Thèse

THESE INSA Rennes
sous le sceau de l'Université européenne de Bretagne
pour obtenir le titre de
DOCTEUR DE L'INSA DE RENNES
Spécialité : Informatique

présentée par
Sébastien Hillaire
ECOLE DOCTORALE : *MATISSE*
LABORATOIRE : *IRISA – Orange Labs*

Contribution à l'étude des modèles d'attention visuelle et du suivi de regard pour améliorer le retour visuel dans les applications 3D interactives

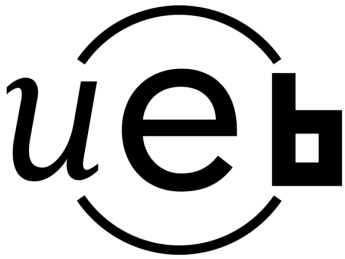
Contribution to the Study of Visual Attention Models and Gaze Tracking to Improve Visual Feedback in 3D Interactive Applications

Thèse soutenue le 21.01.2010
devant le jury composé de :

Bruno Arnaldi
Professeur, INSA de Rennes / *Président*
Alan Chalmers
Professeur, Warwick University / *Rapporteur*
Sabine Coquillart
Directrice de recherche, INRIA Grenoble / *Rapporteur*
Martin Hachet
Chargé de recherche, INRIA Bordeaux / *Examinateur*
Marie-Paule Cani
Professeur, Institut Polytechnique de Grenoble / *Examinateur*
Rémi Cozot
Maitre de conférences, Université de Rennes 1 / *Examinateur*
Gaspard Breton
Ingénieur d'études, Orange Labs Rennes / *Encadrant de thèse*
Anatole Lécuyer
Chargé de recherche, INRIA Rennes / *Directeur de thèse*

Contribution à l'étude des modèles d'attention visuelle et du suivi de regard pour améliorer le retour visuel dans les applications 3D interactives

Contribution to the Study of Visual Attention Models and Gaze Tracking to Improve Visual Feedback in 3D Interactive Applications



En partenariat avec

Centre INRIA Rennes
Bretagne Atlantique

France Telecom
Orange Labs Rennes

Equipe Projet BUNRAKU
IRISA-INRIA Rennes

Whoever fights monsters should see to it that in the process he does not become a monster.
And if you gaze long enough into an abyss, the abyss will gaze back into you.
Beyond Good and Evil
Friedrich Nietzsche

Remerciements

Je tiens à remercier très chaleureusement Dr. Anatole Lécuyer et Dr. Gaspard Breton qui m'ont encadré tout au long de cette thèse. Je vous remercie beaucoup pour vos innombrables conseils, votre soutien moral et votre sympathie. Un grand merci aussi a Rémi Cozot, Jérôme Royant et Daniel Pelé qui m'ont énormément suivi tout au long de ces trois années.

Je tiens à remercier les personnes ayant acceptées de faire partie de mon jury de thèse. Je remercie donc les rapporteurs Pr. Alan Chalmers et Dr. Sabine Coquillart. Je remercie aussi Pr. Marie-Paule Cani et Dr. Martin Hachet pour avoir accepté d'examiner mon travail. Finalement, je remercie Pr. Bruno Arnaldi qui a accepté de présider mon jury.

Je tiens également à remercier toutes les personnes avec qui j'ai pu collaborer : Dr. Géry Casiez, Léo Terziman, Dr. Kevin Boulanger, Dr. Guillaume François, Dr. Nizar Ouarti, Dr. Tony Regia-Corte, Gabriel Cirio et Dr. Maud Marchal.

Je remercie beaucoup les équipes entières de TECH/ASAP/IACA d'Orange Labs et BUNRAKU de l'INRIA avec lesquelles j'ai passé ces trois fabuleuses années. Je tiens à remercier plus particulièrement Olivier Aubault pour sa *gné* attitude, Nicolas "Basket" Stoiber pour *les jeudis du sport*, Philippe "Papi mignon" Vonwyl pour son inépuisable bonne humeur, Noémie Esnault et Rozen Bouville pour la *girl touch* et, finalement, Kadi Bouatouch et Fabrice Lamarche pour leur immense connaissance et sympathie. Je remercie également toutes les autres personnes de ces équipes.

Et bien sûr, un immense merci à ma famille et à ma *Crevette* qui m'ont encouragé tout au long de mon aventure de doctorant.

Contents

Introduction

Our research activities belong to the field of Virtual Reality (VR). A definition of Virtual Reality is given by Burdea and Coiffet [18] as "*a high-end user-computer interface that involves real-time simulation and interactions through multiple sensory channels. These sensory modalities are visual, tactile, auditory, smell, and taste*". VR research is seeking for novel communication channels that could improve human-computer interaction with 3D Virtual Environments (VE) [15]. Together with novel input peripherals [25] and methods [41], several interaction channels have already been identified such as gesture, voice and more recently brain-computer interfaces [35]. Multimodal virtual reality systems can use several of these communication channels in parallel and can integrate various sensory feedback channels for an improved VR experience such as visual, haptic or audio feedback [62]. As for today, VR is used in many applications such as sport training, architectural visits, industrial virtual prototyping, etc. VR is a wonderful tool to help people train and prototype before effective realization. Furthermore, after a few research years, human-computer interfaces and interaction methods developed for VR tend to be provided on the mass market for daily use by consumers, e.g. gesture interaction in video games like Nintendo *Wiimote* or Microsoft *Kinect*. VR can be viewed as an *avant-garde* experimental domain where novel interfaces and interaction paradigms are designed, tested and refined by researchers and industrials before everyone can use them for daily use.

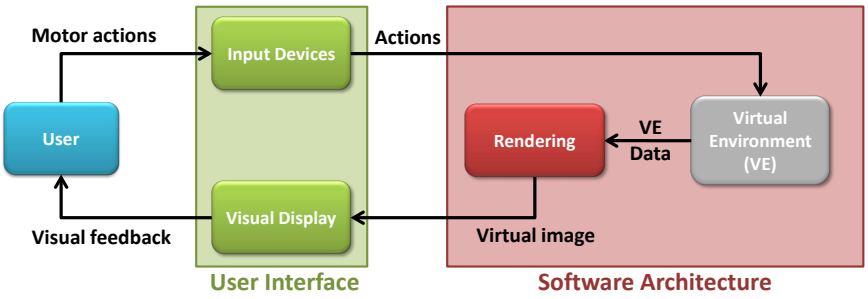


Figure 1: A typical real-time graphical application.

As depicted in Figure 1, in a typical VR setup, a user is interacting with real-time graphic applications using motor actions on input peripherals. Then, these actions are interpreted by the application to modify the 3D VE. Finally, a view of the VE is rendered on the display peripheral as a visual feedback to the user, thus closing the interaction/visualization loop.

Our work focuses on the visual sensory channel: the link between the user's eyes and the display peripheral. Nowadays, this channel is generally used as a one-way link, from the display peripheral to the user, in order to provide a visual feedback of the VE, i.e. visualization. Moreover, the visual feedback in current VR systems does not take into account human perception. As a result, virtual scenes often look unnatural. The downside of this unnatural look is that it can result in low implication and

immersion feelings of users in VE.

In this PhD manuscript, we propose to consider the visual sensory channel as a two-way link. In this case, VR applications are aware of the user's attention or gaze point, i.e. the point he is looking at. Thus, during the rendering process, it would become possible to apply perception based visual effects taking into account user's gaze position in order to improve the visual feedback.

Our research work is organized in two major research axes corresponding to the input/output decomposition:

1. The first axis focuses on the inputs of the application and is dedicated to the improvement of gaze-tracking in 3D virtual environments using human visual attention and gaze behavior. This axis represents the link *from* the user's eyes *to* the display peripheral, i.e. the user's attention.

2. The second axis focuses on the outputs of the application and is dedicated to the improvement of the visual feedback of 3D virtual environments based on visual perception and visual attention. This axis represents the link *from* the display peripheral *to* the user's eyes, i.e. the visual feedback.

The first research axis is dedicated to the development of novel methods to improve the tracking of user's visual attention or gaze point in interactive 3D applications. Nowadays, hardware and software solutions exist for gaze/attention tracking. On the one hand, gaze tracking systems can be used to compute user's gaze point. However, these solutions are often expensive or intrusive, thus dedicated to research or industrial uses. Furthermore, each of these systems have a limited accuracy resulting from the hardware pieces and vision algorithms used. On the other hand, real-time computational human attention simulation have recently emerged, namely visual attention models. These models are designed to compute users' attention over a picture or a virtual scene by simulating several attentional and cognitive components of the human visual system. A final gaze position can be computed based on the attention distribution. However, these models can only simulate human attention as a general case and are often limited to the simulation of few human attentional components. Moreover, few of these models [73] have been designed to be used in our context, i.e. first-person navigation in VE. As a consequence, the first axis will focus on the research of novel way to compute the user's attention using visual attention models and/or gaze tracking systems in VE.

The second research axis is dedicated to the proposition of a novel use of the user's attention/gaze: using perception-based visual effects adapted to the user's gaze. Human attention is used for various purposes but, as for today, it has only been used for two main goals in real-time VR systems: interaction and perception-based rendering. Concerning interaction, the use of the user's gaze point has been reported as a convenient and natural input [60]. However, this feature must be used carefully because it can result in unintentional actions from users not familiar with explicitly control of their actions using eyes movements. Concerning perception-based rendering, the gaze point has been used to manage Level-of-Detail (LoD) geometric [78] or lighting [87] of a virtual scene. These methods decrease the rendering quality in areas the user is not gazing at. This is possible because the accuracy of the human visual system decreases according to the visual angle as compared to the viewing direction. As a result, these simplifications accelerate the rendering process. As compared to previous perception-based approaches dedicated to the simplification of the rendering process, our work is the first attempt to propose perception–based visual effects adapted in real-time to the user's gaze point in order to improve the quality of the rendering process. To sum up, most previous approaches propose to **reduce image quality** in areas located far from the user's focus point. Instead, we propose to study how to **improve image quality** using perception-based rendering algorithms based on focus point information.

Research framework

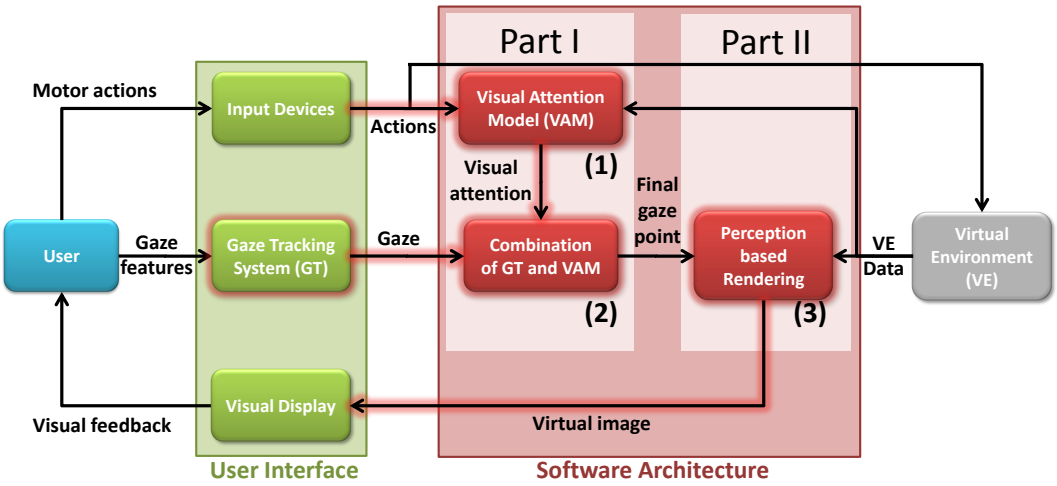


Figure 2: Research framework. Main contributions are emphasized in red color.

We present our research framework in Figure 2, where our main contributions are emphasized in red. As compared with Figure 1, the application now has two inputs from the user interface: the user's motor actions and gaze position (if a gaze tracker is available). The input layer of the application, corresponding to our first research axis, is composed of a visual attention model simulating human visual attention (Figure 2(1)). This component can be used either by its own to evaluate the user's attention in VE, or in combination with the gaze tracking system (Figure 2(2)). This first layer then sends the computed user's gaze position to the second and last layer. This last layer, corresponding to our second research axis, now have access to information about the user's attention. Thus, it is able to apply perception-based visual effects adapted in real-time to user's gaze position when rendering the VE (Figure 2(3)). The final computer generated view of the VE, the output of our framework, is finally displayed on the visual display as a feedback to the user.

We now details the scientific questions related to the two aforementioned research axes described in two main parts.

Part 1: Improving Gaze-Tracking in 3D Virtual Environments using Human Visual Attention and Gaze Behavior

- Novel visual attention model more adapted to the exploration of 3D virtual environments

Visual attention models have been shown as good predictors of human gaze distribution [94] over pictures and videos. However, surprisingly, few visual attention models have been proposed to evaluate human attention during exploration of virtual environments. Indeed, visual attention models could be a nice software alternative to hardware gaze tracking systems. To our best knowledge, only one model adapted to this context has been proposed by Lee et al. [73]. This model includes state-of-the-art attentional components as well as new components adapted to active navigation in virtual environments. Moreover, it has been designed to only compute a set of possibly gazed at objects by the user. First, we would like to improve this model by proposing new components adapted to the fact that the user is navigating in virtual environments with a first person view, and using its motor actions on the computer interface

(Figure 2 (1)). Second, we would like our model to compute a gaze position on the screen instead of a list of possibly gazed at objects to facilitate the use of existing and future gaze based methods.

- Novel approach to improve gaze tracking systems using visual attention models

Currently, the user's gaze point can be computed using two distinct methods: either a gaze tracking system *or* a visual attention model. Indeed, several gaze tracking systems have been proposed: from accurate high-end [118] to webcam based gaze tracker [7]. To evaluate a gaze point position, these systems only rely on a video stream of the user. Thus, improving the accuracy of gaze tracking systems is typically done by using more advanced hardware pieces or researching novel vision/tracking algorithms. However, as for today, these systems always have a limited accuracy. In our case, we would like to improve the accuracy of any gaze tracking system by taking into account what the user is looking at and the properties of the human visual system. To this aim, we would like to investigate the coupling of gaze tracking systems with visual attention models (Figure 2 (2)). We would like our methods to take advantage of the strength of both methods to increase the accuracy and reliability of any existing gaze tracking system.

Part 2: Improving Visual Feedback in 3D Virtual Environments Based on User's Visual Attention

In real-time applications, the user's gaze point position on screen has been mainly used to increase the user's interaction capacity [111] or to manage unperceptible LoD when rendering a VE for the sake of performance [78] [87]. The gaze point proved its utility in each of these cases. Instead of simplifying the rendered scene, we would like to propose a novel use of the user's gaze point: adding gaze-based visual effects simulating phenomenons occurring in human vision (Figure 2 (3)). We would also like to study users preferences concerning the proposed visual effects as well as their effects on perception of the VE.

Approach and contributions

The following manuscript describes the work we conducted in order to address the two research axes aforementioned in the context of 3D VR. Our researches are particularly focused on the specific context of *real-time first-person navigation in 3D VE*, i.e. the virtual camera is positioned at the level of the eyes of the user's avatar in the virtual world. Such situations are very immersive, and are used in many interactive 3D applications such as video games, virtual visits of urban or architectural projects, training, etc. These researches have been conducted specifically for the most commonly encountered situation where a user is sitting in front of a desktop computer and a regular display screen.

The first Chapter of this thesis exposes the background. It details the human visual system from light entering the eye to information processing in the brain. Several important features of the human visual system are covered, such as perception accuracy, eye movements, etc. Second, human visual attention is described with details about how perceived light and cognitive processes are integrated to efficiently analyze a scene. Then, we present a survey of existing computational visual attention models simulating the human visual system. Finally, we report about previous work focusing on the use of human visual attention and gaze tracking in VR systems.

The following Chapters are dedicated to our scientific contributions emphasized in red in Figure 2. Our contributions are separated in two parts corresponding to the two major research axes: Part 1) is

dedicated to our work focused on improving the user's attention and gaze tracking in VE, whereas Part 2) is dedicated to our novel approach using the user's gaze point to apply perception-based visual effect when rendering a VE.

Part 1: Improving Gaze-Tracking in 3D Virtual Environments using Human Visual Attention and Gaze Behavior

When using a visual attention model to compute human attention, it has been suggested that using attentional components adapted to the context in which the model is used could lead to better results [73]. However, none of the existing visual attention models take into account the specificity of human gaze behavior observed during first-person navigation in VR. During real pedestrian walks, specific behaviors can be observed such as the anticipatory gaze behavior before a turn [39] and the fact that we are "going where we are looking" and not "we are looking where we are going" [10]. In order to propose novel visual attention components, we studied human gaze when walking in virtual environments using a first-person view in order to search for gaze behavior similar to those observed in real life. Consequently, in **Chapter 2**, we conduct an experiment to **study user behavior when walking** forward and taking turns in virtual environments during a first person navigation. Then, we propose **gaze behavior models that can predict the user's gaze** direction in VE when walking and turning in first-person navigation.

Nowadays, gaze tracking systems are not available on the mass market: only researchers and industrial companies have access to such systems for VR setups. We would like to propose a method to have access to the user's visual attention in VE without any expensive gaze tracking hardware and for applications ranging from VR setups to video games. To this aim, a real-time visual attention model simulating human visual attention could be used. Surprisingly, to the best of our knowledge, only one visual attention model adapted to our context has already been proposed [73]. Furthermore, this model also has the drawback of only computing a set of possibly gazed at objects instead of a single gaze point. Consequently, in **Chapter 3**, we propose a novel visual attention model computing human visual attention during real-time exploration of virtual environments using a first-person view. In our context, this model is the first to **compute a single continuous gaze point** position on the screen. We also propose a **new representation of visual objects based on surface elements** instead of meshes. This representation takes advantage of the parallel computational power of GPU hardware and performs efficient computation of attentional components for real-time performance.

Many gaze tracking systems have already been proposed. These systems are often accurate but, because of their intrusiveness, or cost, cannot be sold on the mass market for daily use. Today, it would be valuable for interactive 3D applications, such as video games, to have a low-cost gaze-tracking system relying, for instance, on a basic web cam [126]. However, contemporary web cam based gaze trackers are not accurate enough [126]. Moreover, gaze tracking systems suffer from many problems like drift or high frequency noise. Particularly, given the fact that each gaze tracking system has its own accuracy, an uncertainty window should be considered instead of a single gaze position. Contrary to gaze trackers, visual attention models can only evaluate the user's attention distribution as a general case and have the disadvantage of being deterministic. Thus, our first question was: is it possible to improve existing gaze tracking systems using a visual attention model? Consequently, in **Chapter 4** we investigate the **combination of a gaze tracking system together with a visual attention model** to potentially **improve the stability and accuracy of the computed gaze position**. We report on the results of two experiments which suggest that this combination can lead to an increase of the overall tracking accuracy.

Part 2: Improving Visual Feedback in 3D Virtual Environments Based on the User's Visual Attention

Thanks to the recent rise of fast programmable graphic hardware, virtual scenes can now be rendered in real-time with visual quality close to real pictures. But, as for today, human perception and attention is not often taken into account. That is to say, rendered pictures can sometimes look too perfect due to the lack of natural effects occurring in human vision like, for example, Depth-of-Field (DoF) or motion blur. Therefore, there is a need to add perception based visual effect simulating human vision. Consequently, in **Chapter 5**, we propose a novel **real-time and automatic DoF blur** algorithm to improve the visual feedback of the VE based on human perception. Our method features an **auto-focus system** taking into account the user's tasks to automatically compute DoF input parameters. We also report on an **experiment conducted to study users subjective preferences** concerning the depth-of-field blur effect using a common gamer apparatus. Finally, we report on the **influence of these effects on gamers performance** during first-person shooting game sessions.

The use of the user's gaze point feature can give developers several advantages. For instance, it has often been used in perception based rendering algorithms in order to manage 3D objects [78] and lighting [87] LoD to accelerate the rendering process. These methods can be used to decimate not gazed at areas of a virtual scene without the user noticing the drop of quality. Instead of simplifying the rendered picture, we propose a novel use of the gaze point feature: adding gaze-based visual effects simulating natural phenomenon occurring in human vision. This enrichment of the rendered image aims at improving the visual feedback to the user. Using such perception-based visual effects could lead to an increase of the user's immersion and perception of the VE. Consequently, in **Chapter 6**, we study the **automatic adaptation of the visual feedback based on the user's gaze point** according to two perception-based visual effects: (1) a **compensated camera motion** which simulates eyes movement when walking and (2) a **Depth-of-Field blur effect** which simulates the fact that humans perceive sharp objects only within a range of distances around the focal distance. Finally, we report on an experiment to study users preferences toward these effects when they are computed with or without gaze tracking.

Chapter 1

Background: tracking and use of human visual attention

In this section, we survey related work on human perception and visual attention as well as how it has been exploited in VR systems. First, we describe how humans perceive their environment using vision, from light entering in the eye to high level interpretation in the brain with details about eye movements and visual strategy involved to analyze a scene. Then, we review previous work on human attention simulation and gaze tracking. Finally, we report on the existing applications based on human attention tracking and simulation.

1.1 Human visual perception and visual attention

1.1.1 Human visual system

The human visual system can be divided in three major parts: (1) the eye (the system entry point), the visual pathway (trasmitting encoded visual information to the brain) and the visual cortex (part of the brain processing visual information).

1.1.1.1 Human eye

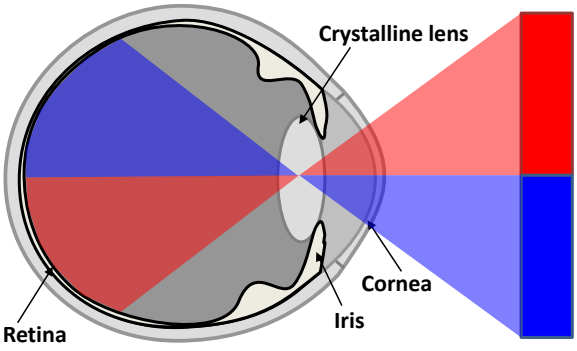


Figure 1.1: Sketch of a human eye.

The human eyes are the first organs of the complex human visual system process. These organs allow us to perceive our environment by processing the light reaching its photo-receptors. Indeed, the crystalline acts as a lens by projecting incoming lights on the eye background called the retina [16] (see Figure 1.1). The retina, positionned in the background of the eye, is covered with several photo-receptors. These photo-receptors are of two types:

- *Cones* allow Photopic (sensibility under high luminosity) and color vision. Three types of cones exist, each type being characterized by its sensibility to a certain range of wavelength: S-cones (short wavelength: blue part of the light spectrum), M-cone (middle wavelength: green-to-yellow part of the light specturm) and L-cones (large wavelength: green-to-yellow-to-red part of the light spectrum).
- *Rods* allows a Scotopic (sensibility under low luminosity) and low colored vision.

Humans only perceive a small part of the whole light spectrum [4]: perceivable wavelengths range only from 400 to 700 nanometer. Smaller (X ray, ultraviolet) and larger (radio wave and infra red) wavelengths cannot be perceived.

There are over 6 millions of cones mostly concentrated at the fovea part of the retina (Figure 1.2) [124]. They allow us to accurately perceive our environment only on 2 degree of angle of the visual field. Whereas, there are over 120 millions rods distributed at the periphery of the retina, thus surrounding the fovea. On Figure 1.2, we can observe the optical disc. This is through this *hole* in the retina where visual information from photoreceptors is forwarded to the visual pathway using the optic nerves. We do not perceive our environment at this place: this hole in visual information is filled with visual information coming from the other eye thanks to an integration process occuring at the level of the visual cortex.

The optic nerve is composed of 1.2 million fibers whereas the retina is composed of 130 millions of photo-receptors. Indeed, a part of the visual information is directly processed and organized at the level of the retina. Raw visual information coming from photo-receptors is not directly sent to the visual cortex. Indeed, the retina is also composed of ganglion cells which regroup and transform information coming from photo-receptors.

Ganglion cells organize photo-receptors in a concentric shape composed of a center and a surrounding. There exists two types of ganglion cells: *center-on* and *center-off* cells. By default, these cells

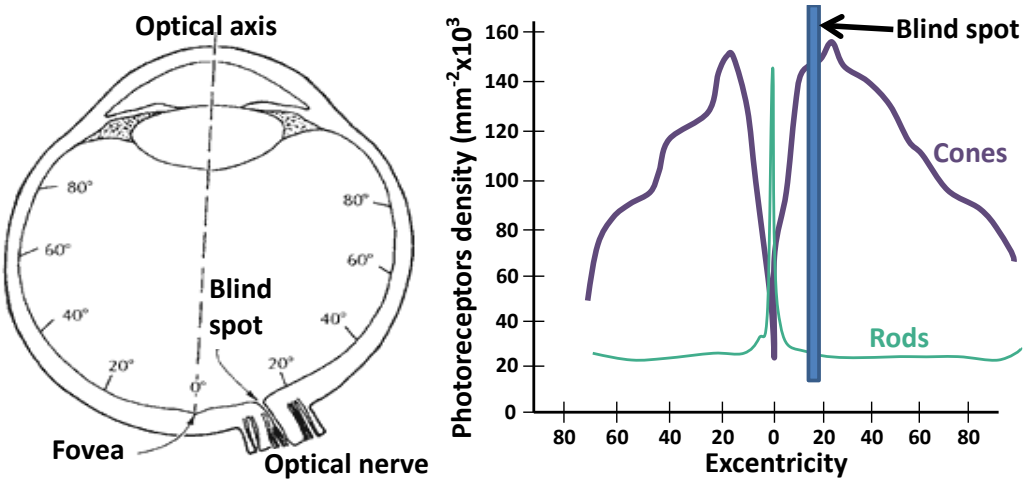


Figure 1.2: Rods (green) and cones (purple) density on the retina as a function of eccentricity [124].

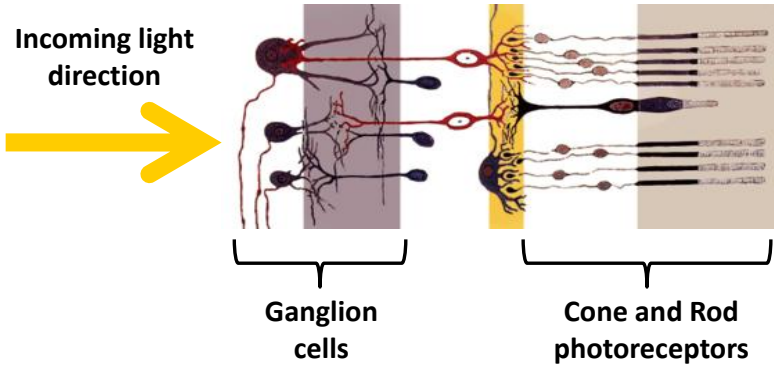


Figure 1.3: Organization of photo-receptors and ganglion cells on the retina. Light comes from the left and travel through ganglion cells to finally reach photo-receptors.

spontaneously send a continuous signal [16]. In the case of a center-on cell, if the center is stimulated by lights and not the surrounding region, a strong signal will be sent instead of the default one. Whereas, if the surrounding region is also stimulated by reaching its photo-receptors, a low signal, but still stronger than the default one, will be sent. In the case when only, the surrounding region is stimulated by light, a signal lower than the default one will be generated. An inverse behavior is observable in the case of the center-off ganglion cells.

To sum up with, ganglion cells are sensitive to luminosity gradients: as a result, they are sensitive to edges [16]. It is also important to note that the center and surrounding regions are composed of different types of photo-receptors, thus being more sensitive to certain colored contrasts. Indeed, humans are particularly sensitive to red-green and blue-yellow colored difference. Thanks to their center-surround shape, ganglion cells signal is not influenced by the orientation of a stimulus. Also, the more we move away from the fovea, the larger retina area a ganglion cell will cover. Thus, the size of a detectable stimulus will depends on their radius.

Once the quantity of visual information have been reduced by ganglion cells, the resulting visual information can be sent to the visual cortex through the visual pathway.

1.1.1.2 Visual pathway

Visual information coming from the eyes, through the optic nerve, are first passing through the optic chiasma [16]. Data from each eye will be reorganized by half visual field. Actually, visual information coming from both eyes contains some redundancy resulting from the intersection of the two visual fields. Indeed, visual information on the left of each visual fields will be moved to the right visual cortex and visual information on the right of each visual fields will be moved to the left visual cortex.

Before reaching the visual cortex, some parts of the brain along the visual pathway, have early access to visual information [16]: the superior colliculus to help initiating fast movements of the eyes toward a visual target, the suprachiasmatic nucleus controlling the endogeneous circadian cycle (day night biological cycle) and also the pretectum controlling pipular reflexes to light intensity (closing the iris when too much light reaches the retina). Then, visual information can finally move through the lateral (left and right) geniculate bodies composed of three types of cells (similar to ganglion cells on the retina) [16]:

- magnocellulare cells : process small and colored details of the visual fields.

- parvocellulare cells : less sensitive to details and can only process luminosity. This cells are sensitive to fast motions.
- koniocellulare cells : involved in the proprioception induced by visual information.

The high specialization of these cells is a proof that each different visual primitive (color, motion, etc) are sent to the visual cortex via several different channels [76].

1.1.1.3 Visual cortex

The visual cortex is the main visual information processing site. This part of the brain, positioned at the back of the skull, is responsible for the detection of complex visual primitive as well as their integration for the purpose of a high level interpretation [16]. The visual cortex is divided in several sub-parts: V1 (primary visual cortex), V2, etc, to V6.

In contrary to retinal ganglion cells, cortical cells are sensitives to orientation and motion. In the visual vortex, there exists two types of cells [99]:

- Simple cells: sensitive to outline (edges, silhouettes) and slow motion.
- Complex cells: sensitive to fast motion.

Each cells are sensitive to a pre-defined orientation with a tolerance interval of 15 degrees. Several cells will be required to detect stimuli having different orientations.

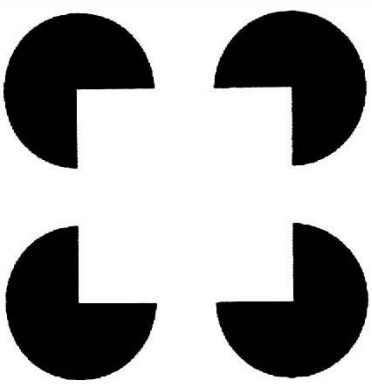


Figure 1.4: The suggested square illusion. The square does not exist as it is was not explicitly drawn, but our brain suggests its existence.

These computed data are then sent to the dorsal and ventral pathes [16]. The dorsal path is in charge of the spatial aspect of perception such as motion analysis or detection of relative positions of objects. The ventral path is in charge of accurate perception (data coming from the fovea) as well as object recognition from a semantic point of view. Thus, object can often detect objects which are partially visible or occluded. As a result, as shown on Figure 1.4, the brain infers the presence of a square even if it has not been explicitly drawn.

To sum up, the visual cortex processes incoming data from the perceived image using higher level perception than previous stages focusing on low level visual feature. The whole processing chains from the eyes to the visual cortex will finally define the final visual accuracy.

1.1.1.4 Visual Acuity

Several visual acuity measure have been proposed by researchers [4]. They are expressed in minutes of angle, 60 minutes of angle corresponding to 1 degree. For each measure, the visual angle is defined by the angle between two vectors having their origin on the retina and pointing to two different visual features. Here are some example of acuity measures:

- point-acuity : ability to distinguish two points based on the angle between them. (1 minute of angle)
- grid-acuity : ability to distinguish several white and black line on a grey background. (1-2 minutes of angle)
- letter-acuity : ability to read a letter based on its size. A mark of 20/20 indicates that 90% of a set of letters having a size of 5 minutes of angle were succesfully read.
- stereo-acuity : ability to distinguish a depth difference between two objects based on the angle between them. (10 minutes of angle)

Retina acuity is maximum at its center, at the level of the fovea. It corresponds to the center of the perceived image. The more we move away from the fovea, the less accurate the perception is. Indeed, at the periphery, cones become progressively replaced by rods (Figure 1.2) which does not allow an accurate perception. At the level of the fovea, humans can perceive details having a size down to 0.5 minute of angle. This precision increases up to 35 times at the periphery of the perceived image.

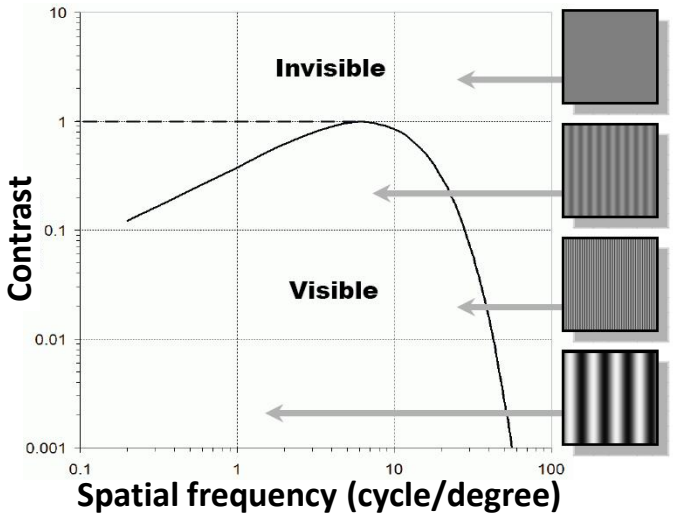


Figure 1.5: The contrast-sensitivity of humans is function of contrast and frequency [123].

The human visual system acuity can also be described in term of contrast sensitivity [78]. This measure represents the ability to distinguish a sinusoidal wave as a function of its amplitude and frequency (Figure 1.5). This clearly demonstrates that the human visual system is limited as it can perceive a sub-space of possible sin waves.

The level of detail perceivable on an object decreases as the angular speed of the object increases relatively the eye position and orientation. These moving objects become progressively blurred: this is

a direct consequence of the integration capacity of retina photo-receptors. Furthermore, despite the fact that peripheral vision has a low accuracy, it is extremely sensitive to motion [20][4]. This characteristic is strongly related to survival capacity of human as a motion is potentially a source of danger.

1.1.2 Ocular movements

Ocular movements are part of the human visual system. There are several types of ocular movements. Each of these movements, being intentional or not, are used in specific cases: scene analysis, gaze stability, motion compensation, etc. Physically, eyes movements are controlled by 3 types of muscles: lateral rectus muscles (left and right), rectus musles (inferior and superior) and oblique muscles (inferior and superior).

1.1.2.1 Classification of ocular movements

Fixation/saccade movement is the most used by our visual system. It is composed of two repeated steps [103]: a saccade (duration varies between 120ms to 300ms) followed by a fixation (duration varies between 200ms to 600ms).

This ocular movement is used when we observe a picture, scene or when we are reading a text. Our eyes jump (saccade step) from place to place to sequentially analyze in details all parts of the perceived image (fixation step). During fixation, the accurate visual field is enlarged by 0.2 degree of angle using micro-saccade. Then, when a new visual point of interest is defined (Section 1.1.3), eyes operate a fast rotation to orient themselves toward this new position. This rotation speed can reach a speed of 900˚/s. The saccade movement must be fast because, during this step, our perception mechanism is disabled. As an illustration, you can try to alternatively look at your right eye, then at left eye, in a mirror: you will not perceive that your eyes are moving.

The **smooth pursuit** movement is used to constantly maintain the projection of an object on the fovea despite the fact that this object is moving. This ocular movement can only be produced when looking at a continuously moving objects. Furthermore, it can only be maintained for angular speed up to 30˚/s [103]. Once this limits is reached, ocular movements become fixation/saccade.

When our body and head are translated or rotated, e.g. during pedestrian walk, our eyes have to operate a compensation in order to continuously fixate a point in a scene. These movements are automatically initiated and guided by several types of reflexes: **angular and linear vestibulo-ocular reflexes** (for head movements) and linear and angular vestibulocolic reflexes (for body movements). These reflexes play an important role on the eyes orientation and entirely rely on the human vestibular system. The vestibular system allows humans to perceive self motion, head position and orientation relative to gravity [98]. This system relies particularly on the six semi-circular channel positioned near the auditive system. These channels, 3 on each side of the head, detect angular acceleration of the head in space, in contrary to otolith channels which detect linear acceleration. The vestibulo-ocular reflexes are described in details in Section 1.1.2.2 in the case of pedestrian walk.

Ocular nystagmus is a movement occurring when the eyes are subject to a rapid visual motion [81] (opto-kinetic nystagmus). For example, this movement can be observed when a person is looking at a scrolling landscape from the window of a moving train. This movement consists of alternating continuous pursuit in the direction of the visual flow (slow phase) followed by rapid eye saccade (quick phase) in the opposite direction of the visual flow in order to re-orient eyes axes.

Our two eyes do not have the same parallel orientation axis. Instead, they are converging on a visual object: this phenomenon is called **ocular vergence** [4]. The eyes are oriented in the direction of the

observed object by taking into account its distance. When the object is positioned too close to the head of a human: he is starting to go cross-eyed. Ocular vergence is used to project a single point in space on the fovea of both eyes. This phenomenon also provides a better perception of an object volume or depth in a scene (stereo vision).

1.1.2.2 Ocular movements during pedestrian walk

When walking, human body is subject to a complex combined motion of all its moving parts. Indeed, the walking motion is periodic [84]: when walking at a moderate speed (100m/min), the head oscillates vertically at a frequency of 2Hz with accelerations up to $0.37g$. Vertical oscillations of the head are also generated with a frequency of 1.0Hz and accelerations of up to $0.1g$. Also, yaw and pitch rotations of the head can be as fast a 17°/s. In order to stabilize its focus and keep a stable and sharp image of the scene, the eyes have to operate several rotations to compensate for the motion resulting from the walk. In this case, vestibulo-ocular and vestivulocolic reflexes generate such compensation rotation (Section 1.1.2.1).

Also, it is important to dissociate horizontal and vertical vestibulo-ocular reflexes: their behavior is different depending on the distance a human is looking at. Horizontal vestibulo-ocular reflexes always compensate for the head motion [84] whereas vertical ones start to be coordinated with head motion when a human is looking at a distance less than 25cm [83]. Moreover, vestibulocolic reflexes are also used to stabilize the global human posture when walking [52].

Gaze behavior of Humans when walking and turning in real life has been longly studied by physiologists. Grasso *et al.* [39] studied gaze behavior of Humans taking a 90-degree curved path by forcing them to turn around obstacles. They showed that when taking curves, the eyes deviate toward the direction of the curved trajectory with an anticipation time of one second. This anticipatory nature of gaze reflects the fact that Humans need to prepare their intended action in order to compensate for the delays due to bio-mechanical inertia. When turning, the maximum gaze angle is reached at the corner. Surprisingly, the same behavior was observed when lights in the room were switched off [39]. This emphasizes the behavior reported by Berthoz [10] saying that humans *are going where they are looking instead of looking where they are going*.

When a human is turning, parts of his body does not turn continuously but periodically [38]. Indeed, the direction of the body and head are corrected at regular time steps. This perpetual correction could explain the required anticipatory nature of humans gaze in order to compensate for the delays due to bio-mechanical inertia, but also to anticipate their actions.

1.1.2.3 Summary

We have reported several human eye movements, each of these movement being adapted to a given situation. We have also reported that some of these movements can indeed be correction of body or head rotations in order to stabilize gaze direction by compensating for the head and body movements. Finally, we have exposed the anticipatory nature of human gaze which aims at compensating delays due to bio-mechanical inertia and to anticipate intended actions.

The most used gaze movement is the saccade/fixation one. However, this movement is not instantaneous and so it can take time to analyze a whole scene. In the next section, we present the mechanism used by the human visual system in order to cleverly choose the next gaze direction to quickly perceive the most important information about a scene.

1.1.3 Human visual attention

> "*Everyone knows what attention is. It is the taking possession by the mind, in clear and vivid form, of one out of what seems several simultaneously possible objects or trains of thought. Focalization, concentration, of consciousness are of its essence. It implies withdrawal from some things in order to deal effectively with others, and is a condition which has a real opposite in the confused, dazed, scatterbrained state which in French is called* distraction*, and* zerstreutheit *in German.*"
>
> William James [61]

Visual attention represents the capacity of a human to focus on a visual object. Due to the structure of our brain and the fact that we only accurately perceive the environment within 2 degrees of the visual field [22], the human visual system does not have the capabilities to analyze a whole scene in parallel. Actually, the human visual system spatio-temporally filter the perceived image in order to sequentially analyze small parts of this image [119]. What is this filter? Which mechanism is used to identify part of a perceived image that must be analyzed in priority?

Human visual attention is composed of two components [55]: the bottom-up component (also called *exogenous*), representing visual reflexes, and top-down component (*endogenous*), representing the cognitive process.

1.1.3.1 Bottom-up visual attention

The bottom-up, i.e. exogenous, component of human visual attention refers to visual reflexes: automatic attraction of attention based on perceived stimuli. Indeed, the human visual system cannot analyze a whole scene in parallel. Actually, it can only detect primitive features in parallel [119].

The feature integration theory of Treisman *et al.* [119] is largely accepted by the research community as a good modeling of the bottom-up process. In this theory, visual information of the whole perceived image are processed in a pre-attentive way and in parallel. As reported in Section 1.1.1, the human visual system is sensitive to several visual stimuli called salient *features*. Several visually salient features have been identified in previous researches [119][57]: red/green and blue/yellow antagonistic colors, intensities and orientations. According to the feature integration theory, each salient feature is integrated into a representation called saliency map [119][57]: A saliency value is defined for each area of the perceive image representing its attractiveness, i.e. the higher saliency of an area, the more a human is likely to look at this area. As a result, the highest salient area is likely to be processed, or gazed, in priority.

As an example, when someone first looks at a scene, his/her gaze is first unconsciously attracted by visually *salient* areas, e.g. a red ball on a green grass. The bottom-up process being unconscious, a single variation in a picture will automatically attract gaze in first. Figure 1.6 illustrates this process: because the only variation in the picture is a T being in red, your gaze should have been first unconsciously attracted by this letter which pops-out among other letter.

Among color, intensity and edge, several other features have been identified by researchers. Depth variations have been reported as a three dimensional feature that attracts gaze [34]. Dynamic features have been reported as having a high influence in defining saliency resulting in a higher attention priority over static features [20][4]. Such dynamic features are flickering [20] and motion [88].

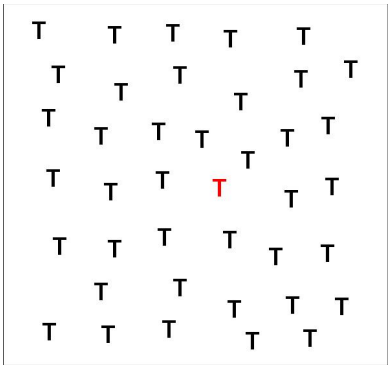


Figure 1.6: Test of the bottom-up component: Find the red letter.

Human visual attention is subject to *inhibition of return*. It means that once an area has been gazed at, there is an unconscious brain process which prevents human to gaze a second time at the same area or visual object within a short amount of time [68].

The resolution of the competition between salient areas is processed by several parts of the path taken by visual information [109] (retina, visual pathway, visual cortex). Moreover, visual attention is not only controlled by visual reflexes resulting from the bottom-up component of the human visual system. Indeed, the next gazed area also depends of the top-down component.

1.1.3.2 Top-down visual attention

Human visual attention is not only controlled by reflexes resulting from visual stimuli, but also by the cognitive process that takes place in the brain: top-down visual attention, i.e. endogenous. This component, of higher level than its bottom-up counterpart, is involved in the strategies humans use to analyze a scene.

Some researches [108] suggest that the top-down attention may be based on visual objects instead of areas in the perceived image. Following these theory, visual selection may be processed on discretized visual objects, i.e. the smallest unit that could receive attention, instead of the continuous visual field [114]. Furthermore, following Sears *et al.* [108] Multiple Object Tracking (MOT) theory, each object is assigned with an attention priority value that is computed in a stimulus-driven manner [119]. A set of 3 to 5 objects are indexed based on this value. This set of object can then be attended rapidly and before any other objects in the visual field.

Several viewing conditions seem to have an influence on the top-down component. For instance, Yarbus [127] has shown that the way people look at pictures strongly depends on the task they have to achieve. This research shows that the gaze pattern can be very different and depends of the question that was asked to participants just before they look at a picture (Figure 1.7). Thus, it was observed that for the question “How old are people in the scene?”, participants were mainly looking at the face of people. In contrary, for the question “Estimate the material circumstance of the family?”, participants were mainly looking at objects present in the house and rarely at people’s face.

The top-down component is also influenced by the knowledge of each individual based on short- and long-term memory, as well as personal experience and knowledge [86]. Hayhoe *et al.* [42] suggests that visual search might be planed beforehand and that regularities observed in the scene may help to acquire new information that might appear later (short-term memory). Also, Adams *et al.* [1] have shown that current task may influence some bottom-up features like colors and motion. Thus, a critical color for

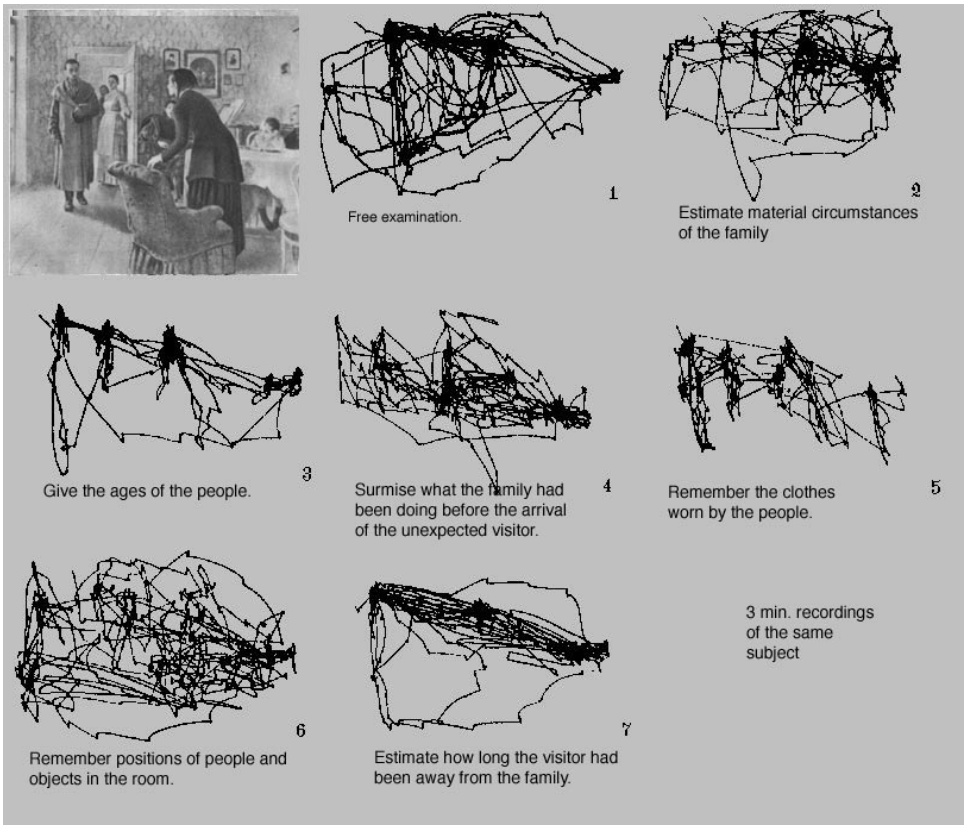


Figure 1.7: Yarbus' experiment [127] showing that human gaze behavior change based on the question they have to answer, i.e. the task to achieve.

current task, e.g. red if you are searching for a fire-extinguisher to put out a fire, might be biased at the level of the bottom-up component in order to accelerate its process. As a result, visual features being critical to achieve the current task will have a higher influence and are likely to be processed faster than others.

Another feature of the top-down component is called habituation. This phenomenon refers to the fact that objects become familiar over time [77], and we become oblivious to them [89]. Indeed, when exploring a scene, we may observe each object in the scene. Then, there is a high probability that known objects that are not important might not receive more attention later.

Figure 1.8 illustrates the top-down component of the human visual attention. In this figure, the task is to search for the red L hidden in the picture. Because the L letter does not pop out from other surrounding letters, since it does look like a T letter, the research strategy generally consists in a sequential parsing of every red letters. All black letters will be ignored. Then, the parsing might be different from an individual to another: for instance, people reading from left to right might parse the red letter from top to bottom and left to right.

1.1.4 Summary

Human visual attention is made of complex processes that take place from the retina in the eye to the visual cortex in the brain. These processes can be separated in two components: the bottom-up component

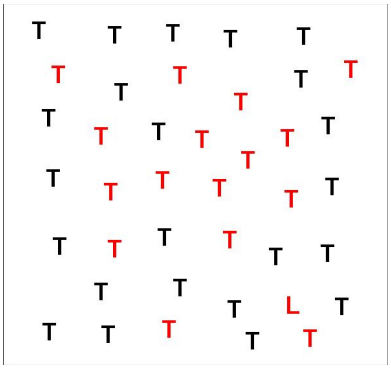


Figure 1.8: Test of the top-down component: find the red L letter.

representing visual reflexes and the top-down component representing the cognitive process that takes place in the human brain.

The bottom-up component is based on the perceived image of the visual field with attention being computed continuously on the image area. The top-down component is based on higher level data, i.e. task, knowledge, habituation with attention being computed per visual object. Furthermore, many more complex interactions between both components have also been reported such as color feature bias in the bottom-up by the top-down component.

To sum-up, human attention is a complex behavior processing raw visual information, acquired by the retina, at every steps through the visual pathway to the visual cortex. Because of this path, the resulting visual attention is a complex mix between visual reflexes and cognitive processes.

1.2 Hardware and software solution to human visual attention and gaze tracking

Computing human attention is not a trivial task. A possible way to evaluate human attention is to compute the *gaze* point position, the point someone is looking at. Only two alternatives can be used to compute someone's attention. The first solution is to use a hardware solution commonly called a *gaze tracking system*. Nowadays, this solution mostly rely on eye-tracking to determine the user's gaze direction. The second alternative is a software solution called *visual attention modeling* taking as input the visual feedback returned to the user and trying to determine the most probable gaze position using a visual attention model.

1.2.1 Gaze tracking systems

The majority of gaze tracking systems are designed to compute the gaze position onto a flat screen. Since their creation in the late 19th century, before the computer existed, they have advanced considerably [37]. Interest in these systems has grown thanks to their usefulness in several domains: from human studies in psychology to VR systems, as an aid for people with disabilities or to accelerate the rendering process. Table 1.1 summarizes the existing gaze tracking systems, considering the required hardware and their current accuracy.

Intrusive gaze tracking systems are generally restrictive as users have to wear heavy and uncomfortable equipment. As an example, Kaufman et al. [65] use electrooculography to measure the eyes'

Category	**Reference**	**Hardware**	**Intrusive**	**Horizontal accuracy** (degree)	**Vertical accuracy** (degree)	**Limitations**
Intrusive trackers	Kaufman *et al.* [65]	electrodes	yes	1.5 to 2	1.5 to 2	intrusive
	Duchowski *et al.* [33]	helmet with two screens	yes	0.3	0.3	intrusive and expensive
Remote gaze trackers	Beymer *et al.* [11]	use of two steerable cameras	no	0.6	0.6	expensive and cumbersome
	Tobii [118]	dedicated capture system	no	0.5	0.5	expensive
	Yoo *et al.* [129]	infra-red LED and CCD camera	no	1.0	0.8	user must stay between 30 to 40cm from the screen
	Hennessey *et al.* [43]	infra-red LED and CCD camera	no	1.0	1.0	infra-red light
	Guestrin *et al.* [40]	two lights and one CCD camera	no	0.9	0.9	needs the use of 2 specific light sources
	Yamazoe *et al.* [126]	CCD camera	no	5.0	7.0	low accuracy
ANN-based gaze trackers	Baluja *et al.* [7]	640 × 480 CCD camera	no	1.5	1.5	non robust calibration
	Piratla *et al.* [96]	640 × 480 webcam	yes	not available	not available	non robust calibration
Visual objects trackers	Lee *et al.* [73]	no hardware	no	object based	object based	depends on the VE and user's task

Table 1.1: Summary of existing gaze tracking systems.

muscular activity. This method requires the user to wear electrodes. Another technique requires the user to wear induction coil contact lenses [37]. The gaze direction can be computed by measuring the high-frequency electro-magnetic fields produced by these lenses. Both these techniques require the user's head to stay still. To overcome this problem, Duchowski et al. [33] propose a helmet with an embedded screen for each eye. Two gaze trackers based on the pupil-cornal reflection (P-CR) method are used (one for each eye). Intrusive systems are precise enough to be interesting for a research purpose, however, as shown in Table 1.1, few gaze tracking systems are intrusive and the current trend is towards the development of non-intrusive systems.

Few gaze tracking systems based on an Artificial-Neural-Network (ANN) have been proposed in the literature [7][96]. Baluja and Pomerleau [7] propose to send a low resolution image of a single eye directly to an ANN. Piralta and Jayasumana [96] compute features describing the current user's state , i.e. eyes' center, distance between upper and lower eyelid, etc. These features are then used as the input of an ANN. Such systems only need a 640×480 web cam and represent the screen as a discretized two

dimensional grid.

A new kind of gazetracker has recently emerged: remote gaze tracker. Remote gaze tracking systems are "systems that operate without contact with the user and permit free head movement within reasonable limits without losing tracking" [12]. These systems use either a high resolution camera or low resolution web cam and allow users to feel more free because they do not have to wear any devices. However, they are generally less accurate than other gaze trackers. Therefore, a lot of research is still going on to improve remote gaze tracking systems. Beymer and Flickner [11] propose a multi camera system tracking first the head of the user using a camera with a wide field of view, then, one of his eyes using a steerable high resolution narrow camera. Finally, a 3D representation of the eye is used jointly with the infra-red light glint to evaluate the user's gaze position. Tobii technology [118] proposes a non-intrusive gaze tracking system which enables some moderate movement of the user's head. It uses expensive dedicated tracking devices using infra-red lights, though further implementation details are not available [85]. Table 1.1 shows that these non-intrusive systems are very accurate but most of them require high expertise, are cumbersome [11] or very expensive [118].

Other remote gaze tracking systems have been designed to be used in everyday life by non-expert users with a simple and fast calibration process [40]. Some of the proposed systems [43] [129] still require infra-red LEDs but are able to achieve an accuracy of one degree under head movement. All the presented remote gaze trackers use either a 3D representation of the eye [11] or the P-CR method [33] to compute the gaze direction. The system developed by Yamazoe et al. [126] is able to compute the gaze position without infra-red light nor a calibration sequence. This system is dedicated to everyday use since it uses a single video camera. It has a low accuracy of 5 degrees horizontally and 7 degrees vertically, but the results are promising and yet could be improved.

1.2.2 Visual attention models

Visual attention modeling is a complex task. Actually, both components of human visual attention are not perfectly known today. However, despite this lack of knowledge, researchers have already proposed many visual attention models to simulate human attention.

1.2.2.1 Bottom-up visual attention modeling

The bottom-up component simulates the visual reflexes of the human visual system. Actually, the human visual system can detect primitive features in parallel, defining salient areas in the visual field. Then, it uses a sequential visual search to quickly analyze a scene [119].

Inspired by the feature integration theory [119], bottom-up visual attention models have been developed to computes a saliency map from an image [57]. 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 higher saliency of an area, the more a human is likely to look at this area.

The first bottom-up visual attention model widely accepted by the scientific community is the one proposed by Itti and Koch [57] [69]. The proposed method compute a saliency map corresponding to a perceived image, defining regions of interest as areas having a high saliency. In the case of a discretized picture, a saliency value can be defined for each pixel. First, in the original model [57] (Figure 1.9), raw data called *feature maps* containing color (red, green, blue and yellow simulating cones photo-receptors), intensity (simulating rod photo-receptor) and orientation features (simulating cells of the visual cortex) are computed for each pixel. For each of these primitives, a dyadic Gaussian pyramid is created. Then, a center-surround operator is applied on each of these pyramids to generate *conspicuity maps*. The

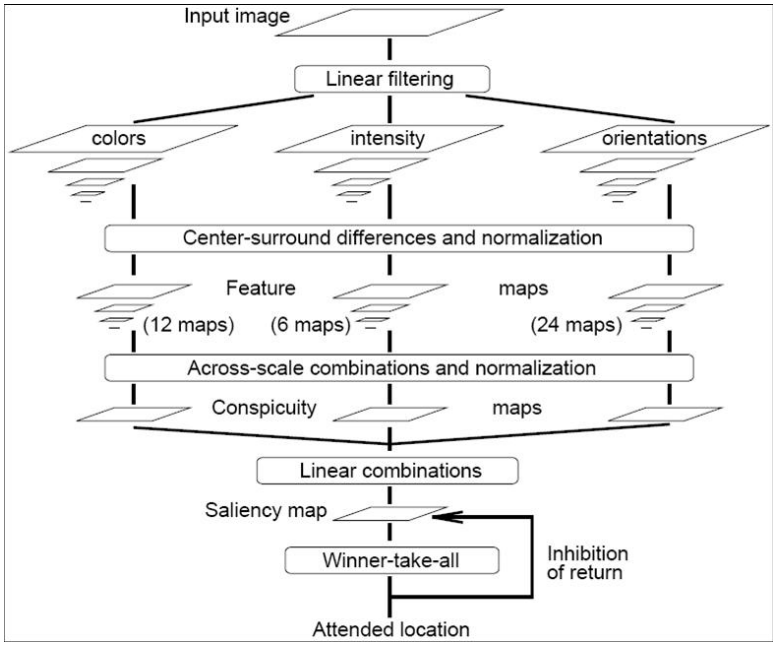


Figure 1.9: Architecture of the bottom-up model of Itti and Koch [57].

center-surround operator is indeed a multiple sum of differences between fine and coarse levels of the dyadic Gaussian pyramid. This process simulates concentric antagonistic behavior of ganglion cells in the retina and geniculate bodies in the brain. The resulting conspicuity maps, one for each feature map, are normalized using the $\mathcal{N}$ operator [57] inhibiting conspicuity maps not containing local strong peak of value, i.e. having a uniform distribution of attention. Finally, the *saliency map* is computed as a linear combination of all conspicuity maps (Figure 1.10). Having the saliency map of the perceived image, human attention can be evaluated using a Winner-Takes-All (WTA) neural network: the pixel having the highest saliency value is considered as the one receiving attention. Then, inhibition of return is simulated by inhibiting all neurons located around the one receiving attention and another WTA algorithm can be run to search for the next region of interest.

Figure 1.10: A picture and its associated saliency map computed [57].

More recently, Hu *et al.* [50] have proposed another method to compute salient features in a picture in a more robust way than [57]. To this aim, a polar transformation is applied on picture primitive (hue and intensity). Then, using a Principal Component Analysis, a binary picture representing region of interest

is generated. Despite the fact that the results are interesting, this paper is not based on physiological observation.

The model proposed by itti *et al.* [57], has often been used in its original form [89][114]. However, it has also been adapted to many situations. To compute a saliency map from a video, Cheng *et al.* [28] et Itti *et al.* [56] proposed to include a motion feature map using the Reichardt pyramidal method. The authors also proposed another feature map taking into account flickering in the video. Motion has also been considered in the case of 3D virtual environment (VE) by computing motion for each pixel [77] [128] in screen space or for each objects in world space [73]. Because depth have also been reported as a salient feature [34], it has also been included in some models as a new feature map [73] [77] [30]. Originally, Itti *et al.* [57] model uses red/green and blue/yellow antagonistic colors as well as intensities. In their model, antagonistic colors were computed using simple operation on RGB components. Lee *et al.* [73] improved this computation by using the Hue and Luminance value of the HLS color space. Furthermore, this original model has often been simplified in order to reach real-time performance. The first performance acceleration proposed by researcher was to port the saliency map computation to the Graphics Processing Unit (GPU). As an example, Longhurst *et al.* [77] have replaced the costly Gabord filter by a simpler but faster Canny edge filter. Computing a dyadic Gaussian pyramid for each feature map can be a costly operation on some hardware. As a replacement, Lee *et al.* [73] have proposed to use an interesting feature of GPU: automatic hardware mipmap pyramid generation. Thus, the dyadic Gaussian pyramid is approximated by a mipmap pyramid.

Concerning the implementation of the bottom-up component of a visual attention model, the modelitti *et al.* [57] has often been used as a reference. Then, It has been optimized and modified by adding new features which have been revealed as attracting human attention.

1.2.2.2 Top-down visual attention modeling

Few visual attention models purely based on a top-down component have been proposed. Indeed, these models are used in few specific cases, especially because human behavior is closely related to the task to achieve and memory.

Bordeux *et al.* [14] have proposed a complete pipeline to simulate the perception of autonomous virtual agent immersed in a VE. Every object in the scene were contained in a database which was interrogated in real-time using filters. The request filter vary based on the current task of each autonomous agents. For instance, if an agent is searching for an orange, the database will be interrogated with a filter composed with the *spherical shape* and *orange color* properties. Moreover, only objects that are in the visual field of the autonomous agents are kept for the final exploration. Depending on its behavior, each autonomous agent is able to decide how to compose and modify the interrogation filter. Finally, each agent has its own memory and has the capacity to remember or forget objects properties.

Kuffner *et al.* [70] have also proposed a pure top-down visual attention model dedicated to simulate navigation of autonomous agents in 3D VE. This system is mainly used for collision detection and avoidance with static and dynamic objects populating the VE. For this, each object visible by an agent is recorded in his personal memory, together with their speed, observation date, etc. Each agent is subject to oblivious based on the last observation date of each object. But some objects, important for current task, will be kept in mind longer , e.g. closed doors for a task consisting in finding the exit of a building. With all these information, autonomous agent were able to navigate realistically in the VE.

The pure top-down models presented in this section are effective in their particular context of use. However, these models do not take into account the lighting of the scene, i.e. some objects might not be perceptible for some agent under some particular lighting configuration. This is the drawback of

purely top-down visual attention models: they do not take into account perception. This problem emphasizes the importance of using hybrid visual attention model, implementing both bottom-up and top-down components.

1.2.2.3 Hybrid visual attention modeling

Hybrid visual attention models implement both component of human visual attention. These models are able to simulate more complex gaze behaviors than purely bottom-up or top-down models due to the fact that they take into account humans visual reflexes and cognition.

In a game scenario, Sundstedt *et al.* [116] have shown that a saliency map alone is not sufficient to efficiently compute user's gaze. Even without an explicit task, users will automatically assume a task by themselves. This is probably a consequence of the fact that gaze is highly influenced by the top-down component, e.g. the task to achieve [127]. As suggested by [55], it seems that a saliency map only suggests a set of potential gaze locations and that another higher level component, i.e. top-down, may choose a gaze position in this set. As a result, it seems necessary for a visual attention model to simulate and combine both bottom-up and top-down components [116].

To our best knowledge, the first hybrid visual attention model is the one proposed by Wolfe *et al.* [125]. They have improved the computation of the saliency map by weighting feature maps based on current task. For instance, for a task consisting in searching red object, the feature map containing red data will have a higher weight than other feature maps. Also, this work might be improved using recent findings in physiology stating that bottom-up attention is fine grained, i.e. there exists several weights for a single visual feature [90]. As an example, when searching for a low intensity red object, high luminosity red objects may be inhibited by the top-down component.

As suggested by the MOT theory [108], hybrid visual attention models often contain a part that computes object-based attention. For instance, Sun and Fisher [114] proposed to organize visible object in a graph according to their relations, e.g. spatial, in a picture. For example, the *sea* node of the graph may have a *boat* child which itself have several *sailor* childs. This type of graph helps to control hierarchical orientation of the gaze. For instance, the gaze may only go down the graph under the *sea* node only if the current task is to find a fisherman in a perceived image. In this model, the bottom-up process is taken into account by modulating each node relevance to current task by the mean of all saliency values of each pixels inside the node area in the image. This type of modulation is implemented to stick to the MOT theory and is used in more recent models such as in [73].

The use of hybrid visual attention model in virtual environment has been studied by many researchers. Several of these researches have proposed the use of new components. For instance, Longhurst *et al.* [77] introduces the habituation phenomenon which simulates the fact that we become oblivious to visual objects. In this case, the top-down attention on objects will decrease with time. Lee *et al.* [73] proposed another top-down component called the spatio-temporal context. The spatial context is higher when the object is close to the screen center and when the object is within a certain range of distances to the viewer. The temporal context reflects the fact that the values of the spatial context are high and constant during a certain period. Many models computing attention in VE also include a feature that could be considered as the most important concerning the top-down component: task relevance of visual objects [23] [73] [21] [22]. Actually, a semantic value is given to objects according to their relevance to current task: the higher the relevance, the higher the semantic value is. The use of a task weight in visual attention model has been reported as being very effective in [21] [22]. All presented top-down attention value are linearly combined into final top-down attention level. In the final step of each of the presented model, the final bottom-up saliency is modulated with the final cognitive top-down value,

either per pixel [23] in image space or per object [73] [22], to result in the final estimated attention value.

1.2.2.4 Real-Time Tracking of Visually Attended Objects in 3D virtual environments

Surprisingly, to the best of our knowledge, only one research has been dedicated to the use of visual attention models for real-time attention prediction during exploration of 3D VE. Indeed, the original model of Itti could be used but it would only take into account what is visible on the screen. More advanced hybrid visual attention models such as the ones proposed in [77] [23] [22] could also be used but they lack some specific attention component adapted to the context of first-person navigation. Indeed, only the visual attention model proposed by Lee *et al.* [73] has been specifically designed for this context.

This model is based on the MOT theory [108] and efficiently uses the graphic hardware to achieve real-time performance. As described in the MOT theory, it does not compute a gaze point position on the screen but returns the object, or set of objects, that could potentially receive more attention from the user. To do so, a visual attention model must simulate both the bottom-up and top-down components of the human visual system.

In the model of Lee *et al.* [73], the bottom-up part is simulated by computing a saliency map using luminance, hue, depth, objects' motion and on-screen objects' size as visual image features. Then, the bottom-up saliency level of each object is computed as the mean of the saliency values of pixels belonging to it on the screen. Finally, objects have their saliency value modulated by a top-down factor. The top-down factor is computed using spatial and temporal context of objects. The spatial context is higher when the object is close to the screen center and when the object is within a certain range of distances to the viewer. The temporal context reflects the fact that the values of the spatial context are high and constant during a certain period. This model is able to predict the object gazed by the user 48% of the time during free navigation and 62% of the time during a research task involving navigation in static and dynamic VE. As future work, it is suggested to take into account the novelty of objects and other top-down contexts [73].

1.2.3 Summary

To sum up with, the goal of a visual attention model is to simulate the visual reflexes and cognitive processes taking place in the human brain. The bottom-up component is based on the image displayed on the screen with attention being computed per pixel [57] [69] [125]. The top-down component is based on higher level data, i.e. habituation or task, with attention level computed either per pixel [23] [77] [22] [128] or per visual object [73] [114] [14] [70]. The visual attention model finally outputs an attention value for each area on the screen or per visual object: areas or objects having a high attention value are likely to receive more attention than others. However, few of these visual attention models are designed for real-time use in 3D VE.

As reported in this section, many visual attention models have been proposed in the literature. Indeed, this is an active research area due to the fact that attention prediction is useful for many applications.

1.3 Applications using gaze point and visual attention models

Several uses of user's gaze point have been proposed by researchers. Indeed, this feature can give developer several advantages to improve navigation, interaction as well as to accelerate the rendering process of virtual environments.

Over the years, visual attention models have become more and more popular and are now used in several domains for several purposes. For instance, they are used to distribute available computational resources in order to efficiently render high quality lighting solution of a virtual scene or to improve the realism of virtual avatar animations and behaviors.

1.3.1 Human-computer interaction in desktop applications based on the gaze point

First, the gaze point can be used to improve applications control. It has been reported as a natural and convenient interface for human-computer interaction [60]. Jacob has describe two styles of interaction based on the gaze point [59]:

- command based: users have to validate their actions/choices using a command, e.g. by pressing a button.

- non-command based: the software analyzes user's gaze behavior and automatically guess user's intention.

Non-command based gaze interfaces are subject to the *midas touch* problem [59] related to the fact that non-intentional gaze pattern can be interpreted as a command by the system. These unwanted actions annoy the user and make him feel like he do not control the application. Several solutions can be used such as a validation step before the action is effectively validated [58] such as an eye-blink or a button pressed. Indeed, by forcing the use of blinks, the interface becomes unatural.

In the case of 2D interfaces, Salvucci *et al.* [107] have improved a WIMP interface by always positioning the mouse pointer at the level of the gaze point. Experimental results show that this type of interface result in an increase of selection performances for half participants. Furthermore, it seems that participants were easily learning how to use efficiently this system. Zhai *et al.* [130] have proposed two more advanced methods, called liberal and conservative combining gaze tracking and mouse pointing. Using the liberal method, the mouse pointer is wrapped on the position of the gaze point. Then, the user can freely move the mouse pointer inside a circle of confidence centered on the gaze point. Using the conservative method, the mouse position is not wrapped at the position of the gaze point but on the edge of the circle of confidence. The position is in fact the intersection between the circle and the line from the gaze point to the position of the mouse just before the user started to move. The authors have shown that the liberal method was really appreciated by participants, more than the conservative method, and that they had the feeling to selected targets faster. Concerning performance, the liberal method allowed participants to select items faster than traditional mouse method.

Vertegaal and Fono [121] also improved a WIMP interface by changing windows size: the current window of interest get the maximum size whereas other windows have their size reduced, i.e. compressed, to the edge of the screen. Bolt [13] also proposed such a system to easily parse and view multiple videos displayed on a screen. In this paper, the method was presented as an *orchestration*.

Several other interfaces have been proposed for multiple tasks. For instance, Duchowski *et al.* [32] proposed a gaze based internet navigation system. In this system, users were able to select links with their gaze and to easily go back to the previous web pages. To conclude with the use of gaze in 2D interfaces, Isokoski *et al.* [54] allows users to write textual data using their gaze allowing disabled people to communicate.

1.3.2 Use of the gaze point in 3D interactive applications

1.3.2.1 Human-computer interaction

Navigation in computer games is commonly achieved using either the mouse and keyboard combo, a joystick or a joypad. Recently, some researches were conducted on using user's gaze as a controller to navigate in virtual environment. Both first-person and third-person navigation have been investigated.

In the case of third-person navigation, the user is traditionnaly pointing then clicking where he wants his avatar to move to. Smith *et al.* [111] have proposed to replace the mouse pointer by user's gaze position in the game Neverwinter Night [53]. Using this method, 83% of participants felt more immersed in the game environment and 67% have found this interface more natural and easy to understand than traditional point-and-click. Indeed, when communicating, humans use naturally their eyes when they want to show something. This may explains why this gaze-based communication with the avatar seems more natural.

In the case of first-person navigation, Smith *et al.* [111] have also proposed to remove the use of the mouse using the gaze point. Using their method, movements on the ground are still achieved using the arrow keys on the keyboard but camera orientation is based on gaze orientation. Actually, the camera orientation is constantly changed in order to have its forward vector matching the gaze direction in the game environment. Thus, the point the user is looking at is constantly moved to the center of the screen. Despite the fact that the *midas touch* problem was encountered by participants, i.e. they can't look at an area without activating a camera rotation, 83% of them felt they were more immersed in the game environment. Finally, in a first person-shooting game, Jonsson[63] proposed to orient player's weapon toward the direction he is looking at. Thus, players were able to aim and fire at the 3D position they were gazing at: all participants to this experiment have preferred this type of first-person shooter interaction.

1.3.2.2 Perception based rendering

Many researchers have proposed to exploit the gaze point feature to manage LoD when rendering complex sets of data. In these cases, LoD managment reduces the complexity of the rendered data without the user noticing it. These methods are mainly designed to increase the rendering speed.

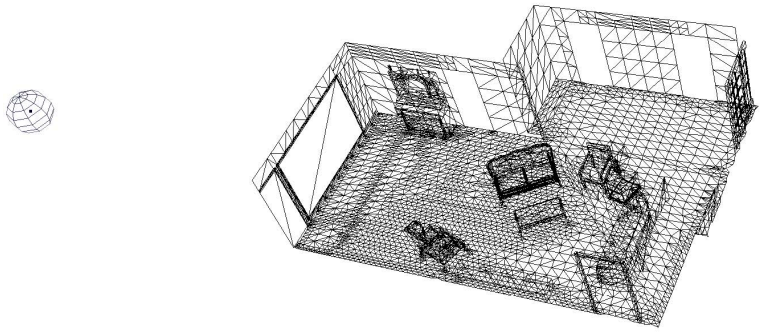


Figure 1.11: Perceptual simplification of a 3D mesh based on the gaze point (sphere on the left side) [78].

Ohshima *et al.* [92] have proposed to reduce the level of details of rendered objects as a function of their on-screen distance to the user's gaze point. First, for each visible mesh, their system computes a perceptual fidelity value based on eccentricity, velocity, and fusion properties of the human visual

system: less details are perceived for objects at the periphery of the field of view, moving fast or far from the distance to which eyes converge. This fidelity value is finally used in order to choose an appropriate LoD. Instead of a mesh based LoD hierarchy, Luekbe *et al.* [78] proposed to use a continuous simplification of 3D meshes (Figure 1.11). The authors exploit human blindness to spatial frequency changes at the periphery of the field-of-view [19]. An experiment shown that modification to meshes were not perceptible, thus validating their perceptual simplification method.

Rendering polygonal mesh is not the only time consuming task: volumetric rendering is also an expensive task. Levoy [74] have developed a ray-casting algorithm in order to accelerate the rendering of volumetric data. In this method, the number of rays cast per pixel, i.e. rendering quality, decreases as a function of its distance to the gaze point. To further increase rendering speed, a lower resolution version of the volumetric data, i.e. mipmap level, will be accessed for pixels far away from the gaze point. Using this perception-based rendering method, frame-rate increased from 13 to 59.6 frame per seconds.

1.3.3 Use of Visual attention models for data processing

1.3.3.1 Perceptual data visualization

Multimedia applications have to manage more and more data whether these are images, sounds or videos. Also, data overview is often used before full data are loaded in the computer memory for a complete visualization. Several methods based on visual attention models have been proposed to automatically process these data and to produce fast and smart pre-visualization. Then, once a data have been selected for full visualization, it is important to efficiently load it and ease its visualization.

Chen *et al.* [27] have proposed a method in order to automatically detect and visualize important parts of large images, i.e. HD pictures or panoramic views, on small display peripherals, i.e. smart phones and PDAs. This method uses a visual attention model to identify image parts containing visually and semantically important objects. The top-down part of human visual attention is modelized using vision algorithm detecting human faces and textual information. Small part of the large picture are finally extracted and exposed to the user. In the contrary, Liu et al. [75] use Munsell color space to segment the original picture instead of vision algorithm. Then, each identified objects are sorted based on the mean of the saliency value of each pixel belonging to them in the saliency map. Finally, small pictures containing visual objects are extracted and exposed to the user.

Automatic generation of preview pictures from complex 3D models is a different problem than the picture case. Indeed, generating preview pictures from a 3D model to show its most important features is not an easy problem. Lee et al. [72] have proposed a method computing a saliency map on the mesh surface using its local curvature and saliency. A gradient descent method is finally used to find several local maxima. These maxima are then used in order to position and orient a virtual camera and take a screenshot of the model. These screenshots constitute the final set of preview pictures. This model outperforms a previous model based only on curvature.

Displaying large scaled virtual environment is a complex task, mostly due to the fact that all data cannot fit in computer memory thus requiring streaming. Streaming method often predict data to load using camera trajectory [6]. Beeharee et al. [9] proposed to take into account human attention as an additional information in order to stream data where the user is more likely to look. This method results in a better use of the available network bandwidth.

1.3.3.2 Perceptual data encoding and simplification

Multimedia data take more and more space and memory, e.g. HD videos. Even with efficient compression algorithms, high quality data requires more and more storage space. In order to reduce this constraint, visual attention models have been employed in order to perceptually encode and/or compress data such as picture or video.

In the case of video processing, Cheng *et al.* [28] have proposed a framework to automatically compute region of interest. In this method, a visual attention model [57] is applied on each frame. Visual motion is taken into account by using video motion data, i.e. 2D tensors. Then, more storage space can be used to encode more visual details in perceptually important regions. Thus, the bandwidth used can be reduced by removing details in less important regions, i.e. region that are not likely to be gazed at. Another approach is to maintain a constant bandwidth and to compute a better distribution of visual details to increase overall video quality.

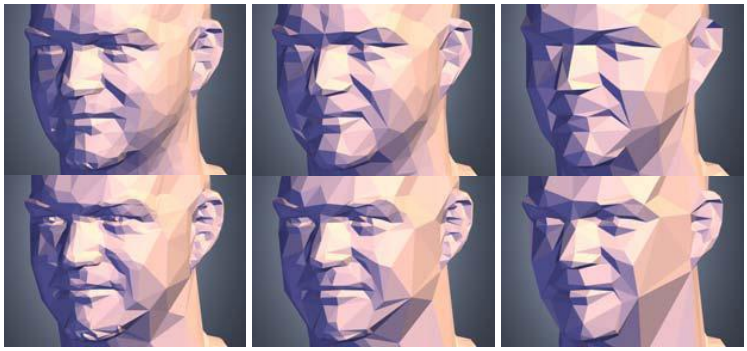


Figure 1.12: Perceptual 3D mesh simplification. Top: using a non-perceptual state-of-the-art method. Bottom: using the perceptual approach proposed in [72].

Concerning 3D models, it is important to reduce the number of triangles in order to achieve real-times frame rate. Mata *et al.* [79] propose a simplification method based on a bottom-up model [57]. Indeed, the saliency value is used to guide a continuous simplification algorithm resulting in high simplifications on non salient areas. Lee et al. [72] have also proposed such a method. By using visual feature more correlated with the simplification task such as curvature. As a example (Figure 1.12), their method is able to efficiently reduces a model by keeping visually salient and relevant features such as nose, eyes, mouth or ears.

1.3.4 Use of Visual attention models for 3D virtual environments rendering

1.3.4.1 Perceptual high-quality rendering of illuminated virtual scenes

The rendering of realistically illuminated virtual scene is one of the hardest challenge in computer graphics. This problematic is mainly encountered in the movie industry for the purpose of full CG movies or special effects rendering. The full rendering of a CG movie can be weeks long. Several solutions exist to overcome this problem. The first one is to use cloud-computing by distributing several part of the movie, or a scene, over a cluster of computers. More recently, researchers have proposed a novel solution which is to take into account human perception when rendering a virtual scene. This solution relies on the fact that the human visual system is not perfect [21] [22]. Thus we will not notice lower quality on the part of the scene we are not looking at. Indeed, in this solution, visual attention models are used to deter-

mine where humans are more likely to look in a virtual scene in order to better distribute computational resources.

Yee *et al.* [128] have proposed a framework taking into account the spatio-temporal sensitivity of the human visual system. Firstly, a saliency map is computed similarly to [57]. Secondly, The authors compute a spatio-temporal error tolerance function based on the contrast/sensitivity function and motion, i.e. human eye are less sensible to contrast on moving object. Finally, it is combined with the saliency map and the aleph map are combined to form the *Aleph* map which store. This aleph map represents the per-pixel level of quality required for humans to not perceive a difference. Actually, the quality value will be used to control the number of iterations used to compute global illumination using a luminance caching method. Even with the time required to compute the aleph map, this method accelerates the computation of the global illumination of a scene from 2 to 9 times without any perceptible loss of quality by humans.

Yee *et al.* [128] system is based only on the bottom-up part of the human visual system. Cater *et al.* [21] [22] proposed a new method tuning the rendering quality by exploiting the top-down part of the human visual system. Using an experiment, authors have shown that it is possible to exploit human inattentional blindness when a task is involved [21]. Indeed, in this case, humans will mostly look at objects that are related to the task [127]. Finally, based on the top-down attentional component, a framework similar to the one in [128] have been developed. In this case, the per-pixel level of quality is tuned using on a task map representing per-pixel attentional level based on current task, objects relevance to the task and point of view in the scene.

A method exploiting both the bottom-up and top-down part of human visual system have been proposed by Chalmers *et al.* [23]. This method first computes a saliency map [57]. Second, the top-down object relevance to current task is computed and updated based on the time they are display on the screen to simulate human habituation. An importance value is computed for each pixel on the screen based on the position of relevant objects and their relevant value. During this step, the reflection of important objects is also taken into account. Indeed, this method is similar to the one proposed by Longhurst et al. [77] and both these methods are also accelerated using GPU.

The rendering of high quality participating media is also a difficult challenge in computer graphics. Light diffusion and absorption is highly time consuming process. In this case, visual attention models based on saliency map have also been proposed [3].

1.3.4.2 Real-time virtual avatar attention simulation

Thanks to the power of highly parallel architectures such as GPU, visual attention models can now be executed in real-time applications. Several real-time 3D application can benefit from using a visual attention model, for example, to realistically animate autonomous agents or attention prediction.

Simulating believable virtual avatar is a complex task. Even if we would have access to perfect rendering and animation methods, the avatars will not look realistic if its behavior does not look natural. To this aim, several researchers investigated the use of visual attention models to increase avatar's behavior and gaze naturalness in virtual environment. For instance, Courty *et al.* [30] proposed to change the orientation of avatar's head based on the saliency map. This saliency map is computed only from the depth map of the virtual environment from avatar's point of view. The saliency map is attenuated at its borders to simulate the fact the humans perceive less details at the periphery of their field of view. Regions of interest are set to the area of the saliency map having a high saliency value. In the presented demo, the avatar was walking in a street and his head was oriented toward the region having the highest saliency value.

Peters *et al.* [93] have proposed the same concept but they were using the model of Itti *et al.*[57]. In this case, thanks to the simulation of inhibition-of-return, the gaze behavior was more realistic since the avatar was prevented to gaze two times in a row at the same area. Also, the proposed gaze behavior model was able to simulate saccades and fixations. A more advanced model have been proposed by Itti *et al.* [56]. This model is the same as [93] but it involves a full simulation of head motion together with complex eye movements such as saccade/fixation, smooth pursuit and eyes blinking. Also, realistic animation of the face was achieved using the facial animation method developed by Pighin *et al.* [95]. More higher level avatar behavior such as perception and realistic navigation using visual attention have been proposed. Please refer to Section 1.2.2.2 for more details.

1.4 Conclusion

This chapter exposed a synthetic survey of the human visual system, the methods to compute human attention and their applications. In this regard, we have first detailed how visual information are processed from the eyes to the visual cortex in the brain. We have shown that the human visual attention is made of complex processes separated in two components: the bottom-up component representing visual reflexes and the top-down component representing the cognitive process that takes place in the human brain, e.g. knowledge, memory, or habituation. Indeed, human attention is driven by a complex mix of these two components exploiting visual data from the retina to the end of the visual pathway, i.e. the visual cortex. The human visual system is also limited by several of its components, for instance, loss of color perception in low luminance environment and loss of accuracy at the periphery of the field of view and on moving objects. To perceive their environment, humans have to move their eyes and we have reported several of these gaze patterns when simply looking at a picture or when walking. Then, we have exposed two different means to evaluate/compute human attention.

The first possibility is to use a gaze tracking system computing a single gaze point: the point the user is looking at. The second alternative is to simulate human attention using a complex mix of bottom-up and top-down attention simulation. Such a visual attention model would compute attention distribution in a scene from which a gaze point could be computed. Finally, we have surveyed gaze point and visual attention applications, with a particular focus on computer applications. We have seen that the gaze point information is mostly used in real-time applications to improve interaction or for the purpose of level of details to accelerate the rendering process of 3D applications. In the contrary, visual attention models are used in off-line applications, i.e. Perceptual high-quality rendering of illuminated virtual scenes and data pre-processing, as well as in on-line applications, i.e. realistic avatar animation and user attention computation.

This first chapter has highlighted a separation between gaze tracking system and visual attention model. However, each of these methods have their strengths and weaknesses. It would be interesting to combine these two methods in order to improve human attention computation. Concerning visual attention model in the context of user attention computation, it has been reported that it should be mandatory to mix bottom-up and top-down visual attention simulation [116]. Furthermore, it has been suggested that it is important to develop and evaluate new visual attention components adapted to the context in which the visual attention model is used [73]. Finally, we have shown that few visual attention model have been designed to compute and predict human attention in virtual environment [73] [116]. These three points emphasizing a need of improvements in the computation of human attention are addressed in this manuscript and more particularly in the three next chapters gathered within Part 1: Visual attention

models for gaze tracking and attention prediction.

Several 3D applications have been developed to take advantage of the user's gaze point in order to apply a perceptual level of detail algorithm simplifying the virtual environment without the user noticing it. This simplification is a strategy used to accelerate the rendering process. In contrary, no research has been conducted on the use of user's gaze point in order to improve their perception of and immersion feeling in virtual environments. Such an effect could be a depth-of-field blur to increase accuracy concerning depth/distance perception. Indeed, this effect is used in several contemporary games [112]. However, the preference of gamers concerning this effect has never been studied when it is not adapted to their gaze point position on the screen. Furthermore, the use of a gaze tracking system to compute the gaze point position could increase preference of gamers toward the activation of this kind of effect. These two points are addressed in the two final chapter of this manuscript gathered in Part 2: Improving visual feedback in 3D virtual environments.

Part I

Improving Gaze-Tracking in 3D Virtual Environments using Human Visual Attention and Gaze Behavior

Chapter 2

Analysis and modeling of gaze behavior during first person navigation in 3D virtual environments

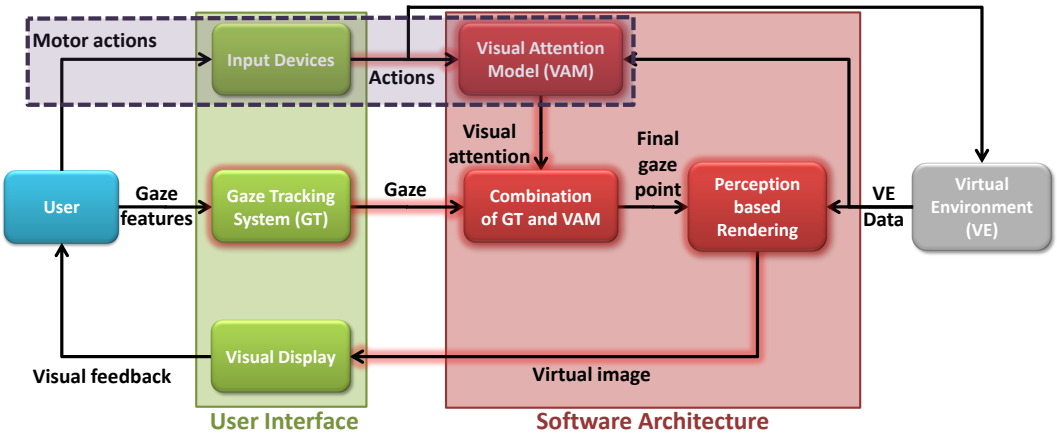


Figure 2.1: Analysis and modeling of human gaze behavior during first person navigation in visual attention models.

2.1 Introduction

Walking on the streets or piloting an air plane do not involve the same human gaze behavior. For instance, specific gaze behavior has been observed during real pedestrian walk [39]. In VR, a visual attention model can be used to predict human gaze. Moreover, it has been suggested that using attentional components adapted to the context in which the visual attention model is used could lead to better results [73]. However, none of the existing visual attention models take into account the specificity of the human gaze behavior when walking in VE. Thus, there is a need to study human gaze behavior when walking in VE using a first-person view.

In this chapter, we want to answer to the following research questions:

1. Can we observe the same gaze behavior in a virtual environment (using input peripherals such as keyboard and mouse) as in previous studies conducted in the real world?

2. How are users' gaze points related to input controls?

3. Can we design a model to predict users' gaze positions in VE using their actions on peripherals?

First, we have studied and analyzed the gaze behavior of humans walking in a virtual environment using a first-person view. Our study was conducted with participants navigating using a keyboard and a mouse. In this chapter, we report on the gaze behavior we have observed on users walking straight forward and taking turns in a VE. We then compare this behavior to the ones observed during real pedestrian walks.

Second, we have proposed three simple gaze prediction models taking as input: (1) the motion of the user as given by the rotation velocity of the camera on the yaw axis (considered here as the virtual heading direction), and/or (2) the optic flow on screen. We finally report on the performance of these models and discuss their possible future uses.

The chapter is organized as follows: Section 2.2 details the experiment we have conducted to study users' gaze behavior when walking in a VE. The results of this experiment are reported in Section 2.4. In Section 2.5, based on these data and on observations, we propose 3 models to estimated gaze position. Finally, in Section 2.6, we evaluate and discuss the accuracy of these models.

2.2 Analysis of human gaze behavior during first-person navigation in 3D virtual environments

In this section, we describe the experiment conducted to measure and record the gaze behavior of participants navigating and taking turns in a VE.

2.2.1 Population

Ten naïve volunteer participants (9 males, 1 female) with a mean age of 25.8 (SD=1.7) participated in our experiment. We have only accepted participants that were familiar with the first-person navigation paradigm and had a normal or corrected-to-normal vision.

2.2.2 Experimental apparatus

During this experiment, we used the Tobii x50 gaze tracker to compute participants gaze positions at 50Hz. Participants were positioned in front of a 19" flat screen at a resolution of 1280×1024. The screen was 37.5cm wide. Participants' heads were positioned on a head-support to ensure that they remained at a distance of 60cm from the screen. The virtual environment was rendered in real-time at a constant frame-rate of 75Hz.

The navigation in the virtual environment is achieved using first-person viewing mode. In this case, the virtual camera is placed at the eye level of the user's avatar, as in first-person shooter games. The control mechanism allowed two degrees of freedom: walking forward/backward and changing the horizontal camera orientation, i.e. yaw viewing direction. Walking forward or backward was achieved using the up or down arrow keys of the keyboard. Changing camera horizontal orientation (yaw angle) was achieved using horizontal movements of the mouse. Mouse correction and filtering (available under the operating system to improve selection) were disabled. A horizontal mouse movement of 2.5cm on the table

resulted in a rotation of 90 degrees of the camera in the VE (36°/cm). Vertical movements of the mouse were not taken into account and the camera was always oriented toward the horizon. Avatar properties were inspired of real world data : height was 1.75 meters and walking speed was 1.15 m/s [39].

For each virtual environment used in our experiment, the ground was flat and the ceiling was 3-meter high. Each component of the VE was rendered using the same noisy texture. This reduces the probability of having some parts of the VE more salient than others, in order to make gaze data as independent from textures as possible. No light sources were involved in the rendering process: only constant ambient lighting was used. These choices allow participants to perceive surfaces' orientations, depth and optic flow when navigating. Before each navigation session, the Tobii gaze tracker was calibrated in order to ensure constant and accurate gaze tracking. The whole experiment lasted around 30 minutes.

During each session of the experiment, we recorded several data: time, camera position, yaw angle of the camera and 2D gaze position on the screen as well as 3D gaze direction in world space. Horizontal gaze position on screen is a floating value ranging from -1 (left) to 1 (right). We also recorded participants' actions on the keyboard (forward and backward translation) and on the mouse (yaw rotation). This data allowed us to replay each session in order to analyze participants' gazes during navigation.

2.2.3 Experimental procedure

Before the experiment started, participants navigated in an empty VE as a training session. Then, for each participant, the experiment was divided into three parts. The data of the first part were used to analyze gaze behavior and to build the gaze prediction models. The data of the second and third parts were used to validate the models.

2.2.3.1 Data collected for gaze behavior analysis

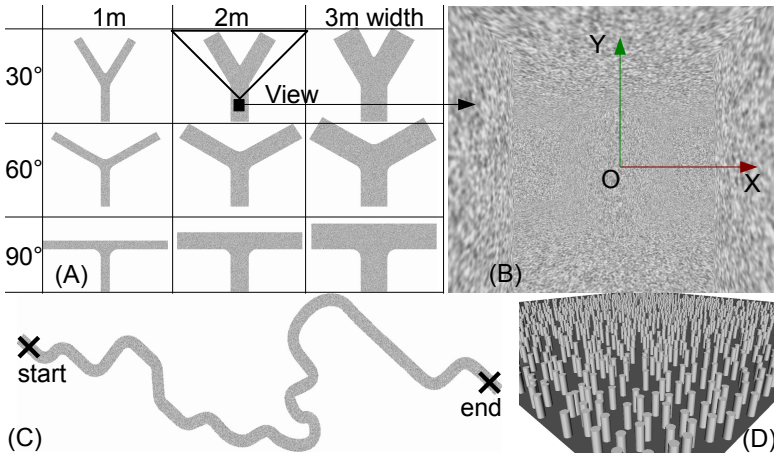


Figure 2.2: Virtual environments used during the experiment. (A): Top view of turns with different angles and widths. (B): Screen view of first-person navigation for one turn condition (30 degrees, width=2m). (C): Top view of the tunnel path. (D): Perspective view of the room used with randomly positioned posts.

During the first part, participants were asked to take virtual turns. As shown on Figure 2.2A, several turns were proposed with different angles and widths. Turn angles have values of ±30, ±60 or ±90 degrees and width of tunnels was either 1, 2 or 3 meters. Our experiment involved three navigation conditions:

- *Active*: the turn direction was indicated to participants before the navigation started. Then, participants navigated using the keyboard and mouse to the end of the trajectory.
- *PassiveK*: participants watched a replayed navigation sequence. The turn direction was indicated to participants before the navigation started. Thus, participants knew the future turn direction.
- *PassiveNoK*: Participants watched a replayed navigation sequence. The turn direction was not indicated to participants. Thus, they had no prior knowledge about the turn direction.

Each navigation condition was randomly presented two times to each participants. For each presentation, 18 turn configurations (6 angles, 3 widths) were randomly presented. In conditions where the navigation direction was indicated to participants, a green signboard was displayed on the screen with a red arrow indicating the turn direction for two seconds. When the signboard was displayed, navigation commands were disabled. Each navigation session lasted around 2 minutes and 30 seconds.

2.2.3.2 Data collected to validate gaze prediction models

During the second part, participants were asked to actively navigate inside a tunnel until they reached its end (Figure 2.2C). This 2-meter width path was composed of turns with various angles. This session consisted in an active navigation and lasted about 1 minute.

The third part of the experiment consisted again in free navigation. The virtual environment was a wide square room with several randomly positioned posts (Figure 2.2D). Participants were asked to navigate freely in the room. This session consisted in an active navigation and lasted 1 minute and 30 seconds.

2.3 Analysis of gaze behavior

In this section, we analyze data collected in the first part of the experiment. We describe observed gaze behavior of participants during active navigation and compare our results to previous work conducted in the real world. We also compare participants gaze behavior between active and passive navigations.

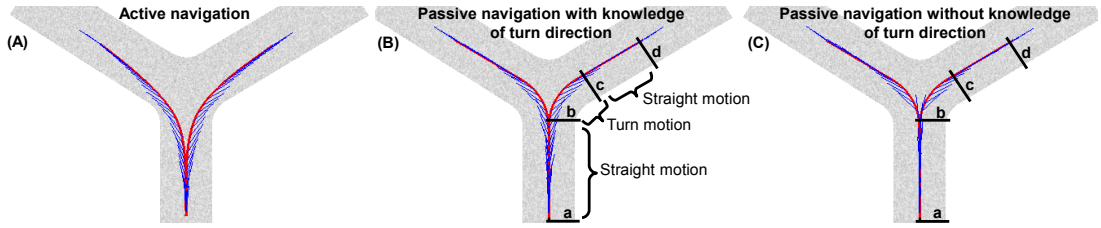


Figure 2.3: Mean path and gaze direction of participant during: (A) Active navigation, Passive navigation with (B) and without (C) knowledge of the trajectory. On this example, angle of turn is ±60 degrees and the path width is 2 meters.

2.3.1 Visual flow and focus of expansion

Before analyzing the gaze behavior of human walking in VE, it is important to understand visual features generated by a walk. In this section, we consider two relevant features: the optic flow and the focus of expansion.

Walking in an environment induces a motion on the retinal image called the optic flow. Optic flow provides information about self-motion direction and results in eye movements to stabilize the perceived image on the retina [91]. An important feature of the optic flow is the focus of expansion (FoE). The FoE is a point in the retinal image where "all visual motion moves away" [91]. For instance, when walking straight forward in a VE, the FoE will be at the center of the screen and the optic flow will radially expand from this point.

Niemann *et al.* [91] reported that when passively looking at a navigation in a textured VE, users tend to look near the FoE. Fukuchi and Koch [36] also report an attentional bias toward the FoE when zooming on natural scene pictures. Their results suggest that the FoE may play a role in human attention. This behavior has also been noticed in real situations with bicycle and car drivers [104]: 90% of all saccades were oriented to a circle of ±4 degrees radius of the visual field centered on the FoE. With such a visual strategy, the driver could collect information about the road given the fact that the FoE is along the moving direction.

2.3.2 Gaze behavior during active navigation

In this section, we analyze descriptively the gaze behavior of participants in the Active condition, i.e. when they controlled their avatar using a keyboard and mouse.

Firstly, it appears that participants' gaze systematically deviated toward the future trajectory direction as found by Grasso *et al.* [39] during real walking. This anticipatory gaze orientation behavior is visible on Figure 2.3 and on Figure 2.4. For small width (1 meter), the gaze was oriented toward the upcoming direction nearly 1 second before the camera started turning. Then, the larger the path, the earlier the gaze was oriented toward the future direction since participants were able to see the far-end of the trajectory earlier.

We can descriptively separate participants into two groups: those who apply high rotations to the camera punctually (Figure 2.4, Subject 5) and those who apply continuous rotation to the camera (Figure 2.4, Subject 1). However, when turning, participants always look toward the direction they want to move to. At the end of each turn, gaze returns to baseline, i.e. at the center of the screen.

In our VE, walls delimit the left and right boundaries of each curved path. In some cases (mainly 90 degrees turns), we noticed that participants often looked at the tangent of the wall in the direction that they were navigating to. In Figure 2.3A, gaze seems indeed oriented toward the tangent of the wall inside the curved path. We call it *border* since it is a depth discontinuity between the near and far walls from the current point of view. By analyzing all sequences of the participants, we noticed that 69.9% of all gaze points (75.8% when turning only) contained a border into a circle of ±4 degrees radius of the visual field (Table 2.1). This could be due to the fact that this is the area where participants progressively discover the path they were going to take [91];

Concerning optic flow on screen, we found that 68.7% of all gaze points (44.3% when turning only) were inside a circle of ±4 degrees radius of the visual field centered on the FoE (Table 2.1). This percentage is higher when considering all frames than when only considering turning, since when walking forward, participants often look at the center of the screen, i.e. where the FoE is. These results are consistent with previous findings in real situations [104] which showed that bicycle or car drivers have 90% of their saccades oriented to a circle of ±4 degrees radius of the visual field centered on the FoE. The lower percentage in our case could be due to the fact that turning with a bicycle and a car involves continuous rotations which results in smooth movements of the FoE, whereas, in our case, for participants who applied a high punctual rotation velocity to the camera, the FoE vanished out of the screen and it was no longer possible to gaze at it.

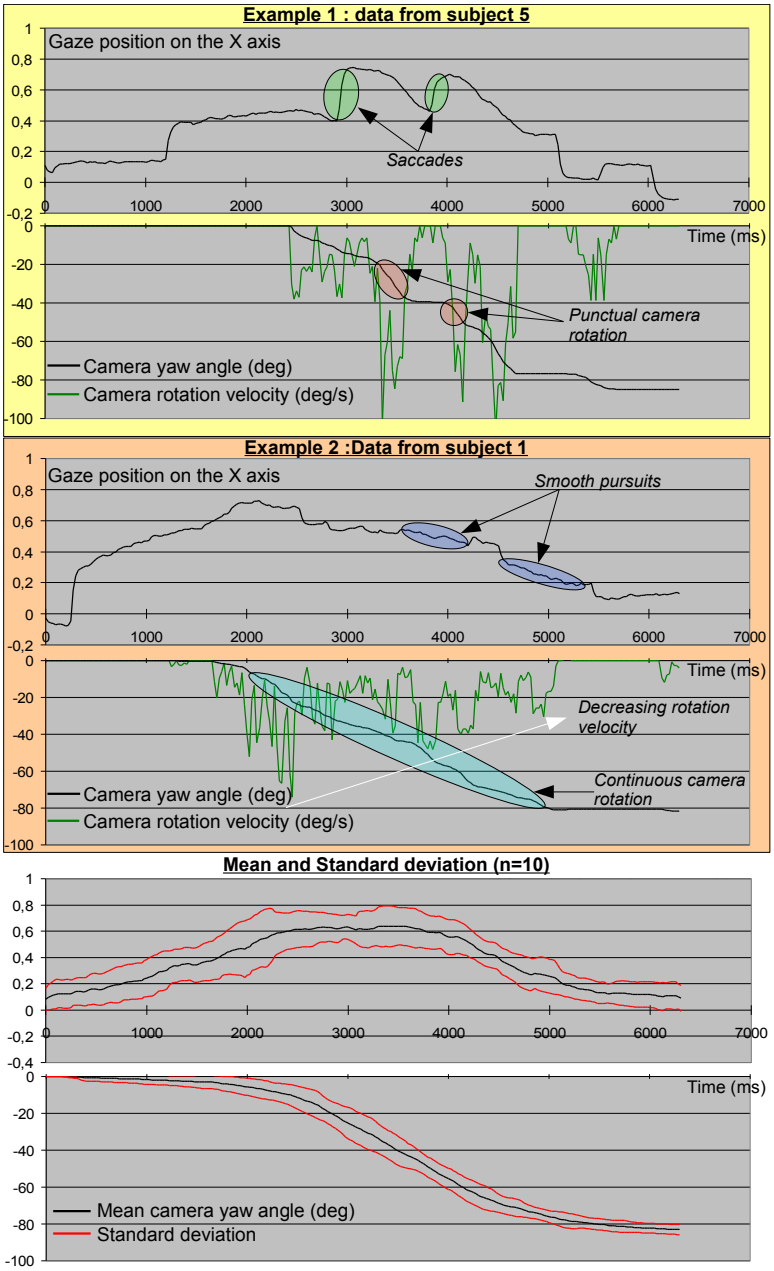


Figure 2.4: Gaze behavior during Active navigation (tunnel width=2m, angle=90°).

Figure 2.5 shows the evolution of gaze positions according to the rotation velocity of the virtual camera in the active condition. It shows strong correlation between the rotation velocity of the virtual camera and the position of the gaze point on the screen. When a left, or right, rotation is applied to the camera, gaze density is higher on the left, or right, of the screen. The effect is symmetric considering negative and positive camera rotations. Furthermore, the higher the rotation velocity is, the closer the gaze point is to the side of the screen corresponding to the rotation direction. As a result, during active navigation, it seems that the rotation velocity of the camera can also be related to the eccentricity of the

gaze point on the screen. On Figure 2.4 (Subject 5), we notice that the farther the gaze position is from the center of the screen, the higher the punctual rotation velocities are. On Figure 2.4 (Subject 1), we also notice that during continuous rotation of the camera, the overall rotation velocity decreases as the distance from the center of the screen to the gaze point decreases. It is assumed that Humans go where they are looking instead of looking where they are going [10]. Our data could corroborate this behavior. It is probably not because there is a rotation velocity to the left that the gaze point will be on the left of the screen. The causality could be the opposite : it is probably because the gaze point is on the left of the screen that participants applied a rotation to the camera to go where they were looking.

	Turning only	Global
FoE	44.3%	68.7%
FoE [104]	N.A.	90.0%
Border	75.8%	69.9%

Table 2.1: Percentage of time the FoE or "border" is located inside a circle of ±4 degrees radius of the visual field centered on gaze point during active navigation. (Percentages for border are computed only for frames where a border is visible)

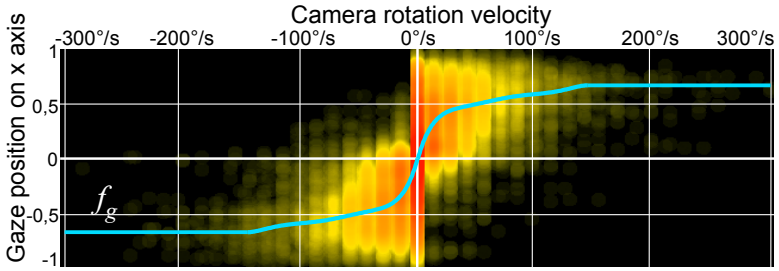


Figure 2.5: Density of horizontal gaze positions on screen as function of camera rotation velocity.

2.3.3 Gaze behavior during passive navigation

In this section, we analyze the gaze behavior of participants in the two passive conditions, i.e. when they only watch replayed trajectories. We could have used replayed sequences of participants but there were too many variabilities (punctual/continuous camera rotation, trajectory, etc). We chose to use an artificial case with a continuous camera rotation.

Two passive conditions were presented: a passive condition with knowledge of the trajectory direction, i.e. incoming turn direction, (PassiveK) and without knowledge of the trajectory direction (PassiceNoK). Sequences of passive navigation can be cut into three sections (Figure 2.3): a first straight walk to the turn (section a-b), a turning part with camera rotation according to a constant angular velocity (section b-c), and a final straight walk to the end of the path (section c-d). The gaze behaviors described in this section can be seen on Figure 2.3.

In the PassiveK condition, we notice a similar behavior as in the Active condition. Participants tend to look in the future direction before the camera began to turn (Figure 2.6, Figure 2.3B section a-b). Interestingly, in the PassiveNoK condition, due to the fact that participants did not know the future turning direction, the mean gaze direction remains aligned with the forward walking vector during the

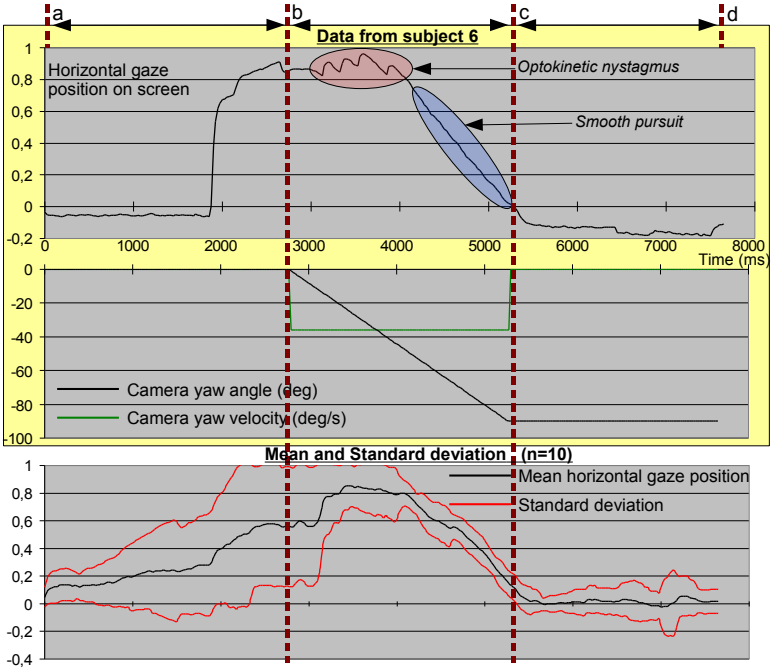


Figure 2.6: Gaze behavior during passive navigation with knowledge of the trajectory (PassiveK) (tunnel width=2m, angle=90°).

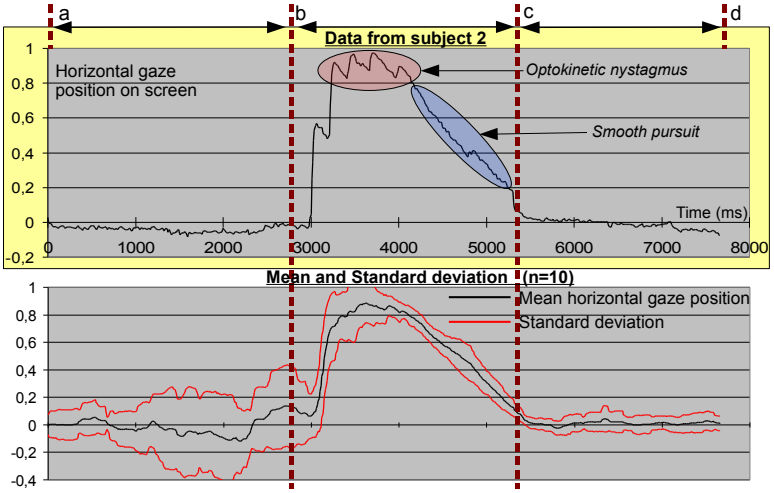


Figure 2.7: Gaze behavior during passive navigation without knowledge of the trajectory (PassiveNoK) (tunnel width=2m, angle=90°).

first straight motion prior turn (Figure 2.7, Figure 2.3 C), i.e. they look at the center of the screen in section a-b.

In the second part of the walk (Figure 2.3 section b-c), when the camera begins to change its orientation at constant velocity, all participants directed their gaze toward the direction of the turn in the PassiveK and PassiveNoK conditions (Figure 2.6 and 2.7). In the PassiveNoK condition, participants instantaneously oriented their gaze in the direction of the rotation once they perceived the first change in

camera orientation (Figure 2.6). Interestingly, the mean gaze behavior, when the camera was turning, is very similar in both passive modes (Figure 2.6, 2.7, 2.3B and 2.3C).

In the final part of the trajectories (Figure 2.3 section c-d), the gaze behavior is again similar for both PassiveK and PassiveNoK conditions (Figure 2.6 and 2.7). As in the Active condition, the gaze is oriented to the center of the tunnel far-end.

Concerning optic flow, we found that 57.6% of all gaze points were inside a circle of ±4 degrees radius of the visual field centered on the FoE in the PassiveK condition, 67.3% in the PassiveNoK condition. The higher percentage in the PassiveNoK condition could be due to the fact that at the beginning, users do not know the upcoming rotation direction. As a result, they look more often at the center of the screen, where the FoE is.

2.3.4 Discussion

Our results show several similarities between navigation in the real world and in the virtual world using mouse and keyboard control. Even without proprioceptive and vestibular cues, we notice similar gaze behavior of participants as reported for instance by Grasso *et al.* [39] in their real-life experiment.

Looking closely at participant's gaze points, we notice several known gaze movements: saccades, smooth pursuits and Opto-Kinetic Nystagmus (OKN). Saccades are very fast rotations of the eyes from one visual target to another (Figure 2.4). Smooth pursuits are continuous rotations of the eyes activated when gazing at a slowly moving target. It is observed for continuous rotation of the virtual camera when participants were looking at the far end of the path (Figure 2.6 and 2.7). OKN is activated when the optic flow elicits a sensation of being rotated and is composed of rhythmic back and forth movements of the eyes, slowly in the direction of the optic flow and fast in the opposite direction. It is visible when the camera is continuously rotating and participants do not find a stable visual target, i.e. in passive mode when the camera started rotating and the far end of the path was not visible (Figure 2.6 and 2.7).

We found that the anticipatory nature of gaze during a real pedestrian walk can also be observed when walking in a VE with mouse and keyboard control and no body nor head motion. This is true for the Active and PassiveK conditions. Interestingly, the PassiveK condition exhibits almost the same gaze behavior of the participants as in the Active condition. Only the PassiveNoK condition is different from the Active condition since, without knowledge of the future trajectory, participants looked more often at the center of the screen at the beginning of the navigation.

We found that several visual features could have influenced participants' gaze behavior, namely: FoE, velocity rotation and presence of border. Considering the Active condition, we found that when they were turning, participants applied a rotation of the camera that, half of the time, positioned the FoE at the level of their gaze position. This resulted in a stable optic flow where they were looking and it is considered as a strategy to easily acquire information about the environment on the navigation trajectory [91]. We also found a correlation between gaze point positions on screen and the rotation velocity applied to the camera on the yaw axis (Figure 2.5). It suggests that the farther the gaze point is from the screen center, the higher the rotation velocity applied to the camera is.

These results are the basis for the development of the gaze prediction models we propose in the following section. These models can be used to improve prediction of users' gaze positions during first person navigation in VE, especially when turning.

2.4 Model of gaze behavior during first-person navigation in 3D virtual environments

In this section, we propose three gaze models to predict gaze positions when turning in a VE. They are built on data gathered in the Active condition of the first part of our experiment. They use two parameters: the yaw rotation velocity and/or the optic flow visible on the screen.

2.4.1 Gaze prediction model based on yaw rotation velocity

As seen on Figure 2.5, the higher the yaw rotation is, the farther the gaze point is from the center of the screen in the direction of the rotation. This camera rotation seems to be applied to align the virtual camera with the gaze direction of the user in order to go where he is looking [10]. Thus, we first propose a simple gaze prediction model computing the horizontal gaze position g_x on screen using only the yaw rotation velocity ω of the virtual camera: $g_x = f_g(\omega)$.

The computation of $f_g(\omega)$ is based on the data displayed on Figure 2.5. Function $f_g(\omega)$ is computed as the best-fit curve on the median values of gaze points as a function of velocities. Due to the discrete nature of collected data, we had to pre-process these data. We first computed several sets containing gaze position according to 15 velocity intervals ranging from $0°/s$ to $150°/s$ with a step of $10°/s$. We also mixed gaze points corresponding to left and right rotations. A best-fitted polynomial curve $f_g(\omega)$ on the median points is found (Equation 2.1), and shown on Figure 2.5, with $a_1 = 0.0048$, $a_2 = 0.1088$, $a_3 = 0.0327$, $a_4 = 0.004$, $a_5 = 0.0002$ and $a_6 = 5 \times 10^{-6}$. This fitted curve has a correlation factor $R^2 = 0.9983$ with the median points and has a central symmetry on coordinates (0,0).

$$\begin{array}{ll} \forall \omega \in [-150, 150], & f_g(\omega) = \sum_{i=1}^{6} a_i \times x^i \\ \forall \omega \in]-\infty, -150[, & f_g(\omega) = f_g(-150) \\ \forall \omega \in]150, +\infty[, & f_g(\omega) = f_g(150) \end{array} \tag{2.1}$$

Thanks to this very simple model, we can finally output the horizontal gaze point position on screen using only the rotation velocity of the camera in degrees per second.

2.4.2 Gaze prediction model based on optic flow

Second, we notice that the optic flow (FoE) seems to play a major role in gaze attraction. Thus, we propose a model to predict gaze position using the optic flow visible on the screen. From this optic flow, attentional weights are computed for each pixel on the screen (Figure 2.8). Then, using these weights, our method searches for the most probable gaze point position on the screen.

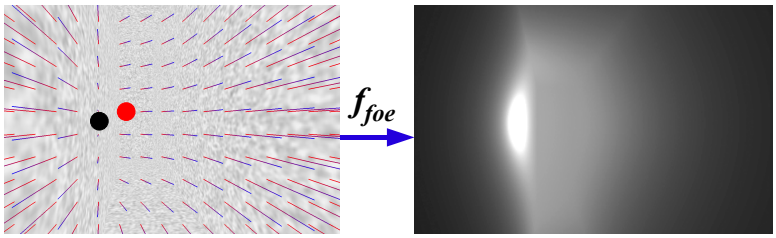


Figure 2.8: Left: optic flow during a left rotation (black and red point are the focus-of-expansion and user's gaze point). Right: corresponding pixel weights W_{foe} computed using f_{foe} function

The model gives more importance to parts of the screen where the visual flow is radially expanding, i.e. where the optic flow has a low intensity. When the user is walking forward, the gaze point would be evaluated at the center of the screen. When turning, the gaze point would be evaluated in the direction of the turn, where the optic flow is low. We can compute the optic flow in screen-space using the method of Rosado [106] developed for the purpose of the motion blur visual effect. This method uses the transformation matrices of the virtual camera for the previous and current frames to compute the position difference of each pixel in view space from frame to frame. This difference is then projected on the screen to obtain a velocity vector considered as the optic flow. For each pixel p, we compute an attention weight $W_{foe} = f_{foe}(p)$ using Equation 2.2 where $V(p)$ is the velocity vector of pixel p and δ is a small value to avoid division by zero. The final value is in the range [0,1]. Finally, the algorithm searches for the pixel with the highest W_{foe} value and positions the final gaze on it. For the sake of performance, the buffer containing per pixel attentional weights can be at a low resolution. In our case, we used a resolution of 160×128.

$$f_{foe}(p) = \min\left(\frac{1}{\delta + \|V(p)\|}, 1\right) \tag{2.2}$$

As a result, this model computes the gaze point position on screen using only the computed optic flow on screen.

2.4.3 Gaze prediction model based on yaw rotation velocity and optic flow

Last, we propose a model combining both information of rotation velocity of the camera and optic flow visible on screen to improve the overall performance.

To this aim, we use two attention weights for each pixel on the screen: one weight based on the optic flow (see Section 2.4.2) and the other weight based on rotation velocity of the camera. Finally, each weight of each pixel is combined as the final attentional value. Using the resulting pixel attention weights, the algorithm searches for the most probable gaze point position on the screen.

2.4.3.1 Computing per-pixel attentional weight based on yaw rotation velocity

The first component of our model computes an attentional weight W_ω of each pixel given current rotation velocity. We define a discrete 2D function $f_\omega(p, \omega)$ which returns the attentional weight W_ω of a pixel p according to its horizontal position p_x on the screen and current rotation velocity ω.

To build the function f_ω, we have discretized the horizontal screen position space (p_x) in 19 vertical parts of equal widths: one at the center and respectively 9 columns for left and right. We also have discretized the velocity space (ω) in 15 parts ranging from $0°/s$ to $150°/s$ using a step of $10°/s$. Finally, for a given velocity, we have computed the probability of finding a gaze point on each of the 19 vertical parts of the screen. As in Section 2.4.1, we mixed together gaze point positions for negative and positive rotation velocities. The resulting function f_ω is presented on Figure 2.9. To get smoother results, we have slightly blurred this function on the horizontal gaze position (p_x) axis. We also normalized each part, corresponding to a range of velocity, in order to get probabilities in the range [0,1] with a value of 1 on the most probable position.

As a result, this first component computes attentional weight for each pixel using only the yaw rotation velocity of the virtual camera.

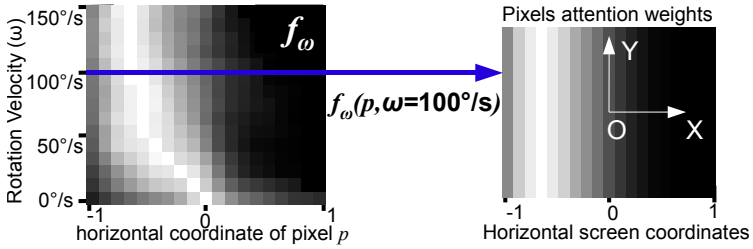


Figure 2.9: Computation of attentional weight of pixels as function of camera rotation velocity. (white color=high attentionnal weight)

2.4.3.2 Combining per-pixel attentional weights corresponding to yaw rotation velocity and optic flow

For each pixel p on the screen, we can then combine the two attentional weights corresponding to: (1) the optic flow visible on the screen $W_{foe} = f_{foe}(p)$ (Section 2.4.2), and (2) the rotation velocity of the camera $W_\omega = f_\omega(p, \omega)$ (Section 2.4.3.1). Using these weights, we compute the final attention level of each pixel as $W_f(p) = W_{foe} \times W_\omega$. Finally, the gaze position on screen is computed as the position of the pixel having the highest attentional value W_{fmax}.

For the sake of performance, the buffer containing per pixel attentional weights can be at low resolution. In our implementation, we used a resolution of 160×128. and all computations were achieved on the GPU.

In conclusion, this third model takes as input the visible optic flow on the screen and the rotation velocity of the camera to compute a gaze position on the screen.

2.5 Evaluation

In this section, we assess the performance of the proposed gaze prediction models. We analyze the efficiency of these models using the data recorded during the second and third parts of our experiment described in Section 2.2.3.2, i.e. navigation inside a tunnel path and in a room. We compare gaze predictions obtained with these models to a control condition where the estimated gaze point is always positioned at the center of the screen.

To compare each model, we evaluate the percentage of time spent by computed gaze points inside a circle area having a radius of 4 degrees of the visual field centered on the recorded gaze points (as in [104]). The results are given in *global*, i.e. considering the full duration of the navigation, and *when turning only*. Results are shown in Table 2.2. *Velocity only* is the model using the fitted curve, *FoE only* is the model only based on optic flow on screen and *Velocity+FoE* is the model combining the camera rotation velocity and optic flow information. The *Center* model is considered as the control condition: the gaze point is constantly positioned at the center of the screen.

2.5.1 Performance

We conducted a repeated measure analysis of variance (ANOVA) with the independent variables being the model used and the dependent variable the percentage of gaze falling inside the circle area [91][104]. For the different comparison analyses, a correction for experiment-wise error was realized by using a

VE / Models	Turning		Global	
	Path	Room	Path	Room
Center	18.35 (3.9)	34.87 (12.7)	32.4 (8.5)	45.9 (12.5)
Velocity only (f_g)	**65.6** (11.4)	**64.8** (12.4)	**48.4** (12.0)	**54.9** (10.7)
Velocity+FoE (W_f)	61.2 (12.7)	62.9 (10.6)	46.2 (12.7)	54.3 (10.5)
FoE only (f_{foe})	49.5 (13.2)	42.2 (7.12	43.3 (10.8)	44.1 (9.1)

Table 2.2: Performance of Visual Attention Models: Mean (and standard deviation) of percentage of time spent by recorded gaze points inside a circle area having a radius of 4 degrees of the visual field centered on the gaze point computed by the various models.

Bonferroni-adjusted alpha level. Thus, in order to compare each model pairwise (6 comparisons), the alpha level was adjusted to $p = 0.05/6 = 0.0083$.

A one-way within subject design ANOVA on the percentage of gaze falling inside the circle area revealed a significant main effect of the model used for each VE in global and when turning only: Path-Global ($F(3, 27) = 47.5$, $p < 0.001$), Path-When-Turning ($F(3, 27) = 82.8$, $p < 0.001$), Room-Global ($F(3, 27) = 13.0$, $p < 0.001$), Room-When-Turning ($F(3, 27) = 18.8$, $p < 0.001$).

Concerning the Path environment, follow up t test revealed that all models were significantly better than the control *Center* condition when turning (*Velocity+FoE* ($t(9) = -11.84$, $p < 0.001$), *FoE only* ($t(9) = -7.77$, $p < 0.001$) and *Velocity only* ($t(9) = -13.71$, $p < 0.001$)), and in global (*Velocity+FoE* ($t(9) = -8.13$, $p < 0.001$), *FoE only* ($t(9) = -6.39$, $p < 0.001$) and *Velocity only* ($t(9) = -10.28$,$p < 0.001$)). We also found that the *Velocity only* model achieves significantly better gaze prediction than the *FoE only* ($t(9) = 4.68$, $p < 0.002$) and *Velocity+FoE* ($t(9) = 4.44$, $p < 0.002$) models when turning; and than the *FoE only* ($t(9) = 3.47$, $p < 0.008$) and *Velocity+FoE* ($t(9) = 6.34$, $p < 0.001$) models in global.

Concerning the Room environment and only the case when the camera was turning, the *Velocity+FoE* ($t(9) = -4.43$, $p < 0.002$) and *Velocity only* models ($t(9) = 4.25$, $p < 0.003$) performed significantly better than the *Center* condition, but this was not the case for the *FoE only* model ($t(9) = -1.81$, $p = 0.1$). Globally, we found that all models were marginally better than the control *Center* condition (*Velocity+FoE* ($t(9) = -3.13$, $p = 0.012$), *FoE only* ($t(9) = 0.81$, $p = 0.44$) and *Velocity only* ($t(9) = -2.97$, $p = 0.015$)). We also noticed that *FoE only* model gave significantly worse results than the *Velocity+FoE* model in both global ($t(9) = 5.63$, $p < 0.001$) and when turning ($t(9) = 5.35$, $p < 0.001$) conditions.

2.5.2 Discussion

Firstly, we found that the proposed models are significantly better than the control condition for predicting gaze position on the screen when turning. In the Path environment, the *Velocity only* and *Velocity+Foe* models perform up to 3 times better than the control condition (Table 2.2). Indeed, this VE is very similar to the conditions in which we recorded the data used to build the models. Concerning a more different navigation condition, i.e. the Room environment, we find that only the *Velocity+FoE* and *Velocity only* models are significantly better (nearky 2 times) than the control condition, but not the *FoE only* model. We also found that this later model is significantly worst than the *Velocity only* model in the Path environment and than the *Velocity+FoE* model in the Room environment. This could be due to the fact that the *FoE only* model can not be trusted for participants applying high rotation velocities to the camera which makes the FoE vanish from the screen. As a result, this model being included as a component in the *Velocity+FoE* model could have reduced its performance.

Secondly, taking into account data in global, including motion without turns, we found that all models were still better than the control condition for the Path environment. Concerning the Room environment, only the *Velocity+FoE* and *Velocity only* models get marginally better results than the *Control* condition. This marginality could be due to the fact that in the Room environment, participants often followed a straight trajectory or applied lower rotation velocities to the camera (mean of 30.9°/s as compared to a mean of 38.7°/s in the Path environment), thus looking often at the center of the screen (see Chapter 5). In this case, all models predicted the same gaze point position, i.e. the center of the screen. Moreover, the *Velocity only* model is found to be significantly better than the *Velocity+FoE* and *FoE only* models for the Path environment. This result again suggests that taking into account optic flow may reduce performance due to participants applying punctual high rotation velocities to the camera instead of a continuous rotation.

The effectiveness of these methods during first person navigation in virtual environments could make them useful for real time use in VR or game applications. For instance, the raw *Velocity only* model could be used to position the center of the auto-focus zone we propose in Chapter 5 for the purpose of automatic depth-of-field blur. Moreover, the per pixel attentional weight computed by the *Velocity+FoE* model could be used as a new top-down component in real-time visual attention models computing users' gaze positions [73].

2.6 Conclusion

In this chapter, we studied the gaze behavior of users when walking and taking turns in virtual environments using a first-person view [45].

Firstly, we have conducted an experiment to collect data about the gaze behavior of users when actively and passively walking and turning in virtual environment. We have found that the gaze behavior in graphical application is similar to the one observed in real life even without vestibular or proprioceptive cues, i.e. behavior such as the anticipation of turning direction by the gaze. We also have noticed a relation between the rotation velocity applied to the camera and the distance between the gaze point and the center of the screen.

Secondly, we have proposed three simple gaze prediction models, taking as input the rotation velocity of the camera and/or the optic flow on the screen. These models were tested on data collected in various virtual environments. Our results show that these models are significantly more accurate than a control model which always estimates the gaze position at the screen center, up to three times better when turning. Particularly, we would recommend using the model based only on the rotation velocity of the camera which seems to be the best trade off between simplicity and efficiency.

Finally, we suggest that our approach could be used in several interactive applications to accelerate the rendering process or to improve the visual feedback of virtual environments during interactive first-person navigation. The models we have proposed could also be used as new top-down components in any real-time visual attention models estimating users gaze position on screen.

Indeed, the full visual attention model we propose in the next chapter takes advantage of the velocity based model proposed in this chapter. It uses the f_ω function, providing the attentional weights on the screen according to current camera yaw rotation velocity, together with other attentional components to try to predict user's gaze point position in real time.

Chapter 3

A novel visual attention model for first-person navigation in 3D virtual environments

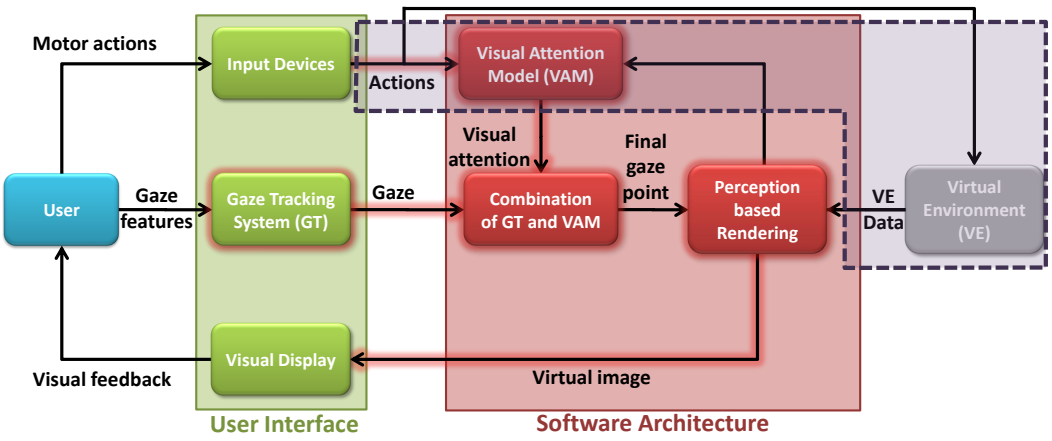


Figure 3.1: A novel visual attention model to compute human attention during real-time first person navigation in 3D virtual environments.

3.1 Introduction

Nowadays, gaze tracking systems are not available on the mass market: only researchers have access to such systems for VR and experimental setups. As for today, the only possible alternative to gaze tracking systems is using a visual attention model. Surprisingly, few research has been dedicated to gaze computation in real-time 3D applications featuring first-person navigation. To the best of our knowledge, only the visual attention model proposed by Lee *et al.* [73] is adapted to this context. However, this model computes a set of possibly gazed objects instead of a single gaze point. Hence, it is not possible to directly use existing gaze-based algorithms.

In this chapter, we introduce a novel visual attention model to compute user's gaze position automatically, i.e. without using an eye-tracking system. Our model is specifically designed for real-time free

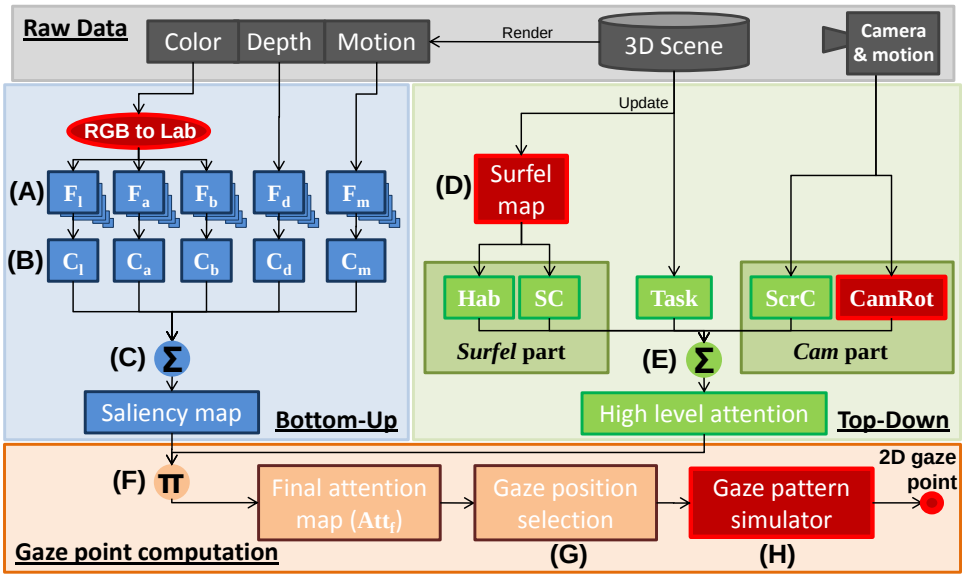


Figure 3.2: Overview of our visual attention model architecture. A) feature maps, B) conspicuity maps, C) bottom-up attention (saliency map), D) update of per-surfel data, E) top-down attention, F) computation of final attention on screen, G) computation of the possible next gaze position and H) the gaze pattern simulator computing the final gaze position on screen. Red color emphasizes the novel parts of our visual attention model compared to existing techniques.

exploration of 3D virtual environments, for applications such as video games, virtual visits of architectural projects, virtual training, etc. In this context, it is the first model which computes, in real-time, a continuous 2D gaze point position instead of only recovering a set of 3D objects potentially observed by the user [73].

We propose to study and compare the performance of our method to a state-of-the-art approach [73]. To this aim, we have conducted an experiment where the ground-truth gaze point position is computed using an accurate gaze-tracking system.

The chapter is organized as follows: Section 3.2 presents the bottom-up and top-down components of our novel visual attention model. It also provides in-depth details about the novel representation of visual objects using surfels and presents implementation details. Section 3.3 discusses about the results of the experiment we have conducted in order to evaluate the efficiency of our model as compared to a state-of-the-art model.

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

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

3.2.1 Computation of the bottom-up component

The bottom-up component of our model computes a pixel-level saliency map using several visual feature: intensity [57], antagonistic colors [57], depth [77] and motion [55].

3.2.1.1 Computation of feature maps

The starting point to compute a saliency map is to compute a feature map for each visual features (Figure 3.2-A):

- **Antagonistic colors and luminosity**: Originally, Itti *et al.* [57] used red/green and blue/yellow antagonistic colors as well as intensities. In their model, antagonistic colors were computed using simple operation on RGB components. In our case, we propose to use the perceptual Lab color space which takes into account human perception [102]. Moreover, this color space has the advantage of directly encoding red/green and blue/yellow antagonistic colors as well as intensity, respectively the a, b and L components. They correspond to F_a, F_b and F_l feature maps in Figure 3.2.

- **Depth**: We propose to use a depth map as proposed in [77][73]. The value $F_d(p)$ for each pixel p of the depth feature map F_d is computed using Equation 3.1 with $z(p)$ being the linear depth of pixel p, z_{near} and z_{far} the distances of the near and far clip planes.

$$F_d(p) = \frac{z(p) - z_{near}}{z_{far} - z_{near}} \tag{3.1}$$

- **Motion**: Our model also takes into account visible motion on the screen. Lee *et al.* [73] proposed to approximate this feature using the motion of a single point of each visual object in world space. However this method does not take into account animated objects, e.g. an avatar moving only the hand. The motion feature $F_m(p)$ of each pixel p of the motion feature map F_m is computed using Equation 3.2 with $v(p)$ being the world space motion projected on the screen and δt the time elapsed since last frame.

$$F_m(p) = \frac{\|v(p)\|}{\delta t} \tag{3.2}$$

3.2.1.2 Computation of conspicuity maps

Before computing the saliency map, the feature maps need to be converted into conspicuity maps using the multi-scale Center-Surround difference operator [57] simulating the response of brain neurons which receive stimuli from the visual receptive fields. Instead of a dyadic Gaussian feature map pyramid, we use an approximation consisting of using the fast hardware mipmap pyramid generation of Graphics Processing Units (GPU) (see [73]). The conspicuity maps, i.e. C_l , C_a, C_b, C_d and C_m in Figure 3.2-B, are computed using Equation 3.3 with i and $i + j$ being mipmap pyramid levels. The level i is a fine level and $i + j$ a coarser level of the pyramid.

$$\forall x \in \{l, a, b, d, m\}, C_x = \frac{1}{6} \sum_{i=0}^{2} \sum_{j=3}^{4} \left| F_x^i - F_x^{i+j} \right| \tag{3.3}$$

Finally, the conspicuity maps are normalized using the $\mathcal{N}$ operator as described by Itti *et al.* [57] where we replace the mean of local maxima by the mean of all values in the conspicuity map for the sake of performance.

3.2.1.3 Computation of the final saliency map

The final saliency map can be generated using the conspicuity maps computed in the previous step (Figure 3.2-C). It is the result of a linear combination of each conspicuity map using Equation 3.4. Finally, the saliency map S is normalized in order to have its values mapped into the range $[0, 1]$.

$$S = \frac{1}{5} \times \sum_{x \in \{l,a,b,d,m\}} C_x \tag{3.4}$$

3.2.2 Computation of the top-down component

The top-down component of our model consists in simulating the cognitive processes that take place in the brain. We first propose a novel representation of visual objects based on surfel (instead of a coarse mesh-based representation) to compute spatial context [73] and habituation [77] components. Our model relies also on screen-space weights to take into account the observed gaze behavior of human navigating in VE using a first-person view.

3.2.2.1 Surfel-based representation of visual objects

Previous models for 3D exploration of VE use a representation of visual objects based on meshes [73] [77]. This can be seen as a limitation as it is not possible to differentiate sub-parts of an object. Using this representation, it becomes indeed impossible to know if the user is gazing at the head or leg of an avatar. Moreover, large walls are problematic. Indeed, the wall can be identified as an attended object but it is not possible to differentiate between cases when the user looks at the middle or at a corner of the wall.

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

In this paper, we propose to solve this issue by using a discretization of polygonal surfaces into smaller elements having the same world space size: surface elements also known as *surfel*. Surfel are well-kown and they are used as a lightmap encoding the irradiance per surface elements [26]. Our visual attention model requires a surfel map to be built for each mesh of every potentially visible object in the VE. In our model, a surfel map virtually subdivides a mesh into several pieces (surfels) and stores them in a texture, thus not involving geometry subdivisions. A surfel is defined by its previous and current positions in world space. Concerning static meshes, the surfel map only contains the current position as visible in Figure 3.3. Also, for the sake of performance, the surfel map of dynamic objects is updated only when they move.

3.2.2.2 Generating the surfel maps

To generate the surfel map, which is in fact a texture, we first need texture coordinates for each mesh triangle that will map each triangle to its corresponding 2D position in the texture. This corresponds to unfolding triangles of the 3D meshes in the 2D map. These texture coordinates must respect two constraints: (1) overlapping triangles are forbidden and (2) a triangle must at least contain the center of a texture element (texel) of the surfel map to result in a surfel data that can be used later. 3D modeling softwares such as *Maya* or *3D Studio Max* already propose such a feature. This process is already used in several applications for the purpose of light-mapping [26].

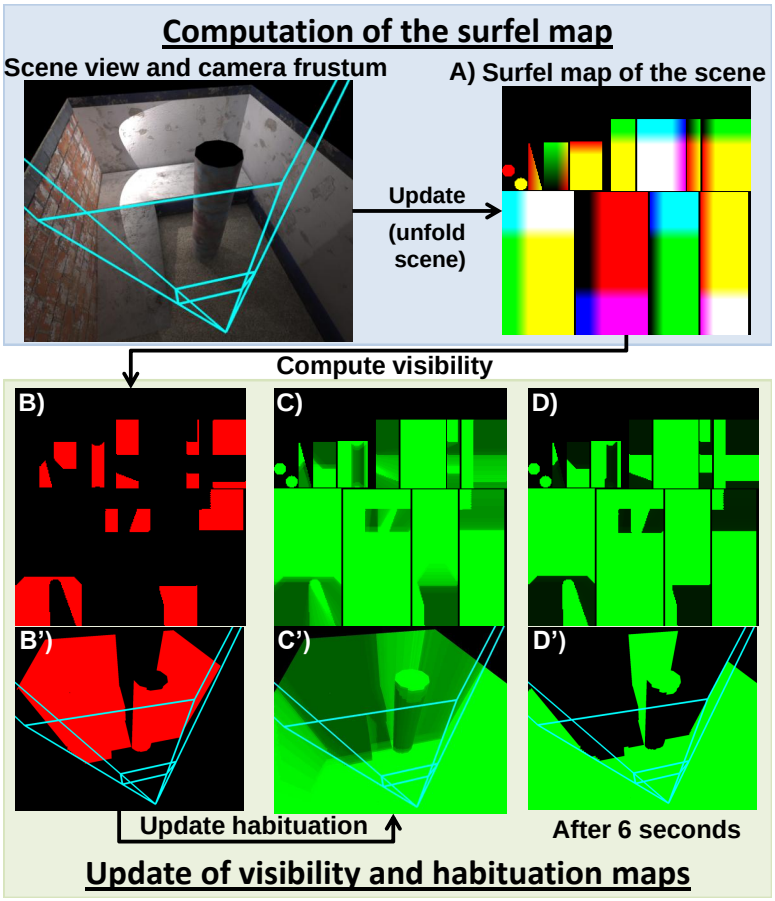


Figure 3.3: Computation of the surfel map, and update of visibility and habituation maps. A) the surfel map containing world space surfel position (XYZ into RGB) B) surfel visibility (red=visible), C) surfel habituation (the greener the surfel, the less habituated the viewer is to it) and D) surfel habituation after waiting 6 seconds. B', C' and D' are views of the surfel map texture mapped on the scene.

To fill the surfel map with data, we simply render the 3D meshes. But instead of applying a 3D projection, we project the meshes' triangles in 2D using the texture coordinates as positions. For each texel, the final 3D coordinates in world space are written in the surfel map according to each model transformation matrix (Figure 3.2-D). Using a surfel map avoids our model being too dependent of the geometry complexity of the scene. It just requires to update the surfel map of dynamic objects when they move. Then, all per-surfel computations are done in the surfel map, i.e. texture space. This way, we can take advantage of the computational power of graphic hardware by processing all surfels in parallel.

After preliminary testing, we have set the world space size of surfel to 20cm. Objects smaller than 20cm are virtually scaled-up. In these cases, we have visually adjusted the scale factor to have each object surfaces represented by at least one surfel. Each VE presented in Section 3.3 fits in a single 256×256 surfel map.

3.2.2.3 Computation of per-surfel components

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

- **Habituation**: The habituation component refers to the fact that visual objects become familiar over time [77]. For each surfel, we compute the habituation value for current frame using Equation 3.5 where δt is the elapsed time in milliseconds.

$$Hab(s) = \begin{cases} Hab_{prev}(s) * \exp(-\frac{\delta t}{h^-}) & \text{if vis(s)==true} \\ Hab_{prev}(s) + h^+ \times \delta t & \text{otherwise} \end{cases} \tag{3.5}$$

When a surfel s is visible, the habituation is attenuated using an exponential decay [77] with interest for s going under 0.1 in 7s ($h^- = 3000$) (see [77] for details). When s is not visible, it is linearly regaining full interest in 20s ($h^+ = 20000$). The visibility $vis(s)$ of a surfel s is determined using shadow mapping (Figure 3.3-A) based on the depth buffer of the rendered view of the scene. Thanks to the surfel-based representation of visual objects, the visual attention model habituates itself to visible parts of a wall but not to the parts that are hidden by other objects (Figure 3.3-B).

- **Spatial context**: Lee *et al.* [73] have proposed a spatial context component to take into account the spatial behavior of the user. This component modulates the importance of each visual object based on its distance to the user. Visual objects (surfels in our case) too close or too far from the distance of interest become progressively less important. We use the same equation as proposed by Lee et al. [73] except that we have removed the condition on the fact that objects must move toward the camera. This was made in order to avoid discontinuities in the spatial context value (Equation 3.6-$SC_d(s)$). Since users tend to get close to objects they want to inspect [73], our spatial component also gives more importance to surfels moving toward the camera (Equation 3.6-$SC_{\Delta d}(s)$). For each surfel s, the final spatial context value $SC(s)$ is computed using Equation 3.6:

$$\begin{aligned} SC(s) &= SC_d(s) * SC_{\Delta d}(s) \\ SC_d(s) &= \frac{d(s)}{C_1} \times exp^{-\left(\frac{d(s)}{C_1}\right)^2} \\ SC_{\Delta d}(s) &= \min(C_2 \times \max(\frac{\delta d(s)}{\delta t}, 0.0), 1.0) \end{aligned} \tag{3.6}$$

Where $C_1 = D/0.707$ and D is the distance when surfel are considered as more important (see [73]). The other parameters are δt the elapsed time since the last frame in seconds, $d(s)$ the distance of the surfel s to the camera and C_2 is a scaling constant. Afer preliminary testing, we chose $C_2 = 0.87$ (it means that an object moving toward the camera with speed equals to the walking speed of user's avatar will have the highest importance value).

3.2.2.4 Computation of statistical screen-space components

In this document, we have studied the gaze behavior of users exploring VE: please refer to chapter 2 and 5. The top-down component of our visual attention model takes into account the observed gaze behavior resulting from the fact that users are navigating in the VE using a first-person view.

- **Global screen-space gaze density**: It has been shown that during a first-person view game, users tend to look more at the center of the screen (Chapter 5). We propose to model this behavior as in [73] using a constant weight $SrcC(p) = \exp^{-Dist2Center(p)}$ applied on the screen where $Dist2Center(p)$ is a function giving the distance of pixel p to the center of the screen (Figure 3.2-E). Screen coordinates are in the range $[0, 1]$ and the middle of the screen is $(0.5, 0.5)$.

- **Gaze behavior during camera rotations**: In our model, we introduce for the first time the real-time attentional component we have proposed in Chapter 2. In this chapter, we have studied user's gaze behavior when turning in VE. They have proposed a new function to compute an attentional weight on the whole screen based on the current rotation velocity of the camera. For instance, for a yaw rotation of the camera to the left, a high attention weight is set to pixels on the left of the screen. Our model introduces this function to compute an attention value $CamRot(p)$ for each pixel p on the screen.

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

3.2.3 Final screen-space attention and gaze position computation

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

3.2.3.1 Final screen-space attention map

Previous methods adapted to our context have proposed to compute the user's level of attention for each visual objects in the scene. To do so, the saliency value is modulated with the top-down attention value [73]. Then, 1 to 3 objects having the highest attentional values are considered as the set of potentially gazed visual objects. However, the use of such a model might be problematic notably when the potentially gazed object is very large on the screen. In our model, we remove this constraint by computing a single continuous gaze position on the screen.

To compute the final screen-space attentional map Att_f, we first compute the top-down attention value $TD(p)$ for each pixel p on the screen using Equation 3.7. This is achieved by rendering visible objects from the camera's point of view. In this equation, $Task(s)$ is a value in the range [0,1] defining surfel relevance for the current task of the user, acting like a semantic weight, s is the position of the surfel in the surfel map, and p is computed using simple texture projection on meshes of $ScrD$ and $CamRot$ textures. In our case, $Task(s)$ is constant for each surfel of a single mesh to reduce memory used but it could also be stored in the surfel map.

$$TD(p) = \frac{Hab(s) + SC(s) + ScrD(p) + CamRot(p) + Task(s)}{5} \tag{3.7}$$

In the last step, the final attention map Att_f is computed from both the top-down and bottom-up components using Equation 3.8 (Figure 3.2-F). Finally, the gaze position is selected as the position of the pixel having the highest attention level (Figure 3.2-G). Then, it is sent to the gaze pattern simulator.

$$Att_f(p) = vis(s) \times S(p) \times TD(p) \tag{3.8}$$

3.2.3.2 Gaze pattern simulator

We have added a gaze pattern simulator in order to process possible gaze position changes and to smooth out the final gaze position (Figure 3.2-H).

In the human visual system, the duration of eye saccades is from 120ms to 300ms long depending on the rotation of the eyes [103], and mean fixation duration varies between 200ms and 600ms. We consider a mean frequency of eyes saccades plus fixation of 600ms. Thus, in order to smooth the final gaze position, we low pass filter the input gaze position using a cut-off frequency of 1.67Hz. This

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

Table 3.1: Computation time in milliseconds for each step of our visual attention model. (*Per-surfel components* refer to Figure 3.2 *surfel part*)

low pass filter simulates the smooth pursuit phenomenon [103], occurring when eyes are following a smoothly moving visual object, while allowing fast gaze jumps simulating saccades.

3.2.4 Implementation details and performance

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

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

During the computation of the *bottom-up* saliency map, the feature and conspicuity maps have all the same resolution of 256×256. Our normalization operator $\mathcal{N}$, simplified as compared to [57], needs parameters such as the maximum and mean values contained in the conspicuity maps. To compute these parameters, we do not iteratively read the entire map using the CPU as this would be too expensive. Instead, we compute the maximum and mean by recursively down-sampling the textures by a factor of two until we reach the size of one texel which contains the final desired values. In this algorithm, at each step, and for each pixel of the coarser level, a fragment program computes the maximum and mean values of the four corresponding pixels sampled from the texture computed in the previous step.

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

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

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

Our visual attention model can run in real time on thanks to the graphic hardware. On a laptop PC (Intel Core 2 2.5Ghz, nVidia GeForce3700M, 4Gb of RAM), the virtual scene VE_1 (see Section 3.3.2)

is rendered at 145 frames-per-second (FPS) with the visual attention model running, as compared to 170FPS without. Detailed GPU computation times are given in Table 3.1. The low computation time of our visual attention model make it usable in several real-time 3D applications and games.

3.3 Experimental evaluation

We have conducted an experiment to evaluate the performance of our visual attention model and compared it to the state-of-the-art model of Lee *et al.* [73]. To the authors' best knowledge, this is the only model adapted to real-time 3D exploration of VE proposed so far.
Twelve naïve participants (10 males, 2 females) with a mean age of 31.8 (SD=6.4) participated in our experiment. They were all familiar with the first-person navigation paradigm and had normal vision.

3.3.1 Experimental apparatus

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

The navigation in the VE was achieved using first-person viewing mode. In this case, the virtual camera is positioned at the level of the eyes of user's avatar. We allowed three degrees of freedom for displacement: walking forward/backward and changing the horizontal and vertical camera orientation, i.e. yaw and pitch angles. Walking forward or backward was achieved using the up and down arrow keys of the keyboard. Changing camera's orientation (yaw and pitch angles) was achieved using movements of the mouse. Mouse correction and filtering were disabled. A horizontal mouse movement of 2.5cm on the table resulted in a rotation of 90 degrees of the camera in the VE (36 degrees/cm). Avatar properties were inspired by real data: height was 1.75m and walking speed was 1.15 m/s [39].

3.3.2 Procedure

For each participant, the experiment was divided in two parts. In each of these parts, participants navigated in 3 different and randomly presented virtual environments: (1) a dynamic and textured VE with moving objects and physics engine (VE_1, Figure 3.4-A), a static and textured VE (VE_2, Figure 3.4-B) and a static and flat colored VE (VE_3, Figure 3.4-C).

During the first part, participants were asked to freely explore the virtual environment without any specific task (T_f, use-case: virtual visits). Then, during the second part, participants were asked to search for keys hidden in the VE (task T_k), and to pick up a maximum of them (use-case: video games). The number of available keys was not given to participants. The second part was meant to study the performance of our model when a task is involved since the presence of a task is known to have an influence on gaze patterns [127]. After preliminary testing, in order to take into account the task involved during the exploration, $Task(s)$ value was set to 1.0 for surfels belonging to keys and 0.5 for all other objects. The same task value was used for the model of Lee *et al.* [73].

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

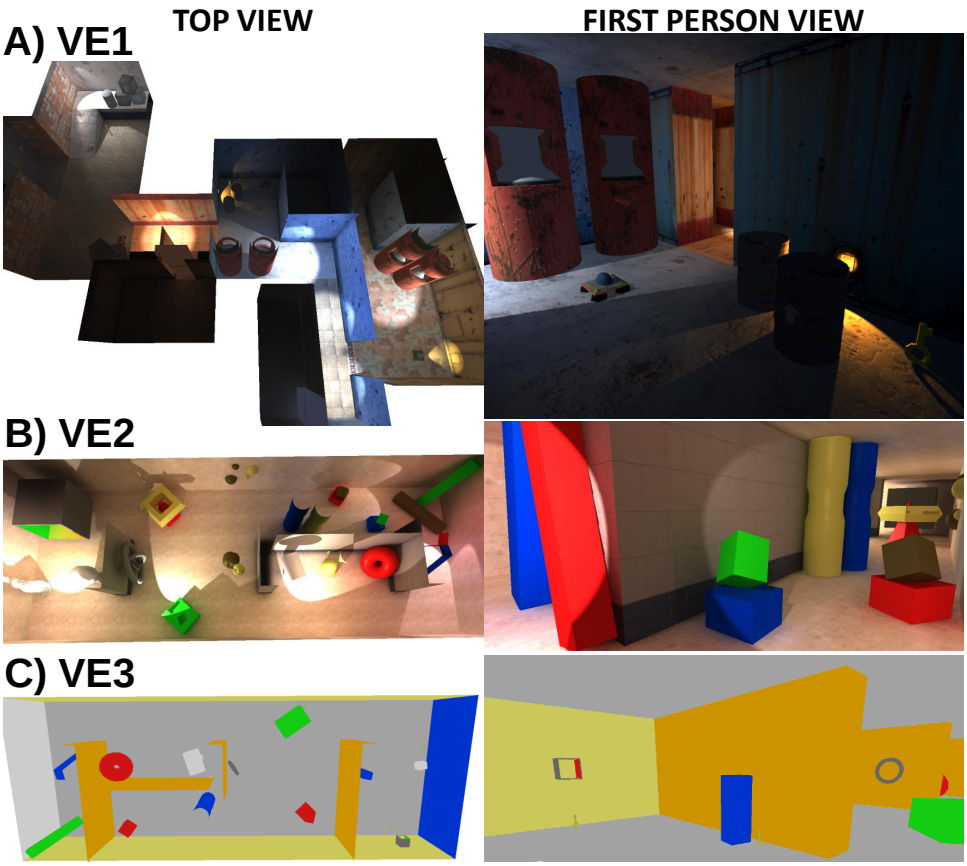


Figure 3.4: The three virtual environments used in our experiment. A) Textured and dynamic VE, B) textured and static VE and C) flat colored and static VE.

3.3.3 Results

To compare our model (*Mour*) with the model of Lee *et al.* [73] (*Mlee*), we computed several performance indicators. Performance indicator P_1 represents the percentage of time spent by the gaze point computed by each model inside a circle area having a radius of r degrees of the visual field and centered on the gaze point G_T computed by the Tobii system (considered here as the ground truth). We have tested several r values for P_1: 2, 4, 6 and 8 degrees (respectively corresponding to a radius of 71, 143, 215 and 287 pixels). Performance indicator P_2 represents the percentage of time spent by the gaze point computed by each model on the same object as the one located at the level of G_T.

Concerning P_1, the model proposed by Lee *et al.* [73] was not designed to output a continuous 2D gaze point. Thus, we propose to compute the final gaze position G_{lee} corresponding to the use of the model of Lee *et al.* [73] as the mean position of pixels belonging to the selected visual object (the one obtaining the highest attention). Our model is not designed to output an attended mesh. Thus, concerning P_2, we have computed the final gazed object of our model as the mesh positioned under the gaze point G_{our} computed by our model.

We have conducted a repeated-measures ANOVA on the dependent variable P_1 with the independent variables being the four radii r and the models *Mour* and *Mlee* (Figure 3.5-A). The ANOVA revealed a significant main effect of the model used ($F(1, 11) = 313.32$, $p < 0.01$). Tukey post-hoc comparisons

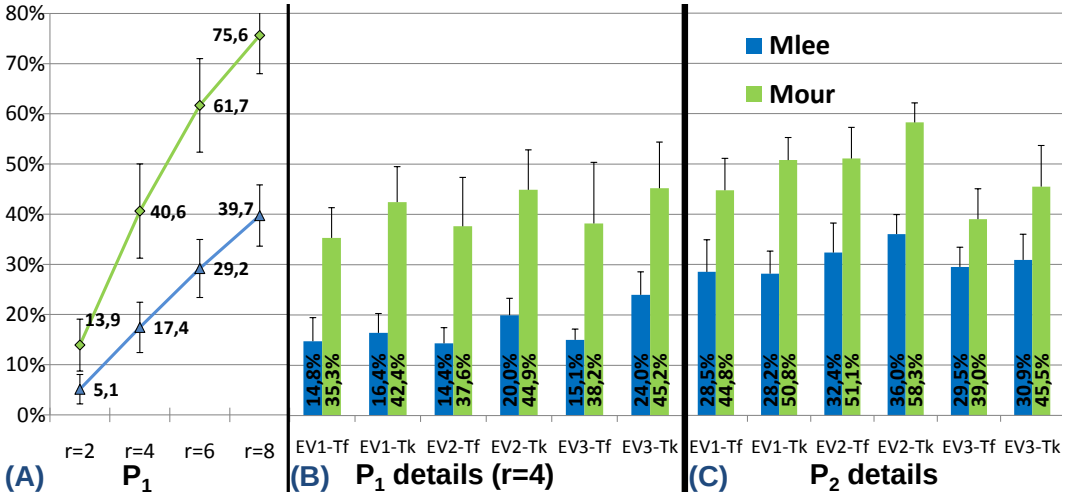


Figure 3.5: Experimental results: performance obtained by various attention models (mean and standard deviation). A, B and C compare our visual attention model (Mour) to the existing model of Lee (Mlee). A and B represent the percentage of time spent by the computed gaze point in a circle having a radius of r degree of the visual field centered on the ground truth gaze position. C represents the percentage of time spent by the computed gaze point on the same object as the one corresponding to the ground truth gaze position. A gives global performance for $r = \{2, 4, 6, 8\}$. B and C give performance details with $r = 4$ for each VE (1,2 and 3) and each Task (T_f and T_k).

showed a significant difference between the two models for each radius value r ($p < 0.01$). The ANOVA also revealed a significant radius × model interaction ($F(3, 33) = 234.68$, $p < 0.01$) meaning that the difference in accuracy between the two models significantly increases when r increases. The detailed comparisons of both models concerning P_1 are presented in Figure 3.5-B for the case where $r = 4$ (as in [104]). In this case, we have conducted a 2 (model) × 3 (VE) × 2 (task) repeated-measures ANOVA. It revealed a significant main effect of the model used on performance ($F(1, 11) = 225.96$, $p < 0.01$). Then, Tukey post-hoc comparisons showed that the performance of *Mour* model was significantly higher than *Mlee* ($p < 0.01$) for all combinations of VE and Task. This is confirmed by the fact that G_{our} was found to be closer to G_T than G_{lee} 71% of the time. Furthermore, Tukey post-hoc comparisons revealed that neither *Mour* nor *Mlee* performance were influenced by the VE. Also, for each VE, They revealed a significant difference of performance for *Mour* and *Mlee* between T_f and T_k (in each case, $p < 0.01$).

Concerning the second performance indicator P_2, a 2 (model) × 3 (VE) × 2 (task) repeated-measures ANOVA again revealed a significant main effect of the model used on performance ($F(1, 11) = 560$, $p < 0.01$). Figure 3.5-C exhibit a mean accuracy of 48.2% (SD = 8.41) for our model *Mour*, and a mean accuracy of 30.9% (SD = 5.6) for the previous model *Mlee*. The ANOVA also revealed a significant Model × Task interaction ($F(1, 11) = 8.32$, $p < 0.05$). Tukey post-hoc comparisons confirmed that *Mour* is significantly influenced by the task ($p < 0.05$) with an accuracy of 51,5% (SD = 7.7) for task T_f and of 44.9% (SD = 7.8) for task T_k. The other model *Mlee* was not found to be significantly influenced by the task ($p = 0.87$), which leads to an accuracy for P_2 of 31,7% (SD = 5.5) for the task T_f and of 30.1% (SD = 5.1) for T_k.

Secondly, we have compared the performance P_1 when using our complete model *Mour* with the use of only its bottom-up component $Mour_{bu}$, or only its top-down component $Mour_{td}$ (Figure 3.6-A). The ANOVA revealed a significant main effect of the model used on performance ($F(2, 22) = 141.75$,

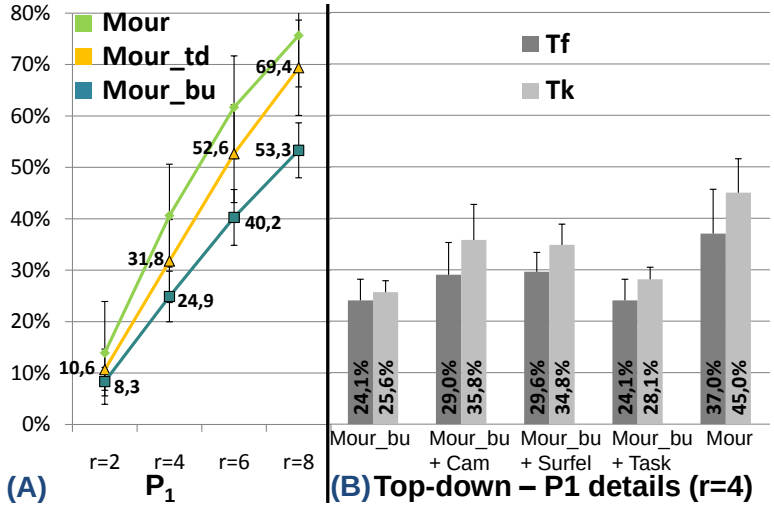


Figure 3.6: Experimental results: performance obtained by various attention models (mean and standard deviation). A compare our complete model to its separated bottom-up (Mour_bu) and top-down (Mour_td) components. B gives details about the contribution of each top-down sub-components when added to the single bottom-up component. A and B represent the percentages of time spent by the computed gaze point in a circle having a radius of r degree of the visual field centered on the ground truth gaze position. A gives global performance for $r = \{2, 4, 6, 8\}$ whereas B gives performance with $r = 4$ for each condition (T_f and T_k).

$p < 0.01$). Then, Tukey post-hoc comparisons showed significant differences between each model for each value of r ($p < 0.01$ in each case).

The top-down part of our model has been designed around three major components: (1) the novel surfel-based representation of visual objects, (2) screen-space weights and (3) task. They correspond to *Surfel part*, *Cam part* and *Task* in Figure 3.2. We have further evaluated the elementary contribution of these three top-down components with that of the bottom-up model only by successively adding their contribution when $r = 4$ (see Figure 3.6-B). A 5 (model) × 2 (task) repeated-measures ANOVA revealed a significant main effect of the model used ($F(4, 44) = 92.73$, $p < 0.01$). Tukey post-hoc comparisons showed that each model was significantly different from the others excepted $Mour_{bu} + Cam$ as compared to $Mour_{bu}$+$Surfel$ ($p = 0.99$), and $Mour_{bu}$ as compared to $Mour_{bu} + Task$ ($p = 0.98$). The ANOVA also revealed a significant model × task interaction ($F(4, 44) = 5.50$, $p < 0.01$). In this case, Tukey post-hoc comparisons showed that each model was significantly influenced by the task excepted $Mour_{bu}$ ($p = 0.79$) and $Mour_{bu} + Task$ ($p = 0.09$).

3.3.4 Discussion

Overall, the results of two performance indicators show that our model performed significantly better than the previous model of Lee *et al.* [73] when exploring various 3D VE.

Firstly, concerning performance indicator P_1, our model performed significantly better than *Mlee* [73] (more than 100% increase in performance), corresponding to a higher percentage of time spent by our computed gaze point close to the ground-truth gaze position (position given by the Tobii gaze-tracker). Interestingly the performance of our model was not significantly influenced by the VE used. This suggests that our model is general enough to support many different kinds of 3D VE. Secondly, we

found that both models performed significantly higher when the searching task was given to participants. This suggests that, when an implicit task is involved during the navigation, users' attention seems more predictable thanks to a higher correlation with the top-down component controling overall gaze direction [55] and including a task related weight. It also confirms Sundstedt *et al.* [116] findings, suggesting that implementing a top-down component is highly beneficial for a visual attention model.

The lower accuracy of *Mlee* concerning P_1 is probably due to the fact that this model was designed to output a 3D object and not a continuous 2D gaze point on the screen. Thus, we have also compared both models using a second performance indicator P_2 which represents the percentage of time spent by the computed gaze point on the same object as the one corresponding to the ground truth gaze position. Surprisingly, even in this case, our model gives significantly better results than the previous model *Mlee*. Besides, performance of *Mlee* was actually lower than the one reported in their paper [73]. This could be due to the fact that in [73] frames showing only the background were all excluded from the analysis, whereas, in our case, all frames were kept except those were G_T was reported as invalid.

Our results also revealed that the performance of our complete model was significantly higher than that when using only its bottom-up component alone or only its top-down component ($Mour$ vs $Mour_{bu}$ or $Mour_{td}$). This suggests that visual attention models based only on a bottom-up or a top-down component would not be as effective at computing human attention as compared to using both components together. In other words, this confirms again the benefit of adding a top-down component to a visual attention model as stated in [116].

Furthermore, we have studied the contribution of separated top-down weights (Figure 3.6-B). Our analysis also revealed that adding screen-space weights, surfel weights or task weights to the single bottom-up component $Mour_{bu}$ resulted in a significant increase in the overall performance (for both T_f and T_k navigations). This suggests that the component Cam, $Surfel$ and $Task$ are reliable top-down weights. Finally, when combining all top-down components together, the performance was also found to be significantly better, suggesting that it is important to mix several top-down components adapted to the context in which the visual attention model is used in order to correctly identify areas of interest to the user.

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

Taken together our results suggest that our novel visual attention model could be used in various real-time 3D applications involving first-person navigation. The computed gaze point could be used to better distribute computational resources to efficiently render a VE [73]. It could also be used to simulate natural effects occurring in human vision to improve graphical rendering and immersion in the VE (Chapter 5).

3.4 Conclusion

In this chapter, we have presented a novel visual attention model to compute user's gaze positions in 3D VE [49]. This model is specifically designed for real-time free exploration of 3D virtual environments and can compute, for the first time, a continuous gaze point position instead of a list of possibly gazed at visual objects. As opposed to previous models that use a mesh-based representation of visual objects, this model features a new data representation based on surfels as a solution to close the gap between screen-space and object-space approaches. The bottom-up and top-down components are combined to create a final attention map and compute the continuous gaze point.

Finally We conducted an experiment to study the performance of our method. Overall, the results show that our model performed significantly better than a state-of-the-art model when exploring various 3D virtual environments with an increase in performance of more than 100%. Taken together our results suggest that our novel visual attention model could be used in various real-time applications such as video games or virtual reality systems. Such a visual attention model could also be used in combination with a gaze tracking system in order to increase its accuracy. This issue is addressed in the next chapter.

Chapter 4

Improving gaze tracking systems using a visual attention model

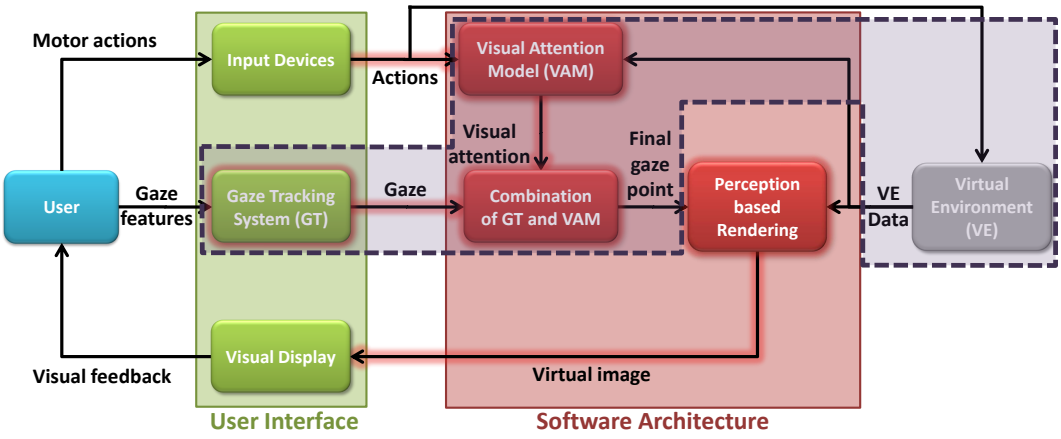


Figure 4.1: Combination of gaze tracking systems with visual attention models.

4.1 Introduction

Many gaze estimation methods have already been proposed in the past. On the one hand, gaze tracking systems are often accurate but, because of their complex calibration procedures, intrusiveness, cost or cumbersomeness cannot be sold on the mass market for daily use. This is mostly due to the fact that in order to increase the accuracy of these systems, costly advanced hardware must be used. However, today, it would be valuable to have a low-cost gaze-tracking system relying, for instance, on a basic webcam [126] or video camera [43] and usable in various conditions without the need for operational expertise. However, contemporary webcam based gaze trackers are not accurate enough [126]. On the other hand, visual attention models can also be used to estimate user's attention when navigating in a virtual environment [73]. However, as opposed to gaze tracking systems, visual attention models can only evaluate user's attention distribution as a general case and have the disadvantage of being deterministic.

This chapter introduce a novel way to improve the accuracy of any gaze tracking systems by using a visual attention model. The method we propose takes advantage of the strength of both of these

approaches. It uses an uncertainty window, based on current system accuracy, to search for a more accurate gaze point position. The final gaze position is estimated inside this uncertainty window using a visual attention model computing image areas which could attract user's attention.

The chapter is organized as follows: Section 4.2 describes the low-cost gaze-tracking system we have developed to compute gaze positions using a webcam and an artificial neural network. This gaze tracker is just an example used to demonstrate the efficiency of our method. Section 4.3 covers the whole architecture of our novel method to combine a gaze tracker with a visual attention model. Finally, Sections 4.4 and 4.5 report on two experiments demonstrating and discussing the promising results of our method in two different contexts: (1) a free exploration of a visually rich 3D virtual environment without a specific task and (2) a video game based on gaze tracking involving a selection task.

4.2 Proposition of an Artificial-Neural-Networks-based Gaze tracker

We first propose a low cost ANN, Artificial-Neural-Networks, based gaze tracking system using a single web cam. This gaze tracker, on its own, is not necessarily new. We use it here as an example of a gaze tracking system that can be improved using a visual attention model.

In this section, we expose the hardware requirements, as well as the software architecture of our ANN-based gaze tracker. We detail the calibration sequence, i.e. how the ANN is trained, and its real-time use. Finally we report on an experiment conducted to measure the accuracy of this system.

4.2.1 ANN-based gaze tracking

Compared to previous gaze trackers based on ANN [7][96], we propose to transform the captured images of the user's eyes into higher primitives. Left and right intersections of the bottom and top eyelid of each eye are manually selected by the user in the video recorded by the web cam using two mouse clicks. Two points per eye are used to extract the image of each eye. During this procedure, the head is maintained in a constant position using a chin-rest. Each time a frame is received from the web cam, the images of the user's eyes are extracted and scaled to images of width W_e and height H_e. Contrarily to Baluja and Pomerleau [7] who send the picture of the eye directly to the ANN, we propose to transform it in order 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 a RGB format to an intensity format, in order to maximize their contrast. In order to reduce the amount of data sent to the ANNs, for each eye image, the pixels of each column and each row are added using Equation 4.1 and 4.2 to respectively obtain two arrays S_x (of size W_e) and S_y (of size H_e). After preliminary testing, we could notice that this computation of gaze positions was more stable than using the raw image.

$$\forall i \in [1, W_e], S_x[i] = \sum_{j=1}^{H_e} eyeImage[i][j] \tag{4.1}$$

$$\forall j \in [1, H_e], S_y[j] = \sum_{i=1}^{W_e} eyeImage[i][j] \tag{4.2}$$

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]. This result is stored in S'_x and S'_y arrays. This mapping is important as it allows us to take advantage of the full working range of each neuron activation function. After preliminary testing, this latter has been set to a linear activation function working in the [0, 1] range.

For each eye, the arrays S'_x and S'_y are sent to the ANNs. Actually, two ANNs per eye are used : one computes the horizontal position of the gaze point based on S'_x and another computes 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 two separate ANNs produces smoother estimations of the gaze position. We also found that $W_e = 40$ pixels and $H_e = 20$ pixels make a suitable size for the scaled image of the eyes given the resolution of the web cam, i.e. 640×480, and learning capabilities of the ANN. Moreover, each ANN is composed of two hidden layers; each one containing twenty neurons. Using this architecture, our gaze-tracker is able to evaluate a continuous gaze position on the screen contrary to previous ANN-based gaze trackers which represent the screen as a 2D grid [7][96].

4.2.2 Calibration sequence and gaze tracking

During the calibration sequence, each ANN is trained in order to compute one coordinate of the gaze point based on its associated eye image. To this aim, the user has to follow a target which moves across the entire screen in order to enable gaze tracking on the full screen surface. Finally, each ANN is trained using the retro-propagation algorithm [122].

At 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.

4.2.3 Environment and hardware setup

The ANN-based gaze tracker we propose only requires one web cam supporting video capture at a resolution of 640×480 pixels. 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 4.2. Furthermore, during preliminary testing, we noticed that our system works better when the height of the eyes is at the level of the center of the screen. For better performance, we recommend positioning the web cam under the screen and not over it. In this case, the eyes are more visible as they are not hidden by dense upper eyelashes. Currently, the system requires the user's head to stay in a constant position and orientation.

4.2.4 Accuracy

We assessed the accuracy of our ANN-based gaze tracking system by conducting an evaluation with 6 participants. During the test, they were positioned in front of a flat 19″ screen with a resolution of 1280×1024 pixels. They were at a distance of 60 cm from the screen, i.e. resulting in a field-of-view of 35 degrees. No sound was played. We used our ANN-based gaze tracking system to compute, in real-time, the participants' gaze position. Their head and eyes were maintained in 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 lasting 3 seconds. We recorded the participants' gaze positions computed by the ANN-based gaze tracker together with the current real positions of the targets.

To assess the accuracy of the ANN-based gaze tracker, we computed the differences between the participants' gaze points measured by the gaze tracker and the real targets' positions on the screen. These differences correspond to the distances between the gaze points and targets on the horizontal and vertical axes in normalized screen coordinates. During this sequence, the first 100 milliseconds after each target changes were not taken into account in order to ignore errors due to saccades. The mean

accuracies are shown in Table 4.1. The accuracy is highly dependent on the shape of the user's eyes. For small and almost closed eyes, the accuracy can decrease to 1.81° whereas for users with wide open eyes, it can increase up to 0.78°.

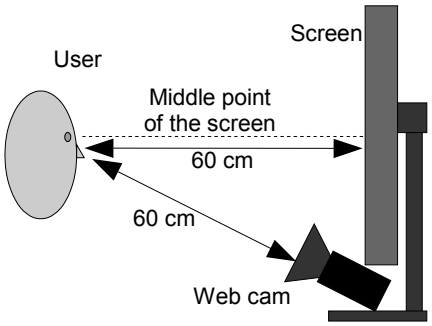


Figure 4.2: Hardware setup.

Since our system does not use infra red light, the web cam needs ambient light to capture clear images of the user's eyes. Moreover, it requires the user's head to be maintained at a constant position, although previous ANN-based gaze trackers support head movements [7] [96] to some extent. However, it is used in this PhD thesis as an example of a gaze tracker that can be improved by using a visual attention model. To make this tracker suitable for a real use, e.g. for gaming, it could be improved by taking into account the eyes position in the video and yaw, pitch and roll angles of the head similarly to [96].

	Horizontal accuracy (degree)	**Vertical accuracy (degree)**
Mean	1.476	1.135
SD	0.392	0.254

Table 4.1: Mean and standard deviation (SD) for the horizontal and vertical accuracy of our ANN-based gaze tracker.

Our ANN-based gaze tracking system has the advantage of computing a continuous gaze point position instead of a position in a two dimensional grid representing the discretized screen [7] [96]. This system is sufficient to achieve various tasks in several environments such as in desktop operating systems or 3D virtual environments. However, it could be improved by taking advantage of the characteristics of the human visual system.

4.3 Using visual attention models to improve gaze tracking

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

4.3.1 General approach

The main idea of our approach consists of looking for salient pixels/objects located near the point given by the gaze tracker and considering that the user is probably looking at these pixels/objects. The global architecture of our algorithm is shown in Figure 4.3. It is composed of two steps: (1) a global step, in which we compute the user's gaze position using, as an example, the ANN-based gaze tracker; and (2) a

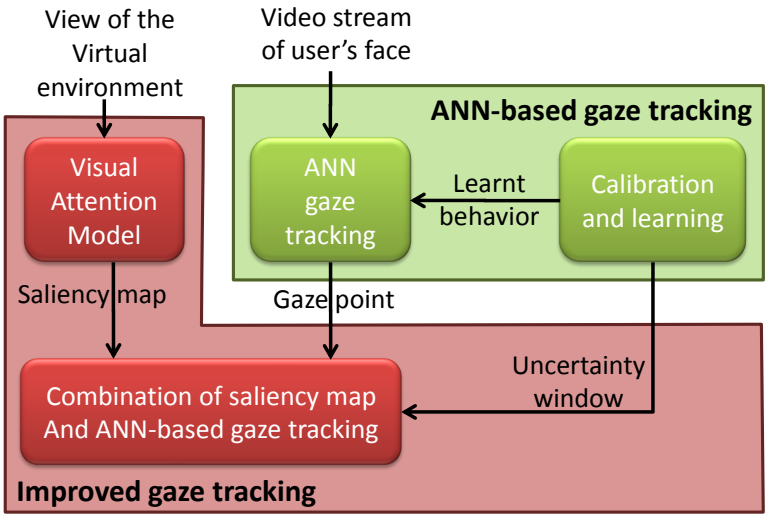


Figure 4.3: Global architecture of our system combining classical ANN-based gaze tracking and visual attention model.

refinement step, in which we compute a saliency map using a bottom-up visual attention model. Therefore, the method we propose to improve gaze tracking systems exploits characteristics of the bottom-up component of the human visual system. We shift the gaze point to the closest most salient pixel, corresponding to the precise/final estimation of the user's gaze point.

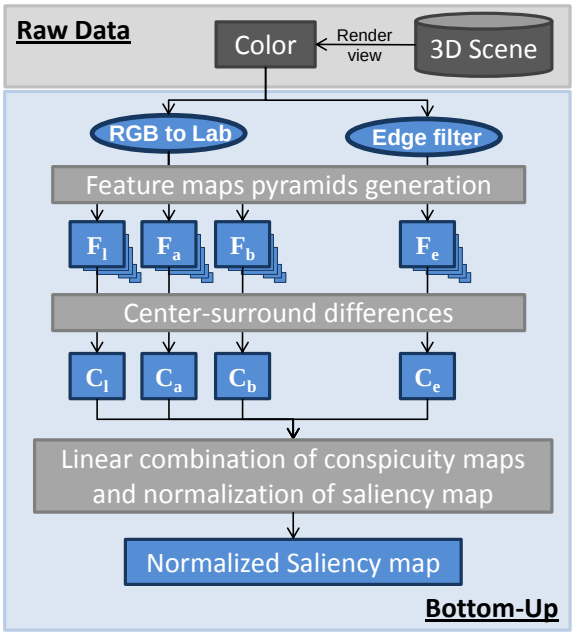


Figure 4.4: Algorithm used to compute the saliency map.

4.3.2 Computation of the saliency map

To compute the saliency map, we use the bottom-up visual attention model presented in Figure 4.4. It is inspired by Itti *et al.* [57] and is a simplified version of the complete model presented in Chapter 3. 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. [57] 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.* [73] improved this computation by using 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 [102]. In this color space, relative differences between colors are "almost perceptually correct" [29]. 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 F_a, F_b and F_l feature maps in Figure 4.4. In Itti *et al.* [57], another feature map encoding the orientations in the image using a Gabor filter was computed. This filter is expensive to compute so we propose to use an edge filter as in Longhurst *et al.* [77]. 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.

Secondly, the feature maps need to be converted into *conspicuity* maps using the multiscale Center-Surround difference operator as in [57]. This operator aims at simulating the response of brain neurons which receive stimuli from the visual receptive fields. Originally, it needs a dyadic Gaussian feature map pyramid [57]. In our case, we use the same approach as Lee *et al.* [73] which consists of using a mipmap pyramid, containing the original feature maps and several down-sampled copies at a lower resolution, computed on the GPU to reduce computation time. The conspicuity maps, i.e. C_l, C_a, C_b and C_e in Figure 4.4, are finally computed using Equation 4.3 with i and $i + j$ being mipmap pyramid levels. The level i is a fine level and $i + j$ a coarser level of the pyramid.

$$\forall x \in \{l, a, b, e\}, C_x = \frac{1}{6} \sum_{i=0}^{2} \sum_{j=3}^{4} \left| F_x^i - F_x^{i+j} \right| \tag{4.3}$$

Finally, the normalized saliency map is computed by a linear combination of the four conspicuity maps using Equation 4.4 where S is the final saliency map, $\mathcal{N}$ a normalization operator and $w_x = M_x - m_x$ with M_x the global maximum and m_x the global mean of the values stored in the conspicuity map C_x. w_x is a factor promoting conspicuity map having small numbers of strong peaks in [57]. However, to simplify the algorithm, we use the global maximum instead of the local maximum [57].

$$S = \mathcal{N}\left(\sum_{x \in \{l,a,b,e\}} w_x \times C_x \right) \tag{4.4}$$

In order to compute the maximum and mean values of C_x, we do not iteratively read the entire conspicuity map using the CPU as this would be too expensive. Instead, we compute the maximum and mean by recursively down-sampling the conspicuity map by a factor of two until we reach the size of one texel which contains the final values. In this algorithm, at each step, and for each pixel of the coarser level, a fragment program computes the maximum and mean values of the conspicuity map's four corresponding pixels computed in the previous step. Once we have obtained these parameters, we can compute w_x for each conspicuity map. In the last step, the final saliency map is normalized using its maximum value (operator $\mathcal{N}$). To find this maximum, we also use this recursive algorithm.

As a result, using our algorithm, the saliency map is computed in real-time using GPU hardware. It takes 0.45 milliseconds for our algorithm to compute a 256×256 normalized saliency map on a nVidia QuadroFx 3700M. To sum up, our algorithm combines techniques of Longhurst *et al.* [77] (orientation approximation by an edge filter) and Lee *et al.* [73] (fast center-surround operator) bottom-up visual attention models with the original model of Itti *et al.* [57]. We have also accelerated the normalization process of the saliency map by using a pyramid algorithm taking advantage of the GPU hardware.

4.3.3 Final computation of the gaze position using a saliency map

Given that the accuracy of the gaze tracking system and the distance between the user and the screen are known, 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 dimension 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 [55] has investigated the contribution of bottom-up saliency to human eye movements. He found that the majority of saccades were directed toward a minority of highly salient areas. Using a normalized saliency map, his experiment showed that 72.3% of the participants' gazes were directed towards an area of the screen containing pixels having a saliency value superior to 0.25. This disk area was centered on the participants' gaze point and has a diameter of 5.6 degrees of their field of view. He suggested that bottom-up saliency may provide a set of gaze 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. Thus, we simply propose to search in this area for highly attractive, salient positions.

Based on Itti's work [55] our algorithm takes into account a saliency threshold S_t. Firstly, it searches inside the uncertainty window for the most salient position sp in the normalized saliency map. Secondly, if the saliency value of sp is greater than the threshold S_t, it sets the final gaze point on sp. On the contrary, if sp is lower than S_t, the gaze point remains unchanged.

Following Itti's work [55], an efficient threshold value for S_t would be 0.25 [55]. However, this value can be adapted according to the application for which the tracker is used. For example, setting S_t to a value of 0 will always set the gaze point position to the most salient pixel inside Wu. In the experiment we present in section 4.4, we expose results for several threshold values and several sizes of uncertainty window.

In our model, we could have included a duration of fixation. However Itti [55] has shown that it is not correlated to saliency values at the level of the gaze point. Moreover, to the best of our knowledge, no other research has found a correlation between saliency maps and gaze duration. Instead, in order to avoid an instantaneous jump between the point estimated by the gaze tracker alone and the gaze tracker improved by the saliency map, the final gaze position estimation is low-pass filtered using a cut-off frequency of 4Hz.

The use of this algorithm is illustrated in Figure 4.5. In this case, the gaze point estimated by the ANN-based gaze tracker is far from the one estimated by the accurate Tobii system. However, when the ANN-based gaze tracker is combined with a saliency map using our method, the final gaze point is inside the "Tobii zone". The Tobii zone takes into account the accuracy of the Tobii system. It 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 gaze tracker (0.5° [118] for the Tobii system) and the distance of the user from the

screen (60 cm). On average, during our two experiments, the sizes of Wu when using our ANN-based gaze tracker were 300×340 pixels for a 1280×1024 screen. For the Tobii gaze tracker, the uncertainty window sizes were 70×70 pixels.

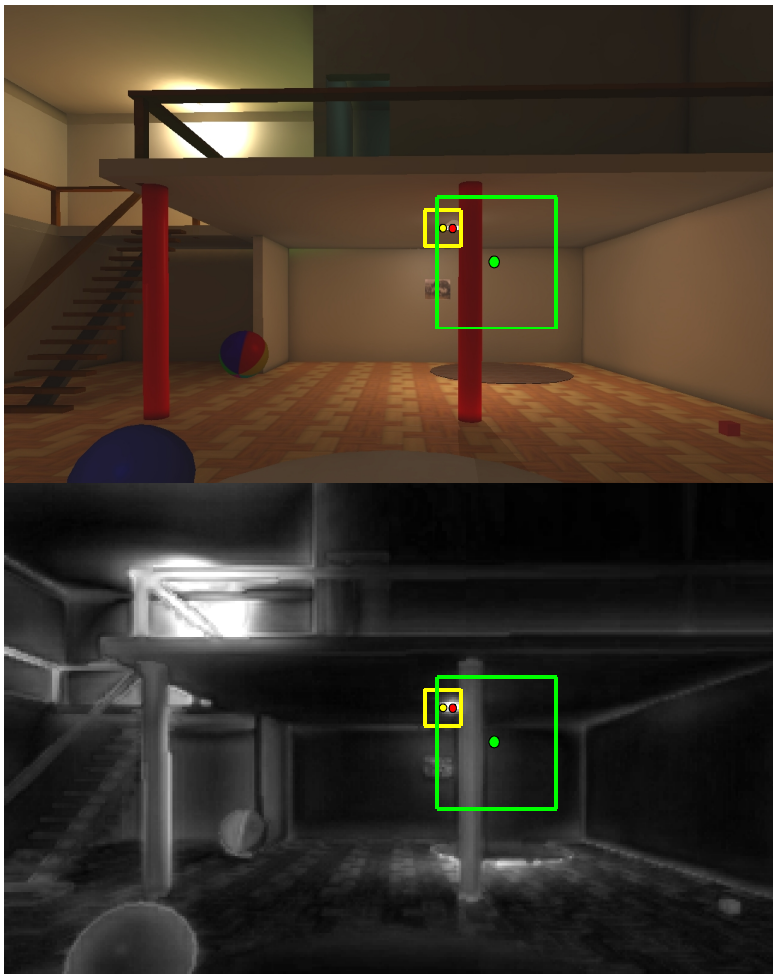


Figure 4.5: Combination of low-cost gaze tracking and saliency map to improve performance. Top: view of the scene, bottom: the corresponding saliency map. In yellow, the gaze point and the uncertainty window of the Tobii system (used as theoretical gaze information). In green, the gaze point and the uncertainty window of the low-cost gaze tracker. In red, the gaze point computed by combining a low-cost ANN-based gaze tracker and saliency map.

4.4 Experiment 1: influence of our algorithm on the accuracy of gaze tracking during free navigation in a virtual environment

Our first experiment aimed at measuring to what extent our algorithm can improve the accuracy of gaze tracking systems during free navigation in a 3D virtual environment without a specific task. We computed the participants' gaze positions using three different systems : (1) ANN-based gaze tracker, (2) ANN-based Gaze Tracker improved by the bottom-up Visual Attention Model (GTVAM) and (3) a Tobii gaze

tracker which is used to compute the "ground truth" gaze position of the user (Tobii).

Ten naïve participants (9 males, 1 female) with a mean age of 25 (SD=2.4) participated in our experiment. They were all familiar with the first-person navigation paradigm and had normal vision.

During this experiment, we used the ANN-based gaze tracker described in Section 4.2 and the Tobii x50 gaze tracker [118]. The ANN-based gaze tracker could be combined with a visual attention model as described in Section 4.3. We tested the performance of our algorithm under several conditions, i.e., with different values for the 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 no sound was played. Their heads and eyes were maintained in 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 4.6.

4.4.1 Procedure

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

The experiment was divided into 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, the participants were free to navigate around the 3D virtual environment. It is important to stress that since we only tested the bottom-up component of the human visual system (visual reflexes only), the navigation duration was short (1 minute) and no particular task was given to the participants.

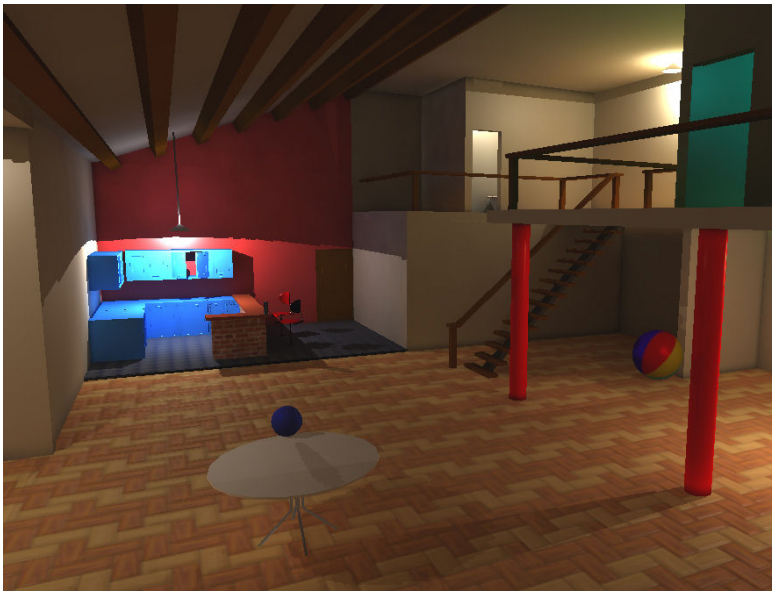


Figure 4.6: 3D virtual environment used for the experiment.

4.4.2 Results

During the experiment, we recorded the participants' gaze positions using the accurate Tobii x50 gaze tracker (computing gaze position at 50Hz) and the ANN-based gaze tracker alone. We also recorded positions and movements in the virtual environment of the virtual camera, as well as position and orientation of dynamic objects. Then, offline, we applied our method designed to improve gaze tracking by replaying the recorded sequences of each participant.

As mentioned before, the two main parameters of our model are the uncertainty window scale factor w_s and the saliency threshold S_t. To assess the influence of these parameters on the accuracy, we tested several values for these parameters: $w_s \in \{1, 1.5, 2, 2.5, 3, 4\}$ and $S_t \in \{0, 0.25, 0.5\}$.

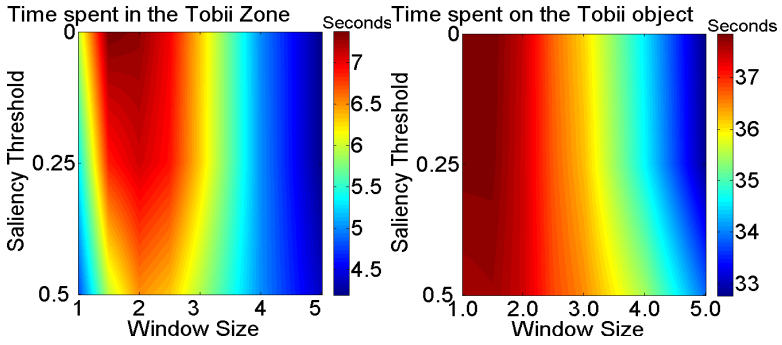


Figure 4.7: Left: time spent looking inside the Tobii zone with our approach (GTVAM) for several values of S_t and w_s. Right: time spent looking at the same virtual object as detected with the Tobii system for several values of S_t and w_s.

We searched for the couple of parameters (w_s*,S_t*) which maximizes the time spent by the computed gaze point inside the Tobii zone. We found that the best values were the same in the two conditions, i.e. the window size w_s* equals to 1.5 and the saliency threshold S_t* equals to 0.0 (see Figure 4.7).

	Saliency map alone	**ANN alone**	**GTVAM (S_t = 0.0, W_s = 1.5)**
time inside Tobii zone	4.4% (2.64s)	5.3% (3.19s)	12.3% (7.37s)
time on same object as Tobii	23.5% (14.07s)	37.9% (22.75s)	63.1% (37.85s)

Table 4.2: Mean performance of our approach (GTVAM) as compared to the ANN-based gaze tracker alone and saliency map alone (i.e., using the whole image on screen).

Then, we compared the performance of our method with performance obtained in two other conditions: ANN alone and saliency map alone. We tested whether the time spent inside the Tobii zone (ZONE) is significantly different for the GTVAM condition as compared with ANN alone and saliency map alone conditions. The same procedure was performed based on the time spent on the same object as the one detected by the Tobii (OBJECT). To compute the object visible at a position on the screen, we used an item buffer containing the unique id of the visible object at each pixel. The results

are summarized in Table 4.2. To test whether the differences observed are significant or not, we used paired Wilcoxon tests. We firstly compared ANN alone and GTVAM gaze tracking conditions using the ZONE and OBJECT measures. We found that the distributions are significantly different for the ZONE ($p < 0.01$) and OBJECT ($p < 0.01$) measures. Then, we compared Saliency map alone and GTVAM conditions. We found that the distributions are significantly different for the ZONE ($p < 0.01$) and OBJECT ($p < 0.01$) measures. These results show that our method is able to increase the accuracy of gaze tracking systems.

4.4.3 Discussion

First, this experiment could be considered as an introduction to a methodology to compute optimal values for the parameters of our algorithm, i.e. S_t and w_s. The values we found can be considered as good indicators for implementing our approach in similar configurations. Of course, for determining optimal values adapted to another configuration (screen size, application, etc), another procedure could be achieved using the same methodology.

Second, 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: time spent by the final gaze point inside the Tobii zone (ZONE) and time spent on the same object as Tobii (OBJECT).

We found that the ZONE and OBJECT measures were significantly improved when using the GTVAM condition as compared to the ANN alone condition. Besides, the accuracy of the Tobii system, which is 0.5 degree of angle, might have reduced the efficiency of our method. The gaze point computed by this system, considered as the ground truth, might sometimes not be located on the object actually gazed by the user. This has been especially observed when the user is gazing at the border of the screen.

Moreover, the uncertainty window may sometimes contain several salient areas which are competing for the final gaze position and the user may not look at the most salient pixel in the normalized saliency map. We can illustrate this with the condition where the higher value in the normalized saliency map of the whole screen is considered as the gaze point, i.e. Saliency map alone condition in Table 4.2. In this case, the selected gaze position is inside the Tobii zone only 4.4% of the global time. This correlates with Itti [55] results which show that the user does not constantly look at the highest salient pixel. In a VE, a task may always be implied and this would reduce the attractivity of some salient areas [116], something that the model we used does not take into account for the moment. Furthermore, with our technique, it could be impossible to fixate on some non-salient areas. In this experiement, we have only tested the use of a bottom-up visual attention model. However, we belive that the use of a top-down visual attention components could remove these two inherent problems. Components such as habituation [77], spatio-temporal context [73] or task relevance of objects [22] could be added to our algorithm.

Taken together, our results suggest that the straightforward method we propose can significantly increase gaze tracker performance, especially in the case of object-based interaction.

4.5 Experiment 2: influence of our algorithm on the accuracy of target selection task during a video game

Our second experiment aimed at measuring to what extent our algorithm can improve gaze tracking systems. To this aim, we have developed a simple video game, involving a selection task, in which users play using only their eyes. We again compared three different gaze tracking approaches: (1) ANN-based gaze tracker, (2) ANN-based gaze tracker improved by the bottom-up visual attention model (GTVAM)

and (3) Tobii gaze tracker.

Ten naïve participants (9 males, 1 female) with a mean age of 26.8 (SD=3.2) participated in our experiment. These people did not participate in the previous experiment and had normal vision.

4.5.1 Procedure

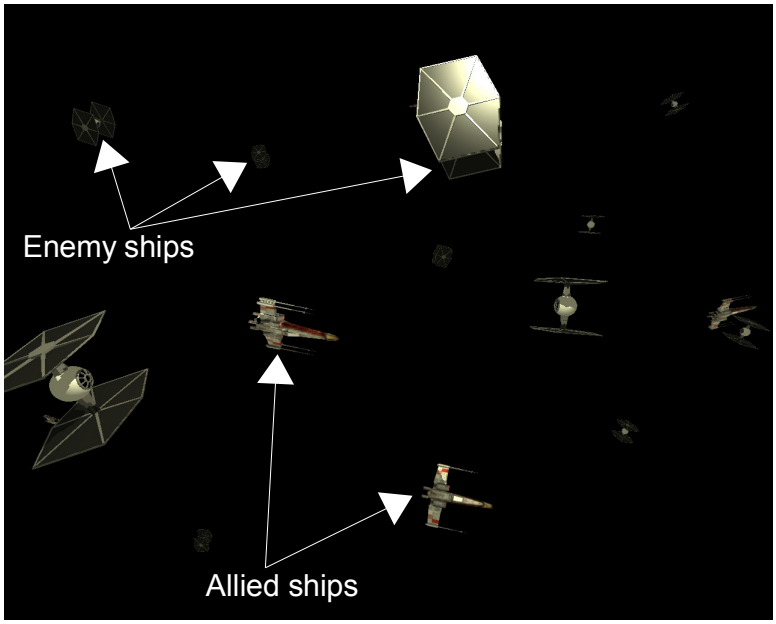


Figure 4.8: Screenshot of the game used where players have to look at space ships to destroy them.

	Enemy ships destroyed	**Allied ships destroyed**	**Game time (s)**	**Destroyed at D1**	**Destroyed at D2**	**Destroyed at D3**	**Time on no object (s)**
ANN	4.55 (2.5)	0.1 (0.3)	90.0 (0.0)	3.8 (1.8)	0.75 (1.3)	0.0 (0.0)	83.0 (2.3)
Tobii	11.6 (2.7)	0.05 (0.2)	85.3 (10.5)	5.0 (0.0)	4.4 (1.1)	2.2 (1.9)	74.1 (11.2)
GTVAM	15.0 (0.0)	0.75 (0.8)	33.2 (18.1)	5.0 (0.0)	5.0 (0.0)	5.0 (0.0)	21.2 (18.5)

Table 4.3: Mean and standard deviation for each measure of the game experiment for each gaze tracking condition.

The same apparatus as described in Section 4.4 was used. For each participant, the task consisted in destroying ships flying through space as shown in Figure 4.8. They just had to look at one ship to destroy it automatically after a dwell-time of 600ms. Participants were asked to destroy only enemy ships and not allied ships. Participants had 90 seconds to destroy as many enemy ships as possible. If all enemy ships were destroyed before the end, the game was automatically stopped.

Participants stared in front of the screen, interacting only with their eyes. The camera in the virtual environment was still. There were 15 enemy ships flying and following randomized straight trajectories: 5 ships flying at a distance of $D1$ =45m from the camera, 5 ships flying at a distance of $D2$ =80m and 5 ships flying at a distance of $D3$ =115m. The ships crossed the screen randomly from left to right, or from

right to left. There were also 5 allied ships flying and following random trajectories at a distance between 45m to 115m. Once a ship left the screen, its new trajectory was computed and its *dwell-time* value was restored to 0ms. In each condition, the final gaze point computed by the system is used to detect if the user is looking at a ship or not. To do so, we use a classical ray-triangle intersection algorithm at the level of the gazed pixel.

During the experiment, the participants played the game six times, two times under each gaze tracking condition. The order of presentation of each gaze tracking condition was counterbalanced. At the end of each game, we recorded the game duration, number of enemy ships destroyed for each distance $D1$ to $D3$, number of allied ships destroyed and time spent gazing at no objects.

4.5.2 Results

Using the Wilcoxon paired test, we found that our technique (GTVAM) is significantly more efficient to destroy enemy ships than the ANN gaze tracker alone ($p < 0.01$). Surprisingly, our algorithm was found even more efficient than the Tobii ($p < 0.01$) gaze tracker. The mean and standard deviations for each measure are summarized in Table 4.3.

We could decompose the performance for each distance $D1$ (near), $D2$ and $D3$ (far). We found that GTVAM gave better performances than ANN alone for each distance $D1$, $D2$ and $D3$ (Wilcoxon paired test $p < 0.01$; $p < 0.01$; $p < 0.01$). This corresponds to 3 different sizes of ship on screen: 200, 100 and 40 pixels. We also found that the difference was not significant between GTVAM compared with Tobii for distance $D1$ and $D2$, but significantly different for distance $D3$ ($p < 0.01$). This result shows that the better performance of GTVAM compared with Tobii is due to the destruction of the farthest targets (smaller ships on screen).

Another way to evaluate the efficiency of our algorithm is to measure the time spent to complete the mission (destruction of the 15 enemy ships). The GTVAM gaze tracking condition is more efficient when compared with ANN alone (Wilcoxon paired test $p < 0.01$) and Tobii (Wilcoxon paired test $p < 0.01$) conditions. We also computed the time spent on no object and we found that GTVAM is more efficient than the ANN alone (Wilcoxon paired test $p < 0.01$) and Tobii (Wilcoxon paired test $p < 0.01$) conditions.

Our algorithm (GTVAM) is significantly less efficient concerning the destruction of allied ships as compared with ANN alone (Wilcoxon paired test $p < 0.01$) and Tobii (Wilcoxon paired test $p < 0.01$). However, the number of erroneous destructions in the case of GTVAM remains very low (on average less than 1 error per participant).

4.5.3 Discussion

Overall, using the ANN-based gaze tracking condition combined with our method (GTVAM), participants performed better in the game as compared to the ANN alone and Tobii gaze tracking conditions (Table 4.3). The GTVAM condition allowed the participants to finish the game faster than under the two other conditions. This is due to the fact that the game ends when all enemy ships are destroyed and this happened only when the GTVAM gaze tracker was used. It can be explained by the lower time spent on no objects in this condition as compared with the two other gaze tracking conditions. However, the number of allied ships destroyed erroneously is significantly higher compared to ANN alone and Tobii conditions. This is a well-know problem in gaze-tracking called the *Midas Touch Problem* [60], i.e. people just want to look at an item but it results in an unwanted action. This emphasizes the fact that the method we propose might result in erroneous, i.e. non intentional, selection. A simple way to reduce

these errors would be to ask the player to push a button to fire at enemies.

As shown by Sundstedt *et al.* [116], a saliency map alone is not sufficient to predict the users attention, especially when a task is explicitly given to the user. In our case, the black background probably helped the GTVAM model. If we had used a more complicated background, e.g. stars and planets, the GTVAM could have performed lower. To overcome this problem, it would be important to take into account the destruction task in the visual attention model such as in [22][77][115][73]. In this case, we would give a high weight to the enemy ships, a lower weight for the allied ships and a slight weight for the decorative background.

Our analysis showed no significant difference in the number of enemy ships destroyed for near distances (D1 and D2) between the Tobii and GTVAM conditions. However, the GTVAM gaze tracker has a significantly higher number of far enemy ships (D3) destroyed. Firstly, it shows that our algorithm can compensate for the low accuracy of a gaze tracker in such a case. Secondly, it suggests that it can compensate for the latency in the estimated gaze point position. This latency is due to the acquisition/processing of video and to the computations needed to evaluate the final gaze position on screen. Indeed, thanks to the uncertainty window, the method we propose is able to set the gaze point on salient objects away from the current gaze point which is subject to latency.

To sum up, despite a small amount of erroneous selections, the GTVAM gaze tracking condition allowed participants to globally perform better in the game. The time to finish the game was the shortest and all enemy ships were destroyed. This experiment emphasizes the fact that the use of a saliency map can increase the tracker accuracy and can also compensate the latency of the tracking systems.

4.6 Conclusion

We have introduced, for the first time, the use of visual attention models to improve the accuracy of gaze tracking systems in interactive 3D applications [44].

We have proposed a method based on a single saliency map meant to improve the accuracy of any gaze tracking system, such as the ANN-based gaze tracking system. It uses an uncertainty window, defined by the gaze tracker accuracy and located around the gaze point, in order to search for the point the user is more likely to look at using a visual attention model. We have presented the results of two experiments conducted to assess the performance of our approach. The results showed that the method we propose can significantly improve the accuracy of a gaze tracking system. For instance, during free first-person navigation, the time spent by the computed gaze point on the same object as the ground truth gaze point is increased by 66%. Thus, our approach seems especially interesting in the case of object-based interaction. It even performs better than the accurate Tobii gaze tracker in a game requiring the selection of small visual objects to destroy them. On average, using our approach, the player could destroy two times more small objects on screen than with the standard Tobii system.

Taken together, our results show a positive influence of our algorithm, i.e. of using visual attention models, on gaze-tracking. Our method could be used in many applications such as for video games or virtual reality applications. An important feature of the proposed approach is that it can be adapted to any gaze tracking system, while the visual attention model can also be extended and adapted to the application in which it is used.

Part II

Improving Visual Feedback in 3D Virtual Environments Based on the User's Visual Attention

Chapter 5

Improving visual feedback in VR using visual perception: simulation of depth-of-field blur effect

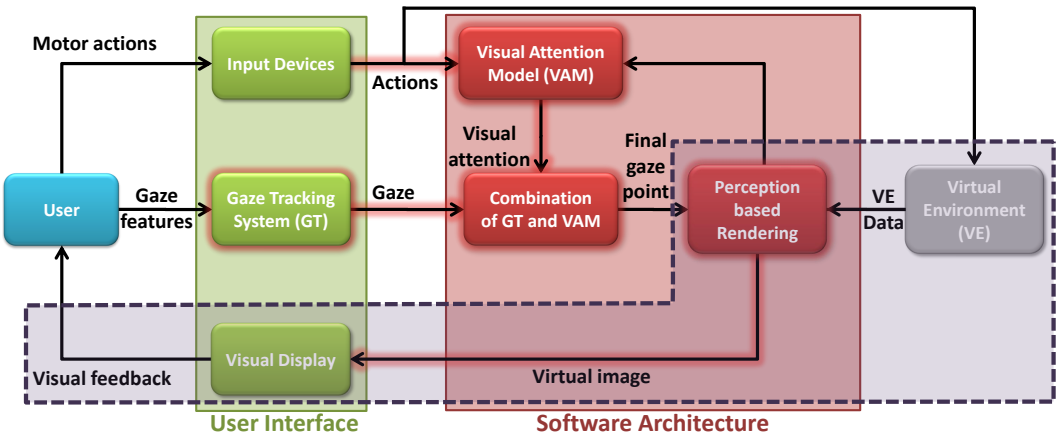


Figure 5.1: Automatic depth-of-field blur effect adapted to first person navigation.

5.1 Introduction

Today, virtual scenes can be rendered in real-time with visual quality close to real pictures. However, these rendered images can sometimes looks too synthetic, or too perfect, due to the lack of natural effects occurring in human vision. Perception-based rendering methods have been proposed, but they often only involve simplifying the VE [99] to accelerate the rendering process, or emphasizing some sub-parts of the visualized data [24]. However, some visual effects have been proposed to simulate phenomena occuring in human vision such as motion blur or Depth-of-Field (DoF) blur. Several video games [112] and movies [64] implement these visual blur effects to increase the naturalness, and thus the visual feedback quality, of a rendered scene to the viewer. Depth-of-Field is a phenomenon resulting in visual objects appearing sharp when inside a range of distances near to the 3D gaze point. Then, because

the crystalline in human eyes acts as a lens, objects behind and in front of the 3D gaze point appear progressively blurred. DoF and its associated blur effects are well-known classical depth cues in human vision [5] and, thus, were added in early Computer Graphics pipelines.

Visual blur effects have not yet been introduced in real-time VR applications. However, the programming capabilities and the processing power of the current graphic cards make it possible today to compute visual blur effects in real-time. Recent games make use of such effects, but we do not know today their influence on both the performance and the subjective appreciation of users. Therefore, there is currently a need for two kinds of investigation on the use of visual blur in Virtual Environments that we will successively address in this chapter:

1. Development of a new model of realistic visual blur effects for virtual environments (VE) that takes interactivity and real-time constraints into account

2. Evaluation of the use of visual blur effects in terms of both performance and subjective preference in VE.

The chapter is organized as follows: Section 5.2 briefly describes related work in the field of depth-of-field blur effects used in computer graphics. Section 5.3 introduces our proposition for dynamic DoF blur adapted to first-person navigation in VE. This blur model is based on an auto-focus with an automatic computation of the focal distance and a temporal accommodation of the blur effect taking into account object relevance to the current task. Section 5.4 details the GPU implementation of this algorithm. Section 5.4 reports on an experiment we have conducted to measure and analyze the gaze point of video gamers during first-person navigation, by means of an eye-tracking system, in order to correctly set the parameters of the auto-focus system. Finally, section 5.5 discusses and reports on the experiment conducted to study the influence of such an effect on the performance of gamers during multiplayer sessions of a first-person-shooter game.

5.2 Previous depth-of-field blur methods

The use of visual blur effects in virtual reality has been suggested by Rokita [105], who stated that such blur effect was “especially important in VR applications”. The simulation of visual blur was introduced early in Computer Graphics to improve the photorealistic aspect of synthetic images. Potmesil and Chakravarty [97] were the first to propose the simulation of an optical lens to simulate depth-of-field blur. Their algorithm uses the original sharp image and the depth of each pixel. Then it applies a post-processing step to the sharp image to add blur. The lens simulation provides the amount of blur for each pixel according to its depth. Using lens simulation, a point which is out of focus becomes a disk or circle after the projection through the lens. The resulting circle is called the Circle of Confusion (CoC). The diameter of this circle of confusion corresponds to the amount of blur [97]. After this pioneer study, most of the researchers used the lens model to compute the amount of Depth-Of-Field blur [31]. However, Barsky [8] introduced the alternative concept of vision-realistic rendering that uses all the characteristics of an individual’s optical system. Barsky could then accurately simulate the foveal image scanned from wavefront data of human subjects.

The main problem of DoF blur algorithms is an artefact called “color leaking” across depth discontinuities. This artefact blurs edges of in-focus objects which are located in front of a blurred background. The existing DoF algorithms vary in the way they compute the blur itself and in the way they avoid color leaking. A survey of DoF algorithms has been provided by Demers [31]. Actually, the different

techniques can be divided into three main categories [31]: the scattering techniques, the gathering techniques and the diffusion techniques. For instance, the gathering techniques (also called reverse-mapping techniques) only use the pixels of the sharp image. For each pixel of the final image, the algorithm gathers and blends colors from the pixels of the source image which belong to the current pixel's circle of confusion. Simple depth tests during the gathering step avoid color leaking artefacts. This approach can be easily implemented on current graphic hardware [101].

Visual blur other than DoF blur can be used to enhance the visual appearance of digital images. For instance, the peripheral blur refers to the coarser acuity of the eyes from the fovea to the periphery [110]. The motion blur [80] simulates the images obtained when using a digital camera in reality. They correspond to the recording of objects that move rapidly. Indeed, the integration of such images during the time the shutter is opened generates a blur.

Brooker et al. [17] investigated the use of DoF blur effect using a stereoscopic display and an eye-tracking system in the case of path-finding in a 3D labyrinth. However, their results did not show evidence of performance improvement. They concluded that it could be due to the very slow frame rate of their application and they suggested to implement and further evaluate real-time DoF blur effect in VE. In videogames, the latest generation of games, *Unreal engine 3* (developer: Epic Games, 2007) and *Crysis engine* (developer: Crytek, 2007) propose temporary DoF blur together with motion blur. These DoF blur effects suffer from leaking artefact. The game *Call of Juarez* (developer: Techland, 2006) also introduces a dynamic DoF blur effect. However, it remains limited to a "sniper mode" with only a few depth plans.

5.3 Visual blur effects for first-person navigation in virtual environments

In this section we describe a model of dynamic visual blur for first-person navigation in VE. It is based on the combination of two blur effects: (1) a depth-of-field blur and (2) a peripheral blur.

5.3.1 Depth-of-Field blur effect

The Depth-of-Field blur effect simulates the visual blurring of the pixels of objects located in front or behind the gaze point. The gaze point is associated with a focal distance (f_d), i.e., the distance between the eyes (or camera) and the gaze point.

5.3.1.1 Use of a lens model

We use here the classical model of the lens introduced by Potmesil and Chakravarty [97]. In this model, the per-pixel amount of blur, i.e. the diameter of the circle of confusion $DCoCd$ of a point projected on the screen, is given by Equation 5.1 in which D is the lens diameter, f the focal length of the lens, f_d the current focal distance and z the depth of the point.

$$DCoC_d = \left| \frac{D \times f \times (f_d - z)}{f_d \times (z - f)} \right| \tag{5.1}$$

The unknown variable of this equation is the focal distance f_d, the distance at which the objects appear sharps. To compute f_d, we propose the auto-focus zone described in the next Section.

5.3.1.2 Use of an auto-focus zone

An optimal way to determine the focal distance in real time would be to use an eye-tracking system (as proposed in Chapter 6). However, such devices are still expensive, complex and not available on a mass market. Thus, in the absence of an eye-tracking system, we propose to use a paradigm used in First-Person-Shooter games (FPS). In FPS games, the user uses his/her 2D mouse and keyboard to manipulate a virtual visor always located at the center of the screen. In this case, we can assume that the user mainly looks at the part of the screen close to the visor. Using an eye-tracking system, Kenny et al. [67] found that gamers of FPS videogames indeed watched more than 82% of the time a central area corresponding to half the size of the monitor. Thus, we introduce the notion of "auto-focus zone", i.e., an area located at the center of the screen that the user is supposed to preferentially look at. This recalls the auto-focus systems used in digital camera which also aim at determining an appropriate focal distance when taking a picture.

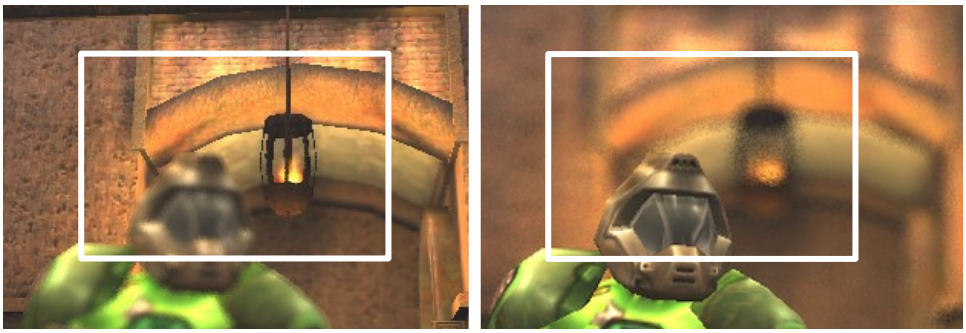


Figure 5.2: Depth-of-Field Blur when using a rectangular auto-focus zone (white rectangle), without (left) and with (right) semantic weighting.

The depth of each pixel of the auto-focus zone is first computed using an auxiliary buffer. Such as in digital camera, the function used to compute the focal distance from all the depths of the pixels enclosed in the auto-focus zone could be Minimum, Maximum or Average. In the case of FPS games, some objects of the environment are known to be more important than others (e.g., enemies or bonus-objects). To this aim, we propose to use a semantic weighting of the pixels depth. The semantic weighting is done to increase the weight of pixels corresponding to objects (or targets) that are known to be important in the scene. To do so, we can add to the initial description of the virtual objects a field corresponding to its visual semantic weight. The semantic weight of each pixel ranges from $WSmin$ to $WSmax$. Figure 5.2 illustrates the use of semantic weighting. In this example, the weight of the character/enemy in front is much higher compared to the one of the background: even if the character covers fewer pixels than the background (less than one quarter of the area), the focus is done systematically on it. In our case, the semantic weights remain constant. However, in other applications, we could use a dynamic weighting of the pixels. For instance, we could simulate the habituation phenomenon by progressively decreasing the semantic value of each object based on its exposition duration on the screen.
In addition, we introduce a spatial weighting that slightly modifies the weight of the central pixels. It is achieved by using a Gaussian function that gives a weight of $WGmax$ to the center and $WGmin$ to the borders of the zone. Finally, the resulting focal distance f_d is computed using Equation 5.2, with $WS(p)$ the semantic weight of pixel p, $WG(x)$ the Gaussian spatial weight for distance x, $d2AC(p)$ the distance of p from the center of the auto-focus zone. In our final implementation, $WGmin$ was set to 0.7, $WGmax$ to 1, $WSmin$ to 0.004 and $WSmax$ to 1.

$$
\begin{aligned}
W(p) &= WS(p) \times WG(d2AC(p)) \\
WD(p) &= W(p) \times depth(p) \\
f_d &= \frac{\sum_{p \in zone} WD(p)}{\sum_{p \in zone} W(p)}
\end{aligned}
\tag{5.2}
$$

5.3.1.3 Computation of focal distance on GPU

The computation of Equation 5.2 on CPU takes a lot of time, especially for a large auto-focus zone. Therefore, we propose to compute the focal distance using a General-Purpose computation technique on GPU (GPGPU).

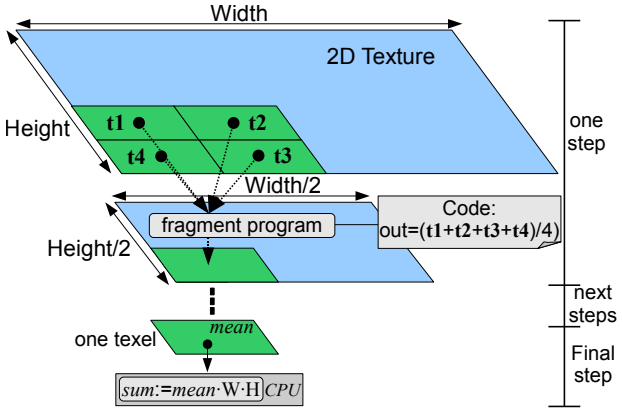


Figure 5.3: Hardware acceleration for the computation of the focal distance.

To evaluate f_d using Equation 5.2, we need to compute $\sum_{p \in zone} WD(p)$ and $\sum_{p \in zone} W(p)$. The computation of these sums can produce overflow. Thus, we replace the sum operation by the mean operation using Equation 5.3.

$$
\begin{aligned}
\sum_{p \in zone} WD(p) &= \overline{WD(p)} \times card(zone) \\
\sum_{p \in zone} W(p) &= \overline{W(p)} \times card(zone)
\end{aligned}
\tag{5.3}
$$

To do so, we first store the values of $WD(p)$ and $W(p)$ in a 2D texture with the size of the auto-focus zone. Then, we compute the mean of this texture by recursively downsampling it by a factor of 2 until we reach the size of one texel which finally contains the means $\overline{WD(p)}$ and $\overline{W(p)}$ (Figure 5.3). In this algorithm, at each step and for each pixel, a fragment-program computes the mean of the 4 corresponding texels of the texture computed at the preceding step. Thus, using this technique, we could finally accelerate the computation of the focal distance f_d using the GPU instead of the CPU.

5.3.1.4 Simulation of accommodation phenomenon

In the human visual system, the accommodation of our eyes when changing the gaze point position takes a few milliseconds. Thus, we propose to simulate the phenomenon of accommodation within our DoF blur effect. To do so, we add a temporal filter to the computation of the final focal distance. After

preliminary testing, we chose a low-pass filtering, given by Equation 5.4, with $\tau = (\pi \times fc)/2$, fc the cut-off frequency in Hertz, Te the sampling period in seconds, $\overline{f_d}(n)$ the filtered focal distance at frame n, and $f_d(n)$ the focal distance given by the auto-focus system (before filtering). In our final implementation, we used Te=1/70s, and fc=5Hz.

$$\overline{f_d}(n) = \left(f_d(n) + \frac{\tau}{Te}\overline{f_d}(n-1)\right)\frac{1}{1+\frac{\tau}{Te}} \tag{5.4}$$

5.3.2 Peripheral blur effect

The peripheral blur effect that we introduce simulates the fact that the sharpness of objects decreases when these objects are located at the periphery of the human field of vision. Thus, this effect progressively blurs the pixels that are located at a certain distance from the center of the focus area, and it is independent of the depth-of-field.

$$DCoC_p = \left(\sqrt{\frac{1}{\mathbf{z}\cdot\mathbf{p}} - 1}\right)^n \tag{5.5}$$

In fact, the peripheral blur is naturally present in human vision. In our case, The main objective of the peripheral blur effect is also to encourage the user to look at the center of the screen. Indeed, by slightly blurring the contour of the image, we hope the user will be implicitly forced to look through the visor, i.e., inside the focus area. The computation of the amount of peripheral blur for each pixel (diameter of Circle of Confusion $DCoC_p$) is given by Equation 5.5, in which $\mathbf{z}$ is the look-at direction (e.g., direction of the camera, in the case when the focus area is at the center of the image) and $\mathbf{p}$ is the normalised direction of the pixel in the camera frame. In our final implementation, we used a power n equal to 2.

5.3.3 Final blurred image

5.3.3.1 Computation of the final amount of blur

Once the computation of both the peripheral and the DoF blur is achieved, we can compute the total amount of blur for each pixel, i.e., the final diameter of its circle of confusion $DCoC_f$. We use the contributions of both the peripheral and DoF blurs as follows:

$$DCoC_f = D_{max} \times \min(1, DCoC_d + DCoC_p) \tag{5.6}$$

with Dmax the maximum amount of blur (maximum diameter of the pixel's final CoC), $DCoC_d$ the normalized diameter of the DoF blur's CoC and $DCoC_p$ the normalized diameter of the peripheral blur's CoC. In our implementation, D_{max} was set to 11 pixels.

5.3.3.2 Blur algorithm based on a rotating sampling kernel

Our blur algorithm is based on a gathering blur technique [101]. In order to compute the blur, this technique mixes the color of 12 samples distributed using a Poisson-disk distribution, forming the sampling kernel, taken inside the circle of confusion having a radius of $DCoC_f$. Using this method, ghosting artefacts appear [101], i.e. objects seem to duplicate (Figure 5.4). To improve the blur rendering, we propose

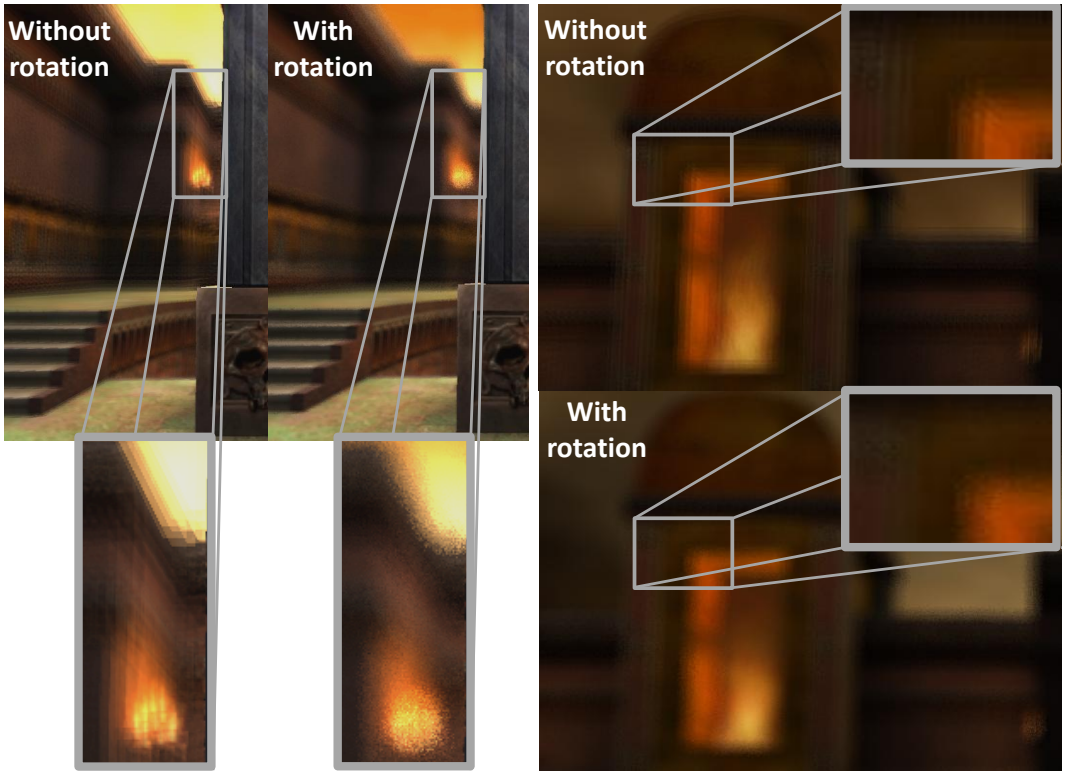


Figure 5.4: Depth-of-field blur rendering. With per-pixel random rotation of the sampling kernel, ghosting artefacts are visible. Without per-pixel random rotation of the sampling kernel, ghosting artefacts are replaced by high frequency noise .

to randomly rotate the sampling kernel per pixel, instead of increasing the number of samples (and thus decreasing performance).

In the case of soft-edged shadow mapping, Uralsky et al. [120] proposed to generate blur by using a set of different sampling kernels stored in a 3D texture. In our case, instead of having many different kernels, we propose to always use the same kernel that will be randomly rotated for each pixel. Thus, the blur computation becomes faster and uses less memory.

For each pixel, we just need information about the 2D rotation of angle α (see Algorithm 1). To construct a two-dimensional rotation matrix, we use $\cos(\alpha)$ and $\sin(\alpha)$ values which are precomputed and stored in a single 2D texture. Finally, we just need to multiply each sample offset defined in the fragment program by this matrix. Using this method, we save a lot of texture memory bandwidth as we only need to read one texel per pixel, as compared to previous methods [120] that require several reading in the 3D texture. As a result, the ghosting artefact [101] is replaced by a high frequency noise (Figure 5.4), a visual cue that human eyes filter out efficiently [120].

Lastly, the elimination of the color leaking artifact, i.e. the blurring of in-focus object on the out-of-focus background, is achieved by simple depth comparisons between the inner and outer samples of the sampling kernel (please refer to for in-depth details [101]). As illustrated in Figure 5.5 the final blur algorithm was successfully implemented in the open source engine of the Quake III Arena™ videogame [51].

Algorithm 1: Pseudo-code of the fragment program computing the blur using a rotation of the sampling kernel

input : Texture texSharp contains the sharp image
input : Texture texCoCD contains pixels depth and circle of confusion size
input : Texture texRot contains per pixel rotation parameters
input : Current texel coordinates **texCoord**
input : Current texel coordinates **texCoordRot** of the rotation texture
input : Samples offsets **samplesOffsets** [12]
input : Maximum circle of confusion diameter *cocMaxD*
output: Pixel final color **Out**
`texRd` (t,c) : read the texture t at the coordinate c

```
//initialization
float pixelCoc = texRd (texCoCD, texCoord).r;
float pixelDepth = texRd (texCoCD, texCoord).g;
vec3 colorSum = texRd (texSharp, texCoord);
float cocSize = cocMaxD ×pixelCoc ;
float totalContrib = 1.0;

//creation of the rotation matrix from cos(α) and sin(α)
rMat = createRMat (texRd (texRot, texCoordRot));

for i=0 to 12 do
    //rotation of the current offset
    vec2 offset = samplesOffsets [i] · rMat ;
    //texture coordinates of the current sample
    vec2 splCoord = texCoord + cocSize ×offset ;

    //sampling
    vec3 splColor = texRd (texSharp, splCoord);
    float splCoc = texRd (texCoCD, splCoord).r;
    float splDepth = texRd (texCoCD, splCoord).g;

    //avoid color leaking artefact using depth comparison
    float sampleContrib = splCoc ;
    if (splDepth > pixelDepth) then
        sampleContrib = 1.0;
    end if

    //sum of all contributions
    colorSum += splCoc × sampleContrib ;
    totalContrib += sampleContrib ;
end for
//output final color
Out = colorSum ÷ totalContrib ;
```

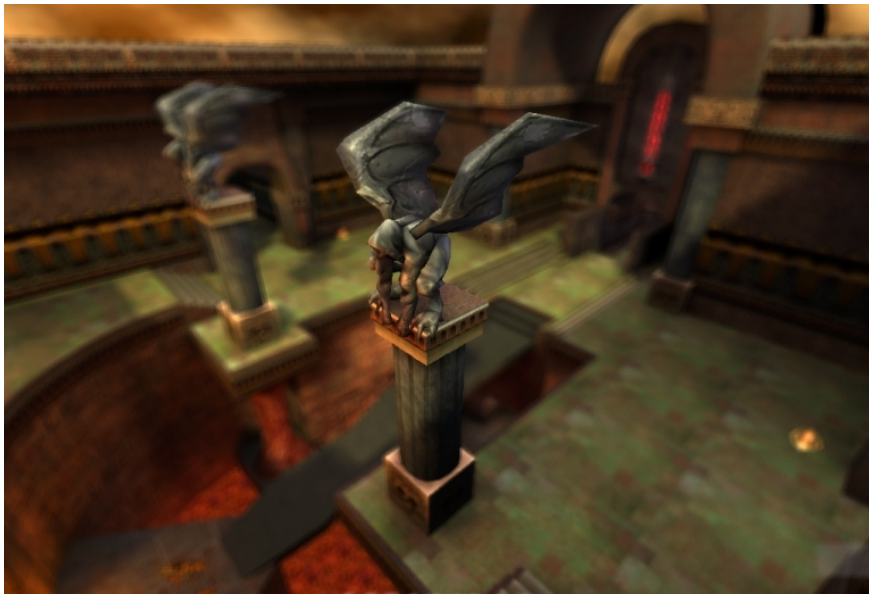


Figure 5.5: Quake III Arena™ videogame with blur effects implemented.

5.4 Implementation

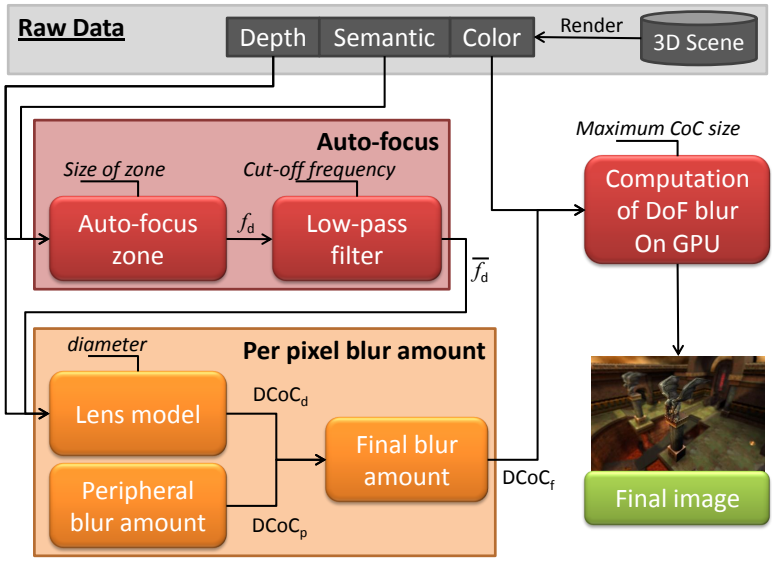


Figure 5.6: Software architecture.

The final software architecture for the computation of visual blur effects is described in Figure 5.6. From the graphic hardware, we get three raw components: the semantic weights, the depths, and the sharp image. Then, the auto-focus algorithm uses the semantic weights, the depths and the computed spatial weights as input in order to determine a focal distance. The focal distance is then filtered by a low-pass filter (accommodation phenomenon). The DoF blur algorithm computes the amount of DoF

blur according to the filtered focal distance and the depths using the lens model. At the same time, the peripheral blur algorithm firstly computes the distances between each pixel and the center of the auto-focus zone. Then it computes the amount of peripheral blur using Equation 5.5. The total amount of blur is computed using the amount of both DoF and peripheral blurs. Finally, the total amount of blur is applied to the sharp image to obtain the blurred image.

For application purpose, our blur effects were implemented in the real-time 3D engine of the famous FPS game Quake III Arena™. Our code is open and available online. We used a desktop computer with an Intel PentiumD CPU at 2.8Ghz, 1.0Go of RAM, and a ATI 1900 Series card with 512Mo of video memory. Table 5.1 shows the performance (framerate) of our videogame application under different conditions: with/without blur effect, with/without accelerated GPU computation, with several screen resolutions and size of the auto-focus zone.

Resolution (pixels)	Framerate (Hz)				
	Without blur	With blur			
		Zone ratio: 0.25		Zone ratio: 0.5	
		CPU	GPU	CPU	GPU
800×600	313	80.0	97.5	55.3	91.5
1024×768	308	65.2	81.7	50.1	78.8
1280×1024	305	48.8	62.2	35.5	61.3

Table 5.1: Framerate of videogame application with and without the blur effects and with various configurations.

5.5 Measurement and analysis of the gaze point during First-Person Navigation

A preliminary experiment has been conducted to measure the gaze point of participants during first-person navigation in a virtual environment. Each of the 6 participants faced various situations of first-person navigation: simple navigation, or first-person-shooter situations in which he had to fight and shoot at enemies. The main objective of the experiment was to analyze the distance between the gaze point of the users and the center of the screen, in order to better set the size of the auto-focus zone introduced in section 5.3.1.2.

5.5.1 Experimental procedure

The gaze point of 6 males with a mean age of 24.2 (standard deviation : 2.8) was recorded during several FPS game sessions using an eye-tracking system. All participants were healthy and had normal or corrected vision. The ASL 6000 eye-tracker system was used with head-mounted optics and a chin-rest to maintain participant's head at the same position. We used a 5:4 monitor screen with a resolution of 1280×1024, positioned 50 cm in front of the participant. The experiment consisted of playing the game Quake 3 Arena™ on the map "Q3DM7" with no blur effects. There were four successive recorded sessions with different conditions: (1) **Navigation**: each participant navigated freely and alone in the map (no enemies) for 3 minutes, (2) **1Enemy**: each participant fought against one enemy for 5 minutes, (3) **4Enemies**: each participant fought against four enemies for 5 minutes and (4) **4EnemiesBonus**: each participant fought against four ennemies for 5 minutes and can pick up life bonus. At each frame, and

during each session, we recorded the 2D position of the gaze point on screen, and whether or not the participant shoots (boolean value).

5.5.2 Results and discussion

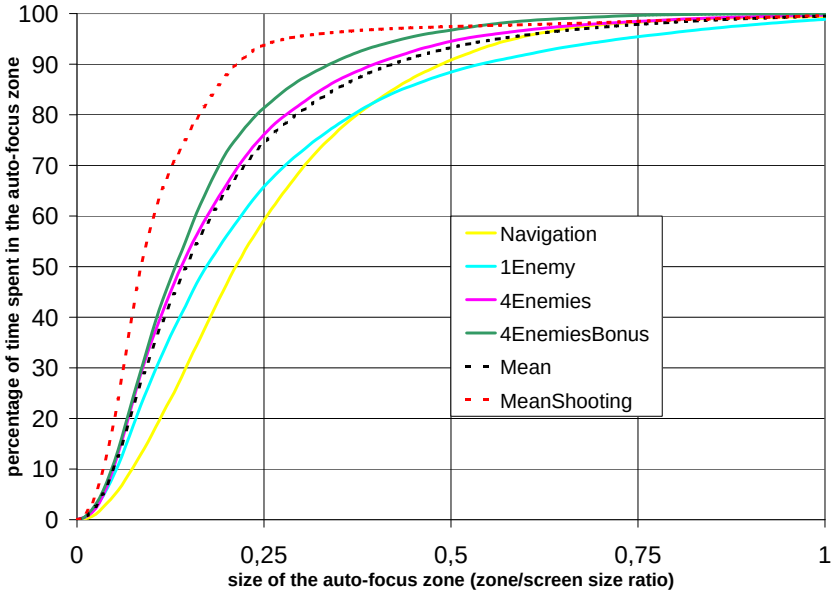


Figure 5.7: First-Person Shooter sessions: Percentage of time spent looking inside a centered zone as function of the size of the zone.

The Figure 5.7 describes the percentage of time spent looking inside a rectangular zone centered on the screen as function of the size of this zone, i.e., as function of the zone-screen size ratio. The zone-screen size ratio corresponds to the ratio between the height of the focus zone and the height of the full image on the screen. This ratio is equal to 1 if the zone covers the entire image and to 0 if the zone is equal to one pixel. If the zone-screen ratio is equal to 0.5, the area of the focus zone corresponds to one quarter of the area of the image.
As expected, participants globally looked close to the center of the screen (Mean Curve). When firing (MeanFiring curve), the gaze point of the participant is naturally getting even closer to the central visor of the FPS. As a result, when the task involves more enemies and thus more shooting (4Enemies Curve), the participants are looking much closer to the center. In other words, the more enemies there are on the map, the more the participants will focus their attention on the center of the screen. In the case of condition 1Enemy, participants probably alternated between navigation and fighting. The 1Enemy curve is thus logically a mix between the Navigation curve and the 4Enemies curve.

On average, when the zone-screen ratio is 0.5, i.e., when the height of the zone corresponds to half the screen, the participants look inside the zone 93% of the time. This result is consistent with previous findings of Kenny et al. [67] who obtained a score of 82%. The difference could be due to the presence of more visual information displayed at the periphery of the screen during the experiment of Kenny et al.

The size of the auto-focus zone described in section 5.3.1.2 can be set using the curves of Figure 5.7. For instance, to ensure that the auto-focus zone will capture 75% of the user's gaze, when fighting against 4 enemies in the FPS, the zone-screen ratio must be 0.25, meaning a rectangle of 320×256 pixels.

5.6 Evaluation of blur effects

A second experiment has been conducted to study the influence of our visual blur effects on the performance and subjective preference of FPS gamers. The DoF blur effect might annoy gamers when they are playing because the displayed image is blurred to some extent. However, we suggest that this effect could make the virtual environment look more realistic and, thus could increase users' feelings of immersion, of being inside the combat and, as a result, their fun during the navigation and their game experience. Thus, prior to the experiment we could formulate the two following hypotheses: (H1) visual blur could degrade performance of participants since the displayed image is blurred to some extent, and (H2) visual blur could improve the subjective preference of users since it provides an additional visual effect that potentially increases the degree of realism of the scene or the fun.

5.6.1 Experimental apparatus

Peripheral and DoF blurs with automatic focal distance computation and accommodation simulation were both implemented in the 3D engine of Quake III Arena^{TM}. Both the semantic and spatial weights were activated. The semantic weights of enemies were constantly set to WSmax while all other semantic weights were constantly set to WSmin. Other numerical values used in our implementation are given in the previous sections.

The Quake III Arena^{TM} map used in our experiment was the map labelled "Q3DM7", with all life packs, special bonuses and weapons removed. We used six PCs connected on a local network with identical graphic cards, monitors and resolutions. There was no perceivable lag. Participants were all provided with an infra-red mouse and a stereo headset for audio feedback. For the purpose of the experiment, the *wait for vertical synchronization* feature of video cards was constantly set to 60 frames per second.

The zone-screen ratio was set to 0.25, i.e., the area of the focus zone covered one eighth of the full image. This value makes sure that the participants looked at least 75% of the time in the auto-focus zone, according to the results of the previous experiment (4Enemies curve in Figure 5.7).

5.6.2 Experimental procedure

The task consisted of playing a First-Person-Shooter game in death-match mode (each player fought against all the other players). Participants were instructed to be as precise as possible when shooting while using as little ammunition as possible. To reduce the variability across subjects, all players had only one weapon (a submachine gun) with unlimited ammunition. The amount of life was increased to 200 points (normal situation: 100 points) to increase the game length.

Thirty participants (28 males, 2 females) with a mean age of 24.0 (SD=2.2) participated in our experiment. 30% of them assessed themselves as expert gamers, 37% as intermediate and 33% as beginners.

A repeated measures within-subjects design was used. The independent variable was the visual effect (VE$_{\text{FFECT}}$) with two levels: no blur effect and blur effects. The experiment lasted 1 hour including breaks. The participants were randomly divided into 5 groups of 6 subjects each. The experiment was then divided into two parts. The first part consisted of a training session (4 minutes) with or without blur followed by the real experiment with a first session of 5 minutes in the same conditions as the training session and a second 5 minutes session with the reverse condition. Participants were asked to fill out a subjective questionnaire after these two sessions. The second part consisted of a performance test including 6 sessions of 5 minutes each. For each of the 6 sessions, 3 participants played with blur and the other 3 participants played without blur. The blur condition was automatically swapped for all players

when running a new session. At the end, a general appreciation questionnaire and a personal form were filled by all participants to know their preferences and skill level. For each part of the experiment, the order of presentation was counterbalanced against participants and groups.

5.6.3 Results

5.6.3.1 Performance

The dependant variables were the number of enemies that the player killed (FRAGS), the number of times the player was killed (DEATHS), the number of his/her shots (TOTAL SHOTS), and his/her precision (PRECISION, in percent) which is computed as the ratio of the number of succeeded shots to the total number of shots.

	No blur effect		Blur effects	
	Mean	SD	Mean	SD
Frags	15.4	6.6	14	6.6
Death	14.4	3.6	15.2	3.4
Total shot	1011.9	271.4	983.4	262.4
Precision (%)	27.0	6.4	25.2	6.2

Table 5.2: Mean and Standard Deviation (SD) for each dependant variable of the performance test, with and without the blur effects.

Repeated measures analyses of variance (ANOVA) showed that the order of presentation of the visual blur effects and groups had no significant effect or interaction on the different dependant variables, indicating that a within-subjects design was appropriate. Repeated measures analysis of variance found a significant main effect for VEFFECT on FRAGS ($F_{1,29} = 8.1$, $p = 0.008$), DEATHS ($F_{1,29} = 5.7$, $p = 0.023$) and PRECISION ($F_{1,29} = 17.3$, $p < 0.0001$). These results show that the PRECISION of participants decreases significantly from 27.1% to 25.2% when blur effects are enabled while the number of FRAGS decreases from 15.4 to 14 and the number of DEATHS increases from 14.4 to 15.2. Interestingly, analyses of variance where the game experience level as reported by each participant is treated as a between-subject factor and VEFFECT as a within-subject factor found a significant interaction between the game experience level and VEFFECT ($F_{2,27} = 4.02$, $p = 0.03$) on PRECISION. This shows that the PRECISION decreases for expert gamers when visual effects are enabled (from 31.2% to 28.7%) and intermediate gamers (from 26.1% to 23.3%) while the precision for beginners remains around 24%.

5.6.3.2 Questionnaire and user feedback

After the two first sessions (with and without blur effects enabled), 21 of the 30 participants were able to notice a difference between the sessions and 20 of them were able to explain that a blur effect was applied to the rendering.

The final and general appreciation questionnaire did not show a significant trend concerning a potential appreciation or detestation of the blur effects. Indeed, participants were balanced concerning: the improvement of realism of the virtual scene (11 preferred with the blur, 13 preferred without, and 6 had no preference), the improvement of fun (9 with, 10 without, 11 no preference), the improvement of perception of depth and distances in the VE (10 with, 14 without, 6 no preference) and the improvement of feeling of presence (11 with, 10 without, 9 no preference).

The participants who preferred the game when the blur effects were activated could be very enthusiastic: “an improvement in realism, especially during volte-face”, “without blur, I felt more visible, I had to hide more”, “funny”, “it focuses my attention”, “surprising, striking”, “much more realistic”, “I have a better precision”, “higher immersion”. It thus seems that many participants would be ready to activate the blur effects for an in-game use.

The participants who preferred the game without the blur effects generally found the blur disturbing and felt tired (headache). They estimated that the blurring effect was responsible for their fatigue (5 participants). Some of them, and more especially some expert gamers, found that the blur was a “discomfort”. Some participants perceived the blur as “too strong”. Furthermore, these people were annoyed by the blur when exploring the image and looking for targets/enemies on the screen.

Meanwhile, a majority of participants (83%, 25/30) declared that the use of blur did not modify their gaming strategy.

5.6.4 Discussion

Our results corroborate the hypothesis H1 that blur effects would degrade performance. Players were indeed less accurate during shooting. This may explain that their score, i.e. their number of frags, was reduced by about 7%. The number of deaths was slightly increased but this difference is less relevant as it was less than one frag. The fact that, with blur effects enabled, players were more often killed and that, in the same time, they killed less enemies and shot with less precision could indicate that the blur effects make the game harder. However, this result is only significant for more expert gamers who are more used to play the game with a low visual quality and all special effects disabled. These players have invested a lot of strategy and tactics in the original game and any change reduces the advantage of that investment. This stresses the importance of learning and “resistance to change” that must be taken into account during the design of the visual rendering and gameplay of a videogame.

Our second hypothesis H2 was that blur effects could improve subjective preference concerning factors like fun of the game. Half of the players who gave an opinion preferred the presence of blur in terms of fun, presence, and realism of the virtual environment. We also noticed that the same number of participants disliked the effects during the experiment. Despite the fact that these results only partially support H2, it seems sufficient for these blur effects to be recommended, for instance in videogames.

The comments of several participants suggest that they did not constantly look inside the focus area, i.e., at the center of the screen. There are at least two phases that can be stressed. In one phase, the player shoots at the enemies with his/her visor. In this case, the focal distance computed was always well adapted to the user’s gaze. In a second phase, the player explored the image to find and identify enemies. During this phase, the focal distance computed by our model might not correspond to the actual attention of users. For some of them, this situation was problematic and it generated both discomfort and fatigue. As mentionned in Section 5.5.2, we get 75% of user’s gaze with an auto-focus zone with a zone-screen ratio of 0.25. This seems to be sufficient for many players who liked the blur effect but not for others. It seems to depend strongly on players’ gaming strategy.

Some participants complained that the blur was too strong. This might explain the fatigue observed and expressed by some of them. It also suggests that the intensity of the blur effects should be carefully tuned, or that the user should have the possibility to change the amount of blur in his/her application.

To our best knowledge, for the new generation of games, it seems that developers simply use the depth of the pixel located at the center of the screen to compute the focal distance. This straightforward technique is likely to increase users discomfort and degrade gamers subjective preferences. We encourage developers to compute the blur effect parameters as proposed in this chapter and provide the option

for users to adjust the blur strength as well as enable/disable the blur effect.

Last, this study was more specifically focused on FPS videogames which provide a challenging scenario for testing DoF blur effects. However, we could imagine other applications in which these visual blur effects could be valuable. The blur effect could be a visual effect around which the game could be designed. The game designer could use it for instance to intentionally focus the player or to distract him/her. In other VR applications, such as for architectural creation and project reviews, the depth-of-field blur could be used to improve the perception of depths and distances in the virtual world. However, further work is now necessary to investigate the future applications and uses of visual blur effects in other contexts and other virtual environments.

5.7 Conclusion

In this chapter, we have studied the use of visual blur effects for first-person navigation in virtual environments [46][47].

First, a model of dynamic visual blur was introduced, based on two types of blur effect: a depth-of-field blur which simulates the blurring of objects related to focalization, and a peripheral blur that simulates the blurring of the periphery of the field of view. To make the DoF blur effect more suitable in VR applications, we have developed both a technique to compute the focal distance automatically on the graphic card, and proposed a temporal filtering of the focal distance to simulate the accommodation phenomenon. Our blur effect does not suffer from ghosting artifact thanks to depth comparisons and per pixel random rotation of the sampling kernel.

We have provided the results of a study used to understand the viewing-behavior of gamers playing a first-person shooter. We have noticed that they look most of the time at the center of the screen. These results were used to set the parameters of our algorithm. Finally, we have reported on an experiment conducted to study the influence of this DoF blur effects on performance and subjective preference of FPS gamers. The study has first revealed that only half of the participants have appreciated the blur effect. Furthermore, it also revealed that expert gamers were significantly less effective when the blur effect was activated. This could be mostly due to the fact that we were changing their habits as they are used to play with low visual quality. Taken together, our results suggest that using blur effects could be suitable for video games and for other virtual environments. However, such an effect should remain an option that can be deactivated.

This chapter addressed the rendering of a depth-of-field blur effect without using a gaze tracking system. However, the depth-of-field blur effect should be adapted in real-time to user's 3D gaze positions in the VE. This last issue is addressed in the next and final chapter of this manuscript.

Chapter 6

Automatic adaptation of visual feedback based on user's gaze point : adaptive depth-of-Field blur and camera motions

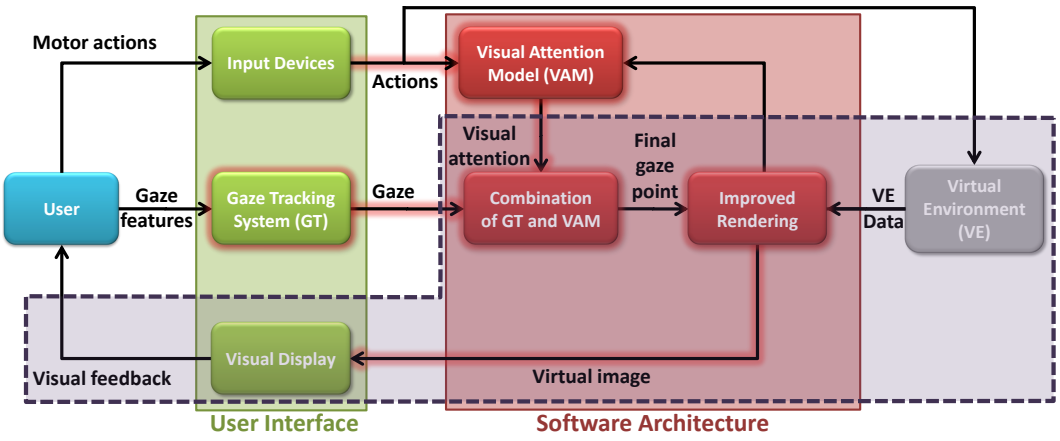


Figure 6.1: Improving visual feedback using gaze.

6.1 Introduction

The user's gaze point can give developers several advantages. For instance, it has often been used in order to manage LoD of a virtual scene [78] [74]. These methods accelerate the rendering process of a virtual scene without the user noticing the drop of quality in regions he is not gazing at. As a complement to these approaches, we propose a novel use of user's gaze point: adding perception-based visual effects adapted to user's gaze point when rendering a virtual scene. These visual effects are aiming at improving the visual feedback of virtual environments by simulating several natural phenomenons occurring in human vision.

In this chapter, we propose the adaptation of two rendering techniques to take into account user's gaze point: (1) a camera motion which simulates eyes movement when walking, i.e., corresponding

to vestibulo-ocular and vestibulocollic reflexes of the eyes compensating body and head movements in order to maintain gaze on a specific target, and (2) a Depth-of-Field (DoF) blur effect which simulates the fact that humans perceive sharp objects only within a range of distances around the focal distance.

We also report on an experiment conducted to study users' subjective preferences concerning these visual effects during free first-person navigation in VE. We particularly compare users preferences toward these effects when they are computed with or without gaze-tracking.

The chapter is organized as follows: Section 6.2 presents the architecture used to compute user's 3D gaze position and focal distance using gaze-tracking. Section 6.3 and 6.4 describes the two gaze-based visual effects we propose in order to improve the visual feedback to the user. Following the preliminary conclusion on these effects in Section 6.5, Section 6.6 reports on the results of the experiment conducted to study participants preferences concerning these visual effects when computed with or without a gaze-tracking system.

6.2 Computation of focal distance and 3D gaze point in 3D virtual environments using a gaze tracker

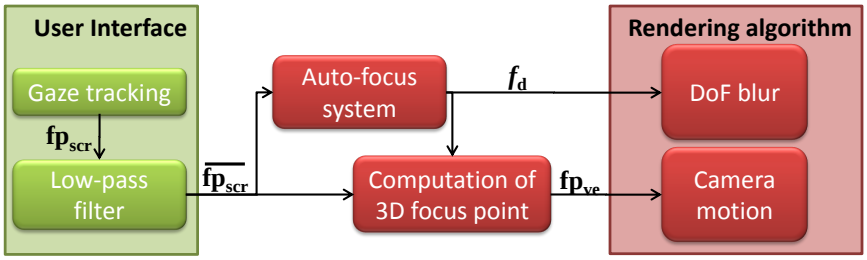


Figure 6.2: Software architecture.

In this section, we describe an algorithm in four steps to compute both the focal distance f_d and gaze point $\mathbf{fp_{ve}}$ in the 3D VE, using the output of an eye-tracking system, i.e., the gaze point on the screen $\mathbf{fp_{scr}}$. The architecture of our algorithm is illustrated in Figure 6.2.

1. **eye-tracking**: we simply recover the position of the users' gaze point on screen $\mathbf{fp_{scr}}$ corresponding to the eye-tracker's 2D output[1].

2. **low-pass filtering**: we compute $\overline{\mathbf{fp_{scr}}}$ which corresponds to the low-pass filtered $\mathbf{fp_{scr}}$ using a cut-off frequency of 15Hz to avoid high frequency jerks.

3. **auto-focus system**: the accuracy of the eye-tracker's output is unfortunately superior to one pixel. Thus, in some cases, the user might look at other pixels located near $\overline{\mathbf{fp_{scr}}}$, or to the shape of an object close to $\overline{\mathbf{fp_{scr}}}$. The depth of a single pixel does not seem sufficient to estimate f_d. Thus, we use the same auto-focus system, as described previously in Section 5, with a square focus zone centered on $\overline{\mathbf{fp_{scr}}}$, as shown in Figure 6.3. The computation of f_d, using the pixels located inside the focus zone, is achieved by averaging the weighted depth of each pixel in the focus zone. The weight of each pixel is computed using the pixel semantic value and the distance to $\overline{\mathbf{fp_{scr}}}$. The

[1]In this chapter, the users' gaze point is retrieved using a gaze tracking system. It is important to note that we could have used the visual attention model described in Chapter 3.

closer the pixel is to $\overline{\mathbf{fp_{scr}}}$, the higher the weight is. If the pixel corresponds to a semantically important object (e.g., a target) its weight is also increased. For further details, see Section 5.

4. **gaze point in VE**: first, we compute user's viewing direction $\mathbf{V_{cam}}$ expressed in the camera reference frame using $\overline{\mathbf{fp_{scr}}}$. Then, we transform it into the reference frame of the VE to obtain $\mathbf{V_{ve}}$. Finally, we can compute user's 3D gaze point in VE $\mathbf{fp_{ve}}$ using Equation 6.1, in which $\mathbf{C}$ is the camera position and f_d the focal distance computed in the previous step.

$$\mathbf{fp_{ve}} = \mathbf{C} + f_d \times \mathbf{V_{ve}} \tag{6.1}$$

Finally, the computed gaze point and focal distance can be sent to the two rendering algorithms described in the following sections: a camera motion and a DoF blur effect.

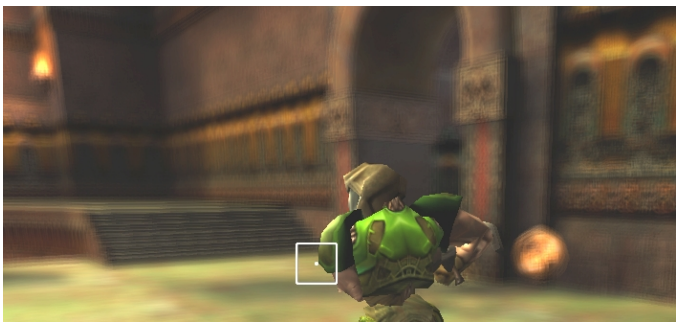


Figure 6.3: Quake III video game with Depth-of-Field blur effect implemented. The white square corresponds to the auto-focus zone that follows the user's gaze point measured by eye-tracking.

6.3 Camera motion for first-person navigation based on user's gaze point

We propose a first effect based on user's gaze: a dynamic compensated camera motion. This is an improved version of Lécuyer et al. [71] camera motion model.

Camera motion is sometimes used in First-Person-Shooter games. It generally consists in applying a sinusoidal motion to the virtual camera to simulate the visual flow corresponding to a walking motion. Lécuyer et al. [71] suggested to improve the classical techniques by changing the camera orientation in addition to the change in position. The compensated orientation stabilizes the optic-flow at the center of the screen. Indeed, it is supposed to simulate eye movements resulting from vestibulo-ocular reflexes [52] allowing a human to fixate a point in space while walking with moving body and head. However, they did not use an eye-tracking system and the camera was always focusing on an object located at the center of the screen. Moreover, they did not provide further details on how to implement such camera motion with an eye-tracking system.

The first step of the computation of the camera motion consists in changing the camera position to simulate walking motion. We apply three sinusoidal offsets to the original camera position (see Figure 6.4-A) on three axes: up vector of the VE $\mathbf{z_{ve}}$, left vector $\mathbf{y_c}$ and forward vector $\mathbf{x_c}$ of the camera. We compute the amplitude of each offset on each axis using Equation 6.2:

$$l_a = K_a \times \sin(2\pi \times \frac{t}{T_a} + O_a), \forall a \in \{\mathbf{x_c}, \mathbf{y_c}, \mathbf{z_{ve}}\} \tag{6.2}$$

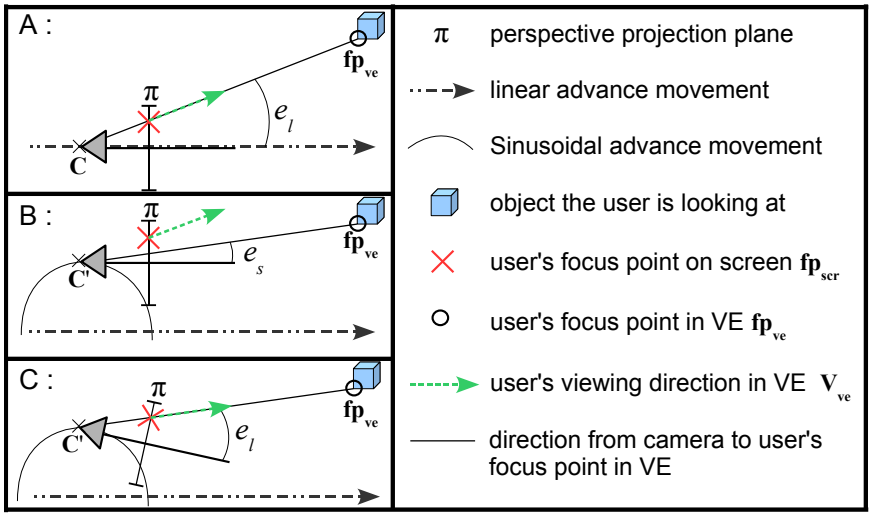


Figure 6.4: computation of the camera orientation using user's gaze point.

where t is the time and a one of the three axis. For each axis a, T_a is the period of walking motion, K_a a constant amplitude, O_a the initial phase and l_a is the resulting amplitude of the offset for the current frame. The final camera position $\mathbf{C}'$ (see Figure 6.4-B) is obtained using Equation 6.3.

$$\mathbf{C}' = \mathbf{C} + l_{x_c} \times \mathbf{x_c} + l_{y_c} \times \mathbf{y_c} + l_{z_{ve}} \times \mathbf{z_{ve}} \tag{6.3}$$

The second step of our computation consists in changing the orientation of the camera. We compute the change in orientation of the camera so to keep $\mathbf{fp_{ve}}$ projected on $\overline{\mathbf{fp_{scr}}}$ on the projection plane π.

Let us note (α, β, γ) the roll, pitch and yaw angles which represent the original camera orientation and $(\alpha', \beta', \gamma')$ the angles of the desired camera orientation as shown in Figure 6.4C. We compute the final orientation using Equation 6.4:

$$\begin{pmatrix} \alpha' \\ \beta' \\ \gamma' \end{pmatrix} = \begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix} + \left(\begin{pmatrix} 0 \\ e_s \\ a_s \end{pmatrix} - \begin{pmatrix} 0 \\ e_l \\ a_l \end{pmatrix} \right) \tag{6.4}$$

where (e_l, a_l) are elevation and azimuth angles of $\mathbf{fp_{ve}}$ in camera reference frame when following the initial linear motion as shown in Figure 6.4-A, i.e., before offsets are applied to camera position C. Then, (e_s, a_s) are elevation and azimuth angles of $\mathbf{fp_{ve}}$ in the camera reference frame at position $\mathbf{C}'$ in Figure 6.4-B. In our final implementation, α and α' angles were set to 0. Our other parameters were set using preliminary testing to: $(T_{x_c}, T_{y_c}, T_{z_{ve}}) = (0ms, 780ms, 390ms)$, $(K_{x_c}, K_{y_c}, K_{z_{ve}}) = (0, 0.015, 0.02)$ and $(O_{x_c}, O_{y_c}, O_{z_{ve}}) = (0, \pi/2, 0)$.

The resulting camera motion is thus compensated in real time based on the point the user is looking at in the VE, thanks to the eye-tracking system.

6.4 Depth-of-Field blur effect based on user's gaze point

The second effect we propose to improve visual feedback using users' gaze point is a DoF blur effect.

Brooker et al. [17] conducted an evaluation of DoF blur effect using a stereoscopic display. However, their results did not show evidences of performance improvement when the DoF effect was computed

using an eye-tracker. They concluded that it could be due to the very slow frame rate of their application and they suggested to implement and further evaluate real time DoF blur effect in VE. In Section 5 of this PhD thesis, we have shown that, in absence of an eye-tracking system, only a half of the participants enjoyed a DoF blur effect computed with a focus zone constantly positioned at the center of the screen. However, we have suggested that using eye-tracking system could improve the results. Thereafter, we describe the implementation of a DoF blur effect adapted in real time to user's focal distance in VE thanks to an eye-tracking system and to the algorithm described in section 2.

The computation of the DoF blur effect is achieved using the classical techniques described in Section 5. We also simulate temporal focal distance accommodation using a low-pass filter. To compute the amount of blur per pixel we use a lens model that takes as parameter this focal distance. Then, by applying a gathering blur technique, we obtain the final blurred image without color leaking, thanks to depth comparisons. The implementation parameters are the same as in Section 5

Figure 6.3 illustrates our algorithm implemented in real time in the Quake III video game engine [51]. Thanks to the focus zone centered on $\mathbf{fp_{scr}}$, the focus is done on the semantically important character instead of the background, even if the character covers few pixels in the auto-focus zone.

6.5 Preliminary Conclusion

We have enhanced two existing techniques suitable for first-person navigation in VE to make them work with an eye-tracking system: (1) a compensated camera motion and (2) a DoF blur effect. They have been successfully implemented in the real time open-source engine of Quake III video game [51]. On a PC with an Intel PentiumD CPU at 3.4Ghz, 2.0Gb of RAM and a nVidia Quadro FX 3450/4000 SDI, performance was of 70 frames per second with the both effects activated, instead of 260 without, at a resolution of 1280×1024 pixels. The video game with our techniques implemented and the hardware configuration aforementioned were used in the experiment described in the next section.

6.6 Experimental evaluation

An experiment has been conducted to study the subjective preference of users regarding the DoF blur effect and camera motion when eye-tracking is enabled or not.

6.6.1 Apparatus

We used our camera motion and DoF blur effect implemented in the video game Quake III as described in section 3 and section 4. In all test sessions, we used Q3DM7 map. The player was alone (no targets) and could move freely. No information was displayed on the screen but the classical central visor. The initial altitude of the camera was set to 50 units.

We used an immersive room with a cylindrical screen (3.8m radius, 2.35m height and 45 degree of arc) with a display resolution of 1280×1024 pixels. The participants were positioned at the center of the cylindrical screen and no sound was played during sessions.

We computed participant's gaze point position on screen by using ASL 6000 eye-tracker system with head-mounted optics. We used a chin-rest to maintain participant's head at the same position.

6.6.2 Participants

Eight men with a mean age of 25,8 (SD=4,3) participated in our experiment. All subjects were familiar with first-person navigation and had normal or corrected vision with lens. Five participants were right handed. All subjects were unaware of our work.

6.6.3 Experimental plan

First, each participant navigated in the VE during 3 minutes as a training session. Then, the experiment was divided into two parts.

The first part consisted in testing four conditions of camera motion: (1) Control (basic linear motion as if the user was driving a car) ; (2) Mv (sinusoidal movement without motion compensation) ; (3) Mv_comp, (sinusoidal movement with motion compensation computed using a focus zone constantly located at the center of the screen) ; (4) Mv_comp_eyeT (sinusoidal movement with motion compensation computed using a focus zone centered on user's gaze point).

The second part consisted in testing three conditions of DoF blur effect: (1) Control (normal scene rendering without DoF blur) ; (2) DOF (scene rendered with DoF blur effect computed using a focus zone constantly located at the center of the screen) ; (3) DOF_eyeT (scene rendered with DoF blur effect computed using a focus zone centered on user's gaze point on screen).

For each part and each participant, the presentation order of each condition was randomized. At the beginning of each part, the eye-tracker was calibrated and participants were informed of what is going on in each condition: the type of camera motion or DoF blur effect used, if gaze-tracking was used or not, etc. Each condition was tested during 5 minutes and the experiment lasted 45 minutes. The eye-tracker was worn during the whole experiment.

At the end of each part, participants were asked to fill up a global appreciation questionnaire during which they were free to test each condition at runtime by selecting them using keyboard keys.

6.6.4 Results

The data were analyzed using the Friedman test followed by Wilcoxon post-hoc analyses.

6.6.4.1 Camera Motion

Participants were asked to grade the four conditions of camera motion using a seven-point Likert scale according to four subjective criteria: (1) impression of walking in the virtual world, (2) "fun", (3) depth perception and (4) immersion in the virtual world. The results are displayed in Figure 6.5.

Friedman analysis revealed an overall significant difference between the techniques for the **impression of walking** in the VE ($\chi^2(3)=18.8$, $p<0.0001$). Bonferroni-corrected Wilcoxon post-hoc analyses confirmed that all sinusoidal movements are significantly preferred compared to the basic linear motion ($p<0.004$). Marginal significant difference between Mv and Mv_comp_eyeT techniques (z-score=-1.8, $p=0.047$) suggest that the compensation improves the impression of walking only when the gaze point is used. Friedman analysis showed a significant main effect across navigation techniques in term of **fun** ($\chi^2(3)=13.6$, $p<0.004$). Post-hoc analyses revealed that the Mv_comp_eyeT technique is significantly different from the others ($p<0.02$). A significant difference between Mv_comp and Control (z-score=-2.3, $p=0.012$) shows that the motion compensation improves the navigation in term of fun. Friedman analysis showed a significant main effect across techniques for the **depth perception** ($\chi^2(3)=12.8$,

p<0.005). Post-hoc analyses revealed that the Mv_comp and Mv_comp_eyeT techniques are significantly different from the Control (p<0.03). Friedman analysis showed a significant main effect across techniques for the **immersion** in the virtual world ($\chi^2(3)=15.9$, $p < 0.001$). Post-hoc analysis revealed significant difference between Mv_comp_eyeT and all other techniques (p<0.02). Significant difference were also found between Control and all other techniques (p<0.03) and between Mv and Mv_comp_eyeT (z-score=-2.2, p=0.016).

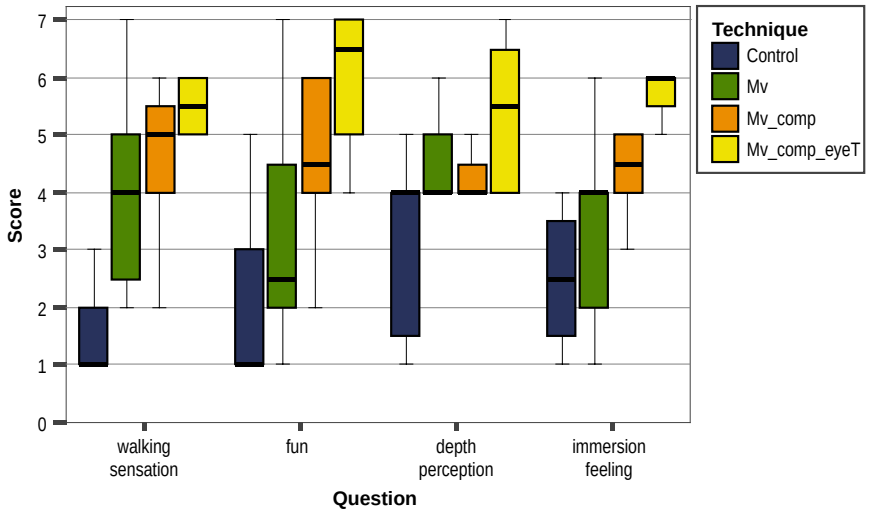


Figure 6.5: Box plot representation of the camera motion subjective preference for each question and each technique.

6.6.4.2 Depth-of-Field blur

Participants were asked to grade the three conditions of DoF blur effect using a seven-point Likert scale according to four subjective criteria: (1) rendering realism, (2) "fun", (3) depth perception and (4) immersion in the virtual world. The results are displayed in Figure 6.6.

Friedman analysis showed no overall significant difference between the techniques for the **rendering realism** ($\chi^2(2)=5.4$, p=0.067). Friedman analysis revealed an overall significant difference between the techniques in term of **fun** ($\chi^2(2)=12.1$, p=0.002). Bonferroni-corrected Wilcoxon post-hoc analyses confirmed a significant difference between DOF_eyeT and the two other techniques (p<0.01). Friedman analysis showed a significant main effect across techniques for the **depth perception** ($\chi^2(2)=8.3$, p=0.016). Post-hoc analysis revealed significant difference between DOF_eyeT and the two other techniques (p<0.05). Friedman analysis showed a significant main effect across techniques for the **immersion** in the VE ($\chi^2(2)=13.2$, p<0.001). Post-hoc analysis revealed significant difference between all techniques (p<0.03).

6.6.5 Discussion

First, concerning the camera motion effect, we found similar results as Lécuyer et al. [71]. Indeed, here also, sinusoidal camera motions are preferred compared to the linear motion. When eye-tracking is not used, the addition of a motion compensation (Mv_comp) is marginally preferred to the sinusoidal camera motions but, as in [71], the result is not significant. Then, we found here that the results are better when

the compensated camera motion is computed using eye-tracking. Indeed, with the eye-tracking system in use, the participants significantly felt more fun and more immersed in the virtual world. The preference failed to reach significance concerning depth perception and walking sensation, possibly due to the small number of participants. Besides, some participants noticed that the difference between Mv_comp and Mv_comp_eyeT conditions was difficult to perceive, except when they were close to virtual objects. This is due to the fact that variations of camera orientation are stronger when the focal distance is close to the eyes.

Concerning the DoF blur effect, the analysis shows a strong influence of using the eye-tracking system. As in Section 5, the DOF condition is not clearly preferred when compared to the Control condition. The DoF blur effect seems to be preferred only when eye-tracking is used (DOF_eyeT condition). In this case, it was significantly preferred in terms of fun and immersion feeling. Furthermore, participants felt that they better perceived the depth of the virtual scene. One participant even said "Sometimes, it's like looking at an auto-stereoscopic screen". It suggests that the DoF blur effect could be used to convey additional depth cues concerning spatial relations between objects, like shadows do.

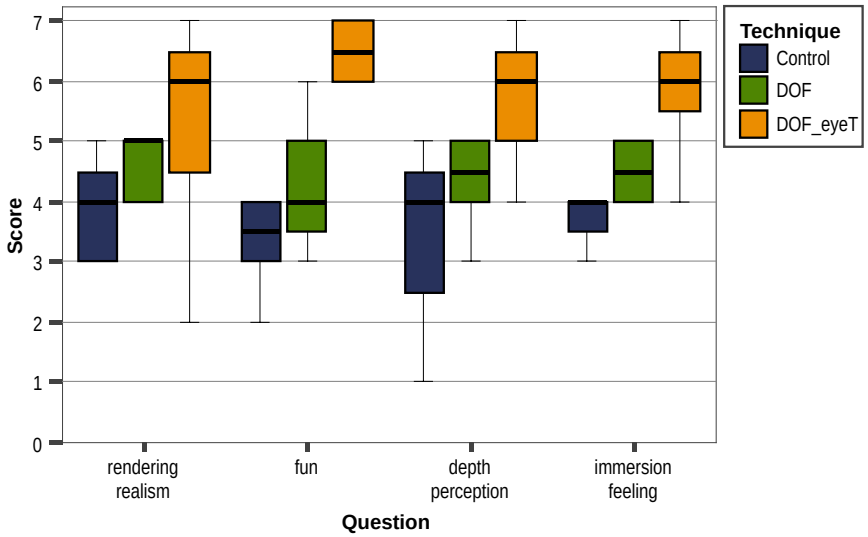


Figure 6.6: Box plot representation of the Depth-of-Field blur subjective preference for each question and each technique.

6.7 Conclusion

In this chapter, we have described the use of user's gaze point on screen to dynamically adapt perception-based visual effects during first-person navigation in VE [48]. We have proposed (1) a compensated camera motion simulating the automatic rotations of the eye in order to compensate for head and body movements (corresponding to vestibulo-ocular and vestibulocollic reflexes), and (2) a DoF blur effect simulating the fact that the crystalline lens in the human eyes acts as a lens and generates blur on out-of-focus objects. Both these effects are adapted in real time to user's gaze point in the VE.

We have conducted an experiment to study users preferences regarding these two effects during free first-person navigation in 3D VE. Results revealed that participants significantly preferred these effects when they were dynamically adapted to their gaze point in the 3D VE. In this case, these effects were

found to improve sensations when exploring the VE, resulting in a better immersion in the VE. These promising results suggest that the use of perception-based visual effects adapted in real-time to users' gaze point could be used in several VR applications involving first-person navigation such as the visit of architectural sites, training simulations, video games, etc.

Conclusion

In this PhD manuscript, we have investigated the use of perception based visual effects adapted in real-time to the user's gaze point. These visual effects are aiming at being used in 3D VR applications featuring first-person navigation. These researches were addressed through two main axes: 1) improving gaze-tracking in 3D virtual environments using human visual attention and gaze behavior, and 2) improving visual feedback in 3D virtual environments based on the user's visual attention.

In Chapter 2, concerning human visual attention, we have first investigated the development of novel models predicting human attention and adapted to first-person navigation in 3D VE. We have first conducted an experiment to analyze the gaze behavior of users when actively and passively walking and turning in 3D VE. This study revealed that the gaze behavior when walking in virtual environments is similar to the one observed during real pedestrian walks. Second, we have proposed three simple gaze prediction models, taking as input the rotation velocity of the camera and/or the optical flow on the screen. Our results show that these models can help predicting the user's gaze point position on the screen during free first-person navigation in VE.

In Chapter 3, we have included one of these models in a broader novel visual attention model automatically computing the user's gaze position on screen during exploration of 3D VE using a first-person view. Our model is the first to include a novel representation of visual objects based on surfels rather than meshes. This new representation is better suited to compute some attentional components, e.g. habituation, and takes advantage of GPUs parallel computational power in order to increase the real-time performance of the simulation. In addition, our model includes several state-of-the-art bottom-up and top-down components. Among all visual attention models, the model of Lee *et al.* [73] was the first being specifically designed for first-person navigation in a VE context. However, it can only compute a set of objects possibly gazed by the user. In this context, our model is the first being able to compute a continuous 2D gaze point position on screen by combining the bottom-up and top-down attentional components. We have conducted an experiment which showed that our model performed significantly better than a state-of-the-art model when exploring various 3D VE with an increase in gaze prediction accuracy of more than 100%.

Finally, in Chapter 4, we have described a novel method to increase the accuracy of any gaze tracking system. On the one hand, gaze tracking systems only rely on a video of the user to estimate his gaze position, but do not take into account what the user is looking at. On the other hand, visual attention models take into account what the user is looking at in the VE to simulate his visual attention. However, in this case, the user's attention is only simulated as a general case. We propose to gather the strength of both approaches by combining them for the first time. First, because existing gaze trackers have a limited accuracy, we consider their output as an uncertainty windows, representing the area in which the user is gazing at, instead of a single point. Second, we compute a saliency map of the image displayed on the screen. Last, our method searches inside the uncertainty window for the highest salient position

which represents the point the user is more likely to gaze at. This point represents the final estimated gaze position. We have conducted two experiments to compare the performance of a low-accuracy gaze tracker, based on artificial neural networks and a web-cam, combined with our method to an accurate gaze tracker. The results revealed that our approach can significantly improve the accuracy of a gaze tracking system. Furthermore, despite the fact that gaze prediction errors can occur, we have found that a low accuracy gaze tracker combined with a pure bottom-up visual attention model was able to perform better than an accurate gaze tracker in a shooting game using gaze-based targeting. To sum up with, our results show a positive influence of our method on gaze-tracking accuracy.

In order to put these results in relation with our initial objectives, we could mention that our first research axis (*improving gaze-tracking in 3D virtual environments using human visual attention and gaze behavior*), has been first addressed by the proposition of new gaze simulation models. These models are based on the fact that we have observed a common gaze behavior of humans when walking in virtual environments as compared to real walk. Then, this axis has been further addressed by the proposition of a complete visual attention model simulating human attention during real-time first-person navigation in 3D VE. Overall, the results of the model we have proposed to compute human attention are significantly better when compared to state-of-the-art approaches and seem to have a positive impact on accuracy. Last, this research axis has also been addressed by the proposition of a novel method combining any existing gaze tracking systems (to estimate the rough gaze position) with a visual attention model (to refine the gaze position by taking into account the image observed by the user). Using this method, we have found that a low accuracy gaze tracker can achieve better performance than an accurate gaze tracker in some cases. Furthermore, this novel approach, taking into account what the user is looking at on the display peripheral, can be used to increase the accuracy of any gaze trackers.

In a second part, we have proposed methods to improve the visual feedback of 3D applications to the users based on their visual attention and human visual perception.

For this purpose, in Chapter 5, we have first developed a depth-of-field blur effect algorithm adapted to first-person navigation. We have proposed an auto-focus system to automatically compute the parameters of this depth-of-field blur when no gaze-tracker is available (as for most platforms today). This is achieved by taking into account the top-down task-relevance of objects, i.e. their semantic. Then, we have studied the influence of using such a blur effect without gaze tracking on gamers' performance during first-person shooting game sessions. Our results revealed that half of the participants appreciated the blur effect and that it only impaired the performance of expert gamers. This could be due to the fact that expert gamers are used to play using simplified graphics, i.e. we were changing their habits.

Finally, in Chapter 6, we have proposed a novel use of the user's gaze point: adding perception-based visual effects adapted to the user's gaze point when rendering a virtual scene. These visual effects are aiming at improving the visual feedback to the user by simulating several natural phenomenons occurring in human vision. We have proposed two real-time gaze-based visual effects: (1) a compensated camera motion and (2) a DoF blur effect. Then, we have conducted an experiment to compare users preferences concerning these effects when they were computed with or without gaze tracking. The results revealed that participants significantly preferred these effects when they were computed using a gaze-tracking system. This suggests that several other effects could benefit from being adapted to the user's gaze point.

Taken together, our results suggest that visual attention models could actually be used either to improve existing gaze tracking systems or to single-handily estimates the user's attention and gaze position during real-time first-person navigation in VE. Also, adding perception-based visual effects adapted in real-time to the user's gaze is a promising approach that could be used to convey to users additional vi-

sual cues, e.g. depth cues, and sensations when navigating in 3D VE. To sum-up, our results suggest that these methods could be used all together in various applications such as video games or VR applications to increase the user's perception, immersion and sensation in real-time 3D virtual environments.

Future work

The work presented in this manuscript leaves some questions unanswered which could be addressed in short-term future work.

In Chapter 2, we have proposed a model to simulate the gaze behavior of users navigating in VE using a keyboard and a mouse. However, this model is specific to our experiment condition. As a consequence, to develop a more general model, it would be interesting to evaluate the influence of several parameters such as walking speed, mouse sensitivity, interaction scheme (joystick vs. mouse; free navigation vs. task) as well as display peripheral size, i.e. field-of-view, on the gaze behavior of users. The same process could then be reproduced to research for attentional components adapted to other contexts such as third-person view navigation or driving simulations. The specific gaze behavior of these contexts could be studied and modeled using analogous approaches to the one we used. Furthermore, apart from visuals features, sounds can also attract attention [113]. As a consequence, future work could deal with including an attentional component taking into account sound sources position in order to propose a multi-modal visual attention model.

Our visual attention model features several bottom-up and top-down components. Usually, top-down components are uniformly weighted. However, the weight of these components could be dynamically modified according to the user's current task. Future researches could deal with the dynamic adaptation of these weights. This complex dynamic weighting could be learned by replaying several training sessions and using an optimization algorithm to evaluate the weights according to the user's current state and progression of the task.

Our visual attention model is able to compute a single gaze point by combining several bottom-up and top-down attentional components. This estimated gaze point is processed by the gaze pattern simulator which currently is a simple low-pass filter. We believe that simulating more realistic gaze movement patterns, such as fixation-saccade or smooth pursuit, could help to predict human attention through time. The effectiveness of this contribution could then be evaluated by analyzing the predicted scan-path as compared to the ground truth scan-path.

A visual attention model is a complex association of several data representations, e.g. surfel or pixel, expressed in different spaces, e.g. screen or 3D world. Actually, these representations are dependents of several parameters such as screen resolution, surfel size in world space, saliency map resolution, etc. It seems that these parameters are often set empirically. We would like to further evaluate the influence of all these parameters on the efficiency of our visual attention model. This could be done by replaying several sessions and comparing the final prediction accuracy.

Taking into account the user's gaze point position when rendering a virtual environment seems highly valuable as discussed in Chapter 6. Indeed, increasing the visual feedback of the virtual environment to users by adding gaze-based visual effects seems to effectively increase their immersion feeling and perception of the VE [117][82]. Having these results in mind, it would be interesting to simulate other natural phenomena occurring in human vision. One novel perception-based visual effect could be a gaze-based luminance adaptation method. Several tone-mapping methods have been proposed to display high dynamic range images on low dynamic range display peripherals [100]. In real life, if a human looks at the sun or at the ground, his eyes will adapt to change the perceivable range of luminance in

order to better perceive the environment. Thus, it would be interesting to propose a dynamic gaze-based luminance adaptation model to reproduce this behavior in low dynamic range physical environments. This would be particularly interesting to perceive the details of a virtual environment rendered with high dynamic range lighting on low dynamic range display peripherals. This should be even more beneficial in the case of large display peripherals like CAVE systems.

Last, our experiments were conducted in a limited number of VE and associated tasks: free navigation, shooting or pick-up tasks. It would be valuable to study users preferences and performances concerning these perception-based visual effects in other cases, e.g. when they are less stressed than in a first-person-shooting game, involving other tasks and scenarios. Furthermore, the gaze-based depth-of-field blur and compensated camera motion could be further investigated in order to study their influence on cybersickness using a cybersickness questionnaire [66].

Perspectives

In addition to the future works mentioned above, this PhD thesis also paved the way to further long-term views. Some of these aspects are described below.

Gaze-tracking for the mass

Nowadays, there is an active research community working on novel interaction peripherals. Sophisticated interaction peripherals for gaming have been recently commercialized such as the Nintendo *Wiimote* and the Sony *motion controller* to interact using hands movements, or Microsoft *Kinect* to interact using the whole human body.

We believe that gaze trackers are systems that could represent another big step in human-computer interaction devices for daily use. Indeed, reliable systems are currently expensive [118] but researchers have already started working on low-cost remote systems, e.g. web-cam based, that can compute a gaze point without requiring any calibration [126]. These systems are not very accurate but results are found promising. It would be interesting to develop a low-cost calibration-free gaze tracking system that could be sold on the mass market of console/computer for daily use, e.g. video game. We suggest that even gaze trackers with limited accuracy could be valuable when not relying solely on the video stream of the user. We believe that taking into account the image at which the user is looking, e.g. the visual feedback, is a promising way to increase the accuracy of any gaze tracking systems. To this aim, our method consisting in combining a visual attention model and a gaze tracking system is a first step and could be further investigated.

Toward gaze-aware VR applications and video games

Many 3D applications could benefit from being aware of user's attention. We could simply mention that they would be able to apply perception-based visual effects to dynamically adapt the rendering process to the user's attention. Among VR applications, video games applications could also benefit from being aware of gamers attention.

Nowadays, game developers are increasingly interested in emerging technologies to study gamers reactions to their games during play-tests. For instance, monitoring gamers physiological activities and states could be used to evaluate their state of mind and level of stress. This is currently studied in order to better balance the game difficulty and manage calm and stress moments as a pre-selling

tuning process [2]. Indeed, several features could be used like heart-rate, skin conductance, electro-encephalography and, in our case, gaze tracking [2]. We believe that visual attention monitoring could also be used in real-time. One possible feature of a gaze aware video game could be a real-time dynamic adaptation of the game difficulty. In the case the player is advancing fast, gaze information could be used to make the game harder by spawning hidden enemies, or hidden bonus items, in areas where he is not often gazing at. This would encourage concentration and exploration of the game VE. We believe that this feature could define a totally new area in the field of game design research such as for novel "attention-based visual effects", "gaze-based interaction protocols" or "gaze-based gameplay".

To conclude, a lot of work and user studies still need to be conducted in order to achieve the visions mentioned above. As visual attention modeling and gaze tracking are still very active research fields, there is no doubt that upcoming years will reveal several innovations in the way the gaze point is computed and in the way it is used to improve the feedback and interaction with virtual environments. We hope that the work presented in this manuscript will contribute to this exciting and promising adventure leading toward gaze aware applications.

Author's references

International conferences

Sébastien Hillaire, Anatole Lécuyer, Tony Regia Corte, Rémi Cozot and Gaspard Breton. A Real-Time Visual Attention Model for Predicting Gaze Point During First-Person Exploration of Virtual Environments, *Proceedings of ACM Symposium on Virtual Reality Software and Technology (VRST)*, Hong-Kong, China, pp.192-198, 2010.

Sébastien Hillaire, Anatole Lécuyer, Gaspard Breton and Tony Regia Corte. Gaze Behavior and Visual Attention Model When Turning in Virtual Environments, *Proceedings of ACM Symposium on Virtual Reality Software and Technology (VRST)*, Kyoto, Japan, pp.43-50, 2009.

Léo Terziman, Anatole Lécuyer, Sébastien Hillaire, Jan M. Wiener. Can Camera Motions Improve the Perception of Traveled Distance in Virtual Environment?, *Proceedings of IEEE Virtual Reality (VR)*, Short paper, Reno, Nevada, USA, pp. 131-134, 2009.

Sébastien Hillaire, Anatole Lécuyer, Rémi Cozot and Géry Casiez. Using an Eye-Tracking System to Improve Depth-of-Field Blur Effects and Camera Motions in Virtual Environments, *Proceedings of IEEE Virtual Reality (VR)*, Short paper, Reno, Nevada, USA, pp. 47-51, 2008.

Sébastien Hillaire, Anatole Lécuyer, Rémi Cozot and Géry Casiez. Depth-of-Field Blur Effects for First-Person Navigation in Virtual Environments, *Proceedings of ACM Symposium on Virtual Reality Software and Technology (VRST)*, Short paper, Newport Beach, California, USA, pp. 204-207, 2007.

International journals

Gabriel Cirio, Maud Marchal, Sébastien Hillaire, Anatole Lécuyer. Six Degrees-of-Freedom Haptic Interaction with Fluids, *To appear in IEEE Transaction on Visualization and Computer Graphics (TVCG)*, 2011.

Sébastien Hillaire, Gaspard Breton, Nizar Ouarti, Rémi Cozot and Anatole Lécuyer. Using a Visual Attention Model to Improve Gaze Tracking Systems in Interactive 3D Applications, *In Computer Graphics Forum*, Vol. 29, No. 6, pp.1830-1841, 2010.

Sébastien Hillaire, Anatole Lécuyer, Rémi Cozot and Géry Casiez. Depth-of-Field Blur Effects for First-Person Navigation in Virtual Environments, *In IEEE Computer Graphics and Application (CG&A)*, Vol. 28, No. 6, pp.47-55, Nov/Dec 2008.

National conferences

Sébastien Hillaire, Anatole Lécuyer, Rémi Cozot and Géry Casiez. Effets de flou visuel pour la navigation en environnements virtuels en vue à la première personnes, *Proceedings of Association Française d'Informatique Graphique (AFIG)*, Marne-la-Vallée, France, pp. 83-90, 2007.

Résumé long de la thèse en Français

Contribution à l'étude de l'utilisation des modèles d'attention visuelle et des systèmes de suivi du regard afin d'améliorer le retour visuel dans les applications 3D interactives

Introduction

Les activités de recherches conduites au cours de cette thèse s'inscrivent dans le cadre de la Réalité Virtuelle (RV). Une définition de la RV est donnée par Burdea et Coiffet [18] comme étant "*une interface homme-machine sophistiquée qui implique une simulation temps-réel et des interactions au travers de multiples canaux sensoriels. Ces modalités sensorielles sont visuelles, tactiles, auditives, olfactives ou gustatives*". Les chercheurs en RV sont toujours en quête de nouveaux canaux de communication qui pourraient améliorer l'interaction homme-machine. Plusieurs canaux d'entrée ont déjà été identifiés comme l'interaction gestuelle, la voix et plus récemment les interfaces cerveau-ordinateur [35]. Les systèmes de RV multimodaux peuvent utiliser plusieurs de ces canaux en parallèle et peuvent intégrer de multiples retours sensoriels comme, par exemple, sonore, visuel et haptique [62], afin d'améliorer l'expérience des utilisateurs dans l'EV. Aujourd'hui, la RV est utilisée pour de multiples applications telles que l'entrainement sportif, les visites architecturales, le prototypage virtuel industriel, etc. La RV peut être vue comme un outil permettant à des personnes de prototyper ou de s'entraîner avant une opération réelle. Pour aller encore plus loin, après quelques années de recherches, les interfaces et les méthodes d'interaction homme-machine développées pour la RV tendent à être disponibles sur le marché de consommation de masse tels que la Nintendo *Wiimote* ou Microsoft *Kinect*. La RV peut donc être vu comme un domaine expérimental d'avant-garde où les nouvelles interfaces et périphériques d'interaction sont conçus, testés et améliorés par les chercheurs et les industriels avant que tout un chacun puisse les utiliser quotidiennement.

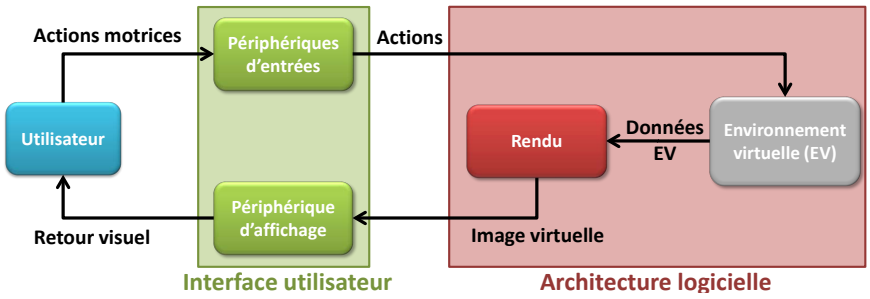


FIGURE 6.7 – A typical real-time 3D VR application.

Dans un dispositif typique de RV (voir Figure 6.7), un utilisateur interagit avec une application 3D temps-réel en utilisant des actions motrices sur les périphériques d'entées. Ensuite, ces actions sont interprétées par le moteur de simulation de l'application afin de modifier l'EV 3D. Finalement, une vue de l'EV est calculée et envoyée au périphérique d'affichage en tant que retour visuel à l'utilisateur, fermant ainsi la boucle d'interaction/visualisation.

Au cours de cette thèse, nous avons concentré nos travaux sur le retour sensoriel visuel : le lien entre les yeux de l'utilisateur et le périphérique d'affichage. De nos jours, ce canal est généralement utilisé

comme un lien à sens unique depuis le périphérique d'affichage vers l'utilisateur afin de fournir un retour visuel de l'EV, c.-à-d. une visualisation. Cependant, dans les applications de RV contemporaines ne tiennent géneralement pas compte de la perception et de l'attention humaine pour générer ce retour visuel. Ceci peut résulter en des scènes virtuelles qui n'appraissent pas naturelles. L'inconvénient de cette apparence non naturelle est qu'elle peut résulter en un faible sentiment d'implication et d'immersion des utilisateurs.

Dans cette thèse, nous proposons de considérer le canal sensoriel visuel comme un lien bidirectionnel. Dans ce cas, les applications de RV pourraient être informées de l'attention visuelle et/ou des points de focalisation des utilisateurs, c.-à-d. le point qu'ils regardent. Ainsi, au cours de la phase de rendu, il deviendrait possible d'appliquer des effets visuels basés sur la perception humaine et qui s'adapteraient à l'attention visuelle des utilisateurs afin d'améliorer le retour visuel.

Nos travaux de recherche sont organisés en deux axes de recherche majeurs correspondant à une décomposition entrée/sortie de l'application de RV :

1. Le premier axe de recherche est dédié aux entrées de l'application et à l'amélioration du suivi du regard des utilisateurs en EV 3D, en utilisant les propriétés de l'attention visuelle humaine et du comportement du regard humain. Cet axe représente le lien *depuis* les yeux de l'utilisateur *vers* le périphérique d'affichage, c.-à-d. l'attention visuelle humaine.

2. Le second axe de recherche est dédié aux sorties de l'application et est particulièrement orienté vers l'amélioration du retour visuel des EV 3D en s'inspirant de la perception et de l'attention visuelle humaine. Cet axe représente le lien *depuis* le périphérique d'affichage *vers* les yeux de l'utilisateur, c.-à-d. le retour visuel.

Le premier axe de recherche est dédié au développement de nouvelles méthodes pour l'amélioration du suivi de l'attention visuelle, ou du point de focalisation, des utilisateurs dans les applications 3D interactives. De nos jours, plusieurs solutions matérielles et logicielles existent afin d'estimer l'attention visuelle. D'un coté, les systèmes de suivi du regard peuvent être utilisés pour cette tâche. Cependant, ces solutions sont très souvent couteuses et intrusives, ainsi réservées pour des utilisations industrielles ou de recherche. De plus, chacun de ces systèmes possèdent une précision limitée due aux matériels et aux algorithmes de vision par ordinateur utilisés. D'un autre côté, les simulations de l'attention visuelle humaine, c.-à-d. les modèles d'attention visuelle, ont récemment émergé. Ces modèles sont conçus dans le but de calculer l'attention visuelle humaine sur une image ou dans une scène virtuelle en simulant plusieurs composantes attentionnelles et cognitives du système visuel humain. Un point de focalisation final peut ensuite être calculé en fonction de la distribution de cette attention visuelle estimée. Cependant, ces modèles ne peuvent calculer l'attention visuelle humaine qu'en tant que cas général et sont souvent limités par la simulation d'un faible nombre de composantes attentionnelles. De plus, peu de ces modèles ont réellement été conçus pour être utilisés dans notre contexte spécifique [73] : la navigation en environnement virtuel 3D avec une vue à la première personne. En conséquence, ce premier axe sera dédié à la recherche de nouvelles méthodes permettant de *calculer l'attention visuelle* des utilisateurs en utilisant un modèle d'attention visuelle *et/ou* un système de suivi du regard.

Le deuxième axe de recherche est dédié à la proposition d'une nouvelle utilisation de l'attention visuelle des utilisateurs : application d'effets visuels basés sur la perception humaine et adaptés à l'attention visuelle de l'utilisateur. L'attention humaine a déjà été utilisée pour de nombreuses finalités, mais, de nos jours, elle a seulement été exploitée pour deux buts principaux : l'interaction et le rendu basé perception. Concernant l'interaction, l'utilisation du point de focalisation a été désigné comme une donnée d'entrée pratique et naturelle [60]. Cependant, cette donnée doit être utilisée prudemment, sinon elle peut résulter en des actions non intentionnelles de la part des utilisateurs qui ne sont pas habitués à interagir en

utilisant les mouvements de leurs yeux. Concernant le rendu basé perception, le point de focalisation a été utilisé principalement afin de régler les niveaux de détail géométrique [78] ou d'éclairage des EV [87]. Ces méthodes diminuent la qualité du rendu dans les zones de l'image où l'utilisateur ne regarde pas. Ceci est possible car la précision de la perception humaine diminue en fonction de l'angle visuel par rapport à la direction principale du regard. Ceci permet d'accélérer le rendu des EV sans que l'utilisateur ne s'aperçoive de la baisse de qualité. En résumé, la plupart des méthodes basées perception existantes proposent de **simplifier** le processus de rendu. De manière complémentaire, nous proposons d'étudier comment la qualité d'une image d'un EV pourrait être **améliorée** en utilisant des algorithmes de rendu basés perception tenant compte de l'attention visuelle de l'utilisateur.

Cadre de recherche

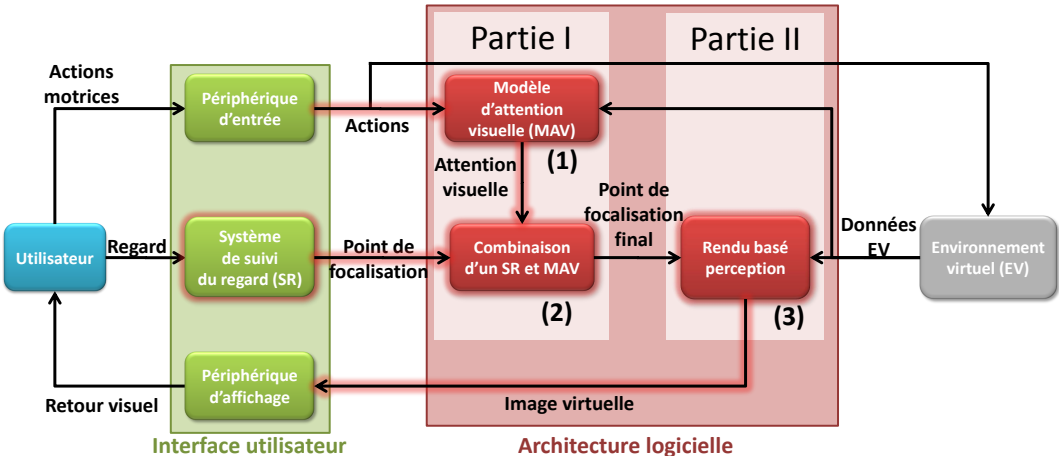


FIGURE 6.8 – Cadre de recherche. Nos principales contributions sont entourées en rouge.

Nous présentons notre cadre de recherche sur la Figure 6.8 où nos principales contributions ont été entourées en rouges. Comparée à la Figure 6.7, cette application possède maintenant deux entrées au niveau de l'interface utilisateur : les actions motrices et la position du point de focalisation. La couche d'entrée de l'application, correspondant à notre premier axe de recherche, est composée d'un modèle d'attention visuelle simulant l'attention visuelle humaine (1). Ce composant peut être utilisé seul pour calculer l'attention de l'utilisateur dans l'environnement virtuel, ou en combinaison avec un système de suivi du regard (2). Cette première couche envoie le point de focalisation calculé à la seconde et dernière couche. Cette dernière, correspondant à notre deuxième et dernier axe de recherche, a maintenant accès à l'attention visuelle de l'utilisateur. Ainsi, le processus de rendu est capable d'appliquer des effets visuels *basés perception* et adaptés en temps réel au point de focalisation de l'utilisateur (3). Le rendu visuel final de l'EV, la sortie de l'application, est finalement affichée sur le périphérique d'affichage en tant que retour visuel.

Nous décrivons maintenant les questions scientifiques des deux parties de ce manuscrit liées aux deux axes de recherche décrits précédemment.

Part 1 : Amélioration du suivi du regard dans les environnements virtuels 3D en utilisant l'attention visuelle et le comportement du regard humain

- Un nouveau model d'attention visuelle adapté à l'exploration des environnements virtuels 3D

Il a été démontré que les modèles d'attention visuelle étaient de bons prédicteurs de l'attention visuelle humaine sur des images et des vidéos [94]. Cependant, étonnamment, peu de modèles d'attention visuelle ont été proposés dans le but de calculer l'attention visuelle humaine durant l'exploration d'EV 3D. En effet, les modèles d'attention visuelle pourraient être une bonne alternative aux systèmes de suivi du regard matériel. A notre connaissance, seulement un modèle adapté à ce contexte a été proposé par Lee et al. [73]. Ce modèle simule plusieurs composantes attentionnelles de l'état de l'art ainsi que de nouvelles composantes adaptées à la navigation active en EV 3D. Ce modèle à été conçu afin de calculer un ensemble d'objets possiblement regardé par l'utilisateur. Tout d'abord, nous voulons améliorer ce modèle en proposant de nouvelles composantes attentionelles tenant compte des actions de l'utilisateur sur les périphériques d'entrée et du fait qu'il navigue dans l'EV en utilisant une vue à la première personne (Figure 6.8-1). Ensuite, nous souhaitons que notre nouveau modèle soit capable de calculer la position du point de focalisation à l'écran au lieu d'un ensemble d'objet possiblement observé par l'utilisateur. Ceci faciliterait l'utilisation des méthodes d'interaction et de rendu basées perception existantes.

- Une nouvelle approche pour améliorer la précision des systèmes de suivi du regard en utilisant les modèles d'attention visuelle

De nos jours, le point de focalisation de l'utilisateur peut être calculé par deux méthodes distinctes : soit par un système de suivi du regard *ou* par un modèle d'attention visuelle. Plusieurs systèmes de suivi du regard ont déjà été proposés : depuis les plus sophistiqués et précis [118] jusqu'aux systèmes de faible coût et peu précis basés webcam [7]. Afin de calculer la position du point de focalisation, ces systèmes s'appuient seulement sur un flux vidéo de l'utilisateur. Ainsi, typiquement, l'amélioration de la précision de ces systèmes est principalement effectuée par l'utilisation de matériel plus précis, ou par la recherche de nouveaux algorithmes de vision par ordinateur. Cependant, ces systèmes ont tout de même toujours eu une précision limitée. En conséquence, nous souhaitons améliorer la précision de n'importe quel système de suivi du regard en tenant compte de ce que l'utilisateur regarde et des propriétés du système visuel humain. Dans cette optique, nous voulons rechercher une méthode permettant de coupler n'importe quel système de suivi du regard avec un modèle d'attention visuelle (Figure 6.8-2). Nous voulons que cette méthode profite des forces de ces deux systèmes afin d'améliorer la précision et la fiabilité du point de focalisation estimé.

Part 2 : Amélioration du retour visuel des environnements virtuels 3D basée sur la perception humaine et l'attention visuelle de l'utilisateur

Dans les applications temps-réel, le point de focalisation de l'utilisateur a été jusqu'ici principalement utilisé dans le but d'améliorer l'interaction [111] ou pour modifier dynamiquement les niveaux de détails de l'EV 3D sans que l'utilisateur ne s'en aperçoive afin d'atteindre de meilleures performances [78] [87]. Le point de focalisation a déjà prouvé son utilité dans chacun de ces cas. Au lieu de simplifier l'EV, nous voulons proposer une nouvelle utilisation du point de focalisation : ajouter des effets visuels adaptés en temps-réel au point de focalisation et simulant des phénomènes visuels naturels présents dans la vision humaine (Figure 6.8-3). Nous voulons aussi étudier la préférence des utilisateurs concernant ces effets, ainsi que leurs bénéfices sur la perception des EV.

Partie 1 : Amélioration du suivi du regard dans les environnements virtuels 3D utilisant l'attention visuelle et le comportement du regard humain

Un nouveau model d'attention visuelle adapté à l'exploration des environnements virtuels 3D

Aujourd'hui, le suivi du regard dans les applications de RV est principalement effectué à l'aide d'un système de suivi du regard. Cependant, les systèmes précis sont coûteux et donc seulement utilisés par les chercheurs et industriels. Une alternative à ces systèmes serait d'utiliser des modèles d'attention visuelle simulant le système visuel humain.

Dans ce chapitre, nous proposons un nouveau modèle d'attention visuelle complet et temps-réel capable de calculer la position du point de focalisation d'un utilisateur en navigation avec une vue à la première personne dans un EV 3D. Dans ce contexte, comparé à l'existant [73], notre modèle est le premier capable de calculer une position de focalisation au lieu d'un ensemble d'objets visuels potentiellement regardé. Nous avons ensuite conduit une expérience afin d'évaluer les performances de ce modèle.

Description du modèle d'attention visuelle

Notre modèle est constitué de deux composantes attentionelles distinctes : *bottom-up* et *top-down*.

La composante bottom-up simule les réflexes du système visuel humain. Pour cela, nous calculons d'abord des cartes de caractéristiques par pixel de l'image perçue renseignant des informations tel que : l'intensité, les couleurs antagonistes, la profondeur et le mouvement. Un opérateur de différence centre-région-contournantes est appliqué sur ces images pour calculer des cartes de conspicuité simulant la réponse à ces stimuli des photo-récepteurs disposés sur la rétine. Finalement, toutes ces cartes de conspicuité sont intégrées et normalisées pour former la carte de saillance [57]. Plus le niveau de saillance d'un pixel est élevé, plus celui-ci a des chances d'attirer le regard.

La composante top-down simule le processus cognitif se déroulant dans le cerveau. Au lieu d'être calculée par pixel, cette composante calcule une valeur attentionnelle par objet visuel. Les précédents modèles utilisaient une représentation par maillage de ces objets visuels. Cependant, dans ce cas, il est impossible de prédire où le regard va porter : au milieu ou à l'extrémité d'un large mur ? Pour résoudre ce problème, nous proposons une nouvelle représentation des objets visuels basée sur les éléments de sur-

face, nommés *surfels*. Ainsi, chaque objet est virtuellement subdivisé en plusieurs petits élements. Pour chacun des ces élements de surface, nous calculons et combinons plusieurs valeurs attentionnelles : l'habituation (qui fait référence au fait que nous regardons moins souvent les objets que nous connaissons) et le contexte spatial (simulant le fait que nous regardons les objets desquels nous nous rapprochons). En plus de ces composantes calculées par surfel, nous calculons aussi deux valeurs attentionnelles dans l'espace écran : un poids plus important est donné au centre de l'écran, simulant le fait que l'attention des utilisateurs est plus souvent portée au centre de l'écran lors de navigation en vue à la première personne, et un poids attentionnel dépendant de la vitesse de rotation de la caméra simulant la nature prédictive du regard (par exemple : le fait que nous regardons dans la direction de la courbure du virage que nous abordons).

Finalement, les composantes top-down et bottom-up sont combinées en une seule afin d'estimer la distribution finale de l'attention visuelle de l'utilisateur sur l'écran. Le point de focalisation est positionné sur le pixel ayant la plus grande valeur attentionnelle. L'ensemble de ces calculs sont effectués sur carte graphique pour une exécution en parallèle efficace et rapide.

Expérience

Nous avons conduit une expérience afin d'évaluer les performances de notre modèle d'attention visuelle sur des utilisateurs navigant dans plusieurs EV 3D (voir Figure 6.9) de manière libre ou avec comme tâche de ramasser des objets. Nous comparons notre modèle au seul système existant proposé par Lee *et al.* [73].

FIGURE 6.9 – Capture d'écran des trois environnements virtuels utilisé dans l'expérience.

Nos résultats montrent que notre modèle possède de meilleures performances pour le calcul du point de focalisation (amélioration de plus de 100% des performances) et pour le calcul de l'objet regardé par

l'utilisateur à un instant donné. Nous avons aussi comparé les performances des composantes bottom-up et top-down séparément. Les résultats montrent que les performances de chacune de ces composantes séparées sont significativement inférieures aux performances du modèle complet, c.-à-d. lorsqu'elles sont combinées.

Ces résultats montrent que la simulation de l'attention visuelle humaine en utilisant un modèle d'attention visuelle pour calculer un point de focalisation est viable. De plus, notre expérience a mis en évidence le fait qu'il était important de combiner plusieurs composantes attentionelles adaptées au contexte d'utilisation afin d'améliorer les performances du suivi de regard. A la vue de ces résultats, nous pouvons en conclure qu'un modèle d'attention visuelle représente une alternative intéressante aux systèmes de suivi du regard.

Une nouvelle approche pour l'amélioration des systèmes de suivi du regard utilisant les modèles d'attention visuelle

Une multitude de systèmes de suivi du regard ont été proposés jusqu'à aujourd'hui. Chacun de ces modèles possède des caractéristiques qui lui sont propre (coût, précision, intrusif ou non, etc.). Cependant, aucun de ces systèmes n'est disponible sur le marché de masse public car trop couteux, requérant trop d'expertise ou n'étant pas assez précis. Il serait pourtant intéressant de proposer un système de suivi du regard grand public pour une utilisation quotidienne.

Un autre moyen de calculer l'attention visuelle de l'utilisateur est d'utiliser un modèle d'attention visuelle [73]. Cependant, contrairement aux systèmes de suivi du regard, ces modèles ne sont capables de simuler l'attention visuelle que globalement.

Dans ce chapitre, nous proposons une nouvelle méthode qui, pour la première fois, combine un système de suivi du regard et un modèle d'attention visuelle pour améliorer la performance globale du suivi du regard.

Système de suivi du regard basé réseaux de neurones articiciels

Dans un premier temps, nous avons développé un système de suivi du regard à bas-coût requérant seulement une webcam. Notre système permet de calculer la position du point de focalisation de l'utilisateur sur un écran en utilisant des réseaux de neurones artificiels.

Les images des yeux de l'utilisateur sont d'abord extraites de l'image de la webcam. Puis, elles sont transformées afin de calculer des histogrammes normalisés des luminances sur les axes X et Y de chaque image après une égalisation d'histogramme CLAHE. Ces histogrammes sont ensuite envoyés en entrée des réseaux de neurones lors de l'apprentissage, puis en temps réel, afin d'estimer la position du point de focalisation à l'écran.

Ce système de suivi du regard requière un niveau d'éclairage ambiant raisonnable et que la tête de l'utilisateur reste fixe. La précision du système a été calculée lors d'une expérience préliminaire à 1.4° horizontalement et 1.1° verticalement. Il est important de noter que ce système a été développé seulement afin de pouvoir le comparer à un système de suivi du regard très précis lorsqu'il est combiné, ou non, avec la méthode d'amélioration du suivi du regard que nous proposons.

Combinaison d'un système de suivi du regard et d'un modèle d'attention visuelle

Itti [55] a suggéré que la composante attentionnelle cognitive (top-down) serve à orienter le regard vers les zones d'intérêt, et qu'ensuite la composante bottom-up serve à affiner la direction du regard vers

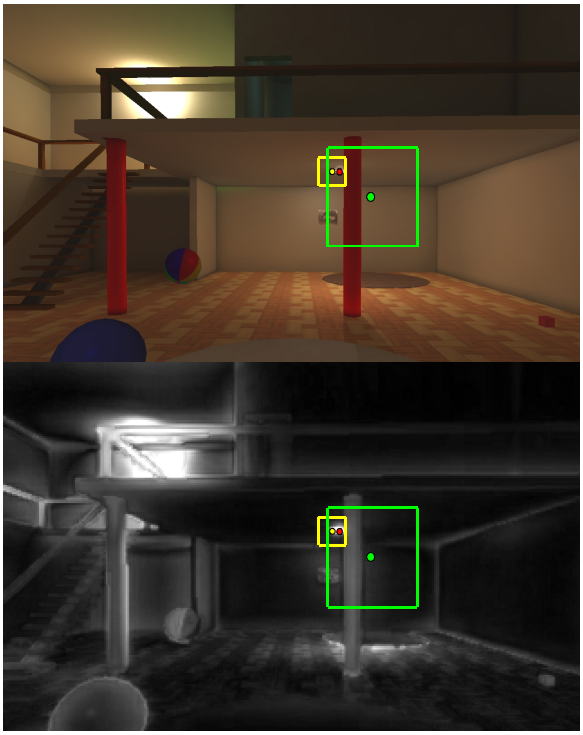


FIGURE 6.10 – Combinaison d'un système de suivi de regard à bas coût avec un modèle d'attention visuelle pour améliorer l'estimation du point de focalisation. Haut : vue de la scène. Bas : la carte de saillance correspondante. En jaune, le point de focalisation et la fenêtre d'incertitude du système de suivi du regard précis Tobii [118] (considéré comme la vérité terrain). En vert, le point de focalisation et la fenêtre d'incertitude du système de suivi du regard basé webcam. En rouge, le point de focalisation final calculé à partir de ce dernier système combiné avec un modèle d'attention visuelle.

une position saillante. En suivant cette idée, dans notre méthode, la direction du regard est calculée par le système de suivi du regard avec une certaine résolution/erreur : le point de focalisation est considéré comme étant à l'intérieur de la fenêtre d'incertitude correspondante. Les dimensions de cette fenêtre sont définies lors de l'apprentissage du système de suivi du regard. Ensuite, une carte de saillance mettant en évidence les zones pouvant attirer le regard est calculée à partir d'un modèle d'attention visuelle simulant seulement les reflexes visuels (bottom-up). Enfin, la position finale du point de focalisation est considérée comme la position la plus saillante à l'intérieur de la fenêtre d'incertitude.

De manière pratique, notre algorithme est constitué de trois phases principales. La première phase consiste à calculer la position du point de focalisation par le système de suivi du regard et la position de la fenêtre d'incertitude centrée sur ce point. La deuxième phase est composée du calcul de la carte de saillance metant en évidence les points d'intérêts de l'image. Dans notre cas, les primitives utilisées pour calculer la carte de saillance sont les suivantes : contours, intensités et couleurs antagonistes [57]. Finalement, la troisième et dernière phase consiste à rechercher la position la plus saillante à l'intérieur de la fenêtre d'incertitude. Le point de focalisation est finalement déplacé sur cette position (voir Figure 6.10).

Expérience

Nous avons conduit deux expériences afin d'évaluer l'amélioration des performances apportée par la combinaison de notre système de suivi du regard basé webcam avec un modèle d'attention visuelle.

La première expérience consistait en une navigation libre dans un environnement virtuel. Dix personnes ont participé à notre expérience. Au cours de la navigation, nous avons comparé les performances de notre système de suivi du regard basé webcam, combiné ou non avec un modèle d'attention visuelle, au système très précis Tobii considéré comme la vérité terrain. Nous avons conduit une analyse ANOVA sur nos résultats qui a révélé que notre système basé webcam possédait des performances significativement meilleures lorsqu'il était combiné avec un modèle d'attention visuelle. Les résultats révèlent aussi que notre système amélioré est aussi significativement meilleur que la carte de saillance seule. Cependant, en rejouant les sessions enregistrées, nous avons remarqué qu'il était impossible d'estimer la position du point de focalisation sur certaines positions non saillantes. Ceci pourrait être amélioré en intégrant des composantes top-down dans le modèle d'attention visuelle.

La deuxième expérience consistait en un mini jeu basé sur le regard. Dans celui-ci, des vaisseaux alliés et ennemis volaient dans des directions aléatoires sur l'écran. Il y avait 5 vaisseaux alliés, et 15 vaisseaux ennemis répartis équitablement sur trois distances, résultant en trois tailles différentes. Les vaisseaux étaient affichés sur un fond noir. Le but pour chacun des 10 participants, était de détruire du regard tout les vaisseaux ennemis sans détruire les vaisseaux alliés, un vaisseau étant détruit après une période de fixation de 600ms. Pour chaque participant, 6 sessions de jeux étaient exécutées et équitablement réparties sur l'utilisation des trois systèmes de suivi du regard : le système précis Tobii, notre système basé webcam et ce dernier amélioré avec notre méthode. Nos résultats ont montré que notre système de suivi du regard basé webcam combiné avec un modèle d'attention visuelle obtenait significativement de meilleurs résultats comparé au deux autres systèmes . Il était le seul permettant aux participants de détruire les vaisseaux ennemis les plus éloignés, c.-à-d. petits, grâce à l'utilisation de la carte de saillance. Par contre, c'est aussi le seul modèle qui a résulté en des vaisseaux alliés détruits. Cependant, ce nombre reste relativement faible (moins d'un vaisseau par participant). Si nous avions utilisé une image complexe au lieu d'un fond noir, nous aurions eu de moins bonnes performances. Cependant, l'ajout d'une composante top-down tenant compte de la tâche de l'utilisateur dans le modèle d'attention visuelle permettrait de rétablir les performances.

Pour résumé, notre approche consistant à combiner un modèle d'attention visuelle avec un système de suivi du regard est prometteuse et permet d'améliorer significativement les performances. Cette approche permet en plus de stabiliser la position du point de focalisation et, en quelque sorte, de compenser le lag dû à l'acquisition vidéo grâce à la fenêtre d'incertitude permettant une recherche large dans l'image. Finalement, notre méthode possède l'avantage de pouvoir être utilisée avec n'importe quel système de suivi du regard.

Partie 2 : Amélioration du retour visuel des environnements virtuels 3D basée sur l'attention visuelle de l'utilisateur

Améliorer le retour visuel en RV en simulant la perception visuelle humaine : effet de flou de profondeur

De nos jours, les scènes virtuelles peuvent être rendues en temps-réel avec une qualité proche de vraies photographies. Cependant, elles apparaissent souvent trop parfaites dû au manque d'effets visuels naturels présents dans le système visuel humain.

Dans ce chapitre, nous proposons un effet de flou de profondeur adapté à la navigation temps-réel en vue à la première personne dans des environnements virtuels. Nous avons ensuite conduit une expérience afin d'évaluer les effets de l'utilisation d'un tel flou de profondeur sur les performances et préférences utilisateurs.

Algorithme de flou de profondeur avec auto-focus

La première étape de notre algorithme est de calculer la vue courante de la scène ainsi que la carte de profondeur correspondante. Nous calculons aussi une carte contenant contenant la valeur sémantique des objets visibles à l'écran. La valeur sémantique représente l'importance des objets de la scène en fonction du contexte et de la tâche courante.

La deuxième étape consiste à calculer les paramètres de notre flou de profondeur. Pour cela, nous calculons tout d'abord la distance de focalisation en utilisant une zone d'autofocus positionnée au centre de l'écran. A la manière des appareils de photographie numérique, cette zone d'autofocus calcule la distance de focalisation comme étant la moyenne des profondeurs des pixels à l'intérieur et pondérée par la valeur sémantique des objets. Ainsi, même si un objet occupe peu de pixel, mais qu'il est considéré comme important (valeur sémantique forte), le focus sera effectué sur cet objet. Finalement, cette distance est filtrée par un filtre passe-bas afin de simuler le phénomène d'accommodation du système visuel humain. Afin de régler les dimensions de la zone d'autofocus, nous avons étudié la distribution du regard de personnes en navigation dans un environnement virtuel. Nous avons remarqué lors d'une expérience préliminaire que, lors d'une navigation en vue à la première personne, les utilisateurs regardaient très souvent au centre de l'écran. Au final, nous avons réglé les dimensions de la zone d'autofocus à 25% la taille de l'écran. Ainsi, en moyenne, 75% des points de focalisations des utilisateurs seront à l'intérieur de la zone d'autofocus. Cette taille, choisie arbitrairement, nous paraît être un bon compromit entre efficacité et coût de calcul.

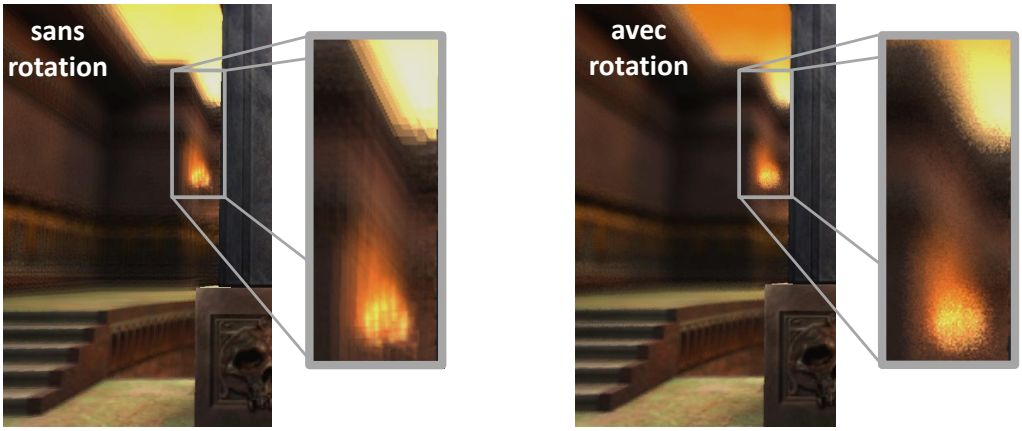


FIGURE 6.11 – Calcul du flou de profondeur sans et avec rotation du noyau d'échantillonage.

La dernière étape consiste à calculer l'image floutée finale en fonction des couleurs de la vue courante, des profondeurs et de la distance de focalisation. Pour cela, nous utilisons la méthode de regroupement proposé par Riguer *et al.* [101]. Nous avons amélioré cette méthode en appliquant une rotation aléatoire par pixel du noyau d'échantillonnage afin de remplacer les artefacts de bande de couleur par un bruit de haute fréquence automatiquement filtré par le système visuel humain (Figure 6.11).

Expérience

Nous avons conduit une expérience dans le but d'étudier l'influence de l'effet de flou de profondeur que nous proposons sur des joueurs lors de sessions d'un jeu de tir avec une vue à la première personne. Cette étude a d'abord révélé que seulement la moitié des participants avait apprécié l'effet de flou. Pour aller plus loin, elle a aussi révélé que les joueurs experts étaient significativement moins efficaces lorsque l'effet de flou était activé. Ceci pourrait être dû au fait que nous changions leurs habitudes qui est de régler les paramètres graphiques au minimum pour un maximum de performance. Nos résultats suggèrent tout de même qu'un tel effet de flou de profondeur est adapté pour les jeux vidéo et les applications de RV. Cependant, cet effet doit rester une option pouvant être désactivée.

Adaptation automatique du rendu visuel basé sur le point de focalisation : flou de profondeur et mouvement de caméra

Le point de focalisation est une information importante qui peut donner plusieurs avantages aux développeurs. Par exemple, il peut être utilisé pour accélérer le rendu des EV en modifiant dynamiquement les niveaux de détails des objets virtuels sans que l'utilisateur ne s'en rende compte [78][74]. D'une manière complémentaire à ces approches, nous proposons une nouvelle utilisation du point de focalisation dont le but est d'ajouter des effets visuels basés sur la perception humaine et adaptés en temps-réel à la position du point de focalisation. Le but de tels effets étant d'améliorer le retour visuel de l'EV aux utilisateurs.

Dans ce chapitre, nous proposons l'adaptation de deux techniques : un mouvement de caméra et un effet de flou de profondeur. Nous avons conduit une expérience afin d'étudier les préférences des participants concernant ces deux effets visuels en fonction qu'ils étaient calculés avec ou sans système de suivi du regard.

Effets visuels adaptés en temps-réel au point de focalisation

A chaque image calculée par notre application, nous demandons au système de suivi du regard la position du point de focalisation courante. Nous appliquons ensuite un filtre passe-bas à cette position afin de limiter les tremblements dus aux imprécisions du système. Nous utilisons ensuite notre système d'autofocus centré sur le point de focalisation afin de calculer la distance de focalisation dans l'EV. Finalement, connaissant la direction du regard dans l'EV (grâce au point de focalisation) et la distance de focalisation, nous pouvons calculer la position 3D du point de focalisation dans l'EV. La distance de focalisation et la position 3D du point de focalisation sont envoyées aux algorithmes de rendu des effets visuels.

Traditionnellement, les mouvements de marche en EV sont opérés par de simples translations, ou par l'application d'un décalage sinusoïdal au cours du temps. Cependant, ces méthodes ne permettent pas à l'utilisateur de ressentir la sensation de marcher ou résultent en un flux optique trop important. Lécuyer *et al.* [71] ont précédement proposé un mouvement de caméra avec un flux optique compensé seulement au centre de l'écran. Notre nouveau modèle de mouvement de caméra simule les réflexes vestibulo-oculaires agissant sur l'orientation des yeux au cours d'une marche réelle afin de stabiliser le flux optique au niveau du point de focalisation. Ainsi, notre modèle permet à l'utilisateur de fixer une cible sans qu'il ne soit perturbé par le flux optique, et tout en ayant la sensation de marcher dans l'EV.

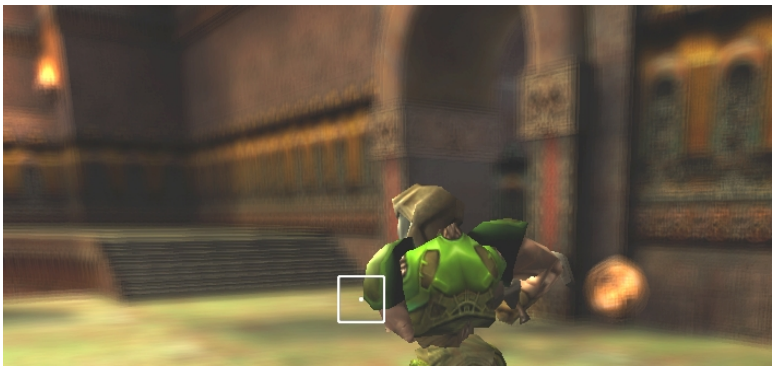


FIGURE 6.12 – Calcul du flou de profondeur en fonction du point de focalisation de l'utilisateur. Le point et le rectangle blanc représentent respectivement le point de focalisation et la zone d'autofocus.

Le flou de profondeur que nous proposons utilise le même algorithme tel que présenté dans la section précédente. La différence est que maintenant, la distance de focalisation est calculée au niveau du point de focalisation au lieu du centre de l'écran (Figure 6.12).

Expérience

Nous avons conduit une expérience afin d'étudier la préférence subjective des utilisateurs concernant les deux effets basés perception que nous proposons en fonction qu'ils étaient calculés avec ou sans suivi du regard.

Concernant les mouvements de caméra, les résultats montrent que le mouvement de caméra compensé a été significativement préféré lorsqu'il était calculé en fonction du point de focalisation de l'utilisateur. Il a aussi été préféré en termes de fun et de sensation d'immersion.

Les résultats ont aussi révélé que le flou de profondeur a été significativement préféré lorsqu'il était calculé en fonction du point de focalisation de l'utilisateur. Cet effet a plus particulièrement été préféré en terme de fun, d'immersion et de perception des profondeurs. Un participant a même reporté que

parfois il avait l'impression de regarder un écran auto-stéréoscopique. Ceci suggère que l'effet de flou de profondeur pourrait aider à véhiculer des indices visuels supplémentaires sur les profondeurs d'une scène comme le font les ombres. Ces résultats prometteurs suggèrent que l'utilisation d'effets visuels basés sur la perception humaine et adaptés en temps-réel au point de focalisation de l'utilisateur pourraient être utilisés dans les applications de RV et dans les jeux vidéo.

Conclusion

Dans ce manuscrit de thèse, nous avons étudié l'utilisation d'effets visuels basés perception et adaptés en temps-réel au point de focalisation de l'utilisateur. Ces effets visuels ont été conçus afin d'être utilisés dans les applications de RV impliquant une navigation avec une vue à la première personne. Ces recherches ont été menées au travers de deux axes principaux : 1) l'mélioration du suivi du regard dans les environnements virtuels 3D en utilisant l'attention visuelle et le comportement du regard humain, et 2) l'amélioration du retour visuel des environnements virtuels 3D basée sur l'attention visuelle de l'utilisateur.

Tout d'abord, nous avons proposé un nouveau modèle d'attention visuelle dont le but est de calculer automatiquement la position du point de focalisation d'un utilisateur naviguant dans un EV 3D. Ce modèle inclut une nouvelle représentation des objets visuels basée sur les éléments de surface. Cette nouvelle représentation est plus appropriée pour le calcul de certaines composantes attentionnelles telle que l'habituation. Ce modèle est aussi le premier ayant été conçu spécialement pour le contexte de la navigation en EV 3D avec une vue à la première personne et permettant de calculer un point de focalisation au lieu d'une liste d'objets visuels potentiellement regardés. Nous avons conduit une expérience qui a montré que notre modèle permettait de calculer une position de focalisation valide deux fois plus souvent que le seul modèle concurrent existant.

Pour conclure sur ce premier axe de recherche, nous avons finalement étudié la possibilité d'améliorer la précision des systèmes de suivi du regard en utilisant un modèle d'attention visuelle. Nous avons proposé une méthode permettant de combiner ces deux méthodes pour la première fois. Le système de suivi du regard est utilisé pour calculer une fenêtre d'incertitude dans laquelle l'utilisateur regarde et dont la taille est fonction de la précision du suivi. Ensuite, le modèle d'attention visuelle est utilisé pour rechercher la position attirant le plus le regard à l'intérieur de cette fenêtre pour définir le point de focalisation final. Les expériences que nous avons menées ont révélé une influence positive de notre méthode qui peut être utilisés avec n'importe quel système de suivi du regard.

Concernant le second et dernier axe de recherche, nous avons tout d'abord développé un algorithme de flou de profondeur s'adaptant automatiquement à la vue de l'environnement en utilisant un système d'autofocus similaire à celui des caméras digitales. Ce système tient compte des objets visibles à l'écran et du contexte. Nous avons ensuite étudié les répercussions de l'utilisation de cet effet de flou sur les performances et les préférences des joueurs lors de sessions d'un jeu de tir en vue à la première personne. Nos résultats ont révélé que la moitié des joueurs ont apprécié cet effet visuel et que seuls les joueurs experts obtenaient de plus faibles performances parce que l'on changeait leurs habitudes.

Finalement, nous avons proposé une nouvelle utilisation du point de focalisation de l'utilisateur : ajouter des effets visuels adaptés en temps-réel au point de focalisation et simulant des phénomènes visuels naturels présents dans la vision humaine. Pour cela, nous avons proposé deux effets visuels : 1)

un effet de flou de profondeur simulant le fait que le cristallin agit dans l'œil tel une lentille, et 2) un mouvement de caméra compensé permettant de simuler les reflexes vestibulo-oculaires compensant les mouvements de la tête lors de la marche afin de stabiliser le flux optique au niveau du point de focalisation. Nous avons conduit une expérience qui a révélé que ces effets visuels étaient significativement plus appréciés par les participants lorsqu'ils étaient calculés en fonction de leur attention visuelle. Plus particulièrement, ces effets ont été préférés en termes d'immersion et de perception des profondeurs. Nos résultats suggèrent que plusieurs autres effets visuels pourraient bénéficier d'une adaptation dynamique au point de focalisation de l'utilisateur, et que de tels effets pourraient être utilisés dans les applications de RV et les jeux video.

Tous ensemble, nos résultats suggèrent donc que les modèles d'attention visuelle peuvent être utilisés pour prédire le point de focalisation de l'utilisateur et pour améliorer la précision de n'importe quel système de suivi du regard existant. De plus, l'ajout d'effets visuels basés sur la perception humaine et dynamiquement adaptés au point de focalisation de l'utilisateur est une approche prometteuse qui peut être utilisée pour renvoyer à l'utilisateur des indices visuels et des sensations supplémentaires concernant l'environnement virtuel. Toutes les méthodes présentées dans cette thèse peuvent être utilisées conjointement dans le but d'améliorer la perception, l'immersion et les sensations des utilisateurs lors de navigation temps-réel en vue à la première personne dans des environnements virtuels 3D interactifs.

Glossary

VE : Virtual Environment
VR : Virtual Reality
MOT : Multiple Object Tracking (theory)
WTA : Winner Takes All (algorithm)
GPU : Graphics Processing Units
HLS : Hue-Luminance-Saturation
FoE : Focus Of Expansion (point)
WIMP : Windows, Icons, Menus Pointing device
LoD : Level-of-Detail
Pixel : Picture Element
Texel : Texture Element
Surfels : Surface Element
FPS : Frame-Per-Second or First-Person-Shooter
ANN : Artificial Neural Networks

Bibliography

[1] Wendy J. Adams and Pascal Mamassian. The effects of task and saliency on latencies for colour and motion processing. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 271(1535):139–146, 2004.

[2] Mike Ambinder. Valve's approach to playtesting: The application of empiricism. In *Game Developer's Conference*, 2009.

[3] Oscar Anson, Veronica Sundstedt, Diego Gutierrez, and Alan Chalmers. Efficient selective rendering of participating media. In *Proceedings of the 3rd symposium on Applied perception in graphics and visualization*, pages 135–142, 2006.

[4] Coltekin Arzu. *Foveation For 3D Vizualisation And Stereo Imaging*. PhD thesis, Helsinki University Of Technology, February 2006.

[5] David A. Atchinson and George Smith. *Optics of the Human Eye*. Elsevier Health Science, 2000.

[6] Olivier Aubault. *Visualisation interactive de scènes vastes et complexes à travers un réseau*. PhD thesis, University Of Rennes 1, 2006.

[7] Shumeet Baluja and Dean Pomerleau. Non-intrusive gaze tracking using artificial neural networks. Technical report, Carnegie Mellon University, 1994.

[8] Brian A. Barsky. Vision-realistic rendering: Simulation of the scanned foveal image from wavefront data of human subjects. *Proceedings of Symposium on Applied Perception in Graphics and Visualization*, pages 73–81, 2004.

[9] Ashweeni Kumar Beeharee, Adrian J. West, and Roger Hubbold. Visual attention based information culling for distributed virtual environments. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 213–222, 2003.

[10] Alain Berthoz. *The Brains Sense of Movement*. Harvard University Press, 2000.

[11] David Beymer and Myron Flickner. Eye gaze tracking using an active stereo head. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II – 451–8 vol.2, jun. 2003.

[12] Martin Bohme, Andre Meyer, Thomas Martinetz, and Erhardt Barth. Remote eye tracking: State of the art and directions for future development. *Proceedings of COGAIN: conference on Communication by Gaze Interaction*, 2006.

[13] Richard A. Bolt. Gaze-orchestrated dynamic windows. *Computer Graphics*, 15:109–119, 1981.

[14] Christophe Bordeux, Ronan Boulic, and Daniel Thalmann. An efficient and flexible perception pipeline for autonomous agents. In *Computer Graphics Forum (Eurographics '99)*, volume 18(3), pages 23–30. The Eurographics Association and Blackwell Publishers, 1999.

[15] Doug A. Bowman, Sabine Coquillart, Bernd Froehlich, Michitaka Hirose, Yoshifumi Kitamura, Kiyoshi Kiyokawa, and Wolfgang Stuerzlinger. 3d user interfaces: New directions and perspectives. *IEEE Computer Graphics and Applications*, 28(6):20–36, 2008.

[16] Marc S. Breedlove. *Biological Psychology*. Sinauer Associates, 2007.

[17] Julian P. Brooker and Paul M. Sharkey. Operator performance evaluation of controlled depth of field in a stereographically displayed virtual environment. *Stereoscopic Displays and Virtual Reality Systems VIII*, A. J. Woods, M. T. Bolas and J. O. Merritt (Eds), 4297(1):408–417, 2001.

[18] Grigore Burdea and Philippe Coiffet. *Virtual Reality Technology*. John Wiley and Sons, 2003.

[19] F.W. Campbell and R.W. Gubisch. Optical quality of the human eye. *Journal of Physiology*, page 186, 1966.

[20] Ran Carmi and Laurent Itti. Casual saliency effects during natural vision. *Proceedings of ACM Eye Tracking Research and Application*, pages 11–17, 2006.

[21] Kirsten Cater, Alan Chalmers, and Patrick Ledda. Selective quality rendering by exploiting human inattentional blindness: looking but not seeing. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 17–24, 2002.

[22] Kirsten Cater, Alan Chalmers, and Greg Ward. Detail to attention: exploiting visual tasks for selective rendering. *Proceedings of the 14th Eurographics workshop on Rendering*, pages 270–280, 2003.

[23] Alan Chalmers, Kurt Debattista, and dos Santos Luis Paulo. Selective rendering: computing only what you see. In *Proceedings of the 4th international conference on Computer graphics and interactive techniques in Australasia and Southeast Asia*, pages 9–18, 2006.

[24] Ming-Yuen Chan, Yingcai Wu, Wai-Ho Mak, Wei Chen, and Huamin Qu. Perception-based transparency optimization for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 15:1283–1290, 2009.

[25] Jean-Rémy Chardonnet, André De Carvalho Amaro, Jean-Claude Léon, Damien Huyghe, and Marie-Paule Cani. Designing and evolving hands-on interaction prototypes for virtual reality. In *Virtual Reality International Conference*, pages 25–34, 2010.

[26] Hao chen and Xinguo Liu. Lighting and material of halo 3. *Siggraph 2008 courses*, 2008.

[27] Li-Qun Chen, Xing Xie, Xin Fan, Wei-Ying Ma, Hong-Jiang Zhang, and He-Qin Zhou. A visual attention model for adapting images on small displays. *Multimedia Systems*, pages 353–364, 2003.

[28] Wen-Huang Cheng, Wei-Ta Chu, and Ja-Ling Wu. A visual attention based region-of-interest determination framework for video sequences. *Transactions on Information and Systems*, E88-D(7):1578–1586, 2005.

[29] International commission on illumination. Recommendations on uniform color spaces, color-difference equations, psychometric color terms. *Supplement No.2 to CIE publication No.15*, 1971.

[30] Nicolas Courty, Eric Marchand, and Bruno Arnaldi. A new application for saliency maps: Synthetic vision of autonomous actors. *Proceedings of IEEE International Conference on Image Processing*, 2003.

[31] Joe Demers. Depth of field: A survey of techniques. In *GPU Gems*, pages 375–390. Addison-Wesley Professional, 2004.

[32] Andrew T. Duchowski, Kevin Juang, Frank Jasen, Akshay Katrekar, and Joe Ahn. Use of eye movement gestures for web browsing. *Technical report, Departmentof Computer Science, Clemson University*, 2005.

[33] Andrew T. Duchowski, Eric Medlin, Anand Gramopadhye, Brian Melloy, and Santosh Nair. Binocular eye tracking in vr for visual inspection training. *Proceedings of ACM Eye Tracking Research and Application*, pages 89–96, 2002.

[34] James T. Enns. Three dimensional features that pop out in visual search. *Visual Search*, pages 37–45, 1990.

[35] Lotte Fabien. *Study of Electroencephalographic Signal Processing and Classification Techniques towards the use of Brain-Computer Interfaces in Virtual Reality Applications*. PhD thesis, PhD Thesis from the National Institute of Applied Sciences (INSA) Rennes, 2008.

[36] Masaki Fukuchi and Christof Koch. The focus of expansion acts as a cue for visual attention. *Journal of vision*, 8(6):880–880, 5 2008.

[37] Arne John Glenstrup and Theo Engell-Nielsen. Eye controlled media : Present and future state. Master thesis, University of Copenhaguen, 1995.

[38] Renato Grasso, Stefan Glasauer, Yasuhiko Takei, and Alain Berthoz. The predictive brain: anticipatory control of head direction for the steering of locomotion. *Neuroreport*, 26:1170–1174, 1996.

[39] Renato Grasso, Pascal Prevost, Yuri P. Ivanenko, and Alain Berthoz. Eye-head coordination for the steering of locomotion in humans: an anticipatory synergy. *Neuroscience Letters*, 253(2):115 – 118, 1998.

[40] Elias Daniel Guestrin and Moshe 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.

[41] Martin Hachet, Fabrice Decle, Sebastian Knödel, and Pascal Guitton. Navidget for 3d interaction: Camera positioning and further uses. *International Journal of Human-Computer Studies*, 67(3):225–236, 2009.

[42] Mary M. Hayhoe, Dana H. Ballard, Jochen Triesch, Hiroyuki Shinoda, Pilar Aivar, and Brian Sullivan. Vision in natural and virtual environments. In *Proceedings of the 2002 symposium on Eye tracking research & applications*, pages 7–13, 2002.

[43] Craig Hennessey, Borna Noureddin, and Peter Lawrence. A single camera eye-gaze tracking system with free head motion. *Proceedings of ACM symposium on Eye tracking research & applications*, pages 87–94, 2006.

[44] Sébastien Hillaire, Gaspard Breton, Nizar Ouarti, Rémi Cozot, and Anatole Lécuyer. Using a visual attention model to improve gaze tracking systems in interactive 3d applications. *Computer Graphics Forum*, 29:47–55, 2010.

[45] Sébastien Hillaire, Anatole Lécuyer, Gaspard Breton, and Tony Regia Corte. Gaze behavior and visual attention model when turning in virtual environments. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, pages 43–50, 2009.

[46] Sébastien Hillaire, Anatole Lécuyer, Rémi Cozot, and Géry Casiez. Depth-of-field blur effects for first-person navigation in virtual environments. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 203–206, 2007.

[47] Sébastien Hillaire, Anatole Lécuyer, Rémi Cozot, and Géry Casiez. Depth-of-field blur effects for first-person navigation in virtual environments. *IEEE Computer Graphics and Applications*, 28:47–55, November 2008.

[48] Sébastien Hillaire, Anatole Lécuyer, Rémi Cozot, and Géry Casiez. Using an eye-tracking system to improve camera motions and depth-of-field blur effects in virtual environments. In *Proceedings of the IEEE Virtual Reality conference*, pages 8–12, 2008.

[49] Sébastien Hillaire, Anatole Lécuyer, Tony Regia-Corte, Rémi Cozot, Jérome Royan, and Gaspard Breton. A real-time visual attention model for predicting gaze point during first-person exploration of virtual environments. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, pages 191–198, 2010.

[50] Yiqun Hu, Deepu Rajan, and Liang-Tien Chia. Detection of visual attention regions in images using robust subspace analysis. *Journal of Visual Communication and Image Representation*, 19(3), 2008.

[51] IdSoftware. Quake 3 arena. source code v1.32b under GPL, www.idsoftware.com, 2006.

[52] Takao Imai, Steven T. Moore, Theodore Raphan, and Bernard Cohen. Interaction of the body, head and eyes during walking and turning. *Experimental Brain Research*, 136:1–18, 2001.

[53] Bioware Inc. Neverwinter nights. www.bioware.com.

[54] Poika Isokoski. Text input methods for eye trackers using off-screen targets. In *Proceedings of Symposium on Eye Tracking Research and Applications*, pages 15–21, 2000.

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

[56] Laurent Itti, Nitin Dhavale, and Frederic Pighin. Realistic avatar eye and head animation using a neurobiological model of visual attention. In *Applications and Science of Neural Networks, Fuzzy Systems, and Evolutionary Computation VI*, pages 64–78. SPIE, 2004.

[57] Laurent Itti, Christopher Koch, and Ernst 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.

[58] Robert J. K. Jacob. The use of eye movements in human-computer interaction techniques : What you look at is what you get. *In ACM Transactions on Information Systems*, 9(3):152–169, 1991.

[59] Robert J. K. Jacob. Eye movement-based human-computer interaction techniques: Toward non-command interfaces. *Advances in Human-Computer Interaction*, 4:151–190, 1993.

[60] Robert J. K. Jacob. Eye tracking in advanced interface design. *In Virtual Environments and Advanced Interface Design*, pages 258–288, 1995.

[61] William James. The principles of psychology, 1890.

[62] Sreng Jean. *Contribution to the study of Visual, auditory and haptic rendering of information of contact in virtual environments*. PhD thesis, PhD Thesis from Institut National Des Sciences Appliquees (INSA) Rennes, 2008.

[63] Erika Jonsson. If looks could kill : An evaluation of eye tracking in computer games. Master's thesis, University of Stockholm, 2005.

[64] Michael Kass, Aaron Lefohn, and John Owens. Interactive depth of field using simulated diffusion on a GPU. Technical Report #06-01, Pixar Animation Studios, January 2006.

[65] Arie E. Kaufman, Amit Bandopadhay, and Bernard D. Shaviv. An eye tracking computer user interface. *Proceedings of IEEE Research Frontier in Virtual Reality Workshop*, 1993.

[66] Robert S. Kennedy and Norman E. Lane. Simulator sickness questionnaire: An enhanced method for quantifying simulator sickness an enhanced method for quantifying simulator sickness. *The international journal of aviation psychology*, 3(3):203–220, 1993.

[67] Alan Kenny, Hendrik Koesling, and Declan Delaney. A preliminary investigation into eye gaze data in a first person shooter game. In *Proceedings of European Conference on Modelling and Simulation*, pages 146–152, 2005.

[68] Raymond M. Klein. Inhibition of return. *Trends in Cognitive Sciences*, 4(4):138–47, 2000.

[69] Christopher Koch and S. Ullman. Shifts in selective visual attention: towards the underlying neural circuitry. *Hum Neurobiol*, 4(4):219–227, 1985.

[70] James J. Kuffner and Jean-Claude Latombe. Fast synthetic vision, memory, and learning models for virtual humans. In *Proceedings of the Computer Animation*, page 118, 1999.

[71] Anatole Lecuyer, Jean-Marie Burkhardt, Jean-Marie Henaff, and Stephane Donikian. Camera motions improve the sensation of walking in virtual environments. In *Proceedings of the IEEE conference on Virtual Reality*, pages 11–18, 2006.

[72] Chang Ha Lee, Amitabh Varshney, and David Jacobs. Mesh saliency. *Proceedings of ACM Transactions on Graphics*, pages 659–666, 2005.

[73] Sungkil Lee, Gerard J. Kim, and Seungmoon Choi. Real-time tracking of visually attended objects in virtual environments and its application to LOD. *In IEEE Transactions on Visualization and Computer Graphics*, 15(1):6–19, Jan.-Feb. 2009.

[74] Marc Levoy and Ross Whitaker. Gaze-directed volume rendering. In *Proceedings of the ACM symposium on Interactive 3D graphics*, pages 217–223, 1990.

[75] Huiying Liu, Shuqiang Jiang, Qingming Huang, Changsheng Xu, and Wen Gao. Region-based visual attention analysis with its application in image browsing on small displays. In *Proceedings of the 15th international conference on Multimedia*, pages 305–308, 2007.

[76] Margaret Livingstone and David Hubel. Segregation of Form, Color, Movement, and Depth: Anatomy, Physiology, and Perception. *Science*, 240:740–749, 1988.

[77] Peter Longhurst, Kurt Debattista, and Alan Chalmers. A GPU based saliency map for high-fidelity selective rendering. *Proceedings of the 4th international conference on Computer graphics, virtual reality, visualisation and interaction in Africa*, pages 21–29, 2006.

[78] David P. Luebke and Benjamin Hallen. Perceptually-driven simplification for interactive rendering. In *Proceedings of the 12th Eurographics Workshop on Rendering Techniques*, pages 223–234, 2001.

[79] Susana Mata, Luis Pastor, José Juan Aliaga, and Angel Rodríguez. Incorporating visual attention into mesh simplification techniques. In *Proceedings of the 4th symposium on Applied perception in graphics and visualization*, pages 134–134, 2007.

[80] Nelson L. Max and Douglas M. Lerner. A two-and-a-half-d motion-blur algorithm. In *Proceedings of ACM SIGGRAPH*, volume 19, pages 85–93, 1985.

[81] Daniel R. Mestre and G.S. Masson. Ocular responses to motion parallax stimuli: The role of perceptual and attentional factors - ii. physiology. *Vision Research*, 37:1627–1641, June 1997.

[82] Mathias Moehring, Antje Gloystein, and Ralf Doerner. Issues with virtual space perception within reaching distance: Mitigating adverse effects on applications using hmds in the automotive industry. *IEEE Virtual Reality Conference*, pages 223–226, 2009.

[83] Steven T. Moore, Eishi Hirasaki, Theodore Raphan, and Bernard Cohen. Effect of viewing distance on the generation of vertical eye movements during locomotion. *Experimental Brain Research*, 129:347–361, 1999.

[84] Steven T. Moore, Eishi Hirasaki, Theodore Raphan, and Bernard Cohen. The human vestibulo-ocular reflex during linear locomotion. *Neuroscience Letters*, 942:139–147, 2001.

[85] Carlos H. Morimoto and Marcio RM Mimica. Eye gaze tracking techniques for interactive applications. *In Computer Vision and Image Understanding*, 98(1):4–24, 2005.

[86] Michael C. Mozer, Michael Shettel, and Shaun Vecera. Top-down control of visual attention: A rational account. *Neural Information Processing Systems*, 2008.

[87] Karol Myszkowski. Perception-based global illumination, rendering, and animation techniques. In *Proceedings of the 18th spring conference on Computer graphics*, pages 13–24. ACM, 2002.

[88] Ken Nakayama and Gerald H. Silverman. Serial and parallel processing of visual feature conjunctions. *Nature*, 320:264–265, 1986.

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

[90] Vidhya Navalpakkam and Laurent Itti. Top-down attention selection is fine-grained. *Journal of Vision*, 6(11):1180–1193, Oct 2006.

[91] T. Niemann, M. Lappe, A. Büscher, and K. P. Hoffmann. Ocular responses to radial optic flow and single accelerated targets in humans. *Vision Research*, 39(7):1359 – 1371, 1999.

[92] Toshikazu Ohshima, Hiroyuki Yamamoto, and Hideyuki Tamura. Gaze-directed adaptive rendering for interacting with virtual space. In *VRAIS '96: Proceedings of the 1996 Virtual Reality Annual International Symposium (VRAIS 96)*, page 103, 1996.

[93] Christopher Peters and Carol O' Sullivan. Bottom-up visual attention for virtual human animation. In *Proceedings of the 16th International Conference on Computer Animation and Social Agents (CASA 2003)*, page 111, 2003.

[94] Robert J. Peters and Laurent Itti. Applying computational tools to predict gaze direction in interactive visual environments. *ACM Transaction on Applied Perception*, 5(2):Article 8, 2008.

[95] Frédéric Pighin, Jamie Hecker, Dani Lischinski, Richard Szeliski, and David H. Salesin. Synthesizing realistic facial expressions from photographs. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 75–84, 1998.

[96] Nischal M. Piratla and Anura P. Jayasumana. A neural network based real-time gaze tracker. *In Journal of Network and Computer Applications*, 25(3):179–196, 2002.

[97] Michael Potmesil and Indranil Chakravarty. A lens and aperture camera model for synthetic image generation. In *Proceedings of ACM SIGGRAPH*, volume 19, pages 298–306, 1981.

[98] Dale Purves. *Neuroscience*. Sinauer Associates, 2008.

[99] Martin Reddy. *Perceptually Modulated Level of Detail for Virtual Environments*. PhD thesis, University of Edinburgh, 1997.

[100] Erik Reinhard, Timo Kunkel, Yoann Marion, Jonathan Brouillat, Rémi Cozot, and Kadi Bouatouch. Image display algorithms for high and low dynamic range display devices. *Journal of the Society for Information Display*, 15(12):997–1014, December 2007.

[101] Guennadi Riguer, Natalya Tatarchuk, and John Isidoro. Real-time depth of field simulation. In Wordware, editor, *ShaderX2: Shader Programming Tips and Tricks with DirectX 9*, pages 529–556. Engel, Wolfgang, 2003.

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

[103] D. A. Robinson. The mechanics of human smooth pursuit eye movement. *Journal of Physiology*, 180:569–591, 1965.

[104] T.H. Rockwell. Eye-movement analysis of visual information acquisition in driving: an overview. *Proceedings of the 6th Conference of the Australian Road Research Board*, pages 316–331, 1972.

[105] Przemyslaw Rokita. Generating depth-of-field effects in virtual reality applications. *Computer Graphics and Applications, IEEE*, 16(2):18–21, 1996.

[106] Gilberto Rosado. Motion blur as a post-processing effect. *In GPU Gems 3*, pages 575–582, 2007.

[107] Dario D. Salvucci and John R. Anderson. Intelligent gaze-added interfaces. In *Proceedings of ACM Human Factors in Computing Systems Conference*, 2000.

[108] Christopher R. Sears and Zenon W. Pylyshyn. Multiple object tracking and attentional processing. *Journal of Experimental Psychology*, 54(1):1–14, 2000.

[109] John T. Serences and Steven Yantis. Selective visual attention and perceptual coherence. *Trends in Cognitive Sciences*, 10:38–45, 2006.

[110] M.I. Sereno, A.M. Dale, J.B. Reppas, K.K. Kwon, J.W. Belliveau, T.J. Brady, B.R. Rosen, and R.B. Tootell. Borders of multiple visual areas in humans revealed by functional magnetic resonance. *Science*, 268(5212):889–893, 1995.

[111] David J. Smith and Nicholas T. C. Graham. Use of eye movements for video game control. *Proceedings of ACM SIGCHI international conference on Advances in computer entertainment technology*, pages 20–28, 2006.

[112] T. Sousa. Crysis next gen effects. *Game Developer Conference*, 2008.

[113] Rainer Stiefelhagen, Jie Yang, and Alex Waibel. Estimating focus of attention based on gaze and sound. In *PUI '01: Proceedings of the 2001 workshop on Perceptive user interfaces*, pages 1–9. ACM, 2001.

[114] Yaoru Sun and Robert Fisher. Object-based visual attention for computer vision. *Artificial Intelligence*, 146(1):77–123, 2003.

[115] Veronica Sundstedt, Kurt Debattista, Peter Longhurst, Alan Chalmers, and Tom Troscianko. Visual attention for efficient high-fidelity graphics. In *Proceedings of the 21st spring conference on Computer graphics*, pages 169–175, 2005.

[116] Veronica Sundstedt, Efstathios Stavrakis, Michael Wimmer, and Erik Reinhard. A psychophysical study of fixation behavior in a computer game. In *Proceedings of the 5th symposium on Applied perception in graphics and visualization*, pages 43–50, 2008.

[117] Leo Terziman, Anatole Lecuyer, Sebastien Hillaire, and Jan M. Wiener. Can camera motions improve the perception of traveled distance in virtual environments? *IEEE Virtual Reality Conference*, pages 131 –134, 2009.

[118] Tobii. http://www.tobii.com. Last accessed May 2009.

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

[120] Yury Uralsky. Efficient soft-edged shadows using pixel shader branching. In Matt Pharr, editor, *GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purspose*, pages 269–282. Addison Wesley, 2005.

[121] Roel Vertegaal and David Fono. Eyewindows : Evaluation of eye-controlled zooming windows for focus selection. In *Proceedings of ACM Human Factors in Computing Systems Conference*, pages 151–160, 2005.

[122] Paul J. Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[123] Nathaniel Williams, David Luebke, Jonathan D. Cohen, Michael Kelley, and Brenden Schubert. Perceptually guided simplification of lit, textured meshes. In *Proceedings of the 2003 symposium on Interactive 3D graphics*, pages 113–121, 2003.

[124] Samuel J. Williamson and Herman Z. Cummins. *Light and Color in Nature and Art*. Wiley, 1983.

[125] J. M. Wolfe and G. Gancarz. Guided search 3.0: a model of visual search catches up with jay enoch 40 years later. In V. Lakshminarayanan, editor, *Basic and Clinical Applications of Vision Science*, pages 189–192. Dordrecht, Netherlands: Kluwer Academic Publishers, 1997.

[126] Hirotake Yamazoe, Akira Utsumi, Tomoko Yonezawa, and Shinji Abe. Remote gaze estimation with a single camera based on facial-feature tracking without special calibration actions. *Proceedings of ACM symposium on Eye tracking research & applications*, pages 245–250, 2008.

[127] Alfred L. Yarbus. *Eye motion and vision*. Plenum Press, 1965.

[128] Hector Yee, Sumanita Pattanaik, and Donald P. Greenberg. Spatiotemporal sensitivity and visual attention for efficient rendering of dynamic environments. *ACM transaction on graphics*, 20(1):39–65, 2001.

[129] Dong Hyun Yoo, Myung Jin Chung, Dan Byung Ju, and In Ho Choi. Non-intrusive eye gaze estimation using a projective invariant under head movement. *Proceedings of IEEE International Conference on Robotics and Automation*, 2006.

[130] Shumin Zhai, Carlos Morimoto, and Steven Ihde. Manual and gaze input cascaded (magic) pointing. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 246–253, 1999.

List of Figures

Résumé

En réalité virtuelle, l'interaction entre un homme et une machine peut être effectuée au travers de multiples canaux sensoriels. Le canal visuel est généralement utilisé afin de fournir à l'utilisateur une représentation de l'environnement virtuel avec lequel il interagit. L'objectif de cette thèse est d'améliorer ce retour visuel de manière interactive en tenant compte de l'attention visuelle de l'utilisateur.

La première partie de cette thèse est dédiée à l'évaluation de l'attention humaine dans des applications interactives. Nous avons cherché à calculer en temps réel le point de focalisation d'un utilisateur naviguant avec une vue à la première personne dans un environnement virtuel 3D. Pour cela, nous avons d'abord étudié l'attention visuelle humaine lors de navigation en environnements virtuels et avons montré qu'il existe de nombreux points communs avec la navigation en situation réelle. Nous avons ensuite proposé un modèle permettant de simuler ce comportement visuel. Puis, nous avons intégré ce composant dans un nouveau modèle d'attention visuelle complet permettant de prédire, en temps-réel, l'attention visuelle d'un utilisateur naviguant dans un environnement virtuel 3D. Nos évaluations ont montré que notre modèle est capable de prédire l'attention visuelle des utilisateurs de manière plus efficace que les modèles existants. Enfin, nous avons proposé une nouvelle utilisation des modèles d'attention visuelle dans le but d'améliorer la précision des systèmes de suivi du regard. Une étude a montré que cette approche pouvait améliorer la précision globale de ces systèmes.

Dans la deuxième partie de cette thèse, nous avons cherché à enrichir le retour visuel dans le but d'améliorer le sentiment d'immersion et la perception de l'environnement virtuel. Nous avons proposé une nouvelle utilisation du point de focalisation dans le but de simuler des effets visuels naturellement présents dans la vision humaine: un effet de flou de profondeur et un mouvement de caméra compensé. Une étude a montré que ces effets étaient fortement préférés par les utilisateurs lorsqu'ils étaient calculés en fonction de leur point de focalisation, et donc de manière plus interactive.

Abstract

In virtual reality, the interaction between a human and a computer can be achieved through multiple sensory channels. The visual channel is generally used in order to provide a visual feedback to the user interacting with a virtual environment. The goal of this thesis is to improve this visual feedback by taking into account user's visual attention in an interactive manner.

The first part of this thesis is dedicated to human attention. We wanted to evaluate in real-time the gaze point of a user navigating in a 3D virtual environment using a first-person view. We have first studied human visual attention when navigating in a virtual environment and have shown that there are several common behaviors when compared to a real pedestrian walk. We have then proposed a model in order to simulate this behavior. We have included this component in a novel, and global, visual attention model able to predict, in real-time, the attention of a user navigating in a virtual environment. An evaluation has shown that our model was able to predict the users' visual attention more efficiently than existing models. Finally, we have proposed a novel use of visual attention models in order to improve the accuracy of any gaze tracking systems. Our study has shown that our approach could improve the global accuracy of these systems.

In the second part of this thesis, we have studied new ways of improving the visual feedback in order to improve immersion feelings and perception of virtual environments. We have proposed a novel use of the gaze point in order to simulate natural visual effects present in human vision: a depth-of-field blur effect and a compensated camera motion. A study has shown that these effects were strongly preferred by participants when they were computed based on the gaze point, in a more interactive manner.

N° d'ordre : **D11-02**

Institut National des Sciences Appliquées de Rennes

20, Avenue des Buttes de Coësmes . CS 14315 . F-35043 Rennes Cedex

Tel : 02 23 23 82 00 – Fax : 02 23 23 83 96