2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
September 24–28, 2017, Vancouver, BC, Canada

MFNet: Towards Real-Time Semantic Segmentation for Autonomous Vehicles with Multi-Spectral Scenes

Qishen Ha, Kohei Watanabe, Takumi Karasawa, Yoshitaka Ushiku and Tatsuya Harada

Abstract— **This work addresses the semantic segmentation of images of street scenes for autonomous vehicles based on a new RGB-Thermal dataset, which is also introduced in this paper. An increasing interest in self-driving vehicles has brought the adaptation of semantic segmentation to self-driving systems. However, recent research relating to semantic segmentation is mainly based on RGB images acquired during times of poor visibility at night and under adverse weather conditions. Furthermore, most of these methods only focused on improving performance while ignoring time consumption.**

The aforementioned problems prompted us to propose a new convolutional neural network architecture for multi-spectral image segmentation that enables the segmentation accuracy to be retained during real-time operation. We benchmarked our method by creating an RGB-Thermal dataset in which thermal and RGB images are combined. We showed that the segmentation accuracy was significantly increased by adding thermal infrared information.

I. INTRODUCTION

An autonomous vehicle (also known as a self-driving or robotic car) is a vehicle that is able to sense and understand its surrounding environment, and navigate automatically without the need for human intervention. Autonomous vehicles have increasingly drawn attention from the vehicle industry and the worldwide research community and are expected to change transportation dramatically [1]. Basically, autonomous driving has three main benefits: Firstly, it reduces pollution in urban areas by optimizing the usage of energy. Secondly, autonomous vehicles are likely to speed up the transportation of people and goods. Finally, these vehicles are expected to increase driving safety by reducing human errors. Despite autonomous vehicles becoming increasingly important, the related technology is far from mature. An autonomous driving system is required to be accurate, fast, and robust. Although the technology has been studied for many years, satisfactory robustness and real-time performance still need to be achieved.

Autonomous vehicles have to be highly aware of their surroundings before they are able to make a decision. Thus, the first step in many autonomous driving systems is perception, which relates to the input information detected by the sensors with which an autonomous car is equipped. The perception system of autonomous vehicles usually includes multiple sensors, such as cameras and ultrasound, radar, and laser sensors. The advantage of cameras is that they are able to not only record the location of obstacles but also further detailed information (e.g., category and color), and semantic segmentation allows the captured scenes to be fully understood pixel-wise. Thus, in the research presented in this paper, we focus on semantic segmentation with the aim of enabling autonomous vehicles to fully perceive their surrounding environment.

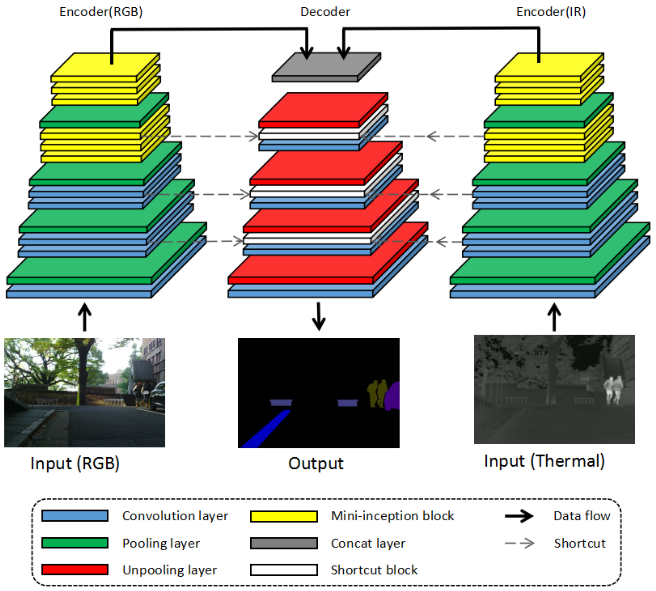


Fig. 1: Illustration of the proposed Multi-spectral Fusion Networks (MFNet) architecture. Multi-spectral images are separated into RGB images and IR images before being passed into the network, after which the process encodes each step separately. The output of the RGB and IR encoders are fused in the decode step. Information of the lower layers in the encoder is added into the higher layers of the decoder by using a shortcut. This model can be trained in an end-to-end manner. Details of the mini-inception and shortcut blocks are shown in Figure 2.

Lately, Convolutional Neural Networks (CNNs) [2] have revolutionized computer vision. Methods adopting CNNs have surpassed human performance for some computer vision tasks such as image classification [3], [4] or traffic sign recognition [5]. However, the improvement in accuracy was achieved by increasing the size of CNNs [6], [7], [3], which means that the time complexity of these state-of-the-art CNN architectures has become increasingly larger. Some methods adopted CNNs for semantic segmentation and have improved the state-of-the-art accuracy remarkably (e.g., SegNet [8] and others [9], [10], [11], [12]). However, most of these methods focused only on increasing the accuracy while overlooking

This work was partially supported by the New Energy and Industrial Technology Development Organization (NEDO).

Qishen Ha, Kohei Watanabe, and Takumi Karasawa are Master's degree candidates with the Harada/Ushiku Lab., Graduate School of Information Science and Technology, The University of Tokyo.

978-1-5386-2682-5/17/$31.00 ©2017 IEEE

the inference speed, which made these large networks are impossible or at least very challenging to implement in a self-driving car.

Public image segmentation datasets built for autonomous driving, such as CamVid [13], Cityscapes [14], and Daimler Urban Segmentation [15], are based on visible spectral (or RGB) images. However, the segmentation system for self-driving systems based on visible spectra is limited because of insufficient illumination in the evening and at night. A qualified self-driving car is required to be sufficiently robust to navigate safely during both daytime and nighttime, even during bad weather. Obviously, to build a segmentation system by using only the visible spectrum is not sufficient to achieve this robustness.

The specific contribution of this paper includes:

- Proposed a new CNN architecture, named Multi-spectral Fusion Networks (MFNet), for real-time semantic segmentation using multi-spectral images for autonomous vehicles.
- Published a new semantic segmentation dataset[1], which contains 1569 RGB-Thermal urban scene images. Pixel-level annotation of eight classes of common obstacles in traffic environments is also provided.
- Proved that using thermal infrared information can significantly improve the performance of a semantic segmentation system for self-driving at night as well as the segmentation of objects of which the temperature is higher than the surrounding environment.

II. RELATED WORK

Semantic segmentation is important for understanding the content of scenes or images and locating target objects. We proposed a new CNN architecture that performs real-time semantic segmentation for autonomous vehicles using multi-spectral images. We evaluated our proposed architecture by conducting comparisons with other comparable methods.

CNNs [2] have significantly improved the accuracy of image classification tasks [16] since 2012 and it has also been adopted in semantic segmentation in recent years. In 2015, Long *et al.* [9] proposed a semantic segmentation method, named Fully Convolutional Networks (FCN), which can be trained end-to-end for semantic segmentation and can outperform the methods that rely on the use of hand-crafted features without further machinery.

The concept of Encoder-Decoder network architecture for semantic segmentation has been introduced in SegNet [8]. The encoder of SegNet is a pre-trained VGG16 architecture, whereas the decoder is used for up-sampling the output of the encoder step by step (In FCN, the deconvolutional layer can be considered as its decoder with only one layer). SegNet achieved state-of-the-art accuracy, while gaining relatively fast inference speed. In subsequent years, considerable research went into the design of a larger decoder (than FCN) in their network architecture [10], [12], [11], [17], [18]. However, most of these networks are slow during inference because of their large number of parameters and complicated architecture. In this regard, ENet [19] is an exception in that it also followed the Encoder-Decoder architecture, but is optimized for fast inference and high accuracy. ENet can process images at an average speed of 15 ms/image (480×640 RGB) using a single NVIDIA Geforce Titan X GPU. However, ENet failed to perform as good as SegNet on datasets with many images characterized by complicated backgrounds, such as SUN RGB-D [20].

Deconvolutional Network (DeconvNet) [10] and SegNet [8] use the max locations indices of the pooling layer in the encoder to perform nonlinear up-sampling in the decoder to transfer information from the lower layer to the decoder and sharpen the boundaries of the output segmentation maps. They showed that this strategy, which we termed "indexed unpooling," could improve the segmentation accuracy.

Yu *et al.* [11] claimed that Dilated Convolution is able to aggregate multi-scale contextual information without losing resolution (note that they still used down-sampling operations), because the dilated filter can achieve a larger sized receptive field with the same number of parameters as a normal 3×3 convolutional filter. This can be used to simplify existing network architectures, as in [12], while only slightly reducing the accuracy. Our work also adopted dilated convolution in order to optimize the network architecture.

Some methods adopting CNN were designed for the RGB-Depth dataset [21], [20], which contains images that were acquired by using Kinect [22], [23]. Although there is a difference between RGB-D and multi-spectral images, we found that some ideas in this work were useful for designing our method. Hazirbas *et al.* [22] proposed a new CNN network named FuseNet, which contains two encoders that simultaneously extract features from RGB and depth images. In [23], RGB and depth feature maps were not only processed separately in the encoding phase, but also in the decoding phase.

To our knowledge, a public RGB-Thermal image segmentation dataset for autonomous driving still does not exist. Most existing urban scenario image segmentation datasets are based on visible spectral images (RGB images), such as those in CamVid [13], Cityscapes [14], and Daimler Urban Segmentation [15]. Naturally, semantic segmentation methods based on these datasets can only be used to process RGB images. Furthermore, most of these methods focus only on improving the segmentation accuracy while overlooking the inference speed. In this work, we propose new CNN architecture for multi-spectral image segmentation, with real-time performance and retention of segmentation accuracy. Then, to benchmark our method, we created an RGB-Thermal dataset that combines thermal and RGB images. We showed that the segmentation accuracy is significantly increased by adding thermal infrared information.

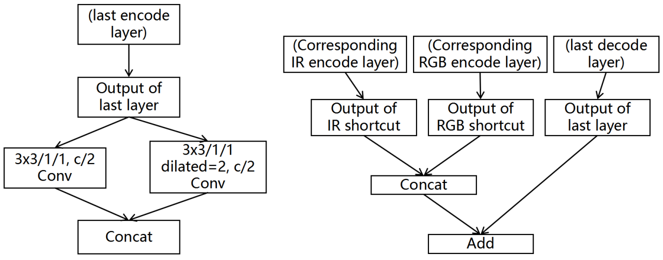
III. MULTI-SPECTRAL FUSION NETWORK

This section introduces our CNN architecture for semantic segmentation, MFNet.

[1] Our multispectral semantic segementation dataset is available online: `http://www.mi.t.u-tokyo.ac.jp/static/projects/mil_multispectral`

TABLE I: MFNet architecture. Output size is given for input consisting of a 480×640 RGB-Thermal image. During the training phase, the last layer of the decoder was followed by a softmax cross-entropy function for backpropagation.

		Type	Output size (RGB)	Output size (IR)
Encoder	stage 1	conv 3×3	16×480×640	16×480×640
		max-pooling 2×2	16×240×320	16×240×320
	stage 2	2 * conv 3×3	48×240×320	16×240×320
		max-pooling 2×2	48×120×160	16×120×160
	stage 3	2 * conv 3×3	48×120×160	16×120×160
		max-pooling 2×2	48×60×80	16×60×80
	stage 4	3 * mini-inception blocks	96×60×80	32×60×80
		max-pooling 2×2	96×30×40	32×30×40
	stage 5	3 * mini-inception blocks	96×30×40	32×30×40
		Type	Output size	
Decoder	stage 5	concat	128×30×40	
	stage 4	unpooling	128×60×80	
		shortcut block	128×60×80	
		conv 3×3	64×60×80	
	stage 3	unpooling	64×120×160	
		shortcut block	64×120×160	
		conv 3×3	64×120×160	
	stage 2	unpooling	64×240×320	
		shortcut block	64×240×320	
		conv 3×3	32×240×320	
	stage 1	unpooling	32×480×640	
		conv 3×3	9×480×640	



(a) (b)

Fig. 2: (a) Mini-inception block in the MFNet encoder. 3×3/1/1, $c/2$ conv denotes a convolutional layer with 3×3 filters, stride=1, zero-padding=1, and the number of output channels is $c/2$. Here c is the number of output channels of the mini-inception block. (b) Shortcut block in MFNet decoder. “Concat” and “Add” signify concatenation and pixel-wise addition, respectively.

A. Objectives

We aimed to design a model capable of real-time performance and which provides satisfactory accuracy for multispectral input images. The design objectives include:

- Efficiency. One of our main goals towards applications for autonomous vehicles is real-time performance. Based on a single NVIDIA Geforce Titan X GPU, CUDA 7.5 [24], and the cuDNN 5.0 [25] library, our goal is to design a network architecture that can process more than 50 images per second. In this survey, all reported results are based on the same environment.
- Accuracy. We aimed to compare the accuracy of our method with that of SegNet [8]. SegNet is a recent state-of-the-art image segmentation method with a high inference speed. Our goal is to provide accuracy that is equal to or greater than that of SegNet and to achieve real-time performance.
- Robustness. Compared to using RGB images only, we consider the use of both visible and thermal spectral images capable of improving the accuracy of semantic segmentation, especially at night.

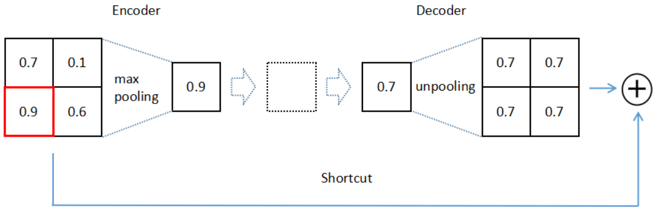


Fig. 3: Example of a shortcut in MFNet.

Rather than maximizing the accuracy, our goal is to achieve a good balance between accuracy and inference speed.

B. Network Architecture

The architecture of our network is illustrated in Figure 1 and additional details are provided in Table I. As with most recent image segmentation networks [10], [8], [26], [18], [11], MFNet adopted the Encoder-Decoder structure. Initially, we found the performance when training using a 4-channel (RGB-Thermal) segmentation network to be even worse than that of the usual 3-channel (RGB) network. Similar to FuseNet [22], we designed two encoders to extract features from RGB images and IR images, respectively. The architecture of these encoders is exactly the same except for the number of input and output channels of the convolutional layers.

Although it is usually necessary to process images with higher resolution (e.g., 480×640 or even higher) in image segmentation, SegNet or DeconvNet adopted VGG16 [6], which was designed to only process 224 × 224 images. Because the input images we need to process have a relatively larger resolution size, we adopted dilated convolution in the block “mini-inception” in the later stage of the encoders. It has a single input tensor to operate 3×3 convolutions and 3×3, dilated=2 convolution [11] in parallel, after which the output of these two convolutional layers are concatenated back into a single tensor as the output of the block. This process is depicted in Figure 2 (a). A mini-inception block has the same time complexity and number of parameters, compared to a normal 3×3 convolutional layer. We proved that this approach outperformed normal convolution or the use of dilated convolution on its own.

The number of channels of the convolutional layers in MFNet is set to a relatively small number to increase the speed of the network, and each convolutional layer in MFNet is followed by batch normalization [27], following which the nonlinear activation function leaky-Rectified Linear Unit (leaky-ReLU) [28] is applied. The subsampling operation in the encoder of MFNet is performed by employing max-pooling layers, all of which have a 2×2 window and stride 2 (non-overlap). On the other hand, up-sampling in the decoder is performed by unpooling (filling all units in the unpooling filter with the same number). A subsampling operation is used to ensure a degree of translational invariance and reduce the spatial resolution of the output feature maps in order to decrease the time cost of following layers.

We designed a small decoder that only contains a single convolutional layer in each stage in order to reduce the parameters of the architecture as well as to speed up the inference. When decoding begins, the outputs of both the RGB and IR encoders are combined by a concatenation operation. During decoding, the data obtained directly from the last layer are combined with low-level feature maps stored from the corresponding layers of the encoder by element-wise addition operation (an example is shown in Figure 3) in order to improve the up-sampling. We term the block containing these operations the "shortcut block", which is illustrated in Figure 2 (b). The up-sampled feature maps are then passed to convolutional layers to produce dense feature maps. The output of the final convolutional layer of the decoder in MFNet is a c channels image, where c is the number of classes of the training dataset ($c = 9$ in our dataset). During training, the output is fed into a softmax cross-entropy layer for backpropagation. During inference, pixels are independently classified using the output obtained without using the softmax operation to reduce the inference time. Although softmax provides a more acceptable mathematical explanation, it is actually unnecessary for prediction.

C. *Training*

During training and testing, we simply projected all pixels from range $[0, 255]$ to range $[0, 1]$. As an optimizer, we adopted momentum stochastic gradient descent [29] with a learning rate of 0.01, and at each epoch, the learning rate was multiplied by a decay rate of 0.94. We trained each model for 80 epochs until the loss converged. The training set contained 784 images, and 1568 randomly shuffled images (including 784 flipped images) were fed into the network at each epoch. The size of a mini-batch was set to 6 and each image was used only once in an epoch. We used the cross-entropy loss [9] as the objective function for backpropagation.

As indicated in Table IV, the inference speed of MFNet is approximately 55 images/s on a single NVIDIA Geforce Titan X GPU. According to the latest news from NVIDIA, the Drive PX2 GPU has a higher FLOPS value than the Geforce Titan X. Thus, higher speed performance can be expected when the former of these two GPUs is practically implemented in autonomous vehicles. In addition, the number of parameters of MFNet is sufficiently small such that it would be possible to easily fit the whole model into the memory of the embedded processor in an autonomous car.

IV. DATASET

We used an InfRec R500 as our RGB and IR camera. This camera can capture images in both the visible and thermal infrared spectrum (814μm) with different lenses and sensors. For convenience, we henceforth use the term color camera to indicate the visible spectral part of the R500 and thermal camera to indicate the thermal infrared spectral part of the R500. Both the color and thermal camera have 480×640 pixels of spatial resolution. The color camera has a 100° horizontal field of view, whereas the thermal camera has a 32° horizontal field of view. Because the thermal camera has a smaller field of view compared to the color camera, we cropped a fixed area from each of the RGB images and resized them to 480×640, before combining them with the corresponding IR images.

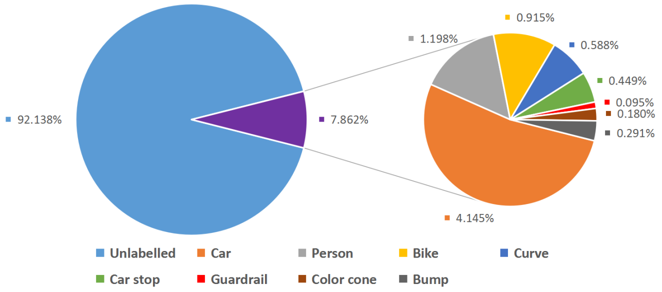


Fig. 4: Percentage of the number of pixels in each class of our dataset.

As far as we know, our dataset is the first RGB-Thermal urban scene image dataset with pixel-level annotation. We published this new RGB-Thermal semantic segmentation dataset in support of further development of autonomous vehicles in the future. This dataset contains 1569 images (820 taken at daytime and 749 taken at nighttime). Eight classes of obstacles commonly encountered during driving (car, person, bike, curve, car stop, guardrail, color cone, and bump) are labeled in this dataset. Because only a few categories of each entire scene are labeled in our dataset, unlabeled pixels occupied the majority of all pixels (much sparser than VOC2012 [30]) and the number of pixels is extremely unbalanced between each class, similar to any other image segmentation dataset. The percentage of the number of pixels in each class is showed in Figure 4.

We divided our dataset into three parts: the training set, validation set and test set. The training set includes 50% of the daytime images and 50% of the nighttime images, whereas the validation and test sets contain 25% of the daytime images and 25% of the nighttime images, respectively. The dataset was separated into three parts according to the time series when the images were captured, without balancing the frequency of each class in each set of the data manually. Therefore, the difference in distribution between each part of the dataset introduces additional challenges to the segmentation algorithm.

V. RESULTS

In this section, we benchmarked the performance of MFNet on our dataset, which we introduced in the last section, to demonstrate real-time performance and accuracy. We compared the performance of our method with that of SegNet [8] and ENet [19]. This is because SegNet [8] is fast and achieves state-of-the-art accuracy and ENet is currently the fastest semantic segmentation network architecture. Our models were trained, tested, and evaluated using Python with the Chainer 1.17.0 [31] machine learning library, CUDA 7.5 [24], and the cuDNN 5.0 backend [25] with a single NVIDIA Geforce Titan X GPU. We used LUA with Torch7 library when testing the inference speed, because, contrary to most machine-learning libraries, the inference speed of CNNs in Torch7 is independent of the input batch size.

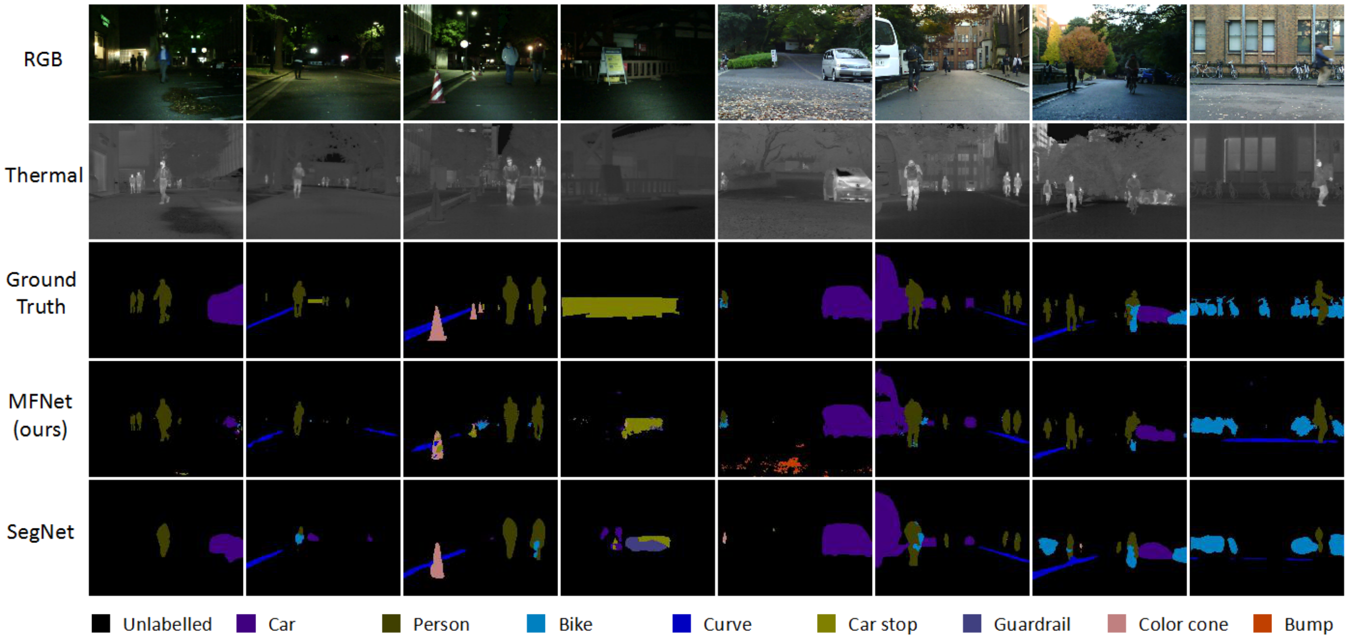


Fig. 5: Some acceptable prediction examples of MFNet(ours) and SegNet [8] on our dataset.

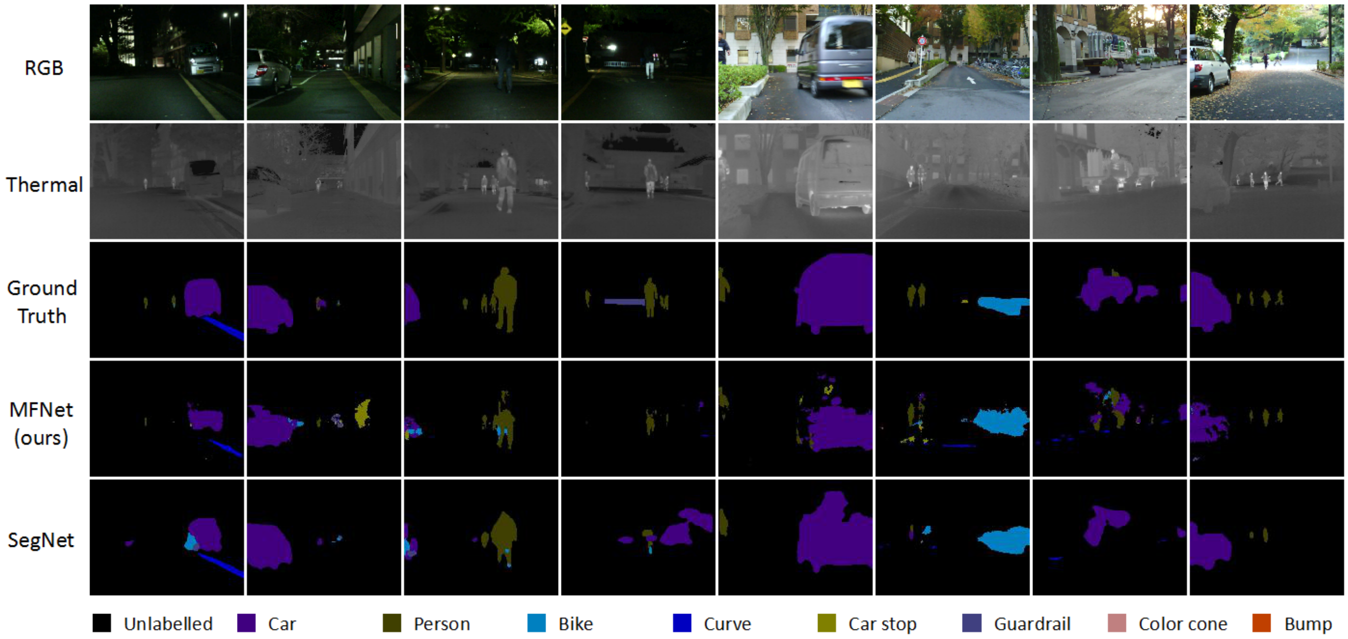


Fig. 6: Some unacceptable prediction examples of MFNet(ours) and SegNet [8] on our dataset.

We adopted two measures to evaluate the segmentation accuracy of models trained on our dataset. The first is Average Class Accuracy (class avg.) and the other is Mean Intersection of Union (mIoU). Class avg. is the average value of accuracy on each class, whereas IoU is the intersection of the inferred segmentation and the ground truth, divided by the union, and mIoU is the average value of IoU of each class. These measures are defined as follows:

$$classAvg. = \frac{1}{n}\sum_{i=1}^{n}\left(P_{ii}\middle/\sum_{j=1}^{n}P_{ij}\right) \qquad (1)$$

$$mIoU = \frac{1}{n-1}\sum_{i=2}^{n}\left(P_{ii}\middle/\left(\sum_{j=2}^{n}(P_{ij}+P_{ji})-P_{ii}\right)\right) \qquad (2)$$

Here n is the number of classes (including unlabeled) and class ID=1 indicates “unlabeled,” P_{ij} is the number of pixels belonging to class i and is predicted as class j . Note that we did not take the class “unlabeled” (class number 1) into account when calculating mIoU because “unlabeled” constitutes the majority of all pixels (more than 92%), which would cause the mIoU of those models to be similar to each other.

As seen in Table IV and II, MFNet achieves real-time performance while providing similar or higher accuracy compared to SegNet. The test data of our dataset is split into two parts, the nighttime and daytime parts, respectively. We tested SegNet, ENet, and MFNet on all the test data, after which the daytime and nighttime parts were tested separately by SegNet and MFNet. Here SegNet and ENet only use RGB images from our dataset. SegNet_4ch is a model that was trained on 4-channel (RGB-Thermal) images.

Some prediction examples of MFNet and SegNet are presented in Figure 5, 6. Compared to SegNet, the ability of MFNet to predict pedestrians has sharper boundaries. However, regarding the boundaries of the prediction of cars or bikes, SegNet performs more accurately.

The limitation of MFNet is as follows:

- In order to speed up the network, we set the number of channels in each convolutional layer to a small number compared to SegNet or DeconvNet [10], which means some important features could be detected by SegNet that cannot be detected by MFNet. This leads to a slight decrease in accuracy. If we were to continue to reduce the number of channels based on current MFNet architecture, the segmentation accuracy would decrease significantly.
- We reduced the size and accelerated the inference time of the network by designing a tiny decoder for MFNet. This decoder only has one convolutional layer in each decoding stage, which means the size of the receptive field of the decoder in MFNet is smaller compared to SegNet. Thus, MFNet cannnot bound some large objects as well as SegNet.
- Limited by the size of the dataset, the distribution of the training set and test set is quite different. Thus, if we were to test an image with a complex background, models would be confused by it and the result of the prediction would not be interpretable.

A. Multi-Encoders

In this experiment, we showed that the use of multi-encoder architecture can improve the accuracy of semantic segmentation, whereas segmentation models directly trained on combined RGB-Thermal images cannot achieve this improvement. SegNet_4ch (model A) in Table III is a model based on SegNet, in which we changed the number of input channels of the first layer from three to four and used our 4-channel images (RGB-Thermal) for training. Compared to the result of SegNet and model A in Table III, we found that directly combining thermal images with RGB images and using 4-channel images as input data made the accuracy even worse compared to using the normal 3-channel (RGB) images. The original images were slightly misaligned due to the positions of the cameras and the timing at which images were captured, so we surmise that the reduced accuracy is because the insufficient size of the receptive field in lower layers (particularly the first convolutional layer) cannot compensate for this misalignment. MFNet_RGB (model B) is a model based on MFNet that neither employed the IR encoder nor the shortcut blocks (the shortcut of the RGB encoder is retained). This model was trained on RGB images in our dataset. The results of MFNet and model B in Table III show that using RGB-Thermal images processed by a multi-encoder can significantly improve the segmentation accuracy.

B. Mini-Inception

We investigated the effect of using a mini-inception block. The time complexity of the mini-inception block is the same as a normal 3×3 convolutional layer when the number of input and output channels is the same. However, the size of the receptive field is enlarged. In this experiment, we trained two models, named MFNet_noInc1 (model C) and MFNet_noInc2 (model D) in Table III. Model C is based on MFNet but replaces all mini-inception blocks with normal 3×3 convolutional layers without changing the number of input and output channels. Model D is also based on MFNet but replaces all mini-inception blocks with 3×3, dilated=2 convolutional layers. Comparing the results of MFNet and model C, D in Table III, we found that the accuracy was improved by using mini-inception blocks.

C. Activation Function

This experiment showed the benefits of using leaky-ReLU [28]. Normal ReLU [32] has been the most common nonlinear activation function used in CNNs lately. Many well-known CNN architectures applied ReLU as their non-linear activation function, e.g., AlexNet [16], VGG16 [6], GoogLeNet [7], and ResNet [3]. However, we encountered significant degradation in accuracy when we adopted ReLU in MFNet. We considered the degradation in accuracy by using ReLU to be caused by the insufficient number of channels in the convolutional layers in the network. We devised an experiment to investigate this and reported the results in Table III. MFNet_relu (model F) in Table III is a model based on MFNet for which all activation functions were changed from leaky_ReLU [28] to ReLU. Comparing MFNet and model F, the decrease in accuracy is obvious. Subsequently, we doubled the number of channels of all convolutional layers in MFNet and model F, respectively. These two models are named MFNet_2x (model E) and MFNet_relu_2x (model G). The results show that although the accuracy of model G continues to be worse than its leaky_ReLU version (model E), the difference in accuracy is not as significant as the difference between MFNet and model F. According to the mechanism of ReLU, a large gradient flowing through a ReLU neuron could cause the weights to update in a way that would prevent the neuron from ever activating on any data again. The output and the gradients of neurons such as this (we refer to them as "dead" neurons) would be zero for all the time. These "dead" neurons cannot be activated by any incoming data, which means they are unable to detect any features and do not contribute to the accuracy of the model. When designing a relatively small CNN architecture with a small number of input and output channels, we suggest that using an activation function that would provide a non-zero gradient when the unit is not active could enhance the performance.

TABLE II: Results on images from our test set: [8], ENet [19], and MFNet(ours).

Model	Image type	Unlabeled	Car	Pedestrian	Bike	Curve	Car stop	Guardrail	Color cone	Bump	Class avg.	mean IoU
SegNet [8]	both	0.969	0.833	0.721	**0.768**	0.583	**0.319**	0.	0.	0.639	0.537	0.583
SegNet_4ch (A)		0.961	**0.890**	0.823	0.	0.614	0.217	0.	0.	**0.867**	0.486	0.504
ENet [19]		0.885	0.586	0.427	0.247	0.301	0.181	0.003	0.458	0.230	0.370	0.449
MFNet (**ours**)		0.968	0.829	**0.852**	0.742	**0.615**	0.273	0.	**0.607**	0.433	**0.591**	**0.649**
SegNet	day	0.970	0.894	0.769	**0.770**	**0.432**	**0.316**	0.	0.	0.	0.461	0.488
	night	0.969	0.578	0.627	**0.759**	0.679	0.323	0.	0.	**0.928**	0.540	0.552
SegNet_4ch (A)	day	0.954	**0.909**	0.801	0.	0.328	0.156	0.	0.	**0.260**	0.379	0.482
	night	0.968	**0.791**	0.856	0.	0.696	**0.342**	0.	0.	0.891	0.505	0.511
MFNet (**ours**)	day	0.969	0.885	**0.830**	0.740	0.379	0.232	0.	**0.258**	0.003	**0.477**	**0.574**
	night	0.967	0.557	**0.910**	0.752	**0.772**	0.325	0.	**0.724**	0.710	**0.635**	**0.621**

TABLE III: Summary of results of all experiments in section V.

	Model	class avg.	mean IoU	Inference time
Base	SegNet [8]	0.5368	0.5833	119 ms
	MFNet (**ours**)	0.5910	0.6486	18 ms
sec V-A	SegNet_4ch (A)	0.4858	0.5041	120 ms
	MFNet_RGB (B)	0.4971	0.5835	14 ms
sec V-B	MFNet_noInc1 (C)	0.5360	0.6379	18 ms
	MFNet_noInc2 (D)	0.5572	0.6350	18 ms
sec V-C	MFNet_2x (E)	0.6493	0.6861	43 ms
	MFNet_relu (F)	0.4248	0.4738	18 ms
	MsFNet_relu_2x (G)	0.5898	0.6375	43 ms
sec V-D	MFNet_idUp (H)	0.3138	0.1936	18 ms
	MFNet_idUp_2x (I)	0.5023	0.4691	43 ms
	MFNet_noSc (J)	0.5800	0.6406	18 ms
sec V-E	MFNet_sqQuart (K)	0.4964	0.5882	17 ms
	MFNet_sqHalf (L)	0.4949	0.5990	18 ms
	MFNet_sqHalf_noSc (M)	0.5003	0.5926	18 ms
	MFNet_sqSame (N)	0.5405	0.6484	19 ms

TABLE IV: Inference speed and model size of SegNet [8], ENet [19], and MFNet. The inference speed was measured by running a test on a single NVIDIA Geforce Titan X GPU given an input image with 480×640 resolution.

Model	Inference Speed		Parameters	Model Size
	RGB	RGB+Thermal		
SegNet [8]	8.3 fps	8.3 fps	29.42M	110.2MB
ENet [19]	66.7 fps	–	0.37M	1.4MB
MFNet (**ours**)	–	55.6 fps	0.73M	3.7MB

D. Shortcut and Indexed Unpooling

In this experiment, we showed that the shortcuts can be adopted in CNN network architecture with a small number of channels to improve the accuracy while indexed unpooling cannot. SegNet and DeconvNet use pooling indices computed in the max-pooling layer of the corresponding layer in the encoder. This improved the accuracy of semantic segmentation without adding much to the computational cost (we refer to this strategy as "indexed unpooling"). Indexed unpooling is used to transfer the information on lower layers to higher layers in order to compensate for the loss of resolution caused by down-sampling. Our attempts to adopt indexed unpooling in MFNet revealed that the segmentation accuracy of the model became critically low. Therefore, we decided to experiment to investigate its cause. Eventually, we found that the convolutional layers need a large number of channels to represent the feature maps because of the sparsity of the output of indexed unpooling. MFNet_idUp, MFNet_idUp_2x (model H,I) in Table III were designed for this experiment. Both of these models are based on MFNet by removing shortcuts and replacing unpooling with indexed unpooling. In addition, in model I, the number of channels of each convolutional layer is doubled. The results of model H, I in Table III show that providing a larger number of channels in each convolutional layer increases the accuracy remarkably, which means a large number of channels in convolutional layers is necessary if we want to adopt indexed unpooling to improve our network. In other words, indexed unpooling cannot be adopted in a small CNN network architecture with small number of channels to increase the accuracy.

We used the shortcut to utilize the lower layer information to improve up-sampling. In MFNet, the feature maps of the last convolutional layer of each stage in the encoder are stored and combined with the corresponding layers in the decoder by pixel-wise addition operation. This process is illustrated in Figure 2 (b). Compared to indexed unpooling, a shortcut retains more detailed information in the lower layers and outputs denser feature maps. This approach makes it possible to improve the segmentation accuracy of a relatively small network with a small number of channels for the convolutional layers. We demonstrated the effectiveness of a shortcut by designing MFNet_noSc (model N), which is based on MFNet but from which all shortcut blocks are removed. A comparison of the results of model H and MFNet in Table III shows that the segmentation accuracy is improved by using the shortcut, at the expense of adding inference time by approximately 0.5 ms.

E. 1×1 Convolution

This experiment proved that using 1×1 convolution does not contribute to improve the performance of semantic segmentation. SqueezeNet [33] achieved a 50× reduction in the model size compared to AlexNet while maintaining the accuracy. However, we found that using 1×1 convolutional layers to "squeeze" the number of channels of convolutional layers in a semantic segmentation network has a significantly negative effect on accuracy. In this experiment, we designed three models to demonstrate the degradation in accuracy by using 1×1 convolutional layers. MFNet_sqQuart, MFNet_sqHalf, MFNet_sqHalf_noSc, MFNet_sqSame (model I,J,K,L) in Table III are models based on MFNet, with a 1×1 convolutional layer added to the beginning of each mini-inception block of these models. The number of channels is squeezed to one quarter by 1×1 convolution in model I. Then, in model L and K, in all 1×1 convolutional layers the number of channels is reduced to half of the number of input channels, model L retains the shortcut, whereas model M does not. In model N, the 1×1 convolutional layers have the same number of channels as the number of input channels. Comparing the results of MFNet, the accuracy of model I and J was reduced significantly when squeezing the number of channels by 1×1 convolution. The result of model N shows that keeping the number of channels constant in 1×1 convolutional layers does not improve the accuracy either. Model M shows that the degradation in accuracy is not caused by the shortcut.

VI. CONCLUSIONS

We proposed a new CNN architecture for the semantic segmentation of RGB-Thermal images for autonomous vehicles. A new multispectral dataset with pixel-level annotation was also introduced.

We used a NVIDIA Geforce Titan X GPU to show that a semantic segmentation network can be small and sufficiently fast to achieve real-time performance55 images/s. At the same time, our approach was shown to provide similar or higher accuracy than state-of-the-art segmentation methods such as SegNet [8]. We designed two encoders to process RGB and thermal images, respectively, and improved the accuracy significantly.We also proposed a new way of adopting dilated convolution [11] in the form of a mini-inception block, which improved the accuracy yet obviated the need to increase the time complexity and the number of parameters. Besides global features, we also made use of local features in the lower layers of the network to sharpen the class boundaries of the prediction by shortcuts.

We found that, for a relatively small number of channels in the convolutional layers in the network, our suggestion that the use of activation functions that provide a non-zero gradient when the unit is not activated, such as leaky-ReLU [28], can enhance the performance considerably compared to using ReLU [32].

REFERENCES

[1] J. Ziegler, P. Bender, M. Schreiber, H. Lategahn, T. Strauss, C. Stiller, T. Dang, U. Franke, N. Appenrodt, C. G. Keller, *et al.*, "Making bertha drivean autonomous journey on a historic route," *IEEE Intelligent Transportation Systems Magazine*, vol. 6, no. 2, pp. 8–20, 2014.

[2] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[4] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[5] D. CireşAn, U. Meier, J. Masci, and J. Schmidhuber, "Multi-column deep neural network for traffic sign classification," *Neural Networks*, vol. 32, pp. 333–338, 2012.

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv:1409.1556*, 2014.

[7] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *CVPR*, 2015.

[8] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *arXiv:1511.00561*, 2015.

[9] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *CVPR*, 2015.

[10] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *CVPR*, 2015.

[11] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *ICLR*, 2016.

[12] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," in *ICLR*, 2015.

[13] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla, "Segmentation and recognition using structure from motion point clouds," in *ECCV*, 2008.

[14] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *CVPR*, 2016.

[15] T. Scharwächter, M. Enzweiler, U. Franke, and S. Roth, "Efficient multi-cue scene segmentation," in *GCPR*, 2013.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.

[17] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *CVPR*, 2015.

[18] S. Hong, H. Noh, and B. Han, "Decoupled deep neural network for semi-supervised semantic segmentation," in *NIPS*, 2015.

[19] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "Enet: A deep neural network architecture for real-time semantic segmentation," *arXiv:1606.02147*, 2016.

[20] S. Song, S. P. Lichtenberg, and J. Xiao, "Sun rgb-d: A rgb-d scene understanding benchmark suite," in *CVPR*, 2015.

[21] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik, "Learning rich features from rgb-d images for object detection and segmentation," in *ECCV*, 2014.

[22] C. Hazirbas, L. Ma, C. Domokos, and D. Cremers, "Fusenet: Incorporating depth into semantic segmentation via fusion-based cnn architecture," in *ACCV*, 2016.

[23] J. Wang, Z. Wang, D. Tao, S. See, and G. Wang, "Learning common and specific features for rgb-d semantic segmentation with deconvolutional networks," in *ECCV*, 2016.

[24] J. Nickolls, I. Buck, M. Garland, and K. Skadron, "Scalable parallel programming with cuda," *Queue*, vol. 6, no. 2, pp. 40–53, 2008.

[25] S. Chetlur, C. Woolley, P. Vandermersch, J. Cohen, J. Tran, B. Catanzaro, and E. Shelhamer, "cudnn: Efficient primitives for deep learning," *arXiv:1410.0759*, 2014.

[26] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr, "Conditional random fields as recurrent neural networks," in *CVPR*, 2015.

[27] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*, 2015.

[28] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *ICML*, 2013.

[29] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Cognitive modeling*, vol. 5, no. 3, p. 1, 1988.

[30] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International Journal of Computer Vision*, vol. 111, no. 1, pp. 98–136, 2015.

[31] S. Tokui, K. Oono, S. Hido, and J. Clayton, "Chainer: a next-generation open source framework for deep learning," in *NIPS*, 2015.

[32] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks." in *Aistats*, 2011.

[33] F. N. Iandola, M. W. Moskewicz, K. Ashraf, S. Han, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 1mb model size," *arXiv:1602.07360*, 2016.