Deep MANTA: A Coarse-to-fine Many-Task Network for joint 2D and 3D vehicle analysis from monocular image

Florian Chabot[1] , Mohamed Chaouch[1] , Jaonary Rabarisoa[1] , Céline Teulière[2] , Thierry Chateau[2]
[1] CEA-LIST Vision and Content Engineering Laboratory, [2] Pascal Institute, Blaise Pascal University
[1]{florian.chabot, mohamed.chaouch, jaonary.rabarisoa}@cea.fr
[2]{celine.teuliere, thierry.chateau}@univ-bpclermont.fr

Abstract

In this paper, we present a novel approach, called Deep MANTA (Deep Many-Tasks), for many-task vehicle analysis from a given image. A robust convolutional network is introduced for simultaneous vehicle detection, part localization, visibility characterization and 3D dimension estimation. Its architecture is based on a new coarse-to-fine object proposal that boosts the vehicle detection. Moreover, the Deep MANTA network is able to localize vehicle parts even if these parts are not visible. In the inference, the network's outputs are used by a real time robust pose estimation algorithm for fine orientation estimation and 3D vehicle localization. We show in experiments that our method outperforms monocular state-of-the-art approaches on vehicle detection, orientation and 3D location tasks on the very challenging KITTI benchmark.

1. Introduction

Over the last years, traffic scene analysis has been improved thanks to deep learning approaches which paves the way to multiple applications, especially, autonomous driving. Impressive recent work in 2D object detection [33, 15, 14] already provides important information related to scenes content but does not yet allow to describe objects in the 3D real world scene. In this paper, we are interested in both 2D and 3D vehicle analysis from monocular images in the context of self-driving cars. This is a relevant research field because currently most cars are equipped with a single camera. For an autonomously driving vehicle, it is essential to understand the traffic and predict critical situations based on the information extracted from the image of the scene. For the recovery of speed and direction of the surrounding cars, 3D vehicle localization and orientation jointly used with temporal description are necessary. Additionally, for proper traffic understanding it is important to describe surrounding vehicles in a fine way. For example, correct localization of high lights is required to interpret vehicle direction indicators, for which knowledge of the exact location of vehicle parts is needed. Finally, for interpretation of the overall scene the characterization of the visibility of vehicle parts needs also to be obtained. Thus it will be known if a vehicle is hidden by other vehicles or environment obstacles. Here we propose an approach that, given a single image, provides accurate vehicle detections, vehicle part localization, vehicle part visibility, fine orientation, 3D localization and 3D template (3D dimension). Figure 1 illustrates the outputs of our approach.

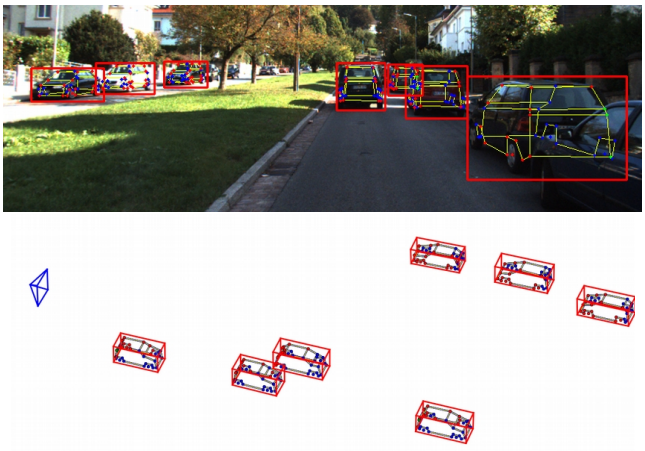

Figure 1. System outputs. *Top*: 2D vehicle bounding boxes, vehicle part localization and part visibility. In this example, red dots correspond to visible parts, green dots to occluded parts and blue dots to self-occluded parts. *Bottom*: 3D vehicle bounding box localization and 3D vehicle part localization. The camera is represented in blue.

Our first contribution is to encode 3D vehicle information using characteristic points of vehicles. The underlying idea is that 3D vehicle information can be recovered using monocular images because vehicles are rigid objects with well known geometry. Our approach localizes vehicle parts even if these parts are hidden due to occlusion, truncation or self-occlusion in the image. These parts are

found using regression instead of using a part detector. In this way, the approach predicts the position of hidden parts which are essential for robust 3D information recovering. We use a 3D vehicle dataset composed of 3D meshes with real dimensions. Several vertices are annotated for each 3D model. These 3D points correspond to vehicle parts (such as wheels, headlights, etc) and define a 3D shape for each 3D model. The main idea of the approach is to recover the projection of these 3D points (2D shape) in the input image for each detected vehicle. Then, the best corresponding 3D model for each detection box is chosen. 2D/3D matching is performed between 2D shapes and selected 3D shapes to recover vehicle orientation and 3D location.

The second contribution is the introduction of the Deep Coarse-to-fine Many-Task Convolutional Neural Network called Deep MANTA. This network outputs accurate 2D vehicle bounding boxes, 2D shapes, part visibility and 3D vehicle templates. Its architecture contains several originalities. Firstly, inspired by the Region proposal network [33], the MANTA model is able to propose coarse 2D bounding boxes which are then iteratively refined, by multi-pass forward, to provide accurate scored 2D detections. Secondly, this network is based on the many-task concept. That means that the same feature vector can be used to predict many tasks. We optimize in the same time six tasks: region proposal, detection, 2D box regression, part localization, part visibility and 3D template prediction.

The last contribution is related to the training dataset. Deep neural networks require many samples and labels to be efficiently learned. Furthermore, it is very fastidious and almost impossible to annotate manually vehicle parts which are not visible. For this purpose, we propose a semi-automatic annotation process using 3D models to generate labels on real images for the Deep MANTA training. Labels from 3D models (geometry information, visibility, etc) are automatically projected onto real images providing a large training dataset without labour-intensive annotation work.

In the next section, related work is reviewed. The section 3 explains the proposed model. Finally, we show that our approach outperforms monocular state-of-the-art methods related to vehicle detection, orientation and 3D localization on the very challenging KITTI dataset [12].

2. Related work

Object analysis is a well studied topic and we divide it into two main categories: 2D object detection/coarse pose estimation and 3D object detection/fine pose estimation.

2D Object detection and coarse pose estimation. There are two ways to perform 2D object detection. The first one is the standard sliding window scheme used in many detection systems as [10, 34]. The second one is the 2D object proposal based methods [15, 14, 38, 5, 1]. The goal of object proposal methods is to propose several boxes with high objectness confidence score. These proposals are then given to a detector which is able to classify objects and background. The main advantage of object proposal methods is the processing time because that considerably reduces the search space. In parallel, Deep Convolutional Neural Networks (CNN) have proven their effectiveness in many computer vision fields such as object classification [36, 16, 19, 37], object detection [15, 14, 33] and scene segmentation [26, 9]. Thus, the success of object proposal methods as well as CNN, leads people to directly learn *Region Proposal Networks* (RPN) sharing weights with the down-stream detection network [33, 43, 40, 18]. RPN provides strong objectness confidence regions of interest computed on deep feature maps. Experiments show that this kind of method increases detection accuracy. The proposed approach uses the RPN framework but uses several steps of 2D bounding box refinement to significantly increase object detection performance. 2D object detection is often associated with pose estimation and many methods address the two issues. They generally divide the viewing sphere in several bins to learn multi-class models where each bin corresponds to a class [27, 41, 47, 22, 29]. These approaches allow to get coarse information on objects and do not provide continuous viewpoint estimation.

3D Object detection and fine pose estimation. To go further than 2D reasoning, several approaches are designed to detect vehicles in 3D space and are able to give a detailed 3D object representation. A part of them consists in fitting 3D models [23, 32, 2, 17], active shape model [44, 46, 45, 24, 42] or predicting 3D voxel patterns [39] to recover the exact 3D pose and detailed object representation. These methods generally use an initialization step providing the 2D bounding box and the coarse viewpoint information. More recently, people have proposed to use 3D object proposals generated while using monocular images [7] or disparity maps [8]. In these approaches, 3D object proposals are projected in 2D bounding boxes and given to a CNN based detector which jointly predicts the class of the object proposal and the object fine orientation (using angle regression). In the proposed approach, vehicle fine orientation estimation is found using a robust 2D/3D vehicle part matching: the 2D/3D pose matrix is computed using all vehicle parts (visible or hidden) in contrast to other methods such as [44, 46, 45, 24] which focus on visible parts. That clearly increases the precision of orientation estimation.

3. Deep MANTA approach

In this section, we describe the proposed approach for 2D/3D vehicle analysis from monocular images. Our system has two main steps. First, the input image is passed through the Deep MANTA network that outputs 2D scored bounding boxes, associated vehicle geometry (vehicle part

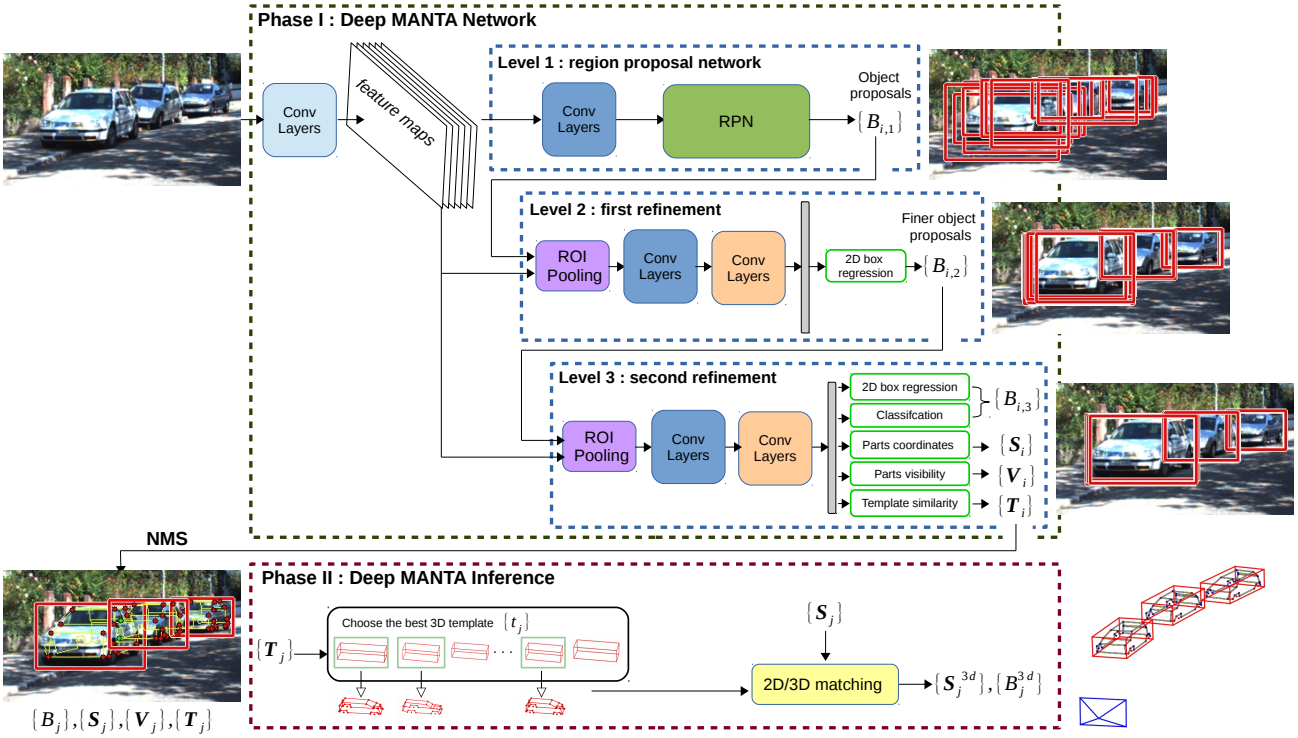

Figure 2. Overview of the Deep MANTA approach. The entire input image is forwarded inside the Deep MANTA network. Conv layers with the same color share the same weights. Moreover, these three convolutional blocks correspond to the split of existing CNN architecture. The network provides object proposals $\{B_{i,1}\}$ which are iteratively refined ($\{B_{i,2}\}$ and then the final detection set $\{B_{i,3}\}$). 2D part coordinates $\{\mathbf{S}_i\}$, part visibility $\{\mathbf{V}_i\}$ and template similarity $\{\mathbf{T}_i\}$ are associated to the final set of detected vehicle $\{B_{i,3}\}$. A non-maximum suppression (NMS) is then performed. It removes redundant detections and provides the new set $\{B_j, \mathbf{S}_j, \mathbf{V}_j, \mathbf{T}_j\}$. Using these outputs, the inference step allows to choose the best corresponding 3D template using template similarity $\mathbf{T}_j$ and then performs 2D/3D pose computation using the associated 3D shape.

coordinates, 3D template similarity) and part visibility properties. The Deep MANTA network architecture is detailed in the section 3.3. The second step is the inference which uses Deep MANTA outputs and a 3D vehicle dataset to recover 3D orientations and locations. This step is detailed in the section 3.4. In this method, we use a dataset of 3D shapes and one of 3D templates. These two datasets encode the variability of vehicles in terms of dimension, type, and shape. These datasets are presented in the section 3.1. In the section 3.2, we define the adopted 2D/3D vehicle model for a given vehicle in a monocular image.

3.1. 3D shape and template datasets

We use a dataset of M 3D models corresponding to several types of vehicles (Sedan, SUV, etc). For each 3D model m, we annotate N vertices (called 3D parts). These parts correspond to relevant vehicle regions. For one 3D model m, we denote its 3D shape aligned in canonical view as $\mathbf{S}_m^{3d} = (p_1, p_2, .., p_N)$ with $p_k = (x_k, y_k, z_k)$ corresponding to the 3D coordinate of the k^{th} part. The 3D template (*i.e* 3D dimension) associated to the 3D model m is defined as $t_m^{3D} = (w_m, h_m, l_m)$ where w_m, h_m, l_m are the width, the height and the length of the 3D model respectively. Figure 3 shows some examples from the 3D shape dataset $\{\bar{\mathbf{S}}_m^{3d}\}_{m\in\{1,..,M\}}$ and the 3D template dataset $\{\bar{t}_m^{3d}\}_{m\in\{1,..,M\}}$.

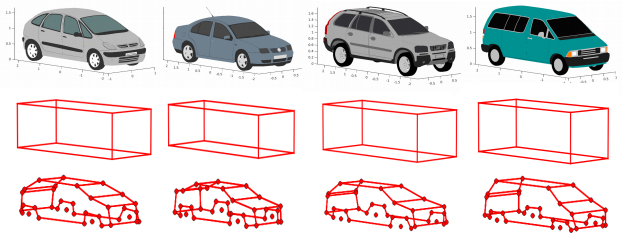

Figure 3. Some examples from the 3D template and 3D shape dataset. Each 3D model m (first line) is associated to a 3D template $\bar{t}_m^{3d}$ (second line) and a 3D shape $\bar{\mathbf{S}}_m^{3d}$ (third line). The 3D shape corresponds to manually annotated vertices.

3.2. 2D/3D vehicle model

We represent each vehicle in a monocular image with a 2D/3D model. It is formally defined by the following attributes:

$$(B, B^{3d}, \mathbf{S}, \mathbf{S}^{3d}, \mathbf{V})$$

$B = (c_x, c_y, w, h)$ is the 2D vehicle bounding box in the image where (c_x, c_y) is the center and (w, h) represents the width and the height respectively. $B^{3d} = (c_x, c_y, c_z, \theta, t)$ is the 3D bounding box characterized by its 3D center (c_x, c_y, c_z), its orientation θ and its 3D template $t = (w, h, l)$ corresponding to its 3D real size. $\mathbf{S} = \{q_k = (u_k, v_k)\}_{k\in\{1,..,N\}}$ is the vehicle 2D part coordinates in the image. $\mathbf{S}^{3d} = \{p_k = (x_k, y_k, z_k)\}_{k\in\{1,..,N\}}$ is the vehicle 3D part coordinates in the 3D real word coordinate system. $\mathbf{V} = \{v_k\}_{k\in\{1,..,N\}}$ is the part visibility vector where v_k denotes the visibility class of the k^{th} part. Four classes of visibility are defined: (1) visible if the part is observed in the image, (2) occluded if the part is occluded by another object, (3) self-occluded if the part is occluded by the vehicle and (4) truncated if the part is out of the image. Figure 4 shows an example of a 2D/3D vehicle model.

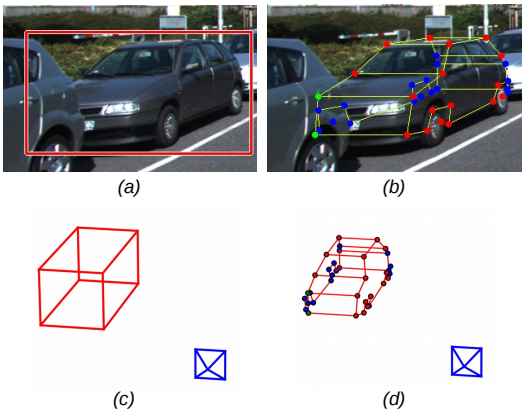

Figure 4. Example of one 2D/3D vehicle model. *(a)* the bounding box B, *(b)* 2D part coordinates $\mathbf{S}$ and part visibility $\mathbf{V}$: visible parts (red), occluded parts (green) and self-occluded parts (blue). *(c)* the 3D bounding box B^{3d} and *(d)* the associated 3D shape $\mathbf{S}^{3d}$.

3.3. Deep MANTA Network

The Deep MANTA network is designed to detect vehicles using a coarse-to-fine bounding box proposal as well as to output other finer attributes such as vehicle part localization, part visibility, and template similarity.

Coarse-to-fine forward. Given an entire input image, the network returns a first set of K object proposals $\mathbf{B}_1 = \{B_{i,1}\}_{i\in\{1,..,K\}}$ as the region proposal network proposed by [33]. These regions are then extracted from a feature map and pooled to a fixed size using ROI Pooling introduced by [14]. Extracted regions are forwarded in a network (sharing some weights with the first level) and refined by offset transformations. A second set of K objects $\mathbf{B}_2 = \{B_{i,2}\}_{i\in\{1,..,K\}}$ is proposed. This operation is repeated one last time to provide the final set of bounding box $\mathbf{B}_3$. These three levels of refinement are illustrated in Figure 2. This procedure differs than Faster-RCNN [33] in that our iterative refinement steps overcome the constraints of large object scale variations and provide more accurate detection. Furthermore, in our approach, ROI pooled regions are extracted on the first convolution feature maps for keeping high resolution to detect hard vehicles.

Many-task prediction. The Deep MANTA architecture outputs a final bounding box set $\mathbf{B}_3 = \{B_{i,3}\}_{i\in\{1,..,K\}}$. For each bounding box $B_{i,3}$, the MANTA network also returns all 2D vehicle part coordinates $\mathbf{S}_i$, part visibility $\mathbf{V}_i$ and 3D template similarity $\mathbf{T}_i$. The template similarity vector $\mathbf{T}_i$ is defined as $\mathbf{T}_i = \{r_m\}_{m\in\{1,..,M\}}$. $r_m = (r_x, r_y, r_z)$ corresponds to the three scaling factors to apply on the 3D template $\bar{t}_m^{3d}$ to fit the real 3D template of the detected vehicle i. This vector encodes the similarity between the detected vehicle and all the 3D templates $\{\bar{t}_m^{3d}\}_{m\in\{1,..,M\}}$ of the 3D template dataset.

At this stage of the approach, non-maximum suppression is performed to remove redundant detections. This provides a new set of K' detections and associated attributes $\{B_j, \mathbf{S}_j, \mathbf{V}_j, \mathbf{T}_j\}_{j\in\{1,..,K'\}}$.

3.4. Deep MANTA Inference

The inference step uses the Deep MANTA network outputs, the 3D shape dataset $\{\mathbf{S}_m^{3d}\}_{m\in\{1,..,M\}}$ and the 3D template dataset $\{\bar{t}_m^{3d}\}_{m\in\{1,..,M\}}$ defined in 3.1 to recover 3D information. Given a vehicle detection j provided by the Deep MANTA network, the inference consists in two steps. In the first step, we choose the closest 3D template $c \in \{1, .., M\}$ in the 3D template dataset $\{\bar{t}_m^{3d}\}_{m\in\{1,..,M\}}$ using the template similarity $\mathbf{T}_j = \{r_m\}_{m\in\{1,..,M\}}$ returned by the network. For each sample $\bar{t}_m^{3d}$ of the 3D template dataset we apply the scaling transformation r_m. The resulting 3D templates are defined by $\{t_m^{3d}\}_{m\in\{1,..,M\}}$. The best 3D template c is the one that minimizes the distance between t_m^{3d} and $\bar{t}_m^{3d}$:

$$c = \underset{m\in\{1,..,M\}}{\operatorname{argmin}} \; d(\bar{t}_m^{3d}, t_m^{3d}).$$

In other words, the best 3D template is the one that is predicted closer to $(1, 1, 1)$ by the Deep MANTA network.

In the second step, 2D/3D matching is applied using 3D shape $\mathbf{S}_c^{3d}$. It is rescaled to fit the 3D template $t_j = t_c^{3d}$. Then, a pose estimation algorithm is performed to match the rescaled 3D shape $\bar{\mathbf{S}}_c^{3d}$ with the 2D shape $\mathbf{S}_j$ using a standard 2D/3D matching [20]. This last step provides the 3D bounding box B_j^{3d} and the 3D part coordinates $\mathbf{S}_j^{3d}$. The last block in Figure 2 illustrates the inference step.

4. Deep MANTA Training

This section defines all the tasks of the MANTA network and the associated loss functions. In the following, we consider three levels of refinement $l \in \{1, 2, 3\}$ and five functions to minimize: $\mathcal{L}_{rpn}, \mathcal{L}_{det}, \mathcal{L}_{parts}, \mathcal{L}_{vis}$ and $\mathcal{L}_{temp}$. $\mathcal{L}_{rpn}$ is the RPN loss function defined in [33]. $\mathcal{L}_{det}$ is the detection loss function focusing on discriminating vehicle and background bounding box as well as regressing bounding boxes. $\mathcal{L}_{parts}$ is the loss corresponding to vehicle part localization. $\mathcal{L}_{vis}$ is the loss related to part visibility. $\mathcal{L}_{temp}$ is the loss related to template similarity. We use the Faster-RCNN framework [33] based on RPN to learn the end-to-end MANTA model. Given an input image, the network joint optimization minimizes the global function:

$$\mathcal{L} = \mathcal{L}^1 + \mathcal{L}^2 + \mathcal{L}^3$$

with

$$\mathcal{L}^1 = \mathcal{L}_{rpn},$$

$$\mathcal{L}^2 = \sum_i \mathcal{L}^2_{det}(i) + \mathcal{L}^2_{parts}(i),$$

$$\mathcal{L}^3 = \sum_i \mathcal{L}^3_{det}(i) + \mathcal{L}^3_{parts}(i) + \mathcal{L}_{vis}(i) + \mathcal{L}_{temp}(i),$$

where i is the index of a proposal object. These three losses correspond to the three levels of refinement of the Deep MANTA architecture: finer is the level, bigger is the amount of information learned.

4.1. Many-task loss functions

Here, we will detail the different task losses used in the global function presented above. In the following, each object proposal at each level of refinement l, is indexed by i and it is represented by its box $B_{i,l} = (c_{x_{i,l}}, c_{y_{i,l}}, w_{i,l}, h_{i,l})$. The closest ground-truth vehicle box B to $B_{i,l}$ is selected. Associated ground-truth parts $\mathbf{S}$, ground-truth visibility $\mathbf{V}$ and ground-truth template t are also selected (see section 3.2). We denote the standard log softmax loss as P and the robust SmoothL1 loss defined in [14] as R.

Detection loss. The object proposal i at the refinement level l is assigned to a class label $C_{i,l}$. $C_{i,l}$ is 1 if the object proposal is a vehicle and 0 otherwise. The classification criteria is the overlap between the box $B_{i,l}$ and the ground-truth box B. The predicted class returned by Deep MANTA network for the proposal is $C^*_{i,l}$. A target box regression vector $\Delta_{i,l} = (\delta_x, \delta_y, \delta_w, \delta_h)$ is also defined as follows:

$$\delta_x = (c_{x_{i,l}} - c_x)/w \qquad \delta_w = log(w_{i,l}/w)$$
$$\delta_y = (c_{y_{i,l}} - c_y)/h \qquad \delta_h = log(h_{i,l}/h)$$

The predicted regression vector returned by Deep MANTA network is $\Delta^*_{i,l}$. The detection loss function is defined by:

$$\mathcal{L}^l_{det}(i) = \lambda_{cls} P(C^*_{i,l}, C_{i,l}) + \lambda_{reg} C_{i,l} R(\Delta^*_{i,l} - \Delta_{i,l})$$

with λ_{cls} and λ_{reg} the regularization parameters of box classification and box regression respectively.

Part loss. Using the ground-truth parts $\mathbf{S} = (q_1, .., q_N)$ and the box $B_{i,l}$ associated to the object proposal i at level l, normalized vehicle parts $\mathbf{S}_{i,l} = (\bar{q}_1, .., \bar{q}_N)$ are computed as follows:

$$\bar{q}_k = (\frac{u_k - c_{x_{i,l}}}{w_{i,l}}, \frac{v_k - c_{y_{i,l}}}{h_{i,l}}).$$

The predicted normalized parts are $S^*_{i,l}$. The part loss function is defined as:

$$\mathcal{L}^l_{parts}(i) = \lambda_{parts} C_{i,l} R(\mathbf{S}^*_{i,l} - \mathbf{S}_{i,l})$$

with λ_{parts} the regularization parameter of part loss.

Visibility loss. This loss is only optimized on the final level of refinement $l = 3$. The ground-truth visibility vector $\mathbf{V}_i = \mathbf{V}$ is assigned to the object proposal i. The predicted visibility vector is $\mathbf{V}^*_i$. The visibility loss function is defined as:

$$\mathcal{L}_{vis}(i) = \lambda_{vis} C_{i,3} P(\mathbf{V}^*_i, \mathbf{V}_i)$$

with λ_{vis} the regularization parameter of visibility loss.

Template similarity loss. This loss is only optimized on the final level of refinement $l = 3$. Instead of directly optimizing the three dimensions of the 3D template t, we encode it as a vector $\mathbf{T}$ using the 3D template dataset as explained in 3.3. For training, the log function is applied to each element of $\mathbf{T}$ for better normalization (similarity values are thus in $[-1, 1]$). The ground-truth template similarity vector vector $\mathbf{T}_i = \mathbf{T}$ is assigned to the object proposal i. The predicted template similarity vector is $\mathbf{T}^*_i$. The template similarity loss function is defined as:

$$\mathcal{L}_{temp}(i) = \lambda_{temp} C_{i,3} R(\mathbf{T}^*_i - \mathbf{T}_i)$$

with λ_{temp} the regularization parameter of template similarity loss.

Notice that if the object proposal i is not positive (*i.e* $C_{i,l} = 0$) the loss functions associated to bounding box regression, part location, visibility and template similarity are null because it does not make sense to optimize vehicle properties on background regions.

4.2. Semi-automatic annotation

A semi-automatic annotation process is used to provide useful labels to train our Deep MANTA network (vehicles part coordinates, part visibility, 3D template). To perform the annotation process, we only need a weakly annotated real dataset providing 3D bounding boxes of vehicle and a 3D CAD dataset. For this purpose, we use a 3D CAD dataset composed of M 3D car models. We manually annotate N vertices on each 3D model. For each vehicle in the weakly annotated real dataset, we choose automatically

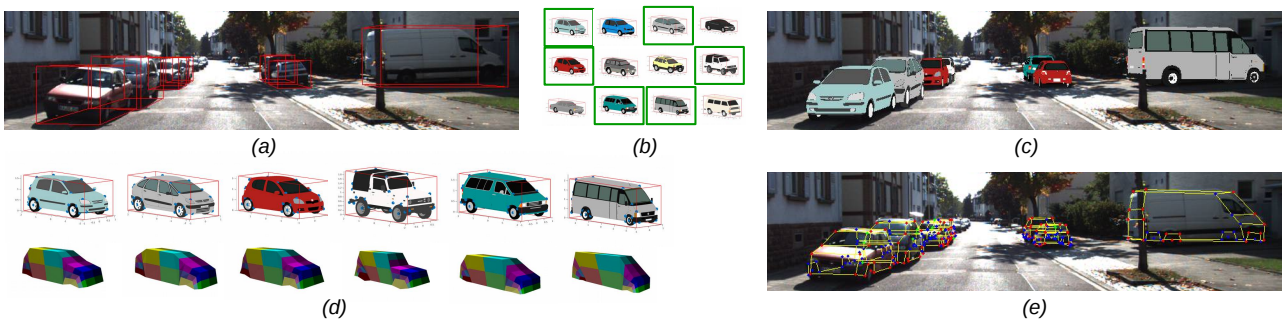

Figure 5. Semi-automatic annotation process. *(a)* weak annotations on a real image (3D bounding box). *(b)* best corresponding 3D models in green. *(c)* projection of these 3D models in the image. *(d)* corresponding mesh of visibility (each color represents a part). *(e)* Final annotations (part localization and visibility). Red dots: visible parts, green dots: occluded parts, bleu dots: self-occluded parts.

the best corresponding 3D model in the 3D model dataset. This is done by choosing the 3D model which has its 3D bounding box closest to the real 3D vehicle bounding box in the image (in terms of 3D dimensions). 3D parts associated to the chosen CAD are projected onto the image to get 2D part coordinates. The visibility of each projected part is computed using a mesh of visibility. This mesh is a low resolution 3D model where each face is associated to an annotated vehicle 3D part. Figure 5 illustrates this process.

5. Experiments

In this section, we evaluate the proposed approach on the challenging KITTI object detection benchmark dedicated to autonomous driving [12]. This dataset is composed of 7481 training images and 7518 testing images. The calibration matrix is given. Since ground truth annotations for the testing set are not released, we use train/validation splits from the training set to validate our method. To compare our approach to other state-of-the-art methods, we use two train/val splits: *val1* used by [40, 39] and *val2* used by [8, 7]. This is a means to compare our approach to these methods for tasks which are not initially evaluated on the KITTI benchmark. We use the 3D CAD dataset provided by [11, 6] composed of $M = 103$ 3D vehicle models for semi-automatic annotation. We annotate $N = 36$ vehicle parts on each 3D model. We train the Deep MANTA using the GoogLenet [36] and the VGG16 [35] architectures with the standard stochastic gradient descent optimization. The Deep MANTA is initialized using pre-trained weights learned on ImageNet. We use 7 aspect ratios and 10 scales for the RPN providing 70 anchors at each feature map location as proposed by [40]. During training, an object proposal is considered positive if its overlap with a ground-truth box is greater than 0.7. For experiments, all regularization parameters λ are set to 1 except for the part localization task where $\lambda_{parts} = 3$. The choice of these parameters are discussed at the end of this section.

We present results for several tasks: 2D vehicle detection and orientation, 3D localization, 2D part localization, part visibility and 3D template prediction. In all presented results, we use 200 object proposals and an overlapping threshold of 0.5 for non-maximum suppression. Results are presented for three levels of difficulty (Easy, Moderate and Hard) as proposed by the KITTI Benchmark [12].

2D vehicle detection and orientation. We use mean Average Precision (mAP) with overlapping criteria of 0.7 to evaluate 2D vehicle detection. We use average orientation similarity (AOS) to evaluate vehicle orientation as proposed by the KITTI Benchmark [12]. Table 1 shows results for these two tasks on the two train/val splits. Table 2 shows results on the KITTI testing set. We can see that our method outperforms others for the two tasks on the two train/val split as well as on the test set. In addition, our approach is less time consuming. This is due to the resolution of the input image. Many state-of-the-art object proposal based approaches [40, 7, 8] upscale the input image by a factor of 3 on the KITTI dataset. This is done to not lose information on spatially reduced feature maps. Our coarse-to-fine approach overcomes this loss of information and that allows to give an input image at initial resolution. The coarse-to-fine architecture of the Deep MANTA is also evaluated and results are shown in Table 3. We compare the presented Deep MANTA to two other networks. The first line is a network which does not use refinement steps and where pooling regions are extracted on the feature map at the 5th level of convolution (as the original Faster-RCNN [33]). The second line is a network without refinement steps and where pooling regions are extracted at the first level of convolution. We can see that extracting regions on the first convolution level clearly boosts detection and orientation score (around 24% up for moderate). The last line is the presented Deep MANTA architecture (with refinement step and regions extracted on the first convolution maps). These results shows that the coarse-to-fine architecture increases detection and orientation estimation (around 4% up for moderate).

			AP			AOS		
Method	Type	Time	Easy	Moderate	Hard	Easy	Moderate	Hard
3DVP [39]	Mono	40 s	80.48 / -	68.05 / -	57.20 / -	78.99 / -	65.73 / -	54.67 / -
Faster-RCNN [33]	Mono	2 s	82.91 / -	77.83 / -	66.25 / -	- / -	- / -	- / -
SubCNN [40]	Mono	2 s	95.77 / -	86.64 / -	74.07 / -	94.55 / -	85.03 / -	72.21 / -
3DOP [8]	Stereo	3 s	- / 94.49	- / 89.65	- / 80.97	- / 92.98	- / 87.34	- / 78.24
Mono3D [7]	Mono	4.2 s	- / 95.75	- / 90.01	- / 80.66	- / 93.70	- / 87.61	- / 78.00
Ours GoogLenet	Mono	0.7 s	**97.90 / 97.58**	91.01 / 90.89	**83.14** / 82.72	**97.60 / 97.44**	90.66 / 90.66	**82.66** / 82.35
Ours VGG16	Mono	2 s	97.45 / 97.2	**91.47 / 91.85**	81.79 / **85.15**	97.10 / 97.09	**91.01 / 91.57**	81.14 / **84.72**

Table 1. Results for 2D vehicle detection (AP) and orientation (AOS) on KITTI val sets. Results on the two validation sets: *val1* / *val2*.

	AP			AOS		
	Easy	Moderate	Hard	Easy	Moderate	Hard
LSVM-MDPM-sv [10, 13]	68.2	56.48	44.18	67.27	55.77	43.59
ACF-SC [3]	69.11	58.66	45.95	-	-	-
MDPM-un-BB [10]	71.19	62.16	48.43	-	-	-
DPM-VOC+VP [31]	74.95	64.71	48.76	72.28	61.84	46.54
OC-DPM [30]	75.94	65.95	53.56	73.50	64.42	52.40
SubCat [28]	84.14	75.46	59.71	83.41	74.42	58.83
3DVP [39]	87.46	75.77	65.38	87.46	75.77	65.38
AOG [21]	84.80	75.94	60.70	33.79	30.77	24.75
Regionlets [25]	84.75	76.45	59.70	-	-	-
Faster R-CNN [33]	86.71	81.84	71.12	-	-	-
3DOP [8]	93.04	88.64	79.10	91.44	86.10	76.52
Mono3D [7]	92.33	88.66	78.96	91.01	86.62	76.84
SDP + RPN [43]	90.14	88.85	78.38	-	-	-
MS-CNN [4]	90.03	89.02	76.11	-	-	-
SubCNN [40]	90.81	89.04	79.27	90.67	88.62	78.68
Ours Googlenet	95.77	90.03	80.62	95.72	89.86	80.39
Ours VGG16	**96.40**	**90.10**	**80.79**	**96.32**	**89.91**	**80.55**

Table 2. Results for 2D vehicle detection (AP) and orientation (AOS) on the KITTI test set.

3D localization. We use Average Localization Precision (ALP) metric proposed by [39]. It consists in replacing orientation similarity in AOS with localization precision. A 3D location is correct if its distance from the ground truth 3D location is smaller than a threshold. Table 4 presents results on the two train/val splits for a threshold distance of 1 meter and 2 meters. Our Deep MANTA approach clearly outperforms other monocular approaches [7, 39] for the 3D localization task (around 16% up compared to Mono3D [7]). Figure 6 shows recall/3D localization precision curves of Deep MANTA and Mono3D [7]. Compared to 3DOP [8], which uses stereo information, the Deep MANTA performances are equivalent at a threshold error distance of 2 meters but less accurate at 1 meter: Deep MANTA only uses a single image contrarily to the 3DOP approach which uses disparity information.

3D template, part localization and visibility. We also evaluate the precision of part localization, part visibility classification accuracy as well as 3D template prediction. Given a correct detection, we use the following three metrics. For part localization, a part is considered well localized if the normalized distance to the ground-truth part is less than a threshold (20 pixels). Distances are normalized using a fixed bounding box height (155 pixels) as proposed by [45]. The visibility metric is the accuracy over the four visibility classes. Finally, we evaluate 3D template prediction by comparing the three predicted dimensions (w, h, l) to the ground-truth 3D box dimensions (w_{gt}, h_{gt}, l_{gt}) pro-

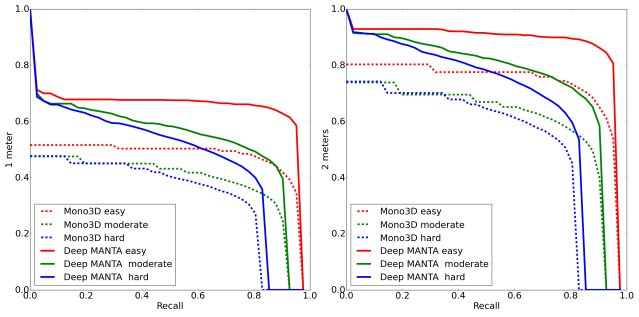

Figure 6. Recall/3D localization precision curves for 1 meter (left) and 2 meters (right) precision on the *val2* used by Mono3D [7].

Methode	Refinement	ROI Pooling on	AP			AOS		
			Easy	Moderate	Hard	Easy	Moderate	Hard
Deep MANTA	No	conv5	80.64	62.45	53.86	79.68	61.49	52.58
	No	conv1	95.19	86.85	78.62	94.98	86.52	78.05
	Yes	conv1	**97.58**	**90.89**	**82.72**	**97.44**	**90.66**	**82.35**

Table 3. Coarse-to-fine comparison for 2D vehicle detection (AP) and orientation estimation (AOS) on the validation set *val2*. These experiments show the importance of the refinement step as well as the influence of the feature maps chosen for region extraction.

			1 meter			2 meters		
Method	Type	Time	Easy	Moderate	Hard	Easy	Moderate	Hard
3DVP [39]	Mono	40 s	45.61 / -	34.28 / -	27.72 / -	65.73 / -	54.60 / -	45.62 / -
3DOP [8]	Stereo	3 s	- / **81.97**	- / **68.15**	- / **59.85**	- / **91.46**	- / **81.63**	- / **72.97**
Mono3D [7]	Mono	4.2 s	- / 48.31	- / 38.98	- / 34.25	- / 74.77	- / 60.91	- / 54.24
Ours GoogLenet	Mono	0.7 s	**70.90** / 65.71	**58.05** / 53.79	**49.00** / 47.21	**90.12** / 89.29	**77.02** / 75.92	**66.09** / 67.28
Ours VGG16	Mono	2 s	66.88 / 69.72	53.17 / 54.44	44.40 / 47.77	88.32 / 91.01	74.31 / 76.38	63.62 / 67.77

Table 4. 3D localization accuracy (ALP) on KITTI val sets for 1 meter and 2 meters precision. Results on the two validation sets: *val1* / *val2*.

vided by KITTI. A 3D template (w, h, l) is considered correct if $|\frac{w_{gt}-w}{w_{gt}}| < 0.2$ and $|\frac{h_{gt}-h}{h_{gt}}| < 0.2$ and $|\frac{l_{gt}-l}{l_{gt}}| < 0.2$. Table 5 shows the good performances for these tasks.

Metric	Easy	Moderate	Hard
Part localization	97.54	90.79	82.64
Part visibility	92.48	85.08	76.90
3D template	94.04	86.62	78.72

Table 5. Part localization, part visibility, 3D template evaluation on the validation set *val2*.

Many-task and regularization parameters. Table 6 shows results with different sets of regularization parameters. These results also aim to compare performances of the Deep MANTA approach with networks optimized on fewer tasks. In Table 6, D corresponds to the detection task, P to the part localization task, V to the part visibility task and T to the template similarity task. With these notations, the first line of Table 6 is the Deep MANTA trained only on the detection task ($\lambda_{parts} = \lambda_{vis} = \lambda_{temp} = 0$). As part localization and template similarity are not trained, orientation and 3D localization cannot be predicted in this case. The second line is the Deep MANTA trained without the visibility task ($\lambda_{vis} = 0$) and with $\lambda_{parts} = 3$. The third line is the complete Deep MANTA (all tasks) but with the regularization parameter associated to part localization $\lambda_{parts} = 1$. Finally, the last line is the Deep MANTA with $\lambda_{parts} = 3$ (the one presented in all above results). These results are interesting for several reasons. First, we can see that increasing the number of learned tasks (*i.e* enriching the vehicle description) does not significantly affect performances (it is slightly higher for detection and orientation accuracy but slightly lower on 3D localization). That proves the relevance of the Many-Task concept: a neural network is able to learn one feature representation which can be used to predict many tasks. Secondly, we can see that the parameter λ_{parts} is very important for 3D localization. Learning the Deep MANTA with $\lambda_{parts} = 3$ improves the 3D localization by 6% for 1 meter distance precision.

	AP	AOS	1 m	2 m
D	89.86	-	-	-
DPT / $\lambda_{parts} = 3$	89.73	89.39	**58.37**	**78.11**
DPVT / $\lambda_{parts} = 1$	89.58	89.27	51.47	73.93
DPVT / $\lambda_{parts} = 3$	**90.54**	**90.23**	57.44	77.58

Table 6. The influence of the amount of tasks learned as well as different regularization parameters. This table gives results for vehicle detection (AP), orientation (AOS), and 3D localization for 1 meter and 2 meters precision (ALP). Given results are averaged over the two validation sets and over the three levels of difficulty (Easy, Moderate, Hard). See text for details.

6. Conclusion

To conclude, we propose a new approach for joint 2D and 3D vehicle analysis from monocular image. It is based on the Many-task CNN (Deep MANTA) which proposes accurate 2D vehicle bounding boxes using multiple refinement steps. The MANTA architecture also provides vehicle part coordinates (even if these parts are hidden), part visibility and 3D template for each detection. These fine features are then used to recover vehicle orientation and 3D localization using robust 2D/3D point matching. Our approach outperforms state-of-the-art methods for vehicle detection and fine orientation estimation and clearly increases vehicle 3D localization compared to monocular approaches. One perspective is to adapt this framework to other rigid objects and build a multi-class Deep MANTA network.

References

[1] P. Arbeláez, J. Pont-Tuset, J. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. *CVPR*, 2014.

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

[3] C. Cadena, A. Dick, and I. Reid. A fast, modular scene understanding system using context-aware object detection. *ICRA*, 2015.

[4] Z. Cai, Q. Fan, R. Feris, and N. Vasconcelos. A unified multi-scale deep convolutional neural network for fast object detection. *ECCV*, 2016.

[5] J. Carreira and et al. Constrained parametric min-cuts for automatic object segmentation. *CVPR*, 2010.

[6] L.-C. Chen, S. Fidler, A. L. Yuille, and R. Urtasun. Beat the mturkers: Automatic image labeling from weak 3d supervision. 2014.

[7] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun. Monocular 3d object detection for autonomous driving. *CVPR*, 2016.

[8] X. Chen, K. Kundu, Y. Zhu, A. Berneshawi, H. Ma, S. Fidler, and R. Urtasun. 3d object proposals for accurate object class detection. *NIPS*, 2015.

[9] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *PAMI*, 2013.

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

[11] S. Fidler, S. Dickinson, and R. Urtasun. 3d object detection and viewpoint estimation with a deformable 3d cuboid model. *NIPS*, 2012.

[12] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. *CVPR*, 2012.

[13] A. Geiger, C. Wojek, and R. Urtasun. Joint 3d estimation of objects and scene layout. *NIPS*, 2011.

[14] R. Girshick. Fast r-cnn. *ICCV*, 2015.

[15] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CVPR*, 2014.

[16] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CVPR*, 2016.

[17] J.J.Lim, A.Khosla, and A.Torralba. Fpm : Fine pose parts-based model with 3d cad models. *ECCV*, 2014.

[18] T. Kong, A. Yao, Y. Chen, and F. Sun. Hypernet: Towards accurate region proposal generation and joint object detection. *CVPR*, 2016.

[19] J. Krause, H. Jin, J. Yang, and L. Fei-Fei. Fine-Grained Recognition without Part Annotations. *CVPR*, 2015.

[20] V. Lepetit, F.Moreno-Noguer, and P.Fua. Epnp: An accurate o(n) solution to the pnp problem. *IJCV*, 2009.

[21] B. Li, T. Wu, and S.-C. Zhu. Integrating context and occlusion for car detection by hierarchical and-or model. *ECCV*, 2014.

[22] J. Liebelt and C. Schmid. Multi-view object class detection with a 3d geomtrique model. *CVPR*, 2010.

[23] J. Lim, H. Pirsiavash, and A. Torralba. Parsing ikea objects: Fine pose estimation. *ICCV*, 2013.

[24] Y.-L. Lin, V. Morariu, W. Hsu, and L. Davis. Jointly Optimizing 3D Model Fitting and Fine-Grained Classification. *ECCV*, 2014.

[25] C. Long, X. Wang, G. Hua, M. Yang, and Y. Lin. Accurate object detection with location relaxation and regionlets relocalization. *ACCV*, 2014.

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

[27] R. Lopez-Sastre, T. Tuytelaars, and S. Savarase. Deformable part models revisited: A performance evaluation for object category pose estimation. *ICCV*, 2011.

[28] E. Ohn-Bar and M. M. Trivedi. Learning to detect vehicles by clustering appearance patterns. *T-ITS*, 2015.

[29] B. Pepi, M. Stark, P. Gehler, and B. Schielen. Teaching 3d geometry to deformable part models. *CVPR*, 2012.

[30] B. Pepik, M. Stark, P. Gehler, and B. Schiele. Occlusion patterns for object class detection. *CVPR*, 2013.

[31] B. Pepik, M. Stark, P. Gehler, and B. Schiele. Multi-view and 3d deformable part models. *TPAMI*, 2015.

[32] B. Pepik, M. Stark, P. V. Gehler, T. Ritschel, and B. Schiele. 3d object class detection in the wild. *CVPR*, 2015.

[33] S. Ren, K. He, R. B. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *NIPS*, 2015.

[34] P. Sermanet, X. Z. D. Eigen, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *ICLR*, 2014.

[35] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2014.

[36] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *CVPR*, 2015.

[37] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface : Closing the gap to human-level performance in face verification. *CVPR*, 2014.

[38] J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders. Selective search for object recognition. *IJCV*, 2013.

[39] Y. Xiang, W. Choi, Y. Lin, and S. Savarese. Data-driven 3d voxel patterns for object category recognition. *CVPR*, 2015.

[40] Y. Xiang, W. Choi, Y. Lin, and S. Savarese. Subcategory-aware convolutional neural networks for object proposals and detection. *arXiv:1604.04693*, 2016.

[41] Y. Xiang, R. Mottaghi, and S. Savarese. Beyond pascal : A benchmark for 3d object detection in the wild. *WACV*, 2014.

[42] Y. Xiang and S. Savareses. Estimating the aspect layout of object categories. *CVPR*, 2012.

[43] F. Yang, W. Choi, and Y. Lin. Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers. *CVPR*, 2016.

[44] M. Z. Zia, M. Stark, B. Schiele, and K. Schindler. Revisiting 3d geometric models for accurate object shape and pose. *ICCV-WS*, 2011.

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

[46] M. Z. Zia, M. Stark, and K. Schindler. Explicit occlusion modeling for 3d object class representations. *CVPR*, 2013.

[47] M. zuysal, V. .Lepetit, and P. Fua. Pose estimation for category specific multiview object localization. *CVPR*, 2009.