FisheyeBEVSeg: Surround View Fisheye Cameras based Bird's-Eye View Segmentation for Autonomous Driving

Senthil Yogamani*, David Unger†, Venkatraman Narayanan‡, Varun Ravi Kumar‡

Abstract

Semantic segmentation is an effective way to perform scene understanding. Recently, segmentation in 3D Bird's Eye View (BEV) space has become popular as its directly used by drive policy. However, there is limited work on BEV segmentation for surround-view fisheye cameras, commonly used in commercial vehicles. As this task has no real-world public dataset and existing synthetic datasets do not handle amodal regions due to occlusion, we create a synthetic dataset using the Cognata simulator comprising diverse road types, weather, and lighting conditions. We generalize the BEV segmentation to work with any camera model; this is useful for mixing diverse cameras. We implement a baseline by applying cylindrical rectification on the fisheye images and using a standard LSS-based BEV segmentation model. We demonstrate that we can achieve better performance without undistortion, which has the adverse effects of increased runtime due to pre-processing, reduced field-of-view, and resampling artifacts. Further, we introduce a distortion-aware learnable BEV pooling strategy that is more effective for the fisheye cameras. We extend the model with an occlusion reasoning module, which is critical for estimating in BEV space. Qualitative performance of FisheyeBEVSeg is showcased in the video at https://youtu.be/HfTPwMabgS0.

1. Introduction

The semantic Bird's Eye View (BEV) segmentation task aims to rasterize the real world into a 2D grid, with each cell containing semantic information regarding the corresponding real-world object. BEV grid generation has emerged as a pivotal task in Automated Driving (AD) and Robotics [1, 2]. Four wide-angle fisheye cameras are commonly used in AD systems for near-field sensing. However, most academic research in AD focuses on pinhole cameras, largely due to their prevalence in open-source datasets. Furthermore, fisheye cameras introduce substantial distortions to the images, making it challenging to extend standard pinhole approaches. The literature in various fisheye perception tasks [3–8] indicate that special attention and design are needed to handle the large radial distortion.

The work on Lift-Splat-Shoot (LSS) [9] forms the baseline for generating a BEV map from standard pinhole cameras using geometry. Many derivative works adapt LSS for downstream AD tasks in the BEV space. This work proposes a novel architecture that performs Semantic Segmentation in BEV Space for fisheye cameras. Our main contributions include:

- Creation of a fisheye BEV segmentation dataset with occlusion masks using a commercial-grade simulator.
- Design of a novel distortion-aware learnable pooling strategy using camera intrinsics for adaptation.
- Generalized framework for BEV semantic segmentation from raw images supporting various camera models.
- A end-to-end multi-task model that provides semantic classes and occlusion reasoning in ambiguous scenarios.

2. Method

This section presents our approach to learning bird's-eye-view representations of scenes from image data captured by a surround fisheye camera rig. Our goal is, given four surround-view fisheye images $X_k \in R^{3\times H\times W}$ each with an extrinsic matrix $E_k \in R^{3\times 4}$ and an intrinsic matrix $I_k \in R^{3\times 3}$, we seek to find a rasterized representation of the scene in the BEV coordinate frame $y \in R^{C\times X\times Y}$ with semantic information for each cell. The end-to-end architecture of FisheyeBEVSeg is outlined in Figure 2.

1. Fisheye Camera BEV features: First, similar to the LSS [9] approach, we use the camera parameters to project the image features from the image encoder into the 3D world. The camera parameters map the image feature to the corresponding ray in the BEV map, while the estimated depth determines the distance. For this to work in fisheye cameras, we convert the camera features into a direction vector for each feature projected into 3D space. The camera features in pixel coordinates, u, v, are converted to camera co-ordinates, x, y, using the camera intrinsic matrix

*Automated Driving, QT Technologies Ireland Limited.
†Internship at Qualcomm Technologies International GmbH.
‡Automated Driving, Qualcomm Technologies, Inc.

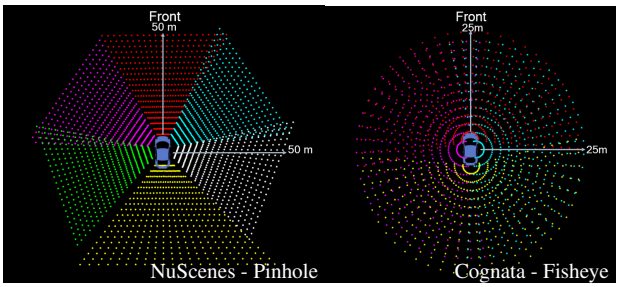

Figure 1. **The visualization of BEV Grid space from each camera pixel**. Left: NuScenes and Right: Cognata dataset.

(I_k). As depicted in Equation 1, using the Kannala-Brandt model [10], we generate spherical angles (θ, φ) from the camera coordinate features.

$$\varphi = \arctan\left(\frac{y}{x}\right) \quad r = \sqrt{x^2 + y^2}$$
$$\theta = p_1.r + p_2.r^2 + p_3.r^3 + ... + p_9.r^9 \tag{1}$$

We use a generic equation for radial distortion angle (θ) here. Our method generalizes the inverse mappings of radial distortion models, such as *Polynomial, UCM, eUCM, Rectilinear, Stereographic, Double Sphere* [11].

The BEV grid is generated by *Splatting* [9] the depth estimates with the direction vectors, (X, Y, Z). Unlike traditional cameras with a narrower field of view, fisheye lenses capture a wide-angle view, resulting in significant overlap between adjacent images. This overlap means that features from multiple camera perspectives contribute to the same areas in the BEV grid, as seen in Figure 1, creating a rich and redundant source of information. The depth resolution of the BEV grid in the figure has been altered for visualization. This redundancy can be leveraged to enhance the robustness and accuracy of the BEV representation, as features from multiple viewpoints provide complementary information about the scene. Therefore, in the fisheye BEV grid, the challenge lies in handling distortion and effectively integrating and leveraging the overlapping features to construct a comprehensive representation of the environment. The architectural changes needed for Fisheye cameras are explained in Section 3.

2. Distortion-aware Learnable Pooling: The BEV features from each camera sensor are pooled into a single BEV grid. Most current methods use the *BEV Pooling* [12] strategy to associate the camera BEV features to the BEV grid along with a symmetric function (e.g., mean, max, and sum) to aggregate the features within each BEV grid. However, the features in overlapping regions are treated equally and do not contextualize the sensor parameters. Our experiments showcased that it does not translate well to surround-view fisheye cameras because of the non-linear scaling and larger FOV. Also, symmetric aggregation is disadvantageous when the surround fisheye cameras do not share the same sensor configuration.

Hence, we convert the pooling function into a learnable weighting function, shown in Equation 2, wherein we add a learnable embedding $E_k \in R^{C \times X \times Y}$ per sensor while incorporating the sensor's intrinsic parameters $I_k \in R^{3\times3}$. The BEV features from individual camera sensors are also scaled with their mean μ_k. We found that using a learnable embedding in pooling the BEV features, enables the network to focus on the right camera feature in overlapping regions on BEV grid, thereby improving our overall results.

$$F_{bev}^{total} = \sum_{k=1}^{N} I_k \cdot (F_{bev}^{k} - \mu_k) + E_k \tag{2}$$

3. Occlusion reasoning in BEV Segmentation: BEV Segmentation task is designed to represent image information in BEV space, focusing solely on grid cells perceivable by vehicle cameras. Occluded areas should not contribute to predictions to avoid speculative inference. This occlusion reasoning is particularly crucial for near-field sensing in urban driving and parking scenarios, where occlusion is prevalent. To address this, we propose a novel multi-task approach, which includes *Occlusion reasoning*, wherein the model outputs an occlusion probability for each grid cell, a feature included in our dataset. In synthetic engines, the BEV camera records the class of the closest object, whereas the desired semantic BEV labels must contain the class relevant for the ego vehicle to navigate the world. This contrast leads to the misrepresentation of BEV semantic classes when trees or bridges obstruct the BEV camera. On the contrary, there will also be label mismatches when objects are occluded in the ego vehicle's camera sensors but not in the BEV camera.

To generate the occlusion map $p(o)$, a circular kernel is applied to the camera BEV features to count local pixel occurrences within grid cells. The occurrences are normalized with a threshold (τ) to derive an occlusion score $p(o)$. When vehicles are visible in multiple cameras, the entire vehicle is designated as occupied and non-occluded in the BEV grid. This allows the network to comprehend vehicle dimensions, including occlusion effects. Figure 2 visualizes occluded areas as black regions. In practice, the occupancy probability ($p^{'}(o) = 1 - p(o)$) is used for ease of loss assignment and visualization.

4. Fisheye BEV Segmentation: We incorporate a multi-task head that makes classification and occupancy predictions for each cell to circumvent fisheye camera optics. However, we keep the model architecture of heads unchanged from state-of-the-art BEV segmentation techniques [12]. As illustrated in Equation 3, we use weighted cross-entropy loss, based on [13], for training the classification head. The occlusion head, depicted in Equation 4, is trained using binary cross-entropy loss.

The classification loss is a function of occupancy probability ($p(o)$). The predicted ($p(c_i)$) and ground truth ($q(c_i)$) class-probabilities are weighted by α_i. The next section, un-

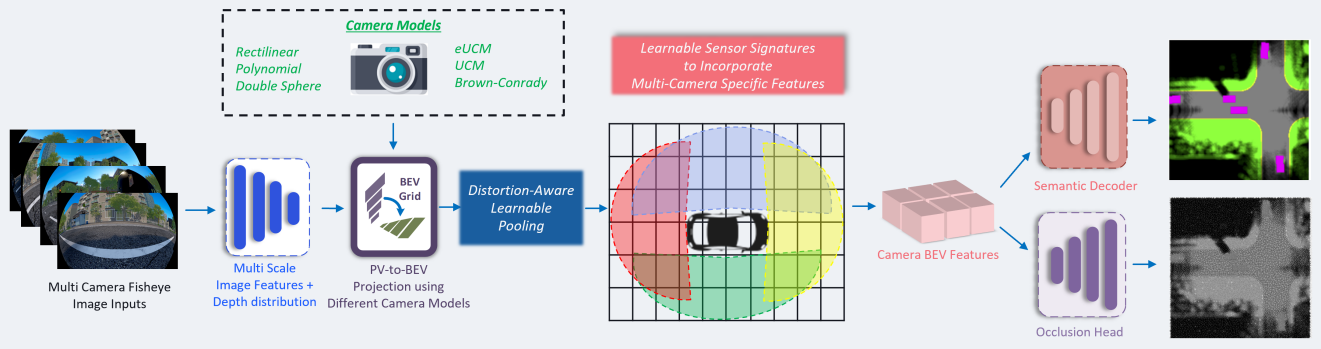

Figure 2. **Block diagram of the proposed FisheyeBEVSeg algorithm**, with distortion-aware BEV grid generation and multi-task semantic and occlusion segmentation.

der ablation studies, explains the chosen weighting strategy for training.

$$\mathcal{L}_{semantic}(p,q) = -p(o) \sum_{c_i \in C} \alpha_i \cdot p(c_i) \cdot \log q(c_i) \quad (3)$$

$$\mathcal{L}_{occ}(p(o), p(\hat{o})) = -[p(o) \cdot \log p(\hat{o}) + (1 - p(o) \cdot \log 1 - p(\hat{o})] \quad (4)$$

Where $p(o)$ is the occupancy probability predicted by the occupancy head and $p(\hat{o}) \in \{0, 1\}$ is the ground truth occupancy label. The combination of semantic classification loss ($\mathcal{L}_{semantic}$), and occupancy loss ($\mathcal{L}_{occ}$), presented in Equation 5, is used to train the BEV Segmentation model end-to-end. When combined with semantic loss, we introduce a weighting parameter (λ) to occupancy oss. This focuses the training on the BEV Segmentation task and helps in training stability and convergence.

$$\mathcal{L}_{total} = \mathcal{L}_{semantic} + \lambda \cdot \mathcal{L}_{occ} \quad (5)$$

3. Implementation

Datasets: For fisheye BEV segmentation, there are no real-world datasets, but there are two possible synthetic datasets, namely SynWoodScape[14] and F2BEV (FBSSEM dataset) [15]. These datasets do not address the occlusion problem described in the previous Occlusion reasoning section. Besides this issue, F2BEV is confined to simulating just parking lots and is not suitable for general driving. SynWoodScape has released only a small initial sample of the dataset. Therefore, we generated a synthetic dataset using the Cognata simulation platform. To overcome the occlusion problem, we enrich the information from the BEV camera using additional sensor information from other vehicle cameras. We project the perspective semantic labels from other camera sensors in the vehicle to 3D space to rectify the BEV camera semantic labels. Our final dataset comprises four fisheye cameras and six pinhole cameras with 1920×1208 resolution, BEV ground truth images (400×400) with five semantic classes: *invalid, vehicles, lane markings, street, and background.*

In addition, we generate an occlusion probability map indicating the likelihood of occlusion in each grid cell. The training dataset comprises five virtual scenes of about 90 seconds each from Pittsburgh, Munich, and HaSharon. Each scene is simulated with different traffic, weather, and daylight conditions. The whole dataset contains more than 12,000 different frames corresponding to 50,000 fisheye images.

Architecture Details: As described in Section 2, we select the architectural approach from Lift-Splat-Shoot [9] to implement the BEV segmentation using fisheye cameras. Given the near-field applications of fisheye cameras, we choose a BEV grid cell size of $0.25m$ instead of $0.5m$. Further, we set a coverage range of $25m$ around the ego vehicle. The input resolution to the network is 480×302. Additionally, while training, we incorporate color jittering and other commonly used data augmentation techniques such as scaling, rotation, and cropping.

4. Experiments and Results

We perform several training runs to evaluate implementation details and improve the results. We use a GPU server with four Nvidia Quadro RTX5000 GPUs with a batch size 64. We use Adam optimizer with a learning rate of 10^{-4}. All implementation is done in PyTorch.

We generate 3 sets of test scenes: *easy, medium, and hard*. The easy scene replicates the traffic and weather scenarios of the training set. The medium and hard sets are new highways and complex road geometry urban scenes. We use Intersection over Union (IoU) as the primary metric for semantic classification. It is computed with predicted semantic class (y) and ground truth semantic class ($\hat{y}$). We track and report the IoU for each class separately to account for the skew in classes (background vs. pedestrians/vehicles), along with mIoU, which is the mean of IoU of all the classes. We also use IoU to predict occupancy by converting it to binary masks.

Table 1 shows the results of training and test datasets. Comparing the three test sets shows the performance drop with the increasing difficulty level. The medium test set scores are significantly lower than the easy test set, espe-

Table 1. **Quantitative Performance of FisheyeBEVSeg** on different test dataset slices. Input image size is 480×302.

Method	*Image Type*	*Dataset*	*IoU*					
			mIoU	*Occlusions*	*Vehicles*	*Markings*	*Street*	*Background*
LSS [9]	Cyl. Rectified	Easy	0.659	-	0.607	0.379	0.773	0.875
		Medium	0.540	-	0.539	0.214	0.645	0.763
		Hard	0.314	-	0.317	0.114	0.523	0.303
FisheyeBEVSeg	Raw Distorted	Easy	0.796	0.815	0.776	0.517	0.895	0.978
		Medium	0.690	0.682	0.764	0.364	0.858	0.782
		Hard	0.466	0.666	0.464	0.176	0.572	0.449

cially for the lane marking and the background. However, the network learns to position the vehicles quite well. We also compared against standard baseline [9] by applying a cylindrical projection to rectify the fisheye images. Cylindrical projection preserves the vertical lines from fisheye images. The results in Table 1 indicate that our model performs better than the rectification process. It is an intuitive outcome, given that rectification introduces unnecessary object stretching and interpolation artifacts to the image in addition to losing some of the field of view. We provide qualitative results of FisheyeBEVSeg on the test data at https://youtu.be/HfTPwMabgS0. It can be observed that the vehicles and lane markings are detected accurately. The network has problems predicting the road precisely, especially in regions that have more than 10 m away.

Ablation Studies: During our experiments, we used a class weighting of $(13\times, 3\times, 1\times, 1\times)$ for vehicles, lane markings, streets, and backgrounds, respectively. However, from Table 2, the model with uniformly weighted loss performs better for almost all classes. Our explanation for counter-intuitive behavior is that, unlike the image classification tasks, for semantic BEV segmentation, the presence and detectability of class are more important than just class occurrence. Given the geometry-based approach, the network cannot estimate occluded areas. We do not assign a loss to occluded areas so the network does not hallucinate. Loss removal for occluded regions avoids the network updating its weights based on these occluded regions. Table 2 illustrates the impact of occupancy loss on accuracy.

5. Conclusion

In this work, we introduce a novel pipeline to perform semantic Bird's Eye View (BEV) segmentation using images from fisheye cameras. We perform our experiments on a synthetic dataset due to the lack of public datasets with fisheye cameras. Future directions for our work include bridging the reality gap between synthetic and real-world data and extending to additional tasks with fisheye cameras. We hope that this work encourages further research in performing BEV perception for fisheye cameras.

Table 2. **Ablation Study of FisheyeBEVSeg on *Medium* test dataset**. Under *Class Weight* ablation, ***X-X-X-X*** represents weights for Vehicles, Lane Markings, Street and Background .

Approaches	*IoU*					
	mIoU	*Occlusion*	*Vehicles*	*Markings*	*Street*	*Background*
Class Weights						
13-3-1-1	0.658	**0.687**	0.716	0.346	0.803	0.736
1-1-1-1	**0.690**	0.682	**0.764**	**0.364**	**0.858**	**0.782**
Loss for Occluded areas						
✓	**0.658**	**0.687**	0.716	**0.346**	**0.803**	**0.736**
✗	0.631	0.675	**0.724**	0.342	0.752	0.662
BEV Pooling methods						
Sum	0.666	0.665	0.719	0.308	**0.862**	0.776
Maxpool	0.656	0.602	0.748	0.309	0.857	0.764
FisheyeBEVSeg (Cam Intrin + Emb)	**0.690**	**0.682**	**0.764**	**0.364**	0.858	**0.782**

References

[1] J. Huang, G. Huang, Z. Zhu, Y. Yun *et al.*, "Bevdet: High-performance multi-camera 3d object detection in bird-eye-view," *arXiv preprint arXiv:2112.11790*, 2021. 1

[2] S. Mohapatra, S. Yogamani, H. Gotzig *et al.*, "Bevdetnet: bird's eye view lidar point cloud based real-time 3d object detection for autonomous driving," *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. 1

[3] V. R. Kumar, M. Klingner *et al.*, "Svdistnet: Self-supervised near-field distance estimation on surround view fisheye cameras," *IEEE Transactions on ITS*, 2021. 1

[4] C. Eising, J. Horgan *et al.*, "Near-field perception for low-speed vehicle automation using surround-view fisheye cameras," *IEEE Transactions on ITS*, 2021.

[5] G. Sistu, I. Leang *et al.*, "Neurall: Towards a unified visual perception model for automated driving," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*.

[6] N. Tripathi and S. Yogamani, "Trained trajectory based automated parking system using Visual SLAM," *Proc. of the Computer Vision and Pattern Recognition Workshops*, 2021.

[7] H. Rashed, E. Mohamed *et al.*, "Fisheyeyolo: Object detection on fisheye cameras for autonomous driving," in *Proc. of the ML for Autonomous Driving NeurIPS 2020 Workshop*.

[8] H. Rashed, S. Yogamani, A. El-Sallab *et al.*, "Optical flow augmented semantic segmentation networks for automated driving," in *Proceedings of VISAPP*, 2019. 1

[9] J. Philion and S. Fidler, "Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d," *European Conference on Computer Vision*, 2020. 1, 2, 3, 4

[10] J. Kannala and S. Brandt, "A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses," *IEEE Trans. on PAMI*, vol. 28, no. 8, 2006. 2

[11] V. R. Kumar, C. Eising *et al.*, "Surround-view fisheye camera perception for automated driving: Overview, survey & challenges," *IEEE Transactions on ITS*, 2023. 2

[12] Z. Liu, H. Tang *et al.*, "Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation," *2023 IEEE Intl. Conf. on Robotics and Automation (ICRA)*. 2

[13] Z. Zhang and M. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," *Advances in neural information processing systems*, 2018. 2

[14] A. R. Sekkat, Y. Dupuis *et al.*, "Synwoodscape: Synthetic surround-view fisheye camera dataset for autonomous driving," *IEEE Robotics and Automation Letters*, 2022. 3

[15] E. U. Samani, F. Tao, H. R. Dasari *et al.*, "F2bev: Bird's eye view generation from surround-view fisheye camera images for automated driving," *IEEE/RSJ IROS*, 2023. 3