Fixation-identification in dynamic scenes: Comparing an automated algorithm to manual coding

Susan M. Munn†*  Leanne Stefano‡  Jeff B. Pelz†

† Chester F. Carlson Center for Imaging Science
‡ Department of Psychology
Multidisciplinary Vision Research Laboratory
Rochester Institute of Technology

Abstract

Video-based eye trackers produce an output video showing where a subject is looking, the subject’s point-of-regard (POR), for each frame of a video of the scene. Fixation-identification algorithms simplify the long list of POR data into a more manageable set of data, especially for further analysis, by grouping PORs into fixations. Most current fixation-identification algorithms assume that the POR data are defined in static two-dimensional scene images and only use these raw POR data to identify fixations. The applicability of these algorithms to gaze data in dynamic scene videos is largely unexplored. We implemented a simple velocity-based, duration-sensitive fixation-identification algorithm and compared its performance to results obtained by three experienced users manually coding the eye tracking data displayed within the scene video such that these manual coders had knowledge of the scene motion. We performed this comparison for eye tracking data collected during two different tasks involving different types of scene motion. These two tasks included a subject walking around a building for about 100 seconds (Task 1) and a seated subject viewing a computer animation (approximately 90 seconds long, Task 2). It took our manual coders on average 75 minutes (stdev = 28) and 80 minutes (17) to code results from the first and second tasks, respectively. The automatic fixation-identification algorithm, implemented in MATLAB and run on an Apple 2.16 GHz MacBook, produced results in 0.26 seconds for Task 1 and 0.21 seconds for Task 2. For the first task (walking), the average percent difference among the three human manual coders was 9% (3.5) and the average percent difference between the automatically generated results and the three coders was 11% (2.0). For the second task (animation), the average percent difference among the three human coders was 4% (0.75) and the average percent difference between the automatically generated results and the three coders was 5% (0.9).


Keywords: fixations, eye tracking, dynamic scenes

* e-mail: susan.m.munn@gmail.com

1 Introduction

Eye tracking is used to identify where a person is looking. Light entering our eye is captured on our retina and the portion of this light that is captured at our fovea, close to the center of our retina, is perceived in greatest detail. It is commonly accepted that regions in the world at which our eye is directed correspond to regions of importance or interest to the viewer [Irwin 1992]. Eye tracking is used to capture these points-of-regard (PORs) as an observer performs a task. The raw data stream output of an eye tracker typically contains a list of POR positions that indicate the approximate points, over the course of the data collection, at which an observer’s fovea is directed within a given scene. These POR data are then analyzed to determine regions that an observer is drawn to, information that is gathered by the observer, etc.

Depending on the length of time during which POR data is being collected and the sampling rate of the eye tracker, the amount of POR data can be overwhelming. Fixation-identification routines are commonly used to group these POR data into fixations such that a minimum number of consecutive PORs that fit within a specified radius around a point in the world are considered to belong to a single fixation. These routines, which may be coded within an automatic algorithm or performed manually by a user, can greatly simplify (and are sometimes required for) the analysis of eye tracking data. Most current fixation-identification algorithms assume that the POR data are defined within a single image; in other words, they assume that the observer and everything viewed by the observer is stationary. Therefore, we were interested in whether or not one of these routines would be applicable to gaze data defined in dynamic scene videos.

The analysis of POR data in dynamic videos can be very complex [Goldstein et al. 2007] and produce ambiguous results. There is no commonly accepted or well-defined way of performing this analysis. Some researchers who analyze POR positions in video [Peters and Itti 2006; Carmi and Itti 2006] tend to look at saccades as opposed to fixations. Saccades are rapid movements of the eyes to a new region on the retina. Therefore, a saccade detector will identify saccades using a velocity threshold which is also a common way to identify fixations (where fixations fall below the threshold and saccades fall above it). In the case of a stationary observer viewing a static 2D image, the observer will execute a saccade-fxate behavior quite recognizable by a velocity-based identification routine. If the observer and/or the fixated object in the scene is moving, then the observer may execute smooth eye movements to stabilize a region on the retina. Because they have lower velocities than saccades, it might be harder to discriminate these smooth eye movements from fixations; this is what we aim to determine.

A user manually coding POR data into fixations can take into account movements in the scene video when distinguishing fixations from saccades. The drawback of manual coding is that the user must view the output POR data overlaid in a video of the scene
one frame at a time. This process can be very time-consuming, cumbersome and subjective. We compare the fixation results manually obtained by three experienced users to the results of a simple velocity-based fixation-identification routine for two different tasks involving different types of scene motion. For the first task, a subject walked around a building (looking for a room number) such that motion in the scene video was due to the movement of the scene camera (which moved with the subject). For the second task, a seated subject watched a computer animation such that motion in the scene video was due to movements in the animation.

The following section presents the background for this paper including further description on eye tracking, eye movements and fixation-identification routines. Section 3 will describe the velocity-based fixation-identification algorithm that we implemented and Section 4 will discuss how we compared this algorithm to manual coders and the results we obtained. We will discuss our conclusions in Section 5.

## 2 Background

In this section we present background on eye tracking and the data obtained via eye tracking. We also briefly discuss eye movements, fixations and a few methods for identifying fixations.

### 2.1 Eye tracking

Modern video-based eye trackers, or Video-Oculography (VOG) systems, may be remote or head-mounted. All video-based eye trackers utilize a video camera to record the subject’s eye. Portable head-mounted eye trackers incorporate a scene camera in addition to this eye camera so that the subject’s view is recorded as the subject moves, allowing for free body and head movement without loss of information on where the subject is looking in the world. Remote eye trackers, on the other hand, do not necessarily require a scene camera. These remote trackers allow the subject to be free from the eye tracking apparatus but require the subject to be seated and are typically used in accordance with two-dimensional stimuli presentations. The video output by a remote tracking system is typically composited from what is being displayed on a computer monitor in front of the subject. The algorithm that we implemented here is applicable to both head-mounted and remote eye trackers.

Some video-based methods, like the one used in this paper, involve tracking the center of the pupil and a corneal reflection (CR). An example eye frame is shown in Figure 1. By setting a grayscale threshold value at an appropriate level, the pupil region can be isolated. Locating the pupil center is robust and relatively low in noise because the pupil subtends a relatively large area. The CR region in the eye image is much smaller than the pupil region (Figure 1); for this and other reasons, the tracked CR positions tend to be more noisy than the pupil positions. When grouping PORs into fixations, this noise should be considered.

Aside from noise, we must also account for track losses when analysing eye tracking data. A track loss occurs when the eye tracking software fails to properly detect the pupil or CR centers. This may occur as a result of the eyelids and/or eyelashes occluding portions of the pupil and/or CR (e.g., when the subject blinks). Also, especially with varying illumination, it can be difficult to find a consistent threshold for segmenting the pupil or CR from the background for all frames of the eye video in which case, for a few frames, other portions of the eye image can be incorporated into the determination of these centers or there may be “holes” in these regions that affect this calculation. Track losses are common and intermittent but usually affect less than 5% of the final data (based on our findings). Nevertheless, it is necessary to take them into account when analyzing the data.

### 2.2 Eye movements and fixations

The oculomotor system serves two main purposes: (1) stabilization of a target on the retina, and (2) movement so that a new target or area can be imaged at the fovea. Stabilization is necessary to allow the photoreceptors time to react to incoming light as well as to allow time for the signals to be sent to and processed by the brain. Movement of the fovea is necessary because the distribution of cones in our retina, and therefore visual acuity, decreases greatly from the fovea towards the periphery. Typically eye tracking studies look at the occurrence of saccades and fixations during “simple tasks” (e.g., reading) but less restrictive eye trackers, including the RIT Wearable Eye Tracker [Babcock and Pelz 2004] used for this study, can record different types of eye movements as they are executed during more complex tasks.

The eye movements to be studied via monocular eye tracking experiments consist of saccades, smooth pursuit, optokinetic nystagmus (OKN) and vestibular-ocular reflex (VOR). Saccades are rapid movements made to shift the point of regard. In contrast, smooth pursuit is employed to track a moving object. OKN is similar to smooth pursuit in that it contains a smooth movement invoked to stabilize an image of a moving target on the retina except it is involuntarily elicited by repetitive movement of one’s entire field-of-view or an object subtending a large portion of the field-of-view; this occurs, for example, when watching a train pass by. OKN movements include pairing smooth pursuit to track the object or scene and a saccade to bring the observer’s focus back to its previous position. VOR, also invoked by head or body movements, stabilizes the image of a fixated object as the head or body moves relative to the object. These eye movements are all described in greater detail by Carpenter [1988].

Fixations are stabilizations of the eyes for high acuity at a given point. The eyes may be moving within the head during a fixation but remain approximately stationary with respect to the fixated target. Whether or not this can be considered a fixation, due to the fact that the eye is moving, may be debatable but the fact that the person is still looking at the same thing (which is what is important to us when grouping POR data) remains and therefore we include this in our use of the term fixation. The durations of fixations are often measured and directly related to a subject’s interests because of increased visual attention during fixations [Just and Carpenter 1976]. According to Salvucci and Goldberg [2000], fixations are rarely less than 100 ms and typically range from 200 to 400 ms although these numbers vary slightly in the literature (e.g., Irwin [1992] states the range as 150 to 600 ms) and are highly dependent on task. Miniature eye movements occur during fixations and are often referred to as fixational eye movements. These movements affect the orientation of the eye by less than 5° [Lerotic and Yang 2006] and therefore are below the noise level in video-based eye tracking systems.
2.3 Fixation-identification algorithms

Fixation-identification algorithms in use today are well-described but not well-evaluated in the literature. Most of these algorithms rely on identifying fixations from POR data defined in a single two-dimensional scene image. In other words, these algorithms process raw 2D POR position data to determine fixations assuming that if the image coordinates defined by the POR positions are spatially close together, the corresponding points in the world are also spatially close together and in a static environment. They are typically very fast and beneficial for analyzing eye tracking data collected by stationary eye trackers and for studying tasks that do not require smooth eye movements but their applicability to POR data that vary in dynamic scenes is unknown. Some different types of fixation-identification algorithms are discussed in this section. Information presented in this section on these algorithms is summarized from a taxonomy by Salvucci and Goldberg [2000].

Fixation-identification algorithms can be classified as velocity-based, dispersion-based or area-based depending on the spatial criteria they employ [Salvucci and Goldberg 2000]. Velocity-based algorithms take advantage of the fact that saccades are rapid movements while fixations are pauses on particular regions. Velocity is considered a spatial criteria, in this sense, because with a known sampling rate, only the distance from the previous sample needs to be used for a velocity measure. Dispersion-based methods take into account the fact that PORs within fixations are spatially close together. The criteria used by velocity-based and dispersion-based methods are extremely similar but the way in which they are used determines their distinction. Dispersion-based algorithms evaluate the dispersion between all 2D points within a fixation and would therefore be more sensitive to scene motion over the duration of a fixation. Therefore, we do not believe that dispersion-based techniques (e.g., I-DT and I-MST [Salvucci and Goldberg 2000]) would be applicable to dynamic scene data. Area-based methods (e.g., I-AOI [Salvucci and Goldberg 2000]) simply categorize PORs as being within fixations if they are within designated areas-of-interest. Of course, these methods are not robust at distinguishing saccades from fixations (which is our goal) but are used to quantify the time spent in specific regions. We will focus the remainder of this section on velocity-based techniques as these show the most potential for analysing gaze data in dynamic scenes.

Temporal criteria may also play a part in fixation-identification methods, in which case they can be further categorized as duration-sensitive and/or locally-adaptive. Duration-sensitive algorithms take advantage of the fact that fixations have a minimum duration and that their durations fall within an expected range. Locally adaptive algorithms allow the classification of a particular POR (i.e., fixation or saccade) to be influenced by the classification of surrounding PORs.

2.3.1 Velocity-based methods

As mentioned, velocity-based fixation-identification algorithms are based on the fact that PORs during saccades are separated by higher velocities than PORs during fixations. With a constant sampling rate, velocities are simply measured using the distance between sample points. Salvucci and Goldberg [2000] provide examples of two different velocity-based methods which they term Velocity-Threshold Identification (I-VT) and Hidden Markov Model Identification (I-HMM). I-VT is the most basic – simple and easy-to-implement – fixation-identification algorithm. It simply uses one parameter, a velocity threshold, to distinguish saccades from fixations. If the distance between the current sample and the previous sample is below threshold, the sample is categorized as a fixation point otherwise it is categorized as a saccade point. Basic I-VT does not use any temporal criteria but only requires a single parameter, is extremely fast and can be easily implemented for real-time application.

I-HMM methods, as their name entails, use Hidden Markov Models to determine the most likely pattern of fixations and saccades via probabilistic analysis. I-HMM is not as fast or easy to implement as I-VT but, according to Salvucci and Goldberg [2000], is more accurate and robust. A disadvantage of I-HMM is that it requires 8 parameters to be set: 4 for the observation probabilities (mean and variance for the velocity distribution of each state) and 4 for the transition probabilities (2 for each state = 1 for remaining in state + 1 for leaving the state).

2.4 Manual coding of fixations

To manually code gaze data, a person watches a composite video and codes, or scores, information about gaze behavior. In the composite video, a crosshair is superimposed on each frame indicating the POR for that frame. Also, a small picture-in-picture image of the eye, from the frame of the eye video that corresponds in time to the displayed scene frame, is typically positioned somewhere over the scene frame (Figure 4). While this composite video can be viewed at varying speeds, it is typically evaluated one frame at a time. Often, the video is viewed in software that allows timestamping of events that are coded by the user via keystrokes or mouse clicks.

A large advantage of manual coding over automatic coding is the human’s powerful and accurate ability to perform pattern recognition. For instance, instead of coding the start and end of fixations, a human can code more descriptively the start and end of fixations on task-relevant objects (e.g., a teapot in a “find the teapots” task) versus fixations on task-irrelevant objects (e.g., a vase in the same task) whereas an algorithm would have a much more difficult time distinguishing these events. For these types of analyses, an automatic fixation-identification algorithm can be used as a first step to make the manual coding process more efficient. For instance, instead of requiring the coder to find the start and end of every fixation (which requires the coder to navigate the frames one at a time around the vicinity of the desired frames) the coder can just be given the middle frame for each fixation as determined by the automatic algorithm, with the POR overlaid, and append the desired information (e.g. teapot or vase). This would greatly reduce the time for the coder to code the video and greatly reduce the repetitive and tedious aspects of manual coding. On the other hand, a more advanced algorithm may also be devised that performs this pattern recognition and be incorporated into the fixation-identification algorithm.

3 Algorithm

We chose to implement a duration-sensitive version of the I-VT algorithm. I-VT was chosen over I-HMM for its simplicity, speed and small number of parameters. Since I-VT is sensitive to noise, we smoothed our output POR data over 2 frames (≈ 67 msec). We made our I-VT implementation duration-sensitive because human coders are typically given instructions as to the minimum duration required for a fixation. The disadvantage of making the algorithm duration-sensitive is that this makes the algorithm no longer applicable for real-time applications (because the start of a fixation is not determined until the minimum duration is met).

Once again, we use the term fixation to mean the same target is being looked at; therefore, we include in this definition situations in which the observer is tracking a target (performing smooth pursuit) or looking at a target while his head is moving with respect to the target (making VOR movements). A flowchart of the algo-
Algorithm we implemented is presented in Figure 2; all parameters and variables are presented in Table 1. Track losses (including blinks) are identified by PORs that fall outside of the bounds of our scene frames; PORs that fall within these bounds we refer to as good PORs. We chose to treat track losses the way a human coder might. Track losses that occur in the middle of a fixation are ignored and the duration of the track loss is counted towards the duration of the fixation. Track losses that occur any time between fixations (including those that occur immediately before or immediately after a fixation) are ignored and not counted towards fixations. Keeping a record of when these track losses occur (via the `trackloss` array) mediates our calculation of fixation durations. Of course, one may revise this treatment of track losses as desired; for instance one could treat blinks differently from other track losses (if they can be distinguished, e.g., via knowledge of pupil diameter).

The `trackloss` array is initialized to an N x 1 array of zeros indicating, initially, that no track losses exist. Before each track is processed, we check to determine if the POR is outside of the image bounds. A POR outside of the image bounds (y or y coordinates < 1, x > 720, or y > 480) is designated as a track loss, the `trackloss` array is updated and processing skips to the next POR. In the case of a good POR, the distance between the POR (b_{\text{por}} = [x_b, y_b]) and the previous good POR (a_{\text{por}} = [x_a, y_a]) is computed. This distance, the standard Euclidean distance $d = \sqrt{(x_b - x_a)^2 + (y_b - y_a)^2}$, is then compared to $d_{\text{max}}$. If $d$ is less than $d_{\text{max}}$, the two PORs are potentially within one fixation, otherwise they are not. A count, `potfix`, is updated each time a POR matches this criteria. When a POR no longer matches the criteria, `potfix` is reset to 1. When `potfix` reaches the minimum fixation duration, $t_{\text{min}}$, a fixation is found and the start of the fixation is determined (using `potfix` and `trackloss`). The fixation ends as soon as a POR no longer fits within $d_{\text{max}}$ of the previous good POR and the duration of the fixation is calculated (also using `potfix` and `trackloss`).

```
Figure 2: Flowchart of algorithm. The expression $d(a_{\text{por}}, b_{\text{por}})$ represents the Euclidean distance between $a_{\text{por}}$ and $b_{\text{por}}$.
```

### Table 1: Parameters and variables used in the algorithm.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$d_{\text{max}}$</td>
<td>Maximum distance between consecutive PORs in one fixation.</td>
</tr>
<tr>
<td>$t_{\text{min}}$</td>
<td>Minimum fixation duration.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F$</td>
<td>Current number of fixations (initial $F = 0$).</td>
</tr>
<tr>
<td>$N$</td>
<td>Total number of video frames.</td>
</tr>
<tr>
<td>$a_{\text{por}}$</td>
<td>2D image coordinates of POR in frame a.</td>
</tr>
<tr>
<td>inafix</td>
<td>Flag, 1 = in fixation, 0 otherwise (initial inafix = 0).</td>
</tr>
<tr>
<td>potfix</td>
<td>Count for the current number of matched PORs (initial potfix = 1).</td>
</tr>
<tr>
<td>fixstart</td>
<td>$F \times 1$ array containing frame indices for start of each fixation.</td>
</tr>
<tr>
<td>fixdur</td>
<td>$F \times 1$ array containing durations of each fixation.</td>
</tr>
<tr>
<td>trackloss</td>
<td>$N \times 1$ array. $trackloss(a) = 1$, if frame $a$ represents a track loss, else $0$.</td>
</tr>
</tbody>
</table>

### 4 Testing and results

Two male subjects (mean age = 38.5) were eye tracked while performing different tasks using the RIT Wearable Eye Tracker [Babcock and Pelz 2004], a portable video-based eye tracker; this eye tracker captured video at 30 fps. Both subjects had been eye tracked before and volunteered for the study. Subject 1 was tracked while walking through the hallways of a building with the task of finding a specific room number that he did not previously know the location of; this will be referred to as Task 1 in our analysis. The second subject was tracked while watching a short computer animation that he had never seen before (Task 2). These tasks were intended to provide examples of very different dynamic scenes. Two videos were produced as a result of these tracks; these output videos consisted of the grayscale scene video captured by the scene camera (attached to the eye tracker headgear) that moved with the observer’s head with a crosshair overlaid indicating the observer’s POR and the corresponding frame of the eye video inset in the top left corner for each frame (as in Figures 4(a) and 7(a)).

The two videos were each manually coded by three experienced users (1 male undergraduate, 1 female undergraduate and 1 female graduate student); we will refer to these users as coders. These coders had an average of about 300 hours of coding experience and had a fundamental knowledge of eye tracking, eye movements and human vision. The coders were given no information about the I-VE algorithm. Coders were given the instructions to code the start and end of fixations (with tags: “Start Fixation” and “End Fixation”) and that the minimum acceptable fixation duration was 6 video frames (200 ms); this is the same value used within our algorithm for the minimum fixation duration. These instructions are typical instructions given to coders. Our interpretation of the term fixation was clearly described to the coders as time during which the person was assumed to be looking at the same point on an object; their interpretation of this assumption was entirely subjective and they did not discuss with each other the decisions that they made.

Our I-VE algorithm, implemented in MATLAB, processed only raw POR data (i.e., it did not use any video frames). This is in contrast to the output composite videos – containing the scene frames, POR crosshairs and inset eye frames – used by the manual
coders to determine fixations. Each frame of these videos contained 480×720 pixels. For Task 2, the animation that the subject watched did not span the entire scene video (Figure 7(a)).

4.1 Parameter setting

For our implementation, we break up our maximum distance \((d_{\text{max}})\) parameter into two components such that \(d_{\text{max}} = r_{\text{un}} + d_{\text{VOR}}\). The purpose of the first component, which we term the radius of uncertainty \((r_{\text{un}})\), is to account for noise in eye tracking data and calibration error. We computed this value to be the maximum error in POR during calibration (Task 1: \(r_{\text{un}} = 15\); Task 2: \(r_{\text{un}} = 10\)). The second component \((d_{\text{VOR}})\) was added for Task 1. We found that the motion of the subject’s head (and consequently the scene camera worn by the subject) in the case of the subject walking around needed to be taken into account because of the VOR movements elicited that were not present in the case of Task 2. We chose our value of \(d_{\text{VOR}}\) for Task 1 based on the distance between PORs that were within one fixation but occurred during large scene camera motion (and consequently VOR movements). We set \(d_{\text{VOR}} = 10\) for Task 1 and \(d_{\text{VOR}} = 0\) for Task 2. It was sufficient to use a large \(d_{\text{max}}\) for Task 1 because the task of walking around a building looking at room numbers elicited larger saccades then the task of watching an animation.

We made our implementation of the I-VT algorithm partially duration-sensitive by requiring fixations to have a minimum time duration, \(t_{\text{min}}\). We did this because manual coders are typically provided a minimum fixation duration criterion so, for fair comparison, we provided our algorithm with the same value. We set \(t_{\text{min}} = 6\) (6 frames = 200ms for our 30 Hz eye tracker) based on the values presented by Salvucci and Goldberg [2000]; \(t_{\text{min}}\) should be altered proportionally for different frame rates (e.g., for a 60 Hz eye tracker, \(t_{\text{min}} = 12\) frames = 200ms). Of course, some may want to use a smaller threshold for the minimum fixation duration. We did not set a maximum fixation duration because we include in our fixation definition times during which the subject is performing VOR and smooth pursuit, and these durations can be much longer than a fixation during which the subject and fixated object are both stationary. We did not give our manual coders any instructions on maximum fixation duration.

4.2 Task 1 results

For Task 1, the subject wore the RIT Wearable Eye Tracker and looked at 5 specified points marked on a dry-erase board in front of him for calibration. The subject was then asked to find a specific room (room number 2155) in the building. This task required the subject to walk through hallways around one floor of the building. The task took the subject about 1 minute and 45 seconds (including calibration) resulting in a 3,143-frame composite video for the manual coders to code and 3,143 POR data samples for the algorithm to process. An electronic strobe was triggered at the start and end of the task to synchronize the results of the I-VT algorithm to those of the manual coders. The algorithm classified 6% of the data from Task 1 as track losses.

Figure 3(a) shows the coded fixation results across every frame of the video. Each strip in this figure spans 449 frames. White areas represent frames that were coded as within fixations and black areas represent frames that were not coded as within fixations (saccades and track losses). The wide black region in the middle of the fourth strip is due to a long track loss caused by a sudden change in illumination. The first three rows of each strip present the results for the three manual coders, the fourth row contains the results from the I-VT algorithm and the bottom row designates disagreements (white = disagree, black = agree). The term ‘disagreement’ is used in this context to represent the case where the algorithm disagreed with the unanimous agreement of the manual coders. These disagreements can be the result of the algorithm coding a fixation as longer or shorter than the three coders, the algorithm missing a fixation identified by the three coders, the algorithm combining two fixations (as coded by the manual coders) into one or the algorithm splitting one fixation (as coded by the manual coders) into two. For Task 1, 13.6% of the frames contained a disagreement. We investigate two of these disagreements, arguably the most severe, in the Task 1 data: (1) the algorithm missed an entire fixation that all the manual coders coded; and (2) the algorithm coded a fixation that all the manual coders missed. We reserve examples of disagreements due to incorrect (missing or extra) breaks in fixations for our Task 2 results.

Figure 4 shows an example of a disagreement during which the algorithm coded a fixation that all the manual coders missed; this occurred at the location of the first arrow in Figure 3(a) (second strip). Based on this figure (and viewing the video through these frames), it appears that this is a true fixation and we speculate that the manual coders missed this fixation because they were playing the composite video at full speed through a long track loss and accidentally skipped over this fixation. The very next disagreement in Figure 3(a) can be described in the same manner.

Figure 5 shows an example of a fixation that was coded by all manual coders but missed by the algorithm; this disagreement occurred at the location of the second arrow in Figure 3(a) (second to last strip). The frames in this figure were all coded as being within a fixation by the manual coders but not by the I-VT algorithm. The first frame in this set of frames represented a track loss and was therefore intentionally skipped by the I-VT algorithm. The POR in the last frame of this set was also affected by errors in the eye tracking software. The manual coders were able to notice and account for these errors because they were able to see the eye frame that corresponded to these points. The eye frame in these images show that the eye is steady and therefore the frames were manually coded as being within a fixation. The I-VT algorithm was not given access to this information and therefore could not make this determination.

Figure 3(b) shows the results of correlating each of the signals displayed in Figure 3(a). Correlation gives us a measure of the similarity between two signals such that a correlation value of 1 for a pair of signals indicates that the two signals are identical and a value of -1 indicates that the two signals are exact opposites. The correlation matrix displayed in Figure 3(b) is shaded such that higher correlation values (signal pairs that are most similar) are displayed over lighter squares. These correlation values show us that the most similar pair of results was between manual coders 1 and 3 (MC-1 and MC-3). The results of the I-VT algorithm were slightly more similar to MC-1 than MC-2 was to MC-1 and most similar to the results produced by MC-3.

4.3 Task 2 results

For Task 2, Subject 2 watched the 99-second animation “A Great Big Robot From Outerspace Ate My Homework” on a computer monitor; this animation was screened at SIGGRAPH 2006. Permission to use this video was obtained from Vancouver Film School. Since the subject was sitting down for this task, the backpack of the RIT Wearable Eye Tracker was placed on the table next to the subject who wore the headgear containing the eye and scene cameras. The subject was instructed to sit comfortably and watch the movie as he desired; his movements were not restricted in any way. The subject sat approximately 90 to 100 cm away from a 37” computer monitor with 16:9 aspect ratio. The animation was displayed at about 30 × 45 cm so that it subtended an approximate viewing
Figure 3: (a) Task 1 signals from fixations coded by three manual coders (MC-1 to MC-3) and the I-VT algorithm. In the top 3 rows of each strip, white regions indicate fixations and black regions are between fixations. The bottom row of each strip contains a white vertical bar every time the results of the I-VT algorithm disagree with all three manual coders (i.e., DA=disagreements). The first arrow in this figure (second strip) indicates the region displayed in Figure 4 and the second arrow (second to last strip) corresponds to the set of frames shown in Figure 5. (b) Correlation of fixation signals for Task 1.

Figure 4: Consecutive frames coded as within one fixation by the I-VT algorithm but not by any of the manual coders. Picture-in-picture of eye frames was enlarged for illustration.

Figure 5: Consecutive frames coded by all manual coders as within one fixation but missed by the I-VT algorithm. Picture-in-picture of eye frames was enlarged for illustration.

angle of 18° in the vertical direction and 26° in the horizontal direction. These values are approximate because the subject was free to move during the viewing.

We calibrated the subject by having him look at 9 points on the computer monitor. The composite video to be coded was cut into a 90-second clip that started at the first frame of the animation and ended at the last frame of the animation before the credits; the video was shortened (calibration and credits removed) to reduce the time required of our manual coders. The output video to be manually coded contained 2,694 frames (2,694 POR data samples for automatic analysis). As in the case of Task 1, a strobe was triggered at the start and end of the task to synchronize the results of the I-VT algorithm to those of our manual coders. 1.5% of the data from Task 2 were categorized as track losses.

Figure 6 shows the results for Task 2 across every frame of the video (as described in Section 4.2 for Figure 3(a)); 5.5% of the frames contained a disagreement between the algorithm and all manual coders. For this task, there were no disagreements in which an entire fixation was missed by the algorithm or coded by the algorithm and missed by all manual coders. Figure 7 shows a region, indicated
by the first arrow in Figure 6, over which the manual coders marked the end of one fixation and the start of a new fixation while the I-VT algorithm connected these two fixations into one single fixation. Over this time period, it appears as though the subject may be looking from the character’s left eye to his nose but the POR is hardly moving. This was determined to be a saccade by all three manual coders but not by the I-VT algorithm.

Figure 8 corresponds to the second arrow in Figure 6 and a region during which the I-VT algorithm separated two fixations but all three manual coders connected these frames into one fixation (the opposite disagreement as in Figure 7). It appears from Figure 8 (and from watching the video through these frames) that the subject is looking from the robot’s chin to the robot’s eyes. None of the three manual coders separated these frames into two fixations. We assume that the reason the manual coders did not mark this break in fixations is because it occurred during the end of the video and the coders were fatigued. The time it took each coder to code these videos is presented in Table 2 and discussed in the following section.

Figure 6(b) shows the correlation values for Task 2. The I-VT algorithm matched both MC-2 and MC-3 better than MC-1 matched MC-2 and MC-3. The correlation values for Task 2, overall, are lower than those for Task 1. This is most likely due to the fact that saccades during the animation viewing task were much smaller and their distinction more subjective.

### 4.4 The time factor

Table 2 tabulates the time it took each manual coder to code the two video tapes as well as the time it took the I-VT algorithm to process the POR data. Each manual coder coded the composite video from one task during a single sitting. The I-VT algorithm was implemented in MATLAB and run on an Apple 2.16 GHz MacBook. The I-VT algorithm took 0.26 seconds to produce its results for Task 1 and the manual coders took an average of 80 minutes to produce their results. For Task 2, the I-VT algorithm took 0.21 seconds and the manual coders took on average 75 minutes. Also, MC-1 made an error in the process of coding her results for Task 2 which required correction. She did not code one “End Fixation” such that two “Start Fixation” codes appeared in a row. She was asked to revisit the composite tape and mark the frame for the missing “End Fixation” code before the manual results were compared to the I-VT results.

## 5 Conclusion

Manual coding is a very tedious and expensive process but can lend itself to very informative results and greatly enhance the analysis of eye tracking data. Fixation-identification prior to manual coding can mediate the manual coding process so that it is less tedious and time-consuming. Since most current fixation-identification algorithms typically just utilize raw POR data and process them based on distance or velocity, their applicability to eye tracking data collected in dynamic scenes was in question. We investigated the performance of a velocity-based duration-sensitive fixation-identification algorithm on data collected during tasks involving different types of scene and eye motions. In comparing the performance of this algorithm to results obtained by three experienced coders, the algorithm performed remarkably well. Whether we are viewing static or dynamic stimuli, saccades must be executed to bring our eyes from one fixation to another and the speed of these saccades is much greater than that of VOR or smooth pursuit. Therefore, velocity-based identification has shown great potential for identifying fixations even when these fixations are within dynamic scenes. In some cases, the algorithm actually performed better than the manual coders (examples were provided in Figures 4 and 8).

Since we are effectively blind during saccades, by separating saccades from fixations, we can quantify the amount of time spent tak-
ing in information versus time spent relocating the eyes. The I-VT algorithm that we implemented could be used on POR data collected by any video-based eye tracker – remote or portable.

We used our fixation-identification results from the I-VT algorithm to enhance our composite videos that visualize our eye tracking results (see color plate, Figure 9). These videos – which previously included the eye frame, scene frame and POR crosshair for each frame – were appended with a circle around the POR, if the algorithm determined that the POR was part of a fixation. The color of this circle alternated between red and blue for every other fixation so that the end of one fixation and the start of another could be clearly conveyed. If more descriptive information is desired of each fixation, manual coders could code these descriptions using this video. This would eliminate the tasks of advancing frame by frame to find the start and end of fixations, counting the number of consecutive frames matched to determine if the minimum fixation duration has been reached, and the potentially difficult task of deciding whether or not PORs are within the same fixation. The coder could simply pause the video each time the color of the fixation circle changes and append the desired description. These new output composite videos that we created for each task with the results of the I-VT algorithm are available at http://www.cis.rit.edu/mvrl/APGV.

Figure 7: (a) Composite frame coded by the manual coders. (b-f) Zoomed-in views of one location from consecutive frames; (b) is from the same frame as (a). (d) Coded as a one frame break between two fixations by all manual coders and not by I-VT (I-VT coded all these frames as within the same fixation). [Animation courtesy of Vancouver Film School]

Fixation-identification in dynamic scenes is the first step towards categorizing fixations. For example, one may wish to separate fixations into the types of eye movements (or lack thereof) executed (e.g., fixed eye-in-head, smooth pursuit and VOR). Distinguishing these movements may be possible automatically by computing the motion in the scene video along with the motion of the POR or performed by the manual coders via the visualization described in the previous paragraph. Similarly, the I-VT algorithm could be used as a first step towards coding the objects that are fixated via automated pattern recognition or user interaction.

Figure 8: (a) Frame for reference of region size in scene video. (b-g) One region cropped from consecutive video frames. (d,e) Coded by I-VT as between two fixations while all frames were coded as within the same fixation by all manual coders. [Animation courtesy of Vancouver Film School]

We have started to investigate a way to incorporate scene knowledge into the I-VT algorithm to improve fixation-identification. So far, incorporation of scene knowledge has only lowered the performance of the I-VT algorithm. Also, the need to load each scene frame and compute information from these scene frames greatly increases the processing time of the algorithm. Our implementation of the I-VT algorithm exhibited a desirable balance of speed and accuracy.

References


Fixation-identification in dynamic scenes: Comparing an automated algorithm to manual coding
Munn, Stefano and Pelz

Figure 9: Frames from a video compiled using the results of our implementation of the I-VT algorithm on Task 2. The green crosshair shows the POR obtained by the eye tracker. A circle indicates that the POR is in a fixation and the color of the circle changes to signify a switch from one fixation to another (18 frames from the middle of the “blue” fixation were removed to conserve space). [Animation courtesy of Vancouver Film School]