3D Attention-Driven Depth Acquisition for Object Identification

Kai Xu[1,2,3] Yifei Shi[1] Lintao Zheng[1] Junyu Zhang[4] Min Liu[1] Hui Huang[3,4]* Hao Su[5] Daniel Cohen-Or[6] Baoquan Chen[2]*
[1]HPCL, National University of Defense Technology [2]Shandong University
[3]Shenzhen University [4]SIAT [5]Stanford University [6]Tel-Aviv University

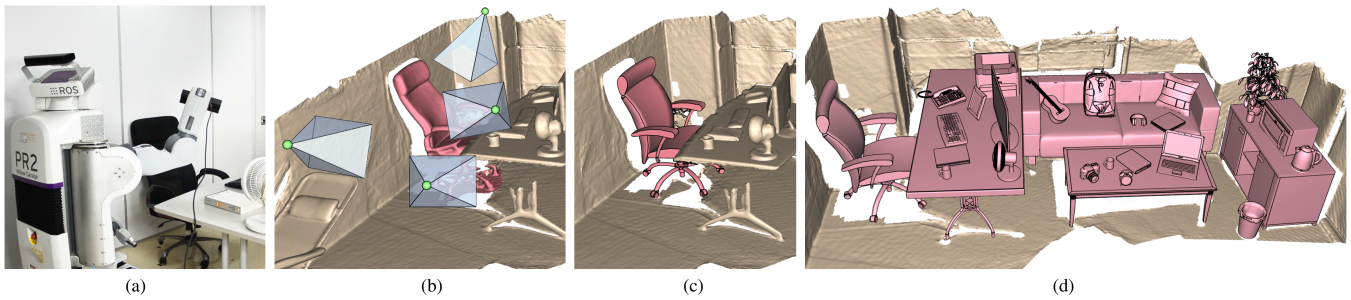


Figure 1: *We present an autonomous system for active object identification in an indoor scene (a), with consecutive depth acquisitions. The scene is first roughly scanned, and segmented to generate 3D object proposals. Targeting an object proposal (b), the robot performs multi-view object identification, based on a 3D shape database, driven by a 3D Attention Model. The retrieved 3D models are inserted into the scanned scene (c), replacing the corresponding object scans, thus incrementally constructing a 3D scene model (d).*

Abstract

We address the problem of autonomously exploring unknown objects in a scene by consecutive depth acquisitions. The goal is to reconstruct the scene while online identifying the objects from among a large collection of 3D shapes. Fine-grained shape identification demands a meticulous series of observations attending to varying views and parts of the object of interest. Inspired by the recent success of attention-based models for 2D recognition, we develop a *3D Attention Model* that selects the best views to scan from, as well as the most informative regions in each view to focus on, to achieve efficient object recognition. The region-level attention leads to focus-driven features which are quite robust against object occlusion. The attention model, trained with the 3D shape collection, encodes the temporal dependencies among consecutive views with deep recurrent networks. This facilitates order-aware view planning accounting for robot movement cost. In achieving instance identification, the shape collection is organized into a hierarchy, associated with pre-trained hierarchical classifiers. The effectiveness of our method is demonstrated on an autonomous robot (PR) that explores a scene and identifies the objects to construct a 3D scene model.

Keywords: 3D acquisition, depth camera, next-best-view, object identification, attention-based model, shape classification

Concepts: •**Computing methodologies** → *Shape analysis;*

1 Introduction

Autonomous acquisition and modeling of indoor environments by a robot has a wide variety of potential applications, ranging from object manipulation by service robots, to content preparation for virtual and augmented reality. In view of the fast development of 3D sensing techniques and the proliferation of 3D geometric shape datasets, we approach online indoor scene reconstruction via utilizing depth acquisition and adopting data-driven shape analysis. Given a series of progressively acquired 2.5D depth images, the task involves first extracting potential objects, and then identifying for each object the most similar 3D shape from a database to build the scene. The process of identification would guide the robot to select the best view for the next observation. Our work focuses on this second part, i.e., guided acquisition for object identification.

Autonomous object identification is a challenging setting. First of all, unlike coarse-level shape classification, e.g., discriminating a chair from a bicycle for which a single view would suffice, fine-grained object identification over a large shape collection is a significantly harder endeavor. The latter often requires meticulous observations from varying view angles. During robot-operated multi-view recognition, it is indispensable to attain intelligent view planning accounting for both recognition efficiency and robot movement cost. On the other hand, since the object is unknown, planning the views to best discriminate it within a shape collection is a daunting task. Indeed, object recognition and view planning are coupled problems, calling for a unified optimization framework. Moreover, real indoor scenes are often cluttered, causing the target objects to be occluded. Occlusion may not only degrade recognition accuracy but also invalidate the off-line learned viewing policy.

The recent interest in attention mechanism has led to great success in various vision tasks based on 2D images (e.g. [Mnih et al. 2014; Xu et al. 2015b]). Attention-based models achieve both efficiency and accuracy by focusing their processing only on the most informative parts of the input, with respect to a specific task. This is inspired by the observation that humans focus attention selectively on regions of the image being observed. Information gained progressively from different fixations is integrated to approach a desired goal and to guide future attention. Such mechanism, being simultaneously goal-directed and stimulus-driven [Corbetta and Shulman 2002], fits well to our problem setting by solving both object recognition and guided acquisition in a unified optimization.

In this work, we propose a *3D Attention Model* with the objective of object identification. The model encompasses two levels of attention. The first level selects the next-best-views (NBVs) for

*Corresponding authors: {hhzhiyan, baoquan.chen}@gmail.com

SA '16 Technical Papers,, December 05-08, 2016, , Macao
ISBN: 978-1-4503-4514-9/16/12
DOI: http://dx.doi.org/10.1145/2980179.2980224

depth acquisition targeting at an object of interest, while the second concentrates on the most discriminative regions in each view for part-based recognition. Both levels are trained using synthetic 3D models, with respect to the task of object classification. The view-level attention is formulated as *sequential NBV regression* based on Recurrent Neural Networks (RNNs). In each time step, it integrates the order-aware information of all past views and estimates the next best one. The region-level attention is embodied in learning occlusion-tolerant features for object classification. The resulting focus-driven features attend only to the parts which are both visible and characteristic, significantly increasing the chance of successful recognition under partial occlusion.

Our method is integrated with an autonomous system of scene scanning and object extraction [Xu et al. 2015a], operated by a personal robot holding a depth camera. Focusing on an extracted object, the robot performs active depth acquisition guided by predicted NBVs, to identify the object. The retrieved 3D model is inserted into the scene, to replace the object scan, thus progressively constructing a 3D scene model (Figure 1). Our main contributions include:

- A 3D attention model for active 3D object identification with multi-view depth acquisition.
- Sequential modeling of NBV regression based on a new recurrent architecture integrating efficient view aggregation.
- Focus-driven feature learning based on part-level attention for occluded object recognition.
- An autonomous system of online object identification.

2 Related Work

Online scene analysis and modeling. With the fast development of commodity depth cameras and realtime reconstruction techniques (e.g., [Newcombe et al. 2011; Nießner et al. 2013]), online scene analysis during the scanning process becomes increasingly popular, due to its advantages facilitating instant user interaction [Zhang et al. 2014; Valentin et al. 2015], autonomous scanning guidance [Xu et al. 2015a], and scene modeling with 3D shape database [Salas-Moreno et al. 2012]. Our work also takes the data-driven approach. But unlike existing works that rely either on partial scans [Li et al. 2015a] or depth videos [Chen et al. 2014], we opt for 3D recognition based on a sparse set of view observations, taking advantage of recent advances on multi-view deep learning models for 3D geometric data [Su et al. 2015a].

Active reconstruction and recognition. The classical next-best-view problem has mostly been considered in active 3D acquisition for surface reconstruction, aiming at covering the surface of an object with a minimal scanning effort [Wu et al. 2014]. Other works develop NBV methods to reduce the recognition uncertainty with a minimal number of views [Atanasov et al. 2014; Wu et al. 2015]. Recently, Xu et al. [2015a] present robot-operated scene reconstruction with active object segmentation. The geometry-based object segmentation can generate object proposals for our work.

The recent work of Wu et al. [2015] is the most closely related to ours. They adopt a volumetric representation of 3D shapes and train a Convolutional Deep Belief Network (CDBN) to model the joint distribution over volume occupancy and shape category, serving as a shape classifier. By sampling from the distribution, they can perform shape completion based on the observed depth images, over which virtual scanning is performed to estimate the information gain of a view. Our method differs from theirs in three major aspects. *First*, our attention model is sequential in nature, facilitating a global view planning as well as the incorporation of movement cost. *Second*, we address the problem of instance-level recognition from a large shape database, for which a single classifier is infeasible for the highly fine-grained discrimination. *Third*, our NBV predictor is fully trained in the off-line stage and no online sampling is required, making it efficient for online active recognition.

Fine-grained shape classification. Fine-grained object classification has been a hot topic in computer vision (e.g. [Krause et al. 2015]), and recently draws increasing attention from the graphics community [Huang et al. 2013; Kleiman et al. 2015]. Fine-grained classification often calls for localized analysis with more discriminative features. In our work, we cast the problem of object identification from a 3D shape collection into instance-level classification, in analogy to the person re-identification problem [Haque et al. 2016]. Therefore, the number of classes handled by our method is extremely huge when a large shape repository (e.g., ShapeNet [Su et al. 2015c]) is employed. A large number of classes amplifies the difficulty of lacking training data. We address these difficulties by learning a hierarchy of coarse-to-fine classifiers.

Learning a hierarchy has been widely practiced in scalable image classification, with either supervised construction based on semantic taxonomies [Li et al. 2010], or unsupervised discovery from a collection of images [Bart et al. 2008]. To mitigate early misclassification at high levels, relaxed hierarchy of binary classifiers is introduced [Gao and Koller 2011]. At each node, a subset of confusing instances is extracted, whose classification is postponed to the finer-grained classifiers of lower levels. We learn a relaxed hierarchy of multi-class classifiers in a weakly supervised fashion, where the features at each level are progressively enhanced to accommodate the increasingly fine-grained classification.

Attention-based models. Sequential modeling with deep learning models has been gaining extensive research lately, which reactivated the interest on attention-based models in various tasks. Attention models exploit the spatial or temporal structure of the input by a sequence of partial observation and combine the acquired information over time to build up an internal representation of an object or a scene. Such models have achieved notable success in 2D object recognition [Mnih et al. 2014; Ba et al. 2014], image classification [Xiao et al. 2015], image captioning [Xu et al. 2015b], and more recently for person identification from volumetric 3D data [Haque et al. 2016]. To our knowledge, our work is the first to adapt attention mechanism for view-based 3D shape recognition.

3 Overview

Online object identification. Our system starts by roughly scanning the target scene, and performs 3D geometry-based object-level segmentation over the 3D scan using the method in [Xu et al. 2015a]. The segmented objects serve as 3D object proposals to bootstrap our active recognition. Targeting at one object proposal, our goal is to identify it from among a large shape collection, with multi-view depth observations. The 3D shape which is the most similar to the object is retrieved and placed into the scene with proper orientation and scaling, to replace the corresponding object scan. This repeats until all object proposals are processed, resulting in a partially modeled 3D scene.

Recurrent 3D attention model. To approach active 3D object identification, we propose a recurrent 3D attention model (3D-RAM) which performs both instance-level shape classification and sequential NBV regression (Figure 2d). At each time step, the robot acquires a depth image, updates the internal state of the recurrent model by aggregating the new view with the past ones, conducts shape classification, and regresses the next-best-view. Motivated by the effective view aggregation of Multi-View Convolutional Neural

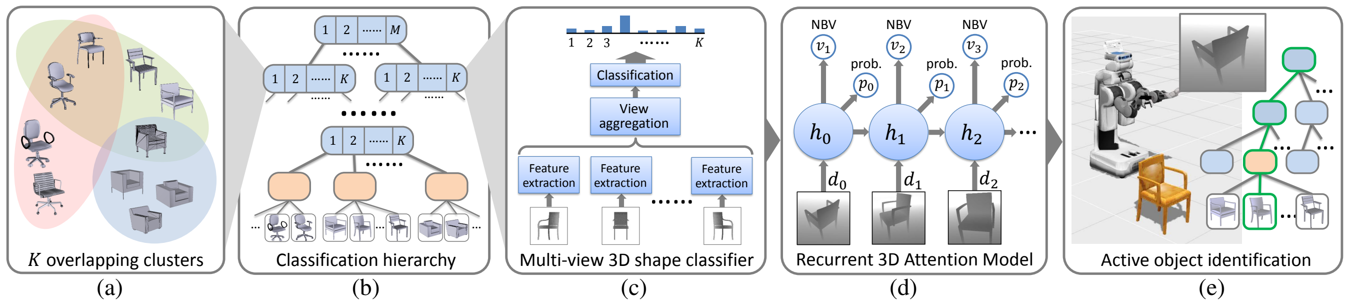


Figure 2: *Our core contribution is a recurrent 3D attention model (3D-RAM) (d). At each time step, it takes a depth image as input, updates its internal state, performs shape classification and estimates the next-best-view. To make instance-level classification feasible, we organize the shape collection with a classification hierarchy (b). At each node of the hierarchy, a multi-view 3D shape classifier (c) assigns the associated shapes into K overlapping groups (a). Online object identification is guided by the 3D-RAM while traversing down the hierarchy for coarse-to-fine classification (e). The feature extractor and classifier of the corresponding node are employed by the 3D-RAM.*

Networks (MV-CNN) [Su et al. 2015a], we implant the MV-CNN into our recurrent network and derive a Multi-View RNN (MV-RNN), achieving more efficient model training and inference.

Enhancement to 3D-RAM. We introduce two enhancements to the visual feature encoding of depth images, for improved fine-grained classification and robust recognition under occlusion, respectively. *Firstly*, we organize the shape collection with a hierarchy of coarse-to-fine MV-CNN classifiers, to progressively approach instance-level classification (Figure 2b). The CNN at a node is trained by fine-tuning the one inherited from its parent node, thus learning enhanced features particularly effective for the fine-grained task of the current level. *Secondly*, to tolerate object occlusion, we learn at each node focus-driven features base on part-level attention. This feature encodes the information of both characteristic part discerning and part-based shape discrimination, thus being quite robust against partial occlusion. Both the learning and inference of MV-RNN are coupled with hierarchy traversing. At a given node during the traverse, the corresponding features learned for that node are employed by the MV-RNN (Figure 2e).

4 3D attention model for object identification

Attention models are particularly amenable for tasks where the agent has only partial observation against the target object/scene. Different from existing attention-based models for 2D vision tasks, where the focuses are regions of a 2D image, the attentions in our problem setting are naturally the views in 3D space pointing to a 3D object. Let us denote by $Q^t := (v_1, v_2, \dots, v_t)$ a sequence of t sensor views targeting at the object of interest, or t virtual views sampled around a 3D model. Denote by d_t the depth image captured/rendered from view v_t. When the context is clear, we refer to a view and its corresponding depth image interchangeably.

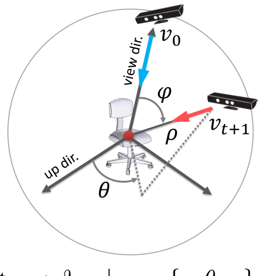
4.1 Goal and stimulus

There are two driving factors in attention mechanism: 1) a *goal* that an agent tries to achieve and 2) *stimulus* the agent receives at every time step. Both factors direct the agent in selecting its attention for obtaining new information. In light of this, we start by discussing i) our core task of 3D object identification as well as ii) the view-based depth observation serving as visual stimulus.

3D object identification. We formulate this problem as instance-level 3D shape classification. To learn a 3D shape classifier, we adopt the projective approach and perform classification based on multi-view rendered depth images, inspired by the recent work of Multi-View CNN (MV-CNN) trained with rendered RGB images [Su et al. 2015a]. Working with multi-view depth images offers a natural connection to the setting of 3D recognition based on 2.5D depth acquisition, avoiding the costly conversion between views and a full volume as in [Wu et al. 2015].

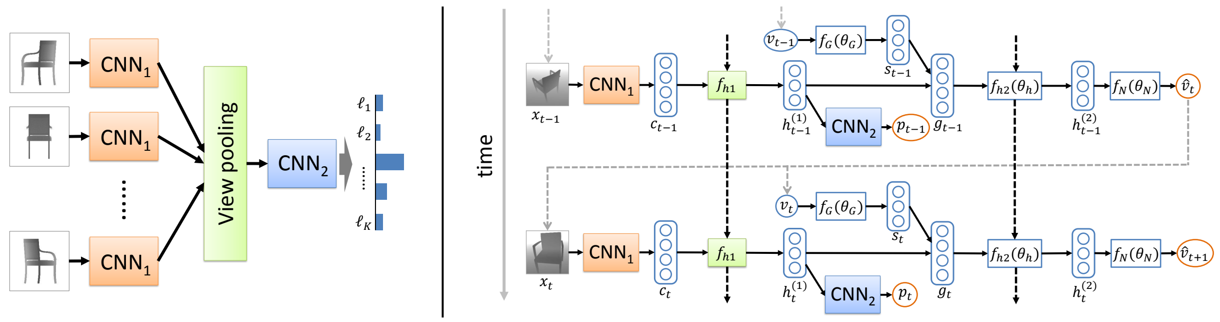
The success of MV-CNN can be attributed to the per-view feature learning and the multi-view feature aggregation. The key idea is to insert a *view pooling* layer between the feature extraction layers (denoted as CNN$_1$) and the classification ones (CNN$_2$) of a CNN; see Figure 3(a) for illustration. Multi-view depth images are fed into the multiple view channels, each represented by several convolutional layers with shared weights, for feature extraction. The multi-view feature maps are then integrated via *max pooling* operation and passed to the remaining layers for classification. We embed a pre-trained MV-CNN into our attention model as a visual subnetwork to perform multi-view object recognition.

View-based observation. In order to achieve *continuous view planning*, we work with the full viewing space parameterized on a viewing sphere around a 3D model in training or an object of interest during testing (see the inserted figure). At any given view, a 2.5D depth image is either rendered from the targeted 3D model or captured for the object of interest, which is input into our attention model. We cast the problem of NBV estimation into a view regression over the viewing parameters. Taking a sequence of t past views Q^t as input, the trained NBV regressor estimates the parameters of the optimal next view $\hat{v}_{t+1}$, which is defined as the *relative sensor movement* with respect to the *initial* view v_0 (the blue arrow). Specifically, the next view v_{t+1} (red arrow) is parameterized in the *local* spherical coordinate system built w.r.t. v_0: $\vartheta_{t+1}|_{v_0} = \{\rho, \theta, \varphi\}_{v_0}$, where ρ, θ and φ are the radius distance to the shape center, azimuthal and polar angle, respectively. Note that by working with a fixed local reference frame, the subsequent views can be directly obtained, without the requirement of object orientation.



4.2 Recurrent attention model for NBV regression

We model 3D view-based attention using recurrent neural networks (RNNs), inspired by the recently proposed Recurrent Attention Model (RAM) for 2D digit recognition [Mnih et al. 2014]. RNN can be viewed as a discrete-time dynamic system with sequential input and output. At each time step, it processes the input obser-



(a) MV-CNN. (b) MV-RNN.

Figure 3: *Our recurrent 3D attention model (b) is a two-layer stacked recurrent neural network. The subnetworks responsible for feature extraction, feature aggregation and classification are substituted by CNN$_1$, view pooling and CNN$_2$ of an MV-CNN (a), respectively. The corresponding subnetworks are shaded with consistent colors in (a) and (b). (b) shows a two-time-step unfolding of MV-RNN, where the dashed arrows indicate information flow across time steps while the solid ones represent that within a time step.*

vation, aggregates information using internal state, performs action (e.g., classification), and deploys observation for the next step.

To adapt this model to our problem setting, we take the advantage of MV-CNN in multi-view 3D object recognition and propose a 3D-RAM, based on a new architecture which we call MV-RNN (Figure 3b). Similar to RAM, our recurrent network is comprised of subnetworks responsible for input processing, information aggregation, action generation and next view prediction, respectively. There are, however, two major differences. *First* of all, we devise max-pooling-based recurrent units to achieve powerful view aggregation as in MV-CNN. *Second*, the glimpse integration of depth images and view parameters is postponed after the above view aggregation, to attain more informative NBV estimation. Below, we explain the detailed design and arrangement of our network.

View aggregation network. Given the input depth image of the current view, we extract its feature map by passing it through CNN$_1$: $c_t = f_{\text{CNN1}}(d_t; \theta_{\text{CNN1}})$. θ_{CNN1} summarizes the parameters of CNN$_1$ which are pre-trained outside the recurrent network. The extracted feature map c_t is then integrated with those of all past views via max pooling: $h^{(1)}_t = f_{\text{MP}}(c_t, h^{(1)}_{t-1})$, where f_{MP} performs element-wise $\max$ operation for two vectors. The hidden recurrent units $h^{(1)}_t$ store the feature map aggregated for views until time t. See these subnetworks in Figure 3(b), with the color-coding in correspondence to those in MV-CNN in 3(a). By embedding pre-trained MV-CNN, the resulting MV-RNN achieves more efficient training and more accurate classification (Figure 4).

View glimpse network. This network amalgamates the information of both aggregated depth information and the parameters of the previous view, ϑ_{t-1}, and outputs a feature vector describing the viewing "glimpse" of the current step. The former part simply comes from the hidden recurrent unit $h^{(1)}_t$ output by the view aggregation network. The viewing parameters are encoded into a feature vector by a non-linear function: $s_t = f_{\text{G}}(\vartheta_t; \theta_{\text{G}})$, which shares the same dimension with $h^{(1)}_t$. The final glimpse vector is formed by the element-wise multiplication between $h^{(1)}_t$ and s_t: $g_t = h^{(1)}_t \odot s_t$. Since the glimpse network is postponed after the view aggregation, the resulting glimpse vector encodes the integration of all past observations. This provides a more efficient way of imparting contextual information to the following view regression.

Glimpse aggregation network. To aggregate the information gained from all past glimpses, we employ a second layer of recurrent hidden units, $h^{(2)}_t$, which relay the past information from $h^{(2)}_{t-1}$ while absorbing new data from the current glimpse g_t: $h^{(2)}_t = f_{\text{R}}(h^{(2)}_{t-1}, g_t; \theta_{\text{R}})$. We use "vanilla" recurrent units instead of Long-Short Term Memory (LSTM) ones [Hochreiter and Schmidhuber 1997] since our view sequences are generally not very long.

NBV regression network. To regress the best view for the next time step, this subnetwork takes the current state of the above recurrent units as input and produces a vector of NBV parameters. This amounts to a fully connected layer that maps the hidden units $h^{(2)}_t$ to viewing parameters: $\hat{\vartheta}_{t+1} = f_{\text{N}}(h^{(2)}_t; \theta_{\text{N}})$.

Classification network. At each time step, our model makes a prediction about the class label y of the object of interest, based on the *first* recurrent hidden units $h^{(1)}_t$. We opt for conducting classification right after the view aggregation layer, instead of the glimpse aggregation one, since the former already makes prediction with high accuracy. To this end, we simply put CNN$_2$ after $h^{(1)}_t$ to produce a vector of classification probabilities over all labels: $p_t(y|Q^t) = f_{\text{CNN2}}(h^{(1)}_t; \theta_{\text{CNN2}})$. Note that CNN$_2$ is also pre-trained beforehand, independent of MV-RNN.

4.3 Training and inference

Our MV-RNN is trained using 3D shape database via identifying them with virtually scanned depth images. Since the ground-truth label for each shape is known *a priori*, we can formulate the learning as a supervised classification problem using cross-entropy loss. Minimizing the loss is equivalent to maximizing the log-likelihood of class label, which can be marginalized over the latent observation views v: $\log \sum_v p(v|S,\Theta)p(y|v,S,\Theta)$ with S being the 3D shape and Θ the parameters of MV-RNN.

The marginalized objective function can be optimized via maximizing its lower bound as shown in [Ba et al. 2014]:

$$\text{maximize} \quad \mathcal{F} = \sum_v p(v|S,\Theta) \log p(y|v,S,\Theta). \qquad (1)$$

To maximize (1), we estimate the gradient of $\mathcal{F}$ with respect to model parameter Θ:

$$\sum_v p(v|S,\Theta) \left[\frac{\partial \log p(y|v,S,\Theta)}{\partial \Theta} + \log p(y|v,S,\Theta) \frac{\partial \log p(v|S,\Theta)}{\partial \Theta} \right],$$

The first term in the summation is the gradient of classification prediction. Since our classification network is pre-trained outside the

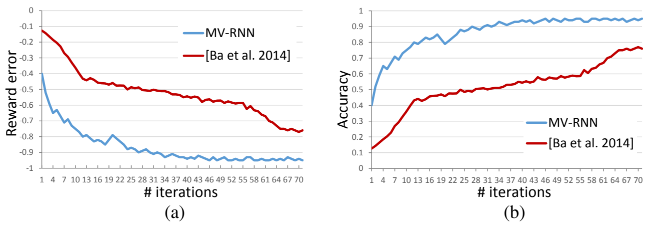


Figure 4: *For the task of multi-view 3D shape classification, our MV-RNN achieves faster convergence in training (a) and more accurate classification in testing (b), compared to the RNN architecture proposed in [Ba et al. 2014].*

MV-RNN, we simply omit this term. The second term can be used to train the NBV regression network by "simulating" the classification of S with a sequence of T views, starting from a random view. To avoid examining exponentially many view combinations over time, we sample the views at each time step using Monte Carlo method: $\tilde{v}_t^m \sim p(v_t|S,\Theta) = \mathcal{N}(v_t;\hat{v}_t,\Sigma)$, where $\hat{v}_t$ is the estimated view at time step t and Σ is a predefined standard deviation (referred to as view variance). The training is performed for M episodes. So the gradient becomes:

$$\frac{1}{M}\sum_{m=1}^{M}\sum_{t=1}^{T}\log p(y|\tilde{v}_t^m,S,\Theta)\frac{\partial \log p(\tilde{v}_t^m|S,\Theta)}{\partial\Theta}. \qquad (2)$$

The log-likelihood $\log p(y|\tilde{v}_t^m,S,\Theta)$ can be approximated by a variance-bounded *reward* function R_t^m (see definition below) indicating the correctness of classification at each time step. Using the baseline technique [Mnih et al. 2014] to center the reward, the gradient approximation becomes:

$$\frac{1}{M}\sum_{m=1}^{M}\sum_{t=1}^{T}(R_t^m - b_t)\frac{\partial \log p(\tilde{v}_t^m|S,\Theta)}{\partial\Theta}, \qquad (3)$$

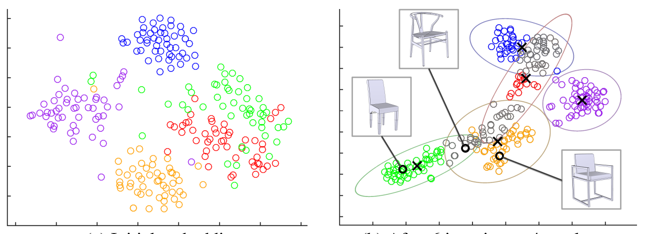
The baseline at time step t can be estimated based on the second layer hidden units: $b_t = f_{\mathrm{B}}(h_t^{(2)};\theta_{\mathrm{B}})$, which can be learned by regressing towards the expected value of R_t^m. This gradient corresponds to the REINFORCE learning rule in a Partially Observable Markov Decision Process (POMDP) [Williams 1992].

Reward. At each time step, the robot receives a reward signal $r_{t'}$. The accumulative reward until time step t is: $R_t = \sum_{t'=1}^{t} r_{t'}$. Our objective in each training episode is to maximize the accumulative reward during a time period of T, as shown in Equation (3). In practice, we train our model using varying instead of fixed T, to accommodate the unknown number of views required in a real case. The definition of the reward at each time step takes both classification result and sensor movement into account. Specifically, the reward is composed of the *prediction accuracy* against ground-truth, the *information gain* of the current classification prediction over the last one, as well as the *movement cost* caused by view change:

$$r_t = H_t(p_t,\bar{p}) + I_t(p_t,p_{t-1}) - C_t. \qquad (4)$$

The first term measures the classification accuracy as the cross-entropy between the predicted distribution p_t and the ground-truth one $\bar{p}$, which is exactly the log-likelihood $\log p(y|\tilde{v}_t^m,S,\Theta)$ in Equation (2). To bound its variance, we approximate it with a binary indicator function:

$$H_t(p_t,\bar{p}) = \begin{cases} 1 & y = \arg\max_y \log p(y|\tilde{v}_t^m,S,\Theta) \\ 0 & \text{otherwise} \end{cases} \qquad (5)$$



(a) Initial embedding. (b) After 6 iterations w/ overlap.

Figure 5: 280 *chair models in five classes are clustered into five overlapping groups after* 6 *iterations. Colors indicate ground-truth labels and grey is for overlapping data points.*

The information gain is defined as decrease of Shannon entropy:

$$I_t(p_t,p_{t-1}) = H(p_{t-1}) - H(p_t), \qquad (6)$$

with $H(p) = -\sum_k p(y_k)\log p(y_k)$. The movement cost is defined as the great circle distance between v_{t-1} and v_t, normalized by the maximum possible arc movement:

$$C_t = \frac{\sqrt{(\Delta_t\rho)^2 + (\rho_{t-1}\Delta_t\phi)^2 + (\rho_{t-1}\sin\phi\Delta_t\theta)^2}}{\sqrt{(\Delta_{\mathrm{m}}\rho)^2 + (\pi\rho_{\mathrm{m}})^2}}, \qquad (7)$$

where $\Delta_t\rho = \rho_t - \rho_{t-1}$ and the same goes for $\Delta\phi$ and $\Delta\theta$. $\Delta_{\mathrm{m}}\rho$ is the maximum possible change of radius and ρ_{m} the maximum viewing radius allowed. Note that minimizing the movement cost alone would simply keep the view unchanged. This is compensated by the first two terms, which force the agent to change view, to gain new information while making correct prediction.

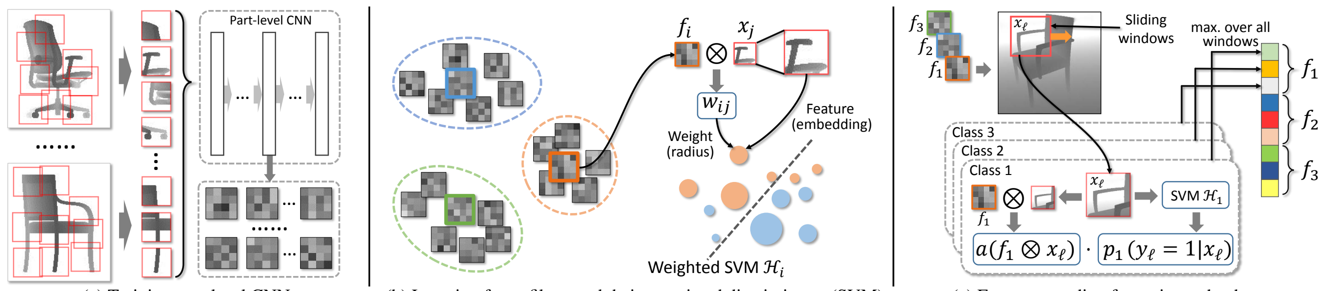
MV-RNN inference and inaccessible views handling. Starting from an initial view, the robot keeps performing depth acquisition and model inference, until the classification certainty is sufficiently high. At each time step, the acquired depth image is passed through MV-RNN to predict object class and regress the next best view. There are cases where some predicted NBVs are physically inaccessible. With our MV-RNN model, this physical restriction can be naturally handled by inputting a null feature ($c_t = \mathbf{0}$) into the view aggregation network, which means that the first-layer hidden units pass on the values of the last step: $h_t^{(1)} = h_{t-1}^{(1)}$. Since the second-layer hidden units are trained to avoid generating views which were selected before, our model would simply skip the inaccessible view and continue the observation with new views. Another practical issue is object occlusion, which often has severe adverse effect on model inference. We resolve this problem by enhancing the visual feature learning of CNN_1, which will be discussed in the next.

5 Visual feature learning enhancement

We propose two essential enhancements to the visual feature extraction. The first is a hierarchical organization of the shape collection to faciliate accurate, coarse-to-fine classification. We employ a cluster-and-classify scheme to discover fine-level classes in an unsupervised fashion, and enhance the visual feature learning to be discriminative against those classes. To deal with object occlusion, we introduce a part-level attention mechanism and learn occlusion-tolerant focus-driven features at each node of the hierarchy.

5.1 Hierarchical 3D shape classification

Starting from the root containing all 3D shapes, we construct a hierarchy of coarse-to-fine classifiers, to partition the shape collection into a hierarchical organization. At the root node, the coarsest-level



(a) Training part-level CNN. (b) Learning focus filters and their associated discriminator (SVM). (c) Feature encoding for an input depth.

Figure 6: *Pipeline of focus-driven feature encoding.*

classifier is trained over all models to classify them into their categories which are pre-defined in the database (e.g., the top-level semantic labels in ShapeNet). Specifically, we use MV-CNN due to its state-of-the-art performance. At each of the following nodes, an enhanced MV-CNN is trained using the set of shapes associated with that node, by fine-tuning the network of its parent node. These finer-level classifiers are *relaxed* such that one shape may be classified into more than one classes. Nodes that contain less than 50 shapes are set as leaf-nodes. At each leaf-node, an instance-level classifier is trained using every single shape as a class.

Unsupervised fine-grained class discovery. At each non-root node, we perform an unsupervised *cluster-and-classify* process to cluster the associated shapes into K overlapping groups, while training a relaxed classifier based on these overlapping groups. Specifically, we first perform clustering using the Gaussian Mixture Model (GMM), based on the initial features learned by the MV-CNN (output by CNN_1) of its parent node. For each shape S_i, GMM sets a probability vector $\mathbf{p}_i \in \mathbb{R}^K$ with $\mathbf{p}_i(k)$ indicating the probability of S_i belonging to cluster k. Instead of computing a hard partitioning over the shape set, we allow overlap among clusters by stipulating that a shape S_i belongs to cluster k if $\mathbf{p}_i(k) > p_o$, where p_o is referred to as *overlap probability*.

In the classification step, a relaxed MV-CNN classifier is trained based on the clustering result. Specifically, we train a (K+1)-way classifier, with a dummy class corresponding to *all* overlapping regions. A shape S_i which is clustered into k-th cluster with sufficient confidence, $\mathbf{p}_i(k) > p_c$ (*classify probability*), is used as a training datum with label k. All the remaining shapes are labeled to the dummy class. This training is efficient since it only amounts to fine-tuning the MV-CNN of current node based on its parent node. The refined features are effective in discriminating the above training data, leading to a better clustering for K clusters in the next iteration. The alternating clustering and classification repeats until the clusters become stable. If a shape is classified into the dummy class, we use the associated GMM to determine to which exact overlapping region it belongs, and duplicate it in each of the nodes corresponding to those overlapping clusters. Figure 5 visualizes the effect of feature enhancement by plotting the feature embedding of a set of shapes being clustered into five classes with overlap.

5.2 Focus-driven occlusion-tolerant features

From a given view of observation, the key to successfully recognizing an object in the presence of occlusion is to learn features that could robustly *spot* informative visible parts of the partially occluded object, and accurately *discriminate* shape class based on those parts. To this end, we introduce part-level attention to the depth feature encoding in terms of a given shape classification task. This results in focus-driven features which attend only to the parts which are both visible and characteristic, and base the classification on these focused parts instead of the entire depth image.

Our basic idea is to train a *part-level CNN* and use the learned mid-level filters as focuses to drive characteristic part discerning and part-based shape discrimination. The part-level CNN is trained at each node of the classification hierarchy we have constructed, based on the associated 3D shapes with their class labels. Figure 6 shows the pipeline of our focus-driven feature encoding.

Focus extraction. At a given node of the hierarchy, we collect the depth images rendered for the associated 3D shapes from all sampled views. From each depth image, we randomly extract a set of patches using selective search [Uijlings et al. 2013], representing various parts of the 3D shapes observed from a specific view. These patches are used to train a part-level CNN over the associated shape classes (Figure 6a). An empirical observation on this part-level CNN is that neurons of its hidden layers exhibit strong grouping pattern; a group of neurons may be especially sensitive to some specific part [Xiao et al. 2015]. Inspired by this, we extract the learned convolutional filters from a middle convolutional layer, and cluster them using self-tuning spectral clustering [Zelnik-Manor and Perona 2004]. The dissimilarity between filters is measured by the Earth Mover's Distance. The filters corresponding to the cluster centers, denoted as $\{f_i\}_{i=1}^F$, represent potential focuses to attend for part-sensitive feature encoding (Figure 6b).

Focus-driven feature encoding. We desire features encoding information for both part discerning and part-based discrimination. The former can be achieved by performing sliding-window convolution operations over the input depth image, using the learned focus filters. Parts which are relevant to the focus filters will receive high activations. A part-based feature can be constructed by concatenating the activation values corresponding to all the filters, after max pooling operations as in [Bansal et al. 2015]. To further endow the feature with discriminating power against the shape classes, we train a discriminative model for each focus filter and encode the classification information into the feature.

Suppose there are two classes at a given node of the hierarchy. For each focus filter in f_i, we train an *activation-weighted SVM* (Figure 6b, right) using all the patches extracted before, $\{(x_j, y_j)\}_{j=1}^n$, by minimizing the following weighted loss function:

$$L^{\mathrm{w}}(\sigma_i) := \sum_{j=1}^{n} w_{ij}\ell(y_j, \sigma_i(x_j)), \tag{8}$$

where σ_i is the SVM decision function. The weight w_{ij} takes the activation of patch x_j by the convolution filter f_i. To perform this convolution, the patch needs to be re-scaled into the filter size. The distance between two patches, used by SVM, is the Euclidean distance between their features extracted by the part-level CNN above. When there are more than two classes, the SVM can be replaced by multiple weighted one-vs-all SVMs plus a soft-max classifier, which produces a probability distribution over all classes.

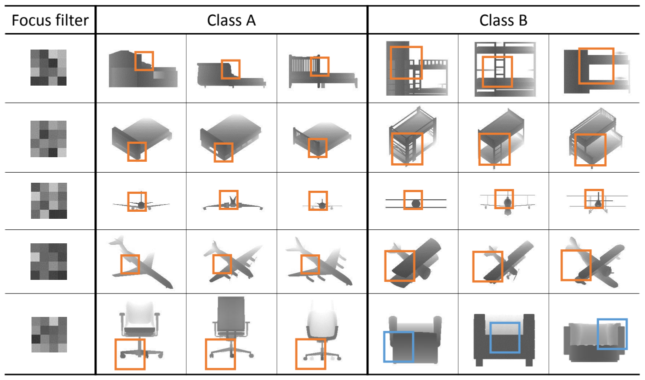


Figure 7: *In each row, we show a focus filter along with six highly activated patches (red boxes) classified into two classes by the associated SVM. The filter in the last row, being sensitive to wheels of swivel chairs, was not activated on sofas (blue indicates low activation). In such case, the two shape classes can still be correctly discriminated, based solely on activation values. This verifies that both activation and discrimination take effect in object recognition.*

The final feature combines both filter response and classification prediction. Specifically, given a depth image, we use each focus filter f_i to perform sliding-window operation over it. For each window patch x_l, we compute both convolution activation a and classification probability p_i over classes $k = 1, \ldots, K$. The feature entry corresponding to filter f_i and class k is:

$$u(f_i, k) := \max_l \left(a(x_l \otimes f_i) \cdot p_i(y_l = k|x_l) \right).$$

Consequently, the feature for a depth image is a $K \times F$ dimensional vector for K classes and F focus filters; see Figure 6(c).

Our focus-driven feature encoder is used to enhance the MV-RNN, by substituting its CNN_1 while re-training the CNN_2. In contrast to the features encoding the information of the entire image, our attention-based features focus on many distributed local parts, each of which may contribute to shape characterization. This increases the chance that the shape is recognized under partial occlusion. Figure 7 demonstrates several examples of learned focus filters, along with the classification of their strongly activated patches. The results show that the corresponding shapes can be correctly classified with various focuses. In practice, we used two levels of patch size in both focus learning and feature encoding, to attain scale-invariance.

6 Realizing online object identification

Coupling hierarchy and MV-RNN. We now describe how to use the classification hierarchy to guide both training and testing of the MV-RNN. The procedures for both are similar, as shown in Algorithm 1. Let us describe the training process for example. Given a training 3D shape from database, we "simulate" the active identification of it based on rendered depth images. Starting from the root node, we traverse the hierarchy for coarse-to-fine classification while performing MV-RNN inference for view estimation.

The depth image rendered from the current view is passed into MV-RNN for shape classification (Line 5). Here, the subnetworks for classification in MV-RNN take the feature encoder and CNN_2 trained for the current node (see the correspondingly colored subnetworks in Figure 8). The MV-RNN then proceeds to regress the NBV (Line 7). The above process is repeated until the classification is sufficiently certain, when the traverse moves down to the child node corresponding to the predicted class (Line 11). Note

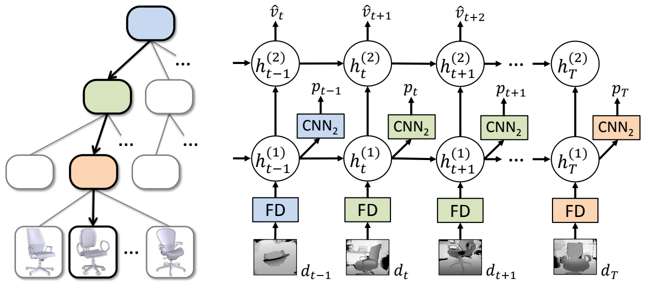


Figure 8: *MV-RNN training or testing guided by classification hierarchy. Note how the various time steps of MV-RNN utilize the focus-driven feature encoder (FD) and classifier (CNN_2) of the corresponding nodes being visited (shaded in consistent color).*

that not every node needs acquiring new depth images; if the classification at a node is sufficiently confident, given the ever-acquired ones, the traverse directly moves on. The MV-RNN is trained with the rewards calculated based on the classification (Line 5 and 9). The process of testing is the same, except that neither reward calculation nor network tuning is needed. For testing, the traverse stops at a leaf node and the corresponding instance-level classifier would return the identified shape (Line 13).

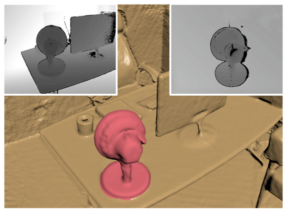
Algorithm 1: Hierarchy-guided MV-RNN training (testing)

```
Input  : The MV-RNN to be trained (tested): $\Psi$,
         the classification hierarchy: $\mathcal{T} = \{C_n\}$, and
         a shape $S$ (object $O$) used for training (testing).
Output: Identified 3D shape: $S^*$ (only for testing)
1  $v \leftarrow v_0$ ;                         // Initial view (randomly selected for training)
2  $D \leftarrow$ Depth($O$, $v$) ;             // Render (capture) depth image
3  $n \leftarrow$ RootNode($\mathcal{T}$) ;     // Set the current node to root
4  repeat
5      $\{k, h\} \leftarrow$ Classify($\Psi$, $C_n$, $D$) ;   // Calc. reward; train
6      while $h > h_t$ do                        // Entropy larger than threshold
7          $v \leftarrow$ RegressNBV($\Psi$, $D$) ;
8          $D \leftarrow$ Depth($O$, $v$) ;     // Render (capture) depth image
9          $\{k, h\} \leftarrow$ Classify($\Psi$, $C_n$, $D$) ; // Calc. reward; train
10     if $n$ is not a leaf node then
11         $n \leftarrow$ ChildNode($\mathcal{T}$, $n$, $k$) ;
12     else
13         $S^* \leftarrow$ ChildNodeShape($\mathcal{T}$, $n$, $k$) ;
14 until $n$ is a leaf node;
15 return $S^*$;
```

Online object identification. Our online object identification can be integrated into a standard pipeline of online scene reconstruction using a depth camera [Choi et al. 2015], to *automate* scene modeling based on 3D model retrieval [Salas-Moreno et al. 2012]. We base our method on the auto-scanning system in [Xu et al. 2015a] (running on a PR2 robot holding a Microsoft Kinect v2) for reconstructing a rough scene geometry. The main purpose of this rough reconstruction is (1) to facilitate 3D segmentation for object extraction and background removal, (2) to provide a basis for scene modeling by 3D model placement and, (3) to support the testing of physical constraints. These tasks do not require detailed scene geometry, so we disable the scan refinement phase in the original system. The object identification is based purely on depth images, but not on the reconstructed geometry which is too rough.

Based on the partial and rough scene reconstruction, we perform

3D object extraction through major planar surface (ground, walls, etc.) removal [Zhang et al. 2014] and 3D geometry-based foreground segmentation [Xu et al. 2015a]. Focusing on one extracted object, the robot performs guided scanning to incrementally acquire new depth images. We use the extracted object as a foreground mask to remove the background clutter in the depth image for better recognition (see the inserted figure). Note, however, background clutter removal does not resolve inter-object occlusion, which is handled by our feature encoding. Once the object is identified, the returned 3D model is placed into the scene through a view-based alignment, with the help of the view features extracted by MV-CNN, and a bounding box based re-scaling [Li et al. 2015a].



Based on the partially reconstructed scene geometry, we can examine the physical accessibility of a NBV candidate and skip it by feeding null input into MV-RNN as described in Section 4.3. Specifically, we test accessibility based on the partially reconstructed scene geometry. This is implemented with the collision detection package built on top of ROS (Robot Operating System) [ROS 2014]. If the accessibility test fails for 10 estimated NBVs consecutively, the recognition halts with a failure.

7 Implementation, results and evaluation

7.1 Parameters and statistics

Database and preprocessing. We have tested our method with two large-scale 3D shape datasets: (1) The ShapeNet dataset containing $57,452$ 3D shapes in 57 top-level object categories; (2) The Princeton ModelNet40 dataset [Wu et al. 2015] collecting $12,311$ models in 40 coarse-level classes. Each 3D model is rendered into a *basic set* of 2.5D depth images from 52 sampled views, serving as multi-view training data. The views are sampled in the spherical coordinate system around the shape center, with 5, 8 and 2 discretizing levels on azimuthal angle, polar angle and radius distance, respectively. The two distance levels are chosen as 1.5ℓ (near) and 3ℓ (far) with ℓ being the diagonal length of shape bounding box. All depth values are normalized into $[0, 255]$.

To tolerate camera shifting and tilting in real acquisition and meanwhile avoid overfitting, we perform camera jittering to augment the training data. Specifically, for each view, we introduce 5 randomly perturbed depth images through altering its viewing direction by an angle up to $\pm 10^\circ$ for near view and $\pm 5^\circ$ for far one. If the rendered model occupies less than 20% area in a perturbed depth image, we discard the image. Consequently, each 3D shape contributes an *enhanced set* of 260 training depth images in total.

Classification hierarchy. The construction of classification hierarchy is initialized by training the coarsest level classifier using the available top-level semantic classes. At each node in the following levels, a relaxed K-way MV-CNN classifier is trained using the enhanced view set of each shape. At a leaf node with less than 50 models, we train an instance-level classifier. Table 1 reports some statistics about the hierarchies built for the ShapeNet and ModelNet40 databases, respectively.

A key factor to the performance of object identification is the classification accuracy at top levels; early mistakes can lead to dramatically wrong result. The latter is in fact non-trivial to avoid in the incremental observation setting, since the initial view can be arbitrarily bad, making the top-level classification difficult. To mitigate this issue, we make two design choices. *Firstly*, when learning the relaxed classifier, we allow a larger overlap for higher level nodes by starting from a large overlap probability: $p_o = 0.3$, and raising it by 20% for every level down, until the maximum value of 0.6 is reached. The classify probability is fixed ($p_c = 0.8$) for all levels. *Secondly*, in both training and testing of MV-RNN (Algorithm 1), we set a higher entropy threshold h_t, used for judging the confidence of classification, for higher level nodes. This is to avoid mis-classification at high levels by forcing the robot to gather more views before making a decision. Specifically, we set $h_t = 0.98$ for the top level and decrease it by 5% for every level down until reaching the minimum value of 0.9.

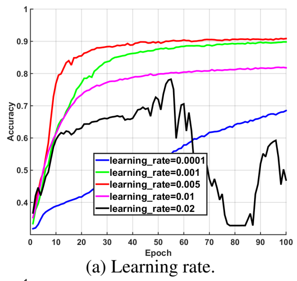
Database	#Model	K	$D_{\max}$	$D_{\min}$	D_{avg}
ShapeNet	$50,673$	20	5	3	3.73
ModelNet40	$12,311$	15	4	3	3.36

Table 1: *Statistics on the classification hierarchies built with two databases. K is the number of clusters/branches for each node in the intermediate levels. The rest three values are the maximum, minimum and averaged depth of hierarchy, respectively.*

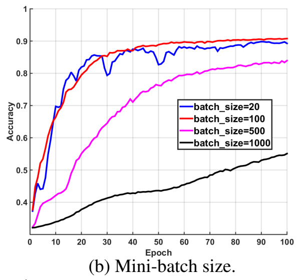
MV-CNN and focus-driven features. The MV-CNN of each node of the hierarchy is implemented with the AlexNet architecture [Krizhevsky et al. 2012] which contains 5 convolutional layers (conv1~5) and 3 fully connected layers (fc6~8) followed by a softmax layer. Same parameter settings as the original paper are used for the various layers. For view aggregation, we add a max pooling layer between fc7 and fc8. Therefore, CNN_1 corresponds to the layers before fc7 and CNN_2 to those after fc8. The fine-tuning in hierarchy construction is conducted over con5 and fc6~8. In focus-driven feature encoding, the part-level CNN adopts the LeNet [LeCun et al. 1998] with patches in sizes 40 and 80. We then extract 40 focus filters from the conv4 layer of LeNet. The feature size is $40K$ for the K-way classification at a node of hierarchy.

MV-RNN. Our MV-RNN takes a focus-driven feature as input, and outputs a vector of three view parameters. The unit size for each of the hidden layers is 1024. We use the ReLU activation functions for all hidden layers, $f(x) = max(0, x)$, for fast training. There are about $315M$ parameters in total being optimized with stochastic gradient descent. This does not include those of the subnetworks for feature encoding and classification, which have been pre-trained outside MV-RNN. The hyper-parameters used in training MV-RNN are critical for a successful policy learning. Figure 9 demonstrates the evaluations of four key hyper-parameters used in MV-RNN, which helped us in choosing their values. The hyper-parameters we evaluate include: 1) the learning rate, which is the step size in gradient descent; 2) the mini-batch size which is the number of input training data for each training cycle (iteration); 3) the view variance, i.e., the standard deviation in the Monte Carlo view sampling (see Section 4.3); 4) the scaling factor of reward used to stabilize the learning process. For each hyper-parameter, we plot the recognition accuracy over epoch (number of training cycles), under different parameter settings.

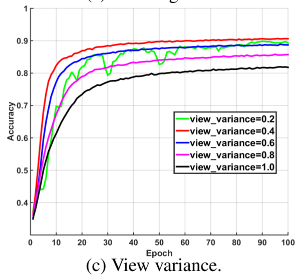
Larger learning rates lead to faster convergence, however, an overly large value may cause stability issue (Figure 9a). We use 0.008 in our implementation. A large mini-batch size generally stabilizes the training but also slows down the convergence (Figure 9b); we choose 128. The view variance indicates how large a range the view sampling searches around the expectation during MV-RNN training (see Equation (2)). A large range would increase the chance of finding a better view, but scarify the training speed at the same time (Figure 9c). We set the variance to 0.3. From Figure 9(d), it is clear that scaling down the reward in each training cycle helps



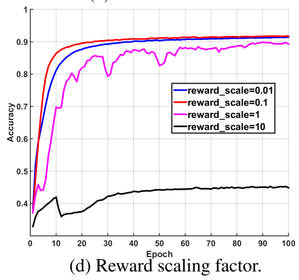
(a) Learning rate.



(b) Mini-batch size.



(c) View variance.



(d) Reward scaling factor.

Figure 9: *Evaluating four key hyper-parameters used in training MV-RNN. For each hyper-parameter, difference values are evaluated by plotting the recognition accuracy over epoch.*

stabilize the training, for which we use 0.5. As such, the selections of these hyper-parameters were carried out experimentally and once appropriate settings have been found, we adhere to them.

Complexity and timing. The complexity of hierarchy construction is $O(N \log_K N)$, with N being the total number of models and K the number of clusters for each node. Both MV-RNN learning and inference can be seen as tree search with time complexity $O(\log_K N)$. Table 2 reports the timings for the various components running on a workstation with an Intel® Xeon E5-2670 @ 2.30GHz $\times$ 24, 64GB RAM and an Nvidia® Quadro M5000 graphics card.

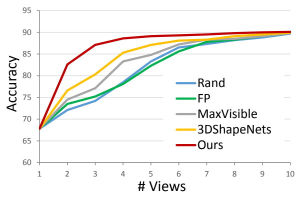
Database	Hierarchy	Focus	MV-RNN train	MV-RNN test
ShapeNet	47 hr.	39 hr.	49 hr.	0.1 sec.
ModelNet40	23 hr.	21 hr.	22 hr.	0.1 sec.

Table 2: *Timings for hierarchy construction, focus-driven feature learning, MV-RNN training and testing are listed for the two databases. MV-RNN testing is for a single NBV prediction.*

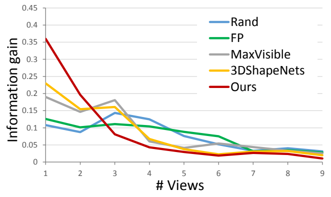
7.2 Evaluation and comparison

NBV estimation. To evaluate the performance of NBV estimation, we compare our MV-RNN against four alternative methods. We implemented two baseline view planners, i.e., *Random* which selects the next view randomly among candidates and, *Farthest* which chooses the view which is farthest away from the previous one (distance measured by arc-length). We also compare to the recent state-of-the-art NBV technique proposed along with the *3DShapeNets* [Wu et al. 2015], as well as the *MaxVisible*, a baseline built on top of their work. The latter estimates NBV as the view with the highest new visibility based on the volumetric shape representation of 3DShapeNets. The tests are conducted on all models in ModelNet40 with the task of classification over the 40 classes. For a fair comparison, each method predicts their own NBVs, but uniformly uses MV-CNN for multi-view shape classification.

Figure 10(a) plots the recognition rate over the number of views. Our method obtains the steepest accuracy increase. To further compare the efficiency of the NBV predictions, we plot in (b) the information gain of different views, which is the decrease of Shannon entropy of the classification probability distribution. Compared to the alternatives, our NBV sequences attain larger information gain in early steps, demonstrating their effectiveness. In addition, our NBV estimation is much faster than 3DShapeNets. Besides the view-based representation we adopt and the fast testing of RNN, an important reason is that our method is specifically targeted to and fully trained for NBV problem, while 3DShapeNets, proposed as a generic shape representation, is not particularly optimized for it. We also tested the effect of movement cost incorporation for our method. The result shows that taking movement cost into account saves 26% moving distance on average, when the same classification accuracy is reached for both cases.



(a)

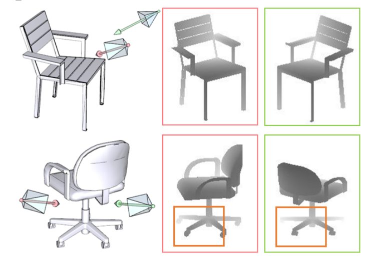


(b)

Figure 10: *Comparison of the performance of NBV selection by five methods (Random, Farthest, MaxVisible, 3DShapeNets [Wu et al. 2015] and ours). (a): Classification accuracy over the number of views. (b): Information gain of different views.*

NBV estimation under occlusion. The advantage of our method shall be best demonstrated in real scenarios where the targeted objects are partially occluded due to scene clutter. We evaluate this on a *benchmark dataset* containing both synthetic and real test scenes, with inter-object occlusion (see a summary of the data collection in the supplemental material). The targeted objects in all test scenes are labeled with a ground-truth finest-level category (defined in ShapeNet) manually. The classification accuracy is measured against the finest-level categories of ShapeNet. Our method is trained with ShapeNet dataset while using only the top-level classes as supervision. For comparison, we show in Figure 11 results of our method with focus-driven and MV-CNN (using CNN_1) feature extraction, respectively. Results of Random and Farthest with MV-CNN are also provided in the same plot. The results show that our MV-RNN, enhanced by focus-driven depth features, achieves significantly higher accuracy on both synthetic and real data, demonstrating its strong resilience against occlusion.

Handling inaccessible views. In real scenarios, predicted views are sometimes inaccessible due to various physical constraints. In this case, the agent has to estimate a second best view (referred to as SNBV). A good SNBV should be 1) informative, delivering both high accuracy and large information gain and, 2) distant away from the NBV to have a better chance to jump out of the inaccessible region. Therefore, the quality of the SNBV at time step t, denoted as $\hat{v}'_t$, can be measured with respect to the NBV $\hat{v}_t$: $\eta(\hat{v}'_t) = H_t(p'_t, \bar{p}) I_t(p'_t, p_t) d_{\text{arc}}(\hat{v}'_t, \hat{v}_t)$, where H_t and I_t are defined similarly as in Equation (5-6). d_{arc} is the normalized arc-length distance. Table 3 compares the quality of the SNBV selected at different time steps, by our method (with and w/o movement cost), 3DShapeNets and two baselines, with the same settings as in Figure 10. All results are obtained on the classification



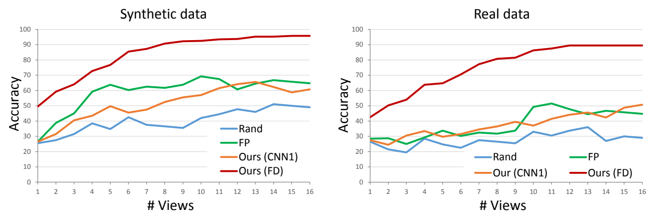


Figure 11: *Comparison of performance of NBV selection on synthetic and real data with occlusion. Our method with focus-driven feature encoding (FD) delivers significantly better accuracy.*

task on ModelNet40 and averaged over all shapes tested. The inserted figure shows an interesting phenomenon of our SNBV selection. For shapes possessing symmetries, the SNBV (green arrow) tends to be nearly symmetric to the NBV (red arrow), with respect to the symmetry plane/axis. The corresponding depth images, as well as the focused parts, are shown to the right.

Time step	Ours (w/)	Ours (w/o)	3DShapeNets	Rand.	Far.
$t = 1$	0.32	0.37	0.21	0.07	0.18
$t = 2$	0.27	0.36	0.19	0.10	0.13
$t = 3$	0.19	0.29	0.16	0.06	0.13

Table 3: *Comparison of second next-best-view estimation when the NBV is inaccessible, at different time steps.*

Robustness against view perturbation. In real settings, views can hardly be taken precisely. One main reason is that accurate estimation of object center is difficult due to imperfect object location and segmentation. There are two mechanisms in our method designed for handling imperfect view observation: (1) allowing variable viewing distance in NBV estimation and (2) tolerating camera shifting by view jittering. In this experiment, we evaluate the effect of these two factors on the robustness against view perturbation. To do so, we introduce view perturbations during the testing phase of virtual identification, based on camera jittering similar to training data generation in Section 7.1. We evaluate the performance of identification using L2 distance between the (focus-driven) features of the query model and the identified one. The smaller the distance, the better the identification performs. In Figure 12, we plot the histogram of the number of models (drawn from ShapeNet) for various feature distance intervals. The result demonstrates that our method gains robustness with both view jittering and distance sampling.

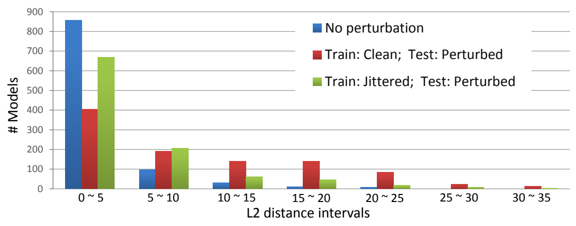


Figure 12: *Plots of #models for different intervals of feature distance between query and identified. When perturbation is added in test, the performance (red bars) degrades dramatically compared to the ideal case (blue). The performance is significantly improved when view jittering is added in training data (green).*

Evaluation of focus-driven features. We first evaluate the performance of our focus-driven feature encoding for varying patch size and focus filter count. The task is fine-grained classification,

Dataset	#M	#C	Details (100 models per category)
Airplane	500	5	jet, straight-wing, fighter, delta-wing, swept-wing
Bed	500	5	single, bunk, king-size, couch, hammock
Chair	500	5	armchair, folding, swivel, four-leg, sofa
Table	500	5	round, slice, desk, kitchen, counter

Table 4: *Statistics of four datasets with fine-grained subcategories, collected from ShapeNet [Su et al. 2015c].*

conducted over four fine-grained datasets collected from ShapeNet, each with five ground-truth subcategories. Table 4 summarizes the four datasets. Each dataset is split into training (80%) and testing (20%) subsets. Our feature learning is conducted on *full* depth images *without* occlusion. We then train a soft-max classifier using the training data with focus-driven feature encoding. To stress test our features for partial occlusion, we add synthetic occlusion to every single-view depth image rendered for testing shapes by randomly putting a square mask of 0.15~0.45 of the input image size. Figure 13 reports the average classification accuracy for different feature learning settings. A noteworthy fact is although the feature learning is performed on non-occluded inputs, the final features work quite well on occluded ones.

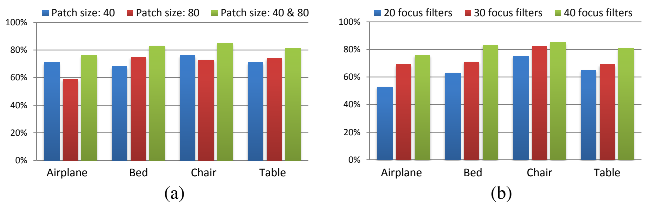


Figure 13: *Accuracy of fine-grained classification on four datasets with varying patch sizes (a) and focus filter counts (b). The classification is performed on single-view images with partial occlusion.*

In Figure 14, we compare our focus-driven method against four alternatives including (1) CNN feature extraction for entire images (CNN), (2) the CNN-based two-level attention model (CNN-AM) proposed in [Xiao et al. 2015], (3) the Deep Sliding Shapes by Song and Xiao [2016] and, (4) Region-based CNN for depth input (Depth R-CNN) from [Gupta et al. 2015]. The last two methods are trained for object detection and recognition from a scene, without part-level feature encoding. CNN-AM achieves object detection based on part-level attention, but without activation-weighted part discrimination. Our method extracts focus-driven features encoding more informative discriminative model with activation-weighted SVM. For the four datasets, we plot the fine-grained classification accuracy over varying sizes of occlusion (up to 0.45 of image size). The results demonstrate the superiority of our method.

Comparison of instance-level 3D shape classification. For instance-level classification, we compare our hierarchical method against the classical Vocabulary Tree (VT) [Nister and Stewenius 2006], which was originally proposed for 2D images and then extended to 3D shapes by Atanasov et al. [2014]. Note, however, VT is conceptually different from our method in that it utilizes a classification tree to organize local features, instead of data instances like ours, to compute a global feature encoding. For reference, the results of MV-CNN and 3DShapeNets are also reported. For a fair comparison, our hierarchical classifiers adopt MV-CNN. The experiment was conducted on the four fine-grained datasets. While our method is trained with only the top-level labels, all the other methods are trained as an instance-level classifier with each shape as a distinct class. Figure 15 plots the retrieval results (precision-recall curves) for each dataset separately. Our method, albeit trained only

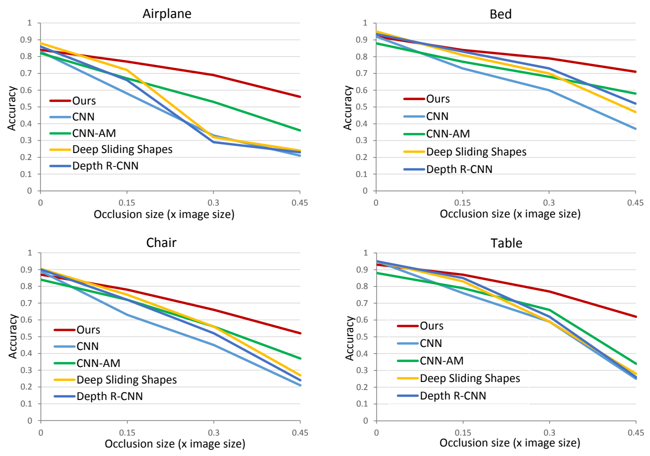


Figure 14: *Comparison of fine-grained classification based on* single-view *depth images with varying degrees of partial occlusion.*

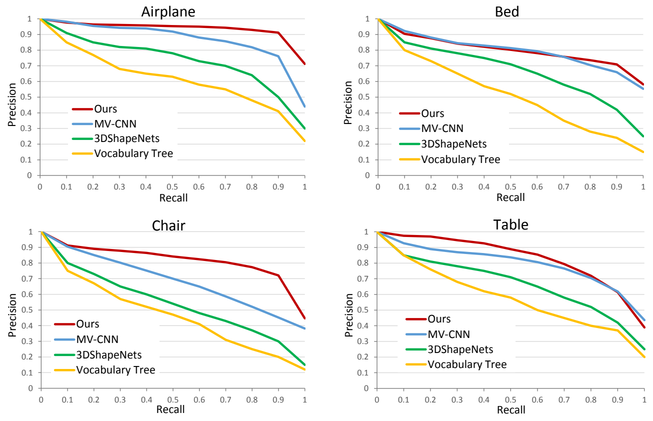


Figure 15: *Comparison of instance-level classification on the four datasets in Table 4. Our method performs the best for the Airplane and Chair datasets and comparable to MV-CNN for the other two.*

with weak supervision, achieves better results than the alternatives, due to the efficient hierarchical organization of shape collection.

Visualization of attentions. To visually investigate the behavior of our attention model, in Figure 16, we visualize the attended regions on both acquired depth images and retrieved shape surfaces, for the identification of several real objects. Along with the sequence of depth images, we show the attended parts during the process of coarse-to-fine classification. The surface-based attention plot is done by back-projecting the focused parts in depth images onto the surface of the finally identified shape. The representative shapes associated with each node, during hierarchy traversing, are also shown along the sequence, to provide a visual correlation between the attended regions and the current classification task.

Besides robot-operated object identification in real scenes, we also test our algorithm on the newly released excellent dataset of RGBD object scans [Choi et al. 2016]. This dataset provides for each object a round scan of RGBD video captured by humans. Since the depth frames do not come with camera transformations, we run SLAM-based depth fusion [Nießner et al. 2013] with each depth video and record the camera transformation for each frame. This parameterized depth video serves as our viewing space, from which our method can acquire depth observations: Given a view parameter output by our NBV estimator, a depth image can be retrieved from the closest view in the space. The results show that our method can correctly recognize the objects with plausibly planned views. More results can be found in the supplemental material.

Results on real scenes. We have tested our method in scanning and modeling three real scenes including an office (Figure 1), an apartment and a cybercafe (Figure 17), based on the ShapeNet database. For each scene, we show two corners containing several objects recognized and retrieved by our system. Among the segmented objects, our system achieves a recognition rate of 42%, for the three scenes on average. Although we have devised special mechanisms to deal with occlusion and physical constraint, the recognition rate in real scenes is still relatively low mainly due to two reasons. First, our method relies on 3D segmentation for object proposal and background removal in depth images. However, correct object segmentation itself is quite challenging when scene is cluttered. Second, our current implementation does not consider contextual information among scene objects. The latter has been shown to be quite useful in resolving ambiguities (e.g. the failure case in Figure 17) in 3D scene modeling in several previous works [Fisher et al. 2012; Xu et al. 2013; Chen et al. 2014].

8 Discussion, limitations, and future work

We have presented a technique for estimating next-best-views while increasing the confidence in identifying an object during the course of depth acquisition. The identified 3D models are retrieved from shape database and then placed into the partially reconstructed scene. Our NBV technique is data-driven; it learns a large set of shapes and expects to identify (re-identify) one of them, to match the object of interest, by an efficient series of scans. To model the sequential nature of view planning, we propose a recurrent 3D attention model which simultaneously optimizes for observation views, focused parts and classification accuracy. Results demonstrate that our method is quite effective for object identification in real scenarios with occlusion and inaccessibility issues.

Coupling of hierarchy and MV-RNN. In our 3D attention model, MV-RNN is bound with the classification hierarchy, for both training and testing. Such coupling conforms well to our conceptual design of object identification with progressive, coarse-to-fine perceptions. Essentially, the hierarchy guides the coarse-to-fine search, based on an efficient organization of the 3D shape database. MV-RNN learns these "guiding rules" and compiles them into a principled viewing policy. A potential issue with this coupling, however, is the trained MV-RNN may not work an stand-alone NBV estimator independent on the hierarchy. This is because the training of MV-RNN is highly task-driven. This could be a restriction: when the hierarchy is significantly updated with newly added shape categories, the MV-RNN should be re-trained. An interesting future direction is how to minimize the re-training through, e.g., incremental update of RNN, or how to achieve transductive learning with RNN.

Alternative occlusion handling. A straightforward alternative is to train a classification model on data with synthetic occlusion. While this approach might work well when sufficient training data is supplied to a deep learning model, we opted for a more principled solution by introducing part-level attention into feature encoding. Besides the much less training data required and faster patch-level processing, we believe our method is more versatile in handling intra-class shape variations since we only look at characteristic parts but not their exact spatial positions. However, it is worth investigating the utilization of contextual information among

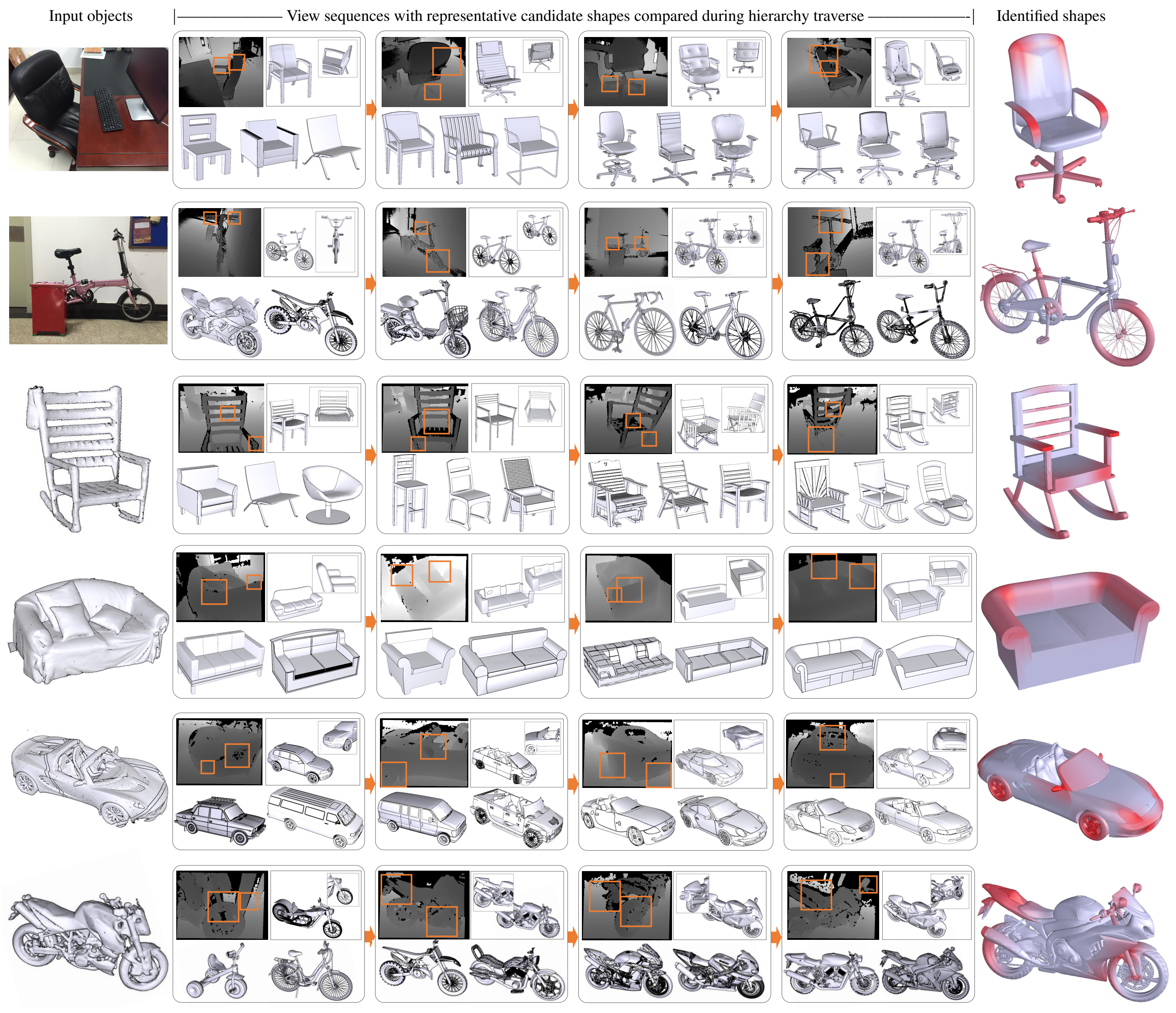

Input objects
View sequences with representative candidate shapes compared during hierarchy traverse
Identified shapes

Figure 16: *Visualization of identification process and attention (both on acquired depth images and identified shapes) for six real objects.*

parts [Doersch et al. 2015], for more discriminant feature encoding.

Integrating two levels of attention. In our current framework, the two levels of attention are employed separately, in NBV estimation and view feature extraction, respectively. A natural question is whether it is possible to integrate the two in one optimization framework, leading to an attention model that selects regions on shape surface by simultaneous view and fixation estimation.

Instance-level classification. The instance-level classifiers at leaf nodes are actually trained with a small number (less than 50 in our implementation) of 3D shapes. This may potentially lead to overfitting. However, this is acceptable for our case since the task at leaf nodes is identification within a candidate set, which is actually not expected to generalize. If the object of interest does not appear in the shape set of the reached leaf node, the instance-level classifier would still return the most similar shape anyways.

Limitations. The main limitations of our method include:

- *Recognizable objects*: The classes of recognizable objects are confined to the 3D model database. For objects which are unseen in the database, our system would fail to recognize. Also, since our classification relies only on depth cues, it can not recognize objects without distinct shape characteristics. A failure case can be found in Figure 17, where the rolled-up quilt on the bed is mistakenly recognized as a CD player. Our method does not work well for tiny objects (e.g. a mouse) either, since the small depth range makes it hard to recognize.

- *Recovering from mistakes*: Although special considerations have been made (Section 7), early mis-classification is still unavoidable. An early mis-classification is fatal to our method since we currently do not support backtracking during hierarchy traversing. When doing backtracking in the hierarchy, the MV-RNN inference should be simultaneously rolled back for a corresponding number of time steps. This requires storing the internal states of all past steps in the history.

- *Background clutter*: This problem is orthogonal to partial occlusion. In fact, background clutter has adverse effect on

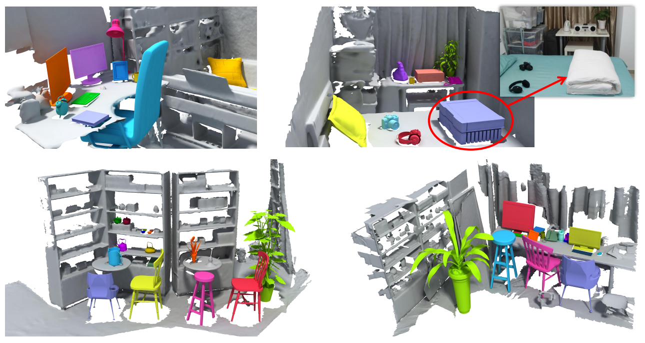


Figure 17: *Online modeling results for two indoor scenes (top and bottom rows). Note the implausible model (circled) retrieved for the rolled-up quilt, without contextual information.*

focus-driven feature encoding by distracting its attentions. Currently, we remove background based on the 3D geometry-based object segmentation. A more self-contained solution is to consider background during training, by generating training data with synthetic background clutter.

Future work. Besides addressing the above issues and limitations, we would also like to explore the potential of our technique for RGB cameras with no depth. We believe such extension is straightforward, which amounts to training shape classifiers using rendered RGB images [Su et al. 2015b]. Incorporating RGB input, on the other hand, would lead to another interesting direction of multi-modal object recognition, by jointly utilizing 3D shape and 2D image databases. Indeed, joint shape analysis based on 2D and 3D data is receiving much attention lately [Li et al. 2015b; Hueting et al. 2015]. Another interesting direction is to let NBV prediction assist segmenting an object of interest against its background directly in multi-view depth images, for which intelligently selected views are necessary to make the task easier.

Acknowledgements

We thank the anonymous reviewers for their valuable comments and suggestions. This work was supported in part by NSFC (61572507, 61532003, 61622212, 61522213, 61232011), National 973 Program (2015CB352501, 2014CB360503), Guangdong Science and Technology Program (2014B050502009, 2014TX01X033, 2015A030312015, 2016A050503036) and Shenzhen Innovation Program (JCYJ20151015151249564).

References

ATANASOV, N., SANKARAN, B., NY, J. L., PAPPAS, G. J., AND DANIILIDIS, K. 2014. Nonmyopic view planning for active object classification and pose estimation. *IEEE Trans. on Robotics 30*, 5, 1078–1090.

BA, J., MNIH, V., AND KAVUKCUOGLU, K. 2014. Multiple object recognition with visual attention. *arXiv preprint arXiv:1412.7755*.

BANSAL, A., SHRIVASTAVA, A., DOERSCH, C., AND GUPTA, A. 2015. Mid-level elements for object detection. *arXiv preprint arXiv:1504.07284*.

BART, E., PORTEOUS, I., PERONA, P., AND WELLING, M. 2008. Unsupervised learning of visual taxonomies. In *Proc. CVPR*, IEEE, 1–8.

CHEN, K., LAI, Y.-K., WU, Y.-X., MARTIN, R., AND HU, S.-M. 2014. Automatic semantic modeling of indoor scenes from low-quality rgb-d data using contextual information. *ACM Trans. on Graph. (SIGGRAPH Asia) 33*, 6, 208:1–208:15.

CHOI, S., ZHOU, Q.-Y., AND KOLTUN, V. 2015. Robust reconstruction of indoor scenes. In *Proc. CVPR*, 5556–5565.

CHOI, S., ZHOU, Q.-Y., MILLER, S., AND KOLTUN, V. 2016. A large dataset of object scans. *arXiv:1602.02481*.

CORBETTA, M., AND SHULMAN, G. L. 2002. Control of goal-directed and stimulus-driven attention in the brain. *Nature Reviews Neuroscience 3*, 201–215.

DOERSCH, C., GUPTA, A., AND EFROS, A. A. 2015. Unsupervised visual representation learning by context prediction. In *Proc. ICCV*, 1422–1430.

FISHER, M., RITCHIE, D., SAVVA, M., FUNKHOUSER, T., AND HANRAHAN, P. 2012. Example-based synthesis of 3D object arrangements. *ACM Trans. on Graph. (SIGGRAPH Asia) 31*, 6, 135:1–135:11.

GAO, T., AND KOLLER, D. 2011. Discriminative learning of relaxed hierarchy for large-scale visual recognition. In *Proc. ICCV*, 2072–2079.

GUPTA, S., ARBELÁEZ, P., GIRSHICK, R., AND MALIK, J. 2015. Aligning 3d models to RGB-D images of cluttered scenes. In *Proc. CVPR*, 4731–4740.

HAQUE, A., ALAHI, A., AND FEI-FEI, L. 2016. Recurrent attention models for depth-based person identification. In *Proc. CVPR*.

HOCHREITER, S., AND SCHMIDHUBER, J. 1997. Long short-term memory. *Neural computation 9*, 8, 1735–1780.

HUANG, Q.-X., SU, H., AND GUIBAS, L. 2013. Fine-grained semi-supervised labeling of large shape collections. *ACM Trans. on Graph. 32*, 6, 190:1–190:10.

HUETING, M., OVSJANIKOV, M., AND MITRA, N. J. 2015. Crosslink: joint understanding of image and 3d model collections through shape and camera pose variations. *ACM Trans. on Graph. 34*, 6, 233.

KLEIMAN, Y., VAN KAICK, O., SORKINE-HORNUNG, O., AND COHEN-OR, D. 2015. SHED: hape edit distance for fine-grained shape similarity. *ACM Trans. on Graph. 34*, 6, 235:1–235:14.

KRAUSE, J., JIN, H., YANG, J., AND FEI-FEI, L. 2015. Fine-grained recognition without part annotations. In *Proc. CVPR*, 5546–5555.

KRIZHEVSKY, A., SUTSKEVER, I., AND HINTON, G. E. 2012. ImageNet classification with deep convolutional neural networks. In *Proc. NIPS*, 1097–1105.

LECUN, Y., BOTTOU, L., BENGIO, Y., AND HAFFNER, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE 86*, 11, 2278–2324.

LI, L.-J., WANG, C., LIM, Y., BLEI, D. M., AND FEI-FEI, L. 2010. Building and using a semantivisual image hierarchy. In *Proc. CVPR*, IEEE, 3336–3343.

LI, Y., DAI, A., GUIBAS, L., AND NIESSNER, M. 2015. Database-assisted object retrieval for real-time 3D reconstruction. *Computer Graphics Forum (Eurographics) 34*, 2.

LI, Y., SU, H., QI, C. R., FISH, N., COHEN-OR, D., AND GUIBAS, L. J. 2015. Joint embeddings of shapes and images via cnn image purification. *ACM Trans. on Graph. 34*, 6, 234.

MNIH, V., HEESS, N., GRAVES, A., ET AL. 2014. Recurrent models of visual attention. In *Proc. NIPS*, 2204–2212.

NEWCOMBE, R. A., DAVISON, A. J., IZADI, S., KOHLI, P., HILLIGES, O., SHOTTON, J., MOLYNEAUX, D., HODGES, S., KIM, D., AND FITZGIBBON, A. 2011. KinectFusion: Real-time dense surface mapping and tracking. In *Proc. IEEE Int. Symp. on Mixed and Augmented Reality*, 127–136.

NIESSNER, M., ZOLLHÖFER, M., IZADI, S., AND STAMMINGER, M. 2013. Real-time 3D reconstruction at scale using voxel hashing. *ACM Trans. on Graph. (SIGGRAPH Asia) 32*, 6, 169:1–169:11.

NISTER, D., AND STEWENIUS, H. 2006. Scalable recognition with a vocabulary tree. In *Proc. CVPR*, 2161–2168.

ROS, 2014. ROS Wiki. http://wiki.ros.org/.

SALAS-MORENO, R. F., NEWCOMBE, R. A., STRASDAT, H., KELLY, P. H. J., AND DAVISON, A. J. 2012. SLAM++: Simultaneous localisation and mapping at the level of objects. In *CVPR*, 1352–1359.

SONG, S., AND XIAO, J. 2016. Deep sliding shapes for amodal 3d object detection in rgb-d images. In *Proc. CVPR*.

SU, H., MAJI, S., KALOGERAKIS, E., AND LEARNED-MILLER, E. 2015. Multi-view convolutional neural networks for 3D shape recognition. In *Proc. ICCV*.

SU, H., QI, C. R., LI, Y., AND GUIBAS, L. 2015. Render for CNN: Viewpoint estimation in images using cnns trained with rendered 3d model views. In *Proc. ICCV*.

SU, H., SAVVA, M., YI, L., CHANG, A. X., SONG, S., YU, F., LI, Z., XIAO, J., HUANG, Q., SAVARESE, S., FUNKHOUSER, T., HANRAHAN, P., AND GUIBAS, L. J. 2015. ShapeNet: An information-rich 3d model repository. http://www.shapenet.org/.

UIJLINGS, J. R., VAN DE SANDE, K. E., GEVERS, T., AND SMEULDERS, A. W. 2013. Selective search for object recognition. *Int. J. Computer Vision. 104*, 2, 154–171.

VALENTIN, J., VINEET, V., CHENG, M.-M., KIM, D., SHOTTON, J., KOHLI, P., NIESSNER, M., CRIMINISI, A., IZADI, S., AND TORR, P. 2015. SemanticPaint: Interactive 3D labeling and learning at your finger tips. *ACM Trans. on Graph. 34*, 5.

WILLIAMS, R. J. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning 8*, 3-4, 229–256.

WU, S., SUN, W., LONG, P., HUANG, H., COHEN-OR, D., GONG, M., DEUSSEN, O., AND CHEN, B. 2014. Quality-driven poisson-guided autoscanning. *ACM Trans. on Graph. (SIGGRAPH Asia) 33*, 6, 203:1–203:12.

WU, Z., SONG, S., KHOSLA, A., YU, F., ZHANG, L., TANG, X., AND XIAO, J. 2015. 3D ShapeNets: A deep representation for volumetric shapes. In *Proc. CVPR*, 1912–1920.

XIAO, T., XU, Y., YANG, K., ZHANG, J., PENG, Y., AND ZHANG, Z. 2015. The application of two-level attention models in deep convolutional neural network for fine-grained image classification. In *Proc. CVPR*, 842–850.

XU, K., CHEN, K., FU, H., SUN, W.-L., AND HU, S.-M. 2013. Sketch2Scene: Sketch-based co-retrieval and co-placement of 3D models. *ACM Trans. on Graph. (SIGGRAPH) 32*, 4, 123:1–123:10.

XU, K., HUANG, H., SHI, Y., LI, H., LONG, P., CAICHEN, J., SUN, W., AND CHEN, B. 2015. Autoscanning for coupled scene reconstruction and proactive object analysis. *ACM Trans. on Graph. 34*, 6, 177:1–177:14.

XU, K., BA, J., KIROS, R., COURVILLE, A., SALAKHUTDINOV, R., ZEMEL, R., AND BENGIO, Y. 2015. Show, attend and tell: Neural image caption generation with visual attention. *arXiv preprint arXiv:1502.03044*.

ZELNIK-MANOR, L., AND PERONA, P. 2004. Self-tuning spectral clustering. In *Proc. NIPS*, 1601–1608.

ZHANG, Y., XU, W., TONG, Y., AND ZHOU, K. 2014. Online structure analysis for real-time indoor scene reconstruction. *ACM Trans. on Graph. 34*, 5, 159:1–159:12.