3D Eye Movement Analysis

Andrew Duchowski, Eric Medlin, Nathan Cournia, and Hunter Murphy
Department of Computer Science, Clemson University

Anand Gramopadhye, Santosh Nair, Jeenal Vorah, and Brian Melloy
Department of Industrial Engineering, Clemson University

This paper presents a novel 3D eye movement analysis algorithm for binocular eye tracking within Virtual Reality. The user's gaze direction, head position and orientation are tracked to allow recording of the user's fixations within the environment. While the linear signal analysis approach is itself not new, its application to eye movement analysis in three dimensions advances traditional 2D approaches since it takes into account the 6 degrees of freedom of head movements and is resolution independent. Results indicate that the 3D eye movement analysis algorithm can successfully be used for analysis of visual process measures in Virtual Reality. Process measures can not only corroborate performance measures, but can also lead to discoveries of 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 participants.

Background

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 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 2-5° visual angle for 300 ms, then 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 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 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 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 on the measurement of visual angle, since the farther eye movements are made away from the central axis, 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 di-

This work was supported in part by a University Innovation grant (# 1-20-1906-51-4087), NASA Ames task (# NCC 2-1114), and NSF CAREER award IIS-9984278.

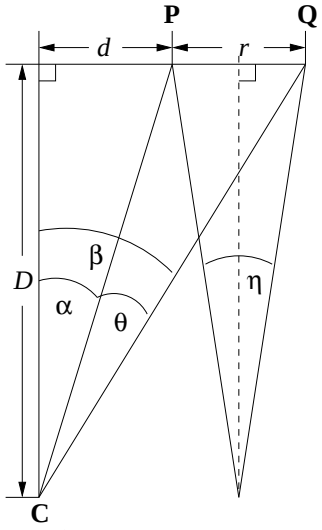


Figure 1. 2D geometry.

rectly 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 extents of the POR and head center, respectively, e.g., $\mathbf{P} = (d,0,D)$ and $\mathbf{Q} = (d+r,0,D)$ if **C** defines the origin and gaze is recorded along the horizontal viewing axis. The vector-based approach forms the basis of our 3D 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 visual angle per second.

We present a velocity-based eye movement analysis algorithm in three dimensions, applicable to the three-dimensional eye movement data recorded during immersion in a Virtual Environment (VE) (Duchowski, Medlin, Gramopadhye, Melloy, & Nair, 2001). Traditional 2D 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 choose a different approach by mapping raw POR data to world coordinates first, followed by eye movement analysis in three-space. We favor this approach since the calculated gaze points in three-space provide a composite three-dimensional representation of both left and right eye movements. Applying the traditional 2D 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 describe our operational platform and derive applicable gaze vector calculations including a 2D-to-3D 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, are then discussed. The novel 3D eye movement analysis algorithm is then presented followed by an evaluation of the algorithm featuring a comparative analysis of several velocity and acceleration filters for saccade detection. Finally, we describe our application testbed: a Virtual Environment used for aircraft visual inspection training and 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 8 raster managers and 8 MIPS® R12000™ processors, each with 8MB secondary cache.[1] It is equipped with 8Gb of main memory and 0.5Gb of texture memory.

Multi-modal hardware components include a binocular eye tracker mounted within a Virtual Research V8 Head Mounted Display. 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 helmet. 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 infra-red LEDs mounted within the helmet to measure eye movements. Figure 2 shows the dual cameras and infra-red LEDs of the binocular assembly. Mounted below the HMD

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

Figure 2. Binocular eye tracker optics (with HMD inset above).

lenses, the eye imaging cameras peer upwards 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 degrees over a ± 20 degree 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 three-dimensional 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, 2D eye tracker data expressed in eye tracker screen coordinates must be mapped to the 2D dimensions of the near viewing frustum. The 3D viewing frustum employed in the perspective viewing transformation is defined by the parameters `left, right, bottom, top, near, 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:

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

$$y = \texttt{bottom} + \frac{(512 - y')(\texttt{top} - \texttt{bottom})}{512} \quad (2)$$

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: (a) the size of the application window, and (b) 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 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. Based on the measurements shown in Figure 4, the linear coordinate mapping is:

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

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

While 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., pin-cushion effect), the linear mapping is sufficient for coordinate registration (Duchowski, 1998).

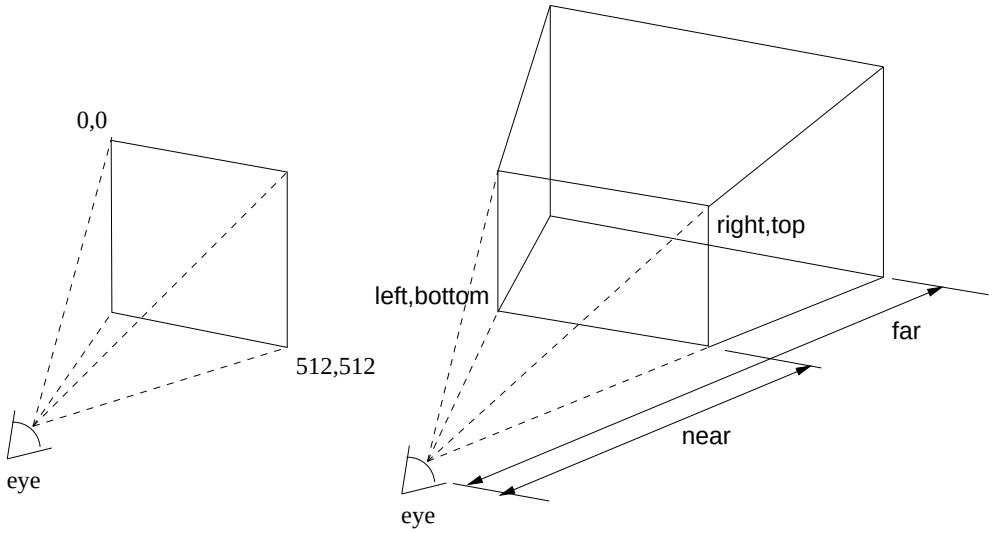


Figure 3. Eye tracker to 3D viewing frustum screen coordinate mapping.

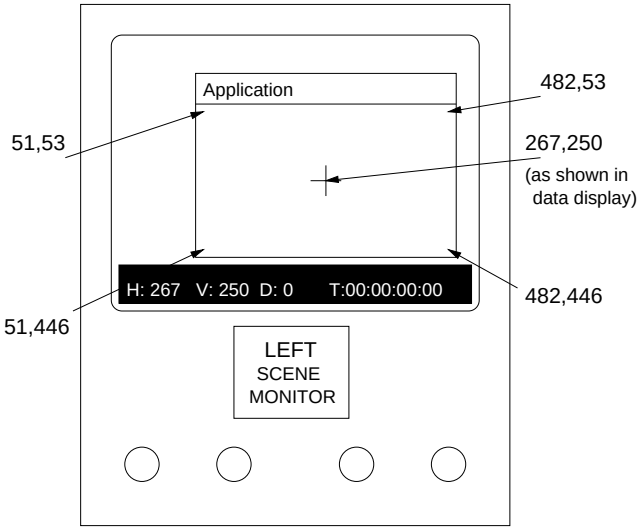


Figure 4. Mapping measurement example.

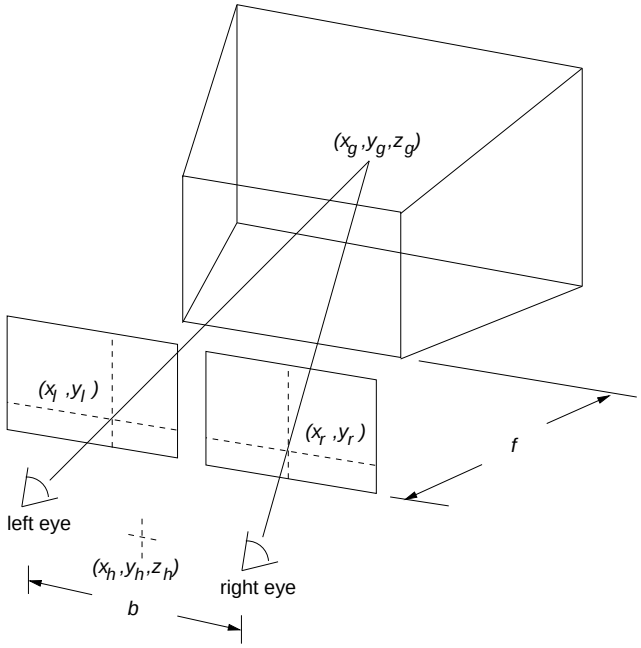


Figure 5. Basic binocular geometry.

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 three-dimensional 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. 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 relations:

$$x_g = (1-s)x_h + s(x_l + x_r)/2 \quad (5)$$

$$y_g = (1-s)y_h + s(y_l + y_r)/2 \quad (6)$$

$$z_g = (1-s)z_h + sf \quad (7)$$

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),

$$\begin{aligned} x_g &= x_h + s\left(\frac{x_l + x_r}{2} - x_h\right) \\ y_g &= y_h + s\left(\frac{y_l + y_r}{2} - y_h\right) \\ z_g &= z_h + s(f - z_h) \end{aligned}$$

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, i.e.,

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

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 three-dimensional 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 three-dimensional 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 as Gaze Intersection Points (GIPs) for brevity, are each found on the closest polygon to the viewer intersecting the gaze ray, assuming all 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 3D mouse). Our gaze-based selection mechanism is similar, however, our derivation of the gaze ray is slightly different due 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 polygons are opaque. This polygon is found by testing all 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 only returns 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 inter-pupillary distance, b) and focal distance f used in Equations (5)–(7) is difficult. Inter-pupillary 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 inter-pupillary distance outside VR, e.g., at the start of the viewing session, is

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

of course possible, however, conversion of such a measurement to VR coordinates is problematic (i.e., virtual coordinates are often unitless but 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 thought to be related to the unknown inter-pupillary distance, a 3D calibration procedure was designed to estimate the inter-pupillary distance scaling factor *s* empirically. The calibration procedure is currently specific to our application testbed (see below).

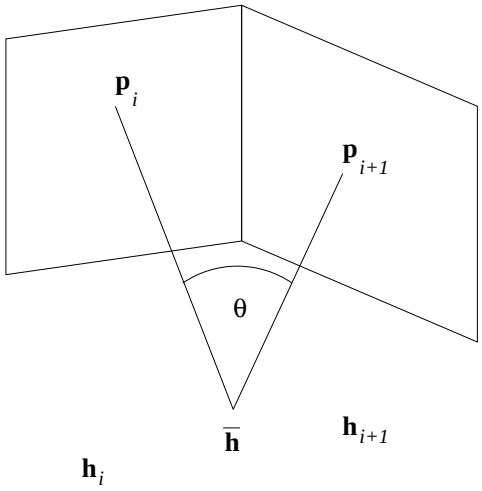


Figure 6. Eye movement analysis in 3D.

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 2D 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, $\mathbf{p}_i = (x_i, y_i, z_i)$ and $\mathbf{p}_{i+1} = (x_{i+1}, y_{i+1}, z_{i+1})$, and the head position at each instance, $\mathbf{h}_i$ and $\mathbf{h}_{i+1}$, 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:

$$\theta_i = \cos^{-1} \frac{\mathbf{v}_i \cdot \mathbf{v}_{i+1}}{\| \mathbf{v}_i \| \, \| \mathbf{v}_{i+1} \|}, \quad i \in [0, n), \tag{10}$$

where n is the sample size and $\mathbf{v}_i = \mathbf{p}_i - \overline{\mathbf{h}}$ and $\overline{\mathbf{h}}$ 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 the head (Watson, Walker, & Hodges, 1997).

With visual angle, θ_i, and timestamp difference between $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$, the same velocity-based thresholding is used as in the traditional 2D case. No conversion 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 2-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 using a convolution filter with coefficients $\{1, 1\}$, i.e., a 2-tap Finite Impulse Response (FIR) filter.

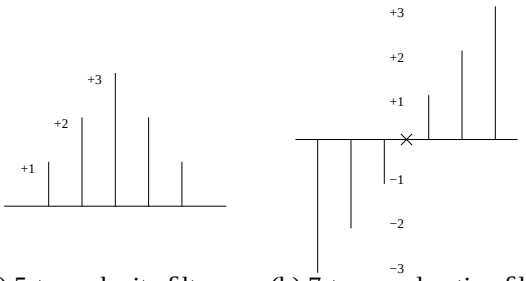
A preliminary study was conducted to evaluate the 3D eye movement analysis algorithm. Results indicated that due to the somewhat noisy signal analysis approach, the algorithm underestimated the identified number of fixations and fixation durations (Duchowski et al., 2001). This result was not wholly 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 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 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 177s 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 due to its short length, it is known to be quite noisy. For more robust off-line fixation analysis a longer filter should be used. In the following sections, we 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 2-tap

FIR filter with a 5-tap FIR filter, shown in Figure 7(a). Due to



(a) 5-tap velocity filter. (b) 7-tap acceleration filter.

Figure 7. FIR filters.

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 7(b), and is convolved with eye movement velocity data as obtained via either the 2-tap or 5-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 8(b). 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 $\mathbf{h}_k$, are essentially unnormalized low-pass filters which tend to smooth and amplify the underlying signal. Division by the duration of the sampling window yields velocity, i.e.,

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

expressed in deg/s, where k is the filter length, $\Delta t = k - i$. We compare the performance of the 5-tap filter to the previously implemented 2-tap filter with coefficients $\{1,1\}$ below.

Acceleration is obtained via a subsequent convolution of velocity, $\dot{\theta}_i$, with the acceleration filter, $\mathbf{g}_j$, shown in Figure 7(b). That is,

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

where k is the filter length, $\Delta t = k - i$. The acceleration filter is essentially an unnormalized high-pass differential filter. The resulting value, $\ddot{\theta}_i$, expressed in deg/s^2, is checked against threshold A. If the absolute value of $\ddot{\theta}_i$ is greater than A, then the corresponding gaze intersection point $\mathbf{p}_i$ 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 to detect the end of the saccade. Two additional conditions are evaluated to locate a saccade, as given by Tole and Young. 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. This is because Tole and Young applied their filters (the reverse of ours, essentially) to the positional eye movement signal ($\mathbf{p}$), while our filters are applied to the signal amplitude (θ). Pseudocode of the technique is presented in Algorithm 1.

(a) Pos.

(b) Vel.

(c) Acc.

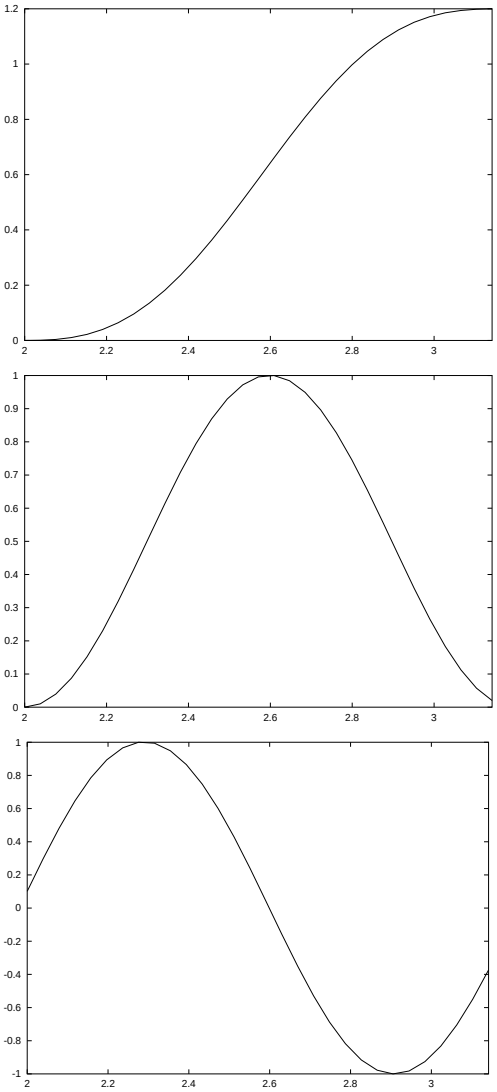


Figure 8. Characteristic saccade signal and filter responses.

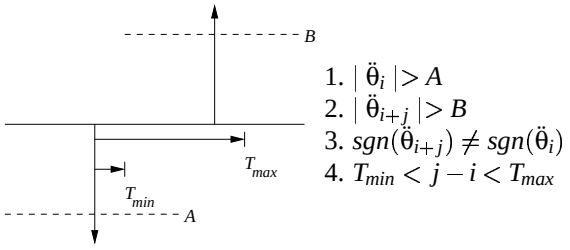


Figure 9. Acceleration thresholding.

Algorithm 1 Acceleration-based saccade detection.

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

```
// calculate instantaneous visual angle
for i = 0 to n − 1 do
    θ_i = cos^{-1}(v_i · v_{i+1} / ‖ v_i ‖‖ v_{i+1} ‖)
end for
// initialize accumulation arrays (convolution results)
for i = 0 to n − k − 1 do
    θ̇_i = θ̈_i = 0
end for
// convolve with vel. filter
for i = 0 to n − k − 1 do
    for j = 0 to k do
        θ̇_i = θ̇_i + θ_{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
        θ̈_i = θ̈_i + θ̇_{i+j} ∗ g_j
    end for
end for
for i = 0 to n − k − 1 do
    // condition 1
    if | θ̈_i | ≥ A then
        // condition 4 (implicit in loop)
        for j = i+T_min to (n − k) − i ∧ (j − i) ≤ T_max do
            // conditions 2 & 3
            if | θ̈_{i+j} | ≥ B ∧ sgn(θ̈_{i+j}) ≠ sgn(θ̈_i) then
                for l = i to j do
                    p_l = saccade
                end for
            else
                p_i = fixation
            end if
        end for
    end if
end for
```

Parameter Estimation

Thresholds are needed for saccade velocity, acceleration, and duration, since our fixation detection algorithm relies on the detection of saccades. While eventually determined empirically, algorithm fine tuning was guided by a review of the literature, briefly summarized here for context. While scanpath characteristics may be task-dependent, i.e., differing when looking at pictures than when reading, for the purpose of initial estimation of parameters, we assumed that, when 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 reading eye movement characteristics 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 degrees 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 degrees, but can range up to about 50°, reaching velocity saturation at about 1000 deg/s (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/s (Hain, 1999). For saccade detection via velocity filtering, we chose a threshold of 130 deg/s for both 2-tap and 5-tap filters. Using the asymptotic model of the main sequence relationship between saccade amplitude and velocity (limited by 750 deg/s), 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 10(a) (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 ms and 300 ms for T_{min} and T_{max}, respectively, to cover a fairly wide range 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/s^2 in saccades of 10° with a saturation limit of 35,000 deg/s^2 for $\theta < 15°$, while other findings are given of 20° saccades with average peak acceleration of 26,000 deg/s^2

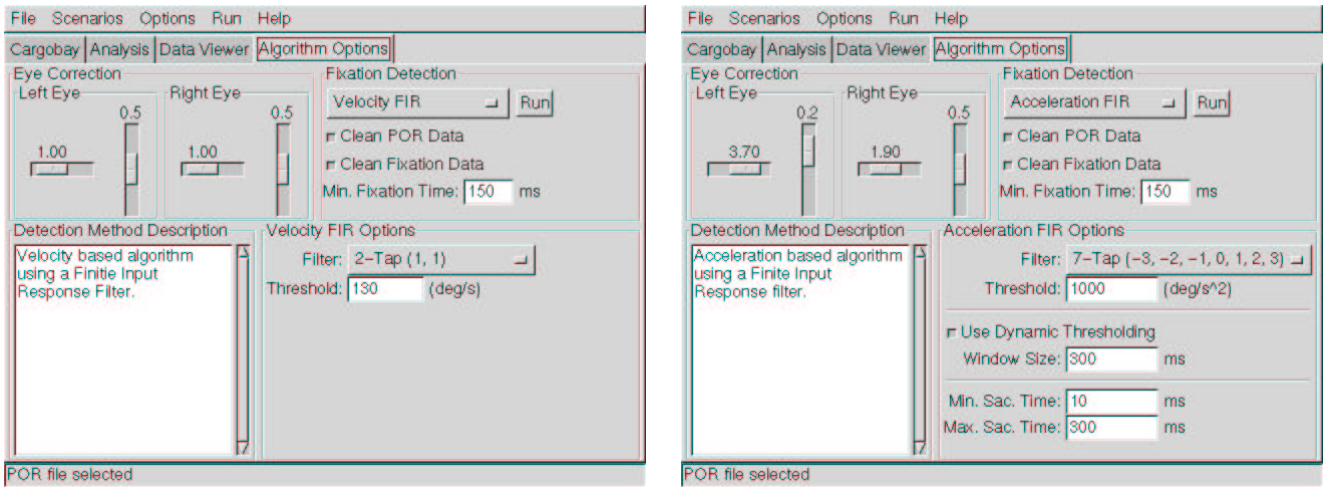


Figure 10. User interface prior to (a, left), and following (b, right) binocular scale factor adjustment.

(Becker, 1989). Since we followed Tole and Young's acceleration filtering algorithm (incidentally, these authors report acceleration limits approaching 80,000 deg/s^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 10(b) (bottom-right quadrant).

Tole and Young (1981) point 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 which dynamically adjusts the threshold, based on 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:

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

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

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

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 from 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 which can be employed in real-time systems.

Fixation Grouping

The above algorithm classifies each GIP as either part of a fixation or 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, due 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 ms (Irwin, 1992)). This parameter is also user-adjustable, and is shown in Figures 10(a) and (b) (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 while the other is not. We believe this occurs due to calibration errors. To address this problem we developed a heuristic mirroring technique of the non-null POR eye movement data. The table below shows an example of this technique. The left eye POR at time $t+1$ is recorded as an invalid null point.

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

To estimate a non-null 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 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 will clearly 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 studies: a short pilot study 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 following section).

Data Mirroring

A short study 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 the eyes' POR during the experiment. The 3D calibration scenario was loaded and the subject was asked to look at each numbered calibration point. The experiment duration was 44.4s. 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 is 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 1 compares the results of the mirroring technique over the artificially altered POR data file. The first column of Table 1 lists eye movement data statistics over the unaltered data. Using the 2-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 resembles the original (nearly lossless) signal. In other words, the heuristic mirroring technique allows the estimation of monocular data that would normally be lost due to eye tracker miscalibration.

Table 1
Mirroring algorithm.

	Original Data	No Mirroring	Mirroring
Experiment Duration	44.4 s	44.4 s	44.4 s
Usable Data	44.0 s	37.5 s	44.0 s
Fixation Count	71	62	75
Mean Fixation Duration	159 ms	196 ms	144 ms

Preliminary Filter Comparison

Using the nearly lossless data obtained from the 44.4s immersion experiment above, we compared 6 different filter combinations: both 2-tap and 5-tap velocity filters, and the 7-tap acceleration filter applied to velocity following either 2-tap or 5-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 to evaluate the different filters. All algorithms employed the data mirroring technique discussed above. Results from velocity filtering are listed in Table 2 and those from acceleration filtering in Table 3. 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 20°. Our observed data ranges up to 136°(M: 1.5°, SD: 9.7°, median: 0.3°), which appear to be within normal limits, except for a few outliers (possibly due to head motion or head/eye tracking instrument noise) (see Figure 11(a)). Our observed velocity averages at 106 deg/s (SD: 635 and 451 deg/s), depending on the filter (see Table 2 and also Figures 11(b) and 11(c)). Our observed acceleration averages at 4,453 deg/s^2 (SD: 22,475 deg/s^2) and 3,966 deg/s^2 (SD: 17,470 deg/s^2), depending on the velocity filter used (see Table 3 and also Figure 11(d)).

The 2-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 2-tap and 5-tap velocity filters appear to miss short-duration fixations. The adaptive thresholding acceleration-based technique generates the best overall results detecting fixations of longest duration. It is also more complicated to use since it requires estimation and control of a larger number of parameters. Compared to 150-650 ms fixation durations reported as common during reading (Irwin, 1992), our fixation durations (17 detected fixations with mean duration of 1.9 sec) are quite long. Although reading eye movements may resemble those 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 the vestibulo-ocular system is involved in combing the gen-

Table 2
Velocity algorithm comparisons.

Statistics	2-tap	5-tap
fixation groups	30	21
mean fixation duration (ms)	1079	1450
time spent in fixations	73%	69%
min $\dot{\theta}$ (deg/s)	0	1
max $\dot{\theta}$ (deg/s)	12,385	5,592
M $\dot{\theta}$ (deg/s)	106	106
SD $\dot{\theta}$ (deg/s)	635	451

Table 3
Acceleration algorithm comparisons.

	2-tap		5-tap	
Statistics	adaptive	constant	adaptive	constant
fixation groups	20	17	17	14
mean fixation duration (ms)	1633	1583	1937	1983
time spent in fixations	74%	61%	74%	63%
min $\ddot{\theta}$ (deg/s^2)		-257,653		-182,037
max $\ddot{\theta}$ (deg/s^2)		248,265		167,144
M $\ddot{\theta}$ (deg/s^2)		4,453		3,966
SD $\ddot{\theta}$ (deg/s^2)		22,475		17,470

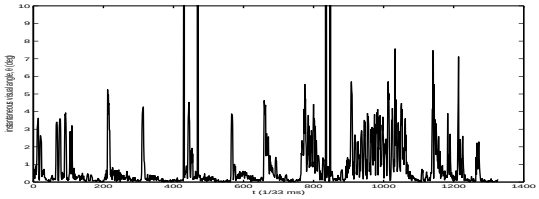
eration of saccades (saccadic restraint) (McDonald, Bahill, & Friedman, 1983). In reading, there is a distinctive pattern of successive saccades on the words of the text, reflecting the serial processing of the information. In picture viewing, by contrast, there is no canonical scanpath for particular 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, while 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 while 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

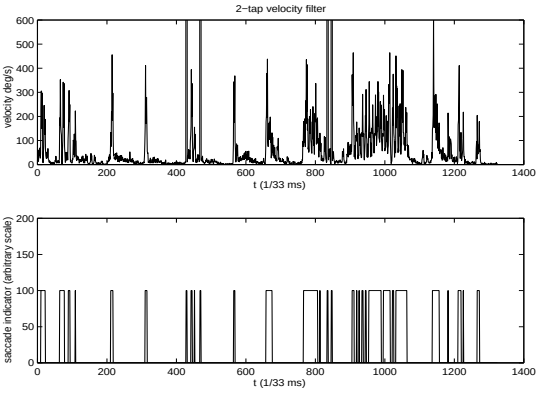
Aircraft inspection and maintenance are an essential part of a safe, reliable air transportation system. Training has been identified as the primary intervention strategy in improving inspection performance (Gramopadhye, Bhagwat, Kimbler, & Greenstein, 1998). If training is to be successful, inspectors need to be provided with training tools to help enhance their inspection skills. In response to this need, a diagnostic eye tracking Virtual Reality (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 Point Of Regard (POR) within the virtual environment during visual inspection.

The goal of the construction of the virtual environment is to match the appearance of the physical inspection environment, an aircraft cargo bay, shown in Figure 12. The physical environment is a complex three-dimensional 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 three-dimensional enclosure (e.g., a cube), specified by the dimensions of the real inspection environment (i.e., an aircraft's cargo bay). The model is built entirely out of planar polygons. There are two pragmatic reasons for this design choice. First, since the representation of the true complexity of the airframe structure is avoided, fast display rates are possible. Second, planar polygons (quadrilaterals) facilitate texture mapping.

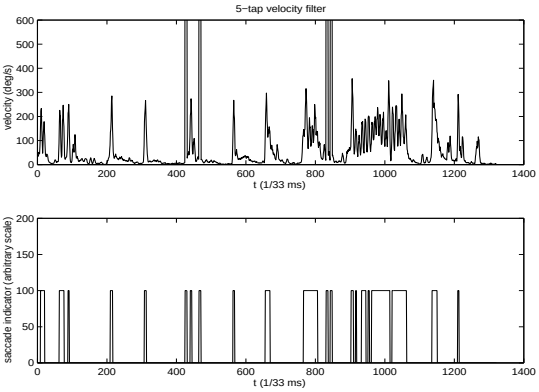
Raw output from the eye tracker is shown in Figure 13, where the left and right eye POR is represented by a small circle and small crosshair, respectively, superimposed by the eye tracker's scene imaging hardware. The VR scene image signal is split (via VGA active passthrough) prior to HMD input, and diverted to the eye tracker. Thus the eye tracker and HMD simultaneously display the same image seen by the



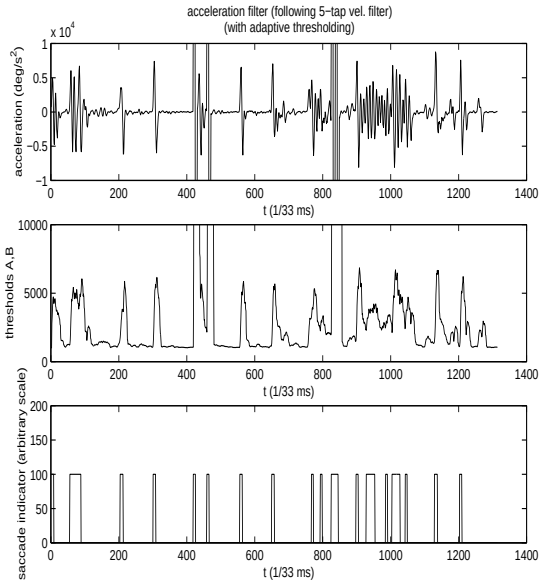
(a) Eye movement amplitude θ (deg. visual angle).



(b) Velocity $\dot{\theta}$ (deg/s) 2-tap filter; sac. indicator.



(c) Velocity $\dot{\theta}$ (deg/s) 5-tap filter; sac. indicator.



(d) Accel. $\ddot{\theta}$ (deg/s^2); threshold; sac. indicator.

Figure 11. Eye movement signal and filter responses.

Figure 12. Aircraft cargo bay physical environment.

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 seconds apart.

While 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 participants immersed in the environment (Vora et al., 2001). Analysis of responses 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 walkthrough of a real aircraft prepared for inspection. We expect 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., due to limited resolution of the HMD, coarse and 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 & 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 dif-

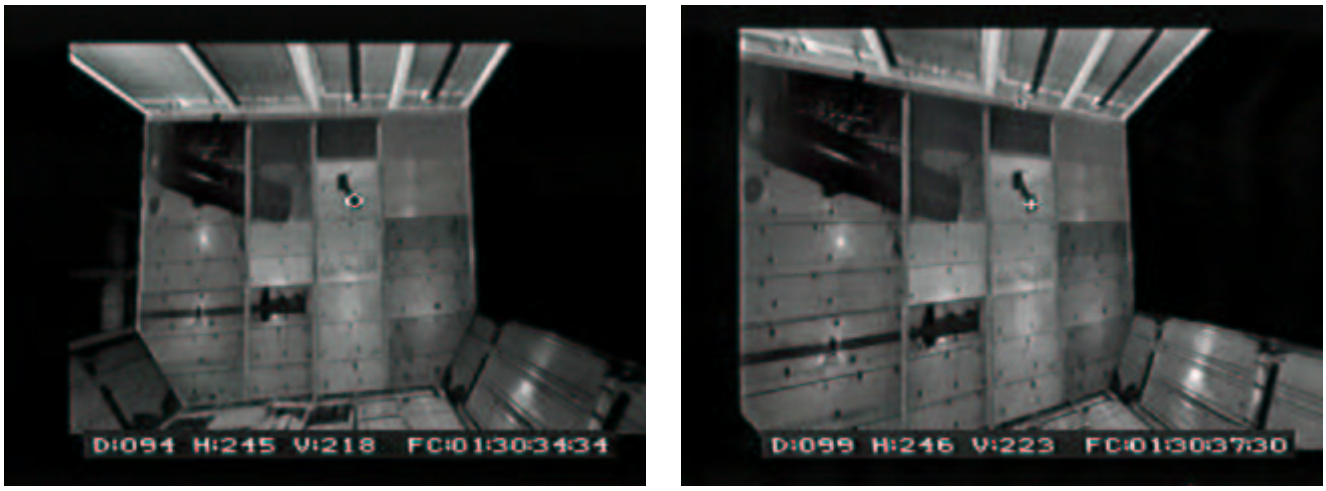


Figure 13. Raw eye tracker output: (a, left) left eye POR, (b, right) right eye POR.

ferent 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. Assuming 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 are expected to decrease with the adoption of an improved visual search strategy (Drury, Gramopadhye, & Sharit, 1997) (e.g., following training).

Stimulus. The airframe inspection simulation featured inspection of an aircraft cargo bay with dimension similar to that 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 multi-modal 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.

3. Damaged conduits: shown as either broken or delaminated electrical conduits in the aircraft cargo bay.

Figure 14(a) shows an example of corrosion defects, with target defects highlighted in Figure 14(b) (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 was 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 subjects terminated the search in a region by deciding the region does not contain faults.

3. Number of faults detected (hits), recorded separately for each fault type.

4. Number of faults that were not identified (misses).

Fixation analysis enabled the collection of cognitive feedback measures, which were provided to 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 area covered; and (4) total trial time. Cognitive feedback measures were graphically displayed offline by rendering a 3D environment identical to the aircraft cargo bay which 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 eighteen graduate students were chosen as subjects, all in the 20-25 year old age group. Subjects were screened for 20/20 corrected vision. Subjects were randomly assigned to three different groups (6 per group): Performance Feedback Group (PFG), Cognitive Feedback Group (CFG), and Cognitive + Performance Feedback Group (CPFG). 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 was used from seven subjects aged between 20 and 30 years of age, selected randomly from a pop-

Figure 14. Registering ROIs in VR: (a, left) simulated corrosion; (b, right) highlighted environmental defects.

ulation of graduate and undergraduate students at Clemson University. Subjects were screened for 20/20 corrected vision.

Experimental Design. The training study used a 3 × 2 experimental design with 3 groups (PFG, CFG, and CPFG) and 2 trials (before-training and after-training). Six subjects were placed in each of the three groups. Grouping allowed testing of between-subject factors, while within-subject factors were tested between trials. Performance and cognitive feedback measures together constitute 8 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 4 algorithm groups represented the following filter combinations: both 2-tap and 5-tap velocity filters, and the 7-tap acceleration filter applied to velocity following either 2-tap or 5-tap velocity filtering, with adaptive thresholding.

Calibration Procedure. Prior to each experimental trial the user must first complete two short calibration trials: (1) a 5-point 2D calibration sequence to calibrate the eye tracker, and (2) the 3D calibration to enable accurate GIP calculation. The 3D software calibration procedure relies on a specially marked environment, containing 9 clearly visible fixation targets, illustrated in Figure 15. The 9 numerical targets are distributed on 5 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 may appear stretched or compressed in the horizontal or vertical direction, as shown in Figure 15(a) (only 5 targets are visible in the figure).

To shorten the trial duration eye movement data is stored for off-line analysis. The scalar parameter s is 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 manipulates the scale factor sliders, GIP data is re-calculated and displayed interactively. The goal is to align the calculated GIP locations with the environmental targets which the user was instructed to fixate during calibration. An example of this type of adjustment is shown in Figure 15(b). Notice that the GIPs (represented by transparent spheres) are now better aligned over the targets than the raw data in Figure 15(a). Once determined, the scale factor s is used to adjust each participant's eye movement data in all subsequent trials.

Training Procedure. Each subject was requested to complete a consent form and demographic questionnaire. Written and oral instructions were provided to ensure subjects' understanding of the experiment. All subjects were given information about their required task. Following device and software calibration, 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. Subjects were then presented with a familiarization task similar to the actual trials in the Virtual Reality 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. Subjects searched for defects on the walls, floor, and the ceiling of the simulated 3D cargo bay. The entire search task was divided into a series of six subtasks listed in Table 4. To cancel out order effects, all six participants 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 six participants.

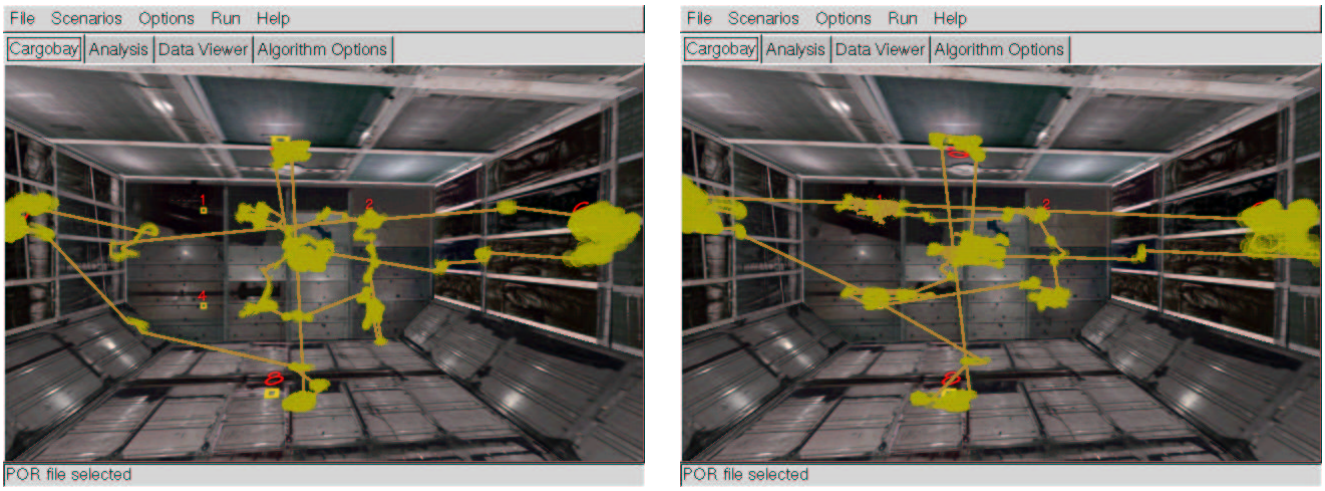


Figure 15. Detected fixations (2-tap vel. filter, ungrouped) prior to (a, left), and following (b, right) binocular scale factor adjustment.

Table 4
Description of subtasks.

#	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 defects
6	No (zero) defect	Search entire area with no defects

On completion of the before-training trials, all subjects underwent respective training sessions for each of the three groups. The first step in the training sessions was completion of a multi-defect search task. Subjects received feedback training according to the respective feedback training groups:

- Performance Feedback Group. Subjects in this group received performance measures feedback performance (search times, errors).
- Cognitive Feedback Group. Subjects in this group 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 area covered. For graphical feedback, subjects viewed a graphical visualization of their scanpaths representing their search patterns with fixation indices showing their visual search progression.
- Cognitive + Performance Feedback Group. Subjects in this group received both forms of feedback, performance feedback training as well as cognitive feedback training.

On completion of the training sessions, all subjects performed an after training criterion task. This subtask was counterbalanced to eliminate order effects.

Results

Process Measures & Training Effects. Analysis of variance (ANOVA) showed no significant differences between subjects (feedback groups). However, ANOVA showed significant differences in mean search time, percentage 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 $\times$ filter interaction effect. The trial factor was found to be statistically significant ($F(7,49) = 38.84$, $p < 0.001$) indicating 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 ($F(7,49) = 20.64$, $p < 0.001$) indicating there was difference in the number of fixations identified by each filter combination.

Further post hoc analysis revealed there was a significant reduction in the number of fixations between before- and after-training trials and this was evident for all four algorithms. A significant difference was found in the computation of number of fixations between the 2-tap velocity filter and the other filter combinations. It was found that the 2-tap velocity filter generated the highest number of fixations whereas the 7-tap acceleration filter generated the lowest number of fixations. The 5-tap velocity filter found a significantly different number of fixations from both the acceleration filters. There was no significant difference between the mean numbers of fixations found by either of the acceleration filters.

Filter Comparison: Fixation Durations. A 2-factor ANOVA for fixation durations revealed no significant trial $\times$ filter interaction effect. The trial factor was not found to be significant indicating no significant change in the mean fixation durations between the before- and after-training trials.

The computation for duration by different filters was found to be statistically different from each other ($F(7,49) = 7.91$, $p < 0.001$).

Further post hoc analysis found no statistical difference between the two velocity filters or between the two acceleration filters in computation of fixation durations. There was a statistical difference in the computation of fixation durations between the velocity filters and the acceleration filters. The 2-tap velocity filter found the shortest durations and the 7-tap acceleration filter detected the longest durations.

A 2-factor ANOVA of raw fixation points revealed no significant trial $\times$ filter interaction effects and no significant filter main effects. The trial factor was found to be significant ($F(7,49) = 8.61$, $p < 0.001$). 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 is provided in Table 5.

Filter Comparison: 3D Visualization. Figures 16(b, 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 16(b,right) shows the resulting scanpath following 2-tap velocity-based analysis (the scanpath resulting from 5-tap velocity filtering is not shown but is similar). Figure 17 (a, left) shows the resulting scanpath following acceleration-based analysis with adaptive thresholding, Figure 17 (b, right) shows acceleration-based analysis without adaptive thresholding. Both acceleration-based methods better represent long fixations due 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 common most effective 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. 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 decrease following training. These results generally appear to agree with the expectation of reduced number of fixations with the adoption of an improved visual search strategy (e.g., due to learning or familiarization of the task). The implication of reduced number of fixations (without an increase in mean fixation time) suggests that, in the post-training case, subjects tend to employ a greater number of saccadic eye movements. That is, an improved visual search strategy may be one where subjects inspect the environment more quickly (perhaps due to familiarity gained through training), reducing the time required to visually rest on particular features.

Conclusion

The paper presented new developments for eye movement analysis in 3D, 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 due to miscalibration. 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 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. While 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 2-tap filter in particular) may perform more accurately (Bahill & McDonald, 1983). In contrast, due 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 Virtual Reality 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, then 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 can not only corroborate performance gains, but can also 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.

Table 5
Mean and SD data for number of fixations, fixation duration and raw fixation points.

	Number of Fixations		Fixation Durations (ms)		Raw Fixation Points	
Algorithm	Before	After	Before	After	Before	After
2-tap vel.	172.00	138.81	805.31	946.33	4212.36	3253.40
	(51.13)	(56.85)	(301.88)	(317.27)	(1069.77)	(1661.24)
5-tap vel.	148.19	117.86	934.62	881.86	4081.90	3325.12
	(45.9)	(42.66)	(392.55)	(360.47)	(1206.74)	(1615.10)
2-tap vel./7-tap acc.	131.74	100.52	1089.67	1331.64	4152.98	3592.00
	(34.92)	(42.39)	(339.66)	(898.67)	(1167.68)	(1621.45)
5-tap vel./7-tap acc.	117.71	87.36	1306.60	1578.79	4482.21	3657.00
	(34.48)	(33.97)	(468.59)	(1021.86)	(1159.30)	(1575.83)

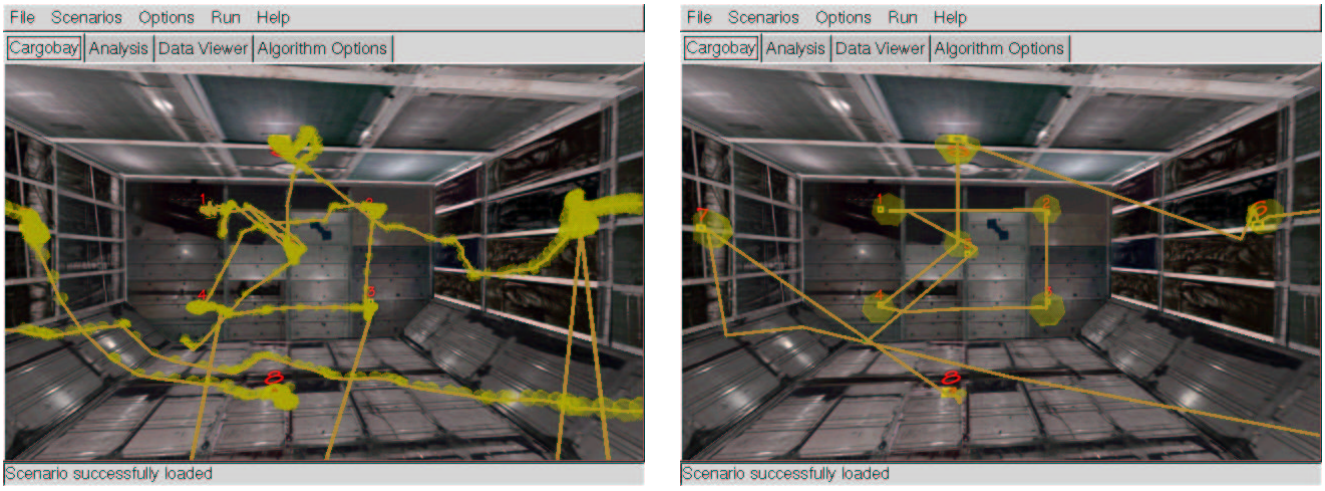


Figure 16. Raw data (a, left), 2-tap velocity-based analysis (b, right).

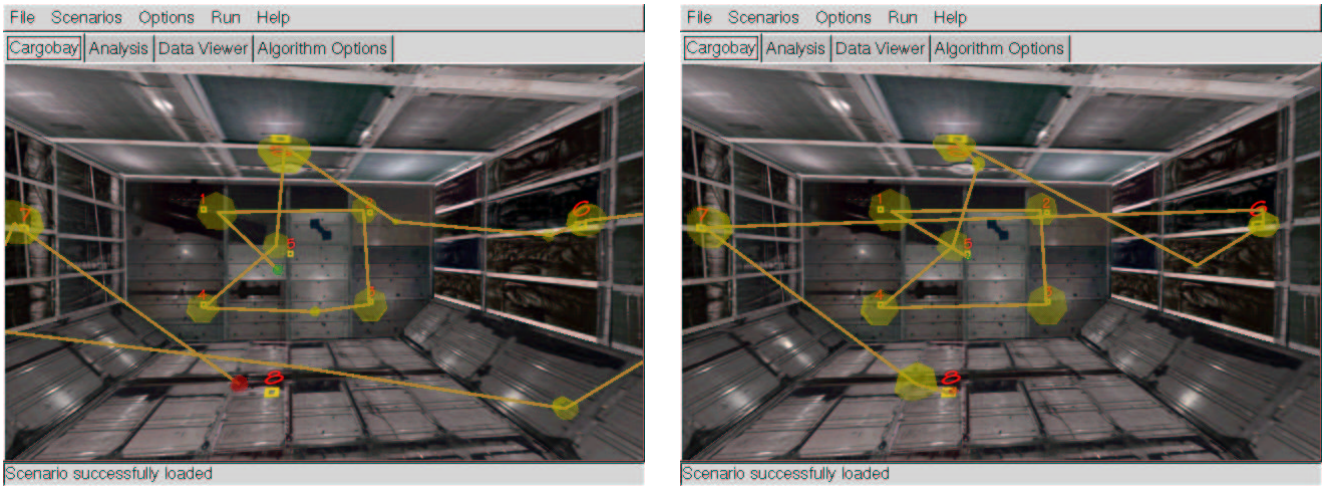


Figure 17. Acceleration-based (5-tap) analysis, with (a, left) adaptive thresholding, and (b, right), without.

Acknowledgments

We would like to thank Dario Salvucci and a second anonymous referee for their helpful comments on the manuscript.

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* (p. 185-202). Hillsdale, NJ: Lawrence Erlbaum Associates.

Bahill, A. T., Clark, M., & Stark, L. (1975). The Main Sequence, A Tool for Studying Human Eye Movements. *Mathematical Biosciences, 24*(3/4), 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* (p. 13-68). New York, NY: Elsevier Science Publishers BV (Biomedical Division).

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.* New York, NY.

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., Medlin, E., Gramopadhye, A., Melloy, B., & Nair, S. (2001). Binocular Eye Tracking in VR for Visual Inspection Training. In *Virtual Reality Software & Technology (VRST).* Banff, AB, Canada.

Duchowski, A., Shivashankaraiah, V., Rawls, T., Gramopadhye, A., Melloy, B., & Kanki, B. (2000). Binocular Eye Tracking in Virtual Reality for Inspection Training. In *Eye Tracking Research & Applications Symposium* (p. 89-96). Palm Beach Gardens, FL.

Duchowski, A. T. (1998). Incorporating the Viewer's Point-Of-Regard (POR) in Gaze-Contingent Virtual Environments. In *Stereoscopic Displays and Virtual Reality Systems V.* Bellingham, WA.

Glassner, A. S. (Ed.). (1989). *An Introduction to Ray Tracing.* San Diego, CA: Academic Press.

Gramopadhye, A., Bhagwat, S., Kimbler, D., & Greenstein, J. (1998). The Use of Advanced Technology for Visual Inspection Training. *Applied Ergonomics, 29*(5), 361-375.

Hain, T. C. (1999). *Saccade (Calibration) Tests.* (Online Manual, URL: <http://www.tchain.com/otoneurology/practice/saccade.htm> (last accessed October 2001))

Horn, B. K. P. (1986). *Robot Vision.* Cambridge, MA: The 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* (p. 146-165). New York, NY: Springer-Verlag. (Springer Series in Neuropsychology)

Kennedy, A. (1992). The Spatial Coding Hypothesis. In K. Rayner (Ed.), *Eye Movements and Visual Cognition: Scene Perception and Reading* (p. 379-396). New York, NY: Springer-Verlag. (Springer Series in Neuropsychology)

Knox, P. C. (2001). *The Parameters of Eye Movement.* (Lecture Notes, URL: <http://www.liv.ac.uk/ pcknox/teaching/Eymovs/params.htm> (last accessed October 2001))

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.* Springer-Verlag. (Springer Series in Neuropsychology)

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, and Cybernetics, SMC-13*(3), 167-174.

Megaw, E. D., & Richardson, J. (1979). Eye Movements and Industrial Inspection. *Applied Ergonomics, 10*(3), 145-154.

Salvucci, D. D., & Goldberg, J. H. (2000). Identifying Fixations and Saccades in Eye-Tracking Protocols. In *Eye Tracking Research & Applications (ETRA) Symposium* (p. 71-78). Palm Beach Gardens, FL.

Tanriverdi, V., & Jacob, R. J. K. (2000). Interacting with Eye Movements in Virtual Environments. In *Human Factors in Computing Systems: CHI 2000 Conference Proceedings* (p. 265-272). 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* (p. 7-17). Hillsdale, NJ: Lawrence Erlbaum Associates.

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.* Minneapolis, MN.

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 & Technology: Proceedings of the VRST'97* (p. 59-63). ACM.

Witmer, B. G., & Singer, M. J. (1998). Measuring Presence in Virtual Environments: A Presence Questionnaire. *Presence, 7*(3), 225-240.