Novel Skeletal Representation For Articulated Creatures

Gabriel J. Brostow, Irfan Essa, Drew Steedly, and Vivek Kwatra

Georgia Institute of Technology, Atlanta GA 30332, USA,
WWW home page: http://www.cc.gatech.edu/cpl/projects/spines

Abstract. Volumetric structures are frequently used as shape descriptors for 3D data. The capture of such data is being facilitated by developments in multi-view video and range scanning, extending to subjects that are alive and moving. In this paper, we examine vision-based modeling and the related representation of moving articulated creatures using *spines*. We define a spine as a branching axial structure representing the shape and topology of a 3D object's limbs, and capturing the limbs' correspondence and motion over time.
Our spine concept builds on *skeletal* representations often used to describe the internal structure of an articulated object and the significant protrusions. The algorithms for determining both 2D and 3D skeletons generally use an objective function tuned to balance stability against the responsiveness to detail. Our representation of a spine provides for enhancements over a 3D skeleton, afforded by temporal robustness and correspondence. We also introduce a probabilistic framework that is needed to compute the spine from a sequence of surface data.
We present a practical implementation that approximates the spine's joint probability function to reconstruct spines for synthetic and real subjects that move.

1 Introduction

We are interested in the detection and tracking of features in volumetric images. Volume images capture shape as a temporal sequence of boundary voxels or other forms of 3D surfaces. Specifically, we wish to address situations where the subject is known to have and is exercising an articulated structure. This assumption grants us use of a specific class of geometric modeling solutions. The various methods for skeletonizing 2D and 3D images share the objectives of identifying extrema, features with some geometric significance, and capturing the spatial relationships between them [9]. Skeletons, much like generalized cylinders [4, 21], serve the purpose of abstracting from raw volume or surface data to get higher level structural information.

We propose that evaluating volumetric data of a subject over *time* can disambiguate real limbs from noisy protrusions. In a single image, knowledge of the specific application alone would dictate the noise threshold to keep or cull small branches of the skeleton. Many such algorithms exist. In the case of articulated

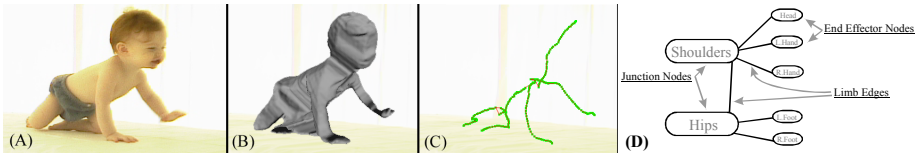


Fig. 1. (A) Articulated subject, (B) reconstructed surface, (C) extracted skeleton, (D) spine graph limbs encoding motion over time; nodes labeled for illustration only.

moving subjects, the volumetric images change but the underlying structure stays the same. We hypothesize that the parts of the skeleton within each image that are consistent over time more reliably capture the subject's structure. To this end, we introduce our notion of spines.

As defined in [4], a generalized cylinder is a surface obtained by sweeping a planar cross section along an axis, or space curve. To represent a body made of multiple generalized cylinders, we need to merge axes of the different limbs into one branching axial structure. The branching structure can be represented by a graph, $G(LimbBoundaries, Limbs)$, where edges are limbs, leaf nodes are end effectors, and the remaining nodes (all of degree > 2) are limb junctions (see Figure 1D). So far, we have described the general formulation of a skeleton [5]. To parameterize the motion of a skeleton, we express the new spine graph as a function over time:

$$Spine_t = F(G, t). \tag{1}$$

For a given time t, the limbs of G will be in a specific pose, captured by F's mapping of G's topology to axial curves in 3D – a single *skeleton*. When estimating a data set's *spine* in the subsequent sections, we will constrain F to manipulate the limbs of a G that represents a series of topologically consistent skeletons. These skeletons are determined as probable given the input data.

The implementation of our algorithm is a modular pipeline. It first reduces the complexity of multi-view video data to voxels, further to polygons, and finally to spines. The resulting model captures the original degrees of freedom needed to play back the subject's motions (see Figure 1).

2 Related and Motivating Work

The 2D analogue to our problem is the tracking of correspondence in medial axes, which were first introduced by Blum [5]. Given any of the numerous 2D skeletonizing techniques, including the classic grassfire models based on distance and the more robust area-based techniques [3], the work of Sebastian et al. [23] can determine correspondence by minimizing edit-distances of skeleton graphs in 2D.

The medial axes of 3D surfaces are not directly applicable because they generate 2D manifold "sheets" through a surface. While medial scaffolds can be

calculated fairly robustly [24, 19], they require further processing [28] to estimate good 1D axes.

Several 3D skeletonization algorithms have been developed using 3D Voronoi cells to partition the space within a mesh [2, 13, 25, 12, 16]. The cell-walls of these convex polyhedra land at equal distances from their designated surface start-points – some at or near the medial axis. This approach, with various extensions of projection and pruning, can generally serve to synthesize axes. In contrast to these, our approach and implementation are based on two sub-domains of solutions: measuring of geodesic distance from geometric modeling, and principal curves from statistics.

Geodesic Distance: In Section 4.1 we will discuss in greater detail how a surface can be treated as a piecewise continuous distance field that separates features from each other. Verroust and Lazarus [27] used such a technique to determine axes of symmetry within limbs, and how to connect them to critical points (special topological features) on the mesh surface. In an application not requiring branching axes, Nain et al. [22] used geodesic distances on colon models to determine center-lines for virtual colonoscopy navigation. Recently, a geodesic distance based metric was used by Katz and Tal [17] to help assign patches as members of explicit limbs, resulting in course animation control-skeletons. All these approaches benefit from works such as [15] which identify extrema, or features that protrude from or into a surface mesh. Our approach uses such extrema-finding and a geodesic distance metric to better model skeleton branching.

Principal Curves: Hastie and Stuetzle [14] defined principal curves as passing through the middle of a multidimensional data set, as a representation of self-consistency to generalize principal components. For fixed length curves in a geometric setting, Kegl et al. [18] showed how to minimize the squared distance between the curve and points sampled randomly from the encompassing shape. Most recently, [7] and [8] extended this notion of principal curves to 3D, formalizing the problem as an optimization which also seeks to minimize the curve length. Our extension is to incorporate branching and temporal correspondence.

3 Spine Formulation and Estimation

We build on the axial representation of generalized cylinders of [8, 7] because of their elegant mathematical formulation. They treat the regression problem of finding a single curve for a surface as the minimization of a global energy function. Much like the previous work on principal curves [14, 18], they seek to minimize the total distance from the axial curve to the surface. But in addition, [7] incorporates a term which penalizes the curve's length. This augmentation helps force the shorter curve to smoothly follow the middle of a surface, instead of, for example, spiraling through all the boundary points.

For our spine formulation, we seek to further incorporate: (a) skeletons S that model branching curves of individual surfaces X and (b) data captured

over a period of time T. We propose a discriminative probabilistic approach to computing spines by finding G, S, and limb end effectors E, which maximize:

$$P(G, S_{1:T}, E_{1:T}|X_{1:T}) = P(G|S_{1:T}, E_{1:T}, X_{1:T}) \cdot P(S_{1:T}, E_{1:T}|X_{1:T}) \quad (2)$$

To compute and optimize the joint probability $P(S_{1:T}, E_{1:T}|X_{1:T})$ requires searching over all skeletons over all time simultaneously. In order to make the solution more computationally tractable, we make the assumption that S_t and E_t are independent of $S_{t'}$ and $E_{t'}$ $\forall(t' \neq t)$, given X_t:

$$P(G, S_{1:T}, E_{1:T}|X_{1:T}) \approx P(G|S_{1:T}, E_{1:T}, X_{1:T}) \cdot \prod_{t=1}^{T} P(S_t, E_t|X_t) \quad (3)$$

This assumption can lead to temporal inconsistencies that can be resolved once G is estimated (as shown in Section 4.2). We use a bottom-up approach that individually approximates each S_t and E_t individually, and then estimates G. Ideally, we would like to estimate G, S, and E using an EM-like algorithm by iterating back and forth between estimates of G and (S_t, E_t). However, we have found that the greedy estimate of S and E, while noisy, is sufficient to determine a G consistent with the subject's limb topology.

4 Temporally Constrained Branching Spines

In this section, we will start by describing our method for locating the set of end effectors E_t and extracting a branching skeleton graph from a single 3D surface X_t. Using this or other techniques, we can generate an individual skeleton S_t at each time t, $1 \leq t \leq T$. These (S_t, E_t) will be inherently noisy, as a result of being calculated independently for each t. In Section 4.2, we describe how we combine these individual and often overly complex graphs into a consistent, representative spine for the entire time sequence.

The fairly significant attention given to the problem of building a single branching 3D skeleton includes numerous approaches. After experimenting with portions of several of these [20, 15], we have developed our own extension to the level-set method of [27]. In theory, any 3D skeleton-finding technique would be suitable, if it meets the following requirements:

1. Is self-initializing by automatically finding extrema E_t.
2. Generates a principal curve leading to each extremum.
3. Constructs internal junctions of curves only as necessary to make a connected tree.

More precision might be achieved with more iterations or other techniques, but these might only further improve the results of applying our general probabilistic framework of (3). We proceed to explain our greedy method for obtaining a 3D branching skeleton S_t from a surface, with just one iteration of maximizing (3)'s second term followed by correspondence tracking.

4.1 Creating a Skeleton for a Single Surface

Once we have a 3D surface X_t for volumetric image (or frame) t, we want to extract a skeleton from it. We accomplish this goal in two stages. First we find the tip of each extremity and grow a skeleton from it. Then we merge the resulting skeletons to maximize the presence of the highest quality portions of each. In terms of maximizing $P(S_t, E_t|X_t)$, we are first finding a set of candidates for the end effectors of E_t and the limbs of S_t. We then pick from these the combination that is optimal with respect to our probability metric.

Growing Skeletons: This part of our algorithm is based on the work of [27]. Starting at a seed point on an extremity of the mesh, they sweep through the surface vertices, labelling each with its increasing geodesic distance. These distances are treated as a gradient vector field, which is in turn examined for topological critical points. The critical points are used as surface attachment sites for virtual links (non-centered) between the axes when the mesh branches.

But for our purposes, we want a skeleton that always traverses through the middle of the subject's extremities. Locating meaningful extremal points is itself an open problem, though the difficulties are generally application specific. Much like the above algorithm which has one source, the vertices of a surface mesh can be labelled with their *average* geodesic distance (AGD) to *all* other points. Surface points thus evaluated to be local extrema of the AGD function correspond to protrusions. Knowledge of the expected size of "interesting" protrusions can be used as a threshold on which local maxima qualify as global extrema.

Hilaga et al. [15] address the significant computational cost of finding the AGD by approximating it with uniformly distributed base seed-points. Applying the simpler base-point initialization of [27, 10] in a greedy manner located the desired candidates for E_t for our data sets.

Instead of the separate *distance* and *length* terms minimized by [7], we use the isocontours of geodesic distance to build level sets that serve as our error metric. The vertices of the mesh are clustered into those level-sets by quantizing their distances from the seed point into a fixed number of discrete bins (usually 100). Figures 2C-D illustrate this process. Each skeleton node is constructed by minimizing the distance between the vertices in the level set and the node, *i.e.,* the centroid of the vertices.

By walking along edges of the surface graph from the seed point's level set toward the last one, skeleton-nodes are added and progressively connected to each other. Figure 3A illustrates this process in 2D. This approach successfully creates a tree graph of nodes, or skeleton, which represents the central axes and internal branching points of genus zero meshes.

The skeleton-generation algorithm is repeated for each of the other limb-tips, producing a total of five skeleton-graphs for the starfish example (see Figure 2). These are our candidates for the best S_t for this X_t. Note that the most compact level-sets usually appear as tidy cylindrical rings on the limb where that respective skeleton was seeded.

Merging Skeletons: All of the constituent skeletons S_t serve as combined estimates of the mesh's underlying limb structure. The best representation of

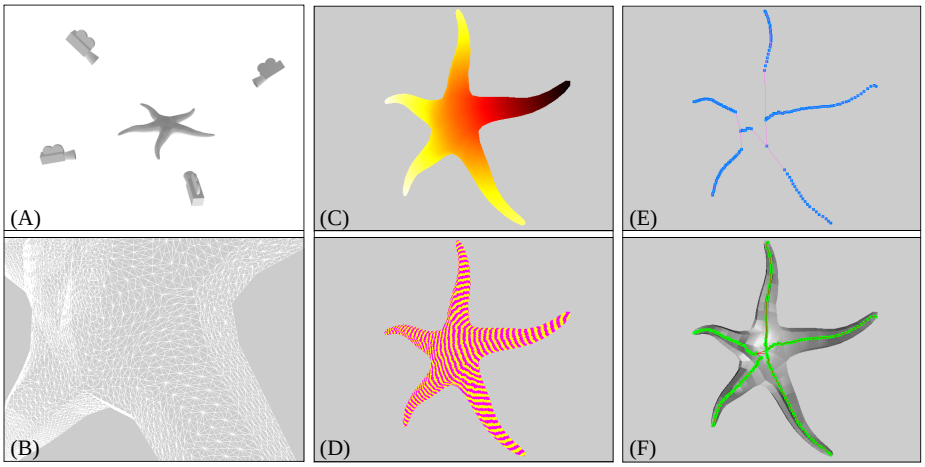


Fig. 2. Example of generating a skeleton for a synthetic starfish mesh. (A) Capture images of the starfish from a variety of vantage points (B) Extract a 3D surface using generalized voxel carving and improved marching cubes (C) Starting at one extremity tip, calculate geodesic distances for each vertex (D) Quantize distances and cluster vertices into bins of the same distance (E) Create a skeleton by walking through the progression of level set rings (F) Repeat C-E for each tip and merge into a single representative skeleton.

that structure comes from unifying the most precise branches of those skeletons – the ones with smallest error, or equivalently, maximum $P(S_t, E_t|X_t)$. A high quality skeleton node best captures the shape of its "ring" of vertices when the ring is short and has small major and minor axes. With this metric, we calculate a cost function C for each node in the constituent skeletons:

$$C_i = \frac{\sigma_1^2 + \sigma_2^2 + \sigma_3^2}{\#\text{ of points in ring } i}. \tag{4}$$

The σ quantities come from singular values of the decomposition $\bar{\mathbf{P}} = \mathbf{U}_P \Sigma_P \mathbf{V}_P^T$, where $\bar{\mathbf{P}}$ represents the mean-centered coordinates of the points p_i in this ring. Note that the resulting $\mathbf{v}_i$ vectors in $\mathbf{V}_P^T = \{\mathbf{v}_1|\mathbf{v}_2|\mathbf{v}_3\}^T$ will usually represent the ring's major, minor, and central axes. Replacing $\mathbf{v}_3$ with $\mathbf{v}_1 \times \mathbf{v}_2$ produces a convenient local right-hand coordinate frame for each node.

Each chain of bi-connected nodes represents a limb. To assemble the single representative graph of this frame, we copy the best version of each limb available in the constituent skeletons. Limb quality $\mathbf{Q}_L$ is measured as:

$$\mathbf{Q}_L = N - \sum_1^N C_i, \tag{5}$$

where N is the total number of nodes in limb L. Since nodes from different skeletons are being compared through (5), the C_i's must be normalized by dividing them all by the $\max(C_i)$ of all the skeletons.

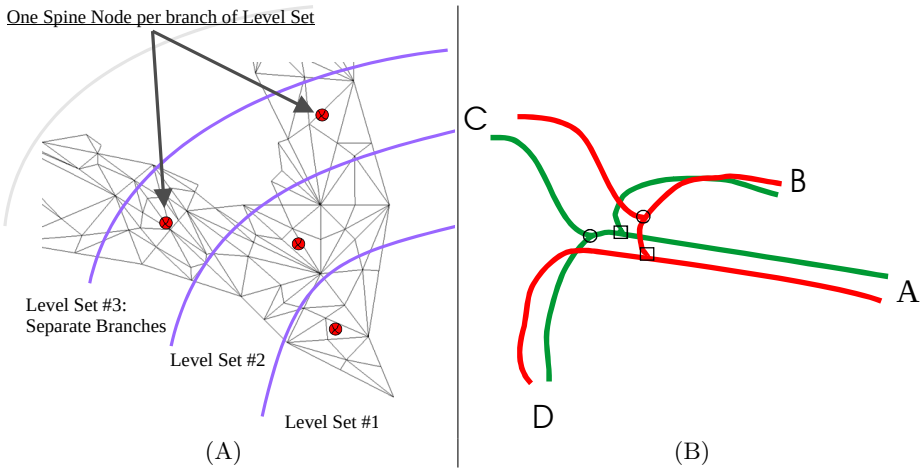


Fig. 3. (A) 2D example of clustering connected vertices into bins of similar geodesic distance and walking through the resulting level set rings. (B) In the right figure, the red and green skeletons represent the same "creature," possibly seeded from two different places. Wishing to copy nodes from the best limbs each constituent skeleton has to offer, we developed a leaf-node seeking topology matching algorithm that recognizes that these pairs of three-way junctions should be a single four-way junction.

Figure 3B illustrates a novel algorithm that we developed to generate limb-correspondences for topologically perturbed tree graphs of the same structure. There appears to be no previously established graph theoretic solution for this problem, and our approach is simply:

1. Tag all limb-tips that we are confident of as *Supernodes*; i.e. nodes on both color graphs located at [A, B, C, D] correspond to each other.
2. Traversing inward, the next encountered branch-node in each graph also corresponds to that of the other color: walking from supernode A, the skeleton-nodes at the square-symbols should be grouped into a supernode of their own. From C, the circles will form a supernode. Iterating this process from the outside inward will reveal that the circle and square supernodes should be merged into a four-way *metanode*, which would serve as the point of unification when merging limbs from the red and green skeletons.

4.2 Correspondence Tracking

Now that we can estimate a single skeleton that represents one volumetric image, we adapt the process to handle a sequence of volumes. All the measurements from the sequence of $X_{1:T}$ are now abstracted as $(S_{1:T}, E_{1:T})$, simplifying the first term in (3) to $P(G|S_{1:T}, E_{1:T})$. Finding the G that maximizes this probability eliminates extraneous limbs which might have resulted from overfitting. The danger of overfitting exists because skeleton elements may be created in support of surface-mesh elements that looked like protrusions in that frame only.

Our 3D correspondence problem of finding the best G is significantly easier to automate than trying to perform surface-vertex matching between two dense meshes of the sequence. Assuming the subject grows no new appendages and with no other priors, we can safely choose the appropriate number of tips to be the most frequently observed number of limb tips. This number of tips, or leaf nodes in G, is K = the mode of $|E_t|$, $1 \leq t \leq T$ (see Figure 7).

Knowing how many appendages to look for, we spatially align each exploratory skeleton from the sequence with respect to its temporal neighbors to reveal the $|E_t| - K$ superfluous tips that should be culled. We start with all the subsequences of frames that already have the correct number of tips K, and tag the frame from the middle of the largest such cluster as the reference frame; allowing that longer sequences may need to automatically select multiple reference frames. Each frame is then processed in turn, constructing a combinatorial list of possible tip-correspondences between the reference tips $\mathbf{A}$ and the tips in the current frame $\mathbf{B}$. Each possible mapping of $\mathbf{B} \rightarrow \mathbf{A}$ is evaluated using the point-cluster alignment algorithm of [1]. Their technique aligns point clouds as much as possible using only translation and rotation. The combination with the smallest error, $E_{\min}$, is kept as the correct assignment, where

$$E = \sum_{k=1}^{K} \|B_k - \hat{\mathbf{R}} A_k - \hat{\mathbf{T}}\|^2. \tag{6}$$

Here $\hat{\mathbf{R}}$ and $\hat{\mathbf{T}}$ are the least-squares optimal rotation and translation. $\hat{\mathbf{T}}$ simply comes from the alignment of the point clouds' centroids. $\hat{\mathbf{R}}$ is calculated by maximizing the $Trace(\hat{\mathbf{R}}\mathbf{H})$, where $\mathbf{H}$ is the accumulated point correlation matrix:

$$\mathbf{H} = \sum_{k=1}^{K} A_k B_k^T. \tag{7}$$

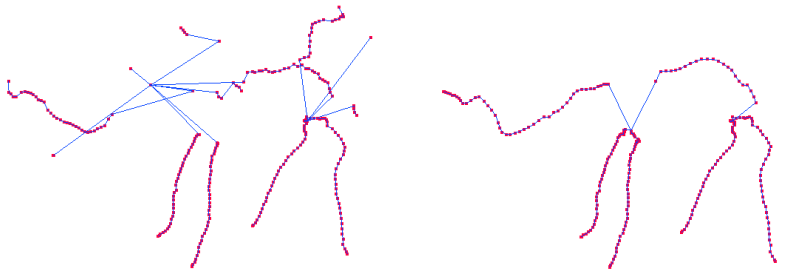
By decomposing $\mathbf{H} = \mathbf{U}_R \Sigma_R \mathbf{V}_R^T$, the optimal rotation is:

$$\hat{\mathbf{R}} = \mathbf{V}_R \mathbf{U}_R^T. \tag{8}$$

After assigning the tips of all these frames, we apply the same error metric to try out the combinations of tip-assignments with frames having alternate numbers of tips. However, these frames are compared to both the reference frame and the frame nearest in time with K tips. This brute-force exploration of correspondence is computationally tractable and robust for creatures that exhibit some asymmetry and have a reasonable number of limbs (typically < 10).

4.3 Imposing a Single Graph on the Spine

With the known trajectories of corresponding limb tips throughout the sequence, we can re-apply the skeleton merging technique from Section 4.1. This time however, we do not keep all the limbs as we did in the exploratory phase, only



A: Without refinement B: With temporal constraint

Fig. 4. Refinement through imposing of correspondence into the sequence.

those that correspond to the K limb-tips. The results of this portion of the algorithm are pictured in Figure 4 and discussed further in Section 5.

Except for the frames of the sequence where the subject's limbs were hidden or tucked too close to the body, we can expect the topology of skeletons throughout the sequence to be identical. The most frequently occurring topology is established as G, and corresponds to the first term in 3. This correspondence and trajectory information allows us to construct a single character spine for playback of the whole sequence of poses by parameterizing on each limb's length. Each topologically consistent limb of the skeleton sequence is resampled at the same interval producing a single spine.

5 Experiments & Results

We tried our algorithm on a variety of small creatures after building a data-capture stage that would both be comfortable for our subjects and minimize the need for video segmentation beyond chromakeying. Twenty video cameras were attached to an aluminum exoskeleton shaped roughly like a cylinder 3 meters in diameter. Their viewing angles were chosen heuristically to maximize viewing coverage and to minimize instances of cameras seeing each other's lenses. The capture volume itself is $(75cm)^3$, and can accommodate creatures that stay within the space (Figure 5). Our subjects often required human proximity and were too heavy for our transparent flooring, so we were only able to leverage a subset of the cameras present.

With this setup, we are able to obtain video from a dome of inward facing, calibrated and synchronized cameras [29, 6]. This allowed us to employ the Generalized Voxel Carving (GVC) algorithm of [11]. Their system functions as a hybrid form of wide-baseline stereo and voxel-carving, enabling the resulting voxel model to reflect concavities found on parts of the subject's surface. Each second of multi-view footage produces 30 voxel models similar to the system of [26].

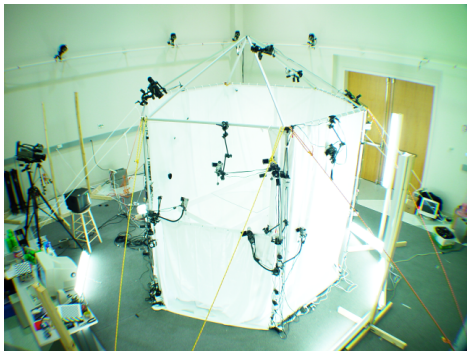


Fig. 5. Our Capture Setup: Twenty video cameras were attached to an aluminum exoskeleton shaped roughly like a cylinder 3 meters in diameter. Their viewing angles were chosen heuristically to maximize viewing coverage of subjects raised in the middle, and to minimize instances of cameras seeing each other's lenses. The capture volume itself is $(75cm)^3$.

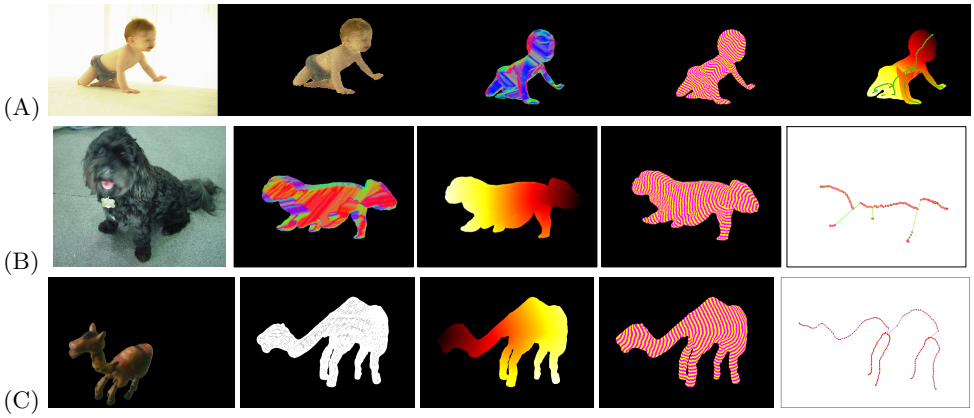


Fig. 6. (A) Baby dataset: From left to right, one of the views, voxels, polygonal model, level sets, and skeleton with distance function. (B) Dog dataset: subject, polygonal model, distance function, level sets, and resulting spine. (C) Camel Puppet dataset: one view, wireframe, distance function, level sets, and resulting spine.

5.1 Real Subjects

Baby: The baby data is the result of filming an 11-month old infant using nine cameras. The sequence is 45 frames long because that was the speed with which she crawled down the length of the stage. Her progress forward is mostly due to her arms and right leg, while she tends to drag her left leg which causes frequent merging of her voxel-model from the waist down. The spine generation models her head and arms very consistently, but the correspondence tracker cannot resolve her legs and mis-assigns one leg or the other for the majority of frames.

Dog: The dog was the most challenging of our test-subjects simply because we had only seven cameras that could operate without also filming the dog's handlers. The volume reconstructions are all close to their average of 1.04M voxels. Examination of the polygonal-mesh sequence reveals that much of this bulk comes from the ghost-voxels under his stomach that were carved successfully in the previous and subsequent test subjects when more cameras were running.

Camel Puppet: The camel marionette, pictured in Figure 6C, is 26 cm long and stretches to a height of 42 cm. While the subject didn't change in volume throughout shooting, its representation varied throughout the sequence between 600k and 800k voxels, largely due to self-occlusions. The polygonal representations averaged 200k polygons. The sequence has 495 frames, and was filmed using 12 color cameras. The camel's motion changes in the sequence from leg-jostling at the start to vigorous kicking and raising of the neck by the end. Our system was only hindered by the occasional "merging" of legs as they tucked underneath or appeared close enough to each other to be joined in the voxel stage. With mostly good frames, the exploratory skeleton-generation fed the correspondence tracker, which in turn determined that there were five limbs. The resulting creature spine is pictured in Figure 4B. As illustrated,the correspondence tracking balances out the greedy limb inclusion of the exploratory skeletons. The online video also demonstrates this.

The average processing times for skeleton-generation using our unoptimized implementation of the algorithms were consistently under four minutes per mesh on a Pentium 4 PC with one or more GB of memory. The correspondence-tracking portion of our algorithm (Section 4.2) took ten minutes on our 495 frame camel sequence, and less than three minutes on all our other sequences. The preprocessing stage leading to input meshes is an implementation of GVC that adds approximately 12 minutes to each frame, with 3-8 seconds for Marching Cubes. GVC is not part of our contribution, and can be exchanged for other dense stereo or silhouette-carving algorithms, some of which may, though we have not yet tested this, have superior run-time performance without impacting quality. We have data of other example subjects that will be posted on our website, and the volumetric data has already been shared with other researchers.

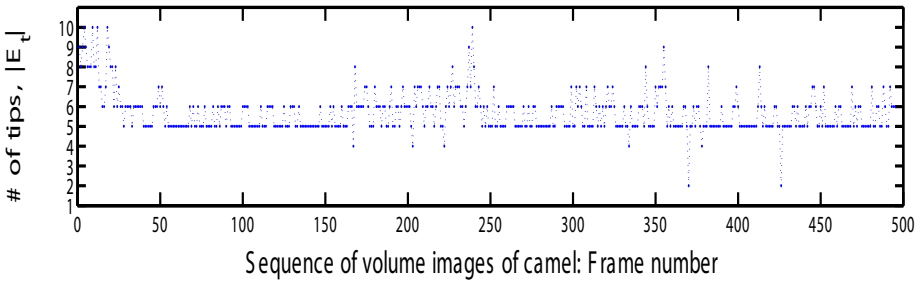


Fig. 7. Number of skeleton tips found per-frame during greedy search.

6 Conclusion and Future Work

We have proposed *spines* as a novel 3D spatio-temporal representation for sequences of volume images. This shape and motion descriptor introduces a method for imposing temporal correspondence on limb topologies when dealing with articulated subjects. We also present an algorithm for efficiently extracting branching spines from surface data. Finally, we have presented example data where the temporally integrated canonical graph improves the quality of individual skeletons.

Where the current fully bottom-up work leaves off, extensions are planned that will allow a prior skeleton estimate to be forced on the data. This will especially apply to meshes where the limbs tuck in or become genus 1+. While the current results reflect that fairly noisy data, without priors, still reveals the real end effectors and underlying structure, further work is needed to track pose even in very poor data.

7 Acknowledgements

The authors are grateful to Greg Slabaugh and Hewlett-Packard Laboratories for his assistance and for sharing their GVC code. Data capture and processing was possible thanks to the assistance provided by Jonathan Shaw, Steve Park, Stephen Du, Anil Rohatgi, and the indomitable Spencer Reynolds. We also thank Bella Steedly as the baby, Hilary and Davis King for bringing their dog Barnaby, and Odest Chadwicke Jenkins, Quynh Dinh, and the anonymous reviewers.

References

1. Arun, K. S., Huang, T. S., and Blostein, S. D. 1987. Least squares fitting of two 3-d point sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-9*, 2 (March), 698–700.
2. Attali, D. and Montanvert, A. 1997. Computing and simplifying 2d and 3d continuous skeletons. *Computer Vision and Image Understanding* **67***(3): 261–273.*
3. Betelu, S., Sapiro, G., Tannenbaum, A., Giblin, P. J. *2000. Noise-resistant affine skeletons of planar curves,* ECCV00*, pp. I: 742–754.*
4. Binford, T. *1987 (first presented in 1971). Generalized cylinder representation,* Encyclopedia of A. I.*, John Wiley & Sons, pp. 321–323.*
5. Blum, H. *1973. Biological shape and visual science (part I),* Journal of Theoretical Biology **38***: 205–287.*
6. Bradksi, G., and Pisarevsky, V. *2000. Intel's computer vision library: Applications in calibration, stereo, segmentation, tracking, gesture, face, and object recognition. In* Proceedings of IEEE CVPR 2000*, vol. II, II:796–797. Demonstration Paper.*
7. Cao, Y. *2003.* Axial Representations of 3D Shapes*, PhD thesis, Brown University.*
8. Cao, Y., and Mumford, D. *2002. Geometric structure estimation of axially symmetric pots from small fragments,* Proc. IASTED SPPRA.
9. Chu, C., Jenkins, O., and Mataric, M. *2003. Markerless kinematic model and motion capture from volume sequences,* CVPR03*, pp. II: 475–482.*

10. Cormen, T. H., Leiserson, C. E., and Rivest, R. L. *1990.* Introduction to Algorithms. *MIT Press/McGraw-Hill.*
11. Culbertson, W. B., Malzbender, T., and Slabaugh, G. *1999. Generalized voxel coloring. In* ICCV Vision Algorithms Workshop*, Springer-Verlag, no. 1883 in LNCS, 100–115.*
12. Dey, T. K., and Zhao, W. *2002. Approximate medial axis as a voronoi subcomplex,* Proceedings of the Seventh ACM Symposium on Solid Modeling and Applications*, ACM Press, pp. 356–366.*
13. Ferley, E., Cani, M.-P., and Attali, D. *1997. Skeletal reconstruction of branching shapes,* Computer Graphics Forum **16***(5): 283–293.*
14. Hastie, T., and Stuetzle, W. *1989. Principal curves,* Journal of the American Statistical Association **84***: 502–516.*
15. Hilaga, M., Shinagawa, Y., Kohmura, T., and Kunii, T. L. *2001. Topology matching for fully automatic similarity estimation of 3d shapes. In* Proceedings of ACM SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, 203–212.*
16. Hubbard, P. M. *1996. Approximating polyhedra with spheres for time-critical collision detection.* ACM Transactions on Graphics 15*, 3 (July), 179–210.*
17. Katz, S., and Tal, A. *2003. Hierarchical mesh decomposition using fuzzy clustering and cuts,* ACM Transactions on Graphics **22**.
18. Kégl, B., Krzyżak, A., Linder, T., and Zeger, K. *2000. Learning and design of principal curves,* IEEE Transactions on Pattern Analysis and Machine Intelligence **22***(3): 281–297.*
19. Leymarie, F. F., and Kimia, B. B. *2001. The shock scaffold for representing 3d shape,* in *G. S. d. B. C. Arcelli, L.P. Cordella (ed.),* Visual Form 2001*, number LNCS 2059 in* Lecture Notes in Computer Science*, Springer-Verlag, pp. 216–229.*
20. Li, X., Toon, T. W., and Huang, Z. *2001. Decomposing polygon meshes for interactive applications. In* Proceedings of the 2001 Symposium on Interactive 3D graphics*, ACM Press, 35–42.*
21. Marr, D., and Nishihara, H. *1978. Representation and recognition of the spatial organization of three-dimensional shapes,* Proc. of the Royal Society of London, series B*, Vol. 200, pp. 269–294.*
22. Nain, D., Haker, S., Kikinis, R., and Grimson, W. E. L. *2001. An interactive virtual endoscopy tool,* Workshop on Interactive Medical Image Visualization and Analysis satellite symposia of MICCAI, IMIVA'01*, Utrecht, The Netherlands.*
23. Sebastian, T. B., Klein, P. N., and Kimia, B. B. *2001. Recognition of shapes by editing shock graphs. In* ICCV*, I: 755–762.*
24. Siddiqi, K., Bouix, S., Tannenbaum, A., and Zucker, S. W. *2002. Hamilton-jacobi skeletons.* IJCV 48*, 3 (July/August), 215–231.*
25. Teichmann, M., and Teller, S. *1998. Assisted articulation of closed polygonal models. In* Proceeding of Eurographics Workshop on Computer Animation and Simulation 1998.
26. Vedula, S., Baker, S., Seitz, S., and Kanade, T. *2000. Shape and motion carving in 6D. In* Proceedings of Computer Vision and Pattern Recognition (CVPR2000)*, 592–598.*
27. Verroust, A., and Lazarus, F. *2000. Extracting skeletal curves from 3d scattered data.* The Visual Computer 16*, 1.*
28. Wade, L., and Parent, R. E. *2002. Automated generation of control skeletons for use in animation,* The Visual Computer **18***(2): 97–110.*
29. Zhang, Z. *1998. A flexibe new technique for camera calibration. Tech. Rep. 98-71, Microsoft Research. www.research.microsoft.com/∼zhang/Calib/.*