Robust Optical Flow Estimation in Rainy Scenes

Li Ruoteng[1], Robby T. Tan[1,2], and Loong-Fah Cheong[1]

[1]National University of Singapore
[2]Yale-NUS

Abstract

Optical flow estimation in the rainy scenes is challenging due to background degradation introduced by rain streaks and rain accumulation effects in the scene. Rain accumulation effect refers to poor visibility of remote objects due to the intense rainfall. Most existing optical flow methods are erroneous when applied to rain sequences because the conventional brightness constancy constraint (BCC) and gradient constancy constraint (GCC) generally break down in this situation. Based on the observation that the RGB color channels receive raindrop radiance equally, we introduce a residue channel as a new data constraint to reduce the effect of rain streaks. To handle rain accumulation, our method decomposes the image into a piecewise-smooth background layer and a high-frequency detail layer. It also enforces the BCC on the background layer only. Results on both synthetic dataset and real images show that our algorithm outperforms existing methods on different types of rain sequences. To our knowledge, this is the first optical flow method specifically dealing with rain.

1. Introduction

Optical flow methods have been developed for many decades, and achieved significant results in terms of accuracy and robustness. They are shown to generally work when applied to outdoor scenes in clear daylight. However, under realistic outdoor conditions, a range of dynamic weather phenomena such as rain, snow, and sleet will pose a grim problem for these methods. In particular, of all the environmental degradations, [18] showed that rain has the most marked detrimental impact on performance. To our knowledge, no methods have been proposed to handle optical flow estimation under rainy scenes. We consider addressing this problem is important, since more and more vision systems, such as self-driving cars and surveillance, are deployed in outdoor scenes, and rain is an inevitable natural phenomena or even an everyday occurrence in some regions
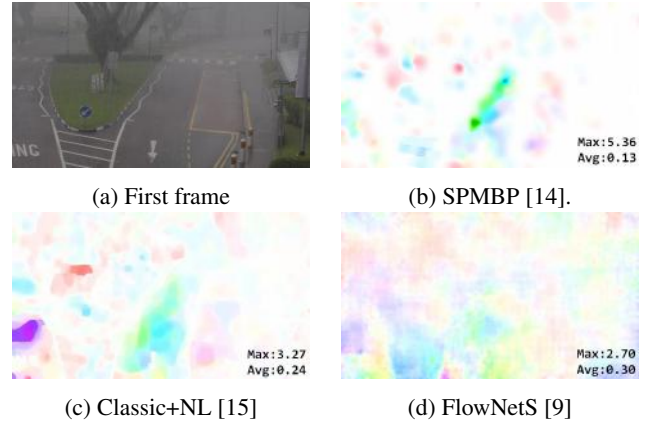

| (a) First frame | (b) SPMBP [14]. |
| (c) Classic+NL [15] | (d) FlowNetS [9] |

Figure 1: Comparison of methods on two consecutive frames of a rainy scene with static background[1]. Every object except for the rain in the scene is static, so genuine optical flow should be zero (represented by white) everywhere. The intensity of the color plots thus indicates the magnitude of the errors. The maximum flow amplitude and average flow amplitude (in pixel) of each method are denoted at the corner of each flow field. The flow field visualization of our algorithm is **perceptually white** (see also Fig 6). (Maximum flow: **0.0018** px. Average flow: **0.000195** px)

of the world. In this paper, we develop an algorithm that can handle rain in optical flow estimation. In the following, our discussion focuses on rain, though the discussion and the resulting formulation are generally applicable to other dynamic weather conditions such as snow and sleet. (Some examples and experiments on snow and sleet can be found in our supplementary material).

The challenge of estimating optical flow in rainy scenes can be categorized into two problems. One problem refers to rain streaks, which due to their dynamic nature, appear in different locations from frame to frame, thus causing viola-

---

[1]For comparison purpose, the flow is normalized to the same scale for visualization for all the methods.

tion to the brightness constancy constraint (BCC). The spurious gradients created by the rain streaks also pose problems for the gradient constancy constraint (GCC). The other problem refers to the rain streak accumulation (Fig. 1). Visually rain streaks throughout some space are accumulated and we can no longer see the streaks individually (visually similar to fog). Images affected by rain accumulation generally suffer from a veiling effect and low contrast. Under torrential downpour or heavy snow, the second problem is severe enough to warrant a special mechanism to come to grips with the issue. Existing optical flow methods usually do not consider these degradation effects, and hence fail to produce satisfactory results.

Most existing optical flow methods rely on the brightness and gradient constancy constraints, which in rainy scenes do not hold anymore or become highly susceptible to noise due to the two aforementioned problems. Various robust statistics measures proposed by the community [3][2][23][21] help treat limited noise from rains, but do not work robustly on image degradation as strong and complex as heavy rains. A direct solution is to apply a deraining method before optical flow computation. However, most of the video-based deraining methods are designed only for rain streaks removal, and assume static background. Crucially for the optical flow estimation problem, most of these **single image based** deraining methods process each frame independently, and therefore consistency across frames cannot be guaranteed. Moreover, most of the deraining methods introduce artifacts, such as blur around the rain streak region, high frequency texture loss, image color change, etc. These artifacts are also inconsistent in their appearance throughout an image sequence, thus rendering the brightness constancy constraint invalid.

In this paper, our goal is to develop an optical flow algorithm that can work robustly and accurately in the presence of rain streaks and rain accumulation. To achieve this goal, our idea is based on the observation that the radiance of most raindrops has the same intensity on each RGB channel. Hence, by subtracting the maximum channel by the minimum channel of the rain image, we can reduce the rain streak influence in the resultant map (which we call the residue channel). Besides, to handle rain accumulation, we use image decomposition to separate the image into a piecewise-smooth background layer which captures the diagnostic structure of the image, and a high-frequency detail layer which contains the noise, rain, and the fine local details of the background.

Our contributions in this paper are (1) proposed residue channel for reducing the effect of rain streaks; (2) a layer decomposition scheme to extract the principal structure of the image, with the latter providing more reliable information under low contrast, (3) a new real rain optical flow dataset for evaluation.

2. Related Work

Optical flow algorithms that are robust to noise and outliers have been studied for a long time [3][2][23][21]. While these techniques may be able to handle a moderate amount of corruptions such as those brought about by a drizzle, they are unlikely to prevail against the heavy corruptions caused by a torrential downpour. Brox et al.'s [4] utilizes the GCC to improve robustness against illumination change. However, in rainy scenes, rain streaks create spurious gradients which violate the GCC. Compounding these issues is the loss of contrast caused by rain accumulation; it renders both the brightness constancy constraint and gradient constancy constraint highly susceptible to noise.

One of the popular practices in optical flow estimation is to perform some kind of layer separation. Trobin et al.'s [22] is the first work to introduce structure-texture decomposition denoising [20] into the computation of optical flow. The purpose is to remove shadow and shading from the texture layer. However, for rainy scenes, high frequency rain streaks will appear in the texture layer and compromise the utility of the texture layer for flow estimation. Recently Yang et al.'s [25] proposes a double-layer decomposition framework to handle transparency or reflection, based on the assumption that both layers obey sparse image gradient distributions. This method cannot be used to remove the rain layer since the rain streaks result in a lot of gradients.

Mileva et.al's [17] proposes an illumination-robust variational method using color space transformation to handle shadow and highlights. Unfortunately, the HSV colour space and $r\phi\theta$ color space approaches do not result in measures that are invariant under the effects of rain streaks and hence cannot be directly applied to rainy scenes.

It is beyond the scope of this paper to offer a comprehensive review of the immense optical flow literature, but the emerging deep learning approach certainly deserves a mention. Several Convolutional Neural Network (CNN) approaches [9][8][12][19] demonstrate the possibility of using a deep learning framework to estimate flow, but these methods are meant for optical flow estimation under normal scenes. CNN-based methods are heavily optimized over a lot of training data. Unfortunately, obtaining the optical flow ground-truths for rainy scenes is not easy. This issue is compounded if we want the method to be applicable to not just rain but a variety of dynamic weather phenomena such as snow and sleet. In contrast, our method leverages on the physics of the image formation process; theoretically, it can offer a much more parsimonious solution to a range of problems posed by different weather phenomena.

A number of single-image rain streaks removal methods have been proposed [13][15][26]. Kang et al.'s [13] decomposes an input image into low frequency (rain streak free) and high frequency components, and subsequently extracted geometric details from the high frequency compo-

Figure 2: **Left**: Image captured in a rainy scene with strong rain streaks. **Right**: Residue channel of this image. The rain streaks are significantly reduced in residue channel.

nent to recover the de-rained image. Li et al.'s [15], decomposes the rain image into a rain-free background image layer and a rain streak layer and solves this formulation by introducing GMM priors of the background and rain streaks. Yang et al.'s [26] incorporates the convolutional neural network to learn the binary rain region features and rain streak intensity features. In the output image of these deraining methods [13][15][26], the rain streak regions are blurred, and the background geometric details can be lost. Hence, the derained sequences of both approaches violate both the BCC and the GCC. Our experiments show that existing optical flow method with a state-of-the-art deraining pre-processing step does not work properly due to the artifacts introduced by the deraining algorithms.

3. Residue Channel

The purpose of the residue channel is to reduce the effect of rain streaks based on the observation that the radiance of a raindrop has generally equal intensity in each RGB channel[27]. The details are as follows. The appearance of rain streaks is caused by the movement of raindrops during the camera exposure[11]. If we assume the exposure time is T and the elapsed time while a raindrop is passing through a pixel $\mathbf{x}$ is τ, the rain image I captured by the camera is a linear combination of the average raindrop radiance $\bar{E}_r$ and the background radiance E_b:

$$I(\mathbf{x}) = \tau \bar{E}_r(\mathbf{x}) + (T - \tau) E_b(\mathbf{x}), \tag{1}$$

where

$$\bar{E}_r = \frac{1}{\tau} \int_0^{\tau} E_r \mathbf{d}t, \quad 0 \leqslant \tau \leqslant T.$$

E_r is the radiance of the raindrop at a particular time. Following [27], the radiance of a raindrop on each chromatic channel, $\bar{E}_r^R, \bar{E}_r^G, \bar{E}_r^B$, is approximately the same:

$$\bar{E}_r^R = \bar{E}_r^G = \bar{E}_r^B. \tag{2}$$

For a rain image I, we define

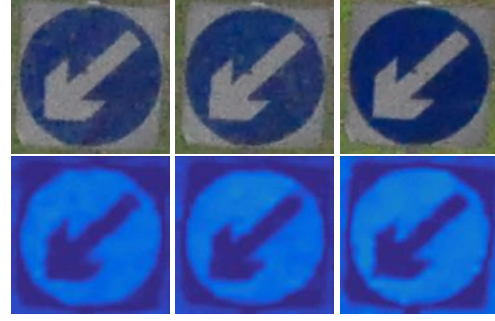$$I_{res}(\mathbf{x}) = I^M(\mathbf{x}) - I^m(\mathbf{x}), \tag{3}$$

Figure 3: Rain streaks comparison on rainy images (**Top**) and their residue channels (**Bottom**). **Left:** A sign board captured in rain at time t. **Middle:** The same sign board captured in rain at time t+δt. **Right:** Sign board captured in clear daytime. One may find the rain streaks are significantly reduced in the circle area of the sign board.

where $I^M(\mathbf{x})$ and $I^m(\mathbf{x})$ are the maximum-intensity color channel and minimum-intensity color channel of the rain image I at pixel $\mathbf{x}$ respectively. We call I_{res} the *residue channel* of image I as shown in Fig. 2.

By combining Eq.(1) and Eq.(2), the radiance due to the raindrops is cancelled:

$$I_{res} = (T - \tau)(E_b^M - E_b^m).$$

Thus, the residue channel is linearly proportional to the channel difference of the background radiance alone:

$$I_{res} = \alpha(E_b^M - E_b^m), \tag{4}$$

where $\alpha = T - \tau$ and $0 \leqslant \alpha \leqslant T$.

We consider the residue channel to be more robust to rain than the original RGB channels for optical flow estimation. Fig. 3 shows the sporadic noise points introduced by the rain streaks in the residue channel is much smaller than that in the original rain image. We have examined more than 1000 captured rain image to verify this observation (details shown in supplementary material). Based on these observation we can see that the intensity variance caused by rain in the residue channel is considerably smaller than that of the RGB channels. The rain streaks have been significantly reduced in the residue channel. This is a strong support for our residue channel hypothesis.

Rain Accumulation Rain images typically have a severe rain accumulation effect particularly in the heavy rainy scenes (*e.g.* the remote objects in Fig. 4, Left). For each pixel, the intensity contributed by the rain is the accumulation of all the raindrops along the line of sight from the camera to the background object. Thus, we model the pixel intensity by taking the summation of the radiance of the
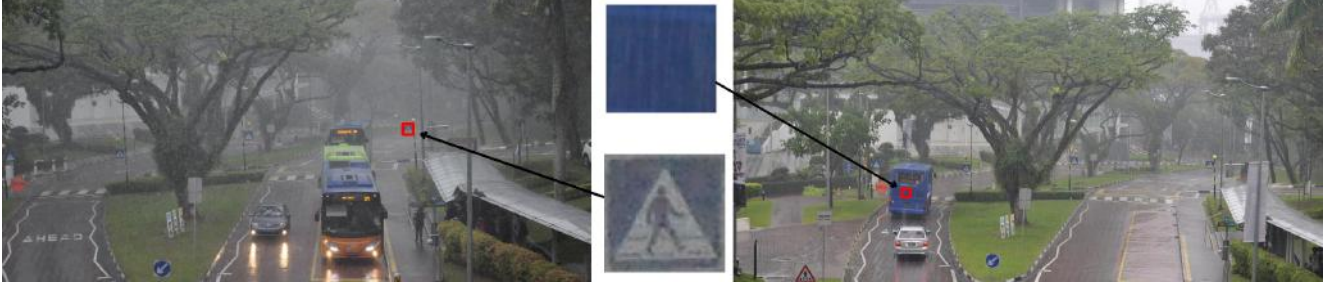
Figure 4: Rain accumulation (Left) and rain streaks (Right) effect on images. The sign board suffers from poor visibility due to rain accumulation effect.

background object and radiance of all the raindrops passing through this pixel during the exposure time.

$$I(\mathbf{x}) = \sum_{i}^{N} \alpha_i \bar{E}_i(\mathbf{x}) + (1 - \sum_{i}^{N} \alpha_i)E_b(\mathbf{x}), \quad (5)$$

where N is the number of accumulated rain-streaks along the line of sight with respect to a pixel, $\mathbf{x}$. And α_i denotes the ratio of the radiance of each raindrop to the entire radiance received by the camera. Because of light scattering and attenuation, the rain image will suffer from low contrast (Fig. 5 Left), the severity of which depends on the amount of rain accumulation. Generating the residue channel of this image, the contrast turns out to be even lower, since this is based on the residual difference between channels (Fig. 5 Right). In such low contrast images stemming from severe rain accumulation, the background texture cannot provide reliable matching due to the noise and raindrops. We can only rely on information supplied by the coarse version of the principal regions of the image, including the object boundaries. For this purpose, we embed in our method a decomposition step that separates the image into a piecewise-smooth layer and a fine-details layer.

4. Proposed Method

4.1. Residue Map

In the variational framework, the optical flow objective function is expressed as:

$$E(\mathbf{u}; I_1, I_2) = \Phi_D[I_1(\mathbf{x}) - I_2(\mathbf{x} + \mathbf{u})] + \lambda_s \Phi_S(\nabla \mathbf{u}), \quad (6)$$

where I_1, I_2 are the input sequences with spatial index $\mathbf{x}$. $\mathbf{u}$ is the flow vector with λ_s as a regularization parameter and ϕ_D and ϕ_S are the data and spatial penalty functions. We include in our objective function an additional data constraint based on the residue map with a corresponding weighting

parameter:

$$\begin{aligned} E(\mathbf{u}; I_1, I_2) = &(1 - \mathbf{w}) \odot \Phi_D[I_1(\mathbf{x}) - I_2(\mathbf{x} + \mathbf{u})] \\ &+ \mathbf{w} \odot \Phi_D[R_1(\mathbf{x}) - R_2(\mathbf{x} + \mathbf{u})] \\ &+ \lambda_s \Phi_S(\nabla \mathbf{u}), \end{aligned} \quad (7)$$

where R_1, R_2 are residue channels of rain image I_1, I_2 respectively. $\odot$ represents element-wise multiplication. w is the weighting factor defined as follows:

$$\mathbf{w} = \gamma \sqrt{(I_1^R - I_1^G)^2 + (I_1^G - I_1^B)^2 + (I_1^B - I_1^R)^2}, \quad (8)$$

where I_1^R, I_1^G, I_1^B are the RGB channels of image I_1 and γ is a scaling parameter. An object or scene with a low level of color saturation will yield low intensity and low contrast in the residue channel. In the extreme case of white and gray objects, their residue intensity would become black. Since low contrast image will be susceptible to noise, we weigh the additional residue data constraint with w that is given by the Euclidean distance between the pixel color and mid-tone gray in RGB color space.

4.2. Piecewise-smooth + Fine-detail Decomposition

When rain is relatively heavy, detailed textures on the background are severely corrupted by the ubiquitous raindrops, and are difficult to recover by the regular ROF decomposition. In this heavily degraded scenario, we resort to a more impoverished and coarse version of the scene to supply the constraint on optical flow. This version of the scene will include the principal contours of the image. For this purpose, we decompose the rain image into a piecewise-smooth layer describing the principal regions of the image and a fine-detail layer containing the background textures, raindrops, and noise. Formally, the observed rain image I can be modeled as a linear combination of the piecewise-smooth layer J and the fine-detail layer L:

$$I = J + L \quad (9)$$

In many cases of heavy rain scenarios like that in Fig. 4, the background details are seriously contaminated by the rain.

Figure 5: Sign board (**Left**) and its residue channel (**Right**) under heavy rain accumulation. The residue channel has been normalized to [0,255] for better visibility. The contrast of the residue channel is poorer than the rain image.

The fine-details are unreliable so that our method will rely on the piecewise-smooth structure of the image. Thus the decomposition can be expressed by the following energy minimization form:

$$\min_{J} \parallel I - J \parallel^2 + \parallel \nabla J \parallel_0 \quad (10)$$

where $\nabla = (\partial x, \partial y)^T$. Hence, taking the optical flow into consideration, we obtain the energy minimization problem as our formulation:

$$E(J_1, J_2, \mathbf{u}; I_1, I_2) = \Phi_D[J_1(\mathbf{x}) - J_2(\mathbf{x} + \mathbf{u})]$$
$$+ \lambda_s \Phi_S(\nabla \mathbf{u}) + \alpha(||I_1 - J_1||^2 + ||I_2 - J_2||^2) \quad (11)$$
$$+ \beta(||\nabla J_1||_0 + ||\nabla J_2||_0)$$

where I_1 I_2 are two input rain frames. J_1, J_2 are the piecewise-smooth background layers of the two frames respectively. λ_s is the smoothness parameter for the flow $\mathbf{u}$. β is the parameter controlling the gradient threshold. The higher the β, the fewer boundaries in the piecewise-smooth background layer.

4.3. Overall Objective Function

By introducing the residue channels R_1, R_2 and its corresponding weight parameter w, our overall objective function is:

$$E(J_1, J_2, \mathbf{u}; I_1, I_2) = \lambda_d\{(1 - \mathbf{w}) \odot \Phi_D[J_1(\mathbf{x}) - J_2(\mathbf{x} + \mathbf{u})]$$
$$+ \mathbf{w} \odot \Phi_D[R_1(\mathbf{x}) - R_2(\mathbf{x} + \mathbf{u})]\} + \lambda_s \Phi_S(\nabla \mathbf{u})$$
$$+ \alpha(||I_1 - J_1||^2 + ||I_2 - J_2||^2)$$
$$+ \beta(||\nabla J_1||_0 + ||\nabla J_2||_0),$$
$$\quad (12)$$

where R_1 and R_2 represent the residue channel maps of each frame correspondingly. Except for the gradients of the J layers, all the other terms are in $L2$-norm.

5. Optimization

In order to optimize our objective function, we iteratively solve the following sub-tasks given some initialization.

Sub-problem 1: Optical Flow Computation Given current piecewise-smooth background layers (J_1, J_2), we obtain the residue channel maps (R_1, R_2) and estimate the optical flow vector $\mathbf{u}$:

$$\min_{\mathbf{u}} \sum_{\mathbf{x}} \lambda_d\{(1 - \mathbf{w}) \odot \Phi_D[J_1(\mathbf{x}) - J_2(\mathbf{x} + \mathbf{u})]$$
$$+ \mathbf{w} \odot \Phi_D[R_1(\mathbf{x}) - R_2(\mathbf{x} + \mathbf{u})]\} + \lambda_s \Phi_S(\nabla \mathbf{u}) \quad (13)$$

Sub-problem 2: Layer Separation Given the current optical flow $\mathbf{u}$, we compute the piecewise-smooth background layer J_1, and J_2 separately:

$$\min_{J_1} \sum_{\mathbf{x}}\{\lambda_d \Phi_D[J_1(\mathbf{x}) - J_2(\mathbf{x} + \mathbf{u})] + \alpha(||I_1 - J_1||^2$$
$$+ \beta(||\nabla J_1||_0\} \quad (14)$$

$$\min_{J_2} \sum_{\mathbf{x}}\{\lambda_d \Phi_D[J_1(\mathbf{x}) - J_2(\mathbf{x} + \mathbf{u})] + \alpha(||I_2 - J_2||^2$$
$$+ \beta(||\nabla J_2||_0\}. \quad (15)$$

We initialize the optical flow via Eq.(13) by assuming the original rain input image as the piecewise-smooth background layer. Eq.(13) is solved via standard method in Horn-Schunck based optical flow algorithms. Eq.(14) and Eq.(15) are non-convex because of the $L0$-norm terms. Therefore, we adopt the alternating optimization strategy from [24], by introducing two auxiliary variables to decouple the unsmooth gradient term and the smooth quadratic terms. Although there is no guarantee for convergence to this non-convex problem, with initialization as proposed above, this algorithm performs well in practice. In our experiments, we have run our algorithm on hundreds of different rain scenes and it showed good convergence. The details of the steps are summarized in Algorithm 1.

6. Experiments

We evaluate our method by comparing it with representative existing methods [21] [5] [14] [9] on synthetic rain (Sect. 6.2) and real rain (Sect. 6.3) data. For synthetic rain, we rendered rain streaks following the rain model from [10] on Middlebury [1], Sintel[6] and KITTI [16] optical flow dataset. The rain streaks are generated separately and overlaid on top of the original sequences according to Eq.(1). We render the rain streaks with different intensity and direction randomly. The rain streaks' strength τ uniformly varies from 0 to 0.5. The rain streaks direction uniformly varies from -15 degree to 15 degree to the vertical axis. For real rain, we create a new dataset called the Flying Vehicles
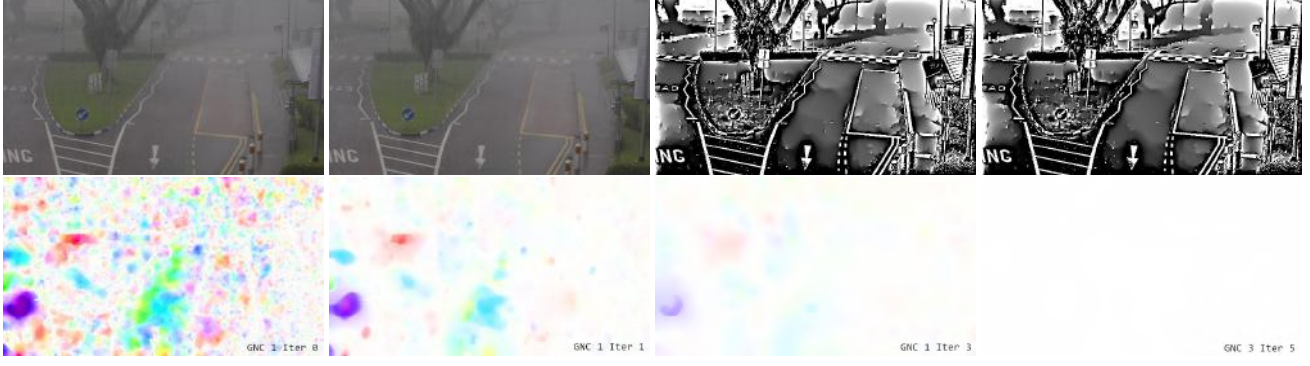
Figure 6: **Top**: Image captured in a rainy scene with strong rain streaks and rain accumulation and the decomposed background layer of this image pair. **Bottom**: From left to right: the initial estimated flow field before decomposition, intermediate result at GNC level 1, iteration 1 and iteration 2, and the final output flow field at GNC level 3 iteration 5.

Algorithm 1

1: **Input:** Image sequence I_1, I_2, regularization parameter λ_s, parameter α, β, maximum iteration M.

2: **Initialization:** Assign $J_1^{(0)} \leftarrow I_1, J_2^{(0)} \leftarrow I_2$. Estimate residue channel $R_1^{(0)} \leftarrow J_1^{(0)}, R_2^{(0)} \leftarrow J_2^{(0)}$ and initial flow $\mathbf{u}^0 \leftarrow J_1^{(0)}, J_2^{(0)}, R_1^{(0)}, R_2^{(0)}$

3: **for** iteration i = 1, ..., M **do**

4:     Compute $J_1^{(i+1)} \leftarrow J_2^{(i)}, I_1, \mathbf{u}^{(i)}$

5:     Compute $J_2^{(i+1)} \leftarrow J_1^{(i)}, I_2, \mathbf{u}^{(i)}$

6:     Compute residue channel $R_1^{(i+1)}, R_2^{(i+1)}$

7:     Estimate Flow
       $\mathbf{u}^{(i+1)} \leftarrow J_1^{(i+1)}, J_2^{(i+1)}, R_1^{(i+1)}, R_2^{(i+1)}$

8: **end for**

9: **Output:** Estimated flow field $\mathbf{u}^{(M)}$

Figure 7: Two examples from the Flying Vehicles with Rain (FVR). From left to right are generated image pair and the color coded flow field ground truth.

with Rain (FVR), using a combination of real rain images and synthetic 3D vehicle models applied with affine transformation (for more details, see Sect. 6.1). The camera we use to capture rain images is a NIKON D90 with a focal length of 45mm. All the experiments are run on a desktop with Intel(R) 12-core 3.06 GHz CPU. The time taken to process an image pair with 388x584 image resolution is around 1 minute.

6.1. Flying Vehicles with Rain

To provide a real rain dataset that can quantitatively evaluate methods, we carefully capture two consecutive frames of a static background in a rainy scene and manually crop out all the moving objects (if any) so that every object except the rain is static (shown in Fig. 6). We cut each of the resultant image frames into 12 corresponding patches as a background, each of which has a resolution of 512×384. For the foreground objects, we collect 3D vehicle mod-

els from the 'car' category in Trimble 3D Warehouse[2] and project them onto 24 2D images each with different poses. In order to render the rain onto the vehicles, we extract the rain streaks and fog-like rain accumulation from the captured rain frames using [26] and [7] respectively, and apply the rain effects extracted from the regions, which are covered by the vehicles, back on top of those vehicles.

To generate motion, we randomly sample 2D Euclidean transformation parameters from a family of Gaussian distribution for the background and the vehicle models. The background motion contains horizontal and vertical translation parameters (T_x^{bg}, T_y^{bg}) sampled from a Gaussian distribution $X \sim \mathcal{N}(0, 10)$. The background motion can be interpreted as camera translation. The foreground car models contain translation parameters (T_x^{fg}, T_y^{fg}) and rotation parameter θ^{fg} which are sample from $X \sim \mathcal{N}(0, 5)$. The car models are uniformly distributed in the first image and are rendered on the second image using the transformation

---

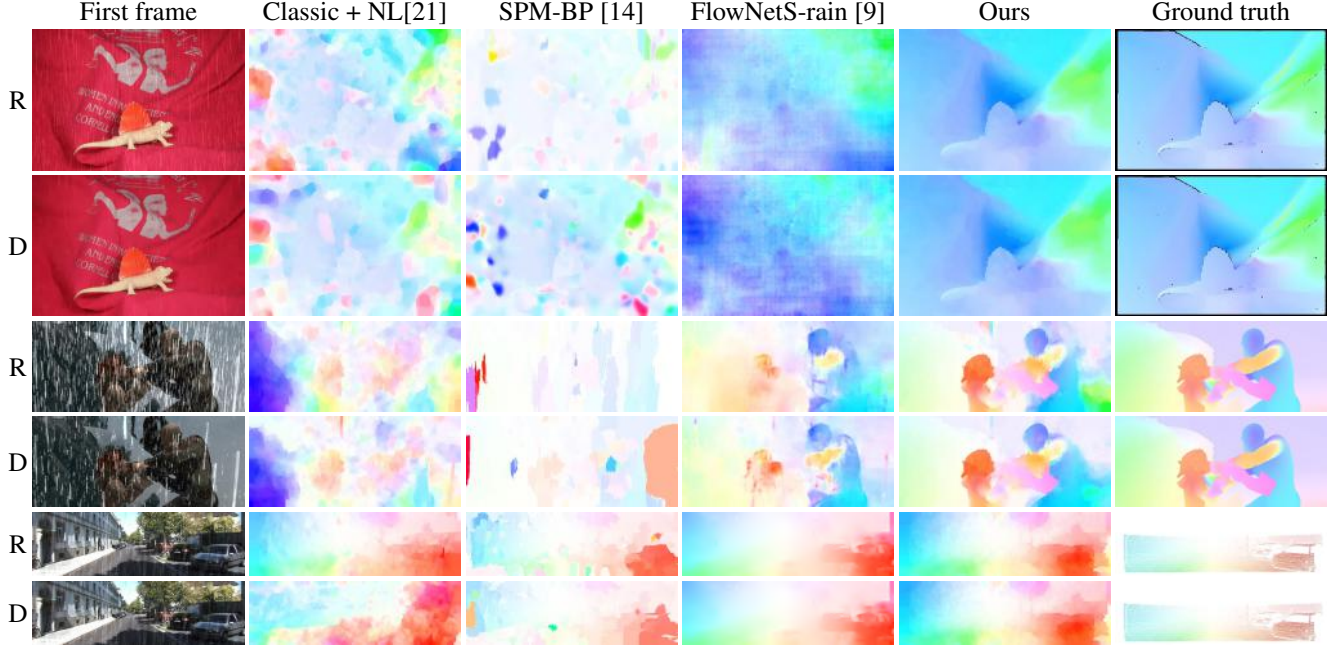[2]General model licence of Trimble Inc.

Figure 8: Method comparison on Middlebury, MPI Sintel, and KITTI datasets, all rendered with rain. The first column "R" and "D" represent synthesized rain sequences and the same sequences after [26]'s de-rain method. (**Best zoom in on screen**).

Method	Middlebury		Sintel		KITTI2012		FVR	
	Rain	De-rain	Rain	De-rain	Rain	De-rain	Rain	De-rain
Classic+NL [21]	0.90	0.60	12.89	9.51	9.17	9.14	2.79	3.20
LDOF [5]	0.90	0.66	18.26	11.84	10.17	9.90	3.83	4.09
SP-MBP [14]	0.93	0.60	14.18	7.33	15.71	15.94	5.18	5.94
FlowNetS [9]	2.58	1.54	50.90	21.96	17.43	18.73	4.75	5.05
FlowNetS-Rain	1.45	1.42	7.91	6.21	6.84	6.91	2.25	2.31
Ours	**0.30**	**0.332**	**6.11**	**5.27**	**6.65**	**6.67**	**1.90**	**2.15**

Table 1: A comparison of our algorithm with several top-performing methods on synthesized rain datasets. 'De-rain' indicates the results of each method performed on the sequences after Yang et al.'s [26] de-rain method.

parameters. As a result, this dataset contains 12 real rain image pairs with optical flow ground truth and is meant for algorithm testing and evaluation[3]. Because the vehicles are randomly generated in the rainy scene, we call it "Flying Vehicles with Rain" dataset.

6.2. Synthetic Rain Results

In the synthetic rain experiment, we compare our algorithm with some classic conventional methods, i.e. Classic+NL [21], LDOF [5], and SP-MBP [14], as well as deep learning method FlowNet [9] on the three existing datasets rendered with rain streaks. For a fairer comparison, we also

utilize the recent deraining method [26] as preprocessing and compare the results obtained in this way. The quantitative results are shown in Table 1. Fig. 8 shows the qualitative results of these comparisons. The original synthesized rain data examples are denoted with 'R' in the beginning of every row. Those examples after deraining preprocessing are denoted with 'D'. We have tuned the parameters of method [21] [5] to show their best performance in the Table 1. Due to the limited space, we include the quantitative and qualitative results of our method performed on the aforementioned three existing **clean** datasets (*i.e.* no rain) in the supplementary material. Though our method discards some fine texture details and relies more on the structure information, it still performs well on the clean datasets.

Since the training data of FlowNetS [9] does not include any rain, the FlowNetS model may not perform well under rain conditions. Hence, we render the Flying Chair dataset [9] with synthetic rain streaks using the same rain streak model as the test dataset (shown in Fig. 8) following [10]. We then train the FlowNet network using the same parameter as FlowNetS provided by the authors on this Flying Chair dataset with rain. The model is denoted as FlowNetS-rain.

6.3. Real Rain Results

For evaluations in real rain scenarios, we first present analysis of a basic static scene, followed by qualitative and quantitative results of real rain sequences with moving cam-
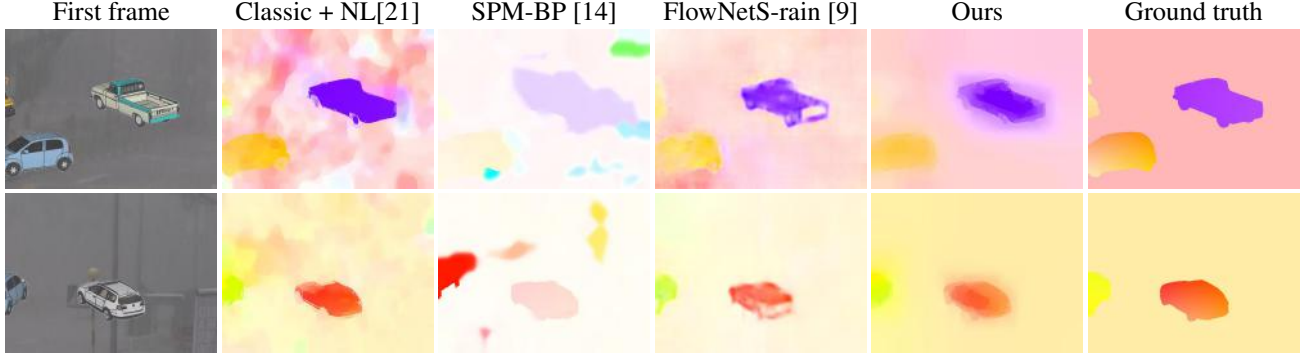
---

[3]This dataset is available at `https://github.com/liruoteng/FlyingVehiclesWithRain`

| First frame | Classic + NL[21] | SPM-BP [14] | FlowNetS-rain [9] | Ours | Ground truth |

Figure 9: Method comparison on Flying Vehicle with Rain (FVR) dataset. (**Best viewed on screen**).

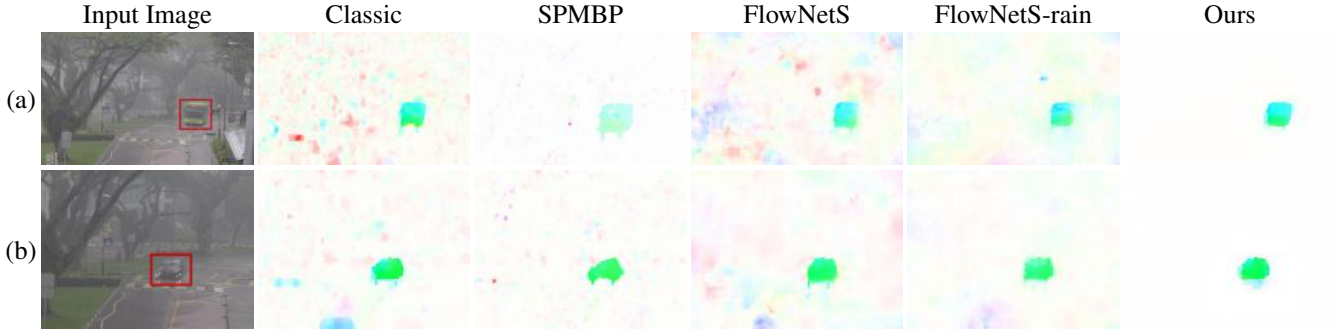| Input Image | Classic | SPMBP | FlowNetS | FlowNetS-rain | Ours |

Figure 10: Method comparison on real rainy scenes with different severity level. The red box indicates the only motion in each image pair. (**Best zoom in on screen**).

eras. Due to the lack of ground-truths for real rain scenes, we compare different methods on the FVR dataset quantitatively, because it contains real rain streaks and rain accumulation effect on a real background scene.

Static Scene Analysis   To verify the correctness and effectiveness of our algorithm, we perform a sanity check on the baseline algorithms and our algorithm on the static real-rain image pairs at one quarter (649×362) of the original image size due to the GPU memory constraint when testing FlowNetS[9]. Since this is a static scene under heavy rain, the true optical flow for the background should be zero everywhere. From Fig. 1 one can see that the baseline methods produce erroneous flow due to the motion of the rain. In comparison, the result of our algorithm is shown in the bottom row and last column of Fig.6. The average magnitude of our flow field is **0.000195** pixel, which is essentially zero flow. Fig. 6 also shows the intermediate estimated flow fields of our algorithm. One can observe that the flow field is being increasingly cleaned up in each iteration.

Quantitative Result   We compare the baseline methods with our algorithm on the FVR dataset for quantitative evaluation as shown in the last column of Table 1. We also apply

the de-rain pre-processing using [26] on the FVR dataset and compare the results of ours with the baseline methods. The results show that our algorithm outperforms those of the baseline methods even with the derain preprocessing applied. Some qualitative examples are shown in Fig.9.

Qualitative Result   Finally, we compare our algorithm with the aforementioned methods on real rainy sequences. The qualitative results are shown in Fig. 10. In order to better visualize the object motion, we select the rain sequences with only one object motion included in the red box in the images (See more results in the supplementary material). One can see that the classic and SPMBP methods generate sporadic erroneous flow on the background regions due to the rain. The result of FlowNetS-rain has a clearer background flow field than FlowNetS, which benefits from the synthesized rain training data. However, one can still observe the 'colorful' background flow and the blurry motion boundary. Our algorithm is able to generate clean blank background without losing the sharp motion boundaries of the moving car.

7. Conclusion

We have introduced a robust algorithm for optical flow estimation in rainy scenes. To come to grips with rain streaks and rain accumulation effect, we propose the residue channel and a layer decomposition scheme. In our experiments, the quantitative and qualitative results show that our method outperforms other state-of-the-art methods on both synthetic and real rain datasets.

References

[1] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, 92(1):1–31, Mar. 2011.

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

[3] M. J. Black and P. Anandan. The robust estimation of multiple motions. *Comput. Vis. Image Underst.*, 63(1):75–104, Jan. 1996.

[4] 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)*, volume 3024 of *Lecture Notes in Computer Science*, pages 25–36. Springer, May 2004.

[5] T. Brox and J. Malik. Large displacement optical flow: descriptor matching in variational motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3):500–513, 2011.

[6] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. *A Naturalistic Open Source Movie for Optical Flow Evaluation*, pages 611–625. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[7] B. Cai, X. Xu, K. Jia, C. Qing, and D. Tao. Dehazenet: An end-to-end system for single image haze removal. *IEEE Trans. Image Processing*, 25(11):5187–5198, 2016.

[8] C. B. Choy, J. Gwak, S. Savarese, and M. Chandraker. Universal correspondence network. In *Advances in Neural Information Processing Systems 29*. 2016.

[9] A. Dosovitskiy, P. Fischer, E. Ilg, V. Golkov, P. Haeusser, C. Hazrba, V. Golkov, P. Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[10] K. Garg and S. K. Nayar. Photorealistic rendering of rain streaks. *ACM Trans. Graph.*, 25(3):996–1002, July 2006.

[11] K. Garg and S. K. Nayar. Vision and rain. *Int. J. Comput. Vision*, 75(1):3–27, Oct. 2007.

[12] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. *CoRR*, abs/1612.01925, 2016.

[13] L. W. Kang, C. W. Lin, and Y. H. Fu. Automatic single-image-based rain streaks removal via image decomposition. *IEEE Transactions on Image Processing*, 21(4):1742–1755, April 2012.

[14] Y. Li, D. Min, M. S. Brown, M. N. Do, and J. Lu. Spm-bp: Sped-up patchmatch belief propagation for continuous mrfs. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 4006–4014, Dec 2015.

[15] Y. Li, R. T. Tan, X. Guo, J. Lu, and M. S. Brown. Rain streak removal using layer priors. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[16] M. Menze and A. Geiger. Object scene flow for autonomous vehicles. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[17] Y. Mileva, A. Bruhn, and J. Weickert. *Illumination-Robust Variational Optical Flow with Photometric Invariants*, pages 152–162. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[18] S. R. Richter, Z. Hayder, and V. Koltun. Playing for benchmarks. 09 2017.

[19] A. Ranjan and M. J. Black. Optical flow estimation using a spatial pyramid network. *CoRR*, abs/1611.00850, 2016.

[20] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Phys. D*, 60(1-4):259–268, Nov. 1992.

[21] D. Sun, S. Roth, and M. J. Black. Secrets of optical flow estimation and their principles. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2432–2439. IEEE, June 2010.

[22] W. Trobin, T. Pock, D. Cremers, and H. Bischof. An unbiased second-order prior for high-accuracy motion estimation. In *Pattern Recognition, 30th DAGM Symposium, Munich, Germany, June 10-13, 2008, Proceedings*, pages 396–405, 2008.

[23] A. Wedel, T. Pock, C. Zach, H. Bischof, and D. Cremers. Statistical and geometrical approaches to visual motion analysis. chapter An Improved Algorithm for TV-L1 Optical Flow, pages 23–45. Springer-Verlag, Berlin, Heidelberg, 2009.

[24] L. Xu, C. Lu, Y. Xu, and J. Jia. Image smoothing via l0 gradient minimization. *ACM Transactions on Graphics (SIGGRAPH Asia)*, 2011.

[25] J. Yang, H. Li, Y. Dai, and R. T. Tan. Robust optical flow estimation of double-layer images under transparency or reflection. 2016.

[26] W. Yang, R. T. Tan, J. Feng, J. Liu, Z. Guo, and S. Yan. Joint rain detection and removal via iterative region dependent multi-task learning. *CoRR*, abs/1609.07769, 2016.

[27] X. Zhang, H. Li, Y. Qi, W. K. Leow, and T. K. Ng. Rain removal in video by combining temporal and chromatic properties. In *2006 IEEE International Conference on Multimedia and Expo*, pages 461–464, July 2006.