Gaze Prediction using Machine Learning for Dynamic Stereo Manipulation in Games

George Alex Koulieris* Technical University of Crete INRIA

George Drettakis† INRIA

Douglas Cunningham‡ Technical University of Cottbus

Katerina Mania§ Technical University of Crete

Figure 1: We propose a gaze predictor (a), used to perform localized stereo grading (b). We compare to no stereo grading (c) and OSCAM [Oskam et al. 2011] (d). We provide rich and comfortable depth. Please use red/cyan anaglyph glasses; best viewed on a monitor.

Abstract

Comfortable, high-quality 3D stereo viewing is becoming a requirement for interactive applications today. Previous research shows that manipulating disparity can alleviate some of the discomfort caused by 3D stereo, but it is best to do this locally, around the object the user is gazing at. The main challenge is thus to develop a gaze predictor in the demanding context of real-time, heavily task-oriented applications such as games. Our key observation is that player actions are highly correlated with the present state of a game, encoded by game variables. Based on this, we train a classifier to learn these correlations using an eye-tracker which provides the ground-truth object being looked at. The classifier is used at runtime to predict object category – and thus gaze – during game play, based on the current state of game variables. We use this prediction to propose a dynamic disparity manipulation method, which provides rich and comfortable depth. We evaluate the quality of our gaze predictor numerically and experimentally, showing that it predicts gaze more accurately than previous approaches. A subjective rating study demonstrates that our localized disparity manipulation is preferred over previous methods.

Keywords: Gaze Prediction, Stereo Grading, Perception

Index Terms: I.3.3 [Computer Graphics]: Three-Dimensional Graphics and Realism—Display Algorithms;

1 Introduction

Comfortable, high-quality stereo 3D is an important and timely requirement for real-time applications, especially given recent popularity of commodity Head Mounted Displays (HMDs). It is well known that 3D stereo viewing often results in discomfort and eye fatigue, most commonly because of excessive disparities and the vergence-accommodation conflict [21]. To counter these problems, recent methods [38, 32] manipulate stereo content for more comfortable viewing, a process called *stereo grading*. Focusing these changes in the region of the user's attention has been shown to be particularly effective [1], corresponding to the fact that human stereo is based on where we look. Locally adapting stereo in attended regions is thus important for real-time applications such as games. We enable attention-driven stereo manipulation for such applications, by addressing two challenges: effective real-time gaze prediction and dynamic stereo grading to improve depth perception while maintaining stereo comfort.

Previous work has shown that in interactive applications the *task* strongly influences gaze and more generally attention (e.g., [50, 15, 27]). Existing solutions for high-level gaze predictors require *manual categorization* of objects in relation to the context of the scene [27] or tasks [50], which is time consuming and impractical in our context of interactive Virtual Environments (VEs) such as games. Previous disparity mapping operators based on image-based saliency estimates are off-line [32] or when they are interactive, apply a global disparity transformation over the entire scene [38]. Such manipulations often sacrifice depth, resulting in "flat" imagery (known as the cardboarding effect [11]).

Since players have full control in a game, it is hard, if not impossible, to accurately guess where a player is looking at any given instant without knowing their current goal. Our key observation is that a player's current goal is highly correlated with the present state of the game, as *encoded by game variables*. For instance, in a Virtual Reality (VR) shooting game, the player's current goal will be related to the health and ammo of his/her character and of the enemies' movements. Based on this insight, we train a classifier to learn the correlation between game variables and object class the user looks at, using eye-tracking ground-truth data recorded during a training session. The resulting classifier can predict object category – and thus user gaze – for any subsequent game-play. Our approach is automatic since it uses machine learning to build the classifier, avoiding the need for manual object tagging and/or explicit definition of objects important to a task. Using our online gaze predictor we introduce dynamic and localized disparity manipulation, which provides high-quality depth information in a scene without sacrificing comfort.

In summary we make two primary contributions:

- We design a machine learning gaze predictor based on game

*galekoul@gmail.com
†george.drettakis@inria.fr
‡cunningh@tu-cottbus.de
§k.mania@ced.tuc.gr

variables and eye tracking data from a video game.

- We develop a stereo grading algorithm based on gaze for dynamic disparity management in video games.

We evaluate the quality of our gaze predictor using standard machine learning tools and by a confirmatory eye-tracking study. This study shows that our predictor is more accurate than previous alternatives in the context of task-driven activities such as game-play, and that our stereo grading method is preferred over previous methods in subjective ratings.

2 Related Work

There is a large body of work on attention/gaze-based graphics and stereo viewing. Overviews can be found in corresponding surveys [37, 23, 5]; we review literature most relevant to our work.

2.1 Attention and Gaze in Graphics

The gaze point of an observer can be either determined directly via eye-tracking or estimated by incorporating an attention model.

Attention and Gaze. An eye tracker has been employed to monitor fixations for gaze-directed rendering, allowing 3D model geometry to be simplified more aggressively in the periphery than at the center of user gaze [33, 18]. However, real-time eye tracking is rarely available for common applications such as video games. In addition, stereo rendering constrains attention modeling [8]. When attending a specific depth due to disparity, objects in other depth planes are not attended to, and attention shifts faster to nearby objects than to objects deep into the scene [19]. A probabilistic approach using Bayesian inference has been proposed to accurately correlate gaze data to scene objects [2]. In our game scenario gaze to object mapping techniques did not prove necessary, since our game objects occupy a quite large viewing angle in the view frustum. However, in a game setup populated by hundreds of smaller objects, gaze to object mapping methods could be employed.

Early models estimate attention based on low level image features such as contrast, luminance and motion [22]. However, low level saliency itself cannot predict fixations when there is an overt or covert task in an environment, e.g., when observers have to actively ignore low-level features [41]. Several methods have been proposed to take task into account, usually restricted to a specific task [10, 51, 41, 50], where the objects attended were manually pre-defined. Recent context-based high-level attention predictors require intense manual pre-processing in terms of tagging to define object semantics and/or high-level scene descriptions (schemas), restricted to scenes with static objects [27, 28]. The scene graph representation of scenes was used to map gaze positions to objects and object semantics [49]. Importance maps generated from offline eye tracking data were used as a heuristic to predict user attention according to object properties present at runtime [3]. Both approaches require manually pre-defined task objects and task objectives. Eye tracking data has shown that depending on the type of game, players may mainly attend the center of the screen (First Person Shooters, FPSs), or search the entire screen (Action-Adventure) [15].

Our approach to gaze prediction using game state enables us to overcome the limitations of these previous methods. We use machine learning to *automatically* learn gaze patterns for different object categories and tasks without manual tagging, and accurately predict gaze including for complex task-dependent situations which would be very hard to encode explicitly.

Video Game State Variables. Game structure is defined during the design phase of a video game, and is used to represent relationships between objects and player actions (obstacles to overcome etc.). Structure is represented in the source code via variables, which are used to represent commands and storage needs [12]. For example, a player in a typical shooting game may be described by two vectors indicating location and rotation in the game scene, a value indicating health and a value indicating available ammo. The value of these variables in relation to the location or availability of ammo as well as to the location of an Artificial Intelligence (AI) Non-Playable Character (NPC) influences the behaviour of the player, e.g., whether he will run away from an enemy or engage in close combat. A key idea of our work is that the players' behavior is related to the game state, and their gaze or attention will be related to their behavior. Consequently, by automatically analyzing exposed variables of a video game, we can predict where they are looking.

2.2 Machine Learning in Games

Machine learning algorithms have been used in video games to accomplish a more believable, variable and challenging AI [29]. In commercial video games it has been employed both to learn at design-time, where the results of machine learning are applied before publishing the game and to learn at runtime, for an individually customized game experience. For example, *LiveMove*™ is a machine learning tool recognizing motion and converting it to gameplay actions to train a computer opponent. Another example is *Black and White*™ where the player's pet learns what to do in the game via reward and punishment.

A pre-defined set of static photographs was used to implicitly model high-level saliency via machine learning techniques applied to eye-tracking data [25]. As an alternative to standard machine learning methods, a prototype self-refining fluid dynamics game that learns from crowd-sourced player data has been proposed [47], concentrating computation in states the user will most likely encounter to improve simulation quality.

In this work we use Decision Forests (DFs), which provide powerful multi-label classification [7] and support our goal of predicting object class the user looks at, based on game state in real-time while a player is actively involved in game-play. Games involve tasks that can be used to predict attention regardless of genre, although this is not usually the case throughout the entire gameplay. Even the most complex gameplays have parts with active motion through a complex 3D environment. Whenever there is a goal-oriented task in a game our method can be applied. For example, in the popular Action-Adventure game *Watch Dogs*™, gameplay alternates between action sequences similar to FPS combat sequences (either by using guns or melee contact) and solving mathematical and technical quizzes to hack into computer systems. RPGs such as *Skyrim*™, display sequences where the player interacts with NPCs -though not always to shoot them- and objects to achieve a goal.

2.3 Stereoscopic Viewing Overview

In this section the basic geometry of stereoscopic vision relevant to this work is presented [54]. In stereoscopic rendering, to induce stereo perception, each eye obtains its own view rendered with a slightly offset camera location. The virtual screen is then perceived on the intersection of the left and right frusta. A stereo projection matrix is defined as a horizontally offset version of the regular monoscopic projection matrix, both offsetting for the left and right eyes along the X axis. The projection axes should be parallel in order to avoid a converged configuration that introduces keystone distortions into the image, which can produce visual discomfort [48].

We use the standard asymmetric viewing frusta, as presented by Woods et al. [54] shown in Fig. 2.

The two cameras are symmetrically offset from the origin of x-axis at points L and R. The separation LR between them is D_{eye}. The cameras are directed parallel to one another, looking down z-axis. D_{near} is the near clipping distance and C is the the distance between the camera and the perceived plane of focus, known as convergence distance. The left and right extremities of the virtual screen lie at points A and B respectively.

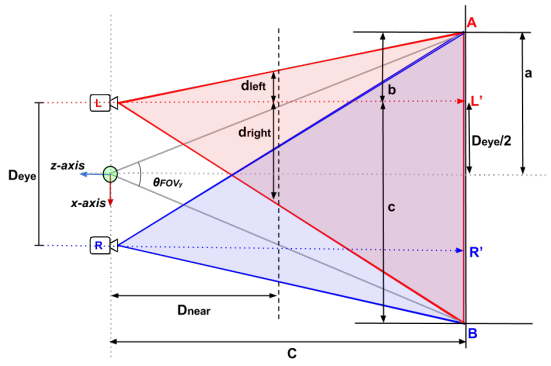


Figure 2: Asymmetric frustum stereo geometry.

2.4 Visual Discomfort and Fatigue in Stereo Viewing

Viewer fatigue due to the vergence - accommodation conflict is common when viewing stereoscopic 3D content. The conflict is caused because the plane of focus (i.e., the screen) is fixed whereas eye vergence movements continuously occur when fusing stereoscopic content. Large stereoscopic disparities in video games further increase visual fatigue [21, 31]. Symptoms range from an insensible overload of the visual system or slight discomfort that can cause major eye strain, provoke visually-induced headaches and lead to total loss of depth perception [30]. The level of discomfort increases with the exposure time to 3D content which is not optimized for comfortable viewing.

High disparities force the eyes to rotate unnaturally in relation to each other. The standard tolerated disparity threshold is 24 arcmin between the same point on two retinas [24] which however reproduces very small perceivable depths. Disparities can go well above this low threshold, however, visual discomfort may build up which should be avoided.

A solution to these issues is the stereo-grading process, i.e., altering the depth structure of a scene by drawing objects in a user's comfortable disparity range, also known as the *comfort zone* of the observer [44]. When objects are drawn in the comfort zone, clear and correctly fused binocular vision is achieved and discomfort is minimized [44]. Such approaches have been developed for interactive stereoscopic applications or film, attempting to match the depths between cuts [52] or compress the depths of a scene [32, 38, 9]. A perceptual disparity metric that can compare one stereo image to another to assess the magnitude of the perceived disparity change has been proposed [13]. Universal depth compression may lead to limited depth perception or the cardboarding effect [36, 11].

Interactive VEs such as video games accentuate the vergence - accommodation conflict since the viewer is not simply gazing at a virtual 3D space but is also interacting with it, altering the distance from objects in real time [17]. A virtual object closer to the observer than the in-focus depth plane exerts strong negative (crossed) disparity that may result in uncomfortable viewing, eye strain and diplopia. This discomfort in interactive scenes is due to both the selected disparity parameters and the lateral or in-depth motion of the objects [24, 14].

Our solution introduces *dynamic and localized* disparity management applied to 3D video games, for attended objects or areas based on the current task. Our approach smoothly relocates the perceived depth of attended objects/areas into the comfort zone, maintaining a rich sense of depth.

3 Gaze Prediction using Machine Learning

Our machine learning approach has three steps: identification of important game variables and object classes, data collection and classifier training. We used the *RFPS Toolkit*^{TM} from the Unity3D^{TM} Asset Store, which was one of the most realistic free game levels available to demonstrate our approach. Screenshots of the game are shown in Fig. 3 and 4.

3.1 Identifying Important Variables and Object Classes

We both investigated variable range and employed a high pass filter on variable derivatives to measure their variation. We run the filter on all internal variables of the game as well as agent location/distance variables exposed by the game AI (Fig. 3). In our example game, the total number of game variables was over 300. Similar to other machine learning algorithms that perform dimensionality reduction, we ignored the variables that exhibited little variability [43], focusing on the most informative 5% of the variables; a fraction selected from a pilot data acquisition process which indicated that the rest of the variables did not vary significantly in time. The feature vector thus consists of 13 game variables. These can be seen in Table 1; e.g., the variables Robot_{dx}, Robot_{dy}, Robot_{dz} encode the distance to the closest robot. All variables thus have valid values at any given time.

Prop_{dx}	Robot_{dx}	NPC_{dx}	Health	Ammo
Prop_{dy}	Robot_{dy}	NPC_{dy}	Hunger	
Prop_{dz}	Robot_{dz}	NPC_{dz}	Thirst	

Table 1: The most informative variables that were selected for data collection. dx, dy, dz variables denote distances from the closest Prop (gun, knife, etc), non-playable-characters or robots.

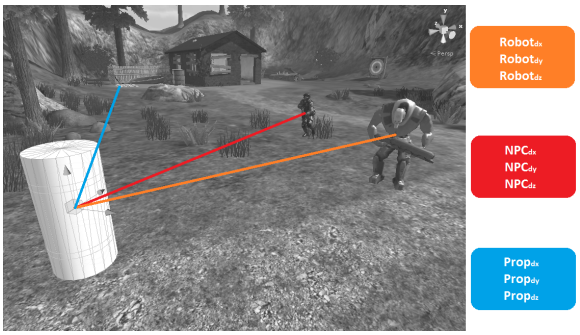


Figure 3: Distance vectors exposed by game AI were recorded.

To determine object classes, we parsed the game scene hierarchy generating a set Λ of object categories or *class labels* used for training. Parsing was possible since game objects often follow standard naming conventions [16]. The naming scheme for the assets of the toolkit used in our implementation was "Identifier - Modifier - Variant - Footprint - Optical Distinction". For example "Tree broadLeaved 01 2x2 Green". We exploited this scheme by employing a 3D model name parser that inferred abstract object classes from object names. Using this approach, 25 categories were found (Table 2).

The data collection step uses eye-tracking to identify the correlation between the feature vector and the class labels, based on the object class being attended given the current state of the game.

3.2 Data Collection

Data collection involves an experimental setup using an HMD, equipped with an eye-tracker.

FallenLog	Boat	WoodFence	Fence	Can
Ammo	Barrels	Brickhouse	Crate	Door
Rock	Tree	Water Pickable	Woodboard	Pond
Platform	Elevator	Robot	Soldier	Bush
Zombie	Mine	Food Pickable	Gun Pickable	

Table 2: Automatically extracted class labels

Stimuli. We modified the game level to have 60-90 seconds of game-play for data collection. The players are required to reach a flaming spaceship while avoiding threatening robots, soldiers, zombies and mines (Fig. 4). During data collection, the starting position of the player and the spaceship were all similar for every player and trial. An equal number of robots, soldiers, zombies and mines were spawned in random locations in the level during gameplay, ensuring necessary variability in the stimuli, and a dense dataset of possible fixation patterns.

Figure 4: The player must reach the flaming spaceship while avoiding soldiers, zombies, robots and mines.

Pilot Study. A pilot study indicated that due to differences in individual performance, it was best to fix play time to at least 20 minutes rather than fix the number of trials. We use a speed based sample rate, with a low rate of 5 samples/second, which was increased linearly when the user moved faster through the environment. This allowed a reliable sampling of the obstacle configuration space.

Participants. Ten people participated in the study (2 female, mean age 25). We selected only experienced FPS gamers since our goal is to provide a stereo optimizer for gamers, rather than general VE navigation. All participants played a training level to (i) subjectively verify that participants were indeed experienced gamers, and (ii) familiarize the participants with input controls and the VE. To avoid a training effect for a search task, participants were instructed to locate a spacecraft during the training session. Participants were given candy as compensation.

Apparatus and Procedure. The stimuli were displayed on a NVisor^{TM} SX111 HMD, which has stereo SXGA resolution and a FoV of 102 degrees horizontal by 64 degrees vertical. Participants navigated through the VE using a keyboard and mouse to simulate the FPS input paradigm; the HMD head tracker was disabled. Experienced gamers had no trouble controlling the game with standard *WASD keys* despite the keyboard been occluded by the HMD. Eye tracking information was recorded using a twin-CCD binocular eye-tracker by Arrington Research^{TM}, which was attached to the HMD. The eye tracker was updated at a frequency of 30Hz. The FoV of the HMD was restricted to 47.8 degrees horizontally and 23 degrees vertically to simulate a 24" display placed at a 60cm observer distance [46]. This was necessary since the eye tracked HMD of our lab is a partial overlap HMD, which would otherwise force us to converge the virtual cameras. However, the central 50 degrees of its FoV are fully overlapped. By setting the horizontal FoV to be less then 50 degrees, no converging camera setup was necessary allowing us to correctly test our parallel camera stereo grading method. This is not a limitation of the method; employing a desktop eye tracker would yield similar results.

Procedure. All participants underwent a RANDOT stereo vision test [45] and an eye dominance test to select the dominant eye for eye tracking before proceeding with the main experiment. Then the standard Arrington Research^{TM} eye tracker calibration procedure was performed by each participant. We measured and set the correct Interpupillary Distance (IPD) for each subject and selected very conservative parameters for the stereo pair to obtain a fail-safe and comfortable stereo for all participants with minimal depth complexity. We collected 200 minutes of gameplay data for all participants.

Figure 5: The experimental setup.

During game-play, game state variables were recorded for every sample instance (e.g., $\overrightarrow{Robot}_{\mathrm{dx,dy,dz}} = (x,y,z)$). Eye-tracking fixations were used to identify the object via ray-casting and the object together with the game state were inserted into a database. Technical details on eye-tracking data de-projection can be found in the supplemental material.

At the end of data collection, our database contains training data T, having N samples of $M = 13$ features (Table 1), $T = (X_1, y_1), (X_2, y_2), ..., (X_N, y_N)$ that will be used to train the DF. Each record $\in T$ includes an input feature vector, $X_i = x_{i1}, x_{i2}, ..., x_{iM}$ and the object class label $y_i \in \Lambda$ (Table 2) indicated by the eye tracker at the specific moment that sample X_i was taken.

3.3 DF Generation, Training and Tuning

Following the experiment, data were extracted from the database and imported to the R statistical language [40] to optimize parameter selection for a DF classifier. We use standard tree growing and tuning procedures, as described in [7].

Frequent Classes Under-sampling. When initially processing the data we encountered a class imbalance issue. Since the participants mostly attended moving objects (soldiers, robots, etc.) in the environment, more samples for these objects were recorded. When a subset of the classes accounts for the majority of the data, the classifier achieves high accuracy by erroneously classifying all the observations into these most frequent classes. This gives high accuracy for frequent classes, but poor predictions for the least frequent ones. To partially compensate we randomly under-sampled frequently sampled classes to balance the data and then trained the model with this balanced data [6].

DF Validation and Tuning. In DFs, there is no need for cross-validation or a separate test set to get an unbiased estimate of the test set error in contrast to other ML methods [4]. The study of

error estimates for bagged classifiers in [6] indicates that the *Out-of-Bag* (OOB) estimate is as accurate as using a novel test set of the same size as the training set. Using the OOB error estimate removes the need for a set aside test set. Tuning is necessary for the forest to grow optimally in terms of OOB error. We validate test-time prediction accuracy experimentally in Sec. 5.1.

To estimate the OOB error [7], after creating the *ntree* classifier trees, we proceed for each (X_i, y_i) in the original training set T and select all T_k which do not include (X_i, y_i). This subset is a set of boostrap datasets which does not contain any record from the original dataset. This is known as the OOB set. There exist N such subsets, one for each data record in the original dataset T. The OOB classifier is the aggregation of votes only over T_k such that it does not contain (X_i, y_i). The OOB estimate for the generalization error is the error rate of the OOB classifier on the training set, compared to known y_i's. Simply put, the error rate for classification on the OOB portion of the data for each tree is recorded and the same is done after permuting all predictor variables. The difference between the two is then averaged over all trees, and then normalized by the standard deviation of the differences. The OOB error estimate is estimated internally, during forest generation.

The final, balanced set T' spanned $N = 55151$ samples $\times M = 13$ features + Class. We optimized the number of trees to make the OOB error rate converge in terms of a pre-selected error threshold. In the optimized dataset we only kept the object categories for which prediction error rate $< 45\%$ was achieved. The prediction error rate for the 8 object categories that fall behind this threshold can be seen in Table 3. The imbalance between sampling rates described previously explains why objects encountered less frequently (e.g., "Food") have a higher prediction error rate in the DF structure compared to frequently encountered objects (e.g., "Explosives"). The average DF OOB estimate of error rate was found to converge to 16.26% for 100 trees. The class imbalance issue for complex games can be amended by increasing the DF training samples. This is expected to increase the number of successfully predicted object categories.

Robot	Soldier	Zombie	Health Pack
7.1	11.2	19.9	20.9
Gun Pickable	Explosives	Ammo	Food Pickable
25	36.3	37.8	40.6

Table 3: Prediction error rate for each object category.

In-game DF Generation and Classification. We employed a custom-made Iterative Dichotomizer 3 (ID3) method in C# to generate a decision tree from the dataset [39]. As suggested in [7], each tree was grown using $mtry = \left\lfloor \frac{\log M}{\log 2} \right\rfloor$ random features/game variables for each split of the tree and we have confirmed that this value is the optimal splitting parameter in terms of OOB error.

The procedure yields *ntree* datasets of the same size as T, grown from a random re-sampling of data in T with-replacement, N times for each dataset. 64% of the data in T were used for the generation of each tree [7]. This results in $T_1, T_2, ..., T_{ntree}$ *bootstrap* datasets. For each T_i bootstrap dataset a tree is grown. To classify any new input data $D = x_1, x_2, \ldots, x_M$ we test them against each tree to produce *ntree* results $Y = y_1, y_2, ..., y_{ntree}$. For classification, the prediction for this data is the majority vote on this set (Fig. 6).

4 Dynamic Stereo Grading using Gaze Prediction

Now that we have a classifier that can predict object class and thus gaze based on game state, we can place these objects inside the comfort zone and as close to the plane of zero disparity possible, i.e., onto the virtual screen plane. We describe this dynamic stereo grading process next.

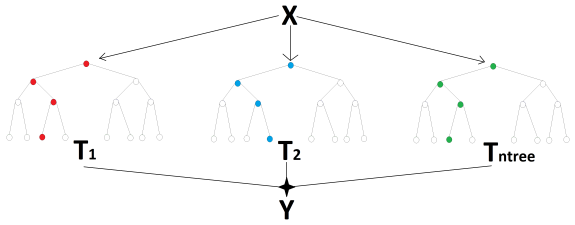


Figure 6: Each new vector instance is tested against each tree. The majority vote on this sample is the prediction of the DF.

Our system linearly interpolates camera separation and asymmetric frustum parameters to avoid visual artifacts and observers becoming aware of the change. We use the standard asymmetric viewing frusta, as presented in Section 2.3. Precise technical details of the stereo manipulation are presented in supplemental material. Compared to previous stereo grading algorithms, e.g., [38] our method performs *automatic localization* of the disparity manipulation.

The trained DF component of the stereo grader receives a game state variable vector as an input, and generates an object category prediction. On our test setup (Intel Core I7@3.4GHz, 8Gb RAM) a DF structure based on 100 decision trees generates a prediction every 500 milliseconds at runtime. The frame rate of the game was fixed to 60 frames per second; frustum parameters were thus readjusted every 30 frames. An optimized implementation of DFs would greatly increase classification throughput. After obtaining a prediction, the system searches for same-category objects in the viewing frustum, with three possible outcomes: A single, multiple or no objects of that category are found. If a single object is found, its distance to the camera is estimated via ray casting the depth buffer. Then asymmetric frustum parameters are estimated [54] that shift the zero-parallax plane and thus the comfort zone close to the barycenter of that object. The distance of the object from the zero-parallax plane is a parameter that defines how deep or shallow these objects are perceived. To estimate this parameter we take into account the object's bounding volume radius. If multiple objects of that category are found, the combined barycenter of these objects is estimated. The zero-parallax plane is then brought close to the novel barycenter. If no object of that category is found this indicates that the predictor has failed. To achieve fail-safe stereo, the zero parallax plane is brought close to the largest object adjacent to the center of the view frustum. A video capture of an interactive session can be seen in the paper video. [Link][https://www.youtube.com/watch?v=tjmOJ77ceck]

Optimizations. The system is more aggressive when grading negative disparities that cause more strain than positive disparities [37]. If the estimated negative disparity is larger than 3% of the distance of the virtual camera pair to the virtual plane, our method pushes the objects that are closest to the screen further back to minimize eye strain [37]. By employing this approach, the system manages to keep all important objects in terms of gaze inside the comfort volume. For large displays model parameters can be constrained to an upper positive disparity limit thus avoiding diverging eye movements.

We linearly interpolate all camera parameters in time so that changes are not perceived by an observer, similarly to [38]. However, OSCAM adjusts disparity in terms of the whole scene depth or by pre-determined *manually* selected depth ratios that are least-squares fit to the desired mapping. OSCAM's automatic fail-safe mode suffers from cardboarding when grading scenes with large

depths, as shown in the accompanying video. Our method generalizes OSCAM's by minimizing cardboarding *automatically*.

Discussion. When an attended object is very close to the virtual cameras, camera frustum parameters are changed to move the zero parallax plane forward, closer to the observer. The disparity is modified globally but since the observer is attending the object and the change is gradual obervers are not cognitively aware of the change [38, 24]; stereo grading does not affect fixations. However, if future research suggests otherwise, control of the effect may be given to the game designers for artistic purposes.

An extremely close up object never appeared in our game. The only objects that moved into view were enemies that were tracked by the machine learning component. If large objects suddenly appear in a different game setup or the attention predictor has to be temporarily deactivated to optimize performance, a set of rules similar to OSCAM may be selectively activated in conjunction to our method prohibiting large sudden disparities.

In terms of game complexity, introducing more enemies does not require additional training data. Once the model captures enemy mechanics, attention can be predicted regardless of their number. Class imbalance issues due to more training data can be solved trivially during training by excluding samples of the dominant classes.

5 Evaluation and Results

We experimentally validated the success of our predictor. We also compared its performance to a low-level saliency predictor and measured the perceived quality of our stereo grading compared to other approaches.

The validation experiment was split in 3 sessions with a 10-minute break between sessions to reduce eye strain. The 3 sessions were a pairwise comparison of Standard stereo (no disparity management), Ours and OSCAM: **a**: Standard <> Ours, **b**: Ours <> OSCAM, **c**: OSCAM <> Standard. We compare to the OSCAM method since it is the de facto standard and state-of-the-art stereo grading method and also could be directly compared to ours. Other stereo grading methods either are based-on/similar to OSCAM, only work offline [32], require eye-tracking at runtime [1], control cameras in real setups [20] or were developed for multi-focal displays [34]. We used our HMD's eye tracker to obtain gaze data only during session **a**. Ten participants not previously involved in any related experiment (2 female, mean age 23.5) completed the sessions successfully.

In each session players played 10 pairwise 10-second game rounds of predetermined gameplay, lasting in total for 200 seconds (10 pairs $\times$ 2 conditions $\times$ 10 sec). The order in each pair was randomized. Session pairs intentionally imposed large disparity changes: objects moving in the view frustum, camera wildly panning, etc. Example conditions were designed having in mind common *disparity events* that cause strain and affect depth perception in a game. For example, a moving object (e.g., an enemy) is about to appear, and will be far away in depth. Depths should be compressed in time to prepare the observer for the moving object to avoid intense convergence motion. A second example is an object that is expected to appear due to a lateral movement and which will introduce an extreme disparity. Another example is an enemy that is shooting and threatening the -devoid of ammo- player but the player will not look at the enemy, instead the player will search for ammo lying on the floor. Example video captures of interactive sessions can be seen in the supplemental video. [Link][https://www.youtube.com/watch?v=mMazlouUDjI]

5.1 DF-based Predictor Quality

We have already presented a validation of the training using OOB estimates in Section 3; here we perform test-time evaluation of our DF predictor using eye-tracking and compare to a state-of-the-art low-level predictor [53].

Figure 7: Visualization of successful attention predictions of our method with green beams. The blue beams indicate eye tracking data. The grey beams indicate failed predictions. The cyan colored squares indicate major frustum reconfigurations.

Est.	Object gazed	Hits
R	random object	$< 5\%$
Low	x,y,radius	44.1%
DF	object category	76.2%

Table 4: The ratio of frames for which the attended object was predicted correctly during session **a**.

During session **a**, low level (x,y) predicted coordinates, DF predictions and eye tracking data of the view frustum were obtained at a rate of 1Hz, 1Hz and 30Hz respectively. We only considered fixations of a duration greater than 300 milliseconds for our analysis [42]. Thus for every second we obtain up to 3 possible fixation locations. We also define a baseline estimator *R* that selects a random object in the same view frustum at 1Hz.

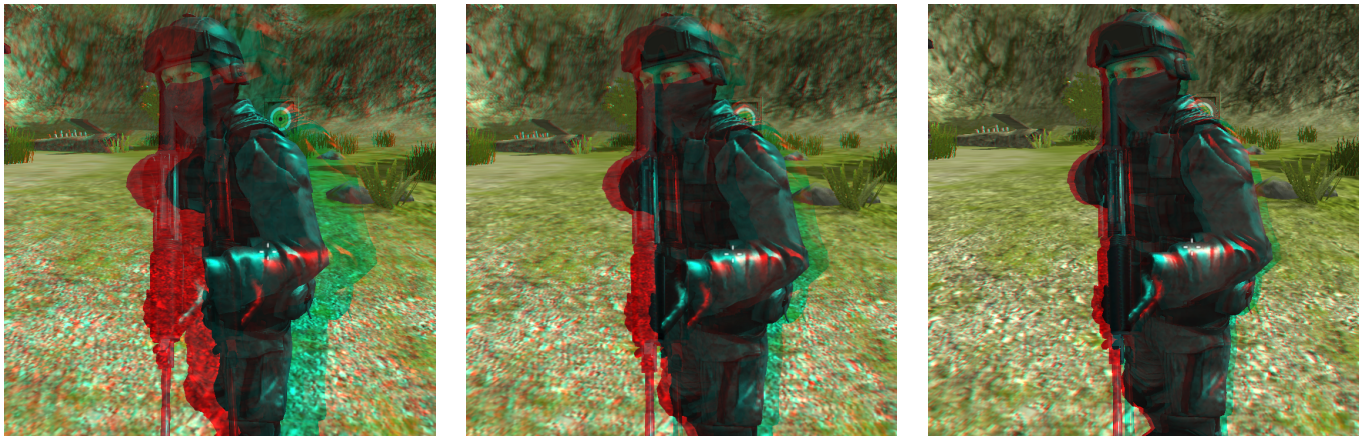
We perform a temporal window integration comparison for the 3 predictors (Low-level, DF and baseline). Since the sampling rate of all three compared predictors is 1Hz, if any of the user fixations within this one-second window are predicted correctly by a method, we consider that a hit. In particular, for the low level predictor, a prediction is considered a hit if the actual fixation lies inside a 128 pixel radius circle around the predicted x,y coordinates.

In our approach, if a fixation is on an object of a predicted category this is considered a prediction hit. We visualize our predictions compared to eye-tracking in Fig. 7. Table 4 shows the success rate of low-level and DF predictors. Our DF predictor outperforms the low level predictor when task-imposed constraints exist (Fig. 8).

Figure 8: Comparison of low level gaze prediction (middle) and our DF predictor (right) for the same scene (left). The player is threatened by the soldier.

5.2 Dynamic Disparity Management

We used a protocol inspired by both [38, 32]. At the end of each pair in a session, participants were asked to choose between the two sessions of a given pair, to determine (i) which one had more depth and (ii) which was more comfortable in terms of diplopia and eye fatigue.



No display management Ours OSCAM

Figure 9: Left to right: No display management, Ours, OSCAM. Please use red/cyan anaglyph glasses; best viewed on a monitor.

We received 600 answers in total (2 questions $\times$ 3 sessions $\times$ 10 conditions $\times$ 10 participants) (Table 5). For the first question about which condition had more depth, our method outperforms both OSCAM and Standard. Our method was preferred 71% of the time when compared to Standard, and 67% of the time when compared to OSCAM. We also confirmed that OSCAM is preferred over standard (68% of the time). All results are statistically significant (t-test, $p < 0.01$).

For question 2 on comfort our method and OSCAM outperform Standard being preferred 73% and 78% of the time respectively. These results are statistically significant (t-test, $p < 0.01$). However, when comparing our method to OSCAM in terms of eye strain, participants did not have a clear preference (52% vs 48% respectively, $p = 0.27$).

	Ours/Stan.	**Ours/OSCAM**	**OSCAM/Stan.**
Depth	Ours 71% stdev: 14.4	Ours 67% stdev: 24.5	OSCAM 68% stdev: 7.9
Strain	Ours 78% stdev: 13.3	Ours 52 % stdev: 10.3	OSCAM 78% stdev: 6.3

Table 5: Preferred method of stereo grading for each question and session. Ours<>OSCAM results for eye strain are non-significant.

6 Discussion and Conclusion

Our predictor successfully predicts gaze in a video game, is automatic, avoiding the need for manual tagging of objects and supports object motion in contrast to previous task-based and high level approaches. Our localized stereo grading approach provides better perceived depth than previous global methods, while maintaining similar levels of viewing comfort.

Discussion and Limitations. Although our experiments have focused on eye-glass-based stereo displays, similar problems appear for any single-screen display that supports stereo viewing [35]. While HMDs offering collimated lenses that transfer the focus plane to infinity are becoming available, this does not eliminate the vergence-accommodation conflict for objects in close proximity. In addition, there is no indication that VR gaming is any different regarding stereo perception than standard stereoscopic content. In such cases our method is meaningful. Like all stereo grading methods, our approach modifies depth and speed of objects in a scene. Even though speed modification did not cause problems in our experiments, a more involved stereo-motion speed-preserving optimization strategy [26] could be required. Currently, the speed of the classifier is low (2 queries/second); a more optimized implementation of DFs would improve this speed. The training phase required an eye-tracked HMD; the typical cost of such a setup is probably a realistic option for a game-studio, however, a desktop eye tracker could be used as well.

Future Work. We demonstrated our ideas on a prototype game level. In a production context, our approach could serve as a basis for the development of a viable and attractive solution in game design. Training is easy and can be incorporated to the existing game testing pipelines by simply adding an eye tracker in the game testing rig. The trained model does not capture absolute eye tracking coordinates but learns vergence patterns based on game mechanics. We expect that a trained instance of the model may be extended to the other levels of the same game or different games with similar mechanics. If prediction rate becomes faster, our learning based predictor could be employed to adjust level-of-detail, depth-of-field or game difficulty based on user gaze.

Acknowledgements

This research has been co-financed by the European Union (European Social Fund - ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) - Research Funding Program: Heracleitus II: Investing in knowledge society through the European Social Fund. The work was partly supported by EU FP7 project ICT-611089-CR-PLAY www.cr-play.eu.

References

[1] M. Bernhard, C. Dell'mour, M. Hecher, E. Stavrakis, and M. Wimmer. The effects of fast disparity adjustment in gaze-controlled stereoscopic applications. In *Proc. of the Symp. on Eye Tracking Res. and Appl.*, pages 111–118. ACM, 2014.

[2] M. Bernhard, E. Stavrakis, M. Hecher, and M. Wimmer. Gaze-to-object mapping during visual search in 3D virtual environments. *ACM Trans. on Applied Perception (TAP)*, 11(3):14:1–14:17, Aug. 2014.

[3] M. Bernhard, E. Stavrakis, and M. Wimmer. An empirical pipeline to derive gaze prediction heuristics for 3D action games. *ACM Trans. on Applied Perception (TAP)*, 8(1):4, 2010.

[4] C. M. Bishop et al. *Pattern recognition and machine learning*, volume 4. Springer New York, 2006.

[5] A. Borji and L. Itti. State-of-the-art in visual attention modeling. *IEEE Trans. on PAMI*, 35(1), 2013.

[6] L. Breiman. Bagging predictors. *Mach. Learn.*, 24(2):123–140, 1996.

[7] L. Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, 2001.

[8] N. D. Bruce and J. K. Tsotsos. An attentional framework for stereo vision. In *Computer and Robot Vision, 2005. Proc.. The 2nd Canadian Conference on*, pages 88–95. IEEE, 2005.

[9] K. Calagari, K. Templin, T. Elgamal, K. Diab, P. Didyk, W. Matusik, and M. Hefeeda. Anahita: A system for 3d video streaming with depth customization. In *Proceedings of the ACM International Conference on Multimedia*, pages 337–346. ACM, 2014.

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

[11] A. Chapiro, O. Diamanti, S. Poulakos, C. OSullivan, A. Smolic, and M. Gross. Perceptual evaluation of cardboarding in 3D content visualization. *Proc. of ACM SAP*, 2014.

[12] O.-H. U. Crawford, Chris. *The art of computer game design*. Osborne/McGraw-Hill Berkeley, CA, 1984.

[13] P. Didyk, T. Ritschel, E. Eisemann, K. Myszkowski, H.-P. Seidel, and W. Matusik. A luminance-contrast-aware disparity model and applications. *ACM Transactions on Graphics (TOG)*, 31(6):184, 2012.

[14] S.-P. Du, B. Masia, S.-M. Hu, and D. Gutierrez. A metric of visual comfort for stereoscopic motion. *ACM Transactions on Graphics (TOG)*, 32(6):222, 2013.

[15] M. S. El-Nasr and S. Yan. Visual attention in 3D video games. In *Proc. of the 2006 ACM SIGCHI Intl. Conf. on Advances in computer entertainment technology*. ACM, 2006.

[16] A. Gahan. *3ds Max Modeling for Games: Insider's guide to game character, vehicle, and environment modeling*. CRC Press, 2013.

[17] S. Gateau and R. Neuman. Stereoscopy from xy to z. *Courses of SIGGRAPH Asia*, 63, 2010.

[18] B. Guenter, M. Finch, S. Drucker, D. Tan, and J. Snyder. Foveated 3d graphics. *ACM Transactions on Graphics (TOG)*, 31(6):164, 2012.

[19] S. Han, X. Wan, and G. W. Humphreys. Shifts of spatial attention in perceived 3-d space. *The Quarterly Journal of Experimental Psychology*, 58(4):753–764, 2005.

[20] S. Heinzle, P. Greisen, D. Gallup, C. Chen, D. Saner, A. Smolic, A. Burg, W. Matusik, and M. Gross. Computational stereo camera system with programmable control loop. In *ACM Transactions on Graphics (TOG)*, volume 30, page 94. ACM, 2011.

[21] D. M. Hoffman, A. R. Girshick, K. Akeley, and M. S. Banks. Vergence–accommodation conflicts hinder visual performance and cause visual fatigue. *Journal of vision*, 8(3):33, 2008.

[22] L. Itti, C. Koch, E. Niebur, et al. A model of saliency-based visual attention for rapid scene analysis. *IEEE Trans. on PAMI*, 20(11):1254–1259, 1998.

[23] R. J. Jacob and K. S. Karn. Eye tracking in human-computer interaction and usability research: Ready to deliver the promises. *Mind*, 2(3):4, 2003.

[24] G. R. Jones, D. Lee, N. S. Holliman, and D. Ezra. Controlling perceived depth in stereoscopic images. In *Photonics West 2001-Electronic Imaging*, pages 42–53. International Society for Optics and Photonics, 2001.

[25] T. Judd, K. Ehinger, F. Durand, and A. Torralba. Learning to predict where humans look. In *ICCV*, pages 2106–2113. IEEE, 2009.

[26] P. Kellnhofer, T. Ritschel, K. Myszkowski, and H.-P. Seidel. Optimizing disparity for motion in depth. In *Computer Graphics Forum*, volume 32, pages 143–152, 2013.

[27] G. A. Koulieris, G. Drettakis, D. Cunningham, and K. Mania. An automated high level saliency predictor for smart game balancing. *ACM Trans. on Applied Perception (TAP)*, 11(4), 2014.

[28] G. A. Koulieris, G. Drettakis, D. Cunningham, and K. Mania. C-LOD: Context-aware material level-of-detail applied to mobile graphics. *Computer Graphics Forum (Proc. EGSR)*, 33(4):41–49, 2014.

[29] J. Laird and M. VanLent. Human-level AI's killer application: Interactive computer games. *AI magazine*, 22(2):15, 2001.

[30] M. Lambooij, M. Fortuin, I. Heynderickx, and W. IJsselsteijn. Visual discomfort and visual fatigue of stereoscopic displays: a review. *J. of Imaging Sci. and Tech.*, 53(3), 2009.

[31] M. Lambooij, W. IJsselsteijn, and I. Heynderickx. Visual discomfort of 3D tv: Assessment methods and modeling. *Displays*, 32(4):209–218, 2011.

[32] M. Lang, A. Hornung, O. Wang, S. Poulakos, A. Smolic, and M. Gross. Nonlinear disparity mapping for stereoscopic 3D. *ACM Trans. on Graphics*, 29(4):75, 2010.

[33] D. Luebke and B. Hallen. *Perceptually driven simplification for interactive rendering*. Springer, 2001.

[34] B. Masia, G. Wetzstein, C. Aliaga, R. Raskar, and D. Gutierrez. Display adaptive 3d content remapping. *Computers & Graphics*, 37(8):983–996, 2013.

[35] B. Masia, G. Wetzstein, P. Didyk, and D. Gutierrez. A survey on computational displays: Pushing the boundaries of optics, computation, and perception. *Computers & Graphics*, 37(8):1012–1038, 2013.

[36] L. M. Meesters, W. A. IJsselsteijn, and P. J. Seuntiens. A survey of perceptual evaluations and requirements of three-dimensional tv. *Circuits and Systems for Video Technology, IEEE Trans. on*, 14(3):381–391, 2004.

[37] B. Mendiburu. *3D movie making: stereoscopic digital cinema from script to screen*. CRC Press, 2012.

[38] T. Oskam, A. Hornung, H. Bowles, K. Mitchell, and M. H. Gross. Oscam-optimized stereoscopic camera control for interactive 3D. *ACM Trans. on Graphics*, 30(6):189, 2011.

[39] J. R. Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[40] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, 2015.

[41] C. A. Rothkopf, D. H. Ballard, and M. M. Hayhoe. Task and context determine where you look. *Journal of vision*, 7(14):16, 2007.

[42] D. D. Salvucci and J. H. Goldberg. Identifying fixations and saccades in eye-tracking protocols. In *Proc. of the 2000 Symp. on Eye Tracking Res. & Appl.*, pages 71–78. ACM, 2000.

[43] C. E. Shannon. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55, 2001.

[44] T. Shibata, J. Kim, D. M. Hoffman, and M. S. Banks. The zone of comfort: Predicting visual discomfort with stereo displays. *Journal of vision*, 11(8):11, 2011.

[45] K. Simons. A comparison of the frisby, random-dot e, tno, and randot circles stereotests in screening and office use. *Archives of ophthalmology*, 99(3):446–452, 1981.

[46] M. Slater, B. Spanlang, and D. Corominas. Simulating virtual environments within virtual environments as the basis for a psychophysics of presence. *ACM Trans. on Graphics*, 29(4):92, 2010.

[47] M. Stanton, B. Humberston, B. Kase, J. F. O'Brien, K. Fatahalian, and A. Treuille. Self-refining games using player analytics. *ACM Trans. on Graphics*, 33(4), 2014.

[48] L. B. Stelmach, W. J. Tam, F. Speranza, R. Renaud, and T. Martin. Improving the visual comfort of stereoscopic images. In *Electronic Imaging 2003*, pages 269–282. International Society for Optics and Photonics, 2003.

[49] V. Sundstedt, M. Bernhard, E. Stavrakis, E. Reinhard, and M. Wimmer. Visual attention and gaze behavior in games: An object-based approach. In *Game Analytics*, pages 543–583. Springer, 2013.

[50] V. Sundstedt, E. Stavrakis, M. Wimmer, and E. Reinhard. A psychophysical study of fixation behavior in a computer game. In *Proc. of the 5th Symp. on Applied perception in graphics and visualization*, pages 43–50. ACM, 2008.

[51] B. W. Tatler, M. M. Hayhoe, M. F. Land, and D. H. Ballard. Eye guidance in natural vision: Reinterpreting salience. *Journal of vision*, 11(5):5, 2011.

[52] K. Templin, P. Didyk, K. Myszkowski, M. M. Hefeeda, H.-P. Seidel, and W. Matusik. Modeling and optimizing eye vergence response to stereoscopic cuts. *ACM Trans. on Graphics*, 33(4):145, 2014.

[53] D. Walther and C. Koch. Modeling attention to salient proto-objects. *Neural Networks*, 19(9):1395–1407, 2006.

[54] A. J. Woods, T. Docherty, and R. Koch. Image distortions in stereoscopic video systems. In *IS&T/SPIE's Symp. on Electronic Imaging: Science and Technology*, pages 36–48. International Society for Optics and Photonics, 1993.