Computers & Graphics 35 (2011) 112–121

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Technical Section

Exoskeleton: Curve network abstraction for 3D shapes

Fernando de Goes [a,*], Siome Goldenstein [b], Mathieu Desbrun [a], Luiz Velho [c]

[a] *California Institute of Technology, Pasadena, CA 91125, USA*
[b] *Instituto de Computação, UNICAMP, Caixa Postal 6176, 13084-971 Campinas, SP, Brazil*
[c] *IMPA, Instituto de Matemática Pura e Aplicada, Estrada Dona Castorina 110, 22460 Rio de Janeiro, RJ, Brazil*

ARTICLE INFO

Article history:
Received 16 August 2010
Received in revised form
13 November 2010
Accepted 18 November 2010
Available online 25 November 2010

Keywords:
Shape abstraction
Segmentation
Shape analysis

ABSTRACT

In this paper, we introduce the concept of an *exoskeleton* as a new abstraction of arbitrary shapes that succinctly conveys both the perceptual and the geometric structure of a 3D model. We extract exoskeletons via a principled framework that combines segmentation and shape approximation. Our method starts from a segmentation of the shape into perceptually relevant parts and then constructs the exoskeleton using a novel extension of the Variational Shape Approximation method. Benefits of the exoskeleton abstraction to graphics applications such as simplification and chartification are presented.

1. Introduction

Shape abstraction is a topic common to art and science. Not surprisingly, computer graphics has investigated shape abstraction intensively for a wide range of purposes, including animation [1,2], shape matching and retrieval [3], modeling [4,5], editing [6,7], and non-photorealistic rendering [8,9].

Abstracting a shape amounts to constructing a depiction that reveals the shape's key structural parts [10]. Existing techniques address the extraction of shape abstractions in two complementary trends: *geometry-driven approaches* produce detailed shape abstractions by either extracting feature lines or fitting geometric primitives, while *perceptually driven approaches* detect meaningful spatial parts used to derive skeleton-like representations. Notably lacking is a shape abstraction that combines the advantages of both geometric and perceptual methods, resulting in a representation that contains both local (feature) and global (structural) information (Fig. 1). This intermediate abstraction would however have numerous benefits in graphics applications, including mesh simplification, user-guided texture mapping, structure-aware remeshing, and high-order surface fitting.

1.1. Goal and contributions

The goal of this work is to construct shape abstractions that concisely capture both the global structure and the geometric complexity of 3D shapes. To this end, we present a principled framework that combines part-based segmentation and geometric approximation. The resulting abstraction consists of a sparse network of curves bounding disk-like patches. We call such a network of curves an *exoskeleton* since it resembles the skeleton of a shape except instead, it lies *on* the surface (Fig. 2). We perform the abstraction of a shape in two stages: we first segment the input mesh to identify structural parts; we then use these parts as priors to guide the extraction of a network of curves. Our extraction algorithm is implemented as a simple, yet powerful extension of the variational shape approximation (VSA) method [11] that weighs principal curvature lines based on structural priors—we will denote this novel structure-aware VSA method "S-VSA" (Figs. 6 and 7).

We will use the term *perceptual* in our work to stress that our approach is inspired by research in perception, in particular by the idea that the human visual system understands shapes in terms of parts [14–16]. Guided by this observation, we design our technique by exploiting parts of a shape as a means to encode its global structure. While we will show that the output of our algorithm seems to agree with human visual perception, we expect that each component of our approach can be revised or reworked as more becomes known about how people understand shapes.

Our method is general and can handle man-made as well as organic shapes. While man-made models usually have a few dominant sharp features that can be conveniently used for shape abstraction [17], general shapes can be abstracted in many different ways. For instance, the torso in Fig. 4 can be abstracted by stressing either its bounding box or its symmetry planes. The decision of which abstraction to use is very domain-dependent. To deal with this complexity of shape abstraction, we introduce a *select and propagate* scheme that extracts features part by part.

[*] Corresponding author.
E-mail addresses: fdegoes@caltech.edu (F. de Goes), siome@ic.unicamp.br (S. Goldenstein), mathieu@caltech.edu (M. Desbrun), lvelho@impa.br (L. Velho).

0097-8493/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cag.2010.11.012

1.2. Outline

In the remainder of the paper, we first discuss previous work on shape abstraction and introduce the key definitions and concepts that underlie our approach. We then present the algorithm to build exoskeletons and showcase how our shape abstraction can depict very different classes of shapes. Finally, we illustrate how the exoskeleton benefits traditional partitioning and simplification schemes as example applications.

2. Related work

To motivate our approach, we present a brief overview of the various techniques related to shape abstraction that have been proposed over the years.

Artworks: Artists such as Alexander Calder and Pablo Picasso have used networks of wires and strokes to create abstractions of 3D shapes. These networks, in general, describe a shape by outlining its structural curves, including silhouettes, joints and limbs. Our work draws motivation from artworks and depicts a 3D shape by combining structural regions and principal curvature lines.

Conveying shapes: Non-photorealistic rendering methods have introduced a wealth of visual cues to help depict shapes [8,9]. Cues can be either view-dependent such as silhouettes [18], suggestive contours [19] and highlights [20] or intrinsic such as ridges and valleys [21,13,22]. In both cases, feature lines are computed individually generating an unstructured and scattered set of curves. Our method, on the other hand, proceeds globally by extracting interconnected curves that balance out local features and global structure of the shape.

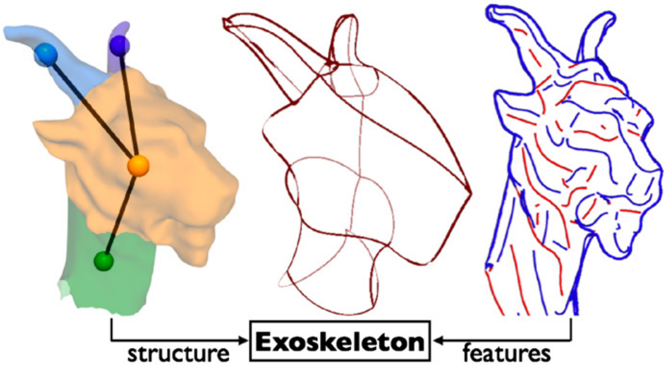

Fig. 1. *Abstraction*. Existing techniques abstract 3D shapes either by highlighting its structural parts [12] (left) or detecting scattered feature lines [13] (right). Our exoskeleton strikes a balance between global structure and geometric complexity through a concise curve network abstraction.

Approximation: Shape approximation is a common approach to provide a description with a small number of geometric elements. The QSlim method [23], for instance, simplifies a shape by collapsing edges in regions of low curvature. Other methods proceed in a multiresolution fashion [24–26] or by face clustering [27–29]. The VSA method [11], in particular, performs a global energy minimization by fitting planes that best match the shape anisotropy and has been successfully applied for the abstraction of man-made objects with few predominant sharp features [17]. However, approximation techniques are poorly descriptive when a very low polygon count is reached: Lee et al. [30] point out this limitation and propose a perception-based saliency measure to achieve better visual approximations. Our approach mixes part-based segmentation and geometric features to achieve a similar goal.

Atlases and remeshing: Other topics within computer graphics have proposed various approaches to shape abstraction as a means to achieve particular applications. For instance, a large number of techniques have been proposed to generate a texture atlas that resembles how artists would intuitively cut the shape, hence segmenting the surface in a particularly relevant manner. Most approaches partition the surface guided by parameterization distortion measures [31–34]. Semantic information is often added through user interaction to unwrap a surface [35,36], or extracted automatically in the restricted case of coarse quadrilateral input meshes [37]. Similarly, remeshing often targets properties such as orientation, alignment and regularity as a means to recover meaningful and visually pleasing features of a shape [38–41]. However, most of these methods require significant user input to guide the remeshing process or they fail to capture the structure of a shape with very low polygon count.

Skeletons and part-based segmentation: In the last decade, a broad variety of techniques has been presented to extract skeletons from 3D shapes [42]. Examples of such techniques include mesh contraction [2], Reeb graphs [43], and volumetric analyses [44]. Particularly relevant to our approach, skeletons can be extracted from part-based segmentation methods [45,46]. These methods decompose shapes into meaningful parts, mimicking how our vision identifies salient parts. However, the perceptual relevance of a region is highly subjective and application-dependent. Consequently, different categories of objects have received distinct segmentation criteria. Articulated bodies, for example, can be decomposed into rigid parts, while furniture and cutlery are usually segmented into convex components. In order to exploit the growing spectrum of segmentation criteria, we let the user pick *any* segmentation method as the building block of our algorithm—and then enrich it by constructing an exoskeleton.

3. Definitions and concepts

As our goal is to provide 3D shape abstractions that strike a balance between compactness (number of curves), global structure

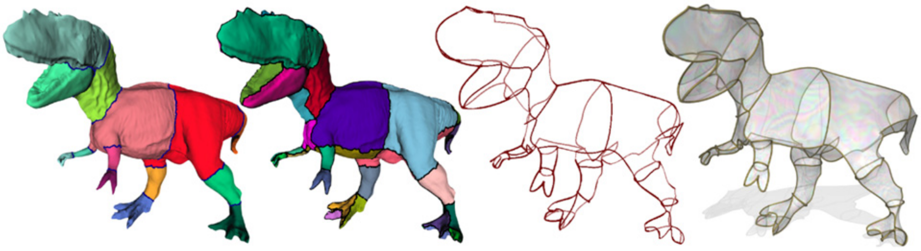

Fig. 2. *Exoskeleton*. Building upon a part-based segmentation of a 3D model (left), our approach decomposes a shape into patches through S-VSA, a novel structure aware extension of the Variational Shape Approximation method [11] (center left). The boundaries of the resulting patches form the *exoskeleton* of the shape (center right), a high-level abstraction that concisely conveys the defining characteristics of the input geometry. This exoskeleton proves to be enough to capture the structure of the model, thus providing a good handle for further editing, texturing, and animation. From the exoskeleton alone, a soap film attached to it (i.e., a minimal surface inside each patch) closely resembles the initial geometry (right).

and expressiveness, we first need to discuss the various notions of features that will drive the construction of exoskeletons.

3.1. Types of features

Predominant features of a given shape are commonly recognized as singularities in the normal field of the surface. These geometric features are closely related to principal curvature lines and indicate local anisotropy. Examples include sharp features, ridges, valleys, and separatrices [21,13,22]. Hereafter we refer to this type of features as *normal-based features*.

Other characteristic curves enclose structural regions and are equally important to our apprehension of a shape. Their exact definition, however, is perceptually driven and depends on the targeted application. For this reason, we refer to them as *perceptual features*. In articulated bodies, for example, these curves correspond to body articulations and can be computed using rigidity measures [47]; for mechanical parts, they contour shape primitives and can be identified through geometric fitting [28,29]. Perceptual features are, more generally, shape characteristics whose relevancy cannot be detected solely by normal discontinuities and thus require either additional geometric criteria or a priori knowledge.

3.2. Notion of shape exoskeleton

Our approach abstracts 3D shapes by merging perceptual and normal-based features into a network of curves we call the *exoskeleton*. Intuitively speaking, this abstraction is similar to a shape skeleton as it provides a concise representation of the shape, but with the significant difference that it lies upon the surface, providing a *scaffolding* of the relevant regions. An exoskeleton strikes a balance between local and global descriptions: at a fine scale, the exoskeleton depicts shape details using features such as prominent ridges, valleys, concavities and protrusions; at a global scale, however, the exoskeleton preserves the shape structure, connecting relevant features and leaving out local idiosyncrasies (see Fig. 1).

The construction of an exoskeleton seeks a compromise between expressiveness and simplicity of the shape abstraction. We balance these conflicting aspects inspired by evidence from cognitive science [14–16] that human vision recognizes shapes in term of perceptual parts. Our work starts with this simple premise, and computes a network of features in two sequential stages: we first derive *perceptual features* from boundaries of structural parts, and we then detect *normal-based features* connecting adjacent perceptual features.

3.3. Exoskeleton properties

In order to enhance the structure of our exoskeletons, we define a small set of properties we wish to respect. We form a shape exoskeleton through extraction of normal-based features that favor *orientation*, *regularity*, *continuity*, and *orthogonality* of the resulting network (Fig. 3):

- The *orientation* condition indicates that normal-based features should be mostly along principal curvature lines, to highlight the anisotropy of the surface.
- *Regularity* imposes that normal-based features must be well-spaced and adapted to the local surface anisotropy. This condition results in a density of features proportional to the local complexity of the shape.
- *Continuity* favors the placement, in the network structure of the exoskeleton, of long curves that encode the structure of the shape. We promote continuity by biasing our placement of normal-based features based on their positions in adjacent perceptual parts, i.e., we try to extend existing features as much as possible.
- The *orthogonality* condition controls the angle in which normal-based features intersect perceptual features. In order to convey a strong visual cue of the shape and its volume, the two types of features are forced to be as orthogonal as possible to each other, thus forming a *near-orthogonal* network.

It is worth pointing out that the properties above are simply one possible model to guide the extraction of curve networks. In our approach, these properties proved to be adequate as they result in visually pleasing exoskeletons.

4. Algorithm

We now describe our framework to construct exoskeletons of 3D shapes (see Fig. 5 and the supplemental video for an overview of the basic flow of our method). Our approach is efficient and flexible, fulfilling the properties we previously mentioned in two stages: we first segment the shape to find the main perceptual features; based on these perceptual features, we then extract the exoskeleton through a variant of the VSA algorithm.

4.1. Perceptual features via part-based segmentation

The first stage of our algorithm detects perceptual features. In the last decade, a large spectrum of approaches has been proposed

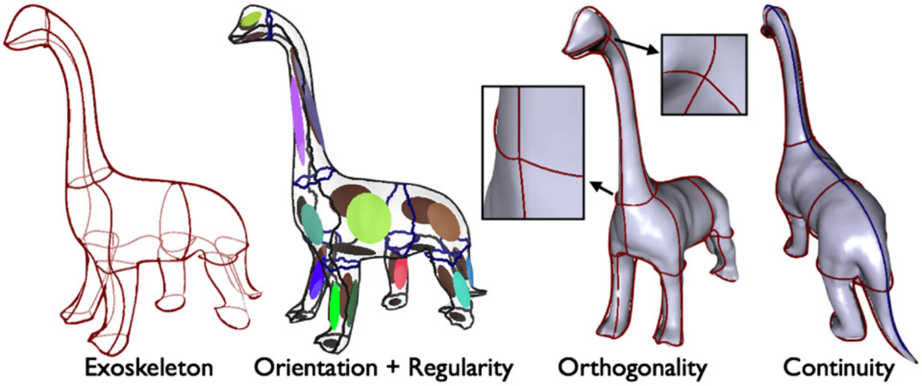

Fig. 3. *Exoskeleton properties*. Left to right: Abstraction of the *dinosaur* model; *Orientation* and *Regularity*: patches are formed following the shape anisotropy, placing well-spaced features oriented to curvature lines. *Orthogonality*: Normal-based and perceptual features are preferentially placed in orthogonal directions. *Continuity*: Exoskeleton curves do not exhibit undesired visually distracting fragmentation (as demonstrated by the spine-aligned curve), and thus highlight the global structure of the shape.

to extract salient regions of various categories of 3D models. As the notion of perceptual features is highly subjective and depends on the application being sought, a segmentation method typically selects different criteria such as convexity (e.g., for furniture and cutlery), slippage (e.g., for mechanical objects), or rigidity (e.g., for articulated bodies), based on the final goal (see [45,46] for a thorough survey on segmentation). Given these criteria, the segmentation algorithm clusters triangles of a mesh into *parts*. In order not to restrict our shape abstraction framework, we remain agnostic as to which method or criteria are used (and in fact, we will perform our exoskeleton construction based on several different segmentation techniques in Section 5): we simply use the resulting segmented regions as building blocks for the next stage of our algorithm. *Perceptual features* are defined hereafter as the *boundaries between each pair of adjacent segmented regions*.

4.2. Normal-based features via "select and propagate"

The second stage of our algorithm starts from the segmented regions and further extracts normal-based features of the shape. From a given segmentation, one can design various exoskeletons that emphasize different aspects of the shape: for instance, the torso in Fig. 4 can be abstracted either through its bounding box or its symmetry planes. This diversity of abstractions stems from the fact that the relevance of normal-based features is fundamentally application-dependent.

To exploit and disambiguate different abstractions of a shape, we introduce a *select and propagate* strategy that computes normal-based features incrementally by propagating features extracted in a region to its neighboring regions. As shown in Fig. 5, each extraction of normal-based features in a segmented region $\mathcal{R}$ will end up partitioning its perceptual features $\partial\mathcal{R}$ into *arcs* $\mathcal{A}_i$ that we will call *priors*. The set $\mathcal{A}=\{\mathcal{A}_i\}$ of these priors will then be used as boundary conditions for the extraction of normal-based features for the neighboring regions.

Extracting normal-based features within a segmented region is achieved through a modification of the VSA method as described next. Our version combines the set of priors $\mathcal{A}$ with the VSA (normal-based) planarity metric $L^{2,1}$ in order to segment $\mathcal{R}$ into patches that strike a balance of the four global properties of an exoskeleton as listed in Section 3.3; we then identify the *normal-based features* as the *boundaries between pairs of adjacent patches*.

The select-and-propagate strategy introduces an extra degree of freedom to our framework: from a given segmentation, we can now change the order in which the regions are processed to create different exoskeletons. In Fig. 4B, for instance, we handled the regions starting from the extremities to create an abstraction that reveals the symmetry axes of the shape. In Fig. 4A, on the other hand, we reversed the processing order of the segmented regions, resulting in extracted features akin to bounding boxes. Even though an interactive process can easily be implemented to guide this ordering, our experiments have shown that simple automatic heuristics, such as rooting extremity segments and applying a

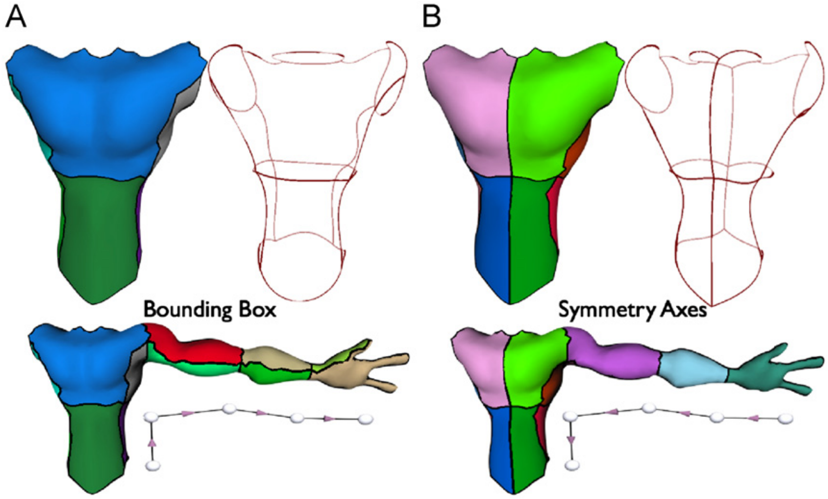

Fig. 4. *Select-and-propagate.* We provide a flexible mechanism to emphasize different aspects of the shape: (A) if the segmented regions are ordered from the abdomen to the hand, the resulting exoskeleton exhibits clothing seams; (B) if the order is reversed, the exoskeleton stresses symmetry axes instead.

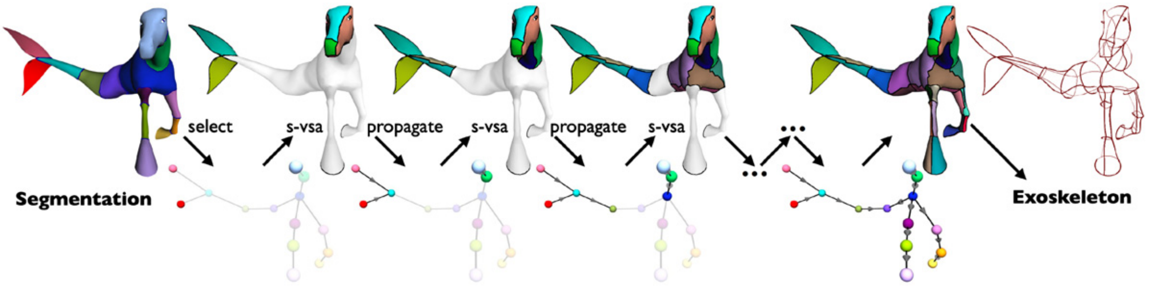

Fig. 5. *Exoskeleton algorithm.* From a segmentation decomposing a shape into perceptual parts (left), our algorithm constructs patches incrementally, organizing the perceptual parts in a directed graph (center). The boundaries of these patches compose a network of curves that follows perceptual and normal-based feature lines (right).

breadth first traversal on the remaining segments, are sufficient to achieve the results presented in the paper (see Section 5 for details).

4.3. Variational shape approximation with perceptual parts

We now discuss how we find normal-based features from the existing priors of a segmented region.

Original VSA method: Variational shape approximation [11] consists in finding the best-fitting planes to approximate a shape by minimizing the $L^{2,1}$ metric between a set of plane proxies and the surface, hence capturing the anisotropy of the shape with optimal asymptotic behavior. Applied to a segmented region, the result decomposes $\mathcal{R}$ into patches $\{\mathcal{P}_i\}$ associated with their proxy planes $\{P_i\}$. Algorithmically, the minimization is tackled by successive iterations alternating shape partitioning and plane fitting. Each iteration reduces the approximation error between patches and planes in three steps. First, each patch $\mathcal{P}_i$ is seeded with the triangle $\mathbf{s}_i$ that best fits the plane P_i. Then a flooding process is launched from the seeds to greedily generate new patches that minimize the metric distortion with respect to the planes. Finally, the planes are updated by computing the closed-form optimal fitting for each patch.

Incorporating priors: Our modification to the VSA method seeks to find patches tiling $\mathcal{R}$ and complying with the global properties of an exoskeleton. We achieve these properties by combining the $L^{2,1}$ metric with the constraints imposed by perceptual features: *while the $L^{2,1}$ metric provides orientation and regularity*, *we use perceptual features as priors to incorporate orthogonality and continuity*. Fig. 6 illustrates the effects of these properties in our S-VSA method.

We represent priors and normal-based features of a triangle mesh by a simple list of adjacent edges. We say that features are *fragmented* when normal-based features do not intersect prior arcs at their endpoints, as they harm the continuity property of the exoskeleton (Fig. 6B). To avoid fragmentation, we must ensure that all edges of a prior belong to the same patch. We incorporate this condition into the VSA method through a harmonic function h_i computed for each prior $\mathcal{A}_i$. Such a function satisfies the Laplace equation with Dirichlet boundary conditions based on $\mathcal{A}_i$:

$$\begin{cases} \Delta h_i(p) = 0, & p \in \mathcal{R} \backslash \partial\mathcal{R}, \\ h_i(p) = 1, & p \in \mathcal{A}_i, \\ h_i(p) = 0, & p \in \partial\mathcal{R} \backslash \mathcal{A}_i. \end{cases} \quad (1)$$

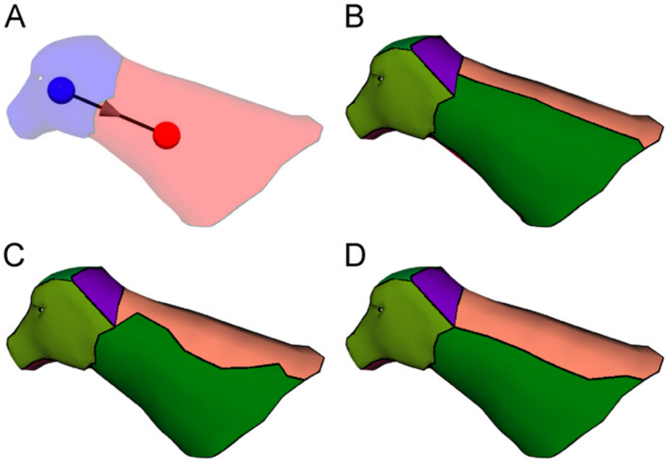

Fig. 6. *S-VSA.* Given segmented parts (A), the VSA method [11] generates fragmented features affecting the global structure of the shape abstraction (B). Our modified S-VSA method recovers the shape structure with two simple operations: we introduce the *attachment step* in order to force the continuity of feature curves (C); and we extend the $L^{2,1}$ metric based on boundary priors to favor near-orthogonal curves (D).

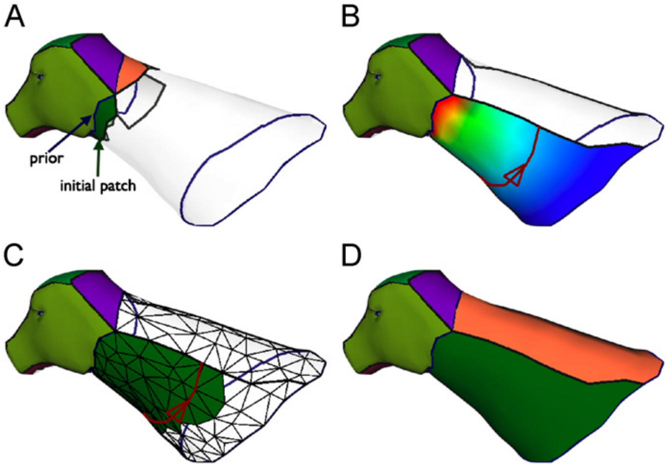

Fig. 7. *Attachment step.* We incorporate priors in the S-VSA method by initializing a segment with one patch per prior (A) and by computing a harmonic function for each prior (B). The pseudo-colors in (B) indicate the function values decreasing from red to blue. At every S-VSA iteration, we also perform the *attachment step* by tracing the isocurve corresponding to the function value at the seed triangle (as indicated by the curve and triangle in red) and marking the section of the patch that goes from the prior arc to the isocurve (C). This section is then kept fixed in the next flooding step (D). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

The harmonic function h_i represents the equilibrium of a diffusion process with sources in $\mathcal{A}_i$ and sinks in $\partial\mathcal{R}\backslash\mathcal{A}_i$, and therefore decreases smoothly away from the prior $\mathcal{A}_i$ (Fig. 7B). Numerically, these functions correspond to solutions of sparse linear systems using the well-known *cotangent* coefficients [48].

Based on these harmonic functions, we force the *continuity* property in $\mathcal{R}$ by keeping each prior $\mathcal{A}_i$ assigned to a fixed patch $\mathcal{P}_i$ (Fig. 6C and D). To this end we initialize the VSA method with one patch per prior (Fig. 7A), and introduce a new step within each iteration (Fig. 7B and C). We call this new step the *attachment step* and apply it at every VSA iteration between the seeding and the flooding step. Its purpose is to mark the triangles of $\mathcal{P}_i$ close to $\mathcal{A}_i$ such that they do not change patch in the next flooding step. We mark a triangle t with tag i if $t \in \mathcal{P}_i$ and $h_i(t) \geq \overline{h_i}(\mathbf{s}_i)$, where $h_i(t)$ and $\overline{h_i}(t)$ are the maximum and the average of h_i in the triangle t, respectively, and $\mathbf{s}_i$ is the seed of $\mathcal{P}_i$. Note that this region corresponds to the section of $\mathcal{P}_i$ that goes from the prior arc $\mathcal{A}_i$ to the isocurve $h_i(\mathbf{s}_i)$.

Modified flooding: During the flooding step, we cluster the triangles in $\mathcal{R}$ into new patches through a region growing process. From each seed $\mathbf{s}_i$, we insert its three adjacent triangles in a priority queue with a label i and with a priority equal to the metric distortion between each triangle and P_i. As in VSA, the flooding is performed by repeatedly popping the triangle with the least distortion until the priority queue is empty. For each triangle t with label i popped out from the queue, we check two conditions: (1) t has not been assigned to any patch and (2) the attachment tag of t is nil or equal to i. If both tests are true, then we add t to $\mathcal{P}_i$ and push its unlabeled neighbors to the priority queue with tag i. Otherwise, we simply go to the next triangle in the queue.

Modified metric: The flooding procedure is responsible for effectively placing the normal-based features in $\mathcal{R}$. The location of the features is determined according to the metric distortion. Our use of the $L^{2,1}$ metric tends to place normal-based features in anisotropic regions, thus enforcing the *orientation* and the *regularity* requirements (Fig. 6C). However, to also induce *orthogonality*, we modulate the $L^{2,1}$ metric by the harmonic functions to obtain a new metric:

$$L_S^{2,1}(t,P_i) = (1-h_i(t))\|\mathbf{N}_t - \mathbf{N}_{P_i}\|^2 A_t, \quad (2)$$

where $\mathbf{N}_{P_i}$ is the normal of P_i (best-fitting plane of $\mathcal{P}_i$), $\mathbf{N}_t$ is the normal of triangle t, and A_t is the area of t. $L_S^{2,1}$ balances shape anisotropy and proximity to priors, and consequently, will bias the placement of features to be along curvature lines that cut adjacent priors *perpendicularly* (Fig. 6D).

Further considerations: The method we described so far creates a number of patches in $\mathcal{R}$ equal to the number of priors. In order to make our method more flexible, we provide two additional operations to control the number of patches. As done in [11] to increase the number of proxies, we insert new patches in $\mathcal{R}$ incrementally using a farthest-point sampling heuristic. For each insertion, we first find the current patch with maximum $L^{2,1}$ distortion. Then we pick the triangle *not adjacent to the prior arcs* with the worst error and use it as an initial seed for the next flooding. This will add a new patch in the most needed part of $\mathcal{R}$. On the other hand, if we wish to reduce the number of normal-based features, we perform a simple merging procedure. For each pair of adjacent patches in $\mathcal{R}$, we simulate a merging of the two regions and compute the resulting $L^{2,1}$ average error. We then greedily merge the pair with the smallest error. Note that the merging procedure does not affect the location of the remaining features, and thus preserves the compactness of our shape abstraction.

We also investigated mechanisms to better deal with near-planar regions: in such cases, the metric $L_S^{2,1}$ may not be sufficient to reveal visually relevant features. We solve this potential issue by *adding* the harmonic functions h_i as a balancing term to the flooding distortion of Eq. (2):

$$L_S^{2,1}(t,P_i)+\alpha(1-h_i(t)). \tag{3}$$

where α is a user-defined constant. If the value of α is large, we generate patches equidistant to the priors resembling a Voronoi-like decomposition. We used Eq. (3) with $\alpha=10$ in Fig. 4B to emphasize the symmetry axes of the torso model. For the other examples, we kept α equal to zero.

We summarize our S-VSA method in Algorithm 1. The underlined parts indicate our modifications to account for segmented regions; note that our approach reduces to the original VSA method if a region has no prior.

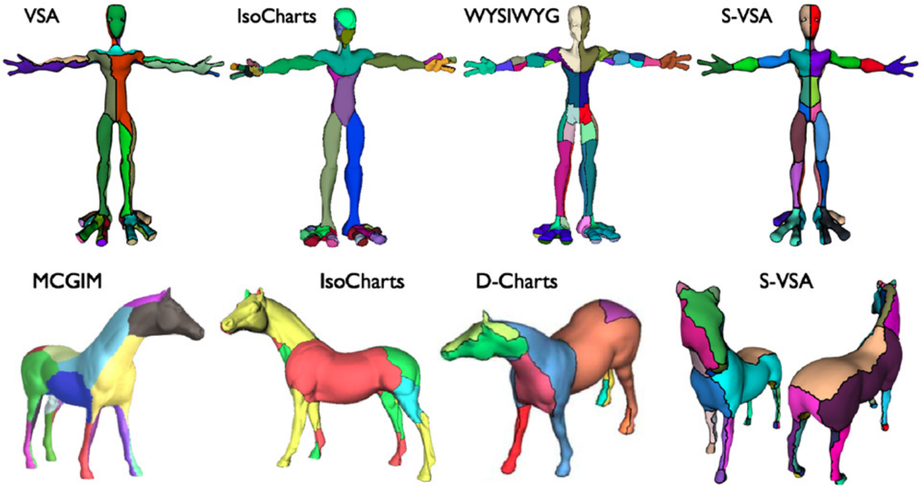

Fig. 8. *Comparing chartification*. Our S-VSA creates charts particularly intuitive and geometrically relevant. First row: VSA [11], IsoCharts [33], WYSIWYG [37], and S-VSA; second row: MCGIM [32], IsoCharts [33], D-Charts [34], and S-VSA.

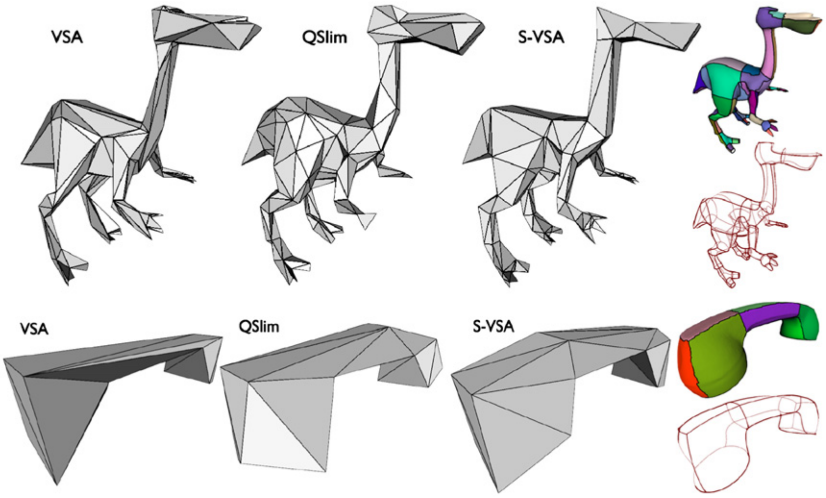

Fig. 9. *Comparing simplifications*. Exoskeletons may be used to create very coarse meshes; from left to right: VSA [11], QSlim [23], and our S-VSA (same # of triangles). While our results are more stylized, the dinopet simplification still results in a symmetric Hausdorff distance of 0.7% of the bounding box, while the phone is at 1.2% (compared to 0.3% and 0.7% for QSlim, respectively).

Algorithm 1. Pseudocode of our S-VSA method.

```
Input: Mesh region $\mathcal{R}$ and priors $\mathcal{A}$
Output: patches $\{\mathcal{P}_i\}$
1   foreach $\mathcal{A}_i \in \mathcal{A}$ do
2       compute harmonic function $h_i$ create patch $\mathcal{P}_i$
3
4   repeat
5       seeding step
6       if $\mathcal{A} \neq \emptyset$ attachment step flooding stepfitting step
7
8
9   until convergence
10  if $\mathcal{R}$ is not segmented enough then
11      insert patch
12      goto line 4
13  while $\mathcal{R}$ is over-segmented do
14      merge a pair of patches
```

5. Discussion

In this section, we describe our experiments on the extraction of exoskeletons and discuss some of our implementation decisions and their respective limitations.

5.1. Implementation details

Our implementation follows the algorithm described in Section 4: we first decompose the shape into perceptual parts and then extract features using the S-VSA algorithm. For the select-and-propagate scheme, we implemented a GUI that allows the user to pick a few perceptual parts to be processed first before propagating the resulting priors to the neighboring regions using a breadth-first traversal. For all the examples shown in this paper, we used 3–6 clicks (typically, at the extremities of the shape) to select the roots of the breadth-first traversal. For the S-VSA algorithm, we allow the user to prescribe a desirable number of patches in each mouse-selected part and we propagate these numbers for the remaining segments of the shape. If necessary, the user can also update the number of patches in any particular part using insertion and merging operations as discussed in Section 4.3. These simple implementation decisions proved sufficient, and our typical session took less than 2 min on a 2.4 GHz machine with 2 GB RAM to generate an abstraction for meshes of 30 K vertices. Refer to the accompanying video for an interactive session performed using our system.

Similar to the VSA method [11], our approach cannot guarantee convergence to the global minimum of the metric distortion. However, since our modifications to incorporate priors restrict the space of solutions, we observed in practice a fast convergence to local minima taking in average less than 25 S-VSA iterations. In our experiments, these local minima proved adequate for our abstraction and, therefore, we did not use *region teleportation* [11] in any of our examples.

5.2. Results

To demonstrate the versatility of our approach, we computed exoskeletons for a variety of models. Fig. 11 showcases results for organic and man-made models, with concise abstractions containing 10–100 curves. We display each individual curve of the network by fitting a cubic Bezier curve in the least squares sense, offering a curve representation of the abstraction that removes minor artifacts (e.g., jagged edges) inherited from the input mesh tessellation.

As opposed to Mehra et al. [17] that only handle man-made shapes, our framework is flexible and adaptable to different applications and classes of objects as we do not restrict our method to any particular segmentation criteria. To further demonstrate this point, we experimented with several segmentation techniques, including the segmentation benchmark available in [46] and recent results obtained in [12,49]. In Fig. 10, we show side by side exoskeletons extracted from two sets of segmentations. For the

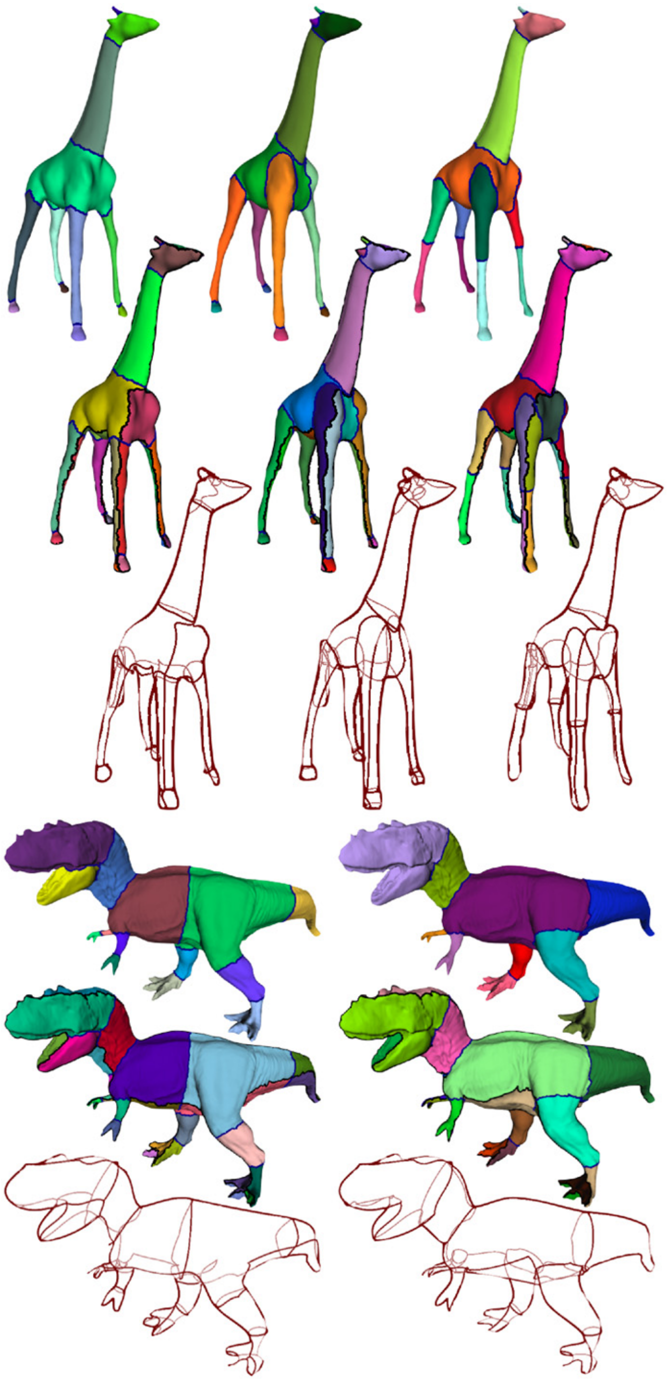

Fig. 10. *Comparing segmentations*. From different part-based segmentations, our method produces consistent results revealing the global structure of the 3D shapes. For the *giraffe* model, we used three different human-generated segmentations available in [46]. For the *tyrannosaur* model, we used a segmentation obtained in [12] and one manually designed.

giraffe model, we computed exoskeletons using three distinct human-generated segmentations available in [46]. For the *tyrannosaur* model, on the other hand, we used the segmentation obtained in [12] and a manual segmentation. While we notice slightly different exoskeletons (since they come from different perceptual features), in both cases we obtain a very consistent and visually pleasing global structure.

5.3. Applications

Our exoskeletons may improve a wide range of geometry processing algorithms. Applications include editing [6,7], modeling [4,5], quad remeshing [40,41] and high-order surface fitting [50,51]. We now discuss some of the benefits of our exoskeletons, in particular to traditional partitioning and simplification schemes.

Texture atlas: In Fig. 8, we compare the patches generated by the S-VSA algorithm to existing chartification techniques. While state-of-the-art approaches aim to create patches with low distortion parameterization, our method finds the relevant features and highlights the structural aspects of the shape. Consequently, our results present a more natural set of patches that resembles how artists unwrap 3D shapes. While we are not directly targeting low parameterization distortion, our results are on par with current methods: using the parameterization scheme presented in [52], we found our quasi-conformal errors to be typically within 1.5% of recent published results for a comparable number of patches.

Simplification: We also used our exoskeleton to extract a very coarse version of the input mesh. To this end, we triangulated our patches using the constrained Delaunay triangulation proposed in [11]. Fig. 9 compares our simplified meshes to meshes obtained by QSlim [23] and VSA [11]. Our approach creates a rather stylized simplification, doing a much better job at highlighting the symmetries of the shape. Additionally, because our method involves perceptual regions, we can always preserve structural parts of the mesh even under extreme simplification. Note, for example in the *dinopet* model, that the QSlim method eventually reduces the hand to a simple triangle, while S-VSA naturally maintains its structure. Nonetheless, QSlim outperforms our approach in terms of computational time and symmetric Hausdorff approximation error.

5.4. Limitations

The main limitation of our framework to abstract 3D shapes lies in its inability to *automatically* deal with the combinatorial number of orders in which perceptual parts can be organized by the select-and-propagate scheme. Although we provided a few basic automatic orders that turned out to be appropriate for a large range of shapes, it remains unclear how to pick the best order for a specific application—in fact, it is not even clear that there *is* a notion of best ordering, as it is not only application-dependent, but most likely user-dependent. As a future work, we plan to investigate how the perceptual parts influence the relevance of feature curves. Note that preliminary results have already been presented for 2D shapes [10]. In this work, however, we took the pragmatic standpoint that a one-size-fits-all approach can be generated if needed, but that a *minimalist* interactive user input is the best way to go in practice.

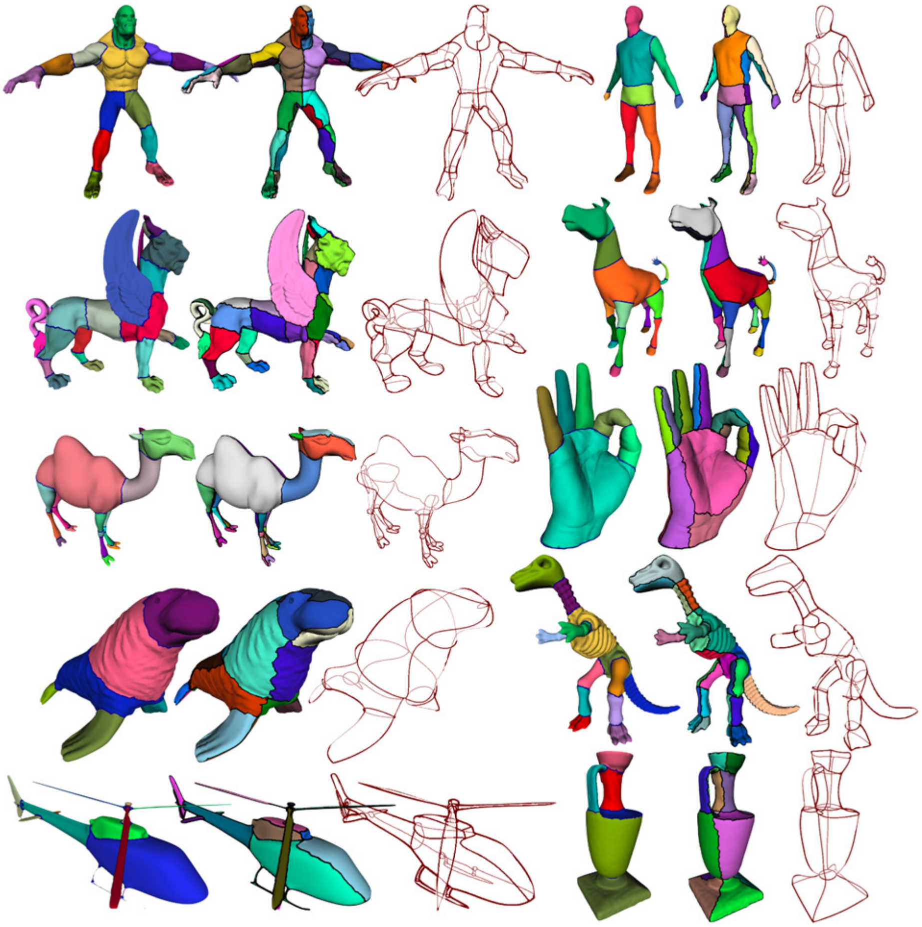

Fig. 11. *Exoskeleton gallery*. Resulting abstraction for various 3D shapes. Left to right: perceptual parts obtained through segmentation, patches extracted by S-VSA, and final network of feature curves.

6. Conclusion

We presented a principled and versatile algorithm for the creation of 3D shape abstractions. The resulting representation uses a network of perceptual and normal-based curves that capture the defining characteristics of the input geometry, and provides a structure aware abstraction of the input surface. Our method combines shape approximation and part-based segmentation to get the best of both approaches, and can be directly used for editing, charting, texturing, and simplification of 3D models.

For future work, we intend to combine our exoskeletons with editing systems such as FiberMesh [6] and iWires [7]: we believe that the balance of coarseness and global structure of our abstraction can provide significant benefits towards intuitive handling of 3D shapes, especially for organic models. In the same direction, we plan to revisit methods like [53] where fair surfaces are designed using curve networks as geometric constraints. Investigating the similarities between our exoskeletons and the way shapes are described in modeling systems such as ILoveSketch [4] and EverybodyLovesSketch [5] is also of interest. Another potential avenue is to relate the principles behind our exoskeleton construction to the Gestalt theory of perception [54]. Furthermore, the distinction between normal-based and perceptual features in our work could be exploited to define new analytic tools to evaluate shape properties such as parallelism, symmetry, and hierarchy of perceptual parts.

Acknowledgments

We would like to thank Alan H. Barr and Lance Williams for their helpful comments, and Patrick Mullen for proof-reading the paper. Meshes are courtesy of the AIM@Shape Project, Autodesk, and the Princeton Benchmark. This research was partially funded by NSF grants (CCF-0811373, CMMI-0757106, and CCF-1011944), CNPq (Proc. 309254/2007-8, 200717/2010-3), and FAPESP (Proc. 2007/52015-0, 2009/18438-7).

Appendix A. Supplementary data

Supplementary data associated with this article can be found in the online version at doi:10.1016/j.cag.2010.11.012.

References

[1] Mohr A, Gleicher M. Building efficient, accurate character skins from examples. In: SIGGRAPH. ACM; 2003. p. 562–8.

[2] Au OK-C, Tai C-L, Chu H-K, Cohen-Or D, Lee T-Y. Skeleton extraction by mesh contraction. In: SIGGRAPH. ACM; 2008. p. 1–10.

[3] Cornea ND, Demirci MF, Silver D, Shokoufandeh A, Dickinson SJ, Kantor PB. 3d object retrieval using many-to-many matching of curve skeletons. In: Proceedings of the international conference on shape modeling and applications. IEEE Computer Society; 2005. p. 368–73.

[4] Bae S-H, Balakrishnan R, Singh K. ILoveSketch: as-natural-as-possible sketching system for creating 3D curve models. In: Symposium on user interface software and technology. ACM; 2008. p. 151–60.

[5] Bae S-H, Balakrishnan R, Singh K. EverybodyLovesSketch: 3D sketching for a broader audience. In: Symposium on user interface software and technology. ACM; 2009. p. 59–68.

[6] Nealen A, Igarashi T, Sorkine O, Alexa M. Fibermesh: designing freeform surfaces with 3D curves. ACM Trans Graph (SIGGRAPH) 2007b;26(3). Art. 41.

[7] Gal R, Sorkine O, Mitra N, Cohen-Or D. iWIRES: an analyze-and-edit approach to shape manipulation. ACM Trans Graph (SIGGRAPH) 2009b;28. Art. 33.

[8] Cole F, Golovinskiy A, Limpaecher A, Barros HS, Finkelstein A, Funkhouser T, Rusinkiewicz S. Where do people draw lines? ACM Trans Graph (SIGGRAPH) 2008b;27(3). Art. 88.

[9] Cole F, Sanik K, DeCarlo D, Finkelstein A, Funkhouser T, Rusinkiewicz S, Singh M. How well do line drawings depict shape? ACM Trans Graph (SIGGRAPH) 2009b;28(3). Art. 28.

[10] Mi X, DeCarlo D, Stone M. Abstraction of 2d shapes in terms of parts. In: Symposium on non-photorealistic animation and rendering. p. 15–24.

[11] Cohen-Steiner D, Alliez P, Desbrun M. Variational shape approximation, ACM Tran. Graph (SIGGRAPH) 2004, p. 905–14.

[12] de Goes F, Goldenstein S, Velho L. A hierarchical segmentation of articulated bodies. Comput Graph Forum (Symposium on Geometry Processing) 2008b;27:1349–56.

[13] Hildebrandt K, Polthier K, Wardetzky M. Smooth feature lines on surface meshes. In: Symposium on geometry processing. p. 85–90.

[14] Hoffman D, Richards W. Parts of recognition. Cognition 1983b;18:65–96.

[15] Biederman I. Recognition-by-components: a theory of image understanding. Psychological Review 1987b;94:115–47.

[16] Singh M, Hoffman DD. Part-based representations of visual shape and implications for visual cognition. In: From fragments to objects: segmentation and grouping in vision, advances in psychology, vol. 130. p. 401–59.

[17] Mehra R, Zhou Q, Long J, Sheffer A, Gooch A, Mitra NJ. Abstraction of man-made shapes. ACM Trans Graph (SIGGRAPH Asia) 2009b;28(5). Art. 137.

[18] Hertzmann A, Zorin D. Illustrating smooth surfaces, in: ACM SIGGRAPH Proceedings, p. 517–26.

[19] DeCarlo D, Finkelstein A, Rusinkiewicz S, Santella A. Suggestive contours for conveying shape. ACM Trans Graph (SIGGRAPH) 2003b;22(3):848–55.

[20] DeCarlo D, Rusinkiewicz S. Highlight lines for conveying shape. In: Symposium on non-photorealistic animation and rendering. p. 63–70.

[21] Yoshizawa S, Belyaev A, Seidel H-P. Fast and robust detection of crest lines on meshes. In: Symposium on solid and physical modeling. p. 227–32.

[22] Weinkauf T, Günther D. Separatrix persistence: extraction of salient edges on surfaces using topological methods. Comput Graph Forum (Symposium on Geometry Processing) 2009b;28:1519–28.

[23] Garland M. Heckbert PS. Surface simplification using quadric error metrics. In: ACM SIGGRAPH proceedings. p. 209–16.

[24] Garland M, Willmott A, Heckbert PS. Hierarchical face clustering on polygonal surfaces. In: Symposium on interactive 3D graphics. p. 49–58.

[25] Marinov M, Kobbelt L. Automatic generation of structure preserving multiresolution models. Comput Graph Forum 2005b;24(3):479–86.

[26] Attene M, Falcidieno B, Spagnuolo M. Hierarchical mesh segmentation based on fitting primitives. Vis Comput 2006b;22(3):181–93.

[27] Gelfand N, Guibas LJ. Shape segmentation using local slippage analysis. In: Symposium on geometry processing. ACM; 2004. p. 214–23.

[28] Wu J, Kobbelt L. Structure recovery via hybrid variational surface approximation. Comput Graph Forum 2005b;24(3):277–84.

[29] Yan D-M, Liu Y, Wang W. Quadric surface extraction by variational shape approximation. In: Proceedings of geometric modeling and processing. p. 73–86.

[30] Lee CH, Varshney A, Jacobs DW. Mesh saliency. In: SIGGRAPH. ACM; 2005. p. 659–66.

[31] Lévy B, Petitjean S, Ray N, Maillot J. Least squares conformal maps for automatic texture atlas generation. ACM Trans Graph (SIGGRAPH) 2002b;21(3):362–71.

[32] Sander PV, Wood ZJ, Gortler SJ, Snyder J, Hoppe H. Multi-chart geometry images. In: Symposium on geometry processing. p. 146–55.

[33] Zhou K, Synder J, Guo B, Shum H-Y, Iso-charts: stretch-driven mesh parameterization using spectral analysis. In: Symposium on geometry processing. p. 45–54.

[34] Julius D, Kraevoy V, Sheffer A. D-charts: quasi-developable mesh segmentation. Comput Graph Forum 2005b;24(3):581–90.

[35] Autodesk Maya, Modeling and animation software package, 〈http://www.autodesk.com〉, 2010.

[36] Autodesk 3ds, Modeling, animation, and rendering package, 〈http://www.autodesk.com〉, Max.

[37] Vallet B, Lévy B. What you seam is what you get: automatic and interactive UV unwrapping, Technical Report, INRIA, 2009.

[38] Boier-Martin I, Rushmeier H, Jin J. Parameterization of triangle meshes over quadrilateral domains. In: Symposium on geometry processing. p. 193–203.

[39] Dong S, Kircher S, Garland M. Harmonic functions for quadrilateral remeshing of arbitrary manifolds. Comput Aided Geom Des 2005b;22(5):392–423.

[40] Huang J, Zhang M, Ma J, Liu X, Kobbelt L, Bao H. Spectral quadrangulation with orientation and alignment control. ACM Trans Graph (SIGGRAPH Asia) 2008b;27(5):1–9.

[41] Bommes D, Zimmer H, Kobbelt L. Mixed-integer quadrangulation. ACM Trans Graph 2009b;28(3):1–10.

[42] Cornea ND, Silver D, Min P. Curve-skeleton properties, applications, and algorithms. IEEE Trans Visualization Comput Graph 2007b;13:530–48.

[43] Pascucci V, Scorzelli G, Bremer P-T, Mascarenhas A. Robust on-line computation of Reeb graphs: simplicity and speed. ACM Trans Graph 2007b; 26:58.

[44] Shapira L, Shamir A, Cohen-Or D. Consistent mesh partitioning and skeletonisation using the shape diameter function. Vis Comput 2008b;24:249–59.

[45] Shamir A. A survey on mesh segmentation techniques. Comput Graph Forum 2008b;27:1539–56.

[46] Chen X, Golovinskiy A, Funkhouser T. A benchmark for 3D mesh segmentation. ACM Trans Graph (SIGGRAPH) 2009b;28(3). Art. 73.

[47] Huang Q, Wicke M, Adams B, Guibas L. Shape decomposition using modal analysis. Comput Graph Forum 2009b;28.

[48] Desbrun M, Meyer M, Schröder P, Barr AH. Implicit fairing of irregular meshes using diffusion and curvature flow. In: ACM SIGGRAPH proceedings. p. 317–24.

[49] Zheng Y, Tai C-L. Mesh decomposition with cross-boundary brushes. Comput Graph Forum (Proc. of Eurographics 2010) 2010b;29:527–35.

[50] Li W-C, Ray N, Lévy B. Automatic and interactive mesh to t-spline conversion. In: Proceedings of the fourth Eurographics symposium on geometry processing. Eurographics Association, 2006. p. 191–200.

[51] Myles A, Pietroni N, Kovacs D, Zorin D. Feature-aligned t-meshes. ACM Trans Graph 2010b;29:1–11.

[52] Mullen P, Tong Y, Alliez P, Desbrun M. Spectral conformal parameterization. In: Proceedings of the symposium on geometry processing. p. 1487–94.

[53] Moreton HP, Séquin CH. Functional optimization for fair surface design. In: SIGGRAPH: Proceedings of the 19th annual conference on computer graphics and interactive techniques. ACM; 1992. p. 167–76.

[54] Arnheim R. Art and Visual Perception: a psychology of the creative eye. London: Faber and Faber; 1956.