Occlusion-Net: 2D/3D Occluded Keypoint Localization Using Graph Networks

N. Dinesh Reddy Minh Vo Srinivasa G. Narasimhan
Carnegie Mellon University
{dnarapur,mpvo,srinivas}@cs.cmu.edu

Abstract

We present Occlusion-Net[1], a framework to predict 2D and 3D locations of occluded keypoints for objects, in a largely self-supervised manner. We use an off-the-shelf detector as input (e.g. MaskRCNN [16]) that is trained only on visible key point annotations. This is the only supervision used in this work. A graph encoder network then explicitly classifies invisible edges and a graph decoder network corrects the occluded keypoint locations from the initial detector. Central to this work is a trifocal tensor loss that provides indirect self-supervision for occluded keypoint locations that are visible in other views of the object. The 2D keypoints are then passed into a 3D graph network that estimates the 3D shape and camera pose using the self-supervised reprojection loss. At test time, Occlusion-Net successfully localizes keypoints in a single view under a diverse set of occlusion settings. We validate our approach on synthetic CAD data as well as a large image set capturing vehicles at many busy city intersections. As an interesting aside, we compare the accuracy of human labels of invisible keypoints against those predicted by the trifocal tensor.

1. Introduction

Virtually any scene has occlusions. Even a scene with a single object exhibits self-occlusions - a camera can only view one side of an object (left or right, front or back), or part of the object is outside the field of view. More complex occlusions occur when one or more objects block part(s) of another object. Understanding and dealing with occlusions is hard due to the large variation in the type, number and extent of occlusions possible in scenes. As such, occlusions are an important reason for failure of many computer vision approaches for object detection [9, 14, 34, 16], tracking[49, 5, 44, 41], reconstruction [20, 19] and recognition, even today's advanced deep learning based ones.

The computer vision community has collectively attempted numerous approaches to deal with occlusions [12, 13, 26, 35] for decades. Bad predictions due to occlusions are dealt with as noise/outliers in robust estimators. Many methods provide confidence or uncertainty estimates to downstream approaches that need to sort out whether the uncertainty corresponds to occlusion. But it is hard to predict performance as they usually do not take occlusions explicitly into account.

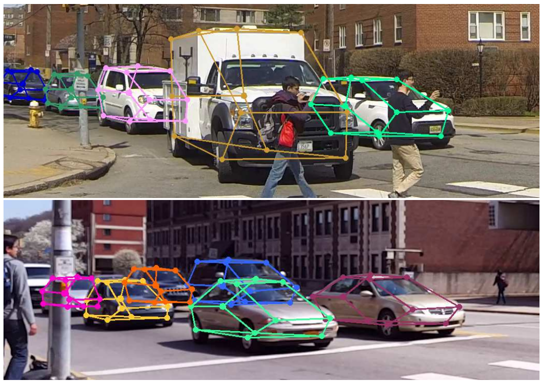

Figure 1: Accurate 2D keypoint localization under severe occlusion in our CarFusion dataset. Different colors depicts different objects in the scene.

On the other hand, occlusions are explicitly treated as missing parts in model fitting methods [50, 40]. These approaches have had better success as they exploit a statistical model of a particular type of object (e.g. car, human, etc.). But much remains to be done. For instance, severe occlusions, such as when a large part of an object is blocked, can result in poor fitting[52]. Further, often these approaches do not explicitly know which parts of an object are missing and attempt to simultaneously estimate the model fit as well as the missing parts.

In this work, we present an approach to explicitly predict 2D and 3D keypoint locations of the occluded parts of an object using graph networks, in a largely self-supervised manner. Our method receives as input, the output of any detector (e.g., using the MaskRCNN architecture [16]) that has been trained on a particular category of object with human supervision of *only visible keypoints* and their types (e.g., front, back, left, right). Implicitly, then, the key points that are not labeled are assumed to be invisible. This is the

[1] The code and dataset can be found at http://www.cs.cmu.edu/~ILIM/projects/IM/CarFusion/

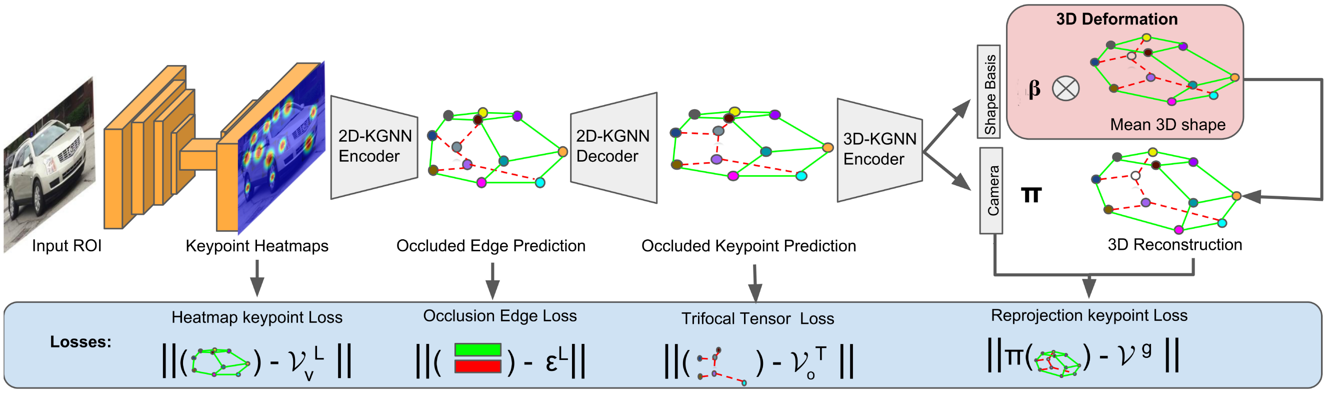

Figure 2: Occlusion-net: We illustrate the overall approach to training a network to improve localization of occluded keypoints. The input is a ROI region from any detector, which is passed through multiple convolutional layers to predict the heatmaps with a confidence score. These confidences are passed through a graph encode-decoder network and trained using multi-view trifocal tensor loss for localization of occluded 2D keypoints. The output from the decoder is passed through a 3D encoder to predict the shape basis and the camera orientation. This network is a self-supervised graph network and trained using reprojection loss with respect to the 2D decoder output.

only human supervision used in this work. The detector usually provides an uncertainty of all key point locations. We first show that the distribution of the uncertainties for visible and occluded points overlap significantly, making it hard to predict which key points are occluded at test time. To address this issue, we design an encoder-decoder graph network that first predicts which edges have an occluded node, and then localizes the occluded node in 2D in the decoder. Visible or invisible edge classification is trained using the implicit non-labeled supervision of occluded points.

We then train the decoder graph network to localize invisible keypoints using multiple wide-baseline views of objects. Our observation is that while some parts may be missing in one view, they are visible and labeled in another view. But how do we provide supervision for a hidden point location in a view? We use two views where a keypoint is seen (and labeled by humans) and compute the trifocal tensor using camera matrices to predict its location in the view where the keypoint is occluded. We call this the **Trifocal tensor loss**, which is minimized to correct the 2D keypoint positions from the initial detector. Compared to other approaches that use multiple views [38, 32, 37], our approach explicitly predicts occluded keypoints.

The predicted 2D keypoints (both occluded and visible) are then used in a graph network to estimate the 3D object shape and the camera projection matrix. Similar to previous work [52, 39], we will estimate the parameters of a shape basis computed a priori of the object of interest. The training is performed in a self-supervised way by minimizing the reprojection loss i.e. error between the reprojection and the predicted 2D keypoint locations. We train the entire pipeline, called Occlusion-net, end-to-end with the aforementioned losses.

We evaluate our approach on images of vehicles captured at busy city intersections with numerous types and severity of occlusions. The dataset extends the previous CarFusion dataset [32] to include many more city intersections, where 18 views of the intersection are simultaneously recorded. A MaskRCNN car detector is trained using 100000 cars, with human labeled visible keypoints to produce a strong baseline for our method to compare to and build upon. Our Occlusion-net significantly outperforms (about 10%) this baseline across many metrics and performs well even in the presence of significant occlusions (see Figure 1). As an interesting exercise, we also show a comparison of the trifocal loss against human labeling of the 2D occluded point locations and observe that humans label around 90% of the points to lie within the acceptable range of error. We also evaluate our approach on a large synthetic CAD dataset, showing similar performance benefits and improvements of up to 20% for occluded keypoints. Our network is efficient to train and can localize keypoints in 2D and 3D in real-time (more than 30 fps) at test-time. While we have demonstrated our approach on vehicles, the framework is general and applies to any object category.

2. Related Work

Occlusion Detection: While there has been significant progress in predicting the visible keypoints by using part detectors learned from CNNs [33, 42, 26, 2, 27, 46], most of these methods fail short to precisely localize occluded keypoints. Using synthetic data, Moreno et al. [31] show that such occlusion modeling is crucial. To address this problem, many methods employ active shape models [6] for vehicle detection under occlusion [51, 52, 43]. However, these methods only model self-occlusions and omit often seen occlusions by other objects. Recently, [37, 32] propose a multi-view bootstrapping approach to generate accurate CNN training data when precise human labeling is not possible. However, their methods are trained in stages and do not explicitly model the interaction between visible and occluded points. Most related to our work, [25] only incorporates intermediate keypoint supervisions from CAD model during training. Interestingly, they show that training such a model on synthetic images can generalize to real images. We train our model on real images and incorporate multiview constraints to propagate ground truth visible keypoints from multiple views to supervise occluded points.

Graph Neural Networks: Modeling keypoints as a graph problem can be dated back to the first attempt at scene understanding [11, 30]. Multiple works have built on this graph representation and solved pose using belief propagation [10, 36]. Recently, [8, 21, 1, 17, 7] have extended classical graphical modeling to a deep learning paradigm and showed better modeling capability for unstructured data. Based on the success of these methods on the graph classification tasks, multiple recent works have extended the methods to address multiple 3D problems like Shape segmentation [48], 3D correspondence [28] and CNN on surfaces [29]. We model keypoint prediction as a deformable graph that is learned using multi-view supervision.

3. Occlusion-Net

Occlusion-Net consists of three main stages - visible keypoints detection, occluded 2D keypoint localization and 3D keypoint localization networks - as shown in Figure 2. The 2D-Keypoint Graph Neural Network deforms the graph nodes to infer the 2D image locations of the occluded keypoints. The 3D-Keypoint Graph Neural Network localizes the 3D keypoints of the graph using a self-supervised training procedure. We combine these networks to accurately predict the 3D and 2D keypoint locations. Each of these stages is described in the following sections.

3.1. 2D-Keypoint Graph Neural Network

The 2D-Keypoint Graph Neural Network(2D-KGNN) consists of three components: initial keypoint heatmap prediction, a graph encoder to model the occlusion statistics of the graph, and a graph decoder infering the 2D locations of the occluded keypoints. We use the heatmap based methods [16][33] to compute the location of all the keypoints in an image. The input to the graph network consists of k keypoints, which are further categorized as v visible keypoints and o invisible/occluded keypoints. We denote the vertex of the graph as $\mathcal{V} = (\mathcal{V}_1, ..., \mathcal{V}_k)$ for k keypoints. The relationship between all nodes is encoded in the edge $\mathcal{E}_{ij} = \{\mathcal{V}_i, \mathcal{V}_j\}$, where

$$\mathcal{E}_{ij} = \begin{cases} 1, & \text{if } i \in v \; and \; j \in v \\ 0, & \text{otherwise} \end{cases}$$

We also denote $\mathcal{V}^l$ as labeled keypoint annotations and $\mathcal{V}^g$ as keypoints predicted from 2D-KGNN, respectively.

2D-KGNN Encoder: Occluded Edge Predictor The 2D keypoint graph network (2D-KGNN) needs to infer the locations of the occluded keypoints (or, edges $\mathcal{E}_{ij}$) from the keypoint heatmaps. We convert the heatmap into a graph by encoding the location and confidence of each keypoint into a node feature. The feature for keypoint i, can be more formally represented as $\mathcal{V}_i = \{x_i, y_i, c_i, t_i\}$, where (x_i, y_i) is the location, c_i is the confidence and t_i is defined as the type of the keypoint. Since, we do not know the underlying graph, we use the GNN to predict the latent graph structure. The encoder is modeled as $q(\mathcal{E}_{ij}|\mathcal{V}) = softmax(f_{enc}(\mathcal{V}))$ where $f_{enc}(\mathcal{V})$ is a GNN acting on the fully connected graph produced from the heatmaps. Given the input graph our encoder computes the following message passing operations to produce the occlusion statistics:

$$h_j^1 = f_{enc}(\mathcal{V}_j) \tag{1}$$

$$v \rightarrow e : h_{(i,j)}^1 = f_e^1([h_i^1, h_j^1]) \tag{2}$$

$$e \rightarrow v : h_j^2 = f_v(\sum_{i \neq j} h_{(i,j)}^1) \tag{3}$$

$$v \rightarrow e : h_{(i,j)}^2 = f_e^2([h_i^2, h_j^2]) \tag{4}$$

In the above equations, h^t denotes the t^{th} hidden layer of the network, while v and e denote the vertex and edge of the graph. Here, $v \rightarrow e$ shows a convolution operation from vertex to edge, while $e \rightarrow v$ represents the operation from edge to vertex. The functions $f()$ are implemented as fully connected layers. The edge loss for this encoder is the cross-entropy loss between the predicted edges and the ground truth edges, given as:

$$L_{Edge} = - \sum_{i,j \in k} \mathcal{E}_{ij} log(\mathcal{E}_{ij}^l) \tag{5}$$

The $\mathcal{E}_{ij}^l$ is the visibility statistics for each edge computed from the labeled keypoints.

2D-KGNN Decoder: Occluded Point Predictor The decoder predict consistent 2D keypoint locations of the occluded keypoints from the erroneous initial graph and the edges predicted from the encoder. This can mathematically be represented as estimating $P_\theta(\mathcal{V}^g|\mathcal{V}, \mathcal{E})$, where $\mathcal{V}^g$ represents the output graph from the decoder and $\mathcal{E}$ is the input from encoder, while $\mathcal{V}$ is the graph from the initial heatmap. The following message passing steps are computed on the graph network:

$$v \rightarrow e : h_{(i,j)} = \sum_p \mathcal{E}_{ij,p} f_e^p([\mathcal{V}_i, \mathcal{V}_j]) \tag{6}$$

$$e \rightarrow v : \mu_j^g = \mathcal{V}_j + f_v(\sum_{i \neq j} h_{(i,j)}) \tag{7}$$

$$P_\theta(\mathcal{V}^g|\mathcal{V}, \mathcal{E}) = \mathcal{N}(\mu_j^g, \rho^2 I) \tag{8}$$

Here $\mathcal{E}_{ij,p}$ denotes the p-th element of the vector $\mathcal{E}_{ij}$. An important thing to observe is the current state is added into Eq. 7, so inherently the model is learning to deform the keypoints i.e predict the difference $\Delta\mathcal{V} = \mathcal{V}^g - \mathcal{V}$. Further in Eq. 7, μ is the mean location predictor and $\mathcal{N}$ produces the probability of the locations. We only minimize the distance between the predicted and ground truth occluded points in this network using a trifocal tensor loss.

Trifocal Tensor Loss. We exploit multiple views of the object captured "in the wild" to estimate the occluded keypoints. The assumption is that the keypoints occluded in one view are visible in two or more different views. Thus, the trifocal tensor [15] can transfer the locations in the two visible views to the occluded view. Then, the loss for each occluded keypoint is computed as:

$$L_{Trifocal} = \sum_{j \in o} [\mathcal{V}_j^g]_\times (\sum_i (\mathcal{V}')_j^i T_i) [\mathcal{V}_j'']_\times, \qquad (9)$$

where i represents the three views considered for the trifocal tensor T, $\mathcal{V}_j^g$ is the prediction from the decoder for the occluded keypoint j in the current view, and $\mathcal{V'}_j$ and $\mathcal{V''}_j$ are the annotated keypoints j in two different views. We computed T using the camera poses in the object reference frame. In our setting, since the object (vehicle) is rigid, the two visible views could come from any camera viewing the same object at any other time instants.

3.2. 3D-Keypoint Graph Neural Network

Given the graph from the 2D-KGNN decoder, the 3D-keypoint graph neural network encoder predicts a 3D object shape W and the camera projection matrix π. This encoder takes as input the graph and predicts the 3D location of the all the keypoints using a self-supervised projection loss. Mathematically, this is formulated as $q(\beta, \pi|\mathcal{V}) = f_{enc}(\mathcal{V})$, where, β are the deformation coefficients of PCA shape basis of the object and π is the camera projection matrix.

Shape Basis: We model the shape as a set of 3D keypoints corresponding to the predicted 2D keypoints. We compute the mean shape b_0 and n principal shape components b_j and corresponding standard deviations σ_j, where $1 \leq j \leq n$, using the 3D repository of the object [3] with annotations of 3D keypoints from [26]. Given the shape bases, any set of deformable 3D keypoints can be represented as a linear combination of the n principal components β as $W = b_0 + \sum_{k=1}^{n} \beta_k * \sigma_k * b_k$.

Camera Projection Matrix: Let $\pi(W)$ be the function that projects a set of 3D keypoints W onto the image coordinates. We use the perspective camera model and describe π as a function of the camera focal length f, the rotation q, represented as quaternion, and translation t of the object in the camera coordinate frame [15]. We assume the principle point of the camera is at the origin. To account for the normalization of the image to a square matrix from the original dimensions, we re-scale the projected 2D points by $s = w/h$, where w and h denote the width and height of the input image (see [22] for further details).

Keypoint Reprojection Loss: We train the 3D-Keypoint Graph network in a self-supervised manner using the reprojection loss, i.e. the difference between the projected 3D keypoints and the keypoints computed from the 2D-KGNN:

$$L_{Reproj} = \sum_{j \in k} ||\pi(W_j) - \mathcal{V}_j^g||^2 \qquad (10)$$

The use of the 3D basis shape allows explicit enforcement of 3D symmetry which provides further constraints for the 2D keypoint estimation via the reprojection loss.

3.3. Total Loss

Our Occlusion-Net is trained to minimize the sum of the aforementioned losses:

$$L = L_{Keypoints} + L_{Edge} + L_{Trifocal} + L_{Reproj}, \quad (11)$$

where, $L_{Keypoints}$ is the cross-entropy loss over a t^2-way softmax output between the predicted keypoints and the ground truth labels [16]. Here, t is the number of keypoints.

4. Experimental Results

We demonstrate the ability of our approach to infer occluded keypoints and 3D shape from a single view on the new and challenging CarFusion dataset. We first describe this dataset in section 4.1. We then perform ablative analysis of the algorithm in Section 4.2. Finally, we show qualitative comparisons against the state of art Mask-RCNN [16] detector in section 4.3. For a fair comparison, we retrain this baseline model on our dataset. In the evaluation metrics, 2D-KGNN refers to the output after the decoder layer and 3D-KGNN refers to the projections of predicted 3D keypoints onto the image.

4.1. Datasets

Car-render Self-occlusion dataset: We use the 472 cars sampled from shapenet [4] and 3D annotated by [26]. We select 12 keypoints from the annotated 36 keypoints and render them from different viewpoints. The viewpoints are randomly selected on a level 5 Icosahedron, at varying focal lengths and distances from the object. We use 300 synthetic CAD models for training, 72 for validation and 100 for testing. We project the 3D keypoint annotations of the CAD model with visibility. we trace a ray toward the object from a pixel and check if the first intersection is close to the ground truth location to determine visibility.

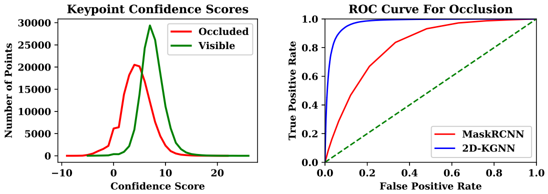

Figure 3: We analyze the need for a 2D-KGNN encoder. The left image shows the confidence score of the heatmaps from the baseline method (the distribution is colored based on Ground Truth visibility). The right image shows the ROC curve of the predictions from graph encoder and baseline. At 0.1 false positive rate, the baseline returns 0.5 true positive rates compared to 0.8 of the 2D-KGNN.

CarFusion dataset: To model a wide range of real occlusions, we collect an extensive dataset captured simultaneously by multiple mobile cameras at 60fps at 5 crowded traffic intersections (extending previous work [32]). This extended dataset consists of 2.5 million images out of which 53000 images were sampled at uniform intervals from each video sequence. Approximately, 100000 cars detected in these images were annotated with 12 keypoints each. Each annotation contains the visible and occluded keypoint locations on the car. We do not use the occluded keypoints for training the Occlusion-Net. We selected four annotated intersections to train the network while using one intersection to test it, which split the annotation data into 36000 images for training and 17000 for testing. We further compute 90-10 train validation split on the training data to validate our training algorithm. The dataset was completely captured "in the wild" and contains numerous types and severity of occlusions.

Preprocessing: Computing the trifocal loss requires the virtual camera poses in the object frame. For every image, the virtual pose is estimated by solving a PnP [23] between the visible keypoints and the 3D points computed from [32].

4.2. Quantitative Evaluation

We compare our approach with other state-of-the-art keypoint detection networks. We use the PCK metric [47] to analyze both the 2D and the 3D occluded keypoint locations. According to the PCK metric, a keypoint is considered correct if it lies within the radius αL of the ground truth. Here L is defined as the maximum of length and width of the bounding box and $0 < \alpha < 1$. To evaluate the 3D reconstruction, we project the reconstructed keypoints into their respective views and compute the 2D PCK error.

Occlusion Prediction: We demonstrate that the confidence scores computed using MaskRCNN is insufficient to predict occlusions. The left image in Fig 3 shows the distributions of confidence scores of occluded and visible points. These distributions overlap significantly making it hard to distinguish occluded points from visible points. In contrast, by modeling a graph network to exploit relative locations of the keypoints, we observe a significant boost in the accuracy of occlusion prediction as seen from the right image in figure 3. We observe an AUC of 0.83 with MaskRCNN, whereas 2D-KGNN gives an AUC of 0.95.

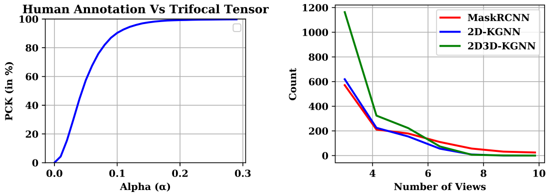

Figure 4: On the left, we show accuracy of human annotations with respect to geometrically obtained keypoints. We observe that most of the keypoints are labeled within $\alpha = 0.1$ PCK error. On the right, count of multi-view correspondences of keypoints predicted using different methods. When few views are available, the occluded points predicted by Occlusion-Net provide much more correspondences to improve multi-view reconstruction.

Evaluations of visible points: We show evaluation of our network with respect to existing visible keypoint estimation methods. Both 3D-KITTI[24] and PASCAL3D+ [45] datasets have annotations only for *visible keypoints* and do not contain occluded point annotations or multiple views to directly evaluate our method. The 2D keypoint predictions in [24] are evaluated only on visible keypoints and the 3D model is evaluated by fitting only visible keypoints on objects that are not truncated or occluded by other objects (”Full” in their table). Our model has *not* been trained on either of these datasets or the CAD dataset from [24]. Table 1 compares our method against those on the annotated 2D *visible* points in 3D-KITTI. Table 1 also shows the evaluation against the ground truth 3D model for the ”Full” (unoccluded) case - the only case mentioned in [24]. We observe that our approach outperforms the other methods for two categories .i.e. Truncation and oth-Occlusion. This can be attributed to the fact that our dataset models a range of occlusion types and severity.

Importance of 3D-KGNN: The 3D pose computed is useful for traffic analysis (speed, flow) and understanding/visualizing activity at busy city intersections. 3D-KGNN can also be used to find correspondence across views for multi-view reconstruction, especially when there are very few views available and the keypoints may be occluded. Figure 4 demonstrates that 3D-KGNN finds significantly more inliers for multiview correspondence compared to 2D-KGNN or MaskRCNN.

Method	2D					3D	yaw(Error)
	Full	Truncation	Car-Occ	Oth-Occ	All	Full	Full
[18]	88.0	76.0	81.0	82.7	82.0	NA	
[52]	73.6	NA				73.5	7.3
[25]	**93.1**	78.5	**82.9**	85.3	85.0	**95.3**	2.2
Ours	89.73	**87.41**	81.68	**86.45**	**88.8**	93.2	**1.9**

Table 1: PCK Evaluation[α=0.1] and comparison of Occlusion-Net on 2D *visible* keypoints annotated in KITTI-3D. Full denotes unoccluded cars, Truncation denotes cars not fully contained in the image, Car-Occ denotes cars occluded by cars, and Oth-Occ denotes cars occluded by other objects. All represents combining the statistics for all the occlusion categories. Our method outperforms in most of the occlusion categories. The 3D keypoint localization (last two columns) in [25] is only evaluated on Full.

Human Annotation vs Geometric Prediction: The CarFusion dataset has annotated keypoints for occluded points as well as the visible points across multiple views. Thus, as an interesting aside, we evaluate the accuracy of hand-labeled occluded points with respect to those obtained using the trifocal tensor, as shown in Figure 4. We observe that at $\alpha = 0.1$, nearly 90% of the hand-labeled keypoints lie within the region of the geometrically consistent keypoints.

Accuracy Analysis: Figure 5 depicts the change in accuracy with respect to Alpha on Car-render dataset. We show four different plots with different occlusion configuration, ranging from 3 (very less occluded) to 9 (highly occluded) invisible points out of 12 keypoints in total. We observe that our method outperforms the baseline method in all configurations for occluded keypoints. At α=0.1 we observe a boost of 22% for 3 invisible points and 10% for 9 invisible points. Figure 6 shows the change in accuracy with respect number of occlusions for Car-render dataset. We plot the graph for two different value of α and observe that 2D graph method is more stable with increasing occlusion compared to the 3D-KGNN. We show similar accuracy vs. alpha plots on CarFusion dataset in Figure 8. We observe that with increasing occlusions our method shows higher accuracy improvement compared to the baseline MaskRCNN. At $\alpha = 0.1$ we nearly gain a boost of at least 6% in all the occlusion categories and nearly 12% boost for 5 occluded points. Figure 9 depicts the change in accuracy with increasing number of occluded points on CarFusion dataset. For the case of 4 invisible points configuration, our approach is nearly 25% higher compared to the baseline. To conclude we observe that the accuracy of KGNN on occluded points is higher than using the baseline method.

Robustness Analysis: We analyze the effect of adding error to input locations of the graph to analyze the robustness of the learned model. Figure 10 shows the accuracy with respect to different Gaussian error added to the input graph. We observe that 3D-KGNN is more stable with increasing error while 2D-KGNN performs well for highly occluded points but falls steeply with increasing error in input.

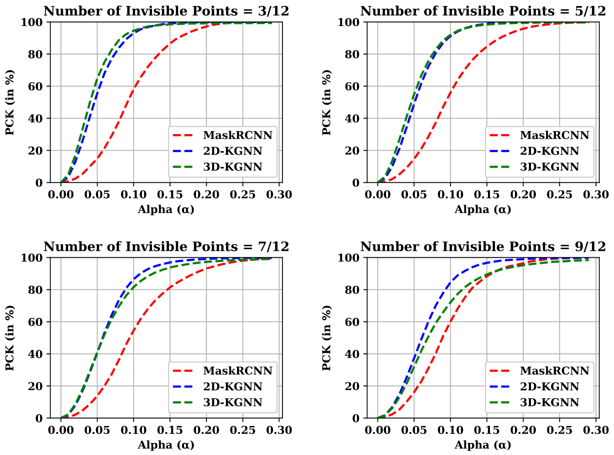

Figure 5: Accuracy with respect to different alpha values of PCK for the Car-render dataset. Graph based methods (2D/3D) outperform the MaskRCNN trained keypoints for all the occlusion types. Specifically at alpha=0.1 we observe an increase of 22% for cases with 3 invisible points and 10% in case of 9 invisible points (out of 12 keypoints).

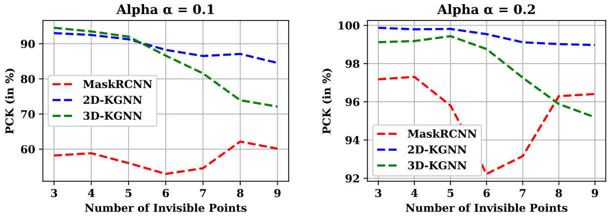

Figure 6: Accuracy plots with varying number of occluded keypoints on the Car-render dataset. Graph based methods (2D/3D) outperform the baseline (in red) in the case of $\alpha = 0.1$. For a more conservative alpha, the performances are comparable. The 2D KGNN plots in both the alpha scenarios have a variance of 5% and are robust to occlusion, compared to the 3D KGNN plot (15%) and the baseline MaskRCNN plot (25%).

4.3. Qualitative Evaluation

In this section, we analyze the visual improvements of our method across different categories of occlusion. Figure 11 depicts the visual results of the algorithm in different occlusion situations. We demonstrate results on four occlusion types namely, self-occlusion, vehicle occluding car, other objects occluding car, and truncation where the car is partially visible. The first column depicts the output from the MaskRCNN keypoints. The color is coded blue because the output from heatmaps does not give statistics about the occlusion categories of the keypoints. The other column show ablation results on our approach. The results demonstrate

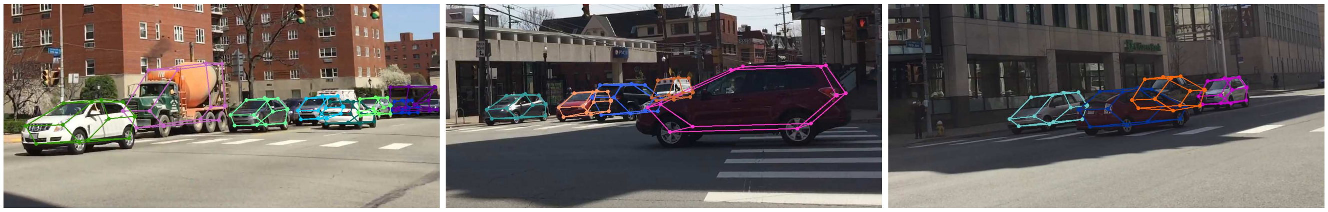

Figure 7: Example results of occlusion-net on sample images of the CarFusion dataset. We accurately localize occluded keypoints under a variety of severe occlusions. See supplementary for additional results. Different colors depict different vehicles in the scene.

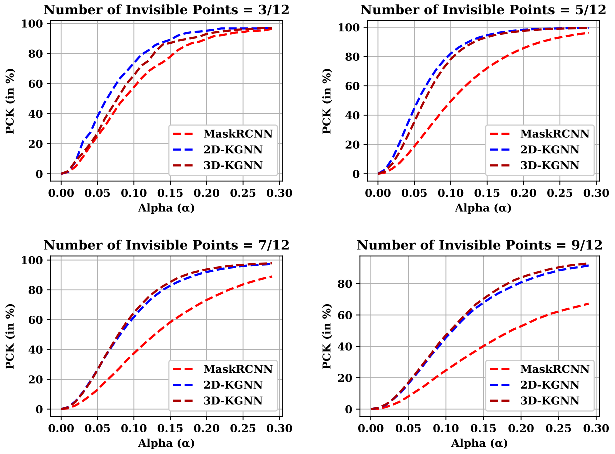

Figure 8: Accuracy vs Alpha on the CarFusion dataset. Focusing on Alpha=0.1 across the plots, graph based methods show an improvement of 6% for cases where only 3 (out of 12) points are occluded and nearly 10% or more improvement for more severe occlusion, justifying the usage of graph networks for occlusion modeling.

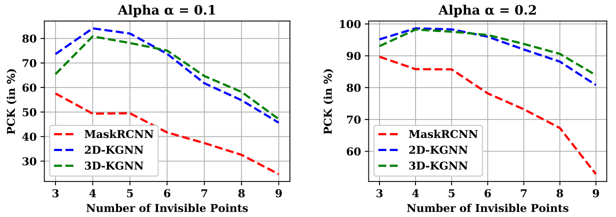

Figure 9: Accuracy analysis with varying occlusion configurations. Notice for occlusions with 4 (out of 12) visible points, our approach is nearly 25% higher compared to the baseline for occluded points.

that predicting occluded keypoints as a heatmap generate large errors in localization while learning a graph based latent space improves the location of the occluded keypoints with respect to the visible points. Specifically, in high occlusion scenarios, graph-based methods show large improvement visually compared to MaskRCNN. We further show the results of our method on multiple cars simultaneously in Figure 7. Our method performs accurate occluded keypoint localization on very challenging occluded cars.

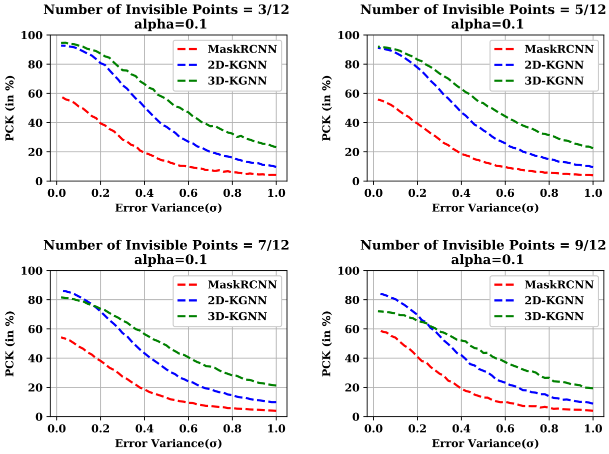

Figure 10: The plots depict the change in accuracy for the methods when Gaussian noise is added to the input keypoints. As expected, 3D-KGNN (green) performs much better in the presence of strong noise.

5. Conclusion

We presented a novel graph based architecture to predict the 2D and 3D locations of occluded keypoints. Since supervision for 2D occluded keypoints is challenging, we computed the error using labeled visible keypoints from different views. We proposed a self-supervised network to lift the 3D structure of the keypoints from the 2D keypoints. We demonstrated our approach on synthetic CAD data as well as a large image set capturing vehicles at many busy city intersections and improve localization accuracy (about 10%) with respect to the baseline detection algorithm.

Acknowledgements

This work was funded in parts by Heinz Endowments, US DOT RITA (University Transportation Center and Mobility 21 Center), NSF #CNS-1446601 and DARPA REVEAL Phase 2 contract HR0011-16-C-0025.

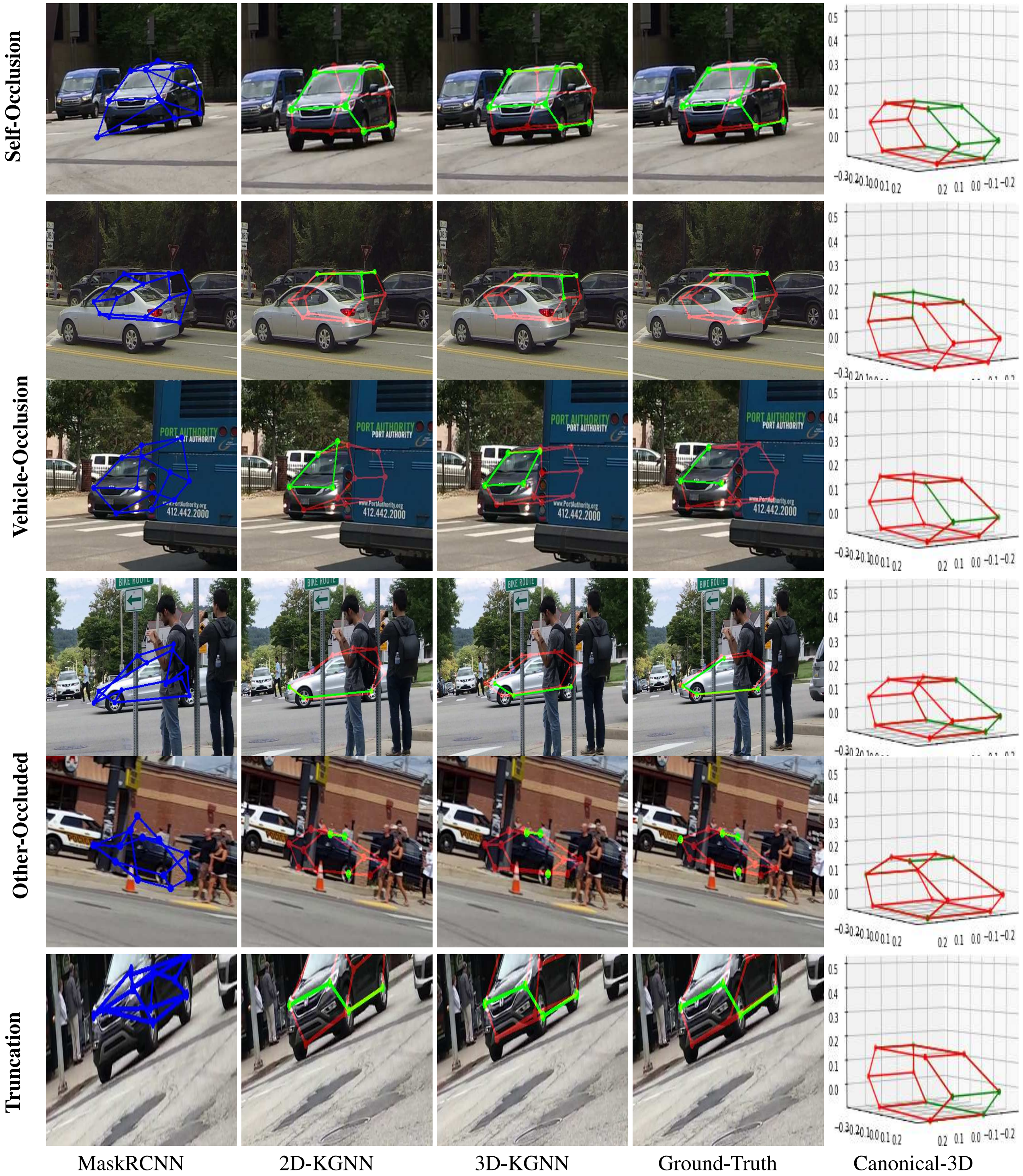

Figure 11: Qualitative evaluation of the 2D/3D keypoint localization for different occlusion categories of cars from the CarFusion dataset. The initial detector was trained using the MaskRCNN on the visible 2D keypoints. We use our self-supervised 2D-KGNN and 3D-GNN to localize keypoints from a single view. 2D reprojections of the 3D keypoints are shown in third column. The second and third columns show clear improvement in the localization of the occluded keypoints with respect to the baseline MaskRCNN. The canonical 3D views computed using 3D-KGNN are shown in the last column. The ground truth is obtained by applying trifocal tensor on the human labeled visible points to estimate the invisible points. Green represents visible edges and red represents occluded edges.

References

[1] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *CoRR*, abs/1312.6203, 2013.

[2] Florian Chabot, Mohamed Chaouch, Jaonary Rabarisoa, Céline Teulière, and Thierry Chateau. Deep manta: A coarse-to-fine many-task network for joint 2d and 3d vehicle analysis from monocular image. *arXiv preprint arXiv:1703.07570*, 2017.

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

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

[5] Wongun Choi. Near-online multi-target tracking with aggregated local flow descriptor. In *ICCV*, 2015.

[6] Timothy F Cootes, Christopher J Taylor, David H Cooper, and Jim Graham. Active shape models-their training and application. *CVIU*, 1995.

[7] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *CoRR*, abs/1606.09375, 2016.

[8] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alan Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2224–2232. Curran Associates, Inc., 2015.

[9] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *TPAMI*, 2010.

[10] Pedro F Felzenszwalb and Daniel P Huttenlocher. Pictorial structures for object recognition. *International journal of computer vision*, 61(1):55–79, 2005.

[11] Martin A Fischler and Robert A Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on computers*, 100(1):67–92, 1973.

[12] Rik Fransens, Christoph Strecha, and Luc Van Gool. A mean field em-algorithm for coherent occlusion handling in map-estimation prob. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 300–307. IEEE, 2006.

[13] Tianshi Gao, Benjamin Packer, and Daphne Koller. A segmentation-aware object detection model with occlusion handling. In *CVPR 2011*, pages 1361–1368. IEEE, 2011.

[14] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[15] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[16] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017.

[17] Mikael Henaff, Joan Bruna, and Yann LeCun. Deep convolutional networks on graph-structured data. *CoRR*, abs/1506.05163, 2015.

[18] Angjoo Kanazawa, David W Jacobs, and Manmohan Chandraker. Warpnet: Weakly supervised matching for single-view reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3253–3261, 2016.

[19] Angjoo Kanazawa, Shubham Tulsiani, Alexei A. Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. *CoRR*, abs/1803.07549, 2018.

[20] Abhishek Kar, Shubham Tulsiani, Joao Carreira, and Jitendra Malik. Category-specific object reconstruction from a single image. In *CVPR*, 2015.

[21] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[22] Abhijit Kundu, Yin Li, and James M. Rehg. 3d-rcnn: Instance-level 3d object reconstruction via render-and-compare. In *CVPR*, 2018.

[23] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o (n) solution to the pnp problem. *International journal of computer vision*, 81(2):155, 2009.

[24] Bo Li, Tianlei Zhang, and Tian Xia. Vehicle detection from 3d lidar using fully convolutional network. In *Robotics: Science and Systems*, 2016.

[25] Chi Li, M. Zeeshan Zia, Quoc-Huy Tran, Xiang Yu, Gregory D. Hager, and Manmohan Chandraker. Deep supervision with intermediate concepts. *CoRR*, abs/1801.03399, 2018.

[26] Chi Li, M Zeeshan Zia, Quoc-Huy Tran, Xiang Yu, Gregory D Hager, and Manmohan Chandraker. Deep supervision with shape concepts for occlusion-aware 3d object parsing. *arXiv preprint arXiv:1612.02699*, 2016.

[27] Yen-Liang Lin, Vlad I Morariu, Winston Hsu, and Larry S Davis. Jointly optimizing 3d model fitting and fine-grained classification. In *European Conference on Computer Vision*, pages 466–480. Springer, 2014.

[28] Or Litany, Tal Remez, Emanuele Rodola, Alex Bronstein, and Michael Bronstein. Deep functional maps: Structured prediction for dense shape correspondence. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 5660–5668. IEEE, 2017.

[29] Haggai Maron, Meirav Galun, Noam Aigerman, Miri Trope, Nadav Dym, Ersin Yumer, Vladimir G Kim, and Yaron Lipman. Convolutional neural networks on surfaces via seamless toric covers. *ACM Transactions on Graphics (TOG)*, 36(4):71, 2017.

[30] David Marr and Herbert Keith Nishihara. Representation and recognition of the spatial organization of three-dimensional shapes. *Proc. R. Soc. Lond. B*, 200(1140):269–294, 1978.

[31] Pol Moreno, Christopher KI Williams, Charlie Nash, and Pushmeet Kohli. Overcoming occlusion with inverse graphics. In *European Conference on Computer Vision*, pages 170–185. Springer, 2016.

[32] Minh Vo N Dinesh Reddy and Srinivasa G. Narasimhan. Carfusion: Combining point tracking and part detection for

dynamic 3d reconstruction of vehicle. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2018*. IEEE, June 2018.

[33] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision*, pages 483–499. Springer, 2016.

[34] Jimmy Ren, Xiaohao Chen, Jianbo Liu, Wenxiu Sun, Jiahao Pang, Qiong Yan, Yu-Wing Tai, and Li Xu. Accurate single stage detector using recurrent rolling convolution. *arXiv preprint arXiv:1704.05776*, 2017.

[35] Samuel Schulter, Menghua Zhai, Nathan Jacobs, and Manmohan Chandraker. Learning to look around objects for top-view representations of outdoor scenes. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[36] Leonid Sigal, Michael Isard, Horst Haussecker, and Michael J Black. Loose-limbed people: Estimating 3d human pose and motion using non-parametric belief propagation. *International journal of computer vision*, 98(1):15–48, 2012.

[37] Tomas Simon, Hanbyul Joo, Iain Matthews, and Yaser Sheikh. Hand keypoint detection in single images using multiview bootstrapping. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1145–1153, 2017.

[38] Shubham Tulsiani, Tinghui Zhou, Alexei A. Efros, and Jitendra Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. *CoRR*, abs/1704.06254, 2017.

[39] Shubham Tulsiani, Tinghui Zhou, Alexei A. Efros, and Jitendra Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *Computer Vision and Pattern Regognition (CVPR)*, 2017.

[40] Andrea Vedaldi and Andrew Zisserman. Structured output regression for detection with partial truncation. In *Advances in neural information processing systems*, pages 1928–1936, 2009.

[41] Shaofei Wang and Charless C Fowlkes. Learning optimal parameters for multi-target tracking with contextual interactions. *International Journal of Computer Vision*, 122(3):484–501, 2017.

[42] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4724–4732, 2016.

[43] Jiajun Wu, Tianfan Xue, Joseph J Lim, Yuandong Tian, Joshua B Tenenbaum, Antonio Torralba, and William T Freeman. Single image 3d interpreter network. In *European Conference on Computer Vision*, pages 365–382. Springer, 2016.

[44] Yu Xiang, Alexandre Alahi, and Silvio Savarese. Learning to track: Online multi-object tracking by decision making. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4705–4713, 2015.

[45] 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.

[46] Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking. *arXiv preprint arXiv:1804.06208*, 2018.

[47] Yi Yang and Deva Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1385–1392. IEEE, 2011.

[48] Li Yi, Hao Su, Xingwen Guo, and Leonidas J Guibas. Syncspeccnn: Synchronized spectral cnn for 3d shape segmentation. In *CVPR*, pages 6584–6592, 2017.

[49] Li Zhang, Yuan Li, and Ramakant Nevatia. Global data association for multi-object tracking using network flows. In *CVPR*, 2008.

[50] Xiaowei Zhou, Spyridon Leonardos, Xiaoyan Hu, and Kostas Daniilidis. 3d shape estimation from 2d landmarks: A convex relaxation approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4447–4455, 2015.

[51] M Zeeshan Zia, Michael Stark, Bernt Schiele, and Konrad Schindler. Detailed 3d representations for object recognition and modeling. *TPAMI*, 2013.

[52] M Zeeshan Zia, Michael Stark, and Konrad Schindler. Towards scene understanding with detailed 3d object representations. *IJCV*, 2015.