Multi-stream CNN based Video Semantic Segmentation for Automated Driving

Ganesh Sistu[1], Sumanth Chennupati[2] and Senthil Yogamani[1]
[1]*Valeo Vision Systems, Ireland*
[2]*Valeo Troy, U.S.A.*

Keywords: Semantic Segmentation, Visual Perception, Automated Driving.

Abstract: Majority of semantic segmentation algorithms operate on a single frame even in the case of videos. In this work, the goal is to exploit temporal information within the algorithm model for leveraging motion cues and temporal consistency. We propose two simple high-level architectures based on Recurrent FCN (RFCN) and Multi-Stream FCN (MSFCN) networks. In case of RFCN, a recurrent network namely LSTM is inserted between the encoder and decoder. MSFCN combines the encoders of different frames into a fused encoder via 1x1 channel-wise convolution. We use a ResNet50 network as the baseline encoder and construct three networks namely MSFCN of order 2 & 3 and RFCN of order 2. MSFCN-3 produces the best results with an accuracy improvement of 9% and 15% for Highway and New York-like city scenarios in the SYNTHIA-CVPR'16 dataset using mean IoU metric. MSFCN-3 also produced 11% and 6% for SegTrack V2 and DAVIS datasets over the baseline FCN network. We also designed an efficient version of MSFCN-2 and RFCN-2 using weight sharing among the two encoders. The efficient MSFCN-2 provided an improvement of 11% and 5% for KITTI and SYNTHIA with negligible increase in computational complexity compared to the baseline version.

1 INTRODUCTION

Semantic segmentation provides complete semantic scene understanding wherein each pixel in an image is assigned a class label. It has applications in various fields including automated driving (Horgan et al., 2015) (Heimberger et al., 2017), augmented reality and medical image processing. Our work is focused on semantic segmentation applied to automated driving which is discussed in detail in the survey paper (Siam et al., 2017a). Recently, this algorithm has matured in accuracy which is sufficient for commercial deployment due to advancements in deep learning. Most of the standard architectures make use of a single frame even when the algorithm is run on a video sequence. Efficient real-time semantic segmentation architectures are an important aspect for automated driving (Siam et al., 2018). For automated driving videos, there is a strong temporal continuity and constant ego-motion of the camera which can be exploited within the semantic segmentation model. This inspired us to explore temporal based video semantic segmentation. This paper is an extension of our previous work on RFCN (Siam et al., 2017c).

In this paper, we propose two types of architectures namely Recurrent FCN (RFCN) and Multi-Stream FCN (MSFCN) inspired by FCN and Long short-term memory (LSTM) networks. Multi-Stream Architectures were first introduced in (Simonyan and Zisserman, 2014) in which a two stream CNN was proposed for action recognition. They were also successfully used for other applications like Optical Flow (Ilg et al., 2016), moving object detection (Siam et al., 2017b) and depth estimation (Ummenhofer et al., 2016). However, this has not been explored for semantic segmentation using consecutive video frames to the best of our knowledge. The main motivation is to leverage temporal continuity in video streams. In RFCN, we temporally processed FCN encoders using LSTM network. In MSFCN architecture, we combine the encoder of current and previous frames to produce a new fused encoder of same feature map dimension. This would enable keeping the same decoder.
The list of contributions include:

- Design of RFCN & MSFCN architectures that extends semantic segmentation models for videos.
- Exploration of weight sharing among encoders for computational efficiency.

Sistu, G., Chennupati, S. and Yogamani, S.
Multi-stream CNN based Video Semantic Segmentation for Automated Driving.
DOI: 10.5220/0007248401730180
In *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2019)*, pages 173-180
ISBN: 978-989-758-354-4

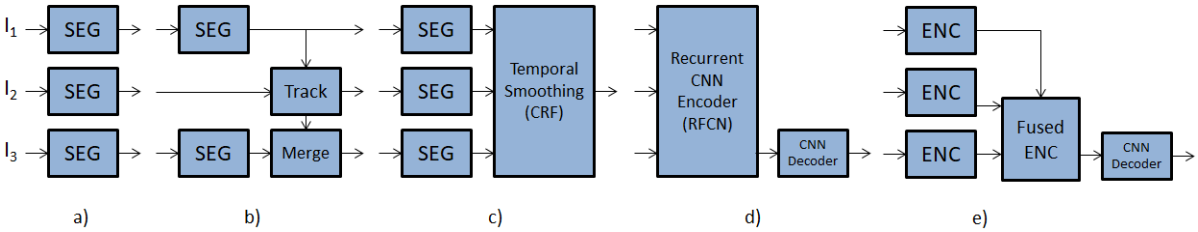


Figure 1: Comparison of different approaches to extend semantic segmentation to videos - a) Frame-level output b) Detect and track c) Temporal post processing d) Recurrent encoder model and e) Fused multi-stream encoder model.

- Implementation of an end-to-end training method for spatio-temporal video segmentation.
- Detailed experimental analysis of video semantic segmentation with automated driving datasets KITTI & SYNTHIA and binary video segmentation with DAVIS & SegTrack V2 datasets.

The rest of the paper is structured as follows. Section 2 discusses different approaches for extending semantic segmentation to videos. Section 3 explains the different multi-stream architectures designed in this work. Experimental setup and results are discussed in section 4. Finally, section 5 provides the conclusion and future work.

2 EXTENDING SEMANTIC SEGMENTATION TO VIDEOS

In this section, we provide motivation for incorporating temporal models in automated driving and explain different high level methods to accomplish the same. Motion is a dominant cue in automated driving due to persistent motion of the vehicle on which the camera is mounted. The objects of interest in automotive are split into static infrastructure like road, traffic signs, etc and dynamic objects which are interacting like vehicles and pedestrians. The main challenges are posed due to the uncertain behavior of dynamic objects. Dense optical flow is commonly used to detect moving objects purely based on motion cues. Recently, HD maps is becoming a commonly used cue which enables detection of static infrastructure which is previously mapped and encoded. In this work, we explore the usage of temporal continuity to improve accuracy by implicitly learning motion cues and tracking. We discuss the various types of temporal models in Fig 1 which illustrates the different ways to extend image based segmentation algorithm to videos.

Single Frame Baseline: Fig 1 (a) illustrates the typical way the detector is run every frame independently. This would be the reference baseline for comparing accuracy of improvements by other methods.

Detect and Track Approach: The premise of this approach is to leverage the previously obtained estimate of semantic segmentation as the next frame has only incrementally changed. This can reduce the computational complexity significantly as a lighter model can be employed to refine the previous semantic segmentation output for the current frame. The high level block diagram is illustrated in Fig1 (b). This approach has been successfully used for detection of bounding box objects where tracking could even help when detector fails in certain frames. However, it is difficult to model it for semantic segmentation as the output representation is quite complex and it is challenging to handle appearance of new regions in the next frame.

Temporal Post Processing: The third approach is to use a post-processing filter on output estimates to smooth out the noise. Probabilistic Graphical Models (PGM) like Conditional Random Fields (CRF) are commonly used to accomplish this. The block diagram of this method is shown in Fig 1 (c) where recurrence is built on the output. This step is computationally complex because the recurrence operation is on the image dimension which is large.

Recurrent Encoder Model: In this approach, the intermediate feature maps from the encoders are fed into a recurrent unit. The recurrent unit in the network can be an RNN, LSTM or a GRU. Then the resulting features are fed to a decoder which outputs semantic labels. In Fig 2a, the ResNet50 encoder conv5 layer features from consecutive image streams are passed as temporal features for LSTM network. While conv4 and conv3 layer features can also be processed via the LSTM layer, the conv4 and conv3 features from two stream are concatenated followed by a convolution layer to keep the architecture simple and memory efficient.

Fused Multi-stream Encoder Model: This method can be seen as a special case of Recurrent model in

some sense. But the perspective of multi-stream encoder will enable the design of new architectures. As this is the main contribution of this work, we will describe it in more detail in next section.

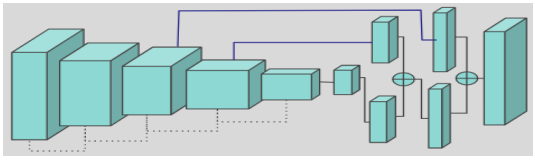
3 PROPOSED CNN ARCHITECTURES

In this section, we discuss the details of the proposed multi-stream networks shown in Fig 2b, 2c & 2d. Multi stream fused architectures (MSFCN-2 & MSFCN-3) concatenate the output from each encoder and fuse them via 1x1 channel-wise convolutions to obtain a fused encoder which is then fed to the decoder. Recurrent based architecture (RFCN) uses an LSTM unit to feed the decoder.

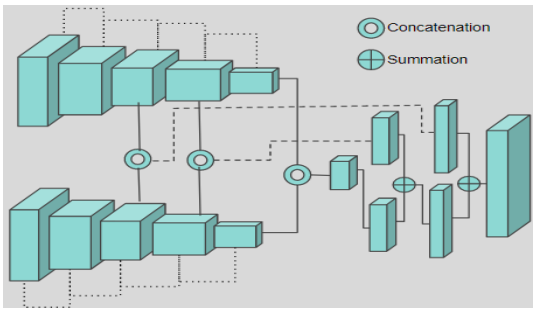
Single Stream Architecture: A fully convolution network (FCN) shown in Fig 2a is inspired from (Long et al., 2015) is used as the baseline architecture. We used ResNet50 (Kaiming He, 2015) as the encoder and conventional up-sampling with skip-connections to predict pixel wise labels. Initializing model weights by pre-trained ResNet50 weights, alleviates over-fitting problems as these weights are the result of training on a much larger dataset namely ImageNet.

Multi-stream Fused Architectures: Multi-Stream FCN architecture is illustrated in Fig 2b & 2c. We used multiple ResNet50 encoders to construct the multi-stream architectures. Consecutive input frames are processed by multiple ResNet50 encoders independently. The intermediate feature maps obtained at 3 different stages (conv3, conv4 and conv5) of encoder are concatenated and added to the up-sampling layers of the decoder. MSFCN-2 is constructed using 2 encoders while MSFCN-3 uses 3 encoders. A channel-wise 1x1 convolution is applied to fuse the multiple encoder streams into a single one of the same dimension. This will enable the usage of the same decoder.

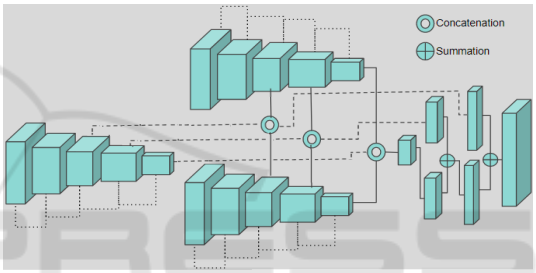
Multi-stream Recurrent Architecture: A recurrent fully convolutional network (RFCN) is designed to incorporate a recurrent network into a convolutional encoder-decoder architecture. It is illustrated in Fig 2d. We use the generic recurrent unit LSTM which can specialize to simpler RNNs and GRUs. LSTM operates over the encoder of previous N frames and produces a filtered encoder of the same dimension which is then fed to the decoder.



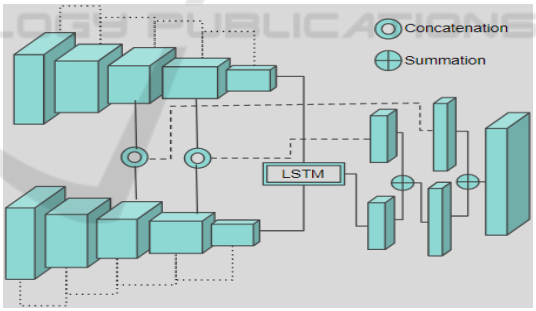
(a) FCN: Single Encoder Baseline



(b) MSFCN-2: Two stream fusion architecture



(c) MSFCN-3: Three stream fusion architecture



(d) RMSFCN-2: Two stream LSTM architecture

Figure 2: Four types of architectures constructed and tested in the paper. (a) Single stream baseline, (b) Two stream fusion architecture, (c) Three stream fusion architecture and (d) Two stream LSTM architecture.

Weight Sharing Across Encoders: The generic form of multi-stream architectures have different weights for the different encoders. In Fig 1 (e), the three encoders can be different and they have to be recomputed each frame. Thus the computational complexity of the encoder increases by a factor of three. However, if the weights are shared between the encoders, there is no need of recomputing it each frame. One

Table 1: Semantic Segmentation Results on SYNTHIA Sequences. We split the test sequences into two parts, one is Highway for high speeds and the other is City for medium speeds.

Dataset	Architecture	Mean IoU	Sky	Building	Road	Sidewalk	Fence	Vegetation	Pole	Car	Lane
Highway	FCN	85.42	0.91	0.67	0.89	0.02	0.71	0.79	0.01	0.81	**0.72**
	MSFCN-2	93.44	0.92	0.66	0.94	0.28	0.85	0.78	0.11	0.82	0.71
	RFCN-2	94.17	**0.93**	**0.71**	0.95	**0.31**	0.82	**0.83**	**0.13**	**0.87**	0.7
	MSFCN-3	**94.38**	**0.93**	0.69	**0.96**	**0.31**	**0.87**	0.81	0.12	**0.87**	**0.72**
City	FCN	73.88	**0.94**	**0.94**	0.72	0.78	0.34	0.54	0	0.69	0.56
	MSFCN-2	87.77	0.87	**0.94**	0.84	**0.83**	**0.68**	0.64	0	**0.8**	**0.8**
	RFCN-2	88.24	0.91	0.92	**0.87**	0.78	0.56	**0.67**	0	**0.8**	0.74
	MSFCN-3	**88.89**	0.88	0.89	0.86	0.74	0.64	0.53	0	0.71	0.72

Table 2: Semantic Segmentation Results on KITTI Video Sequence.

Architecture	NumParams	Mean IoU	Sky	Building	Road	Sidewalk	Fence	Vegetation	Car	Sign
FCN	23,668,680	74.00	46.18	86.50	80.60	69.10	37.25	81.94	74.35	35.11
MSFCN-2 (shared weights)	23,715,272	85.31	47.89	91.08	**97.58**	88.02	**62.60**	92.01	**90.26**	58.11
RFCN-2 (shared weights)	31,847,828	84.19	**50.20**	**93.74**	94.90	88.17	59.73	87.73	87.66	55.55
MSFCN-2	47,302,984	**85.47**	48.72	92.29	96.36	90.21	59.60	**92.43**	89.27	**70.47**
RFCN-2	55,435,540	83.38	44.80	92.84	91.77	**91.67**	58.53	86.01	87.25	52.87

Table 3: Semantic Segmentation Results on SYNTHIA Video Sequence.

Architecture	Mean IoU	Sky	Building	Road	Sidewalk	Fence	Vegetation	Pole	Car	Sign	Pedestrain	Cyclist	Lane
FCN	84.08	97.2	92.97	87.74	81.58	34.44	62	1.87	72.75	0.21	0.01	0.33	93.08
MSFCN-2 (shared)	88.88	97.08	93.14	93.58	**86.81**	47.47	75.11	46.78	**88.22**	0.27	**32.12**	2.27	95.26
RFCN-2 (shared)	88.16	96.85	91.07	**94.17**	85.62	28.29	**83.2**	**47.28**	87.6	**19.12**	16.89	**3.01**	93.97
MSFCN-2	**90.01**	**97.34**	**95.97**	93.14	86.76	73.52	73.63	35.02	87.86	3.62	27.57	1.11	**95.35**
RFCN-2	89.48	97.15	94.01	93.76	85.88	**76.26**	70.35	39.86	87.5	8.16	28.05	1.28	94.67

encoder feature extraction per frame suffices and the fused encoder is computed by combination of previously computed encoders. This weight sharing approach drastically brings down the complexity with negligible additional computation relative to the single stream encoder. We demonstrate experimentally that the weight shared encoder can still provide a significant improvement in accuracy.

4 EXPERIMENTS

In this section, we explain the experimental setting including the datasets used, training algorithm details, etc and discuss the results.

4.1 Experimental Setup

In most datasets, the frames in a video sequence are sparsely sampled temporally to have better diversity of objects. Thus consecutive video frames are not provided for training our multi-stream algorithm. Synthetic datasets have no cost for annotation and ground truth annotation is available for all consecutive frames. Hence we made use of the synthetic autonomous driving dataset SYNTHIA (Ros et al., 2016) for our experiments. We also made use of DAVIS2017 (Pont-Tuset et al., 2017) and SegTrack V2 (Li et al., 2013) which provides consecutive frames, they are not automotive datasets but realistic.

We implemented the different proposed multi-stream architectures using Keras (Chollet et al., 2015). We used ADAM optimizer as it provided faster convergence. The maximum order (number of consecutive frames) used in the training is three (MSFCN-3) because of limitation of memory needed for training. Categorical cross-entropy is used as loss function for the optimizer. Maximum number of training epochs is set to 30 and early stopping with a patience of 10 epochs monitoring the gains is added. Mean class IoU and per-class IoU were used as accuracy metrics. All input images were resized to 224x384 because of memory requirements needed for multiple streams.

4.2 Experimental Results and Discussion

We performed four sets of experiments summarized in four tables. Qualitative results are provided in Figure 4 for KITTI, Figure 5 for DAVIS and Figure 6 for SYNTHIA. We also provide a video sequence demonstrating qualitative results for larger set of frames.

Table 1: Firstly, we wanted to evaluate different orders on multi-stream and understand the impact. We also wanted to understand the impact on high speed and medium speed scenarios. SYNTHIA dataset was used for this experiment as it had separation of various speed sequences and it was also a relatively larger dataset. Two-stream networks provided a considerable increase in accuracy compared to the baseline. MSFCN-2 provided an accuracy improvement of 8% for Highway and 14% for City sequence. RFCN-2 provided a slightly better accuracy relative to MSFCN-2. MSFCN-3 provided marginal improvement over MSFCN-2 and thus we did not explore higher orders.

Table 2: KITTI is a popular automotive dataset and

thus we used it to perform experiments on this real life automated driving dataset. We reduced our experiments to MSFCN-2 and RFCN-2 but we added shared weight versions of the same. MSFCN-2 provided an accuracy improvement of 11% and the shared weight version only lagged behind slightly.

Table 3: We repeated the experiments of the same networks used in Table 2 on a larger SYNTHIA sequence. MSFCN-2 provided an accuracy improvement of 6% in Mean IoU. MSFCN-2 with shared weights lagged by 1%. RFCN-2 versions had slightly lesser accuracy compared to its MSFCN-2 counterparts with and without weight sharing.

Table 4: As most automotive semantic segmentation datasets do not provide consecutive frames for temporal models, we tested in real non-auomotive datasets namely SegTrack and DAVIS. MSFCN-3 provided an accuracy improvement of 11% in SegTrack and 6% in DAVIS. This demonstrates that the constructed networks provide consistent improvements in various datasets.

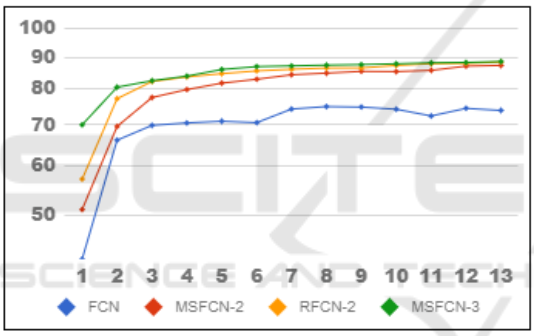


Figure 3: Accuracy over epochs for SYNTHIA dataset.

Table 4: Comparison of Multi-stream network with its baseline counterpart on DAVIS and SegTrack.

Dataset	Architecture	Mean IoU
SegTrack V2	FCN	83.82
	MSFCN-3	**94.61**
DAVIS	FCN	77.64
	MSFCN-3	**83.42**
	BVS(Märki et al., 2016)	66.52
	FCP(Perazzi et al., 2015)	63.14

We have chosen a moderately sized based encoder namely ResNet50 and we will be experimenting with various sizes like ResNet10, ResNet101, etc for future work. In general, multi-stream provides a significant improvement in accuracy for this moderately sized encoder. The improvements might be larger for smaller networks which are less accurate. With shared weights encoder, increase in computational complexity is minimal. However, it increases memory usage and memory bandwidth quite significantly due to maintenance of additional encoder feature maps. It also increases the latency of output by 33 ms for a 30 fps video sequence. From visual inspection, the improvements are seen mainly in refining the boundaries and detecting smaller regions. It is likely due to temporal aggregation of feature maps for each pixel from past frames.

MSFCN vs FCN: The single frame based FCN suffers to segment weaker classes like poles and objects at further distances. Table 3 shows IoU metrics for weaker classes like Pole, Fence and Sidewalk have significantly improved in case of multi stream networks compared to single stream FCN. Fig 4 visually demonstrates that the temporal encoder modules help in preserving the small structures and boundaries in segmentation.

MSFCN-2 vs MSFCN-3: The increase in the temporal information has clearly increased the performance of the semantic segmentation. But this brings an extra latency for real time applications.

MSFCN-2 vs RFCN: For a multi stream network the recurrent encoder feature fusion has shown quite a decent improvement compared to feature concatenation technique. It is also observed that the recurrent networks helped in preserving the boundaries of the weaker classes like poles and lane markings. However, RFCN demands more parameters and takes longer training time for convergence as shown in Fig 3.

Weight Sharing: In most of the experiments, MSFCN-2 with shared weights provided good improvement over the baseline and its performance deficit relative to the generic MSFCN-2 is usually small around 1%. However, shared weights version provide a drastic improvement in computational complexity as shown by the number of parameters in Table 2. Shared weights MSFCN-2 has a negligible increase in number of parameters and computational complexity as well whereas the generic MSFCN-2 has double the number of parameters. Thus it is important to make use of weight sharing.

5 CONCLUSIONS

In this paper, we designed and evaluated two video semantic segmentation architectures namely Recurrent FCN (RFCN) and Multi-Stream FCN (MSFCN) networks to exploit temporal information. We implemented three architectures namely RFCN-2, MSFCN-2 and MSFCN-3 using ResNet50 as base encoder and evaluated on SYNTHIA sequences. We obtain promising improvements of 9% and 15% for Highway and New York-like city scenarios over the baseline network. We also tested MSFCN-3 on real datasets like SegTrack V2 and DAVIS datasets where 11% and 6% accuracy improvement was achieved, respectively. We also explored weight sharing among

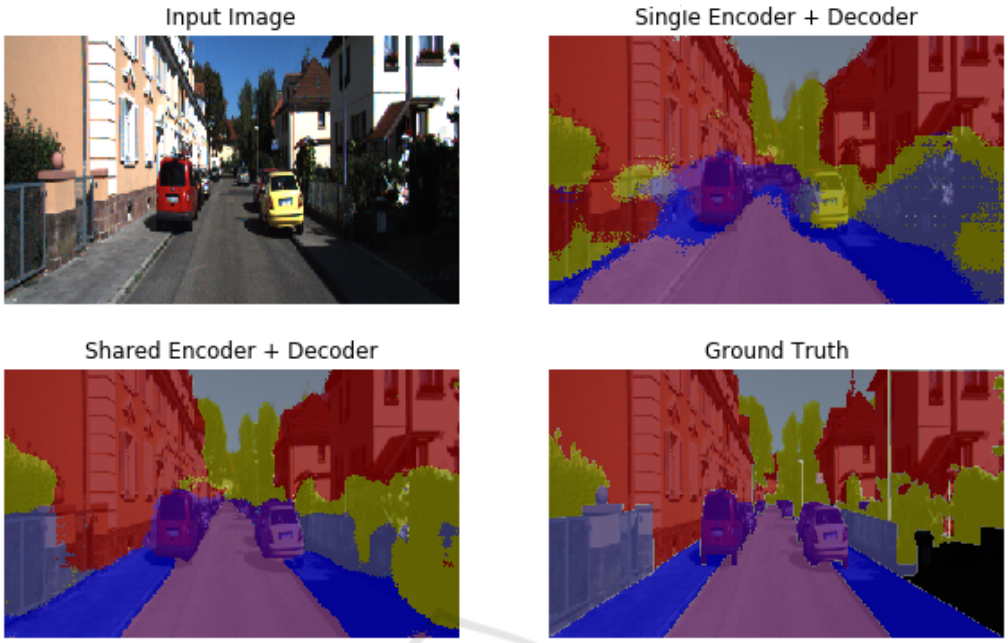


Figure 4: Results on KITTI dataset.

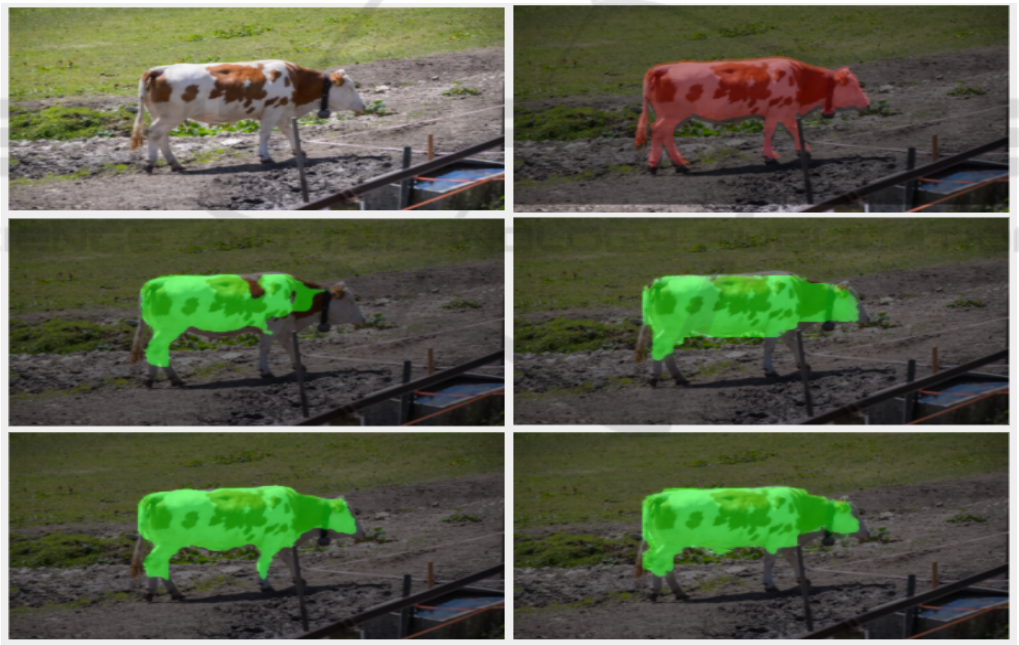


Figure 5: Results over DAVIS dataset. Left to right: RGB image, Ground Truth, Single encoder (FCN), Two stream encoder (MSFCN-2), Two stream encoder + LSTM (RFCN-2), Three stream encoder (MSFCN-3).

encoders for better efficiency and produced an improvement of 11% and 5% for KITTI and SYNTHIA using MSFCN-2 with roughly the same complexity as the baseline encoder. In future work, we plan to explore more sophisticated encoder fusion techniques.

REFERENCES

Chollet, F. et al. (2015). Keras. https://keras.io.

Heimberger, M., Horgan, J., Hughes, C., McDonald, J., and Yogamani, S. (2017). Computer vision in automated parking systems: Design, implementation and challenges. *Image and Vision Computing*, 68:88–101.

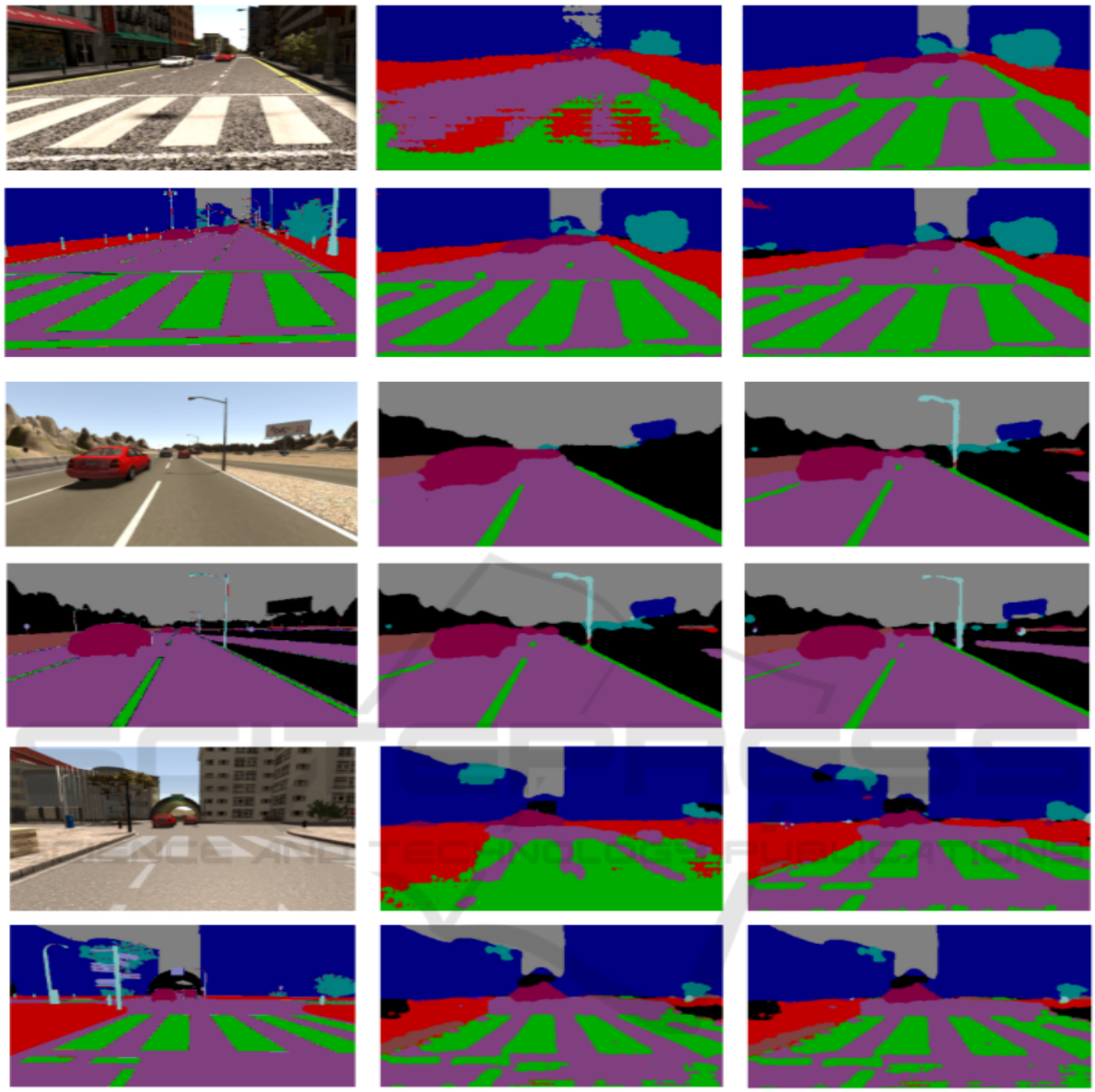


Figure 6: Qualitative results of experiments with SYNTHIA dataset. Left to right: RGB image, Single encoder (FCN), Two stream encoder (MSFCN-2), Ground Truth, Two stream encoder + LSTM (RFCN-2) and Three stream encoder (MSFCN-3).

Horgan, J., Hughes, C., McDonald, J., and Yogamani, S. (2015). Vision-based driver assistance systems: Survey, taxonomy and advances. In *Intelligent Transportation Systems (ITSC), 2015 IEEE 18th International Conference on*, pages 2032–2039. IEEE.

Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., and Brox, T. (2016). Flownet 2.0: Evolution of optical flow estimation with deep networks. *arXiv preprint arXiv:1612.01925*.

Kaiming He, Xiangyu Zhang, S. R. J. S. (2015). Deep residual learning for image recognition. *CoRR*, abs/1512.03385.

Li, F., Kim, T., Humayun, A., Tsai, D., and Rehg, J. M. (2013). Video segmentation by tracking many figure-ground segments. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2192–2199.

Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440.

Märki, N., Perazzi, F., Wang, O., and Sorkine-Hornung, A. (2016). Bilateral space video segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 743–751.

Perazzi, F., Wang, O., Gross, M., and Sorkine-Hornung, A.

(2015). Fully connected object proposals for video segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3227–3234.

Pont-Tuset, J., Perazzi, F., Caelles, S., Arbeláez, P., Sorkine-Hornung, A., and Van Gool, L. (2017). The 2017 davis challenge on video object segmentation. *arXiv:1704.00675*.

Ros, G., Sellart, L., Materzynska, J., Vazquez, D., and Lopez, A. M. (2016). The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Siam, M., Elkerdawy, S., Jagersand, M., and Yogamani, S. (2017a). Deep semantic segmentation for automated driving: Taxonomy, roadmap and challenges. In *International Conference on Intelligent Transportation Systems (ITSC)*, pages 29–36. IEEE.

Siam, M., Gamal, M., Abdel-Razek, M., Yogamani, S., and Jagersand, M. (2018). Rtseg: Real-time semantic segmentation comparative study. *arXiv preprint arXiv:1803.02758*.

Siam, M., Mahgoub, H., Zahran, M., Yogamani, S., Jagersand, M., and El-Sallab, A. (2017b). Modnet: Moving object detection network with motion and appearance for autonomous driving. *arXiv preprint arXiv:1709.04821*.

Siam, M., Valipour, S., Jagersand, M., Ray, N., and Yogamani, S. (2017c). Convolutional gated recurrent networks for video semantic segmentation in automated driving. In *International Conference on Intelligent Transportation Systems (ITSC)*, pages 29–36. IEEE.

Simonyan, K. and Zisserman, A. (2014). Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576.

Ummenhofer, B., Zhou, H., Uhrig, J., Mayer, N., Ilg, E., Dosovitskiy, A., and Brox, T. (2016). Demon: Depth and motion network for learning monocular stereo. *arXiv preprint arXiv:1612.02401*.