Scene Flow by Tracking in Intensity and Depth Data

Julian Quiroga, Frédéric Devernay, James L. Crowley

Scene Flow by Tracking in Intensity and Depth Data

Julian Quiroga Frédéric Devernay James Crowley
PRIMA team, INRIA Grenoble Rhone-Alpes
Montbonnot, France
{julian.quiroga,frederic.devernay,james.crowley}@inria.fr

Abstract

The scene flow describes the motion of each 3D point between two times steps. With the arrival of new depth sensors, as the Microsoft Kinect, it is now possible to compute scene flow with a single camera, with promising repercussion in a wide range of computer vision scenarios. We propose a novel method to compute scene flow by tracking in a Lucas-Kanade framework. Scene flow is estimated using a pair of aligned intensity and depth images, but rather than computing a dense scene flow as in most previous methods, we get a set of 3D motion fields by tracking surface patches. Assuming a 3D local rigidity of the scene, we propose a rigid translation flow model that allows to solve directly for the scene flow by constraining the 3D motion field both in intensity and depth data. In our experimentation we achieve very encouraging results. Since this approach solves simultaneously for the 2D tracking and for the scene flow, it can be used for action recognition in existing 2D tracking based methods or to define scene flow descriptors.

1. Introduction

The scene flow corresponds to the 3D motion field of the scene [31] and since it provides the motion of 3D points it can be used in several tasks, such as object segmentation, motion analysis and tracking. In particular, in human activity understanding the scene flow can be useful to provide features for classification and recognition algorithms. However, there is not work that directly compute scene flow and perform tasks like human action recognition. Probably, this is due to the fact that most existing methods compute a dense scene flow, spending a lot of processing time, and they require a stereo or multi-view camera systems which are not always available. On the other hand, the optical flow, that corresponds to the scene flow projection onto the image plane, has been successfully used in human action recognition. Optical flow is commonly used in state of the art techniques as part of descriptors formed by histograms [17, 26] or computed to extract 2D trajectories [23, 22, 27].

With the arrival of depth cameras, based either on time-of-flight (ToF) or structured light sensing, it has been possible to compute scene flow using a single image sequence as in [18, 13]. Moreover recently, some attempts have been made to include depth data in human action recognition tasks. A bag of 3D points extracted from the depth data is used for recognition in [19] while in [25] descriptors obtained by well known techniques [17, 6] were complemented with depth information, and outperform the original methods. Depth sensors have decreased system requirements needed to compute 3D motion field, opening the door to incorporate scene flow based features in common recognition tasks.

We propose a method to compute scene flow in a Lucas-Kanade framework [2] by using an aligned pair of intensity and depth images. Assuming a 3D local rigidity of the scene, we model the image flow induced by the surface motion by means of a *rigid translation flow model* which allows to constrain the 3D motion field in both intensity and depth images. Using the depth information, our approach improves tracking precision in the image domain and allows at same time to compute scene flow. We solve simultaneously for the 2D tracking and for the scene flow so that this method can be used directly in existing 2D tracking based methods or to define scene flows descriptors.

1.1. Related work

Scene flow was first introduced by Vedula *et al.* [31] as the full 3D motion field in the scene. Most scene flow methods assume a stereo, or multi-view camera system, in which the motion and the geometry of the scene are jointly estimated, in some cases, under a known scene structure. Since optical flow is the projection of the 3D motion field onto the camera image plane, an intuitive way to compute scene flow is to reconstruct it from the optical flow measured in a multiple camera system, as proposed by Vedula *et al.* in [32]. However, it is difficult to recover a scene flow compatible with several observed optical flows which may be contradictory.

The most common approach to estimating scene flow

is to perform an optimization on a global energy function, including photometric constraint, and some regularization. Some authors introduce constraints of a full calibrated stereo structure. Huguet and Devernay [16] simultaneously compute the optical flow field and two disparities maps, while Wedel *et al.* [34] decouple the disparity at the first time step. Valgaerts *et al.* [30] assume that only the camera intrinsics are known and they show that scene flow and the stereo structure can be estimated.

All these methods suffer from the smoothness constraints brought by 2D parametrization. Basha *et al.* [4] improve the estimation by formulating the problem as a point cloud in 3D space and the scene flow is regularized using total variation (TV). Recently, Vogel *et al.* [33] regularize the problem by encouraging a locally rigid 3D motion field, outperforming TV regularization. Other methods simultaneously solve the 3D surface and motion [24, 11].

When a depth camera is available, the sensor provides structure information and surface estimation is not needed. Spies *et al.* [29] estimate the scene flow by simultaneously constraining the flow in intensity and depth images of an orthographically captured surface. Lukins and Fisher [21] extend this approach to multiple color channels and an aligned depth image. Letouzey *et al.* [18] directly estimate the 3D motion field using photometric constraints and a global regularization term. Recently, Hadfield and Bowden [13] estimate the scene flow by modeling moving points in 3D using a particle filter, reducing the over-smoothing caused by global regularization. In a similar way, we estimate the scene flow using a pair of aligned intensity and depth images but rather than computing a dense scene flow we get a set of 3D motion fields by tracking surface patches.

The work in this paper is inspired by that of Devernay *et al.* [9], in which a sparse scene flow is derived from 3D trajectories using several cameras. In our approach, instead of tracking 3D points we use a Lucas-Kanade tracking framework [2] to directly calculate the scene flow of small surface patches. As in [33], we assume that the scene is composed of independently, but rigidly moving 3D parts, avoiding smoothness constraints in the image domain used in [29, 21]. Unlike previous Lucas-Kanade methods that use a 2D warping [2], we model the image flow as a function of the 3D motion field with help from a depth sensor, improving the accuracy of the optical flow and solving directly for the scene flow. Without expecting a planar surface as in surfels based techniques [9], this motion model allows the constraint of scene flow in intensity and depth images, and its extension to several cameras is straightforward.

2. Motion model

The instantaneous motion, relative to the camera, of a rigid surface in the scene can be decomposed into two components: a translation velocity $\mathbf{V} = (V_X, V_Y, V_Z)$ and an angular velocity $\mathbf{W} = (\Omega_X, \Omega_Y, \Omega_Z)$. The camera frame is defined with its origin at the optical center and its (X, Y, Z) axes respectively in the direction of the image axes and the optical axis, with positive Z pointing in the direction of sight. Let $\mathbf{X} = (X, Y, Z)$ be coordinates at time $t - 1$ of a surface point in the camera reference frame and let $\mathbf{X}' = (X', Y', Z')$ be the corresponding coordinates at time t. If, between $t - 1$ and t, the surface performs a rotation $\mathbf{W}$ followed by a translation induced by velocity $\mathbf{V}$, then

$$\mathbf{X}' = \mathbf{R}\mathbf{X} + \mathbf{V}, \tag{1}$$

where the rotation matrix $\mathbf{R}$ can be approximated (assuming small values of $\mathbf{W}$ [10]) by

$$\mathbf{R} = \begin{pmatrix} 1 & -\Omega_Z & \Omega_Y \\ \Omega_Z & 1 & -\Omega_X \\ -\Omega_Y & \Omega_X & 1 \end{pmatrix}. \tag{2}$$

2.1. Rigid translation model

Let $\mathbf{x} = (x, y)$ be the projection of a surface point $\mathbf{X}$ on the image plane, if camera matrix $\mathbf{M}$ is given by

$$\mathbf{M} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \tag{3}$$

then $x = \frac{X}{Z} f_x + c_x$ and $y = \frac{Y}{Z} f_y + c_y$ (we suppose in the rest of this paper that nonlinear distortion was removed from the images). In a similar way the image coordinates $\mathbf{x}' = (x', y') = \hat{\mathbf{M}}(\mathbf{X}')$ is the projection of the point $\mathbf{X}'$ where $\hat{\mathbf{M}}$ is the projective function. The 3D movement of the surface generates a motion of its projection. Let (u, v) be the optical flow induced on the pixel $\mathbf{x}$ by rotation and translation of the surface, then

$$u = x' - x = \left(\frac{X - Y\Omega_Z + Z\Omega_Y + V_X}{Z - X\Omega_Y + Y\Omega_X + V_Z} - \frac{X}{Z} \right) f_x \tag{4}$$

and

$$v = y' - y = \left(\frac{Y - X\Omega_Z + Z\Omega_X + V_Y}{Z - X\Omega_Y + Y\Omega_X + V_Z} - \frac{Y}{Z} \right) f_y \tag{5}$$

We define our *Rigid Translation Flow Model* assuming that $i)$ the contribution of the inter-frame rotation is negligible and $ii)$ the Z component of the translation velocity is negligible w.r.t. Z, i.e., $|V_Z/Z| \ll 1$. Then we have:

$$u = \frac{f_x}{Z} \left(V_X - \frac{X}{Z} V_Z \right) = \frac{1}{Z} \left[f_x V_X - (x - c_x) V_Z \right] \tag{6}$$

and

$$v = \frac{f_y}{Z} \left(V_Y - \frac{Y}{Z} V_Z \right) = \frac{1}{Z} \left[f_y V_Y - (y - c_y) V_Z \right]. \tag{7}$$

Therefore, the pixel motion induced on a pixel (x, y) by the surface translation is modeled by

$$\begin{pmatrix} u \\ v \end{pmatrix} = \frac{1}{Z} \begin{pmatrix} 1 & 0 & -x \\ 0 & 1 & -y \end{pmatrix} \begin{pmatrix} V_X \\ V_Y \\ V_Z \end{pmatrix} \tag{8}$$

where, for brevity and without loss of generality, we have considered the focal lengths as the unit and the optical center at the image origin. The resulting image flow in expression (8) agrees with that of Longuet *et al.* in [20] for the translational component the instantaneous velocity of a retinal image point.

2.2. Other motion models

The inter-frame image flow induced by the 3D motion of a rigid surface has been modeled in the literature using different assumptions. The instantaneous velocity of a retinal point (x, y) is modeled in [20] as

$$u = (V_X - xV_Z)\frac{1}{Z} - xy\Omega_X + \left(1 + x^2\right)\Omega_Y - y\Omega_Z \tag{9}$$

$$v = (V_Y - yV_Z)\frac{1}{Z} - \left(1 + y^2\right)\Omega_X + xy\Omega_Y + y\Omega_Z \tag{10}$$

with $(u, v) = \left(\frac{dx}{dt}, \frac{dy}{dt}\right)$, and (V_X, V_Y, V_Z) and $(\Omega_X, \Omega_Y, \Omega_Z)$ the instantaneous translation and angular velocities, respectively. Horn *et al.* in [15] use this instantaneous velocity model to describe the inter-frame pixel motion where velocities become displacements between $t-1$ and t, and flow is defined as $(u, v) = (x'-x, y'-y)$. Adiv in [1] shows that equations (9) and (10) can be used to approximate the optical flow defined by (4) and (5) under the assumptions that the view of field of the camera is not very large, and $V_Z T_Z/Z$ ratio and rotation velocities are very small. This model is called the *Rigid Body Model* in the hierarchy classification of motions proposed by [5], and it is characterized by 6 degrees of freedom which are functions of the 3D motion and the depth Z.

If the optical flow corresponds to the projection of a planar surface $k_X X + k_Y Y + k_Z Z = 1$, the parameters of the plane constrain the flow induced by the 3D motion of the surface. This optical flow model is called *Planar Surface Flow* in [5] and according to [1] it can be expressed in function of 8 parameters as

$$\begin{aligned} u &= a_1 + a_2 x + a_3 y + a_7 x^2 + a_8 xy \\ v &= a_4 + a_5 x + a_6 y + a_7 xy + a_8 y^2, \end{aligned} \tag{11}$$

where coefficients $\{a_1, ..., a_8\}$ are functions of the motion parameters $\{(V_X, V_Y, V_Z), (\Omega_X, \Omega_Y, \Omega_Z)\}$ and surface parameters (k_X, k_Y, k_Z). This 8 DOF model is also known as the *Homography Model* in [2].

In [5] the surface is considered to be far from the camera and rotation components Ω_x and Ω_y are assumed to be zero, which leads to the *Affine Flow Model*, defined as

$$\begin{aligned} u &= a_1 + a_2 x + a_3 y \\ v &= a_4 + a_5 x + a_6 y, \end{aligned} \tag{12}$$

and the flow induced is modeled by the coefficients $\{a_1, ..., a_6\}$ which are functions of the motion parameters (V_X, V_Y, V_Z), the depth Z and the rotation Ω_Z. Finally, the most simple model is defined assuming negligible angular velocities and a orthographic camera model in which Z displacements do not affect the projection of the 3D point in the image plane. In this model the optical flow is given by $u = a_1$ and $v = a_2$ where a_1 and a_2 are functions of V_X and V_Y, respectively.

3. Locally rigid tracking approach

The goal of Lucas-Kanade algorithm is to align a template image $T(\mathbf{x})$ to an input image $I(\mathbf{x})$, so that, it allows to compute optical flow or to track an image patch between two frames. Following [2], this problem can be stated as finding the parameter vector $\mathbf{P} = (p_1, ..., p_n)^T$ which minimizes a squared sum of intensities between the template T an the current image I:

$$\mathbf{P} = \arg\min_{\mathbf{P}} \sum_{\mathbf{x}} \left[I\left(\mathbf{W}(\mathbf{x}; \mathbf{P})\right) - T\left(\mathbf{x}\right)\right]^2, \tag{13}$$

where the warp function $\mathbf{W}(\mathbf{x}; \mathbf{P})$ maps each template pixel $\mathbf{x}$ to a pixel in the image. The warp function can be defined using a motion model such as those presented in Section 2. In our approach, we assume that all template pixels belong to a rigid surface in the scene, which performs a translational motion between consecutive frames.

3.1. Locally rigid warp

Let $I^t(\mathbf{x})$ be the intensity of the image pixel $\mathbf{x}$ in time t. Under brightness constancy assumption, points $\mathbf{X}$ at time $t-1$ and $\mathbf{X}' = \mathbf{X} + \mathbf{V}$ at time t are projected with the same intensity in the image according to

$$I^{t-1}(\hat{\mathbf{M}}(\mathbf{X})) = I^t(\hat{\mathbf{M}}(\mathbf{X}')) \tag{14}$$

and

$$I^{t-1}(\mathbf{x}) = I^t(\mathbf{x}') \tag{15}$$

where $\hat{\mathbf{M}}$ is the projective function and, $\mathbf{x}$ and $\mathbf{x}'$ correspond to the projections of 3D points $\mathbf{X}$ and $\mathbf{X}'$, respectively. Let $T(\mathbf{x})$ stand for the previous frame $I^{t-1}(\mathbf{x})$ (intensity template) and let $I(\mathbf{x})$ represent the current frame $I^t(\mathbf{x})$, then for a set of surface points $\mathbf{S} = \{\mathbf{X}\}$ under the same translation $\mathbf{V}$ and satisfying the brightness constancy assumptions we have

$$\sum_{\mathbf{X} \in \mathbf{S}} \left[I\left(\hat{\mathbf{M}}\left(\mathbf{X} + \mathbf{V}\right)\right) - T\left(\hat{\mathbf{M}}(\mathbf{X})\right)\right]^2 = 0 \tag{16}$$

If set $\mathbf{s} = \{\mathbf{x}\}$ contains the projections of $\mathbf{S}$ onto the image and $\mathbf{W}(\mathbf{x}; \mathbf{V})$ is a warp mapping each pixel $\mathbf{x}$ in image T to the corresponding pixel $\mathbf{x}'$ in image I, (16) becomes

$$\sum_{\mathbf{x} \in \mathbf{s}} \left[I\left(\mathbf{W}(\mathbf{x}; \mathbf{V})\right) - T\left(\mathbf{x}\right) \right]^2 = 0 \tag{17}$$

The warp function that satisfies (17) can be defined as

$$\mathbf{W}(\mathbf{x}; \mathbf{V}) = \mathbf{x} + \Delta(\mathbf{x}; \mathbf{V}) = \mathbf{x}' \tag{18}$$

with the delta function modeling the image flow induced by translation $\mathbf{V}$. Using the rigid translation flow model defined in (8) we have

$$\Delta(\mathbf{x}; \mathbf{V}) = \frac{1}{Z(\mathbf{x})} \begin{pmatrix} f_x & 0 & c_x - x \\ 0 & f_y & c_y - y \end{pmatrix} \begin{pmatrix} V_X \\ V_Y \\ V_Z \end{pmatrix} \tag{19}$$

where $Z(\mathbf{x})$ is the depth of pixel $\mathbf{x}$ (Z component of $\mathbf{X}$).

3.2. Tracking in the intensity image

We propose the warp function given by equation (18), where the image motion in a region is supposed to be generated by a translation motion of a local rigid object. In this case, the ideal parameter vector $\mathbf{P}$ corresponds to the translation velocity vector $\mathbf{V}$. Although the appropriate motion model is chosen, the brightness constancy assumption could be violated due to factors as noise sensor, illumination changes, specular surfaces, etc. To reduce the influence of outliers we use the function $\psi(x^2) = \sqrt{x^2 + \epsilon^2}$ [8], a differentiable variant of the L_1-norm. Therefore we formulate the tracking problem as finding the velocity vector $\mathbf{V}$ that minimizes

$$\sum_{\mathbf{x}} \psi\left(E_I(\mathbf{x}; \mathbf{V})^2 \right), \tag{20}$$

where $E_I\left(\mathbf{x}; \mathbf{V}\right) = I\left(\mathbf{W}\left(\mathbf{x}; \mathbf{V}\right)\right) - T\left(\mathbf{x}\right)$ stands for the intensity difference. Minimization of expression (20) is a nonlinear optimization problem, which can be solved by IRLS [12]. Using the first order Taylor expansion on the intensity function I, equation (20) becomes

$$\sum_{\mathbf{x}} \psi\left(\left(E_I\left(\mathbf{x}; \mathbf{V}\right) + \nabla_I \frac{\partial \mathbf{W}}{\partial \mathbf{V}} \Delta \mathbf{V} \right)^2 \right), \tag{21}$$

where $\nabla_I = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right)$ is the row vector gradient evaluated at $\mathbf{W}\left(\mathbf{x}; \mathbf{V}\right)$. The term $\frac{\partial \mathbf{W}}{\partial \mathbf{V}}$ is the Jacobian $\mathbf{J}$ of the warp, and if $\mathbf{W}\left(\mathbf{x}; \mathbf{V}\right) = \left(W_x, W_y\right)^T$ then

$$\mathbf{J} = \begin{pmatrix} \frac{\partial W_x}{\partial V_X} & \frac{\partial W_x}{\partial V_Y} & \frac{\partial W_x}{\partial V_Z} \\ \frac{\partial W_y}{\partial V_X} & \frac{\partial W_y}{\partial V_Y} & \frac{\partial W_y}{\partial V_Z} \end{pmatrix}. \tag{22}$$

Finding the minimum of expression (21) requires an iterative solution. Taking the partial derivative with respect to $\Delta \mathbf{V}$ gives

$$2 \sum_{\mathbf{x}} \psi_I'\left(\mathbf{x}; \mathbf{V}; \Delta \mathbf{V}\right) \left(\nabla_I \mathbf{J}\right)^T \left(E_I\left(\mathbf{x}; \mathbf{V}\right) + \left(\nabla_I \mathbf{J}\right) \Delta \mathbf{V} \right) \tag{23}$$

where $\psi_I'\left(\mathbf{x}; \mathbf{V}; \Delta \mathbf{V}\right) = \psi'\left(E_I\left(\mathbf{x}; \mathbf{V}\right) + \nabla_I \mathbf{J} \Delta \mathbf{V}\right)$. At the minimum of (21), (23) is zero, so that $\Delta \mathbf{V}$ can be computed as follows

1. Initialize $\Delta \mathbf{V} = 0$
2. Compute $\psi_I'\left(\mathbf{x}; \mathbf{V}; \Delta \mathbf{V}\right)$ and then the velocity update

$$\Delta \mathbf{V} = -H^{-1} \sum_{\mathbf{x}} \psi_I'\left(\mathbf{x}; \mathbf{V}; \Delta \mathbf{V}\right) \left(\nabla_I \mathbf{J}\right)^T E_I\left(\mathbf{x}; \mathbf{V}\right) \tag{24}$$

where H, the Gauss-Newton approximation of the Hessian, is given by

$$H = \sum_{\{\mathbf{x}\}} \psi_I'\left(\mathbf{x}; \mathbf{V}; \Delta \mathbf{V}\right) \left(\nabla_I \mathbf{J}\right)^T \left(\nabla_I \mathbf{J}\right) \tag{25}$$

4. Update step $\mathbf{V} \leftarrow \mathbf{V} + \Delta \mathbf{V}$
5. if $\Delta \mathbf{V} < \epsilon$ stop, otherwise goto 1.

Over each iteration the Hessian matrix can be expressed as

$$H = \sum_{\{\mathbf{x}\}} \frac{\psi_I'\left(\mathbf{x}; \mathbf{V}; \Delta \mathbf{V}\right)}{Z(\mathbf{x})^2} \begin{pmatrix} I_x^2 & I_x I_y & I_x I_\Sigma \\ I_x I_y & I_y^2 & I_y I_\Sigma \\ I_x I_\Sigma & I_y I_\Sigma & I_\Sigma^2 \end{pmatrix} \tag{26}$$

with $I_\Sigma = -\left(x I_x + y I_y\right)$.

4. Tracking in intensity and depth

Assuming a local rigidity we formulated in the previous section a tracking algorithm in a Lucas-Kanade framework in which the parameter vector corresponds to 3D motion field. The warp function we proposed uses the depth value of each image pixel to compute the image motion induced by the estimated 3D motion field. However, in this approach the motion field is constrained only using the brightness information and depth information is not fully exploited.

We propose to incorporate a constraint based on the depth velocity to overcome some of the knowns problems of the brightness constancy assumption. The new constraint is computed from the squared difference between the depth estimation of each template pixel using V_Z and the current depth measure given by the sensor. As in the earlier approach, a locally rigid motion is assumed.

4.1. Depth velocity constraint

Let $Z(\mathbf{x})$ be the depth of the image pixel $\mathbf{x}$ at time t. Knowing that $\mathbf{X}' = \mathbf{X} + \mathbf{V}$, we obtain that $Z' = Z + V_Z$ and the depth image must satisfy

$$Z^{t-1}(\hat{\mathbf{M}}(\mathbf{X})) + V_Z = Z^t(\hat{\mathbf{M}}(\mathbf{X}')). \tag{27}$$

If $T_Z(\mathbf{x})$ stands for the previous depth frame $Z^{t-1}(\mathbf{x})$ (depth template) and $Z(\mathbf{x})$ for the current one $Z^t(\mathbf{x})$, then for a set of points $\mathbf{S} = \{\mathbf{X}\}$ on the surface we have

$$\sum_{\mathbf{X}\in\mathbf{S}} \left[Z\left(\hat{\mathbf{M}}\left(\mathbf{X}+\mathbf{V}\right)\right) - \left(T_Z\left(\hat{\mathbf{M}}(\mathbf{X})\right) + V_Z\right)\right]^2 = 0 \tag{28}$$

Using the warp function (18) equation (28) becomes

$$\sum_{\mathbf{x}\in\mathbf{s}} \left[Z\left(W(\mathbf{x};\mathbf{V})\right) - \left(T_Z\left(\mathbf{x}\right) + V_Z\right)\right]^2 = 0, \tag{29}$$

where $\mathbf{s}$ is the projection of set $\mathbf{S}$ onto the image.

4.2. Tracking algorithm

Using the brightness and depth velocity constraints with the proposed warp (18), the tracking problem is reformulated as finding the 3D motion field vector $\mathbf{V}$ that minimizes

$$\sum_{\{\mathbf{x}\}} \left[I\left(\mathbf{W}(\mathbf{x};\mathbf{V})\right) - T\left(\mathbf{x}\right)\right]^2 + \lambda\left[Z\left(\mathbf{W}(\mathbf{x};\mathbf{V})\right) - \left(T_Z\left(\mathbf{x}\right) + D^T\mathbf{V}\right)\right]^2 \tag{30}$$

where $\lambda = \sigma_I^2/\sigma_Z^2$, with σ_I^2 and σ_Z^2 the noise variance in intensity and depth sensors, respectively. The vector $D^T = (0,0,1)$ isolates the Z component of the motion field.

We write the brightness difference as $E_I\left(\mathbf{x};\mathbf{V}\right) = \left[I\left(\mathbf{W}\left(\mathbf{x};\mathbf{V}\right)\right) - T\left(\mathbf{x}\right)\right]$ and the depth velocity difference as $E_Z\left(\mathbf{x};\mathbf{V}\right) = \left[Z\left(\mathbf{W}\left(\mathbf{x};\mathbf{V}\right)\right) - \left(T_Z\left(\mathbf{x}\right) + D^T\mathbf{V}\right)\right]$ in the following derivations. Assuming an initial estimation of $\mathbf{V}$, each optimization step searches for $\mathbf{\Delta V}$ which minimizes

$$\sum_{\mathbf{x}} E_I(\mathbf{x};\mathbf{V}+\Delta\mathbf{V})^2 + \lambda E_Z(\mathbf{x};\mathbf{V}+\Delta\mathbf{V})^2$$

Equation (31) is linearized using the first order Taylor expansions on I and Z, to give

$$\sum_{\mathbf{x}} \left[E_I\left(\mathbf{x};\mathbf{V}\right) + (\nabla_I\mathbf{J})\Delta\mathbf{V}\right]^2 + \lambda\left[E_Z\left(\mathbf{x};\mathbf{V}\right) + \left(\nabla_Z\mathbf{J} - D^T\right)\Delta\mathbf{V}\right]^2 \tag{31}$$

where $\nabla_I = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right)$ and $\nabla_Z = \left(\frac{\partial Z}{\partial x}, \frac{\partial Z}{\partial y}\right)$, both evaluated at $\mathbf{W}\left(\mathbf{x};\mathbf{V}\right)$. Now, taking the partial derivative of (31) with respect $\Delta\mathbf{V}$, setting this expression to zero and solving for $\Delta\mathbf{V}$, the update rule can be stated as

$$\Delta\mathbf{V} = -H^{-1}\sum_{\mathbf{x}} \left(\nabla_I\mathbf{J}\right)^T E_I\left(\mathbf{x};\mathbf{V}\right) + \lambda\left(\nabla_Z\mathbf{J} - D^T\right)^T E_Z\left(\mathbf{x};\mathbf{V}\right) \tag{32}$$

where matrix H is given by

$$H = \sum_{\{\mathbf{x}\}} \left(\nabla_I\mathbf{J}\right)^T\left(\nabla_I\mathbf{J}\right) + \lambda\left(\nabla_Z\mathbf{J} - D^T\right)^T\left(\nabla_Z\mathbf{J} - D^T\right) \tag{33}$$

The Hessian matrix can be expressed as

$$H = \sum_{\{\mathbf{x}\}} \frac{1}{Z(\mathbf{x})^2} \begin{pmatrix} I_x^2 + \lambda Z_x^2 & I_xI_y + \lambda Z_xZ_y & I_xI_\Sigma + \lambda Z_x\left(Z_\Sigma - 1\right) \\ I_xI_y + \lambda Z_xZ_y & I_y^2 + \lambda Z_y^2 & I_yI_\Sigma + \lambda Z_y\left(Z_\Sigma - 1\right) \\ I_xI_\Sigma + \lambda Z_x\left(Z_\Sigma - 1\right) & I_yI_\Sigma + \lambda Z_y\left(Z_\Sigma - 1\right) & I_\Sigma^2 + \lambda\left(Z_\Sigma - 1\right)^2 \end{pmatrix} \tag{34}$$

with $I_\Sigma = -\left(xI_x + yI_y\right)$ and $Z_\Sigma = -\left(xZ_x + yZ_y\right)$.

In order to get a close solution we have not utilized a robust norm in formulation (30) but in our implementation we apply the same variant of the L_1-norm that in (20) to each one of the terms. Setting parameter λ to be zero the intensity and depth objective function (30) becomes the only intensity tracking formulation of Section 3.2.

A reliable solution of (30) requires that matrix H given by (34) must be both above of the noise of the images (intensity and depth) and well conditioned. We perform some experiments in the next section.

5. Experimental results

5.1. Middlebury datasets

In order to evaluate the performance of our method we have conducted a set of experiments using an existing scene flow benchmark. The proposed algorithm that performs tracking in intensity and depth images is called RT and its version with $\lambda = 0$ is detonated as RT0.

As in [16, 4, 13] Middlebury stereo datasets [28], *Cones* and *Teddy*, were used for the experiments. Using images of one of these datasets es equivalent to a fixed camera observing a moving object along X axis. We took image 2 of each dataset as the first frame and image 6 as the second frame, both in quater-size. Parameters of the camera setup were estimated and the depth data was generated from the disparity map of each image. The ground truth for the optical flow is given by disparity map from frame 1.

Since errors in the image plane do not necessarily correspond with errors in the scene [4], we evaluate both 2D and 3D errors in the experiments. To measure errors in 2D we compute the *root mean squared* ($\mathrm{RMS_{OF}}$) and the *average angle error* ($\mathrm{AEE_{OF}}$) of the optical flow. Moreover as in [28], we use the statistic RX of $\mathrm{RMS_{OF}}$ to observe the percentage of pixels that have an error measure above X, for $X = 1.0$ and 5.0. In order to asses errors in the scene flow we compute the *normalized root mean squared* ($\mathrm{NRMS_V}$) of the 3D flow vector, similar to [33], but we normalize the rms error by the (constant) magnitude of the real scene flow ground truth given by the baseline, allowing calculation of the error as percentage of the ground truth. As in the 2D case, we compute the statistic RX of $\mathrm{NRMS_V}$, now for the percentages $X = 5\%$ and 20%.

		RT	RT0	OF_R	[29]	[7]
Teddy	RMS_{OF}	2.71	2.92	4.18	4.62	7.21
	R1.0	9.54	12.9	23.7	28.9	40.2
	R5.0	2.5	3.7	13.25	19.1	21.9
	AEE_{OF}	1.17	1.38	1.18	1.48	1.67
Cones	RMS_{OF}	2.32	2.31	5.20	4.02	4.70
	R1.0	16.3	16.7	33.1	40.3	39.9
	R5.0	2.15	4.29	17.2	14.6	17.6
	AEE_{OF}	1.14	1.28	1.84	1.42	1.24

Table 1. Errors in the optical flow. The proposed algorithm and its variant outperform other trackers in the magnitude estimation of the image flow. Statistic R5.0 shows that the number of outliers is reduced when the proposed motion model is used. The results in the estimation of direction of all the algorithms are similar.

		RT	RT0	OF_R	[29]	[7]
Teddy	$NRMS_V$	11.4	60.6	41.2	41.2	74.1
	R5%	18.6	29.1	37.2	38.3	40.2
	R20%	7.06	12.6	20.4	23.2	22.2
Cones	$NRMS_V$	10.8	51.2	95.6	93.7	89.9
	R5%	15.6	28.5	38.6	39.8	35.9
	R20%	2.89	6.77	14.5	14.9	16.0

Table 2. Errors in the scene flow. The RT tracker achieves the most accurate results. Small pixel deviations in the image domain have a strong influence in error computed in the scene. Statistics R5% and R20% allows to perform a more detailed comparison since $NRMS_V$ is highly sensitive to outliers.

We compare our method with two version of the Lukas-Kanade tracker (KLT) as well as with other scene flow algorithms. The scene flow is computed from optical flow methods by tracking in the intensity image and inferring the 3D motion using the depth data. As optical flow algorithms we use the C pyramidal implementation of KLT by Bouguet [7] and our own pyramidal implementation with a robust norm (OF_R). We also implemented a pyramidal version of the orthographic intensity and depth tracker presented in [29]. All algorithms were configured to use a fixed window size of 11×11, in a Gaussian pyramidal decomposition of 5 levels. In the experiments we perform the computation over a regular grid covering the 85% of the image area, with a distance of 5 pixels between window centers. Results over non-occluded regions are presented in Table 1 for the optical flow and in Table 2 for the scene flow.

In order to observe the performance of each algorithm in particular regions, the scene flow is computed on textured and untextured regions, as well as on depth discontinuities. Points for texture analysis are chosen by observing the minimum eigenvalue (λ_{min}) of the Hessian matrix of I in a region. Higher values of λ_{min} are used to select textured (Tex) regions while lower values of λ_{min} define untextured (Utex) regions. Depth discontinuities (DD) regions are chosen by finding higher values in the depth image gradient. For each of these regions the averaged error on *Teddy* and *Cones* images is presented in Table 3. For all regions our method outperforms classical KLT and results shows that better regions to track correspond to textured regions but avoiding strong depth discontinuities.

	RT			OF_R		
	Tex	*Utex*	*DD*	*Tex*	*Utex*	*DD*
RMS_{OF}	4.99	3.11	6.91	5.75	7.20	7.05
R1.0	16.5	39.8	32.5	38.9	58.8	68.7
$NRMS_V$	23.2	10.9	26.5	96.7	202	188
R5%	12.1	28.5	25.1	32.0	51.5	69.6

Table 3. Errors by regions. The accuracy is reduced on DD regions since the number of outliers increases due to occluded regions and unmeasured depth points. Although scene flow error R5% is strongly affected on DD, the use of depth data reduce the number of outliers in the image domain w.r.t. OF_R. In average errors on untextured regions are low but the percentage of points with inaccurate optical flow or scene flow is the highest, as can be seen in R1.0 R5%. Finally, on textured regions the error $NRMS_V$ is increased due to the presence of depth discontinuities, however, by observing R1.0 R5% precision on this regions is high both in optical flow and scene flow.

There is not a standard metric to asses scene flow algorithms, so that, each author use a error metric with or without normalization. In order to compare our method with other scene flows approaches we choose the RMS_{OF} because it is available or can be derivate from all works. Table 4 presents a summary of the errors of methods proposed in [16], [4] and [13] in comparison with the proposed algorithm in this work. We have denormalized the result of the magnitude presented in [13] by the range (40.25 for *Teddy* and 49.5 for Cones) of the optical flow magnitude to obtain RMS_{OF}. Also using depth and intensity data, the proposed method achieved better than [13] both in magnitude and direction. On the other hand, comparison with stereo based techniques is not straightforward since they use closer images of the Middlebury datasets. In [16] and [4], images 2 and 6 are taken as the stereo pair at time $t-1$ and images 4 and 8 correspond to the stereo pair at time t. Accordingly, the optical flow ground truth is half of the disparity and there are fewer occluded regions. However, normalizing by the range of the optical flow magnitude of the corresponding ground truth our methods outperforms [4].

5.2. Kinect images

Some experiments were performed from image sequences of a Microsoft Kinect sensor where intensity and depth images were accessed using libfreenect library in C++. We used the Kinect calibration toolbox [14] to get intrinsic and extrinsic parameters of the depth and RGB cam-

		RT	[16]	[4]	[13]
Teddy	RMS_{OF}	4.3	1.25	2.85	4.42
	AEE_{OF}	2.73	0.51	1.01	5.04
Cones	RMS_{OF}	3.5	1.11	3.07	4.36
	AEE_{OF}	2.22	0.69	0.39	5.02

Table 4. Comparison of scene flow algorithms in 2D.

era pair and to perform the image alignment. A cropped and aligned pair of rgb and depth images is shown in Figures 2(a) and 2(c), where depth data is encoded by gray levels, where lighter pixels correspond to closer points from the sensor. Unmeasured regions are encoded in black and they correspond to occluded or out of range regions to the depth camera, or to points belonging to reflective surfaces.

In Figure 2 we show the results of the motion field estimation between a pair of images where a person is performing a motion with both hands. In the sequence the left (in the image) hand is moving away from the sensor while the right one is approaching it. We computed scene flow over a regular grid of points equally spaced in 5 pixels with with a window size of 11×11 and a pyramidal decomposition of 2 levels. The resulting optical flow and scene flow are presented in Figures 2(d) and 2(e). Points on the hand surface present almost the same scene flow while the optical flow magnitude is function of the depth. Using only the optical flow it is not possible to distinguish the different types of movement that each hand performs, this ambiguity can be resolved by means of the scene flow, e.g. using its Z component as illustrated in Figure 2(f).

In the pyramidal decomposition points with unmeasured depth are propagated in the different levels increasing the effective area of the regions where scene flow can not be computed. Figure 1 shows the projection of the scene flow on a image region, where not computed values correspond to regions without enough depth information.

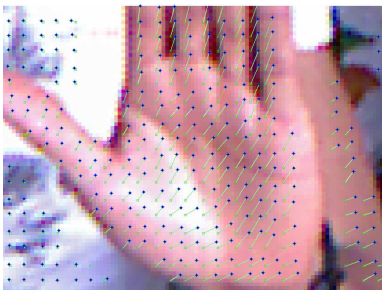

Figure 1. Scene flow projection onto the image

6. Conclusions

We have proposed a novel approach to compute scene flow by tracking surface patches in a Lucas-Kanade framework. Using a rigid translation motion model we state scene flow computation as a 2D tracking problem in intensity and depth data. Although we have chosen a specific flow model on the image, using some assumptions, the presented formulation for the scene flow problem is valid with other motion models wherever motion field $\mathbf{V} = (V_Y, V_Y, V_X)$ can be computed from the vector parameter $\mathbf{P}$. As we solve simultaneously for the 2D tracking and for the scene flow this method could be used directly in action recognition task to generate accurate trajectories or to create scene flow based descriptors.

Through some experiments we demonstrated the validity of our method and we showed that it outperforms other Lucas-Kanade tracker for optical flow and scene flow computation. We also perform comparison with other scene flow techniques with very encouraging results.

Since our method require the inversion of a 3×3 matrix which is function of the gradients of intensity and depth images, we expect to obtain a criterion for selecting good region to compute scene flow.

References

[1] G. Adiv. Determining three-dimensional motion and structure from optical flow generated by several moving objects. *PAMI*, 1985.

[2] S. Baker and I. Matthews. Lucas-Kanade 20 years on: A unifying framework. *IJCV*, 2004.

[3] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. Black, and R. Szeliski. A database and evaluation methodology for optical flow. In *ICCV*, 2007.

[4] T. Basha, Y. Moses, and N. Kiryati. Multi-view scene flow estimation: A view centered variational approach. In *CVPR*, 2010.

[5] J. Bergen, P. Anandan, K. Hanna, and R. Hingorani. Hierarchichal model-based motion estimation. In *ECCV*, 1992.

[6] A. Bobick and J. Davis. The representation and recognition of action using temporal templates. *PAMI*, 2001.

[7] J.-Y. Bouguet. Pyramidal implementation of the lucas-Kanade feature tracker. *Open CV Documentation, Intel Corporation, Microporcessor Research Labs*, 1999.

[8] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *ECCV*, 2004.

[9] F. Devernay, D. Mateus, and M. Guilbert. Multi-camera scene flow by tracking 3D points and surfels. In *CVPR*, 2006.

[10] J. Fang and T. Huang. Solving three-dimensional small-rotation motion equations. In *CVPR*, 1983.

[11] Y. Fukurawa and J. Ponce. Dense 3D motion capture from synchronized video streams. In *CVPR*, 2008.

[12] P. Green. Iteratively reweighted least squares for maximum likelihood estimation, and some robust and resistant alternatives. *Journal of the Royal Statistical Society*, 1984.

[13] S. Hadfield and R. Bowden. Kinecting the dots: Particle based scene flow form depth sensors. In *ICCV*, 2011.

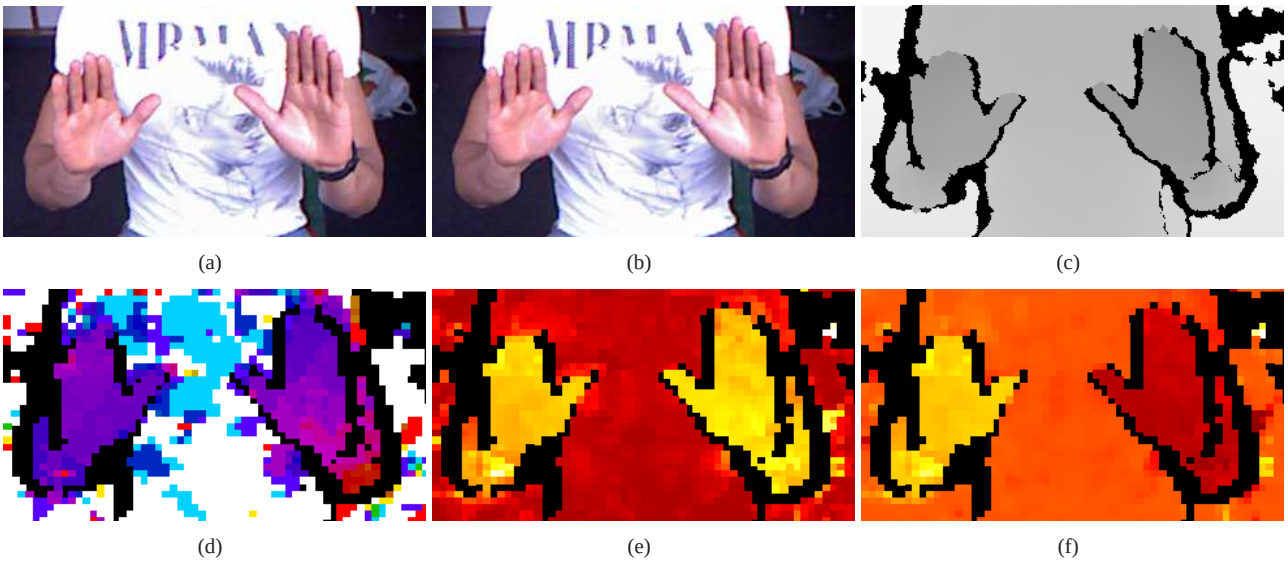

Figure 2. Results from a pair of Kinect images. (a-b) frames at time $t-1$ and t, (c) depth image at time $t-1$, (d) resulting optical flow, (e) magnitude of the resulting scene flow and its (f) z component. Motion estimation in regions with unmeasured depth is not possible and it is presented in black. Optical flow is presented using the color-coded by [3]. The scene flow and its Z component are encoded in a red-yellow-white color map. In the magnitude map (e) static regions of the body are in red and faster regions in light yellow. For the Z component the 0 velocity is orange, red colors represent negative velocities (approaching pixels) and yellow colors are positives velocities.

[14] D. Herrera, J. Kannala, and J. Heikkila. Accurate and practical calibration of a depth and color camera pair. In *CAIP*, 2011.

[15] B. Horn and E. Weldon. Direct methods for recovering motion. *IJCV*, 1988.

[16] F. Huguet and F. Devernay. A variational method for scene flow estimation fro stereo sequences. In *ICCV*, 2007.

[17] I. Laptev and T. Lindeberg. Space-time interest points. In *ECCV*, 2003.

[18] A. Letouzey, B. Petit, and E. Boyer. Scene flow from depth and color images. In *BMVC*, 2011.

[19] W. Li, Z. Zhang, and Z. Liu. Action recognition based on a bag of 3D points. In *CVPR Workshops*, 2010.

[20] H. Longuet-Higgins and K. Prazdny. The interpretation of a moving retinal image. *Royal Society London*, 1980.

[21] T. Lukins and R. Fisher. Colour constrained 4d flow. In *BMVC*, 2005.

[22] P. Matikainen, M. Hebert, and R. Sukthankar. Trajectons: Action recognition through the motion analysis of tracked features. In *ICCV Workshops*, 2009.

[23] R. Messing, C. Pal, and H. Kautz. Activity recognition using the velocity histories of tracked keypoints. In *ICCV*, 2009.

[24] J. Neumann and Y. Aloimonos. Spatio-temporal stereo using multi-resolution division surfaces. *IJCV*, 2002.

[25] B. Ni, G. Wang, and P. Moulin. Rgbd-hudaact: A color-depth video database for human daily activity. In *CVPR Workshops*, 2011.

[26] J. Niebles, C. Chen, and L. Fei-Fei. Modeling temporal structure of decomposable motion segments for activity classification. In *ECCV*, 2010.

[27] M. Raptis and S. Soatto. Tracklet descriptors for action modeling and video analysis. In *ECCV*, 2010.

[28] D. Scharstein and R. Szeliski. High-accuracy stereo depth maps using structured light. In *CVPR*, 2003.

[29] H. Spies, B. Jahne, and J. Barron. Dense range flow from depth and intensity data. In *ICPR*, 2000.

[30] L. Valgaerts, A. Bruhn, H. Zimmer, J. Weickert, C. Stoll, and S. Theobalt. Joint estimation of motion, structure and geometry from stereo sequences. In *ECCV*, 2010.

[31] S. Vedula, S. Baker, P. Rander, and R. Collins. Three-dimensional scene flow. In *ICCV*, 1999.

[32] S. Vedula, S. Baker, P. Render, R. Collins, and T. Kanade. Three-dimensional scene flow. *PAMI*, 2005.

[33] C. Vogel, K. Schindler, and S. Roth. 3D scene flow estimation with a rigid motion prior. In *ICCV*, 2011.

[34] A. Wedel, C. Rabe, T. Vaudrey, T. Brox, U. FRanke, and D. Cremers. Efficient dense scene flow from sparse of dense stereo data. In *ECCV*, 2008.