RECONSTRUCTION FOR 3D IMMERSIVE VIRTUAL ENVIRONMENTS

D. S. Alexiadis[⋆], *G. Kordelas* [⋆†], *K. C Apostolakis* [⋆], *J. D. Agapito*[‡], *J. M. Vegas*[‡], *E. Izquierdo* [†], *P. Daras*[⋆]

[⋆] Centre for Research and Technology - Hellas, Informatics and Telematics Institute, Thessaloniki, Greece
[†]Multimedia and Vision Group (MMV), Queen Mary University, London, UK
[‡]Computer Science Department, University of Valladolid, Valladolid, Spain

ABSTRACT

The future of tele-conferencing is towards multi-party 3D Tele-Immersion (TI) and TI environments that can support realistic inter-personal communications and virtual interaction among participants. In this paper, we address two important issues, pertinent to TI environments. The paper focuses on techniques for the real-time, 3D reconstruction of moving humans from multiple Kinect devices. The off-line generation of real-life 3D scenes from visual data, captured by non-professional users is also addressed. Experimental results are provided that demonstrate the efficiency of the methods, along with an example of mixing real with virtual in a shared space.

1. INTRODUCTION

With current tele-conferencing systems, although participants can talk to each other as if they were in the same location, they remain captives in a 2-D screen projection. The future of tele-conferencing systems towards realistic inter-personal communications is multi-party 3D Tele-Immersive (TI) environments [1]. Three-dimensional TI environments, based on the idea of generating realistic 3D representations of users in real-time and placing them inside a shared virtual space or even inside real 3D reconstructed scenes (Fig. 1), have a great potential to promote collaborative work among geographically distributed users.

In this paper, some important issues, pertinent to TI environments, are addressed. The paper focuses on techniques for the real-time, realistic 3D reconstruction of moving humans (section 2), as well as the off-line generation of real-life 3D scenes from visual data, captured by non-professional users (section 3). Robust, fast and accurate generation of full 3D data from real-life scenes, is still a challenging task, especially considering the high frame-rate demands of TI systems. Although many accurate 3D reconstruction methods exist in the literature (e.g. [2]), these are not applicable in TI applications, due to very high required computational time. Only a few, mainly visual hull-based methods (e.g. [3]), are quite fast (near real-time), but lack the ability to reconstruct concavities. On the other hand, most of the methods exploited in real-time TI applications focus mainly on synthesizing intermediate views for user-given viewpoints, rather than producing complete 3D models. In this work, real-time full 3D reconstruction is achieved by the fusion of depth-maps, captured by multiple Kinect devices. Since the release time of Kinect (Nov.2010) only a few relevant published works can be found (e.g. [4]), non of them focusing on the real-time reconstruction from multiple Kinects.

This work was supported by the European Commission under contracts FP7-247688 3DLife and FP7-287723 REVERIE.

Fig. 1. Distant users, sharing a virtual world.

The paper is organized as follows. In section 2, methods for real-time 3D reconstruction of moving humans are shortly presented. In section 3 the off-line reconstruction of real-life 3D scenes from non-professional user data is addressed. Additional issues are discussed in section 4, before concluding.

2. REAL-TIME, FULL 3D RECONSTRUCTION

The final objective here is the real-time construction of a single, full 3D textured mesh from the depth data, captured simultaneously by multiple Kinect devices.

The used capturing system is composed of $M = 4$ Kinect sensors, connected on a single host PC that features an Intel i7 processor and 4GB RAM, along with a CUDA-enabled GPU NVidia GTX 560 Ti. All computation times, presented in the paper correspond to these characteristics. The Kinect devices are horizontally positioned at the vertices of an imaginable square, with a diagonal of length approximately 3 meters, all pointing to the center of the working volume. An external calibration procedure was employed to estimate the orientation and position of each Kinect device with respect to a reference one. The employed approach uses a hand-made calibration

bar, composed of two LEDs at a fixed distance, and constitutes a combination of the methods in [5] and [6]. Due to space limitations, the reader is referred to [5, 6]. The achieved mean re-projection error is less than 0.8 pixels. Additionally, one may optionally use an on-line registration approach, in order to refine the alignment data, every few frames. In this case, a global all-to-all registration approach is required, that simultaneously registers multiple point-clouds. In this work, we exploited the global registration approach of [7], which embeds the generalized procrustes analysis into an Iterative-Closest-Point (ICP) framework.

2.1. Real-time explicit fusion of multiple Kinect data

The overall approach, an adapted version of the mesh-zippering algorithm of [8], is described in Fig. 2. The approach can be summarized as follows. (a) Initially, $M = 4$ separate meshes $\mathcal{A}_m, m = 1 \dots M$ are generated via soft-thresholding of the depth maps (to keep only the foreground object) and Step Discontinuity Constraint Triangulation (Fig. 3(a)); each mesh corresponds to a single Kinect device. The generated separate meshes present redundancy and overlap to each other. Therefore, (b) the method continues with the decimation/removal of the redundant, overlapping mesh regions, working in pairs of adjacent meshes. Finally, (c) a "clipping" step, which is based on the detection of the adjacent mesh regions and local constrained Delaunay triangulation, "stitches" together the multiple meshes and produces the final single mesh. Reconstruction results are given in Fig. 3(b).

The second step (b) of the method, iteratively decimates redundant triangles from the boundaries of two meshes, until no overlapping triangles are present. Due to its iterative nature, this algorithmic part constitutes the bottleneck part of the whole reconstruction approach and the achieved reconstruction time is of the order of a few seconds. To obtain faster reconstruction rates, two important things have to be taken into account: In each iteration, in order to decide whether a triangle is redundant or not, its distance to the closest triangle on the other mesh has to be found, which normally requires a full search among all triangles of the other mesh. However, given that each 3D triangle is associated with a 2D triangle on the depth image plane, the search can be performed on the image plane and limited in a local 2D region. Additionally, a coarse-to-fine strategy can further speed-up the the underlying algorithmic step, based on the fact that each triangle in a specific scale is associated with four triangles in the next finer scale. With these two modifications of the algorithm, the reconstruction time can be reduced below 1sec. Finally, in order to speed up the algorithm's execution and realize higher frame rates, some of its parts were implemented exploiting the CUDA (Compute Unified Device Architecture) parallel computing architecture. Independent pixel-wise calculations can be mapped into many parallel blocks of independent processing threads, run on the GPU. With such a GPU implementation, the algorithm runs at frame rates close to 8fps.

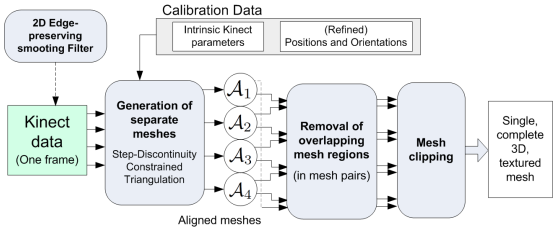

Fig. 2. Explicit fusion of multiple depth-maps.

(a) (b)

Fig. 3. (a) The idea of Step Discontinuity Constraint Triangulation: For each 2×2 neighborhood on the depth-map, two triangles are generated, unless the absolute difference of the neighbor depth values is greater than a predefined threshold. (b) Explicit fusion method: Results from various viewpoints.

2.2. Real-time implicit fusion of multiple Kinect data

Following a different approach, the data from multiple Kinect sensors can be fused in an implicit way, based on the notion of volumetric Truncated Signed Distance (TSD) Function [9]. The approach is schematically described in Fig. 4 and can be summarized as follows. (a) For each frame, after soft-thresholding of the depth maps, the 3D bounding box $\mathcal{BB}$ of the foreground object is estimated and the discrete volumetric space $\mathbf{X} = (X, Y, Z) \in \mathcal{BB}$ is considered. (b) Each voxel $\mathbf{X}$ is projected on each depth-map $m = 1 \dots M$ and using the maps' values the correpsonding "actual" 3D points $\mathbf{X}_m^a = (X_m^a, Y_m^a, Z_m^a)$ are found. (c) The SD functions are calculated: $\text{SD}_m(\mathbf{X}) = \text{sgn}(Z_m^a - Z) \cdot d(\mathbf{X}, \mathbf{X}_m^a)$ where $d(\mathbf{X}, \mathbf{X}_m^a)$ is the Euclidean 3D distance between $\mathbf{X}$ and $\mathbf{X}_m^a$. The SD functions are truncated, according to: $\text{SD}_m(\mathbf{X}) = \text{NaN}$, if $|\text{SD}_m(\mathbf{X})| > \mu$, where NaN stands for "not defined value" and μ is a threshold, selected equal to 2.5cm in our experiments, based on the expected accuracy of the Kinect depth measurements and the external calibration accuracy. (d) A combined volumetric SD function is calculated from the weighted combination of the separate functions: $\text{GSD}(\mathbf{X}) = \frac{\sum_m w_m(\mathbf{X}) \cdot \text{SD}_m(\mathbf{X})}{\sum_m w_m(\mathbf{X})}$. The calculation of the appropriate weights is discussed in the next paragraph. (e) The extraction of the isosurface $\text{GSD}(\mathbf{X}) = 0$, using Marching Cubes [10], produces the final complete 3D mesh.

The weight $w_m(\mathbf{X})$ should depend on the angle $\theta_m(\mathbf{X})$ between the line connecting $\mathbf{X}$ with the m-th camera and the surface normal vector at point $\mathbf{X}$. Therefore, a surface normal map is calculated from each depth map, during an ini-

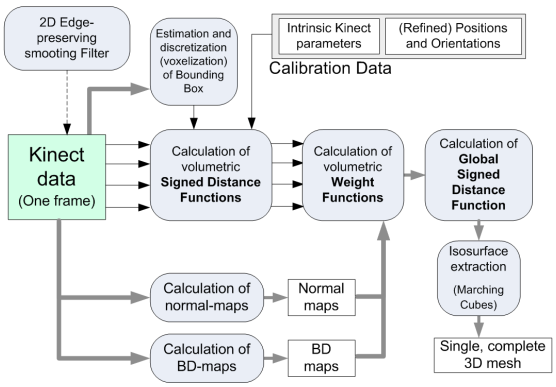

Fig. 4. Implicit fusion of multiple depth-maps.

tial stage of the algorithm. Additionally, experiments showed that the Kinect depth measurements near the boundaries of the captured foreground object are quite noisy. Therefore, for each depth map an associated distance-to-background map (notated as BD map in Fig. 4) is calculated: $b_m(x,y) = \max\{d(x,y)/d_0, 1\}$, where $d(x,y)$ is the distance of pixel (x,y) to the background pixels and $d_0 = 20$pixels a predefined parameter. The total weight is calculated as follows: $w_m(\mathbf{X}) = \cos(\theta_m(\mathbf{X})) \cdot b_m(\Pi_m(\mathbf{X}))$, where $\Pi_m(\mathbf{X})$ stands for the projection of $\mathbf{X}$ on the m-th depth-map plane.

Some reconstruction results are given in Fig. 5, with the discretized 3D space consisting of $2^7 \times 2^8 \times 2^7$ voxels. This results into voxels of size 0.5^3cm^3, considering a reasonable size of the object's (human's) bounding box, such as $2^6 \times 2^7 \times 2^6$cm^3. The described volumetric algorithm is suitable for a CUDA-based parallel computing implementation, since most of its stages involve pixel-wise or voxel-wise calculations. With such a GPU implementation, the algorithm runs at frame rates close to 10fps. Compared to the zippering approach of subsection 2.1, the volumetric approach has the advantage of performing implicitly the fusion, resulting in a higher robustness (especially in areas of high surface curvature) and small gaps' filling near the boundaries. On the other hand, considering a fixed number of voxels (almost fixed computational time), the voxel size increases for large volumes (bounding boxes), resulting into quality degradation.

3. OFFLINE RECONSTRUCTION OF REAL SCENES

A prerequisite step for the dense 3D reconstruction of a scene from unstructured user generated data is to recover the structure of the scene. Bundler framework [11] provides an ideal solution for this purpose. This structure-from-motion approach is capable of computing accurately the 3D poses of the cameras that captured a scene, as well as their intrinsic and extrinsic parameters. Figure 6(a) depicts the recovered cameras poses and the sparse geometry of a scene captured in a dataset of 14 images.

In order to handle the input visual dataset, the images

Fig. 5. Implicit fusion: Results for two frames.

are organized into stereo pairs, where pairs are comprised by images whose cameras positions have the minimum baseline distance. The availability of cameras information offers a wide range of choices that could undertake the 3D reconstruction task. Two approaches are considered in this paper, in order to generate high-resolution depth-maps for each stereo pair. The first one exploits the methodology described in [12]. This work revisits the geometry of sweeping and a spherical parametrization of the sweeping surface is proposed and evaluated against plane sweeping. Experimental results proved that spherical sweeping attained more accurate reconstructions, while preserving the time efficiency of plane sweeping. This approach is able to generate fast 3D point clouds of the scene per stereo pair. By back-projecting the generated 3D point cloud on one of the stereo-pair images, a depth map is obtained.

The second approach uses dense feature matching to compute the disparity between image pairs, searching for similarities across epipolar lines. Daisy descriptor [13] is selected for dense feature matching, since its performance is better in terms of accuracy against other prevalent descriptors. This approach is used to compute the disparity map per stereo pair, which is then translated into the corresponding depth-map.

The depth maps, generated by one of the above described methods, are combined into a single 3D mesh using the volumetric - implicit fusion method of section 2.2. The visual results indicated that, though the first approach is fast, it is not very accurate and needs to be followed by an approach that uses multiview visibility reasoning to refine the reconstruction result. On the other hand, the second method sacrifices time efficiency for reconstruction accuracy. Reconstruction results obtained from this method are given in Fig. 6(b), as well as in Fig. 1.

4. REAL WITH VIRTUAL - ADDITIONAL ISSUES

3D representations of distant users, captured and generated at each user's site can be (mesh-) coded and transmitted to a central server site, where the users' representations are placed inside a shared virtual world. The architecture is scalable, in the sense that the accuracy of the users' representations, generated and transmitted, depends on the user's available capturing setup, as well as the network capacity. Studying appropriate coding techniques and network architectures is beyond the scope of this paper. The shared world is enriched with

Fig. 6. (a) Camera positions as obtained by the bundler framework [11]. (b) Reconstructed dense mesh from 4 stereo pairs.

Fig. 7. Animated dancing avatar and a reconstructed human.

detailed 3D reconstructions of real large-scale static scenes, generated by the methods of section 3. Finally, autonomous or controlled virtual characters (avatars) can participate in the action within the shared space. An example is given at Fig. 1.

A low-poly, three-dimensional human character (avatar) was hand-modeled. The static mesh was complemented by a skeletal rig, consisting of 18 control structures (bones), connected to one another using rotational joints, ultimately forming a hierarchy. Animation is applied to the model by rotating specific joints, before a smooth skinning procedure calculates the bones' weighted influences on the mesh vertices and modifies each vertex accordingly. The textured model was exported to the COLLADA Digital Asset and FX Exchange Schema (DAE) format and animated in our OpenGL environment, using the joints' orientations of a human dancer [14], captured using the OpenNI skeleton-tracking API (Fig. 7). This could be useful in an online dance class scenario.

5. CONCLUSIONS

Two challenging tasks, necessary in TI applications, were addressed. More specifically, methodologies for the real-time reconstruction of moving humans, as well as the generation of realistic 3D models of real-life scenes, were presented. The experimental results demonstrated the appropriateness of the described methods. An example of mixing real with virtual in a shared virtual space was given. Future work includes the investigation and elaboration of efficient (mesh-) coding techniques that would increase the efficiency in data transmission, as well as studying appropriate methods for user activity analysis and interaction mechanisms in various scenarios.

6. REFERENCES

[1] R. Vasudevan, G. Kurillo, E. Lobaton, T. Bernardin, O. Kreylos, R. Bajcsy, and K. Nahrstedt, "High quality visualization for geographically distributed 3D teleimmersive applications," *IEEE MM*, vol. 13(3), 2011.

[2] G. Vogiatzis, C. Hernandez, P. Torr, and R. Cipolla, "Multiview stereo via volumetric graph-cuts and occlusion robust photo-consistency," *IEEE PAMI*, vol. 29(12), pp. 2241 –2246, 2007.

[3] J.-S. Franco and E. Boyer, "Efficient polyhedral modeling from silhouettes," *IEEE PAMI*, vol. 31, 2009.

[4] S. Izadi, R. Newcombe, D. Kim, O. Hilliges, D. Molyneaux, S. Hodges, P. Kohli, A. Davison, and A. Fitzgibbon, "Kinectfusion: Real-time dynamic 3D surface reconstruction and interaction," in *ACM SIGGRAPH*, 2011.

[5] T. Svoboda, D. Martinec, and T. Pajdla, "A convenient multicamera self-calibration for virtual environments," *Presence*, vol. 14(5), pp. 407–422, 2005.

[6] G. Kurillo, Z. Li, and R. Bajcsy, "Wide-area external multi-camera calibration using vision graphs and virtual calibration object," in *ICDSC*, September 7-11, 2008.

[7] R. Toldo, A. Beinat, and F. Crosilla, "Global registration of multiple point clouds embedding the generalized procrustes analysis into an ICP framework," in *3DPVT*, 2010.

[8] G. Turk and M. Levoy, "Zippered polygon meshes from range images," in *SIGGRAPH*, 1994.

[9] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," in *SIGGRAPH*, 1996.

[10] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," *Computer Graphics*, vol. 21(4), pp. 163–169, 1987.

[11] N. Snavely, S.M. Seitz, and R. Szeliski, "Modeling the world from internet photo collections," *IJCV*, pp. 189–210, 2008.

[12] X. Zabulis, G. Kordelas, K. Mueller, and A. Smolic, "Increasing the accuracy of the space-sweeping approach to stereo reconstruction, using spherical backprojection surfaces," in *IEEE ICIP*, 2006, pp. 2965–2968.

[13] E. Tola, V. Lepetit, and P. Fua, "Daisy: an efficient dense descriptor applied to wide baseline stereo," *IEEE PAMI*, vol. 32(5), pp. 815–830, 2010.

[14] D. Alexiadis, P. Kelly, P. Daras, N. O' Connor, T. Boubekeur, and M. Ben Moussa, "Evaluating a dancers performance using kinect-based skeleton tracking," in *ACM Multimedia*, 2011.