A Real-Time Visual Attention Model for Predicting Gaze Point During First-Person Exploration of Virtual Environments

Sebastien Hillaire* Orange Labs / INRIA
Anatole Lecuyer† INRIA
Tony Regia-Corte‡ INRIA
Remi Cozot§ INRIA / Univ. of Rennes 1
Gaspard Breton¶ Orange Labs

Abstract

This paper introduces a novel visual attention model to compute user's gaze position automatically, i.e. without using a gaze-tracking system. Our model is specifically designed for real-time first-person exploration of 3D virtual environments. It is the first model adapted to this context which can compute, in real-time, a continuous gaze point position instead of a set of 3D objects potentially observed by the user. To do so, contrary to previous models which use a mesh-based representation of visual objects, we introduce a representation based on surface-elements. Our model also simulates visual reflexes and the cognitive process which takes place in the brain such as the gaze behavior associated to first-person navigation in the virtual environment. Our visual attention model combines the bottom-up and top-down components to compute a continuous gaze point position on screen that hopefully matches the user's one. We have conducted an experiment to study and compare the performance of our method with a state-of-the-art approach. Our results are found significantly better with more than 100% of accuracy gained. This suggests that computing in real-time a gaze point in a 3D virtual environment is possible and is a valid approach as compared to object-based approaches.

CR Categories: I.2.0 [Artificial Intelligence]: User/Machine Systems—human factors; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Virtual Reality; H.5.2 [Information Interfaces and Presentation]: User Interfaces—Input devices and strategies, Interaction styles, User-centred design

Keywords: visual attention model, first person exploration, gaze tracking

1 Introduction

The gaze point is the point a user is looking at. In Virtual Environments (VE), knowing the gaze point position can give developers several advantages to efficiently display a high quality virtual scene. Applications can take advantage of this feature to better distribute available computational resources to efficiently render a virtual scene or to simulate natural effects occurring in human vision improving users' perception of the VE. For instance, Luebke *et al.* [Luebke and Hallen 2001] proposed a method to accelerate the rendering process of a VE by progressively decimating meshes based on their distance to the gaze point. Hillaire *et al.* [Hillaire et al. 2008a] proposed two visual effects to increase users' immersion and perception in VE based on the gaze point: a depth-of-field blur effect and a compensated camera motion both adapted to user's gaze point in real-time.

A straightforward way to compute user's gaze point position on a screen is to use a gaze tracking system [Glenstrup and Engell-Nielsen 1995]. Since their creation in the late 19th century, before the computer existed, these systems have been considerably enhanced [Glenstrup and Engell-Nielsen 1995]. The interest in these systems has grown thanks to their usefulness in several domains: human studies in psychology to VR systems, aid for people with disabilities or graphic rendering. However, accurate gaze tracking systems are still expensive and can only be accessed by a limited number of researchers or companies.

Another way to compute the gaze point is to use a visual attention model simulating human attention. Many researches have been dedicated to the evaluation of human attention when looking at pictures [Itti et al. 1998] or video [Itti 2005] and its simulation using visual attention models. However, these models have not yet been studied nor adapted to the context of first person exploration of VE. To the best of our knowledge, only the model proposed by Lee et al. [Lee et al. 2009] is adapted to this context but it can only compute attention per objects.

In this paper, we propose a novel visual attention model adapted to the context of real-time first-person exploration of VE. It is composed of new, and improved, components and data representations. Our main contributions are:

- A novel visual attention model specifically designed for real-time exploration of 3D VE which can compute a continuous 2D gaze point position;
- A novel representation of visual objects based on surface-elements (*surfels*). We will show that this representation has several advantages over the mesh-based representation for visual attention models;
- The use of a novel component simulating the gaze behavior of users walking and turning in VE [Hillaire et al. 2009], implemented for the first time in a real-time visual attention model.
- An extended evaluation of our model as compared to a state-of-the-art approach in various 3D VE.

In the remainder of this paper, we will describe related work concerning human visual attention and visual attention models. Then, we will detail the novel visual attention model we propose. Finally, we will report on an experiment conducted to evaluate the efficiency of our model as compared to a state-of-the-art model. Last, we will discuss efficiency and usability of the proposed model in VR applications.

2 Related Work

Visual attention represents the capacity of a human to focus on a visual object. It is well known that human visual attention is composed of two components [Itti 2005]: bottom-up and top-down components.

*e-mail: shillair@irisa.fr
†e-mail:anatole.lecuyer@irisa.fr
‡e-mail:tregiaco@irisa.fr
§e-mail:remi.cozot@irisa.fr
¶e-mail:gaspard.breton@orange-ftgroup.com

The bottom-up component represents the visual reflexes of the human visual system. Due to the structure of our brain and the fact that we only accurately perceive the environment within 2 degrees of the visual field [Cater et al. 2003], the human visual system does not have the capabilities to analyze a whole scene in parallel. Actually, the human visual system can only detect primitive features in parallel, defining salient areas in the visual field. Then, it uses a sequential visual search to quickly analyze the scene [Treisman and Gelade 1980]. For example, when someone first looks at a scene, his/her gaze is first unconsciously attracted by visually salient areas to rapidly perceive the most important parts of the scene [Itti et al. 1998]. Several visually salient features have been identified in previous researches [Treisman and Gelade 1980][Itti et al. 1998]: red/green and blue/yellow antagonistic colors, intensities, orientations, etc. Inspired by the feature integration theory [Treisman and Gelade 1980], bottom-up visual attention models have been developed to compute a saliency map from an image [Itti et al. 1998]. The saliency value of each pixel of the saliency map represents its attractiveness, i.e. the higher saliency of an area, the more a human is likely to look at this area. Other features have been progressively added in the computation of saliency maps such as flickering [Itti 2005], depth [Lee et al. 2009] or motion [Itti 2005].

Moreover, visual attention is not only controlled by reflexes resulting from visual stimuli, but also by the cognitive process that takes place in the brain, i.e. the top-down component. It is involved in the strategies we use to analyze a scene. For example, Yarbus [Yarbus 1967] has shown that the way people look at pictures strongly depends on the task they have to achieve. Furthermore, the top-down component is subject to the habituation phenomenon [Longhurst et al. 2006], i.e. objects become familiar over time, and we become oblivious to them. Several models have been proposed to simulate the multiple top-down components using task-map [Cater et al. 2003], habituation [Longhurst et al. 2006], memory [Navalpakkam and Itti 2005] as well as spatio-temporal contexts [Lee et al. 2009].

Bottom-up only models[Itti et al. 1998][Itti 2005] have been reported as good predictors as they were able to predict a non-negligible fraction of human gaze targets, i.e. there is a strong correlation between gaze positions and salient areas in a saliency map. However, in a game scenario, Sundstedt *et al.* [Sundstedt et al. 2008] have shown that a saliency map alone is not sufficient to efficiently compute user's gaze. Even without an explicit task, users will automatically assume a task by themselves. As suggested by [Itti 2005], it seems that a saliency map only suggests a set of potential gaze locations, and that another higher level component, i.e. top-down, may choose a gaze position in this set. As a result, it seems necessary for a visual attention model to simulate both bottom-up and top-down components [Sundstedt et al. 2008].

Sears *et al.* [Sears and Pylyshyn 2000] suggested that human visual attention could be based on objects in their Multiple Object Tracking (MOT) theory. Following this theory, each object has an attention priority value that is assigned in a stimulus-driven manner. A set of 3 to 5 objects are then indexed based on this value. This set of indexed objects can then be attended rapidly and before other objects in the visual field.

Surprisingly, few researches have been dedicated to the use of visual attention models for real-time attention or gaze prediction during exploration of 3D VE. To our best knowledge, only the visual attention model based on the MOT theory proposed by Lee *et al.* [Lee et al. 2009] has been specifically designed for this aim. As described in the MOT theory, it does not compute a gaze point position on the screen but returns the object, or set of objects, that could potentially receive more attention from the user. It is able to predict the object gazed by the user 48% of the time during free navigation and 62% of the time during a research task involving navigation

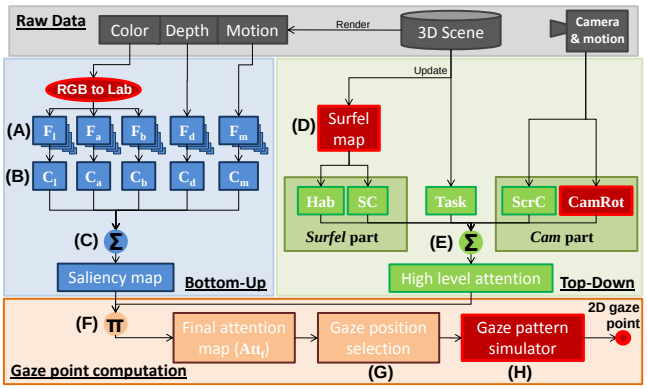

Figure 1: *Overview of our visual attention model architecture. A) feature maps, B) conspicuity maps, C) bottom-up attention (saliency map), D) update of per-surfel data, E) top-down attention, F) computation of final attention on screen, G) computation of the possible next gaze position and H) the gaze pattern simulator computing the final gaze position on screen. Red color emphasizes the novel parts of our visual attention model compared to existing techniques.*

in static and dynamic VE. However, this mesh-based discretization might be considered as a limitation as it is not possible to know precisely where the user is looking at on a particular object, e.g. is he looking at the corner or middle of a wall? This can be viewed as a coarse approximation, especially for large objects, and it is important to compute a 2D point when applying gaze-based methods such as proposed in [Luebke and Hallen 2001][Hillaire et al. 2008a]. As future work, Lee *et al.* [Lee et al. 2009] suggested to take into account the novelty of objects and other adapted top-down components. To this aim, Hillaire et al. [Hillaire et al. 2009] have studied user's gaze behavior when walking and turning in VE. They have proposed a new model taking into account first-person navigation in order to compute an attentional weight on the whole screen based on the current rotation velocity of the camera.

3 A novel visual attention model for real-time exploration of virtual environments

This section describes the complete computational visual attention model we propose. This model is able to estimate in real-time a gaze position that hopefully matches user's gaze without the need of physical devices such as web-cam or expensive gaze tracker. In the following subsections, we will describe the computation of both the bottom-up then top-down component. Finally, we will present our method to combine these two components in order to estimate a continuous 2D gaze position on the screen.

3.1 Computation of the bottom-up component

The bottom-up component of our model computes a pixel-level saliency map using several visual features: intensity [Itti et al. 1998], antagonistic colors [Itti et al. 1998], depth [Longhurst et al. 2006] and motion [Itti 2005].

3.1.1 Computation of feature maps

The starting point to compute a saliency map is to compute a feature map for each visual features (Figure 1-A):

- **Antagonistic colors and luminosity**: Originally, Itti *et al.* [Itti

et al. 1998] used red/green and blue/yellow antagonistic colors as well as intensities. In their model, antagonistic colors were computed using simple operation on RGB components. In our case, we propose to use the perceptual Lab color space which takes into account human perception [Robertson 1990]. Moreover, this color space has the advantage of directly encoding red/green and blue/yellow antagonistic colors as well as intensity, respectively the a, b and L components. They correspond to F_a, F_b and F_l feature maps in Figure 1.

- **Depth**: We propose to use a depth map as proposed in [Longhurst et al. 2006][Lee et al. 2009]. The value $F_d(p)$ for each pixel p of the depth feature map F_d is computed using Equation 1 with $z(p)$ being the linear depth of pixel p, z_{near} and z_{far} the distances of the near and far clip planes.

$$F_d(p) = \frac{z(p) - z_{near}}{z_{far} - z_{near}} \tag{1}$$

- **Motion**: Our model also takes into account visible motion on the screen. Lee *et al.* [Lee et al. 2009] proposed to approximate this feature using the motion of a single point of each visual object in world space. However this method does not take into account animated objects, e.g. an avatar moving only the hand. The motion feature $F_m(p)$ of each pixel p of the motion feature map F_m is computed using Equation 2 with $v(p)$ being the world space motion projected on the screen and t the time elapsed since last frame.

$$F_m(p) = \frac{\|v(p)\|}{t} \tag{2}$$

3.1.2 Computation of conspicuity maps

Before computing the saliency map, the feature maps need to be converted into conspicuity maps using the multi-scale Center-Surround difference operator [Itti et al. 1998] simulating the response of brain neurons which receive stimuli from the visual receptive fields. Instead of a dyadic Gaussian feature map pyramid, we use an approximation consisting of using the fast hardware mipmap pyramid generation of Graphics Processing Units (GPU) (see [Lee et al. 2009]). The conspicuity maps, i.e. C_l , C_a, C_b, C_d and C_m in Figure 1-B, are computed using Equation 3 with i and $i + j$ being mipmap pyramid levels. The level i is a fine level and $i + j$ a coarser level of the pyramid.

$$\forall x \in \{l, a, b, d, m\}\,, C_x = \frac{1}{6} \sum_{i=0}^{2} \sum_{j=3}^{4} \left| F_x^i - F_x^{i+j} \right| \tag{3}$$

Finally, the conspicuity maps are normalized using the $\mathcal{N}$ operator as described by Itti *et al.* [Itti et al. 1998] where we replace the mean of local maxima by the mean of all values in the conspicuity map for the sake of performance.

3.1.3 Computation of the final saliency map

The final saliency map can be generated using the conspicuity maps computed in the previous step (Figure 1-C). It is the result of a linear combination of each conspicuity map using Equation 4. Finally, the saliency map S is normalized in order to have its values mapped into the range $[0, 1]$.

$$S = \frac{1}{5} \times \sum_{x \in \{l,a,b,d,m\}} C_x \tag{4}$$

3.2 Computation of the top-down component

The top-down component of our model consists in simulating the cognitive processes that take place in the brain. We first propose a novel representation of visual objects based on surfel (instead of a coarse mesh-based representation) to compute spatial context [Lee et al. 2009] and habituation [Longhurst et al. 2006] components. Our model relies also on screen-space weights to take into account the observed gaze behavior of human navigating in VE using a first-person view.

3.2.1 Surfel-based representation of visual objects

Previous models for 3D exploration of VE use a representation of visual objects based on meshes [Lee et al. 2009]. This can be seen as a limitation as it is not possible to differentiate sub-parts of an object. Using this representation, it becomes indeed impossible to know if the user is gazing at the head or leg of an avatar. Moreover, large walls are problematic. Indeed, the wall can be identified as an attended object but it is not possible to differentiate between cases when the user looks at the middle or at a corner of the wall.

A possible solution could be to cut large objects in several parts. However, it is not always possible to easily modify existing assets (3D models). Also, for the sake of performance, subdividing a mesh in several sub-meshes is risky, i.e. too many triangles might impair rendering performance. Furthermore, a visual object could be embedded in the object texture, e.g. bullet holes on a wall.

In this paper, we propose to solve this issue by using a discretization of polygonal surfaces into smaller elements having the same world space size: surface elements also known as *surfel*. Surfel are wellkown and they are used as a lightmap encoding the irradiance per surface elements [Chen and Liu 2008]. Our visual attention model requires a surfel map to be built for each mesh of every potentially visible object in the VE. In our model, a surfel map virtually subdivides a mesh into several pieces (surfels) and stores them in a texture, thus not involving geometry subdivisions. A surfel is defined by its previous and current positions in world space. Concerning static meshes, the surfel map only contains the current position as visible in Figure 2. Also, for the sake of performance, the surfel map of dynamic objects is updated only when they move.

3.2.2 Generating the surfel maps

To generate the surfel map, which is in fact a texture, we first need texture coordinates for each mesh triangle that will map each triangle to its corresponding 2D position in the texture. This corresponds to unfolding triangles of the 3D meshes in the 2D map. These texture coordinates must respect two constraints: (1) overlapping triangles are forbidden and (2) a triangle must at least contain the center of a texture element (texel) of the surfel map to result in a surfel data that can be used later. 3D modeling softwares such as *Maya* or *3D Studio Max* already propose such a feature. This process is already used in several applications for the purpose of light-mapping [Chen and Liu 2008].

To fill the surfel map with data, we simply render the 3D meshes. But instead of applying a 3D projection, we project the meshes' triangles in 2D using the texture coordinates as positions. For each texel, the final 3D coordinates in world space are written in the surfel map according to each model transformation matrix (Figure 1-D). Using a surfel map avoids our model being too dependent of the geometry complexity of the scene. It just requires to update the surfel map of dynamic objects when they move. Then, all per-surfel computations are done in the surfel map, i.e. texture space. This allows to take advantage of the computational power of graphic hardware by processing all surfels in parallel.

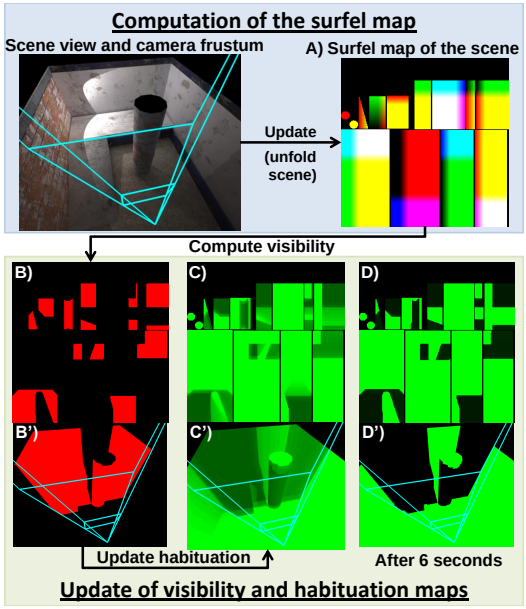

Figure 2: *Computation of the surfel map, and update of visibility and habituation maps. A) the surfel map containing world space surfel position (XYZ into RGB) B) surfel visibility (red=visible), C) surfel habituation (the greener the surfel, the less habituated the viewer is to it) and D) surfel habituation after waiting 6 seconds. B', C' and D' are views of the surfel map texture mapped on the scene.*

After preliminary testing, we have set the world space size of surfel to 20cm. Objects smaller than 20cm are virtually scaled-up. In these cases, we have visually adjusted the scale factor to have each object surfaces represented by at least one surfel. Each VE presented in Section 4 fit in a single 256×256 surfel map.

3.2.3 Computation of per-surfel components

Our visual attention model computes two attentional components per surfel (or visual object) which are habituation and spatial context:

- **Habituation**: The habituation component refers to the fact that visual objects become familiar over time [Longhurst et al. 2006]. For each surfel, we compute the habituation value for current frame using Equation 5 where t is the elapsed time in milliseconds.

$$Hab(s) = \begin{cases} Hab_{prev}(s) * \exp(-\frac{t}{h^-}) & \text{if vis(s)==true} \\ Hab_{prev}(s) + h^+ \times t & \text{otherwise} \end{cases} \quad (5)$$

When a surfel s is visible, the habituation is attenuated using an exponential decay [Longhurst et al. 2006] with interest for s going under 0.1 in 7s ($h^- = 3000$) (see [Longhurst et al. 2006] for details). When s is not visible, it is linearly regaining full interest in 20s ($h^+ = 20000$). The visibility $vis(s)$ of a surfel s is determined using shadow mapping (Figure 2-A) based on the depth buffer of the rendered view of the scene. Thanks to the surfel-based representation of visual objects, the visual attention model habituates itself to visible parts of a wall but not to the parts that are hidden by other objects (Figure 2-B).

- **Spatial context**: Lee *et al.* [Lee et al. 2009] have proposed a spatial context component to take into account the spatial behavior of the user. This component modulates the importance of each visual object based on its distance to the user. Visual objects (surfels in our case) too close or too far from the distance of interest become progressively less important. We use the same equation as proposed by Lee et al. [Lee et al. 2009] except that we have removed the condition on the fact that objects must move toward the camera. This was made in order to avoid discontinuities in the spatial context value (Equation 6-$SC_d(s)$). Since users tend to get close to objects they want to inspect [Lee et al. 2009], our spatial component also gives more importance to surfels moving toward the camera (Equation 6-$SC_{\Delta d}(s)$). For each surfel s, the final spatial context value $SC(s)$ is computed using Equation 6:

$$\begin{aligned} SC(s) &= SC_d(s) * SC_{\Delta d}(s) \\ SC_d(s) &= \frac{d(s)}{C_1} \times exp^{-\left(\frac{d(s)}{C_1}\right)^2} \\ SC_{\Delta d}(s) &= \min(C_2 \times \max(\frac{\delta d(s)}{\delta t}, 0.0), 1.0) \end{aligned} \quad (6)$$

Where $C_1 = D/0.707$ and D is the distance when surfel are considered as more important (see [Lee et al. 2009]). The other parameters are t the elapsed time since the last frame in seconds, $d(s)$ the distance of the surfel s to the camera and C_2 is a scaling constant. Afer preliminary testing, we chose $C_2 = 0.87$ (it means that an object moving toward the camera with speed equals to the walking speed of user's avatar will have the highest importance value).

3.2.4 Computation of statistical screen-space components

The gaze behavior of users exploring VE has been studied in [Hillaire et al. 2009][Hillaire et al. 2008b]. The top-down component of our visual attention model takes into account the observed gaze behavior resulting from the fact that users are navigating in the VE using a first-person view.

- **Global screen-space gaze density**: It has been shown that during a first-person view game, users tend to look more at the center of the screen [Hillaire et al. 2008b]. We propose to model this behavior as in [Lee et al. 2009] using a constant weight $ScrC(p) = \exp^{-Dist2Center(p)}$ applied on the screen where $Dist2Center(p)$ is a function giving the distance of pixel p to the center of the screen (Figure 1-E). Screen coordinates are in the range $[0, 1]$ and the middle of the screen is $(0.5, 0.5)$.

- **Gaze behavior during camera rotations**: In our model, we introduce for the first time the real-time attentional component recently proposed in [Hillaire et al. 2009]. In this paper, authors have studied user's gaze behavior when turning in VE. They have proposed a new function to compute an attentional weight on the whole screen based on the current rotation velocity of the camera. For instance, for a yaw rotation of the camera to the left, a high attention weight is set to pixels on the left of the screen. Our model introduces this function to compute an attention value $CamRot(p)$ for each pixel p on the screen.

Using these two screen-space attentional weights, our visual attention model takes into account important gaze behaviors observed during real-time first-person navigation in VE.

3.3 Final screen-space attention and gaze position computation

To compute the final 2D gaze position on the screen, the bottom-up and top-down components described previously need first to be combined in a single screen-space attention map.

3.3.1 Final screen-space attention map

Previous methods adapted to our context have proposed to compute the user's level of attention for each visual objects in the scene. To

do so, the saliency value is modulated with the top-down attention value [Lee et al. 2009]. Then, 1 to 3 objects having the highest attentional values are considered as the set of potentially gazed visual objects. However, the use of such a model might be problematic notably when the potentially gazed object is very large on the screen. In our model, we remove this constraint by computing a single continuous gaze position on the screen.

To compute the final screen-space attentional map Att_f, we first compute the top-down attention value $TD(p)$ for each pixel p on the screen using Equation 7. This is achieved by rendering visible objects from the camera's point of view. In this equation, $Task(s)$ is a value in the range [0,1] defining surfel relevance for the current task of the user, acting like a semantic weight, s is the position of the surfel in the surfel map, and p is computed using simple texture projection on meshes of $ScrD$ and $CamRot$ textures. In our case, $Task(s)$ is constant for each surfel of a single mesh to reduce memory used but it could also be stored in the surfel map.

$$TD(p) = \frac{Hab(s) + SC(s) + ScrD(p) + CamRot(p) + Task(s)}{5} \tag{7}$$

In the last step, the final attention map Att_f is computed from both the top-down and bottom-up components using Equation 8 (Figure 1-F). Finally, the gaze position is selected as the position of the pixel having the highest attention level (Figure 1-G). Then, it is sent to the gaze pattern simulator.

$$Att_f(p) = vis(s) \times S(p) \times TD(p) \tag{8}$$

3.3.2 Gaze pattern simulator

We have added a gaze pattern simulator in order to process possible gaze position changes and to smooth out the final gaze position (Figure 1-H).

In the human visual system, the duration of eye saccades is from 120ms to 300ms long depending on the rotation of the eyes [Robinson 1965], and mean fixation duration varies between 200ms and 600ms. We consider a mean frequency of eyes saccades plus fixation of 600ms. Thus, in order to smooth the final gaze position, we low pass filter the input gaze position using a cut-off frequency of 1.67Hz. This low pass filter allows the simulation of the smooth pursuit phenomenon [Robinson 1965], occurring when eyes are following a smoothly moving visual object, while allowing fast gaze jumps simulating saccades.

3.4 Implementation details and performance

To sum up, our visual attention model combines both bottom-up and top-down attention components into a single attention map. Using this attention map, it finally computes a continuous 2D gaze position which is filtered by the gaze pattern simulator.

Our visual attention model is implemented using OpenGL and GLSL. We have developed our own exporter from *Maya* which automatically generates the surfel map texture coordinates. The VE are rendered using dynamically shadowed point and spot lights together with global illumination baked in a lightmap using *Mental Ray* software. The renderer also features HDR rendering with simple luminance adaptation.

During the computation of the *bottom-up* saliency map, the feature and conspicuity maps have all the same resolution of 256×256. Our normalization operator $\mathcal{N}$, simplified as compared to [Itti et al. 1998], needs parameters such as the maximum and mean values contained in the conspicuity maps. To compute these parameters, we do not iteratively read the entire map using the CPU as this would be too expensive. Instead, we compute the maximum and mean by recursively down-sampling the textures by a factor of two until we reach the size of one texel which contains the final desired values. In this algorithm, at each step, and for each pixel of the coarser level, a fragment program computes the maximum and mean values of the four corresponding pixels sampled from the texture computed in the previous step.

Components	Feature maps	Conspicuity maps	Saliency map	Per-surfel components	Total
Performance	0.14ms	0.13ms	0.18ms	0.45ms	0.92ms

Table 1: *Computation time in milliseconds for each step of our visual attention model. (*Per-surfel components *refer to Figure 1* surfel part*)*

Updating the surfel maps first requires the copy of texture containing current surfel positions into the texture containing old surfel positions. Then, the current surfel texture is updated only for objects that have moved since last frame. Furthermore, we update the surfel texture only every 100ms in order to increase overall performance. To update a surfel map, we must bind it as the current render target and this is a costly operation. To avoid multiple render target switches, the surfel maps of dynamic objects are packed in a large texture atlas.

The final step of our visual attention model consists in computing the final gaze position on the screen as the pixel having the highest attentional value. For this aim, the $Att_f(p)$ is a 3-channel texture which stores the final attention level in the *red* component and pixel positions in the *green* and *blue* components. We then use the same recursive down-sampling method, as to compute the normalisation operator parameters, but we keep the coordinates of the pixel having the highest attentional value.

The visual attention model we propose needs several input parameters: linear depth, screen space motion and surfel texture coordinate. This could be considered as computationally too expensive. However these raw data are already computed by many existing 3D game engines to add visual effects such as depth-of-field or motion blur as well as atmosphere light scattering. Thus, adding our model to an existing engine should not require too many additional resources.

Our visual attention model can run in real-time thanks to the graphic hardware. On a laptop PC (Intel Core 2 2.5Ghz, nVidia GeForce3700M, 4Gb of RAM), the virtual scene VE_1 (see Section 4.2) is rendered at 145 frames-per-second (FPS) with the visual attention model running, as compared to 170FPS without. Detailed GPU computation times are given in Table 1. The low computation time of our visual attention model would allow it to be used in several real-time 3D applications and games.

4 Experimental evaluation

We have conducted an experiment to evaluate the performance of our visual attention model and compare it to the state-of-the-art model of Lee *et al.* [Lee et al. 2009]. To the authors best knowledge, this is the only model adapted to real-time 3D exploration of VE proposed so far.
Twelve naïve participants (10 males, 2 females) with a mean age of 31.8 (SD=6.4) participated in our experiment. They were all familiar with the first-person navigation paradigm and had normal

vision.

4.1 Experimental apparatus

During this experiment, we used a Tobii x50 gaze tracker to compute participants' gaze position. This gaze position is considered as the ground truth. Participants were positioned in front of a 19" flat screen at a resolution of 1280×1024. The screen was 37.5cm width. They were at a distance of 60cm from the screen and no sound was played. The VEs were rendered in real-time with a constant refresh-rate of 75Hz.

The navigation in the VE was achieved using first-person viewing mode. In this case, the virtual camera is positioned at the level of the eyes of user's avatar. We allowed three degrees of freedom of displacement: walking forward/backward and changing the horizontal and vertical camera orientation, i.e. yaw and pitch angles. Walking forward or backward was achieved using the up and down arrow keys of the keyboard. Changing camera's orientation (yaw and pitch angles) was achieved using movements of the mouse. Mouse correction and filtering were disabled. A horizontal mouse movement of 2.5cm on the table resulted in a rotation of 90 degrees of the camera in the VE (36 degrees/cm). Avatar properties were inspired by real data: height was 1.75m and walking speed was 1.15 m/s [Hillaire et al. 2009].

4.2 Procedure

For each participant, the experiment was divided in two parts. In each of these parts, participants navigated in 3 different and randomly presented virtual environments: (1) a dynamic and textured VE with moving physical objects (VE_1, Figure 3-A), a static and textured VE (VE_2, Figure 3-B) and a static and flat colored VE (VE_3, Figure 3-C).

During the first part, participants were asked to freely explore the virtual environment without a specific task (T_f, use-case: virtual visits). Then, during the second part, participants were asked to search for keys hidden in the VE (task T_k), and to pick up a maximum of them (use-case: video games). The number of available keys was not given to participants. The second part was meant to study the performance of our model when a task is involved since the presence of task is known to have an influence on gaze patterns [Yarbus 1967]. To take into account the task involved during the exploration, the $Task(s)$ value was set to 1.0 for surfels belonging to keys and 0.5 for all other objects. The same task value was used for the model of Lee *et al.* [Lee et al. 2009].

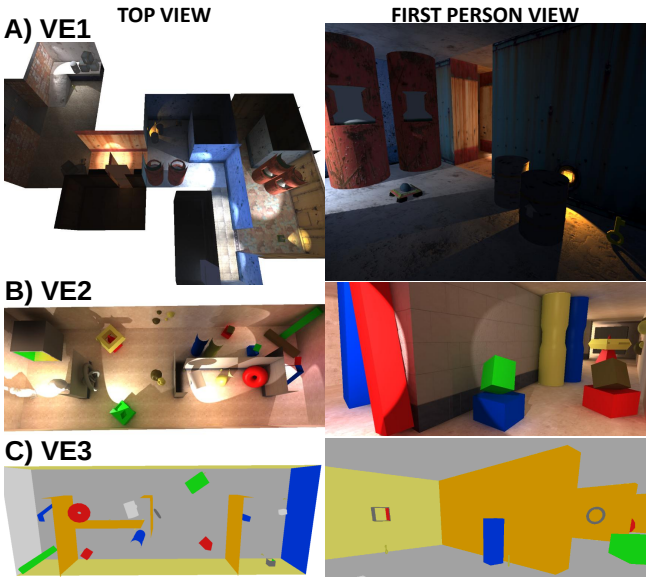

Figure 3: *The three virtual environments used in our experiment. A) Textured and dynamic VE, B) textured and static VE and C) flat colored and static VE.*

The experiment started with a training session in which participants were able to get used with the navigation protocol during 1 minute. Each navigation session of each part lasted 2 minutes. A calibration of the Tobii gaze tracker was conducted at the beginning of each part. For each participant, the overall experiment lasted 20 minutes. All sessions were recorded, and we were able to replay each session to evaluate the performances of the various visual attention models.

4.3 Results

To compare our model ($Mour$) with the model of Lee *et al.* [Lee et al. 2009] ($Mlee$), we computed several performance indicators. Performance indicator P_1 represents the percentage of time spent by the gaze point computed by each model inside a circle area having a radius of r degrees of the visual field and centered on the gaze point G_T computed by the Tobii system (considered here as the ground truth). We have tested several r values for P_1: 2, 4, 6 and 8 degrees (respectively corresponding to a radius of 71, 143, 215 and 287 pixels). Performance indicator P_2 represents the percentage of time spent by the gaze point computed by each model on the same object as the one located at the level of G_T.

Concerning P_1, the model proposed by Lee *et al.* [Lee et al. 2009] was not designed to output a continuous 2D gaze point. Thus, we propose to compute the final gaze position G_{lee} corresponding to the use of the model of Lee *et al.* [Lee et al. 2009] as the mean of positions of pixels belonging to the selected visual object (the one obtaining the highest attention). Our model is not designed to output an attended mesh. Thus, concerning P_2, we have computed the final gazed object of our model as the mesh positioned under the gaze point G_{our} computed by our model.

We have conducted a repeated-measures ANOVA on the dependent variable P_1 with the independent variables being the four radii r and the models $Mour$ and $Mlee$ (Figure 4-A). The ANOVA revealed a significant main effect of the model used ($F(1,11) = 313.32$, $p < 0.01$). Tukey post-hoc comparisons showed a significant difference between the two models for each radius value r ($p < 0.01$). The ANOVA also revealed a significant radius $\times$ model interaction ($F(3,33) = 234.68$, $p < 0.01$) meaning that the difference in accuracy between the two models significantly increases when r increases. The detailed comparisons of both models concerning P_1 are presented in Figure 4-B for the case where $r = 4$ (which was related work [Hillaire et al. 2009]). In this case, we have conducted a 2 (model) $\times$ 3 (VE) $\times$ 2 (task) repeated-measures ANOVA. It revealed a significant main effect of the model used on performance ($F(1,11) = 225.96$, $p < 0.01$). Then, Tukey post-hoc comparisons showed that the performance of $Mour$ model was significantly higher than $Mlee$ ($p < 0.01$) for all combinations of VE and Task. This is confirmed by the fact that G_{our} was found to be closer to G_T than G_{lee} 71% of the time. Furthermore, Tukey post-hoc comparisons revealed that neither $Mour$ nor $Mlee$ performance were influenced by the VE. Also, for each VE, They revealed a significant difference of performance for $Mour$ and $Mlee$ between T_f and T_k (in each case, $p < 0.01$).

Concerning the second performance indicator P_2, a 2 (model) $\times$ 3 (VE) $\times$ 2 (task) repeated-measures ANOVA again revealed a significant main effect of the model used on performance ($F(1,11) =$

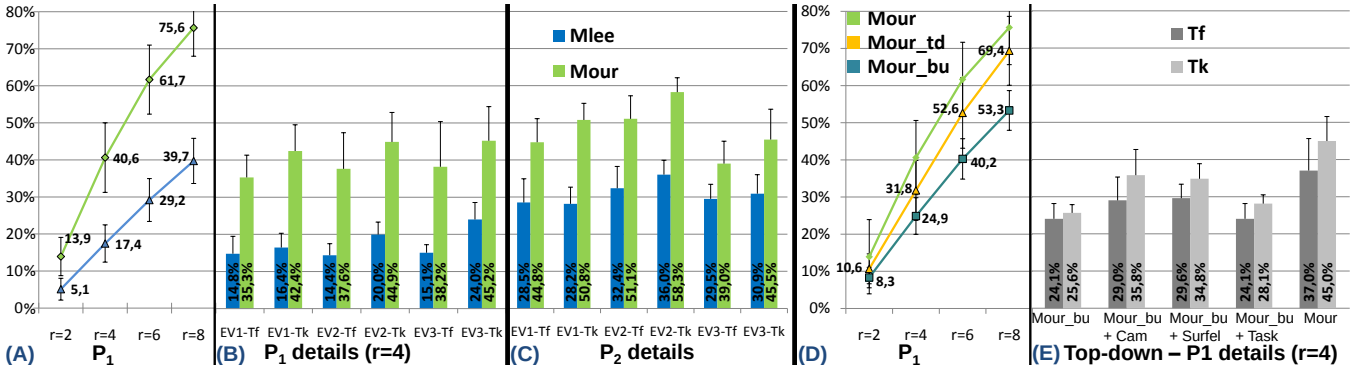

Figure 4: *Experimental results: performance obtained by various attention models (mean and standard deviation). A, B and C compare our visual attention model (Mour) to the existing model of Lee (Mlee). D and E compare our complete model to its separated bottom-up (Mour_bu) and top-down (Mour_td) components. A, B, D and E represent the percentage of time spent by the computed gaze point in a circle having a radius of r degree of the visual field centered on the ground truth gaze position. C represents the percentage of time spent by the computed gaze point on the same object as the one corresponding to the ground truth gaze position. A and D give global performance for* $r = \{2, 4, 6, 8\}$. *B, C and E give performance details with* $r = 4$ *for each VE (1,2 and 3) and each Task (*T_f *and* T_k*).*

560, $p < 0.01$). Figure 4-C exhibit a mean accuracy of 48.2% (SD = 8.41) for our model $Mour$, and a mean accuracy of 30.9% (SD = 5.6) for the previous model $Mlee$. The ANOVA also revealed a significant Model $\times$ Task interaction ($F(1, 11) = 8.32$, $p < 0.05$). Tukey post-hoc comparisons confirmed that $Mour$ is significantly influenced by the task ($p < 0.05$) with an accuracy of 51,5% (SD = 7.7) for task T_f and of 44.9% (SD = 7.8) for task T_k. The other model $Mlee$ was not found to be significantly influenced by the task ($p = 0.87$), which leads to an accuracy for P_2 of 31,7% (SD = 5.5) for the task T_f and of 30.1% (SD = 5.1) for T_k.

Secondly, we have compared the performance P_1 when using our complete model $Mour$ with the use of only its bottom-up component $Mour_{bu}$, or only its top-down component $Mour_{td}$ (Figure 4-D). The ANOVA revealed a significant main effect of the model used on performance ($F(2, 22) = 141.75$, $p < 0.01$). Then, Tukey post-hoc comparisons showed significant differences between each model for each value of r ($p < 0.01$ in each case).

The top-down part of our model has been designed around three major components: (1) the novel surfel-based representation of visual objects, (2) screen-space weights and (3) task. They correspond to *Surfel part*, *Cam part* and *Task* in Figure 1. We have further evaluated the elementary contribution of these three top-down components with that of the bottom-up model only by successively adding their contribution when $r = 4$ (see Figure 4-E). A 5 (model) $\times$ 2 (task) repeated-measures ANOVA revealed a significant main effect of the model used ($F(4, 44) = 92.73$, $p < 0.01$). Tukey post-hoc comparisons showed that each model was significantly different from the others excepted $Mour_{bu} + Cam$ as compared to $Mour_{bu}$+*Surfel* ($p = 0.99$), and $Mour_{bu}$ as compared to $Mour_{bu} + Task$ ($p = 0.98$). The ANOVA also revealed a significant model $\times$ task interaction ($F(4, 44) = 5.50$, $p < 0.01$). In this case, Tukey post-hoc comparisons showed that each model was significantly influenced by the task excepted $Mour_{bu}$ ($p = 0.79$) and $Mour_{bu} + Task$ ($p = 0.09$).

4.4 Discussion

Overall, the results on two performance indicators show that our model performed significantly better than the previous model of Lee *et al.* [Lee et al. 2009] when exploring various 3D VE.

Firstly, concerning performance indicator P_1, our model performed significantly better than $Mlee$ [Lee et al. 2009] (more than 100% increase in performance), corresponding to a higher percentage of time spent by our computed gaze point close to the ground-truth gaze position (position given by the real gaze-tracker). Interestingly the performance of our model was not significantly influenced by the VE used. This suggests that our model is general enough to support many different kinds of 3D VE. Secondly, we found that both models performed significantly higher when the searching task was given to participants. This suggests that, when an implicit task is involved during the navigation, users' attention seems more predictable thanks to a higher correlation with the top-down component controlling overall gaze direction [Itti 2005] and including a task related weight. It also confirms Sundstedt *et al.* [Sundstedt et al. 2008] findings, suggesting that implementing a top-down component is highly beneficial for a visual attention model.

The lower accuracy of $Mlee$ concerning P_1 is probably due to the fact that this model was designed to output a 3D object and not a continuous 2D gaze point on the screen. Thus, we have also compared both models using a second performance indicator P_2 which represents the percentage of time spent by the computed gaze point on the same object as the one corresponding to the ground truth gaze position. Surprisingly, even in this case, our model gives significantly better results than the previous model $Mlee$. Besides, performance of $Mlee$ was actually lower than the one reported in their paper [Lee et al. 2009]. This could be due to the fact that in [Lee et al. 2009] frames showing only the background were all excluded from the analysis, whereas, in our case, all frames were kept except those were G_T was reported as invalid.

Our results also revealed that the performance of our complete model was significantly higher than that when using only its bottom-up component alone or only its top-down component ($Mour$ vs $Mour_{bu}$ or $Mour_{td}$). This suggests that visual attention models based only on a bottom-up or a top-down component would not be as effective at computing human attention as compared to using both components together. In other words, this confirms again the benefit of adding a top-down component to a visual attention model as stated in [Sundstedt et al. 2008].

Furthermore, we have studied the contribution of separated top-down weights (Figure 4-E). Our analysis also revealed that adding screen-space weights, surfel weights or task weights to the single bottom-up component $Mour_{bu}$ resulted in a significant increase in the overall performance (for both T_f and T_k navigations). This suggests that the component Cam, $Surfel$ and $Task$ are reliable

top-down weights. Finally, when combining all top-down components together, the performance was also found to be significantly better, suggesting that it is important to mix several top-down components adapted to the context in which the visual attention model is used in order to correctly identify areas of interest to the user.

When replaying sessions, we could visually observe that the habituation simulation was very useful to predict participants' gaze when discovering new rooms or looking behind objects. The habituation simulation seems particularly effective during the searching task T_k. Indeed, in this case, participants were actively searching for keys in places they did not explore before. When navigating freely, the two statistical top-down components, $CamRot$ and $ScrC$, were also found helpful to better position the computed gaze point at the center of the screen and/or in the direction of the camera rotation when turning, i.e. where humans often gaze. However, we could sometimes observe that participants were rapidly parsing several areas of the screen in less than 2 seconds. In such cases, no components were able to simulate and account for this fast gaze pattern. This suggests that our model would benefit from the implementation of a more advanced gaze behavior simulator that would accurately simulate saccade and smooth pursuit gaze patterns as well as fast scene-parsing behavior.

Taken together our results suggest that our novel visual attention model could be used in various real-time 3D applications involving first-person navigation. The computed gaze point could be used to better distribute computational resources to efficiently render a VE [Lee et al. 2009]. It could also be used to simulate natural effects occurring in human vision to improve graphical rendering and immersion in the VE [Hillaire et al. 2008a].

5 Conclusion

In this paper, we have presented a novel visual attention model to compute user's gaze position on screen in real-time. This model is specifically designed for exploration of 3D virtual environments and can compute, for the first time, a continuous gaze point position. This novel visual attention model is made of two components: a bottom-up and a top-down component. Contrary to previous models, which used a mesh-based representation of visual objects, we have introduced a new data representation based on surface-elements. We propose this solution to close the gap between screen-space and object-space approaches. The bottom-up and top-down components are combined to create a final attention map and compute the continuous gaze point.

We have conducted an experiment to study the performance of our method. Overall, the results show that our model performed significantly better than a state-of-the-art model when exploring various 3D virtual environments with an increase in performance of more than 100%. Taken together our results suggest that our novel visual attention model could be used in various real-time applications such as video games or virtual reality systems.

Future work could first concern the improvement of our visual attention model. We could notably enhance the gaze pattern simulator by simulating real fixation/saccade and smooth pursuit pattern. Second, we would like to further evaluate our model in various VE and with various contexts such as virtual training, architectural visits or games.

References

CATER, K., CHALMERS, A., AND WARD, G. 2003. Detail to attention: exploiting visual tasks for selective rendering. *Proc. of the 14th Eurographics workshop on Rendering*, 270–280.

CHEN, H., AND LIU, X. 2008. Lighting and material of halo 3. *Siggraph 2008 courses*.

GLENSTRUP, A., AND ENGELL-NIELSEN, T., 1995. Eye controlled media : Present and future state. Master thesis, University of Copenhaguen.

HILLAIRE, S., LÉCUYER, A., COZOT, R., AND CASIEZ, G. 2008. Using an eye-tracking system to improve camera motions and depth-of-field blur effects in virtual environments. *Proc. of IEEE Virtual Reality*, 47–50.

HILLAIRE, S., LÉCUYER, A., COZOT, R., AND CASIEZ, G. 2008. Depth-of-field blur effects for first-person navigation in virtual environments. *IEEE Computer Graphics and Applications 28*, 6, 47–55.

HILLAIRE, S., LÉCUYER, A., BRETON, G., AND REGIA-CORTE, T. 2009. Gaze behavior and visual attention model when turning in virtual environments. In *Proceedings of ACM Symposium on Virtual Reality Software and Technology*, 43–50.

ITTI, L., KOCH, C., AND NIEBUR, E. 1998. A model of saliency-based visual attention for rapid scene analysis. *in IEEE Transactions on Pattern Analysis and Machine Intelligence 20*, 11, 1254–1259.

ITTI, L. 2005. Quantifying the contribution of low-level saliency to human eye movements in dynamic scenes. *In Visual Cognition 12*, 1093–1123.

LEE, S., KIM, G., AND CHOI, S. 2009. Real-time tracking of visually attended objects in virtual environments and its application to LOD. *In IEEE Transactions on Visualization and Computer Graphics 15*, 1 (Jan.-Feb.), 6–19.

LONGHURST, P., DEBATTISTA, K., AND CHALMERS, A. 2006. A GPU based saliency map for high-fidelity selective rendering. *Proc. of the 4th international conference on Computer graphics, virtual reality, visualisation and interaction in Africa*, 21–29.

LUEBKE, D. P., AND HALLEN, B. 2001. Perceptually-driven simplification for interactive rendering. In *Proceedings of the 12th Eurographics Workshop on Rendering Techniques*, 223–234.

NAVALPAKKAM, V., AND ITTI, L. 2005. Modeling the influence of task on attention. *In Vision Research 45*, 2, 205–231.

ROBERTSON, A. R. 1990. Historical development of cie recommended color difference equations. *In Color Research and Application 15*, 3, 167–170.

ROBINSON, D. A. 1965. The mechanics of human smooth pursuit eye movement. *Journal of Physiology 180*, 569–591.

SEARS, C., AND PYLYSHYN, Z. 2000. Multiple object tracking and attentional processing. *Journal of Experimental Psychology 54*, 1, 1–14.

SUNDSTEDT, V., STAVRAKIS, E., WIMMER, M., AND REINHARD, E. 2008. A psychophysical study of fixation behavior in a computer game. In *Proceedings of the 5th symposium on Applied perception in graphics and visualization*, 43–50.

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

YARBUS, D. 1967. *Eye motion and vision*. Plenum Press.