Behavior Research Methods, Instruments, & Computers
2002, 34 (4), 573-591

3-D eye movement analysis

ANDREW DUCHOWSKI, ERIC MEDLIN, NATHAN COURNIA, HUNTER MURPHY,
ANAND GRAMOPADHYE, SANTOSH NAIR, JEENAL VORAH, and BRIAN MELLOY
Clemson University, Clemson, South Caroliina

This paper presents a novel three-dimensional (3-D) eye movement analysis algorithm for binocular eye tracking within virtual reality (VR). The user's gaze direction, head position, and orientation are tracked in order to allow recording of the user's fixations within the environment. Although the linear signal analysis approach is itself not new, its application to eye movement analysis in three dimensions advances traditional two-dimensional approaches, since it takes into account the six degrees of freedom of head movements and is resolution independent. Results indicate that the 3-D eye movement analysis algorithm can successfully be used for analysis of visual process measures in VR. Process measures not only can corroborate performance measures, but also can lead to discoveries of the reasons for performance improvements. In particular, analysis of users' eye movements in VR can potentially lead to further insights into the underlying cognitive processes of VR subjects.

A common goal of eye movement analysis is the detection of fixations in the eye movement signal over the given stimulus or within stimulus regions of interest (ROIs). Most techniques rely on the measurement of visual angle, where it is often tacitly assumed that the head is located at a fixed distance to, and usually also perpendicular to, the stimulus screen. Applicable signal analysis techniques can be grouped into three broad categories: position variance, velocity based, and ROI based. A good classification of current techniques is given by Salvucci and Goldberg (2000; an earlier classification by Anliker, 1976, is also relevant).

In position-variance schemes, the visual angle is used to threshold the stationary portion of the signal (e.g., in terms of position). For example, if gaze remains invariant in an area subtending 2º–5º of visual angle for 300 msec, this portion of the signal is deemed a fixation. In velocity-based schemes, the speed of successive data points is used to distinguish fixations from saccades (the fast, often ballistic, eye movements used to reposition the fovea). The latter analysis is usually accomplished by thresholding eye movement velocity, expressed in degrees of visual angle per second. Anywhere the signal exhibits fast velocity (above threshold), this portion of the signal is deemed a saccade, and conversely, everywhere else, the signal can be considered a fixation (or some other type of relatively slow eye movement, such as smooth pursuit). The velocity-based saccade detection method can therefore be used as a type of delineation scheme to find fixations in the eye movement signal and is adopted as the underlying strategy for eye movement analysis in virtual reality (VR). It should be noted that for identifying fixations in raw eye movement data recorded at a fixed sampling rate, both position-variance and velocity-based schemes are virtually identical.

The traditional two-dimensional (2-D) eye movement analysis approach starts by measuring the visual angle of the object under inspection between a pair (or more) of raw eye movement data points in the time series (i.e., composed of a sequence of the so-called *point of regard*, or POR, denoted by $[x_i, y_i]$). Given the distance between successive POR data points, $r = \| (x_i,y_i),(x_j,y_j) \|$, the visual angle, η, is calculated by the equation $\eta = 2\tan^{-1}(r/2D)$, where D is the (perpendicular) distance from the eyes to the viewing plane, as is shown in Figure 1.

The arctangent approach assumes that D is measured along the line of sight, which is assumed to be perpendicular to the viewing plane. In general, however, the assumption of a perpendicular visual target plane does not hold. This has a significant implication for the measurement of visual angle, since the farther away from the central axis eye movements are made, the smaller the visual angle. Upon further inspection of Figure 1, the visual angle corrected for this foreshortening effect is calculated as

$$\theta = \beta - \alpha = \tan^{-1}\frac{r+d}{D} - \tan^{-1}\frac{d}{D},$$

where $d + r/2$ is the distance of the POR center from the projected central view axis. For large d (and constant r and D), $\eta > \theta$. That is, the traditional=] arctangent approach overestimates the visual angle at off-axis locations. An alternate calculation of the corrected visual angle θ can be made directly by examining the relationship between view vectors:

$$\theta = \cos^{-1}\frac{(\mathbf{P}-\mathbf{C})\cdot(\mathbf{Q}-\mathbf{C})}{\|(\mathbf{P}-\mathbf{C})\|\|(\mathbf{Q}-\mathbf{C})\|},$$

where **P**, **Q**, and **C** define the three-dimensional (3-D) extents of the POR and head center, respectively—for ex-

This work was supported in part by University Innovation Grant 1-20-1906-51-4087, NASA Ames Task NCC 2-1114, and NSF CAREER Award IIS-9984278. We thank Dario Salvucci and a second, anonymous referee for their helpful comments on the manuscript. Correspondence concerning this article should be addressed to A. Duchowski, Department of Computer Science, Clemson University, 451 Edwards Hall, Clemson, SC 29634-0974 (e-mail: andrewd@cs.clemson.edu).

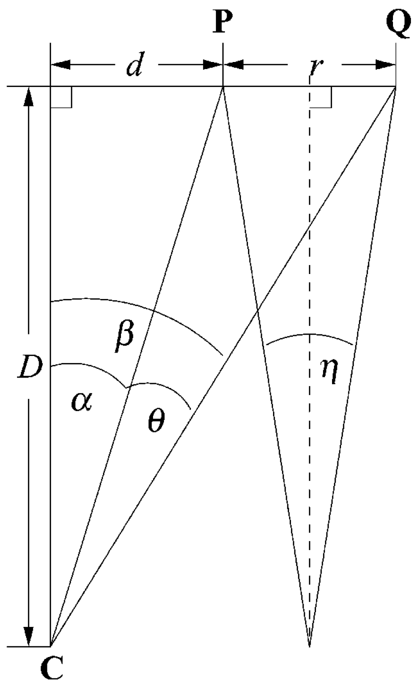


Figure 1. Two-dimensional geometry.

ample, **P** = (d,0,D) and **Q** = ($d+r$,0,D) if **C** defines the origin and gaze is recorded along the horizontal viewing axis. The vector-based approach forms the basis of our 3-D eye movement analysis.

The method of calculation of the visual angle notwithstanding, eye movement analysis generally depends on the size of fixated element r, which in turn is dependent on the viewing distance D. Note that r and D, expressed in like units (e.g., pixels or inches), are dependent on the resolution of the screen on which the POR data was recorded. A conversion factor is usually required to convert one measure to the other (e.g., screen resolution in dots per inch [dpi] converting D to pixels). The visual angle θ and the difference in timestamps Δt between the POR data points allows velocity-based analysis, since $\theta/\Delta t$ gives eye movement velocity in degrees of visual angle per second.

We present a velocity-based eye movement analysis algorithm in three dimensions, applicable to the 3-D eye movement data recorded during immersion in a virtual environment (VE; Duchowski, Medlin, Gramopadhye, Melloy, & Nair, 2001). Traditional 2-D eye movement analysis methods can be applied directly to raw POR data in the eye tracker reference frame. As a result, identified fixations could then be mapped to world coordinates to locate fixated ROIs within the VE. We chose a different approach by mapping raw POR data to world coordinates first, followed by eye movement analysis in three-space. We favored this approach because the calculated gaze points in three-space provide a composite 3-D representation of both left- and right-eye movements. Applying the traditional 2-D approach prior to mapping to (virtual) world coordinates suggests a component-wise analysis of left- and right-eye movements (in the eye tracker's reference frame) possibly ignoring depth (as generally would be the case with monocular eye tracking). In three dimensions, depth information, derived from binocular eye tracking, is implicitly taken into account prior to analysis.

The paper is organized as follows. First, we will describe our operational platform and derive applicable gaze vector calculations including a 2-D-to-3-D mapping required for the calculation of gaze points in the VE. Device and software calibration techniques, developed specifically to address the use of a binocular eye tracker, will then be discussed. The novel 3-D eye movement analysis algorithm will then be presented, followed by an evaluation of the algorithm featuring a comparative analysis of several velocity and acceleration filters for saccade detection. Finally, we will describe our application testbed, a VE used for aircraft visual inspection training, and will discuss results obtained from experiments conducted in the VE.

EYE TRACKING IN VIRTUAL REALITY

Our primary rendering engine is a dual-rack dual-pipe Silicon Graphics Onyx2 InfiniteReality2 system with eight raster managers and eight MIPS R12000 processors, each with an 8-MB secondary cache.[1] It is equipped with 8 Gb of main memory and 0.5 Gb of texture memory.

Multimodal hardware components include a binocular eye tracker mounted within a Virtual Research V8 head-mounted display (HMD). The V8 HMD offers 640 × 480 pixel resolution per eye with individual left- and right-eye feeds. HMD position and orientation tracking is provided by an Ascension 6 Degree-of-Freedom (6DOF) Flock Of Birds (FOB). The HMD is shown in Figure 2 (inset), with the FOB sensor just visible on top of the hel-

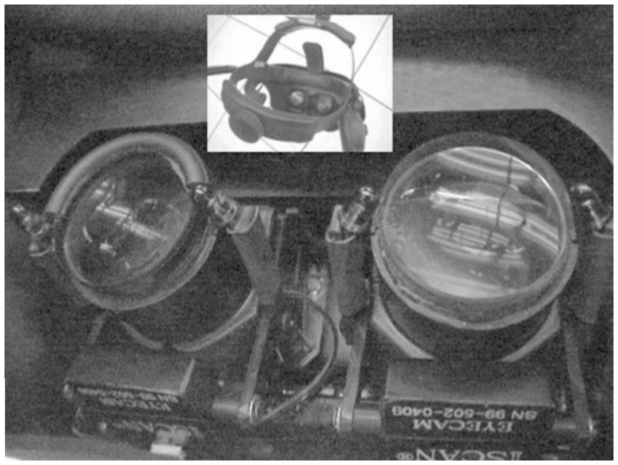


Figure 2. Binocular eye tracker optics (with head-mounted display inset above).

met. A 6DOF-tracked hand-held mouse provides a means to represent a virtual tool for the user in the environment.

The eye tracker is a video-based corneal reflection unit built jointly by Virtual Research and ISCAN. Each of the binocular video eye trackers is composed of a miniature camera and infrared light sources, with the dual optics assemblies connected to a dedicated personal computer (PC). The ISCAN RK-726PCI high resolution pupil/corneal reflection processor uses corneal reflections (first Purkinje images) of infrared LEDs mounted within the helmet to measure eye movements. Figure 2 shows the dual cameras and infrared LEDs of the binocular assembly.

Mounted below the HMD lenses, the eye imaging cameras peer upward through a hole cut into the lens stem, capturing images of the eyes reflected by a dichroic mirror placed behind the HMD lenses. The processor typically operates at a sample rate of 60 Hz; however, while in binocular mode, our measured sample rate decreases to 30 Hz. The subject's eye position is determined with an accuracy of approximately 0.3° over a ±20° horizontal and vertical range, using the pupil/corneal reflection difference. The maximum spatial resolution of the calculated POR provided by the tracker is 512 × 512 pixels per eye.

The binocular eye-tracking assembly allows the measurement of vergence eye movements, which in turn provides the capability of calculating the 3-D virtual coordinates of the viewer's gaze. Using the vendor's proprietary software and hardware, the PC calculates the subject's real-time POR from the video eye images. In the current VR configuration, the eye tracker is treated as a black box delivering real-time eye movement coordinates (x_l, y_l, t) and (x_r, y_r, t) over a 19.2-Kbaud RS-232 serial connection and can be considered as an ordinary positional tracking device.

Eye Tracker Coordinate Mapping

Several processing steps are required to accurately calculate the user's gaze within the environment. Once the gaze direction has been obtained, the resultant gaze vector is used to identify fixated regions in the environment by first calculating the gaze/environment intersection points and then applying signal analysis techniques to identify fixations.

Given the extents of both application and eye tracker screen coordinates, a simple linear interpolation mapping is used to map raw POR data to the graphics screen coordinates (Duchowski et al., 2000). Specifically, 2-D eye tracker data expressed in eye tracker screen coordinates must be mapped to the 2-D dimensions of the near viewing frustum. The 3-D viewing frustum employed in the perspective viewing transformation is defined by the parameters *left*, *right*, *bottom*, *top*, *near*, and *far*. Figure 3 shows the dimensions of the eye tracker screen (left) and the dimensions of the viewing frustum (right).

To convert the eye tracker coordinates (x', y') to graphics coordinates (x, y), the following linear interpolation mapping is used:

$$x = \text{left} + \frac{x'(\text{right} - \text{left})}{512} \tag{1}$$

and

$$y = \text{bottom} + \frac{(512 - y')(\text{top} - \text{bottom})}{512}. \tag{2}$$

Since the eye tracker origin is at the top left of the screen and the viewing frustum's origin is at the bottom left (a common discrepancy between imaging and graphics applications), the term $(512 - y')$ in Equation 2 handles the necessary y-coordinate mirror transformation.

The above coordinate mapping assumes that the eye tracker coordinates are in the range [0,511]. In practice, the usable, or effective, coordinates will be dependent on (1) the size of the application window and (2) the position of the application window. Proper mapping between eye tracker and application coordinates is achieved through the measurement of the application window's extents in the eye tracker's reference frame. This is accomplished by

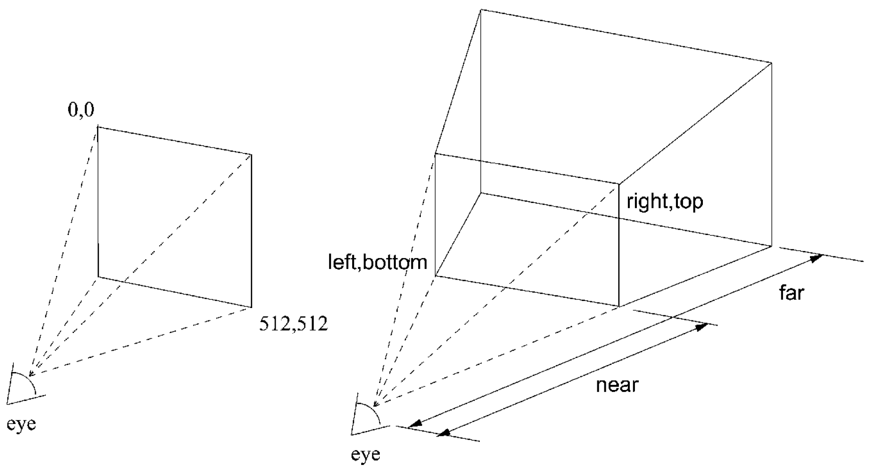


Figure 3. Eye tracker to three-dimensional viewing frustum screen coordinate mapping.

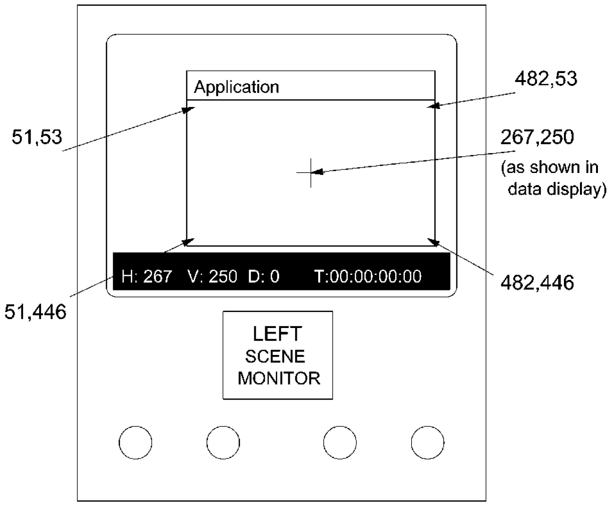


Figure 4. Mapping measurement example.

using the eye tracker's own fine-grained cursor movement and cursor location readout.

To obtain the extents of the application window in the eye tracker's reference frame, the application window's corners are measured with the eye tracker's cursor. These window extents are then used in the linear mapping equation. Figure 4 illustrates an example of a 600 × 450 application window as it would appear on the eye tracker scene monitor.

On the basis of the measurements shown in Figure 4, the linear coordinate mapping is

$$x = \frac{x' - 51}{(482 - 51 + 1)}(600) \tag{3}$$

and

$$y = 449 - \frac{y' - 53}{(446 - 53 + 1)}(450). \tag{4}$$

Although seemingly trivial, this mapping is key to proper calculation of the gaze vector in world coordinates from raw POR data and is also essential for alignment of target points displayed by the application program during calibration of the eye tracker. Correct registration between eye tracker and image coordinates is achieved if the linearly mapped computer-generated calibration target points align with the calibration points generated by the eye tracker. Because both coordinates are ultimately subject to the same optical distortions of the HMD (e.g., the pincushion effect), the linear mapping is sufficient for coordinate registration (Duchowski, 1998).

Gaze Vector Calculation

The calculation of gaze in three-space depends only on the relative positions of the two eyes on the horizontal axis. The parameters of interest are the 3-D virtual coordinates, (x_g, y_g, z_g), which can be determined from traditional stereo geometry calculations (Horn, 1986). Figure 5 illustrates the basic binocular geometry.

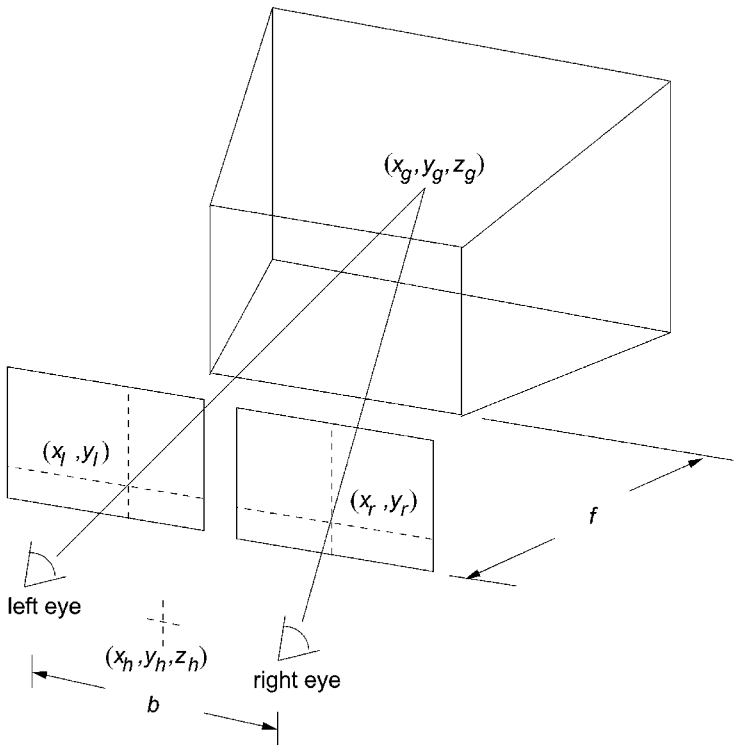


Figure 5. Basic binocular geometry.

Helmet tracking determines both helmet position and the (orthogonal) directional and up vectors, which determine head-centric coordinates. The helmet position is the origin, the helmet directional vector is the optical z-axis, and the helmet up vector is the y-axis.

Given instantaneous eye-tracked coordinates, (x_l,y_l) and (x_r,y_r), in the left and right image planes (mapped from eye tracker screen coordinates to the near view plane), and head-tracked head position coordinates, (x_h,y_h,z_h), the coordinates of the gaze point, (x_g,y_g,z_g), are determined by the following relations:

$$x_g = (1 - s)\,x_h + s(x_l + x_r)/2, \tag{5}$$

$$y_g = (1 - s)\,y_h + s(y_l + y_r)/2, \tag{6}$$

and

$$z_g = (1 - s)\,z_h + sf, \tag{7}$$

where $s = b/(x_l - x_r + b)$, b is the interpupillary distance at parallel vergence (looking at an infinitely distant object), and f is the distance to the near viewing plane along the head-centric z-axis.

Note that since the vertical eye-tracked coordinates y_l and y_r are expected to be equal (since gaze coordinates are assumed to be epipolar), the vertical coordinate of the central view vector defined by $(y_l + y_r)/2$ is somewhat extraneous; either y_l or y_r would do for the calculation of the gaze vector. However, since eye tracker data is also expected to be noisy, this averaging of the vertical coordinates enforces the epipolar assumption.

To enable collection of fixated points in the environment, it is necessary to calculate the intersection of the user's gaze with the environmental polygons. To calculate gaze direction, the gaze point is expressed parametrically as a point on a ray with origin (x_h,y_h,z_h), with the ray emanating along a vector scaled by parameter s. That is, rewriting Equations 5–7,

$$x_g = x_h + s\left(\frac{x_l + x_r}{2} - x_h\right),$$

$$y_g = y_h + s\left(\frac{y_l + y_r}{2} - y_h\right),$$

and

$$z_g = z_h + s(f - z_h),$$

or, in vector notation,

$$\mathbf{g} = \mathbf{h} + s\mathbf{v}, \tag{8}$$

where $\mathbf{h}$ is the head position, $\mathbf{v}$ is the central gaze vector, and s is the scale parameter as defined previously. The view vector $\mathbf{v}$ is obtained by subtracting the helmet position from the midpoint of the eye-tracked x-coordinate and focal distance to the near view plane—that is,

$$\begin{aligned}\mathbf{v} &= \begin{bmatrix}(x_l + x_r)/2\\(y_l + y_r)/2\\f\end{bmatrix} - \begin{bmatrix}x_h\\y_h\\z_h\end{bmatrix}\\ &= \mathbf{m} - \mathbf{h},\end{aligned} \tag{9}$$

where $\mathbf{m}$ denotes the left- and right-eye coordinate midpoint. To align the gaze vector with the current head orientation, it is first transformed to the instantaneous head-centric reference frame (instantaneous head orientation). This is accomplished by multiplying the gaze vector $\mathbf{v}$ by the orientation matrix returned by the head tracker. Given the 3-D gaze vector, $\mathbf{v}$, specified by Equation 9, Equation 8 gives the coordinates of the gaze point parametrically along a ray originating at the head position (x_h,y_h,z_h). The depth of the 3-D gaze point in world coordinates is valid only if $s > 0$.

Calculating Gaze Intersection Points

The computed gaze direction vector $\mathbf{v}$ is used for calculating gaze/polygon intersections via traditional ray/polygon intersection calculations commonly used in ray tracing (Glassner, 1989). These points, termed here the *gaze intersection points* (GIPs) for brevity, are each found on the closest polygon to the viewer intersecting the gaze ray, assuming all the polygons are opaque. Adapted to gaze in VR, this technique is similar to the traditional ray-casting approach to selection in virtual environments (Bowman & Hodges, 1997). For comparison, Tanriverdi and Jacob (2000) used a similar gaze-based ray-casting method for selection of objects. In their comparison of selection modalities, Tanriverdi and Jacob showed that interaction with eye movements was faster than interaction with hand-pointing (using a 3-D mouse). Our gaze-based selection mechanism is similar; however, our derivation of the gaze ray is slightly different, owing to our use of binocular eye-tracking optics.

Each gaze/polygon intersection point is found on the closest polygon to the viewer intersecting the gaze ray, assuming all the polygons are opaque. This polygon is found by testing all the polygons in the scene for intersection with the gaze ray. To find the intersection point $\mathbf{g}$ of the gaze ray with the closest polygon, a new interpolant t is obtained by calculating the gaze ray intersections with all scene polygons. All such intersections are examined for which $t > 0$.[2] Note that the ray/polygon intersection algorithm returns only the intersection point of the ray and the infinite plane defined by the polygon's face normal. Because the normal defines a plane of infinite extent, the point $\mathbf{g}$ must be tested against all of the polygon's edges, to establish whether the point lies inside the polygon. This is an instance of a solution to the well-known *point-in-polygon* problem. If the point $\mathbf{g}$ is bounded by the perpendicular planes defined by the polygon's edges, then $\mathbf{g}$ lies within the polygon; otherwise, it lies on the plane defined by the face normal, but outside the polygonal region. The resulting algorithm generates a scanpath constrained to lie on polygonal regions within the virtual environment. Provided the number of polygons is sufficiently small, the algorithm executes in real time.

DEVICE AND SOFTWARE CALIBRATION

In practice, determination of the scalar s (dependent on interpupillary distance, b) and focal distance f, used in

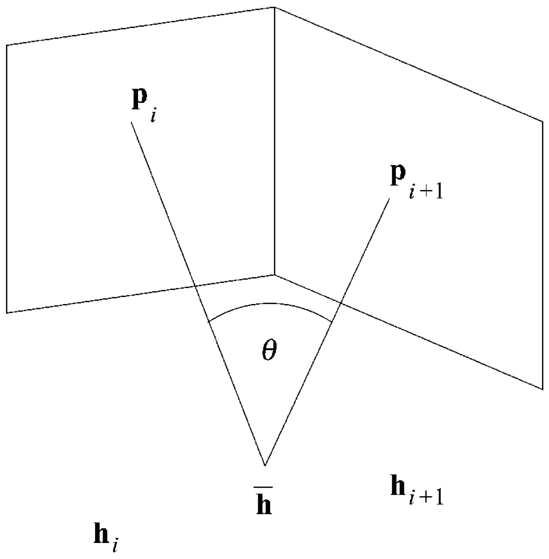


Figure 6. Eye movement analysis in three dimensions.

Equations 5–7, is difficult. Interpupillary distance is not easily measured in VR, since the left- and right-eye tracking components function independently. That is, there is no common reference point. Physical measurement of interpupillary distance outside VR—for example, at the start of the viewing session—is, of course, possible; however, conversion of such a measurement to VR coordinates is problematic (i.e., virtual coordinates are often unitless but are generally homogeneously scalable, depending on the required mapping between virtual and real dimensions). Preliminary experiments were conducted to informally gauge this problem. Calculated GIPs were compared against raw POR video footage. Frame-by-frame visual inspection of video footage revealed a discrepancy between calculated GIPs and the visual features subjects appeared to be fixating. Since this error appeared to be variable between but consistent within subjects and was thought to be related to the unknown interpupillary distance, a 3-D calibration procedure was designed to estimate the interpupillary distance scaling factor s empirically. The calibration procedure is currently specific to our application testbed (see below).

EYE MOVEMENT ANALYSIS

Operating directly on GIP data in (virtual) world coordinates, our initial fixation detection algorithm was based on an estimate of velocity. Given raw gaze intersection points in three dimensions, the velocity-based thresholding calculation is, in principle, identical to the traditional 2-D approach, with the following important distinctions.

1. The head position, h, must be recorded to facilitate the calculation of the visual angle.

2. Given two successive GIP data points in three-space, $p_i = (x_i, y_i, z_i)$ and $p_{i+1} = (x_{i+1}, y_{i+1}, z_{i+1})$, and the head position at each instance, h_i and h_{i+1}, the estimate of instantaneous visual angle at each sample position, θ_i, is calculated from the dot product of the two gaze vectors defined by the difference of the gaze intersection points and averaged head position:

$$\theta_i = \cos^{-1}\frac{v_i \cdot v_{i+1}}{\|v_i\|\|v_{i+1}\|}, i \in [0, n), \qquad (10)$$

where n is the sample size and $v_i = p_i - \bar{h}$ and $\bar{h}$ is the averaged head position over the sample time period. Head position is averaged since the eyes can accelerate to reach a target fixation point much more quickly than can the head (Watson, Walker, & Hodges, 1997).

With visual angle, θ_i, and timestamp difference between p_i and p_{i+1}, the same velocity-based thresholding is used as in the traditional 2-D case (see Figure 6). No con-

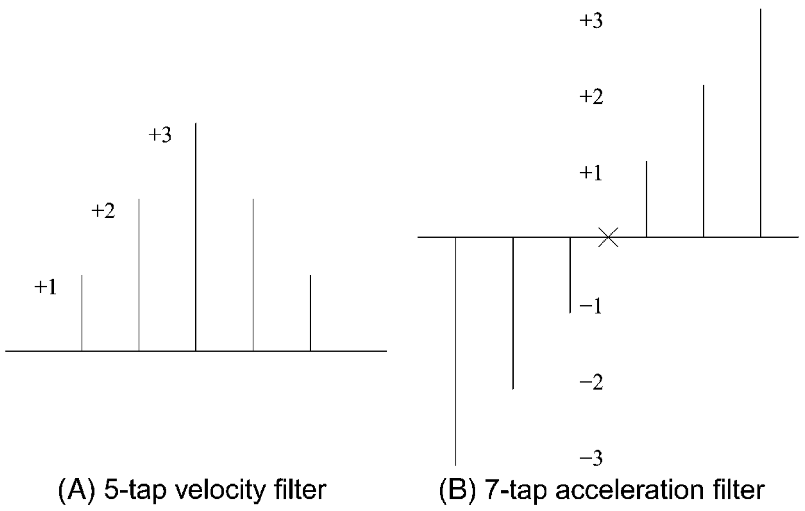


Figure 7. Finite impulse response filters.

version between screen resolution and distance to target is necessary, because all calculations are performed in world coordinates.

Although the algorithm generalizes to the use of wider filters (by changing the subscript $i+1$ to $i+k$ for $k > 1$) for improved smoothing, in our previous work we relied on a short two-tap filter to estimate velocity. That is, using Equation 10 to calculate θ_i, only two successive data points were used to calculate eye movement velocity. This is analogous to the calculation of velocity by using a convolution filter with coefficients {1,1}—that is, a two-tap finite impulse response (FIR) filter.

A preliminary study was conducted to evaluate the 3-D eye movement analysis algorithm. The results indicated that owing to the somewhat noisy signal analysis approach, the algorithm underestimated the identified number of fixations and fixation durations (Duchowski et al., 2001). This result was not wholly unexpected. The velocity-based saccade detection method is known to be a weak fixation detector when used in isolation. However, it is often a necessary first step to locating slow-moving eye movements, which can then be processed further to isolate and group fixation points.

Furthermore, as was expected, we noted a high degree of noise in the data. The two main sources of noise are, most likely, the eye tracker and the short filter used in the velocity-based algorithm. The eye tracker is inherently somewhat noisy and frequently delivers null POR values, usually coinciding with blinks. Sample data with null values for either the left or the right POR was previously automatically eliminated by our algorithm. Over all trials, we observed an estimated mean 10% data loss. Considering mean trial durations of 177 sec and a sample rate of 30 Hz, this data loss rate is quite high. The short filter used in the velocity-based analysis is another source of noise. The filter is mathematically appropriate for gauging velocity (when applied to saccade amplitude), but owing to its short length, it is known to be quite noisy. For more robust offline fixation analysis, a longer filter should be used. In the following sections, we will compare results of the short filter to longer versions of velocity and acceleration filters.

Velocity and Acceleration Filtering

To address excessive noise in the eye movement signal collected in previous studies, we began by replacing our two-tap FIR filter with a five-tap FIR filter, shown in Figure 7A.

Owing to its longer sampling window, the filter is more effective at signal smoothing (anti-aliasing). We also compared the results of the velocity filter's utility versus the use of an acceleration filter, following the work of Tole and Young (1981). The acceleration filter is shown in Figure 7B, and is convolved with eye movement velocity data as obtained via either the two-tap or the five-tap velocity filter. The filter responses resemble the real velocity and acceleration curves for a saccade characterized in Figure 8.

Our new algorithm calculates the velocity and acceleration at each instantaneous estimate of visual angle, θ_i. Note that θ_i is effectively a measure of instantaneous eye movement magnitude (i.e., amplitude), and, therefore, implicitly represents eye movement velocity. That is, the signal resembles the positively oriented velocity peaks shown in Figure 8B. Withholding division by the time difference between successive samples (Δt) facilitates the measurement of velocity with arbitrarily long filters.

Velocity is obtained via convolution with pattern-matching FIR filters of variable length. When convolved, these filters respond to sampled data with profiles matching that of the filter. These filters, denoted by h_k, are essentially unnormalized low-pass filters that tend to smooth and amplify the underlying signal. Division by the duration of the sampling window yields velocity—that is,

$$\dot{\theta}_i = \frac{1}{\Delta t} \sum_{j=0}^{k} \theta_{i+j} h_j, i \in [0, n-k),$$

expressed in degrees/second, where k is the filter length and $\Delta t = k - i$. We compare the performance of the five-tap filter with the previously implemented two-tap filter with coefficients {1,1} below.

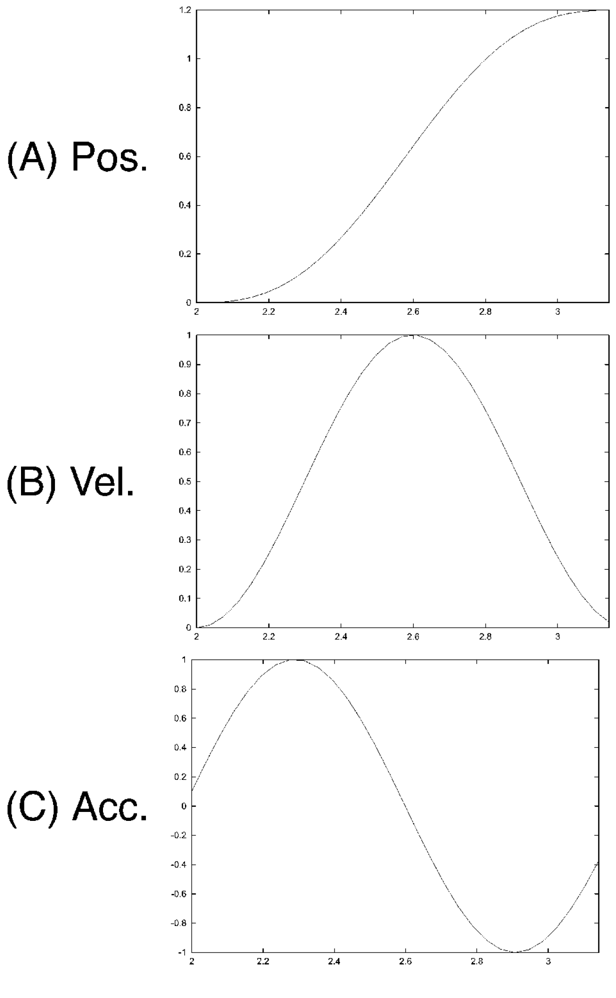


Figure 8. Characteristic saccade signal and filter responses.

Algorithm 1
Acceleration-Based Saccade Detection

Input: $p(n)$, gaze intersection points, $h(k)$, $g(k)$, velocity and acceleration filters, respectively
Output: classification of each p_i as fixation or saccade

```
// calculate instantaneous visual angle
for $i = 0$ to $n - 1$ do
    $\theta_i = \cos^{-1}(v_i \cdot v_{i+1} / \| v_i \| \| v_{i+1} \|)$
end for
// initialize accumulation arrays (convolution results)
for $i = 0$ to $n - k - 1$ do
    $\dot{\theta}_i = \ddot{\theta}_i = 0$
end for
// convolve with vel. filter
for $i = 0$ to $n - k - 1$ do
    for $j = 0$ to $k$ do
        $\dot{\theta}_i = \dot{\theta}_i + \theta_{i+j} * h_j$
    end for
end for
// convolve with acc. filter
for $i = 0$ to $n - k - 1$ do
    for $j = 0$ to $k$ do
        $\ddot{\theta}_i = \ddot{\theta}_i + \dot{\theta}_{i+j} * g_j$
    end for
end for
for $i = 0$ to $n - k - 1$ do
    // condition 1
    if $|\ddot{\theta}_i| \geq A$ then
        // condition 4 (implicit in loop)
        for $j = i + T_{\min}$ to $(n - k) - i \wedge (j - i) \leq T_{\max}$ do
            //conditions 2 & 3
            if $| \ddot{\theta}_{i+j} | \geq B \wedge sgn(\ddot{\theta}_{i+j}) \neq sgn(\ddot{\theta}_i)$ then
                for $l = i$ to $j$ do
                    $p_l$ = saccade
                end for
            else
                $p_l = fixation$
            end if
        end for
    end if
end for
```

Acceleration is obtained via a subsequent convolution of velocity, θ_i, with the acceleration filter, g_j, shown in Figure 7B. That is,

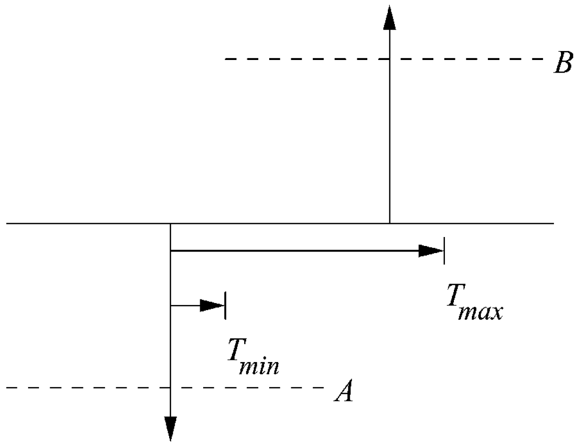
$$\ddot{\theta}_i = \frac{1}{\Delta t} \sum_{j=0}^{k} \dot{\theta}_{i+j} g_j, i \in [0, n-k),$$

where k is the filter length and $\Delta t = k - i$. The acceleration filter is essentially an unnormalized high-pass differential filter. The resulting value $\ddot{\theta}_i$, expressed in degrees/second^2, is checked against threshold A. If the absolute value of $\ddot{\theta}_i$ is greater than A, the corresponding gaze intersection point p_i is treated as the beginning of a saccade. Scanning ahead in the convolved acceleration data, each subsequent point is tested in a similar fashion against threshold B in order to detect the end of the saccade. Two additional conditions are evaluated to locate a saccade, as given by Tole and Young (1981). The four conditions are listed and illustrated in Figure 9.

Note that our velocity and acceleration filters differ from those used by Tole and Young (1981). This is because Tole and Young applied their filters (the reverse of ours, essentially) to the positional eye movement signal (p), whereas our filters are applied to the signal amplitude (θ). Pseudocode of the technique is presented in Algorithm 1.

Parameter Estimation

Thresholds are needed for saccade velocity, acceleration, and duration, since our fixation detection algorithm relies on the detection of saccades. Although eventually determined empirically, algorithm fine tuning was guided



1. $|\ddot{\theta}_i| > A$
2. $|\ddot{\theta}_{i+j}| > B$
3. $sgn(\ddot{\theta}_{i+j}) \neq sgn(\ddot{\theta}_i)$
4. $T_{\min} < j - i < T_{\max}$

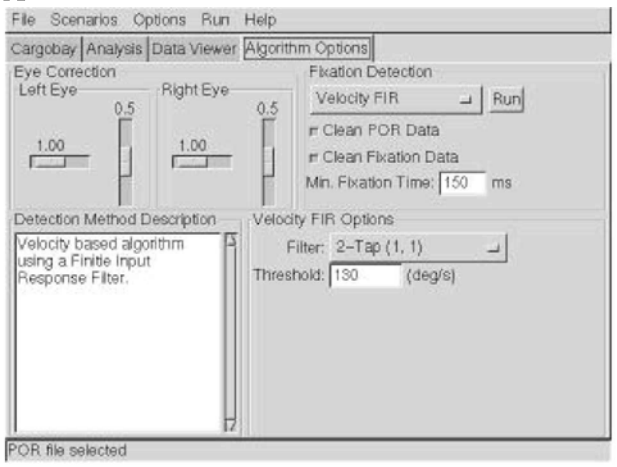
Figure 9. Acceleration thresholding.

by a review of the literature, briefly summarized here for context. While scanpath characteristics may be task dependent—that is, differing when one is looking at pictures than when one is reading—for the purpose of initial estimation of parameters, we assumed that, when one is looking at pictures, normal scanpaths are characterized by a number of saccades similar in amplitude to those exhibited during reading. This is largely a matter of convenience, since eye movement characteristics during reading are better established and more readily available than eye movement characteristics for scene viewing.

The duration of saccades is related in a nonlinear manner to their amplitude over a thousandfold range (3′–50°; Bahill, Clark, & Stark, 1975). Saccades of less than 15° or 20° in magnitude are physiologically the most important, since most naturally occurring saccades fall in this region. The saccade *main sequence* describes the relationships between saccade duration, peak velocity, and magnitude (amplitude). Because saccades are generally stereotyped, the relationship between saccade amplitude and duration can be modeled by the linear equation $\Delta t = 2.2\theta + 21$ (Knox, 2001). Peak velocity reaches a soft saturation limit up to about 15° or 20° but can range up to about 50°, reaching velocity saturation at about 1,000 deg/sec (Clark & Stark, 1975). In practice, the main sequence relationship between amplitude and velocity can be modeled by the asymptotic equation $\dot{\theta} = \lambda(1 - e^{-\theta/15})$, with velocity upper limit (asymptote λ) set to 750 deg/sec (Hain, 1999). For saccade detection via velocity filtering, we chose a threshold of 130 deg/sec for both two-tap and five-tap filters. Using the asymptotic model of the main sequence relationship between saccade amplitude and velocity (limited by 750 deg/sec), we reasoned that this threshold would effectively detect saccades of amplitude roughly greater than 3°. User-adjustable threshold settings for the velocity filter are shown in Figure 10A (bottom right quadrant).

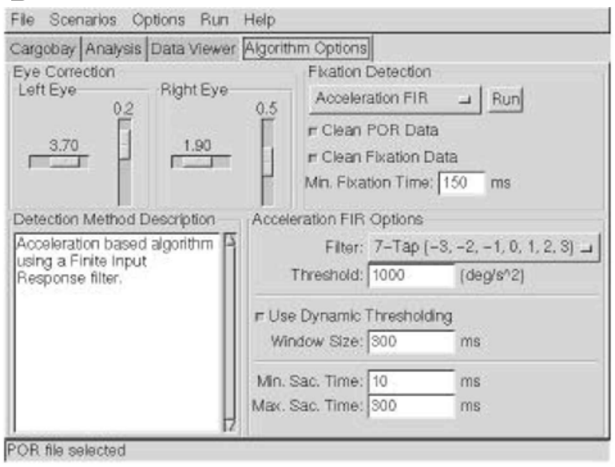
Saccade detection via acceleration filtering requires setting a larger number of parameters. In our current implementation, we have chosen values of 10 and 300 msec for $T_{\min}$ and $T_{\max}$, respectively, to cover a fairly wide range

A



Eye Correction
Left Eye 0.5 1.00
Right Eye 0.5 1.00
Fixation Detection: Velocity FIR
Clean POR Data
Clean Fixation Data
Min. Fixation Time: 150 ms
Detection Method Description: Velocity based algorithm using a Finitie Input Response Filter.
Velocity FIR Options
Filter: 2-Tap (1, 1)
Threshold: 130 (deg/s)
POR file selected

B



Eye Correction
Left Eye 0.2 3.70
Right Eye 0.5 1.90
Fixation Detection: Acceleration FIR
Clean POR Data
Clean Fixation Data
Min. Fixation Time: 150 ms
Detection Method Description: Acceleration based algorithm using a Finite Input Response filter.
Acceleration FIR Options
Filter: 7-Tap (-3, -2, -1, 0, 1, 2, 3)
Threshold: 1000 (deg/s^2)
Use Dynamic Thresholding
Window Size: 300 ms
Min. Sac. Time: 10 ms
Max. Sac. Time: 300 ms
POR file selected

Figure 10. User interface prior to (A, left) and following (B, right) binocular scale factor adjustment.

Table 1
Example Technique

Time	Left Eye	Right Eye	Correction Estimate
t	(−0.5, 0)	(0.3,0)	$dx = x_{rt+1} - x_{rt} = 0.4 - 0.3 = 0.1$
t + 1	(0, 0)	(0.4,0)	$dy = y_{rt+1} - y_{rt} = 0.0 - 0.0 = 0.0$

of saccade acceleration impulse pairs. The choice of the remaining threshold for saccade acceleration was made difficult, since no applicable models of saccadic acceleration (e.g., a main sequence) could readily be found. In fact, unlike commonly listed limits of amplitude, duration, and velocity, there seems to be some disagreement regarding upper limits of acceleration. Peak acceleration has been reported to average at about 30,000 deg/sec^2 in saccades of 10º with a saturation limit of 35,000 deg/sec^2 for $\theta < 15°$, whereas other findings are given of 20º saccades with average peak acceleration of 26,000 deg/sec^2 (Becker, 1989). Since we followed Tole and Young's (1981) acceleration filtering algorithm (incidentally, these authors reported acceleration limits approaching 80,000 deg/sec^2), we decided to start with the authors' recommended thresholds for saccade acceleration. User-adjustable threshold settings for the acceleration filter are shown in Figure 10B (bottom right quadrant).

Tole and Young (1981) pointed out that variable noise characteristics depend on the subject's actions (e.g., different noise profile while gritting teeth). To adapt to such signal changes, the authors recommend an adaptive thresholding technique that dynamically adjusts the threshold, on the basis of the current estimate of noise level. Indeed, we also noted a very large peak-to-peak acceleration signal variance (see below). Following Tole and Young's recommendation, we decided to implement an adaptive thresholding technique in an effort to automatically set acceleration thresholds A and B:

$$A = B = 1{,}000 + \sqrt{\frac{1}{k}\sum_{i=0}^{k}\left(\ddot{\theta}_{i+k}\right)^2}\ \text{deg/sec}^2,$$

where k is the number of samples in time T proportional to the length of the acceleration filter—that is,

$$T = \frac{\text{filter length}}{\text{sampling rate}} = \frac{9}{30\text{Hz}} = 300\,\text{msec}.$$

This is a slightly different implementation of adaptive thresholding than Tole and Young's. Our threshold value is slightly lower, and its adaptive adjustment relies on explicit calculation of the acceleration root-mean squared (RMS). Also, our sampling window for this purpose is also much shorter than the authors' recommended window of $T > 4$ sec. Finally, in our implementation, the adaptive technique currently employs a "look-ahead" scan of the acceleration data, suitable for off-line analysis. Changing the $i + k$ subscript to $i - k$ provides a "look-behind" scan that can be employed in real-time systems.

Fixation Grouping

The above algorithm classifies each GIP as either part of a fixation or a saccade. Once each GIP has been classified, each string of consecutive fixation GIPs is condensed to a single fixation point by finding the centroid of the group. However, owing to the nature of the new algorithm, we observed that, at times, isolated noisy GIPs were also included in fixation groups. To prevent the inclusion of such outlying points, we implemented a simple check to verify that each fixation group's duration is greater than or equal to the minimum theoretical fixation duration (i.e., 150 msec; Irwin, 1992). This parameter is also user adjustable and is shown in Figures 10A and 10B (top right quadrant).

Eye Movement Data Mirroring

Although our new eye movement analysis algorithm is mathematically more robust at handling signal noise, our system is still susceptible to noise generated by the eye tracker. In particular, our eye-tracking equipment randomly drops POR data. In some cases (e.g., during a blink), null POR values are recorded for both left and right eyes. However, in some instances, only one eye's POR is null, whereas the other is not. We believe this occurs because of calibration errors. To address this problem, we developed a heuristic mirroring technique of the nonnull POR eye movement data. Table 1 shows an example of this technique. The left eye POR at time t + 1 is recorded as an invalid null point.

To estimate a nonnull left eye coordinate at t + 1, the difference between successive right-eye POR values is calculated and used to update the left-eye POR values at $t + 1$, as is shown in the equation above, giving $(x_{l_{t+1}}, y_{l_{t+1}}) = (-0.5 + dx, 0 + dy) = (-0.4, 0)$. Note that this solution assumes static vergence eye movements. It is assumed that the eyes remain at a fixed interocular distance during movement. That is, this heuristic strategy clearly will not account for vergence eye movements occurring within the short corrective time period.

ALGORITHM EVALUATION

Evaluation of the eye movement analysis algorithm was conducted by two experiments: a short pilot experiment to evaluate the data-mirroring technique, followed by a comparative evaluation of several saccadic filter combinations in the context of our chosen application testbed (see the following section).

Data Mirroring

A short experiment was conducted to measure the performance of our new heuristic data mirroring technique. A subject was asked to don the HMD, and the eye tracker was carefully calibrated to ensure minimal loss of either of

Table 2
Mirroring Algorithm

	Original Data	No Mirroring	Mirroring
Experiment duration	44.4 sec	44.4 sec	44.4 sec
Usable data	44.0 sec	37.5 sec	44.0 sec
Fixation count	71	62	75
Mean fixation duration	159 msec	196 msec	144 msec

Table 3
Velocity Algorithm Comparisons

Statistics	Two-Tap	Five-Tap
Fixation groups	30	21
Mean fixation duration (msec)	1,079	1,450
Time spent in fixations	73%	69%
Min $\dot{\theta}$ (deg/sec)	0	1
Max $\dot{\theta}$ (deg/sec)	12,385	5,592
M $\dot{\theta}$ (deg/sec)	106	106
SD $\dot{\theta}$ (deg/sec)	635	451

the eyes' POR during the experiment. The 3-D calibration scenario was loaded, and the subject was asked to look at each numbered calibration point. The experiment duration was 44.4 sec. Following immersion, it was noted that less than 0.005% of the generated data contained missing POR information for either eye. The POR file was copied, and monocular POR data were manually decimated at random points in the data stream. Overall, 15% of the data was artificially decimated to simulate noise caused by problematic calibration.

Table 2 compares the results of the mirroring technique over the artificially altered POR data file.

The first column of Table 2 lists eye movement data statistics over the unaltered data. Using the two-tap velocity-based algorithm, the second and third columns compare the effects of the mirroring heuristic. The eye-mirroring technique recovers nearly all of the 15% of the artificially decimated data. Using recovered data, the velocity-based algorithm reported an increase in fixation counts of 17% (75 fixations vs. 62 fixations with no mirroring). This suggests that the recovered data, following the heuristic mirroring technique, fairly closely resemble the original (nearly lossless) signal. In other words, the heuristic mirroring technique allows the estimation of monocular data that would normally be lost owing to eye tracker miscalibration.

Preliminary Filter Comparison

Using the nearly lossless data obtained from the 44.4-sec immersion experiment above, we compared six different filter combinations: both the two-tap and the five-tap velocity filters, and the seven-tap acceleration filter applied to velocity following either two-tap or five-tap velocity filtering, with and without adaptive thresholding. Fixation count (following grouping), mean fixation duration, proportional time spent in fixations, and visual representations of the scanpath were compared in order to evaluate the different filters. All algorithms employed the data-mirroring technique discussed above. The results from velocity filtering are listed in Table 3 and those from acceleration filtering in Table 4.

Figure 11 shows typical plots of the eye movement signal and filter responses. According to accepted saccade amplitude estimates, we expected the measured instantaneous eye movement amplitude (θ) to range up to about 20º. Our observed data ranged up to 136º (*M*, 1.5º; *SD*, 9.7º; median, 0.3º), which appears to be within normal limits, except for a few outliers (possibly owing to head motion or head/eye-tracking instrument noise; see Figure 11A). Our observed velocity averaged at 106 deg/sec (*SD*s, 635 and 451 deg/sec), depending on the filter (see Table 3 and also Figures 11B and C). Our observed acceleration averaged at 4,453 deg/sec^2 (*SD*, 22,475 deg/sec^2) and 3,966 deg/sec (*SD*, 17,470 deg/sec^2), depending on the velocity filter used (see Table 4 and also Figure 11D).

The two-tap velocity filter performed surprisingly well against other filter combinations (outperforming the constant thresholding acceleration filter). However, visual inspection of the resulting scanpath revealed that both the two-tap and the five-tap velocity filters appeared to miss short-duration fixations. The adaptive thresholding acceleration-based technique generated the best overall results for detecting fixations of longest duration. It was also more complicated to use, since it required estimation and control of a larger number of parameters. As compared with the 150- to 650-msec fixation durations reported as common during reading (Irwin, 1992), our fixation durations (17 detected fixations with mean duration of 1.9 sec) were quite long. Although reading eye movements may resemble those made during picture viewing (Bahill et al., 1975), there may be at least three reasons for our findings: (1) Our analysis technique effectively eliminates low-amplitude saccades, (2) the sampling rate of our eye tracking apparatus is too low, or (3) contrary to the above assumption, eye movements in VR may exhibit different characteristics than in reading—it has been noted that eye movements recorded during voluntary head rotation are remarkably free of saccades, implying that the vestibulo- ocular system is involved in combing the generation of saccades (saccadic restraint; McDonald, Bahill, & Friedman, 1983). In reading, there is a distinctive pattern of successive saccades on the words of the text, reflecting the serial processing of the information. In picture viewing, by contrast, there is no canonical scanpath for particular objects (i.e., there is no particular "right way" to look at objects; Kennedy, 1992). Kennedy suggests that the reading task is composed almost exclusively of saccades, whereas picture viewing is composed of shifts, pursuits, and drifts. There may be context differences at play. Continuing the debate about context effects for scenes and sentences, Kroll (1992) states that although

Table 4
Acceleration Algorithm Comparisons

	Two-Tap		Five-Tap	
Statistics	Adaptive	Constant	Adaptive	Constant
Fixation groups	20	17	17	14
Mean fixation duration (msec)	1,633	1,583	1,937	1,983
Time spent in fixations	74%	61%	74%	63%
Min $\ddot{\theta}$ (deg/sec^2)	−257,653		−182,037	
Max $\ddot{\theta}$ (deg/sec^2)	248,265		167,144	
M $\ddot{\theta}$ (deg/sec^2)	4,453		3,966	
SD $\ddot{\theta}$ (deg/sec^2)	22,475		17,470	

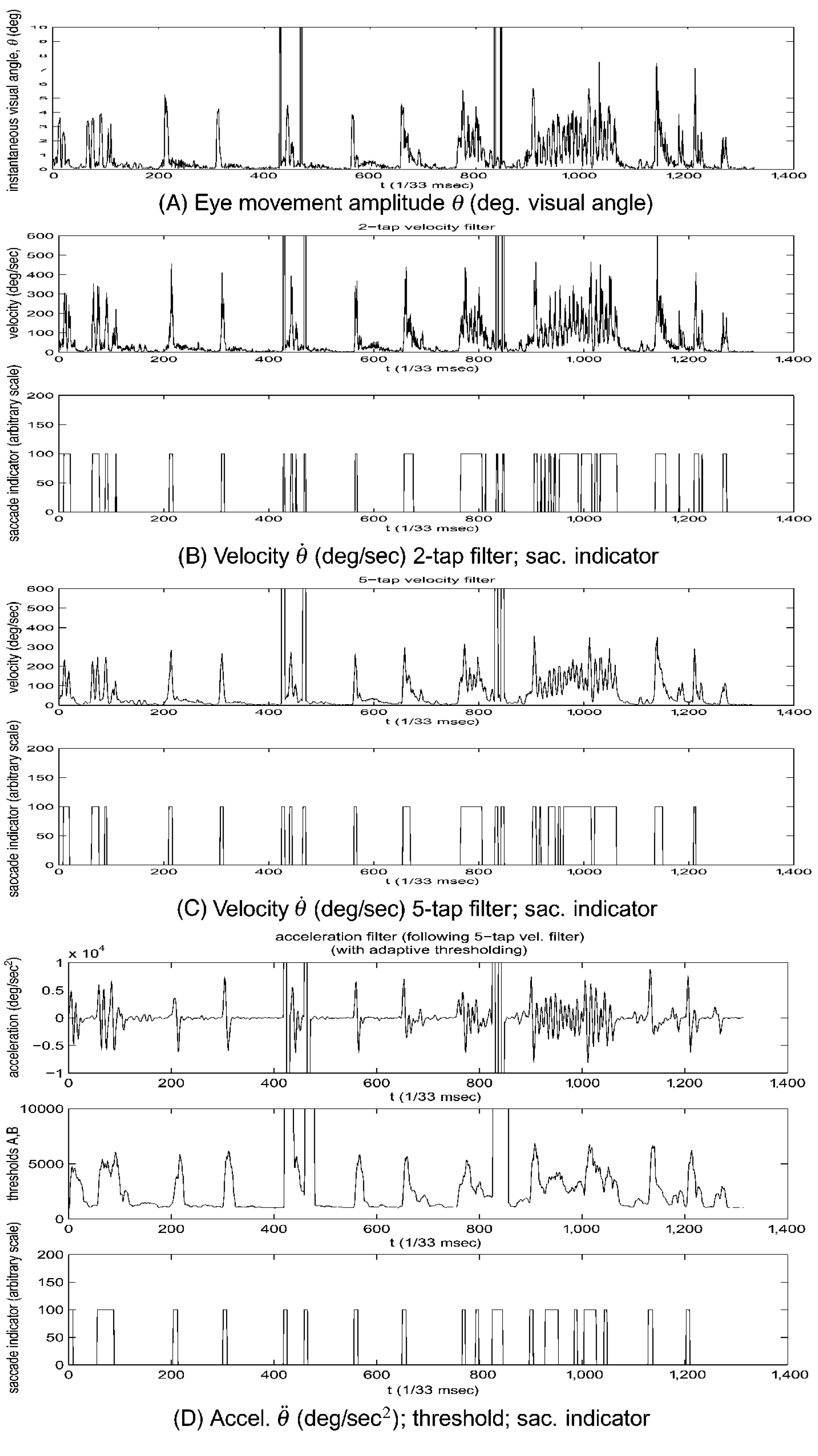


Figure 11. Eye movement signal and filter responses.

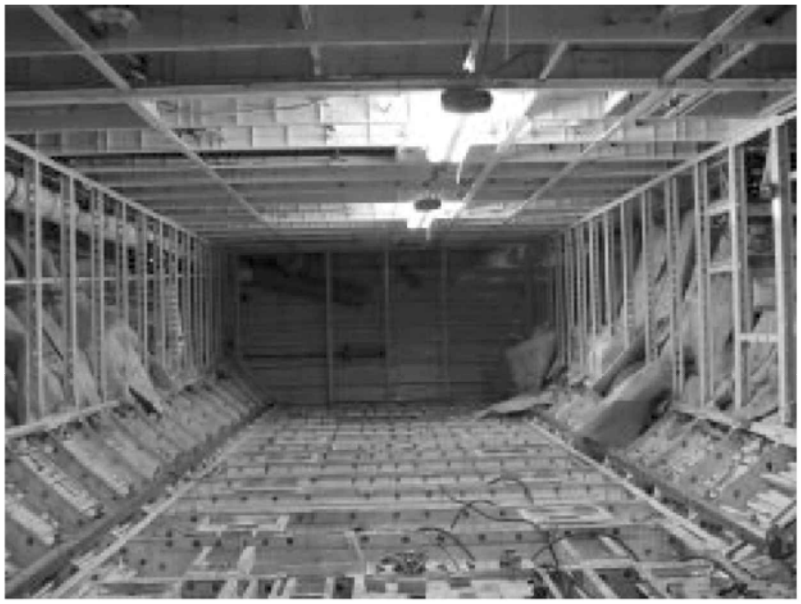


Figure 12. Aircraft cargo bay physical environment.

there may be similarities between the two tasks, the tasks are very different. Eye movements in reading are, to a large extent, driven by the well-known, practiced task. In VR, viewers' eye movement strategies may differ significantly from those adopted for reading.

APPLICATION: A VIRTUAL ENVIRONMENT FOR AIRCRAFT VISUAL INSPECTION TRAINING

Aircraft inspection and maintenance are an essential part of a safe, reliable air transportation system. Training has been identified as the primary intervention strategy in improving inspection performance (Gramopadhye, Bhagwat, Kimbler, & Greenstein, 1998). If training is to be successful, inspectors need to be provided with training tools to help enhance their inspection skills. In response to this need, a diagnostic eye-racking VR system was developed for the purpose of recording process measures (head and eye movements,) as well as performance measures (search time and success rate), during immersion in a VR aircraft inspection simulator (Duchowski et al., 2000). The VR simulator utilizes the binocular eye tracker to record the user's dynamic POR within the VE during visual inspection.

The goal of the construction of the VE is to match the appearance of the physical inspection environment, an air-

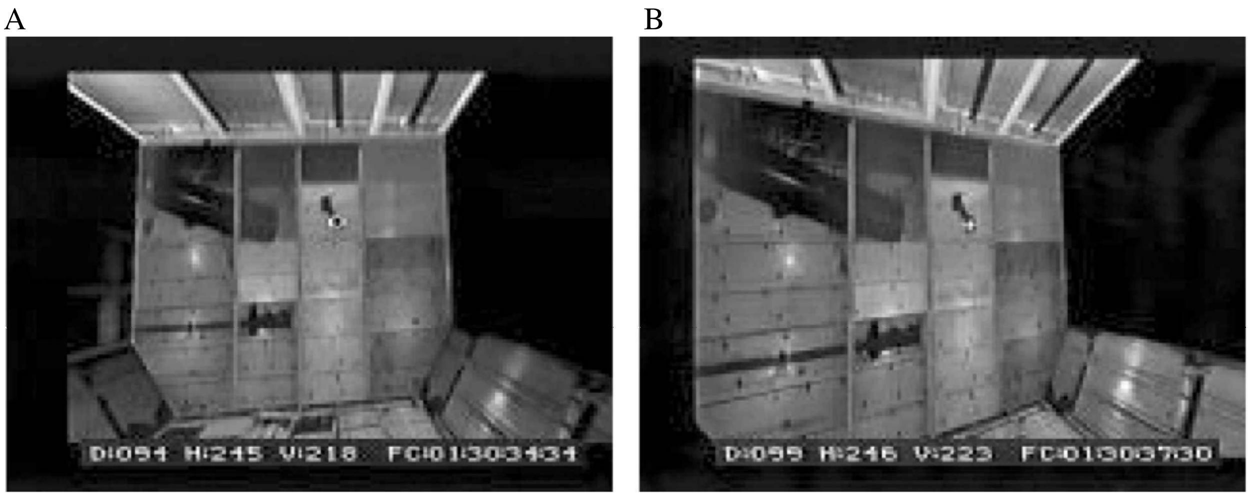
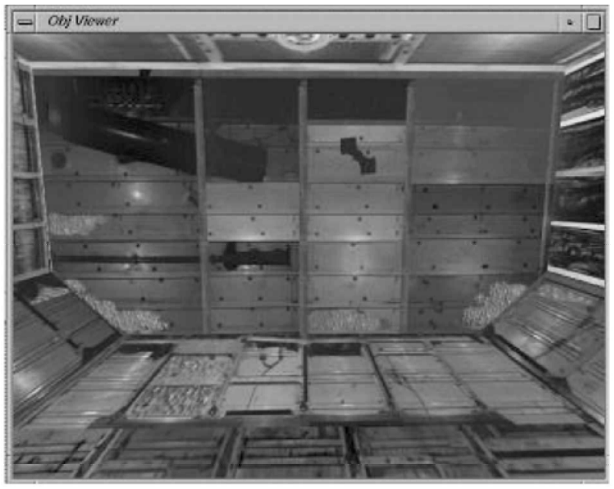


Figure 13. Raw eye tracker output: (A, left) left-eye point of regard (POR); (B, right) right-eye POR.

A



B

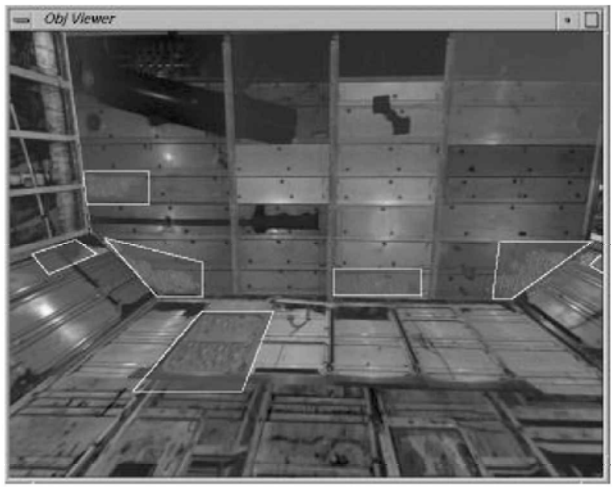


Figure 14. Registering regions of interest in virtual reality: (A, left) simulated corrosion; (B, right) highlighted environmental defects.

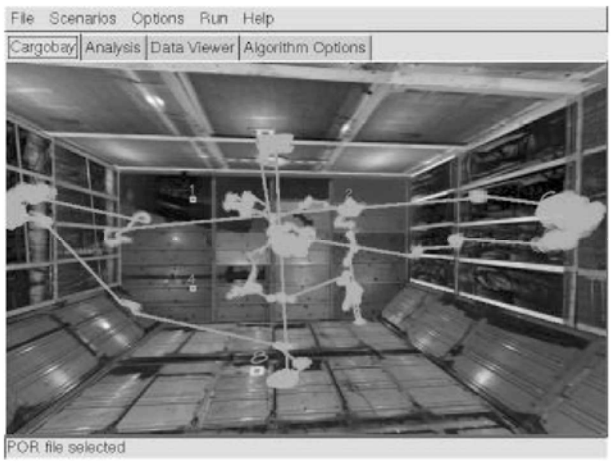
craft cargo bay, shown in Figure 12. The physical environment is a complex 3-D cube-like volume, with airframe components (e.g., fuselage ribs) exposed for inspection. A typical visual inspection task of the cargo bay involves searching for surface defects, such as corrosion and cracks.

The model of the virtual inspection environment was patterned after a simple 3-D enclosure (e.g., a cube), specified by the dimensions of the real inspection environment (i.e., an aircraft's cargo bay). The model was built entirely out of planar polygons. There were two pragmatic reasons for this design choice. First, since the representation of the true complexity of the airframe structure was avoided, fast display rates were possible. Second, planar polygons (quadrilaterals) facilitated texture mapping.

Raw output from the eye tracker is shown in Figure 13, where the left- and the right-eye PORs are represented by a small circle and a small crosshair, respectively, superimposed by the eye tracker's scene-imaging hardware. The VR scene image signal is split (via VGA active passthrough) prior to HMD input and is diverted to the eye tracker. Thus, the eye tracker and the HMD simultaneously display the same image as that seen by the user in the HMD. In addition, each scene image generated by the eye tracker contains the superimposed POR indicator and a status bar, at the bottom, indicating current pupil diameter, horizontal and vertical POR coordinates, and the video frame counter (HH:MM:SS:FF). Note that the images shown in the figure were captured 3 sec apart.

Although our graphical environment is relatively simple, it appears to be sufficiently realistic for the purposes of inspection training. An experiment conducted to evaluate the subjective quality of the simulator attempted to measure the degree of presence felt by subjects immersed in the environment (Vora et al., 2001). Analysis of re-

A


File Scenarios Options Run Help
Cargobay Analysis Data Viewer Algorithm Options
POR file selected

B

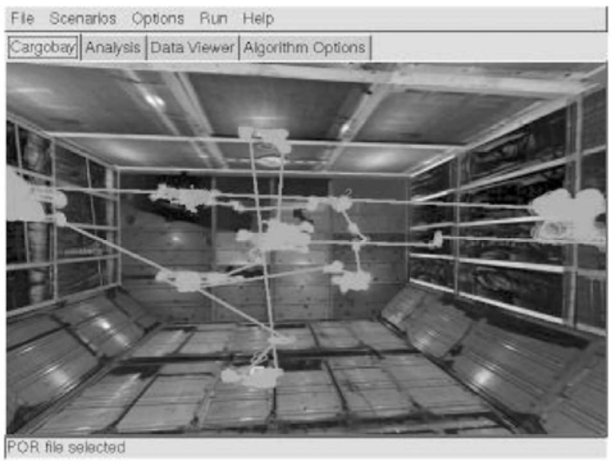

File Scenarios Options Run Help
Cargobay Analysis Data Viewer Algorithm Options
POR file selected

Figure 15. Detected fixations (two-tap velocity filter, ungrouped) prior to (A, left) and following (B, right) binocular scale factor adjustment.

Table 5
Description of Subtasks

No.	Scenario	Task Description
1	No (zero) defect	search entire area with no defects
2	Single defect	find corrosion defects
3	Single defect	find crack defects
4	Single defect	find damaged conduit defects
5	Multiple defect	find all three types of defects
6	No (zero) defect	search entire area with no defects

sponses to a modified version of Witmer and Singer's (1998) Presence Questionnaire revealed that the system scored high on presence-related questions. Visual aspects of the environment, sense of objects, anticipation of system response, surveying, and experience in the environment all contributed to a reported high level of involvement in VR. Although student subjects were not qualified inspectors, on average they indicated their experience in the virtual environment to be consistent with a walk-through of a real aircraft prepared for inspection. We expect that trained inspectors will find the simulator similarly consistent with the real environment, at least in the context of simulating the visual search task. We realize our simulator is not necessarily *photo-realistic* (e.g., owing to the limited resolution of the HMD, and the coarse flat appearance of texture maps); however, since the purpose of the simulator is to train search behavior, we believe the simulator is sufficiently *functionally realistic* for this purpose.

Filter Comparison, Process Measures, and Training Effects

An experiment was conducted to measure the training effects of the VR aircraft inspection simulator. The objectives of the experiment included (1) comparative analysis of different saccadic filter combinations, (2) validation of performance measures used to gauge training effects, and (3) evaluation of the eye movement data as cognitive feedback for training. If we assume that eye movement analysis correctly identifies fixations and the VR simulator is effective for training (i.e., a positive training effect can be measured), the number of detected fixations would be expected to decrease with the adoption of an improved visual search strategy (e.g., following training; Drury, Gramopadhye, & Sharit, 1997).

Method

Stimulus. The airframe inspection simulation featured inspection of an aircraft cargo bay with dimensions similar to those of a real cargo bay of an L1011 aircraft. Texture maps used in the virtual aircraft cargo bay were created from photographs of an actual cargo bay (see above).

For user interaction with the virtual environment and performance measurement during immersion, a 6DOF mouse was used as a multimodal device (see above). The 6DOF mouse allows subjects to perform a pointing and clicking function to indicate selection. The criterion task consisted of inspecting the simulated aircraft cargo bay in search of defects. Several defects can occur in a real-environment situation. Three types of defects were selected to create inspection scenarios: (1) corrosion, represented by a collection of gray and white globules on the inner walls of the aircraft cargo bay and located roughly at knee level, (2) cracks, represented by a cut in any direction on the structural frames inside the aircraft cargo bay, and (3) damaged conduits, shown as either broken or delaminated electrical conduits in the aircraft cargo bay. Figure 14A shows an example of corrosion defects, and the target defects are highlighted in Figure 14B (highlighted defects are shown to the operator but are not typically displayed for the subject).

Performance and process measures. Data for performance and cognitive feedback measures were collected, using search timing and eye movement information, respectively. The following performance measures were collected: (1) search time from region presentation to fault detection, (2) incremental stop time when the subjects terminated the search in a region by deciding that the region did not contain faults, (3) number of faults detected (hits), recorded separately for each fault type, and (4) number of faults that were not identified (misses).

Fixation analysis enabled the collection of cognitive feedback measures, which were provided to the subjects during the training session. Cognitive feedback measures were based on the eye movement parameters that contribute to search strategies, as defined by Megaw and Richardson (1979), including (1) total number of fixations, (2) mean fixation duration, (3) percentage of the area covered, and (4) total trial time. Cognitive feedback measures were graphically displayed offline by rendering a 3-D environment identical to the aircraft cargo bay that was used during immersive trials. This display represented the scanpaths of each trial to indicate the subject's visual search progression.

Subjects. To gauge training effects, 18 graduate students were chosen as subjects, all in the 20- to 25-year-old age group. The subjects were screened for 20/20 corrected vision. The subjects were randomly assigned to three different groups (6 per group): the performance feedback group (PFG), the cognitive feedback group (CFG), and the cognitive + performance feedback group (CPFG). The subjects received different forms of feedback during training sessions before and after trials (see below).

To examine eye movement results from different filter combinations, data were used from 7 subjects between 20 and 30 years of age, selected randomly from a population of graduate and undergraduate students at Clemson University. The subjects were screened for 20/20 corrected vision.

Experimental design. The training study used a 3×2 experimental design with three groups (PFG, CFG, and CPFG) and two trials (before training and after training). Six subjects were placed in each of the three groups. Grouping allowed testing of between-subjects factors, whereas within-subjects factors were tested between trials. Performance and cognitive feedback measures together constituted eight dependent variables, with training scenarios (immersion in different defect inspection scenarios) serving as the independent variable (training treatment).

A 4×2 complete block experimental design was used to compare saccadic filter combinations, with subjects acting as blocking factors. The four algorithm groups represented the following filter combinations: Both two-tap and five-tap velocity filters and the seven-tap acceleration filter applied to velocity following either two-tap or five-tap velocity filtering, with adaptive thresholding.

Calibration procedure. Prior to each experimental trial, the user first had to complete two short calibration trials: (1) a 5-point 2-D calibration sequence to calibrate the eye tracker and (2) the 3-D calibration to enable accurate GIP calculation. The 3-D software calibration procedure relied on a specially marked environment, containing nine clearly visible fixation targets, illustrated in Figure 15.

The nine numerical targets were distributed on five walls of the environment to allow head position to be taken into account during analysis. Without a precise estimate of b and f, computed GIPs might appear stretched or compressed in the horizontal or the vertical di-

Table 6
Mean and Standard Deviation Data for Number of Fixations, Fixation Duration, and Raw Fixation Points

	Number of Fixations				Fixation Durations (msec)				Raw Fixation Points			
	Before		After		Before		After		Before		After	
Algorithm	*M*	*SD*	*M*	*SD*	*M*	*SD*	*M*	*SD*	*M*	*SD*	*M*	*SD*
Two-tap velocity	172.00	51.13	138.81	56.85	805.31	301.88	946.33	317.27	4,212.36	1,069.77	3,253.40	1,661.24
Five-tap velocity	148.19	45.9	117.86	42.66	934.62	392.55	881.86	360.47	4,081.90	1,206.74	3,325.12	1,615.10
Two-tap velocity/seven-tap acceleration	131.74	34.92	100.52	42.39	1,089.67	339.66	1,331.64	898.67	4,152.98	1,167.68	3,592.00	1,621.45
Five-tap velocity/seven-tap acceleration	117.71	34.48	87.36	33.97	1,306.60	468.59	1,578.79	1,021.86	4,482.21	1,159.30	3,657.00	1,575.83

rection, as is shown in Figure 15A (only five targets are visible in the figure).

To shorten the trial duration, eye movement data were stored for off-line analysis. The scalar parameter *s* was obtained manually through the use of a simple interface, shown in Figure 10 (adjustment sliders are in the upper left quadrant of the GUI—note the different scale factors in the two screenshots). As the operator manipulated the scale factor sliders, GIP data were recalculated and displayed interactively. The goal was to align the calculated GIP locations with the environmental targets that the user was instructed to fixate during calibration. An example of this type of adjustment is shown in Figure 15B; note that the GIPs (represented by transparent spheres) are now better aligned over the targets than are the raw data shown in Figure 15A. Once determined, the scale factor *s* was used to adjust each subject's eye movement data in all the subsequent trials.

Training procedure. Each subject was requested to complete a consent form and a demographic questionnaire. Written and oral instructions were provided to ensure the subjects' understanding of the experiment. All the subjects were given information about their required task. Following device and software calibration, the subjects were then shown the entire search area of the virtual aircraft cargo bay and were provided with graphical and verbal descriptions of possible types of defects. The subjects were then presented with a familiarization task similar to the actual trials in the VR simulator and were shown how to use the 6DOF mouse for pointing at and selecting targets.

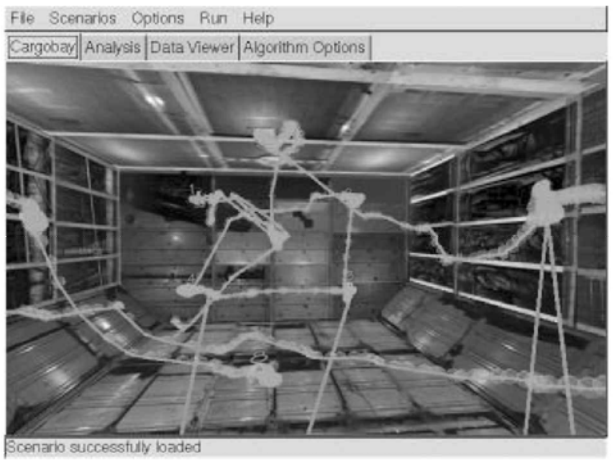
The before-training criterion task was an unpaced visual inspection search task. The subjects searched for defects on the walls, the floor, and the ceiling of the simulated 3-D cargo bay. The entire search task was divided into a series of six subtasks, listed in Table 5.

To cancel out order effects, all 6 subjects in each group completed their assigned subtasks, following a counterbalanced order using a 6×6 Latin square design. Treatments were randomly assigned to each of the 6 subjects.

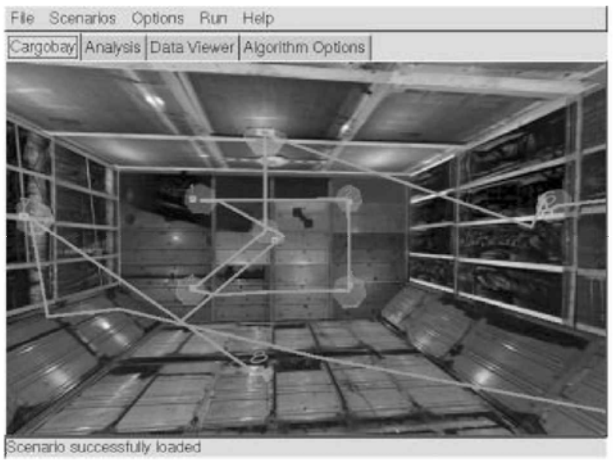
On completion of the before-training trials, all the subjects underwent respective training sessions for each of the three groups. The first step in the training sessions was completion of a multidefect search task. The subjects received feedback training according to the respective feedback training groups. (1) The subjects in the PFG received performance measures feedback performance (search times, errors). (2) The subjects in the CFG received two forms of cognitive feedback: statistical and graphical. Statistical feedback included the number of fixations, mean fixation duration, number of fixations in ROIs, mean fixation duration in the ROIs, and percentage of area covered. For graphical feedback, the subjects viewed a graphical visualization of their scanpaths representing their search patterns, with fixation indices showing their visual search progression. (3) The subjects in the GPFG received both forms of feedback, performance feedback training as well as cognitive feedback training. On completion of the training sessions, all the subjects performed an after-training criterion task. This subtask was counterbalanced to eliminate order effects.

Results

Process measures and training effects. An analysis of variance (ANOVA) showed no significant differences between subjects (feedback groups). However, an ANOVA showed significant differences in mean search time, per-

A


Scenario successfully loaded

B


Scenario successfully loaded

Figure 16. Raw data (A, left); two-tap velocity-based analysis (B, right).

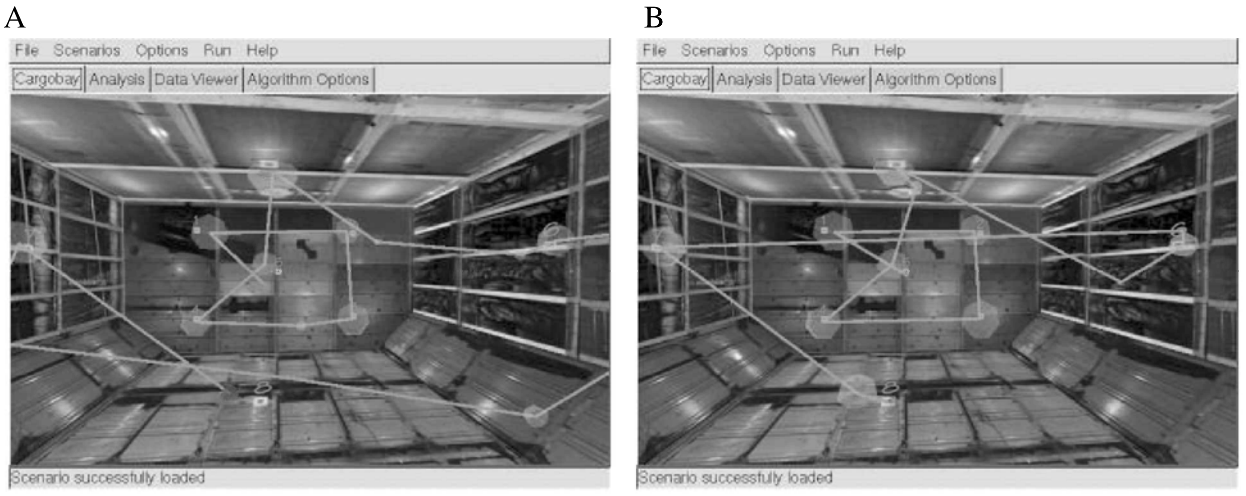

A
B
File Scenarios Options Run Help
Cargobay Analysis Data Viewer Algorithm Options
Scenario successfully loaded

Figure 17. Acceleration-based (five-tap) analysis with adaptive thresholding (A, left) and without (B, right).

centage of defects detected, incremental stopping time, and total trial time within subjects.

Filter comparison: Number of fixations. A two-factor ANOVA for number of fixations revealed no significant trial × filter interaction. The trial factor was found to be statistically significant [$F(7,49) = 38.84, p < .001$], indicating that there was a difference in the mean number of fixations between before- and after-training trials. Similarly, the algorithm factor was found to be significant [$F(7,49) = 20.64$, $p < .001$], indicating that there was a difference in the number of fixations identified by each filter combination.

Further post hoc analysis revealed that there was a significant reduction in the number of fixations between before- and after-training trials and that this was evident for all four algorithms. A significant difference was found in the computation of number of fixations between the two-tap velocity filter and the other filter combinations. It was found that the two-tap velocity filter generated the highest number of fixations whereas the seven-tap acceleration filter generated the lowest number of fixations. The five-tap velocity filter showed a significantly different number of fixations from both the acceleration filters. There was no significant difference between the mean numbers of fixations shown by either of the acceleration filters.

Filter comparison: Fixation durations. A two-factor ANOVA for fixation durations revealed no significant trial × filter interaction. The trial factor was not found to be significant, indicating no significant change in the mean fixation durations between the before- and the after-training trials. The computations for duration by different filters were found to be statistically different from each other [$F(7,49) = 7.91, p < .001$].

Further post hoc analysis showed no statistical difference between the two velocity filters or between the two acceleration filters in the computation of fixation durations. There was a statistical difference in the computation of fixation durations between the velocity filters and the acceleration filters. The shortest durations were found with the two-tap velocity filter, and the longest durations were detected with the seven-tap acceleration filter.

A two-factor ANOVA of raw fixation points revealed no significant trial × filter interactions and no significant filter main effects. The trial factor was found to be significant [$F(7,49) = 8.61, p < .001$]. Further post hoc analysis revealed that there was no significant difference between the mean raw fixation points as labeled by all four filters for any of the trials (before or after). The overall mean data for number of fixations, fixation durations, and raw fixation points are provided in Table 6.

Filter comparison: 3-D visualization. Figures 16B (right) and 17 show typical "raindrop" visualizations of the resulting analysis following fixation grouping. The radius of each fixation sphere is proportional to fixation duration.

Figure 16B (right) shows the resulting scanpath following two-tap velocity-based analysis (the scanpath resulting from five-tap velocity filtering is not shown but is similar).

Figure 17A (left) shows the resulting scanpath following acceleration-based analysis with adaptive thresholding, Figure 17B (right) shows acceleration-based analysis without adaptive thresholding. Both acceleration-based methods better represent long fixations, owing to localization of fewer saccades.

Discussion

Analysis indicates that, overall, training in the VR aircraft simulation has a positive effect on subsequent search performance in VR, although there is apparently no difference in the type of feedback given to subjects. Cognitive feedback, in the form of visualized scanpaths, does not appear to be any more effective than performance feedback. It may be that the most effective common contributor to training is the immersion in the VR environment—that is, the exposure to the given task, or at least to the simulated task.

Whether the eye tracker, by providing cognitive feedback, contributes to the improvement of inspection performance is inconclusive. Users may benefit just as much from performance feedback alone. However, the eye tracker is a valuable tool for collecting process measures. An analysis of results leads to two observations. First, mean fixation times do not appear to change significantly following training. This is not surprising, since eye movements are, to a large extent, driven by physiology (i.e., muscular and neurological functions) and cognitive skill. In this case, the search task itself may not have altered cognitive load per se; rather, prior experience in the simulator may have facilitated a more efficient search in subsequent trials. Second, the number of fixations decreases following training. These results generally appear to agree with the expectation of a reduced number of fixations with the adoption of an improved visual search strategy (e.g., owing to learning or familiarization of the task). The implication of a reduced number of fixations (without an increase in mean fixation time) suggests that, in the posttraining case, subjects tend to employ a greater number of saccadic eye movements. That is, an improved visual search strategy may be one in which subjects inspect the environment more quickly (perhaps owing to familiarity gained through training), reducing the time required to visually rest on particular features.

CONCLUSION

In this paper, we have presented new developments for eye movement analysis in 3-D, specifically dealing with improved noise suppression. The paper described (1) the use of velocity and acceleration filters for eye movement analysis in three-space, (2) the utility of adaptive thresholding and fixation grouping, and (3) a heuristic method to recover lost eye movement data owing to miscalibration. The results indicate that heuristic data mirroring is an effective strategy for recovering lost short-duration eye movement data. Fixation grouping appears to be an effective means for the elimination of spurious fixation outliers following analysis. Provided proper thresholds are selected, both velocity-based and acceleration-based filtering approaches appear to generate acceptable results. Although velocity-based analysis is easier to deal with, it is more sensitive to noise (i.e., resulting in classification of a greater number of saccades). Under different circumstances (e.g., with 12-bit sampled data), velocity filters in general (and the two-tap filter in particular) may perform more accurately (Bahill & McDonald, 1983). In contrast, owing to the greater degree of freedom in parameter estimation, the acceleration-based technique can be adjusted to be less sensitive to smaller amplitude saccades, resulting in a more robust approach to fixation detection.

From our experiments conducted in our chosen eye-tracked VR application, we note that performance measures quantify the level of improvement of subjects' inspection performance (i.e., *how* the subject performed). If improvement can be shown, we may conclude that training contributes to performance improvement and, additionally, that the VR simulator is a suitable environment for training. In addition, process measures not only can corroborate performance gains, but also can lead to discoveries of reasons for performance improvements (i.e., *what* the subject performed). In particular, tracking the users' eyes can potentially lead to further insights into the underlying cognitive processes of human inspectors.

REFERENCES

Anliker, J. (1976). Eye movements: On-line measurement, analysis, and control. In R. A. Monty & J. W. Senders (Eds.), *Eye movements and psychological processes* (pp. 185-202). Hillsdale, NJ: Erlbaum.

Bahill, A. T., Clark, M., & Stark, L. (1975). The main sequence: A tool for studying human eye movements. *Mathematical Biosciences*, **24**, 191-204.

Bahill, A. T., & McDonald, J. D. (1983). Frequency limitations and optimal step-size for the two-point central difference derivative algorithm with applications to human eye movement data. *IEEE Transactions on Biomedical Engineering*, **BME-30**, 191-194.

Becker, W. (1989). Metrics. In R. H. Wurtz & M. E. Goldberg (Eds.), *The neurobiology of saccadic eye movements* (pp. 13-68). New York: Elsevier.

Bowman, D. A., & Hodges, L. F. (1997). An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. In *Symposium on Interactive 3D Graphics* (pp. 35-38). New York: ACM Press.

Clark, M. R., & Stark, L. (1975). Time optimal behavior of human saccadic eye movement. *IEEE Transactions on Automatic Control*, **20**, 345-348.

Drury, C. G., Gramopadhye, A. K., & Sharit, J. (1997). Feedback strategies for visual inspection in airframe structural inspection. *International Journal of Industrial Ergonomics*, **19**, 333-344.

Duchowski, A. T. (1998). Incorporating the viewer's point of regard (POR) in gaze-contingent virtual environments. In M. T. Bolas, S. S. Fisher, & J. O. Merritt (Eds.), *Stereoscopic Displays and Virtual Reality Systems V* (Proceedings of SPIE, Vol. 3295, pp. 332-343). Bellingham, WA: SPIE Press.

Duchowski, A. T., Medlin, E., Gramopadhye, A., Melloy, B., & Nair, S. (2001). Binocular eye tracking in VR for visual inspection training. In *Virtual Reality Software & Technology (VRST)* (pp. 1-8). New York: ACM Press.

Duchowski, A. T., Shivashankaraiah, V., Rawls, T., Gramopadhye, A., Melloy, B., & Kanki, B. (2000). Binocular eye tracking in virtual reality for inspection training. In A. Duchowski, K. Karn, & J. Senders (Eds.), *Eye Tracking Research and Applications (ETRA) Symposium* (pp. 89-96). New York: ACM Press.

Glassner, A. S. (Ed.). (1989). *An introduction to ray tracing*. San Diego: Academic Press.

Gramopadhye, A., Bhagwat, S., Kimbler, D., & Greenstein, J. (1998). The use of advanced technology for visual inspection training. *Applied Ergonomics*, **29**, 361-375.

Hain, T. C. (1999). *Saccade (calibration) tests* (On-line manual). Retrieved October 2001 at http://www.tchain.com/otoneurology/practice/saccade.htm.

Horn, B. K. P. (1986). *Robot vision*. Cambridge, MA: MIT Press.

Irwin, D. E. (1992). Visual memory within and across fixations. In K. Rayner (Ed.), *Eye movements and visual cognition: Scene perception and reading* (pp. 146-165). New York: Springer-Verlag.

Kennedy, A. (1992). The spatial coding hypothesis. In K. Rayner (Ed.), *Eye movements and visual cognition: Scene perception and reading* (pp. 379-396). New York: Springer-Verlag.

Knox, P. C. (2001). *The parameters of eye movement* (Lecture notes). Retrieved October 2001 at http://www.liv.ac.uk/~pcknox/teaching/Eymovs/params.htm.

Kroll, J. F. (1992). Making a scene: The debate about context effects for scenes and sentences. In K. Rayner (Ed.), *Eye movements and visual cognition: Scene perception and reading* (pp. 284-292). New York: Springer-Verlag.

McDonald, J. D., Bahill, A. T., & Friedman, M. B. (1983). An adaptive control model for human head and eye movements. *IEEE Transactions on Systems, Man, & Cybernetics*, **SMC-13**, 167-174.

Megaw, E. D., & Richardson, J. (1979). Eye movements and industrial inspection. *Applied Ergonomics*, **10**, 145-154.

Salvucci, D. D., & Goldberg, J. H. (2000). Identifying fixations and saccades in eye-tracking protocols. In A. Duchowski, K. Karn, & J. Senders (Eds.), *Eye Tracking Research and Applications (ETRA) Symposium* (pp. 71-78). New York: ACM Press.

Tanriverdi, V., & Jacob, R J. K. (2000). Interacting with eye movements in virtual environments. In *Human factors in computing systems: CHI 2000 conference proceedings* (pp. 265-272). New York: ACM Press.

Tole, J. R., & Young, L. R. (1981). Digital filters for saccade and fixation detection. In D. F. Fisher, R. A. Monty, & J. W. Senders (Eds.), *Eye movements: Cognition and visual perception* (pp. 7-17). Hillsdale, NJ: Erlbaum.

Vora, J., Nair, S., Medlin, E., Gramopadhye, A., Duchowski, A. T., & Melloy, B. (2001). Using virtual technology to improve aircraft inspection performance: Presence and performance measurement studies. In *Proceedings of the Human Factors and Ergonomics Society 45th Annual Meeting* (pp. 1867-1871). Santa Monica, CA: Human Factors and Ergonomics Society.

Watson, B., Walker, N., & Hodges, L. F. (1997). Managing level of detail through head-tracked peripheral degradation: A model and resulting design principles. In *Virtual reality software and technology: Proceedings of the VRST'97* (pp. 59-63). New York: ACM Press.

Witmer, B. G., & Singer, M. J. (1998). Measuring presence in virtual environments: A presence questionnaire. *Presence*, **7**, 225-240.

NOTES

1. Silicon Graphics, Onyx2, and InfiniteReality are registered trademarks of Silicon Graphics, Inc.

2. If $t < 0$, the polygon may intersect the gaze ray, but behind the viewer.

(Manuscript received March 14, 2002;
revision accepted for publication July 18, 2002.)