Unsupervised Optical Flow Estimation with Temporal Smoothing

Akshay Sharma
akshaysh@andrew.cmu.edu

Muhammad Suhail
msaleem2@andrew.cmu.edu

Zeeshan Ahmed
zeeshana@cmu.edu

Carnegie Mellon University
5000 Forbes Ave, Pittsburgh, PA 15213

Abstract

Optical flow estimation has been tackled with deep convolutional neural networks (CNNs) in recent years. We utilize FlowNetC[3] architecture in an unsupervised framework and introduce temporal smoothing using three consecutive frames. Incorporating temporal smoothing leads to a reduction of MSE loss between the original images and ones predicted using the optical flow produced by the CNN. Motivation behind unsupervised learning stems from the lack of labeled ground truth data of optical flow and the issues that arise from domain mismatched between synthetic and real-world data.

1. Introduction

Optical flow is the two-dimensional motion field of two consecutive images. It can be represented as a two-dimensional vector field on the image plane.[13] Generally, it is solved by minimizing the brightness or color difference between corresponding pixels summed over the image. It is an under-constrained problem that has two classical solutions: patch-based, which performs summation locally over overlapping regions, and by regularizing by adding a smoothness term on one of the flow dimensions.[12]

Convolutional neural networks have achieve tremendous success in classifying images. The challenge with using CNNs for optical flow arises from the needs to correlate two sequential images, pixel-by-pixel. Fischer *et al*. [3] introduced a CNN which includes a correlation layer for matching features. However, it was found that the correlation did not add significant accuracy.

Abundance of video data makes optical flow estimation a powerful tool in various applications. Object tracking in autonomous driving, video semantic understanding, and object segmentation from video can benefit all from improvement in optical flow estimation. The biggest challenge remains the lack of pixel-by-pixel labeling of ground truth optical flow. This leads to challenges in training CNNs.

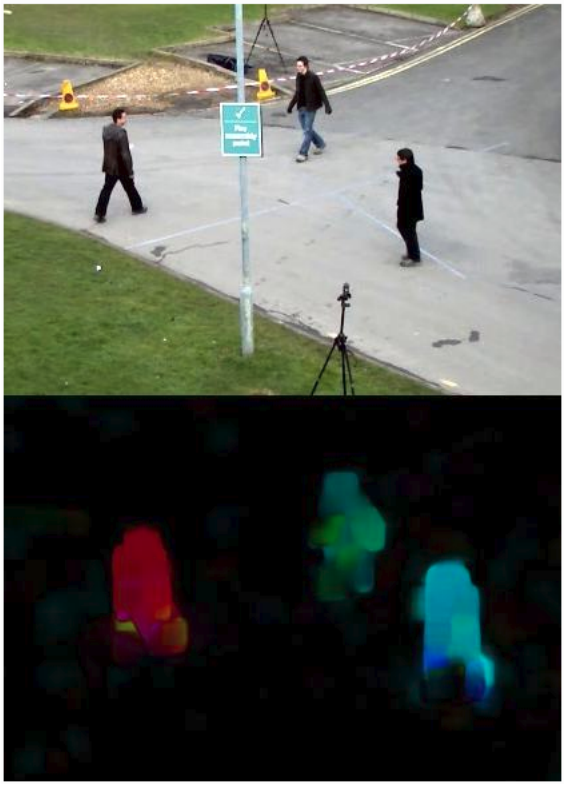

Figure 1. An example of optical flow [1]

Meister *et al*. [10] work on unsupervised CNN used bidirectional flow estimation and a robust census transform. Image pairs were passed twice, with the order swapped. They utilized FlowNet achitecture from Fischer *et al*. [3] and achieved competitive accuracy to supervised training methods.

Beyond the classical assumptions of brightness consistency and smoothness, we introduce temporal consistency.

Given a sequence of consecutive images, it can be assumed that the real world motion (or optical flow in x and y direction), can not undergo significant change over consecutive time steps. That is to say, objects in the real world, or motion of a camera, can not undergo unreasonable changes in acceleration. As an example, if a still camera was observing a cyclist at 5 meters per second in one frame, the optical flow between frame 1 and 2 can not be significantly different around the cyclist than the optical flow between frame 2 and 3, since the acceleration can only change by a reasonable amount.

This idea of temporal consistency or temporal smoothing is implemented in our work. The unsupervised framework is achieved by creating image prediction by warping optical flow with the first image and comparing to the second, original image. The CNN used is the correlation version of FlowNet. Instead of using the conventional method of using image pairs, we used three consecutive images. Optical flow maps are generated between the first and second images and also the second and third images. Warping the first optical flow with itself produces a prediction of the second optical flow with identical flow vectors. This is a method to incorporate the temporal consistency in the loss function, described in detail in the methodology section.

2. Related Work

One of the primary methods of optical flow estimation that is used in standard computer vision libraries like OpenCV[1] is Lucas-Kanade[8]. Belonging to the class of differential methods, it uses Taylor series approximation for estimation. Apart from the approximation, the algorithm also assumes that the flow is constant between neighboring pixels. Although the technique produces good results, these assumptions limit the accuracy of the estimation. This was the primary motivation for using a deep learning based approach for optical flow estimation.

One of the more prominent works amongst the deep learning based approach is FlowNet (Learning Optical Flow with Convolutional Networks)[3]. In this work Dosovitskiy et al. develop two different techniques FlowNetSimple and FlowNetCorr. They generate consecutive frames (Flying chair dataset) and passed it through a Convolutional Neural Network to predict the flow. Since the frames are synthetically generated, ground truth of optical flow was available. FlowNet 2.0[6] a paper succeeding FlowNet, discusses FLowNetS and FlowNetC in combination and traines individually on synthetic datasets in a specific order so as to improve the accuracy. SpyNet(Optical Flow Estimation using a Spatial Pyramid Network)[11] discusses the estimation of combining several individually trained networks as a spatial pyramid network. Here the flow is estimated at the highest level of the pyramid using low resolution images. The computed residual flow is passed onto the lower levels until we obtain the flow at the highest resolution. The major drawback of these approaches was the fact that they had used synthetic data which would significantly impact the accuracy of their model when tested on real world data. Furthermore, trying to combine several individual networks and training them independently on different datasets [9] in different orders induces several parameters which require tuning, hence is extremely cumbersome.

An approach which overcomes the above mentioned drawbacks is UnFlow (Unsupervised Learning of Optical Flow with a Bidirectional Census Loss) [10]. This is an unsupervised learning approach which trains by predicting the optical flow between frames and warping the first frame with the predicted flow to predict the second frame, which is then compared with the original second frame using some very interesting criteria. Further, they even make the model occlusion aware by using bi-directional estimation. Our work has been inspired from UnFlow and is an attempt to improve the performance of this approach using temporal smoothing, which will be explained in detail in the following sections.

Several other techniques like using a Soft Mask module[14] in the final layer of optical flow estimation have been proposed. However these use FlowNet as the base network and propose changes in the final layer of the network. We are proposing a technique which would overcome the drawbacks of FlowNet itself, and hence usage of such techniques on our network is bound to produce better results.

3. Key Idea

The key idea behind our work is that the optical flow between two time steps does not differ significantly. Hence, our aim is to achieve better optical flow estimation by smoothing the optical flow across consequent time steps. As proof of concept, we smooth the optical flow across three consecutive frames which we provide as input to our model. Losses and optimization is described in the methodology section.

4. Data

One of the major reasons we adopted an unsupervised learning technique is due to the lack of labelled real world data. The supervised algorithms in the field of optical flow estimation tend to make use of synthetic data, which is not an accurate representation of the real world and hence could lead to a reduction in accuracy.

For training our model we made use of the MCL-V database[7]. Three sequential images were generated from full HD (1920 x 1080) videos from the database. These images were converted from RGB format to YCbCr color space and only the Y channel was used for training. The Y channel captures all the required features and training only

on that channel improves the training speed considerably without any loss in accuracy. Finally, we normalized the pixel intensities of the images to the range [-1,1].

To evaluate our model, we used the videos from the KITTI dataset[4]. Similar to training data, we generated tuples of three sequential images, converted to YCbCr format, normalized the Y channel and input to the model.

5. Methodology

The input to our network is a tuple consisting of the Y channels of three consecutive frames $I_1,\ I_2\ , I_3$. The output is a set of two optical flows, one from the first frame to the second frame and another from the second frame to the third frame. The three consecutive frames are processed two at a time, (I_1, I_2) and (I_2, I_3) to generate their respective optical flow maps using FlownetC[3] architecture. However, instead of comparing the generated flow maps directly with ground truth optical flow, we adopt an unsupervised approach, taking inspiration from [10], and warped the first image of each set to generate the second image. So we get $OF_1,\ OF_2$ as the flow maps, and we generate,

$$I_{2,pred} = W(I_1, OF_1)$$

and

$$I_{3,pred} = WI_2, OF_2)$$

where W is the warp function which returns a warped version of the input images using the optical flow map. Now with these warped images and the flow maps, we calculate the following losses:

- MSE loss between the original images and warped images
- Spatial loss for individual flow maps to ensure spatial smoothing[5],[10]
- Temporal loss between the two flow maps for two consecutive sets of frames to ensure temporal smoothing

5.1. Architecture

The basic skeleton of the architecture to generate the optical flow maps comes from FlownetC[3].

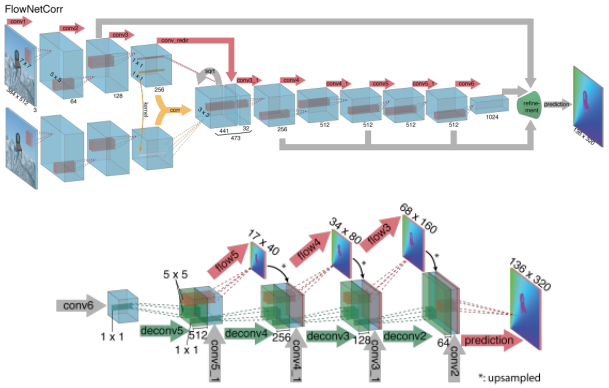

Figure 2. FlownetC[3] architecture

Instead of the shown 3 channel inputs in Figure 2, we just provide a single Y channel as input. Now the overall architecture works on the images as shown below.

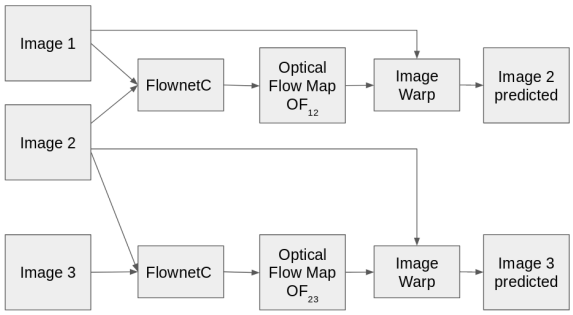

Figure 3. Full Architecture

5.2. Losses

As shown in Figure 3, we get two image predictions and two optical flow maps for every input of three images. Training the network uses three different losses as mentioned below.

5.2.1 MSE loss

FlownetC[3] by itself is an supervised framework but due to the lack of ground truth data for optical flows, we pivoted to unsupervised learning. In our framework, we warp the predicted optical flow between two images with the first image and generate an estimate of the second images. Then we use the second image which was a part of the input to calculate MSE loss between the estimated image and the original image. We calculate this loss for both the estimated second image and the estimated third image in our input. It has to be kept in mind that FlowNetC estimates optical flows in five different sizes which are bi-linearly up-sampled to the image size and are compared with the original flow. Similarly, we are using these five flows to predict five images which are then compared with the original image and a weighted sum of the losses are considered.

MSE loss between the original images and the predicted images:

$$loss_1 = MSE(I_2, I_{2,pred}) + MSE(I_3, I_{3,pred}) \quad (1)$$

5.2.2 Spatial loss

To ensure spatial smoothness[5] of the optical flow map generated by the network, we calculate a spatial loss over the generated optical flow which ensures that the flow value for each pixel is not very different from the flow values of its neighbouring pixels. We calculate the spatial loss using the same strategy as [10] which involves convolving the optical flow maps with fixed filters which give a measure of difference of the flow for a pixel with the nearby pixels. We use two separate filters, the first one calculates the difference with the horizontal and vertical pixels, whereas the second filter calculates the difference with the diagonal pixels.

$$F_1 = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad (2a)$$

$$F_2 = \begin{bmatrix} -1 & 0 & -1 \\ 0 & 4 & 0 \\ -1 & 0 & -1 \end{bmatrix} \quad (2b)$$

We calculate the smoothing loss using the Charbonnier loss function $\mathbf{C}(x)$ as used by[10]. It is defined as below:

$$\mathbf{C}(x) = (x^2 + \epsilon^2)^\gamma \quad (2c)$$

where ϵ is a small number in the range $(0, 1)$, and γ is an exponent which is usually kept at 0.45.

$$loss_2 = \sum_{\mathbf{x}} \mathbf{C}\Big[OF_{12}[N(\mathbf{x})] * F_1 + OF_{12}[N(\mathbf{x})] * F_2\Big] + \sum_{\mathbf{x}} \mathbf{C}\Big[OF_{23}[N(\mathbf{x})] * F_1 + OF_{23}[N(\mathbf{x})] * F_2\Big] \quad (2d)$$

where $N(\mathbf{x})$ defines the neighbourhood of the pixel location $\mathbf{x}$.

5.2.3 Temporal loss

The idea of spatial loss states that for any given small patch in an image for any pixel can not have a very different optical flow than its neighbours because images of real objects are rigid and thus there should be a constraint on their optical flow values. If we extend this idea in the time domain we can come up with a concept of temporal smoothing. If we consider the same object being filmed at two consecutive time steps, the flow values of the pixels representing that object can not change by significantly because real objects have a finite acceleration, and if the time steps at which the images were taken are very close than they are bound to have a very small change in their velocities. Thus, their optical flow values at two consecutive time steps should be similar.

But there is a problem in directly applying a constraint on the optical flow maps of consecutive time steps. We can not simply subtract the optical flow maps. This is because the pixel locations of objects in motion change between I_1 and I_2, described by the velocity or the optical flow value of pixel in I_1. If a particular object has a pixel location of $\mathbf{x}$ in OF_{12}, the same pixel location in OF_{23} will not give you the values of flow for that object. The object now occupies a different pixel location, given by: $\mathbf{x} + OF_{12}(\mathbf{x})$.

To compare the flow values of the same object in the two optical flow maps, we need to the locate the correct pixels in both the flow maps which refer to the same object. This can be done by warping the first flow map with itself. The resultant warped flow will map the flow values from the frame of reference of I_1 to that of I_2.

$$OF_{12,warped} = W(OF_{12}, OF_{12}) \quad (3)$$

Now we can directly take a difference of this warped flow $OF_{12,warped}$, and OF_{23} to calculate the variation in the flow values of the same object at two different time steps. Similar to calculating the spatial loss, we use the Charbonnier loss function on the above mentioned difference to get an estimate of the temporal loss.

$$loss_3 = \sum_{\mathbf{x}} \mathbf{C}\Big(OF_{12,warped}(\mathbf{x}) - OF_{23}(\mathbf{x})\Big) \quad (4)$$

Finally we add all the three losses to get the total loss of our network.

$$Total\ loss = loss_1 + loss_2 + loss_3 \quad (5)$$

5.2.4 Implementation

We have written the entire code from scratch in torch[2] which can be found at the following link: https://github.com/akshay-sharma1995/unflow.

6. Testing

As mentioned before, for training the model we used frames from the MCL-V dataset and defined spatial smoothing, temporal smoothing, and reconstruction loss as optimization criteria. However, for testing we made use of KITTI dataset. We generated triplets of image frames from the video, which we passed to our model. The output from the model consisted of optical flows from frame 1 to 2 and frame 2 to 3. These optical flows were used to warp image

1 to predicted image 2, and image 2 to predicted image 3 respectively. These predicted images were then compared with the actual images 2 and 3, using reconstruction loss as the criterion.

As a comparative analysis of our method, we implemented another network which worked exactly similar to ours, except that it did not consider temporal smoothing as a criterion for training. This way we will be able to accurately estimate the effect of temporal smoothing on optical flows.

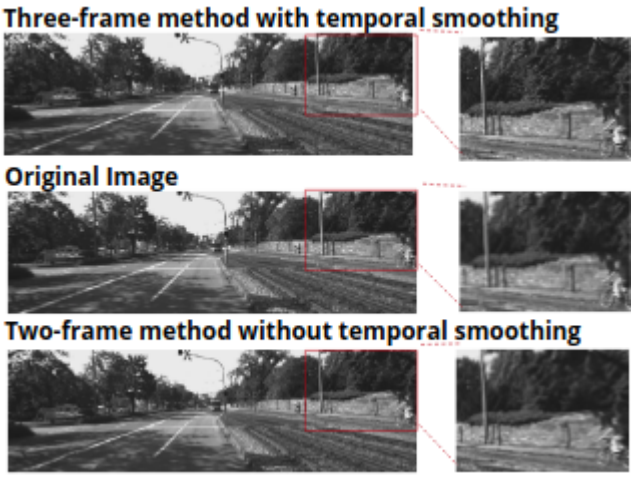

Figure 4. Predicted images found by warping the predicted flow with the previous frames. (Clear distortion could be seen around the shadow on the wall just after the cyclist, and at the pole for the image predicted by two frame method)

Figure 5. Predicted optical flow for the above frame. (Indicating how there's minimal flow towards the center of the frame as compared to the corners)

Our evaluation shows that the performance of optical flow estimation significantly improved by using temporal smoothing as an optimization criterion. Figure 4 clearly shows that the image predicted when not use temporal smoothing contains significant distortion. On the other hand, our method using temporal smoothing shows the pole and shadow to be significantly straighter, matching the original image.

During testing, a consistent reduction in loss was observed when using temporal loss. The results of the normalized loss for some of the testing frames for both the networks have been shown in Figure 6.

Method	Mean Loss
Without Temporal Loss	26487.839
With Temporal Loss	22669.432

Table 1. Comparison of Mean reconstruction loss

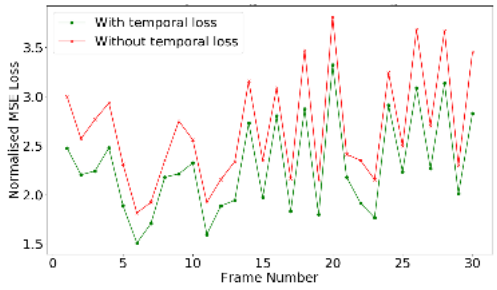

Figure 6. Comparison of reconstruction loss for the warped image using the predicted optical flows

Overall, it can be confirmed that not considering temporal loss has resulted in an increase of the mean reconstruction loss by 16.84%, which is a significant increase. It is to be noted that these losses are a weighted combination of several of the losses that we have mentioned in the methodology section and the large nature of these numbers do not mean that the estimations are poor.

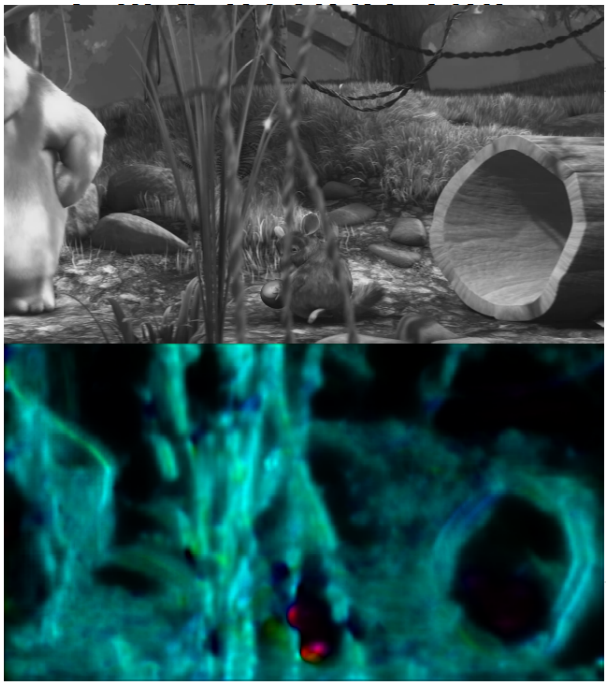

Figure 7. Optical flow map for a video where the mouse moves from left to right. (Clearly shown by the dark patch in the map)

7. Conclusion

The results show significant improvement in optical flow estimation using temporal smoothing. When comparing the exact same sequences of images in the same CNN architecture, incorporating temporal smoothing reduced mean reconstruction loss by 16.84%. When numerous sequential frames are available from videos, we strongly recommend the usage of temporal smoothing to better estimate the optical flow.

8. Future Work

Given the positive results of using temporal smoothing, it is important that we investigate the impact of smoothing the flow over more than three consecutive frames. In addition, we are considering using two frames as input and generating a middle frame using CNN and then incorporating that in the process of predicting optical flow by smoothing the flow over these three images. This method will enable a representative comparison to other networks which only use two images as input.

9. Contributions and Acknowledgements

We thank Advanced Agents - Robotics Technology Lab at Carnegie Mellon University for allowing us to train the CNN and run experiments on their GPU enabled server. We thank Krishna Toshniwal for guidance during this project. And finally we thank Dr. Amir Barati Farimani for teaching Machine Learning and Artificial Intelligence for Engineers and encouraging us to pursue a challenging project. All authors contributed equally to this work.

References

[1] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[2] R. Collobert, K. Kavukcuoglu, and C. Farabet. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*, 2011.

[3] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2758–2766, 2015.

[4] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[5] Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981.

[6] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. 12 2016.

[7] Joe Yuchieh Lin, Rui Song, Chi-Hao Wu, TsungJung Liu, Haiqiang Wang, and C-C Jay Kuo. Mcl-v: A streaming video quality assessment database. *Journal of Visual Communication and Image Representation*, 30:1–9, 2015.

[8] Bruce D Lucas, Takeo Kanade, et al. An iterative image registration technique with an application to stereo vision. 1981.

[9] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. arXiv:1512.02134.

[10] Simon Meister, Junhwa Hur, and Stefan Roth. Unflow: Unsupervised learning of optical flow with a bidirectional census loss. *arXiv preprint arXiv:1711.07837*, 2017.

[11] Anurag Ranjan and Michael J Black. Optical flow estimation using a spatial pyramid network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, page 2. IEEE, 2017.

[12] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[13] Andreas Wedel and Daniel Cremers. *Stereo scene flow for 3D motion analysis*. Springer Science & Business Media, 2011.

[14] Xi Zhang, Di Ma, Xu Ouyang, Shanshan Jiang, Lin Gan, and Gady Agam. Layered optical flow estimation using a deep neural network with a soft mask. *arXiv preprint arXiv:1805.03596*, 2018.