Using Visual Attention Models and Saliency Maps to improve Gaze Tracking in Interactive 3D Applications

Category: Research and Application

ABSTRACT

This paper introduces the use of visual attention models and saliency maps to improve the accuracy of gaze tracking systems.

Firstly, we propose a low-cost gaze tracking system using Artificial Neural Network (ANN) and a web cam. We propose a new way to present data to the ANN and compare our system to existing ANN-based gaze tracking systems and other accurate gaze tracking systems.

Secondly, we propose to improve the accuracy of gaze tracking system using a visual attention model. The visual attention model simulates the human visual system, defining a saliency map for the image, i.e., giving an attention weight to every pixel of the image. Our algorithm uses an uncertainty window, defined by the gaze tracker accuracy, and located around the gaze point given by the tracker. Then, it searches for the most salient points, or objects, located inside this uncertainty window, and determines a novel and, hopefully, better gaze point. Finally, we present the results of an experiment conducted to assess the performance of our approach.

The whole system can be used as a real-time gaze tracking system in many interactive 3D applications such as video games, virtual reality applications, etc. The use of a visual attention model can be adapted to any gaze tracker and the visual attention model can also be adapted to the application in which it is used.

Keywords: gaze-tracking, artificial neural network, visual attention model, saliency map, human-computer interaction

Index Terms: H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Artificial, augmented, and virtual realities; K.7.m [Information Interfaces and Presentation]: User Interfaces—Interaction styles, User-centered design

1 INTRODUCTION

Gaze trackers are systems used to compute the gaze position of a human [8]. A majority of gaze trackers are designed to compute the gaze position onto a flat screen. Since their creation in the late 19th century, before the computer existed, these systems have come a long way [8]. They have become more and more accurate and less cumbersome. Moreover, the interest in these systems has grown thanks to their usefulness in several domains: from human studies in psychology to VR systems, e.g. to accelerate the rendering process, or as an aid for disabled people.

Many gaze estimation methods have already been proposed. However, many of them suffer from their complex calibration procedures, their poor usability, their intrusivity [13], their cost [20] or their cumbersomeness [3]. These systems are often accurate but, for the reasons aforementioned, cannot be sold on the mass market for daily use. Today, it would be valuable to have a low-cost eye-tracking system usable without needing a high expertise and in various conditions. For example, such a system could be enter onto the mass market of consoles or could be used with a web cam connected to a PC. They could be valuable in interactive 3D applications such as video games, virtual reality applications, etc.

A new kind of gaze-trackers has recently emerged: remote gaze-tracker. Remote gaze tracking systems are qualified as "systems that operate without contact with the user and permit free head movement within reasonable limits without losing tracking" [4]. These systems generally use only a camera or web cam at a low resolution. However, they are generally less accurate than previous gaze trackers. Therefore, a lot of research is still going on to improve remote gaze tracking systems.

In this paper, we present a novel way to use visual attention models in order to improve the accuracy of any gaze tracking system such as remote gaze trackers. For this aim, we first propose a low-cost gaze tracking system based on a web cam and an artificial neural network (ANN). Then, we describe an algorithm to combine a gaze tracker with a visual attention model in order to improve the overall accuracy. It uses an uncertainty window, which size is defined by the accuracy of the gaze tracker, to search for more coherent gaze position using a saliency map encoding visually salient area. It is computed using a visual attention model.

In the remainder of this paper, after exposing related work, we detail the algorithm and the architecture we propose to compute, in real time, user's gaze position using a web cam and an artificial neural network. The accuracy and usability of this system is discussed. In a second part, a new way to use visual attention models is presented to improve the accuracy of the gaze tracker. Finally, we report on an experiment conducted to evaluate the accuracy of the proposed method. The paper ends with a general discussion and conclusion.

2 RELATED WORK

In the last decade many gaze-tracking systems have been developed for various applications, such as for virtual reality and interactive 3D applications. Table 1 summarizes the existing gaze tracking systems, putting an emphasis on the required hardware and their current accuracy.

Intrusive systems are generally restrictive for users who have to wear heavy and uncomfortable equipment. As an example, Kaufman et al. [13] use electrooculography to measure eyes muscular activity. This method requires the user to wear electrodes on their head. Knowing this activity, they can evaluate the orientation of the eyes and compute the gaze position with an accuracy of 1.5 to 2 degrees. Another technique requires the user to wear induction coil contact lens [8]. The gaze direction can be computed by measuring the high-frequency electro-magnetic fields produced by these lens. Both these techniques require user's head to stay still. To overcome this problem, Duchowski et al. [7] propose an helmet with an embedded 600×450 screen for each eye. Two gaze trackers, one for each eye, are used. The tracker use the corneal reflection of infra-red light sources to detect the pupil and use its shape to compute the view direction and the gaze point. Furthermore, this system is able to compute a 3D gaze point using the vergence phenomenon of both tracked eyes. Intrusive systems are precise enough to be interesting for a research purpose, however, it would be better to have non-intrusive gaze tracking systems. As shown in Table 1, few gaze tracking systems are intrusive and current trend is toward the development of non intrusive systems.

Non-intrusive gaze trackers allow users to feel more free because they do not wear any device and can move their head. For this aim, Beymers and Flickner [3] propose a multi camera system. The users head is first tracked using a camera with a wide field of view. The users face being tracked, a high resolution narrow camera is steered in the direction of one eye of the user. Finally, a 3D representation of the eye is used with the infra-red light glint position to evalu-

Category	Reference	Hardware	Intrusive	Horizontal accuracy (degree)	Vertical accuracy (degree)	Limitations
Intrusive trackers	Kaufman *et al.* [13]	electrodes	yes	1.5 to 2	1.5 to 2	intrusive
	Duchowski *et al.* [7]	helmet with two screens	yes	0.3	0.3	intrusive and expensive
Trackers based on specific hardware	Beymers *et al.* [3]	one wide and one narrow steerable camera	no	0.6	0.6	expensive and cumbersome
	Tobii [20]	dedicated capture system	no	0.5	0.5	expensive
Remote gaze trackers	Yoo *et al.* [24]	infra-red LED and CCD camera	no	1.0	0.8	user must remain between 30 to 40cm to the screen
	Hennessey *et al.* [10]	infra-red LED and CCD camera	no	1.0	1.0	infra-red light
	Guestrin *et al.* [9]	two light and one CCD camera	no	0.9	0.9	cumbersome (use of two lights)
	Yamazoe *et al.* [22]	CCD camera	no	5.0	7.0	low accuracy
ANN-based gaze trackers	Baluja *et al.* [2]	640×480 CCD camera	no	1.5	1.5	non robust calibration
	Piratla *et al.* [18]	640×480 webcam	yes	not available	not available	non robust calibration
Attended objects trackers	Lee *et al.* [15]	no hardware	no	object based	object based	highly dependent on the VE and user's task

Table 1: Summary of existing gaze tracking systems.

ate user's gaze position on screen with an accuracy of 0.6 degrees. Tobii technology [20] proposes a non-intrusive gaze tracking system which allows moderate movement of user's head. The Tobii system uses expensive dedicated tracking devices but has an accuracy of 0.5 degrees. This system uses infra-red lights. However, implementation details are not available. Table 1 shows that these non-intrusive systems are very accurate but require high expertise, they are cumbersome [3] or very expensive [20].

Recently, a lot of research has been conducted on remote eye-tracking systems. These systems are designed to be used in every day life by non-expert users with a simple and fast calibration process. Some of the proposed systems [10] [24] still require infra-red LED but are able to achieve an accuracy of one degree. The calibration sequence of the system proposed by Hennessey et al. [10] only requires the user to look at five points. The gaze tracker of Guestrin and Eizenman [9] only requires two normal light sources and one CCD camera to compute the gaze position with an accuracy of 0.9 degrees in the best case under free head movement. When only one light is used, the system still works if the users' head remains at a constant position. All the presented remote gaze trackers use a 3D representation of the eye to compute the gaze direction using reflection glint. The system developed by Yamazoe et al. [22] is able to compute the gaze position without infra-red light nor calibration sequence. This system is aimed for everyday use since it uses a single video camera. It has a low accuracy of 5 degrees horizontally and 7 degrees vertically. The results found are promising and they could be improved.

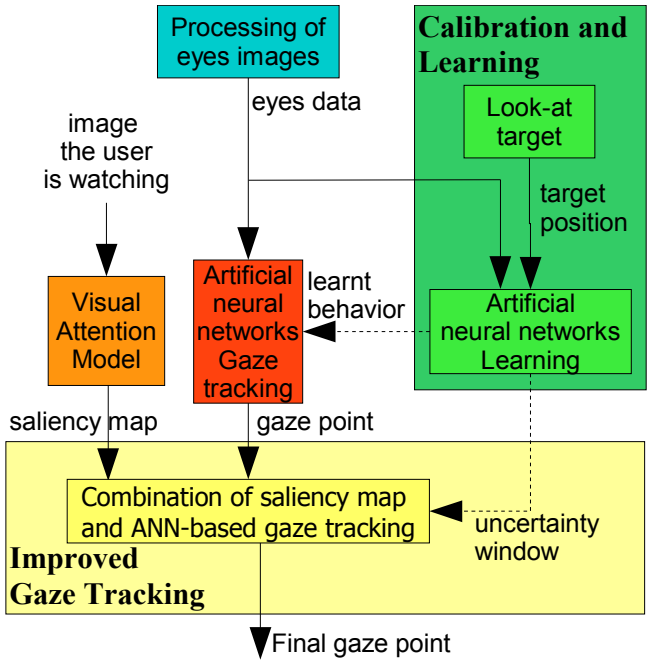

Figure 1: Global architecture.

3 Artificial Neural Network Based Gaze Tracker

We propose a low cost ANN-based gaze tracking system using a single web cam. This system is designed to compute the gaze point of a user onto a flat screen.

In this section, previous ANN-based gaze trackers are described. We then expose hardware requirements as well as the software architecture of our ANN-based gaze tracker (red, green and blue boxes of the global architecture presented in Figure 1). We detail the calibration sequence, i.e. how the ANN is trained, and real time use. Finally we report on a experience conducted to measure the accuracy of this system.

3.1 Existing gaze trackers based on Artificial Neural Network

Few gaze tracking systems based on ANN have been proposed in the literature. Baluja and Pomerleau [2] proposed to directly send an image of one eye of the user to an ANN. This image was scaled to a size of 15×15. Their system is able to achieve an accuracy of 1.5 degree. Piralta and Jayasumana [18] used a different approach. They computed some features describing current user state using the video stream and vision algorithms such as geometric pattern tracking, color detection, etc. These features were position of markers, position on eyes center, distance between upper and lower

eyelid, etc. They represent the input of the ANN.

Both systems only need a 640×480 web cam and represent the screen as a two dimensional grid. In addition, the gaze tracker proposed by Piralta and Jayasumana [18] is intrusive since it requires the user to wear a yellow and black stripped bar stick on their head.

3.2 Our approach for ANN-based Gaze Tracking

Compared to previous gaze trackers based on ANN, we propose to transform the captured images of user's eyes in an new data format. Left and right intersection of the bottom and top eyelid of each eye must be manually selected by the user in the video recorded by the web cam. We obtain two points per eye we use to extract the image of each eye. During this procedure, the head is maintained at a constant position using a chin-rest.

Each time we receive a frame from the web cam, the images of user's eyes are extracted from the video stream and scaled to images of width W_e and height H_e. Contrarily to Baluja and Pomerleau [2] who directly send the picture of the eye to the ANN, we propose to transform it to reduce the number of input of the ANN. First, we apply a contrast-limited adaptive histogram equalization filter to both images, previously transformed from RGB format to intensity format, in order to maximize their contrast. Then, for each eye image, pixels of each column and each row are added using Equation 1 to obtain two arrays S_x (of size W_e) and S_y (of size H_e) from each eye image.

$$\forall i \in [1, W_e], S_x[i] = \sum_{j=1}^{H_e} eyeImage[i][j]$$

$$\forall j \in [1, H_e], S_y[j] = \sum_{i=1}^{W_e} eyeImage[i][j] \quad (1)$$

Finally, for each eye, S_x and S_y have their values mapped from their range $[min\ value, max\ value]$ to the range $[0,1]$ and this result is stored in S'_x and S'_y arrays. This mapping is important because it allows us to take advantage of the full working range of each neurone activation function. This function is a linear activation function which works in the $[0,1]$ range.

The arrays S'_x and S'_y for each eye are then sent to the ANN. Actually, we use two ANN per eye : one which computes the horizontal position of the gaze point based on S'_x and another one to compute the vertical position of the gaze point based on S'_y. After preliminary testing, we found that using the S'_x and S'_y arrays as input of separate ANN produces smoother estimations of the gaze position. We also found that $W_e = 40$ pixels and $H_e = 20$ pixels were adapted size for the scaled image of the eyes given the resolution of the web cam and learning capabilities of the ANN. Moreover, each ANN is composed of three layers with twenty neurones in each hidden layer. Using this architecture, our algorithm is able to evaluate continuous gaze position on the screen contrary to previous ANN-based gaze trackers which represent the screen as a 2D grid [2][18].

3.3 Calibration sequence and gaze tracking

In order to have the system computing user's gaze position in real time, we need to calibrate it. During this sequence, each ANN will be trained to compute one coordinate of the gaze point based on its associated eye image.

The calibration sequence trains the ANN of the gaze tracking system to compute the gaze point position based on the arrays S'_x and S'_y of each eye. For this aim, the system tells the user to follow a target which moves onto the whole screen area. The target moves slowly in order to reduce the latency due to the refresh rate of the web cam. We consider (x_t, y_t) the normalized screen coordinate of the target ranging from 0, i.e. bottom and left of the screen, to 1, i.e. top and right of the screen. At each frame, we add to the training set S'_x and S'_y computed in real time for the left and right eyes, and the corresponding gaze position (x_t, y_t) on screen, i.e. the target position on screen. Finally, each ANN is trained using the retro-propagation algorithm.

After the end of the ANN training, the real-time gaze tracking sequence is initiated. As explained before, the gaze tracker computes a gaze position on the screen for each eye. The final gaze position is computed as the mean of the two resulting gaze positions: this produces smoother gaze movements.

3.4 Environment and hardware setup

The ANN-based gaze tracker we propose only requires one web cam supporting 640×480 video capture. This system is designed to compute the user's gaze position on a flat screen.

The user's head is expected to remain within the range of 40 to 80 cm in front of the screen as illustrated in Figure 2. Furthermore, the system we propose works better when the height of the user is at the level of the center of the screen. For a better performance, we recommend to position the web cam under the screen and not over. In this case, eyes are more visible as they are not hidden by dense upper eyelashes. Currently, the system requires the user's head to stay at a constant position and orientation.

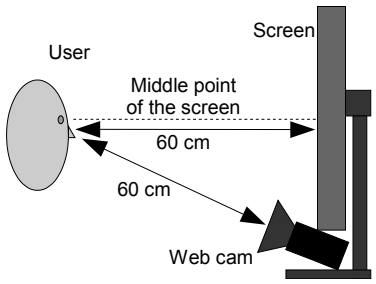

Figure 2: Hardware setup.

3.5 Accuracy

We assessed the accuracy of the ANN gaze tracking system we propose by conducting an evaluation with 6 naïve participants.

3.5.1 Procedure

Six people participated in this evaluation. During the test, they were positioned in front of a flat $19'$ screen at a resolution of 1280×1024. They were at a distance of 60 cm from the screen and the web cam, and no sound was played. We used the ANN gaze tracking system presented in this paper to compute, in real-time, the participants' gaze position. Their head and eyes were maintained at a constant position using a chin-rest.

For each participant, after the calibration sequence, the experiment consisted in successively looking at nine white targets, each one during 3 seconds. We recorded the participants' gaze position on the screen, computed using the ANN gaze tracker together with the current real target position.

3.5.2 Results and Discussion

To assess the accuracy of the ANN-based gaze tracker, we computed the difference between participants gaze points measured by the gaze tracker and real targets' positions on the screen. These differences were the distances between the gaze points and targets on the horizontal and vertical axes. Then, using these errors together with users' distance from the screen and screen size, we can compute the average accuracy in degrees. During this sequence, we do not take into account the first 100 milliseconds after each target switch in order to ignore errors due to saccades.

Participants	Horizontal accuracy in degree	Vertical accuracy in degree
1	0.777	1.048
2	1.684	1.449
3	1.300	1.050
4	1.780	0.872
5	1.814	1.452
6	1.499	0.937
Mean	1.476	1.135
Standard Deviation	0.392	0.254

Table 2: Horizontal and vertical accuracy of the ANN-based gaze tracker.

The global mean accuracy, as well as the accuracy for each participant, of our system are shown in Table 2. We found that the ANN gaze tracking system has a mean accuracy of 1.48° on the horizontal axis and 1.14° on the vertical axis. This accuracy is highly dependent on the user's eyes shape. For small and almost closed eyes, the accuracy can decrease to 1.81° whereas for users with wide open eyes, it can increase to 0.78°.

Our system requires a comfortable amount of ambient light to have enough contrast in the captured images of the user's eyes. Moreover, it requires the user's head to be maintained at a constant position although previous ANN gaze trackers support head movements [2] [18]. However, the calibration sequence of these systems are highly dependent of users which were forced to move their head. We could improve our system by taking into account the eyes position in the video and yaw, pitch and roll angles of the head similarly to [18].

The accuracy of our ANN-based gaze tracking system seems sufficient for the user to achieve various tasks in several environments such as operating systems desktop or 3D virtual environments. However, this system could be improved by taking advantage of the characteristics of the human visual system. This is addressed in the following section and it is considered as the main contribution of our approach.

4 Using visual attention models to improve gaze tracking

We propose to improve the accuracy of gaze tracking systems by using a visual attention model.

In this section, we describe the human visual system and previous work on visual attention models. Then, we detail an algorithm using a visual attention model and a saliency map to improve the accuracy of any gaze tracker. Finally we report on a experience conducted to measure the advantages of using this method.

4.1 Visual attention models

Visual attention represents the capacity of a human to focalize on a visual object of a scene. The brain does not have the capacity to analyze a whole scene in one time. Thus, it uses some viewing strategies to quickly analyze a scene [12] [11]. It is well known that visual attention is composed of two components: *bottom-up* and *top-down*.

The bottom-up component simulates visual reflexes of the human visual system. Due to the structure of our brain and the fact that we only accurately perceive our environment on 2 degrees of our visual field [1], the human visual system does not have the capabilities of analyzing a whole scene in parallel. Instead, it analyzes a scene sequentially, i.e. area by area. As an example, when someone first looks at a scene, his/her gaze is first unconsciously attracted by visually salient areas to rapidly perceive the most important area of that scene [12]. Several visually salient features have been identified in previous research [21] [12]: red/green and blue/yellow antagonistic colors, intensities, orientations, etc. Inspired by the feature integration theory [21], bottom-up visual attention models have been developed to compute a saliency map from an image [12] (for details on how to compute a saliency map, refer to section 4.3). When a human looks at a picture without any task to do, the saliency value of each pixel of the saliency map represents its attractiveness, i.e. the more the saliency of an area is high the more a human will look at this area. Other features have progressively been added in the computation of saliency maps such as flickering [11], depth [15] or motion [15].

Visual attention is not only controlled by reflexes resulting from visual stimuli, but also by the cognitive process that takes place in the brain, i.e. the top-down component. It is involved in the strategies we use to analyze a scene. For example, Yarbus [23] has shown that the way people look at pictures strongly depends on the task they have to achieve. Moreover, The top-down component is subject to the habituation phenomenon [16], i.e. objects become familiar over time, and oblivion [17]. Several models have been proposed to simulate the top-down component using task-map [5], habituation [16], memory [17] or spatial context [15].

Nowadays, visual attention models are used in various domains for several tasks. For example, they are used to accelerate the computation of global illumination of virtual environments [5] [16], for dynamic avatar animation [6], smart mesh decimation [14], etc.

4.2 General approach

The main idea of our approach consists in looking for salient (i.e., most important) pixels/objects located near the point given by the gaze tracker and consider that the user is probably looking at these pixels/objects. The whole approach thus consists in considering two phases: (1) a global phase in which we compute the point given by the (ANN-based) gaze tracker corresponding to the raw estimation of the gaze point; and (2) a second phase in which we refine this gaze point by searching for the most salient point that is closest to it, corresponding to the precise/final estimation of the gaze point.

Therefore, the method we propose to improve gaze tracking systems exploits characteristics of the bottom-up component of the human visual system. The global architecture of our algorithm is shown on Figure 1. The method is based on a saliency map computed using a bottom-up visual attention model. Our algorithm is then composed of two parts. In the first part we compute the gaze point from the ANN-based tracker together with the saliency map of the current image the user is looking at. In the second part, we use this saliency map in order to refine the accuracy of the gaze tracking system.

4.3 Computing the saliency map

To compute the saliency map, we use the bottom-up visual attention model presented on Figure 3. It is inspired by Itti *et al.* [12], however, to reduce the computation time, it is implemented on GPU hardware using shaders.

Firstly, from the 3D virtual environment image rendered from the current point of view, we compute four *feature* maps. Originally, Itti et al. [12] also used four feature maps: red/green and blue/yellow antagonistic colors, intensities and orientations. In this model, antagonistic colors were computed using simple color differences. Lee *et al.* [15] improved this computation by using with the Hue value of the Hue-Luminance-Saturation color space. In our case, we propose to use the *Lab* color space which takes into account the human visual system [19]. In this color space, relative differences between colors are "almost perceptually correct". Moreover, this color space has the advantage of directly encoding red/green and blue/yellow antagonistic colors as well as intensity, i.e. respectively the a, b and L components. They correspond to

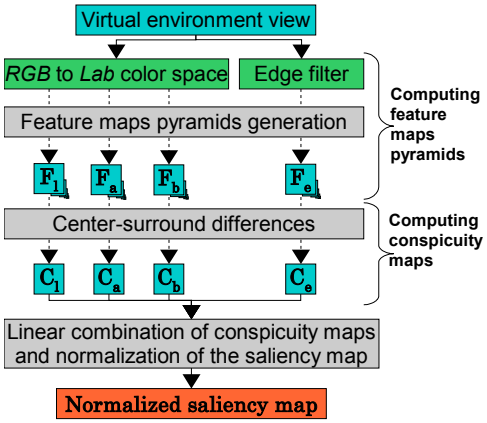

Figure 3: Algorithm used to compute the saliency map.

F_a, F_b and F_l feature maps on Figure 3. In Itti *et al.* [12], another feature map was computed based on orientations in the image using a Gabor filter. This filter is expensive to compute so we propose to use an edge filter as in Longhurst *et al.* [16]. It results in the feature map F_e. These feature maps are directly computed in real-time on the GPU using a shader and stored in a single four-component texture. Finally, feature maps pyramids, containing the original feature maps and several down sampled copies at lower resolution, are generated using the automatic mipmap generation capacity of GPU as proposed by Lee *et al.*[15].

Secondly, the feature maps need to be converted in *conspicuity* maps using the multiscale Center-Surround difference operator as in [12]. This operator aims at simulating the response of brain neurones which receive stimuli from the visual receptive fields. Originally, it needs a dyadic Gaussian pyramid of feature map [12]. In our case, we use the same approach as Lee *et al.* [15] which consists in using the mipmap pyramid to reduce computation time. The conspicuity maps, i.e. C_l, C_a, C_b and C_e on Figure 3 , are finally computed using Equation 2 with i and $i+j$ being mipmap pyramid levels. The level i is a fine level and $i+j$ a coarse level of the pyramid.

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

Finally, the normalized saliency map is computed by a linear combination of the four conspicuity maps as in Lee *et al.*[15] using Equation 3. In our case, we use $w_l = w_a = w_b = 1.0$ and $w_e = 0.8$ due to the fact that edge conspicuity map values are often higher than others conspicuity map values. In the end, the saliency map is normalized using operator $\mathcal{N}$.

$$S = \mathcal{N}\left(\sum_{x\in\{l,a,b,e\}} w_x \times C_x\right) \quad (3)$$

To normalize the saliency map, we need to know the maximum value in the saliency map. We do not iteratively search for the maximum value in the entire saliency map using the CPU because it would be expensive to compute. Instead, we compute the maximum by recursively downsampling the saliency map by a factor of two until we reach the size of one texel which finally contains the maximum value. In this algorithm, at each step, and for each pixel of the coarser level, a fragment program computes the maximum value of the saliency map's four corresponding pixels of the finer level computed at the preceding step. Once we have obtained the maximum value, we normalize the saliency map in a final rendering pass on the GPU.

As a result, using our algorithm, the saliency map is computed in real-time using GPU hardware. It takes 22 ms for our algorithm to compute a 512×512 normalized saliency map. To sum up with, our algorithm combines advantages of Itti *et al.* [12], Longhurst *et al.* [16], i.e. orientation approximation by an edge filter, and Lee *et al.* [15], i.e. fast center-surround operator, bottom-up visual attention models. In addition, we have proposed to use the *Lab* color space to compute a more perceptually correct saliency map. We have also accelerated the normalization process of the saliency map by using a pyramid algorithm taking advantage of GPU hardware.

4.4 Final computation of the gaze position using a saliency map

We know the accuracy of the gaze tracking system and the distance of the user from the screen. Thus, we can compute the accuracy of the gaze tracking system in screen coordinates. We define the accuracy Acc_x on the x axis and Acc_y on the y axis in screen coordinates. From these values, we can define an uncertainty window Wu. The size of Wu are $Wu_x = w_s \times 2.0 \times Acc_x$ on the x axis and $Wu_y = w_s \times 2.0 \times Acc_y$ on the y axis, with w_s being a scale factor. Assuming that the user is gazing inside Wu, we propose to improve the gaze tracker accuracy by searching inside Wu for potentially more coherent, i.e. salient, gaze points.

Itti [11] have investigated the contribution of bottom-up saliency on human eye movements. He found that a majority of saccades were directed to a minority of highly salient area. His experiment showed that 72.3% of participants' gaze were directed at locations having $s > 0.25$, with s the maximum saliency of the normalized saliency map inside a disk of 5.6 degrees of diameter centered on the gaze point. As a result, he suggested that bottom-up saliency may provide a set of saccade locations and that the final gaze point is chosen according to a top-down process. In our case, we know in which area of the screen the user is gazing thanks to the gaze point estimated by the gaze tracker we described in Section 3. Thus, inversly to Itti [11], we propose to search in Wu for highly attractive, salient, position.

Based on Itti's work [11] our algorithm takes into account a saliency threshold S_t. It computes the novel gaze point position using the saliency map values close to the gaze position given by the gaze tracker. It first searches inside the uncertainty window for the most salient position sp in the normalized saliency map. If the saliency value of sp is superior to the threshold S_t, we set the final gaze point on sp. In the contrary, if it is inferior to S_t, we rely upon the gaze tracking system: the gaze point remains unchanged.

Following Itti's work [11], a good threshold value would be 0.25 [11]. This value can be adapted according to the application for which the tracker is used. For example, a S_t of 0 will always set the gaze point position on the most salient pixel inside Wu. In the experiment we present in section 5, we expose results for several threshold values and uncertainty window sizes.

In our model, we could have included a duration of fixation but Itti [11] has shown that it is not correlated to saliency values at the level of the gaze point. Moreover, to our best knowledge, no other research works have found a correlation between a saliency map and gaze duration. Instead, to avoid instantaneous jump between the point extimated by the gaze tracker alone and the gaze tracker improved by the saliency map, we apply a low pass filter to the final gaze position.

The use of this algorithm is illustrated on Figure 4. In this case, the gaze point estimated by the ANN-based gaze tracker is far from the one estimated by the Tobii. However, when the ANN-based gaze tracker is combined with a saliency map using our method, the final gaze point position is inside the Tobii zone. The Tobii zone is a window centered on the gaze point computed by the Tobii gaze tracker. The size of this zone is defined by both the accuracy of the Tobii system (0.5° [20]) and the distance of the user from the

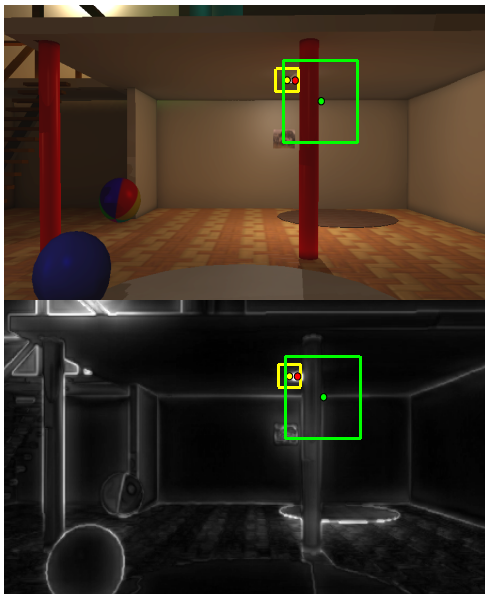

Figure 4: Combination of low-cost gaze tracking and saliency map to improve performance. Top: view of the scene, bottom: the corresponding saliency map. In yellow, gaze point and uncertainty window of the Tobii system (used as theoretical gaze information). In green, gaze point and uncertainty window of the low-cost gaze tracker. In red, the gaze point computed by combining low-cost gaze tracker and saliency map.

screen (60 cm).

5 Evaluation

We conducted an experiment to measure to which extent our algorithm can improve gaze tracking system during free navigation in 3D virtual environment. We recorded participants' gaze positions using three different approaches : (1) the ANN-based gaze tracker, (2) the ANN-based gaze tracker improved by the bottom-up visual attention model and (3) a Tobii gaze tracker which is used to convey the "theoretical" gaze position of the user.

5.1 Apparatus

During this experiment, we used the ANN gaze tracker described in section3 and the tobii x50 gaze tracker [20]. The ANN gaze tracker was used with our algorithm to improve its accuracy as described in Section 4. We tested the performance of our algorithm under different conditions, i.e., with different values of saliency threshold S_t and scale factor of the uncertainty window w_s.

Participants were positioned in front of a flat $19'$ screen at a resolution of 1280×1024. They were at a distance of 60 cm from the screen and the web cam, and no sound was played. Their head and eyes were maintained at a constant position using a chin-rest. The virtual environment was rendered in real-time at a constant frame-rate of 50Hz. It represented the interior of a house as shown in Figure 5.

5.2 Procedure

For each participant, the task consisted in visiting the 3D virtual environment freely. They navigated using a first-person navigation paradigm using a keyboard to control the advance direction on the horizontal plane or climb stairs, and the mouse to look around.

5 participants (4 males, 1 female) with a mean age of 26 (SD=3.2) participated in our experiment. They were all familiar with first-person navigation paradigm and had normal vision.

Figure 5: 3D virtual environment used for the experiment.

The experiment was divided in two parts. The first part consisted in the calibration of the Tobii and the ANN-based gaze tracking system. The training sequence of the ANN lasted 30 seconds. Then, the second part of the experiment began. During this part, participants were free to navigate in the 3D virtual environment dunring 1 minute.

During each session, positions and movements in the virtual environment and gaze positions were recorded as well as position and orientation of dynamic objects . Thus, we were able to replay and analyze each session.

5.3 Results

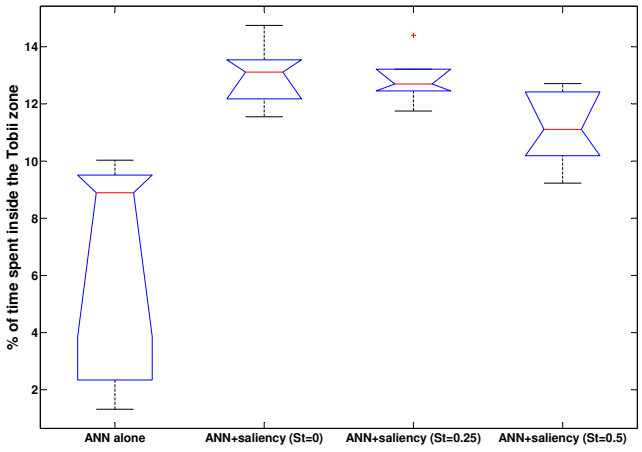

Figure 6: Time spent inside the Tobii window in different gaze-tracking conditions (using a window scale factor of 2).

During the experiment, we recorded participants' gaze position using the accurate Tobii x50 gaze tracking system. We also recorded the gaze position computed by the ANN-based gaze tracking system alone. Then, in a post processing step, we applied our method designed to improve gaze tracking by replaying the recorded sequences of each participant. Thus, we were able to test several parameters of our method. These parameters were the uncertainty window scale factor w_s with values $\{1, 1.5, 2, 2.5, 3, 4\}$ and the saliency threshold S_t with values $\{0, 0.25, 0.5\}$.

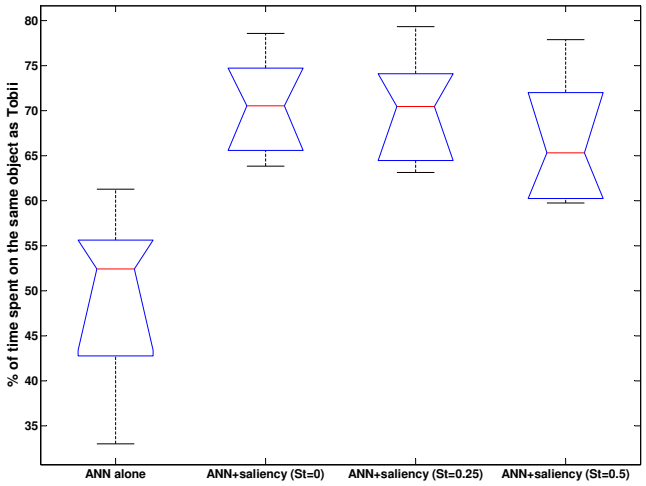

Figure 7: Time spent looking at the same virtual object as detected with the Tobii system in different gaze-tracking conditions (using a window scale factor of 2).

Window size (scale factor)	p value for % of time in Tobii zone	p value for % of time on same object
1.0	0.336	0.050
1.5	0.133	0.014*
2.0	0.005*	0.013*
2.5	0.110	0.026*
3.0	0.084	0.066
4.0	0.925	0.0501

Table 3: P value of non-parametric ANOVA conducted on performance indicators.

The dependent variables were: ZONE the percentage of time the gaze point computed by the gaze tracking system is inside the Tobii zone, and OBJECT the percentage of time this gaze point is on the same virtual object as the Tobii system. A non-parametric ANOVA (kruskal-wallis) was conducted in order to measure the effect of S_t for several values of w_s.

The ANOVA found a significant main effect for S_t on the dependent variable ZONE when $w_s = 2.0$ (see Table 3). As shown on Figure 6 When our method is used, the percentage of time spent inside the Tobii zone is increased from 6.45% (standard deviation (SD)=4.11), in the case when the ANN-based gaze tracker is used alone, to 13% (SD=1.18) for $S_t = 0.0$, 12.87% (SD=0.96) for $S_t = 0.25$ and 11.18% (SD=1.41) for $S_t = 0.5$.

The ANOVA found a significant main effect for S_t on the dependent variable OBJECT when $w_s = 1.5$, $w_s = 2.0$ and $w_s = 2.5$ (see Table 3). For the single ANN-based gaze tracker, the percentage of time spent the percentage of time spent on the same object as the Tobii gaze point is 49.3% (SD=10.6%) as shown on Figure 7. In the most significant case, when $w_s = 2.0$, the time spent on the same object as the Tobii gaze point is increased to 70.52% (SD=5.85) for $S_t = 0.0$, 70.04% (SD=6.44) for $S_t = 0.25$ and 66.69% (SD=7.52) for $S_t = 0.5$.

5.4 Discussion

In this study, we defined two measures to assess the validity of using a saliency map computed from a bottom-up visual attention model to improve gaze tracking. OBJECT

Firstly, we compared the time spent by the gaze points computed with several methods inside the Tobii zone. We found that the time was significantly increased for the gaze point computed by the ANN-based gaze tracker improved by our method as compared to the ANN-based gaze tracker alone. The fact that the improvement is only significant for $w_s = 2.0$ suggests that smaller uncertainty windows were not large enough to overlap the Tobii zone. Also, the fact that scale factors higher than 2.0 did not improve results indicates that large uncertainty windows contain too many salient areas that are competing for the final gaze point position. In this case, the area the user is gazing at may be a high salient area but not the higher. We can illustrate this by computing the result of a gaze tracker that always takes the higher salient area on the screen as the final gaze position. In this case, the gaze points were inside the Tobii zone only 3.17% of the global time (Table 4). Moreover, as visible on Figure 6, the use of a saliency threshold does not seem to have a significant effect on the time spent in the Tobii zone. It seems that using a high saliency threshold degrade performance. This correlates Itti [11] results which show that users do not constantly look at the highest salient area.

	ANN alone	ANN with saliency map ($S_t = 0.0$, $W_s = 2$)	Saliency map alone
% of time in Tobii zone	6.45	12.98	3.17
% of time on same object	49.29	70.52	36.91

Table 4: Summary of the results. Best performance of our approach as compared to: the use of (1) ANN-based gaze tracker alone and (2) saliency map alone (i.e., using the whole image on screen).

Secondly, we found significantly better results concerning the time spent on the same object as the gaze point estimated by the Tobii eye tracker by the ANN-based eye tracker when it is combined with our method. In this case, three window scale factors present a significant improvement of our method over the single ANN-based gaze tracker. The fact that three window factors increase performance, compared to one for the percentage of time spent in the Tobii zone, can be explained by the fact that objects users are looking at are large on the screen, i.e. cover a wide area, since users tend to be close to objects they are looking at. Moreover, as visible on Figure 7, the use of a saliency threshold does not seem to have an significant effect on the time spent in the tobii zone. Even high saliency thresholds seem to degrade performance. This can be explained by the fact the object the user is looking at may not always contain the highest saliency value of the saliency map. This suggests that to improve the performance of our method, a top-down visual attention model could be used to modulate image space saliency by object-based interest values as proposed by Lee *et al.* [15].

Our results suggest that the straightforward method we propose could significantly increase gaze trackers performance, especially in the case of object-based interaction as shown in Table 4.

6 EXAMPLES OF APPLICATION

Our novel gaze-tracking approach has been implemented in several applications and for various purposes. We present two examples of interaction in 3D VE with gaze. These examples can be used during a navigation in the VE.

Firstly, user can light his environment by orienting a torchlight in the direction he is gazing at. The torch light can be simulated using a spot light. Another way to light the environment would be to position a point light near the surface the user is gazing at, as shown in Figure 8.

Secondly, user can move the dynamic objects of the VE by simply looking at them. As an example, while gazing at an object, the

Figure 8: Use of gaze point to light the virtual environment.

Figure 9: Use of gaze point to move dynamic objects closer to the user's avatar.

user can bring it closer to him/her by just pressing a key as shown on Figure 9. Users can also push the object away.

7 Conclusion

We have introduced the use of visual attention models and saliency maps to improve the accuracy of gaze tracking systems in interactive 3D applications.

We have first proposed a low-cost gaze tracking system based on artificial neural networks and using a single web cam. This system was found experimentally to achieve real-time gaze tracking with an average vertical accuracy of 1.14° and an average horizontal accuracy of 1.48°.

Then we have proposed an algorithm which is meant to improve the accuracy of any gaze tracking system such as the ANN-based system described previously. It uses an uncertainty window, defined by the gaze tracker accuracy, and located around the gaze point given by the tracker. Then, the visual attention model searches for the most salient points, or objects, located inside this uncertainty window, and determines a novel and, hopefully, better gaze point. Finally, we have presented the results of an experiment conducted to compare the performance of our approach during a first person navigation in a 3D virtual environment. Taken together, our results show a positive influence of our algorithm, i.e. of using visual attention models, on gaze-tracking performance. We have briefly given some examples of 3D interactive applications of our final algorithm within a first-person navigation in a virtual environment.

Our approach could be used as a real-time low-cost gaze tracking system in many applications such as for video games or virtual reality. Furthermore, the algorithm can be adapted to any gaze tracking system and the visual attention model can also be extended and adapted to the application in which it is used.

Future work could first concern the improvement of our algorithm by adding a top-down visual attention model. Second, we would also like to conduct more evaluations with higher-level tasks and in other contexts. Last, we would like to investigate novel interaction techniques based on our gaze-tracking system.

References

[1] C. Arzu. *Foveation for 3d vizualisation and stereo imaging*. PhD thesis, Helsinki University Of Technology, 2006.

[2] S. Baluja and D. Pomerleau. Non-intrusive gaze tracking using artificial neural networks. Technical report, Carnegie Mellon University, 1994.

[3] B. Beymer and M. Flickner. Eye gaze tracking using an active stereo head. *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, 2003.

[4] M. Bohme, A. Meyer, T. Martinetz, and E. Barth. Remote eye tracking: State of the art and directions for future development. *Proc. of COGAIN*, 2006.

[5] K. Cater, A. Chalmers, and G. Ward. Detail to attention: exploiting visual tasks for selective rendering. *Proc. of the 14th Eurographics workshop on Rendering*, pages 270–280, 2003.

[6] N. Courty, E. Marchand, and B. Arnaldi. A new application for saliency maps: Synthetic vision of autonomous actors. *Proc. of IEEE International Conference on Image Processing*, 2003.

[7] A. Duchowski, E. Medlin, N. Cournia, A. Gramopadhye, B. Melloy, and S. Nair. Binocular eye tracking in vr for visual inspection training. *Proc. of ACM Eye Tracking Research and Application*, 2002.

[8] A. Glenstrup and T. Engell-Nielsen. Eye controlled media : Present and future state. Master thesis, University of Copenhaguen, 1995.

[9] E. Guestrin and M. Eizenman. General theory of remote gaze estimation using the pupil center and corneal reflections. *In IEEE Transactions on Biomedical Engineering*, 53(6):1124–1133, 2006.

[10] C. Hennessey, B. Noureddin, and P. Lawrence. A single camera eye-gaze tracking system with free head motion. *Proc. of ACM symposium on Eye tracking research & applications*, pages 87–94, 2006.

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

[12] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *in IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1254–1259, 1998.

[13] A. Kaufman, A. Bandopadhay, and B. Shaviv. An eye tracking computer user interface. *Proc. of IEEE Research Frontier in Virtual Reality Workshop*, 1993.

[14] C. H. Lee, A. Varshney, and D. Jacobs. Mesh saliency. In *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2005)*, volume 24, pages 659–666, 2005.

[15] S. Lee, G. J. Kim, and S. Choi. Real-time tracking of visually attended objects in interactive virtual environments. *Proc. of ACM symposium on Virtual reality software and technology*, pages 29–38, 2007.

[16] P. Longhurst, K. Debattista, and A. Chalmers. A gpu based saliency map for high-fidelity selective rendering. *Proc. of the 4th international conference on Computer graphics, virtual reality, visualisation and interaction in Africa*, pages 21–29, 2006.

[17] V. Navalpakkam and L. Itti. Modeling the influence of task on attention. *In Vision Research*, 45(2):205–231, 2005.

[18] N. M. Piratla and A. P. Jayasumana. A neural network based real-time gaze tracker. *In J. Netw. Comput. Appl.*, 25(3):179–196, 2002.

[19] A. R. Robertson. Historical development of cie recommended color difference equations. *In Color Research and Application*, 15(3):167–170, 1990.

[20] Tobii. http://www.tobii.com.

[21] A. M. Treisman and G. Gelade. A feature-integration theory of attention. *In Cognitive Psychology*, 12(1):97–136, 1980.

[22] H. Yamazoe, A. Utsumi, T. Yonezawa, and S. Abe. Remote gaze estimation with a single camera based on facial-feature tracking without special calibration actions. *Proc. of the ACM symposium on Eye tracking research & applications*, pages 245–250, 2008.

[23] D. Yarbus. *Eye motion and vision*. Plenum Press, 1967.

[24] D. Yoo and M. Chung. Non-intrusive eye gaze estimation using a projective invariant under head movement. *Proc of. IEEE International Conference on Robotics and Automation*, 2006.