TECHNISCHE
UNIVERSITÄT
DARMSTADT

PROBABILISTIC OPTICAL FLOW AND ITS IMAGE-ADAPTIVE REFINEMENT

A dissertation submitted to
TECHNISCHE UNIVERSITÄT DARMSTADT
Fachbereich Informatik

in fulfillment of the requirements for the degree of
Doctor rerum naturalium (Dr. rer. nat.)

presented by

ANNE SABINE WANNENWETSCH
M.Sc.

born in Stuttgart, Germany

Examiner:	Prof. Stefan Roth, Ph.D.
Co-examiner:	Prof. Dr. Thomas Brox
Date of Submission:	14th of October, 2020
Date of Defense:	29th of January, 2021

Darmstadt, 2021

Anne Sabine Wannenwetsch:
Probabilistic Optical Flow and its Image-Adaptive Refinement
Darmstadt, Technische Universität Darmstadt

Year thesis published in TUprints: 2021
Date of defense: 29.01.2021

ABSTRACT

Optical flow estimation, *i. e.* the prediction of motion in an image sequence, is an essential problem in low-level computer vision. Optical flow serves particularly as an input for many other tasks such as navigation, object tracking, or image registration. In the estimation of flow fields, certain image regions are particularly challenging due to task-inherent difficulties such as illumination changes and occlusions as well as common prediction mistakes, *e. g.* for large displacements or near motion boundaries. Therefore, the reliability of optical flow estimates varies heavily across the image domain.

The first part of this thesis thus focuses on probabilistic optical flow methods, which predict a posterior distribution over the flow field conditioned on the input images. The first proposed method obtains probabilistic estimates by using variational inference to approximate a posterior derived from energy-based optical flow formulations. With *ProbFlow*, a fully probabilistic optical flow approach shows for the first time competitive results on popular benchmark datasets. The model-inherent confidence measure performs superior in comparison to previous work and the uncertainties are beneficially applied to improve optical flow estimates and a subsequent motion segmentation.

In a follow-up work, *SVIGL* is developed to combine stochastic approaches for variational inference with gradient linearization – a frequently used procedure in optical flow energy methods due to its good optimization properties. SVIGL shows faster convergence and higher robustness than standard approaches for stochastic variational inference of complex posteriors. Moreover, it provides probabilistic optical flow without the tedious derivation of update equations required in ProbFlow while maintaining comparable performance.

Although confidence measures detect unreliable regions, they do not directly improve the estimated flow fields. The second part of this thesis thus targets the refinement of optical flow in the context of neural networks. Here, the input images guide the post-processing as they provide valuable information about the structure of correct predictions. The first approach builds on an existing method for image-adaptive convolutions in a high-dimensional space. This space is spanned by feature dimensions that are now learned from data to improve the concept of pixel similarity used in the filtering operation. When applying the so-called *semantic lattice* to replace the bilinear upsampling step of state-of-the-art deep networks, one sees a clear improvement of the predictions, in particular at motion boundaries.

In the last contribution, the two goals of this thesis are combined and per-pixel confidence estimates are leveraged for the image-adaptive

refinement of deep optical flow predictions. As such, the proposed *probabilistic pixel-adaptive convolutions* (PPACs) do not only weigh pixels in a neighborhood according to learned similarity characteristics but also based on their individual reliability. The proposed PPAC refinement networks lead to substantial improvements in comparison to the underlying optical flow estimates. The obtained results are state-of-the-art on several benchmarks and show smooth flow fields with crisp boundaries as well as improved results in unreliable regions.

ZUSAMMENFASSUNG

Die Schätzung des optischen Flusses, also die Vorhersage der Bewegung in einer Bildsequenz, ist ein grundlegendes Problem im Bereich der Computer Vision. Die Information über den optischen Fluss wird insbesondere als Grundlage für verschiedene weitere Aufgaben verwendet, wie etwa Navigation, Objektverfolgung oder Bildregistrierung. Bei der Schätzung von Flussfeldern sind einige Bildregionen besonders herausfordernd – entweder wegen aufgabenspezifischer Schwierigkeiten wie Beleuchtungsänderungen und Verdeckungen oder aufgrund häufiger Schätzfehler, beispielsweise bei großen Verschiebungen und nahe am Übergang zwischen unterschiedlichen Bewegungen. Über das Bild betrachtet kann die Verlässlichkeit einer Schätzung des optischen Flusses somit stark variieren.

Der erste Teil dieser Arbeit konzentriert sich daher auf probabilistische Methoden für optischen Fluss, welche eine A-Posteriori Verteilung über das Flussfeld bedingt auf den Eingabebildern vorhersagen. Die erste Methode nutzt Variationsinferenz, um eine A-Posteriori Verteilung zu approximieren, welche von energiebasierten Flussmodellen abgeleitet ist. Mit *ProbFlow* zeigt dabei ein vollständig probabilistischer Ansatz zum ersten Mal kompetitive Ergebnisse auf populären Benchmark Datensätzen. Das modell-inhärente Konfidenzmaß zeigt sich überlegen gegenüber früheren Arbeiten und die Unsicherheiten können gewinnbringend zur Verbesserung des optischen Flusses sowie einer nachfolgenden Bewegungssegmentierung eingesetzt werden.

In einer Folgearbeit wird *SVIGL* entwickelt, um stochastische Ansätze für Variationsinferenz mit Gradientenlinearisierung zu kombinieren – ein Verfahren, welches aufgrund seiner guten Optimierungseigenschaften häufig bei Energiemethoden für optischen Fluss eingesetzt wird. SVIGL zeigt eine schnellere Konvergenz und eine höhere Robustheit als Standardansätze bei der stochastischen Variationsinferenz komplexer A-Posteriori Verteilungen. Zusätzlich ermöglicht SVIGL die Bestimmung eines probabilistischen optischen Flusses mit gleichbleibend guten Ergebnissen aber ohne die aufwändige Ableitung von Updateschritten, welche für ProbFlow erforderlich sind.

Obwohl Konfidenzschätzungen in der Lage sind, unzuverlässige Bereiche zu erkennen, verbessern sie nicht direkt die geschätzten Flussfelder. Der zweite Teil dieser Arbeit befasst sich daher mit der Verfeinerung optischer Flussvorhersagen im Kontext neuronaler Netze. Hierbei wird die Nachbearbeitung des Flusses durch die Eingabebilder beeinflusst, da diese wertvolle Informationen über die Struktur korrekter Vorhersagen beinhalten. Der erste Ansatz basiert auf einer bestehenden Methode für bildadaptive Faltungen in einem hochdimensionalen Raum, welcher von unterschiedlichen Merkmalsdimensionen

aufgespannt wird. Diese werden in der vorliegenden Arbeit aus Daten gelernt, um das bei der Filterung verwendete Konzept der Pixelähnlichkeit zu verbessern. Der resultierende *semantic lattice* wird daraufhin angewandt, um das einfache bilineare Upsampling moderner neuronaler Netze zu ersetzen. Dadurch zeigt sich eine klare Verbesserung der Schätzungen, insbesondere an Bewegungskanten.

Im letzten Beitrag werden die beiden Ziele dieser Arbeit kombiniert und die Konfidenzschätzung an jedem Pixel zur bildadaptiven Verfeinerung optischer Flussvorhersagen genutzt. Hierfür werden Pixel in einer Nachbarschaft durch die vorgeschlagenen *probabilistic pixel-adaptive convolutions* (PPACs) nicht nur abhängig von gelernten Ähnlichkeitsmerkmalen gewichtet, sondern auch basierend auf ihrer individuellen Zuverlässigkeit. Die beschriebenen PPAC-Netzwerke führen zu wesentlichen Verbesserungen im Vergleich zu den zugrunde liegenden Schätzungen des optischen Flusses. Die erzielten Ergebnisse sind state-of-the-art auf mehreren Benchmarks und zeigen gleichmäßige Flussfelder mit scharfen Bewegungsgrenzen sowie verbesserten Schätzungen in unzuverlässigen Regionen.

ACKNOWLEDGMENTS

First of all, I would like to thank my supervisor Stefan Roth. Thank you for giving me the chance to work in Computer Vision, for supporting and supervising me in numerous technical discussions, for having the right ideas and intuitions when needed and for teaching me so many different things. Thanks Margret for giving ProbFlow the push it needed and your valuable advice on how to navigate academia. Many thanks to Peter, Martin, and the remaining Amazonians for an interesting, insightful and overall unforgettable summer in Tübingen. May the adventure continue.

Without the members of the Visual Inference Lab, my time in Darmstadt would not have been the same. Danke an Nicole und Horst für Euren unermüdlichen Einsatz und Eure Hilfe in allen Lebenslagen! Thanks to Faraz for our hours of interesting conversations, to my flow buddy Jun, to my lovely birthday twin Shweta, and to Nikita who always kept his smile. Danke an Tobias, Stephan und Jochen für Euer Wissen, unsere vielen Diskussionen und dafür, dass Ihr mich so oft davon überzeugt habt, nicht aufzugeben.

Ich danke meinen Eltern und meiner Schwester. Danke, dass Ihr die Neugier in mir geweckt und ertragen habt, immer an meiner Seite seid und vor allem genau dann einen klaren Kopf habt, wenn meiner mal (wieder) verloren geht.

Ohne Dich wäre ich wahrscheinlich irgendwann verrückt geworden. Danke, Jonas.

CONTENTS

LIST OF FIGURES

ACRONYMS

AEE average end-point error

KL Kullback-Leibler

MAP maximum a-posteriori

PAC pixel-adaptive convolution (Su et al., 2019)

PPAC probabilistic pixel-adaptive convolution (Wannenwetsch and Roth, 2020)

SGD stochastic gradient descent

SVI stochastic variational inference

1 INTRODUCTION

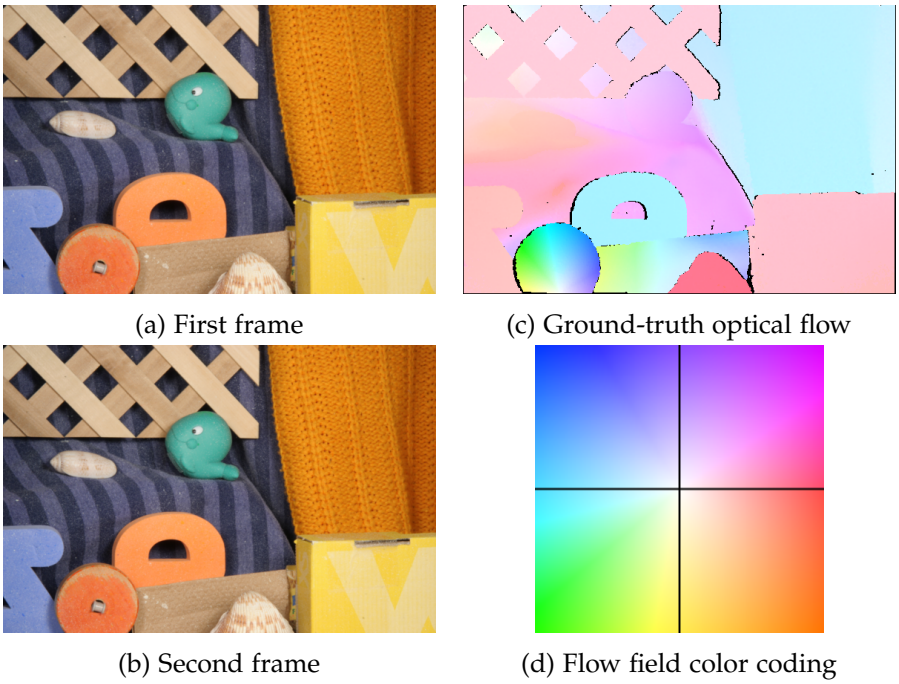
CONTENTS

1.1 OPTICAL FLOW ESTIMATION

Motion and moving objects are an essential part of our day to day lives. Not only we are moving in our environment but objects around us are equally dynamic. Therefore, recognizing motion is an essential ability of humans to detect objects of interest, avoid collisions, interact with the environment, and anticipate the behavior of others.

At first sight, the corresponding task of motion estimation from image data might seem simple but it remains, up until now, a challenging problem in computer vision. First, motion corresponds to a change in the scene such that one has to consider a sequence of inputs and match the frames appropriately. Here, motion causes parts of the scene to get occluded by other objects, which complicates the matching. Moreover, the task of motion estimation is – as many other tasks in computer vision – ill-defined: the motion happens in three-dimensional space while cameras provide us only with two-dimensional image data.

The following thesis especially targets the task of *optical flow prediction*. In the fundamental work of Horn and Schunck (1981), optical flow is defined as "the distribution of apparent velocities of movement of brightness patterns in an image" (Horn and Schunck, 1981, p. 185). More generally, an optical flow field can be understood as the two-dimensional motion between two images, which might be caused by movements of the observer or moving objects in the scene. A visualization of the corresponding task can be found in Fig. 1.1.



(a) First frame (c) Ground-truth optical flow

(b) Second frame (d) Flow field color coding

Figure 1.1. Optical flow estimation. *Left:* Input image pair from the Middlebury *RubberWhale* sequence (Baker et al., 2007). Both input frames are captured at different time steps leading to changes and thus motion in the scene. *Right:* Expected output corresponding to the ground truth optical flow field of the shown input images. The optical flow visualization uses a color coding in which color values indicate the direction of a motion vector and the intensity represents its magnitude.

APPLICATIONS. Over the years, optical flow has found widespread application in different fields and benefited various other tasks. For instance, optical flow estimates are frequently leveraged as an input cue for many computer vision tasks such as motion segmentation (Anthwal and Ganotra, 2019), object tracking (Li et al., 2013; Yilmaz et al., 2006), and action recognition (Jhuang et al., 2013; Sevilla-Lara et al., 2018; Simonyan and Zisserman, 2014). Moreover, the field of robotics benefits greatly from optical flow, especially for navigation purposes (Bonin-Font et al., 2008; Chao et al., 2014; Yousif et al., 2015). The same holds for the area of medical image analysis where optical flow is, among other things, used for image registration (Oliveira and Tavares, 2014). Finally, optical flow can be beneficially applied in the processing of videos, *e. g.* in video restoration (Bar et al., 2007; Buades et al., 2016; Werlberger et al., 2011) or video compression (Lu et al., 2019; Rippel et al., 2019; Wu et al., 2018a).

All tasks mentioned above thus benefit from an accurate and reliable optical flow prediction. The present thesis can be seen as a puzzle piece of the challenging task to fulfill this requirement.

DEFINING OPTICAL FLOW. For the following thesis, I will first define a consistent notation and introduce the term optical flow more formally. Therefore, let $I = \{I_1, I_2\}$ denote the pair of input images including the first and the second frame.[1] I further assume that the first image I_1 includes pixels (i, j) for $i = 1, \ldots, n$ and $j = 1, \ldots, m$. The corresponding optical flow then represents the *motion field* between I_1 and I_2. In the following, optical flow will be denoted as

$$\mathbf{y} = (\mathbf{y}_{ij})_{ij} = (u_{ij}, v_{ij})_{ij}^{\mathrm{T}}. \tag{1.1}$$

Here, the two-dimensional velocity field is split into its horizontal and vertical components $\mathbf{u} = (u_{ij})_{ij}$ and $\mathbf{v} = (v_{ij})_{ij}$, respectively.

An important characteristic is the fact that optical flow fields are defined *w. r. t.* the first image, *i. e.* dense optical flow methods provide a motion estimate for all pixels in I_1. Therefore, the optical flow field can be equally understood as a property that connects all pixels $(i, j) \in I_1$ with their target locations in I_2 given as

$$(i', j') = (i + u_{ij}, j + v_{ij}). \tag{1.2}$$

As such, (i', j') represents the spatial location to which a pixel (i, j) is moving in between the input frames I_1 and I_2.

As defined above, pixels (i, j) are located on the uniform grid that constitutes image I_1. However, the same does not generally hold for positions (i', j') and the discrete grid of the second image I_2. An illustrative example is the case in which the two images I_1 and I_2 are related by a zooming operation. If we zoom out between the frames, the set of locations (i', j') does not cover the entire second frame but corresponds only to a subset of the image values in I_2. The fact that pixels can be compressed to a smaller target area also illustrates another important property: optical flow is subpixel accurate. This means that pixels of the first frame can be connected to points (i', j') that are located in between the discrete pixel grid of image I_2.

OPTICAL FLOW CONSTRAINTS. As described above, optical flow estimation needs to find corresponding image locations in a sequence of images I. One fundamental assumption that thus underlies most optical flow approaches is the fact that corresponding locations in I_1 and I_2 should have a similar appearance. This certainly does not hold for all pixels since there exist lighting changes, occlusions, and similar. However, such pixel locations are generally perceived as outliers, which violate the overall similarity assumption.

While different properties can be leveraged to define appearance, optical flow was traditionally assumed to satisfy a brightness constancy constraint for all pixel locations $(i, j) \in I_1$, which can be expressed as

$$I_1(i, j) = I_2(i', j') = I_2(i + u_{ij}, j + v_{ij}) \tag{1.3}$$

[1] While there exist approaches that use additional frames as inputs, I will restrict myself to the basic case of two frames in this thesis.

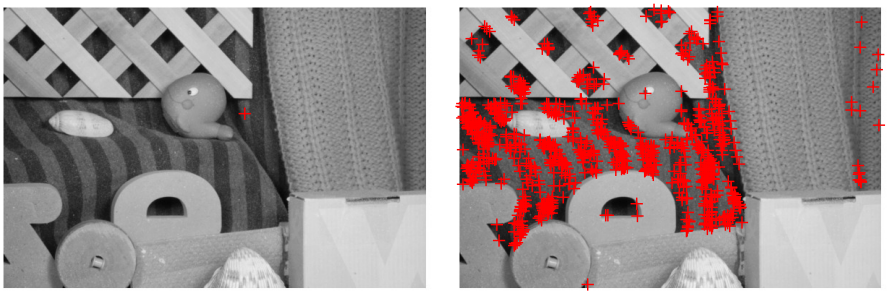


Figure 1.2. Ambiguity of per-pixel brightness constancy. *Left:* A pixel is randomly selected in the first gray scale image of the *RubberWhale* sequence from the Middlebury dataset (Baker et al., 2007). *Right:* Several locations in the second frame exhibit the same gray value as the selected pixel and thus satisfy a brightness constancy constraint as in Eq. (1.3).

with gray scale images I_1 and I_2. However, using such a constraint in an optimization approach is difficult in practice (Fortun et al., 2015).

Therefore, classic methods often assume sufficiently small motion and apply the linearized version

$$I_x \cdot u_{ij} + I_y \cdot v_{ij} + I_t = 0 \tag{1.4}$$

of the brightness constancy assumption. Here, I_t is the temporal difference at location (i, j) and I_x as well as I_y correspond to the partial image derivatives in x- and y-directions, respectively.

Unfortunately, for many pixels in I_1, per-pixel similarity constraints as the one in Eq. (1.3) and the linearized version in Eq. (1.4) can be satisfied by different locations in I_2, *cf.* Fig. 1.2. This means that there often exist multiple optical flow fields that satisfy the requirement of similar appearance. As such, the optical flow task is inherently ill-posed and additional conditions are generally required in order to determine an appropriate optical flow field (Poggio et al., 1985).

1.1.1 *Short Historical Overview*

Optical flow research has been performed for four decades and the corresponding literature is vast. To provide some background on the topic, I will summarize the most important and relevant steps in the following but refer to (Fortun et al., 2015; Janai et al., 2019; Sun, 2012) and the references therein for a broader overview.

LOCAL APPROACHES. Early works on optical flow were differential methods taking into account the linearized version of the brightness constancy as described in Eq. (1.4). The approach by Lucas and Kanade (1981) additionally assumes constant optical flow over a small neighborhood leading to several equations per flow estimate. The resulting equation system can then be solved by using least squares if the corresponding system matrix is invertible, *e. g.* in textured regions with

non-zero image gradients. So-called local methods strongly depend on the choice of a neighborhood, which was, for instance, mitigated by the usage of adaptive image regions (Black and Jepson, 1996; Bouthemy and Francois, 1993; Kanade and Okutomi, 1994; Maurizot et al., 1995). Similarly, the spatial constancy requirement can be replaced by the assumption of a less restrictive but still parametric form of optical flow, *e. g.* an affine motion model (Bergen et al., 1992). Nowadays, even the original Lucas-Kanade approach is still popular due to its simplicity and speed (Fortun et al., 2015).

GLOBAL APPROACHES. Alternatively to local methods, Horn and Schunck (1981) proposed to assume smooth optical flow, *i. e.* small changes of the flow field between neighboring pixels, in addition to a brightness constraint. Both assumptions are combined in a so-called global energy function

$$E(\mathbf{y}; I) = E_D(\mathbf{y}; I) + \lambda E_S(\mathbf{y}). \tag{1.5}$$

Here, the data term $E_D(\cdot)$ enforces the linearized brightness constancy assumption in Eq. (1.4) and $E_S(\cdot)$ imposes the aforementioned spatial smoothness term. The scalar factor λ allows a trade-off between both components. An optical flow estimate $\mathbf{y}^\star$ is then obtained as the solution of the minimization of $E(\mathbf{y}; I)$, *i. e.*

$$\mathbf{y}^\star = \arg\min_y E(\mathbf{y}; I). \tag{1.6}$$

Over the years, many adjustments to the original energy formulation have been proposed. For instance, Black and Anandan (1996) replaced the quadratic penalties in $E_D(\cdot)$ and $E_S(\cdot)$ to robustly penalize violations of model assumptions and avoid over-smoothing of flow fields. The data term was improved, *e. g.* by pre-processing the underlying images (Sun et al., 2014; Wedel et al., 2009) or by using more advanced consistency assumptions (Brox et al., 2004; Stein, 2004; Vogel et al., 2013). The same holds for the smoothness term, which was – among other things – adapted to the underlying image (Alvarez et al., 1999; Nagel and Enkelmann, 1986; Sun et al., 2008) or reformulated as a higher-order term (Braux-Zin et al., 2013; Trobin et al., 2008). Moreover, energy functions complementing the original data and smoothness components have been proposed, *e. g.* terms considering extended non-local regions (Sun et al., 2014; Werlberger et al., 2010) or symmetries between forward and backward motion (Alvarez et al., 2007).

ENERGY OPTIMIZATION. Even for relatively simple model assumptions enforced in the energy function, the corresponding optimization is challenging (Fortun et al., 2015; Sun et al., 2014). For instance, the linearization of the data term as shown in Eq. (1.4) is only valid for small motions and the optimization objectives are generally highly

complex. To counteract problems due to large motion, coarse-to-fine warping approaches (Anandan, 1989; Black and Anandan, 1996; Brox et al., 2004) were introduced. These procedures iteratively estimate optical flow on differently downscaled versions of the input images and successively refine the predictions. Moreover, methods denoted as iterative gradient linearization (Nikolova and Chan, 2007) or fixed point iteration schemes (Brox et al., 2004) allow to deal with the fact that energy gradients are often non-linear *w.r.t.* to the optical flow. Here, the corresponding terms are linearly approximated in an iterative scheme, *e.g.* non-linear terms are fixed to the values of the previous optical flow estimate. In each iteration, the resulting linear system of equations can then be solved with common approaches. Despite the rather complex optimization, global energy methods are still used due to their high flexibility and good accuracy.

CORRESPONDENCE APPROACHES. Another line of research leverages pixel correspondences for optical flow estimation, *e.g.* obtained by sparse descriptor matching (Brox and Malik, 2011; Revaud et al., 2015; Xu et al., 2012), nearest neighbor search (Bailer et al., 2015; Chen et al., 2013), or the discrete optimization of global flow objectives (Boykov et al., 2001; Chen and Koltun, 2016; Steinbrücker et al., 2009). Such methods are especially beneficial for large displacements, which are not covered by the small motion assumption of differential methods. Moreover, pixel matches help to recover motion of small objects, which are eliminated by the downsampling applied in common coarse-to-fine schemes. However, the resulting correspondences may be sparse and the corresponding optical flow is generally not subpixel accurate. As such, pixel matches were, for instance, integrated into energy minimization approaches (Brox and Malik, 2011; Xu et al., 2012) or used as initialization for a further refinement step (Bailer et al., 2015; Chen and Koltun, 2016; Chen et al., 2013; Revaud et al., 2015).

NETWORK APPROACHES. Most recently, deep learning based approaches have also revolutionized the field of optical flow estimation. While early results (Dosovitskiy et al., 2015) led to significantly lower runtimes, the flow accuracy could not keep up with state-of-the-art approaches. Since then, deep methods have clearly improved, for instance, by using complex (synthetic) datasets, elaborated training procedures, and advanced network structures (Ilg et al., 2017; Sun et al., 2018; Teed and Deng, 2020; Yin et al., 2019). Moreover, different unsupervised approaches learn deep optical flow without the supervision of ground truth flow fields, *e.g.* (Liu et al., 2019; Meister et al., 2018; Yu et al., 2016). As of now, such methods often require at least a small number of labeled data for fine-tuning to compete with supervised networks. Overall, most leading optical flow approaches

are nowadays based on deep neural networks, which equally lead to fast and accurate estimates.

1.1.2 *Quantitative Evaluation*

In order to observe the progress in optical flow research and keep track of different approaches, it is essential to have a comparable experimental evaluation including suitable quantitative measures. Therefore, it is particularly important to have optical flow ground truth, *i. e.* correct flow fields, to which estimates can be compared. For optical flow, ground truth is difficult to obtain as there are no task-specific sensors available and the data cannot be easily annotated by humans (Butler et al., 2012). As such, creating optical flow datasets with underlying ground truth has been an important topic of research over the last decades, *e. g.* (Baker et al., 2007; Barron et al., 1994; Butler et al., 2012; Dosovitskiy et al., 2015; Geiger et al., 2012; Janai et al., 2017; Kondermann et al., 2016; Mac Aodha et al., 2013; Richter et al., 2017).

DATASETS. Barron et al. (1994) were the first to quantitatively compare several optical flow approaches on different scenes, *e. g.* the well known *Yosemite* sequence. The test data was composed of real and synthetic input images, which can be considered as rather simple nowadays. Baker et al. (2007) presented the *Middlebury* dataset, which equally includes synthetic as well as real-world scenes. The online benchmark allows to easily compare results of new algorithms to previous approaches and ensures a fair comparison due to withheld test ground truth. However, the number of images is small – 8 sequences with ground truth for training and test each – such that its importance has slightly decreased over the last years. Butler et al. (2012) obtained the so-called *Sintel* dataset with more than 1600 images from an animated 3D movie. The dataset distinguishes between the clean and final version of the input data; the latter includes additional effects such as different kinds of blur to create more realistic scenes. Even though the dataset is frequently used, all images are synthetic and thus not fully comparable to real world scenarios. In contrast, the *KITTI 2012* dataset (Geiger et al., 2012) provides optical flow ground truth for recorded driving scenes. As the corresponding flow is obtained by a laser scanner, the available ground truth is sparse and does not include moving objects. The *KITTI 2015* dataset (Menze et al., 2018) introduces an additional set of images in which 3D CAD models are fitted to all moving vehicles to include their ground truth motion. Both KITTI datasets are comparably small including roughly 200 images for training and test, respectively. At present, the Sintel as well as the KITTI datasets are frequently used for evaluation of new approaches. See Fig. 1.3 for an illustration of sample sequences from those two optical flow benchmarks as well as from the Middlebury dataset.

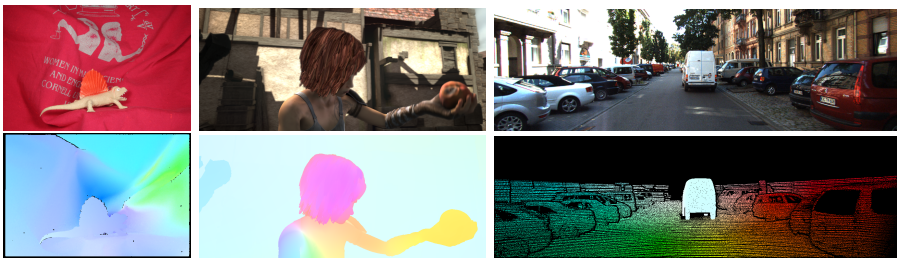


Figure 1.3. Illustration of different optical flow datasets showing the first image of a sample sequence *(top)* and the corresponding optical flow ground truth *(bottom)*. *Left:* Middlebury dataset (Baker et al., 2007). *Middle:* Sintel dataset (Butler et al., 2012). *Right:* KITTI 2015 dataset (Menze et al., 2018).

METRICS. Given accurate ground truth, there exist several error metrics to evaluate optical flow predictions but only two are used for the experimental evaluations presented in this thesis. Let $\mathbf{z} = (\mathbf{z}_k)_k$ be the ground truth for valid pixels $k = 1, \dots, V$ and $\mathbf{y} = (\mathbf{y}_k)_k$ the corresponding optical flow estimates. Then, the average end-point error (AEE) (Otte and Nagel, 1994) is given as

$$AEE(\mathbf{y}, \mathbf{z}) = \frac{1}{V} \sum_{k=1}^{V} \|\mathbf{y}_k - \mathbf{z}_k\|_2 \tag{1.7}$$

with $\|\cdot\|_2$ denoting the Euclidean norm. Here, the average is only taken over all valid pixels, *i.e.* pixels for which ground truth is available. The average end-point error metric is, for instance, used in the Middlebury as well as the Sintel dataset.

The KITTI datasets consider a pixel k as erroneous as soon as the end-point error exceeds a certain threshold, *i.e.* $\|\mathbf{y}_k - \mathbf{z}_k\|_2 > \tau$. The *outlier rate* then corresponds to the percentage of erroneous pixels using a threshold of $\tau = 3$ pixels as a default value (Geiger et al., 2012). On KITTI 2012, the main error metric is given by the outlier rate evaluated on non-occluded pixels only. For flow fields on KITTI 2015, the optical flow error of a pixel needs to additionally exceed 5% of the ground truth flow magnitude to be considered as incorrect (Menze et al., 2018). Moreover, KITTI 2015 uses the outlier rate averaged over all pixels as its main metric for comparison.

1.1.3 *Challenges in Optical Flow*

As illustrated in the previous sections, there is a long history of optical flow prediction. As such, one might wonder why research still struggles with this task. There are manifold answers to this question and I will summarize the most essential ones in the following.

AMBIGUITIES. As described before, optical flow algorithms commonly share the assumption that corresponding locations have similar appearance at both time steps. The most prominent problem in optical

flow estimation is thus the fact that motion fields are generally ambiguous. This holds especially in homogeneous areas where multiple pixels are visually similar. As such, an optical flow field cannot be uniquely determined based on pixelwise similarity, *i.e.* optical flow estimation is ill-posed. The underlying ambiguity may only be resolved by considering more than a single pixel at a time. This observation is strongly linked to the so-called *aperture problem*, which states that one can only estimate motion orthogonal to edges given a local observation of a one-dimensional spatial structure, *e.g.* a line. This means that there are several different flow fields that could lead to the same exact motion when observed with a small aperture.

ILLUMINATION CHANGES. Another fact that might severely violate appearance assumptions is the change of lighting conditions in a scene. For instance, objects might completely alter their appearance when moving from light into shadow. The same holds if reflections change their location in a scene, which severely affects the corresponding image values. For such cases, robust data features have been applied in classic approaches but often require additional computational time. Moreover, such methods are only able to reduce the problem of appearance change but cannot completely erase the issue.

OCCLUSIONS. Likewise, the similarity assumption does not hold for pixels that get occluded between the two frames, *i.e.* pixels that are only visible in one image but hidden in the other. Such pixels might still have non-zero motion but corresponding locations cannot be retrieved by visual comparison. One might estimate optical flow for these regions by either propagating the estimates from neighboring pixels or, for longer sequences, from neighboring frames. However, such a propagation is not straightforward and might thus result in erroneous predictions. The same problem exists for pixels that move out of the image boundary as their new position cannot be observed.

MOTION BOUNDARIES. Apart from occluded regions, motion boundaries in the flow field similarly challenge estimation algorithms. In most areas, optical flow varies smoothly but there exist sudden changes if close objects move differently. Here, the estimated transitions should be sharp and well aligned with the underlying content. However, the task to produce non-blurry motion boundaries is difficult for most approaches. Traditional methods explicitly assume spatially smooth optical flow, which is not satisfied in such regions. Moreover, motion boundaries correspond to a comparably small number of pixels and can further diminish in deep feature encoders. As such, deep network based approaches may equally show difficulties in boundary regions when learning the appearance of flow fields from data.

LARGE DISPLACEMENTS. Large displacements are another major source of estimation errors. An exhaustive search over all possible image locations is computationally infeasible for most cases. Instead, the search range is restricted or other simplifying assumptions, *e. g.* for small motion, have to be made. Here, every optical flow estimate for a location that surpasses such an implicit or explicit maximal displacement will be inevitably wrong. To reduce this problem, coarse-to-fine procedures are applied for classic as well as deep prediction approaches. The iterative estimates will, however, remain wrong if predictions at lower resolutions are slightly incorrect. Additionally, the applied downsampling operations eliminate smaller objects such that matching based approaches seem beneficial. However, the latter methods also often need to restrict their search range and show problems in homogeneous and especially occluded regions.

INFORMATION LOSS. Another problem of many optical flow algorithms is their inherent loss of information. On the one hand, traditional methods need to apply rather simple models to keep the corresponding optimization problem manageable. As such, energy functions cannot include exact representations for prior and posterior terms but remain a trade-off between the degree of approximation and their complexity. Similarly, pooling operations as well as convolutional layers are part of network architectures to keep deep learning approaches fast and trainable for large image sizes. However, such layers lead to information loss by mixing properties of differently moving objects. Additionally, neural networks often output optical flow at a resolution lower than the size of the input data and the subsequently applied bilinear upsampling further restricts their capacity.

MISSING GROUND TRUTH. As mentioned above, a problem that generally affects recent neural network approaches is the fact that only limited ground truth data is available for optical flow estimation. Deep networks are generally very data hungry and require a sufficiently large number of samples in order to tune their immense set of parameters. Synthetic data has been frequently used to attenuate this limitation but enlarges the gap between the applied training images and real-world test data. Here, unsupervised methods might be a solution since their learning is only based on image data.

Some of the above problems boil down to the fact how optical flow estimation is approached in practice, *e. g.* information loss in models. In contrast, other issues are inherent in the task itself, *e. g.* ambiguity or occlusion problems. The latter points severely challenge the optical flow prediction and can hardly be overcome completely by current algorithms. Instead, estimated flow fields remain uncertain for specific locations and the per-pixel reliability varies for different regions.

1.2 PROBABILISTIC OPTICAL FLOW

As described in the previous section, optical flow is inherently an uncertain estimation task and the reliability depends on the properties of a specific pixel. As such, it seems reasonable to include the underlying uncertainty into the estimation task itself as done by *probabilistic optical flow* approaches. Most optical flow methods provide a flow field that consists of a two-dimensional point estimate per pixel. In contrast, I refer to a probabilistic optical flow method as an algorithm that predicts a posterior distribution $p(\mathbf{y}|I)$ of the flow field $\mathbf{y}$ given the input sequence I. With an estimated posterior, there are different approaches to obtain an optical flow prediction per pixel, for instance, a Bayes estimator. In this case, an optical flow field $\mathbf{y}^\star$ is given as the minimizer of the expected loss over the estimated posterior

$$\mathbf{y}^\star = \arg\min_{\tilde{\mathbf{y}}} \mathbb{E}_{p(\mathbf{y}|I)}\, l(\mathbf{y}, \tilde{\mathbf{y}}) \tag{1.8}$$

with $l(\cdot,\cdot)$ denoting an appropriate loss function.

Here, global energy approaches correspond to a special case of the above described Bayesian risk minimization. An energy function $E(\mathbf{y}; I)$, as the one in Eq. (1.5), describes a Markov Random Field such that the corresponding posterior can be derived in its Gibbs form as

$$p(\mathbf{y}|I) = \frac{1}{Z(I, T, \lambda)} \exp\left\{-\frac{1}{T} E(\mathbf{y}; I)\right\} \tag{1.9}$$

with temperature T and partition function $Z(\cdot)$. Now, let $l(\cdot,\cdot)$ be the 0-1 loss function with $l(\mathbf{y}, \tilde{\mathbf{y}}) = \mathbb{1}\left[\mathbf{y} \neq \tilde{\mathbf{y}}\right]$ and indicator function $\mathbb{1}\left[\cdot\right]$. Then, one obtains the so-called maximum a-posteriori (MAP) estimate

$$\mathbf{y}^\star = \arg\max_{\tilde{\mathbf{y}}} p(\tilde{\mathbf{y}}|I) \tag{1.10}$$

as a solution of Eq. (1.8). This again corresponds to the commonly applied minimization of the energy function due to the definition of $p(\mathbf{y}|I)$. However, no explicit posterior distribution is predicted in this case, which is in contrast to probabilistic optical flow approaches.

Taking instead a fully probabilistic approach to optical flow has two main advantages: First, a probabilistic handling might improve the predictions as the probability can be taken into account during the estimation process. Second, the posterior distribution can be leveraged to assess the reliability of individual predictions after the estimation.

The history of probabilistic formulations for optical flow is almost as long as the one of general prediction algorithms. I will omit an historical overview by referring to the related work in our papers (Wannenwetsch and Roth, 2020; Wannenwetsch et al., 2017). Instead, this section focuses on the application of probabilistic optical flow in the context of *confidence measures*, which aim to classify pixels according to the marginal posterior of their estimates.

DEFINING THE TERM PROBABILITY. As a first step, let me thus clarify how the term *probability* is used in this thesis. For instance, probability might denote the full marginal posterior at every pixel. However, this property is generally unknown and can only be approximated in most cases. Thus, I follow our definition in (Wannenwetsch and Roth, 2020) and denote as *optical flow probability* more generally a measure that approximates or summarizes the full marginal posterior over the optical flow estimates, *e. g.* by using the corresponding density of the chosen optical flow value. In the following, the terms probability, confidence, and reliability are used interchangeably. Additionally, uncertainties are inversely related to probability values such that high values correspond to unreliable or uncertain estimates.

1.2.1 *Post-Hoc and Model-Inherent Confidences*

Naturally, there exist many approaches to confidence measures that do not necessarily predict approximate marginal distributions but estimate confidences differently. For instance, so-called *post-hoc* approaches generate reliabilities based on the input data, *e. g.* (Barron et al., 1994; Haußecker and Spies, 1999), the output flow fields, *e. g.* (Kondermann et al., 2007, 2008a), or both *e. g.* (Mac Aodha et al., 2013). Another common approach is to evaluate the forward-backward error of an optical flow estimate. Here, one assumes that a subsequent evaluation of corresponding forward and backward optical flow should again lead to the start position. Obviously, this approach always requires an additional estimation of backward optical flow from $I' = \{I_2, I_1\}$.

In contrast, *model-inherent* methods consider the estimation process itself for their reliability estimates. Confidence measures derived from the posterior of a probabilistic optical flow algorithms are a special case of model-inherent approaches since flow field and confidences both rely on the estimated posterior. Another model-inherent method, introduced by Bruhn and Weickert (2006), uses the same energy function for the prediction of optical flow and its confidences. Here, optical flow is obtained as a MAP estimate and the per-pixel contribution to the remaining energy value is leveraged as an uncertainty measure.

ADVANTAGES OF POST-HOC METHODS. Post-hoc as well as model-inherent methods are considerably different in their approach to assess uncertainties but both have their own benefits. First, post-hoc approaches are generally applicable to all kinds of algorithms as they do not consider how a certain estimate has been obtained. Additionally, such methods are often comparably simple to evaluate and can be turned on or off as required, *i. e.* uncertainties are only computed if needed. Most importantly, post-hoc confidence measures are able to detect errors due to incorrect or simplifying modeling assumptions and can, for instance, learn where certain approaches fail.

ADVANTAGES OF MODEL-INHERENT METHODS. In contrast, model-inherent methods do not necessarily require an additional training but corresponding uncertainties are often estimated together with the flow field. Moreover, such measures are especially tailored to the chosen prediction approach and the knowledge of uncertainties might be beneficial for the estimation of optical flow itself. The latter is especially the case if a probabilistic setup is chosen. However, one could, for instance, also imagine to re-weight per-pixel predictions based on their reliability in the smoothness term of energy-based, iterative methods. Finally, model-inherent approaches can leverage information of the estimation process, which provides valuable knowledge about pixels affected by occlusions, illumination changes, or similar.

In general, assessing the reliability of optical flow is far from easy and existing approaches have various different strengths and weaknesses. Therefore, the optimal choice of an uncertainty measure does not only depend on the underlying optical flow method but it is clearly conditioned on the situation in which the confidences are required.

1.2.2 *Applications of Confidence Measures*

In principle, all tasks that leverage optical flow cues as an input could benefit from the availability of an additional uncertainty estimate. Especially in safety-critical areas, measures to assess the reliability of optical flow estimates are desirable, *e. g.* for autonomous driving (Janai et al., 2019) or medical applications (Oliveira and Tavares, 2014). In order to further illustrate the broad applicability of optical flow uncertainties, I present a few concrete exemplary cases in which optical flow probabilities have shown to be beneficial in the past.

EGO-MOTION. Domke and Aloimonos (2007) estimate an optical flow probability distribution based on Gabor filters for each pixel. The resulting correspondence distributions are subsequently used in a probabilistic framework for ego-motion prediction. The authors observe that the probabilistic treatment allows to extract more correspondence information from the images, which again leads to more accurate ego-motion estimates in their approach.

MULTI-FRAME SUPER-RESOLUTION. In (Kanaev and Miller, 2013), optical flow is used in multi-frame super-resolution to warp low-resolution images onto a reference frame. The energy minimization applied to the super-resolution task weighs pixels differently according to their optical flow reliability. Here, the used confidence measure is based on the warping error between corresponding images.

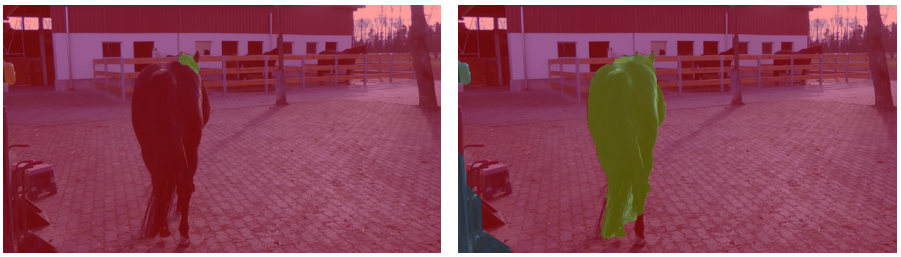


Figure 1.4. Motion segmentation with optical flow confidences. *Left:* A motion segmentation on a sample image of the FBMS-59 dataset (Ochs et al., 2014) is obtained by the method introduced in (Keuper et al., 2015) using FlowFields optical flow (Bailer et al., 2015) as input cue. *Right:* The motion segmentation using the same approach is clearly improved when leveraging optical flow as well as corresponding uncertainties from ProbFlowFields (Wannenwetsch et al., 2017). See Section 2.1 and (Wannenwetsch et al., 2017) for details.

MOTION BOUNDARIES. Weinzaepfel et al. (2015) target the prediction of motion boundaries by leveraging image and optical flow information as inputs to a structured random forest. Additionally, confidences for the optical flow estimates are included in the form of image warping errors evaluating color as well as gradient constancy assumptions. Based on the presented ablation study, the authors conclude that optical flow reliability cues are beneficial for the estimation of motion boundaries in challenging datasets.

MOTION SEGMENTATION. In (Brox and Malik, 2010; Keuper et al., 2015), optical flow is applied to construct long-term point trajectories for a subsequent motion segmentation. Here, forward-backward consistency as well as a confidence measure based on the optical flow gradient are used to stop trajectories as soon as unreliable estimates are detected. In our paper (Wannenwetsch et al., 2017), we show that this gradient-based criterion can be successfully replaced by more advanced uncertainties, *cf.* Fig. 1.4 for an illustration.

1.3 OPTICAL FLOW REFINEMENT

In Section 1.1.3, I have summarized several types of regions in which optical flow estimation is especially challenged, *e. g.* areas close to motion boundaries. With the help of confidence measures, it is possible to detect pixels that have a high probability for incorrect predictions. While it is certainly helpful for subsequent applications to know locations of low reliability, this knowledge alone does not change the incorrect optical flow estimates. Instead, approaches that use motion cues as an input would certainly also benefit from an improved accuracy of the optical flow field in difficult regions. Sevilla-Lara et al. (2018) actually showed for an optical flow method applied in action recognition that correct estimates at motion boundaries are of high importance for the task-specific performance. Similarly, well aligned

and sharp optical flow edges might be helpful for methods that base their estimates on motion boundaries, *e. g.* for object segmentation in videos (Papazoglou and Ferrari, 2013).

As such, one might wonder how we can improve optical flow predictions either based on prior knowledge of difficult regions or due to available confidence estimates. For instance, an explicit refinement of optical flow might allow to sharpen edges in the optical flow field, which are frequently blurred by traditional as well as neural network methods. Similarly, reliable estimates could be propagated into occluded regions or to pixels that are affected by local illumination changes. Moreover, the removal of noise or outliers in the flow field would clearly improve the overall quality of the predictions.

1.3.1 *Overview of Refinement Methods*

In fact, there exist many procedures that allow to refine and improve optical flow estimates in erroneous regions. Various algorithms are *image-adaptive*, which means that they take the input image sequence into account for their refinement. Here, one could also imagine that other information, *e. g.* a segmentation of the images, can be leveraged to improve the predictions. Therefore, additional information used to guide the refinement will be more generally denoted as *guidance data* in the following. I will now summarize the most important concepts of optical flow refinement, again with no claim to completeness.

INPAINTING. The first group of refinement methods is based on inpainting or interpolation approaches, which replace missing predictions. Here, one can determine regions that are likely to be erroneous by using confidence measures (Berkels et al., 2009; Kondermann et al., 2008b), occlusion estimation (Ince and Konrad, 2008), forward-backward consistency tests (Bailer et al., 2015; Chen and Koltun, 2016), or similar. Subsequently, the corresponding predictions are erased and replaced based on the remaining flow field entries and the chosen interpolation scheme. The same approaches can be leveraged if optical flow is only sparsely estimated, *e. g.* due to sparse descriptor matches.

Different methods (Berkels et al., 2009; Ince and Konrad, 2008; Kondermann et al., 2008b) build optical flow inpainting on energy formulations, which may also be adaptive to edges of the underlying input images (Berkels et al., 2009; Ince and Konrad, 2008). Moreover, a frequently applied method to determine missing optical flow values is the EpicFlow interpolation approach (Revaud et al., 2015). Here, an edge-preserving distance measure provides nearest neighbors for a missing pixel and the optical flow value is subsequently obtained with a locally-weighted affine motion estimator. Li et al. (2016) propose a hierarchical, iterative interpolation approach based on weighted least

squares. The method uses not only image edges but the full color information of the underlying image sequence as guidance data.

In general, inpainting-based refinement includes a decision which pixels are unreliable and should thus be replaced. However, such a binary choice is rather restrictive and might be difficult in practice.

FILTERING. Another common approach to improve the results is a filtering step applied to the estimated optical flow. Such approaches do not necessarily rely on the confidences of individual pixels but rather base the filtering on the knowledge of common estimation mistakes.

For instance, median filtering is frequently applied in energy-based methods to remove outliers of the flow field (Bartolini and Piva, 1997; Wedel et al., 2009). Here, the median filter acts similar to an extended, non-local smoothness term (Sun et al., 2014). In (Xiao et al., 2006), a data-based energy minimization step is alternated with a modified version of bilateral filtering for an image-adaptive regularization of the flow field. Subsequently, bilateral filtering was also applied as a refinement step for optical flow, *e. g.* (Mozerov, 2013; Xu et al., 2012).

Both above approaches are combined in weighted non-local energy terms (Sun et al., 2014; Werlberger et al., 2010), which use spatial and color similarity of pixels to perform a kind of image-adaptive median filtering. In (Hosni et al., 2013), a separate weighted median filter is proposed for the post-processing of unreliable optical flow values.

A general disadvantage of filtering-based approaches is the possible propagation of errors in the optical flow field since incorrect estimates can be easily spread to surrounding pixels.

ENERGY MINIMIZATION. Nowadays, most state-of-the-art predictions are not entirely based on energy models anymore. However, it is not uncommon to refine the optical flow with a few iterations of energy minimization at full resolution, *e. g.* employing the popular EpicFlow energy (Revaud et al., 2015). Similarly, optical flow can be refined by other image-aware optimization approaches, *e. g.* the Fast Bilateral Solver (Barron and Poole, 2016). One interesting characteristic of the latter approach is the fact that confidences of individual estimates are used during the post-processing step to determine how close a refined estimate should remain to its initial prediction.

Energy minimization refinement is especially applied to flow fields that are obtained through matching approaches and, if required, a subsequent interpolation (Bailer et al., 2015; Chen and Koltun, 2016; Chen et al., 2013; Revaud et al., 2015). In contrast to the pixel matches, the energy optimization allows for subpixel accurate optical flow predictions. However, such methods remain restricted to the predefined energy function and might be computationally expensive.

NEURAL NETWORKS. Neural networks are by now a commonly used tool for optical flow estimation. As such, refinement approaches based on deep network layers have gained importance as they can be trained in an end-to-end fashion when added to the architecture.

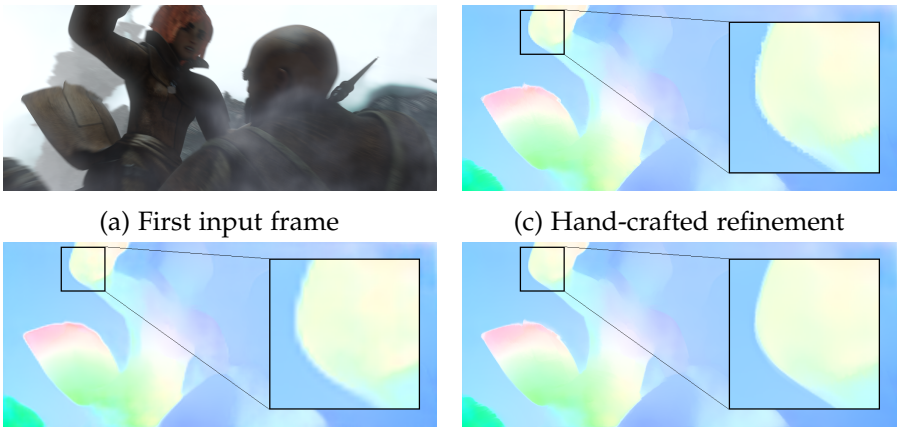
Ilg et al. (2017) propose to stack several networks to refine optical flow estimates. Therefore, the later networks are provided with the warped second image, the current predictions as well as their brightness errors. Context networks, as used in (Sun et al., 2018; Yin et al., 2019), are applied for optical flow refinement since the included dilated convolutions increase the receptive field and thus the contextual information. Moreover, content-adaptive convolutions, *e. g.* (Hui et al., 2018; Hur and Roth, 2019; Su et al., 2019), have shown to be beneficial as part of the optical flow decoder to obtain smooth estimates with crisp image boundaries. Here, the explicit usage of input images as guidance data allows to – at least partly – compensate for the information loss in networks or to improve the simple bilinear upsampling step commonly used to enlarge the low-resolution network outputs.

A disadvantage of many network-based refinement approaches is the large number of additional parameters, which might complicate training in practice. Please see the corresponding related work sections in (Wannenwetsch and Roth, 2020; Wannenwetsch et al., 2019) for a more detailed review of neural network refinement.

1.3.2 *Benefits of Learned Approaches*

Many of the approaches shown in the previous section are based on rather restrictive assumptions. For instance, the relation between image and optical flow edges is often modeled in a simple way, *e. g.* by assuming that edges detected in image data coincide with the ones of optical flow (Berkels et al., 2009; Ince and Konrad, 2008; Revaud et al., 2015). Additionally, pre-defined energy functions, *e. g.* (Barron and Poole, 2016; Revaud et al., 2015), and hand-crafted pixel similarity measures, *e. g.* (Hosni et al., 2013; Sun et al., 2014; Xiao et al., 2006), are applied to refine predictions or propagate estimates across the image.

Even though the mentioned methods have shown good results in practice, there are reasons to argue that more flexible, learned models provide advantages for the refinement of optical flow. First, human intuition about the meaningfulness of certain representations can be suboptimal. For instance, better similarity features than the RGB values of a guidance image might be available to define the closeness of pixels. This is in line with the observation that specifically learned image features have shown to be beneficial for matching-based optical flow (Gadot and Wolf, 2016; Güney and Geiger, 2016). Additionally, more complex procedures might be learned if no modeling by hand is required. Here, an advanced filter applied to the initial optical flow might improve over the commonly used Gaussian formulation.



(a) First input frame (c) Hand-crafted refinement

(b) Optical flow from PWC-Net (d) Learned refinement

Figure 1.5. Optical flow refinement with learned components. *(a):* First frame of a sequence included in the Sintel dataset (Butler et al., 2012). *(b):* An optical flow field is estimated with PWC-Net (Sun et al., 2018). *(c):* Optical flow can be refined with the semantic lattice (Wannenwetsch et al., 2019) using hand-crafted components. *(d):* Optical flow refinement is improved with the semantic lattice including components learned from data. See Section 2.3 and (Wannenwetsch et al., 2019) for further details.

Finally, approaches that are learned from data can be easily adapted to particular problems of a certain method or to a specific dataset.

To conclude, the usage of more flexible, learned frameworks shows great potential for a further improvement of current optical flow refinement methods. This assumption is experimentally analyzed in (Wannenwetsch et al., 2019), which will be further described in Section 2.3. A visualization of the obtained results is shown in Fig. 1.5. Nevertheless, current learning approaches are not always easy to apply since they often require a large number of samples and a non-negligible amount of training. This is an important aspect that should not be ignored but taken into account when designing a new method to refine and improve optical flow predictions.

2

PAPERS AND CONTRIBUTIONS

CONTENTS

2.1 PROBFLOW: JOINT OPTICAL FLOW AND UNCERTAINTY ESTIMATION

This section summarizes the paper

> Anne S. Wannenwetsch, Margret Keuper, and Stefan Roth
> *ProbFlow: Joint Optical Flow and Uncertainty Estimation*,

which was published in *2017 IEEE International Conference on Computer Vision (ICCV)*.

2.1.1 *Motivation*

As described in Section 1.2.2, uncertainty measures are an important source of information in addition to the actual optical flow estimates. Such measures are especially interesting for tasks that use optical flow as an input and generally assume reliable flow field predictions.

Over the years, different post-hoc approaches (Barron et al., 1994; Kondermann et al., 2008a; Mac Aodha et al., 2013) were proposed but do not leverage information of the optical flow estimation procedure. Instead, the confidence prediction is restricted to a one-shot process and an additional, measure-specific training might be needed. We thus aim for a model-inherent approach and accept possible challenges to detect errors associated with the chosen prediction method, *cf.* Section 1.2.1. Here, one should keep in mind that optical flow algorithms are generally chosen with care for the area of interest such that the corresponding modeling errors remain small. In contrast, the information available in the estimation procedure is of great value and should not be neglected to assess the reliability of optical flow predictions.

More broadly, our goal is to construct a fully probabilistic optical flow setup, which simultaneously estimates optical flow and a confidence measure based on the predicted posterior distribution. At the same time, the flow field estimates should be as accurate as current optical flow methods and the obtained uncertainties should perform equally well or better than competing confidence approaches.

The presented paper thus introduces a framework built on energy-based optical flow models. For a chosen energy, we easily obtain the optical flow posterior p as the corresponding Gibbs distribution shown in Eq. (1.9). We can then get an optical flow prediction as the result of the Bayesian risk minimization in Eq. (1.8) and aim to apply the AEE as a loss function since it is better suited for optical flow than the commonly used 0-1 loss. Additionally, a powerful uncertainty measure should be derived based on the marginal posterior at each pixel.

2.1.2 *Proposed Method and Findings*

Given the complexity of optical flow energies, exact posterior inference – to obtain flow fields and uncertainties as described above – is generally intractable. As such, we propose to fall back to a variational approximation of the original posterior p. We further use a naive mean-field assumption leading to a fully-factorized approximation distribution q. Optical flow components are assumed to be marginally distributed as uncorrelated Gaussians with diagonal covariance matrices. Moreover, we model the penalty functions of the energy terms as the negative logarithm of different Gaussian Scale Mixtures. To avoid difficulties when estimating the parameters of q, we further use explicit latent variables for each of the mixture components. We

assume a multinomial distribution for latent variables and equally update their parameters during the variational optimization.

Using the approximating distribution q, we show in the supplemental material that a minimization of the expected loss over q (*cf.* Eq. (1.8)) leads to optical flow predictions that correspond to the mean values of its Gaussian components. We further propose an uncertainty measure based on the estimated Gaussian variances.

However, the optimization of the Kullback-Leibler (KL) divergence $D_{KL}(q \,\|\, p)$ between the original distribution p and the approximating posterior q is far from straightforward. We thus use a custom block-coordinate descent scheme, which alternatingly updates the mean and the variance parameters of the Gaussian distributions as well as the latent variables. Here, we found it especially important to use closed-form update equations for all types of variables and to simultaneously update optical flow estimates for all pixels. Additionally, best practices of optical flow, *e. g.* pre-processing and gradient averaging of input images, were equally important in a probabilistic framework.

Keeping in mind the earlier described trade-off between accurate estimates and good uncertainties, we carefully select the weights of the energy terms in p. During Bayesian optimization, we consider a combined measure for the accuracy of the flow fields and the confidence measure to avoid a performance drop for one of the two properties.

We apply our ProbFlow method to two energy formulations: the Horn-Schunck-inspired ClassicA approach from Sun et al. (2014) and the more recent EpicFlow energy (Revaud et al., 2015) to post-process FlowFields matches from (Bailer et al., 2015). For both methods, the AEE on the Middlebury flow dataset (Baker et al., 2007) is evaluated, while we only analyze the faster ProbFlowFields approach on the Sintel benchmark (Butler et al., 2012). For both datasets and energy terms, we observe our probabilistic optical flow predictions to be on par or even moderately better than the underlying MAP estimates.

To evaluate our uncertainty measure, we use the area under curve of sparsification plots (Bruhn and Weickert, 2006; Kondermann et al., 2008a; Mac Aodha et al., 2013) and the Spearman's rank correlation coefficient between uncertainties and per-pixel end-point errors. Only in one out of six evaluations, our model-inherent uncertainty is slightly outperformed by the learned confidences of Mac Aodha et al. (2013). For all remaining energies, datasets, and metrics, the closest confidence approach from previous work is clearly inferior with a relative change of $4 - 20\%$ in comparison to our ProbFlow uncertainties.

To illustrate the applicability of probabilistic optical flow, we perform two further experiments. First, we use the Fast Bilateral Solver (Barron and Poole, 2016) as an image-adaptive, confidence-aware post-processing method. Applied to the ProbFlowField results on Sintel, we observe a clear benefit when using our uncertainties in comparison to a uniform and thus uninformative confidence map. Second, ProbFlow-

Field is leveraged as input to the motion segmentation of Keuper et al. (2015). Our uncertainty measure and a forward-backward consistency check are used to generate point trajectories out of reliable flow predictions. In comparison to a MAP baseline, we observe an improved F-measure on the FBMS-59 dataset (Ochs et al., 2014) for the sparse as well as the corresponding densified motion segmentations.

2.1.3 *Discussion*

When evaluating our ProbFlow predictions, one might first expect a larger benefit in AEE in comparison to the underlying MAP estimates. However, the similar results are not surprising; the optical flow models remain almost unchanged and the MAP optimization schemes have been intensively studied. Instead, we present with ProbFlow the first fully probabilistic method that – at time of publication – performed competitively when evaluated on official benchmarks. In contrast, earlier approaches did not use a fully probabilistic setup, applied over-simplifying assumptions and/or resulted in unsatisfying performance, *e. g.* (Chantas et al., 2014; Roy and Govindu, 2000; Simoncelli et al., 1991; Sun et al., 2008). Here, the careful setup of our task-specific inference procedure is crucial to get the method to work in practice.

Our paper shows how probabilistic optical flow methods can be leveraged in the context of uncertainty measures. We demonstrate the clear benefit of our proposed confidences in comparison to post-hoc as well as previous model-inherent approaches. Nevertheless, model-inherent uncertainties – as the one proposed in the paper – might be disadvantageous if underlying model assumptions fail. This is at least partly the case when we apply our algorithm tailored to the Sintel images on the FBMS-59 dataset. However, one can then complement the model-inherent uncertainty estimates with reliability measures that are orthogonal to the proposed approach, *e. g.* forward-backward consistency checks as done for the task of motion segmentation.

In general, probabilistic optical flow and the estimation of corresponding uncertainties often remain a trade-off between accuracy and computational time. Probabilistic predictions mostly require additional computations, *e. g.* ProbFlowFields increases the runtime by a factor of 1.9 in comparison to the simple MAP baseline as additional variables have to be estimated. However, this increase seems well justified if the probabilistic approach can be beneficially applied, *e. g.* to obtain a powerful uncertainty measure as demonstrated in our paper.

As of now, ProbFlow cannot exactly be considered as state-of-the-art anymore due to more recent advances in the field since its publication in 2017. However, our work might have increased the attention to probabilistic optical flow approaches. For instance, Gast and Roth (2018) and Ilg et al. (2018) show methods for uncertainty estimation in deep networks for optical flow. Yin et al. (2019) cast optical flow

estimation as a probabilistic correspondence problem which leads to very competitive results on several major benchmarks.

2.1.4 *Contributions*

I implemented the ProbFlow algorithm and performed all optical flow experiments.[1] Moreover, I contributed the derivation of update equations, the proof for Bayesian risk minimization in the supplemental material and worked on the identification of the required specifics to deploy the optimization scheme in practice. Margret Keuper proposed to apply our estimated flow fields and the corresponding uncertainties to the task of motion segmentation and conducted the corresponding experiments. Stefan Roth led the idea generation to estimate probabilistic optical flow with variational inference and additionally supported the conceptual design as well as the entire scientific process of the project. All authors contributed to the writing of the paper.

2.2 STOCHASTIC VARIATIONAL INFERENCE WITH GRADIENT LINEARIZATION

This section summarizes the paper

> Tobias Plötz*, Anne S. Wannenwetsch*, and Stefan Roth
> *Stochastic Variational Inference with Gradient Linearization*,

which was published in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

2.2.1 *Motivation*

In the previous section, variational inference allowed us to estimate probabilistic optical flow and predict per-pixel uncertainties. Here, ProbFlow heavily relies on closed-form update equations to determine a variational distribution that approximates the original posterior. This approach is not ideal as it requires tedious derivations to be done by hand, *cf.* supplemental material of (Wannenwetsch et al., 2017). Moreover, the same work needs to be repeated as soon as a different energy model and therefore a different true posterior is chosen.

Stochastic approaches (Kingma and Welling, 2014; Ranganath et al., 2014; Rezende et al., 2014) clearly simplify the task of variational inference by only requiring the gradient of the considered log-posterior as an input and mainly working in a black-box fashion otherwise. While stochastic variational inference (SVI) methods are often applicable in practice, *e. g.* (Tran et al., 2017), they are not equally suitable

[1] Code is available at https://github.com/visinf/probflow.
* Authors contributed equally

for all kinds of posteriors due to their gradient-based optimization scheme. For instance, energy-based optical flow estimation mainly applies more advanced optimization approaches due to the complex structure of the used energy terms, *cf.* Section 1.1.1. The frequently applied scheme of gradient linearization, *e. g.* described in (Nikolova and Chan, 2007), alternates between the linearization of the energy gradient around the current optical flow prediction and a simultaneous update of all estimates as the solution of the equation system obtained from the linearization. In our ProbFlow paper (Wannenwetsch et al., 2017), we found such a joint update procedure to be equally essential during variational inference as otherwise the parameter optimization converges slowly and may lead to poor local minima.

The aim of the following contribution is thus the combination of both approaches, *i. e.* leveraging the benefits of gradient linearization in the context of SVI. Such an advanced stochastic inference scheme can then be used to obtain a variational approximation q of a complex posterior p, which may, for instance, be derived from energy functions for optical flow or similar. At the same time, the developed method should be as easily applicable as the standard SVI approaches.

2.2.2 *Proposed Method and Findings*

In the presented paper, we derive an algorithm for Stochastic Variational Inference with Gradient Linearization, denoted as SVIGL. Therefore, we consider the gradient of the KL divergence $D_{KL}(q \,\|\, p)$ between the original posterior p and the approximating distribution q. A stochastic approximation of the linearized gradient is derived and depends on the linearized gradient of the underlying energy term. Here, the re-parameterization trick (Kingma and Welling, 2014; Ranganath et al., 2014) is used to approximate the included expectation values by a finite set of samples from a standard normal distribution.

As for ProbFlow, we assume a fully-factorized approximating posterior q with marginal distributions being modeled as uncorrelated Gaussians. If we further approximate the gradient of the KL divergence, we can split its components into terms that depend linearly on the Gaussian parameters. New estimates for mean and variance can thus be obtained as the solution of the resulting linear system of equations. As such, an update of the variational parameters can again be expressed as a step of the gradient linearization procedure. The only interaction of the resulting SVIGL algorithm with the underlying energy model remains the usage of the linearized energy gradient.

As discussed more closely in our paper, SVIGL fits in each of its iterations a quadratic function to the stochastically approximated KL divergence. We show that this quadratic approximation is minimized in each update step if the matrix included in the gradient linearization of the underlying energy model is positive semi-definite. For the

latter, we further provide and prove a proposition that states two mild conditions under which this assumption holds.

In our experimental evaluation, we first apply SVIGL to the task of probabilistic optical flow estimation. We use the same energy function as for ProbFlowFields but leverage generalized Charbonnier penalties from Barron (2019). For comparison of our new SVI approach, we use gradient-based minimization of the KL divergence with the widely used stochastic gradient descent (SGD) and Adam optimizers and evaluate the resulting, unnormalized KL divergence obtained by sampling. On crops of randomly chosen Sintel images, SVIGL leads in a similar runtime to a clearly lower KL divergence than both gradient-based approaches. Moreover, SVIGL shows a moderate improvement in comparison to the KL divergence evaluated for a Laplace approximation of the posterior p around the MAP estimates.

We further apply SVIGL for stochastic updates of the continuous variables in the ProbFlowFields energy function using the original Gaussian Scale Mixtures as well as closed form updates for the latent variables. When compared to ProbFlowFields on full scale images of our Sintel test set, the AEE obtained by SVIGL is a bit worse due to a single outlier image. In contrast, confidence estimates show competitive when evaluated with the metrics as presented in Section 2.1.

The proposed SVIGL approach is not limited to the field of optical flow prediction. Instead, we show experiments for Poisson-Gaussian denoising using an energy function with a location-dependent Gaussian likelihood and the same smoothness term as for optical flow. SVIGL obtains a slightly better KL divergence as Adam on a subset of the Berkeley segmentation dataset (Martin et al., 2001) and converges considerably faster. Moreover, results of SVIGL clearly outperform the ones obtained with SGD and the evaluated Laplace approximation.

Finally, the applicability of SVIGL to the field of 3D surface reconstruction is demonstrated. Here, SVIGL is able to denoise a 3D point cloud and correctly identifies outlier points with low reliability as well as more difficult regions due to a higher noise level.

2.2.3 *Discussion*

Our proposed SVIGL algorithm is – depending on the application at least slightly – better than gradient-based SVI methods in terms of the obtained KL divergence. More importantly, SVIGL shows a clearly faster convergence for both examined applications of optical flow prediction and Poisson-Gaussian denoising. Additionally, we observe our approach to be more robust than Adam and SGD optimizers, which both highly depend on the chosen step size parameters.

In comparison to the underlying MAP estimates, SVIGL does not only allow to obtain confidences from the estimated marginal posteriors but it also maintains or even moderately improves application

performance for optical flow as well as Poisson-Gaussian denoising. This finding is consistent with our observations for the ProbFlow method. As such, SVIGL can be seen as a valuable tool for SVI since it is able to maintain and repurpose best practices from MAP estimation.

Our SVIGL algorithm is sufficiently convenient to apply as only the linearized gradient needs to be derived for the selected energy model. This property is often available due to the usage of gradient linearization for MAP predictions based on the same energy models. The easy application is a major advantage in comparison to model-specific methods, such as ProbFlow, which require tedious derivations of the update equation. However, SVIGL as well as other stochastic approaches remain an approximation due to their sampling-based procedure. Therefore, update steps might be noisy and one has to find a suitable trade-off for the used number of samples to balance the runtime as well as the robustness of the gradient. In general, sampling-based approaches require additional computation time due to the repeated evaluation of the (linearized) gradient for different samples. As such, closed-form update equations might be beneficial for a fixed energy model due to their faster runtime. Moreover, we observed – at least for the examined optical flow model – a slightly better performance of ProbFlowFields *w. r. t.* optical flow estimates as well as the uncertainty measure. Nevertheless, SVIGL is clearly the better choice for complex energy functions if an analytic derivation of the update equations is not possible or if a simple solution for the variational approximation of a posterior is required.

2.2.4 *Contributions*

The concept to combine Stochastic Variational Inference with Gradient Linearization as well as the corresponding SVIGL algorithm were jointly developed by Tobias Plötz and myself.[2] The same holds for the two propositions and the corresponding proofs presented in our paper. Tobias Plötz conducted experiments for Poisson-Gaussian denoising and 3D surface reconstruction. I performed the shown analyses for the task of optical flow estimation and worked on a scheme to use SVIGL for the ProbFlowFields energy term. Stefan Roth supported the conceptual design and provided scientific guidance throughout the entire project. All authors contributed to the writing process.

[2] Our implementation is available at https://github.com/visinf/svigl.

2.3 LEARNING TASK-SPECIFIC GENERALIZED CONVOLUTIONS IN THE PERMUTOHEDRAL LATTICE

This section summarizes the paper

> Anne S. Wannenwetsch, Martin Kiefel, Peter V. Gehler, and Stefan Roth
> *Learning Task-Specific Generalized Convolutions in the Permutohedral Lattice*,

which was published in the *41st DAGM German Conference on Pattern Recognition (DAGM GCPR 2019)*.

2.3.1 *Motivation*

In this section, we take one step back and look at general, state-of-the-art optical flow approaches as well as common prediction errors. Nowadays, many optical flow methods are based on neural networks, especially the ones leading popular optical flow benchmarks such as the Sintel (Butler et al., 2012) and KITTI (Geiger et al., 2012; Menze et al., 2018) datasets. As described in Section 1.1.3, convolutional layers are frequently used in deep networks for optical flow as well as for other dense prediction tasks. However, the applied standard convolutions are not content-adaptive and do not respect image boundaries, which is especially problematic if pyramidal structures are used to encode input images. In this case, the subsequent predictive part of a network is often not able to recover all location information lost during the feature generation step but the resulting estimates show boundary artifacts (Harley et al., 2017; Wu et al., 2018c). Moreover, dense prediction networks frequently output an estimate of low resolution, which is then upsampled by rather simple methods such as bilinear interpolation, *e. g.* (Chen et al., 2018; Sun et al., 2018; Yin et al., 2019). As such, information of different objects is combined, which leads to a further loss of accuracy in dense predictions. For the estimation of optical flow, we frequently observe blurry flow fields as well as the disappearance of small object details and their individual motion.

To counteract this behavior, we aim for a convolution operation that defines closeness between pixels not only based on spatial distance. Instead, we want to leverage guidance data, such as the RGB input images, as it contains valuable information about the structure of the correct flow fields. Here, a key focus lies on the fact that the relation between guidance and prediction data should be learned to improve the usefulness of such guided convolutions, *cf.* Section 1.3.2.

Generalized convolutions with learned guidance features have been presented before. However, the methods frequently put restrictions on the guidance data, *e. g.* (Pan et al., 2019; Wu et al., 2018b), or the filtering operation itself, *e. g.* (Gharbi et al., 2017). Instead, we aim for

a powerful approach that is applicable to a large variety of tasks. A drawback of other generalized filtering operations is the substantial number of additional outputs, *e. g.* Dynamic Filter Networks (De Brabandere et al., 2016) predict location-specific kernels for all pixels. In contrast, the proposed method should only introduce a small number of additional parameters to reduce the risk of overfitting.

2.3.2 *Proposed Method and Findings*

Our generalized convolution approach, denoted as *semantic lattice*, is based on the permutohedral lattice of Adams et al. (2010). In the corresponding permutohedral space, each point is characterized by its data values as well as its features. The latter describe the location of every pixel in the multi-dimensional lattice and thus its closeness to other input data. Depending on their location, the inputs are splat to grid points of the lattice on which the actual Gaussian convolution is then performed. In a last slicing step, the results of the convolution are interpolated to obtain data at the desired output locations.

The permutohedral lattice has already been extended to learned, non-Gaussian filter weights in (Jampani et al., 2016; Kiefel et al., 2015). We further generalize the permutohedral operations *w. r. t.* learnable features, *i. e.* we learn parameters that characterize the lattice space from data via backpropagation. The resulting optimization problem is complex, we thus propose two important simplifications: First, we assure that input and output features are consistent and lead to locations that are sufficiently close in permutohedral space. This is necessary as gradients can only propagate over lattice points that are connected by the convolution operation. Therefore, basic features, *e. g.* RGB values of a guidance image, are processed by a convolutional *embedding network* that is shared between inputs and outputs to ensure the required consistency. Second, we ensure that input points are not too distant in lattice space. Here, the permutohedral lattice tends to push apart different inputs as this reduces the blur introduced to the data values during the operations. However this tendency also leads to output locations without any input information and thus no convolution result. Such points have an equally uninformative gradient, which does not anymore contribute to gradient-based optimization. As such, we use batch normalization as the last layer of our embedding network to keep the output range of our features limited.

Using some further optimization details, we apply the semantic lattice to different guided upsampling tasks. First, we consider joint color upsampling where gray-scale images guide the upsampling process of low-resolution color data. We demonstrate that learned features substantially improve the performance of the permutohedral operations. Moreover, the results of the semantic lattice also clearly outperform baseline approaches as well as previous learned methods.

We further use the task of color upsampling to validate different configurations of our chosen setup and demonstrate, for instance, the necessity to restrict the outputs of the embedding network.

In a second part, the semantic lattice is applied to state-of-the-art deep networks to replace the last upsampling step of dense predictions. We leverage the high-dimensional (first) input image as additional guidance data and adapt PWC-Net (Sun et al., 2018) for optical flow estimation as well as DeepLabv3+ (Chen et al., 2018) for semantic segmentation. Using images of the Sintel (Butler et al., 2012) and Pascal VOC 2012 (Everingham et al., 2015) datasets, the semantic lattice leads to improved results in comparison to the underlying networks using simple bilinear upsampling. Moreover, we again observe an improvement when learning feature parameters instead of using hand-crafted feature dimensions. In comparison to related work, the semantic lattice shows better results for optical flow and is competitive for the upsampling of segmentation maps. In both tasks, improvements over the baseline networks are especially visible at content boundaries. For instance, optical flow results are clearly better when evaluating the AEE close to motion discontinuities. Moreover, flow fields are crisper and segmentation maps are better aligned with the underlying objects.

2.3.3 *Discussion*

We have shown a clear benefit of learning task-specific features of the permutohedral lattice operations for different tasks of joint upsampling. Moreover, we have demonstrated the benefits of replacing the widely used bilinear upsampling step by an image-adaptive version in state-of-the-art deep networks for optical flow and semantic segmentation. For the task of optical flow estimation, we observe a clear visual improvement with less blurry flow fields. Even though the overall differences in AEE are moderate, we observe a clear quantitative improvement when evaluating the AEE in regions close to motion boundaries. Additionally, one might expect a larger benefit if the semantic lattice was trained together with the dense prediction network. Our presented experiments with a fixed baseline network only correspond to a lower bound on the possible overall improvements.

The presented approach is by no means restricted to the upsampling setup described in our paper. Instead, it seems possible to construct entire networks with semantic lattice convolutions and build, for instance, multi-dimensional networks for different problem domains, *e. g.* classification or recognition. Moreover, the type of used basic features can be easily changed. While we only leverage RGB image values, the approach can be applied to different guidance data such as semantic classes or the examined predictions themselves.

Given the named applications, one should always keep in mind the relatively high computational time of the permutohedral lattice. Here,

the runtime increases with the number of feature dimensions, the number of considered input and output points as well as the neighborhood size chosen for the convolution (Adams et al., 2010; Kiefel, 2017). Since the number of points is generally fixed, the design of a semantic lattice layer is a trade-off between a faster runtime and a more expressive convolution operation. For colorization, we observed that an increased number of neighbors and a higher dimensionality of the lattice space do not necessarily correspond to a (large) increase in accuracy. As such, a suitable hyperparameter choice always depends on the task of interest. For all examined problems, we found a neighborhood size of one and a relatively low number of features to work well in practice.

A second potential drawback of the semantic lattice is its slightly involved training procedure. In a first step, scaling factors of the basic features have to be obtained via grid search. Subsequently, spatial features are fixed and individual learning rates for feature networks and permutohedral kernels need to be determined for a final, joint learning cycle. As such, the semantic lattice might be especially useful for applications in which the permutohedral lattice has already shown to be beneficial due to its specific properties, e.g. for the processing of sparse data (Kiefel et al., 2015; Su et al., 2018). In such cases, the semantic lattice is clearly an interesting extension as features generally remain hand-crafted and the corresponding parameters are determined manually. Here, the additional learning allows to use more powerful features and is thus likely to improve the filtering results.

2.3.4 *Contributions*

I led the implementation process of the semantic lattice[3], performed its training including the required specifics and conducted all experiments presented in the paper. Martin Kiefel and Peter V. Gehler developed the idea to learn features for the permutohedral lattice. Moreover, Martin Kiefel provided a first implementation of the semantic lattice layer and (partly) reviewed the remaining code together with Peter V. Gehler. Stefan Roth, Martin Kiefel and Peter V. Gehler all contributed to the conceptual design as well as the experimental setup and provided scientific guidance. The paper was written by Stefan Roth and myself with input from all authors.

[3] All code is available at https://github.com/visinf/semantic_lattice.

2.4 PROBABILISTIC PIXEL-ADAPTIVE REFINEMENT NETWORKS

This section summarizes the paper

> Anne S. Wannenwetsch and Stefan Roth
> *Probabilistic Pixel-Adaptive Refinement Networks*,

which was published in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

2.4.1 *Motivation*

In the previous section, image-adaptive upsampling with learned similarity features has shown to be beneficial for outputs of dense prediction networks. For instance, the semantic lattice allowed to smooth flow fields while respecting learned content boundaries.

However, the method proposed in (Wannenwetsch et al., 2019) as well as other content-adaptive refinement approaches for neural networks, *e. g.* (Pan et al., 2019; Su et al., 2019; Wu et al., 2018b), do not leverage confidence data during post-processing. This is counterintuitive as uncertainty estimates generally provide valuable information about the per-pixel reliability and thus the corresponding prediction errors. For instance, we already showed in (Wannenwetsch et al., 2017) that uncertainty measures are a helpful tool to refine flow fields. Here, we used the Fast Bilateral Solver (Barron and Poole, 2016) for optical flow post-processing and applied a parametrized sigmoid function to normalize the estimated confidences. As argued in Section 1.3.2, this approach can be improved since it uses a pre-defined filtering scheme, which can only be slightly adjusted by changing its hyperparameters. Moreover, the Fast Bilateral Solver does not further process its inputs; neither the used image information nor the optical flow uncertainties.

In this section, we thus aim for a deep network approach that considers guidance and reliability data to refine dense predictions. As such, the method is simultaneously able to respect learned content boundaries and to propagate only reliable estimates. Moreover, the approach should be fast, content-adaptive and as flexible as possible, *i. e.* our goal is to learn the filtering operation itself, an embedding of similarity features, and a further processing of input confidences.

To obtain the confidences required by the proposed method, one could directly leverage the outputs of dense classification tasks as the applied networks often output log-probabilities of the examined classes. For other dense predictions, networks have to explicitly predict corresponding uncertainties; a task that has gained increased interest lately, *e. g.* (Gast and Roth, 2018; Novotny et al., 2018; Yin et al., 2019).

2.4.2 *Proposed Method and Findings*

We build our approach on pixel-adaptive convolutions (PACs) by Su et al. (2019) due to their fast runtime and broad applicability. PACs split the weight parameter of a standard convolution into two components: a fixed weight learned from data as well as a location-specific kernel. The kernel has a pre-defined shape, *e. g.* the form of a Gaussian radial basis function, and compares pixels according to learned features, *i. e.* it weighs pixel values more if their characteristics are similar.

In a first step, we extend PACs with an advanced normalization scheme adapted to the PAC context from (Wannenwetsch et al., 2019). This additional operation is beneficial as the pixel similarity in convolutional neighborhoods varies across the image, which leads - without normalization - to smaller output values for certain pixels, e.g. at object boundaries or in textured areas. Therefore, we propose to use a normalization scheme that takes into account the closeness or similarity of neighboring pixels as well as their individual weighting in the filtering operation. This ensures PAC results with consistent magnitudes and thus simplifies the learning of convolution weights.

In a second step, we adjust PACs such that each pixel value in a neighborhood is additionally weighted according to its own confidence. This allows to replace erroneous estimates with reliable predictions and avoids to propagate values of outlier pixels into surrounding areas. The resulting probabilistic pixel-adaptive convolutions (PPACs) can thus perform a content-adaptive and reliability-aware filtering step.

We propose to apply PPACs in lightweight refinement networks, which take predictions of task-specific models as well as probability data and high-resolution guidance images as inputs. In a first step, the estimates are upscaled to full resolution by the default method of the original network. We then propose to pre-process uncertainty and guidance data in individual branches to obtain powerful representations for both properties. All intermediate outputs are finally processed by a small number of PPACs to generate refined predictions.

To compare the proposed PPAC network, we evaluate two further baselines: A simple refinement network takes the same inputs as the PPAC refinement but only applies standard convolutions. The so-called PAC refinement network replaces PPACs with non-probabilistic PACs and concatenates confidence inputs to the remaining guidance data.

In our evaluation, we apply all types of refinement networks to state-of-the-art HD3 optical flow predictions from Yin et al. (2019) and compare the AEE on subsets of Sintel clean and final (Butler et al., 2012) as well as both KITTI datasets (Geiger et al., 2012; Menze et al., 2018). As guidance data, we again use the first input image and provide the networks with nearest neighbor estimates of the discrete HD3 probability estimates. In a first experiment, we demonstrate the benefit of our advanced normalization scheme and show that it leads to a

substantial performance increase when applied to the above described PAC refinement networks. Moreover, probabilistic refinement with PPACs outperforms both evaluated baselines as well as approaches from related work by a large margin. Visualizations of the refined optical flow show a clear improvement at motion boundaries and overall smoother flow fields with crisp boundaries.

In the supplemental material, we further analyze the results of PPAC refinement by splitting the flow fields predicted by the HD3 method into reliable and unreliable estimates according to their probabilities. On both Sintel subsets, we observe a far more substantial improvement of the AEE for unreliable pixels, which justifies our assumption that PPAC refinement allows to propagate correct predictions into erroneous regions. Additionally, the application of PPACs leads to a clearly higher accuracy benefit than the corresponding PAC refinement in uncertain areas of our KITTI test set. We conclude that PPACs can handle unreliable pixels better than their non-probabilistic counterparts.

When retrained on all training images and submitted to the official benchmark, PPAC refined HD3 optical flow shows very competitive and ranks 4th on Sintel final among all previously published two-frame optical flow methods. Even more impressively, our proposed PPAC-HD3 predictions outperform – at time of publication – all other optical flow methods on both KITTI benchmarks. Here, PPACs lead to a relative improvement of 11.1% and 7.5% in comparison to the underlying HD3 outlier rate on KITTI 2012 and 2015, respectively.

We also demonstrate the benefits of PPAC refinement for the task of semantic segmentation on the Pascal VOC 2012 dataset (Everingham et al., 2015). Using input images for guidance and network-inherent probabilities, we apply PPAC refinement to segmentation maps from DeepLabv3+ (Chen et al., 2018). Here, the post-processing leads to a considerably improved mean intersection over union. As for optical flow, the PPAC approach shows clearly superior in comparison to the evaluated baselines as well as related work. We further observe that the refined segmentation maps align better with (small) objects and show improved segmentations at the intersection of objects.

2.4.3 *Discussion*

The usage of PPAC refinement leads to a clear reduction of boundary artifacts and substantially improves the accuracy for optical flow and semantic segmentation. The experiment in the supplemental material validates the conclusion that the usage of confidences is beneficial for unreliable regions. Overall, PPACs lead to clearly stronger performance gains than the method presented in Section 2.3. This holds even though we continue to train only the refinement networks. We attribute the larger benefits at least partly to the easier optimization of PPACs as well as to the faster runtime in comparison to the semantic lattice.

Here, the improved computational time allows for higher flexibility during training and an increased number of training iterations.

While the proposed PPAC refinement has many beneficial properties, it is not completely without drawbacks. On the one hand, the proposed normalization scheme requires additional time as almost the full convolution operation is done twice. The same holds for the entire refinement process, which has a computational overhead of 1.7x on KITTI 2015 in comparison to the underlying HD3 method. However, the substantially improved results seem to clearly justify the additional runtime. Another disadvantage is the fact that PPAC refinement remains dependent on the correctness of its inputs, *i. e.* the quality of predictions and their underlying confidences. For instance, HD3 optical flow shows some artifacts, especially for sky regions of KITTI images, which are not covered by the sparse ground truth. While outlier pixels might be improved, it is not possible to completely erase a larger erroneous region with our current setup. In such cases, extended PPAC filtering sizes might be beneficial but the larger number of parameters simultaneously increases the risk of overfitting.

Even though PPAC refinement might depend on the quality of the input estimates, it remains independent of the estimation process itself. As such, the method is broadly applicable to different approaches and is likely to benefit from more accurate predictions. For the task of optical flow, experiments have shown the potential for further improvements when using oracle confidences obtained from per-pixel end-point errors. Similarly, one would expect that PPAC refinement would also benefit from improved input predictions.

In general, the presented paper proposes a very lightweight refinement network with a manageable computational overhead and only as little as 12k additional parameters for the task of optical flow estimation. The approach is flexible and learns the embedding of guidance and confidence information as well as the filtering step from data. As such, PPAC refinement is a powerful tool, which leads to state-of-the-art results especially on optical flow benchmarks.

2.4.4 *Contributions*

I led the idea generation to perform image-adaptive neural network refinement with probabilistic cues, contributed the implementation[4] and performed the data evaluation. Stefan Roth supported the conceptual design, contributed to the experimental setup and provided scientific guidance. Both authors contributed to the writing of the paper.

[4]Code is publicly available at https://github.com/visinf/ppac_refinement.

3 DISCUSSION

CONTENTS

This thesis targets the task of probabilistic optical flow estimation as well as the image-adaptive refinement of estimates with or without corresponding probabilities. In the following, I will summarize my contributions, discuss some potential limitations of the presented approaches and conclude with an outlook on possible future work.

3.1 SUMMARY OF CONTRIBUTIONS

With ProbFlow (Wannenwetsch et al., 2017), I presented the first method for fully probabilistic optical flow that showed – at time of publication – state-of-the-art performance on different, popular benchmarks. The model-inherent uncertainty measure clearly outperforms previous confidence approaches. Moreover, the uncertainties are shown to be beneficial for optical flow refinement as well as in an application to motion segmentation. The applied variational inference approach requires update equations for all variables to fit a suitable approximating distribution to the modeled posterior. The corresponding simple but tedious, manual derivations can be avoided by the proposed SVIGL algorithm (Plötz et al., 2018). The approach combines ideas from Stochastic Variational Inference with the procedure of gradient linearization, which improves the applicability of SVI to complex posterior models. For the examined tasks, SVIGL shows faster convergence speed and is more robust *w. r. t.* to hyperparameters than gradient-based optimizers that are widely used for SVI.

In (Wannenwetsch et al., 2019), I demonstrated how a generalized convolution operation can be improved by learning per-pixel similarity features based on guidance data. The approach is applied to joint upsampling tasks, among others, of neural network outputs for optical flow. The resulting semantic lattice with learned feature parameters leads to improved results over different baselines and related work. This especially holds in visual comparison where one can observe a substantial reduction of boundary artifacts. My last contribution (Wannenwetsch and Roth, 2020) merges both research fields and proposes probabilistic pixel-adaptive refinement networks. In comparison to

previous work, the approach leverages reliability estimates on top of image guidance data for a learned filtering operation. As such, the method is not only able to refine estimates at content boundaries but also improves unreliable predictions and avoids their propagation. When applied to a recent probabilistic neural network for optical flow, PPAC refinement achieves state-of-the-art results with a substantial improvement on different popular benchmark datasets.

3.2 POTENTIAL LIMITATIONS

ADDITIONAL TIME REQUIREMENT. One fact that connects probabilistic optical flow as well as optical flow refinement is the additional amount of computational resources, which are required in comparison to basic methods. It is more challenging to predict an entire optical flow posterior distribution and the refinement is generally performed on top of the initial predictions. As such, methods in both areas remain a trade-off between the overall runtime and the benefits of the estimates, *i. e.* a distribution of optical flow or a refined flow field.

For ProbFlowFields and PPAC-HD3, we have – in comparison to the underlying methods – observed a computational overhead of 1.9x and 1.7x, respectively. While the additional time requirement is not overly extensive, the more difficult computations seem only reasonable if the obtained results are beneficial in practice. For probabilistic optical flow as well as refinement approaches, this is certainly the case if the algorithms lead to better estimates. Then, subsequent applications leveraging optical flow as a motion cue can directly benefit from improved results, *e. g.* (Chao et al., 2014; Jhuang et al., 2013; Oliveira and Tavares, 2014). Here, we have shown a clear improvement for the first refinement approach, the semantic lattice, and an even more significant performance gain using PPAC refinement networks.

Additionally, the estimated distribution or its marginals might be beneficially applied in subsequent applications, *e. g.* (Domke and Aloimonos, 2007; Janai et al., 2019; Weinzaepfel et al., 2015). Therefore, we demonstrated the potential of our ProbFlow confidence for the task of motion segmentation in (Wannenwetsch et al., 2017). Finally, we showed that probabilities can be applied for optical flow refinement, which again leads to better estimates, *cf.* Sections 2.1 and 2.4.

One minor point concerning additional time requirements is the fact that explicit refinement networks might need an additional training step unless they are fully integrated into the overall estimation network and training procedure. Here, one should consider that such an integration into existing approaches is possible for both approaches proposed in this thesis. Moreover, learning deep neural networks for optical flow is generally a lengthy procedure consisting of several pretrainings on large (synthetic) datasets and a subsequent fine-tuning (Sun et al., 2020). In contrast, the training of a refinement network can

be done in a relatively short time, especially if only a small number of refinement parameters is used – as for both proposed methods.

CUSTOMIZED ALGORITHMS. In the papers underlying this thesis, different applications are described for the presented algorithms. Nevertheless, all approaches remain – at least to a certain extent – customized for their specific use cases. As such, it is not always trivial to apply the methods in a new context but some adaptations might be necessary when using them for a different task.

For ProbFlow, closed-form update equations, which depend on the application-specific energy formulation, have to be determined. Even though different energy models might contain similar terms, the manual derivations remain tedious, *cf.* supplemental material of (Wannenwetsch et al., 2017). In contrast, the application of SVIGL is more convenient since it uses stochastic approximations instead of deriving analytical expressions for the variable updates. SVIGL still requires a linearization of the energy gradient but this property is often easier to obtain and might be available from MAP estimation. Moreover, ProbFlow and SVIGL have a few hyperparameters, which influence the accuracy of the predictions. Here, appropriate values for several hyperparameters can be automatically obtained by using Bayesian optimization as described in the corresponding papers.

The image-adaptive convolutions described in (Wannenwetsch et al., 2019) and (Wannenwetsch and Roth, 2020) both have a slightly more complex learning procedure than the one used for standard convolutions. For instance, the semantic lattice additionally requires to determine scaling factors for the individual features. Similarly, PPAC refinement networks might benefit from individual learning rates for the pre-processing and combination branches. In both cases, appropriate parameter values have to be determined based on the specific application. Therefore, the papers describe general learning procedures for both approaches, which have shown to be applicable to different refinement tasks, *e. g.* for optical flow and semantic segmentation.

LIMITED ATTENTION TO PROBABILISTIC OPTICAL FLOW. Even though I have described many beneficial properties of probabilistic optical flow methods and confidence measures in general, probabilistic approaches account only for a comparably small share of the entire research on optical flow estimation. One of the major reasons for this may be the fact that optical flow prediction is generally a difficult problem, which is not eased by the requirement to estimate a corresponding posterior distribution. Additionally, many previous works on probabilistic optical flow are not (anymore) competitive *w. r. t.* their predictive accuracy, *e. g.* due to simplifying model assumptions or estimation procedures (Chantas et al., 2014; Roy and Govindu, 2000; Simoncelli et al., 1991). However, the accuracy of the flow fields ob-

tained from a probabilistic approach is as important as the correct prediction of underlying probabilities. If a method leads to reliable probability predictions but the corresponding optical flow estimates are too inaccurate, how can it then still be important that mainly erroneous predictions are correctly classified as such?

Therefore, it is clearly one of our major contributions that we presented with ProbFlow an algorithm that performs at least on par or even better than its MAP estimation counterparts and thus demonstrates the potential of fully probabilistic methods. Our work will hopefully continue to encourage further research on probabilistic approaches for optical flow, which jointly estimate accurate flow fields and reliable probabilities. In this regard, it seems to be a good sign that HD3 optical flow (Yin et al., 2019) – as one of the currently leading approaches – is also based on a fully probabilistic framework.

FUTURE NECESSITY OF OPTICAL FLOW REFINEMENT. Considering the topic of optical flow refinement, one can finally wonder about its necessity in the future. Here, it is certainly difficult to assess how the research and especially leading methods are going to evolve in the next years. Nevertheless, one observes a continuous improvement of optical flow algorithms on popular optical flow benchmarks (Baker et al., 2007; Butler et al., 2012; Geiger et al., 2012; Menze et al., 2018). It thus seems plausible that optical flow predictions improve further over time and will soon show a smaller number of prediction errors.

However, there are many challenges, *e. g.* occlusions and illumination changes, that are inherent in the task of optical flow estimation. As such, some of the problems described in Section 1.1.3 might be overcome with better algorithms but there are certainly image regions that will remain more challenging than well-structured, time-consistent areas. As such, I would assume a refinement to stay beneficial even if the underlying approaches may improve overall.

Instead, it seems plausible that post-processing steps will be more closely integrated into the estimation process. For instance, different methods explicitly target occluded pixels by predicting optical flow as well as occlusions and using a dedicated procedure for corresponding pixels (Alvarez et al., 2007; Hur and Roth, 2017; Ince and Konrad, 2008; Kennedy and Taylor, 2015; Xiao et al., 2006). Similarly, image-adaptive convolutions can be directly integrated into optical flow networks, *e. g.* (Hui et al., 2018; Hur and Roth, 2019). The approaches in Sections 2.3 and 2.4 could thus be used as part of a larger deep network and trained in an end-to-end fashion together with other parameters.

3.3 FUTURE WORK

Given the contributions presented earlier in this thesis, there are many directions for possible future work. Rather straightforward follow-up work of the presented papers could target specific, smaller limitations of the individual approaches. For instance, it seems interesting to obtain not only refined optical flow estimates with our probabilistic pixel-adaptive refinement networks but simultaneously provide a reliability measure for the post-processed predictions. This goal could be achieved by equally refining the initial probability values, *e. g.* by learning separate, dedicated filtering operations from data or by using the available filter weights as done in (Eldesokey et al., 2018) for a special form of normalized convolutions (Knutsson and Westin, 1993). Similarly, it might be promising to combine approaches for model-inherent as well as post-hoc confidence measures more closely. For instance, one could integrate a symmetry requirement for forward-backward optical flow into the presented ProbFlow framework.

Thinking more broadly about future work, I see several promising directions. First, neural networks are by now clearly an essential tool in optical flow estimation (Ilg et al., 2017; Sun et al., 2018; Yin et al., 2019). However, the absence of large amounts of realistic ground truth severely challenges different approaches as such data is generally difficult to obtain (Butler et al., 2012). Therefore, deep optical flow estimation trained in an unsupervised setting has gained importance with more and more researchers following this interesting path, *e. g.* (Liu et al., 2019; Meister et al., 2018; Yu et al., 2016). Here, a combination of probabilistic optical flow with unsupervised learning seems to have great potential for future work. On the one hand, different approaches (Liu et al., 2019; Zhu et al., 2017) leverage optical flow data obtained from other algorithms – so-called teacher methods – for their unsupervised training. As this data only approximates the unknown ground truth, it would be interesting to investigate to what extent such methods can benefit from probabilistic estimates of the teacher approaches. More general, probabilistic methods for unsupervised deep optical flow seem promising as the unsupervised prediction task is highly ill-posed and in practice even more challenging than learning from ground truth data. While such methods for unsupervised optical flow prediction are still rare, Poggi et al. (2020) have shown the possible benefits of uncertainty estimates for the slightly related task of self-supervised monocular depth estimation.

Additionally, different recent approaches (Brickwedde et al., 2019; Ranjan et al., 2019; Yin and Shi, 2018; Zou et al., 2018) aim for the joint estimation of optical flow as well as other properties such as depth, ego-motion and/or object segmentation. A joint predictions of several quantities is a promising strategy since the combination of different tasks allows to access a larger amount of information and provides

an advanced way to regularize the corresponding estimates based on geometric or temporal constraints (Ranjan et al., 2019). Brickwedde et al. (2019) leverage a probabilistic formulation for depth estimation and show its beneficial application in the context of monocular scene flow prediction. In (Dharmasiri et al., 2018), single-image depth prediction and camera pose estimation are improved by additionally training the network for probabilistic optical flow. These observations suggest that joint approaches could benefit from probabilistic optical flow methods as they allow to assess the reliability of the estimates in subsequent prediction approaches or in combined constraints.

Finally, as stated before, the introduced as well as other image-adaptive refinement layers can be integrated into existing neural networks and learned in an end-to-end fashion. Here, it seems interesting to broadly investigate the usage of refinement methods in unsupervised neural networks for optical flow and/or related tasks as these approaches frequently miss homogeneous predictions with well aligned, sharp boundaries. Image-adaptive refinement could help to leverage additional information from guidance and, if available, probability data, to obtain accurate estimates also in challenging regions.

a

APPENDIX

A.1 PROBFLOW: JOINT OPTICAL FLOW AND UNCERTAINTY ESTIMATION

Anne S. Wannenwetsch, Margret Keuper, and Stefan Roth
2017 IEEE International Conference on Computer Vision (ICCV)

Abstract

Optical flow estimation remains challenging due to untextured areas, motion boundaries, occlusions, and more. Thus, the estimated flow is not equally reliable across the image. To that end, post-hoc confidence measures have been introduced to assess the per-pixel reliability of the flow. We overcome the artificial separation of optical flow and confidence estimation by introducing a method that jointly predicts optical flow and its underlying uncertainty. Starting from common energy-based formulations, we rely on the corresponding posterior distribution of the flow given the images. We derive a variational inference scheme based on mean field, which incorporates best practices from energy minimization. An uncertainty measure is obtained along the flow at every pixel as the (marginal) entropy of the variational distribution. We demonstrate the flexibility of our probabilistic approach by applying it to two different energies and on two benchmarks. We not only obtain flow results that are competitive with the underlying energy minimization approach, but also a reliable uncertainty measure that significantly outperforms existing post-hoc approaches.

ProbFlow: Joint Optical Flow and Uncertainty Estimation

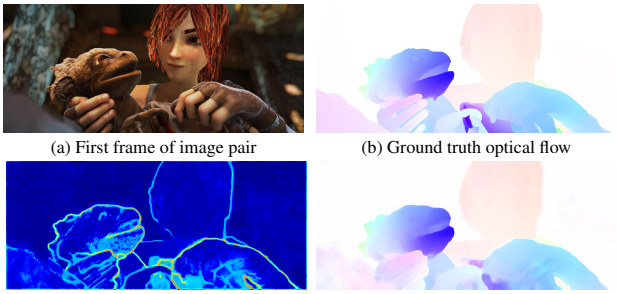
Anne S. Wannenwetsch[1] Margret Keuper[2] Stefan Roth[1]
[1]TU Darmstadt [2]University of Mannheim

Abstract

Optical flow estimation remains challenging due to untextured areas, motion boundaries, occlusions, and more. Thus, the estimated flow is not equally reliable across the image. To that end, post-hoc confidence measures have been introduced to assess the per-pixel reliability of the flow. We overcome the artificial separation of optical flow and confidence estimation by introducing a method that jointly predicts optical flow and its underlying uncertainty. Starting from common energy-based formulations, we rely on the corresponding posterior distribution of the flow given the images. We derive a variational inference scheme based on mean field, which incorporates best practices from energy minimization. An uncertainty measure is obtained along the flow at every pixel as the (marginal) entropy of the variational distribution. We demonstrate the flexibility of our probabilistic approach by applying it to two different energies and on two benchmarks. We not only obtain flow results that are competitive with the underlying energy minimization approach, but also a reliable uncertainty measure that significantly outperforms existing post-hoc approaches.

1. Introduction

Optical flow estimation has been extensively studied for more than three decades [5, 9, 14, 21, 36]. However, motion boundaries, large displacements, and occlusions still lead to erroneous flow fields, especially in challenging imaging conditions. In contrast, flow predictions are almost errorless in textured regions of uniform motion, which causes the reliability of optical flow predictions to vary greatly across the image. Uncertainty measures[1] aim to predict the reliability of the flow estimates and rate each estimate according to its predicted accuracy [24, 27, 30]. One application of such measures is to improve optical flow estimation using different ways of post-processing [4, 36]. Moreover, uncertainty measures represent an important tool when optical flow is used as a cue for other computer vision tasks, such as image segmentation or tracking, which consequently depend on the correctness of the flow. In such cases, error propagation can be avoided if only flow estimates of high confidence are taken into consideration, *e.g.* [33, 47].



(a) First frame of image pair (b) Ground truth optical flow

(c) Uncertainty map from our method (d) Flow estimate from our method

Figure 1. Our uncertainty measure accurately detects regions with reliable flow predictions (dark blue) as well as parts of the image with erroneous estimates (dark red). *Best viewed on screen.*

Several post-hoc confidence measures have been introduced to directly estimate the uncertainty of flow predictions, *e.g.* [3, 20, 24, 30]. However, optical flow estimation is known to be rather challenging as a one-shot process [14, 15, 43]. We thus believe it is natural to ask why confidence estimation should be restricted to non-iterative (post-hoc) approaches. Moreover, flow algorithms for a concrete problem are carefully selected for the task at hand and provide us with an underlying model. We, thus, argue that it is very desirable to apply a *model-inherent* uncertainty measure tailored to the chosen method. We even go one step further and aim to *jointly predict* optical flow and a corresponding uncertainty in order to preserve and extract important information contained in the flow estimation process.

Our work focuses on energy-based methods [3, 15, 43] as they represent a flexible framework in widespread use. Starting from a general energy formulation, we derive a probability distribution of the flow field conditioned on the input images. In this setting, optical flow can be determined by minimizing the expected loss under the modeled posterior and the corresponding uncertainty measure is naturally obtained as the marginal entropy of the posterior at every pixel. Due to the intractability of exact posterior inference in general, we rely on variational approximate inference and

[1]Uncertainty measures are closely related to confidence measures as their values are inversely related to confidence estimates.

use a mean-field approach. In this way, flow predictions and uncertainty estimations can be obtained as the result of a joint minimization problem that makes an additional training step of the uncertainty measure unnecessary [17, 30]. Fig. 1 shows an exemplary flow field and uncertainty prediction of our probabilistic approach obtained on the Sintel benchmark [11]. We term our approach *ProbFlow*.

To the best of our knowledge, we introduce the first *fully* probabilistic flow estimation approach that shows *competitive results* on established benchmarks. Moreover, our work is unique in that it is *broadly applicable* to a large number of existing energy-based optical flow algorithms and that we demonstrate how to benefit from the probabilistic framework in the context of uncertainty measures.

We apply ProbFlow to a classic Horn-Schunck-style energy as proposed in [43] and to the more recent EpicFlow formulation [36]. We show that the obtained flow fields are competitive with or even outperform the corresponding energy minimization results on the Middlebury [2] and Sintel [11] datasets. To assess the performance of the uncertainties estimated by our method, we rely on existing evaluation approaches and propose a new criterion based on the Spearman's rank correlation coefficient. Our uncertainty measure is clearly superior in comparison to various competing approaches and significantly outperforms the best existing methods. We further show the benefits of our uncertainty estimates in an application to motion segmentation [22] by generating highly reliable point trajectories.

2. Related Work

Starting from the work of Horn and Schunck [21], global energy minimization approaches have been used extensively for optical flow estimation; see [3, 15, 43] for in-depth overviews of existing methods and best practices. Despite their conceptual simplicity, global energy formulations are still up-to-date [13, 29]. Moreover, energy-based approaches are frequently used for post-processing optical flow predictions obtained through other means [1, 14, 36].

In the past, a variety of optical flow uncertainty measures have been proposed. Various simple methods are based on the characteristics of the input data only, *e.g.* [3, 20]. We refer to [24, 27] for a more comprehensive summary of such confidence approaches. In comparison to our work, all of these measures omit important sources of information by not considering the estimated optical flow field itself.

A second class of confidence measures relies on an analysis of the estimated flow fields. Kondermann *et al.* [23] learn a linear subspace of true spatio-temporal flow neighborhoods and use the reconstruction error of an estimated flow vector to evaluate its reliability. In [24], a probabilistic model of flow patches is learned and approximated with a Gaussian distribution, which yields a confidence measure based on hypothesis testing. But this second class of confidence measures does not consider all aspects of flow uncertainty either, as the input images are not taken into account.

Recently, Mac Aodha *et al.* [30] proposed a confidence measure based on the input as well as the output of an optical flow algorithm. Using a multi-cue feature vector, a classifier is trained to predict whether the endpoint error at a certain pixel is smaller than a previously defined threshold. The probability output of the classifier is used as a confidence measure. However, [30] does not take advantage of uncertainty information available in the flow estimation procedure itself and requires a separate training step that is unnecessary for our model-inherent approach.

Uncertainty measures tailored to energy minimization approaches have been proposed especially for *local* energy methods, *e.g.*, [3, 31, 40, 41]. Model-inherent confidence measures designed for *global* energy formulations are quite rare, on the other hand. Kybic and Nieuwenhuis [27] introduce a method that relies on bootstrap resampling. Based on varying pixel contributions, repeated optical flow estimation is performed and a confidence measure is determined as the total standard deviation of the obtained flow predictions. Bruhn and Weickert [8] propose the inverse local energy as a confidence measure, which is applicable to a broad variety of energies. A large uncertainty is thus associated with a strong violation of the local model assumptions encoded in the energy formulation. Gehrig and Scharwächter [17] use the energy and combine it with several features such as the spatial and temporal flow variance in order to obtain a real-time confidence estimate. In contrast to the two above approaches, our method does not explicitly limit the receptive field of the confidence measure by only considering the local neighborhood of a pixel. Instead, uncertainty propagation is facilitated by means of iterative spatial inference.

In the past, several methods based on a probabilistic flow formulation have been proposed [35, 39, 40, 41]. However, many of these works are based on simplifying assumptions such as locally constant flow or a Gaussian distribution of brightness constancy errors. In [19, 26, 44], probabilistic approaches are applied to obtain improved models of optical flow. In contrast to our work, these methods do not apply a fully probabilistic approach, but fall back to a maximum a-posteriori (MAP) estimate, *i.e.* minimize the underlying energy. Glocker *et al.* [18] use flow uncertainties in a dynamic Markov random field but rely on a discrete approach. The most closely related work of Chantas *et al.* [12] applies a variational-Bayes approach similar to ours. However, their method is not designed as a stand-alone flow algorithm, but only as an improved initialization in comparison to a simple Horn-Schunck approach. Therefore, the obtained results are not competitive with respect to the state of the art. Moreover, the paper does not take advantage of the probabilistic approach in the context of uncertainty measures.

3. Energy Framework

Since our joint, model-inherent uncertainty estimation developed below is broadly applicable to different kinds of energy-based optical flow approaches [3, 15, 43], we first introduce a formalization that allows us to describe previously proposed optical flow methods in a unified manner.

In the following, we estimate optical flow between an image pair $I = \{I_1, I_2\}$ and denote the estimate as $\mathbf{y} = (\mathbf{y}_{ij})_{ij} = \left(u_{ij}, v_{ij}\right)_{ij}^{\mathrm{T}}$ for pixels (i, j), $i = 1, \ldots, n$, $j = 1, \ldots, m$. Energy-based approaches estimate the optical flow as the minimizer $\mathbf{y}^\star$ of an energy function

$$E\left(\mathbf{y}; I\right) = E_{\mathrm{D}}\left(\mathbf{y}; I\right) + \lambda_{\mathrm{S}} E_{\mathrm{S}}\left(\mathbf{y}\right) \tag{1}$$

with $E_{\mathrm{D}}\left(\mathbf{y}; I\right)$ denoting a data term that encourages the flow to be consistent with input images I_1 and I_2. The spatial term $E_{\mathrm{S}}\left(\mathbf{y}\right)$ imposes a (smoothness) prior on the flow, and λ_{S} represents a trade-off parameter between the terms.

In the following, we use functions $f_{\mathrm{D}}(\cdot)$ and $f_{\mathrm{S}}(\cdot)$ to formalize violations of the assumptions underlying the chosen optical flow model. For instance, the intensity difference of corresponding pixels in I_1 and I_2 may be evaluated to model a brightness constancy assumption for the data term. So-called penalty functions $\rho_{\mathrm{D}}(\cdot)$ and $\rho_{\mathrm{S}}(\cdot)$ penalize violations of the assumptions modeled in $f_{\mathrm{D}}(\cdot)$ and $f_{\mathrm{S}}(\cdot)$. The energy terms E_{D} and E_{S} can then be described as a sum of the contributions from all pixels such that

$$E_{\mathrm{D}}\left(\mathbf{y}; I\right) = \sum_{i,j} \rho_{\mathrm{D}}\Big(f_{\mathrm{D}}\big(\mathbf{y}_{ij}; I\big)\Big) \tag{2}$$

$$E_{\mathrm{S}}\left(\mathbf{y}\right) = \sum_{i,j} \sum_{(i',j') \in S(i,j)} \rho_{\mathrm{S}}\Big(f_{\mathrm{S}}\big(\mathbf{y}_{ij}, \mathbf{y}_{i'j'}\big)\Big) \tag{3}$$

with $S(i, j)$ describing a set of neighbors of pixel (i, j).

It has been shown that the estimated flow field $\mathbf{y}^\star$ can be improved by adding a non-local term to the energy function in Eq. (1) [25, 43]. Thus, we optionally use an additional non-local term $E_{\mathrm{N}}\left(\mathbf{y}\right)$ akin to $E_{\mathrm{S}}\left(\mathbf{y}\right)$, considering an extended neighborhood $N(i, j)$. Again, $E_{\mathrm{N}}\left(\mathbf{y}\right)$ is described by a model assumption $f_{\mathrm{N}}(\cdot)$ and its corresponding penalty function $\rho_{\mathrm{N}}(\cdot)$.

Common choices for penalty functions $\rho(\cdot)$ are, *e.g.*, a quadratic function [21], a Lorentzian function [5], or a (generalized) Charbonnier function [9, 43]. Many of these functions can be described by the negative logarithm of a Gaussian Scale Mixture (GSM) [44, 46]. Thus, in the following we rely on this simple but powerful class of functions and represent the penalty terms as GSMs of L components

$$\rho\left(z\right) = -\log\left[\sum_{l=1}^{L} \pi_l \, \mathcal{N}\left(z; 0, \sigma_l^2\right)\right] \tag{4}$$

with $\mathcal{N}\left(z; \mu, \sigma\right)$ being a normal distribution with mean $\mu = 0$ and variance $\sigma = \sigma_l$; the weights $\pi_l \geq 0$ sum to 1. As we will see, GSMs also benefit our probabilistic approach.

4. Probabilistic Interpretation and Inference

We now aim to approach optical flow estimation in a probabilistic manner. As the energy function $E\left(\mathbf{y}; I\right)$ from Eq. (1) describes a Markov random field, it is easy to derive the corresponding posterior distribution in its Gibbs form as

$$p\left(\mathbf{y} \mid I\right) = \frac{1}{Z} \exp\left\{-\frac{1}{T} E(\mathbf{y}; I)\right\} \tag{5}$$

with partition function $Z \equiv Z(I, T, \lambda_{\mathrm{S}}, \lambda_{\mathrm{N}})$ and temperature T. In the following, w.l.o.g. we set $T = 1$ and introduce an additional parameter λ_{D} scaling the data term E_{D}.

To ease probabilistic inference, we use the same procedure as [43] and introduce an auxiliary flow field $\hat{\mathbf{y}}$ that allows us to decouple the non-local potential from the remaining terms of the posterior distribution in Eq. (5), *i.e.*

$$\begin{aligned} p\left(\mathbf{y}, \hat{\mathbf{y}} \mid I\right) = \frac{1}{Z} \exp\big\{ &- \lambda_{\mathrm{D}} E_{\mathrm{D}}(\mathbf{y}; I) - \lambda_{\mathrm{S}} E_{\mathrm{S}}(\mathbf{y}) \\ &- \lambda_{\mathrm{C}} E_{\mathrm{C}}\left(\mathbf{y}, \hat{\mathbf{y}}\right) - \lambda_{\mathrm{N}} E_{\mathrm{N}}\left(\hat{\mathbf{y}}\right) \big\} \end{aligned} \tag{6}$$

with $E_{\mathrm{C}}\left(\mathbf{y}, \hat{\mathbf{y}}\right) = \sum_{i,j} \left\|\mathbf{y}_{i,j} - \hat{\mathbf{y}}_{i,j}\right\|_2^2$.

The log-posterior of Eq. (6) now has terms in the form of the logarithm of a sum of exponentials, which is challenging when deriving closed-form mean-field updates below. Here, we benefit from our choice to model each penalty function $\rho(\cdot)$ as a GSM. We follow [16, 28] to retain explicit latent variables $\mathbf{h} = (\mathbf{h}_{\mathrm{D}}, \mathbf{h}_{\mathrm{S}}, \mathbf{h}_{\mathrm{N}})$, which are chosen to follow a discrete distribution and to have a 1-of-L representation. We then obtain an augmented penalty function $\rho_\gamma\left(z, \mathbf{h}_\gamma\right)$ with $\gamma \in \{\mathrm{D}, \mathrm{S}, \mathrm{N}\}$ as

$$\rho_\gamma\left(z, \mathbf{h}_\gamma\right) = -\log\left[\prod_{l=1}^{L} \pi_l^{h_{\gamma,l}} \, \mathcal{N}\left(z; 0, \sigma_l^2\right)^{h_{\gamma,l}}\right]. \tag{7}$$

At this point, the posterior $p(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h} \mid I)$ represents the probabilistic equivalent of the energy $E\left(\mathbf{y}; I\right)$ in Eq. (1).

In our probabilistic setup, the flow estimate $\mathbf{y}^\star$ is chosen to minimize the expected loss over $p(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h} \mid I)$, *i.e.*

$$\mathbf{y}^\star = \arg\min_{\tilde{\mathbf{y}}} \mathbb{E}_{p(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h} \mid I)}\left[l(\mathbf{y}, \tilde{\mathbf{y}})\right] \tag{8}$$

with $l\left(\cdot, \cdot\right)$ being a suitable loss function such as the Average End-Point Error (AEPE) [34]. The desired uncertainty measure can be obtained by considering the marginal distribution $p(\mathbf{y}_{ij} \mid I)$ of the flow estimates. In particular, we propose to use the marginal entropy at every pixel as a model-inherent uncertainty estimate of the flow prediction.

Inference. To obtain a flow estimate and the underlying marginal distributions at every pixel, we rely on an approximate inference scheme. We use variational inference [45][2]

[2] Not to be confused with variational formulations common in flow.

and approximate $p(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h} \mid I)$ with the help of a tractable distribution $q(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h}; \boldsymbol{\theta})$ and variational parameters $\boldsymbol{\theta}$.

Following a naive mean-field assumption, we choose the parametric distribution q to be factorized over $\mathbf{y}$, $\hat{\mathbf{y}}$, as well as $\mathbf{h}$. The marginal distribution of flow vectors $\mathbf{y}_{ij}$ and $\hat{\mathbf{y}}_{ij}$ is assumed to be Gaussian, *e.g.*

$$q\left(\mathbf{y}_{ij}; \boldsymbol{\theta}\right) \sim \mathcal{N}\left(\mathbf{y}_{ij}; \boldsymbol{\mu}_{ij}, \boldsymbol{\Sigma}_{ij}\right). \tag{9}$$

It is reasonable to assume the horizontal and vertical flow components to be uncorrelated [37]. Thus, the covariances $\boldsymbol{\Sigma}_{ij}$ are modeled as diagonal matrices. As in [16, 28], the distributions of the latent variables are chosen to be multinomial

$$q(\mathbf{h}_{\gamma,ij}; \boldsymbol{\theta}) = \prod_{l=1}^{L} k_{\gamma,ij,l}^{h_{\gamma,ij,l}} \tag{10}$$

so that the parameters $k_{\gamma,ij,l} \geq 0$ satisfy $\sum_l k_{\gamma,ij,l} = 1$. The approximating distribution q is then given as

$$q(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h}; \boldsymbol{\theta}) = \prod_{i,j} \Big[q\left(\mathbf{y}_{ij}; \boldsymbol{\theta}\right) \cdot q\left(\hat{\mathbf{y}}_{ij}; \boldsymbol{\theta}\right) \cdot \prod_{\gamma \in \{\mathrm{D,S,N}\}} q\left(\mathbf{h}_{\gamma,ij}; \boldsymbol{\theta}\right) \Big] \tag{11}$$

with $\boldsymbol{\theta} = \left\{ \boldsymbol{\mu}_{ij}, \hat{\boldsymbol{\mu}}_{ij}, \boldsymbol{\Sigma}_{ij}, \hat{\boldsymbol{\Sigma}}_{ij}, \mathbf{k}_{\gamma,ij} \right\}_{ij,\gamma}$.

Suitable variational parameters $\boldsymbol{\theta}^\star$ of q are determined such that the Kullback-Leibler (KL) divergence between p and its approximating distribution q is minimized, *i.e.*

$$\boldsymbol{\theta}^\star = \arg\min_{\boldsymbol{\theta}} D_{KL}\big(q(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h}; \boldsymbol{\theta}) \mid p(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h} \mid I)\big) \tag{12}$$

$$= \arg\min_{\boldsymbol{\theta}} \mathbb{E}_{q(\mathbf{y},\hat{\mathbf{y}},\mathbf{h};\boldsymbol{\theta})}\big[\log q(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h}; \boldsymbol{\theta})\big] - \mathbb{E}_{q(\mathbf{y},\hat{\mathbf{y}},\mathbf{h};\boldsymbol{\theta})}\big[\log p(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h} \mid I)\big]. \tag{13}$$

Due to the usage of explicit latent variables $\mathbf{h}$, it is possible to compute the expectations in Eq. (13) in an analytic way. While the derivation of the corresponding equations is tedious, the individual steps are elementary; see supplemental material for a more detailed explanation of the procedure.

We now estimate the flow by replacing the posterior p in Eq. (8) with its approximating distribution q. When performing Bayesian risk minimization of the AEPE, the optical flow prediction is obtained as $\mathbf{y}_{ij}^\star = \boldsymbol{\mu}_{ij}$ and, therefore, corresponds to the mode of the variational distribution q.

As $q(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h}; \boldsymbol{\theta})$ was defined in a factorized way, the corresponding marginal distribution at pixel (i, j) is given as $q\left(\mathbf{y}_{ij}; \boldsymbol{\theta}\right)$ and our proposed model-inherent uncertainty measure can be obtained as the marginal entropy

$$\Psi_{\text{ProbFlow}} = H(\mathbf{y}_{ij}) = \log\left(\det\left(\boldsymbol{\Sigma}_{ij}\right)\right) + \text{const}. \tag{14}$$

5. Specific Models

We now apply our ProbFlow approach to two specific energy functions commonly used for optical flow estimation.

5.1. Probabilistic Classic Flow

We first consider a classical Horn-Schunck-based objective based on brightness constancy as given in [43]. Following the common approach to use a first-order Taylor approximation for a linearization of the data term, we obtain

$$f_{\mathrm{D}}\left(\mathbf{y}_{ij}; I\right) = I_2\left(i + u_{ij}^0, j + v_{ij}^0\right) - I_1\left(i, j\right) + \nabla_2 I_2\left(i + u_{ij}^0, j + v_{ij}^0\right)^{\mathrm{T}}\left(\mathbf{y}_{ij} - \mathbf{y}_{ij}^0\right), \tag{15}$$

where $\mathbf{y}_{ij}^0$ is the point of approximation and $\nabla_2 I_2$ denotes the spatial derivatives of I_2. The smoothness prior in [43] assumes small flow gradients over a 4-neighborhood of horizontal and vertical flow components. The non-local term encourages smoothness over a neighborhood of size 5×5. In both cases, the smoothness assumption is represented as

$$f_{\mathrm{S}}\left(x_{ij}, x_{i'j'}\right) = f_{\mathrm{N}}\left(x_{ij}, x_{i'j'}\right) = x_{ij} - x_{i'j'}. \tag{16}$$

Akin to energy minimization approaches [43], we found the use of the non-local term crucial for obtaining accurate optical flow estimates, since mean-field inference using only the terms E_{D} and E_{S} is rather outlier-prone.

5.2. Probabilistic FlowFields

In a second setup, we aim for a probabilistic version of FlowFields [1]. We follow Bailer *et al.* and consider the energy used in the post-processing step of EpicFlow [36], which uses a data term based on gradient constancy. As proposed in [49], the image gradient terms are normalized w.r.t. the spatial derivatives, which is helpful to avoid outliers in the flow field. We obtain the linearized data assumption as

$$f_{\mathrm{D}}\left(\mathbf{y}_{ij}; I\right) = \left\| \sum_{r=1}^{3} \theta_{ij}^r \circ \Big[\nabla_2 I_2^r\left(i + u_{ij}^0, j + v_{ij}^0\right) - \nabla_2 I_1^r\left(i, j\right) + \mathbf{H}\Big(I_2^r\left(i + u_{ij}^0, j + v_{ij}^0\right)\Big)\left(\mathbf{y}_{ij} - \mathbf{y}_{ij}^0\right) \Big] \right\|_2. \tag{17}$$

Here, $r = 1, \ldots, 3$ indicates the RGB color channels, $\mathbf{H}\left(I_2^r\right)$ denotes the Hessian of I_2^r, and $\circ$ is the Hadamard product. The normalization coefficient θ_{ij}^r is given as

$$\theta_{ij}^r = \begin{pmatrix} \theta_{ij,1}^r \\ \theta_{ij,2}^r \end{pmatrix}, \; \theta_{ij,k}^r = \frac{1}{\sqrt{\left\|\mathbf{H}(I_2^r) \cdot \mathbf{e}_k\right\|_2^2 + \zeta^2}} \tag{18}$$

with 2D unit vectors $\mathbf{e}_k$ and a small constant $\zeta > 0$ [49].

The smoothness term of [36] is based on the flow gradient norm and uses additional filters of size 2×3 and 3×2 as described in [38] when calculating the flow derivatives. To keep the inference problem consistent, our approach differs slightly from [36] in that we only use the forward gradient filter described in Eq. (16) and obtain

$$f_{\mathrm{S}}\left(\mathbf{y}_{ij}, \mathbf{y}_{i'j'}\right) = \sqrt{\left(u_{ij} - u_{i'j'}\right)^2 + \left(v_{ij} - v_{i'j'}\right)^2} \tag{19}$$

for $\mathbf{y}_{i'j'}$ in a 4-neighborhood of $\mathbf{y}_{ij}$.

Following [36], a locally adaptive trade-off parameter $\lambda_{\mathrm{S}}(\mathbf{x}) = \exp(-\kappa\|\nabla_2 I_1(\mathbf{x})\|)$ is used. Similar to MAP estimation, the minimization of the KL divergence in Eq. (13) obtained from f_{D}, f_{S}, and $\lambda_{\mathrm{S}}(\mathbf{x})$ performs well in practice. Therefore, an additional non-local term can be neglected.

6. Implementation

Optical flow estimation by energy minimization is known to be far from easy. Similarly, an application of mean-field inference is non-trivial in this context. Moreover, it is essential to consider several details commonly used with energy-based approaches to obtain satisfying results [15, 43]. A summary of basic design choices as well as an extensive analysis of the influence of all described specifics can be found in the supplemental material.

Common per-pixel updates, *e.g.* [48], do not work well for the mean-field inference of Eq. (13). Instead, we keep the corresponding optimization procedure as efficient as possible and use a block-coordinate descent scheme updating flow estimates $\boldsymbol{\mu}$, variances $\boldsymbol{\Sigma}$, and latent variables $\mathbf{k}$ in an alternating manner. We derive the gradient of the KL divergence in Eq. (13), set it to zero, and obtain an update equation for each set of variables (see supplemental). As with MAP, we found a joint update of the flow predictions at all pixels to be crucial to obtaining smooth flow fields.

Parameters. To determine suitable trade-off parameters λ, it is not sufficient to follow the common approach and choose parameters that lead to the smallest AEPE on a training set [15, 43]. Instead, we evaluate the quality of both the obtained flow predictions and the uncertainty measure in order to ensure accurate flow estimates *and* meaningful entropies. To assess the quality of our uncertainty measure, we follow the approaches in [8, 24, 27, 30] and compute so-called sparsification plots. To that end, the pixels of a flow field are sorted according to the estimated uncertainties. Subsequently, an increasing percentage of the pixels is removed and the AEPE of the remaining pixels is calculated. In order to evaluate how well different uncertainty measures perform on an entire dataset, we propose to normalize the graphs, calculate the area under curve (AUC), and average over the sequences. To consider the trade-off between flow accuracy and quality of the uncertainty measure, we evaluate an F_1-score

$$\mathrm{F}_1 = \frac{\mathrm{AEPE} \cdot c\ \mathrm{AUC}}{\mathrm{AEPE} + c\ \mathrm{AUC}} \tag{20}$$

over the training data with constant c weighting the influence of the two metrics. We then determine parameters using Bayesian optimization [42] of Eq. (20).

Concerning the penalty functions, we follow the approach in [44] and learn appropriate GSM models for data likelihood, smoothness prior, as well as the non-local term from the respective training datasets or a randomly chosen subset thereof. Using manually determined variances σ_l, a simple expectation maximization algorithm is used to obtain the corresponding weights π_l. We observe that GSMs with $L = 10$ components perform well in practice. To save computational time, we resort to $L = 5$ components for our probabilistic implementation of Classic Flow.

Details. For Probabilistic Classic Flow, we follow the underlying energy approach (ClassicA, [43]) and use zero flow as an initialization; we denote the method as *ProbClassicA*. Variances are initialized as $\sigma_{\text{init}} = 1\mathrm{e}{-7}$, latent variables as $k_{\text{init}} = 1/L$. During the inference process, the parameter λ_{C} is annealed as in [43]. The parameters λ_{D}, λ_{S}, and λ_{N} are obtained with Bayesian optimization of the F_1-score (Eq. 20) using $c = 5$. As suggested by Sun *et al.* in the context of MAP, we use the variables $\hat{\boldsymbol{\mu}}$ as the flow prediction. The corresponding uncertainties are then obtained from $\hat{\boldsymbol{\Sigma}}$.

For Probabilistic FlowFields, we use the state-of-the-art FlowFields method [1] to generate sparse matches. Following Bailer *et al.*, we apply the EpicFlow [36] post-processing step with Sintel parameters to interpolate the matches and use the results to initialize our algorithm. This method will be denoted as *ProbFlowFields*. Variances and latent variables are initialized as for ProbClassicA. Trade-off parameters λ_{D}, λ_{S}, and κ are obtained again with Bayesian optimization [42]. The parameter of the F_1-score (Eq. 20) is chosen as $c = 100$ since the AEPE is significantly higher on Sintel. We will make code available for ProbClassicA as well as for ProbFlowFields.

7. Experiments & Results

In the following, we evaluate our probabilistic flow approach and assess the quality of our uncertainty measure by comparing it to different approaches from the literature.

7.1. Competing uncertainty measures

We apply existing uncertainty measures on top of the corresponding energy minimization approaches (*i.e.* ClassicA or FlowFields) in order to analyze the benefit of our combined flow prediction and uncertainty estimation. We limit ourselves to a few select methods here and give a more extensive comparison in the supplemental material.

Barron *et al.* [3] suggest a confidence measure based on the spatial gradient of the input image. The corresponding uncertainty is obtained as $\Psi_{\text{Gradient}} = -\|\nabla_2 I_1\|$ using central differences to approximate the gradient.

The learned confidence measure in [30] is based on a classifier that predicts with probability $\tilde{p}$ whether the error of the flow estimate at a certain pixel is smaller than a specified threshold ϵ_T. We use the implementation of MacAodha *et al.* and train $\Psi_{\text{Learned}} = -\tilde{p}$ as described in [30].

Method	training		test	
	AEPE	rel. chg.	AEPE	rel. chg.
Classic++ [43]	**0.285**	**-0.04**	**0.406**	**-0.07**
ClassicA [43]	0.295	>-0.01	–	–
ProbClassicA *(ours)*	0.296	0.00	0.435	0.00

Table 1. Average end-point error (AEPE) and its relative change (rel. chg.) in comparison to ProbClassicA on Middlebury.

As proposed by Bruhn and Weickert [8], we use the local energy contribution of the MAP estimate $\mathbf{y}_{\text{MAP}}$ as a model-inherent uncertainty $\Psi_{\text{Energy}} = E(\mathbf{y}_{\text{MAP}}; I)$. We apply the non-linear version of the underlying energy function as it shows an improved performance [10]. A data penalty for out-of-boundary pixels is determined on the training set.

For an additional baseline, we perform a Laplace approximation of the posterior $p(\mathbf{y}, \hat{\mathbf{y}} \mid I)$ around $\mathbf{y}_{\text{MAP}}$. With $\mathbf{H}$ denoting the Hessian of the linearized energy $E(\mathbf{y}_{\text{MAP}}; I)$, the covariance is given as $\mathbf{\Sigma}_L = \mathbf{H}^{-1}$. Similar to our approach, we obtain $\Psi_{\text{Laplace}} = -\log\left(\det\left(\mathbf{H}\right)\right) + \text{const}$.

Finally, we consider an oracle uncertainty measure Ψ_{Oracle}, for which the uncertainty of a pixel is given by its end-point error. Thus, the estimate Ψ_{Oracle} provides a bound for the best possible uncertainty estimation. Note that we consider oracle uncertainties from the predictions obtained by MAP estimation; the oracle uncertainties for our flow predictions show a very similar behavior.

7.2. ProbClassicA

We first evaluate the application of ProbFlow to the ClassicA model described in Sec. 5.1. We rely on the Middlebury dataset [2], which has frequently been used to evaluate the performance of optical flow confidence measures. In order to compare different uncertainty approaches, ground truth optical flow is needed. As the Middlebury training set includes only 8 image pairs, we need to resort to training and testing the uncertainty measures on the same data.

To compare the performance of different flow estimation algorithms, we rely on the commonly used AEPE. Table 1 gives results on Middlebury training and test. Our method (ProbClassicA) performs on par with ClassicA. On the one hand, this is to be expected as both approaches share the same energy formulation and energy-based methods use highly elaborate schemes. On the other hand, this is the first time that a fully probabilistic method achieves competitive results on a public benchmark. For completeness, we also include the related Classic++, which shows slightly better results as its median filtering step allows for a more effective outlier suppression than an additional nonlocal term [25]. The median filter cannot easily be applied to our case, but the results for ClassicA and ProbClassicA could be further improved by adding weights to the nonlocal term [43].

The evaluation of uncertainty measures is more challenging. Sparsification plots are commonly considered [8, 24, 27, 30]. To be able to compare the performance of different uncertainty measures over an entire dataset, we calculate the AUC as described in Sec. 6. However, as argued by Márquez-Valle *et al*. [31], sparsification plots do not allow to estimate how strongly an uncertainty estimate is related to the underlying pixel errors. To address this, we propose to compute the Spearman's rank correlation coefficient (CC), which estimates how well the examined uncertainty values can be mapped onto the corresponding end-point errors using an arbitrary monotonic function.

Uncertainty measure	AUC	rel. chg.	CC	rel. chg.
Gradient [3]	0.971	1.08	0.023	0.94
Laplace	0.656	0.41	0.160	0.57
Energy [8]	0.498	0.07	0.303	0.19
Learned [30]	0.496	0.06	0.324	0.13
ProbClassicA *(ours)*	**0.466**	**0.00**	**0.374**	**0.00**
Oracle	*0.255*	–	*1.000*	–

Table 2. Area under curve (AUC), Spearman's rank correlation coefficient (CC), and relative change (rel. chg.) in comparison to our uncertainty measure on the Middlebury dataset.

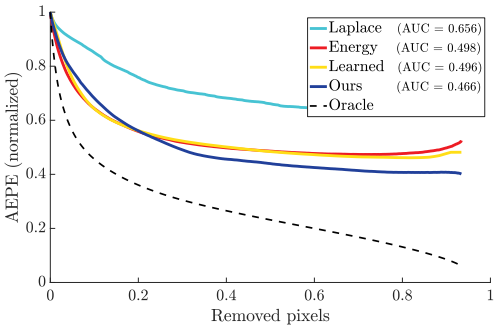


Figure 2. Sparsification plots averaged over Middlebury training.

Table 2 and the sparsification plots in Fig. 2 show that our uncertainty predictions are clearly superior in detecting the most reliable flow estimates. The gradient uncertainty has almost no ability to rank the pixels according to their accuracy. Hence, the simple consideration of input data appears insufficient. Similarly, the Laplace measure does not lead to satisfying results. Ψ_{Energy} and Ψ_{Learned} lead to very similar AUC results, whereby the learned uncertainty takes substantial time for training. Our method improves the AUC over previous ones by more than 6%. Moreover, the evaluation of the CC reveals that our uncertainty measure shows by far the highest correlation between the assigned uncertainty value and the per-pixel end-point error with a relative improvement of 13% over the learned uncertainty.

7.3. ProbFlowFields

Next, we apply our probabilistic approach to the competitive FlowFields method as described in Sec. 6. We use

Method	validation		test	
	AEPE	rel. chg.	AEPE	rel. chg.
Initialization	3.303	0.06	–	–
FlowFields [1]	3.147	<0.01	5.727†	<0.01
FlowFields⋆	3.161	0.01	–	–
ProbFlowFields *(ours)*	3.127	0.00	5.696	0.00
ProbFlowFields + BS	**3.052**	**-0.02**	**5.628**	**-0.01**

Table 3. Average end-point error (AEPE) and relative change (rel. chg.) w.r.t. to ProbFlowFields on Sintel. †FlowFields shows better results than the ones published on the website. See text for details.

Uncertainty measure	AUC	rel. chg.	CC	rel. chg.
Gradient [3]	1.022	1.57	-0.009	1.02
Laplace	0.657	0.65	0.257	0.54
Energy [8]	0.470	0.18	0.434	0.23
Learned [30]	0.474	0.19	0.451	0.20
ProbFlowFields *(ours)*	**0.398**	**0.00**	**0.563**	**0.00**
Oracle	*0.182*	–	*1.000*	–

Table 4. Area under curve (AUC), Spearman's rank correlation coefficient (CC), and relative change (rel. chg.) in comparison to our uncertainty measure on a Sintel benchmark validation set.

the more recent Sintel benchmark [11], which in comparison to the Middlebury dataset, allows to partition its more than 1000 flow sequences into training and validation sets. Exemplary flow and uncertainty estimates as well as the corresponding ground truth can be seen in Fig. 3.

Table 3 summarizes the AEPEs on our validation set and the test set of the Sintel benchmark. For comparison, we also report results of *FlowFields⋆*, based on the same consistent EpicFlow energy variant underlying ProbFlowFields, *c.f*. Sec. 5.2. Again, our estimates from ProbFlowFields are competitive with its underlying energy method and we observe results on par with the original FlowFields.

Evaluated on the Sintel test set, ProbFlowFields currently ranks 6$^{\text{th}}$ in comparison to previously published methods. Please note that the FlowFields test results shown in Table 3 are superior to the publicly available AEPE of 5.810. To suppress the effect of the random component in the matches of FlowFields and thus have a fair comparison, we have re-evaluated the original FlowFields implementation using the exact same matches as for our approach.

The performance of the competing uncertainty measures is evaluated in Table 4. Again, the Gradient and Laplace uncertainties perform considerably worse than the remaining approaches. The measures Ψ_{Learned} and Ψ_{Energy} result in similar AUC values. Our method again leads to a clear improvement of over 18%. Moreover, we also improve the CC by 20% in comparison to the second best measure. The sparsification plots in Fig. 4 show that our uncertainty measure leads to the best result for all fractions of removed pixels. Strikingly, the energy-based uncertainty as well as the Laplace measure show a strong increase of the AEPE when only a small fraction of pixels are kept. This is clearly undesirable as it indicates that the optical flow predictions are incorrect for pixels that are considered as highly reliable. Our probabilistic approach does not show this behavior.

To illustrate the benefits of our uncertainties in post-processing, we apply the fast bilateral solver [4] on top of ProbFlowFields, improving the AEPE by 2.4% and 1.2% on the validation and test set, respectively (*c.f*. Table 3). In comparison, post-processing assuming equally reliable flow yields an improvement of only 0.4% on the validation set.

In a last experiment, we evaluated the overall runtime on our Sintel validation set (Intel Core i7-3930K, 3.2 GHz, 6 cores). The average runtimes are 19.9s for FlowFields and 38.1s for ProbFlowFields, *i.e*. the additional estimation of uncertainties has an overhead of ∼1.9x. The best post-hoc uncertainty measure (learned [30]) requires 123.8s on average, thus takes significantly longer than our joint approach.

7.4. Application to motion segmentation

We now show the benefit of our uncertainty estimates when optical flow is used as a cue for motion segmentation. Current state-of-the-art methods (*e.g*. [22, 33]) build upon precomputed point trajectories, *i.e*. spatio-temporal curves that describe the individual point motion over extended periods. Ideally, such point trajectories are sampled with a regular density, they are long and reliable [6].

We build on the minimum cost multicut approach of [22] to obtain sparse motion segmentations. We follow Keuper *et al*. and generate point trajectories as in [6]. In a first setup, we use FlowFields [1] to compute forward-backward (FB) flow for all consecutive image pairs. Subsequently, points are sampled on a regular grid in the first frame and tracked as long as *(1)* their FB flows are consistent, and *(2)* the gradient magnitude of the flow is below a threshold. If tracks are stopped, new points are inserted to preserve sampling regularity unless no points can be tracked in a region.

In a second setting, we apply ProbFlowFields using parameters trained on Sintel to compute FB flow along with normalized forward uncertainties Ψ^{F}. Here, we keep the FB condition (1) and use a new condition *(3)* such that we end any track passing through location (i, j) if the pixel uncertainty Ψ^{F}_{ij} falls below an empirically determined threshold.

Our results in terms of segmentation precision, recall, F-measure, as well as the achieved trajectory density on the FBMS-59 dataset [33] are given in Table 5. For both approaches, sparse trajectories are sampled at 8 pixel distance, and framewise dense segmentations are computed using the variational approach of [32]. Moreover, we compare to the current state-of-the-art [22] on FBMS-59, which is based on Large Displacement Optical Flow (LDOF) [7].

The evaluation of sparse segmentations shows that ProbFlow allows for a higher average point density of more

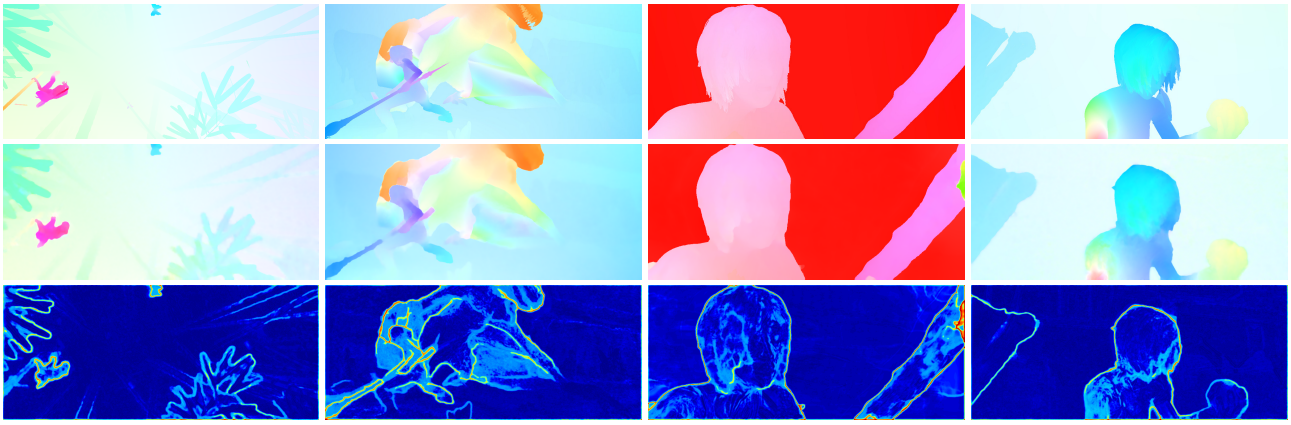


Figure 3. Examples of ground truth (top), flow predictions (middle), and uncertainty estimates (bottom) from ProbFlowFields on Sintel.

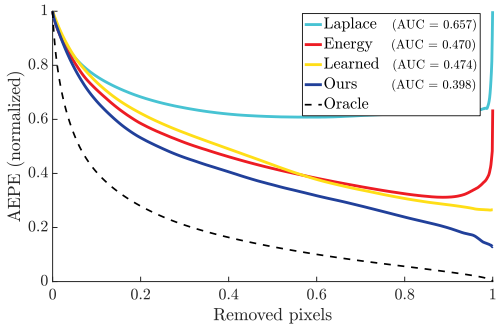


Figure 4. Sparsification plots averaged over Sintel validation.

than 1.18% compared to a maximal density of 0.89% with FlowFields [1] or LDOF [7]. Low point density usually indicates flow inconsistencies in homogeneous regions and therefore a more uneven sampling. As can be seen by the increased F-measure on sparse and especially on densified segmentations, the improvement in the trajectory computation directly translates to an improved segmentation quality.

To illustrate the complementarity of our uncertainties to the FB condition, we perform an experiment using ProbFlowFields estimates and the FB check (1), but not the corresponding uncertainties, *i.e.* omitting (3). In this case, the F-measure of the sparse matches drops significantly. Using FlowFields estimates and the post-hoc measure Ψ_{Energy}, the uncertainties again complement (1). However, the usage of ProbFlowFields yields a higher F-measure, thus showing a clear benefit also in this application.

8. Conclusion

To address the issue that optical flow estimates are not equally reliable throughout an image, we introduced ProbFlow – a probabilistic framework for the joint estimation of optical flow and its underlying uncertainty. Starting from conventional energy minimization methods, we derived the posterior distribution and used variational inference to estimate the flow and a model-inherent uncertainty. This is unlike existing uncertainty measures that detect regions of unreliable flow in a post-hoc step. We applied our approach to two different energy formulations on the Middlebury and Sintel benchmarks, where we obtain competitive flow estimates and significantly improved uncertainties. Applying our uncertainty estimates in the context of motion segmentation, we were able to discard erroneous flow estimates and generate highly reliable point trajectories.

Training set (29 seq.)	D	P	R	F
LDOF [7]	0.81%	86.73%	73.08%	79.32%
FlowFields (1+2)	0.83%	87.19%	74.33%	80.25%
FlowFields (1)	1.17%	85.10%	71.36%	77.63%
FlowFields + Ψ_{Energy}	1.17%	84.62%	72.57%	78.13%
ProbFlowFields (1+3)	1.18%	**87.68%**	**75.13%**	**80.92%**
ProbFlowFields (1)	**1.34%**	84.96%	72.14%	78.03%
FlowFields [1] dense	100%	86.14%	67.28%	75.55%
ProbFlowFields dense	100%	**87.00%**	**70.15%**	**77.67%**
Test set (30 seq.)	D	P	R	F
LDOF [7]	0.87%	**87.88%**	67.70%	76.48%
FlowFields (1+2)	0.89%	86.88%	69.74%	77.37%
ProbFlowFields (1+3)	**1.19%**	84.99%	**72.83%**	**78.44%**
FlowFields [1] dense	100%	84.38%	61.03%	70.83%
ProbFlowFields dense	100%	**85.41%**	**66.93%**	**75.05%**

Table 5. Motion segmentation results on the FBMS-59 dataset. We report point density (D), average precision (P), average recall (R), and F-measure. All results are computed for a sparse trajectory sampling at 8 pixel distance with MCe motion segmentation [22]. All results for LDOF are taken from [22].

Acknowledgments. The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP/2007–2013)/ERC Grant agreement No. 307942. We would like to thank Tobias Plötz and Jochen Gast for helpful discussions.

References

[1] C. Bailer, B. Taetz, and D. Stricker. Flow fields: Dense correspondence fields for highly accurate large displacement optical flow estimation. *ICCV*, 2015. 2, 4, 5, 7, 8

[2] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. *Int. J. Comput. Vision*, 92(1):1–31, 2011. 2, 6

[3] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *Int. J. Comput. Vision*, 12(1):43–77, 1994. 1, 2, 3, 5, 6, 7

[4] J. T. Barron and B. Poole. The fast bilateral solver. *ECCV*, 2016. 1, 7

[5] M. J. Black and P. Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Comput. Vis. Image Und.*, 63(1):75–104, 1996. 1, 3

[6] T. Brox and J. Malik. Object segmentation by long term analysis of point trajectories. *ECCV*, 2010. 7

[7] T. Brox and J. Malik. Large displacement optical flow: Descriptor matching in variational motion estimation. *IEEE T. Pattern Anal. Mach. Intell.*, 33(3):500 – 513, 2011. 7, 8

[8] A. Bruhn and J. Weickert. A confidence measure for variational optic flow methods. In R. Klette, R. Kozera, L. Noakes, and J. Weickert, editors, *Geometric Properties for Incomplete Data*, pp. 283–298. Springer, 2006. 2, 5, 6, 7

[9] A. Bruhn, J. Weickert, and C. Schnörr. Lucas/Kanade meets Horn/Schunck: Combining local and global optic flow methods. *Int. J. Comput. Vision*, 61(3):211–231, 2005. 1, 3

[10] M. Brumm, J. M. Marcinczak, and R. Grigat. Improved confidence measures for variational optical flow. *VISAPP*, 2015. 6

[11] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. *ECCV*, 2012. 2, 7

[12] G. K. Chantas, T. Gkamas, and C. Nikou. Variational-Bayes optical flow. *J. Math. Imaging Vision*, 50(3):199–213, 2014. 2

[13] Q. Chen and V. Koltun. Full flow: Optical flow estimation by global optimization over regular grids. *CVPR*, 2016. 2

[14] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. *ICCV*, 2015. 1, 2

[15] D. Fortun, P. Bouthemy, and C. Kervrann. Optical flow modeling and computation: A survey. *Comput. Vis. Image Und.*, 134:1–21, 2015. 1, 2, 3, 5

[16] J. Gast, A. Sellent, and S. Roth. Parametric object motion from blur. *CVPR*, 2016. 3, 4

[17] S. K. Gehrig and T. Scharwächter. A real-time multi-cue framework for determining optical flow confidence. *ICCV Workshops*, 2011. 2

[18] B. Glocker, N. Paragios, N. Komodakis, G. Tziritas, and N. Navab. Optical flow estimation with uncertainties through dynamic MRFs. *CVPR*, 2008. 2

[19] V. M. Govindu. Revisiting the brightness constraint: Probabilistic formulation and algorithms. *ECCV*, 2006. 2

[20] H. Haußecker and H. Spies. Motion. In B. Jähne, H. Haußecker, and P. Geißler, editors, *Handbook of Computer Vision and Applications*, vol. 2. Academic Press, 1999. 1, 2

[21] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1–3):185–203, 1981. 1, 2, 3

[22] M. Keuper, B. Andres, and T. Brox. Motion trajectory segmentation via minimum cost multicuts. *ICCV*, 2015. 2, 7, 8

[23] C. Kondermann, D. Kondermann, B. Jähne, and C. Garbe. An adaptive confidence measure for optical flows based on linear subspace projections. *DAGM*, 2007. 2

[24] C. Kondermann, R. Mester, and C. Garbe. A statistical confidence measure for optical flows. *ECCV*, 2008. 1, 2, 5, 6

[25] P. Krähenbühl and V. Koltun. Efficient nonlocal regularization for optical flow. *ECCV*, 2012. 3, 6

[26] K. Krajsek and R. Mester. Bayesian model selection for optical flow estimation. *DAGM*, 2007. 2

[27] J. Kybic and C. Nieuwenhuis. Bootstrap optical flow confidence and uncertainty measure. *Comput. Vis. Image Und.*, 115(10):1449–1462, 2011. 1, 2, 5, 6

[28] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Efficient marginal likelihood optimization in blind deconvolution. *CVPR*, 2011. 3, 4

[29] Y. Li, D. Min, M. S. Brown, M. N. Do, and J. Lu. SPM-BP: Sped-up PatchMatch belief propagation for continuous MRFs. *ICCV*, 2015. 2

[30] O. Mac Aodha, A. Humayun, M. Pollefeys, and G. J. Brostow. Learning a confidence measure for optical flow. *IEEE T. Pattern Anal. Mach. Intell.*, 35(5):1107–1120, 2013. 1, 2, 5, 6, 7

[31] P. Márquez-Valle, D. Gil, and A. Hernàndez-Sabaté. A complete confidence framework for optical flow. *ECCV Workshops and Demonstrations*, 2012. 2, 6

[32] P. Ochs and T. Brox. Object segmentation in video: A hierarchical variational approach for turning point trajectories into dense regions. *ICCV*, 2011. 7

[33] P. Ochs, J. Malik, and T. Brox. Segmentation of moving objects by long term video analysis. *IEEE T. Pattern Anal. Mach. Intell.*, 36(6):1187 – 1200, 2014. 1, 7

[34] M. Otte and H.-H. Nagel. Optical flow estimation: Advances and comparisons. *ECCV*, 1994. 3

[35] D. Piao, P. G. Menon, and O. J. Mengshoel. Computing probabilistic optical flow using Markov random fields. *Computational Modeling of Objects Presented in Images. Fundamentals, Methods, and Applications*, pp. 241–247. Springer, 2014. 2

[36] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. EpicFlow: Edge-preserving interpolation of correspondences for optical flow. *CVPR*, 2015. 1, 2, 4, 5

[37] S. Roth and M. J. Black. On the spatial statistics of optical flow. *Int. J. Comput. Vision*, 74(1):33–50, 2007. 4

[38] S. Roth and M. J. Black. Steerable random fields. *ICCV*, 2007. 4

[39] S. Roy and V. Govindu. MRF solutions for probabilistic optical flow formulations. *ICPR*, 2000. 2

[40] E. P. Simoncelli, E. H. Adelson, and D. J. Heeger. Probability distributions of optical flow. *CVPR*, 1991. 2

[41] A. Singh. An estimation-theoretic framework for image-flow computation. *ICCV*, 1990. 2

[42] J. Snoek, H. Larochelle, and R. P. Adams. Practical Bayesian optimization of machine learning algorithms. *NIPS*2012*, pp. 2951–2959. 5

[43] D. Sun, S. Roth, and M. J. Black. A quantitative analysis of current practices in optical flow estimation and the principles behind them. *Int. J. Comput. Vision*, 106(2):115–137, 2014. 1, 2, 3, 4, 5, 6

[44] D. Sun, S. Roth, J. P. Lewis, and M. J. Black. Learning optical flow. *ECCV*, 2008. 2, 3, 5

[45] M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305, 2008. 3

[46] M. J. Wainwright and E. P. Simoncelli. Scale mixtures of gaussians and the statistics of natural images. *NIPS*1999*. 3

[47] A. Wedel, A. Meißner, C. Rabe, U. Franke, and D. Cremers. Detection and segmentation of independently moving objects from dense scene flow. *EMMCVPR*, 2009. 1

[48] J. Winn and C. M. Bishop. Variational message passing. *J. Mach. Learn. Res.*, 6:661–694, 2005. 5

[49] H. Zimmer, A. Bruhn, and J. Weickert. Optic flow in harmony. *Int. J. Comput. Vision*, 93(3):368–388, 2011. 4

ProbFlow: Joint Optical Flow and Uncertainty Estimation
– Supplemental Material –

Anne S. Wannenwetsch[1] Margret Keuper[2] Stefan Roth[1]
[1]TU Darmstadt [2]University of Mannheim

Preface. In this supplemental material we derive update equations for the mean-field inference in Eq. (13) and show a proof for the solution of the Bayesian risk minimization in Eq. (8). We give further implementation details of ProbClassicA and ProbFlowFields, and present an analysis considering different design choices. Finally, we evaluate the performance of additional uncertainty measures and apply, for completeness, ProbFlowFields on the Middlebury benchmark [2].

A. Mean-field Update Equations

In the following, we show how to derive the mean-field update equations for ProbClassicA. Update equations for ProbFlowFields can be obtained similarly.

Notation. Note that strictly speaking the latent variables are given as $\mathbf{h} = (\mathbf{h}_{\gamma,ij,l})_{\gamma,ij}$ with $\gamma \in \{\mathrm{D}, \mathrm{S}_1, \dots, \mathrm{S}_p, \mathrm{N}_1, \dots, \mathrm{N}_q\}$, $p = |S(i,j)|$, $q = |N(i,j)|$ as we have separate latent variables for all penalty functions. In the following, we therefore use the notation $\mathbf{h}_{S_e}$, $\mathbf{h}_{N_e}$ and $f_{S_e}(\cdot)$, $f_{N_e}(\cdot)$ for $e \in S(i,j)$ and $e \in N(i,j)$, respectively. The flow vector of the corresponding neighboring pixel is denoted as $\mathbf{y}_e$. Moreover, GSM parameters π_l and σ_l differ for data, smoothness, and non-local potentials. For better readability, we drop indices D, S, and N that explicitly distinguish between the different GSMs.

Variational objective. As shown in Eq. (13), variational parameters $\boldsymbol{\theta}^\star$ are determined to minimize the Kullback-Leibler divergence between posterior p and its approximating distribution q, *i.e.*

$$\boldsymbol{\theta}^\star = \arg\min_{\boldsymbol{\theta}} D_{KL}\big(q(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h}; \boldsymbol{\theta}) \mid p(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h} \mid I)\big) \tag{22a}$$

$$= \arg\min_{\boldsymbol{\theta}} \underbrace{\mathbb{E}_{q(\mathbf{y},\hat{\mathbf{y}},\mathbf{h};\boldsymbol{\theta})}\big[\log q(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h}; \boldsymbol{\theta})\big]}_{\text{①, entropy term}} - \underbrace{\mathbb{E}_{q(\mathbf{y},\hat{\mathbf{y}},\mathbf{h};\boldsymbol{\theta})}\big[\log p(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h} \mid I)\big]}_{\text{②}}. \tag{22b}$$

Recall that we defined the variational distribution q in Eq. (11) as

$$q(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h}; \boldsymbol{\theta}) = \prod_{i,j} \Big[q\left(\mathbf{y}_{ij}; \boldsymbol{\theta}\right) \cdot q\left(\hat{\mathbf{y}}_{ij}; \boldsymbol{\theta}\right) \cdot \prod_{\gamma} q\left(\mathbf{h}_{\gamma,ij}; \boldsymbol{\theta}\right) \Big]. \tag{23}$$

Then, the entropy term in ① can be split up as follows:

$$\mathbb{E}_{q(\mathbf{y},\hat{\mathbf{y}},\mathbf{h};\boldsymbol{\theta})}\Big[\log q(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h}; \boldsymbol{\theta})\Big] = \mathbb{E}_{q(\mathbf{y},\hat{\mathbf{y}},\mathbf{h};\boldsymbol{\theta})}\left[\sum_{i,j} \log q\left(\mathbf{y}_{ij}; \boldsymbol{\theta}\right) + \sum_{i,j} \log q\left(\hat{\mathbf{y}}_{ij}; \boldsymbol{\theta}\right) + \sum_{i,j} \sum_{\gamma} \log q\left(\mathbf{h}_{\gamma,ij}; \boldsymbol{\theta}\right) \right] \tag{24a}$$

$$= \sum_{i,j} \mathbb{E}_{q(\mathbf{y}_{ij};\boldsymbol{\theta})} \log q\left(\mathbf{y}_{ij}; \boldsymbol{\theta}\right) + \sum_{i,j} \mathbb{E}_{q(\hat{\mathbf{y}}_{ij};\boldsymbol{\theta})} \log q\left(\hat{\mathbf{y}}_{ij}; \boldsymbol{\theta}\right) + \sum_{i,j} \sum_{\gamma} \mathbb{E}_{q(\mathbf{h}_{\gamma,ij};\boldsymbol{\theta})} \log q\left(\mathbf{h}_{\gamma,ij}; \boldsymbol{\theta}\right). \tag{24b}$$

Using the well-known entropy of a Gaussian and a multinomial distribution, we obtain

$$\mathbb{E}_{q(\mathbf{y},\hat{\mathbf{y}},\mathbf{h};\boldsymbol{\theta})}\Big[\log q(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h}; \boldsymbol{\theta})\Big] = -\frac{1}{2} \sum_{i,j} \log\left(\det(\boldsymbol{\Sigma}_{ij})\right) - \frac{1}{2} \sum_{i,j} \log\left(\det(\hat{\boldsymbol{\Sigma}}_{ij})\right) + \sum_{i,j} \sum_{\gamma} \sum_{l} k_{\gamma,ij,l} \ \log \ k_{\gamma,ij,l} + \text{const.} \tag{25}$$

In order to evaluate the term ② in Eq. (22b), it is necessary to compute

$$
\begin{aligned}
&\log p(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h} \mid I) \\
&= \log \left(\frac{1}{Z} \prod_{i,j} \left[\prod_l \pi_l^{h_{\mathrm{D},ij,l}} \mathcal{N}\Big(f_{\mathrm{D}}\big(\mathbf{y}_{ij}; I\big); 0, \sigma_l^2 \Big)^{h_{\mathrm{D},ij,l}} \right]^{\lambda_{\mathrm{D}}} \cdot \prod_{e \in S(i,j)} \left[\prod_l \pi_l^{h_{\mathrm{S}_e,ij,l}} \mathcal{N}\Big(f_{\mathrm{S}_e}\big(\mathbf{y}_{ij}, \mathbf{y}_e\big); 0, \sigma_l^2 \Big)^{h_{\mathrm{S}_e,ij,l}} \right]^{\lambda_{\mathrm{S}}} \right. \\
&\qquad \left. \cdot \exp\Big[- f_{\mathrm{C}}\big(\mathbf{y}_{ij}, \hat{\mathbf{y}}_{ij}\big)^2 \Big]^{\lambda_{\mathrm{C}}} \cdot \prod_{e \in N(i,j)} \left[\prod_l \pi_l^{h_{\mathrm{N}_e,ij,l}} \mathcal{N}\Big(f_{\mathrm{N}_e}\big(\hat{\mathbf{y}}_{ij}, \hat{\mathbf{y}}_e\big); 0, \sigma_l^2 \Big)^{h_{\mathrm{N}_e,ij,l}} \right]^{\lambda_{\mathrm{N}}} \right) \qquad (26a) \\
&= \sum_{i,j} \left[\lambda_{\mathrm{D}} \sum_l h_{\mathrm{D},ij,l} \Big(\log \pi_l + \log \mathcal{N}\Big(f_{\mathrm{D}}\big(\mathbf{y}_{ij}; I\big); 0, \sigma_l^2 \Big) \Big) + \lambda_{\mathrm{S}} \sum_{e \in S(i,j)} \sum_l h_{\mathrm{S}_e,ij,l} \Big(\log \pi_l + \log \mathcal{N}\Big(f_{\mathrm{S}_e}\big(\mathbf{y}_{ij}, \mathbf{y}_e\big); 0, \sigma_l^2 \Big) \Big) \right. \\
&\qquad \left. - \lambda_C \, f_{\mathrm{C}}\big(\mathbf{y}_{ij}, \hat{\mathbf{y}}_{ij}\big)^2 + \lambda_{\mathrm{N}} \sum_{e \in N(i,j)} \sum_l h_{\mathrm{N}_e,ij,l} \Big(\log \pi_l + \log \mathcal{N}\Big(f_{\mathrm{N}_e}\big(\hat{\mathbf{y}}_{ij}, \hat{\mathbf{y}}_e\big); 0, \sigma_l^2 \Big) \Big) \right] - \log Z \qquad (26b) \\
&= \sum_{i,j} \left[\lambda_{\mathrm{D}} \sum_l h_{\mathrm{D},ij,l} \left(\log \pi_l - \log \sigma_l - \frac{f_{\mathrm{D}}\big(\mathbf{y}_{ij}; I\big)^2}{2\sigma_l^2} \right) + \lambda_{\mathrm{S}} \sum_{e \in S(i,j)} \sum_l h_{\mathrm{S}_e,ij,l} \left(\log \pi_l - \log \sigma_l - \frac{f_{\mathrm{S}_e}\big(\mathbf{y}_{ij}, \mathbf{y}_e\big)^2}{2\sigma_l^2} \right) \right. \\
&\qquad \left. - \lambda_{\mathrm{C}} \, f_{\mathrm{C}}\big(\mathbf{y}_{ij}, \hat{\mathbf{y}}_{ij}\big)^2 + \lambda_{\mathrm{N}} \sum_{e \in N(i,j)} \sum_l h_{\mathrm{N}_e,ij,l} \left(\log \pi_l - \log \sigma_l - \frac{f_{\mathrm{N}_e}\big(\hat{\mathbf{y}}_{ij}, \hat{\mathbf{y}}_e\big)^2}{2\sigma_l^2} \right) \right] + \text{const}, \qquad (26c)
\end{aligned}
$$

where we have defined $f_{\mathrm{C}}\big(\mathbf{y}_{ij}, \hat{\mathbf{y}}_{ij}\big) = \|\mathbf{y}_{ij} - \hat{\mathbf{y}}_{ij}\|_2$. We now take the expectation over $\mathbf{h}$ and simplify the remaining expectations as

$$
\begin{aligned}
&\mathbb{E}_{q(\mathbf{y},\hat{\mathbf{y}},\mathbf{h};\boldsymbol{\theta})} \log p(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h} \mid I) \\
&= \mathbb{E}_{q(\mathbf{y},\hat{\mathbf{y}};\boldsymbol{\theta})} \sum_{i,j} \left[\lambda_{\mathrm{D}} \sum_l k_{\mathrm{D},ij,l} \left(\log \pi_l - \log \sigma_l - \frac{f_{\mathrm{D}}(\mathbf{y}_{ij}; I)^2}{2\sigma_l^2} \right) \right. \\
&\qquad + \lambda_{\mathrm{S}} \sum_{e \in S(i,j)} \sum_l k_{\mathrm{S}_e,ij,l} \left(\log \pi_l - \log \sigma_l - \frac{f_{\mathrm{S}_e}\big(\mathbf{y}_{ij}, \mathbf{y}_e\big)^2}{2\sigma_l^2} \right) \\
&\qquad \left. - \lambda_{\mathrm{C}} \, f_{\mathrm{C}}\big(\mathbf{y}_{ij}, \hat{\mathbf{y}}_{ij}\big)^2 + \lambda_{\mathrm{N}} \sum_{e \in N(i,j)} \sum_l k_{\mathrm{N}_e,ij,l} \left(\log \pi_l - \log \sigma_l - \frac{f_{\mathrm{N}_e}\big(\hat{\mathbf{y}}_{ij}, \hat{\mathbf{y}}_e\big)^2}{2\sigma_l^2} \right) \right] + \text{const} \qquad (27a) \\
&= \sum_{i,j} \left[\lambda_{\mathrm{D}} \sum_l k_{\mathrm{D},ij,l} \Big(\log \pi_l - \log \sigma_l \Big) + \lambda_{\mathrm{S}} \sum_{e \in S(i,j)} \sum_l k_{\mathrm{S}_e,ij,l} \Big(\log \pi_l - \log \sigma_l \Big) \right. \\
&\qquad \left. + \lambda_{\mathrm{N}} \sum_{e \in N(i,j)} \sum_l k_{\mathrm{N}_e,ij,l} \Big(\log \pi_l - \log \sigma_l \Big) \right] \\
&\quad - \sum_{i,j} \left[\lambda_{\mathrm{D}} \sum_l \frac{k_{\mathrm{D},ij,l}}{2\sigma_l^2} \underbrace{\mathbb{E}_{q(\mathbf{y};\boldsymbol{\theta})} \, f_{\mathrm{D}}\big(\mathbf{y}_{ij}; I\big)^2}_{③,\, g_{\mathrm{D}}} + \lambda_{\mathrm{S}} \sum_{e \in S(i,j)} \sum_l \frac{k_{\mathrm{S}_e,ij,l}}{2\sigma_l^2} \underbrace{\mathbb{E}_{q(\mathbf{y};\boldsymbol{\theta})} \, f_{\mathrm{S}_e}\big(\mathbf{y}_{ij}, \mathbf{y}_e\big)^2}_{④,\, g_{\mathrm{S}_e}} \right. \\
&\qquad \left. + \lambda_{\mathrm{C}} \underbrace{\mathbb{E}_{q(\mathbf{y},\hat{\mathbf{y}};\boldsymbol{\theta})} \, f_{\mathrm{C}}\big(\mathbf{y}_{ij}, \hat{\mathbf{y}}_{ij}\big)^2}_{⑤,\, g_{\mathrm{C}}} + \lambda_{\mathrm{N}} \sum_{e \in N(i,j)} \sum_l \frac{k_{\mathrm{N}_e,ij,l}}{2\sigma_l^2} \underbrace{\mathbb{E}_{q(\hat{\mathbf{y}};\boldsymbol{\theta})} \, f_{\mathrm{N}_e}\big(\hat{\mathbf{y}}_{ij}, \hat{\mathbf{y}}_e\big)^2}_{⑥,\, g_{\mathrm{N}_e}} \right] + \text{const}. \qquad (27b)
\end{aligned}
$$

To solve the expectation value w.r.t. the linearized brightness constancy in ③, we define $a = I_2\left(i + u_{ij}^0, j + v_{ij}^0\right) - I_1\left(i, j\right)$ and $\mathbf{b} = \nabla_2 I_2\left(i + u_{ij}^0, j + v_{ij}^0\right)^{\mathrm{T}}$. Then we have that

$$
\begin{aligned}
g_{\mathrm{D}}\left(\boldsymbol{\mu}_{ij}, \boldsymbol{\Sigma}_{ij}; I\right) &= \mathbb{E}_{q(\mathbf{y};\boldsymbol{\theta})}\ f_{\mathrm{D}}\left(\mathbf{y}_{ij}; I\right)^2 && \text{(28a)}\\
&= \mathbb{E}_{q(\mathbf{y}_{ij};\boldsymbol{\theta})}\left[\left(a + \mathbf{b}^{\mathrm{T}}\left(\mathbf{y}_{ij} - \mathbf{y}_{ij}^0\right)\right)^2\right] && \text{(28b)}\\
&= \mathbb{E}_{q(\mathbf{y}_{ij};\boldsymbol{\theta})}\left[a^2 + 2a\mathbf{b}^{\mathrm{T}}\left(\mathbf{y}_{ij} - \mathbf{y}_{ij}^0\right) + \left(\mathbf{y}_{ij} - \mathbf{y}_{ij}^0\right)^{\mathrm{T}}\left(\mathbf{b}\mathbf{b}^{\mathrm{T}}\right)\left(\mathbf{y}_{ij} - \mathbf{y}_{ij}^0\right)\right] && \text{(28c)}\\
&= a^2 + 2a\mathbf{b}^{\mathrm{T}}\left(\boldsymbol{\mu}_{ij} - \mathbf{y}_{ij}^0\right) + \left(\boldsymbol{\mu}_{ij} - \mathbf{y}_{ij}^0\right)^{\mathrm{T}}\left(\mathbf{b}\mathbf{b}^{\mathrm{T}}\right)\left(\boldsymbol{\mu}_{ij} - \mathbf{y}_{ij}^0\right) + \mathrm{Tr}\left(\mathbf{b}\mathbf{b}^{\mathrm{T}}\boldsymbol{\Sigma}_{i,j}\right). && \text{(28d)}
\end{aligned}
$$

We solve the expectation value g_{S_e} in ④ for an exemplary function $f_{\mathrm{S}_e}\left(\mathbf{y}_{ij}, \mathbf{y}_e\right) = u_{ij} - u_e$. All remaining terms as well as the terms g_{N_e} in ⑥ can be resolved in the same manner. Using $\mathbf{A}_1 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$, it holds that

$$
\begin{aligned}
g_{\mathrm{S}_e}\left(\boldsymbol{\mu}_{ij}, \boldsymbol{\Sigma}_{ij}, \boldsymbol{\mu}_e, \boldsymbol{\Sigma}_e\right) &= \mathbb{E}_{q(\mathbf{y};\boldsymbol{\theta})}\ f_{\mathrm{S}_e}\left(\mathbf{y}_{ij}, \mathbf{y}_e\right)^2 && \text{(29a)}\\
&= \mathbb{E}_{q(\mathbf{y}_e;\boldsymbol{\theta})}\ \mathbb{E}_{q(\mathbf{y}_{ij};\boldsymbol{\theta})}\left[\left(\mathbf{y}_{ij} - \mathbf{y}_e\right)^{\mathrm{T}}\mathbf{A}_1\left(\mathbf{y}_{ij} - \mathbf{y}_e\right)\right] && \text{(29b)}\\
&= \mathbb{E}_{q(\mathbf{y}_e;\boldsymbol{\theta})}\left[\left(\boldsymbol{\mu}_{ij} - \mathbf{y}_e\right)^{\mathrm{T}}\mathbf{A}_1\left(\boldsymbol{\mu}_{ij} - \mathbf{y}_e\right) + \mathrm{Tr}\left(\mathbf{A}_1\boldsymbol{\Sigma}_{ij}\right)\right] && \text{(29c)}\\
&= \left(\boldsymbol{\mu}_{ij} - \boldsymbol{\mu}_e\right)^{\mathrm{T}}\mathbf{A}_1\left(\boldsymbol{\mu}_{ij} - \boldsymbol{\mu}_e\right) + \mathrm{Tr}\left(\mathbf{A}_1\boldsymbol{\Sigma}_{ij}\right) + \mathrm{Tr}\left(\mathbf{A}_1\boldsymbol{\Sigma}_e\right) && \text{(29d)}\\
&= \left(\mu_{ij}^{(1)} - \mu_e^{(1)}\right)^2 + \left(\boldsymbol{\Sigma}_{ij}\right)_{1,1} + \left(\boldsymbol{\Sigma}_e\right)_{1,1} && \text{(29e)}
\end{aligned}
$$

with $\mu_{ij}^{(1)}$ denoting the first (*i.e.*, horizontal) component of the mean flow vector at pixel (i, j). The term g_{C} in ⑤ can be determined as

$$
\begin{aligned}
g_{\mathrm{C}}\left(\boldsymbol{\mu}_{ij}, \boldsymbol{\Sigma}_{ij}, \hat{\boldsymbol{\mu}}_{ij}, \hat{\boldsymbol{\Sigma}}_{ij}\right) &= \mathbb{E}_{q(\mathbf{y},\hat{\mathbf{y}};\boldsymbol{\theta})}\ f_{\mathrm{C}}\left(\mathbf{y}_{ij}, \hat{\mathbf{y}}_{ij}\right)^2 && \text{(30a)}\\
&= \mathbb{E}_{q(\mathbf{y}_{ij};\boldsymbol{\theta})}\ \mathbb{E}_{q(\hat{\mathbf{y}}_{ij};\boldsymbol{\theta})}\left[\left(\mathbf{y}_{ij} - \hat{\mathbf{y}}_{ij}\right)^{\mathrm{T}}\left(\mathbf{y}_{ij} - \hat{\mathbf{y}}_{ij}\right)\right] && \text{(30b)}\\
&= \mathbb{E}_{q(\hat{\mathbf{y}}_{ij};\boldsymbol{\theta})}\left[\left(\boldsymbol{\mu}_{ij} - \hat{\mathbf{y}}_{ij}\right)^{\mathrm{T}}\left(\boldsymbol{\mu}_{ij} - \hat{\mathbf{y}}_{ij}\right) + \mathrm{Tr}\left(\boldsymbol{\Sigma}_{ij}\right)\right] && \text{(30c)}\\
&= \left(\boldsymbol{\mu}_{ij} - \hat{\boldsymbol{\mu}}_{ij}\right)^{\mathrm{T}}\left(\boldsymbol{\mu}_{ij} - \hat{\boldsymbol{\mu}}_{ij}\right) + \mathrm{Tr}\left(\boldsymbol{\Sigma}_{ij}\right) + \mathrm{Tr}\left(\hat{\boldsymbol{\Sigma}}_{ij}\right). && \text{(30d)}
\end{aligned}
$$

Update equations. To obtain update equations, we compute the derivative of the KL divergence in Eq. (22b), set it to zero, and solve for the desired variable. Please note that update equations for boundary pixels may slightly differ from the ones shown below. From now on, spatial derivatives of I are denoted as I_x and I_y, the temporal derivative is given as I_t. Moreover, $\mathrm{diag}(\cdot)$ represents a diagonal matrix and we define vectors $\mathbf{K}_\gamma = \left(\sum_l \frac{\mathbf{k}_{\gamma,ij,l}}{\sigma_l^2}\right)_{ij}$.

As the update of each mean flow estimate $\boldsymbol{\mu}_{ij}$ depends on other entries of $\boldsymbol{\mu}$, it is desirable to jointly solve for all components of the flow field. Therefore, $\boldsymbol{\mu}$ is obtained as the solution of a linear equation system, *c.f.* [50, 37], such that

$$
\mathbf{A}\mathbf{x} = \mathbf{b}, \quad \mathbf{x} = \left(\mu_{11}^{(1)}, \ldots, \mu_{nm}^{(1)}, \mu_{11}^{(2)}, \ldots, \mu_{nm}^{(2)}\right)^{\mathrm{T}}, \quad \mathbf{A} = \mathbf{A}_{\mathrm{D}} + \mathbf{A}_{\mathrm{S}} + \mathbf{A}_{\mathrm{C}}, \quad \mathbf{b} = \mathbf{b}_{\mathrm{D}} + \mathbf{b}_{\mathrm{S}} + \mathbf{b}_{\mathrm{C}}. \tag{31}
$$

The components of the linear equation system are determined as

$$
\begin{aligned}
\mathbf{A}_{\mathrm{D}} &= \lambda_{\mathrm{D}}\begin{pmatrix} \mathrm{diag}\left(\mathbf{K}_{\mathrm{D}}\right)\mathrm{diag}\left(I_x^2\right) & \mathrm{diag}\left(\mathbf{K}_{\mathrm{D}}\right)\mathrm{diag}\left(I_x \cdot I_y\right) \\ \mathrm{diag}\left(\mathbf{K}_{\mathrm{D}}\right)\mathrm{diag}\left(I_x \cdot I_y\right) & \mathrm{diag}\left(\mathbf{K}_{\mathrm{D}}\right)\mathrm{diag}\left(I_y^2\right) \end{pmatrix}, && \text{(32a)}\\
\mathbf{A}_{\mathrm{S}} &= \lambda_{\mathrm{S}}\begin{pmatrix} \mathbf{F}_1^{\mathrm{T}}\,\mathrm{diag}\left(\mathbf{K}_{\mathrm{S}_1}\right)\mathbf{F}_1 + \mathbf{F}_2^{\mathrm{T}}\,\mathrm{diag}\left(\mathbf{K}_{\mathrm{S}_2}\right)\mathbf{F}_2 & \mathbf{0} \\ \mathbf{0} & \mathbf{F}_1^{\mathrm{T}}\,\mathrm{diag}\left(\mathbf{K}_{\mathrm{S}_3}\right)\mathbf{F}_1 + \mathbf{F}_2^{\mathrm{T}}\,\mathrm{diag}\left(\mathbf{K}_{\mathrm{S}_4}\right)\mathbf{F}_2 \end{pmatrix}, && \text{(32b)}\\
\mathbf{A}_{\mathrm{C}} &= 2\lambda_{\mathrm{C}}\mathbf{I}, && \text{(32c)}\\
\mathbf{b}_{\mathrm{D}} &= \mathbf{A}_D\mathbf{y}_0 - \lambda_{\mathrm{D}}\begin{pmatrix} \mathrm{diag}\left(\mathbf{K}_{\mathrm{D}}\right)\mathrm{diag}\left(I_x \cdot I_t\right)\mathbf{1} \\ \mathrm{diag}\left(\mathbf{K}_{\mathrm{D}}\right)\mathrm{diag}\left(I_y \cdot I_t\right)\mathbf{1} \end{pmatrix}, \ \mathbf{b}_{\mathrm{S}} = \mathbf{0}, \ \mathbf{b}_{\mathrm{C}} = 2\lambda_{\mathrm{C}}\hat{\boldsymbol{\mu}}. && \text{(32d)}
\end{aligned}
$$

Here, $\mathbf{F}_1$ and $\mathbf{F}_2$ represent filter matrices corresponding to the derivative filters $\mathbf{H}_1 = \begin{bmatrix}1, -1\end{bmatrix}^{\mathrm{T}}$ and $\mathbf{H}_2 = \begin{bmatrix}1, -1\end{bmatrix}$, which are used in $f_{\mathrm{S}_e}\left(\mathbf{y}_{ij}, \mathbf{y}_e\right)$, *c.f.* [37]. $\mathbf{I}$ is the identity matrix and $\mathbf{0}$ is a matrix of all zeros.

When updating the auxiliary flow means $\hat{\boldsymbol{\mu}}$, a 5×5 neighborhood has to be considered. Therefore, a joint update of all estimates is computationally expensive and we follow [43] assuming fixed values for neighboring pixels, *i.e.*

$$\hat{\boldsymbol{\mu}}_{ij,t} = \begin{pmatrix}\hat{\mu}^{(1)}_{ij,t} \\ \hat{\mu}^{(2)}_{ij,t}\end{pmatrix}, \quad \hat{\mu}^{(k)}_{ij,t} = \frac{2\lambda_{\mathrm{C}}\, \mu^{(k)}_{ij} + \lambda_{\mathrm{N}} \sum_{e \in N^k(i,j)} \left(\mathbf{K}_{N^k_e}\right)_{ij,t-1} \cdot \hat{\mu}^{(k)}_{e,t-1}}{2\lambda_{\mathrm{C}} + \lambda_{\mathrm{N}} \sum_{e \in N^k(i,j)} \left(\mathbf{K}_{N^k_e}\right)_{ij,t-1}}. \tag{33}$$

Here, $N^k(i,j)$ represents the set of neighbors in terms of the k^{th} optical flow component.

For the flow variances $\boldsymbol{\Sigma} = \begin{pmatrix}\Sigma_1 & 0 \\ 0 & \Sigma_2\end{pmatrix}$ we derive a closed-form update dependent only on the latent variables $\mathbf{k}$ with

$$\Sigma_1 = \left(\lambda_{\mathrm{D}}\, \mathrm{diag}\left(I_x^2\right) \cdot \mathbf{K}_{\mathrm{D}} + \lambda_{\mathrm{S}} \left[\mathrm{abs}\left(\mathbf{F}_1^{\mathrm{T}}\right) \cdot \mathbf{K}_{\mathrm{S}_1} + \mathrm{abs}\left(\mathbf{F}_2^{\mathrm{T}}\right) \cdot \mathbf{K}_{\mathrm{S}_2}\right] + 2\lambda_{\mathrm{C}}\right)^{-1} \tag{34a}$$

$$\text{and} \qquad \Sigma_2 = \left(\lambda_{\mathrm{D}}\, \mathrm{diag}\left(I_y^2\right) \cdot \mathbf{K}_{\mathrm{D}} + \lambda_{\mathrm{S}} \left[\mathrm{abs}\left(\mathbf{F}_1^{\mathrm{T}}\right) \cdot \mathbf{K}_{\mathrm{S}_3} + \mathrm{abs}\left(\mathbf{F}_2^{\mathrm{T}}\right) \cdot \mathbf{K}_{\mathrm{S}_4}\right] + 2\lambda_{\mathrm{C}}\right)^{-1}, \tag{34b}$$

where the absolute value function $\mathrm{abs}(\cdot)$ is applied element-wise.

We assume fixed neighboring values also for the update of the auxiliary flow variances $\hat{\boldsymbol{\Sigma}}$, and obtain

$$\hat{\boldsymbol{\Sigma}}_{ij,t} = \begin{pmatrix}\hat{\Sigma}^{(1)}_{ij,t} & 0 \\ 0 & \hat{\Sigma}^{(2)}_{ij,t}\end{pmatrix}, \quad \hat{\Sigma}^{(k)}_{ij,t} = \frac{1}{2\lambda_{\mathrm{C}} + \lambda_{\mathrm{N}} \sum_{e \in N^k(i,j)} \left(\mathbf{K}_{N^k_e}\right)_{ij,t-1}}. \tag{35}$$

To derive an update equation for a latent variable $\mathbf{k}_{\gamma,ij}$, we need to consider a Lagrangian function including the KL divergence in Eq. (22b) as well as the constraint $\sum_l k_{\gamma,ij,l} = 1$. Solving the resulting linear equation system analytically gives us, *e.g.*,

$$k_{\mathrm{D},ij,l} = \left(\frac{\pi_l}{\sigma_l}\right)^{\lambda_{\mathrm{D}}} \exp\left[-\lambda_{\mathrm{D}} \frac{g_{\mathrm{D}}\left(\boldsymbol{\mu}_{ij}, \boldsymbol{\Sigma}_{ij}; I\right)}{2\sigma_l^2}\right] \cdot Z_{\mathrm{D},ij} \tag{36a}$$

$$\text{with} \qquad Z_{\mathrm{D},ij} = \left(\sum_{l=1}^{L} \left(\frac{\pi_l}{\sigma_l}\right)^{\lambda_{\mathrm{D}}} \exp\left[-\lambda_{\mathrm{D}} \frac{g_{\mathrm{D}}\left(\boldsymbol{\mu}_{ij}, \boldsymbol{\Sigma}_{ij}; I\right)}{2\sigma_l^2}\right]\right)^{-1} \tag{36b}$$

for the latent variables of the data term using the expectation values $g_{\mathrm{D}}\left(\boldsymbol{\mu}_{ij}, \boldsymbol{\Sigma}_{ij}; I\right)$ as derived in Eq. (28d). Update equations for the remaining latent variables are derived similarly.

B. Bayesian Risk Minimization

We aim to show that the solution of the Bayesian risk minimization in Eq. (8) is given as $\mathbf{y}^\star_{ij} = \boldsymbol{\mu}_{ij}$ when replacing the posterior p with its approximating distribution q and using the Average End-Point Error (AEPE) as a loss function.

Recall that the AEPE is defined as $l\left(\mathbf{y}, \tilde{\mathbf{y}}\right) = \sum_{i,j} \ell\left(\mathbf{y}_{ij}, \tilde{\mathbf{y}}_{ij}\right) = \sum_{i,j} \left\|\mathbf{y}_{ij} - \tilde{\mathbf{y}}_{ij}\right\|_2$ with

$$\begin{aligned}
\nabla_2\, \ell\left(\mathbf{a} - \mathbf{x}, \tilde{\mathbf{x}}\right) &= \left(\mathbf{a} - \mathbf{x} - \tilde{\mathbf{x}}\right) / \left\|\mathbf{a} - \mathbf{x} - \tilde{\mathbf{x}}\right\|_2 && (37a)\\
&= -\left(\mathbf{x} - \left(\mathbf{a} - \tilde{\mathbf{x}}\right)\right) / \left\|\mathbf{x} - \left(\mathbf{a} - \tilde{\mathbf{x}}\right)\right\|_2 && (37b)\\
&= -\nabla_2\, \ell\left(\mathbf{x}, \mathbf{a} - \tilde{\mathbf{x}}\right) && (37c)
\end{aligned}$$

for arbitrary $\mathbf{a} \in \mathbb{R}^2$. W.l.o.g. we minimize the expected risk of $l(\mathbf{y}, \tilde{\mathbf{y}})$ and therefore set $f(\tilde{\mathbf{y}}) = \mathbb{E}_{q(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h}; \boldsymbol{\theta})}\left[l(\mathbf{y}, \tilde{\mathbf{y}})\right]$. Note that we omit the variational parameters $\boldsymbol{\theta}$ in the following for brevity. Using the properties of q, we obtain

$$\begin{aligned}
f(\tilde{\mathbf{y}}) &= \int_{\mathcal{Y}} \int_{\hat{\mathcal{Y}}} \sum_{\mathcal{H}} q(\mathbf{y}, \hat{\mathbf{y}}, \mathbf{h}) \cdot l\left(\mathbf{y}, \tilde{\mathbf{y}}\right) d\mathbf{y}\, d\hat{\mathbf{y}} && (38a)\\
&\overset{\text{q fac.}}{=} \int_{\mathcal{Y}} q(\mathbf{y}) \cdot l\left(\mathbf{y}, \tilde{\mathbf{y}}\right) d\mathbf{y} && (38b)\\
&= \sum_{i,j} \underbrace{\int_{\mathbb{R}^2} q(\mathbf{y}_{ij}) \cdot \ell\left(\mathbf{y}_{ij}, \tilde{\mathbf{y}}_{ij}\right) d\mathbf{y}_{ij}}_{=: f_{ij}(\tilde{\mathbf{y}}_{ij})}. && (38c)
\end{aligned}$$

For fixed $\mathbf{y}_{ij} \in \mathbb{R}^2$, the function $q(\mathbf{y}_{ij}) \cdot \ell\left(\mathbf{y}_{ij}, \tilde{\mathbf{y}}_{ij}\right)$ is convex in $\tilde{\mathbf{y}}_{ij}$. Therefore, the objective $f(\tilde{\mathbf{y}})$ is convex in $\tilde{\mathbf{y}}$ and the Bayesian risk minimization has a unique solution given by

$$\mathbf{y}_{ij} = \arg\min_{\tilde{\mathbf{y}}_{ij}} f_{ij}(\tilde{\mathbf{y}}_{ij}). \tag{39}$$

It only remains to be shown that $\nabla_2\ f_{ij}(\tilde{\mathbf{y}}_{ij}) = 0$ holds for $\tilde{\mathbf{y}}_{ij} = \boldsymbol{\mu}_{ij}$. Setting $\tilde{\mathbf{y}}_{ij} = \boldsymbol{\mu}_{ij}$ we obtain

$$\int_{-\infty}^{\boldsymbol{\mu}_{ij}} q(\boldsymbol{\tau}) \cdot \nabla_2\, \ell\left(\boldsymbol{\tau}, \boldsymbol{\mu}_{ij}\right) d\boldsymbol{\tau} \overset{(\mathbf{z}_1 = \boldsymbol{\tau} - \boldsymbol{\mu}_{ij})}{=} \int_{-\infty}^{\mathbf{0}} q(\boldsymbol{\mu}_{ij} + \mathbf{z}_1) \cdot \nabla_2\, \ell\left(\boldsymbol{\mu}_{ij} + \mathbf{z}_1, \boldsymbol{\mu}_{ij}\right) d\mathbf{z}_1 \tag{40a}$$

$$\overset{q\ \text{sym.}}{=} \int_{-\infty}^{\mathbf{0}} q(\boldsymbol{\mu}_{ij} - \mathbf{z}_1) \cdot \nabla_2\, \ell\left(\boldsymbol{\mu}_{ij} + \mathbf{z}_1, \boldsymbol{\mu}_{ij}\right) d\mathbf{z}_1 \tag{40b}$$

$$\overset{(\mathbf{z}_2 = \boldsymbol{\mu}_{ij} - \mathbf{z}_1)}{=} \int_{\boldsymbol{\mu}_{ij}}^{\infty} q(\mathbf{z}_2) \cdot \nabla_2\, \ell\left(2\boldsymbol{\mu}_{ij} - \mathbf{z}_2, \boldsymbol{\mu}_{ij}\right) d\mathbf{z}_2 \tag{40c}$$

$$\overset{(37a)-(37c)}{=} -\int_{\boldsymbol{\mu}_{ij}}^{\infty} q(\mathbf{z}_2) \cdot \nabla_2\, \ell\left(\mathbf{z}_2, \boldsymbol{\mu}_{ij}\right) d\mathbf{z}_2 \tag{40d}$$

and finally

$$\nabla_2\, f_{ij}\left(\boldsymbol{\mu}_{ij}\right) = \int_{\mathbb{R}^2} q(\boldsymbol{\tau}) \cdot \nabla_2\, \ell\left(\boldsymbol{\tau}, \boldsymbol{\mu}_{ij}\right) d\boldsymbol{\tau} \tag{41a}$$

$$= \int_{-\infty}^{\boldsymbol{\mu}_{ij}} q(\boldsymbol{\tau}) \cdot \nabla_2\, \ell\left(\boldsymbol{\tau}, \boldsymbol{\mu}_{ij}\right) d\boldsymbol{\tau} + \int_{\boldsymbol{\mu}_{ij}}^{\infty} q(\boldsymbol{\tau}) \cdot \nabla_2\, \ell\left(\boldsymbol{\tau}, \boldsymbol{\mu}_{ij}\right) d\boldsymbol{\tau}$$

$$\overset{(40d)}{=} -\int_{\boldsymbol{\mu}_{ij}}^{\infty} q(\boldsymbol{\tau}) \cdot \nabla_2\, \ell\left(\boldsymbol{\tau}, \boldsymbol{\mu}_{ij}\right) d\boldsymbol{\tau} + \int_{\boldsymbol{\mu}_{ij}}^{\infty} q(\boldsymbol{\tau}) \cdot \nabla_2\, \ell\left(\boldsymbol{\tau}, \boldsymbol{\mu}_{ij}\right) d\boldsymbol{\tau} \tag{41b}$$

$$= 0. \tag{41c}$$

C. Implementation Details

In this section, we present our design choices following the best-practices of energy-based optical flow techniques, and give an analysis evaluating the influence of the specifics. Moreover, we give details of our post-processing approach using the fast bilateral solver [4].

C.1. ProbClassicA

In our ProbClassicA algorithm, we perform three steps of graduated non-convexity and apply coarse-to-fine estimation with 10 warping steps per layer. As in [43], we restrict the flow update to an absolute value of 1 and pre-process the images using a structure-texture decomposition. Spline-based cubic interpolation as well as an averaging of image gradients $\nabla_2 I_1$ and $\nabla_2 I_2$ are applied. During the inference, the variable sets $\{\boldsymbol{\mu}, \boldsymbol{\Sigma}, \mathbf{k}\}$ and $\left\{\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}}, \hat{\mathbf{k}}\right\}$ are updated in an alternating way. As an inner update step, we apply five iterations of the block-coordinate descent scheme on $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$ and $\mathbf{k}$. For the set $\left\{\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}}, \hat{\mathbf{k}}\right\}$, a number of three inner updates performs better.

C.2. ProbFlowFields

For ProbFlowFields, we follow [36] and pre-smooth images using a Gaussian kernel of size 9×9 with $\sigma = 1.1$. For warping, we apply bilinear interpolation and averaged image derivatives. Moreover, we perform five warping steps, each with five iterations of our block-coordinate descent scheme. We follow Revaud *et al*. and compute optical flow updates with 30 iterations of successive over relaxation, which performs noticeably faster than the solver used in [43].

C.3. Evaluation of design choices

Table 6 summarizes results of AEPE, AUC, and CC on the Middlebury and Sintel benchmarks using varying setups of ProbClassicA and ProbFlowFields. In a first step, we evaluate a setting for ProbClassicA in which parameters λ_{D}, λ_{S}, and λ_{N} are determined by having the Bayesian optimization [42] consider only the AEPE or only the AUC instead of the F_1-score

ProbClassicA Middlebury	AEPE	rel. chg.	AUC	rel. chg.	CC	rel. chg.
Baseline	0.296	–	0.466	–	0.374	–
Bayesian optim. w.r.t. AEPE only	0.290	**-0.02**	0.471	0.01	0.351	**0.06**
Bayesian optim. w.r.t. AUC only	0.312	**0.05**	0.436	**-0.06**	0.451	**-0.21**
$E_{\mathrm{N}} = E_{\mathrm{C}} = 0$	0.411	**0.39**	0.889	**0.91**	0.125	**0.67**
No structure-texture decomposition	0.290	**-0.02**	0.445	**-0.05**	0.361	**0.03**
ProbFlowFields Sintel validation	AEPE	rel. chg.	AUC	rel. chg.	CC	rel. chg.
Baseline	3.127	–	0.398	–	0.563	–
Bayesian optim. w.r.t. AEPE only	3.128	<0.01	0.475	**0.19**	0.407	**0.28**
Bayesian optim. w.r.t. AUC only	3.219	**0.03**	0.381	**-0.04**	0.644	**-0.14**
Spatially constant λ_{S}	3.127	0.00	0.400	<0.01	0.562	<0.01
$\theta_{ij}^r = 1$	3.125	>-0.01	0.396	>-0.01	0.548	**0.03**
No gradient averaging	3.135	<0.01	0.398	0.00	0.557	0.01
No Gaussian smoothing	3.135	<0.01	0.441	**0.11**	0.497	**0.12**
10 warping steps	3.123	>-0.01	0.421	**0.06**	0.538	**0.04**

Table 6. Analysis of several design choices for ProbClassicA on Middlebury and ProbFlowFields on the Sintel validation set. Bold entries denote strong deviations from the baseline.

proposed in Eq. (20). In both cases, we observe that the performance w.r.t. the evaluation metric that is not considered during the Bayesian optimization drops significantly. This highlights the importance of the F_1-score to balance the accuracy of flow and uncertainty estimates. Moreover, we show that the AEPE as well as the performance of the uncertainty measure is clearly inferior if no additional nonlocal term is applied ($E_{\mathrm{N}} = E_{\mathrm{C}} = 0$). When using ProbClassicA without structure-texture decomposition as pre-processing, we surprisingly obtain improved results for the AEPE (2%) as well as the AUC (5%). This is in contrast to energy minimization, where this pre-processing helps [43]. For fairness of comparison to the underlying energy minimization approach, we continue to use a structure-texture decomposition.

Considering ProbFlowFields, we observe the same behavior as for ProbClassicA when Bayesian optimization is carried out only with respect to one of the evaluation metrics. Note that the parameter setting obtained by a Bayesian optimization w.r.t. to the AEPE performs better than the baseline on the training set even though no improvement of the AEPE is visible on the validation set. The usage of a spatially constant trade-off parameter λ_{S}, turning off the normalization of the spatial derivatives ($\theta_{ij}^r = 1$, *c.f.* Eqs. (17) and (18)), and not averaging the image gradients, respectively, only lead to minor changes. When no Gaussian smoothing is applied for image pre-processing, a clear effect on the AUC as well as the CC can be observed whereas the AEPE is only slightly changed. Finally, the application of 10 warping steps only results in small improvements of the AEPE and even decreases the performance of the uncertainty measure. This justifies the usage of a reduced number of 5 steps to save computational time.

C.4. Post-processing using the fast bilateral solver

As described in Sec. 7.3, we apply the fast bilateral solver [4] on top of ProbFlowFields in order to illustrate the benefits of uncertainty predictions for a further improvement of the flow estimates. In doing so, we normalize the estimated uncertainties with a sigmoid function and invert the values to obtain the confidences required by the fast bilateral solver. A Bayesian optimization [42] is performed on our Sintel training set to obtain appropriate sigmoid parameters as well as a suitable trade-off parameter for the fast bilateral solver. See Fig. 5 for a screenshot of the private Sintel benchmark table showing results after post-processing (ProbFlowFields + BS). For the reported baseline, we process the estimates of ProbFlowFields assuming a uniform confidence of 0.5.

D. Additional Uncertainty Measures

In the following, we evaluate several additional uncertainty measures on the Middlebury as well as the Sintel benchmark. Haußecker and Spies [20] introduce three confidence measures based on the spatio-temporal structure tensor

$$\mathbf{S} = G(\tilde{\sigma}) * \left[(\nabla_3 I)(\nabla_3 I)^{\mathrm{T}}\right] \qquad \text{with} \qquad \nabla_3 I = (I_x, I_y, I_t)^{\mathrm{T}}, \tag{42}$$

where I_x and I_y denote the spatial image derivatives computed with central differences and I_t is the temporal difference between I_1 and I_2. Following [30], we smooth the derivatives with a Gaussian filter $G(\tilde{\sigma})$ of size 7×7 and a standard

	EPE all	EPE matched	EPE unmatched	d0-10	d10-60	d60-140	s0-10	s10-40	s40+	
GroundTruth [1]	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	Visualize Results
DCFlow [2]	5.119	2.283	28.228	4.665	2.108	1.440	1.052	3.434	29.351	Visualize Results
FlowFieldsCNN [3]	5.363	2.303	30.313	4.718	2.020	1.399	1.032	3.065	32.422	Visualize Results
MR-Flow [4]	5.376	2.818	26.235	5.109	2.395	1.755	0.908	3.443	32.221	Visualize Results
FTFlow [5]	5.390	2.268	30.841	4.513	1.964	1.366	1.046	3.322	31.936	Visualize Results
S2F-IF [6]	5.417	2.549	28.795	4.745	2.198	1.712	1.157	3.468	31.262	Visualize Results
InterpoNet_ff [7]	5.535	2.372	31.296	4.720	2.018	1.532	1.064	3.496	32.633	Visualize Results
RegionalFF [8]	5.562	2.595	29.741	4.921	2.393	1.639	1.122	3.477	32.625	Visualize Results
PGM-C [9]	5.591	2.672	29.389	4.975	2.340	1.791	1.057	3.421	33.339	Visualize Results
RicFlow [10]	5.620	2.765	28.907	5.146	2.366	1.679	1.088	3.364	33.573	Visualize Results
InterpoNet_cpm [11]	5.627	2.594	30.344	4.975	2.213	1.640	1.042	3.575	33.321	Visualize Results
ProbFlowFields+BS [12]	5.628	2.543	30.773	4.680	2.169	1.683	1.086	3.538	33.210	Visualize Results
CPM_AUG [13]	5.645	2.737	29.362	4.707	2.150	1.918	1.087	3.306	33.925	Visualize Results
ProbFlowFields [14]	5.696	2.545	31.371	4.696	2.150	1.686	1.146	3.658	33.188	Visualize Results

Figure 5. Screenshot of private Sintel table (final) showing results for ProbFlowFields and ProbFlowFields + BS (status as of July 2017).

Uncertainty measure	AUC	rel. chg.	CC	rel. chg.
Ct [20]	1.058	1.27	-0.106	1.28
Cs [20]	1.014	1.18	-0.057	1.15
Cc [20]	0.967	1.08	-0.022	1.06
Ev3 [27]	0.989	1.12	0.058	0.84
Noise	0.512	0.10	0.286	0.24
ProbClassicA *(ours)*	**0.466**	**0.00**	**0.374**	**0.00**
Oracle	*0.255*	–	*1.000*	–

Table 7. Area under curve (AUC), Spearman's rank correlation coefficient (CC), and relative change (rel. chg.) in comparison to the our uncertainty measure on the Middlebury dataset.

Uncertainty measure	AUC	rel. chg.	CC	rel. chg.
Ct [20]	1.130	1.84	-0.128	1.23
Cs [20]	1.154	1.90	-0.149	1.26
Cc [20]	0.915	1.30	0.129	0.77
Ev3 [27]	1.024	1.57	-0.030	1.05
Noise	0.512	0.29	0.382	0.32
ProbFlowFields *(ours)*	**0.398**	**0.00**	**0.563**	**0.00**
Oracle	*0.182*	–	*1.000*	–

Table 8. Area under curve (AUC), Spearman's rank correlation coefficient (CC), and relative change (rel. chg.) in comparison to our uncertainty measure on a Sintel benchmark validation set.

deviation $\tilde{\sigma} = 2$. In [20], eigenvalues λ_1, λ_2, and λ_3 of $\mathbf{S}$ are computed such that $\lambda_1 \geq \lambda_2 \geq \lambda_3$. Uncertainty measures are then obtained as

$$\Psi_{\text{Ct}} = -\left(\frac{\lambda_1 - \lambda_3}{\lambda_1 + \lambda_3}\right)^2, \quad \Psi_{\text{Cs}} = -\left(\frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2}\right)^2, \quad \text{and} \quad \Psi_{\text{Cc}} = \Psi_{\text{Ct}} - \Psi_{\text{Cs}}. \tag{43}$$

Moreover, we evaluate a baseline uncertainty measure as used in [27] defined as $\Psi_{\text{Ev3}} = -\lambda_3$.

Finally, we compare to a sampling-based measure similar to the idea of Kybic and Nieuwenhuis [27]. That is, we estimate the uncertainty as the variance of the optical flow estimates resulting from small, random perturbations of the input data. Specifically, we apply zero-mean Gaussian noise on the input images and determine appropriate values for the variance of the noise on the training set. The uncertainty measure is then obtained as $\Psi_{\text{Noise}} = \sqrt{\sigma_u^2 + \sigma_v^2}$ with σ_u and σ_v denoting the standard derivation of the horizontal and vertical flow estimates per pixel.

As can be seen in Tables 7 and 8, all measures based on the structure tensor perform considerably worse than our proposed uncertainty measure. Ψ_{Cc} and Ψ_{Ev3} lead to more meaningful uncertainties than the two remaining approaches on both datasets, but perform similar to the simple gradient-based measure [3]. The noise uncertainty – especially on the Middlebury dataset – performs comparably to Ψ_{Energy} and Ψ_{Learned}. However, our ProbFlow approach clearly leads to superior results.

E. ProbFlowFields on Middlebury

For completeness, we report the results of ProbFlowFields on Middlebury. To reproduce the Middlebury results shown in [1] we applied the default settings of the EpicFlow interpolation. Moreover, we use GSM potentials trained on the Sintel

Method	training		test	
	AEPE	rel. chg.	AEPE	rel. chg.
Initialization	0.307	0.38	–	–
FlowFields [1]	0.240	0.08	0.331†	0.10
FieldsFields⋆	0.230	0.04	–	–
ProbFlowFields *(ours)*	**0.222**	**0.00**	**0.301**	**0.00**

Table 9. Average end-point error (AEPE) and relative change (rel. chg.) in comparison to the ProbFlowFields method on the Middlebury benchmark. †Please note that we did not re-evaluate FlowFields, but show the publicly available results.

Uncertainty measure	AUC	rel. chg.	CC	rel. chg.
Gradient [3]	1.244	1.72	-0.077	1.21
Laplace	0.539	0.18	0.297	0.20
Energy [8]	0.563	0.23	0.253	0.32
Learned [30]	0.473	0.04	**0.374**	**>-0.01**
ProbFlowFields *(ours)*	**0.457**	**0.00**	0.371	0.00
Oracle	*0.247*	–	*1.000*	–

Table 10. Area under curve (AUC), Spearman's rank correlation coefficient (CC), and relative change (rel. chg.) in comparison to the energy uncertainty measure on the Middlebury training set.

Average endpoint error	avg. rank	Army (Hidden texture) GT im0 im1			Mequon (Hidden texture) GT im0 im1			Schefflera (Hidden texture) GT im0 im1			Wooden (Hidden texture) GT im0 im1			Grove (Synthetic) GT im0 im1			Urban (Synthetic) GT im0 im1			Yosemite (Synthetic) GT im0 im1			Teddy (Stereo) GT im0 im1		
		all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext
NNF-Local [87]	3.4	**0.07** 1	0.20 2	**0.05** 1	0.15 2	0.51 4	0.12 5	0.18 2	0.37 2	0.14 2	0.10 2	0.49 5	0.06 2	**0.41** 1	**0.61** 1	0.21 2	0.23 2	0.66 3	0.19 2	0.10 6	0.12 12	0.17 13	0.34 2	0.80 6	0.23 2
PMMST [114]	9.3	0.09 32	0.21 5	0.07 16	0.18 11	0.51 4	0.16 28	0.21 9	0.42 8	0.17 16	0.10 2	**0.33** 1	0.08 13	0.51 5	0.74 4	0.28 7	0.24 3	0.65 2	0.20 5	0.11 18	0.12 12	0.17 13	0.37 4	0.74 2	0.35 4
OFLAF [77]	9.8	0.08 9	0.21 5	0.06 6	0.16 6	0.53 8	0.12 5	0.19 3	0.37 2	0.14 2	0.14 9	0.77 29	0.07 5	0.51 5	0.78 7	0.25 4	0.31 11	0.76 4	0.25 16	0.11 18	0.12 12	0.21 40	0.42 10	0.78 4	0.63 18
MDP-Flow2 [68]	10.6	0.08 9	0.21 5	0.07 16	0.15 2	**0.48** 1	**0.11** 1	0.20 5	0.40 5	0.14 2	0.15 22	0.80 37	0.08 13	0.63 19	0.93 19	0.43 20	0.26 5	0.76 4	0.23 9	0.11 18	0.12 12	0.17 13	0.38 6	0.79 5	0.44 6
NN-field [71]	11.8	0.08 9	0.22 17	**0.05** 1	0.17 8	0.55 10	0.13 11	0.19 3	0.39 4	0.15 7	**0.09** 1	0.48 4	**0.05** 1	**0.41** 1	**0.61** 1	**0.20** 1	0.52 60	**0.64** 1	0.26 19	0.13 42	0.13 36	0.20 34	0.35 3	0.83 8	**0.21** 1
ComponentFusion [96]	13.9	**0.07** 1	0.21 5	**0.05** 1	0.16 6	0.55 10	0.12 5	0.20 5	0.44 9	0.15 7	0.11 4	0.65 9	0.06 2	0.71 35	1.07 40	0.53 37	0.32 15	1.06 27	0.28 22	0.11 18	0.13 36	0.15 8	0.41 9	0.88 13	0.54 9
TC/T-Flow [76]	19.5	**0.07** 1	0.21 5	**0.05** 1	0.19 18	0.68 34	0.12 5	0.28 26	0.66 31	0.14 2	0.14 9	0.86 47	0.07 5	0.67 29	0.98 29	0.49 31	**0.22** 1	0.82 9	0.19 2	0.11 18	0.11 2	0.30 85	0.50 27	1.02 30	0.64 20
WLIF-Flow [93]	19.8	0.08 9	0.21 5	0.06 6	0.18 11	0.55 10	0.15 23	0.25 18	0.56 20	0.17 16	0.14 9	0.68 10	0.08 13	0.61 16	0.91 17	0.41 18	0.43 35	0.96 15	0.29 28	0.13 42	0.12 12	0.21 40	0.51 32	1.03 33	0.72 38
NNF-EAC [103]	21.3	0.09 32	0.22 17	0.07 16	0.17 8	0.53 6	0.13 11	0.23 11	0.49 12	0.15 7	0.16 36	0.80 37	0.09 28	0.60 13	0.89 13	0.40 16	0.38 25	0.78 6	0.28 22	0.12 31	0.12 12	0.18 26	0.57 45	1.24 49	0.69 32
Layers++ [37]	21.9	0.08 9	0.21 5	0.07 16	0.19 18	0.56 13	0.17 35	0.20 5	0.40 5	0.18 27	0.13 8	0.58 7	0.07 5	0.48 3	0.70 3	0.33 9	0.47 47	1.01 19	0.33 47	0.15 65	0.14 58	0.24 53	0.46 17	0.88 13	0.72 38
LME [70]	22.9	0.08 9	0.22 17	0.06 6	0.15 2	0.49 2	**0.11** 1	0.30 35	0.64 26	0.31 89	0.15 22	0.78 33	0.09 28	0.66 25	0.96 24	0.53 37	0.33 16	1.18 44	0.28 22	0.12 31	0.12 12	0.18 26	0.44 12	0.91 16	0.61 14
IROF++ [58]	23.0	0.08 9	0.23 24	0.07 16	0.21 32	0.68 34	0.17 35	0.28 26	0.63 24	0.19 39	0.15 22	0.73 22	0.09 28	0.60 13	0.89 13	0.42 19	0.43 35	1.08 30	0.31 37	0.10 6	0.12 12	0.12 4	0.47 19	0.98 23	0.68 31
nLayers [57]	23.8	**0.07** 1	**0.19** 1	0.06 6	0.22 40	0.59 16	0.19 57	0.25 18	0.54 16	0.20 48	0.15 22	0.84 43	0.08 13	0.53 7	0.78 7	0.34 11	0.44 39	0.84 10	0.30 33	0.13 42	0.13 36	0.20 34	0.47 19	0.97 22	0.67 29
HAST [109]	25.1	**0.07** 1	0.20 2	**0.05** 1	0.18 11	0.54 8	0.13 11	**0.17** 1	**0.32** 1	**0.12** 1	0.15 22	0.90 58	0.06 2	0.49 4	0.74 4	0.22 3	0.58 70	1.09 31	0.44 70	0.19 96	0.17 87	0.47 114	**0.32** 1	**0.64** 1	0.33 3
PH-Flow [101]	25.8	0.08 9	0.24 31	0.07 16	0.21 32	0.68 34	0.17 35	0.23 11	0.49 12	0.19 39	0.16 36	0.83 41	0.09 28	0.56 9	0.83 9	0.38 13	0.30 9	0.81 7	0.24 14	0.15 65	0.13 36	0.30 85	0.43 11	0.85 9	0.66 27
FC-2Layers-FF [74]	25.8	0.08 9	0.21 5	0.07 16	0.21 32	0.70 42	0.17 35	0.20 5	0.40 5	0.18 27	0.15 22	0.76 28	0.08 13	0.53 7	0.77 6	0.37 12	0.49 53	1.02 20	0.33 47	0.16 76	0.13 36	0.29 80	0.44 12	0.87 12	0.64 20
Correlation Flow [75]	26.5	0.09 32	0.23 24	0.07 16	0.17 8	0.58 15	**0.11** 1	0.43 65	0.99 67	0.15 7	0.11 4	0.47 3	0.08 13	0.75 41	1.08 41	0.56 42	0.41 31	0.92 13	0.30 33	0.14 52	0.13 36	0.27 69	0.40 8	0.85 9	0.42 5
AGIF+OF [85]	28.0	0.08 9	0.22 17	0.07 16	0.23 55	0.73 47	0.18 47	0.28 26	0.66 31	0.18 27	0.14 9	0.70 13	0.08 13	0.57 10	0.85 10	0.38 13	0.47 47	0.97 16	0.31 37	0.13 42	0.13 36	0.22 46	0.51 32	0.99 26	0.74 47
RNLOD-Flow [121]	28.0	**0.07** 1	0.20 2	0.06 6	0.19 18	0.68 34	0.13 11	0.33 49	0.79 50	0.17 16	0.14 9	0.73 22	0.07 5	0.69 33	1.03 33	0.48 27	0.37 24	0.99 17	0.29 28	0.16 76	0.16 79	0.29 80	0.45 14	0.88 13	0.65 25
ProbFlowFields [128]	28.8	0.10 47	0.31 73	0.08 44	0.19 18	0.63 23	0.17 35	0.27 22	0.63 24	0.22 59	0.11 4	0.49 5	0.07 5	0.82 51	1.22 56	0.59 45	0.25 4	1.05 26	0.21 6	0.09 4	0.12 12	0.17 13	0.58 47	1.33 51	0.62 16
⋮																										
DeepFlow [86]	65.9	0.12 79	0.31 73	0.11 91	0.28 81	0.82 72	0.22 82	0.44 71	1.00 68	0.33 90	0.26 86	1.34 96	0.15 81	0.81 49	1.21 53	0.58 44	0.38 25	1.55 78	0.25 16	0.11 18	0.11 2	0.24 53	0.93 91	1.82 93	1.12 89
ProbClassicA [127]	66.5	0.10 47	0.28 54	0.08 44	0.22 40	0.78 61	0.16 28	0.57 88	1.17 84	0.22 59	0.21 72	1.24 86	0.11 56	0.86 58	1.30 72	0.63 52	0.55 65	1.74 93	0.37 66	0.16 76	0.14 58	0.34 99	0.81 82	1.74 85	0.93 71
TriangleFlow [30]	66.7	0.11 59	0.29 62	0.09 65	0.26 73	0.95 88	0.17 35	0.47 74	1.07 72	0.18 27	0.16 36	0.87 51	0.09 28	1.07 97	1.47 104	1.10 98	0.87 86	1.39 65	0.57 87	0.15 65	0.19 109	0.23 52	0.63 55	1.33 51	0.84 61

Figure 6. Screenshot of private Middlebury table showing results for ProbFlowFields and ProbClassicA (status as of July 2017).

dataset for our ProbFlowFields approach. The results evaluating the AEPE on the Middlebury benchmark can be found in Table 9. We outperform the original FlowFields approach on training and test and obtain improved results in comparison to FlowFields⋆. Please note that the Middlebury benchmark policy allows no more than one entry per method in the public table. Therefore, we decided to show the results of ProbFlowFields on the Middlebury website whereas the results of ProbClassicA from Table 1 of the main paper are only visible in a private table, see Fig. 6 for a screenshot.

Table 10 shows an evaluation of different uncertainty measures. In contrast to our remaining experiments, the Laplace and learned uncertainty measures both outperform the energy-based approach. Our uncertainty measure is slightly outperformed by Ψ_{Learned} w.r.t. the CC metric. However, ProbFlowFields shows clearly superior results considering the AUC.

References

[50] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. *ECCV*, 2004. 3

A.2 STOCHASTIC VARIATIONAL INFERENCE WITH GRADIENT LINEARIZATION

Tobias Plötz*, Anne S. Wannenwetsch*, and Stefan Roth
2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition

Abstract

Variational inference has experienced a recent surge in popularity owing to stochastic approaches, which have yielded practical tools for a wide range of model classes. A key benefit is that stochastic variational inference obviates the tedious process of deriving analytical expressions for closed-form variable updates. Instead, one simply needs to derive the gradient of the log-posterior, which is often much easier. Yet for certain model classes, the log-posterior itself is difficult to optimize using standard gradient techniques. One such example are random field models, where optimization based on gradient linearization has proven popular, since it speeds up convergence significantly and can avoid poor local optima. In this paper we propose stochastic variational inference with gradient linearization (SVIGL). It is similarly convenient as standard stochastic variational inference – all that is required is a local linearization of the energy gradient. Its benefit over stochastic variational inference with conventional gradient methods is a clear improvement in convergence speed, while yielding comparable or even better variational approximations in terms of KL divergence. We demonstrate the benefits of SVIGL in three applications: Optical flow estimation, Poisson-Gaussian denoising, and 3D surface reconstruction.

*Authors contributed equally

Stochastic Variational Inference with Gradient Linearization

Tobias Plötz* Anne S. Wannenwetsch* Stefan Roth
Department of Computer Science, TU Darmstadt

Abstract

Variational inference has experienced a recent surge in popularity owing to stochastic approaches, which have yielded practical tools for a wide range of model classes. A key benefit is that stochastic variational inference obviates the tedious process of deriving analytical expressions for closed-form variable updates. Instead, one simply needs to derive the gradient of the log-posterior, which is often much easier. Yet for certain model classes, the log-posterior itself is difficult to optimize using standard gradient techniques. One such example are random field models, where optimization based on gradient linearization has proven popular, since it speeds up convergence significantly and can avoid poor local optima. In this paper we propose stochastic variational inference with gradient linearization (SVIGL). It is similarly convenient as standard stochastic variational inference – all that is required is a local linearization of the energy gradient. Its benefit over stochastic variational inference with conventional gradient methods is a clear improvement in convergence speed, while yielding comparable or even better variational approximations in terms of KL divergence. We demonstrate the benefits of SVIGL in three applications: Optical flow estimation, Poisson-Gaussian denoising, and 3D surface reconstruction.

1. Introduction

Computer vision algorithms increasingly become building blocks in ever more complex systems, prompting for ways of assessing the reliability of each component. Probability distributions allow for a natural way of quantifying predictive uncertainty. Here, variational inference (VI, see [43] for an extensive introduction) is one of the main computational workhorses. Stochastic approaches to variational inference [17, 21, 31, 33] have recently rejuvenated the interest in this family of approximate inference methods. Part of their popularity stems from their making variational inference applicable to large-scale models, thus enabling practical systems [40]. Another benefit, which should not be underestimated, is that they allow to apply variational inference in a black-box fashion [31, 40], since it is no longer required to carry out tedious and moreover model-specific derivations of the update equations. This allows practitioners to apply variational inference to new model classes very quickly. The only required model specifics are gradients of the log-posterior w.r.t. its unknowns, which are typically much easier to derive than variational update equations. Moreover, automatic differentiation [4] can be used to further reduce manual intervention.

While this makes stochastic variational inference techniques attractive from the user's perspective, there are some caveats. In this paper we specifically focus on the limitations of gradient-based optimization techniques in the context of certain highly nonlinear model classes. One such category are random field models [6], which often arise in dense prediction tasks in vision. Let us take optical flow [8, 32] as an illustrative example. The data model is highly multimodal and the prior frequently relies on non-convex potentials, which complicate inference [5]. Gradient-based optimization is severely challenged by the multi-modal energy function. Hence, approaches based on energy minimization [8, 32, 42] often rely on a optimization technique called gradient linearization [29], which proceeds by iteratively linearizing the gradient at the current estimate and then solving the resulting system of linear equations to obtain the next iterate. Our starting point is the following question: If gradient linearization is beneficial for maximum a-posteriori (MAP) estimation in certain model classes, would not stochastic variational inference benefit similarly?

In this paper, we derive *stochastic variational inference with gradient linearization (SVIGL)* – a general optimization algorithm for stochastic variational inference that only hinges on the availability of linearized gradients of the underlying energy function. In each iteration, SVIGL linearizes a stochastic gradient estimate of the Kullback-Leibler (KL) divergence and solves for the root of the linearization. We show that each step of this procedure optimizes a sound objective. Furthermore, we make interesting experimental findings for challenging models from optical flow estimation and Poisson-Gaussian denoising. First, we observe that SVIGL leads to faster convergence of the variational objective function than gradient-based stochas-

*Authors contributed equally

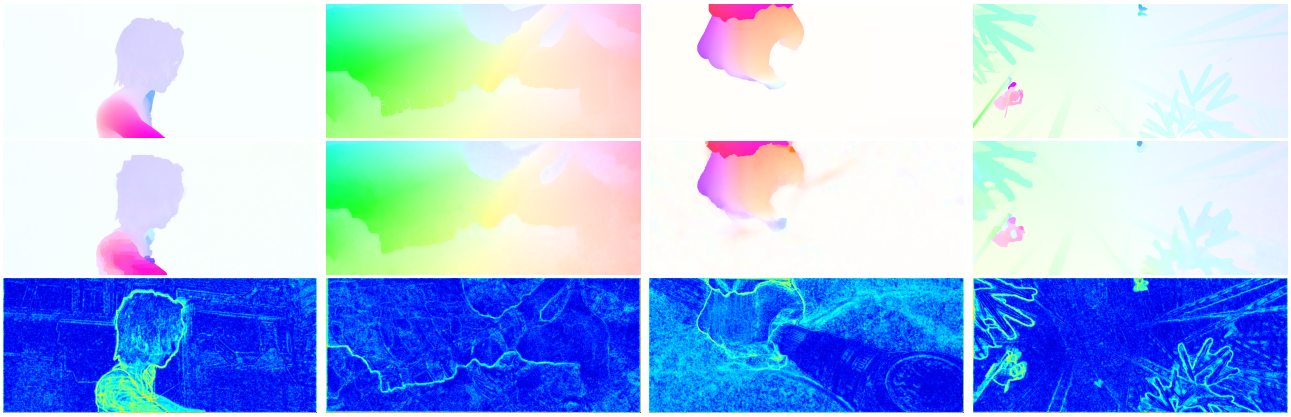


Figure 1. *Example application of SVIGL to optical flow estimation:* Ground truth (top), flow predictions (middle), and uncertainty estimates (bottom) on Sintel final [9]. Note that the uncertainties agree well with the flow errors.

tic variational inference (SVI) with the strong optimizers Adam [20] and stochastic gradient descent (SGD). Second, we show that SVIGL is more robust w.r.t. its optimization parameters than standard gradient-based approaches. Finally, SVIGL enables re-purposing existing well-proven energy minimization schemes and implementations to obtain uncertainty estimates while maintaining, or even improving, application performance. Figure 1 shows exemplary flow fields and uncertainty predictions of SVIGL. As expected intuitively, high uncertainty values coincide with errors in the estimated flow field, *e.g.* near motion discontinuities. Finally, we show that SVIGL benefits problems beyond dense prediction by employing it for 3D surface reconstruction.

2. Related Work

Variational inference. For Bayesian networks, VI w.r.t. to the exclusive Kullback-Leibler divergence KL $(q \,||\, p)$ has usually been restricted to certain model classes. The parametric form of the approximating distribution q is chosen such that update equations for the variational parameters of q are analytically tractable. Here, conjugate-exponential models [47] are very common as they often arise in the context of topic modeling, *e.g.* in the LDA model [7, 38].

In other application areas, *e.g.* in computer vision, Markov random field (MRF) models are more common. Traditionally, VI has only been applied to specific model classes with closed-form updates, *e.g.* [11, 23, 24, 27, 36]. Miskin and MacKay [27] pioneered the use of VI for Bayesian blind deconvolution, but made the restrictive assumption that the prior is fully factorized. Levin *et al.* [23] use a mixture of Gaussian prior on the image derivatives. However, this more powerful prior comes at the cost of additionally maintaining a variational approximation of all mixture components. Krähenbühl and Koltun [22] consider fully-connected conditional random fields (CRF) with Gaussian edge potentials. In this special case mean-field inference can be done efficiently through filtering. Schelten and Roth [36] apply VI to high-order random fields.

In all of the previously mentioned works the variational inference algorithm is closely tied to the probabilistic model at hand and oftentimes requires tedious derivations of analytical update equations. In this paper, we aim to make VI more practical as the only interaction with the probabilistic model is through the linearized gradient of its log probability density function, thus allowing for easy variational inference for a rich class of graphical models.

Stochastic variational optimization. Recently, it was shown that the KL divergence is amenable to stochastic optimization if the approximating distribution q can be reparameterized in terms of a base distribution that does not or only weakly depend on the parameters of q [21, 33, 35]. While SVI was originally proposed for learning deep latent variable models, such as variational auto-encoders, it is also applicable more generally to graphical models. Reparameterization allows for deriving efficient stochastic estimators of the gradient of the KL divergence [21, 28]. Only the unnormalized log-density and its gradient w.r.t. the hidden variables are required, thus enabling black-box VI [19, 31, 40]. Note that by stochastic variational inference we do not just refer to the method of Hoffman *et al.* [17], which, in contrast, requires the true posterior to be from the conjugate-exponential family. Instead, we use the term more generally to describe VI using stochastic optimization.

Having access to a gradient estimator, stochastic algorithms [34] are employed to do the actual optimization. Nowadays, one of the default choices is Adam [20], but other approaches are in use as well, *e.g.* RMSprop [39], AdaGrad [13], or L-BFGS-SGVI [14]. These algorithms each implement a gradient descent method that is tuned with the recent history of gradient evaluations. In contrast, we assume that we observe a linearization of the gradient and use the information contained therein to modify the direction of

the parameter updates. This can be seen as a gradient descent with a special preconditioner [29], see supplemental.

Applications of uncertainties. Aside from being a popular inference tool in many areas of computer vision, *e.g.* [22, 23], VI yields an assessment of the uncertainty, which can be exploited to post-process point estimates, *e.g.* with the fast bilateral solver [3]. When used as input for higher-level tasks, optical flow uncertainties allow to discard unreliable estimates and avoid error propagation [45], *e.g.* in image segmentation [30] or tracking [46]. Uncertainties in image restoration can be beneficial in video restoration, where estimates are fused over several frames [12].

3. Preliminaries

Variational inference [43] generally aims to approximate an intractable distribution p with a tractable distribution q. Since our applications are based on CRFs, we will specifically look at finding approximations to a posterior distribution $p(\mathbf{x} \mid \mathbf{y})$. Note, however, that our approach can be applied to marginal and joint distributions as well. We assume that p is a density function over continuous variables, and can be expressed as a Gibbs distribution with its energy function $E(\mathbf{x}, \mathbf{y})$ and partition function $Z(\mathbf{y})$ as

$$p(\mathbf{x} \mid \mathbf{y}) = \frac{1}{Z(\mathbf{y})} \exp\Big\{ - E(\mathbf{x}, \mathbf{y}) \Big\}. \tag{1}$$

To ease notation, we assume the temperature parameter to be subsumed into $E(\mathbf{x}, \mathbf{y})$, which we furthermore assume to be differentiable. The approximating distribution q is chosen to be a member of some parameterized family of distributions with parameter $\boldsymbol{\theta}$, usually from the exponential family [43]. To determine q, variational inference then aims to find variational parameters $\hat{\boldsymbol{\theta}}$ that minimize the exclusive Kullback-Leibler divergence $\mathrm{KL}\left(q \,||\, p\right)$, *i.e.*

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathrm{KL}\left(q \,||\, p\right) \tag{2a}$$

$$= \arg\min_{\boldsymbol{\theta}} - \mathbb{E}_{q(\mathbf{x};\boldsymbol{\theta})}[\log p(\mathbf{x} \mid \mathbf{y})] + \mathbb{E}_{q(\mathbf{x};\boldsymbol{\theta})}[\log q(\mathbf{x};\boldsymbol{\theta})] \tag{2b}$$

$$= \arg\min_{\boldsymbol{\theta}} - \mathbb{E}_{q(\mathbf{x};\boldsymbol{\theta})}[\log p(\mathbf{x} \mid \mathbf{y})] - H(q), \tag{2c}$$

where $H(q) = H\big(q(\mathbf{x};\boldsymbol{\theta})\big)$ denotes the entropy of q.

Gradient linearization. We now take a step back and first look at MAP estimation for the energy $E(\mathbf{x}, \mathbf{y})$ in Eq. (1), *i.e.* the problem of finding

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} \log p(\mathbf{x} \mid \mathbf{y}) = \arg\min_{\mathbf{x}} E(\mathbf{x}, \mathbf{y}). \tag{3}$$

Assuming that E is differentiable, we could now apply a standard gradient method, but this may lead to slow convergence. On the other hand, second-order methods may be difficult to apply as the Hessian can be tedious to obtain and/or too dense. For many large-scale prediction problems in computer vision, *e.g.* estimating optical flow [8, 32], denoising [41], or deblurring [42], this has been addressed through iterative *gradient linearization* (GL). In this procedure, given a current estimate $\mathbf{x}^{(t)}$, the gradient of the energy function E w.r.t. $\mathbf{x}$ is linearized around $\mathbf{x}^{(t)}$ as

$$\nabla_{\mathbf{x}} E(\mathbf{x}, \mathbf{y}) \approx \bar{\nabla}_{\mathbf{x}} E\big(\mathbf{x}; \mathbf{x}^{(t)}\big) = \mathbf{A}_{\mathbf{x}}\big(\mathbf{x}^{(t)}\big)\mathbf{x} + \mathbf{b}_{\mathbf{x}}\big(\mathbf{x}^{(t)}\big). \tag{4}$$

For notational brevity, we omit $\mathbf{y}$ here and in the following. Note that the linearized gradient $\bar{\nabla}_{\mathbf{x}} E\big(\mathbf{x}; \mathbf{x}^{(t)}\big)$ is exact at $\mathbf{x} = \mathbf{x}^{(t)}$. To obtain the next iterate $\mathbf{x}^{(t+1)}$, we set $\bar{\nabla}_{\mathbf{x}} E$ to zero and solve the resulting linear system of equations

$$\mathbf{x}^{(t+1)} = -\mathbf{A}_{\mathbf{x}}^{-1}\big(\mathbf{x}^{(t)}\big)\mathbf{b}_{\mathbf{x}}\big(\mathbf{x}^{(t)}\big) \tag{5}$$

using an exact or approximate standard solver. Like in any iterative optimization, an initial guess $\mathbf{x}^{(0)}$ is required.

Iterative GL is also known by various other names. Nikolova and Chan [29] showed it to be equivalent to multiplicative half-quadratic minimization [16] for Gaussian likelihoods. Moreover, it is closely related to iteratively reweighted least squares through their equivalence to half-quadratic approaches [18]. Finally, GL can be seen as preconditioned gradient descent using $\mathbf{A}_{\mathbf{x}}^{-1}$ as preconditioner [29], *c.f.* supplemental. In comparison to Newton's method no second-order derivatives are required – a benefit that is shared with other quasi-Newton methods, such as the popular L-BFGS [10]. However, every regular gradient step couples variables only within a local spatial neighborhood. In contrast, one iteration of GL (Eq. 5) causes a joint update of all variables leading to faster convergence in highly non-linear objectives (see Fig. 2 for an example).

4. Stochastic Variational Inference with Gradient Linearization (SVIGL)

We now aim to leverage the advantages of GL in the context of stochastic variational inference. To that end, we assume access to a linearized gradient, given by $\mathbf{A}_{\mathbf{x}}$ and $\mathbf{b}_{\mathbf{x}}$ in Eq. (4). By applying the re-parameterization trick [21, 33], we can rewrite the KL divergence of Eq. (2) as

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} - \mathbb{E}_{\mathbf{z}\sim\mathcal{G}}\Big[\log p\big(\mathbf{x}(\mathbf{z}) \mid \mathbf{y}\big)\Big] - H(q), \tag{6}$$

where $\mathbf{x}(\mathbf{z}) \equiv \mathbf{x}(\mathbf{z};\boldsymbol{\theta})$, and $\mathbf{z}$ is distributed following a base distribution $\mathcal{G}$ independent of $\boldsymbol{\theta}$. In the following, we approximate the full expectation over $\mathbf{z}$ with a finite set of samples $\mathcal{Z} = \{\mathbf{z}_i\}$. Using the approximation to the true gradient given by $\mathbf{A}_{\mathbf{x}}$ and $\mathbf{b}_{\mathbf{x}}$, we can then easily derive a stochastic approximation of the gradient of the KL diver-

gence in Eq. (6) with respect to the parameters $\boldsymbol{\theta}$:

$$\begin{aligned}
&\nabla_{\boldsymbol{\theta}} \operatorname{KL}(q \,||\, p) \\
&\overset{(6)}{=} -\mathbb{E}_{\mathbf{z}\sim\mathcal{G}}\Big[\nabla_{\mathbf{x}} \log p\big(\mathbf{x}(\mathbf{z}) \mid \mathbf{y}\big) \cdot \nabla_{\boldsymbol{\theta}}\, \mathbf{x}(\mathbf{z})\Big] - \nabla_{\boldsymbol{\theta}}\, H(q) && \text{(7a)} \\
&\approx -\frac{1}{|\mathcal{Z}|}\sum_{\mathbf{z}_i\in\mathcal{Z}} \nabla_{\mathbf{x}} \log p\big(\mathbf{x}(\mathbf{z}_i) \mid \mathbf{y}\big) \cdot \nabla_{\boldsymbol{\theta}}\, \mathbf{x}(\mathbf{z}_i) - \nabla_{\boldsymbol{\theta}}\, H(q) && \text{(7b)} \\
&\overset{(4)}{\approx} \frac{1}{|\mathcal{Z}|}\sum_{\mathbf{z}_i\in\mathcal{Z}} \Big(\mathbf{A}_{\mathbf{x}}\big(\mathbf{x}(\mathbf{z}_i)\big)\mathbf{x}(\mathbf{z}_i) + \mathbf{b}_{\mathbf{x}}\big(\mathbf{x}(\mathbf{z}_i)\big)\Big) \cdot \nabla_{\boldsymbol{\theta}}\, \mathbf{x}(\mathbf{z}_i) \\
&\quad - \nabla_{\boldsymbol{\theta}}\, H(q) && \text{(7c)} \\
&\equiv \bar{\nabla}_{\boldsymbol{\theta}} \operatorname{KL}(q \,||\, p)\,. && \text{(7d)}
\end{aligned}$$

Gaussian mean field inference. To illustrate the use of this approximation, we now apply the common naive mean-field framework [11, 21, 23] and assume that the variational distribution q factorizes along all elements of $\mathbf{x} = (x_l)_l$ for $l = 1, \ldots, L$. Moreover, q is modeled as an uncorrelated Gaussian distribution with $\boldsymbol{\theta} = \{\boldsymbol{\mu}, \boldsymbol{\sigma}\}$:

$$q(\mathbf{x}) = \prod_{l=1}^{L} \mathcal{N}(x_l \mid \mu_l, \sigma_l^2). \tag{8}$$

Following [21], $\mathbf{z}$ is thus chosen to be standard normally distributed, *i.e.* $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$, and we set $\mathbf{x}(\mathbf{z}) = \mathbf{z} \cdot \boldsymbol{\sigma} + \boldsymbol{\mu}$ with element-wise operations.

For the case of a fully-factorized Gaussian q, it is now possible to express $\bar{\nabla}_{\boldsymbol{\theta}} \operatorname{KL}(q \,||\, p)$ again in the form of a linearized gradient. To do this, we consider the individual parameter gradients w.r.t. $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$. For the gradient with respect to $\boldsymbol{\mu}$, we exploit that the entropy of a Gaussian distribution does not depend on its mean. Hence, we arrive at

$$\begin{aligned}
&\bar{\nabla}_{\boldsymbol{\mu}} \operatorname{KL}(q \,||\, p) \\
&= \frac{1}{|\mathcal{Z}|}\sum_{\mathbf{z}_i\in\mathcal{Z}} \Big(\mathbf{A}_{\mathbf{x}}\big(\mathbf{x}(\mathbf{z}_i)\big)\mathbf{x}(\mathbf{z}_i) + \mathbf{b}_{\mathbf{x}}\big(\mathbf{x}(\mathbf{z}_i)\big)\Big) \cdot \nabla_{\boldsymbol{\mu}}\, \mathbf{x}(\mathbf{z}_i) \\
&\quad - \nabla_{\boldsymbol{\mu}}\, H(q) && \text{(9a)} \\
&= \frac{1}{|\mathcal{Z}|}\sum_{\mathbf{z}_i\in\mathcal{Z}} \mathbf{A}_{\mathbf{x}}\big(\mathbf{x}(\mathbf{z}_i)\big)\big(\mathbf{z}_i \cdot \boldsymbol{\sigma} + \boldsymbol{\mu}\big) + \mathbf{b}_{\mathbf{x}}\big(\mathbf{x}(\mathbf{z}_i)\big) && \text{(9b)} \\
&= \Big[\frac{1}{|\mathcal{Z}|}\sum_{\mathbf{z}_i\in\mathcal{Z}} \mathbf{A}_{\mathbf{x}}\big(\mathbf{x}(\mathbf{z}_i)\big)\Big]\boldsymbol{\mu} \\
&\quad + \Big[\frac{1}{|\mathcal{Z}|}\sum_{\mathbf{z}_i\in\mathcal{Z}} \mathbf{A}_{\mathbf{x}}\big(\mathbf{x}(\mathbf{z}_i)\big)\mathbf{D}(\mathbf{z}_i)\Big]\boldsymbol{\sigma} \\
&\quad + \Big[\frac{1}{|\mathcal{Z}|}\sum_{\mathbf{z}_i\in\mathcal{Z}} \mathbf{b}_{\mathbf{x}}\big(\mathbf{x}(\mathbf{z}_i)\big)\Big] && \text{(9c)} \\
&\equiv \mathbf{A}_{\boldsymbol{\mu},\boldsymbol{\mu}}(\boldsymbol{\theta})\ \boldsymbol{\mu} + \mathbf{A}_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\boldsymbol{\theta})\ \boldsymbol{\sigma} + \mathbf{b}_{\boldsymbol{\mu}}(\boldsymbol{\theta}), && \text{(9d)}
\end{aligned}$$

where $\mathbf{D}(\mathbf{z}_i)$ denotes a diagonal matrix comprised of the elements of $\mathbf{z}_i$. The gradient w.r.t. $\boldsymbol{\sigma}$ involves the derivative of the Gaussian entropy, *i.e.* $\nabla_{\boldsymbol{\sigma}} H(q) = \nabla_{\boldsymbol{\sigma}}(\log \boldsymbol{\sigma} + \text{const})$, which can be linearized in several ways. We opt for using the element-wise second-order Taylor expansion of the logarithm around the current estimate: $\boldsymbol{\sigma}^{(t)}$:

$$\begin{aligned}
\log \boldsymbol{\sigma} &\approx \log \boldsymbol{\sigma}^{(t)} + \frac{1}{\boldsymbol{\sigma}^{(t)}}\big(\boldsymbol{\sigma} - \boldsymbol{\sigma}^{(t)}\big) - \frac{1}{\big(\boldsymbol{\sigma}^{(t)}\big)^2}\big(\boldsymbol{\sigma} - \boldsymbol{\sigma}^{(t)}\big)^2 && \text{(10a)} \\
&= \frac{1}{\boldsymbol{\sigma}^{(t)}}\boldsymbol{\sigma} - \frac{1}{\big(\boldsymbol{\sigma}^{(t)}\big)^2}\big(\boldsymbol{\sigma} - \boldsymbol{\sigma}^{(t)}\big)^2 + \text{const}. && \text{(10b)}
\end{aligned}$$

With that we can derive our stochastic approximation to the linearized gradient of the KL divergence w.r.t. $\boldsymbol{\sigma}$ as

$$\begin{aligned}
&\bar{\nabla}_{\boldsymbol{\sigma}} \operatorname{KL}(q \,||\, p) \\
&= \frac{1}{|\mathcal{Z}|}\sum_{\mathbf{z}_i\in\mathcal{Z}} \Big(\mathbf{A}_{\mathbf{x}}\big(\mathbf{x}(\mathbf{z}_i)\big)\mathbf{x}(\mathbf{z}_i) + \mathbf{b}_{\mathbf{x}}\big(\mathbf{x}(\mathbf{z}_i)\big)\Big) \cdot \nabla_{\boldsymbol{\sigma}}\mathbf{x}(\mathbf{z}_i) \\
&\quad - \nabla_{\boldsymbol{\sigma}}\, H(q) && \text{(11a)} \\
&\approx \frac{1}{|\mathcal{Z}|}\sum_{\mathbf{z}_i\in\mathcal{Z}} \mathbf{D}(\mathbf{z}_i)\Big(\mathbf{A}_{\mathbf{x}}\big(\mathbf{x}(\mathbf{z}_i)\big)\big(\mathbf{z}_i \cdot \boldsymbol{\sigma} + \boldsymbol{\mu}\big) + \mathbf{b}_{\mathbf{x}}\big(\mathbf{x}(\mathbf{z}_i)\big)\Big) \\
&\quad - \frac{3}{\boldsymbol{\sigma}^{(t)}} + \frac{2}{\big(\boldsymbol{\sigma}^{(t)}\big)^2}\boldsymbol{\sigma} && \text{(11b)} \\
&= \Big[\frac{1}{|\mathcal{Z}|}\sum_{\mathbf{z}_i\in\mathcal{Z}} \mathbf{D}(\mathbf{z}_i)\mathbf{A}_{\mathbf{x}}\big(\mathbf{x}(\mathbf{z}_i)\big)\Big]\boldsymbol{\mu} \\
&\quad + \Big[\frac{1}{|\mathcal{Z}|}\sum_{\mathbf{z}_i\in\mathcal{Z}} \mathbf{D}(\mathbf{z}_i)\mathbf{A}_{\mathbf{x}}\big(\mathbf{x}(\mathbf{z}_i)\big)\mathbf{D}(\mathbf{z}_i) + \frac{2}{\big(\boldsymbol{\sigma}^{(t)}\big)^2}\Big]\boldsymbol{\sigma} \\
&\quad + \Big[\frac{1}{|\mathcal{Z}|}\sum_{\mathbf{z}_i\in\mathcal{Z}} \mathbf{z}_i\mathbf{b}_{\mathbf{x}}\big(\mathbf{x}(\mathbf{z}_i)\big) - \frac{3}{\boldsymbol{\sigma}^{(t)}}\Big] && \text{(11c)} \\
&\equiv \mathbf{A}_{\boldsymbol{\sigma},\boldsymbol{\mu}}(\boldsymbol{\theta})\ \boldsymbol{\mu} + \mathbf{A}_{\boldsymbol{\sigma},\boldsymbol{\sigma}}(\boldsymbol{\theta})\ \boldsymbol{\sigma} + \mathbf{b}_{\boldsymbol{\sigma}}(\boldsymbol{\theta}). && \text{(11d)}
\end{aligned}$$

From Eqs. (9d) and (11d), we now obtain an approximate linearized gradient of the KL divergence in Eq. (2) with respect to $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$. Following the GL procedure, the optimization proceeds by solving the linear system of equations

$$\boldsymbol{\theta}^{(t+1)} = -\mathbf{A}_{\boldsymbol{\theta}}\Big(\boldsymbol{\theta}^{(t)}\Big)^{-1}\mathbf{b}_{\boldsymbol{\theta}}\Big(\boldsymbol{\theta}^{(t)}\Big) \tag{12}$$

with

$$\mathbf{A}_{\boldsymbol{\theta}}(\boldsymbol{\theta}) = \begin{bmatrix} \mathbf{A}_{\boldsymbol{\mu},\boldsymbol{\mu}}(\boldsymbol{\theta}) & \mathbf{A}_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\boldsymbol{\theta}) \\ \mathbf{A}_{\boldsymbol{\sigma},\boldsymbol{\mu}}(\boldsymbol{\theta}) & \mathbf{A}_{\boldsymbol{\sigma},\boldsymbol{\sigma}}(\boldsymbol{\theta}) \end{bmatrix},\ \mathbf{b}_{\boldsymbol{\theta}}(\boldsymbol{\theta}) = \begin{bmatrix} \mathbf{b}_{\boldsymbol{\mu}}(\boldsymbol{\theta}) \\ \mathbf{b}_{\boldsymbol{\sigma}}(\boldsymbol{\theta}) \end{bmatrix}. \tag{13}$$

Note that we can treat the underlying energy E as a black box. The only interaction with E is through its linearized gradient. Algorithm 1 summarizes our approach.

Discussion. Each gradient iteration in Eq. (12) can be interpreted as fitting a quadratic function to the Monte Carlo

approximation of the KL divergence (Eq. 7b), such that the quadratic approximation and the KL divergence agree on their first-order derivatives at $\boldsymbol{\theta}^{(t)}$. This alone does not guarantee that the extremum $\boldsymbol{\theta}^{(t+1)}$ of the quadratic function is actually a minimum of the approximation. Hence, we now show that the Hessian of the quadratic approximation $\mathbf{A}_{\boldsymbol{\theta}}$ is positive semi-definite, thus ensuring that $\boldsymbol{\theta}^{(t+1)}$ minimizes the approximated KL divergence.

Proposition 1. $\mathbf{A}_{\boldsymbol{\theta}}\big(\boldsymbol{\theta}^{(t)}\big)$ *is positive semi-definite,* i.e. $\boldsymbol{\theta}^T \mathbf{A}_{\boldsymbol{\theta}}\big(\boldsymbol{\theta}^{(t)}\big)\boldsymbol{\theta} \geq 0, \forall \boldsymbol{\theta}, \boldsymbol{\theta}^{(t)} \in \mathbb{R}^{2L}$*, if the matrix* $\mathbf{A}_{\mathbf{x}}\big(\mathbf{x}(\mathbf{z})\big)$ *of the energy GL is positive semi-definite for all* $\mathbf{x}(\mathbf{z})$.

Proof. Let us first assume that we just draw a single sample $\mathbf{z}$. To simplify notation let $\mathbf{A}_{\mathbf{x}} \equiv \mathbf{A}_{\mathbf{x}}\big(\mathbf{x}(\mathbf{z})\big)$ and $\mathbf{A}_{\boldsymbol{\theta}} \equiv \mathbf{A}_{\boldsymbol{\theta}}\big(\boldsymbol{\theta}^{(t)}\big)$. Now, for $\boldsymbol{\theta} = [\boldsymbol{\mu}, \boldsymbol{\sigma}]^T$ we have that

$$
\begin{aligned}
&\boldsymbol{\theta}^T \mathbf{A}_{\boldsymbol{\theta}}\, \boldsymbol{\theta} \\
&= \boldsymbol{\mu}^T \mathbf{A}_{\boldsymbol{\mu},\boldsymbol{\mu}}\, \boldsymbol{\mu} + \boldsymbol{\sigma}^T \mathbf{A}_{\boldsymbol{\sigma},\boldsymbol{\mu}}\, \boldsymbol{\mu} + \boldsymbol{\mu}^T \mathbf{A}_{\boldsymbol{\mu},\boldsymbol{\sigma}}\, \boldsymbol{\sigma} + \boldsymbol{\sigma}^T \mathbf{A}_{\boldsymbol{\sigma},\boldsymbol{\sigma}}\, \boldsymbol{\sigma} && \text{(14a)} \\
&= \boldsymbol{\mu}^T \mathbf{A}_{\mathbf{x}} \boldsymbol{\mu} + \boldsymbol{\sigma}^T \mathbf{D}(\mathbf{z})^T \mathbf{A}_{\mathbf{x}} \boldsymbol{\mu} + \boldsymbol{\mu}^T \mathbf{A}_{\mathbf{x}} \mathbf{D}(\mathbf{z}) \boldsymbol{\sigma} && \text{(14b)} \\
&\quad + \boldsymbol{\sigma}^T \mathbf{D}(\mathbf{z})^T \left(\mathbf{A}_{\mathbf{x}} + \mathbf{D}\left({}^2/_{\left(\boldsymbol{\sigma}^{(i)}\right)^2} \right) \right) \mathbf{D}(\mathbf{z}) \boldsymbol{\sigma} \\
&= \big(\boldsymbol{\mu} + \mathbf{D}(\mathbf{z})\boldsymbol{\sigma}\big)^T \mathbf{A}_{\mathbf{x}} \big(\boldsymbol{\mu} + \mathbf{D}(\mathbf{z})\boldsymbol{\sigma}\big) && \text{(14c)} \\
&\quad + \big(\mathbf{D}(\mathbf{z})\boldsymbol{\sigma}\big)^T \mathbf{D}\left({}^2/_{\left(\boldsymbol{\sigma}^{(i)}\right)^2} \right) \big(\mathbf{D}(\mathbf{z})\boldsymbol{\sigma}\big) \\
&\geq 0, && \text{(14d)}
\end{aligned}
$$

where we inserted the definition of the individual matrices (Eqs. 9d and 11d). For the last step, we used our assumption that $\mathbf{A}_{\mathbf{x}}$ is positive semi-definite. The case of multiple samples $\mathbf{z}_i$ can be shown analogously by expanding each of the four terms in Eq. (14a) into a sum. □

To put the above proposition into perspective, we now give two mild conditions on the energy function such that the corresponding matrix $\mathbf{A}_{\mathbf{x}}$ is positive semi-definite.

Proposition 2. *An energy function can be linearized with a positive semi-definite matrix* $\mathbf{A}_{\mathbf{x}}$ *if it is composed of a sum of energy terms* $\rho_i(\mathbf{w}_i)$ *that fulfill the following conditions:*

1. *Each penalty function* $\rho_i(\cdot)$ *is symmetric and* $\rho_i'(\mathbf{w}_i) \geq 0$ *for all* $\mathbf{w}_i \geq 0$. $(\star)$
2. *Each penalty function* $\rho_i(\cdot)$ *is applied element-wise on* $\mathbf{w}_i$*, which is of the form* $\mathbf{w}_i = \mathbf{K}_i\mathbf{x} + \mathbf{g}_i(\mathbf{y})$*, with filter matrix* $\mathbf{K}_i$ *and function* $\mathbf{g}_i$ *not depending on* $\mathbf{x}$. $(\star\star)$

Proof. See supplemental material. □

The above assumptions of Proposition 2 are not very restrictive but met by many MRF/CRF potentials [6], including the smoothness term used in optical flow and Poisson-Gaussian denoising, as well as the data term of our flow energy, *c.f.* Sec. 5.1 and 5.2. Moreover, positive semi-definiteness of $\mathbf{A}_{\mathbf{x}}$ can also be shown for more complex energy formulations such as the data term of Poisson-Gaussian denoising used in our experiments.

Algorithm 1 Gaussian mean field inference with SVIGL

Require: $\boldsymbol{\theta}^{(0)}$: Initial variational parameters
  $\mathbf{A}_{\mathbf{x}}, \mathbf{b}_{\mathbf{x}}$: Gradient linearization of the model energy
  for $t = 0, \ldots, T-1$ **do**
    Generate samples $\mathbf{z}_i$
    $\mathbf{x}_i \leftarrow \boldsymbol{\sigma} \cdot \mathbf{z}_i + \boldsymbol{\mu}$
    Compute $\mathbf{A}_{\mathbf{x}}\big(\mathbf{x}_i\big)$ and $\mathbf{b}_{\mathbf{x}}\big(\mathbf{x}_i\big)$
    Compute $\mathbf{A}_{\boldsymbol{\theta}}\big(\boldsymbol{\theta}^{(t)}\big)$ and $\mathbf{b}_{\boldsymbol{\theta}}\big(\boldsymbol{\theta}^{(t)}\big)$ as in Eq. (13)
    $\boldsymbol{\theta}^{(t+1)} \leftarrow -\mathbf{A}_{\boldsymbol{\theta}}\big(\boldsymbol{\theta}^{(t)}\big)^{-1} \mathbf{b}_{\boldsymbol{\theta}}\big(\boldsymbol{\theta}^{(t)}\big)$
  end for
  return $\boldsymbol{\theta}^{(T)}$

Implementation details. Solving the linear system of equations of Eq. (12) exactly is too costly for many large-scale problems, which may involve millions of variables. Hence, we consistently apply 100 iterations of successive over-relaxation [48] with a relaxation factor of 1.95 and the current estimate $\boldsymbol{\theta}^{(t)}$ as initialization. We also experimented with a conjugate gradient optimizer, but found convergence to be too slow, probably due to the need of an effective preconditioner. One limitation of our method is that we cannot guarantee that $\boldsymbol{\sigma}$ stays positive after each optimization step. Therefore, we replace each new iterate $\boldsymbol{\sigma}^{(t+1)}$ with its absolute value. In practice, however, we found that usually the entropy term is enough to force $\boldsymbol{\sigma}$ to stay positive. Since the gradient of the KL divergence cannot be expressed conveniently as linear in $\log \boldsymbol{\sigma}$, we do not use the usual trick of optimizing for $\log \boldsymbol{\sigma}$ to directly enforce positivity of $\boldsymbol{\sigma}$.

5. Experiments

We now demonstrate that SVIGL provides a convenient and efficient way of obtaining accurate variational approximations for popular energy functions of diverse computer vision problems, yielding uncertainty estimates that correlate well with estimation errors. Specifically, we quantitatively evaluate on the tasks of optical flow estimation and Poisson-Gaussian denoising. We compare SVIGL against gradient-based optimization of the KL divergence with SGD as well as the Adam optimizer [20], the default choice in the popular Edward library [40]. To assess the quality of the obtained approximate posterior, we evaluate the KL divergence $\mathrm{KL}\left(q \,||\, p\right)$ as well as application specific performance metrics. We always report KL divergences approximated by sampling (*c.f.* Eq. 6) and up to the unknown, but constant log partition function $\log Z(\mathbf{y})$.

We conduct several experiments for each application. We begin by evaluating the robustness of Adam (in the con-

text of stochastic variational inference) and SVIGL w.r.t. to their parameters. We first vary the step size α of Adam while using $|\mathcal{Z}| = 50$ samples per iteration to approximate the KL divergence gradient. Next, we use the best step size and vary the size of the sample set $|\mathcal{Z}|$ for both Adam and SVIGL. For a sample set size of 50, 25, and 12, we set the number of iterations to 100, 200, and 400 for SVIGL and 1000, 2000, and 4000 for Adam, respectively. For SGD, we similarly tune the hyperparameters and find that 4000 iterations with 12 samples and an initial step size of 10^{-6}, which is cut after each third of iterations by a factor of ten, works best for both applications. We compare the best configurations of SVIGL and SVI with SGD and Adam to a Laplace approximation and MAP estimation baselines. Runtimes refer to an Intel Xeon E5-2650v4, 2.2 GHz, 12 cores. We furthermore show qualitative results for 3D surface reconstruction to demonstrate the benefit of SVIGL for non-vision applications.

5.1. Optical flow

We first apply SVIGL to estimate an optical flow field $\mathbf{x}$, describing the motion between images $\mathbf{y} = \{I_1, I_2\}$. We use the EpicFlow energy of [32] to induce a Gibbs distribution akin to Eq. (1). Its likelihood encourages the flow to be consistent with the images and is based on a gradient consistency assumption, whereas the prior assumes small flow gradients over a 4-neighborhood, *i.e.*

$$E(\mathbf{x}, \mathbf{y}) = \lambda_{\mathrm{D}} \sum_{l=1}^{L} \rho_{\mathrm{D}} \left(\left\| \left(\nabla \tilde{I}_2(\mathbf{x}) - \nabla I_1 \right)_l \right\|_2 \right) + \lambda_{\mathrm{S}} \sum_{j=1}^{J} \sum_{l=1}^{L} \rho_{\mathrm{S}} \left(\left\| \left(\mathbf{f}_j * \mathbf{x} \right)_l \right\|_2 \right). \tag{15}$$

Here, ∇I_1 denotes the spatial derivatives of I_1, $\tilde{I}_2(\mathbf{x})$ is the second image warped by $\mathbf{x}$, and $\mathbf{f}_1, \dots, \mathbf{f}_J$ represent (derivative) filters. Functions ρ_{D} and ρ_{S} are robust penalty functions weighted with parameters λ_{D}, λ_{S}. Following standard practice, we linearize the likelihood around the current flow.

Setup. As in [45], we initialize our estimates with sparse FlowFields matches [1], densified with the EpicFlow interpolation [32]. Variances are initialized as $\sigma = 10^{-3}$. We use generalized Charbonnier penalties [2] and obtain their parameters as well as the ratio $\lambda_{\mathrm{D}}/\lambda_{\mathrm{S}}$ through Bayesian optimization [37]. To that end, we evaluate the average endpoint error (AEPE) of MAP estimates on a subset of Sintel train [9]. The absolute scale of λ_{D} and λ_{S} is subsequently calibrated such that the AEPE of the SVIGL estimates remains comparable to the MAP estimates on the training set.

Results. We conduct experiments on a validation set of 104 images randomly chosen from Sintel training (excluding images used for parameter optimization). We first motivate the use of gradient linearization by comparing the results of MAP estimation performed with up to 200 iterations of L-BFGS to 20 iterations of GL. The results averaged over the validation set are depicted in Fig. 2. We observe a significantly faster minimization of the energy using GL, which highlights its benefits for highly non-linear objectives.

We now compare SVIGL to SVI with Adam. In order to keep the runtime of Adam manageable, we perform the evaluation on manually cropped patches of size 100×100. In a first setting, we vary the step size α of Adam using 1000 iterations for Adam and 100 iterations for SVIGL. In Fig. 3a, we evaluate the KL divergence plotted against the runtime. SVIGL reduces the KL divergence two orders of magnitude faster than Adam on this challenging energy function. Moreover, the optimization by Adam is highly dependent on the chosen step size; too small or too large a value may equally lead to slow convergence. In contrast, SVIGL does not require the selection of a step size. For the following experiments we fix the step size for Adam to $\alpha = 0.005$. Now, we vary the number of samples and iterations as described above. The results are shown in Fig. 3b. Again, SVIGL attains a significantly better variational approximation than SVI with Adam for all examined settings.

Table 1 summarizes the KL divergence and the average runtime for the best settings of Adam ($\alpha = 0.01$, $|\mathcal{Z}| = 12$), SGD, and SVIGL ($|\mathcal{Z}| = 12$). In a similar runtime, SVIGL achieves a significantly lower KL divergence than SVI with Adam or SGD. We additionally evaluate the diagonal Laplace approximation around the MAP estimates using the Hessian of the linearized energy. SVIGL shows a moderate improvement over the Laplace approximation. However, the Laplace method requires second-order derivatives, which are tedious and error-prone to derive. Moreover, the Laplace approximation does not lead to consistently good results, *c.f.* Sec. 5.2.

Finally, we evaluate SVIGL on the *full-size* images of Sintel test. Since SVI with Adam is too slow, we only compare to MAP baselines with 200 iterations of L-BFGS and 20 iterations of GL, respectively. For SVIGL we use 50 samples and also 20 iterations. Both SVIGL and GL yield an AEPE of 5.74 and therefore outperform the L-BFGS baseline with an AEPE of 5.81.

Interpretation. The interdependent updates of SVIGL (Eq. 12) causes information to flow between all variables

Table 1. Unnormalized KL divergence and average runtime on 100×100 crops from a Sintel validation set.

Method	KL[$*10^7$]	runtime [s]
Initialization	5.13	–
GL + Laplace	3.83	–
SVI + SGD	4.45	**551**
SVI + Adam	4.24	1148
SVIGL *(ours)*	**3.78**	584

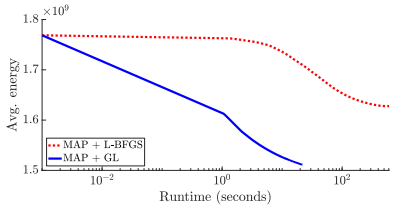
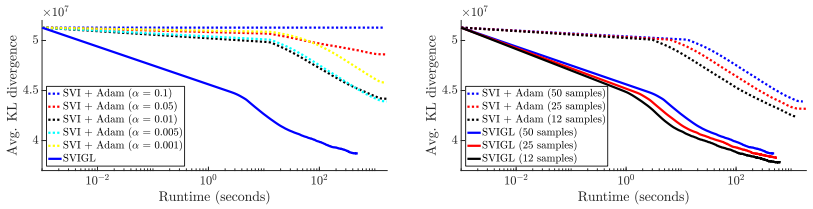


Figure 2. Optical flow energy *vs*. runtime for MAP estimation with L-BFGS and GL. Values averaged over the validation dataset. GL is clearly superior to standard L-BFGS.



(a) (b)

Figure 3. Unnormalized KL divergence *vs*. runtime for SVIGL and SVI with Adam on optical flow with different step sizes (a) and different numbers of samples and iterations (b). Values averaged over the validation set.

while a regular gradient step propagates information in a local spatial neighborhood only. We attribute the observed performance gain of GL and SVIGL over gradient-based methods at least partly to this global update.

Uncertainty estimates. Finally, we assess the quality of the per-pixel uncertainty estimates. To this end, we compare to the recent strong baseline ProbFlowFields [45]. Specifically, we apply SVIGL to update the continuous variables of their energy formulation; see supplemental material for further implementational details. Table 2 shows the metrics introduced in [45], averaged over the full-size images of our validation set. The uncertainty estimates obtained by SVIGL are competitive with the ones of ProbFlowFields. More importantly and unlike [45], the application of SVIGL does not require the tedious derivation of update equations. Example flow fields and the inferred uncertainty maps are shown in Fig. 1.

5.2. Poisson-Gaussian denoising

We next apply SVIGL to the problem of removing Poisson-Gaussian noise [15]. Here, it is assumed that image noise comes mainly from two sources that inherently affect any camera sensor. First, the Poissonian arrival process of photons hitting the pixels, and second an additive Gaussian component arising from noise in the electronics of the sensor. The Poisson distribution can be well approximated by a Gaussian [15], giving rise to a Gaussian likelihood with intensity dependent variance, *i.e*.

$$\mathbf{y}_l \sim \mathcal{N}\big(\mathbf{x}_l, \sigma(\mathbf{x}_l)^2\big) \text{ with } \sigma(\mathbf{x}_l)^2 = \beta_1 \mathbf{x}_l + \beta_2, \qquad (16)$$

where the noise distribution is specified by the parameters β_1 and β_2. We specifically set $\beta_1 = 0.05$ and $\beta_2 = 0.0001$ in order to simulate strong noise (Poisson rate 20). Combining this likelihood with a 4-connected pairwise MRF with generalized Charbonnier potentials [2] as image prior leads to the energy

$$E(\mathbf{x}, \mathbf{y}) = \frac{\lambda_\mathrm{D}}{2} \sum_{l=1}^{L} \frac{(\mathbf{x}_l - \mathbf{y}_l)^2}{\sigma(\mathbf{x}_l)^2} + \lambda_\mathrm{S} \sum_{j=1}^{J} \sum_{l=1}^{L} \rho_S\Big(\big(\mathbf{f}_j * \mathbf{x}\big)_l\Big), \qquad (17)$$

where the $\mathbf{f}_j$ denote horizontal and vertical image derivative filters. The temperature is subsumed by the weights $\lambda_\mathrm{D}, \lambda_\mathrm{S}$.

Setup. We select the relative importance of λ_D and λ_S as well as the exponent of the robust penalty through Bayesian optimization [37]. To this end, we optimize the peak-signal-to-noise ratio (PSNR) after 20 steps of GL on a set of 100 images from the BSDS training set [26]. We then calibrate the posterior for VI by determining the absolute scale of the weights on the training set. To synthesize noisy images for parameter tuning and testing, we apply Poisson-Gaussian noise to clean ground truth images. Afterwards, we rescale the intensities such that the ground truth lies in $[0, 1]$ and clip the noisy image to that range. For test time inference, we initialize $\boldsymbol{\mu}$ with the noisy image and $\boldsymbol{\sigma}$ as 10^{-3}.

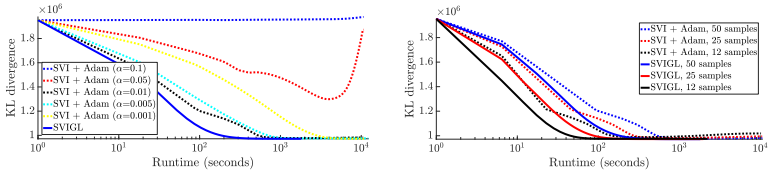
Results. In Fig. 4 we plot the unnormalized KL divergence against runtime for SVIGL and SVI with Adam, using varying step sizes for Adam and varying sizes of the sample set $\mathcal{Z}$ for both methods. It becomes apparent that the performance of Adam highly depends on these two parameters.

Table 2. AEPE, area under curve (AUC) of the sparsification plots, and Spearman's rank correlation coefficient for SVIGL and ProbFlowFields on our validation set, *c.f*. [45] for further details. †Difference in AEPE is caused by one outlier image pair.

Method	AEPE	AUC	CC
ProbFlowFields [45]	3.13	0.40	0.56
SVIGL *(ours)*	3.21†	0.42	0.50

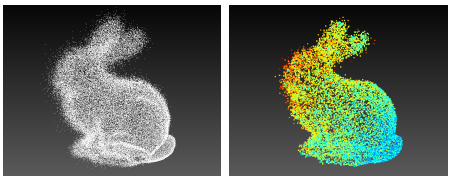
Table 3. Unnormalized KL divergences, PSNR values, and SSIM [44] for SVIGL and baseline methods in denoising.

Method	KL [$*10^6$]	PSNR [dB]	SSIM
Initialization	1.95	17.29	0.287
GL + Laplace	1.57	24.71	0.662
SVI + SGD	1.23	19.49	0.384
SVI + Adam	0.98	24.70	0.680
SVIGL *(ours)*	**0.97**	**24.77**	**0.693**
MAP + L-BFGS	–	23.17	0.605
MAP + GL	–	24.71	0.662



(a) (b)

Figure 4. Runtime *vs*. unnormalized KL divergence for denoising with SVIGL and SVI with Adam with different stepsize parameters α (a) and varying sizes of the sample set $|\mathcal{Z}|$ (b). Values averaged over the BSDS test set.



(a) (b)

Figure 5. Noisy input point cloud (a) and smoothed point cloud (b); colors indicate posterior uncertainty (blue – low, red – high).

Too small a step size slows down convergence, while setting it too high leads to a KL divergence inferior to the initialization. In contrast, SVIGL does not require setting a step size and converges faster than Adam with the best step size $\alpha = 0.01$. For instance, SVIGL reaches the same KL divergence as Adam in only $1/5$ of the time. When looking at the size of the sample set, we note that smaller sample sets speed up each iteration and hence lead to faster progress of the optimization. However, the solution found by Adam deteriorates after a certain number of iterations with smaller sample set sizes, while SVIGL is not affected by this issue. In summary, SVIGL yields faster convergence while being robust to the setting of nuisance parameters.

The converged solutions are evaluated in Table 3. SIVGL ($|\mathcal{Z}| = 50$) not only converges significantly faster than Adam ($\alpha = 0.01$, $|\mathcal{Z}| = 50$), but obtains even slightly improved solutions. SGD performs significantly worse than SVIGL and Adam. A Laplace approximation around the mode obtained with 100 iterations of GL provides a poor fit to the denoising posterior since the dependence of the variances $\sigma(\mathbf{x}_l)$ on the noise-free intensities $\mathbf{x}_l$ results in a skewed distribution. Furthermore, we see that SVIGL obtains a better solution in terms of the standard image quality metrics PSNR and SSIM [44] than the MAP estimation baselines obtained with GL and L-BFGS, *e.g.* +1.6 dB in PSNR compared to L-BFGS. In the supplemental material we show denoised images obtained by SVIGL along with their uncertainty estimates.

5.3. 3D surface reconstruction

In order to demonstrate that SVIGL is not limited to low-level problems in computer vision, we apply it to the task of reconstructing a smooth point cloud from noisy input data. Specifically, we use the energy of [25] given as

$$E(X,P,C) = \sum_{i=1}^{|X|}\sum_{j=1}^{|P|}\|x_i - p_j\| \cdot h\big(\|c_i - p_j\|\big) - \sum_{i=1}^{|X|}\sum_{i'=1}^{|C|}\lambda_i\|x_i - c_{i'}\| \cdot h\big(\|c_i - c_{i'}\|\big). \quad (18)$$

Here, $p_j \in P$ denote the noisy input points; the current and the new estimate of the smoothed points are given by $c_i \in C$ and $x_i \in X$, respectively. The contribution of each term is weighted by a Gaussian kernel $h(\cdot)$. Following Lipman *et al*. [25], we use this energy in a fixed point scheme, *i.e.*

$$X_{t+1} = \arg\min_X E(X, P, X_t), \quad (19)$$

where X_0 is an L_2 projection of the input points. The supplemental material describes the setup in more detail.

In order to exemplify the use of SVIGL for 3D surface reconstruction, we synthesize a noisy input point cloud of the Stanford bunny by adding noise on the positions of reference points. The noise strength gradually increases from tail to face. Figure 5 shows both the noisy input point cloud as well as the variational approximation from SVIGL with color coded uncertainty $\boldsymbol{\sigma}$. It is apparent that the uncertainty increases with input noise strength, thus reflecting the difficulty of the reconstruction task. Moreover, at points further away from the true surface, the uncertainty is generally higher, *c.f*. the outliers at the ears.

6. Conclusion

Motivated by the success of gradient linearization techniques for MAP estimation in highly multimodal posteriors, we proposed to combine the benefits of gradient linearization with stochastic variational inference. As a result we obtain SVIGL, an easy-to-use variational inference scheme that only requires access to a gradient linearization of the posterior energy and allows to simply repurpose well-proven energy minimization schemes. We applied SVIGL to optical flow estimation as well as Poisson-Gaussian denoising and demonstrated its significantly faster convergence compared to standard stochastic variational inference. Moreover, we showed that the optimization accuracy of SVIGL is robust to the choice of parameters. The inferred uncertainty estimates are competitive with state-of-the-art but can be obtained without tedious derivations of update equations. Finally, we demonstrate that SVIGL is not restricted to dense 2D prediction tasks by applying it successfully to the task of 3D surface reconstruction.

Acknowledgments. The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP/2007–2013)/ERC Grant agreement No. 307942.

References

[1] C. Bailer, B. Taetz, and D. Stricker. Flow fields: Dense correspondence fields for highly accurate large displacement optical flow estimation. In *ICCV*, pages 2030–2038, 2015. 6

[2] J. T. Barron. A more general robust loss function. *arXiv:1701.03077*, 2017. 6, 7

[3] J. T. Barron and B. Poole. The fast bilateral solver. In *ECCV*, volume 3, pages 617–632, 2016. 3

[4] C. H. Bischof, A. Bouaricha, P. M. Khademi, and J. J. Mor. Computing gradients in large-scale optimization using automatic differentiation. *INFORMS Journal on Computing*, 9(2):185–194, May 1997. 1

[5] M. J. Black and P. Anandan. Robust dynamic motion estimation over time. In *CVPR*, pages 296–302, 1991. 1

[6] A. Blake, P. Kohli, and C. Rother, editors. *Markov Random Fields for Vision and Image Processing*. MIT Press, 2011. 1, 5

[7] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *J. Mach. Learn. Res.*, 3(1):993–1022, Jan. 2003. 2

[8] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *ECCV*, volume 4, pages 25–36, 2004. 1, 3

[9] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, pages 611–625, 2012. 2, 6

[10] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, Sept. 1995. 3

[11] G. Chantas, N. Galatsanos, A. Likas, and M. Saunders. Variational Bayesian image restoration based on a product of t-distributions image prior. *IEEE T. Image Process.*, 17(10):1795–1805, Oct. 2008. 2, 4

[12] J. Chen and C.-K. Tang. Spatio-temporal Markov random field for video denoising. In *CVPR*, pages 2232–2239, 2007. 3

[13] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12(7):2121–2159, July 2011. 2

[14] K. Fan, Z. Wang, J. M. Beck, J. T. Kwok, and K. A. Heller. Fast second-order stochastic backpropagation for variational inference. In *NIPS*2015*, pages 1387–1395. 2

[15] A. Foi, M. Trimeche, V. Katkovnik, and K. Egiazarian. Practical Poissonian-Gaussian noise modeling and fitting for single-image raw-data. *IEEE T. Image Process.*, 17(10):1737–1754, Oct. 2008. 7

[16] D. Geman and G. Reynolds. Constrained restoration and the recovery of discontinuities. *IEEE T. Pattern Anal. Mach. Intell.*, 14(3):367–383, Mar. 1992. 3

[17] M. D. Hoffman, D. M. Blei, C. Wang, and J. W. Paisley. Stochastic variational inference. *J. Mach. Learn. Res.*, 14(1):1303–1347, May 2013. 1, 2

[18] J. Idier. Convex half-quadratic criteria and interacting auxiliary variables for image restoration. *IEEE T. Image Process.*, 10(7):1001–1009, July 2001. 3

[19] D. J. Im, S. Ahn, R. Memisevic, and Y. Bengio. Denoising criterion for variational auto-encoding framework. In *AAAI*, pages 2059–2065, 2017. 2

[20] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 2, 5

[21] D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In *ICLR*, 2014. 1, 2, 3, 4

[22] P. Krähenbühl and V. Koltun. Efficient inference in fully connected CRFs with Gaussian edge potentials. In *NIPS*2011*, pages 109–117. 2, 3

[23] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman. Efficient marginal likelihood optimization in blind deconvolution. In *CVPR*, pages 2657–2664, 2011. 2, 3, 4

[24] A. C. Likas and N. P. Galatsanos. A variational approach for Bayesian blind image deconvolution. *IEEE T. Signal Process.*, 52(8):2222–2233, Aug. 2004. 2

[25] Y. Lipman, D. Cohen-Or, D. Levin, and H. Tal-Ezer. Parameterization-free projection for geometry reconstruction. 26(3):22, 2007. 8

[26] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *ICCV*, volume 2, pages 416–423, 2001. 7

[27] J. Miskin and D. J. C. MacKay. Ensemble learning for blind image separation and deconvolution. In M. Girolami, editor, *Advances in Independent Component Analysis*, Perspectives in Neural Computing, chapter 7, pages 123–141. Springer London, 2000. 2

[28] A. Mnih and D. J. Rezende. Variational inference for Monte Carlo objectives. In *ICML*, pages 2188–2196, 2016. 2

[29] M. Nikolova and R. H. Chan. The equivalence of half-quadratic minimization and the gradient linearization iteration. *IEEE T. Image Process.*, 16(6):1623–1627, June 2007. 1, 3

[30] P. Ochs, J. Malik, and T. Brox. Segmentation of moving objects by long term video analysis. *IEEE T. Pattern Anal. Mach. Intell.*, 36(6):1187–1200, June 2014. 3

[31] R. Ranganath, S. Gerrish, and D. Blei. Black box variational inference. In *AISTATS*, pages 814–822, 2014. 1, 2

[32] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. EpicFlow: Edge-preserving interpolation of correspondences for optical flow. In *CVPR*, pages 1164–1172, 2015. 1, 3, 6

[33] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *ICML*, pages 1278–1286, 2014. 1, 2, 3

[34] H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, Mar. 1951. 2

[35] F. R. Ruiz, M. K. Titsias, and D. M. Blei. The generalized reparameterization gradient. In *NIPS*2016*, pages 460–468. 2

[36] K. Schelten and S. Roth. Mean field for continuous high-order MRFs. In *DAGM*, pages 52–61, 2012. 2

[37] J. Snoek, H. Larochelle, and R. P. Adams. Practical Bayesian optimization of machine learning algorithms. In *NIPS*2012*, pages 2951–2959. 6, 7

[38] Y. W. Teh, K. Kurihara, and M. Welling. Collapsed variational inference for HDP. In *NIPS*2007*, pages 1481–1488. 2

[39] T. Tieleman and G. Hinton. Lecture 6.5 – RMSprop: Divide the gradient by a running average of its recent magnitude. Technical report, COURSERA: Neural networks for machine learning, 2012. 2

[40] D. Tran, M. D. Hoffman, R. A. Saurous, E. Brevdo, K. Murphy, and D. M. Blei. Deep probabilistic programming. In *ICLR*, 2017. 1, 2, 5

[41] C. R. Vogel and M. E. Oman. Iterative methods for total variation denoising. *SIAM Journal on Scientific Computing*, 17(1):227–238, Jan. 1996. 3

[42] C. R. Vogel and M. E. Oman. Fast, robust total variation-based reconstruction of noisy, blurred images. *IEEE T. Image Process.*, 7(6):813–824, June 1998. 1, 3

[43] M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305, Jan. 2008. 1, 3

[44] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE T. Image Process.*, 13(4):600–612, Apr. 2004. 7, 8

[45] A. S. Wannenwetsch, M. Keuper, and S. Roth. ProbFlow: Joint optical flow and uncertainty estimation. In *ICCV*, pages 1182 – 1191, 2017. 3, 6, 7

[46] A. Wedel, A. Meißner, C. Rabe, U. Franke, and D. Cremers. Detection and segmentation of independently moving objects from dense scene flow. In *EMMCVPR*, pages 14–27, 2009. 3

[47] J. Winn and C. M. Bishop. Variational message passing. *J. Mach. Learn. Res.*, 6:661–694, Apr. 2005. 2

[48] D. M. Young. *Iterative solution of large linear systems*. Academic Press, New York, July 1971. 5

Stochastic Variational Inference with Gradient Linearization
– Supplemental Material –

Tobias Plötz* Anne S. Wannenwetsch* Stefan Roth
Department of Computer Science, TU Darmstadt

Preface. In this supplemental material, we show that SVIGL can be interpreted as a gradient descent approach using a special preconditioner and provide the gradient linearization for the optical flow and Poisson-Gaussian energies used in the main manuscript. Furthermore, we provide a proof for Proposition 2, show the hyperparameter evaluation for SVI with SGD, and give additional details of our optical flow experiment in Table 2. Finally, we show some exemplary results of Poisson-Gaussian denoising and provide details on the experiment on 3D surface reconstruction.

A. SVIGL as Preconditioned Gradient Descent

Here, we show that an update step of SVIGL as given in Eq. (12) can be interpreted as one iteration of preconditioned gradient descent. To simplify notation let $\mathbf{A}_{\boldsymbol{\theta}} \equiv \mathbf{A}_{\boldsymbol{\theta}}\big(\boldsymbol{\theta}^{(t)}\big)$ and $\mathbf{b}_{\boldsymbol{\theta}} \equiv \mathbf{b}_{\boldsymbol{\theta}}\big(\boldsymbol{\theta}^{(t)}\big)$. Following, *e.g.* [29], we have

$$\begin{aligned}
\boldsymbol{\theta}^{(t+1)} &= -\mathbf{A}_{\boldsymbol{\theta}}^{-1}\mathbf{b}_{\boldsymbol{\theta}} && \text{(20a)}\\
&= \boldsymbol{\theta}^{(t)} - \mathbf{A}_{\boldsymbol{\theta}}^{-1}\mathbf{b}_{\boldsymbol{\theta}} - \boldsymbol{\theta}^{(t)} && \text{(20b)}\\
&= \boldsymbol{\theta}^{(t)} - \mathbf{A}_{\boldsymbol{\theta}}^{-1}\Big(\mathbf{b}_{\boldsymbol{\theta}} + \mathbf{A}_{\boldsymbol{\theta}}\boldsymbol{\theta}^{(t)}\Big) && \text{(20c)}\\
&= \boldsymbol{\theta}^{(t)} - \mathbf{A}_{\boldsymbol{\theta}}^{-1}\nabla_{\boldsymbol{\theta}}\,\mathrm{KL}\left(q\,||\,p\right). && \text{(20d)}
\end{aligned}$$

Therefore, SVIGL performs gradient descent with preconditioner $P = \mathbf{A}_{\boldsymbol{\theta}}^{-1}$. This interpretation also allows to introduce a step size parameter α to SVIGL

$$\begin{aligned}
\boldsymbol{\theta}^{(t+1)} &= \boldsymbol{\theta}^{(t)} - \alpha\mathbf{A}_{\boldsymbol{\theta}}^{-1}\nabla_{\boldsymbol{\theta}}\,\mathrm{KL}\left(q\,||\,p\right) && \text{(21a)}\\
&= \boldsymbol{\theta}^{(t)} - \alpha\mathbf{A}_{\boldsymbol{\theta}}^{-1}\Big(\mathbf{b}_{\boldsymbol{\theta}} + \mathbf{A}_{\boldsymbol{\theta}}\boldsymbol{\theta}^{(t)}\Big) && \text{(21b)}\\
&= (1-\alpha)\boldsymbol{\theta}^{(t)} + \alpha\hat{\boldsymbol{\theta}}^{(t+1)}, && \text{(21c)}
\end{aligned}$$

with $\hat{\boldsymbol{\theta}}^{(t+1)} = -\mathbf{A}_{\boldsymbol{\theta}}^{-1}\mathbf{b}_{\boldsymbol{\theta}}$ denoting the SVIGL estimate as given in Eq. (12). In practice, our experiments have shown that the performance of SVIGL is not sensitive to the choice of the step size parameter. We thus simply set $\alpha = 1$.

*Authors contributed equally

B. Linearized Gradients

In the following, we show how linearized gradients can be obtained for the presented applications of SVIGL in optical flow estimation and Poisson-Gaussian denoising. For other applications, including many models in computer vision, it is possible to derive parameters $\mathbf{A}_{\boldsymbol{\theta}}$ and $\mathbf{b}_{\boldsymbol{\theta}}$ in a similar fashion.

B.1. Optical flow

Here, we show the derivation of a linearized gradient for a simple optical flow energy using the brightness constancy assumption, *i.e.*

$$\begin{aligned}
E(\mathbf{x},\mathbf{y}) =& \lambda_{\mathrm{D}}\sum_{l=1}^{L}\rho_{\mathrm{D}}\left(I_{t,l} + \begin{pmatrix}I_{x,l}\\ I_{y,l}\end{pmatrix}^{\mathrm{T}}\left(\mathbf{x}_l - \mathbf{x}_l^0\right)\right)\\
&+ \lambda_{\mathrm{S}}\sum_{j=1}^{J}\sum_{l=1}^{L}\rho_{\mathrm{S}}\left(\left\|\left(\mathbf{f}_j * \mathbf{x}\right)_l\right\|_2\right) && \text{(22a)}\\
=& \lambda_{\mathrm{D}}E_{\mathrm{D}}(\mathbf{x},\mathbf{y}) + \lambda_{\mathrm{S}}E_{\mathrm{S}}(\mathbf{x}), && \text{(22b)}
\end{aligned}$$

with $I_{t,l} = I_2\left(l + \mathbf{x}_l^0\right) - I_1\left(l\right)$, $\begin{pmatrix}I_{x,l}\\ I_{y,l}\end{pmatrix} = \nabla I_2\left(l + \mathbf{x}_l^0\right)$, and $\mathbf{x}_l^0$ denoting the point of approximation of the Taylor linearization. The derivations for the EpicFlow energy function in Eq. (15) are more tedious, but can be done analogously.

Data term. In a first step, we derive the linearized gradient for the data energy term. Here, it holds that

$$\begin{aligned}
\nabla_{\mathbf{x}_l}E_{\mathrm{D}}(\mathbf{x},\mathbf{y}) =& \nabla_{\mathbf{x}_l}\rho_{\mathrm{D}}\left(I_{t,l} + \begin{pmatrix}I_{x,l}\\ I_{y,l}\end{pmatrix}^{\mathrm{T}}\left(\mathbf{x}_l - \mathbf{x}_l^0\right)\right) && \text{(23a)}\\
=& \rho_{\mathrm{D}}'\left(I_{t,l} + \begin{pmatrix}I_{x,l}\\ I_{y,l}\end{pmatrix}^{\mathrm{T}}\left(\mathbf{x}_l - \mathbf{x}_l^0\right)\right)\cdot\begin{pmatrix}I_{x,l}\\ I_{y,l}\end{pmatrix}. && \text{(23b)}
\end{aligned}$$

The derivative of the generalized Charbonnier [2] used for

$\rho_{\mathrm{D}}(\cdot)$ can be written as:

$$\rho'_{\mathrm{D}}(x) = \frac{x}{c^2}\left(\frac{(x/c)^2}{\max(1, 2-a)} + 1\right)^{(a/2-1)} \tag{24a}$$

$$\equiv \tilde{\rho}_{\mathrm{D}}(x)\, x. \tag{24b}$$

Using Eqs. (23b) and (24b), we have

$$\begin{aligned}\nabla_{\mathbf{x}_l} E_{\mathrm{D}}(\mathbf{x}, \mathbf{y}) = \\ = \tilde{\rho}_{\mathrm{D}}\left(I_{t,l} + \begin{pmatrix} I_{x,l} \\ I_{y,l}\end{pmatrix}^{\mathrm{T}}\left(\mathbf{x}_l - \mathbf{x}_l^0\right)\right) \\ \cdot \left(\begin{pmatrix} I_{x,l}I_{t,l} \\ I_{y,l}I_{t,l}\end{pmatrix} + \begin{pmatrix} I_{x,l}^2 & I_{x,l}I_{y,l} \\ I_{x,l}I_{y,l} & I_{y,l}^2\end{pmatrix}\left(\mathbf{x}_l - \mathbf{x}_l^0\right)\right).\end{aligned} \tag{25}$$

The last identity (Eq. 25) allows us to easily identify a linearized form of the gradient of the data term as

$$\nabla_{\mathbf{x}} E_{\mathrm{D}}(\mathbf{x}, \mathbf{y}) = \mathbf{A}_{\mathbf{x}}^{\mathrm{D}}(\mathbf{x})\mathbf{x} + \mathbf{b}_{\mathbf{x}}^{\mathrm{D}}(\mathbf{x}), \tag{26}$$

with

$$\mathbf{A}_{\mathbf{x}}^{\mathrm{D}}(\mathbf{x}) = \begin{pmatrix} \mathbf{D}\left(\tilde{\rho}_{\mathrm{D}} \cdot I_x^2\right) & \mathbf{D}\left(\tilde{\rho}_{\mathrm{D}} \cdot I_x I_y\right) \\ \mathbf{D}\left(\tilde{\rho}_{\mathrm{D}} \cdot I_x I_y\right) & \mathbf{D}\left(\tilde{\rho}_{\mathrm{D}} \cdot I_y^2\right)\end{pmatrix} \tag{27}$$

and

$$\mathbf{b}_{\mathbf{x}}^{\mathrm{D}}(\mathbf{x}) = \begin{pmatrix} \mathbf{D}\left(\tilde{\rho}_{\mathrm{D}} \cdot I_x I_t\right)\mathbf{1} \\ \mathbf{D}\left(\tilde{\rho}_{\mathrm{D}} \cdot I_y I_t\right)\mathbf{1}\end{pmatrix} - \mathbf{A}_{\mathbf{x}}(\mathbf{x})\mathbf{x}^0. \tag{28}$$

Here, $\mathbf{x} = \left(x_1^{(1)}, \ldots, x_L^{(1)}, x_1^{(2)}, \ldots, x_L^{(2)}\right)^{\mathrm{T}}$ denotes the stacked vector of all horizontal and vertical flow components. $\mathbf{D}(\cdot)$ turns the argument vector into a diagonal matrix (short for $\mathrm{diag}\{\cdot\}$), and the product is applied element-wise.

Smoothness term. For the smoothness term let us first express the convolution $\mathbf{f}_j * \mathbf{x}$ as a matrix-vector product $\mathbf{F}_j \cdot \mathbf{x}$, with $\mathbf{F}_j$ denoting the convolution matrix corresponding to $\mathbf{f}_j$ and $\mathbf{x}$ the vectorized flow as before. With that, the gradient of the smoothness term E_{S} can be written as:

$$\nabla_{\mathbf{x}} E_{\mathrm{S}}(\mathbf{x}) = \nabla_{\mathbf{x}} \sum_{j=1}^{J}\sum_{l=1}^{L} \rho_{\mathrm{S}}\left(\left(\mathbf{F}_j\mathbf{x}\right)_l\right) \tag{29a}$$

$$= \sum_{j=1}^{J} \mathbf{F}_j^T \rho'_{\mathrm{S}}(\mathbf{F}_j\mathbf{x}). \tag{29b}$$

Using the derivative ρ'_{S} as given in Eq. (24b), we obtain

$$\sum_{j=1}^{J} \mathbf{F}_j^T \rho'_{\mathrm{S}}(\mathbf{F}_j\mathbf{x}) = \sum_{j=1}^{J} \mathbf{F}_j^T\, \mathbf{D}\left(\tilde{\rho}_{\mathrm{S}}\left(\mathbf{F}_j\mathbf{x}\right)\right)\, \mathbf{F}_j\mathbf{x} \tag{30a}$$

$$= \left(\sum_{j=1}^{J} \mathbf{F}_j^T\, \mathbf{D}\left(\tilde{\rho}_{\mathrm{S}}\left(\mathbf{F}_j\mathbf{x}\right)\right)\, \mathbf{F}_j\right)\mathbf{x} \tag{30b}$$

$$\equiv \mathbf{A}_{\mathbf{x}}^{\mathrm{S}}(\mathbf{x})\mathbf{x}. \tag{30c}$$

Complete linearized gradient. We now summarize the results of Eqs. (27), (28), and (30c) to obtain the linearized gradient as

$$\nabla_{\mathbf{x}} E(\mathbf{x}, \mathbf{y}) = \lambda_{\mathrm{D}} \nabla_{\mathbf{x}} E_{\mathrm{D}}(\mathbf{x}, \mathbf{y}) + \lambda_{\mathrm{S}} \nabla_{\mathbf{x}} E_{\mathrm{S}}(\mathbf{x}) \tag{31a}$$

$$= \left(\lambda_{\mathrm{D}}\mathbf{A}_{\mathbf{x}}^{\mathrm{D}}(\mathbf{x}) + \lambda_{\mathrm{S}}\mathbf{A}_{\mathbf{x}}^{\mathrm{S}}(\mathbf{x})\right)\mathbf{x} + \lambda_{\mathrm{D}}\mathbf{b}_{\mathbf{x}}^{\mathrm{D}} \tag{31b}$$

$$\equiv \mathbf{A}_{\mathbf{x}}(\mathbf{x})\mathbf{x} + \mathbf{b}_{\mathbf{x}}. \tag{31c}$$

B.2. Poisson-Gaussian denoising

Let us first recap the energy function for Poisson-Gaussian denoising:

$$E(\mathbf{x}, \mathbf{y}) = \frac{\lambda_{\mathrm{D}}}{2}\sum_{l=1}^{L}\frac{\left(\mathbf{x}_l - \mathbf{y}_l\right)^2}{\sigma(\mathbf{x}_l)^2} + \lambda_{\mathrm{S}}\sum_{j=1}^{J}\sum_{l=1}^{L}\rho_{\mathrm{S}}\left(\left(\mathbf{f}_j * \mathbf{x}\right)_l\right), \tag{32a}$$

$$= \lambda_{\mathrm{D}} E_{\mathrm{D}}(\mathbf{x}, \mathbf{y}) + \lambda_{\mathrm{S}} E_{\mathrm{S}}(\mathbf{x}), \tag{32b}$$

where

$$\sigma(\mathbf{x}_l)^2 = \beta_1 \mathbf{x}_l + \beta_2. \tag{33}$$

We will derive the linearized gradients for the data term E_{D} and the smoothness term E_{S} separately.

Data term. The gradient of the data term is given as

$$\nabla_{\mathbf{x}} E_{\mathrm{D}}(\mathbf{x}, \mathbf{y}) = \frac{(\mathbf{x} - \mathbf{y})}{\sigma(\mathbf{x})^2} - \frac{\beta_1(\mathbf{x} - \mathbf{y})^2}{2\sigma(\mathbf{x})^4} \tag{34a}$$

$$= \frac{\mathbf{x}}{\sigma(\mathbf{x})^2} - \frac{\mathbf{y}}{\sigma(\mathbf{x})^2} - \frac{\beta_1\mathbf{x}^2}{2\sigma(\mathbf{x})^4} + \frac{\beta_1\mathbf{x}\mathbf{y}}{\sigma(\mathbf{x})^4} - \frac{\beta_1\mathbf{y}^2}{2\sigma(\mathbf{x})^4} \tag{34b}$$

$$= \mathbf{x}\left(\frac{1}{\sigma(\mathbf{x})^2} - \frac{\beta_1\mathbf{x}}{2\sigma(\mathbf{x})^4} + \frac{\beta_1\mathbf{y}}{\sigma(\mathbf{x})^4}\right) - \left(\frac{\mathbf{y}}{\sigma(\mathbf{x})^2} + \frac{\beta_1\mathbf{y}^2}{2\sigma(\mathbf{x})^4}\right), \tag{34c}$$

where all operations are element-wise. The linearized gradient of the data term can then be obtained as

$$\mathbf{A}_{\mathbf{x}}^{\mathrm{D}}(\mathbf{x}) = \mathbf{D}\left(\frac{1}{\sigma(\mathbf{x})^2} - \frac{\beta_1 \mathbf{x}}{2\sigma(\mathbf{x})^4} + \frac{\beta_1 \mathbf{y}}{\sigma(\mathbf{x})^4}\right) \tag{35}$$

$$\mathbf{b}_{\mathbf{x}}^{\mathrm{D}}(\mathbf{x}) = -\left(\frac{\mathbf{y}}{\sigma(\mathbf{x})^2} + \frac{\beta_1 \mathbf{y}^2}{2\sigma(\mathbf{x})^4}\right). \tag{36}$$

Smoothness term. For the smoothness term we can reuse the linearized gradient derived in Eq. (30c).

Complete linearized gradient. We can now put the results of Eqs. (30c), (35), and (36) together to obtain a linearized gradient of the energy for Poisson-Gaussian denoising, *c.f.* Eqs. (31a) – (31c).

C. Proof Proposition 2

In this section, we provide a proof for Proposition 2 of the main paper.

Proposition 2. *An energy function can be linearized with a positive semi-definite matrix $\mathbf{A}_{\mathbf{x}}$ if it is composed of a sum of energy terms $\rho_i(\mathbf{w}_i)$ that fulfill the following conditions:*

1. *Each penalty function $\rho_i(\cdot)$ is symmetric and $\rho_i'(\mathbf{w}_i) \geq 0$ for all $\mathbf{w}_i \geq 0$. $(\star)$*
2. *Each penalty function $\rho_i(\cdot)$ is applied element-wise on $\mathbf{w}_i$, which is of the form $\mathbf{w}_i = \mathbf{K}_i\mathbf{x} + \mathbf{g}_i(\mathbf{y})$, with filter matrix $\mathbf{K}_i$ and $\mathbf{g}_i$ not depending on $\mathbf{x}$. $(\star\star)$*

Proof. From assuming a symmetric $\rho_i(\cdot)$ in $(\star)$, it follows that $\rho_i'(\cdot)$ is point symmetric. Due to $\rho_i'(\mathbf{w}_i) \geq 0$ for all $\mathbf{w}_i \geq 0$ we then find that $\rho_i'(\mathbf{w}_i)$ can be written as

$$\rho_i'(\mathbf{w}_i) \equiv \tilde{\rho}_i(\mathbf{w}_i) \cdot \mathbf{w}_i \quad \text{with a} \quad \tilde{\rho}_i(\mathbf{w}_i) \geq 0. \tag{37}$$

For an energy term as described in $(\star\star)$, the gradient w.r.t. $\mathbf{x}$ is given as

$$\nabla_{\mathbf{x}} \rho_i(\mathbf{w}_i) = \mathbf{K}_i^{\mathrm{T}} \cdot \mathbf{C}_i \cdot (\mathbf{K}_i \cdot \mathbf{x} + \mathbf{g}_i(\mathbf{y})), \tag{38}$$

$$\text{with} \quad \mathbf{C}_i = \mathbf{D}\big(\tilde{\rho}_i\left(\mathbf{K}_i \cdot \mathbf{x} + \mathbf{g}_i(\mathbf{y})\right)\big). \tag{39}$$

A linearization can then be obtained using

$$\mathbf{A}_{\mathbf{x}}^i = \mathbf{K}_i^{\mathrm{T}} \cdot \mathbf{C}_i \cdot \mathbf{K}_i, \qquad \mathbf{b}_{\mathbf{x}}^i = \mathbf{K}_i^{\mathrm{T}} \cdot \mathbf{C}_i \cdot \mathbf{g}_i(\mathbf{y}). \tag{40}$$

Since $\mathbf{C}_i$ is a diagonal matrix of non-negative elements (Eq. 37), $\mathbf{A}_{\mathbf{x}}^i$ is positive semi-definite as

$$\mathbf{x}^{\mathrm{T}} \mathbf{A}_{\mathbf{x}}^i \mathbf{x} = \mathbf{x}^{\mathrm{T}} \mathbf{K}_i^{\mathrm{T}} \mathbf{C}_i \mathbf{K}_i \mathbf{x} = \mathbf{v}^{\mathrm{T}} \mathbf{C}_i \mathbf{v} \geq 0. \tag{41}$$

As the sum of positive semi-definite matrices is positive semi-definite, a matrix $\mathbf{A}_{\mathbf{x}}$ composed of energy terms that fulfill $(\star)$ and $(\star\star)$ is positive semi-definite. □

D. Hyperparameters for SGD

In the following, we aim to find optimal hyperparameters for the SVI baseline based on SGD. For all experiments we select an initial step size α_0, which is cut after each third of iterations by a factor of ten. An evaluation of the unnormalized KL divergence for optical flow plotted against the runtime for different initial step sizes α_0 of SGD is shown in Fig. 6a. Here, the KL divergence deteriorates severely using SGD with a step size larger than 10^{-6}. For smaller step sizes, SVI with SGD shows a slow convergence such that we set $\alpha_0 = 10^{-6}$.

Following the same procedure, we perform several experiments for Poisson-Gaussian denoising and evaluate different settings for the initial step size parameter α_0 of SGD in Fig. 7a. Again, an initial step size $\alpha_0 = 10^{-6}$ proves to be most effective. Smaller step sizes converge too slowly, while SGD with bigger step size values converges faster but to a worse local optimum. For an initial step size of $\alpha_0 = 10^{-5}$ optimization diverges immediately.

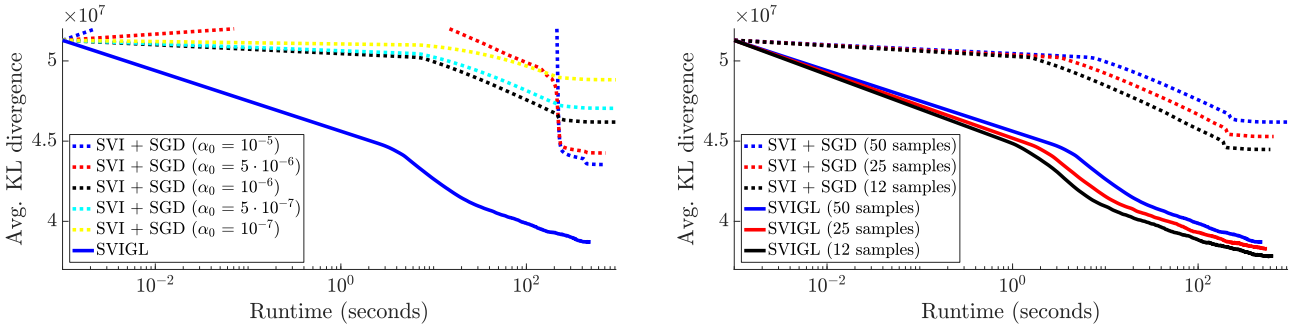
Applying SVI with SGD, we observe in both applications a faster convergence of the KL divergence with a smaller sample size, but a larger number of iterations, *c.f.* Figs. 6b and 7b. We therefore choose $|\mathcal{Z}| = 12$ with 4000 iterations of SGD for the experiments in the main paper.

E. Comparison with ProbFlowFields

In Table 2 of the main paper we evaluate the quality of the posterior variances obtained with SVIGL. Here, we follow Wannenwetsch *et al.* [45] and derive an uncertainty measure by computing the marginal entropy of the flow at every pixel. To have a fair comparison with [45], we use the same EpicFlow [32] energy formulation with learned Gaussian scale mixture penalty functions and explicit indicator variables for their mixture components. Since SVIGL is designed for variational inference in distributions with continuous random variables, we alternate closed-form updates of the latent indicator variables with SVIGL updates for the continuous flow variables. For the discrete update, we approximate the tedious analytical expectation values over the flow variables with a Monte-Carlo estimator (*c.f.* Eq. 7b). This effectively reduces the optimization w.r.t. the indicator variables to an independent update – thus maintaining the ease of use of SVIGL. Weighting parameters λ_{D} and λ_{S} are determined on a training set with Bayesian optimization [37] using the F1-score as described in [45].

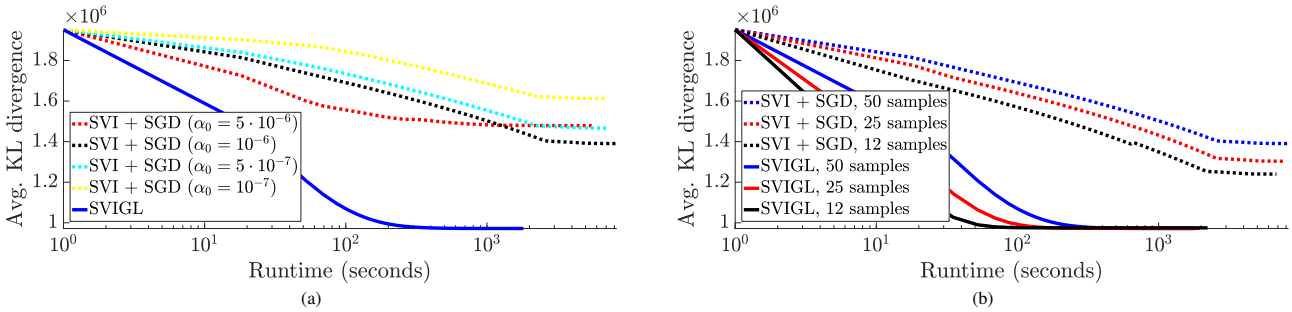
F. Results of Poisson-Gaussian Denoising

Fig. 8 shows some example results of SVIGL applied to Poisson-Gaussian denoising on the BSDS dataset. High uncertainties can be observed especially on object boundaries. Due to the high amount of noise, a strong smoothness term



(a) (b)

Figure 6. Unnormalized KL divergence *vs*. runtime for optical flow with SVIGL and SVI with SGD with different step sizes (a) and different numbers of samples and iterations (b). Values averaged on the validation set.



(a) (b)

Figure 7. Unnormalized KL divergence *vs*. runtime for denoising with SVIGL and SVI with SGD with different step sizes (a) and with different numbers of samples and iterations (b). Values averaged over the BSDS test set.

maximizes the PSNR on the training set. Therefore, the denoised images tend to be rather smooth in general.

G. Results on Sintel Test

As described in Sec. 5.1, we apply SVIGL as well as two MAP baselines on the full-sized Sintel test images in order to evaluate their performance. Figure 9 shows a screenshot of the private Sintel benchmark table with results for both methods. SVIGL outperforms the underlying FlowFields method [1] as well as the L-BFGS baseline and shows an AEPE result on par with the corresponding MAP estimate using GL. Moreover, SVIGL estimates are competitive with the finetuned version of FlowNet2 [49], *i.e*. the state-of-the-art baseline for optical flow prediction with convolutional neural networks.

H. 3D Surface Reconstruction

We now give more details on the application of SVIGL to 3D surface reconstruction. First, we restate the energy of Lipman *et al*. [25], which is given by

$$E(X,P,C) = \sum_{i=1}^{|X|}\sum_{j=1}^{|P|}\|x_i - p_j\| \cdot h\big(\|c_i - p_j\|\big) - \sum_{i=1}^{|X|}\sum_{i'=1}^{|C|}\lambda_i\|x_i - c_{i'}\| \cdot h\big(\|c_i - c_{i'}\|\big). \quad (42)$$

Here, $p_j \in P$ denote the noisy input points, $c_i \in C$ are the current estimates of the smoothed points, and $x_i \in X$ the new estimates of the smoothed points. While the first part of the energy forces the new estimates to be close to the input points, the second term pushes the reconstructed points apart by penalizing points in X that are too close to points in C. The contribution of each term is weighted by the Gaussian kernel $h(\cdot)$.

A closed-form solution to minimizing the above energy is given in [25]. This solution is then used in a fixed point scheme as

$$X_{t+1} = \arg\min_X E(X, P, X_t), \quad (43)$$

where X_0 is initialized as a L_2 projection of the input points.

In a variational inference setting, closed-form updates are no longer possible due to introducing the additional vari-

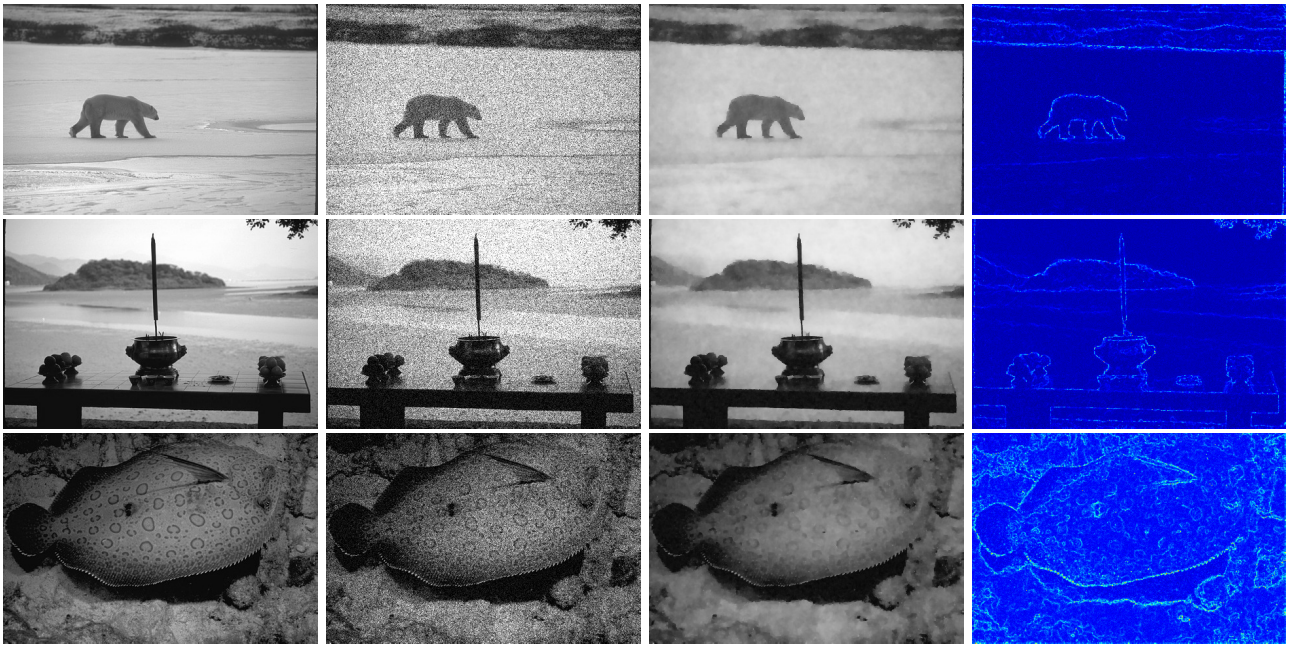


Figure 8. Examples of ground truth (left), noisy images (second column), estimated clean images (third column), and uncertainty estimates (right) from SVIGL on the BSDS test set.

	EPE all	EPE matched	EPE unmatched	d0-10	d10-60	d60-140	s0-10	s10-40	s40+
GroundTruth [1]	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
⋮			⋮						⋮
MAP_GL [25]	5.735	2.524	31.896	4.789	2.126	1.619	1.107	3.659	33.708
FlowNet2-ft-sintel [26]	5.739	2.752	30.108	4.818	2.557	1.735	0.959	3.228	35.538
SVIGL [27]	5.740	2.526	31.924	4.792	2.128	1.619	1.107	3.661	33.740
MAP_LBFGS [28]	5.805	2.626	31.710	4.890	2.233	1.709	1.206	3.803	33.456
FlowFields [29]	5.810	2.621	31.799	4.851	2.232	1.682	1.157	3.739	33.890

Figure 9. Screenshot of the private Sintel benchmark table (final) with results for SVIGL, MAP + GL, MAP + L-BFGS, and the original FlowFields approach [1] (status as of March 2018).

ance variables $\boldsymbol{\sigma}$ of the variational posterior. Hence, we employ SVIGL updates instead. To be able to apply SVIGL, we require a linearization of the energy gradient. The specific form of the energy in Eq. (19) allows for a diagonal linearization:

$$\nabla_{x_i} E(X,P,C) = \sum_{j\in J}(x_i - p_j)\frac{h(\|c_i - p_j\|)}{\|x_i - p_j\|} - \sum_{i'\in I}(x_i - c_{i'})\frac{h(\|c_i - c_{i'}\|)}{\|x_i - c_{i'}\|}. \quad (44)$$

In total, we run 10 iterations of Eq. (43). In each iteration, we compute a single SVIGL update with a sample set size of $|\mathcal{Z}| = 5$.

References

[49] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *CVPR*, pages 1647 – 1655, 2017. 4

A.3 LEARNING TASK-SPECIFIC GENERALIZED CONVOLUTIONS IN THE PERMUTOHEDRAL LATTICE

Anne S. Wannenwetsch, Martin Kiefel, Peter V. Gehler, and Stefan Roth
41st DAGM German Conference on Pattern Recognition (DAGM GCPR 2019)

Abstract

Dense prediction tasks typically employ encoder-decoder architectures, but the prevalent convolutions in the decoder are not image-adaptive and can lead to boundary artifacts. Different generalized convolution operations have been introduced to counteract this. We go beyond these by leveraging guidance data to redefine their inherent notion of *proximity*. Our proposed network layer builds on the *permutohedral lattice*, which performs sparse convolutions in a high-dimensional space allowing for powerful non-local operations despite small filters. Multiple features with different characteristics span this permutohedral space. In contrast to prior work, we *learn* these features in a task-specific manner by generalizing the basic permutohedral operations to learnt feature representations. As the resulting objective is complex, a carefully designed framework and learning procedure are introduced, yielding rich feature embeddings in practice. We demonstrate the general applicability of our approach in different joint upsampling tasks. When adding our network layer to state-of-the-art networks for optical flow and semantic segmentation, boundary artifacts are removed and the accuracy is improved.

Learning Task-Specific Generalized Convolutions in the Permutohedral Lattice

Anne S. Wannenwetsch[1⋆][0000−0002−7016−3820],
Martin Kiefel[2][0000−0001−9432−5428],
Peter V. Gehler[2][0000−0002−5812−4052], Stefan Roth[1][0000−0001−9002−9832]

[1] TU Darmstadt, Germany [2] Amazon, Germany

Abstract. Dense prediction tasks typically employ encoder-decoder architectures, but the prevalent convolutions in the decoder are not image-adaptive and can lead to boundary artifacts. Different generalized convolution operations have been introduced to counteract this. We go beyond these by leveraging guidance data to redefine their inherent notion of *proximity*. Our proposed network layer builds on the *permutohedral lattice*, which performs sparse convolutions in a high-dimensional space allowing for powerful non-local operations despite small filters. Multiple features with different characteristics span this permutohedral space. In contrast to prior work, we *learn* these features in a task-specific manner by generalizing the basic permutohedral operations to learnt feature representations. As the resulting objective is complex, a carefully designed framework and learning procedure are introduced, yielding rich feature embeddings in practice. We demonstrate the general applicability of our approach in different joint upsampling tasks. When adding our network layer to state-of-the-art networks for optical flow and semantic segmentation, boundary artifacts are removed and the accuracy is improved.

1 Introduction

Deep learning approaches are the backbone of many state-of-the-art methods across computer vision [7,36,39]. Convolutional neural networks (CNNs) are particularly common as they greatly lower the number of parameters compared to fully-connected networks and thus scale to practically relevant image sizes. While early CNNs employed large filters [28], it is now common to use small kernels stacked into deep networks [16,35]. Chaining several smaller filters requires fewer parameters for the same receptive field of a single large filter, and leads to more discriminative features by virtue of having more non-linearities [35].

While convolutions build a fundamental block of deep learning, they are not without drawbacks. First, they are not image-adaptive, *i.e.* content boundaries in a feature map are not respected but smoothed over. This is especially disadvantageous for dense prediction tasks, *e.g.* semantic segmentation or optical flow, leading to accuracy loss at boundaries [14,46]. Moreover, convolutions have

⋆ This project was mainly done during an internship at Amazon, Germany.

This is a post-peer-review, pre-copyedit version of an article published in DAGM GCPR 2019: Pattern Recognition. The final authenticated version is available online at: https://doi.org/10.1007/978-3-030-33676-9_24.

a limited and predefined receptive field, which connects spatially close regions but cannot leverage similar, but more distant image structures. Here, a new definition of *pixel proximity* is needed that goes beyond two-dimensional (2D) spatial distance. For instance, image values themselves or abstract properties such as object classes could be used to define similarity in a more general setting.

Several methods have been proposed to counteract the named disadvantages. Sampling-based approaches [19,33] rearrange the image content but remain restricted to the 2D concept of proximity. Location specific networks [22,46] predict pixelwise filter kernels, but require many additional parameters. Other methods [8,40] determine neighboring pixels in an image-adaptive manner. However, the convolutional structure is fixed and only the position of neighbors is adjustable.

Image-adaptive filters, *e.g.* [15,41], have been used in traditional computer vision for years. The bilateral filter [41] adapts a Gaussian kernel according to the spatial distance and color difference of neighboring pixels. In [20,24], this concept is leveraged to construct bilateral convolution layers (BCLs) based on the *permutohedral lattice* [1] – a fast approximation of the bilateral filter. Filtering corresponds to a sparse convolution in a high-dimensional space, which is spanned by different features, *e.g.* spatial location and color. Jampani *et al.* [20,24] extend the Gaussian kernel to a general, image-adaptive convolution and learn the kernels from data. However, the features constituting the lattice space remain fixed. Feature parameters are not adjustable during training, which complicates integration into end-to-end learning. More importantly, relying on predefined features without further processing omits a possible source of improvements.

To counteract this disadvantage of BCLs, we present the *semantic lattice layer*. We rely on the permutohedral lattice as a backbone and show how to generalize its operations w.r.t. features with learnable parameters. The resulting computations are involved and may lead to practical challenges. We hence propose a specific setting in which basic features – as used in [20,24] – are processed by a CNN. This greatly simplifies the optimization since it allows to combine and especially refine features that are known to be beneficial for image-adaptive filtering. We further present various measures to avoid difficulties during learning. For instance, as the sparsity of the semantic lattice may avoid propagation of information if pixels are too distant, we restrict the output range of the embedded features. This rather simple measure has a large effect in practice.

Our setup enables us to learn meaningful feature embeddings from data. It allows to integrate feature parameters into training and effectively leverages guidance data to connect pixels due to their similar characteristics. As such, the semantic lattice is able to perform non-local operations while keeping the filter kernels and consequently the number of learnt parameters small and manageable.

We show the benefits of the semantic lattice in different areas for image-adaptive upsampling. For the task of color upsampling, the semantic lattice outperforms previous approaches by a large margin. We further replace bilinear upsampling in state-of-the-art networks for optical flow and semantic segmentation. Here, the semantic lattice leads to better aligned and crisper content boundaries and also improves the accuracy, especially at discontinuities.

2 Related Work

Generalized Convolutions. We begin by reviewing work that generalizes convolution operations. Jaderberg *et al.* [19] introduce Spatial Transformers (STs), which transform feature maps depending on the data itself. Similar to warped convolutions [17], STs aim for invariance to certain transformations, *e.g.* rotation or scaling. [29] applies STs to allow for irregular patches in dense prediction tasks. In [33], saliency-based sampling emphasizes regions of high interest. These methods rearrange data in 2D space. In contrast, we can leverage additional feature dimensions to redefine the concept of pixel proximity.

Filter-weight networks [22] generate location-specific filters dependent on the input image. In [23], adaptive weights incorporate side information about the scene context. However, adaptive filters introduce many additional parameters in comparison to location-invariant networks and remain restricted to local transformations due to a fixed receptive field. Wu *et al.* [46] apply location-specific convolutions not only to the position itself but to several sampled neighboring regions, which extends their receptive field but requires additional computations.

Dilated convolutions use a fixed spacing between considered pixels to extend the spatial resolution [6,47]. It is possible to learn offsets for the input locations of each filter [21] or have them depend on the input and spatial location [8]. When using mixtures of Gaussians as filters, size and location of the receptive fields are learnable [40]. Structure-aware convolutions [5] use univariate functions as filters and are also applicable to non-Euclidean data. We do not learn individual neighborhoods for all filters but convolutions are instead performed consistently in a learnt feature space. Moreover, our convolution structure is not fixed; the number of neighbors is flexible and homogenous areas can be compressed.

Permutohedral Lattice. Adams *et al.* [1] propose the permutohedral lattice as a fast method for high-dimensional Gaussian filtering. It found widespread application, especially for fast inference in dense Conditional Random Fields (CRFs) [26,32,48] and upsampling or densification of data [10,34]. In contrast to our work, these approaches use fixed Gaussian filters and predefined features. [27] extends the fast inference method of [26] to learn parameters of dense CRFs, but the setup is restricted to Gaussian filters and customized to its application.

In [20,24], the high-dimensional filtering in permutohedral space is generalized by learnable convolution parameters. The proposed BCLs are beneficial in neural networks as they allow to redefine proximity of pixels w.r.t. different characteristics [12,20,30]. Moreover, BCLs can inherently cope with sparse data [24], *e.g.* in 3D point cloud processing [38]. Again, all methods rely on predefined features and thus restrict the flexibility of the generalized convolutions. We will show that a general setup with learnt features leads to better results in practice.

Another line of research aims for further speed-up of the permutohedral lattice. For instance, [9] proposes to encode its operations in a deep neural net.

Learnt Representations. Our embedding network aims to encode guidance data as discriminatively as possible for the task at hand. As such, our work is related to general embedding or metric learning; see [37,44] for an overview.

Image-Adaptive Filtering. We leverage additional properties for our generalized convolutions, which closely relates our approach to image-adaptive filtering methods such as bilateral filtering [41], non-local means [3], and guided image filtering [15]. All of these filters have been included into deep networks, *e.g.* for semantic segmentation [12,14], image processing [45], or video classification [42].

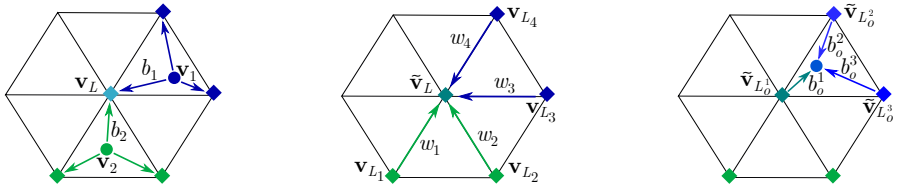
Only few approaches aim to learn guidance features for the filtering step in a general context. Harley *et al.* [14] propose segmentation-aware convolutions, which leverage image-adaptive masks from an embedding network. Object class labels are required for pre-training and large filter kernels increase the risk of overfitting. Deep joint image filtering (DJIF) [31] uses two individual networks to preprocess guidance and data features and subsequently merges the two branches for joint filtering. However, explicit knowledge about the relation between guidance and target data is not leveraged. Gharbi *et al.* [13] reproduce image enhancement operators with locally-affine models and upsample the low-resolution outputs guided by a learnt feature channel. Here, the offline learning of the models puts strong restrictions on the approximated operators. Deep guided filters [45] allow to learn a guidance image but restrict its dimensionality to a one-dimensional signal per output channel.

In contrast to previous work, the semantic lattice is applicable to a large variety of tasks and puts no restrictions on the guidance data. Moreover, the rich feature representations allow us to keep the applied filter kernels small.

3 The Semantic Lattice

To allow for the non-local combination of data, we build on the permutohedral lattice [1] to redefine the notion of *proximity* between the pixels of an image. The permutohedral lattice assumes that each input point is characterized by two properties – *features* and *data*. The feature value $\mathbf{f} \in \mathbb{R}^d$ indicates the location of the respective pixel in the d-dimensional permutohedral space, while the data value $\mathbf{v} \in \mathbb{R}^c$ describes the information stored at this location. In a first step, the data is projected into the lattice grid using the features to determine its position. Convolutions can then be performed in permutohedral space, considering a neighborhood defined by the feature dimensions. For instance, color values can be considered to connect visually similar areas and respect object boundaries [20,24]. This is in contrast to regular convolutions where the spatial location is used as the only feature to determine neighboring pixels. Finally, the convolved output is extracted at certain feature positions, which can but do not have to coincide with the input locations depending on the task at hand.

In the following, we introduce the permutohedral lattice and its properties more formally. We then extend the work in [20,24] to eliminate the usage of fixed, hand-crafted features. In particular, we show how to learn an appropriate feature space based on spatial positions as well as guidance data. As this approach allows to leverage semantically meaningful properties that go beyond the concept of predefined features, we refer to our proposed setup as the *semantic lattice*.



(a) Splat. (b) Convolve. (c) Slice.

Fig. 1. Basic operations in the permutohedral lattice.

3.1 Permutohedral Lattice

Following [1], the permutohedral lattice is defined as the projection of the regular grid $\mathbb{Z}^{d+1}$ onto the hyperplane $\mathcal{H} : \mathbf{h} \cdot \mathbf{1} = 0 \subseteq \mathbb{R}^{d+1}$. The projected grid points thus represent the corners of permutohedral simplices, which split the hyperplane $\mathcal{H}$ into uniform cells. We refer to [1] for further details of the lattice structure.

The operation to read data into the lattice grid is denoted as *splatting*, see Fig. 1a. The feature vector $\mathbf{f}_i$ is used to place an input point $\mathbf{i} = (\mathbf{f}_i, \mathbf{v}_i)$ into the permutohedral lattice. Then, the data value is splat onto the enclosing lattice points according to its barycentric coordinates. The data value of a lattice point L is given as

$$\mathbf{v}_L = \sum_{i \in \mathcal{I}(L)} b_i \cdot \mathbf{v}_i, \tag{1}$$

where b_i denote the barycentric coordinates of input points $\mathcal{I}(L)$ splatting on L.

The *convolution* step is subsequently performed on the permutohedral grid points considering corners in a neighborhood $\mathcal{N}(L)$, *c.f.* Fig. 1b. If a neighboring corner was not set during splatting, its value is assumed to be zero. Using a kernel $\mathbf{W} = (w_1, \ldots, w_N)$, the convolution results in the updated lattice data

$$\tilde{\mathbf{v}}_L = \sum_{L_n \in \mathcal{N}(L)} w_n \cdot \mathbf{v}_{L_n}. \tag{2}$$

Fig. 1c illustrates the final *slicing* operation, which interpolates the data from lattice points to an output pixel $\mathbf{o}$. The value at pixel position $\mathbf{f}_o$ is obtained as

$$\tilde{\mathbf{v}}_o = \sum_{k=1}^{d+1} b_o^k \cdot \tilde{\mathbf{v}}_{L_o^k}, \tag{3}$$

with enclosing simplex corners L_o^k and barycentric coordinates b_o^k, $1 \leq k \leq d+1$.

In [20,24], the permuthedral lattice is integrated into deep learning by providing partial derivatives of the permutohedral operations w.r.t. the input data $\mathbf{v}$ and the kernel $\mathbf{W}$. As such, the original Gaussian kernel [1] is transformed into a general convolution with a flexible filter $\mathbf{W}$ learnt from data. As the filter operation is performed in lattice space, the convolution respects the notion of proximity introduced by the features $\mathbf{f}$ that span the permutohedral lattice.

3.2 Generalized Features in the Semantic Lattice

To define the permutohedral space, [20,24] resort to predefined features, which are usually taken as $\mathbf{f} = (x, y, r, g, b)$. Here, x and y describe the spatial x- and y-coordinates of a pixel, which are concatenated with corresponding RGB values.

Our semantic lattice instead aims to learn feature embeddings from data to leverage the full capacity of the lattice. To that end, we introduce generalized input features $\mathbf{f}(\mathbf{i}; \boldsymbol{\theta}_I)$ that depend on each pixel $\mathbf{i}$ as well as a global set of parameters $\boldsymbol{\theta}_I$. The splatting operation in Eq. (1) then generalizes to

$$\mathbf{v}_L(\boldsymbol{\theta}_I) = \sum_{i \in \mathcal{I}(L; \boldsymbol{\theta}_I)} b_i(\boldsymbol{\theta}_I) \cdot \mathbf{v}_i, \tag{4}$$

since the splatting points as well as the corresponding barycentric coordinates depend on the feature values and thus also on $\boldsymbol{\theta}_I$. Due to the fixed lattice structure, the set of neighbors for the convolution remains unchanged. However, the data value $\mathbf{v}_L$ at each lattice point depends on the inputs that splatted to this exact corner such that we rewrite the convolution in Eq. (2) as

$$\tilde{\mathbf{v}}_L(\boldsymbol{\theta}_I) = \sum_{L_n \in \mathcal{N}(L)} w_n \cdot \mathbf{v}_{L_n}(\boldsymbol{\theta}_I). \tag{5}$$

Finally, the set of lattice points surrounding an output pixel $\mathbf{o}$ and its barycentric coordinates are again dependent on its features $\mathbf{f}(\mathbf{o}, \boldsymbol{\theta}_O)$, which are parametrized by a set $\boldsymbol{\theta}_O$. This definition results in a generalized slicing operation given as

$$\tilde{\mathbf{v}}_o(\boldsymbol{\theta}_I, \boldsymbol{\theta}_O) = \sum_{k=1}^{d+1} b_o^k(\boldsymbol{\theta}_O) \cdot \tilde{\mathbf{v}}_{L_o^k(\boldsymbol{\theta}_O)}(\boldsymbol{\theta}_I). \tag{6}$$

As operations in the lattice require specific computations, common automatic differentiation packages cannot be easily applied. Instead, we rely on customized functions for the above generalized operations as well as their parameter gradients. The derivatives then allow us to apply gradient based optimizers to learn task-specific feature representations $\mathbf{f}(\mathbf{i}; \boldsymbol{\theta}_I)$ and $\mathbf{f}(\mathbf{o}; \boldsymbol{\theta}_O)$ from data.

However, the nested occurance of the parameter sets $\boldsymbol{\theta}_I$ and $\boldsymbol{\theta}_O$ already suggests that learning these generalized features may not be straightforward. Reconsidering the generalized operations in Eqs. (4) – (6), we observe that information between input and output pixels only propagates via a set of lattice corners defined by the neighborhood size of the convolution step. It is thus essential that input and output feature positions are sufficiently close in lattice space when starting the learning process. Otherwise, the loss gradient does not affect the input feature parameters $\boldsymbol{\theta}_I$ and no learning occurs.

To avoid this situation, we propose a specific framework as illustrated in Fig. 2 for the sample task of color upsampling. For given input and output points $\mathbf{p}$, we first generate several *basic features* $\mathbf{f}_B(\mathbf{p}) \in \mathbb{R}^{d'}$, *i.e.* hand-crafted features that we assume to be helpful for the task of interest. In the example case, the spatial location of each pixel and the corresponding grayscale image are chosen as basic

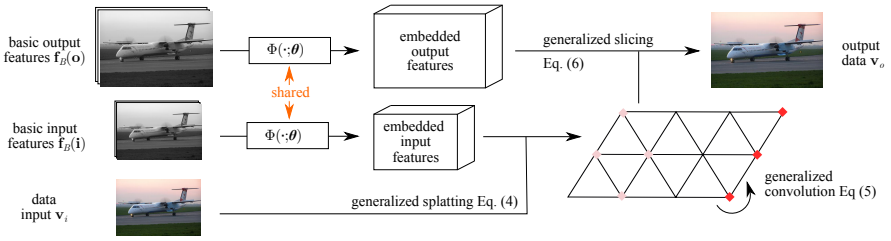


Fig. 2. Visualization of generalized feature learning in the semantic lattice; illustrated for the task of guided color upsampling.

features. Here, the additional grayscale information needs to be available for the input as well as output pixels. We denote it more generally as *guidance data* in the following. Then, a parametric function $\Phi : \mathbb{R}^{d'} \rightarrow \mathbb{R}^d$ is defined, which takes the basic features $\mathbf{f}_B$ as an input and returns a learnt feature embedding $\Phi(\mathbf{f}_B(\mathbf{p}); \boldsymbol{\theta})$ in $\mathbb{R}^d$. The parameter set $\boldsymbol{\theta}$ is shared across input and output points, *i.e.* $\boldsymbol{\theta} = \boldsymbol{\theta}_I = \boldsymbol{\theta}_O$, to ensure the necessary consistency of the feature embedding. We propose to use a multi-layer CNN for the embedding function Φ and refer to this network in the following as *feature* or *embedding network*. The parameter set $\boldsymbol{\theta}$ thus corresponds to the network weights. Embedded features as well as data inputs $\mathbf{v}_i$ are then used for the generalized operations in Eqs. (4) – (6).

As there are no restriction on the basic features, the semantic lattice is able to learn different kinds of non-local operations. Due to the characteristics of the permutohedral lattice, input and output positions are rearranged according to the learnt features and spatially far pixels are connected if they share the same characteristics. With this redefinition of proximity, the number of weight parameters remains limited while the semantic lattice yet operates globally.

4 Training the Semantic Lattice

The training and setup of the semantic lattice requires careful consideration. We detail this in the following and provide an experimental analysis in Sec. 5.2.

4.1 Training Procedure

Feature Scaling. As the size of the permutohedral lattice cells is fixed, a scaling factor applied to the individual features determines the importance of the different dimensions as well as the number of pixels that fall into one lattice cell. These scaling factors thus constitute important hyperparameters. Following [20,24], we determine factors for our basic features via grid search. While with the semantic lattice it is possible to refine these factors in end-to-end training through backpropagation, we found little benefit in our experiments. Hence, we use the scaled features from the grid search as inputs to our embedding network.

Data Centering. If predefined features are used to map to the lattice space, the output is largely invariant to a global translation of the features. In contrast,

we found feature network training to be more stable with zero mean input. We thus subtract the dataset mean from the basic features before feature scaling.

Explicit Spatial Features. As the embedding network combines basic features with various scale factors, we find that a random initialization may lead to a poor initial accuracy. While the feature network is able to recover a reasonable embedding starting from a random initialization, we observe long training times in practice as well as occasional convergence to poor local minima (inferior to the scaled basic features themselves). We find that this is mainly caused by the absence of reliable spatial coordinate features in the initial embedding network. Hence, we do not input the spatial coordinates into the embedding network, and instead explicitly concatenate the scaled spatial features and the learnt feature embedding to jointly define the lattice space for the subsequent convolutions.

Normalization Weights. The number of points per lattice cell can vary considerably, resulting in differing ranges of absolute data values at corner points (Eq. 4). Moreover, the flexible structure of the lattice results in a variable number of non-zero neighbors for the convolution in Eq. (5). For this reason, computations in permutohedral space require a normalization step on the slice result in Eq. (6). We divide by a normalization value, which is obtained by performing all lattice operations with a placeholder input with the same features as the regular input and $\mathbf{v}_i = 1$. This implies that an all-one input remains unchanged by the lattice operations. Note that this normalization becomes invalid as convolution weights turn negative. [20,24] resolve this by introducing a separate set of convolution weights for the normalization. They rely on a fixed Gaussian filter, which reduces the flexibility. In contrast, we explicitly learn separate convolution weights for the normalization step and constrain them to be non-negative.

Learning Rates. If the feature network and permutohedral kernels are learnt simultaneously, individual learning rates are applied to both parameter sets.

4.2 Architecture

We use a CNN with 3×3 filters and leakyReLU activations as our feature embedding network. The non-linearities are omitted after the last convolution to allow for positive and negative features.[1] We experimented with ResNet-like feature networks [16], but observed little benefit. In contrast, we found it essential to add a batch normalization layer [18] at the end of the embedding network, *c.f.* Sec. 5.2. This can be understood as follows: Even in our carefully designed semantic lattice, it is possible that no data is splat to the lattice cells surrounding a specific output location. In such a case, the slice operation returns zero and the corresponding gradient with respect to the output location turns zero as well. Without further gradient signals from such pixels, learning keeps pushing more pixels into this disadvantageous state and the accuracy starts to degrade. Consequently, it is necessary to restrict the output range of the feature network, which batch normalization admits. While other normalization methods

[1] Details of the network architecture are provided in the supplemental material.

Table 1. PSNR for *color upsampling* on the Pascal VOC 2012 Segmentation test set.

	PSNR [dB]		PSNR [dB]
Semantic lattice (*scaled basic features*)	36.55	Nearest neighbors	22.17
Semantic lattice (*learnt kernels*)	36.62	Bicubic upsampling	23.45
Semantic lattice (*learnt embedding*)	36.81	DGF [45]	35.17
Semantic lattice (*both learnt*)	**36.83**	DJIF [31]	23.99

are possible, *e.g.* a simple min-max normalization, they show no clear benefit over batch normalization, which is commonly available in deep learning libraries.

In permutohedral space, we use a single kernel per input channel with a neighborhood of size one, *c.f.* supplemental material. For upsampling tasks as in Sec. 5.1, transitions of the sparse inputs between lattice cells may cause sudden changes of training loss. For this reason, we apply a nearest neighbor upsampling to the low-resolution guidance and input data before feeding them into the feature network and lattice, respectively. The spatial features are adapted to the upsampled input, which spreads the data more evenly over the lattice and leads to more reliable gradients w.r.t. the features.

5 Experiments

5.1 Color Upsampling

Guided upsampling is a common application of image-adaptive filters [2,25,31,45]. Here, guidance data is available at a higher resolution than the data of interest. This is particularly interesting if sensor data is available at different resolutions.

We evaluate our approach on the the task of joint color upsampling in which a grayscale image guides the upsamling of a low-resolution color image. Following [20], we use images of the Pascal VOC Segmentation splits [11] for training, validation, and test from which we removed grayscale images for fair comparison. Bilinear interpolation is used to downsample color and grayscale images by $4\times$.

The semantic lattice is applied to learn the offset between grayscale images and the RGB data. We use spatial coordinates and grayscale values as basic features. The semantic lattice is trained for 100 epochs on random crops of size 200×272 with learning rates of 0.001 and 0.01 for the feature network and permutohedral kernels, respectively. For comparison, we also train the deep guided filter (DGF) [45] in the same setting for 150 epochs using their procedure for image processing tasks. Again, the DGF predicts the offset between RGB and grayscale images as this slightly improves the results. We also compare with Deep Joint Image Filtering (DJIF) [31] by applying their residual network trained for the task of depth upsampling to predict color offsets.

Table 1 summarizes color upsampling results on Pascal VOC Segmentation test. When learning the permutohedral kernels (*learnt kernels*), we observe only a small benefit in comparison to the usage of a Gaussian filter (*scaled basic*

features). In contrast, our learnt feature embedding (*learnt embedding*) leads to a significant improvement, highlighting the importance of using task-specific features. Combining both leads to another (minor) gain. Overall, we outperform the baselines of nearest neighbor and bicubic upsampling as well as related work [31,45] by a large margin. Please see supplemental material for visualizations.

5.2 Validation of Architectural Choices

We now compare different settings for feature learning to validate our proposed lattice setup. Table 2 summarizes results obtained with fixed Gaussian kernels. First, we train an embedding network without batch normalization to evaluate the importance of restricting its output range. We observe a significant drop in PSNR with a result only slightly better than that with scaled basic features. This is due to the fact that input and output locations do not necessarily coincide, which may lead to empty cells without gradients, *c.f.* Sec. 4.2. If the output range of the network is restricted, the number of such pixels can be kept small.

Next, we validate feeding our embedding network with guidance data and concatenating its output with spatial features. We first learn the scale factor of x- and y-coordinates jointly with the embedding. As this yields a negligible improvement over our baseline, we generally do not refine the scale factors. However, note that bigger benefits may be obtained from scale refinement if the initial scale factors are estimated only coarsely. In a second experiment, we apply the embedding network to all features, *i.e.* spatial coordinates and grayscale values. The network learns reasonable features from random initialization, but the PSNR is clearly lower than our baseline despite training 9× longer.

Finally, we evaluate our new normalization approach and learn kernels using scaled basic features. Applying a fixed Gaussian filter rather than a flexible, positive kernel for normalization reduces the PSNR by 0.04dB.

Table 2. Validation of architectural choices for color upsampling on Pascal VOC test.

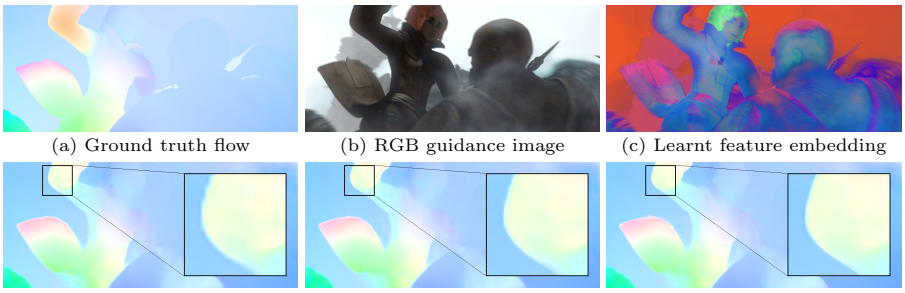
	PSNR [dB]
Baseline (*learnt embedding*)	36.81
W/o batch normalization layer	36.61
Learnt spatial scale factor	**36.82**
Spatial features embedded	36.55
Baseline (*learnt kernels*)	**36.62**
Gaussian normalization	36.58

5.3 Dense Prediction Tasks

We next apply our semantic lattice in deep networks for challenging dense prediction tasks, where networks typically operate on downsampled images and use bilinear upsampling as a last step, *e.g.* [7,39].

Optical Flow. We first consider optical flow for which PWC-Net [39] performs competitively on different benchmarks, *e.g.* [4]. However, the calculated flow looks blurry and boundary details are oversmoothed, see Fig. 3. We attribute this to the non-adaptive upsampling that enlarges the estimated flow by $\sim 4\times$.

To obtain sharper and more detailed flow, we replace the bilinear upsampling with a single convolution in the semantic lattice. As basis, we use the so-called



(a) Ground truth flow (b) RGB guidance image (c) Learnt feature embedding

(d) Output PWC-Net [39] (e) Output DGF [45] (f) Output semantic lattice

Fig. 3. Flow fields and corresponding guidance data for a Sintel sequence.

Table 3. Average-end-point error (AEE) and boundary AEE (bAEE) on our Sintel test split (ours) and the official Sintel test set (off.). Semantic lattice abbreviated to SL.

	clean (ours)		final (ours)		clean (off.)	final (off.)
	AEE	bAEE	AEE	bAEE	AEE	AEE
SL (*scaled basic features*)	1.27	7.84	1.66	8.65	–	–
SL (*learnt embedding*)	1.26	7.50	1.66	8.61	–	–
SL (*both learnt*)	**1.25**	**7.49**	**1.65**	**8.56**	**3.84**	**4.89**
PWC-Net [39]	1.30	8.52	1.67	8.98	3.90	4.90
DGF [45]	1.29	8.31	1.67	8.91	–	–

PWC-Net_ROB model trained on a variety of datasets. For fair comparison, we do not backpropagate into the network itself but only update the parameters of the semantic lattice, since bilinear upsampling cannot benefit from learning. Spatial coordinates and color values of the first image are leveraged as basic features. The high-resolution guidance image is equally used for input and output features. Our setup is trained on the Sintel dataset [4], which we split randomly into 862 training, 80 validation, and 99 test images. We use the average end-point error (AEE) as loss function and train all configurations for 100 epochs on random 281×512 crops. Learning rates are set to $1e-3$ and $1e-7$ for embedding parameters and permutohedral kernels, respectively. We again compare our approach to DGF [45], which we trained for 500 epochs using their setup for computer vision tasks and hyperparameters tuned on validation.

Table 3 shows results on our own test split of Sintel clean and final as well as on the official test images of the benchmark. Our proposed semantic lattice layer leads to a moderate AEE improvement on both sets. This is to be expected as our experimental setup can only refine the flow estimates. However, sharper flow boundaries are clearly visible when considering the results in Fig. 3. As the AEE is known to be insensitive towards boundary accuracy, we also evaluate a boundary average end-point error (bAEE). It focuses on accuracy close to motion discontinuities, which are determined from ground truth flow by applying a threshold to the flow gradient norm, *c.f.* [43]. As the varying motion ranges

require different thresholds [4,43], we follow Weinzaepfel *et al.* [43] and generate multiple masks using values in $\{1, 3, 7, 10\}$. These masks are subsequently dilated with a structuring element of size 3. We finally calculate the bAEE by evaluating flow on boundary regions only and averaging over the different boundary masks. Our proposed approach shows a clear benefit for boundary regions, improving the bAEE much more significantly than DGF [45] on Sintel clean and final.

Semantic Segmentation. We finally consider the task of semantic segmentation and replace the bilinear upsampling of the recent DeepLabv3+ [7] with our semantic lattice. Again, we only update parameters of the semantic lattice and keep the remaining network fixed to an XCeption65 model trained on COCO and Pascal VOC augmented, *c.f.* [7]. Basic features and the setup of our lattice layer remain the same as for optical flow. We train the semantic lattice with random crops of size 200×272 on the training set of Pascal VOC 2012 [11], which we further split into training and validation. The embedding network is trained for 25 epochs with a learning rate of $1e-3$, which we reduce by $10\times$ for the remaining 75 epochs. The learning rate for permutohedral kernels is fixed to $1e-8$. DGF is trained as for optical flow with hyperparameters used in [45].

While the semantic lattice without learnt embedding performs slightly worse than the original implementation, the full semantic lattice and DGF outperform DeepLabv3+. Table 4 summarizes results on Pascal VOC 2012 validation; see supplemental for visualizations. The overall improvement is rather small, which we attribute mainly to DeepLabv3+ being highly engineered, with particular focus on the decoder (unlike the previous DeepLabv3). Nevertheless, image-adaptive filters may benefit further from jointly training with the entire network.

Table 4. Mean intersection over union (mIoU) for *semantic segmentation* on our Pascal VOC 2012 test set.

	mIoU
Semantic lattice (*scaled basic features*)	82.17%
Semantic lattice (*learnt embedding*)	82.24%
Semantic lattice (*both learnt*)	82.25%
DeepLabv3+ [7]	82.20%
DGF [45]	**82.26%**

6 Conclusion

We introduced the semantic lattice layer, a task-specific, generalized convolution. Our approach is built on the permutohedral lattice that rearranges input data according to different features and thus performs non-local operations with small filter kernels. First, we generalized the operations in permutohedral space to feature representations that can be learnt from data. We then showed how rich feature embeddings can be learnt in practice and validated the proposed architecture. When applying the semantic lattice to color upsampling, learning task-specific features showed a clear benefit. Adding the semantic lattice to decoders in deep neural networks for optical flow and semantic segmentation allowed to reduce boundary artifacts and improved the accuracy for both tasks.

References

1. Adams, A., Baek, J., Davis, M.A.: Fast high-dimensional filtering using the permutohedral lattice. Comput. Graph. Forum **29**(2) (2010) 2, 3, 4, 5
2. Barron, J.T., Poole, B.: The fast bilateral solver. In: ECCV (2016) 9
3. Buades, A., Coll, B., Morel, J.M.: A non-local algorithm for image denoising. In: CVPR (2005) 4
4. Butler, D.J., Wulff, J., Stanley, G.B., Black, M.J.: A naturalistic open source movie for optical flow evaluation. In: ECCV (2012) 10, 11, 12, 17
5. Chang, J., Gu, J., Wang, L., Meng, G., Xiang, S., Pan, C.: Structure-aware convolutional neural network 3
6. Chen, L., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Semantic image segmentation with deep convolutional nets and fully connected CRFs. In: ICLR (2015) 3
7. Chen, L., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: ECCV 2018 1, 10, 12, 19
8. Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., Wei, Y.: Deformable convolutional networks. In: ICCV (2017) 2, 3
9. Dai, L., Tang, L., Xie, Y., Tang, J.: Designing by training: Acceleration neural network for fast high-dimensional convolution 3
10. Dolson, J., Baek, J., Plagemann, C., Thrun, S.: Upsampling range data in dynamic environments. In: CVPR (2010) 3
11. Everingham, M., Eslami, S.M.A., Gool, L.V., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL visual object classes challenge: A retrospective. Int. J. Comput. Vision **111**(1), 98–136 (2015) 9, 12, 17
12. Gadde, R., Jampani, V., Kiefel, M., Kappler, D., Gehler, P.: Superpixel convolutional networks using bilateral inceptions. In: ECCV (2016) 3, 4
13. Gharbi, M., Chen, J., Barron, J.T., Hasinoff, S.W., Durand, F.: Deep bilateral learning for real-time image enhancement. SIGGRAPH (2017) 4
14. Harley, A.W., Derpanis, K.G., Kokkinos, I.: Segmentation-aware convolutional networks using local attention masks. In: ICCV (2017) 1, 4
15. He, K., Sun, J., Tang, X.: Guided image filtering. In: ECCV (2010) 2, 4
16. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2017) 1, 8
17. Henriques, J.F., Vedaldi, A.: Warped convolutions: Efficient invariance to spatial transformations. In: ICML (2017) 3
18. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: ICML (2015) 8
19. Jaderberg, M., Simonyan, K., Zisserman, A., Kavukcuoglu, K.: Spatial transformer networks. In: NIPS*2015 2, 3
20. Jampani, V., Kiefel, M., Gehler, P.V.: Learning sparse high dimensional filters: Image filtering, dense CRFs and bilateral neural networks. In: CVPR (2016) 2, 3, 4, 5, 6, 7, 8, 9
21. Jeon, Y., Kim, J.: Active convolution: Learning the shape of convolution for image classification. In: CVPR (2017) 3
22. Jia, X., Brabandere, B.D., Tuytelaars, T., Gool, L.V.: Dynamic filter networks. In: NIPS*2016 2, 3
23. Kang, D., Dhar, D., Chan, A.B.: Incorporating side information by adaptive convolution. In: NIPS*2017 3

24. Kiefel, M., Jampani, V., Gehler, P.V.: Permutohedral lattice CNNs. In: ICLR Workshop Track (2016) 2, 3, 4, 5, 6, 7, 8
25. Kopf, J., Cohen, M.F., Lischinski, D., Uyttendaele, M.: Joint bilateral upsampling. ACM T. Graphics **26**(3) (2007) 9
26. Krähenbühl, P., Koltun, V.: Efficient inference in fully connected CRFs with Gaussian edge potentials. In: NIPS*2011 3
27. Krähenbühl, P., Koltun, V.: Parameter learning and convergent inference for dense random fields. In: ICML (2013) 3
28. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: NIPS*2012 1
29. Li, J., Chen, Y., Cai, L., Davidson, I., Ji, S.: Dense transformer networks. arXiv:1705.08881 [cs.CV] (2017) 3
30. Li, S., Seybold, B., Vorobyov, A., Lei, X., Kuo, C.J.: Unsupervised video object segmentation with motion-based bilateral networks. In: ECCV 2018 3
31. Li, Y., Huang, J.B., Ahuja, N., Yang, M.H.: Joint image filtering with deep convolutional networks. In: IEEE T. Pattern Anal. Mach. Intell. (2019) 4, 9, 10, 17
32. Perazzi, F., Krähenbühl, P., Pritch, Y., Hornung, A.: Saliency filters: Contrast based filtering for salient region detection. In: CVPR (2012) 3
33. Recasens, A., Kellnhofer, P., Stent, S., Matusik, W., Torralba, A.: Learning to zoom: A saliency-based sampling layer for neural networks. In: ECCV 2018 2, 3
34. Russell, C., Yu, R., Agapito, L.: Video pop-up: Monocular 3d reconstruction of dynamic scenes. In: ECCV (2014) 3
35. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2015) 1
36. Singh, B., Najibi, M., Davis, L.S.: Sniper: Efficient multi-scale training 1
37. Song, H.O., Xiang, Y., Jegelka, S., Savarese, S.: Deep metric learning via lifted structured feature embedding. In: CVPR (2016) 3
38. Su, H., Jampani, V., Sun, D., Maji, S., Kalogerakis, E., Yang, M.H., Kautz, J.: Splatnet: Sparse lattice networks for point cloud processing. In: CVPR (2018) 3
39. Sun, D., Yang, X., Liu, M.Y., Kautz, J.: PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In: CVPR (2018) 1, 10, 11, 17, 18
40. Tabernik, D., Kristan, M., Leonardis, A.: Spatially-adaptive filter units for deep neural networks. In: CVPR (2018) 2, 3
41. Tomasi, C., Manduchi, R.: Bilateral filtering for gray and color images. In: ICCV (1998) 2, 4
42. Wang, X., Girshick, R., Gupta, A., He, K.: Non-local neural networks. In: CVPR (2018) 4
43. Weinzaepfel, P., Revaud, J., Harchaoui, Z., Schmid, C.: Learning to detect motion boundaries. In: CVPR (2015) 11, 12
44. Wu, C.Y., Manmatha, R., Smola, A.J., Krähenbühl, P.: Sampling matters in deep embedding learning. In: ICCV (2017) 3
45. Wu, H., Zheng, S., Zhang, J., Huang, K.: Fast end-to-end trainable guided filter. In: CVPR (2018) 4, 9, 10, 11, 12, 17, 18
46. Wu, J., Li, D., Yang, Y., Bajaj, C., Ji, X.: Dynamic filtering with large sampling field for convnets. In: ECCV 2018 1, 2, 3
47. Yu, F., Koltun, V.: Multi-scale context aggregation by dilated convolutions. In: ICLR (2016) 3
48. Zhang, Z., Fidler, S., Urtasun, R.: Instance-level segmentation for autonomous driving with deep densely connected MRFs. In: CVPR (2016) 3

Learning Task-Specific Generalized Convolutions in the Permutohedral Lattice

– Supplemental Material –

Anne S. Wannenwetsch[1⋆][0000−0002−7016−3820],
Martin Kiefel[2][0000−0001−9432−5428],
Peter V. Gehler[2][0000−0002−5812−4052], Stefan Roth[1][0000−0001−9002−9832]

[1] TU Darmstadt, Germany [2] Amazon, Germany

A Implementation Details

We implement forward and backward passes of our semantic lattice layer in C++ and CUDA and wrap them in MXNet [50]. To derive the necessary gradients, we manually apply the principles of reverse automatic differentiation, *c.f.* [49]. Code is available at https://github.com/visinf/semantic_lattice.

The grid search for scale parameters is performed on full-size training images using the respective evaluation metrics. If the basic features include color channels, we only determine a single scale factor across all color channels to keep the grid search feasible. Table 5 summarizes scale factors λ_S and λ_I used in our experiments for spatial and intensity features, respectively. Remaining hyperparameters, *e.g.* the learning rates, are chosen using the validation set.

We use a default batchsize of 16 and average over multiple batches if memory permits only fewer samples. In rare cases, we observe large gradients in training the embedding network, which we counter with gradient clipping (at 0.1). The feature network is randomly initialized with a default Xavier initialization [51] while the filter weights in permutohedral space are initialized as Gaussian kernels. The non-negativity of normalization filters is ensured by learning in the log-domain.

Table 5. Feature scale factors.

Task	λ_S	λ_I
Color upsampling, 2×	1.25	5.0
Color upsampling, 4×	0.65	5.0
Color upsampling, 8×	0.20	7.5
Optical flow upsampling	0.15	70.0
Semantic segmentation ups.	0.15	25.0

B Network architectures

The architecture of our embedding networks is described in Table 6. The parameter $\tilde{d}$ denotes the number of embedded features. We set $\tilde{d} = 1$ for color upsampling and $\tilde{d} = 3$ for dense prediction tasks. Since embedded features are concatenated with two-dimensional spatial coordinates, the semantic lattice receives features of dimensionality $d = \tilde{d} + 2$. All convolution layers use a stride

⋆ This project was mainly done during an internship at Amazon, Germany.

Table 6. Architecture of embedding networks.

Layer	1	2	3
Kernel size	3	3	3
Channels	15	15	$\tilde{d}$
Groups	1	1	1
Bias	✓	✓	✓
Non-linearity	✓	✓	✗
Batch normalization	✗	✗	✓

Table 7. Architecture in perm. space.

Layer	1
Neighborhood size	3
Channels	c
Groups	c
Bias	✗
Non-linearity	✗
Batch normalization	✗

of one and zero padding to preserve the input size. We apply leakyReLU activations with slope coefficients $\alpha = 0.2$ as non-linearities. For batch normalization, we use default parameters and apply the transformation to the channel axis.

Table 7 specifies the setup in permutohedral space used for our experiments. The number of outputs c depends on the specific task and is determined by the dimensionality of the input data. As such, we have $c = 3$ for color upsampling, $c = 2$ for optical flow and $c = 21$ for semantic segmentation. All convolutions in the permutohedral lattice are performed per channel, *i.e.* we set the number of groups to c and learn a separate convolution kernel for each data dimension.

C Additional Experiments Color Upsampling

We start with a small ablation study performed on the task of color upsampling. In a first experiment, we keep the permutohedral weights fixed and increase the number of embedded features from one to two. However, the additional feature does not lead to improved results but we even observe a small drop in PSNR on the test split from 36.81 to 36.79. As a larger amount of feature dimensions leads to an increased runtime, we choose the dimensionality of the feature embedding to equal the number of basic features provided to the embedding network.

In a second setup, we evaluate the performance of the semantic lattice with larger kernels in permutohedral space. Therefore, we learn permutohedral weights of neighborhood size two with fixed basic features. We obtain a PSNR of 36.64 on the test set in comparison to a PSNR of 36.62 with a neighborhood size of one. Again, we choose the smaller neighborhood size due to improved runtime.

Finally, we evaluate color upsampling for additional upsampling factors in Table 8. We again observe that the semantic lattice performs best if the feature embedding as well as the kernels are learnt. Interestingly, the benefit gets more significant as the difficulty of the task increases, *i.e.* for larger upsampling factors. As before, we clearly outperform baselines and related work.

D Additional Visualizations

We provide visualizations for the different tasks discussed in the paper.

Color upsampling. Visualizations of the 4× color upsampling task are given in Fig. 4. The fully learnt semantic lattice is clearly superior to the deep guided

Table 8. Evaluation of additional upsampling factors for the task of *color upsampling* on the Pascal VOC 2012 Segmentation test set. *No pretrained network available.

	PSNR [dB], ×2	PSNR [dB], ×8
Semantic lattice (*scaled basic features*)	40.05	33.93
Semantic lattice (*learnt kernels*)	40.10	34.06
Semantic lattice (*learnt embedding*)	40.20	34.23
Semantic lattice (*both learnt*)	**40.22**	**34.33**
Nearest neighbors	25.91	19.46
Bicubic upsampling	27.23	20.73
DGF [45]	37.80	32.97
DJIF [31]	–*	20.57

filter as it correctly reconstructs small and thin color regions, *e.g.* the green and blue strips on the white train. Moreover, the lattice shows considerably fewer color bleeding artifacts, *e.g.* at the red parts of the bars in the first row.

Dense prediction tasks. Fig. 5 shows visualizations for ground truth and predicted optical flow on several sequences of the Sintel dataset [4]. As already discussed in the main paper, the semantic lattice leads to less blurry flow fields in comparison to the original PWC-Net [39]. Additionally, it allows to recover fine details at motion boundaries, *e.g.* the structure of the hair in the last row.

In Fig. 6, examples for segmentations on Pascal VOC 2012 [11] are provided. Considering the results of DeepLabv3+, one observes that the segmentation masks frequently exceed the borders of detected objects. Applying the semantic lattice with learnt embedding and kernels allows us to reduce such margins. As such, the obtained segmentations align better with the underlying objects.

References

49. Baydin, A.G., Pearlmutter, B.A., Radul, A.A., Siskind, J.M.: Automatic differentiation in machine learning: a survey. J. Mach. Learn. Res. **18**, 1–43 (2018)
50. Chen, T., Li, M., Li, Y., Lin, M., Wang, N., Wang, M., Xiao, T., Xu, B., Zhang, C., Zhang, Z.: MXNet: A flexible and efficient machine learning library for heterogeneous distributed systems. arXiv:1512.01274 [cs.DC] (2015)
51. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: AISTATS (2010)

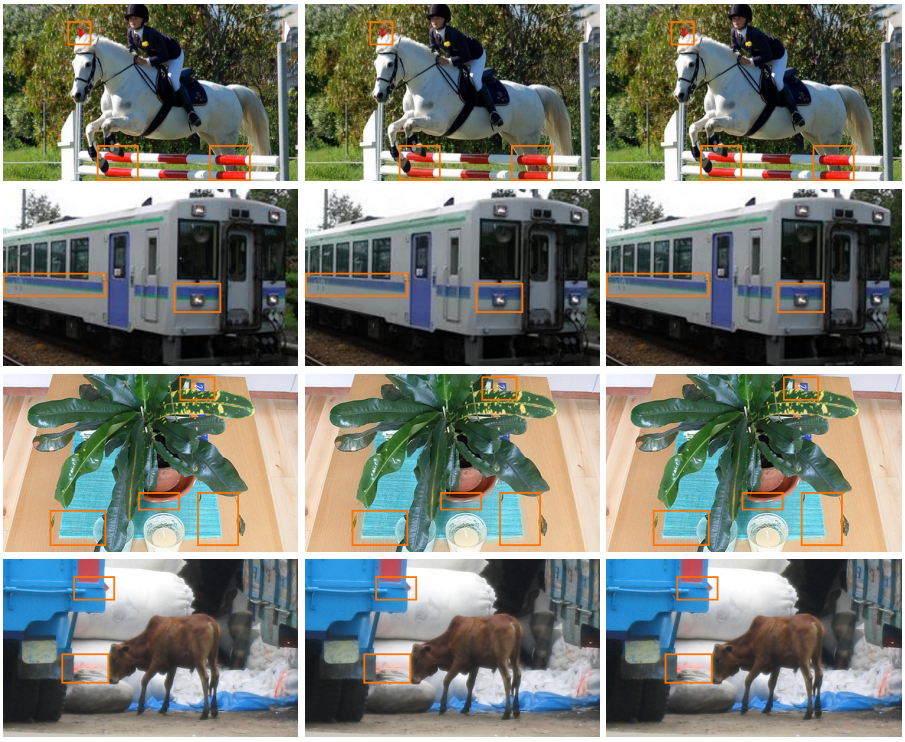


Fig. 4. Left to right: Crops of ground truth, outputs of DGF [45], and outputs of the semantic lattice for 4× color upsampling on Pascal VOC 2012. Best viewed on screen.

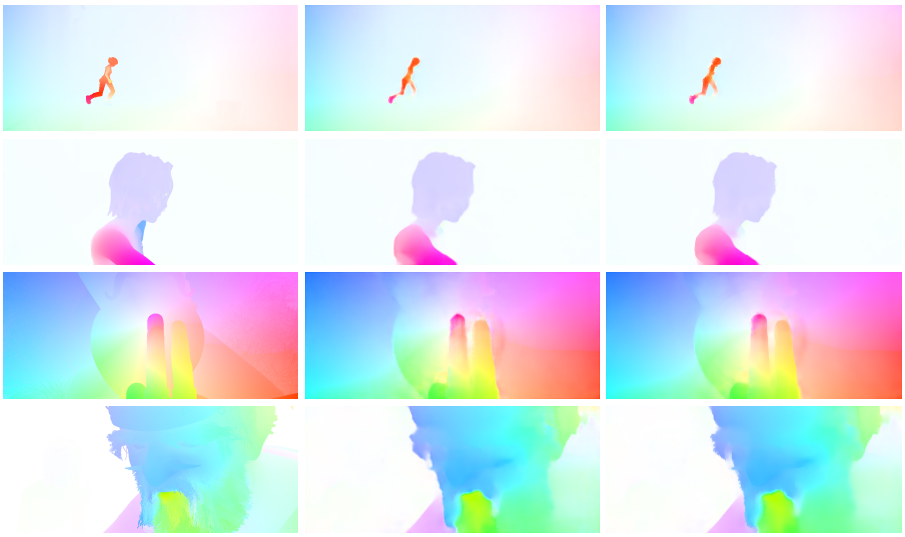


Fig. 5. Left to right: Ground truth, outputs of PWC-Net [39], and outputs of the fully learnt semantic lattice on different Sintel sequences. Best viewed on screen.

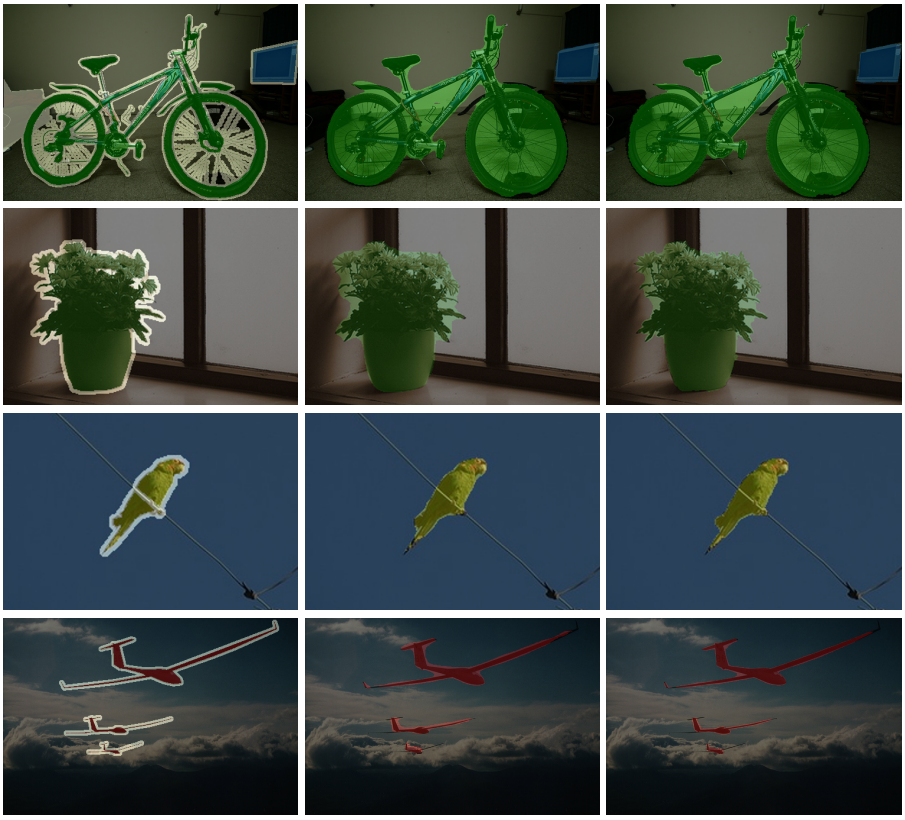


Fig. 6. Left to right: Crops of ground truth, outputs of DeepLabv3+ [7], and outputs of the semantic lattice for segmentation on Pascal VOC 2012. Best viewed on screen.

A.4 PROBABILISTIC PIXEL-ADAPTIVE REFINEMENT NETWORKS

Anne S. Wannenwetsch and Stefan Roth
2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition

Abstract

Encoder-decoder networks have found widespread use in various dense prediction tasks. However, the strong reduction of spatial resolution in the encoder leads to a loss of location information as well as boundary artifacts. To address this, image-adaptive post-processing methods have shown beneficial by leveraging the high-resolution input image(s) as guidance data. We extend such approaches by considering an important orthogonal source of information: the network's confidence in its own predictions. We introduce *probabilistic pixel-adaptive convolutions (PPACs)*, which not only depend on image guidance data for filtering, but also respect the reliability of per-pixel predictions. As such, PPACs allow for image-adaptive smoothing and simultaneously propagating pixels of high confidence into less reliable regions, while respecting object boundaries. We demonstrate their utility in refinement networks for optical flow and semantic segmentation, where PPACs lead to a clear reduction in boundary artifacts. Moreover, our proposed refinement step is able to substantially improve the accuracy on various widely used benchmarks.

Probabilistic Pixel-Adaptive Refinement Networks

Anne S. Wannenwetsch[1,2*] Stefan Roth[2]
[1]Amazon, Germany [2] TU Darmstadt, Germany

Abstract

Encoder-decoder networks have found widespread use in various dense prediction tasks. However, the strong reduction of spatial resolution in the encoder leads to a loss of location information as well as boundary artifacts. To address this, image-adaptive post-processing methods have shown beneficial by leveraging the high-resolution input image(s) as guidance data. We extend such approaches by considering an important orthogonal source of information: the network's confidence in its own predictions. We introduce probabilistic pixel-adaptive convolutions (PPACs)*, which not only depend on image guidance data for filtering, but also respect the reliability of per-pixel predictions. As such, PPACs allow for image-adaptive smoothing and simultaneously propagating pixels of high confidence into less reliable regions, while respecting object boundaries. We demonstrate their utility in refinement networks for optical flow and semantic segmentation, where PPACs lead to a clear reduction in boundary artifacts. Moreover, our proposed refinement step is able to substantially improve the accuracy on various widely used benchmarks.*

1. Introduction

Convolutional neural networks (CNNs) have become a standard tool in computer vision. Especially in dense prediction tasks [7, 41, 49, 60], encoder-decoder or pyramid-structured CNNs are a common choice. While originating in unsupervised learning [21], such architectures have become popular also in supervised settings. The encoder builds a powerful feature representation, reducing the spatial resolution of the inputs to aggregate global information [41]. The decoder takes the feature representation from the bottleneck, enlarges its size, and transforms it into the desired output, *e.g*. a segmentation map or optical flow field.

While downsampling in the encoder increases the receptive field and allows to deal with large image sizes, it also leads to a drastic loss in spatial resolution. As such, valuable location information is lost and boundary artifacts can arise [18, 41], *e.g*. segmentation maps that are misaligned w.r.t. the underlying objects. Moreover, the decoder typically yields low-resolution outputs and simple components are used to upscale predictions to the input size. This often results in blurry outputs since estimates of different objects are combined, *e.g*. motion from background and foreground objects is mixed as can be seen in Fig. 1.

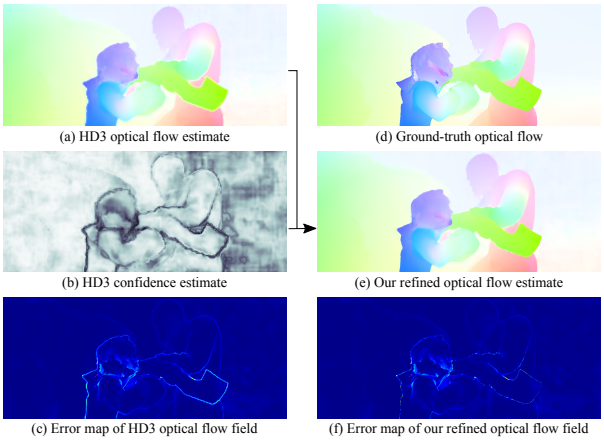


Figure 1. Our PPAC refinement method leverages the close relationship between estimated confidences and prediction errors to refine and improve the prediction itself, here the optical flow field.

Several approaches have been proposed to reduce these disadvantages, *e.g*. skip connections [41, 49] or densely connected blocks [24, 32]. Additionally, different types of generalized convolutions, taking into account a high-resolution RGB guidance image, have shown beneficial as part of the decoder [25, 26] or in a separate upsampling and/or refinement step [31, 48, 56, 58]. Many of the above approaches require a large number of additional parameters, are computationally expensive, or restrictive w.r.t. to the filtering method and the applicable guidance data. Most recently, pixel-adaptive convolutions (PACs) were introduced by Su *et al*. [51]. PACs combine spatially-invariant convolution weights with a content-adaptive kernel that depends on guidance data. In [51], PACs are shown to yield state-of-the-art results in joint upsampling tasks.

In this paper, we argue that we can leverage another source of information for refinement that is complementary

*This work was done at TU Darmstadt prior to Anne S. Wannenwetsch joining Amazon.

to the input image: *the uncertainty of each pixel's estimate*. For dense classification networks, this quantity is generally provided implicitly. For instance, segmentation networks usually output log-probabilities of the classes, which are passed through an $\arg\max$ operation at test time. Even though these uncertainties might not be well calibrated [16], we argue that they still contain valuable information for refining the predictions. Beyond classification, explicit uncertainty estimates are also gaining increased attention for dense regression problems such as geometry [47] or motion estimation [12, 27, 60]. They are, for example, helpful for applications in which the reliability of network estimates is crucial, *e.g*. in autonomous driving. Here, we show that we can also leverage them to refine the regression output itself.

Fig. 1 shows an optical flow field estimated by the probabilistic HD3 method [60], as well as the corresponding confidence map and endpoint error per pixel. We observe that regions of high uncertainty (*b*, dark gray) correspond quite well with large errors (*c*, dark red). When applying post-processing to the network output, it seems desirable to take the available pixel uncertainty into account. As such, only reliable pixels should be spatially propagated while uncertain pixels can be replaced. To allow for probability-aware[1] filtering, we propose *probabilistic pixel-adaptive convolutions (PPACs),* and therefore extend the adaptive convolution operation of [51]. The kernels of PPACs and thus the filtering output vary dependent on two properties: guidance data, *e.g*. the input image, as well as a probability map estimated by the deep network, either inherently or explicitly.

This paper focuses on the application of PPACs for the refinement of outputs from dense prediction networks, illustrated with the tasks of optical flow estimation and semantic segmentation. Therefore, we introduce a *PPAC refinement network*, which leverages RGB guidance data and probabilities for several content- and probability-adaptive convolutions. For both tasks, PPACs not only allow to improve estimates at boundaries but also remove outliers of low reliability. As shown in Fig. 1, blurry edges in the flow field *(a)* are transformed into crisp boundaries and the overall prediction is smoothed *(e)*. Along with the visual improvements, PPAC refinement leads to a clear accuracy gain in optical flow and semantic segmentation. For instance, PPACs substantially improve state-of-the-art HD3 [60] optical flow estimates on the widely used KITTI 2012 and 2015 datasets. Our proposed PPAC-HD3 method ranks 1$^{\text{st}}$ among published optical flow approaches[2] on both benchmarks, improving the outlier rate by $\sim$11.1% and $\sim$7.5% over the underlying baseline method.

[1]By *probability* we refer to a measure that approximates or summarizes the marginal posterior over the network's estimates, *e.g*. by taking the marginal posterior of the chosen prediction value. To ease readability, the terms probability, confidence, (un)certainty, and reliability will be used interchangeably in the paper.

[2]All rankings at the time of publication.

2. Related Work

Probabilistic deep networks. Combining probabilistic approaches with deep networks is an active field of research, which is pursued to cope with model and/or input uncertainty [35]. As such, we can only provide a rough summary and refer to the cited references for a broader overview.

Bayesian neural networks [5, 15, 20, 37, 42, 57] often learn parametric distributions over the network weights to capture model uncertainty. The predictive distribution over the outputs is obtained by taking an expectation over the weights through approximate inference [5, 20]. However, Bayesian neural networks introduce many additional parameters and are not always easy to handle in practice [39].

Sampling-based approaches, *e.g*. [2, 11, 35], are often simpler to apply and include a random component, such as dropout, in the network structure. At test time, the predictive uncertainty is computed as Monte Carlo estimates from several network passes. Similarly, an ensemble of networks can be trained and combined at test time [23, 39]. A major drawback of both avenues is the increased runtime as they require multiple forward passes.

Another line of research uses deep networks to output the parameters of an assumed predictive distribution, either directly [35, 46] or by propagation of input uncertainty [12, 54]. There, the difficulty is to find a parametric distribution that is sufficiently easy to handle in practice and appropriately describes the quantity of interest.

Beyond such general purpose probabilistic treatments, probabilistic networks have also been developed in the context of specific vision problems. Yin *et al*. [60] propose a method to aggregate correspondence uncertainty in the context of optical flow and stereo matching through various spatial scales. In [27], a multi-hypothesis network for optical flow estimation is trained to output an ensemble at once. Novotny *et al*. [47] use uncertainty estimates obtained with probabilistic losses to predict the reliability of descriptors.

Content-adaptive convolutions. One category of content-adaptive convolutions adjusts the sampling location of neighbor pixels [8, 33]. Deformable convolutions [8] predict data-dependent offsets to determine at which locations neighboring pixels should be sampled for a spatially-invariant convolution. Another line of research adjusts the convolution weights [34, 59] of standard convolutions. Dynamic filter networks [34] use a subnetwork to predict location-specific weight kernels, which have already shown benefits for optical flow estimation, *e.g*. [25, 26]. A common drawback is the significant amount of additional parameters, which increases the risk of overfitting – especially if only a limited amount of training data is available.

Several works, therefore, approach content-adaptive convolutions in a more constrained setting. Spatial transformers [30] as well as CARAFE [55] rearrange features

in a content-adaptive way with a global or local transformation before performing the convolution itself. However, they remain restricted to a 2D grid structure. [31, 56] perform image-adaptive convolutions with predefined or learned features in the high-dimensional permutohedral lattice [1]. Such convolutions are computationally expensive and thus not suitable for fast processing. [18] incorporates semantics by learning input-dependent attention masks but requires object classes for (pre-)training. Deep guided filters [58] extend the classical guided filter [19] to learned guidance data. In [48], the parameters for spatially-variant linear representation models of the guided filter are learned with a CNN. In both approaches only 1D guidance data can be used for each output channel. Deep joint image filtering [40] applies standard convolutions to the concatenation of pre-processed guidance data and estimates, but misses explicit knowledge on the relation of both components.

We base our approach on pixel-adaptive convolutions (PACs) [51]. Here, the convolution kernels are split into a fixed weight as well as a location-specific component that depends upon a feature embedding learned from guidance data. PACs have a small computational overhead and are easy to train in practice. Nevertheless, like most adaptive filtering approaches, PACs do not allow to explicitly leverage knowledge about the reliability of filter inputs. While such information can be included as guidance data for the content-adaptive part of the weight kernel, our explicit probabilistic formulation leads to a significant performance gain.

Probabilistic joint filtering. There are only few filtering approaches that jointly consider guidance data as well as probabilities. Different filtering methods have been applied to refine semantic segmentations [50], optical flow [43, 53], and especially depth [17, 38, 45]. However, these approaches are task-specific, tailored to certain filtering methods, and/or rely on time-intensive iterative approaches. [14] replaces unreliable pixels using a network that takes uncertainties and guidance data as input, but uncertainties are not explicitly leveraged to improve estimates. The Fast Bilateral Solver [4] and the Domain Transform Solver [3] allow to perform fast, edge-aware optimization on different estimates. They leverage uncertainty by requiring a closer connection between inputs and outputs for more reliable pixels. As these approaches optimize a predefined objective, their flexibility is restricted. Moreover, [3, 4] cannot backpropagate into the features used to determine the pixel similarity. Closest to ours with regard to probabilistic joint filtering are [22, 29], where confidences are used to extend the bilateral and the guided filter, respectively, by weighing a pixel's importance with its confidence. However, both methods rely on predefined features for pixel similarity, hand-crafted reliability measures, and fixed filter kernels.

In comparison to previous work, our approach is very general as pixel feature embeddings, the filter weights, as well as a pre-processing of the probabilities are learned from data. Moreover, the proposed approach is fast and easily integrable into different task-specific neural networks.

3. Probabilistic Pixel-Adaptive Convolutions

We begin by presenting pixel-adaptive convolutions (PACs) as introduced in [51]. We then propose an advanced normalization approach for pixel-adaptive convolutions and finally extend PACs to allow for probabilistic filtering.

3.1. Pixel-adaptive convolutions

Assume, we aim to perform a convolution with neighborhood size s that transforms features $\mathbf{v} \in \mathbb{R}^d$ into features $\tilde{\mathbf{v}} \in \mathbb{R}^{d'}$. We denote the corresponding convolution weights as tensor $\mathbf{W} \in \mathbb{R}^{d' \times d \times s \times s}$ and the bias term as $\mathbf{b} \in \mathbb{R}^{d'}$. Following the notation of [51], the output of a standard convolution at pixel i is then given as

$$\tilde{\mathbf{v}}_i = \sum_{j \in \mathcal{N}(i)} \mathbf{W}\left[\mathbf{p}_i - \mathbf{p}_j\right] \mathbf{v}_j + \mathbf{b}, \tag{1}$$

where $\mathcal{N}(i)$ denotes the $s \times s$ neighborhood of the pixel. The vectors $\mathbf{p}_i$ and $\mathbf{p}_j$ represent the 2D pixel positions. $\mathbf{W}\left[\mathbf{p}_i - \mathbf{p}_j\right] \in \mathbb{R}^{d' \times d}$ corresponds to the 2D slice from weight tensor $\mathbf{W}$, evaluated at position $\mathbf{p}_i - \mathbf{p}_j$.

PACs generalize such spatially-invariant convolutions by augmenting the convolution weight with an additional location-adaptive component $K\big(\mathbf{f}_i, \mathbf{f}_j\big)$. In [51], the vectors $\mathbf{f}_i$ and $\mathbf{f}_j$ are denoted as *pixel features* and characterize the pixels i and j, respectively. For instance, one could use the RGB components of a guidance image as feature $\mathbf{f}_{(\cdot)}$, as is done in the bilateral filter [52]. However, more advanced features learned from data have shown to be advantageous [51]. The function $K(\mathbf{f}_i, \mathbf{f}_j) = K(\mathbf{f}_i - \mathbf{f}_j)$ is a (fixed) kernel, which evaluates the difference between $\mathbf{f}_i$ and $\mathbf{f}_j$. If pixels i and j show similar characteristics, $K(\mathbf{f}_i, \mathbf{f}_j)$ weighs the corresponding values $\mathbf{v}_j$ more than the ones of a more deviating pixel. Various choices for $K(\cdot, \cdot)$ are possible; we will apply a Gaussian RBF kernel in the following, *i.e.*

$$K\big(\mathbf{f}_i, \mathbf{f}_j\big) = e^{-\frac{1}{2}(\mathbf{f}_i - \mathbf{f}_j)^{\mathrm{T}}(\mathbf{f}_i - \mathbf{f}_j)}. \tag{2}$$

The PAC convolution [51] is then defined as

$$\tilde{\mathbf{v}}_i = \sum_{j \in \mathcal{N}(i)} K\big(\mathbf{f}_i, \mathbf{f}_j\big) \cdot \mathbf{W}\left[\mathbf{p}_i - \mathbf{p}_j\right] \mathbf{v}_j + \mathbf{b}. \tag{3}$$

The same feature kernel K is used for all input channels, while the weight $\mathbf{W}$ differs dependent on the spatial location within the mask and for each feature channel.

3.2. Advanced normalization step

PACs are a powerful tool for deep dense prediction architectures, but they are not without challenges. One major issues is the fact that the number of closely related pixels, *i.e.* pixels j that show a high value of $K(\mathbf{f}_i, \mathbf{f}_j)$, varies

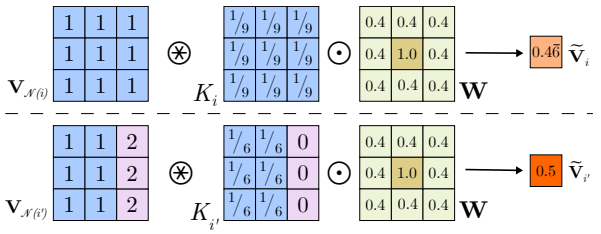


Figure 2. PAC normalization w.r.t. kernel only (see text).

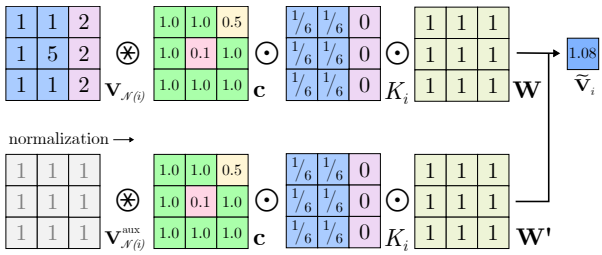


Figure 3. Illustration of our proposed probabilistic PAC approach. The weighting of estimates according to their pixel feature similarity K as well as their confidence $\mathbf{c}$ allows to remove outliers.

across the image. This is natural and even desirable, since only a restricted neighborhood region should be taken into account at object boundaries or similar. Nevertheless, applying PACs in different neighborhoods should lead to results with the same output magnitude as long as the input variables are equal. Otherwise, learning of convolution weights might be difficult since the output values are inevitably smaller at boundaries or in highly structured areas.

Basic scheme. The implementation of PACs provides an option to normalize kernels such that $\sum_j K(\mathbf{f}_i, \mathbf{f}_j) = 1$ for all pixels i. However, we argue that such a *kernel normalization* is not sufficient. Consider the illustration of two pixels i and i' and their neighborhoods $\mathcal{N}(i)$ and $\mathcal{N}(i')$ in Fig. 2. For simplicity, we assume equal input values for pixels of the same object. Pixel i is part of a homogeneous area and the kernel function leads to an equal distribution of the kernel weights. In contrast, the neighborhood of pixel i' contains also elements from a different object. Here, the kernel weights are only distributed over the elements from the same object as pixel i'. Even though both kernels sum to one, their convolution with an exemplary weight $\mathbf{W}$ leads to clearly different results. As mentioned above, this complicates the learning of PACs.

Our advanced scheme. We address this issue with an advanced normalization scheme. To that end, we adapt the normalization from [56] to the context of PACs. An auxiliary array $\mathbf{v}^{aux} = \mathbf{1}$ with constant value 1 is defined and passed through the filtering step in Eq. (3):

$$\tilde{\mathbf{v}}_i^{aux} = \sum_{j \in \mathcal{N}(i)} K\big(\mathbf{f}_i, \mathbf{f}_j\big) \cdot \mathbf{W}\,[\mathbf{p}_i - \mathbf{p}_j] \cdot \mathbf{1}. \qquad (4)$$

In slight abuse of notation, we now denote by $\tilde{\mathbf{v}}_i$ the PAC output in Eq. (3) *before* adding the bias term. Then, the normalized convolution output $\tilde{\mathbf{v}}_{i,\text{norm}}$ is given by

$$\tilde{\mathbf{v}}_{i,\text{norm}} = \tilde{\mathbf{v}}_i / \tilde{\mathbf{v}}_i^{aux} + \mathbf{b}. \qquad (5)$$

With the novel normalization, not only the number of similar pixels is taken into account but also their weighting by $\mathbf{W}$. However, the normalization becomes invalid as soon as the weight $\mathbf{W}$ becomes negative [56]. We thus follow [56] to introduce an additional *normalization weight* $\mathbf{W}'$, which replaces $\mathbf{W}$ in the normalization convolution in Eq. (4). $\mathbf{W}'$ is ensured to remain positive and set to the same initialization as $\mathbf{W}$. During training, we independently update $\mathbf{W}$ as well as $\mathbf{W}'$ using regular gradient-based optimization and thus omit to explicitly enforce their similarity.

Reconsidering the example in Fig. 2, both output values remain close when using our advanced normalization with the appropriate $\mathbf{W}' \approx \mathbf{W}$. We will show in Sec. 6.1 that the proposed scheme leads to superior results in practice.

3.3. Probabilistic pixel-adaptive convolutions

While the definition of PACs allows for more advanced filtering than a standard convolution, the approach is still restricted. The kernel function $K(\cdot, \cdot)$ in Eq. (2) only takes differences of pixel features as input, thus excluding properties that cannot be reasonably expressed as such. Consider, for instance, that we have a per-pixel probability alongside the estimate. In this case, it seems beneficial to consider such information during filtering to improve unreliable estimates. As $K(\cdot, \cdot)$ rewards similar pixel features, neighbors with a similar level of reliability are more closely connected. However, this seems counterintuitive, given that *the values of reliable pixels should particularly propagate to neighbors with a very different,* i.e. *low, confidence*.

Therefore, we extend PACs such that we are able to perform convolutions that also take unary properties – especially probabilities – into account. Let c_j describe the confidence assigned to a certain pixel location j. For consistency, we assume that only one confidence estimate is given per spatial location.[3] Similar to [29], we then propose to define a *probabilistic pixel-adaptive convolution (PPAC)* as:

$$\tilde{\mathbf{v}}_i = \sum_{j \in \mathcal{N}(i)} c_j \cdot K\big(\mathbf{f}_i, \mathbf{f}_j\big) \cdot \mathbf{W}\,[\mathbf{p}_i - \mathbf{p}_j]\,\mathbf{v}_j + \mathbf{b}. \qquad (6)$$

Here, each pixel value is not only weighted by its distance to the center pixel but also with its individual confidence.

To illustrate our proposed approach, consider Fig. 3: An outlier pixel is surrounded by more reliable estimates,

[3] An extension to individual confidences per channel is straightforward.

which belong to the same and a different object. For simplicity, we assume that $\mathbf{W}$ performs an averaging operation. If the pixel confidence was not taken into account, the outlier value would spread to the surrounding pixels due to its higher magnitude. In contrast, the proposed PPAC allows to propagate the reliable pixel values from the same object and thus almost completely replaces the outlier in the center.

The normalization as proposed in Sec. 3.2 can be easily extended to PPACs. To that end, $\tilde{\mathbf{v}}_i^{aux}$ is obtained as

$$\tilde{\mathbf{v}}_i^{aux} = \sum_{j \in \mathcal{N}(i)} c_j \cdot K\left(\mathbf{f}_i, \mathbf{f}_j\right) \cdot \mathbf{W}\left[\mathbf{p}_i - \mathbf{p}_j\right] \cdot \mathbf{1}. \quad (7)$$

The normalization step is performed as before and the PPAC outputs are divided per pixel by $\tilde{\mathbf{v}}_i^{aux}$ before the bias (Eq. 5)

4. Refinement Networks with PPACs

Deep dense predictors often output predictions at a scale lower than the input resolution to save time and parameters, *e.g.* [7, 60]. The low-resolution estimates are then upscaled by simple methods such as bilinear interpolation. In [51, 56, 58], image-adaptive convolutions have proven very helpful in this upsampling step. Following that, we propose a *PPAC refinement network*, which takes image and reliability data into account to upscale and refine network outputs.

A straightforward approach is to upscale results with transposed PPACs. However, we found that this can lead to difficulties, especially for optical flow. This is due to the fact that flow networks often assume the input sizes to be divisible by a certain power of 2, *e.g.* 2^6 [60]. As this is mostly not the case, *e.g.* for the Sintel benchmark [6], input images are resized and the output flow is afterwards rescaled with a non-integer factor. To apply transposed convolutions, which can only upscale by integers, one has to pad the inputs and crop the output after upscaling. We observed that this leads to severe artifacts, which clearly reduce the accuracy.

Instead, we propose to first upscale the estimates by the default method of the original network. A lightweight network with PPACs is then applied at full resolution. As we only use a small number of PPACs, the computational expense of the approach remains low and the prediction accuracy does not decrease due to padding artifacts or similar.

Our proposed refinement networks consist of three branches as illustrated in Fig. 4. In addition to the upscaled estimates, the network takes corresponding probability data, *e.g.* a full marginal posterior per pixel or other probability measures, and the high-resolution images as inputs. The first subnetwork transforms the probability data into scalar confidence values for the individual PPACs. The second branch processes the guidance images to generate meaningful pixel features. Both intermediate outputs as well as the underlying network predictions are then fed into the PPACs of the combination branch to create a refined estimate.

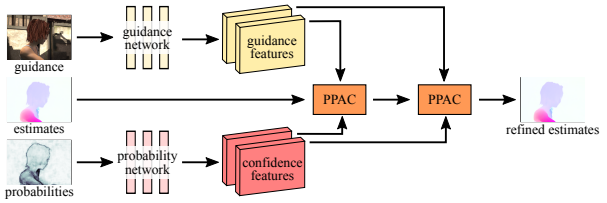


Figure 4. Exemplary architecture of our proposed PPAC refinement networks.

5. Implementation

5.1. Network architecture

To illustrate their capabilities, we experiment with PPAC refinement networks for the tasks of optical flow estimation and semantic segmentation. All networks are fully convolutional and include two consecutive PPACs. We use log values as inputs to the probability branch and add a sigmoid function at its end for normalization. Please refer to the supplement for a detailed description of the networks' setup.

Here, we only highlight architectural choices that differ significantly from the ones in [51]. First, we found no benefit from increasing the number of channels in the combination branch. Moreover, using group convolutions with the number of groups being equal to the number of inputs did not decrease the performance, but significantly lowers the number of parameters. We even go further and share the convolution weights across all channels, which proved especially beneficial for semantic segmentation, possibly due to the large reduction in parameters. Following the upscaling setup in [51], we also experimented with standard convolution layers to pre- or post-process the data itself. However, we found no benefit from such convolutions and thus stick with a combination branch that only includes PPACs.

5.2. Additional baselines

To assess the benefits of our PPAC refinement networks, we introduce two baseline networks. The *simple refinement network* takes estimates, log-probabilities, and input images and processes them jointly with standard convolutions. Here, we set the channel depth such that the number of parameters used for the PPAC network is approximately equal to the simple baseline. Additionally, we use a PAC baseline that replaces all PPACs with its non-probabilistic PAC counterpart. For this network, the probability branch is removed and the probabilities are instead concatenated with the guidance images and fed to the guidance subnetwork. We again ensure that the number of parameters remains comparable for PAC and PPAC refinement networks.

5.3. Training procedure

For fair comparison, we only train the refinement networks and do not backpropagate into the original networks

Table 1. Average end-point error *(AEE)* and 3-pixel outlier rate *(out)* on our Sintel and KITTI test splits for different normalizations of a PAC refinement network.

	Sintel (AAE)		KITTI	
	clean	final	AEE	out
HD3 [60]	1.672	1.357	1.990	6.14 %
PAC w/o normal.	1.665	1.352	1.924	6.34 %
PAC w/ kernel normal.	1.622	1.323	1.921	6.15 %
PAC w/ adv. normal. *(ours)*	**1.594**	**1.302**	**1.868**	**5.81** %

Table 2. Average-end-point error *(AEE)* and 3-pixel outlier rate *(out)* on our Sintel and KITTI test splits for different refinements.

	Sintel (AAE)		KITTI	
	clean	final	AEE	out
HD3 [60]	1.672	1.357	1.990	6.14 %
Simple refinement	1.638	1.334	1.872	5.92 %
PAC network [51]	1.594	1.302	1.868	5.81 %
PPAC network *(ours)*	**1.562**	**1.283**	**1.848**	**5.50** %
Oracle network	*1.430*	*1.149*	*1.500*	*4.48* %
SL [56]	1.634	1.340	1.953	6.41 %
FBS [4]	1.643	1.354	–	–

themselves. However, our proposed probabilistic refinement is also easily applicable in fully end-to-end training. For networks with PACs or PPACs, we apply our advanced normalization from Sec. 3.2. Here, we found it important to initialize weights $\mathbf{W}$ and $\mathbf{W}'$ with the same values. We thus initialize both with positive, random numbers and ensure that $\mathbf{W}'$ remains positive by learning in log-space.

As we aim to compare several different approaches, carrying out ablations on the official test datasets is not feasible. Thus, we split the data with available ground truth randomly into custom training, validation, and test sets. Learning rates for the optimization with Adam [36] are determined for all networks individually on the validation set. Please see Sec. 6 as well as the supplement for further training specifics. Our PyTorch code is publicly available.[4]

6. Experiments

6.1. Optical flow

We first apply our probabilistic refinement networks to the task of optical flow estimation. As underlying network, we use the state-of-the-art HD3 method [60], which yields competitive results on the major benchmarks. HD3 predicts flow in a residual fashion and estimates a discrete probability distribution at each scale. In [60], the full (discrete) matching distribution of the flow is composed, which is time- and memory-consuming. We instead upsample all probability maps via bilinear interpolation and provide the network with the probability value of the respective flow residual at all five scales. Since the residuals are subpixel-accurate and mostly fall outside of the discrete grid, we use a nearest neighbor interpolation of the probabilities.

We evaluate the proposed refinement networks on two widely used optical flow benchmarks: Sintel [6] and KITTI [13, 44]. The Sintel data is split into 862 images for training, 80 for validation, and 99 for test. Moreover, we merge the KITTI 2012 and 2015 images to obtain 319 training, 31 validation, and 44 test images. Using the procedure described in [60], we fine-tune two individual HD3 models on our own Sintel and KITTI training splits. We initialize the networks from the author-provided checkpoint pre-trained on FlyingChairs [9] as well as FlyingThings3D [28] and determine the best models on our validation images. Following [60], only images from Sintel final are used for fine-tuning of the Sintel model.

All refinement networks are trained for 500 epochs with the average end-point error (AEE) as loss function and a batch size of 8. The base learning rates are cut by a factor of two every 100 epochs. As augmentations, we only apply random cropping to sizes $(384, 768)$ and $(320, 896)$ for Sintel and KITTI data, respectively. On Sintel, individual networks are trained on the final and clean subsets.

Comparison of normalization approaches. We first demonstrate the benefit of our proposed advanced normalization scheme. Therefore, we train PAC refinement networks without normalization, similar to [51], and with kernel normalization, *c.f*. Sec. 3.2. Table 1 shows results evaluated on our test sets of Sintel and KITTI. Note that the results on Sintel clean tend to be worse than on final as this data has not been seen during HD3 fine-tuning. We observe that a PAC network with kernel normalization is able to improve the accuracy over the HD3 baseline as well as the PAC approach without normalization. However, the same network leveraging our proposed advanced normalization shows clearly the best accuracy on all test sets. We attribute this to the fact that kernel normalization does not sufficiently compensate different neighborhood conditions.

Evaluation of PPACs. We now evaluate refinement networks including PPACs in comparison to the simple and PAC baselines as introduced in Sec. 5.2. The results of flow refinement on our test splits are summarized in Table 2. The simple refinement approach based only on standard convolutions (with the same inputs) improves the flow predictions only slightly on all datasets. Applying guidance data, including the estimated probabilities, explicitly in a PAC refinement network already leads to a clear improvement. However, our proposed PPAC refinement network outperforms the PAC approach by a large margin, improving the AEE by 6.6%, 5.5% and 7.1% on Sintel clean, Sintel final

[4] https://github.com/visinf/ppac_refinement

Figure 5. Examples of ground truth *(top)*, HD3 optical flow *(middle)*, and PPAC-refined optical flow *(bottom)* on KITTI. *Best viewed on screen.*

and KITTI, respectively. We also observe that the improvement of using content-adaptive, probabilistic convolutions over a simple setup with standard convolutions is more significant on Sintel than on KITTI. We attribute this to the fact that guidance data has shown to be most effective in boundary regions [56], which play a lesser role in the KITTI dataset as the ground truth is sparse.

Even more striking than these significant accuracy gains, are the improvements in visual quality. Figs. 1 and 5 show example flow fields on Sintel final and KITTI. PPACs clearly improve the underlying HD3 estimates and lead to substantial improvements especially near motion boundaries. Additionally, our proposed approach is able to correctly propagate flow estimates into outlier regions.

To further understand the results of PPAC refinement, we evaluate an additional *oracle network*, which takes oracle confidences as input to the probability branch, which we take to be the inverse of the AEE. As this correctly assesses the reliability of each pixel, this networks provides an upper bound on the possible accuracy improvement from probabilistic refinement. Comparing the results in Table 2, we observe that an even more significant improvement would be possible if more precise probability estimates were available. This observation holds especially for the evaluation on KITTI, where a rather small amount of ground truth is available during fine-tuning. This suggests that future work on improved probability estimates in deep network has the potential of improving the accuracy in difficult areas further.

We finally compare our proposed PPAC refinement to other approaches from the literature. As PACs have shown to be the state-of-the-art method for joint upsampling, we restrict further comparison to the Semantic Lattice (SL) [56], which appeared concurrently to [51], and the Fast Bilateral Solver (FBS) [4] as a representative method that explicitly considers confidences for post-processing. The SL is trained as described in [56]. For the FBS, we use the probabilities of the last output layer of HD3 [60]. All FBS weight parameters and the trade-off parameter λ are determined via grid search on the validation set. Note that we were not able to find stable parameters for FBS on KITTI.

Table 3. Average-end-point error *(AEE)* and AEE of regions $\leq$10 pixels from occlusion boundaries *(d0-10)* evaluated on Sintel test.

	clean		final	
	AEE	d0-10	AEE	d0-10
HD3 [60]	4.788	3.225	4.666	3.786
PPAC-HD3 *(ours)*	**4.589**	**2.788**	**4.599**	**3.521**

Table 4. 3-pixel outlier rate of non-occluded/all pixels *(Out-Noc/all)* and runtimes evaluated on KITTI 2012 and 2015 test.

	KITTI 2012		KITTI 2015	
	Out-Noc	Out-all	Out-all	Runtime
HD3 [60]	2.26 %	5.41 %	6.55 %	**0.11 s**
PPAC-HD3 *(ours)*	**2.01** %	**5.09** %	**6.06** %	0.19 s

Table 2 shows that SL and FBS are both able to improve the HD3 baseline accuracy. However, PPAC refinement outperforms both previous methods by a large margin.

Evaluation on benchmarks. We finish our flow experiments by evaluating PPAC-HD3, *i.e*. HD3 optical flow [60] with PPAC refinement, on the official benchmarks. For Sintel, we initialize HD3 with the same checkpoint as used in [60]. On KITTI, we use the fine-tuned checkpoint with context module as provided online. To train our PPAC network, we leverage the entire training data provided for the Sintel and KITTI benchmarks, respectively. Again, we train separate networks for Sintel clean and final, as well as a joint refinement network for both KITTI benchmarks. In comparison to our previous experiment, we adjust the number of epochs in the learning scheme such that the total number of iterations remains approximately the same despite the larger number of images. Note that we do not use other augmentations than random cropping, since we found our refinement network to be robust w.r.t. overfitting. We attribute this to the lightweight structure of our PPAC network, which only adds approximately 12k parameters.

Results on Sintel test are summarized in Table 3. PPAC-HD3 clearly improves the accuracy of the underlying HD3 method on clean and final splits by $\sim$4.2% and $\sim$1.4%, respectively. The larger improvement on clean might be partly due to the fact that no data from this pass was used during fine-tuning of the HD3 baseline. Moreover, we observe a very significant improvement of 7.0% on the final pass and of $\sim$13.6% on Sintel clean when considering the AEE of regions closer than 10 pixels to motion boundaries *(d0-10)*. Overall, PPAC-HD3 ranks 4$^{\text{th}}$ on Sintel final among published two-frame methods and is thus highly competitive.

In Table 4, we show results on the official test sets of KITTI 2012 and 2015. PPAC-HD3 outperforms its underlying method by a large margin, leading to a substantial relative improvement of $\sim$11.1% for the outlier rate of non-

Table 5. Mean intersection over union *(mIoU)* evaluated on our Pascal VOC 2012 test split. †Results taken from [56].

	mIoU
DeepLabv3+ [7]	82.20 %
PAC refinement [51]	82.39 %
PPAC refinement *(ours)*	**82.62** %
SL [56]†	82.25 %
FBS [4]	82.28 %

occluded pixels on KITTI 2012. On KITTI 2015, PPAC refinement improves the outlier rate of all pixels from 6.55% to 6.06% (relative improvement of ∼7.5%). Our proposed method clearly ranks 1st among published optical flow approaches on both datasets. PPAC-HD3 is even able to outperform several strong scene flow methods, which leverage additional stereo data as input. Table 4 also shows computational times measured on a single GTX 1080 Ti GPU. Adding PPACs has a computational overhead of $\sim 1.7\times$, which seems justified by the strongly improved results.

6.2. Semantic segmentation

In a second set of experiments, we apply our probabilistic refinement networks to the task of semantic segmentation. We choose DeepLabv3+ [7] as a baseline using the XCeption65 variant of the model. Before feeding the logits of the segmentation network into the probability branch of our PPAC network, we normalize them with a log-softmax operation. The logits are equally used as input to the combination branch. Here, we left the values unnormalized as we found a normalization to lead to inferior results.

We evaluate our probabilistic refinement network on Pascal VOC 2012 [10] and use the training, validation, and test split of [56]. When training PAC and PPAC networks, we found it important to use different learning rates for the preprocessing branches as well as the weights in the combination branch. To determine appropriate values, we first fixed the PPAC or PAC weights to a Gaussian kernel with standard deviation $\sigma = 1$ and searched for the optimal learning rate of the guidance and probability subnetworks on the validation set. In a second step, all weights are randomly initialized and a second learning rate is determined for the convolution weights in the combination branch. Furthermore, we observed improved accuracy when initializing the bias term of PACs and PPACs to zero. All networks are trained for 500 epochs with constant learning rate, a batchsize of 16, and random image crops with size 200×272. We use the cross entropy as loss function and evaluate the mean intersection over union for validation. Please see the supplement for details, *e.g.* the network architectures.

Table 5 summarizes the results on our test split of Pascal VOC 2012. In this setting, our PPAC network requires on average 0.055s per image on a single GTX 1080 Ti GPU and

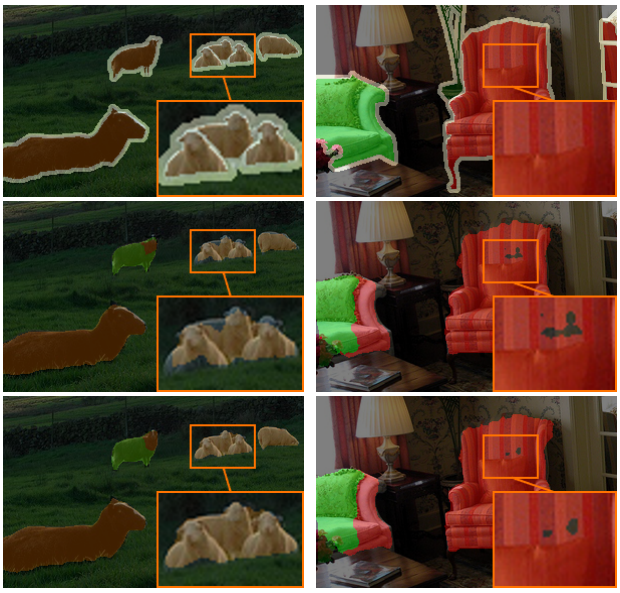


Figure 6. Cropped examples of ground truth *(top)*, DeepLabv3+ *(middle)*, and PPAC refined segmentation maps *(bottom)* on Pascal VOC 2012. *Best viewed on screen.*

is able to improve the segmentation accuracy even though DeepLabv3+ takes already special care of the decoder. In contrast, PAC refinement shows a considerably smaller benefit. We were not able to find a simple configuration that improved the original results. As for optical flow, we also compare to SL [56] and FBS [4], and use the probability of the most likely class as confidence input. Both previous methods are clearly outperformed by our proposed PPACs.

Fig. 6 shows examples of segmentation maps from Pascal VOC 2012. PPAC refinement leads to a better alignment with the underlying objects especially at object intersections or for smaller objects. Moreover, PPACs are able to successfully close smaller holes in the segmentation maps.

7. Conclusion

We introduced probabilistic pixel-adaptive convolutions (PPACs), which allow for filtering operations that respect guidance data as well as per-pixel confidences. Building on the work of [51], we first proposed an advanced normalization scheme, which we show to clearly improve the results in practice. Subsequently, we extend PACs to include confidence information during the filtering step to especially improve regions of low reliability. We proposed to use PPACs for the refinement of dense prediction networks and demonstrated their benefits for optical flow estimation and semantic segmentation. Here, PPAC refinement resulted in significant accuracy gains; our PPAC-HD3 clearly leads both KITTI benchmarks for optical flow. Moreover, refined estimates show fewer boundary artifacts and are smoother overall while correctly respecting object boundaries.

References

[1] Andrew Adams, Jongmin Baek, and Myers Abraham Davis. Fast high-dimensional filtering using the permutohedral lattice. *Comput. Graph. Forum*, 29(2):753–762, 2010.

[2] Murat Seçkin Ayhan and Philipp Berens. Test-time data augmentation for estimation of heteroscedastic aleatoric uncertainty in deep neural networks. In *International Conference on Medical Imaging with Deep Learning*, 2018.

[3] Akash Bapat and Jan-Michael Frahm. The domain transform solver. In *CVPR*, pages 6014–6023, 2019.

[4] Jonathan T. Barron and Ben Poole. The fast bilateral solver. In *ECCV*, volume 3, pages 617–632, 2016.

[5] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. In *ICML*, pages 1613–1622, 2015.

[6] Daniel J. Butler, Jonas Wulff, Garrett B. Stanley, and Michael J. Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, volume 4, pages 611–625. 2012.

[7] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, volume 7, pages 833–851, 2018.

[8] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, pages 764–773, 2017.

[9] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Häusser, Caner Hazırbaş, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. FlowNet: Learning optical flow with convolutional networks. In *ICCV*, pages 2758–2766, 2015.

[10] Mark Everingham, S. M. Ali Eslami, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The PASCAL visual object classes challenge: A retrospective. *Int. J. Comput. Vision*, 111(1):98–136, 2015.

[11] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, pages 1050–1059, 2016.

[12] Jochen Gast and Stefan Roth. Lightweight probabilistic deep networks. In *CVPR*, pages 3369–3378, 2018.

[13] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *CVPR*, pages 3354–3361, 2012.

[14] Spyros Gidaris and Nikos Komodakis. Detect, replace, refine: Deep structured prediction for pixel wise labeling. In *CVPR*, pages 5248–5257, 2017.

[15] Alex Graves. Practical variational inference for neural networks. In *NIPS*2011*, pages 2348–2356.

[16] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *ICML*, pages 1321–1330, 2017.

[17] Bumsub Ham, Minsu Cho, and Jean Ponce. Robust guided image filtering using nonconvex potentials. *IEEE T. Pattern Anal. Mach. Intell.*, 40(1):192–207, 2018.

[18] Adam W. Harley, Konstantinos G. Derpanis, and Iasonas Kokkinos. Segmentation-aware convolutional networks using local attention masks. In *ICCV*, pages 5038–5047, 2017.

[19] Kaiming He, Jian Sun, and Xiaoou Tang. Guided image filtering. In *ECCV*, volume 1, pages 1–14, 2010.

[20] José Miguel Hernández-Lobato and Ryan P. Adams. Probabilistic backpropagation for scalable learning of Bayesian neural networks. In *ICML*, pages 1861–1869, 2015.

[21] Geoffrey E. Hinton and Richard S. Zemel. Autoencoders, minimum description length and Helmholtz free energy. In *NIPS*1993*, pages 3–10.

[22] Jobst Hörentrup and Markus Schlosser. Confidence-aware guided image filter. In *ICIP*, pages 3243–3247, 2014.

[23] Gao Huang, Yixuan Li, Geoff Pleiss, Zhuang Liu, John E. Hopcroft, and Kilian Q. Weinberger. Snapshot ensembles: Train 1, get M for free. In *ICLR*, 2017.

[24] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *CVPR*, pages 770–778, 2017.

[25] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. LiteFlowNet: A lightweight convolutional neural network for optical flow estimation. In *CVPR*, pages 8981–8989, 2018.

[26] Junhwa Hur and Stefan Roth. Iterative residual refinement for joint optical flow and occlusion estimation. In *CVPR*, pages 5754–5763, 2019.

[27] Eddy Ilg, Ozgun Cicek, Silvio Galesso, Aaron Klein, Osama Makansi, Frank Hutter, and Thomas Brox. Uncertainty estimates and multi-hypotheses networks for optical flow. In *ECCV*, pages 652–667, 2018.

[28] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. FlowNet 2.0: Evolution of optical flow estimation with deep networks. In *CVPR*, pages 1647–1655, 2017.

[29] Jörn Jachalsky, Markus Schlosser, and Dirk Gandolph. Confidence evaluation for robust, fast-converging disparity map refinement. In *ICME*, pages 1399–1404, 2010.

[30] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. In *NIPS*2015*, pages 2017–2025.

[31] Varun Jampani, Martin Kiefel, and Peter V. Gehler. Learning sparse high dimensional filters: Image filtering, dense CRFs and bilateral neural networks. In *CVPR*, pages 4452–4461, 2016.

[32] Simon Jégou, Michal Drozdzal, David Vazquez, Adriana Romero, and Yoshua Bengio. The one hundred layers Tiramisu: Fully convolutional DenseNets for semantic segmentation. In *CVPR Workshops*, pages 1175–1183, 2017.

[33] Yunho Jeon and Junmo Kim. Active convolution: Learning the shape of convolution for image classification. In *CVPR*, pages 4201–4209, 2017.

[34] Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc Van Gool. Dynamic filter networks. In *NIPS*2016*, pages 667–675.

[35] Alex Kendall and Yarin Gal. What uncertainties do we need in Bayesian deep learning for computer vision? In *NIPS*2017*, pages 5574–5584.

[36] Diederik P. Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[37] Diederik P. Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. In *NIPS*2015*, pages 2575–2583.

[38] Patrick Knöbelreiter and Thomas Pock. Learned collaborative stereo refinement. In *GCPR*, pages 3–17, 2019.

[39] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NIPS*2017*, pages 6402–6413.

[40] Yijun Li, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Joint image filtering with deep convolutional networks. *IEEE T. Pattern Anal. Mach. Intell.*, 41(8):1909–1923, 2019.

[41] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 3431–3440, 2015.

[42] Christos Louizos and Max Welling. Multiplicative normalizing flows for variational Bayesian neural networks. In *ICML*, pages 2218–2227, 2017.

[43] Tan Khoa Mai, Michèle Gouiffès, and Samia Bouchafa. Optical flow refinement using reliable flow propagation. In *VISAPP*, pages 451–458, 2017.

[44] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *CVPR*, pages 3061–3070, 2015.

[45] Marcus Müller, Frederik Zilly, and Peter Kauff. Adaptive cross-trilateral depth map filtering. In *3DTV*, 2010.

[46] David A. Nix and Andreas S. Weigend. Estimating the mean and variance of the target probability distribution. In *ICNN*, pages 55–60, 1994.

[47] David Novotny, Samuel Albanie, Diane Larlus, and Andrea Vedaldi. Self-supervised learning of geometrically stable features through probabilistic introspection. In *CVPR*, pages 3637–3645, 2018.

[48] Jinshan Pan, Jiangxin Dong, Jimmy S. Ren, Liang Lin, Jinhui Tang, and Ming-Hsuan Yang. Spatially variant linear representation models for joint filtering. In *CVPR*, pages 1702–1711, 2019.

[49] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *MICCAI*, pages 234–241, 2015.

[50] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Guided curriculum model adaptation and uncertainty-aware evaluation for semantic nighttime image segmentation. In *ICCV*, pages 7373–7382, 2019.

[51] Hang Su, Varun Jampani, Deqing Sun, Orazio Gallo, Erik Learned-Miller, and Jan Kautz. Pixel-adaptive convolutional neural networks. In *CVPR*, pages 11158–11167, 2019.

[52] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *ICCV*, pages 839–846, 1998.

[53] Christoph Vogel, Patrick Knöbelreiter, and Thomas Pock. Learning energy based inpainting for optical flow. In *ACCV*, volume 6, pages 340–356, 2018.

[54] Hao Wang, Xingjian Shi, and Dit-Yan Yeung. Natural-parameter networks: A class of probabilistic neural networks. In *NIPS*2016*, pages 118–126.

[55] Jiaqi Wang, Kai Chen, Rui Xu, Ziwei Liu, Chen Change Loy, and Dahua Lin. CARAFE: Content-Aware ReAssembly of FEatures. In *ICCV*, pages 3007–3016, 2019.

[56] Anne S. Wannenwetsch, Martin Kiefel, Peter V. Gehler, and Stefan Roth. Learning task-specific generalized convolutions in the permutohedral lattice. In *GCPR*, pages 345–359, 2019.

[57] Anqi Wu, Sebastian Nowozin, Edward Meeds, Richard E. Turner, José Miguel Hernández-Lobato, and Alexander L. Gaunt. Deterministic variational inference for robust Bayesian neural networks. In *ICLR*, 2019.

[58] Huikai Wu, Shuai Zheng, Junge Zhang, and Kaiqi Huang. Fast end-to-end trainable guided filter. In *CVPR*, pages 1838–1847, 2018.

[59] Jialin Wu, Dai Li, Yu Yang, Chandrajit Bajaj, and Xiangyang Ji. Dynamic filtering with large sampling field for convnets. In *ECCV*, volume 10, pages 188–203, 2018.

[60] Zhichao Yin, Trevor Darrell, and Fisher Yu. Hierarchical discrete distribution decomposition for match density estimation. In *CVPR*, pages 6044–6053, 2019.

Probabilistic Pixel-Adaptive Refinement Networks
– Supplemental Material –

Anne S. Wannenwetsch[1,2*] Stefan Roth[2]
[1]Amazon, Germany [2] TU Darmstadt, Germany

In this supplemental material, we give further implementation details of the different types of refinement networks and provide results for a more comprehensive comparison on optical flow benchmarks. Moreover, we present an analysis considering the PPAC improvements on unreliable pixels as well as additional visualizations of PPAC-refined optical flow fields and segmentation maps.

A. Additional Implementation Details

A.1. Learning procedure

To train our networks, we use the Adam optimizer [36] with default parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$ and without weight decay. PPAC refinement networks are trained with a learning rate of 1×10^{-3} for networks on Sintel and a learning rate of 5×10^{-3} on KITTI. For semantic segmentation on Pascal VOC 2012, we use a learning rate of 1×10^{-4} for guidance and probability branches and 1×10^{-5} for the remaining PPAC parameters. The image inputs to all networks are normalized while estimates and log-probabilities remain unchanged. For faster training of all refinement networks, we save the outputs of the underlying backbone networks (*i.e*. HD3 or DeepLabv3+), and only propagate through the refinement step.

A.2. Network architectures

Tables 6 – 8 show the network structures used for PPAC, PAC, and our baseline simple refinement network, respectively. Here, 'C' represents standard convolution layers, 'P' layers with non-probabilistic PACs, and 'PP' layers with our PPACs. The networks for optical flow and semantic segmentation differ mainly by the number of input and output channels (2 or 21, respectively). For optical flow, the guidance branch uses only the first image as input since the flow fields should be aligned w.r.t. the objects in this image. All standard as well as PAC and PPAC-convolutions pad the inputs with zeros to preserve the feature size and use a stride of one. Moreover, a bias term is learned for all types of convolutions. The output of guidance and, if applicable, probability branches is split equally by the number of PAC or PPACs such that individual guidance is used for the components of the combination branch. For the simple setup, we equally experimented with networks with two convolutions and thus more channels but found the given one with three convolutions to perform better.

Table 6. Network structure of our PPAC networks for optical flow/semantic segmentation with ∼12.3k/14.3k parameters.

	layer type	kernel size	non-linearity	shared weights
guidance branch	C: $3 \to 15$	5×5	ReLU	✗
	C: $15 \to 15$	5×5	ReLU	✗
	C: $15 \to 10$	5×5	✗	✗
probability branch	C: $5/21 \to 5$	5×5	ReLU	✗
	C: $5 \to 5$	5×5	ReLU	✗
	C: $5 \to 2$	5×5	Sigmoid	✗
combination branch	PP: $2/21 \to 2/21$	7×7	✗	✓
	PP: $2/21 \to 2/21$	7×7	✗	✓

Table 7. Network structure of our PAC baseline networks for optical flow/semantic segmentation with ∼12.6k/15.5k parameters.

	layer type	kernel size	non-linearity	shared weights
guidance branch	C: $8/24 \to 15/13$	5×5	ReLU	✗
	C: $15/13 \to 15/13$	5×5	ReLU	✗
	C: $15/13 \to 10$	5×5	✗	✗
combination branch	P: $2/21 \to 2/21$	7×7	✗	✓
	P: $2/21 \to 2/21$	7×7	✗	✓

Table 8. Network structure of our simple baseline network for optical flow with a total of ∼12.4k parameters.

	layer type	kernel size	non-linearity	shared weights
simple branch	C: $10 \to 11$	7×7	ReLU	✗
	C: $11 \to 11$	7×7	ReLU	✗
	C: $11 \to 2$	7×7	✗	✗

*This work was done at TU Darmstadt prior to Anne S. Wannenwetsch joining Amazon.

Table 9. Average end-point error *(AEE)* of top-ranked two-frame optical flow methods on Sintel train and test. ⋆Re-evaluated for comparability.

	train		test	
	clean	final	clean	final
VCN [67]	(1.66)	(2.24)	**2.81**	**4.40**
IRR-PWC [26]	(1.92)	(2.51)	3.84	4.58
PWC-Net+ [64]	(1.71)	(2.34)	3.45	4.60
PPAC-HD3 *(ours)*	**(1.54)**	**(1.05)**	4.59	4.60
HD3 [60]	(1.68)⋆	(1.15)⋆	4.79	4.67

Table 10. Average end-point error *(AEE)* and 3-pixel outlier rate on non-occluded/all pixels *(Out-Noc/all)* of top-ranked optical flow methods on KITTI 2012 train and test. Results in parentheses indicate that data was used in training. †Methods use left and right stereo images. ⋆Re-evaluated for comparability.

	train	test		
	AEE	AEE	Out-Noc	Out-all
PPAC-HD3 *(ours)*	**(0.71)**	1.2	**2.01** %	5.09 %
HD3 [60]	(0.81)⋆	1.4	2.26 %	5.41 %
PRSM† [65]	–	**1.0**	2.46 %	**4.23** %
LiteFlowNet2 [62]	–	1.4	2.63 %	6.16 %
SPS-StFl† [66]	–	1.3	2.82 %	5.61 %

Table 11. Average end-point error *(AEE)*, 3-pixel outlier rate on all pixels *(Out-all)*, and runtimes *(time)* of top-ranked methods on KITTI 2015 train and test. Results in parentheses indicate that data was used in training. †Methods use left and right stereo images. ⋆Re-evaluated for comparability.

	train		test	
	AEE	Out-all	Out-all	time
UberATG-DRISF† [63]	–	–	**4.73** %	0.75 s
PPAC-HD3 *(ours)*	(1.20)	**(3.56** %)	6.06 %	0.19 s
ISF† [61]	–	–	6.22 %	600 s
VCN [67]	**(1.16)**	(4.1 %)	6.30 %	0.18 s
HD3 [60]	(1.40)⋆	(4.39 %)⋆	6.55 %	**0.11 s**⋆

B. Detailed Comparison on Optical Flow Benchmarks

For completeness, we give a more detailed comparison on benchmarks for optical flow, including the training results of PPAC-HD3 as well as the results of related work.

Table 9 shows results on Sintel clean and final. For comparability, we re-evaluated the flow fields of HD3 on the training splits, taking into account the available invalid masks. Our proposed PPAC-HD3 ranks 4$^{\text{th}}$ w.r.t. to the AEE on Sintel final.

Tables 10 and 11 summarize results for the best-ranked published methods on KITTI 2012 and 2015. For completeness, we also include scene flow methods. Note, however, that such approaches are not fully comparable as they leverage additional stereo images to compute flow. On both datasets, PPAC-HD3 ranks 1$^{\text{st}}$ among optical flow methods and 2$^{\text{nd}}$ over all published approaches on KITTI 2015. As we used the publically available checkpoint for HD3, which differs slightly from the one used in [60], we report re-evaluated results on the training sets. Moreover, we provide HD3 runtimes evaluated on the same GTX 1080 Ti GPU as PPAC-HD3 for fair comparison.

Table 12. Relative improvement of average end-point error *(AEE)*, evaluated on the 10% most unreliable and the remaining pixels of our Sintel and KITTI test splits.

	Sintel		KITTI
	clean	final	
Most unreliable pixels	9.86 %	8.93 %	4.28 %
Remaining pixels	4.11 %	3.01 %	9.42 %

C. Improvement of Unreliable Pixels

In the main paper, we argue that probabilistic pixel-adaptive refinement allows to propagate correct estimates into unreliable regions. Here, we examine the influence of PPACs on unreliable pixels in more detail. We evaluate the refinement of optical flow by computing the AEE on the 10% most unreliable pixels of each flow field and comparing it to the AEE of the remaining pixels. To assess the reliability of a pixel estimate, we upsample the probabilities of the last output scale and use nearest neighbor interpolation if the estimated residuals fall outside the probability grid. We found these reliabilities to correlate better with the optical flow errors than the ones obtained from the composed full matching probability distribution proposed in [60]. Moreover, we use the same PPAC refinement networks as trained for the experiments in Table 2.

Table 12 shows the relative improvement on unreliable and remaining pixels evaluated on our test splits of Sintel and KITTI. On Sintel clean and final, we clearly observe a more significant improvement on the unreliable pixels, justifying the conclusion that PPACs allow to replace pixels of low reliability. In contrast, our evaluation on KITTI shows a larger improvement for the remaining pixels. This correlates well with the fact that we found the output probabilities of [60] to be less well calibrated on KITTI, judging by the comparatively larger benefit of oracle confidences, *c.f*. Sec. 6.1. However, when comparing the relative improvements of PPACs to the ones obtained by PACs (8.87% for more reliable and 2.72% for uncertain pixels), we observe that PPACs nevertheless allow for better handling of unreliable regions even if the reliability estimates are not completely accurate themselves.

Figure 7. Examples of ground truth *(top)*, HD3 optical flow [60] *(middle)*, and our PPAC-refined optical flow *(bottom)* on Sintel final. *Best viewed on screen.*

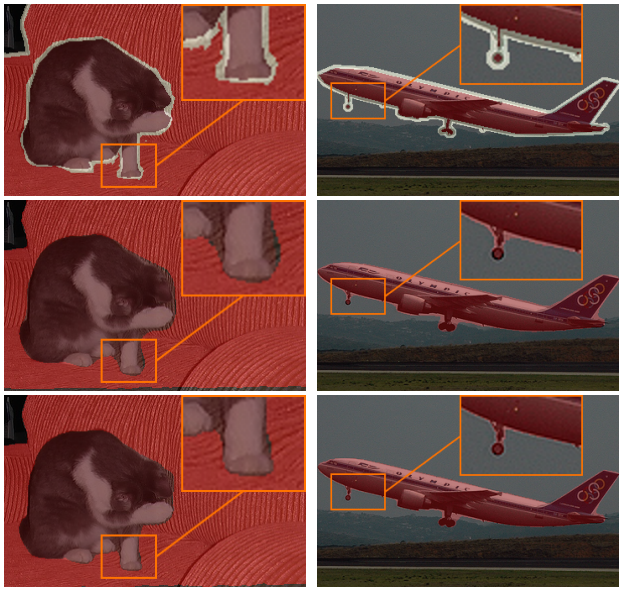


Figure 8. Additional examples of cropped ground truth *(top)*, DeepLabv3+ [7] *(middle)*, and PPAC-refined segmentation maps *(bottom)* on Pascal VOC 2012. *Best viewed on screen.*

D. Additional Visualizations

Fig. 7 shows additional visualizations of refined optical flow fields on our own validation and test splits of Sintel final. As such, none of these flow fields was presented to the PPAC refinement network during training. We clearly observe improved motion boundaries but also the ability of our approach to correctly propagate estimates into erroneous regions, *e.g*. the bird wings on the leftmost example.

In Fig. 8, we provide additional visualizations of refined segmentation maps on Pascal VOC 2012. PPAC refinement leads to a clear reduction of errors near object boundaries, *e.g*. by considerably minimizing the segmentation margin visible at the cat paw.

References

[61] Aseem Behl, Omid Hosseini Jafari, Siva Karthik Mustikovela, Hassan Abu Alhaija, Carsten Rother, and Andreas Geiger. Bounding boxes, segmentations and object coordinates: How important is recognition for 3D scene flow estimation in autonomous driving scenarios? In *ICCV*, pages 2574–2583, 2017.

[62] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. A lightweight optical flow CNN – Revisiting data fidelity and regularization. *arXiv:1903.07414 [cs.CV]*, 2019.

[63] Wei-Chiu Ma, Shenlong Wang, Rui Hu, Yuwen Xiong, and Raquel Urtasun. Deep rigid instance scene flow. In *CVPR*, pages 3614–3622, 2019.

[64] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Models matter, so does training: An empirical study of CNNs for optical flow estimation. *arXiv:1809.05571 [cs.CV]*, 2018.

[65] Christoph Vogel, Konrad Schindler, and Stefan Roth. 3D scene flow estimation with a piecewise rigid scene model. *Int. J. Comput. Vision*, 115(1):1–28, 2015.

[66] Koichiro Yamaguchi, David McAllester, and Raquel Urtasun. Efficient joint segmentation, occlusion labeling, stereo and flow estimation. In *ECCV*, volume 5, pages 756–771, 2014.

[67] Gengshan Yang and Deva Ramanan. Volumetric correspondence networks for optical flow. In *NeurIPS*2019*, pages 784–805.

BIBLIOGRAPHY

Adams, Andrew, Jongmin Baek, and Myers Abraham Davis (2010). "Fast High-Dimensional Filtering Using the Permutohedral Lattice." In: *Comput. Graph. Forum* 29.2, pp. 753–762.

Alvarez, Luis, Julio Esclarín, Martin Lefebure, and Javier Sánchez (1999). "A PDE Model for Computing the Optical Flow." In: *XVI Congreso de Ecuaciones Diferenciales y Aplicaciones*, pp. 1349–1356.

Alvarez, Luis, Rachid Deriche, Théo Papadopoulo, and Javier Sánchez (2007). "Symmetrical Dense Optical Flow Estimation with Occlusions Detection." In: *International Journal of Computer Vision* 75.3, pp. 371–385.

Anandan, Padmanabhan (1989). "A Computational Framework and an Algorithm for the Measurement of Visual Motion." In: *International Journal of Computer Vision* 2.3, pp. 283–310.

Anthwal, Shivangi and Dinesh Ganotra (2019). "An Overview of Optical Flow-based Approaches for Motion Segmentation." In: *The Imaging Science Journal* 67.5, pp. 284–294.

Bailer, Christian, Bertram Taetz, and Didier Stricker (2015). "Flow Fields: Dense Correspondence Fields for Highly Accurate Large Displacement Optical Flow Estimation." In: *Proceedings of the Fifteenth IEEE International Conference on Computer Vision*, pp. 2030–2038.

Baker, Simon, Daniel Scharstein, J. P. Lewis, Stefan Roth, Michael J. Black, and Richard Szeliski (2007). "A Database and Evaluation Methodology for Optical Flow." In: *Proceedings of the Eleventh IEEE International Conference on Computer Vision*.

Bar, Leah, Benjamin Berkels, Martin Rumpf, and Guillermo Sapiro (2007). "A Variational Framework for Simultaneous Motion Estimation and Restoration of Motion-Blurred Video." In: *Proceedings of the Eleventh IEEE International Conference on Computer Vision*.

Barron, John L., David J. Fleet, and Steven S. Beauchemin (1994). "Performance of Optical Flow Techniques." In: *International Journal of Computer Vision* 12.1, pp. 43–77.

Barron, Jonathan T. (2019). "A General and Adaptive Robust Loss Function." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 4331–4339.

Barron, Jonathan T. and Ben Poole (2016). "The Fast Bilateral Solver." In: *Proceedings of the 14th European Conference on Computer Vision*. Vol. 9907. Lecture Notes in Computer Science, pp. 617–632.

Bartolini, Franco and Alessaiidro Piva (1997). "Enhancement of the Horn and Schunck Optic Flow Algorithm by Means of Median Filters." In: *Proceedings of the International Conference on Digital Signal Processing*. Vol. 2, pp. 503–506.

Bergen, James R., Padmanabhan Anandan, Keith J. Hanna, and Rajesh Hingorani (1992). "Hierarchical Model-Based Motion Estimation." In: *Proceedings of the Second European Conference on Computer Vision*. Vol. 588. Lecture Notes in Computer Science, pp. 237–252.

Berkels, Benjamin, Claudia Kondermann, Christoph Garbe, and Martin Rumpf (2009). "Reconstructing Optical Flow Fields by Motion Inpainting." In: *Proceedings of the 7th International Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition*. Vol. 5681. Lecture Notes in Computer Science, pp. 388–400.

Black, Michael J. and Padmanabhan Anandan (1996). "The Robust Estimation of Multiple Motions: Parametric and Piecewise-Smooth Flow Fields." In: *Computer Vision and Image Understanding* 63.1, pp. 75–104.

Black, Michael J. and Allan D. Jepson (1996). "Estimating Optical Flow in Segmented Images Using Variable-Order Parametric Models with Local Deformations." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 18.10, pp. 972–986.

Bonin-Font, Francisco, Alberto Ortiz, and Gabriel Oliver (2008). "Visual Navigation for Mobile Robots: A Survey." In: *Journal of Intelligent and Robotic Systems* 53.3, pp. 263–296.

Bouthemy, Patrick and Edouard Francois (1993). "Motion Segmentation and Qualitative Dynamic Scene Analysis from an Image Sequence." In: *International Journal of Computer Vision* 10.2, pp. 157–182.

Boykov, Yuri, Olga Veksler, and Ramin Zabih (2001). "Fast Approximate Energy Minimization via Graph Cuts." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 23.11, pp. 1222–1239.

Braux-Zin, Jim, Romain Dupont, and Adrien Bartoli (2013). "A General Dense Image Matching Framework Combining Direct and Feature-based Costs." In: *Proceedings of the Fourteenth IEEE International Conference on Computer Vision*, pp. 185–192.

Brickwedde, Fabian, Steffen Abraham, and Rudolf Mester (2019). "Mono-SF: Multi-View Geometry Meets Single-View Depth for Monocular Scene Flow Estimation of Dynamic Traffic Scenes." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2780–2790.

Brox, Thomas and Jitendra Malik (2010). "Object Segmentation by Long Term Analysis of Point Trajectories." In: *Proceedings of the 11th European Conference on Computer Vision*. Vol. 6315. Lecture Notes in Computer Science, pp. 282–295.

Brox, Thomas and Jitendra Malik (2011). "Large Displacement Optical Flow: Descriptor Matching in Variational Motion Estimation." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33.3, pp. 972–986.

Brox, Thomas, Andrés Bruhn, Nils Papenberg, and Joachim Weickert (2004). "High Accuracy Optical Flow Estimation Based on a Theory for Warping." In: *Proceedings of the Eighth European Conference on Computer Vision*. Vol. 3024. Lecture Notes in Computer Science, pp. 25–36.

Bruhn, Andrés and Joachim Weickert (2006). "A Confidence Measure for Variational Optic Flow Methods." In: *Geometric Properties for Incomplete Data*. Springer, pp. 283–298.

Buades, Antoni, Jose Luis Lisani, and Marko Miladinovic (2016). "Patch-Based Video Denoising With Optical Flow Estimation." In: *IEEE Transactions on Image Processing* 25.6, pp. 2573–2586.

Butler, Daniel J., Jonas Wulff, Garrett B. Stanley, and Micheal J. Black (2012). "A Naturalistic Open Source Movie for Optical Flow Evaluation." In: *Proceedings of the 12th European Conference on Computer Vision*. Vol. 7577. Lecture Notes in Computer Science, pp. 611–625.

Chantas, Giannis K., Theodosios Gkamas, and Christophoros Nikou (2014). "Variational-Bayes Optical Flow." In: *Journal of Mathematical Imaging and Vision* 50.3, pp. 199–213.

Chao, Haiyang, Yu Gu, and Marcello R. Napolitano (2014). "A Survey of Optical Flow Techniques for Robotics Navigation Applications." In: *Journal of Intelligent and Robotic Systems* 73, pp. 361–372.

Chen, Liang-Chieh, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam (2018). "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation." In: *Proceedings of the 15th European Conference on Computer Vision*. Vol. 11211. Lecture Notes in Computer Science.

Chen, Qifeng and Vladlen Koltun (2016). "Full Flow: Optical Flow Estimation By Global Optimization over Regular Grids." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 4706–4714.

Chen, Zhuoyuan, Hailin Jin, Zhe Lin, Scott Cohen, and Ying Wu (2013). "Large Displacement Optical Flow from Nearest Neighbor Fields." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2443–2450.

De Brabandere, Bert, Xu Jia, Tinne Tuytelaars, and Luc Van Gool (2016). "Dynamic Filter Networks." In: *Advances in Neural Information Processing Systems*. Vol. 29, pp. 667–675.

Dharmasiri, Thanuja, Andrew Spek, and Tom Drummond (2018). "ENG: End-to-End Neural Geometry for Robust Depth and Pose Estimation Using CNNs." In: *Proceedings of the Thirteenth Asian Conference on Computer Vision*. Vol. 11361. Lecture Notes in Computer Science, pp. 625–642.

Domke, Justin and Yiannis Aloimonos (2007). "A Probabilistic Framework for Correspondence and Egomotion." In: *Proceedings of the International Workshop on Dynamical Vision*. Vol. 4358. Lecture Notes in Computer Science, pp. 232–242.

Dosovitskiy, Alexey, Philipp Fischer, Eddy Ilg, Philip Häusser, Caner Hazırbaş, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox (2015). "FlowNet: Learning Optical Flow with Convolutional Networks." In: *Proceedings of the Fifteenth IEEE International Conference on Computer Vision*, pp. 2758–2766.

Eldesokey, Abdelrahman, Michael Felsberg, and Fahad S. Khan (2018). "Propagating Confidences through CNNs for Sparse Data Regression." In: *Proceedings of the British Machine Vision Conference*.

Everingham, Mark, S. M. Ali Eslami, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman (2015). "The PASCAL Visual Object Classes Challenge: A Retrospective." In: *International Journal of Computer Vision* 111.1, pp. 98–136.

Fortun, Denis, Patrick Bouthemy, and Charles Kervrann (2015). "Optical Flow Modeling and Computation: A Survey." In: *Computer Vision and Image Understanding* 134, pp. 1–21.

Gadot, David and Lior Wolf (2016). "PatchBatch: A Batch Augmented Loss for Optical Flow." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 4236–4245.

Gast, Jochen and Stefan Roth (2018). "Lightweight Probabilistic Deep Networks." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3369–3378.

Geiger, Andreas, Philip Lenz, and Raquel Urtasun (2012). "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3354–3361.

Gharbi, Michaël, Jiawen Chen, Jonathan T. Barron, Samuel W. Hasinoff, and Frédo Durand (2017). "Deep Bilateral Learning for Real-time Image Enhancement." In: *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)* 36.4, 118:1–118:12.

Güney, Fatma and Andreas Geiger (2016). "Deep Discrete Flow." In: *Proceedings of the Thirteenth Asian Conference on Computer Vision*. Vol. 10115. Lecture Notes in Computer Science, pp. 207–224.

Harley, Adam W., Konstantinos G. Derpanis, and Iasonas Kokkinos (2017). "Segmentation-Aware Convolutional Networks Using Local Attention Masks." In: *Proceedings of the Sixteenth IEEE International Conference on Computer Vision*, pp. 5048–5057.

Haußecker, Horst and Hagen Spies (1999). "Motion." In: *Handbook of Computer Vision and Applications*. Vol. 2. Academic Press.

Horn, Berthold K. P. and Brian G. Schunck (1981). "Determining Optical Flow." In: *Artificial Intelligence* 17.1–3, pp. 185–203.

Hosni, Asmaa, Christoph Rhemann, Michael Bleyer, Carsten Rother, and Margrit Gelautz (2013). "Fast Cost-Volume Filtering for Visual Correspondence and Beyond." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.2, pp. 504–511.

Hui, Tak-Wai, Xiaoou Tang, and Chen Change Loy (2018). "LiteFlowNet: A Lightweight Convolutional Neural Network for Optical Flow Estimation." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8981–8989.

Hur, Junhwa and Stefan Roth (2017). "MirrorFlow: Exploiting Symmetries in Joint Optical Flow and Occlusion Estimation." In: *Proceedings of the Sixteenth IEEE International Conference on Computer Vision*, pp. 312–321.

Hur, Junhwa and Stefan Roth (2019). "Iterative Residual Refinement for Joint Optical Flow and Occlusion Estimation." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 5754–5763.

Ilg, Eddy, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox (2017). "FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1647–1655.

Ilg, Eddy, Özgün Çiçek, Silvio Galesso, Aaron Klein, Osama Makansi, Frank Hutter, and Thomas Brox (2018). "Uncertainty Estimates and Multi-hypotheses Networks for Optical Flow." In: *Proceedings of the 15th European Conference on Computer Vision*. Vol. 11211. Lecture Notes in Computer Science, pp. 677–693.

Ince, Serdar and Janusz Konrad (2008). "Occlusion-Aware Optical Flow Estimation." In: *IEEE Transactions on Image Processing* 17.8, pp. 1443–1451.

Jampani, Varun, Martin Kiefel, and Peter V. Gehler (2016). "Learning Sparse High Dimensional Filters: Image Filtering, Dense CRFs and Bilateral Neural Networks." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 4452–4461.

Janai, Joel, Fatma Güney, Jonas Wulff, Michael Black, and Andreas Geiger (2017). "Slow Flow: Exploiting High-Speed Cameras for Accurate and Diverse Optical Flow Reference Data." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1406–1416.

Janai, Joel, Fatma Güney, Aseem Behl, and Andreas Geiger (2019). "Computer Vision for Autonomous Vehicles: Problems, Datasets and State of the Art." In: *arXiv:1704.05519v2 [cs.CV]*.

Jhuang, Hueihan, Jürgen Gall, Silvia Zuffi, Cordelia Schmid, and Michael J. Black (2013). "Towards Understanding Action Recognition." In: *Proceedings of the Fourteenth IEEE International Conference on Computer Vision*, pp. 3192–3199.

Kanade, Takeo and Masatoshi Okutomi (1994). "A Stereo Matching Algorithm with an Adaptive Window: Theory and Experiment." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 16.9, pp. 920–932.

Kanaev, Andrey V. and Christopher W. Miller (2013). "Multi-Frame Super-Resolution Algorithm for Complex Motion Patterns." In: *Optics Express* 21.17, pp. 19850–19866.

Kennedy, Ryan and Camillo J. Taylor (2015). "Optical Flow with Geometric Occlusion Estimation and Fusion of Multiple Frames." In: *Proceedings of the 10th International Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition*. Vol. 8932. Lecture Notes in Computer Science, pp. 364–377.

Keuper, Margret, Björn Andres, and Thomas Brox (2015). "Motion Trajectory Segmentation via Minimum Cost Multicuts." In: *Proceedings of the Fifteenth IEEE International Conference on Computer Vision*, pp. 3271–3279.

Kiefel, Martin (2017). "Tractable Structured Prediction using the Permutohedral Lattice." Ph.D. thesis. ETH Zürich, Switzerland.

Kiefel, Martin, Varun Jampani, and Peter V. Gehler (2015). "Permutohedral Lattice CNNs." In: *Proceedings of the International Conference on Learning Representations Workshop Track*.

Kingma, Diederik P. and Max Welling (2014). "Auto-Encoding Variational Bayes." In: *Proceedings of the International Conference on Learning Representations*.

Knutsson, Hans and Carl-Fredrik Westin (1993). "Normalized and Differential Convolution." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 515–523.

Kondermann, Claudia, Daniel Kondermann, Bernd Jähne, and Christoph Garbe (2007). "An Adaptive Confidence Measure for Optical Flows Based on Linear Subspace Projections." In: *Pattern Recognition, Proceedings of the 29th DAGM-Symposium*. Vol. 4713. Lecture Notes in Computer Science, pp. 132–141.

Kondermann, Claudia, Rudolf Mester, and Christoph Garbe (2008a). "A Statistical Confidence Measure for Optical Flows." In: *Proceedings of the Tenth European Conference on Computer Vision*. Vol. 5304. Lecture Notes in Computer Science, pp. 290–301.

Kondermann, Claudia, Daniel Kondermann, and Christoph Garbe (2008b). "Postprocessing of Optical Flows Via Surface Measures and Motion Inpainting." In: *Pattern Recognition, Proceedings of the 30th DAGM-Symposium*. Vol. 5096. Lecture Notes in Computer Science, pp. 355–364.

Kondermann, Daniel, Rahul Nair, Katrin Honauer, Karsten Krispin, Jonas Andrulis, Alexander Brock, Burkhard Gussefeld, Mohsen Rahimimoghaddam, Sabine Hofmann, Claus Brenner, and Bernd Jähne (2016). "The HCI Benchmark Suite: Stereo and Flow Ground Truth With Uncertainties for Urban Autonomous Driving." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 19–28.

Li, Xi, Weiming Hu, Chunhua Shen, Zhongfei Zhang, Anthony Dick, and Anton Van Den Hengel (2013). "A Survey of Appearance Models in Visual Object Tracking." In: *ACM Transactions on Intelligent Systems and Technology* 4.4, 58:1–58:48.

Li, Yu, Dongbo Min, Minh N. Do, and Jiangbo Lu (2016). "Fast Guided Global Interpolation for Depth and Motion." In: *Proceedings of the 14th European Conference on Computer Vision*. Vol. 9907. Lecture Notes in Computer Science, pp. 717–733.

Liu, Pengpeng, Michael R. Lyu, Irwin King, and Jia Xu (2019). "SelFlow: Self-Supervised Learning of Optical Flow." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 4571–4580.

Lu, Guo, Wanli Ouyang, Dong Xu, Xiaoyun Zhang, Chunlei Cai, and Zhiyong Gao (2019). "DVC: An End-To-End Deep Video Compression Framework." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 11006–11015.

Lucas, Bruce D. and Takeo Kanade (1981). "An Iterative Image Registration Technique with an Application to Stereo Vision." In: *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 674–679.

Mac Aodha, Oisin, Ahmad Humayun, Marc Pollefeys, and Gabriel J. Brostow (2013). "Learning a Confidence Measure for Optical Flow." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.5, pp. 1107–1120.

Martin, David, Charless Fowlkes, Doron Tal, and Jitendra Malik (2001). "A Database of Human Segmented Natural Images and its Ap-

plication to Evaluating Segmentation Algorithms and Measuring Ecological Statistics." In: *Proceedings of the Eighth IEEE International Conference on Computer Vision*. Vol. 2, pp. 416–423.

Maurizot, M., P. Bouthemy, B. Delyon, A. Juditski, and J. M. Odobez (1995). "Determination of Singular Points in 2D Deformable Flow Fields." In: *Proceedings of the IEEE International Conference on Image Processing*. Vol. 3, pp. 488–491.

Meister, Simon, Junhwa Hur, and Stefan Roth (2018). "UnFlow: Unsupervised Learning of Optical Flow with a Bidirectional Census Loss." In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pp. 7251–7259.

Menze, Moritz, Christian Heipke, and Andreas Geiger (2018). "Object Scene Flow." In: *ISPRS Journal of Photogrammetry and Remote Sensing* 140, pp. 60–76.

Mozerov, Mikhail G. (2013). "Constrained Optical Flow Estimation as a Matching Problem." In: *IEEE Transactions on Image Processing* 22.5, pp. 2044–2055.

Nagel, Hans Hellmut and Wilfried Enkelmann (1986). "An Investigation of Smoothness Constraints for the Estimation of Displacement Vector Fields from Image Sequences." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8.5, pp. 565–593.

Nikolova, Mila and Raymond H. Chan (2007). "The Equivalence of Half-Quadratic Minimization and the Gradient Linearization Iteration." In: *IEEE Transactions on Image Processing* 16.6, pp. 1623–1627.

Novotny, David, Samuel Albanie, Diane Larlus, and Andrea Vedaldi (2018). "Self-supervised Learning of Geometrically Stable Features Through Probabilistic Introspection." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3637–3645.

Ochs, Peter, Jitendra Malik, and Thomas Brox (2014). "Segmentation of Moving Objects by Long Term Video Analysis." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.6, pp. 1187–1200.

Oliveira, Francisco P.M. and João Manuel R.S. Tavares (2014). "Medical Image Registration: A Review." In: *Computer Methods in Biomechanics and Biomedical Engineering* 17.2, pp. 73–93.

Otte, Michael and Hans-Hellmut Nagel (1994). "Optical Flow Estimation: Advances and Comparisons." In: *Proceedings of the Third*

European Conference on Computer Vision. Vol. 800. Lecture Notes in Computer Science, pp. 49–60.

Pan, Jinshan, Jiangxin Dong, Jimmy S. Ren, Liang Lin, Jinhui Tang, and Ming-Hsuan Yang (2019). "Spatially Variant Linear Representation Models for Joint Filtering." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1702–1711.

Papazoglou, Anestis and Vittorio Ferrari (2013). "Fast Object Segmentation in Unconstrained Video." In: *Proceedings of the Fourteenth IEEE International Conference on Computer Vision*, pp. 1777–1784.

Plötz, Tobias, Anne S. Wannenwetsch, and Stefan Roth (2018). "Stochastic Variational Inference with Gradient Linearization." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1566–1575.

Poggi, Matteo, Filippo Aleotti, Fabio Tosi, and Stefano Mattoccia (2020). "On the Uncertainty of Self-Supervised Monocular Depth Estimation." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3224–3234.

Poggio, Tomaso, Vincent Torre, and Christof Koch (1985). "Computational Vision and Regularization Theory." In: *Nature* 317, pp. 314–319.

Ranganath, Rajesh, Sean Gerrish, and David Blei (2014). "Black Box Variational Inference." In: *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, pp. 814–822.

Ranjan, Anurag, Varun Jampani, Lukas Balles, Kihwan Kim, Deqing Sun, Jonas Wulff, and Michael J Black (2019). "Competitive Collaboration: Joint Unsupervised Learning of Depth, Camera Motion, Optical Flow and Motion Segmentation." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 12240–12249.

Revaud, Jerome, Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid (2015). "EpicFlow: Edge-Preserving Interpolation of Correspondences for Optical Flow." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1164–1172.

Rezende, Danilo J, Shakir Mohamed, and Daan Wierstra (2014). "Stochastic Backpropagation and Approximate Inference in Deep Generative

Models." In: *Proceedings of the 31st International Conference on Machine Learning*, pp. 1278–1286.

Richter, Stephan R., Zeeshan Hayder, and Vladlen Koltun (2017). "Playing for Benchmarks." In: *Proceedings of the Sixteenth IEEE International Conference on Computer Vision*, pp. 2232–2241.

Rippel, Oren, Sanjay Nair, Carissa Lew, Steve Branson, Alexander G. Anderson, and Lubomir Bourdev (2019). "Learned Video Compression." In: *Proceedings of the Seventeenth IEEE International Conference on Computer Vision*, pp. 3454–3463.

Roy, Sébastien and Venu Govindu (2000). "MRF Solutions for Probabilistic Optical Flow Formulations." In: *Proceedings of the 15th International Conference on Pattern Recognition*. Vol. 3, pp. 1041–1047.

Sevilla-Lara, Laura, Yiyi Liao, Fatma Güney, Varun Jampani, Andreas Geiger, and Michael J. Black (2018). "On the Integration of Optical Flow and Action Recognition." In: *Proceedings of the 40th German Conference on Pattern Recognition*. Vol. 11269. Lecture Notes in Computer Science, pp. 281–297.

Simoncelli, Eero P., Edward H. Adelson, and David J. Heeger (1991). "Probability Distributions of Optical Flow." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 310–315.

Simonyan, Karen and Andrew Zisserman (2014). "Two-Stream Convolutional Networks for Action Recognition in Videos." In: *Advances in Neural Information Processing Systems*. Vol. 27, pp. 568–576.

Stein, Fridtjof (2004). "Efficient Computation of Optical Flow Using the Census Transform." In: *Pattern Recognition, Proceedings of the 26th DAGM-Symposium*. Vol. 3175. Lecture Notes in Computer Science, pp. 79–86.

Steinbrücker, Frank, Thomas Pock, and Daniel Cremers (2009). "Large Displacement Optical Flow Computation without Warping." In: *Proceedings of the Twelfth IEEE International Conference on Computer Vision*, pp. 1609–1614.

Su, Hang, Varun Jampani, Deqing Sun, Subhransu Maji, Evangelos Kalogerakis, Ming-Hsuan Yang, and Jan Kautz (2018). "SPLATNet: Sparse Lattice Networks for Point Cloud Processing." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2530–2539.

Su, Hang, Varun Jampani, Deqing Sun, Orazio Gallo, Erik Learned-Miller, and Jan Kautz (2019). "Pixel-Adaptive Convolutional Neural Networks." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 11166–11175.

Sun, Deqing (2012). "From Pixels to Layers: Joint Motion Estimation and Segmentation." Ph.D. thesis. Brown University, Providence, Rhode Island.

Sun, Deqing, Stefan Roth, J. P. Lewis, and Michael J. Black (2008). "Learning Optical Flow." In: *Proceedings of the Tenth European Conference on Computer Vision*. Vol. 5304. Lecture Notes in Computer Science, pp. 83–97.

Sun, Deqing, Stefan Roth, and Michael J. Black (2014). "A Quantitative Analysis of Current Practices in Optical Flow Estimation and the Principles Behind Them." In: *International Journal of Computer Vision* 106.2, pp. 115–137.

Sun, Deqing, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz (2018). "PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8934–8943.

Sun, Deqing, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz (2020). "Models Matter, So Does Training: An Empirical Study of CNNs for Optical Flow Estimation." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42.6, pp. 1408–1423.

Teed, Zachary and Jia Deng (2020). "RAFT: Recurrent All-Pairs Field Transforms for Optical Flow." In: *Proceedings of the 16th European Conference on Computer Vision*. Vol. 12347. Lecture Notes in Computer Science, pp. 402–419.

Tran, Dustin, Matthew D. Hoffman, Rif A. Saurous, Eugene Brevdo, Kevin Murphy, and David M. Blei (2017). "Deep Probabilistic Programming." In: *Proceedings of the International Conference on Learning Representations*.

Trobin, Werner, Thomas Pock, Daniel Cremers, and Horst Bischof (2008). "An Unbiased Second-Order Prior for High-Accuracy Motion Estimation," in: *Pattern Recognition, Proceedings of the 30th DAGM-Symposium*. Vol. 5096. Lecture Notes in Computer Science, pp. 396–405.

Vogel, Christoph, Konrad Schindler, and Stefan Roth (2013). "An Evaluation of Data Costs for Optical Flow." In: *Proceedings of the 35th*

German Conference on Pattern Recognition. Vol. 8142. Lecture Notes in Computer Science, pp. 343–353.

Wannenwetsch, Anne S. and Stefan Roth (2020). "Probabilisitic Pixel-Adaptive Refinement Networks." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11639–11648.

Wannenwetsch, Anne S., Magret Keuper, and Stefan Roth (2017). "ProbFlow: Joint Optical Flow and Uncertainty Estimation." In: *Proceedings of the Sixteenth IEEE International Conference on Computer Vision*, pp. 1182 –1191.

Wannenwetsch, Anne S., Martin Kiefel, Peter V. Gehler, and Stefan Roth (2019). "Learning Task-Specific Generalized Convolutions in the Permutohedral Lattice." In: *Proceedings of the 41st German Conference on Pattern Recognition*. Vol. 11824. Lecture Notes in Computer Science, pp. 345–359.

Wedel, Andreas, Thomas Pock, Christopher Zach, Horst Bischof, and Daniel Cremers (2009). "An Improved Algorithm for TV-L1 Optical Flow." In: *Statistical and Geometrical Approaches to Visual Motion Analysis*, pp. 23–45.

Weinzaepfel, Philippe, Jérôme Revaud, Zaïd Harchaoui, and Cordelia Schmid (2015). "Learning to Detect Motion Boundaries." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2578–2586.

Werlberger, Manuel, Thomas Pock, and Horst Bischof (2010). "Motion Estimation with Non-Local Total Variation Regularization." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2464–2471.

Werlberger, Manuel, Thomas Pock, Markus Unger, and Horst Bischof (2011). "Optical Flow Guided TV-L1 Video Interpolation and Restoration." In: *Proceedings of the 8th International Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition*. Vol. 6819. Lecture Notes in Computer Science, pp. 273–286.

Wu, Chao-Yuan, Nayan Singhal, and Philipp Krähenbühl (2018a). "Video Compression Through Image Interpolation." In: *Proceedings of the 15th European Conference on Computer Vision*. Vol. 11212. Lecture Notes in Computer Science, pp. 425–440.

Wu, Huikai, Shuai Zheng, Junge Zhang, and Kaiqi Huang (2018b). "Fast End-to-End Trainable Guided Filter." In: *Proceedings of the*

IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1838–1847.

Wu, Jialin, Dai Li, Yu Yang, Chandrajit Bajaj, and Xiangyang Ji (2018c). "Dynamic Filtering with Large Sampling Field for ConvNets." In: *Proceedings of the 15th European Conference on Computer Vision*. Vol. 11214. Lecture Notes in Computer Science, pp. 188–203.

Xiao, Jiangjian, Hui Cheng, Harpreet Sawhney, Cen Rao, and Michael Isnardi (2006). "Bilateral Filtering-Based Optical Flow Estimation with Occlusion Detection." In: *Proceedings of the Ninth European Conference on Computer Vision*. Vol. 9905. Lecture Notes in Computer Science, pp. 211–224.

Xu, Li, Jiaya Jia, and Yasuyuki Matsushita (2012). "Motion Detail Preserving Optical Flow Estimation." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.9, pp. 1744–1757.

Yilmaz, Alper, Omar Javed, and Mubarak Shah (2006). "Object Tracking: A Survey." In: *ACM Computing Surveys* 38.4, 13:1–13:45.

Yin, Zhichao and Jianping Shi (2018). "GeoNet: Unsupervised Learning of Dense Depth, Optical Flow and Camera Pose." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1983–1992.

Yin, Zhichao, Trevor Darrell, and Fisher Yu (2019). "Hierarchical Discrete Distribution Decomposition for Match Density Estimation." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 6037–6046.

Yousif, Khalid, Alireza Bab-Hadiashar, and Reza Hoseinnezhad (2015). "An Overview to Visual Odometry and Visual SLAM: Applications to Mobile Robotics." In: *Intelligent Industrial Systems* 1.4, pp. 289–311.

Yu, Jason J., Adam W. Harley, and Konstantinos G. Derpanis (2016). "Back to Basics: Unsupervised Learning of Optical Flow via Brightness Constancy and Motion Smoothness." In: *Proceedings of the 14th European Conference on Computer Vision Workshops*. Vol. 9915. Lecture Notes in Computer Science, pp. 3–10.

Zhu, Yi, Zhenzhong Lan, Shawn Newsam, and Alexander G. Hauptmann (2017). "Guided Optical Flow Learning." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*.

Zou, Yuliang, Zelun Luo, and Jia-Bin Huang (2018). "DF-Net: Unsupervised Joint Learning of Depth and Flow using Cross-Task Consistency." In: *Proceedings of the 15th European Conference on Computer Vision*. Lecture Notes in Computer Science, pp. 36–53.

PUBLICATIONS

Anne S. Wannenwetsch and Stefan Roth
Probabilistic Pixel-Adaptive Refinement Networks. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

Anne S. Wannenwetsch, Martin Kiefel, Peter V. Gehler, and Stefan Roth
Learning Task-Specific Generalized Convolutions in the Permutohedral Lattice. In: *Proceedings of the 41st German Conference on Pattern Recognition*, 2019.

Tobias Plötz*, Anne S. Wannenwetsch*, and Stefan Roth
Stochastic Variational Inference with Gradient Linearization. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

Anne S. Wannenwetsch, Margret Keuper, and Stefan Roth
ProbFlow: Joint Optical Flow and Uncertainty Estimation. In: *Proceedings of the Sixteenth IEEE International Conference on Computer Vision*, 2017.

*Authors contributed equally