

Orthographic Feature Transform for Monocular 3D Object Detection

Thomas Roddick Alex Kendall Roberto Cipolla
 University of Cambridge
 {tr346, agk34, rc10001}@cam.ac.uk

Abstract

3D object detection from monocular images has proven to be an enormously challenging task, with the performance of leading systems not yet achieving even 10% of that of LiDAR-based counterparts. One explanation for this performance gap is that existing systems are entirely at the mercy of the perspective image-based representation, in which the appearance and scale of objects varies drastically with depth and meaningful distances are difficult to infer. In this work we argue that the ability to reason about the world in 3D is an essential element of the 3D object detection task. To this end, we introduce the orthographic feature transform, which enables us to escape the image domain by mapping image-based features into an orthographic 3D space. This allows us to reason holistically about the spatial configuration of the scene in a domain where scale is consistent and distances between objects are meaningful. We apply this transformation as part of an end-to-end deep learning architecture and achieve state-of-the-art performance on the KITTI 3D object benchmark.¹

1. Introduction

The success of any autonomous agent is contingent on its ability to detect and localize the objects in its surrounding environment. Prediction, avoidance and path planning all depend on robust estimates of the 3D positions and dimensions of other entities in the scene. This has led to 3D bounding box detection emerging as an important problem in computer vision and robotics, particularly in the context of autonomous driving. To date the 3D object detection literature has been dominated by approaches which make use of rich LiDAR point clouds [37, 33, 15, 27, 5, 6, 22, 1], while the performance of image-only methods, which lack the absolute depth information of LiDAR, lags significantly behind. Given the high cost of existing LiDAR units, the sparsity of LiDAR point clouds at long ranges, and the need for sensor redundancy, accurate 3D object detection from

¹We will release full source code and pretrained models upon acceptance of this manuscript for publication.



Figure 1. 3D bounding box detection from monocular images. The proposed system maps image-based features to an orthographic birds-eye-view and predicts confidence maps and bounding box offsets in this space. These outputs are then decoded via non-maximum suppression to yield discrete bounding box predictions.

monocular images remains an important research objective. To this end, we present a novel 3D object detection algorithm which takes a single monocular RGB image as input and produces high quality 3D bounding boxes, achieving state-of-the-art performance among monocular methods on the challenging KITTI benchmark [8].

Images are, in many senses, an extremely challenging modality. Perspective projection implies that the scale of a single object varies considerably with distance from the camera; its appearance can change drastically depending on the viewpoint; and distances in the 3D world cannot be inferred directly. These factors present enormous challenges to a monocular 3D object detection system. A far more innocuous representation is the orthographic birds-eye-view map commonly employed in many LiDAR-based methods [37, 33, 1]. Under this representation, scale is homogeneous; appearance is largely viewpoint-independent; and distances between objects are meaningful. Our key insight therefore is that as much reasoning as possible should be performed in this orthographic space rather than directly on the pixel-based image domain. This insight proves essential to the success of our proposed system.

It is unclear, however, how such a representation could be constructed from a monocular image alone. We therefore introduce the *orthographic feature transform* (OFT): a differentiable transformation which maps a set of features extracted from a perspective RGB image to an orthographic birds-eye-view feature map. Crucially, we do not rely on any explicit notion of depth: rather our system builds up an internal representation which is able to determine which

features from the image are relevant to each location on the birds-eye-view. We apply a deep convolutional neural network, the *topdown* network, in order to reason locally about the 3D configuration of the scene.

The main contributions of our work are as follows:

1. We introduce the orthographic feature transform (OFT) which maps perspective image-based features into an orthographic birds-eye-view, implemented efficiently using integral images for fast average pooling.
2. We describe a deep learning architecture for predicting 3D bounding boxes from monocular RGB images.
3. We highlight the importance of reasoning in 3D for the object detection task.

The system is evaluated on the challenging KITTI 3D object benchmark and achieves state-of-the-art results among monocular approaches.

2. Related Work

2D object detection Detecting 2D bounding boxes in images is a widely studied problem and recent approaches are able to excel even on the most formidable datasets [30, 7, 19]. Existing methods may broadly be divided into two main categories: *single stage* detectors such as YOLO [28], SSD [20] and RetinaNet [18] which predict object bounding boxes directly and *two-stage* detectors such as Faster-RCNN [29] and FPN [17] which add an intermediate region proposal stage. To date the vast majority of 3D object detection methods have adopted the latter philosophy, in part due to the difficulty in mapping from fixed-sized regions in 3D space to variable-sized regions in the image space. We overcome this limitation via our OFT transform, allowing us to take advantage of the purported speed and accuracy benefits [18] of a single-stage architecture.

3D object detection from LiDAR 3D object detection is of considerable importance to autonomous driving, and a large number of LiDAR-based methods have been proposed which have enjoyed considerable success. Most variation arises from how the LiDAR point clouds are encoded. The Frustum-PointNet of Qi *et al.* [27] and the work of Du *et al.* [6] operate directly on the point clouds themselves, considering a subset of points which lie within a frustum defined by a 2D bounding box on the image. Minemura *et al.* [22] and Li *et al.* [16] instead project the point cloud onto the image plane and apply Faster-RCNN-style architectures to the resulting RGB-D images. Other methods, such as TopNet [33], BirdNet [1] and Yu *et al.* [37], discretize the point cloud into some birds-eye-view (BEV) representation which encodes features such as returned intensity or average height of points above the ground plane. This representation turns out to be extremely attractive since it does not

exhibit any of the perspective artifacts introduced in RGB-D images for example, and a major focus of our work is therefore to develop an implicit image-only analogue to these birds-eye-view maps. A further interesting line of research is sensor fusion methods such as AVOD [15] and MV3D [5] which make use of 3D object proposals on the ground plane to aggregate both image-based and birds-eye-view features: an operation which is closely related to our orthographic feature transform.

3D object detection from images Obtaining 3D bounding boxes from images, meanwhile, is a much more challenging problem on account of the absence of absolute depth information. Many approaches start from 2D bounding boxes extracted using standard detectors described above, upon which they either directly regress 3D pose parameters for each region [14, 26, 24, 23] or fit 3D templates to the image [2, 35, 36, 38]. Perhaps most closely related to our work is Mono3D [3] which densely spans the 3D space with 3D bounding box proposals and then scores each using a variety of image-based features. Other works which explore the idea of dense 3D proposals in the world space are 3DOP [4] and Pham and Jeon [25], which rely on explicit estimates of depth using stereo geometry. A major limitation of all the above works is that each region proposal or bounding box is treated independently, precluding any joint reasoning about the 3D configuration of the scene. Our method performs a similar feature aggregation step to [3], but applies a secondary convolutional network to the resulting proposals whilst retaining their spatial configuration.

Integral images Integral images have been fundamentally associated with object detection ever since their introduction in the seminal work of Viola and Jones [32]. They have formed an important component in many contemporary 3D object detection approaches including AVOD [15], MV3D [5], Mono3D [3] and 3DOP [4]. In all of these cases however, integral images do not backpropagate gradients or form part of a fully end-to-end deep learning architecture. To our knowledge, the only prior work to do so is that of Kasagi *et al.* [13], which combines a convolutional layer and an average pooling layer to reduce computational cost.

3. 3D Object Detection Architecture

In this section we describe our full approach for extracting 3D bounding boxes from monocular images. An overview of the system is illustrated in Figure 3. The algorithm comprises five main components:

1. A front-end ResNet [10] feature extractor which extracts multi-scale feature maps from the input image.
2. A orthographic feature transform which transforms the image-based feature maps at each scale into an orthographic birds-eye-view representation.

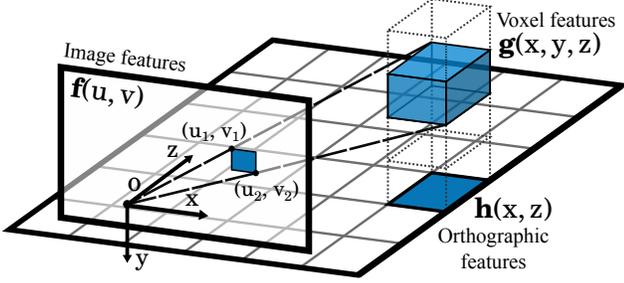


Figure 2. Orthographic Feature Transform (OFT). Voxel-based features $\mathbf{g}(x, y, z)$ are generated by accumulating image-based features $\mathbf{f}(u, v)$ over the projected voxel area. The voxel features are then collapsed along the vertical dimension to yield orthographic ground plane features $\mathbf{h}(x, z)$.

3. A *topdown* network, consisting of a series of ResNet residual units, which processes the birds-eye-view feature maps in a manner which is invariant to the perspective effects observed in the image.
4. A set of output heads which generate, for each object class and each location on the ground plane, a confidence score, position offset, dimension offset and a orientation vector.
5. A non-maximum suppression and decoding stage, which identifies peaks in the confidence maps and generates discrete bounding box predictions.

The remainder of this section will describe each of these components in detail.

3.1. Feature extraction

The first element of our architecture is a convolutional feature extractor which generates a hierarchy of multi-scale 2D feature maps from the raw input image. These features encode information about low-level structures in the image, which form the basic components used by the topdown network to construct an implicit 3D representation of the scene. The front-end network is also responsible for inferring depth information based on the size of image features since subsequent stages of the architecture aim to eliminate variance to scale.

3.2. Orthographic feature transform

In order to reason about the 3D world in the absence of perspective effects, we must first apply a mapping from feature maps extracted in the image space to orthographic feature maps in the world space, which we term the Orthographic Feature Transform (OFT).

The objective of the OFT is to populate the 3D *voxel* feature map $\mathbf{g}(x, y, z) \in \mathbb{R}^n$ with relevant n -dimensional features from the *image*-based feature map $\mathbf{f}(u, v) \in \mathbb{R}^n$ extracted by the front-end feature extractor. The voxel map is defined over a uniformly spaced 3D lattice \mathcal{G} which is

fixed to the ground plane a distance y_0 below the camera and has dimensions W, H, D and a voxel size of r . For a given voxel grid location $(x, y, z) \in \mathcal{G}$, we obtain the voxel feature $\mathbf{g}(x, y, z)$ by accumulating features over the area of the image feature map \mathbf{f} which corresponds to the voxel’s 2D projection. In general each voxel, which is a cube of size r , will project to hexagonal region in the image plane. We approximate this by a rectangular bounding box with top-left and bottom-right corners (u_1, v_1) and (u_2, v_2) which are given by

$$\begin{aligned} u_1 &= f \frac{x - 0.5r}{z + 0.5 \frac{x}{|x|} r} + c_u, & v_1 &= f \frac{y - 0.5r}{z + 0.5 \frac{y}{|y|} r} + c_v, \\ u_2 &= f \frac{x + 0.5r}{z - 0.5 \frac{x}{|x|} r} + c_u, & v_2 &= f \frac{y + 0.5r}{z - 0.5 \frac{y}{|y|} r} + c_v \end{aligned} \quad (1)$$

where f is the camera focal length and (c_u, c_v) the principle point.

We can then assign a feature to the appropriate location in the voxel feature map \mathbf{g} by average pooling over the projected voxel’s bounding box in the image feature map \mathbf{f} :

$$\mathbf{g}(x, y, z) = \frac{1}{(u_2 - u_1)(v_2 - v_1)} \sum_{u=u_1}^{u_2} \sum_{v=v_1}^{v_2} \mathbf{f}(u, v) \quad (2)$$

The resulting voxel feature map \mathbf{g} already provides a representation of the scene which is free from the effects of perspective projection. However deep neural networks which operate on large voxel grids are typically extremely memory intensive. Given that we are predominantly interested in applications such as autonomous driving where most objects are fixed to the 2D ground plane, we can make the problem more tractable by collapsing the 3D voxel feature map down to a third, two-dimensional representation which we term the *orthographic* feature map $\mathbf{h}(x, z)$. The orthographic feature map is obtained by summing voxel features along the vertical axis after multiplication with a set of learned weight matrices $W(y) \in \mathbb{R}^{n \times n}$:

$$\mathbf{h}(x, z) = \sum_{y=y_0}^{y_0+H} W(y) \mathbf{g}(x, y, z) \quad (3)$$

Transforming to an intermediate voxel representation before collapsing to the final orthographic feature map has the advantage that the information about the vertical configuration of the scene is retained. This turns out to be essential for downstream tasks such as estimating the height and vertical position of object bounding boxes.

3.2.1 Fast average pooling with integral images

A major challenge with the above approach is the need to aggregate features over a very large number of regions. A

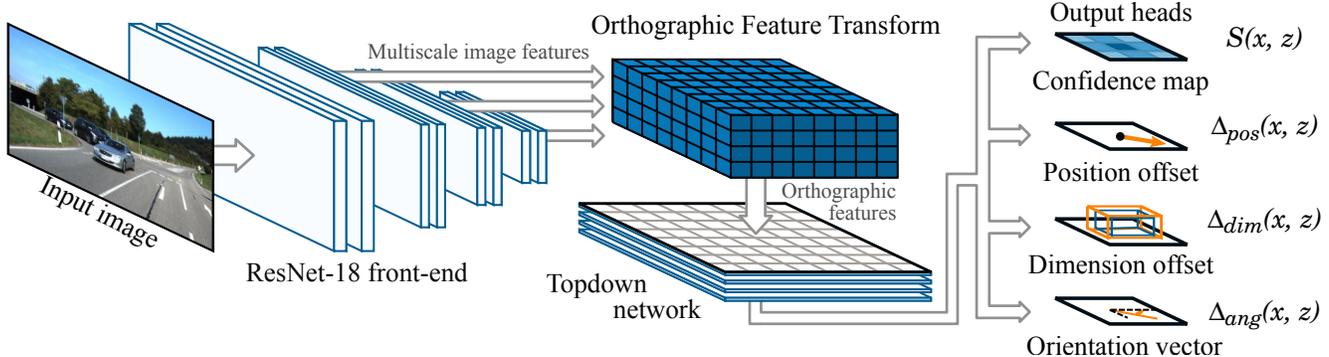


Figure 3. Architecture overview. A front-end ResNet feature extractor generates image-based features, which are mapped to an orthographic representation via our proposed orthographic feature transform. The topdown network processes these features in the birds-eye-view space and at each location on the ground plane predicts a confidence score S , a position offset Δ_{pos} , a dimension offset Δ_{dim} and an angle vector Δ_{ang} .

typical voxel grid setting generates around 150k bounding boxes, which far exceeds the $\sim 2k$ regions of interest used by the Faster R-CNN [29] architecture, for example. To facilitate pooling over such a large number of regions, we make use of a fast average pooling operation based on integral images [32]. An integral image, or in this case integral feature map, \mathbf{F} , is constructed from an input feature map \mathbf{f} using the recursive relation

$$\mathbf{F}(u, v) = \mathbf{f}(u, v) + \mathbf{F}(u-1, v) + \mathbf{F}(u, v-1) - \mathbf{F}(u-1, v-1). \quad (4)$$

Given the integral feature map \mathbf{F} , the output feature $\mathbf{g}(x, y, z)$ corresponding to the region defined by bounding box coordinates (u_1, v_1) and (u_2, v_2) (see Equation 1), is given by

$$\mathbf{g}(x, y, z) = \frac{\mathbf{F}(u_1, v_1) + \mathbf{F}(u_2, v_2) - \mathbf{F}(u_1, v_2) - \mathbf{F}(u_2, v_1)}{(u_2 - u_1)(v_2 - v_1)} \quad (5)$$

The complexity of this pooling operation is independent of the size of the individual regions, which makes it highly appropriate for our application where the size and shape of the regions varies considerably depending on whether the voxel is close to or far from the camera. It is also fully differentiable in terms of the original feature map \mathbf{f} and so can be used as part of an end-to-end deep learning framework.

3.3. Topdown network

A core contribution of this work is to emphasize the importance of reasoning in 3D for object recognition and detection in complex 3D scenes. In our architecture, this reasoning component is performed by a sub-network which we term the topdown network. This is a simple convolutional network with ResNet-style skip connections which operates on the 2D feature maps \mathbf{h} generated by the previously described OFT stage. Since the filters of the topdown network are applied convolutionally, all processing is invariant to the location of the feature on the ground plane. This

means that feature maps which are distant from the camera receive exactly the same treatment as those that are close, despite corresponding a much smaller region of the image. The ambition is that the final feature representation will therefore capture information purely about the underlying 3D structure of the scene and not its 2D projection.

3.4. Confidence map prediction

Among both 2D and 3D approaches, detection is conventionally treated as a classification problem, with a cross entropy loss used to identify regions of the image which contain objects. In our application however we found it to be more effective to adopt the confidence map regression approach of Huang *et al.* [11]. The confidence map $S(x, z)$ is a smooth function which indicates the probability that there exists an object with a bounding box centred on location (x, y_0, z) , where y_0 is the distance of the ground plane below the camera. Given a set of N ground truth objects with bounding box centres $\mathbf{p}_i = [x_i \ y_i \ z_i]^\top, i = 1, \dots, N$, we compute the ground truth confidence map as a smooth Gaussian region of width σ around the center of each object. The confidence at location (x, z) is given by

$$S(x, z) = \max_i \exp\left(-\frac{(x_i - x)^2 + (z_i - z)^2}{2\sigma^2}\right). \quad (6)$$

The confidence map prediction head of our network is trained via an ℓ_1 loss to regress to the ground truth confidence for each location on the orthographic grid \mathcal{H} . A well-documented challenge is that there are vastly fewer positive (high confidence) locations than negative ones, which leads to the negative component of the loss dominating optimization [31, 18]. To overcome this we scale the loss corresponding to negative locations (which we define as those with $S(x, z) < 0.05$) by a constant factor of 10^{-2} .

3.5. Localization and bounding box estimation

The confidence map S encodes a coarse approximation of the location of each object as a peak in the confidence score, which gives a position estimate accurate up to the resolution r of the feature maps. In order to localize each object more precisely, we append an additional network output head which predicts the relative offset Δ_{pos} from grid cell locations on the ground plane (x, y_0, z) to the center of the corresponding ground truth object p_i :

$$\Delta_{pos}(x, z) = \left[\frac{x_i - x}{\sigma} \quad \frac{y_i - y_0}{\sigma} \quad \frac{z_i - z}{\sigma} \right]^\top \quad (7)$$

We use the same scale factor σ as described in Section 3.4 to normalize the position offsets within a sensible range. A ground truth object instance i is assigned to a grid location (x, z) if any part of the object’s bounding box intersects the given grid cell. Cells which do not intersect any ground truth objects are ignored during training.

In addition to localizing each object, we must also determine the size and orientation of each bounding box. We therefore introduce two further network outputs. The first, the dimension head, predicts the logarithmic scale offset Δ_{dim} between the assigned ground truth object i with dimensions $d_i = [w_i \ h_i \ l_i]$ and the mean dimensions $\bar{d} = [\bar{w} \ \bar{h} \ \bar{l}]$ over all objects of the given class.

$$\Delta_{dim}(x, z) = \left[\log \frac{w_i}{\bar{w}} \quad \log \frac{h_i}{\bar{h}} \quad \log \frac{l_i}{\bar{l}} \right]^\top \quad (8)$$

The second, the orientation head, predicts the sine and cosine of the objects orientation θ_i about the y-axis:

$$\Delta_{ang}(x, z) = \left[\sin \theta_i \quad \cos \theta_i \right]^\top \quad (9)$$

Note that since we are operating in the orthographic birds-eye-view space, we are able to predict the y-axis orientation θ directly, unlike other works e.g. [23] which predict the so-called *observation* angle α to take into account the effects of perspective and relative viewpoint. The position offset Δ_{pos} , dimension offset Δ_{dim} and orientation vector Δ_{ang} are trained using an ℓ_1 loss.

3.6. Non-maximum suppression

Similarly to other object detection algorithms, we apply a non-maximum suppression (NMS) stage to obtain a final discrete set of object predictions. In a conventional object detection setting this step can be expensive since it requires $\mathcal{O}(N^2)$ bounding box overlap computations. This is compounded by the fact that pairs of 3D boxes are not necessarily axis aligned, which makes the overlap computation more difficult compared to the 2D case. Fortunately, an additional benefit of the use of confidence maps in place of anchor box classification is that we can apply NMS in the more conventional image processing sense, i.e. searching for local maxima on the 2D confidence maps S . Here, the orthographic

birds-eye-view again proves invaluable: the fact that two objects cannot occupy the same volume in the 3D world means that peaks on the confidence maps are naturally separated.

To alleviate the effects of noise in the predictions, we first smooth the confidence maps by applying a Gaussian kernel with width σ_{NMS} . A location (x_i, z_i) on the smoothed confidence map \hat{S} is deemed to be a maximum if

$$\hat{S}(x_i, z_i) \geq \hat{S}(x_i + m, z_i + n) \quad \forall m, n \in \{-1, 0, 1\}. \quad (10)$$

Of the produced peak locations, any with a confidence $S(x_i, y_i)$ smaller than a given threshold t are eliminated. This results in the final set of predicted object instances, whose bounding box center p_i , dimensions d_i , and orientation θ_i , are given by inverting the relationships in Equations 7, 8 and 9 respectively.

4. Experiments

4.1. Experimental setup

Architecture For our front-end feature extractor we make use of a ResNet-18 network without bottleneck layers. We intentionally choose the front-end network to be relatively shallow, since we wish to put as much emphasis as possible on the 3D reasoning component of the model. We extract features immediately before the final three downsampling layers, resulting in a set of feature maps $\{f^s\}$ at scales s of 1/8, 1/16 and 1/32 of the original input resolution. Convolutional layers with 1×1 kernels are used to map these feature maps to a common feature size of 256, before processing them via the orthographic feature transform to yield orthographic feature maps $\{h^s\}$. We use a voxel grid with dimensions $80m \times 4m \times 80m$, which is sufficient to include all annotated instances in KITTI, and set the grid resolution r to be 0.5m. For the topdown network, we use a simple 16-layer ResNet without any downsampling or bottleneck units. The output heads each consist of a single 1×1 convolution layer. Throughout the model we replace all batch normalization [12] layers with group normalization [34] which has been found to perform better for training with small batch sizes.

Dataset We train and evaluate our method using the KITTI 3D object detection benchmark dataset [8]. For all experiments we follow the train-val split of Chen *et al.* [3] which divides the KITTI training set into 3712 training images and 3769 validation images.

Data augmentation Since our method relies on a fixed mapping from the image plane to the ground plane, we found that extensive data augmentation was essential for the network to learn robustly. We adopt three types of widely-used augmentations: random cropping, scaling and horizontal flipping, adjusting the camera calibration parameters f and (c_u, c_v) accordingly to reflect these perturbations.

Table 1. Average precision for birds-eye-view (AP_{BEV}) and 3D bounding box (AP_{3D}) detection on the KITTI test benchmark.

Method	Modality	AP_{3D}			AP_{BEV}		
		Easy	Moderate	Hard	Easy	Moderate	Hard
3D-SSMFCNN [24]	Mono	2.28	2.39	1.52	3.66	3.19	3.45
OFT-Net (Ours)	Mono	2.50	3.28	2.27	9.50	7.99	7.51

Table 2. Average precision for birds-eye-view (AP_{BEV}) and 3D bounding box (AP_{3D}) detection on the KITTI validation set.

Method	Modality	AP_{3D}			AP_{BEV}		
		Easy	Moderate	Hard	Easy	Moderate	Hard
3DOP [4]	Stereo	6.55	5.07	4.10	12.63	9.49	7.59
Mono3D [3]	Mono	2.53	2.31	2.31	5.22	5.19	4.13
OFT-Net (Ours)	Mono	4.07	3.27	3.29	11.06	8.79	8.91

Training procedure The model is trained using SGD for 600 epochs with a batch size of 8, momentum of 0.9 and learning rate of 10^{-7} . Following [21], losses are summed rather than averaged, which avoids biasing the gradients towards examples with few object instances. The loss functions from the various output heads are combined using a simple equal weighting strategy.

4.2. Comparison to state-of-the-art

We evaluate our approach on two tasks from the KITTI 3D object detection benchmark. The 3D bounding box detection task requires that each predicted 3D bounding box should intersect a corresponding ground truth box by at least 70% in the case of cars and 50% for pedestrians and cyclists. The birds-eye-view detection task meanwhile is slightly more lenient, requiring the same amount of overlap between a 2D birds-eye-view projection of the predicted and ground truth bounding boxes on the ground plane. At the time of writing, the KITTI benchmark included only one published approach operating on monocular RGB images alone ([24]), which we compare our method against in Table 1. We therefore perform additional evaluation on the KITTI validation split set out by Chen *et al.* (2016) [3]; the results of which are presented in Table 2. For monocular methods, performance on the pedestrian and cyclist classes is typically insufficient to obtain meaningful results and we therefore follow other works [3, 4, 24] and focus our evaluation on the car class only.

It can be seen from Tables 1 and 2 that our method is able to outperform all comparable (i.e. monocular only) methods by a considerable margin across both tasks and all difficulty criteria. The improvement is particularly marked on the hard evaluation category, which includes instances which are heavily occluded, truncated or far from the camera. We also show in Table 2 that our method performs competitively with the stereo approach of Chen *et al.* (2015) [4], achieving close to or in one case better performance than

their 3DOP system. This is in spite of the fact that unlike [4], our method does not have access to any explicit knowledge of the depth of the scene.

4.3. Qualitative results

Comparison to Mono3D We provide a qualitative comparison of predictions generated by our approach and Mono3D [3] in Figure 4. A notable observation is that our system is able to reliably detect objects at a considerable distance from the camera. This is a common failure case among both 2D and 3D object detectors, and indeed many of the cases which are correctly identified by our system are overlooked by Mono3D. We argue that this ability to recognise objects at distance is a major strength of our system, and we explore this capacity further in Section 5.1. Further qualitative results are included in supplementary material.

Ground plane confidence maps A unique feature of our approach is that we operate largely in the orthographic birds-eye-view feature space. To illustrate this, Figure 5 shows examples of predicted confidence maps $S(x, z)$ both in the topdown view and projected into the image on the ground plane. It can be seen that the predicted confidence maps are well localized around each object center.

4.4. Ablation study

A central claim of our approach is that reasoning in the orthographic birds-eye-view space significantly improves performance. To validate this claim, we perform an ablation study where we progressively remove layers from the topdown network. In the extreme case, when the depth of the topdown network is zero, the architecture is effectively reduced to RoI pooling [9] over projected bounding boxes, rendering it similar to R-CNN-based architectures. Figure 7 shows a plot of average precision against the total number of parameters for two different architectures.

The trend is clear: removing layers from the topdown network significantly reduces performance. Some of this

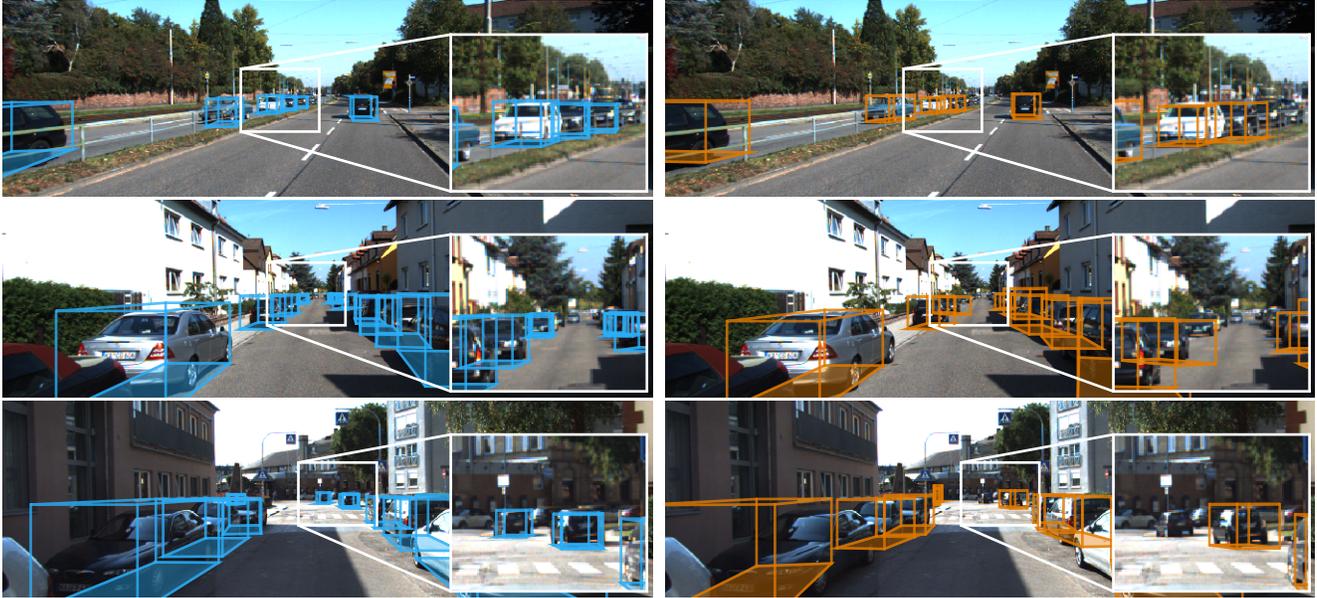


Figure 4. Qualitative comparison between our method (left) and Mono3D [3] (right) on the KITTI validation set. Inset regions highlight the behaviours of the two systems at large distances. We are able to consistently detect distant objects which are beyond the range of Mono3D.



Figure 5. Examples of confidence maps generated by our approach, which we visualize both in birds-eye-view (right) and projected onto the ground plane in the image view (left). We use the pre-computed ground planes of [4] to obtain the road position: note that this is for visualization purposes only and the ground planes are not used elsewhere in our approach. Best viewed in color.

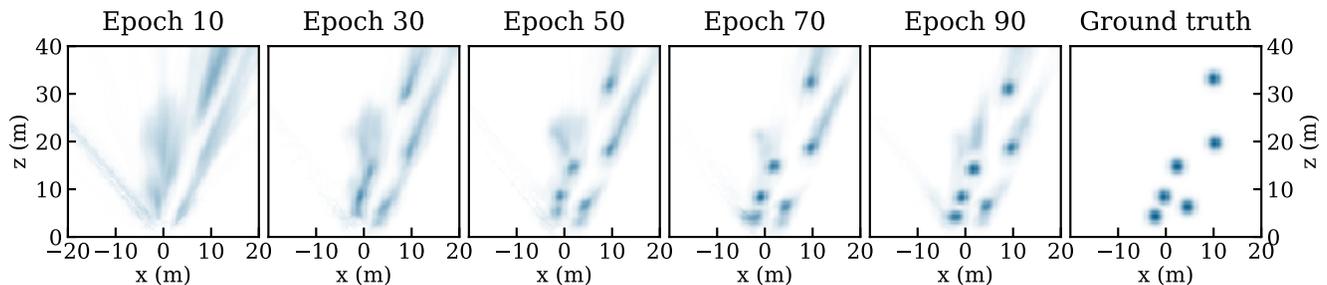


Figure 6. Evolution of ground plane confidence maps during training. The network initially exhibits high uncertainty in the depth direction but gradually resolves this uncertainty as training progresses.

decline in performance may be explained by the fact that reducing the size of the topdown network reduces the overall depth of the network, and therefore its representational power. However, as can be seen from Figure 7, adopting a shallow front-end (ResNet-18) with a large topdown network achieves significantly better performance than a deeper network (ResNet-34) without any topdown layers, despite the two architectures having roughly the same number of parameters. This strongly suggests that a significant part of the success of our architecture comes from its ability to reason in 3D, as afforded by the 2D convolution layers operating on the orthographic feature maps.

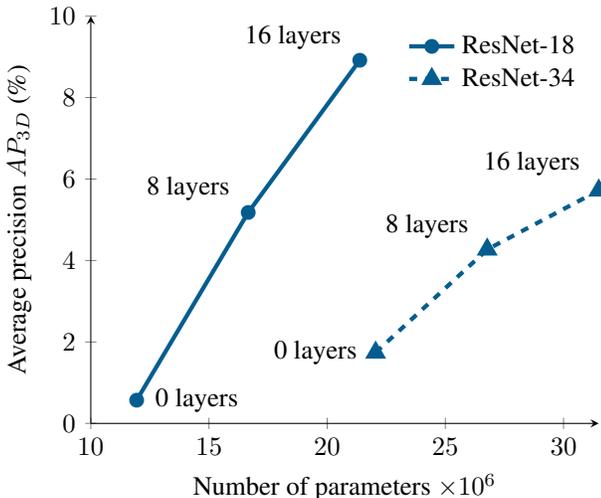


Figure 7. Ablation study showing the effect of reducing the number of layers in the topdown network on performance for two different frontend architectures. Zero layers implies that topdown network has been removed entirely.

5. Discussion

5.1. Performance as a function of depth

Motivated by the qualitative results in Section 4.2, we wished to further quantify the ability of our system to detect and localize distant objects. Figure 8 plots performance of each system when evaluated only on objects which are at least the given distance away from the camera. Whilst we outperform Mono3D over all depths, it is also apparent that the performance of our system degrades much more slowly as we consider objects further from the camera. We believe that this is a key strength of our approach.

5.2. Evolution of confidence maps during training

While the confidence maps predicted by our network are not necessarily calibrated estimates of model certainty, observing their evolution over the course of training does give valuable insights into the learned representation. Figure 6 shows an example of a confidence map predicted by the network at various points during training. During the early

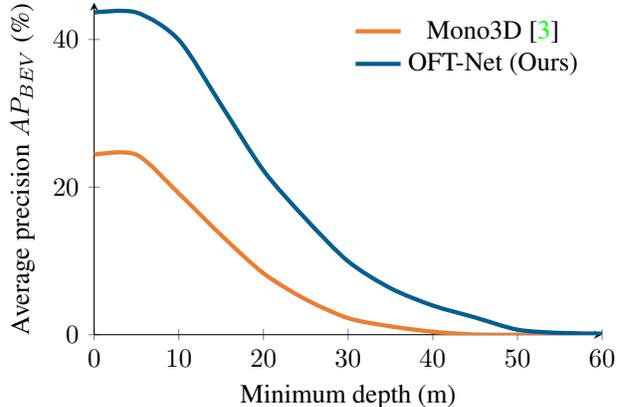


Figure 8. Average BEV precision (val) as a function of the minimum distance of objects from the camera. We use an IoU threshold of 0.5 to better compare performance at large depths.

stages of training, the network very quickly learns to identify regions of the image which contain objects, which can be seen by the fact that high confidence regions correspond to projection lines from the optical center at $(0, 0)$ which intersect a ground truth object. However, there exists significant uncertainty about the depth of each object, leading to the predicted confidences being blurred out in the depth direction. This fits well with our intuition that for a monocular system depth estimation is significantly more challenging than recognition. As training progresses, the network is increasingly able to resolve the depth of the objects, producing sharper confidence regions clustered about the ground truth centers. It can be observed that even in the latter stages of training, there is considerably greater uncertainty in the depth of distant objects than that of nearby ones, evoking the well-known result from stereo that depth estimation error increases quadratically with distance.

6. Conclusions

In this work we have presented a novel approach to monocular 3D object detection, based on the intuition that operating in the birds-eye-view domain alleviates many undesirable properties of images which make it difficult to infer the 3D configuration of the world. We have proposed a simple *orthographic feature transform* which transforms image-based features into this birds-eye-view representation, and described how to implement it efficiently using integral images. This was then incorporated into part of a deep learning pipeline, in which we particularly emphasized the importance of spatial reasoning in the form of a deep 2D convolutional network applied to the extracted birds-eye-view features. Finally, we experimentally validated our hypothesis that reasoning in the topdown space does achieve significantly better results, and demonstrated state-of-the-art performance on the KITTI 3D object benchmark.

References

- [1] J. Beltran, C. Guindel, F. M. Moreno, D. Cruzado, F. Garcia, and A. de la Escalera. BirdNet: a 3D object detection framework from LiDAR information. *arXiv preprint arXiv:1805.01195*, 2018. 1, 2
- [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. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2040–2049, 2017. 2
- [3] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun. Monocular 3D object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2147–2156, 2016. 2, 5, 6, 7, 8
- [4] X. Chen, K. Kundu, Y. Zhu, A. G. Berneshawi, H. Ma, S. Fidler, and R. Urtasun. 3D object proposals for accurate object class detection. In *Advances in Neural Information Processing Systems*, pages 424–432, 2015. 2, 6, 7
- [5] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia. Multi-view 3D object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, page 3, 2017. 1, 2
- [6] X. Du, M. H. Ang Jr, S. Karaman, and D. Rus. A general pipeline for 3D detection of vehicles. *arXiv preprint arXiv:1803.00387*, 2018. 1, 2
- [7] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The Pascal Visual Object Classes (VOC) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010. 2
- [8] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012. 1, 5
- [9] R. Girshick. Fast R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015. 6
- [10] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 2
- [11] L. Huang, Y. Yang, Y. Deng, and Y. Yu. Densebox: Unifying landmark localization with end to end object detection. *arXiv preprint arXiv:1509.04874*, 2015. 4
- [12] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of The 32nd International Conference on Machine Learning*, pages 448–456, 2015. 5
- [13] A. Kasagi, T. Tabaru, and H. Tamura. Fast algorithm using summed area tables with unified layer performing convolution and average pooling. In *Machine Learning for Signal Processing (MLSP), IEEE 27th International Workshop on*, pages 1–6, 2017. 2
- [14] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab. SSD-6D: Making rgb-based 3d detection and 6d pose estimation great again. In *Proceedings of the International Conference on Computer Vision*, pages 22–29, 2017. 2
- [15] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. Waslander. Joint 3D proposal generation and object detection from view aggregation. *arXiv preprint arXiv:1712.02294*, 2017. 1, 2
- [16] B. Li, T. Zhang, and T. Xia. Vehicle detection from 3D lidar using fully convolutional network. *arXiv preprint arXiv:1608.07916*, 2016. 2
- [17] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, page 3, 2017. 2
- [18] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018. 2, 4
- [19] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer, 2014. 2
- [20] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single shot multi-box detector. In *European Conference on Computer Vision*, pages 21–37. Springer, 2016. 2
- [21] D. Masters and C. Luschi. Revisiting small batch training for deep neural networks. *arXiv preprint arXiv:1804.07612*, 2018. 6
- [22] K. Minemura, H. Liau, A. Monroy, and S. Kato. Lmnet: Real-time multiclass object detection on CPU using 3D LiDARs. *arXiv preprint arXiv:1805.04902*, 2018. 1, 2
- [23] A. Mousavian, D. Anguelov, J. Flynn, and J. Košecká. 3D bounding box estimation using deep learning and geometry. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5632–5640. IEEE, 2017. 2, 5

- [24] L. Novak. *Vehicle detection and pose estimation for autonomous driving*. PhD thesis, Masters thesis, Czech Technical University in Prague, 2017. Cited on, 2017. [2](#), [6](#)
- [25] C. C. Pham and J. W. Jeon. Robust object proposals re-ranking for object detection in autonomous driving using convolutional neural networks. *Signal Processing: Image Communication*, 53:110–122, 2017. [2](#)
- [26] P. Poirson, P. Ammirato, C.-Y. Fu, W. Liu, J. Kosecka, and A. C. Berg. Fast single shot detection and pose estimation. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 676–684. IEEE, 2016. [2](#)
- [27] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas. Frustum pointnets for 3D object detection from RGB-D data. *arXiv preprint arXiv:1711.08488*, 2017. [1](#), [2](#)
- [28] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. [2](#)
- [29] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2015. [2](#), [4](#)
- [30] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. [2](#)
- [31] A. Shrivastava, A. Gupta, and R. Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 761–769, 2016. [4](#)
- [32] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages I–I. IEEE, 2001. [2](#), [4](#)
- [33] S. Wirges, T. Fischer, J. B. Frias, and C. Stiller. Object detection and classification in occupancy grid maps using deep convolutional networks. *arXiv preprint arXiv:1805.08689*, 2018. [1](#), [2](#)
- [34] Y. Wu and K. He. Group normalization. *European Conference on Computer Vision*, 2018. [5](#)
- [35] Y. Xiang, W. Choi, Y. Lin, and S. Savarese. Data-driven 3D voxel patterns for object category recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1903–1911, 2015. [2](#)
- [36] Y. Xiang, W. Choi, Y. Lin, and S. Savarese. Subcategory-aware convolutional neural networks for object proposals and detection. In *Applications of Computer Vision (WACV), IEEE Winter Conference on*, pages 924–933. IEEE, 2017. [2](#)
- [37] S.-L. Yu, T. Westfechtel, R. Hamada, K. Ohno, and S. Tadokoro. Vehicle detection and localization on birds eye view elevation images using convolutional neural network. In *IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, volume 5, 2017. [1](#), [2](#)
- [38] M. Zeeshan Zia, M. Stark, and K. Schindler. Are cars just 3D boxes? Jointly estimating the 3D shape of multiple objects. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3678–3685, 2014. [2](#)