Design and Application of Real-Time Visual Attention Model for the Exploration of 3D Virtual Environments

Sébastien Hillaire, Anatole Lécuyer, Tony Regia-Corte, Rémi Cozot, Jérôme Royan, and Gaspard Breton

Abstract—This paper studies the design and application of a novel visual attention model designed to compute user's gaze position automatically, i.e., without using a gaze-tracking system. The model we propose 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 processes which take place in the brain such as the gaze behavior associated to first-person navigation in the virtual environment. Our visual attention model combines both bottom-up and top-down components to compute a continuous gaze point position on screen that hopefully matches the user's one. We 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 sometimes more than 100 percent of accuracy gained. This suggests that computing a gaze point in a 3D virtual environment in real time is possible and is a valid approach, compared to object-based approaches. Finally, we expose different applications of our model when exploring virtual environments. We present different algorithms which can improve or adapt the visual feedback of virtual environments based on gaze information. We first propose a level-of-detail approach that heavily relies on multiple-texture sampling. We show that it is possible to use the gaze information of our visual attention model to increase visual quality where the user is looking, while maintaining a high-refresh rate. Second, we introduce the use of the visual attention model in three visual effects inspired by the human visual system namely: depth-of-field blur, camera motions, and dynamic luminance. All these effects are computed based on the simulated gaze of the user, and are meant to improve user's sensations in future virtual reality applications.

Index Terms—Visual attention model, first-person exploration, gaze tracking, visual effects.

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 [1] or to simulate natural effects occurring in human vision improving users' perception of the VE [2].

A straightforward way to compute user's gaze point position on a screen is to use a gaze tracking system [3]. Since their creation in the late 19th century, before the computer existed, these systems have been considerably enhanced [3]. 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. A lot of research has been dedicated to the evaluation of human attention when looking at pictures [4] or video [5] 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. [6] 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 VEs. It is composed of new, and improved, components and data representations. We also report on the wide application possibilities the gaze point offers and present several methods to take advantage of it. Our main contributions are:

- A visual attention model specifically designed for real-time exploration of 3D VEs which can compute a continuous 2D gaze point position. This model notably introduces a novel representation of visual

- *S. Hillaire is with the Orange Labs and INRIA/IRISA, 181 rue de Nantes, Rennes 35700, France. E-mail: shillair@irisa.fr.*
- *A. Lécuyer and T. Regia-Corte are with the INRIA, Campus Universitaire de Beaulieu, Rennes Cedex F-35042, France. E-mail: anatole.lecuyer@inria.fr.*
- *R. Cozot is with the INRIA and the University of Rennes 1, Campus Universitaire de Beaulie, Rennes Cedex F-35042, France.*
- *J. Royan is with the Orange Labs, 4 rue du clos courtel, Cesson-Sévigné 35510, France.*
- *G. Breton is with the Orange Labs, 80 avenue des Buttes de Coësmes, Espace Entreprises Gallium, Rennes 35700, France.*

Manuscript received 10 Feb. 2011; revised 21 July 2011; accepted 25 Aug. 2011; published online 12 Sept. 2011.
Recommended for acceptance by T. Komura, Q. Peng, G. Baciu, R. Lau, and M. Lin.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCGSI-2011-02-0028.
Digital Object Identifier no. 10.1109/TVCG.2011.154.

objects based on surface-elements (*surfels*). We will show that this representation has several advantages over the mesh-based representation. Our model also implements for the first time a novel component simulating the gaze behavior of users walking and turning in VEs [7];

- An extended evaluation of our model as compared to a state-of-the-art approach;
- A presentation of methods taking advantage of the gaze feature in order to apply level-of-detail and perceptually inspired visual effects.

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. We will report on an experiment conducted to evaluate the efficiency of our model as compared to a state-of-the-art model. We will also discuss the efficiency and usability of the proposed model in VR applications. Finally, we will expose several visual effect adapted in real time to the gaze point position of the user for the purpose of level-of-detail and visual feedback improvement.

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 [5]: bottom-up and top-down components.

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 [8], the human visual system does not have the capabilities to analyze a scene in parallel. Actually, the human visual system can only perceive primitive features in parallel [9] in order to detect highly contrasted areas. According to the feature integration theory [9], all this contrast information is combined all together to finally identify salient areas in the visual field. Then, the human visual system uses a sequential visual search to quickly analyze the scene [9]. For example, when someone first looks at a scene, his/her gaze is first unconsciously attracted by visually salient areas to rapidly perceive the important parts of the scene [4]. Several visually salient features have been identified in previous research [9], [4]: red/green and blue/yellow antagonistic colors, intensities, orientations, etc. Inspired by the feature integration theory [9], bottom-up visual attention models have been developed to compute a saliency map from an image [4]. The saliency value of each pixel of the saliency map represents its attractiveness, i.e., the higher the 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 [5], depth [6], or motion [5].

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 and is highly dependent on the knowledge, memory, and task. For example, Yarbus [10] 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 [11], 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 [8], habituation [11], or memory [12].

Bottom-up only models [4], [5] have been reported as good predictors as they were able to predict a nonnegligible 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. [13] 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 [5], 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 [13].

Sears and Pylyshyn [14] 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 three to five 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, little research has been dedicated to the use of visual attention models for real-time attention or gaze prediction during exploration of 3D VEs. To the best of our knowledge, only the visual attention model based on the MOT theory proposed by Lee et al. [6] 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 percent of the time during free navigation and 62 percent of the time during a research task involving navigation 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 [1], [2]. As future work, Lee et al. [6] suggested taking into account the novelty of objects and other adapted top-down components. To this aim, Hillaire et al. [7] have studied user's gaze behavior when walking and turning in VEs. 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 for physical devices such as a webcam or an expensive gaze tracker. In the following sections, 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 saliency map using several visual features: intensity and antagonistic colors [4], depth [11], and motion [5].

3.1.1 Computation of Feature Maps

The computation of a saliency map first requires a feature map for each visual features (Fig. 1A):

- **Antagonistic colors and luminance**. Originally, Itti et al. [4] used red/green and blue/yellow antagonistic colors as well as intensities. In their model, antagonistic colors were computed using simple operations on RGB components. In our case, we propose to use the perceptual Lab color space which takes into account human perception [15]. Moreover, it 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 Fig. 1.
- **Depth**. We propose to use a depth map as proposed in [11], [6]. The value $F_d(p)$ for each pixel p of the depth feature map F_d is computed using (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. [6] 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 (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 multiscale Center-Surround difference operator [4] simulating the response of brain neurons which receive stimuli from the visual fields. Instead of a dyadic Gaussian pyramid, we use an approximation consisting of using the fast hardware mipmap pyramid generation of Graphics Processing Units (GPU) (see [6]). The conspicuity maps, i.e., C_l , C_a, C_b, C_d, and C_m in Fig. 1B, are computed using (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. This filter is of lower quality, i.e., a box filter, and can miss the detection of some visual features. But, as [6], we found that, practically, it works well

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

Finally, the conspicuity maps are normalized using the $\mathcal{N}$ operator as described by Itti et al. [4]. For the sake of performance, we replace the mean of local maxima by the mean of all values in the conspicuity map.

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 (Fig. 1C). It is the result of a linear combination of each conspicuity map using (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 of 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 [6] and habituation [11] components. Our model relies also on screen-space weights to take into account the observed gaze behavior of humans navigating in VEs using a first-person view.

3.2.1 Surfel-Based Representation of Visual Objects

Previous models for 3D exploration of VEs use a representation of visual objects based on meshes [6]. This can be seen as a limitation as it is not possible to differentiate subparts 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

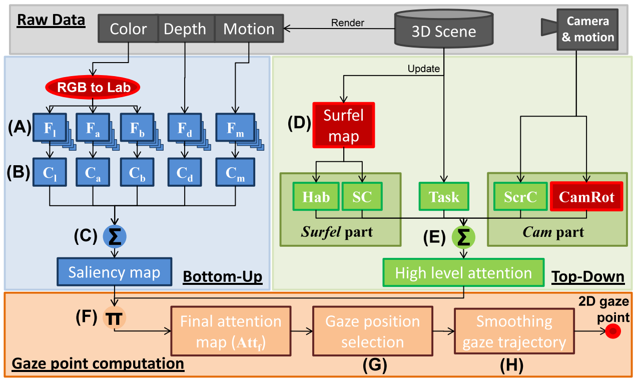

Fig. 1. Overview of our visual attention model architecture. (A) feature maps, (B) conspicuity maps, (C) bottom-up attention (saliency map), (D) update of per-surfel data, (E) top-down attention, (F) computation of final attention on screen, (G) computation of the possible next gaze position, and (H)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

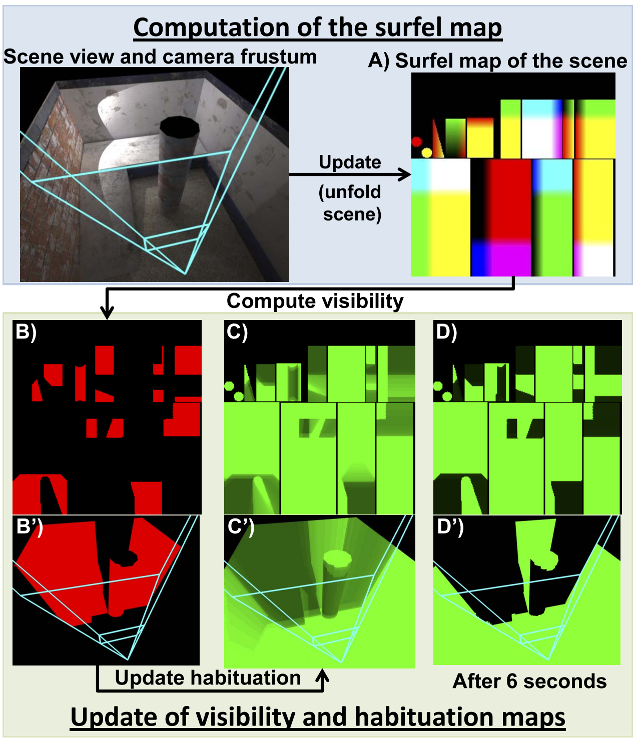
Computation of the surfel map
Scene view and camera frustum
A) Surfel map of the scene
Update
(unfold scene)
Compute visibility
B)
C)
D)
B')
C')
D')
Update habituation
After 6 seconds
Update of visibility and habituation maps

Fig. 2. Computation of the surfel map, visibility, and habituation maps. A) the surfel map containing world space surfel position, 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.

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 submeshes 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, as an alternative solution to mesh subdivision, we propose to use a discretization of polygonal surfaces into smaller elements having the same world space size: surface elements also known as *surfel*. Surfel are well known and they are used as a lightmap encoding the irradiance per surface elements [16]. 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 Fig. 2. Also, for the sake of performance, the surfel map of dynamic objects is only updated when they move.

There are several advantages in using a surfel representation of visual objects. First, this representation is not dependent on the way the scene has been modeled: each surfel is represented as a small square patch having the same size in world space. Second, thanks to this uniform world space size, we obtain a uniform distribution of visual objects. Finally, each surfel can be easily represented and stored in a texture (see Section 3.2.2). Thus, their update and use for visual attention estimation can take advantage of the computational power of modern graphic hardware by processing all surfels in parallel.

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 area in the texture. This corresponds to unfolding triangles of the 3D meshes in the 2D texture. 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 effectively 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 [16].

To fill the surfel map with data, we simply render the 3D meshes into the texture. 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 (Fig. 1D). The habituation map is created when the application starts. At runtime, few areas of the habituation map are updated for dynamic objects and only when they move. Then, all per-surfel computations are done in the surfel map using the highly parallel computation power of graphic hardware.

After preliminary testing, we have set the world space size of surfels to 20 cm. Objects smaller than 20 cm, i.e., small keys in our experiment, are virtually scaled up. In these cases, we have visually adjusted the scale factor to have each object's surfaces represented by at least one surfel. Each VE presented in Section 4 fits in a single 256×256 surfel map with dynamic objects being stored in another 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 [11]. For each surfel, we compute the habituation value for current frame using (5) where t is the elapsed time in milliseconds

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

Surfel are initialized, first, with a high-interest value. When a surfel s become visible, the habituation is attenuated using an exponential decay [11] with interest in s going under 0.1 in 7 s ($h^- = 3{,}000$) (see [11] for details). When s is not visible, it is linearly regaining full interest in 20 s ($h^+ = 20{,}000$). The visibility $vis(s)$ of a surfel s is determined using shadow mapping (Fig. 2A) and takes advantage of the already computed depth buffer of the rendered view point in 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 (Fig. 2B). Then, when the user is moving in the VE, areas that are discovered/new have a high interest and are more likely to attract his/her attention.

- **Spatial context**. Lee et al. [6] 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. [6] 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 ((6)-$SC_d(s)$). Since users tend to get close to objects they want to inspect [6], our spatial component also gives more importance to surfels moving toward the camera ((6)-$SC_{\Delta d}(s)$). For each surfel s, the final spatial context value $SC(s)$ is computed using

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

where $C_1 = D/0.707$ and D is the distance when surfel are considered as more important (for more details, please refer to [6]). 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 a speed equal 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 VEs has been studied in [7], [17]. 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 [17]. We propose to model this behavior as in [6] 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 (Fig. 1E). 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 [7]. In this paper, authors have studied user's gaze behavior when turning in VEs. They have proposed a new function able to compute an attentional weight on the whole screen corresponding to 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 (Fig. 3). Our model introduces this function to compute an attention value $CamRot(p)$ for each pixel p on the screen.

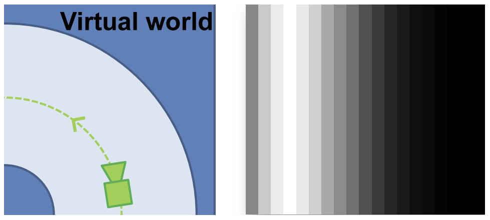

Fig. 3. Attentional weights computed on screen (right) in the case of a left-turn of $100^\circ/s$ (left) in the VE using the component described in [17] (white pixels correspond to high-attention levels).

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 VEs.

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 [6]. 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 (7) (Fig. 4B). 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

$$\begin{aligned} TD(p) = (&Hab(s) + CamRot(p) + Task(s) \\ &+ SC(s) + ScrD(p)) \times 0.2. \end{aligned} \tag{7}$$

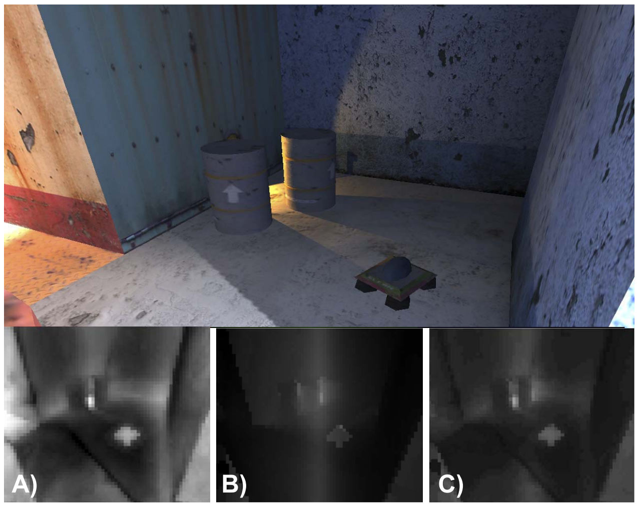

Fig. 4. The saliency map (A), the top-down map (B), and the final attention map (C) computed from a view of a virtual environment.

In the last step, the final attention map Att_f (Fig. 4C) is computed from both the top-down (Fig. 4A) and bottom-up (Fig. 4B) components using (8) (Fig. 1F) where $vis(s)$ is visibility of surfel s. Because our model is simulating visual attention as a general case, using equal weights for each component is globally sufficient. Finally, the gaze position is selected as the position of the pixel having the highest attention level (Fig. 1G). Then, it is sent to the filter smoothing the gaze trajectory

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

3.3.2 Smoothing Gaze Trajectory

Raw gaze positions computed using our visual attention model are processed in order to smooth out the final gaze position and trajectory (Fig. 1H).

In the human visual system, the duration of eye saccades is from 120 to 300 ms long depending on the rotation of the eyes [18]. Mean fixation duration varies between 200 and 600 ms. We consider a mean frequency of eyes saccades plus fixation of 600 ms. Thus, we filter the input gaze position with a low pass using a cut-off frequency of 1.67 Hz. This low-pass filter allows the simulation of the smooth pursuit phenomenon [18], occurring when looking at a visual object smoothly moving, while allowing fast gaze jumps simulating saccades.

3.4 Implementation Details and Performance

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 VEs are rendered using our own rendering engine featuring 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. This allows us to study the accuracy of our model in several situations, from very simple to game-like VEs.

During the computation of the *bottom-up* saliency map, features, and conspicuity, maps are stored in two 3-channel textures containing, respectively, (F_l, F_a, F_b) and (F_d, F_m). All the textures used to compute the saliency map have a resolution of 256×256. Our normalization operator $\mathcal{N}$, simplified as compared to [4], 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.

The previous and current surfel positions are required to compute the spatial context component. To do so, 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 only update the surfel texture every 100 ms 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 last step of our model consists in computing the final gaze position as the pixel having the highest attentional value. For this aim, $Att_f(p)$ is a 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, used to compute the normalization 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. It could be considered as computationally too expensive. However, this raw data are already computed by many existing game engines to add visual effects such as depth-of-field (DOF) or motion blur. Thus, adding our model to an existing engine should not require too many changes or additional resources.

Our visual attention model can run in real time thanks to the graphic hardware. On a laptop PC (Intel Core 2 2.5 Ghz, nVidia GeForce3700M), 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 170 FPS without. Detailed GPU computation times are given in Table 1. The low computation time of our model would allow it to be used in several real-time 3D applications and games.

4 Experimental Evaluation

We 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. [6]. To the authors best knowledge, this is the only model adapted to real-time 3D exploration of VEs proposed so far. Twelve naïve participants (10 males, 2 females) with a mean age of 31.8 (SD $= 6.4$) participated in

TABLE 1
Computation Time in Milliseconds for Each Step of the Visual Attention Simulation

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

our experiment. They were all familiar with the first-person navigation 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 $1{,}280 \times 1{,}024$. The screen was 37.5 cm width. They were at a distance of 60 cm from the screen and no sound was played. The VEs were rendered in real time with a constant refresh-rate of 75 Hz.

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 2.5 cm 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 1.75 m and walking speed was 1.15 m/s [7].

4.2 Procedure

For each participant, the experiment was divided into two parts. In each of these parts, participants navigated in three different and randomly presented virtual environments: 1) a dynamic and textured VE with moving physical objects (VE_1, Fig. 5A), 2) a static and textured VE (VE_2, Fig. 5B), and 3) a static and flat colored VE (VE_3, Fig. 5C).

During the first part, participants were asked to freely explore the VE 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 as many as they could (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 [10]. 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. [6].

The experiment started with a training session in which participants were able to get used with the navigation protocol for 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.

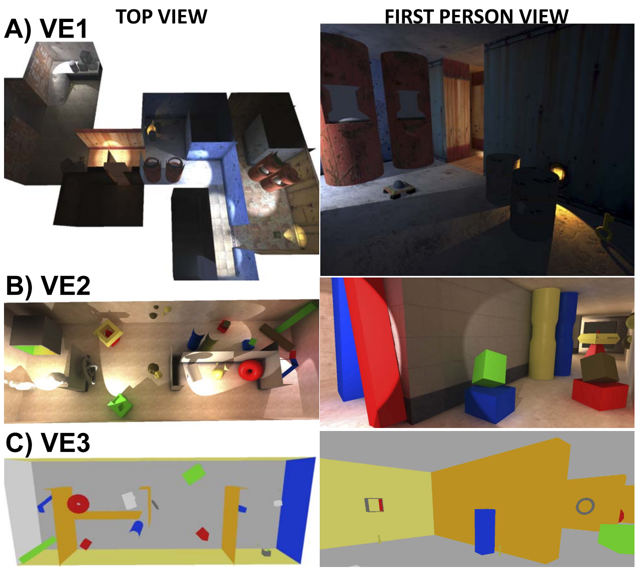

Fig. 5. The three virtual environments used in our experiment. A) Textured and dynamic VE, B) textured and static VE, and C) flat colored and static VE.

4.3 Results

Prior to the experiment we could formulate three hypotheses: (H1) When computing a gaze position on the screen, our model is more accurate than the state-of-the-art model [6], (H2) When computing the attended object, our model is more accurate than state-of-the-art model [6], and (H3) our model is more efficient when using all components together than when using only bottom-up or top-down component alone.

To compare our model ($Mour$) with the model of Lee et al. [6] ($Mlee$), we computed several performance indicators:

- 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 (ground truth).
- 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, 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). Also, the model proposed by Lee et al. [6] 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. [6] 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.

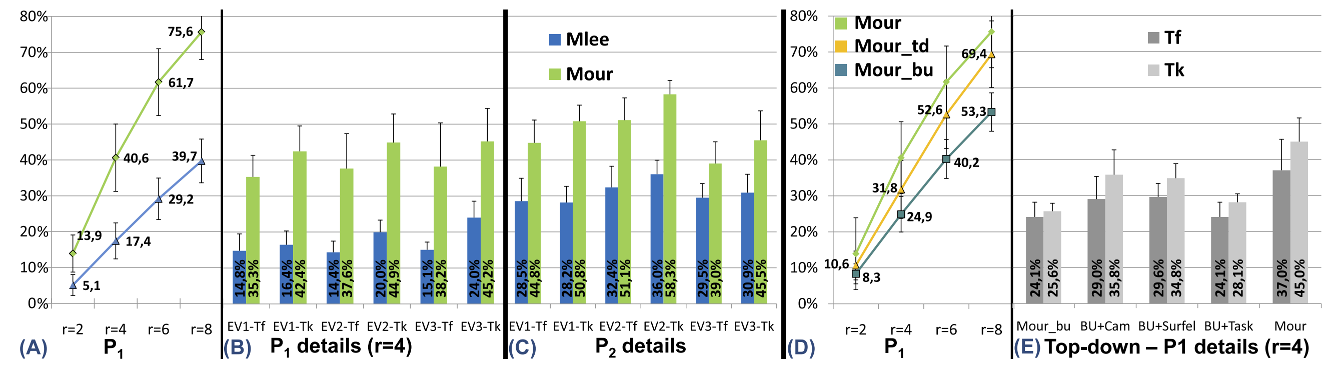

Fig. 6. Experimental results: performance obtained by various attention models (mean and standard deviation). (A), (B), and (C) compare our visual attention model (Mour) to the existing model of Lee (Mlee). (D) and (E) compare our model to its separated bottom-up (Mour_bu) and top-down (Mour_td) components. (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 and each Task.

Considering (H1), We 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 $Mlee$ (Fig. 6A). The ANOVA revealed a significant main effect of the model used ($F(1, 11) = 313.32$, $p < 0.01$). Tukey posthoc comparisons showed a significant difference between the two models for each radius 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 Fig. 6B for the case where $r = 4$ (which is related work [7]). In this case, we conducted a $2(\text{model}) \times 3\ (\text{VEs}) \times 2(\text{task})$ 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 posthoc 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. A Tukey posthoc 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, $p < 0.01$).

Considering (H2), a $2(\text{model}) \times 3(\text{VEs}) \times 2\ (\text{task})$ repeated-measures ANOVA on the second performance indicator P_2 again revealed a significant main effect of the model used on performance ($F(1, 11) = 560$, $p < 0.01$). Fig. 6C exhibits a mean accuracy of 48.2 percent ($\text{sd} = 8.41$) for our model $Mour$, and a mean accuracy of 30.9 percent ($\text{sd} = 5.6$) for the model $Mlee$. The ANOVA also revealed a significant Model $\times$ Task interaction ($F(1, 11) = 8.32$, $p < 0.05$). Tukey posthoc comparisons confirmed that $Mour$ is significantly influenced by the task ($p < 0.05$) with an accuracy of 51,5 percent ($\text{sd} = 7.7$) for task T_f and of 44.9 percent ($\text{sd} = 7.8$) for task T_k. The 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 percent ($\text{sd} = 5.5$) for the task T_f and of 30.1 percent ($\text{sd} = 5.1$) for T_k.

Second, considering (H3), 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 $Mour_{td}$ (Fig. 6D). The ANOVA revealed a significant main effect of the model used on performance ($F(2, 22) = 141.75$, $p < 0.01$). Then, Tukey posthoc 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 Fig. 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 Fig. 6E). A $5(\text{model}) \times 2\ (\text{task})$ repeated-measures ANOVA revealed a significant main effect of the model used ($F(4, 44) = 92.73$, $p < 0.01$). Tukey posthoc comparisons showed that each model was significantly different from the others except $Mour_{bu} + Cam$ when compared to $Mour_{bu} + Surfel$ ($p = 0.99$), and $Mour_{bu}$ when 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 posthoc comparisons showed that each model was significantly influenced by the task except $Mour_{bu}$ ($p = 0.79$) and $Mour_{bu} + Task$ ($p = 0.09$).

4.4 Discussion

Overall, the results on two performance indicators show that our model performed better than the model of Lee et al. [6] when exploring various 3D VE.

First, concerning performance indicator P_1, our model performed significantly better than $Mlee$ [6] (more than 100 percent increase in performance), corresponding to a higher percentage of time spent by our computed gaze point close to the ground-truth gaze position. 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. Second, 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 [5] and including a task related weight. It also confirms Sundstedt et al. [13] 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 [6]. This could be due to the fact that in [6] all frames showing only the background were excluded from the analysis, whereas, in our case, all frames were kept except those were G_T was reported as invalid (0.98 percent).

Our results also revealed that the performance of our complete model was significantly higher than that when using only its bottom-up components alone or only its top-down components ($Mour$ versus $Mour_{bu}$ or $Mour_{td}$). This suggests that visual attention models based only on a bottom-up or a top-down components 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 [13].

Furthermore, we have studied the contribution of separated top-down weights (Fig. 6E). Our analysis 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 analyzing 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 an advanced gaze behavior simulator that would accurately simulate saccade and smooth pursuit gaze patterns as well as fast scene-analysis behavior.

Taken together our results suggest that our novel model could be used in various real-time 3D applications involving first-person navigation. Using a visual attention model to compute a gaze position seems a valid approach to be used as in combination [19] or as a replacement to expensive gaze tracking systems.

4.5 Limitations

In its current state, our model simulates visual attention as a general case and with real-time performance. As such, it exhibits some limitations. As described in previous sections, we have simplified some algorithms in order to accelerate the computation time. The simplification of the $\mathcal{N}$ operator is identical as in [6], but could result in few differences in the saliency maps than that obtained in [4]. Currently, top-down components are combined using equal weights. However, for a search task, habituation might be a stronger factor and thus should be associated with a stronger weight. A search task involving a red object could give more importance to red areas in the bottom-up process. Our design was conducted with performance in mind as compared to more complete top-down models that require many extra data such as objects hierarchy [12]. Moreover, since our model is simulating visual attention as a general case, using equal weights seems globally sufficient. However, together with the lack of simulation of advanced gaze movements, these simplifications could be a reason for the gaze miss-predictions we obtain (Fig. 6).

Although our computed gaze point is often found close to the ground truth, small errors can result in undesired behavior of the system that uses it. Considering visual effects adapted to where the use is gazing at, e.g., depth-of-field, a small error in image space might cause high-depth differences (edge of objects) thus resulting in images that could annoy users [17] (wrong focal distance). In the context of perceptually-based rendering, wrong objects level-of-detail could be selected. However, it is important to note that we have also experienced these problems when using an accurate gaze tracker. Reaching a 100 percent accuracy still remains a challenging goal today in the gaze-tracking community. One way to solve this problem could be to combine both visual attention modeling and gaze-tracking approaches together [19].

5 Application of Visual Attention Model for Gaze-Based Rendering

In this section, we propose and illustrate different applications of gaze-tracking, either using a gaze-tracker or our visual attention model, for gaze-based rendering. Thereafter, we present several methods to modify the visual feedback to the user according to where he/she is looking at on the screen. We first propose a level-of-detail method for visual effects that heavily rely on multiple texture fetches. The goal of the presented approach is to accelerate the rendering process of this type of visual effects. Then, we expose three different gaze-based visual

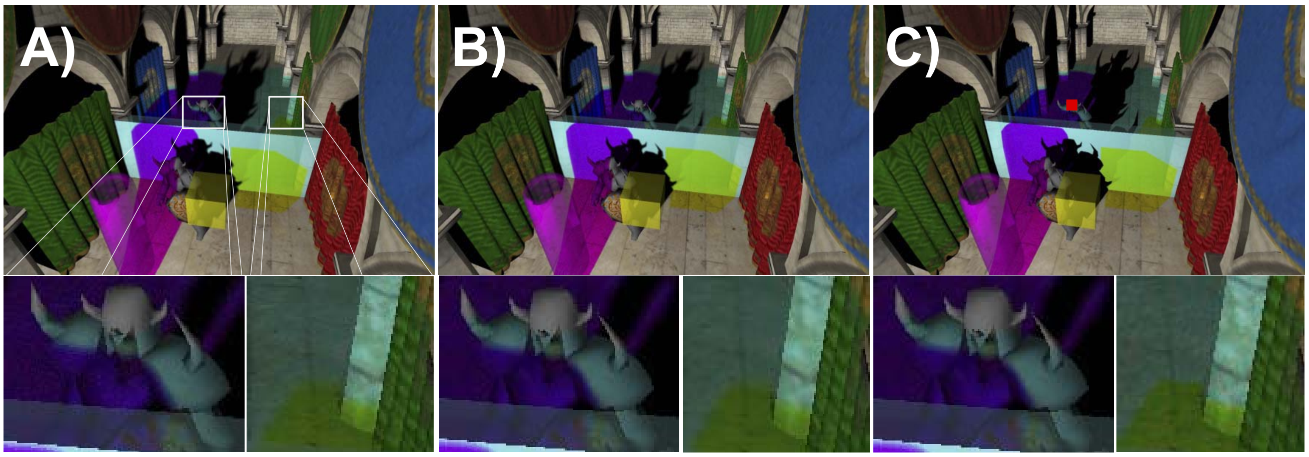

Fig. 7. Rendering of the Crytek's Sponza Atrium model [21] with transparent surfaces using the original CSSM algorithm (A), CSSMHQ (B) or CSSMGAZE (C) approaches. Close-up views show the difference in quality of the different algorithm. The red square in image C represents the gaze position of the user.

effects inspired from the human visual system which are adapted in real time to the user's gaze position. The goal when using these effects is to improve user's perception and immersion in a VE.

5.1 Gaze-Based Per-Pixel Level-of-Detail

5.1.1 Concept

In the rendering research area, the gaze point has been mainly used to manage the level-of-detail of the rendered VE in order to accelerate the rendering process and thus maintain a high-refresh rate of the application. Previous methods reduce details of objects that are far away from gaze point of the user on the screen. This has been achieved by progressively decimating meshes [1] or by choosing an appropriate mesh to render from a predefined set [6].

Nowadays, texture sampling on graphic hardware has become more and more costly compared to *Arithmetic Logic Unit* operations. Unfortunately, a lot of visual effects rely on multiple texture sampling for each pixel of the rendered image, e.g., *soft shadow mapping* or *relief mapping*. In this section, we propose to dynamically adapt the quality of the *Colored Stochastic Shadow-Mapping* (CSSM) method [20] by exploiting the gaze position of the user on the screen. To reduce the computation time, we propose to dynamically tune the number of texture samples required for each pixel based on their distance to the user's gaze point. For pixels close to the gaze position, a lot of samples will be taken in order to ensure an accurate visual quality. Then, the farther the pixels from the gaze position, the less samples of samples.

5.1.2 Gaze-Base Colored Stochastic Shadow Mapping

The CSSM algorithm solves an important challenge of real-time rendering: order-independent and colored shadows. It means that the rendering of partially covered and colored transmissive surfaces can be properly rendered in a unified way. This method uses a stochastic approach to sample the attenuation of light along a ray and distribute the sample spatially in the shadow map. For more details about this method, we refer the reader to the original paper [20]. As for all stochastic approaches, the drawback of this method is that it requires a large sampling kernel when rendering the shadow map to avoid high-frequency noise. This is especially true in the case of CSSM for surfaces having low transparency. The authors have proposed a *box-plus-cross-shaped* filter kernel consisting of 13 samples which results in low noise for a reasonably wide type of transparent surfaces. However, for low transparency surfaces that block most of the light, a lot of noise can appear (Fig. 7A). This is especially true when light rays travel through several surfaces. To solve this issue, the straightforward approach is to densely sample a wider area of the shadow map (CSSMHQ). As visible on Fig. 7B, using a 11-pixel wide box filter mostly removes the noise and results in a smooth color. However, this filter requires 121 samples per pixel. As a consequence, the refresh rate of the application drops significantly (see Table 2).

To improve the CSSM algorithm, we propose to dynamically adapt the number of samples per pixel based on their distance to the gaze position (CSSMGAZE). We define a quality value q for each pixel which is 1 at the gaze position and that linearly drops to 0 at a distance of half the height of the display screen. The width w of the box filter is then set per pixel to $w = 1 + 2 \times \lfloor q \times w_{max} \rfloor$. After preliminary testing, we have set $w_{max} = 6$ to get a high quality 11-pixel wide box filter around the gaze position.

Using the CSSMGAZE algorithm, the lighting solution near the gaze position is of high quality (Fig. 7). Then, the filtering quality progressively decreases when moving away from the gaze position (Fig. 7). As the user is not looking at this outer area, he/she is not able to perceive the difference while the application is able to render at a higher refresh rate as compared to the CSSMHQ algorithm. The computation time are shown in Table 2 when rendering the pictures of Fig. 7.

TABLE 2
Performance of CSSM Algorithms for Two Shadow Map Resolutions

Algorithm	512 × 512	2048 × 2048
CSSM	2.7ms (235fps)	4.6ms (113fps)
CSSMHQ	19.4ms (44fps)	24.9ms (30fps)
CSSMGAZE	5.6ms (130fps)	7.6ms (80fps)

5.1.3 Conclusion

Dynamically modifying the sampling kernel of visual effects on today's hardware seems a valid approach. This may be due to the fact that dynamic branching in shader is becoming increasingly effective. We believe that other methods could also benefit from this gaze-based scaling of the number of texture fetches.

5.2 Gaze-Based Visual Effects

The gaze point position can also be used to improve the visual feedback to the user by adding gaze-based visual effects inspired from the human visual system. Such effects are designed to improve the perception and immersion of the users in VE. To this aim, we present three gaze-based visual effects: two effects we have already proposed in previous work (compensated camera motion and depth-of-field blur [2]) as well as a novel one (luminance adaptation).

5.2.1 Gaze-Based Compensated Camera Motion

Recently, researchers have introduced the use of oscillating camera motions to enhance the sensation of walking in VE by reproducing the associated visual flow. This was shown to improve the feeling of immersion [2] and the perception of traveled distance [22]. However, the general visual flow might become a problem when fixating static points.

To overcome this issue, we propose to apply a compensation to the camera motion in order to smooth the visual flow at the level of the user's gaze point [2]. This more complex camera motion simulates the vestibulo-ocular reflexes of our eyes which allow a human to fixate a single point in space while he is walking by compensating his/her body and head motion. As a consequence, the user still has the impression of walking in the VE while being now able to fixate a single point in space without any effort [2].

5.2.2 Gaze-Based Depth-of-Field Blur

In reality, we do not perceive our whole surrounding environment sharply but partially blurred. This is mostly due to the fact that the chrystalin in our eyes acts as a lens: objects we are looking at, i.e., at the focal distance, appear sharp. Other objects in the visual field appear progressively blurred according to their distance to the focal distance.

To reproduce this phenomenon, we propose to adapt a depth-of-field blur effect to user's gaze point position on the visual display [2]. To this aim, we propose to dynamically adapt the focal distance to the distance at which the user is looking at [2]. A partially blurred version of the original sharp image can then be computed using a lens equation. Our visual attention model can be used as an alternative to gaze trackers to straightforwardly compute the gaze point of the user as the input of the DoF blur effect. Studies has revealed that users significantly preferred the depth-of-field blur when it was dynamically adapted to where they were looking at [2] and that it could help users to better reproduce distances in VE during manual interactions [23].

5.2.3 Gaze-Based Luminance Adaptation

High dynamic range (HDR) images refer to picture being composed of color values not limited to a maximum value of 1.0. Displaying such images on low-dynamic range (LDR) display peripherals is not a trivial task and is still a very active research field in the graphics community. The goal of a tone-mapping operator is to make all the details of a HDR image visible to the viewer when displayed on a LDR screen [24]. The human visual system is able to perceive a wide range of luminance values from dim interiors to outside areas brightly lit by the sun [24] at the same time. It is also able to adapt itself dynamically when the lighting condition changes rapidly. Inspired by this capacity of the human visual system, we propose a gazed-based tone-mapping algorithm.

The method we propose is targeted toward real-time 3D applications featuring high dynamic range lighting. It is based on the tone-mapping method proposed by Reinhard et al. [24]. This method computes the final luminance L_f of each pixel from the original luminance value L using (9) where g is middle gray, L_{min} is the minimum luminance to be mapped to black, L_{white} the smallest luminance that will be mapped to white, and L_{mean} the mean luminance of the view field. To compute the value of these parameters, we consider a low-resolution version of the displayed HDR image. For the sake of real-time performance, after preliminary testing, we propose to use the level of its mipmap pyramid having a width of 128. From this texture, we can easily compute L_{min} as the minimum luminance and L_{white} as the maximum luminance of all the pixels. The L_{mean} value is supposed to represent the mean luminance of the whole scene image. In our case, we choose to set this value to the luminance that is at the position the user is looking at, using the bilinear filtered luminance from the four neighboring pixels

$$\begin{aligned} L &= g\left(\frac{L - L_{min}}{L_{mean}}\right), \\ L_f &= \frac{L\left(1 + \frac{L}{L_{white}^2}\right)}{1 + L}. \end{aligned} \tag{9}$$

This dynamic adaptation of the Reinhard tone-mapping operator [24] allows the user to feel like he is looking at a HDR environment, although he cannot be dazzled, thanks to the fact that the synthesized image display on the LDR display peripheral is constantly adapted to the area he is looking at. The effect can be smoothed by applying a low-pass filter on L_{mean} to simulate temporal adaptation. The result of using this method is displayed on Fig. 8. One can notice that when the user is looking at a bright region, the overall image appears dimmer in order to perceive more details where the user is looking. Conversely, the scene appears brighter when the user is looking at dark areas. As the DoF blur effect, the expected outcomes for the users are a better immersion feeling and perception of the VE.

6 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 the 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

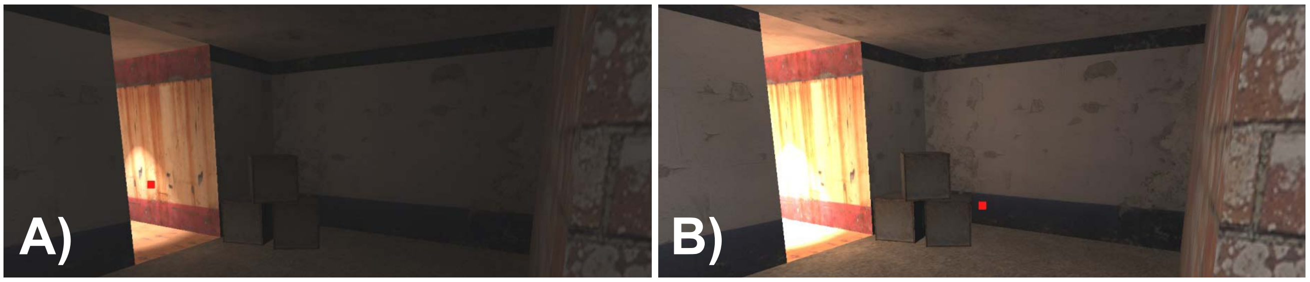

Fig. 8. Dynamic luminance adaptation based on the user's gaze point on the screen (the red square).

surfels as a solution to close the gap between screen-space and object-space approaches.

We 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. 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.

Finally, we have proposed to take advantage of the computed gaze point to adapt the rendering of the virtual environment to the user's attention. To this aim, we have first proposed to dynamically adapt the quality of visual effects in order to accelerate rendering process while maintaining a high frame-rate. Second, we have exposed several gaze-based visual effects inspired from the human visual system. These effects aim at improving the immersion of the user. Although not being a fully perceptually correct simulation of the human visual system, we believe that using such visual effects inspired from it is an interesting path to follow in order to improve the overall perception, immersion, and experience of users exploring a virtual world.

7 Future Work and Perspectives

Further research efforts could be directed toward the improvement of our visual attention model. First, we would like to study and propose other ways to combine the various components used in our model. Second, we would like to further evaluate our model in various VEs and contexts such as virtual training or games.

Concerning the gaze-based visual effects, it would be interesting to improve the luminance adaptation method in order to take into account the whole environment surrounding the user. This could be particularly useful for CAVE-like display. Existing 3D stereo rendering algorithms could also take into account the gaze position to compute a better convergence point to make this technology more appealing to users.

After gesture-based interaction, the gaze feature could represent another advance in human-computer interaction for daily use. Nowadays, game developers are increasingly interested in emerging technologies to improve the gamer experience. We strongly believe that the gaze feature could define a new area in the field of game design for novel gaze-based visual effects, interaction protocols, or gameplay.

References

[1] D.P. Luebke and B. Hallen, "Perceptually-Driven Simplification for Interactive Rendering," *Proc. Eurographics Workshop Rendering Techniques,* pp. 223-234, 2001.

[2] S. Hillaire, A. Lécuyer, R. Cozot, and G. Casiez, "Using an Eye-Tracking System to Improve Camera Motions and Depth-of-Field Blur Effects in Virtual Environments," *Proc. IEEE Virtual Reality Conf.,* pp. 47-50, 2008.

[3] A. Glenstrup and T. Engell-Nielsen, "Eye Controlled Media: Present and Future State," master's thesis, 1995.

[4] L. Itti, C. Koch, and E. Niebur, "A Model of Saliency-based Visual Attention for Rapid Scene Analysis," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 20, no. 11, pp. 1254-1259, Nov. 1998.

[5] L. Itti, "Quantifying the Contribution of Low-Level Saliency to Human Eye Movements in Dynamic Scenes," *Visual Cognition,* vol. 12, pp. 1093-1123, 2005.

[6] S. Lee, G. Kim, and S. Choi, "Real-Time Tracking of Visually Attended Objects in Virtual Environments and Its Application to LOD," *IEEE Trans. Visualization and Computer Graphics,* vol. 15, no. 1, pp. 6-19, Jan./Feb. 2009.

[7] S. Hillaire, A. Lécuyer, G. Breton, and T. Regia-Corte, "Gaze Behavior and Visual Attention Model when Turning in Virtual Environments," *Proc. ACM Symp. Virtual Reality Software and Technology,* pp. 43-50, 2009.

[8] K. Cater, A. Chalmers, and G. Ward, "Detail to Attention: Exploiting Visual Tasks for Selective Rendering," *Proc. 14th Eurographics Workshop Rendering,* pp. 270-280, 2003.

[9] A.M. Treisman and G. Gelade, "A Feature-Integration Theory of Attention," *Cognitive Psychology,* vol. 12, pp. 97-136, 1980.

[10] D. Yarbus, *Eye Motion and Vision.* Plenum Press, 1967.

[11] P. Longhurst, K. Debattista, and A. Chalmers, "A GPU Based Saliency Map for High-Fidelity Selective Rendering," *Proc. Fourth Int'l Conf. Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa,* pp. 21-29, 2006.

[12] V. Navalpakkam and L. Itti, "Modeling the Influence of Task on Attention," *Vision Research,* vol. 45, no. 2, pp. 205-231, 2005.

[13] V. Sundstedt, E. Stavrakis, M. Wimmer, and E. Reinhard, "A Psychophysical Study of Fixation Behavior in a Computer Game," *Proc. Fifth Symp. Applied Perception in Graphics and Visualization,* pp. 43-50, 2008.

[14] C. Sears and Z. Pylyshyn, "Multiple Object Tracking and Attentional Processing," *J. Experimental Psychology,* vol. 54, no. 1, pp. 1-14, 2000.

[15] A.R. Robertson, "Historical Development of CIE Recommended Color Difference Equations," *Color Research and Application,* vol. 15, no. 3, pp. 167-170, 1990.

[16] H. Chen and X. Liu, "Lighting and Material of Halo 3," *Proc. SIGGRAPH '08 Classes,* 2008.

[17] S. Hillaire, A. Lécuyer, R. Cozot, and G. Casiez, "Depth-of-Field Blur Effects for First-Person Navigation in Virtual Environments," *IEEE Computer Graphics and Applications,* vol. 28, no. 6, pp. 47-55, Nov./Dec. 2008.

[18] D.A. Robinson, "The Mechanics of Human Smooth Pursuit Eye Movement," *J. Physiology,* vol. 180, pp. 569-591, 1965.

[19] S. Hillaire, G. Breton, N. Ouarti, R. Cozot, and A. Lécuyer, "Using a Visual Attention Model to Improve Gaze Tracking Systems in Interactive 3D Applications," *Computer Graphics Forum,* vol. 29, pp. 47-55, 2010.

[20] M. McGuire and E. Enderton, "Colored Stochastic Shadow Maps," *Proc. ACM Symp. Interactive 3D Graphics and Games,* 2011.

[21] F. Meinl, "Crytek Sponza Atrium," http://www.crytek.com/cryengine/cryengine3/downloads, 2011.

[22] L. Terziman, A. Lécuyer, S. Hillaire, and J. Wiener, "Can Camera Motions Improve the Perception of Traveled Distance in Virtual Environments?," *Proc. IEEE Virtual Reality Conf.*, 2009.

[23] M. Moehring, A. Gloystein, and R. Doerner, "Issues with Virtual Space Perception within Reaching Distance: Mitigating Adverse Effects on Applications Using hmds in the Automotive Industry," *Proc. IEEE Virtual Reality Conf.*, pp. 223-226, 2009.

[24] E. Reinhard, M. Stark, P. Shirley, and J. Ferwerda, "Photographic Tone Reproduction for Digital Images," *Proc. 29th Ann. Conf. Computer Graphics and Interactive Techniques*, pp. 267-276, 2002.

Sébastien Hillaire received the PhD degree in computer science from the French National Institute of Applied Science (INSA) in 2010. He is a R&D graphic engineer at Dynamixyz, Rennes, France. His main research interests include virtual reality, visual attention models, and real-time high-quality rendering.

Anatole Lécuyer received the PhD degree in computer science from the University of Paris XI in 2001. He is a senior researcher at the French National Research Institute for Computer Science and Control (INRIA) in Rennes, France. His main research interests include virtual reality, 3D interaction, haptic interaction, and brain-computer interfaces. He is an associate editor of the *ACM Transactions on Applied Perception* and *the International Journal for Human-Computer Studies*.

Tony Regia-Corte received the PhD degree in cognitive psychology from the University of Lille III in 2006. He is a postdoctoral researcher at the French National Research Institute for Computer Science and Control (INRIA) in Rennes, France. His main research interests include perception of affordances, visual and haptic perception, and virtual reality.

Rémi Cozot received the PhD degree in computer science from the University of Rennes 1. He is an assistant professor at the University of Rennes 1 and a member of the FRVSense team at IRISA, Rennes, France. His main research interests include enhancement of real-time rendering using human vision features and color appearance models.

Jérôme Royan received the PhD degree in computer science from the University of Rennes I in 2001. He is a research engineer at Orange Labs in Rennes, France. His main research interests include Web3D technology, 3D streaming, network architecture for virtual environments, interoperability for virtual worlds, 3D visualization, and interaction of data. He was general chair of the ACM Web3D 2011 conference.

Gaspard Breton received the PhD degree in the field of 3D real time facial animation in 2002. He joined Orange Labs in 2002, where he has lead researches around embodied conversational agents and 3D animation such as lip synchronization, emotional display, motion capture, and photorealistic rendering. In 2010, he founded Dynamixyz, a startup specializing in high-quality 3D facial analysis and synthesis.