Neural Textured Deformable Meshes for Robust Analysis-by-Synthesis

Angtian Wang[1]* Wufei Ma[1]* Alan Yuille[1] Adam Kortylewski[2,3]

[1]Johns Hopkins University [2]University of Freiburg [3]Max-Planck-Institute for Informatics

Abstract

Human vision demonstrates higher robustness than current AI algorithms under out-of-distribution scenarios. It has been conjectured such robustness benefits from performing analysis-by-synthesis. Our paper formulates triple vision tasks in a consistent manner using approximate analysis-by-synthesis by render-and-compare algorithms on neural features. In this work, we introduce Neural Textured Deformable Meshes (NTDM), which involve the object model with deformable geometry that allows optimization on both camera parameters and object geometries. The deformable mesh is parameterized as a neural field, and covered by whole-surface neural texture maps, which are trained to have spatial discriminability. During inference, we extract the feature map of the test image and subsequently optimize the 3D pose and shape parameters of our model using differentiable rendering to best reconstruct the target feature map. We show that our analysis-by-synthesis is much more robust than conventional neural networks when evaluated on real-world images and even in challenging out-of-distribution scenarios, such as occlusion and domain shift. Our algorithms are competitive with standard algorithms when tested on conventional performance measures.

1. Introduction

Deep neural networks are typically designed to perform a single vision task and can achieve high performance on that task. However, humans are capable of performing multiple recognition tasks simultaneously and in a highly robust manner, *i.e.*, generalizing under occlusion or environmental changes. Cognitive studies suggest that the robustness of the human visual perception arises from the analysis-by-synthesis process [27, 42]. Current generative AI systems also employ the analysis-by-synthesis process by typically using a graphics pipeline, along with an explicit 3D representation of the object, such as a CAD model. These systems generate images of an object class and then search for the model parameters that best reconstruct a given test image. Recent research has shown that performing render-and-compare algorithms on neural features can significantly improve the robustness of these systems under partial occlusion and domain shift [37].

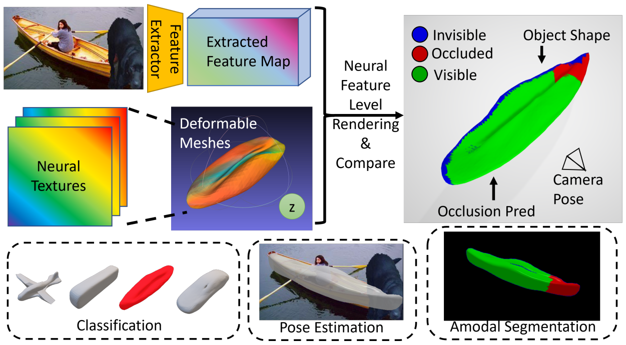

Figure 1. NTDM represents objects as category-level neural textured deformable meshes. For inference, we optimize camera pose, shape latent z, and object scale via gradient-based minimization of a reconstruction error between the extracted features and the rendered features. Using the optimized model parameters, NTDM predicts object pose, shape, category, and occlusion jointly and consistently in a robust manner.

Another limitation of current computer vision algorithms is that they are unable to recognize objects comprehensively in the same way that humans can, without being constrained by a particular scope. For example, the algorithm's predictions are limited to specific tasks, unlike humans who can identify objects in a broader context. While standard deep neural network approaches for multi-task learning typically involve adding multiple heads, where each head provides predictions for a specific task. However, this approach suffers from fundamental limitations, such as predictions for each task needing to be determined when designing network architecture, annotations for each task required for training, and lacking consistency in predictions from different branches [43]. On the other hand, analysis-by-synthesis ap-

[*]Joint first authors

proaches infer 3D scene parameters by reconstructing the input, which leads to a comprehensive recognition of the object with inherent consistency among vision tasks.

In this work, we introduce NTDM, a 3D geometry-aware neural network architecture that implements an analysis-by-synthesis approach to computer vision, and hence is able to predict multiple visual recognition tasks in a unified manner, while also being exceptionally robust (Figure 1). Our model builds on and significantly extends recent work on generative models of neural network features. Specifically, we build on the concept of neural mesh models [11, 37] that represent objects as meshes and learn a generative model of the neural feature activations at each mesh vertex. These models solve vision tasks like pose estimation and 3D-aware image classification through a render-and-compare process. The key advantage of performing render-and-compare on neural network features is that these can be trained to be invariant to instance-specific details, which makes the inference process efficient and robust. The core limitation that prevents these methods from predicting other vision tasks, such as segmentation, is that they assume a fixed mesh geometry, which simplifies the learning and inference process but prevents them from estimating the object boundary accurately.

In order to perform analysis-by-synthesis for both object pose and geometry in an efficient manner, we introduce the concept of *Deformable Meshes with Neural Textures* (NTDM). We present a framework for learning NTDM and describe the inference process that enables the model to perform multi-task visual recognition in a unified manner. At the core of our model is a deformable mesh geometry that is represented by a mesh template and a deformation field that is parameterized by a multi-layer perceptron (MLP) [44] and trained from a few CAD models of the object class (typically 4-10 models). Related works often represent deformable object geometries implicitly as a level-set in a volume of signed distances [29] or occupancies [24]. However, these representations are computationally expensive to render, while mesh representations in general can be rendered very efficiently and hence are preferable for render-and-compare approaches. We model the appearance of the object as a neural texture map, which is trained in a discriminative manner to enhance the classification performance while also avoiding local optima in the reconstruction loss. During inference, first a feature map is extracted using a CNN, and subsequently the 3D pose and deformation of the mesh is optimized via render-and-compare based on the reconstruction error between the rendered feature map and the target. After convergence, we perform image classification, pose estimation, and amodal segmentation using the optimized model parameters.

We evaluate NTDM on the PASCAL3D+ [41], the occluded-PASCAL3D+ [39] dataset, and the OOD-CV [46] dataset, which was explicitly designed to evaluate out-of-distribution generalization in computer vision models. Our experiments show that NTDM performs competitively to all baselines while being highly robust in OOD scenarios.

In summary, our main contributions are:

- We introduce NTDM, a neural network architecture that implements an analysis-by-synthesis approach and is hence able to perform multi-tasking robustly. NTDM is composed of a deformable mesh geometry that is parameterized by a template mesh, a neural field of shape deformations, and a discriminatively trained neural texture.
- We demonstrate the versatility of our network architecture on a variety of datasets, where it performs competitively to single-task models while also being highly robust in out-of-distribution scenarios.

2. Related Work

Category-level pose estimation. Category-level pose estimation estimates 3D orientations of objects in a certain category. A classical approach was to formulate pose estimation as a classification problem [25, 36]. Subsequent works can be categorized into two groups, keypoint-based methods and render-and-compare methods. Keypoint-based methods [12, 19, 20, 31, 47] first detect semantic keypoints and then solve a Perspective-n-Point problem to find the optimal 3D pose. Render-and-compare methods [3,40] predict the 3D pose by fitting a 3D rigid transformation to minimize a reconstruction loss. [11, 13, 37, 45] proposed feature-level render-and-compare that are invariant to intra-category nuisances and variations.

Amodal segmentation. Amodal segmentation aims to predict the region of both visible and occluded parts of an object. Related works on amodal segmentation often adopt a fully-supervised approach, with training supervisions coming from human annotations [6, 32] or synthetic occlusions [17, 21, 28]. Recent work [35] introduces a Bayesian approach that is trained on non-occluded objects only and does not require any amodal supervision. Moreover, our model takes a 3D-aware approach for amodal segmentation such that our probabilistic model is built on top of deformable object meshes. As a result, our model does not require any amodal segmentation annotations but achieves more accurate boundaries compared to baseline models.

Multi-tasking. Multi-task models are trained to solve multiple tasks simultaneously and are widely adopted in many areas [4, 5, 33]. It has been found to generalize better by leveraging domain-specific information contained in training signals of related tasks [2] and be more parameter-efficient [23]. Previous works usually rely on a multi-head architecture [9, 18] and despite models being supervised with auxiliary loss functions [14, 18], predictions from individual heads tend to be inconsistent, especially in out-of-distribution scenarios. Instead, we propose DMNT that substitutes multiple prediction heads with a 3D model of generative features and solves multiple tasks from a unified perspective.

Analysis-by-Synthesis. Our work is built on feature-level render-and-compare [37] which approximates the analysis-by-synthesis approach [7, 8] in computer vision. Analysis-by-synthesis approaches are found to enable efficient learning [38] and largely enhance robustness in out-of-distribution scenarios, such as partial occlusion [16, 22, 37, 39] and out-of-distribution textures and shapes [46]. Our DMNT extends the previous works with a deformable 3D representation of neural features that learns a more characteristic representation of the scene and allows the model to solve multiple tasks jointly.

3. Deformable Meshes with Neural Textures

In the following, we first introduce the definition of our deformable mesh representation and the neural texture 3.1. Subsequently, we define the probabilistic model of NTDM 3.2 and introduce the training 3.3 and inference pipeline 3.4.

3.1. Deformable Meshes and Neural Textures

We introduce **Deformable Meshes** Γ, which represent variable instances of an object category with a continuous deformation field on mesh vertices. Specifically, given a sphere template mesh Υ, the deformable mesh is defined as:

$$\Gamma(z) = \{s \cdot (v + \Psi(v, z)), v \in \Upsilon\} \tag{1}$$

where Ψ is an MLP that controls the mesh deformation [44] via displacement of each vertex v, and $s = [s_h, s_w, s_d]$ is the average scale of the deformable mesh, which is optimized during the training. The latent variable z controls the shape deformation of the mesh.

The mesh deformation network Ψ is trained with a set of training meshes $\{\Lambda_k\}$, *i.e.*, CAD models from the dataset. While the existing CAD models provide variable 3D geometries for each category, it is difficult to estimate the correspondence between them due to varying typologies and number of vertices. Thus, it is difficult to directly deform the CAD models via a common template. Instead, we propose to learn the correspondence among the provided meshes to build a deformable mesh representation. Specifically, we evaluate the distance between our deformable mesh with a specified latent z_k and a target mesh Λ_k via using the distance between vertex v to the mesh faces f:

$$d(\Gamma(z_k), \Lambda_k) = \sum_{v \in \Gamma(z_k)} \min_{f \in \Lambda_k} d(v, f) + \sum_{v \in \Lambda_k} \min_{f \in \Gamma(z_k)} d(v, f) \tag{2}$$

where z_k is a one-hot encoding vector of the index of the mesh k, *e.g.* $z_1 = [1, 0, ..., 0]$. To train the network Ψ and s, we minimize $\sum_{k=1}^{N} d(\Gamma(z_k), \Lambda_k)$. We apply consistency constraints on the surface normals and a Laplacian smoothing loss [26] on each $\Gamma(z_k)$ to regularize the shape (see supplementary for details).

We define the **Neural Textures** $\Theta \in \mathbb{R}^{b \times q \times q \times d}$, where q is the size of the feature map, b is the number of viewing bins, on the surface of the mesh Γ, which are stored as square feature maps that contain feature vectors on each pixel $\theta_{b,u,v} \in \mathbb{R}^d$ for each viewing bin. We denote the mesh surface as $\mathcal{S}$. The coordinate of (u, v) is defined via the polar coordinates of locations on the sphere template mesh, such that for each point on the surface, we can easily compute its corresponding (u, v) via Equation 1.

To avoid local minima during optimization of the shape and pose, the Neural Textures are trained to learn 3D discriminative features, such that the distances in features space are correlated to the distance between their locations in the 3D space, *i.e.*, features are similar to each other when near each other spatially, and vice versa. Additionally, the features are also learned to be spatially smooth via controlling the discriminability. The learned neural features provide a correct gradient on parameters when reconstructing the feature observations.

3.2. Analysis-by-Synthesis via Rendering Neural Textures

We formulate NTDM as a probabilistic generative model of neural feature activations. Given an input image I, we extract the features via a convolutional neural networks $\Phi(I) = F^l$. Then, we normalize the extracted feature activations $F = F^l / \|F^l\|$. We compute the object likelihood via:

$$p(F|\Gamma, \Theta, c, m, B) = \prod_{i \in \mathcal{FG}} p(f_i|\Gamma, \Theta, c, m) \prod_{i \in \mathcal{BG}} p(f_i|B) \tag{3}$$

where $\mathcal{FG}$ and $\mathcal{BG}$ denote the foreground and background, respectively, i donates the pixel on the image plane, m is the camera extrinsic parameters, B is a set of feature vectors that model backgrounds. The foreground feature likelihoods follow a Gaussian distribution:

$$p(f_i|\Gamma, \Theta, c, m) = \sum_b \frac{\alpha_b}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma^2}\|f_i - \theta^c_{u,v,b}\|^2\right) \tag{4}$$

where $(u, v) \in \mathcal{S}$ is the corresponding location on the object surface that projects onto the pixel i on the image plane. As Figure 3 shows, α_b is the viewing coefficient:

$$\alpha_b = \frac{exp(T \cdot d \cdot R_b(\mathbf{n}))}{\sum_b exp(T \cdot d \cdot R_b(\mathbf{n}))}, b \in \mathbb{B} \tag{5}$$

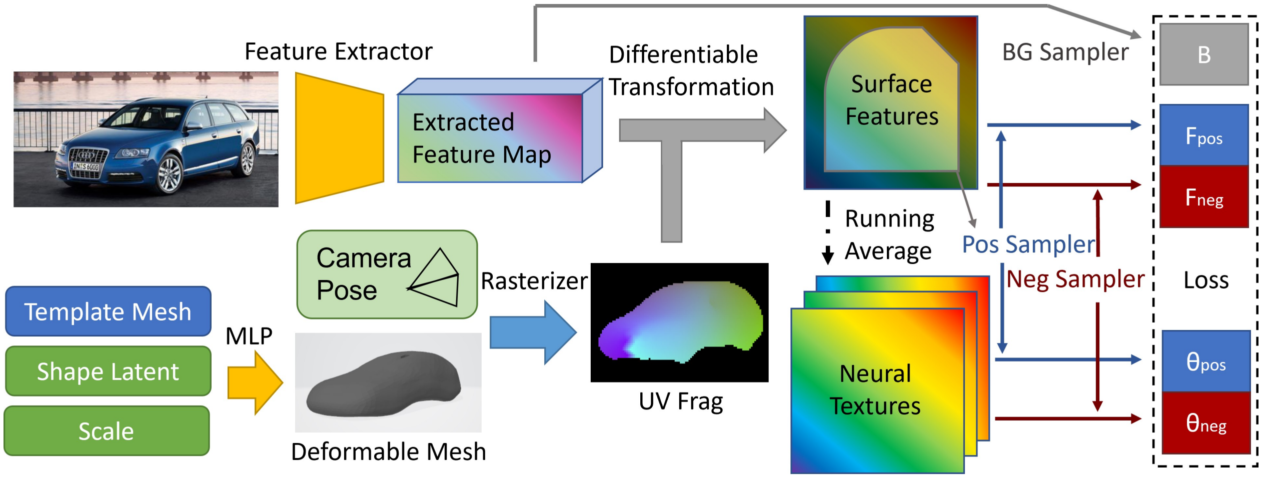

Figure 2. Training pipeline of NTDM. We first extract features from the training image. Then we compute the deformable mesh via deforming a sphere template using an MLP with ground-turth latent z inputs. A UV fragment is computed by rasterizing the deformable mesh under ground-truth camera. The surface features are computed via a differentiable transformation from image features given the UV fragments. Finally, we update the neural textures and compute the constrastive training loss via sampling positive and negative examples from image features for the surface features, neural textures, and background features.

where T is a softmax temperature, d is the normalized viewing direction from camera location to surface point, $\mathbb{B}$ is a set of rotations vectors R_b. Using each rotation vector, on each pixel (u, v), we compute the direction with surface normals $d^{'}_{(u,v)} = R_b(\mathbf{n}_{(u,v)})$. In practice, $\mathbb{B}$ is a fixed set.

The background feature likelihoods are computed by:

$$p(f_{i'}|B) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma^2}\|f_{i'} - \beta\|^2\right) \tag{6}$$

where $\beta \in \mathbb{R}^d$ is each features in B.

3.3. Training NTDM

Figure 2 shows the training pipeline of NTDM. In order to train the feature extractor and learn the neural texture jointly, we utilize the EM-type learning strategy introduced by CoKe [1], which iteratively trains the feature extractor and updates the stored neural features.

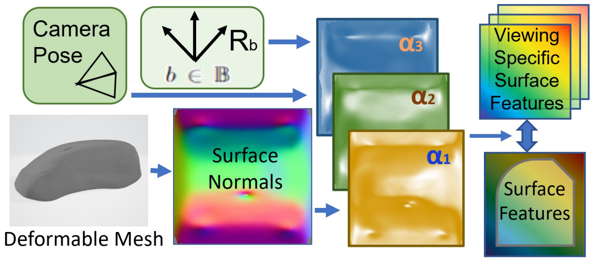

Figure 3. Conversion between viewing specific and non-specific surface features. Given a deformable mesh, we first compute the surface normals on mesh surface $\mathcal{S}$. Then we compute rotated surfaces by applying a set of rotations $R_b(\mathbf{n})$, and compute the viewing coefficients α_b with the dot product of $R_b(\mathbf{n})$ and viewing direction d. Viewing specific and non-specific surface features are converted via doting the coefficients α_b.

We first obtain the normalized feature from the image $F = \Phi_W(I)$, where W are the parameters in the feature extractor. Simultaneously, we compute the deformable mesh Γ with a ground-truth z and rasterize the mesh into the UV fragment $\mathrm{U} = \Re(m, \Gamma(z))$ under a ground-truth pose m. Then, we transform the features from the image plane into the surface features $f_{u_i,v_i} = f_i$ for all $i \in \mathrm{U}$. The transformation interpolates features in each pair of nearest pixels into quadrilateral regions onto the surface feature map, which also provides a mask of the visible region $\mathcal{V}$ of $\mathcal{S}$. Subsequently, we compute the view-specific features following Equation 5: $f_{u_i,v_i,b} = \alpha_b \cdot f_{u_i,v_i}$. We update Θ with those features from visible regions $\mathcal{V}$ of the transformed feature map using the momentum update strategy [1].

To learn the spatial discriminative features, we maximize the log-likelihood between the corresponding image feature and neural texture via minimize the feature distance:

$$\mathcal{D}_{ML}(F, \Theta_{\mathcal{S}}) = \sum_{(u,v)\in\mathcal{S}} \frac{1}{2\sigma^2}\|f^c_{u,v} - \theta^{b,c}_{u,v}\|^2$$

We also optimize the feature extractor to maximize the feature distance between features far from each other in the 3D space. To apply the loss, we first randomly sample a set of features from the visible part of the surface feature maps $f_{u,v}, (u, v) \in \mathcal{P} \subset \mathcal{V}$. Then, for each sampled (u, v), we samples a set of points $(u^{'}, v^{'}) \in \mathcal{N} \subset \mathcal{S}$ that $\|(u^{'}, v^{'}) - (u, v)\|^2 > \tau$ as negative training examples, where τ is the threshold for controlling the spatial discriminability. Then we compute the feature distance between corresponding image feature and neural texture:

$$\mathcal{D}_{Con}(F, \Theta_{\mathcal{S}}) = \sum_{(u,v)\in\mathcal{P}} \sum_{(u^{'},v^{'})\in\mathcal{N}} \|f^c_{u,v} - \theta^c_{u^{'},v^{'}}\|^2$$

To achieve the classification ability, for each image of $c \in \mathbf{C}$, we maximize the distance to a set features $\theta_{u',v'}, (u',v') \in \mathcal{M}$ from neural texture maps of other classes:

$$\mathcal{D}_{Class}(F, \Theta_{\mathcal{S}}^{\mathbf{C}}) = \sum_{c \in \mathbf{C}, c' \neq c} \sum_{(u,v) \in \mathcal{P}} \sum_{(u',v') \in \mathcal{M}} \|f_{u,v}^{c} - \theta_{u',v'}^{c'}\|^2$$

Similarly, we retain a set of features $B = \{\beta_j\}$, which stores the negative examples from the background of images. This allows us to maximize the objectness in contrast of the background:

$$\mathcal{D}_{Back}(F, B) = -\sum_{(u,v) \in \mathcal{P}} \sum_{j \in \mathcal{BG}} \|f_{u,v}^{c} - \beta_j^{c}\|^2. \tag{7}$$

In experiments, we find it hard to converge when we directly increase or decrease the feature distance by simply summing them together. Thus, we compute the overall loss using the contrastive loss [1]:

$$\mathcal{L}(F, \Gamma, \Theta, m, B) = \frac{\exp(\mathcal{D}(F, \Theta_{\mathcal{S}}))}{\exp(\mathcal{D}_{ML}(F, \Theta_{\mathcal{S}})) + W_{Con}\exp(\mathcal{D}_{Con}(F, \Theta_{\mathcal{S}})) + W_{Class}\exp(\mathcal{D}_{Class}(F, \Theta_{\mathcal{S}}^{\mathbf{C}})) + W_{BG}\exp(\mathcal{D}_{Back}(F, B))} \tag{8}$$

In the experiment, we set $W_{Con} = W_{Class} = 1$ and $W_{BG} = 0.1$.

3.4. Multi-Tasking via Robust Optimization

Figure 4 shows the inference pipeline of NTDM. We first extract features via the trained feature extract from the image $F = \Phi(I)$. Then we conduct maximum likelihood estimation of $p(F|\Gamma(z), \Theta, m, B)$. Specifically, given k initialized latent z_{init} (empirically we choose one-hot z_{init} corresponding to each ground-truth subtype), and an object instance scale $s^{'}$ (initialized with 1 in all direction), we compute the deformable mesh $\Gamma(z, s^{'})$. Using an initial camera pose $m = m_{init}$, we render the neural textured deformable mesh into a feature map F'. Then we conduct the foreground-background segmentation [37] on all pixels covered by the projected object $\mathcal{O}$, which indicates if the pixel belongs to $\mathcal{FG}$ or $\mathcal{BG}$ by comparing the feature similarity $\sum_i f_i \cdot f'_i$ and $\sum_i f_i \cdot \beta$. Subsequently, we compute the feature reconstruction loss:

$$\begin{aligned}\mathcal{L}_{rec} &= 1 - \ln p(F|\Gamma(z, s'), \Theta, m, B) \\ &= 1 - (\sum_{i \in \mathcal{FG}} f_i * f'_i + \sum_{i \in \mathcal{BG}} f_i \cdot \beta)\end{aligned} \tag{9}$$

We optimize camera pose m, shape latent z, and the object instance scale s' via gradient to minimize the reconstruction loss. We use PyTorch3D [34] to conduct the differentiable feature rendering and standard Adam optimizers [15].

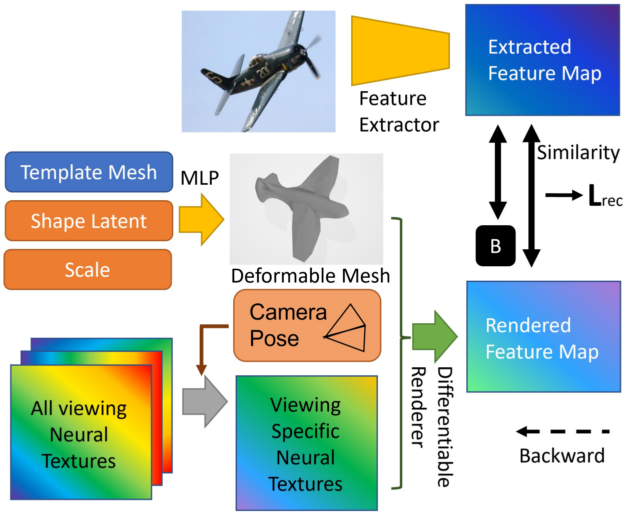

Figure 4. Inference pipeline of NTDM. First, a feature map F is extracted via the trained feature extractor. Next, the deformable mesh is initialized with a random latent and the average scale. Given a camera pose, we compute a viewing specific neural textures on mesh surface $\mathcal{S}$. Next, a differentiable renderer reconstructs the feature map $F^{'}$. Then Object scale, latent and camera pose are jointly optimized via gradient descent that minimizes the reconstruction error between F and $F^{'}$.

Once the optimization has converged, we obtain camera pose m directly. The shape prediction is obtained by computing $\Gamma(z, s')$. The visible object segmentation is obtained from $\mathcal{FG}$, while the amodal segmentation is obtained via $\mathcal{O}$. For classification, we compute the reconstruction loss $\mathcal{L}_{rec}$ with neural textured meshes of all classes under the predicted parameters. We conduct classification by finding the class with minimal reconstruction loss $\mathcal{L}_{rec}$.

4. Experiments

We evaluate the multi-tasking ability of NTDM under I.I.D., which indicates the training and evaluation are under a same data distribution, and O.O.D., which evaluates on data out of the training distribution. We compare NTDM with both task-specific approaches, *e.g.* pose estimation (Res50 [10], StarMap [47]), robust pose estimation (NeMo [37]), amodel segmentation (Bayesian [16]), and multi-task models (Multi-Task Mask R-CNN [9]). Our results show that NTDM achieves competitive performance on all tasks quantitatively, with a 3D interpretation of objects (figure 5).

4.1. Experimental Setup

We evaluate NTDM and baselines on PASCAL3d+ [41], Occluded PASCAL3d+ [39] and OOD-CV [46] datasets, following the data processing and training setup of NeMo.

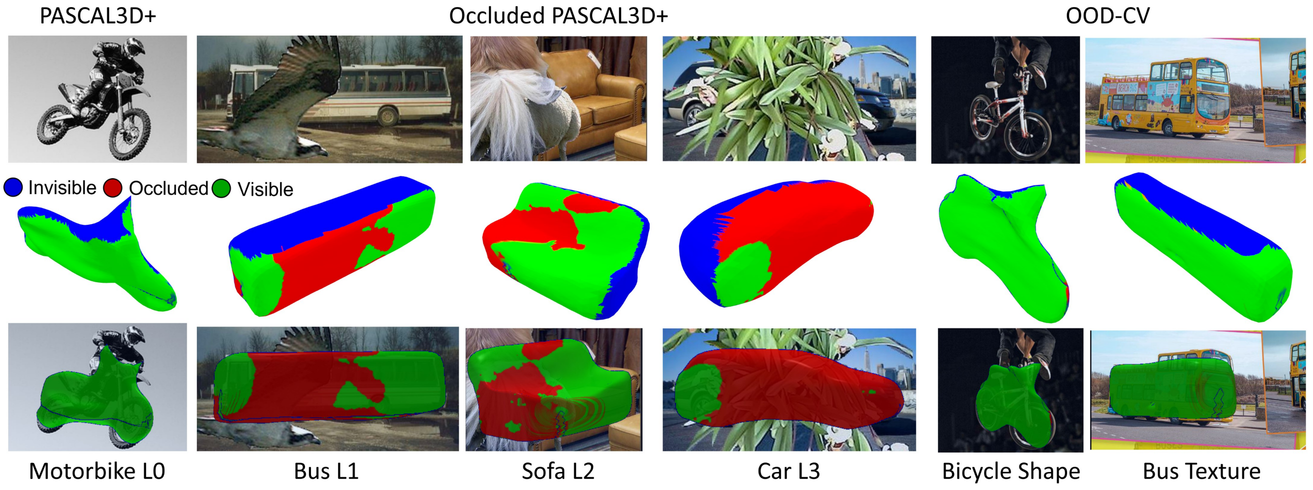

Figure 5. Visualizations for predictions of NTDM on PASCAL3D+ (1st column), Occluded PASCAL3D+ (2nd-4th column) and OOD-CV (5th-6th column). The first row shows the input images. The second row visualizes the shape estimation of the deformable mesh, with colors indicates areas that are visible, invisible or occluded. For the third row, we render the predicted mesh under the predicted pose and superimposed it onto the input image. Note rendering process also gives the amodal segmentation predictions.

4.1.1 Datasets

PASCAL3d+. We evaluate 3D object pose estimation, amodal segmentation and classification of NTDM and baselines methods on the PASCAL3D+ dataset [41]. The PASCAL3D+ dataset contains 11045 training images and 10812 validation images of 12 man-made object categories with class, segmentation and object pose annotations. PASCAL3D+ dataset also includes object CAD models. Note the CAD models are not accurately aligned to each object in image, but only give example of instances for each category (4-10 instance per category). We crop all training and evaluation images to center the object following NeMo.

Occluded PASCAL3d+. The Occluded PASCAL3d+ [39] dataset is an extension of the PASCAL3D+ with man-made occlusion, which is created by superimposing occluders collected from the MS-COCO dataset onto objects in PASCAL3D+. In our experiment, we evaluate on three occlusion levels with increasing occlusion noted as L1 to L3.

OOD-CV. The OOD-CV dataset [46] is a benchmark introduced to evaluate model robustness in out-of-distribution scenarios. It includes O.O.D. examples of 10 categories that cover unseen variations of nuisances including pose, shape, texture, context, and weather.

4.1.2 Implementation Details

NTDM uses the PyTorch3d [34] rasterizer to infer the UV fragments which indicate the 3D correspondence from the image coordinates to the object surface. We implement the feature transformation using CUDA and develop a PyTorch API as a differentiable function, which computes the gradient not only toward the features but also to the UV fragments. Note this function could also be used for differentiable texture extraction beyond our current work. For implementation details, please refer to the supplementary. For both training and inference, we use the Perspective camera with a fixed focal length. The template mesh is a geodesic sphere with 2562 vertices.

Training. We train NTDM on the PASCAL3D+ dataset for 800 epochs with Adam Optimizer and an exponential learning rate starting from 10^{-4}. NTDM use the same feature extractor as NeMo, which is a ResNet50 with two additional upsampling layers. The Neural Textures contain Feature Maps of 7 viewing bins, which are computed by rotation vectors with 60-degree angular distance from each other. Each feature map has a resolution of 256×256 with 128 channels. We update the Neural Textures Maps with a 0.9 momentum. During training, we use a positive sampler with 1000 selections and a negative sampler with 2000 selections on each training example. Note we don't include any data augmentation in the training process.

Inference. We use the differentiable rasterizer to infer the UV fragment and use the grid sample function to convert Neural Textures into feature maps since the image features have a lower resolution compared to neural textures. Following NeMo [37], to speed up the inference process, we initialize the optimization process with 144 different camera poses (12 azimuths, 4 elevations, 3 in-plane rotations) and latent. We compute the reconstruction loss for each initialized combination and pick the one with the minimum loss as the starting point of optimization. Then we update all parameters with an Adam optimizer for 300 epochs with $lr = 0.05$. The average inference time for pose and shape joint estimation on each image takes 6s on a single RTX Titan GPU.

	Pose Estimation ACC$_{\frac{\pi}{6}}$ ↑				Pose Estimation ACC$_{\frac{\pi}{18}}$ ↑				Amodal Segm IoU ↑			Classification ACC (%) ↑			
Occlusion Level	w/o	L1	L2	L3	w/o	L1	L2	L3	L1	L2	L3	w/o	L1	L2	L3
Res50-Pose	88.1	70.4	52.8	37.8	44.6	25.3	14.5	6.7	✗	✗	✗	✗	✗	✗	✗
StarMap [47]	**89.4**	71.1	47.2	22.9	59.5	34.4	13.9	3.7	✗	✗	✗	✗	✗	✗	✗
NeMo [37]	86.1	**76.0**	**63.9**	**46.8**	61.0	**46.3**	**32.0**	**17.1**	✗	✗	✗	✗	✗	✗	✗
Bayesian [35]	✗	✗	✗	✗	✗	✗	✗	✗	59.4	55.4	47.6	✗	✗	✗	✗
Res50-Class	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	**98.7**	**91.6**	**74.1**	42.6
MT Mask R-CNN	85.0	66.6	56.2	47.3	46.1	29.9	15.6	5.9	66.6	56.2	47.3	95.7	78.7	52.8	25.6
DMNT	86.4	74.8	61.0	40.2	**61.3**	44.8	30.1	13.7	**67.9**	**63.5**	**57.5**	94.1	85.0	67.8	**43.2**

Table 1. Comparison of multi-task performance on PASCAL3D+ dataset [41] (L0) and Occluded PASCAL3D+ dataset [39] (L1-L3). Best scores show with **bold**, and secondary best scores show with underline.

4.1.3 Baselines

NTDM learns a neural representation of 3D deformable meshes and can simultaneously predict object classes, 3D poses, and object boundaries. In experiments, we compare our model with baselines from individual tasks, as well as a multi-task extension of a multi-head deep neural network.

3D pose estimation. We compare NTDM with StarMap [47], NeMo [37], as well as standard deep neural network classifiers, ResNet-50 [10], that formulate pose estimation as a classification problem. StarMap is a keypoint-based approach, and NeMo learns contrastive features for render-and-compare. We follow the implementations in [37, 47] to train the ResNet-50 pose estimation model.

Amodal segmentation. We compare our model with Bayesian-Amodal [35], which extends deep neural networks with a Bayesian generative model of neural features. We use their official implementations to train on all categories in PASCAL3D+ dataset and evaluate on Occluded PASCAL3D+ dataset.

Classification. We also train a standard ResNet-50 as classification baseline using the PyTorch official version [30].

Multi-task deep neural network. To compare our model with traditional multi-head network architectures, we extend a Mask R-CNN model [9] with a pose estimation head that formulates object pose estimation as a classification problem. The model is end-to-end trained with ground-truth annotations including object classes, 3D poses, and object masks produced by known 3D meshes. For more implementation details refer to the supplementary materials.

4.2. Multi-tasking in I.I.D. scenarios

We are going through all tasks one by one in the following, but note that in contrast to most of our baselines, our model does all tasks jointly.

Pose Estimation. The 3D object pose is defined via three rotation parameters (azimuth, elevation, in-plane rotation) of the viewing camera. Following previous works [37, 47], we evaluate the error between the predicted rotation matrix and the ground truth rotation matrix: $\Delta\left(R_{pred}, R_{gt}\right) = \frac{\left\|\log m\left(R_{pred}^{T} R_{gt}\right)\right\|_{F}}{\sqrt{2}}$. We report the accuracy of the pose estimation under given thresholds, $\frac{\pi}{6}$ and $\frac{\pi}{18}$.

Amodal Segmentation. Amodal segmentation predicts the region of both the visible and occluded parts of an object. Following previous works [32, 35], we evaluate the average IoU between the predicted segmentation masks and the groundtruth segmentation masks.

Image Classification. We evaluate the classification ability of both NTDM and baselines. We report the top-1 accuracy between ground-truth class labels and predictions.

Results. Table 1 show the multi-task performance for both NTDM and baseline approaches. For the I.I.D setup, NTDM achieve comparative performance compared to the single task approaches and significantly better pose estimation ability compared to multi-tasking Mask R-CNN. NTDM achieves the *highest Pose accuracy under* $\frac{\pi}{18}$, which may benefit from the accuracy geometry compare to NeMo that uses cuboids as 3D geometry representation. Figure 5 shows both qualitative results of the pose estimation and segmentation, along with a 3D interpretation of the object produced by our model and visualized as a colored mesh. To obtain this colored mesh, we first conduct the $\mathcal{FG}$ and $\mathcal{BG}$ segmentation on the original image. We also compute the deformable mesh with the predicted latent z and render it under the predicted camera pose. Using our introduced transformation function, we transform the occlusion segmentation onto the surface of the mesh, and fill the uncovered areas as invisible. Finally, we convert the segmentation into RGB maps and save the colored deformable mesh as a textured mesh file (.obj). This visualization demonstrates that NTDM produces a *comprehensive 3D understanding* of the object in a human interpretable way, which makes it feasible for downstream tasks with ours pipeline.

4.3. Robustness in O.O.D. scenarios

We adopt the same evaluation protocol in Section 4.2 and evaluate multi-task performance on out-of-distribution scenarios in Occluded PASCAL3D+ and OOD-CV dataset.

Task & Metric Nuisance			Pose Estimation ACC$_{\frac{\pi}{6}}$ ↑			
	shape	pose	texture	context	weather	mean
Res50-Pose	50.5	34.5	**61.6**	**57.8**	**60.0**	**51.8**
NeMo [37]	49.6	35.5	57.5	50.3	52.3	48.1
MT Mask R-CNN	40.3	18.6	53.3	43.6	47.7	39.4
DMNT	**51.5**	**38.0**	56.8	52.4	54.5	50.0
Task & Metric Nuisance			Pose Estimation ACC$_{\frac{\pi}{18}}$ ↑			
	shape	pose	texture	context	weather	mean
Res50-Pose	15.7	**12.6**	22.3	15.5	23.4	18.1
NeMo [37]	19.3	7.1	**33.6**	21.5	30.3	21.7
MT Mask R-CNN	15.6	1.6	24.3	13.8	22.9	15.3
DMNT	**20.7**	**12.6**	32.6	**16.6**	**33.5**	**23.6**
Task & Metric Nuisance			Amodal Segmentation IoU ↑			
	shape	pose	texture	context	weather	mean
Res50-Pose	✗	✗	✗	✗	✗	✗
NeMo [37]	✗	✗	✗	✗	✗	✗
MT Mask R-CNN	46.6	44.5	51.3	44.7	46.0	46.3
DMNT	**48.0**	**48.9**	**54.7**	**49.4**	**53.8**	**51.0**

Table 2. Comparison of multi-task performance – pose estimation and amodal segmentation on OOD-CV dataset [46]. Best scores show with **bold**, and secondary best scores show with underline.

Setup	Pose $\frac{\pi}{6}$	Pose $\frac{\pi}{18}$	Seg IoU
full NTDM	**91.4**	**60.5**	80.7
w/o $\mathcal{BG}$	89.5	58.2	80.9
single α_b	90.0	58.3	80.1
fix shape	88.8	56.5	75.3
random	90.1	59.4	80.4
average shape $1/n$	90.2	59.7	**81.2**

Table 3. Ablation study on three object categories (sofa, bus, motorbike) from PASCAL3D+ dataset. We evaluate pose estimation accuracy and amodal segmentation IoU.

Results under occlusion. Table 1 shows the results of DMNT and baseline models on pose estimation, amodal segmentation, and image classification under occlusion. As we can see, NTDM achieves comparable or better performance compared to the state-of-the-art task specific models, while *significant outperforms multi-task Mask R-CNN*. Regarding pose estimation, DMNT outperforms regression-based baselines and StarMap, and achieves comparable performance with NeMo under occlusion. Moreover, with a deformable mesh of the object on top of the feature backbone, DMNT solves object boundaries from a holistic perspective and outperforms all amodal segmentation baselines by a wide margin. NTDM also perform object classification in a robust manner under partial occlusion.

Results under domain shifts. We evaluate the pose estimation and amodal segmentation performance on OOD-CV and investigate the robustness under domain shifts – shape, pose, texture, context, and weather. From Table 2, we can see NTDM achieves comparable pose estimation performance under $\pi/6$ accuracy and *outperforms all state-of-the-art models* when evaluating with a finer $\pi/18$ accuracy. Regarding amodal segmentation, NTDM also outperform multi-task Mask R-CNN. We suggest that the deformable mesh and the spatial discriminative features learned by DMNT can adapt well to domain shifts, *e.g.* shape and pose, and potentially useful in downstream tasks that requires highly robustness.

4.4. Ablation Study

As Table 3 shows, we evaluate the contribution of each proposed component. Specifically, we evaluate the model on three categories (sofa, bus, motorbike) from the PASCAL3D+ dataset. The *w/o $\mathcal{BG}$* setup indicates we only consider pixels in $\mathcal{FG}$ in reconstruction loss (equation 9). In the *single α_b* setup, we use only one viewing bin which keeps the d_b along the surface normal direction. In this case, there is only a single neural texture map for each deformable mesh. For *fix shape*, we use the average shape of deformable mesh to conduct inference, *i.e.*, fixing latent as the average latent. *random* and *average* show the result using different initialization of object shapes latent during inference, *i.e. random* initialized with a random vector, *average* initialized with the average object shape. The experiment demonstrates that all introduced components contribute to the final performance.

5. Conclusion

In this work, we propose NTDM, which conducts *multiple vision task simultaneously in a consistent and robust manner*. The core idea of NTDM is the neural textured deformable meshes, which conducts gradient-based optimization of the shape and scale of the object, as well as the camera parameters simultaneously via neural feature level analysis-by-synthesis. We introduce the learning pipeline for NTDM that learn deformable meshes, neural textures, and feature extractor together so that components can cooperate with each other to enhance the performance. Experiments demonstrate that NTDM produces a competitive performance compared to the task-specific approaches, and *extraordinary robustness* under occlusion and O.O.D. scenarios. Besides, we show that the predictions (shape, pose, occlusion) of NTDM can be visualized as colored meshes, which makes the decision process of NTDM interpretable and understandable. Due to time and space limitations, we are not able to explore NTDM in more downstream tasks. However, benefits from the generalization ability of the deformable 3D representation, NTDM can be easily extended to more vision tasks, *e.g.* part segmentation.

Acknowledgements. AW and WM gratefully acknowledge funding support from the Office of Naval Research with grants N00014-23-1-2641, N00014-21-1-2812. AK acknowledges support via his Emmy Noether Research Group funded by the German Science Foundation (DFG) under Grant No. 468670075.

References

[1] Yutong Bai, Angtian Wang, Adam Kortylewski, and Alan Yuille. Coke: Contrastive learning for robust keypoint detection. *WACV*, 2023. 4, 5

[2] Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997. 2

[3] Xu Chen, Zijian Dong, Jie Song, Andreas Geiger, and Otmar Hilliges. Category level object pose estimation via neural analysis-by-synthesis. In *European Conference on Computer Vision*, pages 139–156. Springer, 2020. 2

[4] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167, 2008. 2

[5] Li Deng, Geoffrey Hinton, and Brian Kingsbury. New types of deep neural network learning for speech recognition and related applications: An overview. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 8599–8603. IEEE, 2013. 2

[6] Patrick Follmann, Rebecca König, Philipp Härtinger, Michael Klostermann, and Tobias Böttger. Learning to see the invisible: End-to-end trainable amodal instance segmentation. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1328–1336. IEEE, 2019. 2

[7] Ulf Grenander. A unified approach to pattern analysis. In *Advances in computers*, volume 10, pages 175–216. Elsevier, 1970. 3

[8] Ulf Grenander. *Elements of pattern theory*. JHU Press, 1996. 3

[9] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 2, 5, 7

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5, 7

[11] Shun Iwase, Xingyu Liu, Rawal Khirodkar, Rio Yokota, and Kris M Kitani. Repose: Fast 6d object pose refinement via deep texture rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3303–3312, 2021. 2

[12] You-Yi Jau, Rui Zhu, Hao Su, and Manmohan Chandraker. Deep keypoint-based camera pose estimation with geometric constraints. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4950–4957. IEEE, 2020. 2

[13] Artur Jesslen, Guofeng Zhang, Angtian Wang, Alan Yuille, and Adam Kortylewski. Robust 3d-aware object classification via discriminative render-and-compare. *arXiv preprint arXiv:2305.14668*, 2023. 2

[14] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7482–7491, 2018. 3

[15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[16] Adam Kortylewski, Qing Liu, Angtian Wang, Yihong Sun, and Alan Yuille. Compositional convolutional neural networks: A robust and interpretable model for object recognition under occlusion. *International Journal of Computer Vision*, 129(3):736–760, 2021. 3, 5

[17] Ke Li and Jitendra Malik. Amodal instance segmentation. In *European Conference on Computer Vision*, pages 677–693. Springer, 2016. 2

[18] Peixuan Li, Huaici Zhao, Pengfei Liu, and Feidao Cao. Rtm3d: Real-time monocular 3d detection from object keypoints for autonomous driving. In *European Conference on Computer Vision*, pages 644–660. Springer, 2020. 2, 3

[19] Yunzhi Lin, Jonathan Tremblay, Stephen Tyree, Patricio A Vela, and Stan Birchfield. Keypoint-based category-level object pose tracking from an rgb sequence with uncertainty estimation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 1258–1264. IEEE, 2022. 2

[20] Yunzhi Lin, Jonathan Tremblay, Stephen Tyree, Patricio A Vela, and Stan Birchfield. Single-stage keypoint-based category-level object pose estimation from an rgb image. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 1547–1553. IEEE, 2022. 2

[21] Huan Ling, David Acuna, Karsten Kreis, Seung Wook Kim, and Sanja Fidler. Variational amodal object completion. *Advances in Neural Information Processing Systems*, 33:16246–16257, 2020. 2

[22] Wufei Ma, Angtian Wang, Alan Yuille, and Adam Kortylewski. Robust category-level 6d pose estimation with coarse-to-fine rendering of neural features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 3

[23] Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. *arXiv preprint arXiv:2106.04489*, 2021. 2

[24] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4460–4470, 2019. 2

[25] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Kosecka. 3d bounding box estimation using deep learning and geometry. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7074–7082, 2017. 2

[26] Andrew Nealen, Takeo Igarashi, Olga Sorkine, and Marc Alexa. Laplacian mesh optimization. In *Proceedings of the 4th international conference on Computer graphics and interactive techniques in Australasia and Southeast Asia*, pages 381–389, 2006. 3

[27] Ulric Neisser et al. Cognitive psychology. 1967. 1

[28] Khoi Nguyen and Sinisa Todorovic. A weakly supervised amodal segmenter with boundary uncertainty estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7396–7405, 2021. 2

[29] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019. 2

[30] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 7

[31] Georgios Pavlakos, Xiaowei Zhou, Aaron Chan, Konstantinos G Derpanis, and Kostas Daniilidis. 6-dof object pose from semantic keypoints. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 2011–2018. IEEE, 2017. 2

[32] Lu Qi, Li Jiang, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Amodal instance segmentation with kins dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3014–3023, 2019. 2, 7

[33] Bharath Ramsundar, Steven Kearnes, Patrick Riley, Dale Webster, David Konerding, and Vijay Pande. Massively multitask networks for drug discovery. *arXiv preprint arXiv:1502.02072*, 2015. 2

[34] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020. 5, 6

[35] Yihong Sun, Adam Kortylewski, and Alan Yuille. Amodal segmentation through out-of-task and out-of-distribution generalization with a bayesian model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1215–1224, 2022. 2, 7

[36] Shubham Tulsiani and Jitendra Malik. Viewpoints and keypoints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1510–1519, 2015. 2

[37] Angtian Wang, Adam Kortylewski, and Alan Yuille. Nemo: Neural mesh models of contrastive features for robust 3d pose estimation. *arXiv preprint arXiv:2101.12378*, 2021. 1, 2, 3, 5, 6, 7, 8

[38] Angtian Wang, Shenxiao Mei, Alan L Yuille, and Adam Kortylewski. Neural view synthesis and matching for semi-supervised few-shot learning of 3d pose. *Advances in Neural Information Processing Systems*, 34:7207–7219, 2021. 3

[39] Angtian Wang, Yihong Sun, Adam Kortylewski, and Alan L Yuille. Robust object detection under occlusion with context-aware compositionalnets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12645–12654, 2020. 2, 3, 5, 6, 7

[40] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2642–2651, 2019. 2

[41] Yu Xiang, Roozbeh Mottaghi, and Silvio Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *IEEE winter conference on applications of computer vision*, pages 75–82. IEEE, 2014. 2, 5, 6, 7

[42] Alan Yuille and Daniel Kersten. Vision as bayesian inference: analysis by synthesis? *Trends in cognitive sciences*, 10(7):301–308, 2006. 1

[43] Amir R Zamir, Alexander Sax, Nikhil Cheerla, Rohan Suri, Zhangjie Cao, Jitendra Malik, and Leonidas J Guibas. Robust learning through cross-task consistency. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11197–11206, 2020. 1

[44] Jason Zhang, Gengshan Yang, Shubham Tulsiani, and Deva Ramanan. Ners: Neural reflectance surfaces for sparse-view 3d reconstruction in the wild. *Advances in Neural Information Processing Systems*, 34:29835–29847, 2021. 2, 3

[45] Yi Zhang, Pengliang Ji, Angtian Wang, Jieru Mei, Adam Kortylewski, and Alan Yuille. 3d-aware neural body fitting for occlusion robust 3d human pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9399–9410, 2023. 2

[46] Bingchen Zhao, Shaozuo Yu, Wufei Ma, Mingxin Yu, Shenxiao Mei, Angtian Wang, Ju He, Alan Yuille, and Adam Kortylewski. Ood-cv: A benchmark for robustness to out-of-distribution shifts of individual nuisances in natural images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 2, 3, 5, 6, 8

[47] Xingyi Zhou, Arjun Karpur, Linjie Luo, and Qixing Huang. Starmap for category-agnostic keypoint and viewpoint estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 318–334, 2018. 2, 5, 7