Behavior Research Methods
2007, 39 (3), 552-560

A technique for simulating visual field losses in virtual environments to study human navigation

Francesca C. Fortenbaugh, John C. Hicks, Lei Hao, and Kathleen A. Turano
Johns Hopkins University, Baltimore, Maryland

The following paper describes a new technique for simulating peripheral field losses in virtual environments to study the roles of the central and peripheral visual fields during navigation. Based on Geisler and Perry's (2002) gaze-contingent multiresolution display concept, the technique extends their methodology to work with three-dimensional images that are both transformed and rendered in real time by a computer graphics system. In order to assess the usefulness of this method for studying visual field losses, an experiment was run in which seven participants were required to walk to a target tree in a virtual forest as quickly and efficiently as possible while artificial head and eye-based delays were systematically introduced. Bilinear fits were applied to the mean trial times in order to assess at what delay lengths breaks in performance could be observed. Results suggest that breaks occur beyond the current delays inherent in the system. Increases in trial times across all delays tested were also observed when simulated peripheral field losses were applied compared to full FOV conditions. Possible applications and limitations of the system are discussed. The source code needed to program visual field losses can be found at lions.med.jhu.edu/archive/turanolab/Simulated_Visual_Field_Loss_Code.html.

The central and peripheral visual fields differ both in terms of their anatomical structure and the type of visual information that is perceived from these regions. As a result of this, it is important to understand what roles these two areas may play in completing tasks that require the use of external visual information. Researchers studying activities such as reading and visual search have attempted to dissociate the individual contributions of each field by artificially restricting the other (Cornelissen, Bruin, & Kooijman, 2005; Rayner, 1998; Rayner & Bertera, 1979). Techniques such as the "moving window" or "foveal mask" have been created where a participant's eye movements are followed with an eye-tracker while the participant sits with his or her head immobilized. A mask is then tethered to the participant's point of fixation so that information is continuously blocked from reaching the central or peripheral visual field (Bertera & Rayner, 2000; Rayner, 1998). While eye-tracking technology has allowed researchers to utilize techniques like these for decades, similar techniques could not originally be developed for studies focusing on the role of vision in mobility and human navigation. This is because navigational studies require not only portable eye-tracking capabilities but also a coupling of eye-tracking software with programs that can monitor head movements to determine the position of one's body and direction of gaze several times per second. Furthermore, if one wants to be assured that all participants are viewing the same scene, such as displaying camera or virtual images through a head-mounted display (HMD), additional resources need to be devoted to rendering the appropriate image and then masking or blurring the periphery. These system requirements tax resources and, until recently, were too costly to address.

Traditionally, studies investigating the relationship between visual field size and navigational ability have relied on recruiting participants with actual visual impairments (Haymes, Guest, Heyes, & Johnston, 1996; Lovie-Kitchin, Mainstone, Robinson, & Brown, 1990; Marron & Bailey, 1982). From a clinical perspective, this approach is ideal and studies in this area have been critical for the development of effective orientation and mobility rehabilitation. However, if one's purpose is more directed at understanding the individual roles of the central and peripheral visual fields in navigation, simulating visual field losses offers several advantages over studying participants with actual field losses. First, virtual reality (VR) offers a unique opportunity to isolate the roles of specific visual field regions in navigation by presenting participants with exactly the same field of view (FOV) while they navigate through a virtual environment. For example, if a study is only concerned with a certain area of the peripheral field, say information lying 20º or more from the fovea, then participants can be presented with a symmetrical field loss that only excludes information lying 20º or more from the fovea. This sort of precision would be impossible to obtain with naturally occurring visual field losses. Second, because participants with normal vision would be recruited, problems with finding a large enough sample would be alleviated. Recruiting persons with normal vision also has another advantage. Because these individuals normally

K. A. Turano, kturano1@jhmi.edu

have full use of their visual field there will not be time for adaptation to a simulated field loss to occur without it being observed by the experimenters. Individuals who have been dealing with vision impairments for years have had time to adjust to a loss of information in some regions and whether adaptation occurs on a physiological, cognitive, or behavioral level, changes in some form will inevitably occur. Thus, simulating visual field losses may give a more accurate assessment of the roles that the central and peripheral visual fields play when they are accustomed to acting in concert. Fourth, although there are perceptual concerns when comparing virtual reality with natural settings, a virtual environment also allows researchers to control the physical aspects of an environment, assuring that all participants see the same objects under the same lighting conditions. Finally, virtual environments allow participants to navigate through a variety of settings and obstacles within the confines of a single laboratory space. Given the possibly debilitating effect of losing one's peripheral vision, this allows for greater participant safety as one can allow participants to run into virtual objects without the risk of injury.

In order for a VR system to simulate a natural field loss, however, the system used to present the virtual environment must be able to track a participant's position and gaze, render updated images, and mask parts of the visual field without a delay that would compromise a participant's performance. In the first part of this paper we describe such a technique for simulating peripheral field losses in virtual environments based on Geisler and Perry's (2002) technique for gaze-contingent multiresolution displays. In the second part we describe an experiment designed to test the effects of various system delays on participant performance to determine whether the inherent system delays accompanying this method are acceptable for research applications.

A GAZE-CONTINGENT MULTIRESOLUTION DISPLAY FOR SIMULATING PERIPHERAL FIELD LOSS

Overview

Computer scientists and electrical engineers originally developed gaze-contingent multiresolution displays (GCMD) in order to decrease the bandwidth necessary to send information over the Internet as well as to increase the processing speed in virtual reality systems (Duchowski, 2002; Loschky, 2003; Parkhurst & Niebur, 2002; Reingold, McConkie, & Stampe, 2003). For applications in virtual reality, decreasing the amount of resources needed for creating high-resolution images allows resources to be redirected into other areas, such as increasing frame rates. This is important because in order for researchers to be able to utilize virtual reality systems to study naturally occurring phenomena, the virtual environments must have the appearance of reality to the participants immersed in them. Slow refresh rates cause individuals to lose the sense that they are actually immersed in an environment and, for some individuals, lead to feelings of motion sickness (Parkhurst & Niebur, 2002; Reingold et al., 2003). A few researchers (Reingold & Loschky, 2002; Watson, Walker, Hodges, & Worden, 1996, 1997) have addressed the issue of slow refresh rates by decreasing the level of detail in areas surrounding the point of gaze in generated images. However, just having two levels of detail (high resolution and low resolution) does not mimic natural decreases in resolution across retinal eccentricity and a sharp break in resolution levels can affect participant performance. Reingold and Loschky (2002) have referred to this phenomenon as the *window effect*. Furthermore, decreasing resolution in the periphery does not always increase refresh rates. In general, savings in processing requirements are only seen if the images being masked are pre-generated (i.e., the masking is the only processing occurring in real time) or if the amount of blurring is restricted to one continuous area and not more than two levels of resolution.

Geisler and Perry (1998, 1999) developed a unique solution to the problem of increasing image transmission rates with a more natural decrease in resolution. In their method, information toward the periphery of a display becomes increasingly blurred while the clarity of information in the area being fixated on is preserved. In brief, Geisler and Perry's method creates a 3-D surface from a 2-D image, with the new dimension being the spatial resolution of the image; that is, moving along the *z*-axis, layers become increasingly blurred, and a final 2-D image is created by selecting the *z*-coordinate from a pregenerated map as the image is copied. The surface is referred to a multiresolution pyramid, where each layer stores a smaller (lower-resolution) image than the previous layer. This technique decreases the risk of changes in resolution affecting perception and performance by matching the resolution of the image across the display to the spatial resolution of the human eye at differing eccentricities. Furthermore, unlike the work of Reingold and Loschky (2002), Geisler and Perry's (2002) technique also allows for the shape of the high-resolution area to be arbitrary. Different regions of the field of view can be high-resolution, low resolution, or anywhere in between, so complex maps resembling natural visual fields or real field losses occurring from ocular diseases can be created. Unfortunately, implementing these more natural and complex deficits in resolution can be computationally expensive as multiple copies of each image must be made and blended together. This leads to increases, rather than decreases, in system delay. However, this does not mean that the technique loses applicability. As Geisler and Perry (2002) noted, this information reduction technique could also be used to study visual field losses, moving it from the realm of image transmission to that of research. On the basis of data from patients suffering from glaucoma, Geisler and Perry (2002) were able to create resolution maps depicting levels of spatial resolution across retinal eccentricities found with an actual peripheral field loss. It is this method that became the basis for the approach described below.

Implementation for Virtual Environments

One of the powerful features of recent GPUs is the ability to incorporate user-programmable procedures inside the

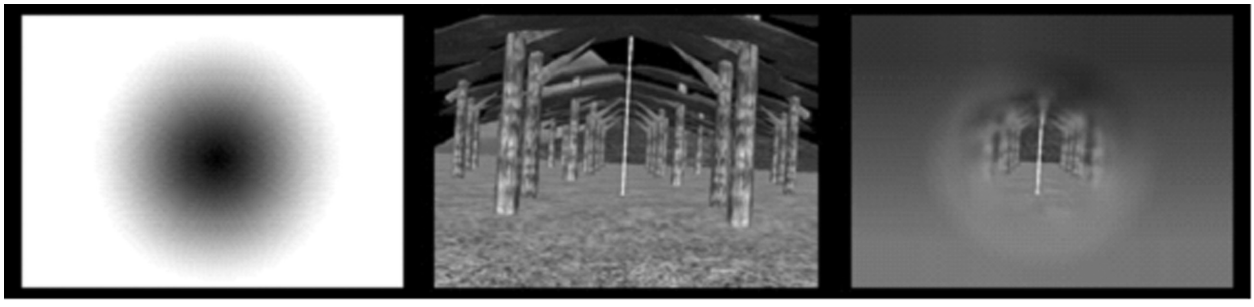

Figure 1. An exemplary resolution map is shown at left and source map at center. When the source map is blended using parameters specified by the resolution map the resulting image is blurred in the periphery but not in the center, as shown on the right side of the panel. Note that the size of the resulting image is still 48º by 38º, but only a 20º (diam) circle is visible.

texture blending pipeline of their rendering engines. These capabilities make today's graphics hardware much more flexible than previous generations, and allows programmers to leverage the GPU to effectively implement Geisler and Perry's (2002) method with greatly enhanced performance. Geisler and Perry's software uses the CPU to transform images that have been loaded into system memory from some external source, and copies the resulting image into video memory. Since our scenes are rendered by the GPU in video memory, we would like the GPU to do all of our graphics processing; to use the GPU to produce a scene and the CPU to modify it afterward would involve transferring the contents of the frame buffer from video memory to system memory and back, which is such an expensive operation that performance would be unacceptable regardless of how fast the CPU could do its part. Furthermore, by using the GPU, we can achieve much greater pixel processing power. Geisler and Perry's software is able to process one 640 × 304 color image at 28 fps on a 1500 MHz CPU; by using a modern GPU, we are able to render and process two 800 × 600 images at 60 fps while leaving the CPU available to do other work. The remainder of this section will present a brief description of the specific methods used to make the GPU perform this transformation.

To begin with, the blending algorithm takes as input a source image and a resolution map. The source image is simply the picture that would be seen before any blending takes place (i.e., the original 3-D scene rendered by the GPU). The resolution map is a grayscale image that tells the algorithm how much it should blur each pixel in the source image. As seen in Figure 1, a pixel value of 0.0 (black) on this map indicates that the corresponding region of the source image will be shown at full resolution (i.e., not blended at all), while a value of 1.0 (white) indicates that the corresponding region is to be blurred by the maximum amount.

In the experiments that have been conducted thus far using this technique, a pre-generated resolution map depicting resolution levels across retinal eccentricities expected from a peripheral field loss is loaded from an image file. The resolution map is then projected onto the inside of a sphere, which is rotated according to data received in real-time from an eye-tracking system, to produce the final resolution map for each frame (see Figure 2). This transformation allows the center of the resolution map, corresponding to the region of highest resolution, to remain centered in front of the fovea of participants. This in turn creates the perception of the simulated field loss "following" the gaze of participants, as one would expect from a real field loss.

Once the resolution map is loaded and the image of the current scene rendered, the first step in combining them according to Geisler and Perry's method is to produce the multiresolution pyramid from the source image. To produce this structure, the scene is rendered onto a surface, and that surface is then used as a texture to be rendered onto a smaller surface. This new surface is then used as a texture to render an even smaller surface, and so on, to whatever depth the predetermined pyramid prescribes.

However, in order to gradually decrease spatial resolution with increasing eccentricity, the program must do more than simply down-sample at each step; it must also blur each level before down-sampling it. This is accomplished by convolving with a 3 × 3 Gaussian kernel. When a texture is rendered onto a surface, the hardware will linearly interpolate between pixels from the source image when they do not line up exactly with the pixels from the destination surface. Consequently, if the GPU shifts the source image to the left by half a pixel and renders it to a surface, the GPU effectively convolves the image by the kernel [(0 1 1)]. If the GPU had shifted the image to the right it would have convolved the image by the mirror image of the same kernel [(1 1 0)]. If both shifts are completed and then added together, the image is convolved by [(1 2 1)], which is the one-dimensional form of the kernel that is needed for the present task. To extend this to two dimensions, four copies of the source image are added together: the image shifted up and left, up and right, down and left, and down and right, by half a pixel each (see Figure 3). The multitexturing features of the GPU can be used to perform this entire convolution in a rendering pass. The GPU will also down-sample if the image is rendered to a smaller surface than the source image, and this down-sampling can be done in the same rendering pass as the convolution. (Note: nVIDIA [Santa Clara, CA] had a good demo of this effect.)

Finally, the layers of the multiresolution pyramid need to be combined according to the value of the resolution

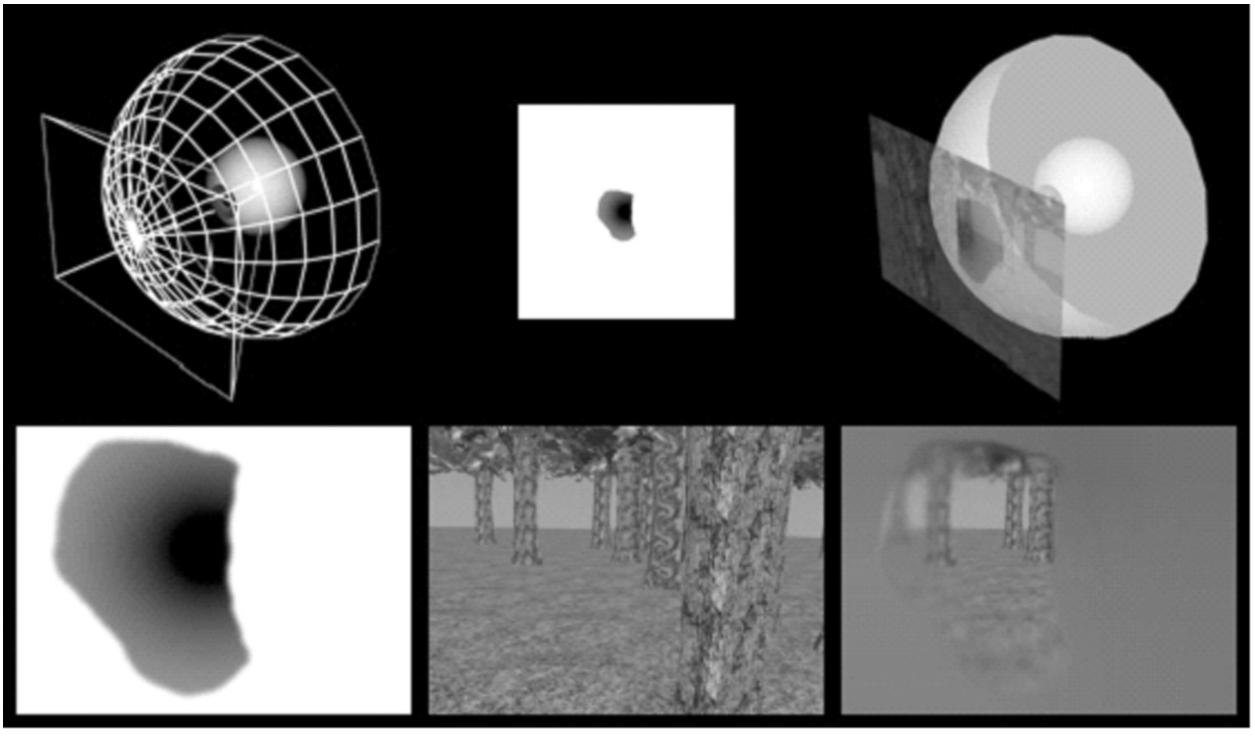

Figure 2. A picture depicting how the resolution map for each frame is created using the original resolution map projected onto a sphere and modified for eye-movements via information from the eye-tracker. The bottom images, from left to right, are the visual field map being used, the scene as originally rendered, and the final result that is displayed on screen.

map at each pixel. Where the value of the resolution map is 0.0, the pixel from the highest resolution image will be chosen. Where the value is 1.0, the pixel from the lowest resolution image will be chosen. If there are 5 levels to the pyramid, for example, then the values 0.25, 0.5, and 0.75 will map to the pixels of the 2nd, 3rd, and 4th layers, respectively. Pixels for values that fall between layers will be chosen by linearly interpolating between the pixels of the two layers bracketing them.

This linear interpolation is implemented with alpha shading, which allows the GPU to render an image onto another image using an alpha value to specify the opacity of the overlaying image (that is, whether to use the overlaying image's value, the original image's value, or some linear combination of the two). Each layer of the pyramid is rendered, iteratively, to the screen, starting with the highest resolution image. For each layer, a pixel shader is used to slice out the band of values relevant to that layer and expand that range. For example, when rendering the 3rd layer of a 5-layer pyramid, the pixel shader will perform a subtraction and a multiplication to map the range [0.25, 0.5] from the resolution map to the range [0,1] to be used as the alpha value for the layer. Values outside that range are simply clamped to [0,1], so that every layer will overwrite the pixel rendered on the previous pass up until the appropriate layer is blended in, and every layer following the one in question becomes completely transparent (see Figure 4). Since the pixel shader must process each pixel even if the resulting alpha value is 0, the contents of the mask do not impact performance; a mask that is entirely black will take as long to apply as one that is entirely white, even though applying the former is effectively a null operation.

Rendering an 8-level pyramid on two 800 × 600 displays simultaneously was done at 30 frames per second (FPS) on a GeForce FX 5900. However, following data collection it was discovered that the GPU uses a method of deferred rendering to improve its performance in terms of frame rate. This is transparent to the developer and adds 2–3 frames of latency to the VR system (in this case 66–100 msec). This makes sense for typical applications such as 3-D games, where such small latencies aren't noticed as long as frame rate is maintained. However, to eliminate this behavior one can lock the frame buffer after render-

$$\begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{1} & 2 & 1 \\ 2 & 4 & 2 \\ \mathbf{1} & 2 & 1 \end{bmatrix}$$

Figure 3. Convolving with a 3×3 Gaussian kernel.

ing the scene, as if reading from it, forcing the driver to immediately execute all pending commands before allowing the program to proceed. This comes with a penalty to frame rate. However, by upgrading to a more recent GPU (GeForce 6800 Ultra), we found that we could process two screens at 60 FPS, the refresh rate of our head-mounted LCD displays, even when forcing the GPU to flush the rendering pipeline immediately after every frame. Using this method can reduce the latency contributed by the rendering process to a minimum of 16.7 msec in some environments, although actual performance will depend on the complexity of the scene.

After the data for the first part of the experiment described in this article were collected, a third group of delays were tested, using the method described above to reduce latency. The environment used in the experiment described in this paper is the most complex of any we've used to date, and the rendering process was measured to contribute ~25–30 msec latency, for a total system latency of ~60 msec, even when no mask is applied. With a mask, the system runs at 30 fps, and the rendering process contributes ~55–60 msec latency, for a total system latency of ~90 msec. Using a simple environment, the minimum rendering latency is 16.7 msec, for a total system latency of ~46.7 msec. Whenever possible, future experiments will use environments which can be rendered at 60 fps with an 8-level pyramid applied. Rendering latency will then be 16.7–33.3 msec (~46.7–63.3 msec total).

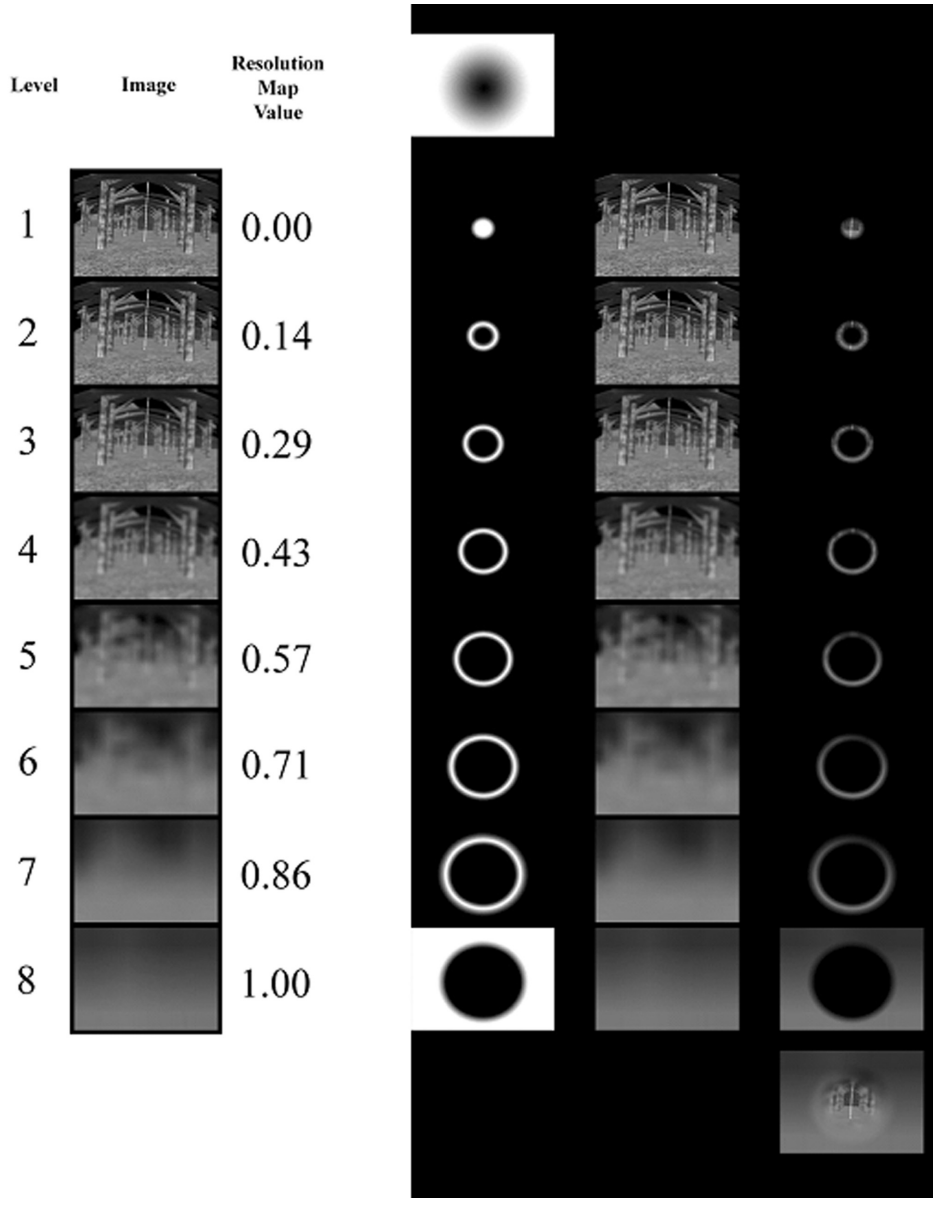

Figure 4. Illustration of how one frame from a scene is repeatedly down sampled and blurred. Regions of these copies are then added together and blended to produce the final image where the resolution at any point on the scene matches the resolution at the same point on the resolution map.

SYSTEM LAGS AND HUMAN PERFORMANCE

As Duchowski (2002) has noted, one of the major obstacles facing researchers utilizing gaze-contingent systems, or any immersive virtual reality system, is the effect of system lags on performance. While it is inevitable that delays will exist between the onset of participant movement and the time when displays are updated, previous research has shown that long delays can negatively impact performance and/or may cause feelings of motion sickness (Parkhurst & Niebur, 2002; Reingold et al., 2003; Stanney, Mourant, & Kennedy, 1998). Experimental data on the effect of lags on human performance are relatively rare and one must take into consideration both the systems and contexts in which the experiments are conducted. For example, using a sensory-motor task with an interactive system, MacKenzie and Ware (1993) found that a delay as short as 25 msec could cause changes in performance, though performance was not severely degraded until they increased the lag from 75 to 225 msec. Another study (So & Griffin, 1995) using a pursuit-tracking task found that lags as short as 80 msec could decrease performance. Examining the effects of lags on visual search performance using a gaze-contingent multiresolution display, Loschky and McConkie (2000) found that a lag of 45 msec caused changes in eye-movements, though no effect was seen on performance parameters. Given the variability in the available data on the effects of lags on human performance, and the lack of any data for the method of simulating visual field losses presented here, the following experiments were conducted to determine at what point performance would break down due to system delays. This is important to assess because the current method was designed to be used in behavioral studies examining the impact of peripheral field losses on human navigation and thus, its validity depends on minimizing the impact on behavior due to system lags.

During the course of the experiment, three groups of delays were tested. While all participants did not complete trials with all of the delays tested, all trials were completed using the same virtual environment and paradigm. Therefore, for the sake of clarity, the data for all of the delays tested across all the participants will be presented collectively.

METHOD

Participants

A total of 7 healthy adults, 4 men and 3 women (including coauthors K.A.T. and F.C.F.), with normal vision and no muscular-skeletal disorders participated in this experiment. The age of the participants ranged from 20 to 51 years old, with a mean age of 34 years. All participants completed a vision test, which included measuring their visual acuity and contrast sensitivity, to assure that they fell within the normal range. All participants gave written consent and all participants except K.A.T. and F.C.F. were compensated for their time.

Materials

An immersive virtual forest was created using 3-D Studio Max software (Discreet, Montreal). The program was exported to a graphics engine developed in-house (coauthor J.C.H.) with C++ and Microsoft's DirectX. The graphics program used the output from a HiBall head tracker attached to the head-mounted display (HMD) together with the imported scene to determine the subject's current point of view in the environment. Perspective views of the environment were displayed in the HMD using a GeForce3 graphics board (nVIDIA, Santa Clara, CA). The forest consisted of 29 trees, only one of which was the target. The target tree differed from the other trees in terms of its bark, which had a pattern that swirled as opposed to the vertical, linear pattern of the bark on the other trees. Forty-five unique configurations were created, in which the target tree was located in one of five positions on a horizontal line 10 m from the starting position. The five positions were 0 m, $\pm$0.333 m, or $\pm$0.667 m from the center of the line, corresponding to offset angles of 0º, $\pm$1.9º or $\pm$3.8º from the starting position. Between the target and starting position, three obstacle trees were placed in one of four positions on horizontal lines placed at 2 m, 4.667 m, and 7.334 m from the starting position, with the restriction that exactly one tree be placed on each line per configuration. Positions of the obstacle trees were either $\pm$0.333 m or $\pm$0.667 m from the center of each line. These corresponded to offset angles of $\pm$9.46º and $\pm$18.43º for the obstacles at 2 m, $\pm$4.09º and $\pm$8.13º for the obstacles 4.667 m, and $\pm$2.6º and $\pm$5.19º for the obstacles at 7.334 m. Two other distractor trees were placed in locations corresponding to corners in the actual laboratory walls in order to discourage participants from walking too close to them during the experiment. All of the remaining 23 trees were placed along the periphery such that the participants were unable to walk into them within the confines of the actual laboratory space.

Apparatus

Head-tracking. A HiBall-3000 Optical Tracker (3rd Tech, Chapel Hill, NC) was used to monitor head position and orientation. Infra-red LEDs were housed on the ceiling tiles of the testing room and their signal was detected by optical sensors mounted in a holder that was attached to the top of the headset. Head position and orientation were sampled every 7 msec. Tracker resolution is reported to be 0.2 mm, with an angular precision less than 0.03º. The output of the head tracker was filtered using an exponential smoothing function with an 80-msec time constant (with the exception of one block of head lag trials that one participant completed with an average 50 filter to increase the delay lengths). Walking path and point of view were inferred from the head position and orientation data collected.

Eye-tracking. To monitor eye position, single chip micro-cameras with an infrared bandpass film filter were housed within the headset in front of each eye. The luminance of the eyes was increased by adding infrared LEDs to the head-set cavity. The eye images were captured and digitized using an image acquisition board with high-resolution, 10-bit digitization (IMAQ PCI-1409, National Instruments Corp., Austin, TX). Pupil tracking was performed using software developed in-house by coauthor L.H. The algorithm consisted of calculating the geometric center of a threshold value within a specified region-of-interest. A 5-point calibration was performed prior to testing each block of trials, and a drift-correction calibration was performed before each trial. Eye and head position coordinates were communicated to other PCs via the Ethernet connection with TCP protocol. (In order to minimize the time delay of Internet communication between two PCs, TCP_NODELAY was turned on by calling the setsockopt function.)

Head-mounted display. The display device was a head-mounted video display system (a modified Low Vision Enhancement System developed by Robert Massof at the Wilmer Eye Institute). The headset contained two color microdisplays (SVGA, 800 $\times$ 600 3-D OLED Microdisplay, Emagin Corp.). The field of view of each was 48º (H) $\times$ 38º (V), with spatial resolution approximately 0.06º/pixel. The displays have a refresh rate of 60 Hz. Spatially offset images are sent to each display producing a stereo view.

Eye- and head-based lags. To determine the inherent lag in the system, we compared the successive images of two recorded videos,

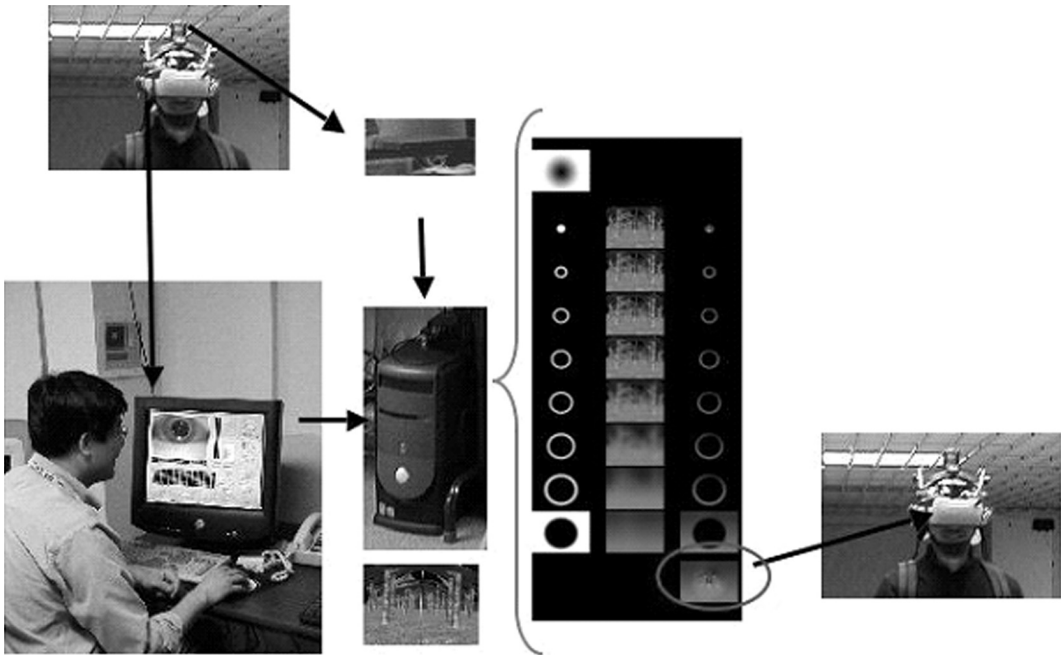

Figure 5. Schematic illustration of the system used. In the top left corner is the headset worn by participants. Arrows indicate the separate flows of information coming from the headset and the eye-tracking software to the GPU that allow for purely head-based or eye-based lags to be introduced.

taken at the time of a sudden jerk of the headset: one camcorder recorded the movement of the head tracker and the other camcorder recorded the display. A single tone was applied to the audio channel of the two camcorders and used to trigger the capture by the video capture board (IMAQ PCI-1409, National Instruments Corp., Austin, TX). Each field of the NTSC video image was digitized, and the time difference between corresponding events determined. The result showed an 8-field delay (133.3-msec time lag) from the beginning of the headset movement to the corresponding change in the image on the display. When a mask was added to the display, the delay increased to 10-fields, equivalent to a 166.7-msec time lag. As noted above, it was discovered that by locking the frame buffer system delays could be decreased. While the first two groups of delays in the current experiment were tested before this discovery was made (increasing the minimum lag time possible to 167 msec when a mask was used), the final group of delays was tested with the frame buffer locked. Using the same technique described above, it was found that the minimum system delay with the frame buffer locked was 60 msec without a mask and 90 msec when a mask was added.

Additional lags were created through the use of a "buffer." Information coming in from the Hi-ball is sent to the CPU of a second computer that is then responsible for rendering new images based on the updated position and direction of gaze. Information coming from the eye-tracking program is also sent to the same computer, providing updated information on eye position and thus, where the mask should be placed over the current image (see Figure 5). The buffer in essence holds either the information from the Hi-ball or the eye-tracking program for the allotted number of frames, after which time, the information is sent on and a new image based on that information is created. As information from the Hi-ball is only used to determine the position and orientation of the participant's head and the information from the eye-tracking software is only used to determine the position of the eyes, holding information back from one program or the other before processing new images allows for the creation of purely head or eye-based lags.

However, because frame rate is decreased when a mask is added, the length of the additional delay incurred from buffering frames depends on the presence of a mask. Specifically, when the masking technique described in this paper was utilized to simulate a peripheral field loss, buffering 1 frame to create head or eye-based lags added 33.3 msec of delay. On the other hand, when no mask was added, buffering 1 frame to create head-based delays without a restricted FOV only added 16.7 msec of delay.

Procedure

After being fitted with the headset, all participants were led to the starting position by one of the experimenters. The participants were informed that they would be presented with a virtual forest and that their task was to walk to the tree with the swirled bark (i.e., the target tree) as quickly and efficiently as possible without hitting any of the other trees in the forest. Before beginning any experimental trials the experimenter checked to make sure that all participants could easily distinguish the target tree from the distractors.

Group 1. The first group of delays, tested on six of the seven participants, investigated the effect of eye lags on navigation performance by buffering 0, 2, or 4 frames to create eye lags of 166.7 msec, 233.3 msec, or 300 msec, respectively. A 20º FOV was used and all six participants completed this block of trials. For four of the six participants these eye lags were crossed with 0, 2, or 4 frames of head lag, creating head-based delays of 166.7 msec, 233.3 msec, or 300 msec, respectively. (The head lag trials were also completed with a 20º FOV, increasing delay length for each added frame from 16.7 msec to 33.3 msec). Throughout the experiment every block of trials was designed such that the five trials tested for each delay were completed in succession. Presentation order for delay lengths and type of delay (i.e., eye- or head-based lag) was randomly assigned.

Group 2. In the second group of delays tested, three of the original six participants returned to complete two more blocks of trials. All three participants completed another block of trials with a 20º FOV and 6, 8, or 10 frames buffered corresponding to eye lags of 366.7 msec, 433.3 msec, and 500 msec. Two of the three participants then completed an additional block of trials without a restricted FOV and 0, 4, 6, 8, 10, or 12 frames buffered to create head lags equal to 133.3 msec, 200 msec, 233.3 msec, 266.7 msec, 300 msec, or

333.3 msec, respectively. The third subject completed two blocks of trials without a FOV restriction and 6, 8, or 10 frames buffered. In the first block an exponential 80 filter was used creating head lags of 233.3 msec, 266.7 msec, or 300 msec. In the second block an average 50 filter was used creating head lags of 283.3 msec, 316.7 msec, and 350 msec, respectively. While the technique described in this paper was designed to simulate peripheral field losses, the head lag trials completed without a restricted FOV in this group were tested to allow for comparison of the participants' performances with and without a restricted FOV.

Group 3. In the third group of delays the frame buffer was locked, decreasing the minimum system delay without a restricted FOV to 60 msec as mentioned above. Two participants (one participant had participated in Group 1 and Group 2, the second participant had not participated before) completed two blocks of fifteen trials each. While both blocks of trials tested head-based delays, one block was completed with a 20º FOV while the other block was completed without any simulated field loss. For the block completed with a 20º FOV, 0, 1, or 2 frames were buffered to create head-based delays of 90 msec, 123.3 msec, and 156.7 msec, respectively. For the block of trials completed without a FOV restriction, 1, 3, or 5 frames were buffered to create head-based delays of 76.7 msec, 110 msec, and 143.3 msec, respectively. It should be noted that due to the 30 msec difference in base system delays across the two conditions it was not possible to create equivalent delay lengths. The delay lengths tested in this group of trials were therefore chosen in an attempt to make the delays as similar as possible while exploring the minimum delays possible under the current system limitations. Block order was counterbalanced across the two participants and the order of the delays within each block was randomly assigned.

RESULTS AND DISCUSSION

Individual mean trial times (the time from when the scene was first displayed until participants reached the target tree) for the five trials completed for each delay were computed and from these overall means and standard errors for each delay were determined. The overall mean trial times for all head and eye-based delays are shown in Figure 6. (The means for the head-based delay trials completed without a restricted FOV using an average 50 filter are plotted without error bars as only one participant completed these trials.)

In order to assess whether any breaks in performance could be identified for the three types of delays tested (eye-based delays with a restricted FOV, head-based delays with a restricted FOV, and head-based delays without a restricted FOV), a bilinear fit was applied to the overall mean trial times. The equation for a bilinear fit, $y = a + b * (\text{delay} - \text{break}) * (\text{delay} > \text{break})$, was obtained from and applied in the statistical package JMP (Cary, NC). Results of the model show breaks in performance at delays of 233.3 msec for both the head-based delays completed without a restricted FOV and eye-based delays (completed with a restricted FOV). For the head-based delays completed with a restricted FOV, a break point was found at 166.7 msec.

Overall, there are two main findings from the present study. First, inspection of Figure 6 shows that trial times for both eye and head-based delays with a restricted FOV tended to be longer than mean trial times for equivalent head-based delays without a restricted FOV. Second, the break points for all three conditions are well above the minimum delays at which the system currently runs using the present environment. Furthermore, as noted earlier, the environment used in this experiment was highly detailed and therefore minimum system delays when using different environments in future studies will probably be even lower. The break points observed in the present study also fall within the range proposed by Wloka (1995) for virtual environments, wherein performance begins to be negatively impacted by 100 msec and a sense of presence in the environment is lost with lags over 300 msec. While it is possible that performance was already negatively impacted by the minimum delay of 60 msec, the break points found for the head-based delays without a restricted FOV suggest that the performance of participants is not wholly compromised by current the minimum delays. More importantly, however, the difference in performance between trials completed with a restricted FOV and trials completed without a restricted FOV suggests that changes in performance due to system delays do not occlude changes in performance due to restricting the peripheral visual field. As the current technique was created in order to allow for this type of measurement, the results suggest that this technique is a viable method for examining changes in performance across decreasing FOV.

CONCLUSION

The results of the present study suggest that valuable information can be gained about the effects of peripheral field losses using the simulation technique described in the beginning of this paper. Given the youth of virtual technologies, it will be some time before immersive virtual

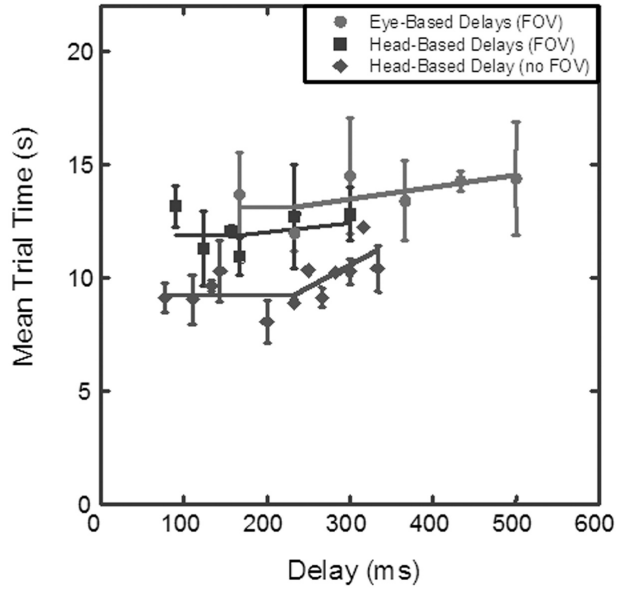

Figure 6. (a) Mean trial times as a function of delay length for the three delay types tested (eye based delays with restricted FOV, head-based delays with restricted FOV, and head based delays without restricted FOV). The equation used for the fits were $y = a + b * (\text{delay} - \text{break}) * (\text{delay} > \text{break})$. Error bars represent ±1 *SE* of the group means.

systems will be able to allow individuals to move through environments with refresh rates truly approaching "real time." And in fact, delays will always be present in any virtual system. But this should not discourage researchers from using these applications to learn about how humans interact with their environments. As noted in the introduction, simulating peripheral field losses while having participants walk through virtual environments not only gives researchers the ability to isolate the roles of different parts of the visual field during navigation but also provides participants with a controlled and safe environment in which to explore. While past research and the present data suggest that delays do affect behavior, they do not imply that behavioral changes from a field loss are hopelessly intertwined with changes caused by system delays. What they imply is that researchers must be careful in the questions that they ask and not forget to acknowledge that delays may be influencing any observed behavior. For now, questions should focus on how behavior changes as the variable of interest is manipulated over equivalent system delays and environments. For example, the question of how performance is affected as FOV decreases can still be examined. By looking at rates of change the effects of the inherent system delays are incorporated into all measures. Thus, while it would not be appropriate to state an absolute time observed in a virtual environment as the average speed at which someone with a 20º FOV walks, it would be appropriate to say that individuals with a 20º FOV tend to walk at a slower pace than those with a larger FOV. As with most research, the validity of any study depends not only on the manner in which a subject is investigated, but also on the questions that one seeks to answer.

AUTHOR NOTE

This research was supported by National Institutes of Health/National Eye Institute Grant EY-07839 to K.A.T. Correspondence concerning this article should be addressed to K. A. Turano, Lions Vision Center, Wilmer Eye Institute, 550 N. Broadway, Baltimore, MD 21205 (e-mail: kturano1@jhmi.edu).

REFERENCES

BERTERA, J. H., & RAYNER, K. (2000). Eye movements and the span of the effective stimulus in visual search. *Perception & Psychophysics*, **62**, 576-585.

CORNELISSEN, F. W., BRUIN, K. J., & KOOIJMAN, A. C. (2005). The influence of artificial scotomas on eye movements during visual search. *Optometry & Vision Science*, **82**, 27-35.

DUCHOWSKI, A. T. (2002). A breadth-first survey of eye-tracking applications. *Behavior Research Methods, Instruments, & Computers*, **34**, 455-470.

GEISLER, W. S., & PERRY, J. S. (1998). A real-time foveated multi-resolution system for low-bandwidth video communication. *SPIE Proceedings: Human Vision & Electronic Imaging*, **3299**, 294-305.

GEISLER, W. S., & PERRY, J. S. (1999). Variable-resolution displays for visual communication and simulation. *Society for Information Display*, **30**, 420-423.

GEISLER, W. S., & PERRY, J. S. (2002). Real-time simulation of arbitrary visual fields. In *Proceedings of the Eye Tracking Research & Applications Symposium 2002* (pp. 83-87). New York: ACM Press.

HAYMES, S., GUEST, D., HEYES, A., & JOHNSTON, A. (1996). Mobility of people with retinitis pigmentosa as a function of vision and psychological variables. *Optometry & Vision Science*, **73**, 621-637.

LOSCHKY, L. C. (2003). *Investigating perception and eye-movement control in natural scenes using gaze-contingent multi-resolution displays.* Unpublished doctoral dissertation, University of Illinois at Urbana-Champaign.

LOSCHKY, L. C., & MCCONKIE, G. W. (2000). User performance with gaze contingent displays. In *Proceedings of the Eye Tracking Research & Applications Symposium 2000* (pp. 97-103). New York: ACM Press.

LOVIE-KITCHIN, J., MAINSTONE, J., ROBINSON, J., & BROWN, B. (1990). What areas of the visual field are important for mobility in low vision patients? *Clinical Vision Sciences*, **5**, 249-263.

MACKENZIE, I. S., & WARE, C. (1993). Lag as a determinant of human performance in interactive systems. In *Proceedings of the ACM Conference on Human Factors in Computing Systems, INTERCHI 1993* (pp. 488-493). New York: ACM Press.

MARRON, J. A., & BAILEY, I. L. (1982). Visual factors and orientation-mobility performance. *American Journal of Optometry & Physiological Optics*, **59**, 413-426.

PARKHURST, D. J., & NIEBUR, E. (2002). Variable-resolution displays: A theoretical, practical, and behavioral evaluation. *Human Factors*, **44**, 611-629.

RAYNER, K. (1998). Eye movements in reading and information processing: 20 years of research. *Psychological Bulletin*, **124**, 372-422.

RAYNER, K., & BERTERA, J. H. (1979). Reading without a fovea. *Science*, **206**, 468-469.

REINGOLD, E. M., & LOSCHKY, L. C. (2002). Saliency of peripheral targets in gaze-contingent multiresolution displays. *Behavior Research Methods, Instruments, & Computers*, **34**, 491-499.

REINGOLD, E. M., MCCONKIE, G. W., & STAMPE, D. M. (2003). Gaze-contingent multiresolution displays: An integrative review. *Human Factors*, **45**, 307-328.

SO, R. H., & GRIFFIN, M. J. (1995). Effects of lags on human operator transfer functions with head-coupled systems. *Aviation, Space, & Environmental Medicine*, **66**, 550-556.

STANNEY, K. M., MOURANT, R. R., & KENNEDY, R. S. (1998). Human factors issues in virtual environments: A review of the literature. *Presence*, **7**, 327-351.

WATSON, B., WALKER, N., HODGES, L. F., & WORDEN, A. (1996). *Effectiveness of peripheral level of detail degradation when used with head-mounted displays* (Tech. Rep. No. GIT-GVU-96-04). Atlanta: Georgia Institute of Technology.

WATSON, B., WALKER, N., HODGES, L. F., & WORDEN, A. (1997). Managing level of detail through peripheral degradation: Effects on search performance with a head-mounted display. *ACM Transactions on Computer–Human Interaction*, **4**, 323-346.

WLOKA, M. M. (1995). Lag in multiprocessor virtual reality. *Presence*, **4**, 50-63.

(Manuscript received August 2, 2005;
revision accepted for publication June 14, 2006.)