Computers & Graphics 35 (2011) 500–509

Contents lists available at ScienceDirect

Computers & Graphics

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

SMI 2011: Full Paper

Point set silhouettes via local reconstruction

Matt Olson [a,*], Ramsay Dyer [b], Hao Zhang [a], Alla Sheffer [c]

[a] Simon Fraser University, Canada
[b] INRIA Sophia Antipolis, France
[c] University of British, Vancouver, British Columbia, Canada

ARTICLE INFO

Article history:
Received 11 December 2010
Received in revised form
14 March 2011
Accepted 14 March 2011
Available online 24 March 2011

Keywords:
Point clouds
Geometry processing
Local reconstruction
Silhouette extraction
Boundary detection

ABSTRACT

We present an algorithm to compute the silhouette set of a point cloud. Previous methods extract point set silhouettes by thresholding point normals, which can lead to simultaneous over- and under-detection of silhouettes. We argue that additional information such as surface curvature is necessary to resolve these issues. To this end, we develop a local reconstruction scheme using Gabriel and intrinsic Delaunay criteria and define point set silhouettes based on the notion of a *silhouette-generating set*. The mesh *umbrellas*, or local reconstructions of one-ring triangles surrounding each point sample, generated by our method enable accurate silhouette identification near sharp features and close-by surface sheets, and provide the information necessary to detect other characteristic curves such as creases and boundaries. We show that these curves collectively provide a sparse and intuitive visualisation of point-cloud data.

1. Introduction

Point clouds acquired using laser scanners account for many of the digital 3D models in common use. However, despite much research, converting a point cloud into a quality mesh remains a difficult and costly process. This difficulty is one of the main motivations for developing geometry processing techniques which operate directly on points [20]. In this setting, even basic tasks such as rendering can be challenging, since points do not *a priori* contain normal or orientation information, and even determining which points are visible is non-trivial [26,32]. Even with visibility resolved, displaying points without normal information can hide important surface features; see Fig. 1 (left column). Methods that generate high-quality point-cloud renderings, such as splatting [40], rely on elaborate filtering especially near characteristic curves such as silhouettes.

Perception research demonstrates that rendering the characteristic curves of input models, and specifically their silhouettes, provides an effective visualisation enhancement or even alternative to rendering the entire model [28]. Silhouettes are important visual cues for shape perception [28] and are very effective at conveying shapes [17,21]. As such accurate rendering of point-cloud silhouettes can enhance visualisation of the cloud data in a concise yet effective way. In this paper, we provide a method for accurately and efficiently computing the silhouette set of a point cloud, *i.e.* the subset of points which best approximate the silhouette of the underlying surface. Such a set can be used to quickly and intuitively depict a point-based model (Fig. 1) or adapted to other tasks that use silhouettes such as shadow volumes [9] and object tracking [35].

According to the standard definition of silhouettes for a smooth surface, a point p is on the silhouette if its normal is perpendicular to the view vector at p. On a point cloud where only disconnected points are available, Zakaria and Seidel [39] proposed using *normal thresholding*, defining a point p to be on the silhouette if the scalar product between the point normal at p and the view vector is sufficiently close to zero. However, as shown in Fig. 2, this method can simultaneously under- and over-detect silhouettes, introducing thick point patches instead of narrow silhouettes in low-curvature regions and leaving gaps in high-curvature regions, especially near sharp features. Both errors reduce the utility of the resulting silhouette sets. In general, it is not possible to choose a suitable fixed threshold to simultaneously correct both types of errors.

Our point set silhouette construction algorithm is driven an alternative characterisation of geometric silhouettes that allows a unified treatment of silhouettes for different forms of surface models. In general, we associate with a point p on a surface model M a *silhouette-generating set* or *SGS* such that p is on the silhouette if the viewpoint is contained in its SGS. Given appropriate selections for the SGS of a point or edge, this definition is equivalent to the traditional definition for smooth surfaces and polygon meshes (Fig. 3). For example, on a smooth surface S, we identify the tangent plane at p as its SGS. For a mesh surface, the set difference between the union and the intersection of all the half-spaces defined by the normals of a vertex's *umbrella* (the triangles adjacent to the vertex) provides a natural SGS, though this is typically defined only on edges as a double wedge [2].

* Corresponding author.
E-mail address: matt.j.olson@gmail.com (M. Olson).

0097-8493/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cag.2011.03.034

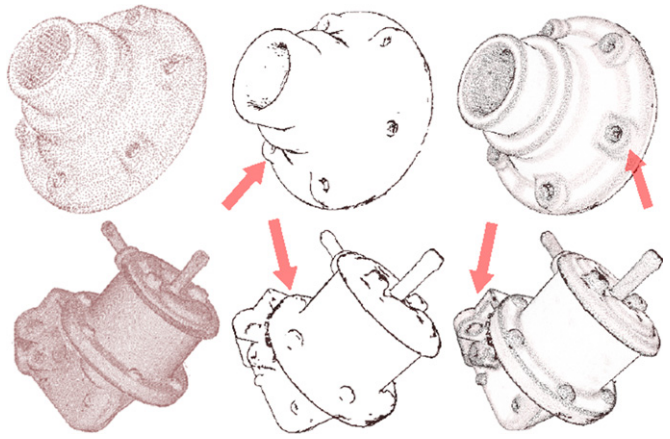

Fig. 1. Surface features in raw point clouds are difficult to visualize, even with visibility resolved (left). By rendering the point set silhouettes (middle), and especially the detected sharp features (right), geometric details of the underlying shapes are better revealed.

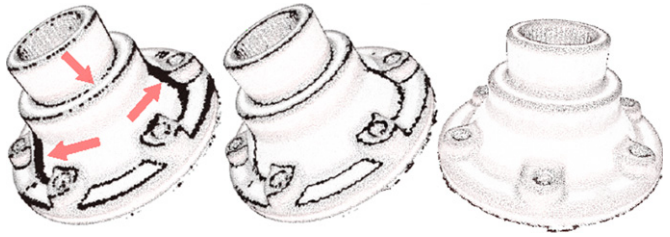

Fig. 2. Normal thresholding (left) can over- and under-detect point set silhouettes. Results using our method (middle) based on SGS and local reconstruction show visible improvement on silhouette accuracy. An important note is that the camera view which produced (a) and (b) were chosen to best reveal the silhouette set returned. This view is different from the view (right) that generated the silhouettes.

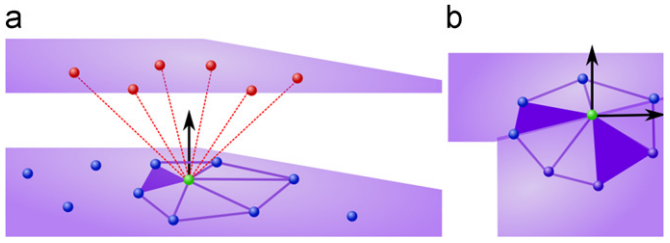

Fig. 3. To generate the SGS of a point p (green) based on its k nearest neighbours, we (a) find a Gabriel triangle on p, discard neighbours distant from the triangle's plane (red), and build a Delaunay triangulation from the remainder (blue). If p lies on a sharp feature edge (b), we build separate umbrellas for each smooth patch on that edge. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Inspired by this observation, we seek to approximate the set of planes tangent to a point p's intrinsic Voronoi cell on S (see Fig. 4), which we argue describes the SGS of p. To compute the SGS of a point p in a point cloud P, we search for an optimal umbrella around it, performing a *local* reconstruction of the underlying surface. As shown in Figs. 2 and 12, our construction leads to significantly more accurate silhouette extraction than normal thresholding. The local reconstruction also allows us to efficiently extract characteristic model features such as sharp edges to facilitate visualisation of point clouds; see Fig. 1.

Our local reconstruction algorithm is based on the assumption that the underlying surface S is a *piecewise-smooth* manifold, which is smooth everywhere except at *feature curves* such as sharp edges or boundaries. For a non-feature sample point p we obtain an initial

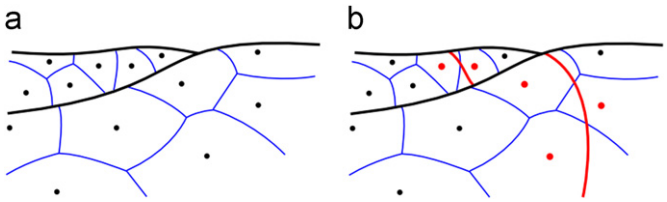

Fig. 4. Finding point samples on a surface's silhouette. (a) Point samples on an underlying smooth surface S and their intrinsic Voronoi cells. (b) A silhouette curve on S. The points whose Voronoi cells are crossed by the curve (highlighted) are on the silhouette.

estimate of the unoriented surface normal using a simple and novel technique, and show that the error in this estimate is bounded by the sampling radius. We then construct an umbrella by performing Delaunay edge flips [14] on k nearest neighbours. Our algorithm is also able to identify feature curves in a small local neighbourhood. If p is at or near a feature, we construct a partial umbrella for each smooth patch involved. We focus on the construction of these umbrellas even in the presence of sharp features, and do not attempt to robustly estimate oriented surface normals.

In particular, we exploit the geometric insight we develop in Section 4 to identify sharp edges between smooth surface patches, as well as boundaries on the underlying sampled surface, *e.g.* in an incomplete scan of a real-world object. Many global surface-reconstruction algorithms in common use have difficulty with open manifolds or cannot reconstruct these features with high fidelity; we hope to provide a simple way to augment these methods. Our local measure performs well on samples taken from open manifolds and the produced results are shown to be comparable to those from more sophisticated methods such as Peel [12].

Our umbrella construction does not require an oriented point cloud or extra sampling at feature curves and boundaries, though we do make lenient assumptions about the distribution of samples. We produce satisfactory results using a pair of independent parameters to indicate the density and uniformity of the sampling, using the same default values for both for all the examples in this paper. Our method is able to produce plausible local reconstructions, and thus accurate silhouettes, on inputs containing sharp features and close-by surface sheets, without resorting to expensive statistical techniques. Its formulation is based on a small number of nearest neighbours, making it well-suited to efficient implementation on architectures such as GPUs in the same manner as [27], heavily parallelized computers, out-of-core applications, and asymmetric processors, where the random global memory access required by many global reconstruction methods is difficult, costly, or impractical.

One theoretical limitation of our method is sensitivity to noise; however, we have successfully computed silhouettes on numerous noisy examples after denoising them using standard methods such as WLOP [23].

2. Related work

Silhouette extraction: Silhouette extraction for meshes is a well studied problem [25]. Image-based silhouette extraction methods, such as [10], are generally quite fast at producing a set of silhouette pixels in the projection plane. Object-space silhouettes [21,33] provide additional surface information which can be used for stylised rendering and can be naturally combined with extraction and rendering of other features. In particular, by extracting the full silhouette rather than only its visible component, object-space algorithms facilitate applications such as shadow rendering [9] and collision detection [7]. When operating in object space,