Forward Warping for Unsupervised Learning of Optical Flow

Fredrik Ekholm
fwe21@cam.ac.uk

Abstract

We propose and evaluate a new approach to unsupervised learning of optical flow, based on the splatting operation introduced by Niklaus et. al.[1]. Unsupervised Optical Flow is typically done by performing backwards warping to compute a photometric loss, and the reverse flow is used to generate an occlusion mask. By utilising a forward warping procedure instead, occlusions pose a lesser problem and only the forward flow need to be predicted during training. The code is available at https://github.com/Istlemin/softsplat-unsupervised-flow

1 Introduction

Optical Flow is a fundamental problem in computer vision, where one is given two images, and the goal is to find the movement of each pixel from the first to the second. Optical Flow has numerous applications, including video stabilization [2], frame interpolation [1], object detection [3] and object tracking [4]. Improvements in optical flow algorithms often directly translates to improvements in these downstream tasks, both considering accuracy and latency.

Traditionally, numerous approaches for Optical Flow have been proposed including energy-based, correlation-based, and gradient-based methods [5]. More recently, deep learning methods have shown success in the optical flow problem, where CNNs are trained to predict flow directly from two input images [6, 7, 8, 9]. Such neural network based methods now hold state-of-the art accuracy on all major benchmarks [10]. Another advantage over traditional approaches is inference latency of typically less than 100ms on a GPU, compared to many seconds or minutes [11].

One major drawback of learning based methods for optical flow is the requirement of large amount of annotated data to train on. Many available datasets are synthetically created [12, 13], and might not generalise well to all real world scenarios. Unsupervised learning methods aim to mitigate this by learning flow not from a reference flow but instead using photometric consistency and flow-smoothness assumptions. A common approach is to *backward-warp* the second image according to the predicted flow, and calculate a loss based on similarity with the first image This leads to problems with *occlusions*, i.e. when a pixel visible in the source image becomes occluded in target image the backwards-warped colour will be wrong, even if the flow prediction is correct. Handling occlusions properly is a key component of unsupervised optical flow [14]. One popular technique relies on predicting the flow in both directions, and using the range map of the backwards flow as on occlusion mask for the forwards flow.

Forward-warping is an alternative to backward-warping where the source image is warped to match target image by following the predicted flow. This deals better with occlusions, as some areas of the target image can have no pixels mapped to them. However, there is now a new problem of multiple pixels in the source image potentially being mapped to the same target location in the target image. I hypothesise that this is the reason forward-warping has seen less use in unsupervised optical flow learning. Niklaus and Liu [1] propose several methods of handling the ambiguity when multiple pixels map to one location in a differentiable manner. I apply this technique to train an unsupervised optical flow model while handling occlusion correctly, without requiring separate estimation. Potential benefits over the range-map based approach include that a single model inference per training step is needed instead of two, and potentially higher accuracy and/or better convergence due to occlusion handling and warping being handled together. This project demonstrates that the second is not the case, and that forward warping performs slightly worse.

2 Related Work

Optical Flow is a fundamental problem in computer vision and has received a lot of attention since foundational works [15]. Most traditional approaches are energy based, in that they optimise the flow based on an energy function representing photometric consistency and spatial smoothness of flow [16, 11]. Other trends include motion cost-volume calculation and estimating flow at different scales to deal with large motion [17, 18]. Traditional methods have also suggested various ways of handling occlusions. Wang et. al. [19] categorises occlusion methodology into three categories: treating occluded pixels as outliers [20, 21], predicting occlusion from range-map of backwards flow and ignore occluded pixels in loss function [22, 23], or building a more sophisticated representation of the image where depth information guide occlusion resolution [24, 25]. Most previous unsupervised optical flow methods follows closely to the second category. This project's approach will instead fall more into the third category, as we predict *z-values* for each pixel used to resolve ambiguity when multiple source pixels map to the same target pixel.

With the success of deep learning, models that predict flow directly from two input images start to appear. FlowNet [6, 7] first demonstrates the viability of using a CNN to predict flow, and improves state-of-the-art on the synthetic FlyingChairs dataset [13]. However, its performance lags behind on the more realistic dataset KITTI [26]. PWC-NET [8] suggest an improved architecture, encoding images into a spatial feature pyramids, predicting flow at different scales, and introducing motion cost-volume as an additional feature for flow prediction. PCW-NET is smaller, faster to train, and further improve accuracy on many datasets. RAFT [9] employs a recurrent approach, where flow is predicted at a single resolution but refined in multiple steps. MS-RAFT adds multiscale flow prediction, and is currently one of the best performing methods [27, 10]. State-of-the-art on the KITTI2015 benchmark is held by DDVM [28], which utilises a diffusion model pretrained on a large amount of image data.

Unsupervised Optical Flow aims to train a model for flow prediction without any labelled ground truth flow data. Large, diverse datasets for optical flow are challenging to produce, and current supervised approached often rely on synthetic data for training [9]. This has proven successful for performance on the real-world dataset KITTI2015, but might not generalise well to all scenarios. Unsupervised Optical flow need only video as training data, and it's therefore easier both to adapt training data to use-case and pre-train on a much larger dataset. Yu et. al. [29] first demonstrate the viability of unsupervised optical flow. They introduce the basic framework inspired by energy-based methods, where the loss consists of a photometric term calculated by backwards-warping the second image, and a smoothness term. [19] introduces occlusion handling, where the flow is predicted both ways (forward flow from I1 to I2 and reverse flow from I2 to I1). The range of the reverse flow is used as an occlusion mask to ignore occluded pixels in calculation of the photometric loss. UFlow [14] compares and combines many proposals from different papers into a new state-of-the-art method. In terms of occlusion handling, it considers the range-map approach and forward-backward flow consistency, finding range-map based occlusion reasoning to be the superior technique. SMURF significantly improve performance, by integrating the findings from UFlow [14] with the RAFT model [9].

The forward warping (also known as *splatting*) transformation, where pixels from the source image are "splatted" onto the target image by following the flow vectors, has seen significantly less attention in unsupervised optical flow than the backward warping operation. [19] uses a differentiable forward warping of a constant image following the reverse flow, but do not need to solve the multiple-source same target conflicts, as the warped image is only used as a mask. [1] introduces various ways of resolving same-target conflicts in a differentiable way, for the purpose of video frame interpolation. [30] uses the differentiable splatting operation defined by [1] in a supervised, mutliframe optical flow learning framework. This paper proposes using the same splatting operation as to directly to calculate a photometric loss for unsupervised optical flow, and I compare the different methods of resolving same-target conflicts.

3 Method

Overview Due to compute and time constraints, I choose to evaluate the viability of forward splatting in a framework similar to that of [19], as opposed to a newer, more complex and compute intensive, framework such as SMURF [31]. Specifically, we define a neural netwok M that predicts a flow F given two input images I_1, I_2. We also define a loss function $L(F, I_1, I_2)$ consisting of a photometric term and a smoothness term. The loss function make use of the splatting operation defined in [1]. The network is then optimised using gradient descent to minimize the loss on a given dataset of picture pairs, in our case the FlyingChair dataset [13].

Network architecture Given a pair of images I_1 and I_2, we use PWC-NET [8] to predict flows F_1, F_2, F_3, F_4, F_5 at different resolutions, starting at 25% of the original image resolution and halving the resolution in each level. From each level, PWC-NET also produces a feature map H_i, which is used in the prediction of flow in the next level. In order to resolve same-target conflicts for splatting, we use a single

trainable 3×3 convolutional layer c_i for each resolution, to predict z-values for each pixel: $Z_i = c_i(H_i)$.

Forwards and backwards warping The backwards warping operation $\overleftarrow{\omega}$ takes as a flow F and an image I_2 as input, and returns an images I_1'. For each pixel $\boldsymbol{p}$ in I_1', its colour is found by sampling I_2 at the target location of the flow starting at p. As the target location doesn't necessarily fall on an integer, a bilinear kernel is used for interpolation:

$$b(\boldsymbol{u}) = \max(1 - |\boldsymbol{u}_x|, 0) \cdot \max(1 - |\boldsymbol{u}_y|, 0)$$
$$I_1'[\boldsymbol{p}] = \sum_{\forall q \in I_2} b((\boldsymbol{p} + F[\boldsymbol{p}]) - \boldsymbol{q}) \cdot I_2[\boldsymbol{q}]$$
$$\overleftarrow{\omega}(I_2, F) = I_1'$$

Forward warping also takes a flow F a and an image I_1 as input, but instead splats each pixel $\boldsymbol{p}$ in I_1 onto the output image I_2' according to a bilinear kernel. The most basic form is summation splatting, where the contributions from multiple source pixels are simple summed up:

$$I_2'[\boldsymbol{p}] = \sum_{\forall q \in I_1} b((\boldsymbol{q} + F[\boldsymbol{q}]) - \boldsymbol{p}) \cdot I_1[\boldsymbol{q}]$$
$$\vec{\Sigma}(I_1, F) = I_2'$$

However, using summation splatting straight away would cause unwanted brightness increase when multiple source pixels map to the same target. Thus, [1] introduce average splatting, linear splatting and softmax splatting. Average splatting $\vec{\Phi}$ normalises the output by dividing by the total contribution, solving the brightness issue:

$$\vec{\Phi}(I_1, F) = \frac{\vec{\Sigma}(I_1, F)}{\vec{\Sigma}(\mathbf{1}, F)}$$

However, this still causes moving foreground objects to blend into the background. To alleviate this, linear splatting $\vec{*}$ and softmax splatting $\vec{\sigma}$ weigh the contribution from different pixels by a z-value mask Z:

$$\vec{*}(I_1, F, Z) = \frac{\vec{\Sigma}(Z \cdot I_1, F)}{\vec{\Sigma}(Z, F)}$$
$$\vec{\sigma}(I_1, F, Z) = \frac{\vec{\Sigma}(\exp(Z) \cdot I_1, F)}{\vec{\Sigma}(\exp(Z), F)}$$

Average, linear and softmax splatting all suffer from poorly behaved gradients for pixels where the numerator becomes small. This was found to severely slow down convergence. I try different solutions for this, including clipping and scaling the gradient by the numerator. Clipping was the only method I found to achieve reliable convergence, and I settle on clipping flow gradients to the interval $[-0.03, 0.03]$

Photometric Loss For each resolution level, the input images I_1 and I_2 are bilinearly resized to images I_{1i} and I_{2i} of the correct resolution. This image is then forward warped using the predicted flow to get $I_{2i}' = f(I_{1i}, F_i)$ where f is summation or average splatting, or $I_{2i}' = f(I_{1i}, F_i, Z_i)$ where f is linear or softmax splatting. We also compute an occlusion mask M_i representing how much weight each pixels in I_{2i}' should have in calculating the loss:

$$M_i = \vec{\Sigma}(\mathbf{1}, F)$$

. We stop gradients at mask calculation, as I found propegating gradients through led to highly unstable training. The photometric loss L_p can now be calculated as a weighted average of photometric discrepancy and image gradient discrepancy:

$$L_i^{p1} = \frac{\sum_{\forall p \in I_2} \psi((I_{2i}'[p] - I_{2i}[p]) \cdot M[p])}{\sum_{\forall p \in I_2} M[p]}$$
$$L_i^{p2} = \frac{\sum_{\forall p \in I_2} \psi((\nabla I_{2i}'[p] - \nabla I_{2i}[p]) \cdot M[p])}{\sum_{\forall p \in I_2} M[p]}$$
$$L_p = \sum_{i=1}^{5} L_i^{p1} + L_i^{p2}$$

where ψ is the Charbonnier function $\psi(x) = \sqrt{x^2 + \epsilon^2}$. We use $\epsilon = 10^{-3}$.

Smoothness loss We employ the edge-aware smoothness loss introduced in [19]. It weighs the smoothness penalty according to the gradient of the source image. We only consider first order smoothness:

$$L_i^s = \sum_{d \in \{x,y\}} \sum_{\forall p \in F_i} \psi(|\partial_d F_i[p]| \cdot e^{-\alpha |\partial_d I_{1i}[p]|})$$
$$L_s = \sum_{i=1}^{5} L_i^s$$
$$L = L_s + \lambda L_p$$

For our experiments, we set $\alpha = 50$ and $\lambda = 1 \cdot 10^{-5}$

4 Evaluation

I evaluate the approach on the FlyingChairs dataset [13], which is a synthetic dataset for optical flow constructed by superimposing pictures of chairs moving across backgrounds from Flickr. Due to the low diversity of types of flow seen in the images, it is considered of the easier datasets. We randomly split it into 95% training data and 5% testing data. All our models are trained with the Adam optimizer [32], with

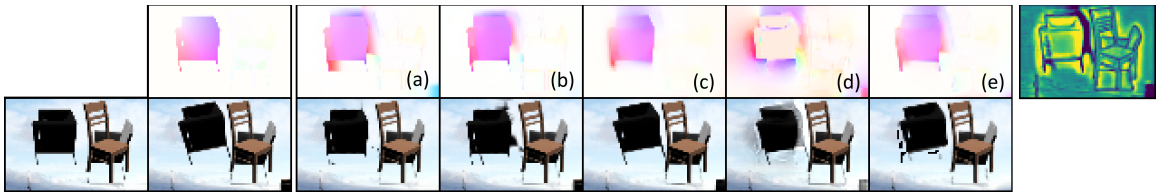

Figure 1: Predicted flows and resulting warped images for the various approaches. On the left: input image pair and ground truth flow. In the middle: Predicted flow and warped images for (a) Backwards Warping. (b) Backwards Warping + range-map occlusion. (c) softmax Splatting. (d) Summation splatting (e) Average splatting. The black artefacts to the left of the chair in (e) are pixels outside the range of the flow, and thus masked out. On the right: predicted z-values for softmax splatting

Warping mode	Train EPE	Test EPE	Loss
backwards warping	**5.10** ± 0.02	**5.23** ± 0.05	6.16
backwards warping + range-map occlusion	**4.55** ± 0.02	**4.70** ± 0.08	5.15
softmax Splatting	**5.01** ± 0.01	**5.25** ± 0.02	4.27
Summation Splatting	**11.13** ± 0.00	**11.68** ± 0.02	5.56
Linear Splatting	**11.59** ± 0.15	**11.93** ± 0.10	11.42
Average Splatting	**4.97** ± 0.04	**5.06** ± 0.06	5.95
Average Splatting without grad clipping	**8.17** ± 0.06	**8.77** ± 0.01	7.56

Table 1: Results on the FlyingChairs dataset. The **EPE** columns show average and standard deviation of EPE across the last three epochs. The **Loss** column shows the average photometric loss in the last epoch

$\beta_1 = 0.9$ and $\beta_2 = 0.999$, and we use a batch size of 4. Due to computational constraints, we only train for 10 epochs with a learning rate of 10^{-4}, followed by 5 epochs with a learning rate of $5 \cdot 10^{-5}$. This is significantly shorter than the upwards of 100 epochs unsupervised flow models typically are trained for [19, 14, 29]. Results are reported in terms of Average End-Point Error (EPE), which is defined as the average euclidean distance between the predicted flow vector and the ground truth flow vector, over the entire dataset.

I evaluate the four splatting approaches introduced in section 3, by comparing them against two baselines: backwards warping with no occlusion handling, and backwards warping with range-map occlusion masking, as introduced by Wang et. al. [19]. All hyperparameters are kept identical, other than the choice of warping operation. It would be preferable to run training multiple times to report a mean and variance of EPE, but due to computational limits I instead report mean and variance EPE over the last three epochs of training. Results are shown in table 1.

Among the forward splatting approaches, only average splatting beat the backwards warping baseline, and none beat backwards warping with range-map occlusion. Softmax performs comparably to the baseline, while neither Summation Splatting nor Linear Splatting manages to perform any meaningful optical flow prediction, and both end up worse than predicting a constant flow of zero (11.04 EPE on train and 11.66 on test). For Linear Splatting, this is caused by highly unstable training, presumably due to ill-behaved gradients of z-values. Summation Splatting however shows stable training and achieves a low photometric loss of 5.56. However, due to the brightness artefacts discussed in section 3, this low loss does not align with correct warping. In the example in figure 1, the flow at the chair is predicted to be small, while a larger flow is predicted above and to the right in order to darken that area.

Comparing softmax and average splatting, we see that softmax splatting achieves a lower photometric loss but a worse EPE. This indicates that the additional expressability in learning to predict z-values does help in constructing a warped image similar to the target. However, similarly to summation splatting it doesn't seem to promote correct flow prediction. In figure 1, we see that softmax splatting shows less artefacts around where the black armchair start to cover the sky, as the z-values put the chair in front of the sky. However, it does show additional artifacts around the bottom left are of the chair, where it is predicting a movement of the sky to fill in the area no longer covered by the chair. In this area, the background has a predicted z-value higher than the chair, which is incorrect.

5 Conclusions

Splatting has been shown to be a viable option for unsupervised optical flow, but does for now lag behind backwards flow with occlusion reasoning in performance. Among the options for splatting strategies, average splatting performs the best for our limited experiments. It is possible that results can be improved through fine-tuning parameter and training for more epochs, and might also not generalise to other datasets and more modern models for optical flow. Further parameter tuning and using splatting in more recent unsupervised approaches such as SMURF [31] would be good future directions.

References

[1] Simon Niklaus and Feng Liu. *Softmax Splatting for Video Frame Interpolation*. 2020. arXiv: 2003.05534 [cs.CV].

[2] Shuaicheng Liu et al. "SteadyFlow: Spatially Smooth Optical Flow for Video Stabilization". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 4209–4216. DOI: 10.1109/CVPR.2014.536.

[3] Xizhou Zhu et al. *Flow-Guided Feature Aggregation for Video Object Detection*. 2017. arXiv: 1703.10025 [cs.CV].

[4] JY Bouguet. *Pyramidal implementation of the affine Lucas Kanade feature tracker description of the algorithm, Intel Corp., 5, 4*. 2001.

[5] John L Barron, David J Fleet, and Steven S Beauchemin. "Performance of optical flow techniques". In: *International journal of computer vision* 12 (1994), pp. 43–77.

[6] Alexey Dosovitskiy et al. "Flownet: Learning optical flow with convolutional networks". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 2758–2766.

[7] Eddy Ilg et al. "Flownet 2.0: Evolution of optical flow estimation with deep networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2462–2470.

[8] Deqing Sun et al. "Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 8934–8943.

[9] Zachary Teed and Jia Deng. "Raft: Recurrent all-pairs field transforms for optical flow". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer. 2020, pp. 402–419.

[10] Azin Jahedi et al. "MS-RAFT+: High Resolution Multi-Scale RAFT". In: *International Journal of Computer Vision* (2023), pp. 1–22.

[11] Jerome Revaud et al. "Epicflow: Edge-preserving interpolation of correspondences for optical flow". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1164–1172.

[12] D. J. Butler et al. "A naturalistic open source movie for optical flow evaluation". In: *European Conf. on Computer Vision (ECCV)*. Ed. by A. Fitzgibbon et al. (Eds.) Part IV, LNCS 7577. Springer-Verlag, Oct. 2012, pp. 611–625.

[13] A. Dosovitskiy et al. "FlowNet: Learning Optical Flow with Convolutional Networks". In: *IEEE International Conference on Computer Vision (ICCV)*. 2015. URL: http://lmb.informatik.uni-freiburg.de/Publications/2015/DFIB15.

[14] Rico Jonschkowski et al. "What Matters in Unsupervised Optical Flow". In: *CoRR* abs/2006.04902 (2020). arXiv: 2006.04902. URL: https://arxiv.org/abs/2006.04902.

[15] Steven S. Beauchemin and John L. Barron. "The computation of optical flow". In: *ACM computing surveys (CSUR)* 27.3 (1995), pp. 433–466.

[16] Thomas Brox et al. "High accuracy optical flow estimation based on a theory for warping". In: *Computer Vision-ECCV 2004: 8th European Conference on Computer Vision, Prague, Czech Republic, May 11-14, 2004. Proceedings, Part IV 8*. Springer. 2004, pp. 25–36.

[17] Thomas Brox and Jitendra Malik. "Large displacement optical flow: descriptor matching in variational motion estimation". In: *IEEE transactions on pattern analysis and machine intelligence* 33.3 (2010), pp. 500–513.

[18] Christian Bailer, Bertram Taetz, and Didier Stricker. "Flow fields: Dense correspondence fields for highly accurate large displacement optical flow estimation". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 4015–4023.

[19] Yang Wang et al. "Occlusion aware unsupervised learning of optical flow". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 4884–4893.

[20] Alper Ayvaci, Michalis Raptis, and Stefano Soatto. "Occlusion detection and motion estimation with convex optimization". In: *Advances in neural information processing systems* 23 (2010).

[21] Alper Ayvaci, Michalis Raptis, and Stefano Soatto. "Sparse occlusion detection with optical flow". In: *International journal of computer vision* 97 (2012), pp. 322–338.

[22] Luis Alvarez et al. "Symmetrical dense optical flow estimation with occlusions detection". In: *International Journal of Computer Vision* 75 (2007), pp. 371–385.

[23] Junhwa Hur and Stefan Roth. "MirrorFlow: Exploiting symmetries in joint optical flow and occlusion estimation". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 312–321.

[24] Deqing Sun, Erik Sudderth, and Michael Black. "Layered image motion with explicit occlusions, temporal consistency, and depth ordering". In: *Advances in Neural Information Processing Systems* 23 (2010).

[25] Laura Sevilla-Lara et al. “Optical flow with semantic segmentation and localized layers”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 3889–3898.

[26] Moritz Menze, Christian Heipke, and Andreas Geiger. “Object Scene Flow”. In: *ISPRS Journal of Photogrammetry and Remote Sensing (JPRS)* (2018).

[27] Azin Jahedi et al. *Multi-Scale RAFT: Combining Hierarchical Concepts for Learning-based Optical FLow Estimation*. 2022. arXiv: 2207.12163 [cs.CV].

[28] Saurabh Saxena et al. *The Surprising Effectiveness of Diffusion Models for Optical Flow and Monocular Depth Estimation*. 2023. arXiv: 2306.01923 [cs.CV].

[29] Jason J. Yu, Adam W. Harley, and Konstantinos G. Derpanis. *Back to Basics: Unsupervised Learning of Optical Flow via Brightness Constancy and Motion Smoothness*. 2016. arXiv: 1608.05842 [cs.CV].

[30] Bo Wang et al. *SplatFlow: Learning Multi-frame Optical Flow via Splatting*. 2023. arXiv: 2306.08887 [cs.CV].

[31] Austin Stone et al. *SMURF: Self-Teaching Multi-Frame Unsupervised RAFT with Full-Image Warping*. 2021. arXiv: 2105.07014 [cs.CV].

[32] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].