ORIGINAL RESEARCH
published: 06 February 2017
doi: 10.3389/fnins.2016.00596

Event-Based 3D Motion Flow Estimation Using 4D Spatio Temporal Subspaces Properties

Sio-Hoi Ieng, João Carneiro and Ryad B. Benosman*

Institut National de la Santé et de la Recherche Médicale, UMRI S 968; Sorbonne Université, University of Pierre and Marie Curie, Univ Paris 06, UMR S 968; Centre National de la Recherche Scientifique, UMR 7210, Institut de la Vision, Paris, France

State of the art scene flow estimation techniques are based on projections of the 3D motion on image using luminance—sampled at the frame rate of the cameras—as the principal source of information. We introduce in this paper a pure time based approach to estimate the flow from 3D point clouds primarily output by neuromorphic event-based stereo camera rigs, or by any existing 3D depth sensor even if it does not provide nor use luminance. This method formulates the scene flow problem by applying a local piecewise regularization of the scene flow. The formulation provides a unifying framework to estimate scene flow from synchronous and asynchronous 3D point clouds. It relies on the properties of 4D space time using a decomposition into its subspaces. This method naturally exploits the properties of the neuromorphic asynchronous event based vision sensors that allows continuous time 3D point clouds reconstruction. The approach can also handle the motion of deformable object. Experiments using different 3D sensors are presented.

Keywords: neuromorphic vision, event-based sensing, scene flow, 3D point clouds, motion estimation, motion from structure

OPEN ACCESS

Edited by:
Jorg Conradt,
Technische Universität München, Germany

Reviewed by:
Sadique Sheik,
University of California, San Diego, USA
Soumyajit Mandal,
Case Western Reserve University, USA

****Correspondence:***
Sio-Hoi Ieng
sio-hoi.ieng@upmc.fr

Specialty section:
This article was submitted to Neuromorphic Engineering, a section of the journal Frontiers in Neuroscience

Received: *08 September 2016*
Accepted: *14 December 2016*
Published: *06 February 2017*

Citation:
Ieng S-H, Carneiro J and Benosman RB (2017) Event-Based 3D Motion Flow Estimation Using 4D Spatio Temporal Subspaces Properties. Front. Neurosci. 10:596. doi: 10.3389/fnins.2016.00596

1. INTRODUCTION

1.1. Scene Flow

The motion of 3D structures is an important information to extract from a scene to build geometric and dynamic descriptions of its content. Such information is also essential to a large set of vision applications such as: virtual reality synthesis, scene segmentation and autonomous navigation. Scene flows are vector fields that map points of a 3D structure to their instantaneous velocity vectors. Because of this close relationship, estimating the scene flow usually implies to estimating the structure and vice-versa.

The Structure From Motion (SFM) is one of the classical computer vision problems that have been largely studied during the past few decades by the machine vision community (Maybank, 1993). However, SFM's high vulnerability to images' noise and to camera calibration errors raised questions regarding its applicability in real-world scenarios (Tomasi and Zhang, 1995). Currently, with the increasing demand for realistic and high definition 3D content, many ready-to-use sensors are now able to provide dense 3D points clouds in real-time (such as: laser range-finders, structured light vision sensors,...). These devices allow to decouple the structure reconstruction from the motion estimation and to focus the effort on motion extraction and its characterization.

To achieve dense scene flow estimation, state-of-the-art techniques estimate depth maps and compute optical flows for each camera separately. In a second stage they combine both to estimate the 3D flow. This approach parametrizes the motion problem on the image plane, i.e., in 2D and is the most commonly found in the existing literature (Vedula et al., 1999; Zhang et al., 2001; Isard and MacCormick, 2006; Wedel et al., 2011). A 2D parametrization is however more prone to discontinuities since a smooth 3D signal may be projected into a discontinuous 2D signal due to occlusions.

In Basha et al. (2013), the depth map and the optical flow are solved simultaneously rather than in a sequential manner, as authors argue, for a better coupling between spatial and temporal information. In Hadfield and Bowden (2014) and Park et al. (2012), the motion flow is extracted and refined directly from the 3D points clouds by using particle filtering or tensor voting techniques. Optical flows are only estimated for comparison purposes or for initial scene flow estimation.

A second requirement for obtaining dense flow estimation is to introduce some form of regularization. For that purpose, one recurrent hypothesis is to assume local rigid body motion and therefore induce local constant velocity, i.e., points on a non-deformable surface will have the same velocity. Regularization is often performed by minimizing an energy function with variational formulations (Zhang et al., 2001; Min and Sohn, 2006; Huguet and Devernay, 2007). Energy minimization has proven to be a successful technique for both 2D and 3D flow parametrization. It is however computationally greedy and it makes it difficult to achieve real-time estimation without embedding a dedicated powerful computational unit (e.g., GPU). Scene flow can also be computed from local descriptors of reconstructed surfaces such as surfel that encodes the local geometry and the reflectance information of the shapes (Carceroni and Kutulakos, 2002). Motion is then estimated in an integrative manner by matching descriptors over time. Several authors adopted the same idea of addressing the scene flow as a problem of characterization and tracking 3D surfaces over time. Varanasi et al. (2008) proposes to describe and track the surfaces by sparse features matching and extend this to a dense estimation using smoothing operations based on the Laplacian diffusion. Patch based techniques have also been used in Popham et al. (2010) and Cagniart et al. (2010) to split complex surfaces into simpler ones. Their matching and relative pose estimation for each patch allows to estimate the scene flow densely.

This paper introduces a new solution to estimate scene flow using properties of 4D (3D space+time) spaces without the need to use luminance. We will show that the use of the time allows to go beyond the conventional framework that relies on the combined use of luminance and depth information (Herbst et al., 2013). The paper is initially intended to operate on high temporal resolution 3D depth information output from a binocular neuromorphic event-based camera stereo rig. As introduced in Rogister et al. (2012) and Carneiro et al. (2013), event-based cameras allow to estimate depth and produce 3D point clouds at unprecedented accuracy ($>$1 kHz in real-time) at very low computational and energy cost using conventional processing hardware. We will show that the method can be used even in the case of lower temporal resolution and it can be applied to any 3D data such as the ones output from: RGB-D cameras (Khoshelham and Elberink, 2012), time-of-flight range-imaging sensors (Hansard et al., 2012), laser range finder and even conventional camera based systems that are also able to provide robust 3D reconstructions with a reasonable accuracy using optimized implementations. It is however important to notice that beyond the heavy computational and energy requirement, all these techniques rarely exceed frame rates beyond 90 Hz.

We will then show that the use of timed 4D spaces (3D space + time) allow to derive more efficient techniques than state of art techniques. The method assumes locally non-deformable spatiotemporal surfaces swept by 3D moving structures. We show that under such hypothesis, the velocity estimation is reduced to a one dimensional search over $\mathbb{R}$, the set of real numbers, and the dense estimation is directly achieved using local spatiotemporal planes. An additional advantage is its ability to determine velocities collinear to moving edges assuming it is possible to identify local 3D structures across the trajectory. This work can be seen as a generalization of the previous work on the event-based estimation of 2D visual motion flow (Benosman et al., 2014) to higher dimensional spaces.

1.2. Asynchronous Event-Based Vision

Biological retinas do not encode visual scenes as collection of static frames, but rather as a continuous stream of asynchronous spikes. Neuromorphic vision sensors replicate partially this mechanism by encoding visual information with high temporal resolution asynchronous streams of events. Since the pioneering work of Mahowald (1992) that built the first retina on silicon, several major improvements have been made for what is now refered to as the "neuromorphic silicon retinas." One of the most important achievements is the Dynamic Vision Sensor (DVS) (Lichtsteiner et al., 2008), a 128 $\times$ 128 pixel resolution sensor which encodes light intensity changes into a stream of asynchronous events. Each pixel responds independently to contrast changes producing ON and OFF events (respectively to increase or decrease in light intensity) at microsecond resolution.

Posch et al. designed the Asynchronous Time-based Imaging Sensor (ATIS) (Posch et al., 2011), a 302 $\times$ 240 pixel resolution sensor which measures absolute luminance information when a contrast change event occurs. The sensor provides a 143 dB dynamic range gray-level information asynchronously encoded as the temporal difference of two exposure measurement events. Its typical temporal accuracy is around 1 μs. The reader can refer to Delbrück et al. (2010) for a complete review of the existing neuromorphic visual sensors.

2. MATERIALS AND METHODS

2.1. Scene Flow Parametrization

We define a 3D event as a 4-components vector $(x, y, z, t)^T$. It can be increased to 5 components if the luminance information is available. Let us consider a smooth edge $\mathcal{C}$ which can be assumed planar within a small enough spatial neighborhood. If the velocity of $\mathcal{C}$ is constant, then as time increases, the edge generates a ruled surface $\mathcal{S}$ in the direction of the velocity $\mathbf{v}$. The surface can be algebraically defined by the equation:

$$\begin{array}{rcl} \mathcal{S}\colon \mathbb{R}^3 \times \mathbb{R}^+ & \rightarrow & \mathbb{R} \\ (x,y,z,t) & \mapsto & \mathcal{S}(\mathbf{p}+t\mathbf{v}) = 0 \end{array} \quad (1)$$

where $\mathbf{p} \in \mathcal{C}$. **Figure 1** shows an illustration of such ruled surface.

The velocity vector is according to Equation (1) the directrix of the ruled surface swept by the edge, hence the estimation of $\mathbf{v}$ is equivalent to determining the surface's directrix. In addition to Equation (1), if the surface is smooth (i.e., of class $\mathcal{C}^1$ at least), we get a second equation satisfied by $\mathbf{v}$:

$$(\nabla\mathcal{S})^T\mathbf{v} = 0, \quad (2)$$

because the directrix $\mathbf{v}$ is contained in the tangent plane $\mathcal{T}_p$ (Sommerville, 1934). $\nabla\mathcal{S}$ refers to the gradient of $\mathcal{S}$. Only the direction of $\mathbf{v}$ can be deduced from the two scalar equations since $\mathbf{v}$ has 3 components. Its norm can be set arbitrarily to 1. To determine the exact amplitude, additional constraints are required. A possible way to estimate amplitudes is to apply a shape registration technique, since the velocity vector is the vector joining the two consecutive positions of the shape when it moves. We then propose to estimate the velocity in a two steps operation:

1. a local fitting of a smooth surface to the 3D points clouds is operated to derive as much equations similar to (1) and (2) as possible,
2. $\mathbf{v}$ is then estimated from the equations established in step 1 by adding a shape registration algorithm.

To get enough equations to estimate $\mathbf{v}$, we propose to study three surfaces derived from $\mathcal{S}$. Let $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$ be respectively the surfaces built from Equation (1) in each coordinate frame (X,Y,T), (Y,Z,T), and (Z,X,T). Because of the constant velocity hypothesis, we get three surfaces with implicit equations of the form:

$$\mathcal{S}_k(i,j,t) = \mathcal{S}_k\left(\begin{pmatrix} p_i \\ p_j \\ 0 \end{pmatrix} + t\begin{pmatrix} v_i \\ v_j \\ 1 \end{pmatrix}\right) = 0, \quad (3)$$

where (i,j) is any pair of elements in $\{(x,y),(y,z),(z,x)\}$ and k indexes the k^{th} element of this list e.g., if $k = 1$, $(i,j) = (x,y)$. This means we are working with the x, y and t components of S.

These surfaces are also ruled surfaces of respective directrices $(v_x,v_y,1)^T$, $(v_y,v_z,1)^T$ and $(v_z,v_x,1)^T$ and their generatrices are the restrictions of $\mathcal{C}$ to (X,Y,T), (Y,Z,T) and (Z,X,T). For the same reason, given Equation (2), we can establish for each $\mathcal{S}_k$ the equation:

$$(\nabla\mathcal{S}_k)^T\begin{pmatrix} v_i \\ v_j \\ 1 \end{pmatrix} = \frac{\partial\mathcal{S}_k}{\partial i}v_i + \frac{\partial\mathcal{S}_k}{\partial j}v_j + \frac{\partial\mathcal{S}_k}{\partial t} = 0. \quad (4)$$

As illustrated by **Figure 2**, we now have three geometric constraints, which can be rearranged into a matrix form:

$$\underbrace{\begin{pmatrix} \mathcal{S}_{1,x} & \mathcal{S}_{1,y} & 0 \\ 0 & \mathcal{S}_{2,y} & \mathcal{S}_{2,z} \\ \mathcal{S}_{3,x} & 0 & \mathcal{S}_{3,z} \end{pmatrix}}_{M}\mathbf{v} = -\begin{pmatrix} \partial\mathcal{S}_1/\partial t \\ \partial\mathcal{S}_2/\partial t \\ \partial\mathcal{S}_3/\partial t \end{pmatrix}, \quad (5)$$

with the convention that $\mathcal{S}_{k,x}$ (respectively y,z) is the partial derivative with respect to x (respectively y,z). To determine $\mathbf{v}$, the ideal case would be to have M invertible i.e., it is full ranked. There is no obvious way to tell from the general expression of M.

2.2. Plane Approximation

Solving Equation (5) for $\mathbf{v}$ cannot be done without knowing the analytic equations of $\mathcal{S}_k$, so we propose to apply a local plane fitting to establish the matrix M. The choice of a plane instead of a more complex surface is motivated by the fitting simplicity and its computational cost even though planes give rise to rank-2 matrices M, as it will be shown further.

Let Π_1, Π_2, and Π_3 be the planes that are fitted locally to the surfaces $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$ respectively. They then can be locally expressed using the plane's implicit equation as:

$$\mathcal{S}_k(i,j,t) = \Pi_1{}^T\begin{pmatrix} i \\ j \\ t \\ 1 \end{pmatrix} = 0, \quad (6)$$

where $\Pi_\mathbf{k}{}^T = (a_k, b_k, c_k, d_k)$, for $1 \leq k \leq 3$.

If we derive Equation (6) with respect to each of the spatial and temporal components and for each $\mathcal{S}_k$, then Equation (5) becomes

$$\begin{pmatrix} a_1 & b_1 & 0 \\ 0 & a_2 & b_2 \\ b_3 & 0 & a_3 \end{pmatrix}\mathbf{v} = -\begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix}. \quad (7)$$

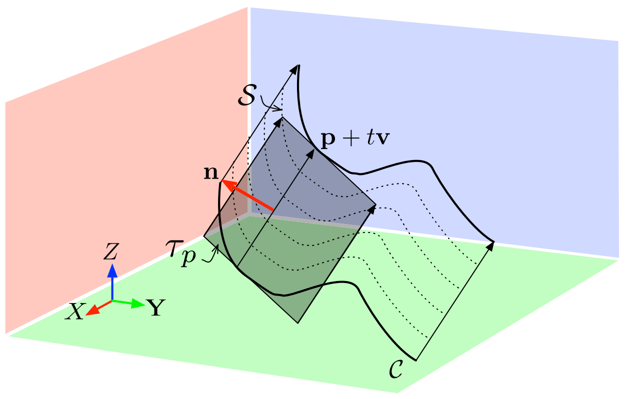

FIGURE 1 | The non-deformable surface hypothesis allows to assume the velocity v is locally constant. The surface $\mathcal{S}$ swept by the edge $\mathcal{C}$ in the direction $\mathbf{v}$ is a ruled surface whose tangent plane τ_p at $\mathbf{p}$ allows to recovering $\mathbf{v}$ if sufficient geometric constraints can be derived. The vector $\mathbf{n}$ is the normal to τ_p.

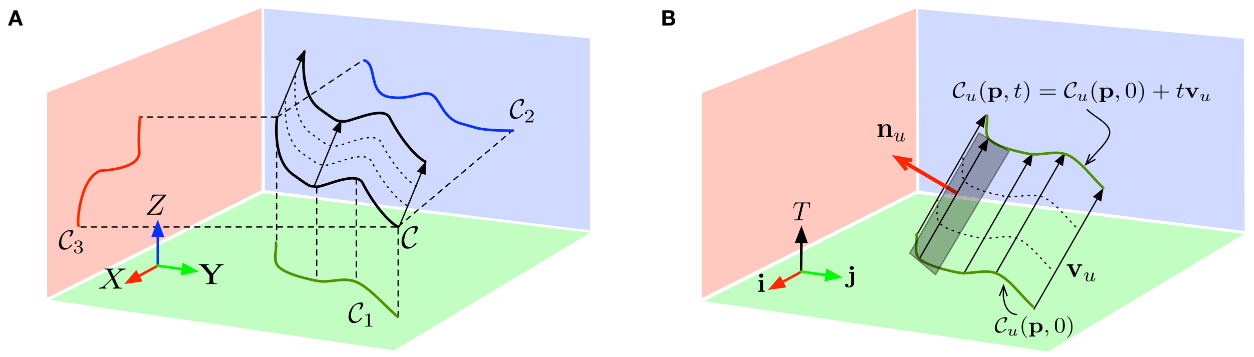

FIGURE 2 | (A) A 3D edge $\mathcal{C}$ moving at constant velocity **v** is projected as 2D curves in each of the three planes (O, X, Y), (O, Y, Z) and (O, Z, X). **(B)** Each of the projected curve C_u for $1 \le u \le 3$ is also moving at constant speed $\mathbf{v}_u = (v_i, v_j, 1)^T$ in the coordinate frames (ijT) $((i, j)$ being any element in the set $\{(x, y), (y, z), (z, x)\})$ and is sweeping a ruled surface as t increases.

2.3. Rank of *M*

Under the local plane hypothesis we previously made, it is possible to determine the rank of M. For that purpose, we assume the hypothesis that the edge $\mathcal{C}$ is a straight line segment defined by a point $\mathbf{p}_0$, a direction vector $\mathbf{u}$, and parametrized by a real α:

$$\mathbf{p} \in \mathcal{C} \Rightarrow \mathbf{p} = \mathbf{p}_0 + \alpha\mathbf{u}, \tag{8}$$

and the equation of $\mathcal{S}$ is changed into:

$$\mathcal{S}(\mathbf{p}, \mathbf{v}, t) = \mathcal{S}(\mathbf{p}_0 + \alpha\mathbf{u} + t\mathbf{v}) = 0. \tag{9}$$

Figure 3 depicts the case where $\mathcal{C}$ is a line and the resulting ruled surface $\mathcal{S}$, obtained by sweeping lines in the direction of $\mathbf{v}$ is a plane. The vector $(u_i, u_j, 0)^T$ is by construction parallel to Π_k, then:

$$\mathbf{n}_k^T \begin{pmatrix} u_i \\ u_j \\ 0 \end{pmatrix} = 0, \tag{10}$$

where $\mathbf{n}_k = (a_k, b_k, c_k)^T$ is the normal to Π_k. The three similar equations for the three possible k lead to:

$$M\mathbf{u} = 0. \tag{11}$$

This shows $\mathbf{u}$ as an element of the kernel of M. $\mathbf{u}$ is not the null vector because $\mathcal{C}$ is not reduced to a point, thus M is non-invertible and the rank of M is not larger than 2. The rank deficiency of M means we only have two linearly independent scalar equations from Equation (7), however we can still express two of the velocity components as functions of the last one, e.g., v_x:

$$\mathbf{v} = \begin{pmatrix} v_x \\ \frac{-a_1 v_x + c_1}{b_1} \\ \frac{-b_3 v_x + c_3}{a_3} \end{pmatrix} = v_x \underbrace{\begin{pmatrix} 1 \\ -\frac{a_1}{b_1} \\ -\frac{b_3}{a_3} \end{pmatrix}}_{\mathbf{q}} + \underbrace{\begin{pmatrix} 0 \\ \frac{c_1}{b_1} \\ \frac{c_3}{a_3} \end{pmatrix}}_{\mathbf{r}}, \tag{12}$$

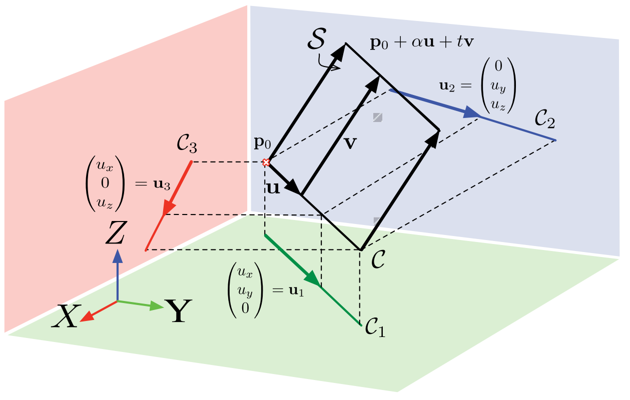
$\mathcal{S}$
$\mathbf{p}_0 + \alpha\mathbf{u} + t\mathbf{v}$
$\mathbf{u}_2 = \begin{pmatrix} 0 \\ u_y \\ u_z \end{pmatrix}$
$\mathcal{C}_2$
$\mathcal{C}_3$
$\mathbf{p}_0$
$\mathbf{v}$
$\begin{pmatrix} u_x \\ 0 \\ u_z \end{pmatrix} = \mathbf{u}_3$
$\mathbf{u}$
Z
$\mathcal{C}$
$\begin{pmatrix} u_x \\ u_y \\ 0 \end{pmatrix} = \mathbf{u}_1$
X
Y
$\mathcal{C}_1$

FIGURE 3 | The local fitting of a plane to the point cloud allows approximating the plane tangent to the surface swept by an edge as it moves. If the velocity is constant, the so built surface is called ruled surface and the velocity vector **v** is its directrix. To estimate **v**, it is, up to approximation errors, equivalent to working on the tangent plane.

where $M\mathbf{q} = \begin{pmatrix} 0 \\ \frac{\det(M)}{a_3 b_1} \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$.

This last equation shows that $\mathbf{q}$ is collinear to $\mathbf{u}$ if M is rank 2, hence we deduce from Equation (7) that $M\mathbf{r} = (c_1, c_2, c_3)^T$.

Remark 1. M's rank can only be reduced to one if there is no spatial translation. The swept structures in the subspaces defined by (ijT) are vertical lines. Such a case is a particular case which is detected when no ruled surface is generated. It does not concern structures undergoing rotations since points not on the rotation axis will have a non null tangential velocity.

Remark 2. Expressing $\mathbf{v}$ *as a one parameter vector fails if and only if the rank of M is less than 2 i.e., if edges do not generate planes. However, some plane configurations require larger effort to achieve the closed form of* $\mathbf{v}$ *such as the case where the plane is perpendicular to one of the spatial frame axes. For example, when the X-axis is normal to the plane, Equation (12) is not valid as b_1*

and a_3 are equal to zero. This problem can be solved by expressing $\mathbf{v}$ *either as a function of v_y or v_z. In that case, we can see that $v_x = -c_3/b_3$ and v_z is a function of v_y. The problem of finding* $\mathbf{v}$ *is again reduced to the search for the correct value of one of its component.*

2.4. Velocity Estimation

As shown in the previous section, from Equation (12), the assumption of local constant velocity motion of straight edges allows to establish a simple linear relation between the velocity vector and the surface swept by the edge points. Estimating the velocity becomes equivalent to identifying the correct real value v_x. This is a registration problem for which we need to initiate the point cloud within a spatio-temporal neighborhood as a given structure. We then translate it according to vectors $\mathbf{v}$, parametrized by v_x. A matching operation is then performed for several sampled values of v_x, the correct v_x is the one producing the smallest matching error at the time and location given by the velocity vector (see **Figure 4**). The procedure to estimate the velocity via the shape registration is explained in detail by Algorithm 1: the search for $\mathbf{v}$ is now a minimization problem of a error cost function E, which is built as explained in the next section.

2.5. Error Cost Function

A local point cloud centered on the event $(\mathbf{p}_1, t_1)^T$, is temporally consistent in the sense that any of its element will be captured at closely the same time. If in addition, the luminance L of the events is available, then the cloud local rigidity also ensures that L is consistent independently of time. We can therefore state that when the point cloud that moves from $\mathbf{p}_1$ at t_1 to $\mathbf{p}_2$ at t_2, the local geometric structure and the luminance should be preserved. We can formalize the structure matching operation as a minimization of the energy E problem and stated as follows:

$$E = E_S + E_T + E_L, \quad (13)$$

where E_S, E_T, and E_L are respectively the geometric, the temporal and the luminance energies. E_T and E_S are minimal as long as the cloud is not deforming when it moves from $\mathbf{p}_1$ to $\mathbf{p}_2$. If the events' brightness is also preserved during this motion then E_L is also minimal.

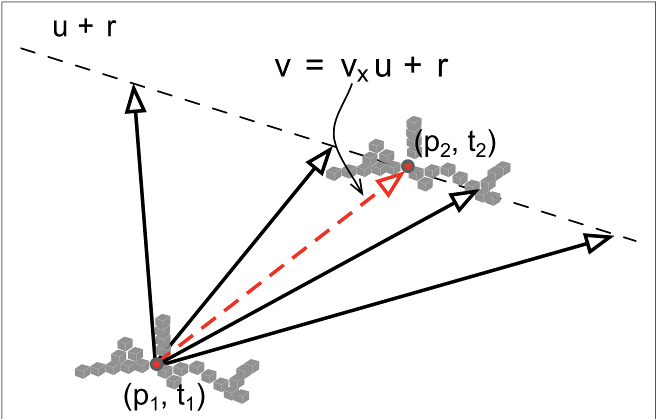

FIGURE 4 | The velocity is to be determined locally along a line spanned by u and passing by p + r. This is achieved by matching local structure defined by a set of 3D points (gray cubes in the figure).

We define the 3D events cloud $S(\mathbf{p}_i, t_i)$ as:

$$S(\mathbf{p}_i, t_i) = \{\mathbf{q}_j \in \mathbb{R}^3 \mid ||\mathbf{q}_j - \mathbf{p}_i|| \leq \Delta_s \ , t_j - t_i \leq \Delta_t \text{ and } t_j > t_i\}. \quad (14)$$

This set contains all 3D points spatiotemporally close to $\mathbf{p}_i$ i.e., points within a neighborhood of $\mathbf{p}_i$ of radius Δ_s in space and length Δ_t in time. The energy cost associated to each sampled velocity vector for a given point $\mathbf{p}_0$ is computed according to Algorithm 2.

E_L is the sum of the smallest luminance difference between all pairs of $(\mathbf{p}_i, \mathbf{q}_j)$ and E_S is the mean value of the smallest distances of each $\mathbf{p}_i$ to each $\mathbf{q}_j$. It is also called the mean closest point between both points clouds and is a dissimilarity measure often used for example in the Iterative Closest Point (ICP) problem (Besl and McKay, 1992). The correct $\mathbf{v}$ is given by the

Algorithm 1 | 3D flow algorithm

```
Require: Stream of 3D events obtained from third-party device/algortihm
 1: for each 3D event $(\mathbf{p}, t)$ do
 2:     Determine the spatio-temporal neighborhood of 3D event close to $(\mathbf{p}, t)$.
 3:     Fit 3 planes $\Pi_1 = (a_1, b_1, c_1, d_1)^T, \Pi_2 = (a_2, b_2, c_2, d_2)^T, \Pi_3 = (a_3, b_3, c_3, d_3)^T$ using a least-square technique to minimize the three scalars:
 4:
        $|(p_x, p_y, t, 1)\Pi_1|, |(p_x, p_z, t, 1)\Pi_2|, |(p_y, p_z, t, 1)\Pi_3|$
 5:     Initialize a large enough interval $R = [R_1, R_r]$ of length $l$ such that $\tilde{v}_x \in R$. Set n = 1.
 6:     while $E >$ threshold and $n <$ max-iteration do
 7:         Divide $R$ into $r$ intervals $R_k$ of size $\frac{l}{r}$ and define the set $\{v_k\}$ such that $v_k$ is the center of $R_k$.
 8:         for each $v_k$ do
 9:             Compute $E_k$ according to Algorithm 2,
10:             if $E_k$ is minimal then
11:                 Update $E \leftarrow E_k$,
12:                 Update $l \leftarrow l(r-1)/r$,
13:                 Update $R \leftarrow [R_k - \frac{l}{2}, R_k + \frac{l}{2}]$.
14:                 Compute v:
                        $\mathbf{v} = \left(v_k, -\frac{a_1 v_k + c_1}{b_1}, -\frac{b_3 v_k + c_3}{a_3}\right)^T$
15:                 Update $n \leftarrow n + 1$.
16:             end if
17:         end for
18:     end while
19:     Return v
20: end for
```

Algorithm 2 | Energy cost computation

```
Require: $\mathbf{p}_0$, the set $S(\mathbf{p}_0)$, $v_x$.
1: Apply Equation 12 with the given $v_x$ to build vector $\mathbf{v}$.
2: Define $S(\mathbf{p}_0) + \mathbf{v}$, the translated local structure $S(\mathbf{p}_0)$ by $\mathbf{v}$. Define $S(\mathbf{p}_0 + \mathbf{v})$ the set of points that occur in the neighborhood of $\mathbf{p}_0 + \mathbf{v}$ at $t_i + dt$.
3: With the convention that $\mathbf{p}_i \in S(\mathbf{p}_0) + \mathbf{v}$, and $\mathbf{q}_j \in S(\mathbf{p}_0 + \mathbf{v})$, we compute the energy function $E(\mathbf{v})$ with:
       $E_S = \frac{1}{n}\sum_{i=1}^{n} \min_{\mathbf{q}_j} ||\mathbf{p}_i - \mathbf{q}_j||,$   (15)
       $E_T = \frac{1}{n}\sum_{i=1}^{n} \min_{\mathbf{q}_j} |t_i - t_j|.$   (16)
   where $t_j$ is the time at which $\mathbf{q}_j$ occured.
4: Finally, if luminance is available:
       $E_L = \sum_{i=1}^{n} \min_{\mathbf{q}_j} |L(\mathbf{p}_i) - L(\mathbf{q}_j)|.$   (17)
5: Return $E$.
```

value v_x which minimizes the energy function E:

$$\tilde{v}_x = \underset{v_x \in \mathbb{R}}{\operatorname{argmin}}\, E. \tag{18}$$

More elaborate registration techniques to track deformable 3D surfaces may be used for this matching operation. We can mention the most notable ones, Starck and Hilton (2007), Ahmed et al. (2008), and Zeng et al. (2010), that are not using any shapes prior. Accurate registrations are achieved by combined use of several surface features, followed by a coarse to fine scheme. These techniques are however not suitable in their actual form for processing textureless and event-based inputs.

To minimize E with respect to v_x, we also applied a dichotomic search strategy to sample possible values of v_x and match local 3D structure accordingly. Let $R = [R_1, R_r]$ be a real interval that is set large enough at the beginning of the search to make sure it contains $\tilde{v}_x$. Fixing R large enough is only necessary when no recent past estimations of the velocity have been calculated at $\mathbf{p}_i$, otherwise the length of R is defined from the previous estimation of v_x. To determine precisely $\tilde{v}_x$, R is subdivided into r equal length intervals and the centers of all intervals give a set of possible values for v_x. The error cost function is computed for each v_x and the interval producing the smallest E is used to update R.

This operation is iterated until E is below a preset threshold and after a minimum number of iterations. This threshold is defined experimentally with the purpose of optimizing the structure matching process by limiting the search to an acceptable matching error. This threshold can be related to the point cloud density and if it is set to zero, then the maximum number of matching iterations is always performed. r is usually set to 5, however it can be larger. Estimation accuracy increases with r but at the cost of longer processing time.

2.6. Optimal Spatiotemporal Neighborhood

The correct estimation of the velocity is conditioned by the spatiotemporal neighborhood, defined as the spatiotemporal volume of dimensions $(\Delta_x \times \Delta_y \times \Delta_z \times \Delta_t)$, in which the 3D point cloud has moved from time t to $t + dt$. A large neighborhood will allow to find the correct match, but at the cost of processing a large set of data, on the contrary, a too small one will not allow to match the local structures. The spatiotemporal neighborhood must also be resized automatically and dynamically in accordance to the 3D points' velocity. In our implementation, we deal with the problem by adjusting a linear function on the neighborhood size e.g., $\mathbf{s}_k = (\Delta_x, \Delta_y, \Delta_z, \Delta_t)^T$ is a linear combination of the m previous values $\mathbf{s}_{k-1}, ..., \mathbf{s}_{k-m}$:

$$\mathbf{s}_k = \sum_{i=1}^{m} a_i \mathbf{s}_{k-i}, \tag{19}$$

where the coefficients a_i are estimated with a standard linear prediction coding scheme (Durbin, 1959). The value of m is usually set to 5 according to experimental results while the initial value $\mathbf{s}_0$ is deduced from the coarse estimation of the initial velocity i.e., the mean translation between the first two frames. Thus, we have $\mathbf{s}_0 = (\mathbf{v}_0 dt, dt)^T$, assuming $\mathbf{v}_0$ is the initial estimate of the velocity.

The asynchronous 3D flow extraction from points clouds can be achieve by implementing Algorithms 1, 2 and improved if necessary with the optimal neighborhood estimation. This proposed approach does not require clusters of 3D points captured at the same t as one uses to have with frame-based reconstructions, yet it can still be applied if the inputs are frame-based.

3. RESULTS

The first set of experiments are performed on synthetic scenes, where both 3D structures and motion (velocity and trajectory) are known. These results measure the theoretical performance (without noise or reconstruction errors) of our method through comparison between estimated velocity vectors and the known motion.

The second set of experiments are performed on natural scenes, with the purpose of showing the performance of the event-based fitting method when dealing with real data. The algorithm is applied to two sources of 3D data: a Microsoft Kinect (an RGBD sensor that outputs frames of 3D points aligned with RGB information) and an asynchronous event-based 3D reconstruction system as introduced in Carneiro et al. (2013).

3.1. Simulated Scenes

Four simulated scenes are synthetized: (**Figure 5**) a smooth translation of a wire cube at constant amplitude; (**Figure 6**) a 3D car model undergoing a straight translation at 10 m per second;

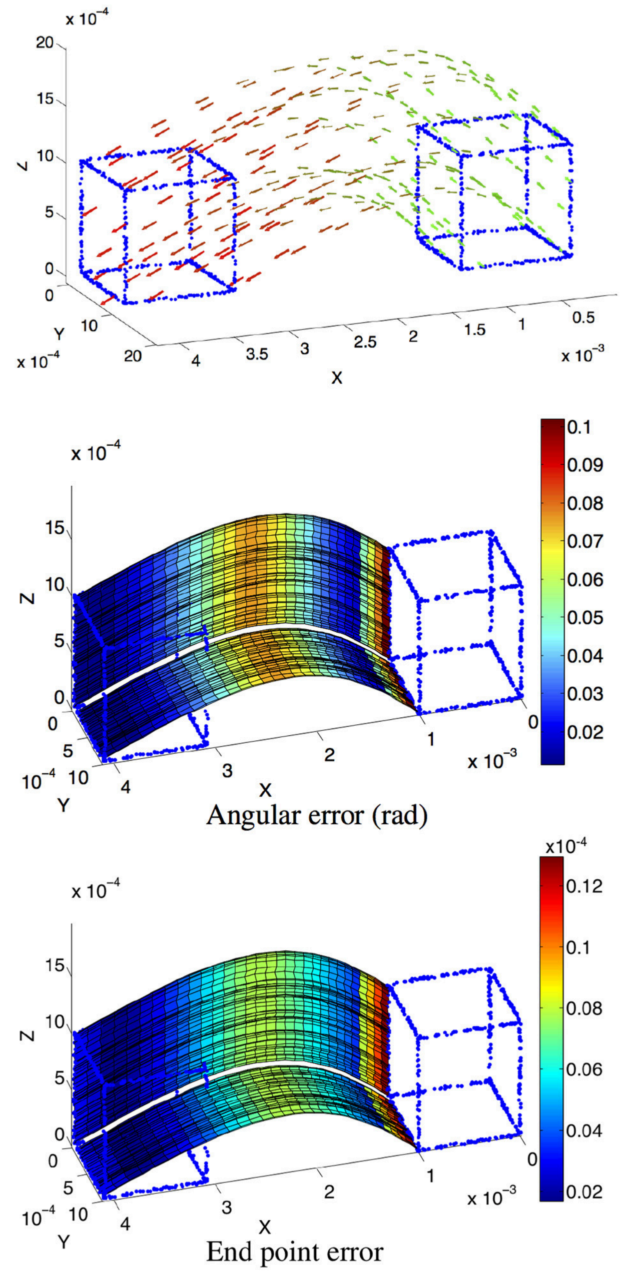

FIGURE 5 | (Top) Scene flow of a cube with the color coding time, as the cube moves from right to left. **(Center and Bottom)** Angular and endpoint errors of the estimated velocity field. The patches of planes are underlined to show the locally constant velocity assumption. For visibility purpose, the velocity is only shown for two edges. All axes are expressed in length unit except for the angle color scale.

(**Figure 7**) the same 3D car model describing a circular motion; (**Figure 8**) a pure rotation of a sphere at constant angular speed.

For each scene, the velocity flow is computed using the geometric structure information alone (only the 3D points' positions and timestamps are given in the simulation). The energy cost function in Algorithm 2 is reduced to E_s. The flow performance is measured by two quantities conventionally used to validate optical flow, the angular error, which is the angle defined by the estimated normalized velocity vector $\tilde{\mathbf{v}}$ and the ground-truth $\mathbf{v}$. The angle is given by the inverse cosine of the scalar product of 2 vectors:

$$\arccos\left(\tilde{\mathbf{v}}^T\mathbf{v}/|\tilde{\mathbf{v}}||\mathbf{v}|\right). \tag{20}$$

This measure has been introduced in Fleet and Jepson (1990) to assess the accuracy of the flow direction. However, the angular error can be biased by large velocity vectors for which the differences in amplitude can be significant and in the same time, the angular errors are small. In that sense the angular error is favoring the large motion over the smaller ones. To compensate for that bias, a second performance measure, the endpoint error, introduced by Otte and Nagel (1994) is conjointly used. This endpoint error is the norm of the difference between the estimated velocity and the real one:

$$|\mathbf{v} - \tilde{\mathbf{v}}|. \tag{21}$$

Both estimated angular error and endpoint error are represented with a color scaled representation (**Figures 5–15**). For the moving cube, the maximal error occurs at the beginning of the motion and is due to the fitting spatio-temporal neighborhood, chosen as the best compromise for the entire motion.

The results on synthetic data, summarized in **Table 1**, show the ability of the method to estimate densely and smoothly the velocity field. The rotating sphere is a challenging because the 3D points composing the surface are not spatially uniform. The non uniform acceleration on the sphere also implies non uniform tangential velocity of 3D points on which we fit the local planes. This explains why the velocity estimation is slightly less accurate for rotations. For translations (cube and car), the velocity is estimated with higher accuracy since the direction has a mean angular error of 0.04 *rad* (with a peak value of 0.1 *rad*) and a mean endpoint error of 0.8%, with a peak value of 1.2% when normalized by the ground-truth objects sizes (respectively the cube edge length, the sphere diameter, and the length of the car). For rotations, the accuracy has the same order of magnitude: around 0.15 *rad* and 2.2% for respectively the angular and the endpoint error.

3.2. Natural Scenes

The second set of results is obtained from real scenes showing a moving person in the scene. The 3D point clouds are provided by a Kinect sensor that also measures the RGB intensity. The Kinect provides depth information for every detected pixel. The background pixels representing the room's walls are removed via depth segmentation. In these sequences, the person is a nice example of a deformable target with limbs moving at different non-constant velocities. However, the local constant speed hypothesis holds. It is sufficient to allow a smooth estimation of the scene flow. Scene flows estimations are given as two sets of results. The first one uses only geometric constraints, when the scene luminance is not available for the structure

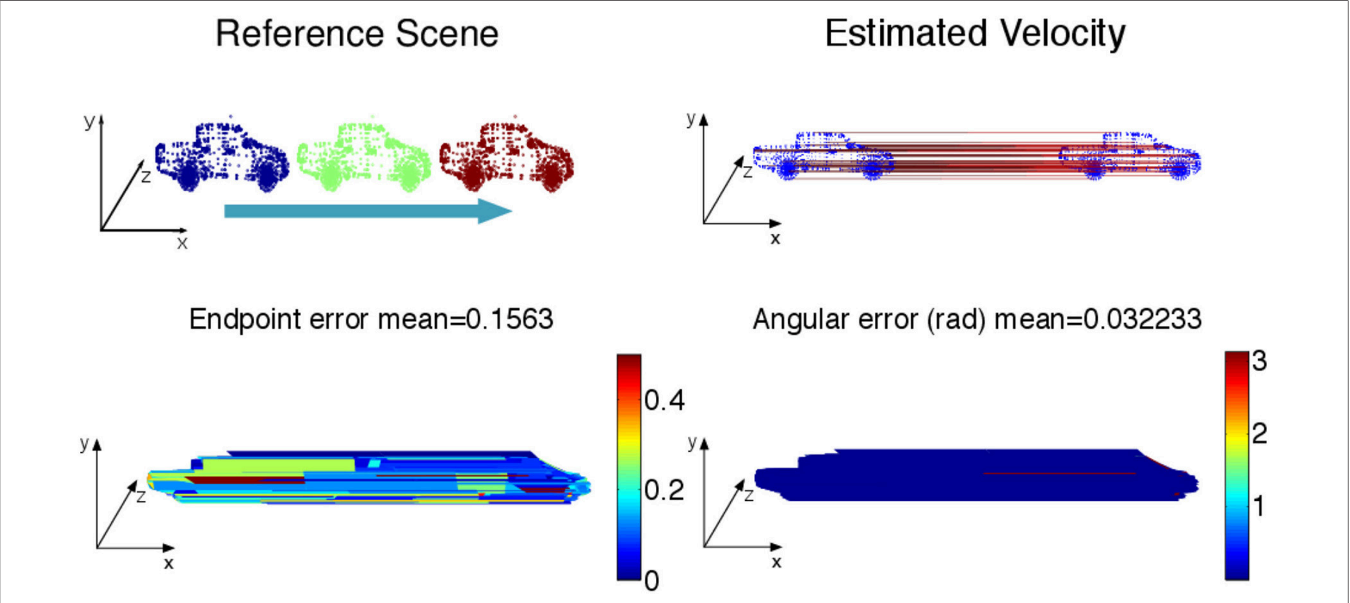

FIGURE 6 | (Top) Constant velocity translation with the color coding time, as the car moves in a straight line from left to right at 10 m/s. The estimated velocity is shown on the right. **(Bottom)** Angular and endpoint errors of the estimated velocity field with the 3D flow technique.

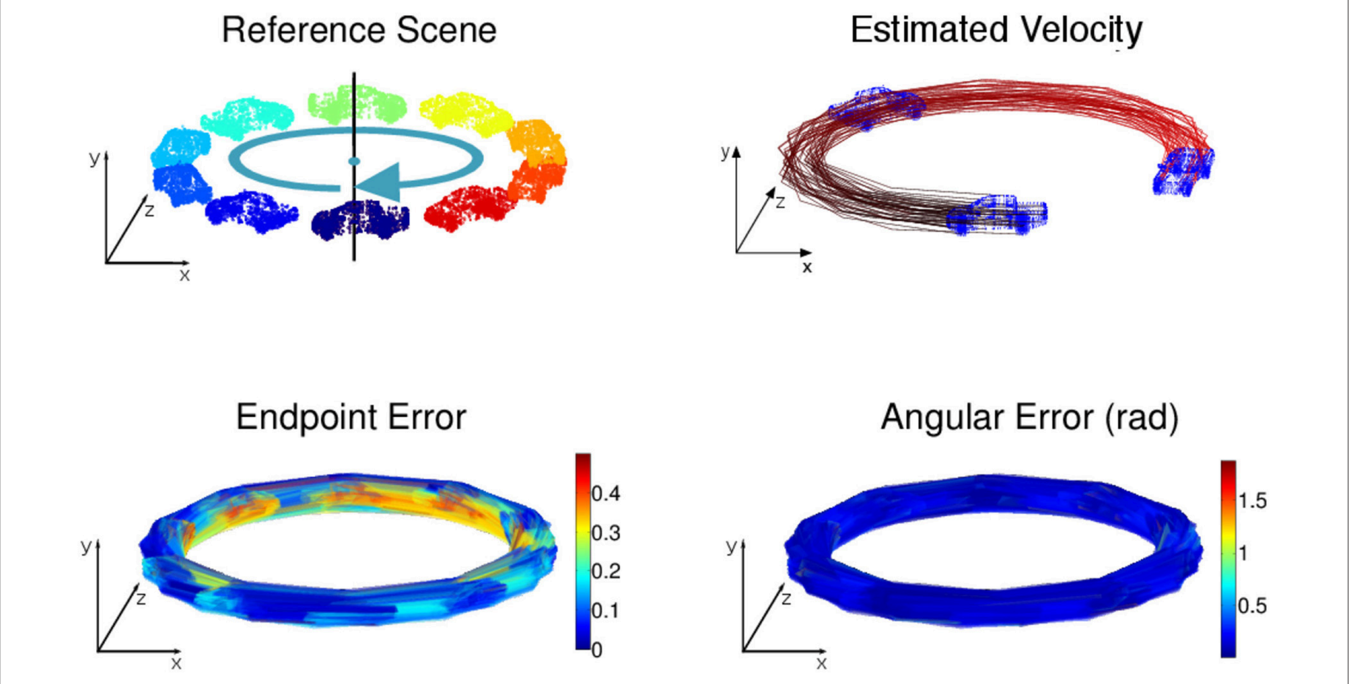

FIGURE 7 | (Top) Circular motion with no tangential acceleration: the car's trajectory is outlined via the velocity estimation. **(Bottom)** Angular and endpoint errors of the estimated velocity field.

registration operation. The second set uses the additional information brought by the luminance in addition to the geometry.

The flow estimation for each sequence is assessed in two ways:

1. A reference speed is established using the person's head to compute speed across frames. The head's position at time t is annotated manually to build a reference motion scene. This is then used as ground-truth to evaluate the event-based fitting method.

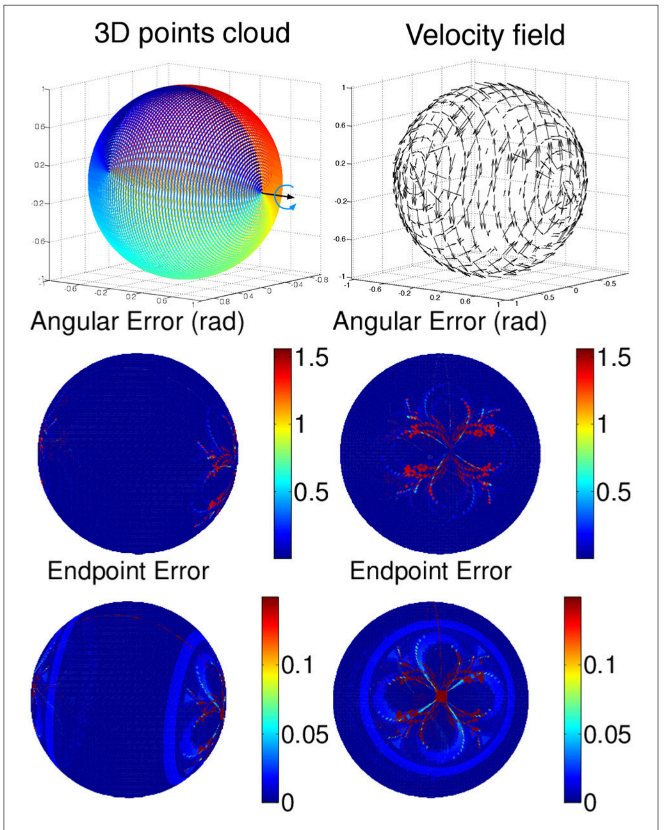

FIGURE 8 | Scene flow of a rotating sphere. (Top) Time color coded representation of the sphere and the velocity field are represented on the top row. Angular and endpoint errors of the estimated velocity field are shown for two points of view. Error are shown in absolute values: the endpoint errors are not exceeding 0.1 length unit in the dark blue regions and is higher as we are close to the poles because of the high density of samples for which the fitting parameters are no optimal enough.

2. If $S(t)$ designates an arbitrary point cloud in the scene at time t then $S(t)+\mathbf{v}dt$ is the morphing of $S(t)$ by the translation vector $\mathbf{v}dt$.

   Let $\mathbf{p}_i \in S(t)$ and $\mathbf{q}_i \in S(t) + \mathbf{v}dt$ such that:

$$\mathbf{q}_i = \underset{\mathbf{q}\in S(t)+\mathbf{v}dt}{argmin} ||\mathbf{q} - (\mathbf{p}_i + \mathbf{v}dt)||. \quad (22)$$

   We define the morphing error as the mean error of each pair $(\mathbf{p}_i, \mathbf{q}_i)$:

$$\frac{1}{N}\sum_i ||\mathbf{p}_i - \mathbf{q}_i||. \quad (23)$$

   This morphing error, normalized by the mean ground-truth velocity amplitude (provided by tracking limbs from the Kinect output for the 3 sequences), is used as the second performance measurement for the rest of the paper.

3.2.1. First Sequence

In the first sequence, shown in **Figure 9**, a person walks in front of the cameras at a constant pace. The velocities' amplitudes, and the directions are shown separately for both estimations without using luminance information. These figures show the performance of the algorithm in the presence of a deformable object. The limbs, in particular, the legs and the fingertips which are subject to the largest velocity changes show clear phases of acceleration: when the legs reach the end of the step, the speed is close to zero (1st and 3rd images), it reaches a maximal value when the legs are in the middle of the step (5th image). The velocity changes are also visible in the color coded motion directions: the silhouettes are not all green as the hands swing. The floor, as it is scanned by the Kinect sensor, was also processed by the algorithm. The estimated speeds are largely coherent with what it is expected: they are close to zero, thus negligible with respect to the moving person. The measured velocity variation (in amplitude and direction) from the floor are mainly due to several sources of noise coming from the sensor itself, the lighting change induced by the motion, etc. The background wall has been removed using depth information before the scene flow estimation is applied. This eliminates any non relevant events/pixel changes due to shadows.

In this experiment, the person walks across the scene, in front of the cameras at a constant speed of 1 m/s. This reference speed is measured by manually segmenting the head's point cloud for each frame. The speed is also extracted for the head from the estimated 3D flow with Algorithm 1. The top row of **Figure 11** shows both speed curves, plot together. Square markers represent the reference speed, circle markers show the speed estimated without luminance information while the diamond markers represent the result achieved with the luminance (through the term E_L in Equation 13). The speed estimated from the geometric constraint has a mean value of $0.99 m.s^{-1}$ and the one using luminance is around $1.2 m.s^{-1}$. The relative mean difference between the two estimations is around 17%. This shows that both estimates are coherent.

The small fluctuations of the estimated speed are not surprising as the trajectory of the head is not a straight translation: body weight transfer happens at each step and it modifies subsequently the head velocity in amplitude and direction. Finally, the color coded flow directions are consistent. Results show that the flow is pointing at 180°, i.e., from right to left for most of the body except for the person's hands. Floor's directions however have a random distribution. We can explain this result by two causes: noise in the acquired data as the floor is a matt surface diffusing randomly the neon lighting and the shape registration procedure in the algorithm which is unlikely to register correctly structures on a uniform surface.

3.2.2. Second Sequence

In the second sequence (**Figure 10**), a more complex motion is tested, showing a person jumping. The velocity amplitude changes several times throughout the sequence: it increases at the beginning and reaches a maximum, then decreases to 0 when the person is at the top of its trajectory. The amplitude then increases again during the fall until he reaches the ground. This sequence of speed change is shown at the bottom row of **Figure 11**. Similarly to the walking sequence, both reference speed curves and estimation are shown together. However, in this experiment, it is more difficult to assess the accuracy of the estimation since

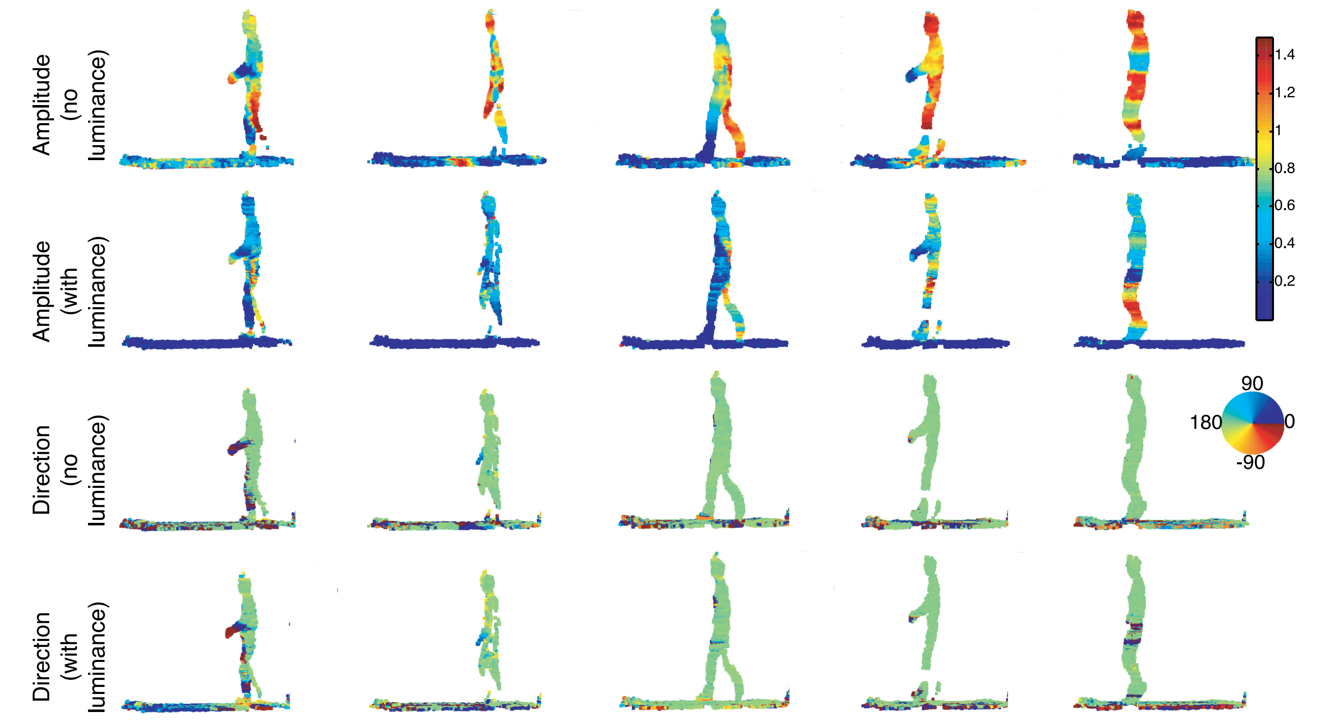

FIGURE 9 | Sequence of a person walking at constant speed across the scene. The amplitude of each 3D point is color-coded and shows that the event-based plane fitting technique is able to estimate non rigid object velocity without and with luminance. A color scale is also used for the flow directions. One can see the person going from the right to the left as confirmed by the color (green in the color scale i.e., an angle of 180°). Again, we can observe that directions can be accurately estimated by using only time and geometry. The addition of luminance provides slight improvements.

the reference speed itself is built with a low accuracy. This is due to the difficulty to manually segment the head's 3D point since the speed changes too quickly.

The jumping sequence is an ideal example of a non-rigid body moving at a totally unconstrained speed. The arms, in particular, show the largest velocity changes since the person swings them to gather momentum from the first half of the jump and he folds them back once the body begins to fall. In this sequence one can also observe the velocity estimated for the floor which is again mostly equal to zero, except at the right under the jumping point because of the moving shadow of the person. The velocities are pointing mainly up (i.e., angle of 90°) during the ascending phase and pointing down when he is falling (i.e., angle of $-90°$).

For both walking and jumping sequences, a higher accuracy is achieved in estimating the velocity when luminance is used, as shown in **Table 2** for five frames taken from the sequences. The mean morphing error is below 3% for the walking sequence and slightly higher than 2% for the jumping one when luminance information is used. The estimation performance is slightly lower when the luminance is removed. In these cases, the morphing errors increase respectively to 5 and 7%. Two main observations should be retained from these results: first, morphed point clouds still consist of well defined objects. This shows the computed motion is consistent for the full scene as morphing objects do not produce incoherent shapes. Secondly, the estimated scene flow is shown being consistent with the real motion since $S(t) + \mathbf{v}dt$, the morphed point cloud, matches correctly $S(t+dt)$, the point cloud at $t + dt$.

3.3. 3D Point Clouds from Event-Based Vision Sensors

This subsection provides the 3D scene flow using event-based cameras (DVS) as described in Carneiro et al. (2013). Computed 3D data have a high resolution of 1 μs. The input to the scene flow estimation are asynchronous 3D point clouds of a hand closing and opening in front of the stereo rig, while the second sequence is a moving face captured by the same stereo rig. The hand speed is of the order of one meter per second while the face moved slower (several cm per second). The first sequence's results are shown in **Figures 12**, **13**. The events generated by the hand's contours are sufficient to estimate 3D flow estimation. The direction and amplitude are consistent with the motion. The mean morphing error, (8.7%), is at the same order of magnitude than the previous experiments.

The second 3D scene flow estimation from the event-based cameras are shown without providing morphing error for readability reason. The second sequence shows a face moving in front of the event-based cameras. The color-coded flow are shown in **Figure 14** with row (a) color-coding the flow's directions while (b), shows the motion amplitudes.

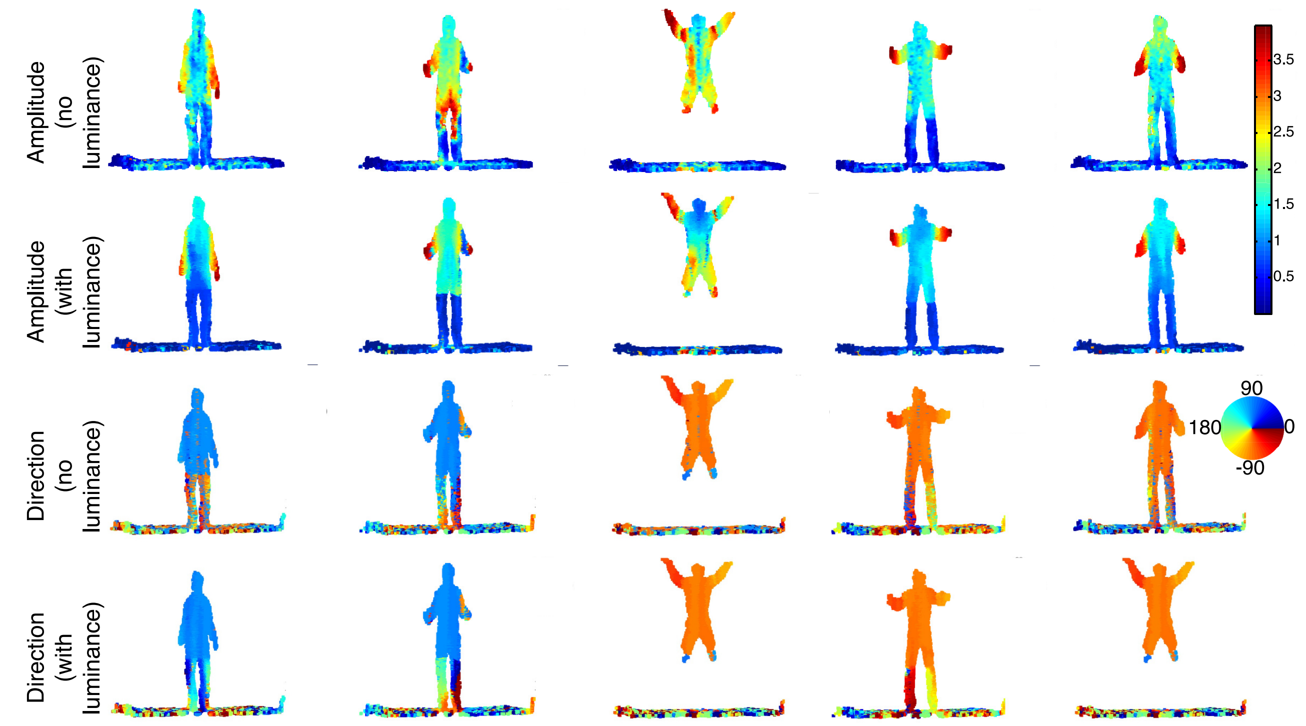

FIGURE 10 | Sequence of a jumping person. This is a complex motion which comprises several rapid changes of the velocity in direction and amplitude. The amplitude plot of the velocity for each 3D points is color-coded. Parts of the body can be segmented according to the velocity e.g., the arms, the legs and rest of the body which have distinctive amplitude. The color coded flow directions (expressed in degree) are well estimated as we can see for the whole body, the direction is pointing up (i.e., angle close to 90°) and pointing to the bottom when the person is falling (i.e., angle around −90°).

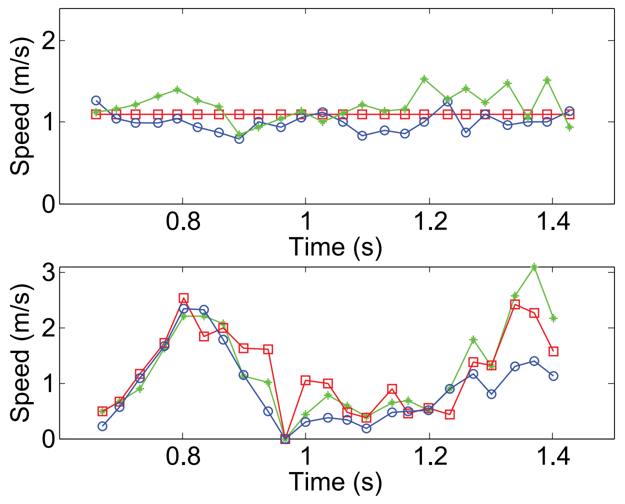

FIGURE 11 | Mean velocity computed for the head in the walking (top) and the jumping **(bottom)** sequences. The circle curves are estimations achieved by time and geometric information. The diamond curves are results one gets when luminance is used for the structure registration. Finally, the square curve represents the velocity of the manually segmented head's 3D points.

3.4. Comparison to the Particle Filter Scene Flow Estimation

This third set of results obtained with the event-based plane fitting technique (without luminance information) is compared to the method published in Hadfield and Bowden (2014). This paper models a set of moving 3D points using a particle filter that supports multiple motion hypotheses to estimate the 3D scene flow from the 3D points provided by a Kinect.

The sequence and its estimated scene flow presented in Hadfield and Bowden (2014) were kindly provided by the authors and are shown as output without additional processing. **Figure 15** shows samples of this sequence along with the estimated 3D scene flow: the velocity amplitude and direction are shown in two separate color-coded representations for the two methods. One can notice that the event-based plane fitting method produces smoother results that are consistant with the scene content, especially the velocity is expected to be maximal at the foot when the kick is accomplished.

We provide the morphing error in **Table 3**. It shows the error computed for 5 frames taken at some arbitrary regular time interval, using both method. This is consistant with the previous analysis showing that the event-based plane fitting method's performance is higher than the

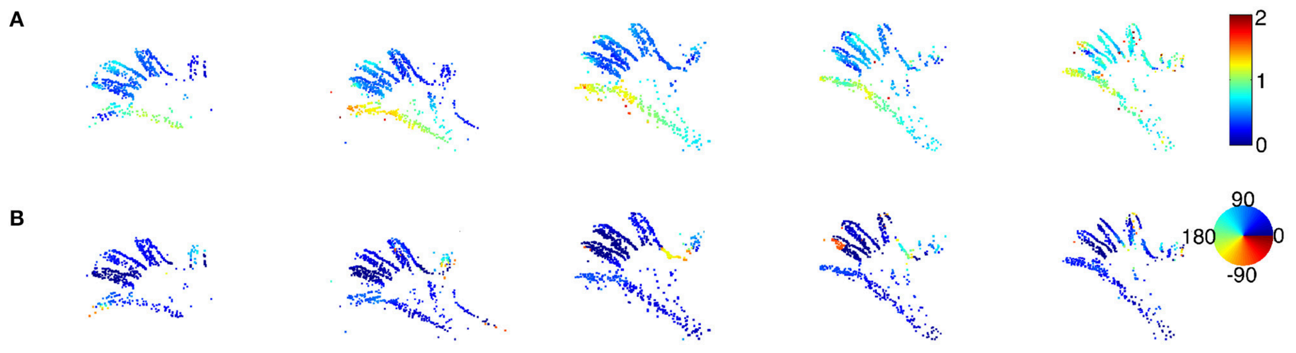

FIGURE 12 | A sequence of a moving hand acquired using a stereo-rig composed of event-based silicon retinas (DVS) as sets of event {(p, *t*)}. No luminance information is available from these sensors. In **(A,B)** color-coded plots of the estimated amplitude and direction of the velocity for each reconstructed 3D points. In contrast with the previous resultsare shown, they correspond to the locations of dense 3D points. The motion is a smooth translation from left to right as shown by the direction plots pointing toward 45°.

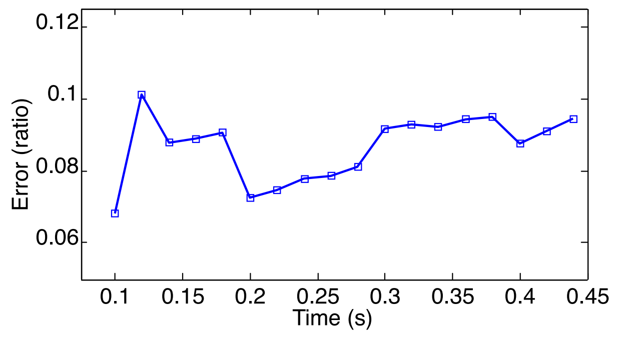

FIGURE 13 | Morphing error for the event-based 3D point clouds of a waving hand calculated for a set of randomly selected times. The Mean error is around 8.7%, this is consistant with previous estimations if no intensity information is used or available.

TABLE 1 | Average motion estimation errors for the synthetic scenes.

	Cube translation	Car cst. translation	Car circular motion	Rotating sphere
Angular error (rad)	0.04	0.03	0.15	0.15
Endpoint error (%)	0.8	1.5	2.2	2

TABLE 2 | Morphing error for several sets of five randomly selected times in the sequences.

	Morphing error (ratio)			
Frame	**Walking**		**Jumping**	
	Without *L*	**With *L***	**Without *L***	**With *L***
1	0.0542	0.0214	0.0250	0.0131
2	0.0695	0.0277	0.0635	0.0270
3	0.0786	0.0221	0.0698	0.0414
4	0.0863	0.0424	0.0510	0.0128
5	0.0433	0.0278	0.0481	0.0143
Mean	0.0664	0.0283	0.0515	0.0217

Errors are measured with and without luminance information. In the first case, the mean error are around 6.6 and 5.2% respectively. These ratios are improved to 2.8 and 2.2% when luminance is added i.e., errors are reduced by a factor close to 2.

state-of-the-art frame-based technique. Unsurprisingly, the overall estimation accuracy for both methods is lower than the previous experiments mainly because the Kinect's inability to accurately capture fast motions that give rise to blurry images.

4. DISCUSSION

This paper introduced a new technique of dense 3D scene flow estimation. This is so far, the first 3D scene flow algorithm developed for asynchronous sensing using event-based cameras. The event-based formulation of the flow applies the rule of "one event equals one computation" that allows an incremental update of the 3D scene flow in an almost continuous manner. This formulation allows also a straightforward extension to frame-based representation as long as time is used as the main computation feature.

The motion inference and 3D reconstruction from multiple cameras are usually coupled tasks in frame-based computer vision. They are solved by stereovision mechanisms which require highly accurate calibration operation. Conventionally, the dense scene flow is estimated and refined from the dense optical flow which is its projection on the image planes in an iterative feedback loop scheme. The scene flow computation is therefore a complex problem which is an optimization problem under several conflicting constraints.

The proposed technique is based on the local constant motion of the 3D point clouds and on their locally non deformable geometry. These hypotheses, when satisfied, tell us that an object moving through space, locally generates ruled surfaces from which the velocity vectors can be extracted. The solution we proposed is simple as it constraints the 3D velocity estimation to a search for a parametrization value over the set of real numbers.

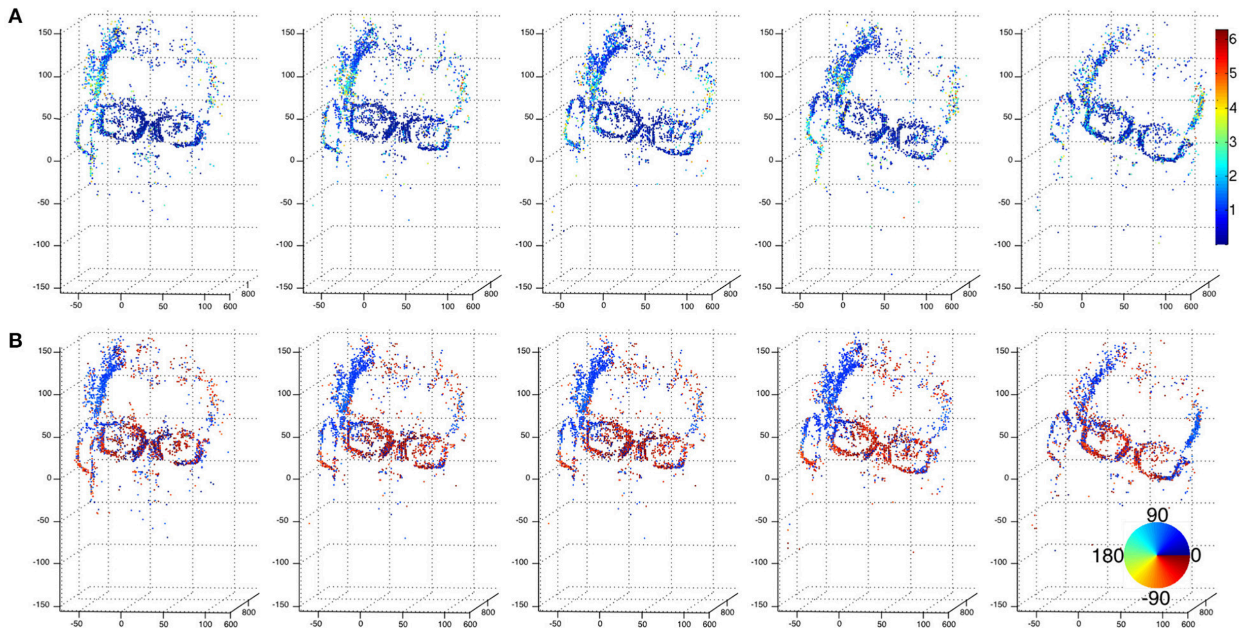

FIGURE 14 | Sequence of a moving face 3D point clouds computed using event-based cameras. (A) Shows the color coded flow's amplitude while **(B)** is its color coded directions in a fronto parallel plane. The computed flow outlines the actual motion of the face that rotates from right to left: temples motion is pointing to 30° (blue) while the glasses motion is almost horizontal and pointing to the right (dark red). The global motion is close to be constant over the samples shown by the figure. The amplitude is expressed in cm/s.

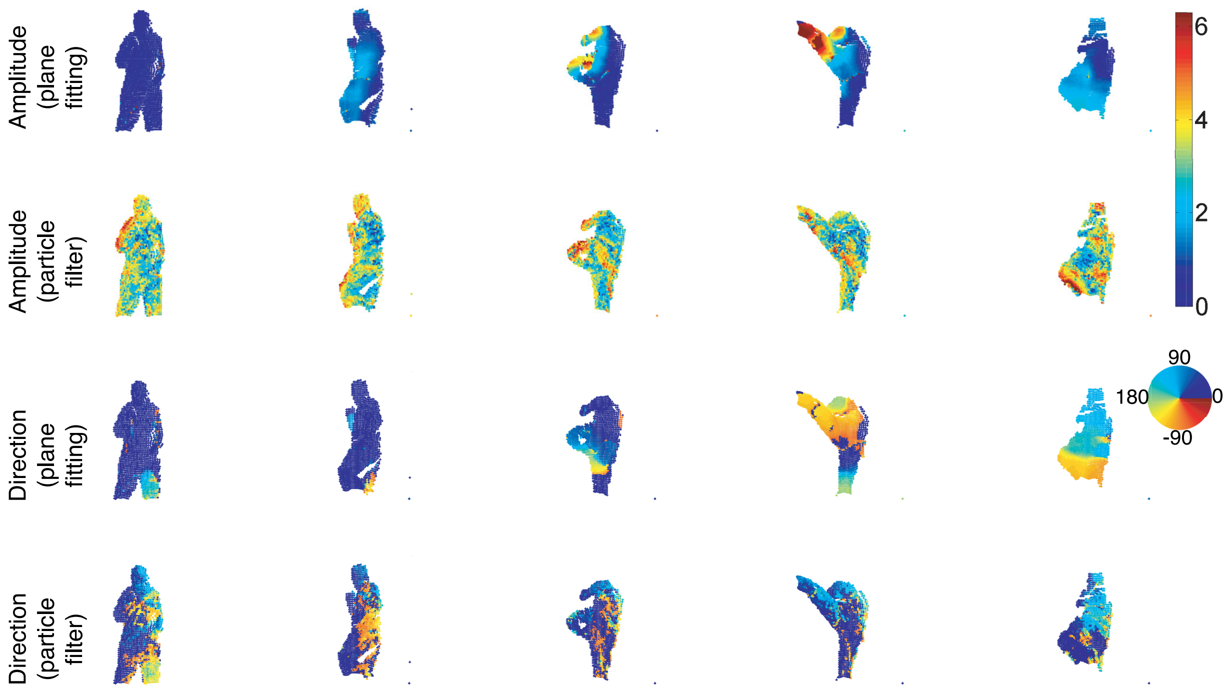

FIGURE 15 | Sequence of a person performing a kick. The amplitude of each 3D point is color-coded and shows respectively the flow amplitude estimation using the event-based plane fitting and the particle filter method proposed in Hadfield and Bowden (2014). The flow direction estimation from the event-based plane fitting and the particle filter methods are shown in the 2 last lines.

TABLE 3 | Event-based plane fitting technique compared to particle filtering technique.

	Morphing error (ratio)	
Frame	**Plane fitting**	**Scene particle**
1	0.0124	0.0265
2	0.0382	0.0461
3	0.0496	0.0505
4	0.0344	0.0378
5	0.0436	0.0480
Mean	0.0356	0.0418

Errors are normalized by the mean velocity of the left leg.

To achieve this search, we developed a local 3D structure matching strategy using the geometric consistency and when it is available, luminance as an additional constraint to identify structures across time. Experimental results obtained from synthetic and natural scenes show the technique to be particularly suitable in estimating the velocity vectors of deformable objects, undergoing arbitrary unconstrained motions. This approach allows flow estimation from any data output by sensors that capture the spatiotemporal information that but do not necessarily provide nor use luminance such as range finders (e.g., the LiDAR).

The method also provides a dense estimation of the velocity field as an alternative method to using a variational formulation (a very powerful but also highly computationally demanding technique) for flow estimation. Since the plane fitting we applied for the flow regularization is relatively inexpensive operation, the resources are mainly needed for the structure matching operation. We can sketch an idea about the complexity of that matching operation according to Algorithm 1:

- There are 2 nested loops (line 6 and 8 in the algorithm), if we assume that the while loop is satisfied in n iterations and that we have n values of v_k to test in the for loop, then the algorithm is at least in $\mathcal{O}(n^2)$.
- In the case of a stream of z events, the complexity is in $\mathcal{O}(z.n^2)$.

From that perspective, the complete scene flow algorithm‘ complexity is at least in $\mathcal{O}(z.n^2)$. We can reasonably state that this is still less complex compared to conventional scene flow estimation techniques based on particular filter or variational approaches.

AUTHOR CONTRIBUTIONS

SI: drafting the work and revised the final version for submission. JC: conception and design of the work; data acquisition and analysis; drafting the paper. RB: drafting the work and revising it critically.

ACKNOWLEDGMENTS

The authors are grateful to Simon Hadfield for providing us data and results without which we could not have achieve in a short time accurate comparison results. They also thank Marc Osswald for providing additional 3D event-based point clouds used in the paper. Finally, this work was supported by the LABEX LIFESENSES [ANR-10-LABX-65] which is supported by French state funds managed by the ANR within the Investissements d’Avenir programme [ANR-11-IDEX-0 004-02].

REFERENCES

Ahmed, N., Theobalt, C., Rossl, C., Thrun, S., and Seidel, H.-P. (2008). “Dense correspondence finding for parametrization-free animation reconstruction from video,” in *Conference on Commputer Vision and Pattern Recognition* (Miami Beach, FL). doi: 10.1109/cvpr.2008.4587758

Basha, T., Moses, Y., and Kiryati, N. (2013). Multi-view scene flow estimation: a view centered variational approach. *Int. J. Comput. Vis.* 101, 6–21. doi: 10.1007/s11263-012-0542-7

Benosman, R., Clercq, C., Lagorce, X., Ieng, S.-H., and Bartolozzi, C. (2014). Event-based visual flow. *IEEE Trans Neural Netw Learn Syst.* 25, 407–417. doi: 10.1109/TNNLS.2013.2273537

Besl, P., and McKay, H. (1992). A method for registration of 3-D shapes. *IEEE Trans. Patt. Anal. Mach. Intell.* 14, 239–256. doi: 10.1109/34.121791

Cagniart, C., Boyer, E., and Ilic, S. (2010). “Free-form mesh tracking: a patch-based approach,” in *Conference on Vision and Pattern Recognition* (San Francisco, CA). doi: 10.1109/cvpr.2010.5539814

Carceroni, R., and Kutulakos, K. (2002). Multi-view scene capture by surfel sampling: from video streams to non-rigid 3D motion, shape and reflectance. *Int. J. Comput. Vis.* 49, 175–214. doi: 10.1023/A:1020145606604

Carneiro, J., Ieng, S.-H., and Benosman, R. (2013). Asynchronous event-based 3D reconstruction from neuromorphic retinas. *Neural Netw.* 45, 27–38. doi: 10.1016/j.neunet.2013.03.006

Delbrück, T., Linares-Barranco, B., Culurciello, E., and Posch, C. (2010). “Activity-driven, event-based vision sensors,” in *International Symposium on Circuits and Systems* (Paris: IEEE), 2426–2429.

Durbin, J. (1959). Efficient estimation of parameters in moving average models. *Biometrica* 46, 306–317. doi: 10.1093/biomet/46.3-4.306

Fleet, D. J., and Jepson, A. (1990). Computation of component image velocity from local phase information. *Int. J. Comput. Vis.* 5, 77–104. doi: 10.1007/BF00056772

Hadfield, S., and Bowden, R. (2014). Scene particles: unregularized particle based scene flow estimation. *IEEE Trans Patt. Anal. Mach. Intell.* 36, 564–576. doi: 10.1109/TPAMI.2013.162

Hansard, M., Lee, S., Choi, O., and Horaud, R. (2012). *Time-of-Flight Cameras: Principles, Methods and Applications*. Paris: Springer Briefs in Computer Science.

Herbst, E., Ren, X., and Fox, D. (2013). “RGB-D flow: Dense 3-D motion estimation using color and depth,” in *IEEE International Conference on Robotics and Automation* (Karlsruhe: ICRA). doi: 10.1109/icra.2013.6630885

Huguet, F., and Devernay, F. (2007). “A variational method for scene flow estimation from stereo sequences,” in *International Conference on Computer Vision* (New York, NY: Springer; IEEE), 1–7. doi: 10.1109/iccv.2007.4409000

Isard, M., and MacCormick, J. (2006). “Dense motion and disparity estimation via loopy belief propagation,” in *Proceedings of the 7th Asian conference on Computer Vision - Volume Part II, ACCV’06* (Berlin; Heidelberg: Springer-Verlag), 32–41. doi: 10.1007/11612704_4

Khoshelham, K., and Elberink, S. O. (2012). Accuracy and resolution of kinect depth data for indoor mapping applications. *Sensors* 12, 1437–1454. doi: 10.3390/s120201437

Lichtsteiner, P., Posch, C., and Delbruck, T. (2008). A 128 × 128 120 DB 15 μs latency asynchronous temporal contrast vision sensor. *IEEE J. Solid State Circ.* 43, 566–576. doi: 10.1109/JSSC.2007.914337

Mahowald, M. (1992). *VLSI Analogs of Neuronal Visual Processing: A Synthesis of Form and Function*. Ph.D. thesis, California Institut of Technology.

Maybank, S. (1993). *Theory of Reconstruction from Image Motion*. Springer series in information sciences. New York, NY: Springer-Verlag. doi: 10.1007/978-3-642-77557-4

Min, D., and Sohn, K. (2006). "Edge-preserving simultaneous joint motion-disparity estimation," in *Proceedings of the 18th International Conference on Pattern Recognition - Volume 02, ICPR '06* (Washington, DC: IEEE Computer Society), 74–77.

Otte, M., and Nagel, H.-H. (1994). "Optical flow estimation: advances and comparisons," in *Computer Vision — ECCV '94, Volume 800 of Lecture Notes in Computer Science*, ed J.-O. Eklundh (Berlin; Heidelberg: Springer), 49–60.

Park, J., Oh, T., Jung, J., Tai, Y., and Kweon, I. (2012). "A tensor voting approach for multi-view 3D scene flow estimation and refinement," in *European Converence on Computer Vision* (Florence).

Popham, T., Bahalerao, A., and Wilson, R. (2010). "Multi-frame scene-flow estimation using a patch model and smooth motion prior," in *Proceedings of the BMVC 2010 UK Postgraduate Workshop* (Aberystwyth).

Posch, C., Matolin, D., and Wohlgenannt, R. (2011). A QVGA 143 dB dynamic range frame-free PWM image sensor with lossless pixel-level video compression and time-domain CDS. *IEEE J. Solid State Circ.* 46, 259–275. doi: 10.1109/JSSC.2010.2085952

Rogister, P., Benosman, R., Ieng, S.-H., Lichtsteiner, P., and Delbrück, T. (2012). Asynchronous event-based binocular stereo matching. *IEEE Trans. Neural Netw. Learning Syst.* 23, 347–353. doi: 10.1109/TNNLS.2011.2180025

Sommerville, D. (1934). *Analytic Geometry of Three Dimensions*. Cambridge: Cambridge University Press.

Starck, J., and Hilton, A. (2007). "Correspondence labelling for wide-timeframe free-form surface matching," in *International Conference on Computer Vision* (Rio de Janeiro). doi: 10.1109/iccv.2007.4409108

Tomasi, C., and Zhang, J. (1995). "Is structure-from-motion worth pursuing?," in *Proceedings of the Seventh International Symposium on Robotics Research*. (Springer: London).

Varanasi, K., Zaharescu, A., Boyer, E., and Horaud, R. (2008). "Temporal surface tracking using mesh evolution," in *European Conference on Computer Vision* (Marseille).

Vedula, S., Rander, P., Collins, R., and Kanade, T. (1999). "Three-dimensional scene flow," in *International Conference on Computer Vision*, Vol. 27-3, (Toronto, ON) 475–480. doi: 10.1109/iccv.1999.790293

Wedel, A., Brox, T., Vaudrey, T., Rabe, C., Franke, U., and Cremers, D. (2011). Stereoscopic scene flow computation for 3D motion understanding. *Int. J. Comput. Vision* 95, 29–51. doi: 10.1007/s11263-010-0404-0

Zeng, Y., Wang, C., and Wang, Y. (2010). "Dense non-rigid surface registration using high-order graph matching," in *International Conference on Computer Vision* (San Francisco, CA).

Zhang, Y., Kambhamettu, C., and Kambhamettu, R. (2001). "On 3D scene flow and structure estimation," in *IEEE Conference on Computer Vision and Pattern Recognition* (Kauai), 778–785. doi: 10.1109/cvpr.2001.991044

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.