# Chapter 3
# Experimental Evaluation of Four Intermediate Filters to Improve the Motion Field Estimation


**Vanel Lazcano and Claudio Isa-Mohor**

**AMS Classification** 65D19 · 68T45 · 65G35 · 70G75 · 34E13

## 3.1 Introduction

Optical Flow estimation is one of the most challenging problems in computer vision. Optical flow is defined as the per-pixel motion between two consecutive digital images. Optical flow has many applications, such as video post-production, particle velocimetry, video compression, control of autonomous vehicles, and many others. In Fig. 3.1, we show an example of these applications. A transparent plastic model full of water containing black plastic tracers is used to estimate the fluid's velocity.

In Fig. 3.1, we show velocity estimation inside a process showing an oxygen inlet and small black plastic tracers. Oxygen is injected from the left side of the plastic model, causing the fluid to move. Tracer particles move at the same velocity as the fluid. This application aims to determine the location where maximum velocity is reached to predict erosion of internal walls.

In Fig. 3.2, we show the second application of the optical flow. We show two consecutive images. In these images, we show a person riding a bike that moves to the left. We estimated the optical flow between the two images; with this optical flow,

V. Lazcano (✉)
Núcleo de Matemática, Física y Estadística, Facultad de Estudios Interdisciplinarios, Universidad Mayor, Manuel Montt 318, Providencia, Santiago, Chile
e-mail: vanel.lazcano@umayor.cl

C. Isa-Mohor
Isa y Sabaj Ltda., Santiago, Chile
e-mail: ciisa@isaysabaj.cl



P. V. Subrahmanyam et al. (eds.), *Synergies in Analysis, Discrete Mathematics, Soft Computing and Modelling*, Forum for Interdisciplinary Mathematics,
https://doi.org/10.1007/978-981-19-7014-6_3

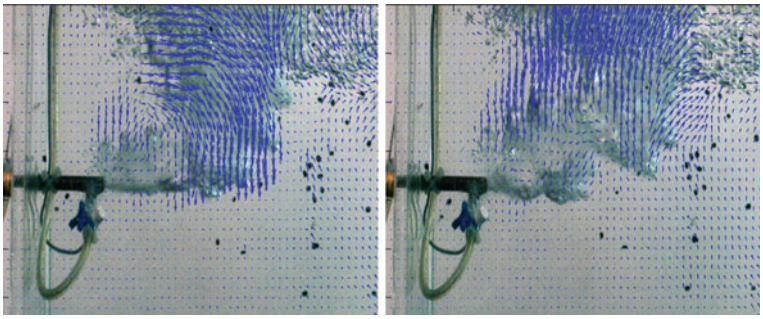

**Fig. 3.1** Velocity fluid estimation inside a plastic model of a flow dynamic process. Optical flow is represented with blue arrows in these two consecutive images


**Fig. 3.2** Creation of an image in between. **a** Current image. **b** Next image. **c** Image in between created based on optical estimation

we know the displacement for each pixel, so an image in between can be created. We show in Fig. 3.2c an interpolation of two images using optical flow.

### *3.1.1 Related Works*

Since the seminal work of [1], many contributions have been made in order to improve the optical flow estimation. In that work, the authors proposed a variational model to estimate the optical flow. The proposal is an energy model to estimate the optical flow estimation error, and the argument that minimizes that energy is the optical flow of the sequence of images. The proposal is a model that uses a quadratic error, and it means that the model is susceptible to outliers and the presence of noise. Zach et al. [2] proposed another model based on the absolute value of the error. All those models in an iterative way minimize the energy error model. In each iteration, some of them filter the optical estimation to eliminate noise or outliers, avoiding noise and outliers propagating across the iterations.

Bidimensional Empirical Mode Decomposition (BEMD) presented in [3] is a method to decompose a 2D signal in its frequency modes. This work was applied to texture extraction and also 2D signal filtering. Their algorithm consists of the extraction of features at multiple scales or spatial frequencies. These features, called intrinsic mode functions, are extracted by sifting. The author performs this sifting using morphological operators to detect regional maxima and radial basis functions for surface interpolation. The author demonstrates the efficiency of their proposal with synthetic and natural images.

### *3.1.2 Optical Flow Filtering*

In [4], the authors integrated a Median filter in the optical flow model. Their proposal formalizes the Median filter with a new model that integrates OF estimation over a local neighborhood. In [5], the Median filter of size $3 \times 3$ is used to eliminate irregularities of the optical flow and also noisy estimation. In [6], to enhance the OF estimation, a bilateral filter is used. In that work, the authors replaced the anisotropic diffusion of their proposed model with a novel multi-cue-driven bilateral filter that considers the estimated occlusion.

Dérian et al. [7] utilizes another approach, where an optical flow model based on wavelet analysis is presented. The multi-resolution approach used in the optical flow estimation is similar to the multi-resolution used in wavelets analysis. The authors constructed a scale-space representation of the optical flow; furthermore, they provide a mechanism to locally approach the optical flow using high-order polynomials by trunking wavelets at fine scales. This methodology was not evaluated in a contemporaneous dataset but evaluated video sequences of moving fluids.

#### Motivation

In [8] an optical flow study is presented. This study consider the optical flow estimation using image pyramid and also considers a theoretical analysis of warping, but this study does not consider the study of intermediate filters. The study in [9] varies the number of warpings, the image pyramid levels, and the influence of parameters but does not take into account the intermediate filters. Those facts motivate us to compare the optical flow estimation performance considering different intermediate filters.

### *3.1.3 Contribution of this Work*

In this study, we use the optical flow estimation model proposed in [5]. The used model is based on the absolute value of the optical flow estimation error ($TV - L^1$)

and, thanks to other components, is robust to illumination changes and medium displacements. We present, in this work, the following contributions:

- (a) Evaluation of the OF performance estimation considering four intermediate filters: bilateral filter, median filter, weighted median filter, and a balanced weighted median filter.
- (b) We proposed an adaptive or weighted sum of the bilateral and the median filter.
- (c) We performed final evaluation in the complete MPI-Sintel test dataset and submitted the results to the MPI-Sintel benchmark web page [10].

In Sect. 3.2, we explain the principal strategies used to estimate the optical flow. We will briefly explain the linearization of the optical flow constraint, warping an image, and image pyramid. We explain these strategies in order to make our manuscript self-contained. In Sect. 3.3, we explain the filters considered in this work. In section 3.4, we explain experiments and used dataset. In Sect. 3.5, we present our obtained results and a brief discussion about other methodologies and our results. Finally, in Sect. 3.6 we present our conclusions and future work.

## 3.2 Preliminary

In order to state a model of the optical flow estimation, we consider two consecutive color images $I_0$ (reference image) and $I_1$ (target image), where $I_0$, $I_1 : \Omega \to \mathbb{R}^3$ and $\Omega$ a rectangular image domain; let $\mathbf{u} : \Omega \to \mathbb{R}^2$ be the optical flow between these two consecutive images (reference and target) where $\mathbf{u}(\mathbf{x}) = (u_1(\mathbf{x}), u_2(\mathbf{x}))$ has two components, that is to say, $u_1, u_2 : \Omega \to \mathbb{R}$. Optical flow estimation aims to determine a motion field $\mathbf{u}(\mathbf{x})$ such that $I_0(\mathbf{x})$ and $I_1(\mathbf{x} + \mathbf{u}(\mathbf{x}))$ are measures of the same pixel $\mathbf{x}$, i.e.:

$$I_0(\mathbf{x}) - I_1(\mathbf{x} + \mathbf{u}(\mathbf{x})) = 0. \tag{3.1}$$

Equation 3.1 in the literature is called the color constancy constraint. Equation 3.1 is highly non-linear and a linearization is applied around a known optical flow $\mathbf{u}_0(\mathbf{x})$.

### *3.2.1 Linearized Color Constancy Constraint*

Considering a known value $\mathbf{u_0}(\mathbf{x})$ of the optical flow, a Taylor expansion is valid:

$$I_0(\mathbf{x}) - I_1(\mathbf{x} + \mathbf{u}_0(\mathbf{x})) - \langle \nabla I_1(\mathbf{x} + \mathbf{u}_0(\mathbf{x})), \mathbf{u}(\mathbf{x}) - \mathbf{u}_0(\mathbf{x}) \rangle = 0, \tag{3.2}$$

with $\langle , \rangle$ being the scalar product, $I_1(\mathbf{x} + \mathbf{u}_0(\mathbf{x}))$ the warped image ($I_1(\mathbf{x})$) by a known optical flow $\mathbf{u}_0(\mathbf{x})$, and $\nabla I_1(\mathbf{x} + \mathbf{u}_0(\mathbf{x}))$ a gradient vector of $I_1(\mathbf{x} + \mathbf{u}_0(\mathbf{x}))$.

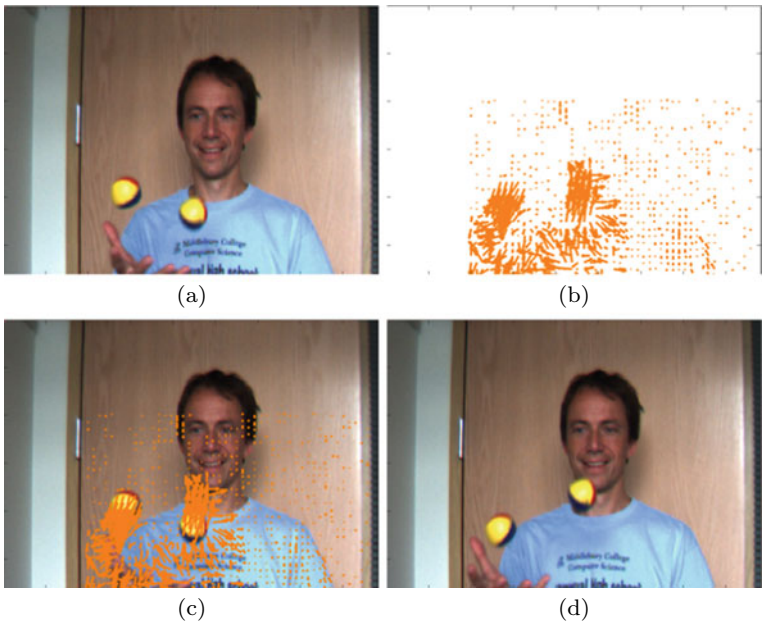


**Fig. 3.3** Warping performed in a picture extracted from [11]. **a** Original image $I(\mathbf{x})$. **b** Graphic representation of the optical flow using orange arrows $\mathbf{u_0}(\mathbf{x})$. **c** Optical flow superimposed over the original image. **d** Warped image by the optical flow or compensated image $I(\mathbf{x} + \mathbf{u_0}(\mathbf{x}))$

### 3.2.2 Image Warping

Image warping is a process of image manipulation such that it distorts any shape contained in the image. Given an image $I(\mathbf{x})$ and an optical flow $\mathbf{u}_0(\mathbf{x})$, we can warp the image $I(\mathbf{x})$ to obtain $I(\mathbf{x} + \mathbf{u_0}(\mathbf{x}))$ as we show in Fig. 3.3.

### 3.2.3 Image Pyramid

In large displacements, the Taylor approximation does not hold, and we use a coarse-to-fine strategy. This strategy constructs a multi-scale pyramid down-sampling images with a factor of 2. In Fig. 3.4, a scheme is shown of the image pyramid, considering four levels.

Figure 3.4 begins with the coarsest scale, and each of them doubles the image dimension in the previous one. At each level, the optical flow is filtered with an intermediate filter to eliminate noise and outliers.

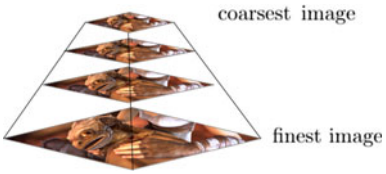


**Fig. 3.4** Image pyramid using an image of the sequence bandage. At the top, we have the coarsest image and the finest at the bottom level. The computed optical flow in each level is used as an initial condition for the optical flow estimation in the consecutive finer level

## 3.3 Intermediate Filters

Traditionally, OF methods perform the computation in an image pyramid. In each resolution, intermediate filter processing was applied to the analysis to eliminate estimation irregularities and noisy estimation. We explain the scheme using pseudo-code in algorithm 1 where $N_{\text{levels}}$ are the level numbers in the pyramid and $N_{\text{warpings}}$ is the warping number in each scale once the optical flow $\mathbf{u}(\mathbf{x})$ is computed. Intermediate filtering is applied in every iteration of warping Algorithm 1.

**Input** : Two consecutive frames $I_0$, $I_1$
**Output**: Optical flow $\mathbf{u}$
down-scaled images (image pyramidal) $I_0^s$, $I_1^s$ for $s = 1, \ldots, N_{\text{levels}}$;
Initialize $\mathbf{u}^{N_{\text{levels}}}$;
**for** $s \leftarrow N_{\text{levels}}$ **to** 1 **do**
  **for** $w \leftarrow 1$ **to** $N_{\text{warpings}}$ **do**
    Warp Image $I_1^s(\mathbf{x} + \mathbf{u}(\mathbf{x}))$;
    Compute gradient $\nabla I_1^s(\mathbf{x} + \mathbf{u}(\mathbf{x}))$;
    Compute Optical flow $\mathbf{u}^s$;
    Intermediate filtering;
  **end**
  **If** $s > 1$ **t**hen up-sampling $\mathbf{u}^s$ to $\mathbf{u}^{s-1}$;
**end**
$u = u^1$

**Algorithm 1:** Pseudo-code for a traditional optical flow.

### *3.3.1 Bilateral Filter*

The Bilateral filter can be represented as

$$u_{if}(\mathbf{x}) = \frac{1}{\sum_{\mathbf{y}\in\mathcal{N}(\mathbf{x})} w(\mathbf{x},\mathbf{y})} \sum_{\mathbf{y}\in\mathcal{N}(\mathbf{x})} w(\mathbf{x},\mathbf{y}) u_i(\mathbf{y}), \tag{3.3}$$

with $\mathcal{N}(\mathbf{x})$ a neighborhood around $\mathbf{x}$, $w(\mathbf{x}, \mathbf{y})$ are the exponential weights, and $u_i$ (with $i = 1, 2$) are the vertical and horizontal components of the optical flow:

$$w(\mathbf{x}, \mathbf{y}) = \phi_S(\mathbf{x} - \mathbf{y})\phi_I(I_0(\mathbf{x}) - I_0(\mathbf{y})), \tag{3.4}$$

where $\phi_S$ is the spatial distance between $\mathbf{x}$ and $\mathbf{y}$, and $\phi_I$ is the photo-metric distance between the pixels in $I_0(\mathbf{x})$ and $I_0(\mathbf{y})$. $\phi_S$ and $\phi_I$ are Gaussian kernels given by $\phi_S(\mathbf{x}) = e^{-\frac{\|\mathbf{x}\|^2}{2\sigma_S^2}}$ and, $\phi_I(I_0(\mathbf{x})) = e^{-\frac{\|I_0(\mathbf{x})\|^2}{2\sigma_I^2}}$ where $\sigma_S > 0$, $\sigma_I > 0$, and $I_0(\mathbf{x})$ is the reference image.

### 3.3.2 Median Filter

The Median filter is a non-linear filter, which is used to remove noise and outliers from the optical flow estimation. Given a neighborhood around $\mathbf{x}$ (let's say $\mathcal{N}(\mathbf{x})$),

$$u_{if}(\mathbf{x}) = median_{\mathbf{y}\in\mathcal{N}(\mathbf{x})}u_i(\mathbf{y}). \tag{3.5}$$

We considered square geometries to implement the Median filter, i.e., we filter a $3 \times 3$ square around each $\mathbf{x}$ point. Each component of the optical flow is filtered with this filter.

### 3.3.3 Weighted Median Filter

The Weighted median filter is a non-linear filter applied to a bidimensional distribution of weights and pixel intensities. Each pixel in a neighborhood $\mathcal{N}(\mathbf{x})$ has a weight. The goal is to sort the pixel intensity values, given the distribution of weights. In this case, for each pixel $\mathbf{x}$, we use bilateral weights. The following expression represents weights in each neighborhood:

$$w(\mathbf{x}, \mathbf{y}) = \phi_S(\mathbf{x} - \mathbf{y})\phi_I(I_0(\mathbf{x}) - I_0(\mathbf{y})). \tag{3.6}$$

We sorted both the weights $w$ and also the values in $u(\mathbf{x})$. The auxiliary variables are $s_w$ and $s_u$, representing weight values and optical flow values, respectively. We found in the array $s_w$ positions $p^*$ that hold

$$\sum_{i}^{p*} s_w(i) < \frac{1}{2}\sum_{i}^{N} s_w(i). \tag{3.7}$$

Finally, using the position $p^*$, we assign the weighted median filtered values of $u$:

$$u_{wf}(\mathbf{x}) = s_u(p^*). \tag{3.8}$$

### *3.3.4 Balanced Median Filter*

Our proposal is a weighted combination of the bilateral and the median. This linear combination has an adaptive weight $\alpha(\mathbf{x})$. This adaptive weight balances the contribution of the bilateral and the median filter in the intermediate filtering.

Let $u_{bl}(\mathbf{x})$ and $u_m(\mathbf{x})$ be the filtered estimated optical flow by the bilateral filter and the median filter, respectively. Following the ideas in [12], we constructed a balance weight:

$$\alpha(\mathbf{x}) = \frac{1}{1 + e^{(D_{bl}(\mathbf{x}) - D_m(\mathbf{x}))}}, \tag{3.9}$$

where $D_{bl}(\mathbf{x})$ is given as

$$D_{bl}(\mathbf{x}) = |I_0(\mathbf{x}) - I_1(\mathbf{x} + \mathbf{u}_{bl}(\mathbf{x}))|, \tag{3.10}$$

and $D_m(\mathbf{x})$ is given as

$$D_m(\mathbf{x}) = |I_0(\mathbf{x}) - I_1(\mathbf{x} + \mathbf{u}_m(\mathbf{x}))|. \tag{3.11}$$

These two terms, $D_{bl}(\mathbf{x})$ and $D_m(\mathbf{x})$, represent the error of the OF in the point $\mathbf{x}$ when the bilateral filter or the median filter are applied, respectively. The convex combination gives the combined optical flow:

$$\mathbf{u}(\mathbf{x}) = (1 - \alpha(\mathbf{x}))\mathbf{u}_m(\mathbf{x}) + \alpha(\mathbf{x})\mathbf{u}_{bl}(\mathbf{x}). \tag{3.12}$$

Depending on the values of $\alpha_i(\mathbf{x})$, the flow can be more confident in the median filter or the bilateral filter. Equation 3.9 shows that when $D_{bl} >> D_m$, $\alpha_i$ value is almost 0 and when $D_{bl} << D_m$, $\alpha$ is almost 1.

## 3.4 Dataset and Experiments

This section presents the performed experiments and the dataset used to evaluate optical flow estimation performance.

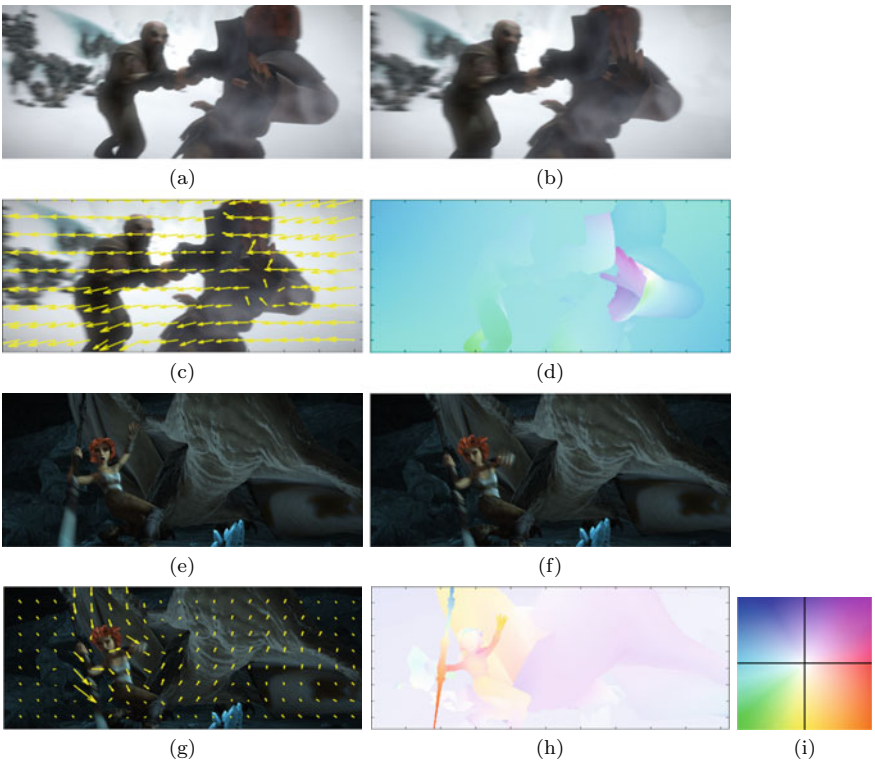


**Fig. 3.5** Examples of the MPI-Sintel video dataset. **a** frame_0001 and **b** frame_0002 of video sequence ambush_2. **c** Arrow representation of the ground truth OF of the video. **d** OF color-coded representation. **e** frame_0001 and **f** frame_0002 images of the video sequence cave_4. **g** available ground truth optical flow of two consecutive images. **h** Color-coded ground truth optical flow. **i** Used optical flow color code

### *3.4.1 Dataset*

The dataset contains different video sequences that present blur, fog, different illumination, and many scenes with large displacement and fast displacements. The dataset is divided into two subsets. One set is a training set, and the second one is a validation set. The training set is also divided into two subsets, one called clean and the other called final. The final stage considers the effects mentioned above. This set is more challenging than the clean one; therefore, we performed the experiments in this set. Figure 3.5 shows examples of video sequences in the MPI-Sintel dataset, and we offer the available ground truth optical flow with arrows and also using color code. We show the used color code in Fig. 3.5i.

**Table 3.1** The number of images in each image sequence in MPI-Sintel training set

| Sequence | Alley | Ambush | Bamboo | Bandage | Cave | Market | Mountain | Shaman | Sleeping | Temple |
|---|---|---|---|---|---|---|---|---|---|---|
| # of images | 100 | 174 | 100 | 100 | 100 | 140 | 50 | 100 | 100 | 100 |

In Table 3.1, we show a numerical description of each sequence of the final MPI-Sintel training set. In the MPI-Sintel dataset, the optical flow ground truth is available as shown in Fig. 3.5. Thus, this ground truth let us compute end-point-error ($EPE$) and angular-average-error ($AAE$); these errors are giving by the following expressions:

$$\begin{aligned} EPE &= \frac{1}{n}\sum_{i=1}^{n}\sqrt{(g_{1i}-u_{1i})^2+(g_{2i}-u_{2i})^2} \\ AAE &= \frac{1}{n}\sum_{i=1}^{n} cos^{-1}\left(\frac{1+g_{1i}u_{1i}+g_{2i}u_{2i}}{\sqrt{1+g_{1i}^2+g_{2i}^2}\sqrt{1+u_{1i}^2+u_{2i}^2}}\right). \end{aligned} \tag{3.13}$$

### *3.4.2 Experiments*

In the following, we explain experiments performed using the MPI-Sintel dataset.

(i) Evaluation using median filter
We evaluated the complete MPI-Sintel training set in the final version using the median filter of size $3 \times 3$ as an intermediate filter.

(ii) Evaluation using weighted median filter, median filter, and balanced median filter
We evaluated the complete MPI-Sintel training set in the final version using the weighted median filter considering $\sigma_I = 200$ and $\sigma_s = 200$.

(iii) Evaluation using Balanced weighted median filter also evaluated our proposed combined filter, which combines a weighted median filter and a bilateral filter.

(iv) Evaluation using a bilateral filter.

## 3.5 Results and Discussion

We have evaluated the filters in sequences that contain medium displacements, which are the sequences: market, mountain, shaman, sleeping, and temple. In these evaluation sets, we assessed $EPE$ and $AAE$ for these intermediate filters. In these experiments, we computed OF in the training set, which is around 490 images. We present in Table 3.2 second and third columns, the obtained numerical results. As a resume, we obtained an $EPE = 4.47$ and $AAE = 9.22$ and finally $EPE + AAE = 13.69$ by the Median filter. In Table 3.2 in the fourth and fifth column, we show results

**Table 3.2** Results were obtained by different proposed filters, EPE and AAE. The second and third columns show results obtained by the Median filter. In the fourth and fifth columns, results were obtained by the weighted Median filter, and in the sixth and seventh columns, results by Balanced weighted median filter

| Sequence Name | Median | Filter | Weighted | Median filter | Balanced | Median filter | Bilateral | Filter |
|---|---|---|---|---|---|---|---|---|
| | $EPE$ | $AAE$ | $EPE$ | $AAE$ | $EPE$ | $AAE$ | $EPE$ | $AAE$ |
| Market | 8.50 | 13.07 | 8.50 | 13.08 | 8.43 | 12.83 | 10.25 | 15.51 |
| Mountain | 0.95 | 10.15 | 0.95 | 10.16 | 0.95 | 9.88 | 1.32 | 11.24 |
| Shaman | 0.37 | 7.11 | 0.37 | 7.10 | 0.35 | 6.58 | 0.40 | 6.99 |
| Sleeping | 0.11 | 1.79 | 0.11 | 1.78 | 0.10 | 1.73 | 0.13 | 2.05 |
| Temple | 9.06 | 12.90 | 9.04 | 12.96 | 9.04 | 12.86 | 11.82 | 16.22 |
| Total $EPE$ | 4.47 | 9.22 | 4.47 | 9.23 | 4.44 | 8.99 | 5.56 | 10.71 |
| $EPE + AAE$ | 13.69 | | 13.70 | | 13.44 | | 16.27 | |

obtained in the MPI-Sintel data set using the Weighted Median filter. We obtained $EPE = 4.47$ and $AAE = 9.23$ and also $EPE + AAE = 13.70$. These results are worser than the results obtained by the Median filter.

In Table 3.2 sixth and seventh columns, that the $EPE + AAE = 13.44$. Comparing results obtained by the Median filter ($EPE + AAE = 13.69$) and Weighted Median ($EPE + AAE = 13.70$), and Balanced filter, we see that the obtained results are very similar. There are differences of 0.26 between the Weighted Median filter and the balanced median filter. We observe a small difference in the $AAE$ between the Median filter and balanced median filter. The Balanced median filter performs better than the median filter ($AAE = 8.99$). This result indicates that the estimated optical flow is better aligned w.r.t. the ground truth.

### 3.5.1 Comparison with Other Methods

We submitted the obtained results in the MPI-Sintel to the MPI-Sintel webpage. Those results were ranked in a benchmark that compares different OF method results. Figure 3.6 shows the obtained performance. In Fig. 3.6, we show the performance of our proposal. Our model using the combination of bilateral filter and weighted median filter (called TVL1_BWMFilter) outperforms classic methods like Horn-Schunck [1]. Our proposal presents an $EPE = 9.034$ outperforming the $TV - L^1$ classic formulation [13] and the non-local optical flow in Classic+NL [4]. Our proposal performs similar to Motion Detail Preserving Optical [12] flow, which considers additional correspondences, giving hints to guide the OF estimation. Figure 3.7 is shown with some examples of obtained results by our method.

| | EPE all | EPE matched | EPE unmatched | d0-10 | d10-60 | d60-140 | s0-10 | s10-40 | s40+ |
|---|---|---|---|---|---|---|---|---|---|
| GeoFlow [250] | 8.459 | 3.908 | 45.553 | 5.805 | 3.550 | 2.899 | 1.398 | 4.667 | 52.653 |
| FlowNetADF [255] | 8.532 | 4.768 | 39.222 | 8.002 | 4.979 | 3.060 | 1.846 | 6.304 | 47.067 |
| CPNFlow [256] | 8.589 | 4.575 | 41.327 | 7.234 | 4.542 | 2.986 | 1.856 | 5.484 | 49.511 |
| Back2FutureFlow_UFO [263] | 8.814 | 5.031 | 39.647 | 7.153 | 4.880 | 3.904 | 1.752 | 5.961 | 50.725 |
| TV-L1+EM [268] | 8.916 | 4.712 | 43.157 | 7.021 | 4.788 | 3.401 | 1.456 | 5.188 | 54.932 |
| TVL1_BWMFilter [269] | 9.034 | 4.761 | 43.824 | 7.245 | 4.931 | 3.286 | 1.450 | 4.998 | 56.396 |
| LDOF [270] | 9.116 | 5.037 | 42.344 | 6.849 | 4.928 | 4.003 | 1.485 | 4.839 | 57.296 |
| Classic+NL [281] | 9.153 | 4.814 | 44.509 | 7.215 | 4.822 | 3.427 | 1.113 | 4.496 | 60.291 |
| Horn+Schunck [269] | 9.610 | 5.419 | 43.734 | 7.950 | 5.658 | 3.976 | 1.882 | 5.335 | 58.274 |

**Fig. 3.6** Obtained results by our proposal in MPI-Sintel benchmark


**Fig. 3.7** Examples of flow estimation in MPI-Sintel test set. **a** frame_0024 and **b** frame_0025 of sequence PERTURBED_shaman_1. **c** estimated optical flow for sequence PERTURBED_shaman_1. **d** image frame_0041 of sequence tiger1. **e** image frame_0042 of sequence tiger1. **f** estimated optical flow for sequence tiger_1

In Fig. 3.7c and f is shown obtained OF for images of the sequence PERTURBED_shaman_1 and tiger1 of the MPI-Sintel test dataset. In these two images of the sequence PERTURBED_shaman_1, we obtained a $EPE = 1.648$ (Fig. 3.7a and b), and in the two images of sequence tiger1 (Fig. 3.7d and e) we obtained $EPE = 1.678$.

### *3.5.2 Bidimensional Empirical Mode Decomposition*

We have filtered the estimated optical flow using Bidimensional Empirical Mode Decomposition (BEMD) as a proof of concept. We used a MATLAB implementation available on the web. In Fig. 3.8, we show a video sequence where a dragon runs following a chicken. We estimated the optical flow, and we extracted the BEMD.

We observe in Fig. 3.8d the edges of estimated optical flow in (c), and in Figure (f) intermediate spatial frequencies are shown. In Fig. 3.8, low frequencies are shown,

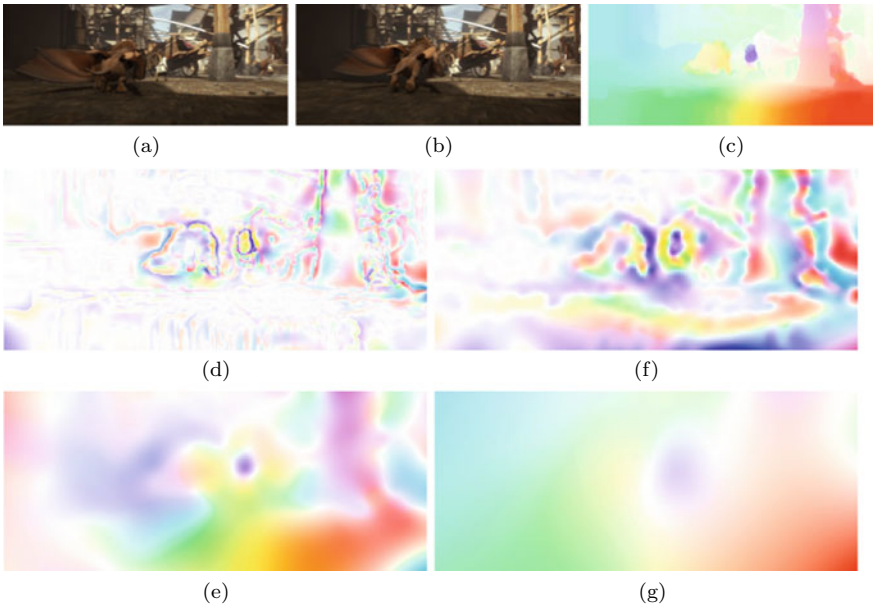


**Fig. 3.8** Bidimensional Empirical Mode Decomposition. **a** and **b** two images of the video sequence market_6. In **c**, we show the color-coded estimated optical flow. **d** and **f** BEMD showing highest spatial frequencies and intermediate-high frequencies, respectively. **e** BEMD showing low frequencies. **g** lowest spatial frequencies

but the edges were not preserved, i.e., the EMD performs as an anisotropic filter. Finally, in (f), we observe the continuum component of the flow, which is very blurred and does not preserve shapes.

The MATLAB implementation of the BEMD decomposition method runs four iterations in 1435 seconds, which is not suitable for real time.

### *3.5.3 Processing Time*

We measured the total processing time at each scale, which depended on the image size. That is, at different scales, we have different processing times. The measure was performed on a Laptop MSi-i7, running on a single core (please see Table 3.3).

### *3.5.4 Discussion*

Regarding the BEMD method, we observe in Fig. 3.8f that the method performs similar to the Gaussian filter, i.e., it blurs the image. Edge preserving in optical flow

**Table 3.3** Processing time at each scale—Balanced filter

| Image size | Time processing |
|---|---|
| 256 × 109 | 0.005186 s |
| 512 × 218 | 0.016479 s |
| 1024 × 436 | 0.101669 s |

estimation is an essential feature that a method should have. We think that a minor modification of BEMD decomposition is possible in order to preserve edges. As a future work, we will consider modifying the BEMD method to consider anisotropic morphological operators. And then evaluate its effect on the optical flow estimation.

Concerning another approach to optical flow estimation as in [7], the author didn't evaluate their proposal in the standard dataset such as MPI-Sintel [10] or [11]. Also, there is no available code in order to compare our results with their proposal. Considering implementing the proposal in [7], as future work, we will consider assessing this proposal with ours in a standard dataset such as MPI-Sintel.

Concerning the obtained results, we were comparing the columns in Table 3.2 associated with the Median filter and Balanced median filter. We observe for sequence shaman that the average angular error dropped from 7.11 to 6.58, which is 0.43 degrees, and EPE dropped from 0.37 to 0.35 with 0.02 pixels. We observe that the most significant reduction was in the angular error, which means that the optical flow is better aligned with the ground truth for this sequence. We highlight these results because some of these sequences contain large displacement or minimal displacements, and shaman contains medium displacements. The proposal is better suited for small and medium displacements.

## 3.6 Conclusions

To perform our intermediate filter study, we have used a robust model for illumination changes and can handle large displacements. With this model, we assessed the performance of these filters: Median, Weighted Median, Balanced Median filter, and bilateral filter. We evaluated the performance in the MPI-Sintel dataset and submitted our results to a benchmark webpage. We obtained that the Balanced Median filter outperforms the other three filters; thus, we process the MPI-Sintel test. The obtained results show that our proposal outperforms the classical Horn-Schunck method and TV-L1 model and other models, robust to large displacements (LDOF) or non-local methods (Classic+NL). It also outperforms current methods such as GeoFlow [14] and CPNFlow [15]. In future work, we can investigate the uses of the BEMD decomposition method. However, with an anisotropic consideration to preserve edges and shapes and as future work, we will consider implementing the wavelet optical flow

to assess this proposal and our proposal in a standard dataset such as MPI-Sintel to compare performance.

## References


1. Horn, B.K., Schunck, B.H.: Determining optical flow. Artif. Intell. **17**, 185–203 (1981)
2. Zach, C., Pock, T., Bischof, H.: A duality based approach for realtime TV-L1 optical flow. In: Hamprecht, F.A., Schnörr, C., Jähne, B. (eds.) Pattern Recognition. DAGM 2007. Lecture Notes in Computer Science, vol. 4713. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978_3_540_74936_3_22
3. Nunes, J., Bouaoune, Y., Delechelle, E., Niang, O., Bunel, Ph.: Image analysis by bidimensional empirical mode decomposition. Image Vis. Comput. **21**(12), 1019–1026 (2003)
4. Sun, D., Roth, S., Black, M.J.: Secrets of optical flow estimation and their principles. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 2432–2439 (2010)
5. Lazcano, V.: Study of specific location of exhaustive matching in order to improve the optical flow estimation. In: 15th ITNG, April 2018, Las Vegas, Nevada, pp. 603–661 (2018)
6. Xiao, J., Cheng, H., Sawhney, H., Rao, C., Isnardi, M.: Bilateral Filtering-based flow estimation with occlusion detection. In: Proceedings of the ECCV, pp. 221–224 (2006)
7. Dérian, P., Héas, P., Herzet, C., Mémin, E.: Wavelets and optical flow motion estimation. Numer. Math. J. Chin. Univ. Nanjing Univ. Press **2013**(6), 116–137 (2003)
8. Brox, T., Bruhn, A., Papenberg, N., Weickert, J.: High accuracy optical flow estimation based on a theory for warping. In: Pajdla, T., Matas, J. (eds.) Computer Vision–ECCV 2004. Lecture Notes in Computer Science, vol. 3024, pp. 25–36. Springer, Berlin (2004)
9. Meinhardt-Llopis, E., Sánchez-Pérez, J., Kondermann, D.: Horn-Schunck optical flow with a multi-scale strategy. Image Process. Line **3**, 151–172 (2013). https://doi.org/10.5201/ipol.2013.20
10. Butler, D.J., Wulff, J., Stanley, G.B., Black, M.J.: A naturalistic open source movie for optical flow evaluation. In: Fitzgibbon, A., et al. (eds.) European Conference on Computer Vision (ECCV). Part IV, LNCS 7577, Oct, pp. 611–625. Springer (2012)
11. Baker, S., Scharstein, D., Lewis, J.P., Roth, S., Black, M.J., Szeliski, R.: A database and evaluation methodology for optical flow. IJCV **92**(1), 1–31 (2011)
12. Xu, L., Jia, J., Matsushita, Y.: Motion detail preserving optical flow. In: IEEE CVPR (2010)
13. Wedel, A., Pock, T., Zach, C., Bischof, H., Cremers, D.: An improved algorithm for TV-L1 optical flow. In: Statistical and Geometrical Approaches to Visual Motion Analysis. LNCS, vol. 5604 (2009)
14. Mei, L., Chen, Z., Lai, J.: Geodesic-based probability propagation for efficient optical flow. Electron. Lett. **54**(12), 758–760,14,6, (2018). https://doi.org/10.1049/el.2018.0394
15. Yang, Y., Soatto, S.: Conditional prior networks for optical flow. In: The European Conference on Computer Vision (ECCV), pp. 271–287 (2018)