Are We Ready for Vision-Centric Driving Streaming Perception? The ASAP Benchmark

Xiaofeng Wang[1,2] Zheng Zhu[3] Yunpeng Zhang[3] Guan Huang[3]
Yun Ye[3] Wenbo Xu[3] Ziwei Chen[4] Xingang Wang[1*]
[1]Institute of Automation, Chinese Academy of Sciences
[2]University of Chinese Academy of Sciences   [3]PhiGent Robotics   [4]Southeast University

Abstract

*In recent years, vision-centric perception has flourished in various autonomous driving tasks, including 3D detection, semantic map construction, motion forecasting, and depth estimation. Nevertheless, the latency of vision-centric approaches is too high for practical deployment (e.g., most camera-based 3D detectors have a runtime greater than 300ms). To bridge the gap between ideal researches and real-world applications, it is necessary to quantify the trade-off between performance and efficiency. Traditionally, autonomous-driving perception benchmarks perform the **offline** evaluation, neglecting the inference time delay. To mitigate the problem, we propose the **A**utonomous-driving St**reA**ming **P**erception (ASAP) benchmark, which is the first benchmark to evaluate the **online** performance of vision-centric perception in autonomous driving. On the basis of the 2Hz annotated nuScenes dataset, we first propose an annotation-extending pipeline to generate high-frame-rate labels for the 12Hz raw images. Referring to the practical deployment, the **S**treaming **P**erception **U**nder const**R**ained-computation (SPUR) evaluation protocol is further constructed, where the 12Hz inputs are utilized for streaming evaluation under the constraints of different computational resources. In the ASAP benchmark, comprehensive experiment results reveal that the model rank alters under different constraints, suggesting that the model latency and computation budget should be considered as design choices to optimize the practical deployment. To facilitate further research, we establish baselines for camera-based streaming 3D detection, which consistently enhance the streaming performance across various hardware. ASAP project page:* https://github.com/JeffWang987/ASAP.

1. Introduction

Vision-centric perception in autonomous driving has drawn extensive attention recently, as it can obtain richer
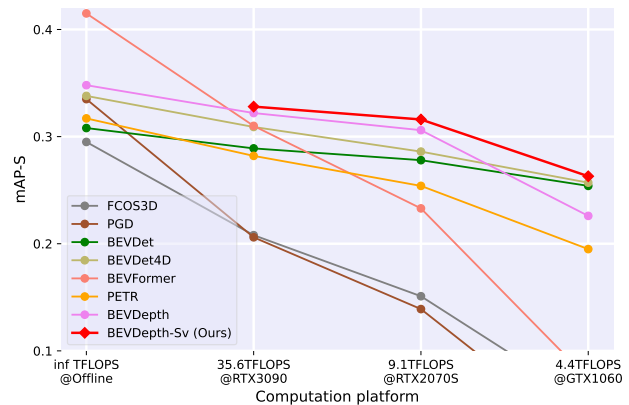

Figure 1. Comparison of streaming performances on the ASAP benchmark, where the model rank changes under different computational resources. Note that our baseline BEVDepth-Sv (built upon [30]) consistently improves the streaming performance on different platforms.

semantic information from images with a desirable budget, compared to LiDAR-based perception. Notably, the past years have witnessed the blooming of vision-centric perception in various autonomous driving tasks (*e.g.*, 3D detection [21, 22, 29–31, 33, 34, 41, 56, 61], semantic map construction [28, 40, 43, 45, 64, 72], motion forecasting [1, 20], and depth estimation [16, 17, 55, 57, 58, 60]).

Despite the growing research interest in vision-centric approaches, the high latency of these methods still prevents the practical deployment. Specifically, in the fundamental task of autonomous-driving perception (*e.g.*, 3D detection), the inference time of most camera-based 3D detectors [21, 29–31, 34, 61, 70] is longer than 300ms (on the powerful NVIDIA RTX3090), which is ∼6× longer (see Tab. 1) than the LiDAR-based counterparts [25, 62, 66]. To enable practical vision-centric perception in autonomous driving, a quantitative metric is in an urgent need to balance the accuracy and latency. However, previous autonomous-driving benchmarks [3, 4, 7, 12, 13, 23, 38, 46, 49, 59, 67] mainly fo-

---

Table 1. Comparison between autonomous-driving perception dataset, where *L&C* represents LiDAR and camera, *#sensors* denotes number of sensors, *Ann. frequency* is the annotation frequency, and *Model speed* denotes the typical inference speed of the model on RTX3090. For 2D detectors [2, 11, 15, 44, 50], they achieve ∼45mAP@30FPS on COCO [32]. For LiDAR-based 3D detectors [25, 62, 66], they achieve ∼70mAP@20FPS on Waymo [49]. For camera-based 3D detectors [21, 30, 31, 70], they achieve ∼40mAP@3FPS on nuScenes [4], which is 6×∼10× slower than the above two tasks.

Dataset	Stream.	Modality	#sensors	Task	Ann. frequency	Model speed
KITTI [12]	✗	L&C	-	Multi-task	-	-
Argoverse [59]	✗	L&C	-	Multi-task	-	-
Waymo [49]	✗	L&C	-	Multi-task	-	-
nuScenes [4]	✗	L&C	-	Multi-task	-	-
Argoverse-HD [27]	✓	C	1	2D det.	30Hz	∼30FPS
Waymo [18]	✓	L	1	L-3D det.	10Hz	∼20FPS
nuScenes-H	✓	C	6	C-3D det.	12Hz	∼3FPS

cus on the **offline** performance metrics (*e.g.*, Average Precision (AP), Truth Positive (TP)), and the model latency has not been particularly studied. Although [18, 27] leverage the streaming perception paradigm [27] to measure the accuracy-latency trade-off, these benchmarks are designed for 2D detection or LiDAR-based 3D detection.

To address the aforementioned problem, this paper proposes the **A**utonomous-driving **S**tre**A**ming **P**erception (ASAP) benchmark. To the best of our knowledge, this is the first benchmark to evaluate the online performance of vision-centric perception in autonomous driving. The ASAP benchmark is instantiated on the camera-based 3D detection, which is the core task of vision-centric perception in autonomous driving. To enable the streaming evaluation of 3D detectors, an annotation-extending pipeline is devised to increase the annotation frame rate of the nuScenes dataset [4] from 2Hz to 12Hz. The extended dataset, termed nuScenes-H (High-frame-rate annotation), is utilized to evaluate 3D detectors with 12Hz streaming inputs. Referring to the practical deployment, we delve into the problem of ASAP under different computational resources. Specifically, the **S**treaming **P**erception **U**nder const**R**ained-computation (SPUR) evaluation protocol is constructed: (1) To compare the model performance on varying platforms, multiple GPUs with different computation performances are assigned for the streaming evaluation. (2) To analyze the performance fluctuation caused by the sharing of computational resources [9, 61, 65, 70], the streaming evaluation is performed while the GPU is simultaneously processing other perception tasks. As depicted in Fig. 1, the streaming performances of different methods drop steadily as the computation power is increasingly constrained. Besides, the model rank alters under various hardware constraints, suggesting that the offline performance cannot serve as the deterministic criterion for different approaches. Therefore, it is necessary to introduce our streaming paradigm to vision-

centric driving perception. Based on the ASAP benchmark, we further establish simple baselines for camera-based streaming 3D detection, and experiment results show that forecasting the future state of the object can compensate for the delay in inference time. Notably, the proposed BEVDepth-Sv improves the streaming performance (mAP-S) by ∼2%, ∼3%, and ∼16% on three GPUs (RTX3090, RTX2070S, GTX1060).

The main contributions are summarized as follows: (1) We propose the ASAP benchmark to quantitatively evaluate the accuracy-latency trade-off of camera-based perception methods, which takes a step towards the practical vision-centric perception in autonomous driving. (2) An annotation-extending pipeline is proposed to annotate the 12Hz raw images of the popular nuScenes dataset, which facilitates the streaming evaluation on camera-based 3D detection. (3) Simple baselines are established in the ASAP benchmark, which alleviates the influence of inference delay and consistently improves the streaming performances across different hardware. (4) The SPUR evaluation protocol is constructed to facilitate the evaluation of practical deployment, where we investigate the streaming performance of the proposed baselines and seven modern camera-based 3D detectors under various computational constraints.

2. Related Work

2.1. Autonomous-Driving Benchmark

Thanks to the release of various benchmarks [3, 4, 7, 12, 13, 23, 38, 46, 49, 59, 67, 68], the last decade has witnessed immense progress on autonomous-driving perception. Among these benchmarks, CamVid [3], Cityscapes [7], Mapillary Vistas [38], Apolloscape [23], BDD100K [67] and CityPerson [68] focus on the 2D annotation (segmentation masks or detection boxes). To facilitate 3D perception in autonomous driving, several benchmarks [4,5,12, 13, 42, 46, 49] collect multi-modal data (RGB images, LiDAR, RADAR, GPS/IMU) with comprehensive 3D annotations. Among these 3D annotated dataset, nuScenes [4], Waymo [49], Argoverse [5], A2D2 [13], Lyft L5 [46] provide surround-view image data, which boosts camera-based 3D perception. Especially in the nuScenes dataset [4], a vision-centric 3D perception trend [21,22,29–31,33,34,41, 56, 61] demonstrates that camera-based 3D detectors can achieve promising accuracy. Nevertheless, these benchmarks evaluate perception methods in an offline manner, neglecting the inference time delay.

2.2. Vision-Centric Driving Perception

Compared with the costly LiDAR, cameras can be deployed with much lower budgets. Besides, cameras-based methods own the desirable merits to extract rich semantic information from dense color and texture information

[26, 37], which facilitates versatile vision-centric perception in autonomous driving. *e.g.*, 3D detection [8, 35, 52–54, 69], semantic map construction [36, 40, 47], and depth estimation [16, 17, 55, 57, 58, 60]. Recently, the Bird's Eye View (BEV) representation further promotes the development of vision-centric perception [1, 20–22, 28–31, 33, 34, 41, 43, 45, 56, 61, 64, 72]. Particularly, in the fundamental task of autonomous driving (*e.g.*, 3D detection), BEVDet4D [21], BEVFormer [31], PETRv2 [34], BEVDepth [30], BEVStereo [29], STS [56], SOLOFusion [41] have achieved promising detection accuracy, approaching that of LiDAR-based counterparts [25, 51, 62, 66, 71]. However, even on the powerful RTX3090, the runtime of most methods exceeds 300ms, which is far from practical deployment.

2.3. Streaming Perception

The concept of streaming perception is first proposed in [27], where a benchmark is introduced to evaluate the accuracy-latency trade-off of 2D detectors. Faced with model latency, Kalman filter [24], dynamic scheduling [27], and reinforcement learning [14] are utilized to alleviate the problems caused by inference time delay. To further enhance the streaming performance, [63] simplifies the streaming perception to the task of *predicting the next frame* by an efficient detector [11]. To investigate the streaming perception in LiDAR-based 3D detection, [18] proposes to split the full-scan LiDAR points into multiple slices and process the streaming LiDAR slices with a recurrent neural network. Following the same setting, [10] preserves past slice features and concatenates them with current slice data. [6] regards that LiDAR slices can be naturally represented in polar coordinates and the polar-pillar representation is utilized in the streaming perception. The aforementioned streaming benchmarks [18, 27] are designed for 2D detection or LiDAR-based 3D detection, but the streaming paradigm of vision-centric perception in autonomous-driving (e.g., camera-based 3D detection) is still under investigation.

3. The ASAP benchmark

In this section, the concept of ASAP is first introduced. Then we analyze the difficulty to evaluate streaming algorithms on the original nuScenes dataset [4], and introduce the high frame-rate nuScenes-H dataset. Subsequently, the SPUR evaluation protocol is presented to assess the streaming performance under different computational resources. Finally, we propose simple baselines to alleviate the inference time delay in streaming detection.

3.1. Autonomous-Driving Streaming Perception

Referring to the streaming paradigm [27], the ASAP benchmark conducts the evaluation in an online manner, and the key insight is to perform the evaluation at *every*
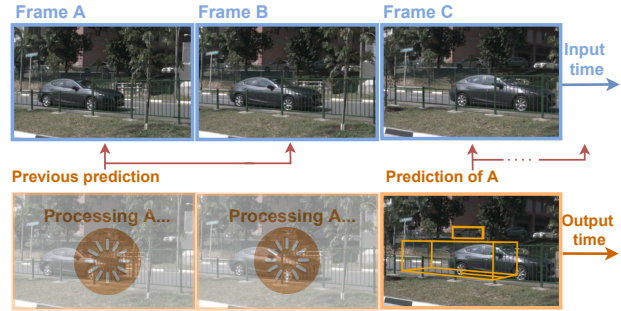

Figure 2. Illustration of the streaming evaluation in the ASAP benchmark. For *every* input timestamp, the ASAP benchmark evaluates the most recent prediction if the processing of current frame is not finished.

input timestamp even if the processing of the current sample is not complete. Specifically, given streaming inputs $\{X_i\}_{i=1}^{T}$, where X_i is the surround-view images at timestamp t_i and T is the total number of input timestamps. The perception algorithms are required to make an online response to the input instance, and the entire online predictions are $\{\hat{Y}_j\}_{j=1}^{M}$, where $\hat{Y}_j$ is the prediction at timestamp t_j, and M represents the total number of predictions. Notably, the prediction timestamps are not synchronized with the input timestamps, and the model inference speed is typically slower than the input frame rate (*i.e.*, $M < T$). To evaluate the predictions at input timestamp t_i, the ground truth Y_i is desired to match with the most recent prediction, yielding the pair $(Y_i, \hat{Y}_{\theta(i)})$, where $\theta(i) = \arg\max_{j} t_j < t_i$. Based on the matching strategy, the ASAP benchmark evaluates the online performance at *every* input timestamp:

$$\mathcal{L}_{\text{ASAP}} = \mathcal{L}(\{(Y_i, \hat{Y}_{\theta(i)})\}_{i=1}^{T}), \tag{1}$$

where $\mathcal{L}(\cdot)$ is the streaming evaluation metric, which will be elaborated in Sec. 3.3. The streaming evaluation is illustrated in Fig. 2, where the prediction of frame A is leveraged for evaluation at the timestamp of frame C. For frame A and frame B, the previous predictions are evaluated. Notably, ASAP instantiates the streaming paradigm on camera-based 3D detection, and the key insights also generalize to other vision-centric perceptions in autonomous driving.

3.2. nuScenes-H

The nuScenes dataset [4] is a popular autonomous-driving perception benchmark, which significantly facilitates the vision-centric perception trend [21, 22, 29–31, 33, 34, 41, 56, 61]. Consequently, it is natural to leverage the nuScenes dataset to investigate the streaming performance of various camera-based 3D detectors. However, the annotation frame rate of the original nuScenes dataset is 2Hz, which is slower than the inference speed of the majority of
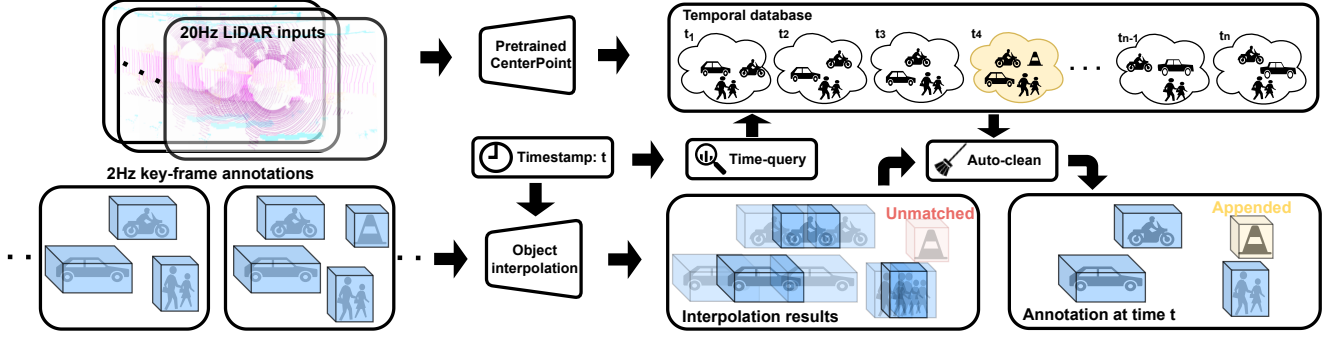
Figure 3. Overall architecture of the annotation-extending pipeline, where the 12Hz annotations are calculated by object interpolation of 2Hz key-frames, and the temporal database is established to append annotations that are missed by interpolation.
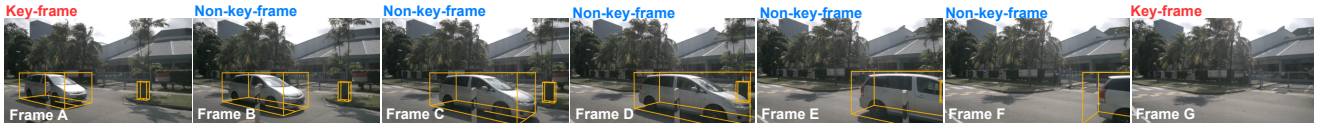

Figure 4. Visualization of the 12Hz annotated nuScenes-H dataset, where frame A and frame G are key-frames with the original 2Hz annotations, and B ∼ F are non-key-frames with 12Hz annotations.

camera-based 3D detectors. Therefore, it can hardly distinguish across models with different latencies. To mitigate the problem, we take advantage of the 12Hz raw images of the nuScenes dataset as the streaming input. Specifically, an annotation-extending pipeline is proposed to annotate the 12Hz raw images. The overall architecture is illustrated in Fig 3. Given 2Hz key-frame annotations at time t_s and t_e, we can calculate the intermediate annotation at time t ($t_s < t < t_e$) using the object interpolation:

$$Tr(t) = \frac{t_e - t}{t_e - t_s} Tr(t_s) + \frac{t - t_s}{t_e - t_s} Tr(t_e),$$
$$R(t) = \mathcal{F}_s(R(t_s), R(t_e), \frac{t_e - t}{t_e - t_s}), \tag{2}$$

where $Tr(t)$ and $R(t)$ represent the object translation and rotation at time t, respectively. Notably, to avoid *Gimbal Lock*, we employ the quaternion representation for rotation $R(t)$, and $\mathcal{F}_s$ denotes the *Spherical Linear Interpolation* [48]. Thanks to the instance token in the nuScenes-devkit [39], we can match the corresponding objects in the sequential key-frames, which facilitates object interpolation. However, the interpolation is ignored when the object is not co-visible in sequential key-frames, thus the intermediate annotations are missed. To mitigate the problem, the temporal database $\{(t_i, Y_i^{\mathcal{L}})\}_{i=1}^n$ is established, where the predicted bounding boxes[1] $\{Y_i^{\mathcal{L}}\}_{i=1}^n$ at 20Hz input timestamps $\{t_i\}_{i=1}^n$ are stored. Then we can query the prediction

Y_{query} at time t from the temporal database:

$$Y_{\text{query}} = Y_j^{\mathcal{L}}(j = \arg\min_i |t_i - t|). \tag{3}$$

Subsequently, the auto-clean is performed by an Intersection over Union (IoU) matching between the interpolated annotation and Y_{query}, which filters the redundant predictions in Y_{query}. And the left predictions are appended to the final annotations at time t. The visualization of 12Hz annotations is shown in Fig 4, where the vehicle is visible in the key-frame A, but not in the key-frame G. Hence, the intermediate annotations are ignored under the interpolation. However, such issue can be remedied by the temporal database, from which the accurate annotations are appended to the intermediate frames. Equipped with the annotation-extending pipeline, 1M training images and 0.2M validation images are annotated in the nuScenes-H dataset. Notably, we benchmark seven 3D detectors [21, 22, 30, 31, 33, 53, 54] on the original nuScenes dataset and the extended nuScenes-H dataset, and the Pearson correlation coefficient of the two offline scores is 0.962, which validates the effectiveness of our annotation-extending method. The comparison between nuScenes-H and other autonomous-driving perception datasets [4, 12, 18, 27, 49, 59] is shown in Tab. 1. To the best of our knowledge, nuScenes-H is the first dataset that facilitates the streaming evaluation on camera-based 3D detection.

3.3. SPUR Evaluation Protocol

Considering that the offline evaluation protocol in nuScenes [4] cannot be directly adapted to the streaming system, we design the Streaming Perception Under

---

[1]We firstly train the CenterPoint [66] on 2Hz LiDAR key-frames of the nuScenes `trainval` set, then the 3D bounding boxes of 20Hz LiDAR inputs are predicted using the CenterPoint.

constRained-computation (SPUR) evaluation protocol to comprehensively investigate the streaming performance of various 3D detectors. In the next, the streaming metrics are first introduced, then the computation-constrained evaluation is elaborated.

Streaming metrics. Average Translation Error (ATE), Average Scale Error (ASE), Average Orientation Error (AOE), Average Velocity Error (AVE), Average Attribute Error (AAE), NuScenes Detection Score (NDS) and mean Average Precision (mAP) are the official metrics in the original nuScenes dataset. These metrics can be naturally extended to the streaming metric (Eq. 1) except for the AVE. In the streaming evaluation, the predicted bounding boxes are displaced from the ground-truth locations due to inference time delay, especially for fast-moving objects. Consequently, the majority of the anticipated true positives are slow-moving or static objects, and the AVE metric only measures the velocity error of true positive objects, which makes the streaming velocity error even lower than the offline velocity error. To address the issue, we calculate AVE as the original offline metric, and other metrics are measured with the streaming evaluation, which are termed mAP-S, ATE-S, ASE-S, AOE-S, and AAE-S. For the NDS-S, following [4], we calculate it as:

$$ \text{NDS-S} = \frac{1}{10}[5\text{mAP-S} + \sum_{\text{mTP} \in \mathbb{TP}} (1 - \min(1, \text{mTP}))], \quad (4) $$

where $\mathbb{TP} = \{\text{AVE, ATE-S, ASE-S, AOE-S, AAE-S}\}$ is the set of true positive metrics.

Computation-constrained evaluation. Notably, in the ASAP benchmark, the model inference time is associated with computational resources, which influences streaming performances. Specifically, two computation-constrained evaluation protocols are investigated:

- To compare the streaming performance on varying platforms, multiple GPUs with different performances (*e.g.*, NVIDIA RTX3090, NVIDIA RTX 2070S, and NVIDIA GTX 1060) are assigned to evaluate 3D detectors.

- To analyze the performance fluctuation caused by computational resources sharing [9, 61, 65, 70], we evaluate 3D detectors while the GPU is simultaneously processing other perception tasks (*e.g.*, conducting N classification tasks using ResNet18 [19]).
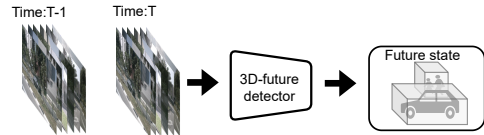
3.4. ASAP Baselines

As mentioned in Sec. 3.1, the ASAP benchmark evaluates the most recent predictions if the current computation is not finished, resulting in a mismatch between the previously processed observation and the current one. In this subsection, we discuss how to mitigate the mismatch problem induced by the inference time delay. Naturally, forecasting the future state emerges as a simple solution to compensate for the delay. To make future state estimations, we establish the velocity-based baseline that updates future states by the predicted object motion. Besides, we investigate a learning-based baseline that directly estimates the future locations of objects (illustrated in Fig. 5).
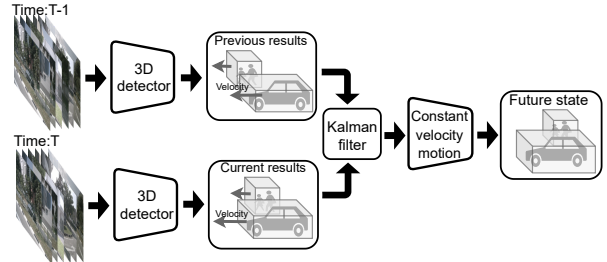
(a) Velocity-based updating baseline, where the Kalman filter is utilized to associate and refine multi-frame detection results, and the future state is predicted by the constant velocity motion model.

(b) Learning-based forecasting baseline, where the future state is directly estimated by the 3D-future detector.

Figure 5. Illustration of the proposed ASAP baselines.

Velocity-based updating baseline. Object velocity estimation is an essential task in the original nuScenes benchmark, and various 3D detectors [21, 29–31, 34, 41, 56] have been investigated to produce accurate velocity estimations. We empirically find that simply using the *constant velocity motion model* can benefit streaming 3D detection:

$$ Tr(t_{i+1}) = Tr(t_i) + (t_{i+1} - t_i)V(t_i), \quad (5) $$

where $Tr(\cdot)$ and $V(\cdot)$ represent the predicted object translation and velocity, and t_i, t_{i+1} denote the previous input timestamp and the current evaluation timestamp. Such a velocity-based updating strategy is straightforward, but the velocity estimations are independent in each frame, neglecting that the predicted velocity should be consistent and change smoothly. To alleviate the problem, we associate predictions in consecutive frames and refine the predictions using the first-order Kalman filter [24]. Specifically, IoU-based greedy matching is applied to associate 3D bounding boxes across frames. And state representation in the Kalman filter is $\{x, y, z, \dot{x}, \dot{y}\}$, where (x, y, z) denotes the center location of the bounding box, and $\dot{x}, \dot{y}$ is the estimated velocity in the BEV plane. The updated state representations are leveraged to refine per-frame predictions. For objects that do not have correspondence in sequential frames, we simply use the *constant velocity motion model*

Table 2. Comparison of different camera-based 3D detectors on the nuScenes-H `val` set, where the BEVDepth-Sv is the velocity-based updating baseline built upon [30], and BEVDepth-Sf is the learning-based forecasting baseline built upon [30]. For *Streaming=*✗, we report the 2Hz offline metrics. For *Streaming=*✓, we report the streaming performance on the 12Hz ASAP benchmark.

Methods	GPU	FPS	#params.	GFLOPs	Streaming	mAP(-S)↑	NDS(-S)↑	ATE(-S)↓	ASE(-S)↓	AOE(-S)↓	AVE↓	AAE(-S)↓
FCOS3D	-	-	52.5M	2008.2	✗	0.295	0.372	0.806	0.268	0.511	1.315	0.170
FCOS3D	RTX3090	1.7	52.5M	2008.2	✓	0.208	0.326	0.828	0.269	0.512	1.315	0.175
FCOS3D	RTX2070s	0.8	52.5M	2008.2	✓	0.151	0.294	0.836	0.270	0.522	1.315	0.187
FCOS3D	GTX1060	0.3	52.5M	2008.2	✓	0.051	0.234	0.858	0.271	0.585	1.315	0.200
PGD	-	-	51.3M	2223.0	✗	0.335	0.409	0.732	0.263	0.423	1.285	0.172
PGD	RTX3090	1.6	51.3M	2223.0	✓	0.206	0.327	0.817	0.273	0.488	1.285	0.185
PGD	RTX2070s	0.7	51.3M	2223.0	✓	0.139	0.289	0.818	0.276	0.512	1.285	0.195
PGD	GTX1060	0.2	51.3M	2223.0	✓	0.016	0.199	0.909	0.342	0.536	1.285	0.300
BEVDet	-	-	52.6M	215.3	✗	0.308	0.411	0.729	0.265	0.445	1.051	0.175
BEVDet	RTX3090	12.6	52.6M	215.3	✓	0.289	0.370	0.730	0.273	0.533	1.051	0.209
BEVDet	RTX2070s	8.5	52.6M	215.3	✓	0.284	0.367	0.734	0.273	0.536	1.051	0.209
BEVDet	GTX1060	3.3	52.6M	215.3	✓	0.254	0.348	0.751	0.275	0.547	1.051	0.218
BEVDet4D	-	-	53.6M	222.0	✗	0.338	0.476	0.672	0.274	0.460	0.337	0.185
BEVDet4D	RTX3090	11.9	53.6M	222.0	✓	0.309	0.450	0.755	0.275	0.480	0.337	0.200
BEVDet4D	RTX2070s	6.9	53.6M	222.0	✓	0.286	0.438	0.757	0.275	0.481	0.337	0.201
BEVDet4D	GTX1060	3.2	53.6M	222.0	✓	0.257	0.419	0.775	0.276	0.492	0.337	0.211
BEVFormer	-	-	68.7M	1322.2	✗	0.415	0.517	0.672	0.274	0.369	0.397	0.198
BEVFormer	RTX3090	2.4	68.7M	1322.2	✓	0.310	0.452	0.760	0.276	0.385	0.397	0.216
BEVFormer	RTX2070s	1.1	68.7M	1322.2	✓	0.233	0.408	0.774	0.278	0.410	0.397	0.228
BEVFormer	GTX1060	0.3	68.7M	1322.2	✓	0.074	0.311	0.819	0.280	0.516	0.397	0.246
PETR	-	-	36.7M	297.2	✗	0.317	0.367	0.839	0.280	0.614	0.936	0.232
PETR	RTX3090	6.7	36.7M	297.2	✓	0.282	0.341	0.883	0.288	0.639	0.936	0.249
PETR	RTX2070s	3.2	36.7M	297.2	✓	0.254	0.323	0.897	0.289	0.658	0.936	0.258
PETR	GTX1060	1.3	36.7M	297.2	✓	0.195	0.291	0.918	0.291	0.659	0.936	0.266
BEVDepth	-	-	76.6M	662.6	✗	0.348	0.481	0.616	0.272	0.415	0.440	0.196
BEVDepth	RTX3090	8.6	76.6M	662.6	✓	0.323	0.464	0.654	0.272	0.414	0.440	0.198
BEVDepth	RTX2070s	4.4	76.6M	662.6	✓	0.306	0.453	0.664	0.273	0.420	0.440	0.205
BEVDepth	GTX1060	1.4	76.6M	662.6	✓	0.226	0.404	0.686	0.275	0.449	0.440	0.235
BEVDepth-Sv	RTX3090	8.6	76.6M	662.6	✓	0.328	0.466	0.654	0.272	0.416	0.440	0.198
BEVDepth-Sv	RTX2070s	4.4	76.6M	662.6	✓	0.316	0.459	0.662	0.272	0.419	0.440	0.198
BEVDepth-Sv	GTX1060	1.4	76.6M	662.6	✓	0.263	0.428	0.683	0.273	0.436	0.440	0.199
BEVDepth-Sf	RTX3090	8.6	76.6M	662.6	✓	0.329	0.467	0.653	0.272	0.415	0.440	0.197
BEVDepth-Sf	RTX2070s	4.4	76.6M	662.6	✓	0.313	0.457	0.663	0.272	0.420	0.440	0.198
BEVDepth-Sf	GTX1060	1.4	76.6M	662.6	✓	0.235	0.413	0.685	0.274	0.442	0.440	0.205

to update the locations. Notably, the velocity-based updating strategy can be applied to any modern 3D detector (*e.g.*, in the experiment, BEVDepth-Sv is built upon BEVDepth [30]). Besides, the updating pipeline is lightweight (∼10ms on CPU), which has a negligible impact on streaming delays.

Learning-based forecasting baseline. The above baseline exploits velocity as the intermediate surrogate to predict future states. From another perspective, the streaming 3D detector should be inherently predictive of the future. Therefore, we craft a simple framework for directly forecasting the future locations of objects. Specifically, a 3D-future detector is built upon BEVDepth [30] (termed BEVDepth-Sf), where the algorithm leverages history frames as input and forecasts the detection results of the **next frame**. The model architecture and training strategy are similar to those of [30], except that the loss is calculated using annotations from the subsequent frame. For samples that do not have

annotations for future frames, we discard them in the training phase.

4. Streaming Evaluation on ASAP Benchmark

In this section, the experiment setup is first given. Subsequently, we delve into the computation-constrained assessment, including streaming evaluation with platforms altering and resource sharing. Finally, we analyze the association between streaming performance and input size/backbone selection.

4.1. Experiment Setup

In the ASAP benchmark, vision-centric perception is instantiated on camera-based 3D detection, which is the fundamental task in autonomous driving perception. The extended nuScenes-H dataset is leveraged to evaluate 3D detectors. Following [27], the streaming evaluation is con-

ducted with a hardware-dependent simulator. For camera-based 3D detectors [21, 22, 30, 31, 33, 53, 54] evaluated in the ASAP benchmark, the model inference times are measured with their open-sourced code on a specific GPU with batch size 1. For measuring the inference time of monocular paradigms [53, 54], we set the batch size as 6, as [53, 54] process the surround-view images independently. More implementation details are in the supplement.

4.2. Computation-Constrained Assessment

Equipped with the ASAP benchmark, we analyze the streaming performance of seven modern 3D detectors (FCOS3D [53], PGD [54], BEVDet [22], BEVDet4D [21], BEVFormer [31], PETR [33], BEVDepth [30]) and the proposed baselines (BEVDepth-Sv, BEVDepth-Sf) under three platforms (RTX3090, RTX2070S, and GTX1060). From the results in Tab. 2, it can be observed that:

(1) Compared with the offline evaluation, all these 3D detectors suffer from performance drops on the ASAP benchmark. Even equipped with the powerful computation platform (RTX3090), the mAP-S of BEVFormer, FCOS3D and PGD relatively drop by 25.3%, 29.5% and 38.5% than the offline counterparts. For efficient models (frame rate $\approx$ 10FPS) BEVDet, BEVDet4D, BEVDepth, the mAP-S still relatively drop by 6.1%, 8.6%, 7.2%, as any detection results miss at least one frame in the streaming evaluation.

(2) The streaming performance degrades continually as the computation power is increasingly constrained. As the Tera Floating Point Operations Per Second (TFLOPS) alters from 35.6TFLOPS@RTX3090 to 9.1TFLOPS@RTX2070S, the mAP-S of FCOS3D, PGD, BEVDet, BEVDet4D, BEVFormer, PETR, BEVDepth relatively drop by 27.4%, 32.5%, 1.7%, 7.4%, 24.8%, 9.9%, 5.3%, respectively. Besides, when the computation performance is further constrained on 4.4TFLOPS@GTX1060, the mAP-S further decreases by a large margin. Notably, for models (FCOS3D, PGD, BEVFormer) with higher model latency, the inference frame rates are lower than 0.5FPS on GTX1060, and the corresponding mAP-S of FCOS3D, PGD, and BEVFormer are 0.051, 0.016 and 0.074, which are far from practical deployment.

(3) The model rank alters under different computation performances. An illustrative comparison is in Fig. 1, where BEVFormer ranks 1st in the offline evaluation and relatively outperforms the 2nd-best competitor BEVDepth by 19.3%. However, BEVDepth suppresses BEVFormer during the streaming evaluation on RTX3090. Moreover, on the RTX2070S, the mAP-S of efficient 3D detectors (BEVDet, BEVDet4D, PETR) exceed that of BEVFormer, and the performance gap between BEVFormer and [21, 22, 30, 33] is further enlarged on GTX1060.

(4) Future state estimation can compensate for the in-

Table 3. Streaming performance (mAP-S) of FCOS3D [53], PGD [54], BEVFormer [31] and the corresponding velocity-based updating baselines. The experiment are conducted on RTX3090, and we report AP-S on high-speed category (*e.g.*, car, bus), and slow-motion category (*e.g.*, pedestrian).

Method	mAP-S↑	AVE↓	Car	Bus	Ped.
FCOS3D	0.208	1.315	0.244	0.111	0.300
FCOS3D-Sv	0.218 (+4.8%)	1.315	0.273	0.133	0.310
PGD	0.206	1.285	0.240	0.092	0.293
PGD-Sv	0.217 (+5.3%)	1.285	0.266	0.124	0.302
BEVFormer	0.310	0.397	0.373	0.236	0.402
BEVFormer-Sv	0.344 (+10.9%)	0.397	0.477	0.324	0.424

ference time delay, which improves the streaming performance. For the velocity-based updating baseline built upon BEVDepth, BEVDepth-Sv relatively improves mAP-S by 1.5%, 3.3%, and 16.4% on RTX3090, RTX2070S, and GTX1060. More baseline results are in Tab. 3, where FCOS3D, PGD and BEVFormer relatively enhance mAP-S by 4.8%, 5.3% and 10.9% on RTX3090. Note that BEVFormer-Sv obtains higher improvement due to the accurate velocity estimation (*i.e.*, AVE@BEVFormer is 0.397, which is significantly lower than that of FCOS3D (1.315) and PGD (1.285)). Besides, we empirically find that high-speed objects particularly benefit from the velocity-based updating strategy. *e.g.*, for BEVFormer@RTX3090, the AP-S of *car* and *bus* are relatively improved by 27.9% and 37.3%, while the relative improvement of slow-motion objects (*pedestrian*) is within 5%, which inspires future streaming algorithms to consider the discrepancy in per-category velocity. For the learning-based forecasting baseline built upon BEVDepth, BEVDepth-Sf relatively improves mAP-S by 1.9%, 3.0%, and 4.0% on RTX3090, RTX2070S, and GTX1060. However, the improvements on RTX2070S and GTX1060 (3.0%, 4.0%) are inferior to that of BEVDepth-Sv (3.3%, 16.4%), as the model frame rates on RTX2070S (4.4FPS) and GTX1060 (1.4FPS) are greatly slower than the streaming input speed (12FPS), and simply predicting the next frame is not sufficient to compensate for the inference time delay. This issue may be alleviated by forecasting future states of the next $N(N \geq 2)$ frames, while the multi-forecasting architecture may increase the inference time and hamper the streaming perception. Therefore, end-to-end streaming 3D detection remains a promising open problem for future research. More experiment results are in the supplement.

To analyze the performance fluctuation caused by computational resources sharing [9, 61, 65, 70], we evaluate 3D detectors (BEVFormer [31] and BEVDepth [30]) while the GPU (RTX3090) is simultaneously processing N ResNet18-based [19] classification tasks. As illustrated in Fig. 6, BEVFormer and BEVDepth suffer performance drops when the number of classification tasks increases, as

fewer computational resources are allocated to the 3D detection task. Specifically, the mAP-S of BEVFormer and BEVDepth relatively drop by 37.7% and 25.1% as the number of classification tasks increases from 0 to 10. Notably, the proposed velocity-based updating baselines consistently improve the streaming performance under computation sharing, and BEVDepth-Sv, BEVFormer-Sv relatively improve the mAP-S by 11.6% and 8.3% when 10 classification tasks are executing.
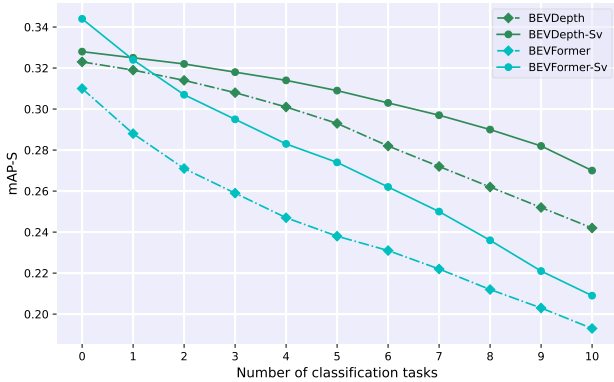

Figure 6. Comparison of streaming performance of BEVFormer [31], BEVFormer-Sv, BEVDepth [30] and BEVDepth-Sv under computational resources sharing, where the x-axis denotes the number of ResNet18-based [19] classification tasks.

The above experiment results reveal that the computational resources significantly influence streaming performance. While the high-performance detector [31] generates accurate predictions with powerful computation, its streaming performance suffers drops with constrained computation. In contrast, the streaming performance of efficient detectors [21, 22, 30] are more consistent across different constraints, indicating that the model latency and computation budget should be regarded as design choices to optimize the practical deployment.

4.3. Analysis on Input Size and Backbone Selection

In this subsection, experiments are conducted to investigate the association between streaming performance and input size/backbone selection. Specifically, we evaluate BEVDepth [30] and BEVDepth-Sv with different image sizes (704×256, 1408×512) and backbones (ResNet50, ResNet101 [19]). The results are shown in Tab. 4. For the offline evaluation, BEVDepth@ResNet101 relatively improves BEVDepth@ResNet50 by 3.4%. Besides, the improvement is 18.4% as the image resolution is further expanded ($704 \times 256 \rightarrow 1408 \times 512$). For the streaming evaluation on BEVDepth, replacing ResNet50 with ResNet101 relatively enhances mAP-S by 2.5%. However, the mAP-S relatively drops by 3.7% when the input size is further expanded. Notably, our baseline BEVDepth-Sv obtains

Table 4. Evaluation with different input sizes and backbones. For *Streaming*=✓, we conduct the streaming evaluation on RTX3090. For *Streaming*=✗, the offline evaluation is performed.

Method	Backbone	Input Size	Streaming	mAP(-S) ↑
BEVDepth	R-50	704×256	✗	0.348
BEVDepth	R-101	704×256	✗	0.360 (+3.4%)
BEVDepth	R-101	1408×512	✗	0.412 (+18.4%)
BEVDepth	R-50	704×256	✓	0.323
BEVDepth	R-101	704×256	✓	0.331 (+2.5%)
BEVDepth	R-101	1408×512	✓	0.311 (-3.7%)
BEVDepth-Sv	R-50	704×256	✓	0.328
BEVDepth-Sv	R-101	704×256	✓	0.340 (+3.7%)
BEVDepth-Sv	R-101	1408×512	✓	0.341 (+4.0%)

mAP-S improvement with large input size (1408×512) and ResNet101 backbone, but the improvement (4%) is significantly lower than that of the offline metric (18.4%). The above results inspire that high-resolution inputs and stronger backbones may hinder streaming performance due to high latency. Therefore, the input size/backbone selection should be meticulously designed in the practical deployment.

5. Conclusion

In this paper, the ASAP benchmark is proposed to evaluate the online performance of vision-centric driving perception approaches. Specifically, we extend 12Hz raw images of nuScenes dataset, and introduce the nuScenes-H dataset for camera-based streaming 3D detection. Besides, the SPUR protocol is established for computation-constrained evaluation. Additionally, we propose ASAP baselines to compensate for the inference time delay, which consistently enhance the streaming performance on three hardware. Equipped with the ASAP benchmark, we investigate the streaming performance of seven modern camera-based 3D detectors and two proposed baselines under various computation constraints. The experiment results reveal that the model latency and computation budget should be regarded as design choices for practical deployment.

6. Limitation and Future Work

The proposed ASAP benchmark takes a step to practical vision-centric perception in autonomous driving. Currently, the ASAP benchmark utilizes modern GPUs (*e.g.*, NVIDIA RTX3090, NVIDIA RTX2070S, NVIDIA GTX1060) to conduct the streaming evaluation. A more practical strategy is to evaluate with the computation of system-on-a-chip (*e.g.*, NVIDIA Thor, Mobileye EyeQ5, Horizon Journey5), and deploy algorithms with 8-bit int/floating point (INT8, FP8) precision acceleration. Furthermore, in future work, more autonomous-driving tasks (*e.g.*, semantic map construction, depth estimation, motion forecasting) should be considered in the SPUR evaluation protocol, towards multi-task and end-to-end autonomy.

References

[1] Adil Kaan Akan and Fatma Güney. Stretchbev: Stretching future instance prediction spatially and temporally. *arXiv preprint arXiv:2203.13641*, 2022. 1, 3

[2] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020. 2

[3] Gabriel J. Brostow, Jamie Shotton, Julien Fauqueur, and Roberto Cipolla. Segmentation and recognition using structure from motion point clouds. *ECCV*, 2008. 1, 2

[4] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. *CVPR*, 2019. 1, 2, 3, 4, 5

[5] Ming-Fang Chang, Deva Ramanan, James Hays, John Lambert, Patsorn Sangkloy, Jasvinder A. Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter W. Carr, and Simon Lucey. Argoverse: 3d tracking and forecasting with rich maps. *CVPR*, 2019. 2

[6] Qi Chen, Sourabh Vora, and Oscar Beijbom. Polarstream: Streaming object detection and segmentation with polar pillars. *NeurIPS*, 2021. 3

[7] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. *CVPR*, 2016. 1, 2

[8] Anton Konushin Danila Rukhovich, Anna Vorontsova. Imvoxelnet: Image to voxels projection for monocular and multi-view general-purpose 3d object detection. In *WACV*, 2022. 3

[9] José Duato, Francisco D Igual, Rafael Mayo, Antonio J Pena, Enrique S Quintana-Ortí, and Federico Silla. An efficient implementation of gpu virtualization in high performance clusters. In *ECPP*, 2009. 2, 5, 7

[10] Davi Frossard, Shun Da Suo, Sergio Casas, James Tu, and Raquel Urtasun. Strobe: Streaming object detection from lidar packets. *CoRL*, 2020. 3

[11] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. Yolox: Exceeding yolo series in 2021. *arXiv preprint arXiv: 2107.08430*, 2021. 2, 3

[12] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 1, 2, 4

[13] Jakob Geyer, Yohannes Kassahun, Mentar Mahmudi, Xavier Ricou, Rupesh Durgesh, Andrew S. Chung, Lorenz Hauswald, Viet Hoang Pham, Maximilian Mühlegg, Sebastian Dorn, Tiffany Fernandez, Martin Jänicke, Sudesh Ganapati Mirashi, Chiragkumar Savani, Martin Sturm, Oleksandr Vorobiov, Martin Oelker, Sebastian Garreis, and Peter Schuberth. A2d2: Audi autonomous driving dataset. *arXiv preprint arXiv: 2004.06320*, 2020. 1, 2

[14] Anurag Ghosh, Akshay Uttama Nambi, Aditya Singh, Harish Yvs, and Tanuja Ganu. Adaptive streaming perception using deep reinforcement learning. *arXiv preprint arXiv: 2106.05665*, 2021. 3

[15] glenn jocher et al. yolov5. https://github.com/ultralytics/yolov5, 2021. 2

[16] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J. Brostow. Digging into self-supervised monocular depth estimation. In *ICCV*, 2018. 1, 3

[17] Vitor Guizilini, Rares Ambrus, Sudeep Pillai, Allan Raventos, and Adrien Gaidon. 3d packing for self-supervised monocular depth estimation. In *CVPR*, 2019. 1, 3

[18] Wei Han, Zhengdong Zhang, Benjamin Caine, Brandon Yang, Christoph Sprunk, Ouais Alsharif, Jiquan Ngiam, Vijay K. Vasudevan, Jonathon Shlens, and Zhifeng Chen. Streaming object detection for 3-d point clouds. *ECCV*, 2020. 2, 3, 4

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 5, 7, 8

[20] Anthony Hu, Zak Murez, Nikhil Mohan, Sofia Dudas, Jeffrey Hawke, Vijay Badrinarayanan, Roberto Cipolla, and Alex Kendall. Fiery: Future instance prediction in bird's-eye view from surround monocular cameras. *ICCV*, 2021. 1, 3

[21] Junjie Huang and Guan Huang. Bevdet4d: Exploit temporal cues in multi-camera 3d object detection. *arXiv preprint arXiv: 2203.17054*, 2022. 1, 2, 3, 4, 5, 7, 8

[22] Junjie Huang, Guan Huang, Zheng Zhu, Ye Yun, and Dalong Du. Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. *arXiv preprint arXiv: 2112.11790*, 2021. 1, 2, 3, 4, 7, 8

[23] Xinyu Huang, Peng Wang, Cheng Xinjing, Dingfu Zhou, Qichuan Geng, and Ruigang Yang. The apolloscape open dataset for autonomous driving and its application. *TPAMI*, 2020. 1, 2

[24] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 1960. 3, 5

[25] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. *CVPR*, 2018. 1, 2, 3

[26] Hongyang Li, Chonghao Sima, Jifeng Dai, Wenhai Wang, Lewei Lu, Huijie Wang, Enze Xie, Zhiqi Li, Hanming Deng, Hao Tian, Xizhou Zhu, Li Chen, Yulu Gao, Xiangwei Geng, Jia Zeng, Yang Li, Jiazhi Yang, Xiaosong Jia, Bohan Yu, Yu Qiao, Dahua Lin, Si Liu, Junchi Yan, Jianping Shi, and Ping Luo. Delving into the devils of bird's-eye-view perception: A review, evaluation and recipe. *arXiv preprint arXiv: abs/2209.05324*, 2022. 3

[27] Mengtian Li, Yu-Xiong Wang, and Deva Ramanan. Towards streaming perception. *ECCV*, 2020. 2, 3, 4, 6

[28] Qi Li, Yue Wang, Yilun Wang, and Hang Zhao. Hdmapnet: An online hd map construction and evaluation framework. In *ICRA*, 2022. 1, 3

[29] Yinhao Li, Han Bao, Zheng Ge, Jinrong Yang, Jianjian Sun, and Zeming Li. Bevstereo: Enhancing depth estimation in multi-view 3d object detection with dynamic temporal stereo. *arXiv preprint arXiv: 2209.10248*, 2022. 1, 2, 3, 5

[30] Yinhao Li, Zheng Ge, Guanyi Yu, Jinrong Yang, Zengran Wang, Yukang Shi, Jianjian Sun, and Zeming Li. Bevdepth: Acquisition of reliable depth for multi-view 3d object detection. *arXiv preprint arXiv: 2206.10092*, 2022. 1, 2, 3, 4, 5, 6, 7, 8

[31] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Yu Qiao, and Jifeng Dai. Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. *ECCV*, 2022. 1, 2, 3, 4, 5, 7, 8

[32] 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. 2

[33] Yingfei Liu, Tiancai Wang, Xiangyu Zhang, and Jian Sun. Petr: Position embedding transformation for multi-view 3d object detection. *ECCV*, 2022. 1, 2, 3, 4, 7

[34] Yingfei Liu, Junjie Yan, Fan Jia, Shuailin Li, Qi Gao, Tiancai Wang, Xiangyu Zhang, and Jian Sun. Petrv2: A unified framework for 3d perception from multi-camera images. *arXiv preprint arXiv: 2206.01256*, 2022. 1, 2, 3, 5

[35] Zechen Liu, Zizhang Wu, and Roland Tóth. Smoke: Single-stage monocular 3d object detection via keypoint estimation. In *CVPR Workshops*, 2020. 3

[36] Chenyang Lu, Marinus Jacobus Gerardus van de Molengraft, and Gijs Dubbelman. Monocular semantic occupancy grid mapping with convolutional variational encoder-decoder networks. *RAL*, 2019. 3

[37] Yuexin Ma, Tai Wang, Xuyang Bai, Huitong Yang, Yuenan Hou, Yaming Wang, Yu Qiao, Ruigang Yang, Dinesh Manocha, and Xinge Zhu. Vision-centric BEV perception: A survey. *arXiv preprint arXiv: abs/2208.02797*, 2022. 3

[38] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulò, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. *ICCV*, 2017. 1, 2

[39] nuScenes Contributors. The devkit of the nuscenes dataset. https://github.com/nutonomy/nuscenes-devkit, 2019. 4

[40] Bowen Pan, Jiankai Sun, Ho Yin Tiga Leung, Alex Andonian, and Bolei Zhou. Cross-view semantic segmentation for sensing surroundings. *RAL*, 2020. 1, 3

[41] Jinhyung Park, Chenfeng Xu, Shijia Yang, Kurt Keutzer, Kris Kitani, Masayoshi Tomizuka, and Wei Zhan. Time will tell: New outlooks and A baseline for temporal multi-view 3d object detection. *arXiv preprint arXiv: 2210.02443*, 2022. 1, 2, 3, 5

[42] Abhishek Patil, Srikanth Malla, Haiming Gang, and Yi-Ting Chen. The h3d dataset for full-surround 3d multi-object detection and tracking in crowded urban scenes. *ICCV*, 2019. 2

[43] Lang Peng, Zhirong Chen, Zhangjie Fu, Pengpeng Liang, and Erkang Cheng. Bevsegformer: Bird's eye view semantic segmentation from arbitrary camera rigs. *arXiv preprint arXiv:2203.04050*, 2022. 1, 3

[44] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 2

[45] Thomas Roddick and Roberto Cipolla. Predicting semantic map representations from images using pyramid occupancy networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11138–11147, 2020. 1, 3

[46] O. Scheel, L. Bergamini, M. Woczyk, B Osiński, and P. Ondruska. Urban driver: Learning to drive from real-world demonstrations using policy gradients. *CoRL*, 2021. 1, 2

[47] Samuel Schulter, Menghua Zhai, Nathan Jacobs, and Manmohan Chandraker. Learning to look around objects for top-view representations of outdoor scenes. In *ECCV*, 2018. 3

[48] Ken Shoemake. Animating rotation with quaternion curves. In *SIGGRAPH*, 1985. 4

[49] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay K. Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. *CVPR*, 2020. 1, 2, 4

[50] Mingxing Tan, Ruoming Pang, and Quoc V. Le. Efficientdet: Scalable and efficient object detection. In *CVPR*, 2020. 2

[51] Sourabh Vora, Alex H. Lang, Bassam Helou, and Oscar Beijbom. Pointpainting: Sequential fusion for 3d object detection. *CVPR*, 2019. 3

[52] Tai Wang, Jiangmiao Pang, and Dahua Lin. Monocular 3d object detection with depth from motion. In *ECCV*, 2022. 3

[53] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. FCOS3D: Fully convolutional one-stage monocular 3d object detection. In *ICCV Workshops*, 2021. 3, 4, 7

[54] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. Probabilistic and Geometric Depth: Detecting objects in perspective. In *CoRL*, 2021. 3, 4, 7

[55] Xiaofeng Wang, Zheng Zhu, Guan Huang, Xu Chi, Yun Ye, Ziwei Chen, and Xingang Wang. Crafting monocular cues and velocity guidance for self-supervised multi-frame depth learning. *arXiv preprint arXiv:2208.09170*, 2022. 1, 3

[56] Zengran Wang, Chen Min, Zheng Ge, Yinhao Li, Zeming Li, Hongyu Yang, and Di Huang. STS: surround-view temporal stereo for multi-view 3d detection. *arXiv preprint arXiv: 2208.10145*, 2022. 1, 2, 3, 5

[57] Jamie Watson, Oisin Mac Aodha, Victor Adrian Prisacariu, Gabriel J. Brostow, and Michael Firman. The temporal opportunist: Self-supervised multi-frame monocular depth. In *CVPR*, 2021. 1, 3

[58] Yi Wei, Linqing Zhao, Wenzhao Zheng, Zheng Zhu, Yongming Rao, Guan Huang, Jiwen Lu, and Jie Zhou. Surround-depth: Entangling surrounding views for self-supervised multi-camera depth estimation. *CoRL*, 2022. 1, 3

[59] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and James Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting. In *NeurIPS*, 2021. 1, 2, 4

[60] Felix Wimbauer, Nan Yang, Lukas von Stumberg, Niclas Zeller, and Daniel Cremers. Monorec: Semi-supervised dense reconstruction in dynamic environments from a single moving camera. In *CVPR*, 2020. 1, 3

[61] Enze Xie, Zhiding Yu, Daquan Zhou, Jonah Philion, Anima Anandkumar, Sanja Fidler, Ping Luo, and Jose M. Alvarez. M^{2}bev: Multi-camera joint 3d detection and segmentation with unified birds-eye view representation. *arXiv preprint arXiv: 2204.05088*, 2022. 1, 2, 3, 5, 7

[62] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 2018. 1, 2, 3

[63] Jinrong Yang, Songtao Liu, Zeming Li, Xiaoping Li, and Jian Sun. Real-time object detection for streaming perception. In *CVPR*, 2022. 3

[64] Weixiang Yang, Qi Li, Wenxi Liu, Yuanlong Yu, Yuexin Ma, Shengfeng He, and Jia Pan. Projecting your view attentively: Monocular road scene layout estimation via cross-view transformation. *CVPR*, 2021. 1, 3

[65] Ting-An Yeh, Hung-Hsin Chen, and Jerry Chou. Kubeshare: A framework to manage gpus as first-class and shared resources in container cloud. In *HPTC*, 2020. 2, 5, 7

[66] Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Center-based 3d object detection and tracking. *CVPR*, 2021. 1, 2, 3, 4

[67] 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. 1, 2

[68] Shanshan Zhang, Rodrigo Benenson, and Bernt Schiele. Citypersons: A diverse dataset for pedestrian detection. *CVPR*, 2017. 2

[69] Yunpeng Zhang, Jiwen Lu, and Jie Zhou. Objects are different: Flexible monocular 3d object detection. In *CVPR*, 2021. 3

[70] Yunpeng Zhang, Zheng Zhu, Wenzhao Zheng, Junjie Huang, Guan Huang, Jie Zhou, and Jiwen Lu. Beverse: Unified perception and prediction in birds-eye-view for vision-centric autonomous driving. *arXiv preprint arXiv: 2205.09743*, 2022. 1, 2, 5, 7

[71] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. *CVPR*, 2018. 3

[72] Jiayu Zou, Junrui Xiao, Zheng Zhu, Junjie Huang, Guan Huang, Dalong Du, and Xingang Wang. Hft: Lifting perspective representations via hybrid feature transformation. *arXiv preprint arXiv:2204.05068*, 2022. 1, 3