DeepHHD: Learning Helmholtz-Hodge Decomposition to Predict Optical Flow

Abrar Anwar
Department of Computer Science
University of Texas
Austin, Texas
abraranwar@utexas.edu

Abstract

Convolutional neural networks (CNNs) have been successful in per-pixel prediction tasks such as optical flow; however, it does have its flaws. Optical flow can be viewed as a vector field, where there exists much ongoing research on vector field reconstruction for the purpose of fluid simulation. We propose and explore a method using a supervised architecture that attempts to predict the Helmholtz-Hodge decomposition of a vector field, whose sum is the expected optical flow between two images. The code can be found at the following repository: https://github.com/AbrarAnwar/DeepHHD.

1 Introduction

Flownet [4][8] created a new approach to optical flow estimation using CNNs. Much of the research in the past used variational approaches inspired from the Horn-Schunck [6] and Lucas-Kanade [14] methods. Since then, a variety of methods using CNNs to estimate optical flow have popped up [13][12][15], adding features such as unsupervised learning or using generative models.

In this paper, we introduce a method for optical flow estimation using the Helmholtz-Hodge decomposition (HHD). We then discuss the results and compare it to other common approaches. Then we discuss future options for continuing this research.

2 Background

2.1 Classical Optical Flow Estimation

Optical flow is the detection of the motion of pixels between two consecutive frames. Historically, two differential methods for estimation of optical flow are the Lucas-Kanade and Horn-Schunck methods [14] [6], developed in the 1980s. Numerous assumptions are typically made in an optical flow model such as a brightness constancy constraint. The assumption typically is that a given pixel moving across multiple frames, the frames will have the same brightness across those frames. This is typically written as

$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \Delta t) \tag{1}$$

In addition, it is assumed that the motion of a pixel between two consecutive images are small. These constraints together develop the brightness constancy equation.

$$\frac{dI}{dt} = \frac{\partial I}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial I}{\partial y}\frac{\partial y}{\partial t} + \frac{\partial I}{\partial t} = 0 \tag{2}$$

The equation can be simplified in it's shorthand form as

$$\frac{dI}{dt} = I_x u + I_y v + I_t = 0 \tag{3}$$

I_x and I_y are the spatial derivatives, which are calculated through image gradient techniques such as the Sobel operator. u and v is the optical flow which is desired. It is easy to see Equation 3 is a linear equation and lies on a line. Thus, multiple values of u and v can represent a valid solution given it is an undetermined problem.

The Horn-Schunck method assumes smoothness in the flow, thus tries to smooth the image iteratively while using while solving a linear system. Modern approaches inspired by Lucas-Kanade's approach uses image pyramids, where multiple resolutions of the image are used to get information across multiple scales.

2.2 Deep Learning Approaches to Optical Flow Estimation

Since the advent of deep learning, optical flow was a popular problem to solve. FlowNet [4] was the first to use convolutional neural networks (CNNs) to learn and predict optical flow with striking accuracy and speed in an end-to-end fashion. CNNs have shown great strengths in per-pixel prediction in tasks such as segmentation, thus should also show strengths in optical flow. Continued work for a direct end-to-end approach includes SpyNet [17] which used a spatial pyramid network to handle multiple scales and PWC-Net [18] which uses warping and a cost volume layer to estimate optical flow at various scales. One of the latest approaches is VCN [19], which doesn't solve for optical flow directly, but instead tries to treat the problem as a hypothesis selection problem for pixel correspondences using a Siamese network. FlowNet and FlowNet2.0 happen to be the largest network of all of these approaches, where FlowNetC has over 38M parameters and FlowNet2 having 162.49M parameters. In comparison, PWC-Net has 8.75M and VCN has 6.20M parameters, while achieving performance that exceeds that of the variants of FlowNet [7].

There exist several approaches beyond an end-to-end trainable network for optical flow. ProFlow [15] treats each frame as a model and learns online. It follows the assumption that a dataset to train under is not sufficient to generalize for some test dataset, thus learns online. Each model per frame is location dependent, thus ProFlow is able to predict in occluded locations. SelFlow [13] is a semi-supervised method to learning optical flow that uses a photometric loss from unsupervised methods (which use image warping to learn) and combines it with the addition of multiple forms of synthetic occlusions to the data to improve performance. They conclude it is reasonable to not train on a prelabeled dataset, as pretraining using a self-supervised model on unlabeled data shows good performance.

FlowNet develops multiple architectures. The simplest one consists of a straight-forward, multi-layer CNN architecture where the input is the two temporally consecutive images stacked. This method would work but would also likely lack the ability to find a good minimum. FlowNetCorr is an upgrade where each image is put into separate CNN processing streams where a smaller representation of the image is generated. Next, a 'correlation layer' performs multiplicative patch comparisons between the two representations. This is done by convolving patches (not filters) from one feature map onto the other. These are fed into a series of convolutional layers and are then refined through a series of unpooling and deconvolutional layers to go from a coarse feature map to a finer, dense per-pixel estimation. This correlation layer is the industry standard for state-of-the-art approaches to optical flow estimation. PWC-Net and VCN also uses the same correlation layer as FlowNet. [18] [19].

The loss used is endpoint error (EPE), which is the average Euclidean distance between the predicted flow vectors for each pixel to each ground truth flow vector. This is a common loss function for deep learning approaches for optical flow estimation problems, as well as the metric used to compare the accuracy between different techniques. This loss function can be shown as:

$$L_{EPE} = \frac{1}{WH} \sum_{ij}^{W,H} ||u_{ij} - \hat{u}_{ij}||_2 \tag{4}$$

where u_ij represents the ground truth flow flow vector of column pixel i and row pixel j for an image with a height of H and width of W. $\hat{u}_{ij}$ is the predicted output of the neural network.

FlowNet 2.0 [8] attempts to solve the multiple issues in its predecessor, namely the slowness and the inability to handle small displacements of images. To handle this, they developed a complex, stacked architecture of multiple FlowNets to handle large displacements and small displacements. This stacked network decreased speed compared to the original FlowNet, but increased accuracy by

about 50%. Different variations of their stacked networks can increase speed heavily, however the official FlowNet 2.0 architecture is slightly slower than it's predecessor.

The main takeaway from Flownet's approaches for the creation of our architecture is the use of a correlation layer. A representation through a correlation layer provides more interesting information for the network compared to simply appending two images together.

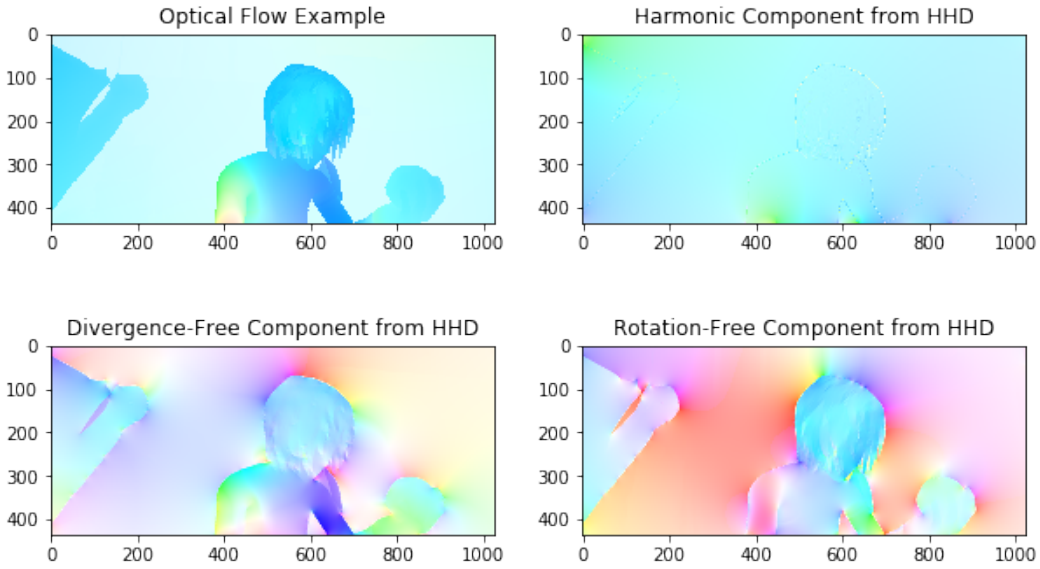

Figure 1: HHD on a flow from the Sintel dataset

2.3 Vector Field Reconstruction

Optical flow fields are vector fields. Work on vector field reconstruction has existed for decades, focusing on simulating fluids. DeepFlow [10] was the first generative network that reconstructed fluid simulations from a set of reduced parameters. They used a novel stream function based loss defined as:

$$L_G(c) = ||u_c - \nabla \times G(c)||_1 \tag{5}$$

u_c is the simulation sample from the training data built using parameters c and $G(c)$ is the network output given parameters c. The parameters c are a combination of the x position of the source, the source's width, and the current time frame. Using this, they are able to stunningly recreate their fluid simulations hundreds of times faster than analytical methods. The curl of the output of the neural network is built to be the reconstruction target, and is thus guaranteed to be divergence free by construction, thus can reconstruct incompressible flows reliably; however, it is better to use a direct loss function if fluids are compressible. This approach simply removed the curl operator as seen in Equation 6.

$$L_G(c) = ||u_c - G(c)||_1 \tag{6}$$

This equation begins to resemble FlowNet's EPE loss function. In order to handle icompressible and compressible flows, the loss function that was most effective was a weighted combination of the following.

$$L_G(c) = \lambda_1||u_c - \hat{u}_c||_1 + \lambda_2||\nabla u_c - \nabla \hat{u}_c||_1 \tag{7}$$

where $\hat{u}_c = \nabla \times G(c)$. This augmentation uses gradient information to improve the vector field data, as seen in the latter portion. This approach to reconstruct optical flow from an input parameter set c will inspire our approach to reconstruct vector fields from the image data.

2.4 Helmholtz-Hodge Decomposition

The Helmholtz-Hodge Decomposition (HHD) decomposes an arbitrary vector field into rotation-free, divergence-free, and harmonic vector fields [3]. Given a vector field $\vec{v}$, the decomposition is as such:

$$\vec{v} = \vec{d} + \vec{r} + \vec{h} \tag{8}$$

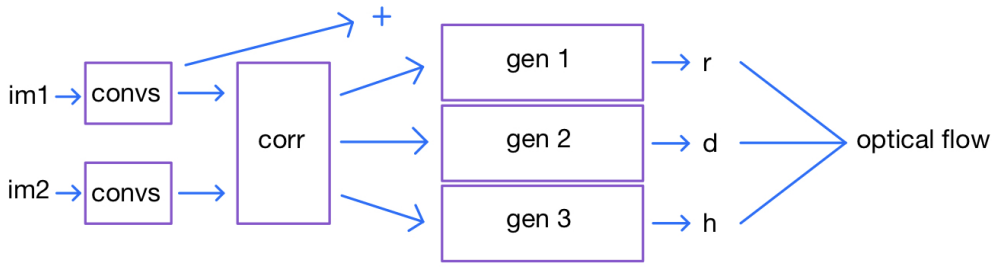

Figure 2: Network architecture visualized. The plus symbol after the correlation layer is meant to show the concatenation of the first image to the correlation layer's output before it's fed into the generators

Method	MPI Sintel Train
FlowNetS	4.50
FlowNetC	4.31
FlowNet2.0	2.93
DeepFlow	3.31
Horn+Schunck	8.739
HHD + EPE	13.2615302
HHD Loss	13.1203958

Table 1: EPE Results for various approaches (EPEs for other methods come from [13])

$\vec{d}$ is rotation-free, $\vec{r}$ is divergence-free, and $\vec{h}$ is both divergence-free and rotation-free. By definition of rotation-free and divergence free, we get the following equalities:

$$\begin{aligned} \nabla \cdot \vec{d} &= \nabla \cdot \vec{v} \\ \nabla \times \vec{r} &= \nabla \times \vec{v} \end{aligned} \tag{9}$$

HHD has been used in the past for a variety of applications, including limited uses for optical flow. For example, the navigation of robots using 3D optical flow was calculated using HHD [16]. Kiristis et. al [11] used various vector field decomposition and variational regularization methods in order calculate optical flow for fluorescence response of zebrafish microscopy image sequences. [5] showed that discrete HHD was effective at decomposing the motion fields of synthetic datasets.

There exists a variety of methods to calculate HHD, the two most popular being discrete HHD [1] and natural HHD [3]. Although their decompositions are both valid, the latter was chosen due to the ease of access to their HHD solver. The natural HHD is calculated by separating the flows through internal and external influences on the domain. $\vec{d}$ and $\vec{r}$ are calculated using their natural components to represent the flows inside the domain. The last part, the natural harmonic $\vec{h}$, is thus influenced by the external boundary of the domain.

3 Architecture

The architecture implemented for DeepHHD, as seen in Figure 2, is that given two images, we pass them through two image towers that learn a reduced representation of the images. This is then followed by the correlation layer seen in FlowNet. This correlation layer produces a representation of the two images together. An additional convolutional layer is used with the image representation of the first image and then concatenated to the end of the output of the correlation layer. This output is used when network branches off into three generators, each of which produces a vector field representing its respective HHD component. An example of the training data output can be seen in Figure 1.

The generator starts with a convolutional layer followed by a Resnet-like architecture consisting of 4 big blocks that are made up of a small blocks that consist of 4 repeated flat convolutional layers

with a Leaky ReLU activation. The big block ends with an additive skip connection and an upsample to two times input size. This is meant to take the reduced representation and slowly upsample it by order of two until it reaches the proper size (which takes 4 upscales to reach). Finally, after 4 repeated big blocks, the network ends with a last convolutiopnal layer that brings the size of the input to the proper size of a vector field. This generator was inspired by DeepFluids' generator network as it was shown to be reliable for reconstructing vector fields.

The goal is for each generator to learn how to reconstruct simpler representations of the vector fields needed to construct the final optical flow. The loss function used ensures the reconstruction of the HHD components are accurate:

$$L_{HHD} = ||r - \hat{r}|| + ||d - \hat{d}|| + ||h - \hat{h}|| \tag{10}$$

An additional loss function that could be used is as follows:

$$L_{combined} = L_{HHD} + L_{EPE} \tag{11}$$

Equation 11 could be useful in reconstructing the vector field while also recreating the decompositions accurately. In addition, another potential loss function could take advantage of the equalities defined in Equation 9.

$$L_{eq} = ||\nabla \cdot \hat{d} - \nabla \cdot v|| + ||\nabla \times \hat{r} - \nabla \times v|| \tag{12}$$

This loss could be interesting, as we would start estimating $\vec{d}$ and $\vec{r}$ without having to compute the expensive decomposition for the training data beforehand.

4 Results and Discussion

Due to limitations in training hardware, training and testing were done only on the MPI Sintel Dataset's training set. The Sintel dataset is a popular synthetically generated optical flow dataset created from a CGI movie. Table 1 shows the average EPEs on the dataset using various methods. It appears that the results of our approach on the dataset are pretty poor compared even to the baseline Horn+Schunck method. Interestingly, the use of HHD loss alone seems to slightly have an edge outperforming the HHD+EPE loss. A major limitation of this project was the lack of GPU capabilities, as use of minibatch was limited due to a batch size of one taking up an entire GPU's memory. Due to this, a single test of new iterations of the architecture would take an entire day, quickly eating up all the allotted time.

Although the results were poor, they also show promise, as the inspection of the distribution of the resultant EPEs in Figure 3 show that it is incredibly high for some values, pushing the average EPE towards the higher end. I believe these high outliers are caused by the lack of per-pixel prediction of flow across multiple scales. The network as it is now is able to handle coarse predictions, which welcomes small amounts of motion. Numerous works in the past have handled multiple scales of motion using image pyramids [14], developing pyramid networks [17], as well as FlowNet's approach of a refinement process for the coarse output of a neural network.

In addition, most state-of-the-art optical flow networks are pretrained on some separate, synthetically generated dataset. FlowNet uses a Flying Chairs dataset developed from randomly generated chairs on random backgrounds. CNNs require a lot of data in order to be able to generalize, and this pretraining process augments the network. Flying Chairs with over 22k image pairs with their associated optical flows, totaling to over 30 gigabytes of data. The dataset was too large to run on my machine due to space and internet constraints.

5 Conclusion and Future Work

In the future, work on computing motions at varying scales is needed, and could take inspiration from new developments in variational methods such as normalizing flows and variational autoencoders. These methods might be able to handle the multi-scale issue along with predicting what occurs when the image is occluded (which the Sintel datasets counts as unmatched pixels). These unmatched portions of the test dataset tend to have incredibly high rates of error compared to matched portions of the images. In addition, normalizing flow's ability for accurate density estimation may provide

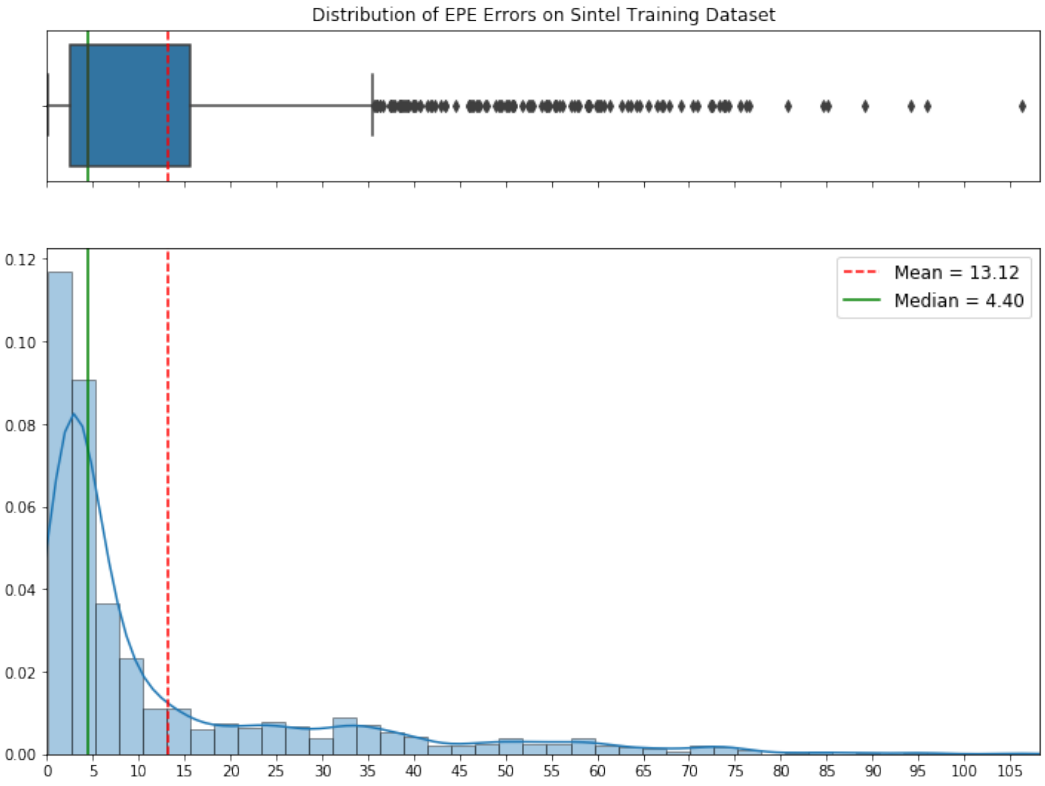

Figure 3: EPE distribution using HHD loss

useful for match density estimation which has been shown as effective for estimating optical flow and stereo matching by learning pixel correspondences [20].

In conclusion, a network for reconstructing the HHD components of an optical flow estimate was developed and tested. Although the results were not great, if a proper implementation for the generative portion of the network were to be created, my intuition is that a network estimating the simpler components of an optical flow would result in increased accuracy than computing it all in one go. Ideally, an upgrade to the generator will be low in the number of parameters, as all new research in the field has been moving in that direction, especially since DeepHHD aims to reconstruct three vector fields.

An interesting idea to continue this work would be to use convex optimization layers [2] as proposed by Amos et al. This differentiable optimization layer allows for easy embedding of constraints into a deep learning frameworks. A recent paper by Jiang et al. [9] used bi-level optimization to embed an epipolar geometric constraint into a deep learning network in order to predict both an optical flow estimate and an essential matrix estimate. They were able to get competitive results for optical flow as well as significant results on essential matrix estimation. I believe that convex optimization layers could be used to embed constraints given by the equalities in Equation 9 into the predicted outputs of the network.

References

[1] E. Ahusborde, M. Azaïez, J.-P. Caltagirone, M. Gerritsma, and A. Lemoine. DISCRETE HODGE HELMHOLTZ DECOMPOSITION. page 10.

[2] B. Amos and J. Z. Kolter. OptNet: Differentiable optimization as a layer in neural networks.

[3] H. Bhatia, V. Pascucci, and P.-T. Bremer. The natural helmholtz-hodge decomposition for open-boundary flow analysis. 20(11):1566–1578. ISSN 1941-0506. doi: 10.1109/TVCG.2014.2312012. Conference Name: IEEE Transactions on Visualization and Computer Graphics.

[4] P. Fischer, A. Dosovitskiy, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox. FlowNet: Learning optical flow with convolutional networks.

[5] Q. Guo, M. K. Mandal, and M. Y. Li. Efficient hodge–helmholtz decomposition of motion fields. 26(4): 493–501. ISSN 01678655. doi: 10.1016/j.patrec.2004.08.008.

[6] B. K. P. Horn and B. G. Schunck. Determining optical flow. In *Techniques and Applications of Image Understanding*, volume 0281, pages 319–331. International Society for Optics and Photonics. doi: 10.1117/12.965761.

[7] J. Hur and S. Roth. Optical flow estimation in the deep learning age.

[8] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. FlowNet 2.0: Evolution of optical flow estimation with deep networks.

[9] S. Jiang, D. Campbell, M. Liu, S. Gould, and R. Hartley. Joint unsupervised learning of optical flow and egomotion with bi-level optimization.

[10] B. Kim, V. C. Azevedo, N. Thuerey, T. Kim, M. Gross, and B. Solenthaler. Deep fluids: A generative network for parameterized fluid simulations. 38(2):59–70. ISSN 0167-7055, 1467-8659. doi: 10.1111/cgf.13619.

[11] C. Kirisits, L. F. Lang, and O. Scherzer. Decomposition of optical flow on the sphere. 5(1):117–141. ISSN 1869-2672, 1869-2680. doi: 10.1007/s13137-013-0055-8.

[12] W.-S. Lai, J.-B. Huang, and M.-H. Yang. Semi-supervised learning for optical flow with generative adversarial networks. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 354–364. Curran Associates, Inc.

[13] P. Liu, M. Lyu, I. King, and J. Xu. SelFlow: Self-supervised learning of optical flow. version: 1.

[14] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. page 9.

[15] D. Maurer and A. Bruhn. ProFlow: Learning to predict optical flow. version: 1.

[16] Y. Mochizuki and A. Imiya. Spatial reasoning for robot navigation using the helmholtz-hodge decomposition of omnidirectional optical flow. In *2009 24th International Conference Image and Vision Computing New Zealand*, pages 1–6. doi: 10.1109/IVCNZ.2009.5378430. ISSN: 2151-2205.

[17] A. Ranjan and M. J. Black. Optical flow estimation using a spatial pyramid network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2720–2729. IEEE. ISBN 978-1-5386-0457-1. doi: 10.1109/CVPR.2017.291.

[18] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz. PWC-net: CNNs for optical flow using pyramid, warping, and cost volume.

[19] G. Yang and D. Ramanan. Volumetric correspondence networks for optical flow. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d. Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 794–805. Curran Associates, Inc.

[20] Z. Yin, T. Darrell, and F. Yu. Hierarchical discrete distribution decomposition for match density estimation.