Real-Time Tracking of Visually Attended Objects in Interactive Virtual Environments

Sungkil Lee*
Haptics and Virtual Reality Laboratory
POSTECH

Gerard Jounghyun Kim†
Digital eXPerience Laboratory
Korea University

Seungmoon Choi‡
Haptics and Virtual Reality Laboratory
POSTECH

Abstract

This paper presents a real-time framework for computationally tracking objects visually attended by the user while navigating in interactive virtual environments. In addition to the conventional bottom-up (stimulus-driven) features, the framework also uses top-down (goal-directed) contexts to predict the human gaze. The framework first builds feature maps using preattentive features such as luminance, hue, depth, size, and motion. The feature maps are then integrated into a single saliency map using the center-surround difference operation. This pixel-level bottom-up saliency map is converted to an object-level saliency map using the item buffer. Finally, the top-down contexts are inferred from the user's spatial and temporal behaviors during interactive navigation and used to select the most plausibly attended object among candidates produced in the object saliency map. The computational framework was implemented using the GPU and exhibited extremely fast computing performance (5.68 msec for a 256×256 saliency map), substantiating its adequacy for interactive virtual environments. A user experiment was also conducted to evaluate the prediction accuracy of the visual attention tracking framework with respect to actual human gaze data. The attained accuracy level was well supported by the theory of human cognition for visually identifying a single and multiple attentive targets, especially due to the addition of top-down contextual information. The framework can be effectively used for perceptually based rendering without employing an expensive eye tracker, such as providing the depth-of-field effects and managing the level-of-detail in virtual environments.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Virtual Reality; I.2.0 [Artificial Intelligence]: General—Cognitive Simulation

Keywords: Visual Attention, Saliency Map, Attention Tracking, Bottom-Up Feature, Top-Down Context, Virtual Environment

1 Introduction

Knowing the region or objects where a user attends to in an image provides a number of advantages for creating an effective interactive virtual environment. For example, once a visually focused region or objects are identified, its rendering fidelity can be selectively controlled for lower computational costs or better perceptual effects. This simple idea has resulted in a number of useful computer graphics or virtual reality (VR) techniques, including level-of-detail (LOD) management [Cater et al. 2002; Brown et al. 2003], mesh simplification [Lee et al. 2005], information culling in distributed virtual environments [Beeharee et al. 2003], peripheral degradation (or foveation) [Watson et al. 1997], and global illumination [Yee et al. 2001; Longhurst et al. 2006].

A straightforward way to locate a visually attended region or objects in an image is to use an eye tracking device. However, such devices are still very expensive and uncomfortable to wear, and require a cumbersome calibration procedure. Moreover, most tracking devices do not allow its wearer to move and thus are not appropriate to many VR applications. A prominent alternative is to apply principles learned in human visual perception to computationally estimate a region or objects where the user might be looking and focusing at.

It is well known that human visual attention operates by balancing between automatic capture resulted from *bottom-up* (*stimulus-driven*) salient stimuli and volitional shifts guided by *top-down* (*goal-directed*) factors [Henderson 2003; Connor et al. 2004]. Color, intensity, and motion in images are the common examples of bottom-up stimuli [Treisman and Gelade 1980], and prior knowledge, memories, and goals are those of top-down factors [Loftus and Mackworth 1978; Wolfe and Jeremy 1993]. Among the numerous bottom-up attention models that have been conceived in the visual perception literature, the feature integration theory of Treisman and her colleagues [Treisman and Gelade 1980] is one of the most widely recognized and accepted. In this theory, an area in an image is called visually *salient* when it stands out relative to its surrounding neighborhood, and a 2D map that represents salient regions or objects of the image is called the *saliency map*. It has been also verified that saliency of a region is determined by integrating the low-level features such as color, intensity, and motion.

Inspired by the feature integration theory, several computational models have been proposed to estimate visual saliency in an image [Koch and Ullman 1985; Culhane and Tsotsos 1992; Mozer and Sitton 1998; Itti et al. 1998; Backer et al. 2001]. In particular, the model of Itti et al. [Itti et al. 1998] is relatively simple and has been used more frequently in practice. It first computes individual contrast maps for bottom-up features using the center-surround difference[1] and then integrates them into a final saliency map using the Winner-Take-All network [Koch and Ullman 1985]. This framework has been applied and extended to various applications including object recognition [Rutishauser et al. 2004], video compression [Ouerhani et al. 2001], and video summarization [Ma et al. 2005].

Once salient regions are determined by the preattentive bottom-up features, top-down factors guide a user to select one region to visu-

*e-mail: yskill@postech.ac.kr
†e-mail:gjkim@korea.ac.kr
‡e-mail:choism@postech.ac.kr

[1] Given an image, a center-surround difference refers to a difference between coarser and finer images, both generated from the original image using image pyramids [Itti et al. 1998] or the difference-of-Gaussian filter [Jobson et al. 1997].

ally focus on. Unlike the bottom-up features, this top-down guidance is task-dependent [Wolfe and Jeremy 1993; Henderson 2003]. For example, if red and green balls are in sight under no other specific conditions, a user is likely to look at either one (the red and green are opponent channels in the visual cortex [Engel et al. 1997], and thus they have comparable bottom-up saliency levels). However, a particular task (e.g., focus on a red region) given to the user would modulate the probability, fixing the user's attention to the red ball. Such task dependence has been a limiting factor for including the effect of top-down factors in estimating visual attention. Cater et al. introduced the importance factor of objects relative to a task in computing saliency [Cater et al. 2002], but the factor values had to be set manually. Features such as object distance from a viewer, image coverage [Haber et al. 2001], and novelty [Yee et al. 2001; Longhurst et al. 2006] have also been treated as top-down factors. However, these factors generally do not reflect the user's intention, and thus are more appropriate to be classified as bottom-up features. To the best of our knowledge, no major attempts have been made to systematically incorporate the effect of user intention for predicting attention.

Confirming whether a computational model for visual attention agrees to the actual human attention is crucial to the usability of the model in applications. Previously, Ouerhani et al. compared the bottom-up saliency model of Itti et al. [Itti et al. 1998] to actual human attention using an eye tracker and confirmed the existence of correlated regions in images [Ouerhani et al. 2004]. Santella et al. applied the bottom-up saliency model to non-photorealistic scenes and compared its prediction performance with that found by an eye tracker [Santella and DeCarlo 2004]. Compared to the abundant theories and applications of saliency maps, efforts toward their formal validation have been rather insufficient.

In this paper, we present a real-time computational framework for tracking visual attention in dynamic and interactive virtual environments. In dynamic virtual environments, the conventional computational saliency models have not been employed for human gaze estimation, although such models can be used for more perceptually based rendering such as automatic foveation and depth-of-field effects. Four reasons that are mainly responsible for that are discussed in the following, along with our resolutions for each.

First, applications in dynamic virtual environments require temporally coherent saliency tracking of "objects" rather than that of pixels. Early cognitive studies revealed that visual attention is object-based rather than location-based, and it moves with the objects [O'Craven et al. 1999; Sears and Pylyshyn 2000], which inspired the idea of attended object tracking. However, most saliency computation models only look for pixel-based regions, and cannot guarantee the coherence of object saliency tracking during interactive simulation. Our computational framework allows the estimation of visual attention levels of objects in a three-dimensional (3D) virtual environment and also provides continuous and smooth tracking of attended objects over time.

Second, the region of interest derived from a saliency map based on conventional bottom-up features does not reflect a user's volitional factors. This may result in incorrect attention prediction, since the top-down factors play a vital role in visual attention as reported in the previous literature on visual perception [Loftus and Mackworth 1978; Wolfe and Jeremy 1993; Henderson 2003]. Our attention tracking framework employs top-down contexts for the user's spatial and temporal intention inferred from the user's behavior during navigation in a dynamic virtual environment.

Third, the computational cost for generating a saliency map has been relatively expensive. To resolve this issue, Longhurst et al. [Longhurst et al. 2006] suggested using the Graphics Processing Unit (GPU) to accelerate computation. Our work, which has been carried out independently from that of Longhurst et al., has followed a similar but improved approach by using the hardware mipmap generation of the GPU. The implemented attention tracking framework shows a remarkable real-time performance and is fast enough to be used for interactive virtual environments.

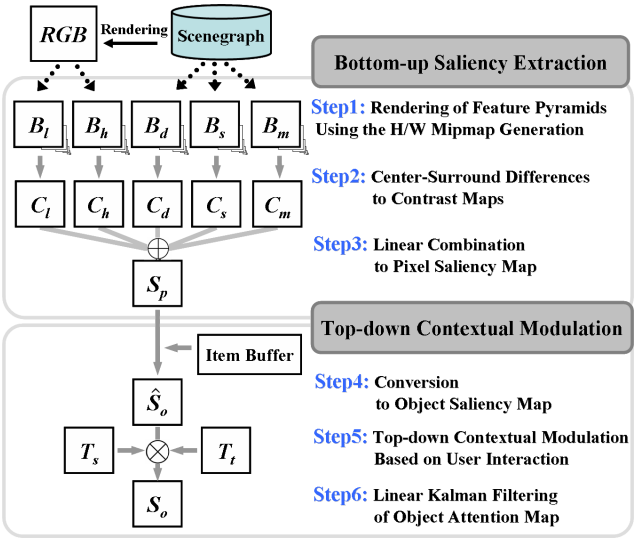

Figure 1: *The overall procedure of the proposed real-time visual attention tracking.*

Finally, the accuracy of visual attention estimation needs to be proved for such computational methods to be used in real applications, but this has been rather neglected in the literature, as discussed earlier. We conducted a user experiment using an eye tracker to evaluate the accuracy of our visual attention tracking framework and to validate the relative contributions of the bottom-up and top-down factors toward visual attention tracking.

The rest of this paper is laid out as follows. Section 2 introduces the overall structure of our framework for visual attention tracking in interactive virtual environments. Section 3 then explains how a bottom-up saliency map is constructed using the GPU in real-time. Section 4 presents the object-level tracking for attentive objects modulated by top-down contexts. Section 5 describes the implementation details and reports its computational performance. In Section 6, the accuracy of attention prediction is experimentally confirmed. Finally, Section 7 concludes this article along with a plan for future work.

2 Framework Overview

The overall procedural flow of our framework for visual attention tracking is summarized in Figure 1. As shown in the figure, the framework consists of largely two parts, one for building a bottom-up saliency map (upper block in the figure) and the other for modulating the saliency map using top-down contexts (lower block in the figure). Basically, our bottom-up saliency map extends that of [Itti et al. 1998] using two image features (luminance and hue) and three 3D dynamic features (depth, object size, and object motion). With 3D geometries, simulation models, and a RGB image rendered from the models, feature maps for luminance, hue, depth, size, and motion are generated as image pyramids (step 1 in Figure 1). Then, the image pyramids are converted by the center-surround difference to build contrast maps (or called the conspicuity maps in [Itti et al. 1998]) for each feature (step 2 in Figure 1), which

indicates regions with abrupt changes of pixel values in the corresponding feature map. This procedure is computed in real-time using GPU fragment shader. Finally, the contrast maps are linearly combined into a single saliency map, which represents the saliency of each pixel founded from the bottom-up features (step 3 in Figure 1).

The pixel saliency map is converted to an object saliency map, such that pixels corresponding to an object have the same saliency value (step 4 in Figure 1). The relevance of an object to a given task is also incorporated in this step. Then, top-down contextual factors (e.g., spatial and temporal) are computed, and the object saliency map is modulated with the contexts (step 5 in Figure 1). The spatial context refers to the effect of the spatial layout of objects in predicting the observer's attention (more effective for short-term tasks), and the temporal context refers to the changing trend of the spatial context associated with long-term goals. Finally, the map is linearly filtered for each object using the Kalman filter for smooth attention tracking (step 6 in Figure 1). Detailed explanation for each computational step follows in the next sections.

3 Real-time Bottom-Up Saliency Map

This section presents the details of our computational framework for building a bottom-up saliency map in real-time. Inputs to the framework are an input image, 3D object models, and a simulation model, all of which are commonly required in an interactive virtual environment.

3.1 Bottom-Up Features and their Image Pyramids

An initial step of building a saliency map is to construct feature maps using the five low-level features: luminance, hue, depth, size, and motion. Earlier neurological studies showed that all these features are preattentive (see [Treisman and Gelade 1980] for luminance, [Nagy and Sanchez 1990] for hue, [Enns 1990] for depth, [Treisman and Gelade 1980] for size, and [Nakayama and Silverman 1986] for motion). Feature values can be specified either for each pixel (luminance, hue, and depth) or for each object (size and motion).

The first two feature maps for luminance (B_l) and hue (B_h) are taken from the luminance and hue components in the HLS (Hue-Luminance-Saturation) color model converted from the original RGB image. Each pixel value for the depth map (B_d) is obtained from the z-buffer and normalized as:

$$B_d = \frac{z - z_{near}}{z_{far} - z_{near}}, \tag{1}$$

where z, z_{near}, and z_{far} are the 3D, near, and far clipping depths, respectively.

A feature map for size (B_s) is defined in the object-level considering the image coverage of an object. For object k, pixel values belonging to the object are set to:

$$B_s(k) = \frac{\text{number of pixels belonging to object } k}{\text{total number of pixels in the image}}. \tag{2}$$

This method has a merit when an object size is larger than the view volume or an object is partially culled by the viewport. The number of pixels corresponding to each object is counted using the item buffer [Weghorst et al. 1984].

A motion feature map (B_m) represents the moving velocity of an object obtained from the difference of 3D positions at consecutive simulation frames τ and $\tau - 1$. It is computed for object k as:

$$B_m(k) = \|\mathbf{p}^{\tau}(k) - \mathbf{p}^{\tau-1}(k)\|, \tag{3}$$

where $\mathbf{p}$ denotes the position of a vertex that is in the outermost position from the object center. This allows to simultaneously consider both translation and self-rotation of the object.

Each rendered feature map is successively down-sampled and converted into a set of lower resolution images, forming image pyramids from the finest original map (level 0) to the coarsest (level 6). These image pyramids are used for the center-surround difference operation to build contrast maps in the next step. Note that the processes are executed in a single batch using the hardware mipmap generation capability in the graphics hardware for real-time computation (see Section 5 for more implementation details).

3.2 Pixel-Level Saliency Map

The five feature maps are converted to local contrast (or conspicuity) maps, C_l, C_h, C_d, C_s, and C_m, via the multi-scale center-surround difference [Itti et al. 1998], an operation that detects locations standing out from their surroundings. The center-surround difference for a contrast map, C_f ($f \in \{l, h, d, s, m\}$), is calculated as:

$$C_f = \frac{1}{6} \sum_{i \in \{0,1,2\}} \sum_{j \in \{3,4\}} |B_f^i - B_f^{i+j}|, \tag{4}$$

where B_f^i and B_f^{i+j} represent feature maps at pyramid level i (finer) and $i+j$ (coarser), respectively. This operation is quite effective at finding contrasts [Itti et al. 1998], and widely used for bottom-up saliency map computation.

The contrast maps are merged into a single topographical saliency map, $\bar{S_p}$, by linearly combining them as:

$$\bar{S_p} = \sum_{f \in \{l,h,d,s,m\}} w_f C_f, \tag{5}$$

where w_f's are linear combination weights. Even though various schemes can be used for determining the weights [Itti et al. 1998; Itti 2000], most of them are not suitable for real-time use. In our framework, the weights are simply set to:

$$w_f = \frac{1}{\max_{(u,v)} C_f(u,v)}, \tag{6}$$

where (u, v) represents a pixel position in C_f. Finally, $\bar{S_p}$ is normalized to S_p (the pixel-level saliency map), such that each pixel value in S_p varies in $[0, 1]$.

4 Modulation with Top-Down Contexts

As discussed earlier, top-down contextual information can improve visual attention tracking by providing additional criteria for selecting objects among visually salient candidates found from the bottom-up features. With navigation in a virtual environment as a primary task, we have modeled a few high-level contexts (i.e., task-related object importance and spatial and temporal contexts) and included them in our computational framework for improved and plausible attention estimation.

4.1 Object-Level Attention Map

We first convert the pixel-level saliency map to an object-level saliency map ($\hat{S_o}$). As noted in [O'Craven et al. 1999; Sears and Pylyshyn 2000], an object-level saliency map is more appropriate for tracking and applications in virtual environments, and this is carried out by averaging pixel values belonging to each object as:

$$\hat{S_o}(k) = T_i(k) \frac{1}{n(k)} \sum_{(u,v) \in \text{ object } k} S_p(u,v), \tag{7}$$

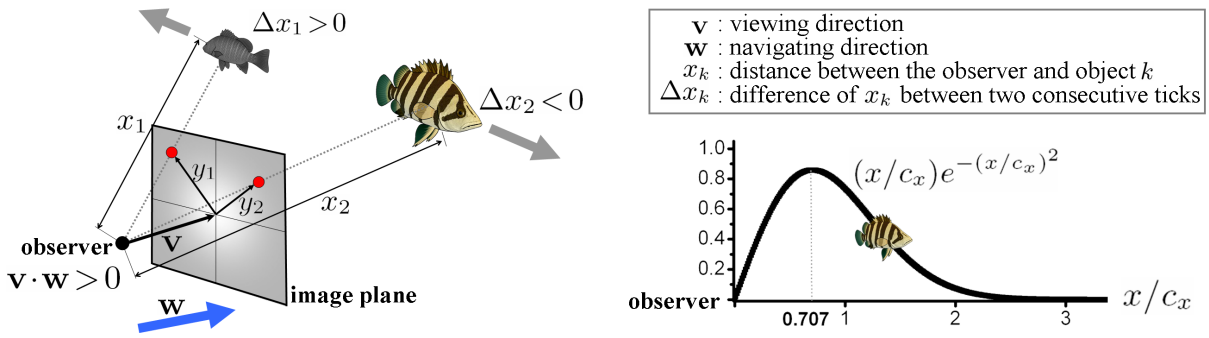

Figure 2: *An example of the spatial context model. An observer is currently moving toward the target object ($\mathbf{v} \cdot \mathbf{w} > 0$). The left fish moving to the opposite direction ($\Delta x > 0$) of navigation is disregarded as a top-down context. The other fish is under the influence of the distance emphasis model (Weibull curve in the right bottom) and the screen emphasis model (the 2D Gaussian weights on the image plane).*

where $n(k)$ is the number of pixels belonging to object k, (u, v) is a pixel position, and $T_i(k)$ is the user-defined task-related importance of object k. The pixels that are a part of object k are determined using the item buffer [Weghorst et al. 1984]. The task-related importance is added to exclude unimportant background objects (e.g., wall, floor, sky, and seawater) from consideration.

The object saliency map is further elaborated with spatial and temporal contexts (T_s and T_t) that are used to infer the user's intention during interactive navigation. The models of the spatial and temporal contexts are described in the next following sections. Once the spatial and temporal contexts are determined, the final object attention map, $S_o(k)$, is computed as:

$$S_o(k) = (T_s(k) + T_t(k))\hat{S}_o(k). \tag{8}$$

Note that $S_o(k)$ values may be temporally unstable due to the bilinear magnification in the texture lookup used in the center-surround difference operation. This is a generally evoked problem when the texture magnification is used. We stabilize the attention levels of objects using a linear Kalman filter [Welsh and Bishop 1995].

4.2 Spatial Context Based on User Motion

Typical models of spatial perception such as the topological 3-stage model (landmarks, route, and survey knowledge) require a hierarchical cognitive map of objects [Siegel and White 1975; Kuipers 1978]. These representations are related to supervised long-term goals rather than immediate responses of perception-action. Such cognition models are usually too complicated to be used in practice for predicting a user's high level behavior.

Assuming that landmarks (e.g., foreground objects) exist in a virtual environment, one effective way to estimate the gross area of a user's interest without any cognitive map is to find the moving direction of the user's egocentric view based on atomic interaction data. Hinted by earlier findings of Vishton et al. [Vishton and Cutting 1995], we modeled three spatial behaviors of an observer who navigates in a virtual environment from the landmark objects.

Let x be a normalized distance from the observer to an object in the 3D scene, y be a normalized distance between the center of a screen and the object in the screen coordinates, $\mathbf{v}$ be the viewing direction of the observer, and $\mathbf{w}$ be the moving direction of the observer (see Figure 2 for conceptual illustrations). $\Delta x = x^\tau - x^{\tau-1}$ is the difference of x between two consecutive simulation frames, where $\tau - 1$ and τ are corresponding time indices. First, we note that observers tend to situate themselves so that they can see objects in the center of a screen during navigation, and thus objects far from the screen center are not likely to be attended to. A modulation factor for this behavioral pattern is expressed as an exponential decay of the distance between the screen and object centers: $e^{-c_y y^2}$, where c_y is a scaling constant. Second, observers tend to maintain a proper distance from objects of interest, which requires modulating the spatial emphasis of objects according to the distance from an observer. Our modulation factor for this pattern follows the Weibull distribution: $(x/c_x)e^{-(x/c_x)^2}$, where $c_x = L/0.707$ is a scaling constant that is determined from a desired distance (L) of maximum spatial emphasis. Third, when observers move toward the target object ($\mathbf{v} \cdot \mathbf{w} > 0$; see also Figure 2), they usually approach objects that they want to see, and thus objects becoming more distant from the observer ($\Delta x > 0$) are very unlikely to receive any attention from the observers. Combining the three modulation factors, the spatial context model for object k is defined as:

$$T_s(k) = \begin{cases} 0 & \text{if } \mathbf{v}\cdot\mathbf{w} > 0 \text{ and } \Delta x > 0 \\ c_s(\frac{x}{c_x})e^{-\left(\frac{x}{c_x}\right)^2}e^{-c_y y^2} & \text{otherwise} \end{cases}, \tag{9}$$

where c_s represents a normalization constant to keep $T_s(k)$ within $[0, 0.5]$. Figure 2 conceptually illustrates our spatial context model.

4.3 Temporal Context

High spatial context values observed for an object during a certain time period implies that the object has been followed by the user. Considering this temporal property is useful for estimating the long-term intention of the user whereas an immediate task is mediated by the spatial context. To reflect this pattern, we define temporal context, $T_t(k)$, for object k using a running average of the spatial contexts as:

$$T_t(k) = \frac{1}{\lambda}\sum_{\tau=\tau_0-\lambda+1}^{\tau_0} {T_s}^{\tau}(k), \tag{10}$$

where λ denotes a time duration for controlling the long-term interest, and τ_0 and T_s^τ are the current simulation frame and the spatial context at τ, respectively. An additional role of the temporal context is to compensate for the spatial context values that may be directly affected due to unintended control errors by utilizing the history of the spatial context.

5 Implementation Details and Computational Performance

The proposed framework for real-time visual attention tracking was implemented using the OpenGL Shading Language (GLSL,

[Kessenich et al. 2004]) and the OpenSceneGraph [Burns and Osfield 2006] on a 3.2 GHz Pentium 4 PC with a GeForce 7900GTX graphics card. In the preprocessing stage, objects represented in 3D models were segmented in a scenegraph, and a unique ID was assigned to each object. Task-related object importance (T_i) was also pre-assigned to each object, particularly to prune off background objects from the attention map.

During run-time, the computational procedures shown in Figure 1, accelerated using the GPU for real-time performance, are applied at every simulation frame. The computation steps consist of four stages: (1) updating the object-level features and top-down (spatial and temporal) contexts, (2) building a pixel saliency map using the GPU acceleration, (3) converting the pixel saliency map into an object saliency map and modulating it with the top-down contexts, and (4) storing the result to an object attention list and filtering it using the linear Kalman Filter [Welsh and Bishop 1995].

In the first stage, the simulation engine updates the object-level features (the motion and size) and the spatial and temporal contexts based on the data recorded in the previous simulation frames. The spatial context for each object is computed using the object position and the viewing matrix from user interaction. The computed spatial context is inserted into a queue that stores the spatial contexts of previous simulation frames, and their running average, i.e., temporal context, is computed using the queue.

In the second stage, the graphics engine renders bottom-up feature maps and contrast maps using the GPU to construct a unified pixel-level saliency map. First, a typical RGB image is rendered from the 3D virtual object models. To obtain a contrast map for one feature, we need to build multi-scale pyramids for the feature and compute the center-surround difference using the pyramids. This incurs the most computational load in the framework, because magnification of a coarser map to a finer map is required for the pixel-by-pixel subtraction. We accelerate this process by building a mipmap texture for the finest image and take absolute differences between the mipmaps, since the fast texture lookup operation (enabled by the GPU) for a coarser scale corresponds to the magnification of an image. This idea significantly improves computational performance, allowing real-time construction of the pixel-level saliency map.

In actual implementation, three channels of information (usually for red, green, and blue) contained in one texture image are simultaneously processed in a fragment shader. We thus render two feature map textures for the five bottom-up features explained in Section 3.1, LH (B_l, B_h, $\cdot$) and DSM (B_d, B_s, B_m), and generate their mipmap pyramids using the hardware mipmap generation. The center-surround differences of the LH and DSM textures are computed in one step for each texture image, which provides five contrast maps required for the computation of the pixel-level saliency map. A sample fragment program in GLSL for the center-surround differences is shown in Figure 3. The pixel saliency map (S_p) is then built by linearly combining the two center-surrounded textures (contrast maps) with the normalization weights. The weight of a contrast map is the inverse of a maximum pixel value in the map.

The third stage is to convert the pixel-level saliency map into an object-level saliency map. For this, the correspondence between pixel position (u, v) and object k is found using the item buffer [Weghorst et al. 1984], which is an image where each pixel contains the ID of the object that the pixel belongs to. Using the ID image and the top-down contexts (T_i, T_s and T_t), the attention value of each object can be computed using Equations 7 and 8. For efficiency, the object-level attention map is stored into a linked-list.

Finally, the computed attention value of each object is smoothed and tracked using a discrete Kalman Filter [Welsh and Bishop 1995] based on the position and velocity state model.

```
uniform sampler2D DSM;         // texture sampler
#define tc0 gl_TexCoord[0].st  // texture coordinate

void main()
{
  vec3 c[7];
  for(int i=0; i<7; i++)
    c[i] = texture2D(DSM,tc0,float(i)).rgb;
  vec3 C = abs(c[0]-c[3])+abs(c[0]-c[4])+
           abs(c[1]-c[4])+abs(c[1]-c[5])+
           abs(c[2]-c[5])+abs(c[2]-c[6]);
  gl_FragColor = vec4(C/6.0,1.0);
}
```

Figure 3: *Sample fragment shader for the center-surround difference operation of the DSM texture image in the GLSL.*

Figure 4 shows examples of intermediate and final output images of our attention tracking framework applied to static (upper image) and dynamic environments (lower image). Object-level lists (S_o, T_s, and T_t) were represented in maps for illustration. The whitest object (the bowling pin in the upper image and the orange fish in the lower image) in each object attention map (S_o) is the most attentive object. In all these examples, a number of bottom-up candidates for attentive objects are first suggested with similar saliency levels in the bottom-up feature contrast maps. Thus, the pixel-level saliency maps (S_p) alone do not clearly indicate the most salient objects. With the aid of the top-down contexts (T_s and T_t), the most attentive objects are finally selected with much agreement to our visual attention (to be validated in the next section).

In order to compare the computational performance of our framework with a typical CPU-based method and the most recent GPU-based method [Longhurst et al. 2006], the same algorithms were implemented using the Intel OpenCV toolkit [OpenCV 2006] and GLSL [Kessenich et al. 2004]. To remove the effect of the polygonal model size and the number of objects, we excluded the time required to model rendering (i.e., RGB, depth, and item buffer images). The computation time for the top-down contexts was not included as well, since the top-down contexts are unique in our framework. Practically, the time required to compute the top-down contexts is negligible compared to that of the bottom-up saliency map (e.g., less than 0.3 msec for the virtual undersea model in Figure 4). The comparisons were performed for four saliency map sizes (64×64, 128×128, 256×256, and 512×512). Figure 5 shows the comparison results. The speed-up compared to Longhurst et al.'s method is from 1.57 times (in the 64×64 image) to 1.15 times (in the 512×512 image), and that compared to the CPU-based method is from 2.20 times (in the 64×64 image) to 6.27 times (in the 512×512 image). The real-time use of our attention tracking framework is possible with up to 256×256 saliency maps . Even if the model rendering times excluded in the computation are included, our framework can produce 256×256 saliency maps in real-time for 3D environments consisted of up to a million polygons (attention computation $\approx$ 5.68 msec, total 30 frames per sec.).

Very recently, Longhurst et al. reported a similar saliency map computation method that generated the Gaussian pyramids with multi-pass rendering and performed the center-surround difference in a GPU program [Longhurst et al. 2006]. Our work, which has been independently conducted from that of Longhurst et al., is much more comprehensive in the following aspects. First, our implementation using the hardware mipmap generation requires one-pass rendering, and is much more efficient than that of Longhurst et al. where multi-pass rendering is needed. Second, we use more comprehensive features, including top-down contexts that were not considered in the study of Longhurst et al. Third, our attention tracking is for objects, unlike the study of Longhurst et al. In pixel-level saliency maps, both saliency prediction techniques may suffer

Figure 4: *Example results of the visual attention tracking framework for static (upper) and dynamic (lower) scenes. All the images were rendered in real-time. In the static scene, the motion feature was not used. The whitest object (the bowling pin in the upper image and the orange fish in the lower image, respectively) is the most attentive object in the corresponding object attention map (S_o). The 3D Fish models are the courtesy of Toru Miyazawa at Toucan Co.*

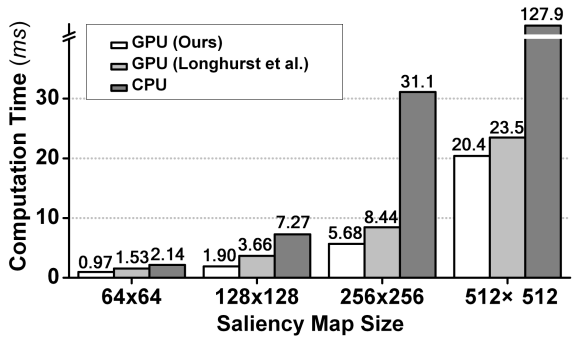

Figure 5: *Comparison of computation times for generating a saliency map between the CPU- and GPU-based (ours and Longhurst et al.'s) implementations.*

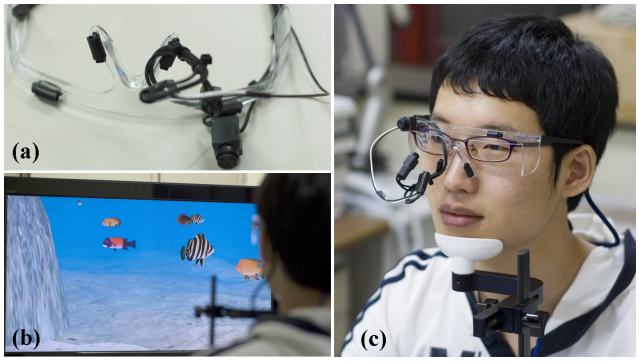

Figure 6: *A 42-inch LCD display and a monocular eye tracker (Arrington Research, Inc.) used in the experiment.*

from instability since the magnification by texture lookup are used in common. However, we largely suppress such instability in the object-level attention map. Finally, this article also includes the perceptual validation of our computational framework that will be described in the next section.

6 Experiment for Attention Estimation Accuracy

This section reports the design and results of a user experiment conducted to validate the accuracy of our attention tracking compared to the actual human gaze pattern.

6.1 Methods

A participant's eye movements were recorded using a monocular eye tracking device (Arrington Research Inc.; see Figure 6(a)) with a 60 Hz sampling rate and 640×240 video resolution. A 42-inch LCD display with a resolution of 1600×900 was used for the presentation of visual scenes (see also Figure 6(b)). The participant wearing an eye tracker was seated at approximately 1.5 m in front of the display in a lighted room, which allowed a wide field of view as well as precise eye tracking.

One dynamic and one static virtual environments were modeled and used in the experiment. The dynamic virtual environment modeled a virtual undersea (shown in the upper row of Figure 7) where 30 animated fishes (six kinds, and five fishes per each kind) were

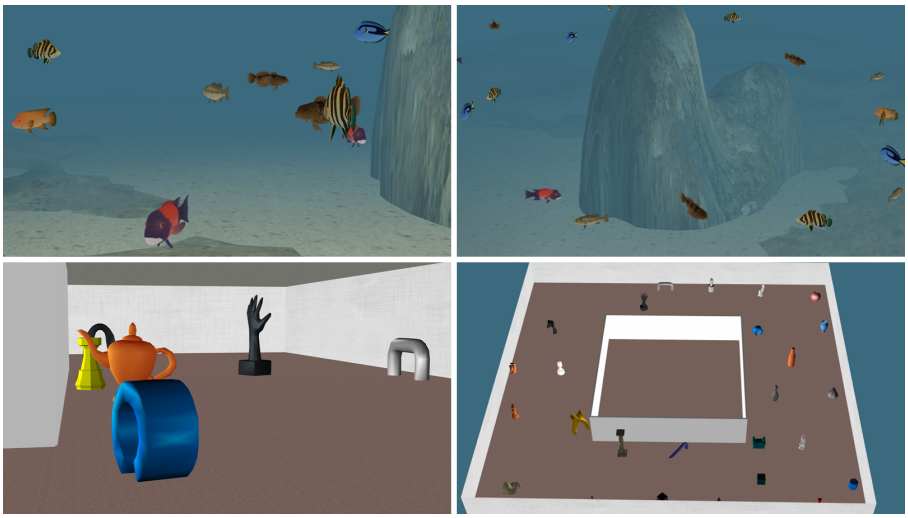

Figure 7: *The virtual undersea (dynamic; upper row) and virtual gallery (static; lower row) environments used in the experiment.*

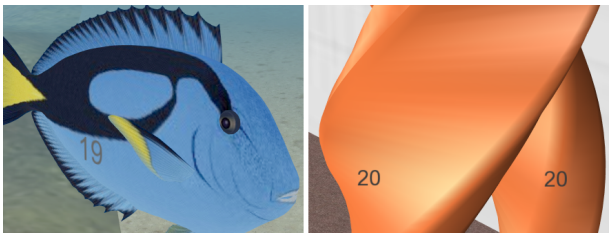

Figure 8: *Two examples of number-attached objects. For fishes in the virtual undersea, tags were attached to both sides of the abdomen, and for the virtual gallery, four tags around the middle height of an object.*

swimming around. The static virtual environment was a virtual art gallery shown in the lower row of Figure 7.

16 paid subjects (15 male and 1 female) participated in the experiment. Their ages varied between 18 and 36 years, with a mean of 25.1. All participants had normal or corrected-to-normal vision. They were assigned with two tasks, free navigation and visual search. For free navigation, the participants were asked to simply move around the virtual environments and see virtual objects using the keyboard for controlling navigation motion. For visual search, objects in the two virtual environments contained numbered tags on their body, and the participants were instructed to find objects that had specified numbers (see Figure 8 for examples). The numbers were 5, 15, 25, and 35 for objects in the undersea, and 7, 17, 27, and 37 for the gallery. The participants were asked to press the spacebar in the keyboard when they found objects with the specified numbers. This instruction was solely to keep the participants concentrated to the search task, and whether they indeed found the objects was irrelevant to the purpose of the experiment. With the two virtual environments and the two tasks, the experiment had a 2×2 within-subject design.

Before beginning an experimental session, a participant was briefed about the experimental procedure and went through a training session to learn the navigation scheme in the art galley model with no objects appeared. Three degrees-of-freedom control (forward/backward translation and turns for yaw and pitch) was used for navigation. Four (2×2) main sessions followed the training session, and their presentation order was balanced using the Latin squares. Each session started with calibration of the eye tracker to map screen-space points in 5×5 grids to eye-space coordinates. In order to find eye-space coordinates, oval-fitted centers of a pupil or a glint were extracted from an input video of the eye tracker. After the calibration, the participant was instructed not to move one's head, and this was facilitated with a chin rest (see Figure 6(c)). The task for each session was verbally explained to the participant in the beginning of the session. Each session lasted for three minutes, and the participant took a rest for a few minutes before starting another session.

6.2 Data Analysis

Per each experimental condition and each participant, 10,800 (60 pt./sec.$\times$180 sec.) screen positions stared at by a participant were measured with the eye tracker and recorded in terms of normalized screen-space coordinates ((0.0, 0.0) to (1.0, 1.0)). These data were classified into four categories (blink, saccadic, drift, and fixation) according to the aspect ratio of a fitted ellipse and eye movement velocity. If the aspect ratio of an ellipse filled to the eye was less than a threshold (0.7 for our data analysis), the eye movement was considered to be a blink. If not, the eye movement velocity was used for further classification. A frame that showed eye movement velocity slower than 0.03 was regarded as fixation, a frame with eye velocity faster than 0.1 as saccadic, and a frame with eye velocity in between as drift. Only fixation and drift were used for analysis.

Three quantitative measures (A_1, A_2, and A_3) were defined in order to assess the accuracy of our attention tracking framework. In each simulation frame, if an object that was found to be the most attentive and an object gazed by the participant were identical, this frame was regarded as a frame with correct attention estimation, and the total number of frames with correct attention estimation was counted. Note that during the counting, frames that showed only backgrounds were excluded, since no attentive objects were available in the participant's sight. The number of frames with correct attention estimation was divided by the total number of frames that contained foreground objects, and the result was stored into A_1. Accuracy where two (or three) most attentive objects were compared with the participant-stared object was also computed and denoted by A_2 (or A_3). Using the three measures, we evaluated the overall prediction capability of our attention tracking framework.

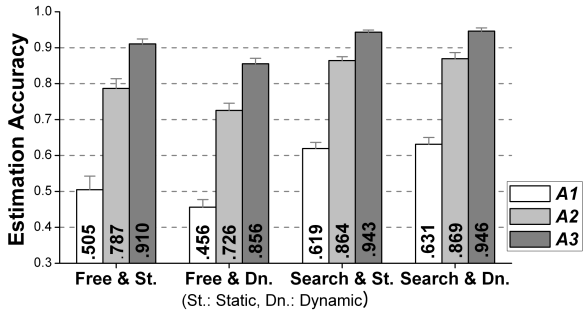

Figure 9: *Means and standard errors of the three estimation accuracies (A_1, A_2, and A_3).*

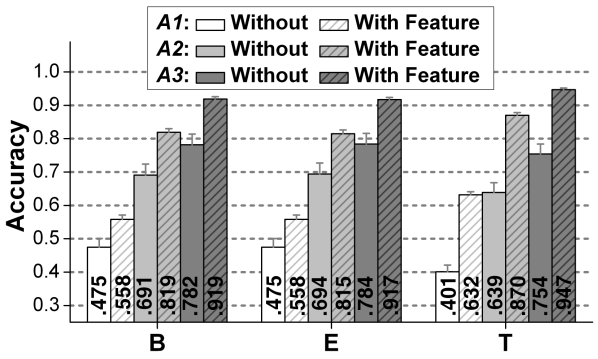

Figure 10: *Means and standard errors of the accuracies with and without B, E, and T.*

Table 1: *Source table of the within-subject 2×2×2 ANOVA for the effects of task (K) and environment (V).*

	A_1		A_2		A_3	
	$F_{1,15}$	p	$F_{1,15}$	p	$F_{1,15}$	p
K	34.75*	<.0001	30.64*	<.0001	22.80*	.0002
V	1.31	.2700	3.36	.0866	7.40*	.0158
K×V	3.59	.0776	6.93*	.0188	22.17*	.0003

Note: $^*p < .05$. This result was analyzed using SAS Proc GLM.

We also investigated the relative contributions of features to attention estimation. The features were classified into three groups: image features (B: $\{B_l, B_h\}$), extended 3D/object features (E: $\{B_d, B_s, B_m\}$), and top-down context (T: $\{T_s, T_t\}$). The image features were most typically used for saliency estimation, while the extended 3D/object features were the bottom-up features available only for 3D virtual environments. The top-down contexts are our new additions to attention tracking. To investigate the role of each feature group, attention maps generated with and without the feature group were generated, and their estimation accuracies were compared to each other. Thus, total eight (2^3) attention maps were generated and their estimation accuracies were compared using statistical analyses. Note that the *baseline* attention map, which corresponds to a map generated with no feature groups, cannot be clearly defined since such a map bears no information for attentive points or objects. However, ignoring the baseline case would yield an unbalanced missing cell design, which causes the interpretation of main factor effects to be unclear in ANOVA [Speed et al. 1978]. To avoid this problem, we simulated the baseline attention map by randomly choosing a pixel on the screen as the most attentive pixel.

6.3 Results

The measured attention estimation accuracies using the attention map generated with all components included are summarized in Figure 9. The accuracy averages for the four conditions are shown along with the standard errors as the error bars. Overall, the average accuracy was increased from 0.553 for A_1 (when only the most attentive object was compared to the actually attended object by the participant) to 0.914 for A_3 (when three most attentive objects were compared). The effects of the two independent variables, task (K) and environment (V), were analyzed via a two-way within-subject ANOVA for each accuracy measure, A_1, A_2, and A_3, and the results are summarized in Table 1 that shows all relevant statistics. The effect of task on all three accuracies was statistically significant with very small p-values. The visual search task yielded higher accuracies than the free navigation task with the overall mean difference of 0.096. The effect of environment on the estimation accuracies was rather weaker than that of task. It was statistically significant for A_3, but was not for A_1 and A_2. Even with A_3, the p-value (0.0158) was much larger than the corresponding p-value of task (0.002). The interaction effects between task and environment were statistically significant for A_2 and A_3, but not for A_1.

Table 2: *Means and standard errors of the estimation accuracies for the eight combinations of feature groups (B, E, and T). In the table, $B = 0$ means that the B group features (image features) were not used for attention map generation, and $B = 1$ that the features were used. E and T are to be interpreted similarly.*

B	E	T	A_1	A_2	A_3
0	0	0	.145(.005)	.259(.008)	.349(.010)
1	0	0	.484(.018)	.773(.019)	.892(.013)
0	1	0	.487(.019)	.762(.020)	.885(.013)
1	1	0	.485(.020)	.762(.020)	.891(.013)
0	0	1	.640(.019)	.873(.016)	.949(.009)
1	0	1	.631(.020)	.870(.017)	.947(.010)
0	1	1	.628(.019)	.867(.016)	.945(.010)
1	1	1	.631(.019)	.869(.017)	.946(.009)

The relative contributions of the three feature groups (B, E, and T) to attention prediction were examined by three-way ANOVA. Due to the limited space, we only report the results from the data collected with the visual search task in the dynamic environment (the virtual undersea) as a representative. The main factor means of B, E, and T are shown in Figure 10 for A_1, A_2, and A_3. For all feature groups, significant increases were observed, with the T group (top-down contexts) exhibiting the largest increase. This indicates that the top-down contexts played an instrumental role for correctly identifying visually attended objects. The means and standard errors for all eight combinations of feature groups are provided in Table 2. The accuracies except for the baseline varied from 0.484 (only B for A_1) to 0.949 (only T for A_3). We can also confirm from the table that the accuracies with and without the top-down contexts showed the largest differences. A 2×2×2 ANOVA carried out with B, E, and T as independent variables showed that all main and interaction effects were statistically significant ($p < .0001$) for all accuracies. Similar trends were also observed from the data of other experimental conditions.

6.4 Discussion

As discovered in earlier psychological studies [O'Craven et al. 1999; Awh and Pashler 2000; Sears and Pylyshyn 2000], human visual attention can be laid on multiple objects (or split foci) at the same time. Therefore, it can be inherently difficult to precisely define a single object that a person stares at in the first place. More-

over, since the eye tracking device has inevitable error (up to 3 inch at 42-inch screen), it was sometimes impossible to exactly pinpoint the gazed object when several objects were closely located on the screen. Considering all these facts, our attention tracking framework showed very promising results; the best accuracy was up to 0.64 for a single attentive candidate and 0.949 for three candidates. Our framework can be used to find a single candidate (e.g., for direct gaze estimation required in depth-of-field rendering that enhances visual realism of a scene and spatial and depth perception [Marshall et al. 1996; Mather 1997]) and multiple candidates (e.g., for gaze-contingent LOD management in real-time virtual environments and automatic generation of a camera path in large virtual environments).

As expected, the specific task (free navigation versus visual search) imposed on the participants brought statistically significant differences in attention estimation accuracies. During free navigation, the participants were often more interested in perceiving the spatial layout of an environment, rather than catching the details of objects. This must have degraded the overall attention estimation accuracies for the free navigation task. For visual search, the participants more focused on the objects rather than the overall environment layout. Furthermore, even when their attention was disturbed by abrupt appearance of objects that were highly salient in terms of the bottom-up features, their interest tended to immediately return to the currently pursuing object due to the long-term goal (i.e., visual search). Such behavior is well reflected in our top-down context models, enabling good performance for the visual search.

Among the three feature groups (B, E, and T), the top-down factors (T) were shown to be the most contributing to improving the accuracies of attention tracking, which brought an important insight for the use of a bottom-up saliency map. That is, even though attentive candidates can be somewhat estimated from a saliency map generated with bottom-up features only, correct prediction of one (or a few) most attentive objects is greatly benefited by considering the top-down contexts related to the intention of a user. It would be therefore necessary to categorize the common user behaviors in a virtual environment (e.g., navigation, manipulation, and selection) and develop appropriate high-level context models to accomplish more precise attention tracking.

There was one notable top-down behavior associated with the novelty of objects observed during the analysis of the users' gaze patterns. The first spatial context that emphasizes the objects in and to the center of the screen was shown to be quite effective. However, after finishing the task (e.g., reading the numbers on the objects), the attention of the participant shifted from the objects in the center of the screen to the boundaries (or sides) of the screen to find new objects to stare at, often moving to unexplored areas in the virtual environments using the navigation control keys. Adding this behavior to our spatial context model may significantly raise the accuracy of our predictive tracking. Another feature that is not included in our computational framework is the head movement of a user. Tracking the movement of the user's head may bring about the same improvement on the tracking accuracy, but this would require an additional device such as a camera. Note that in the present experiment, the user's head had to be fixed on the chin rest due to the restriction of the eye tracking device.

Our experiment also showed that a few features may have a key role in predicting attention. If such key factors are included in attention estimation, reasonably high performance is expected without using all features (as demonstrated in Table 2), saving the computational cost. Also note that bottom-up features may still be required for some virtual environments, because top-down information and object-level segmentation are not always possible (e.g., image-based virtual environment).

We also tested the effects of saliency map size on estimation accuracy. In general, using a small saliency map is expected to show difficulty in correctly selecting small objects as salient. Using the data collected for the visual search task in the virtual undersea, we tested four saliency map sizes (from 512×512 to 64×64) and computed the accuracies. The results showed very marginal differences below 1 %. This is likely due to the fact that the virtual environment used for the experiment had relatively large objects.

7 Conclusion and Future Work

We proposed a computational framework for tracking visually attended objects in interactive virtual environments, which utilizes the bottom-up preattentive features such as luminance, hue, depth, size, and motion and the newly added top-down contextual information such as spatial and temporal intention inferred from the user's navigation patterns. The framework was accelerated using the GPU fragment shader, and can render up to 256×256 saliency maps in real-time. The prediction accuracy of our framework was also evaluated in a user experiment, and the results confirmed the critical role of top-down contexts in attention prediction.

In the future, we will work on improving the spatial context model to include the novelty of object in consideration and developing adequate top-down context models for other common user tasks in interactive virtual environments. We will then apply the attention tracking framework to the depth-of-field rendering and the LOD management for virtual environments, preferably in more immersive virtual display systems such as the CAVE™.

Acknowledgements

The authors would like to thank Toru Miyazawa at Toucan Co. for granting the use of the 3D fish models and four anonymous reviewers for their thoughtful comments. This work was supported in parts by the Korea Science and Engineering Foundation (KOSEF) grant funded by the Korea government (MOST) (No. R01-2006-000-11142-0 and R01-2006-000-10808-0), by the Second Brain Korea 21 Program, and by a grant for the education of regional S/W engineers based on the global network from Daegu Digital Industry Promotion Agency.

References

AWH, E., AND PASHLER, H. 2000. Evidence for split attentional foci. *Journal of Experimental Psychology 26*, 2, 834–846.

BACKER, G., MERTSCHING, B., AND BOLLMANN, M. 2001. Data- and model-driven gaze control for an active-vision system. *IEEE Trans. Pattern Analysis and Machine Intelligence 23*, 12, 1415–1429.

BEEHAREE, A. K., WEST, A. J., AND HUBBOLD, R. J. 2003. Visual attention based information culling for distributed virtual environments. In *Proceedings of ACM Symposium on Virtual Reality Software and Technology*, 213–222.

BROWN, R., COOPER, L., AND PHAM, B. 2003. Visual attention-based polygon level of detail management. In *Proceedings of GRAPHITE 2003*, 55–62.

BURNS, D., AND OSFIELD, R., 2006. OpenSceneGraph. http://www.openscenegraph.org.

CATER, K., CHALMERS, A., AND LEDDA, P. 2002. Selective quality rendering by exploiting human inattentional blindness: looking but not seeing. In *Proceedings of ACM Symposium on Virtual Reality Software and Technology*, 17–24.

CONNOR, C., EGETH, H., AND YANTIS, S. 2004. Visual attention: Bottom-up vs. top-down. *Current Biology 14*, 19, 850–852.

CULHANE, S. M., AND TSOTSOS, J. K. 1992. An attentional prototype for early vision. In *Proceedings of European Conference on Computer Vision 1992*, 551–560.

ENGEL, S., ZHANG, X., AND WANDELL, B. 1997. Colour tuning in human visual cortex measured with functional magnetic resonance imaging. *Nature 388*, 6637, 68–71.

ENNS, J. T. 1990. Three-dimensional features that pop out in visual search. In *Visual Search*, Taylor and Francis, Eds. New York, 37–45.

HABER, J., MYSZKOWSKI, K., YAMAUCHI, H., AND SEIDEL, H.-P. 2001. Perceptually guided corrective splatting. *Computer Graphics Forum 20*, 3.

HENDERSON, J. M. 2003. Human gaze control during real-world scene perception. *Trends in Cognitive Sciences 7*, 11, 498–504.

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

ITTI, L. 2000. *Models of Bottom-Up and Top-Down Visual Attention*. PhD thesis, California Institute of Technology, Pasadena, California.

JOBSON, D. J., UR RAHMAN, Z., AND WOODELL, G. A. 1997. Properties and performance of a center/surround retinex. *IEEE Trans. on Image Processing 6*, 3, 451–462.

KALMAN, R. E. 1960. A new approach to linear filtering and predictive problems. *Trans. ASME, Journal of basic engineering 82*, 34–45.

KESSENICH, J., BALDWIN, D., AND ROST, R., 2004. The OpenGL Shading Language. Version 1.10.59. 3Dlabs, Inc. Ltd. http://developer.3dlabs.com/documents/index.htm.

KOCH, C., AND ULLMAN, S. 1985. Shifts in selective visual attention. *Human Neurobiology 4*, 219–227.

KUIPERS, B. 1978. Modeling spatial knowledge. *Cognitive Science 2*, 129–153.

LEE, C. H., VARSHNEY, A., AND JACOBS, D. W. 2005. Mesh saliency. In *Proceedings of SIGGRAPH 2005*, 659–666.

LOFTUS, G. R., AND MACKWORTH, N. H. 1978. Cognitive determinants of fixation duration during picture viewing. *Journal of Experimental Psychology 4*, 565–572.

LONGHURST, P., DEBATTISTA, K., AND CHALMERS, A. 2006. A GPU based saliency map for high-fidelity selective rendering. In *Proceedings of AFRIGRAPH 2006*, 21–29.

MA, Y.-F., HUA, X.-S., LU, L., AND ZHANG, H. 2005. A generic framework of user attention model and its application in video summarization. *IEEE Trans. on Multimedia 7*, 5, 907–919.

MARSHALL, J., BURBECK, C., ARIELY, D., ROLLAND, J., AND MARTIN, K. 1996. Occlusion edge blur: a cue to relative visual depth. *Journal of the Optical Society of America 13*, 681–688.

MATHER, G. 1997. The use of image blur as a depth cue. *Perception 26*, 1147–1158.

MOZER, M. C., AND SITTON, M. 1998. Computational modeling of spatial attention. In *Attention*, H. Pashler, Ed. UCL Press, London, 341–393.

NAGY, A. L., AND SANCHEZ, R. R. 1990. Critical color differences determined with a visual search task. *Journal of the Optical Society of America 7*, 7, 1209–1217.

NAKAYAMA, K., AND SILVERMAN, G. 1986. Serial and parallel processing of visual feature conjunctions. *Nature 320*, 264–265.

O'CRAVEN, K. M., DOWNING, P. E., AND KANWISHER, N. 1999. fMRI evidence for objects as the units of attentional selection. *Nature 401*, 6753, 584–587.

OPENCV, 2006. http://sourceforge.net/projects/opencvlibrary/.

OUERHANI, N., BRACAMONTE, J., HUGLI, H., ANSORGE, M., AND PELLANDINI, F. 2001. Adaptive color image compression based on visual attention. In *Proceedings of ICIAP*, 416–421.

OUERHANI, N., VON WARTBURG, R., AND HUGLI, H. 2004. Empirical validation of the saliency-based model of visual attention. *Electronic Letters on Computer Vision and Image Analysis 3*, 1, 13–24.

RUTISHAUSER, U., WALTHER, D., KOCH, C., AND PERONA, P. 2004. Is bottom-up attention useful for object recognition? In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition 2004*, 37–44.

SANTELLA, A., AND DECARLO, D. 2004. Visual interest and NPR: an evaluation and manifesto. In *Proceedings of Symposium on Non-Photorealistic Animation and Rendering*, 71–150.

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

SIEGEL, A. W., AND WHITE, S. H. 1975. The development of spatial representations of large-scale environments. In *Advances in Child Development and Behavior*, H. Reese, Ed., vol. 10. Academic Press, New York, 10–55.

SPEED, F. M., HOCKING, R. R., AND HACKNEY, O. P. 1978. Methods of analysis of linear models with unbalanced data. *Journal of the American Statistical Association 73*, 361, 105–112.

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

VISHTON, P., AND CUTTING, J. 1995. Wayfinding, displacements, and mental maps: velocity field are not typically used to determine one's aimpoint. *Journal of Experimental Psychology 21*, 978–995.

WATSON, B., WALKER, N., HODGES, L. F., AND WORDEN, A. 1997. Managing level of detail through peripheral degradation: Effects on search performance with a head-mounted display. *ACM Trans. on Computer-Human Interaction 4*, 4, 323–346.

WEGHORST, H., HOOPER, G., AND GREENBERG, D. P. 1984. Improved computational methods for ray tracing. *ACM Trans. on Graphics 3*, 1, 52–69.

WELSH, G., AND BISHOP, G. 1995. An introduction to the Kalman filter. Tech. Rep. 95-041, Univ. of North Carolina at Chapel Hill.

WOLFE, AND JEREMY, M. 1993. Guided search 2.0. In *Proceedings of the Human Factors and Ergonomics Society 37th Annual Meeting*, 1295–1299.

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