BEVFusion: Multi-Task Multi-Sensor Fusion with Unified Bird’s-Eye View Representation

Zhijian Liu\textsuperscript{*1}, Haotian Tang\textsuperscript{*1}, Alexander Amini\textsuperscript{1}, Xinyu Yang\textsuperscript{1}, Huizi Mao\textsuperscript{2}, Daniela L. Rus\textsuperscript{1}, Song Han\textsuperscript{1}

Abstract—Multi-sensor fusion is essential for an accurate and reliable autonomous driving system. Recent approaches are based on point-level fusion: augmenting the LiDAR point cloud with camera features. However, the camera-to-LiDAR projection throws away the semantic density of camera features, hindering the effectiveness of such methods, especially for semantic-oriented tasks (such as 3D scene segmentation). In this paper, we propose BEVFusion, an efficient and generic multi-task multi-sensor fusion framework. It unifies multi-modal features in the shared bird’s-eye view (BEV) representation space, which nicely preserves both geometric and semantic information. To achieve this, we diagnose and lift the key efficiency bottlenecks in the view transformation with optimized BEV pooling, reducing latency by more than 40\%. BEVFusion is fundamentally task-agnostic and seamlessly supports different 3D perception tasks with almost no architectural changes. It establishes the new state of the art on the nuScenes benchmark, achieving 13.6\% higher mIoU and NDS on 3D object detection and 13.6\% higher mIoU on BEV map segmentation, with 1.9\% lower computation cost. Code to reproduce our results is available at https://github.com/mit-han-lab/bevfusion.

I. INTRODUCTION

Autonomous driving systems are equipped with diverse sensors. For instance, Waymo’s self-driving vehicles have 29 cameras, 6 radars, and 5 LiDARs. Different sensors provide complementary signals: e.g., cameras capture rich semantic information, LiDARs provide accurate spatial information, while radars offer instant velocity estimation. Thus, multi-sensor fusion is essential for accurate and reliable perception.

Data from different sensors are expressed in fundamentally different modalities: e.g., cameras capture data in perspective view and LiDAR in 3D view. To resolve this view discrepancy, we have to find a unified representation that is suitable for multi-task multi-modal feature fusion. Due to the tremendous success in 2D perception, the natural idea is to project the LiDAR point cloud onto the camera and process the RGB-D data with 2D CNNs. However, this LiDAR-to-camera projection introduces severe geometric distortion (see Figure 1a), which makes it less effective for geometric-oriented tasks, such as 3D object recognition.

Recent sensor fusion methods follow the other direction. They augment the LiDAR point cloud with semantic labels [1], CNN features [2], [3] or virtual points from 2D images [4], and then apply an existing LiDAR-based detector to predict 3D bounding boxes. Although they have demonstrated remarkable performance on large-scale detection benchmarks, these point-level fusion methods barely work on semantic-oriented tasks, such as BEV map segmentation [5], [6], [7], [8]. This is because the camera-to-LiDAR projection is semantically lossy (see Figure 1b): for a typical 32-beam LiDAR scanner, only 5\% camera features will be matched to a LiDAR point while all others will be dropped. Such density differences will become even more drastic for sparser LiDARs (or radars).

In this paper, we propose BEVFusion to unify multi-modal features in a shared bird’s-eye view (BEV) representation space for task-agnostic learning. We maintain both geometric structure and semantic density (see Figure 1c) and naturally support most 3D perception tasks (since their output space can be naturally captured in BEV). While converting all features to BEV, we identify the major prohibitive efficiency bottleneck in the view transformation: i.e., the BEV pooling operation alone takes more than 80\% of the model’s runtime. Then, we propose a specialized kernel with precomputation and interval reduction to eliminate this bottleneck, achieving more than 40\% speedup. Finally, we apply the fully-convolutional BEV encoder to fuse the unified BEV features and append a few task-specific heads to support different target tasks.

BEVFusion sets the new state-of-the-art 3D object detection performance on both nuScenes and Waymo benchmarks. It outperforms all published methods with or without test-time augmentation and model ensemble. BEVFusion demonstrates even more significant improvements on BEV map segmentation. It achieves 6\% higher mIoU than camera-only models and 13.6\% higher mIoU than LiDAR-only models, while existing fusion methods hardly work. Moreover, BEVFusion
Multi-View RGB Images

LiDAR Point Cloud

Camera Encoder

Camera Features

Camera-to-BEV View Transform

Camera Feat. (in BEV)

LiDAR Encoder

LiDAR Features

Flatten (along z-axis)

LiDAR Feat. (in BEV)

BEV Encoder

Fused BEV Features

BEVFusion

3D Object Detection

Task-Specific Heads

BEV Map Segmentation

Fig. 2: BEVFusion extracts features from multi-modal inputs and converts them into a shared bird’s-eye view (BEV) space efficiently using view transformations. It fuses the unified BEV features with a fully-convolutional BEV encoder and supports different tasks with task-specific heads.

is highly efficient, delivering all these results with $1.9 \times$ lower computation cost.

While point-level fusion has been the go-to choice over the past three years, BEVFusion provides a fresh perspective to the field of multi-sensor fusion by rethinking “Is LiDAR space the right place to perform sensor fusion?” It showcases the superior performance of an alternative paradigm that has been previously overlooked. Simplicity is also its key strength. We hope this work will serve as a simple yet strong baseline for future sensor fusion research and inspire the researchers to rethink the design and paradigm for generic multi-task multi-sensor fusion.

II. RELATED WORK

LiDAR-Based 3D Perception. Researchers have designed single-stage 3D object detectors [9], [10], [11], [12], [13], [14] that extract flattened point cloud features using PointNets [15] or SparseConvNet [16] and perform detection in the BEV space. Later, [17], [18], [19], [20], [21], [22], [23] explore anchor-free single-stage 3D object detection. Another stream of research [24], [25], [26], [27], [28], [29] focuses on two-stage object detector design, which adds an RCNN network to existing one-stage object detectors.

Camera-Based 3D Perception. Due to the high cost of LiDAR sensors, researchers spend significant efforts on camera-only 3D perception. FCOS3D [30] extends an image detector [31] with additional 3D regression branches, which is later improved by [32], [33] in depth modeling. Instead of performing object detection in the perspective view, [34], [35] design a DETR [36], [37]-based detection head with learnable object queries in the 3D space. Inspired by the design of LiDAR-based detectors, another type of camera-only 3D perception models explicitly converts the camera features from perspective view to the bird’s-eye view using a view transformer [5], [38], [39], [6]. BEVDet [40] and M$^2$BEV [41] extends LSS [6] and OFT [38] to 3D object detection and CaDDN [42] adds explicit depth estimation supervision to the view transformer. Recent research [43], [8] also studies view transformation with multi-head attention.

Multi-Sensor Fusion. Recently, multi-sensor fusion arouses significant interest among the 3D detection community. Existing approaches can be classified into proposal-level and point-level fusion methods. Early approach MV3D [44] creates object proposals in 3D and projects the proposals to images to extract RoI features. [45], [46], [47] all lift image proposals into a 3D frustum. Recent works FUTR3D [48] and TransFusion [49] define object queries in the 3D space and fuses image features onto these proposals. All proposal-level fusion methods are object-centric and cannot trivially generalize to other tasks such as BEV map segmentation. Point-level fusion methods, on the other hand, usually paint image semantic features onto foreground LiDAR points and perform LiDAR-based detection on the decorated point cloud inputs. As such, they are both object-centric and geometric-centric. Among these methods, [1], [2], [4], [50], [51] are (LiDAR) input-level decoration, while DCF [52] and DeepFusion [3] are feature-level decoration.

In contrast to all existing methods, BEVFusion performs sensor fusion in a shared BEV space and treats foreground and background, geometric and semantic information equally. It is a generic multi-task multi-sensor perception framework.

III. METHOD

BEVFusion, as shown in Figure 2, focuses on multi-sensor fusion (i.e., multi-view cameras and LiDAR) for multi-task 3D perception (i.e., detection and segmentation). Given different sensory inputs, we first apply modality-specific encoders to extract their features. We transform multi-modal features into a unified BEV representation that preserves both geometric and semantic information. We identify the efficiency bottleneck of the view transformation and accelerate BEV pooling with precomputation and interval reduction. We then apply the convolution-based BEV encoder to the unified BEV features to alleviate the local misalignment between different features. Finally, we append a few task-specific heads to support different 3D tasks.

A. Unified Representation

Different features can exist in different views. For instance, camera features are in the perspective view, while
LiDAR/radar features are typically in the 3D/bird’s-eye view. Even for camera features, each one of them has a distinct viewing angle (i.e., front, back, left, right). This view discrepancy makes the feature fusion difficult since the same element in different feature tensors might correspond to very different spatial locations (and the naïve elementwise feature fusion will not work in this case). Thus, it is crucial to find a shared representation, such that (1) all sensor features can be easily converted to it without information loss, and (2) it is suitable for different types of tasks.

To Camera. Motivated by RGB-D data, one choice is to project the LiDAR point cloud to the camera plane and render the 2.5D sparse depth. However, this conversion is geometrically lossy. Two neighbors on the depth map can be far away from each other in the 3D space. This makes the camera view less effective for tasks that focus on the object/scene geometry, such as 3D object detection.

To LiDAR. Most state-of-the-art sensor fusion methods [1], [4], [3] decorate LiDAR points with their corresponding camera features (e.g., semantic labels, CNN features or virtual points). However, this camera-to-LiDAR projection is semantically lossy. Camera and LiDAR features have drastically different densities, resulting in only less than 5% of camera features being matched to a LiDAR point (for a 32-channel LiDAR scanner). Giving up the semantic density of camera features severely hurts the model’s performance on semantic-oriented tasks (such as BEV map segmentation). Similar drawbacks also apply to more recent fusion methods in the latent space (e.g., object query) [48], [49].

To Bird’s-Eye View. We adopt the bird’s-eye view (BEV) as the unified representation for fusion. This view is friendly to almost all perception tasks since the output space is also in BEV. More importantly, the transformation to BEV keeps both geometric structure (from LiDAR features) and semantic density (from camera features). On the one hand, the LiDAR-to-BEV projection flattens the sparse LiDAR features along the height dimension, thus does not create geometric distortion in Figure 1a. On the other hand, camera-to-BEV projection casts each camera feature pixel back into a ray in the 3D space (detailed in the next section), which can result in a dense BEV feature map in Figure 1c that retains full semantic information from the cameras.

B. Efficient Camera-to-BEV Transformation

Camera-to-BEV transformation is non-trivial because the depth associated with each camera feature pixel is inherently ambiguous. Following LSS [6], we explicitly predict the discrete depth distribution of each pixel. We then scatter each feature pixel into \(D\) discrete points along the camera ray and rescale the associated features by their corresponding depth probabilities (Figure 3a). This generates a camera feature point cloud of size \(NHWD\), where \(N\) is the number of cameras and \((H, W)\) is the camera feature map size. Such 3D feature point cloud is quantized along the \(x, y\) axes with a step size of \(r\) (e.g., 0.4m). We use the BEV pooling operation to aggregate all features within each \(r \times r\) BEV grid and flatten the features along the \(z\)-axis.

Though simple, BEV pooling is surprisingly inefficient and slow, taking more than 500ms on an RTX 3090 GPU (while the rest of our model only takes around 100ms). This is because the camera feature point cloud is very large: for a typical workload\(^*\), there could be around 2 million points generated for each frame, two orders of magnitudes denser than a LiDAR feature point cloud. To lift this efficiency bottleneck, we propose to optimize the BEV pooling with precomputation and interval reduction.

Precomputation. The first step of BEV pooling is to associate each point in the camera feature point cloud with a BEV grid. Different from LiDAR point clouds, the coordinates of the camera feature point cloud are fixed (as long as the camera intrinsics and extrinsics stay the same, which is usually the case after proper calibration). Motivated by this, we precompute the 3D coordinate and the BEV grid index of each point. We also sort all points according to grid indices and record the rank of each point. During inference, we only need to reorder all feature points based on the precomputed ranks. This caching mechanism can reduce the latency of grid association from 17ms to 4ms.

\(^*\) \(N = 6, (H, W) = (32, 88), \) and \(D = (60 - 1)/0.5 = 118.\) This corresponds to six multi-view cameras, each associated with a 32×88 camera feature map (which is downsampled from a 256×704 image by 8×). The depth is discretized into \([1, 60]\) meters with a step size of 0.5 meter.
within the same BEV grid will be consecutive in the tensor whistles. It breaks the convention of decorating camera features onto the LiDAR point cloud and delivers at least 1.3% higher mAP and NDS with $1.5 \times \text{2}$ lower computation cost. ($^*$: our re-implementation; $^\dagger$: with test-time augmentation)

| Modality          | mAP (test) | NDS (test) | mAP (val) | NDS (val) | MACs (G) | Latency (ms) |
|-------------------|------------|------------|-----------|-----------|----------|--------------|
| M$^2$BEV [41]     | C          | 42.9       | 47.4      | 41.7      | 47.0     | –            |
| BEVFormer [43]    | C          | 44.5       | 53.5      | 41.6      | 51.7     | –            |
| PointPillars [10] | L          | –          | –         | 52.3      | 61.3     | 65.5         | 34.4         |
| SECOND [11]       | L          | 52.8       | 63.3      | 52.6      | 63.0     | 85.0         | 69.8         |
| CenterPoint [17]  | L          | 60.3       | 67.3      | 59.6      | 66.8     | 153.5        | 80.7         |
| PointPainting [1] | C+L        | –          | –         | 65.8$^*$  | 69.6$^*$ | 370.0        | 185.8        |
| PointAugmenting [2]| C+L        | 66.8$^\dagger$ | 71.0$^\dagger$ | –          | –         | 408.5        | 234.4        |
| MVP [4]           | C+L        | 66.4       | 70.5      | 66.1$^*$  | 70.0$^*$ | 371.7        | 187.1        |
| FusionPainting [50]| C+L        | 68.1       | 71.6      | 66.5      | 70.7     | –            | –            |
| AutoAlign [51]    | C+L        | –          | –         | 66.6      | 71.1     | –            | –            |
| FUTR3D [48]      | C+L        | –          | –         | 64.5      | 68.3     | 1069.0       | 321.4        |
| Transfusion [49]  | C+L        | 68.9       | 71.6      | 67.5      | 71.3     | 485.8        | 156.6        |
| **BEVFusion (Ours)** | C+L    | **70.2**  | **72.9** | **68.5**  | **71.4** | **253.2**   | **119.2**   |

**Interval Reduction.** After grid association, all points within the same BEV grid will be consecutive in the tensor representation. The next step of BEV pooling is to aggregate the features within each BEV grid by some symmetric function (e.g., mean, max, and sum). As in Figure 3b, existing implementation [6] first computes the prefix sum over all points and then subtracts the values at the boundaries where indices change. However, the prefix sum operation requires tree reduction on the GPU and produces many unused partial sums (since we only need those values on the boundaries), both of which are inefficient. To accelerate feature aggregation, we implement a specialized GPU kernel that parallelizes directly over BEV grids: we assign a GPU thread to each grid that calculates its interval sum and writes the result back. This kernel removes the dependency between outputs (thus does not require multi-level tree reduction) and avoids writing the partial sums to the DRAM, reducing the latency of feature aggregation from 500ms to 2ms (Figure 3c).

**Takeaways.** The camera-to-BEV transformation is $40 \times$ faster with our optimized BEV pooling: the latency is reduced from more than 500ms to 12ms (only 10% of our model’s end-to-end runtime) and scales well across different feature resolutions (Figure 3d). This is a key enabler for unifying multi-modal sensory features in the shared BEV representation. Two concurrent works of ours also identify this efficiency bottleneck in the camera-only 3D detection. This kernel could potentially benefit from more accurate depth estimation (e.g., supervising the view transformer with ground-truth depth [42], [53]), which we leave for future work.

**C. Fully-Convolutional Fusion**

With all sensory features converted to the shared BEV representation, we can easily fuse them together with an elementwise operator (such as concatenation). Though in the same space, LiDAR BEV features and camera BEV features can still be spatially misaligned to some extent due to the inaccurate depth in the view transformer. To this end, we apply a convolution-based BEV encoder (with a few residual blocks) to compensate for such local misalignments. Our method could potentially benefit from more accurate depth estimation (e.g., supervising the view transformer with ground-truth depth [42], [53]), which we leave for future work.

**D. Multi-Task Heads**

We apply multiple task-specific heads to the fused BEV feature map. Our method is applicable to most 3D perception tasks. For 3D object detection, we follow [17], [49] to use a class-specific center heatmap head to predict the center location of all objects and a few regression heads to estimate the object size, rotation, and velocity. For map segmentation, different map categories may overlap (e.g., crosswalk is a subset of drivable space). Therefore, we formulate this problem as multiple binary semantic segmentation, one for each class. We follow CVT [8] to train the segmentation head with the standard focal loss [54].

**IV. EXPERIMENTS**

We evaluate BEVFusion for camera-LiDAR fusion on 3D object detection and BEV map segmentation, covering both geometric- and semantic-oriented tasks. Our framework can be easily extended to support other types of sensors (such as radars and event-based cameras) and other 3D perception tasks (such as 3D object tracking and motion forecasting).
Waymo, which are large-scale datasets for 3D perception with TransFusion, BEVFusion offers 1.3% improvement with VoxelNet as our LiDAR backbone. We apply FPN (for detection) and 0.1m (for segmentation). As detection and segmentation tasks require BEV feature maps with different spatial ranges and sizes, we apply grid sampling with bilinear interpolation before each task-specific head to explicitly transform between different BEV feature maps.

Tab. III: BEVFusion is robust under different lighting and weather conditions, significantly boosting the performance single-modality models under challenging rainy (+10.7) and nighttime (+12.8) scenes.

Tab. IV: BEVFusion outperforms the state-of-the-art multi-sensor fusion methods by 13.6% on BEV map segmentation on nuScenes (val) with consistent improvements across different categories.

Model. We use Swin-T [57] as our image backbone and VoxelNet [11] as our LiDAR backbone. We apply FPN [58] to fuse multi-scale camera features to produce a feature map of 1/8 input size. We downsample camera images to 256×704 and voxelize the LiDAR point cloud with 0.075m resolution. We use the mean average precision (mAP) across 10 foreground classes and the nuScenes detection score (NDS) as our detection metrics. We also measure the single-inference #MACs and latency on an RTX3090 GPU for all open-source methods. We use a single model with no any test-time augmentation for both val and test results.

As in Table I, BEVFusion achieves state-of-the-art results on the nuScenes detection benchmark, with close-to-real-time (8.4 FPS) inference speed on a desktop GPU. Compared with TransFusion [49], BEVFusion offers 1.3% improvement in test split mAP and NDS, while significantly reduces the MACs by 1.9× and measured latency by 1.3×. It also compares favorably against representative point-level fusion methods PointPainting [1] and MVP [4] with 1.6× speedup, 1.5× MACs reduction and 3.8% higher mAP on the test set. We argue that the efficiency gain of BEVFusion comes from the fact that we choose the BEV space as the shared fusion space, which fully utilizes all camera features instead of just a 5% sparse set. Consequently, BEVFusion can achieve the same performance with much smaller resolution for the camera inputs, resulting in significantly lower MACs. Combined with the efficient BEV pooling operator in Section III-B, BEVFusion transfers MACs reduction into measured speedup.

BEVFusion also achieves state-of-the-art performance on the Waymo open dataset [60] (Table II). BEVFusion outperforms the previous state-of-the-art multi-modal detector, DeepFusion [3] with 60% of input frames. Furthermore, DeepFusion ensembles 25 models evaluated with test-time augmentation, while we deliver better performance by applying test-time augmentation to a single BEVFusion model.

A. 3D Object Detection

We first experiment on the geometric-centric 3D object detection benchmark, where BEVFusion achieves superior performance with lower computation cost and measured latency. We use the mean average precision (mAP) across 10 foreground classes and the nuScenes detection score (NDS) as our detection metrics. We also measure the single-inference #MACs and latency on an RTX3090 GPU for all open-source methods. We use a single model without any test-time augmentation for both val and test results.

As in Table I, BEVFusion achieves state-of-the-art results on the nuScenes detection benchmark, with close-to-real-time (8.4 FPS) inference speed on a desktop GPU. Compared with TransFusion [49], BEVFusion offers 1.3% improvement in test split mAP and NDS, while significantly reduces the MACs by 1.9× and measured latency by 1.3×. It also compares favorably against representative point-level fusion methods PointPainting [1] and MVP [4] with 1.6× speedup, 1.5× MACs reduction and 3.8% higher mAP on the test set. We argue that the efficiency gain of BEVFusion comes from the fact that we choose the BEV space as the shared fusion space, which fully utilizes all camera features instead of just a 5% sparse set. Consequently, BEVFusion can achieve the same performance with much smaller resolution for the camera inputs, resulting in significantly lower MACs. Combined with the efficient BEV pooling operator in Section III-B, BEVFusion transfers MACs reduction into measured speedup.

BEVFusion also achieves state-of-the-art performance on the Waymo open dataset [60] (Table II). BEVFusion outperforms the previous state-of-the-art multi-modal detector, DeepFusion [3] with 60% of input frames. Furthermore, DeepFusion ensembles 25 models evaluated with test-time augmentation, while we deliver better performance by applying test-time augmentation to a single BEVFusion model.

B. BEV Map Segmentation

We further compare BEVFusion with state-of-the-art models on the semantic-centric BEV map segmentation task, where BEVFusion achieves an even larger performance boost. We report the Intersection-over-Union (IoU) on 6 background classes and the class-averaged mean IoU as our evaluation metric. As different classes may have overlappings (e.g., car-parking area is also drivable), we evaluate the binary segmentation performance for each class separately and select the highest IoU across different thresholds [8]. For each frame, we only perform the evaluation in the [-50m, 50m] region around the ego car following [6], [8], [41], [43].

We report the BEV map segmentation results in Table III. In contrast to 3D object detection which is a geometric-oriented task, map segmentation is semantic-oriented. As a result,
We first analyze the performance of Weather and Lighting. Table IV. LiDAR-only models face significant challenges in sizes and object distances, especially under more challenging settings (i.e., sparser point clouds, small/distant objects). BEVFusion consistently outperforms MVP across all sparsity levels with a 1.6× reduction in #MACs and achieves a 12% improvement in the 1-beam LiDAR scenario. MVP decorates the point cloud and directly applies CenterPoint on the painted and densified LiDAR input. As a result, it naturally requires the LiDAR-only detector (CenterPoint) to perform well, which is not valid under sparse LiDAR settings (i.e., 35.8 NDS with 1-beam input in Figure 4c). In contrast, BEVFusion integrates multi-sensor information in the shared BEV space and does not rely solely on a robust LiDAR-only detector.

Fig. 4: BEVFusion outperforms state-of-the-art single- and multi-modality detectors under different LiDAR sparsity, object sizes and object distances, especially under more challenging settings (i.e., sparser point clouds, small/distant objects).
Y. Zhang, J. Shlens, Z. Chen, and D. Anguelov, “Scalability in Perception for Autonomous Driving: Waymo Open Dataset,” in CVPR.