Machine Vision and Applications (2012) 23:887–902
DOI 10.1007/s00138-011-0358-4

ORIGINAL PAPER

Camera–projector matching using unstructured video

Marc-Antoine Drouin · Pierre-Marc Jodoin · Julien Prémont

Received: 23 September 2010 / Revised: 6 May 2011 / Accepted: 5 July 2011 / Published online: 31 July 2011

Abstract This paper presents a novel approach for matching 2-D points between a video projector and a digital camera. Our method is motivated by camera–projector applications for which the projected image needs to be warped to prevent geometric distortion. Since the warping process often needs geometric information on the 3-D scene obtained from a triangulation, we propose a technique for matching points in the projector to points in the camera based on arbitrary video sequences. The novelty of our method lies in the fact that it does not require the use of pre-designed structured light patterns as is usually the case. The backbone of our application lies in a function that matches activity patterns instead of colors. This makes our method robust to pose, severe photometric and geometric distortions. It also does not require calibration of the color response curve of the camera–projector system. We present quantitative and qualitative results with synthetic and real-life examples, and compare the proposed method with the scale invariant feature transform (SIFT) method and with a state-of-the-art structured light technique. We show that our method performs almost as well as structured light methods and significantly outperforms SIFT when the contrast of the video captured by the camera is degraded.

M.-A. Drouin
NRC Institute for Information Technology, 1200 Montreal Road, Building M-50, Ottawa, ON K1A 0R6, Canada
e-mail: Marc-Antoine.Drouin@nrc-cnrc.gc.ca

P.-M. Jodoin (✉) · J. Prémont
Université de Sherbrooke, 2500, boulevard de l'Université, Sherbrooke, QC J1K 2R1, Canada
e-mail: Pierre-Marc.Jodoin@usherbrooke.ca

J. Prémont
e-mail: Julien.Premont@usherbrooke.ca

Keywords 3-D reconstruction · Structured light · Fitting · Video

1 Introduction

In the past decade, digital cameras and LCD/DLP video projectors have become almost ubiquitous as their price kept decreasing. This opens the door to numerous applications involving a projector and a camera, such as multimedia applications, shows, digital arts, and plays to name a few.

There are two fundamental issues that most projector applications have to deal with : photogrammetric and geometric distortion correction. This paper focuses on the second issue. As far as the observer is concerned, geometric distortions appear when the projector is located far from the observer and/or when the 3-D surface is badly oriented. As mentioned by Raskar et al. [27], whenever the projector and the 3-D surface cannot move, the only solution is to prewarp the projected image. Given the 3-D geometry of the scene, one can easily implement such warping function to prevent distortion from the observer's stand point [26,27,31] (if the warping is to be done for the camera's stand point, only the pixel mapping between the camera and the projector is needed [31]). Unfortunately, the geometry of the scene is often unknown *a priori* and thus needs to be estimated at runtime. One usual way of doing so is through the use of a camera and a two-step procedure. First, the camera and the projector are calibrated so that their intrinsic and extrinsic parameters are known [36]. Then, pre-designed patterns of light (the so-called *structured light patterns*) are projected on the surface so that pixels from the projector can be matched with those of the camera. Depending on the complexity of the scene, one can project simple dots of light (in the case of a planar surface, for instance) or more complex patterns [28]

(in the case of a compound surface). Once a sufficiently large number of matches has been found, a 3-D surface is recovered by triangulation. Given that both the scene and the projector stay fixed, the estimated 3-D surface is used to warp the projected image for any viewpoint in the room. In other words, our goal is to prewarp the projected video so that it does not look distorted from a given point of view. Typical applications that could benefit from our method are plays and music shows in which the scene on which the video is projected changes in time. It is the case when the walls pivot on an axis or the floor goes up and down. Our method can also be used in a home application to automatically readjust projected videos.

One obvious problem arises when the camera/projector system and/or the scene is moved after the calibration is over. One typical example is in plays involving artistic staging. In this case, the use of structured light patterns to recover the 3-D geometry becomes inadequate as the system would need to stop projecting the video to readjust. Such an application thus requires the matching to be done directly from the projected to the captured video. Unfortunately, we empirically noticed that pure color-based matching strategies between two such images are doomed to fail since images captured by the camera are heavily degraded by non-linear color distortion (see Fig. 1b vs. c). This is especially true when the white balance and exposure time of the camera automatically readjust and/or when the projection surface is textured. In fact, it has already been shown that SIFT [20], although one of the most robust matching method, fails in such conditions [7].

In this paper, we propose to find camera–projector matches based on *unstructured light patterns*, i.e., based on the projected video itself. In this way, each time the system needs to recover the 3-D scene, our approach analyses *activity patterns* recorded in both videos and use it to find matches. These activity patterns are obtained following a motion detection method applied simultaneously on the video emitted by the projector and the video captured by the camera. They are then bundled with grayscale quanta and embedded into a cost function used to find matches. Once matches have been found, the 3-D structure of the scene is recovered and the projected video warped. In this paper, we focus on piecewise planar surfaces and quadrics such as spheres and cylinders. Interestingly, our system needs between 15 and 30 frames (i.e., at most 1 s of video) to efficiently recover the 3-D scene, This allows an artistic director to perform a quick readjustment of the system unbeknownst to the audience. We tested our method on different videos including music clips, animated movies, and home-made videos.

Fig. 1 **a** Our setup contains an LCD/DLP projector, a camera, and a piecewise planar 3-D surface. The projected (**b**) and captured videos (**c**) are time-synchronized

2 Contribution

The main contribution of this paper lies in the camera–projector matching procedure which combines the following key elements:

1. It is non-intrusive. The matching is performed without alterations to the scene nor the projected video and does not require the use of structured light patterns or featured tracking;
2. Since our method is not color-based, our camera–projector matching procedure does not need a colorimetric calibration.
3. Our method allows for a quick recalibration (after a change in the relative position of the scene and the camera–projector) without interrupting the projection.

3 Previous work

Matching pixels in two images (here camera–projector matches) is the first step of most applications involving a triangulation procedure. There has been a significant effort to develop simple and efficient matching strategies that we summarize in four categories.

3.1 Structured light

Certainly, one of the most implemented strategy, structured light methods use pre-designed patterns of light to encode pixels' position. All kinds of patterns have been proposed including color, binary and grayscale patterns, patterns with spatial coding, others with time-multiplexing coding, some being dense, others being sparse, etc. [28]. As far as our system is concerned, structured light is hardly a solution since the pose between the system and the scene may vary in time. Being incapable of readjusting in real time, the system would need to stop the user-selected video to recalibrate. This, of course, is unacceptable for obvious marketing reasons.

Some structured light methods use a binary code and its inverse following inter-frame differences [28]. Up to a certain point, our method can be viewed as an extension of such approaches. This being said, these methods use only predesigned patterns of light, while we use purely unstructured video. Moreover, the inter-frame difference in our case is used to extract motion features, not the inverse of a pattern of light.

Other methods use dynamic programming to reduce the ambiguity of their structured light patterns [34,35]. In our case, dynamic programming is used to enforce an ordering constraint.

Some other techniques embed imperceptible patterns of light into the projected images [5,38]. One way of doing so is by reducing the dynamic interval of the DLP projector. However, this solution is only conceivable for high-end (and very costly) projectors with a large dynamic interval and for which we know how input color are transformed by the mirror-flipping commands.

Our solution does not suffer from these limitations as it uses *unstructured light* based on the ongoing video.

3.2 Feature-based matching

A second approach consists in matching feature points extracted from the projected and the captured images [18,32,37]. One such approach that has drawn a lot of attention lately is the scale invariant feature transform (SIFT) [20]. SIFT is one of the very few methods which provide a solution for both extracting feature points and finding point-to-point matches. The main advantage with SIFT lies in its robustness to geometric transformations and non-linear illumination distortion. This being said, we empirically observed that the number of SIFT matches decreases rapidly in the presence of severe perspective transformations and/or illumination distortions. This is a major limitation as far as our application is concerned. Empirical results will be shown in Sect. 6.

3.3 Stereovision

Stereovision methods are typically used on images taken by two cameras mounted side-by-side [29]. Unfortunately, it has long been documented that simple (but realtime) greedy optimization strategies such as *winner-take-all* underperform in textureless areas and that only global (and slow) optimizers such as graph cut or belief propagation provide decent matches [29]. This makes the stereovision strategy ill-suited for our camera–projector setup which calls for fast solutions. Also, since there is a significant color distortion between the projected and the captured videos, stereovision methods based on a color–constancy hypothesis are doomed to fail [29]. Note that cost functions based on mutual information have been designed to deal with color inconsistency problems [17]. Nevertheless, these cost functions are computationally expensive and not adapted to systems such as ours.

Let us, however, mention that stereovision could be a sound solution for a two-camera/one-projector system [19,25]. This would be true for highly textured videos allowing simple greedy methods to work. Such methods are known as spatio-temporal stereovision [6,35].

3.4 Cooperative scenes

Markers made of vivid colors and easy-to-locate designs can be physically stitched to the scene. One example of such makers is ArTags [11] which show great robustness. However, we empirically observed that visual tags are not easy to detect when color patterns are projected on them. Also, for obvious aesthetic reasons, some applications involving live shows or home products forbid the use of markers. Let us mention that infrared LEDs with infrared cameras can also be utilized [33]. However, such a method is costly, requires extra hardware, and is not robust in areas where the ambient temperature fluctuates in time.

4 Overview of our method

Let us now give an overview of our method to allow for a high-level understanding of its structure. Our method is based on five steps that will be described in more details in Sect. 5.

1. Calibrate the camera and the projector to estimate their intrinsic and extrinsic parameters.
2. Start projecting and capturing the video at each time t, detect motion and assign a grayscale quantum to each pixel in both videos.
3. Find camera/projector matches based on the grayscale quanta. To simplify this procedure, both image frames are rectified so their epipolar lines are horizontal.

4. Out of these matches, estimate the 3-D surface. Depending on the nature of the 3-D surface, fit planes or quadrics on the 3-D points.
5. Given the recovered 3-D geometry, warp the projected video.

5 Details of our method

5.1 Camera–projector calibration

As opposed to what the schematic representation of Fig. 1a suggests, the camera and the projector are screwed to a common plate so their relative position and orientation stay fixed during the entire projection. The system thus needs to be calibrated only once at the beginning of the process. To do so, we use Zhang's calibration method [36], which calibrates the projector like a camera. To avoid user intervention, structured light patterns are first projected on a flat checkerboard to get a one-to-one correspondence between the pixels of the camera and those of the projector. The checkerboard corners are then detected to calibrate the camera and recover the 3-D plane. The projector is then calibrated using the correspondences and the known 3-D position of the checkerboard's corners. At the end of this stage, the intrinsic and extrinsic parameters of the camera and the projector are known and stored in memory (a similar calibration method can be found in [2]). These parameters are used to rectify the videos (Sect. 5.3), recover the 3-D geometry of the scene (Sect. 5.4), and warp the projected video (Sect. 5.5). Let us stress the fact that calibration has to be performed only once, the parameters being valid as long as the position and orientation of the camera and the projector do not change.

5.2 Motion detection and quantization

Once calibration is over, the system starts projecting the video on the 3-D scene. At the same time, the camera films the scene on which the video is projected (the camera is synchronized with the projector). As mentioned earlier, the goal is to find matches between the projector and the camera so the geometry of the scene can be recovered. This is done based on motion labels that we estimate with a simple background subtraction method.

Let f_t^p and f_t^c be the projected and captured video frames at time t, both containing RGB values. At each time t, a reference image r_t^p and r_t^c is subtracted (and then thresholded) from the input frames so that binary motion fields $\mathcal{X}_t^p$ and $\mathcal{X}_t^c$ are obtained:

$$r_{t+1}^i = \alpha f_t^i + (1-\alpha) r_t^i, \quad \text{and} \quad r_0^i = f_0^i$$
$$U_t^i(x,y) = ||\mathbf{f}_t^i(x,y) - \mathbf{r}_t^i(x,y)||$$
$$\mathcal{X}_t^i(x,y) = \begin{cases} 1 & \text{if } U_t^i(x,y) > \tau \\ 0 & \text{otherwise} \end{cases}$$

where $i = c$ or p, $\alpha \in [0,1]$, τ is a threshold, and $||.||$ stands for the Euclidean norm.

We noticed that noise, illumination changes and local brightness variations make the use of a global and fixed threshold τ error prone. To avoid errors, τ is computed adaptively and locally. In this perspective, images U_t^c and U_t^p are split into $p \times q$ blocks. Then, for each block, we compute the threshold following Otsu's segmentation technique [23]. The value of τ for each pixel (x,y) is then linearly interpolated from the threshold of the four nearest blocks. Quantitative and qualitative comparison between an adaptive and a fixed threshold will be provided in Sect. 6.

To further improve robustness, active pixels (those for which $\mathcal{X}_t^i(x,y) = 1$) are assigned a grayscale quantum ($Q_t^c(x,y)$ and $Q_t^p(x,y)$) following a quantization procedure. Quanta are estimated with the median-cut algorithm [16]. Median-cut divides the histogram of f_t^p and f_t^c (grayscale versions of videos frames) into bins of various sizes, each containing the same number of pixels. Once the algorithm has converged, each active pixel is assigned the bin index (read *quantum*) its grayscale falls into. In this way, $Q_t^i(x,y) \in \{1,2,\ldots,N\}$ for active pixels and $Q_t^i(x,y) = 0$ for inactive pixels.

5.3 Camera–projector matching

Now that every active pixel has been assigned a grayscale quantum, the goal is to find for every pixel (x_c, y_c) in the captured image its corresponding point (x_p, y_p) in the projected image. But before to do so, since the viewing axis of the camera and the projector are not parallel, the video frames are first rectified so their epipolar lines are horizontally aligned [12]. This is done with the intrinsic and extrinsic parameters estimated in Sect. 5.1.

5.3.1 Rectification

Our rectification procedure is based on that of Fusiello et al. [12] which we modified the final step. In their final step, Fusiello et al. average the intrinsic parameters of both views, which is not suited for a camera–projector system (camera's and projector's intrinsic parameters are too different).

In our case, we rectify the camera and projector frames with an homographic transformation based on the 3×3 matrices H_c and H_p. As explained in [12], such homography matrices are computed as follows:

$$H_c = K_{c'} R R_c^{-1} K_c^{-1}$$
$$H_p = K_{p'} R R_p^{-1} K_p^{-1}$$

where R_c and R_p are extrinsic rotation matrices, and matrix R is the camera and projector's pose (that we compute following Fusiello et al's method [12]). K_c and K_p are the camera's and projector's intrinsic matrices, while $K_{c'}$ and $K_{p'}$ are the 3×3 intrinsic matrices of the *rectified* camera and the *rectified* projector, i.e.

$$K_{c'} = \begin{pmatrix} f_x^c & 0 & c_x^c \\ 0 & f_y & c_y^c \\ 0 & 0 & 1 \end{pmatrix}, \; K_{p'} = \begin{pmatrix} f_x^p & 0 & c_x^p \\ 0 & f_y & c_y^p \\ 0 & 0 & 1 \end{pmatrix}.$$

Parameters c_x^c, c_y^c, c_x^p and c_y^p stand for the image center, while f_x^c and f_x^p are the X component of the focal length. Since the epipolar lines are horizontal in the rectified views, the Y component of the focal length f_y is the same for the camera and the projector. However, f_x^c and f_x^p are determined independently for each view. This allows to preserve the X-axis resolution of both the projector and camera rectified video frames and thereby ensure precision of the recovered 3-D points.

The values of f_x^c, f_x^p and f_y are obtained by minimizing the horizontal scale factor between the pixels in the original image and the ones in the rectified image. This is very similar to Gluckman and Nayar's method [13], but with a 1-D pixel stretching criteria instead of their 2-D criteria.

5.3.2 Cost function

At this point of the process, the camera and the projector image frames have been rectified. From now on, the matching procedure will look for the best match (x_p, y_c) (in the rectified projected image) given pixel (x_c, y_c) (in the rectified captured image). We denote "X_p" the correspondence map such that $X_p(x_c, y_c) = x_p$.

The best correspondence map X_p is the one which minimizes a given criteria whose definition is pivotal for our method. Given that each pixel (x_c, y_c) in the camera is assigned a set of quanta $\Gamma^c = \{Q^c_{t-W}(x_c, y_c), \ldots, Q^c_t(x_c, y_c)\}$ over a period of time W, the goal is to find the pixel in the projector with a similar set of quanta $\Gamma^p = \{Q^p_{t-W}(x_p, y_c), \ldots, Q^p_t(x_p, y_c)\}$. This leads to the following formulation

$$X_p = \arg\min_{\hat{X}_p} \sum_{x_c, y_c} C\left(\Gamma^c, \Gamma^p, x_c, y_c, \hat{X}_p\right) \quad (1)$$

where $C(.)$ is a cost function measuring how similar two sets of quanta Γ^c and Γ^p are. Since Γ^c and Γ^p are two vectors of equal length, $C(.)$ could be a simple Euclidean distance. Unfortunately, this function is error prone and needs to be replaced by a more robust function. Our cost function considers that two sets of quanta Γ^c and Γ^p are similar when their activity occurs at the same time and when their spatial and temporal gradients are similar. Mathematically, this leads to

$$C\left(\Gamma^c, \Gamma^p, x_c, y_c, \hat{X}_p\right) = \sum_{\tau=t-W}^{t} g(Q^c_s, Q^p_s, \mathcal{X}^c_s, \mathcal{X}^p_s)$$

where $Q^c_s = Q^c_\tau(x_c, y_c)$, $Q^p_s = Q^p_\tau(x_p, y_c)$, $\mathcal{X}^c_s = \mathcal{X}^c_\tau(x_c, y_c)$ and $\mathcal{X}^p_s = \mathcal{X}^p_\tau(x_p, y_c)$. As for $g(Q^c_s, Q^p_s, \mathcal{X}^c_s, \mathcal{X}^p_s)$, it is equal to 1 when $\mathcal{X}^c_s \neq \mathcal{X}^c_s$, to 0 when $\mathcal{X}^c_s = \mathcal{X}^c_s = 0$ and otherwise it is equal to $\sum_r \omega(Q^c_s - Q^c_r, Q^p_s - Q^p_r)$ where r is a first-order spatio-temporal neighbor of (x_c, y_c, τ) in Q^c and (x_p, y_c, τ) in Q^p and $\omega(a, b)$ returns 0 when $\text{sign}(a) \times \text{sign}(b) \geq 0$ and 2 otherwise.

5.3.3 Limiting the search interval

It is well known that a matching procedure does not need to consider every possible match (x_p, y_c) given a pixel (x_c, y_c) [29]. The reason for this is that some matches lead to 3-D points located behind the camera/projector system and others to 3-D points located too far away in front of the system. Limiting the search interval is thus an easy way to increase the matching accuracy while reducing the processing cost. Although some authors define the search interval in an *ad hoc* manner, one can explicitly compute it based on the minimum ($Z_{\min}$) and maximum ($Z_{\max}$) distances allowed. Since these distances are related to the minimum and maximum disparity (see Eq. (4) in Appendix A), we recover the search interval for a pixel (x_c, y_c) as follows:

$$\begin{bmatrix} x_p^{\min} \\ x_p^{\max} \end{bmatrix} = \begin{bmatrix} \frac{bf_x^p}{Z_{\min}} - \frac{(c_x^c f_x^p - c_x^p f_x^c - f_x^p x_c)}{f_x^c} \\ \frac{bf_x^p}{Z_{\max}} - \frac{(c_x^c f_x^p - c_x^p f_x^c - f_x^p x_c)}{f_x^c} \end{bmatrix}. \quad (2)$$

5.3.4 Optimization method

The goal of the optimization method is to solve Eq. (1). To do so, one could use a simple greedy optimizer such as *winner-take-all* (WTA) [29]. Unfortunately, we empirically observed that WTA generates a large number of outliers (read "bad matches") which propagates errors to the upcoming steps of the method. In order to reduce the number of outliers, we enforce an ordering constraint (OC). The OC states that if a point A lies to the left of a point B in one image, then A must also lie to the left of B in the other image. Although the OC can be violated in scenes containing thin objects and/or large occlusions [9], our system works only on 3-D scenes use as visualization surfaces. The OC is thus fulfilled in all scenes that we deal with. The OC can be enforced without a significant increase in CPU effort, thanks to a dynamic programming (DP) implementation [3,22] (see Appendix C for details on our DP implementation).

As can be seen in Fig. 2, DP significantly reduces the number of outliers as compared with WTA.

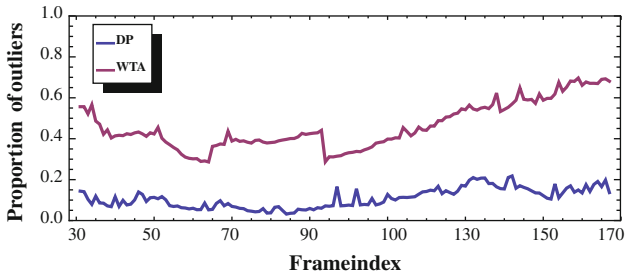

Fig. 2 Proportion of outliers returned by *winner-take-all* and dynamic programming at different time instants of a video

5.4 Fitting m planes on 3-D points

At this stage, a match has been found for every pixel at which activity had been recorded. This gives a sparse correspondence map X_p in which each camera pixel (x_c, y_c) is assigned a horizontal position (x_p, y_c) (see Fig. 3). Since projection surfaces are frequently piecewise planar, m planes can be fitted onto these points to recover the entire 3-D surface. Let us first explain how one plane can be fitted on X_p. We will then show how m planes can be fitted and how outliers are handled.

5.4.1 Fitting one plane

Let $P_t = \{p^1, p^2, \ldots, p^N\}$ be a set of points $p^j = (x_c^j, y_c^j, x_p^j)$ in the projective space estimated at time t and stored in a correspondence map X_p (see Fig. 3). Given that the p^j are all inliers and distributed (more or less some noise) on a plane, a typical way of calculating the best-fitting plane is by minimizing the square of the offsets. The offset of a point is usually its *perpendicular* distance to the plane. However, since our points lie on a rectangular lattice (the correspondence map X_p), we consider instead the offset along the third dimension of p^j since we expect errors to be on the x_p^j coordinate only.

Let $ax_c + by_c + cx_p + d = 0$ be the equation of a plane. Since a projection surface cannot be parallel to the viewing axis, one can set $c = 1$ to reduce by 1 the number of unknowns. Given a point (x_c^j, y_c^j, x_p^j), its depth according to the plane is $x_p = -(ax_c^j + by_c^j + d)$ and its *squared depth offset* is $(x_p^j - x_p)^2$. Thus, the best plane given P_t is the one which minimizes the depth offset for every point, namely

$$E(P, A) = \sum_j \left(\hat{p}^j A + x_p^j\right)^2 \tag{3}$$

where $\hat{p}^j = (x_c^j, y_c^j, 1)$ and $A = (a, b, d)^{\mathrm{T}}$. By forcing $dE/dA = 0$, one can show that $A = -M^{-1}B$ where

$$M = \begin{pmatrix} \sum_j (x_c^j)^2 & \sum_j x_c^j y_c^j & \sum_j x_c^j \\ \sum_j x_c^j y_c^j & \sum_j (y_c^j)^2 & \sum_j y_c^j \\ \sum_j x_c^j & \sum_j y_c^j & \sum_j 1 \end{pmatrix} B = \begin{pmatrix} \sum_j x_c^j x_p^j \\ \sum_j y_c^j x_p^j \\ \sum_j x_p^j \end{pmatrix}.$$

Let us mention that the estimated plane $[a, b, 1, d]$ can be transposed in the 3-D Euclidean space as follows: $T(a, b, 1, d)^{\mathrm{T}}$, where T is the 4×4 matrix defined in Appendix A. The reason why planes are fitted in the projective space (i.e., on the correspondence map X_p) and not on 3-D points in the Euclidean space is described in Appendix B.

5.4.2 Fitting m planes and dealing with outliers

Assuming that the projection surface is piecewise planar, we use a modified version of RANSAC to find m different planes with their respective set of inliers [25]. Since RANSAC can only fit one plane, we retained the following generalization:

```
. minsize ← s*size(P_t), i ← 1, exit ← false
DO
.    (inl[i],A[i]) ← RANSAC(P_t)
.    if (size(inl[i]) < minsize)
.      m ← i − 1, exit ← true
.    else
.      P_t ← remove the inliers inl[i] from P_t.
.      i ← i + 1
WHILE exit == false
```

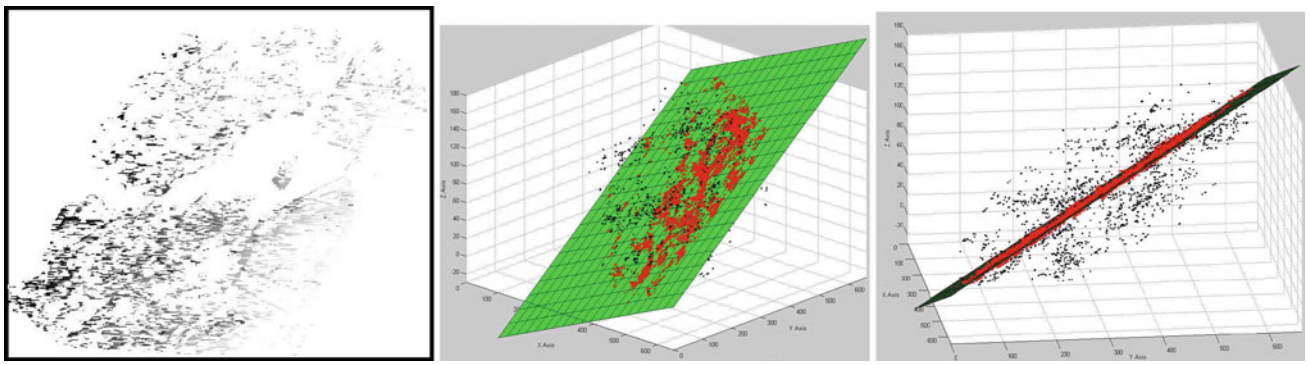

Fig. 3 *Left* Correspondence map X_p obtained at time t with our method. A correspondence has been assigned to each pixel at which activity has been recorded. (*middle* and *right*) two views of a plane estimated with RANSAC. Inliers are sitting on the plane while outliers are distributed in front and behind the plane. Outliers correspond to 17% of the population

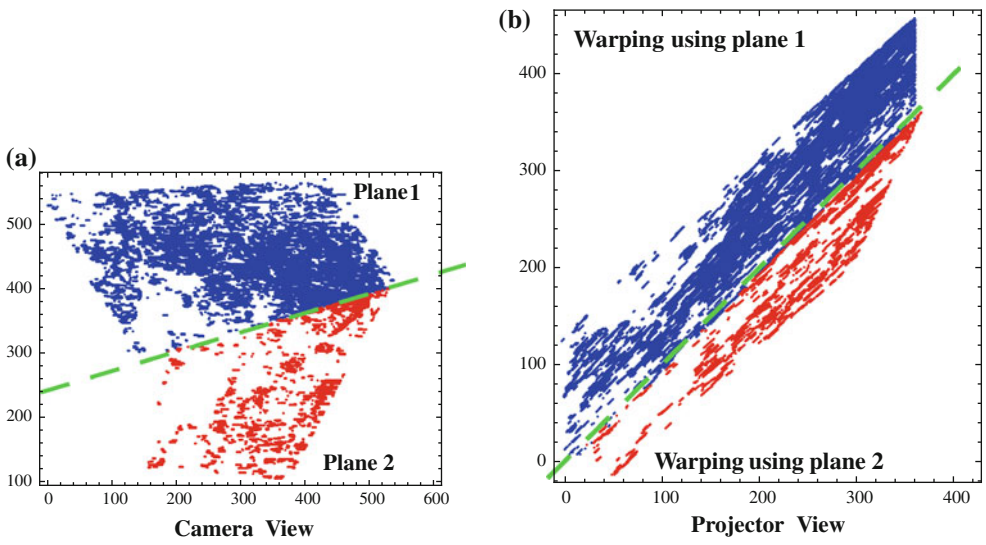

Fig. 4 3-D points recovered by our version of RANSAC as seen from the camera and the projector. The *dotted line* corresponds to the intersection between the two planes

where s is a fraction between 0 and 1. At the end, this procedure returns m plane equations (here A), their related inliers (here inl) and the outliers stored in P_t. Note that this algorithm does not need the number of planes m to be predefined. Similar plane fitting strategies can be found in [4,30]. For more details concerning RANSAC, please refer to [15].

5.5 Warping the projected video

Now that the m plane equations have been recovered, the projected video can be warped. For $m = 1$ plane, the procedure goes as follows:

1. Select a viewpoint for which the geometric correction must be performed. At this position, put a virtual camera and compute its extrinsic parameters with respect to the camera frame.
2. Assign intrinsic parameters to the virtual camera.
3. Using the plane equation and the extrinsic and intrinsic parameters, compute two homography matrices: one relating the projector and the camera and one relating the camera and the virtual camera (see [15] for more details).
4. From the two homography matrices, compute a third homography matrix relating the projector and the virtual camera.
5. Warp the projected image according to the third homography matrix.

Whenever $m > 1$ and the planes are connected (as in Fig. 4), the warping procedure must be applied for each plane:

- Find the intersection between each pair of planes so every pixel in the projector is associated to a plane (see Fig. 4b).
- Apply step 1 to 4.
- For each plane, apply step 5 to its corresponding area in the projected image.

6 Experimental protocol and results

In order to gauge performances, we tested our system on scenes made of one, two and four planes as well as scenes made of quadrics. We tested different videos containing different types of activity (see Fig. 5a, c, e). The first video is called *corridor* (CRD) and contains 167 frames. It is a surveillance video captured by a fixed camera and shows a pedestrian walking from the left to the right. The reason for this video is to evaluate how our method behaves on videos containing little activity. The second video, called *live band* (LB), contains 240 frames and shows a live music band filmed by a hand-held cellphone. This video suffers from severe compression artifacts and contains a lot of activity. The third video, called *traffic circle* (TC), contains 1,036 frames and was captured by a mid-end video camera. This is a typical home-made video showing a left-to-right pan of the scene with jitters due to the hand-held nature of the device.

For every test, we used the following parameters: $\alpha = 0.85$, $N = 6$ and $s = 0.1$. Results labeled as "ours" refer to our method using motion detection with an adaptive threshold and 3-D primitives fitted in the projective space. We also tested two projectors. The first one is a 1,576 × 1,080 projector that we used for the *corridor* sequence. The second one is a 225 lumen LED-based 1,024 × 768 projector that

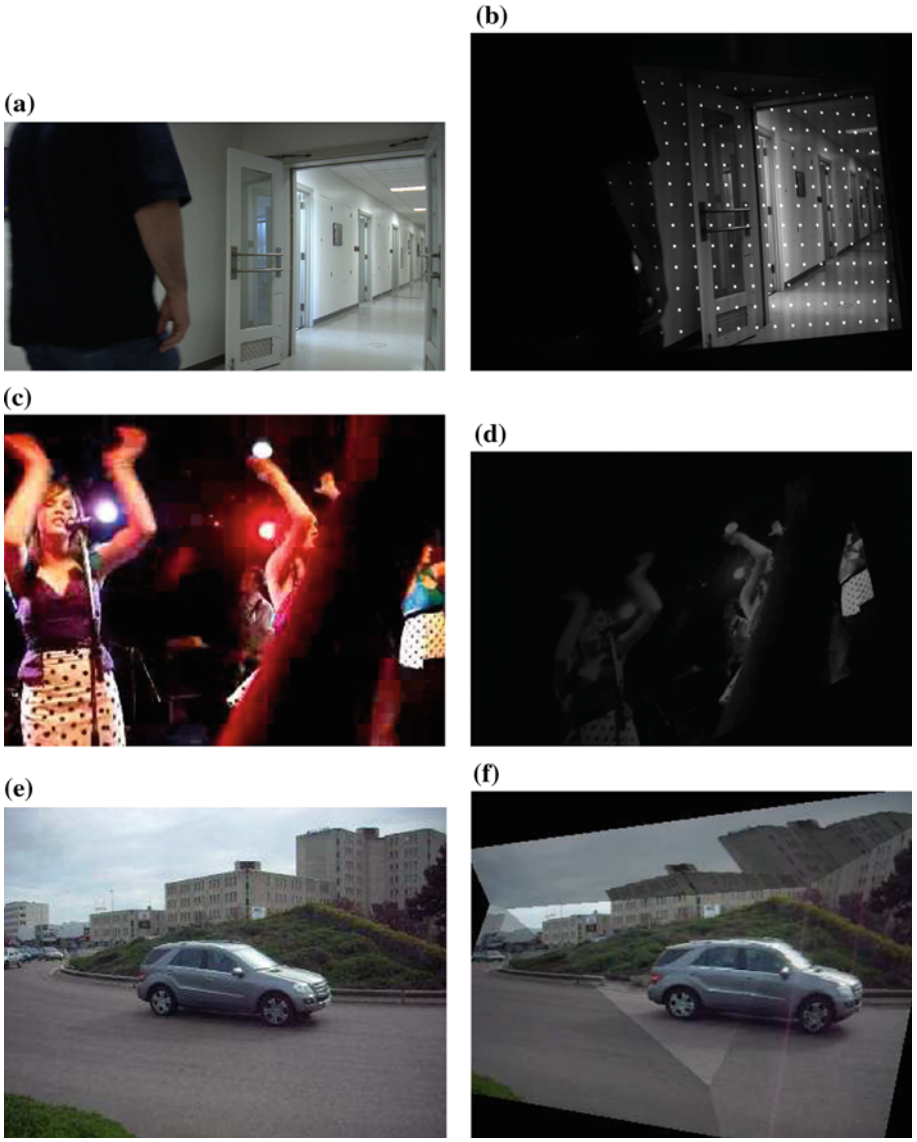

Fig. 5 The *corridor*, *live band* and *traffic circle* videos used for testing. On the *left*, the projected video and on the *right*, the captured video. Note that the lack of contrast in image (**b**) and (**d**) is caused by the shutter delay of the camera that we fixed to 1/30 s (image (**f**) has been acquired in a synthetic environment). Such delay makes sure the captured and projected videos are temporally aligned

we used for the *live band* sequence. We used a $3{,}024 \times 4{,}334$ camera whose images are reduced to fit the resolution of the projected videos.

6.1 RANSAC validation

As mentioned earlier, RANSAC is a robust solution to recover planes from 3-D scatter plots [25,21]. However, to our knowledge, no previous work in the field of plane fitting thoroughly tested RANSAC's robustness to noise and outliers. Our goal in this section is to show that RANSAC can successfully recover many planes even under harsh conditions. We first tested RANSAC on the ground truth disparity map of the *Venus* stereovision setup [29]. As shown in Fig. 6, this setup contains four planes which we heavily corrupted to see how RANSAC would behave. Surprisingly, RANSAC was still capable of recovering the four plane equations with <3° of error.

To further test the robustness of RANSAC, we generated scenes made of eight planes randomly positioned in a $100 \times 100 \times 100$ space. Each of these planes were assigned 5,000 points for a total of 40,000 points (read inliers). As shown in Table 1, for each scene, a certain number of outliers was added (between 0 and 50,000) and the resulting 3-D scatter plot was corrupted by white Gaussian noise with $\sigma \in \{0, 5, 10\}$. After running RANSAC on 270 different scenes, we came to realize that even in the worst case scenarios ($\sigma = 10$ and 50,000 outliers) RANSAC works on the average more than 75% of the time. That is to say, RANSAC recovers most of the time six planes out of eight. When the noise level and/or the number of outliers decreases, RANSAC recovers all eight planes almost every time.

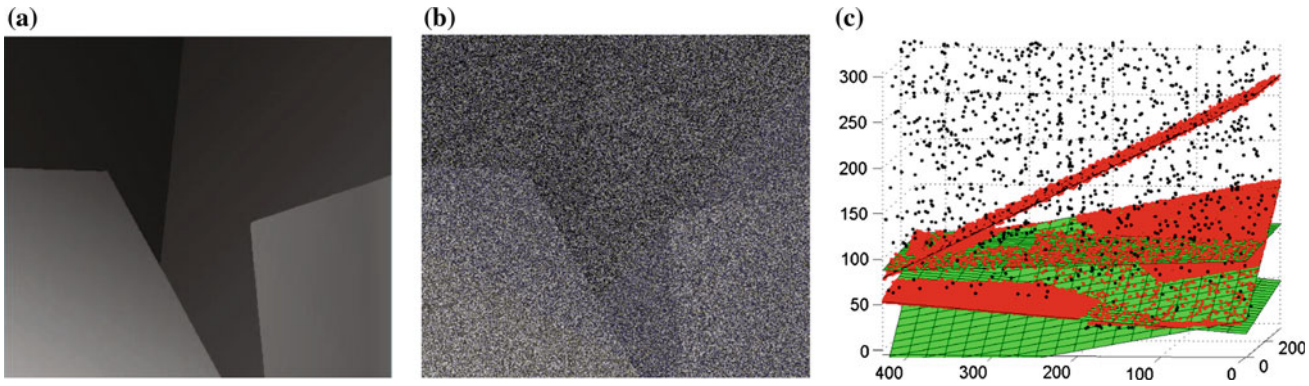

Fig. 6 **a** Ground truth disparity map for the *Venus* stereovision sequence [29], **b** with 25% of corrupted pixels. In **c** 3-D plot showing with four grids the planes recovered by RANSAC. Inliers are on the planes while outliers are distributed around it (most outliers lie outside the field of view of the camera)

Table 1 Percentage of true positives on scenes made of 40,000 points distributed on eight random planes

Noise level	Number of outliers		
	0	25,000	50,000
0	100	100	99.1
5	95.0	93.3	88.8
10	85.4	85.1	75.4

These scenes are corrupted by outliers (between 0 and 50,000) and white Gaussian noise. The standard deviation of noise σ is 0, 5 and 10

6.2 Single-plane setup

Here, videos were projected on a plane to see how our method behaves on a flat surface such as a screen or a wall. The target contains fiducial markers at known positions so the plane equation can be computed using a photogrammetric method (and thus be used as a ground truth). Since these markers are visible in the captured video (see Fig. 5b), it also allows to see how the system behaves when the video is projected on a textured surface.

We first tested the *corridor* sequence which is by far the most difficult sequence, due to the small amount of activity it contains. Given a temporal window W of 30 frames, we recovered the plane equation at each time t. As shown in Fig. 8, our method produces an average error of 2.8° between the ground truth plane and the estimated plane.[1] Figure 8 also shows that fitting a plane in the projective space is more robust than in the Euclidean space.

Figure 7 shows a camera frame at $t = 120$ and its related motion label field. This frame is where SIFT and the Euclidean plane fitting both failed (see Fig. 8). Interestingly, although the active pixels are within a small region of <3.8% of the image, our approach still accurately recovers the plane's position and orientation.

Also, as shown in Fig. 9, an error of 2.8° corresponds to a reprojection error of <4 pixels, which is barely noticeable by the human eye. This is obvious when considering Fig. 12a and b, in which a warped checkerboard has been projected on a planar surface.

We also tested the *live band* video sequence on a 3-D plane with $W = 30$. As shown in Fig. 3, an average of 13,545 inliers has been recovered and the estimated 3-D plane has an angular error of <3°.

6.3 Comparison with SIFT

Here, we implemented the processing pipeline described in Sect. 4, except for the matching procedure (Sect. 5.3), which we replaced by SIFT. Note that to make the comparison fair, every match found by SIFT at time t was propagated to the upcoming frames to allow for more matches.

We used the *corridor* sequence that we projected on a flat textured surface. As can be seen in Fig. 8, our method outperforms SIFT, as it produces a much lower angular error on the average. Due to the texture on the surface and severe spatial and photometric distortions, SIFT finds a small number of good matches (<50 as shown in Fig. 10) whose distribution is aligned at some time instant (see Figs. 9, 10). This leads to an average reprojection error three times larger than that obtained with our method.

Table 2 shows the number of inliers found by our method and by SIFT (RANSAC has been used to filter out outliers) for the *live band* and *corridor* sequences projected on a plane at four different angles. It also contains results obtained with a fixed and an adaptive threshold for our method's motion detection step. Since these tests were performed in a virtual environment, we added a contrast degradation factor between 1 and 4 to simulate the effect of a real-life camera whose images are often very dark due to a small shutter delay.

[1] Our method normally estimates the 3-D surface only once. Here, we estimated a plane at each time t in Figs. 8 and 11 to show how our method behaves at different time instants of the video.

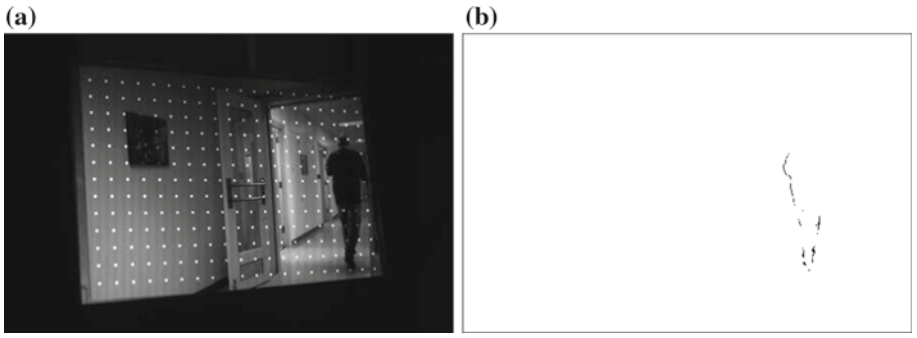

Fig. 7 Camera frame for the *corridor* sequence at $t = 120$, containing little activity (active pixel are in *black* in the *right* image)

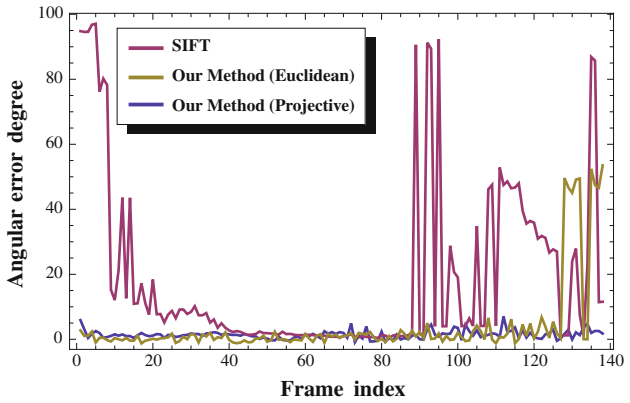

Fig. 8 Angular error between the ground truth plane and the plane estimated with our method and SIFT. Here, the *corridor* sequence has been used

Results clearly show that our method finds more matches than SIFT. In fact, our method (with an adaptive threshold) provides approximately 1,800 times more inliers than SIFT. The greatest variation between the maximum and minimum number of matches is 28.6% for our method as opposed to 99% for SIFT. This clearly shows that our method is more stable when dealing with low contrast video frames. This also shows the importance of adaptive thresholding as the number of inliers found with a fixed threshold decreases drastically when the camera image is badly contrasted.

6.4 Two-plane setup

In this test case, we projected the *live band* video on a two-plane wedge located 800 mm away from the camera. We first reconstructed the 3-D wedge with a structured light technique involving gray code and phase shift [28]. Then, we recovered the two plane equations with our method given a temporal window W of 15, 30 (with an adaptive threshold) and 60 (with a fixed threshold). The angular error for both plane at each time t is shown at the top of Fig. 11. At the bottom of Fig. 11, we superimposed the 3-D results obtained with structured light and our method (W = 30 with A.T.). As can be seen, the maximum error of our method is of only 4 mm which corresponds to 0.5% of error. The average depth error for both planes is -1.9 and -1.3 mm while the average angular error is approximately 1.5° with $W = 15$ and 0.9° with $W = 30$. We can see in Fig. 12c a warped checkerboard projected on the wedge and (d) its projection as seen from an arbitrary point of view. In comparison, a fixed threshold yields a larger variation between the average error of the two planes and requires more frames to obtain such results.

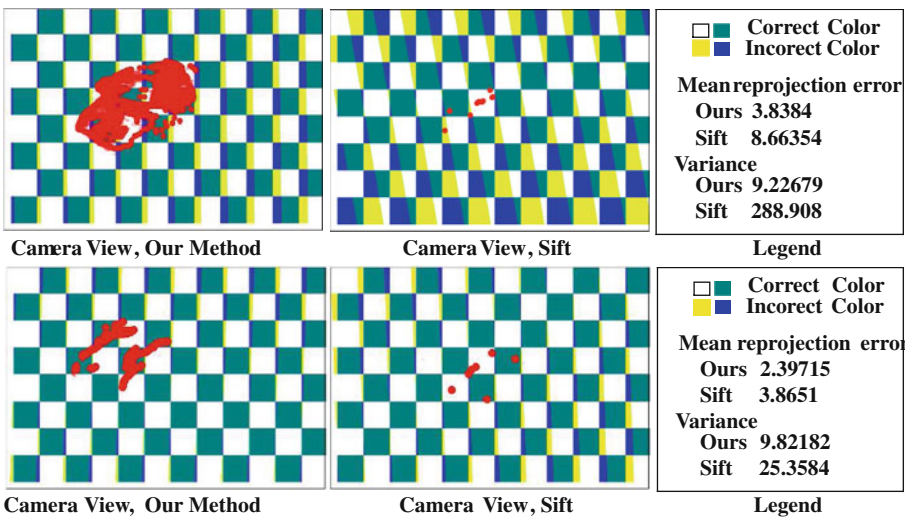

Fig. 9 Reprojection error of a flat checkerboard for the *corridor* sequence. The *dots* are the 3-D points recovered by our method (on the *left*) and SIFT (in the *middle*). Two time instants have been selected, namely $t = 43$ (*first row*) and $t = 140$ (*second row*)

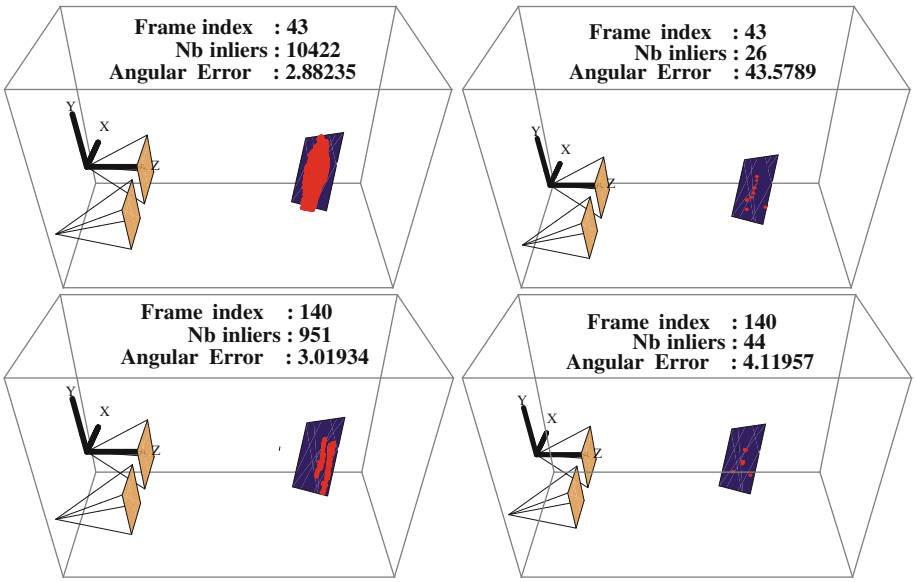

Fig. 10 3-D points recovered by our system (on the *left*) and SIFT (on the *right*) at $t = 43$ and $t = 140$ of the *corridor* sequence. The 3D points are plotted on top of the ground truth plane. As shown in Fig. 8, due to a small number of 3-D points found by SIFT at frame 43, the angular error rises above 40°

Table 2 Number of inliers found by SIFT and our method, using a fixed and an adaptive threshold (F.T. and A.T.) in the motion detection step

Sequence\angle	Number of inliers found			
	0°	5°	25°	50°
Ours (F.T.)-LB-1	66,373	62,873	63,845	43,581
Ours (F.T.)-LB-2	19,897	20,571	18,941	17,102
Ours (F.T.)-LB-4	3,414	3,352	3,192	2,735
Ours (F.T.)-CRD-1	62,186	59,763	57,393	52,375
Ours (F.T.)-CRD-2	6,071	6,309	5,910	5,109
Ours (F.T.)-CRD-4	824	823	670	637
Ours (A.T.)-LB-1	50,232	46,670	35,886	51,035
Ours (A.T.)-LB-2	49,483	41,300	36,457	43,952
Ours (A.T.)-LB-4	46,466	41,900	34,402	44,228
Ours (A.T.)-CRD-1	49,722	45,108	37,291	49,393
Ours (A.T.)-CRD-2	48,342	43,086	38,618	46,358
Ours (A.T.)-CRD-4	45,800	42,322	36,997	46,417
SIFT-LB-1	2,664	2,897	3,198	2,269
SIFT-LB-2	2,117	2,343	2,612	1,872
SIFT-LB-4	1,184	1,358	1,538	1,055
SIFT-CRD-1	276	337	398	331
SIFT-CRD-2	169	213	247	224
SIFT-CRD-4	25	38	49	47

The *live band* (LB) and *corridor* (CRD) sequences have been projected in a synthetic environment. The number next to each sequence's name (1, 2, and 4) is the contrast degradation factor that we applied to the images captured by the camera. Each sequence was projected on a plane tilted at four different angles with respect to the camera

Table 3 contains results for our method with a fixed and adaptive threshold (motion detection step) and with a temporal window of between 5 to 60 frames. This table clearly shows that the adaptive thresholding strategy is more precise and requires a smaller temporal window W. It also underscores the fact that a temporal window of 15 (half a second) is enough to have an error below 1°.

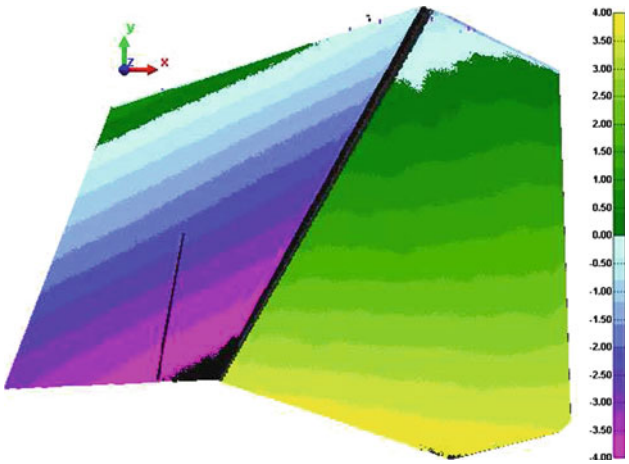

Fig. 11 *Top* Angular error at each time t for both planes using a fixed and an adaptive threshold. *Bottom* The difference in millimeters between the 3-D models obtained with structured light (14 patterns) and our method ($W = 30$ with AT)

6.5 Four-plane setup

In this setup, we projected the *traffic circle* video on a synthetic four-plane setup (Fig. 13a). Results with a temporal window of 30 are shown in Fig. 13c and d. Figure 13 c shows the inliers found by RANSAC in the projection space while Fig. 13d shows the four planes fitted on these inliers. The reprojection error of every inlier assigned to the correct plane

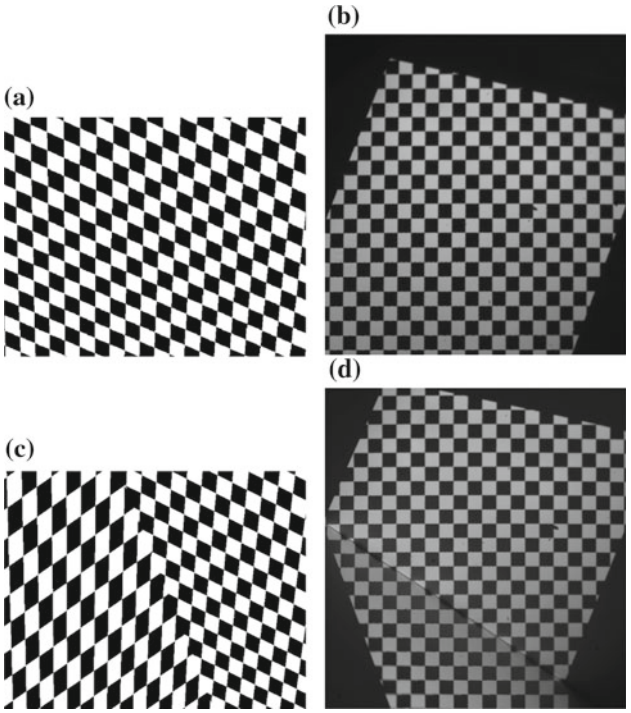

Fig. 12 **a**, **c** Warped video frames according to the 3-D shape recovered by our method (1 and 2 planes). **b**, **d** show the image of the projected image on the 3-D surface captured by the camera. As can be seen, the 4 pixel distortion is barely noticeable

Table 3 Different error metrics for the two-plane example using different temporal window sizes W

Sequence	Angular error			
	Mean	Median	Var	Max
$W = 60\,(FT)$	1.22	1.18	0.08	1.88
$W = 30\,(AT)$	0.86	0.77	0.12	1.64
$W = 25\,(AT)$	0.95	0.83	0.17	1.98
$W = 20\,(AT)$	1.27	1.14	0.30	3.14
$W = 15\,(AT)$	1.32	1.23	0.34	2.65
$W = 10\,(AT)$	1.62	1.63	0.72	3.75
$W = 5\,(AT)$	2.38	2.09	2.89	11.24
$W = 60\,(FT)$	1.66	1.47	1.57	6.41
$W = 30\,(AT)$	0.57	0.50	0.21	2.18
$W = 25\,(AT)$	0.60	0.50	0.22	2.21
$W = 20\,(AT)$	0.73	0.61	0.31	2.85
$W = 15\,(AT)$	1.0	0.84	0.49	3.23
$W = 10\,(AT)$	2.72	1.88	10.31	23.83
$W = 5\,(AT)$	37.89	36.15	469.62	80.85

The first seven lines are for plane 1 and the last seven lines for plan 2
AT stands for adaptive threshold, FT stands for fixed threshold

is of at most 1 pixel. The inliers assigned to the wrong plane (usually near the junction of two planes) have a reprojection error of at most 4 pixels.

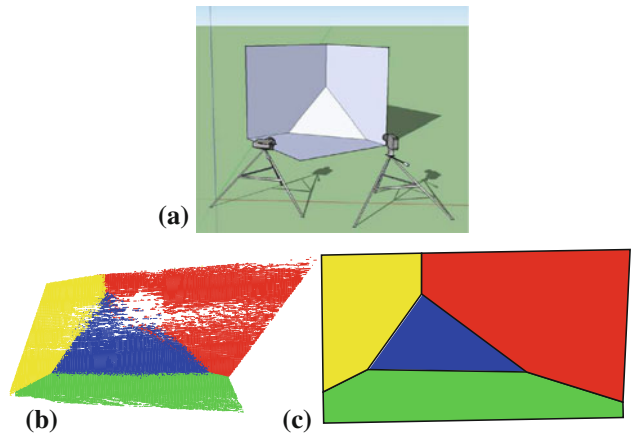

Fig. 13 **a** Schematic view of the four-plane setup and **b** point plot of the inliers in the projection space (with $W = 30$) and **c** the four planes obtained with RANSAC

6.6 Quadrics

We projected the *live band* video on a synthetic scene made of quadrics (two spheres) to show that the reconstruction method generalizes to other primitives. We used the same processing pipeline except for the fitting step which we adapted to quadrics [1,24]. Figure 14 shows results for five temporal windows W. The first row shows disparity maps while the second row show matches in the 3-D space. The third row shows the depth error in millimeters (from −100 to 100 mm). The spheres are located at 4,600 and 4,400 mm in front of the camera and have a radius of 1,750 and 1,105 mm. These results show that a larger temporal window increases the number of inliers while reducing the depth error, especially near the silhouette of the spheres.

6.7 Investigating errors

Since our method uses a pixel-to-pixel matching procedure, the camera-projector matches can only be accurate up to 1/2 pixel, i.e., up to the fundamental quantization error. We thus measured the imprecision of the 3-D geometry recovered in the two-plane setup and compared it to the theoretical quantization error. Table 4 contains the theoretical average quantization error given the orientation of the two planes (last column). The table also contains the average reprojection error of the planes recovered by our method (second column). As one can see, as the size of the temporal window W increases, the difference between the expected quantization error and the mean reprojection error decreases for both planes. At $W = 30$, the difference gets below 0.4 pixels. This leads us to believe that when $W \geq 30$, improvements in the cost function, the optimization method or the calibration procedure will not provide a significant improvement in the results. This underscores the fact that before deploying our

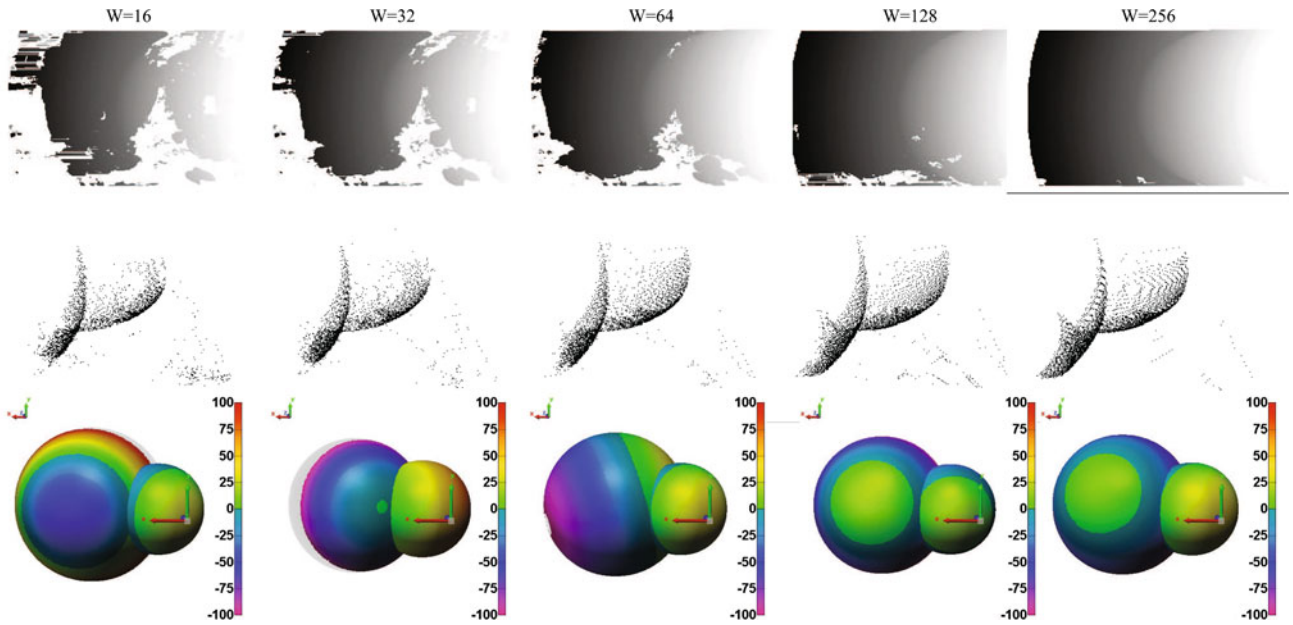

Fig. 14 Results obtained after projecting the *live band* video on two spheres and using five different temporal window W. The *first row* contains the disparity maps while the *second row* shows every match in the 3-D space as seen from a top view. The *third row* shows the depth error in millimeters for both spheres

Table 4 From left to right: temporal window parameter, average reprojection error of the two recovered planes, and expected quantization error obtained using simulation

Sequence	Reprojection error for the 2-plane example	
	Mean	Exp. mean
$W = 30$	0.86	0.55
$W = 25$	0.95	0.53
$W = 20$	1.27	0.54
$W = 15$	1.32	0.53
$W = 30$	0.57	0.20
$W = 25$	0.60	0.20
$W = 20$	0.73	0.20
$W = 15$	1.00	0.20

The first four lines are associated to plane 1 and the last four lines to plane 2. All results were obtained with an adaptive thresholding procedure

system, one can measure the expected quantization error for a given video sequence and use it as a lower error bound.

Another important source of error is the camera-projector miscalibration. This is often due to a bad calibration algorithm, an inaccurate projector-camera model, a bad calibration target, or simply a human error. As one might expect, a calibration error impacts different parts of our system, and especially the image rectification step. Since the DP optimizer works along epipolar lines (which are horizontal after rectification), a rectification error induces mismatches where

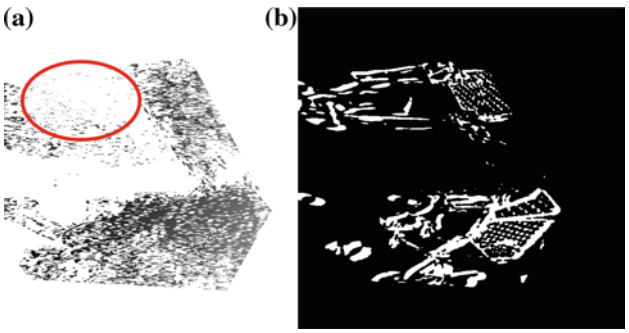

Fig. 15 **a** Correspondence map X_p with its related **b** motion label field at time t. Due to a misalignment of the epipolar lines and horizontally aligned active pixels, the surrounded area in X_p contains few good matches

the video contains horizontally aligned active pixels. Such alignment is illustrated on top of Fig. 15b. The impact of misaligned epipolar lines is illustrated on top of the correspondence map X_p in Fig. 15a. As can be seen, the activities detected in that region produced very few good matches.

7 Conclusion

We presented a camera–projector matching procedure based on activity features instead of colors. This *unstructured light* matching technique performs 3-D reconstruction using an arbitrary video sequence. To be robust to severe geometric

and photometric distortions, our method uses binary motion labels obtained from background subtraction bundled with grayscale quanta. We presented examples from which sparse correspondences are used to recover 3-D primitives that are then used for warping. Numerous experiments have been conducted on real and synthetic scenes. Out of these results, we conclude that:

1. Our method finds significantly more matches than SIFT, especially when the captured video suffers from severe geometric and photometric distortions, and when the projection surface is textured.
2. The 3-D results obtained with our method are close to those obtained with a state-of-the-art structured light technique (gray code + phase shift).
3. The estimated planes have on the average an error of $<3°$, leading to an average reprojection error between 1 and 4 pixels.
4. A temporal window of between 15 and 30 frames is required to find good matches. The length of the temporal window depends on the amount of activity in the video.
5. Our method generalizes well to other primitives such as quadrics.

Our method is motivated by applications requiring digital projection and 3-D reconstruction at the same time. One of the targeted applications is artistic projections for which the 3-D information (initially unknown) is needed to prewarp the projected video. Let us mention that a non-technically savvy artist could easily design visually aesthetic unstructured light patterns that could be used by our matching procedure. Furthermore, patterns for dense correspondences could also be designed using a few seconds of video. The only constraint being that every pixel of the projector be active at some point in time.

In the future, we look forward to extend the pixel matching procedure to a sub-pixel version. While our method requires the scene (and the camera/projector system) to remain static during the acquisition, we would like to combine our approach with featured tracking in order to allow for reconstruction in a dynamic environment.

Appendix A

We showed in Sect. 5.3 how to rectify the camera and projector frames so their epipolar lines are horizontal. This procedure is based on the intrinsic matrices $K_{c'}$ and $K_{p'}$. Interestingly, once $K_{c'}$ and $K_{p'}$ have been recovered, one can map a matching pair $(x_c, y_c), (x_p, y_c)$ in the rectified frames to its related 3-D point in the Euclidean space. This is done by multiplying a 4×4 matrix T to the 3-D point in the projective space $(x_c, y_c, x_p, 1)^{\mathrm{T}}$ where

$$T = \begin{pmatrix} bf_x^p f_y & 0 & 0 & -bf_x^p c_x^c f_y \\ 0 & bf_x^c f_x^p & 0 & -bf_x^c f_x^p c_y^c \\ 0 & 0 & 0 & bf_x^c f_x^p f_y \\ -f_x^p f_y & 0 & f_x^c f_y & (c_x^c f_x^p - c_x^p f_x^c) f_y \end{pmatrix} \quad (4)$$

and b is the baseline between the projector and the camera. Note that the 3-D point is in the coordinate system of the rectified pair and not in the reference frame of the original non-rectified pair (See [12,14] for more details).

Appendix B

As shown in appendix A, a point $(X_E, Y_E, Z_E)^{\mathrm{T}}$ in the 3-D Euclidean space can be obtained as follows:

$$(X_H, Y_H, Z_H, W_H)^{\mathrm{T}} = T(x_c, y_c, x_p, 1)^{\mathrm{T}}$$
$$(X_E, Y_E, Z_E)^{\mathrm{T}} = (X_H/W_H, Y_H/W_H, Z_H/W_H)^{\mathrm{T}}$$

where $(X_H, Y_H, Z_H, W_H)^{\mathrm{T}}$ is in homogeneous coordinates. Considering that noise is distributed along the "X_p" dimension and that the partial derivative of (X_E, Y_E, Z_E) is

$$\begin{pmatrix} \frac{\partial X_E}{\partial x_p} \\ \frac{\partial Y_E}{\partial x_p} \\ \frac{\partial Z_E}{\partial x_p} \end{pmatrix} = \frac{\left(bf_x^c f_x^p\right)}{(c_x^c f_x^p - c_x^p f_x^c - f_x^p x_c + f_x^c x_p)^2} \begin{pmatrix} c_x^c - x_c \\ \frac{\left(f_x^c\left(c_y^c - y_c\right)\right)}{f_y} \\ -f_x^c \end{pmatrix}.$$

the covariance of noise in the Euclidean space can be computed as follows

$$\Sigma = \sigma^2 \begin{pmatrix} \frac{\partial X_E}{\partial x_p} \\ \frac{\partial Y_E}{\partial x_p} \\ \frac{\partial Z_E}{\partial x_p} \end{pmatrix} \left(\frac{\partial X_E}{\partial x_p} \frac{\partial Y_E}{\partial x_p} \frac{\partial Z_E}{\partial x_p} \right).$$

where σ^2 is the variance of noise along the x_p dimension. By combining the last two equations, one can see that noise in the Euclidean space is anisotropic and varies with the distance of a point to the camera. Thus, in order to fit a primitive in the Euclidean space, one has to take into account the anisotropic nature of noise. That is why fitting planes and quadrics in the projective space is simpler and more robust (see Sect. 6.2).

Appendix C

In this section, we explain how we implemented the dynamic programming (DP) optimizer. Since our DP algorithm processes horizontal epipolar lines, let us first simplify notation by representing a camera pixel by x_c and a projector pixel by x_p. This leads to a slight redefinition of the cost function's signature which becomes $C\left(\Gamma^c, \Gamma^p, x_c, x_p\right)$.

What makes our DP implementation different from the other ones is the fact that pixels with no activity are assigned a constant cost function of zero. In this way, our method only considers matches $x_c \rightarrow x_p$ with non-zero activity and lying within the search interval defined previously. For the rest of this section, each inactive projector pixel will be assign the "−1" index.

Like most DP methods used in stereovision [3,22,34], our implementation finds the shortest path within a 2D grid that we call $t(x_c, x_p)$.

Since our DP implementation runs from left to right, $t(x_c, x_p)$ contains the cost of the optimal path from 0 to x_c with the pixel x_c match to x_p. For all camera pixels matched with an inactive projector pixels,

$$t(0,-1) = 0$$
$$t(x_c,-1) = \min_{\hat{x}_p} \left(t(x_c-1,\hat{x}_p)\right)$$

while for camera pixels matched with active projector pixels

$$t(0,x_p) = C(\Gamma^c,\Gamma^p,0,x_p)$$
$$t(x_c,x_p) = C(\Gamma^c,\Gamma^p,x_c,x_p) + \min_{\hat{x}_p} \begin{pmatrix} t(x_c-1,\hat{x}_p) \\ +s(x_c,x_p,\hat{x}_p) \end{pmatrix}$$

where $s(x_c, x_p, \hat{x}_p)$ is a function that becomes ∞ whenever the ordering occlusion constraint is broken for the $x_c \leftrightarrow x_p$ match. Explicitly,

$$s(x_c,x_p,\hat{x}_p) = \begin{cases} \infty & \text{if } x_p \le o(x_c-1,\hat{x}_p) \\ 0 & \text{otherwise.} \end{cases}$$

where $o(x_c - 1, \hat{x}_p)$ contains the index of the rightmost projector pixel match with a camera pixel in the interval 0 to $x_c - 1$ for the shortest path from 0 to $x_c - 1$ with the pixel $x_c - 1$ match to $\hat{x}_p$. Note that o can be computed recursively and this is similar to the recursive occlusion computation presented in [8]. Also, the entries in table t for inactive camera pixels do not need to be explicitly computed. Moreover, to further speed up the process, a *fast message passing strategy* can be used [10] to reduce DP's complexity to that of WTA.

References

1. Al-Subaihi, I.A., Watson, G.A.: Algebraic fitting of quadric surfaces to data. Commun. Appl. Anal. **9**, 539–548 (2005)
2. Ashdown, M., Sato, Y.: Steerable projector calibration. In: Proceedings of the IEEE International Work. Projector–Camera Systems (2005)
3. Belhumeur, P.N.: A Bayesian approach to binocular stereopsis. Int. J. Comput. Vis. **19**(3), 237–260 (1996)
4. Boulaassal, H., Landes, T., Grussenmeyer, P., Tarsha Kurdi, F.: Automatic segmentation of building facades using terrestrial laser data. In: ISPRS Workshop on Laser Scanning and SilviLaser, pp. 65–70 (2007)
5. Cotting, D., Ziegler, R., Gross, M., Fuchs, H.: Adaptive instant displays: continuously calibrated projections using per-pixel light control. Comput. Graph. Forum **24**(3), 705–714 (2005)
6. Davis, J., Nehab, D., Ramamoorthi, R., Rusinkiewicz, S.: Spacetime stereo: a unifying framework for depth from triangulation. IEEE Trans. Pattern Anal. Machine Intell. **27**(2), 296–302 (2005)
7. Drouin, M.-A., Jodoin, P.-M., Premont, J.: Camera–projector matching using an unstructured video stream. In: Proceedings of the IEEE International Work. Projector–Camera Systems, pp. 1–8 (2010)
8. Drouin, M.-A., Trudeau, M., Roy, S.: Fast multiple-baseline stereo with occlusion. In: Proceedings of the Interantional Conference 3-D Digital Imaging and Modeling, pp. 540–547 (2005)
9. Egnal, G., Wildes, R.P.: Detecting binocular half-occlusions: empirical comparisons of five approaches. IEEE Trans. Pattern Anal. Machine Intell. **24**(8), 1127–1133 (2002)
10. Felzenszwalb, P.F., Huttenlocheri, D.P.: Efficient belief propagation for early vision. Int. J. Comput. Vis. **70**(1), 41–54 (2006)
11. Fiala, M.: Artag, a fiducial marker system using digital techniques. In: Proceedings of the IEEE Conference. Comp. Vis. Pattern. Recogn., pp. 590–596 (2005)
12. Fusiello, A., Trucco, E., Verri, A.: A compact algorithm for rectification of stereo pairs. Mach. Vis. Appl. **12**(1), 16–22 (2000)
13. Gluckman, J., Nayar, S.K.: Rectifying transformations that minimize resampling effects. In: Proceedings of the IEEE Conference. Comp. Vis. Pattern Recogn., pp. I:111–117 (2001)
14. Hartley, R., Sturm, P.: Triangulation. Comput. Vis. Image Understand. **68**(2), 146–157 (1997)
15. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision 2nd edn. Cambridge University Press, Cambridge (2004)
16. Heckbert, P.: Color image quantization for frame buffer display. Comput. Graph. **16**, 297–307 (1982)
17. Hirschmuller, H.: Accurate and efficient stereo processing by semi-global matching and mutual information. In: Proceedings of the IEEE Conference. Comp. Vis. Pattern Recogn., pp. 807–814 (2005)
18. Johnson, T., Fuchs, H.: Real-time projector tracking on complex geometry using ordinary imagery. In: Proceedings of the ACM/IEEE International Workshop on Projector Camera Systems, pp. 1–8 (2007)
19. Johnson, T., Welch, G., Fuchs, H., Force, E.L., Towles, H.: A distributed cooperative framework for continuous multi-projector pose estimation. In: IEEE Virt. Reality Conf., pp. 35–42 (2009)
20. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**(2), 91–110 (2004)
21. Mufti, F., Mahony, R., Heinzmann, J.: Spatio-temporal ransac for robust estimation of ground plane in video range images for automotive applications. In: IEEE ITS, pp. 12–15 (2008)
22. Ohta, Y., Kanade, T.: Stereo by intra- and inter-scanline using dynamic programming. IEEE Trans. Pattern Anal. Machine Intell. **7**(2), 139–154 (1985)
23. Otsu, N.: A threshold selection method from gray-level histograms. IEEE Trans. Syst. Man Cybern. **9**(1), 62–66 (1979)
24. Pratt, V.: Direct least-squares fitting of algebraic surfaces. SIGGRAPH **21**(4), 145–152 (1987)
25. Quirk, P., Johnson, T., Skarbez, R., Towles, H., Gyarfas, F., Fuchs, H.: Ransac-assisted display model reconstruction for projective display. In: IEEE Virt. Reality Conf., p. 318 (2006)
26. Raskar, R., Beardsley, P.: A self-correcting projector. In: Proceedings of the IEEE Conference. Comp. Vis. Pattern Recogn., pp. II:504–508 (2001)
27. Raskar, R., van Baar, J., Beardsley, P., Willwacher, T., Rao, S., Forlines, C.: iLamps: geometrically aware and self-configuring projectors. In: ACM Trans. Graph., p. 5, ACM (2003)
28. Salvi, J., Pages, J., Batlle, J.: Pattern codification strategies in structured light systems. Pattern Recogn. **37**(4), 827–849 (2004)

29. Scharstein, D., Szeliski, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. Int. J. Comput. Vis. **47**(1–3), 7–42 (2002)
30. Shimamura, J., Arai, H., Yasuno, T.: Multiple plane detection for flexible projection. In: Proceedings of the IEEE International Work. Projector–Camera Systems, pp. 1–2 (2006)
31. Tardif, J., Trudeau, M., Roy, S.: Multi-projectors for arbitrary surfaces without explicit calibration nor reconstruction. In: Proceedings of the International Conference. 3-D Digital Imaging and Modeling, pp. 217–224 (2003)
32. Yang, R., Welch, G.: Automatic projector display surface estimation using every-day imagery. In: Proceedings of the International Conference. Cent. Eur Comput. Graph. Vis. Comput. Vis. (2001)
33. Yapo, T., Sheng, Y., Nasman, J., Dolce, A., Li, E., Cutler, B.: Dynamic projection surfaces for immersive visualization. In: Proceedings of the IEEE International Work. Projector–Camera Systems, pp. 1–8 (2010)
34. Zhang, L., Curless, B., Seitz, S.: Rapid shape acquisition using color structured light and multi-pass dynamic programming. In: International Symposium. 3D Data Proc. Vis Transm., pp. 24–36 (2002)
35. Zhang, L., Curless, B., Seitz, S.: Spacetime stereo shape recovery for dynamic scenes. In: Proceedings of the IEEE Conference. Comp. Vis. Pattern. Recogn., pp. 367–374 (2003)
36. Zhang, L., Nayar, S.: Projection defocus analysis for scene capture and image display. ACM Trans. Graph. **25**(3), 907–915 (2006)
37. Zhou, J., Wang, L., Akbarzadeh, A., Yang, R.: Multi-projector display with continuous self-calibration. In: Proceedings of the ACM/IEEE International Workshop on Projector camera systems, pp. 1–7 (2008)
38. Zollmann, S., Langlotz, T., Bimber, O.: Passive-active geometric calibration for view-dependent projections onto arbitrary surfaces. J. Virt. Real. Broadcasting **4**(6), 1–10 (2007)

Author Biographies

Marc-Antoine Drouin received his PhD in computer science from Université de Montréal in 2007. The topic of his thesis was on stereo vision and occlusion handling. Since 2007, he has been working at the National Research Council Canada as a research officer. He now works on 3D imaging system with a special interest on fringe projection systems.

Pierre-Marc Jodoin studied physics at Université de Montréal (1994) and received his B.Sc. degree in computer science at the École Polytechnique de Montéal, Canada (1995–2000). He then obtained his M.Sc. degree in computer graphics (2002) and the PhD degree in computer vision and video analysis at the University of Montreal (2007). Dr. Jodoin received both M.Sc. and PhD degrees with honours. He is now associate professor at the Université de Sherbrooke, Canada. His research interests are in video surveillance, video analysis/processing, medical imaging, and computer vision. More information about his work is available at http://www.dmi.usherb.ca/~jodoin/.

Julien Prémont studied computer science at the Université de Sherbrooke. He received his B.Sc. and M.Sc. degrees in 2008 and 2011, respectively. His research interests are in video analytics, computer vision and 3-D reconstruction. He is now a research engineer at Nuance Communications.