Real-time and Offline Filters for Eye Tracking

PONTUS OLSSON

KTH Electrical Engineering

Master's Degree Project
Stockholm, Sweden April 2007

XR-EE-RT 2007:011

Abstract

An eye tracker makes it possible to record the gaze point of a person looking at for example a computer monitor. Modern techniques are very flexible and allow the user to behave naturally without the need of cumbersome equipment such as special contact lenses or electrical probes. This is valuable in psychological research, marketing research and Human Computer Interaction. Eye trackers also give people who are severely paralyzed and unable to type and speak means to communicate using their eyes.

Measurement noise makes the use of digital filters necessary. An example is an eye-controlled cursor for a desktop environment such as Windows. The cursor has to be stable enough to allow the user to select folders, icons or other items of interest. While this type of application requires a fast real-time filter, others are less sensitive to processing time but demand an even higher level of accuracy. This work explores three areas of eye tracking filtration and aims at enhancing the performance of the filters used in the eye tracking systems built by Tobii Technology, Sweden. First, a post-processing algorithm to find fixations in raw gaze data is detailed. Second, modifications to an existing reading detection algorithm are described to make it more robust to natural irregularities in reading patterns. Third, a real-time filter for an eye-controlled cursor to be used in a desktop environment is designed using a low-pass filter in parallel with a change detector.

The fixation filter produced fewer false fixations and was also able to detect fixations lying spatially closer together than the previously used filter. The reading detection algorithm was shown to be robust to natural irregularities in reading such as revisits to previously read text or skipped paragraphs. The eye-cursor filter proved to respond quicker than the previously used moving average filter while maintaining a high level of noise attenuation.

Contents

Acknowledgements

I would like to thank everybody who made this project possible; my supervisor at KTH Björn Johansson, examiner Bo Wahlberg and my supervisor at Tobii, Johan Bouvin. I would also like to thank all testing volunteers, Frida Nilsson for proofreading and my family for endless support and love.

Introduction

Vision is our most prominent sense; our eyes are in constant motion during the day in order to provide us with information. By observing the motions of the eye, it is possible not only to tell where a person is looking, but also to recognize symptoms of certain diseases and draw conclusions about the cognitive mechanisms responsible for how we perceive the world around us.

By using a modern eye tracker it is possible to record where a person is looking, for example on a computer screen. The technique is used in many different fields such as psychology research, cognitive science, marketing research and human-computer interaction. It is also used as a communication channel for people who are severely paralyzed and unable to type and speak. With an eye tracker, a large part of this group can use a PC to communicate via email, speech synthesis or chat.

Due to measurement noise it is necessary to filter the measured gaze position. Depending on the application, a real-time (online) filter or a post-processing (offline) algorithm can be used. Applications requiring real-time filtering include eye-controlled communication, or interaction with a desktop environment such as Windows. Offline algorithms can be used in analysis of recorded data, where the processing time is less critical. An example is marketing studies where the gaze-points of a number of test subjects are recorded while viewing an advertising poster. The goal is to determine if the poster conveys its message with satisfactory efficiency. An offline filter is used to remove noise before further analysis. Tobii Technology, a company developing and manufacturing eye trackers initiated this study to evaluate if the performance of the currently used real-time as well as post-processing filters could be enhanced.

This report begins with an overview of the physiology and motions of the human eye followed by a summary of common eye tracking methods. A chapter on change detection is included due to its great importance in gaze-data filtering. It contains a description of a common algorithm, Cumulative Sum (CUSUM). Three areas of eye tracking filtration have been explored in this work, and are discussed in their respective chapters:

- An offline fixation filter based on change detection to find fixations in noisy data and to estimate their spatial position is proposed in chapter 4, *Offline Fixation Filter*.

- Improvements of an algorithm used to detect reading patterns are suggested in chapter 5, *Reading Filter.*

- A real-time algorithm based on a change detector in parallel with a low-pass filter to achieve satisfactory noise attenuation and a very fast step response for a gaze controlled cursor for the Windows environment is proposed in chapter 6, *Online Cursor Filter.*

Results concerning these three areas are presented at the end of each respective chapter. At the end of the report general conclusions and future research directions are discussed.

1. The Human Eye

In order to design filters to improve human-computer interaction based on eye tracking it is essential to understand the functionality of the human eye.

1.1 Physiology

The human eye is illustrated in figure 1 [1]. The visible parts are the sclera, the white protective coating around the eye, the iris, the colored part, and the pupil, the opening in the middle of the iris. Light rays first enter the eye through the cornea, which is a clear, dome-shaped window covering the iris and the pupil. It is responsible for approximately 2/3 of the focusing power in the eye [1]. After passing through the cornea, the light rays have to pass the pupil in order to reach deeper into the eye. The iris is embedded with muscles able to adjust the diameter of the pupil in order to control the amount of light entering the eye. Inside of the pupil is the lens. The focusing power of the lens can be adjusted by muscles, to adapt the focus to near or far objects.

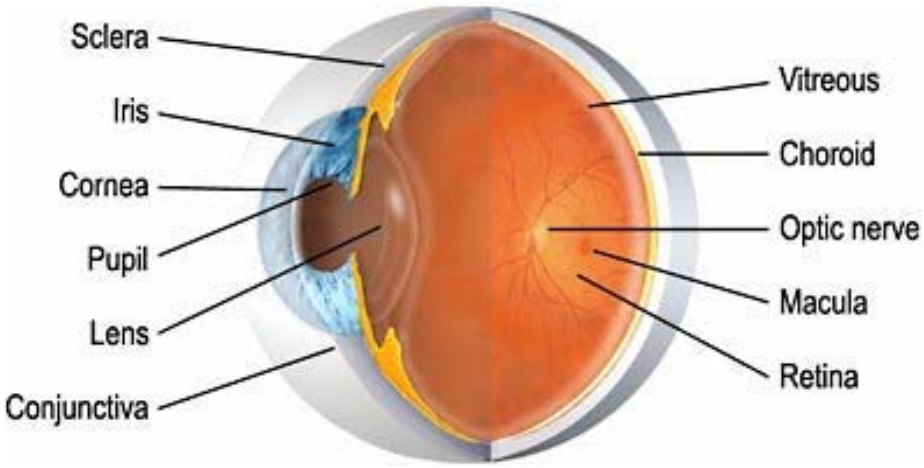

Figure 1. The human eye.

An upside-down picture of the world in front of the eye is projected on the retina at the back of the eye. It is a multi-layer sensory tissue containing photoreceptors that convert light-rays into electrical impulses. These impulses are transmitted through the optic nerve to the brain where they are interpreted as images.

There are two types of photoreceptors in the retina, rods and cones. The approximately 125 million rods are spread out over the retina. They cannot perceive colors, but they work well in dim light and are responsible for our peripheral- and night vision. The approximately 6 million cones are able to perceive colors, but require relatively bright light to function. The cones are located within the area of the retina responsible for central vision, called the macula. The highest density of cones is found within the center region of the macula, the fovea. This is the most sensitive part of the retina, and is used when reading, examining objects, driving, or whenever a high level of detail is of importance [1].

The fact that the acuity is not uniformly distributed across the retina is very useful for eye tracking purposes. The acuity of the retina outside the foveal region is 15 to 50 percent of that of the fovea [2], which is often insufficient to read for example a word. This means that in order to reach the required level of detail, the eyeball has to be rotated so that the foveal axis points directly at the object of interest.

The width of the fovea is about one degree of visual angle [2]. Images can be seen sharply within the whole foveal region, and it is possible to move attention around within this region without moving the eye. Therefore it is impossible to know where a person aims his/her attention with greater accuracy than one degree of visual angle only by observing the eye. At a viewing distance of 50 cm from the screen, this corresponds to an accuracy of approximately 9mm.

1.2 Eye movements

The eye is held in place by three pairs of muscles responsible for left-right motion, up-down motion and rotational motion around the axis from the fovea to the pupil.
The eyeball generally changes orientation with series of quick movements called saccades. These movements are very quick, lasting between 30 and 120 ms [3]. Saccadic movements are ballistic, meaning once a movement is started, the direction cannot be changed during the saccade. Saccades are often followed by a period where the eye is held stable in order to view an area of interest, for example a word or part of a word during reading. These stable periods are called fixations and typically range between 150 and 600 ms [3]. When fixating an object in motion, such as a car passing by, or a football during a soccer game, the eyes follow the object of interest in a smooth manner called smooth pursuit. The smooth pursuit motion consists of a smooth component and corrective saccadic motions to correct tracking errors between gaze point and point of interest. Another type of eye movement is nystagmus. It occurs when presented to a moving pattern, for example when looking through a train window at the moving landscape. The eyes try to follow the pattern, but will regularly snap back. During fixations, the eyes continue to make quick eye-movements of very small amplitude, called micro saccades. The micro saccades are required to maintain vision, since the photoreceptors and neurons in the brain responsible for visual perception would otherwise adapt to the non-changing stimulation much like we lose perception of a scent when exposed to it long enough. Micro saccades movements typically range between 2 to 120 arc-minutes [3]. (1 arc minute = 1/60 of a degree)

2. Eye tracking

In the past, eye tracking was often a cumbersome procedure, where the head of the subject had to be held absolutely still by using a chin-rest or asking the subject to bite on to a fixed structure, limiting their use mainly to research laboratories. They were also often intrusive, meaning that a part of the measurement equipment was in direct contact with the subject. Eye trackers of today are much more flexible, and non-intrusive methods have been developed making it possible to use the technology in more natural environments. The eye tracker used for this work is based on corneal reflection, since it is non-intrusive and flexible.

2.1 Electro-Oculography (EOG)

Electro Oculography uses the fact that the electrical potential on the skin around the eyes changes when the eyes are moved [4]. Electrodes are placed on the skin to measure the potentials, which after amplification can be used to calculate the angular position of the eye. This method can only measure eye position relative to the head. In order to calculate the point of regard, a head tracker has to be used to measure the position of the head.

2.2 Search Coil

By using a contact lens containing a wire coil working as a magnetic field sensor it is possible to achieve very accurate measurements of the angular position of the eyeball, down to 5-10 arc-seconds (1 arc-second = 1/3600 of a degree) over a limited range of 5º [4]. An electromagnetic field is applied and the embedded sensors provide measurements used to calculate their orientation. It is important that the lens is sufficiently large to avoid slipping. These lenses often cover not only the cornea, but also a portion of the white sclera to provide sufficient friction to the surface of the eye. Due to their size the insertion task requires skill, and since they often have wires attached to them they are relatively cumbersome to use.

2.3 Corneal Reflection

A modern approach to track eye motions is to use infrared light emitting diodes causing reflections (glints) on the cornea. A camera sensitive to infrared light captures images of the eyes and an image processing algorithm estimates the center of the pupil and the positions of the glints [5]. When the eyes move, the glints on the corneas remain in approximately the same places which make them useful as reference points. By calculating the relationship between the pupil and the glints the angle of the eyes can be estimated. The point of gaze on for example a computer monitor can be calculated if the head position relative to the monitor is known or estimated. This method does not require the subject to wear cumbersome equipment and is not restricted to a designated lab environment.

2.4 Hardware

The gaze position is acquired using a Tobii 1750 screen, based on the corneal reflection technique. The monitor is a 17” TFT monitor with a resolution of 1280x1024 pixels. The subject is illuminated by five near-infrared diodes placed above the screen (three) and below the screen (two). An image of the subject is obtained by a camera sensitive to infrared light placed below the screen.

Figure 2. Tobii 1750 eye tracker

2.5 Software

Data is collected using ClearView and Tobii Studio, software for gaze-data analysis by Tobii Technology. These programs are used to set up usability test studies where a number of participants interact with for example a website. All eye movements are tracked and can be analyzed using an array of visual tools and filters, for example how much time the user on average spends viewing a specified region of interest. From these studies it is possible to gain insights in how to enhance the site to make it easier to use and navigate. The software is also used to evaluate commercial material, both images and movie clips.

2.6 Calibration

The eye tracker has to be calibrated before use because the features of the eyes may differ from person to person. The user is asked to fixate on a number of calibration points shown on the screen while gaze-data is recorded. This data is used to create a personal mapping from gaze-point to a point on the screen. The number of calibration points can be chosen and an increased number will result in a more accurate mapping (less offset error) between the real and the estimated gaze point, to the cost of a more extensive calibration process. A calibration pattern consisting of five points is a good compromise between accuracy and easy of calibration and was used in all test studies in this project.

2.7 Signal Characteristics

Data acquisition is done at 50 Hz. Several data items are acquired; horizontal and vertical gaze-point obtained using the corneal reflection technique described in section 2.3, position of the eyes relative to the screen and pupil diameters. The gaze-point measurement from the tracker is presented using normalized coordinates where [0, 0] represents the upper left corner and [1, 1] represents the lower right corner. To facilitate the design of eye-controlled applications these coordinates are often transformed into screen coordinates:

$$s_x(n) \in \{1,...,res_x\}$$
$$s_y(n) \in \{1,...,res_y\}$$

$s_x(n)$ is the horizontal position and $s_y(n)$ is the vertical position. res_x and res_y are the horizontal and vertical screen resolutions, respectively. $s_x(n) = s_y(n) = 1$ represents the upper left corner of the screen and $s_x(n) = res_x$, $s_y(n) = res_y$ represents the lower right corner. In this study a resolution of 1280 x 1024 pixels was used. Since the measurement offset error and the magnitude of the measurement noise is well above the diameter of a pixel, the quantization error effect is negligible.

The appearance of the signal varies depending on the activity the subject engages in. However, we assume that given a static stimuli such as a picture or a text, the motion pattern of the eyes will consist of:

- Fixations of different temporal length
- Saccades of different spatial length and direction, and different temporal length

An example of a signal is shown in figure 3. It is a recording of a subject viewing water lilies. The signal contains fixations, saccades and noise.

Figure 3. A recording of a subject viewing water lilies. The signal consists of subsequences of relatively constant mean, quick jumps and noise.

To analyze the noise, a subject was asked to fixate a static point on the screen while the gaze-point was recorded. Since the signal represents a two dimensional gaze-point it is easier to illustrate the characteristics of the signal by plotting each dimension separately versus time, see figure 4.

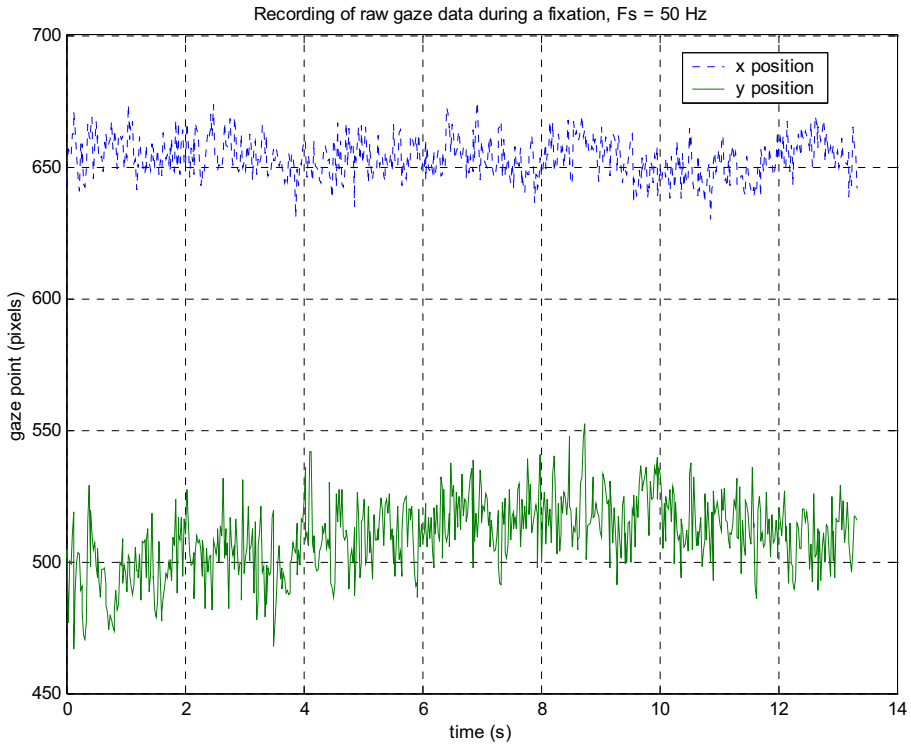

Figure 4. Horizontal (x) and vertical (y) gaze-point versus time. The subject was asked to look steadily at a point, but the signal is not of constant mean. It consists of a high frequency noise component and a low-frequency drift component.

The signal contains a high frequency measurement noise due to noise in the image obtained by the camera, non-static light conditions and imperfections in the image processing algorithm used to estimate the gaze-point. By filtering the recording using a second order Butterworth high-pass filter with a cutoff frequency of 2.5 Hertz (see chapter 6.1, *Filters*) and plotting the distribution in a histogram (figure 5) we see that the high frequency noise resembles Gaussian noise. The variance of the noise can vary between different users. It also tends to increase when the angle between the optical axis of the eye and the optical axis of the camera increases. The fixation point in figure 4 is located in the middle of the screen, which means that this angle is larger vertically than horizontally. (The camera is centered horizontally below the screen.) This may explain the larger vertical variance.

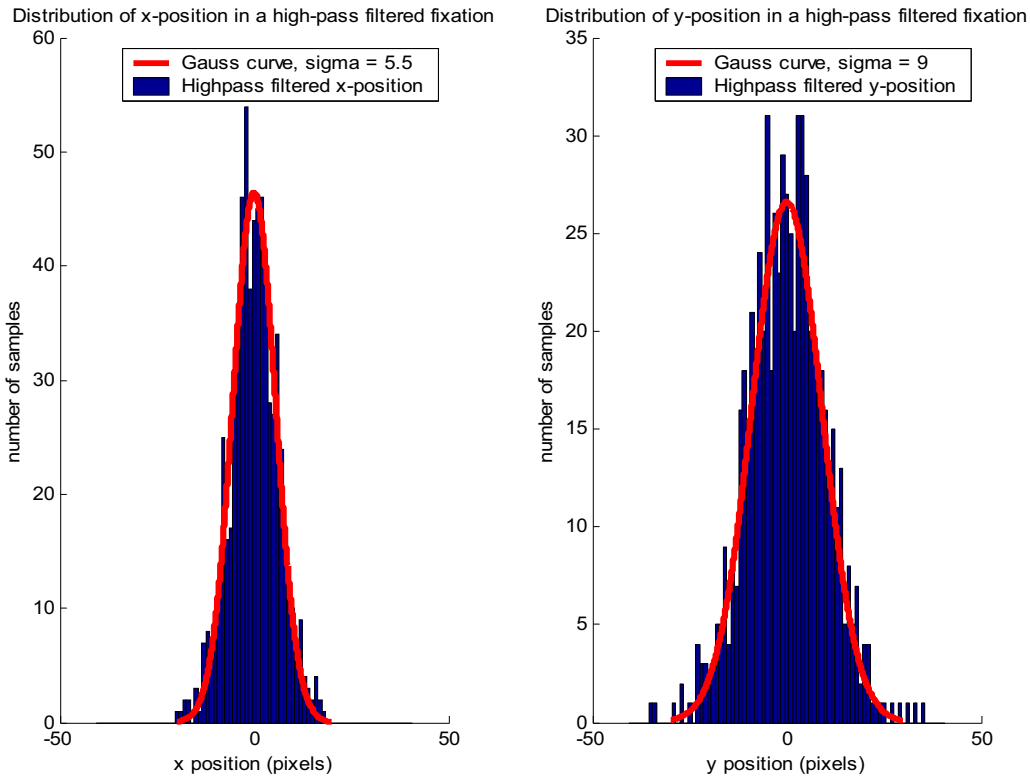

Figure 5. High frequency noise distribution, compared to a Gaussian bell-curve. In this case the standard deviation (sigma) is approximately 5.5 pixels horizontally and 9 pixels vertically.

The recorded fixation also contains a lower frequency drift. This drift can be explained by the fact that it is difficult to keep the eyes absolutely fixed. When trying to fixate a point the eyes tend to drift off slightly, which is corrected by a corrective saccade if the original point still draws our attention. The pupils also change diameter slightly during fixations even if the light conditions are static. This phenomenon will affect the estimate of the gaze position. Head movements also slightly affect the estimate. Unfortunately, individual differences make this effect unpredictable so it is difficult to suppress fully. In addition, there may also be occasional spikes due to blinks present in the signal. Note that fixations as long as the fixation in figure 4 are very rare, however the experiment illustrates that we can not expect a stationary stochastic signal during fixations.

3. Change Detection

When the user fixates on a point on the screen, the measured position will resemble Gaussian noise with a slow change of the mean. The high frequency noise is measurement noise and the low frequency changes are due to a drift caused by pupil size changes and head movements. When going from looking at one point on the screen to another, the signal mean will change rapidly. Thus, in order to separate the fixations from the saccadic motions in the signal, a change detector able to detect fast changes in the signal mean is required. There are many different methods to detect changes in a signal, used for example to detect faults in systems and to quickly give an alarm. As an orientation on how a change detector can function, the commonly used CUSUM method is here described as an example.

3.1 Cumulative Sum (CUSUM)

A common method to detect changes in the mean is Cumulative Sum, CUSUM. The CUSUM algorithm to detect changes in a one dimensional data series is given below [6].

For each sample, the following lines are executed:

$$\begin{aligned} \hat{\theta}_t &= \lambda \hat{\theta}_{t-1} + (1-\lambda) y_t \\ \varepsilon_t &= y_t - \hat{\theta}_{t-1} \\ g_t^{(1)} &= \max(g_{t-1}^{(1)} + \varepsilon_t - \nu, 0) \\ g_t^{(2)} &= \max(g_{t-1}^{(2)} - \varepsilon_t - \nu, 0) \end{aligned} \qquad (1)$$

where y_t is the measured signal containing noise and possibly changes in the mean. The first line describes a low-pass filter producing an estimate, $\hat{\theta}_t$ of the signal. When the signal is of constant mean, $\hat{\theta}_t$ will be a good estimate and the prediction error ε_t is small. By tuning λ the low-pass effect of the filter can be adjusted and thus making the change detector more or less sensitive to gradual changes in the mean. The next two lines function, as the name of the algorithm suggests, as cumulative sums.
Whenever the positive or negative prediction error exceeds a threshold ν, the variables $g_t^{(1)}$ or $g_t^{(2)}$ start to increase. Lastly, $g_t^{(1)}$ or $g_t^{(2)}$ are compared to a threshold h. If the threshold is exceeded, an alarm is given and $g_t^{(1)}$ and $g_t^{(2)}$ are set to zero. The two variables $g_t^{(1)}$ and $g_t^{(2)}$ accumulate the positive and negative prediction error, respectively. This makes the change detector *two-sided*, meaning it can detect both positive and negative changes in the mean. By tuning the variable h, it is possible to make the detector more or less sensitive to changes in the signal. If h is decreased, the detector will be more sensitive to small changes, but the risk of yielding false alarms is increased. Hence, a tradeoff exists between sensitivity and a low false alarm rate.

The cumulative nature of the algorithm makes it robust to noisy data, but since the sum has to build up and surpass the threshold before an alarm is given, the estimated jump times will be delayed. This motivates the investigation of another approach, see chapter 4.1, *Offline fixation filter algorithm*.

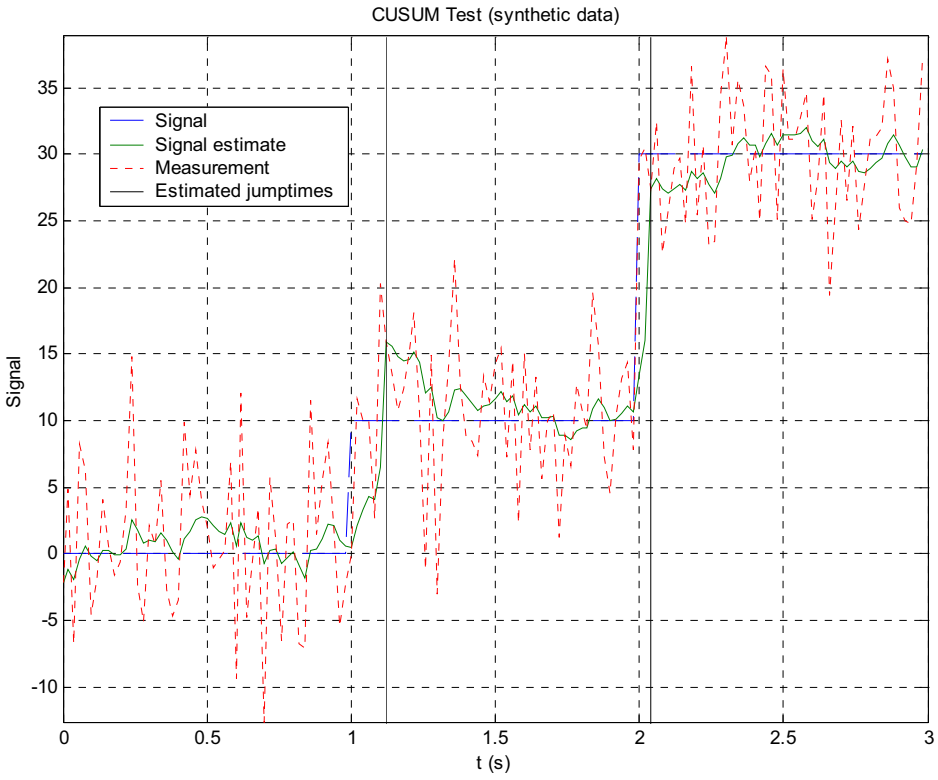

Figure 6. CUSUM test of noisy signal with jumps at t = 1 and t = 2. Noise σ = 5.0, λ = 0.85, ν = 2.3 and h = 32. The detector is able to detect the changes in the mean even with significant noise. The estimated jump time is noticeable delayed for small changes in the mean.

4. Offline Fixation Filter

When analyzing gaze data for commercial or scientific purposes, oftentimes the interest is the spatial position and duration of the fixations and the timestamps of the saccades rather than the full set of raw gaze data. There are several reasons for this; firstly a scan-path broken down into fixations and saccades is much easier to overview than a full gaze-plot of raw data containing noise. Secondly, a scan-path containing only position and duration data of fixations and saccades occupy only a fraction of the data storage space required to store the full raw data vector.

Sometimes two consecutive fixation points are so close together that they lie within the variance of the measurement noise. In these situations it is hard to determine if the user looked at one point, or if several points are grouped close together. The fixation filter has to be sensitive enough to separate fixations that lie close together. It also has to be robust to a noisy gaze position signal, producing as few false fixation estimates as possible.

The area based filter used today in the commercial analysis software checks if the spatial Euclidian distance between the last fixation point estimate and the most recent sample is longer than a specified threshold, t. If it detects that this is the case, it looks for a new place to estimate the next fixation point. This is done by finding the next occurrence of a specified minimum number of samples lying within a circle with radius t. The spatial mean of the samples within this new circle becomes the new fixation estimate. This method works well for clear, well separated fixation points with limited noise and constant mean. To avoid false fixation points due to noise and low-frequency drift the threshold radius t has to be set very large, which causes the filter to merge fixations that lie close together spatially. This causes problems in high level analysis, such as reading detection where higher accuracy is needed.

We here assume that the stimuli is stationary, which is the case of a text or a static picture, allowing us to make the assumption that there are no smooth pursuit motions present in the scan-path. The task for the filter is:

- Find the saccadic eye-motions in the signal and separate these from the fixations.
- Estimate the spatial position of the fixations.
- Return a list of the estimated fixation positions and their durations.

4.1 Algorithm

A post-processing algorithm was developed to estimate the spatial position and duration of the fixations. A description of the algorithm using pseudo code can be found in chapter 9.1 *Pseudo code for Fixation Filter.*

We denote the desired "true gaze position" data vector $\vec{t}(n)$

$$\vec{t}(n) = [t_x(n) \;\; t_y(n)] \tag{2}$$

Where $t_x(n)$ and $t_y(n)$ are the horizontal and vertical position of the "true" gaze-point in each sample, n. Unfortunately $\vec{t}(n)$ is affected by noise $\vec{e}(n)$ allowing us only to see the measured raw signal $\vec{s}(n)$

$$\vec{s}(n) = \vec{t}(n) + \vec{e}(n) \tag{3}$$

$\vec{e}(n)$ consists of a high-frequency Gaussian measurement noise, a low frequency drift due to head movements and pupil size changes, and occasional spikes due to blinking. (See chapter 2.7, *Signal Characteristics*). We want to use our knowledge about the signals $\vec{t}(n)$ and $\vec{s}(n)$ to recreate an estimate $\hat{\vec{t}}(n)$ that is as similar to $\vec{t}(n)$ as possible. Assuming the only eye motions present in the scan-path are fixations and saccades, we make the following distinction:

- If a segment of the signal is of constant or slowly changing mean due to drift, we classify it as a fixation.
- If there is an abrupt change in the mean, we classify it as a saccade.

By designing an algorithm that produces an alarm whenever it detects a quick change in the mean, we separate the saccades from the fixations. When this is done, we can estimate the spatial position of each individual fixation based on the information contained within its timeframe. A very simple and intuitive approach to find changes in the mean is to consider the length of the difference vector between each sample.

$$d(n) = \sqrt{(s_x(n) - s_x(n-1))^2 + (s_y(n) - s_y(n-1))^2} \tag{4}$$

By comparing *d(n)* with a threshold *h*, this method produces an alarm when there is an abrupt change in the mean. The threshold *h* has to be chosen large enough to avoid false alarms due to the noise. Unfortunately, there are several drawbacks with this method. Firstly, the fact that h has to be chosen larger than the variance of the measurement noise means that the detector will fail to detect all changes in the mean that lie within the noise variance. Depending on the magnitude of the noise, and the required precision of the fixation detection this may be a problem.

Secondly, if a saccadic motion spans over several samples and the difference vector *d* is larger than *h* in each of these samples, the detector will return several alarm signals where there was in fact only one saccade. To improve the detector, it was extended to consider the difference between the *means* of the sample points within two sliding windows rather than simply the difference between two samples.

$$\vec{m}_{before}(n) = \left[\frac{1}{r}\sum_{k=1}^{r} s_x(n-k), \frac{1}{r}\sum_{k=1}^{r} s_y(n-k) \right] \tag{5}$$

$$\vec{m}_{after}(n) = \left[\frac{1}{r}\sum_{k=1}^{r} s_x(n+k), \frac{1}{r}\sum_{k=1}^{r} s_y(n+k) \right] \tag{6}$$

$$d(n) = \sqrt{(\vec{m}_{after}(n) - \vec{m}_{before}(n)) \cdot (\vec{m}_{after}(n) - \vec{m}_{before}(n))^T} \tag{7}$$

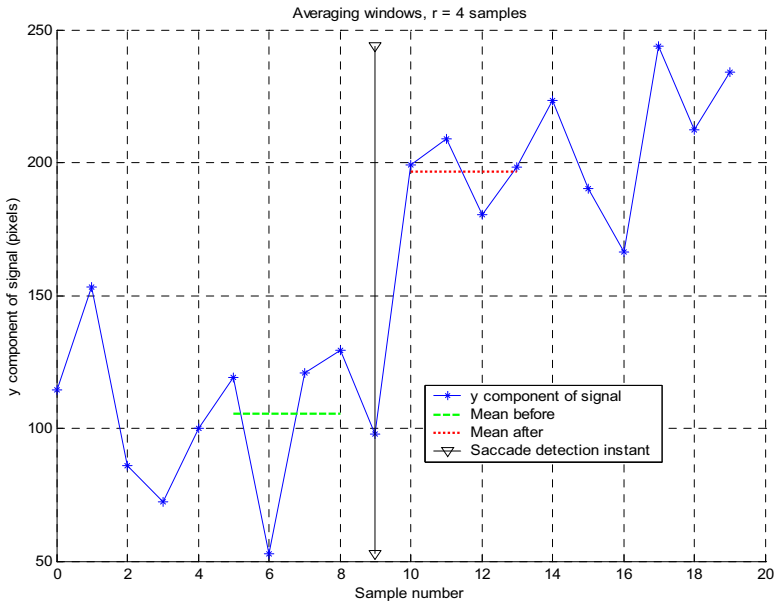

Figure 7. Averaging windows, r = 4 samples. The difference between the two sliding window means will be large when they are on different sides of the change. If there is no change, the mean difference will remain small. Here, the largest difference between the two means was found at sample number 9.

The longer these windows are defined, i.e., the larger value we chose for *r*, the less impact we get from the noise. The difference between the two means reaches a peak when the two sliding windows are on each side of a change in the mean, a saccadic motion. We have seen that it is desired to make the averaging windows as long as possible in order to reduce the impact of the noise. However, we cannot make them infinitely long, since we then risk spanning several saccadic motions within a single averaging window that will result in unreliable alarm signals.

According to Duchowski [3], fixations last between 150 and 600 ms. However, occasional faster fixations do occur. After examining gaze data collected from 10 subjects doing regular work in a Windows environment it was concluded that fixations lasting less than 4 samples equaling 80 ms are very rare. It was therefore decided to set the length *r* of the averaging windows to 4 samples each. Sometimes a large multi-sample saccade will generate more than one peak within the span of *r* samples. Therefore we also assume that if more than one peak is found within the span of *r* samples, only the highest of them is valid.

To find the saccades, all peaks in the *d* vector are identified. A peak is defined as a sample with a higher value than the previous and following sample. All peaks that are lower than a specified threshold are removed to avoid peaks caused by noise and to allow a slow drift without splitting the fixation. This step also reduces the computation time required to iteratively group spatially close fixations in a later stage of the algorithm. Since we allow only one peak per sliding window, we keep the highest peak as a saccade candidate and discard the lower peak(s) if more than one peak is found within the span of *r* samples.

Estimates of the spatial position of the fixations between the candidate saccades are calculated using the median of all samples in the interval. (Equation 8) The indices of the candidate saccades (indices of remaining peaks) are here denoted i_k and estimates of spatial fixation positions are denoted $\vec{f}_k$.

$$\vec{f}_k = median(\vec{s}(i_k), \vec{s}(i_k + 1), ..., \vec{s}(i_{k+1} - 1), \vec{s}(i_{k+1})) \quad (8)$$

Hypothetically, if the subsequence $\vec{s}(i_k), \ldots, \vec{s}(i_{k+1})$ contained only Gaussian noise with a constant mean, the mean gives a lower MSE than the median as seen in figure 8.

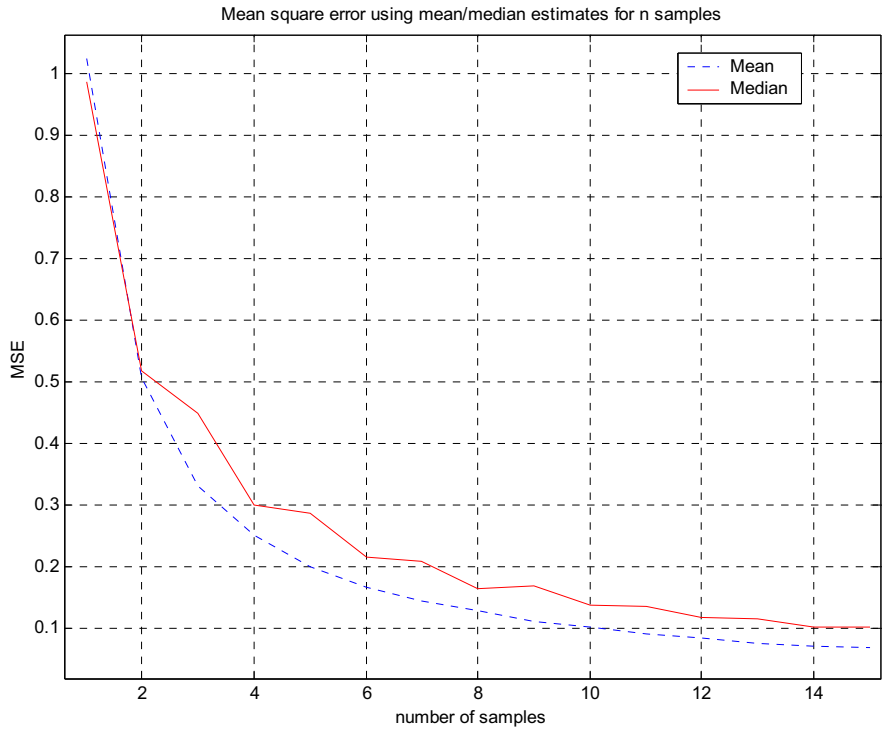

Figure 8. MSE of estimates of a sequence containing n samples Gaussian noise with constant mean, σ = 1.0. In the ideal case of a stationary signal with constant mean, the mean estimate is more accurate. As expected, both the median and mean produce more accurate estimates when using a larger number of samples.

However, the subsequence $\vec{s}(i_k),\ldots,\vec{s}(i_{k+1})$ begins with the end of a saccadic motion at index i_k and ends with the beginning of the next saccadic motion at index i_{k+1}, which notably affects the mean. This motivates the use of the median, see figure 9.

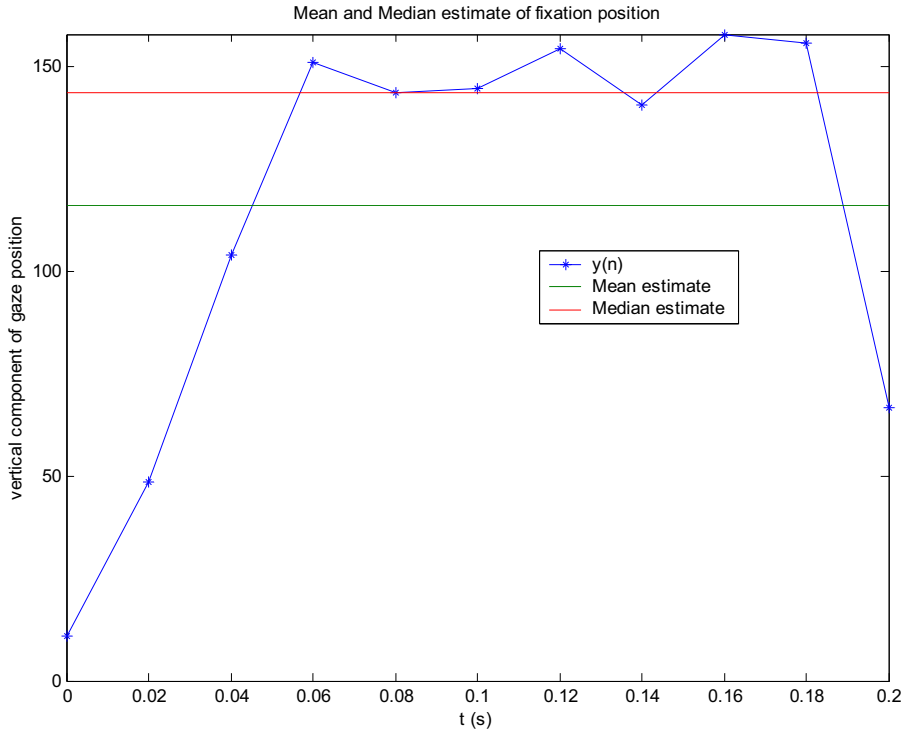

Figure 9. Subsequence of a gaze-plot containing a fixation separated by saccades at t = 0 and t = 0.2. The mean includes a part of the saccadic transients in the estimate, which is not desired. By using the median this problem is avoided, provided the fixation duration is longer than the transient parts.

The estimate of the spatial position of the fixation is affected by the inclusion of the saccadic transitions when using the mean, but not when using the median. An alternative approach would be to form the mean of only the temporal center of the fixation, thus ignoring a specified number of samples assumed to belong to the surrounding saccades. However, if the number of ignored samples is fixed, we risk ignoring relevant data points actually belonging to the fixation of interest since the temporal length of the saccades is not fixed.

Finally, fixations that are closer together than a specified range, *h*, are merged together. The $\vec{f}_k$ vector is searched for the fixations that lie closest together, these are merged first. When two fixations are merged, the peak index that divides them is removed. A new single fixation point is then estimated by using equation 8 on the set of gaze points previously used to make two separate fixation estimates.
Subsequently, fixations with increasing intermediate distance are grouped until no fixations closer than the specified range *h* are found.

4.2 Results

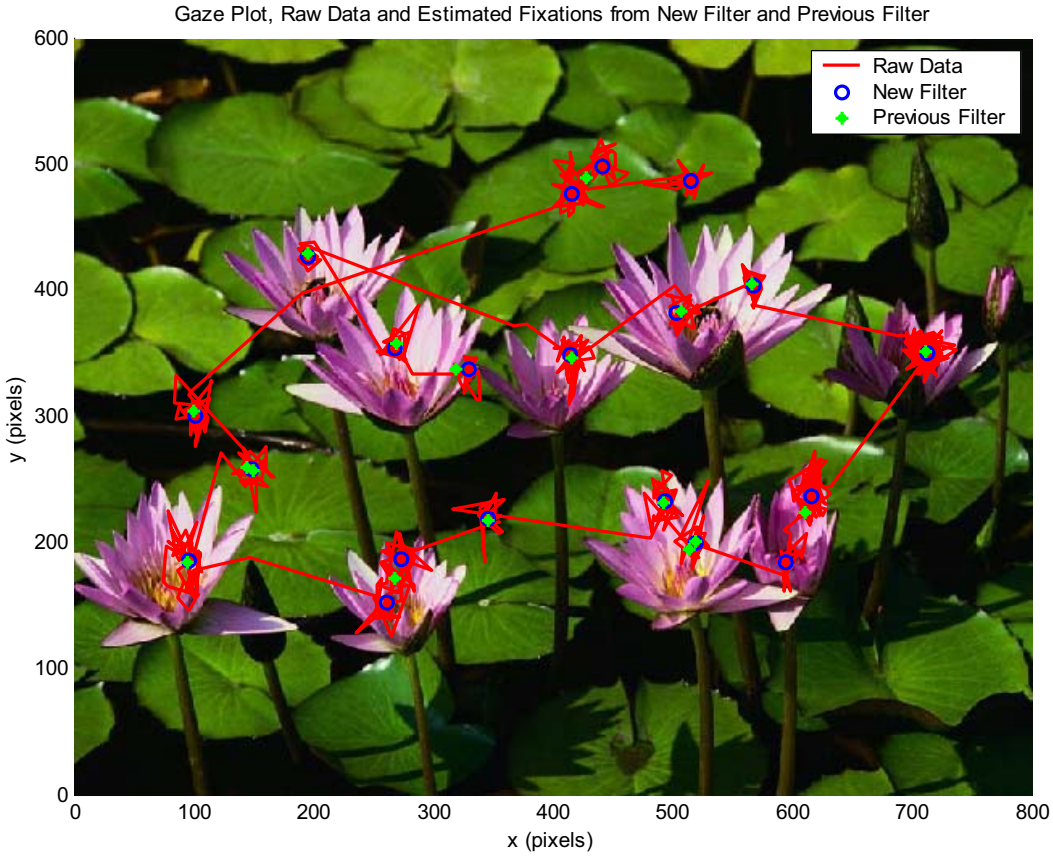

Figure 10. Raw (noisy) data and estimated fixations.

4.3 Conclusions

The proposed fixation filter is able to separate fixations that lie closer together than the previously used, area based filter. It also produces fewer "false" fixations, especially for long fixations where the low-frequency drift is non-negligible. If the recording contains a very high degree of noise, for example if the subject wears thick glasses that reduce the tracking precision, it may be necessary to increase the threshold to avoid undesired division of fixations.

5. Reading Filter

When analyzing recorded scan-paths it can sometimes be beneficial to be able to automatically detect if test subjects have read a certain text. An example could be a company that manufactures toys. The company has just finished the design of a new box to their popular toy “Slim Jim’s Slingshot” and now they want to evaluate if people see the warning text stating “Strictly for outdoor use.”

By defining an area of interest around the warning sign in the analysis software, it is easy to produce statistics measuring to what extent people look at the warning sign. However, to be able to draw conclusions whether people actually read the warning text or not, one has to either go through the scan-paths from all the test subjects and manually determine if the text was read, or employ an automatic reading detection algorithm.

The goal within this study was to find an algorithm able to distinguish between the two cases reading and non-reading. It should be reliable and robust, thus producing few false classifications also in different reading contexts, with subjects of different reading skills and difficulty of text.

5.1 Reading Characteristics

Reading in the western world is in almost all cases conducted left to right, from top to bottom. When we are mildly interested in the topic or when we search for a specific section in a text, we might engage in fast reading, “skimming”. Here, not all words are actually read, but we read a small section here and there, skipping whole lines or whole paragraphs. If a word or a paragraph is hard to understand, it is often re-read to clarify its meaning. These events are called regressions. Studies have shown that there is a distinguishable difference between gaze patterns from subjects engaged in careful reading, skimming a text for a word or paragraph of interest or viewing a stimulus containing no text [7].

In figure 11 and 12 are two examples of recorded reading sequences. Figure 11 shows a nearly ideal reading pattern where all text is read in a linear fashion much like a typewriter. Figure 12 shows a common scenario with skimming and re-reading of words (regressions).

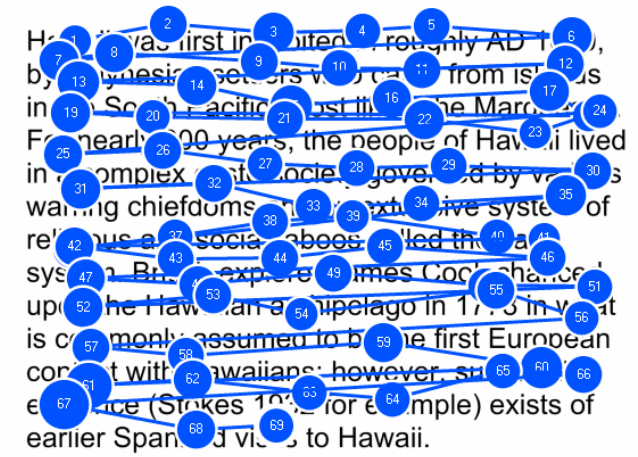

Figure 11. An example of a reading pattern. The subject reads the text carefully and each line is scanned with a number of fixations.

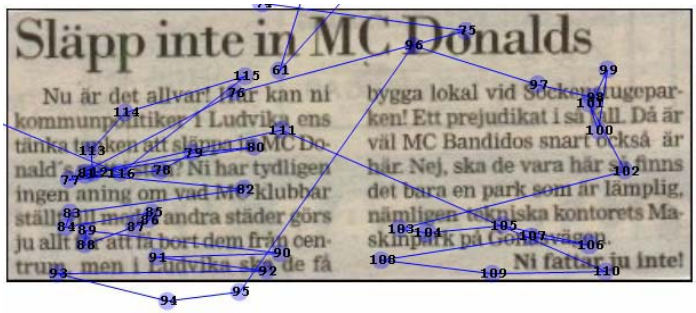

Figure 12. An example of a reading pattern containing regressions and skimming. If the subject is searching for specific information in the text, s/he will often skim the text and not actually pay attention to all words. If a word or a part of the text is difficult to understand, it will often be re-read to clarify its meaning.

5.2 Campbell's reading detection algorithm

An algorithm to detect reading patterns in gaze sequences was suggested by Campbell et al. [8]. The algorithm builds on a state machine with two states; "reading" and "non reading". In order to transit to the reading mode, sufficient evidence suggesting reading has to be accumulated. Evidence is accumulated by considering the difference vectors between the fixations in a recorded scan-path. If they lie within specific regions considered more likely to match reading, a reading score variable is increased. For example, several consecutive short jumps to the right are considered likely to belong to a reading sequence. If this variable exceeds a threshold, the state is changed to reading. Other regions are considered "less likely reading" and will decrease the reading score. Long jumps in any direction but left and slightly down, which is considered "likely reading", since this eye movement is done when transiting to the next row, changes the state back to the non reading state and the reading score is zeroed. The accumulative nature of the algorithm makes it robust to noise, which is necessary since few reading patterns contain only the easily detected sequences of short right-going jumps and row-changing jumps to the left and down.

5.3 Algorithm

It was found that even though Campbell's algorithm is successful in detecting reading patterns, it suffers from a tendency to change back to the non-reading state too easily. During normal reading, occasional long jumps are made to either move the point of gaze to a new column or when re-reading a previous section for clarification. Long jumps are also common during skimming. To handle this problem, instead of having one variable "reading points", two competing variables "reading" and "non reading" were used. Whenever one of them exceeds a threshold, the other is zeroed and a state transit is done. A built-in forgetting factor decreases the "non reading" variable each time the reading variable is increased to limit the impact of old non-reading evidence. The thresholds are set so that it is harder to enter the reading mode than the non-reading mode, but once the reading mode is reached it is tolerant to occasional "non-reading" jumps. To collect evidence, the difference vectors between the fixations are considered. The input to the algorithm is a series of fixations, so any fixation filter with sufficient accuracy can be used. The two dimensional difference vectors between all consecutive fixations are calculated:

$$\vec{d}_k = \vec{f}_k - \vec{f}_{k-1} \tag{9}$$

All fixations $\vec{f}_1, \vec{f}_2, ..., \vec{f}_N$, are described by their horizontal and vertical positions in pixels. Given N fixations, we get N – 1 difference vectors.

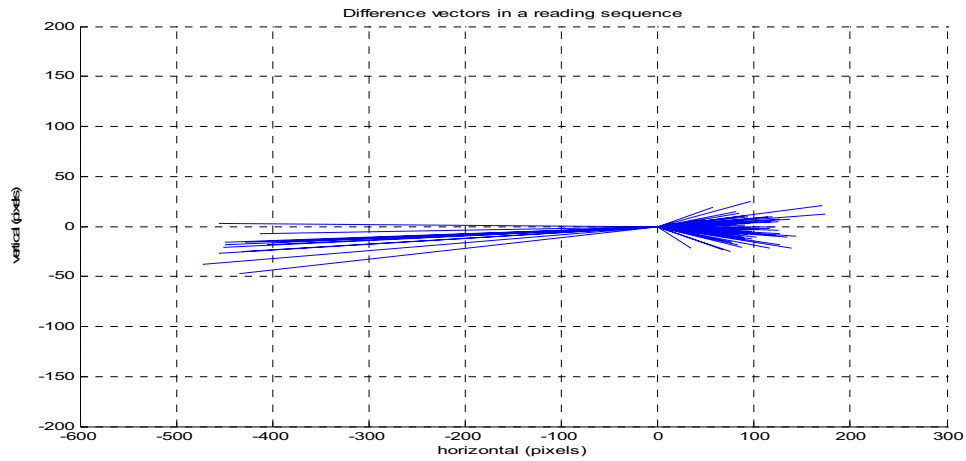

Figure 13. Difference vectors between fixations in the reading sequence in figure 11. The most common pattern when reading is relatively short saccades to the right when reading a line of text and a long saccade to the left and slightly down to proceed to the next line. Occasional short saccades to the left or up occur when re-reading a part of a sentence or a paragraph, (regression) or down when skipping parts of the text (skimming). Here, the subject read the text without regressions.

Whenever a difference vector lies within a predefined area suggesting reading, a reading score is increased by one point. This reading area is surrounded by an area called the neutral area. If a difference vector lies within the neutral area, no points are given. If a difference vector lies outside the reading and neutral areas, a non-reading score is increased by one point. The two cumulative sums compete, and whenever one of them reaches a threshold the state is set to the winning state. At transits to the reading state, it is assumed that the previous α fixations also belong to the reading state where α is the number of fixations suggesting reading required to transit. At transits to the non-reading state, it is assumed that the previous β fixations also belong to the non-reading state where β is the number of fixations suggesting non-reading required to transit.

Whenever a single difference vector lies outside the reading and neutral area, the score counting towards reading is zeroed. The algorithm is tuned this way to make it hard to enter the reading state. The non-reading score is not penalized the same way; it is not zeroed whenever a difference vector lies inside the reading or neutral area. In this case it is instead penalized milder by subtracting a number of points, γ (a forgetting factor) rather than zeroing it. This forgetting factor feature was added to decrease the impact from relatively old non-reading evidence.

These rules are designed so that stronger reading evidence is required in order to change the state to reading compared to what is required to switch back to the non-reading state. Scores are not allowed to grow unlimitedly, but are limited to their respective threshold values α and β. Neither are they allowed to decrease to a negative number. If they were allowed to grow unlimitedly, situations could occur where clear reading sequences are missed because of a large sum of old non-reading evidence. The cumulative evidence component of the algorithm makes it robust to occasional saccades within a sequence of reading or non-reading that would, on its own, suggest the opposite state. This feature is similar to the CUSUM change detection algorithm described in chapter 3, *Change Detection* and is necessary since, as we have seen, natural reading is filled with irregularities such as regressions, skim jumps etc.

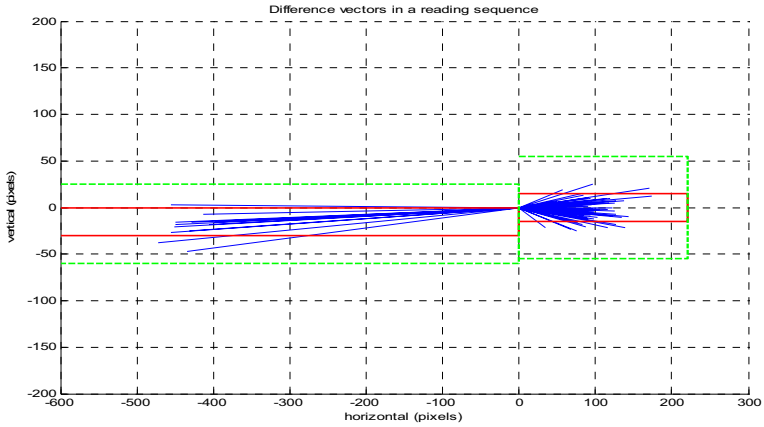

Figure 14. Scoring areas for reading detection. All difference vectors within the red area suggest reading, difference vectors outside the red area but inside the green, dotted area (neutral area) result in no points. Difference vectors outside both the green and red areas suggest non-reading. The areas in the left half-plane span to -∞ since we do not know how wide the text columns are.

As a comparison, a plot of the difference vectors from a non-reading sequence is shown in figure 15. Here, a larger part of the difference vectors lie outside the areas generating reading points.

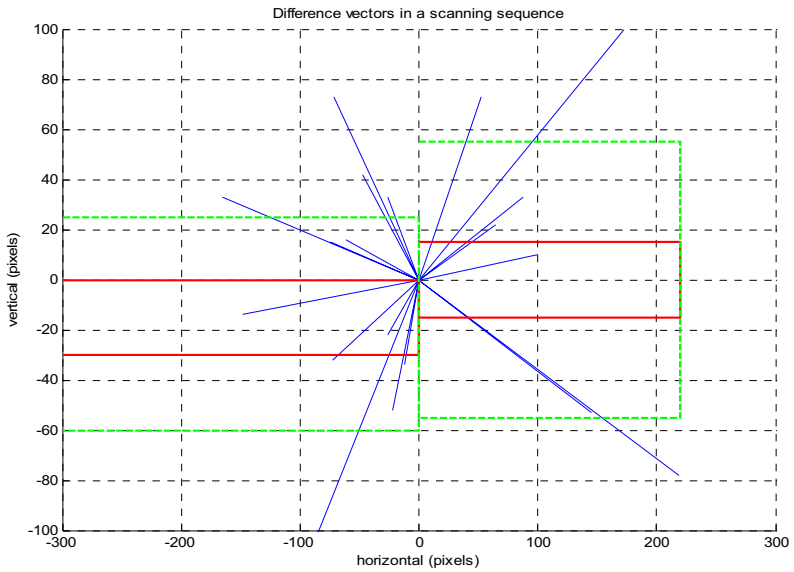

Figure 15. Difference vectors from gaze sequence in figure 3, subject scanning a picture of water lilies.

The sizes of the reading- and non-reading areas shown in figure 14 and 15 as well as the transition thresholds α, β and the forgetting factor γ were chosen by iteratively minimizing the decision error on a data set containing known reading and non-reading fixation sequences. Another set of stimuli and other test subjects were then used to evaluate the performance, see chapter 5.4, *Results*. The constants derived from the iterative process were: α: 5 fixations, β: 2 fixations, γ: 0.2 points.

5.4 Results

An experiment was done where 12 subjects were asked to view a series of 10 stimuli (5 containing only text, 5 containing images but no text, see chapter 9.2, *Stimuli used for evaluation of the Reading Filter*) shown on a monitor with eye tracking capabilities. Each stimulus was shown for 10 seconds, which is a fairly short amount of time to read the texts, so the subjects were told that it is not necessary to read the texts to their ends, but to try to read as naturally as possible without stress. After the experiment, all fixations estimated as reading when a text was shown and all fixations estimated as non-reading when a picture was shown were considered correct. All other fixations were considered incorrect. The subjects were told beforehand that they would be asked questions concerning what they had seen after the experiment to make sure that all items were actually viewed. All participants were able to answer satisfactory to the questions. However, we cannot be sure that the subjects were actually reading the entire time a text was shown; this is a possible error source in the statistics.

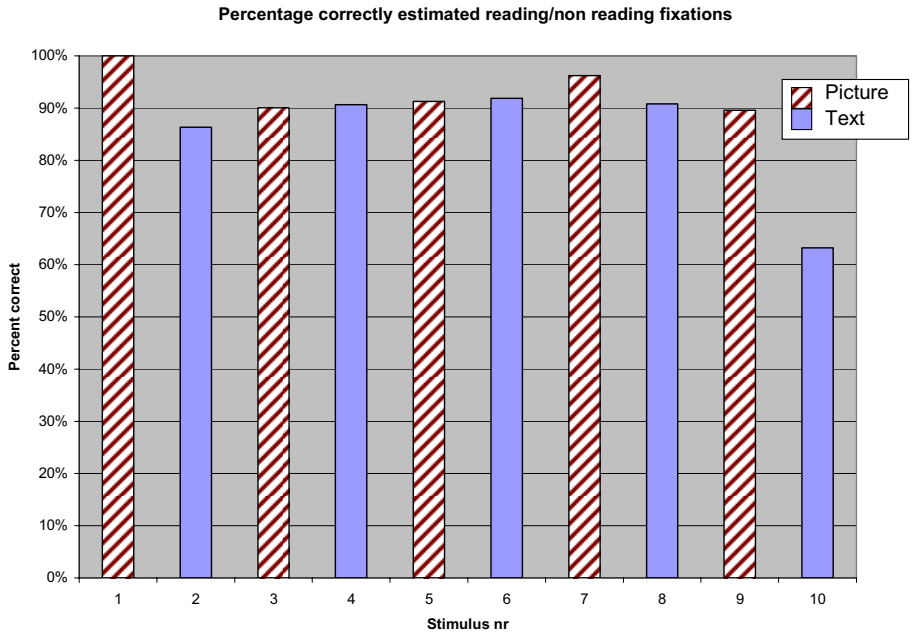

Figure 16. Accuracy of the reading detection filter. Stimulus nr 10 was a recipe containing a narrow list of ingredients. It was difficult for the detector to classify these gaze-patterns as reading due to a large quantity of skimming and regressions.

5.5 Conclusions

The algorithm is able to detect careful reading with high accuracy. Once it has switched to the reading mode, it allows occasional single long jumps in any direction which makes sense when reading a text with several columns. Here, a long jump has to be done to move to the next column. Long jumps are also often done during reading when going back to a previous section to re-read it for clarification. If the subject has a very irregular reading pattern with a high rate of skimming and regressions of whole sections the algorithm might not recognize the reading. Since enough reading points have to be accumulated before it switches to the reading mode, it is not successful in detecting very short reading sequences such as a single sentence or a single line of text.

6. Online Cursor Filter

When using the gaze position to control a cursor in a desktop environment, it is important that the cursor follows the eyes in a way that feels natural to the user. Due to measurement noise, a filter is required to make the cursor stable enough to allow the user to aim at an object of interest, such as an icon. The cursor should also respond promptly to quick jumps the user might make to another area of interest on the screen. This requires careful filter design, since a simple low-pass filter used to filter out the high-frequency measurement noise makes the response to large changes in gaze position sluggish. This can be very frustrating for the user, since s/he has to wait for the cursor to move to the new area of interest. The filter used in the current product attenuates measurement noise by calculating a moving average of previous position data. Its main drawback is a quite slow rise-time.

6.1 Filters

When measuring a signal containing noise, it is sometimes possible to reduce the noise to an acceptable level by using a filter. In the cases where the frequency content of the noise is separated from the frequency content of the desired signal, the filtering is straightforward.

Variants of simple filters are low-pass, band-pass and high-pass filters. As their names suggest, the low-pass filter is transparent to frequencies below a specified cut-off frequency. The band-pass filter allows frequencies within a specified range to pass through, and the high-pass filter is transparent to frequencies higher than a given cut-off frequency. The cut-off frequency is often defined as the frequency at which the output from the filter falls below -3dB compared to the output in the pass-band [9]. Here, mainly the low-pass filter and its characteristics are discussed since it is used as the basis of the filter design used to stabilize the cursor.

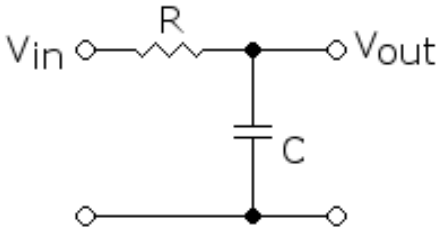

Figure 17. Electronic RC circuit – a simple low-pass filter. When applying a DC voltage to Vin, $V_{out}/V_{in} = 1$. However, the impedance of the capacitor will decrease when the frequency of the signal to V_{in} increases, which causes V_{out}/V_{in} to decrease.

An electronic equivalent of a low-pass filter is shown in figure 17. The cut-off frequency can be set to a desired value by choosing appropriate values of R and C:

$$f_c = \frac{1}{2\pi RC} \tag{10}$$

The frequency response of the filter in figure 17 is shown in figure 18. Here, R and C are chosen such that the cut-off frequency is 1 rad/s.

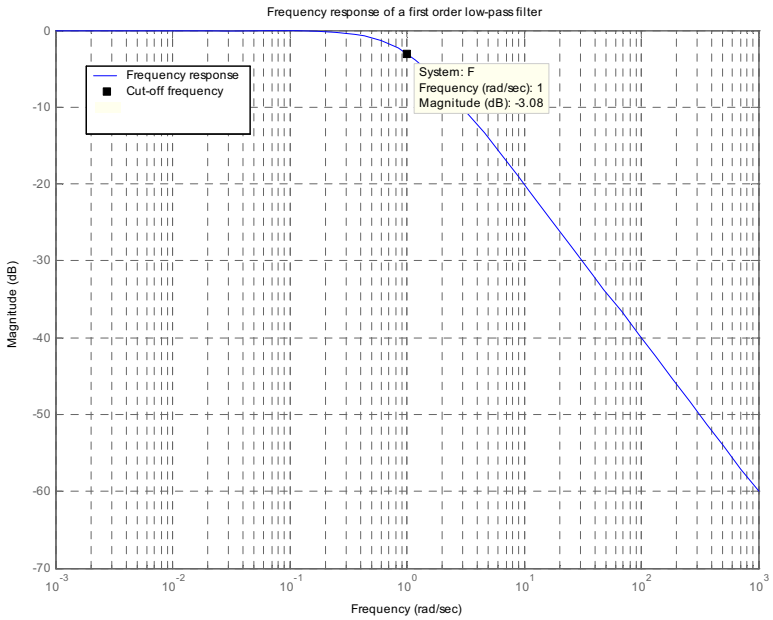

Figure 18. Frequency response of a first order low-pass filter. The cut-off frequency for this filter is 1 rad/s. Here, the amplification drops -3 dB.

Linear time-invariant filters are often described using transfer functions, a mathematical representation of the relation between the input and output of the filter. If x(t) is the input to the filter and y(t) is the output, their corresponding Laplace transforms are:

$$X(s) = \mathcal{L}\{x(t)\} \equiv \int_{-\infty}^{\infty} x(t)e^{-st}\,dt \tag{11}$$

$$Y(s) = \mathcal{L}\{y(t)\} \equiv \int_{-\infty}^{\infty} y(t)e^{-st}\,dt \tag{12}$$

They are related through the transfer function F(s):

$$Y(s) = F(s)X(s) \tag{13}$$

$$F(s) = \frac{Y(s)}{X(s)} \tag{14}$$

The signal X(s) will be scaled and phase-shifted depending on F(s). A totally transparent filter would be F(s) = 1, which simply copies the input signal to the output. A simple first order low-pass filter can be written as:

$$F(s) = \frac{1}{1+Ts} \tag{15}$$

We see that for higher frequencies, $|F(j\omega)|$ decreases due to the larger jω in the denominator causing the filter to attenuate higher frequencies and allow lower frequencies to pass through, which is desired of a low-pass filter.

T in (15) affects the rise time of the filter. The rise time is often defined as the time required for the output signal to rise from 10% to 90% of its new steady state value relative to its previous steady state value when the input signal is a step. However, in (15), T directly corresponds the time required for the output from the filter to reach 63% of the input step signal.

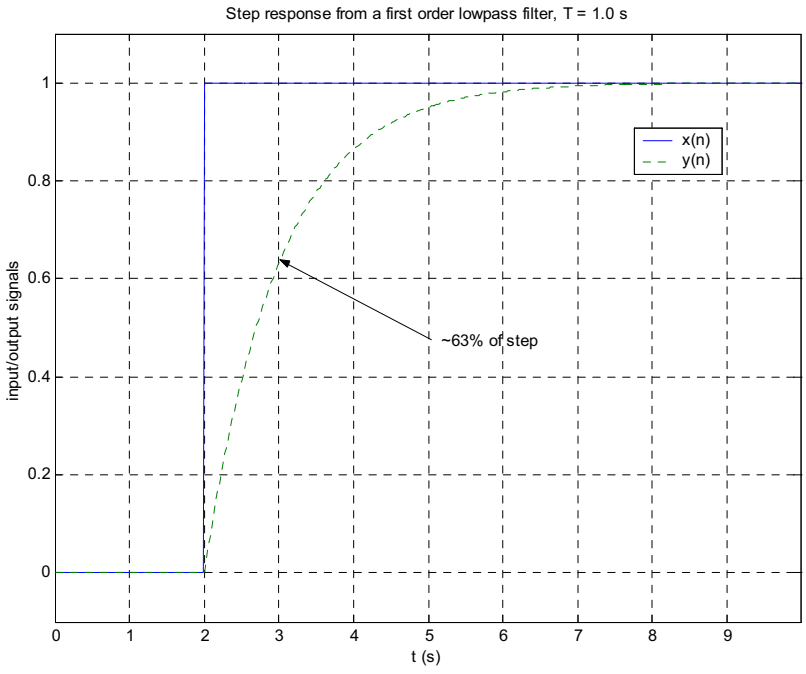

Figure 19. Step response from a first order low-pass filter. T is 1.0 s. After this time, the output from the filter has reached approximately 63% of the step size.

When implementing filters digitally in computers or signal processors, only discrete-time filters can be used. These are often discrete approximations of the continuous-time filters described above. A common way to implement digital filters is the moving average, or finite impulse response (FIR) filter. Here, a finite number of past input data samples are scaled and summed to compose the output signal.

$$y(n) = \sum_{0}^{P} b_i x(n-i) \qquad (16)$$

P is the order of the filter and the number of past data samples used.
b_i are the weights used when composing the output signal.
If $b_0 \ldots b_P = \frac{1}{P+1}$, y(n) is simply the mean of the data samples x(n) … x(n-P).
This method was previously used in the product to stabilize the cursor. If P is chosen large enough, the filter will indeed stabilize the cursor to an acceptable stability level. The filter suffers, however from a slow step response which takes the shape of a ramp. The ramp-shaped response can also induce a feeling of a somewhat stiff and unnatural interaction with the cursor.

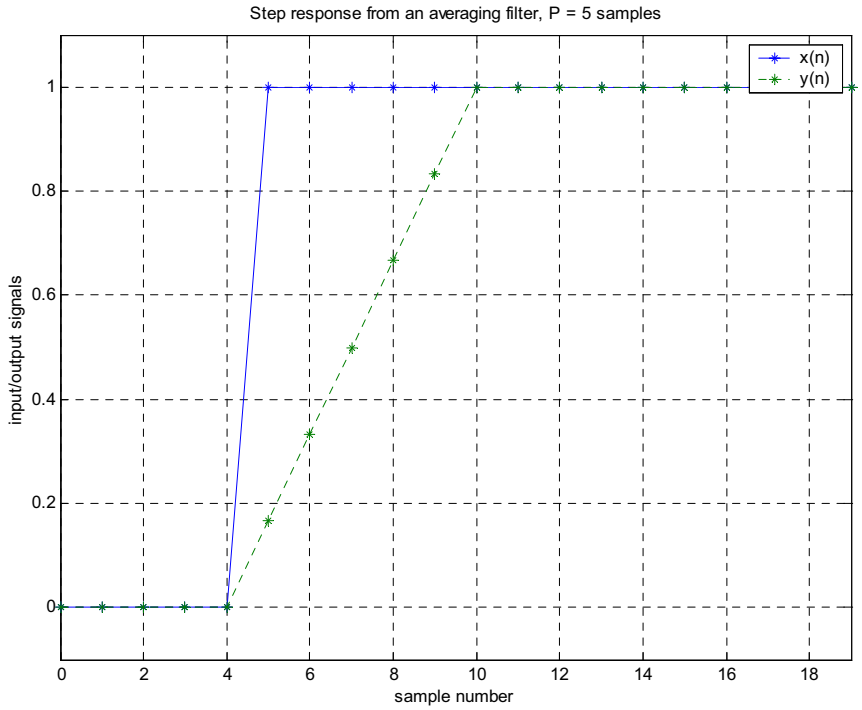

Figure 20. Step response from an averaging filter. Here, a sliding window of 5 samples is used to create the low-pass effect. The step response of an averaging filter is a ramp.

6.2 Algorithm

A common solution to attenuate measurement noise is to use a low-pass filter, see equation 15. Since the filter was to be implemented in a computer, the Backwards Euler approximation was used to discretize the filter:

$$\tilde{\dot{x}}(t_{n+1}) = \frac{x(t_{n+1}) - x(t_{n+1})}{h} \tag{17}$$

With the sampling interval $h = t_{n+1} - t_n$ we get:

$$s \approx \frac{z-1}{zh} \tag{18}$$

Now, (18) is used to make a discrete approximation of the filter:

$$Fd(z) = \frac{1}{1+T\left(\frac{z-1}{zh}\right)} = \frac{1}{1+T\left(\frac{1}{h}-\frac{z^{-1}}{h}\right)} = \frac{1}{1+\frac{T}{h}-\frac{T}{h}z^{-1}} \tag{19}$$

Here, T is the time constant of the filter and h is the sampling interval. Using (19) the filter can be implemented as a digital recursive filter for a stream of two-dimensional gaze-position data:

$$\begin{aligned} \alpha &= \frac{T}{h} \\ \vec{c}(n) &= \frac{\vec{s}(n) + \alpha\, \vec{c}(n-1)}{1+\alpha} \end{aligned} \tag{20}$$

$\vec{s}(n)$ is the measured gaze-position signal and $\vec{c}(n)$ is the filtered data used to control the cursor. The filter uses previous output data from the filter, hence making it a recursive filter. Since the sampling frequency of the eye tracker is 50 Hz, the sampling interval is 0.02 seconds. By choosing a large enough T the measurement noise can be reduced to an acceptable level. The problem with the slow rise time remains however, as illustrated in figure 21 and 22.

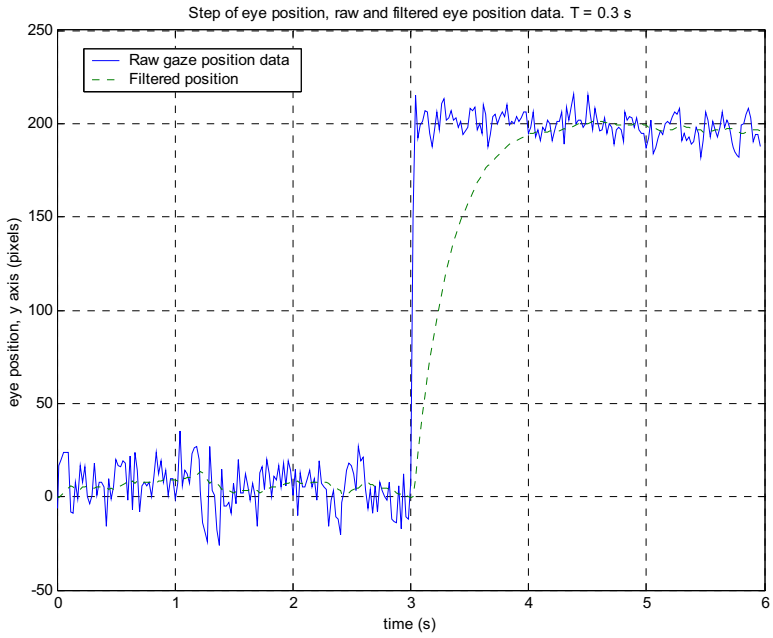

Figure 21. Saccadic eye movement at t = 3 and filtered output, T = 0.3

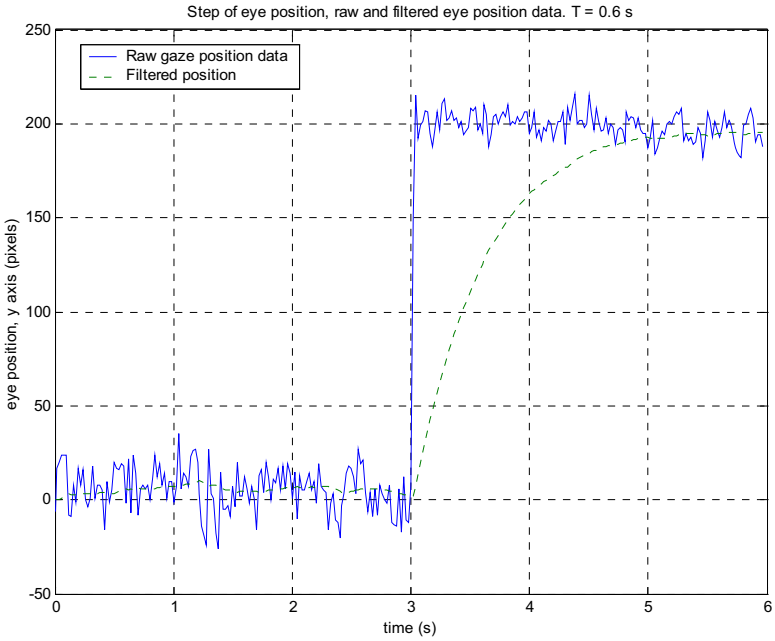

Figure 22. Saccadic eye movement at t = 3 and filtered output, T = 0.6

To solve this problem, a filter that automatically adjusts T depending on the nature of the signal $\vec{s}(n)$ was designed. In the normal case, when the user looks at an object of interest on the screen such as an icon, T is held at a large value T_{slow} to achieve sufficient noise attenuation. When the filter detects a large change in $\vec{s}(n)$, T is temporarily shortened to T_{fast} to make the filter more transparent to large transients.

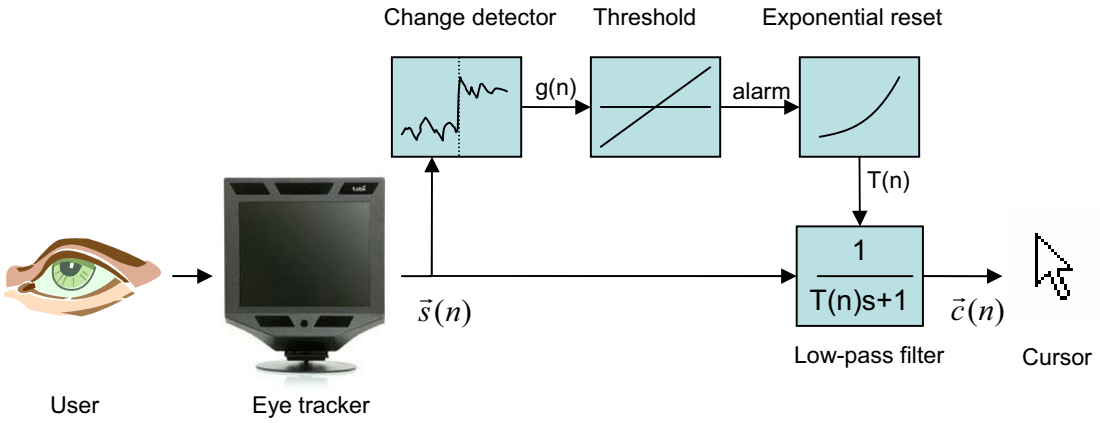

Figure 23. Schematics of filter structure. The low-pass filter works in parallel with a change detector that adjusts the time-constant of the filter when a jump is detected.

If the difference between the means of $\{\vec{s}(n), \vec{s}(n-1), \vec{s}(n-2)\}$ and $\{\vec{s}(n-3), \vec{s}(n-4), \vec{s}(n-5)\}$, g(n), exceeds a threshold, an alarm is given and T is set to T_{fast}. The averaging is used to make it possible to set the threshold to a lower value, thus making the detector more sensitive without increasing the number of false alarms due to measurement noise. Since the detector considers the difference between two means separated at sample n – 2, we will get a delay of two samples. At 50 samples per second, this causes a delay of 40 ms, which is virtually imperceptible by the user. After the transient, T is restored to its original value T_{slow}, following an exponential trajectory to make the transition as imperceptible as possible to the user. It is desired to increase T slowly in the beginning of the reset procedure in order to allow the cursor to quickly move to its new location. However, once the cursor gets close to its new location, T should increase quickly in order to attenuate the measurement noise before it is noticeable by the user. Therefore an exponential reset trajectory is used rather than a linear. It was tuned manually to find a trajectory allowing the cursor to move quickly while making the measurement noise as invisible as possible during and after the transient, see figure 24 and 25. The value of T_{slow} has to be large enough to achieve sufficient noise attenuation, but if it is too large the cursor will move too slowly in the case of a saccade too short for the change detector to detect. A value of 1500 ms was found to sufficiently attenuate the noise while allowing the user to make small corrective saccades without waiting too long for the cursor to follow. Theoretically T_{fast} could be set to zero, which would cause Fd(z) to temporarily equal 1 when a saccadic motion is found and the cursor to instantly follow the raw gaze position. However, a fully transparent filter is not desired since the transparency would make the measurement noise temporarily visible to the user which could be distracting. To avoid this, T_{fast} is set to a small, positive and non-zero value; 50 ms was found to work well. To invoke a click, it is possible to use a so-called dwell time that activates a click if the user fixates on the icon or button of interested at least a specified amount of time. Another alternative is to use a tube connected to a pressure-switch, allowing the user to click on items by blowing air through the tube.

6.3 Results

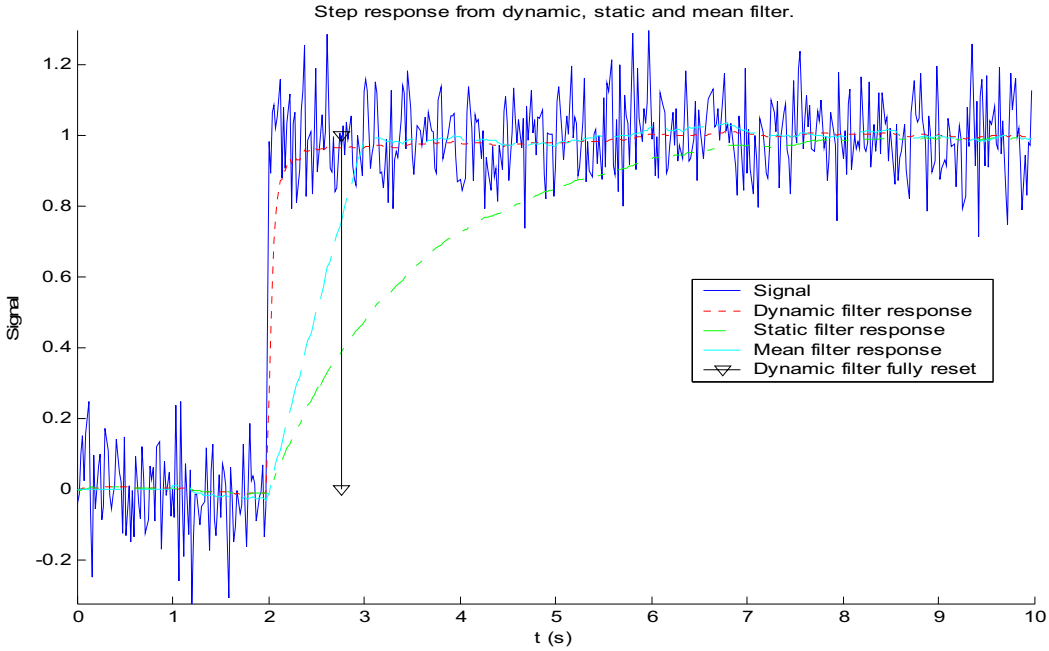

Figure 24. Step responses of dynamic filter with T set to 0.05 when the step is detected at t = 2.0 s. T is then reset along the exponential trajectory shown in figure 25. As a comparison, the step responses of a static low-pass filter with a fixed T = 1.5 and an averaging filter with a sliding window of 50 samples are shown. The rise time of the dynamic filter is much faster, and its noise attenuation is equal to the static filter when T is reset to 1.5. This time instant is marked with a vertical bar in the graph.

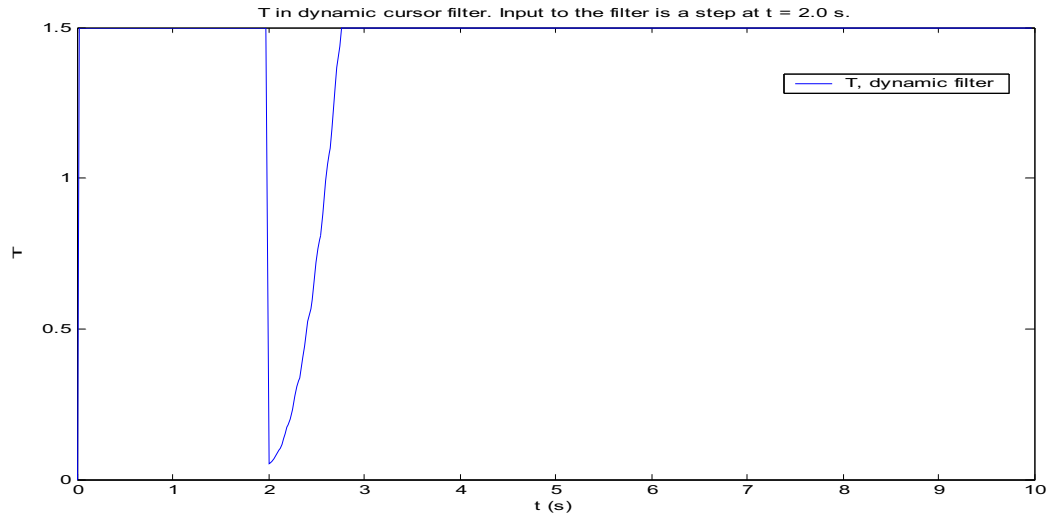

Figure 25. Resetting curve for T in dynamic filter. The acceleration of T is 0.002/sample^2 at 50 Hz.

6.4 Conclusions

By using a dynamic filter together with a change-detector it is possible to achieve much faster rise-times than with a classic filter with a fixed time constant. In the ideal case of a change detector that does not produce false alarms the noise attenuation is equal to that of a fixed filter when the input signal is stationary. During the transients we use the fact that the noise is virtually imperceptible to the subject during the saccadic eye motions.

7. Conclusions and Future Work

Clearly, digital filters can greatly contribute to the performance of modern eye trackers based on corneal reflection. Carefully designed filters may be used to achieve acceptable precision in eye tracking systems based on cheaper components for home use, for example in personal laptops. This would greatly expand the market to include the normal everyday computer user, and potentially make eye tracking as a form of human computer interaction as natural as the keyboard or the mouse. To reach this goal, focus has to be on designing interfaces that fully supports the eye tracking functionality. Future work consists of investigating the possibilities of detecting other high level eye patterns besides reading, such as searching or even confusion and frustration. This would be useful in usability studies, or to make “intelligent” interfaces that (optionally) try to help the user if confusion is detected. Both the post processing fixation filter and the online eye cursor filter could be extended with an adaptive mechanism that automatically adjusts the thresholds to match the current noise variance. One way to achieve this would be to include a noise estimation step in the calibration procedure. Several different eye cursor filters would allow the user to choose the most suitable filter for the task at hand. For example, a very slow filter could be useful when working with graphical painting tools where extra precision and ability to draw smooth curves is needed.

8. References

[1] St. Luke's Cataract and Eye institute, http://www.stlukeseye.com/ (31 Jan 2007)

[2] Jacob R.J.K. (1995) "Eye Tracking in Advanced Interface Design," in Virtual Environments and Advanced Interface Design, ed. by W. Barfield and T.A. Furness, Oxford University Press, New York

[3] Duchowski A (2003) Eye Tracking Methodology: Theory and Practice, Springer

[4] Young LR, Sheena D (1975): Eye movement measurement techniques. Amer. Physiologist 30: 315-30.

[5] Jacob R.J.K. (1991) "The Use of Eye Movements in Human-Computer Interaction Techniques: What You Look At is What You Get," ACM Transactions on Information Systems, vol. 9, no. 3, pp. 152-169

[6] Gustafsson F (2001) Adaptive Filtering and Change Detection, John Wiley & Sons, Ltd

[7] Rayner K, Fischer MH (1996) "Mindless reading revisited: eye movements during reading and scanning are different." in Perception and Psychophysics 1996 Jul; 58(5):734-47.

[8] Campbell C, Maglio P (2001) A Robust Algorithm for Reading Detection Proceedings of the 2001 workshop on Perceptive user interfaces, ACM Press, Orlando, Florida

[9] Answer Corporation, http://www.answers.com/topic/low-pass-filter (31 Jan 2007)

9. Appendix

Description of the fixation filter algorithm using pseudo code and stimuli used for evaluation of the reading detection filter.

9.1 Pseudo code for Fixation Filter.

The ‘rawdata’ vector contains the whole sequence of sampled raw gaze position data. It contains the following fields: integer positions ‘x’ and ‘y’ and boolean flag ‘exists’ to flag if the current sample was captured by the tracker, or if it was lost for example during a blink, or if the subject moved too far away from the camera.

Step 1, Interpolate missing data

Go through the entire raw data vector and make a 0^{th} order interpolation for missing data points by storing the last known position and copy it to the missing samples.

```
for n = 0 to sizeof(rawdata)
{
        if rawdata(n).exists
          {
                last_known_position = rawdata(n)
        }
        else
        {
                rawdata(n) = last_known_position
        }
}
```

Step 2, Calculate the difference vector

Go through the interpolated two-dimensional raw data vector and create a new one-dimensional vector $d(n) = \sqrt{(\vec{m}_{after}(n) - \vec{m}_{before}(n)) \cdot (\vec{m}_{after}(n) - \vec{m}_{before}(n))^T}$ with the differences between the means of the sliding windows m_{before} and m_{after}. Here, r is the length of the sliding windows.

```
for n = r to sizeof(rawdata) - r-1
{
        m_before = m_after = (0, 0)

        for i = 1 to r
          {
                m_before.x = m_before.x + rawdata(n – r).x / r
                m_before.y = m_before.y + rawdata(n – r).y / r
                m_after.x = m_after.x + rawdata(n + r).x / r
                m_after.y = m_after.y + rawdata(n + r).y / r
        }
        d(n) = sqrt( (m_after.x – m_before.x) ^ 2 + (m_after.y – m_before.y) ^ 2)
}
```

Step 3, Find peaks in difference vector

All peaks are first detected in d, by finding values higher than both the preceding and following sample. The peaks are stored in a separate vector, ‘peak’ which is zeroed in the beginning of this step.

```
peak(0, 1…sizeof(rawdata) -1) = 0

for n = 1 to sizeof(rawdata) – 2
{
        if d(n) > d(n – 1) and d(n) > d(n + 1)
        {
                peak(n) = d(n)
        }
}
```

Step 4, Remove peaks that are too close to each other in the time domain.

If more than one peak is found within a window of r samples, the highest peak is kept.

```
for n = r to sizeof(rawdata) – r-1
{
        if peak(n) != 0.0
        {
                for i = n – r to n – 1
                    {
                       if peak(i) < peak(n) then peak(i) = 0.0
                    }
                    for i = n + 1 to n + r
                    {
                       if peak(i) < peak(n) then peak(i) = 0.0
                    }

          }
}
```

Step 5, Create a list with peak indices.

Add the indices of the peaks taller than the given threshold in the peak vector to a list; *PeakIndices*.

```
for n = 0 to sizeof(peaks) – 1
{
        if n >= threshold then PeakIndices.add(n)
}
```

Step 6, Estimate spatial position of fixations

Estimate the positions of the fixations using the median and join fixations that are spatially closer together than the given threshold radius. This is done iteratively beginning with the fixations with the shortest intermediate distance. Two lists are used: *PeakIndices* containing the indices of the previously calculated peaks and *Fixations* which is a list containing two dimensional estimates of the fixation positions.

```
while shortestdistance < radius
{

        /* Clear Fixation list and calculate new estimates for each iteration. */

        Fixations.Clear

        for n = 1 to sizeof(PeakIndices) – 1
```

```
    {
        /* Add median estimate to the Fixation list. Use all rawdata samples
        between PeakIndices(n – 1) and PeakIndices(n) when calculating the
        estimate. */

        Fixations.Add( median ( rawdata ( PeakIndices(n – 1), PeakIndices(n – 1)
        + 1, …, PeakIndices(n) – 1, PeakIndices(n) ) )

     }

     /* Set shortestdistance to Inf. If this is not done, it will remember the shortest
     distance from the previous iteration and no new shortest distance between
     fixation estimates will be found. */

    shortestdistance = Inf

    for n = 1 to sizeof(Fixations) – 1
    {
        /* Calculate the distance as the Euclidean norm between the current and
        previous fixation estimate. If it is shorter than the current shortest
        distance, mark it as the shortest and remember its index. */

        distance = Euclidean_norm(Fixations(n) – Fixations(n – 1))

        if distance < shortestdistance
        {
            shortestdistance = distance
            index = n
        }
    }
     /* If the shortest intermediate distance between two fixation estimates is shorter
     than the threshold radius, remove the peak separating them. During the next
     iteration, they will be joined together. */

     if shortestdistance < radius
    {
        PeaksIndices.Remove(index)
     }

}
```

9.2 Stimuli used for evaluation of the Reading Filter

Stimulus 1

Stimulus 2

Hawaii was first inhabited in roughly AD 1000, by Polynesian settlers who came from islands in the South Pacific, most likely the Marquesas. For nearly 800 years, the people of Hawaii lived in a complex caste society governed by various warring chiefdoms and an extensive system of religious and social taboos called the kapu system. British explorer James Cook chanced upon the Hawaiian archipelago in 1778 in what is commonly assumed to be the first European contact with Hawaiians; however, substantial evidence (Stokes 1932 for example) exists of earlier Spaniard visits to Hawaii.

Stimulus 3

Stimulus 4

Stockholm

Stockholm the capital and the largest city of Sweden is built on 14 islands and has many waterways passing thru it. Because you are never far from the water the city is often called "The Venice of the North".

The city is protected from the open seas by "Skärgården", an archipelago consisting of 24,000 islands, islets, and skerries. Stockholm is very environmentally friendly, the air is significantly cleaner then in many other city's and during the summer it is actually possible to swim and fish in the middle of the town.

The Old Town "Gamla Stan" located in the center of the city is the island where Stockholm began in the 13th century. It is a district of winding narrow alleys and old buildings which includes the Royal Palace. Today the city have grown to a population of over 1 million inhabitants and every year on December 10 the Nobel Prizes are awarded here.

Stimulus 5

Stimulus 6

Eye tracking is a technique used in areas such as cognitive science, psychology (notably psycholinguistics), human-computer interaction (HCI), marketing research and medical research. The most widely used current designs are video-based eye trackers. A camera focuses on one or both eyes and records their movement as the viewer looks at some kind of stimulus. Most modern eye-trackers use contrast to locate the center of the pupil and use infrared and near-infrared non-collumnated light to create a corneal reflection (CR). The vector between these two features can be used to compute gaz intersection with a surface after a simple calibration for an individual.

Stimulus 7

Stimulus 8

The human body is composed primarily of water, and thus has a very similar density to water. Roughly, 70% of the body is water; while the lungs are filled with the air, the body is slightly less dense than the surrounding water, which exerts a buoyant force on it. Thus staying afloat requires only a slight propelling of water downward relative to the body, and transverse motion only a slight propelling of water in a direction opposite to the direction of intended motion. This propelling is accomplished by using the hands and forearms as paddles, and by kicking the legs to push water away from the body (though kicking accounts for relatively little overall). Since salt water (e.g., the ocean) is denser than fresh water (e.g., most swimming pools), less effort is required to stay afloat in salt water than in fresh water.

Stimulus 9

Stimulus 10

Bearnaise Sauce

5 tbspns snipped tarragon
3 tbspns wine vinegar preferabley red
1 tsp white peppercorns coarsely crushed
4 egg yolks
250g butter
3 tbspns snipped chervil
salt

Peel wash and finely chop the shallots.
Put the tarragon shallots vinegar and crushed peppercorns into a shallow pan.
Set over high heat and reduce the mixture by half.
Set aside in a cool place.
When the reduction is cold add the egg yolks and proceed exactly as for Hollandaise sauce (qv) but heat the egg yolks at a fractionafly higher temperature (65C/150F).
Add the chervil just a few minutes before serving so that its flavour and colour are at their best.
Do not strain the sauce but serve it just as it is; we much prefer the coarser texture.