Continuous Visibility Feature

Guilin Liu
gliu2@gmu.edu

Yotam Gingold
ygingold@gmu.edu

Jyh-Ming Lien*
jmlien@cs.gmu.edu

Abstract

In this work, we propose a new type of visibility measurement named Continuous Visibility Feature (CVF). We say that a point q on the mesh is continuously visible from another point p if there exists a geodesic path connecting p and q that is entirely visible by p. In order to efficiently estimate the continuous visibility for all the vertices in a model, we propose two approaches that use specific CVF properties to avoid exhaustive visibility tests. CVF is then measured as the area of the continuously visible region. With this stronger visibility measure, we show that CVF better encodes the surface and part information of mesh than the tradition line-of-sight based visibility. For example, we show that existing segmentation algorithms can generate better segmentation results using CVF and its variants than using other visibility-based shape descriptors, such as shape diameter function. Similar to visibility and other mesh surface features, continuous visibility would have many applications.

1. Introduction

Feature extraction often serves as a fundamental engineering task for higher level applications. For example, almost all of the recognition tasks start with defining or learning features. This is particularly true for two-dimensional image that has a simple and uniform grid structure and pixels can be indexed by a 2D vector in a continuous coordinate system. Each pixel has fixed number (usually 4 or 8) of neighbors. Thus most of the image features are defined by the convolution operations of the local areas. Consequently, there are some default features like SIFT [15] and HOG [5] used widely in computer vision research.

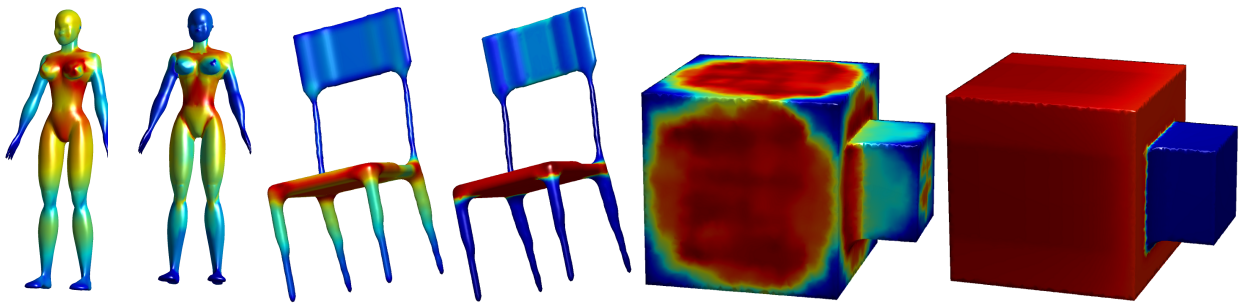
Unlike images, 3D models are usually modeled in the continuous domain which makes defining 3D features challenging. Consequently, many of the features defined for 3D models usually are designed for specific types of shapes.

For example, visibility among points inside a given shape has been used directly or indirectly as shape features, in particular for the task of semantic shape segmentation. The intuition behind most of these visibility-based features is from the observation that two points sampled from the same semantic part tend to be visible from each other. For example, shape diameter function (SDF) [18] is a descriptor to describe the thickness of a mesh defined via local visibility. The idea is initially proposed for shape segmentation based on the assumption that a visually meaningful component should have similar thickness everywhere. However, this assumption does not hold in many cases. For example, for a long component, such as a leg, the thickness at one end of the component may have different thickness from the other end. Another example is a flat component or a component whose size grows gradually in a certain direction.

We believe that the traditional line-of-sight visibility used in all existing visibility-based shape features is insufficient. In this paper, we will describe a new feature named *continuous visibility feature*. The feature provides stronger visibility measure by considering the continuously visible region for a vertex or facet. Thus this feature is defined in a per-vertex manner. We say that a q is continuously visible by a point p if there exists a geodesic path connecting p and q that is entirely visible by p. CVF of p is defined as the area of a set of continuously visible points by p. A more precise definition of CVF can be found in Sections 3. We show that CVF better encodes the surface and part information of mesh than SDF does. Figure 1(a) illustrates some example models color mapped by CVF and SDF values. We also show that existing segmentation algorithms can generate better segmentation results using CVF than using other shape descriptors, such as shape diameter function [18]. We will demonstrate the segmentation results using the Princeton Segmentation Benchmark and applications of CVF and its variants (namely CVF_{avg}, strong CVF and weak CVF) beyond segmentation in Section 5.

A major technical challenge of computing CVF is the computational efficiency because it is known the determining the visibility of two points is expensive and determining their continuous visibility will require many more pairwise visibility checks. In Section 4, we will present two ways to compute CVF efficiently. The first approach will use the continuous property of CVF to construct the continuously

*All authors are with the Department of Computer Science, George Mason University, Fairfax, VA, USA, 22030



(a) SDF (b) CVF (c) SDF (d) CVF (e) SDF (f) CVF

Figure 1. Examples of SDF and CVF on the same models. Red and blue indicate the largest and smallest SDF and CVF values, respectively. From these examples, we can see that CVF provides a better metric to distinguish visually different parts of a model. More extensive comparison can be found in Section 5.

visible region in a Breadth-First-Search manner. The second approach further improves the efficiency by first collecting the potential continuous visible region and then using visibility test to search the true boundaries of the continuous visible region. We find that the second approach provides significant speedup over the first approach.

2. Related Work

Extracting shape features from 3D model is a fundamental operation in shape segmentation, matching, retrieval and even deformation. These features can be roughly classified into per-vertex/facet features and global signatures. However, we should point out that some global signatures are also generated from per-vertex/facet features via statistical approaches. Osada et al. [16] introduced Shape Distributions to capture the statistical information of vertices' and facets' shape function values. Shape matching is achieved by measuring the similarity between two shapes' distribution. Even though the feature function used in [16] is simple, it paves the way for many research works in shape matching, shape retrieval and other shape analysis tasks.

Features can be further classified into several categories depending on whether they are invariant to translation, scaling, and deformation. Some features are designed to be invariant to rigid transformation, e.g., spin image [9] and shape context [2]. Invariance to deformation has attracted more attention. For example, geodesic distance is used to build deformation invariance features in a multi-dimensional scaling based approach [6] and spectral domain analysis [8]. The idea of heat diffusion process has triggered the emergence of diffusion geometry [4]. These features are usually using Laplaca-Beltrami operator, e.g., heat kernel signature [20], global point signature [17] and multi-solution spectral descriptor [12].

Most of the aforementioned features are designed for shape matching, retrieval and correspondence. In shape segmentation, researchers have developed other types of features. These features directly control the quality of the final segmentation results for the approaches based on k-means clustering [19], fuzzy clustering [11], Gaussian-mixture model followed by graph cut [18]. Some other recent work on segmentation which acquires better result also needs some feature definition [1, 21]. Liu et al. [13] used concavity as features. Kalogerakis et al. [10] and Huang [7] proposed the learning-based segmentation using a rich collection of the features.

Visibility-based Features. Several recent works use visibility to derive part-aware features. The intuition behind most of these visibility-based features is that two points in the a visually or functionally meaningful part (such as the leg of a table) tend to be visible from each other. For example, Shape Diameter Function (SDF) [18] captures the thickness of a shape locally among visible points. SDF is determined by sampling rays inside the cone in the anti-normal direction of a facet. The SDF value is the sum of the projected length of the rays inside the model. However, SDF may not correspond well to a visually meaningful component if the thickness is not evenly distributed. For example, as shown in Figure 1(c), the tabletop has quite different feature values at its center from those on the boundary. In addition, the feature values at the ends of the leg are also quite different from those closer to the tabletop.

Another visibility-based feature is Volumetric Shape Image (VSI) [14]. With the motivation of capturing the general volumetric context instead of only getting the local volumetric context of the local cone used for sampling rays, VSI tries to sample rays in more directions. The VSI feature is computed by first finding the proxy center for each vertex and then sampling ray at fixed direction. The field of VSI is achieved by comparing the difference between the sampling of a source vertex and other vertices on the mesh.

Recently, weak convex decomposition proposed by [1] also uses lines-of-sights. It first computes the pairwise visibility between all pairs of the vertices/facets on the mesh. Then the similarity matrix is constructed with each entry's

value to be 0 or 1 indicating the pairwise invisibility or visibility, followed by spectral clustering. Even though the mathematical background seems to be quite straightforward by referring to the similar issues in machine learning problem, the decomposition sometimes is not satisfying especially when a mesh model has a large area of bended parts. van Kaick et al. [21] used the technique in [1] as preprocessing step to over-segment the mesh, but it also needs a lot of post-merging.

In all of these features [18, 14, 1, 21], traditional line-of-sight visibility is used. However, from the Figure 2, we could see that the continuous visibility makes more sense than the general visibility in terms of charactering a potential component.

3. Definitions and Properties of CVF

Visibility among the points inside a given shape has been used directly or indirectly as shape features in the past. The intuition behind most of these visibility-based features is from the observation that two points sampled from the same (visual or functional) part tend to be visible from each other. It usually remains true if we state the property the other way around: two points that are visible from each other are likely to be from the same part. However, there are many exceptions. For example, in a human model in a standing pose, a point from the head may see a point in the heel and in most cases we would distinguish head and heel as different parts of the model.

To overcome the drawbacks of the line-of-sight based visibility approaches and to better capture the component information of a mesh, we define a new feature named **continuous visibility feature** (CVF). We say that a point q on the mesh is continuously visible by p if there exists a geodesic path connecting p and q that is entirely visible by p. Note that, this path may not be the shortest geodesic path between p and q. More specifically, we define the function CV(p,q) that returns **TRUE** if and only if

$$\exists \pi \text{ such that } \forall r \in \pi, \overline{pr} \cap M = \emptyset \,,$$

where π is a geodesic path connecting p to q on mesh M. An illustration of the continuous visibility is shown in Figure 2 (a).

Intuitively, by applying this stronger version of visibility measure, two points from the same part not only should see each other but the entire path connecting them should be visible. Under these definitions, we say a continuously visible region (CVR) of a point p is a collection of all points that are continuously visible by p. That is, we can express CVR of p as the set $\{q \in M \mid \text{CV}(p, q) = \textbf{TRUE}\}$ CVF of a given point p can be represented either in vector or in scalar form. In this paper, we will focus on scalar representation, in which CVF is the area of p's CVR. If the input model is convex, then all vertices in this convex model have the same CVF, which is the surface area of the model.

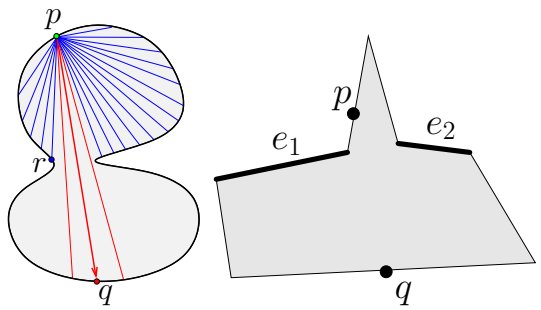


Figure 2. (a) Example of 2D continuous visibility. The vertex r is continuously visible from the vertex p but the vertex q is not. (b) While p is continuously visible from q, q is not continuously visible from p.

Unlike visibility, continuous visibility is not commutative. That is, if CV(p,q) is true, CV(q,p) may not be true. A 2D example is illustrated in Figure 2 (b), in which the point v cannot continuously see the point w because the visibility continuity is interrupted by edges e_1 and e_2. Therefore, when both CV(p,q) and CV(q,p), we say that p and q have *strong continuous visibility*.

4. Computing CVF

In this section, we discuss how continuous visibility can be determined for all vertices of a given mesh. Without loss of generality, we will focus the computation of the continuous visibility of a given vertex v. In Section 4.1, we first introduce a flooding approach using Breadth-First Search (BFS) rooted at v. However, BFS-based approach requires many visibility check between v and all vertices visited by the BFS tree. We propose the second approach to address this efficiency issue in Section 4.2.

4.1. BFS-based Approach

This first approach uses the property that the continuous visible region of the vertex v must be continuous because, by definition, every vertex in the continuous visible region is connected to v via a path that is entirely visible by v. In this Breadth First Search (BFS) based method, we iteratively expand the search only at the vertices that are visible to v and stop at the vertices that are invisible to v. We can prove the correctness of this approach by saying that, it is impossible to find a vertex u that is continuously visible by v but is not discovered by the BFS approach.

Although the BFS-based method is simple to implement and provably correct, it may require many visibility checks. In the next section, we propose an alternative approach to speed up the computation.

4.2. Filtering

We further propose a more efficient way to compute the feature values. The main idea is to filter out the faces that guaranteed to be outside the continuously visible region of a given vertex v. If a triangle f and v form a tetrahedron of positive volume, then, we say that f is a positive triangle. Then a *continuously positive region* of v is a set T_v of positive triangles continuously connected to v. It is clear that v's continuously visible region must be a subset of T_v, and we can construct T_v using the same BFS-based approach discussed in the previous section, except that we start from the incident facets of v.

The next step involves finding the boundaries between the continuously positive region CVR_v and the continuously visible region T_v. This is done iteratively by connecting v to a vertex u that is adjacent to T_v but is not in T_v. This guarantees that $\text{CV}(v, u)$ must be **FALSE**. A geodesic path π connecting v and u now must cross at least one boundary between CVR_v and T_v. Let w be the vertex closest v on π that is invisible to v and w' be the last visible vertex on π before we arrive w. By definition, w' is continuously visible and edge $e = \{w', w\}$ connecting w' and w must be a boundary edge. To determine the entire boundary including e, we first identity the visibility of the third vertex of the triangle t incident to e and then move on to the edge $e' \neq e$ of t whose end points also form a pair of visible and invisible vertices. This process repeats until a closed loop is discovered. Using this loop removes vertices that not continuously visible by v. The algorithm sketching this idea can be found in the supplementary materials.

4.3. Running Time

We use the models from Princeton Segmentation Benchmark to obtain the running time reposted below. Some of the benchmark models are shown in Figure 4. The results are obtained from our implementation in C++ on a machine with Intel Xeon 2.30GHz CPU and 32 GB memory, with 24 models running parallely. Using the BFS-based approach, for models with about $1.5K$ vertices, the number of intersection test is around 0.9 million and the running time is 10 seconds on average. A model with about $5K$ vertices and $10K$ facets, BFS-based approach requires about 8 million intersection tests and the total running time is around 100 seconds. For a model with $10K$ vertices, the total intersection number is around 70 million and the running time is about 900 seconds.

On the other hand, if we use the filter-based approach on the same set of models, the intersection tests for models with less $5K$ vertices reduce by 72% and the running time reduces by 30%. For models with $5K \sim 10K$ vertices, the intersection tests and running time reduce by 82% and 37% respectively. For models with more than $10K$ vertices, the intersection tests and running time reduce by 89% and 49%.

5. Comparison and Applications

In this section, we compare continuous visibility feature (CVF) to shape diameter function (SDF). SDF has been widely used in part-based shape analysis. For example, MeshLab has adopted it as a filter and CGAL has re-implemented SDF for extracting features from shape and then used it for shape segmentation.

5.1. Visual comparisons

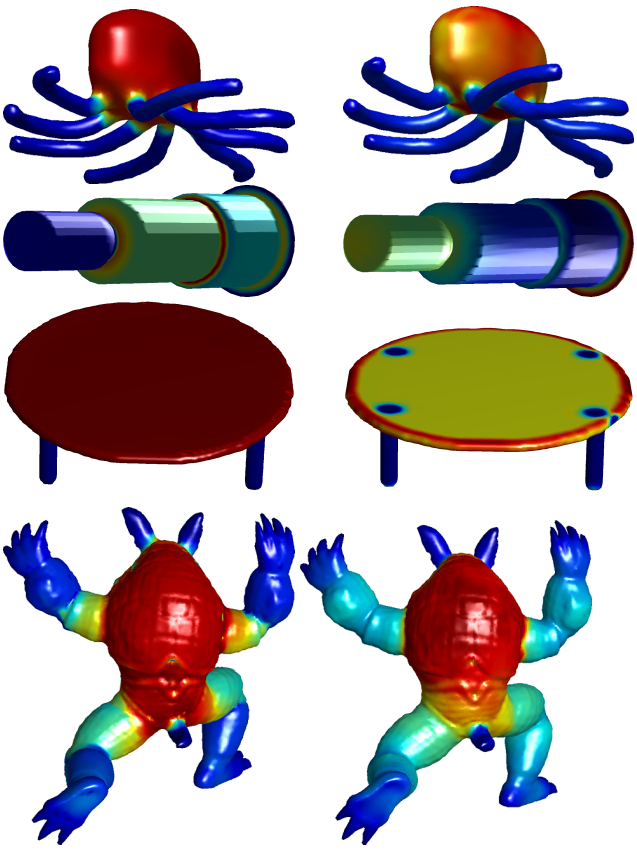
Figure 3 shows a visual comparison between CVF and SDF. From the pictures, we can see that semantic parts in these examples have more constant CVF in each part and higher variance when SDF is used. For example, the head of the octopus model has more consistent CVF values than SDF. For the tool model, the differences between parts are more distinct than those in SDF. Furthermore, SDF colormap shows four spots on the tabletop because the rays sent from these places would reach the leg of the table. Finally, in the legs, hands and torso of the Armadillo model, CVF provides better distinction between the neighboring parts and, again, more consistency within the parts.

5.2. Shape Segmentation

Segmenting a mesh into meaningful parts is a standard step toward part-based shape analysis. In order to have a fair comparison with SDF, we modified CGAL implementation by replacing the SDF features with CVF. The shape segmentation implemented in CGAL first infers a Gaussian Mixture Model (GMM) on the distribution of the feature values (SDF or CVF), and then each facet is assigned a soft label based on the inferred Gaussian Mixture Model. The final segmentation is achieved by applying the k-way graph cut. The data term of the graph cut is encoded by the soft labels, and the smoothness term is encoded by dihedral angle and edge length. Figure 4 shows segmentations using CVF with CGAL implementation. Figure 5 shows the segmentation evaluation results with CVF, SDF and other segmentation algorithms from the Princeton Segmentation Benchmark [3]. The evaluation is measured by Rand Index (RI) scores. Lower RI scores indicates higher similarity to human generated segmentations. Additional comparisons evaluated using other metrics such as consistency error, cut discrepancy, Hamming distance all show similar trend as RI and can be found in the supplementary materials.

The bar labeled as "CVF" shows the evaluation results using the original CVF (as oppose to CVF_{avg} that will be discussed later). We find that the RI score of CVF (0.17) is slightly better than the RI score of SDF (0.18). That fact that CVF does not provide more significant difference leads us to investigate deeper into the segmentation results. Table 1 shows the RI scores for all categories in the benchmark.

Note that the methods compared in the Benchmark used the features such as geodesic distance, cut perimeter, com-



(a) CVF (b) SDF

Figure 3. Side-by-side comparison between CVF and SDF values. In the ideal situation, each semantic part should have a constant feature value. In these pictures, the CVF values have lower variance in each semantic part than SDF values do.

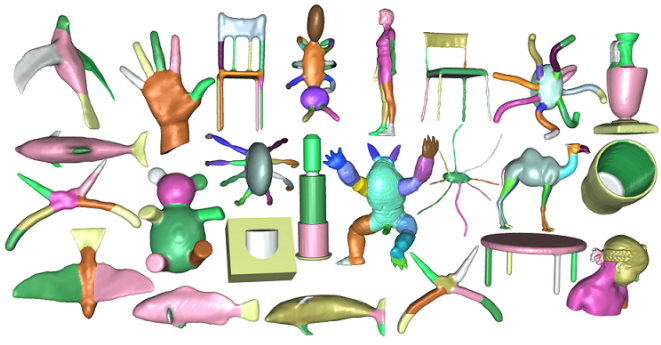


Figure 4. Segmentation created using CVF.

ponent area, etc. Some recent approaches do not introduce new features, but instead combing existing features via heuristics or learning techniques. For example, Kalogerakis et al. [10] use a set of descriptors to learn the segmentation, including curvature, PCA, shape contexts, geodesic distance, spin image etc. van Kaick et al. [21] employ a multi-step process: 1. Generate over-segments; 2. Merge

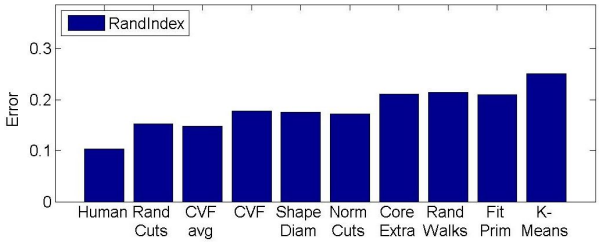


Figure 5. The comparison of segmentations using Princeton Segmentation Benchmark. The y-axis indicates errors measured in Rand Index values.

segments with SDF dissimilarity lower than a threshold; 3. Cut boundary refinement. Thus it is also worthwhile to try to incorporate CVF with the feature-rich approaches to understand how CVF can be combined with other features and how it performs in the combination.

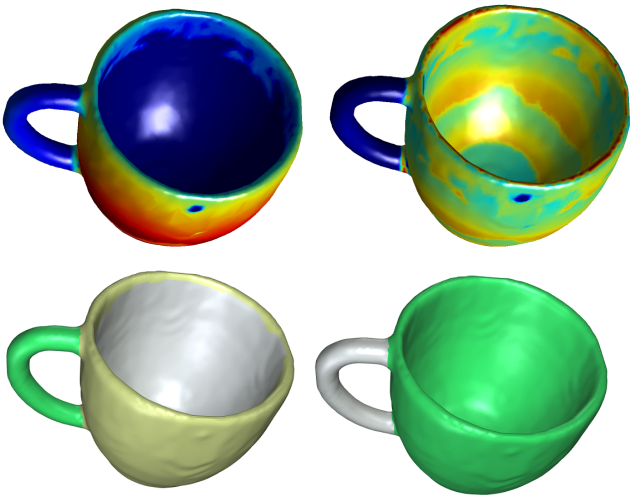
From Table 1, we found that segmentation using CVF in the cup, airplane, and bird categories performed poorly with respect to the human data collected in the benchmark. Figure 6 shows the distinct CVF values from the inside (concave part) and the outside (convex part) of the cup. The reason that CVF would be different for inside and outside parts is that the inside part is concave everywhere. The vertices in the inside parts can hardly continuously see any facets except the ones adjacent to it, while the outside part is convex, the vertices there can continuously see a lot of facets. The top left picture in Figure 6 shows the visualization of the cup. The bottom left shows the segmentation result using the CVF. The GMM-based shape segmentation separates the inside from the outside, which in some sense is a reasonable segmentation but makes the evaluation score rather low because most results from the benchmark segment the cup handle from the cup body. In the next section, we will discuss three variants of CVF that will address these issues.

5.3. CVF variants

CVF is versatile and can be defined in various ways to address several issues identified in plain CVF. In this section, we will describe three variants of CVF that provide better results than CVF. Due to space limitation, detailed comparisons between these variants can be found in the supplementary materials.

In the first CVF variant, we propose to average the CVF values of a facet (or vertex) p. That is, we send a ray in its counter-normal direction and get the nearest facet q intersected by the ray. Then the CVF_{avg} of p is the average between original CVF_p and CVF_q. The upper right picture in Figure 6 shows the visualization of the CVF_{avg}, and the lower right picture in Figure 6 shows the segmentation using CVF_{avg}. We can see that the results are more consistent to human segmentation.

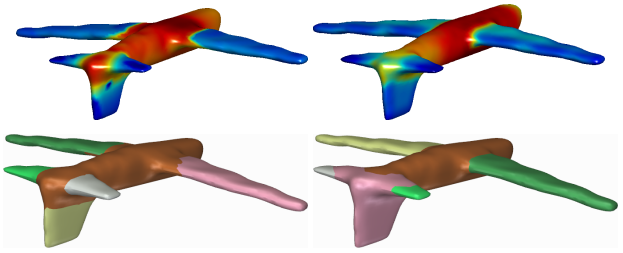
In Table 1, we show the RI scores of CVF_{avg}. We no-



(a) CVF (b) CVF_{avg}

Figure 6. Segmentations created using CVF.

tice that CVF_{avg} improves CVF significantly in the cup category (from 0.45 to 0.23). Moreover, surprisingly, CVF_{avg} also improved most of the RI scores from other categories (with the exceptions in airplane, plier, and fourleg). Consequently, we can see that CVF_{avg} outperforms SDF in 11 categories (out of 19). Even when SDF has lower RI scores, the differences between CVF_{avg} and SDF are usually small, except in the Airplane category. Figure 7 shows an example in this category.



(a) CVF (b) SDF

Figure 7. Limitation of CVF. In the left figures, the vertices (colored in yellow) between the airplane body and wings can continuously see both the wing and body. This results in large CVF values can causes the segmentation cuts to be at the middle of the wings.

The second CVF variant is called *strong CVF* that requires mutual continuous visibility. We say that two points p and q have *strong continuous visibility* if both $\text{CV}(p,q)$ and $\text{CV}(q,p)$. The first two columns in Fig. 9 show the colormap of CVF and strong CVF. In most examples, CVF and strong CVF are almost identical, except in the last example (bottom of Fig. 9). Notice the checker board-like pattern in CVF. The strong CVF provides much more consistent feature values than CVF in this example.

The third CVF variant is designed to address the issues

Table 1. Compare CVF and CVF_{avg} with SDF . The CVF_{avg} means the the feature values for a vertex are are achieved by averaging the CVF values between original CVF values and the the facet that is hit by the ray sent out from that vertex in its anti-normal direction (counter-normal direction.)

	RI			
	SDF	CVF	CVF_{avg}	Diff
Human	0.18	0.16	**0.14**	0.02
Cup	0.36	0.45	**0.23**	0.22
Glasses	0.2	0.21	**0.19**	0.02
Airplane	**0.09**	0.17	0.2	-0.03
Ant	**0.02**	0.04	0.04	0
Chair	0.11	0.1	**0.07**	0.03
Octopus	0.05	0.04	**0.03**	0.01
Table	0.18	0.12	**0.09**	0.03
Teddy	**0.06**	0.08	0.07	0.01
Hand	0.2	0.17	**0.13**	0.04
Plier	0.38	**0.21**	0.22	-0.01
Fish	0.25	**0.18**	**0.18**	0
Bird	**0.12**	0.25	0.18	0.07
Armadillo	**0.09**	0.11	0.1	0.01
Bust	**0.30**	0.31	**0.30**	0.01
Mech	0.24	0.21	**0.14**	0.07
Bearing	**0.12**	0.14	0.13	0.01
Vase	0.24	**0.18**	**0.18**	0
Fourleg	**0.16**	**0.16**	0.17	-0.01
Average	0.18	0.17	0.15	0.03

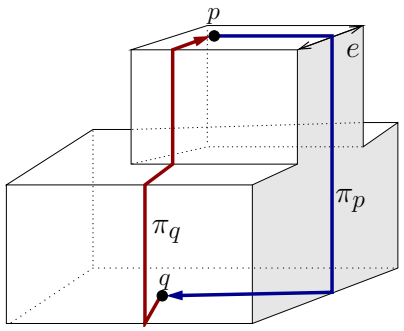


Figure 8. An example of the continuous visibility between two points v and w through different paths. The point v is continuously visible by the point w via path π_w and w is continuously visible by v via a much longer path π_v.

found in the airplane or bird-like models, in which the fuzzy regions between the wings and the body cause poor segmentation as shown in Fig. 7. Now, let us go back to the definition of continuous visibility. Even if both points p and q are continuously visible to each other, the geodesic paths that prove their mutual continuous visibility may be different. A such example illustrated in Fig. 8 shows that two different geodesic paths are taken by points p and q. One can see that path π_p that witnesses $\text{CV}(p,q)$ is much longer than the path π_q that witnesses $\text{CV}(q,p)$. In fact, we can make the path

π_v arbitrary long by moving the right most (shaded) face in Fig. 8 to its right. Intuitively, there is a negative correlation between the likelihood of v and w belong to the same part and the ratio between the continuous visibility path length and the length of the shortest geodesic length. Similarly, the edge length of e in Fig. 8 also has negative effect on the relationship between CVF and the likelihood of v and w belonging to the same part. In all cases, it will be desirable to take the visibility along the shortest geodesic path into consideration. Consequently, we propose that a measure of *weak continuous visibility* WCV. Given two points p and q, $\{\text{WCV}(p,q) = \textbf{TRUE}\}$ if p and q are visible to each other but the length of the invisible part of the shortest geodesic path connecting p and q is smaller than a user defined value τ. Let π be the the shortest geodesic path connecting p and q and let $\bar{\pi}_w$ and $\bar{\pi}_v$ be subset of π invisible to v and w, respectively. The function $\text{WCV}(p,q)$ returns **TRUE** if $\max(||\bar{\pi}_w||, ||\bar{\pi}_v||) \leq \tau$.

The last column in Fig. 9 shows the results of weak CVF. It is clear that weak CVF provides significantly sharper boundaries that can lead to better segmentation. See segmentation results using weak CVF with varying values of tau in the supplementary materials.

Since the CVF and its variants are based on visibility checking, one question may be raised by people is whether CVF and its variants are robust to noises. We believe that CVF is insensitive to bumps and noise. With the original or strong CVF in 3D, it is unlikely that noise or random bumps block the visibility of all paths between two points on the surface. In addition, the weak CVF can tolerate a certain amount of invisibility since it allows part of the shortest geodesic path to be invisible.

5.4. Application: Part-based Retrieval

In most shape retrieval problems, the objective is to retrieve a shape that is globally similar to a query shape. Since CVF can be used to generate segmentation and encode shape signature of each component, we follow the strategy described in [18] to provide a context-aware part retrieval method. We first generate the segmentation using CVF as described in the previous sections and construct a hierarchical part structure of the given shape from the segmentation. In this hierarchical part structure, the whole model is the root of the tree and the leaves are the components in the segmentation. Then each node in the tree is encoded with the CVF histogram as its signature. During the retrieval stage, we reduce the retrieval as a bipartite matching problem. Given a query component, we compare it with all the components in the dataset. To match two nodes (components) p and q, we build a bipartite graph between nodes of the paths leading the root to p and q in the hierarchy.

The total matching score is achieved by solving this maximum weight bipartite matching problem. Fig. 10 shows two examples and top 9 matching parts in the database.

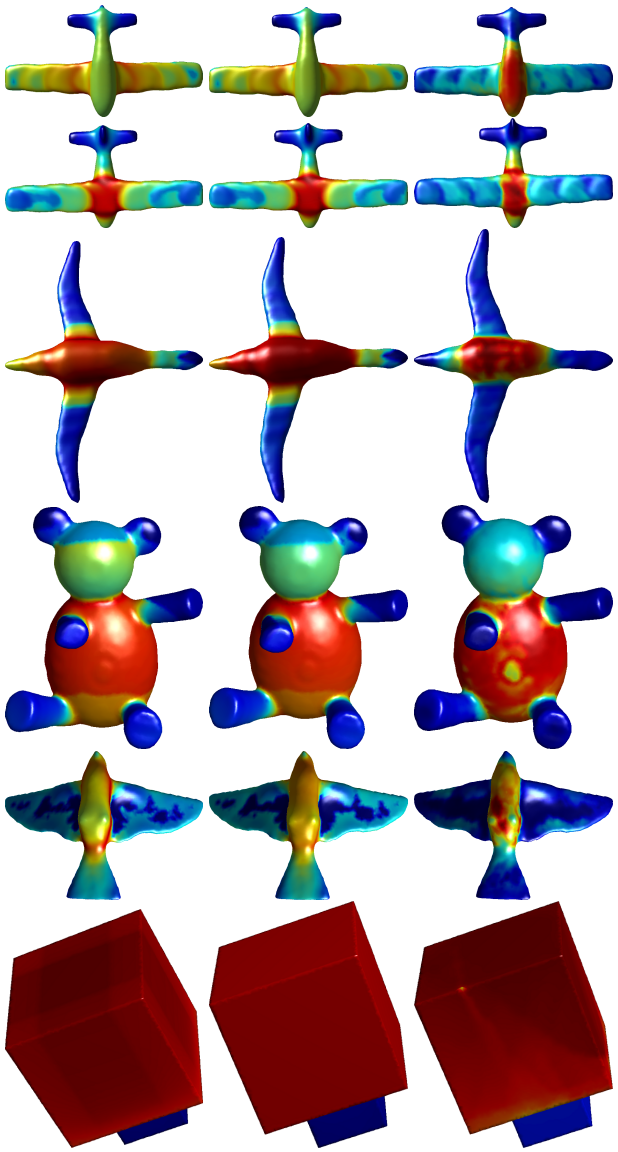


Figure 9. Results obtained from CVF (left) vs. strong CVF (middle) vs. weak CVF (right).

5.5. Application: Medial Axis Extraction

In Figure 11, we show the medial axis sample points generated by CVF. We use the idea of reference points in [14] to construct the skeleton points. Basically, for a vertex p, we send ray to the continuously visible region and each ray gives a reference point. We can see that the skeleton points provide a good approximation of the medial axis.

6. Discussion and Conclusion

In this paper, we showed a new shape feature called continuous visibility feature (CVF). We proposed two algo-

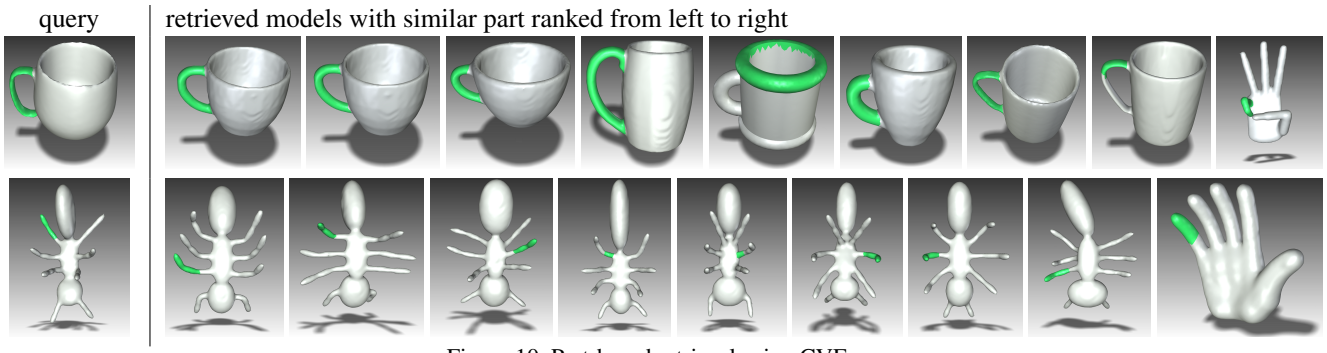


Figure 10. Part-based retrieval using CVF.

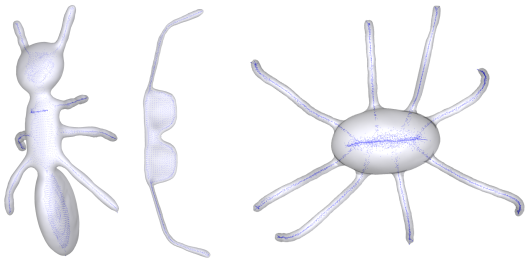


Figure 11. Medial-axis samples created using CVF.

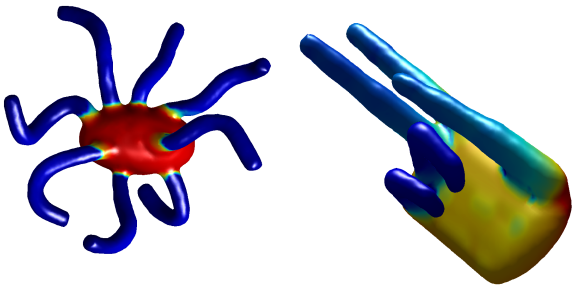


Figure 12. Invariance to pose change and deformation.

rithms to compute CVF for each vertex of a given mesh: one is based on BFS and the other one is based on identifying potential continuous visibility regions. We showed that the second approach is significantly faster than the BFS approach. In the experimental results, we show that CVF provides good results comparing to the methods in Princeton segmentation benchmark. If the CVF values are averaged between vertices on the opposite sides of their anti-normal directions, called CVF_{avg} then we see significant improvement over CVF and SDF. We demonstrated a couple of applications beyond shape segmentation, like shape retrieval and medial axis sampling.

Our experiments show that CVF is insensitive to pose change and deformation. It is true that the CVF value is not isometry invariant; however, it is the distribution of CVF that matters. Pose change or deformation does not change the CVF of one vertex only. Rather, many vertices' CVF change. If we consider the overall distribution of all vertices' CVF, for most poses, the overall distribution still preserves part-awareness. For example, in Fig 12, even though the legs of octopus (fingers of hand) have different poses or deformation, the distribution of CVF can still capture the part information.

We believe the continuous visibility feature would have more applications. Currently we only focus on the scalar representation, more applications of CVF can be explored with vector representation, a list of Booleans representing the continuous visibility of all vertex pairs. For example, modifying the existing features with the constraints of CVF's vector representation.

Acknowledgement

We would like to thank anonymous reviewers for helpful feedback. Authors Liu and Lien are supported in part by NSF IIS-096053, CNS-1205260, EFRI-1240459 and AFOSR FA9550-12-1-0238. And author Gingold is supported in part by the NSF (IIS-1451198 and IIS-1453018) and a Google research award.

References

[1] S. Asafi, A. Goren, and D. Cohen-Or. Weak convex decomposition by lines-of-sight. In *Computer Graphics Forum*, volume 32, pages 23–31. Wiley Online Library, 2013.

[2] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(4):509–522, 2002.

[3] X. Chen, A. Golovinskiy, and T. Funkhouser. A benchmark for 3d mesh segmentation. *ACM Transactions on Graphics (TOG)*, 28(3):73, 2009.

[4] R. R. Coifman and S. Lafon. Diffusion maps. *Applied and computational harmonic analysis*, 21(1):5–30, 2006.

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[6] A. Elad and R. Kimmel. On bending invariant signatures for surfaces. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(10):1285–1295, 2003.

[7] Q. Huang, V. Koltun, and L. Guibas. Joint shape segmentation with linear programming. In *ACM Transactions on Graphics (TOG)*, volume 30, page 125. ACM, 2011.

[8] V. Jain, H. Zhang, and O. van Kaick. Non-rigid spectral correspondence of triangle meshes. *International Journal of Shape Modeling*, 13(1):101–124, 2007.

[9] A. E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(5):433–449, 1999.

[10] E. Kalogerakis, A. Hertzmann, and K. Singh. Learning 3d mesh segmentation and labeling. *ACM Transactions on Graphics (TOG)*, 29(4):102, 2010.

[11] S. Katz and A. Tal. Hierarchical mesh decomposition using fuzzy clustering and cuts. *ACM Trans. Graph.*, 22(3):954–961, 2003.

[12] C. Li and A. B. Hamza. A multiresolution descriptor for deformable 3d shape retrieval. *The Visual Computer*, 29(6-8):513–524, 2013.

[13] G. Liu, Z. Xi, and J.-M. Lien. Dual-space decomposition of 2d complex shapes. In *27th IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Columbus, OH, Jun. 2014. IEEE.

[14] R. Liu, H. Zhang, A. Shamir, and D. Cohen-Or. A part-aware surface metric for shape analysis. In *Computer Graphics Forum*, volume 28, pages 397–406. Wiley Online Library, 2009.

[15] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[16] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin. Shape distributions. *ACM Transactions on Graphics (TOG)*, 21(4):807–832, 2002.

[17] R. M. Rustamov. Laplace-beltrami eigenfunctions for deformation invariant shape representation. In *Proceedings of the fifth Eurographics symposium on Geometry processing*, pages 225–233. Eurographics Association, 2007.

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

[19] S. Shlafman, A. Tal, and S. Katz. Metamorphosis of polyhedral surfaces using decomposition. In *Computer Graphics Forum*, volume 21, pages 219–228. Wiley Online Library, 2003.

[20] J. Sun, M. Ovsjanikov, and L. Guibas. A concise and provably informative multi-scale signature based on heat diffusion. In *Computer Graphics Forum*, volume 28, pages 1383–1392. Wiley Online Library, 2009.

[21] O. van Kaick, N. Fish, Y. Kleiman, S. Asafi, and D. Cohen-Or. Shape segmentation by approximate convexity analysis. *ACM Trans. on Graphics*, to appear, 2014.