Efficient local correlation volume for unsupervised optical flow estimation on small moving objects in large satellite images

Sarra Khairi, Etienne Meunier, Renaud Fraisse, Patrick Bouthemy

Efficient local correlation volume for unsupervised optical flow estimation on small moving objects in large satellite images

Sarra Khairi
Inria, Rennes, France
sarrakhairi2000@gmail.com

Etienne Meunier
Inria, Rennes, France
etienne.pj.meunier@gmail.com

Renaud Fraisse
Airbus Defence&Space, Toulouse, France
renaud.fraisse@airbus.com

Patrick Bouthemy
Inria, Rennes, France
Patrick.Bouthemy@inria.fr

Abstract

With the advent of deep learning methods, performance and efficiency of optical flow estimation has significantly increased, especially for supervised models. However, they do not generalize well to more specific data involving small moving objects in large images, such as high-resolution aerial or satellite sequences. In addition, annotation and realistic simulation are difficult for these contents, which calls for unsupervised alternatives. Yet, the latter are still less accurate than their supervised counterparts. In this paper, we introduce an unsupervised local optical flow estimation method adapted to small moving objects in large-size images by involving no downsampling of the feature maps. We adopt a local correlation search and implement it in an original way with a per-shift computation, which minimizes memory consumption and speed up inference computation for large-scale images. We also design a loss function combining similarity, smoothness and sparsity constraints. We demonstrate the performance of our SMOFlow method on real stabilized aerial videos fully representative of future satellite conditions. SMOFlow favorably compares to other methods. Our SMOFlow method is able to accurately capture the motion of small objects in large images, while efficiently reducing memory consumption.

1. Introduction

In the last years, satellite systems for remote sensing have been developed able to acquire sequences from space at metric resolution [16]. New systems capable of video capture at a given aiming point are coming with resolution around 50cm. Then, characterizing the instantaneous movements of objects of small size (a few pixels) is attainable. A reliable estimation of the motion of small objects will be an essential prerequisite for downstream tasks. The high frame-rate video acquisition allows for optical flow estimation. Optical flow provides a complete and dense information on the motion between two frames of a video [9]. It corresponds to the displacement field formed by the vectors originating from each pixel in the source image and pointing to its next position in the target image. A recent overview of optical flow methods applied to satellite image sequences shows that there is room for improvement in that context. The authors focused on the case of coupling optical flow with stereovision [28].

Nowadays, state-of-the-art optical flow methods leverage the deep learning paradigm. The best performing ones involve supervised models [27], which requires available ground truth on optical flow throughout many videos. They rely on heavy correlation computation, especially when similarity costs are computed for all possible correspondences between every pixel of the source and target images, leading to the global cost volume. For the sake of memory consumption, learned feature maps need to be drastically downsampled compared to the input image size. However, these requirements cause serious problems for some important categories of image sequences.

Indeed, there are configurations where objects of interest are small moving objects in large images, typically, vehicles in aerial or satellite imagery. In that case, the existing methods generalize poorly. They are not able to estimate the motion of the small moving objects in these large images due to the significant downsampling in the learned feature maps. The pyramidal framework associated to a coarse-to-fine process, may be another reason. In addition, optical-flow ground truth is generally not available for these image sequences, and realistic simulations not easy to generate, so that even fine-tuning is not accessible for supervised methods. This calls in addition for unsupervised models.

In this paper, we present an original and efficient un-

supervised model for accurately estimating optical flow in case of small moving objects within large images that involves *no downsampling* of the feature maps. By leveraging the fact that their displacement is limited between two consecutive images, we adopt a *local correlation volume* (LCV) that we implement in an original and efficient way with a *per-shift computation*, which avoids memory peaks and speeds up inference computation when applied to large images. In addition, we have designed an original unsupervised model whose loss function combines three terms: similarity, smoothness and also *sparsity*. Experimental results are reported on sequences of large aerial images.

The rest of the paper is organized as follows. Related work on optical flow estimation is described in Section 2. Section 3 presents our unsupersived model SMOFlow, and highlights our contributions on the local correlation block and the loss function. In Section 4, we report experimental results on real aerial image sequences, including comparison with state-of-the-art methods. Finally, Section 5 contains concluding remarks.

2. Related work

We focus on recent optical flow methods based on deep learning. For a survey of the many earlier variational optical-flow methods derived from the pioneering one [10], we refer the reader to [9, 27]. The general framework of most existing optical flow methods based on deep learning can be summarized in four fundamental steps: *(i)* feature maps of both images at different resolutions are learned; *(ii)* feature correlation is then computed in order to find correspondences between the two images; *(iii)* intermediate optical flow is predicted based on the correlation volume and the previous optical flow estimation; *(iv)* finally, the previous step (or two previous steps depending on the method) is repeated in an iterative loop. The iterative refinement is either performed within the coarse-to-fine warping-based framework [24] or with a single high-resolution flow update technique as in [13, 26, 30].

Most methods are supervised ones. FlowNet2.0 [12] proposed to stack multiple encoder-decoder networks into a large model, each taking the source image and the target image warped with the previously estimated flow. However, the model is huge (160M parameters), and therefore, prone to overfitting. SpyNet [22] and PWC-Net [24] embedded the classical coarse-to-fine approach [3] into the learning framework. SpyNet is a light-weight architecture with only 1.2M parameters, leveraging the classical principles of pyramidal processing and warping. Instead of constructing a fixed image pyramid like SpyNet, PWC-Net designed a learnable feature pyramid. In addition to the two principles aforementioned, PWCNet incorporated the use of cost volumes [6]. A cost volume stores the costs for all the possible correspondences of each pixel in the next image. Feature pyramids from both images are built prior to the cost volume computation.

The coarse-to-fine approach fails to address the well-known challenge of small, fast-moving objects. Indeed, when building the feature pyramid using successive downsamplings, moving objects of small size tend to disappear at the level where their flow can be estimated [4]. Furthermore, the warping process used by both PWC-Net and SpyNet can propagate early errors from higher levels, making the final optical flow estimate unreliable. RAFT [26] overcomes these limitations by operating at a single resolution flow field. An additional context encoder is generally used to extract the features from the source image. RAFT showed that injecting the context into the update block improves optical flow results. The intuition behind this is that it aggregates spatial information with motion boundaries.

Instead of directly operating on the 4D correlation volume as in RAFT, FlowFormer [11] uses the attention mechanism to encode the entire correlation volume into a compact memory that better captures information across pixels. The authors of [13] adopt a sparse global correlation strategy to reduce the memory consumption, while maintaining a global search for correspondences. The Sparse Correlation Volume (SCV) is constructed by computing the k nearest matches in the target feature map for each feature vector in the source feature map. This strategy results in significant memory saving, since the complexity of correlation computation is reduced to $O(Nk)$. Yet, it fails in blurry and featureless regions where top-k correlations might not be sufficient to encompass the correct match. In contrast, DIP [30] introduces a PatchMatch-based correlation volume. The initial PatchMatch method [2] leverages the fact that neighboring pixels usually have coherent matches. DIP consists of two modules: an inverse propagation module and a local search module.

In general, it is difficult to get ground truth for dense optical flow. Therefore, supervised methods usually rely on simulated realistic data for training such as [5, 20]. Unsupervised methods are then appealing alternatives for better generalization to unseen data or specific types of images [15], but generally at the cost of lower accuracy. UnFlow [23] introduces an unsupervised version of FlowNetCorr. ARFlow [17] follows the PWC-Net pipeline, while reducing its number of parameters. It integrates self-supervision from data augmentation (including spatial, occlusion and appearance transformations) to generate challenging scenes. SMURF [1] leverages RAFT architecture and performs self-supervision in a sequence-aware manner.

A few recent papers paid specific attention to the displacement range and to local information. The authors of [14] introduce dilated cost volumes to tackle both small and large displacements. Their paradigm does not require a sequential estimation strategy to compute optical flow. In

[19], a Gaussian attention mechanism can be added to optical flow models to take into account and enforce local properties both in the representation learning stage and in the matching process. It follows a preceding work of the same authors introducing kernel patch attention to account for spatial local affinity. In [7], the objective is to dissociate global motion learning and local flow estimation by considering global matching and local refinement as distinct steps. In [25], the aim was rather to handle occlusion areas.

3. Our unsupervised local method

3.1. Overall framework

Our SMOFlow model consists of three main components as summarized in Fig.1: a feature and context encoder, a local correlation block, an iterative update. In contrast to existing methods, we do not introduce any downsampling in the feature encoder. This choice stems from the objective of handling small moving objects in large images. The feature encoder consists of a 7×7 convolutional layers followed by six residual blocks, each of which has two 3×3 convolutional layers. Since the displacement range of the moving objects is limited, we consider a local cost volume (LCV). It computes the similarity between each pixel $\mathbf{p}$ in the source image I_0 and the pixels within a radius R in the target image I_1. In addition, we have designed an efficient implementation to reduce the memory consumption and computational cost of correlation computation.

We also construct a correlation pyramid by pooling the last two dimensions of the correlation volume, but without any coarse-to-fine mechanism. The first level of this pyramid captures the smallest displacements, while the highest level gives information about larger displacements. The number of layers depends on the displacement range that characterizes the data. Finally, similarly to RAFT, the iterative update stage comprises correlation lookup and GRU-based residual flow estimation. However, our method does not need any upsampling step.

3.2. Loss function

For our unsupervised SMOFlow method, we have defined a loss function composed of three terms, one similarity term and two regularization terms, the first one enforces smoothness of the estimated flow, the other one sparsity of the flow:

$$\mathcal{L}(D,\theta) = \mathcal{L}_m(D,\theta) + \beta_1 \mathcal{L}_{sm}(D,\theta) + \beta_2 \mathcal{L}_{sp}(D,\theta), \quad (1)$$

where D denotes the set of data and θ the parameters of the model. In the experiments, we set $\beta_1 = 1$ and $\beta_2 = 0.5$.

The loss term $\mathcal{L}_m$ is based on the Structural Similarity Index Measure (SSIM) f_{SSIM} [29], computed over the three R, G and B channels of the input images I_0 and I_1:

$$\mathcal{L}_m(D,\theta) = \frac{1}{|\Omega|} \sum_{\mathbf{p} \in \mathbf{\Omega}} f_{SSIM}(I_0(\mathbf{p}), I_1(\mathbf{p}')), \quad (2)$$

where Ω is the image grid and $\mathbf{p}' = \mathbf{p} + \mathbf{w}(\mathbf{p})$, $\mathbf{w}(\mathbf{p})$ denoting the flow vector at $\mathbf{p}$. The loss term $\mathcal{L}_{sm}$ is a classical edge-aware smoothness term (with $\alpha_{sm} = 20$):

$$\mathcal{L}_{sm} = \frac{1}{|\Omega|} \sum_{\mathbf{p} \in \mathbf{\Omega}} \left\| \nabla \mathbf{w}(\mathbf{p}) \right\|_{\mathbf{1}} . \mathbf{e}^{-\alpha_{\mathbf{sm}} \|\nabla \mathbf{I_0}(\mathbf{p})\|_1}. \quad (3)$$

We introduce an original sparsity term defined by:

$$\mathcal{L}_{sp} = \frac{1}{|\Omega|} \sum_{\mathbf{p} \in \Omega} \|\mathbf{w}(\mathbf{p})\|_2 . e^{-\alpha_{sp} |\frac{\partial I}{\partial t}(\mathbf{p})|}, \quad (4)$$

$$\text{with } |\frac{\partial I}{\partial t}(\mathbf{p})| \approx \frac{1}{|\mathcal{V}(\mathbf{p})|} \sum_{\mathbf{q} \in \mathcal{V}(\mathbf{p})} |I_1(\mathbf{q},t) - I_0(\mathbf{q},t)|,$$

where $\mathcal{V}(\mathbf{p})$ is a local neighborhood of $\mathbf{p}$. By adding the sparsity loss term, we leverage the fact that moving objects are scarce and small in the images. In addition, the images of our dataset are stabilized and thus the background should be static. This is expressed by the weighting function parametrized by α_{sp} (with $\alpha_{sp} = 10$). α_{sp} quantifies the extent to which the time derivative of image intensity affects the flow magnitude. We enforce that high flow magnitude is consistent with small moving objects and that the prediction of non-zero background flow is penalized.

3.3. Local correlation block

We describe the contributions necessary to make the local correlation block efficient when dealing with large images without any downsampling, which is a key challenge.

In order to reduce the memory consumption induced by the correlation block and to improve its performance, we propose an efficient way to compute the local cost volume and the gradient of the loss with respect to the feature maps. The expression of the local cost volume (LCV) is given by:

$$\begin{aligned} \text{LCV} &= \left\{ F_0(\mathbf{p}) \cdot F_1(\mathbf{p}+\mathbf{w}) \mid (\mathbf{p},\mathbf{w}) \in \Omega \times \Delta \right\} \quad (5) \\ &= \left\{ \frac{1}{\sqrt{C}} \sum_{c=0}^{C-1} F_0^c(\mathbf{p}) F_1^c(\mathbf{p}+\mathbf{w}) \mid (\mathbf{p},\mathbf{w}) \in \Omega \times \Delta \right\}, \end{aligned}$$

where C is the number of channels in the feature maps F_0 and F_1, and Δ the square local search area of size D^2. To effectively build the cost volume, we consider a *per-shift* computation instead of the usual per-pixel computation as in [26]. In the per-pixel computation, for each $\mathbf{p}$ in I_0, we extract and store the feature values of its neighbors in I_1 before computing their dot product. This approach considers the 4D local correlation volume LCV $\in \mathbb{R}^{D^2 \times H \times W}$ as a stack of $H \times W$ correlation maps of dimension D^2.

In contrast, in the per-shift computation, for each tested displacement $\mathbf{d}$ in $\mathcal{D} = [-D/2, D/2]^2 \cap \mathbb{Z}^2$, we compute the correlation between pixels $\mathbf{p}$ in I_0 and their corresponding pixels at position $\mathbf{p}+\mathbf{d}$ in I_1. The latter are

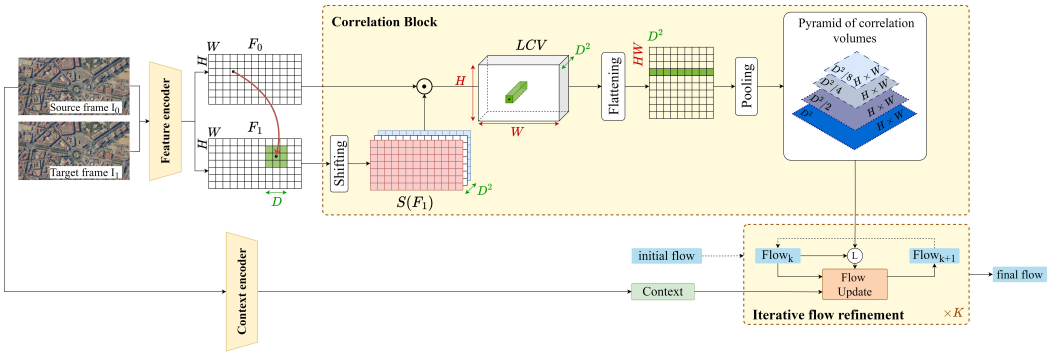


Figure 1. Overall architecture of our SMOFlow model. For the sake of clarity, we did not draw the channels of the feature maps F_0 and F_1. LCV is the local correlation volume. D^2 is the size of the search area for local correlation. $S(F_1)$ denotes the shifted version of F_1.

recovered by shifting the target feature map by $\mathbf{d}$. This approach allows us to formulate the 4D correlation volume LCV as a stack of D^2 correlation maps $\text{LCV}_{d'}$ of dimensions $H \times W$, each one representing the correlation between the source feature map and the $\mathbf{d}$-shifted target feature map. The integer d' refers to a pointer that scans the grid $\mathcal{D}$ in lexicographic order.

In the following, each shift value $\mathbf{d}$ will be mapped by g_{map} to its corresponding integer value d':

$$\begin{aligned} g_{map} : \mathcal{D} &\longrightarrow \{0, \dots, D^2 - 1\} \\ \mathbf{d} = (d_x, d_y) &\longrightarrow d' \end{aligned} \tag{6}$$

This mapping is designed such that the first cost volume LCV_0 stores the similarity costs between pixels $\mathbf{p}$ in I_0 and their top-left corresponding point $\mathbf{p} + \mathbf{d} = (x - R, y - R)$ in I_1, with $D = 2R + 1$, while the last cost volume LCV_{D^2-1} stores the correlations between pixels $\mathbf{p}$ in I_0 and their bottom-right corresponding point $\mathbf{p}+\mathbf{d} = (x+R, y+R)$ in I_1. Also, when computing the similarity between a pixel $\mathbf{p}$ in I_0 and its corresponding pixel at the same position in I_1, there is no need to shift the target feature map, which implies in that case that $d' = \frac{D^2-1}{2}$ and $\mathbf{d} = (0, 0)$.

3.4. Analytical backward

The graph structure of our local correlation block can slow down the computations performed by the autograd engine. Preliminary experiments on the correlation volume also showed that its backward is very slow. In order to improve performance during training, we implement our own custom backpropagation function. This manual (or analytical) backward requires us to express the derivatives of the local correlation volume w.r.t. its input, namely the source and target feature maps. Therefore, given the output gradient $\nabla_{\text{LCV}}\mathcal{L}$, we compute the input gradients $\nabla_{F_0}\mathcal{L}$ and $\nabla_{F_1}\mathcal{L}$.

Figure 2 illustrates the principle of the backward pass. In the backward pass, we receive a tensor containing the gradient of the loss with respect to the output, and we need to compute the gradient of the loss with respect to the input. Let us note that the latter have dimensions $C \times H \times W$, while the former has dimensions $D^2 \times H \times W$.

Given a pixel location $\mathbf{p} = (x, y)$ in the source image, let us first compute the gradient of the loss w.r.t. F_0 and F_1. Following Fig.2, we have for $k \in \{0, 1\}$:

$$\frac{\partial \mathcal{L}}{\partial F_k^{\mathbf{p}}} = \sum_{d=0}^{D^2-1} \sum_{\mathbf{q}} \frac{\partial \mathcal{L}}{\partial \text{LCV}_d^{\mathbf{q}}} \frac{\partial \text{LCV}_d^{\mathbf{q}}}{\partial F_k^{\mathbf{p}}}, \tag{7}$$

where $F_k^{\mathbf{p}}$ stands for $F_k(\mathbf{p})$, and $\frac{\partial \text{LCV}_d^{\mathbf{q}}}{\partial F_k^{\mathbf{p}}}$ are defined below.

Any element of LCV can also be expressed as $\text{LCV}_d^{\mathbf{q}} = F_0(\mathbf{q}) \cdot F_1(\mathbf{q} + \boldsymbol{\tau}_d)$, where $\boldsymbol{\tau}_d = g_{map}^{-1}(d)$ denotes the shift coordinates that correspond to integer d. Regarding the gradient of the loss w.r.t the source feature maps F_0, we have, for a given shift d, $\frac{\partial \text{LCV}_d^{\mathbf{q}}}{\partial F_0^{\mathbf{p}}} = F_1(\mathbf{q} + \boldsymbol{\tau}_\mathbf{d})$ if $\mathbf{q} = \mathbf{p}$, equal to 0 otherwise. Using eq.(7), we get:

$$\frac{\partial \mathcal{L}}{\partial F_0^{\mathbf{p}}} = \sum_{d=0}^{D^2-1} \frac{\partial \mathcal{L}}{\partial \text{LCV}_d^{\mathbf{p}}} F_1^{\mathbf{p}+\boldsymbol{\tau}_d}. \tag{8}$$

Similarly, for the gradient of the loss w.r.t the target feature map F_1, we have, for a given shift d, $\frac{\partial \text{LCV}_d^{\mathbf{q}}}{\partial F_1^{\mathbf{p}}} = F_0(\mathbf{q})$ if $\mathbf{q} = \mathbf{p} - \boldsymbol{\tau}_d$, equal to 0 otherwise. Still using eq.(7):

$$\frac{\partial \mathcal{L}}{\partial F_1^{\mathbf{p}}} = \sum_{d=0}^{D^2-1} \frac{\partial \mathcal{L}}{\partial \text{LCV}_d^{\mathbf{p}-\boldsymbol{\tau}_d}} F_0^{\mathbf{p}-\boldsymbol{\tau}_d}. \tag{9}$$

Like the architecture itself, the backward pass is implemented in Pytorch. Besides, we detail, in the supplementary material, how we have implemented an efficient memory management during inference, both for the feature encoder and the update block.

3.5. Training

Our training dataset consists of large-size image sequences. Among the sequences we process, some have a size up to

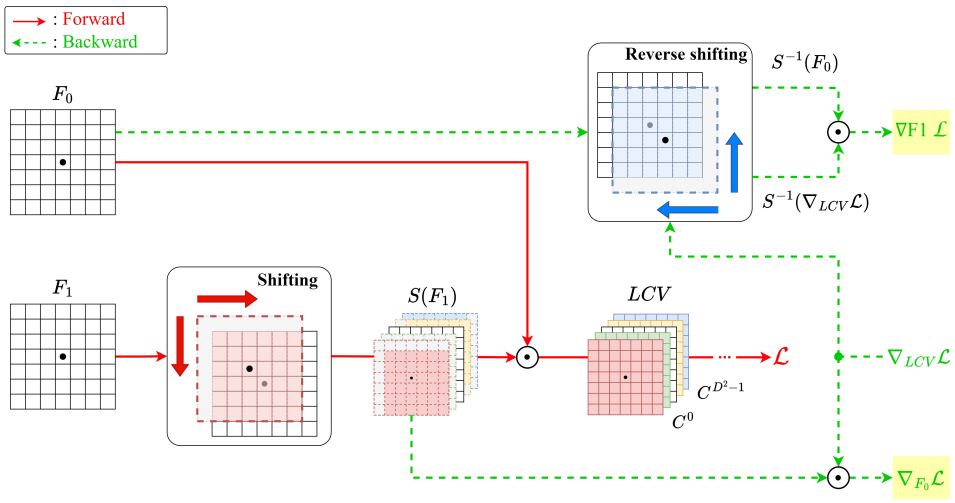


Figure 2. Principle of the backward pass. For the sake of clarity, we represent it for a single channel of the feature maps. $S(.)$ denotes the shift, and $S^{-1}(.)$ the reverse shift. $\mathcal{L}$ is the loss function.

2700×1400. Compared to RAFT [26], where the feature maps are downsampled by $\frac{1}{8}$ (which would result in a size of 337×175 for our data), our model, as it stands, would be very expensive to run at training time. That is why we divide the large-size images into sub-parts for training (but not at inference time). This sub-division can also be seen as data augmentation. Each image in a given sequence is divided into 16 patches. The training process is split into two main steps. First, we train the model on one sequence in order to see whether it converges or not. Then, to check if our model generalizes well to unseen data, we gradually increase the complexity of the training dataset by including challenging scenes. These scenes contain motion parallax or changes in the appearance of moving objects.

4. Experimental results

4.1. Implementation details

In all experiments, we set the number L of levels in the correlation pyramid to $L = 4$, the exponential weighting in the iterative refinement to $\gamma = 0.8$, the number of updates during training to 12, the radius used in the correlation lookup to 4, the maximum displacement range in the local correlation volume to $R = 12$. The latter was set to 12 by observing the vehicle displacements in the image sequences, as explained in the supplementary material. Training and inference were both performed on NVIDIA A100 - 40GB.

4.2. Dataset description

The stabilized aerial videos were provided by our industrial partner, a major player in the satellite industry. They anticipate the short videos that soon-to-be-launched satellites will be able to produce at a given aiming point. They are fully representative of the satellite conditions, in particular in terms of resolution. The resolution is the surface covered by a pixel on ground, here between 17cm and 50cm. This determines the size in pixels of a vehicle and its displacement magnitude (detailed in the supplementary material), whatever the image size. The videos are stabilized to reflect the satellite operating mode. The stabilization used the attitude sensors (like gyro) and a global image registration was performed. The dataset of aerial images contains large-size stabilized sequences involving diverse contents and comprising between 50 and 100 frames. The dataset can be divided into two categories: sequences depicting cities and sequences containing highways. A detailed description of the dataset is given in the supplementary material, along with a video. We use six sequences for training.

4.3. Downsampling impact analysis

We first carried out two experiments that we call Mosaic and Patch, to find why a poular and efficient optical flow method as RAFT [26] fails to capture the optical flow of small moving objects in large images, as illustrated in Fig.3. The Mosaic experiment consists in building a video whose every frame concatenates images of the DAVIS2016 dataset[1] [21], taken at the same time instant, this frame being then downsampled. The optical flow computed on the resulting mosaic video is then compared to the mosaic of the flows computed on each individual video forming the mosaic. The Crop experiment is in a way the reverse one. We crop a subimage of a large aerial image and upscaled it, making the small objects bigger. The two experiments are detailed in the supplementary material with associated results.

[1] https://davischallenge.org/index.html

The first intuition would be to attribute this failure to the lack of satellite image sequences in the dataset used to train RAFT. Actually, it is not the case. The reason is the absolute (small) size of moving objects in the image, associated with the downsampling performed on the feature maps by RAFT. This assessment can be extended to unsupervised methods involving the downsampling stage like SMURF [1].

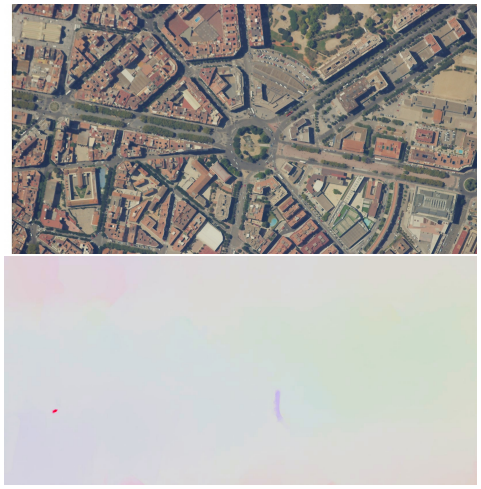


Figure 3. Flow computed with RAFT on the Placa-Tarraco sequence (images of size 1800×950). Top: sample image of the sequence. Bottom: estimated flow. Flows are represented using the usual HSV color code (see the supplementary material) with a post-processing for a better visibility. The flow on the vehicles moving along several streets or in the roundabout is mostly absent.

4.4. Comparative results on aerial sequences

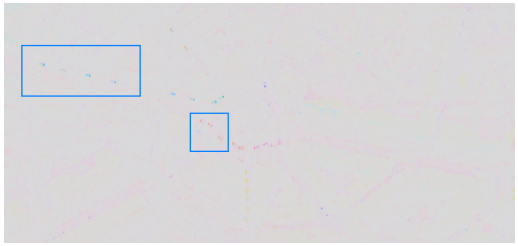
We compare our SMOFlow method with the DIP [30] and SCV [13] methods. We chose these two methods, even if they are supervised and involve a feature downsampling stage, because they tried to mitigate the downsampling and cost volume impact as described in Section 2. DIP and SCV, being supervised, have a competitive advantage, but this is counterbalanced by the fact that we cannot fine-tune them on the satellite sequences due to the lack of ground truth.

4.4.1 Visual results

We first report visual results obtained on the Placa-Tarraco sequence of Fig.3. In Fig.4, we compare flows computed on this sequence with our SMOFlow method and by the DIP and SCV methods. The visual comparison between the results provided by the three methods confirms that SMOFlow is able to correctly deal with small moving objects and to provide accurate enough flow subfields on these objects. We observe that SCV omits a large majority of small moving vehicles. DIP achieves more satisfying results in areas containing fine structures. However, it tends to blend the flows of adjacent vehicles, as shown in the red circled area. In contrast, our method discriminates close vehicles and captures more vehicles, as observed in the blue-framed areas. Visual results obtained by our SMOFlow method on other sequences are provided in the supplementary material.

(a) DIP

(b) SCV



(c) SMOFlow

Figure 4. Sample flow computed with DIP, SCV and SMOFlow on the Placa-Tarraco test sequence one image of which is shown in Fig.3. Red-circled and blue-framed areas commented in the main text. Best viewed in color and by zooming in the pdf.

In addition, we present in Fig.5 zoomed-in visual results for a better understanding of the performance achieved. The displayed flows correspond to a part of the large image for each of the three sequences. We can observe that our SMOFlow method is able to estimate the motion of almost every small vehicles, whereas DIP misses some of them and often blends the flows corresponding to different vehicles. SCV fails in getting the flow in most cases.

4.4.2 Quantitative evaluation

Since no optical-flow ground truth is available for the aerial image sequences, we have carried out an indirect objective evaluation. It is based on the classical displaced frame difference (DFD). In other words, it evaluates the accuracy of the warping provided by each compared method. Even if the DFD cannot assess the quality of any estimated flow

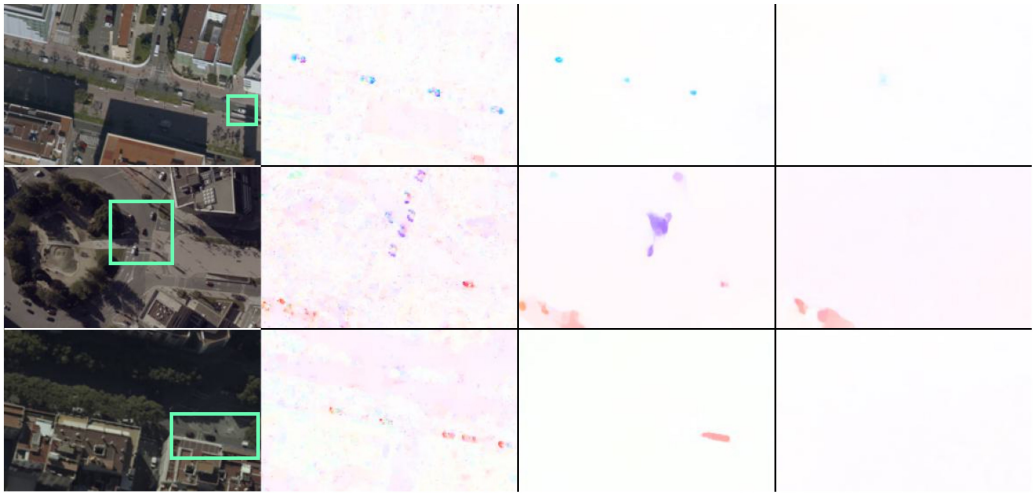


Figure 5. Zoomed-in display of subflows computed on a part of the image for three sequences from top to bottom. From left to right, part of the large image of the video sequence, sub-flows computed respectively with our SMOFlow, DIP [30], and SCV [13] methods. The green-framed areas highlight configurations for which, among others, our SMOFlow performs significantly better than DIP and SCV.

in areas of purely uniform intensity, we believe that it is a good way to get around the absence of ground truth on optical flow for this type of sequences, especially when restricted to the vehicle bounding boxes. It actually provides a convincing quantitative evaluation to compare methods.

We compute the difference between the source image and its displaced (or warped) version, referred to as its reconstructed version in the following. Then, we take its absolute value. For the reconstruction process, we use the flow $\mathbf{w}$ estimated at every pixel betwen the source image I_0 and the target image I_1. Since pixels representing the small moving objects of interest are very sparse in satellite images, averaging DFD on the entire image will not be representative enough of the model performance. Therefore, we also introduce a *sparse* DFD score by masking out background pixels, since vehicle annotations (in the form of bounding boxes) are available for part of the aerial image sequences. Our sparse DFD score is expressed by:

$$\text{DFD}_{sp} = \frac{1}{3|\Omega|} \sum_{\kappa=1}^{3} \sum_{\mathbf{p}\in\Omega} |\hat{I}_0^{\kappa}(\mathbf{p}) - I_0^{\kappa}(\mathbf{p})|.O(\mathbf{p}), \qquad (10)$$

where Ω denotes the image grid, $\hat{I}_0^{\kappa}(\mathbf{p}) = I_1^{\kappa}(\mathbf{p} + \mathbf{w}(\mathbf{p}))$ is the reconstructed source image for each color channel κ of the RGB image, O is the binary mask that occludes background pixels. The lowest the DFD score, the better the model performance in estimating the flow.

Figure 6 illustrates the DFD map and the DFD score for a selected local area in the source image. The more bright pixels in the DFD map, the poorer the model performance. The motion of the middle car has not been estimated by DIP, as highlighted by high error values in the DFD map. The same holds for SCV regarding the left car, and SCV also merges the motion of the two other cars. The warped image that appears the most similar to the source image is the one given by our model SMOFlow, which is confirmed both by the DFD map with almost no bright values and by its sparse DFD score on the selected area. The smallest DFD score equal to 0.0112 is provided by our SMOFlow method, versus 0.0160 for DIP and 0.0164 for SCV, knowing that intensity values are normalized within $[0, 1]$.

In Table 1, we report DFD scores over test sequences. Compared to DIP and SCV methods, our SMOFlow achieves good generalization on the aerial image sequences of the test set. If we restrict the computation of the DFD score to pixels corresponding to the moving vehicles, what we call the sparse DFD score, the performance gap is even more striking. Our method achieves a DFD_{sp} score of 1.20 on the test dataset, which is a 39% lower error than for DIP.

4.5. Performance of our local correlation block

We have also evaluated the performance of the SMOFlow local correlation block with respect to GPU memory consumption and computation time, and compared it to the *unfold* method of Pytorch and a shifting method using the au-

Methods / Scores	DFD_{all} $(\times 10^{-2})$	DFD_{sp} $(\times 10^{-2})$
DIP [30]	1.22	1.98
SCV [13]	1.24	2.14
Ours SMOFlow	**1.00**	**1.20**

Table 1. Quantitative evaluation on the test dataset with DFD scores for DIP, SCV and our SMOFlow. DFD_{all} refers to the DFD averaged on the entire image, while DFD_{sp} is the sparse DFD score computed over the pixels representing vehicles.

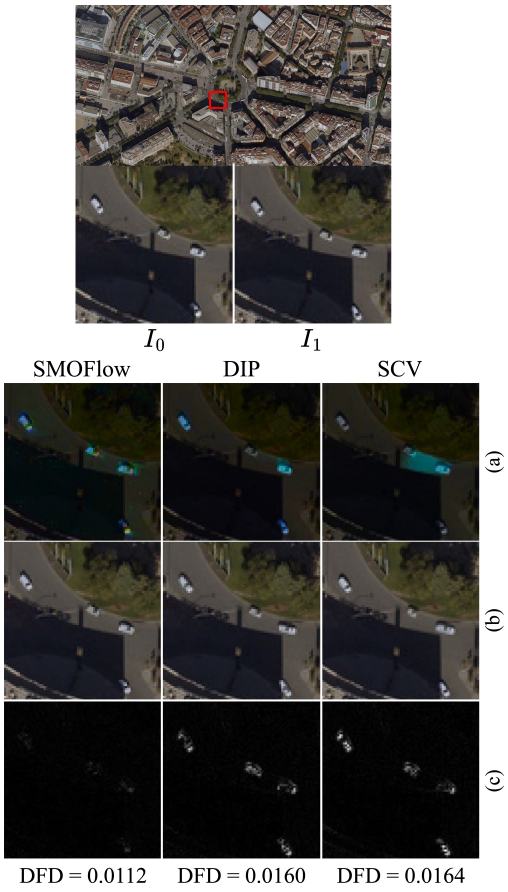


Figure 6. DFD maps computed within a local area for one instant of the Placa-Tarraco sequence for DIP, SCV and our SMOFlow. First block, top: the entire source image with the red-framed selected local area, bottom: zoom on the local area in the source (I_0) and target (I_1) images. Second block, from top to bottom: (a) Estimated flow overlaid on the selected area of I_0 (HSV color code). (b) Reconstructed source image $\hat{I}_0$. (c) DFD map on the local area. DFD$_{sp}$ scores for the local area are given at the very bottom.

tomatic differentiation provided by Pytorch. Unfold[2] is a PyTorch function that extracts sliding local blocks from a batched input tensor. Once one of the image is unfolded, we can compute the local correlation using a matrix product with the other image. It is an efficient operation but causes a large memory overhead as it copies the input data. Therefore, is is not really usable on large input such as aerial or satellite images. Comparative results are plotted in Fig.7 for different values of radius R of the search area related to the local correlation volume. The unfold Pytorch function fails for a too large image size; the bigger the radius area, the sooner the computation fails. Our direct gradient computation performs better than the automatic differentiation when the search radius increases, especially for the computation

[2]https://pytorch.org/docs/stable/generated/torch.nn.Unfold.html

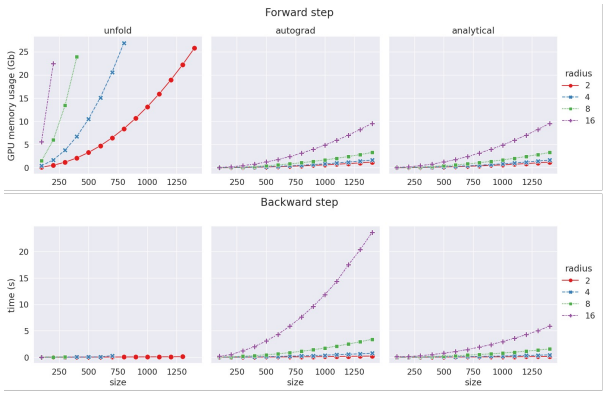


Figure 7. Performance measure of the local correlation block w.r.t. the image size for different values of the search area radius. From left to right: unfold method (which refers to the Pytorch *unfold* function), shifting method using the automatic differentiation provided by Pytorch, shifting method using our direct gradient computation. Top row: GPU memory consumption during the forward pass. Bottom row: computation time of the backward pass.

time of the backward step, crucial for training efficiency.

Let us add that the efficient version of RAFT [26] for large-scale images performs the correlation computation at every iteration. In contrast, we compute the correlations only once for a local neighborhood. Moreover, the efficient RAFT still involves feature downsampling, which means that it suffers from the same limitations than RAFT. Other existing methods involve a LCV. FlowNet [8] limits the correlation computation to a given search area, but implemented in C++. PWC-Net [24] includes a LCV in the pyramidal framework, but with a coarse-to-fine approach. In contrast, our LCV introduces a per-shift computation that is essential along with our analytical backpropagation to meet the requirements of our objective, as demonstrated in Fig.7.

5. Conclusion

We have defined an optical flow method able to reliably and accurately estimate the flow field of small moving objects in large-size images. The three key characteristics of our SMOFlow model are no downsampling of the feature maps, a loss function comprising a sparsity term, and an analytical backward pass with an efficient per-shift computation of the local correlation volume. Our implementation avoids memory peaks and minimizes computation time. Our SMOFlow method compares favorably on diverse challenging aerial image sequences to RAFT, DIP and SCV methods, while being unsupervised. Our local correlation block could be also leveraged for other remote sensing tasks where such a local computation is involved.

Acknowledgements: This work was financially supported by BpiFrance through the LiChIE contract.

References

[1] S. Austin, M. Daniel, A. Alper, A. Anelia, and J. Rico. SMURF: Self-teaching multi-frame unsupervised raft with full-image warping. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 6

[2] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 28(3), Aug. 2009. 2

[3] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *European Conference on Computer Vision (ECCV)*, Prague, 2004. 2

[4] T. Brox, C. Bregler, and J. Malik. Large displacement optical flow. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, Miami Beach, 2009. 2

[5] D.J. Butler, J. Wulff, G.B. Stanley, and M.J. Black. A naturalistic open source movie for optical flow evaluation, In *European Conference on Computer Vision (ECCV)*, 2012. 2

[6] Q. Chen and V. Koltun. Full flow: Optical flow estimation by global optimization over regular grids. In *Conf. on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas 2016. 2

[7] C. Deng, A. Luo, H. Huang, S. Ma, J. Liu, and S. Liu. Explicit motion disentangling for efficient optical flow estimation. In *International Conference on Computer Vision (ICCV)*, Paris, October 2023. 3

[8] P. Fischer, A. Dosovitskiy, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox. FlowNet: Learning optical flow with convolutional networks. In *International Conference on Computer Vision (ICCV)*, Santiago, 2015. 8

[9] D. Fortun, P. Bouthemy, and C. Kervrann. Optical flow modeling and computation: a survey. *Computer Vision and Image Understanding*, 134:1-21, May 2015. 1, 2

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

[11] Z. Huang, X. Shi, C. Zhang, Q. Wang, K. C. Cheung, H. Qin, J. Dai, and H. Li. FlowFormer: A transformer architecture for optical flow. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, New Orleans, 2022. 2

[12] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. FlowNet2.0: Evolution of optical flow estimation with deep networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, July 2017. 2

[13] S. Jiang, Y. Lu, H. Li, and R. Hartley. Learning optical flow from a few matches. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 6, 7

[14] H. Jiang and Erik Learned-Miller. DCVNet: Dilated cost volume networks for fast optical flow. In *Winter Conf. on Applications of Computer Vision (WACV)*, Waikoloa, January 2023. 2

[15] R. Jonschkowski, A. Stone1, J.T. Barron, A. Gordon, K. Konolige, and A. Angelova. What matters in unsupervised optical flow. In *European Conference on Computer Vision (ECCV)*, 2020. 2

[16] L. Lebègue, E. Cazala-Hourcade, F. Languille, S. Artigues, and O. Melet. CO3D: A worldwide one-meter accuracy DEM for 2025. In *International Society for Photogrammetry and Remote Sensing Congress (ISPRS)*, 2020. 1

[17] L. Liu, J. Zhang, R. He, Y. Liu, Y. Wang, Y. Tai, D. Luo, C. Wang, J. Li, and F. Huang. Learning by analogy: Reliable supervision from transformations for unsupervised optical flow estimation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[18] A. Luo, F. Yang, X. Li, S. Liu. Learning optical flow with kernel patch attention. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, New Orleans, 2022.

[19] A. Luo, F. Yang, X. Li, L. Nie, C. Lin, H. Fan, and S. Liu. GAFlow: Incorporating Gaussian attention into optical flow. In *International Conference on Computer Vision (ICCV)*, Paris, October 2023. 3

[20] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Conf. on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, June 2016. 2

[21] F. Perazzi, J. Pont-Tuset, B. Mc Williams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Conf. on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, June 2016. 5

[22] A. Ranjan and M. J. Black. Optical flow estimation using a spatial pyramid network. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, July 2017. 2

[23] M. Simon, H. Junhwa, and R. Stefan. UnFlow: Unsupervised learning of optical flow with a bidirectional census loss. In *Conference on Artificial Intelligence (AAAI)*, New Orleans, Feb. 2018. 2

[24] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, 2018. 2, 8

[25] S. Sun, Y. Chen, Y. Zhu, G. Guo, and G. Li. SKFlow: Learning optical llow with super kernels. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 3

[26] Z. Teed and J. Deng. RAFT: Recurrent all-pairs field transforms for optical flow. In *European Conference on Computer Vision (ECCV)*, 2020. 2, 3, 5, 8

[27] Z. Tu, W. Xie, D. Zhang, R. Poppe, R. C. Veltkamp, B. Li, and J. Yuan. A survey of variational and CNN-based optical flow techniques. *Signal Processing: Image Communication*, 72:9-24, March 2019. 1, 2

[28] X. Wang, M. Wang, and Y. Pi. An optical flow-based terrain extraction framework of VHR optical satellite stereo images. *International Journal of Applied Earth Observation and Geoinformation*, Volume 124, November 2023. 1

[29] Z. Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, April 2004. 3

[30] Z. Zheng, N. Nie, Z. Ling, P. Xiong, J. Liu, H. Wang, and J. Li. DIP: Deep inverse patchmatch for high-resolution optical flow. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, New Orleans, 2022. 2, 6, 7