SKELETON-BASED MULTIVIEW RECONSTRUCTION

Bastien Durix, Géraldine Morin, Sylvie Chambon

IRIT, UMR CNRS 5505
Paul Sabatier University, 31000 Toulouse
firstname.lastname@enseeiht.fr

Céline Roudet, Lionel Garnier

LE2I, UMR CNRS 6306
Burgundy University, 21000 Dijon
firstname.lastname@u-bourgogne.fr

ABSTRACT

The advantage of skeleton-based 3D reconstruction is to completely generate a single 3D object from well chosen views. Having numerous views is necessary for a reliable reconstruction but projections of skeletons lead to different topologies. We reconstruct 3D objects with curved medial axis (whose topology is a tree) from the perspective skeletons on an arbitrary number of calibrated acquisitions. The main contribution is to estimate the 3D skeleton, from multiple images: its topology is chosen as the closest to those of the perspective skeletons on the set of images, which means that the number of topology changes to map the 3D skeleton topology to topologies on images is minimum. This way, the number of matched branches is maximum.

***Index Terms*—** Skeleton, reconstruction, graph-edit distance, topology

1. INTRODUCTION AND PREVIOUS WORK

In movies, video games and many other media, 3D content is more and more present and detailed. Different approaches to ease or to automate 3D generation have been proposed. First, sketching [1] converts a 2D drawing into a 3D model. Other methods require some acquisitions of an object: multiview-stereo [2] recovers a point cloud of the visible parts of an object from a set of images, but without topology; Shape-from-silhouette reconstructs the visual hull of an object [3], which is an implicit volume in which the object is. Recently, Chen *et al.* proposed a reconstruction method from a single image based, on generalised cylinders [4]. This method identifies 3D tubular objects along a guide made by the user and assumes an orthographic projection.

In this article, we propose a skeleton-based reconstruction from a set of images. A skeleton is a structure (that was first introduced by Blum) which characterizes a closed shape [5]. The skeleton of a shape is defined by the set of centers of maximal balls inside the shape (*cf.* Fig. 1), with the corresponding radius to each point. As it is a simplified representation of the shape, it is often used in shape matching [6, 7, 8, 9] and is adapted for animation [10, 11, 12].

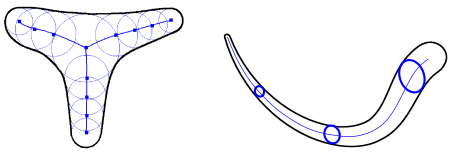

Fig. 1. Left: A skeleton is defined by the set of centers of maximal balls (blue circles), combined with the radius information. Right: A perspective skeleton is defined by the set of perspective projection of maximal spheres, which are maximal ellipses in the 2D shape [13].

Livesu *et al.* compute a curve skeleton of a known 3D object [14, 15] based on the orthogonal projection of the shape on different planes, by triangulating a set of 3D spheres (using epipolar geometry to associate the projections). The topology is then added to the set of spheres. For optimisation purpose, their method has to be used with virtual orthographic projection, but does not consider real perspective images.

In shape reconstruction, Caglioti *et al.* propose an algorithm which estimates the shape of an object that can be represented by a curve-skeleton with constant radius [16]. Their algorithm considers the perspective projection of the shape on a single image. As it is the projection of a canal surface with constant known radius, it can be recovered without ambiguity by applying an homothetic transformation to each sphere of the canal surface.

In a recent work, Durix *et al.* [13] remove the constant radius hypothesis and use two images of the same object. A projective skeleton is estimated for each image, which corresponds to the perspective image of a 3D skeleton. However, to use their method on a set of branches, each branch has to be matched manually on each image. A 3D skeleton is triangulated and a set of canal surfaces is reconstructed. Here we extend this work: first, using an interactive association of the extremities of the perspective skeletons, we estimate a topology for the reconstructed skeleton; second, to increase the accuracy, we extend the reconstruction to more than two images. The proposed method assumes that the skeleton of the shape is a connected curve-skeleton, with no cycles.

Section 2 details our main contribution, a method to match the different branches of a set of curve-skeletons and to pro-

pose a topology for the reconstructed 3D skeleton. Section 3 describes an algorithm able to reconstruct an object from an arbitrary number of images. Finally, qualitative and quantitative results are presented on a set of reconstructions in Section 4.

2. MATCHING TOPOLOGICAL SKELETONS

2.1. Determining a common skeleton

Let us start with the context of the study. We have a set of n calibrated images of the same object, and on each image the corresponding perspective curve-skeleton (*cf.* Fig. 2). We assume that the object is connected and genus-0, and that each projective skeleton is a tree. Although the perspective skeleton is, in general, the projection of the 3D object skeleton, the different projective skeletons do not necessarily have the same topology (for example, because of self-occlusions from the considered viewpoints). Thus, their branches do not match perfectly. The topology of the 3D skeleton can not be recovered exactly using 2D images, thus we assume that the acquired views represent well the topology. Under this hypothesis, we propose a method to estimate the closest topology to the projective skeletons, under the edit distance.

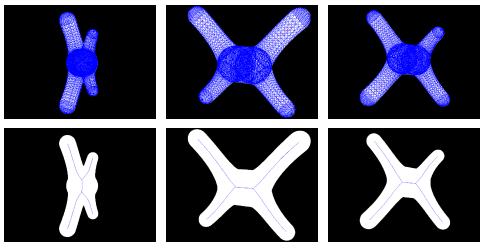

Image 1 Image 2 Image 3

Fig. 2. Three viewpoints of the same 3D object (first row), and their respective projective skeletons (second row). We see that the topology of the first projective skeleton is different from the topology of the second and third skeleton.

A lot of skeleton matching techniques exist [6, 9]. Sebastian *et al.* propose an algorithm to estimate a distance between two shapes, based on their skeleton [17]. The idea is to go from one skeleton to the other, by applying a set of successive editions: collapsing or inserting an edge, removing or adding extremities, but is limited to matching two skeletons. The number of editions is called the edit distance, and computing it is NP-hard [18]. However here, we do not compute the distance between two skeletons or graphs, as many papers in the litterature [19], but we search for a skeleton which is the closest from the set of skeletons.

2.2. Finding a closest skeleton: Transformations

Here, to avoid NP-hardness, we assume that we know the correspondences of each extremity of the skeletons. Then, based on the edit distance between two skeletons, we propose a method to compute the closest skeleton topology to a set of skeletons. For this purpose, we first describe uniquely each edge on different skeletons, based on the extremities information. By describing each edge in the different skeletons, we can then compare which appears the most, and find the skeleton the closest to the different images.

In order to do that, we first convert the skeletons to graphs such that vertices of each graph correspond to either junctions or extremities of each skeleton, and edges to skeleton branches. As a consequence, there are no degree 2 nodes. These graphs model the topology of the skeletons. Formally, each graph $G_i = (V_i, E_i)$ corresponds to the skeleton on the image i, which is a tree by assumption.

We assume that we know the associations of the m extremities (4 in Fig. 2). We have n graphs (3 in Fig. 2, represented on Fig. 3). We first give an intuition of our method to find the closest graph to all graphs of Fig. 3 and then formalize the algorithm to find it in a general setting in section 2.3.

We start with G_c a graph with m extremities, and a single central node, linked to all the extremities (*cf.* Fig. 3 for the example). We can see that inserting the edge e_{13} to the graph G_c gives a graph that is isomorphic to the graph G_1. Inserting the edge e_{23} (resp. e_{33}) gives a graph that is isomorphic to the graphs G_2 (resp. G_3). As there are 3 edge insertions (one for each graph), the total editing cost is 3 for G_c.

We can simplify the two similar insertions in the second and third graphs. To do that we modify the graph G_c, by inserting an edge e_{c3} in the same location than e_{23} and e_{33}. G_c is now isomorphic to the graphs G_2 and G_3, so no editions are needed to obtain these graphs. To obtain the graph G_1 from the graph G_c, two operations are needed: first, we have to collapse the edge e_{c3}, then to insert an edge similar to e_{13}. We now have an editing cost of 2, which is minimum as we can not simplify anymore.

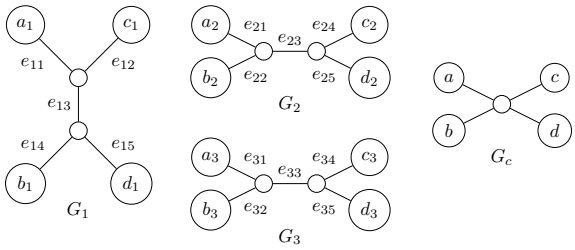

Fig. 3. G_1, G_2, G_3: Topology of the three projective skeletons of Fig. 2. Each $(a_i)_{i=1,2,3}$, $(b_i)_{i=1,2,3}$, $(c_i)_{i=1,2,3}$ and $(d_i)_{i=1,2,3}$ are matched. G_c: Example of an initial simplified graph, corresponding to the G_i.

2.3. Formalization

Here we introduce a node and edge labelling, based on extremities of a graph. We then show that we can recover the

node labels of a graph using only their edge labels. As shown in [20], two graphs are isomorphic if and only if they have the same labelling. Using the fact that we can recover the node labels using only the edge labels, we show that we can recover the set of edges which represents the graph with the closest topology.

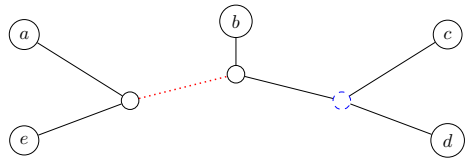

Fig. 4. Visualization of the introduced notation. The blue dashed node separates three sets of extremities: the first one contains a, e and b, the second one c and the third one d, so this node is labelled $\{\{a, e, b\}, \{c\}, \{d\}\}$. The red dotted edge separates the graph in two parts: the first one contains a and e, and the second one contains b, c and d. So the red dotted edge is $\{\{a, e\}, \{b, c, d\}\}$.

Node labelling: Each graph is a tree, so each node separates the graph into k subgraphs. Each subgraph contains a set of extremities: $A_1, \ldots, A_k$, which forms a partition of V_e, the set of extremities. Thus we label each node by the partition of V_e: $\{A_1, \ldots, A_k\}$ (*cf.* Fig. 4). An exception is made on this labelling method for the extremities of the graph: each extremity v_e will be labelled as $\{\{v_e\}, V_e \setminus \{v_e\}\}$.

Edge labelling: Each edge separates the graph into two parts, the first one $G(A_1)$ containing a set of extremities called A_1, and the second one $G(A_2)$ containing the complementary set of extremities denoted A_2. We thus characterize each edge by the pair $\{A_1, A_2\}$ (*cf.* Fig. 4), which is a partition of V_e.

These notations are unique since there is no degree two vertices. We note V_{e_i} the set of all partitions characterizing nodes of the graph G_i, and S_{e_i} the set of all pairs characterizing edges of this graph. To complete the label representation of the graph, we have to find the link between the label of an edge and the labels of its extremities [20].

Link between the labels: Let $\{A_i\}_{1,\ldots,k_1}$ and $\{B_j\}_{1,\ldots,k_2}$ be the labels associated to the extremities of an edge in the graph. We show that these two partitions are two subpartitions of the partition $\{C_1, C_2\}$ representing the edge. Let us consider the extremity labelled by $\{A_i\}_{1,\ldots,k_1}$. Each A_i represents a set of extremities, separated by the node. As the edge is connected to the node, it separates the node from one set of extremities A_i, thus we have $C_1 = A_i$ (swapping the indices if necessary). With the same reasoning, we have some j with $C_2 = B_j$. As $\{C_1, C_2\}$ is a partition of V_e, the partitions representing the nodes are subpartitions of it. Furthermore, this implies that to find the node connected to a set of edges, we only have to find the maximal subpartition of the labels of the edges. But this mean that we have to know which edges are adjacent.

Edge adjacency: The two edges share a node if and only if:

$$\forall \{C_1, C_2\} \in S_{e_i}, A_1 \subset C_1 \subset B_1 \Rightarrow C_1 \in \{A_1, B_1\}.$$

Proof: Let us suppose that there is an edge $\{C_1, C_2\} \in S_{e_i}$, with $A_1 \subsetneq C_1 \subsetneq B_1$. This implies that $B_2 \subset C_2$. We can deduce a property on the edges: $\{A_1, A_2\} \in G(C_1)$ and $\{B_1, B_2\} \in G(C_2)$. This means that they are not adjacent, since $G(C_1)$ and $G(C_2)$ do not share any node. Reciprocally, if two edges do not share any node, this means that there exists an edge $\{C_1, C_2\}$ between them, which means that $\{A_1, A_2\} \in G(C_1)$ and $\{B_1, B_2\} \in G(C_2)$ and we have $A_1 \subsetneq C_1 \subsetneq B_1$. □

Now, using the link between the labels, we can deduce that we only need the edge labels to find a graph associated to them. But all the sets of edges do not work, and there is a condition on the edge labels to be compatible, *i.e.* a condition to be in the same graph.

Edge compatibility: Let us consider two arbitrary edges in S_{e_i}, $\{A_1, A_2\}$ and $\{B_1, B_2\}$. Swapping the indices if necessary, the following property holds (*cf.* Fig. 4):

$$A_1 \subsetneq B_1 \text{ and } B_2 \subsetneq A_2. \tag{1}$$

Proof: As the skeletons are trees, each edge cuts the graph G_i in two subgraphs. Without loss of generality, we suppose that the edge $\{B_1, B_2\}$ is in $G(A_2)$. This edge separates two sets of extremities, B_1 and B_2, and as G_i is agenus , one of those sets is in the graph $G(A_2)$. Without loss of generality, we suppose that the set of extremities B_2 is in $G(A_2)$, which means $B_2 \subsetneq A_2$. We can immediatly deduce that $A_1 \subsetneq B_1$, as A_1 and B_1 are the complements of A_2 and B_2. □

Find the closest topology: As we just have proved that the set S_{e_i} represents the graph G_i, we can insert or collapse edges directly in S_{e_i}. We denote by $i(e)$ the insertion of the edge e, and by $c(e)$, the collapse of the edge e, where e is a partition of V_e, with $\#e = 2$. As sets are unordered, insertions can be done in any order. Based on this, we have to insert in a set S_e all the edges that are contained in at least $n/2 + 1$ sets S_{e_i}.

We have these two assertions: first, the set S_e represents a graph, and second, the graph G associated to S_e is the closest graph to all G_i.

Proof: The second assertion is proved by the fact that all the insertions are commutative. For the first assertion, suppose that we have two incompatible edges e_1 and e_2 in S_e, *i.e.* Equation (1) does not hold. These edges come from a set of graphs, and are contained in at least $n/2+1$ sets S_{e_i}. As there is only n graphs, there exists a graph G_i that contains e_1 and e_2, thus the edges are compatibles, which is a contradiction.

3. RECONSTRUCTION PIPELINE

We describe here the skeleton-based reconstruction pipeline from an arbitrary number of images, which generalizes [13]. As input, we have calibrated acquisitions of an object, that is represented by a curve-skeleton, from different viewpoints. Each pose is computed using markers on the scene [21]. Then

we extract a mask of the object on each image, with the grabcut algorithm [22]. After that, we can extract the projective skeleton, using Voronoï perspective skeletonization [13]. By definition, each point of the projective skeleton corresponds to the projection of a sphere. The skeleton is simplified using the scale axis transform [23], to remove spurious branches.

The next step, related to our contribution, is the association of the skeletal branches. To do this, we manually match skeleton extremities, then determine the common skeleton topology given by the graph computed in Section 2. From the common topology, we match the different branches on the different skeletons: so for each branch of the recovered topology, we have a branch in each projective skeleton. We now need to match them point by point, to triangulate the 3D branch. We describe each of the branches by a function $L_i(\tau_i)$, which associate to each $\tau_i \in [0;1]$ a line in $\mathbb{R}^4$ representing the set of spheres that project on the point.

The idea here is to search for each $(\tau_1, \ldots, \tau_n)$ that corresponds to a point (x, y, z, r) of the 3D skeleton. Thus a function P associates to each $(\tau_1, \ldots, \tau_n)$ the closest point in $\mathbb{R}^4$ to the set of lines $(L_i(\tau_i))_{i=1,\ldots,n}$, under a least squared distance. This $\mathbb{R}^4$ function is then approximated by a cubic spline.

As the points of the reconstructed skeleton correspond to the points where the lines intersect, we also can define a function $D(\tau_1, \ldots, \tau_n)$ which associates the sum of squared distances from $P(\tau_1, \ldots, \tau_n)$ to $(L_i(\tau_i))_{i=1,\ldots,n}$. Furthermore, this function has to be continuous, so our objective here is to estimate a function $\tau : [0;1] \rightarrow [0;1]^n$ which minimises the following functionnal:

$$\int_0^1 \lambda \|\overrightarrow{\tau'(t)}\|^2 + D(\tau(t))dt$$

where λ is a parameter controling the length of the curve. This was solved in [13] using dynamic programming, *i.e.* a Dijkstra algorithm. However, this algorithm searches in a space of dimension n, which is too computationaly expensive to run for $n \geq 2$. Furthermore, it does not take account of the derivability of the function τ. Thus, we use here the Lagrangian method to compute the trajectory and to estimate $\tau(t)$. This method finds the optimal speed $\overrightarrow{\tau'(0)}$ to reach the position $\tau(1)$. This way, the number of images of P can be higher than two and the computation time is less than a minute for the whole skeleton.

The final 3D skeleton is the union of 3D connected branches, from which we compute a mesh associated to the object.

4. RESULTS

We evaluate quantitatively our algorithm on synthetic 3D curve skeletons (*cf.* Tab. 4). Note that the error relative to the original mesh lowers as the number of images increases. In order to evaluate the validity of the recovered topology, we test our algorithm on 7 real objects (see supplementary material). We show a result on Figure 5. Note that our method reconstructs a complete, closed object from the calibrated images and few interactions. Head and arms are well reconstructed, but the body of the blue plush is not as flat as the real one (this is because the method can not reconstruct objects with a surface skeleton). Furthermore, as the topology is the closest to the set of projective skeletons, some assymetries can appear on the estimated topology, for example the arms are not connected to the same node in Figure 5.

Number of images	2	3	4	5
Skeleton 1	2.44%	2.13%	1.49%	1.57%
Skeleton 2	4.87%	3.34%	2.46%	1.90%
Skeleton 3	4.97%	4.44%	4.07%	3.99%
Skeleton 4	4.68%	3.57%	3.51%	3.36%

Table 1. Reconstruction of 4 composed skeletons, with 4, 4, 5 and 6 extremities respectively. Use 2, 3, 4 and 5 calibrated images for each object, the error is the Hausdorff distance between the original and the reconstructed mesh, normalized by the diagonal of the bounding box.

Fig. 5. Reconstruction of two plushes using our method. Left (input): 4 calibrated images. Middle: reconstructed skeleton, in red. Right: reconstruction using the skeleton.

5. CONCLUSION AND FUTURE WORK

In this article, we have shown how to estimate a topology for the 3D reconstructed skeleton from a set of projective skeletons. With this method, we can reconstruct 3D skeletons from a set of real images, from an arbitrary number of images. Furthermore, an advantage of our aproach is to obtain a complete 3D mesh for the object, as we recover the topology.

A first future improvement is to match automatically the extremities of the skeletons. We also could manage non genus-0 objects, by adding new operations on topological skeletons, loop creation and loop separation. Finally, so far we only worked on curve-skeletons, we should extend our method to also reconstruct objects with surface skeletons.

6. REFERENCES

[1] T. Igarashi, S. Matsuoka, and H. Tanaka, "Teddy: a sketching interface for 3D freeform design," in *ACM SIGGRAPH 2007 courses*, 2007, pp. 21–28.

[2] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multi-view stereopsis," *IEEE Transactions on PAMI*, vol. 32, no. 8, pp. 1362–1376, 2010.

[3] K. M. G. Cheung, S. Baker, and T. Kanade, "Shape-from-silhouette of articulated objects and its use for human body kinematics estimation and motion capture," in *IEEE CVPR conference*, 2003, vol. 1, pp. 77–84.

[4] T. Chen, Z. Zhu, A. Shamir, S.-M. Hu, and D. Cohen-Or, "3-sweep: Extracting editable objects from a single photo," *ACM Transactions on Graphics*, vol. 32, no. 6, pp. 195–204, 2013.

[5] H. Blum, "A transformation for extracting new descriptors of shape," *Models for the Perception of Speech and Visual Form*, pp. 362–380, 1967.

[6] K. Siddiqi, A. Shokoufandeh, S. J. Dickinson, and S. W. Zucker, "Shock graphs and shape matching," *IJCV*, vol. 35, no. 1, pp. 13–32, 1999.

[7] H. Sundar, D. Silver, N. Gagvani, and S. Dickinson, "Skeleton based shape matching and retrieval," in *Shape Modeling International*, 2003, pp. 130–139.

[8] X. Bai and L. J. Latecki, "Path similarity skeleton graph matching," *IEEE Transactions on PAMI*, vol. 30, no. 7, pp. 1282–1292, 2008.

[9] Y. Xu, B. Wang, W. Liu, and X. Bai, "Skeleton graph matching based on critical points using path similarity," in *ACCV 2009*, pp. 456–465. Springer, 2010.

[10] P.-C. Liu, W.-C. Wu F.-C.and Ma, R.-H. Liang, and M. Ouhyoung, "Automatic animation skeleton using repulsive force field," in *Pacific Graphics*, 2003, pp. 409–413.

[11] G. Aujay, F. Hétroy, F. Lazarus, and C. Depraz, "Harmonic skeleton for realistic character animation," in *Symposium on Computer Animation*, 2007, pp. 151–160.

[12] L. Kavan, S. Collins, J. Zara, and C. O'Sullivan, "Skinning with dual quaternions," in *Symposium on Interactive 3D Graphics and Games*, 2007, pp. 39–46.

[13] B. Durix, G. Morin, S. Chambon, C. Roudet, and L. Garnier, "Towards skeleton based reconstruction: From projective skeletonization to canal surface estimation," in *International Conference on 3D Vision*, 2015, pp. 545–553.

[14] M. Livesu, F. Guggeri, and R. Scateni, "Reconstructing the curve-skeletons of 3d shapes using the visual hull," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 11, pp. 1891–1901, 2012.

[15] M. Livesu and R. Scateni, "Extracting curve-skeletons from digital shapes using occluding contours," *The Visual Computer*, vol. 29, no. 9, pp. 907–916, 2013.

[16] V. Caglioti and A. Giusti, "Reconstruction of canal surfaces from single images under exact perspective," in *ECCV 2006*, pp. 289–300. Springer, 2006.

[17] T. B. Sebastian, P. N. Klein, and B. B. Kimia, "Recognition of shapes by editing their shock graphs," *IEEE Transactions on PAMI*, vol. 26, no. 5, pp. 550–571, 2004.

[18] K. Zhang, R. Statman, and D. Shasha, "On the editing distance between unordered labeled trees," *Information Processing Letters*, vol. 42, no. 3, pp. 133–139, 1992.

[19] X. Gao, B. Xiao, D. Tao, and X. Li, "A survey of graph edit distance," *Pattern Analysis and Applications*, vol. 13, no. 1, pp. 113–129, 2010.

[20] P. J. Dickinson, H. Bunke, A. Dadej, and M. Kraetzl, "On graphs with unique node labels," in *Proceedings of the 4th IAPR International Conference on Graph Based Representations in Pattern Recognition*, Berlin, Heidelberg, 2003, GbRPR'03, pp. 13–23, Springer-Verlag.

[21] H. Kato and M. Billinghurst, "Marker tracking and HMD calibration for a video-based augmented reality conferencing system," *International Workshop on Augmented Reality*, pp. 85–94, 1999.

[22] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 309–314, 2004.

[23] J. Giesen, B. Miklos, M. Pauly, and C. Wormser, "The scale axis transform," in *Annual Symposium on Computational Geometry*, 2009, pp. 106–115.