Orthographic Feature Transform for Monocular 3D Object Detection

Thomas Roddick
tr346@cam.ac.uk
Alex Kendall
agk34@cam.ac.uk
Roberto Cipolla
rc10001@cam.ac.uk

Department of Engineering,
University of Cambridge,
United Kingdom

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 maps image-based features into an orthographic 3D space, enabling us to reason holistically about the spatial configuration of the scene. We apply this transformation as part of an end-to-end deep learning architecture and demonstrate our approach 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. Vision-only systems offer several advantages over more widespread LiDAR-based methods [1, 5, 6, 14, 21, 26, 31, 36], such as lower cost, higher resolution and robustness to adverse weather conditions. However the lack of absolute depth information means that to date the performance of vision-based systems still lags significantly behind.

In this work we aim to reduce this performance gap by emulating the orthographic birds-eye-view representation popular in many LiDAR-based approaches [1, 31, 36]. This representation is attractive since it is metric, making it possible to reason meaningfully about distances in the 3D space. In order to extract this representation from the perspective 2D image, we introduce the Orthographic Feature Transform (OFT), a differentiable neural network component which uses a combination of geometric and learned reasoning to infer the

[1] Please see www.youtube.com/watch?v=ZpV51Fgc4jE for a video demonstration of our approach.

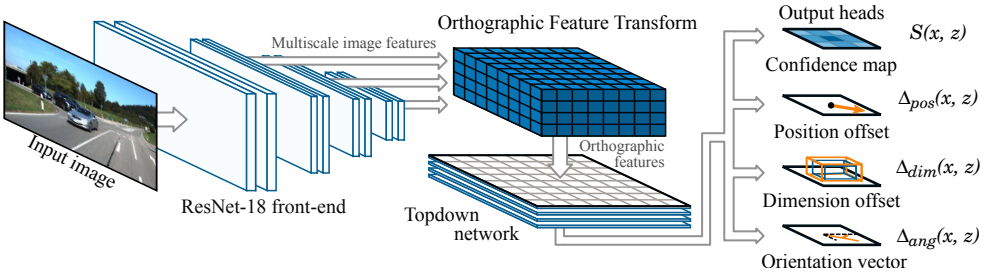

Figure 1: Architecture overview. A front-end ResNet feature extractor generates image-based features, which are mapped to an orthographic representation via our proposed orthographic feature transform. The topdown network processes these features in the birds-eye-view space and at each location on the ground plane predicts a confidence score S, a position offset Δ_{pos}, a dimension offset Δ_{dim} and an angle vector Δ_{ang}.

locations of features on the ground plane. Crucially, we do not rely on any explicit notion of depth, but leave the network free to generate its own implicit representation. The OFT is incorporated into a deep learning architecture for predicting 3D object bounding boxes from monocular images. A central theme of this work is to emphasise the importance of reasoning in the 3D space, which we demonstrate is a key strength of our approach.

2 Related Work

2D object detection Detecting 2D bounding boxes in images is a widely studied problem. Existing methods may broadly be divided into two main categories: *single stage* detectors such as YOLO [27], SSD [19] and RetinaNet [18] which predict object bounding boxes directly and *two-stage* detectors such as Faster R-CNN [28] and FPN [17] which add an intermediate region proposal stage. Our work may be viewed as one of the first to generalize the single stage paradigm to 3D object detection, as most 3D existing approaches adopt proposal-based methods, in part due to the difficulty in mapping from fixed-size regions in 3D to variable-sized regions in the image. This allows 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 [6, 16, 21, 26]. Among these approaches, a common technique is to encode 3D point cloud information as a birds-eye-view feature map [1, 5, 14, 31, 36]. This representation is attractive since it encodes a metric representation of the world which allows networks to exploit strong priors about the shape and size of 3D objects. This provides the inspiration for our orthographic feature transform. Sensor fusion approaches such as AVOD [14] and MV3D [5] rely on similar transformations to match image features to 3D proposals, but these methods presuppose 3D knowledge about the scene whilst our approach infers it implicitly.

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 re-

gion [13, 22, 23, 25] or fit 3D templates to the image [2, 33, 34, 37]. Perhaps most closely related to our work is Mono3D [4] 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 [3] and Pham and Jeon [24], 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 [4], but applies a secondary convolutional network to the resulting proposals whilst retaining their spatial configuration.

A number of recent works have vastly improved the state of the art in monocular object detection by taking advantage of additional components such as pretrained monocular depth estimation [30, 35], 3D shape supervision [15] or synthetic data augmentation [9]. Our method is compatible with these techniques and we believe that further improvements could be achieved by incorporating the merits of each approach.

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 1. The algorithm comprises five main components, which are described in detail in the remainder of this section:

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.
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 confidence scores and position, dimension and orientation offsets for each location on the ground plane.
5. A non-maximum suppression and decoding stage, which identifies peaks in the confidence maps and generates discrete bounding box predictions.

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).

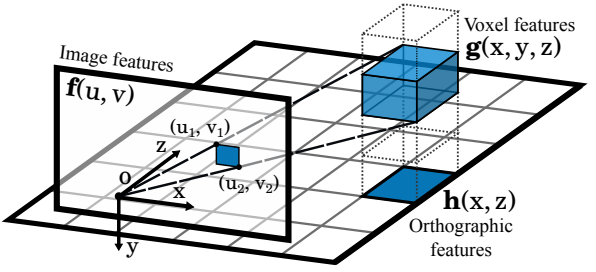

Figure 2: Orthographic Feature Transform (OFT). Voxel-based features $\mathbf{g}(x,y,z)$ are generated by accumulating image-based features $\mathbf{f}(u,v)$ over the projected voxel area. Voxel features are then collapsed along the height dimension to yield orthographic features $\mathbf{h}(x,z)$.

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. 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) given by

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

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, 2D 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 typical voxel grid setting generates around 150k bounding boxes, which far exceeds the ∼2k regions of interest used by the Faster R-CNN [28] 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 [29]. An integral feature map, $\mathbf{F}$, is constructed from an input feature map $\mathbf{f}$ using the recursive relation

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

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

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

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 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. 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]. Given a set of N ground truth objects with bounding box centres $p_i = \begin{bmatrix} x_i & y_i & z_i \end{bmatrix}^\top, i = 1,\ldots,N$, we compute the ground truth confidence map S as a smooth Gaussian region of width σ around the center of each object:

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

The confidence map prediction head is trained via an ℓ_1 loss to regress the ground truth confidence for each location on the orthographic grid $\mathcal{H}$. We downweight the loss associated with low-confidence locations ($S(x,z) < 0.05$) by a factor 10^{-2} to avoid it dominating training.

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:

$$\Delta_{pos}(x,z) = \begin{bmatrix} \frac{x_i - x}{\sigma} & \frac{y_i - y_0}{\sigma} & \frac{z_i - z}{\sigma} \end{bmatrix}^\top \tag{7}$$

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.

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 $d_i = \begin{bmatrix} w_i & h_i & l_i \end{bmatrix}$ and the mean dimensions $\bar{d} = \begin{bmatrix} \bar{w} & \bar{h} & \bar{l} \end{bmatrix}$ over all objects of the given class (Equation 8). The second, the orientation head, predicts the sine and cosine of the objects orientation θ_i about the y-axis (Equation 9).

$$\Delta_{dim}(x,z) = \begin{bmatrix} \log \frac{w_i}{\bar{w}} & \log \frac{h_i}{\bar{h}} & \log \frac{l_i}{\bar{l}} \end{bmatrix}^\top \tag{8}$$

$$\Delta_{ang}(x,z) = \begin{bmatrix} \sin\theta_i & \cos\theta_i \end{bmatrix}^\top \tag{9}$$

The position offset Δ_{pos}, dimension offset Δ_{dim} and orientation vector Δ_{ang} for each cell are trained using an ℓ_1 loss, ignoring cells which do not correspond to any instance.

3.6 Non-maximum suppression

As with other object detection algorithms, we apply a non-maximum suppression (NMS) stage to obtain a final discrete set of object predictions. A major advantage of the orthographic representation is that we can apply NMS in the conventional image processing sense *i.e.* searching for local maxima on the 2D confidence maps S, since object centers are naturally separated in the 3D space. This avoids an expensive $\mathcal{O}(N^2)$ comparison between pairs of non-axis-aligned 3D bounding boxes. We apply a Gaussian smoothing function of width σ_{NMS} to alleviate noise and retain all peaks with a confidence greater that some threshold t.

4 Experiments

Architecture For our front-end feature extractor we make use of a shallow ResNet-18 network. We extract features immediately before the final three downsampling layers, resulting in a set of feature maps $\{\mathbf{f}^s\}$ 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 OFT. We use a voxel grid with dimensions 80m×4m×80m and resolution r of 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. We replace all batch normalization [12] layers with group normalization [32].

Dataset We train and evaluate our method using the KITTI 3D object detection benchmark dataset [7]. For all experiments we follow the train-val split of Chen *et al.* [4] which divides the KITTI training set into 3712 training images and 3769 validation images.

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 [20], 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 equal weights.

4.1 Evaluation on KITTI benchmark

We evaluate our approach on two tasks from the KITTI 3D object detection benchmark. The 3D car bounding box detection task requires that each predicted 3D bounding box should intersect a corresponding ground truth box by at least 70%. The birds-eye-view detection task meanwhile is slightly more lenient, as it ignores the vertical component of the bounding box overlap. We report the results of our approach both on the official KITTI test benchmark (Table 1) and the validation split of Chen *et al.* [3] (Table 2). Since this work was originally submitted to the benchmark, a number of other approaches have been proposed which achieve better performance [9, 15, 35]. We include these results in the tables below, however we note that these methods take advantage of additional training supervision such as explicit depth prediction [35], shape estimation [15] or synthetic training examples [9], and therefore suggest that our method represents a complementary and novel approach which offers further insights into the monocular problem.

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

Method	Modality	AP_{3D}			AP_{BEV}		
		Easy	Moderate	Hard	Easy	Moderate	Hard
MultiFusion [35]	Mono+Depth	7.08	5.18	4.68	13.73	9.62	8.22
A3DODWTDA [9]	Mono+Synth	6.76	6.45	4.87	10.21	10.61	8.64
Ku *et al.* [15]	Mono+Shape	12.57	10.85	9.06	20.25	17.66	15.78
3D-SSMFCNN [23]	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 [3]	Stereo	6.55	5.07	4.10	12.63	9.49	7.59
A3DODWTDA [9]	Mono+Synthetic	10.13	8.32	8.20	15.64	12.90	12.30
MultiFusion [35]	Mono+Depth	10.53	5.69	5.39	22.03	13.63	11.60
Ku *et al.* [15]	Mono+Shape	12.75	11.48	8.59	20.63	18.67	14.45
Mono3D [4]	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

It can be seen from the results above that our method is able to outperform comparable (i.e. monocular only) methods. It is particularly successful on the hard evaluation category, which includes instances which are heavily occluded or distant from the camera. Where most methods exhibit a large performance drop between easy and hard, ours remains relatively stable. We also show that our method performs competitively with the stereo approach of

[3], despite not having access to any explicit knowledge of the depth of the scene. One limiting factor of our approach is the resolution of the voxel grid, currently set at $0.5m$ due to memory requirements. Whilst we have yet to perform a thorough analysis, preliminary experiments suggest that reducing the voxel dimensions would allow us to achieve further performance gains.

4.2 Qualitative results

Comparison to Mono3D We provide a qualitative comparison of predictions generated by our approach and Mono3D [4] in Figure 3. 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 4.4.

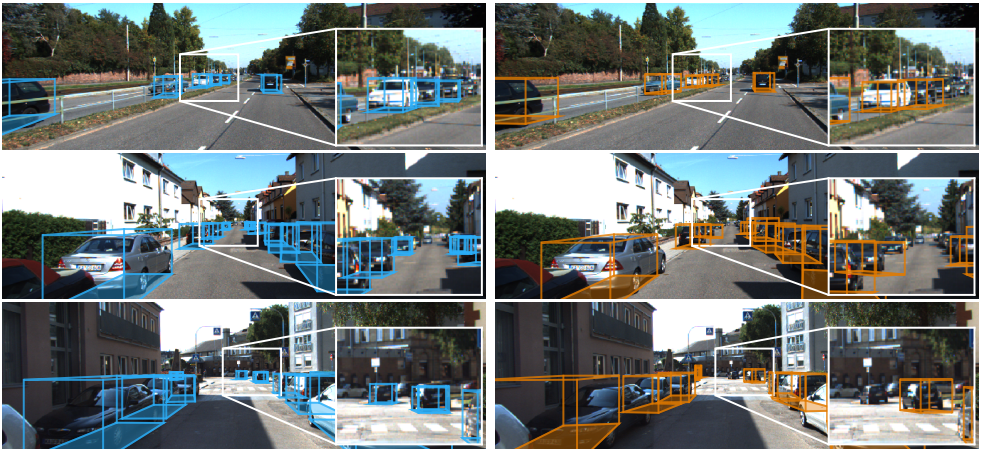

Figure 3: Qualitative comparison between our method (left) and Mono3D [4] (right) on the KITTI validation set. Inset regions highlight the behaviours of both systems at large distances. We consistently detect distant objects which are beyond the range of Mono3D.

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 4 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.

4.3 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 (shown in Figure 5). In the extreme case, when the depth of the topdown network is zero, the architecture is effectively reduced to RoI pooling [8], rendering it similar to R-CNN-based architectures.

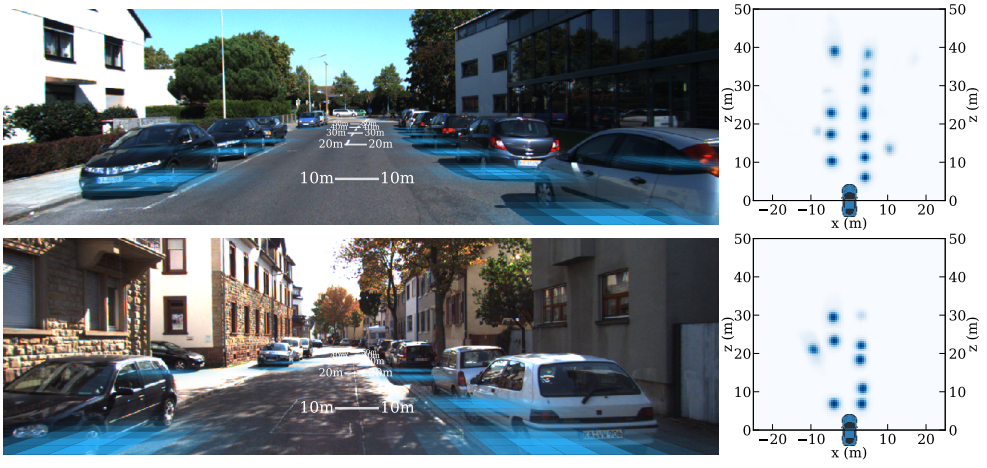

Figure 4: Examples of confidence maps generated by our approach, which we visualize both in birds-eye-view (right) and projected onto the ground plane in the image view (left).

The trend is clear: removing layers from the topdown network significantly reduces performance. Some of this 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 5, 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.

4.4 Performance as a function of depth

Motivated by the qualitative results in Section 4.1, we wished to further quantify the ability of our system to detect and localize distant objects. Figure 6 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.

4.5 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 7 shows an example of a confidence map predicted by the network at various points during training. During the early stages of training, the network quickly learns to identify regions of the image which contain objects but is unable to accurately localize them in the depth dimension, leading to blurred-out predictions extending radially from the camera. As training progresses this uncertainty is reduced, however it can be seen that for objects far away from the camera some blurring remains, strongly corresponding to the intuition that objects further away are harder to localize.

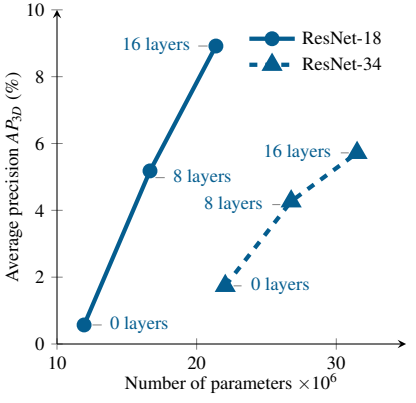

Figure 5: Ablation study showing the effect of removing layers from the topdown network. Zero layers implies that the topdown network has been removed entirely. A network with shallower front-end (ResNet-18) but more topdown layers outperforms a network with a deeper front-end and shallower topdown network, despite having roughly similar numbers of parameters.

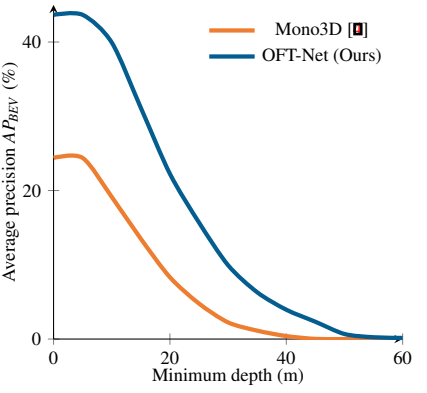

Figure 6: Average BEV precision (val) as a function of the minimum distance of ground truth objects from the camera. We use an IoU threshold of 0.5 to better compare performance at large depths. Our method significantly outperforms Mono3D across all depth ranges but is particularly effective for objects at extreme distances.

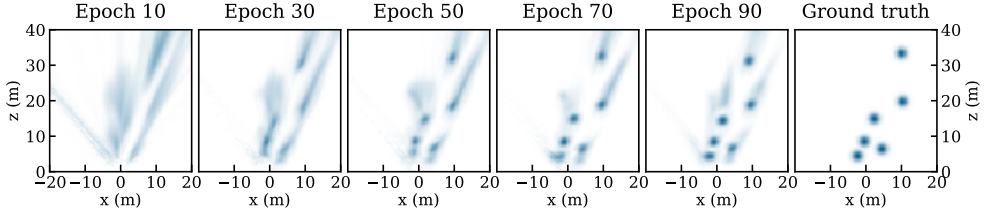

Figure 7: Evolution of confidence maps during training. The network initially exhibits high uncertainty in the depth direction but gradually resolves this as training progresses.

5 Conclusions

In this work we have presented a novel approach to monocular 3D object detection which takes advantage of an orthographic birds-eye-view to accurately estimate object locations in 3D. We are motivated by the intuition that operating in the birds-eye-view enables the network to reason holistically about the 3D relationships between objects and factors out some of the distortional effects of perspective projection. We obtained a mapping from image-space to birds-eye-view via our proposed *orthographic feature transform*, which was implemented efficiently using integral images. By evaluating our deep-learning-based approach on the KITTI detection benchmark we were able to experimentally validate our hypothesis that reasoning in the 3D space improves performance and that our network is robust to objects which are distant or occluded. This work forms the basis for future exploration into other tasks where the birds-eye-view representation is naturally applicable such as 3D object tracking and motion forecasting.

References

[1] Jorge Beltrán, Carlos Guindel, Francisco Miguel Moreno, Daniel Cruzado, Fernando Garcia, and Arturo de la Escalera. Birdnet: A 3d object detection framework from lidar information. 11 2018.

[2] Florian Chabot, Mohamed Chaouch, Jaonary Rabarisoa, Céline Teulière, and Thierry Chateau. Deep MANTA: A coarse-to-fine many-task network for joint 2D and 3D vehicle analysis from monocular image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2040–2049, 2017.

[3] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Andrew G Berneshawi, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3D object proposals for accurate object class detection. In *Advances in Neural Information Processing Systems*, pages 424–432, 2015.

[4] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. Monocular 3D object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2147–2156, 2016.

[5] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian 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.

[6] Xinxin Du, Marcelo H. Ang, Sertac Karaman, and Daniela Rus. A general pipeline for 3d detection of vehicles. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[7] Andreas Geiger, Philip Lenz, and Raquel 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.

[8] Ross Girshick. Fast R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015.

[9] Fredrik Gustafsson and Erik Linder-Norén. Automotive 3d object detection without target domain annotations. Master's thesis, 2018.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[11] Lichao Huang, Yi Yang, Yafeng Deng, and Yinan Yu. Densebox: Unifying landmark localization with end to end object detection. *arXiv preprint arXiv:1509.04874*, 2015.

[12] Sergey Ioffe and Christian 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.

[13] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir 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.

[14] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven Waslander. Joint 3D proposal generation and object detection from view aggregation. *arXiv preprint arXiv:1712.02294*, 2017.

[15] Jason Ku, Alex D Pon, and Steven L Waslander. Monocular 3d object detection leveraging accurate proposals and shape reconstruction. *arXiv preprint arXiv:1904.01690*, 2019.

[16] Bo Li, Tianlei Zhang, and Tian Xia. Vehicle detection from 3D lidar using fully convolutional network. *arXiv preprint arXiv:1608.07916*, 2016.

[17] Tsung-Yi Lin, Piotr Dollár, Ross B Girshick, Kaiming He, Bharath Hariharan, and Serge 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.

[18] Tsung-Yi Lin, Priyal Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[19] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single shot multibox detector. In *European Conference on Computer Vision*, pages 21–37. Springer, 2016.

[20] Dominic Masters and Carlo Luschi. Revisiting small batch training for deep neural networks. *arXiv preprint arXiv:1804.07612*, 2018.

[21] Kazuki Minemura, Hengfui Liau, Abraham Monrroy, and Shinpei Kato. LMNet: Real-time multiclass object detection on CPU using 3D LiDARs. *arXiv preprint arXiv:1805.04902*, 2018.

[22] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana 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.

[23] Libor Novak. Vehicle detection and pose estimation for autonomous driving. Master's thesis, Czech Technical University in Prague, 2017.

[24] Cuong Cao Pham and Jae Wook Jeon. Robust object proposals re-ranking for object detection in autonomous driving using convolutional neural networks. *Signal Processing: Image Communication*, 53:110–122, 2017.

[25] Patrick Poirson, Phil Ammirato, Cheng-Yang Fu, Wei Liu, Jana Kosecka, and Alexander C Berg. Fast single shot detection and pose estimation. In *3D Vision (3DV), Fourth International Conference on*, pages 676–684. IEEE, 2016.

[26] Charles Ruizhongtai Qi, Weiwei Liu, Chenxia Wu, Hao Su, and Leonidas J. Guibas. Frustum pointnets for 3d object detection from rgb-d data. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017.

[27] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[28] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2015.

[29] Paul Viola and Michael 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.

[30] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Weinberger. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *CVPR*, 2019.

[31] Sascha Wirges, Tom Fischer, Jesus Balado Frias, and Christoph Stiller. Object detection and classification in occupancy grid maps using deep convolutional networks. *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 3530–3535, 2018.

[32] Yuxin Wu and Kaiming He. Group normalization. *European Conference on Computer Vision*, 2018.

[33] Yu Xiang, Wongun Choi, Yuanqing Lin, and Silvio 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.

[34] Yu Xiang, Wongun Choi, Yuanqing Lin, and Silvio 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.

[35] Bin Xu and Zhenzhong Chen. Multi-level fusion based 3d object detection from monocular images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2345–2353, 2018.

[36] Shang-Lin Yu, Thomas Westfechtel, Ryunosuke Hamada, Kazunori Ohno, and Satoshi 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.

[37] Muhammad Zeeshan Zia, Michael Stark, and Konrad 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.