A COMPACT 3D REPRESENTATION FOR MULTI-VIEW VIDEO

Jordi Salvador and Josep R. Casas

Image and Video Processing Group
Universitat Politècnica de Catalunya

ABSTRACT

This paper presents a methodology for obtaining a 3D reconstruction of a dynamic scene in multi-camera settings. Our target is to derive a compact representation of the 3D scene which is effective and accurate, whatever the number of cameras and even for very-wide baseline settings. Easing real-time 3D scene capture has outstanding applications in 2D and 3D content production, free viewpoint video of natural scenes and interactive video applications.

The method proposed here has several original contributions on how to accelerate the process: it exploits spatial and temporal consistency for speeding up reconstruction, dividing the problem in two parts. First, 3D surfaces are efficiently sampled to obtain a silhouette-consistent set of colored surface points and normals, using a novel algorithm presented in this paper. Then, a fast, greedy meshing algorithm retrieves topologically correct continuous surfaces from the dense sets of oriented points, providing a suitable representation for multi-view video.

Compared to other techniques in the literature, the presented approach is capable of retrieving 3D surfaces of foreground objects in real-time by exploiting the computing capabilities of GPUs. This is feasible due to the parallelized design of the surface sampling algorithm. The reconstructed surfaces can effectively be used for interactive representations. The presented methodology also offers good scalability to large multi-view video settings.

***Index Terms*—** Multi-View, Surface, Mesh, Surfel, Wide-baseline, Free-Viewpoint Video

Fig. 1. 3D surfaces corresponding to the foreground elements of two frames in a multi-view video sequence captured by 16 calibrated cameras. The original data can be downloaded from the 4D Repository [2]

1. INTRODUCTION

In recent years, significant research effort has focused on the exploitation of the richer visual information available in multi-view video-settings. Two categories of applications exploit the new techniques derived from this on-going work *free-viewpoint video* (FVV) [25, 2, 16] and tele-presence [1, 23]. In the first category, the available visual information from each camera is conveniently fused to a 3D structure from which to generate new (uncaptured) views. In the latter, a more realistic perception of face-to-face communication is obtained, allowing free mobility in virtual conferencing spaces.

Some techniques in the state of the art already make possible to exploit multi-view data in a variety of scenarios with a conveniently high density of input views [9]. However, a more usable scenario for today's computing capabilities is that of a controlled environment, like a studio, where the background is static or slowly varying and we can focus on the dynamic foreground elements of the scene [8]. The reason is that, in such a scenario, the background might be either replaced by a virtual one or reconstructed by means of other techniques or even other technologies, like range scanners [7], thus allowing for sparser multi-camera settings.

In general, two types of approaches might be considered for processing multi-view data. In image-based multi-view

This work has been partially supported by the Spanish Ministerio de Ciencia e Innovación, under project TEC2010-18094.

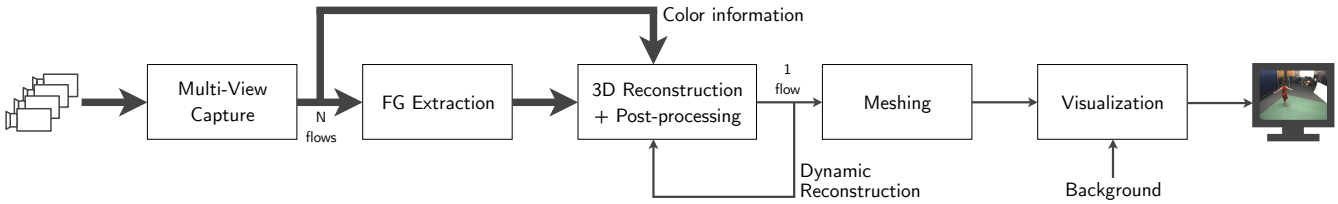


Fig. 2. Proposed free-viewpoint video system for dynamic scenes where a static, known background can be replaced

video rendering [16, 25], image data and, possibly, additional geometric data such as camera calibration parameters are used in order to generate novel views interpolating in space and/or time the contents of the images captured from the viewpoints of the multiple cameras. An advantage of this approach is that, by assuming a small enough baseline between cameras, even the original background can be interpolated in the novel views. In practice, however, this is only usable for generating novel viewpoints in very limited ranges, since current computing capabilities do not allow to use arbitrarily large numbers of input views and the highly dense multi-view setting must therefore be placed in a small solid angle.

Alternatively, reconstruction-based approaches [18, 2] attempt to recover the 3D structure of the scene by exploiting image and calibration data and possibly other types of image features, like automatically extracted foreground silhouettes [20]. Knowledge about the actual 3D structure of the scene can be useful for the addition of special effects in content creation. Another advantage of this approach is that arbitrary viewpoints can be considered beyond those covered by interpolation between cameras. Of course, virtual viewpoints very far away from the original or large zooms may lead to potential degradation of the quality of the rendered scene.

Multi-view reconstruction techniques, particularly the so-called *multi-view stereo* [19], are capable of obtaining results rivaling with those achievable with range-scan techniques, however their computation times and restricted baseline might make them difficult to introduce in practical scenarios. One possible alternative when the information of interest of the scene is the dynamic foreground, is to obtain silhouette-consistent reconstructions [8]. The advantage of this approach is that it focuses on a smaller scene volume. A potential drawback is the introduction of an additional source of errors from the automatic foreground silhouette extraction stage.

1.1. Approach

In order to provide FVV and tele-presence for dynamic scenes captured by multi-camera settings with static or known background, we propose a system based on a surface reconstruction scheme like the one depicted as a block diagram in Fig. 2. In this paper, we will not deal with multi-view capture and automatic foreground (FG) extraction (first two blocks), and will focus on the 3D reconstruction and rendering steps.

About the capture system, when restrictions do not allow obtaining small baselines between views, silhouette information proves as a sufficient visual cue for retrieving a good estimate of the actual surfaces of FG objects. This holds as long as the available views are sufficiently dense and 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. About FG extraction, some previous works [10, 18] show how the accuracy of FG silhouettes can be improved in multi-view scenarios by exploiting redundancies among views. Finally, we should add that calibration information is obtained offline using [4].

However, automatic silhouettes are still prone to errors. Therefore, the 3D surface reconstruction scheme that will be presented in Section 2 provides a mechanism for enhanced robustness, providing *conservative* surface estimates represented as dense sets of oriented, colored 3D points (also called *surface patches* or *surfels* in the literature [12]) in multi-view video sequences. The next block in our system, presented in Section 3, provides continuous surfaces out of the discrete representation provided by the previous stage, represented as polygon meshes. Finally, a suitable visualization module, an example of which is presented in Section 4, provides FVV or tele-presence capabilities and adds background information and special effects to the dynamic scene, if requested.

2. SURFACE RECONSTRUCTION

In order to obtain a set of oriented 3D points lying on the surfaces of FG objects, we interpret the reconstruction problem as a search for 3D locations that produce convenient feature matches between views. This strategy, as shown in this section, can be efficiently implemented if we refer to Montecarlo methods [6] in order to define the search strategy, exploiting the spatial correlation on the location of surface points.

In outline, the algorithm works as follows: given a set of (noisy) FG silhouettes corresponding to the images captured at a certain time instant, a set of randomly generated 3D points will be chosen as lying on the surface based on the result of a cost function defined over the silhouettes. This cost function takes into account the possibility that silhouettes can 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 the p_c belongs to the foreground or not. 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 indicates wether the 3D point belongs to the contour of at least one of the silhouettes. In order to enhance the robustness of the cost function in presence of segmentation errors, we define a relaxation of this cost function as

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

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 the point does not belong to the silhouette-consistent volume.

By producing small 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 small local curvature at small scales and extracting the normal as the director vector with its sign set such that, when projected, it points out of the silhouette for at least one view.

In practice, attempting to find a dense set of surface points in this manner is highly inefficient, due to the possibly large search volume. In the following, we introduce the whole method with improved efficiency. An initial scouting procedure, out of which a sparse set of 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.

2.1. Multi-resolution scouting

Analyzing the second term in the cost function described by Equation 1, it can be observed that a decrease of the resolution of the images will favor the detection of (low-resolution) surface points. This comes from the fact that the volumetric regions in space corresponding to the back-projection of a pixel are larger and, therefore, the ratio between the spatial regions projecting onto silhouette contour pixels and the rest of the working space increases, making the random scouting more efficient.

Considering a multi-view setting such that each of the N_v cameras approximately captures the whole space to be reconstructed from a different viewpoint, the probability of randomly finding a surface point will be proportional to the ratio between the number of contour pixels and the total number of pixels in every image multiplied by the number of views. Given images with resolution $\propto \rho^2$, the silhouette contour will have a length $\propto \rho$. Therefore, the joint probability is $\propto N_v/\rho$. Therefore, decreasing the resolution by a factor δ produces an increase of the probability of finding a (low-resolution) surface point of the same factor.

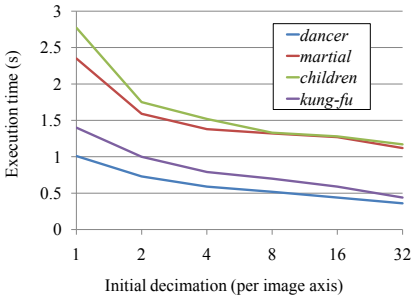


Fig. 3. Total reconstruction time vs. initial level of image decimation in multi-resolution scouting. A decimation of 1 is equivalent to the default scouting with full-resolution images

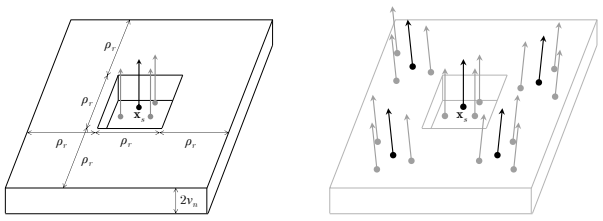
Then, for each low-resolution surface point, a conveniently sized search region (a ball inscribing the low-resolution 3D region corresponding to the contour pixel) is used in order to search the high-resolution surface point, projecting onto the high-resolution silhouettes. Once the final sparse set of surface points is obtained, the algorithm proceeds with the propagation stage presented in Subsection 2.3. In Fig. 3, the total computation time for four different sequences downloaded from [2, 15] is plotted as a function of δ, showing the advantage of using multi-resolution scouting.

2.2. Dynamic scouting

When processing video sequences, valuable information about the scene contents is known from the computation at the previous time instant. Assuming a limited range of motion between consecutive time instants parameterized by a spatial displacement r_d, we can improve the costly scouting stage by setting small search areas (spheres of radius r_d) around each of the initial sparse surface points from the previous time instant. Once the new sparse set of surface points has been obtained with this method, the propagation stage proceeds obtaining the dense surface sampling.

Some achievements have been made in 3D motion estimation by separating the shape and motion estimation tasks, in a *scene flow* estimation approach of the problem using for example voxel colors [22] as matching features. However, in such approaches the shape estimation problem is treated statically, without exploiting temporal redundancies in order to speed up the reconstruction, which is the goal of our proposed scouting strategy.

In Table 1, the resulting total computation times for the same four sequences as in Fig. 3 are shown, using the normal or blind scouting and the dynamic scouting strategy. The results do not need much justification, since it is clear that a



(a) (b)

Fig. 4. (a) Rectangular propagation region around a surface point $\mathbf{x}_s$ enclosing the space where the probability of finding a new surface point is assumed to be uniform. (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

reduced search space around regions very likely containing surface points (due to spatial continuity of surfaces immersed in 3D space) must naturally reduce the required number of random attempts for finding a surface point.

2.3. Fast propagation

Under the assumption that surfaces are locally flat, spatial correlation can be exploited in order to find dense sets of surface points out of the sparse set of surface points uniformly distributed over the surfaces that are obtained by applying any of the scouting approaches presented before.

The main challenge for this stage is to define a search region that actually permits reducing the number of operations required for obtaining a surface sample. In fact, our goal is to locally model the local distribution of the surface.

Given an oriented surface point, at an average distance $2\rho_p$ of its immediate neighbors, we model the local surface distribution as uniform in a spatial region like the shown in Fig. 4. Note that the orientation of the surface point is required in order to correctly model the local distribution of the possible locations of close surface points.

For each existing oriented surface point, 4 new oriented surface samples are found in the mentioned distribution after a much smaller number of tries than the one required to generate the initial sparse representation. Then, in order to further improve the efficiency of the algorithm, the procedure proceeds at decreasing scales. More concretely, we reduce the size of the propagation region by one half at every iteration. The algorithm stops when the desired number of surface samples N_p has been found.

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 1. Dynamic scouting. Total computation time (scouting+propagation) with normal and dynamic scouting

2.4. Post-processing

In order to both improve and complement the geometric information about the surface obtained with the method introduced above, three post-processing stages are cascaded at the output of the reconstruction method. These three methods follow the same design already introduced in [17].

These three stages are: (1) a refinement of the orientation estimation (by using larger neighborhoods in order to fit a plane 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 and including the average point of a local neighborhood while being subject to the 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 each surface point is visible.

By applying these post-processing methods we are, in first place, effectively smoothing the surface, making the detection of 3D points robust against geometric errors due to the discretization of the input images; in second place, we are adding valuable color information that provides a complete and compact description of the reconstructed surfaces.

3. 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 [13]; on the other, by choosing a proper primitive for describing elemental surface patches, such as triangles, standard rendering pipelines can be used for quickly rendering the reconstructed surfaces form any desired viewpoint.

In this section, we present a fast, greedy algorithm that follows in the category of propagation-based meshing algorithms, a reference of which is the *Ball-Pivoting Algorithm* (BPA) [3]. The major difference of our proposed method, apart from using a different set of rules for guiding the propagation while keeping topological correctness, is the fact that, by using an efficient and more flexible method for making spatial queries (nearest neighbors from a kd-tree), our proposed method does not require the iterative application of the propagation method at different scales, thus avoiding the topological problems derived from it.

The algorithm that follows is applied iteratively until at least ΔN_p surface points have been added to the continuous surface, with $\Delta = 0.9$ 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

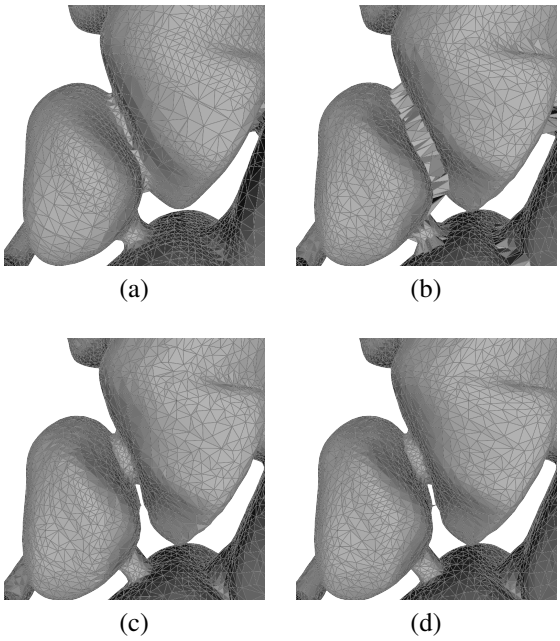


Fig. 5. Detailed views of meshing results using (a) Poisson Reconstruction, (b) BPA and (c) our proposed method, obtained from the oriented vertices of (d) a reference ground-truth mesh

of surface points; on the other, it provides new propagation directions for areas that weren't 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 the edge extracted from the queue to a suitable surface point (either a point part of the current 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 have already been introduced in [17], are defined in order to prevent the surface from folding, by restricting propagation in the forward direction, and from creating two close, overlapped layers.

The algorithm includes a mechanism for detecting and correcting topologically incorrect configurations of triangles. Consisting in removing edges shared by more than two triangles as well as the sharing triangles and proceeding with the propagation from the resulting contour. This was, in our experiments, the only source of erroneous topological configurations, being the tessellated surface a 2D manifold in 3D space.

As a difference with volumetric regularization-based meshing algorithms (Poisson reconstruction [14] being perhaps the best known example), our method is not designed to be robust with respect to 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 computation time.

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

Table 2. Performance of two state-of-the-art meshing algorithms and the proposed one with some datasets from the Stanford 3D scan repository [21]. Time is measured in seconds, memory usage in MB and RMS is the *Root Mean Square Hausdorff Distance* with respect to the ground-truth mesh

Two state-of-the-art meshing algorithms, BPA and Poisson Reconstruction, both available in Meshlab [5], and the proposed method have been used for obtaining continuous surfaces out of several datasets available in the Stanford 3D scan repository [21], for which reference meshes are provided. These are used as the ground-truth in order to compare the accuracy of each of the tested methods. Besides, the computation time and memory usage of each method is also measured. As it is shown in Table 2, the proposed method is consistently more accurate and faster than the references in the state-of-the-art, whereas the memory usage is close to that of the BPA. In Fig. 5, detailed views of the results obtained with each of the three compared methods can be compared visually to a ground-truth mesh.

4. EXPERIMENTS

In this section we first compare the proposed surface-based representation to state-of-the-art techniques in multi-view reconstruction and we show that the proposed methodology is highly competitive in the target scenario (static background and wide-baseline setup).

In the second part of our experiments, we present a Free-Viewpoint Video application that benefits from using the proposed 3D representation for multi-view video sequences. More concretely, we show how interactive FVV with a suitable quality level can be achieved by choosing a sufficient surface sampling density for the surface-based representation of the foreground elements of the multi-view video scene.

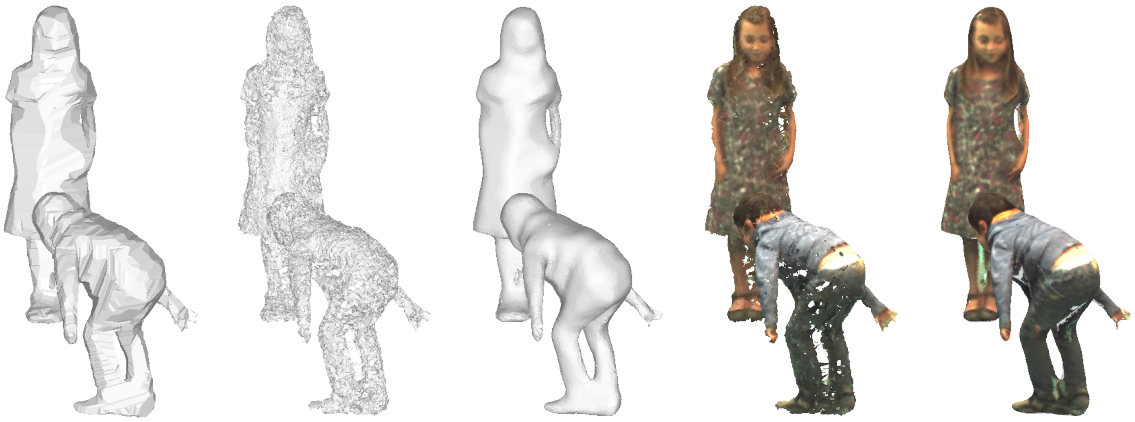


Fig. 6. Reconstructed surface of a frame in a multi-view sequence with 16 views. From left to right: first, geometry obtained with EPVH [8]; second, PMVS2 [9] concatenated with BPA [3]; and third, the proposed method; fourth, per-vertex colored surfaces obtained with PMVS2+BPA; and fifth, with the proposed method. Please note that PMVS2 is capable of working without silhouette information in small-baseline setups; the obtained results just reflect that it is not tailored to the conditions of our target scenario

Fig. 7. Results obtained with different sequences from the 4D Repository and the Kung-fu Girl sequence, using the proposed method with 100000 surface points ($\sim$4fps)

4.1. Validation

We validate the proposed method by qualitatively comparing its performance to that of two state-of-the-art methods, *Exact Polyhedral Visual Hull* (EPVH) [8] and *Patch-based Multi-View Stereo* (PMVS2) [9], which are typically used in wide-baseline setups by extracting foreground silhouettes and small-baseline setups with optional silhouettes, respectively. Fig. 6 shows how the proposed method produces a surface geometry that has a more visually appealing look than the one that can be obtained with EPVH, which is geometrically exact but does not include any regularization or smoothing. This would be very difficult to apply on the EPVH surfaces, due to the irregular shape of the mesh triangles.

Multi-view stereo methods, like PMVS2, are not tailored to a target scenario with a very wide-baseline setup, and usually require a high amount of texture to be present in the input images. As a result, we can assess that our method compares also favorably to this type of state-of-the-art approach, although we must remark that the application of our method would not be possible in more general setups, where background subtraction is not possible, whereas multi-view stereo techniques succeed, granted they have a small baseline. In terms of computation time, the surfaces extracted with PMVS2 required 28 minutes per frame (on a Intel Xeon 3 GHz), whereas our algorithm is capable of extracting a denser set of surface points at a frame rate $\sim$4 fps. In our current OpenCL implementation, the details of which are not included in this paper, we achieve real-time, with speed-ups between $4\times$ and $8\times$.

In Fig. 7, sample results for the four sequences used in the experiments in Section 2 are shown. Two of these sequences (*martial* and *children*) are captured by 16 cameras, whereas *kung-fu* is rendered from 25 virtual viewpoints and *dancer* is captured by just 8 cameras. In all cases, the method is capable of providing visually appealing surfaces that suffice for a correct representation in a FVV application. We observe that the marginal cost of adding a new view to the reconstruction is sub-linear, due to the correct exploitation of the spatial correlation, which translates in good scalability for larger multi-view settings.

4.2. Application

We focus on a Free-Viewpoint Video application, where the contents of interest of multi-view video sequences (the dynamic foreground) are streamed to a visualization application,

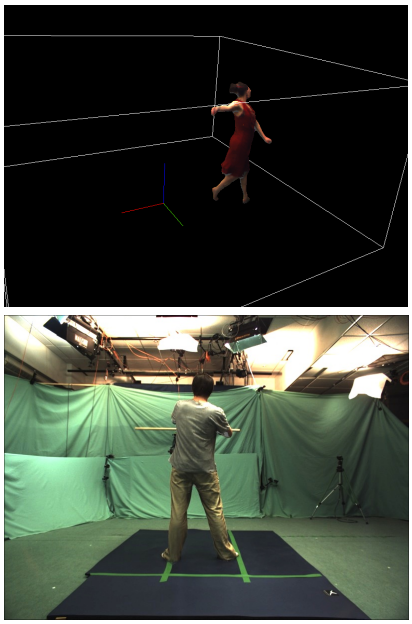


Fig. 8. Top: novel viewpoint of an instant of a scene captured by 8 cameras with a virtual background. Bottom: overlay of a surface reconstructed from 16 views projected onto one of the original viewpoints and the original background.

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.

In Fig. 8, two possible uses of our free-viewpoint viewer are shown. At the top image we have captured an instant of the *dancer* sequence from a novel viewpoint using a virtual background. At the bottom in the same figure, we are projecting a reconstructed surface from another scene from the 4D Repository onto one of the original viewpoints and overlaying it with the corresponding original background. A clear advantage of this representation is that, by sending the background once per sequence and the 3D surface once per frame, we can retrieve the complete N_v views in every time instant.

4.3. Required density of surface points

One question that arises is which is the required density of surface points in order to obtain a good quality representation. In our experiments, we compare the PSNR of the reprojection of the reconstructed surfaces on the view with worst quality against the number of pixels in such view (which in all of our experiments coincided with the view with larger number of foreground pixels). Normalizing these figures for the sequences *dancer*, *children* and *martial*, we obtain the curves in Fig. 9, which show a saturation on the achievable accuracy.

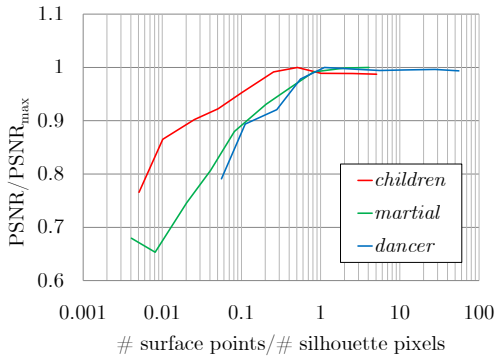


Fig. 9. Relative PSNR with respect to the maximum in sequences from the *dancer*, *children* and *martial* datasets as a function of the ratio between the number of surface samples and the number of foreground pixels in the view where they take most image pixels

As a consequence of this empirical result, a number of surface samples of around twice the number of pixels occupied by the foreground 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 a quasi-cylindrical symmetry of the 3D objects.

Table 3 shows how the proposed representation is a suitable tool for streaming the contents of multi-view video for real-time applications. We compare the amount of data required to represent multi-view sequences in a classical image-based manner (either as individual frames or exploiting temporal redundancies as lossless video sequences) against our approach. The bandwidth or storage resources for transmitting or storing the contents of multi-view video sequences with static background can be notably reduced in our case by introducing a controlled amount of losses. In the table, FG CTM stands for compressing the reconstructed colored surfaces using OpenCTM [11], which reduces in a factor of about 10 the required data support, while PSNR figures in the worst case view are still well above 45 dB. An alternative approach implementing FVV using standard video coding tools can be found in [24].

5. CONCLUSIONS

To sum up, we have presented a complete methodology for extracting and visualizing an alternative representation of the contents of interest in multi-view video sequences. More concretely, we have presented a fast method for multi-view reconstruction with arbitrarily wide baseline settings. This method provides an efficient representation of the dynamic part of the

scene when the static background is known or replaceable. We also provide a rule of thumb for obtaining accurate reconstructions in an economical manner.

By exploiting the parallelized design of the algorithm, it can be easily implemented on a GPU (using for example OpenCL) providing real-time reconstructions of 3D surfaces. This is currently used for interactive applications in our *smart room*, including free-viewpoint video.

For the future work, we are planning to make a better exploitation of the color information available in the images in order to obtain quality levels beyond the current limits of the silhouette-consistent reconstruction. Last but not least, the authors would like to thank the anonymous reviewers for their constructive review, which resulted in an improved manuscript.

6. REFERENCES

[1] 3D Presence. Seventh Research Framework Programme. http://www.3dpresence.eu/, 2011.

[2] 4D Repository. Inria. http://4drepository.inrialpes.fr/, 2011.

[3] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin. The ball-pivoting algorithm for surface reconstruction. *Visualization and Computer Graphics, IEEE Transactions on*, 5(4):349–359, Oct-Dec 1999.

[4] J.Y. Bouguet. Camera calibration toolbox for matlab, 2000.

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

[6] Nando De Freitas, Arnaud Doucet, and Neil Gordon. *Sequential Monte Carlo Methods in Practice*. Statistics for Engineering and Information Science. Springer, New York, NY, 2001.

[7] Faro. Range Scanners. http://www.faro.com/, 2011.

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

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

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 Rae	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

Table 3. Comparison of the data required to represent multi-view sequences in a classical image-based manner, either as individual frames or exploiting temporal redundancies as lossless video sequences, and using our approach while keeping the background as a set of PNG images

[10] Jaime Gallego, Jordi Salvador, Josep R. Casas, and Montse Pardàs. Joint multi-view foreground segmentation and 3d reconstruction with tolerance loop. In *Proc. IEEE ICIP*, 2011.

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

[12] Markus Gross, Stephan Würmlin, Martin Naef, Edouard Lamboray, Christian Spagno, Andreas Kunz, Esther Koller-Meier, Tomas Svoboda, Luc Van Gool, Silke Lang, Kai Strehlke, Andrew Vande Moere, and Oliver Staadt. blue-c: A spatially immersive display and 3d video portal for telepresence. In *ACM Trans. on Graphics*, pages 819–827, 2003.

[13] Kai Hormann, Bruno Lévy, and Alla Sheffer. Mesh parameterization: Theory and practice. In *ACM SIGGRAPH Course Notes*, 2007.

[14] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Eurographics Symp. on Geometry Processing*, pages 61–70, 2006.

[15] 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.

[16] Christian Lipski, Christian Linz, Kai Berger, and Marcus Magnor. Virtual video camera: image-based viewpoint navigation through space and time. In *SIGGRAPH '09*, page 93:1, 2009.

[17] Jordi Salvador, Xavier Suau, and Josep R. Casas. From silhouettes to 3d points to mesh: Towards free viewpoint video. In *Proceedings of the 1st International Workshop on 3D Video Processing*, 3DVP '10, pages 19–24. ACM, 2010.

[18] Muhammad Sarim, Adrian Hilton, Jean-Yves Guillemaut, Hansung Kim, and Takeshi Takai. Wide-baseline multi-view video segmentation for 3d reconstruction. In *Proceedings of the 1st international workshop on 3D video processing*, 3DVP '10, pages 13–18, New York, NY, USA, 2010. ACM.

[19] 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 CVPR*, volume 1, pages 519–528, 2006.

[20] C. Stauffer and W. E. L. Grimson. Adaptive background mixture models for real-time tracking. In *Proc. IEEE Conf. on CVPR*, volume 2, pages 252–259, 1999.

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

[22] S. Vedula, S. Baker, and T. Kanade. Image-based spatio-temporal modeling and view interpolation of dynamic events. *ACM Trans. on Graphics*, 24(2):240–261, 2005.

[23] Vision. CENIT Project of the Spanish Ministry of Industry, Tourism and Commerce. https://www.cenit-vision.org/, 2010.

[24] Stephan Würmlin, Edouard Lamboray, Michael Waschbüsch, Peter Kaufmann, Aljoscha Smolic, and Markus Gross. Image-space free-viewpoint video. In *Proceedings of Vision, Modeling, Visualization (VMV)*, 2005.

[25] C. Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. *ACM Trans. Graph.*, 23:600–608, August 2004.