DA-RNN: Semantic Mapping with Data Associated Recurrent Neural Networks

Yu Xiang and Dieter Fox
Paul G. Allen School of Computer Science & Engineering
University of Washington
{yuxiang, fox}@cs.washington.edu

***Abstract*—3D scene understanding is important for robots to interact with the 3D world in a meaningful way. Most previous works on 3D scene understanding focus on recognizing geometrical or semantic properties of a scene independently. In this work, we introduce Data Associated Recurrent Neural Networks (DA-RNNs), a novel framework for joint 3D scene mapping and semantic labeling. DA-RNNs use a new recurrent neural network architecture for semantic labeling on RGB-D videos. The output of the network is integrated with mapping techniques such as KinectFusion in order to inject semantic information into the reconstructed 3D scene. Experiments conducted on real world and synthetic RGB-D videos demonstrate the superior performance of our method.**

I. Introduction

For many tasks, robots need to understand the 3D structure and semantics of their environment. For example, recognizing the free space and surfaces in a scene helps motion planning in robot navigation and manipulation tasks. Semantic understanding, beyond pure geometry, enables a robot to reason about objects, which is particularly important for manipulation and human robot interaction tasks. Over the last years, various techniques have been proposed for dense 3D scene reconstruction using depth cameras, including RGBD-Mapping, KinectFusion, Kintinuous, and ElasticFusion [12, 21, 33, 34]. These methods jointly reconstruct the 3D scene and track the camera position from RGB-D videos. However, they do not provide semantic information about the scene. In parallel, different approaches for recognizing scene semantics have been proposed. These include methods in object detection [9, 10], object pose estimation [3, 25, 35], and semantic labeling [23, 19]. Most of these methods focus on detecting specific objects or on recognizing scene elements in individual 2D images.

The goal of our work is to use RGB-D videos to reconstruct and label every observed surface element in a 3D scene, providing dense information about small objects, such as bowls and mugs, and larger objects such as tables and chairs. In such a setting, the key question is how the information from the RGB-D frames can be combined to improve recognition accuracy. Recent approaches handle this by incorporating recognition results from individual RGB-D frames into a 3D model, possibly followed by additional reasoning over the 3D structure [17, 18, 20]. However, in these approaches, the reasoning about individual frames and their information accumulation is only loosely coupled.

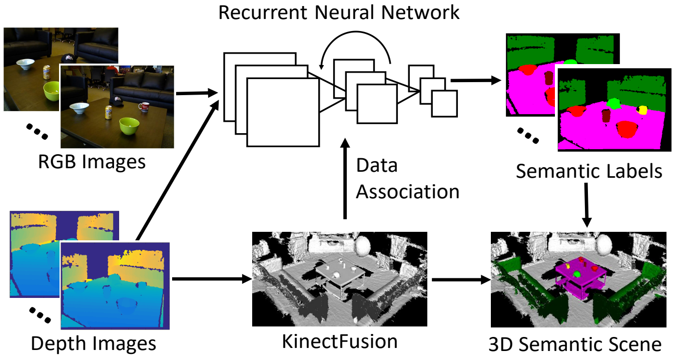

Fig. 1. Overview of the DA-RNN framework. RGB-D frames are fed into a Recurrent Neural Network. KinectFusion provides the 3D reconstruction and the data associations necessary to connect recurrent units between RGB-D frames. The pixel labels provided by the RNN are integrated into the 3D semantic map. The overall labeling and reconstruction process runs at 5fps.

In this work, we introduce DA-RNNs, a deep network architecture that tightly connects the analysis of individual RGB-D frames and their integration over time. To do so, we take advantage of *Recurrent* Neural Networks (RNNs), where recurrent units connect information over time. A naive approach for achieving a strong connection between the mapping and the labeling process would be to establish a fixed network structure in 3D and treat each surface element in a KinectFusion or ElasticFusion map as a recurrent unit in the RNN. Unfortunately, such an approach is not feasible since it would quickly exhaust the memory available even on large-scale GPUs. To overcome this problem, our approach performs recurrent reasoning only over those parts of the map that are currently observed by the RGB-D camera. Specifically, we introduce a new recurrent unit inside our RNN called Data Associated Recurrent Unit (DA-RU). Each DA-RU corresponds to a pixel in the input image. The hidden state of the DA-RU accumulates information about that pixel in time. Crucially, the temporal connectivity between the DA-RU states of consecutive frames is not fixed, but depends on the data association provided by the mapping process. As a result, each DA-RU incorporates the hidden state from the associated DA-RU in the previous frame, allowing information to flow in a spatially consistent way.

In order to semantically reconstruct the 3D scene, we integrate the outputs of our DA-RNN into the 3D voxels of KinectFusion, which provides a consistent semantic labeling of the 3D scene (similar to [18, 20]). Fig. 1 illustrates an overview of our framework.

We have conducted extensive experiments to test our framework on the RGB-D Scene dataset [18] and a synthetic dataset we generated with 3D shapes from the ShapeNet repository [4]. The experimental results demonstrate that DA-RNNs are able to provide superior semantically labeled 3D scenes from RGB-D videos. Our code and data are available at https://rse-lab.cs.washington.edu/projects/darnn/.

In summary, our work has the following key contributions:

- We propose a novel recurrent neural network for semantic labeling on RGB-D videos with a new data associated recurrent unit to capture dependencies across video frames.
- We introduce a novel updating rule for DA-RU's to perform weighted moving averaging of the hidden state.
- We integrate DA-RNN's with KinectFusion for semantic 3D scene reconstruction.
- We contribute pixel-wise semantic labels on the RGB-D Scene dataset [18] and a new synthetic dataset which can benefit future research on 3D semantic mapping.

This paper is organized as follows. After discussing related work, we introduce DA-RNNs, followed by experimental results and a conclusion.

II. Related Work

Our work is mostly related to 3D mapping and semantic labeling methods in the literature.

A. Dense 3D Scene Reconstruction

3D reconstruction techniques can be roughly classified into point-based methods, voxel-based methods and surfel-based methods. Point-based methods use 3D points to represent 3D scenes [29, 7, 12]. Voxel-based methods such as KinectFusion, PatchVolumes, or Kintinuous [21, 13, 33] employ a volumetric representation of the 3D space, which reconstruct dense 3D surfaces of the scene. Surfel-based methods [15, 12, 34] make a trade-off between 3D points and voxels, where the 3D scene is represented compactly by 3D disks, i.e., surfels.

In principle, our DA-RNN framework only requires dense data associations between consecutive frames. It is thus independent of the underlying representation and could be combined with any of the reconstruction techniques described above. Here, we use KinectFusion [21] to achieve a volumetric representation for geometry and semantics.

B. Semantic Labeling

Semantic labeling on images classifies each pixel of an input image into one of the predefined semantic classes. The semantic labeling problem has often been tackled with probabilistic graphical models such as Markov Random Fields (MRFs) or Conditional Random Fields (CRFs) [27, 16], which model the context around pixels. More recently, convolutional neural networks have been applied to semantic labeling [19, 36, 2, 5], which achieve significant improvement over previous methods. However, all these approaches mainly focus on semantic labeling of a single image. Recurrent neural networks [22, 26] have been applied to semantic video segmentation, which exploit the temporal relationship or information provided by multiple viewpoints of a scene. [17, 20] show how the labels extracted from individual RGB-D frames can be incorporated into a voxel or surfel map, resulting in more stable labeling. Further improvements are achieved by performing MRF or CRF inference in the 3D map. Approaches such as [24, 18, 30] perform labeling by conducting 3D object detection through the 3D reconstruction, thereby potentially incorporating information that is not available in any single view.

Different from these works, we propose a recurrent neural network architecture that tightly integrates the information contained in multiple viewpoints of an RGB-D video stream. Both individual frame and across frame parameters are learned in a single network structure. In contrast to existing RNNs, DA-RNNs do not assume a fixed relationship between input images and network structure, but rely on data association to generate the connections between recurrent units on the fly. The recurrent layer we introduce in this work could also be used as a standalone layer and plugged into existing CNN-based methods for semantic video labeling.

III. Method

In this section, we present our framework for 3D semantic mapping using RGB-D videos. We first describe our design of the convolutional neural network for single frame semantic labeling. Then, we extend the single frame network to a recurrent neural network for semantic labeling on videos. Finally, we integrate the recurrent neural network with KinectFusion [21] in order to semantically reconstruct the 3D scene.

A. Single Frame Labeling with Fully Convolutional Networks

The basis of our semantic labeling framework is a Fully Convolutional Network (FCN) for single frame labeling. An influential network architecture for semantic labeling as been introduced by [19], which converts a network for image classification into fully convolutional by treating the fully connected layers in the network as 1×1 convolutional layers. In addition, [19] uses deconvolutional layers to increase the resolution of the network output. Inspired by [19], we design our network architecture for single frame labeling as illustrated in Fig. 2.

1) Single Stream Network: Our single stream network in Fig. 2 takes a single tensor as input, such as an RGB image or a depth image. It consists of 16 convolutional layers, 4 max pooling layers, 2 deconvolutional layers and 1 addition layer. All the convolutional filters in the network are of size 3×3 and stride 1. The max pooling layers are of size 2×2 and stride 2. Therefore, each max pooling layer reduces the resolution of its input by a factor of 2. The output of the 4th max pooling layer is 16 times smaller than the input image. The first deconvolutional layer doubles the resolution of its input, while the second deconvolutional layer increases the resolution by 8 times. As a result, the output of the network has the same resolution as the input image, i.e., dense pixel-wise labeling.

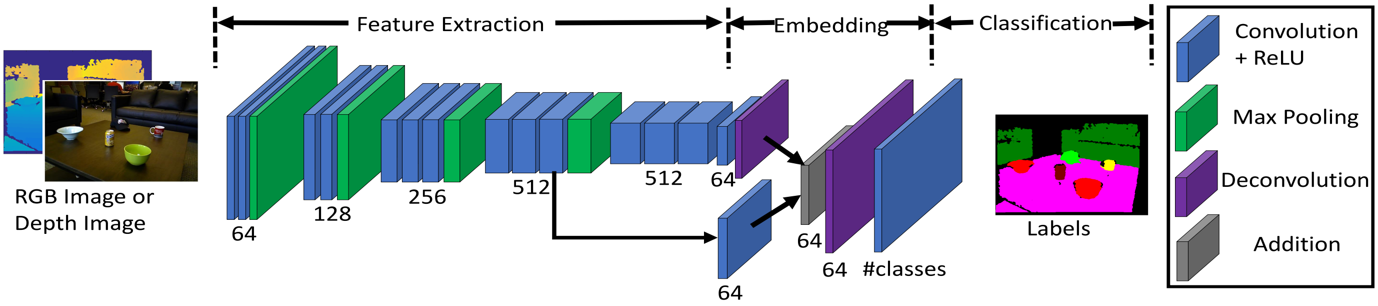

Fig. 2. Architecture of our single stream network for semantic labeling.

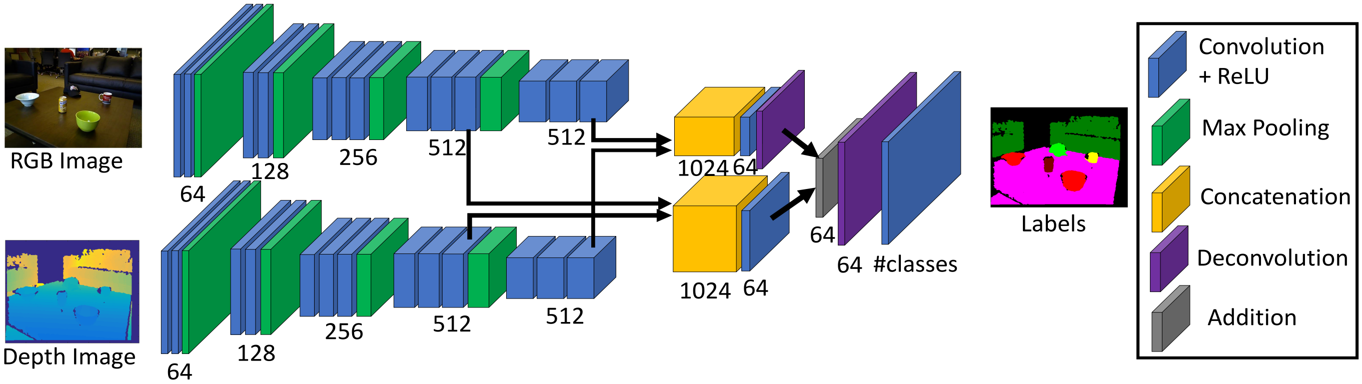

Fig. 3. Architecture of our double stream network for semantic labeling.

We design the network architecture with three phases as in Fig. 2. The first 13 convolutional layers and the 4 max pooling layers are considered to be the feature extraction phase, which extracts 512-dimensional feature vectors for the input image. The second phase is the embedding phase, which embeds the 512-dimensional features into a 64-dimensional space while increasing the resolution of the feature map using deconvolutional layers. A skip link is used in the embedding phase to combine features from an earlier convolutional layer motivated by [19] (i.e., the one before the 4th max pooling layer). The last phase of the network classifies each pixel into a semantic class using a convolutional layer. The output of this convolutional layer is treated as the labeling scores for pixels, which has n channels with n the number of the semantic classes. By applying a softmax layer on the labeling scores, we can obtain the class probabilities of the pixels.

2) Double Stream Network: When the input data is multi-modal such as color and depth, we have designed the double stream network to fuse RGB-D data (Fig. 3). In this network, the RGB image and the depth image are processed separately with different convolutional layers for feature extraction. These layers share the same structure as the feature extraction phase in the single stream network. To combine the two types of features, we introduce two concatenation layers, which stack the 512-dimensional features from the RGB image and the depth image and generate 1024-dimensional features. These features are embedded into a 64-dimensional space and classified as in the single stream network. Note that we utilize the "late fusion" strategy in this network, where features for color and depth are computed independently and then concatenated.

B. Video Semantic Labeling with DA-RNNs

In videos, due to the smooth change in camera motion or object motion, information flows across video frames. How to effectively utilize the temporal information for semantic labeling in videos is still an open question. In this work, we propose a Data Associated Recurrent Neural Network (DA-RNN) for video semantic labeling which stores and passes information across frames.

1) DA-RNN Architecture: The architecture of our DA-RNN for semantic labeling is illustrated in Fig. 4. Based on our double stream network for single frame labeling, we introduce a recurrent layer which takes the embedded features of the current frame as input and generates new features for classification. The recurrent layer is designed to combine features from the previous frames and features in the current frame in order to utilize information across frames.

Specifically, the recurrent layer contains one recurrent unit for each pixel location. These recurrent units maintain and update their hidden states, storing information from previous frames. The outputs of the recurrent units depend on their inputs and hidden states. Two widely used recurrent units are the Long Short-Term Memory (LSTM) unit [14] and the Gated Recurrent Unit (GRU) [6]. Both LSTM and GRU have been shown to perform well in tasks that require capturing long-term dependencies, such as natural language processing, speech recognition and machine translation [31, 11, 32]. However, both of them employ the hyperbolic tangent function in updating their hidden states, which makes the gradient back-propagation training inefficient. In DA-RNN, we introduce a new recurrent unit, which is explicitly designed to fuse features

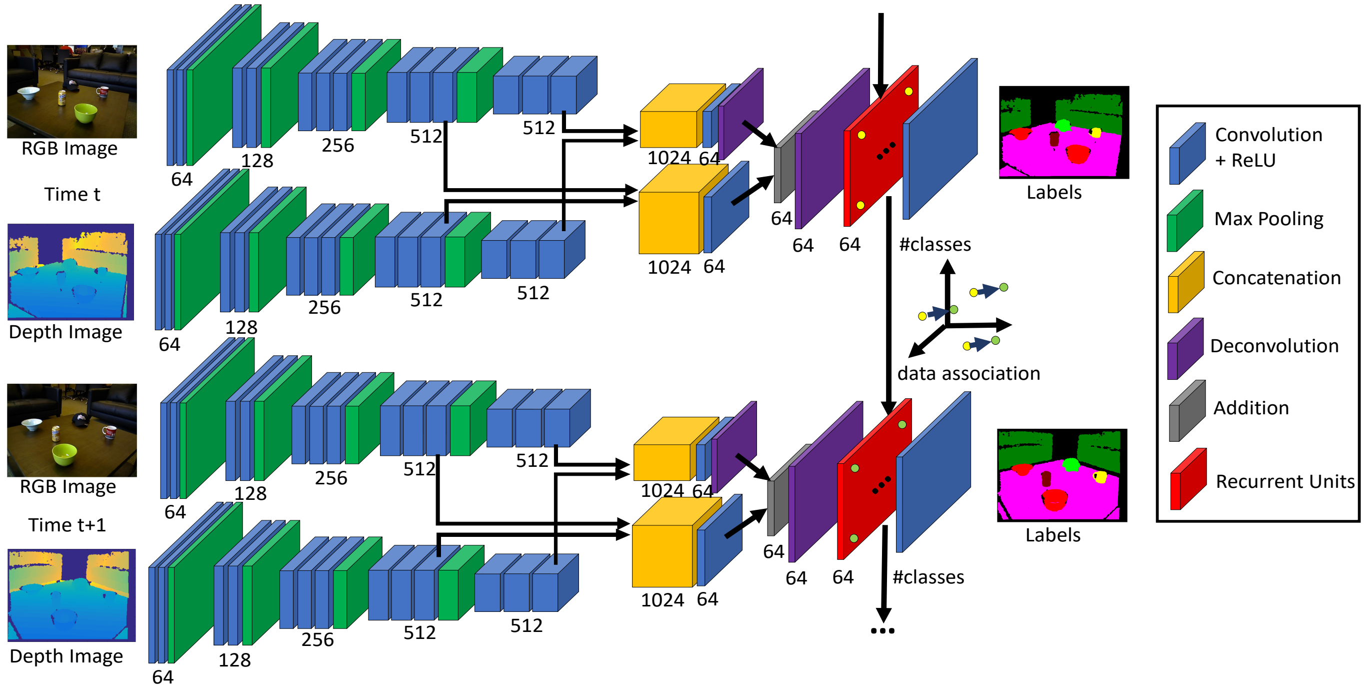

Fig. 4. Architecture of our data associated recurrent neural network for semantic labeling on videos.

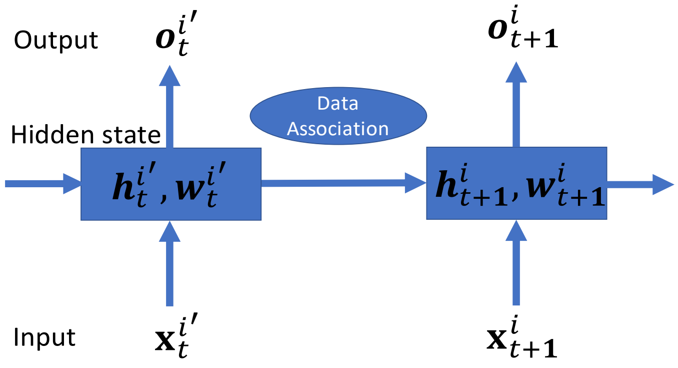
Output
$\boldsymbol{o}^{i'}_t$
$\boldsymbol{o}^{i}_{t+1}$
Hidden state
Data Association
$\boldsymbol{h}^{i'}_t, \boldsymbol{w}^{i'}_t$
$\boldsymbol{h}^{i}_{t+1}, \boldsymbol{w}^{i}_{t+1}$
Input
$\mathbf{x}^{i'}_t$
$\mathbf{x}^{i}_{t+1}$

Fig. 5. The block diagram of our Data Associated Recurrent Unit (DA-RU).

across video frames and can be trained more efficiently by using the Rectified Linear Unit (ReLU) as the activation function.

2) Data Associated Recurrent Unit (DA-RU): The recurrent layer in our RNN contains N recurrent units, where N is the number of pixels in the input image. At time t, the ith recurrent unit stores a pair of vectors $\left\langle \mathbf{h}^i_t, \mathbf{w}^i_t \right\rangle, i = 1, \ldots, N$, where $\mathbf{h}^i_t$ denotes the hidden state of the unit, and $\mathbf{w}^i_t$ indicates the weight vector for the hidden state. $\mathbf{h}^i_t$ and $\mathbf{w}^i_t$ have the same dimension (64-D in our RNN). We can interpret $\mathbf{w}^i_t$ as measuring the importance of the elements of the hidden state. At time $t+1$, given input $\mathbf{x}^i_{t+1}$ from the previous layer (the second deconvolutional layer in our RNN), unit i updates its hidden state and weight vector, and generates its output according to a set of rules described below. Fig. 5 illustrates the block diagram of the DA-RU.

Data association:

$$\left\langle \widetilde{\mathbf{h}}^i_{t+1}, \widetilde{\mathbf{w}}^i_{t+1} \right\rangle = \begin{cases} \langle \mathbf{0}, \mathbf{0} \rangle, \text{ if no association} \\ \left\langle \mathbf{h}^{i'}_t, \mathbf{w}^{i'}_t \right\rangle, \text{ if } p^i_{t+1} \text{ associated to } p^{i'}_t, \end{cases} \tag{1}$$

where p^i_{t+1} and $p^{i'}_t$ denote the corresponding pixels of unit i at time $t+1$ and unit i' at time t respectively. Eq. (1) indicates that a unit at time t passes its hidden state and weight vector to a unit at time $t+1$ via data association between pixels. If a unit at time $t+1$ is not associated with any previous unit, its hidden state and weight vector are initialized as zeros. All the units in the first frame of a video are initialized with zeros.

Computing weights for the input:

$$\widehat{\mathbf{w}}^i_{t+1} = \sigma(\mathbf{W}[\widetilde{\mathbf{h}}^i_{t+1}, \mathbf{x}^i_{t+1}] + \mathbf{b}), \tag{2}$$

where $\widehat{\mathbf{w}}^i_{t+1}$ is the weight vector for the input $\mathbf{x}^i_{t+1}$, which is a function of the hidden state from the previous frame and the input of the current frame. $\mathbf{W}, \mathbf{b}$ are the parameters of the recurrent layer, which are shared by all the units in the layer, $\sigma(\cdot)$ indicates the logistic sigmoid function, $[\cdot, \cdot]$ denotes concatenation of two vectors. $\mathbf{W}$ is a $d \times 2d$ matrix and $\mathbf{b}$ is a d dimensional bias vector, where d is the dimension of the hidden state.

Updating weight vector:

$$\mathbf{w}^i_{t+1} = \widetilde{\mathbf{w}}^i_{t+1} + \widehat{\mathbf{w}}^i_{t+1}. \tag{3}$$

The weight vector at time $t+1$ is the sum between the accumulated weight vector from the previous frame and the weight vector for the current input.

Updating hidden state:

$$\mathbf{h}^i_{t+1} = f((\widetilde{\mathbf{w}}^i_{t+1} \oslash \mathbf{w}^i_{t+1}) \otimes \widetilde{\mathbf{h}}^i_{t+1} + (\widehat{\mathbf{w}}^i_{t+1} \oslash \mathbf{w}^i_{t+1}) \otimes \mathbf{x}^i_{t+1}), \tag{4}$$

where $f(x) = \max(0, x)$ is the Rectified Linear Unit (ReLU) activation function, and $\oslash, \otimes$ denotes element-wise division and element-wise multiplication between vectors respectively. As we can see from Eq. (4), the new hidden state is computed as a weighted sum between the hidden state from the previous frame and the input for the current frame, where the weights are accumulated in time.

Computing Output:

$$\mathbf{o}_{t+1}^{i} = g(\mathbf{h}_{t+1}^{i}). \tag{5}$$

The output of the unit is defined as a function $g(\cdot)$ of the hidden state. In our RNN, we simply use the hidden state as the output of the unit, i.e., $g(\cdot)$ is the identity function.

We name the aforementioned recurrent unit as the Data Associated Recurrent Unit (DA-RU). DA-RU performs weighted moving average of its input in time via data association, where the weights are dynamically generated based on the parameters of the unit and the data it receives, while the parameters are learned during network training. In DA-RNNs, the DA-RUs are used to combine features from the previous frames and features in the current frame for semantic labeling (Fig. 4).

C. Joint 3D Mapping and Semantic Labeling

In DA-RNNs, data association is needed in order to associate DA-RUs in the recurrent layer across video frames. In general, different data association algorithms can be applied, such as optical flow on RGB images or Iterative Closest Point (ICP) on depth images. In this work, we integrate DA-RNNs with KinectFusion [21], a dense 3D mapping technique using depth camera. KinectFusion estimates the camera poses of the video frames, from which we can compute the data association for the recurrent layer in the RNN. In addition, we fuse the semantic labels of pixels into the volumetric space in KinectFusion. Consequently, our system is able to reconstruct and semantically label the 3D scene.

1) Data Association with KinectFusion: KinectFusion represents the 3D scene with a 3D voxel space which stores the values of the Truncated Signed Distance Function (TSDF). The TSDF value of a voxel indicates the signed distance from the voxel to the closest zero crossing, i.e., surface. Given a stream of depth images, these TSDF values are updated per-frame. In order to fuse the depth images into the voxel space, KinectFusion performs camera tracking by estimating the 6DOF camera pose for each frame. The camera pose estimation is achieved by performing ICP between the 3D points from the current depth image and the 3D points extracted from the surface of the KinectFusion map. Given the camera poses of two consecutive frames from KinectFusion, we compute the data association between the two frames by back-projecting one frame into 3D points in the KinectFusion map and then projecting these 3D points onto the other frame using the estimated camera poses.

2) Semantic Fusion: By combining DA-RNNs with 3D mapping techniques such as KinectFusion, we are able to propagate semantic information into the 3D space. In addition, the semantic labels from different views of the same 3D location are fused in order to obtain a consistent understanding of the 3D space. For each voxel in KinectFusion, we store a probability vector of the semantic label space in addition to the TSDF value. Given a new depth image, the TSDF values of the voxels are first updated as in the traditional KinectFusion. Then, for voxels whose signed distances are smaller than the truncated distance threshold, i.e., voxels around the surface, their probability vectors are updated using the probability map of the semantic labels predicted by the RNN. A running averaging is used for both the TSDF and the probability vector to reduce noise in the fusing process. At any time step, the label of a voxel is predicted as the semantic class with the maximum probability according to the stored probability vector in the voxel. Fig. 1 illustrates the 3D mapping and semantic labeling pipeline of our framework.

IV. Experiments

In this section, we conduct experiments to evaluate our proposed system for 3D scene mapping and semantic labeling.

A. Datasets

1) RGB-D Scene Dataset: Two RGB-D video datasets are used to test our method. The first one is the RGB-D Scene dataset introduced by [18], which consists of 14 RGB-D videos captured by Kinect in indoor scenes. Each scene is reconstructed as a 3D point cloud aligned via the Patch Volumes Mapping method [13]. Then these 3D point clouds are labeled by 9 object class labels plus background. However, the RGB-D Scene dataset does not provide pixel-wise labeling for every video frame, which is needed in order to train and test our RNN. We could project the labeled 3D points onto video frames, but the projection only provides sparse labeling of the frames, i.e., not every pixel is labeled. Instead, we use the following procedure to obtain dense labeling on the RGB-D Scene dataset.

Each scene is first reconstructed with KinectFusion. After the 3D reconstruction, we manually annotate the 3D bounding box of the object in the 3D map. For each depth image, we convert it into a 3D point cloud and transform the 3D point cloud into the reconstructed 3D space according to its camera pose estimated from KinectFusion. Finally, 3D points inside a 3D bounding box are labeled with the semantic class of the object inside the box. In this way, we obtain dense labels for all the depth images. Note that pixels with missing depth values are not labeled according to our labeling procedure. We use 7 videos for training (5,808 frames) and the other 7 videos for testing (5,619 frames).

2) ShapeNet Scene Dataset: The second dataset is a synthetic dataset we generated using 3D shapes from the ShapeNet repository [4]. We selected 3D shapes with high quality and texture in 7 object categories: bottle (110 objects), can (25 objects), cap (23 objects), keyboard (36 objects), monitor (95 objects), mug (65 objects) and table (508 objects). We first compose virtual scenes, each with a table on the ground and five table-top objects among bottle, can, cap, keyboard, monitor and mug. Then, we render each virtual scene into RGB images and depth images from a simulated camera trajectory around the table. To obtain the semantic labels of the rendered images, we color the 3D shapes with distinct colors for each class and render the colored scene again using the same camera trajectory. By checking the color of the pixels in this second-pass rendering, we obtain the class labels of the images.

In total, we generated 100 virtual scenes, i.e., 100 RGB-D videos, by randomly sampling 3D shapes from the 7 object categories. 100 frames are rendered for each scene from a sampled camera trajectory. We use 50 videos for training (5,000 frames) and test on the other 50 videos (5,000 frames). Different from the RGB-D Scene dataset, we make sure that there is no overlapping object instances appearing in both the training set and the test set.

B. Evaluation Metrics

We evaluate our method on semantic labeling of pixels and 3D points. For pixel labeling, we adopt the pixel Intersection over Union (IoU) as the evaluation metric, which is the standard metric used for image semantic labeling. Pixel IoU computes the intersection over union of the predicated pixel labels and the ground truth pixel labels on the entire dataset for every class. For 3D point labeling, we follow [18] and use the precision and recall on 3D points as the evaluation metric in order to achieve a fair comparison.

C. Implementation Details

DA-RNN is implemented using the TensorFlow library [1] with Python interface for communication with the KinectFusion module. In training, the parameters of the first 13 convolutional layers in the feature extraction phase are initialized with the VGG16 network [28] trained on ImageNet [8]. Learning is conducted by Stochastic Gradient Descent (SGD) with momentum, where the loss function is the softmax cross entropy loss for pixels. For our single stream FCN and double stream FCN, each SGD mini-batch is a single image, chosen uniformly at random. For DA-RNN, each SGD mini-batch is a video sequence of 3 consecutive frames. In testing, video frames are processed sequentially, and the hidden states of the DA-RNN are passed to the next frame via data association for the entire video sequence. In this way, the DA-RNN captures long term dependencies between pixels.

D. Comparison on Network Architectures

In this experiment, we fix the network input to RGB images and compare different network architectures for pixel-wise semantic labeling. Table I presents the pixel IoU on the RGB-D Scene dataset for four different networks.

i) We compare our single stream FCN (Fig. 2) with the FCN in [19] which is fine-tuned on the RGB-D Scene dataset using the same experimental setup as ours. As we can see from the table, our single stream FCN significantly outperforms the FCN in [19]. [19] converts the VGG16 network into a fully convolutional network for semantic labeling. There are five max pooling layers and two 4096-dimensional fully connected layers (eventually converted to 1×1 convolutional layers) in the network, which outputs blob-like segmentations and cannot capture fine-grained details of the objects. In contrast, our FCN uses fewer max pooling layers and embeds the convolutional features into a low dimensional space (64-D) before classification, which is able to generate shaper segmentations of the objects.

Methods	FCN [19]	Our FCN	Our GRU-RNN	Our DA-RNN
Background	94.3	96.1	96.8	**97.6**
Bowl	78.6	87.0	86.4	**92.7**
Cap	61.2	79.0	82.0	**84.4**
Cereal Box	80.4	87.5	87.5	**88.3**
Coffee Mug	62.7	75.7	76.1	**86.3**
Coffee Table	93.6	95.2	96.0	**97.3**
Office Chair	67.3	71.6	72.7	**77.0**
Soda Can	73.5	82.9	81.9	**88.7**
Sofa	90.8	92.9	93.5	**95.6**
Table	84.2	89.8	90.8	**92.8**
MEAN	78.7	85.8	86.4	**90.1**

TABLE I
COMPARISON IN NETWORK ARCHITECTURES FOR IMAGE PIXEL LABELING ON THE RGB-D SCENE DATASET. THE NETWORK INPUT IS RGB IMAGE.

ii) We compare two types of recurrent unit in our RNN architecture: Gated Recurrent Unit (GRU) [6] and the DA-RU we introduce in this work. From Table I, we can see that our DA-RU achieves better labeling performance than GRU. First, our DA-RU can be trained more efficiently since it uses the ReLU function instead of the hyperbolic tangent function in updating its hidden state. Second, the DA-RU is explicitly designed as a weighted moving average unit, which is more effective for video-based applications.

iii) By comparing our DA-RNN with the FCN, DA-RNN achieves better labeling accuracy, thanks to its ability in capturing the temporal information across video frames.

E. Analysis on Network Inputs

We conduct experiments to analyze the effect of different types of network inputs on semantic labeling. These inputs are RGB image, depth image, normal image and RGB-D image. For the depth image, we normalize the depth values between 0 and 255 and copy it three times to feed it into the network whose input has 3 channels. For the normal image, we compute the surface normals from the depth image, and then convert the surface normal coordinates into a 3-channel image. A pair of the RGB image and the depth image is referred as a RGB-D image. Our single stream network (Fig. 2) is used to process the RGB image, the depth image or the normal image, while the double stream network (Fig. 3) is used to process the RGB-D image. Both networks can be turned into a DA-RNN by adding a recurrent layer as in Fig. 4.

Table II presents the results of our FCN and DA-RNN with different inputs on the RGB-D Scene dataset. i) Using RGB image achieves better performance than using depth image or using normal image. Since the RGB-D Scene dataset consists of a few number of object instances such as two specific bowls or cereal boxes, color is more discriminative than depth and normal. ii) By using RGB-D images with our double stream FCN, the labeling accuracy is improved over using

Methods	FCN RGB	FCN Depth	FCN Normal	FCN RGB-D	DA-RNN RGB	DA-RNN Depth	DA-RNN Normal	DA-RNN RGB-D
Background	96.1	97.0	95.4	97.8	97.6	98.4	98.4	**98.7**
Bowl	**97.0**	94.8	86.5	89.8	92.7	89.8	91.8	93.1
Cap	79.0	86.7	86.7	82.7	84.4	88.9	**90.5**	87.0
Cereal Box	87.5	88.1	58.3	88.5	88.3	90.6	90.3	**94.2**
Coffee Mug	75.7	81.9	83.1	82.2	86.3	83.1	86.3	**89.4**
Coffee Table	95.2	87.2	83.7	96.3	97.3	91.8	91.7	**97.8**
Office Chair	71.6	79.0	74.8	82.4	77.0	84.2	84.5	**87.5**
Soda Can	82.9	84.4	85.7	86.1	88.7	89.9	88.1	**90.7**
Sofa	92.9	94.2	92.6	96.1	95.6	95.6	96.1	**97.9**
Table	89.8	69.6	68.8	92.7	92.8	81.0	81.1	**94.5**
MEAN	85.8	85.3	81.5	89.4	90.1	89.3	89.9	**93.1**
Improvement of DA-RNN over FCN					+4.3	+4.0	+8.4	+3.7

TABLE II
COMPARISON IN NETWORK INPUTS FOR IMAGE PIXEL LABELING ON THE RGB-D SCENE DATASET.

Methods	FCN RGB	FCN Depth	FCN Normal	FCN RGB-D	DA-RNN RGB	DA-RNN Depth	DA-RNN Normal	DA-RNN RGB-D
Background	99.1	99.0	98.1	99.4	**99.5**	99.3	98.8	**99.5**
Bottle	79.8	80.8	76.8	81.3	84.8	**86.1**	83.1	84.5
Can	64.5	83.7	53.2	67.1	65.2	**84.6**	81.1	66.9
Cap	81.3	85.3	87.4	83.1	84.6	87.9	**91.1**	83.6
Keyboard	90.2	88.9	91.3	91.3	91.2	90.6	91.6	**92.4**
Monitor	87.7	90.7	90.8	92.2	91.2	92.8	**93.9**	93.2
Mug	68.9	84.9	66.4	70.7	70.5	**85.0**	81.2	70.6
Table	94.9	93.7	91.5	96.0	95.8	95.1	94.2	**96.3**
MEAN	83.3	88.4	82.0	85.1	85.3	**90.2**	89.4	85.9
Improvement of DA-RNN over FCN					+2.0	+1.8	+7.4	+0.8

TABLE III
COMPARISON IN NETWORK INPUTS FOR IMAGE PIXEL LABELING ON THE SHAPENET SCENE DATASET.

RGB image, depth image or normal image only. iii) Our DA-RNN consistently improves over its FCN counterpart, which demonstrates the advantages of DA-RNNs on the semantic video labeling task.

Table III presents the labeling results on the ShapeNet Scene dataset. i) Depth is more discriminative than color. This is because objects in the ShapeNet Scene dataset are sampled from hundreds of 3D shapes with different colors. The objects in the test set are unseen in the training set, so their general shape from the depth images are more discriminative than their color. ii) Combing RGB image and depth image does not improve over using depth image only. It seems that adding color information confuses the network from differentiating objects in different categories but with similar color. iii) Our DA-RNNs achieve better performance than the FCNs consistently across different input types. Fig. 6 shows some labeling examples from our FCN and DA-RNN on the RGB-D Scene dataset and the ShapeNet Scene dataset.

F. 3D Scene Labeling Results

In this experiment, we evaluate our framework on the 3D point labeling task. Our DA-RNN generates pixel-wise labeling of each video frame that is integrated into the KinectFusion map to label the 3D scene. Since the 3D points provided in the RGB-D Scene dataset are not in the same 3D space of our KinectFusion map, we use the following procedure to obtain the labels of these 3D points. For each 3D point, we project it to all the video frames with camera poses provided by the dataset and then check its visibility in the frames. A 3D point is visible in a frame if its projection is inside the frame and the value of the projection in the depth image is within a range of the depth of the 3D point. Then we accumulate all the visible labels of the 3D point, and use the one with the maximum frequency as the final label for it.

Methods	HMP2D [18]	HMP3D [18]	HMP2D+3D [18]	DA-RNN RGB-D
Background	42.9 / **99.6**	**99.9** / 80.0	95.8 / 95.0	94.7 / 96.4
Bowl	74.4 / 85.0	**100.0** / **96.2**	97.0 / 89.1	95.3 / 91.0
Cap	74.9 / 98.6	91.3 / 98.2	82.7 / **99.0**	**93.5** / 91.1
Cereal Box	79.9 / 98.6	85.1 / **100.0**	96.2 / 99.3	**98.0** / 93.5
Coffee Mug	64.4 / 87.8	90.0 / **93.9**	81.0 / 92.6	**90.5** / 86.3
Coffee Table	11.9 / 17.9	96.1 / **100.0**	**98.7** / 98.0	93.8 / 97.3
Office Chair	17.7 / 17.2	57.6 / **100.0**	89.7 / 94.5	**96.0** / 96.2
Soda Can	78.2 / **98.1**	**100.0** / 81.9	97.7 / 98.0	92.0 / 83.4
Sofa	29.3 / 39.8	82.7 / **100.0**	92.5 / 92.0	**99.6** / 91.3
Table	16.4 / 23.3	95.3 / **98.7**	97.6 / 96.0	**98.2** / 97.2
MEAN	49.0 / 66.6	89.8 / 94.9	92.8 / **95.3**	**95.2** / 92.4

TABLE IV
PRECISION AND RECALL OF 3D POINT LABELING ON THE RGB-D SCENE DATASET.

Table IV presents the 3D point labeling precision and recall on the RGB-D Scene dataset, where we compare our method with three variations of the method proposed in [18]. It is worth to mention that the models in [18] are trained with synthetic data generated from rendering 3D shapes only and tested on all the 14 videos in the RGB-D Scene dataset. To test our method on the same videos, we conduct a two-fold cross validation and obtain the results on all the 14 videos. From Table IV, we can see that our method achieves comparable 3D point labeling precision and recall with the HMP2D+3D model. While [18] employs several heuristics to remove the ground plane and the table-top, our system processes a RGB-D video automatically without such heuristics. Fig. 7 shows some semantic mapping results on the RGB-D Scene dataset.

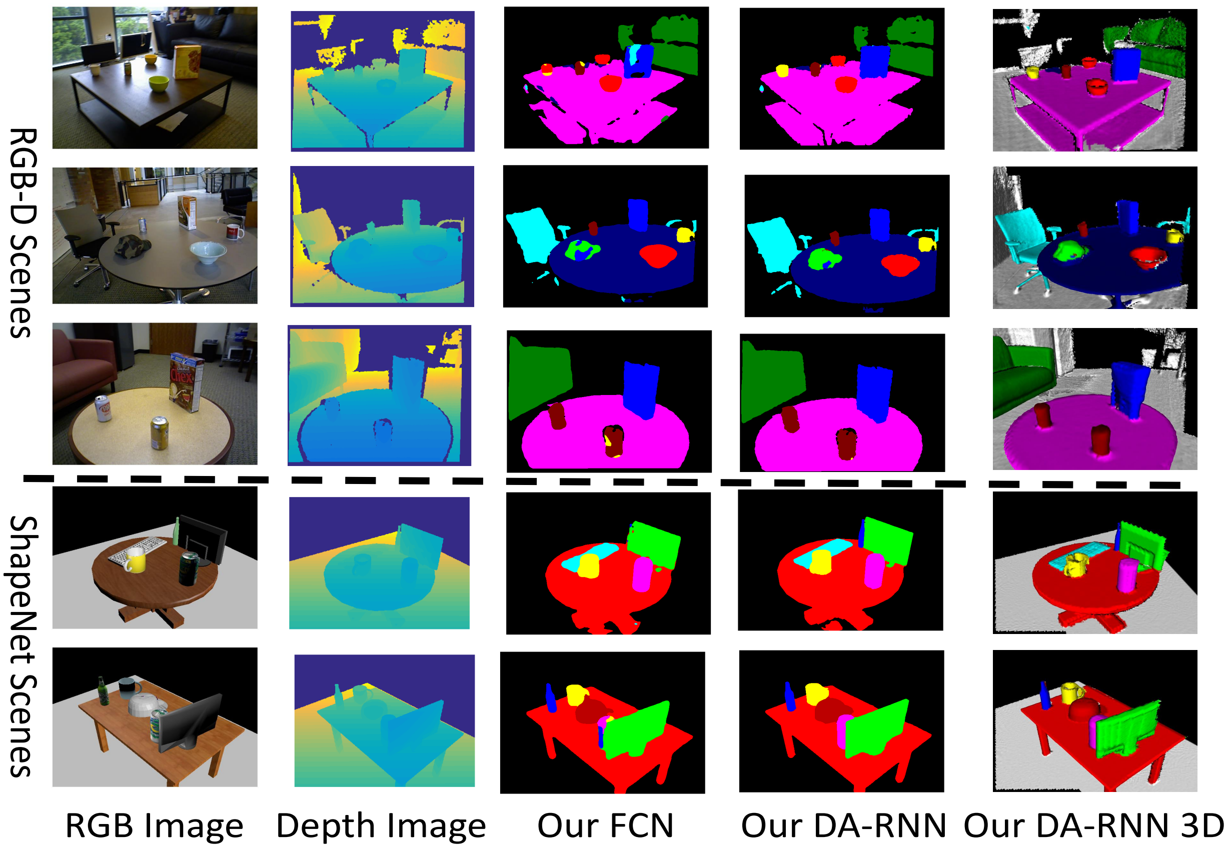

Fig. 6. Semantic labeling examples on the RGB-D Scene dataset and the ShapeNet Scene dataset.

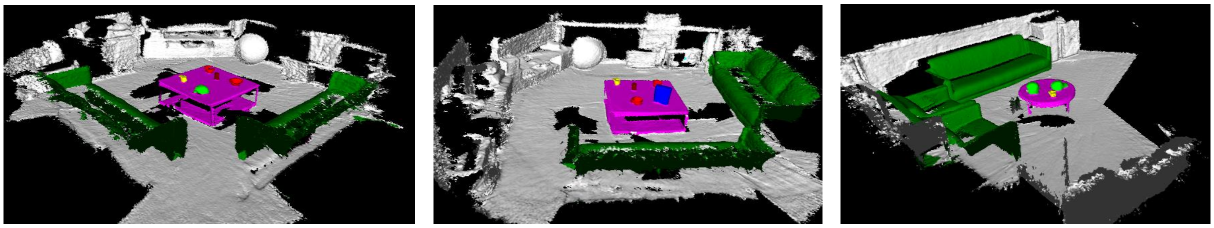

Fig. 7. The semantic 3D mapping built by our method using the RGB-D Scene dataset.

Please see our project website for the result video on the two datasets. The labeling errors are more often caused by confusion between classes with similar 3D shape such as mug and can. Data association accuracy also affects the performance. For example, we sometimes see that the bottom of an object is labeled as table due to wrong assocation to table pixels in the previous frame.

V. Conclusion

In this work, we introduce DA-RNNs, a novel framework for joint 3D mapping and semantic labeling on RGB-D videos. DA-RNNs integrate a recurrent neural network for video semantic labeling with KinectFusion. To achieve a compact network representation, recurrent reasoning is only performed over the currently visible part of the environment, using data association to define the connectivity between recurrent units. The labels predicted by the RNN are fused into the KinectFusion map for dense semantic mapping. Experiments are conducted on a real world dataset and a synthetic dataset of RGB-D videos. The experimental results and analyses demonstrate the advantages of our method on video semantic labeling and 3D scene mapping.

A key advantage of DA-RNNs is their flexibility. While this paper focuses on object class labeling, we believe that the same architecture could be applied to train networks for a wide range of semantic labeling problems, including object instance and pose detection, material recognition, and physical support estimation. Data association between frames can also be obtained in different ways such as using optical flow methods. Another promising avenue for improvement is the incorporation of shape information provided by the 3D map.

Acknowledgments

This work was funded in part by ONR grant N00014-13-1-0720 and by Northrop Grumman. We thank Tanner Schmidt for fruitful discussions and for providing his implementation of KinectFusion.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[2] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv preprint arXiv:1511.00561*, 2015.

[3] Eric Brachmann, Alexander Krull, Frank Michel, Stefan Gumhold, Jamie Shotton, and Carsten Rother. Learning 6D object pose estimation using 3D object coordinates. In *European Conference on Computer Vision (ECCV)*, pages 536–551, 2014.

[4] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3D model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *arXiv preprint arXiv:1606.00915*, 2016.

[6] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[7] David Crandall, Andrew Owens, Noah Snavely, and Dan Huttenlocher. Discrete-continuous optimization for large-scale structure from motion. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3001–3008, 2011.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009.

[9] Pedro Felzenszwalb, David McAllester, and Deva Ramanan. A discriminatively trained, multiscale, deformable part model. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008.

[10] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 580–587, 2014.

[11] Alex Graves, Navdeep Jaitly, and Abdel-rahman Mohamed. Hybrid speech recognition with deep bidirectional LSTM. In *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 273–278, 2013.

[12] Peter Henry, Michael Krainin, Evan Herbst, Xiaofeng Ren, and Dieter Fox. RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments. *The International Journal of Robotics Research*, 31(5):647–663, 2012.

[13] Peter Henry, Dieter Fox, Achintya Bhowmik, and Rajiv Mongia. Patch volumes: Segmentation-based consistent mapping with RGB-D cameras. In *International Conference on 3D Vision (3DV)*, pages 398–405, 2013.

[14] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[15] Maik Keller, Damien Lefloch, Martin Lambers, Shahram Izadi, Tim Weyrich, and Andreas Kolb. Real-time 3D reconstruction in dynamic scenes using point-based fusion. In *International Conference on 3D Vision (3DV)*, pages 1–8, 2013.

[16] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected CRFs with gaussian edge potentials. In *Advances in Neural Information Processing Systems (NIPS)*, pages 109–117, 2011.

[17] Kevin Lai. *Object Recognition and Semantic Scene Labeling for RGB-D Data*. PhD thesis, University of Washington, 12 2013.

[18] Kevin Lai, Liefeng Bo, and Dieter Fox. Unsupervised feature learning for 3D scene labeling. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3050–3057, 2014.

[19] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440, 2015.

[20] John McCormac, Ankur Handa, Andrew Davison, and Stefan Leutenegger. SemanticFusion: Dense 3D semantic mapping with convolutional neural networks. *arXiv preprint arXiv:1609.05130*, 2016.

[21] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 127–136, 2011.

[22] Mircea Serban Pavel, Hannes Schulz, and Sven Behnke. Recurrent convolutional neural networks for object-class segmentation of RGB-D video. In *International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2015.

[23] Xiaofeng Ren, Liefeng Bo, and Dieter Fox. RGB-D scene labeling: Features and algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2759–2766, 2012.

[24] Renato Salas-Moreno, Richard Newcombe, Hauke Strasdat, Paul Kelly, and Andrew Davison. SLAM++: Simultaneous localisation and mapping at the level of objects. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1352–1359, 2013.

[25] Silvio Savarese and Li Fei-Fei. 3D generic object categorization, localization and pose estimation. In

International Conference on Computer Vision (ICCV), pages 1–8, 2007.

[26] Evan Shelhamer, Kate Rakelly, Judy Hoffman, and Trevor Darrell. Clockwork convnets for video semantic segmentation. In *Video Semantic Segmentation Workshop at European Conference in Computer Vision (ECCV)*, pages 852–868, 2016.

[27] Jamie Shotton, John Winn, Carsten Rother, and Antonio Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *European Conference on Computer Vision (ECCV)*, pages 1–15, 2006.

[28] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[29] Noah Snavely, Steven M Seitz, and Richard Szeliski. Skeletal graphs for efficient structure from motion. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[30] Shuran Song and Jianxiong Xiao. Deep sliding shapes for amodal 3D object detection in RGB-D images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 808–816, 2016.

[31] Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. LSTM neural networks for language modeling. In *Interspeech*, pages 194–197, 2012.

[32] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3104–3112, 2014.

[33] Thomas Whelan, Michael Kaess, Maurice Fallon, Hordur Johannsson, John Leonard, and John McDonald. Kintinuous: Spatially Extended KinectFusion. In *RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*, 2012.

[34] Thomas Whelan, Stefan Leutenegger, Renato F Salas-Moreno, Ben Glocker, and Andrew J Davison. ElasticFusion: Dense SLAM without a pose graph. In *Robotics: science and systems*, volume 11, 2015.

[35] Yu Xiang and Silvio Savarese. Estimating the aspect layout of object categories. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3410–3417, 2012.

[36] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1529–1537, 2015.