Learned Two-Plane Perspective Prior based Image Resampling for Efficient Object Detection

Anurag Ghosh N. Dinesh Reddy Christoph Mertz Srinivasa G. Narasimhan
Carnegie Mellon University
{anuraggh, dnarapur, cmertz, srinivas}@cs.cmu.edu

Abstract

Real-time efficient perception is critical for autonomous navigation and city scale sensing. Orthogonal to architectural improvements, streaming perception approaches have exploited adaptive sampling improving real-time detection performance. In this work, we propose a learnable geometry-guided prior that incorporates rough geometry of the 3D scene (a ground plane and a plane above) to resample images for efficient object detection. This significantly improves small and far-away object detection performance while also being more efficient both in terms of latency and memory. For autonomous navigation, using the same detector and scale, our approach improves detection rate by ***+4.1*** AP_S *or* +39% *and in real-time performance by* ***+5.3*** sAP_S *or* +63% *for small objects over state-of-the-art (SOTA). For fixed traffic cameras, our approach detects small objects at image scales other methods cannot. At the same scale, our approach improves detection of small objects by* 195% *(****+12.5*** AP_S*) over naive-downsampling and* 63% *(****+4.2*** AP_S*) over SOTA.*

1. Introduction

Visual perception is important for autonomous driving and decision-making for smarter and sustainable cities. Real-time efficient perception is critical to accelerate these advances. For instance, a single traffic camera captures half a million frames every day or a commuter bus acting as a city sensor captures one million frames every day to monitor road conditions [11] or to inform public services [23]. There are thousands of traffic cameras [26] and nearly a million commuter buses [58] in the United States. It is infeasible to transmit and process visual data on the cloud, leading to the rise of edge architectures [48]. However, edge devices are severely resource constrained and real-time inference requires down-sampling images to fit both latency and memory constraints severely impacting accuracy.

On the other hand, humans take visual shortcuts [20] to recognize objects efficiently and employ high-level semantics [20, 57] rooted in scene geometry to focus on relevant parts. Consider the scene in Figure 1 (c), humans can recognize the distant car despite its small appearance (Figure 1 (d)). We are able to contextualize the car in the 3D scene, namely (1) it's on the road and (2) is of the right size we'd expect at that distance. Inspired by these observations, can we incorporate semantic priors about scene geometry in our neural networks to improve detection?

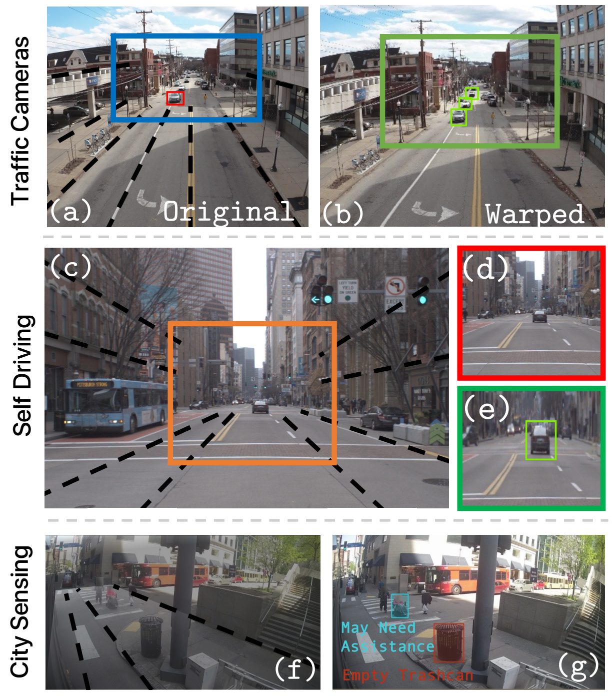

Figure 1. Geometric cues (black dashed lines) are implicitly present in scenes. Our Perspective based prior exploits this geometry. Our method (a) takes an image and (b) warps them, and performs detection on warped images. Small objects which are (d) not detected when naively downsampled but (e) are detected when enlarged with our geometric prior. Our method (f) uses a geometric model to construct a saliency prior to focus on relevant areas and (g) enables sensing on resource-constrained edge devices.

In this work, we develop an approach that enables object detectors to "zoom" into relevant image regions (Figure 1 (d) and (e)) guided by the geometry of the scene. Our approach considers that most objects of interests are present within two planar regions, either on the ground plane or within another plane above the ground, and their size in the image follow a geometric relationship. Instead of uniformly downsampling, we sample the image to enlarge far away regions more and detect those smaller objects.

While methods like quantization [18], pruning [19], distillation [6] and runtime-optimization [17] improve model efficiency (and are complementary), approaches exploiting spatial and temporal sampling are key for enabling efficient real-time perception [22, 29]. Neural warping mechanisms [24, 40] have been employed for image classification and regression, and recently, detection for self-driving [55]. Prior work [55] observes that end-to-end trained saliency networks fail for object detection. They instead turn to heuristics such as dataset-wide priors and object locations from previous frames, which are suboptimal. We show that formulation of learnable geometric priors is critical for learning end-to-end trained saliency networks for detection. We validate our approach in a variety of scenarios to showcase the generalizability of geometric priors for detection in self-driving on Argoverse-HD [29] and BDD100K [62] datasets, and for traffic-cameras on WALT [41] dataset.

- On Argoverse-HD, our learned geometric prior improves performance over naive downsampling by **+6.6** AP and **+2.7** AP over SOTA using the same detection architecture. Gains from our approach are achieved by detecting small far-away objects, improving by **9.6** AP_S (or 195%) over naive down-sampling and **4.2** AP_S (or 63%) over SOTA.
- On WALT, our method detects small objects at image scales where other methods perform poorly. Further, it significantly improves detection rates by **10.7** AP_S over naive down-sampling and **3** AP_S over SOTA.
- Our approach improves object tracking (+4.8% MOTA) compared to baseline. It also improves tracking quality, showing increase of **+7.6%** $MT\%$ and reduction of **-6.7%** $ML\%$.
- Our approach can be deployed in resource constrained edge devices like Jetson AGX to detect 42% more rare instances while being **2.2X** faster to enable real-time sensing from buses.

2. Related Work

We contextualize our work with respect to prior works modelling geometry and also among works that aim to make object detection more accurate and efficient.

Vision Meets Geometry: Geometry has played a crucial role in multiple vision tasks like detection [9, 21, 53, 60], segmentation [30, 51], recognition [15, 52] and reconstruction [27, 38, 49]. Perspective Geometric constraints have been used to remove distortion [63], improve depth prediction and semantic segmentation [28] and feature matching [56]. However, in most previous works [27, 49, 60] exploiting these geometric constraints have mainly been concentrated around improving 3D understanding. This can be attributed to a direct correlation between the constraints and the accuracy of reconstruction. Another advantage is the availability of large RGB-D and 3D datasets [5, 16, 43] to learn and exploit 3D constraints. Such constraints have been under-explored for learning based vision tasks like detection and segmentation. A new line of work interpreting classical geometric constraints and algorithms as neural layers [7, 47] have shown considerable promise in merging geometry with deep learning.

Learning Based Detection: Object detection has mostly been addressed as an learning problem. Even classical-vision based approaches [13, 59] extract image features and learn to classify them into detection scores. With deep learning, learnable architectures have been proposed following this paradigm [4, 37, 42, 44], occasionally incorporating classical-vision ideas such as feature pyramids for improving scale invariance [31]. While learning has shown large improvements in accuracy over the years they still perform poorly while detecting small objects due to lack of geometric scene understanding. To alleviate this problem, we guide the input image with geometry constraints, and our approach complements these architectural improvements.

Efficient Detection with Priors: Employing priors with learning paradigms achieves improvements with little additional human labelling effort. Object detection has traditionally been tackled as a learning problem and geometric constraints were sparsely used for such tasks, constraints like ground plane [21, 53] were used.

Temporality [14, 55, 61] has been exploited for improving detection efficiently. Some of these methods [14, 55] deform the input image using approach that exploit temporality to obtain saliency. This approach handles cases where object size decreases with time (object moving away from the camera in scene), but cannot handle new incoming objects. None of these methods explicitly utilize geometry to guide detection, which handles both these cases. Our two-plane prior deforms the image while taking perspective into account without biasing towards previous detections.

Another complementary line of works automatically learn metaparameters (like image scale) [10, 17, 50] from image features. However, as they do not employ adaptive sampling accounting for image-specific considerations, performance improvements are limited. Methods not optimized for online perception like AdaScale [10] for video object detection do not perform well in real-time situations.

3. Approach

We describe how a geometric model rooted in the interpretation of a 3D scene can be derived from the image. We then describe how to employ this rough 3D model to construct saliency for warping images and improving detection.

3.1. Overview

Object sizes in the image are determined by the 3D geometry of the world. Let us devise a geometric inductive prior considering a camera mounted on a vehicle. Without loss of generality, assume the vehicle is moving in direction of the dominant vanishing point.

We are interested in objects that are present in a planar region (See Figure 2) of width P_1P_2 corresponding to the camera view, of length P_1P_3 defined in the direction of the vanishing point. This is the planar region on the ground on which most of the objects of interest are placed (vehicles, pedestrians, etc) and another planar region $Q_1...Q_4$ parallel to this ground plane above horizon line, such that all the objects are within this region (e.g., traffic lights).

From this simple geometry model, we shall incorporate relationships derived from perspective geometry about objects, i.e., the scale of objects on ground plane is inversely proportional to their depth w.r.t camera [21].

3.2. 3D Plane parameterization from 2D images

We parameterize the planes of our inductive geometric prior. We represent 2D pixel projections $u_1...u_4$ of 3D points $P_1...P_4$. Assume that the dominant vanishing point in the image is $v = (v_x, v_y)$ and let the image size be (w, h). Consider u_1 (Figure 2 (b)). We can define a point on the edge of the image plane,

$$u_L = (0, v_y + v_x \tan\theta_1) \tag{1}$$

u_1 can expressed as a linear combination of v and u_L,

$$u_1 = \alpha_1 u_L + (1 - \alpha_1)v \tag{2}$$

Similarly, for u_2, we can define u_R in terms of v and θ_2 and α_2 while u_3 and u_4 are defined like Equation 1 to represent any arbitrary plane in this viewing direction. However, for simplicity, for ground plane we fix them as $(0, h)$ and (w, h) respectively. Consider the planar region $Q_1...Q_4$ at height H above the horizon line. We can similarly define θ_3 and θ_4 to represent the angles from the horizon in the opposite direction and define q_1 and q_2. Again, we set q_3 as $(0, 0)$ and q_4 as $(w, 0)$. We now have 4 points to calculate homographies H_{plane} for both planes.

For now, assume v is known. However, we still do not know the values for θ's and α, and we shall learn these parameters end-to-end from task loss. These parameters are learned and fixed for a given scenario in our learning paradigm. Our re-parameterization aims to ease learning of these parameters as we clamp the values of α's to $[0, 1]$ and θ's to $[-\frac{\pi}{2}, \frac{\pi}{2}]$. It should be noted that all the operations are differentiable.

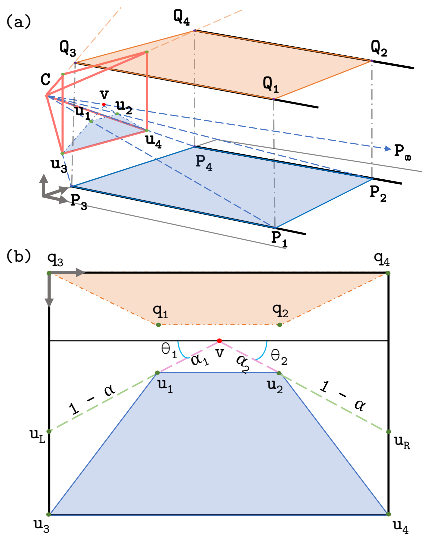

Figure 2. **Geometry Of The Two Plane Perspective Prior:** (a) describes the single view geometry of the proposed two plane prior. Region on the ground plane defined by $P_1, ...P_4$, and rays emanating from camera C to P_i intersect at $u_1...u_4$ on the image plane. The vanishing point v maps to P_∞. This planar region accounts for small objects on the ground plane. To account for objects that are tall or do not lie on the ground plane, we consider another plane $Q_1..Q_4$ above the horizon line. These two planes encapsulate all the relevant objects in the scene. (b) depicts the re-parameterization of the two planes in the 2D image. Instead of representing the planar points $u_1...u_4$ as pixel coordinates, we instead parameterize them in terms of the vanishing point v, θ's and α to ease learning.

3.3. From Planes to Saliency

We leverage geometry to focus on relevant image regions through saliency guided warping [24, 40], and create a saliency map from a parameterized homography using $u_1..u_4$ defined earlier. Looking at ground plane from two viewpoints (Figure 3 (a) and (b)), object size decreases by their distance from the camera [21]. We shall establish a relationship to counter this effect and "sample" far-away objects on the ground plane more than nearby objects.

The saliency guided warping proposed by [40] operates using an inverse transformation $\mathcal{T}_S^{-1}$ parameterized by a saliency map S as follows,

$$I'(x, y) = W_{\mathcal{T}}(I) = I(\mathcal{T}_S^{-1}(x, y)) \tag{3}$$

where the warp $W_{\mathcal{T}}$ implies iterating over output pixel coordinates, using $\mathcal{T}_S^{-1}$ to find corresponding input coordinates (non-integral), and bilinearly interpolating output color from neighbouring input pixel grid points. For each

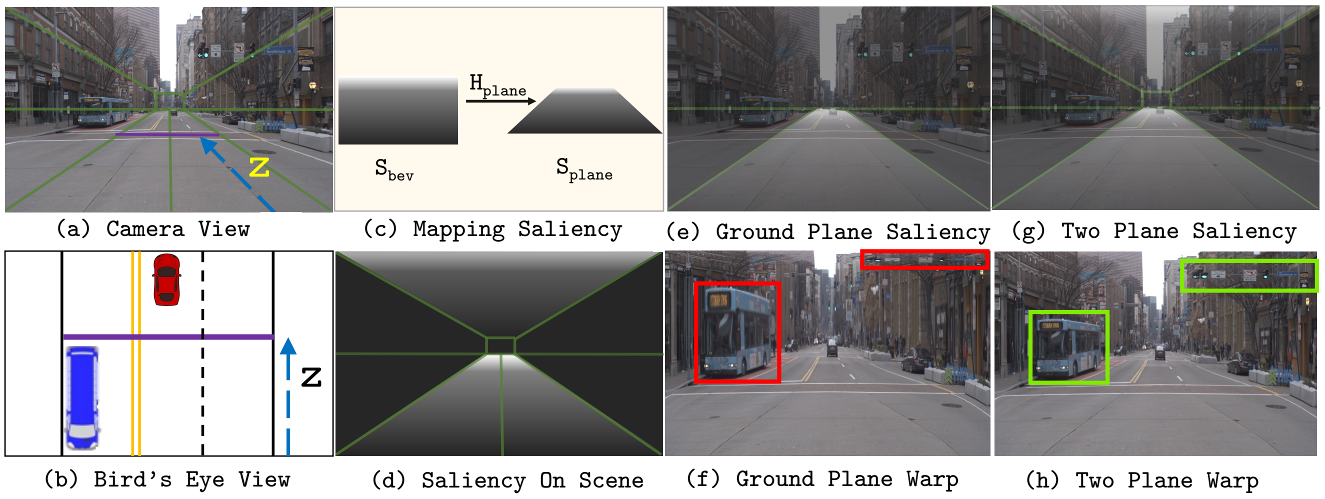

Figure 3. **Two-Plane Perspective Prior based Image Resampling:** Consider the scene of car, bus and traffic light from (a) camera view and (b) (simplified) bird's eye view. (c) Saliency function that captures the inverse relationship between object size (in camera view) and depth (bird's eye view is looking at XZ plane from above) can be transferred to the camera view (d), by mapping row z using H (marked by blue arrows). (e) and (f) shows that ground plane severely distorts nearby tall objects while squishing traffic light. (g) and (h) shows that additional plane reduces distortion for both tall objects and objects not on ground plane.

input pixel (x, y), pixel coordinates with higher $S(x, y)$ values (i.e. salient regions) would be sampled more.

We construct a saliency S respecting the geometric properties that we desire. Let H_{plane} be the homography between the camera view (using coordinates $u_1 ... u_4$) and a bird's eye view of the ground plane assuming plane size to be the original image size (w, h). In bird's eye view, we propose saliency function for a row of pixels z (assuming bottom-left of this rectangle as (0, 0)) as,

$$S_{bev}(z) = e^{\nu(\frac{z}{h}-1)} \tag{4}$$

with a learnable parameter ν (> 1). ν defines the extent of sampling with respect to depth.

To map this saliency to camera view, we warp S_{bev} via perspective transform W_p and H_{plane} (Figure 3 (c)),

$$S_{plane} = W_p(H^{-1}_{plane}, S_{bev}) \tag{5}$$

We have defined saliency S_{plane} given H_{plane} in a differentiable manner. Our saliency ensures that objects on the ground plane separated by depth Z are sampled by the factor $e^{\nu\frac{Z}{h}}$ in the image.

3.4. Two-Plane Perspective Prior

Ground Plane saliency focuses on objects that are geometrically constrained to be on this plane and reasonably models objects far away on the plane. However, nearby and tall objects, and small objects far above the ground plane are not modelled well. In Fig 3 (f), *nearby objects above ground plane* (traffic lights), they are highly distorted. Critically, these *same objects when further away* are rendered small in size and appear close to ground (and thus modelled well). Objects we should **focus** more on are thus the former compared to the latter. Thus, another plane is needed, and direction of the saliency function is reversed to $\hat{S}_{bev}(z) = e^{\hat{\nu}(((h-z)/h)-1)}$ to account for these objects that would otherwise be severely distorted.

To represent the Two-Plane Prior, we represented the planar regions as saliencies. The overall saliency is,

$$S = S_{ground\_plane} + \lambda S_{top\_plane} \tag{6}$$

where λ is a learned parameter.

3.5. Additional Considerations

Warping via a piecewise saliency function imposes additional considerations. The choice of deformation method is critical, saliency sampler [40] implicitly avoids drastic transformations common in other appraoches. For e.g., Thin-plate spline performs worse [40], produces extreme transformations and requires regularization [14].

Fovea [55] observes that restricting the space of allowable warps such that axis alignment is preserved improves accuracy, we adopt the separable formulation of $\mathcal{T}^{-1}$,

$$\mathcal{T}^{-1}_x(x) = \frac{\int_{x'} S_x(x')k(x', x)x'}{\int_{x'} S_x(x')k(x, x')} \tag{7}$$

$$\mathcal{T}^{-1}_y(y) = \frac{\int_{y'} S_y(y')k(y', y)y'}{\int_{y'} S_y(y')k(y, y')} \tag{8}$$

where k is a Gaussian kernel. To convert a saliency map S to S_x and S_y we marginalize it along the two axes. Thus entire rows or columns are "stretched" or "compressed".

Two-plane prior is learnt end to end as a learnable image warp. For object detection, labels need to be warped too, and [40]'s warp is invertible. Like [55], We employ the loss $\mathcal{L}(\mathcal{T}^{-1}(f_\phi(W_\mathcal{T}(I)), L)$ where (I, L) is the image-label pair and omit the use of delta encoding for training RPN [44] (which requires the existence of a closed form $\mathcal{T}$), instead adopting GIoU loss [45]. This ensures $W_\mathcal{T}$ is learnable, as $\mathcal{T}^{-1}$ is differentiable.
We *did not* assume that the vanishing point is within the field of view of our image, and our approach places no restrictions on the vanishing point. Thus far, we explained our formulation while considering a single dominant vanishing point, however, multiple vanishing points can be also considered. Please see supplementary for more details.

3.6. Obtaining the Vanishing Point

We now describe how we obtain the vanishing point. Many methods exist with trade-offs in accuracy, latency and memory which which inform our design to perform warping efficiently with minimal overheads.
Fixed Cameras: In settings like traffic cameras, the camera is fixed. Thus, the vanishing point is fixed, and we can cache the corresponding saliency S, as all the parameters, once learnt, are fixed. We can define the vanishing point for a camera manually by annotating two parallel lines or any accurate automated approach. Saliency caching renders our approach extremely efficient.
Autonomous Navigation: Multiple assumptions simplify the problem. We assume that there is one dominant vanishing point and a navigating car is often moving in the viewing direction. Thus, we assume that this vanishing point lies inside the image, and directly regress v from image features using a modified coordinate regression module akin to YOLO [36, 42]. This approach appears to be memory and latency efficient. Other approaches, say, using parallel lane lines [1] or inertial measurements [3] might also be very efficient. An even simpler assumption is to employ the average vanishing point, as vanishing points are highly local, we observe this is a good approximation.
Temporal Redundancies: In videos, we exploit temporal redundancies, the vanishing point is computed every n_v frames and saliency is cached to amortize latency cost.
General Case: This is the most difficult case, and many approaches have been explored in literature to find all vanishing points. Classical approaches [54] while fast are not robust, while deep-learned approaches [34, 35, 65] are accurate yet expensive (either in latency or memory).

3.7. Learning Geometric Prior from Pseudo-Labels

Prior work [55] have shown improvements in performance on pre-trained models via heuristics, which didn't require any training. However, their method *still employs domain-specific labels* (say, from Argoverse-HD) to generate the prior. Our method can't be used directly with pre-trained models as it learns geometrically inspired parameters end-to-end. However, domain-specific images without labels can be exploited to learn the parameters.
We propose a simple alternative to learn the proposed prior using pre-trained model without requiring additional domain-specific labels. We generate pseudo-labels from our pre-trained model inferred at 1x scale. We fine-tune and learn our warp function (to 0.5x scale) end-to-end using these "free" labels. Our geometric prior shows improvements without access to ground truth labels.

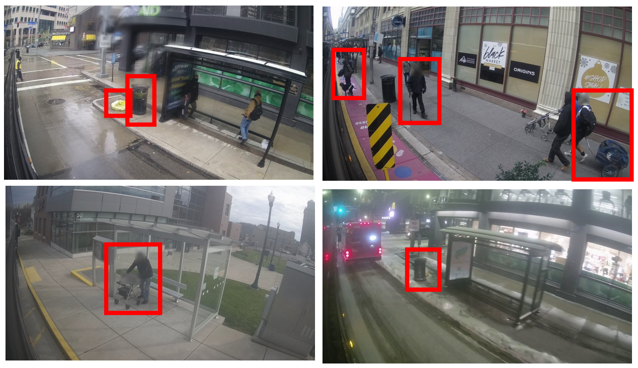
Figure 4. **Commuter Bus Dataset:** The captured data has a unique viewpoint, the average size of objects is small and captured under a wide variety of lighting conditions. Top-left and bottom-right images depict the same bus-stop and trashcan at different time of day and season.

4. Dataset And Evaluation Details

4.1. Datasets

Argoverse-HD [29]: We employ Argoverse-HD dataset for evaluation in the autonomous navigation scenario. This dataset consists of 30fps video sequences from a car collected at 1920×1200 resolution, and dense box annotations are provided for common objects of interest such as vehicles, pedestrians, signboards and traffic lights.
WALT [41]: We employ images from 8 4K cameras that overlook public urban settings to analyze the flow of traffic vehicles. The data is captured for 3-second bursts every few minutes and only images with notable changes are stored. We annotated a set of 4738 images with vehicles collected over the time period of a year covering a variety of day/night/dawn settings, seasonal changes and camera viewpoints. We show our results on two splits (approximately 80% training and 20% testing). The first, **All-Viewpoints**, images from all the cameras are equally represented in the train and test sets. Alternatively, we split by camera, **Split-by-Camera**, images from 6 cameras are part of the training set and 2 cameras are held out for testing.
Commuter Bus Dataset: We curated this dataset from a

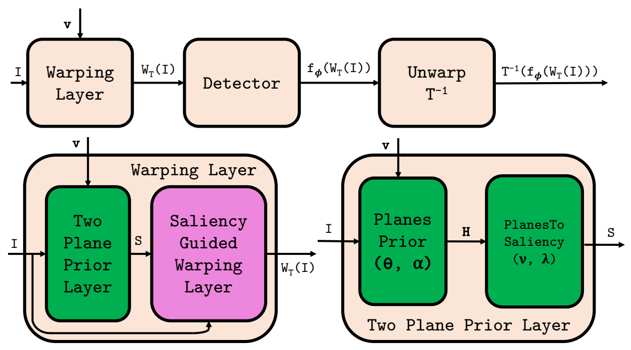

Figure 5. **Two-Plane Prior as a Neural Layer:** We implemented our approach as a global prior that is learned end-to-end from labelled data. Our prior is dependent on a vanishing point estimate to specify the viewing direction of the camera.

commuter bus running on an urban route. The 720p camera of interest has a unique viewpoint and scene geometry (See Figure 4). Annotated categories are trashcans and garbage bags (to help inform public services) and people with special needs (using wheelchairs or strollers), which are a rare occurrence. The dataset size is small with only 750 annotated images (split into 70% training and 30% testing). This is an extremely challenging dataset due to it's unique viewpoint, small object size, rare categories, along with variations in lighting and seasons.

4.2. Evaluation Details

We perform apples-to-apples comparisons on the same detector trained using the datasets with identical training schedule and hardware.

Data: We compare to methods that were trained on fixed training data. In contrast, sAP leaderboards [29] don't restrict data and evaluate on different hardware. We compare with [17, 29] from leaderboard, which follow the same protocols. Other methods on the leaderboard use additional training data to train off-the-self detectors. Our detectors would see similar improvements with additional data.

Real-Time Evaluation: We evaluate using **Streaming AP (sAP)** metric proposed by [29], which integrates latency and accuracy into a single metric. Instead of considering models via accuracy-latency tradeoffs [22], real-time performance can be evaluated by applying real-time constraints on the predictions [29]. Frames of the video are observed every 33 milliseconds (30 fps) and predictions for every frame must be emitted **before** the frame is observed (forecasting is necessary). For a fair comparison, **sAP** requires evaluation on the same hardware. Streaming results are not directly comparable with other work [29, 55, 61] as they use other hardware (say, V100 or 2080Ti), thus we run the evaluation on our hardware (Titan X).

Additional details are in the Supplementary.

Method	Scale	AP	AP_S	AP_M	AP_L	Latency (ms)
Faster R-CNN	0.5x	24.2	4.9	29.0	50.9	78.4 ± 1.8
Fovea (S_D) [55]	0.5x	26.7	8.2	29.7	54.1	83 ± 2.5
Fovea (S_I) [55]	0.5x	28.0	10.4	31.0	**54.5**	85 ± 2.7
Fovea (L:S_I) [55]	0.5x	28.1	10.3	30.9	54.1	85.4 ± 2.7
Two-Plane Pr. (Pseudo.)	0.5x	27.1	9.8	28.9	50.2	104.5 ± 8.5
Two-Plane Prior	0.5x	**30.8**	**14.5**	**31.6**	52.9	105 ± 8.5
Baseline at higher scales						
Faster R-CNN	0.75x	29.2	11.6	32.1	53.3	142 ± 2.5
Faster R-CNN	1.0x	33.3	16.8	34.8	53.6	220 ± 1.7

Table 1. **Evaluation on Argoverse-HD:** Two-Plane Prior outperforms both SOTA's dataset-wide and temporal priors in overall accuracy. Our method improves small object detection by $+4.1 AP_S$ or 39% over SOTA.

5. Results and Discussions

Accuracy-Latency Comparisons: On Argoverse-HD, we compare with Faster R-CNN with naive downsampling (Baseline) and Faster R-CNN paired with adaptive sampling from Fovea [55] which proposed two priors, a dataset-wide prior (S_D) and frame-specific temporal priors (S_I & $L : S_I$; from previous frame detections).

Two-Plane Prior improves (Table 1) upon baseline at the same scale by **+6.6** AP and over SOTA by **+2.7** AP. For small objects, the improvements are even more dramatic, our method improves accuracy by **+9.6** AP_S or **195%** over baseline and **+4.2** AP_S or **45%** over SOTA respectively. Surprisingly, our method at 0.5x scale improves upon Faster R-CNN at 0.75x scale by **+1.6** AP having latency improvement of 35%. Our Two-Plane prior trained via pseudo-labels comes very close to SOTA which employ ground truth labels, the gap is only **-1** AP and improves upon Faster R-CNN model trained on ground truth by **+2.9 AP** inferred at the same scale.

WALT dataset [41] comprises images (only images with notable changes are saved) and not videos, we compare with Fovea [55] paired with the dataset-wide prior. We observe similar trends on both splits (Table 3 and Fig 6) and note large improvements over baseline and a consistent improvement over Fovea [55], specially for small objects.

Real-time/Streaming Comparisons: We use Argoverse-HD dataset, and compare using the **sAP** metric (Described in Section 4.2). Algorithms may choose any scale and frames as long as real-time latency constraint is satisfied.

All compared methods use Faster R-CNN, and we adopt their reported scales (and other parameters). Streamer [29] converts any single frame detector for streaming by scheduling which frames to process and interpolating predictions between processed frames. AdaScale [10] regresses optimal scale from image features to minimize single-frame latency while Adaptive Streamer [17] learns

ID	Method	Scale(s)	sAP	sAP_S	sAP_M	sAP_L
1	Streamer [29]	0.5x	18.3	3.1	15.0	40.9
2	1 + Adascale [10]	0.2x-0.6x	13.4	0.2	8.6	37.4
3	Adaptive Streamer [17]	0.2x-0.6x	21.3	4.2	18.8	47.0
4	1 + FOVEA (S_I) [55]	0.5x	24.1	8.4	24.7	48.7
5	1 + Ours (Avg VP)	0.5x	**29.9**	**13.7**	**31.3**	**52.2**
6	1 + Ours	0.5x	**30.0**	**13.7**	**31.5**	**52.2**
7	1 + Ours (VP Oracle)	0.5x	30.7	14.5	31.6	52.9

Table 2. **Streaming Evaluation on Argoverse-HD:** Ours denotes Two-Plane Prior. Every frame's prediction (streamed at 30FPS) must be emitted **before** frame is observed [29] (via forecasting). All methods evaluated on Titan X GPU. Underlying detector (Faster R-CNN) is constant across approaches, improvements are solely from spatial sampling mechanisms. Notice improved detection of small objects by $+5.3 sAP_S$ or 63% over SOTA.

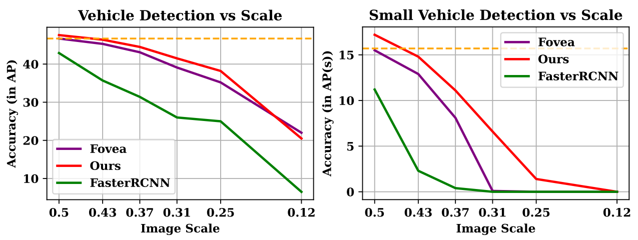

Figure 6. **WALT All-Viewpoints Split:** Our approach (Two-Plane Prior) shows improved overall performance over naive downsampling and state-of-the-art adaptive sampling technique, specially for small objects at all scales (starting from 0.5x). Horizontal line (orange) indicates performance at maximum possible scale (0.6x) the base detector was trained at (memory constraints).

scale choice in the streaming setting. Both these methods employ naive-downsampling. State-of-the-art, Fovea [55] employs the temporal prior (S_I). From Table 2, Two-Plane prior outperforms the above approaches by **+16.5** sAP, **+8.6** sAP and **+5.9** sAP respectively. Comparison with [10, 17, 29] shows the limitations of naive downsampling, even when "optimal" scale is chosen. Our geometric prior greatly improves small object detection performance by **63%** or **+5.3** sAP_S over SOTA. To consider dependence on accurate Vanishing Point detection (and its overheads), we use NeurVPS [65] as oracle (we simulate accurate prediction with zero delay) to obtain an upper bound, we observe even average vanishing point location's performance is within 0.8 sAP.

Accuracy-Scale Tradeoffs: We experiment with WALT **All-Viewpoints** split to observe accuracy-scale trade-offs. The native resolution ($4K$) of the dataset is extremely large and the gradients don't fit within 12 GB memory of our Titan X GPU, thus we cropped the skies and other static regions to reduce input scale (1x) to 1500 $\times$ 2000. Still, the highest scale we were able to train our baseline Faster

Method	Scale	AP	AP_S	AP_M	AP_L
Faster R-CNN	0.25x	16.7	0	7.2	42.3
FOVEA (S_D) [55]	0.25x	23.2	0	13.9	54.1
Two-Plane Prior	0.25x	**25.2**	**1.0**	**18.2**	**54.5**
Faster R-CNN	0.5x	29.2	4.9	24.7	55.5
FOVEA (S_D) [55]	0.5x	34.4	8.7	30.5	**59.4**
Two-Plane Prior	0.5x	**36.4**	**11.6**	**32.3**	59.0
Baseline at higher scales					
Faster R-CNN	0.6x	33.2	9.3	28.6	56.7
Faster R-CNN*	1.0x	34.3	12.6	30.7	54.3

Table 3. **WALT Camera-Split:** The viewpoints on the test set were not seen, and Two-Plane Prior shows better performance over both naive downsampling and state-of-the-art adaptive sampling as it generalizes better to unseen scenes and viewpoints. **Not trained at that scale due to memory constraints on Titan X.*

R-CNN model is 0.6x. So, we use aggressive downsampling scales $\{0.5, 0.4375, 0.375, 0.3125, 0.25, 0.125\}$. The results are presented in Figure 6. We observe a large and consistent improvement over baseline and Fovea [55], specially for small objects. For instance, considering performance at 0.375x scale, our approach is better than baseline by **+13.1** AP and Fovea by **+1.4** AP for all objects.

For small objects, we observe dramatic improvement, at scales smaller than 0.375x, other approaches are unable to detect any small objects while our approach does so until 0.125x scale, showing that our approach degrades more gracefully. At 0.375x scale, our approach improves upon Faster R-CNN by **+10.7** AP_S and Fovea by **+3.0** AP_S.

Generalization to new viewpoints: We use WALT **Camera-Split**, the test scenes and viewpoint are unseen in training. E.g., the vanishing point of one of the held-out cameras is beyond the image's field of view. We operate on the same scale factors in the earlier experiments, and results are presented in Table 3. We note lower overall performance levels due to scene/viewpoint novelty in the test sets. Our approach generalizes better due to the explicit modelling of the viewpoint via the vanishing point (See Section 3.2). We note trends similar to previous experiment, we demonstrate improvements of **+8.5** AP over naive-downsampling and **+2.0** AP over Fovea [55] at 0.25x scale.

Tracking Improvements: We follow tracking-by-detection and pair IOUTracker [2] with detectors on Argoverse-HD dataset. MOTA and MOTP evaluate overall tracking performance. From Table 4, Our method improves over baseline by **+4.8%** and **+0.7%**. We also focus on tracking quality metrics, Mostly Tracked % (**MT%**) evaluates the percentage of objects tracked for atleast 80% of their lifespan while Mostly Lost % (**ML%**) evaluates percentage of objects for less than 20% of their lifespan. In both these cases, our approach improves upon the baseline by **+7.6%** and **-6.7%** respectively. To autonomous navigation, we define two rel-

Method	MOTA ↑	MOTP ↑	MT% ↑	ML% ↓	MOS ↓	ALE% ↑
Faster RCNN	39.8	82.3	30.7	35.6	37.1	59.3 %
Fovea (S_D) [55]	43.9	81.9	34.1	31.9	34.8	+5.4 %
Fovea (S_I) [55]	44.3	81.8	36.7	**28.4**	33.8	+8.4 %
Two-Plane Prior	**44.6**	**83.0**	**38.3**	28.9	**31.6**	**+9.9 %**

Table 4. **Tracking Improvements**: We setup a tracking by detection pipeline and replace the underlying detection method and observe improvements if any. All the detectors employ the Faster R-CNN architecture and are executed at 0.5x scale. We observe improvements in tracking metrics due to Two-Plane Prior.

evant metrics, namely, average lifespan extension **(ALE)** and minimum object size tracked **(MOS)**, whose motivation and definitions can be viewed in supplementary. We observe that our improvements are better than both Fovea (S_D) and (S_I). As we observe, our method improves tracking lifespan and also helps track smaller objects.

Efficient City-Scale Sensing: We detect objects on Commuter Bus equipped with a Jetson AGX. Identifying (Recall) relevant frames is key on the edge. Recall of Faster R-CNN with at 1x scale is $43.3AR$ ($16.9AR_S$) (Latency: 350ms; infeasible for real-time execution) but drops to $31.7AR$ ($0.5AR_S$) at 0.5x scale when naively downsampled (Latency: 154ms). Whereas our approach at 0.5x scale improves recall by 42% over full resolution execution to **61.7** AR (**16.4** AR_S) with latency of 158 ms (+4 ms).

In supplementary, we perform additional comparisons, provide details on handling multiple vanishing points, more tracking results and edge sensing results. We also present some qualitative results and comparisons.

5.1. Ablation Studies

We discuss some of the considerations of our approach through experiments on the Argoverse-HD dataset.

Ground Plane vs Two-Plane Prior: We discussed the rationale of employing multiple planes in Fig 3, and our results are consistent. From Table 5, Two-Plane Prior outperforms Ground Plane prior considerably (**+1.8** AP). Ground Plane Prior outperforms Two-Plane Prior on small objects by **+1** AP_S but is heavily penalized on medium (**-4.1** AP_M) and large objects (**-4.6** AP_L). This is attributed to heavy distortion of tall and nearby objects, and objects that are not on the plane (Figure 3). Lastly, this prior was difficult to learn, the parameter space severely distorted the images (we tuned initialization and learning rate). Thus we did not consider this prior further. The second plane acts as a counterbalance and that warping space is learnable.

Vanishing Point Estimate Dependence: From Table 5, dominant vanishing point in autonomous navigation is highly local in nature, and estimating VP improves the result by **+1.2 AP**. Estimating the vanishing point is a design choice, it's important for safety critical applications like autonomous navigation (performance while navigating turns) however might be omitted for sensing applications.

Method	Scale	AP	AP_S	AP_M	AP_L
Ground Plane Prior	0.5x	29.1	15.5	27.5	48.3
Two-Plane Prior (Psuedo.)	0.5x	27.1	9.8	28.9	50.2
Two-Plane Prior (Avg VP)	0.5x	29.6	12.7	30.7	52.7
Two-Plane Prior	0.5x	30.8	14.5	31.6	52.9

Table 5. **Ablation Study on Argoverse-HD** to justify our design choice of using two planes, dependence on accurate vanishing point detection and choice of pseudo labels vs ground truth.

Using Pseudo Labels vs Ground Truth: Table 5 shows there is still considerable gap (**-3.7** AP) between the Two-Plane Prior trained from pseudo labels and ground truth. We observe that the model under-performs on $stopsign$, $bike$ and $truck$ classes, which are under-represented in the COCO dataset [33] compared to $person$ and car classes. Performance of the pre-trained model on these classes is low even at 1x scale. Hence, we believe that the performance difference is an artifact of this domain gap.

6. Conclusions

In this work, we proposed a learned two-plane perspective prior which incorporates rough geometric constraints from 3D scene interpretations of 2D images to improve object detection. We demonstrated that (a) Geometrically defined spatial sampling prior significantly improves detection performance over multiple axes (accuracy, latency and memory) in terms of both single-frame accuracy and accuracy with real-time constraints over other methods. (b) Not only is our approach more accurate when adaptively down-sampling at all scales, it degrades much more gracefully for small objects, resulting in latency and memory savings. (c) As our prior is learned end-to-end, we can improve a detector's performance at lower scales for "free". (d) Our approach generalizes better to new camera viewpoints and enables efficient city-scale sensing applications. Vanishing point estimation is the bottleneck of our approach [12, 34, 35, 65] for general scenes, and increasing efficiency of its computation we will see substantial improvements. Investigating geometric constraints to improve other aspects of real-time perception systems as future work, like object tracking and trajectory understanding and forecasting, is promising.

Societal Impact Our approach has strong implications for autonomous-driving and city-scale sensing for smart city applications, wherein efficient data processing would lead to more data-driven decision-making and public policies. However, privacy is a concern, and we shall release the datasets after anonymizing people and license plates.

Acknowledgements: This work was supported in part by an NSF CPS Grant CNS-2038612, a DOT RITA Mobility-21 Grant 69A3551747111 and by General Motors Israel.

References

[1] Hala Abualsaud, Sean Liu, David B Lu, Kenny Situ, Akshay Rangesh, and Mohan M Trivedi. Laneaf: Robust multi-lane detection with affinity fields. *Robotics and Automation Letters*, 2021. 5, 11

[2] Erik Bochinski, Volker Eiselein, and Thomas Sikora. High-speed tracking-by-detection without using image information. In *AVSS*, 2017. 7, 13

[3] Federico Camposeco and Marc Pollefeys. Using vanishing points to improve visual-inertial odometry. In *ICRA*, 2015. 5

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 2

[5] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 2

[6] Guobin Chen, Wongun Choi, Xiang Yu, Tony Han, and Manmohan Chandraker. Learning efficient object detection models with knowledge distillation. *NeurIPS*, 2017. 2

[7] Hansheng Chen, Pichao Wang, Fan Wang, Wei Tian, Lu Xiong, and Hao Li. Epro-pnp: Generalized end-to-end probabilistic perspective-n-points for monocular object pose estimation. In *CVPR*, 2022. 2

[8] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 11

[9] 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 *NIPS*, 2015. 2

[10] Ting-Wu Chin, Ruizhou Ding, and Diana Marculescu. Adascale: Towards real-time video object detection using adaptive scaling. *Machine Learning and Systems*, 1:431–441, 2019. 2, 6, 7, 11

[11] Kevin Christensen, Christoph Mertz, Padmanabhan Pillai, Martial Hebert, and Mahadev Satyanarayanan. Towards a distraction-free waze. In *HotMobile*, 2019. 1

[12] James Coughlan and Alan L Yuille. The manhattan world assumption: Regularities in scene statistics which enable bayesian inference. *NeurIPS*, 2000. 8

[13] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 2

[14] Babak Ehteshami Bejnordi, Amirhossein Habibian, Fatih Porikli, and Amir Ghodrati. Salisa: Saliency-based input sampling for efficient video object detection. In *ECCV*, 2022. 2, 4

[15] Andreas Geiger, Martin Lauer, Christian Wojek, Christoph Stiller, and Raquel Urtasun. 3d traffic scene understanding from movable platforms. *TPAMI*, 2014. 2

[16] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 2

[17] Anurag Ghosh, Akshay Nambi, Aditya Singh, Harish YVS, and Tanuja Ganu. Adaptive streaming perception using deep reinforcement learning. *arXiv preprint arXiv:2106.05665*, 2021. 2, 6, 7, 11

[18] Suyog Gupta, Ankur Agrawal, Kailash Gopalakrishnan, and Pritish Narayanan. Deep learning with limited numerical precision. In *ICML*, 2015. 2

[19] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. *NeurIPS*, 2015. 2

[20] Taylor R Hayes and John M Henderson. Scene semantics involuntarily guide attention during visual search. *Psychonomic Bulletin & Review*, 2019. 1

[21] Derek Hoiem, Alexei A Efros, and Martial Hebert. Putting objects in perspective. *IJCV*, 2008. 2, 3

[22] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *CVPR*, 2017. 2, 6

[23] Bret Hull, Vladimir Bychkovsky, Yang Zhang, Kevin Chen, Michel Goraczko, Allen Miu, Eugene Shih, Hari Balakrishnan, and Samuel Madden. Cartel: a distributed mobile sensor computing system. In *SenSys*, pages 125–138, 2006. 1

[24] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. *NeurIPS*, 2015. 2, 3

[25] Ronald Kemker, Marc McClure, Angelina Abitino, Tyler Hayes, and Christopher Kanan. Measuring catastrophic forgetting in neural networks. In *AAAI*, 2018. 13

[26] Sarah Kliff and Soo Oh. America's 4,150 traffic cameras, in one map. `https://www.vox.com/a/red-light-speed-cameras`, 2015. 1

[27] Abhijit Kundu, Yin Li, and James M. Rehg. 3d-rcnn: Instance-level 3d object reconstruction via render-and-compare. In *CVPR*, 2018. 2

[28] Lubor Ladicky, Jianbo Shi, and Marc Pollefeys. Pulling things out of perspective. In *CVPR*, 2014. 2

[29] Mengtian Li, Yu-Xiong Wang, and Deva Ramanan. Towards streaming perception. In *ECCV*, 2020. 2, 5, 6, 7, 11, 13, 15

[30] Xin Li, Zequn Jie, Wei Wang, Changsong Liu, Jimei Yang, Xiaohui Shen, Zhe Lin, Qiang Chen, Shuicheng Yan, and Jiashi Feng. Foveanet: Perspective-aware urban scene parsing. In *ICCV*, 2017. 2

[31] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 2

[32] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 11

[33] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 8

[34] Yancong Lin, Ruben Wiersma, Silvia L Pintea, Klaus Hildebrandt, Elmar Eisemann, and Jan C van Gemert. Deep vanishing point detection: Geometric priors make dataset variations vanish. In *CVPR*, 2022. 5, 8, 11

[35] Shichen Liu, Yichao Zhou, and Yajie Zhao. Vapid: A rapid vanishing point detector via learned optimizers. In *ICCV*, 2021. 5, 8, 11

[36] Yin-Bo Liu, Ming Zeng, and Qing-Hao Meng. D-vpnet: A network for real-time dominant vanishing point detection in natural scenes. *Neurocomputing*, 2020. 5, 11

[37] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 2

[38] Minh Vo N Dinesh Reddy and Srinivasa G. Narasimhan. Carfusion: Combining point tracking and part detection for dynamic 3d reconstruction of vehicle. In *CVPR*, 2018. 2

[39] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *NeurIPS*, 2019. 11

[40] Adria Recasens, Petr Kellnhofer, Simon Stent, Wojciech Matusik, and Antonio Torralba. Learning to zoom: a saliency-based sampling layer for neural networks. In *ECCV*, 2018. 2, 3, 4, 5

[41] N Dinesh Reddy, Robert Tamburo, and Srinivasa G Narasimhan. Walt: Watch and learn 2d amodal representation from time-lapse imagery. In *CVPR*, 2022. 2, 5, 6

[42] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016. 2, 5

[43] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *ICCV*, 2021. 2

[44] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *NeurIPS*, 2015. 2, 5, 11

[45] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *CVPR*, 2019. 5

[46] Edgar Riba, Dmytro Mishkin, Daniel Ponsa, Ethan Rublee, and Gary Bradski. Kornia: an open source differentiable computer vision library for pytorch. In *WACV*, 2020. 11

[47] Chris Rockwell, Justin Johnson, and David F Fouhey. The 8-point algorithm as an inductive bias for relative pose prediction by vits. In *3DV*, 2022. 2

[48] Mahadev Satyanarayanan. The emergence of edge computing. *IEEE Computer*, 2017. 1

[49] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 2

[50] Gur-Eyal Sela, Ionel Gog, Justin Wong, Kumar Krishna Agrawal, Xiangxi Mo, Sukrit Kalra, Peter Schafhalter, Eric Leong, Xin Wang, Bharathan Balaji, et al. Context-aware streaming perception in dynamic environments. In *ECCV*, 2022. 2

[51] Paul Sturgess, Karteek Alahari, Lubor Ladicky, and Philip HS Torr. Combining appearance and structure from motion features for road scene understanding. In *BMVC*, 2009. 2

[52] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *ICCV*, 2015. 2

[53] Patrick Sudowe and Bastian Leibe. Efficient use of geometric constraints for sliding-window object detection in video. In *ICVS*, 2011. 2

[54] Jean-Philippe Tardif. Non-iterative approach for fast and accurate vanishing point detection. In *ICCV*, 2009. 5

[55] Chittesh Thavamani, Mengtian Li, Nicolas Cebron, and Deva Ramanan. Fovea: Foveated image magnification for autonomous navigation. In *ICCV*, 2021. 2, 4, 5, 6, 7, 8, 11, 12, 13

[56] Carl Toft, Daniyar Turmukhambetov, Torsten Sattler, Fredrik Kahl, and Gabriel J Brostow. Single-image depth prediction makes feature matching easier. In *ECCV*, 2020. 2

[57] Antonio Torralba, Aude Oliva, Monica S Castelhano, and John M Henderson. Contextual guidance of eye movements and attention in real-world scenes: the role of global features in object search. *Psychological review*, 2006. 1

[58] U.S. DEPARTMENT OF TRANSPORTATION. Bus profile, bureau of transportation statistics. `https://www.bts.gov/content/bus-profile`, 2022. 1

[59] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, 2001. 2

[60] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *CVPR*, 2019. 2

[61] Jinrong Yang, Songtao Liu, Zeming Li, Xiaoping Li, and Jian Sun. Real-time object detection for streaming perception. In *CVPR*, 2022. 2, 6, 12, 13

[62] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *CVPR*, 2020. 2

[63] Yajie Zhao, Zeng Huang, Tianye Li, Weikai Chen, Chloe LeGendre, Xinglei Ren, Ari Shapiro, and Hao Li. Learning perspective undistortion of portraits. In *ICCV*, 2019. 2

[64] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. In *ECCV*, 2022. 15

[65] Yichao Zhou, Haozhi Qi, Jingwei Huang, and Yi Ma. Neurvps: Neural vanishing point scanning via conic convolution. *NeurIPS*, 2019. 5, 7, 8, 11

[66] Zihan Zhou, Farshid Farhat, and James Z Wang. Detecting dominant vanishing points in natural scenes with application to composition-sensitive image retrieval. *Transactions on Multimedia*, 2017. 11

A. Implementation, Training and Evaluation Details

A.1. Implementation Details

We implemented our approach using Pytorch [39] and mmdetection [8]. The two-plane perspective prior is implemented as a neural network layer with learnable parameters that are global (fixed warps parameterized by vanishing point; Figure 5). We employ differentiable versions of Direct Linear Transform and $warp\_perspective$ from Kornia [46], while we reuse implementation of separable neural warps from [55]. To detect vanishing points, we employ NeurVPS [65] for fixed cameras. We use VPNet [36] with ResNet18 backbone for autonomous navigation.

Parameter Initialization We selected one representative image, and initialized the learnable parameters (i.e. θ's and α's) via visual inspection. The guiding principal was to enlarge far objects while trying to distort the close-by objects as less as possible. The same initial parameters are used for all the datasets. Please look at our code for the initial parameters.

A.2. Evaluation Details

Detection Model Choice We experiment with Faster R-CNN as our base detection model as prior work has shown it occupies the optimal sweet-spot [29] w.r.t latency and accuracy on modern GPUs. However, our approach is agnostic to the choice of detector and our results generalize to other detectors.

Latency: We capture end-to-end latency in milliseconds, that includes image pre-processing, network inference and post-processing, following protocol from prior work [55].

Scale The image down-sampling factor is equal in both spatial dimensions. So an image originally 1920×1200 (1x scale) when down-sampled to 0.25x is a 480×300 image.

A.3. Training Details

For training our proposed approaches, to train the Faster R-CNN model we use the Adam optimizer with a learning rate of 3×10^{-4}. For training any methods by Fovea [55] we follow their protocol. We follow the protocol mentioned by [10, 17] for training their approach.

Argoverse-HD We considered the same base architecture (Faster R-CNN) for all the methods. We compare with SOTA [55] using models provided by their public code release, and follow the training protocol prescribed in their work, training our models for 3 epochs.

WALT We considered the same base architecture (Faster R-CNN) for all the methods. We trained the models (and learnt the warping function parameters, if applicable) using the Adam optimizer with the same learning rate and other parameters for 6 epochs.

Vanishing Point Estimation For NeurVPS, we directly employ the pre-trained model trained on Natural Scenes (TMM17) dataset [66] part of their public code release. While for VPNet [36], as there is no public code release, we implement this architecture employing a ResNet18 backbone attached to a modified YOLO head. We omit the upsampling refinement procedure described in [36], as model's median error in vanishing point prediction is around 10 pixels with an average latency of 28 ms, which is sufficient for our method to work. The off-the-shelf model is executed at $n_v = 30$ to amortize the cost of executing this model. We also tried using LaneAF [1] to obtain lane lines (similar latency), however, we observed the method was prone to errors while clustering lines and obtaining the vanishing point.

B. Multiple Vanishing Points

Our method can consider additional planes that correspond to lines meeting at a different vanishing point. For example, a traffic camera with a wide field of view that is placed at an intersection observing two roads simultaneously would benefit from this. Assuming N vanishing points, considering Saliency S_{v_i} corresponding to vanishing point v_i,

$$S = \sum_{i=0}^{N} \lambda_i S_{v_i} \tag{9}$$

where λ_i's are learnable, initialized as $\frac{1}{N}$. Please observe the case of $N = 2$ in an image from the commuter bus dataset in Figure 7, wherein combining saliencies from two vanishing points (obtained from [34]) ensures far away objects of interest are sampled more.

We observed in the datasets we considered, multiple vanishing points were rare as it generally requires a camera with a large field of view. Thus, we employed models [36, 65] trained on Natural Scenes dataset, which predict only one vanishing point. However, vanishing points can be estimated from other methods [34, 35, 65] which do predict all the vanishing points, but incur higher overheads.

C. Results on Another Detector

Adaptive spatial sampling mechanisms leverage and exploit priors corresponding to the input images in a way that is agnostic to the detection method. We expect our approach to generalize across detectors, similar to observations by such warping mechanisms and saliency priors proposed earlier [55]. We choose RetinaNet [32], a popular single-stage object detector as our archetypal example (Faster R-CNN [44] is the two-stage archetype). Results can be viewed in Table 6. Our approach improves upon both the baseline Faster R-CNN and SOTA, specially for small and medium sized objects, following the trends observed in the main manuscript.

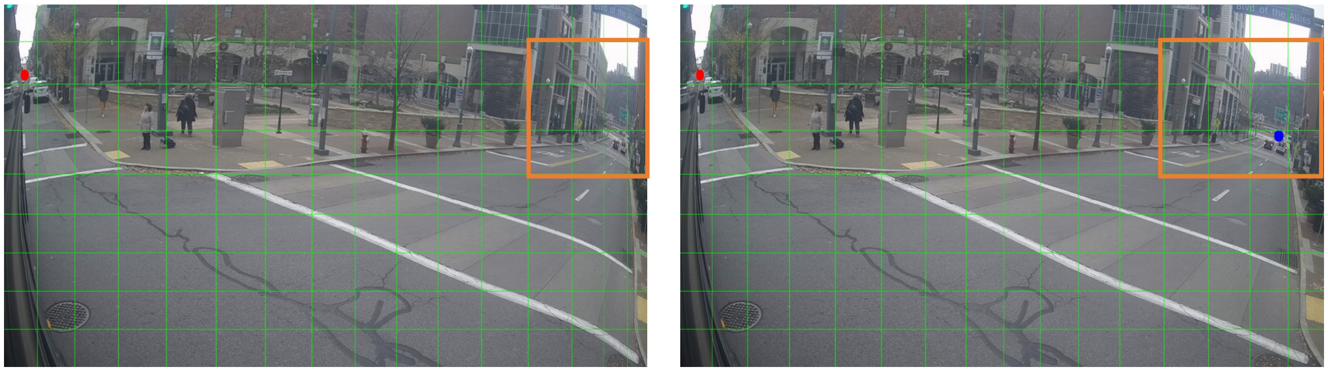

Figure 7. **Multiple Vanishing Points:** Saliency from First Vanishing Point parallel to sidewalk "compresses" far away cars on perpendicular road. Second Vanishing Point ensures those cars are not compressed. Please zoom in to observe the vanishing points and deformation.

Method	Scale	AP	AP_S	AP_M	AP_L
RetinaNet	0.5x	22.6	4.0	22.0	53.1
Fovea (S_I) [55]	0.5x	24.9	7.1	27.7	50.6
Two-Plane Prior	0.5x	**26.3**	**10.1**	**29.2**	50.5
Baseline at higher scales					
RetinaNet	0.75x	29.9	9.7	32.5	54.2

Table 6. **Alternate Detector:** We replace Faster R-CNN with RetinaNet (archetypal one-stage detector), and observe considerable improvements over Baseline (RetinaNet with uniform downsampling) and SOTA trained on Argoverse-HD dataset.

Method	Scale	Model	AP	AP_S	AP_M	AP_L
Faster R-CNN	0.5x	COCO	15.3	1.1	12.5	40.5
Faster R-CNN	0.5x	AVHD	15.1	1.0	10.6	39.0
Fovea (S_D) [55]	0.5x	AVHD	13.7	1.3	10.0	34.7
Fovea (S_I) [55]	0.5x	AVHD	16.4	2.1	12.8	38.6
Two-Plane Prior (Psuedo.)	0.5x	AVHD	16.2	**4.7**	**15.9**	33.3
Two-Plane Prior (Psuedo.)	0.5x	COCO	**20.9**	**5.8**	**19.4**	**44.2**
Baseline at higher scales						
Faster R-CNN	0.75x	AVHD	19.7	3.0	16.1	44.2
Faster R-CNN	0.75x	COCO	20.3	3.7	18.2	45.3
Faster R-CNN	1x	AVHD	22.6	5.7	20.1	45.7
Faster R-CNN	1x	COCO	23.1	6.5	21.7	46.1

Table 7. **Generalization to BDD100K:** Scale in this case is fixed to 0.5x, AVHD refers to Argoverse-HD and COCO datasets respectively. AVHD models are finetuned from the pre-trained COCO model. We compare generalization on the BDD100K dataset. Our method assumes availability of training set images of BDD100K **and not labels**, we generate pseudo-labels from the available model (Section 3.6) to learn the Two Plane prior.

D. Additional Results on Autonomous Driving

We shall consider the comparisons made in Section 5 on Argoverse-HD in the main manuscript and present some additional results (specially across different categories present in the dataset).

Method	sAP	sAP_S	sAP_M	sAP_L
StreamYOLO-L [61]	25.9	8.6	24.2	40.9
StreamYOLO-M [61]	25.9	9.2	24.8	41.0
StreamYOLO-S [61]	29.6	11.0	30.9	51.6
Ours	**30.0**	**13.7**	**31.5**	**52.2**

Table 8. **"Real-Time" Detectors**: Streaming Comparison on Argoverse-HD on Titan X. StreamYOLO-M and StreamYOLO-L single-frame latency is 45.8 ms and 62.9 ms respectively, is greater than 33ms, violating [61]'s "real-time" restriction. StreamYOLO-S satisfies (**20.8 ms**), hence has better performance.

Comparison with Learned Fovea [55]: Fovea [55] also proposed end-to-end global, dataset-wide saliency map S learned via backpropagation (Learned Seperable and Learned Nonseperable). However, they observed worse performance compared to their bounding box priors (S_D and S_I), see Table 9. We show that end-to-end learned saliency is better, with careful geometric parameterization.

Improved Performance on ground plane: We observe improved performance over state-of-the-art on every object category for objects on the ground plane (person, traffic light, bike, stop-sign, car, truck) apart from motorbike (See Table 9). On further observation, this might be an artifact of the label skew of the Argoverse-HD dataset ($mbike$ has the least number of instances). For objects not on the ground plane, like traffic-light, we observe performance as good as SOTA.

Generalization to BDD100K: We compare generalization from approaches trained on Argoverse-HD and COCO datasets to BDD100K MOT dataset (See Table 7). Our approach assumes access to training set images from BDD100K dataset, but not it's ground truth labels. As our priors can be learnt without access to ground truth data, we employ the method detailed in Section 3.6 to generate pseudo labels to learn and adapt the geometric param-

	Method	Scale	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	person	mbike	tffclight	bike	bus	stop	car	truck
	Faster R-CNN (Pre.)	0.5x	21.5	35.8	22.3	2.8	22.4	50.6	20.8	9.1	13.9	7.1	48.0	16.1	37.2	20.2
	Faster R-CNN	0.5x	24.2	38.9	26.1	4.9	29.0	50.9	22.8	7.5	23.3	5.9	44.6	19.3	43.7	26.6
	Fovea (Learned Nonsep.) [55]	0.5x	25.9	42.9	26.5	10.0	28.4	48.5	25.2	11.9	20.9	7.1	39.5	25.1	49.4	28.1
	Fovea (Learned Sep.) [55]	0.5x	27.2	44.8	28.3	12.2	29.1	46.6	24.2	14.0	22.6	7.7	39.5	31.8	50.0	27.8
SOTA	Fovea (S_D) [55]	0.5x	26.7	43.3	27.8	8.2	29.7	54.1	25.4	13.5	22.0	8.0	45.9	21.3	48.1	29.3
	Fovea (S_I) [55]	0.5x	28.0	45.5	29.2	10.4	31.0	**54.5**	27.3	16.9	**24.3**	9.0	44.5	23.2	50.5	28.4
	Fovea (L:S_I) [55]	0.5x	28.1	45.9	28.9	10.3	30.9	54.1	27.5	**17.9**	23.6	8.1	45.4	23.1	50.2	28.7
Ours	Ground Plane Prior	0.5x	**29.1**	**46.2**	**30.5**	**15.5**	27.5	48.3	**28.6**	15.0	10.4	**10.5**	33.7	**46.2**	**55.4**	**32.9**
	Two-Plane Pr. (Psuedo.)	0.5x	27.1	43.4	28.4	9.8	28.9	50.2	**28.2**	16.2	20.6	8.1	**50.8**	26.1	45.2	21.7
	Two-Plane Pr. (Avg VP)	0.5x	**29.6**	**45.9**	**31.6**	**12.7**	**30.7**	52.7	**28.4**	14.3	**24.3**	**11.9**	38.5	**31.6**	**53.9**	**34.2**
	Two-Plane Prior	0.5x	**30.8**	**47.2**	**33.2**	**14.5**	**31.6**	52.9	**30.0**	16.7	24.1	**13.7**	35.9	**35.9**	**55.3**	**35.3**
	Baseline at higher scales															
	Faster R-CNN	0.75x	29.2	47.6	31.1	11.6	32.1	53.3	29.6	12.7	30.8	7.9	44.1	29.8	48.8	30.1
	Faster R-CNN	1.0x	33.3	53.9	35.0	16.8	34.8	53.6	33.1	20.9	38.7	6.7	44.7	36.7	52.7	32.7

Table 9. **Evaluation on the Argoverse-HD dataset:** This is an expanded version of the Table present in the manuscript. We see improvements for most of the objects that are on the ground, with much better overall performance for both small and medium sized objects along with improvements in AP_{50} and AP_{75}. Notice that Two-Plane prior performs at par with SOTA on "traffic-light" category. Pre. denotes Pretrained model, while Psuedo. denotes model trained with Psuedo-Labels from pretrained model (no access to Argoverse-HD labels) as described in Section 3.6. *We have bolded all of our method variations that perform better than SOTA.*

eters on this dataset by training on these pseudo-labels for 1 epoch only.

We observe that our method nearly matches SOTA when adapted starting from a model finetuned on Argoverse-HD, however, dramatically exceeds it's performance when adapted from a model solely trained on COCO. We believe the reason for this mismatch is due to catastrophic forgetting [25] observed in finetuned models when evaluated on out-of-distribution data. Lastly, the results indicate the benefits of learnability of our perspective prior, we observe increase in performance for "free" even when images are available without access to ground truth.

Comparison with "Real-Time" Detectors: Real-time detectors like [61] have been recently proposed which predict boxes G_{t+1} at time t (of frame F_{t+1}; available solely during training and not testing) given F_t to satisfy sAP. Approaches like these constraint the detector to perform the computation within a latency budget ($<$ 33ms or 30 FPS). Our methods are complementary to such detectors, as long as *their* constraint is satisfied.

However, real-time detectors (termed as "fast" strategy) might be suboptimal [29]. Satisfying the real-time detector constraint may not be optimal for every hardware platform, specially on slower edge devices. Such methods [61] are **not** hardware-agnostic, and model architecture choices are optimized for specific hardware (in their case, for a V100 GPU). On Titan X (See Table 8), their streaming performance (which is hardware dependent) is worse.

E. Tracking Smaller Objects for Longer

We provide an analysis of our approach observing how it improves object tracking. We wish to observe if the gains from our method translates to detecting far-away objects for longer period of time. We employ Argoverse-HD dataset for our experiments which have ground truth object IDs.
Setup: We employ a Faster R-CNN as our baseline and the tracker is fixed to IOU Tracker [2]. We additionally pair the priors proposed by Fovea [55] for comparison. All the detectors are executed at 0.5x scale for fair comparison.
Tracking Visualizations: We present some tracking visualization in Figures 8 and 9. These visualizations motivate us to define the following metrics.
Detecting and Tracking for Longer: We wish to understand if Two-Plane Prior is able to detect an object for a longer lifespan. This is important in autonomous driving situations, wherein we want to detect far-away objects as quickly as possible or any object moving away from us.
Prior tracking quality metrics such as MT% and ML% check the ratio of tracks that are mostly tracked or mostly lost. However, this does not capture the track length improvements. We propose to compare the average extension of a track (**ATE**) compared to the baseline detection method. Given a track τ, E_τ can be positive or negative, and is given by,

$$E_\tau(m, b, gt) = (L_m - L_b)/L_{gt} \tag{10}$$

where m is the method, b is baseline and gt is the ground truth track, while L denotes track length. **ATE** is the average over tracks across all sequences. However, as the met-

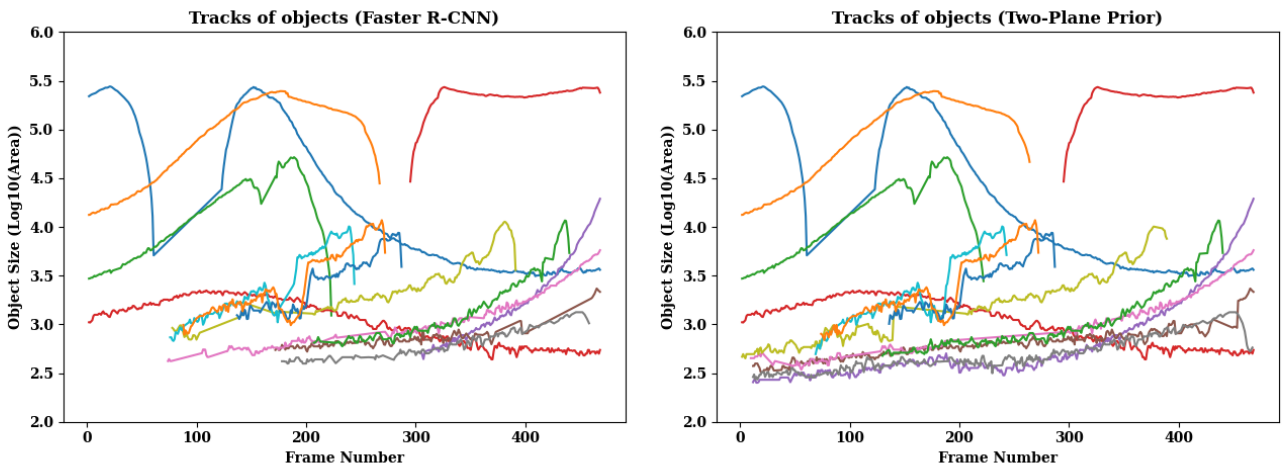

Figure 8. **Tracking Visualization:** To visualize the impact of our two-plane prior, we visualize tracks of length greater than 150 frames tracked by both the methods for a given sequence. We plot object size w.r.t frame numbers (which denotes length). We can observe that some objects are detected earlier and are tracked for a longer time.

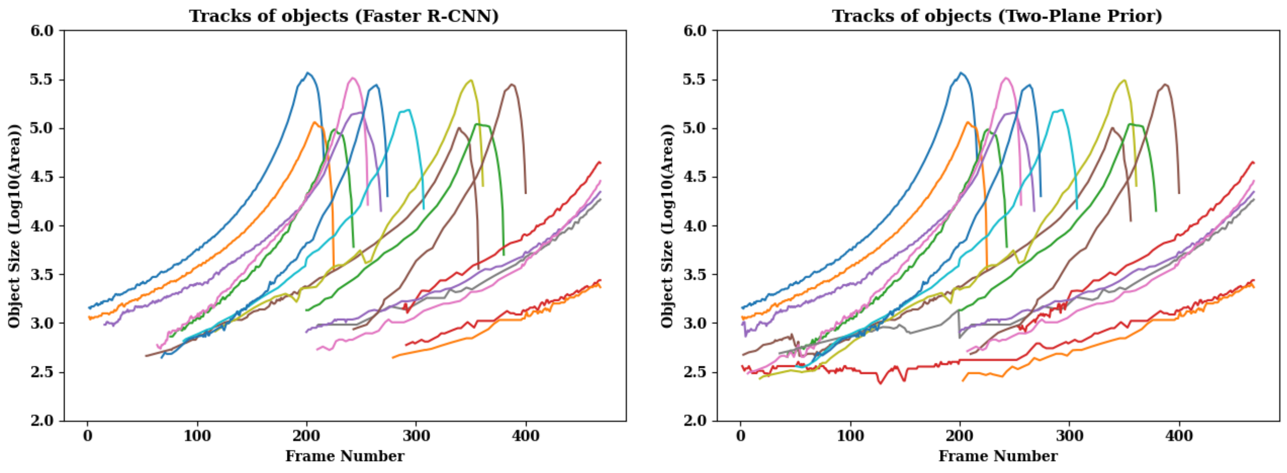

Figure 9. **Tracking Visualization:** To visualize the impact of our two-plane prior, we visualize tracks of length greater than 150 frames tracked by both the methods for a given sequence. We plot object size w.r.t frame numbers (which denotes length). The severe drops of the object sizes for some tracks correspond to nearby object overtaken by our vehicle. We can observe that some objects are detected earlier and are tracked for a longer time.

ric weighs all tracks equally, which is unfair for extremely small track lengths, thus, we only consider ground truth tracks which are atleast 5 seconds or 150 frames long.

Detecting and Tracking Smaller Objects: Given a track, we wish to observe if Two-Plane Prior is able to detect an object when it's "smaller" compared to other methods. This is important in autonomous driving situations, wherein we would like to detect further away objects, which would appear smaller. We wish to compute the minimum object size tracked (**MOS**). We employ a proxy for object size, $size(x) = log(area(x))$ where x denotes an object bounding box, as the area quadratically increases. For a given ground truth track τ, let o_τ denote minimum object size of an object, while O_τ denotes maximum object size. Let c_τ denote the minimum object size in the predicted track currently considered. We can write,

$$M_\tau = \frac{c_\tau - o_\tau}{O_\tau - o_\tau} \tag{11}$$

M_τ is averaged over all tracks across all sequences to obtain **MOS**.

Method	Scale	AP_{50}	AR	AR_S	AR_M	AR_L	Latency (ms)
Faster R-CNN	0.5x	45.0	31.7	0.5	37.3	46.9	154 ± 8.5
Two-Plane Prior	0.5x	**77.2**	**61.7**	**16.4**	**69.9**	**68.7**	158 ± 7.5
Faster R-CNN	0.75x	58.6	41.1	10.5	45.3	54.7	240 ± 8.5
Two-Plane Prior	0.75x	**84.5**	**68.3**	**38.8**	**72.9**	**73.3**	245 ± 10
			Baseline at higher scales				
Faster R-CNN	1x	68.2	41.5	16.9	47.5	57.1	350 ± 15

Table 10. **Rare Object Detection on the Commuter Bus:** We compare our approach with a baseline Faster R-CNN. We observe improved precision and recall over the baseline, specially for small and medium sized objects. Do note, for ≈1FPS throughput over five simultaneous streams, average latency of 200ms should be achieved (however, this is not an enforced latency budget for streaming perception [29]).

F. Detection on the Commuter Bus

The bus is equipped with a Jetson AGX edge device. The edge device communicates with a modified onboard-NVR recording bus data from 7 cameras, two inside the bus and five on the outside of the bus. The cameras record data at $5FPS$ at 720P resolution for 8 working hours of the bus, totalling 1.08 million frames everyday. It is not feasible to transmit and process this data on the cloud due to bandwidth and compute limitations, and privacy concerns. Thus, the edge device and the NVR are part of a distributed edge-cloud infrastructure wherein the edge device is employed to process these simultaneous streams, only relevant frames are transmitted to cloud machines where we do further offline analysis.

We analyze bus streams to build an actionable map of public infrastructure, for instance, which areas need a trash pickup or where does snow needs to be shovelled. We also provide real-time feedback to the bus driver, informing them of people who may need assistance (say, on wheelchairs, or with a stroller or service animal) getting on the bus. Thus we employ an object detector to detect trash cans, garbage bags and people with an assistive device. Our system has to operate at near real-time on all streams simultaneously, rendering cloud-transmission-turn-around infeasible.

As we employ the edge device to filter out relevant frames, detecting all the objects in the scene is more important than the precision and localization accuracy (a frame once, marked "relevant", is sent to cloud where we employ larger models at higher resolutions without constraints).

Dataset Acquisition: For research purposes, we do record all the data[1], which is humongous (≈30 Terabytes till now) and the instances are rare, we were able to identify 3.5K such frames (temporally subsampled to 750) through a semi-automatic method. Firstly, we only sampled frames from the camera that is facing the sidewalk (people entering the bus are visible). We then geo-fenced images from bus-stop locations and major intersections on the bus route reducing the set to 780K images. Then, we employ off-the-shelf Detic Swin-B Large Faster R-CNN with CLIP (for custom vocabulary) [64] and find images with "wheelchair", "stroller", "walker", "crutches", "cane", "dog", "animal", "trolley", "cart", "trash can", "garbage bin", "garbage", "garbage bag" categories with a confidence threshold of 0.25. This model has a high false positive rate for these rare classes, and we were able to automatically filter a set of 21K images, and manually filtered these to yield 3.5K images. As many of these images were part of dense temporal sequences, we further sub-sampled temporally within each sequence yielding 750 samples. We manually annotated these images with object bounding boxes and categories ("trash-can", "garbage-bag" and "person-requiring-assistance"; labels from Detic [64] were not accurate). As the data is recorded over the course of a year, we split the train and test test (70% - 30%) using the date stamp (images taken on the same day are in the same split) so that the model doesn't overfit.

Hardware Platform: We set the Jetson AGX to consume 30+ Watts (MAXN configuration; no power budget). Memory is measured using the `tegrastats` utility, while we use Jetpack 4.6.1 and pytorch 1.6, mmdetection 2.7 (+ mmcv 1.15) compiled for Jetson AGX to measure latency consistently across methods (models can be compiled with TensorRT and trained with mixed precision for additional orthogonal improvements).

Results: In this case, just like autonomous driving, we observe that the vanishing point is highly local. Due to overheads of vanishing point estimate on our edge device, we instead employ the average vanishing point, and cache saliency S, considerably reducing our approach's latency and memory while maximizing accuracy. From Table 10, we observe AR and mAP_{50} for the baseline (Faster R-CNN) and our approach at 0.5x and 0.75x scales. Our method consistently outperforms the baseline method at the same scale, showing both better precision and recall while incurring only **4ms** additional latency and **22 MB** memory overheads.

G. Qualitative Results

We present the variations of our proposed Two-Plane Perspective Prior across different datasets and scenarios in Figure 10. We also show case of the major failure mode of just employing Ground Plane Prior in Fig 11. We also show a qualitative comparison with prior work in Figures 12, 13 and 14. Lastly, we take a closer look at some of the far away objects that were detected in Figures 15 and 16. The accompanying website further illustrates some of the aspects of our method.

[1]Transmission is infeasible, HDD's swapped physically. (Sneakernet)

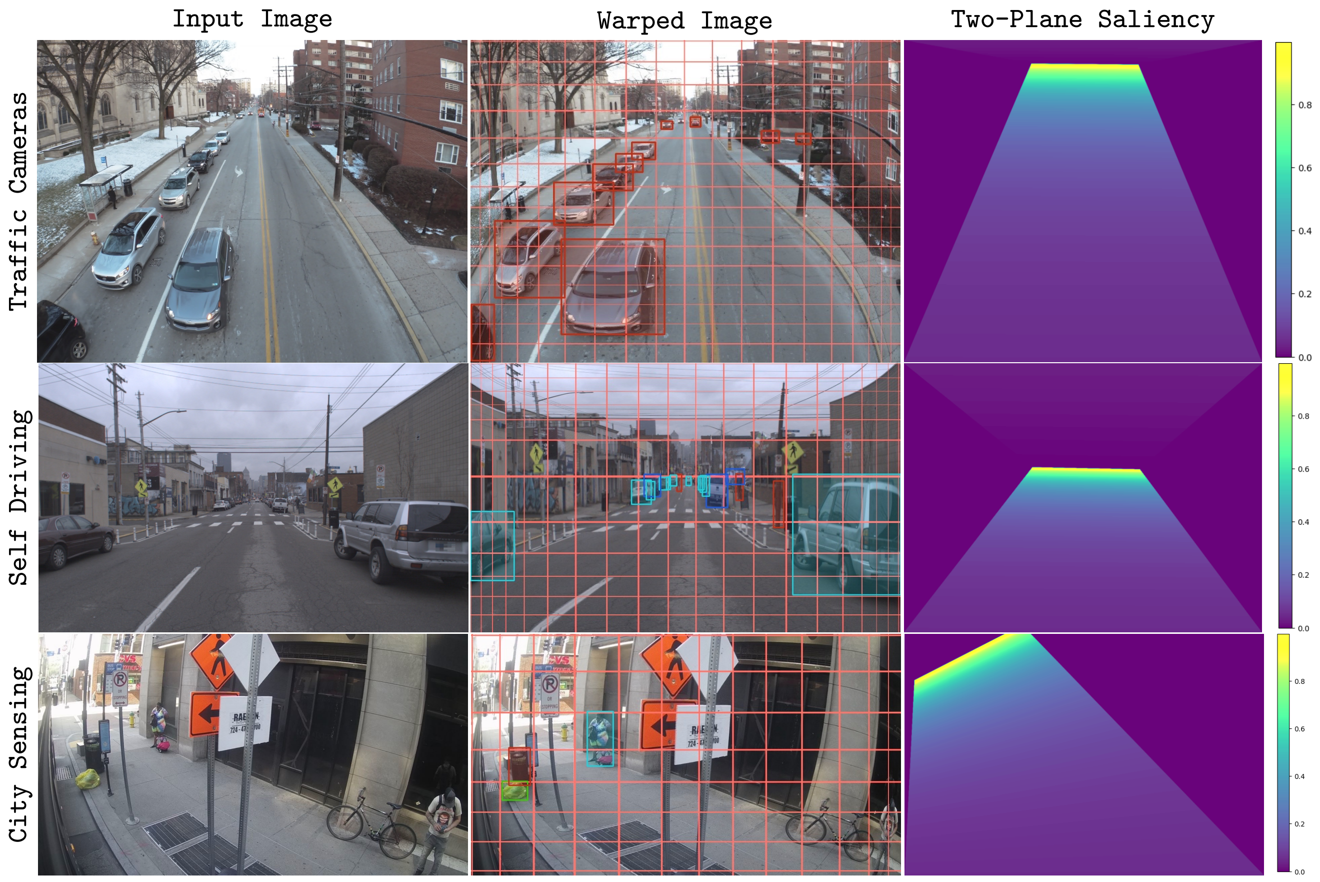

Figure 10. **Two-Plane Prior Based Warping:** Two-Plane Prior is defined by a few parameters that describe two planar regions in the direction of the vanishing point in the 3D scene (See Section 3.1 and 3.2 in manuscript). Firstly, we can observe the Two-Plane Prior's explicit dependence on the vanishing point v in the saliency maps. Next, as we can observe from grid lines (equidistant in the original image) overlaid on top of the warped images, the extent of spatial warping varies across datasets (WALT, Argoverse-HD and Commuter Bus), showing us the need for learnable parameter ν over prior work which do not directly model this relationship. Lastly, notice the second plane's effect in sampling. The second plane acts as a "counter-balance" to reduce distortion, and the plane is **faintly observable** (contrast adjusted for better visibility).

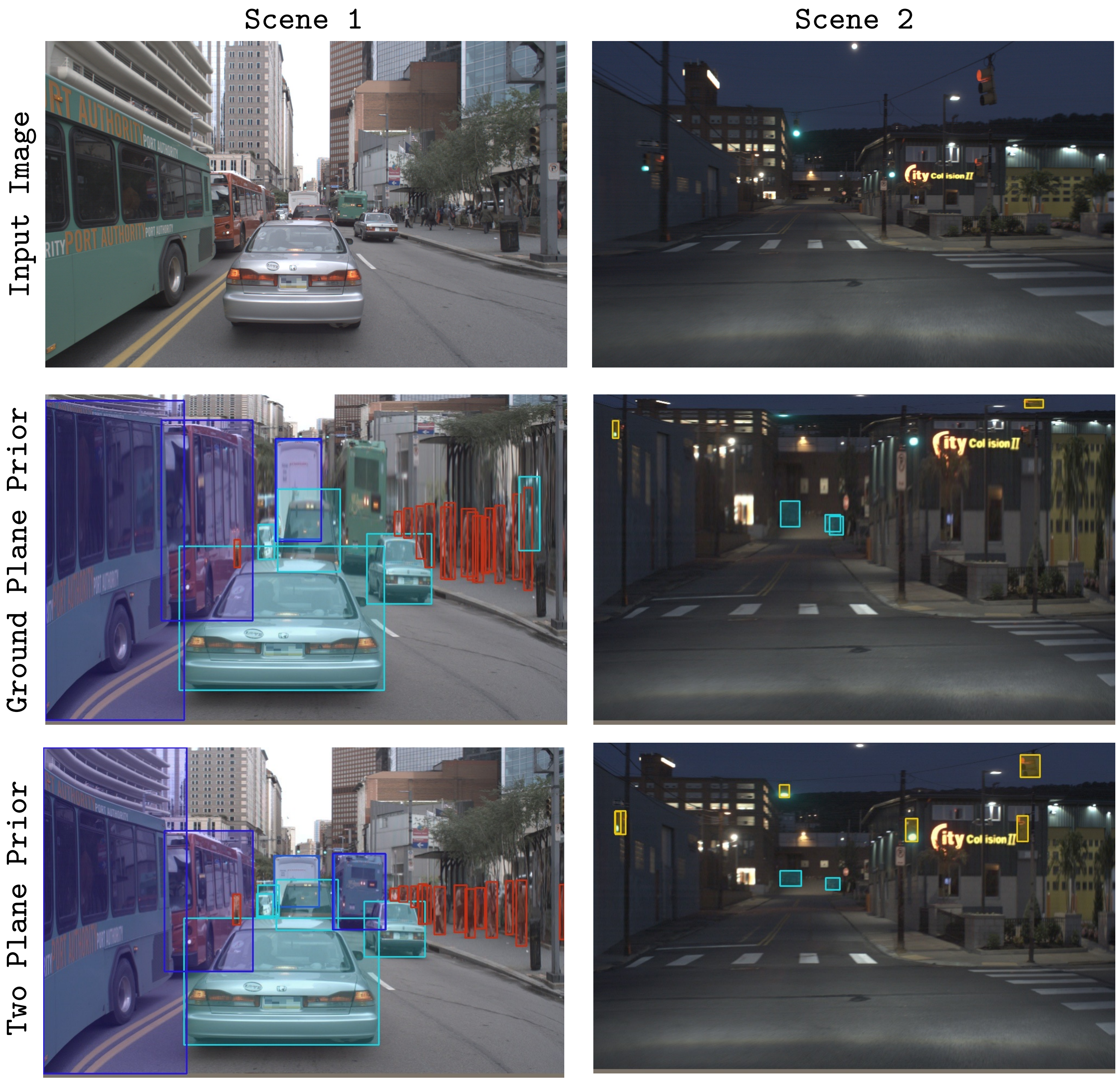

Figure 11. **Ground Plane Prior vs Two-Plane Prior:** This figure demonstrates how crucial it is to model the second plane. Learning is difficult with Ground Plane Prior (Section 5.1 in the manuscript) and causes heavy distortion of non-ground-plane regions.
Scene 1: Detector with Ground Plane Prior misses nearby tall objects because of heavy distortion. Turquoise colored bus on the right (blue box) is detected when Two-Plane prior is used and missed with Ground Plane prior.
Scene 2: Objects not on the ground plane are missed as they are squished by the Ground Plane Prior. Yellow boxes denote the traffic lights. All 6 traffic lights in the scene were detected when Two-Plane prior is used while Ground Plane prior missed 4 traffic lights.

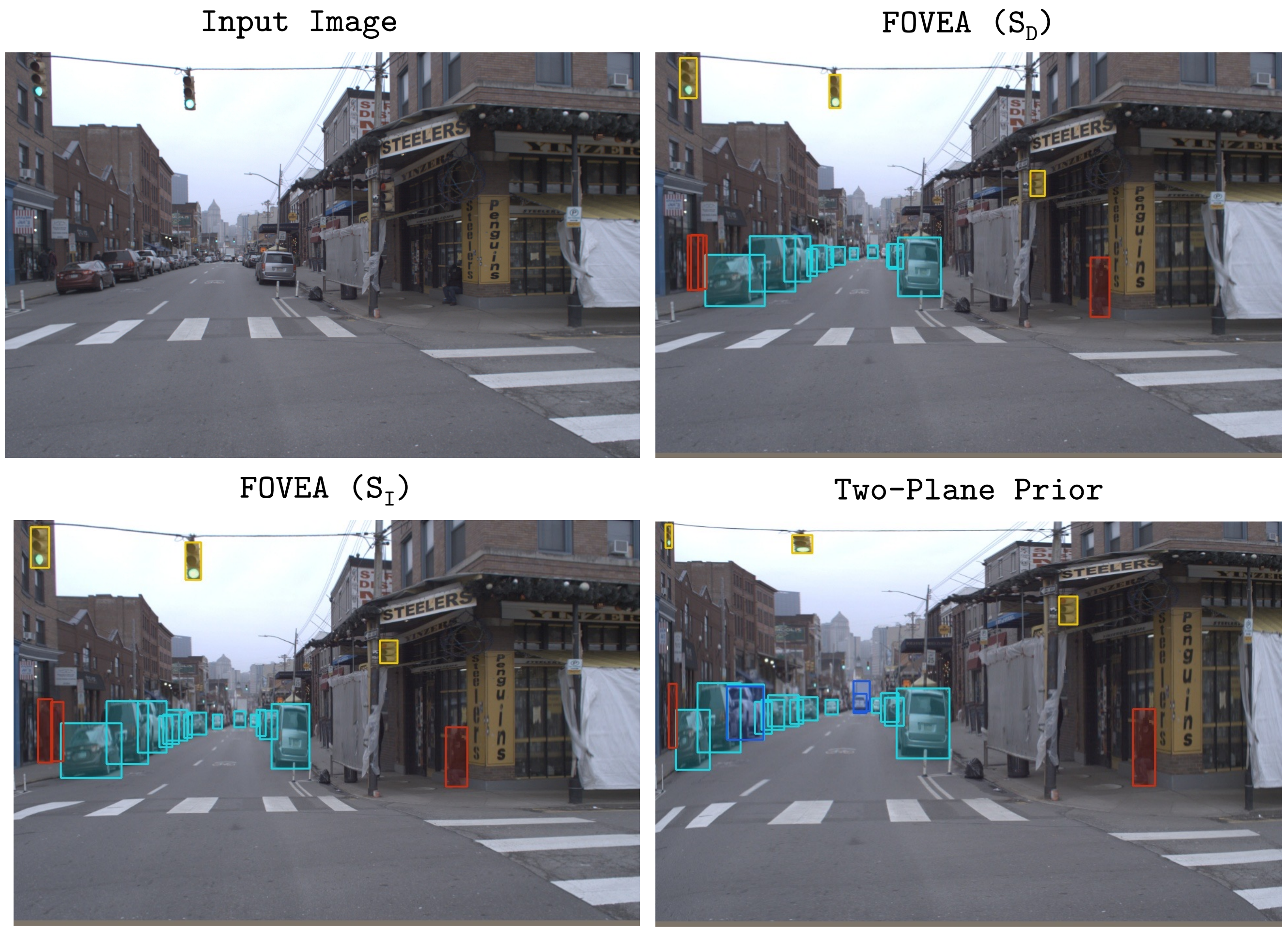

Figure 12. **Qualitative Comparison with Fovea Warps on Argoverse-HD:** We observe that the reliance on the vanishing point v allows the warp to sample in the direction of the road even while making turns. Far ahead on the road, a $truck$ (dark-blue) is not detected by Fovea (S_D or S_I), but correctly detected by our approach.

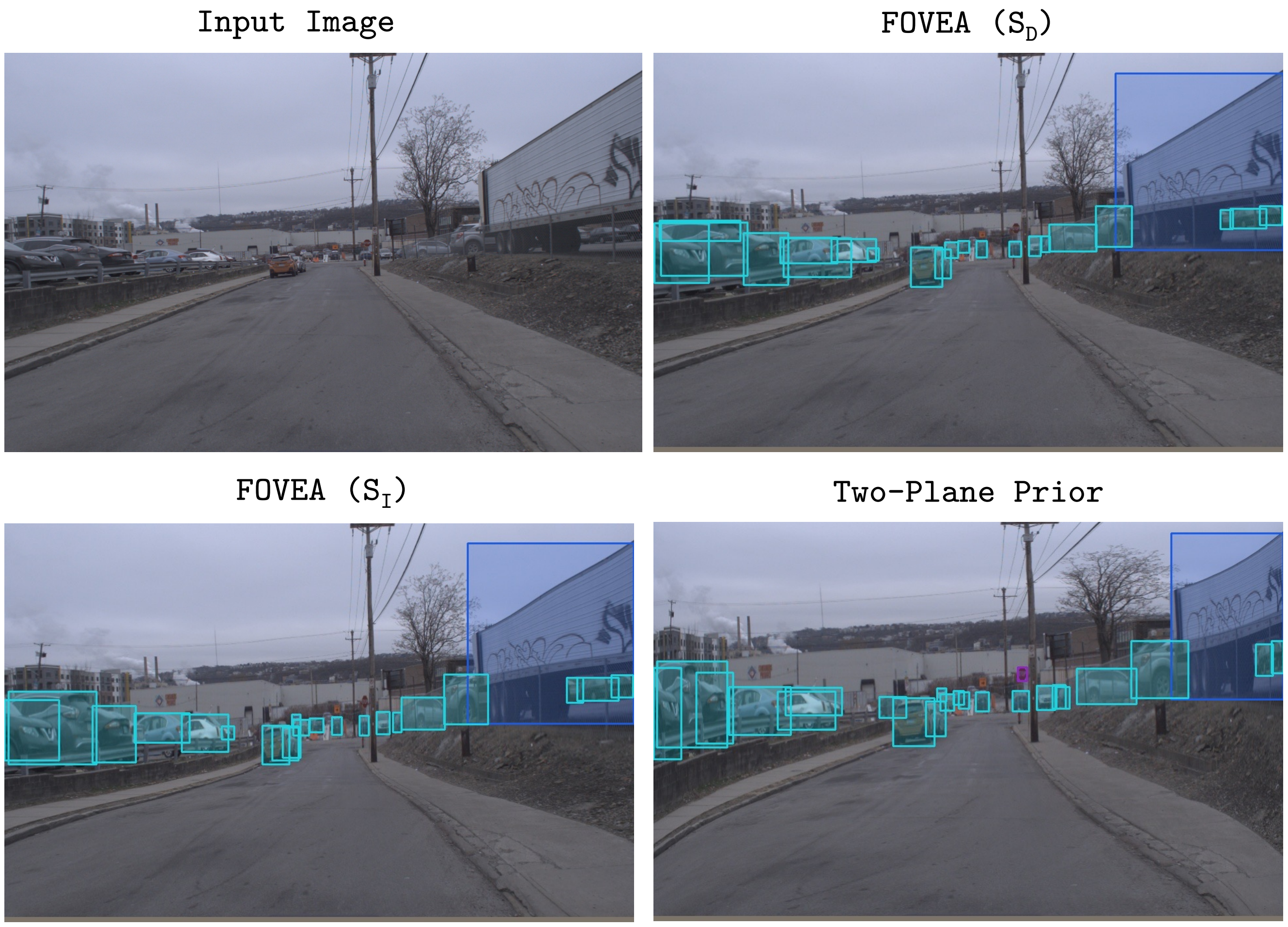

Figure 13. **Qualitative Comparison with Fovea Warps on Argoverse-HD:** We observe that scale factor ν models the extent of sampling better. Fovea (S_D or S_I) misses the $stop - sign$ (a magenta box in the middle of the image) which our method is able to detect (as it's larger in the warped image). Fovea (S_I) notes that in their method, regions immediately adjacent to magnified regions are often contracted which is noticed in this case.

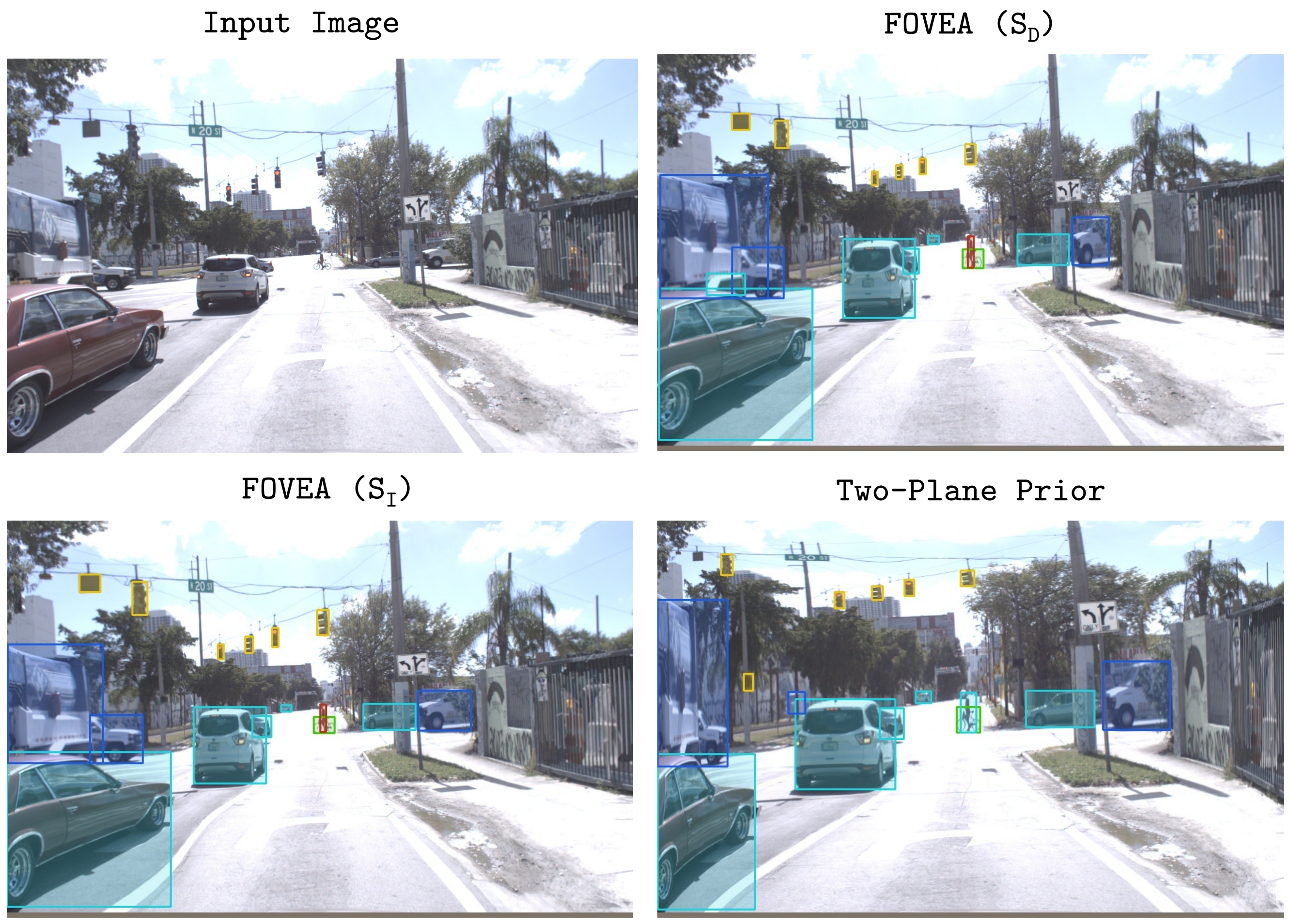

Figure 14. **Qualitative Comparison with Fovea Warps on Argoverse-HD:** *Failure Case:* The model has misclassified a pedestrian as car in the image warped by the Two-Plane Prior while correctly classified by Fovea (S_D or S_I) (red), likely due to the presence of bicycle and heavier distortion.

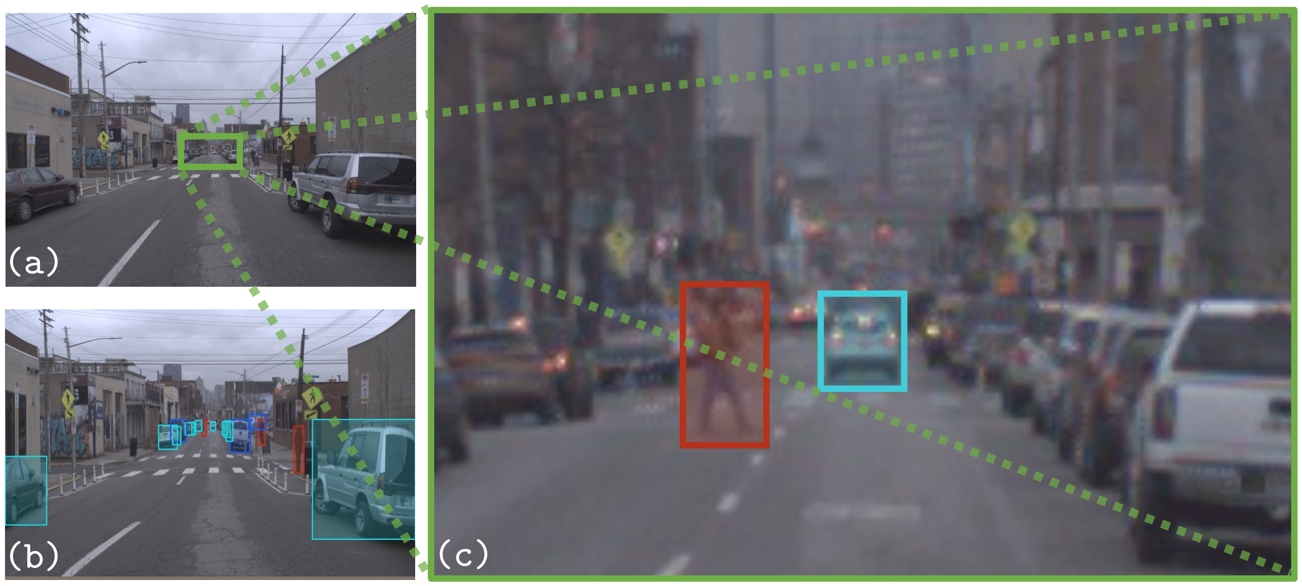

Figure 15. **Detection of Far Away Objects:** Our Two-Plane Prior boosts the detection of small far-away objects at lower resolutions (depicted image from Argoverse-HD dataset). The cropped green region in the (a) original image is (c) zoomed in while (b) shows all the detections in the warped image. Our method detects far-away pedestrian and car.

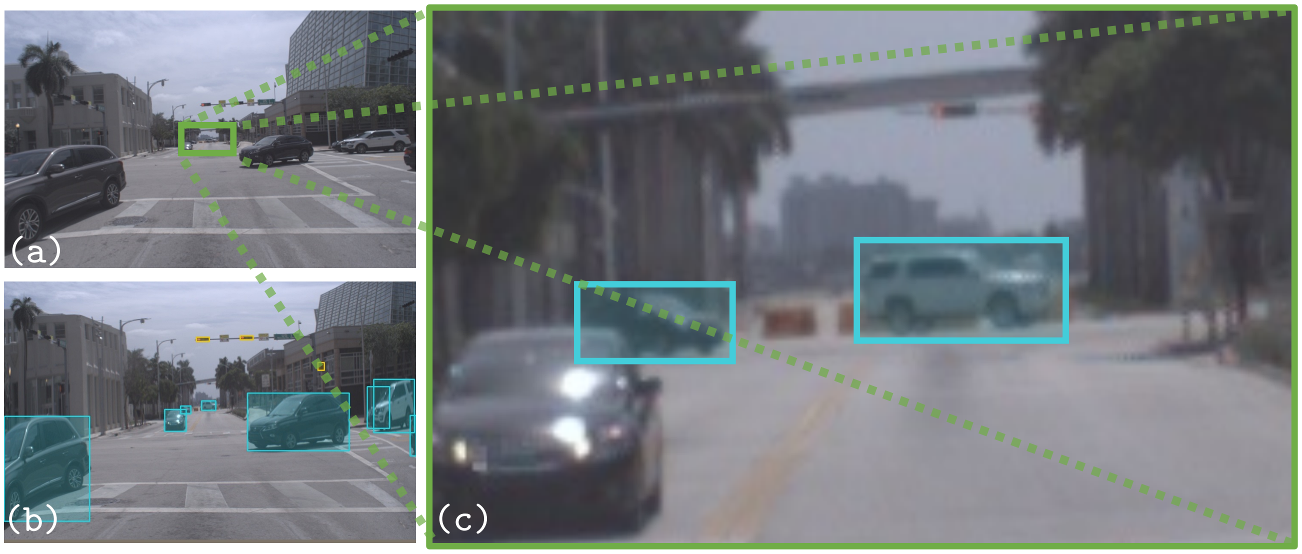

Figure 16. **Detection of Far Away Objects:** Our Two-Plane Prior boosts the detection of small far-away objects at lower resolutions (depicted image from Argoverse-HD dataset). The cropped green region in the (a) original image is (c) zoomed in while (b) shows all the detections in the warped image. Our method is able to detect the occluded car.