Supplementary Material

Anonymous ECCV submission

Paper ID 4584

1 Video Introduction

For a clearer presentation, a video *pipeline.mp4* related to the pipeline of our method is provided.

2 Network Architecture

The network architecture of classification and segmentation branches is shown in Table 1.

Table 1. The architecture of our network. Num_col is the number of predefined row anchors and num_row is the number of griding cells. Num_lane is 4 in our experiments.

Operation	Details	Output Size
Input	C × H × W	3 × 288 × 800
Conv1	Resnet backbone	64 × 144 × 400
Pooling	Resnet backbone	64 × 72 × 200
Layer1	Resnet backbone	64 × 72 × 200
Layer2	Resnet backbone	128 × 36 × 100
Layer3	Resnet backbone	256 × 18 × 50
Layer4	Resnet backbone	512 × 9 × 25
Auxiliary segmentation branch		
Decode1(Layer4)	Conv2D [512, 256, 3, 3], Upsample	256 × 18 × 50
CAT(Decode1, Layer3)	Along axis channel	512 × 18 × 50
Decode2	Conv2D [512, 128, 3, 3], Upsample	128 × 36 × 100
CAT(Decode2, Layer2)	Along axis channel	256 × 36 × 100
Decode3	Conv2D [256, 128, 3, 3]	128 × 36 × 100
Header0	Conv2D [128, 128, 3, 3], LeakyRelu, BN	128 × 36 × 100
Get Segmentation	Conv2D [128, 5, 3, 3], padding=1	5 × 36 × 100
Classification branch		
Conv2 (Layer4)	Conv2D [512, 8, 1, 1]	8 × 9 × 25
Reshape	Flatten	1800
FC1	Linear (1800, 2048), Relu	2048
FC2	Linear (2048, num_col × num_row × num_lane)	

3 Analysis of Structural Loss

In the derivation of structural loss, the two losses are defined as:

$$L_{sim} = \sum_{i=1}^{C}\sum_{j=1}^{h-1} \|P_{i,j,:} - P_{i,j+1,:}\|_1 , \tag{1}$$

and

$$L_{shp} = \sum_{i=1}^{C}\sum_{j=1}^{h-2} \|(Loc_{i,j} - Loc_{i,j+1}) - (Loc_{i,j+1} - Loc_{i,j+2})\|_1. \tag{2}$$

In fact, the two losses are based on the relations of lane locations between different rows. If one of the row is the "no lane" case, the assumption might not hold since the "no lane" case means no lane in this row and has no location of lane.

Fortunately, the above cases are rare and have little effects on our method. The frequency of the mentioned "no lane" case is less than 3.7%, thus it has little effects on the whole optimization. As for L_{shp}, a trick that only computing the loss on the lower part of image is used because no lane is less likely to occur in the lower part of image.

4 Speed Comparison

In order to prove that ultra fast speed of lane detection is necessary, we list the estimated speed performance on the practical devices. As shown in Table 2, The practical device is limited by power and the performance is downgraded. Meanwhile, multiple camera inputs are commonly used. For example, Tesla model X has eight cameras, which requires huge computation cost.

Table 2. Speed comparison with autonomous driving devices. FPS Low is the comparison with the standard definition of real-time. FPS Multiple means the comparison with eight camera inputs.

Device	GTX 1080Ti	Jetson Nano	Jetson TX2	AGX Xavier
TFLOPS	11.5	2	3	5.5
FPS Low	30	<5.3	<7.9	<14.4
FPS-L Multiple(1/8)	3.75	<0.7	<1.0	<1.8
FPS High	322	<57	<85	<155
FPS-H Multiple(1/8)	40.3	<7.1	<10.6	<19.4

From Table 2 we can see that the present standard of real-time (30 FPS) is nearly unusable when equipped with multiple camera inputs while the performance of our method is still acceptable (19.4 FPS with AGX Xavier under eight camera inputs).

5 Post Processing

The post processing of our method is simple. With the predicted probability distribution of lane locations, we can simply use argmax to find the locations. This is also an advantage of our method that our method requires no complex post processing of lane.

If the prediction of all row anchors is "no lane", then the prediction is discarded. In this way, we can handle any number of lanes that doesn't exceed the predefined max number.

6 Visualization

We show the visualization results of our method on eight different scenarios of CULane dataset, as shown in Fig. 1 (next page). From Fig. 1 we can see that our method could generate satisfactory visual results with different scenarios.

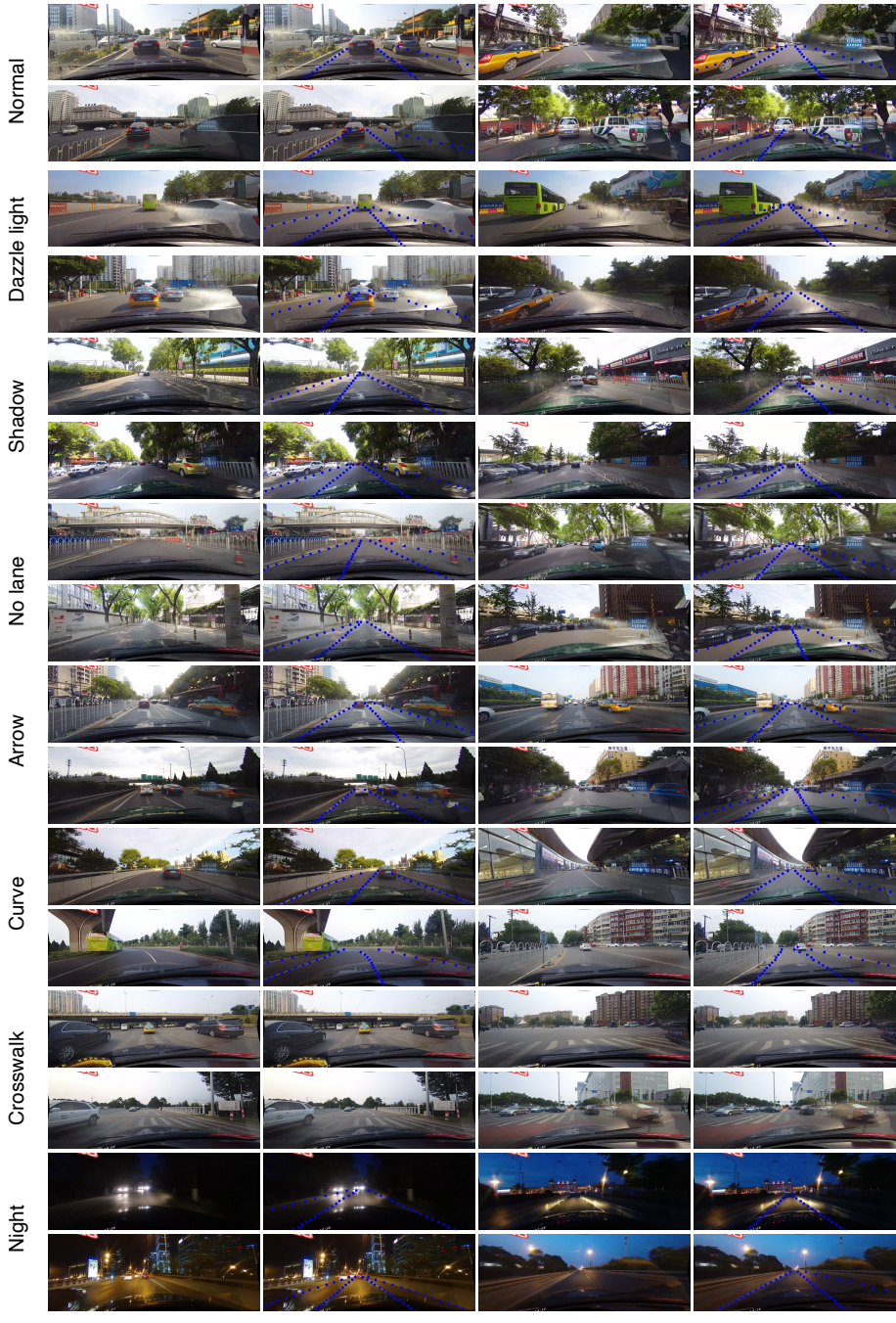

Fig. 1. Visualization of our method on eight different scenarios.