Multi-View Video Representation Based on Fast Monte Carlo Surface Reconstruction

Jordi Salvador, *Member, IEEE,* and Josep R. Casas, *Member, IEEE*

***Abstract*—This article provides an alternative solution for the costly representation of multi-view video data, which can be used for both rendering and scene analysis. First, a new, efficient Monte Carlo discrete surface reconstruction method for foreground objects with static background is presented, which outperforms volumetric techniques and is suitable for GPU environments. Some extensions are also presented, which allow speeding up the reconstruction by exploiting multi-resolution and temporal correlation. Then, a fast meshing algorithm is applied, which allows interpolating a continuous surface from the discrete reconstructed points. As shown by the experimental results, the original video frames can be approximated with high accuracy by projecting the reconstructed foreground objects onto the original viewpoints. Furthermore, the reconstructed scene can be easily projected onto any desired virtual viewpoint, simplifying thus the design of Free-Viewpoint Video applications. In our experimental results, we show that our techniques for reconstruction and meshing compare favorably to the state-of-the-art, and we also introduce a rule-of-thumb for effective application of the method with a good quality vs. representation cost trade-off.**

***Index Terms*—Free-viewpoint, Monte Carlo, Shape-from-silhouette, Surface reconstruction, Meshing**

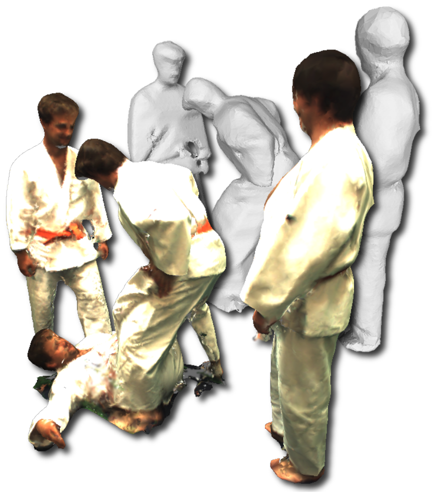

Fig. 1. Surface geometry and colored surface obtained from 16 views by using the proposed method. Both are rendered as viewed from a novel viewpoint. Multi-view data downloaded from [8].

I. Introduction

Multi-view video is a major focus of research in the image processing and computer vision communities. In video analysis applications, the availability of multi-view data allows the understanding of scenes which would be ambiguous if observed by a single camera (due to overlaps in monocular views). Another advantage is the possibility of offering *free-viewpoint video* (FVV). This application aims at rendering photo-realistic views of a dynamic scene captured by a multi-camera rig as viewed from virtual viewpoints. Image-based approaches [1], [2] rely solely on the original video streams in order to render the scene from a new viewpoint. These approaches provide high-quality renderings interpolated between existing viewpoints (the ones captured by the multi-camera rig). However, when a free choice of the viewpoint is desired, either depth maps [3] or 3D geometry models of the objects of interest in the scene [4] must be computed. This paper proposes a surface representation and the corresponding reconstruction method that can be used as an alternative to the original multi-view video data [5], [6], [7] from which it is obtained and can be a useful representation for both multi-view analysis and FVV. In [5], the visual appearance of a multi-view scene is stored in multi-channel images. Whereas this approach allows reusing standard video coding tools, the 3D structure of the scene, useful for analysis, is not explicit. Similarly, in [6], [7] the current goals of the *3D Video* project (3DV) are shown to support advanced stereo display processing and improved support for rendering of multi-views with small baselines, targeting auto-stereoscopic displays, without providing a mechanism for explicitly describing the 3D structure.

The projection process that takes place in a camera in order to generate a 2D image out of the 3D scene is an inherently lossy process, given that most of the 3D information about the scene is lost. Cameras just capture the projections of the 3D surfaces (the visible part of 3D objects). Multi-view data provides cues that can be used to recover part of this 3D structure through reconstruction. Naturally, a large number of views will be preferred, as they provide more information about the scene. An efficient 3D reconstruction method should aim at directly obtaining the position, appearance and orientation of *surfaces* (in contrast with volumetric approaches), since these are the only features of the real scene for which observations are available. For a target resolution, it is also more efficient to obtain a surface description than a volumetric one: surfaces in 3D space are sparse, as opposed to volumes. For an object in 3D space with volume $\propto r^3$, the corresponding enclosing surface has an area $\propto r^2$. Thus, the support of a closed surface in 3D space is more compact than that of its enclosed volume.

J. Salvador is with the Image Processing Lab, Technicolor R&I, Hannover. The work leading to this article was developed while in the Image and Video Processing Group, Universitat Politècnica de Catalunya, Barcelona.

J. R. Casas is with the Image and Video Processing Group, Universitat Politècnica de Catalunya, Barcelona.

Manuscript received October 15, 2012; revised May 21, 2013.

Following this principle, surface reconstruction algorithms of different levels of complexity have appeared in the last years, which dedicate the computational resources to obtaining and representing surfaces. Some of these algorithms provide continuous surfaces (polygon meshes) [9], whereas some others provide discrete ones (represented by surface samples or *surfels*), either as final [10] or intermediate representations [11]. The latter can also be used to obtain continuous representations through the use of robust [12] or speed-oriented [13] meshing algorithms. Among other possible applications exploiting the reconstructed 3D structure from multi-view data, [14] proposes a probabilistic framework for tracking reconstructed surfaces in presence of reconstruction artifacts, relying solely on the available 3D surfaces.

In [15], an impressive system for joint segmentation and reconstruction, robust to calibration errors, is presented. In this method, oriented to offline processing (considering the high computational load of the different involved stages), geometry is extracted using [12]. This leads to sparse sampling in regions with smooth geometry and dense sampling in more complex regions. Although such a solution is efficient in terms of geometry representation, it implies texture (the complexity of which is in general independent of that of the geometry) cannot be properly represented as an attribute of the reconstructed geometry. In consequence, the original image data still needs to be supplied for view-dependent texture mapping.

With real-time applications in mind, it would be desirable to design massively parallel algorithms tuned to the computing capabilities found in *Graphics Processing Units* (GPUs). [16] presents a multi-view stereo approach for small-baseline scenarios (with tight limits on the angular separation between neighbor cameras) based on dense surface sampling for GPU, which greatly reduces the computing time. When working in wide-baseline scenarios with real-time constraints, it is necessary to consider image features robust against perspective discrepancies, like silhouettes of foreground objects.

The methodology described in this article is able to reduce the data complexity in multi-view scenarios by introducing a reasonable amount of computational cost when compared to state-of-the-art surface reconstruction methods (as shown in Section VI). This is achieved thanks to a fast, massively parallel algorithm for 3D reconstruction inspired by the sparsity of the features to be reconstructed. The suitable algorithm design (Section III), inspired by *Markov-Chain Monte Carlo* methods [17], [18] provides good scalability properties to a growing number of input video streams, as shown in Section V-B. We also show that the proposed representation for multi-view video is a compelling approach for real-time streaming applications. We are able to reduce the transmission/storage cost of the multi-view video stream for distributed applications by introducing controlled losses with respect to the original video data, with the advantage of gaining the reconstructed 3D structure of the scene (Section VI-A).

II. Problem statement

We consider a scenario where dynamic scenes are captured by calibrated multi-camera rigs, with static or known background. We first focus on the multi-camera setup. Often, practical constraints do not allow configuring a small-baseline multi-camera rig. This does not only exclude multi-view stereo techniques from being robustly applied to the captured data, but also requires pre-calibration of all involved cameras, since simultaneous reconstruction and calibration techniques (*structure-from-motion* [19]) cannot be applied. Calibration information can be obtained offline using semi-automatic methods like [20].

Since cameras and background are fixed, a pre-processing stage, not covered in this article, will be responsible of segregating background and foreground pixels in each video stream. Foreground (FG) silhouette information proves as a sufficient visual cue for retrieving a good estimate of the actual surfaces of FG objects, which are the dynamic part of the scene and, therefore, the ones to be reconstructed from frame to frame. These estimates will be acceptably accurate as long as the number of available views is sufficiently dense and the cameras are evenly distributed all around the scene. In practice, for scenes without much clutter, it will suffice that angles between viewing directions of neighbor cameras are in the order of 60 degrees, while it would be much smaller (10-20 degrees) for image-based interpolation [2]. In order to extract FG silhouettes, [21], [22] provide improved results by conveniently exploiting existing redundancies among views. The latest results in monocular FG segmentation [23] are capable of robustly detecting shadows and other undesired false detections thanks to the fitting of finely tuned *Gaussian Mixture Models*. Despite of the ever improving results in FG segmentation, automatic silhouettes are still prone to errors. In order to compensate this, the 3D surface reconstruction scheme presented in Section III provides a mechanism for enhanced robustness, resulting in *conservative* estimates of the position of surfaces in the 3D space (it can be interpreted as if the volume enclosed by the reconstructed surfaces had been dilated by spatially-varying 3D structuring elements with geometry given by the camera setup).

Reconstructed surfaces are represented by means of dense sets of oriented, colored 3D points (also called *surface patches* or *surfels* in the literature [24]). Relevant details about this stage, which is the main contribution of this article, are given in Section III. Compared to our prior work [13], we introduce a more efficient and robuster mechanism for surface reconstruction, which is also able to exploit temporal correlation in order to speed up the processing of multi-view video sequences. In Section IV, a fast meshing algorithm produces continuous surfaces out of the discrete surfel representation. Compared to [13], an extension of the meshing method automatically corrects previously unobserved topologic errors under certain configurations. The continuous representation of the multi-view data presents practical advantages: on the one hand, it provides a complete description of the surface topology; on the other hand, it automatically maintains an ordering for interpolating the surface between known surface samples.

III. Surface Reconstruction

Surface reconstruction from multi-view settings can be viewed as a search for a sufficiently dense set of oriented 3D points that match the visual features detected in each view. Whereas the a-priori distribution of the surface points might be unknown (and, therefore, the search of each of these points could be very costly), a Markov Chain can be built, tightly describing the distribution of surface points at different scales around detected ones. This allows to obtain a dense surface reconstruction at a fraction of the expected linear cost of adding each additional camera when a large number of input video streams is available, as shown in Section V-B.

A. Markov-Chain Monte Carlo

The complex distribution of surface samples (or points) can be described as a Markov Chain, where the density of surface points is likely to be higher in regions around detected surface points and lower otherwise. Thus, once a *parent* surface point has been found, its position and orientation can be used to condition the distribution of new surface points around it, as introduced below.

Metropolis-Hastings [25] is a useful Monte Carlo method for Markov Chains when statistical moments (of the surface distribution in our problem) are requested. However, in our surface reconstruction problem, we intend to find just two features: surface positions and corresponding orientations. *Rejection sampling* allows obtaining suitable samples by first generating random ones and then keeping those fitting the desired distribution, which is locally modeled based on previously detected values for the two features.

B. Algorithm overview

In outline, given a set of (noisy) FG silhouettes corresponding to images captured in the same time instant, the goal of our Markov-Chain Monte Carlo approach is to classify a subset of randomly generated 3D points as lying on surfaces. We base the classification on the result of a cost function defined over the silhouettes, taking into account these may contain errors in form of misclassified pixels. Let p_c be the pixel onto which a randomly generated 3D point projects in view c and $I_c(p_c) \in \{1, 0\}$ the image value indicating whether pixel p_c belongs to the foreground. The 4-neighborhood of the pixel $N_4 = \{p : \|p - p_c\|_1 = 1\}$, where $\|\cdot\|_1$ stands for the *Manhattan* distance, is also used to define the cost function $d = \prod_c I_c(p_c) \cdot \left(\sum_c \sum_{p \in N_4} (1 - I_c(p))\right)$. The first term indicates whether the 3D point projects onto a FG pixel for all views, whereas the second one indicates wether the 3D point belongs to the contour of at least one of the silhouettes. Note that the definition of the second term implies not only rim points (*i.e.* points projecting onto the contour of all silhouettes) are reconstructed, but rather any 3D point lying on the surface of the visual hull. In order to enhance the robustness of the cost function in presence of segmentation errors, we define a relaxation of this cost function as

$$\delta = \left\lfloor \frac{1}{N_v - \tau} \sum_c (1 - I_c(p_c)) \right\rfloor \cdot \left(\sum_c \sum_{p \in N_4} (1 - I_c(p)) \right), \tag{1}$$

where the modification of the first term allows that up to τ out of the total N_v views classify the projection of a 3D point as background before deciding that the point does not belong to the silhouette-consistent volume.

By producing clusters of surface points at small distances, the surface orientation can be estimated at these discrete locations. This is done by fitting a plane to each of these clusters (assuming large local curvature radius at small scales) and extracting the normal as the director vector with its sign set such that, when projected onto the available views, it points out of the silhouette for at least one of them.

In the following, we introduce the complete method with improved efficiency. An initial scouting procedure, out of which a sparse set of *seed* surface points is obtained, is followed by a propagation stage, in which arbitrarily dense sets of surface points are efficiently obtained by exploiting spatial correlation (embedded in the Markov-Chain model).

C. Scouting

Let $s_x \times s_y \times s_z$ be the volume of a bounding box enclosing the scene to be reconstructed. Random 3D points are uniformly generated to lie inside of this bounding box. In order to accept a random 3D point $\mathbf{x}$ as a seed surface sample $\mathbf{x}_s$, locally describing the surface of one of the objects to be reconstructed, three conditions, ranging from multi-view consistency tests to topologic requirements, have to be fulfilled:

a) $\mathbf{x}$ projects onto a FG pixel for at least $N_v - \tau$ views.
b) $\mathbf{x}$ projects onto a contour pixel for at least one view.
c) A normal ($\mathbf{x}$'s orientation) can be estimated by operations on a close neighborhood.

In the following we review the details of each of these three conditions and their relation to Eq. (1).

a) Robust consistency: The *first condition* restricts the set of reconstructed points to those in the (conservative) visual hull. Let N_v be the number of cameras with frustum including the 3D point $\mathbf{x}$ under evaluation, i.e. the number of cameras for which the projection lies inside of the image. If the pixel onto which $\mathbf{x}$ projects belongs to the background of the scene for at least $\tau + 1$ of these N_v cameras, the point is discarded. This maps to the first term of Eq. (1).

In presence of segmentation errors, with some pixels wrongly marked as belonging to the foreground, inter-camera consistency will most likely reject points projecting onto one such pixel. The case of missed foreground, however, would introduce severe reconstruction errors.

Assuming segmentation errors are uncorrelated between views, the presence of such an error in $\tau = 1$ of the N_v cameras onto which $\mathbf{x}$ projects is probably accompanied by $N_v - \tau$ $(= N_v - 1)$ correct decisions in the remaining cameras. This assumption holds when the baseline is large enough and the neighbor viewpoints are clearly dissimilar. In other case, the tolerance τ must be set to a higher value. By appropriately setting τ, a better estimate of the actual object can be obtained from noisy foreground detections at the cost of slightly dilated reconstructed shapes, resulting in the conservative estimate of the silhouette-consistent enclosed volume.

b) Topology: The *second condition* checks that the 3D point belongs to the silhouette-consistent surface. Given the pixels $\{\mathbf{p}_c\}$ onto which $\mathbf{x}$ projects in the images corresponding to each camera c, we test if at least one of the pixels in its 4-neighborhood belongs to the background. If this condition is met by at least one of the cameras, the point passes this test. Note that this condition, which maps to the second term in Eq. (1), does not restrict the search to only rim points, but rather to any point lying on the conservative estimate of the silhouette-consistent surface.

c) Orientation: The *third condition* checks that the orientation of each surface point (necessary for an efficient dense search through propagation, as shown in Section III-E) can be estimated by locally approximating the surface by a plane. In order to estimate $\mathbf{x}$'s orientation, a plane is fitted to a small local cluster of surface points (4 points are used in practice) in a close neighborhood of the target one: given $\mathbf{x}$, a new random search is performed in a small ball of radius ρ_l centered at $\mathbf{x}$. ρ_l is an adjustable parameter that defines a small radius such that the local surface is practically flat (constant normal). The three new points generated in the small ball around $\mathbf{x}$ are required to fulfill Eq. (1). Then, a plane is fitted to the set of 4 points by least squares and the director vector, $\hat{\mathbf{n}}$, is taken.

To determine the sign of $\hat{\mathbf{n}}$, an iterative method is applied, which consists in: (1) Add and subtract the orientation of the surface point multiplied by a small scaling value to the position of the surface point ($\mathbf{x}_+ := \mathbf{x} + \epsilon\hat{\mathbf{n}}$ and $\mathbf{x}_- := \mathbf{x} - \epsilon\hat{\mathbf{n}}$). Project $\mathbf{x}_+$ and $\mathbf{x}_-$ onto all views. (2) If one of these points lies inside of the visual hull and the other outside, $\mathbf{x}$ (and its local neighbors) are accepted. If $\mathbf{x}_-$ belongs to the visual hull, the current sign is correct. Otherwise, we change it ($\hat{\mathbf{n}} := -\hat{\mathbf{n}}$). (3) Else, if both projection tests have the same result and $\epsilon > \epsilon_{\text{min}}$, repeat the process with $\epsilon := 4/5\epsilon$. If $\epsilon \leq \epsilon_{\text{min}}$, the point cannot be successfully oriented and $\mathbf{x}$ and its neighbors are discarded.

This condition also constrains the shapes that can be reconstructed. If a volume is *thin* enough as to project onto silhouettes with very narrow widths for all views, surface points may be discarded. One possible solution could be using other models better fitting such regions, e.g. cylindrical.

D. Improved scouting

The result of the scouting stage is a sparse set of seed surface samples uniformly distributed all over the surfaces of foreground objects. However, this stage constitutes the bottleneck of the sampling strategy. Indeed, we observe the search is driven at random in large work volumes (the space where the scene takes place captured by the multi-camera rig). Even though most of the surface samples are to be obtained during the local propagation stage of the method (Section III-E), a large amount of computational resources are spent in this initial search.

Two different techniques are proposed, which reduce the computation time for scouting. The first one (*multi-resolution scouting*) is based on considerations about the spatial sampling density related to the resolution of the input images, whereas the secone one (*dynamic scouting*) is based on the exploitation of temporal correlation between the position of surface samples in consecutive time instants.

a) Multi-resolution scouting: A multi-resolution approach can be used to speed up the scouting stage. As stated above, the projection onto a contour pixel for at least one of the silhouettes is a requirement for a 3D point to be part of the silhouette-consistent surface. We observe that given a set of silhouettes corresponding to different views of a scene, the likelihood of finding a contour pixel by a random search decreases when the resolution of the images increases.

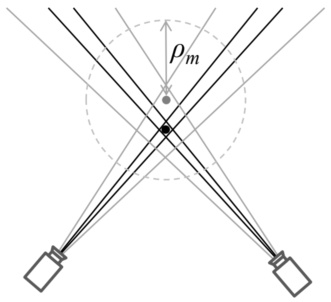

Fig. 2. Local search in a sphere with radius ρ_m centered at a low-res. seed (gray dot) to obtain the high-res. one (black dot). The gray and black half-lines starting at each camera show the back-projection of a low-res. contour pixel and a high-res. one, respectively.

Thus, the presented strategy exploits this fact by (1) first driving a search for low-resolution surface seeds from nearest-neighbor-downscaled silhouettes and (2) then driving a local search for high-resolution surface samples from the original silhouettes in high resolution.

Let $N_x \times N_y$ and l_c be the area of the input images and the length of a contour, respectively. If the resolution of an image is reduced by decimating each dimension by a factor δ, the corresponding contour length is also divided by δ.

Now let a randomly chosen 3D point project onto one of the input views in search of a contour pixel. The success probability at full resolution, π_f, can be considered $\propto \frac{l_c}{N_x \times N_y}$, assuming random points project with equal likelihood to any point in the image. Then, for the success probability after decimation, $\pi_d \propto \frac{l_c/\delta}{N_x/\delta \times N_y/\delta} = \delta\pi_f$. This translates into a reduced number of random attempts to find a surface point.

The low-resolution seed $\mathbf{x}_l$ obtained from the low-resolution silhouettes (step 1) is not used as a surface sample, since it is not accurate with respect to the high-resolution silhouettes. However, similar to the case of dynamic scouting (introduced below), a seed surface sample can be found in a small neighborhood of every low-resolution seed.

Using the high-resolution silhouettes (step 2), a low-resolution seed $\mathbf{x}_l$ can be used as the center of a smaller search region. As shown in Fig. 2, a spherical search region with a *multi-resolution search radius* ρ_m can be defined in order to find a valid seed surface sample. ρ_m should be as small as possible in order to keep the method efficient, but still large enough to be able to find a valid high-resolution seed in the region delimited by the back-projection of the pixels in the low-resolution views.

b) Dynamic scouting: The scouting stage can also benefit from exploiting temporal correlation in multi-view video sequences. Indeed, the information about the position of surface samples in a given time instant can reduce the time required to find the seeds in the next time instant by limiting the volume in which new surface samples are to be searched for.

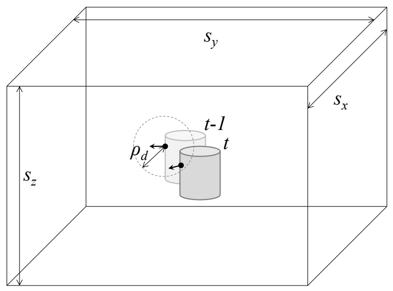

Fig. 3. By exploiting temporal correlation, the search space for scouting can be reduced from the whole volume ($s_x \times s_y \times s_z$) to a smaller one ($4/3\pi\rho_d^3$), thus increasing the search efficiency.

In order to exploit this correlation, the seed samples are transferred from one time instant to the next one using a more efficient search strategy than random search in the whole work volume. The complete, high-density covering of the surfaces is left to the efficient propagation scheme presented in Section III-E.

Assuming seed surface samples are evenly distributed over the surfaces of objects –which actually occurs–, new positions for these samples can be searched for by independently setting a symmetric search region with a large enough *dynamic scouting radius* ρ_d. This way, the placement of new seed samples for the current time instant does not favor any specific direction. Given three random values u, v, w in the ranges $[0, 1]$, $[0, 2\pi]$ and $[0, \pi]$, respectively, the candidate seed sample in the time instant t is computed as:

$$\mathbf{x}_t = \mathbf{x}_{t-1} + u\rho_d(\sin w \cos v, \sin w \sin v, \cos w)^\top. \quad (2)$$

A valid $\mathbf{x}_t$ is found when all three condition introduced in Section III-C are met. Then, the method continues until the complete list of seed surface samples from the previous time instant is processed.

Fig. 3 illustrates the gain in search efficiency that can be achieved by exploiting the temporal correlation of the scene in this manner. The search volume is reduced to $4/3\pi\rho_d^3$, which increases the probability of succeeding at finding a surface sample $\propto \frac{s_x \times s_y \times s_z}{4/3\pi\rho_d^3}$.

E. Fast propagation

New surface samples in the neighborhood of each seed surface sample can be iteratively obtained with progressively smaller distances from each other, resulting in a dense surface sampling, by exploiting the expected local distribution of surface samples. The benefit of introducing new samples in this manner is the reduction of the number of tries until a valid surface sample is found when compared to that of the scouting stage.

A 3D propagation region is defined for each surface sample, where new surface samples are very likely found. The design rules of this region are the following: (1) Assuming surfaces are locally flat, or present a large curvature radius, preference will be given for new surface samples to be placed close to the plane defined by the surface sample's position and orientation; and (2) given the fact that the omni-directional method is going to be applied iteratively, new surface samples will be placed at a sufficiently large distance from the seed surface sample.

Given a seed surface sample, we define two tangential vectors $\hat{\mathbf{u}}$ and $\hat{\mathbf{v}}$ which are orthogonal to each other and also to the normal $\hat{\mathbf{n}}$. By construction, the two former vectors lie on the plane defined by the seed surface sample's position and orientation.

One possible shape for the propagation region is a rectangle, although others could be used with similar results. Thus, we parameterize the sample distribution by the position, orientation and scale of a rectangular propagation region around an existing surface sample. Three random values are required for defining each testing point $\mathbf{x}_p$: (1) normal offset ν_n, or distance between the seed sample and the new testing point along the axis of the seed sample's normal $\hat{\mathbf{n}}$; (2) tangential offset ω_u, or distance between the seed sample and the new testing point along the $\hat{\mathbf{u}}$ axis; and (3) tangential offset ω_v, or distance between the seed sample and the new testing point along the $\hat{\mathbf{v}}$ axis.

The random 3D testing point for local propagation $\mathbf{x}_p$ is computed from these three random values and the position and orientation of an existing seed surface sample $\mathbf{x}_s$ as:

$$\mathbf{x}_p = \mathbf{x}_s + \omega_n\hat{\mathbf{n}} + \omega_u\hat{\mathbf{u}} + \omega_v\hat{\mathbf{v}}. \quad (3)$$

Let ν_n and ρ_p be a maximum offset along the normal direction and a propagation radius, respectively, n a uniform random variable in the range $[0, 1]$, and u and v uniform random variables in the range $[-1, 1]$. The three random values ω_n, ω_u and ω_v, which are used to generate candidate 3D points $\mathbf{x}_p$ in the rectangular propagation region, are obtained as:

$$\begin{aligned} \omega_n &:= \nu_n\left(2n-1\right) \\ \omega_u &:= \begin{cases} \rho_p\left(u+\frac{1}{2}\right), u \geq 0 \\ \rho_p\left(u-\frac{1}{2}\right), u < 0 \end{cases} \\ \omega_v &:= \begin{cases} \rho_p\left(v+\frac{1}{2}\right), v \geq 0 \\ \rho_p\left(v-\frac{1}{2}\right), v < 0 \end{cases} \end{aligned} \quad (4)$$

The appearance of the rectangular propagation region is shown in Fig. 4 (a). The goal of the empty space between a seed surface sample and the new ones obtained by propagation is to let the propagation algorithm be applied iteratively, at increasingly smaller scales, until the desired number of surface samples is reached. With this, an omni-directional search for new surface samples using the same propagation region at

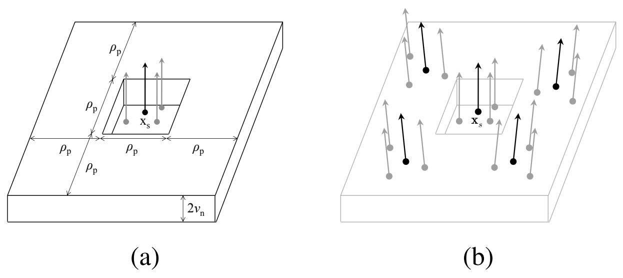

Fig. 4. (a) Rectangular propagation region around a surface sample $\mathbf{x}_s$ (and its neighbor samples, used for estimating the local orientation). (b) It favors propagation on the plane orthogonal to the normal vector while allowing small surface curvatures with variations of up to ν_n along the normal.

different scales can be done without creating undesired clusters of surface samples at small distances. A typical distribution of samples obtained by propagation is shown in Fig. 4 (b).

In more detail, let $q := \{\mathbf{x}_s\}$ be a queue of seed surface samples and N_s the desired total number of surface samples. Initially, q contains N_q seed surface samples obtained in the scouting stage (q does not contain the additional samples obtained for the local estimation of the orientation of each seed surface sample). ν_n and ρ_r are initialized to suitable values (Section V). Then, while the required number of surface samples N_s is larger than the current number N_t (which is initially equal to $4N_q$, the total number of samples including neighbors), the method proceeds iteratively, as summarized in Algorithm 1. As a result, N_s surface samples densely cover

Algorithm 1: FAST PROPAGATION(q, ρ_p)

```
   Data: Seed samples $q$ and propagation radius $\rho_p$
   Result: Propagated ($s$) and neighbor ($n$) samples
1  Create empty $q'$, $s$ and $n$ queues
2  while $N_t < N_s$ do
3      while $q$ not empty do
4          Pop seed sample $\mathbf{x}_s$ from $q$
5          for $i \leftarrow 1$ to $4$ do
6              repeat
7                  Obtain a test sample $\mathbf{x}_p$ (Eq. 3)
8              until $\mathbf{x}_p$ passes tests (Section III-C)
9              Push $\mathbf{x}_p$ into $q'$ and $s$, and neighbors into $n$
10             $N_t \leftarrow N_t + 4$
11     $q \leftarrow q'$   $q' \leftarrow \emptyset$   $\rho_p \leftarrow \frac{1}{2}\rho_p$
12 return $s$ and $n$
```

the surface with an increased sampling efficiency, thanks to the adaptation of the propagation region to the local distribution of surface samples.

F. Post-processing

In order to both improve and complement the geometric information about the surface, three post-processing stages are cascaded at the output of the reconstruction method. These three methods follow the same design as the post-processing stages in [13].

These stages are: (1) a refinement of the orientation estimation (by using larger neighborhoods when fitting planes by least squares); (2) an anisotropic smoothing stage in which each surface point is displaced along the direction of the surface orientation in order to lie on a plane identically oriented passing through the average point of a local neighborhood, while being subject to the (τ-relaxed) silhouette constraints; and (3) a coloring stage adding color information to the geometric description of the surfaces by making a weighted average of the color viewed by the three best oriented cameras where the surface point is visible.

By applying these post-processing stages we make the detection of 3D points robust against geometric errors due to the discretization of the input images. Besides, we also add valuable color information that completes the compact description of the reconstructed surfaces.

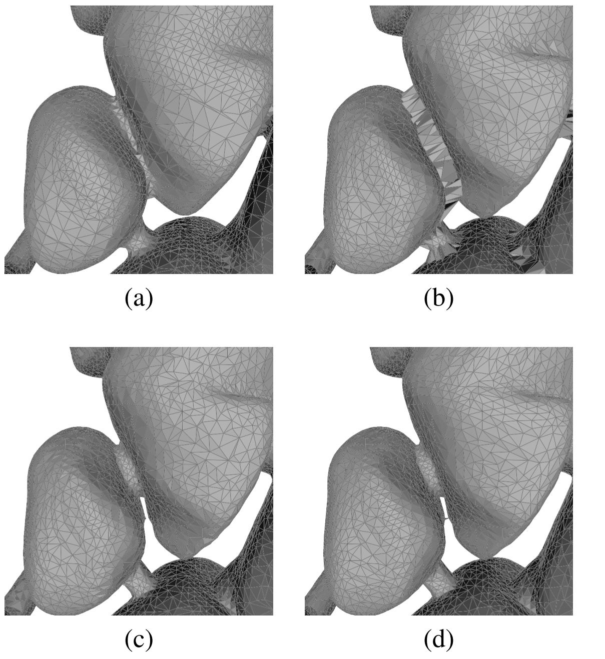

Fig. 5. Detailed views of meshing results using (a) Poisson Reconstruction, (b) BPA and (c) our proposed method, obtained from the oriented vertices of (d) the reference ground-truth mesh. Better viewed when zoomed in.

IV. MESHING

A continuous surface representation is of interest for two reasons. On the one hand, the complete description of the surface topology can be exploited in post-processing [26]; on the other hand, by choosing a proper primitive for describing elemental surface patches, such as triangles, standard rendering pipelines in graphics hardware can be used for quickly rendering the reconstructed surfaces as viewed from any desired virtual viewpoint.

We present an extension of the fast propagation-based meshing algorithm originally presented in [13]. The major differences between our method and the *Ball-Pivoting Algorithm* (BPA) [27] are the introduction of a different set of rules for guiding the propagation while keeping topological correctness and the use of an efficient and more flexible method for making spatial queries (nearest neighbors from a kd-tree), which does not require the iterative application of surface propagation at different scales and avoids topological inconsistencies, as shown in Fig. 5 (b) and (c).

In outline, the algorithm is applied iteratively until at least αN_p surface points have been added to the continuous surface, with $\alpha = 0.9$ in all our experiments. The purpose of the iterative method is two-fold: on the one hand, it allows obtaining continuous surfaces out of disconnected sets of surface points; on the other, it provides new propagation directions for areas that were not tessellated from previous directions due to restrictions on the allowed propagation cases.

Similar to BPA, the proposed propagation algorithm starts by defining a triangle connecting three close coherently oriented surface points and adding the corresponding three edges (initial surface contour) to an *edge queue*. Then, the algorithm processes the edge queue until it is empty. The propagation of the contour basically consists in connecting each edge

TABLE I
MESHING PERFORMANCE WITH KNOWN GROUND-TRUTH

Method		*armadillo*	*dragon*	*happy*
Poisson	Time	35.64	43.61	49.55
	Mem.	224	640	704
	RMSE	591	874	1232
BPA	Time	347.00	1846.49	2988.64
	Mem.	**64**	**192**	**224**
	RMSE	61	105	144
Proposed	Time	**3.06**	**10.11**	**15.44**
	Mem.	96	288	320
	RMSE	**40**	**103**	**113**

extracted from the queue to a suitable surface point (either a point forming part of the current tessellated surface contour or an unused one), thus generating a new triangle and adding up to two new edges to the edge queue. The set of rules, the details of which are given in [13], is defined to prevent the surface from folding, by restricting propagation in the forward direction, and from creating two close, overlapping layers.

As an improvement with respect to the original version of the algorithm described in [13], the extended version includes a mechanism for detecting and correcting topologically incorrect configurations of triangles. The mechanism consists in removing edges shared by more than two triangles, as well as the set of triangles sharing these edges, and proceeding with the propagation from the new contour edges of the resulting hole. These multiple-shared edges were empirically found to be the source of erroneous topological configurations in our extended meshing experiments, shown in Section IV-A. The resulting tessellated surface is a 2D manifold in 3D space.

As a difference with volumetric regularization-based meshing algorithms, e.g. Poisson reconstruction [12], our method is not designed to be robust against noisy surface points but, in exchange, it provides a greater accuracy (avoiding over-smoothed results) and a much smaller computational cost in terms of memory and run-time, as shown below.

A. Meshing tests

In this section we provide a partial testing on the performance of our extended meshing algorithm. Two well-known and extensively used meshing algorithms, BPA [27] and Poisson Reconstruction [12] (both available in Meshlab [28]), and the proposed method have been used for obtaining continuous surfaces out of several sets of oriented points extracted from the *Stanford 3D scan repository* [29], for which reference meshes are provided. These reference meshes are used as the ground-truth in order to compare the accuracy by means of the RMS Hausdorff distance (RMSE). Besides, the computation time and memory usage of each method is also measured. The input used for all methods is the set of oriented vertices of each reference mesh. As shown in Table I, the proposed method is consistently more accurate and faster than the reference methods, whereas the memory usage is close to that of BPA. In Fig. 5, close-up views of the results obtained with each of the three tested methods can be compared to the ground-truth mesh.

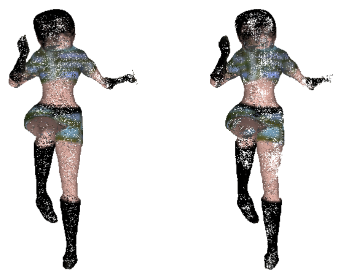
Fig. 6. Left, surface reconstruction using empirically determined parameters; right, erroneous surface reconstruction (non-uniform sampling) due to a wrong choice of the multi-resolution scouting radius ρ_m (1/8 smaller than default).

V. EXPERIMENTAL RESULTS

The surface reconstruction algorithm is parameterized by the number of output surface samples (100000 in our experiments) and the size of the different search regions used in every stage. These parameters, including the ones for improving the *scouting* stage, have been empirically determined in order to provide the expected behavior, derived from the bounding box volume $s_x \times s_y \times s_z$, and are listed in Table II. The datasets used for determining them contain scenes similar to those reconstructed in our tests: they are multi-view (7 views) captures of persons in an empty space with approximate dimensions of $4 \times 3 \times 2.5\ m^3$. About the sensitivity of the parameters, in Fig. 6 we show the effect of an erroneous selection of the initial multi-resolution scouting radius ρ_m, where some parts of the surface are not covered by the seed samples. A nice feature of these parameters is that they are independent of the resolution of the images, since they are defined on 3D space.

Three multi-view real sequences from datasets available in [8] and a synthetic one (*kung-fu girl*, [30]) have been used, the details of which are listed in Table III.

Three experiments have been performed on these data. The first two experiments demonstrate the efficiency increase that can be achieved by applying any of the two proposed improvements for seed surface sample scouting. The third experiment showcases an interesting property about the scaling

TABLE II
SURFACE RECONSTRUCTION PARAMETERS

Parameter	Value
Reference radius ρ_r	$\frac{1}{200}\sqrt{s_x^2 + s_y^2 + s_z^2}$
Initial propagation radius ρ_p	$2\rho_r$
Normal estimation radius ρ_l	$\frac{1}{2}\rho_r$
Multi-res. scouting radius ρ_m	$2\rho_r$
Dynamic scouting radius ρ_d	$8\rho_r$
Propagation normal offset ν_n	ρ_l

TABLE III
MULTI-VIEW SEQUENCES FROM [8] AND [30]

Sequence	# Views	# Frames	Resolution
dancer	8	201	780×582
children	16	339	1624×1224
martial	16	210	1624×1224
kung-fu girl	25	200	320×240

TABLE IV
FRAME PROCESSING TIME WITH DYNAMIC SCOUTING

Sequence	Default	Dynamic
dancer	1.02 s	0.46 s
children	2.73 s	0.92 s
martial	2.35 s	0.88 s
kung-fu girl	1.4 s	0.56 s

TABLE V
AVERAGE RMS HAUSDORFF DISTANCES TO DEFAULT SCOUTING

Sequence	ρ_r	Multi-res.	Dynamic
dancer	0.03	0.0052	0.0043
children	0.03	0.0047	0.0045
martial	0.03	0.0043	0.0055
kung-fu girl	0.35	0.051	0.052

of the proposed surface sampling strategy to large multi-view settings that makes it ideal to be used in such conditions. The results are obtained from a single-threaded implementation running on a Core 2 Duo 2.8 GHz CPU.

A. Dynamic and multi-resolution scouting

These two first experiments demonstrate the increase in efficiency that can be achieved by modifying the initial scouting stage with the two proposed scouting improvements.

In Section III-D-a, a multi-resolution approach has been proposed, the most important adjustable parameter of which is the decimation level in the input silhouettes. In Fig. 7, the execution times with different levels are shown. The case with decimation set to 1 corresponds to the default scouting.

In Section III-D-b, a dynamic scouting strategy has been presented, which reduces the computation time in multi-view video sequences by exploiting temporal correlation. The average computation times for each sequence with and without dynamic scouting are listed in Table IV, showcasing how inter-frame similarity of the position of surfaces can be effectively used for reducing the search regions for seed surface samples. This results in a clear increase in frame-rate.

The resulting efficiency gain obtained by both dynamic and multi-resolution scouting is similar. The extra cost from initially obtaining seed samples at low resolution in multi-resolution scouting is partly balanced with the fact that the low-resolution seeds are in average closer to the actual surface than the high-resolution seeds from a previous time instant in dynamic scouting, which translates in the choice of a multi-resolution scouting radius smaller than its dynamic scouting counterpart. In Table V we list the average RMS Hausdorff distance (RMSE) between the surfaces obtained with default scouting and the ones obtained by using one of the improved scouting strategies for each sequence. For multi-resolution scouting, we have chosen an initial decimation of 8. Distances are one order of magnitude below the reference radius ρ_r.

B. Efficiency in multi-camera environments

A limiting problem when working in multi-camera environments is the cost associated to the addition of each new camera. Typically, algorithms exploiting multi-view data can grow linearly, e.g. in volumetric reconstruction techniques, or even quadratically, e.g. in image-based approaches where multiple cross-validations between camera views take place, with the number of available cameras.

However, the proposed reconstruction strategy benefits from the fact that the ratio between silhouette contour and total number of pixels remains practically constant with respect to the number of available cameras when these are placed at similar distances from the objects of interest. Even better, the ratio between silhouette contour pixels (in image space) and occupied volume (in 3D space) increases with the addition of each new camera. Thus, even if the cost of a projection test grows linearly with the number of views, the probability that a 3D point is a surface point also grows for each additional view, partly balancing this effect. As shown by this experiment, the result is that the proposed reconstruction strategy is well suited to large multi-camera settings.

Fig. 8 shows, in dashed lines, the extrapolated computation time from the first five views up to the whole range of available views (linear time). As shown, the measured time clearly grows at a smaller rate for subsequent views. In Fig. 9, the reconstructed surfaces for one time instant of each sequence with increasing numbers of input views are shown. The marginal increase of detail in the resulting surface becomes smaller, and so does the marginal cost of adding each additional view. This reflects the expected efficiency of the proposed reconstruction strategy.

The modeling of the local distribution of surface points for the fast propagation stage has proven successful at reducing the search space and the corresponding computing time. This results in a reduced marginal cost when introducing additional cameras for finer reconstructions. Although a naive implemen-

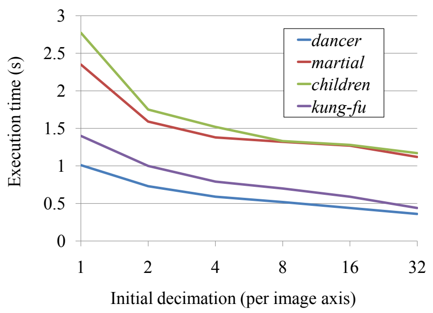

Fig. 7. Reconstruction time vs. initial level of image decimation in multi-resolution scouting. A decimation of 1 is equivalent to the default scouting with full-resolution images.

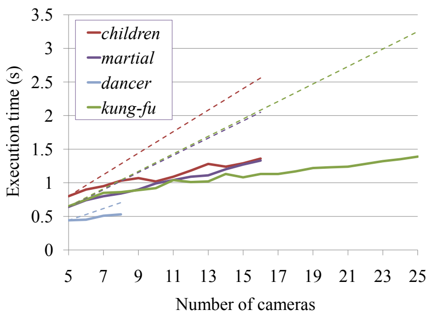

Fig. 8. Reconstruction time vs. number of cameras (continuous lines). The dashed lines show the time that would be required if the cost of adding a view was constant (as in volumetric approaches).

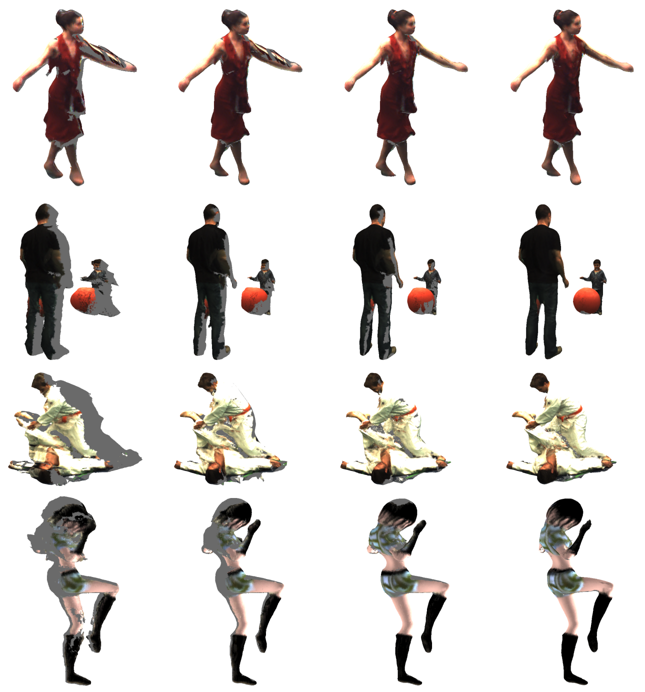
Fig. 9. The leftmost column shows surfaces reconstructed from only 5 viewpoints of each sequence. Towards the right, surfaces obtained with increasing number of views. The rightmost column shows the surfaces reconstructed from the whole multi-view set for each sequence.

tation of the the scouting stage assumes a very uninformative initial model of the distribution of surface samples, we have also shown how the two proposed alternatives (dynamic and multi-resolution) produce tighter distribution models, which further reduce the computing time.

VI. Application

We focus on a Free-Viewpoint Video (FVV) application, where the contents of interest of multi-view video sequences (the dynamic foreground) are streamed to a rendering application implemented in OpenGL, which sends triangles and color arrays to the standard rendering pipeline. The triangles correspond to the meshes obtained from the meshing algorithm, whereas the color arrays correspond to the vertices' colors. No projective texturing is required, thanks to the density of the surface representation. Whereas image-based rendering approaches [1] propose methods optimized for speed in FVV rendering, we focus here on methods for reconstruction, which provide a unique 3D representation of the dynamic contents of the multi-view scene. Besides rendering, scene analysis (including tracking [14] and pose estimation) can also be performed on this alternative representation.[1]

We validate the proposed method by qualitatively comparing the reconstruction performance to that of two state-of-the-art methods, *Exact Polyhedral Visual Hull* (EPVH) [9] and *Patch-based Multi-View Stereo* (PMVS2) [11], which are typically used in wide-baseline setups (by extracting foreground silhouettes) and small-baseline ones (with optional silhouettes), respectively. The selection of methods responds to the fact that EPVH provides projection-accurate estimates of the visual hull (when silhouettes are error-free) and PMVS2 is used at the core of contemporary approaches for large-scale reconstruction [31] and has a high rank in the Middlebury benchmark [32], particularly for sparse rings (16 views). Fig. 10 shows how the proposed method produces a more visually appealing surface geometry than the one obtained with EPVH, which is geometrically exact but does not compensate discretization-derived artifacts. Post-processing is rather difficult to apply on the EPVH surfaces, due to the irregular shape and size of the mesh triangles, besides textures must be separately transmitted or projected from the original images.

Multi-View Stereo (MVS) methods, e.g. PMVS2, are not tailored to scenarios with very wide-baseline setups, and usually require a high amount of texture to be present in the input images. Thus, our method also compares favorably to this approach in our target scenarios, although we must remark that our approach would not be suitable for setups where background subtraction is not possible. MVS is also capable of reconstructing concavities, which are impossible to recover by only imposing silhouette consistency.[2] In terms of computation time, PMVS2 required 28 minutes per frame (on a Intel Xeon 3 GHz CPU), whereas our algorithm is capable of extracting a denser set of surface points at a frame rate ∼4 fps. In our current OpenCL version, the implementation details of which are not included, we achieve real-time, with speed-ups between $4\times$ and $8\times$.

In Fig. 11, two possible uses of our free-viewpoint viewer are shown. On the left image we have captured an instant of the *dancer* sequence rendered from a novel viewpoint using a virtual background. On the right, we re-project a reconstructed surface from a scene from [8] onto one of the original viewpoints and overlay it with the corresponding original background.[3] A clear advantage of this representation is that, by transmitting the static background just once per video sequence and updating the 3D surfaces every frame, we can retrieve all views in every frame up to a certain level of accuracy.

[1]With a MATLAB implementation of [1] running at ∼5 fps we obtain no visible differences with our method (both obtain visual-hull estimates). A GPU implementation would be a possible alternative for FVV.

[2]For reference, using the Middlebury datasets (which contain large concavities) and metrics, the scores of our method for runtimes below 1s with the sparse ring datasets are 4.49 mm and 68.5% for *Temple* and 3.27 mm and 60.9% for *Dino*. Note our method does not introduce photo-consistency and is therefore not suitable for this MVS comparison.

[3]Further free-viewpoint-video examples and binaries can be found under `https://imatge.upc.edu/web/?q=node/1456`.

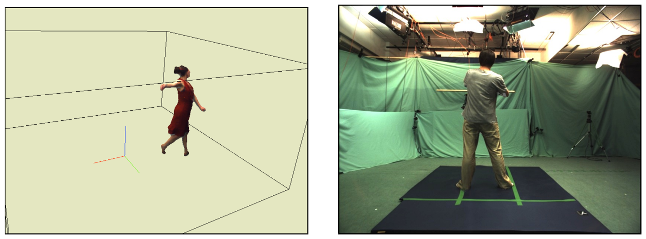
Fig. 11. Left: novel viewpoint of an instant of a scene captured by 8 cameras with virtual background. Right: composition of a surface reconstructed from 16 views projected onto an original viewpoint with original background.

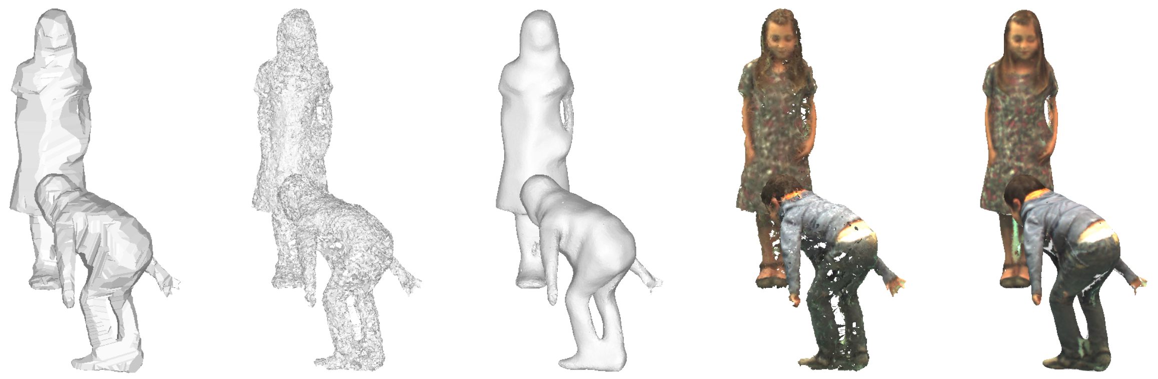

Fig. 10. Reconstructed surface of a frame in a multi-view sequence with 16 views. From left to right: geometry obtained with EPVH [9]; PMVS2 [11] concatenated with BPA [27]; proposed method; per-vertex colored surfaces from PMVS2+BPA; and proposed method.

A. Required density of surface points

One question that arises is: *Which is the required density of surface points to obtain an accurate representation?* We compare the average PSNR of the re-projection of the reconstructed surfaces onto the most challenging (worst PSNR) view against the number of surface points per pixel. In our experiments, the worst PSNR corresponds to the view with the largest number of FG pixels. After normalizing, we obtain the curves in Fig. 12. As observed, a number of surface samples of around twice the number of pixels occupied by the FG objects in the most limiting view seems to be a safe, yet economical, choice for representing scenes with good quality. This rule of thumb reflects that the method is performing at its best when a density of around one surface point per pixel is taken (assuming revolution symmetry for 3D objects).

Table VI shows how the proposed representation is suitable for streaming multi-view video for distributed applications. We compare the amount of data required to represent multi-view sequences with an image-based manner (either as individual frames or exploiting temporal redundancies in lossless video sequences) and with our approach. The bandwidth (storage resources) for transmitting (storing) multi-view video with static background can be notably reduced with our method by introducing a controlled amount of losses. In the table, FG CTM stands for compression of reconstructed colored meshes using OpenCTM [33], which compresses mesh data in a factor of about 10. PSNR figures in the worst view are still well above 45 dB. We are not attempting to use our method for coding, but rather introducing it as a powerful mechanism to obtain an alternative representation for distributed real-time applications where analysis and rendering functions run simultaneously. Our method does not only convey the visual information of the original images within a margin of controlled losses, but also augments this by providing a 3D estimate of the dynamic part of the scene using a suitable representation at the cost of a reasonable computation time.

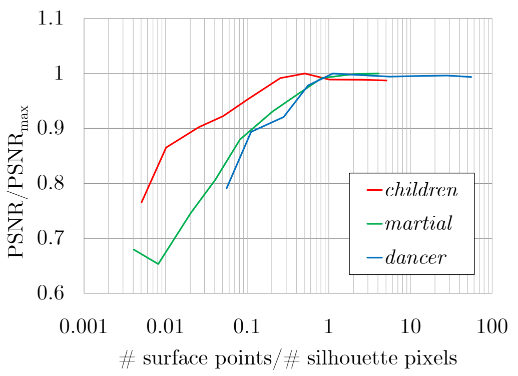

Fig. 12. Relative PSNR (with respect to the maximum) as a function of the ratio between the number of surface samples and the number of foreground pixels in the view where they take most pixels.

VII. Conclusions

An alternative representation method for multi-view video is proposed, based on a robust and efficient algorithm for reconstructing silhouette-consistent surfaces that exploits both spatial (Markov-Chain model) and temporal correlations. The main advantage of this reconstruction strategy is that, although it initially generates a sparse surface sampling with a low efficiency at the scouting stage (rejection sampling), an efficient propagation strategy provides a fast dense final sampling. Two improvements exploiting multi-resolution and temporal correlation have been proposed, which clearly increase the scouting efficiency. The proposed algorithm is GPU-friendly, since all processes can be parallelized. Although the implementation details are not the focus of this paper, the algorithm has also been ported to OpenCL, providing real-time reconstruction for interactive free-viewpoint video. The speed-up is in the range $4\times$ to $8\times$, but there is room for improvements. For the future work, we plan to generalize the proposed approach, introducing visibility estimates for each view, to obtain photo-consistent surfaces. We also plan to introduce regularization by exploiting the hierarchy of surface samples during propagation.

TABLE VI
3D SURFACES AS A REPRESENTATION FOR STREAMING

Format	Compress	*dancer* 8 views	*children* 16 views	*martial* 16 views
Images	PNG	808 MB	12 GB	27.2 GB
	H.264 LS	352 MB	4.6 GB	4.3 GB
Surface	FG Raw	756 MB	2.4 GB	12.1 GB
	FG CTM	76 MB	242 MB	1.2 GB
	(PSNR)	50.72 dB	70.85 dB	48.40 dB

ACKNOWLEDGMENT

This work has been partially supported by the Spanish Ministerio de Ciencia e Innovación, under project TEC2010-18094. We would also like to thank Marc Maceira, for his real-time OpenCL implementation of the reconstruction algorithm, and Albert Gil and Josep Pujal, for their technical support. We also thank the anonymous reviewers for their constructive feedback, which resulted in an improved manuscript.

REFERENCES

[1] W. Matusik, C. Buehler, R. Raskar, S. J. Gortler, and L. McMillan, "Image-based visual hulls," in *Proc. SIGGRAPH*, 2000, pp. 369–374.

[2] C. Lipski, C. Linz, K. Berger, and M. Magnor, "Virtual video camera: image-based viewpoint navigation through space and time," in *ACM SIGGRAPH Posters*, 2009, p. 93.

[3] Z. Zhang, L. Wang, B. Guo, and H.-Y. Shum, "Feature-based light field morphing," *ACM Trans. Graph.*, vol. 21, pp. 457–464, July 2002.

[4] E. de Aguiar, C. Stoll, C. Theobalt, N. Ahmed, H.-P. Seidel, and S. Thrun, "Performance capture from sparse multi-view video," in *ACM Proc. SIGGRAPH*, 2008, pp. 1–10.

[5] S. Würmlin, E. Lamboray, M. Waschbüsch, P. Kaufmann, A. Smolic, and M. Gross, "Image-space free-viewpoint video," in *Proc. of Eurographics Vision, Modeling, Visualization*, 2005.

[6] ISO/IEC JTC 1 /SC 29/ WG 11 N12036 , "Call for Proposals on 3D Video Coding Technology."

[7] L. Chiariglione, "Multimedia standards: Interfaces to innovation," *Proc. IEEE*, vol. 100, no. 4, pp. 893–904, 2012.

[8] 4D Repository, "Inria," http://4drepository.inrialpes.fr/, 2011.

[9] J. S. Franco and E. Boyer, "Efficient polyhedral modeling from silhouettes," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 31, pp. 414–427, 2009.

[10] R. Carceroni and K. Kutulakos, "Multi-view scene capture by surfel sampling: From video streams to non-rigid 3D motion, shape and reflectance," *Int. Journal on Computer Vision*, vol. 49, no. 2-3, pp. 175–214, 2002.

[11] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multi-view stereopsis," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[12] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson surface reconstruction," in *Eurographics Symp. on Geometry Processing*, 2006, pp. 61–70.

[13] J. Salvador, X. Suau, and J. R. Casas, "From silhouettes to 3D points to mesh: towards free viewpoint video," in *ACM Proc. 1st Int. Workshop on 3DVP*, 2010, pp. 19–24.

[14] C. Cagniart, E. Boyer, and S. Ilic, "Probabilistic deformable surface tracking from multiple videos," in *Proc. European Conf. on Computer Vision*, 2010, pp. 326–339.

[15] J.-Y. Guillemaut and A. Hilton, "Joint multi-layer segmentation and reconstruction for free-viewpoint video applications," *Int. J. Computer Vision*, vol. 93, no. 1, pp. 73–100, 2011.

[16] J. Y. Chang, H. Park, I. K. Park, K. M. Lee, and S. U. Lee, "GPU-friendly multi-view stereo reconstruction using surfel representation and graph cuts," *Elsevier Comput. Vis. Image Underst.*, vol. 115, no. 5, pp. 620–634, 2011.

[17] J. Liu and R. Chen, "Sequential monte carlo methods for dynamical systems," *Journal of the American Statistical Association*, vol. 93, no. 5, pp. 1032–1044, 1998.

[18] A. Doucet, S. Godsill, and C. Andrieu, "On sequential monte carlo sampling methods for bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, July 2000.

[19] N. Snavely, S. M. Seitz, and R. Szeliski, "Modeling the world from internet photo collections," *Int. J. Comput. Vision*, vol. 80, no. 2, pp. 189–210, 2008.

[20] J. Bouguet, "Camera calibration toolbox for matlab," 2000. [Online]. Available: http://www.vision.caltech.edu/bouguetj/calib_doc

[21] J. Gallego, J. Salvador, J. R. Casas, and M. Pardàs, "Joint multi-view foreground segmentation and 3d reconstruction with tolerance loop," in *Proc. IEEE Int. Conf. on Image Processing*, 2011.

[22] M. Sarim, A. Hilton, J.-Y. Guillemaut, H. Kim, and T. Takai, "Wide-baseline multi-view video segmentation for 3d reconstruction," in *ACM Proc. 1st Int. Workshop on 3DVP*, 2010, pp. 13–18.

[23] J. Gallego, M. Pardàs, and G. Haro, "Enhanced foreground segmentation and tracking combining bayesian background, shadow and foreground modeling," *Elsevier Pattern Recog. Letters*, vol. 33, p. 1558–1568, 2012.

[24] M. Gross, S. Würmlin, M. Naef, E. Lamboray, C. Spagno, A. Kunz, E. Koller-Meier, T. Svoboda, L. V. Gool, S. Lang, K. Strehlke, A. V. Moere, and O. Staadt, "blue-c: A spatially immersive display and 3d video portal for telepresence," in *ACM Transactions on Graphics*, 2003, pp. 819–827.

[25] W. Hastings, "Monte carlo sampling methods using markov chains and their applications," *Biometrika*, pp. 97–109, 1970.

[26] K. Hormann, B. Lévy, and A. Sheffer, "Mesh parameterization: Theory and practice," in *ACM SIGGRAPH Course Notes*, 2007.

[27] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin, "The ball-pivoting algorithm for surface reconstruction," *IEEE Trans. on Vis. and Comp. Graphics*, vol. 5, no. 4, pp. 349–359, 1999.

[28] P. Cignoni, M. Corsini, and G. Ranzuglia, "Meshlab: an open-source 3D mesh processing system," *ERCIM*, pp. 47–48, 2008.

[29] The Stanford 3D Scanning Repository, "Stanford Computer Graphics Laboratory," http://graphics.stanford.edu/data/3Dscanrep/, 2010.

[30] Kung-Fu Girl, "A synthetic test sequence for multi-view reconstruction and rendering research. Independent Research Group 3: Graphics, Optics, Vision. Max-Planck-Institut Informatik," http://www.mpi-inf.mpg.de/departments/irg3/kungfu/, 2005.

[31] Y. Furukawa, B. Curless, S. Seitz, and R. Szeliski, "Towards internet-scale multi-view stereo," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2010, pp. 1434–1441.

[32] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski, "A comparison and evaluation of multi-view stereo reconstruction algorithms," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, vol. 1, 2006, pp. 519–528.

[33] M. Geelnard, "OpenCTM, the Open Compressed Triangle Mesh file format," http://openctm.sourceforge.net, 2010.

Jordi Salvador holds a Senior Researcher position at the Image Processing Lab, Technicolor R&I in Hannover, Germany. He received a M.Sc. in Telecommunications Engineering in 2006 and a M.Sc. in the European MERIT Master program in 2008, both from the Universitat Politècnica de Catalunya (UPC) in Barcelona. He obtained the Ph.D. degree in 2011, also from UPC, where he was Research Assistant and contributed to projects of the Spanish Science and Technology System (VISION, PROVEC) and also to a European FP6 project (CHIL). His research interests include 3D reconstruction, real-time and parallel algorithms, new computer-human interfaces, image and video restoration, super-resolution and inverse problems.

Josep R. Casas is Associate Professor at the Department of Signal Theory and Communication, Technical University of Catalonia (UPC) in Barcelona. He graduated in Telecommunications Engineering in 1990 and received the PhD in 1996, both from UPC, where he is currently teaching Signals and Systems, Image Processing and Television Systems at the School of Telecommunications Engineering (Telecom BCN). He was visiting researcher at CSIRO Mathematics & Information Sciences in Canberra, Australia from 2000 to 2001. Josep R. Casas is Principal investigator of the project PROVEC ("Video Processing for Controlled Environments") of the Spanish R&D&I Plan started in 2007, and has led or contributed to a number of industry-sponsored projects, projects of the Spanish Science and Technology System (VISION, HESPERIA) and European FP projects (ACTIBIO, SCHEMA, ADVISOR). In particular, he coordinated UPC contribution to CHIL ("Computers in the Human Interaction Loop"), an IP of the IST/EU 6th Framework Program in the strategic objective of Multimodal Interfaces, involving video, audio and natural language technologies. Josep R. Casas has authored or co-authored over 10 papers in international journals, 12 papers in LNCS, 50 contributions to conferences and 9 book chapters and a teaching book in the areas of video coding, analysis, indexing and image processing.