International Journal of Computer Applications (0975 - 8887)
Volume 162 - No.11, March 2017

Surface Decimation by Scene Contents

Pascual Castelló
Escola d'Art i Superior de Disseny de Castelló (EASD Castelló)
Plaça de Fadrell, 1
E-12002 Castelló de la Plana, Spain

ABSTRACT

Today, there is a wide variety of devices ranging from PC's, game consoles, up to smartphones and tablets. These computing devices have major differences in performance and make mesh decimation still active in the field of research. One of the latest topics in the area has been to create simplification algorithms considering visual similarity. However, the full potential of most visual simplification algorithms has yet to be tapped, especially in soft real-time interactive computer simulations such as video games and virtual reality environments. In this paper, a new framework, in which occlusion and visibility are exploited intensively, is introduced in order to simplify models more accurately by taking into account their context in actual 3D scenes. Static background elements are simplified by considering the effect of their surroundings, decreasing the polygon count in the surfaces partially hidden by others. In addition, by allowing users to perform an optimal placement of the cameras in the scene, simplification in regions not seen from such viewpoints is dramatically increased. Dynamic elements, such as characters, accomplish a higher level of simplification since these elements which often consist of multiple meshes, for example, clothes, those resulting from the design stage. These meshes usually cover some regions of the base mesh and are used as occluders in order to increase the amount of polygon reduction in dynamic elements, barely losing image quality.

General Terms

Computer science, graphics

Keywords

User-assisted simplification, information-theoretic measures, viewpoint selection, occlusion

1. INTRODUCTION

Many decimation algorithms for polygonal meshes have been developed in recent years, although the first ones already appeared more than a decade ago. Nevertheless, the problem of polygonal simplification is now even more interesting because there are many types of devices with very different performances. Typical configurations include PC's, game consoles and mobile devices like smartphones, tablets, and smartwatches. Thus, still today, there is room enough for improvement. New proposals that can easily be included in applications such as video games, virtual reality and visualization are necessary. This way, these applications will be able to exploit fully the benefits of mesh simplification.

Virtual environments have been increasing the complexity of their massive worlds as well as employing a wider variety of lighting and shading techniques. Often, these interactive environments require multiple renderings of the same objects from different viewpoints in a single frame. Most common applications include rendering low-resolution versions of objects into a dynamic environment map for reflective or refractive effects; as well as shadow map rendering for multiple shadowing lights or cascading or omnidirectional shadow maps [11]). Low-resolution meshes and level-of-detail (LOD) [23] computations provide convenient optimization for collision detection.

Some important surveys on the state-of-the-art in mesh simplification can be reviewed in [5, 9, 21, 23, 25]. The latest simplification methods have focused on similar appearance [20, 36, 3, 4], human perception [22, 32, 18, 2] and *out-of-core* simplification [19, 33, 15, 29].

Other recent publications include new methods which retain physical features [16] and preserve global geometry features [34]. Kho and Garland [17] suggested the interesting idea of allowing the user to select some especially relevant regions which should be preserved. This concept was applied to QSlim, maybe one of the most remarkable methods of purely geometric simplification based on the *quadric error metric* [10].

Following the proposal by Kho and Garland [17], some other new interesting techniques are presented in this paper. Through these techniques, users can assist automatic methods in improving simplification for purely visual approaches in mesh decimation. For example, users can locate models playing the role of occluders to increase simplification in focused regions. In addition, they can select the position of the cameras to adjust simplification to the visibility of the model in an actual 3D scene. The latter can be very useful for static background geometries which are usually seen from a very narrow field of view.

There are some previous works that integrate occlusion culling and view-dependent rendering such as [1, 7, 13, 12, 35].

[1] proposed a hybrid technique which combines LOD rendering with visibility. El-Sana et al. [7] used a LOD technique that renders occluded regions in low LOD. Their work relies on the View-Dependence Tree as a compact multiresolution hierarchical data structure that supports view-dependent rendering. Gobbetti and Marton [12] integrated visibility culling and out-of-core data management with a LOD framework. At pre-processing time, a coarse volume hierarchy is generated by binary space partitioning of the input triangle soup. At rendering time, this volumetric

structure is refined and rendered in front-to-back order, exploiting vertex programs for GPU evaluation of view-dependent voxel representations, hardware occlusion queries for culling occluded subtrees and asynchronous I/O for detecting and avoiding data access latencies. Yoon et al. [35] presented a view-dependent rendering algorithm (Quick-VDR) for interactive display of massive models. A clustered hierarchy of progressive meshes (CHPM) is used as a scene representation. Quick-VDR relies on an out-of-core algorithm to compute a CHPM that performs a hierarchical cluster decomposition and simplification. Grundhofer et al. [13] combined LOD-based rendering with an efficient multi-pass occlusion culling algorithm. LODs are generated off-line as well as the slicing of the high polygon count objects. The actual rendering is done by a three-pass algorithm which identifies occluded objects during the first two passes. The third pass renders the potentially visible set.

Most of the previous approaches use a multiresolution scheme and a mechanism to detect occluded regions and adapt the LOD according to them. However, current game engines include many complex tasks such as spatial subdivision for indoor engines, some sort of continuous LOD management for outdoor engines, skeletal animation, collision detection, complex particle and decay effects, dynamic shadows, global illumination, artificial intelligence, full-screen post effects such as motion blur which all can be very time-consuming. Therefore, most of these proposals are very difficult to implement in game engines. Mainly because the rendering stage must be adapted accordingly and this can penalize the game engine performance. So far, surface simplification algorithms have traditionally been designed to work in isolation without considering how target models interact with content in true scenes. This leads to a not full profit in the quality of approximations. However, visual simplification methods can help on this issue because they can use the information of the scene in order to improve decimation in the input models. This work proposes a new framework in which to perform mesh simplification taking scene content into account.

The aim of this work is to perform the simplification off-line considering the scene content. This static approach comes with the advantage that is completely independent of the game engine or visualization algorithm. Nevertheless, this framework is less versatile than a dynamic one. But, as pointed out above, it is applicable to any visualization technique.

Figure 1 illustrates an example of how visual simplification methods can take advantage of environmental occluders. As depicted in this figure, the simplification on the root of the tree is very high, because this region of the model is covered by another scene object, a red box that hides the root and the trunk (see Figure 1(c)). Notice that the tree top remains practically unaltered. In this case, a red box was used as an environmental occluder but a fence could have been used or even a wall in a more realistic situation. Nevertheless, the aim here was simply to illustrate the effect that occluders can produce on visual simplification algorithms.

This paper is organized as follows. Section 2 briefly reviews the *viewpoint-driven simplification* method [3, 4]. In Section 3, an extension of the viewpoint-driven simplification algorithm is proposed to include occluders during the simplification stage. Section 4 demonstrates how users can improve simplification for static models in the scene by selecting the position of cameras. Section 5 shows more experimental results. Finally, in Section 6, conclusions and future work are presented.

2. VIEWPOINT-DRIVEN SIMPLIFICATION

First of all, in this section, some viewpoint measures for viewpoint selection based on Information Theory [6] are reviewed, which are the basis of the viewpoint-driven simplification algorithm. Then, a brief explanation of how the method works is presented.

2.1 Viewpoint selection measures: H_v and VMI

Viewpoint entropy, based on the definition of Shannon entropy, was introduced in [26] as a measure of the information provided by a point of view. The relative area of the polygons projected over a sphere of directions centered on the viewpoint is taken as a probability distribution. Thus, given a viewpoint v, the entropy of v (H_v) is defined by

$$H_v = -\sum_{i=0}^{N_f} \frac{a_i}{a_t} \log \frac{a_i}{a_t}, \tag{1}$$

where N_f is the number of polygons in the scene, a_i is the projected area of polygon i over the sphere, a_0 represents the projected area of the background in open scenes, and $a_t = \sum_{i=0}^{N_f} a_i$ is the total area of the sphere. Maximum entropy is obtained when a certain viewpoint can see all the polygons with the same projected area. So, in an open scene the maximum entropy is $\log(N_f + 1)$ and in a closed scene is equal to $\log N_f$. The best viewpoint is defined as the one that has maximum entropy, that is, the maximum amount of captured information. In molecular visualization, both maximum and minimum entropy views show relevant characteristics of a molecule [27].

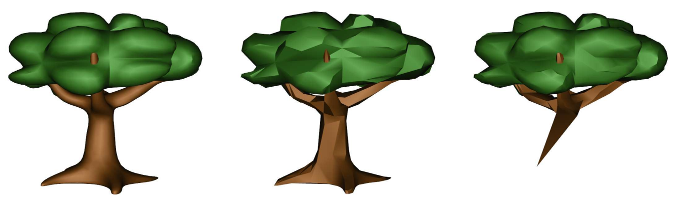
Viewpoint mutual information was introduced in [28, 8] as a measure for viewpoint selection. An *information channel* $V \to O$, called a *viewpoint information channel*, is defined between the random variables V and O. This channel represents, respectively, a set of viewpoints and the set of polygons for the given object. Viewpoints are indexed by v and polygons by o. The marginal probability distribution for V is given by $p(v) = \frac{1}{N_v}$, where N_v is the number of viewpoints. That is, the same probability is assigned to each viewpoint, but other distributions can also be used. The conditional probability $p(o|v) = \frac{a_o}{a_t}$ is defined by the normalized projected area of polygon o over the sphere of directions centered at viewpoint v. a_o is the area of polygon o projected over the sphere and a_t is the total area of the sphere. Conditional probabilities fulfill $\sum_{o \in \mathcal{O}} p(o|v) = 1$. Note that with this notation viewpoint entropy (1) can be rewritten as $H_v = -\sum_{o \in \mathcal{O}} p(o|v) \log p(o|v)$. Finally, the marginal probability distribution of O is given by $p(o) = \sum_{v \in \mathcal{V}} p(v)p(o|v) = \frac{1}{N_v} \sum_{v \in \mathcal{V}} p(o|v)$, which represents the average projected area of polygon o, i.e., the probability of a polygon o to be hit (seen) by a random ray cast from the sphere of viewpoints.

From this channel, the *mutual information* between V and O is given by

$$\begin{aligned} I(V,O) &= \sum_{v \in \mathcal{V}} p(v) \sum_{o \in \mathcal{O}} p(o|v) \log \frac{p(o|v)}{p(o)} \\ &= \frac{1}{N_v} \sum_{v \in \mathcal{V}} I(v,O), \end{aligned} \tag{2}$$

where

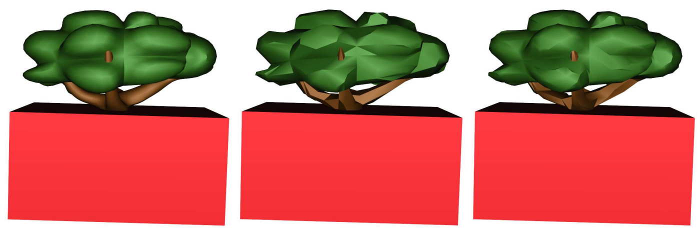
$$I(v,O) = \sum_{o \in \mathcal{O}} p(o|v) \log \frac{p(o|v)}{p(o)}, \tag{3}$$



(a) Simpletree Model, 11 136 triangles

(b) Simplified without environmental occluders, 1 000 triangles

(c) Simplified with environmental occluders, 1 000 triangles



(d) Original with environmental occluders (red box)

(e) Simplified without environmental occluders (red box)

(f) Simplified with environmental occluders (red box)

Fig. 1. An example of mesh simplification performed by a visibility-based simplification method using a red cube as occluder. The model complexity was reduced to about 8.97% of the original one.

called *viewpoint mutual information* (VMI), represents the degree of *dependence* or *correlation* between viewpoint v and the set of polygons O. VMI is a measure of the *quality* for viewpoint v. *Quality* is considered here equivalent to *representativeness*. The best viewpoint is defined as the one that has *minimum* VMI. High values mean a high degree of dependence between viewpoint v and the object, indicating a very *coupled* view. On the other hand, low values correspond to the most *representative* or *relevant* views, which show the maximum possible number of polygons in a balanced way.
Note that $I(v, O) = KL(p(O|v)|p(O))$, where $p(O|v)$ is the conditional probability distribution between v and the object and $p(O)$ is the marginal probability distribution of O, which in this case corresponds to the distribution of the average of projected areas. It is worth observing that $p(O)$ plays the role of the *target* distribution in the Kullback-Leibler (KL) distance or relative entropy [6] and also the role of the *optimal* distribution since the objective is that $p(O|v)$ becomes similar to $p(O)$ to obtain the best views. On the other hand, this role agrees with intuition, since $p(O)$ is the average visibility of polygon o on all viewpoints, i.e., the *mixed distribution* of all views, and $p(O)$ is representing, with a single distribution, the knowledge about the whole scene.

2.2 Simplification metric

Viewpoint-driven simplification [3, 4] is a decimation algorithm based on the half-edge collapse operation [14]. In this operation, the two vertices of an edge are contracted to a single vertex, removing the collapsed edge and its incident triangles. It can also be considered as the vertex removal operator without re-triangulation (see Figure 2). The error at each edge collapse operation is measured by the variation in some viewpoint selection measure, for instance, either viewpoint mutual information (VMI) or viewpoint

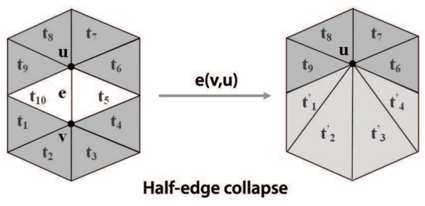


Fig. 2. The half-edge collapse operation. After the edge collapse, edge e vertex v and triangles t_{10} and t_5 are removed and triangles t_1, t_2, t_3 and t_4 are modified.

entropy (H_v). In order to guarantee a unified simplification, all viewpoints are distributed equidistantly over the sphere of viewpoints. Viewpoint selection measures express the accessible information about a given object from a particular viewpoint. The goal of this method is to preserve the visual appearance of the given model by minimizing the change in the shape of its silhouette.
Given a set of viewpoints V, the *simplification error deviation* for edge collapse e is defined by

$$C_e = \sum_{v \in \mathcal{V}} |I_v - I'_v|, \tag{4}$$

where I_v represents the value for the viewpoint selection measure before edge collapse e and I'_v represents its value afterward. Both H_v and VMI measures can be used in the scheme proposed in this paper because they produce good visual simplifications. However, the VMI metric is chosen for all the experimental results shown in this work because VMI accomplishes a better performance than H_v does. This is mainly due to the fact that H_v tends to balance the

```
// $v$ the viewpoint
// $m$ the targeted mesh
// $S$ the 3D scene
function compute_histogram($v$, $m$, $S$)
  // Black for background
  render($m$, $v$) // unique color for each polygon of $m$
  for $m' \in S$ where $m' \neq m$
    render_in_black($m'$, $v$) // black for all polygons of $m'$
  end for
  // Count pixels of the same color
  $h_v$ = count_pixels_in_framebuffer()
  return ($h_v$)
end function
```

Fig. 3. Pseudocode of the new histogram function.

polygon size in the mesh, whereas VMI drastically increases the simplification in the poorly visible regions [4], which is adequate for preserving visual appearance. Other metrics used in [3] such as the viewpoint Kullback-Leibler distance (KL_v), the viewpoint Hellinger distance (HE_v) and the viewpoint Chi-Square distance (CS_v) do not achieve better results in visual appearance compared to H_v and VMI because the former metrics (KL_v, HE_v and CS_v) are likely to retain interior regions. The reason is that they compare the actual area distribution to the projected area distribution for all polygons in the model. Consequently, the outcome is that hidden regions are further preserved in KL_v, HE_v, and CS_v than in measures such as H_v and VMI that only consider the projected area distribution [3].

3. USING MESHES AS OCCLUDERS

The algorithm of viewpoint-driven simplification [3, 4], briefly reviewed in the previous section, originally uses a regular distribution of surrounding viewpoints for the mesh to be simplified. However, the default behavior of the algorithm can be modified if another different set of viewpoints is used. In fact, this simplification method actually captures the information about the surface of the input mesh that is seen from a given point of view. That is, if a surface is hidden then it will be simplified more aggressively. Therefore, this useful feature can be exploited in an actual scene where the targeted mesh might interact with others nearby. The current implementation of the viewpoint-driven simplification approach does not allow the method to be executed in real-time since it has a highly computational cost. Nevertheless, for static background elements, occluders can be included in the simplification algorithm with no additional cost.

The idea is very straightforward. Before starting the simplification of the input model, all objects close to the model in the scene must be also rendered. This way, it is reproduced how the model is actually seen in the scene. The input to the viewpoint-driven simplification algorithm is now the whole geometry of the scene which obviously also includes the input model. However, the following fact must be considered. The current implementation of the simplification algorithm uses histograms to obtain the projected areas of polygons to compute the viewpoint selection measures used in the simplification error. These areas generate the probability distribution required for Equations 3 and 1. In order to find such areas, each polygon is drawn in a unique color. Black is reserved for the background. Projected areas of polygons are estimated by counting the pixels that have the same color in the frame buffer. Obviously, all the objects surrounding the targeted model can be used as occluders if they are rendered in the background color. Figure 3 shows a pseudo-code of the new histogram function modified to consider all meshes in the 3D scene as pointed out above.

Figure 4 shows an example of how the histogram technique is used to include occluders in the viewpoint-driven simplification algorithm. As seen in the figure, with the new histogram technique the occluder, that is, the Pants model (see Figure 4(d)) is rendered in black. This way, the viewpoint-driven simplification method gathers no information about those hidden regions (legs) of the Young model. Therefore, any simplification operation in these regions has virtually no cost. The outcome is that such regions are simplified at the highest level. The assignment of a distinct color for each polygon starts in the red channel of the RGB color space. This is why most of the polygons in the model are shown in red.

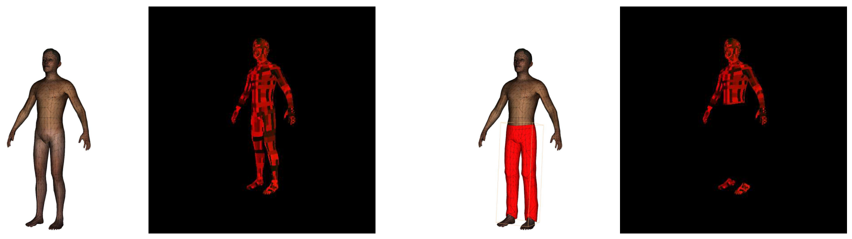
In principle, this idea can be applied to any visual simplification method. For instance, the image-driven simplification method by Lindstrom and Turk [20] can follow this approach because it also uses a set of viewpoints. In fact, in order to consider environmental occlusions, simply the whole scene content together with the input mesh must be rendered. Self-occlusions are already accounted for in the original algorithm. The algorithm introduced by Zhang and Turk [36], which computes visibility off-line and then combines the visual error with a geometric error, can also be extended by including occluders in the off-line process performed to calculate the visibility map. The same idea can also be applied to other algorithms that use mesh saliency from a purely geometric approach [18] and from a visual one [2]. Mesh saliency is usually calculated off-line and used to focus on visually important regions of the mesh, but considering how the human visual system works. This saliency, however, does not consider occlusions produced by other meshes in the scene. So, the mesh saliency map needs to be combined with a visibility map of the scene that includes such occlusions.

4. LOCATING CAMERAS SURROUNDING STATIC OBJECTS

As pointed out in Section 2.2, the original viewpoint-driven simplification algorithm employs by definition a regular distribution of cameras. These cameras are used to obtain information about the model surface. However, most static models in actual 3D scenes are only seen from very few viewpoints, this means that not all their polygons are visible, take, for example, the case of walk-throughs. Therefore, it is desirable that such objects might be mainly simplified taking their visibility into account in the world. Better results can be achieved if simplification is carried out considering only the set of viewpoints from where models are shown in the scene. Users play a remarkable role here because they can conveniently select the optimal set of camera positions.

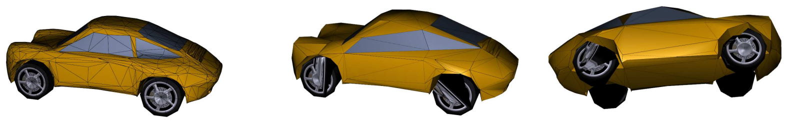
This concept can also be applied to other visual simplification methods that make use of camera positions such as [20, 36].

Figure 5 shows an example of camera placement performed by a user. The user has located 6 cameras from where the Porsche 911 model is shown in an actual scene. When rendered, the Porsche 911 model stays parked on the street next to a building. Notice that this model is a static object only visible from a very limited range of viewpoints. Image 5(c) depicts the 6 viewpoints established by a user, which are some possible positions from where the model is seen. The left image 5(d) shows the model simplified from a camera position within the field of view that defines the set of viewpoints used to simplify the model. As can be appreciated, the final simplified model looks almost identical to the original. The

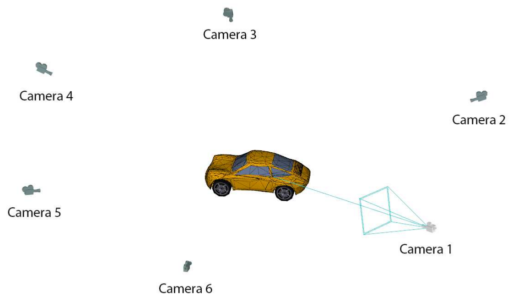


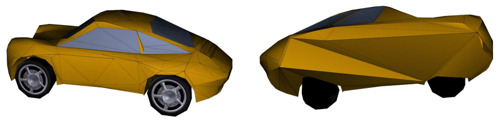
(a) Young model, 6364 triangles (b) Rendering from viewpoint #1 for (a) (c) Young model using the Pants model in red as occluder (d) Rendering from viewpoint #1 for (c)

Fig. 4. An example of how the histogram calculation technique works in order to make a profit out of occluders during the simplification stage. The rendering images were used to compute the projected area of polygons in the model.



(a) Porsche 911 model (b) VMI with 6 uniformly distributed cameras. RMSE=21.59, MS-SSIM*=0.825





(c) The optimal set of 6 cameras surrounding the model selected by a user (d) VMI replacing the original set of cameras with (c). RMSE=15.23, MS-SSIM*=0.882

Fig. 5. An example of optimal camera placement done by a user for the Porsche 911 model (1963 triangles). The model was simplified to 499 triangles. (b) and (d) show the model from a view similar to one of those chosen for simplification. The right images show an opposite view. (d) shows an improvement about 41.75% in visual appearance with respect to (b). Notice that the view corresponding to (b) is one of the views from where the model will be shown in the scene.

right image 5(d) shows the same simplified model from an opposite perspective. As can be seen, the wheels have disappeared because they cannot be seen from the set of viewpoints selected by the user. This is the desirable behavior because the visible parts of the model are better preserved which is the final goal. Image 5(b) shows the model simplified with 6 viewpoints uniformly distributed. Some artifacts have appeared in the wheels because such small number of viewpoints is not enough for the original VMI-based method.

In order to measure the visual error for the experimental results, several image quality assessment metrics: RMSE, SSIM [30], MS-SSIM [31] and MS-SSIM* [24] were used. RMSE stands for the root mean square error, a per-pixel-difference image metric. Based on the idea that the human visual system (HVS) is highly adapted to process structural information, SSIM (Structural SIMilarity) measures the change in luminance, contrast, and structure in an image. The values suggested by the authors were applied to the parameters $K_1 = 0.01$ and $K_2 = 0.03$. Layered on SSIM, MS-SSIM (Multi-Scale Structural SIMilarity) calculates multiple SSIM values at multiple image scales evaluating the quality of the image at different viewing distances. For MS-SSIM was used an 11×11 circular Gaussian sliding window and five scales. MS-SSIM* is an extension of the original MS-SSIM that does away with the stabilization constants and defines concrete values for the edge cases. For MS-SSIM* was used an 8×8 linear window. SSIM, MS-SSIM, and MS-SSIM* range from 0 to 1 and give 1 when the images are equal.

5. RESULTS

All models shown in this paper were simplified on a 21.5-inch iMac computer equipped with an Intel Core 2 Duo 3.06GHz with 4GB DDR3 RAM and an NVIDIA GeForce 9400 256MB graphics card running OS X 10.10.5. To compute the VMI

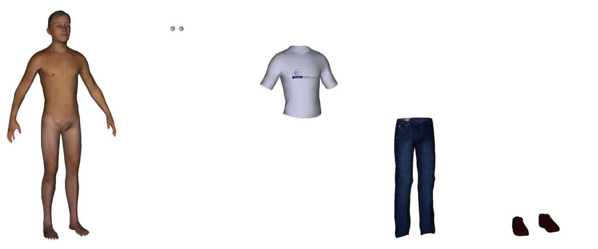

(a) Young model (b) Eyes model (c) Shirt model (d) Pants model (e) Sneakers model

Fig. 6. The different meshes that compose the dressed Young model.

measures, the values suggested by the authors [3, 4] were used, i. e., a 256x256 image resolution and 20 uniformly distributed viewpoints which correspond directly to the vertices of a regular dodecahedron, a Platonic solid. VMI* stands for the original VMI modified to implement the new histogram technique that takes into consideration the occlusion due to other objects in the scene. This section analyzes more deeply the experimental results for the dressed Young model shown in Figure 6. The test model consists of several meshes, a base mesh called Young and other meshes, Shirt, and Pants, which represent some clothes. There are some additional meshes for the eyes and shoes, the Sneakers model.
Figure 7 shows an example of simplification for a dynamic model. Typically, when rendering the Young model in a video game, the model usually wears different types of clothes such as coats, shirts, sweaters, pants, and hats. These clothes are often switched by players at run-time. However, some parts of the body of the model covered with clothes might be hidden most of the time. Image 7(b) shows the Young model simplified with VMI wearing all the different meshes (Pants, Shirt, Eyes, and Sneakers) as commonly seen in a video game scenario. Image 7(c) shows the Young model simplified with VMI*. As can be observed in Figures 7(g) and 7(h), VMI* achieves a reduction about 72% in mesh complexity without hardly losing quality. This is, mostly accomplished, due to the drastic simplification in the hidden regions (trunk and legs) of the Young model (see Figure 6(a)). The other four meshes were also simplified with VMI in order to reduce the total number of polygons in the whole model.
Table 1 depicts the computational time, number of points of view, and percentage of simplification for each mesh. As can be observed in the table, the approach presented in this paper allows for a reduction about 79% in the polygon count without decreasing the quality of the original model's appearance (see Figure 7(a)).
Finally, Figure 8 depicts an example of a 3D scene where all models tested in this paper are rendered. All these models were simplified using VMI*. Furthermore, all of the requirements and conditions are satisfied in this sample scene. The Simpletree model has both the roots and partially the trunk hidden by a wall. The Porsche 911 model is located beside the wall where two of its wheels are hidden. The Young model appears wearing all the additional meshes such as Pants, Shirt, and Sneakers. The total polygon count for the original version of the models is 22 737 whereas those simplified versions actually shown in the scene have an overall number of 4 225 polygons. Note that a reduction about 81% in the polygonal complexity of those models is accomplished without significantly decreasing their visual appearance as this figure shows.

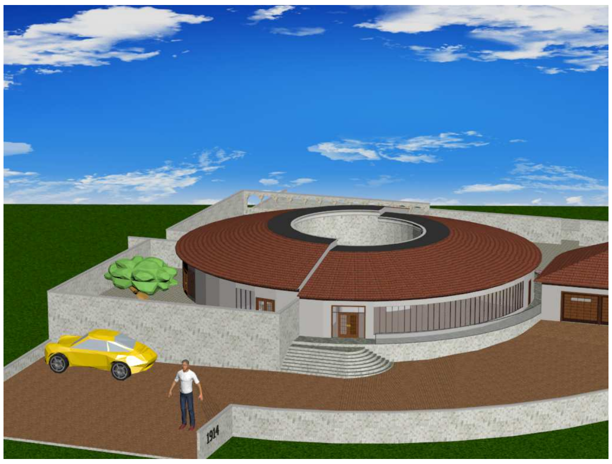


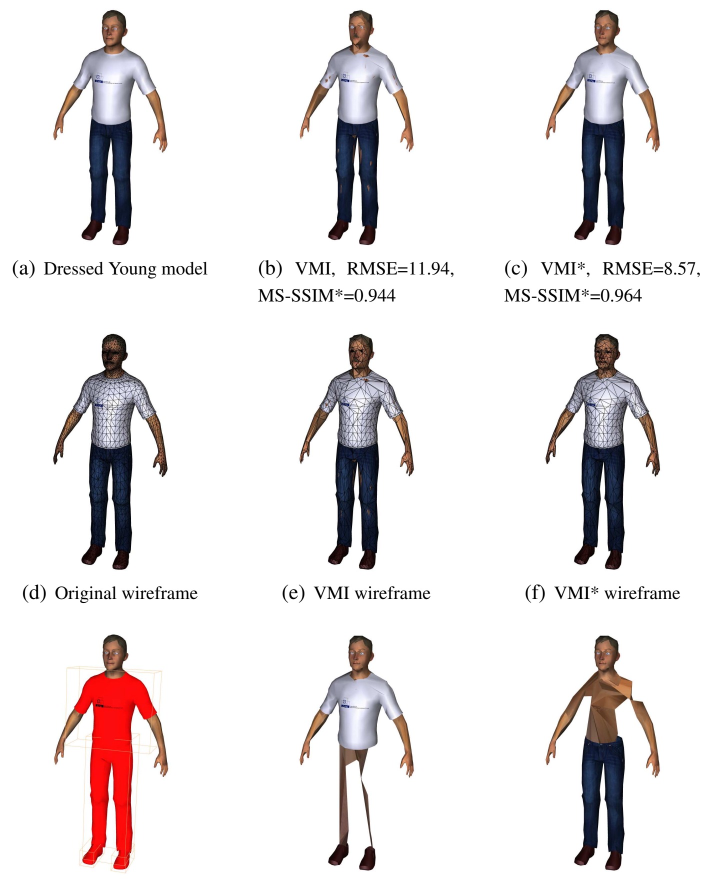
Fig. 8. An example of a 3D scene. In this scene, the Simpletree, Porsche 911, and dressed Young models appear simplified with VMI*.

6. CONCLUSIONS AND FUTURE WORK

Visual simplification algorithms are, by far, more suitable for visualization applications than the geometric ones, because the former can benefit from visibility information. Up to now, visual simplification algorithms have been used to simplify models in isolation without rendering them in actual scenes.
So as to simplify models more efficiently in actual scenes, the framework proposed here exploits some unique characteristics of visual simplification algorithms such as occlusion and viewpoint placement. As shown in this paper, simplification of static models seen from a very reduced range of points of view is enhanced if users select the optimal camera positions. Moreover, by using occluders, better results in simplification of base models are easily achieved because regions with no major visual impact are now simplified more aggressively. An example was shown in the case of characters. But, of course, it can also be applied to a wide variety of meshes of similar features.
This new approach represents an off-line processing. Rendering the simplified versions in a graphics application comes with no additional cost, of course, once they are obtained. This makes it appropriate for video games since most game engines might involve some high degree of computational overload. Furthermore, no adaptation of the game engine is needed.
In this work, the techniques mentioned above were applied to a particular, purely visual simplification approach, the *viewpoint-driven simplification* method [3, 4]. However, such techniques can also be applied to other visual simplification algorithms [20, 36, 18, 2] as previously suggested in this paper.
The idea proposed by Kho and Garland [17] of permitting the user to select some surfaces on the mesh, independently of their simplification error, in order to retain them better because of their interest, can also be combined with the techniques presented in this paper for visual simplification algorithms. Another further extension to this work would be to apply the framework to other visual simplification algorithms and to compare the results.

7. REFERENCES

[1] C. Andújar, C. Saona-Vázquez, I. Navazo, and P. Brunet. Integrating occlusion culling and levels of detail



(a) Dressed Young model (b) VMI, RMSE=11.94, MS-SSIM*=0.944 (c) VMI*, RMSE=8.57, MS-SSIM*=0.964

(d) Original wireframe (e) VMI wireframe (f) VMI* wireframe

(g) Original with occluders in red (h) VMI* no Pants (i) VMI* no Shirt

Fig. 7. An example of simplification for the dressed Young model (9638 triangles) wearing the Pants, Shirt, and Sneakers models. The dressed Young model was simplified to 2726 triangles. The Pants, Shirt, and Sneakers models were used as occluders (see (g)). Some artifacts are seen in (b) when simplifying with VMI. Since VMI cannot account for external occlusions because models are simplified separately. Thus, intersections among models can occur. (h) and (i) show how VMI* drastically simplifies hidden interiors.

Table 1. Results of all models, simplification time in seconds. C is the number of cameras. The Pants, Shirt, and Sneakers models were used as occluders for the Young model alone.The Pants, Shirt, Sneakers and Eyes models were simplified using the original VMI. The dressed Young model (see Figure 7(a)) was simplified with VMI*.

Model	**C**	**Triangles**		**Simplif.**	**Time**	**Visual error**			
		Orig.	**Final**	**%**	**s**	**RMSE**	**SSIM**	**MS-SSIM**	**MS-SSIM***
Dressed Young	20	9638	999	10.36	69.86	8.57	0.994	0.991	0.964
Pants	20	1088	499	45.86	8.59	4.40	0.997	0.997	0.974
Shirt	20	1040	700	67.30	6.01	1.43	0.999	0.999	0.994
Sneakers	20	652	328	50.30	5.71	8.33	0.985	0.988	0.937
Eyes	20	224	200	89.28	0.89	0.07	0.999	0.999	0.992

through hardly-visible sets. *Computer Graphics Forum*, 19(3):499–506, 2000.

[2] P. Castelló, M. Chover, M. Sbert, and M. Feixas. Reducing complexity in polygonal meshes with view-based saliency. *Computer Aided Geometric Design*, 31(6):279–293, 2014.

[3] P. Castelló, M. Sbert, M. Chover, and M. Feixas. Viewpoint-based simplification using f-divergences. *Information Sciences*, 178(11):2375–2388, 2008.

[4] P. Castelló, M. Sbert, M. Chover, and M. Feixas. Viewpoint-driven simplification using mutual information. *Computer Graphics*, 32(4):451–463, 2008.

[5] P. Cignoni, C. Montani, and R. Scopigno. A comparison of mesh simplification algorithms. *Computer Graphics*, 22(1):37–54, 1998.

[6] T. M. Cover and J. A. Thomas. *Elements of information theory*. Wiley-Interscience, NY, USA, 1991.

[7] J. El-Sana, N. Sokolovsky, and T. C. Silva. Integrating occlusion culling with view-dependent rendering. In *Proc. of the conference on Visualization '01 (VIS '01)*, pages 371–378, Washington DC, USA, 2001. IEEE Computer Society.

[8] M. Feixas, M. Sbert, and F. González. A unified information-theoretic framework for viewpoint selection and mesh saliency. *ACM Transactions on Applied Perception*, 6(1):1–23, 2009.

[9] M. Garland. Multiresolution modeling: Survey & future opportunities. In *State of the Art Reports of EUROGRAPHICS '99*, pages 111–131, 1999.

[10] M. Garland and P. S. Heckbert. Surface simplification using quadric error metrics. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 209–216. Addison-Wesley Publishing Co, ACM Press, 1997.

[11] P. Gerasimov. Omnidirectional shadow mapping. In R. Fernando, editor, *GPU Gems: Programming Techniques, Tips, and Tricks for Real-Time Graphics*, volume 20, chapter 12, pages 193–204. Addison-Wesley, 2004.

[12] E. Gobbetti and F. Marton. Far voxels: a multiresolution framework for interactive rendering of huge complex 3d models on commodity graphics platforms. *ACM Transactions on Graphics*, 24(3):878–885, 2005.

[13] A. Grundhöfer, B. Brombach, R. Scheibe, and B. Fröhlich. Level of detail based occlusion culling for dynamic scenes. In *Proc. of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia (GRAPHITE '05)*, pages 37–45, NY, USA, 2005. ACM Press.

[14] H. Hoppe, T. Derose, T. Duchamp, J. Mcdonald, and W. Stuetzle. Mesh optimization. In *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 19–26, 1993.

[15] M. Isenburg, P. Lindstrom, S. Gumhold, and J. Snoeyink. Large mesh simplification using processing sequences. In *VIS '03: Proceedings of the 14th IEEE Visualization*, pages 465–472, Washington DC, USA, 2003. IEEE Computer Society.

[16] B. S. Jong, J. L. Tseng, and W. H. Yang. An efficient and low-error mesh simplification method based on torsion detection. *Visual Computer*, 22(1):56–67, 2006.

[17] Y. Kho and M. Garland. User-guided simplification. In *SI3D '03: Proceedings of the symposium on Interactive 3D graphics*, pages 123–126, NY, USA, 2003. ACM Press.

[18] C. H. Lee, A. Varshney, and D. W. Jacobs. Mesh saliency. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*, pages 659–666, NY, USA, 2005. ACM Press.

[19] P. Lindstrom. Out-of-core simplification of large polygonal models. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 259–262. Addison-Wesley Publishing Co, ACM Press, 2000.

[20] P. Lindstrom and G. Turk. Image-driven simplification. *ACM Transactions on Graphics*, 19(3):204–241, 2000.

[21] D. Luebke. A developer's survey of polygonal simplification algorithms. *IEEE Computer Graphics and Applications*, 21(3):24–35, 2001.

[22] D. Luebke and B. Hallen. Perceptually-driven simplification for interactive rendering. In *Proc. of the 12th EUROGRAPHICS Workshop on Rendering Techniques*, pages 223–234, London, UK, 2001. Springer-Verlag.

[23] D. Luebke, M. Reddy, J. D. Cohen, A. Varshney, B. Watson, and R. Huebner. *Level of detail for 3d graphics*. Elsevier Science, San Francisco, USA, 2003.

[24] D. M. Rouse and S. S. Hemami. Analyzing the role of visual structure in the recognition of natural image content with multi-scale ssim. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 6806, page 38, 2008.

[25] O. M. van Kaick and H. Pedrini. A comparative evaluation of metrics for fast mesh simplification. *Computer Graphics Forum*, 25(14):197–210, 2006.

[26] P. P. Vázquez, M. Feixas, M. Sbert, and W. Heidrich. Viewpoint selection using viewpoint entropy. In *VMV '01: Proceedings of the Vision Modeling and Visualization Conference*, pages 273–280. Aka GmbH, 2001.

[27] P. P. Vázquez, M. Feixas, M. Sbert, and A. Llobet. Realtime automatic selection of good molecular views. *Computer Graphics*, 30(1):98–110, 2006.

[28] I. Viola, M. Feixas, M. Sbert, and M. E. Gröller. Importance-driven focus of attention. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):933–940, 2006.

[29] H. T. Vo and S. P. Callahan. Streaming simplification of tetrahedral meshes. *IEEE Transactions on Visualization and Computer Graphics*, 13(1):145–155, 2007.

[30] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[31] Z. Wang, E. P. Simoncelli, and A. C. Bovik. Multi-scale structural similarity for image quality assessment. In *Proc. of IEEE Asilomar Conference on Signals, Systems and Computers*, volume 2, pages 1398–1402, 2004.

[32] N. Williams, D. Luebke, J. D. Cohen, M. Kelley, and B. Schubert. Perceptually guided simplification of lit, textured meshes. In *SI3D '03: Proceedings of the symposium on Interactive 3D graphics*, pages 113–121, NY, USA, 2003. ACM Press.

[33] J. Wu and L. Kobbelt. A stream algorithm for the decimation of massive meshes. In *Proc. of Graphics Interface*, pages 185–192, 2003.

[34] Y. Wu, Y. He, and H. Cai. Qem-based mesh simplification with global geometry features preserved. In *GRAPHITE '04: Proceedings of the 2nd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, pages 50–57, NY, USA, 2004. ACM Press.

[35] S. Yoon, B. Salomon, R. Gayle, and D. Manocha. Quick-vdr: Interactive view-dependent rendering of massive models. In *Proc. of the conference on Visualization '04 (VIS '04)*, pages 131–138, Washington DC, USA, 2004. IEEE Computer Society.

[36] E. Zhang and G. Turk. Visibility-guided simplification. In *VIS '02: Proceedings of the conference on Visualization*, pages 267–274, Washington DC, USA, 2002. IEEE Computer Society.