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

Anonymous CVPR submission

Paper ID 3898

Abstract

We present Occlusion-Net, 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 (like MaskRCNN) that is trained only on visible key point annotations. This is the only supervision used in this work. A graph network encoder then explicitly classifies invisible edges and a 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 similar 3D graph network that estimates the 3D wireframe (shape basis parameters) and camera projection using the self-supervised re-projection loss. At test time, our approach successfully localizes keypoints in a single view under a wide range and severity of occlusion scenarios. We demonstrate and evaluate 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 obtained from geometric trifocal-tensor loss.

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 [8, 11, 30, 13], tracking[41, 4, 37, 34], reconstruction [16, 15] and recognition, even today's advanced deep learning based ones.

The computer vision community has collectively attempted numerous approaches to deal with occlusions [21] 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 the CarFusion dataset [27].Different color depicts different object in the scene

On the other hand, occlusions are explicitly treated as missing parts in model fitting methods [42]. 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 almost the entirety of a vehicle blocked, can result in poor fitting. 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 [13]) that has been trained on a particular category of object with hu-

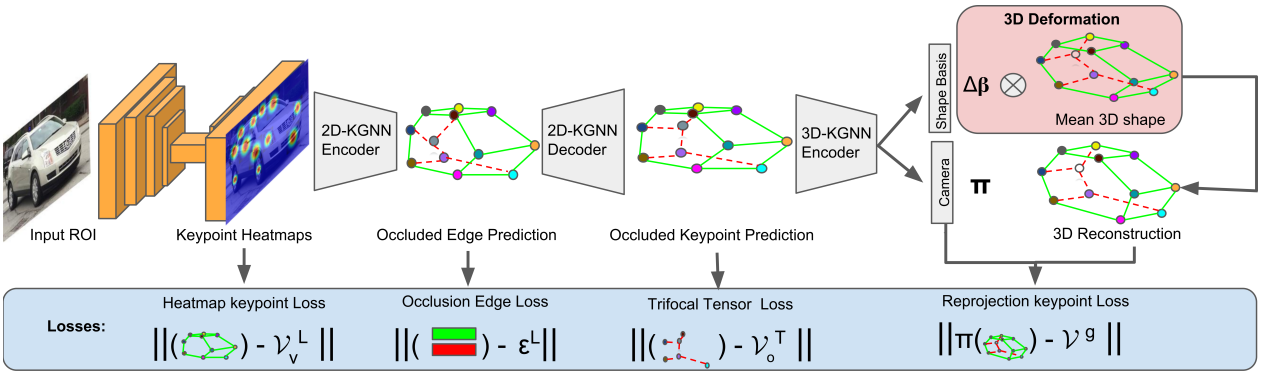

Figure 2: Occlusion-net: We illustrate the overall approach to training a network to improve localization of occluded keypoints. The input is a ROI region from any detector, which is passed through multiple convolutional layers to predict the heatmaps with a confidence score. These confidences are passed 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 re-projection loss with respect to the 2D decoder output.

man supervision of *only visible keypoints* and their types (front, back, left, right, etc). Implicitly, then, the key points that are not labeled are assumed to be invisible. This is the 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. Edge classification is trained using the implicit non-labeled supervision of occluded points.

The decoder graph network to localize invisible keypoints is trained using multiple wide-baseline views of objects. The idea here 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 keypoints from the initial detector. Compared to other approaches that use multiple views [33], our approach explicitly predicts occluded keypoints.

The predicted 2D keypoints (occluded and visible) are then used in a graph neural network to estimate 3D wireframe of the object and the camera projection matrix. Similar to previous work, we will estimate the parameters of a shape basis computed a priori of the object of interest. The graph network can also enforce explicit symmetry constraints. The training is performed in a self-supervised way by minimizing the re-projection loss i.e. error between the re-projection and the predicted 2D keypoint locations. We call the entire pipeline Occlusion-net that is trained 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 [27] 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 visible keypoints labeled, 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 severe 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 at test-time (more than 30 fps). While we have demonstrated our approach on vehicles, the framework is general and applies to any type of object.

2. Related Work

Occlusion Detection: While there has been great progress in predicting the visible keypoints by using strong part detectors learned from CNNs [28, 35, 21, 2, 22, 38], most of these methods fail short in precise localization of the occluded keypoints. Such occlusion modeling is showed to be very important by Moreno et al. [26] using synthetic data. To address this problem, many methods

employ active shape models [5] for vehicle detection under occlusion [43, 44, 36]. However, these methods only model self-occlusions and omit other often seen occlusion types such as car to car occlusion or objects to car occlusion. Recently, [32, 27] 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. Our work is closely related to [20], which only incorporate intermediate keypoint supervisions from CAD model during training. Interestingly, they show that training such model on synthetic images only do generalize to real images. We train our model on real images and incorporate multiview constrain to propagate ground truth visible keypoints from multiple views to supervise occluded points. Our model generalizes well to unseen real-world data with severe occlusions.

Graph Neural Networks: Modeling keypoints as a graph problem can be dated back to the first attempt at scene understanding [10, 25]. Multiple works have build on this graph modelling and solve pose using belief propagation [9] [31]. Deep learning approaches have been successful on data with underlying Euclidean structure, graph neural networks are one of the approaches to extend deep learning to non-Euclidean structure. Specifically, [7, 18, 1, 14, 6] have proposed algorithms to generalize deep learning to unstructured data. With the success of these methods on the graph classification task, multiple recent works have extended the methods to address multiple 3D problems like Shape segmentation [40], 3D correspondence [23], CNN on surfaces [24]. We model keypoint prediction as a deformable graph which is learned using multi-view supervision.

3. Method

We illustrate the complete approach of our algorithm in Fig 2. It consists of three main stages. The 2D-Keypoint Graph Neural Network which deforms the graph to accurately predict the occluded keypoints. The 3D-Keypoint Graph Neural Network, predicts the underlying 3D geometry of the keypoints and then we propose the occlusion-net which combines the best of convolutional and graph networks for accurate keypoint estimation. Each of these stages is described in the following sections.

3.1. 2D-Keypoint Graph Neural Network

Our 2D-Keypoint Graph Neural Network(2D-KGNN) consists of three components: initial keypoint heatmap prediction, an encoder to model the occlusion statistics of the graph, and a decoder inferring the 2D shape of the occluded keypoints. The architecture is inspired from [17], which models graphs for interactions. 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. We model the relationships between all nodes as $\mathcal{E}_{ij} = \{\mathcal{V}_i, \mathcal{V}_j\}$, where

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

3.1.1 Occluded Edges Prediction using 2D-KGNN Encoder

At a higher level, the encoder needs to infer about the $\mathcal{E}_{ij}$ from the heatmaps produce from the previous section. 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) encodes the location, while c_i is defined as the confidence and t_i is defined as the type. Since we do not know the underlying graph, we use GNN to predict the latent graph structure.
The encoder is modelled 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:

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

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

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

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

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 that map between the representations represented in the above equations. The edge loss trained 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.

3.1.2 Occluded Keypoint Localization using 2D-KGNN Decoder

In the case of the decoder, we need to predict 2D consistent keypoint location of the occluded keypoints from the

erroneous initial graph and the edges predicted from the encoder. This can mathematically be represented as predicting $P_\theta(\mathcal{V}^g|\mathcal{V},\mathcal{E})$, 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 heatmap. The following message passing steps are computed on the graph network:

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

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

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

Here $\mathcal{E}_{ij,p}$ denotes the p-th element of the vector $\mathcal{E}_{ij}$ and another 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 $\Delta\mathcal{V} = \mathcal{V}^g - \mathcal{V}$. 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: Estimating the occluded keypoints "in the wild" from human annotations is a hard problem, so we use multi-view consistency to predict the location of the invisible points. The assumption is that with multiple views of the object, the occluded keypoints are visible in a different view to hypothesis the keypoint in the current view. The loss for each occluded keypoint is computed as:

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

where $\mathcal{V}_j^g$ is the prediction from the decoder for the occluded keypoint j in the current view, $\mathcal{V}'_j$ and $\mathcal{V}''_j$ are the annotated keypoint j in two different views. While the T represents a $3 \times 3 \times 3$ trifocal tensor. Which is computed using the camera projection matrices in the object reference frame following the method in [12]. Since we have multiple views of the object, We compute the relative poses in the object reference frame from the world reference frame by solving for the virtual camera with respect to the corresponding visible keypoints.

3.2. 3D-Keypoint Graph Neural Network

Given the graph from the 2D-KGNN decoder, the encoder predicts a 3D wireframe (W) and the camera projection matrix π to capture the 3D structure of the object. We follow the Encoder framework proposed in section 3.1.1, but instead of predicting the edge class, We model it as $q(\Delta\beta, \pi|\mathcal{V}) = softmax(f_{enc}(\mathcal{V}))$. Where $\Delta\beta$ are the deformation parameters from the mean shape.

Shape Reconstruction: We model the shape as a set of 3D keypoints corresponding to the predicted 2D keypoints. We compute the mean shape M and n principal component directions p_j and corresponding standard deviations σ_j, where $1 \leq j \leq m$ by training PCA basis on the 3D repository of the object. Now any set of deformable 3D keypoints can be represented as a linear combination of the r principal components with geometry parameters s, where s_k is the weight of the k^{th} principal component: $W = M + \sum_{k=1}^{r} s_k \sigma_k p_k$. Our encoder directly predicts the geometric parameters, so the 3D keypoints can be computed using $W = M + \Delta\beta p_k$, where $\Delta\beta$ is the basis vector for the given graph.

Camera Projection Matrix: We model the camera matrix as a perspective projection matrix and predict the rotation $q \in \mathbb{R}^4$, translation $t \in \mathbb{R}^3$ and the focal length $f \in \mathbb{R}$. We denote $\pi(W)$ as the projection of a set of 3D keypoints W onto the image coordinates via a perspective project defined by $\pi \equiv (f, t, q)$. Another important aspect is to solve for the normalization of the image to a square matrix from the original dimensions. To minimize the loss with respect to original coordinates, projected 2D points need to be scaled back by $s = w/h$, where w and h denote the width and height of the input image. We refer you to [19] further details on computing the intrinsic camera matrix for scaled input images.

Reprojection Keypoint Loss: The loss to the network is the difference between the projected keypoints and the keypoints computed from the 2D-KGNN. This network is completely self-supervised. i.e. given the 2D locations, it automatically lifts the 3D structure of the object. The loss is :

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

This method of lifting to 3D incorporates symmetry of the object and is category specific. The re-projection loss simultaneously optimizes for the shape and the pose of the object in 3D.

3.3. Occlusion-Net:

Our Occlusion-Net is trained to minized the following loss:

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

where $L_{Keypoints}$ is the standard cross-entropy loss over an t^2 -way softmax output between the predicted keypoints and the ground truth annotation. We optimize the network using Adam optimizer [29].

Figure 3: Sample Ground truth images from the CarFusion and the Car-render CAD dataset. Green solid lines depict the visible edges and nodes, while the red transparent lines represent the occluded points.

4. Evaluation

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 the qualitative comparison with the state of art Mask-RCNN [13] detector in section 4.3. 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. For a fair comparison, we retrain this baseline model on our dataset.

4.1. Dataset

Car-render Self-occlusion dataset: We use the 472 cars sampled from shapenet [3] and 3D annotated by [21]. We pick 12 keypoints from the annotated 36 keypoints and render them from different viewpoints with the occlusion statistics. The viewpoints are randomly selected on a level 5 icosahedron with varying focal length and distance from the object. A sample rendering of the car CAD model is shown on the right side of Figure 3. We use 300 synthetic CAD models for training, 100 for testing and 72 for validation. We project the 3D keypoint annotations of the CAD model with visibility flags. The visibility is computed by ray-tracing the projected point and threshold the difference between the computed and the ground truth 3D keypoint. This dataset only contains self-occlusion scenarios but covers different viewpoints.

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. This dataset consists of 2.5 million images out of which 53000 images were sampled at uniform intervals from each video sequence. Approximately 100,000 cars detected in these images were annotated with

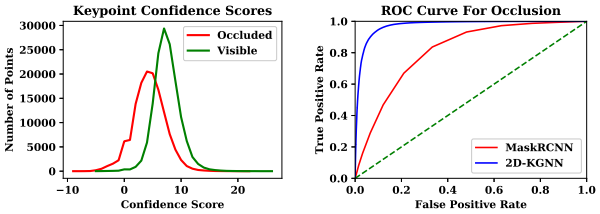

Figure 4: We analyze the need for a 2D-KGNN encoder. The left image shows the confidence score of the heatmaps from the baseline method (the distribution is colored based on Ground Truth visibility). The right image shows the ROC curve of the predictions from graph encoder and baseline. At 0.1 False positive rate baseline returns 0.5 true positive rates compared to 0.8 of the 2D-KGNN.

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. A sample ground truth annotated car from the dataset is displayed on the left side of the Figure 3. The dataset contains different occlusion categories as there was no constraint to capture the data and was captured at an intersection. We will release the data publicly.

4.2. Quantitative Evaluation

We compare our approach with other state-of-the-art keypoint detection networks. We use the PCK metric [39] to analyze the localization of both the 2D occluded keypoint and the 3D occluded keypoint. The PCK is defined as the percentage of the correct keypoint predictions given the object location. According to the PCK metric, a keypoint is 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 analyze the occlusion prediction with respect to confidence scores computed by using MaskRCNN. The left image in Fig 4 shows the occluded points and their respective confidence scores. As can be seen from the plot it is hard to distinguish occluded points from visible points through simple thresholding. On the contrary, by modeling a simple graph network to exploit relative locations of the keypoints, we see a significant boost in the accuracy of occlusion modeling as seen from the right image in figure 4. We observe an AUC of 0.83 with MaskR-

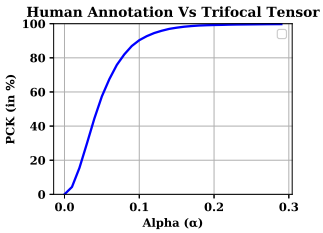

Figure 5: Accuracy of human annotations with respect to geometrically obtained keypoints. We observe that most of the keypoints are labeled within $\alpha = 0.1$ PCK error.

CNN, while 2D-KGNN gives an AUC of 0.95 for the ROC curve on occlusion modeling.

Human Annotation vs Geometric Prediction: The CarFusion dataset has annotated keypoints for occluded points as well as the visible points in multiple views. We evaluate the accuracy of hand-labeled occluded points with respect to the hypothetical ground truth by using trifocal tensor to predict keypoints visible from another view to the occluded view, as shown in figure 5. We observe that at $\alpha = 0.1$, nearly 90% of the hand-labeled keypoint lying within the region of geometrically consistent keypoint. While such PCK value is good for any detection algorithm on occluded points, we will later show that using such hand labeled points for training significantly degrade the detector accurately.

Accuracy Analysis: Figure 6 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 7 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 9, 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 atleast 6% in all the occlusion categories and nearly 12% boost for 5 occluded points. Figure 10 depicts the change in accuracy with increasing number of occluded points on CarFusion dataset. Notice 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.

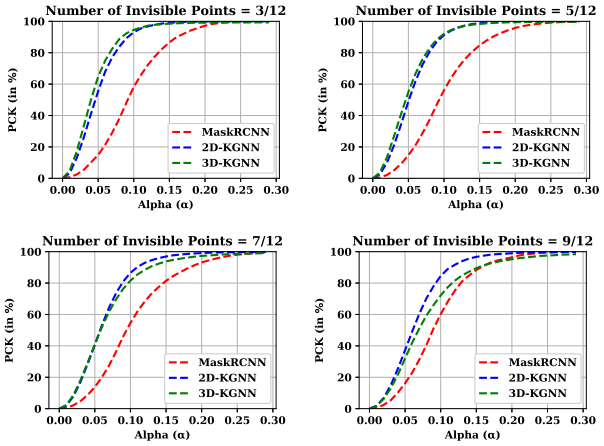

Figure 6: Accuracy with respect to different alpha values of PCK for the Car-render dataset. Graph based methods (2D/3D) outperform the MaskRCNN trained keypoints for all the occlusion types. Specifically at alpha=0.1 we observe a boost of 22% for cases with 3 (out of 12) invisible points while its 10% in case of 9 invisible points (out of 12 keypoints).

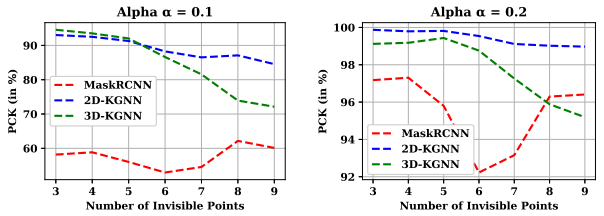

Figure 7: Accuracy plots with varying number of Occluded keypoints on the Car-render dataset. Graph based methods (2D/3D) outperform the baseline (red plot) in the case of $\alpha = 0.1$. For a more conservative alpha, the performances are comparable. The 2D KGNN plots in both the alpha scenarios have a variance of 5% and are robust to occlusion, compared to the 3D KGNN plot (15%) and the baseline MaskRCNN plot (25%).

Robustness Analysis: We analyze the effect of adding error to input locations of the graph to analyze the robustness of the learned model. Figure 11 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.

4.3. Qualitative Evaluation

In this section, we analyze the visual improvements of our method across different categories of occlusion. Fig 12 depicts the visual results of the algorithm in different occlu-

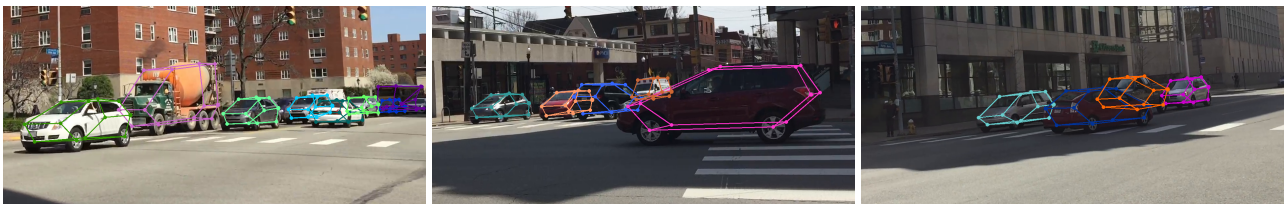

Figure 8: Example results of occlusion-net on sample images of the CarFusion dataset. Observe accurate occluded keypoint localization under a variety of severe occlusions. See supplementary for additional results. Different color depicts different object in the scene.

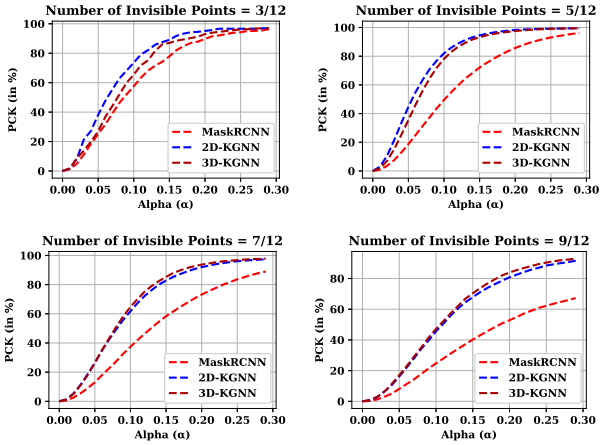

Figure 9: Accuracy vs Alpha on the CarFusion dataset. Focusing on Alpha=0.1 across the plots, graph based methods show an improvement of 6% for cases where only 3(out of 12) points are occluded and nearly 10% or more improvement for increased occlusion scenarios, justifying the usage of graph networks for occlusion modeling.

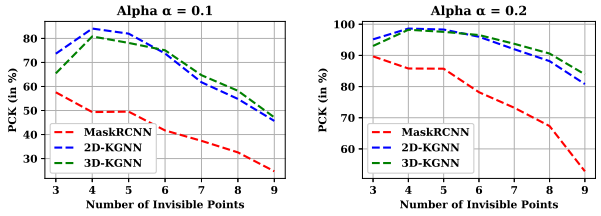

Figure 10: Accuracy analysis with varying occlusion configurations. Notice for occlusions with 4 (out of 12) visible points, our approach is nearly 25% higher compared to the baseline for occluded points.

sion 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

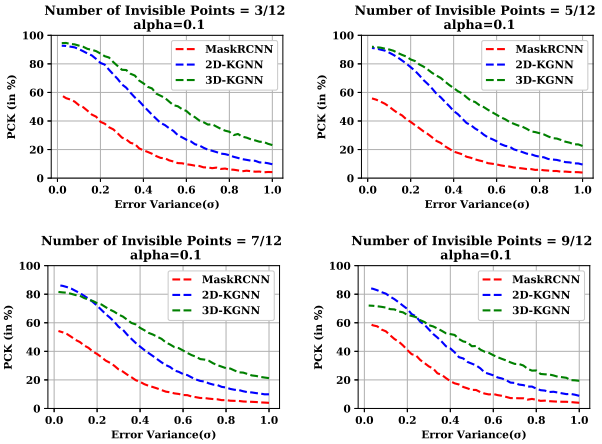

Figure 11: The plots depict the change in accuracy for the methods when Gaussian noise is added to the input keypoints. As expected, 3D-KGNN (green) performs much better in the presence of strong noise.

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 Fig 8. Our method performs accurate occluded keypoint localization on very challenging occluded cars.

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.

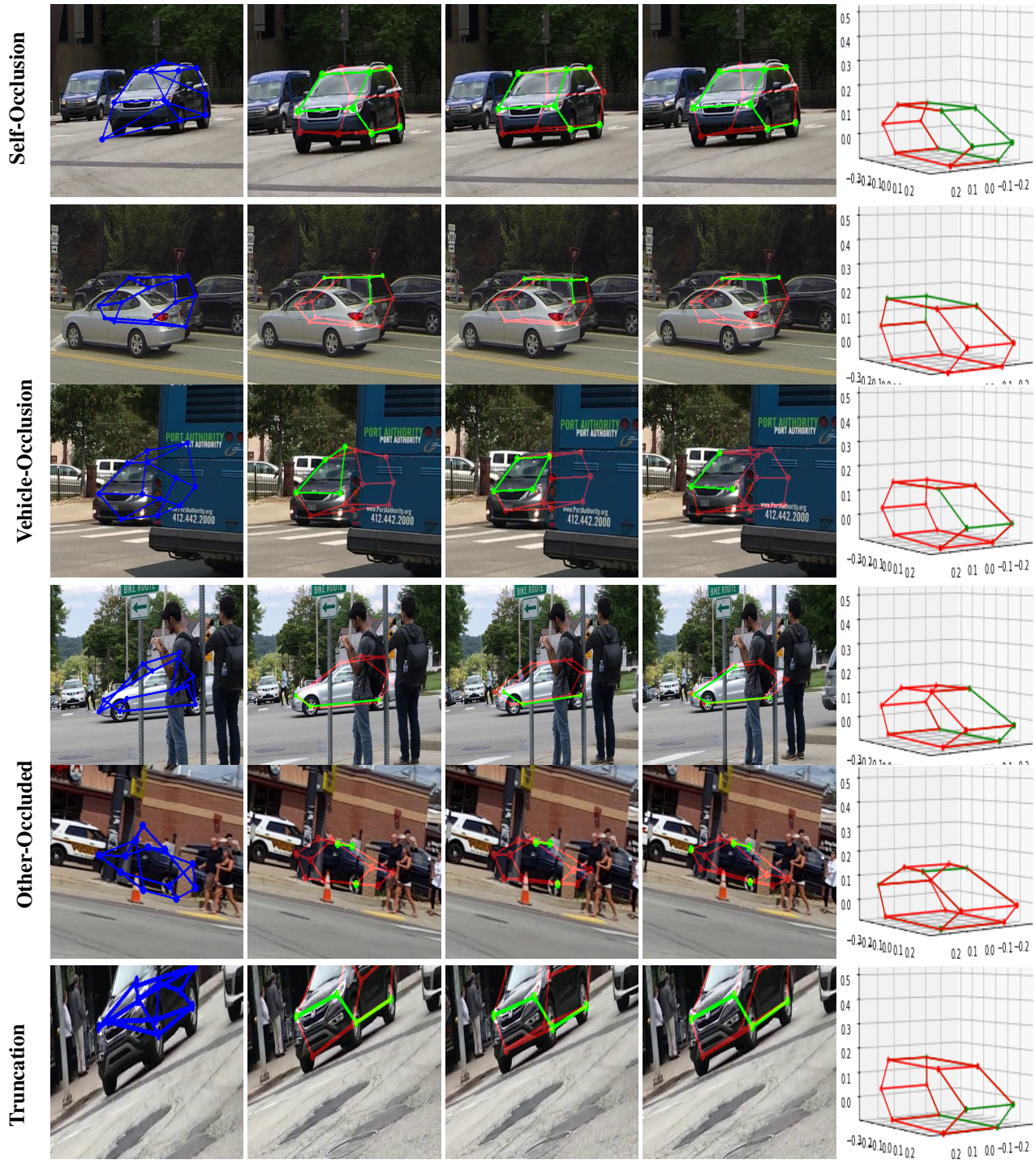

Figure 12: Qualitative evaluation of the 2D/3D keypoint localization for different occlusion categories of cars from the CarFusion dataset. The initial detector was trained using the MaskRCNN on the visible 2D keypoints. We use our self-supervised 2D-KGNN and 3D-GNN to localize keypoints from a single view. 2D re-projections 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] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun. Spectral networks and locally connected networks on graphs. *CoRR*, abs/1312.6203, 2013.

[2] F. Chabot, M. Chaouch, J. Rabarisoa, C. Teulière, and T. 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] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

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

[5] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active shape models-their training and application. *CVIU*, 1995.

[6] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *CoRR*, abs/1606.09375, 2016.

[7] D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, and R. 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.

[8] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *TPAMI*, 2010.

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

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

[11] R. Girshick, J. Donahue, T. Darrell, and J. 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.

[12] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[13] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *ICCV*, 2017.

[14] M. Henaff, J. Bruna, and Y. LeCun. Deep convolutional networks on graph-structured data. *CoRR*, abs/1506.05163, 2015.

[15] A. Kanazawa, S. Tulsiani, A. A. Efros, and J. Malik. Learning category-specific mesh reconstruction from image collections. *CoRR*, abs/1803.07549, 2018.

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

[17] T. Kipf, E. Fetaya, K.-C. Wang, M. Welling, and R. Zemel. Neural relational inference for interacting systems. *arXiv preprint arXiv:1802.04687*, 2018.

[18] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[19] A. Kundu, Y. Li, and J. M. Rehg. 3d-rcnn: Instance-level 3d object reconstruction via render-and-compare. In *CVPR*, 2018.

[20] C. Li, M. Z. Zia, Q. Tran, X. Yu, G. D. Hager, and M. Chandraker. Deep supervision with intermediate concepts. *CoRR*, abs/1801.03399, 2018.

[21] C. Li, M. Z. Zia, Q.-H. Tran, X. Yu, G. D. Hager, and M. Chandraker. Deep supervision with shape concepts for occlusion-aware 3d object parsing. *arXiv preprint arXiv:1612.02699*, 2016.

[22] Y.-L. Lin, V. I. Morariu, W. Hsu, and L. S. Davis. Jointly optimizing 3d model fitting and fine-grained classification. In *European Conference on Computer Vision*, pages 466–480. Springer, 2014.

[23] O. Litany, T. Remez, E. Rodola, A. Bronstein, and M. Bronstein. Deep functional maps: Structured prediction for dense shape correspondence. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 5660–5668. IEEE, 2017.

[24] H. Maron, M. Galun, N. Aigerman, M. Trope, N. Dym, E. Yumer, V. G. Kim, and Y. Lipman. Convolutional neural networks on surfaces via seamless toric covers. *ACM Transactions on Graphics (TOG)*, 36(4):71, 2017.

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

[26] P. Moreno, C. K. Williams, C. Nash, and P. Kohli. Overcoming occlusion with inverse graphics. In *European Conference on Computer Vision*, pages 170–185. Springer, 2016.

[27] M. V. N Dinesh Reddy and S. 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.

[28] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision*, pages 483–499. Springer, 2016.

[29] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[30] J. Ren, X. Chen, J. Liu, W. Sun, J. Pang, Q. Yan, Y.-W. Tai, and L. Xu. Accurate single stage detector using recurrent rolling convolution. *arXiv preprint arXiv:1704.05776*, 2017.

[31] L. Sigal, M. Isard, H. Haussecker, and M. 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.

[32] T. Simon, H. Joo, I. Matthews, and Y. 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.

[33] S. Tulsiani, T. Zhou, A. A. Efros, and J. Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. *CoRR*, abs/1704.06254, 2017.

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

[35] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional pose machines. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4724–4732, 2016.

[36] J. Wu, T. Xue, J. J. Lim, Y. Tian, J. B. Tenenbaum, A. Torralba, and W. T. Freeman. Single image 3d interpreter network. In *European Conference on Computer Vision*, pages 365–382. Springer, 2016.

[37] Y. Xiang, A. Alahi, and S. 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.

[38] B. Xiao, H. Wu, and Y. Wei. Simple baselines for human pose estimation and tracking. *arXiv preprint arXiv:1804.06208*, 2018.

[39] Y. Yang and D. 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.

[40] L. Yi, H. Su, X. Guo, and L. J. Guibas. Syncspeccnn: Synchronized spectral cnn for 3d shape segmentation. In *CVPR*, pages 6584–6592, 2017.

[41] L. Zhang, Y. Li, and R. Nevatia. Global data association for multi-object tracking using network flows. In *CVPR*, 2008.

[42] X. Zhou, S. Leonardos, X. Hu, and K. 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.

[43] M. Z. Zia, M. Stark, B. Schiele, and K. Schindler. Detailed 3d representations for object recognition and modeling. *TPAMI*, 2013.

[44] M. Z. Zia, M. Stark, and K. Schindler. Towards scene understanding with detailed 3d object representations. *IJCV*, 2015.