12. Workshop Fahrerassistenzsysteme und automatisiertes Fahren. FAS2018
Waiting im Altmühltal, 26.09.-28.09.2018
Uni-DAS e.V., 138-146, 2018.

Fusion of Camera and Lidar Data for Object Detection using Neural Networks

Enrico Schröder* Mirko Mählisch* Julien Vitay † Fred Hamker †

Zusammenfassung: We present a novel architecture for intermediate fusion of Lidar and camera data for neural network-based object detection. Key component is a transformer module which learns a transformation of feature maps from one sensor space to another. This allows large parts of the multi-modal object detection network to be trained unimodally, reducing the required amount of costly multi-modal labeled data. We show effectiveness of the transformer as well as the proposed fusion scheme.

Schlüsselwörter: Sensor Fusion, Object Detection, Convolutional Neural Networks, Lidar

1 Introduction

One of the most important tasks in automotive perception is detecting objects and obstacles such as cars or pedestrians in the vehicle's surroundings. For reasons of performance and robustness, multiple sensor modalities such as camera, Lidar and Radar sensors are used to exploit individual strenghts of each sensor type. Traditionally, discrete objects detected by each sensor are fused via some model-based Bayesian fusion framework. Recently, methods based on convolutional neural networks have been used to detect objects in raw images (e.g. [7]) and also Lidar pointclouds ([11], [8]), outperforming methods based on traditional handcrafted features. This makes it viable to explore methods which fuse sensor modalities at a lower level of abstraction directly within these object detection networks to potentially make use of additional information that has not yet been abstracted away by an underlying model. We present a general framework for fusing object detection networks for multiple sensor modalities at an intermediate stage using perspective-invariant features.

2 Related work

In contrast to traditional image classification networks, object detection networks predict position and class of *multiple objects* in the frame. Most object detection networks first feed input images through a convolutional *feature extractor* network such as ResNet [3] or VGG [9]. The resulting feature maps are then passed through a combination of convolutional

*AUDI AG, Development Sensor Fusion & Localization for Automated Driving (e-mail: {enrico.schroeder,mirko.maehlisch}@audi.de)

†TU Chemnitz, Dept. of Computer Sciences (e-mail: {julien.vitay,fred.hamker}@informatik.tu-chemnitz.de)

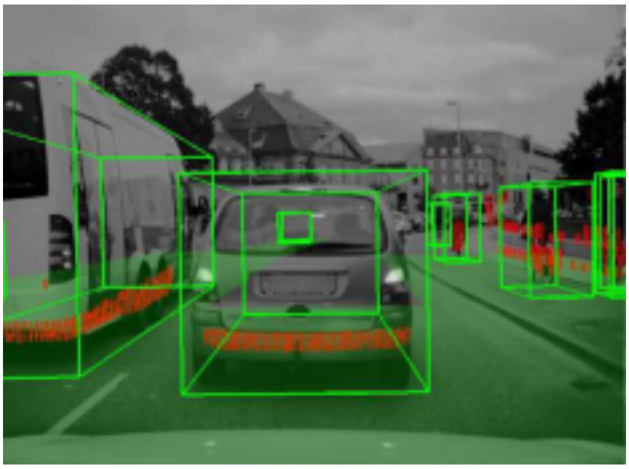


Abbildung 1: A sample frame of our labeled dataset, with Lidar data and 3D groundtruth projected into image space.

and fully-connected layers (the *detector*) to generate object bounding boxes and classes. Work on multi-modal object detection in an automative context has gained traction since introduction of the 2D birdview and 3D object detection challenges based on the KITTI dataset [2]. KITTI provides front camera images as well as Velodyne Lidar point cloud and 3D groundtruth objects. A multitude of approaches based on this dataset have been proposed. Most either use camera detections to create regions of interest in the Lidar pointcloud for final object prediction (such as [6]) or use anchor regions in 3D space and project them into camera and Lidar views to get region proposals for the different modalities. These proposals are then cropped and jointly fed to detector stages to produce final object predictions (MV3D [1] or AVOD [5]).

While many of the available approaches show good performance, they do not focus on the issues which are prevalent in series car development. Most methods train an entire monolithic multi-modal object detection network in one single step, requiring large labeled datasets containing all modalities at once. The process of collecting this data is hugely laborious and costly (and thus unfeasible in series car production, where employed sensor modalities and models change frequently). A modular approach would be desired, where perception for each modality (or sensor respectively) can be trained individually and then only be fine-tuned by a much smaller multi-modal dataset to train specific fusion layers. Additionally, having sensors perform some perception individually would allow for the entire system behavior to be much more traceable since sensors could still be analyzed separately. This is an important aspect aiding functional safety. Our approach fuses information at a single point in the object detection network by making feature map activations from different sensors compatible in locality and properties.

3 Fusion architecture

Our proposed approach focuses on fusing pre-trained object detection networks for single sensor modalities, requiring a significantly smaller amount of multi-modal labeled data compared to end-to-end object detection networks. The architecture respects that most relevant sensors produce depth information (with the exception of monocular cameras)

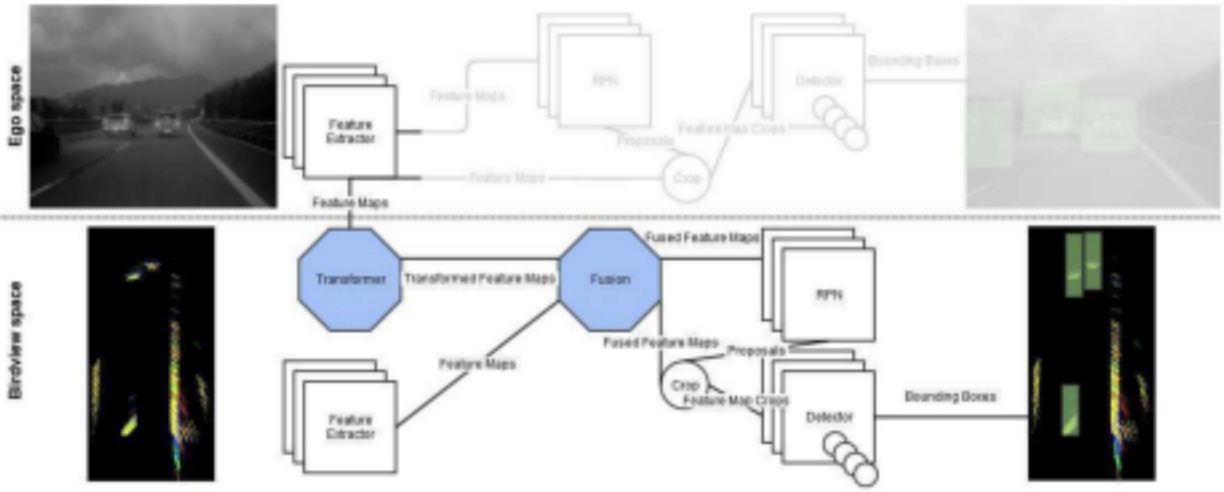


Abbildung 2: Our proposed fusion architecture. White elements denote parts of the network belonging to an arbitrary state-of-the-art object detector (Faster-RCNN [7] in our case). Blue elements denote our transformer and fusion modules mapping feature maps from ego- to birdview. This allows using camera features in addition to Lidar for predicting objects' locations and classes. The camera-only detector is used only for training the feature extractor and can be removed afterwards.

and can be processed efficiently in a 2D birdview representation. Thus, our framework mainly operates in 2D birdview and outputs 2D object bounding boxes. By projecting 3D Lidar pointclouds into 2D birdview we can leverage robust state-of-the-art 2D object detection networks for all relevant sensors.

Our low-level architecture is shown in figure 2. We employ a common two-stage object detection network (Faster-RCNN [7]) which extracts features from the input via a feature extraction network in the first stage, creates region proposals from the extracted features and then classifies and refines these region proposals in the second stage. Our proposed fusion architecture does not have any specific requirements for the object detection network and would thus work with networks other than Faster-RCNN as well. We fuse sensor modalities at a single location, after feature extraction and before region proposal generation. Object detection networks almost entirely comprise of convolutional operations in order to exploit locality in the input data for efficient processing. Main challenge for processing different sensor modalities is to create locality in the extracted features from all sensors. We try to solve this by using a transformer module which learns to translate input features from one sensor space to another and then perform object detection on combined feature maps from all sensors.

3.1 Transformer

Key component of our proposed fusion architecture is the transformer module, which is used to transform feature maps from camera space into birdview, creating locality in the feature maps needed for further processing via convolutional layers. We adopt a convolutional encoder-decoder scheme as employed in [10]. Our transformer architecture is shown in figure 3. It is not dissimilar to an autoencoder. In contrast to traditional autoencoders however, its purpose is not to recreate its original input after it has passed through a low-capacity bottleneck layer, but to transform input data to be similar to data from another sensor domain. It represents a mapping $P : F_{ego} \rightarrow F_{birdview}$ from ego (camera) feature maps to birdview feature maps. Input to the encoder are feature maps F_{ego} from the camera space feature detector. These feature maps contain feature activations with locality w.r.t. to the input camera space. A cascade of convolutions with a stride of two downsamples these feature maps, which are then fed through two fully-connected network layers and are upsampled through transposed convolutions to generate the output feature map $F_{birdview}$.

The main idea behind the encoder-decoder architecture is that by squeezing information through a fully-connected bottleneck layer and then upsampling this information to the desired output space (birdview in our case), the transformer is forced to represent information in an perspective-invariant manner while the convolutional encoder-decoder has to learn a projection from one space to the other. The output space is enforced by letting the system predict objects in the desired output space.

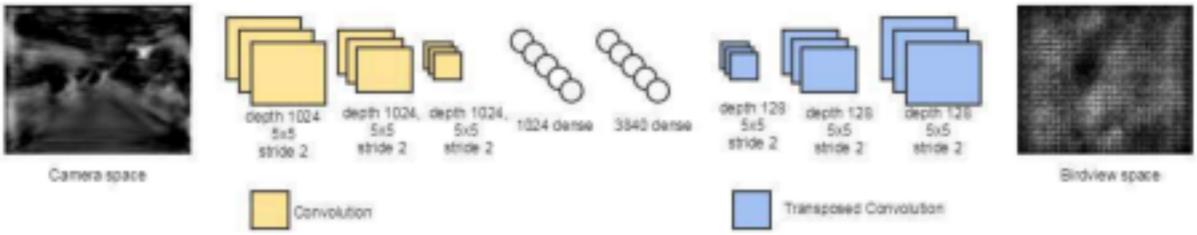


Abbildung 3: Transformer: Encoder-decoder architecture for mapping feature maps from ego space to birdview space. Rectangular elements denote convolutional operations while circle-shaped elements are fully-connected layers.

3.2 Fusion

Since the transformer module outputs transformed ego space feature maps which match the birdview Lidar feature maps in dimension and locality, any method for combining feature maps is applicable. Popular schemes include *element-wise mean* (as adopted in [5] or [1]) or *concatenation* operation (used in [1]). For our experiments we use depth-wise concatenation, meaning a depth-wise stacking of feature maps.

4 Training

For training the multi-modal object detection network we use a three step training method: First, we train **baseline single-modality networks** on camera and projected birdview Lidar images. In the second step we train the **transformer** by taking featuremaps of the previously trained camera network, feed them through the transformer and into the object detection stage of the trained Lidar network. In the third step we train the **fusion network** by taking the pretrained feature extractors of the camera and Lidar networks as well as the transformer and train a second stage object detector on the fused output featuremaps. Objective function for all training steps is the default loss function as employed in the original Faster-RCNN paper [7], minimizing error over class and object location for each of the anchor boxes that the network predicts:

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i,p_i^*) + \lambda\frac{1}{N_{reg}}\sum_i L_{reg}(t_i,t_i^*) \qquad (1)$$

L_{pred} denotes the *classification loss* while L_{reg} is the bounding box regression loss. The objective function is applied to the first stage of the network, where only objectness is predicted (binary class, object or no object) as well as for the second stage, where final multi-class classification and bounding box location refinement are performed. Note that we use the same end-to-end objective function for the different training phases of the fusion network (baseline single-modality object detector, transformer and fusion) and only freeze different parts of the network weights.

4.1 Training of transformer

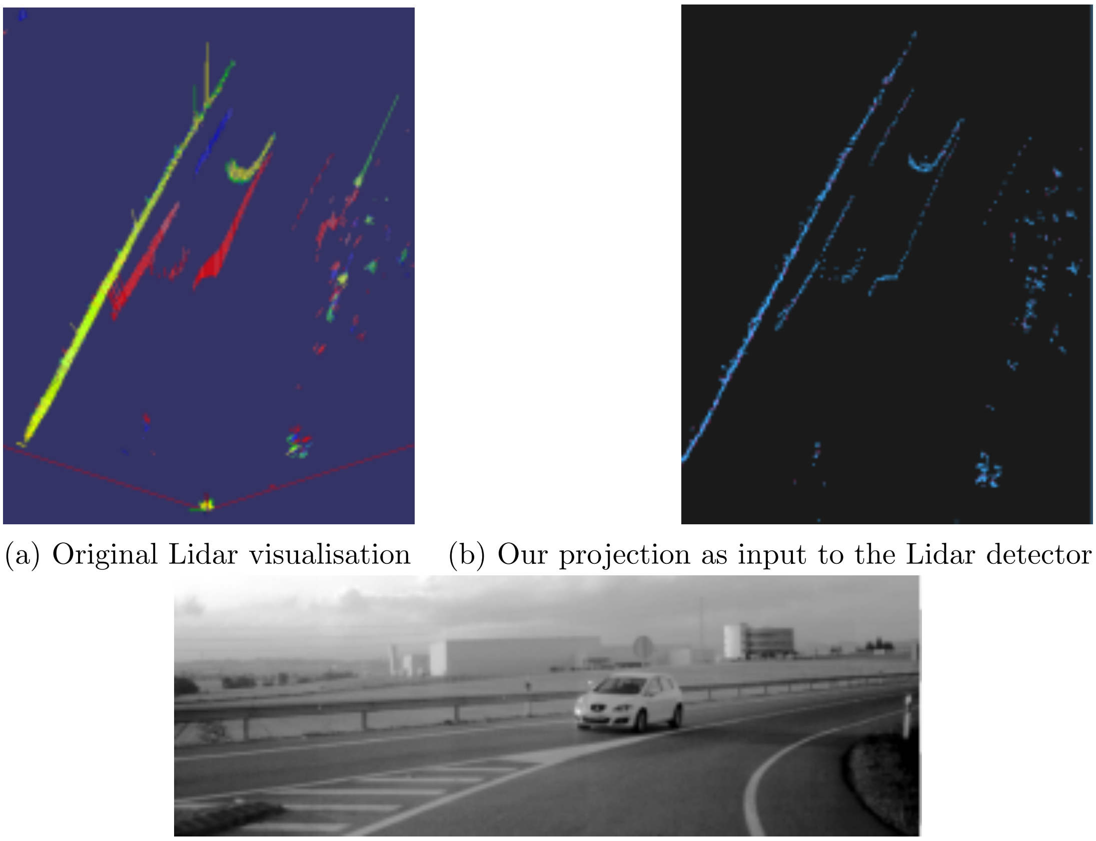
To train the perspective transformer, we take the pre-trained *feature extractor* from the camera object detection model and feed it through the transformer into the region-proposal-network (RPN) and detector network of the pre-trained birdview Lidar model. We thus use ego space camera images to predict objects in birdview space. Fixing weights of all pre-trained modules forces the transformer to learn the projection from ego features to birdview features.

4.2 Training of fusion network

The entire fusion network is trained just like the single-modality object detector network, with the only difference being that feature maps from ego space are now transformed into birdview space by the transformer and concatenated with those of the birdview Lidar feature extractor. Objective is to predict objects in birdview space. We can employ the default Faster-RCNN training procedure as the network is otherwise unmodified. Different schemes of pre-training and weight-fixing different parts of the network are imaginable, depending on the particular use-case.

5 Experiments

We possess our own hand-labeled dataset containing approximately 16h of raw front camera images and ScaLa 4-channel Lidar pointclouds. We project 3D Lidar data into 2D birdview by employing a scheme introduced by [1]: Using two grids, we encode height and intensity of the Lidar reflection with highest azimuth angle per grid location (see figure 4). For experiments we use a subset of 1h length, sampled at 10 Hz and comprised of mixed driving (highway/urban/city) at daytime (see figure 1). Two subsequences of consecutive frames, amounting to 20% of the total length of the sequence, have been put aside for validation, the rest has been used as training data. Reported mean average precision (mAP) figures are on the validation set only. We used a modified version of the Object Detection API [4] for implementation.



(a) Original Lidar visualisation (b) Our projection as input to the Lidar detector

(c) Camera image (cropped), input to the camera detection network

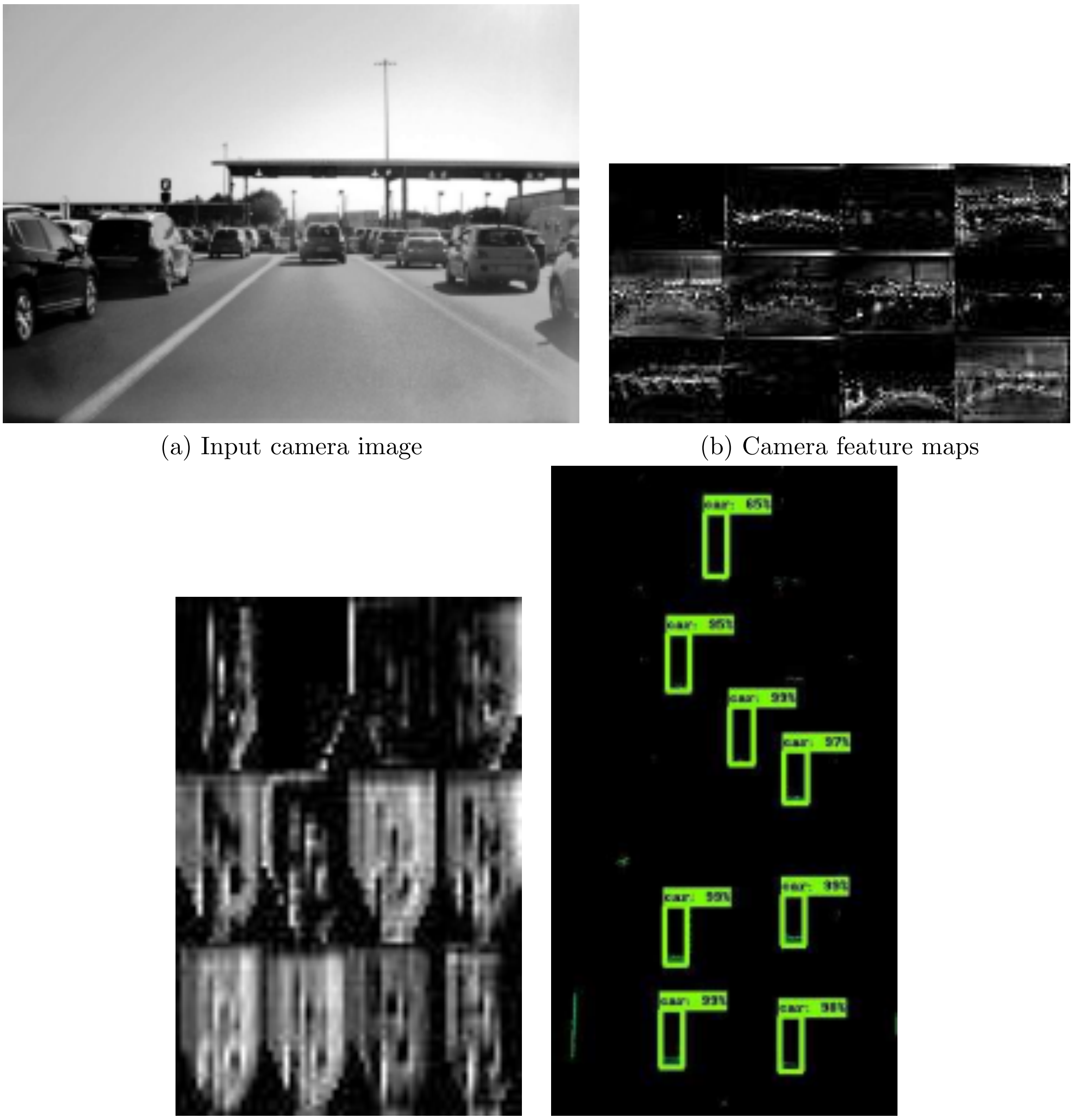
Abbildung 4: Samples of our Lidar projection. **(a)** shows the original Lidar birdview plot with length of the sticks signifying reflection intensity and color denoting one of the four channels. **(b)** shows our two channel projection, one channel encoding height and the other intensity.

5.1 Predicting objects in birdview space from camera images

We train the transformer module with input feature maps from a pre-trained camera-only feature extractor, while providing the transformer's output to a pre-trained RPN and detector module for birdview objects. Figure 5 shows input and output feature maps of the transformer after training to gain results shown in table 1. The transformer successfully learns a mapping from egoview to birdview, allowing the subsequent detector stage to predict birdview objects (and thus implicitly predicting object distance from camera images). Furthermore, the transformer outputs features so that they match the ones from the native Lidar feature extractor, since the detector module was originally trained with those and is now fixed.

5.2 Fusion of camera and Lidar

For fusion experiments, we use the setup as depicted in figure 2 and described in section 3. Camera-space feature maps are fed to the transformer module and afterwards to the fusion module, where they are concatenated with the feature maps from the Lidar network. Weights of the feature extractors are fixed, while the transformer and detector network are being trained. Objective is to predict object's positions (i.e. bounding box coordinates)



(a) Input camera image

(b) Camera feature maps

(c) Transformed feature maps

(d) Fusion predictions

Abbildung 5: (a) shows a sample camera input image of our test set. (b) shows an exemplary plot of the first twelve (of 1024) features maps F_{ego} for a sample frame as input to the transformer. (c) shows resulting transformed feature maps $F_{birdview}$ as output by the transformer after being trained as described in section 4.1. Using $F_{birdview}$ only, we can successfully predict objects' position and class, as shown in section 5.1. Together with birdview features maps from the pre-trained Lidar object detector, we can then predict objects in birdview using both input modalities (shown in section 5.2). (d) shows predictions of object locations for the sample frame. The ego vehicle is at the bottom of the frame, facing upwards.

in birdview. To show effectiveness of our fusion scheme, we provide reference results for a Lidar-only object detector which has been trained under the same conditions (just without the camera path enabled). For results refer to table 1. We can report an increase of 0.17 and 0.07 mAP for classes *truck* and *car* for our employed dataset, showing effectiveness of our proposed fusion scheme.

Tabelle 1: Results on our validation set. We report mean average precision (mAP) at bounding box intersection-over-union (IOU) 0.5.

	Birdview from ego camera	Lidar-only	Fusion of Lidar/camera
mAP Car	0.40	0.80	**0.87**
mAP Truck	0.54	0.63	**0.80**

6 Conclusion

We propose a novel and simple method for learning a transformation of feature maps from one sensor space to another. It allows for multi-modal object detection using state-of-the-art convolution-based object detection networks. Key advantage of the proposed method is that it allows most parts of the object detector networks to be pre-trained using only a single modality, catering to special requirements employing neural network object detectors in a series-production environment (such as multiple sensor combinations or changing sensor models for different variants of a production car). Experiments show that we can successfully predict objects in birdview space from camera images alone, showing effectiveness of the proposed transformer module, and use this transformer module to fuse camera and Lidar data for improved object detection performance.

Literatur

[1] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. *arXiv preprint arXiv:1611.07759*, 2016.

[2] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3354–3361. IEEE, 2012.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *arXiv:1512.03385 [cs]*, December 2015. arXiv: 1512.03385.

[4] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, and Kevin Murphy. Speed/accuracy trade-offs for modern convolutional object detectors. *arXiv:1611.10012 [cs]*, November 2016. arXiv: 1611.10012.

[5] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven Waslander. Joint 3d Proposal Generation and Object Detection from View Aggregation. *arXiv:1712.02294 [cs]*, December 2017. arXiv: 1712.02294.

[6] Charles R. Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J. Guibas. Frustum PointNets for 3d Object Detection from RGB-D Data. *arXiv:1711.08488 [cs]*, November 2017. arXiv: 1711.08488.