

Boosting Real-Time Driving Scene Parsing with Shared Semantics

Zhenzhen Xiang¹, Anbo Bao¹, Jie Li² and Jianbo Su¹

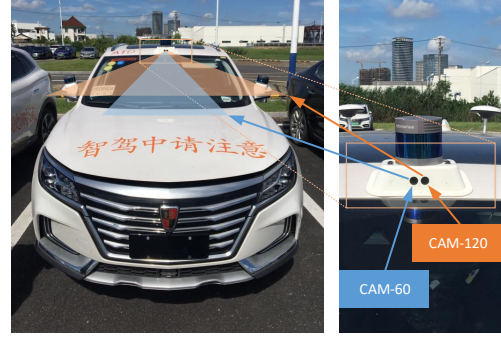
Abstract—Real-time scene parsing is a fundamental feature for autonomous driving vehicles with multiple cameras. In this letter we demonstrate that sharing semantics between cameras with different perspectives and overlapped views can boost the parsing performance when compared with traditional methods, which individually process the frames from each camera. Our framework is based on a deep neural network for semantic segmentation but with two kinds of additional modules for sharing and fusing semantics. On the one hand, a semantics sharing module is designed to establish the pixel-wise mapping between the input images. Features as well as semantics are shared by the map to reduce duplicated workload which leads to more efficient computation. On the other hand, feature fusion modules are designed to combine different modal of semantic features, which leverage the information from both inputs for better accuracy. To evaluate the effectiveness of the proposed framework, we have applied our network to a dual-camera vision system for driving scene parsing. Experimental results show that our network outperforms the baseline method on the parsing accuracy with comparable computations.

I. INTRODUCTION

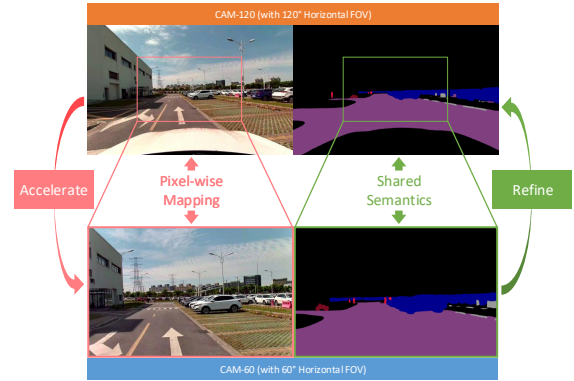
With the development of autonomous driving in recent years, scene parsing as a critical functionality of autonomous vehicles, has attracted more and more attention [1]. Since scene parsing is a dense classification problem, it still remains a difficult task to achieve an accurate performance for real-time applications, especially for vehicles with multiple cameras and limited computational resources.

Taking our autonomous vehicle platform shown in Fig. 1(a) as an example, a dual-camera vision system is mounted at the top of the vehicle, which is a common vision system setup adopted by modern autonomous vehicles, e.g., Tesla Autopilot [2]. Compared with typical stereo cameras, these two cameras are with different field of views (FoVs), which is designed for reliable and accurate perception of objects at various distances. In the figure CAM-60 refers to the camera with a 60° horizontal FoV (HFOV) and CAM-120 stands for the camera with 120° HFOV. To get scene parsing results from both cameras, traditional approaches usually process images from each camera individually, which neglects the connection inside the dual-camera system.

Since the cameras with different perspectives have overlapped perception regions as shown in Fig. 1(b), we consider to find a method to (1) build a pixel-wise mapping to share



(a) Configuration of our dual-camera vision system



(b) Motivation of shared semantics

Fig. 1. Illustration of our dual-camera system and motivation. Shared semantics based on the pixel-wise mapping accelerates the processing of CAM-60 and refines the results of CAM-120.

semantics between two cameras and (2) leverage the compensation of different perspectives to accelerate computation and get refined scene parsing results. More specifically, because the scenes captured by CAM-60 are almost completely contained in the image from CAM-120, processing images from CAM-60 can benefit from the information propagated from CAM-120, which leads to a more efficient computation. At the same time, because CAM-60 has a larger focal length, it has a clearer perception to the scenes far away from the vehicle. Thus CAM-120 can fuse such information to enhance its original segmentation results.

In general, when compared with traditional approaches, our method mainly boosts the scene parsing task for multi-camera systems like our configurations in the following two aspects:

- Reduce the computation load for cameras with narrower FoVs. Feature extraction procedure only needs to be

This work was partially financially supported by the projects of National Natural Science Foundation of China under grant 61533012 and 91748120.

¹Zhenzhen Xiang, Anbo Bao and Jianbo Su are with the Department of Automation, Shanghai Jiao Tong University, Shanghai 200240, China. Emails: zzxang.sjtu@gmail.com, {ab.bao, jbsu}@sjtu.edu.cn

²Jie Li is with SAIC Motor Corporation Limited, Shanghai 201804, China. Email: lijie06@saicmotor.com

done once in the overlapped regions among different cameras. For example, the heavy and slow feature extraction backbone for CAM-60 can be replaced with a lightweight one for extracting complementary features. The semantic information propagated from CAM-120 by a semantic sharing module provides a coarse segmentation for CAM-60, which can be further refined by fusing with its own features.

- Improve the scene parsing quality for cameras with broader FoVs. For example, the semantic features from CAM-60 are also back-propagated to CAM-120 with the same semantic sharing module. By appropriately fusing with the original semantics of CAM-120, those semantics located in the overlapped regions can be further enhanced with the perspective advantage from CAM-60.

II. RELATED WORK

A. Real-Time Scene Parsing

Deep learning based scene parsing has been extensively investigated in recent years, e.g., FCNs [3], SegNet [4]. Although the state-of-the-art semantic segmentation networks can output high-quality results [5], [6], they are too heavy and computationally expensive to be adopted in real-time applications. Recently some lightweight semantic segmentation networks has been designed to work on-line while giving decent outputs [7]–[9]. However, these networks are not naturally designed for those vision systems with multiple cameras, which makes them still too memory or computationally consuming for autonomous driving applications. In our work, we aim to design an optimized architecture to reduce the redundant computation which leads to a more efficient framework.

B. Semantics Sharing

Semantics sharing seeks to find correspondences between different images which have overlapped views, e.g., the image pairs from stereo cameras or the consecutive frames in a video sequence. Semantics sharing is commonly conducted in two levels: pixel-level and feature-level.

1) *Pixel-level Sharing*: For pixel-level sharing, a pixel-wise grid map is built to warp an image from one perspective to the other. The map can be derived from the transformations in geometry space or image space. The transformation in geometry space generally uses the prior knowledge, e.g., the planar assumption for perspective transformation [10], or the depth estimation of the scene [11], [12]. The transformation in image space usually considers the correlations around neighborhoods of a pixel [13]. With recent development in lightweight optical flow estimation networks [14], [15], it is much more practical to exploit an optical estimation network in real-time applications. Xu et al. [16] applied different segmentation strategies to various regions of the input image, which exploited optical flow to preserve the semantics in static regions. Zhu et al. [17] investigated the generation of future semantic segmentation labels from current manual labels by video prediction based on motion vector estimation.

Yin et al. [10] combined a rigid structure reconstructor and a non-rigid motion localizer to warp from one view to the other.

Compared with pure geometry-based methods, the image-based methods are more robust to the errors of camera calibration and time synchronization among different cameras. Therefore, similar to [10], our framework also integrates both geometry-based methods and image-based methods for sharing semantic information between two cameras at the pixel-level.

2) *Feature-level Sharing*: Feature-level sharing propagates information implicitly in the model, which is usually applied in video sequence processing. Jin et al. [18] designed a network to learn predictive features in video scene parsing. Li et al. [19] proposed a framework with adaptive feature propagation for high-level features to reduce the latency of video semantic segmentation. Wang et al. [20] used an unsupervised method to learn feature representations for identifying correspondences across frames. Lee et al. [21] attempted to derive semantic correspondences by object-aware losses. Compared with pixel-level sharing, feature-level sharing is learned by an end-to-end process, and it is thus difficult to directly evaluate its performance. In addition, the feature-level sharing may rely on the training data more heavily than the pixel-level sharing used in our framework.

C. Semantics Fusion

The idea of semantics fusion for improving the segmentation outputs has been widely applied in previous works. For example, in [7] and [8], different modals or levels of features were fused with each other to generate refined results. Li et al. [22] hypothesized a scaled region from the original image by a perspective estimation network, which aimed to refine original segmentation results of small objects. Jiao et al. [23] proposed to improve and distill the semantic features with the estimated depth embeddings by geometry-aware propagation. All of the works above focus on the fusion for a single image, while Hoyer et al. [24] demonstrated a spatial-temporal fusion method for multiple camera sequences but with non-overlapped view. In our work, we have followed the basic idea of semantics fusion and applied it to cameras with different perspectives and shared visions to enhance the overall scene parsing performance.

III. METHODOLOGY

In this section, we will describe the proposed method in detail. First the overview of our framework will be demonstrated. Then the ideas behind the design of each core module will be discussed. The detailed implementation information will be given at the end of the section.

A. Framework Overview

The proposed framework is illustrated in Fig. 2. The final goal is to output the scene parsing results for each input image from both CAM-120 and CAM-60.

From the view of structure, our framework can be divided into two branches. Unlike traditional designs with exactly the

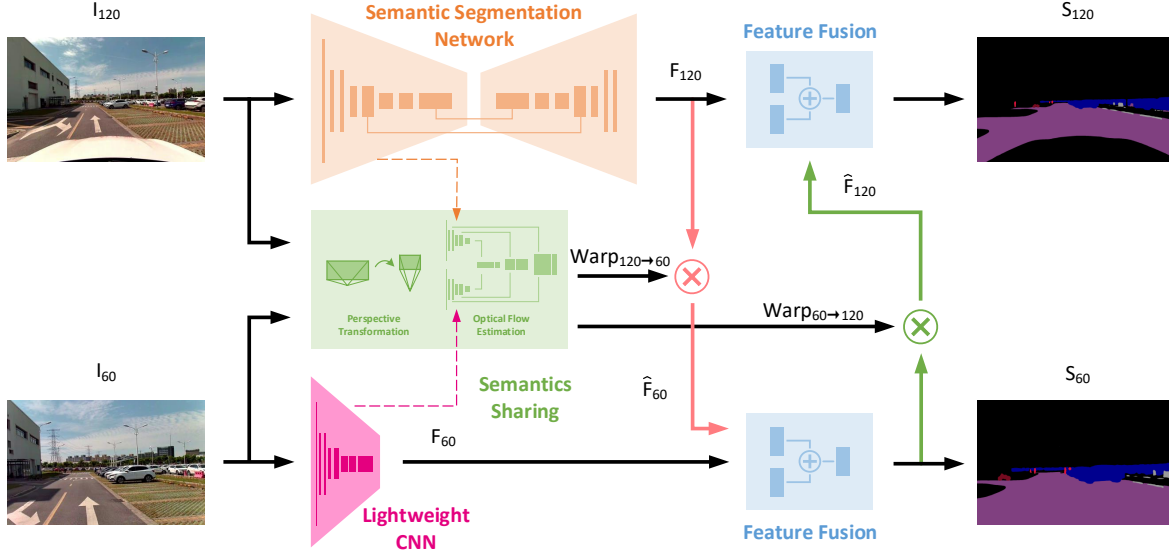


Fig. 2. The proposed scene parsing framework with shared semantics for our dual-camera vision system. The semantic segmentation network can be any real-time encoder-decoder type segmentation models. The lightweight CNN extracts features fast for CAM-60. The semantics sharing module consists of perspective transformation and optical flow estimation, which provides maps for warping between CAM-120 and CAM-60. Note that the dashed orange and magenta arrows means the optical flow estimation reused the features from the semantic segmentation network and the lightweight CNN. Please refer to Sec. III-D.1 for more details. The feature fusion modules merge the shared semantics for final results.

same pipeline for both branches, the input image from CAM-60 passes a much more lightweight convolutional neural network (CNN) compared with a complete semantic segmentation network in the branch of CAM-120. The sharing and fusion of information between two branches are realized with a semantics sharing module and two feature fusion modules, respectively.

From the view of functionalities, four kinds of modules in our framework play different roles. The semantic segmentation network provides high-level semantic features for sharing. The lightweight CNN recovers the detailed and complementary features for refined parsing results, similar to the architecture of ICNet [8]. The semantics sharing module establishes a bridge for bi-directional feature propagation from CAM-120 to CAM-60 and vice versa. The feature fusion modules merge shared semantics for each branch to achieve better parsing results.

Since the semantic segmentation network is a full-function network which can output scene parsing results by itself, it can be easily replaced with any modern networks designed for real-time scene parsing. For the lightweight CNN, it can also be designed as a sequential of several convolutional layers or sharing the structure with the feature extraction backbone in the semantic segmentation network. The implementation details of these two parts will be described in Sec. III-D. In the following we will focus on the details of the semantics sharing module and the feature fusion module.

B. Semantics Sharing Module

The task of the semantics sharing module is to remap the semantic features between two branches. Through such a bridge, the semantic features from branch CAM-120 can be propagated to branch CAM-60 to speed-up its processing,

and then the results of CAM-60 are transferred back to refine the outputs of CAM-120, which forms a closed loop.

Although the pure geometry-based method (e.g., the depth estimation based warping) and feature-level propagation can also be used for sharing semantics, as concluded in Sec. II-B, the results of pixel-level sharing methods are more robust and explicit, and thus we have proposed a two-stage image warping method to build the semantics sharing module as shown in Fig. 3.

1) *Stage I: Geometry-based Warping*: In the first stage, the input image from CAM-120 is warped by the perspective transformation. The homography matrix used in the transformation can be derived from the intrinsic and extrinsic parameters of the dual-camera system [25]:

$$H_{120 \rightarrow 60} = K_{60} R K_{120}^{-1}, \quad (1)$$

where $H_{120 \rightarrow 60}$ is the homography matrix for mapping from CAM-120 to CAM-60, K_{60} and K_{120} are their camera matrices. R is the rotation matrix from CAM-120 to CAM-60. Due to the limitation of perspective transformation, those objects closed to the camera will be distorted after the transformation. Thus the warped image from CAM-120 still needs to be adjusted to accurately match the ground truth image from CAM-60.

2) *Stage II: Image-based Warping*: In the second stage, the warped image from CAM-120 is further warped by the optical flow to compensate the distortion effects. The core process of this stage is the precise estimation of optical flow between the input image pairs. It should be noticed that because the pose variation between two cameras is very small and the input image pairs are correctly synchronized, the scene can be considered as static and the occlusion effect is negligible. Therefore the movement of pixels is not that large

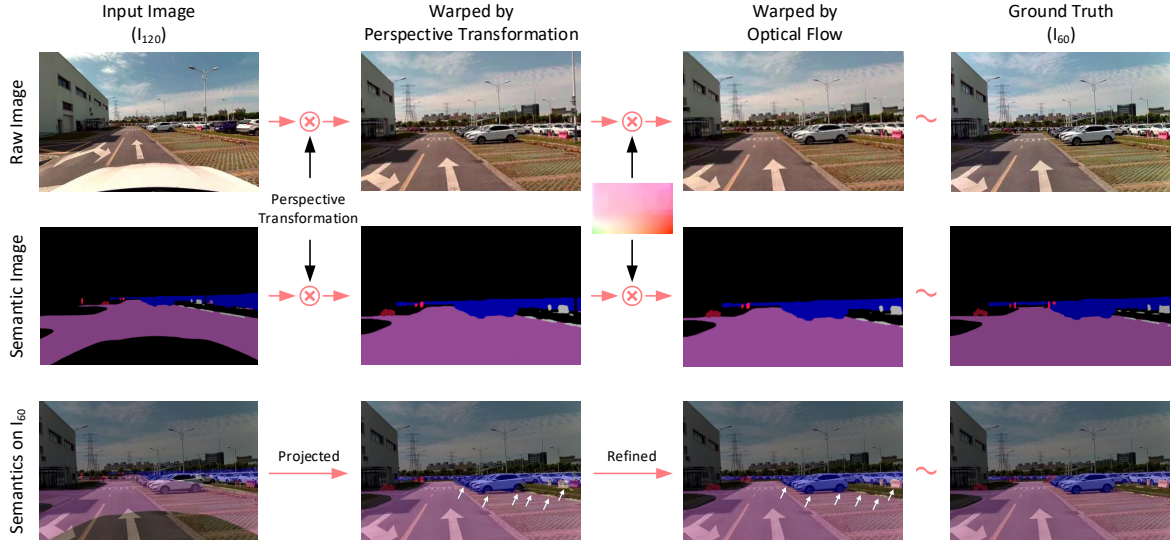


Fig. 3. The pipeline of semantics sharing. The first and second row show the warping progress for the raw and semantic image of CAM-120, respectively. The third row demonstrates the propagated semantics from CAM-120 to CAM-60, where the white arrows point out the improvements after refinement.

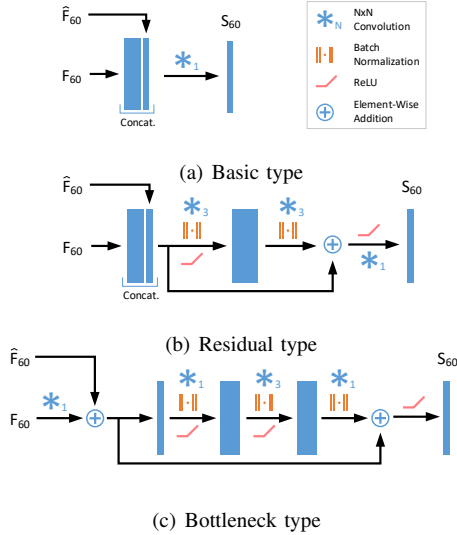


Fig. 4. Three optional types of feature fusion modules in our framework.

and the artifacts of the warped image also can be ignored comparing with the situation of video scene parsing.

C. Feature Fusion Module

The feature fusion modules are used to generate the segmentation results of CAM-60 as well as to refine the results of CAM-120. As shown in Fig. 4, we have implemented and evaluated three different types of the feature fusion modules to compare with the direct output of the semantic segmentation network. In Sec. IV-D.4 we will show the ablation analysis of these blocks which depicts that even integrating the simple basic block can boost the parsing outputs to some extent.

In the following we will take the feature fusion module in the CAM-60 branch as an example to describe their structures.

1) *Basic Type*: The basic type of feature fusion module only concatenates the input feature maps and output the semantic feature maps after an 1×1 conv.

2) *Residual Type*: Since the effectiveness of residual block has been widely proved in previous works, we also apply it to our framework. The inputs are first concatenated and then passed through a standard residual block with 3×3 conv layers. Finally the output is processed by an 1×1 conv for classification.

3) *Bottleneck Type*: Considering to decrease the computation and the amount of parameters in our framework, we also evaluate a bottleneck type of feature fusion module. The inputs are first converted to the same channels with an 1×1 conv, then they are passed through a bottleneck with an expansion, followed by an rectified linear unit (ReLU) for the output.

D. Implementation Details

1) *Structure*: Taking the implementation of the semantics sharing module into account, we have developed two types of structures for our framework: a) loosely-coupled structure and b) tightly-coupled structure. For the loosely-coupled structure, we simply exploit a complete optical flow network following the perspective transformation, which can achieve the best estimation performance.

However, because the optical flow network also has its own feature extraction modules, it is possibly duplicable to those in the semantic segmentation network. Therefore, in the tightly-coupled structure shown in Fig. 5, we remove the feature extraction part of the optical flow network and reused the feature maps from the semantic segmentation network. With such adjustments made, the whole model becomes more compact and the computation load can be further cut down.

2) *Semantic Segmentation Network*: We exploit a real-time semantic segmentation network based on MobileNetV3-large [9] to get the initial semantic features of CAM-120.

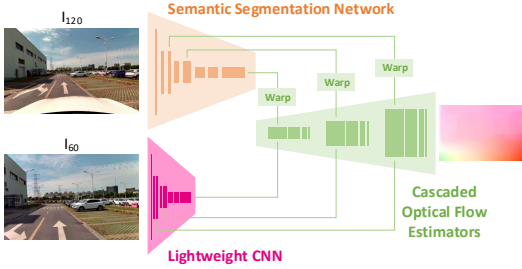


Fig. 5. Tightly-coupled structure of optical flow estimation network.

To train the semantic segmentation network, we apply the common cross entropy loss to supervise the training progress.

3) *Lightweight CNN*: The implementation of the lightweight CNN is based on the structure of the framework. For the loosely-coupled structure, we share the structure with the semantic segmentation network and output a feature pyramid with 1/8 size of the original resolution for later fusion with the results from CAM-120. For the tightly-coupled structure, we reuse the feature extraction part of the optical flow network and adjust it to output a feature pyramid with exactly the same size from 1/4, 1/8 to 1/16 as those from the semantic segmentation network.

The lightweight CNN also shares the weights with the semantic segmentation network in the loosely-coupled structure. In the tightly-coupled structure, it is trained together with the optical flow network.

4) *Semantics Sharing Module*: The main part of the semantics sharing module is an optical flow estimation network. We use a PWC-Net [15] to provide grid maps for warping feature maps. It should be noticed that the original feature pyramid given by PWC-Net is not the same as the MobileNetV3-large. Thus for the tightly-coupled structure, we need to modify the channels of the output feature maps to match those in MobileNetV3-large accordingly.

The training losses of the optical flow network in our cases consist of three different types: a) supervised loss, b) unsupervised loss and c) semantic loss. The supervised loss is applied when the ground-truth flow is available with some synthetic datasets. It is defined as the average end-point error (AEPE):

$$\mathcal{L}_{sup} = \frac{1}{N} \sum_i^N \|\mathbf{w}(p_i) - \hat{\mathbf{w}}(p_i)\|_2, \quad (2)$$

where p is the pixel index and N is the total number of pixels in the flow image. \mathbf{w} and $\hat{\mathbf{w}}$ are the ground-truth and the estimated flow, respectively.

The unsupervised loss is mainly for training on those datasets without the ground-truth flow. We choose three most commonly used losses for unsupervised learning:

$$\mathcal{L}_{unsup} = w_1 \mathcal{L}_1 + w_2 \mathcal{L}_{SSIM} + w_3 \mathcal{L}_{smooth}. \quad (3)$$

Here the first term is defined as the L_1 norm of the pixel intensity difference between the ground-truth image \mathbf{I} and



Fig. 6. Driving route for the video data collection of the dataset.

TABLE I
STATISTICS OF OUR DATASET

Length (km)	Duration (h)	Max. Speed (km/h)	Image Pairs	Image Size	Semantics Classes
~10	1.28	30	4593	1920×1208	6

the flow-warped image $\hat{\mathbf{I}}$:

$$\mathcal{L}_1 = \frac{1}{N} \sum_i^N \|\mathbf{I}(p_i) - \hat{\mathbf{I}}(p_i)\|_1. \quad (4)$$

The second term is the SSIM [26] loss of the ground-truth image and the flow-warped image. The third term is the smoothness loss [27] of the estimated flow. The weights of these three losses w_1 , w_2 and w_3 are set to 0.1, 1.0 and 1.0, respectively.

The semantic loss is for the dataset with semantic labels. It can be regarded as a supervision for flow at the boundaries of each semantic class. Here we also applied the cross entropy loss to supervise the fine-tuning of the optical flow network.

5) *Feature Fusion Module*: For the basic and residual type of feature fusion modules, they are applied to both branches without modification. However, since the bottleneck type has an element-wise addition unit, we will additionally need an 1×1 conv to reshape F_{60} to the same size as \hat{F}_{60} in the CAM-60 branch, as shown in Fig. 4(c).

IV. EXPERIMENTS

A. Dataset and Evaluation Metrics

Since we have not found any public dataset with configurations as our applications, we built our own dataset with a dual-camera system on an autonomous vehicle. The videos were captured by a Sekonix SF3324 (CAM-120) and a Sekonix SF3325 (CAM-60) with an NVIDIA DRIVE AGX platform. The video sequences were collected inside the SAIC Motor Park and the driving route is shown in Fig. 6.

The statistics of the dataset is listed in Table I. The videos from each camera are synchronized by the hardware. We automatically extract images from the videos at the rate of one frame per second. Then we select about 1000 image pairs to manually label six classes of semantics: *background (BG)*, *road*, *person*, *car*, *barrier* and *cycle*. We use these images to train a PSPNet [5] to automatically label the other images as the ground truth for later training the semantic segmentation network based on MobileNetV3-large for real-time parsing.

In addition to the semantic labels, we also need ground-truth optical flow to train the PWC-Net, which is difficult

to obtain. So we turned to synthesize a warped image by a random perspective transformation from an input image. Specifically, the focal length of CAM-120 in its camera matrix is multiplied by a random factor between 0.95 and 1.05; then the image is further randomly translated by ± 10 pixels and rotated by $\pm 5^\circ$. The flows generated along with the transforming process is used as the ground truth. In Sec. IV-D.2 we will show that after training on this dataset, the performance of PWC-Net on the original dataset will be improved.

The performance of our network is evaluated with the mean intersection over union (mIoU) metric for semantic segmentation and average end-point error (AEPE) for optical flow estimation.

B. Training Procedure

We follow a multi-stage training procedure to train each component of our framework:

- Semantic segmentation network: We trained the MobileNetV3-large with a segmentation head for 160K iterations using a mini-batch size of 16. The initial learning rate was set to 0.015 and followed a ‘poly’ policy with power 0.9.
- Optical flow estimation network: For the loosely-coupled structure, the PWC-Net was trained separately to the segmentation network. It was first trained on the *FlyingChairs* dataset with the same settings as [15]. Then we further trained it on our synthetic flow dataset for 300K iterations using a mini-batch size of 8. The initial learning rate was 0.0005 and was scaled by 0.5 at 100K, 200K, 250K. Finally the model was fine-tuned on the real data with unsupervised losses and semantic loss sequentially. For the tightly-coupled structure, only one of the feature extraction parts and the optical flow estimation part were needed to be trained. The training settings remained the same as the loosely-coupled structure.
- Feature fusion modules: The feature fusion modules were trained with the whole network with the fixed weights of MobileNetV3-large and PWC-Net. The feature fusion module in CAM-60 branch was first trained for 60K iterations with a mini-batch size of 4. The initial learning rate was set to 0.001. The other training settings were the same as MobileNetV3-large. The feature fusion module in CAM-120 was also trained in the same way.
- Fine-tuning: The whole network was finally fine-tuned together for 120K iterations with the same settings as training MobileNetV3-large. In order to keep a steady performance of optical flow estimation, the weights of PWC-Net were fixed in the final fine-tuning.

We use PyTorch to implement our network. The network is trained and tested on two NVIDIA Tesla V100 GPUs. Our code and trained models have been made publicly available at: <https://github.com/zhenzhenxiang/SemanticsSharing>.

C. Main Results

We have chosen the MobileNetV3-large based segmentation network [9] as our baseline, which is also used in the CAM-120 branch of our framework. As shown in Table II, we have compared the semantic segmentation performance, model statistics and runtime of the baseline and our network with loosely-coupled and tightly-coupled structures.

1) *Semantic Segmentation*: For the CAM-60 branch, the segmentation results show that our loosely-coupled structure has slightly over-performed than the baseline in general, although there is only a lightweight CNN in this branch. Besides, the performance of our tightly-coupled structure is also very close to the baseline with more reused intermediate features.

For the CAM-120 branch, both of our loosely-coupled and tightly-coupled structure have an obvious improvement comparing with the baseline, especially for the loosely-coupled with the class of *Person* (+2.3%), *Barrier* (+1.7%) and *Cycle* (+5.9%). In addition, the mIoU results in the central view of CAM-120 which overlaps with CAM-60 (c.f. “120-OL”) are also provided for further comparisons. The results show significant improvements on *Person* (+3.7%), *Barrier* (+4.6%) and *Cycle* (+5.6%). This reflects the effectiveness of our semantics sharing module and feature fusion module which propagate and fuse the semantic information from CAM-60 to CAM-120. The sharing of such information compensates and improves the features of those small objects in the view of CAM-120. As shown in Fig. 7, our network successfully recovers the missing small objects that are far from the vehicle (c.f. the first and second group of image pairs), and has a more accurate classification at the boundary of small objects (c.f. the third group of image pairs).

2) *Model*: Our loosely-coupled model has $2.3\times$ more parameters than the baseline, which can be reduced to $1.6\times$ with tightly-coupled structure. The computation is evaluated by input images with 1920×1208 resolution for the MobileNetV3-large and 768×483 for PWC-Net. The results show that our loosely-coupled model has comparable computation with the baseline, while the tightly-coupled model needs even less computation resources.

3) *Runtime*: The results are the average runtime of 300 inferences for each model. As shown in Table II, the baseline and our models are all capable for real-time applications, especially when the size of input image is a half of its original resolution, i.e., 960×604 . When deployed to embedded devices, the models can be further optimized which will lead to a much higher frame rate.

D. Ablation Study

1) *Loosely-Coupled vs. Tightly-Coupled*: The influence of loosely-coupled and tightly-coupled structure for optical flow estimation was evaluated. The results on *Data.Sim* (the hypothesized image pairs) and *Data.Real* (the real image pairs) are listed in Table III. We can find that the tightly-coupled structure has larger AEPE and unsupervised loss in both datasets. This is mainly because that we have fixed the weights of reused feature extraction part from

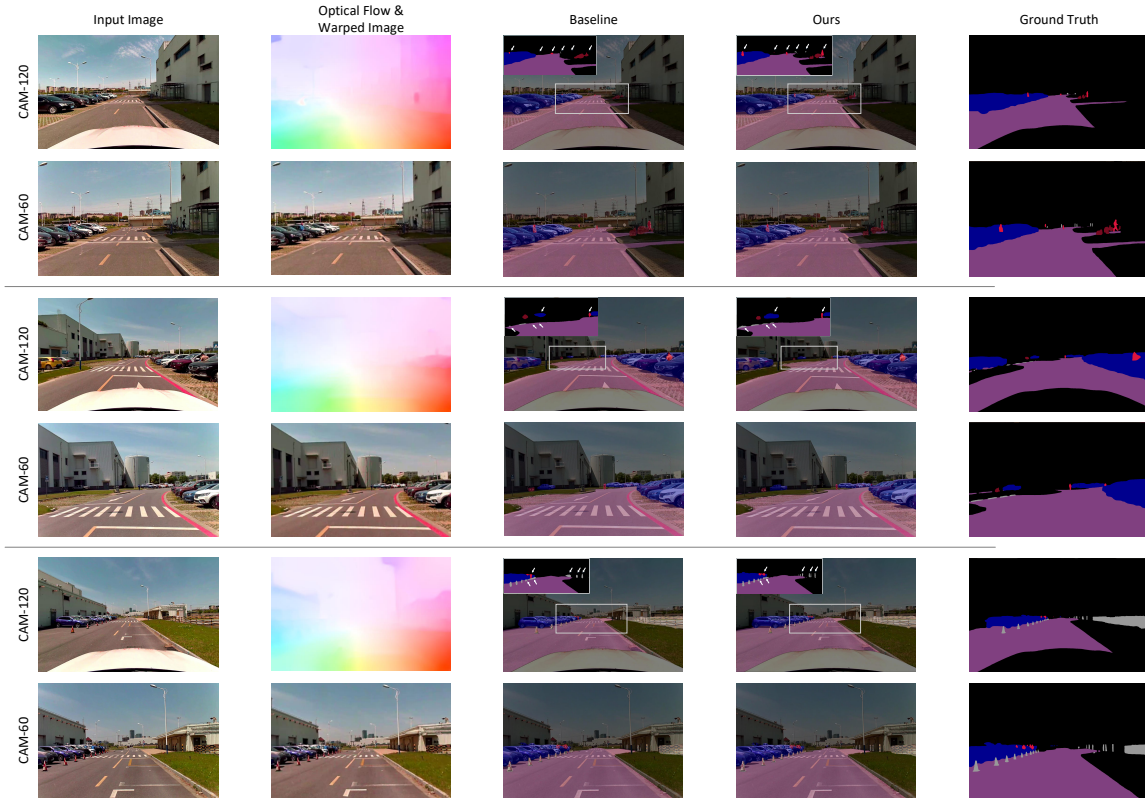


Fig. 7. Visualization of scene parsing results for the baseline and our framework. In the third and fourth column for CAM-120, the main difference between our results and the baseline are pointed out with white arrows.

TABLE II
COMPARISONS AMONG THE BASELINE AND OUR METHOD WITH LOOSELY-COUPLED AND TIGHTLY-COUPLED STRUCTURES.

Network	CAM	Mean IoU of Semantic Segmentation (%)								Model		Runtime (fps)	
		BG	Road	Person	Car	Barrier	Cycle	Avg.	Total	Params	FLOPS	Full Res.	Half Res.
Baseline	60	99.4	99.1	78.6	95.6	73.4	70.4	86.1	85.9	2.81M	34.34G	29.8	103.2
	120	99.5	98.6	74.1	95.2	78.0	68.8	85.7					
	120-OL	99.3	98.8	72.6	94.3	72.5	70.1	84.6	-				
Loosely-coupled	60	99.5	99.2	76.8	95.5	75.4	71.3	86.3	87.0	6.57M	39.28G	19.6	47.7
	120	99.6	98.8	76.4	95.7	80.3	74.7	87.6					
	120-OL	99.4	99.0	76.3	95.2	77.1	75.7	87.1	-				
Tightly-coupled	60	99.4	98.8	75.8	94.6	74.6	69.8	85.5	86.3	4.50M	33.76G	31.0	78.1
	120	99.6	98.7	75.2	95.6	80.1	72.6	87.0					
	120-OL	99.4	99.0	76.1	95.1	76.6	74.7	86.8	-				

MobileNetV3-large and trained the rest part of PWC-Net. From the comparisons of semantic segmentation in Table II for both structures, we can find that such inaccuracy of flow estimation will only remains slight effect on the segmentation results after fine-tuning the whole network.

2) *Optical Flow Estimation with different training schedules*: Since the performance of optical flow estimation can be influenced by the training schedules on different datasets [15], we also evaluated the effectiveness of the synthetic *Data_Sim* dataset. Table IV shows the comparisons of three different types of training schedules. It suggests that the training on *Data_Sim* has positive effects on the original network trained with the *FlyingChairs* [13] dataset and improves its performance on the final *Data_Real* dataset.

3) *Semantic Sharing w/ or w/o Warping by Optical Flow*: The performance of the optical flow estimation network directly affects the shared semantics. In Table V we compare the semantic segmentation results of CAM-60 branch with or without the optical flow warping in the semantics sharing module. Note that we have also skipped the feature fusion modules in the evaluation. The results depict that with only warping by perspective transformation (P.T.), the segmentation results are relatively poor especially for those classes of small objects, which means the semantics are badly propagated. After applying the warping with the optical flow, the performance has a significant enhancement (11.0%) suggesting the importance of accurate remapping.

4) *Semantic Feature Fusion with Different Types of Blocks*: Table VI illustrates the comparisons of integrating

TABLE III

COMPARISONS OF OPTICAL ESTIMATION RESULTS ON DIFFERENT DATASETS WITH DIFFERENT NETWORK STRUCTURES.

Network Structure	AEPE	Unsupervised Loss		
		L_1	SSIM	Smooth
	Data_Sim	Data_Real		
Loosely-coupled	1.86	15.4	0.376	0.018
Tightly-coupled	3.67	17.5	0.422	0.024

TABLE IV

COMPARISONS OF OPTICAL ESTIMATION RESULTS ON DIFFERENT DATASETS WITH DIFFERENT TRAINING SCHEDULES.

Training Schedule	AEPE	Unsupervised Loss		
		L_1	SSIM	Smooth
	Data_Sim	Data_Real		
Chairs	1.07	22.8	0.429	0.192
Chairs-Simulate	0.08	18.7	0.421	0.030
Chairs-Simulate-Real	1.86	15.4	0.376	0.018

TABLE V

COMPARISONS OF SEMANTIC SEGMENTATION RESULTS ON DATA-REAL W/ OR W/O WARPING BY OPTICAL FLOW.

Warping	Mean IoU of Semantic Segmentation for CAM-60 (%)						
	BG	Road	Person	Car	Barrier	Cycle	Avg.
P.T.	98.0	96.5	33.9	84.6	35.7	50.0	66.4
P.T. + Flow	98.9	98.5	61.6	92.4	53.0	59.9	77.4

TABLE VI

COMPARISONS OF SEMANTIC SEGMENTATION RESULTS ON DATA-REAL DATASET WITH DIFFERENT FEATURE FUSION MODULES.

Feature Fusion Module	Mean IoU of Semantic Segmentation for CAM-60 (%)						
	BG	Road	Person	Car	Barrier	Cycle	Avg.
None	98.9	98.5	61.6	92.4	53.0	59.9	77.4
Basic	99.1	98.8	69.2	93.6	58.1	62.5	80.2
Residual	99.3	99.0	71.1	94.2	64.4	63.1	81.8
Bottleneck	99.2	99.0	71.7	94.1	62.0	65.5	81.9

different types of feature fusion blocks in CAM-60 branch as an example. We can find that even the simplest basic block can dramatically boost the final segmentation performance. The bottleneck type achieves similar outputs to the residual type in most classes as well as the total average, although it has much less parameters and needs lower computation.

V. CONCLUSIONS

In this letter we demonstrate how to boost the performance of a scene parsing task for real-time autonomous driving applications with shared semantics. A semantics sharing and fusion framework was proposed to propagate semantic features between two cameras with different perspectives and overlapped views. The shared semantics can not only reduce the duplicable computation in feature extraction procedures, but also refine the segmentation results of both cameras. In the future work we will further investigate to sharing semantics in video scene parsing to realize a more compact and faster semantic perception system.

REFERENCES

- [1] M. Siam *et al.*, "A comparative study of real-time semantic segmentation for autonomous driving," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit. Workshops*, 2018, pp. 587–597.
- [2] Tesla, "Tesla autopilot." [Online]. Available: <https://www.tesla.com/autopilot>
- [3] J. Long *et al.*, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2015, pp. 3431–3440.
- [4] V. Badrinarayanan *et al.*, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, 2017.
- [5] H. Zhao *et al.*, "Pyramid scene parsing network," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 2881–2890.
- [6] L.-C. Chen *et al.*, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proc. Euro. Conf. Comput. Vis.*, 2018.
- [7] C. Yu *et al.*, "Bisenet: Bilateral segmentation network for real-time semantic segmentation," in *Proc. Euro. Conf. Comput. Vis.*, 2018, pp. 325–341.
- [8] H. Zhao *et al.*, "Icnnet for real-time semantic segmentation on high-resolution images," in *Proc. Euro. Conf. Comput. Vis.*, 2018, pp. 405–420.
- [9] A. Howard *et al.*, "Searching for mobilenetv3," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 1314–1324.
- [10] Z. Yin *et al.*, "Geonet: Unsupervised learning of dense depth, optical flow and camera pose," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 1983–1992.
- [11] N. Mayer *et al.*, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 4040–4048.
- [12] C. Godard *et al.*, "Unsupervised monocular depth estimation with left-right consistency," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 270–279.
- [13] A. Dosovitskiy *et al.*, "Flownet: Learning optical flow with convolutional networks," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2015, pp. 2758–2766.
- [14] T.-W. Hui *et al.*, "Liteflownet: A lightweight convolutional neural network for optical flow estimation," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 8981–8989.
- [15] D. Sun *et al.*, "Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 8934–8943.
- [16] Y.-S. Xu *et al.*, "Dynamic video segmentation network," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 6556–6565.
- [17] Y. Zhu *et al.*, "Improving semantic segmentation via video propagation and label relaxation," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2019, pp. 8856–8865.
- [18] X. Jin *et al.*, "Video scene parsing with predictive feature learning," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 5580–5588.
- [19] Y. Li *et al.*, "Low-latency video semantic segmentation," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 5997–6005.
- [20] X. Wang *et al.*, "Learning correspondence from the cycle-consistency of time," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2019, pp. 2566–2576.
- [21] J. Lee *et al.*, "Sfnet: Learning object-aware semantic correspondence," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2019, pp. 2278–2287.
- [22] X. Li *et al.*, "Foveanet: Perspective-aware urban scene parsing," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 784–792.
- [23] J. Jiao *et al.*, "Geometry-aware distillation for indoor semantic segmentation," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2019, pp. 2869–2878.
- [24] L. Hoyer *et al.*, "Short-term prediction and multi-camera fusion on semantic grids," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, 2019.
- [25] R. Szeliski, *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.
- [26] Z. Wang *et al.*, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, 2004.
- [27] P. Heise *et al.*, "Pm-huber: Patchmatch with huber regularization for stereo matching," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 2360–2367.