UNIVERSITY OF CALIFORNIA, SAN DIEGO

Using Optical Flow to Improve Semantic Video Segmentation

A thesis submitted in partial satisfaction of the
requirements for the degree
Master of Science

in

Computer Science

by

Justin Gorgen

Committee in charge:

Professor Zhuowen Tu, Chair
Professor Gary Cottrell
Professor Ravi Ramamoorthi

2017

The thesis of Justin Gorgen is approved, and it is acceptable in quality and form for publication on microfilm:

_____

_____

_____
Chair

University of California, San Diego

2017

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

VITA

2009	B. S. in Electrical Engineering and Computer Science, University of California, Berkeley
2010-2017	Engineer, SPAWAR Systems Center Pacific, San Diego, CA
2017	Master of Science in Computer Science, University of California, San Diego

PUBLICATIONS

Gorgen, J., Lemay, L., and Gebre-Egziabher, D. (2012). "Precise output error characterization for triangulation of visual landmarks from two views with noisy camera pose". *Institute of Navigation International Technical Meeting 2012* 2, 1014-1088

L. Lemay, T. Denewiler, J. Gorgen, B. Fitzsimmons and D. Gebre-Egziabher, "Comparison of Visual Odometry Navigation Algorithms for Ground Robotic Applications," *Proceedings of the ION Joint Navigation Conference*. Colorado Springs, CO. June 2012

L. Lemay, J. Gorgen, C. C. Chu and D. Gebre-Egziabher, "Compensating for Colored Measurement Noise from Vision Based Navigation Sensors in a Loosely Coupled Extended Kalman Filter," *Proceedings of the ION Joint Navigation Conference*. Colorado Springs, CO. June 2012

Gorgen, J. Lemay, L, "Sferics for Time Synchronization and Navigation", *Proceedings of the ION Joint Navigation Conference*, Orlando, FL. June 2015

ABSTRACT OF THE THESIS

Using Optical Flow to Improve Semantic Video Segmentation

by

Justin Gorgen

Master of Science in Computer Science

University of California San Diego, 2017

Professor Zhuowen Tu, Chair

This thesis presents a deep neural network model that augments an existing semantic image segmentation model with optical flow data to improve segmentation performance on video sequences. Three network topologies combining optical flow data layers with RGB data layers are compared. The best performing model, FlowSeg-A, achieves an average per-class accuracy of 72.696% on the SegNet test set. This is an improvement of 4.8 percentage-points versus SegNet, the RGB-only segmentation model on which FlowSeg-A is based. The main accuracy improvements come from the classes SignSymbol (15.4% improvement), Bicyclist(10.2%), and Pole (9.0%). These

accuracy improvements are achieved with only 1,152 (0.004%) more parameters, and FlowSeg-A achieves this performance using the same training set and training schedule as the SegNet algorithm.

1 Introduction and Background

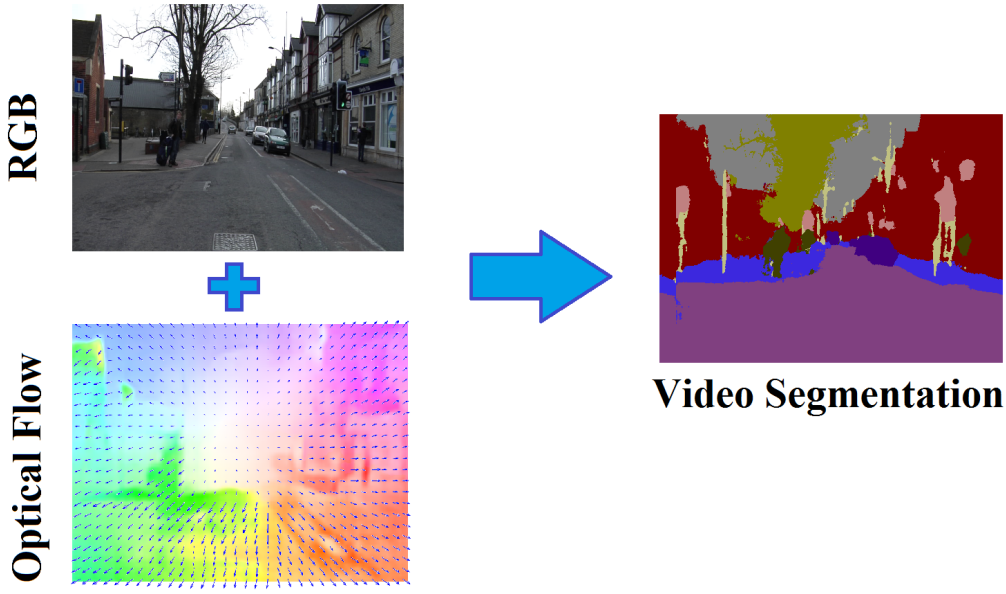

Figure 1.1: **Combining optical RGB and optical flow data** improves semantic segmentation for video sequences.

Semantic video segmentation is the art of assigning categorical labels to each individual pixel in an video. It has applications in video editing, scene understanding, and autonomous navigation for unmanned vehicles [17, 20]. While much of the research in semantic segmentation has focused on segmenting still images, most image-

segmentation algorithms can be applied to video by simply treating each frame of the video as an independent image. However, the individual frames of video are not independent, and the relative motion of objects in the video can be calculated on a pixel-by-pixel basis. This relative motion is called a motion field, or optical flow, and the algorithm presented in this thesis uses optical flow to augment semantic segmentation for video, as shown in Figure 1.1.

1.1 Optical Flow Estimation

Optical flow is an estimation of the motion field in video that assumes luminance is conserved in video sequences. Thus optical flow is calculated by finding the translation vector that "explains" the mean change in luminance over a specified area [18, 9]. In this thesis, a pre-trained deep neural network, FlowNetC [7], is used to estimate the optical flow for the frames. As shown in Figure 1.2, FlowNetC provides an estimate of optical flow with sharper boundaries than the Lucas-Kanade method[18] was able to achieve. FlowNetC is a deep neural network based on convolutional neural networks, and uses a novel cross-correlation layer in order to calculate the displacement between features in a sequence of two images. FlowNetC is trained on a synthetic dataset for which the ground-truth motion field is known, and its estimations of optical flow handle large displacements and require less tuning for individual datasets than hand-coded iterative methods like Lucas-Kanade.

As shown in Figure 1.2, optical flow fields calculated from image sequences

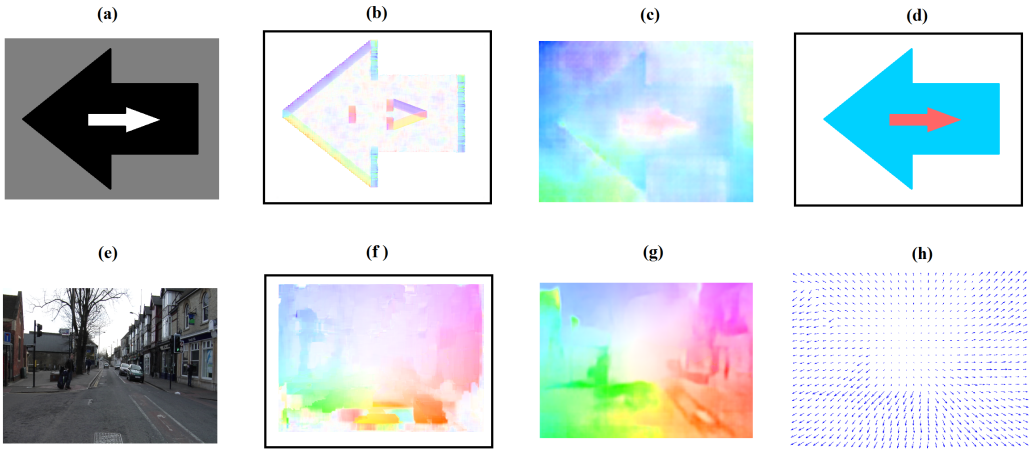

Figure 1.2: **A comparison of optical flow algorithms.** (A) In this synthetic greyscale image of two arrows, the black arrow will translate 5 pixels to the left, and the white arrow will translate 3 units to the right. (B) Lucas-Kanade Optical Flow calculated with a window-size of 15 (C) Optical Flow calculated using FlowNet (D) Ground truth from motion flow (E) An image from the CamVid dataset (F)Optical Flow calculated using iterative Lucas-Kanade [3]. (G)Optical Flow calculated using FlowNet (H) A vector representation of the same flow field in G. Note that both algorithms struggle with the textureless synthetic image, but FlowNetC manages to produce a more accurate region of flow, with recognizable features

provide information about the edges of objects. In this thesis, the segmentation data from optical flow will be exploited to improve the video segmentation performance of an existing segmentation algorithm, SegNet, that is based only on still images.

1.2 Related Research

Video segmentation has been tackled by researchers from both video editing backgrounds and computer vision backgrounds. The video editing approach typically focuses on identifying foreground and background objects [31, 10, 28]. Meanwhile

computer vision researchers have long applied image segmentation techniques to sequences of images that happen to be from video [2, 23], but some recent research [25, 16] has explicitly approached semantic video segmentation as its own topic with unique challenges and constraints.

1.2.1 Video Segmentation with Optical Flow

Many papers have examined the use of optical flow for video segmentation [22, 32, 28], with the main focus on developing algorithms for motion classification [19] or methods that separate foreground and background areas to facilitate later analysis on the foreground areas [31, 27]. These efforts are fascinating in that very accurate segmentation results can be achieved without learning class-specific segmentation features. However, these efforts differ from semantic segmentation in that the segments are not given class labels, they are treated as moving layers for the purpose of assigning depth to the layers or segregating objects based on movement.

1.2.2 Semantic Image Segmentation

Semantic image segmentation, as illustrated by Figure 1.3, assigns class labels to each individual picture in an image. As computers have become more accurate than humans at classifying the main subject of an image [13], the natural extension to classifying single subjects in images is to classify each object in an image and identify its location in the image [23]. For applications such as caption generation and target

tracking, it is often sufficient to identify bounding boxes around each object in an image [20, 23]. However, for other applications, like lane detection for autonomous vehicles[4], it is useful to know exactly which pixels correspond to which semantic class. This has given rise to recent research in the area of semantic image segmentation using deep neural networks.

Figure 1.3: **An example of semantic segmentation.** Semantic segmentation is the art of assigning labels to individual pixels.

The use of neural networks for semantic still-image segmentation is an active area of research. Since Karen Simonyan and Andrew Zisserman's publication of VGG in 2014[26], the standard approach to semantic image segmentation has been to take a convolutional neural network designed for image classification and devise a method for determining which parts of the original image strongly activate the various image classes. Initial approaches involved masking parts of the original image to observe the affect on image classification, and further improvements were made by developing manually-constructed upsampling networks or integrating multi-resolution featuremaps from the downsampling layers to estimate segmentation [24, 8, 17]. Ad-

ditional efforts have re-purposed the VGG16 classification network for class-agnostic edge detection and segmentation [30, 27], which can serve as general segmentation for higher-level classifiers. Recent approaches directly estimate semantic segmentation with an end-to-end training approach [2, 21, 29] ,including the 2015 paper "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation" from Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. These recent efforts use a fully convolutional encoder-decoder approach and differ from earlier approaches by learning the weights for the deconvolution layers. The fully convolutional encoder-decoder approach has the advantage of having the capacity to learn upsampling and deconvolutional filters that are directly optimized for the encoding filters, at the expense of having roughly twice the number of parameters as the convolutional classifier network it is based on. While new approaches have implemented fully connected conditional random field (CRF) layers to refine the edge details of the output segmentation [5], CRF layers are complex with feature depths on the order of the square of the number of classes. Furthermore, performance from fully convolutional meets or surpasses CRF networks at semantic segmentation[29]. The fully convolutional approach retains the advantages of CNNs, (e.g. translation and scale invariance) and gains speed in training and inference from the lack of fully-connected layers. The fully-convolutional network presented in this thesis is directly derived from the SegNet encoder-decoder network presented in [2].

1.2.3 Semantic Video Segmentation

Deep learning approaches to semantic video segmentation typically involve applying a semantic image segmentation to each image independently. However, there has recently been some research specifically into semantic video segmentation using neural networks. Of note, the most closely related research to this thesis are the 2016 CVPR paper "Feature Space Optimization for Semantic Video Segmentation" by Abhijit Kundu, Vibhav Vineet, and Vladlen Koltun [16], and the 2016 *CoRR* paper "Optical Flow with Semantic Segmentation and Localized Layers" by Laura Sevilla-Lara, Deqing Sun, Varun Jampani, Michael J. Black [25]. Kundu's paper [16] uses optical flow indirectly to enforce temporal consistency between semantic segmentation in different video frames after an initial segmentation refinement performed by a fully-connected Conditional Random Field (CRF) layer. Meanwhile, while [25] combines segmentation from RGB data with separately-calculated segmentation from flow in an iterative approach that improves both optical flow and and segmentation borders. The approach presented in this thesis differs from [16] and [25] by using optical flow directly as inputs to a convolutional model, and uses a fully-convolutional model throughout the network without CRF layers.

2 Methods

To evaluate the effects of optical flow data on semantic video segmentation, several network topologies are compared. These different topologies combine optical flow features with RGB features at different layers of the network. The neural network models presented in this thesis are derived from the SegNet model [2]. The SegNet model is chosen because it is fully convolutional, performs well when trained from a small dataset, and fits in the 12GB of memory on an nVidia K40c GPU.

2.1 Models

As shown in Figure 2.1, three methods of integrating per-pixel optical flow features are examined, here described as FlowSeg Models A through C. For ease of comparison to the SegNet results, each of these models follows the SegNet's pattern of having the 13 convolutional layers of FlowSeg-A consists of concatenating the two optical flow channels du, dv onto the data layer of the SegNet model. Thus, FlowSeg-A differs from the SegNet model only in the construction of the first convolutional layer,

which now calculates a 64 convolutional features on a 5-channel, $3 \times 3 \times 5$ receptive field instead of $3 \times 3 \times 3$. FlowSeg-B calculates two layer of convolutional features on an optical flow image independently of the RGB data, and then concatenates this data with the RGB convolutions at a new, third convolutional layer before the first pooling layer. Finally, FlowSeg-C concatenates optical flow features with the output of the SegNet network, a construction which allows FlowSeg-C to share most of its weights with a pre-trained SegNet network. In essence, FlowSeg-C is tacked-on and fine-tuned after the segmentation on RGB is already trained. This makes FlowSeg-C a construction that can be adapted easily to any semantic segmentation. The kernel size of each convolutional layer is shown in Table 2.1. The number of trainable parameters for each of the networks is summarized in Table 2.2.

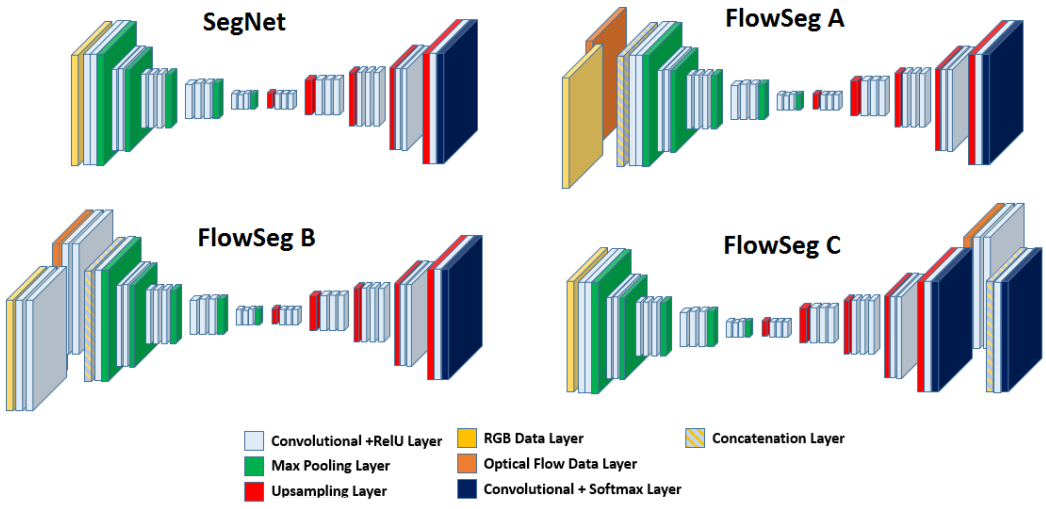

Figure 2.1: **FlowSeg models compared to the SegNet model.** FlowSeg-A and FlowSeg-B combine optical flow features with the input image features, while FlowSeg-C combines optical flow features with the output image labels to refine the labeling.

Table 2.1: **Convolutional kernel sizes** for the SegNet and FlowSeg networks. Columns S, A, B, and C indicate the presence of a layer in the architectures SegNet, FlowSeg-A, FlowSeg-B, and FlowSeg-C, respectively. Here i, j are kernel height and width, k is the number of output channels, and l is the number of input channels.

Layer Name	i	j	k	l	S	A	B	C
conv1_1_5	3	3	64	5		X		
conv1_1	3	3	64	3	X		X	X
conv1_2	3	3	64	64	X	X	X	X
conv1_1_flow	3	3	64	2			X	X
conv1_2_flow	3	3	64	64			X	X
conv1_3	3	3	64	128			X	
conv2_1	3	3	128	64	X	X	X	X
conv2_2	3	3	128	128	X	X	X	X
conv3_1	3	3	256	128	X	X	X	X
conv3_2	3	3	256	256	X	X	X	X
conv3_3	3	3	256	256	X	X	X	X
conv4_1	3	3	512	256	X	X	X	X
conv4_2	3	3	512	512	X	X	X	X
conv4_3	3	3	512	512	X	X	X	X
conv5_1	3	3	512	512	X	X	X	X
conv5_2	3	3	512	512	X	X	X	X
conv5_3	3	3	512	512	X	X	X	X
conv5_3_D	3	3	512	512	X	X	X	X
conv5_2_D	3	3	512	512	X	X	X	X
conv5_1_D	3	3	512	512	X	X	X	X
conv4_3_D	3	3	512	512	X	X	X	X
conv4_2_D	3	3	512	512	X	X	X	X
conv4_1_D	3	3	512	512	X	X	X	X
conv3_3_D	3	3	256	512	X	X	X	X
conv3_2_D	3	3	256	256	X	X	X	X
conv3_1_D	3	3	256	256	X	X	X	X
conv2_2_D	3	3	128	256	X	X	X	X
conv2_1_D	3	3	128	128	X	X	X	X
label_softmax	3	3	11	128	X	X	X	X
conv1D_with_flow	3	3	64	75				X
flowsegc_softmax	1	1	11	64				X

Table 2.2: **Network parameter counts** for SegNet and the three FlowSeg models

Network	Number of parameters
SegNet	31,705,547
FlowSeg-A	31,706,699
FlowSeg-B	31,891,595
FlowSeg-C	31,793,558

All networks are implemented in Caffe [15]. The design of the FlowSeg and SegNet models is straightforward, with standard convolutional layers with stride 1 and pooling layers with stride 2. However, the implementation in Caffe has additional layers to aid training. In Caffe, a batch-normalization layer is placed after each of the convolutional layers, which recursively estimates a mean and variance value for each of the input channels during training of the network [14]. The mean and variance are set to 0 and 1, respectively, through a scale-and-shift transformation. Thus, each $n \times n \times c$ convolutional layer has an additional $2c$ parameters learned from the data that represent a mean and variance for each channel. This batch-normalization whitens the input data for each layer during training. The end result of the batch-normalization during training allows training speed to be improved a with a learning rate of 0.001 during stochastic gradient descent [2]. For deployment of the network for inference, the average scale-and-shift operation learned from each layer during the entire training set is applied as a constant transformation on each layer in the model.

2.2 Weight Matrix Initialization and Surgery

Following the procedure from SegNet [2], transfer learning is used to initialize the 13 convolutional layers borrowed from VGG16. The weight matrices for the SegNet and FlowSeg networks are initialized, where possible, with the weights from VGG16 trained on the ImageNet dataset. The remaining weights for the deconvolution layers and classification layers are initialized with Xavier initialization [12]. For Xavier initialization, a layer's weights w_{ij} corresponding to the n_i outputs and n_j inputs to the layer are sampled from a normal distribution with variance:

$$Var\left[w_{ij}\right] = \frac{2}{n_i + n_j} \tag{2.1}$$

For convolutional layers, with n_k kernels, n_l input channels, and kernel dimensions of (n_i, n_j), the Xavier initialization from (2.1) becomes:

$$Var\left[w_{ijkl}\right] = \frac{2}{n_i n_j n_l + n_i n_j n_k} \tag{2.2}$$

Because the input depth of some convolutional layers are changed from VGG16 in the FlowSeg architectures, some surgery is required to correctly initialize the weights. For FlowSeg-A, the first convolutional layer has a different size kernel than the corresponding layer in VGG16. That is, FlowSeg-A has 5 input channels versus the 3 input channels of VGG16. For each of the 64 kernels in the first convolutional layer, the first 3 layers of weights are copied from VGG16, with the remaining 2 layers initialized

with Xavier initialization.

For FlowSeg-B, a third convolutional layer is added before the first pooling step, and has an input depth of 128, with 64 output channels to maintain compatibility with the higher layers of the VGG16 network. In this layer, the first 64 channels of the input correspond to features from the second convolutional layer operating on the RGB input data. The second set of 64 input channels correspond to features from the second convolutional layer operating on the optical flow data. The weights of the the two convolutional layers above the optical flow data are initialized with Xavier initialization. The kernel weights of the third layer are initialized with a modified form of Xavier initialization:

$$w_{kl} = w_{kl}^{Xavier} + \delta_{kl} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{2.3}$$

Here, the term δ_{kl} is the Kronecker delta function that is 1 if and only if the output kernel index k is equal to the input kernel index l, and 0 otherwise. The term w_{kl}^{Xavier} is the 3x3 array of weights created by Xavier initialization with variance from (2.2). This biases the initial kernels towards using the features learned from the RGB data. This is necessary to improve the segmentation performance of FlowSeg-B because the higher layers of the FlowSeg-B network are initialized with VGG16's ImageNet features. Without this weight initialization, classification accuracy is 3-7% lower per class.

For FlowSeg-C, no such surgery is required because all new layers are added above the VGG16 layers and therefore do not change the size or shape of the VGG16 layers. All weights for the deconvolutional and classification layers are initialized with Xavier initialization.

2.3 Data Set

The data set used in this thesis is the CamVid dataset [4], which consists of 18,202 images taken from 5 separate videos of a vehicle driving on streets in Cambridge, UK. The video is taken 29.97Hz, with a resolution of 960x720 recorded on a . Of the 18,202 video frames, 701 images are accompanied by semantic segmentation ground-truth images that are hand-labeled with 32 classes. The labeled frames are approximately 1 second (30 frames at 29.97 fps) apart.

2.3.1 Data Set Modifications

To enable direct comparison with the published SegNet results, the image and ground truth labels are downsampled to a resolution of 480x360. Additionally, the number of classes are reduced from the 32 original classes to 11, with the excluded classes are relabeled with a 12th class label representing "void." The 11 classes and their proportion of the training sets are shown in Table 2.3. The images are split into the same test and train sets used by SegNet.

Table 2.3: **Data set classes and proportions.** The 11 classes and the proportion of pixels belonging to each class in the training set and test set

Class	Training	Test
Sky	0.1695	0.1701
Building	0.2409	0.2475
Pole	0.0096	0.0190
Road	0.3104	0.2639
Pavement	0.0486	0.1036
Tree	0.1150	0.1200
SignSymbol	0.0046	0.0139
Fence	0.0239	0.0121
Car	0.0645	0.0381
Pedestrian	0.0070	0.0104
Bicyclist	0.0060	0.0015

2.3.2 Optical Flow Calculation from the Data Set

Optical flow is calculated for each of the 701 frames of the CamVid dataset that has an accompanying ground-truth segmentation label. Two sets of optical flow images were calculated, nominally at 1Hz and one at 30Hz. The 1Hz flow calculations used For the 1Hz flow, optical flow for the n-th frame is estimated by using the n-th and $n+30$-th frames of the dataset as inputs to the optical flow estimator FlowNetC. The optical flow for the last frame of each video sequence is calculated using the n and $n-30$-th frames. Similarly, the optical flow for the 30Hz run is calculated using the n and $n+1$-th frames.

2.4 Training

Training is executed on an nVidia Tesla K40 GPU. Training is performed using stochastic gradient descent (SGD), with a learning rate of 0.001, and momentum

of 0.9. Training is performed for 40,000 epochs, with images shuffled and mirrored at random. Because the networks have similar depths and numbers of learn-able parameters, training takes 33 hours for FlowSeg-A, FlowSeg-B, and FlowSeg-C.

3 Results

The results of the FlowSeg architecture are reported in Table 3.1. The FlowSeg-A and FlowSeg-C architectures are able to improve upon the results of SegNet, with a minimal increase in the number of weights.

Several metrics are used in the literature to describe the accuracy of segmentation results. Among these are accuracy, intersect-over-union, and cover.

Inference accuracy, as used by Badrinarayanan, Kendall, and Cipolla in the SegNet paper [2], is defined as:

$$accuracy_i = \frac{c_{ii}}{\sum_j c_{ij}} \tag{3.1}$$

Here, c_{ij} is count of pixels with ground-truth class label i and inferred label j. Average accuracy is thus defined as the arithmetic mean of accuracy over all classes, without weighting for the frequency of each class. The SegNet paper also uses intersect-over-union (IoU), which is defined as the count of correctly labeled pixels for a class divided by the combined number of ground-truth pixels for that class and inferred pixels for that class. This metric is used to punish algorithms that allow class

labels to grow outside the true segmentation boundaries.

$$IoU_i = \frac{c_{ii}}{\sum_j c_{ij} + \sum_j c_{ji} - c_{ii}} \tag{3.2}$$

An additional metric for scoring segmentation algorithms, closely related to intersect-over-union, is cover [1]. Cover, also called weighted intersect-over-union, is calculated using the relative number of pixel counts for each class. This summarizes the segmentation performance of all classes into one value, while accounting for the relative frequency of each class.

$$cover = \frac{1}{\sum_{i'j'} c_{i'j'}} \sum_i \left(IoU_i \sum_j c_{ij} \right) \tag{3.3}$$

As shown in Table 3.1, all three FlowSeg models are able to improve on the average accuracy of the SegNet model. However, it should be noted that while there is no clear difference in performance between FlowSeg-A and FlowSeg-C. For FlowSeg-A, this improvement is gained with only 1,152 extra parameters, or a 0.004% increase. This increase in parameter count comes solely from the extra input parameters required to increase the kernel size from 3x3x3 to 3x3x5 for each of SegNet's 64 convolution kernels in the first layer. The greatest improvement is on the segmentation of Bicyclists, which had a 33.94% accuracy under SegNet and a 47.59% accuracy with FlowSeg-A. This represents an improvement by a factor of 40%; however, this improvement has limited effect on the overall accuracy because only a very small portion (0.15%) of total pixels that are classified as Bicyclist, as shown in Table 2.3. Results

Table 3.1: **Results on the CamVid Dataset.** Results in **bold** are improvements over the SegNet baseline. Results that are underlined are the top score for that category.

	SegNet w/VGG	FlowSegA 1Hz	FlowSegA 30hz	FlowSegB 30hz	FlowSegC 30hz
IoU	0.551	**0.577**	**0.583**	**0.575**	**0.576**
Weighted IoU	0.778	**0.784**	**0.792**	**0.785**	**0.784**
Average Accuracy	0.679	**0.728**	**0.728**	**0.720**	**0.727**
Weighted Acc.	0.868	**0.873**	**0.879**	**0.874**	**0.874**
Per Class Accuracy					
Sky	0.930	**0.932**	0.926	**0.936**	0.928
Building	0.853	0.842	**0.854**	0.849	0.847
Pole	0.402	**0.498**	**0.491**	**0.492**	**0.475**
Road	0.939	0.937	**0.945**	0.939	0.933
Pavement	0.880	**0.887**	**0.889**	0.871	**0.915**
Tree	0.821	**0.838**	**0.846**	**0.838**	**0.826**
SignSymbol	0.410	**0.505**	**0.564**	**0.553**	**0.603**
Fence	0.407	**0.522**	**0.478**	**0.475**	**0.443**
Car	0.772	**0.810**	**0.817**	**0.794**	**0.815**
Pedestrian	0.719	**0.756**	**0.757**	**0.781**	**0.753**
Bicyclist	0.339	**0.476**	**0.442**	**0.396**	**0.459**

from selected images in the test set are shown in Figure 3.1.

Details on improvements in segmentation results around class examples of a Bicyclist, a Car, and a Pole, are shown in Figure 3.2. This example shows that both SegNet and FlowSeg-A tend to over-cover small, skinny segments. FlowSeg-A, however does a better job of keeping skinny segments contiguous. This characteristic is what allows the FlowSeg-A network to improve on the segmentation scores for Bicyclist, Pole, SignSymbol, Fence, and Pedestrian. Learned convolutional filters and the filter responses to optical flow on an image from the test set are shown in Figures 3.3 - 3.8. The filters in FlowSeg-A largely respond to edges in the image, while filters from

FlowSeg-B and C respond to areas of contiguous optical flow.

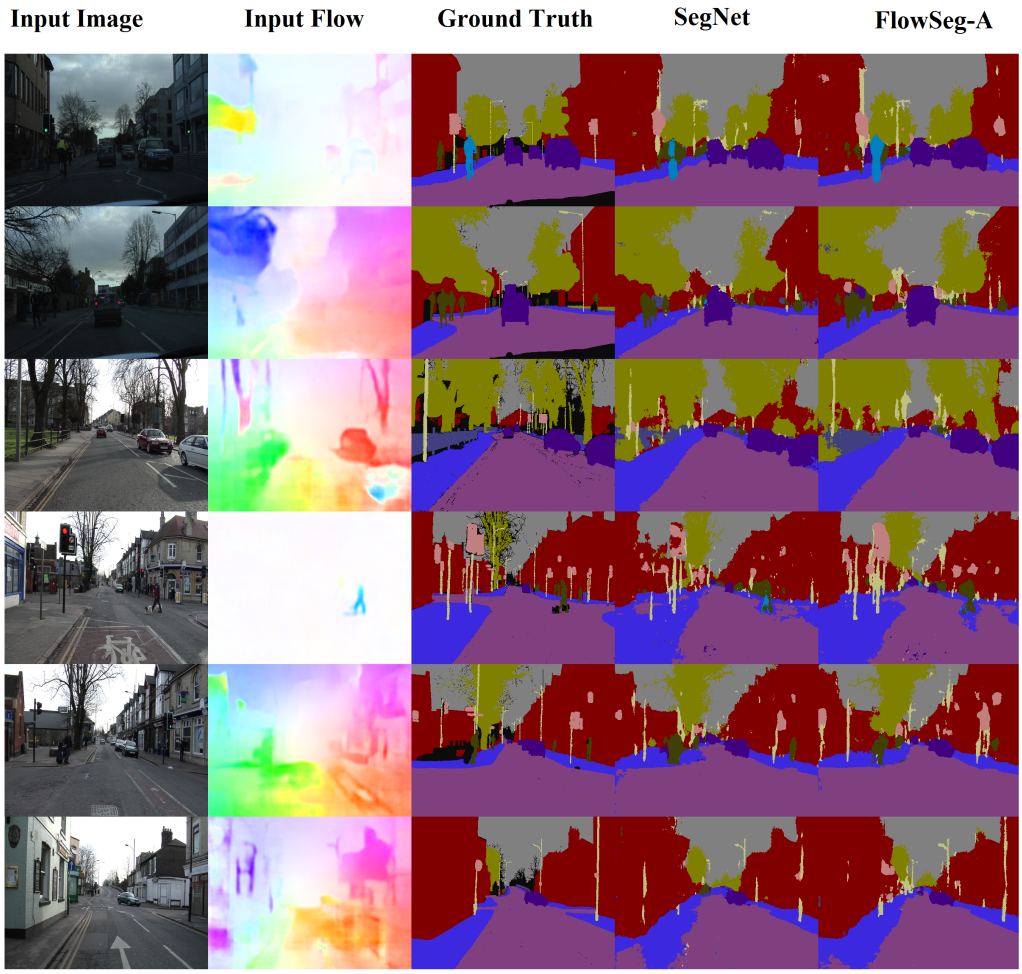

Figure 3.1: **A comparison of SegNet and FlowSeg-A results.** Areas of sky, road, and pavement are well covered by both networks, but FlowSeg-A improves cover of skinny,narrow objects like poles and bicyclists.

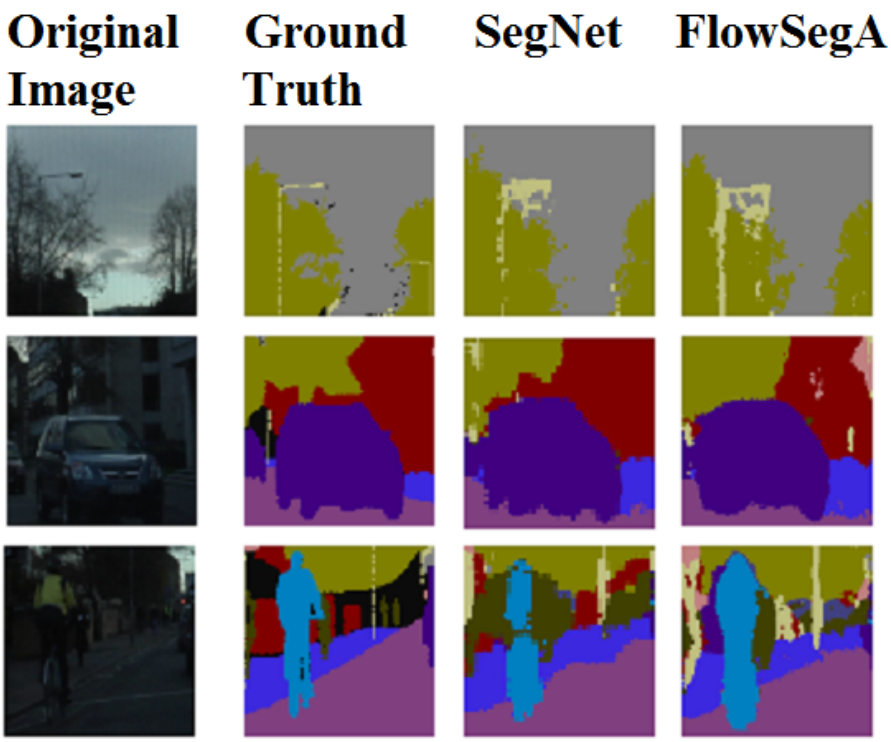

Figure 3.2: **A close-up of segmentation results** from SegNet and FlowSeg, as compared to ground truth. FlowSeg's use of optical flow allows it to keep skinny segments continuous, as illustrated by the pole and bicyclist examples in this figure.

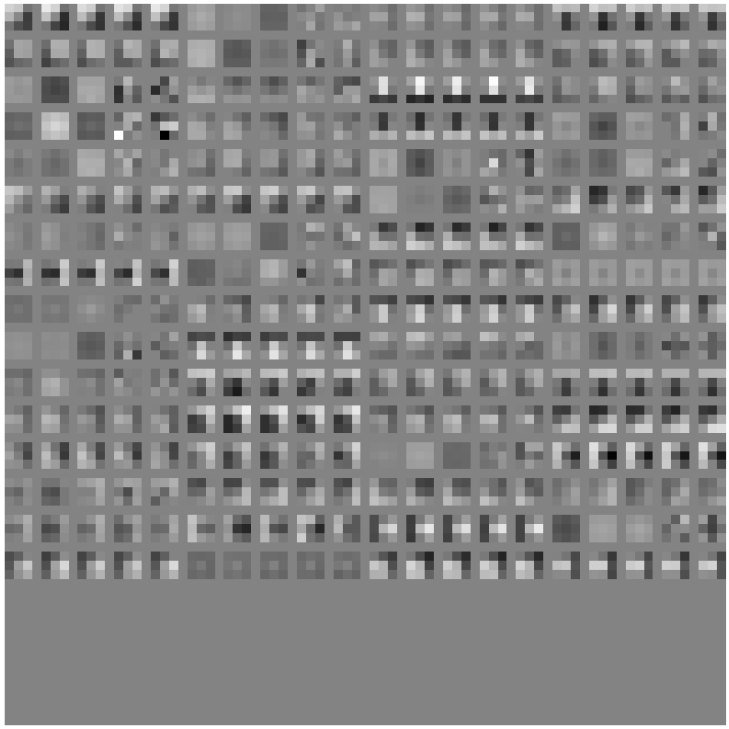

Figure 3.3: **FlowSeg-A kernels.** The 320 convolutional kernels of the first layer of FlowSeg-A. These correspond to the r,g,b, du, and dv channels.

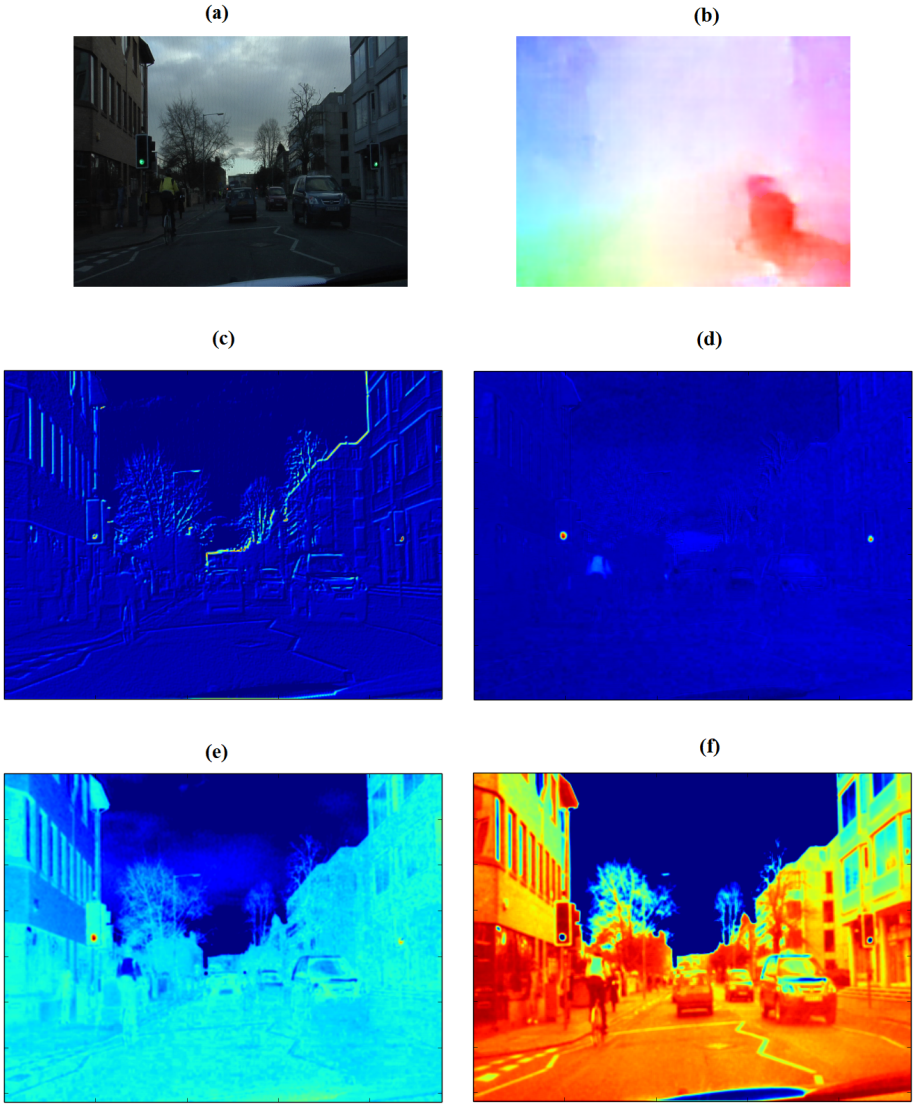

Figure 3.4: **Example FlowSeg-A kernel responses** for the first layer kernels with the highest energy response on the combined optical flow and RGB data. (A) The input image (B) The input optical flow calculated from 30Hz video (C) Response of the 0th filter (D) Response of the 12th filter (E) Response of the 36th filter (F) Response of the 61st filter

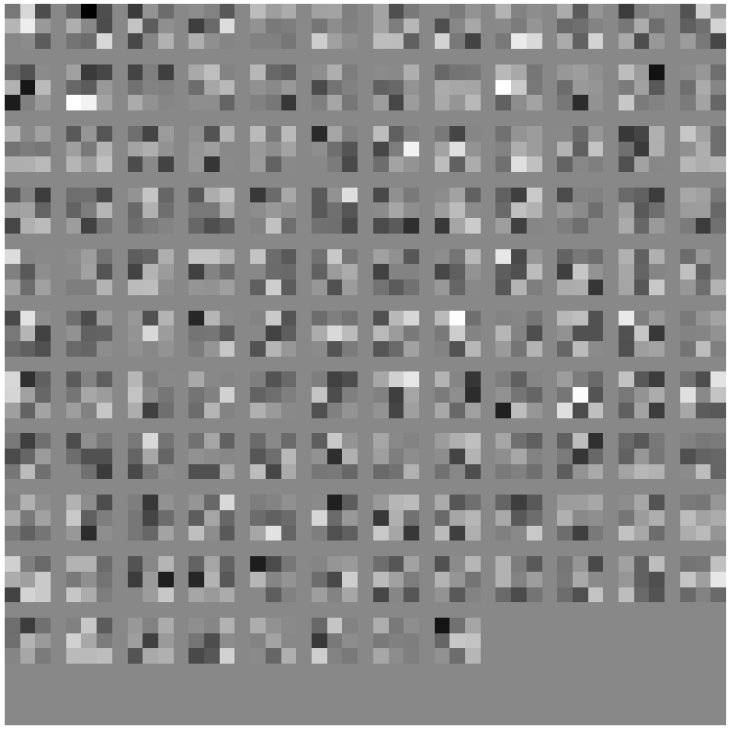

Figure 3.5: **FlowSeg-B kernels.** The 128 convolutional kernels of the first layer of FlowSeg-B that takes optical flow as input. These correspond to the du,dv channels.

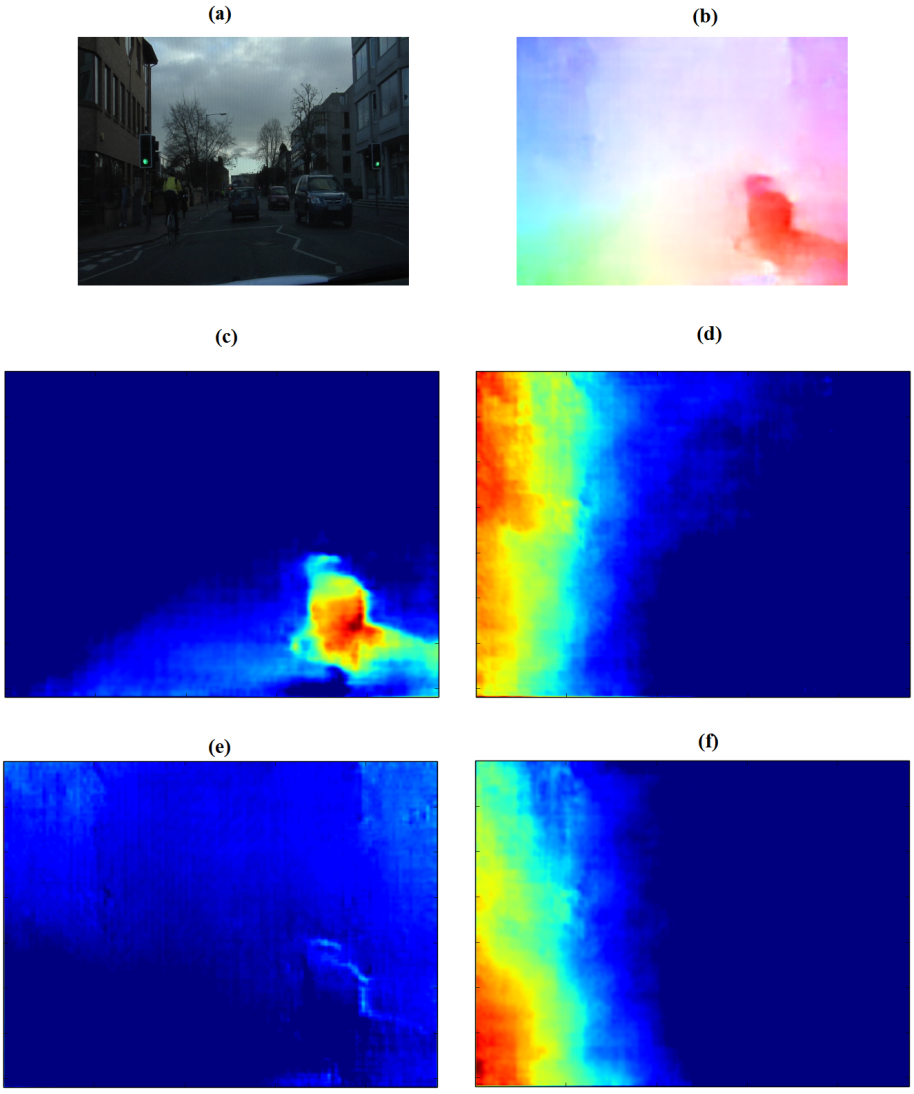

Figure 3.6: **Example FlowSeg-B kernel responses** for the first layer kernels with the highest energy response on the input optical flow. (A) The input image (B) The input optical flow calculated from 30Hz video (C) Response of the 2nd filter (D) Response of the 5th filter (E) Response of the 34th filter (F) Response of the 56th filter

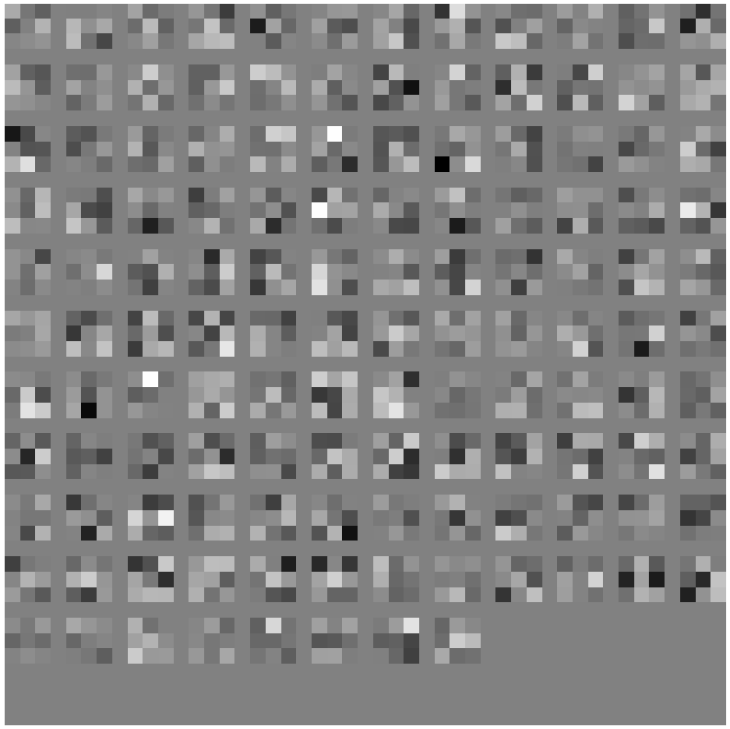

Figure 3.7: **FlowSeg-C kernels.** The 128 convolutional kernels of the first layer of FlowSeg-C that takes optical flow as input. These correspond to the du,dv channels.

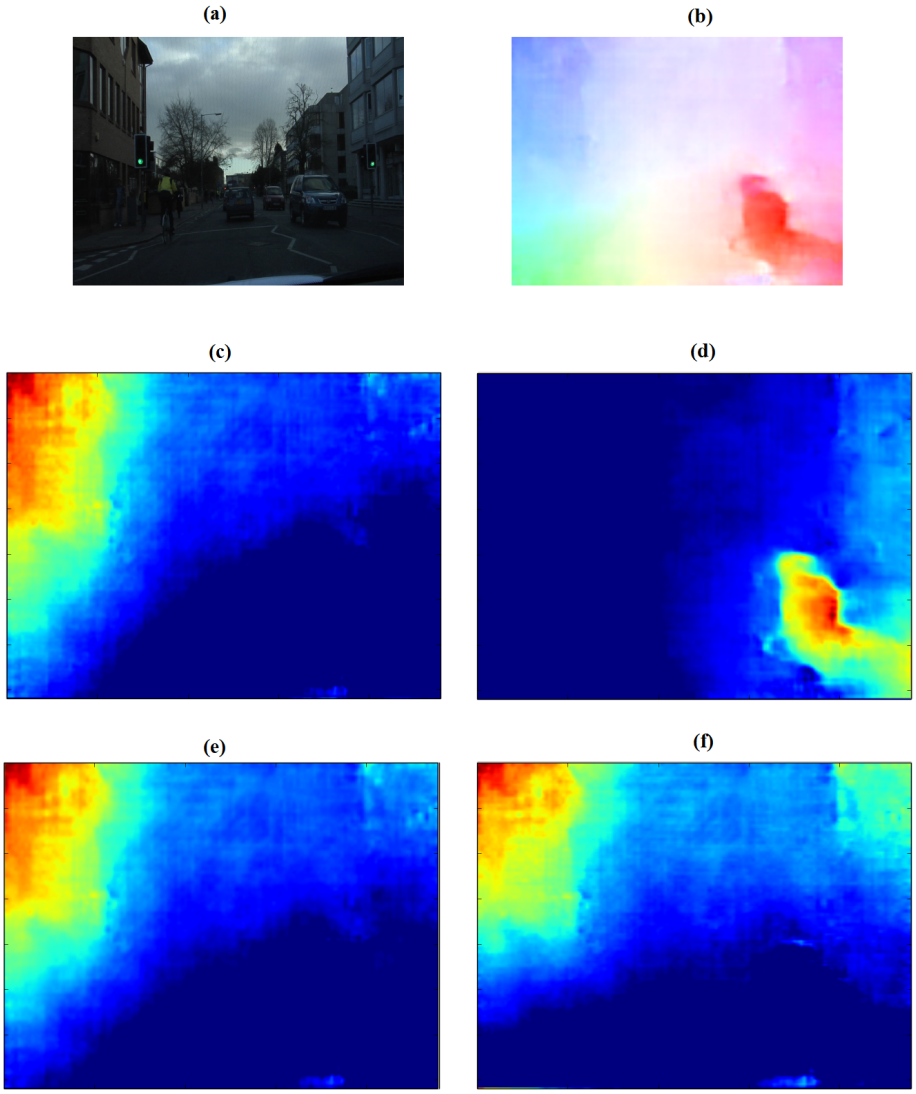

Figure 3.8: **Example FlowSeg-C kernel responses** for the first layer kernels with the highest energy response on the input optical flow. (A) The input image (B) The input optical flow calculated from 30Hz video (C) Response of the 5th filter (D) Response of the 39th filter (E) Response of the 42nd filter (F) Response of the 52nd filter

4 Conclusion

It is clear that video segmentation can be improved by the addition of optical flow. With the best performing model, FlowSeg-A, for only a 0.0005% increase in parameter count, accuracy and cover are improved by 2%. Additionally, FlowSeg-A, achieves an average-per-class accuracy of 72.696%. This is an improvement of 4.8 percentage-points and 7.1% proportionally versus SegNet, the RGB-only segmentation model on which all the FlowSeg models are based. The main accuracy improvements come from the classes SignSymbol (15.4% improvement), Bicyclist(10.2%), and Pole (9.0%). These improvements were earned using the same training set and training schedule as the SegNet algorithm, and therefore demonstrate that the improvement in segmentation performance can be attributed to optical flow.

Nevertheless, there is still room for improvement in the performance of semantic segmentation for video. Newer approaches such as conditional random fields (CRF) and residual networks (ResNet) [29] have shown promising results on still image segmentation. Following the methods in this thesis, optical-flow can be added on to CRF or ResNet networks to improve segmentation. One of the caveats of using

optical-flow for image segmentation is the limited availability of data-sets from video with both ground-truth semantic segmentation and dense enough video frames to calculate optical flow. However, recently released datasets such as VirtualKITTI [11] and Cityscapes [6] have both video sequences and ground-truth segmentation. Theses data-sets create opportunities for further research in applying the techniques of this thesis.

As shown in this thesis, one of the benefits of using optical flow to augment semantic segmentation is that optical flow features can be added on to existing segmentation networks with a very small increase in the number of network parameters. Due to the common practice of having larger feature depths in higher layers of convolutional neural networks, adding channels at the bottom of a network as in FlowSeg-A is the most efficient way of adding optical flow to a semantic segmentation network. Therefore, this thesis demonstrates that adding a few parameters to a neural to process optical flow can create a significant improvement in segmentation with virtually no impact on computational requirements.

Bibliography

[1] Arbelaez, P., Maire, M., Fowlkes, C., and Malik, J. Contour detection and hierarchical image segmentation. *IEEE transactions on pattern analysis and machine intelligence 33*, 5 (2011), 898–916.

[2] Badrinarayanan, V., Kendall, A., and Cipolla, R. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *CoRR abs/1511.00561* (2015).

[3] Bouguet, J.-Y. Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm.

[4] Brostow, G. J., Shotton, J., Fauqueur, J., and Cipolla, R. Segmentation and recognition using structure from motion point clouds. In *ECCV (1)* (2008), pp. 44–57.

[5] Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv preprint arXiv:1606.00915* (2016).

[6] Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., and Schiele, B. The cityscapes dataset for semantic urban scene understanding. *CoRR abs/1604.01685* (2016).

[7] Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., van der Smagt, P., Cremers, D., and Brox, T. Flownet: Learning optical flow with convolutional networks. 2758–2766.

[8] Farabet, C., Couprie, C., Najman, L., and LeCun, Y. Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence 35*, 8 (2013), 1915–1929.

[9] Fleet, D., and Weiss, Y. Optical flow estimation. In *Handbook of mathematical models in computer vision*. Springer, 2006, pp. 237–257.

[10] Fradi, H., and Dugelay, J.-L. Robust foreground segmentation using improved gaussian mixture model and optical flow. In *Informatics, Electronics & Vision (ICIEV), 2012 International Conference on* (2012), IEEE, pp. 248–253.

[11] Gaidon, A., Wang, Q., Cabon, Y., and Vig, E. Virtual worlds as proxy for multi-object tracking analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 4340–4349.

[12] Glorot, X., and Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *Aistats* (2010), vol. 9, pp. 249–256.

[13] He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 770–778.

[14] Ioffe, S., and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR abs/1502.03167* (2015).

[15] Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093* (2014).

[16] Kundu, A., Vineet, V., and Koltun, V. Feature space optimization for semantic video segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 3168–3175.

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

[18] Lucas, B. D., and Kanade, T. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence* (August 1981), pp. 674–679.

[19] Martínez, F., Manzanera, A., and Romero, E. A motion descriptor based on statistics of optical flow orientations for action classification in video-surveillance. In *Multimedia and Signal Processing*. Springer, 2012, pp. 267–274.

[20] Menze, M., and Geiger, A. Object scene flow for autonomous vehicles. In *Conference on Computer Vision and Pattern Recognition (CVPR)* (2015).

[21] Noh, H., Hong, S., and Han, B. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision* (2015), pp. 1520–1528.

[22] Ranchin, F., and Dibos, F. Moving objects segmentation using optical flow estimation. Citeseer.

[23] Ren, S., He, K., Girshick, R., and Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems* (2015), pp. 91–99.

[24] Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., and LeCun, Y. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229* (2013).

[25] Sevilla-Lara, L., Sun, D., Jampani, V., and Black, M. J. Optical flow with semantic segmentation and localized layers. *CoRR abs/1603.03911* (2016).

[26] Simonyan, K., and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *CoRR abs/1409.1556* (2014).

[27] Tsai, Y.-H., Yang, M.-H., and Black, M. J. Video segmentation via object flow. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 3899–3908.

[28] Wei, S.-g., Yang, L., Chen, Z., and Liu, Z.-f. Motion detection based on optical flow and self-adaptive threshold segmentation. *Procedia Engineering 15* (2011), 3471–3476.

[29] Wu, Z., Shen, C., and van den Hengel, A. Wider or deeper: Revisiting the resnet model for visual recognition. *CoRR abs/1611.10080* (2016).

[30] Xie, S., and Tu, Z. Holistically-nested edge detection. *CoRR abs/1504.06375* (2015).

[31] Yalcin, H., Hebert, M., Collins, R., and Black, M. J. A flow-based approach to vehicle detection and background mosaicking in airborne video. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on* (2005), vol. 2, IEEE, pp. 1202–vol.

[32] Zitnick, C., Jojic, N., and Kang, S. B. Consistent segmentation for optical flow estimation. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on* (2005), vol. 2, IEEE, pp. 1308–1315.