Shape and Symmetry Induction for 3D Objects

Shubham Tulsiani[1], Abhishek Kar[1], Qixing Huang[2], João Carreira[1] and Jitendra Malik[1]
[1]University of California, Berkeley [2]Toyota Technological Institute at Chicago
[1]{shubhtuls, akar, carreira, malik}@eecs.berkeley.edu [2]huangqx@ttic.edu

Abstract

Actions as simple as grasping an object or navigating around it require a rich understanding of that object's 3D shape from a given viewpoint. In this paper we repurpose powerful learning machinery, originally developed for object classification, to discover image cues relevant for recovering the 3D shape of potentially unfamiliar objects. We cast the problem as one of local prediction of surface normals and global detection of 3D reflection symmetry planes, which open the door for extrapolating occluded surfaces from visible ones. We demonstrate that our method is able to recover accurate 3D shape information for classes of objects it was not trained on, in both synthetic and real images.

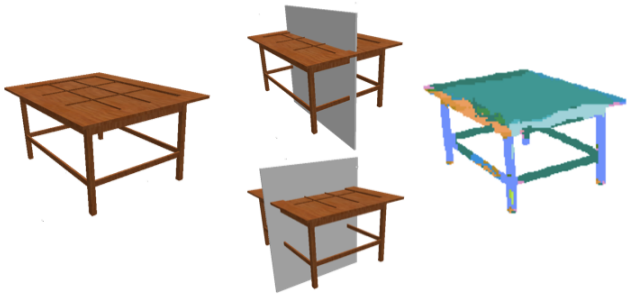

Figure 1: Given a single image of a novel object, our model induces a pixel-wise labeling of its surface normals (right) and predicts the orientations of all 3D planes of reflection symmetry (center).

> "What specifies an object are invariants that are themselves 'formless'"
>
> *J.J. Gibson*

1. Introduction

In this paper we develop a method for understanding the 3D shape of an unfamiliar object from a single image. Our method recovers a 2.5D shape representation by densely labeling normals of object surfaces visible in the image. We target the remaining 0.5D – the shape of occluded surfaces – by inferring shape self-similarities. As one small step in this direction, we introduce the task of detecting the orientation of any planes of reflection symmetry in the 3D object shape.

Recovering 3D object shape from a single image is clearly an ill-posed problem and requires assumptions to be made about the shape. The problem of reconstructing familiar categories has seen some success, but there strong 3D priors can be learned from training data [6, 20]. The problem of reconstructing shapes for previously unseen categories is more subtle and coming up with the right priors seems critical, in particular for recovering occluded surfaces. We pursue patterns of self-similarity in 3D shape given just the image, hoping these will allow filling in occluded geometry with carefully placed copies of visible geometry. We define an entry-level version of the problem: detecting the 3D orientation of planes of reflection symmetry.

Both components of our approach rely heavily on learning large nonlinear classifiers end-to-end, which has become an effective solution to many vision problems, but not yet for 3D object shape reconstruction from a single image. This is due, in part, to technical and logistical difficulties involved in creating large datasets of images of objects with aligned 3D shapes [48, 5].

In this work we leverage synthetic data, and resort to new large-scale shape collections where ground truth symmetry planes and surface orientations can be accurately computed. We then render these models and use the images paired with the symmetry and normal labels to learn Convolutional Neural Network (CNN) [13, 27] based systems for symmetry prediction (Section 3) and normal estimation (Section 4). We show qualitative results on real images and empirically demonstrate the ability of our models to induce these predictions accurately for novel objects (Section 5).

2. Related Work

Recovering an object's surface geometry from a single image is an ill-posed problem in general – e.g. a same image can be caused by different configurations of surface geometry, reflectance and lighting conditions. This problem has been studied from many different perspectives, which we will roughly divide here into physics-based and predictive shape inference. These two paradigms, together with prior work on symmetry detection and learning from CAD model datasets are briefly summarized below.

Physics-based Shape Inference Given an image, the preference for any particular 3D shape depends strongly on the type of priors imposed. Physics-based approaches, such as early shape from shading techniques [18], aimed to optimize shape using variational formulations with regularizers that encoded strong assumptions about albedo and illumination. Modern approaches such as SIRFS [3] extend these by using richer priors and additionally reasoning over reflectances in the solution space.

Predictive Shape Inference. The other common paradigm for shape inference is through supervised learning techniques that leverage training data to boost inference results. Early work on predicting depth (and/or surface normals) utilized graphical models [17, 38] and more recent work have improved performance via hierarchical feature extractors [10]. These previous approaches, however, have focused on inferring scene-level information which differs from our goal of perceiving the shape of objects.

Predicting object pose is another task for which many learning-based methods have been developed. Traditional approaches focused on particular instances and used explicit 3D models [19], but the task has recently evolved into the prediction of category-level pose [45, 36, 14]. Category-specific pose prediction models have still the inconvenience that they do not generalize to novel object categories and require large amounts of labeled data for each of their training categories. Recently, Tulsiani *et al*. [44] showed that treating pose as an attribute [11, 26] and training a prediction system accordingly can enable prediction for unfamiliar classes. Our work aims for a similar generalization to arbitrary objects, but the symmetry and surface orientation representations we infer are much more general and detailed.

Learning from CAD Model Collections. There has been a growing trend of using 3D CAD model renderings to aid computer vision algorithms. The key advantage of these approaches is that it is easy to obtain labeled training data at scale. Examples include approaches for aligning 3D models to images [2, 28], object detection [35] and pose estimation [41]. In this work, we apply this idea to normal estimation and symmetry detection - both easily obtained from 3D models.

3. Symmetry Prediction

> "Symmetry is what we see at a glance; based on the fact that there is no reason for any difference."
>
> - Blaise Pascal, *Pensèes*

Most real-world shapes possess symmetries. For example, all object categories available in popular datasets such as PASCAL VOC and Microsoft COCO exhibit at least bilateral reflection symmetry. Symmetry detection provides cues into the elongation modes and 3D orientation of objects which can influence perceived shape (as illustrated by Ernst Mach square/diamond famous example [34]) and is conjectured to aid grouping [23] and recognition [46] in human vision. Symmetry-based approaches such as Blum's Medial Axis Transform [4] spawned entire subcommunities [39] devoted to their development.

Symmetry is however now rarely pursued in practical vision systems, perhaps because too much emphasis has been placed on"retinal" symmetries – symmetries in planar shapes, that are only moderately distorted when projected into an image. Most objects are not planar and their symmetries can be widely deformed after projecting on to images due to the angle relative to the camera and the geometry of central projection. Consequently, we deviate from the existing techniques for detecting retinal symmetrical structures which seek dense correspondences across feature points, aiming to detect subsets of correspondences that can be realized by the underlying structures [29, 7, 30]. Instead, we present a learning-based framework to directly detect the underlying 3D symmetries for an object.

We propose to infer representations similar to those of existing approaches that rely on 3D shape inputs [33] or depth images [43, 37] - but we aim to do so from a single RGB image. In fact, we leverage existing approaches for detecting 3D symmetries in 3D meshes in order to obtain ground truth symmetries. We can then frame a supervised learning problem using rendered images of these shapes as input. We describe our symmetry extraction, symmetry prediction formulation and learning framework below.

Extracting Symmetries from Shapes. We follow the procedure outlined by Mitra *et al*. [33] to extract the global reflectional symmetries given a shape. We sample the shape uniformly to correct for any biased sampling in the original mesh points. We then consider many symmetry plane hypotheses, parametrized as (n, b) where the points satisfying $n \cdot x = b$ lie on the plane, and iteratively refine each hypothesis via ICP between original and reflected points. We

finally discard planes that do not fit the sampled points well and additionally suppress duplicate planes with very similar orientations. We refer the reader to [33] for the exact mathematical formulation.

Formulation. Given an image I, we aim to predict the symmetry planes of the underlying 3D object. Since the exact placement of a plane only assumes a meaning once we have inferred a reconstruction for the object and is not well defined given a single image, we focus on inferring the orientations of the underlying symmetry planes. Let $\mathcal{N}$ represent the space of unit norm 3D orientation vectors. We first discretize this space via approximately uniform samples on the unit sphere [42] $\{n_1, \cdots, n_K\}$. Our learning task is modeled as a multilabel classification where we aim to learn a mapping f' s.t. $f(I) \in \{0,1\}^K$ and $f(I)[k] = 1$ iff n_k is a correct discretization for some symmetry orientation of the underlying 3D object.

Learning. As mentioned earlier, we rely on a large shape collection to learn prediction of symmetries. We also use a rendering engine E which, given a shape model S and a model rotation R yields a rendered image $E(S, R)$. To obtain training data for our task, we repeatedly sample a shape S belonging to some object category $c \in \mathcal{C}$ from the shape collection and detect the underlying reflectional symmetry planes $\{P_i = (n_i, b_i)\}$ as described above. We then sample a model pose from a view distribution $\mathcal{V}$ and obtain the rendered image $E(S, R)$. The symmetry orientations underlying the 3D shape of the rendered image ($\{R * n_i\}$) are computed by rotating the symmetry orientations of the shape S. These orientations are discretized as into orientation bins described above to obtain a label $l \in \{0,1\}^K$. The pair $(E(S, R), l)$ forms one training exemplar for our problem. We sample models and views repeatedly to generate the training data - the exact details are described in the experiments.

Given the training set constructed above, we train a CNN to predict symmetries given a single image. More concretely, we use an Alexnet [24] based architecture with K outputs in the last layer and use a sigmoid cross entropy loss to enforce the outputs to represent log-probability of the corresponding orientation being a symmetry plane for the underlying 3D object. Note that the system is trained in a category-agnostic way i.e. unlike common detection and pose prediction systems [45, 41] , we share output units across all object categories $c \in \mathcal{C}$. This implicitly enforces the CNN based symmetry prediction system to exploit similarities across object classes and learn common representations that may be useful for generalizing to novel objects. Our experiments empirically demonstrate that the system we describe is indeed capable of predicting symmetries for objects belonging to a category $c \notin \mathcal{C}$.

4. Surface Normal Estimation

The importance of *perceiving the surface layout* was highlighted by Gibson as early as 1950 [15]. These ideas were grounded more computationally as Marr's 2.5D sketch representations [32]. Koenderink, Van Doorn and Kappers later demonstrated [22] the ability of humans to recover surface orientations from pictures and shaded objects. All these seminal works, perceptual as well as computational, emphasized the importance of perceiving surface orientations as an integral part of perception.

Single-image depth [10, 38, 21] and surface normal [12, 9, 47] prediction using CNNs has shown promise when dealing with the shape of scenes. Scenes exhibit strong regularities: the ground and the ceiling is horizontal, the walls are vertical. Here we demonstrate that these models can be leveraged to label the much more complex normals of object surfaces. We describe our formulation and learning procedure below.

Formulation. Our aim is to learn a model that is capable of constructing a mapping from pixels to orientations given an image $I(\cdot, \cdot)$. The desired output, given the input image I is a spatial orientation function $N(\cdot, \cdot)$ such that $N(x, y)$ is the surface orientation of the point in the underlying 3D shape that is projected at pixel (x, y) in the given image. Instead of directly predicting an orientation $n \in \mathcal{N}$ at each spatial location, we follow a formulation motivated by Koenderink's experiment where the subjects were able to reconstruct a dense sampling of surface orientations in images using an element from discrete set of *gauge figures* placed at every location. This discretization of surface orientations has been previously successfully leveraged [25] for estimating surface normals of a scene. Our intuition is that this approach combined with CNN architectures that have shown rapid recent progress for pixelwise classification tasks *e.g.* semantic segmentation can yield promising results in the domain of object shape perception.

Operationally, similar to our approach for symmetry plane orientations, we discretize the space of visible surface orientations into K discrete bins using approximately uniform samples over the half unit sphere [42]. The goal for normal estimation is to then learn a function approximation f s.t $f(I) = N$ where $N(x, y)$ assigns the correct orientation bin for the 3D point projected at (x, y).

Learning. Our data generation process is similar to the task of symmetry prediction described previously. We use a rendering engine E which, given a shape model S and a model pose R yields a rendered image $E(S, R)$ and additionally provides a surface orientation image $\hat{N}(S, R)$. We first sample a category c from training classes $\mathcal{C}$ and then a shape S. A random view R is sampled from a view distri-

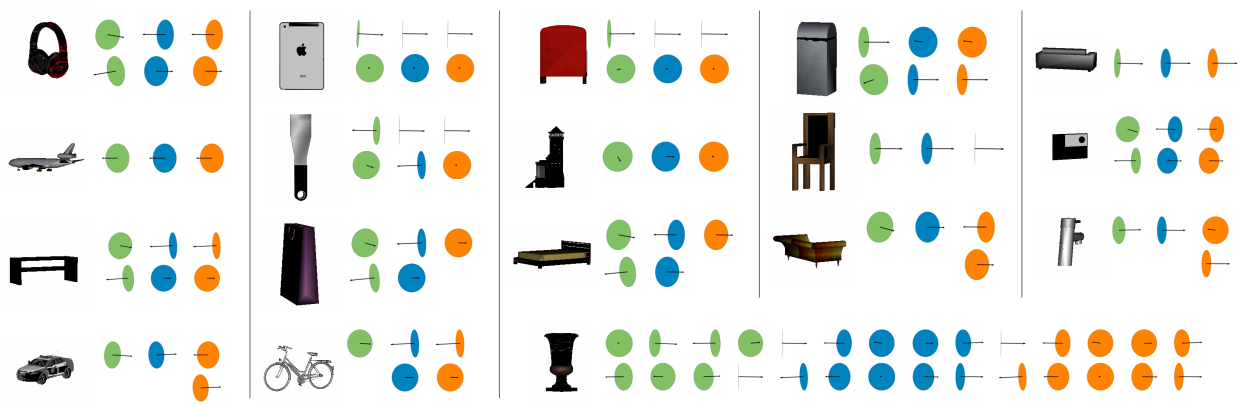

Figure 2: Symmetry predictions for 'Learned' and 'Induced' settings for various test objects in our dataset. Each symmetry plane is visualized via a 3D circle parallel to the plane and an arrow denoting the normal to the plane. The green planes represent the ground-truth symmetries, the blue symmetry planes are predicted in the 'Learned' setting and the orange symmetry planes are predicted under the 'Induced' setting.

bution $\mathcal{V}$ and the engine E yields a rendering and normal image pair $(I, \hat{N})$. We then discretize $\hat{N}$ using the orientation bins above to obtain N where $N(x, y) \in \{1, \cdots K\}$ is the orientation bin for the underlying surface. The pair (I, N) forms a training sample for our learning system.

Given the training set constructed above, we train a CNN to predict pixel-wise surface normals. Our architecture choice is motivated by recent methods that leverage CNNs to predict a dense pixel-wise output *e.g.* semantic segmentation [31] and image synthesis [8]. A common technique used in these architectures is to eschew fully connected layers common for image-level classification tasks and instead use multiple convolution layers followed by *deconvolution* layers (reverse convolution with unpooling) to produce a dense pixelwise output. Let $C(k, s, o), D(k, s, o)$ denote a convolution layer with kernel size k, (downsampling(*conv*) / upsampling(*deconv*)) stride s and o output channels and $P(k, s)$ represent a max-pooling layer with kernel size k and stride s. Using the shorthand $C'(o)$ for $C(3, 1, o) - C(3, 1, o) - C(3, 1, o) - P(2, 2)$, our network architecture is $I - C'(64) - C'(128) - C'(256) - C'(512) - C'(512) - D(3, 2, 256) - D(3, 2, 128) - D(3, 2, 64) - D(3, 2, K)$. The network above takes an input image and produces an output pixelwise log-probability distribution over the K orientation bins. We minimize a softmax loss over the pixelwise log-probabilities predicted and train the CNN described using the Caffe framework. The convolutional layers are initialized using the VGG16 pretrained model for image classification [40] and the deconvolution layers are initialized randomly. The architecture described produces a 113×113 spatial output given an input image of size 224×224 and this resolution allows our model to capture sharp discontinuities. In a similar spirit to symmetry orientation prediction, the category-agnostic formulation and learning of the surface orientation prediction allow us to learn common representations to predict surface normals for novel objects.

5. Experiments

Experiments were performed to investigate the following: 1) the performance of our symmetry and normal prediction systems and 2) their ability to generalize to novel unseen object categories. We first describe our experimental setup and then present results on symmetry detection and surface normal estimation. Finally, we show qualitative results on real world images in Figure 5.

Dataset. We use the ShapeNet [1] dataset to download 3D models for objects corresponding to 57 object classes. These 3D models (collected from large scale 3D model repositories such as 3D Warehouse and Yobi3D) belong to object categories ranging from cars and buses to faucets and washers and form a varied set of commonly occurring rigid objects. We keep up to 200 models per object category (with a 75%/25% train/test split) and use 200 renderings for each 3D model in our training set to train the prediction systems previously described (totalling around 1.5 million images). In addition, we also sample equally from all classes for each training iteration in order to counter the class imbalance in the number of available models. Our testing set includes 3200 rendered images from each of the 57 object categories. For ease of reproducibility, we plan to make our train/test splits and code available.

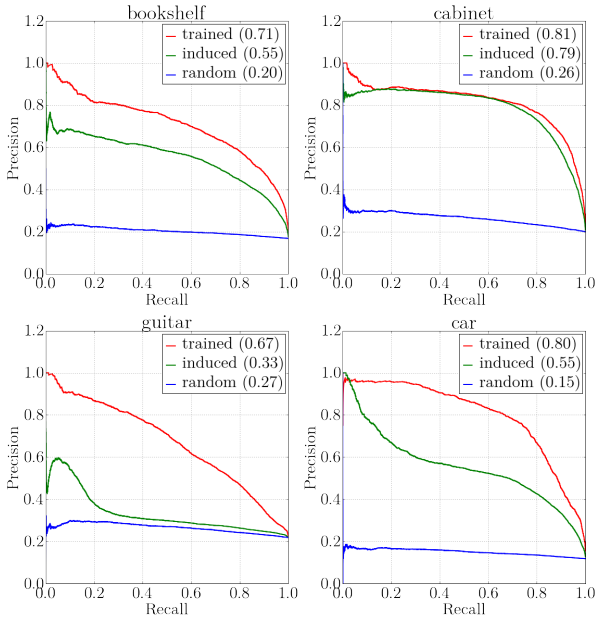

Figure 3: Precision-Recall plots for symmetry detection under 'Induced' and 'Learned' settings for representative classes.

Viewpoint Variability. The viewing angle for an object can be described using three euler angles - azimuth ($\phi \in (-180, 180]$), elevation($\varphi \in (-180, 180]$) and cyclo-rotation($\psi \in (-90, 90]$). Objects, however, tend to follow certain view distributions (*e.g.* we rarely see cars from the bottom). In particular, the primary variation in viewing angle for objects in natural scenes is along the azimuth. To account for this, we sample views uniformly from a set of more *natural* views $V_N = \{\phi \in (-180, 180]\} \times \{\varphi \in [0, 10]\} \times \{\psi \in [0, 0]\}$. It is, however, also important to handle objects seen from arbitrary views. We therefore also train and test our models under a more diverse view sampling from $V_D = \{\phi \in (-180, 180]\} \times \{\varphi \in [0, 50]\} \times \{\psi \in [-30, 30]\}$ to analyze the prediction and induction performance under more challenging settings.

Induction Splits. A primary aim of our experimental evaluation is to analyze the induction ability of our system across novel object classes. For this analysis, we randomly partitioned the object classes $\mathcal{C}$ in the ShapeNet dataset in two disjoint sets $\mathcal{C}_\mathcal{A}$ and $\mathcal{C}_\mathcal{B}$ - the categories in each set are listed in the appendix. For both the shape prediction tasks we study - normal and symmetry prediction, we train 3 models with the same hyper-parameters. One model is trained on the entire set of classes $\mathcal{C}$ and two models over $\mathcal{C}_\mathcal{A}$ and $\mathcal{C}_\mathcal{B}$ respectively. This allows us to empirically estimate the induction performance for a class $c \in (\mathcal{C}_\mathcal{A}$ or $\mathcal{C}_\mathcal{B})$ by comparing the performance of the systems trained over

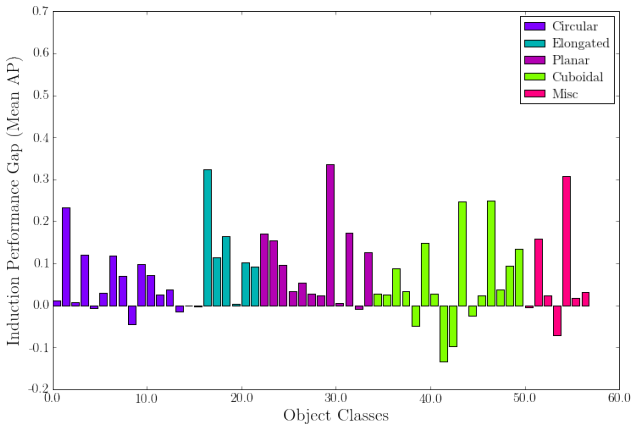

Figure 4: Analysis of performance gap (in AP_s^θ) for object categories between 'Induced' and 'Learned' settings for symmetry prediction.

Mean AP_s^θ	**Setting**		
Viewpoint Sampling	Learned	Induced	Random
V_N	0.69	0.58	0.32
V_D	0.59	0.47	0.07

Table 1: Mean performance across classes for symmetry prediction.

$\mathcal{C}$ and ($\mathcal{C}_\mathcal{B}$ or $\mathcal{C}_\mathcal{A}$) respectively. In all the experiments described below, we report numbers under both the 'Learned' and 'Induced' settings. The 'Learned' setting denotes the performance of our system when trained using *all* object classes and the 'Induced' setting indicates our performance when, for each object class we use the system trained on the set of classes ($\mathcal{C}_\mathcal{A}$ or $\mathcal{C}_\mathcal{B}$) *not* containing the class under consideration.

5.1. Symmetry Prediction

Evaluation Criterion. Since the task we address is not a standard one, we need to decide on evaluation metrics. In the related task of pose prediction, the common practice is to measure the deviation between predicted and annotated pose [45, 41]. Symmetries, however, do not lend themselves to a similar analysis because there can be multiple of them and consequently, a symmetry prediction system would yield multiple symmetry hypotheses with varying confidences. In that respect, our task perhaps has more in common with object detection - given an image with a variable number of symmetry planes (*c.f.* objects), a prediction system outputs a few distinct hypotheses from the continuous space of plane orientations (*c.f.* bounding box locations). We therefore adapt the standard object detection Average Precision (AP) metric for our task.

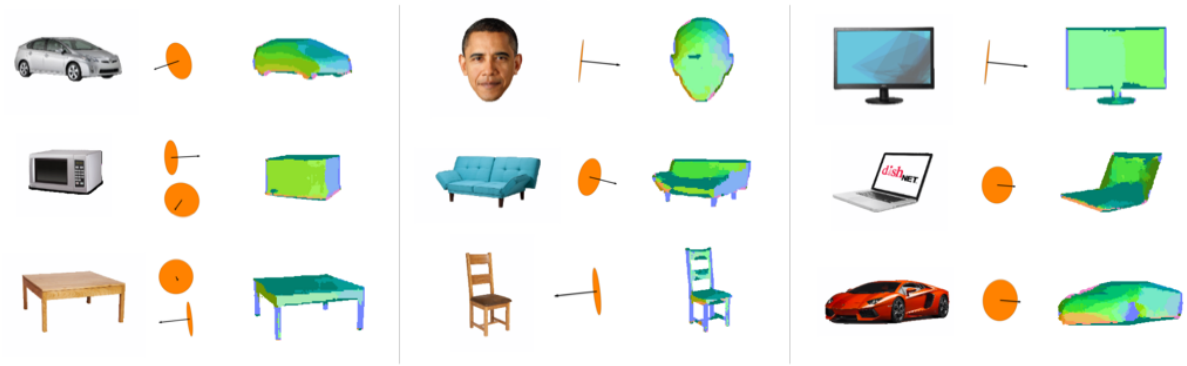

Figure 5: Predicted symmetry orientations and surface normals for objects segmented out from real world images. The symmetries are shown using the convention in Figure 2 and surface normals are mapped into RGB space via the mapping $X \rightarrow$B, $Y \rightarrow$R, $Z \rightarrow$G.

We propose AP_s^θ as a metric to evaluate the performance of a symmetry prediction system. Given the ground-truth symmetries of an object $\hat{\mathcal{N}} = \{\hat{n}_i\}$ and predicted symmetry orientations $\mathcal{N} = \{n_j\}$ alongwith their probability scores p_j, a prediction $n \in \mathcal{N}$ is considered correct if $\exists i$ s.t $\Delta_s(n, \hat{n}_i) \leq \theta$. Akin to the object detection setting, we also prevent double counting of ground-truth symmetries when matching a predicted symmetry. We vary the probability threshold for symmetry detection and consider all instances of a class together to obtain a point on the Precision-Recall curve. The AP_s^θ metric denotes the area under the above Precision-Recall curve.

Results. We report the analysis of our system in Table 1. We use $\theta = \frac{\pi}{18}$ for measuring the performance under the AP_s^θ metric. We report the performance of our systems for both view sampling settings V_N and V_D. The system trained under the view sampling V_N only classifies symmetry plane orientations among 10 possible horizontal directions whereas the system under V_D setting predicts from among 60 possible orientations. We observe that the performance in the 'Induced' setting, where we have not seen a single annotated object of the corresponding class, is comparable to the 'Learned' setting with observed training examples. It is also encouraging that the results hold in the natural as well as diverse view sampling scenarios and that the 'Induced' results are significantly better than an uninformed random baseline, thereby supporting our claim of the ability to generalize symmetry prediction across novel objects.

Analysis and Observations. We manually grouped together object categories in coarse groups based on the shape of the typical bounding convex set. The resulting groups are indicated in Figure 4 which also shows the performance gap, under the V_N view sampling, between the 'Learned' and 'Induced' settings. We observe that the gap for a large fraction of the categories is low. In particular, we observe this trend for 'Circular' and 'Cuboidal' classes - this may perhaps be a result of a large number of such classes being available for training and thus aiding generalization for novel objects of similar classes.

We also show some predictions from our system in Figure 2. Both the induced as well as the trained system correctly predict most of the symmetries present in the objects. One of the primary error modes we observe in the 'Induced' setting is *over-generalization* where the system confidently predicts symmetries in addition to the correct one for objects like motorbikes, rifles *etc*., possibly on account of more commonly occurring classes with multiple symmetries. We show the performance under various settings in Figure 3 for some representative classes. The first two are typical classes with strong generalization results whereas the performance on category 'car' reduces significantly.

5.2. Surface Normal Estimation

Evaluation Metrics. We follow the evaluation protocol from Fouhey *et al*. [12] and evaluate our predicted surface normals against the ground truth using 5 metrics - mean angular error, median angular error and the fraction of 'good' pixels - pixels whose predicted normals lie within 11.25°, 22.5° and 30° of the ground truth normals respectively. All the above metrics are computed per object category and then reported below by averaging across the 57 classes. The mean and median angular error are computed across all object pixels per category (background is ignored) and so are the fraction of 'good' pixels. We also report curves of fraction of 'good' pixels vs. the angular threshold at which they are calculated and compute the area under the curve when the max angular threshold is 30°.

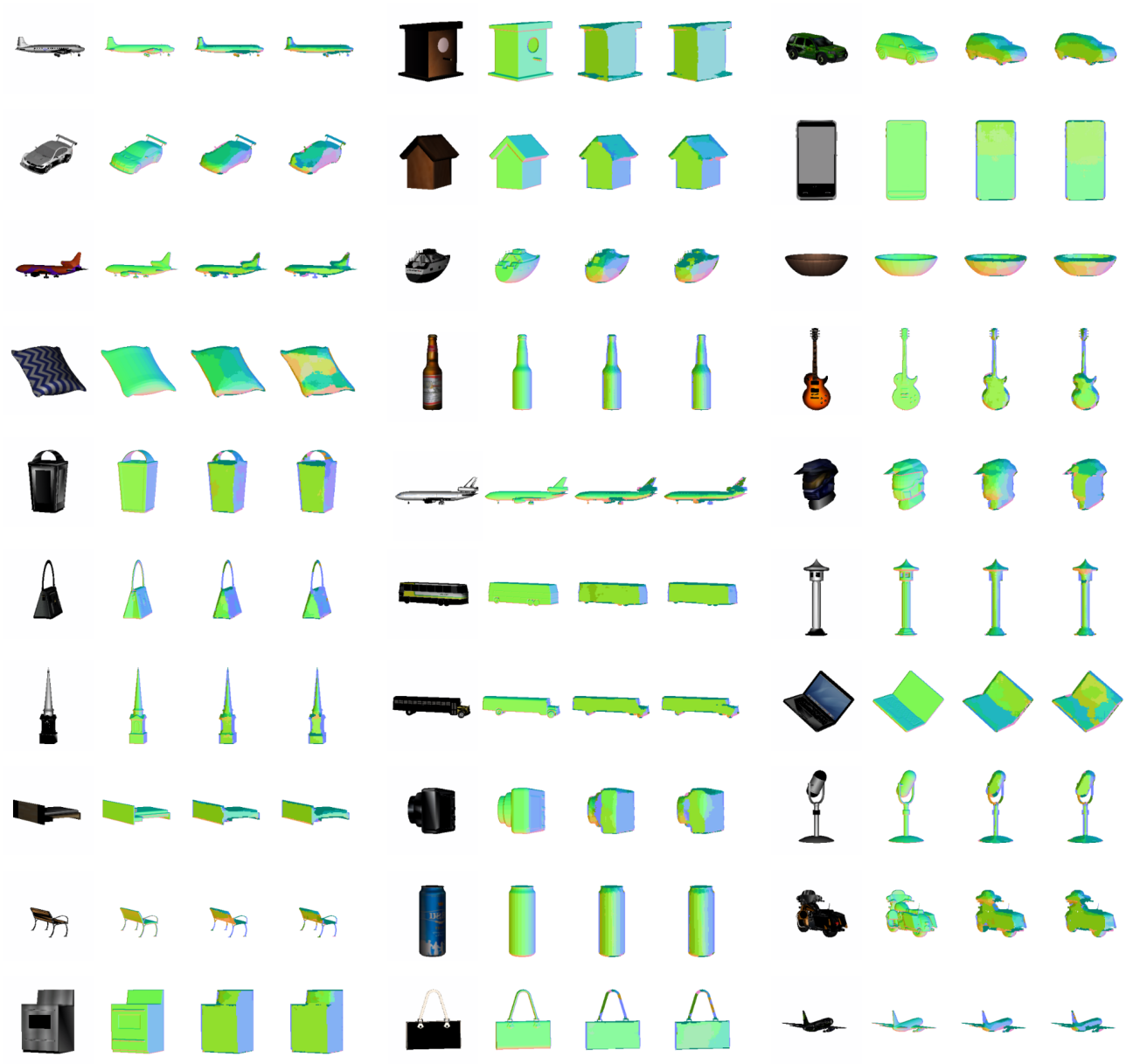

Figure 6: Surface normal predictions using our learned and induced models on held out data. Each unit in the figure shows (from left to right) the image, its ground truth surface normals, surface normals predicted by our 'Learned' model (trained on all classes) and surface normals predicted by our 'Induced' model (trained on the subset of categories not containing this particular class). Surface normals are mapped into RGB space via the mapping $X \rightarrow$B, $Y \rightarrow$R, $Z \rightarrow$G

Results. The results for our surface normal prediction system(s) are shown in Table 2. As in the symmetry prediction task, we report results in both the V_N and V_D settings. For *both* settings, the surface normal direction is discretized into 60 uniformly sampled bins on the hemisphere and the predicted labels are converted back into surface normals by looking up the orientation corresponding to the predicted bin. The 'Learned' and 'Induced' settings again refer to the experimental setups where the system was trained on all object classes and on the split of the dataset ($\mathcal{C}_\mathcal{A}$ or $\mathcal{C}_\mathcal{B}$) *not* containing this object class respectively. It can be seen that our model achieves a pixel-wise median angular error rate of around 15° and moreover the performance for the 'Learned' and 'Induced' settings are comparable, validating the claim that our surface normal prediction system generalizes to unseen object categories. This trend is visible across

Metrics	V_N		V_D	
	Learned	**Induced**	**Learned**	**Induced**
Mean Error	21.3	23.8	23.5	26.5
Median Error	12.7	14.7	14.8	17.1
%GP 11.25°	50.4	45.7	41.6	37.1
%GP 22.5°	71.9	67.0	67.6	62.1
%GP 30.0°	77.8	73.5	74.7	69.5

Table 2: Mean performance across classes for surface normal estimation under various view settings. Lower is better for the top half of the table and higher is better for the percent of 'good' pixels metrics. Please refer to the text for more details on the metrics.

all error metrics as well as across viewpoint variation settings V_N and V_D.

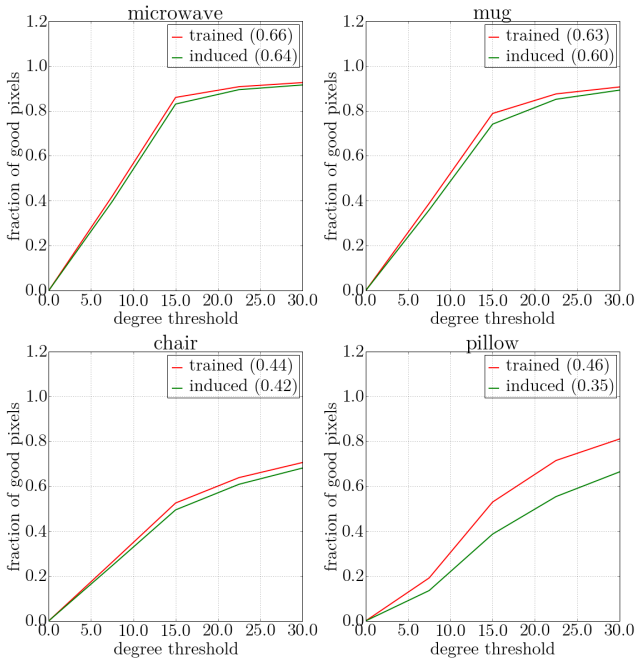

Figure 7: Fraction of good pixels vs. degree threshold plots for surface normal prediction under 'Induced' and 'Learned' settings for representative classes. The area under the curves are mentioned in the plot legends.

Some results from our surface normal predictor are shown in Figure 6. It can be seen that our system is able to reliably predict surface normals at a coarse level while also respecting discontinuities/edges. The major error modes for our system are fine structures which it is unable to handle owing to the large receptive fields in the middle convolutional layers in our architecture. Figure 8 shows the relative performance (in median angular error) of our 'Learned'

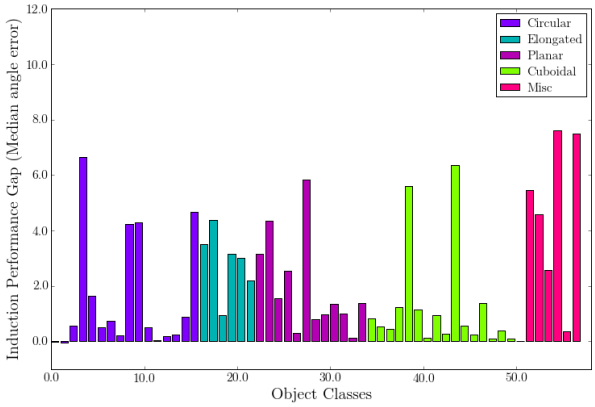

Figure 8: Analysis of performance gap in (median angular error) between 'Induced' and 'Learned' settings for surface normal prediction.

and 'Induced' systems for various shape 'super-categories' such as circular, cuboidal etc. It can be seen that the generalization works best for circular and cuboidal categories (consistent with the symmetry prediction experiments). We also show some plots for fraction of good pixels vs. angular threshold in Figure 7. It can be seen that surface normals for 'microwave' (cuboidal) and 'mug' (cylindrical) generalize well whereas a non-standard shape such as 'pillow' doesn't. 'Chairs' on the other hand are overall worse off than other simpler categories but the coarse structures in them generalize well giving rise to similar curves for 'Learned' and 'Induced'. Detailed results and plots for symmetry prediction and surface normal estimation for all 57 object classes can be found in the appendix.

6. Discussion

Our results suggest that it is feasible to induce surface normals and 3D symmetry planes for objects from unfamiliar categories, by learning hierarchical feature extractors on a large-scale dataset of CAD model renderings. We also demonstrated that our learned models can operate on real images. The techniques we present here can in principle also be applied to predicting other shape properties such as local curvature, rotational symmetries *etc*.

Our approach connects modern representation learning approaches with the spirit of the pioneers in computer vision, that emphasized spatial vision and the understanding of shape. Should reconstruction be an input to classification, as for example Marr postulated [32]? Should it be the other way around? Or are both best handled as parallel processes, as in the dual stream hypothesis of neuroscience [16]? We hope our approach would be useful for applications where shape understanding is important, including robotic perception and human-computer interaction.

Acknowledgements

This work was supported in part by NSF Award IIS-1212798 and ONR MURI-N00014-10-1-0933. Shubham Tulsiani was supported by the Berkeley fellowship. João Carreira was supported by the Portuguese Science Foundation, FCT, under grant SFRH/BPD/84194/2012. Qixing Huang thanks the gift awards from Adobe and Intel. We gratefully acknowledge NVIDIA corporation for GPU donations towards this research.

References

[1] Shapenet. http://www.shapenet.org. 4, 11

[2] M. Aubry, D. Maturana, A. Efros, B. Russell, and J. Sivic. Seeing 3d chairs: exemplar part-based 2d-3d alignment using a large dataset of cad models. In *CVPR*, 2014. 2

[3] J. T. Barron and J. Malik. Shape, illumination, and reflectance from shading. *TPAMI*, 2015. 2

[4] H. Blum. A transformation for extracting new descriptors of shape. In *Proc. Models for the Perception of Speech and Visual Form*, 1967. 2

[5] J. Carreira, S. Vicente, L. Agapito, and J. Batista. Lifting object detection datasets into 3d. 2015. 1

[6] T. J. Cashman and A. W. Fitzgibbon. What shape are dolphins? building 3d morphable models from 2d images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012. 1

[7] D. Ceylan, N. J. Mitra, Y. Zheng, and M. Pauly. Coupled structure-from-motion and 3d symmetry detection for urban facades. *ACM Trans. Graph.*, 2014. 2

[8] A. Dosovitskiy and T. Brox. Inverting convolutional networks with convolutional networks. *arXiv preprint arXiv:1506.02753*, 2015. 4

[9] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *ICCV*, 2015. 3

[10] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *NIPS*, 2014. 2, 3

[11] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *CVPR*, 2009. 2

[12] D. F. Fouhey, A. Gupta, and M. Hebert. Data-driven 3D primitives for single image understanding. In *ICCV*, 2013. 3, 6

[13] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 1980. 1

[14] A. Ghodrati, M. Pedersoli, and T. Tuytelaars. Is 2d information enough for viewpoint estimation? In *BMVC*, 2014. 2

[15] J. J. Gibson. The perception of the visual world. 1950. 3

[16] M. A. Goodale and A. D. Milner. Separate visual pathways for perception and action. *Trends in neurosciences*, 15(1):20–25, 1992. 8

[17] D. Hoiem, A. A. Efros, and M. Hebert. Automatic photo pop-up. In *SIGGRAPH*, 2005. 2

[18] B. K. Horn. Obtaining shape from shading information. In *Shape from shading*, pages 123–171. MIT press, 1989. 2

[19] D. P. Huttenlocher and S. Ullman. Recognizing solid objects by alignment with an image. *IJCV*, 1990. 2

[20] A. Kar, S. Tulsiani, J. Carreira, and J. Malik. Category-specific object reconstruction from a single image. In *CVPR*, 2015. 1

[21] K. Karsch, C. Liu, and S. B. Kang. Depthtransfer: Depth extraction from video using non-parametric sampling. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2014. 3

[22] J. J. Koenderink, A. J. Van Doorn, and A. M. Kappers. Pictorial surface attitude and local depth comparisons. *Perception & Psychophysics*, 58(2):163–173, 1996. 3

[23] K. Koffka. *Principles of Gestalt psychology*, volume 44. Routledge, 2013. 2

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 3

[25] L. Ladickỳ, B. Zeisl, and M. Pollefeys. Discriminatively trained dense surface normal estimation. In *ECCV*. 2014. 3

[26] C. H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR*, 2009. 2

[27] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to hand-written zip code recognition. In *Neural Computation*, 1989. 1

[28] J. J. Lim, H. Pirsiavash, and A. Torralba. Parsing IKEA Objects: Fine Pose Estimation. In *ICCV*, 2013. 2

[29] J. Liu and Y. Liu. Curved reflection symmetry detection with self-validation. In *Asian Conference on Computer Vision(ACCV)*, 2010. 2

[30] Y. Liu, H. Hel-Or, C. S. Kaplan, and L. J. V. Gool. Computational symmetry in computer vision and computer graphics. *Foundations and Trends in Computer Graphics and Vision*, 2010. 2

[31] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 4

[32] D. Marr. Vision: A computational approach, 1982. 3, 8

[33] N. J. Mitra, L. Guibas, and M. Pauly. Symmetrization. *SIGGRAPH*, 2007. 2

[34] S. E. Palmer. *Vision science: Photons to phenomenology*. MIT press Cambridge, MA, 1999. 2

[35] X. Peng, B. Sun, K. Ali, and K. Saenko. Exploring invariances in deep convolutional neural networks using synthetic images. *CoRR*, abs/1412.7122, 2014. 2

[36] B. Pepik, M. Stark, P. Gehler, and B. Schiele. Teaching 3d geometry to deformable part models. In *CVPR*, 2012. 2

[37] J. Rock, T. Gupta, J. Thorsen, J. Gwak, D. Shin, and D. Hoiem. Completing 3d object shape from one depth image. In *CVPR*, 2015. 2

[38] A. Saxena, M. Sun, and A. Y. Ng. Make3d: Learning 3d scene structure from a single still image. *PAMI*, 2009. 2, 3

[39] K. Siddiqi, A. Shokoufandeh, S. J. Dickinson, and S. W. Zucker. Shock graphs and shape matching. *International Journal of Computer Vision*, 35(1):13–32, 1999. 2

[40] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 4

[41] H. Su, C. R. Qi, Y. Li, and L. J. Guibas. Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views. In *ICCV*, 2015. 2, 3, 5

[42] R. Swinbank and R. James Purser. Fibonacci grids: A novel approach to global modelling. *Quarterly Journal of the Royal Meteorological Society*, 132(619):1769–1793, 2006. 3

[43] S. Thrun and B. Wegbreit. Shape from symmetry. In *ICCV*, 2005. 2

[44] S. Tulsiani, J. Carreira, and J. Malik. Pose induction for novel object categories. In *ICCV*, 2015. 2

[45] S. Tulsiani and J. Malik. Viewpoints and keypoints. In *CVPR*, 2015. 2, 3, 5

[46] T. Vetter and T. Poggio. Linear object classes and image synthesis from a single example image. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7):733–742, 1997. 2

[47] X. Wang, D. F. Fouhey, and A. Gupta. Designing deep networks for surface normal estimation. In *CVPR*, 2015. 3

[48] Y. Xiang, R. Mottaghi, and S. Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *Applications of Computer Vision (WACV), 2014 IEEE Winter Conference on*, pages 75–82. IEEE, 2014. 1

Appendix

Induction Splits. For the analysis of the induction ability of our system across novel object classes, we randomly partitioned the object classes $\mathcal{C}$ in the ShapeNet dataset [1] in two disjoint sets $\mathcal{C}_{\mathcal{A}}$ and $\mathcal{C}_{\mathcal{B}}$. The categories in each set are listed in Table 1.

$\mathcal{C}_{\mathcal{A}}$	$\mathcal{C}_{\mathcal{B}}$
airplane	ashcan
bathtub	bag
bed	basket
bicycle	bench
bookshelf	birdhouse
bottle	boat
bowl	cabinet
bus	camera
can	cap
clock	car
computer keyboard	cellular telephone
dishwasher	chair
file	display
loudspeaker	earphone
mailbox	faucet
microphone	guitar
microwave	helmet
mug	jar
piano	knife
pillow	lamp
pistol	laptop
pot	motorcycle
printer	remote control
skateboard	rifle
stove	rocket
table	sofa
telephone	tower
train	vessel
	washer

Table 1: Induction Splits

Shape Groups. We provided additional analysis of our method by manually grouping together object categories in coarse groups based on the shape of the typical bounding convex set. The resulting groups used were as follows -

- **Circular :** ashcan, basket, bottle, bowl, can, cap, clock, helmet, jar, lamp, microphone, mug, pot, rocket, tower, washer
- **Elongated :** computer keyboard, knife, piano, rifle, skateboard, train
- **Planar :** airplane, bag, bench, bicycle, bookshelf, cellular telephone, display, file, laptop, motorcycle, pistol, remote control
- **Cuboidal :** bathtub, bed, bus, cabinet, camera, car, chair, dishwasher, loudspeaker, mailbox, microwave, pillow, printer, sofa, stove, table
- **Misc :** birdhouse, boat, earphone, faucet, guitar, telephone, vessel

Symmetry Prediction. We show the Precision-Recall plots for symmetry prediction for all classes under the view sampling V_n and V_d in Figure 1 and Figure 2 respectively. The performance under AP_s^{θ} metric is also reported in Table 2.

Normal Estimation. We show the performance plots for normal estimation for all classes under the view sampling V_n and V_d in Figure 3 and Figure 4 respectively. The performance under various metrics under the view sampling V_n and V_d are reported in Table 3 and Table 4 respectively.

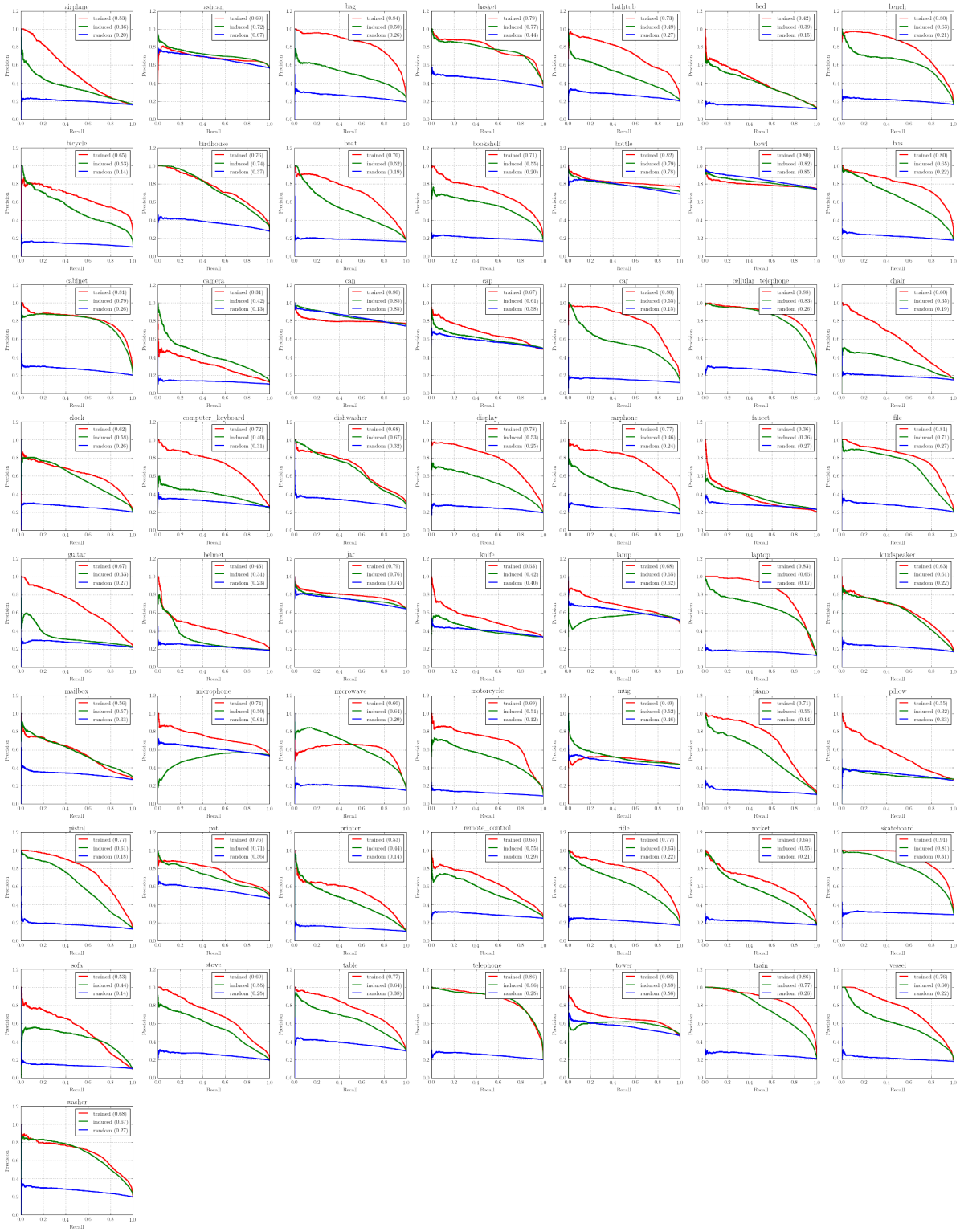

Figure 1: Precision-Recall plots for symmetry detection under 'Induced' and 'Learned' settings and V_n view sampling for all classes.

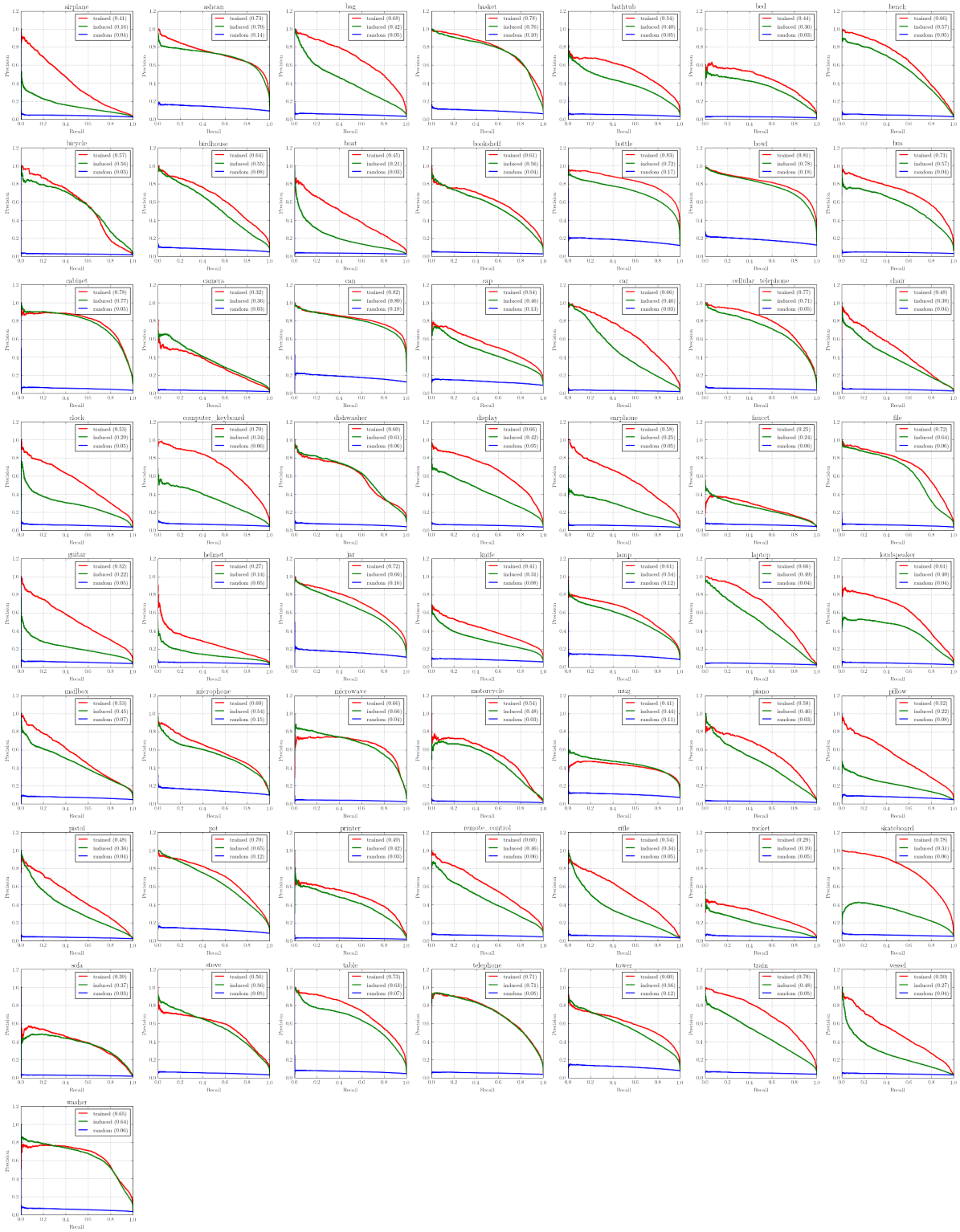

Figure 2: Precision-Recall plots for symmetry detection under 'Induced' and 'Learned' settings and V_d view sampling for all classes.

View Setting	V_D			V_N		
	Learned	Induced	Random	Learned	Induced	Random
airplane	0.41	0.16	0.04	0.53	0.36	0.20
ashcan	0.73	0.70	0.14	0.69	0.72	0.67
bag	0.68	0.42	0.05	0.84	0.50	0.26
basket	0.78	0.76	0.10	0.79	0.77	0.44
bathtub	0.54	0.40	0.05	0.73	0.49	0.27
bed	0.44	0.36	0.03	0.42	0.39	0.15
bench	0.66	0.57	0.05	0.80	0.63	0.21
bicycle	0.57	0.56	0.03	0.65	0.53	0.14
birdhouse	0.64	0.55	0.08	0.76	0.74	0.37
boat	0.45	0.21	0.03	0.70	0.52	0.19
bookshelf	0.61	0.56	0.04	0.71	0.55	0.20
bottle	0.83	0.72	0.17	0.82	0.79	0.78
bowl	0.81	0.78	0.18	0.80	0.82	0.85
bus	0.71	0.57	0.04	0.80	0.65	0.22
cabinet	0.78	0.77	0.05	0.81	0.79	0.26
camera	0.32	0.36	0.03	0.31	0.42	0.13
can	0.82	0.80	0.18	0.80	0.85	0.85
cap	0.54	0.46	0.13	0.67	0.61	0.58
car	0.66	0.46	0.03	0.80	0.55	0.15
cellular_telephone	0.77	0.71	0.05	0.88	0.83	0.26
chair	0.48	0.39	0.04	0.60	0.35	0.19
clock	0.53	0.29	0.05	0.62	0.58	0.26
computer_keyboard	0.70	0.34	0.06	0.72	0.40	0.31
dishwasher	0.60	0.61	0.06	0.68	0.67	0.32
display	0.66	0.42	0.05	0.78	0.53	0.25
earphone	0.58	0.25	0.05	0.77	0.46	0.24
faucet	0.25	0.24	0.06	0.36	0.36	0.27
file	0.72	0.64	0.06	0.81	0.71	0.27
guitar	0.52	0.22	0.05	0.67	0.33	0.27
helmet	0.27	0.14	0.05	0.43	0.31	0.23
jar	0.72	0.66	0.16	0.79	0.76	0.74
knife	0.41	0.31	0.08	0.53	0.42	0.40
lamp	0.61	0.54	0.12	0.68	0.55	0.62
laptop	0.66	0.49	0.04	0.83	0.65	0.17
loudspeaker	0.61	0.40	0.04	0.63	0.61	0.22
mailbox	0.53	0.45	0.07	0.56	0.57	0.33
microphone	0.60	0.54	0.15	0.74	0.50	0.61
microwave	0.66	0.66	0.04	0.60	0.64	0.20
motorcycle	0.54	0.48	0.03	0.69	0.51	0.12
mug	0.41	0.44	0.11	0.49	0.52	0.46
piano	0.58	0.46	0.03	0.71	0.55	0.14
pillow	0.52	0.22	0.08	0.55	0.32	0.33
pistol	0.48	0.36	0.04	0.77	0.61	0.18
pot	0.70	0.65	0.12	0.76	0.71	0.56
printer	0.49	0.42	0.03	0.53	0.44	0.14
remote_control	0.60	0.46	0.06	0.65	0.55	0.29
rifle	0.54	0.34	0.05	0.77	0.63	0.22
rocket	0.29	0.19	0.05	0.65	0.55	0.21
skateboard	0.78	0.31	0.06	0.91	0.81	0.31
sofa	0.39	0.37	0.03	0.53	0.44	0.14
stove	0.56	0.56	0.05	0.69	0.55	0.25
table	0.73	0.63	0.07	0.77	0.64	0.38
telephone	0.71	0.71	0.05	0.86	0.86	0.25
tower	0.60	0.56	0.12	0.66	0.59	0.56
train	0.70	0.48	0.05	0.86	0.77	0.26
vessel	0.50	0.27	0.04	0.76	0.60	0.22
washer	0.65	0.64	0.06	0.68	0.67	0.27

Table 2: Performance across classes for symmetry prediction under AP_s^θ metric.

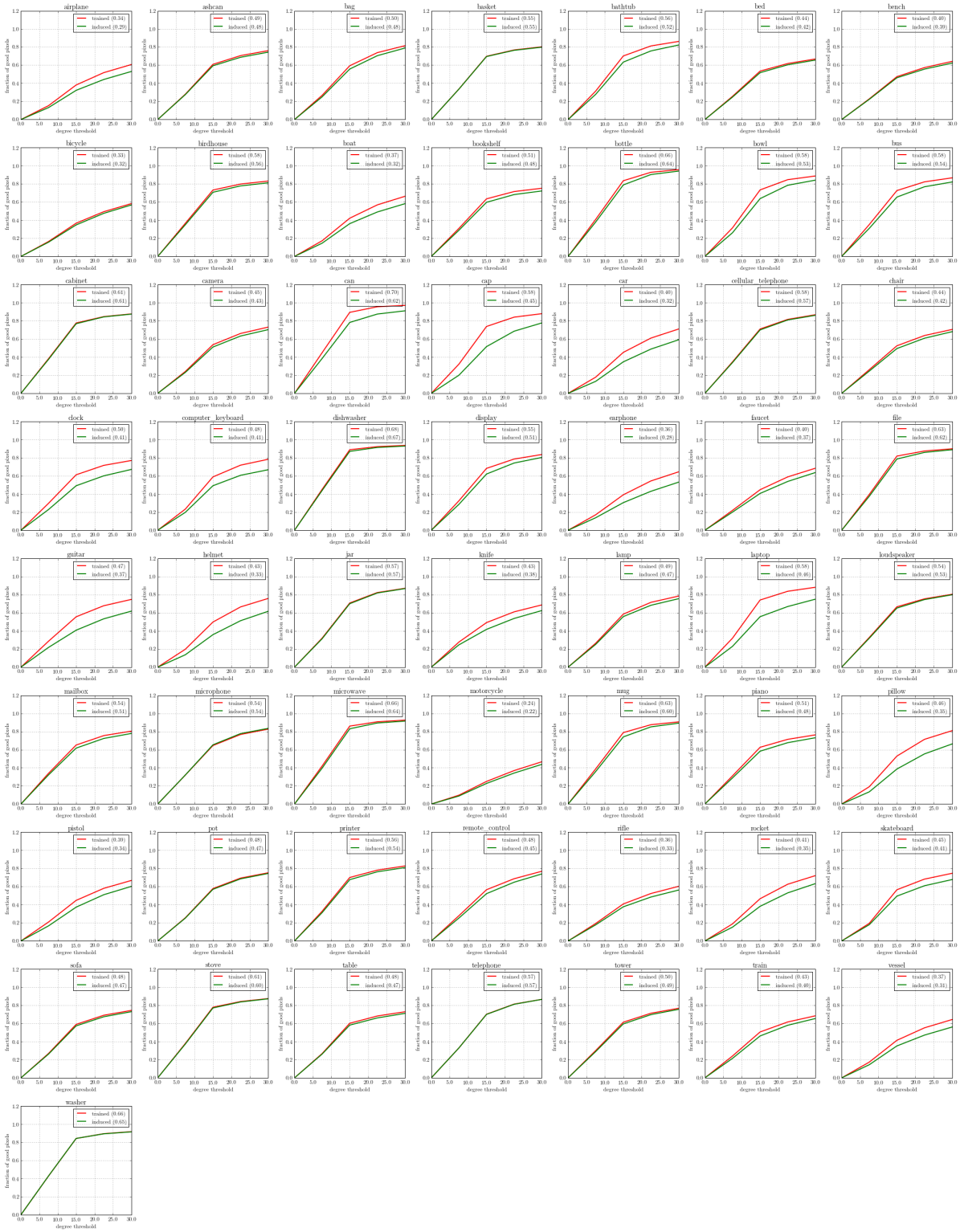

Figure 3: Fraction of good pixels vs. degree threshold plots for surface normal prediction under 'Induced' and 'Learned' settings and V_n view sampling for all classes. The area under the curves are mentioned in the plot legends.

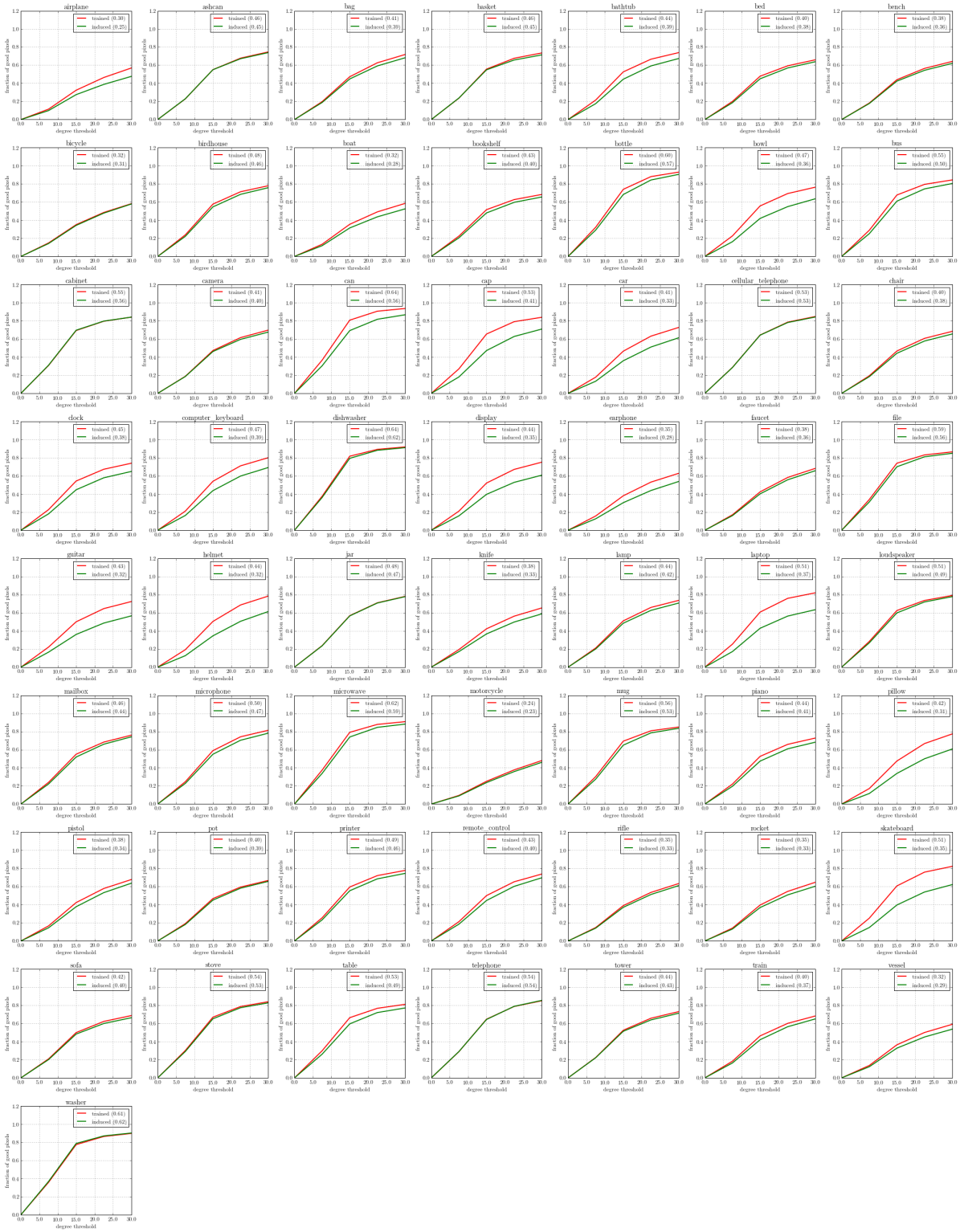

Figure 4: Fraction of good pixels vs. degree threshold plots for surface normal prediction under 'Induced' and 'Learned' settings and V_d view sampling for all classes. The area under the curves are mentioned in the plot legends.

Metrics	**%GP** 11.25°	**%GP** 22.5°	**%GP** 30.0°	**Mean Error**	**Median Error**
airplane	29.3 (25.2)	51.7 (44.2)	60.7 (53.0)	30.9 (34.7)	21.4 (27.2)
ashcan	50.0 (48.8)	70.5 (68.7)	76.0 (74.3)	21.9 (22.9)	11.3 (11.5)
bag	46.2 (43.0)	73.8 (70.4)	81.5 (78.9)	19.2 (20.5)	12.2 (13.2)
basket	60.0 (59.7)	76.8 (76.4)	80.2 (79.8)	20.1 (20.3)	9.4 (9.5)
bathtub	57.6 (50.8)	81.2 (75.7)	86.1 (82.2)	17.2 (19.6)	9.8 (11.1)
bed	44.8 (43.2)	61.8 (60.4)	66.8 (65.6)	28.6 (29.5)	13.2 (14.0)
bench	40.0 (39.1)	57.3 (55.6)	64.1 (62.3)	29.2 (30.3)	16.7 (17.7)
bicycle	28.6 (27.1)	49.4 (47.6)	58.3 (56.8)	31.9 (32.7)	23.0 (24.4)
birdhouse	62.3 (59.6)	79.9 (77.8)	83.1 (81.2)	18.4 (19.5)	9.2 (9.6)
boat	32.0 (27.6)	56.9 (49.0)	66.3 (58.2)	28.2 (32.4)	18.6 (23.2)
bookshelf	54.0 (50.6)	71.6 (68.2)	75.2 (72.2)	23.7 (25.6)	10.3 (11.1)
bottle	69.5 (64.2)	92.8 (90.3)	95.7 (94.2)	11.0 (12.2)	8.5 (9.0)
bowl	58.4 (48.8)	84.7 (78.5)	88.7 (84.1)	15.5 (18.8)	9.9 (11.5)
bus	61.2 (53.2)	82.3 (76.9)	86.7 (82.1)	16.3 (19.5)	9.3 (10.5)
cabinet	67.5 (66.1)	84.6 (84.3)	87.6 (87.4)	15.2 (15.5)	8.7 (8.9)
camera	43.1 (40.9)	66.1 (63.2)	73.1 (70.3)	24.0 (25.6)	13.6 (14.5)
can	76.1 (65.1)	95.5 (87.5)	97.1 (91.0)	9.8 (13.2)	8.0 (8.9)
cap	58.6 (38.3)	84.1 (68.6)	87.9 (77.6)	15.9 (22.2)	9.8 (14.5)
car	33.5 (25.4)	61.0 (48.8)	70.9 (59.0)	25.3 (31.8)	16.9 (23.3)
cellular_telephone	59.3 (58.6)	81.6 (80.9)	86.7 (86.1)	16.3 (16.7)	9.5 (9.7)
chair	43.2 (40.5)	63.8 (60.8)	70.5 (68.0)	25.9 (27.3)	13.9 (15.3)
clock	50.2 (39.3)	71.6 (60.0)	76.9 (67.1)	21.6 (26.9)	11.2 (15.4)
computer_keyboard	47.3 (40.0)	71.8 (60.6)	78.3 (66.8)	21.9 (29.2)	11.9 (15.4)
dishwasher	79.6 (76.9)	92.2 (91.3)	93.6 (92.9)	11.6 (12.2)	7.9 (8.0)
display	56.0 (49.6)	78.5 (74.2)	83.5 (80.3)	18.0 (20.4)	10.0 (11.4)
earphone	29.2 (23.1)	54.4 (43.0)	64.4 (53.0)	28.9 (34.6)	20.0 (27.6)
faucet	35.6 (32.4)	58.8 (53.8)	68.4 (63.6)	26.3 (28.6)	17.5 (20.1)
file	71.1 (66.9)	87.4 (85.9)	89.8 (88.8)	14.2 (15.1)	8.6 (8.9)
guitar	45.0 (32.8)	67.8 (53.3)	74.7 (61.8)	22.9 (29.7)	12.8 (20.3)
helmet	36.6 (25.7)	66.4 (51.2)	75.8 (61.6)	22.6 (30.2)	15.1 (21.7)
jar	56.5 (55.5)	82.3 (81.8)	87.0 (86.7)	16.2 (16.5)	10.1 (10.3)
knife	40.4 (34.6)	61.0 (53.7)	68.6 (62.4)	25.5 (29.1)	15.5 (19.9)
lamp	46.9 (44.7)	71.5 (68.3)	78.4 (75.5)	21.1 (22.6)	12.1 (12.9)
laptop	61.3 (44.7)	83.7 (66.9)	88.1 (75.0)	15.7 (23.4)	9.6 (12.7)
loudspeaker	56.8 (55.2)	75.4 (74.5)	80.4 (79.9)	19.4 (19.8)	9.9 (10.1)
mailbox	54.8 (51.7)	75.5 (72.4)	80.4 (77.9)	19.6 (21.1)	10.2 (10.8)
microphone	52.1 (52.5)	76.7 (77.7)	82.6 (83.5)	18.7 (18.4)	10.7 (10.7)
microwave	76.6 (71.7)	90.7 (89.4)	92.6 (91.5)	12.3 (13.3)	8.1 (8.4)
motorcycle	18.1 (16.4)	36.9 (34.2)	46.6 (43.8)	38.1 (39.7)	32.9 (35.5)
mug	65.7 (60.5)	87.5 (85.1)	90.6 (89.2)	13.9 (15.0)	8.8 (9.4)
piano	52.8 (48.8)	71.4 (67.7)	76.3 (73.1)	22.4 (24.5)	10.6 (11.5)
pillow	37.7 (27.3)	71.4 (55.3)	81.0 (66.3)	20.3 (27.7)	14.2 (19.8)
pistol	35.4 (28.8)	58.0 (51.0)	66.7 (60.2)	27.1 (31.0)	17.5 (21.9)
pot	45.9 (45.2)	69.3 (68.5)	75.2 (74.4)	22.6 (23.0)	12.3 (12.5)
printer	59.1 (56.1)	78.1 (76.3)	82.7 (80.9)	18.6 (19.8)	9.6 (10.1)
remote_control	45.9 (42.0)	68.7 (64.8)	76.9 (73.9)	21.7 (23.3)	12.5 (14.1)
rifle	32.9 (30.4)	52.3 (48.5)	60.2 (56.1)	31.0 (33.7)	20.7 (23.9)
rocket	35.4 (28.9)	62.6 (53.0)	72.0 (63.2)	24.5 (28.9)	16.4 (20.7)
skateboard	46.6 (40.6)	68.2 (60.9)	74.6 (67.7)	23.4 (28.1)	12.3 (15.3)
sofa	48.1 (46.8)	69.1 (67.4)	74.4 (73.0)	24.0 (24.8)	11.7 (12.1)
stove	67.6 (66.2)	84.2 (83.7)	87.5 (87.2)	15.3 (15.6)	8.8 (8.9)
table	52.6 (50.8)	68.2 (66.1)	72.9 (71.1)	24.6 (25.8)	10.4 (10.9)
telephone	58.1 (57.6)	81.4 (81.2)	86.6 (86.6)	16.5 (16.5)	9.8 (9.8)
tower	51.1 (49.2)	71.3 (69.7)	76.7 (75.6)	21.8 (22.4)	11.0 (11.5)
train	41.1 (36.9)	61.6 (58.0)	68.4 (65.4)	26.5 (28.6)	14.7 (16.9)
vessel	31.8 (27.1)	55.2 (47.2)	64.3 (56.1)	29.1 (33.7)	19.3 (24.8)
washer	74.5 (74.9)	89.4 (89.3)	91.7 (91.4)	12.8 (12.9)	8.2 (8.2)

Table 3: Surface normal estimation performance for all classes under the V_n view sampling. The numbers outside the parenthesis denote the 'Learned' setting and the evaluation in the 'Induced' setting is reported in the parenthesis. Higher is better for the first three percent of 'good' pixels metrics and lower is better for last two three error metrics.

Metrics	**%GP** 11.25°	**%GP** 22.5°	**%GP** 30.0°	**Mean Error**	**Median Error**
airplane	23.2 (20.0)	46.4 (38.8)	56.9 (47.6)	32.7 (37.7)	24.9 (32.3)
ashcan	42.0 (42.3)	67.7 (67.1)	74.6 (73.8)	23.2 (23.6)	13.3 (13.3)
bag	35.6 (34.0)	62.6 (59.1)	71.7 (68.1)	24.9 (26.9)	16.1 (17.2)
basket	42.9 (42.9)	67.6 (65.7)	73.5 (71.3)	24.8 (26.0)	13.0 (13.2)
bathtub	39.8 (32.8)	66.6 (59.1)	73.8 (67.3)	24.6 (28.6)	14.1 (17.4)
bed	36.5 (34.0)	59.3 (56.9)	65.9 (63.8)	29.0 (30.5)	16.1 (17.6)
bench	33.3 (32.4)	56.4 (54.3)	64.0 (61.9)	29.4 (30.7)	18.1 (19.2)
bicycle	26.5 (25.6)	48.5 (47.8)	58.2 (57.9)	32.0 (32.2)	23.6 (24.1)
birdhouse	44.4 (42.1)	71.4 (68.5)	78.0 (75.8)	22.2 (23.4)	12.6 (13.3)
boat	25.7 (22.8)	49.2 (43.7)	58.5 (52.4)	32.6 (35.9)	23.1 (27.7)
bookshelf	40.0 (36.7)	62.7 (59.7)	68.4 (65.6)	28.1 (30.0)	14.3 (15.9)
bottle	57.6 (52.0)	88.1 (84.3)	93.0 (90.5)	13.4 (15.0)	10.0 (10.9)
bowl	42.1 (30.9)	69.4 (55.0)	76.4 (63.6)	22.6 (29.5)	13.2 (19.0)
bus	53.0 (46.4)	79.5 (74.5)	84.4 (80.4)	18.2 (21.1)	10.7 (12.0)
cabinet	55.7 (56.3)	79.5 (79.7)	84.1 (84.1)	18.2 (18.2)	10.3 (10.2)
camera	35.4 (34.9)	61.5 (59.6)	69.7 (67.6)	26.4 (27.7)	16.1 (16.7)
can	64.8 (53.9)	90.6 (81.7)	93.6 (86.6)	12.6 (16.9)	9.1 (10.6)
cap	49.5 (34.1)	79.0 (62.7)	83.9 (70.9)	18.9 (26.3)	11.3 (16.0)
car	33.7 (25.7)	63.1 (51.1)	72.6 (61.2)	24.4 (30.7)	16.2 (21.8)
cellular_telephone	50.8 (50.3)	78.6 (78.1)	84.8 (84.2)	17.8 (18.1)	11.1 (11.2)
chair	35.1 (33.1)	60.7 (57.7)	68.3 (65.4)	27.6 (29.2)	16.4 (17.6)
clock	41.7 (33.5)	67.3 (57.9)	73.9 (64.9)	24.2 (29.6)	13.5 (17.4)
computer_keyboard	39.6 (31.5)	71.1 (59.8)	80.0 (69.2)	20.5 (26.7)	13.8 (17.5)
dishwasher	66.9 (64.0)	89.2 (88.2)	91.9 (91.1)	13.4 (14.2)	9.0 (9.3)
display	38.6 (29.3)	67.1 (52.8)	75.1 (60.7)	23.3 (31.5)	14.3 (20.4)
earphone	28.1 (22.4)	53.1 (43.8)	62.8 (53.7)	30.1 (35.0)	20.6 (26.9)
faucet	31.3 (29.5)	58.2 (55.7)	68.2 (65.7)	26.8 (28.3)	18.2 (19.3)
file	59.9 (55.5)	83.1 (81.0)	86.4 (84.9)	16.8 (18.1)	9.7 (10.3)
guitar	38.2 (27.6)	64.6 (48.6)	72.4 (56.6)	24.4 (33.5)	15.0 (23.7)
helmet	36.3 (24.2)	68.6 (50.6)	78.4 (61.3)	21.5 (30.7)	14.9 (22.2)
jar	43.0 (42.7)	71.0 (70.7)	78.0 (77.7)	21.6 (21.7)	12.9 (13.0)
knife	32.9 (28.4)	56.2 (49.8)	65.3 (58.7)	28.2 (31.6)	18.6 (22.7)
lamp	38.3 (36.5)	65.9 (62.6)	73.6 (70.7)	24.1 (25.6)	14.6 (15.5)
laptop	46.2 (31.8)	75.9 (56.3)	82.0 (63.4)	20.1 (30.5)	12.1 (18.4)
loudspeaker	49.7 (47.0)	73.6 (71.9)	79.0 (77.8)	20.9 (22.0)	11.3 (12.0)
mailbox	42.4 (39.4)	68.3 (66.0)	76.0 (74.2)	22.5 (23.7)	13.3 (14.3)
microphone	44.6 (41.3)	74.2 (70.4)	81.4 (78.1)	20.1 (22.0)	12.5 (13.5)
microwave	65.4 (59.4)	87.9 (84.6)	90.8 (88.2)	14.2 (16.3)	9.0 (9.8)
motorcycle	17.9 (16.9)	37.7 (35.8)	48.0 (46.1)	37.4 (38.4)	31.7 (33.3)
mug	54.3 (49.7)	80.8 (78.5)	84.9 (83.5)	18.1 (19.2)	10.5 (11.3)
piano	40.2 (35.7)	65.9 (60.9)	72.8 (68.2)	25.2 (27.9)	14.0 (16.1)
pillow	33.4 (23.4)	66.7 (50.0)	77.3 (60.7)	22.3 (31.1)	15.8 (22.5)
pistol	30.9 (27.2)	57.9 (53.3)	67.7 (63.7)	27.1 (29.5)	18.3 (20.6)
pot	35.5 (33.8)	59.4 (58.2)	66.7 (65.7)	28.2 (28.8)	16.5 (17.3)
printer	46.0 (42.1)	72.1 (68.5)	77.8 (74.5)	22.2 (24.4)	12.2 (13.2)
remote_control	38.0 (33.3)	65.3 (60.2)	73.8 (69.7)	23.9 (26.1)	15.0 (17.3)
rifle	28.5 (27.1)	53.5 (51.3)	63.3 (61.0)	29.5 (30.6)	20.3 (21.7)
rocket	28.7 (26.6)	54.7 (50.8)	64.7 (60.4)	28.6 (31.0)	19.8 (22.0)
skateboard	45.8 (28.7)	75.9 (54.0)	82.3 (62.1)	19.9 (31.0)	12.1 (20.0)
sofa	37.6 (36.5)	62.2 (60.0)	68.7 (66.3)	27.7 (29.2)	15.0 (15.8)
stove	53.5 (51.4)	78.7 (77.4)	83.9 (82.7)	18.3 (19.2)	10.6 (11.0)
table	53.2 (46.4)	76.7 (72.0)	81.1 (77.1)	19.9 (22.8)	10.6 (12.1)
telephone	50.6 (50.5)	78.7 (78.9)	85.0 (85.4)	17.8 (17.7)	11.1 (11.1)
tower	39.8 (39.2)	65.8 (63.9)	73.0 (71.2)	24.5 (25.3)	14.1 (14.5)
train	34.5 (31.2)	60.1 (56.4)	68.3 (65.0)	27.2 (29.1)	16.6 (18.5)
vessel	26.4 (23.7)	50.0 (45.1)	59.0 (53.7)	32.3 (35.3)	22.5 (26.6)
washer	63.4 (65.0)	86.3 (87.0)	89.7 (90.2)	14.9 (14.5)	9.3 (9.1)

Table 4: Surface normal estimation performance for all classes under the V_d view sampling. The numbers outside the parenthesis denote the 'Learned' setting and the evaluation in the 'Induced' setting is reported in the parenthesis. Higher is better for the first three percent of 'good' pixels metrics and lower is better for last two three error metrics.