EURASIP Journal on Applied Signal Processing 2002:10, 1088–1101

# Sliding Adjustment for 3D Video Representation

**Franck Galpin**
*IRISA/INRIA Rennes, Université de Rennes 1, Campus de Beaulieu, 35042 Rennes Cédex, France*
*Email: franck.galpin@irisa.fr*

**Luce Morin**
*IRISA/INRIA Rennes, Université de Rennes 1, Campus de Beaulieu, 35042 Rennes Cédex, France*
*Email: luce.morin@irisa.fr*

*Received 31 August 2001 and in revised form 15 February 2002*

This paper deals with video coding of static scenes viewed by a moving camera. We propose an automatic way to encode such video sequences using several 3D models. Contrary to prior art in model-based coding where 3D models have to be known, the 3D models are automatically computed from the original video sequence. We show that several independent 3D models provide the same functionalities as one single 3D model, and avoid some drawbacks of the previous approaches. To achieve this goal we propose a novel algorithm of sliding adjustment, which ensures consistency of successive 3D models. The paper presents a method to automatically extract the set of 3D models and associate camera positions. The obtained representation can be used for reconstructing the original sequence, or virtual ones. It also enables 3D functionalities such as synthetic object insertion, lightning modification, or stereoscopic visualization. Results on real video sequences are presented.

**Keywords and phrases:** sliding adjustment, 3D model reconstruction, video coding, model-based coding, video manipulation.

## 1. INTRODUCTION

More and more new coding techniques include high-level information in video sequence representation. This information aims to provide high-level functionalities such as interactivity, video content description, video manipulation, or stereo visualization.

For instance, the QuickTime-VR format provides the functionality of interactive visualization of a real static scene, by representing it as a panoramic image [1].

MPEG4 standard describes the video scene content as a set of plane objects called video object plane (VOP) [2], which can be interactively moved or combined during visualization. A panoramic representation of static backgrounds is also proposed in MPEG4 with the Sprite format.

Such 2D representations do not give information on the 3D structure of the scene, and are therefore limited for video manipulation. Panoramic images provide only limited interactivity: zoom and view orientation can be changed but the view-point is fixed. With 2D representations, video manipulation such as hybrid synthetic-real video mixing, involving occlusions, shadows, lightning modification are not feasible in a realistic way. These functionalities require 3D information on the scene.

3D model-based representations for real video sequences have been studied for a long time, since they have very attractive properties. Apart from the functionalities that they provide, they enable very low bit rates and scalable/progressive coding [3].

3D model-based representations can be classified into explicit and implicit representations. Within the explicit representations, we can distinguish representations with known 3D models and unknown 3D models.

In 3D model-based coding with known models, a 3D model of the object in the scene is available, for instance, a textured 3D triangular mesh. The video sequence is processed to compute the 3D object pose (orientation and scaling) for each frame. Sometimes, local deformations are also computed. The video sequence is represented as the 3D model and pose parameters for each frame, with optional parameters for texture and local shape deformation. This representation allows to transmit the original video at low cost and facilitates any 3D manipulation functionalities. This approach is widely used for head and shoulder video sequences coding and body animation analysis and representation [4, 5], for instance, in the MPEG4-SNHC scheme [6]. Its main drawback is that it can only be applied to video with specific contents, such as manufactured objects, head, or body.

In the 3D model-based coding with unknown models, the same principle is applied but the scene contents are unknown and the 3D model shape must also be estimated from the video itself. Since shape and nonrigid motion cannot be

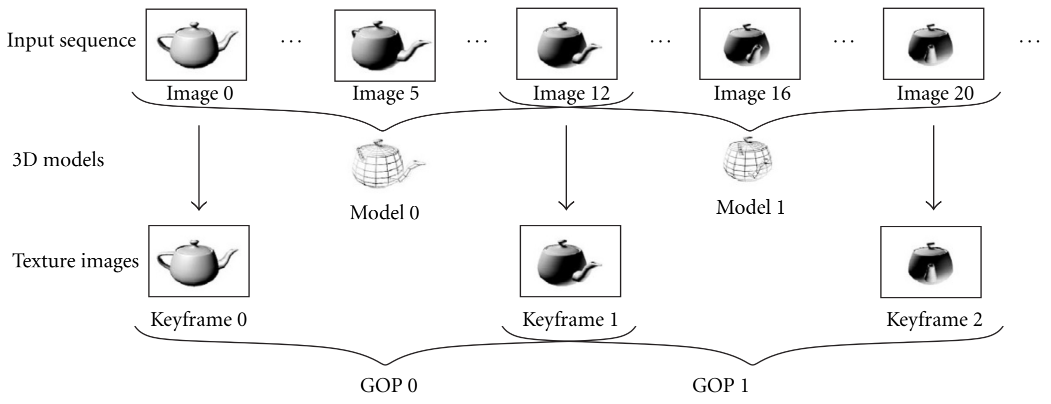

FIGURE 1: Principle of video sequence representation using several overlapped 3D models.

separated, this approach can only be applied to video containing one object undergoing rigid motion, or alternatively, a fixed object viewed by a moving camera. Using computer vision tools [7, 8], the 3D model shape, texture, and rigid motion are estimated from the video sequence. Once the 3D model is computed, it can then be used just as a known 3D model. This approach is limited to fixed objects, but it is very attractive for applications such as realistic modeling of complex objects or interactive navigation in real environments. The most sensitive step is the estimation of camera internal parameters, or self-calibration. A theoretical solution has been established for long. Robust solutions for camera calibration and pose estimation have been proposed in recent work, and are effective for video sequence acquired with a hand-held camcorder [9]. Another solution to deal with generic video data requires that a priori high-level information is integrated in the process through manual and sometimes expert user interaction (http://www.realviz.com).

Such an approach is thus not yet applicable to design a coding algorithm which automatically produces a unique 3D model from a generic and long video sequence of a static scene without simplifying assumptions on the scene or acquisition.

Alternative approaches based on implicit 3D model-based coding have been proposed. The lightfield or the lumigraph [10, 11] do not aim to reconstruct an explicit 3D model but provide some of the same functionalities. However, data acquisition is very constraining, as view-points must lie on a dense regular grid. Some other approaches introduce depth information in the encoded sequence, which allows stereo sequence visualization or scene manipulation [12], but which often requires stereo acquisition.

In this paper, we present an original representation of video sequences using several unknown 3D models and we propose a novel algorithm for ensuring high-level 3D functionalities using this representation. This representation can be applied in the case of a fixed scene viewed by a moving camera. We present an automatic scheme for extracting the proposed representation from the video sequence.

Some previous studies try to extract a single 3D model with a hierarchical and robust estimation of camera positions [9, 13]. A self-calibration step is also performed, allowing the reconstruction of a single 3D model. Such methods generally require a specific type of camera motion (typically a closed image sequence or an inspecting image sequence), in order to perform the self-calibration step. In this paper, we deal with video coding, assuming very long video sequences, we thus need an on-the-fly process. Moreover, we want to deal with any type of camera motion. For instance, one typical application could be navigation on a walking path where camera motion is naturally a rough frontal translation. This type of motion is closed to degenerate cases and scene points appear in a small part of the sequence. Thus we do not make the assumption that it is possible to obtain a reliable, accurate camera self-calibration for any sequences.

Unlike the classical approach of video coding with unknown models described before, we do not aim at reconstructing one single realistic 3D model of the scene. Instead, we compute a succession of 3D models, each 3D model being adequate to represent a small part of the video sequence. This approach can be viewed as an intermediate between 2D motion compensation video-coding and 3D model-based coding. Just as in the 2D approach, one 3D model can be considered as a global motion model which best fits 2D motion in the original video sequence for a group of images (also called here a GOP). Once this motion model is not valid anymore, the GOP is ended and a new 3D model is estimated for representing the next part of the video (see Figure 1). Thus, successive 3D models may contain the same parts of the 3D scene, but each model is related to a specific GOP of the video. Also GOPs size is not fixed but data driven, thus variable.

In order to obtain the same functionalities with several 3D models as with a single 3D model, we propose a novel algorithm of sliding adjustment which ensures consistency of successive 3D models. This step enables applications such as synthetic 3D objects insertion into the video sequence, lightning modification or interactive navigation.

Using several 3D models instead of one single model has several drawbacks:

(i) it is of course less compact, since the models are redundant;

(ii) it is not as simple and easy to insert synthetic data in the video sequence. With several 3D models, an adjustment step is required;

(iii) the set of acceptable virtual views (views reconstructed from a view-point outside the acquisition set) is smaller, since the 3D models are constrained to be consistent only with a subset of original views.

In counterparts, such a representation has several advantages:

(i) global consistency of estimated 3D shape and camera motion along the video sequence is no more expected. Thus, a valid representation is provided even in the case of ill-conditioned camera motion or inaccurate internal camera parameters;

(ii) such a representation is very well suited for large environments, where the observed scene part progressively changes along the sequence;

(iii) the representation is robust to small violations of the rigidity constraint, for instance presence of small moving objects or specular surfaces. Such data is temporarily taken into account by the 3D models, as a change in the model geometry;

(iv) 3D models are constructed sequentially. Streaming of the representation can be easily achieved for communication and compression purposes. This point is particularly important for very long sequences (several thousands of images) for which the automatic reconstruction of a single 3D model is very complex and computationally intensive.

This representation is thus very well suited for representing large natural scenes such as the ones acquired by outdoors walk through in cities, parks, with uncontrolled camera motion. Applications concern virtual tours in realistic environments, with possibility of scene manipulation and interactive navigation.

The paper is organized as follows. We first present the principle of video representation using several 3D models, the coding scheme and visualization procedure. We explain how this representation can be used to regenerate the initial video sequence, as well as virtual ones. The motion estimation step is briefly described because it uses classical techniques. We then present in more details the automatic selection of variable sized GOPs, and the sliding adjustment algorithm. Finally, we validate the method on real applications such as interactive navigation, virtual lightning, synthetic object insertion or stereoscopic visualization. Examples of the obtained results on real video sequences are presented and discussed.

## 2. VIDEO CODING USING SEVERAL UNKNOWN 3D MODELS

Our approach is quite similar to 3D model based coding using a single unknown 3D model: shape and texture are estimated from the video sequence itself, using shape from motion techniques. Camera motion for each frame is also estimated from the video sequence. The following assumptions are made: we use perspective projection model and we assume that the observed scene is fixed, contains mostly Lambertian objects, and is not entirely planar. The same scheme can be applied to a fixed background if moving objects have been segmented out from the video, as an alternative to the MPEG4-Sprite mode, for instance.

In our approach, camera internal parameters are not necessarily known. If not provided they are affected by arbitrary values. Camera motion is not constrained. We yet assume that camera motion is not a pure rotation around optical center.

Instead of computing one single model for the whole sequence, several 3D models are computed for the same 3D scene. Each 3D model is relative to a GOP (see Figure 1). More precisely, for a given GOP, the 3D model and associated camera positions are estimated from the images in the GOP. At the decoder, the estimated model is projected onto the estimated camera positions to reconstruct these images. The coding scheme is thus sequential, as in classical motion-compensation video-coders. The 3D model is expected to best fit the 2D information in the GOP, by minimizing a cost function based on MSE. Thus 3D shape may not be realistic as long as it allows reconstruction of the original sequence with minimum distortion. Subsequently, 3D models for successive GOPs may represent the same 3D object but they may have different shape, different texture, and even different scale.

Only a subset of images are used for 3D reconstruction of the 3D models. These images are called keyframes. Two successive keyframes are used to compute one 3D model. They are the first image and last image of the GOP associated with this 3D model. One keyframe is used as texture image for defining the 3D model texture. All keyframes are thus part of the representation, as texture images of the 3D models. Successive GOPs overlap by one keyframe. Keyframes can thus be reconstructed at the decoder either using one 3D model or the other. This is important for smooth transition between GOPs during visualization (note that what we call a keyframe is different from keyframes delimiting shots in video structure analysis).

3D model reconstruction from two views extracted from a video is known to be very sensitive to the choice of these two views: several criteria must be verified in order for the estimation to be geometrically and numerically stable. This is the reason why GOP size cannot be fixed. GOP size varies depending on data driven keyframe choice. We propose a robust method to select keyframes, which is described in Section 5.

For simple visualization of the original sequence, 3D models can be completely independent: they can have different scales and they can be described in different reference frames. This is still true for visualization along a virtual path, as long as this path contains all the camera positions associated with the keyframes. At these specific viewpoints, transition between 3D models is smooth, because both models, though different in 3D space, project onto the same 2D image: the common keyframe.

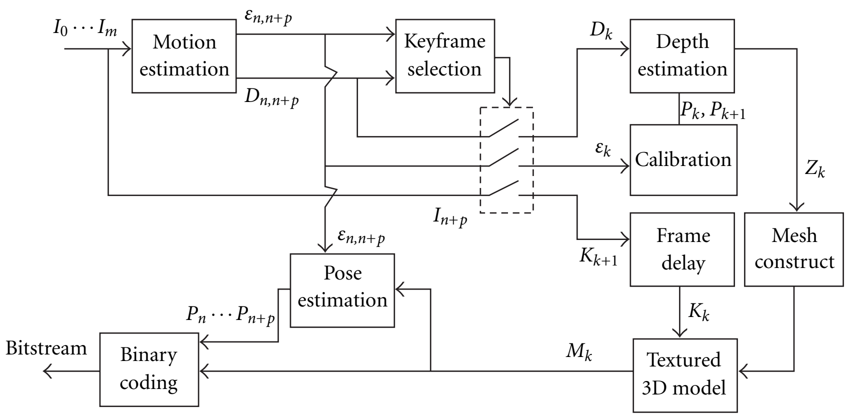

Figure 2: Block diagram of the 3D encoder.

However, independent 3D models are not suited for other 3D augmented reality functionalities, like inserting objects or lights. The 3D object position should be specified in a reference frame valid along the whole sequence. With independent 3D models for each GOP, this is not feasible. We thus propose an accurate method to set a 3D model compatible with the previous and next models. This method is derived from the classical bundle adjustment method, but it is specifically adapted to sequential processing and minimum distortion coding purposes. This algorithm is called *sliding adjustment* and it will be described in Section 6.

The general coding scheme is shown in Figure 2.

## 3. BACKGROUND

In this section, we briefly remember the principle of 3D model reconstruction from 2 views and we set the notations used in the paper.

Consider two images $I$ and $I'$ viewed by cameras $\mathcal{C}$ and $\mathcal{C}'$. We denote by $(R, t)$ the relative rigid transformation between cameras, and by $A$ the matrix containing the internal parameters of the cameras.

Let $\mathcal{E}$ be a set of matched points between images $I$ and $I'$. For $m_i$ a point of $\mathcal{E}$ in $I$ and $m'_i$ a point of $\mathcal{E}$ in $I'$, we have the following relation in homogeneous coordinates:

$$\tilde{m}_i = P \cdot \tilde{M}_i, \qquad \tilde{m}'_i = P' \cdot \tilde{M}_i, \tag{1}$$

where $P = A \cdot (I_3 \mid 0)$ and $P' = A(R \mid t)$.

If $m_i$ and $m'_i$ are matched, then the epipolar constraint is expressed in homogeneous coordinates as

$$\tilde{m}_i'^T \cdot F \cdot \tilde{m}_i = 0, \tag{2}$$

where $F$ denotes the well-known fundamental matrix. The matrix $F$ is defined as $F = A^{-T} \cdot [t]_\times \cdot R \cdot A^{-1}$ and $[\cdot]_\times$ is the matrix associated with the cross-product. We denote by $E = [t]_\times \cdot R$ the essential matrix associated with $F$.

We define the *epipolar residual* associated with $F$ as the sum

$$\frac{1}{2} \sum_i d(m_i, F \cdot m'_i) + d(m'_i, F^T \cdot m_i) \tag{3}$$

computed on all matched points $m_i$, $m'_i$ in $\mathcal{E}$. Nullity of the epipolar residual means that the symmetrical epipolar constraint defined by $F$ is verified for all points in $\mathcal{E}$. If this residual is small (i.e., with sub-pixel value), we then say that $F$ is consistent with $\mathcal{E}$.

In the following, we denote by $K_k$ the keyframe images. For a given keyframe image $K_k$, $R_k$, $t_k$ denote the camera motion parameters and $O_k$ denotes the center of projection for the corresponding camera, that is, $O_k = -R_k^{-1} \cdot t_k$.

Keyframes $K_k$ and $K_{k+1}$ delimit a GOP and $\mathcal{M}_k$ denotes the 3D model associated to this GOP, and $\mathcal{E}_k$ denotes a set of points matched between $K_k$ and $K_{k+1}$.

We also denote by $\vec{u}(m)$ the unitary tangent vector of the view-line passing through pixel $m$ in image $K_k$ (i.e., line $(O_k, m)$).

## 4. 2D AND 3D MOTION ESTIMATION

### 4.1. 2D motion estimation

#### 4.1.1 Dense motion estimation

Motion estimation is performed between two current images $I = I_n$ and $I' = I_{n+p}$. Usually $I = K_k$ is the last selected keyframe and $I'$ is a subsequent image in the video, which is evaluated as a potential next keyframe $K_{k+1}$.

Motion estimation is provided by previously developed algorithms. We use a mesh-based motion estimator based on a multi-resolution scheme over hierarchical meshes. It allows dense estimation between current images $I$ and $I'$, by successive estimation/relaxation steps between successive images [14]. This motion estimator provides a dense motion field between $I$ and $I'$. For each pixel $m_i$ in image $I$, its 2D displacement is a 2D vector denoted $D(m_i)$ and its corresponding position in image $I'$ is thus $m'_i = m_i + D(m_i)$.

#### 4.1.2 Sparse point matching

We also compute a set of matched points $\mathcal{E}$ between current images $I$ and $I'$ using the motion field. These points are chosen among the vertices of the mesh used in motion estimation. We select 200 vertices uniformly scattered in the image and which get highest scores for Harris and Stephens detector [15] in image $I$. More details about motion estimation can be found in [16, 17].

### 4.2. Camera parameters estimation

The parameters needed for reconstruction of the 3D model are internal and external camera parameters for each keyframe.

#### 4.2.1 Internal parameters

Internal camera parameters need not be accurate more than by an order of magnitude (this is one of the advantages of the representation with several models rather than one model). We thus arbitrarily choose parameters which are equal for all keyframes: we fix optical center $(u_0, v_0)$ at image center and we assume square pixels. An order of magnitude for the focal length may be provided by previous calibration or constructor data. If not, focal length is arbitrarily fixed to 500 (this is what is done for all the presented results). Another solution would be to estimate internal parameters through self-calibration techniques directly from the video sequence. However previous studies have shown that it is a highly unstable procedure with general acquisition conditions [18]. It is thus not adapted to automatic coding schemes.

#### 4.2.2 External parameters

External camera parameters $(R_k, t_k)$ are estimated from the set of matched points $\mathscr{E}_k$. The goal is to obtain a set of internal and external parameters which is consistent with $\mathscr{E}_k$, that is, the epipolar residual associated with fundamental matrix $F_k = A^{-T} \cdot [t_k]_\times \cdot R_k \cdot A^{-1}$ must be of sub-pixel value. For the sake of simplicity, index $k$ is omitted in the remaining of the section; all parameters implicitly refer to the current GOP between $K_k$ and $K_{k+1}$.

We first estimate fundamental matrix $F$ by minimizing the epipolar residual for all points in $\mathscr{E}$, using a classical median least squares algorithm [19]. The obtained fundamental matrix is denoted by $F_m$. The essential matrix $E_m$ is obtained from $F_m$ as $E_m = A^T \cdot F_m \cdot A$. A first set of camera parameters, denoted $R_c$ and $t_c$, are then computed from $E_m$ using a state-of-the-art decomposition method [20]. The parameter $R_c$ and $t_c$ minimize

$$||E_m - E_c||_f, \tag{4}$$

where $\|\cdot\|_f$ denotes the Frobenius distance and $E_c = R_c \cdot [t_c]_\times$. This estimation thus minimizes a matrix distance between estimated essential matrix $E_c$ and essential matrix related to matched points $E_m$. However, it does not ensure that the camera parameters are compatible with the set of matched points $\mathscr{E}$. Indeed the corresponding matrix $F_c = A^{-T} \cdot [t_c]_\times \cdot R_c \cdot A^{-1}$ is not similar to $F_m$ and the epipolar residual (3) associated with $F_c$ is large. In other words, $R_c$, $t_c$, and $F_c$ are not consistent with $\mathscr{E}$.

As explained before, the epipolar residual associated with $F_c$ also assesses the projection error for points in $\mathscr{E}$, when 3D reconstruction is performed using $A$, $R_c$, and $t_c$.

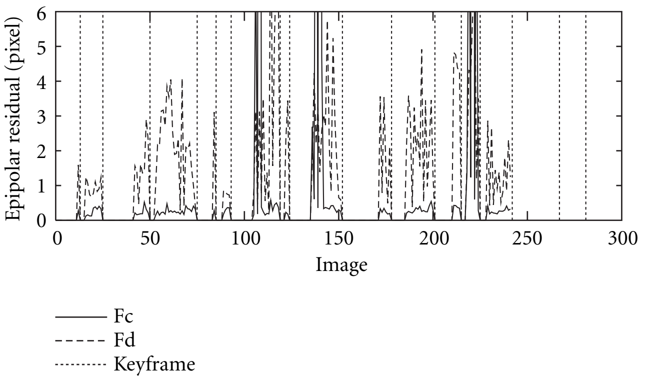

FIGURE 3: Epipolar residuals along the *stairway* video sequence after the calibration step (Fc) and after the localization step (Fd).

This is a very important criterion to take into account as we are looking for a 3D model and camera position which enable to re-create the original images between 2 keyframes. Moreover, the next step of sliding adjustment is sensitive to initialization, so a refinement step is performed on the $(R_c, t_c)$ parameters as follows.

We first compute the 3D position of each point in $\mathscr{E}$ and we then estimate the pose of this 3D points set with DeMenthon algorithm [21]. This technique computes the camera position $(R_d, t_d)$ which minimizes projection error for a given set of 3D points and their corresponding image points. We obtain a new pair $(R_d, t_d)$ which is compatible both with $A$ and $\mathscr{E}$. This is verified by computing the epipolar residual for $F_d = A^{-T} \cdot [t_d]_\times \cdot R_d \cdot A^{-1}$.

Figure 3 shows a comparative plot of the epipolar residuals associated with $F_c$ and $F_d$, as a function of frame number, for the *stairway* sequence. We can see that the proposed refinement greatly improves consistency of camera parameters with image data. Moreover it shows to be a more robust technique, as it often provides reasonable solution (epipolar residual smaller than 1 pixel) when the decomposition method provides an invalid solution (epipolar residual greater than 4 pixels).

At this point the estimated parameters $R_d$, $t_d$ are compatible with $\mathscr{E}$.

Once camera parameters are estimated, results are used to decide whether frame $I'$ should be used as keyframe image $K_{k+1}$. This is done by an automatic keyframe selection algorithm described in Section 5. If $I'$ is detected as an invalid $K_{k+1}$, a further frame is chosen as a candidate and motion estimation steps are started again for the same GOP. If $I'$ is a valid keyframe $K_{k+1}$, 3D model $\mathscr{M}_k$ associated with the current GOP is computed from dense motion field $D_k$ and camera parameters $R_k$ and $t_k$. To achieve global consistency of the 3D models, a sliding adjustment procedure is performed. It provides the 3D model scale and camera parameters $R_k$, $t_k$ which are consistent with the previous GOP. Sliding adjustment will be described in details in Section 6. The whole procedure is then started again for next GOP, with $I = K_{k+1}$ as first image in the GOP.

## 5. KEYFRAME SELECTION

We propose a simple method to select keyframes on the fly in order to ensure a valid 3D model reconstruction. The first selected keyframe $K_0$ is the first image of the video sequence $I_0$. The other keyframes are selected while processing and coding the sequence. Suppose keyframe $K_k$ has been selected and we are looking for the next keyframe $K_{k+1}$. The selection is made by the verification of 3 criteria estimated from the dense motion estimation and matched points.

### 5.1. Selection criteria

The following criteria are computed for each image $I'$ following $I = K_k$ in order to decide if $I'$ will be the next keyframe:

(C1) $\bar{D} > S_m$, where $\bar{D}$ is the average apparent motion and $S_m$ a static threshold fixed to 10 pixels,

(C2) the percentage of outgoing points is less than a threshold $S_o$, that is, no more than $S_o$ percent of points from $I$ in $\mathscr{E}$ are not present in $I'$, with $S_o$ of typical value 30%,

(C3) $\sum_i d^2(m_i, F_d \cdot m'_i) + d^2(m'_i, F_d^T \cdot m_i) < S_f$, where $F_d$ is the fundamental matrix computed from the estimated camera motion, $(m_i, m'_i) \in \mathscr{E}$ is the set of matched points between $I$ and $I'$, and $S_f$ is a static threshold typically fixed to 0.5 pixel.

The first criterion tends to favor a good precision for the depth field: the best precision on depth is clearly achieved with perpendicular lines of views. (C1) is a necessary criterion for a significant change in camera viewpoints, but is not sufficient, since (C1) can be defeated with a large rotation component.

The second criterion (C2) ensures that the two keyframes share a large part of the scene. This is necessary because the 3D model contains only points viewed in both images.

The third criterion (C3) ensures a valid 3D model reconstruction, by testing the epipolar residual. This criterion has two means. First it allows to detect ill-conditioned configurations. In such cases, due to numerical errors, the estimated 3D model, camera motion, and fundamental matrix are not consistent with the image data and motion field. The epipolar residual is then very large. The second means of (C3) is to ensure that the 3D model projects onto image $I'$ with sub-pixel error. The epipolar residual is the average 2D projection error for points in $\mathscr{E}$. Thus it evaluates the ability of the 3D model to accurately represent image $I'$.

These three criteria are used as follows: for a fixed image $I = I_n$, successive images $I_{n+1}, I_{n+2}, \ldots,$ are examined until (C1) and (C2) are verified. Following images are considered as candidates for $K_{k+1}$. For each candidate image $I'$, camera motion and fundamental matrix are estimated in order to evaluate (C3). We then select as $K_{k+1}$ the last candidate $I'$ before (C2) is false or before (C3) is not verified for more than 2 successive frames. One or two successive frames $I'$ with true (C1) and (C2) and false (C3) is considered due to unstable numeric estimation of $F$. This is the reason why the GOP is not ended before 3 or more successive images do not verify (C3).

### 5.2. Validation

This approach has been validated on several video sequences with various camera motions; we show the results on two typical cases. Figure 4 presents the evolution of the 3 criteria on the two test sequences: (C1) on top, (C2) in the middle, (C3) at the bottom. Horizontal lines show the threshold values for each criterion, and vertical dotted lines indicate keyframes. Left column refers to the *street* sequence, and right column refers to the *stairway* sequence (see Section 7.3 for images from the original video sequences).

Thresholds $S_o$ and $S_f$ are manually fixed close to typical values in order to obtain large GOPs. The following parameters were used to encode the *street* sequence: $S_m = 10$, $S_o = 40\%$, and $S_f = 0.35$. Epipolar residuals are computed only when average motion is greater than 10 pixels. We can notice that the 3 criteria allow to select keyframes mostly on outgoing points percentage, because this sequence have a quite stable motion.

The following parameters were used to encode the *stairway* sequence: $S_m = 10$, $S_o = 30\%$, and $S_f = 0.4$. This sequence is more unstable than the previous one, due to unstabilized camera motion during acquisition. However, we can notice that the three criteria allow to select keyframes despite the instability of epipolar geometry.

The GOP size varies according to video contents. For the *street* sequence, where camera motion is homogeneous, GOP size is quite stable, with a value around 40 frames in a GOP. For the *stairway* sequence, GOP size values between 5 to 30 frames with a typical value of 25 frames. GOPs are adapted to scene content and camera motion so that a single 3D model can accurately represent the frames in the GOP.

## 6. SLIDING ADJUSTMENT

At this point, we have a set of camera parameters which are independent, that is, a computed 3D model $\mathcal{M}_k$ whose geometry is optimal for the GOP $k$ between $K_k$ and $K_{k+1}$. Since we want local consistency between each successive 3D models, we use a sliding window to compute the camera positions in order to increase the consistency of a pair (camera, 3D model) with its neighbors.

### 6.1. Initialization

As the proposed sliding adjustment is solved using a nonlinear optimization procedure, initial values for the estimated parameters must be provided. These values should be close enough to the solution to allow the sliding adjustment to converge toward an acceptable solution. The camera position is first initialized with the previously computed $(R_d, t_d)$ parameters. Each 3D model has its own scale factor because camera translation and 3D model are defined up to a scale factor $\alpha$, as shown by the following equation:

$$\begin{aligned} &\forall M = (x, y, z) \in \mathcal{M}_k, \\ &\tilde{m} = A \cdot (R \mid t) \cdot (x, y, z, 1) \qquad (5) \\ \Longleftrightarrow\ &\tilde{m} = A \cdot (R \mid \alpha \cdot t) \cdot (\alpha \cdot x, \alpha \cdot y, \alpha \cdot z, 1). \end{aligned}$$

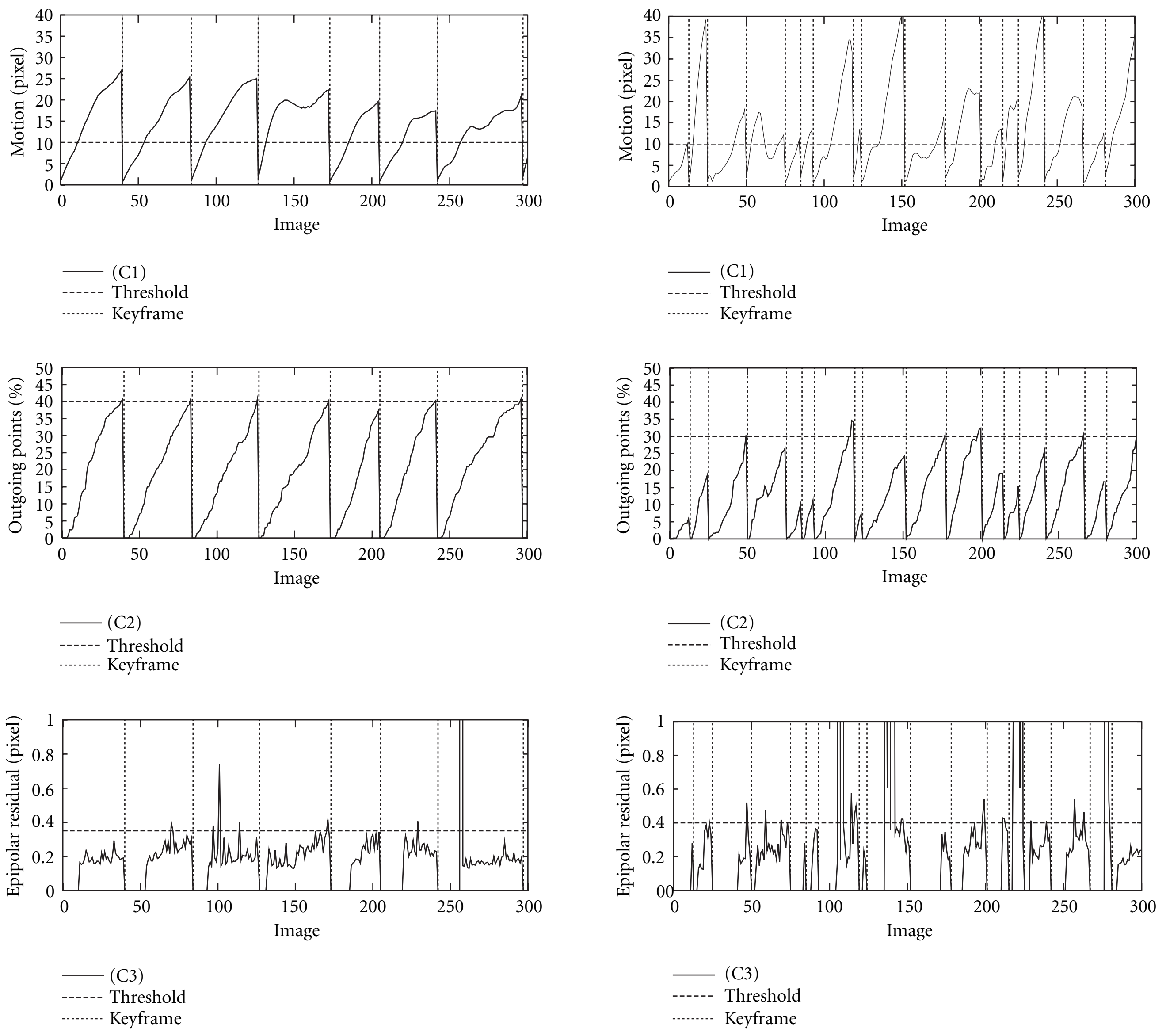

Figure 4: The evolution of the 3 criteria on the *street* video sequence (left) and *stairway* video sequence (right).

Consistent scales for successive models are also estimated by the sliding adjustment procedure. Initial values thus have to be provided as well. We describe here how the initial scale of each 3D model is set to be similar to its previous 3D model.

Scale of first 3D model $\mathcal{M}_0$ is not modified and is taken as a reference for the whole sequence. We compute its gravity center $G_0$ from the set of matched points $\mathcal{E}_0$ (points lying at infinity are not taken into account). Assuming that gravity center $G_k$ and scale factor $\alpha_k$ have been computed for model $\mathcal{M}_k$, the following steps are then iteratively performed:

(i) track points in $\mathcal{E}_k$ from keyframe $K_{k+1}$ to keyframe $K_{k+2}$,

(ii) compute $G'_{k+1}$, the gravity center of the 3D points reconstructed using these points,

(iii) compute scale factor for $\mathcal{M}_{k+1}$: $\alpha_{k+1} = |G_k \cdot R_k + t_k - O_{k+1}|/|G'_{k+1} - O_{k+1}|$,

(iv) rescale 3D model $\mathcal{M}_{k+1}$ and associated matrices $P_{k+1}$, $P_{k+2}$ using $\alpha_{k+1}$,

(v) compute new gravity center $G_{k+1}$ from the set of points $\mathcal{E}_{k+1}$.

At the end of this process, we obtain a set of 3D models $\mathcal{M}_k$ which are defined in the same basis and which have a consistent scale from one 3D model to the next 3D model in the stream.

### 6.2. Algorithm

Our algorithm is based on bundle adjustment [22] but it is adapted to our application, namely

(i) each set $\mathcal{E}_k$ generates a 3D model $\mathcal{M}_k$ whereas in bundle adjustment all 3D points are merged into one single model,
(ii) local consistency is performed on a sliding temporal window: for a given keyframe $K_k$, only neighboring images and camera positions are taken into account,
(iii) some 3D points are constrained to stay on their view-line for coding purpose.

We now define some useful notation.

(1) $P_k = A \cdot (R_k \mid t_k)$ is the projection matrix which perfectly projects the 3D model $\mathcal{M}_k$ on image $K_k$. This matrix is also the projection matrix for the last image of previous GOP: $P_k$ projects the 3D model $\mathcal{M}_{k-1}$ on $K_k$ with an error due to the imperfections of 3D model $\mathcal{M}_{k-1}$.

(2) $\{M_i^k\}$ is a set of 3D points computed with projection matrices $P_k$ and $P_{k+1}$, from the set of robust points $\mathcal{E}_k$ extracted in image $K_k$ and matched in image $K_{k+1}$. $\{M_i^k\}$ can be seen as a subset of 3D model $\mathcal{M}_k$.

(3) $m_i^k$ is a point in $\mathcal{E}_k$ extracted in keyframe $K_k$.

(4) $m_i^{k,l}$ is a point extracted in keyframe $K_k$ and tracked till keyframe $K_l$ by summation of estimated motion fields.

(5) $\hat{m}_i^{k,l} = P_l \cdot M_i^k$ is a point of $\{M_i^k\}$ projected in image $K_l$ with projection matrix $P_l$.

For each keyframe $K_k$, a cost function $f = f_1 + f_2 + f_3$ is minimized. The considered costs are 2D residual errors when projecting the 3D models onto keyframes inside the sliding window. When processing current keyframe $K_k$, the following parameters have been estimated at previous iteration on $K_{k-1}$: point sets $\{M_i^{k-1}\}$, $\{M_i^k\}$, projection matrices $P_{k-1}$, $P_k$, and $P_{k+1}$. Among them $\{M_i^{k-1}\}$, $P_{k-1}$, and $P_k$ are final values, whereas the values of $\{M_i^k\}$ and $P_{k+1}$ are estimated again in the current $K_k$ process.

We now describe each term in the cost function and we give its geometrical interpretation.

(1) The first term $f_1$ ensures that model $\mathcal{M}_k$ projects correctly onto keyframe $K_{k+1}$, by finding the best projection matrix $P_{k+1}$. $P_k$ is expected to perfectly project $\mathcal{M}_k$ onto $K_k$, thus point $M_i^k$ must not be modified on image $K_k$. A 3D point $M_i^k$ thus has only one degree of freedom: moving along its view-line. This constraint writes

$$M_i^k = O_k + \lambda_i^k \cdot \vec{u}\left(m_i^k\right). \tag{6}$$

Under this constraint, the cost function $f_1$ is defined as

$$\begin{aligned} f_1\left(P_{k+1}, \left\{M_i^k\right\}\right) &= \sum_i \left\|m_i^{k,k+1} - \hat{m}_i^{k,k+1}\right\|^2 \\ &= \sum_i \left\|m_i^{k,k+1} - P_{k+1} \cdot M_i^k\right\|^2, \end{aligned} \tag{7}$$

$f_1$ is a function of both the matrix $P_{k+1}$ and the point set $\{M_i^k\}$. The unknown parameters in $f_1$ are $\{\lambda_i^k\}$ which define $\{M_i^k\}$ and $(R_{k+1}, t_{k+1})$ which define $P_{k+1}$.

(2) $f_2$ ensures consistency of keyframe $K_{k+1}$ with 3D model $\mathcal{M}_{k-1}$. The set of points $\{M_i^{k-1}\}$ has been computed on a previous step as well as projection matrices $P_{k-1}$ and $P_k$, and they are thus fixed parameters. We search the best projection matrix $P_{k+1}$ for both $\mathcal{M}_{k-1}$ and $\mathcal{M}_k$, that is, we add a new cost function

$$\begin{aligned} f_2(P_{k+1}) &= \sum_i \left\|m_i^{k-1,k+1} - \hat{m}_i^{k-1,k+1}\right\|^2 \\ &= \sum_i \left\|m_i^{k-1,k+1} - P_{k+1} \cdot M_i^{k-1}\right\|^2. \end{aligned} \tag{8}$$

Minimizing the cost function $f_1 + f_2$ ensures that 3D model $\mathcal{M}_k$ is consistent with the previous model $\mathcal{M}_{k-1}$.

(3) Finally we want consistency of 3D model $\mathcal{M}_k$ with next 3D model $\mathcal{M}_{k+1}$. This is done by estimating $P_{k+2}$ which best projects points $\{M_i^k\}$ and $\{M_i^{k+1}\}$ on image $K_{k+2}$, under the constraint that $\{M_i^{k+1}\}$ project perfectly on $K_{k+1}$. This is ensured by minimizing the cost function $f_3$

$$\begin{aligned} & f_3\left(P_{k+2}, \left\{M_i^k\right\}, \left\{M_i^{k+1}\right\}\right) \\ &\quad = \sum_i \left\|m_i^{k,k+2} - \hat{m}_i^{k,k+2}\right\|^2 + \sum_i \left\|m_i^{k+1,k+2} - \hat{m}_i^{k+1,k+2}\right\|^2 \\ &\quad = \sum_i \left\|m_i^{k,k+2} - P_{k+2} \cdot M_i^k\right\|^2 + \sum_i \left\|m_i^{k+1,k+2} - P_{k+2} \cdot M_i^{k+1}\right\|^2 \end{aligned} \tag{9}$$

under the constraint

$$M_i^{k+1} = O_{k+1} + \lambda_i^{k+1} \cdot \vec{u}\left(m_i^{k+1}\right). \tag{10}$$

The final cost function becomes

$$\begin{aligned} & f\left(P_{k+1}, \left\{M_i^k\right\}, P_{k+2}, \left\{M_i^{k+1}\right\}\right) \\ &\quad = f_1\left(P_{k+1}, \left\{M_i^k\right\}\right) + f_2(P_{k+1}) \\ &\quad\quad + f_3\left(P_{k+2}, \left\{M_i^k\right\}, \left\{M_i^{k+1}\right\}\right). \end{aligned} \tag{11}$$

Equation (11) is a large nonlinear system with the following characteristics:

(i) 6 unknown parameters for the projection matrix $P_{k+1}$: 3 for translation $t_{k+1}$, and 3 for rotation $R_{k+1}$,
(ii) 6 unknown parameters for projection matrix $P_{k+2}$,
(iii) Card($\{M_i^k\}$) unknown parameters for the set of points $\{M_i^k\}$,
(iv) Card($\{M_i^{k+1}\}$) unknown parameters for the set of points $\{M_i^{k+1}\}$,
(v) $2 \cdot$ Card($\{M_i^k\}$) equations for the constraint $f_1$ (one equation on the $x$-axis and another on the $y$-axis),
(vi) $2 \cdot$ Card($\{M_i^{k-1}\}$) equations for the constraint $f_2$,
(vii) $2 \cdot$ Card($\{M_i^k\}$) $+ 2 \cdot$ Card($\{M_i^{k+1}\}$) equations for the constraint $f_3$.

One must notice that this large system is a very sparse system: a given 3D point $M_i$ interferes in 4 or less equations. This system is then solved using a classical nonlinear estimation algorithm which deals with large sparse systems. We have used the MinPack [23] package implementation. At the end of the minimization step $k$, the final 3D model $\mathcal{M}_k$ is

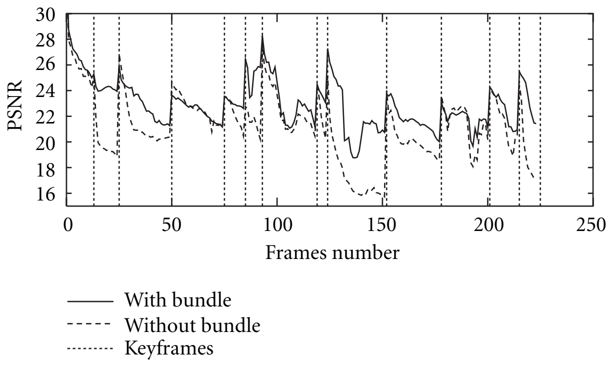

Figure 5: PSNR value for reconstruction of *stairway* video sequence. Each GOP $k$ is reconstructed with the previous corresponding 3D model $\mathcal{M}_{k-1}$. Method with sliding bundle adjustment have a better quality, showing a better consistency of previous 3D model with the current GOP.

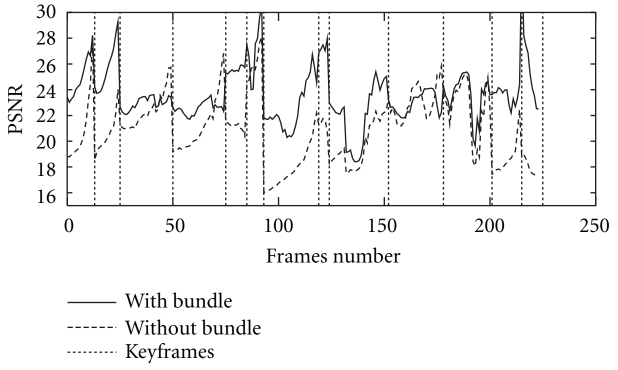

Figure 6: PSNR value for reconstruction of *stairway* video sequence. Each GOP $k$ is reconstructed with the next corresponding 3D model $\mathcal{M}_{k+1}$. Method with sliding bundle adjustment have a better quality, showing a better consistency of next 3D model with the current GOP.

computed using the 2 projection matrices $P_k$ and $P_{k+1}$, and projection matrix $P_{k+2}$ is taken as an initial value for the next step $k+1$.

Figure 5 presents PSNR value obtained with the *stairway* video sequence reconstructed using previous 3D model $\mathcal{M}_{k-1}$ instead of $\mathcal{M}_k$ for each GOP $k$. Figure 6 shows the PSNR obtained when using next 3D model. They show that the sliding adjustment increases the ability of next and previous 3D models to represent the current GOP. The expected consistency of each 3D model with previous and next 3D models is thus achieved.

### 6.3. Extended sliding adjustment

The presented method can be extended to take into account more than 3 successive models. This is done by adding cost functions similar to $f_3$. Consider that we want to increase consistency with the $p$ next keyframes (in the previous section $p = 1$). We generalize function $f_3$ into $g_n$, which gives the contribution of keyframe $K_{k+n}$ into the total cost function

$$g_n\left(P_{k+n+1}, \left\{M_i^k\right\}, \dots, \left\{M_i^{k+n}\right\}\right) = \sum_{q=0}^{n} \sum_i \left\| m_i^{k+q,k+n+1} - P_{k+n+1} \cdot M_i^{k+q} \right\|^2 \quad (12)$$

under the constraints

$$M_i^{k+q} = O_{k+q} + \lambda_i^{k+q} \cdot \vec{u}\left(m_i^{k+q}\right), \quad q = 0, n. \quad (13)$$

For $n = 1$, $g_1$ is the contribution of $K_{k+2}$ and is equal to $f_3$. The final cost function which takes into account $p$ keyframes and 3D models is

$$\begin{aligned} g&\left(P_{k+1}, \left\{M_i^k\right\}, P_{k+2}, \left\{M_i^{k+1}\right\}, \dots, P_{k+p+1}, \left\{M_i^{k+p}\right\}\right) \\ &= f_1\left(P_{k+1}, \left\{M_i^k\right\}\right) + f_2\left(P_{k+1}\right) \\ &\quad + \sum_{n=1}^{p} g_n\left(P_{k+n+1}, \left\{M_i^k\right\}, \dots, \left\{M_i^{k+n}\right\}\right). \end{aligned} \quad (14)$$

The extension to $p$ 3D models ensures consistency of the 3D models for a longer time in the video. However, without a precise set of camera parameters, a larger window may degrade the estimation results.

## 7. RESULTS

### 7.1. Textured 3D model generation

We just present the final step for 3D models generation. The visualization step is also adapted to our representation with local 3D models.

For each pair of successive keyframe images $K_k$ and $K_{k+1}$ a 3D model $\mathcal{M}_k$ is computed. Since we have a dense motion field $D_k$, camera internal parameters $A$ and camera motion parameters $R_k$, $t_k$ and projection matrices $P_k$, $P_{k+1}$ from sliding adjustment, this is fairly simple. For any pixel $m$ in $K_k$, its corresponding position is given by $m' = m + D_k(m)$, and the 3D point is recovered by solving projection equations (1). A dense depth map is then constructed. In order to have a 3D model which can be easily visualized, only vertices of a regular 2D triangular mesh are reconstructed. The reconstructed points define a continuous 3D triangular mesh. This mesh is textured using image $K_k$. This 3D model is simply described in a format Rec3D quite similar to VRML format. A Rec3D file is then generated, which can be interactively visualized in real time with classical 3D rendering libraries like openGL [24].

### 7.2. Adapted visualization through 3D model fading

The proposed representation with the set of 3D models $\{\mathcal{M}_k\}$ can reconstruct the input video sequence. However, some artifacts appear in the reconstructed video sequence, in particular when we switch from one 3D model to another. They are mostly due to illumination changes between 2 keyframes, occluded areas and to the accuracy of the 3D model to reconstruct the GOP. In order to take into account such problems, we propose an original 3D model fading technique. This technique is an approximation of real 3D model morphing. It

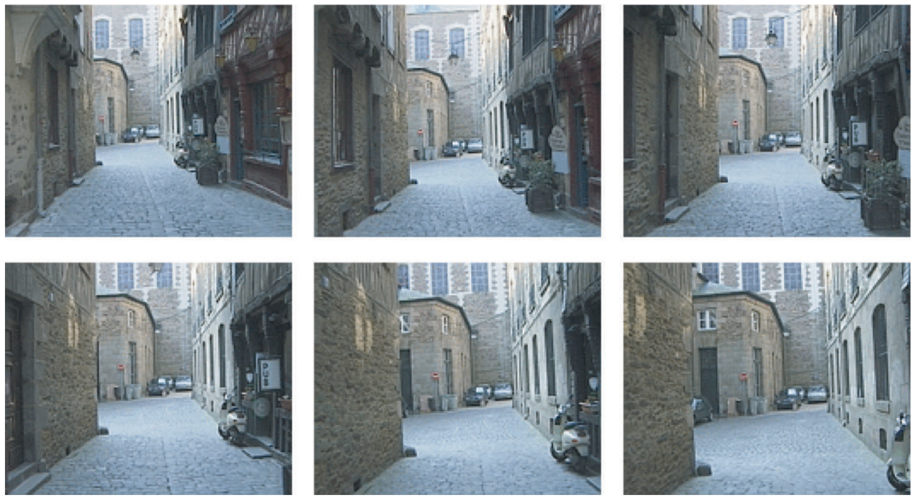

Figure 7: Original *street* sequence. From top-left to bottom-right, image 0, 40, 80, 120, 160, 200.

is a simple way to take into account not only texture changes but also the geometric changes from one 3D model to the next in the stream. A two-passes rendering is performed, a first pass using the current 3D model, and a second using the next 3D model. The resulting reconstructed sequences are then blended with respect to the $\alpha$ factor defined by

$$\alpha(C) = |O_k - C| / |O_k - O_{k+1}|, \tag{15}$$

where $O_k$ and $O_{k+1}$ are camera centers for first and last image of the GOP and $C$ is the current camera position. Thus the $\alpha$ factor balances the reconstructed sequences contribution from first to last image of the GOP, in proportion with the distance of the current camera position from the keyframes camera positions.

### 7.3. Results on real video sequences

The proposed algorithm has been implemented and tested on several real video sequences. Reconstruction of the original video sequence with the 3D models stream is easily done, but is not discussed here since it is similar to classical image interpolation. So, we test the accuracy of the 3D models with classical applications of 3D model manipulation: free navigation, illumination changes, augmented reality, stereo visualization. We perform visual quality estimation. The corresponding video sequences can be found at http://www.irisa.fr/temics/Demos/3D4 showing some applications of the representation (compression aspects are also shown but not discussed here).

The first is a *street* video sequence which is a walk in a city with a global translation along the $z$-axis (Figure 7). Sequence has been acquired with a mechanical stabilizer, internal parameters are unknown and fixed (see Section 4.2), the focal distance is approximated to 500. The dense motion field between image 0 (keyframe 0) and image 40 (keyframe 1) is shown on Figure 8a. The vector scale is 0.25 and the average

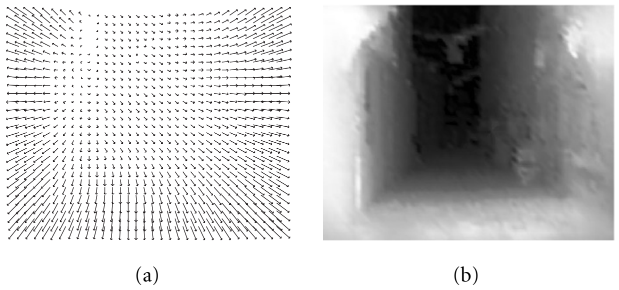

Figure 8: (a) Dense motion field between image 0 and 40 in *street* sequence. (b) Depth field extract from dense motion field.

motion is 41 pixels. The corresponding depth map on image 0 is shown on Figure 8b (the furthest areas from the camera are in dark). We see that the depth map is quite regularized and reconstructed the global shape of the street with details on the relief (pot plant, street lamp, etc.). The corridor shape of the street is more visible on Figure 9, where we clearly see the planarity of the floor and the right angle with the 2 walls on the sides with the floor.

The second test sequence is the *stairway* sequence which is a walk with a global translation along the $x$-axis (Figure 10). The video sequence has been acquired with a simple hand-held camcorder, internal parameters are unknown and fixed, the focal distance is again approximated to 500. This video sequence is quite harder than the previous one due to uniform textures and water in the fountain which confuses the motion estimation. Figures 11a and 11b show motion field and depth map for *stairway* video sequence between image 71 and image 83: vector scale is 0.25 and the average motion is 40.2 pixels. We see that we obtain a valid scene geometry: we find the trees and the shape of the stairway. Figure 12 shows in details the geometry of the scene as depth maps.

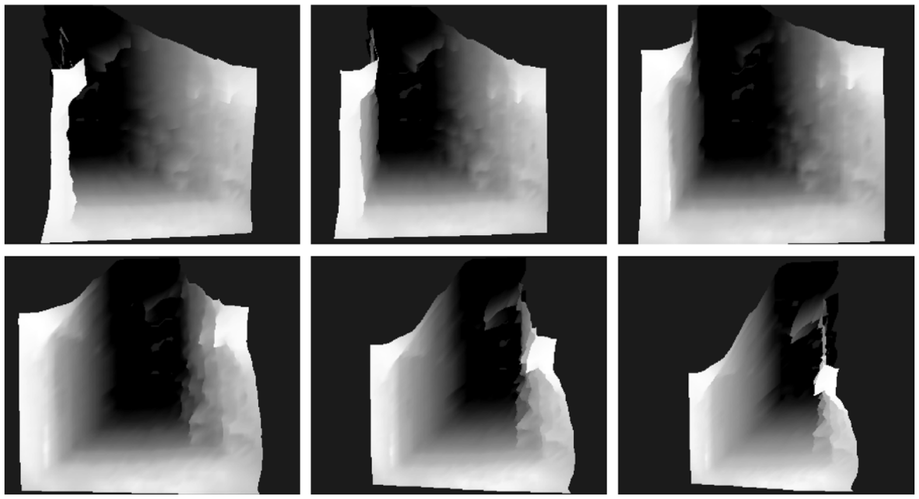

Figure 9: Rotation around a 3D model automatically extracted from *street* sequence. The images are the depth map of each views.

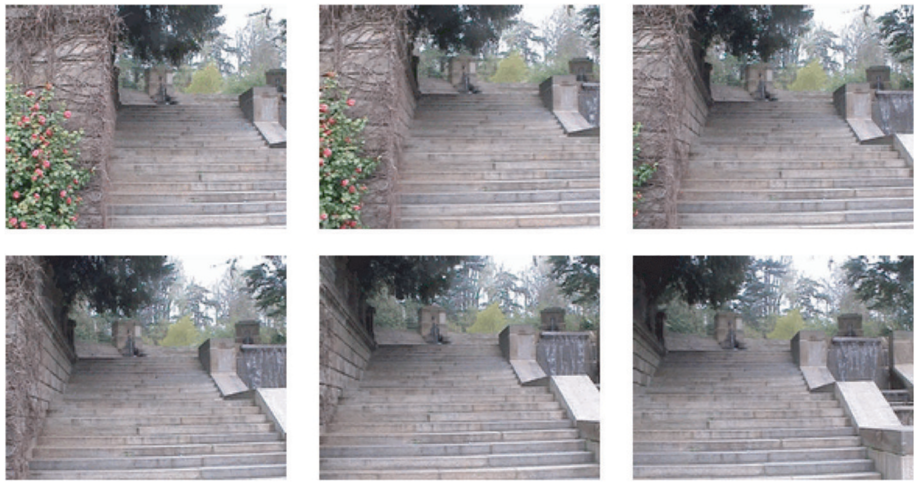

Figure 10: Original *stairway* sequence. From top-left to bottom-right, image 50, 75, 100, 125, 150, 175.

### 7.3.1 *Free navigation*

Free navigation in the representation is performed by just specifying new camera positions. Figure 13 shows a texture image (used for the texture mapping) and a virtual view generated with this image and the corresponding 3D model: we simulate a virtual walker which performs few steps on the left and turn the head to the right. We see that the window and the gate in the background are occulted by the left wall, according to the scene geometry. Figure 14 shows 2 other virtual views taken far away from original viewpoints: (a) shows the image produces with a large rotation to the left and (b) a view in details of the scooter. Texture stretching is visible on surfaces which are not visible with a frontal view or which are in occulted areas.

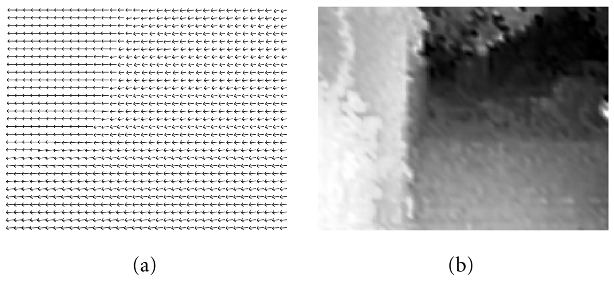

Figure 11: (a) Dense motion field between image 71 and 83 in *street* sequence. (b) Depth field extract from dense motion field.

### 7.3.2 *Lightning modification*

Lightning modification is performed with classical illumination algorithm (see [24] for details). Contrary to global illumination changes of a video sequence, the illumination takes into account 3D information such as distance from the light to the surfaces, giving a better realism. Lightning modification on the street video sequence are presented on

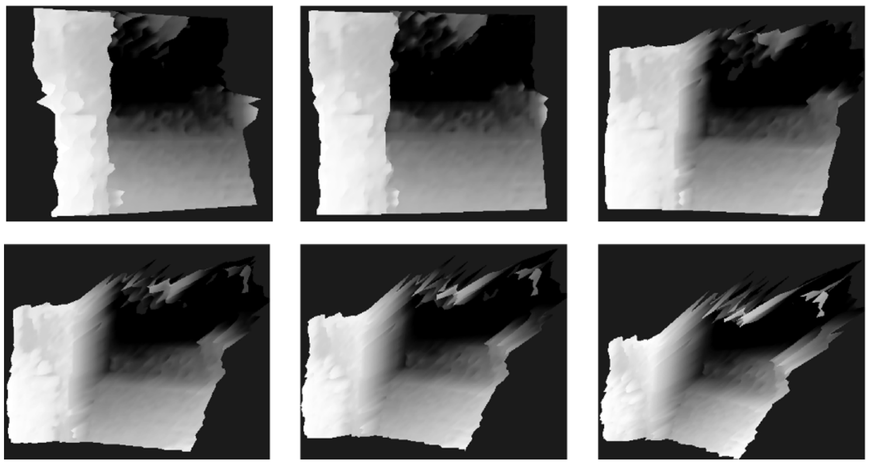

FIGURE 12: Rotation around a 3D model automatically extracted from *stairway* sequence. The images are the depth map of each views.

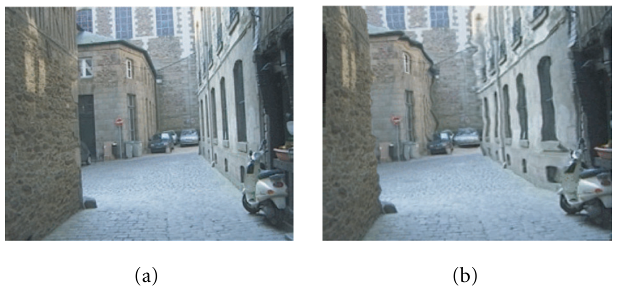

FIGURE 13: (a) Texture image used with 3D model 4 extracted from the *street* video sequence. (b) A virtual view generated with this texture image: the virtual walker performs few steps on the left and turn the head to the right.

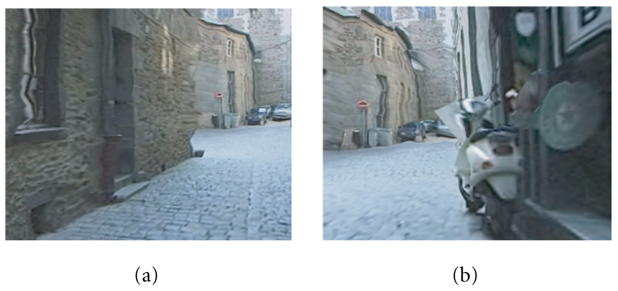

FIGURE 14: Two virtual views of *street* video sequence generated far away from original viewpoints.

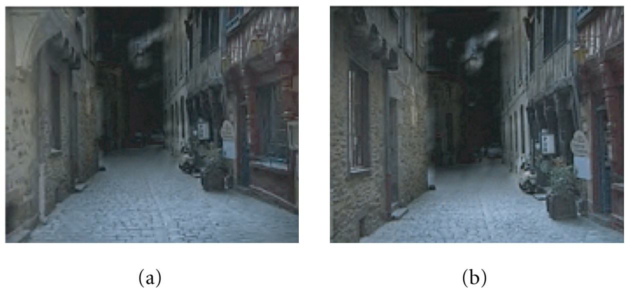

FIGURE 15: Image 0 and 40 from the *street* sequence: a headlight is added to reconstruct the original video sequence.

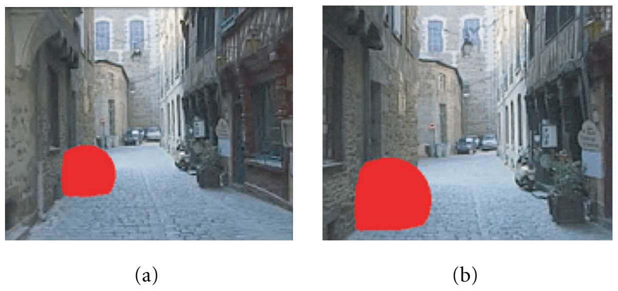

FIGURE 16: A virtual sphere is added: (a) view with extracted 3D model 1, (b) view with extracted 3D model 2. The occlusions are taken into account, and the sphere position is quite stable from one 3D model to the next one.

Figure 15: a headlight have been added to change illumination in the scene. The figure shows 2 images of the reconstructed video sequence. We notice the illumination is consistent with the geometry, that is, decreasing with the distance from the light to the surface. We also notice that illumination is invariant from one 3D model (a) to another (b) and some artifacts on the top street lamp due to 3D mesh continuity.

### *7.3.3 Object insertion*

Insertion of virtual objects in the video sequence taking into account depth and occlusions is easy with the representation, contrary to classical 2D object insertion. Figures 16 and 17 show a virtual sphere added in the sequence *street* and *stairway*. The sphere is placed to intersect the scene, showing the occlusions accuracy. We can also notice the good stability of the sphere's position along the video sequence. Moving

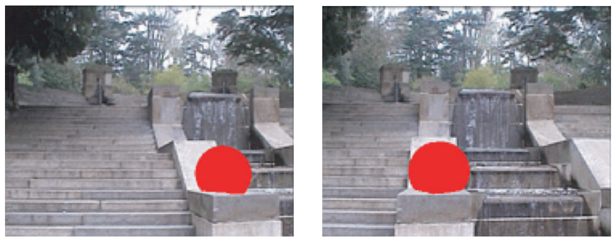
(a) (b)

Figure 17: A virtual sphere is added in the scene. (a) View with extracted 3D model 10. (b) View with extracted 3D model 15. The occlusions are taken into account, and the sphere position is stable from one 3D model to the next one.

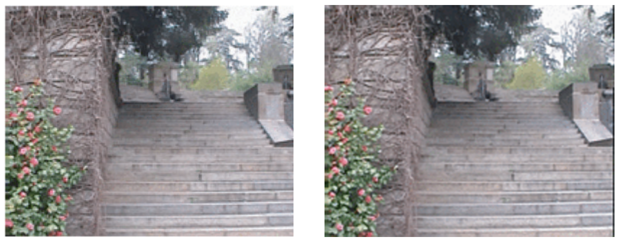
(a) (b)

Figure 18: Left (a) and right (b) views of an image extracted from the reconstructed stereoscopic video sequence.

objects can also be inserted, taking into account depth information given by the representation.

### *7.3.4 Stereo sequence generation*

The generation of stereoscopic video sequences just requires to reconstructed the scene twice (for the left and the right eyes) with a small shift. Figure 18 shows such a pair of images extracted from the *stairway* video sequence. The video sequence is visualized on specific device or with eyes' defocus technique for still images. Stereo visualization has been successfully tested on stereo display.

## 8. CONCLUSION

We have proposed an automatic scheme to extract a stream of 3D models from a video sequence of a fixed scene. The presented scheme offers a good compromise for 3D reconstruction from any video sequence of static scenes when the reconstruction of a unique 3D model is not possible or desirable: this is the case for very long video sequences (computation complexity) which require on-the-fly analysis, or when camera motion is not appropriate for a unique 3D reconstruction (forward translation).

We have proposed a simple technique for automatic video sequence clustering which allows to reconstruct several 3D models. These 3D models are then computed using a sliding adjustment which allows to keep most of the functionalities of a unique 3D model. We have validated our approach on several video sequences.

Such approaches could be extended to any video sequences (not only static scene) where objects segmentation are known. Moreover, as in MPEG4 Sprite coding, very low bitrate coding might be achieved with such a representation. We thus plan to study the coding performance of our approach compared to standard video coders.

## ACKNOWLEDGMENTS

We thank Stéphane Pateux for the fruitful discussions and his cooperation regarding the use of his algorithms. This work was supported by the RNRT project V2NET.

## REFERENCES

[1] S. E. Chen, "Quicktime VR—an image-based approach to virtual environment navigation," *Computer Graphics*, vol. 29, pp. 29–38, 1995.

[2] MPEG-4 standard ISO/IEC JTC1/SC29/WG11, "Generic coding of audio-visual objects: Visual," Information Technology N2802, ISO/IEC, July 1999.

[3] T. S. Huang and L. Tang, "Very low bit-rate video compression using 3D models," in *Proc. IWSNHC3DI*, Thira, Greece, 1995.

[4] B. Girod, P. Eisert, M. Magnor, E. Steinbach, and T. Wiegand, "3-d image models and compression: Synthetic hybrid or natural fit?," in *Proc. IEEE International Conference on Image Processing*, vol. 2, pp. 525–529, Kobe, Japan, October 1999.

[5] F. Prêteux and M. Malciu, "Model-based head tracking and 3D pose estimation," in *Visual Conference on Image Processing*, pp. 94–110, San Jose, Calif, USA, 1998.

[6] ISO/IEC JTC1/SC29/WG11, "Mpeg-4 animation framework extension (afx) vm 3.0," Tech. Rep. N4020, ISO/IEC, March 2001.

[7] H.-Y. Shum and R. Szeliski, "Stereo reconstruction from multiperspective panoramas," in *Proc. IEEE 7th International Conference on Computer Vision*, vol. 1, pp. 14–21, Kerkyra, Corfu, Greece, September 1999.

[8] M. Pollefeys, R. Koch, M. Vergauwen, B. Deknuydt, and L. Van Gool, "Three-dimensional scene reconstruction from images," in *Proc. SPIE Electronic Imaging, Three-Dimensional Image Capture and Applications III*, vol. 3958 of *Proceedings of SPIE*, pp. 215–226, San Jose, Calif, USA, 2000.

[9] D. Nistèr, "Reconstruction from uncalibrated sequences with a hierarchy of trifocal tensors," in *Proc. European Conference on Computer Vision*, vol. 1, pp. 649–663, Dublin, Ireland, 2000.

[10] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen, "The lumigraph," in *Proc. SIGGRAPH*, pp. 43–54, New Orleans, La, USA, August 1996.

[11] R. Kochand, M. Pollefeys, and L. Van Gool, "Realistic 3D scene modeling from uncalibrated image sequences," in *Proc. IEEE International Conference on Image Processing*, Kobe Japan, October 1999, Invited contribution to special session on Image Analysis and Synthesis.

[12] J.-R. Ohm and K. Müller, "Incomplete 3D-multiview representation of video objects," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 9, no. 2, pp. 389–400, 1999.

[13] A. W. Fitzgibbon and A. Zisserman, "Automatic camera recovery for closed or open image sequences," in *Proc. European Conference on Computer Vision*, pp. 311–326, Freiburg, Germany, June 1998.

[14] G. Marquant and S. Pateux, "Mesh and "crack lines": Application to object-based motion estimation and higher scalability," in *Proc. IEEE International Conference on Image Processing*, vol. 2, pp. 554–557, Vancouver, Canada, September 2000.
[15] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proc. 4th Alvey Vision Conference*, pp. 147–151, Manchester, UK, August 1988.
[16] F. Galpin and L. Morin, "Computed 3D models for very low bitrate video coding," in *Proc. SPIE Conference on Visual Communications and Image Processing*, vol. 4310 of *SPIE Proceedings*, San Jose, Calif, USA, January 2001.
[17] F. Galpin and L. Morin, "Video coding using streamed 3D representation," *Calculateurs parallèles, Réseaux et systèmes répartis, numéro spécial: Image et vidéo*, vol. 12, no. 3-4, pp. 431–442, 2000.
[18] S. Bougnoux, "From projective to Euclidean space under any practical situation, a criticism of self-calibration," in *Proc. IEEE International Conference on Computer Vision*, pp. 790–796, Bombay, India, January 1998.
[19] P. H. S. Torr and D. W. Murray, "The development and comparison of robust methods for estimating the fundamental matrix," *International Journal of Computer Vision*, vol. 24, no. 3, pp. 271–300, 1997.
[20] O. D. Faugeras, Q. T. Luong, and S. J. Maybank, "Camera self calibration: Theory and experiments," in *Proc. European Conference on Computer Vision*, pp. 321–334, Santa Margherita Ligure, Italy, June 1992.
[21] D. F. DeMenthon and L. S. Davis, "Model-based object pose in 25 lines of code," *International Journal of Computer Vision*, vol. 15, no. 1-2, pp. 123–141, 1995.
[22] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon, "Bundle adjustment—a modern synthesis," in *Proc. ICCV Vision Algorithms Workshop*, Corfu, Greece, October 1999.
[23] J. Moré, B. Garbow, K. Hillstrom, and M. Report, "User guide for MinPack-1," Tech. Rep. ANL-80-74, Argonne National Laboratory, Argonne, Ill, USA, August 1980.
[24] J. Neider, T. Davis, and M. Woo, *OpenGL Programming Guide*, Addison-Wesley, Reading, Mass, USA, 1993.

**Franck Galpin** was born in Laval, France. He received his M.S. degree in 1998 from DIIC/IFSIC school of the University of Rennes 1, followed by a Ph.D. thesis in computer sciences at IRISA/INRIA Rennes laboratory in the Temics project. His topics of interest deal with video coding and computer vision. He is now a post-doctoral student in Deguchi Laboratory at the University of Tohoku in Japan.

**Luce Morin** was born in Grenoble, France in 1966. She received her M.S. degree in 1989 from ENSPS school in Strasbourg, followed by a 6 month end-of-study internship at the NASA GSFC in Washington D.C. She then prepared a Ph.D. thesis with professor Roger Mohr, at the LIFIA laboratory, in INP-Grenoble. Her subject was dealing with projective invariants applied to computer vision. Since 1993 she has been an Assistant Professor at University of Rennes 1, where she teaches computer science, image processing, and computer vision. She is a member of the Temics project, in the IRISA/INRIA Rennes laboratory. Her research activities deal with 3D modelisation for video sequence communication.