pascal: A benchmark for 3d object detection in the wild. In: Applications of Computer Vision (WACV), 2014 IEEE Winter Conference on, IEEE (2014) 75–82
46. Song, H.O., Xiang, Y., Jegelka, S., Savarese, S.: Deep metric learning via lifted structured feature embedding. ArXiv e-prints (2015)
47. : Cg studio (2016) [Online; accessed 14-March-2016].
48. Barnes, C., Shechtman, E., Finkelstein, A., Goldman, D.: Patchmatch: A randomized correspondence algorithm for structural image editing. ACM Transactions on Graphics-TOG **28**(3) (2009) 24
49. : Openmvs: open multi-view stereo reconstruction library (2015) [Online; accessed 14-March-2016].
50. Moulon, P., Monasse, P., Marlet, R.: Global fusion of relative motions for robust, accurate and scalable structure from motion. In: Proceedings of the IEEE International Conference on Computer Vision. (2013) 3248–3255