Saliency for Animated Meshes with Material Properties

Abdullah Bulbul * Bilkent University
Cetin Koca † Bilkent University
Tolga Capin ‡ Bilkent University
Uğur Güdükbay § Bilkent University

Abstract

We propose a technique to calculate the saliency of animated meshes with material properties. The saliency computation considers multiple features of 3D meshes including their geometry, material and motion. Each feature contributes to the final saliency map which is view independent; and therefore, can be used for view dependent and view independent applications. To verify our saliency calculations, we performed an experiment in which we use an eye tracker to compare the saliencies of the regions that the viewers look with the other regions of the models. The results confirm that our saliency computation gives promising results. We also present several applications in which the saliency information is used.

Keywords: mesh saliency, computer animation, mesh simplification, perception, human visual system

1 Introduction

Animating three-dimensional meshes is a central problem in creating computer games, virtual environments, and digital special effects in movies. Animated meshes are now widely used for face and body animation of virtual humans, cloth animation, physically-based simulation, and deformable object animation. In addition, various research problems, such as animated mesh compression and transmission, mesh simplification, level-of-detail management, and progressive mesh representations, are also based on the efficient representation of animated 3D meshes.

Recently, the use of perception-inspired metrics for efficiently processing static meshes has gained attention; However, there has been comparatively less work done on perceptually guided animated mesh processing that incorporates the effect of temporal properties. Various research groups have proposed incorporating principles of perception in managing the level of detail for rendering static meshes [Luebke and Hallen 2001] [Reddy 2001] [Watson et al. 2001]. Lee et al. [2005] have proposed such a geometrical approach for static meshes, based on the concept of mesh saliency, a measure of regional importance in 3D meshes. This method uses the curvature property of the mesh and a center-surround mechanism to identify regions that are different from their surrounding context. Solutions have been proposed for animated mesh compression [Guskov and Khodakovsky 2004] [Ibarria and Rossignac 2003] [James and Twigg 2005] [Karni and Gotsman 2004] [Müller et al. 2005], representation [Briceo et al. 2003] [Shamir et al. 2000] and level-of-detail adjustment [Shamir and Pascucci 2001] [Kircher and Garland 2005]. Much of this work does not explicitly incorporate low-level models of human visual attention.

*e-mail:bulbul@cs.bilkent.edu.tr
†e-mail:ckoca@cs.bilkent.edu.tr
‡e-mail:tcapin@cs.bilkent.edu.tr
§e-mail:gudukbay@cs.bilkent.edu.tr

APGV 2010, Los Angeles, California, July 23 – 24, 2010.
© 2010 ACM 978-1-4503-0248-7/10/0007 $10.00

We propose an approach to compute the saliency value of animated meshes with material properties. In addition to the curvature property of the mesh, we take into consideration motion and material properties. The saliency computation is view independent, and thus, can also be used in view-independent and view-dependent applications.

The contributions of this paper are:

1. *Animated mesh saliency computation:* We propose a method that uses the acceleration and velocity of the mesh vertices to compute saliency.
2. *A framework for computing overall saliency of meshes with material properties:* We propose a method for incorporating color, texture, and orientation effects to the saliency of the mesh, and present a method to combine the effects of motion and material cues to the overall mesh saliency metric.
3. *Applications using saliency:* We present the usage of computed saliency values with several applications. The applications include mesh simplification, dynamic level-of-detail adjustment, and viewpoint selection.

We begin the paper with a survey of related work on animated meshes and saliency, and then state the main saliency computation algorithm. Next, we present the generalization of mesh saliency to animated meshes, and surfaces with colour and texture. We finish with results, discussion, and future directions of our work.

2 Background

Saliency, which characterizes the level of significance of the subject being observed, has been a focus of cognitive sciences for more than 20 years. It is closely related to many disciplines, including artificial intelligence, neuroscience, psychology, and recently, computer graphics. Itti and Koch [1998] describe one of the earliest methods used to compute the saliency of 2D images. Saliency is essentially the property of an object that gathers human attention. It is mainly related to difference of an object from its surroundings. Saliency can be seen as the bottom-up part of the visual attention mechanism, in which intentional factors such as the user's task do not have an effect. Object properties such as luminance, orientation, or geometry can be considered as bottom-up features; whereas the users' tasks or prior experiences can be considered as top-down features.This paper is mainly focused on bottom-up features while calculating the saliency map.

Lee and Varshney [2005] have introduced the saliency concept for 3D graphical models. In their work, the saliencies of mesh vertices are computed based on the mesh geometry. Their proposed saliency metric is based on the center-surround operator on Gaussian-weighted mean curvatures. They have used the computed saliency values to drive the simplification 3D meshes, using Garland's Qslim method for simplifying objects based on quadric error metrics [Garland and Heckbert 1997].

Another saliency metric and measure for the degree of visibility is proposed by Feixas et al. [2009]. Their saliency metric uses the Jensen-Shannon (JS) divergence of probability distributions by evaluating the average variation of JS-divergence between two polygons, yielding similar results to Lee and Varshney [2005].

A saliency map for selective rendering that uses colors, intensity, motion, depth, edges, and habituation (which refers to saliency reduction over time as the object stays on screen) has been developed using graphical processing unit (GPU) [Longhurst et al. 2006]. Their saliency map is based on the model suggested by Itti and Koch [2000].

To extract the critical points, the mesh saliency metric of [Lee et al. 2005] was modified by Liu et al. [2007], using Morse theory. In their work, they point out two disadvantages of Lee and Varshney's approach [2005]. One is that the Gaussian-weighted difference of fine and coarse scales can result in the same saliency values for two opposite and symmetric vertices, because of the absolute difference in the equation. The other is that combining saliency maps at different scales makes it difficult to control the number of critical points. Instead of the Gaussian filter, Liu et al. [2007] use a bilateral filter and define the saliency of a vertex as the Gaussian-weighted average of the scalar function difference between the neighboring vertices and the vertex itself.

Recently, Kim et al. [2010] performed a user study and compared various approaches to mesh saliency computation by quantifying the correlation between human eye movements and calculated saliencies. According to this study, the mesh saliency metric of Lee and Varshney [2005] models human-eye movements significantly better than a purely random model or a curvature-based model on 3D static meshes.

Saliency and other perceptually inspired metrics have also gained attention in level-of-detail (LOD) rendering and mesh simplification. For a flythrough in a scene, Reddy [2001] uses the models of visual perception, including vision metrics such as visual spatial frequency and contrast, to optimize the visual quality of rendering by removing the non-perceptible components of 3D scenes. Luebke and Hallen [2001] propose a perceptually-driven rendering framework that evaluates local simplification operations according to the worst-case contrast gratings and the worst-case spatial frequency of features that they can induce in the image. In their work, contrast grating is a sinusoidal pattern that alternates between two extreme luminance values, and the worst-case one is a grating with the most perceptible combination of contrast and frequency induced by a simplification operation. They apply the simplification only if a grating with that contrast and frequency is not expected, so they do not get any perceptible effect, which results in a high-fidelity model. A set of experiments has been performed using three groups of tasks for measuring visual fidelity [Watson et al. 2001]. These tasks are naming the model, rating the likeness of the simplified model against a standard one using a seven point scale, and choosing the better model of two equally simplified models using Q-Slim and V-clust [Rossignac and Borrel 1993]. The results of these experiments and some automated fidelity measures [Bolin and Meyer 1998] [Cignoni et al. 1998] show that automated tools are poor predictors of naming times but good predictors of ratings and preferences. Williams et al. [2003] extend Luebke and Hallen's [2001] framework to models with texture and light effects. Howlett et al. use an eye tracker to identify the salient regions and the fixation time on those regions, and they modify Q-Slim to simplify those regions with a weight value [2004]. Because of experiments similar to Watson et al.'s work [2001], it is shown that the modified Q-Slim performs better on natural objects than on man-made artifacts, which indicates that saliency detection is very important.

Although mostly used for simplifying meshes, saliency has also been used as a viewpoint selection criterion. In Yamauchi et al.'s work [2006], viewpoints are selected from among a sample point set, forming the vertices of a graph on the bounding sphere of an object. The graph is partitioned according to the degree of similarity between its edges, and sorted according to the edges' geometric saliency value. A recent work by Shilane uses a database of objects to measure the distinctiveness of different regions of an object [2007]. It is based on the idea that if a region has a unique shape that is used to differentiate the object from other objects, that region is an important part of the object. It works by selecting several random points as centers of overlapping spheres over the surface and generating shape descriptors from the surfaces covered by those spheres. Next, a measurement is taken of how distinctive each region is with respect to a database of multiple object classes, and if the best matches of a region are all from the object's own class, that region is distinctive. Although a database is required, it gives better results than Lee and Varshney's approach [2005] in terms of simplification quality.

Saliency has also been studied for illustration. It is shown that visual attention can be directed by increasing the saliency at user-selected regions using geometric modification [Kim and Varshney 2008]. With a weight change in the center-surround mechanism, Kim and Varshney modify mean curvature values of vertices by using bilateral displacements and use eye trackers to verify that the change increases user attention. Mortara and Spagnuolo use the saliency information to generate thumbnails of meshes [2009]. In addition to the bottom-up saliency calculation, they use semantic information also to determine the important parts of a mesh.

3 Approach

We propose a metric for calculating the saliency of 3D meshes, which is applicable to vertex-animated models as well as static models. Most of the previous saliency detection methods focus on the shape and curvature features of 3D meshes. On the other hand, in addition to these features, motion and material properties of 3D meshes also affect saliency in a significant way. Therefore, for our animation-based saliency calculation task, the proposed solution considers per-vertex curvature, material, and animation properties of 3D animated meshes.

The combination of different features, such as color, orientation, spatial frequency, brightness, direction of movement, into a single saliency model requires an integrated model of attention. The feature-integration theory of attention [Treisman and Gelade 1980], which has been used successfully for 2D images, suggests that the visual scene is initially coded along a number of separable dimensions; and the contribution of any features which are present in the same region are combined. Based on this theory, Itti et al. have proposed a model for integration of the different features in 2D images [1998]. Our model of saliency for 3D animated models uses a similar approach.

In order to derive an integrated approach for saliency calculation, our metric considers multiple features of animated meshes such as their color, geometry, and motion; each as a separate channel. The general structure of the proposed approach is shown in Figure 1. The 3D model is decomposed into a set of dimensions, with each dimension stored in a separate feature map. Different regions of the 3D model then compete for saliency within each feature map, and only those regions that stand out locally from their neighborhood in different scales are kept. Then, the saliency values computed for each dimension are combined into a master saliency map, which approximates the overall attended regions of the animated 3D mesh.

Our saliency calculation depends on the center-surround mechanism of human visual attention [Itti et al. 1998] [Lee et al. 2005]. The mechanism captures the regions that are spatially different from their surroundings. For each feature, we compute the salient vertices by a set of center-surround operations. In these operations, vertices in a small neighborhood of a vertex v constitute the center, and vertices in a larger neighborhood constitute the surround. The

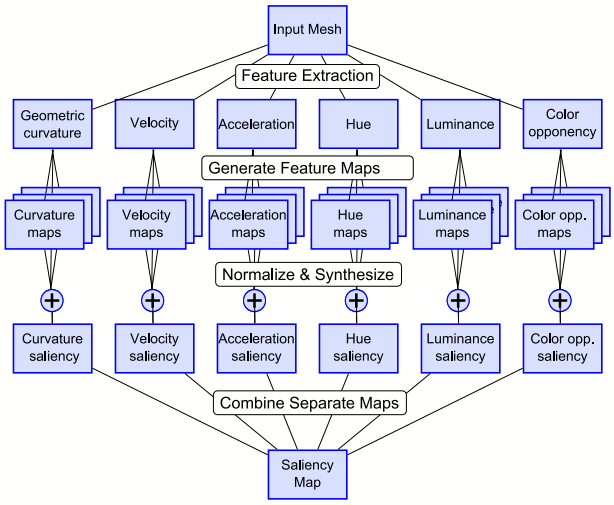


Figure 1: *The proposed saliency computation framework.*

across-scale difference of a feature between the central region and the surrounding region determines the saliency.

We compute different feature maps of multiple scales to account for the saliency in different scales of the mesh. For example, a small scale saliency map may detect the movement of a finger in a human model but it will fail to detect a larger-scale movement such as that of a leg. A large-scale saliency map will fail in the former case and succeed to show the saliency correctly in the latter case.

As shown in Figure 1, our saliency computation framework consists of four steps.

3.1 Feature Extraction

We use geometry, velocity, acceleration, hue, color opponency, and luminance features of a vertex in our saliency computations. Each feature is calculated as follows for each vertex:

Geometry: This feature is used for computing the saliency of a 3D mesh due to its shape. Curvature is a significant feature of a vertex that can indicate its distinctiveness among others. Therefore, for geometry-based computations, we use the mean curvatures of vertices, as previously proposed by Lee and Varshney [2005], using Meyer et al.'s method for curvature computation [Meyer et al. 2002].

Velocity: The velocity of a vertex is calculated by taking the positional difference of a vertex in consecutive frames as follows:

$$vel(v_i, f_i) = \frac{p(v_i, f_j) - p(v_i, f_{j-1})}{diagonal}, \quad (1)$$

where v_i stands for vertex i,; p stands for position which is a vector of length three; f_j stands for j^{th} frame. $diagonal$ is the length of the axis aligned diagonal of the mesh's bounding box. The division by $diagonal$ makes our method scale independent.

Acceleration: The acceleration computation is very similar to the velocity calculation. It is calculated by taking the difference of velocities on consecutive frames but this time diagonal is not used because its effect is already present in velocities.

$$acc(v_i, f_i) = vel(v_i, f_j) - vel(v_i, f_{j-1}), \quad (2)$$

where $acc(v_i, f_i)$ stands for acceleration of vertex i at frame j.

Hue: We extract hue values from the RGB color of a vertex and map them to the 0-360 interval. Let r, g, and b denote the red, green and blue components of the color of a vertex. Corresponding hue value is extracted as follows.

$$hue = \begin{cases} 0, & \text{if } max = min \\ \left(\frac{60\times(g-b)}{max-min} + 360\right) \bmod 360, & \text{if } max = r \\ \left(\frac{60\times(b-r)}{max-min} + 120\right) \bmod 360, & \text{if } max = g \\ \left(\frac{60\times(r-g)}{max-min} + 240\right) \bmod 360, & \text{if } max = b \end{cases} \quad (3)$$

where max and min stand for maximum and minimum values among r, g, and b.

Because the hue values wrap around (e.g., the value 359 is close to 1), while computing the center-surround differences (explained in Section 3.2) we use the smaller distance between two values. For example, the distance between values 350 and 10 is 20 instead of 340.

Color opponency: For the central area of the visual field, the neurons in the primary visual cortex are excited by one of the colors in the Red-Green and Blue-Yellow pairs, and inhibited by the other color. The opposite holds for the surrounding area. Therefore, presence of a color in the center increases saliency of a region, if this region is surrounded by the opponent color. We use the color opponency values described in [Itti et al. 1998].

Luminance: For each vertex, we compute the luminance feature by calculating the average of the RGB components of the vertex.

3.2 Generating Feature Maps

After processing all features of the vertices, we calculate the Gaussian-weighted center-surround differences for several center-surround scales, and then we generate a separate feature map for each pair of center-surround scale and feature. Define this feature map as *featuremap*(c, s, f) which stores the feature for each vertex of a 3D mesh, where c and s stand for the center and surround levels and f stands for the feature. For scale (c, s) and feature f the entry for vertex i in the map is generated as follows:

- Let the neigborhood $N(v, d)$ of vertex v be the set of vertices that have a Euclidean distance smaller than d to vertex v. First, we calculate the Gaussian-weighted average of feature f for the vertices that are in $N(v_i, 2c)$ [Itti et al. 1998].

$$G(f, c, v_i) = \frac{\sum_{x\in N(v_i,2c)} f_x exp\left(-\frac{||v_x - v_i||^2}{2c^2}\right)}{\sum_{x\in N(v_i,2c)} exp\left(-\frac{||v_x - v_i||^2}{2c^2}\right)} \quad (4)$$

- Then, the absolute center-surround differences are stored in the feature map.

$$featuremap(c, s, f, i) = |G(f, c, v_i) - G(f, s, v_i)|, \quad (5)$$

where c is the fine (center) scale and s is the coarse (surround) scale.

For center and surround distances, Itti et al. [1998] have used 2, 3, and 4 pixels for fine scale c and $c+\delta$ pixels for coarse scale s, where $\delta \in \{2, 3\}$, resulting in six different center-surround levels. On the other hand, Lee and Varshney [2005] used 2ϵ, 3ϵ, 4ϵ, 5ϵ, 6ϵ, where ϵ is 0.3% of the diagonal of the bounding box for the center and the

surround is the double of the center. In this case, five different levels of center-surround scales are considered. The largest surround scale of the second approach covers only $12 \times 0.3\%$ = 3.6% of the diagonal of the bounding box, which is not large enough. Hence, we also need to consider larger center-surround scales as well as the narrow scales. The list of center-surround levels used for our calculations follows:

$$L = \{2\epsilon, 3\epsilon, 5\epsilon, 8\epsilon, 13\epsilon, 21\epsilon, 34\epsilon, 55\epsilon\},$$
$$(c, s) \in \{x, y | x = L[i] \wedge (y = L[i+1] \vee y = L[i+2])\} \quad (6)$$

where c stands for center, s stands for surround and ϵ again means 0.3% of the diagonal. Using the Fibonacci sequence decreases the cost of calculating neigborhoods of different scales such that only eight different neighborhoods are calculated to get thirteen different center-surround levels. In this case, the largest surround level covers $55 \times 0.3\%$ = 16.5% of the bounding box.

3.3 Normalization of Feature Maps

After calculating the separate featuremaps, the next step is to combine the maps that belong to the same feature. This is done by linear addition after normalizing the maps using the normalization method defined by Itti et al. for 2D images [1998]. For example, all featuremaps related to velocity are normalized and summed up to determine the velocity-based saliency map. This normalization method works as follows:

- All values in the feature map are mapped to a fixed range $0 - M$ so that the maximum saliency becomes M.
- All vertices with a saliency greater than all of its neighbors' saliencies are signed as local maximums. Let a be the average value of the saliencies of the local maximums.
- The feature map is multiplied with $|M - a|^2$.

Using this normalization technique suppresses the maps in which the saliency values are distributed homogeneously, whereas a map with an outstandingly salient point is promoted. After normalization, all maps related to a feature are linearly added and we get a saliency map for each feature.

Other normalization methods, such as iterative competition between salient locations and simple normalized summation, have been proposed by Itti and Koch for 2D images [1999]. For 3D animated models, however, this approach is the most suitable one regarding the computation-accuracy tradeoff.

After computing the saliency maps based on each feature, we combine these maps to a final saliency map. This combination is performed by linear addition as each feature map has already been normalized in the previous step.

4 Results

In this section, we demonstrate the results of our saliency metric. Each feature has different contribution to the saliency calculation; therefore, it is useful to show the effect of each feature separately before demonstrating the final saliency map.

Geometry based saliency map: It identifies the important regions of a 3D mesh according to its shape. It can be considered as the static saliency of a 3D mesh. Figure 2(a) shows salient parts of a horse model due to its geometry. As seen in the figure, the parts of the mesh that are outstanding according to their mean curvatures (e.g., eyes, feet, joint of tail) are computed as salient.

Velocity and acceleration based saliency maps: Velocity and acceleration features are used to obtain the regions that are salient due to their motion. Note that high saliency of a region is not the direct result of high velocity or acceleration of this region. Saliency is related to the difference of this region's motion with respect to the surrounding area.

Figures 2(b) and 2(c) show examples of velocity-based and acceleration-based saliency maps, respectively. Although these two features are similar, each one is sensitive to a different behaviour of a motion. For example, while velocity-based saliency map finds left feet as salient (Figure 2(b)), the acceleration-based saliency map finds the front-right foot of the horse model as more salient (Figure 2(c)). In Figure 2(d), the combination of motion related saliency maps are shown.

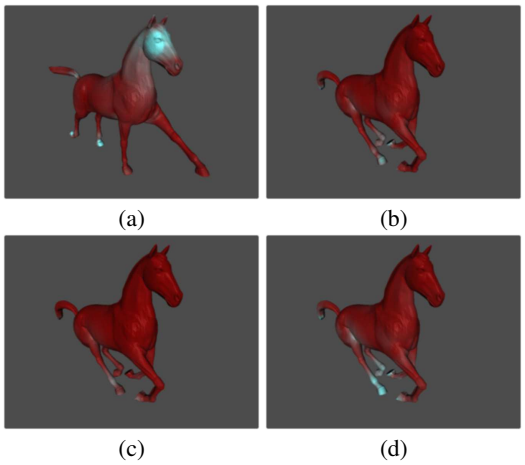


Figure 2: *The calculated saliencies based on geometric mean curvature (a), velocity (b), and acceleration (c) in a horse model. The image in (d) shows the combined saliency map of the velocity and acceleration features. Light-colored areas show the salient regions and are emphasized for illustration purposes.*

Hue, color opponency, and luminance based saliency maps: In addition to the shape and motion related attributes of 3D meshes, our saliency metric also considers per-vertex material properties. Figure 3(b) shows the hue-based saliency map of the cloth model shown in Figure 3(a). As seen in this figure, the hue-based saliency map highlights the regions that have different hue values than their neighbors. Color opponency-based saliency map identifies the regions that are surrounded by the opponent color. In Figure 3(c), we can see that the green parts are indicated as salient since in the original image these regions are surrounded by the opposite color (red). Another color related attribute which is used to identify the salient regions is the intensity of color. The luminance-based saliency map, which is shown in Figure 3(d), points out the salient regions due to their color intensity.

Final saliency map: All features are combined to obtain a final saliency map. In Figure 4, example saliency maps belonging to different animated models are shown. Although these images are only snapshots of the animations of the models, looking at the saliency images we can understand the salient regions of the models due to their animations, such as the legs of the horse model and the left part of the cloth model.

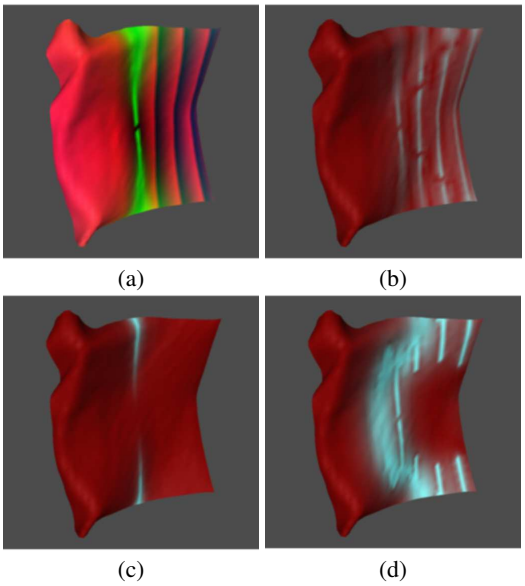


Figure 3: *The animated cloth model (a). The calculated saliencies based on hue, color opponency, and luminance are shown in (b), (c), and (d), respectively. Light-colored areas show the salient regions and are emphasized for illustration purposes.*

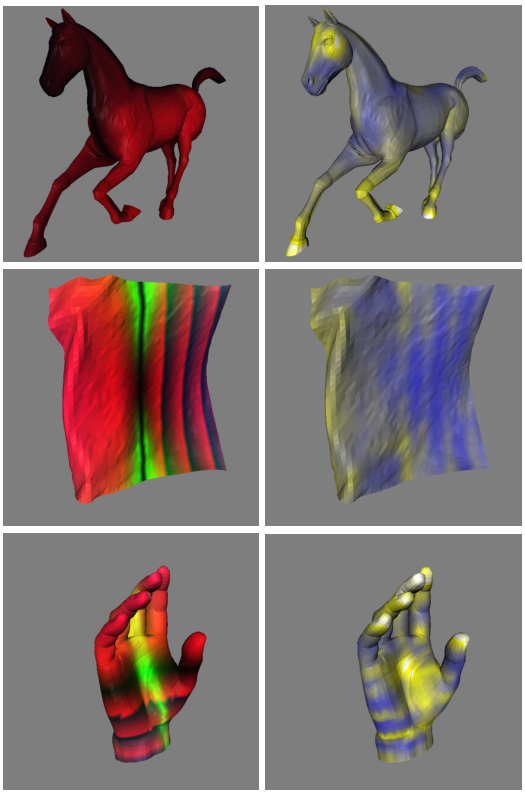


Figure 4: *Left: The models with their original views, right: the final saliency maps of these models.*

4.1 Experiments

To verify our saliency calculations, we have performed an experiment in which our aim is to compare the calculated saliency maps to the actual regions that are looked at by the users. For this purpose, we have used a Tobii 1750 eye-tracker. In the experiment, 3 short video sequences (Figure 5) were shown to 12 subjects and the points that are looked at were captured. The duration of each animation is approximately 15s. The subjects have normal or corrected to normal vision and they freely viewed the animations with no assigned task. In order to evaluate our system, we have followed the steps shown below:

1. We have extracted the saliency maps for the animations that are used in the experiment.
2. For each user, using the eye-tracking results, we have marked the points of the animations that are looked at.
3. For each frame of the animation;
   (a) We have calculated average saliency of all visible points, call it *average saliency*.
   (b) We have calculated average saliency of all points that are marked as looked at, and compared this to *average saliency*.

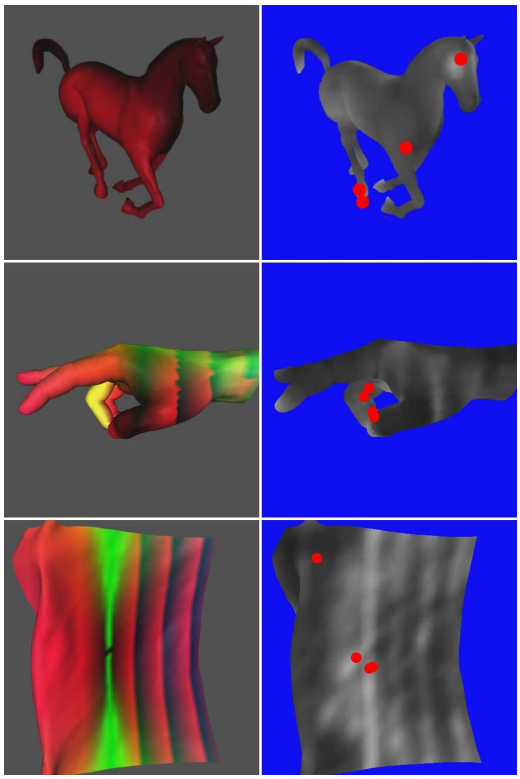


Figure 5: *Samples from the three animation sequences used in the experiment. Left: original frames, right: saliency maps of the frames on the right (red dots indicate the regions that are looked at by the subjects).*

There are several limitations in this experiment. Firstly, there is a margin of error, which is quite large considering that our saliency computation is performed over vertices. In order to tolerate this error, when calculating the average saliency of the points we have used a small circular neighborhood of the point that is looked at. The radius of this circle is approximately 5% of the visible region. While this approximation tolerates the error to some extent, it causes the calculated average saliency of the points that are looked at to be closer to the average saliency of all visible points.

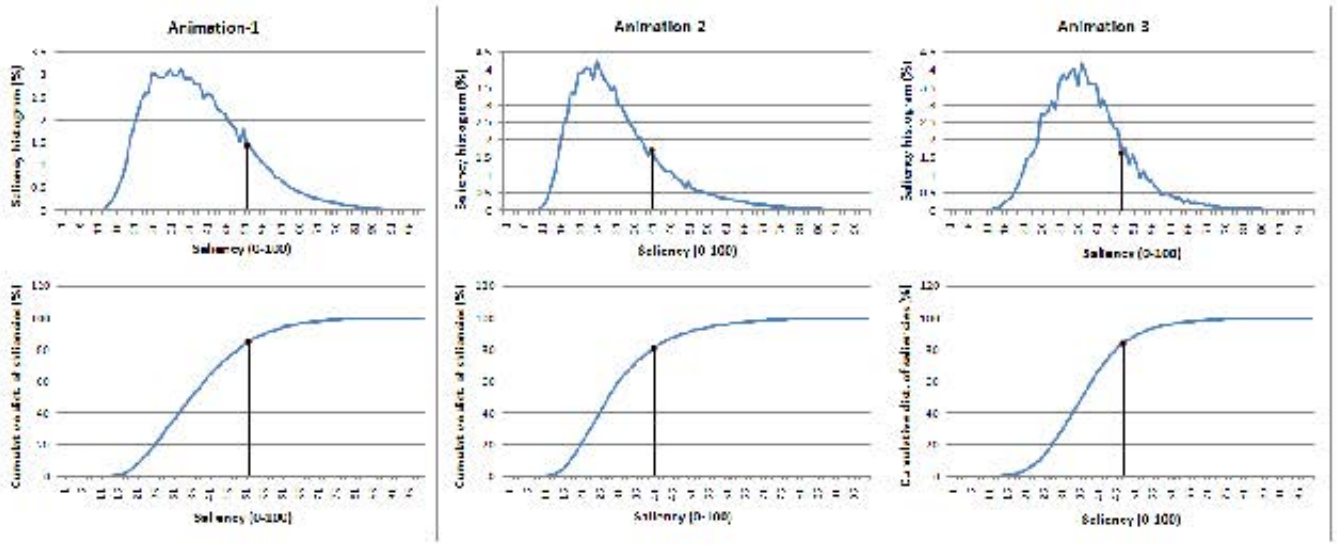


Figure 6: *The results for the animation sequences used in the experiments. Plots at the top row show the average saliency histogram for each animation and the plots at the bottom show the cumulative distribution of saliency values. In each plot, the black dot indicates the average saliencies of the points that are looked at.*

Another limitation is that there is a delay between a motion and the users response to that motion. To tolerate this error, we take the time of eye-tracker backwards by 0.4s, to take into account the subjects' reaction delay due to their perceptual processing of the shown animation.

Despite these limitations, the eye tracking results show that the subjects look at the regions with significantly higher overall saliency than average. In the top row of Figure 6, saliency histograms for each saliency range are shown; whereas the bottom row shows the cumulative distribution of saliency values. Note that the saliency of visible points in these animations are scaled to [0,100] range in order to be able to compare the results. In these plots, the average saliency values of all points that are the output of eye tracker are also shown. As shown in the bottom row of Figure 6, the average saliency of regions that the subjects look at for animation 1, 2, and 3 are ranked in the top 17%, 21%, and 19% of all visible points, respectively.

Furthermore, we have tested the validity of the proposed saliency approach as follows. In addition to the actual users, we have assumed 100 virtual users who look at random screen positions. Then, we have compared the calculated average saliencies of the regions that the actual subjects look at to the regions that the virtual users looked at. In this comparison, we have not considered the points that are not on the animated models. Figure 7 shows that the actual users look at more salient regions compared to the virtual users and the difference is statistically significant ($p < 0.05$) according to the applied t-test.

These results show that our saliency metric identifies the regions that are likely to be looked at and determines the important parts of a 3D mesh. Moreover, these results can be considered as the worst case due to the limitations explained before.

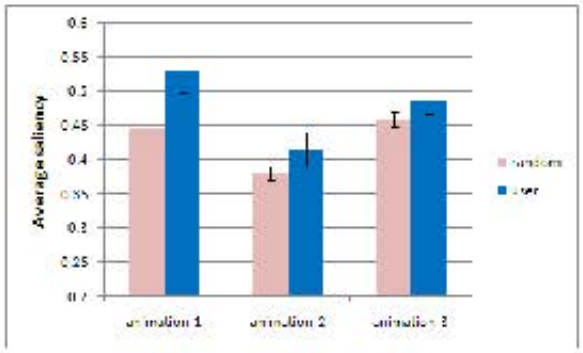


Figure 7: *Comparison between the calculated average saliencies of the regions that are looked at by the actual users, and the randomly generated virtual users. Error bars indicate 95% confidence intervals.*

5 Applications

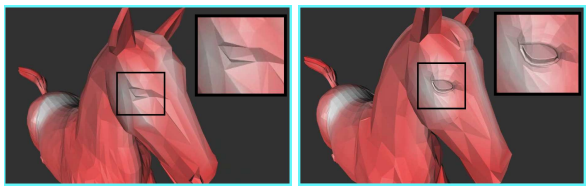
The computed saliency map can be used for different applications, including mesh simplification, viewpoint-selection [Lee et al. 2005], persuading attention [Kim and Varshney 2008], and accelerating global illumination computations [Yee et al. 2001].

Our saliency calculation is performed in a view independent way. Once the saliency values are computed as a preprocessing step, they can be stored as per-vertex attributes in a 3D animated mesh and can be used in different applications without recomputing. We present several applications in which the usage of saliency has a significant importance.

5.1 Mesh Simplification

The first application in which the usage of saliency is illustrated is simplification of 3D animated meshes. This application is based on the idea that a vertex with a higher saliency value means that it resides on a perceptually more interesting region of the mesh. Our goal is to delay the simplification of the salient parts of the mesh, because those parts are presumably the parts of the mesh where the viewers focus on. On the other hand, saliency maps themselves do not suffice to be used as the main simplification metric, but they are rather support data, i.e., as a heuristics for simplification.

We have used the Quadric Error Metric (QEM) described in [Gar-



(a) (b)

Figure 8: *The animated horse model is simplified using quadric error metrics (a) and using our saliency-based simplification method (b). Both of the models have 4600 faces.*

land and Heckbert 1997] as our main simplification metric and incorporated saliency maps as a supporting decision factor. Originally, QEM has been used to compute a score for each vertex in the mesh determining their simplification possibility. In our framework, these QEM scores are blended with the saliency value of the vertices and the final score is assigned to the vertices, according to the following equation:

$$w_i = -q_i \times (1 - \alpha) + s_i \times \alpha \times \mu, \tag{7}$$

where w_i is the final weight, q_i is the QEM score, s_i is the saliency score of vertex i, α is the weight of the saliency score in simplification and μ is the normalization factor of the saliency values.

In our experiments, we have selected $\alpha = 0.5$ and $\mu = 10^{-4}$. After the scores are assigned to each vertex, the vertices are placed on a heap, i.e., a priority queue, according to their scores. A vertex with a high score means that it has a lower QEM error score and/or a higher saliency; thus, we delay the contraction of the vertices with higher scores. For this purpose, we use a min-heap where vertices with lower scores are placed at the top. These vertices are consequently simplified earlier without degrading the quality of the salient regions of the mesh and without increasing the QEM error rate significantly. When two vertices are contracted we assign a new score to the resulting vertex. This score is obtained by the own QEM score of the new vertex and a saliency value which is the computed by linearly interpolating the saliency values of the contracted vertices. This interpolation is performed depending on the distance of the new vertex to the contracted vertices. Figure 8 shows a frame of the result of our simplification method applied on the horse model. The figure illustrates that the new method successfully preserves the salient regions such as the eye of the horse model, and the colored region on the back.

5.2 Dynamic Level-of-Detail

Dynamically adjusting the level-of-detail of the models in the scene depending on the display area they allocate is a widely used technique. This technique provides rendering efficiency by avoiding the time to render the imperceptible details of complex meshes. The view-independent saliency values that are computed as a preprocess are used for real-time level-of-detail adjustment. The idea here is that the viewers mostly focus on objects that are closer to them in the scene, or the objects that allocate the most space on the screen. Therefore, we can further simplify objects that are further away from the viewer. This simplification is performed considering the saliency values of the vertices as explained in Section 5.1.

5.3 Viewpoint Selection

The saliency information can also be used to automatically determine the viewpoint for an animation. Since the saliency values indicate the significant regions of 3D meshes, automatic selection of the viewpoint through animation would be a useful tool for a number of applications. A similar approach can be used for automatically creating a thumbnail for animated 3D meshes; however, in this case, it would be better to use the saliency that is generated using only the geometric and the material properties of a 3D mesh without the motion related attributes since a thumbnail does not include animation. Alternatively, the frame with the largest saliency can be selected.

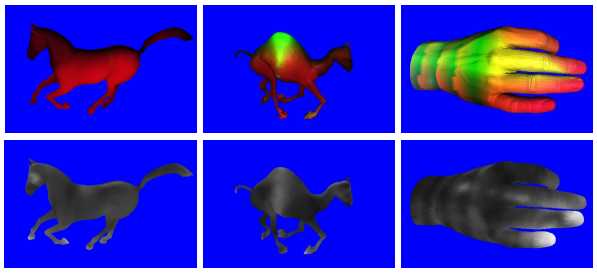


Figure 9: *Selected viewpoints for several meshes.* ***top:*** *original views of the models,* ***bottom:*** *corresponding saliency maps.*

In order to select the viewpoint automatically, we examine a spherical region around a 3D mesh and the viewpoint in which the total saliency of all visible points reaches the maximum is selected as the viewpoint. Figure 9 shows the selected viewpoints for several animated models. This technique can further be extended for automatic control of camera position in an animation.

6 Conclusion

We have proposed a new saliency metric to calculate the level of significance of 3D animated meshes. To be able to identify the saliencies due to different properties, the proposed metric takes various features of 3D models into account while computing the saliency. These features are related to the shape, color, and motion of an animated 3D model. The saliencies due to each feature are computed separately and they are normalized and combined to calculate the final saliency map. The proposed method is view independent; thus, the saliency map can be calculated for an animated 3D mesh once and can be used for both view-dependent and view-independent applications.

In order to validate our saliency metric, we have performed a user study. In this study, we used an eye-tracker to capture the regions of the models that the users look at. We compared the calculated saliencies of these regions to the saliencies of the other regions of the models. According to the results of this experiment, the users look at the regions that are significantly more salient than the average saliencies of the models, which shows that we correctly identified the salient regions. We recommend several applications in which the saliency information could be useful.

Acknowledgements

The horse and camel gallop animations were made available by Robert Sumner and Jovan Popović from the Computer Graphics Group at MIT. The hand model is obtained from The University of Utah 3D Animation Repository. We sincerely appreciate Dr. Kürşat Çağıltay and Human-Computer Interaction Research Group at Middle East Technical University for letting us to use their eye tracker for our experiments. We are grateful to Rana Nelson for proofreading and suggestions. We are also grateful to Ahmet Tolgay and Umut Uyumaz for their efforts in the previous stages of this work.

We appreciate the efforts of all subjects who participated in the experiments.

References

BOLIN, M. R., AND MEYER, G. W. 1998. A perceptually based adaptive sampling algorithm. In *Proceedings of ACM SIGGRAPH '98*, 299–309.

BRICEO, H. M., SANDER, P. V., MCMILLAN, L., GORTLER, S., AND HOPPE, H. 2003. Geometry videos: a new representation for 3D animations. In *Proceedings of ACM SIGGRAPH '03*, 136–146.

CIGNONI, P., ROCCHINI, C., AND SCOPIGNO, R. 1998. Metro: measuring error on simplified surfaces. *Computer Graphics Forum 17*, 167–174.

FEIXAS, M., SBERT, M., AND GONZALEZ, F. 2009. A unified information-theoretic framework for viewpoint selection and mesh saliency. *ACM Transactions on Applied Perception 6*, 1, Article no. 1, 23 pages.

GARLAND, M., AND HECKBERT, P. S. 1997. Surface simplification using quadric error metrics. In *Proceedings of ACM SIGGRAPH '97*, 209–216.

GUSKOV, I., AND KHODAKOVSKY, A. 2004. Wavelet compression of parametrically coherent mesh sequences. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 183–192.

HOWLETT, S., HAMILL, J., AND O'SULLIVAN, C. 2004. An experimental approach to predicting saliency for simplified polygonal models. In *Proceedings of the 1st Symp. on Applied Perception in Graphics and Visualization*, 57–64.

IBARRIA, L., AND ROSSIGNAC, J. 2003. Dynapack: space-time compression of the 3D animations of triangle meshes with fixed connectivity. In *Eurographics/SIGGRAPH Symposium on Computer Animation*, 126–135.

ITTI, L., AND KOCH, C. 1999. A comparison of feature combination strategies for saliency-based visual attention systems. *Journal of Electronic Imaging 10*, 161–169.

ITTI, L., AND KOCH, C. 2000. A saliency-based search mechanism for overt and covert shifts of visual attention. *Vision Research 40*, 10-12, 1489–1506.

ITTI, L., KOCH, C., AND NIEBUR, E. 1998. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

JAMES, D. L., AND TWIGG, C. D. 2005. Skinning mesh animations. *ACM Trans. Graph 24*, 399–407.

KARNI, Z., AND GOTSMAN, C. 2004. Compression of soft-body animation sequences. *Computers & Graphics 28*, 25–34.

KIM, Y., AND VARSHNEY, A. 2008. Persuading visual attention through geometry. *IEEE Transactions on Visualization and Computer Graphics 14*, 4, 772–782.

KIM, Y., VARSHNEY, A., JACOBS, D. W., AND GUIMBRETIÈRE, F. 2010. Mesh saliency and human eye fixations. *ACM Transactions on Applied Perception 7*, 2, 1–13.

KIRCHER, S., AND GARLAND, M. 2005. Progressive multiresolution meshes for deforming surfaces. In *Eurographics/ACM SIGGRAPH Symposium on Computer Animation*, 191–200.

LEE, C. H., VARSHNEY, A., AND JACOBS, D. W. 2005. Mesh saliency. *ACM Transactions on Graphics (Proc. of SIGGRAPH'05) 24*, 3, 659–666.

LIU, Y. S., LIU, M., KIHARA, D., AND RAMANI, K. 2007. Salient critical points for meshes. In *Proceedings of ACM Symposium on Solid and Physical Modeling*, 277–282.

LONGHURST, P., DEBATTISTA, K., AND CHALMERS, A. 2006. A GPU based saliency map for high-fidelity selective rendering. In *Proceedings of the 4th International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*, 21–29.

LUEBKE, P. D., AND HALLEN, B. 2001. Perceptually-driven simplification for interactive rendering. In *Proceedings of Eurographics Workshop on Rendering Techniques*, 223–234.

MEYER, M., DESBRUN, M., SCHRÖDER, P., AND BARR, A. H. 2002. Discrete differential-geometry operators for triangulated 2-manifolds. In *Proceedings of VisMath*.

MORTARA, M., AND SPAGNUOLO, M. 2009. Semantics-driven best view of 3D shapes. *Computers & Graphics 33*, 3, 280–290.

MÜLLER, K., SMOLIC, A., KAUTZNER, M., EISERT, P., AND WIEGAND, T. 2005. Predictive compression of dynamic 3D meshes. In *Proceedings of IEEE International Conference on Image Processing*, vol. 1, I–621–4.

REDDY, M. 2001. Perceptually optimized 3D graphics. *IEEE Computer Graphics and Applications 21*, 5, 68–75.

ROSSIGNAC, J., AND BORREL, P. 1993. Multi resolution 3D approximations for rendering complex scenes. *Geometric Modeling in Computer Graphics*, 455–465.

SHAMIR, A., AND PASCUCCI, P. 2001. Temporal and spatial level of details for dynamic meshes. In *Proceedings of the ACM symposium on Virtual Reality Software and Technology*, 77–84.

SHAMIR, A., PASCUCCI, V., AND BAJAJ, C. 2000. Multi-resolution dynamic meshes with arbitrary deformation. In *Proceedings of the 11th IEEE Visualization 2000 Conference*.

SHILANE, P., AND FUNKHOUSER, T. 2007. Distinctive regions of 3D surfaces. *ACM Transactions On Graphics 26*, 2.

TREISMAN, A. M., AND GELADE, G. 1980. A feature-integration theory of attention. *Cognitive Psychology 12*, 1 (January), 97–136.

WATSON, B., FRIEDMAN, A., AND MCGAEY, A. 2001. Measuring and predicting visual fidelity. In *Proceedings of ACM SIGGRAPH '01*.

WILLIAMS, N., LUEBKE, D., COHEN, D. J., KELLEY, M., AND SCHUBERT, B. 2003. Perceptually guided simplification of lit, textured meshes. In *Proceedings of the ACM Symposium on Interactive 3D Graphics*, 113–121.

YAMAUCHI, H., SALEEM, W., YOSHIZAWA, S., KARNI, Z., BELYAEV, A., AND SEIDEL, H. P. 2006. Towards stable and salient multi-view representation of 3D shapes. In *Proceedings of the IEEE Int. Conf. on Shape Modeling and Applications*.

YEE, H., PATTANAIK, S., AND GREENBERG, D. P. 2001. Spatiotemporal sensitivity and visual attention for efficient rendering of dynamic environments. *ACM Transactions on Graphics 20*, 1, 39–65.