HAL
open science

Temporally Coherent Segmentation of 3D Reconstructions

Kiran Varanasi, Edmond Boyer

To cite this version:

Kiran Varanasi, Edmond Boyer. Temporally Coherent Segmentation of 3D Reconstructions. 3DPVT 2010 - 5th International Symposium on 3D Data Processing, Visualization and Transmission, May 2010, Paris, France. inria-00568905

HAL Id: inria-00568905

https://inria.hal.science/inria-00568905v1

Submitted on 23 Feb 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Temporally Coherent Segmentation of 3D Reconstructions

Kiran Varanasi
INRIA Grenoble Rhône-Alpes
kiran.varanasi@inrialpes.fr

Edmond Boyer
INRIA Grenoble Rhône-Alpes
edmond.boyer@inrialpes.fr

Abstract

In this paper, we address the problem of segmenting consistently an evolving 3D scene reconstructed individually at different time-frames. The spatial reconstruction of 3D objects from multiple views has been studied extensively in the recent past. Various approaches exist to capture the performance of real actors into voxel-based or mesh-based representations. However, without any knowledge about the nature of the scene being observed, such reconstructions suffer from artifacts such as holes and topological inconsistencies. We explore the problem of segmenting such reconstructions in a temporally coherent manner, as a decomposition into rigidly moving parts. We work with mesh-based representations, though our method can be extended to other 3D representations as well. Unlike related works, our method is independent of the scene being observed, and can handle multiple actors interacting with each other. We also do not require an 'a priori' motion-estimate, which we compute simultaneously as we segment the scene. We individually segment each of the reconstructed scenes into approximately convex parts, and compute their reliability through rigid motion estimates over the sequence. We finally merge these various parts together into a holistic and consistent segmentation over the sequence. We present results on publicly available data sets.

1. Introduction

Segmentation of meshes is a well-studied problem in computer graphics [14][19][18][13][15], because it links skeletal representations that are useful for animation with mesh representations that are useful for rendering. However, most of the proposed approaches deal with smooth meshes that are acquired from artists or from laser-scans of real actors. Computer vision provides alternative methods for obtaining 3D meshes, by multi-view stereo from synchronized and calibrated cameras [12][10][16]. These methods aggregate silhouette and photometric information from the different views to build 3D meshes. They have a few advantages in that the captured 3D models are true to reality at every instant and that the texture information is trivially mapped to the reconstructed geometry.

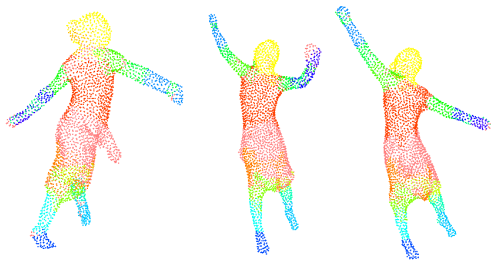


Figure 1. A temporally coherent segmentation of a sequence of visual hull reconstructions

However, without any knowledge of the scene being observed, these methods produce certain topological and geometric artifacts that are unique to them (figure 2). Topological inconsistencies arise when distinct objects get clubbed as they approach each other. For example, the two legs of a human actor may get clubbed together. Holes are introduced into 3D models because of problems in silhouette extraction. Finally, the obtained meshes are not usually smooth and suffer from local surface distortions due to image and silhouette noise. Please note that explicitly specifying a human model [21] can help resolve some of these ambiguities, but limits the scenes from containing more objects (either human or non-human). In this paper, we address arbitrary 3D scenes with multiple actors. Obtaining a segmentation of such meshes into body parts would be beneficial to realizing various common graphics tasks such as shape editing, animation, motion transfer etc.

Unfortunately, the segmentation approaches proposed in computer graphics are limited in their applicability towards these meshes because of their unique artifacts. For example, those approaches often use surface metrics like curvature or dihedral angles which are sensitive to noise on the meshes. Sometimes, they make restrictive assumptions such as limiting the approach to genus zero shapes [14]. Such assump-

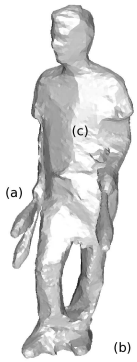


Figure 2. A 3D mesh of a juggler holding clubs, reconstructed from multiple view silhouettes. Visible artifacts of reconstruction (a) the clubs merged with the hands (b) the two feet merged together (c) spurious noise on the surface

tions cannot be made when topological artifacts are present, such as when the limbs of an actor are merged at their ends. A general limitation of model-based approaches is on dealing with unknown scenes, as in the case of multiple interacting actors. We believe that these limitations can be addressed by exploiting the temporal dimension that links these individual reconstructions.

In this paper, we propose an approach to segment such visually reconstructed meshes in a temporally consistent manner. However, we don't assume that a motion estimate is available *a priori* that links the meshes over the sequence. A few methods already exist for performing the motion segmentation of 3D trajectories [9]. But obtaining such motion estimates is a hard problem in its own right. In a sense, 3D segmentation and motion estimation present a chicken and egg dilemma, with each of these steps depending on the other for improvement. In this paper, we attempt to solve these problems together. The crucial insight we exploit is that the convex segments of a shape usually represent the limbs of its articulation structure, and therefore exhibit rigid motions over time.

The main contributions of our work are (a) a novel algorithm to derive approximately convex segments from a single mesh (a) a method for temporally coherent segmentation of a sequence of visually reconstructed meshes (c) a method for motion estimation of these segments over such a sequence. Since we don't assume that the meshes we work with are of high quality, we don't attempt to align the segment boundaries with mesh concavities. Instead, our motivation is to discover the (possibly overlapping) set of rigid body parts which are reliably detected throughout the sequence. The paper is divided as follows. In section 2, we briefly review the related work. In section 3, we provide an overview of our approach and introduce its various steps which are elaborated in later sections. In section 4, we explain the segmentation of a mesh at a *seed frame*. In section 5, we explain the process of arriving at a temporally coherent segmentation. In section 6, we provide results from experiments on different data sets. In section 7, we conclude and provide directions to future work.

2. Related Work

Various functions on the surface have been proposed that are oblivious to pose-changes and articulations. Geodesic distances on the surface of the mesh are invariant to isometry, and can be used for matching surfaces which differ by a non-rigid deformation [4]. However, they are very sensitive to topological changes (which occur often in visually reconstructed meshes). Bronstein *et al.*[5] use a metric based on diffusion over the surface through a heat kernel, which is less sensitive in this regard. Functions defined on the mesh-volume are more resistant to topological changes. Shapira *et al.*[19] use the local thickness of the mesh-volume, termed the shape-diameter function, to consistently partition different surfaces over articulations and pose changes. These and other works in the geometry processing community use surface curvature to derive the final segmentation, which is fine for smooth 3D models but is not reliable for visually reconstructed meshes.

Instead of relying on the supposed consistency of a function, a set of objects can be explicitly marked as related and can be segmented consistently. Golovinskiy and Funkhouser[11] have proposed a method that consistently segments sets of objects such as chairs and airplanes. They need point-wise correspondence between the objects, which they obtain by doing a global ICP registration and taking the closest point pairs between objects. Though this approach is powerful, it is not suited to the context of a sequence of meshes as it doesn't exploit the temporal information. In the context of a mesh sequence, de Aguilar *et al.*[9] use the point trajectories (computed, for example, using [8]) to cluster points into rigidly moving parts.

A graphical model can be used to explicitly match two surfaces differing from a non-rigid deformation [2] [20], and these point correspondences can be later used to recover the articulated structure (skeleton) of the shape, as proposed by Anguelov *et al.*[1]. Chang and Zwicker [6] propose an interesting alternative in the graphical model approach; they compute a putative set of rigid transformations between surface points on two articulated shapes, and treat these transformations (instead of the points themselves) as the labels for the graphical model. These approaches are very powerful but surface matching or motion estimation is a very hard problem, and the results are often inaccurate in the presence of surface occlusions, topological noise and disjoint objects. In this paper, we propose a segmentation algorithm that does not rely on pre-computed correspondences or motion estimates.

Our work is most closely related to that of Cuzzolin *et al.*[7]. In this work, the authors use locally linear embedding (LLE) to represent a cloud of points and derive a segmentation in this space. The segments are then propagated across time to obtain a temporally coherent segmentation of a voxel-sequence into protrusions of the shape, such as head, hands and legs. Though very powerful, this method cannot be used directly for identifying rigid body-parts (for example, separating the upper-arm from the lower-arm) and for working on disjoint objects (for example, two interacting persons). To the best of our knowledge, nobody has attempted the segmentation of a mesh-sequence with multiple interacting persons reconstructed visually in a multi-camera environment.

3. Approach Outline

We consider as input a mesh sequence $\mathcal{M}^{t\in[1..n]}$. Each of the meshes $\mathcal{M}^t$ is an individual 3D reconstruction of a real scene composed of several actors and objects, and is composed of a set of vertices $\mathcal{V}^t$ and a set of facets $\mathcal{F}^t$. We further represent the volumetric information of the mesh $\mathcal{M}^t$ by a set of interior points $\mathcal{X}^t$. An individual surface point is denoted as $v_i^t \in \mathcal{V}^t$ and an individual interior point is denoted as $x_i^t \in \mathcal{X}^t$.

The task is to estimate a set of segments $\Psi = \{\mathcal{S}_j\}$ and their rigid transformations over time $\mathcal{T}_j^{t\in[1..n]}$. Each segment $\mathcal{S}_j$ is detected at a specific *seed-frame* $t = k$ and consists of a set of N_j surface points. These points are represented by a matrix $\mathcal{S}_j^k$ of size $N_j \times 4$, with each column representing a surface point in homogeneous coordinates. The segment points at another frame t are then given by the matrix $\mathcal{S}_j^t = \mathcal{T}_j^t * \mathcal{S}_j^k$ (with $\mathcal{T}_j^k = I$). An individual segment point in the segment S_j^t is denoted as s_{ji}^t.

We say that a set of segments $\Psi = \{\mathcal{S}_j\}$ is an *ϵ-cover* of the mesh sequence if no surface point v_i^t reconstructed at frame t is farther than a distance of ϵ from the corresponding set of segments $\{\mathcal{S}_j^t\}$. *i.e,*

$$\forall k \quad \forall v_i^t \in \mathcal{V}^t, \quad Min_j \quad Dist(v_i^t, \mathcal{S}_j^t) < \epsilon \qquad (1)$$

An ϵ-cover yields a temporally coherent segmentation of the sequence by a margin ϵ. In practice, we achieve this by two steps.

1. **Convex segmentation of a static mesh :** We segment a mesh at a single frame M^k into convex segments. We do this by a novel method of finding visibility clusters in the mesh volume (section 4).

2. **Temporally coherent segmentation of a sequence :** We perform the above step at multiple randomly sampled frames in the mesh sequence. We estimate the reliability of each segment by registering it along the whole sequence and computing the registration error (section 5.1). We then devise an algorithm for building an ϵ-cover of the sequence from the above set of segments. (section 5.2).

Algorithm 1: Convex Segmentation of the Surface

Input: a surface mesh $\mathcal{M} = (\mathcal{V}, \mathcal{F})$
Output: a set of convex segments $\{S_{j\in[1..m]} \subset \mathcal{V}\}$
Obtain a set of interior points $\mathcal{X}$ for the mesh $\mathcal{M}$;
Initialize the set of free points $\mathcal{X}_F = \mathcal{X}$;
Sort $\mathcal{X}_F$ according to a function ρ;
i = 0;
while $\mathcal{X}_F$ *is not empty* **do**
  Select the top element of $\mathcal{X}_F$ as the viewpoint w_i;
  Compute the visible volume VV_i from w_i;
  Remove all points $x \in VV_i$ from $\mathcal{X}_F$;
  Decompose VV_i into convex segments $\{S_j\}$;
  i = i+1;
end

Figure 3. Algorithm for convex segmentation of a mesh

4. Segmentation into Convex Parts

In this section, we describe our method of extracting convex segments from a 3D scene $\mathcal{M}^k = (\mathcal{V}^k, \mathcal{F}^k)$ reconstructed at a particular frame k. Since we deal with the static case, we eschew the superscript and denote the mesh by just $\mathcal{M} = (\mathcal{V}, \mathcal{F})$. As mentioned in section 1, meshes reconstructed from images suffer from a huge amount of geometric and topological noise (figure 2). Surface based metrics such as point curvature or geodesic distances between points are not reliable over such meshes. In our method, we use volumetric information represented by *interior points* of the mesh close to the medial axis, which are more resistant to these artifacts. It is to be noted that computing the medial axis exactly is not trivial in 3D, but we aim only to estimate a cloud of interior points. We associate every surface vertex on the mesh $v \in \mathcal{V}$ with one or more interior points $x \in \mathcal{X}$. We define the problem of convex segmentation of the surface points $\mathcal{V}$ as the segmentation of the interior point set $\mathcal{X}$.

We consider two interior points x and y to be *visible* from each other if the line joining them doesn't intersect any mesh facet in $\mathcal{F}$. From a selected interior point $w_i \in \mathcal{X}$ (termed the *viewpoint*), we compute the set of visible interior points $VV_i \subset \mathcal{X}$ using this definition. As illustrated in the figure 4, this *visible volume* constitutes one or more convex segments. Our algorithm 3 for convex segmentation rests on this observation.

We first compute the set of interior points $\mathcal{X}$ and spatial-hash them for easy access. We initiate by marking all of

them as *free* and sort them according to an ordering function ρ. A simple choice for ρ is the random-ordering function, another choice is mentioned later in section 4.4. We pop out the top element in the ordered free set $\mathcal{X}_F$ as the viewpoint w_i and compute its visible volume VV_i. We decompose VV_i into a set of convex segments $\{S_j\}$, tag every point with the corresponding segment, and remove the tagged points from the free set $\mathcal{X}_F$. We repeat the algorithm until the free set $\mathcal{X}_F$ is empty. In the following, we explain these steps in detail.

4.1. Estimation of interior points

Our objective is to obtain a set of uniformly sampled interior points close to the medial axis. We adopt a method similar in principle to the one described by Shapira *et al.* [19]. For each point on the surface v , we shoot a cone of rays opposite to the normal and identify their points of intersection with the surface. We compute the median length of these line segments as the *shape diameter*. We then trace a vector from v opposite to its normal with half this length, and call it the *medial vector* at v. After we compute medial vectors at every surface point, we perform spatial smoothing such that neighboring points have similar medial vectors. These medial vectors map the set of surface points $\mathcal{V}$ to a set of interior points $\mathcal{X}$. We spatial-hash these interior points for easy access, and construct a k-nearest neighbor graph (termed the *medial graph*) on this point set. Here, we would like to note that $\mathcal{X}$ need not be as densely sampled as $\mathcal{V}$; interior points can be constructed for only a subset of the surface points. For each interior point x, we associate a set of surface vertices (termed the *surface ring*) that are close to it in any direction.

4.2. Computation of a visible volume

Given a specific viewpoint w from which to compute the visible-volume, we first order the set of interior points in a *medial tree* based on the shortest path from w on the *medial graph*. As illustrated in figure 5, this tree has w at the root and captures well the underlying structure of the shape as seen from w.

We then compute the visible *surface* from w by checking for each surface point v if the line joining v with w intersects any mesh-facet from $\mathcal{F}$. This computation may be performed rapidly by spatially hashing the mesh facets. We then descend the *medial tree* computed in the earlier step by accumulating interior points if their associated *surface ring* is entirely visible. When a non-visible surface point is encountered, we mark the corresponding interior point t as an *occluding tip*. We chop the *medial tree* at that point (figure 5-b) and don't grow any further. This way, we build a visible volume from the given viewpoint w, which is usually convex, but can be a star-shaped structure in the general case (as illustrated in figure 6).

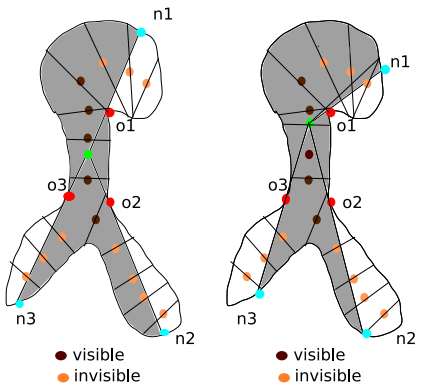
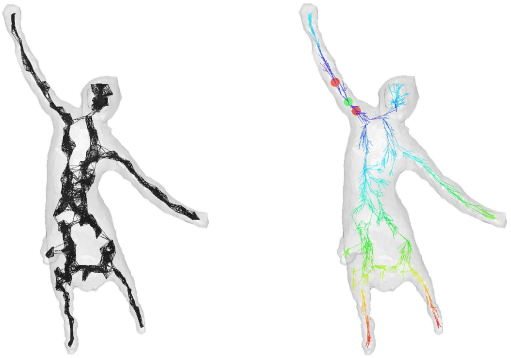


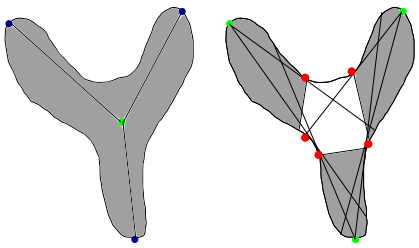
Figure 4. Computation of visible volume, conceptualized in 2D : the viewpoints are colored green, the occluding boundary (o1,o2,o3) are colored red and the non-occluding boundary (n1,n2,n3) are colored blue. Change of viewpoint doesn't affect the occluding boundary. Interior points are shown with their *surface rings* indicated as line-segments traversing the volume, An interior point is considered visible if only its entire surface ring is visible. The two view points yield the same set of visible interior points.



(a) (b)

Figure 5. (a) The k-nearest neighbor graph on the interior point set, termed as the *medial graph* (b) The *medial tree* constructed by graph distance from a viewpoint, shown in green. The tree-branches in warmer colors are farther away from the viewpoint. The two red points are the *occluding tips* which delimit the visible volume from the given viewpoint.

As can be seen in the figure, the boundary of the visible surface can be distinguished into a self-occluding part and a non-occluding part. While the former separates convex segments from each other, the latter is non-informative about the structure of the shape. It is the self-occluding boundary that prompts decisions in our algorithm. After the algorithm to identify the visible volume is terminated, we compute its extent δ_e as the largest distance across any two points inside. If this value is very small, we reject the visible volume as



(a) (b)

Figure 6. Convex decomposition of a visible volume : (a) The visible volume from the view point is not a convex segment, but a star shaped structure. The viewpoint is colored green, and the protruding tips of the star are colored dark blue. (b) Visible volumes are then computed from each tip of the star, which run into occluding boundaries (colored red). A convex segment is identified per each tip (colored gray), and the remaining stub of the visible volume (colored white) becomes an independent convex segment. For rarer and more complex shapes, this process of convex decomposition may need be performed recursively.

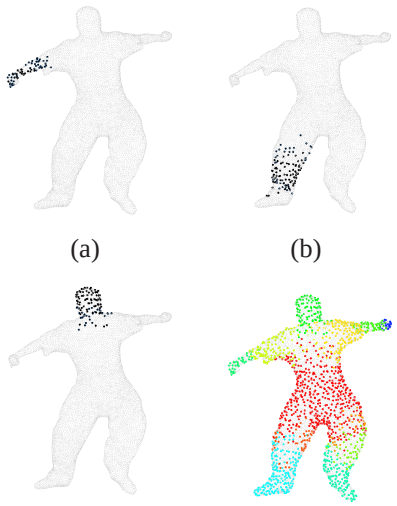
too small to be significant. Otherwise, we proceed to the next step.

4.3. Convex decomposition of a visible volume

A visible volume is shaped as a star, and may contain multiple convex segments (as can be seen in the figure 6). Here we describe how to decompose the visible volume into its constituent segments. We first obtain the *tips* of the visible volume by identifying the set of points which are locally maximal in their neighborhood with respect to the distance from the viewpoint w. The occluding tips discovered as explained in the earlier section are also added to this set.

We then compute a Boolean compatibility graph between the tips based on whether the surface ring associated with each tip is completely visible from the other. If all the tips are mutually compatible with each other, we mark the visible volume as convex and terminate the algorithm. Otherwise, we make each tip as a new viewpoint and recursively compute its visible volume, but this time restricted only to the star-shaped volume visible from the parent viewpoint w. If all the tips are accounted for and there still remains a stub of unassigned points, we mark this stub as an independent convex segment. At the end of this recursive procedure, the visible volume is decomposed into a unique set of convex segments.

Since we deal with noisy meshes reconstructed from images, we test only for approximate convexity and not exact convexity. In practice, this means we use certain thresholds in (1) identifying tips and (2) testing for compatibility between tips. For (1), we perform non-maximal suppression in detecting tips, and don't return a tip within a distance of δ_1 from another tip. For (2), we consider two tips as being incompatible only if there exists a pair of points v_a, v_b



(a) (b)

(c) (d)

Figure 7. Convex segments at a static frame : (a)(b)(c) individual segments (d) color-coded representation of all the segments found

in their corresponding *surface rings*, the line $\overline{v_a v_b}$ joining which is farther than a threshold δ_2 from the surface. The thresholds δ_1 and δ_2 are set as equal to the user specified margin ϵ of equation 1.

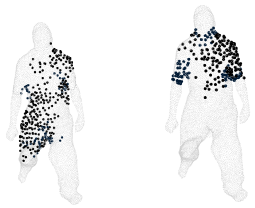
We represent the segments as (potentially overlapping) point-clouds. Example segments are shown in figure 7.

4.4. Heuristics for choosing viewpoints

Our algorithm correctly identifies convex segments irrespective of where the viewpoints are placed. However the algorithm is faster when all the tips of the visible volume are mutually compatible, thereby returning just one convex segment. Such cases are more probable when the viewpoint is placed close to the potential tip of the visible volume. In articulated figures such as human bodies, protrusions such as hands and legs can be detected easily, by computing the average geodesic distance from a point to every other point on the mesh (figure 9-a). The protrusions of a shape come out as local maxima of this function. We use this function as the sorting function ρ in algorithm 3.

5. Temporally Coherent Segmentation

We use the above algorithm to obtain convex segments at multiple frames in the mesh sequence. We denote the holistic set of segments obtained as $\mathcal{P}$. These segments usually identify body parts, but not all of them are equally reliable. Clothing and occlusions create artifacts in the visual reconstructions, making segment-boundaries appear at places not corresponding to body parts (figure 8). Further, distinct body parts may be clubbed together into a single



(a) (b)

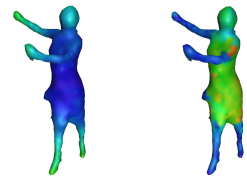
Figure 8. Limitations of static segmentation : (a) due to loose clothing, the left thigh merges with the torso (b) the upper arms get merged with the torso due to reconstruction artifacts

segment either due to an articulation (the upper and lower leg joined together into a single segment) or due to an artifact in reconstruction (the hand clubbed together with the stomach as it approaches closely). Such segments are technically convex, but remain so only in one or few frames. In this section, we describe a method for rejecting such segments and identifying ones which are consistent with the entire mesh sequence. To achieve this, we estimate the motion of each segment over the entire sequence. We observe that the convex segments of a mesh usually correspond to the articulated parts of a body and thus, their motion can be approximated as rigid.

5.1. Reliability estimation

We take the surface point cloud S_j^k of a convex segment S_j detected at frame k. We estimate the motion of this point cloud as a set of rigid transformations $T^{t\in[1..n]}$ over the entire sequence. To do this, we iteratively register the segment's point cloud to the mesh points in the neighboring frames using the ICP algorithm [3]. As the 3D video is captured at a good frame-rate (around 10 to 20 fps), neighboring frames are sufficiently close to each other, justifying the application of the ICP algorithm.

We accelerate the ICP algorithm by using spatial kd-tree organization. When we select closest points for registration, we reject matches between points with widely discrepant surface normals. As observed by Pulli [17], this is an efficient strategy for eliminating outliers in the registration. Since the ICP algorithm is based on local search, it doesn't find good matches across large movements. Following Shapira *et al.*[19], we use the two mesh features introduced in the earlier section (a) the average geodesic distance, which helps in detecting surface protrusions and (b) the shape diameter which gives the thickness of the volume (figures 9-a and 9-b). We reject matches between points with discrepant values for these features, this strategy helps us find better matches than simple closest point search. We used simple geometric registration for ICP, even though more complicated methods exist that account for photometric information or scene-flow.



(a) (b)

Figure 9. (a) The average geodesic distance (b) The shape diameter function. Warmer colors mean higher values in both the figures

We consider the success of registration along the sequence as an estimate of reliability for a convex segment. For each point s_{ji}^k in S_j^k, we compute its estimated position at frame t as $s_{ji}^t = T^t * s_{ji}^k$. The discrepancy $d(s_{ji}^t)$ in this estimate is computed as the least distance to the mesh vertices reconstructed at frame t.

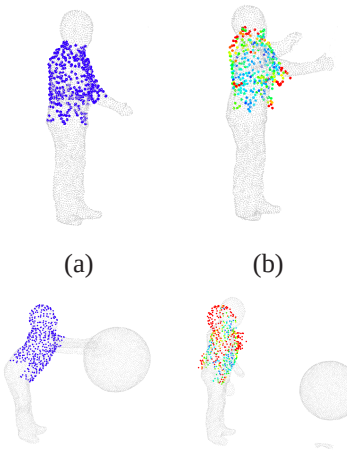
$$d(s_{ji}^t) = Min_a \quad ||T_j^t * s_{ji}^k - v_a^t|| \tag{2}$$

If this value is more than ϵ, we note the point to be lost in registration at this frame. Points that are lost several times during the registration are unlikely to be part of the actual segment. We prune the segment by removing such points from the border of the segment as outliers (figure 10). In certain cases, not just a few, but a vast chunk of points in the segment suffer from discrepancies $\geq \epsilon$. We discard such segments altogether as incorrect. For each remaining segment S_j, we compute its *size* as the total number of mesh vertices in $\{\mathcal{V}^{t\in[1..n]}\}$ that are within the margin ϵ to its estimated position at the corresponding frame S_j^t. We sort the various segments in $\mathcal{P}$ according to their size in ascending order.

5.2. Building an ϵ-cover

We construct $\mathcal{P}$ by performing static segmentation on multiple randomly chosen frames. It is normally sufficient to segment just 3 to 5 frames, because many segments shall be detected repeatedly in all the frames. Following equation 1, a temporally coherent segmentation is given by an $\epsilon - cover$ of segments chosen from $\mathcal{P}$. We take a greedy approach to obtain this.

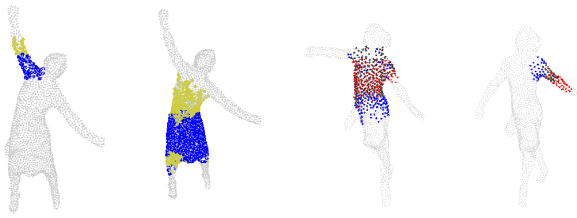
We maintain the current set of accepted segments Ψ and gradually add new segments into it from $\mathcal{P}$ (Ψ is started out as empty). At each stage, We pop out the top element S_j in the sorted set $\mathcal{P}$ and check if it overlaps with any of the accepted segments S_a in Ψ. We call a point s_{ji} in S_j at seed-frame $t = k$ to be within the ϵ-*margin* of a segment S_a, if it is within the distance of ϵ from the estimated positions of any of the points in S_a^k. We run this test from all accepted segments, and mark the points in S_j that are within the ϵ-margin of a prior segment. Examples are shown in figure 11-a,b. We define the overlap between two segments S_j



(a) (b)

(c) (d)

Figure 10. Reliability Estimation of Segments : (a) (c) - the segments detected at their seed-frames, (b) - outlier points with high discrepancy are pruned from the borders of (a), (d) segment is discarded after too many discrepancies in registration from (c). Points with higher discrepancy are shown in warmer colors - the red points are the outliers.

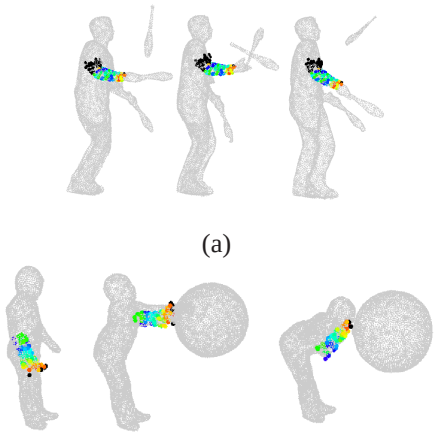


(a) (b) (c) (d)

Figure 11. Comparing segments by overlap estimation : (a)(b) the points colored yellow are discarded as outliers, for being within an ϵ-margin of an earlier accepted segment. The remaining segment points are colored in blue. (c)(d) the blue segment is detected as a repetition of the red one, and is discarded

and S_a as the fraction of points in S_j at its seed-frame $t = k$ that are within an ϵ-*margin* of S_a^k. If this overlap is large, we detect S_j as a repetition of S_a and proceed to the next segment. Due to the nature of the algorithm, it is the smallest of the segment repetitions that is acknowledged, the rest are discarded. This yields tighter segments, as can be seen in figure 11-c,d.

If, on the other hand, no overlap is detected for segment S_j with any of the earlier segments, then Ψ is augmented by adding S_j to it. This process is terminated When every surface point in the mesh-sequence is acknowledged to be within an ϵ-margin of one or more segments of Ψ. The set of segments Ψ now defines an ϵ-*cover* for the sequence.



(a)

(b)

Figure 12. Tracking of a rigid component in a point cloud of diverse objects

6. Results

We tested our approach primarily on visual hull reconstructions from multiple view silhouette data [1]. These silhouettes are extracted from real images taken through a synchronized camera setup in an indoor setting. The *dance* and *flashkick* sequences (figures 1 and 13-a) present fast movements of limbs that are difficult to be tracked.The *kids* and *juggle* sequences (figures 12 and 13-b) show human-human and human-object interactions respectively. In such interactive scenarios, it is very difficult to obtain prior knowledge of the scene, and hence difficult to make assumptions on the number of actors and the topology of these shapes. Our algorithm segments these scenes without making any such assumptions.

There are certain limitations for our approach. One interesting case is detailed in figure 14. The algorithm fails to track the juggler's club properly. When the club leaves the juggler's hand, our algorithm fails to follow the club and registers its points on the hand of the juggler. Then when a new and different club approaches the hand, our algorithm registers these points onto the new club. The features that we use are not discriminative enough to handle cases like these, and the limitations of the ICP registration are manifest. In general, without a global model for tracking, it is difficult to track segments correctly for long sequences. This is especially true for small segments that are not sufficiently distinctive.

7. Conclusion

In this paper, we presented a novel algorithm for coherently segmenting a sequence of visually reconstructed

[1] http://4drepository.inrialpes.fr

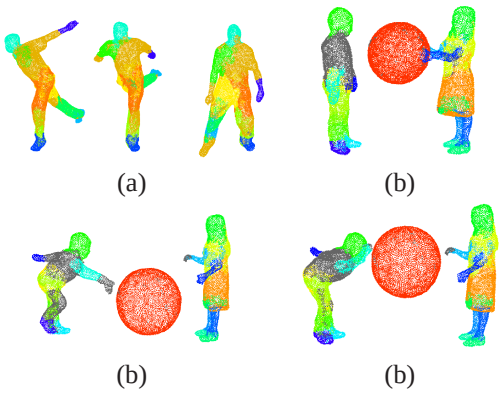


Figure 13. Temporally coherent segments on sequences (a) flash-kick (b) kids

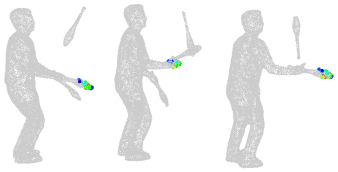


Figure 14. Limitations of the algorithm : the club cannot be distinguished from neighborhood objects, and is registered incorrectly

meshes without making any assumption on the type, number or topology of the objects in the scene. Once such segments are identified, they can provide a basis for learning the spatio-temporal model of the scene. Several potential applications await here to be explored. Our algorithm for registration is currently based on the ICP algorithm, and is thus limited to small displacements. In future work, we would like to overcome this limitation through stronger and more discriminative features for matching.

References

[1] D. Anguelov, D. Koller, H.-C. Pang, P. Srinivasan, and S. Thrun. Recovering articulated object models from 3d range data. In *Uncertainty in Artificial Intelligence*, 2004. 2

[2] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, H. Pang, and J. Davis. The correlated correspondence algorithm for unsupervised registration of non-rigid surfaces. In *Neural Information Processing Systems*, 2004. 2

[3] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2), February 1992. 6

[4] A. M. Bronstein, M. M. Bronstein, and R. Kimmel. Generalized multidimensional scaling: A framework for isometry-invariant partial surface matching. *Proceedings of the National Academy of Sciences*, 103/5:1168–1172, January 2006. 2

[5] A. M. Bronstein, M. M. Bronstein, and R. Kimmel. Topology-invariant similarity of nonrigid shapes. *International Journal of Computer Vision*, 81/3:281–301, March 2009. 2

[6] W. Chang and M. Zwicker. Automatic registration for articulated shapes. In *Proceedings of the Eurographics Symposium on Geometry Processing*, volume 27, 2008. 2

[7] F. Cuzzolin, D. Mateus, D. Knossow, E. Boyer, and R. Horaud. Coherent laplacian 3d protrusion segmentation. In *Computer Vision and Pattern Recognition*, 2008. 3

[8] E. de Aguiar, C. Stoll, C. Theobalt, N. Ahmed, H. P. Seidel, and S. Thrun. Performance capture from sparse multi-view video. In *ACM Transactions on Computer Graphics*, 2008. 2

[9] E. de Aguiar, C. Theobalt, S. Thrun, and H. P. Seidel. Automatic conversion of mesh animations into skeleton-based animations. *Proceedings of EUROGRAPHICS*, 27(2), 2008. 2

[10] Y. Furukawa and J. Ponce. Carved visual hulls for high-accuracy image-based modeling. In *Technical Sketch at SIGGRAPH 2005*, 2005. 1

[11] A. Golovinskiy and T. Funkhouser. Consistent segmentation of 3d models. In *ACM Computer Graphics (Proceedings SIGGRAPH)*, 2009. 2

[12] C. Hernandez and F. Schmitt. Silhouette and stereo fusion for 3D object modeling. *Computer Vision and Image Understanding*, 96(3):367–392, 2004. 1

[13] Q.-X. Huang, M. Wicke, B. Adams, and L. Guibas. Shape decomposition using modal analysis. *Proceedings of EUROGRAPHICS*, 28(2), 2009. 1

[14] S. Katz, G. Leifman, and A. Tal. Mesh segmentation using feature point and core extraction. *The Visual Computer*, 21:649–658, 2005. 1

[15] J.-M. Lien and N. M. Amanto. Approximate convex decomposition of polyhedra. In *Proceedings of the ACM Symposium on Solid and Physical Modeling*, pages 121–131, 2007. 1

[16] J.-P. Pons, R. Keriven, and O. Faugeras. Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score. *International Journal of Computer Vision*, 72(2):179–193, 2007. 1

[17] K. Pulli. *Surface Reconstruction and Display from Range and Color Data*. PhD thesis, University of Washington, 1997. 6

[18] A. Shamir. A survey on mesh segmentation techniques. *Computer Graphics Forum*, 27:1539–1556, 2008. 1

[19] L. Shapira, A. Shamir, and D. Cohen-Or. Consistent mesh partitioning and skeletonisation using the shape diameter function. *The Visual Computer*, 24:249–259, 2008. 1, 2, 4, 6

[20] J. Starck and A. Hilton. Correspondence labelling for wide-timeframe free-form surface matching. In *Proceedings of the 11th International Conference on Computer Vision, Rio de Janeiro, (Brazil)*, 2007. 2

[21] A. Sundaresan and R. Chellappa. Model-driven segmentation of articulating humans in laplacian eigenspace. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(10):1771–1785, October 2008. 1