Estimation de flot optique basé évènements en temps réel

Vincent Brebion, Julien Moreau, Franck Davoine

▶ **To cite this version:**

Vincent Brebion, Julien Moreau, Franck Davoine. Estimation de flot optique basé évènements en temps réel. Congrès Reconnaissance des Formes, Image, Apprentissage et Perception (RFIAP 2022), Jul 2022, Vannes, France. hal-03690073

HAL Id: hal-03690073
https://hal.science/hal-03690073v1

Submitted on 7 Jun 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Estimation de flot optique basé évènements en temps réel

Vincent Brebion Julien Moreau Franck Davoine

Université de technologie de Compiègne (UTC), CNRS, Heudiasyc
CS 60 319 — 60 203 Compiègne Cedex, France

{vincent.brebion, julien.moreau, franck.davoine}@hds.utc.fr

Résumé

En capturant de manière asynchrone les changements de luminosité dans la scène, les caméras à évènements apportent un changement de paradigme important dans le monde de la vision. De par leurs propriétés, elles ouvrent la porte vers de nouvelles applications à très faible latence. Dans cet article, nous nous intéressons au problème du flot optique basé évènements. Nous proposons une nouvelle méthode pour le résoudre en temps réel, aussi bien avec des caméras à évènements de basse que de haute définition. À travers son évaluation, nous montrons qu'elle produit régulièrement de meilleurs résultats que l'état de l'art actuel, tout en atteignant de plus hautes fréquences de fonctionnement : 250Hz pour une résolution de 346×260 pixels, et 77Hz en 1280×720 pixels.

Mots Clef

Caméras neuromorphiques, flot optique, latence basse.

Abstract

By capturing asynchronously changes of illumination in the scene, event cameras constitute a major paradigm shift in the world of computer vision. Thanks to their properties, they allow for new low-latency applications. In this paper, we are interested by the issue of event-based optical flow. We propose a novel real-time approach to solve it, compatible with both low- and high-definition event cameras. Through its evaluation, we show that our approach often produces better results than the current state of the art, while reaching higher output rates: 250Hz at 346×260 pixels and 77Hz at 1280×720 pixels.

Keywords

Neuromorphic cameras, optical flow, low latency.

1 Introduction

L'estimation du mouvement apparent d'une scène visuelle, appelé flot optique, est un problème central en vision par ordinateur. Celui-ci constitue un composant clé pour de nombreuses applications : détection et suivi d'objets [4], segmentation d'images [8], détection et estimation du mouvement [10], et odométrie visuelle [15].

Le flot optique est communément estimé à partir de caméras « traditionnelles » basées images, de par leur prévalence et l'avancement de leur état de l'art. De par leur fonctionnement reposant sur un temps d'exposition du capteur pour intégrer la lumière, ces caméras présentent cependant des limitations en cas de déplacements importants (flou de mouvement), ou encore en présence de niveaux de luminosité variables (sous- ou sur-exposition), pouvant rendre complexe l'estimation du flot optique.

L'arrivée nouvelle de caméras neuromorphiques, appelées « caméras à évènements », vise à pallier ces défauts, en proposant une capture asynchrone des changements de luminosité dans la scène. Cependant, à cause de leur récente émergence et de leur fonctionnement fondamentalement différent de celui des caméras basées images, l'état de l'art et le déploiement de ces caméras reste aujourd'hui encore relativement limité.

En ce qui concerne le flot optique basé évènements en particulier, les approches proposées peuvent à ce jour être classées selon trois catégories : (i) certains auteurs considèrent que les évènements constituent un nuage de points spatio-temporel, et cherchent à résoudre le problème du flot optique par du *plane-fitting* notamment [2]; (ii) d'autres, au contraire, font le choix d'accumuler les évènements sur de courtes périodes de temps, afin de pouvoir se rapprocher des méthodes traditionnelles basées images [16]; et (iii) plus récemment, certains auteurs cherchent à résoudre le problème du flot optique basé évènements via l'utilisation de réseaux de neurones profonds [12, 14].

À travers cet article, nous proposons une nouvelle méthode pour l'estimation du flot optique à l'aide de caméras à évènements. Nous proposons un processus optimisé ainsi qu'une amélioration et extension notable aux travaux d'Almatrafi *et al.* [3], qui introduisaient une approche simple et efficace pour densifier incrémentalement les évènements épars en des représentations basées images. Nos enjeux sont doubles : rendre l'approche compatible avec des caméras à évènements haute définition, et lui permettre de s'exécuter en temps réel en ne sacrifiant pas la qualité de ses résultats.

Concernant l'évaluation de notre méthode, nous nous intéressons notamment aux scènes de conduite, où les pro-

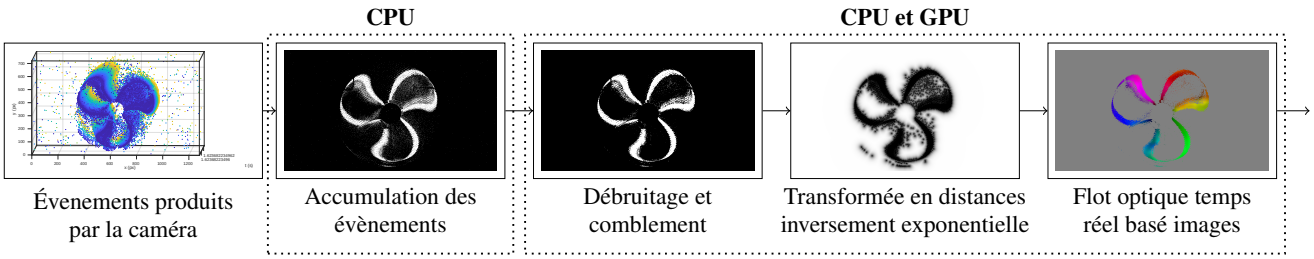

FIGURE 1 – Schéma de notre architecture « RTEF » de calcul de flot optique basé évènements. Chaque bloc est illustré par le résultat qu'il fournit en sortie, pour une séquence exemple de pales de ventilateur tournant devant la caméra.

blématiques de flou de mouvement et de sous- et de sur-exposition sont particulièrement présentes.
Nous mettons également à disposition le code source, consultable à l'adresse suivante : https://github.com/vbrebion/rt_of_low_high_res_event_cameras.

2 Principe de la vision évènementielle

Une caméra neuromorphique produit, pour chacun de ses pixels $\mathbf{u} = (x, y)$, un évènement si le logarithme de l'intensité lumineuse $I^{\mathbf{u}}$ que ce pixel reçoit varie sur une période de temps au-delà d'un seuil C donné :

$$|\log(I_t^{\mathbf{u}}) - \log(I_{t-1}^{\mathbf{u}})| \geq C \tag{1}$$

Chaque évènement e est un petit paquet d'information, indiquant le pixel $\mathbf{u}$ et l'instant t auquel ce changement de luminosité s'est produit, ainsi que sa polarité p (avec $p = \text{sign}(\log(I_t^{\mathbf{p}}) - \log(I_{t-1}^{\mathbf{p}}))$, $+1$ en cas d'augmentation de luminosité, -1 en cas de diminution) :

$$e = \{\mathbf{u}, t, p\} \tag{2}$$

Dans le cadre de cet article, nous dénotons E le flux d'évènements produit par une caméra neuromorphique :

$$E = \{e_i\} \tag{3}$$

3 Architecture proposée

Pour résoudre le problème du flot optique basé évènements, nous proposons l'utilisation d'une architecture en pipeline. Cela nous permet de paralléliser les opérations successives transformant les évènements épars d'entrée en des représentations 2D et calculant le flot optique final. Cette architecture, nommée « RTEF » (pour « Real-Time Event-based Flow ») est illustrée en figure 1, et est décrite dans les sous-sections suivantes.

3.1 Accumulation des évènements

Le premier module de l'architecture proposée est responsable du découpage du flux d'évènements E sur de courtes fenêtres temporelles successives de taille ΔT. Pour chacune de ces fenêtres, les évènements sont ensuite replacés au sein d'une image binaire, où chaque pixel de cette image est marqué si au moins un évènement a été produit pour ce pixel dans la fenêtre de temps concernée.
Ce processus d'accumulation permet d'obtenir une « image de contours » des objets en mouvement de la scène, servant de première représentation stable à partir des évènements afin de calculer le flot optique.

3.2 Débruitage et comblement

De par leur fonctionnement interne, les caméras neuromorphiques génèrent naturellement une quantité significative de bruit, c'est-à-dire d'évènements ne provenant pas de changements réels de luminosité dans la scène. La présence de ce bruit dans les images de contours nuit à leur stabilité, et impacte négativement le calcul du flot optique.
Bien que l'utilisation d'une solution de débruitage telle que celle proposée par les auteurs de [7] permettrait de séparer le bruit des évènements réels avant leur accumulation, cette opération serait coûteuse à cause de la nature éparse et asynchrone des évènements à cette étape.
La solution que nous proposons donc part de l'observation que les évènements de bruit correspondent à des pixels isolés dans l'image de contours. Pour les supprimer, nous éliminons donc tous les pixels de l'image de contours ayant moins de N_d pixels marqués dans leurs quatre voisins directs. Bien que cette méthode tende à retirer également du détail de l'image de contours, elle a en réalité pour avantage d'en garder les contours prédominants les plus stables pour le calcul du flot optique.
Une étape de comblement est également ajoutée, dans l'optique de renforcer la stabilité des images de contours. De manière similaire à l'étape de débruitage, l'objectif est ici de marquer les pixels de l'image débruitée qui ont plus de N_c voisins marqués, c'est-à-dire les pixels pour lesquels un évènement aurait probablement dû être produit par la caméra.
Ces deux étapes sont décrites plus en détails sous forme de pseudo-code dans les Algorithmes 1 et 2.
Il peut être noté en particulier que nos méthodes de débruitage et de comblement s'approchent des opérateurs classiques d'érosion et de dilatation morphologique d'images. L'utilisation des seuils N_d et N_c, cependant, nous permet d'adapter au mieux l'intensité de ces opérations face à la prévalence du bruit ou d'évènements manquants dans la scène observée.

Entrées : Une image de contours I
Le seuil de débruitage N_d
Sortie : L'image de contours débruitée I_d
$I_d \leftarrow I$;
pour chaque *index de pixel* $p \in I$ **faire**
  si $I[p]$ *est un pixel marqué* **alors**
    $n_d \leftarrow$ nombre de pixels marqués parmi les 4 pixels voisins directs de p dans I;
    si $n_d < N_d$ **alors**
      $I_d[p]$ n'est plus marqué;
    fin
  fin
fin

Algorithme 1 : Débruitage

Entrées : Une image de contours débruitée I_d
Le seuil de comblement N_c
Sortie : L'image de contours débruitée et comblée I_{dc}
$I_{dc} \leftarrow I_d$;
pour chaque *index de pixel* $p \in I_d$ **faire**
  si $I_d[p]$ *n'est pas un pixel marqué* **alors**
    $n_c \leftarrow$ nombre de pixels marqués parmi les 4 pixels voisins directs de p dans I_d;
    si $n_c \geq N_c$ **alors**
      $I_{dc}[p]$ devient un pixel marqué;
    fin
  fin
fin

Algorithme 2 : Comblement

3.3 Transformée en distances inversement exponentielle

Les images de contours, même après avoir été nettoyées, ne constituent que des images binaires, à partir desquelles il est difficile de calculer le flot optique. Afin de pallier ce problème, une étape de densification est ajoutée, sous la forme d'une transformée en distances, comme proposé initialement par Almatrafi *et al.* [3].
Cette approche, cependant, possède deux problèmes principaux, dûs au fait que la portée d'un pixel ne soit pas limitée lors de la transformée en distances : (i) la présence d'un seul pixel de bruit non-filtré peut perturber grandement l'apparence de l'image de distances; et (ii) les objets proches et/ou fortement texturés deviennent difficilement distinguables. Une illustration de ces problèmes est donnée en figure 2 (deux premières colonnes).
Afin de résoudre ces deux problèmes conjointement, nous proposons l'emploi d'une nouvelle formulation « inversement exponentielle » pour la transformée en distances :

$$d_{\text{exp}} = 1 - \exp(-d_{\text{euc}}/\alpha), \tag{4}$$

où d_{exp} est notre distance inversement exponentielle, d_{euc} est la distance euclidienne au pixel de contour le plus proche (i.e., la valeur de la transformée en distance originelle), et α un paramètre ajustable contrôlant l'aire d'influence de chaque pixel.

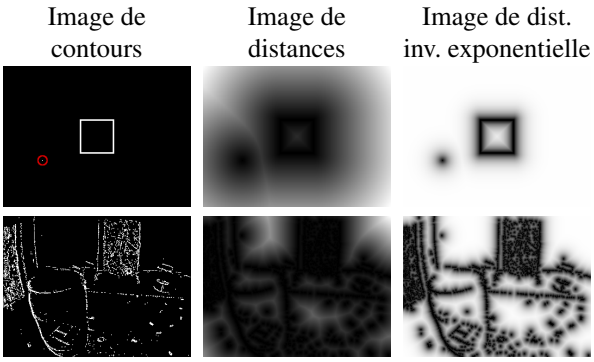

FIGURE 2 – Comparaison entre la transformée en distances originelle et notre formulation inversement exponentielle (avec $\alpha = 2$). La première ligne donne l'exemple d'un carré avec un unique pixel de bruit (entouré en rouge), tandis que la seconde est une scène réelle provenant du dataset MVSEC [17].

Une comparaison entre la transformée en distances originelle et notre formulation inversement exponentielle est fournie en figure 2. Celle-ci permet de visualiser notamment la réduction de l'impact du bruit, ainsi que la plus grande distinguabilité des textures de la scène.

3.4 Flot optique temps réel basé images

Les étapes décrites précédemment ayant mené à l'obtention d'images de distances denses, celles-ci peuvent être utilisées avec n'importe quelle méthode de l'état de l'art permettant le calcul de flot optique.
Dans le cadre de cet article, nous avons opté pour l'utilisation de l'approche proposée par Adarve *et al.* [1]. Ce choix a été motivé par trois raisons complémentaires : (i) leur utilisation d'une mémoire temporelle pour le flot optique, permettant de stabiliser les résultats même en cas de légères instabilités au niveau de la représentation d'entrée; (ii) la justesse des résultats du flot optique, proches de l'état de l'art; et (iii) la mise à disposition d'un code exécutable, capable de traiter jusqu'à 300 images haute définition par seconde.
Il est à noter que, bien que cette méthode retourne un flot optique dense, nous faisons le choix de le limiter aux seuls pixels marqués de l'image de contour, les autres pixels ne contenant que de l'information « reconstruite » suite à la densification de la transformée en distances.

3.5 Implémentation

Afin de respecter la contrainte temps réel imposée, et grâce à l'accumulation des évènements sous forme d'images, les modules de débruitage/comblement, de transformée en distances, et de flot optique ont pu être implémentés pour s'exécuter sur le GPU (comme indiqué sur la figure 1).
En ce qui concerne le module de transformée en distances, en particulier, nous avons choisi d'utiliser l'algorithme pro-

posé par Coeurjolly *et al.* [6]. Ce choix a notamment été guidé par son fonctionnement en deux passes, calculant les distances sur chaque ligne indépendamment dans un premier temps, puis les ajustant sur chaque colonne indépendamment dans un second temps. Chacune de ces deux passes est donc hautement parallélisable, et elles ont donc été implémentées sur GPU pour en tirer profit.

4 Résultats

Suite à la présentation détaillée de notre architecture, nous introduisons dans cette section les évaluations conduites ainsi que les résultats obtenus. Si le lecteur est intéressé, d'autres évaluations et résultats plus complets sont donnés dans l'article [5].

4.1 Métriques

Dans le cadre de l'évaluation de notre méthode, trois métriques différentes sont utilisées dans cet article.
Les deux premières sont des métriques classiques pour l'évaluation du flot optique : l'*Average Enpoint Error* (AEE) et le pourcentage d'intrus, toutes deux utilisées par exemple dans le cadre du benchmark KITTI [9]. L'AEE est une mesure brute de l'erreur sur le flot optique :

$$\text{AEE} = \frac{1}{N}\sum_{i=1}^{N}|v_i - u_i|, \tag{5}$$

où N est le nombre total de vecteurs de flot optique, u_i le $i^{\text{ème}}$ vecteur de flot estimé, et v_i le vecteur vérité terrain correspondant. Le pourcentage d'intrus, de son côté, dénote le pourcentage de vecteurs de flot optique pour lesquels l'AEE est supérieure à 3 pixels.
Ces deux métriques supposent toutefois l'existence d'une vérité terrain sur le flot optique. Afin de pouvoir exploiter des jeux de données n'en contenant pas, nous adoptons donc également la *Flow Warping Loss* (FWL) proposée par Stoffregen *et al.* [14]. Celle-ci évalue la netteté de l'image d'évènements accumulés, compensée à l'aide du flot optique, par rapport à sa version non-compensée. Elle est calculée comme suit :

$$\text{FWL} = \frac{\sigma^2(I_{\text{comp}})}{\sigma^2(I_{\text{n\_comp}})}, \tag{6}$$

où σ^2 représente la variance de l'image, I_{comp} l'image d'évènements compensée à l'aide du flot optique, et $I_{\text{n\_comp}}$ l'image originelle non-compensée. Ainsi, une valeur pour la FWL supérieure à 1 indique que le flot optique a permis de compenser correctement le mouvement des évènements.
Afin de démontrer la validité de nos contributions, deux versions alternatives de notre méthode sont également évaluées :

- RTEF_{SDC} – méthode complète avec débruitage et comblement désactivés;
- RTEF_{TDL} – utilisation de la transformée en distances linéraire originelle.

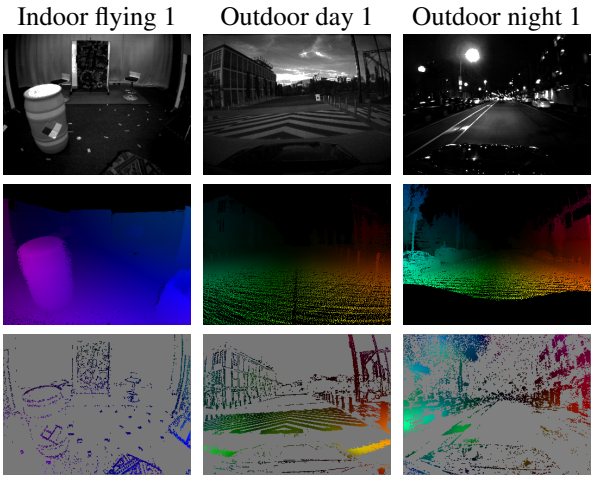

FIGURE 3 – Résultats qualitatifs sur le dataset MVSEC. De haut en bas : l'image de référence, la vérité terrain sur le flot optique, et notre flot optique.

4.2 Évaluations basse définition

Pour l'évaluation de notre méthode sur des données basse définition, nous avons utilisé le dataset MVSEC [17] (346×260). Il s'agit d'un jeu de données populaire, composé de scènes de vol en intérieur et de scènes de conduite, et qui comprend une vérité terrain sur le flot optique. Pour nous conformer à l'évaluation menée par les autres méthodes de l'état de l'art, nous avons fixé le temps d'accumulation des évènements à $\Delta T = 1$ frame. Nous avons également fixé empiriquement $N_d = 1$, $N_c = 4$, et $\alpha = 1.08$, pour compenser la présence importante de bruit dans les enregistrements.
Les résultats obtenus, présentés en tableau 1, démontrent la justesse de notre approche. Ceux-ci sont en effet de manière constante proches ou meilleurs que ceux produits par le réseau état de l'art $\text{EV-FlowNet}_{\text{HQF}}$ de Stoffregen *et al.* [14]. Nos résultats sont aussi et surtout bien meilleurs que ceux de FireFlowNet [12], qui constitue à ce jour la référence en terme de flot optique temps réel basé évènements.
Il peut être observé en revanche que notre méthode atteint ses limites sur les séquences de nuit, pour lesquelles l'erreur sur le flot optique augmente significativement. Cette hausse peut s'expliquer par une association de deux facteurs : (i) un temps d'accumulation imposé par le dataset plus important pour les séquences de nuit ($\Delta T \simeq 97$ms) comparé à celles de jour ($\Delta T \simeq 22$ms); et (ii) la présence d'éclairages urbains, générant un bruit important dans les évènements, et pour lequel notre méthode de débruitage simple est moins adaptée que des solutions plus complexes.
Une illustration des résultats obtenus sur chaque type de séquence du dataset est fournie en figure 3.
En ce qui concerne la comparaison avec les versions alternatives de notre méthode, il est possible de constater que la version utilisant la transformée en distances

TABLE 1 – Résultats obtenus sur le dataset MVSEC

Séquence	Indoor flying 1		Indoor flying 2		Indoor flying 3		Outdoor day 1		Outdoor day 2		Outdoor night 1		Outdoor night 2		Outdoor night 3	
	AEE	% intrus	AEE	% intrus	AEE	% intrus	AEE	% intrus	AEE	% intrus	AEE	% intrus	AEE	% intrus	AEE	% intrus
EV-FlowNet$_{HQF}$ [14]	0.56	-	**0.66**	-	**0.59**	-	0.68	-	0.82	-	-	-	-	-	-	-
FireFlowNet [12]	0.97	2.6	1.67	15.3	1.43	11.0	1.06	6.6	-	-	-	-	-	-	-	-
RTEF	0.52	**0.1**	0.98	5.5	0.71	2.1	**0.53**	**0.2**	**0.74**	**1.2**	**2.91**	**30.6**	**3.45**	**39.1**	**3.62**	**39.8**
RTEF$_{TDL}$	1.81	16.4	2.54	26.4	1.95	18.2	2.12	21.7	1.30	8.7	4.04	45.8	4.78	55.6	5.10	58.7
RTEF$_{SDC}$	**0.49**	**0.1**	0.92	**4.6**	0.68	**1.6**	0.54	0.4	0.75	1.3	2.99	31.8	3.56	40.4	3.70	40.9

TABLE 2 – Résultats (FWL) obtenus sur le dataset « 1 Megapixel Automotive Detection » de Prophesee

Séquence	30 Jan.	15 Fév.	18 Fév.	19 Fév.	21 Fév.	22 Fév.	12 Avr.	18 Avr.	11 Juin	14 Juin	17 Juin	19 Juin	21 Juin	26 Juin	*Global*
RTEF	**1.63**	**1.47**	**1.34**	**1.64**	**1.36**	**1.51**	1.14	**1.54**	**1.80**	**1.35**	**1.46**	**1.38**	**1.54**	**1.56**	**1.46**
RTEF$_{TDL}$	1.44	1.40	1.32	1.41	1.29	1.38	**1.21**	1.27	1.50	1.28	1.27	1.30	1.32	1.42	1.34
RTEF$_{SDC}$	1.43	1.37	1.25	1.52	1.28	1.41	1.08	1.43	1.63	1.27	1.35	1.31	1.46	1.46	1.37

linéaire (RTEF$_{TDL}$) produit les pires résultats, démontrant bien l'apport de notre transformée en distances inversement exponentielle. En ce qui concerne la version sans débruitage ni comblement (RTEF$_{SDC}$), celle-ci fournit sur les séquences « Indoor » des résultats légèrement meilleurs, l'éclairage de ces scènes étant particulièrement bien contrôlé, et peu de bruit étant donc présent. En revanche, pour les scènes de conduite en extérieur, le bruit devient beaucoup plus important, et notre débruitage permet donc de stabiliser les résultats.

4.3 Évaluations haute définition

Même si les caméras à évènements basse définition constituent la vaste majorité des modèles disponibles aujourd'hui, des caméras à évènements haute définition ont commencé à être commercialisées depuis peu. L'évaluation de notre méthode de flot optique sur ces capteurs haute définition nous apparaît donc comme nécessaire.

Dans le cadre de cet article, nous utilisons le dataset « 1 Megapixel Automotive Detection » [13] de Prophesee (1280×720). Bien qu'il ait été originellement prévu pour des applications de détection et de reconnaissance d'objets dans des séquences de conduite, et qu'il ne contienne donc pas de vérité terrain sur le flot optique, ce dataset constitue à ce jour le jeu de données basé évènements haute définition le plus complet disponible publiquement. Étant donné sa densité (1,2 To, 14 heures de données), nous en utilisons uniquement les séquences de « test », représentant plus de deux heures de données.

En adéquation avec la dynamique des scènes représentées, nous avons utilisé pour notre évaluation une fenêtre temporelle courte $\Delta T = 15$ms. Nous avons également fixé empiriquement $N_d = 2$, $N_c = 3$, et $\alpha = 1.08$, permettant de limiter l'influence du bruit tout en conservant les contours principaux des objets de la scène.

Les résultats, présentés en tableau 2, soulignent la justesse de notre méthode, tout comme pour l'évaluation basse définition. Nous obtenons en effet des valeurs de FWL grandement supérieures à 1 sur toutes les séquences, et une valeur moyenne pour la FWL de 1.46 sur les deux heures de séquences.

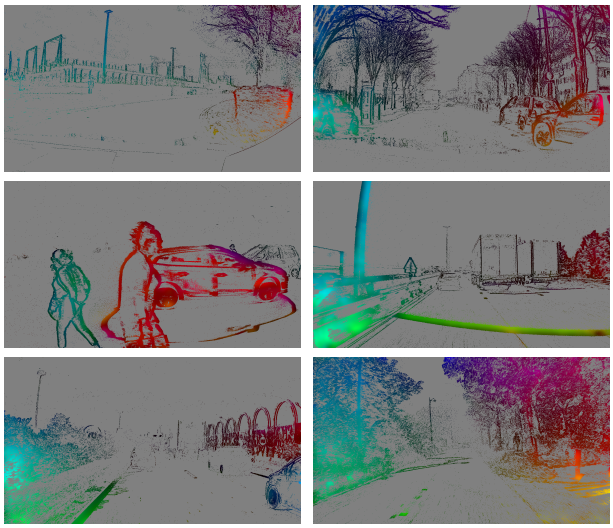

FIGURE 4 – Résultats qualitatifs sur le dataset « 1 Megapixel Automotive Detection ». Ligne par ligne, de gauche à droite : 19 Fév., 22 Fév.; 18 Avr., 11 Juin; 17 Juin, 21 Juin.

En ce qui concerne la comparaison avec les versions alternatives de notre méthode, que ce soit celle avec la transformée en distances linéaire RTEF$_{TDL}$ ou celle sans l'utilisation du débruitage et du comblement RTEF$_{SDC}$, toutes deux résultent en des valeurs bien inférieures pour la FWL. Tout comme pour l'évaluation basse définition également, l'utilisation de la transformée en distances inversement exponentielle est l'élément le plus critique, la version linéraire produisant les moins bons résultats au global.

Des illustrations de quelques uns des résultats obtenus sur ce dataset sont fournies en figure 4. Elles permettent de souligner visuellement la qualité du flot optique, ainsi que d'illustrer sa robustesse à certains phénomènes, tel que le mouvement d'objets contraires au mouvement global de l'image (par exemple, la voiture dépassant par la droite dans la séquence du 17 Juin).

4.4 Temps d'exécution

L'un des éléments principaux relatifs à notre méthode concernant également sa vitesse d'exécution, nous éva-

TABLE 3 – Temps d'exécution moyens et écart-type pour chaque module de notre architecture, pour des entrées basse et haute définition, en millisecondes. Le module limitant du pipeline est souligné pour chaque version.

Version	Accumulation des évts*	Débruitage & comblement	Transformée en distances inv. exp.	Flot optique†	Temps total
Basse définition (346×260)					
CPU seul	0.15±0.03	0.77±0.16	5.07±2.07	*3.59±0.09*	9.57±2.13
CPU & GPU	0.18±0.04	0.50±0.04	1.37±0.09	3.76±0.16	5.82±0.23
Haute définition (1280×720)					
CPU seul	0.54±0.06	3.55±0.30	7.43±0.51	*12.21±0.16*	23.73±0.69
CPU & GPU	0.55±0.07	0.69±0.14	3.75±0.46	11.89±0.76	16.88±1.30

*Ce module utilise uniquement le CPU, expliquant les résultats similaire entre les versions « CPU seul » et « CPU & GPU » pour cette colonne.

†La bibliothèque de flot optique fournie par Adarve *et al.* [1] ne contient pas de version utilisant le CPU seul. Les expérimentations « CPU seul » utilisent donc le GPU pour le calcul du flot optique, expliquant les résultats similaires entre les versions « CPU seul » et « CPU & GPU » pour cette colonne.

luons ici les temps nécessaires à chaque module du pipeline pour traiter les données fournies à leur entrée. Pour cela, nous utilisons les séquences « Indoor flying 1 » du dataset MVSEC et « moorea_2019-01-30_000_td_671500000_731500000_td » du dataset « 1 Megapixel Automotive Detection ». Deux versions sont analysées : la version complète, avec utilisation du CPU et du GPU, ainsi qu'une version « CPU seul ».

Les résultats obtenus sont présentés tableau 3. Ils permettent de déduire tout d'abord que, pour la version complète utilisant le GPU, notre méthode peut rester temps réel si des temps d'accumulation ΔT d'au moins 4ms et 13ms sont utilisés en basse et en haute définition respectivement (le facteur limitant dans les deux cas provenant du module de flot optique). Les valeurs de ΔT utilisées dans le cadre de cet article respectent donc la contrainte temps réel.

Ainsi, notre méthode peut atteindre une fréquence de sortie maximale de 250Hz avec une latence de 10ms (en comptant les $\Delta T = 4$ms d'accumulation) en basse définition, et de 77Hz avec une latence de 30ms (avec $\Delta T = 13$ms) en haute définition.

En guise de comparaison, la référence de l'état de l'art FireFlowNet [12] est capable de produire une fréquence de sortie similaire en basse définition, avec 262Hz. En haute définition, en revanche, notre méthode produit une sortie à une fréquence 2 à 3 fois supérieure, leur approche n'étant capable d'atteindre que 29Hz.

5 Conclusions

Dans le cadre de cet article, une architecture pour le calcul de flot optique basé évènements en temps réel a été proposée. Elle inclut l'utilisation d'une nouvelle transformée en distances inversement exponentielle, permettant la transformation d'évènements épars en une représentation dense sous forme d'une image. Des choix algorithmiques et d'optimisation ont également été présentés, permettant à notre méthode de fournir des résultats de flot optique en temps réel avec une faible latence. Une évaluation sur des jeux de données de l'état de l'art permet également de démontrer la justesse de notre méthode, que ce soit en basse ou en haute définition.

Bien que les approches basées apprentissage machine constituent aujourd'hui l'état de l'art, telles que EV-FlowNet$_{\text{HQF}}$ [14] ou FireFlowNet [12], celles-ci se doivent à ce jour de faire le choix entre vitesse d'exécution et qualité des résultats. Notre proposition, au contraire, est capable de fournir des résultats garantissant à la fois une fréquence et une justesse élevées. De plus, notre méthode ne reposant pas sur des principes d'apprentissage, celle-ci reste indépendante de tout choix de données d'entraînement ou d'une fonction objectif. Ainsi, sous réserve d'une sélection adéquate des paramètres utilisés, nos résultats devraient rester similaire à ceux présentés dans cet article indépendamment du type de scène observée.

Les résultats de l'approche actuelle étant au niveau de l'état de l'art, il serait prometteur d'investiguer quelques directions pour les optimiser encore. L'utilisation d'un temps d'accumulation ΔT fixe, par exemple, nous contraint à prédéterminer sa valeur en fonction du type de scène observée. L'utilisation d'une méthode telle que celle proposée par Jung *et al.* [11] pour ajuster dynamiquement le temps d'accumulation pourrait donc être envisagée, mais rendrait la contrainte temps réel plus compliquée à conserver du fait de sa complexité. Notre débruitage pourrait également être complexifié, afin de mieux couvrir certaines situations (éclairages urbains, par exemple), au détriment de sa vitesse d'exécution. Notre architecture pourrait enfin bénéficier d'une implémentation plus optimisée, sur FPGA par exemple, ce qui nous permettrait de nous passer de l'utilisation d'un GPU et d'avoir potentiellement des latences de sortie encore plus basses.

Remerciements

Ce travail est co-financé par la région Hauts-de-France et par le laboratoire commun SIVALab (Renault-UTC-CNRS).

Références

[1] J. Adarve and R. Mahony. A filter formulation for computing real time optical flow. *IEEE Robot. Autom.*

Lett., 1(2):1192–1199, 2016.

[2] H. Akolkar, S. Ieng, and R. Benosman. Real-time high speed motion prediction using fast aperture-robust event-driven visual flow. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(1):361–372, 2022.

[3] M. Almatrafi, R. W. Baldwin, K. Aizawa, and K. Hirakawa. Distance surface for event-based optical flow. *IEEE Trans. Pattern Anal. Mach. Intell.*, 42(7):1547–1556, 2020.

[4] C. Braillon, C. Pradalier, J. L. Crowley, and C. Laugier. Real-time moving obstacle detection using optical flow models. *Proc. Intell. Veh. Symp.*, pages 466–471, 2006.

[5] V. Brebion, J. Moreau, and F. Davoine. Real-time optical flow for vehicular perception with low- and high-resolution event cameras. *IEEE Trans. Intell. Transp. Syst.*, pages 1–13, 2021.

[6] D. Coeurjolly, A. Montanvert, and J. Chassery. Distances discrètes. In *Géométrie discrète et images numériques*, chapter 5, pages 123–145. Hermès Paris, 2007.

[7] Y. Feng, H. Lv, H. Liu, Y. Zhang, Y. Xiao, and C. Han. Event density based denoising method for dynamic vision sensor. *Appl. Sci.*, 10(6):2024, 2020.

[8] S. Galic and S. Lončarić. Spatio-temporal image segmentation using optical flow and clustering algorithm. *Proc. 1st Int. Workshop Image Signal Process. Anal. Conjunct*, pages 63–68, 2000.

[9] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 3354–3361, 2012.

[10] M. K. Hossen and S. H. Tuli. A surveillance system based on motion detection and motion estimation using optical flow. *Proc. 5th Int. Conf. Informat., Electron. Vis. (ICIEV)*, pages 646–651, 2016.

[11] J. H. Jung and C. G. Park. Constrained filtering-based fusion of images, events, and inertial measurements for pose estimation. *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 644–650, 2020.

[12] F. Paredes-Vallés and G. C. H. E. de Croon. Back to event basics: Self-supervised learning of image reconstruction for event cameras via photometric constancy. *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 3446–3455, 2021.

[13] E. Perot, P. de Tournemire, D. Nitti, J. Masci, and A. Sironi. Learning to detect objects with a 1 megapixel event camera. *Proc. 34th Conf. Neural Inf. Process. Syst.*, pages 1–20, 2020.

[14] T. Stoffregen, C. Scheerlinck, D. Scaramuzza, T. Drummond, N. Barnes, L. Kleeman, and R. Mahony. Reducing the sim-to-real gap for event cameras. In *Proc. ECCV*, pages 534–549, 2020.

[15] C. Tang, X. Zhao, J. Chen, L. Chen, and Y. Zhou. Fast stereo visual odometry based on LK optical flow and ORB-SLAM2. *Multimedia Syst.*, 4:1–10, 2020.

[16] A. Z. Zhu, N. Atanasov, and K. Daniilidis. Event-based feature tracking with probabilistic data association. *Proc. ICRA*, pages 4465–4470, 2017.

[17] A. Z. Zhu, D. Thakur, T. Özaslan, B. Pfrommer, V. Kumar, and K. Daniilidis. The multivehicle stereo event camera dataset: An event camera dataset for 3D perception. *IEEE Robot. Autom. Lett.*, 3(3):2032–2039, 2018.