An empirical pipeline to derive gaze prediction heuristics for 3D action games

MATTHIAS BERNHARD
Technical University of Vienna
and
EFSTATHIOS STAVRAKIS
INRIA/REVES
and
MICHAEL WIMMER
Technical University of Vienna

Gaze analysis and prediction in interactive virtual environments, such as games, is a challenging topic, since the 3D perspective and variations of the view point, as well as the current task, introduce many variables that affect the distribution of gaze. In this article, we present a novel pipeline to study eye-tracking data acquired from interactive 3D applications. The result of the pipeline is an importance map which scores the amount of gaze spent on each object. This importance map is then used as a heuristic to predict a user's visual attention according to the object properties present at runtime. The novelty of this approach is that the analysis is performed in object space and the importance map is defined in the feature space of *high-level properties*. High-level properties are used to encode task relevance and further attributes, like e.g. eccentricity, which may have an impact on gaze behavior.

The pipeline has been tested with an exemplary study on a first-person shooter game. In particular, a protocol is presented describing the data acquisition procedure, the learning of different importance maps from the data and finally an evaluation of the performance of the derived gaze predictors. A metric measuring the degree of correlation between attention predicted by the importance map and the actual gaze yielded clearly positive results. The correlation becomes particularly strong when the player is attentive to an in-game task.

Categories and Subject Descriptors: H.1.2 [**Models and Principles**]: User/Machine Systems—*Human Factors*; I.3.3 [**Computer Graphics**]: Picture/Image Generation—*Viewing Algorithms*

General Terms: Algorithms,Experimentation,Human Factors

Additional Key Words and Phrases: gaze analysis, eye-tracking, visual attention, high-level properties, importance map, gaze predictor, video games, virtual environments

1. INTRODUCTION

While attention is one of the most intensely studied topics within psychology and cognitive neuroscience, it has also raised increasing interest in the field of computer science, especially among those disciplines dealing with imaging technologies, human-computer interaction or computer vision. Knowing and being able to predict which stimuli are attended to (and which are not) may be used to improve media technologies and interface

design for software applications, while computer vision systems can use this knowledge to imitate the mechanics of human visual perception. The common model for visual attention is two-tiered: on the one hand, there are bottom-up processes which react to low-level features present in the stimuli; on the other hand, top-down processes find and select the targets of interest due to the viewer's goals [James 1890].

Inspired by Treisman's feature integration theory [Treisman and Gelade 1980], algorithms to compute so-called *saliency maps* (e.g. [Itti et al. 1998]), which score the conspicuity of image regions due to low-level features, have become popular. These methods introduced the first meaningful heuristics to predict features of the stimuli which are likely to receive more attention than others. Algorithms that compute saliency maps require only digital images as input and, therefore, they are unaware of high-level processes involved in the selection of the viewer's targets. However, in interactive applications, such as 3D computer games, which are inherently task-oriented, viewing behavior of a user can better be reflected by gaze prediction models sensitive to top-down processes. For instance, as shown in figure 1, our predictors revealed a high importance for columns or doors, whereas bottom-up predictors can neither identify objects nor predict that columns and doors are important for the player when he navigates through the level of a game.

While bottom-up processes can be modeled and thus predicted according to a well studied theory independent of a particular application, top-down processes rely on many factors that cannot easily be generalized, like object semantics, viewer task etc., and are very application specific. In this work, we therefore propose to *learn* top-down attention processes by observing actual users interacting with an application through eye-tracking, in order to be able to *predict* attention for other users without having to eye track them. In contrast to bottom-up methods, which are inherently image-based, top-down processes work at the object level [Duncan 1984; O'Craven et al. 1999; Scholl 2001] and therefore our system needs to work with objects, not pixels. In a static scene, counting the number of fixations for each object would give a plausible estimate for the importance (i.e., likelihood of being attended) of each object. However, the major challenge in learning attention for an interactive application like a game is that the environment is *dynamic*, and thus the set of potential fixation targets is constantly changing due to camera and object movement. In this paper, we propose the use of *semantic properties* of the objects (such as category) and the environment (such as player task) as a basis for attention prediction, because such high-level properties are more insensitive to dynamic changes in the scene.

The main contribution of this paper is a novel pipeline which enables the empirical analysis of gaze behavior in interactive 3D application by taking into consideration semantic information of the game objects and the player. The output of this pipeline is an importance map which is learned from eye-tracking data and can be used to predict the likelihood of objects in the stimuli to be attended. These importance maps may find utility in perceptually optimized rendering algorithms, automated in-game advertising, or may assist game designers in their task.

As an example of the effect of using high-level semantic properties, consider that a change in a player property can infer changes in the task, which in turn changes the relevance of certain objects. For instance, when the player of a shooter is unarmed, one of his primary tasks is to search for a weapon, whereas as soon as he has found one, he has a different

(a) (b) (c)

Fig. 1: (a) An example reconstructed framebuffer of the game is overlaid with the visualization of a fixation in the current frame. In Figure (b), importance values predicted with our method, for each object in the scene, are used to visually highlight objects in the framebuffer (brighter objects are more important than darker ones). Figure (c) shows the corresponding saliency map. Since our method is sensitive to semantic properties and common gaze behavior, it can predict better the high importance of doors or objects in the center, while saliency maps are less selective and predict the importance of pixels rather than objects. We discovered for instance that none of the lights mounted on the ceiling were attended, although being indeed salient.

task and other objects are of primary interest. Note that the use of properties requires access to the internal structures of an application. As we do not analyze the distribution of gaze on individual objects and instead abstract them by their properties, we can increase the density of the distribution being inferred, since several objects may share the same properties. Ideally, this strategy allows us to predict the importance of objects in scenes which were never included in the gaze analysis (e.g. of different levels or other players).

2. BACKGROUND

2.1 Visual attention

Since the information processing capacity of our brain is limited, incoming information has to be filtered so that we are able to process the most important sensory inputs. Visual attention is the control mechanism which selects meaningful inputs and suppresses those of low importance. Moreover, our eyes can sense image details only in a 2° foveal region, due to a rapid falloff of spatial acuity towards the periphery of the fovea. Although we have a coherent impression of our field of view, we perceive only a fraction of the detail that is actually present. Full information about an object is only available in the narrow field of the current foveal focus [Henderson et al. 2003].

A vast array of experiments trying to carve out the mechanisms of attention can be found in cognitive psychology and neuroscience literature. A concise overview of important studies on visual attention can be found in [Wolfe 2000]. Most established is a model which divides attention into bottom-up and top-down processes [James 1890]. While the former account for pre-attentive processes, which are considered to be under unconscious control and driven by low-level features, the latter refer to directed voluntary control of attention involving high-level processes such as thought, reasoning and memory [Palmer 1999]. However, bottom-up and top-down processes cannot be separated perfectly and there is much feedback between both [van Zoest and Donk 2004; Wolfe 2007].

Due to this model, low-level features which are likely to be perceptually important tend to

attract attention, which is needed to integrate them ([Treisman and Gelade 1980]) and map them to higher level representations. Meanwhile, cognitive processes consciously direct attention towards those regions in the field of view being of importance according to the intentions of the viewer [Yarbus 1967]. Established theories about visual search assume that low-level features (e.g. color or intensity) characteristic for target objects are enhanced and guide the search [Wolfe 1994]. A similar intuitive interpretation is that top-down control raises the saliency of important objects [Oliva et al. 2003; Navalpakkam and Itti 2005; Elazary and Itti 2008].However, though there are reasonable theories for top-down mechanisms concerning visual search tasks, they do not directly explain how attention is deployed in complex and changing tasks as occuring in computer games. Eye-tracking studies are a reasonable method to investigate top-down attention from the opposite perspective, i.e., analysing how visual attention behaves under particular complex stimuli and tasks.

2.2 Eye-tracking

Since application-oriented research is interested in *where* attention is actually directed to, eye-tracking technology has been the method of choice to obtain an instant estimation of the attentional focus. Apart from estimating the position of the foveal focus, various other features useful for analysis, such as fixation duration, fixation counts or amplitudes of saccades, can be extracted from eye-tracking data [Duchowski 2003]. Notably fixation duration, which we use to weigh fixation counts, is suggested to be a good indicator to estimate how strongly cognitive functions, such as object identification (e.g. [De Graef et al. 1990]), memory (e.g. [Henderson et al. 1999]) and monitoring of task-relevant objects (e.g. [Land et al. 1999]), are involved. The relationship between human gaze control and cognitive behavior in real-world scene perception is reviewed in [Henderson 2003].

Eye-tracking is often used under the assumption that there is a strong correlation between the focus of gaze and the actual focus of visual attention. Indeed it is possible to focus mentally on stimuli outside the foveal region. In this case the internal visual attention system (*covert* visual attention) is focused on a particular place, whereas eye-movements (*overt* visual attention) are directed to other places. The other special case is the phenomenon of *inattentional blindness* [Mack and Rock 1998], which was observed when viewers were concentrated on a task. With the intensity of the task, the probability that they may not notice details irrelevant to the task, even within their foveal focus, increases [Simons and Chabris 1999]. However, overt and covert attention have a strong natural relationship and by tracking eye movements, the local focus of visual attention can be estimated well in most cases.

For many applications, such as rendering 3D environments, the prediction of overt attention may be sufficient to perceptually optimize rendering of specific objects, since regions outside the fovea are not perceived in high detail. For example, [Luebke et al. 2000; Murphy and Duchowski 2001] demonstrated that geometric detail in the periphery of the visual focus can be reduced without decreasing the perceived rendering quality by using an eye-tracker for gaze-contingent rendering optimizations. [Komogortsev and Khan 2006] attempted to predict the visual focus of multiple eye-tracked viewers in order to perform perceptually optimized video and 3D stream compression.

Gaze behavior was also studied when certain tasks had to be carried out. To analyze gaze

behavior in natural tasks, several studies were conducted with easy tasks ranging from hand washing to sandwich making [Hayhoe et al. 2003; Canosa et al. 2003; Pelz and Canosa 2001]. Virtual environments [Rothkopf et al. 2007] or computer games [Kenny et al. 2005; El-Nasr and Yan 2006; Jie and Clark 2007; Sundstedt et al. 2008] recently became interesting stimuli for eye-tracking experiments as well. All these studies support the hypothesis that in conditions where a task has to be carried out, gaze behavior is mainly dominated by task relevance rather than salient features in the stimuli, as task-relevant objects are continuously monitored by the visual system [Land et al. 1999]. Note that once a target is found and monitored during a task, the models for top-down control from visual search are not appropriate anymore.

[Starker and Bolt 1990] proposed to use an eye-tracker to guide synthesis of speech in a way that narration refers to the current object of the user's interest. Although eye-tracking is used for real-time user-to-system feedback, their models of interest map gaze to objects, and successively the user's level of interest for each object is inferred. This resembles our methodology of inferring objects' importance by mapping eye-tracking data to semantic properties.

2.3 Heuristics to predict visual attention

In the absence of an eye-tracker in the final application setup, the focus of attention may be estimated by a predictor algorithm, while eye-tracking is only used to infer the predictor in the first place and to evaluate the performance of predictor heuristics [Marmitt and Duchowski 2002; Peters and Itti 2008]. Most visual attention or scanpath predictors rely on low-level features and thus account for bottom-up attention only. A first biologically inspired computational model for bottom-up attention was proposed in [Koch and Ullman 1985] and was further developed in [Itti et al. 1998; Itti and Koch 2001; Parkhurst et al. 2002]. This method creates feature maps of several low-level features, notably the contrast in color, intensity and orientation, which are combined into a *saliency map* which scores each pixel with one scalar value. Different features can be processed in independent channels following Treisman's theory of feature integration [Treisman and Gelade 1980]. For non-static stimuli such as videos or computer games, it is possible to improve the performance of saliency maps somewhat by introducing temporal features [Itti and Baldi 2006; Peters and Itti 2008], notably motion, flicker or Bayesian surprise. Evaluations on various commercial games [Peters and Itti 2008] showed that even color has a non-negligible contribution, which can be explained by the fact that game designers prefer to texture task-relevant objects with salient colors.

The advantage of saliency map algorithms is that they work on any sequence of digital 2D images, giving per-pixel scores of bottom-up attention. Since they are computed from low-level features only, they perform with greatest reliability in free viewing conditions. On the other hand, performance of bottom-up predictors in interactive environments is limited, as in the presence of a task the majority of gaze targets are task-relevant objects [Rothkopf et al. 2007; Pelz and Canosa 2001]. This holds even if the actions are automated and do not need conscious attention [Land et al. 1999].

Games, in particular, rarely exhibit a free-viewing behavior; even in passive viewing conditions the observer tends to assume a task [Sundstedt et al. 2008]. In order to take this

into account, the idea of selective rendering was proposed [Cater et al. 2002; Cater et al. 2003; Sundstedt et al. 2007]. The idea is to exploit inattentional blindness. By means of a so-called *task map*[Cater et al. 2002] or *importance map*[Sundstedt et al. 2005], the rendering quality of task relevant regions is selectively enhanced. It has been successfully demonstrated that selectively rendered images are not perceived differently to high-quality images, but building task maps (e.g. by manual assigment) was rather simple in the non-interactive examples being studied.

A common trend is to use saliency maps and top-down heuristics in conjunction, for example by biasing saliency values at locations which are suggested to be task-relevant (e.g. [Sundstedt et al. 2005]). Attempts to simulate top-down attention with neural networks [Peters and Itti 2007] or object detection algorithms [Canosa et al. 2003] demonstrated significant improvements over the use of saliency maps alone. [Cerf et al. 2008] used a face recognition algorithm as high-level entity detector, while other important entities were outlined manually with a minimal region of interest. In this way, saliency maps of static images can be biased according to high-level content. Since in VEs an object-based predictor can be more useful (e.g. to switch levels of detail of 3D meshes), [Lee et al. 2007] implemented a framework where an object-based saliency map is computed by mapping pixel-based saliency maps to objects using an item-buffer. Moreover, they use the fact that objects receive more attention than the background (e.g. [Einhäuser et al. 2008]) as a heuristic to account for top-down factors.

An interesting approach was presented in [Navalpakkam and Itti 2005], where a saliency-based search algorithm [Itti and Koch 2000] was extended to perform a task-driven search. An object classification algorithm, which was trained to associate low-level features with object classes, increases the saliency of those image regions that are likely to belong to an object with task-relevant properties. They proposed an ontological model which uses a task graph to capture semantic relations between symbolic classes of objects, their super classes and the task. This model sounds promising, but it was actually designed for visual search tasks in machine vision systems.

In many cases, it is possible to access to the internals of an application. This makes it possible to avoid machine-vision approaches, since task-relevant objects can be determined from the scene graph in a virtual environment, or the object hierarchy in a game. For example, [Sundstedt et al. 2008] carried out an eye-tracking study to generate an importance map based on high-level properties in a computer game. The task of the game was to navigate a small ball through a maze, which was in 3D, but rendered from a fixed bird's-eye view. All items in the maze were tagged with high-level properties, like for instance "correct path", to encode the relevance of certain parts of the maze according to the task of finding the maze exit. They found that the gaze distribution inferred over those properties, correlated considerably among the different participants of the study. However, this study reduced the problem of inferring gaze distributions to a very limited case by assuming a fixed camera and a constant set of objects. In this paper we take a major step further and generalize this approach to a representative 3D scenario with a dynamic viewpoint and a field-of-view with variable content.

3. OVERVIEW

This paper presents a pipeline to analyze gaze data in order to provide an attention predictor for dynamic scenes that works on objects instead of on pixels. We assume that we have access to the internal object hierarchy (scenegraph) of the application. Figure 2 shows the stages of the pipeline: First, we gather data in an *eye-tracking session*, where we record the player's gaze together with the changes to the scenegraph. In the *data preparation* step (Section 4), we reconstruct the scenegraph for each frame and correlate it with the gaze data to find fixated objects. We also extract properties associated with these objects or the game state. In a subsequent *analysis* phase, this data is used to learn importance maps, which assign importance values to each extracted property. In order to predict the attention of a user at runtime, each object is assigned an importance value that is derived from the importance values of the properties it is associated with.

Note that importance map is assigned to properties and not to objects. This allows capturing changes to the scenegraph (an object might turn from a friend into an enemy) and to analyze similarities between objects on a semantic level. However, in practice we can only observe combinations of properties through eye-tracking, and therefore the number of properties used for deriving importance values needs to be controlled in order to obtain statistically relevant estimates. Thus, we introduce an additional step (Section 5.1) where the user of the system defines simple Boolean *high-level properties*. Using a simple scripting language, the user specifies rules how certain properties shall be "interpreted" and selects the high-level properties to be used as the basis of the importance map. In this stage the user of the system intervenes to control the analysis process, allowing him to experiment with various kinds of properties to find a mapping that best encodes the peculiarities of the application (e.g. different tasks and interaction interfaces).

One essential part of the learning process is to provide the user with feedback about the performance of the predictors he has generated. The learning process can be iterated until satisfactory results are achieved. We propose two evaluation strategies (Section 6), which measure the correlation of the predictions with the actual gaze of subjects that were not included in the learning stage of the predictors.

While the theoretical concepts of our algorithms are presented in Sections 4 - 6, we also demonstrate the complete pipeline by means of an exemplary study carried out with a first person shooter 3D game (Section 7). In particular, we describe our experimental setup, the game's design considerations, the data acquisition and analysis stages, and the evaluation of the generated prediction heuristics.

4. DATA PREPARATION

Our pipeline starts with an eye-tracking session. For this, we describe a particular example in Section 7), since the way this session is carried out differs from application to application. The result of the eye-tracking session is *stimulus data* in the form of a replay file which allows reconstructing the complete application state at every frame, and *gaze data* in the form of a gaze data file recorded by the eye-tracking software.

The goal of the *data preparation* step is then to match stimulus data with gaze data and transform both into an abstract form that can be used by the analysis tool to learn gaze

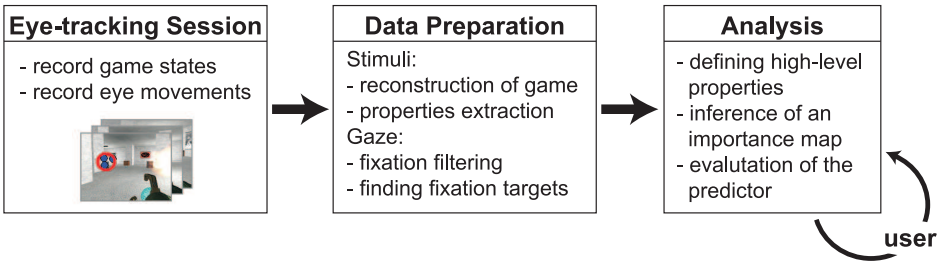


Fig. 2: Overview of our pipeline. The eye-tracking and data preparation steps may be performed once, then the user can design and generate different gaze predictors.

predictors. This makes the analysis step (Section 5) independent of the application and the eye-tracking software and allows rerunning the analysis with different parameters without having to redo the expensive application-specific steps. Data preparation includes reconstruction of the stimuli presented to the player, extraction of the scene graph's properties, and processing of the gaze data. The most expensive step is to determine those objects which were gaze targets.

4.1 Processing stimuli data

4.1.1 *Reconstruction.* Since in this work we venture into the area of object-based predictors, we require access to the internals of the application. In our proof-of-concept experiment, we utilize a game engine developed in our lab, although any other engine, commercial or open-source, could be used. Hence, we can record the object-oriented scene graph instead of capturing the rendered frames with a screen-recording tool (as has to be done for e.g. commercial games [Peters and Itti 2008]). Apart from the significantly decreased requirements for data storage, this method provides us the possibility to fully reconstruct the scene graph with all its properties, for every frame. To this end, we modified the game engine so that it records changes to the scene graph and the camera during runtime. To guarantee sufficient frame rates throughout the experiment, analysis is performed separately using a replay tool which reconstructs the game state at every frame. Another requirement is a reliable synchronization of the recorded frames with the eye-tracking data. This can be achieved by operating the eye-tracker via the game application, by initializing and starting the eye-tracker and the replay simultaneously, so that both have the same temporal starting point.

4.1.2 *Extracting object properties.* During the replay, we need to extract those properties from the scene graph that potentially have an impact on gaze behavior. For each frame, we extract the properties of all elements visible in the player's field-of-view. Exact visibility is determined using the item buffer described in Section 4.2.2. Further, we also store the properties of the player, required to infer the current task the player is performing, and general environment properties. The selection and interpretation of relevant properties is then carried out during the analysis stage and it is controlled by the user via a scripting interface, described in Section 5.1. Another purpose of this step is to normalize numerical properties (Section A.2) for intuitive use.

In our experimental study, it turned out that the strongest properties were the semantic category of an object (which can be easily inferred from the entity name given a consistent labelling of objects) and its screen-space bounding window, which provides information about size and pose.

4.2 Processing gaze data

4.2.1 *Fixation filtering.* The eye-tracker outputs an array of raw gaze points, which are defined by a timestamp and a 2D position on the screen. From this we need to determine the actual foci of user attention. In general, human gaze alternates between *fixations*, where gaze rests on one location, and *saccades*, fast transitions between fixations. Since we base our studies on measuring overt attention, we assume user attention correlates with fixation locations only [Duchowski 2003]. This makes sense since the viewer is unable to perceive details in the stimuli during saccades. Thus, we use a fixation filter to find clusters of consecutive gaze points which are inside a circle of a specified radius and within a specified temporal window, typically within 200 ms to 1000 ms [Salthouse et al. 1981].

4.2.2 *Determining fixated objects.* In order to determine fixated objects, we first identify the objects corresponding to individual gaze points using an item buffer as in [Sundstedt et al. 2008]. The item buffer is rendered on the GPU as an image of the scene where object colors are replaced by unique object ids (Figure 3).

Second, we need to consider the whole collection of gaze points belonging to a fixation. One option is to compute the correlation of the items with a Gaussian splat that approximates the sensor density within the human fovea as proposed by Sundstedt et al. [Sundstedt et al. 2008]. The scene objects of the simple maze-like game used in their study (i.e. walls and floor) were divided in equally sized tiles, each serving as an individual item. The energy of the Gaussian splat was then binned over the covered items, and thus a single fixation could contribute to several items.

However, research on cognition found that attributes belonging only to one object can be discriminated more efficiently. This so-called *single object advantage* indicates that attention is in many cases object-based and only one object can be attended to at a time [Duncan 1984; Baylis and Driver 1993; Behrmann et al. 1998]. Hence, we base our work on the assumption that during a fixation, attention is focused on one object only and is not directly related to the foveal sensor density. We use a probabilistic model to find the object with the highest likelihood of being the fixation target.

As the eye-tracker has limited precision and the human oculomotoric system can not hold gaze stable on a fixed position, we have to deal with the fact that the gaze points, sampled at discrete points in time, are distributed within a known uncertainty region. We approximate the density distribution of the continuous gaze paths during a fixation with a bivariate Gaussian kernel. The parameters of the kernel are derived from the constant uncertainty of the eye-tracker [1] and the spatial distribution of gaze points clustered within the fixation. A bivariate kernel provides a better fit to unidirectional drifts of gaze, which were frequently

[1] In our case 0.7°, corresponding to a radius of about 16 pixels.

observed in the gaze data. This kernel is convolved with the area $A(o,t)$ of object o at time t and integrated over the fixation time:

$$C_{fix}(o) = \int_{t_\alpha(fix)}^{t_\omega(fix)} \int_{(x,y)\in A(o,t)} N_{fix}(x - x_{fix}, y - y_{fix})\mathrm{d}(x,y)\mathrm{d}t \tag{1}$$

$$N_{fix}(dx,dy) = \exp\left(-\frac{1}{2\left(1-(\rho_{fix})^2\right)}\left(\frac{dx^2}{(\sigma^x_{fix})^2} + \frac{dy^2}{(\sigma^y_{fix})^2} - \frac{2\rho_{fix}dxdy}{\sigma^x_{fix}\sigma^y_{fix}}\right)\right) \tag{2}$$

The variables $t_\alpha(fix)$ and $t_\omega(fix)$ represent the begin and end time of fixation fix, (x_{fix}, y_{fix}) is the center of the fixation, σ^x_{fix} and σ^y_{fix} denote the standard deviations of the fixation's uncertainty region in both dimensions, and ρ their correlation in (x,y). In particular we obtain σ^x_{fix} from the square root of the sum of the variance of the spatial distribution of gaze points in x and the quadratic error of the eye-tracker (σ^x_{fix} is obtained analogously). Note that we ignore the normalization factor of the Gaussian since it does not influence the final result. To speed up computation, we ignore pixels that lie outside the fixation filter radius plus the above-mentioned eye-tracker error.

Finally, we determine the object which is most likely the target of the fixation as the object with the maximum value C_{fix}:

$$o_{fix} = \arg\max_o \; C_{fix}(o) \tag{3}$$

(a) (b)

Fig. 3: Figure (a) shows an example framebuffer and Figure (b) the respective itembuffer. Each object is represented by a unique color code in the itembuffer. Explosions are transparent particle systems, therefore we use an opacity threshold to render them as opaque single objects into the itembuffer.

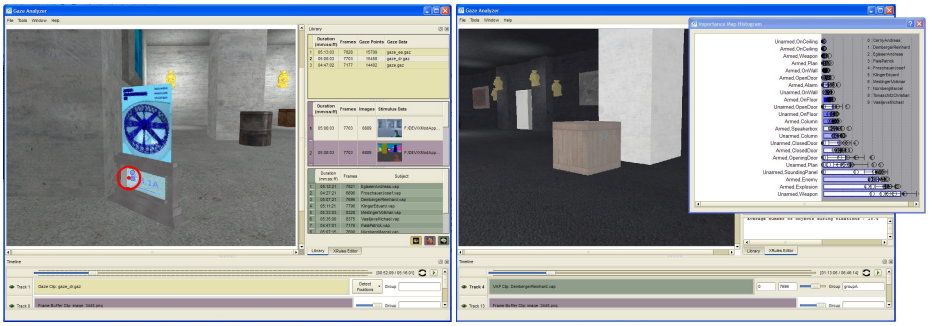


Fig. 4: Screenshots of the analysis software toolbox we implemented to use in our experimental pipeline.

5. CREATING AND USING IMPORTANCE MAPS

Our proposed attention predictor uses an importance map learned from gaze data. In particular, the importance map reflects the distribution of gaze in the space of object *properties*. To obtain enough samples to reliably infer each entry of the importance map, we introduce an additional step which transforms the raw properties to a small set of Boolean *high-level properties*. Importance values are then assigned to combinations ("vectors") of these high-level properties instead of the objects themselves. This allows binning objects with the same set of properties, and, more importantly, taking into account the dynamic *context* of an object (is the object in the center of the screen, was the player armed when the object was fixated etc.).

In this section we will describe how our system transforms the object properties to high-level properties, how the respective importance map is inferred and finally how the derived importance map is used to predict the importance of objects at runtime. Figure 4 shows a screenshot of the tool we use for this workflow.

5.1 Defining high-level properties

In this step, raw properties are "interpreted" and simplified to Boolean values, and additionally, the user also specifies which properties will actually be used in the importance map.

We define a transformation function TF which has the raw properties of the objects $rp(o,t)$ and the player $rp(pl,t)$ at time t as input. The properties of the player are global properties of the current environmental context and may also comprise other global features, such as properties of the camera or the overall brightness in the current frame. Given the set $X = \{\mathrm{x}_1, \ldots, \mathrm{x}_n\}$ of all different high-level properties, the output $\mathrm{x}(o,t) = TF\left(rp\left(o,t\right), rp\left(pl,t\right)\right)$ is a Boolean vector $\mathrm{x} \in \{0,1\}^{|X|}$, called *high-level property vector* (HLV).

In our system, the user can design and modify the transformation function via a scripting language. A script consists of a set of rules which map raw properties to high-level properties (Figure 5):

if CONDITION STATEMENT **then** PROPERTY DEFINITION

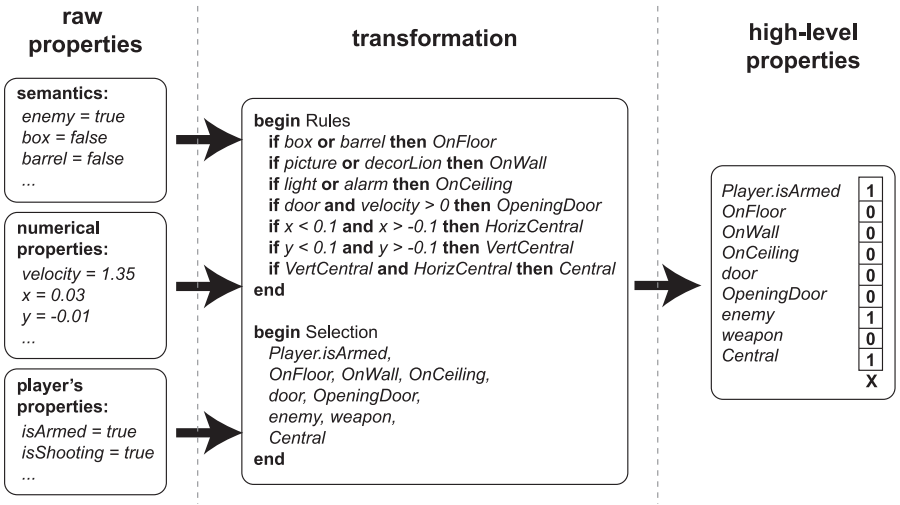


Fig. 5: An illustration of the transformation from raw features to HLVs. In our implementation we pass semantic properties as an array of boolean values to the rule system, while numerical features are represented with an array of float numbers. All rules are executed on each item and the resulting HLVs are represented bitarrays of length $|X|$, where each bit corresponds to one high-level property in X.

In the conditional statement, raw properties can be combined using standard operators, in particular logical operators (e.g. $\wedge$, $\vee$), arithmetical operators (e.g. $+$, $-$) and relational operators (e.g. $>$, $<$). The condition must evaluate to a Boolean value which is then assigned to the property specified by the right hand side of the rule. The resulting property can be used in conditional statements of subsequent rules. Note that numeric properties need to be quantized to a Boolean property, e.g. using natural language terms like "big" or "far".

The second part of the script, the "selection pass", declares which combination of these properties makes up a high-level property vector (HLV). In order to be able to infer stable importance values from reasonable sample sizes, rule scripts have to be carefully written in order to minimize the number of different HLVs that actually occur (i.e., in the worst case this number increases exponentially with the number of properties being combined). The sample size depends on the number of participants, the number of recorded fixations and the distribution of the objects.

5.2 Learning an importance map

A schematic overview of the pipeline can be seen in Figure 6. We denote the importance map as a scoring function $I(\mathrm{x})$, which maps each possible HLV x (interpreted as an integer key) to an importance score.

To calculate $I(\mathrm{x})$, we accumulate the time T_{fix} that objects with that x were fixated and normalize it by the time T_{vis} that objects with that x were visible during a fixation (i.e., when they were a potential fixation target):

$$I(\mathrm{x}) = \frac{T_{fix}(\mathrm{x})}{T_{vis}(\mathrm{x})} \tag{4}$$

This model takes into account changes due to visibility of objects and changes in the global (or player) context, but not changes referring to other objects in the scene. However, generalizing the solution to contextual dependencies is a hard problem, as it would increase the dimensionality of the importance map to the size of the powerset over X, and a sufficient density of gaze samples would be difficult to acquire.

Thus we compute $T_{fix}(\mathrm{x})$ and $T_{vis}(\mathrm{x})$ as follows:

$$T_{fix}(\mathrm{x}) = \sum_{fix \in Fix} \int_{t_\alpha(fix)}^{t_\omega(fix)} \varphi_{\mathrm{x}}(\mathrm{x}(o_{fix}, t))\mathrm{dt} \tag{5}$$

$$T_{vis}(\mathrm{x}) = \sum_{fix \in Fix} \int_{t_\alpha(fix)}^{t_\omega(fix)} \sum_{o \in V_t} \varphi_{\mathrm{x}}(\mathrm{x}(o, t))\mathrm{dt} \tag{6}$$

with the characteristic function:

$$\varphi_{\mathrm{x}_i}(\mathrm{x}_j) = \begin{cases} 1, & \text{if } \mathrm{x}_i = \mathrm{x}_j \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

The function $\mathrm{x}(o,t) = TF(rp(o,t), rp(pl,t))$ returns the HLV for object o at time t. Fix is the set of all fixations taken as input training data. The object assumed as the target of fixation fix is denoted as o_{fix}, and V_t is the set of visible objects in the item buffer at time t. The characteristic function φ_{x} is used to count exclusively for HLV x.

Note that there is a subtlety involved in calculating $T_{vis}(\mathrm{x})$ because there are several methods how to count objects that exhibit the same x in a single frame. In addition to simply counting the objects as shown above, we have also tried (1) just counting the frames in a fixation where x occurred at least once, (2) using the sum of the pixel areas of the objects exhibiting x, and (3) counting the number of objects exhibiting x, divided by the number of visible objects.

Our experience was that (1) is sensitive to objects sharing the same x: if several objects with the same x appear together in some frames, x receives a higher score because the fixation probability increases, but the visibility correction does not increase accordingly; (2) causes unstable scores, since variations in the size between different objects do not seem to correlate well with fixation probability. Compared to (3) the method we use worked better because it lets those situations where the player can select between many objects contribute stronger to the learning process.

5.3 Background objects

There are two problems when including background objects into our prediction algorithm. First, many hits on background objects result from the fact that users tend to scan the sil-

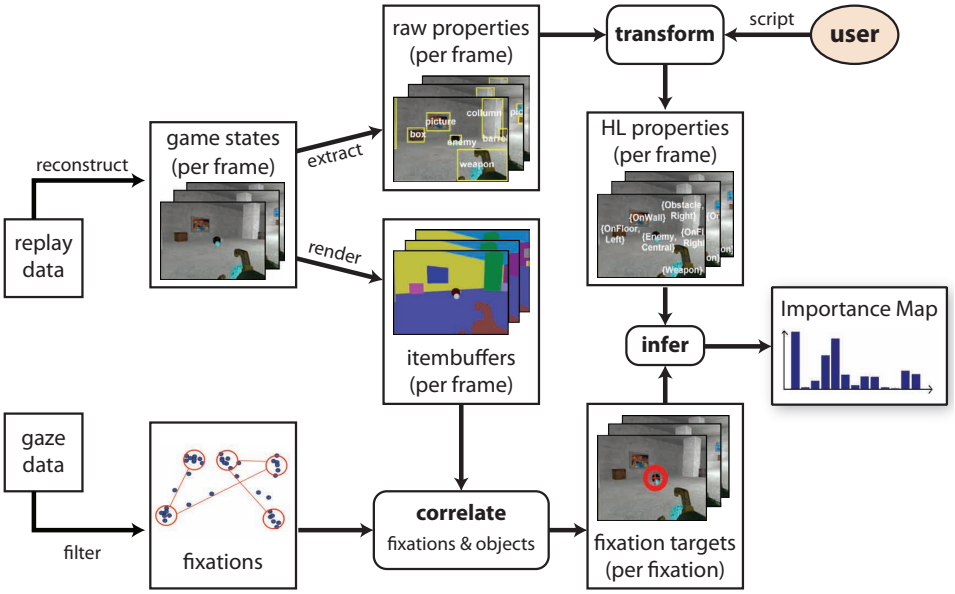
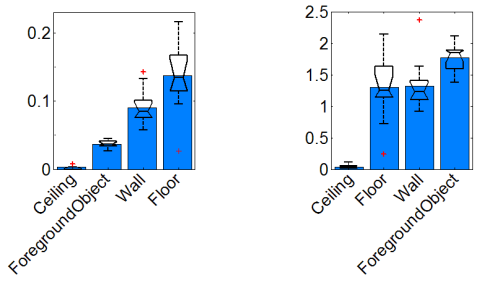


Fig. 6: An overview of the complete pipeline used to derive importance maps.



(a) Normalized with #objects (b) Normalized with #pixels

Fig. 7: Importance maps inferred for different background objects with two different heuristic inference methods. We added boxplots to illustrate the distribution of the scores generated from the data of each individual participant, while the bars are the average values obtained when processing the data of all participants together. The left image depicts the importance map which we get by applying the inference method proposed in Section 5.2, while the second was generated by normalizing with the fraction of pixels the objects cover on the screen. For background objects the second method is more adequate, since background objects can not really be counted consistently (e.g. there is only one floor, but often several walls) and there is a high correlation between the size and the fixation probability.

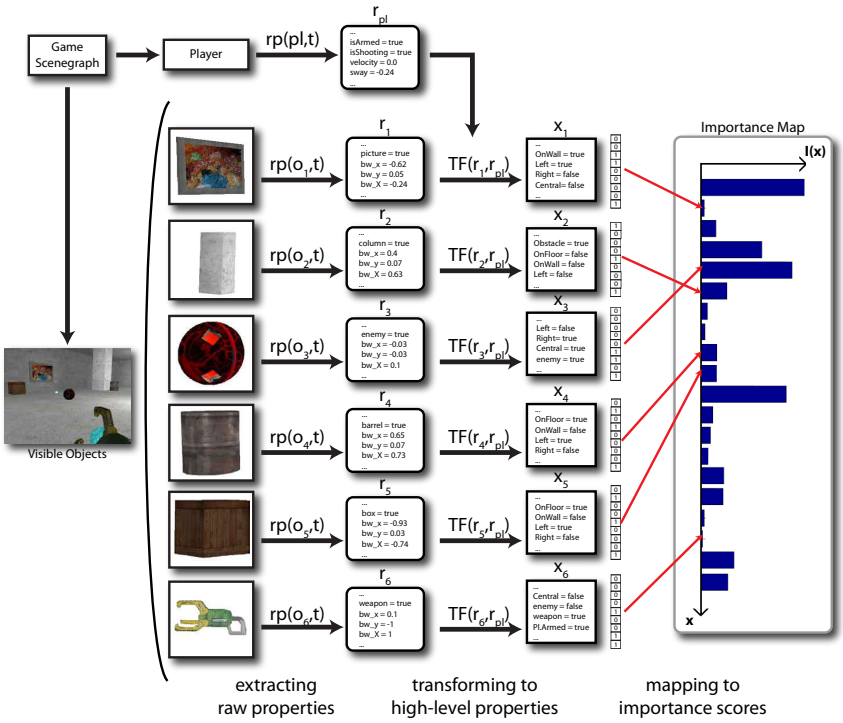


Fig. 8: An illustration how importance values are assigned to the objects in the scene. First, visible objects are determined and their raw properties are inferred from the scenegraph. Then the transformation function TF defines the respective HLVs using the raw properties of a particular object and those of the player as input. Since high-level properties are encoded as words of bits, they can be readily converted to an integer value used as a key to map to the entries of the importance map being learned on the data beforehand.

houettes of foreground objects. These hits should not lead to an increased importance of the background. Second, some background objects are hit very often simply because of their large extent through the whole level (e.g., floor, ceiling, walls). This introduces a bias towards background objects, which is unwanted since they rarely contain important features. While an area-weighted importance could remove this bias, this would not work well for foreground objects as described before (see Figure 7 for results using area weighting).

We therefore exclude background objects from both the set of visible objects V_t and the set of fixated objects (and thus the corresponding fixations from Fix) using a simple rule in the scripting interface. This considerably improved the performance of the predictor. We plan to investigate predictors that also include background objects in future work, for example by using saliency maps or by tiling them to reduce the size dependency.

5.4 Gaze prediction at runtime

After an importance map has been constructed from an eye-tracking study as described above, this map can be used at runtime to predict visual attention by generating an *importance value* for each visible object in a frame at a given time. This requires executing that part of the pipeline shown in Figure 2 that leads to the output of high-level property

sets for each object. However, in this case the extraction of properties needs run in real time. Thus, instead of using an item buffer in an offline stage, we directly compute visible objects using an efficient visibility algorithm [Bittner et al. 2004]. For each visible object, raw properties are converted into the corresponding HLV, which is used as an index into the importance map to retrieve a corresponding importance score, as illustrated in Figure 8.

The final importance value $I(o,t)$ of an object o in a particular frame t is computed by normalizing the importance score with the maximum importance score in the current frame, so that the most important object has an importance value of one:

$$I(o,t) = \begin{cases} \frac{I(\mathrm{x}(o,t))}{\max\{o \in V_t^{fg} | I(\mathrm{x}(o,t))\}}, & \text{if } o \in V_t^{fg} \\ 0, & \text{otherwise} \end{cases} \quad (8)$$

This normalization provides a rough approximation to context sensitivity: in the presence of a very important object (e.g. enemy), the importance score of objects with low importance is suppressed, while it is amplified in the absence of important objects. However, the choice of normalization method actually depends on the application using the predictor, and needs to be further investigated for particular cases. We used it only for visualization purposes up to now.

6. ASSESSING THE QUALITY OF GAZE PREDICTORS

In the previous sections we explained how we learn an importance map from gaze data and properties of the stimuli recorded throughout an eye-tracking session, and how to use it as an attention predictor. The second part of the analysis is to evaluate the performance of an importance map in terms of its gaze prediction capabilities. The purpose of the evaluation is to provide the user of the analysis tool with feedback about the quality of the importance map, so that he can carry out improvements by refining the transformation function.

To this end, the fixations detected in the gaze recordings are correlated with the predictions of the importance map using two different metrics. We use the dataset from one specific recording session and compare the actual gaze points to the attention targets predicted by an importance map learned from other recording sessions. As every player chooses a different path through the level and observes the scene from arbitrary viewpoints, this evaluation strategy demonstrates that predictions of an importance map, which was trained with gaze data of a particular group, can generalize to other subjects, other viewpoint traces, or even other levels, assuming similar tasks and scene content.

Note that in all the evaluation, background objects (and corresponding pixels) are ignored as described above.

6.1 Correlation between predictions and actual gaze

To score the prediction performance, we assess the degree of correlation between fixated objects and high importance values. Assuming a better than chance predictor, we expect an above-average probability for a fixation to be targeted at an object with an above-average importance value. The probability should increase with the reliability of the predictions,

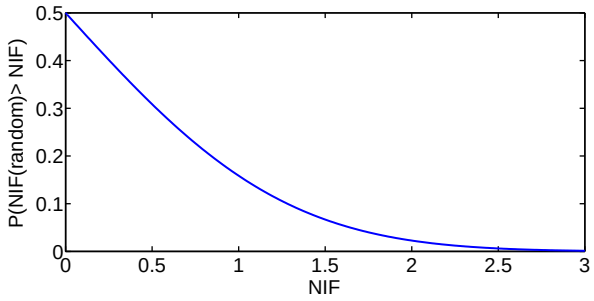


Fig. 9: To get a better understanding of the NIF measure we depicted the probabilities to observe a greater or equal NIF for a randomly chosen object, under the assumption that the importance values in one frame are normally distributed.

and thus the amount of deviation of the importance value of fixated objects to the average is an indicator for the reliability and selectivity of the importance map.

Inspired by the *normalized scanpath saliency* (NSS) proposed in [Peters et al. 2005] for the evaluation of saliency maps as gaze predictors[2], we introduce the measure *Normalized Importance of Fixated Objects* (NIF) as the deviation of the importance value of a fixated object from the average importance value, normalized by the standard deviation of all importance values in the current frame. Higher values of NIF correspond to better prediction performance, meaning that the actually fixated objects are also predicted to be important (see also Figure 9). Since fixation duration also correlates to the importance of a fixation target o_{fix}, we integrate the deviation for one fixation fix between its begin time $t_\alpha(fix)$ and end time $t_\omega(fix)$:

$$NIF(fix) = \frac{1}{t_\omega(fix) - t_\alpha(fix)} \int_{t_\alpha(fix)}^{t_\omega(fix)} \frac{1}{\sigma_t} \left(I(o_{fix}, t) - \bar{I}_t\right) \mathrm{d}t \tag{9}$$

with:

$$\bar{I}_t = \frac{1}{|V_t^{fg}|} \sum_{o \in V_t} I(o,t) \tag{10}$$

$$\sigma_t = \sqrt{\frac{1}{|V_t^{fg}| - 1} \sum_{o \in V_t} (I(o,t) - \bar{I}_t)^2} \tag{11}$$

$I(o,t)$ denotes the predicted importance value for an object o in the frame at time t. The mean importance value $\bar{I}_t$ and the standard deviation of importance values σ_t are both computed from the distribution of importance values assigned to the visible objects V_t at time t.

[2]Saliency maps predict the endpoints of saccades and therefore the future end point of a saccade is often used for evaluation. Our visual attention predictor on the other hand quantifies the duration of a fixation and thus uses the currently fixated object.

For a sequence of fixations $Fix = \{fix_1, \ldots, fix_n\}$, we compute the weighted mean using the fixation durations $\Delta(fix) = t_\omega(fix) - t_\alpha(fix)$ as weights:

$$NIF(Fix) = \frac{1}{T^{Fix}} \sum_{fix \in Fix} \Delta(fix) NIF(fix), \tag{12}$$

with a total fixation time $T^{Fix} = \sum_{fix \in Fix} \Delta(fix)$.

This metric is invariant to rescaling, such as the normalization inherent in $I(o,t)$. For a random predictor, NIF gives 0, while for a good predictor the value is significantly higher than 0, indicating that there is some correlation between objects assigned with above-average importance values and the fixated objects. Normalizing with the standard deviation accounts for the distribution of importance values in a particular frame, so that importance values which have a low probability have a stronger weight. Figure 9 depicts the estimated probabilities of observing an equal or higher NIF for randomly selected objects. Those were computed with the normal cumulative density function under the assumption that importance values in one frame are normally distributed.

Since the prediction quality varies from fixation to fixation, we also compute the standard deviation across a sequence of fixations. To account for the different fixation durations, we use the weighted variance:

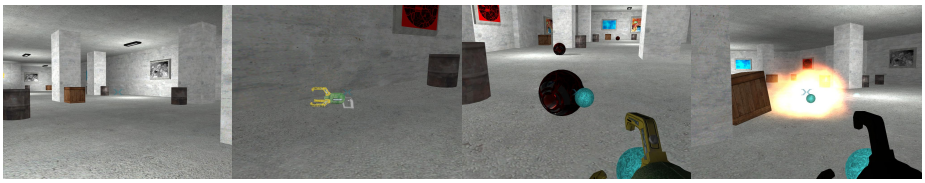
$$\sigma^{NIF}(Fix) = \sqrt{\frac{|Fix| T^{Fix}}{(|Fix| - 1) T^{Fix}} \sum_{fix \in Fix} \Delta(fix) \left(NIF(fix) - NIF(Fix)\right)^2} \tag{13}$$

6.2 Distribution of importance values

While the previous metric provides an assessment of the predictor in a single numeric value, we also provide a more intuitive assessment of the prediction performance by analyzing the distributions of importance values corresponding to random fixations and human fixations.

For human fixations, we compute a histogram D_{fix}, binning the number of fixations according to the importance values of the corresponding fixated objects. For random fixations, we compute either a pixel-based histogram D_{pix}, which bins the amount of visible pixels according to importance values, or an object-based histogram D_{obj}, binning the number of objects instead of pixels. A user which fixates random positions on the screen would produce D_{pix}, whereas a user fixating randomly selected objects would produce D_{obj}. The reason for using two models for random fixations is that it is not entirely clear how much the size of an object actually influences visual attention, since a human observer recognizes a scene as a composition of objects and not pixels. A description how D_{pix}, D_{obj} and D_{fix} are exactly computed can be found in Appendix A.1.

Besides a visual inspection of the distribution, [Peters and Itti 2008] proposed to assess the quality of an importance map in terms of its information value, which we is estimate as the distance between the distribution of importance values expected by random fixations (i.e.,



(a) (b) (c) (d)

Fig. 10: Selected frame buffer images of the level used in our exemplary study. In the first phase of the game the player explores a large environment (a), and searches for the weapon (b). As soon the player retrieves the weapon, enemies will spawn and the player has to defend himself and simultaneously attack the enemies (c). The enemies are spherical robots that are moving in the environment and shoot at the player, and they explode after termination (d).

D_{obj} or D_{pix}) and human fixations. To this end, the symmetric Kullback-Leibler distance (KL) can be used to evaluate the distance between two normalized distributions D_{rand} and D_{fix}:

$$KL(D_{rand}, D_{fix}) = \frac{1}{2}\sum_{i=1}^{N}\left(D_{rand}[i]ln\frac{D_{rand}[i]}{D_{fix}[i]} + D_{fix}[i]ln\frac{D_{fix}[i]}{D_{rand}[i]}\right) \quad (14)$$

The KL metric measures the relative entropy between two distributions. In other words, it computes the number of additional nats[3] required to encode a distribution q, with the coding scheme used for an entropy encoded distribution p. To make the metric symmetric, the relative entropy is computed in both directions and the mean of both is returned.

7. EXPERIMENTAL STUDY

In this section we describe a concrete example application of the proposed pipeline. We carried out a pilot study with a first-person-shooter (FPS) game developed in our lab. Shooter games are one of to the most popular computer game genres, and are thus a reasonable choice for study. In the following sections we will describe how we designed a level, acquired the data, generated different kinds of importance maps and evaluated the prediction performance of the corresponding gaze predictors.

7.1 Data acquisition

7.1.1 *Level design.* The test level used for eye-tracking was designed by a student who was naive about the purpose of the study. The game was set in an indoor environment, composed by a network of rooms that contain attacking enemies. Example images of the game can be seen in Figure 10. The level had three tasks: first, find a weapon (with no enemies yet), then navigate to the "boss room", eliminating any enemies along the way, and finally destroy a crowd of enemies in the boss room to win the game. After solving a task, the player was allowed to enter the next section with the next task.

[3]logarithmic units of entropy

We disabled certain attention-attracting effects and features built into the original game which could potentially bias gaze behavior. For example, our current system does not take into consideration GUI elements, therefore, we disabled all elements of the heads-up display except the cross hairs. The level designer was also instructed to avoid strong variations in the illumination of the environment, as these are not captured by our system.

7.1.2 *Setup.* We used a Tobii x50 eye-tracker, running at 50 Hz, which was placed in front of the display. The calibration and configuration was carried out according to the manufacturer's instructions. The eye-tracker together with the game's state recording functionality were started and stopped simultaneously, at the beginning and end of each gameplay session. For each session, two data files were stored: a file containing the gaze data provided by the eye-tracker, and another file exported by the game recording all changes of the game's scene graph.

All experiments were run on an Intel Core 2 Duo workstation, clocked at 2.4 GHz with 2GB RAM, and an NVIDIA GeForce 8800 GTX graphics card. This setup was sufficiently powerful to simultaneously run the game, including the recording functionality, at a framerate greater than 50 fps, and also operate the eye-tracker. The display was a commodity LCD display (IBM ThinkVision L200p with a resolution of 1600 x 1200 pixels at 100 dpi), set up at a resolution of 1024×768 pixels, displayed at a scaled down effective viewing window with dimensions of 34×27 cm. We used a smaller viewing window than possible with the monitor in order to minimize inaccuracies in eye-tracking near the outer regions of the wide screen. The setup was located in a dark room with dim lighting to avoid reflections on the screen. A commodity PC stereo sound system was used for the audio output and the loudspeakers were placed to the left and right of the screen.

7.1.3 *Eye-tracking session.* Each participant played the level once while the eye-tracker was recording his gaze. The calibration of the eye-tracker was carried out shortly before the game started and the participants were seated comfortably in the best position as recommended by the user's manual of the eye-tracker. To reduce the risk of inaccuracies, the participants were instructed to attempt to hold this position as well as they could. Before the eye-tracking session, all participants played another distinct level of the game as an introduction. To avoid surprise effects, this level contained every object which appeared later in the eye-tracked level of our study. The players were informed in detail about the tasks and goals of the level before the session.

The time the participants needed to complete the level was between four and six minutes. Between one and two minutes were required to find the weapon, about two to navigate to the boss room and kill all the ten enemies on this path, and about one and a half minutes were required to fight the final battle against twelve enemies in the boss room.

At the end of the session, a short questionnaire quantified the gaming skills of the participants. We recorded more than 20 participants. Unfortunately, long eye-tracking sessions cause an increasing drift of the eye-tracker's accuracy. Towards the end of many sessions, the precision of the gaze recordings was unacceptable. Some experiments (e.g. [Sundstedt et al. 2008]) were conducted with a chin-rest to avoid such complications. However, eye movements of participants on a chin-rest are vastly different than when they are free to make both head and eye movements [Collewijn et al. 1992]. After a diligent inspection

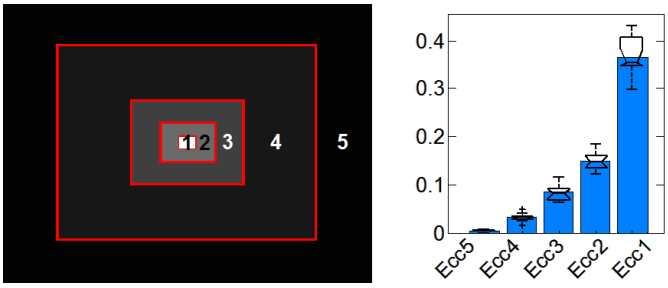


Fig. 11: The eccentricity-based importance map defined in "script 1" (Appendix A.3). In the left image we illustrate the regions defining the eccentricity bins. In the right image, the bars illustrate the importance values generated from ten datasets together. The boxplots illustrate the distribution of the importance values generated from the data of single participants.

with our gaze visualization tool, it turned out that only 10 gaze datasets had a reliable accuracy. The respective participants were all male, had normal or corrected to normal vision and their FPS gaming skills were good to very good.

7.2 Learning importance maps

We will now present three illustrative examples of particular importance maps generated from the data acquired in our study. Since an importance map is specified by a transformation function, we listed in the Appendix the relevant script defining the transformation function for each example.

7.2.1 *Eccentricity.* When we inspected the gaze behavior with our gaze visualization tool, the most obvious observation was that most fixations were close to the center of the screen. Similar results were reported in [Kenny et al. 2005] from a study on gaze behavior in FPS games. We authored a script (see Appendix A.3) to learn an importance map which captures this behavior. Five rules were specified, which generate five bins for the degree of an object's eccentricity (see Figure 11, left). The resulting importance map (see Figure 11, right) has a high peak for the inner eccentricity bin 1, and rapidly decreases with eccentricity. The most probable explanation for this is that the cross hair of the weapon is displayed at the center of the screen.

7.2.2 *Semantic properties.* In order to demonstrate that our approach can generate importance maps based on more complex properties, we created a script (Appendix A.4) to infer importance values according to the *semantic object type*. We combined this with one relevant property of the player, notably whether he is "armed" or "unarmed". With this combination, differences in the gaze distributions according to the current task of the player, which is to search the weapon while being "unarmed" and to shoot enemies as soon he possesses the weapon, should be captured. In Figure 12 we depicted the values according to the HLVs of the respective importance map.

The importance map we derived shows that only few objects are outstanding attention attractors, notably explosions, the enemy, the weapon (as long the player is searching for it), and a panel emitting sound. Explosions occur when an enemy is eliminated. One

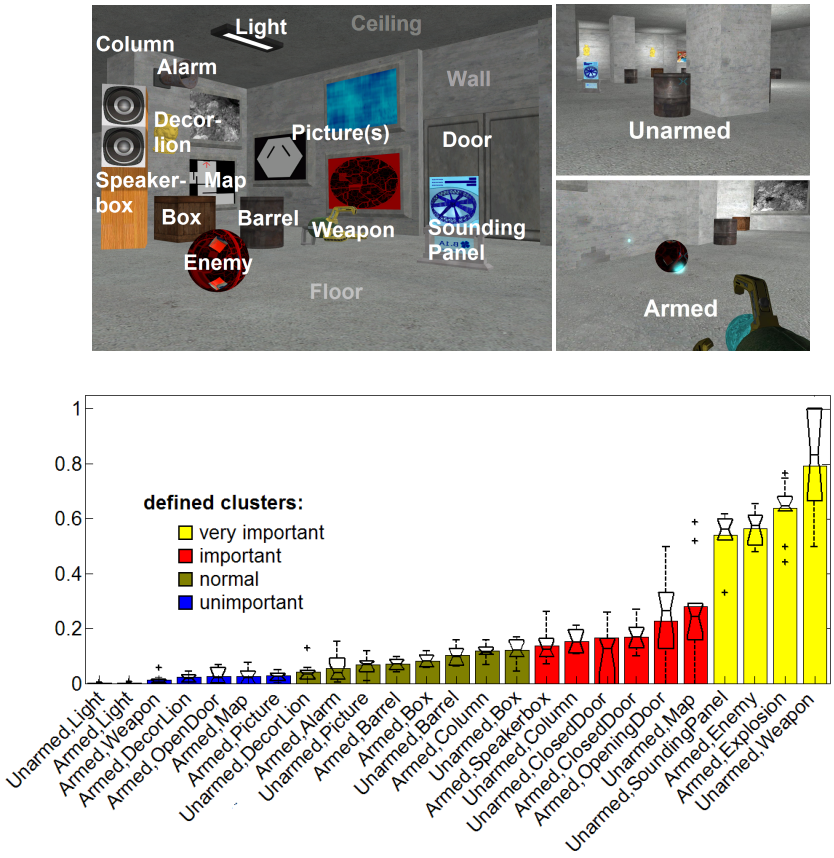


Fig. 12: An importance map from semantic properties. Whether the player is armed or unarmed was added as an additional HL property. Not all properties appeared for both states, since some objects emerged in parts of the level where the player was either armed or unarmed. In the image above we illustrated the semantic categories with the corresponding objects in the game. For the next example, we clustered the semantical properties according to their importance value into four categories, which we use to combine semantical and spatial properties (Section 7.2.3)

reason for their high importance is that the player keeps his focus on the exploding enemy. The weapon has high importance before the player retrieves it, but as soon the player carries the weapon, its importance drops significantly (see scores according to the states "Armed.Weapon" and "Unarmed.Weapon" in Figure 12). The sound-emitting panel is a conspicuous object in the scene. It has a screen displaying a rotating wheel and emits a sound which is congruent to the animation. This kind of crossmodal cue possibly causes a high attentional response. An interesting observation is that the importance of the overview map changes considerably between both tasks. When the player searches the weapon he seems to have time to pay more attention to maps and objects suspended on the walls, while as soon as he is armed and enemies are attacking, he has different priorities.

7.2.3 *Semantics and Eccentricity.* According to the observations from the preceding steps, we tested a combination of both models (Appendix A.5). To get enough samples

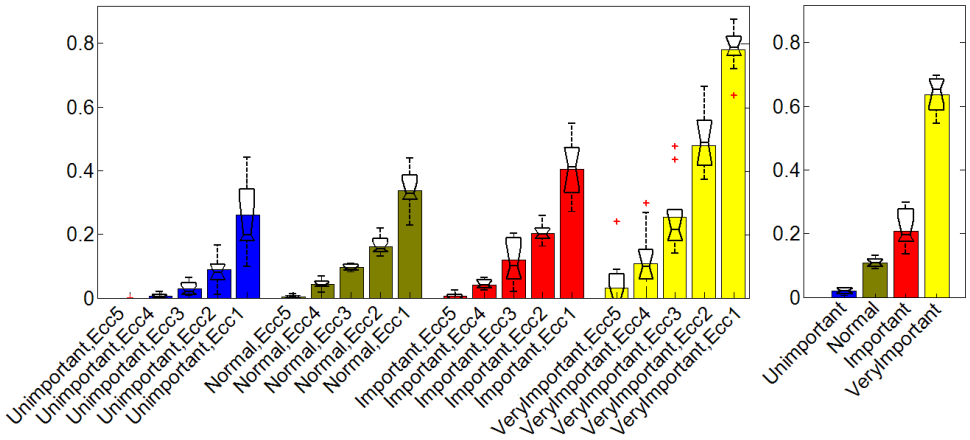


Fig. 13: The left figure shows the score function for a combined model of eccentricity and semantics, generated using "script 3" (Appendix A.5). The right diagram shows the scores for the clusters without combining them with eccentricity properties.

(a) (b) (c)

Fig. 14: Visualization of the predicted importance, (a) from semantic properties, (b) eccentricity, and (c) a combination of both. Our visualization is a grayscale overlay to the framebuffer, where object importance is proportional to the displayed brightness.

for every entry in the importance map, we put the semantic categories into four clusters. These clusters were intuitively specified according to the importance values in the scoring function inferred from "script 2". The clusters are illustrated in Figure 12 with different colors. We labeled them with "very important", "important", "normal" and "unimportant". Figure 13 shows for each cluster the importance map scores per eccentricity bin.

7.3 Evaluation of the gaze predictors

7.3.1 *Results.* The whole walkthruogh of one player (participant 10) with a synchronous visualization of his gaze and the predictions of the importance map was rendered in the accompanying video. A grayscale overlay was used to visualize the importance values, where object importance is proportional to the displayed brightness. Apart from inspecting the correlation of fixations and importance maps visually (e.g. screenshots in Figure 14), an objective analysis with the quality metrics described in Section 6 (NIF value and histograms) was carried out. We perform the analysis for each of the three phases of the

game separately, i.e., *searching* for the weapon, *navigation and battle* on the way to the boss room, and finally the *boss room*. The corresponding three tasks show increasing task intensity, which has an impact on the prediction performance.

The results for the NIF metric (together with the weighted standard deviations across fixations) are depicted in Table I. Recall that NIF measures how much higher the importance values of fixated objects are in comparison to the average importance value of all visible objects, normalized by the standard deviation of importance values. Table II shows the distributions of importance values expected for random fixations (D_{pix}, D_{obj}) in comparison to the recorded (human) fixations (D_{fix}), as described in Section 6.2. A resolution of ten bins was used to build the histograms. We created a histogram for each subject separately (with an importance map generated from the remaining subjects) and then averaged these histograms to arrive at the figures in Table II. The high counts for bin $[0.9, 1.0]$ result from the normalization assigning 1.0 to the most important object in each frame. To allow a better comparison, we also show the relative differences, i.e., $D_{fix}[i]/D_{obj}[i]$ and $D_{fix}[i]/D_{pix}[i]$. A red line through 1 shows which importance values receive more, or respectively less, fixations as expected for randomly chosen fixation targets.

7.3.2 *Discussion.* There is clear evidence for all conditions that the predictors perform better than chance. Eccentricity turns out as the most effective and reliable property to predict visual attention in a FPS. The predictor based exclusively on semantic properties performs clearly better than chance even in the least task-intensive phase (searching) (t-test with NIF: $t = 27.3$, $dof = 9$, $p < 0.001$), where the importance value of fixated objects is on average between 68% (participant 3) and 97% (participant 8) of the standard deviation higher than the mean importance of the current frame. Note one standard deviation (i.e., $NIF = 1.0$) corresponds approximately to a probability of 16% to observe for a randomly selected object an equal or higher importance value than that of the actually fixated object (see Figure 9). In this phase enemies were absent and the task-target "Weapon" was on screen less than 7% of the time. This gives us a strong hint that, apart from obvious task relevance, also other semantic properties provide a considerable potential to discriminate their impact on visual attention. However, the relatively high standard deviation of the NIF across fixations shows that the degree of correlation between importance values and fixations fluctuates considerably.

The KL distance of 0.2 between D_{pix} and D_{fix} during the "Searching" phase (Table II top-left histogram) indicates a weak, although clear difference. We explain the small difference between D_{pix} and D_{fix} in bin $[0.9, 1.0]$ by the fact that the columns were the most important objects in many frames during the "Searching" period. As columns are the largest objects and appear very frequently, they contributed a relative strong weight to the higher bins of D_{pix}. As D_{fix} is invariant to the size of fixated objects, the difference to D_{pix} was lowered.

Comparing NIF values for the three scripts, the importance map which combines semantic properties and eccentricity could improve the performance compared to the eccentricity based map in all phases of the game. The improvements were, however, decent, but even for the worst case, the "Searching" phase, a paired students t-test reveals that the NIF scores for "script 3" are significantly better than those for "script 2" ($t = 4.56$, $dof = 9$, $p = 0.001$). We explain the rather moderate performance increase with a strong correla-

Searching				
I.Map type	semantic	eccentricity	ecc.& sem.	N
participant 1	0.94 ± 1.16	1.24 ± 1.15	1.30 ± 1.11	143
participant 2	0.81 ± 1.05	1.14 ± 1.24	1.19 ± 1.16	149
participant 3	0.68 ± 1.12	1.65 ± 1.13	1.68 ± 1.06	144
participant 4	0.81 ± 1.28	1.03 ± 1.29	1.16 ± 1.30	213
participant 5	0.75 ± 1.27	1.00 ± 1.29	1.06 ± 1.28	111
participant 6	0.80 ± 1.06	1.29 ± 1.32	1.35 ± 1.26	148
participant 7	0.79 ± 1.12	1.12 ± 1.31	1.16 ± 1.29	149
participant 8	0.97 ± 1.32	1.54 ± 1.24	1.57 ± 1.21	147
participant 9	0.72 ± 1.04	1.61 ± 1.32	1.60 ± 1.23	145
participant 10	0.91 ± 1.32	1.18 ± 1.26	1.27 ± 1.25	140

Navigation & Battle				
I.Map type	semantic	eccentricity	ecc.& sem.	N
participant 1	1.66 ± 1.40	1.99 ± 1.18	2.13 ± 1.30	178
participant 2	1.59 ± 1.34	1.78 ± 1.10	1.89 ± 1.22	125
participant 3	1.47 ± 1.25	2.17 ± 1.09	2.30 ± 1.04	175
participant 4	1.97 ± 1.44	2.16 ± 1.05	2.41 ± 1.08	176
participant 5	1.99 ± 1.43	2.05 ± 1.28	2.33 ± 1.32	103
participant 6	1.73 ± 1.10	1.89 ± 1.25	2.09 ± 1.21	141
participant 7	1.53 ± 1.61	1.85 ± 1.22	1.89 ± 1.40	173
participant 8	1.82 ± 1.40	2.11 ± 0.93	2.30 ± 1.11	137
participant 9	2.13 ± 1.29	2.26 ± 1.09	2.47 ± 1.12	151
participant 10	2.02 ± 1.46	2.12 ± 1.35	2.28 ± 1.43	212

Bossroom				
I.Map type	semantic	eccentricity	ecc.& sem.	N
participant 1	1.89 ± 1.34	1.93 ± 1.21	2.09 ± 1.31	206
participant 2	2.14 ± 1.21	1.96 ± 1.14	2.25 ± 1.09	96
participant 3	1.85 ± 1.34	2.13 ± 0.81	2.36 ± 0.80	131
participant 4	2.42 ± 0.91	2.51 ± 0.88	2.70 ± 0.79	85
participant 5	2.27 ± 0.99	2.45 ± 0.82	2.67 ± 0.79	84
participant 6	2.02 ± 0.99	2.15 ± 0.98	2.32 ± 0.96	128
participant 7	2.31 ± 1.27	2.42 ± 0.91	2.62 ± 0.97	82
participant 8	2.07 ± 1.34	2.12 ± 1.05	2.38 ± 1.09	130
participant 9	2.34 ± 0.95	2.23 ± 0.94	2.44 ± 0.84	84
participant 10	2.02 ± 0.92	2.15 ± 0.70	2.21 ± 0.76	65

Table I: Evaluation of the importance maps generated from different scripts. We display the mean and the standard deviation of the NIF, which was evaluated for all N fixations of each participant (excluding those which fell on background objects, according to the exclusion rules of the scripts). The importance map which was evaluated on the gaze data of one participant was always generated from the datasets of the other nine participants.

tion of the eccentricity and the importance inferred from semantics. If an important object appears on screen, the player tends to orientate the camera soon towards this object and the eccentricity-based predictor will provide a good prediction, but with delay, whereas semantic predictors can better estimate which object will probably be attended next.

7.3.3 *Conclusion.* Though eccentricity provided very good predictions which to start with, we discovered that there is a statistically significant (even if it is weak) improvement when semantic properties are combined with eccentricity. The strong focus towards the center seems to be characteristic for first-person games (where the camera is controlled by the user), but for other games, predicting attention may require more than eccentricity to achieve a good performance. Hence, we used our system to demonstrate that also semantic properties (without spatial information) combined with the player's current task predict attention significantly better than chance, which should open a new avenue for future work. Overall, we conclude that visual attention in a first-person game is affected by both, spatial organization and semantics of objects.

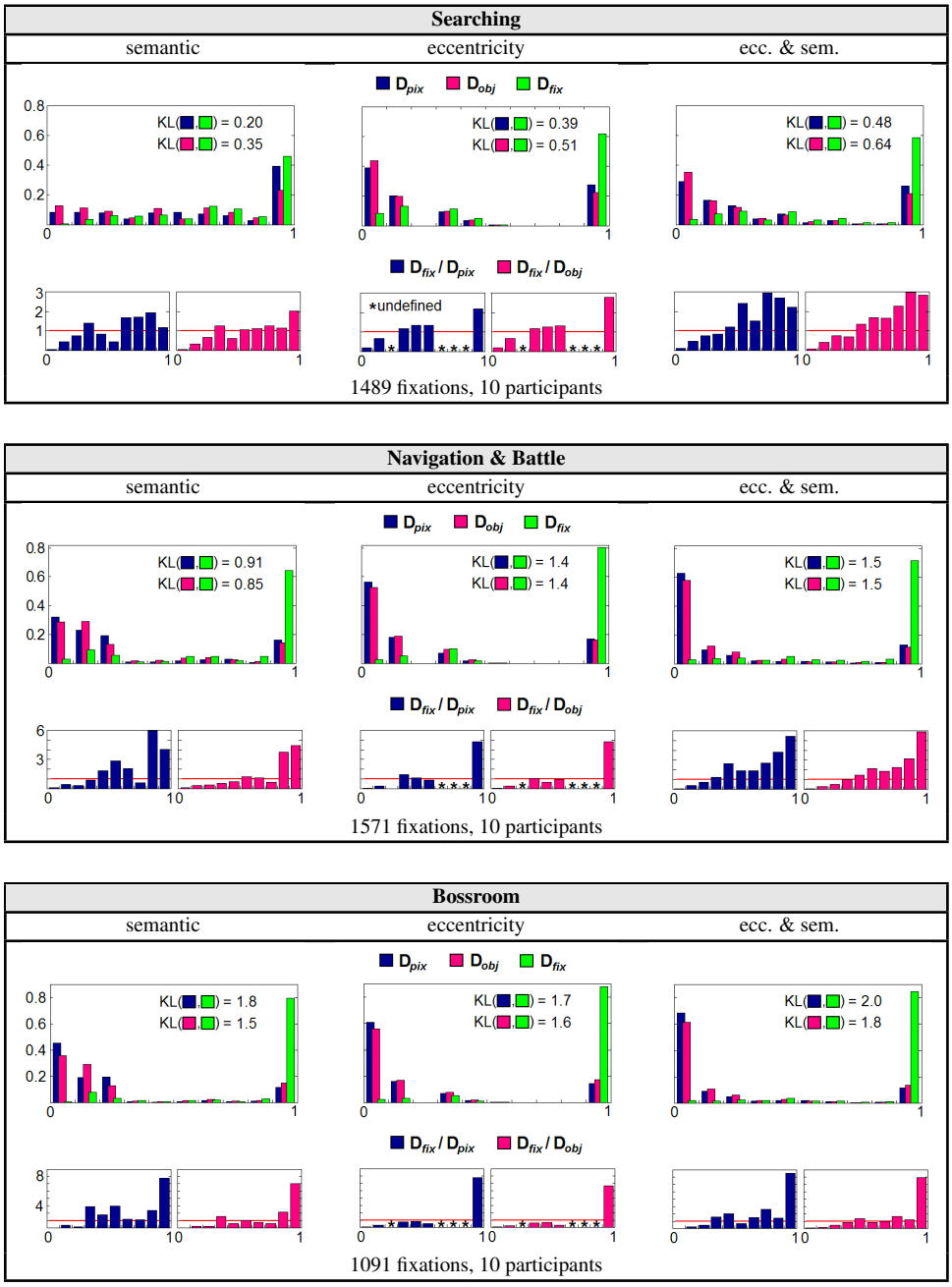


Table II: The distribution of importance values expected by fixations on randomly chosen objects or positions, in comparison to the distribution of importance values among actually fixated objects. The image on the top depicts the distributions side-a-side, while the images below illustrate the difference in D_{fix} relative to the distributions expected by random fixations.

8. GENERAL CONCLUSION AND OUTLOOK

8.1 Contribution

In this article, we have presented a pipeline to infer attention predictors from eye-tracking data. The main contribution is that the approach relates attention to *object and scene properties*, and thus allows predicting attention in dynamic environments with dynamically changing viewpoints (and thus object visibilities) and tasks. The approach allows a human operator to specify through a scripting interface how importance values are assigned to combinations of object properties based on observed gaze data.

Besides providing attention predictors that can be evaluated in real time, for example for selective rendering, the system can also be used as an analysis tool (e.g. for game designers). According to the results of our examples, one would, for instance, concentrate most efforts on the design of enemies or explosions, and invest less resources to polish the appearance of objects on the ceiling, or decorations in rooms where many enemies attack the player. The fact that eccentricity serves as a very strong predictor in FPS games could also be utilized to build importance maps without an eye-tracker, but using the crosshair instead.

Although the examples in this paper revealed results that are mostly not surprising (e.g., with respect to eccentricity), our analysis method can quantify the evidence of certain suspicions about gaze behavior. Due to the generic system design, the approach is capable to analyze gaze behavior in environments with rich detail and sophisticated tasks, where it is rather difficult to predict gaze intuitively.

8.2 Limitations

Our results suggest that we can already predict attention well, but we need to point out that we did not address bottom-up mechanisms affecting the control of attention. We believe that the two approaches are complementary and can be combined in future work. For example, while saliency maps are good at predicting what could be the next gaze target, our approach tries to answer the question how long attention is likely to dwell on a particular object.

Further, while image space approaches work on arbitrary images sequences, this work focused on object-based importance maps, which requires an application where the internal object structure is accessible and item buffers can be created. However, the interface to the analysis tools to be integrated is light weight and it should be easy to adapt host applications to this requirement. Also note that image space approaches require object recognition and identification.

In our experiment, we excluded background objects from the analysis and used objects with a clear semantic category and a clear geometrical outline. However, environments in commercial games frequently comprise more difficult content, e.g. large environment models or vegetations. In future work, solutions need to be investigated which allow decomposing all elements of a scene adequately, so that gaze can be analyzed according to the key features of major impact. Particular extensions we are considering, are hierarchical decompositions of difficult objects, like for example trees or houses, and screen-space

approaches to subdivide large models or background into regions in such a manner that features with a different response on visual attention can be spatially separated.

8.3 Future work

There are many ways in which the basic pipeline presented in this paper can be improved. We propose the following application-oriented directions:

The first is to test the approach for more complex game settings and environments, imposing challenging tasks and missions, comparable to commercial games.

Second, to investigate certain extensions to improve the performance of the predictors. This includes, on the one hand, to account also for bottom-up mechanisms and to investigate how we can effectively combine the image-space-oriented saliency maps and the object-space-oriented importance maps. On the other hand, it is also important to account for the entire multimodal stimulus presented to the player of an action game. Notably, including the impact of sound sources in 3D space and elements in the head-up-display (HUD) of games. Due to the very general design, our system is capable to be extended with items for sound sources and HUD elements.

And third, the exploration of target applications for gaze predictors. Apart from selective rendering, we think that there are many other potential applications which are unexplored, or even unexpected, yet. For example, the prediction of the attentional response to advertisements in gaming environments may be a useful application. The idea would be to use the gaze predictor to estimate when attention is directed to, or distracted from, a particular advertising element of a game. This could enable online game vendors to develop more sophisticated costing mechanisms for their advertising clients.

Finally, we think that many of the tools we proposed, in particular computing gaze distributions according to object semantics in interactive 3D environments, may find great application in the study of human behavior in virtual reality.

Acknowledgments

The authors wish to thank Chris Chiu and the Computer Graphics Club at that Vienna University of Technology who have developed the Penta-G game, used in our study, and Erwin Starl and Matthias Pluch for their great assistance during preparation and conduction of the study. In addition, we would like to thank George Drettakis, Olivier Warusfel and all other people involved in the Crossmod project, for their invaluable scientific input throughout this work. This work was funded by the EU (project no. FP6-14891) and the Austrian Science Fund (project no. P-21130). Finally, we are grateful to the anonymous reviewers for their valuable suggestions and comments.

REFERENCES

BAYLIS, G. AND DRIVER, J. 1993. Visual attention and objects: evidence for hierarchical coding of location. *Journal of Experimental Psychology: Human Perception and Performance 19,* 3 (June), 451–470.

BEHRMANN, M., ZEMEL, R., AND MOZER, M. 1998. Object-based attention and occlusion evidence from normal participants and a computational model. *Journal of Experimental Psychology: Human Perception and Performance 24*, 1011–1036.

BITTNER, J., WIMMER, M., PIRINGER, H., AND PURGATHOFER, W. 2004. Coherent hierarchical culling: Hardware occlusion queries made useful. *Computer Graphics Forum 23,* 3 (Sept.), 615–624. Proceedings Eurographics 2004.

CANOSA, R. L., PELZ, J. B., MENNIE, N. R., AND PEAK, J. 2003. High-level aspects of oculomotor control during viewing of natural-task images. In *Human Vision and Electronic Imaging VIII. Proceedings of the SPIE Conference*, B. E. Rogowitz and T. N. Pappas, Eds. Vol. 5007. IS&T / SPIE, 240–251.

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

CATER, K., CHALMERS, A., AND WARD, G. 2003. Detail to attention: Exploiting visual tasks for selective rendering. In *Proceedings of the 14th Eurographics Workshop on Rendering*. EuroGraphics Association, 270–280.

CERF, M., FRADY, E. P., AND KOCH, C. 2008. Using semantic content as cues for better scanpath prediction. In *ETRA '08: Proceedings of the 2008 symposium on Eye tracking research & applications*. ACM, New York, NY, USA, 143–146.

COLLEWIJN, H., STEINMAN, R. M., ERKELENS, C. J., PIZLO, Z., AND VAN DER STEEN, J. 1992. Effect of freeing the head on eye movement characteristics during three dimensional shifts of gaze and tracking. In *The Head-Neck Sensory Motor System*, A. Berthoz, P. P. Vidal, and W. Graf, Eds. Oxford University Press, 412–418.

DE GRAEF, P., CHRISTIAENS, D., AND D'YDEWALLE, G. 1990. Perceptual effects of scene context on object identification. *Psychological Research 52,* 4, 317–29.

DUCHOWSKI, A. T. 2003. *Eye tracking methodology: Theory and practice*. Springer, New York.

DUNCAN, J. 1984. Selective attention and the organization of visual information. *Journal of experimental psychology. General 113,* 4 (December), 501–517.

EINHÄUSER, W., SPAIN, M., AND PERONA, P. 2008. Objects predict fixations better than early saliency. *Journal of Vision 8,* 14 (11), 1–26.

EL-NASR, M. S. AND YAN, S. 2006. Visual attention in 3d video games. In *ACE'06: Proceedings of the 2006 ACM SIGCHI international conference on Advances in computer entertainment technology*. ACM, New York, NY, USA, 22.

ELAZARY, L. AND ITTI, L. 2008. Interesting objects are visually salient. *Journal of Vision 8,* 3:3 (Mar), 1–15.

HAYHOE, M. M., SHRIVASTAVA, A., MRUCZEK, R., AND PELZ, J. B. 2003. Visual memory and motor planning in a natural task. *Journal of Vision 3,* 1 (2), 49–63.

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

HENDERSON, J., WILLIAMS, C., CASTELHANO, M., AND FALK, R. 2003. Eye movements and picture processing during recognition. *Perception and Psychophysics 65,* 5 (July), 725–34.

HENDERSON, J. M., WEEKS, P. A., AND HOLLINGWORTH, A. 1999. The Effects of Semantic Consistency on Eye Movements During Complex Scene Viewing. *Journal of Experimental Psychology: Human Perception and Performance 25*, 210–228.

ITTI, L. AND BALDI, P. 2006. Bayesian surprise attracts human attention. In *Advances in Neural Information Processing Systems, Vol. 19 (NIPS*2005)*. MIT Press, Cambridge, MA, 547–554.

ITTI, L. AND KOCH, C. 2000. A saliency-based search mechanism for overt and covert shifts of visual attention. *Vision Research 40,* 10-12 (May), 1489–1506.

ITTI, L. AND KOCH, C. 2001. Computational modelling of visual attention. *Nature Reviews Neuroscience 2,* 3 (Mar), 194–203.

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

JAMES, W. 1890. *The Principles of Psychology, Vol. 1*. Dover Publications.

JIE, L. AND CLARK, J. J. 2007. Game design guided by visual attention. In *Entertainment Computing, ICEC 2007*, L. Ma, M. Rauterberg, and R. Nakatsu, Eds. Lecture Notes in Computer Science, vol. 4740. Springer, 345–355.

KENNY, A., KOESLING, H., DELANEY, D., MCLOONE, S., AND WARD, T. 2005. A preliminary investigation into eye gaze data in a first person shooter game. In *19th European Conference on Modelling and Simulation*. ECMS, 733–740.

KOCH, C. AND ULLMAN, S. 1985. Shifts in selective visual attention: towards the underlying neural circuitry. *Human Neurobiology 4*, 219–227.

KOMOGORTSEV, O. AND KHAN, J. 2006. Perceptual attention focus prediction for multiple viewers in case of multimedia perceptual compression with feedback delay. In *ETRA '06: Proceedings of the 2006 symposium on Eye tracking research & applications*. ACM, New York, NY, USA, 101–108.

LAND, M., MENNIE, N., AND RUSTED, J. 1999. The roles of vision and eye movements in the control of activities of daily living. *Perception 28,* 11, 1311–1328.

LEE, S., KIM, G. J., AND CHOI, S. 2007. Real-time tracking of visually attended objects in interactive virtual environments. In *ACM Symposium on Virtual Reality Software and Technology*. ACM, 29–38.

LUEBKE, D., HALLEN, B., NEWFIELD, D., AND WATSON, B. 2000. Perceptually driven simplification using gaze-directed rendering.

MACK, J. AND ROCK, I. 1998. *Inattentional Blindness*. The MIT Press, Cambridge, MA.

MARMITT, G. AND DUCHOWSKI, A. T. 2002. Modeling visual attention in VR: Measuring the accuracy of predicted scanpaths. In *Eurographics 2002, Short Presentations*. EuroGraphics Association, 217–226.

MURPHY, H. AND DUCHOWSKI, A. 2001. Gaze-contingent level of detail rendering. In *Proceedings of EuroGraphics 2001 (Short Papers)*. EuroGraphics Association.

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

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

OLIVA, A., TORRALBA, A., CASTELHANO, M. S., AND HENDERSON, J. M. 2003. Top-down control of visual attention in object detection. In *Proceedings of the IEEE Int'l Conference on Image Processing (ICIP '03)*. IEEE.

PALMER, S. E. 1999. *Vision science: Photons to phenomenology*. MIT Press, Boston.

PARKHURST, D., LAW, K., AND NIEBUR, E. 2002. Modeling the role of salience in the allocation of overt visual attention. *Vision Research 42,* 1 (January), 107–123.

PELZ, J. B. AND CANOSA, R. 2001. Oculomotor behavior and perceptual strategies in complex tasks. *Vision Research 41*, 3587–96.

PETERS, R. J. AND ITTI, L. 2007. Beyond bottom-up: Incorporating task-dependent influences into a computational model of spatial attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 1–8.

PETERS, R. J. AND ITTI, L. 2008. Applying computational tools to predict gaze direction in interactive visual environments. *ACM Transactions on Applied Perception 5,* 2, 1–19.

PETERS, R. J., IYER, A., ITTI, L., AND KOCH, C. 2005. Components of bottom-up gaze allocation in natural images. *Vision Research 45,* 8 (Aug), 2397–2416.

ROTHKOPF, C. A., BALLARD, D. H., AND HAYHOE, M. M. 2007. Task and context determine where you look. *Journal of Vision 7,* 14, 1–20.

SALTHOUSE, T. A., ELLIS, C. L., DIENER, D. C., AND SOMBERG, B. L. 1981. Stimulus processing during eye fixations. *Journal of Experimental Psychology: Human Perception and Performance 7,* 3, 611–623.

SCHOLL, B. J. 2001. Objects and attention: the state of the art. *Cognition 80,* 1-2 (June), 1–46.

SIMONS, D. J. AND CHABRIS, C. F. 1999. Gorillas in our midst: Sustained inattentional blindness for dynamic events. *Perception 28*, 1059–74.

STARKER, I. AND BOLT, R. A. 1990. A gaze-responsive self-disclosing display. In *CHI '90: Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, New York, NY, USA, 3–10.

SUNDSTEDT, V., DEBATTISTA, K., LONGHURST, P., CHALMERS, A., AND TROSCIANKO, T. 2005. Visual attention for efficient high-fidelity graphics. In *SCCG '05: Proceedings of the 21st spring conference on Computer graphics*. ACM, New York, NY, USA, 169–175.

SUNDSTEDT, V., GUTIERREZ, D., ANSON, O., BANTERLE, F., AND CHALMERS, A. 2007. Perceptual rendering of participating media. *ACM Transaction on Applied Perception 4,* 3, 15.

SUNDSTEDT, V., STAVRAKIS, E., WIMMER, M., AND REINHARD, E. 2008. A psychophysical study of fixation behavior in a computer game. In *Proceedings of the 5th Symposium on Applied Perception in Graphics and Visualization*, S. Creem-Regehr and K. Myszkowski, Eds. ACM, 43–50.

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

VAN ZOEST, W. AND DONK, M. 2004. Bottom - up and top - down control in visual search. *Perception 33*, 927–937.

WOLFE, J. 2000. *Seeing*, 2nd ed. Handbook of Perception and Cognition. Academic Press, Chapter 8, 335–386.

WOLFE, J. M. 1994. Guided search 2.0: A revised model of visual search. *Psychonomic Bulletin and Review 1,* 2, 202–238.

WOLFE, J. M. 2007. Guided search 4.0: Current progress with a model of visual search. In *Integrated Models of Cognitive Systems*, W. Gray, Ed. Oxford University Press, 99–119.

YARBUS, A. L. 1967. Eye movements during perception of complex objects. In *Eye Movements and Vision*. Plenum Press, New York, 171–196.

A. APPENDIX

A.1 Binning the distribution of importance values

Binning in screen-space. Using N bins with a size of $\Delta b = (1+d)/N, d = +0$ [4], the value for bin $i \in \{1, \ldots, N\}$ is computed by:

$$D^{'}_{pix}[i] = \sum_{fix \in Fix} \int_{t_\alpha(fix)}^{t_\omega(fix)} \frac{|\{pixel \in F_t | (i-1)\Delta b \leq I(pixel,t) < i\Delta b\}|}{|F_t|} \mathrm{d}t \tag{15}$$

Each bin accumulates the fraction of pixels F_t belonging to a visible object with an importance value ($I(pixel, t)$) falling into bin i at time t. The fraction is computed relative to the number of pixels covered by visible objects at time t.

Binning in object-space. In the same manner we compute the distribution of importance values in object space:

$$D^{'}_{obj}[i] = \sum_{fix \in Fix} \int_{t_\alpha(fix)}^{t_\omega(fix)} \frac{|\{o_j \in V_t | (i-1)\Delta b \leq I(o_j,t) < i\Delta b\}|}{|V_t|} \mathrm{d}t \tag{16}$$

In this case we normalize with the number of visible objects $|V_t|$.

Binning for fixated objects. The distribution of importance values among fixated objects is computed by:

$$D^{'}_{fix}[i] = \sum_{fix \in Fix} \int_{t_\alpha(fix)}^{t_\omega(fix)} \varphi_i(fix,t) \mathrm{d}t \tag{17}$$

with:

$$\varphi_i(fix,t) = \begin{cases} 1, & \text{if } (i-1)\Delta b \leq I(fix,t) < i\Delta b \\ 0, & \text{otherwise} \end{cases} \tag{18}$$

Normalization:. To obtain a unique scale we further normalize all distributions with their totals:

$$D[i] = \frac{D^{'}[i]}{\sum_{i=1}^{N} D'[i]} \tag{19}$$

[4] To avoid more equations, we add an infinitesimal delta to assure that $1.0 < N\Delta b$

A.2 Scripting Language

A.2.1 *Operators.* The syntax supports common standard operators, including arithmetic operators ($+$, $-$, $*$, $/$), comparison operators ($>$, $<$, $\geq$, $\leq$, $=$), which evaluate to a boolean value, and logical operators (disjunction: $\vee$, conjunction: $\wedge$, not-operator: $\neg$).

A.2.2 *Numerical properties.* Properties which have a numerical value are arithmetic operands and can be related by common arithmetic operators and comparison operators.

In our examples we used the following numeric properties:

Screen space bounding window:
is defined by the lower left corner (*bw_x*,*bw_y*) and the upper right corner (*bw_X*,*bw_Y*). For convenient usage we normalized the coordinates to [-1,1] in both dimensions; the center of the screen is located at (0,0).

velocity:
the velocity of an object in world space in units per second

size_x:
the width of the bounding window normalized to [0,aspectRatio]

size_y:
the height of the bounding window normalized to [0,1]

pixelCoverage:
the fraction of visible pixels. It is computed by dividing the number of visible pixels of a particular objects in the itembuffer by the number of all pixels ($= numVisPixels/(768 * 1024)$)

A.2.3 *Boolean operands.* All objects are assigned with one or more semantic properties. We choose to represent such properties as booleans. Hence, the value of a particular semantic property is TRUE, if it does apply to the object, and FALSE, if not. For simplicity reasons "*a*" corresponds to "*a* = TRUE", and " $\neg$*a*" to "*a* = FALSE".

The following boolean properties we employed in our examples correspond to the semantic categories depicted in Figure 12:

ceiling, *floor*,	*picture*,	*column*, *light*,	*soundingPanel*,	*loudspeaker*,
wall, *barrel*, *box*,	*decorLion*, *map*,	*door*,	*alarm*,	*enemy*, *weapon*

A.2.4 *Properties of the player.* Unlike scene object properties, which are locally persistent while a rule set is executed for one object, properties of the player are globally persistent during processing all objects in the current frame.

In the examples, the boolean property *isArmed* is used to reflect whether the player has a weapon, or not ($\neg$*isArmed*).

A.3 Script 1: Semantic

begin *Rules*

```
//assign an exclude property to background objects
```
if *ceiling* $\vee$ *floor* $\vee$ *wall* **then** *Exclude*

```
//define windows to estimate eccentricity
```
if $bw\_X > -0.05 \wedge bw\_x < 0.05 \wedge bw\_Y > -0.05 \wedge bw\_y < 0.05$ **then** *Ecc1*
if $bw\_X > -0.15 \wedge bw\_x < 0.15 \wedge bw\_Y > -0.15 \wedge bw\_y < 0.15$ **then** *EccWindow2*
if $bw\_X > -0.3 \wedge bw\_x < 0.3 \wedge bw\_Y > -0.3 \wedge bw\_y < 0.3$ **then** *EccWindow3*
if $bw\_X > -0.7 \wedge bw\_x < 0.7 \wedge bw\_Y > -0.7 \wedge bw\_y < 0.7$ **then** *EccWindow4*

```
//subtract inner windows
```
if $\neg$*Ecc1* $\wedge$ *EccWindow2* **then** *Ecc2*
if $\neg$*Ecc1* $\wedge$ $\neg$*EccWindow2* $\wedge$ *EccWindow3* **then** *Ecc3*
if $\neg$*Ecc1* $\wedge$ $\neg$*EccWindow2* $\wedge$ $\neg$*EccWindow3* $\wedge$ *EccWindow4* **then** *Ecc4*
if $\neg$*Ecc1* $\wedge$ $\neg$*EccWindow2* $\wedge$ $\neg$*EccWindow3* $\wedge$ $\neg$*EccWindow4* **then** *Ecc5*

end

begin *Selection*

Ecc1,*Ecc2*,*Ecc3*,*Ecc4*,*Ecc5*,*Exclude*

end

A.4 Script 2: Eccentricity

begin *Rules*

if *ceiling* $\vee$ *floor* $\vee$ *wall* **then** *Exclude*

```
//if the door is in motion it is opening
```
if *door* $\wedge$ *velocity* > 0 **then** *OpeningDoor*

```
//we have sliding doors which vanish into the walls when they open.
//so we assume that highly occluded doors have been opened
```
if $pixelCoverage < 0.1 * size\_x * size\_y$ **then** *OpenDoor*

```
//if a door is neither opening nor opened, it can only be closed
```
if $\neg$(*OpeningDoor* $\vee$ *OpenDoor*) $\wedge$ *door* **then** *ClosedDoor*

end

begin *Selection*

Player.isArmed,
barrel, *box*, *decorLion*, *picture*, *map*, *column*,
OpeningDoor, *OpenDoor*, *ClosedDoor*,
alarm, *speakerbox*, *soundingPanel*,
enemy, *explosion*, *weapon*
Exclude

end

A.5 Script 3: Semantic & Eccentricity

begin *Rules*

if *ceiling* $\vee$ *floor* $\vee$ *wall* **then** *Exclude*

if *door* $\wedge$ *velocity* > 0 **then** *OpeningDoor*
if *pixelCoverage* $< 0.1 * size\_x * size\_y$ **then** *OpenDoor*
if $\neg($ *OpeningDoor* $\vee$ *OpenDoor* $) \wedge$ *door* **then** *ClosedDoor*

```
//there is only one weapon !
//if the player is armed he bears the weapon
```
if *Player.isArmed* $\wedge$ *weapon* **then** *BearedWeapon*

```
//if the player is unarmed the weapon is a search target
```
if $\neg$*Player.isArmed* $\wedge$ *weapon* **then** *WeaponTarget*

```
//define clusters:
```
if *explosion* $\vee$ *enemy* $\vee$ *WeaponTarget* $\vee$ *soundingPanel* **then** *VeryImportant*

if $\neg$*Player.isArmed* $\wedge$ (*map* $\vee$ *column*) **then** *Important*
if *OpeningDoor* $\vee$ *ClosedDoor* $\vee$ speakerbox **then** *Important*

if *box* $\vee$ *barrel* $\vee$ *alarm* **then** *Normal*
if $\neg$*Player.isArmed* $\wedge$ (*decorLion* $\vee$ *picture*) **then** *Normal*
if *Player.isArmed* $\wedge$ *column* **then** *Normal*

if *Player.isArmed* $\wedge$ (*picture* $\vee$ *map* $\vee$ *decorLion*) **then** *Unimportant*
if *OpenDoor* $\vee$ *BearedWeapon* $\vee$ *light* **then** *Unimportant*

```
//define eccentricity properties
```
if $bw\_X > -0.05 \wedge bw\_x < 0.05 \wedge bw\_Y > -0.05 \wedge bw\_y < 0.05$ **then** *Ecc1*
if $bw\_X > -0.15 \wedge bw\_x < 0.15 \wedge bw\_Y > -0.15 \wedge bw\_y < 0.15$ **then** *EccWindow2*
if $bw\_X > -0.3 \wedge bw\_x < 0.3 \wedge bw\_Y > -0.3 \wedge bw\_y < 0.3$ **then** *EccWindow3*
if $bw\_X > -0.7 \wedge bw\_x < 0.7 \wedge bw\_Y > -0.7 \wedge bw\_y < 0.7$ **then** *EccWindow4*
if $\neg$*Ecc1* $\wedge$ *EccWindow2* **then** *Ecc2*
if $\neg$*Ecc1* $\wedge$ $\neg$*EccWindow2* $\wedge$ *EccWindow3* **then** *Ecc3*
if $\neg$*Ecc1* $\wedge$ $\neg$*EccWindow2* $\wedge$ $\neg$*EccWindow3* $\wedge$ *EccWindow4* **then** *Ecc4*
if $\neg$*Ecc1* $\wedge$ $\neg$*EccWindow2* $\wedge$ $\neg$*EccWindow3* $\wedge$ $\neg$*EccWindow4* **then** *Ecc5*

end

begin *Selection*

VeryImportant, *Important*, *Unimportant*,
Ecc1, *Ecc2*, *Ecc3*, *Ecc4*, *Ecc5*,
Exclude

end