https://doi.org/10.2352/ISSN.2470-1173.2020.9.IQSP-288

The Cone Model: Recognizing gaze uncertainty in virtual environments

Anjali K. Jogeshwar [1], ***Gabriel J. Diaz*** [1], ***Susan P. Farnand*** [2], ***Jeff B. Pelz*** [1]
[1] ***Chester F. Carlson Center for Imaging Science, Rochester Institute of Technology, Rochester, NY 14623, USA***
[2] ***Program of Color Science, Rochester Institute of Technology, Rochester, NY 14623, USA***

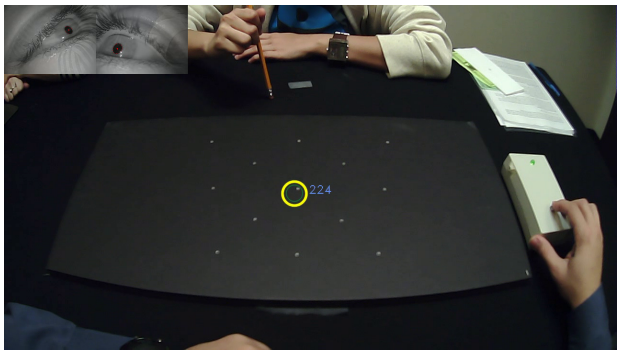
Abstract

Eye tracking is used by psychologists, neurologists, vision researchers, and many others to understand the nuances of the human visual system, and to provide insight into a person's allocation of attention across the visual environment. When tracking the gaze behavior of an observer immersed in a virtual environment displayed on a head-mounted display, estimated gaze direction is encoded as a three-dimensional vector extending from the estimated location of the eyes into the 3D virtual environment. Additional computation is required to detect the target object at which gaze was directed. These methods must be robust to calibration error or eye tracker noise, which may cause the gaze vector to miss the target object and hit an incorrect object at a different distance. Thus, the straightforward solution involving a single vector-to-object collision could be inaccurate in indicating object gaze. More involved metrics that rely upon an estimation of the angular distance from the ray to the center of the object must account for an object's angular size based on distance, or irregularly shaped edges - information that is not made readily available by popular game engines (e.g. Unity©/Unreal©) or rendering pipelines (OpenGL). The approach presented here avoids this limitation by projecting many rays distributed across an angular space that is centered upon the estimated gaze direction.

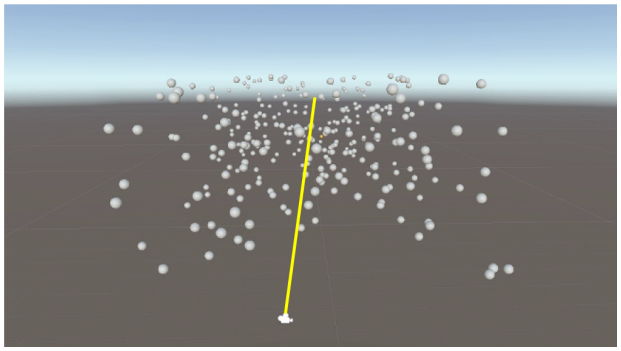
Introduction

In the past several years, the virtual reality industry has begun to integrate affordable, consumer-grade eye trackers into virtual reality head-mounted displays (HMDs). Although the greatest anticipated use is in video gaming, there is also great potential for widespread adoption in the sciences. Many studies in psychology and neuroscience have already adopted mobile eye tracking as a tool to investigate visual attention in the real world. The presence of high accuracy and affordable HMD integrations for eye tracking will facilitate a new era of research in simulated environments that offer greater experimental control and manipulations, which are not possible in more natural contexts.

Like eye tracker integrations into HMD's, many mobile eye trackers (in the absence of an HMD) are video based, with dedicated cameras to record the eyes as the user visually explores the real-world scene. These video-based eye trackers estimate a single (x,y) point in scene-camera coordinates as the gaze location on the basis of visual features present in the eye images. To clearly view the gaze point during data collection and analysis, the estimated gaze is often represented as a cross-hair or circle overlaid upon the scene imagery. This estimate varies systematically with features related to eye orientation, such as the location of the pupil centroid [4].



(a) Analyzing 2D gaze data



(b) Analyzing 3D gaze data

Figure 1. *Analysis of eye tracking data in different environments*

Video noise and errors in the process of feature detection invariably introduce uncertainty in the estimation of gaze location. Many software suites designed to help the experimenter interpret gaze data allow one to visualize this uncertainty by adjusting the radius of a disc overlaid upon the scene imagery, rather than a point, as seen in Figure 1(a). Note that because the circle is representative of a 2D projection of conical shape emanating from the eye, the disc corresponds to a region of uncertainty of fixed angular size around the estimated gaze location. Analyzing this uncertain region around the estimated gaze gives a better understanding of all the potential objects the observer could be looking at.

Eye trackers used for studying natural tasks can provide accurate and precise gaze data (under ideal conditions, 0.6 degrees and 0.08 degrees respectively [2]). The quality of gaze estimation is affected by a number of elements that contribute to the accuracy and the precision of the eye tracker. A degradation in the quality

of images obtained from the cameras will negatively influence the eye-feature detection. Because the scene camera is offset from eye cameras, fixations far from the plane at which the eye tracker has been calibrated can introduce parallax error.

If the calibration of the eye tracker is off by some margin, then the estimated gaze (as reported by the eye tracker) would not be the same as true gaze. Because the experimenter does not know at any instant the exact location of the gaze point, there exists uncertainty in where a person is truly looking.

Reducing the error between true and estimated gaze position is important, but some uncertainty will remain. Some common methods for reducing the error include multiple point per-plane calibration, addition of an offset, and calibration at multiple depth planes [1]. Binaee et al. [3] used post-hoc calibration to deal with the deteriorating gaze accuracy in-between calibrations.

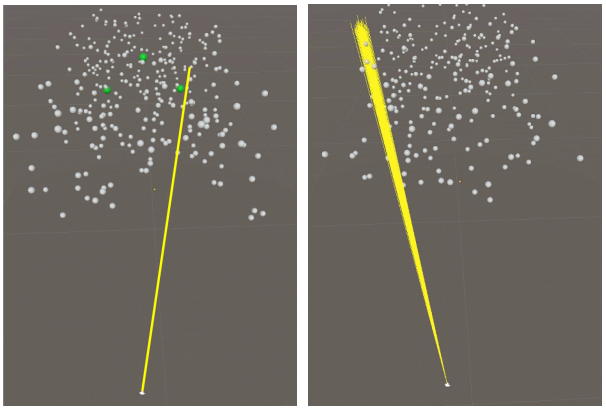
If an object is close to our eyes, for example a cellphone, and if the estimated gaze is off by 0.5^{o}, then at least the correct icon on the cellphone screen can be detected. However, if the same object (say the cellphone) is held a couple meters away, and the estimated gaze is off by 0.5^{o}, then the complete object (i.e., the icon) will be missed during detection. This would potentially lead to mis-attribution of gaze location upon the wrong icon or Object-Of-Interest (OOI). Hence, uncertainty in screen space increases as an object (i.e., cell phone) moves farther from the observer. The uncertainty in the gaze estimation directly affects the data quality as the distance of the OOI increases from the observer. For further distances, the object marked by the eye tracker, i.e., OOI, would not necessarily be the one the participant was actually looking at. This makes it crucial to acknowledge the uncertainty we obtain from the eye tracker. In this manuscript, we propose a visualization tool to accommodate the noise inherent in the estimation of gaze direction in virtual environments.

Some studies are impractical to be carried out in the real world. For example, eye tracking a driver to understand their behaviour or response during a car crash is unrealistic and infeasible. In contexts like these, it may be more practical to carry out research in a safe and controlled virtual environment. To track eyes in virtual reality, eye trackers are placed inside the HMD and the calibration is also done inside the virtual environment. To analyze eye tracking data in virtual reality, gaze can be visually represented as a ray projected in the direction of the gaze, and the OOI identified by observing the object that the gaze ray collides with as seen in Figure 1(b). However, this representation can be misleading, as it does not account for the uncertainty in the process of gaze estimation.

Consider the small mislocalizations or inaccuracies in the process of feature detection which can introduce errors in the estimation of gaze location of varying size. This may result in misclassification of the OOI, which is especially problematic for the interpretation of gaze in the presence of visually cluttered environments. Scenes with many small objects, or large objects at a distance, subtending small portions of the visual field increase the chances of incorrectly classifying the OOI.

Solutions in the 3D environment include casting a flattened sphere which orients itself tangential to the surface. Another method includes training a deep neural network with gaze direction and ground truth to learn the eye movements and estimate gaze effectively.

None of these methods allow the user the flexibility to interpret gaze on the basis of uncertainty defined in terms of the angular error around the sensed direction of gaze as shown in Figure 1(a) for mobile eye tracking in natural environments. In that case, the circle is drawn on a 2D projection of the 3D environment so it subtends a fixed angular extent in the real world. If this representation could be visualized from a third-person point of view, it would appear to be a cone projected into the 3D environment. Our approach is to represent gaze not as a ray, but as a conical projection from the eye into the 3D environment, that subjects an angular radius proportional to the uncertainty of the gaze estimate real-time, similar to the approach proposed by Watson et al.[5].



(a) Traditional: Single ray collision based detection of the object-of-interest (OOI)

(b) Proposed: Multiple ray cone model based detection of the object-of-interest OOI

Figure 2. *Analysis techniques for virtual environment*

This method of gaze projection then provides the user with a list of potential objects of interest (as well as their locations in the virtual environment) based on an angle of uncertainty provided by the experimenter. Figure 2 shows two sub-figures. One is the detection of the viewed object with a single ray cast (the gaze vector). The other is the detection of all the possible viewed objects within 1^{o} uncertainty around the estimated gaze vector.

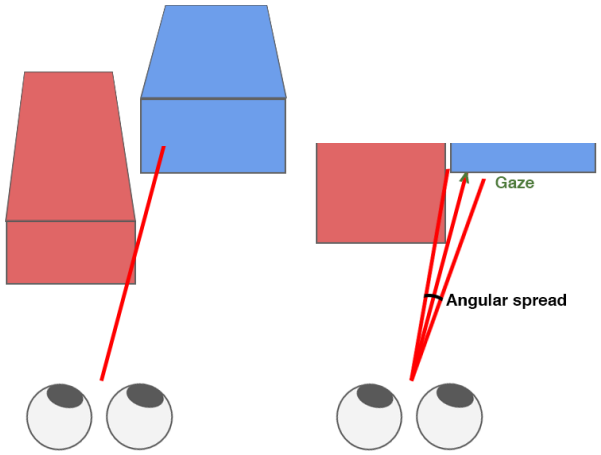
The Cone Model

The model is first explained conceptually and mechanically. The actual implementation is then covered in detail.

Concept

When an observer is viewing a 3-dimensional scene, uncertainty in gaze position can also result in ambiguity about the depth at which the observer is attending. This is common to eye tracking in real and virtual environments. However, in real environments, the objects of the real world are projected onto the scene-camera image plane and the uncertainty is acknowledged by projecting a fixed-size circle on the scene-camera image. Because objects at all distances are projected onto the 2D image plane, depth information is lost and a fixed-size circle represents a constant visual angle. In virtual environments, where the 3D coordinates of objects are maintained, we take account of the uncertainty in three dimensions with a 3D shape (nominally a cone) projected into the viewing volume. The shape subtends a constant angular extent scaled to the uncertainty in gaze position. The scaled cone is created with its apex at the cyclopean eye in the virtual world, and

its axis is aligned with the gaze direction in the world coordinate system as obtained from the eye tracker. Figure 3(b) shows a 2D projection of the cone with an angular spread overlaid on the gaze vector of Figure 3(a).



(a) Object of interest detection with single ray

(b) Top-down view with cone visualization

Figure 3. *3-Dimensional analysis of eye tracking data*

By projecting a cone of, for example, 1° of visual angle centered on the estimated gaze direction, we generate a list of OOIs that the observer could be looking at.

Mechanics

To detect all objects in the volume around the gaze vector, the cone has to be updated on the basis of gaze direction in real time. We explored several alternative implementations, including casting rays only on the boundary around the estimated gaze direction with a radius scaled to the uncertainty (forming a 'hollow' cone); casting rays randomly within a scaled Gaussian envelope; casting a fixed number of rays uniformly distributed within a cone; and projecting an n-sided polygonal pyramid centered on the estimated gaze. The mathematical implementation for all alternatives had the same fundamentals. The cone is first created with its axis on the y-axis in 3-dimensional space with the apex of the cone at the origin. To align the cone on the gaze direction, the normal (n) is calculated between the gaze direction and the axis of the cone (i.e., the y-axis). The cone is then rotated about the normal by the angle between the apex and the gaze direction. Lastly, the cone is displaced from the origin to the cyclopean eye. The final implementation of the cone was casting multiple rays in the shape of the cone. Even though the cone is not a solid object, the math for orientation and displacement remains the same.

The cone updates its location and orientation based on the cyclopean eye in 3-dimensional space for each gaze estimate received from the eye tracker in the HMD. The software was developed in Unity Version 2019.2.9f1. Within the Unity API, the cone is visible to the researcher in the scene view, but not to the participant in the HMD/game view.

Implementation

Since a constant visual angle covers different amounts of information at different depth planes, there was a necessity to implement a structure that would grow in size as it moved along

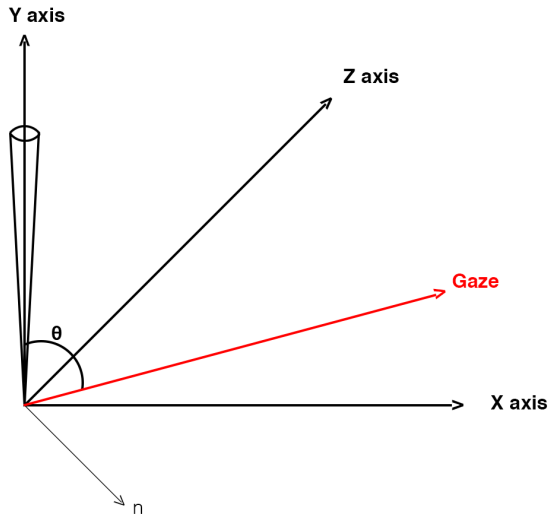


Figure 4. *Mathematical cone model*

the gaze vector away from the cyclopean eye. The cone formation suited the prerequisite well. Shooting a few rays around the estimated gaze with a radius scaled to the uncertainty creates a 'hollow' cone that confines the volume around the estimated gaze. However, the drawback with this approach is that if an object subtends an angle that is smaller than the angular spread of the cone created, the object will go undetected as shown in Figure 5.

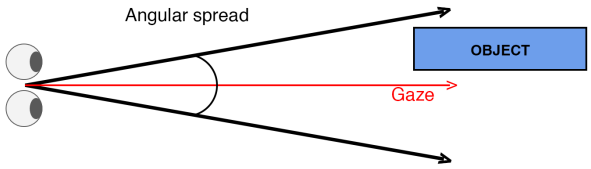


Figure 5. *Cone missing the small object*

To never miss an object in the 3-dimensional volume around the gaze would require a solid, enclosed volume. So an n-sided pyramid was implemented, and to give the "cone" an enclosed volume property, it was assigned a mesh and a collider object. The collider object detected all the collisions with the n-sided pyramid. However, the drawback with this implementation is that as of Unity 2018.3.11f1, the physics based collisions with physical objects defined by 3D mesh do not give any information on the location of the collision on the objects with respect to the cone or with respect to the world.

Instead of using a solid cone, our solution is to fill a conical space with a dense projection of rays. Hundreds of rays are projected from the cyclopean eye around the gaze vector in the shape of a cone. These rays are chosen from a 2-dimensional Gaussian distribution scaled to the defined uncertainty. Since eye trackers are designed to be as accurate as possible, we expect the true gaze to be around the estimated gaze. The farther we go from the estimated gaze, the probability of being close to true gaze decreases. Thus, this is modeled by 2-dimensional Gaussian Distribution where the mean is the estimated gaze and variance is the angular uncertainty of the eye tracker. Figure 6 shows a cone

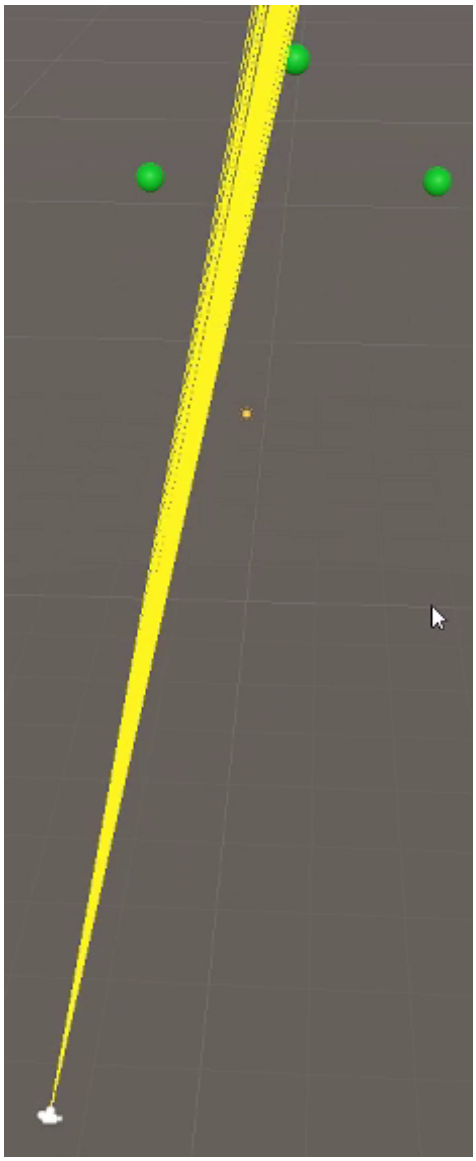


Figure 6. *Hundreds of rays shot in the shape of a cone*

created using 300 rays. This ray method outputs the point of collision of the ray with an object, which provides the distance of each OOI from the cyclopean eye. This can then be used to create an ordered list of potential OOIs by their distance from the observer.

Cone in practice

The cone is designed to detect all the potential objects one could be viewing. Figure 7 illustrates the cone concept. The figure shows

1. Three basic shapes representing the silhouettes of three objects placed in the virtual environment *(black solid lines)*
2. True gaze direction *(green square)*
3. Estimated gaze direction which, due to eye tracker noise, is offset from true gaze location by varying amounts in random directions *(red circle)*
4. Discs of radius 1^o and 3^o representing the 2D projection of the 3D cone within which rays would be cast when using the Cone Model for gaze estimation *(blue dotted and purple dashed lines)*

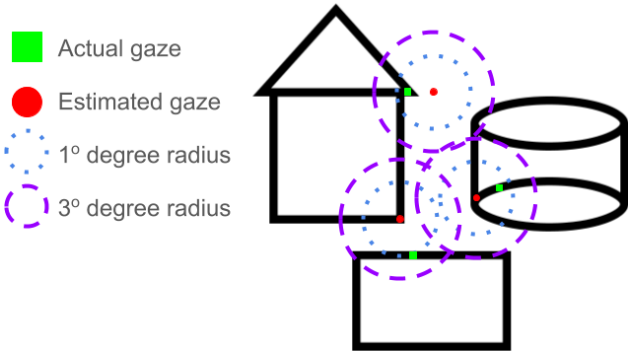


Figure 7. *Cone with various sizes (conceptual illustration)*

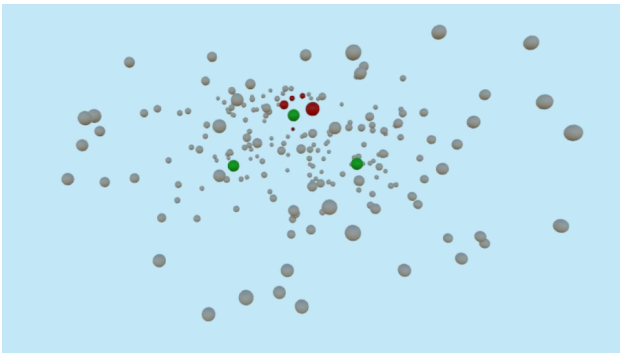
First, consider the conclusions that would be drawn if the OOI were inferred based upon estimated gaze location in the absence of an estimate of uncertainty (such as a single ray, visually indicated by the red markers in Figure 7). One object (the cylinder) would be detected correctly, one object would be misidentified (the rectangle as the house) and one object would be missed (the house) even though it would have been viewed.

Estimates can be improved if one accounts for the uncertainty in gaze estimation. For example, to approximate uncertainty as a 1^o radius around estimated gaze direction (blue dotted circles in Figure 7), would result in correct detection of the house and cylinder as the OOI, but return two potential OOIs for the lower-left estimated gaze location. Accounting for uncertainty as high as a 3^o radius around the estimated gaze direction (purple dashed circles in Figure 7), would result in two potential OOIs for every estimated gaze direction. For the Figure 7, one can comment if the potential detection is correct or incorrect only if they know where the true gaze was. However, this information is never available in practice. The eye trackers are designed to be as close to the true gaze as possible but there still exists uncertainty in gaze estimation.

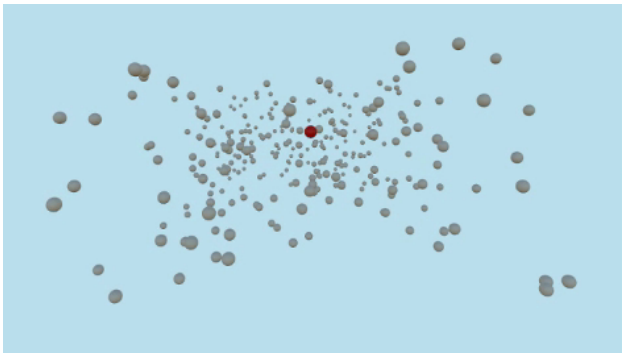
As seen before, due to uncertainty, one can miss out on detections, or obtain correct or incorrect detections. Hence, it is preferred to obtain extra information i.e., a list of all potential objects then miss out on the OOI or obtain a false OOI.

The example presented in Figure 7 demonstrates that cone-based OOI estimates may return multiple objects, and/or objects that the subject was not visually attending to, but that were near to the estimated gaze direction. Resolving these ambiguities requires additional assumptions, and analysis of such data then lies at the discretion of the researcher. For example, s/he may choose the OOI based on priors, or maintain a list of multiple possible OOIs.

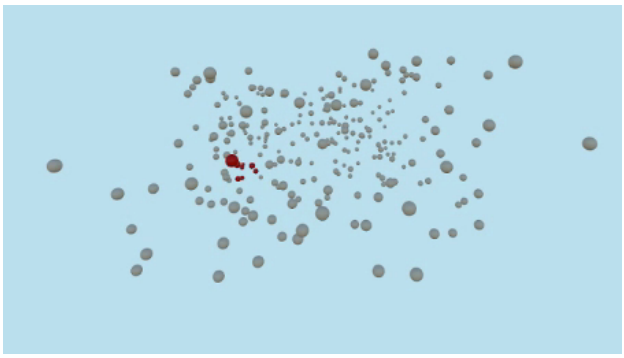
To illustrate the practical implementation of the cone model for gaze estimation, some scenarios were created in VR. Figure 8(a) shows a number of white, red and green spheres. The three green spheres are the targets the observer was supposed to look at. The white spheres contribute additional objects (or complexity) to the environment. The white spheres also turn red (for fifty milliseconds) upon collision with the cone. If the target is known,



(a) Detection with cone and known targets. This image depicts a frame following instructions for the observer to fixate at the top-most green sphere.



(b) Detection without cone and known targets.



(c) Detection with cone and without known targets

Figure 8. *Situations for analysis of gaze data in virtual environment*

we can say that the red spheres surrounds one of the green spheres in the environment and thus, the subject was perhaps looking at the lower-right green sphere.

In Figures 8(b) and 8(c), the true OOI is unknown, but possible OOIs are shaded red based upon either a single ray in the direction of the gaze estimate (Figure 8(b)), or multiple rays-member of the cone projection model (Figure 8(c)). In the case of a single red sphere, it is impossible to tell if that sphere was a correct OOI or an incorrect one. However, in the case of multiple red spheres, uncertainty is incorporated in the cone model and we can say with confidence that one of the red spheres is the correct OOI. One key point to note here is that the cone is designed to acknowledge and incorporate the uncertainty. However, the cone is not designed to calculate or resolve the uncertainty.

Hardware

The cone model was developed in Unity 2019.2.9f1 and tested using an HTC Vive Eye Pro with a Tobii Pro Eye Tracker. To run the eye tracker in the HMD, SRanipal version 1.1.0.1 was used. The entire setup was on a Windows machine with 32 GB RAM and GeForce 2080 RTX GPU with 8GB memory.

Evaluation

To understand how the cone model affects the performance of a scene in the virtual environment, we correlate the frame rate to the complexity (in number of objects and rays) in two scenes.

Table 1. Mean frame rate and standard deviation for simple scene

Rays	Mean	Std Dev
0	90.13	0.16
50	90.11	0.22
100	90.09	0.16
250	89.90	0.31
500	88.77	0.64
750	85.75	0.97
1000	82.11	0.79
2000	70.68	0.87

The first scene was designed to estimate the effect of the number of rays used in OOI estimation using the cone model on the frame rate. This scene included only three cubes embedded in the virtual environment with lighting that cast shadows, received shadows and accommodated for dynamic occlusion (see Figure 9). Each cube was of side one unit length subtending approximately 2.9^o from the observer's viewpoint. The participant's task was to look at the three cubes repeatedly, one after another for five seconds which contributed to one trial.

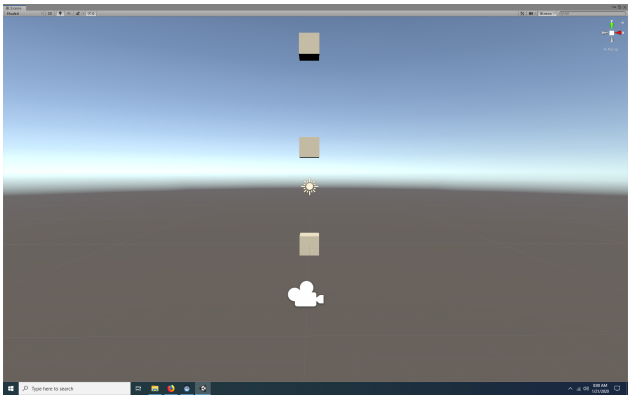


Figure 9. *Scene one: few objects, dynamic lighting*

This was repeated for a cone model with a fixed diameter of 1^o within which we projected 0, 50, 100, 250, 500, 750, 1000 and 2000 rays. For each number of rays, every trial is conducted 31 times. Table 1 summarizes the performance of the cone model in terms of frame rate on the scene. For a scene with dynamic lighting and a small number of objects, we can see that the performance gets affected upon casting more than 500 rays.

Table 2. Mean frame rate and standard deviation for varying complexities

Sphere Ray	0	50	100	250	500	750	1000
0	90.14 ± 0.33	90.04 ± 0.29	90.12 ± 0.17	89.82 ± 0.57	87.64 ± 1.08	79.91 ± 2.02	73.43 ± 1.66
50	89.88 ± 1.04	90.16 ± 0.17	90.10 ± 0.31	89.86 ± 0.66	86.64 ± 1.74	78.98 ± 1.64	72.93 ± 1.97
100	90.11 ± 0.27	90.11 ± 0.32	90.07 ± 0.33	89.70 ± 0.54	85.82 ± 2.23	78.36 ± 1.55	72.36 ± 1.73
250	90.15 ± 0.20	90.04 ± 0.31	89.99 ± 0.46	89.12 ± 1.19	82.93 ± 1.96	76.42 ± 1.57	70.89 ± 1.53
500	90.03 ± 0.73	89.87 ± 0.62	89.36 ± 1.06	86.87 ± 1.46	78.82 ± 1.80	73.32 ± 1.63	68.22 ± 1.82
750	90.07 ± 0.37	88.87 ± 1.21	87.34 ± 2.08	82.73 ± 1.50	75.83 ± 1.47	70.51 ± 1.57	65.55 ± 1.85
1000	89.47 ± 1.03	86.68 ± 1.72	84.21 ± 1.87	79.12 ± 1.49	73.04 ± 1.54	67.91 ± 1.73	62.68 ± 2.65
2000	80.92 ± 2.17	74.98 ± 1.63	72.77 ± 1.78	68.53 ± 1.39	63.37 ± 2.16	59.08 ± 1.59	54.89 ± 2.44

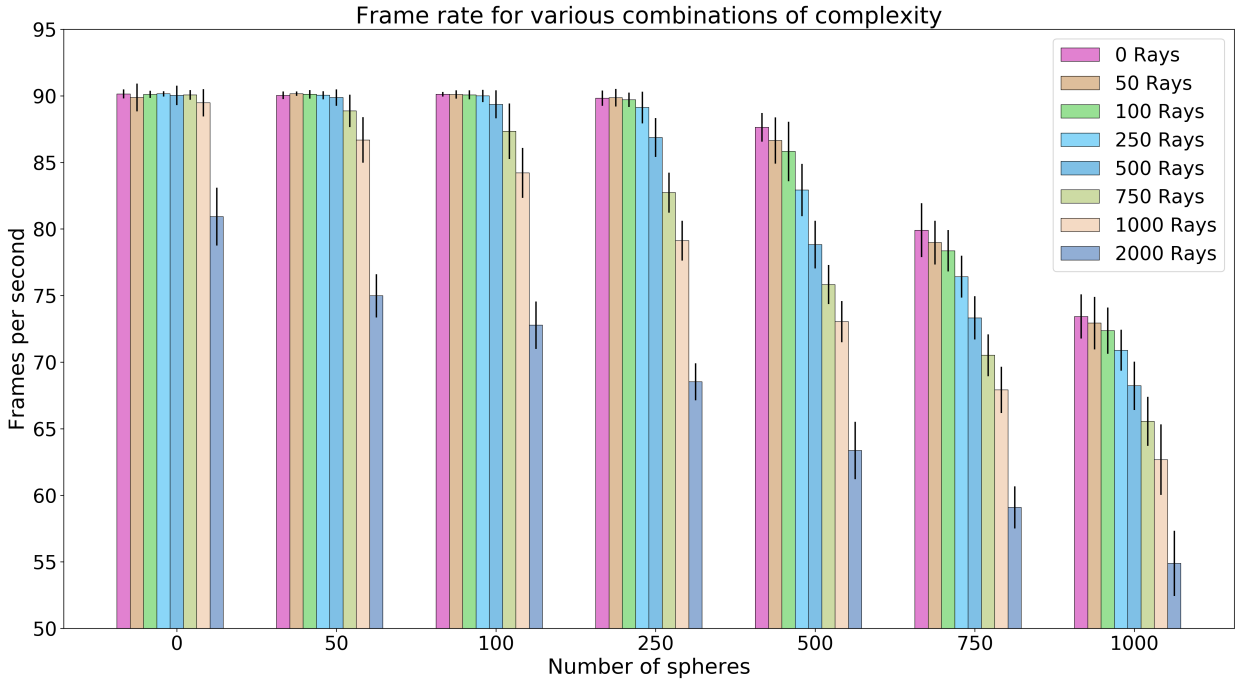


Figure 10. *Effect of scene complexity on frame rate*

Another scene was also designed in Unity where the number of spheres increased from 0 to 1000 in non-uniform steps. The aim of this scene was to measure the relationship between the overhead added by the complexity of the scene and the number of rays cast in the cone model. The lighting in the scene did not cast or receive any shadows and did not accommodate for dynamic occlusion either. However, the number of spheres is drastically higher compared to the prior scene with only three cubes. The spheres were of radius 0.25 unit length. The sets of spheres seen by the observer are 0, 50, 100, 250, 500 and 1000. For every set of spheres, the number of rays cast are 0, 50, 100, 250, 500, 750, 1000 and 2000. A total of 56 combinations comprising different spheres and rays were created. Each of these combinations was tested for 5 seconds per trial, 31 times. We calculated average frame rate by dividing the total number of frames by 5 seconds (the duration of one trial). Table 2 comprises the mean frame rate and the standard deviation for 56 combinations. Figure 10 shows the bar plots for the performance. We can see that generating 500 spheres in the scene starts affecting the frame rate, even without the rays being cast. For the case when there are no spheres, casting 2000 rays drastically affects the frame rate. The impact of the cone with 50 rays, and the cone with 100 rays was observed to be similar on the frame rate.

Discussion

We presented a novel methodology for the estimation of object(s)-of-interest from an uncertain estimate of gaze direction when immersed in a virtual environment. The radius of a cone-projection from the location of the virtual viewpoint was designed to be customized based upon the estimated magnitude of the eye tracker noise. The model does not affect frame rate with a few hundred rays, however, the frame rate is reduced when the number of rays approaches 2000 rays, and the complexity of the scene also influences the overall frame rate.

The final per-frame output of the cone methodology includes a list of all the objects that collided with the cone, all the points of collision and the distance of the points of collision from the cyclopean eye, for each instance of estimated gaze. There exist

enough data in the final output to post-process and calculate the angle between the ray that hit an object and the estimated gaze. One can rearrange the data on distance of collision, angle, number of hits, etc, to understand the importance of every object in the field of view.

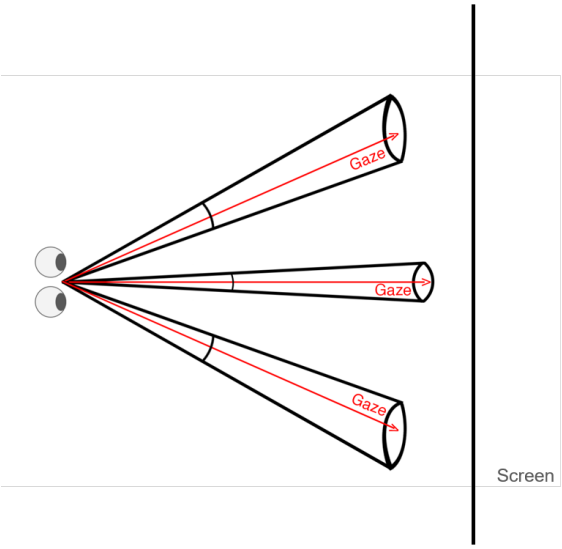


Figure 11. *Oblique elliptical cone at the peripheral area of the HMD*

As one starts to look away from the central field of view i.e., in the periphery, the uncertainty in estimated gaze increases. Future steps for the cone model include being adaptive to the field of view of the HMD and adjusting the angular spread accordingly. Moving towards the edges of the field of view in the HMD would require the right circular cone to shift to an oblique elliptical cone in real time as shown in Figure 11.

Acknowledgments

The authors would like to thank Catherine Fromm for sharing her expertise on Unity.

References

[1] Pfeiffer, Thies, Marc Erich Latoschik, and Ipke Wachsmuth. ”Evaluation of binocular eye trackers and algorithms for 3D gaze interaction in virtual reality environments.” JVRB-Journal of Virtual Reality and Broadcasting 5, no. 16 (2008).

[2] Kassner, Moritz, William Patera, and Andreas Bulling. ”Pupil: an open source platform for pervasive eye tracking and mobile gaze-based interaction.” In Proceedings of the 2014 ACM international joint conference on pervasive and ubiquitous computing: Adjunct publication, pp. 1151-1160. 2014.

[3] Binaee, Kamran, Gabriel Diaz, Jeff Pelz, and Flip Phillips. ”Binocular eye tracking calibration during a virtual ball catching task using head mounted display.” In Proceedings of the acm symposium on applied perception, pp. 15-18. 2016.

[4] DiScenna, Alfred O., Vallabh Das, Ari Z. Zivotofsky, Scott H. Seidman, and R. John Leigh. ”Evaluation of a video tracking device for measurement of horizontal and vertical eye rotations during locomotion.” Journal of neuroscience methods 58, no. 1-2 (1995): 89-94.

[5] Watson, Marcus R., Benjamin Voloh, Christopher Thomas, Asif Hasan, and Thilo Womelsdorf. ”USE: An integrative suite for temporally-precise psychophysical experiments in virtual environments for human, nonhuman, and artificially intelligent agents.” Journal of neuroscience methods 326 (2019): 108374.

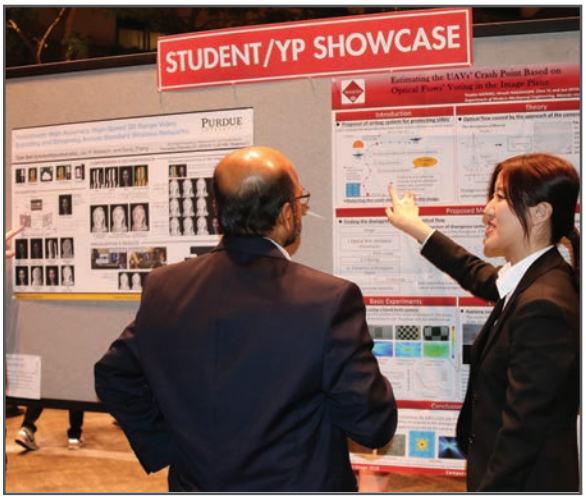
JOIN US AT THE NEXT EI!

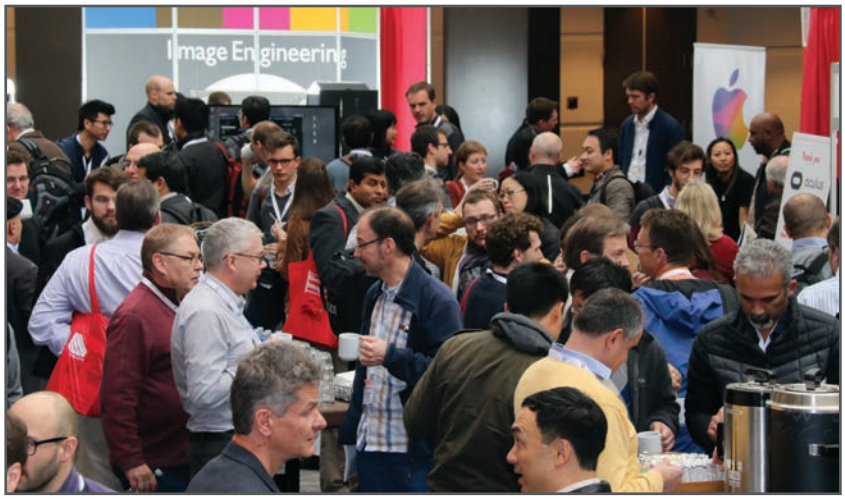
IS&T International Symposium on

Electronic Imaging

SCIENCE AND TECHNOLOGY

Imaging across applications . . . Where industry and academia meet!





• SHORT COURSES • EXHIBITS • DEMONSTRATION SESSION • PLENARY TALKS •
• INTERACTIVE PAPER SESSION • SPECIAL EVENTS • TECHNICAL SESSIONS •

www.electronicimaging.org

IS&T
imaging.org