Abstract

LiDAR and camera are two important sensors for 3D object detection in autonomous driving. Despite the increasing popularity of sensor fusion in this field, the robustness against inferior image conditions, e.g., bad illumination and sensor misalignment, is under-explored. Existing fusion methods are easily affected by such conditions, mainly due to a hard association of LiDAR points and image pixels, established by calibration matrices.

We propose TransFusion, a robust solution to LiDAR-camera fusion with a soft-association mechanism to handle inferior image conditions. Specifically, our TransFusion consists of convolutional backbones and a detection head based on a transformer decoder. The first layer of the decoder predicts initial bounding boxes from a LiDAR point cloud using a sparse set of object queries, and its second decoder layer adaptively fuses the object queries with useful image features, leveraging both spatial and contextual relationships. The attention mechanism of the transformer enables our model to adaptively determine where and what information should be taken from the image, leading to a robust and effective fusion strategy. We additionally design an image-guided query initialization strategy to deal with objects that are difficult to detect in point clouds. TransFusion achieves state-of-the-art performance on large-scale datasets with sparser point clouds, such as nuScenes [1] and Waymo [43]. LiDAR-only methods are surely insufficient for robust 3D detection due to the sparsity of point clouds. For example, small or distant objects are difficult to detect in LiDAR modality. In contrast, such objects are still clearly visible and distinguishable in high-resolution images. The complementary roles of point clouds and images motivate researchers to design detectors utilizing the best of the two worlds, i.e., multi-modal detectors.

Existing LiDAR-camera fusion methods roughly fall into three categories: result-level, proposal-level, and point-level. The result-level methods, including FPointNet [29] and RoarNet [39], use off-the-shelf 2D detectors to seed 3D proposals, followed by a PointNet [30] for object localization. The proposal-level fusion methods, including MV3D [5] and AVOD [12], perform fusion at the region proposal level by applying RoIPool [31] in each modality for shared proposals. These coarse-grained fusion methods show unsatisfactory results since rectangular regions of interest (RoI) usually contain lots of background noise. Recently, a majority of approaches have tried to do point-level fusion and achieved promising results. They first find a hard association between LiDAR points and image pixels based on calibration matrices, and then augment LiDAR features with the segmentation scores [47, 52] or CNN features [10, 22, 40, 48, 63] of the associated pixels through point-wise concatenation. Similarly, [16, 17, 51, 60] first project a point cloud onto the bird’s eye view (BEV) plane and then fuse the image features with the BEV pixels.

Despite the impressive improvements, these point-level fusion methods suffer from two major problems, as shown in Fig. 1. First, they simply fuse the LiDAR features and image features through element-wise addition or concatenation, and thus their performance degrades seriously with low-quality image features, e.g., images in bad illumination conditions. Second, finding the hard association between sparse LiDAR points and dense image pixels not only wastes many image features with rich semantic information, but also heavily relies on high-quality calibration between two sensors, which is usually hard to acquire due to the inherent spatial-temporal misalignment [64].

To address the shortcomings of the previous fusion approaches, we introduce an effective and robust multi-modal
Improves the effectiveness and robustness of our LiDAR-camera fusion methods. Overall, the corporation of these components significantly shortens the focus of the fusion process, from hard-association to soft-association, leading to the robustness against degenerated image quality and sensor misalignment.

Specifically, we design a sequential fusion method that leverages a sparse set of object queries to produce initial bounding boxes from LiDAR features. Unlike input-independent object queries in 2D [2, 45], we make the object queries input-dependent and category-aware so that the queries are enriched with better position and category information. Next, the second transformer decoder layer adaptively fuses object queries with useful image features associated by spatial and contextual relationships. We leverage a locality inductive bias by spatially constraining the cross attention around the initial bounding boxes to help the network better visit the related positions. Our fusion module not only provides rich semantic information to object queries, but also is more robust to inferior image conditions since the association between LiDAR points and image pixels is established in a soft and adaptive way. Finally, to handle objects that are difficult to detect in point clouds, we introduce an image-guided query initialization module to involve image guidance on the query initialization stage. Overall, the corporation of these components significantly improves the effectiveness and robustness of our LiDAR-camera 3D detector. To summarize, our contributions are fourfold:

1. Our studies investigate the inherent difficulties of LiDAR-camera fusion and reveal a crucial aspect to robust fusion, namely, the soft-association mechanism.
2. We propose a novel transformer-based LiDAR-camera fusion model for 3D detection, which performs fine-grained fusion in an attentive manner and shows superior robustness against degenerated image quality and sensor misalignment.
3. We introduce several simple yet effective adjustments for object queries to boost the quality of initial bounding box predictions for image fusion. An image-guided query initialization module is also designed to handle objects that are hard to detect in point clouds.
4. We achieve the state-of-the-art 3D detection performance on nuScenes and competitive results on Waymo. We also extend our model to the 3D tracking task and achieve the 1st place in the leaderboard of the nuScenes tracking challenge.

2. Related Work

**LiDAR-only 3D Detection** aims to predict 3D bounding boxes of objects in given point clouds [3, 4, 27, 28, 38, 47, 54, 67, 69, 70]. Due to the unordered, irregular nature of point clouds, many 3D detectors first project them onto a regular grid such as 3D voxels [53, 68], pillars [14] or range images [8, 44]. After that, standard 2D or 3D convolutions are used to compute the features in the BEV plane, where objects are naturally separated, with their physical sizes preserved. Other works [36, 37, 55, 56] directly operate on raw point clouds without quantization. The mainstream of 3D detection head is based on anchor boxes [14, 68] following the 2D counterparts, while [49, 58] adopt a center-based representation for 3D objects, largely simplifying the 3D detection pipeline. Despite the popularity of adopting the transformer architecture as a detection head in 2D [2], 3D detection models for outdoor scenarios mostly utilize the transformer for feature extraction [21, 24, 35]. However, the attention operation in each transformer layer requires a computation complexity of $O(N^2)$ for $N$ points, requiring a carefully designed memory reduction operation when handling LiDAR point clouds with millions of points per frame. In contrast, our model retains an efficient convolution backbone for feature extraction and leverages a transformer decoder with a small set of object queries as the detection head, making the computation cost manageable. The concurrent works [19, 20, 23] adopt transformer as a detection head but focus on indoor scenarios and extending these methods to outdoor scenes is non-trivial.

**LiDAR-Camera 3D Detection** has gained increasing attention due to the complementary roles of point clouds and images. Early works [5, 29, 39] adopt result-level or proposal-level fusion, where the fusion granularity is too coarse to release the full potential of two modalities. Since Point-Painting [47] was proposed, the point-level fusion methods [10, 40, 48] have shown great advantages and promising results. However, such methods are easily affected by the sensor misalignment due to the hard association between points and pixels established by calibration matrices. Moreover, the simple point-wise concatenation ignores the quality of real data and contextual relationships between two modalities, and thus leads to degraded performance when the image features are defective. In our work, we explore a more robust and effective fusion mechanism to mitigate these limitations during LiDAR-camera fusion.
3. Methodology

In this section, we present the proposed method TransFusion for LiDAR-camera 3D object detection. As shown in Fig. 2, given a LiDAR BEV feature map and an image feature map from convolutional backbones, our transformer-based detection head first decodes object queries into initial bounding box predictions using the LiDAR information, and then performs LiDAR-camera fusion by attentively fusing object queries with useful image features. Below we will first provide the preliminary knowledge about a transformer architecture for detection and then present the detail of TransFusion.

3.1. Preliminary: Transformer for 2D Detection

Transformer [46] has been widely used for 2D object detection [9, 45, 57, 71] since DETR [2] was proposed. DETR uses a CNN backbone to extract image features and a transformer architecture to convert a small set of learned embeddings (called object queries) into a set of predictions. The follow-up works [45, 57, 71] further equip the object queries with positional information. The final predictions of boxes are the relative offsets w.r.t. the query positions to reduce optimization difficulty. We refer readers to the original papers [2, 71] for more details. In our work, each object query contains a query position providing the localization of the object and a query feature encoding instance information, such as the box’s size, orientation, etc.

3.2. Query Initialization

Input-dependent. The query positions in the seminal works [2, 45, 71] are randomly generated or learned as network parameters, regardless of the input data. Such input-independent query positions will take extra stages (decoder layers) for their models [2, 71] to learn the moving process towards the real object centers. Recently, it has been observed in 2D object detection [57] that with a better initialization of object queries, the gap between 1-layer structure and 6-layer structure could be bridged. Inspired by this observation, we propose an input-dependent initialization strategy based on a center heatmap to achieve competitive performance using only one decoder layer. Specifically, given a d dimensional LiDAR BEV feature map \( F_L \in \mathbb{R}^{X \times Y \times d} \), we first predict a class-specific heatmap \( \mathcal{S} \in \mathbb{R}^{X \times Y \times K} \), where \( X \times Y \) describes the size of the BEV feature map and \( K \) is the number of categories. Then we regard the heatmap as a \( X \times Y \times K \) object candidates and select the top-\( N \) candidates for all the categories as our initial object queries. To avoid spatially too closed queries, following [66], we select the local maximum elements as our object queries, whose values are greater than or equal to their 8-connected neighbors. Otherwise, a large number of queries are needed to cover the BEV plane. The positions and features of the selected candidates are used to initialize the query positions and query features. In this way, our initial object queries will locate at or close to the potential object centers, eliminating the need of multiple decoder layers [20, 23, 50] to refine the locations.

Category-aware. Unlike their 2D projections on the image...
plane, the objects on the BEV plane are all in absolute scale and has small scale variance among the same categories. To leverage such properties for better multi-class detection, we make the object queries category-aware by equipping each query with a category embedding. Specifically, using the category of each selected candidate (e.g., \( S_{ijk} \) belonging to the \( k \)-th category), we element-wisely sum the query feature with a category embedding produced by linearly projecting the one-hot category vector into a \( \mathbb{R}^d \) vector. The category embedding brings benefits in two aspects: on the one hand, it serves as a useful side information when modelling the object-object relations in the self-attention modules and the object-context relations in the cross-attention modules. On the other hand, during prediction, it could deliver valuable prior knowledge of the object, making the network focus on intra-category variance and thus benefiting the property prediction.

3.3. Transformer Decoder and FFN

The decoder layer follows the design of DETR [23] and the detailed architecture is provided in the supplementary Sec. A. The cross attention between object queries and the feature maps (either from point clouds or images) aggregates relevant context onto the object candidates, while the self attention between object queries reasons pairwise relations between different object candidates. The query positions are embedded into \( d \)-dimensional positional encoding with a Multilayer Perceptron (MLP), and element-wisely summed with the query features. This enables the network to reason about both context and position jointly.

The \( N \) object queries containing rich instance information are then independently decoded into boxes and class labels by a feed-forward network (FFN). Following Center-Point [58], our FFN predicts the center offset from the query position as \( \delta x, \delta y \), bounding box height as \( z \), size \( l, w, h \) as \( \log(l), \log(w), \log(h) \), yaw angle \( \alpha \) as \( \sin(\alpha), \cos(\alpha) \) and the velocity (if available) as \( v_x, v_y \). We also predict a per-class probability \( \hat{p} \in [0,1]^K \) for \( K \) semantic classes. Each attribute is computed by a separate two-layer \( 1 \times 1 \) convolution. By decoding each object query into prediction in parallel, we get a set of predictions \( \{b_i, \hat{p}_i\}_i^N \) as output, where \( b_i \) is the predicted bounding box for the \( i \)-th query. Following [23], we adopt the auxiliary decoding mechanism, which adds FFN and supervision after each decoder layer. Hence, we can have initial bounding box predictions from the first decoder layer. We leverage such initial predictions in the LiDAR-camera fusion module to constrain the cross attention, as explained in the next section.

3.4. LiDAR-Camera Fusion

Image Feature Fetching. Although impressive improvement has been brought by point-level fusion methods [47, 48], their fusion quality is largely limited by the sparsity of LiDAR points. When an object only contains a small number of LiDAR points, it can fetch only the same number of image features, wasting the rich semantic information of high-resolution images. To mitigate this issue, we do not fetch the multiview image features based on the hard association between LiDAR points and image pixels. Instead, we retain all the image features \( F_C \in \mathbb{R}^{N_c \times H \times W \times d} \) as our memory bank, and use the cross-attention mechanism in the transformer decoder to perform feature fusion in a sparse-to-dense and adaptive manner, as shown in Fig. 2.

SMCA for Image Feature Fusion. Multi-head attention is a popular mechanism to perform information exchange and build a soft association between two sets of inputs, and it has been widely used for the feature matching task [34, 41]. To mitigate the sensitivity towards sensor calibration and inferior image features brought by the hard-association strategy, we leverage the cross-attention mechanism to build the soft association between LiDAR and images, enabling the network to adaptively determine where and what information should be taken from the images.

Specifically, we first identify the specific image in which the object queries are located using previous predictions as well as the calibration matrices, and then perform cross attention between the object queries and the corresponding image feature map. However, as the LiDAR features and image features are from completely different domains, the object queries might attend to visual regions unrelated to the bounding box to be predicted, leading to a long training time for the network to accurately identify the proper regions on images. Inspired by [9], we design a spatially modulated cross attention (SMCA) module, which weighs the cross attention by a 2D circular Gaussian mask around the projected 2D center of each query. The 2D Gaussian weight mask \( M \) is generated in a similar way as CenterNet [66], \( M_{ij} = \exp\left(-\frac{(i-c_x)^2+(j-c_y)^2}{\sigma_r^2}\right) \), where \( (i,j) \) is the spatial indices of the weight mask \( M \), \( (c_x, c_y) \) is the 2D...
center computed by projecting the query prediction onto the image plane, \( r \) is the radius of the minimum circumscribed circle of the projected corners of the 3D bounding box, and \( \sigma \) is the hyper-parameter to modulate the bandwidth of the Gaussian distribution. Then this weight map is element-wisely multiplied with the cross-attention map among all the attention heads. In this way, each object query only attends to the related region around the projected 2D box, so that the network can learn where to select image features based on the input LiDAR features better and faster. The visualization of the attention map is shown in Fig. 3. The network typically tends to focus on the foreground pixels close to the object center and ignore the irrelevant pixels, providing valuable semantic information for object classification and bounding box regression. After SMCA, we use another FFN to produce the final bound box predictions using the object queries containing both LiDAR and image information.

### 3.5. Label Assignment and Losses

Following DETR [23], we find the bipartite matching between the predictions and ground truth objects through the Hungarian algorithm [13], where the matching cost is defined by a weighted sum of classification, regression, and IoU cost:

\[
C_{\text{match}} = \lambda_1 L_{\text{cls}}(p, \hat{b}) + \lambda_2 L_{\text{reg}}(b, \hat{b}) + \lambda_3 L_{\text{iou}}(b, \hat{b}),
\]

where \( L_{\text{cls}} \) is the binary cross entropy loss, \( L_{\text{reg}} \) is the L1 loss between the predicted BEV centers and the ground-truth centers (both normalized in \([0,1]\)), and \( L_{\text{iou}} \) is the IoU loss [65] between the predicted boxes and ground-truth boxes. \( \lambda_1, \lambda_2, \lambda_3 \) are the coefficients of the individual cost terms. We provide sensitivity analysis of these terms in the supplementary Sec. C. Since the number of predictions is usually larger than that of GT boxes, the unmatched predictions are considered as negative samples. Given all matched pairs, we compute a focal loss [18] for the classification branch. The bounding box regression is supervised by an L1 loss for only positive pairs. For the heatmap prediction, we adopt a penalty-reduced focal loss following CenterPoint [58]. The total loss is the weighted sum of losses for each component. We adopt the same label assignment strategy and loss formulation for both decoder layers.

### 3.6. Image-Guided Query Initialization

Since our object queries are currently selected using only LiDAR features, it potentially leads to sub-optimality in terms of the detection recall. Empirically, our model already achieves high recall and shows superior performance over the baselines (Sec. 5). Nevertheless, to further leverage the ability of high-resolution images in detecting small objects and make our algorithm more robust against sparse LiDAR point clouds, we propose an image-guided query initialization strategy, which selects object queries leveraging both the LiDAR and camera information.

Specifically, we generate a LiDAR-camera BEV feature map \( F_{LC} \) by projecting the image features \( F_C \) onto the BEV plane through cross attention with LiDAR BEV features \( F_L \). Inspired by [32], we use the multiview image features collapsed along the height axis as the key-value sequence of the attention mechanism, as shown in Fig. 4. The collapsing operation is based on the observation that the relation between BEV locations and image columns can be established easily using camera geometry, and usually there is at most one object along each image column. Therefore, collapsing along the height axis can significantly reduce the computation without losing critical information. Although some fine-grained image features might be lost during this process, it already meets our need as only a hint on potential object positions is required. Afterward, similar to Sec. 3.2, we use \( F_{LC} \) to predict the heatmap, which is averaged with the LiDAR-only heatmap \( \hat{S} \) as the final heatmap \( \hat{S}_{LC} \). Using \( \hat{S}_{LC} \) to select and initialize the object queries, our model is able to detect objects that are difficult to detect in LiDAR point clouds.

Note that proposing a novel method to project the image features onto the BEV plane is beyond the scope of this paper. We believe that our method could benefit from more research progress [26, 32, 33] in this direction.

### 4. Implementation Details

**Training.** We implement our network in PyTorch [25] using the open-sourced MMDetection3D [6]. For nuScenes, we use the DLA34 [61] of the pretrained CenterNet as our 2D backbone and keep its weights frozen during training, following [48]. We set the image size to \( 448 \times 800 \), which performs comparably with full resolution \((896 \times 1600)\). VoxelNet [53, 68] is chosen as our 3D backbone. Our training consists of two stages: 1) We first train the 3D backbone with the first decoder layer and FFN for 20 epochs, which only needs the LiDAR point clouds as input and produces the initial 3D bounding box predictions. We adopt the same data augmentation and training schedules as prior LiDAR-only works [58, 69]. Note that we also find the copy-and-paste augmentation strategy [53] benefits the convergence
but could disturb the real data distribution, so we do not use any test time augmentation. Please find detailed results here. In Table 1, we report the results on the nuScenes website, which are better than what they reported in their paper. Note that CenterPoint and PointAugmenting utilize double-flip testing while we do not use any test time augmentation. Please find detailed results here.²

### 5. Experiments

In this section, we first make comparisons with the state-of-the-art methods on nuScenes and Waymo. Then we conduct extensive ablation studies to demonstrate the importance of each key component of TransFusion. Moreover, we design two experiments to show the robustness of our TransFusion against inferior image conditions. Besides TransFusion, we also include a model variant, which is based on the first training stage, i.e., producing the initial bounding box predictions using only point clouds. We denote it as TransFusion-L and believe it can serve as a strong baseline for LiDAR-only detection. We provide the qualitative results in supplementary Sec. I.

#### nuScenes Dataset

The nuScenes dataset is a large-scale autonomous driving dataset for 3D detection and tracking, consisting of 700, 150, and 150 scenes for training, validation, and testing, respectively. ‘C.V.’ and ‘T.C.’ are short for construction vehicle, pedestrian, and traffic cone, respectively. ‘L’ and ‘C’ represent LiDAR and Camera, respectively. The best results are in boldface. For FusionPainting [52], we report the results on the nuScenes website, which are better than what they reported in their paper. Note that CenterPoint and PointAugmenting utilize double-flip testing while we do not use any test time augmentation. Please find detailed results here.²

#### 5.1 Main Results

**nuScenes Results.** We submitted our detection results to the nuScenes evaluation server. Without any test time augmentation or model ensemble, our TransFusion outperforms all competing non-ensembled methods on the nuScenes leaderboard at the time of submission. As shown in Table 1, our TransFusion-L already outperforms the state-of-the-art LiDAR-only methods by a significant margin (+5.2% mAP, +2.9% NDS) and even surpasses some multi-modality methods. We ascribe this performance gain to the relation modeling power of the transformer decoder as well as the proposed query initialization strategies, which are ablated in Sec. 5.3. Once enabling the proposed fusion components, our TransFusion receives remarkable performance boost (+3.4% mAP, +1.5% NDS) and outperforms all the previous methods, including FusionPainting [52], which

### Table 1. Comparison of SOTA methods on the nuScenes test set. ‘C.V.’, ‘Ped.’, and ‘T.C.’ are short for construction vehicle, pedestrian, and traffic cone, respectively. ’L’ and ‘C’ represent LiDAR and Camera, respectively. The best results are in boldface. For FusionPainting [52], we report the results on the nuScenes website, which are better than what they reported in their paper. Note that CenterPoint and PointAugmenting utilize double-flip testing while we do not use any test time augmentation. Please find detailed results here.²

| Method            | Modality | Voxel Size (m) | mAP | NDS | Car | Truck | C.V. | Bus | Trailer | Barrier | Motor. | Bike | Ped. | T.C. |
|-------------------|----------|----------------|-----|-----|-----|-------|------|-----|---------|---------|--------|------|------|------|
| PointPillar [14]  | L        | (0.2, 0.2, 0.2)| 40.1| 53.0| 76.0| 31.0 | 11.3 | 32.1 | 36.6    | 56.4    | 34.2   | 14.0 | 64.0 | 45.6 |
| CBGS [69]         | L        | (0.1, 0.1, 0.1)| 52.8| 63.3| 81.1| 48.5 | 15.9 | 49.8 | 42.9    | 65.7    | 51.5   | 22.3 | 80.1 | 70.9 |
| CenterPoint [58]  | L        | (0.075, 0.075, 0.2)| 60.3| 67.3| 85.2| 53.5 | 20.0 | 63.6 | 56.0    | 71.1    | 59.5   | 30.7 | 84.6 | 78.4 |
| PointPainting [47]| L        | (0.2, 0.2, 0.2)| 56.4| 58.1| 87.9| 53.8 | 15.8 | 56.2 | 57.5    | 60.2    | 41.5   | 24.1 | 73.3 | 62.4 |
| 3D-CVF [60]       | L        | (0.05, 0.05, 0.2)| 52.7| 62.3| 83.0| 45.0 | 15.9 | 48.8 | 49.6    | 65.9    | 51.2   | 30.4 | 74.2 | 62.9 |
| PointAugmenting [48]| L     | (0.075, 0.075, 0.2)| 66.8| 71.0| 87.5| 57.3 | 28.0 | 65.2 | 60.0    | 72.6    | 72.4   | 53.0 | 79.7 | 83.6 |
| MVP [59]          | L        | (0.075, 0.075, 0.2)| 66.4| 70.5| 86.8| 58.5 | 26.1 | 67.4 | 57.3    | 74.8    | 70.0   | 49.3 | 89.1 | 85.0 |
| FusionPainting [52]| L     | (0.075, 0.075, 0.2)| 68.1| 71.6| 87.1| 60.8 | 30.0 | 68.5 | 61.7    | 71.8    | 74.7   | 53.5 | 88.3 | 85.0 |
| TransFusion-L     | L        | (0.075, 0.075, 0.2)| 65.5| 76.5| 86.2| 56.7 | 28.2 | 66.3 | 58.6    | 78.2    | 69.3   | 44.2 | 86.7 | 82.5 |
| TransFusion       | L        | (0.075, 0.075, 0.2)| 68.9| 71.7| 87.1| 60.0 | 33.1 | 68.3 | 60.8    | 78.1    | 73.6   | 52.9 | 88.4 | 86.7 |

²https://www.nuscenes.org/object-detection
uses extra data to train their segmentation sub-networks. Moreover, thanks to our soft-association mechanism, TransFusion is robust to inferior image conditions including degenerated image quality and sensor misalignment, as shown in the next section.

**Waymo Results.** We report the performance of our model over all three classes on Waymo validation set in Table 2. Our fusion strategy improves the mAPH of pedestrian and cyclist classes by 0.3 and 1.5x, respectively. We suspect two reasons for the relatively small improvement brought by the image components. First, the semantic information of images might have less impact on the coarse-grained categorization of Waymo. Second, the initial bounding boxes from the first decoder layer are already with accurate locations since the point clouds in Waymo are denser than those in nuScenes (see more discussions in supplementary Sec. H). Note that CenterPoint achieves a better performance with the same tracking algorithms adopted by CenterPoint. We refer readers to the original paper [58] for details. As shown in Table 3, our model significantly outperforms CenterPoint but relies on CenterPoint to get the predictions outside its FOV for a full-region detection, making their system less flexible.

## 5.2. Robustness against Inferior Image Conditions

We design three experiments to demonstrate the robustness of our proposed fusion module. Since the nuScenes test set only allows at most three submissions, all the experiments are conducted on the validation set. For fast iteration, we reduce the first stage training to 12 epochs and remove the fade strategy. All the other parameters are the same as the main experiments. To avoid overstatement, we additionally build two baseline LiDAR-camera detectors by equipping our TransFusion-L with two representative fusion methods on nuScenes: fusing LiDAR and image features by point-wise concatenation (denoted as CC) and the fusion strategy of PointAugmenting (denoted as PA).

### Nighttime

We first split the validation set into daytime and nighttime based on scene descriptions provided by nuScenes and show the performance gain under different situations in Table 4. Our method brings a much larger performance gain during nighttime, where the worse lighting negatively affects the hard-association based fusion strategies CC and PA.

### Degenerated Image Quality

In Table 5, we randomly drop several images for each frame by setting the image features of such images to zero during inference. Since both CC and PA fuse LiDAR and image features in a tightly-coupled way, their performance drops significantly when some images are not available during inference. In contrast, our TransFusion is able to maintain a high mAP under all cases. When all the six images are not available, CC and PA suffer from 23.8% and 17.2% mAP degradation, respectively, while TransFusion still keeps the mAP at a competitive level of 61.7%. This advantage comes from the sequential design and the attentive fusion strategy, which first generates initial predictions based on LiDAR data and then only gathers useful information from image features adaptively. Moreover, we could even directly disable the fusion module if the camera malfunctioning is known, such that the whole system could still work seamlessly in a LiDAR-only mode.

| # Dropped Images | 0   | 1   | 3   | 6   |
|------------------|-----|-----|-----|-----|
| CC               | 63.3| 59.8| 50.9| 49.9| 49.4 (+0.2) | 63.4 (+3.1) |
| PA               | 64.2| 61.6| 55.4| 47.0| 51.0 (+1.8) | 64.3 (+4.0) |
| TransFusion      | 65.6| 65.1| 63.9| 61.7| 55.2 (+6.0) | 65.7 (+5.4) |

Table 5. mAP under different numbers of dropped images. The number in each bracket is the mAP drop from the standard input.

### Sensor Misalignment

We evaluate different fusion methods under a setting where LiDAR and images are not well-calibrated following RoarNet [39]. Specifically, we randomly add a translation offset to the transformation matrix from camera to LiDAR sensor. As shown in Fig. 5, TransFusion achieves better robustness against the calibration error compared with other fusion methods. When two sensors
are misaligned by 1 m, the mAP of our model only drops by 0.49%, while the mAP of PA and CC degrades by 2.33% and 2.85%, respectively. In our method, the calibration matrix is only used for projecting the object queries onto images, and the fusion module is not strict with the projected locations since the attention mechanism could adaptively find the relevant image features around based on the context information. The insensitivity towards sensor calibration also enables the possibility to pipelining the 2D and 3D fusion components by removing the feature fusion module such that the LiDAR features are fused with the features from the previous images [47].

5.3. Ablation Studies

We conduct ablation studies on the nuScenes validation set to study the effectiveness of the proposed components.

![Figure 5. mAP under sensor misalignment cases. The X axis refers to the translational discrepancy between two sensors.](image)

Table 6. Ablation of the query initialization module. C.A.: category-aware; I.D.: input-dependent.

| C.A. | I.D. | #Layers | #Epochs | mAP | NDS |
|------|------|---------|---------|-----|-----|
| a)   | ✓    | 1       | 12      | 60.0| 66.8|
| b)   | ✓    | 1       | 12      | 54.3| 63.9|
| c)   | ✓    | 3       | 12      | 59.9| 67.1|

Table 8. mAP breakdown over BEV distance between object center and ego vehicle in meters.

| Dist (m) | mAP | NDS |
|----------|-----|-----|
| <15m     | 70.4| 59.5|
| 15-30m   | 75.5 (+5.1)| 66.9 (+7.4)|
| >30m     | 43.7 (+8.4)|

Table 7. Ablation of the proposed fusion components. 18.34 represents the parameter size of the 2D backbone. The latency is measured on an Intel Core i7 CPU and a Titan V100 GPU. For CenterPoint, we use re-implementations in MMDetection3D.

Table 8. mAP breakdown over BEV distance between object center and ego vehicle in meters.

Table 7. Ablation of the proposed fusion components. 18.34 represents the parameter size of the 2D backbone. The latency is measured on an Intel Core i7 CPU and a Titan V100 GPU. For CenterPoint, we use re-implementations in MMDetection3D.

6. Conclusion

We have designed an effective and robust transformer-based LiDAR-camera 3D detection framework with a soft-association mechanism to adaptively determine where and what information should be taken from images. Our TransFusion sets the new state-of-the-art results on the nuScenes detection and tracking leaderboards, and shows competitive results on Waymo detection benchmark. The extensive ablative experiments demonstrate the robustness of our method against inferior image conditions. We hope that our work will inspire further investigation of LiDAR-camera fusion for driving-scene perception, and the application of a soft-association based fusion strategy to other tasks, such as 3D segmentation.

Acknowledgements. This work is supported by Hong Kong RGC (GRF 16206819, 16203518, T22-603/15N), Guangzhou Okay Information Technology with the project GZETDZ18EG05, and City University of Hong Kong (No. 7005729).
References

[1] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Gi-an Carlo Baldan, and Oscar Beijbom. nuScenes: A multi-modal dataset for autonomous driving. CVPR, 2020.

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

[3] Qi Chen, Lin Sun, Ernest C. H. Cheung, and A. Yuille. Every View Counts: Cross-view consistency in 3d object detection with hybrid-cylindrical-spherical voxelization. NeurIPS, 2020.

[4] Qi Chen, Lin Sun, Zhixin Wang, K. Jia, and A. Yuille. Object as Hotspots: An anchor-free 3d object detection approach via firing of hotspots. CVPR, 2020.

[5] Xiaozhi Chen, Huimin Ma, Jixiang Wan, B. Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. CVPR, 2017.

[6] MMDetection3D Contributors. MMDetection3D: OpenMMLab next-generation platform for general 3D object detection. https://github.com/open-mmlab/mmdetection3d, 2020.

[7] M. Everingham, L. Gool, Christopher K. I. Williams, J. Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. IJCV, 2009.

[8] Lue Fan, Xuan Xiong, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. RangeDet: In defense of range view for lidar-based 3d object detection. ICCV, 2021.

[9] Peng Gao, Minghang Zheng, Xiaogang Wang, Jifeng Dai, and Hongsheng Li. Fast convergence of detr with spatially modulated co-attention. ICCV, 2021.

[10] Tengteng Huang, Zhe Liu, Xiwu Chen, and X. Bai. EPNet: Enhancing point features with image semantics for 3d object detection. ECCV, 2020.

[11] Aleksandr Kim, Aljosa Osep, and Laura Leal-Taixé. EagerMOT: 3D multi-object tracking via sensor fusion. ICRA, 2021.

[12] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L. Waslander. Joint 3D proposal generation and object detection from view aggregation. IROS, 2018.

[13] H. Kuhn. The hungarian method for the assignment problem. Naval Research Logistics Quarterly, 1955.

[14] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. PointPillars: Fast encoders for object detection from point clouds. CVPR, 2019.

[15] Zhichao Li, Feng Wang, and Naiyan Wang. LiDAR R-CNN: An efficient and universal 3D object detector. NeurIPS, 2020.

[16] Ming Liang, Binh Yang, Yun Chen, Rui Hu, and R. Urtasun. Multi-task multi-sensor fusion for 3d object detection. CVPR, 2019.

[17] Ming Liang, Binh Yang, Shenlong Wang, and R. Urtasun. Deep continuous fusion for multi-sensor 3d object detection. ECCV, 2018.

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

[19] Zili Liu, Guodong Xu, Honghui Yang, Minghao Chen, Kuoliang Wu, Zheng Yang, Haifeng Liu, and Deng Cai. Suppress-and-refine framework for end-to-end 3d object detection. arXiv, 2021.

[20] Ze Liu, Zheng Zhang, Yue Cao, Han Hu, and Xin Tong. Group-free 3d object detection via transformers. ICCV, 2021.

[21] Jiageng Mao, Yujing Xue, Minzhe Niu, Haoyue Bai, Jiashui Feng, Xiaodan Liang, Hang Xu, and Chunjing Xu. Voxel transformer for 3d object detection. ICCV, 2021.

[22] Gregory P. Meyer, Jake Charland, Darshan Hegde, Ankita Gajanan Laddha, and Carlos Vallespi-Gonzalez. Sensor fusion for joint 3d object detection and semantic segmentation. CVPR, 2019.

[23] Ishan Misra, Rohit Girdhar, and Armand Joulin. An End-to-End Transformer Model for 3D Object Detection. ICCV, 2021.

[24] Xuran Pan, Zhuofan Xia, Shijii Song, L. Li, and Gao Huang. 3d object detection with pointformer. CVPR, 2021.

[25] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. NeurIPS-W, 2017.

[26] Jonah Philion and S. Fidler. Lift, Splat, Shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. ECCV, 2020.

[27] C. Qi, Xinlei Chen, O. Litany, and L. Guibas. ImVoteNet: Boosting 3d object detection in point clouds with image votes. CVPR, 2020.

[28] C. Qi, O. Litany, Kaiming He, and L. Guibas. Deep hough voting for 3d object detection in point clouds. ICCV, 2019.

[29] C. Qi, W. Liu, Chenxia Wu, Hao Su, and L. Guibas. Frustum pointnets for 3D object detection from rgb-d data. CVPR, 2018.

[30] C. Qi, Hao Su, Kaichun Mo, and L. Guibas. PointNet: Deep learning on point sets for 3d classification and segmentation. CVPR, 2017.

[31] Shaoqing Ren, Kaiming He, Ross B. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. TPAMI, 2015.

[32] Thomas Roddick and R. Cipolla. Predicting semantic map representations from images using pyramid occupancy networks. CVPR, 2020.

[33] Thomas Roddick, Alex Kendall, and R. Cipolla. Orthographic feature transform for monocular 3d object detection. BMVC, 2019.

[34] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning feature matching with graph neural networks. CVPR, 2020.

[35] Huai-Long Sheng, Sijia Cai, Yuan Liu, Bing Deng, Jianqiang Huang, Xiansheng Hua, and Min-Jian Zhao. Improving 3D object detection with channel-wise transformer. CVPR, 2021.

[36] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. PV-RCNN: Point-voxel feature set abstraction for 3d object detection. CVPR, 2020.
Supplementary Material

The supplementary document is organized as follows:

- Sec. A depicts the detailed network architectures of our transformer decoder layers.
- Sec. B provides the implementation details of TransFusion and the training settings on nuScenes and Waymo.
- Sec. C reports our sensitivity analysis of the matching cost during label assignment.
- Sec. D presents the effect of NMS on TransFusion and CenterPoint.
- Sec. E provides the results of using PointPillars as our 3D backbone.
- Sec. F discusses the effect of 2D backbone in TransFusion.
- Sec. G shows the results with different number of object queries.
- Sec. H discusses the performance gain of image information on Waymo.
- Sec. I provides visualization results on the nuScenes and Waymo datasets.

A. Network Architectures

The detailed architectures of the respective transformer decoder layers for initial bounding box prediction and LiDAR-camera fusion are shown in Fig. 6. Following [20], we adopt the common practice of transformer except that we use the learned positional encoding instead of the fixed sine positional encoding [46]. For the image-guided query initialization module, we use the LiDAR BEV features as query sequence and collapsed image features as key-value sequence, and only perform cross attention to save the computation cost. Our model can benefit from the efficient attention mechanisms in recent works such as [71].

B. Implementation Details

Our implementation is based on the open-sourced codebase MMDetection3D [6], which provides many popular 3D detection methods, including PointPillar, VoxelNet, and CenterPoint. For our 3D backbone, we set its hyperparameters according to CenterPoint-Voxel’s official implementation. For the transformer-decoder-based detection head, the hidden dimension $d$ is set to 256 and dropout is set to 0.1. We use $N = 200$ and 300 queries for nuScenes and Waymo since the max numbers of objects in one frame are 142 and 185, respectively. Since our object queries are non-parametric, we are able to modify the number of queries during inference. We provide the ablations on the number of object queries in Sec. G. When selecting object queries from the heatmap, we pick local maximum pixels whose values are greater than or equal to their 8-connected neighbors. To avoid mistakenly suppress nearby instances for small objects, we do not check the local maximum for pedestrian and traffic cone on nuScenes and for pedestrian and cyclist on Waymo. Following PointAugmenting [48], we adopt DLA34 of CenterNet pre-trained on monocular 3D detection task as our 2D backbone for nuScenes. Since there is no public available 2D backbone pre-trained on Waymo dataset, we train a Faster-RCNN [31] using the 2D labels provided by Waymo and use its ResNet50 and FPN as our 2D backbone. We freeze the weight of image backbone during training and set the image resolution to half of the full resolution for both nuScenes and Waymo to speed up the training process.

nuScenes. Following the common practice, we transform previous ten LiDAR sweeps into the current frame to produce a denser point cloud input for both training and inference. The detection range is set to $[-51.2m, 51.2m]$ for X and Y axes, and $[-5m, 3m]$ for Z axis. The maximum numbers of non-empty voxels for training and inference are set to 120,000 and 160,000, respectively. In terms of the data augmentation strategy, we adopt random flipping along both X and Y axes, global scaling with a random factor from [0.9, 1.1], global rotation between $[-\pi/8, \pi/8]$, as well as the copy-and-paste augmentation [53]. We follow CBGS [69] to perform class-balanced sampling and optimize the network using the AdamW optimizer with one-cycle learning rate policy, with max learning rate 0.001, weight decay 0.01, and momentum 0.85 to 0.95. We train the 3D backbone with the first decoder layer and FFN for

![Figure 6. Left: Architecture of the transformer decoder layer for initial bounding box prediction. Right: Architecture of the transformer decoder layer for image fusion.](image-url)
20 epochs, and the LiDAR-camera fusion components for 6 epochs with batch size of 16 using 8 Tesla V100 GPUs. We use gradient clipping at an $l_2$ norm of 0.1 to stabilize the training process. The weighting coefficients of heatmap loss, classification loss, and regression loss are 1.0, 1.0 and 0.25, respectively. The coefficients of matching cost $\lambda_1$, $\lambda_2$, $\lambda_3$ are 0.15, 0.25, 0.25, respectively. The sensitivity analysis of the matching cost coefficients is provided in the next section.

**Waymo.** For Waymo, we only use a single sweep as input and set the detection range to $[-75.2m, 75.2m]$ for X and Y axes, and $[-2m, 4m]$ for Z axis. The maximum number of non-empty voxels is set to 150,000. We adopt the same training strategies as nuScenes except: 1) The first stage training last for 36 epochs with batch size of 16 under a max learning rate of 0.001. 2) The weighting coefficient of regression loss is changed to 2.0, following CenterPoint. 3) The matching cost coefficients are set to 0.075, 0.25, 0.25, respectively.

**C. Label Assignment Strategy**

Following DETR, we perform label assignment by finding the bipartite matching between predictions and ground-truth objects through the Hungarian algorithm. In Table 12, we study the effect of the coefficient of each matching cost term on the detection performance of TransFusion-L. Since the matching results are only decided by the relative values of individual matching cost terms, we keep $\lambda_2 = 0.25$ and try different values for $\lambda_1$ and $\lambda_3$. As shown in Table 12, we find the network suffers from a convergence issue when the coefficient of the classification cost is too large, and the detection performance is not sensitive to the coefficient within a reasonable range. Since the weighting coefficients of the matching cost need some tuning, we additionally propose a heuristic label assignment strategy (denoted as Heuristic) to avoid hyper-parameter tuning. The Heuristic assignment strategy follows the simple rules: each GT box will only be assigned to the predicted box with the same category and the smallest center distance. If a conflict appears, the predicted box will be matched to the closer GT box. In this way, we also find the one-to-one matching between prediction and GT but with some GT boxes unused. We find Heuristic works quite well for uncrowded scenes but for objects in a crowded scene, such as pedestrian or traffic cone in nuScenes, it is unable to prevent duplicate predictions, which will be further explained in Sec. D.

**D. Effect of NMS**

Recently, many works [2, 45, 71] in 2D detection have focused on removing the last non-differentiable component, Non-Maximum Suppression (NMS), in the detection pipeline. OneNet [42] systematically compares the end-to-end detectors with non-end-to-end detectors, and claims that the one-to-one positive sample assignment as well as classification cost in the matching cost are the two key factors in producing a large score gap between duplicate prediction and building an end-to-end detection system without NMS. We refer readers to the original paper [42] for more details. Following DETR’s label assignment strategy, our method naturally satisfies these two requirements and do not need NMS. As show in Table 10, our method still maintains a high mAP without NMS while CenterPoint drops about 12% mAP. This advantage eliminates the hand-designed NMS post-processing step and makes TransFusion more practical and handy for deployment to new scenarios in the real applications. Besides, since the Heuristic strategy mentioned in Sec. C does not have classification cost involved in the assignment stage, this strategy is unable to produce a large score gap between duplicate prediction. This is why it does not perform as well as the baseline model on Pedestrian and Traffic cone.

**E. Pillar-based 3D Backbone**

To demonstrate our framework’s compatibility with other 3D backbones, we use PointPillars as our 3D backbone to produce the BEV features, while keeping all the other settings the same as the main experiments. The voxel size is set to $(0.2m, 0.2m)$. As shown in Table 11, our model outperforms CenterPoint by a remarkable margin under the same pillar-based backbone, showing its great generalization ability.
### Table 11. Results on nuScenes validation set.

| Model          | mAP   | NDS   | mAP   | NDS   |
|----------------|-------|-------|-------|-------|
| PointPillars   | 50.3  | 60.2  | 59.6  | 66.8  |
| VoxelNet       |       |       |       |       |
| CenterPoint    | 54.5  | 62.7  | 65.1  | 70.1  |
| TransFusion-L  | 58.3  | 64.5  | 67.5  | 71.3  |
| TransFusion    |       |       |       |       |

### F. Discussions of the 2D Network

Current multi-modality detection models usually employ CNN features from 2D networks pre-trained on different tasks (i.e., segmentation or detection) and with different resolution (i.e., different levels from ResNet or DLA). There is no existing work analyzing what kind of image features are most useful for a 3D detection model, and using improper image features might prevent the release of the potential for a multi-modality detection system. We believe that the sequential design of our method enables a flexible and off-the-shelf experiment base to explore the effects of different image features. Therefore, we explore this question by fixing the 3D backbone with the first decoder layer and performing the second stage training with different image features.

### Table 12. mAP under different 2D backbones. ‘Det.’ and ‘Seg.’ are short for detection and segmentation, respectively. For DLA34 on monocular 3D detection, we acquire the CenterNet pre-trained on nuScenes following PointAugmenting. For ResNet50 on instance segmentation, we acquire the model pre-trained on nuImages from MMDetection3D. For ResNet50 on 2D detection, we train the model by ourself using MMDetection3D since there is no open-sourced model weights.

| Arch.           | Task                   | mAP | NDS |
|-----------------|------------------------|-----|-----|
| DLA34           | Monocular 3D Det.      | 65.6| 69.7|
| R50+FPN Level 0 | 2D Det.                | 66.4| 70.1|
| R50+FPN Level 0 | 2D Instance Seg.       | 66.6| 70.1|
| R50+FPN Level 1 | 2D Instance Seg.       | 66.5| 70.1|
| R50+FPN Level 2 | 2D Instance Seg.       | 66.3| 70.0|
| R50+FPN Level 3 | 2D Instance Seg.       | 65.4| 69.6|

From Table 12, we find image features of the 2D instance segmentation model bring the largest performance boost compared with that of detection models. In terms of different levels of the feature pyramid, the feature map of level 0 (stride 4) brings a slightly larger performance gain. We suspect the image features at that level contain more fine-grained information which is important to distinguish small or distant objects. Image features from level 1 (stride 8) and level 2 (stride 16) can bring a similar gain with a smaller resolution of feature maps, while image features from level 3 (stride 32) yields a drop of 1.2% mAP in comparison with the level-0 counterpart due to the row resolution.

### G. Adap Queries at Test Time

Unlike DETR, our object queries are non-parametric and input-dependent. These two characteristics allow us to use different numbers of queries during inference. It could be useful when we have some prior knowledge about a scene, such as its crowdedness. In Table 13, we provide the performance evaluated under different object queries for the same model trained under \( N = 200 \) queries. Note that we use \( N = 200 \) to get all the numbers in the main text for its better performance-efficiency trade-off and use \( N = 300 \) for online submission for a slightly better performance.

| #queries | 100   | 200   | 300   | 500   |
|----------|-------|-------|-------|-------|
| mAP      | 64.2  | 65.6  | 65.9  | 66.0  |
| NDS      | 69.2  | 69.7  | 69.8  | 69.8  |

Table 13. Results with different numbers of queries. We keep the model unchanged and only use different numbers of queries for evaluation.

### H. Discussions on Waymo

Our TransFusion brings smaller performance gain over TransFusion-L on Waymo compared with that on nuScenes. We speculate that this is mainly due to the following two reasons:

1. As shown in Table 1, compared with TransFusion-L, TransFusion brings the largest performance increase for bicycle (+8.7%), motorcycle (+5.4%), and construction vehicle (+4.9%) in terms of mAP on nuScenes. Due to the geometrical ambiguity, objects from the above three categories are difficult to distinguish using LiDAR information only, and thus the semantic information of images is extraordinarily important for more accurate classification. However, the categorization of Waymo is rather coarse-gained (i.e., vehicle, pedestrian, cyclist), which hides the improvement brought by the image information to some extent.

2. The LiDAR point clouds in Waymo are much denser than those in nuScenes (see Sec. I for visualizations). Thus the bounding box predictions of TransFusion-L are already with accurate localization, which reduces the room for further improvement by image fusion.

### I. Qualitative Results

We first compare the detection results of TransFusion and TransFusion-L on the nuScenes dataset in Fig. 7. The image information improves the performance of the LiDAR-only baseline through reducing the False Positive and False Negative. More visualization results on Waymo and nuScenes datasets are shown in Fig. 8 and Fig. 9, respectively.
Figure 7. Qualitative comparison between TransFusion-L and TransFusion on the nuScenes dataset. Blue boxes and green boxes are the predictions and ground-truth, respectively. Best viewed with color and zoom-in.
Figure 8. Visualization of detection results on the Waymo dataset. Our model predicts highly accurate bounding boxes for nearby vehicles and pedestrians (note that cyclists are very rare in the dataset) and also handles objects with severe occlusion. Blue boxes and green boxes are the predictions and ground-truth, respectively. Best viewed with color and zoom-in.

Figure 9. Visualization of detection results on the nuScenes dataset. Compared with Waymo, nuScenes has much sparser point clouds and smaller objects such as traffic cones. Nevertheless, our model successfully detects such objects even with only few points observed. Blue boxes and green boxes are the predictions and ground-truth, respectively. Best viewed with color and zoom-in.