Suivi et estimation de profondeur avec un banc stéréo événementiel embarqué sur un véhicule autonome

Anass El Moudni, Sébastien Kramm, Fabio Morbidi, Rémi Boutteau

Suivi et estimation de profondeur avec un banc stéréo événementiel embarqué sur un véhicule autonome
An event-based stereo 3D mapping and tracking pipeline for autonomous vehicles

A. El Moudni[1] S. Kramm[1] F. Morbidi[2] R. Boutteau[1]

[1] Univ Rouen Normandie, INSA Rouen Normandie, Université Le Havre Normandie, Normandie Univ, LITIS UR 4108, F-76000 Rouen, France

[2] Université de Picardie Jules Verne, Laboratoire MIS, UR 4290, Amiens, France

E-mail : anass.el-moudni@univ-rouen.fr

Résumé

Les caméras événementielles sont des capteurs bio-inspirés, activés par le mouvement, qui génèrent un flux d'événements asynchrones au lieu d'images à une fréquence fixe. Ces capteurs se sont avérés beaucoup plus performants que les caméras traditionnelles pour des mouvements très rapides et dans des conditions d'éclairage difficiles. Au cours de la dernière décennie, le flux d'événements produit par une caméra événementielle a été exploité dans de nombreuses tâches de perception 3D (estimation de la profondeur, suivi à 6 DDL, odométrie visuo-inertielle, etc.). Dans cet article, nous proposons un prototype de pipeline stéréo événementiel pour la reconstruction 3D et le suivi d'une caméra en mouvement. Le module de reconstruction 3D repose sur la fusion DSI ("disparity space image"), tandis que le module de suivi utilise les surfaces temporelles comme champs de distance anisotropes, pour estimer la pose de la caméra. L'efficacité de la solution proposée a été démontrée à l'aide d'expérimentations conduites sur des jeux de données événementielles publics, enregistrés par une voiture dans un milieu urbain.

Mots Clef

Caméra événementielle, Estimation de profondeur par une caméra stéréo, Odométrie visuelle, Image de l'espace des disparités, Surface temporelle, Véhicule intelligent.

Abstract

Event cameras are bio-inspired, motion-activated sensors which generate asynchronous events instead of intensity images at a fixed rate. These sensors have been shown to outperform by large margins traditional frame-based cameras in case of high-speed motions and scenes with a difficult lighting conditions. In the last decade, the continuous stream of events produced by an event camera has been exploited in numerous 3D perception tasks (depth estimation, 6-DoF tracking, visual-inertial odometry, etc.). In this paper, we propose a prototype event-based stereo pipeline for simultaneous 3D mapping and tracking. The mapping module relies on DSI (disparity space image) fusion, and the tracking module makes use of time surfaces as anisotropic distance fields, to estimate the pose of the stereo camera. Numerical experiments with a publicly-available event dataset recorded by a car in urban environments, demonstrate favorable performance of the proposed solution.

Keywords

Event-based camera, Stereo depth estimation, Visual odometry, Disparity space image, Time surface, Intelligent vehicle.

1 Introduction

Les véhicules intelligents, ou plus généralement les systèmes avancés d'aide à la conduite (ADAS), offrent des solutions aux problèmes liés au trafic et rencontrent aujourd'hui un succès croissant. Leurs principaux objectifs incluent la réduction du nombre d'accidents (94% d'entre eux est causés par des erreurs humaines), la diminution du niveau de stress induit par la conduite dans de zones à trafic intense, et la simplification du transport des personnes à mobilité réduite et des marchandises [2].

Pour effectuer toutes ces tâches, la *perception* joue un rôle crucial, car elle assure que le véhicule ait une bonne compréhension du milieu environnant. Les véhicules modernes sont équipés d'une multitude de capteurs avancés, tels que des caméras RGB (8 caméras dans le Model Y de Tesla), des radars [3], des LiDARs [4], des caméras RGB-D [5], et plus récemment, des *caméras événementielles* [6].

Les caméras événementielles sont des capteurs asynchrones et bio-inspirés qui ont récemment déclenché un changement de paradigme dans l'acquisition de l'information visuelle. En effet, contrairement aux caméras stan-

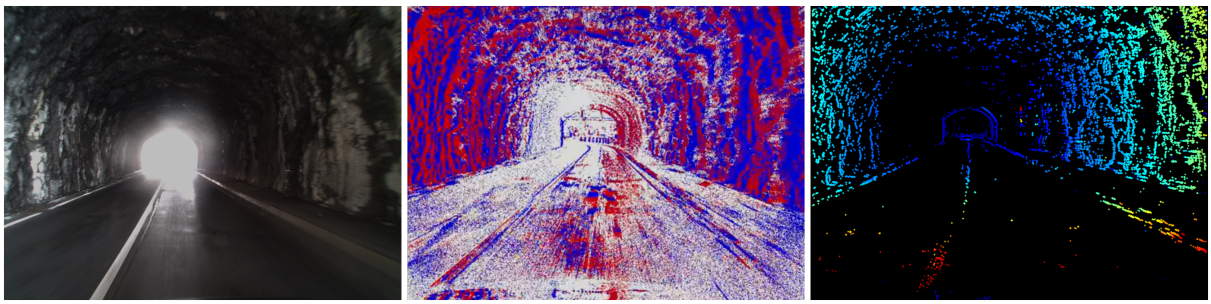


FIGURE 1 – *L'entrée/sortie de tunnel est notoirement problématique pour les automobilistes* : (A gauche) image RGB de la caméra de gauche du banc stéréo dans le dataset DSEC [1] ; (Au centre) Frame reconstruit à partir de la caméra événementielle de gauche de DSEC ; (A droite) Carte de profondeur inverse obtenue par fusion DSI (palette de couleurs "jet").

dards qui génèrent des images à une fréquence fixe, les capteurs événementiel ne réagissent qu'aux objets en mouvement dans la scène, en détectant les changements de luminosité au niveau de chaque pixel. Leur plage dynamique élevée (jusqu'à 140 dB, contre 60 dB pour les caméras traditionnelles), leur permet de fonctionner correctement dans des conditions d'éclairage difficiles (par exemple, faible luminosité, forte pluie, sortie d'un tunnel, voir la Fig. 1). D'autres propriétés intéressantes, telles que la faible latence, la consommation d'énergie réduite et la haute résolution temporelle (de l'ordre de la microseconde) en font un choix idéal pour les applications automobiles. Dans ce contexte, le principal défi consiste à concevoir des algorithmes capables de traiter le flux continu d'événements et d'en exploiter tout le potentiel, soit en le fusionnant avec les mesures d'autres capteurs [7], soit en concevant des représentations adaptées des événements qui permettent de reproduire le comportement des caméras classiques [8] afin d'utiliser les différents algorithmes déjà existants en vision.

La plupart des travaux existants traitent le problème de l'estimation de pose d'une caméra et de la reconstruction 3D, comme deux tâches distinctes. Pour le module d'estimation de profondeur, une solution directe consiste à reconstruire une image en niveaux de gris à partir des événements en les accumulant sur une fenêtre temporelle fixe ou via un réseau de neurones récurrent [9], puis à appliquer des algorithmes classiques de vision par ordinateur. Pour pouvoir respecter la nature asynchrone des données, des méthodes reposent sur l'association de données 'matching' [10, 11] (mise en correspondance pour le calcul de la disparité). D'autres méthodes ne nécessitent pas cette étape souvent coûteuse en temps de calcul : toutefois, la connaissance de la trajectoire de la caméra reste indispensable pour estimer la profondeur dans ce cas là en la reprojetant sur une vue de référence [12]. Concernant le problème du suivi, une variante du filtre de Kalman a été utilisée dans [13] pour prédire le mouvement à 6 DDL d'une caméra. Des réseaux de neurones à impulsions (*Spiking Neural Network*, SNN) ont également été proposés pour effectuer une régression numérique et estimer la vitesse angulaire d'une caméra événementielle [14].

D'après la revue de l'état de l'art précédente, nous pouvons observer que les algorithmes pour les caméras événementielles monoculaires sont très répandus. ESVO (Event-based Stereo Visual Odometry) [15] est l'une des rares méthodes existantes pour les *caméras événementielles stéréoscopiques*, et elle constitue le cœur du module de l'estimation de pose dans notre pipeline. Plus précisément, cet article apporte une amélioration de [15], qui résulte en un pipeline de suivi et de reconstruction 3D précis et autonome (en effet, il n'est pas nécessaire d'avoir une connaissance a priori de la pose de la caméra). Les contributions originales de cet article peuvent être résumées comme suit :

— Remplacement du module de reconstruction 3D d'ESVO par un module plus performant, basé sur la fusion des images de l'espace de disparité (DSI) [16].
— Validation du pipeline complet pour le suivi et la cartographie 3D, avec un jeu de données événementielles public (DSEC [1]), dans différentes situations de conduite.

Le reste de cet article est organisé comme suit : la Section. 2 passe en revue les travaux connexes sur la vision événementielle appliquée aux véhicules autonomes. Le problème étudié dans cet article est formulé dans la Section. 3. La Section. 4 présente les résultats des expérimentations réalisées avec des données événementielles réelles dans un milieu urbain. Enfin, les principales contributions de l'article, ainsi que quelques orientations pour les travaux futurs sont discutées dans la Section. 5.

2 État de l'art

Dans cette section, nous passerons brièvement en revue les représentations des événements les plus répandues, en mettant l'accent sur les surfaces temporelles (Section. 2.1). Les travaux connexes portant sur l'estimation de la profondeur (Section. 2.2) et de la pose d'une caméra événementielle (Section. 2.3), sont aussi révisés.

2.1 Représentation des événements

Comme nous l'avons vu dans la Section. 1, les caméras événementielles génèrent un flux asynchrone d'événements induits par le mouvement de la caméra et/ou de la scène observée. Afin de fournir plus de contexte et d'extraire plus d'informations sur l'environnement, les événements sont souvent agrégés dans l'espace et/ou dans le temps avant d'être traités. Les représentations des événements les plus utilisées sont les suivantes :

- **Événements individuels** ou **impulsions** : Les événements individuels sont représentés par un quadruplet, $\boldsymbol{e}_k = (x_k, y_k, t_k, p_k)$, où (x_k, y_k) sont les coordonnées pixelliques du k-ième événement, t_k est l'horodatage, et $p_k \in \{-1, +1\}$ est la polarité de l'événement ($p_k = +1$ si la variation du logarithme de l'intensité au pixel (x_k, y_k) est positive, et $p_k = -1$, sinon).
- **Groupes d'événements** : L'information sur le voisinage spatio-temporel apportée par un seul événement est généralement faible. Pour cette raison, les événements sont souvent regroupés en paquets (ou "clusters") de N_e éléments : $\mathcal{E} = \{\boldsymbol{e}_k\}_{k=1}^{N_e}$. Le choix de N_e dépend de la complexité de l'algorithme de traitement des données et de la puissance de calcul disponible.
- **Event frames** : Les algorithmes classiques de vision par ordinateur peuvent être utilisés avec des données événementielles, en définissant des *Event Frames* (ou Images Événementielles). Les événements sont accumulés sur une fenêtre temporelle de durée fixe T et représentés sous forme d'image, avec trois valeurs possibles par chaque pixel ($+1$ si la polarité de l'événement est positive, -1 si la polarité est négative, et 0 si aucun événement n'a été détecté dans l'intervalle T).
- **Surfaces temporelles (TS)** : Les surfaces temporelles sont des cartes 2D qui stockent l'horodatage de l'événement le plus récent [17]. L'intensité, dans ces pseudo-images, encode l'information temporelle et de mouvement : l'intensité associée aux événements plus anciens est plus faible par rapport à celle des événements plus récents (en vision par ordinateur classique, ces cartes 2D sont généralement appelées "motion history images"). En théorie, la carte devrait être mise à jour à chaque événement en entrée, en utilisant un noyau (pour conserver la nature éparse et asynchrone des événements). Cependant, en pratique, comme un seul événement n'est pas suffisamment informatif, plusieurs événements sont accumulés sur une fenêtre temporelle de durée T (le paramètre T doit être choisi avec attention : il dépend de la nature de la scène et des spécifications techniques de la caméra événementielle).

D'autres représentations des événements (telles que les voxel grids/cubes, les représentations basées graphes, etc.) existent dans la littérature, et elles présentent différents avantages et inconvénients, en fonction de l'application visée. Pour plus de détails, le lecteur est renvoyé à [6, Section. 3.1].

2.2 Estimation de la profondeur

Le problème de l'estimation de la profondeur avec des caméras événementielles a été largement étudié ces dernières années et il a été abordé sous différents angles.

La plupart des travaux existants utilisent un *banc stéréo événementiel* synchronisé pour l'estimation instantanée de la profondeur. Une approche en deux étapes est généralement adoptée : le problème de la mise en correspondance est d'abord résolu (en exploitant, par exemple, la contrainte épipolaire), puis une triangulation est effectuée en connaissant les paramètres intrinsèques et extrinsèques des caméras événementielles. L'idée de base est de définir une fonction coût pour l'appariement et de déterminer le candidat qui la minimise, tout en traitant les événements grâce à une fenêtre spatio-temporelle glissante. La propagation de croyance [10] et ESGM ("Event-driven Semi-Global Matching") [11] sont deux méthodes qui ont été utilisées pour optimiser la fonction de coût. L'inconvénient majeur de ces méthodes est leur complexité algorithmique, l'étape d'association des données étant le principal goulot d'étranglement. Pour contourner ce problème (et minimiser le nombre d'aberrances), on peut exploiter des sources d'information supplémentaires, comme l'orientation des bords ou la polarité des événements. Dans [18], les auteurs ont recours à la vitesse de la caméra pour estimer la profondeur de la scène : ils l'utilisent pour générer un volume événementiel de disparité synchronisé dans le temps, où les régions ayant la bonne disparité sont mises au point.

En revanche, les méthodes basées sur les *caméras événementielles monoculaires* pour l'estimation de la profondeur, adoptent une approche totalement différente, puisque la corrélation temporelle entre les événements dans la caméra de gauche et de droite n'est plus disponible. Certains des algorithmes existants reposent sur une connaissance préalable de la trajectoire de la caméra, comme la méthode de balayage de l'espace qui tire parti de la rareté des événements pour estimer la profondeur, sans nécessiter d'une étape d'association des données [12].

Globalement, les méthodes existantes pour l'estimation de la profondeur traitent les événements soit individuellement soit par paquets, en fonction de la représentation des événements adoptée. Les méthodes instantanées fonctionnent en temps réel et elles ont une précision satisfaisante, tandis que les méthodes qui traitent les événements par paquets s'avèrent être très précises, mais gourmandes en ressources de calcul. Le compromis précision/efficacité n'a pas encore été quantifié dans la littérature, et dans cet article nous proposons un nouveau pipeline (voir la Section. 3), qui bénéficie du meilleur des deux mondes.

2.3 Estimation de pose et suivi de la caméra

Les caméras événementielles sont souvent combinées avec d'autres capteurs traditionnels afin d'améliorer la préci-

sion et la robustesse de l'estimation de pose : par exemple, une centrale inertielle (IMU) pour l'odométrie visuo-inertielle [19], ou une caméra standard 'frame based' [20].
La majorité des méthodes événementielles existantes résolvent le problème de l'odométrie visuelle grâce à un suivi de "features", en traitant conjointement les événements et les images en niveaux de gris. En particulier, dans la littérature, on retrouve deux familles de méthodes :
Les *méthodes indirectes* traitent les événements à travers une représentation intermédiaire, généralement des cartes 2D (par exemple, des surfaces temporelles [17]), dont elles extraient des points d'intérêt à suivre. Bien que ces méthodes fonctionnent relativement bien avec des images standard en niveaux de gris [21], la nature éparse et le manque d'informations locales (voisinage spatial), rendent le suivi des événements beaucoup plus problématique.
En revanche, les *méthodes directes* traitent chaque événement singulièrement et des résultats satisfaisants en termes de précision, sont présentés dans [22].
Cependant, les algorithmes d'extractions de caractéristiques "features" et de *suivi* existants [23, 24] ne peuvent pas être facilement adaptés à l'odométrie visuelle événementielle car ils ne sont pas suffisamment précis et stable.
En général, les méthodes précédentes ont été conçues pour garantir une faible latence (c'est-à-dire pour estimer, en principe, la pose de la caméra à chaque événement entrant). Cependant, l'information apportée par un seul événement est généralement trop limitée pour permettre une estimation fiable de la pose.
Plus proche de l'objectif de ce travail, dans [15], les problèmes de cartographie (reconstruction 3D) et d'estimation de pose (suivi) sont traités simultanément pour une caméra stéréo événementielle : l'estimation initiale de la profondeur est calculée à partir d'une paire de TS, via une correspondance semi-globale, tandis que le module de suivi utilise les TS comme des champs de distance anisotropes. L'idée principale est d'aligner les régions "sombres" dans deux TS successives, et de trouver la transformation rigide qui donne le meilleur score de corrélation. Ce problème d'optimisation est résolu en utilisant la méthode de Lucas-Kanade [25]. Le module d'estimation de profondeur, quant à lui, exploite les notions de contrainte épipolaire et de co-occurrence des événements : en effet, chaque point sur les bords d'un objet observé génère deux événements simultanés sur les lignes épipolaires correspondantes, dans la caméra de gauche et de droite. En pratique, le problème de mise en correspondance est résolu en minimisant une fonction de coût de similarité basée sur les surfaces temporelles reprojetées des deux caméras.

3 Formulation du problème

La méthode proposée dans cet article s'appuie sur ESVO [15], où le module de reconstruction 3D a été remplacé par un module plus efficace, basé sur la fusion des images de l'espace de disparité [16]. À titre d'illustration, l'organigramme complet de notre méthode est présenté dans la Fig. 3.

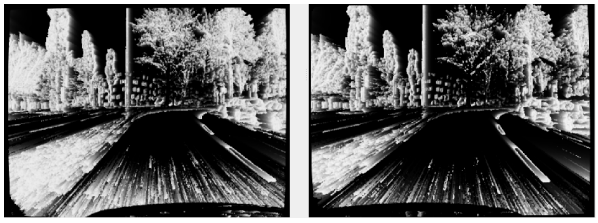


FIGURE 2 – Paire de surfaces temporelles générées à partir de données événementielles issues de la séquence "Zurich04" du dataset DSEC [1].

Les événements en entrée sont traités de deux façons différentes : le flux d'événements individuels $\boldsymbol{e}_k = (x_k, y_k, t_k, p_k)$ est d'abord traité pour construire les TS en utilisant le noyau exponentiel,

$$\mathcal{T}(\mathbf{x},\, t) \,=\, \exp\!\left(-\frac{t - t_{\text{last}}(\mathbf{x})}{\eta}\right), \tag{1}$$

où t_{last} dénote le temps de l'événement le plus récent à l'emplacement $\mathbf{x} = (x_k,\, y_k)$, et le paramètre $\eta > 0$ est le taux de décroissance. Les TS sont efficaces en mémoire et stockent l'historique du mouvement des contours dans la scène (les pixels les plus récents apparaissent en clair). Dans ESVO [15], les TS sont utilisées à la fois dans les modules de reconstruction 3D et de suivi, alors que dans ce travail nous les exploitons uniquement dans le module de suivi (blocs oranges dans la Fig. 3).
Après une étape d'initialisation dans laquelle une méthode modifiée du "semi global matching' est exploitée, une carte de profondeur préliminaire est construite et la première pose est estimée en corrélant la carte de profondeur avec la TS re-projetée, en utilisant la matrice de transformation rigide estimée. Les poses sont stockées dans une base de données et sont accessibles à tout moment grâce à une interpolation dans SE(3). La pose de la caméra est ensuite utilisée, avec les événements bruts, pour effectuer une rétro-projection dans l'espace (création de DSI), via la méthode EMVS (Event-based Multi-View Stereo) [12]. Enfin, le module de reconstruction 3D fusionne les DSI entre les caméras et dans le temps [16] : les rayons rétro-projetés ont une densité maximale à la bonne profondeur Z, ce qui permet de déterminer les positions 3D (ces positions peuvent être stockés dans la base de données sous forme de nuages de points). La carte de profondeur locale peut finalement être utilisée pour affiner l'estimation de la pose de la caméra.

3.1 Module d'estimation de la pose

Le module de suivi (blocs oranges dans la Fig. 3) utilise les TS comme champs de distance anisotropes (voir la Fig. 2). Les valeurs les plus grandes dans le TS correspondent aux contours récents, tandis que les valeurs les plus petites sont associées aux contours les plus anciens. Il s'agit d'une pro-

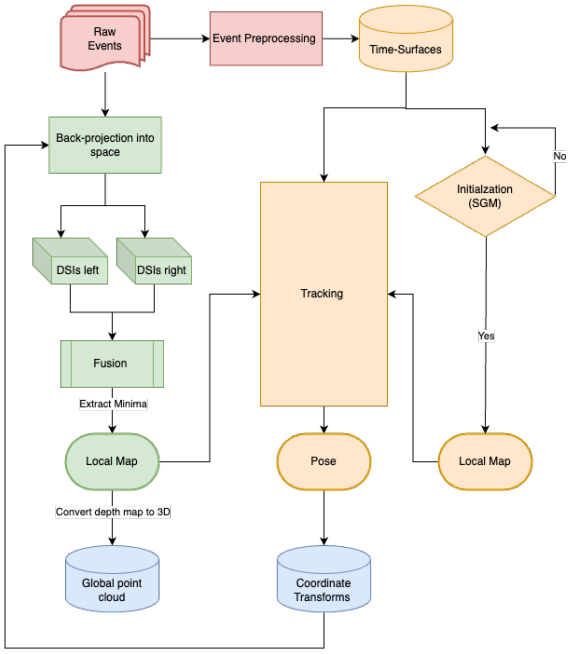


FIGURE 3 – Le pipeline proposé pour la cartographie 3D et le suivi (le module de reconstruction 3D est représenté en vert, le module de suivi en orange, les blocs pour le stockage de l'information en sortie en bleu, et les blocs d'entrées en rouge). L'organigramme a été adapté de [15].

priété intéressante des TS qui peut être utilisée pour suivre le mouvement de la caméra : en effet, il suffit de surveiller le taux de croissance des valeurs dans les TS. L'idée de base de la méthode, est d'essayer d'aligner les régions sombres dans la TS négative, définie par

$$\overline{\mathcal{T}}(\mathbf{x},\, t) \;=\; 1 - \mathcal{T}(\mathbf{x},\, t), \tag{2}$$

avec les cartes de profondeur inverses locales, lorsqu'elles sont projetées sur un repère donné de la caméra (généralement celui de gauche). La pose qui assure le meilleur alignement entre les régions sombres de la TS négative et les points de la carte de profondeur, correspond alors à la solution recherchée.
Ce problème peut être formulé comme suit. Supposons un paquet de pixels $\mathcal{S}_{\mathcal{F}_{\text{ref}}} = \{\mathbf{x}_i\}_{i=1}^{N_e}$ avec une profondeur connue Z_i dans le repère $\mathcal{F}_{\text{ref}}$, et une TS négative au temps k, notée $\mathcal{T}_{\text{left}}(\cdot,\, k)$. Ensuite, on cherche la pose qui permet de mieux aligner les minima de $\overline{\mathcal{T}}_{\text{left}}(\cdot,\, k)$ et la carte semi-dense reprojetée qui satisfait :

$$\boldsymbol{\theta}^* \;=\; \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \sum_{\mathbf{x}\,\in\,\mathcal{S}_{\mathcal{F}_{\text{ref}}}} \Big[\overline{\mathcal{T}}_{\text{left}}\big(W(\mathbf{x},\, Z,\, \boldsymbol{\theta}),\, k\big)\Big]^2, \tag{3}$$

où $W(\cdot,\, \cdot,\, \cdot)$ est la fonction de déformation (reprojection) qui transforme les points exprimés dans $\mathcal{F}_{\text{ref}}$, dans un autre repère donné.

La procédure consiste en trois transformations consécutives afin de trouver la matrice $\mathbf{T}$ qui garantit le meilleur alignement. Tout d'abord, on projette un événement à la position $\mathbf{x}$ dans un repère 3D, en utilisant la profondeur estimée. On multiplie ensuite par la matrice de transformation candidate $\mathbf{T}$, et enfin, la nouvelle position 3D est re-projetée dans le repère de gauche en utilisant la fonction de projection inverse.
La matrice de transformation $\mathbf{T} \in \text{SE(3)}$ est retrouvée grâce la fonction $\boldsymbol{G}(\boldsymbol{\theta}) : \mathbb{R}^6 \mapsto \text{SE(3)}$ qui met en correspondance le vecteur $\boldsymbol{\theta}$, contenant les 3 paramètres CGR (Cayley-Gibbs-Rodrigues) [26, 27] pour l'orientation et les 3 paramètres pour la translation, avec le groupe spécial euclidien tridimensionnel.

3.2 Module de reconstruction 3D

Pour décrire le module d'estimation de profondeur, il convient d'abord de rappeler la méthode de rétro-projection, car elle permet de générer une image dans l'espace des disparités (DSI). En vision par ordinateur classique, l'algorithme de balayage spatial permet de résoudre le problème multi-vues stéréo (MVS), et il a le grand avantage de générer des reconstructions 3D sans besoin d'une mise en correspondance des données entre les caméras. Contrairement à la majorité des approches existantes, qui exploitent les intensités des pixels pour résoudre le problème MVS en deux étapes (mise en correspondance puis triangulation), la méthode de balayage spatial s'appuie uniquement sur les contours des images, et ces données éparses ressemblent beaucoup à celles générées par une caméra événementielle.
L'approche de balayage spatial *classique* se décompose comme suit :

1. Rétro-projeter les points caractéristiques de l'image (les bords dans notre cas) sous forme de rayons dans un DSI. En d'autres termes, chaque fois qu'un point de contour déclenche un événement, il est compté comme un rayon à travers le DSI après rétro-projection. Si un pixel est vu par une caméra à une position donnée, il est compté comme un rayon à travers le DSI.
2. Compter le nombre de rayons traversant chaque voxel de la DSI et créez une fonction de densité de rayons qui est incrémentée chaque fois qu'un rayon traverse un voxel.
3. Choisir quel voxel correspond à la position 3D réelle de chaque pixel sur les bords. Pour cela, un seuil sur la fonction de densité de rayon est utilisé.

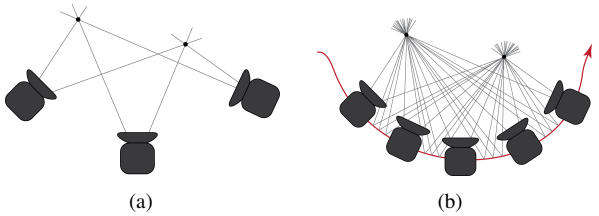
Dans le cas des *caméras événementielles*, aucun algorithme de détection de contours n'est nécessaire, car les événements sont déclenchés par des variations de luminosité, qui se produisent naturellement le long des contours.
Dans la Fig. 4(b), nous pouvons remarquer que les bords déclencheront des événements à partir de plusieurs points de vue "consécutifs", en raison du processus de mesure

continu et asynchrone. La méthode de balayage de l'espace basée sur les événements, prend donc des paquets d'événements $\{\mathbf{x}_k\}$ en entrée pour remplacer les caractéristiques ponctuelles qui seront ensuite rétro-projetées dans le DSI, puisque la position de la caméra (points de vue) est censée être connue à chaque instant, selon l'hypothèse MVS. Évidemment, plus le nombre de rayons générés par les multiples points de vue est élevé, plus la détection des régions d'intérêt dans la DSI est facile : il s'agit d'une simple analyse de la densité des rayons. Pour créer le DSI, nous commençons par choisir une vue de référence virtuelle au temps $t = T/2$ où T est la fenêtre temporelle du sous-ensemble d'événements traités. Ensuite, on définit un volume V qui est cohérent avec les spécifications techniques de la caméra événementielle. En particulier, la taille du volume dépend de la résolution de la caméra (largeur w et hauteur h), et de la résolution en profondeur (nombre de plans de profondeur N_z). Par conséquent, la taille du volume DSI est de $w \times h \times N_z$.

Le score est ensuite stocké en utilisant une fonction $f : \mathbb{R}^3 \rightarrow \mathbb{R}_{>0}$, qui compte les rayons de visualisation rétro-projetés passant par chaque voxel du volume DSI. Un seuil est introduit pour ne retenir que les voxels ayant une densité maximale de rayons, ce qui nous donne la localisation exacte en profondeur de l'objet observé dans la scène. Visuellement, sur le DSI, un objet sera flou s'il est éloigné du plan de profondeur correct, et net sinon.

Différentes fonctions de fusion sont considérées dans [16] pour maximiser la précision des estimations de profondeur et minimiser le nombre de valeurs aberrantes. Les DSI sont fusionnés entre les deux caméras et dans le temps, en subdivisant la fenêtre temporelle T en N_s sous-intervalles. La moyenne harmonique pour la fusion entre les caméras "H_c", et la moyenne arithmétique pour la fusion dans le temps "A_t", ont été empiriquement démontrées dans [16] pour fournir les meilleurs résultats.

En effet, la moyenne harmonique tend à donner plus de poids aux régions DSI ayant des valeurs élevées et similaires. En définitive, la moyenne harmonique maximise le score de corrélation entre les événements refocalisés dans les deux caméras, puisque les DSI fusionnés peuvent être définis comme des fonctions de similarité entre les événements refocalisés.



(a) (b)

FIGURE 4 – Comparaison de l'algorithme de balayage spatial, entre une caméra traditionnelle et une caméra événementielle. (a) Deux points dans l'espace sont observés par une caméra traditionnelle en mouvement ; (b) Les mêmes deux points sont observés par une caméra événementielle : le nombre de rayons est nettement supérieur car l'acquisition est faite en continu.

4 Expérimentations et résultats

Nous avons évalué l'architecture proposée avec le jeu de données public DSEC [1]. Pour créer ce dataset, les auteurs ont équipé une voiture avec deux caméras événementielles synchronisées Prophesee Gen 3.1. Les cartes de profondeur fournissant la vérité terrain, proviennent d'un LiDAR et d'une caméra stéréo RGB montée sur le toit de la voiture. Afin d'assurer la reproductibilité de nos résultats, le Tableau 1 rapporte les paramètres utilisés dans nos expérimentations. "Adaptive_threshold_c" est un paramètre du filtre adaptatif gaussien de seuillage appliqué à la DSI avant l'extraction de la carte de profondeur, et "Max_confidence" est utilisé pour définir la limite supérieure des valeurs de densité de rayons dans chaque DSI.

En comparant la Fig. 5(gauche) avec la Fig. 5(droite), nous pouvons remarquer que les cartes de profondeur générées par le module de reconstruction 3D de ESVO sont plus denses que celles produites par le module de fusion DSI, ce qui donne des nuages de points 3D plus grands. Cependant, comme l'indique le Tableau 5 dans [16] et comme le confirment nos expérimentations, ce dernier module améliore la précision de façon significative.

Nous avons également remarqué que les performances peuvent varier de manière significative en fonction du mouvement apparent des objets dans la scène (les mouvements faibles, correspondant par exemple à des véhicules se dirigeant dans la même direction que la voiture, sont plus problématiques).

Un nombre satisfaisant d'événements (bords) est généré pour détecter, par exemple, la présence d'un camion, mais il ne est pas suffisant pour rétro-projeter tous les véhicules et garantir des bords continus sur la carte de profondeur. La Fig. 6 montre l'une de ces occurrences : l'encadré rouge délimite la zone où se trouve le camion dans la carte de profondeur reconstruite. Le petit nombre d'événements générés par des objets dynamiques à mouvement apparent lent,

TABLE 1 – Paramètres utilisés dans notre module de reconstruction 3D.

Paramètres	**Valeurs**
w [pixels]	640
h [pixels]	480
Profondeur min [m]	4
Profondeur max [m]	150
N_z	100
Méthode de fusion	H_c
Adaptive_threshold_c	4
Max_confidence	468

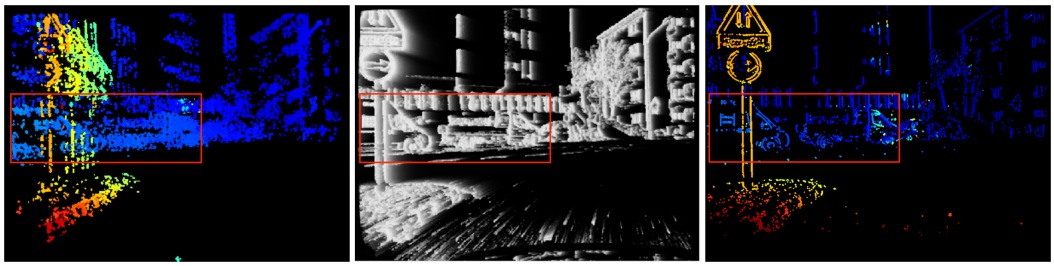


FIGURE 5 – Comparaison des deux modules de reconstruction 3D sur la séquence “Zurich04_a” du dataset DSEC : (A gauche) ESVO et (à droite) fusion DSI. La figure au centre montre la surface temporelle correspondante. On peut remarquer que les deux véhicules en mouvement devant la voiture à l’intérieur de l’encadré rouge, sont clairement reconnaissables dans la figure au centre et à droite, mais pas dans la figure de gauche.

TABLE 2 – Résultats quantitatifs de notre méthode.

Séquence	***Métriques d’évaluation***		
	Erreur moy. [m]	**Erreur méd. [m]**	**RMSE [m]**
Zurich04_a	4.4495	1.3023	7.4057
Zurich04_b	5.1118	1.6082	10.5680
Zurich02_a	3.8775	0.8251	10.7409
Interlaken_c	2.6765	0.6682	10.0547
Interlaken_d	3.3667	0.6109	8.5041

explique les zones “vides” présentes dans les cartes de profondeur estimées. Les surfaces réfléchissantes et sans texture de certains véhicules exacerbent encore ce problème, qui peut être contourné en ayant des surfaces type *granulaires* sur les véhicules qui génèrent plus d’évènements et augmente par conséquent la probabilité d’avoir plusieurs rayons reprojetés au même voxel du DSI, amenant à des cartes de profondeurs plus denses.

Enfin, pour une évaluation rigoureuse du pipeline proposé, nous avons effectué une analyse quantitative en utilisant cinq séquences urbaines du jeu de données DSEC. Le Tableau 2 présente l’erreur moyenne, l’erreur médiane et la racine de l’erreur quadratique moyenne (RMSE) des estimations de profondeur en mètres. Les résultats du tableau sont comparables à ceux rapportés dans [16]. Notre erreur est légèrement plus grande dans certains séquences, car nous utilisons les poses estimées de la caméra et non les trajectoires vérités terrains, comme dans [16]. Dans le Tableau 2, nous pouvons observer une certaine variabilité de l’erreur moyenne dans les cinq séquences : cela dépend des conditions d’enregistrement des données, des réglages de la caméra, ainsi que du nombre et de la vitesse des objets en mouvement dans la scène.

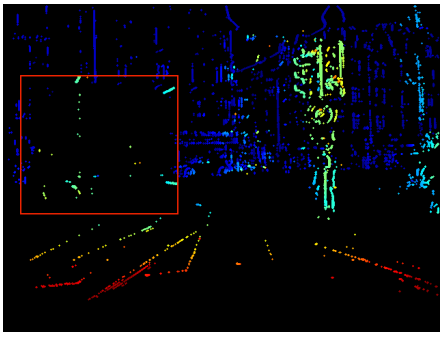


FIGURE 6 – Exemple de carte de profondeur inverse estimée dans la séquence “Zurich04_b”. L’encadré rouge délimite l’emplacement réel d’un camion, mais il contient seulement quelques pixels (vert et cyan).

5 Conclusion et perspectives

Dans cet article, nous avons présenté un prototype de pipeline complet pour le reconstruction 3D et le suivi d’une caméra stéréo événementielle montée sur un véhicule en mouvement. En particulier, nous avons remplacé le module de cartographie utilisé dans ESVO [15] avec un module plus performant basé sur la fusion DSI, qui ne nécessite pas de l’étape critique de mise en correspondance. Les expérimentations préliminaires réalisées avec des données événementielles réelles ont permis de valider l’architecture proposée en conditions opérationnelles.

Ce travail ouvre plusieurs pistes intéressantes pour les recherches futures. Tout d’abord, nous envisageons de réduire le nombre de degrés de liberté de notre module de suivi afin de l’adapter à la dynamique simplifiée d’un véhicule en mouvement, et de l’améliorer encore ses performances en fusionnant les données événementielles avec les mesures provenant d’autres capteurs embarqués comme l’IMU ou le LiDar. Dans un avenir proche, nous prévoyons d’effectuer une comparaison quantitative des différentes fonctions de fusion mentionnées dans la Section. 3.2, et de construire notre propre jeu de données multimodales pour tester le pipeline proposé dans différentes conditions météorologiques et de trafic. Une représentation commune des données, pour les modules de reconstruction 3D et de suivi, devrait également rendre notre pipeline plus robuste et rapide, avec des performances proches du temps réel.

Remerciements

Ce travail a été réalisé dans le cadre du projet CERBERE. Le projet, sous la référence ANR-21-CE22-0006, a été financé par l'Agence Nationale de la Recherche (ANR).

Références

[1] M. Gehrig, W. Aarents, D. Gehrig, and D. Scaramuzza. DSEC : A Stereo Event Camera Dataset for Driving Scenarios. *IEEE Rob. Autom. Lett.*, 6(3) :4947–4954, 2021.

[2] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda. A Survey of Autonomous Driving : Common Practices and Emerging Technologies. *IEEE Access*, 8 :58443–58469, 2020.

[3] F. Roos, J. Bechter, C. Knill, B. Schweizer, and C. Waldschmidt. Radar Sensors for Autonomous Driving : Modulation Schemes and Interference Mitigation. *IEEE Microw. Mag.*, 20(9) :58–72, 2019.

[4] S. Royo and M. Ballesta-Garcia. An Overview of Lidar Imaging Systems for Autonomous Vehicles. *Applied Sciences*, 19(9) :4093, 2019.

[5] J. Yang, C. Wang, H. Wang, and Q. Li. A RGB-D Based Real-Time Multiple Object Detection and Ranging System for Autonomous Driving. *IEEE Sensors J.*, 20(20) :11959–11966, 2015.

[6] G. Gallego, T. Delbrück, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A.J. Davison, J. Conradt, K. Daniilidis, and D. Scaramuzza. Event-Based Vision : A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(1) :154–180, 2022.

[7] A. Di Mauro, M. Scherer, D. Rossi, and L. Benini. Kraken : A Direct Event/Frame-Based Multi-sensor Fusion SoC for Ultra-Efficient Visual Processing in Nano-UAVs. In *Proc. IEEE Hot Chips 34 Symposium*, pages 1–19, 2022.

[8] A. Sironi, M. Brambilla, N. Bourdis, X. Lagorce, and R. Benosman. HATS : Histograms of Averaged Time Surfaces for Robust Event-Based Object Classification. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 1731–1740, 2018.

[9] H. Rebecq, R. Ranftl, V. Koltun, and D. Scaramuzza. High Speed and High Dynamic Range Video with an Event Camera. *IEEE Trans. Pattern Anal. Mach. Intell.*, 43(6) :1964–1980, 2019.

[10] Z. Xie, S. Chen, and G. Orchard. Event-Based Stereo Depth Estimation Using Belief Propagation. *Front. Neurosci.*, 11, 2017. Article no. 535.

[11] Z. Xie, J. Zhang, and P. Wang. Event-based stereo matching using semiglobal matching. *Int. J. Adv. Rob. Syst.*, 15(1), 2018.

[12] H. Rebecq, G. Gallego, D. Scaramuzza, and E. Müggler. EMVS : Event-Based Multi-View Stereo—3D Reconstruction with an Event Camera in Real-Time. *Int. J. Comput. Vision*, 126(12) :1394–1414, 2018.

[13] H. Kim, S. Leutenegger, and A.J. Davison. Real-time 3D reconstruction and 6-DOF tracking with an event camera. In *Proc. Europ. Conf. Comput. Vis.*, pages 349–364, 2016.

[14] M. Gehrig, S.B. Shrestha, D. Mouritzen, and D. Scaramuzza. Event-Based Angular Velocity Regression with Spiking Networks. In *Proc. IEEE Int. Conf. Robot. Automat.*, pages 4195–4202, 2020.

[15] Y. Zhou, G. Gallego, and S. Shen. Event-based Stereo Visual Odometry. *IEEE Trans. Robot.*, 37(5) :1433–1450, 2021.

[16] S. Ghosh and G. Gallego. Multi-Event-Camera Depth Estimation and Outlier Rejection by Refocused Events Fusion. *Adv. Intell. Syst.*, 4(12) :2200221, 2022.

[17] X. Lagorce, G. Orchard, F. Galluppi, B.E. Shi, and R.B. Benosman. HOTS : A Hierarchy of Event-Based Time-Surfaces for Pattern Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(7) :1346–1359, 2017.

[18] A.Z. Zhu, Y. Chen, and K. Daniilidis. Realtime Time Synchronized Event-based Stereo. In *Proc. Europ. Conf. Comput. Vis.*, pages 433–447, 2018.

[19] W. Guan, P. Chen, Y. Xie, and P. Lu. PL-EVIO : Robust Monocular Event-based Visual Inertial Odometry with Point and Line Features. *arXiv :2209.12160v*, Sept. 2022.

[20] J. Hidalgo-Carrió, G. Gallego, and D. Scaramuzza. Event-aided Direct Sparse Odometry. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 5771–5780, 2022.

[21] R. Mur-Artal and J.D. Tardós. ORB-SLAM2 : An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Trans. Robot.*, 33(5) :1255–1262, 2017.

[22] J. Engel, V. Koltun, and D. Cremers. Direct Sparse Odometry. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(3) :611–625, 2018.

[23] E. Müggler, C. Bartolozzi, and D. Scaramuzza. Fast event-based corner detection. In *Proc. British Machine Vis. Conf.*, pages 1–8, 2017.

[24] I. Alzugaray and M. Chli. Asynchronous corner detection and tracking for event cameras in real time. *IEEE Rob. Autom. Lett.*, 3(4) :3177–3184, 2018.

[25] S. Baker and I. Matthews. Lucas-Kanade 20 Years on : A unifying framework. *Int. J. Comput. Vision*, 56(3) :221–255, 2004.

[26] M.D. Shuster. A Survey of Attitude Representations. *J. Astronaut. Sci.*, 41(4) :439–517, 1993.

[27] J.E. Hurtado. Interior parameters, exterior parameters, and a Cayley-like transform. *ASME J. Dyn. Syst. Meas. Contr.*, 32(2) :653–657, 2009.