Kalman Filtering in the Design of Eye-Gaze-Guided Computer Interfaces

Oleg V. Komogortsev and Javed I. Khan

Perceptual Engineering Laboratory, Department of Computer Science, Kent State University, Kent, OH, USA 44242
okomogor@cs.kent.edu, javed@kent.edu

Abstract. In this paper, we design an Attention Focus Kalman Filter (AFKF) - a framework that offers interaction capabilities by constructing an eye-movement language, provides real-time perceptual compression through Human Visual System (HVS) modeling, and improves system's reliability. These goals are achieved by an AFKF through identification of basic eye-movement types in real-time, the prediction of a user's perceptual attention focus, and the use of the eye's visual sensitivity function and eye-position data signal de-noising.

Keywords: Human Visual System Modeling, Kalman Filter, Human Computer Interaction, Perceptual Compression.

1 Introduction

There is a significant amount of research being conducted in the area of eye-tracking. An eye-tracker is a web-camera-like device that provides information about the location of the user's eye-gaze. Eye-gaze can be used as a direct input to a computer. Eye-tracking technology has recently become non-intrusive and more appealing to the user.

There are many advantages of eye-based input. Eye-movements are essentially faster than the current popular methods of input, plus each person naturally looks at the desired destination before any other action is taken [2]. In addition, the control-to-display relationship for an eye-tracking device is already established in our brain [1]. For motion-impaired people an eye-tracker is extensively used as an input device to a computer. Eye tracking data provides the means for perceptual compression as well. Perceptual compression uses properties of the Human Visual System (HVS) to provide content compression that is imperceptible to the viewer. The perceptual compression can be applied as bandwidth reduction of the video stream or the computational burden reduction for the 3D-based content. Interactive and content compression components make eye-tracking extremely attractive in the design of future interfaces and in the production and distribution of multimedia streams.

In this paper we propose an Attention Focus Kalman Filter (AFKF) framework that merges together the paradigms of interaction, compression and noise removal.

2 Previous Work

Jakob [1] has proposed an eye-gaze-based interaction interface. The activation of the interface components was done through eye-fixation detection. Different interface actions were performed by specific eye-fixation durations. A computer application was designed to test the proposed interface. It was reported that the interface accuracy was comparable to the touch screen displays rather than a mouse, but the system provided an impression of responding to the user's intention rather than explicit input.

Perceptual compression was investigated previously for still images by Tsumura et. al. [9], Stelmach and Tam [10]; for real-time video by Komogortsev and Khan [4,5,6,7]; for 3D content by Murphy and Duchowsky [11].

Eye-movement detection by the Kalman Filter was proposed by Sauter et. al. [8]. Sauter used the innovations generated by a Kalman Filter to identify the eye-movements called saccades. Grindiger [5] used a Kalman Filter to identify different eye-movement types following the idea proposed by the Sauter. Gridinger's implementation used a mouse-generated signal as a testbed for saccade detection. The saccade detection parameters proposed by Gridinger allowed the construction of a well-behaved saccade detection filter.

3 Human Visual System

3.1 Basic Eye-Movements

There are three major types of eye-movements that are exhibited by the Human Visual System during the perception of multimedia and interaction with the interface components.

1. Fixation: - "eye movement which stabilizes the retina over a stationary object of interest" [4]. Eye-fixations are accompanied by drift, small involuntary saccades and tremor. A human's eye perceives the highest quality picture during an eye-fixation.
2. Saccades: - "rapid eye movements used in repositioning the fovea to a new location in the visual environment" [4]. Usually saccades transition the HVS from one eye-fixation to another. The HVS is blind during a saccade .
3. Smooth pursuit: - eye movement that develops when the eyes are tracking a moving visual target. The quality of vision varies during smooth pursuit eye-movements.

3.2 Human Visual System Sensitivity

A visual sensitivity function allows us to perform multimedial compression through the knowledge of current eye-gaze position. The formula was discussed in our previous work in [7].

4 Kalman Filtering

The Kalman filter is a recursive estimator that is used for computing a future estimate of the dynamic system state from a series of incomplete and noisy measurements. A

Kalman Filter minimizes the mean of the squared estimate error, between the prediction of the system's state and the measurement. Only the estimated state from the previous time step and the new measurements are needed to compute the new estimate for the current dynamic system's state.

A Kalman filter works with a dynamic system that is modeled by an n-by-1 state vector that is updated through this discrete time equation:

$$x_{k+1} = Ax_k + Bu_k + w_k \tag{4.1}$$

In the equation above, **A** is an n-by-n state transition matrix, **B** is an n-by-m optional control input matrix, which relates control vector m-by-1 $\mathbf{u_k}$ to the dynamic system's state $\mathbf{x_k}$. **w** is an n-by-1 process noise vector with covariance $\mathbf{Q_k}$.

We should note that here lower-case, bold letters denote vectors, and upper-case, bold letters denote matrices.

Every dynamic system's state has a j-by-1 observation/measurement vector:

$$z_k = Hx_k + v_k \tag{4.2}$$

H is a j-by-n observation model matrix which maps the true state into the observed space. $\mathbf{v_k}$ is an observation noise j-by-1 vector with covariance $\mathbf{R_k}$.

The Discrete Kalman filter has two distinct phases which are used to compute the next dynamic system state's estimate.

Predict
project the state vector ahead:

$$\hat{x}_{k+1}^{-} = A\hat{x}_k + Bu_{k+1} \tag{4.3}$$

project the error covariance matrix ahead:

$$P_{k+1}^{-} = AP_kA^T + Q_k \tag{4.4}$$

The predict phase uses the state estimate information from the past time step to produce an estimate for the future state.

Update
compute the Kalman gain:

$$K_{k+1} = P_{k+1}^{-}H^T(HP_{k+1}^{-}H^T + R_k)^{-1} \tag{4.5}$$

update the estimate of the state vector with a measurement z_k:

$$\hat{x}_{k+1} = \hat{x}_{k+1}^{-} + K_{k+1}(z_{k+1} - H\hat{x}_{k+1}^{-}) \tag{4.6}$$

update the error covariance matrix:

$$P_{k+1} = (I - K_{k+1}H)P_{k+1}^{-} \tag{4.7}$$

Once the future system's state becomes current, the new measurement information is used to refine the predictions made in the predict phase, which allows the Kalman Filter to come to the more precise dynamic system's state estimate.

5 Attention Focus Kalman Filter Design

We model HVS as a system that has two state vectors $x_k = \begin{bmatrix} \theta_x(k) \\ \dot{\theta}_x(k) \end{bmatrix}$ and $y_k = \begin{bmatrix} \theta_y(k) \\ \dot{\theta}_y(k) \end{bmatrix}$. $\theta_x(k)$ represents the horizontal and $\theta_y(k)$ represents vertical eye position on the screen. $\dot{\theta}_x(k)$, $\dot{\theta}_x(k)$ represent the horizontal and vertical eye-velocity, respectively, at the time k. The state transition matrix is $A_k = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}$.

where Δt is the system's eye-gaze sampling interval. The observation matrix for both state vectors is: $H_k = [1 \quad 0]$

The standard deviation for the instrument noise relates to the accuracy of the eye-tracker equipment and is bounded by one degree of the visual angle, thus making the standard deviation of measurement noise $R_k = \delta_v^2 = 1°$.

In the scenario when the eye-position signal is corrupted $R_k = \delta_v^2 = 1000°$ The standard deviation of the process noise, in our case the noise inside of the eye, has to do with three eye-sub-movements during an eye-fixation: drift, small involuntary saccades and tremor. Among those three, involuntary saccades have the highest amplitude - around half of the visual angle. We create an upper boundary of 1° as a system's noise estimation and use the following covariance matrix for the system's noise process: $Q_k = \begin{bmatrix} \delta_w^2 & 0 \\ 0 & \delta_w^2 \end{bmatrix}°$ where $\delta_w^2 = 1°$ represents the variance of the HVS noise.

6 Designing Interaction with AFKF

Eye Movement Detection by AFKF

Saccade detection was performed through the method proposed by Sauter [8]. Chi square test monitors the difference between predicted and observed eye-velocity:

$$\chi^2 = \sum_{i=1}^{p} \frac{(\dot{\theta}_i^- - \dot{\theta}_i)^2}{\delta^2} \tag{6.1}$$

where $\dot{\theta}_i^-$ is the predicted eye-velocity computed with Equation 4.3 and $\dot{\theta}_i$ is the observed eye-velocity. δ is the standard deviation of the measured eye-velocity during the sampling interval under consideration. Once a certain threshold of the χ^2 is achieved a saccade is detected (value of 150 is used in our system). It was reported by Giringer that the filter behaves better if the standard deviation δ is a constant. Our experiments use values $\delta^2 = 1000$ and p=5 proposed by Giringer.

We have developed a function to map the value of χ^2 to the amplitude of the corresponding saccade. The development of such a function is possible due to the fact that HVS uses phasic (fast) eye-muscle fibers with high motoneuronal firing rate for large saccades and tonic (slow) eye-muscle fibers with lower motoneuronal firing rate for the saccades of lesser amplitude [6]. This mechanism ensures different rate of rise of eye-muscle force for the saccades of various amplitudes providing higher acceleration to the eye globe during saccades of high amplitude. We have derived the function matching saccade amplitude to the value by empirical testing. Let A_{sac_deg} represent the amplitude of a saccade measured in degrees, then:

$$A_{sac\_amp} = -0.000024\chi^6 + 0.0536\chi^4 + 1.5 \quad (6.2)$$

Once the amplitude of the saccade is determined the duration of the saccade is calculated through the equation developed by Carpenter [12].

$$D_{sac\_dur} = (2.2A_{sac\_amp} + 21)/1000 \quad (6.3)$$

D_{sac_dur} is saccade duration measured in seconds.
Eye-fixation detection analysis is performed on updated through the AFKF eye-positions (Equation 4.6). If the eye velocity threshold did not exceed 0.5 deg/sec for a specified period of time (minimum 100 msec.), then an eye-fixation was detected.
Smooth pursuit was detected when the eye-position sample was not a part of an eye-fixation or a saccade and the eye-velocity did not exceed 140 deg/sec (the termination condition for smooth pursuit).

6.2 Eye-Movement-Based Interaction Language

Eye-movement language can be created by identifying basic eye-movement types and using them as tokens for more complex language structures. The spatial and temporal information for each language token combined with content and context information has to be considered before each sentence is evaluated into a specific command. The challenge in the eye-movement language design is due to the fact that humans usually do not use their eye-movements to control the environment. The HVS consists of eye-movements that can be intentionally controlled and ones that cannot. As an example, an eye-fixation is a voluntary eye movement - each of us can look at a point of interest and examine it at will. The length of this examination can be varied voluntarily as well. Saccadic eye movements can be voluntary and can be involuntary. If we move our eyes from one point of the screen to another point, the saccadic eye movements involved would be voluntary. If we are looking at the screen and something catches our attention in the periphery, then the HVS moves to the new target with involuntary saccades. Smooth pursuit eye-movements are involuntary. The interaction approach proposed by Jacobs [1] is to use eye-fixations of different durations to manipulate various interface components. In the implementation of our system, we use eye-fixation tokens with a length of 500 msec. to select the interface components. The smooth pursuit tokens were used to center the object of interest on the screen.

7 Perceptual Compression with AFKF

Feedback Loop Delay
The feedback loop delay T_d is the period of time between the instant the eye position is detected by an eye-tracker and the moment when a perceptually compressed image is displayed. The delay existence disrupts the eye-gaze-based systems due to uncertainty that it brings [7].

Perceptual Attention Focus Window
Our system compensates for the feedback loop delay by constructing a Perceptual Attention Focus Window (W^{PAW}) [7]. Figure 1 present W^{PAW} diagram.

Enhanced Perceptual Attention Window
This paper introduces Enhanced Perceptual Attention Window (W^{EPAW}) that has a better performance than W^{PAW}.

The most significant shifts in eye-position happen during saccades. Once a saccade is detected through a chi-square test, the saccade amplitude and duration are calculated using Equations 6.2 and 6.3. The eye-movement trajectory during a saccade is predicted by Robinson's model [13].

$$\theta_{sac}(t) = \theta(t) + A_{sac\_amp}(20t + 0.24e^{-83t} - 0.24) \tag{6.1}$$

where A_{sac_amp} is the amplitude of the saccade calculated from Equation 6.2, t=0 is the onset of the saccade, $\theta(t)$ is eye-position at the beginning of the saccade. At the end of the saccadic eye-movement Human Visual System takes at least 200 msec. or longer to calculate the next saccade target [12]. This assumption allows us to place W^{EPAW} center during the saccade and for additional 200 msec. after the saccade at the coordinates provided by Robinson's model. In all other cases the W^{EPAW} center coordinates are calculated through the prediction done by AFKF:

$$x_{EPAW\_center}(k) = \dot{\theta}_x^-(k - T_d) \quad y_{EPAW\_center}(k) = \dot{\theta}_y^-(k - T_d) \tag{6.2}$$

The future predicted eye-speed is calculated as:

$$V_{EPAW\_FPES\_x}(k) = \sum_{i=m}^{k} \frac{\sqrt{(x_{EPAW\_center}(i - T_d) - \theta_x(i - T_d))^2}}{k - m} \tag{6.3}$$

where $\theta_x(i - T_d)$ is the observed eye-gaze position by the eye tracker.

W^{EPAW} model transforms visual sensitivity function discussed earlier into more complex function which was presented in our previous work [7].

8 Experiment Setup

Equipment
The proposed system was implemented with Applied Science Laboratories eye-tracker model 504. The system was tested with a 24 inch flat screen monitor and a Core Duo E6600 powered computer with 2GB of RAM.

Interaction
The interaction capabilities of the AFKF where tested by playing the World of Warcraft video game. World of Warcraft is a massively multiplayer online role-playing game with a dynamic virtual 3D environment. A player creates an in game avatar that he or she controls. The avatar interacts with the environment by moving around in a virtual world, selecting different objects, and trading or destroying those objects. The mouse cursor in the game was controlled by the AFKF-generated eye-movement language tokens described in Section 6. The project was implemented

under the *World of Warcraft Percept Interface* name using the Microsoft Foundation Class Library (MFC); more details are available at [3].

Perceptual Compression

The system's perceptual compression capabilities were tested with the following MPEG-2 video clip:

Shamu: This video captures an evening performance of Shamu at a Sea World, during night time under a tracking spotlight. The video consists of several moving objects: Shamu, the trainer, and the crowd. Each object is moving at different speeds during various periods of time. The background of the video was constantly moving due to the fact that the camera was trying to follow Shamu.

Participants

The experiments were conducted with one male subject with normal vision.

9 Results

Signal Noise Removal

Eye-tracker eye-position data was classified as noisy when the eye-tracker failed to report proper eye-position. The failure to identify proper eye-position usually happens due to the subject's jerky head movements, changes in the content's lighting, etc. When the eye-position cannot be properly identified the eye-tracker reports it with negative eye-position values. In this case the AFKF uses the signal noise measurement covariance matrix defined in Section 4.

An example of noise removal by the AFKF is presented in Figure 2.

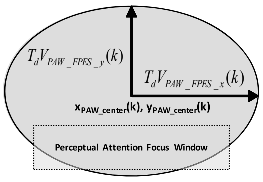

$T_d V_{PAW\_FPES\_y}(k)$
$T_d V_{PAW\_FPES\_x}(k)$
$x_{PAW\_center}(k)$, $y_{PAW\_center}(k)$
Perceptual Attention Focus Window

Fig. 1. Perceptual Attention Focus Window diagram

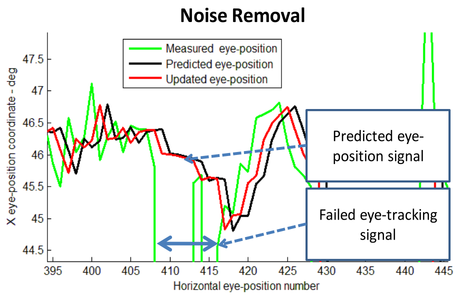


Fig. 2. Eye-tracker signal de-tracker signal de-tracker signal

Interaction

It was possible to interact with the World of Warcraft game using eye-movement language tokens generated by the AFKF and described in Section 6. Selection of an object inside the game is presented by Figure 3. Through eye-movement interaction

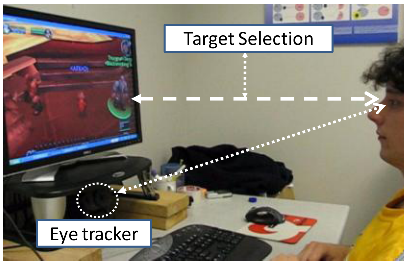


Fig. 3. Target selection byan eye-fixation token generated by the AFKF

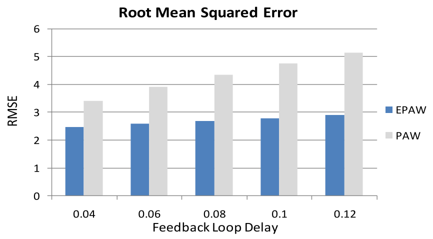


Fig. 4. W^{EPAW} vs. W^{PAW}. Root Mean Squared Error estimation.

the game environment feels more alive: the world talks back when you look at it. The selection of objects is done much faster than through the use of mouse and it feels as the system anticipates the user's intentions.

Several difficulties were encountered during the system's testing as well. 1) The Midas touch problem described by Jacob [1] - no matter what you look at it gets activated. It becomes annoying when you would like to inspect something rather than activate it. This problem can probably be solved by changing the eye-fixation token's activation lengths for different types of menus and objects. 2) It was hard to select very small menu items inside the game due to the fact that the eye-tracker accuracy is around 1°. 3) The slippage in eye calibaration.The recalibration was usually required after 5 minutes in order to restore the accuracy of the eye-tracking. 4) Another problem is that a player would like to do several things simultaneously such as select objects, use different spells or character abilities to interact with these objects (e.g. trade, destroy), and move around. Previously simultaneous actions were possible through the use of a keyboard and mouse, but executing the same sequence of actions with eye-movement language tokens requires more time. This issue can be partially resolved by adding a voice input to the interface. With such multichannel inputs the interface interaction speed increases considerably. More information is available at the project's website [3].

Enhanced Perceptual Attention Focus Window Evaluation

There are three evaluation parameters that we use to validateW^{EPAW} model: the average eye-gaze containment (AEGC), the average perceptual resolution gain (APRG), and the root mean squared error (RMSE) between the W^{EPAW} and the observed eye-position.

The RMSE measures the quality of the constructed window and it is calculated using the following formula:

$$RMSE_{EPAW}(k) = \sum_{i=m}^{k} \frac{\sqrt{(x_{EPAW\_center}(i) - \theta_x(i))^2}}{k-m} \tag{9.1}$$

where $x_{PAW\_center}(i)$ is the center of the W^{PAW} at the time k, $\theta_x(i)$ is the horizontal eye-position at the time i, m is the beginning of the sampling interval (m=T_d in our system's case) and k is the end of the sampling interval (full experiment length). The RMSE shows how close the predicted W^{PAW} center is to the actual eye-position.

Average Eye-Gaze containment

The AEGC is a percentage of the eye-position samples contained within the W^{EPAW}.

$$AEGC(k) = \frac{100}{k-m}\sum_{i=m}^{k} GAZE^{EPAW}(i) \tag{9.2}$$

The variable $GAZE^{EPAW}(i)$ equals one when the i^{th} eye-position is inside of the W^{EPAW}, and it equals zero otherwise. The AEGC includes all of these types of the eye-movements: eye-fixations, saccades, and smooth pursuit. As we have investigated in our previous research, the AEGC provides a more conservative estimation than does the average eye-fixation containment.

Average Perceptual Resolution Gain

The actual amount of bandwidth and computational burden reduction when using the W^{PAW} depends on these two parameters: the size of the area which requires high quality coding (W^{PAW}) and the visual degradation of the periphery. the APRG mathematically estimates the amount of perceptual compression, but the actual implementation numbers may differ.

$$APRG(k) = \frac{H * W * (k-m)}{\sum_{i=m}^{k}\int_0^W\int_0^H S_i(x,y)dxdy} \tag{9.3}$$

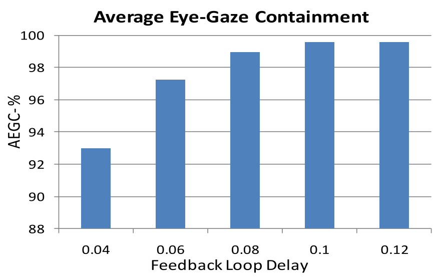
$S_i(x,y)$ – is the eye sensitivity function. One degree of visual angle is added to each dimension of the W^{PAW}. This is done to address the situation when the center of an eye-fixation falls on the boundary of the W^{PAW} - an eye can see approximately one degree of visual angle with highest quality from the center of an eye-fixation. This approach assures that the viewer will not see the degradative effect if the W^{PAW} contains this eye-fixation. W and H are the width and height of the visual image.

Evaluation parameters

Root Mean Squared Error: Figure 4 shows the RMSE values for the W^{PAW} constructed previously in [7] and the Enhanced Perceptual Attention Focus Window designed in Section 7. W^{EPAW} performed much better for all delay scenarios, reducing RMSE by up to 2° per sampling interval.

Average Eye-Gaze Containment: Figure 5 presents the AEGC value achieved by the W^{EPAW} for different delay scenarios. In all cases the AEGC is higher than 90%. For a 0.1 sec. delay and higher the AEGC is close to 100%. This result means that the perceptual compression is going to be completely unnoticed by the AFKF system's user.

Average Perceptual Resolution Gain: Figure 6 presents the APRG value achieved by the W^{EPAW} for different delay scenarios. The highest APRG value of 2.3 was achieved by the lowest feedback loop delay scenario of 40 msec. The size of the W^{EPAW} grows considerably when the delay is increased. The increase in delay decreases the compression factor of perceptual compression. For example, delay values of more than 0.12 sec. produce an APRG close to 1, indicating than no compression is possible. The higher APRG value the higher is bandwidth reduction and computational burden reduction. The exact numbers will depend on how visual sensitivity function is mapped to the specific encoding scheme.



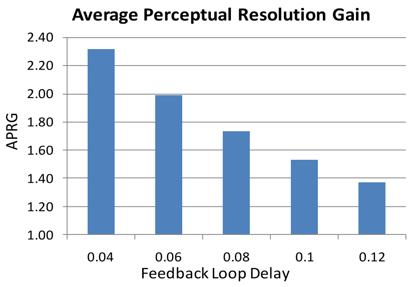


Fig. 5. Average Eye-Gaze Containment achieved by the WEPAW for various feedback delay values

Fig. 6. Average Perceptual Resolution Gain achieved by the WEPAW for various feedback delay values

10 Conclusion

The human computer interaction world is rapidly growing. The community searches for new methods and inputs to provide a more natural, seamless way of communication. Eye-tracking technology can be successfully used to substitute or enhance already existing interaction models and provide more ubiquitous interactive environments.

In this paper we have designed the Attention Focus Kalman Filter framework that removes the noisy signal from the eye-position data stream, creates eye-movement language tokens for interaction, and provides the means for perceptual compression.

The advantage of the proposed framework is the fact that it is equipment- and media- independent. Any eye-tracker vendor can use the AFKF to improve the accuracy of the eye-tracking signal.

The eye-movement language can be applied to any eye-gaze-based interface by adjusting the detection and triggering parameters. We successfully tested the interaction capabilities of the eye-movement language in [3].

The concept of a Perceptual Attention Focus Window can be applied to any visual content by mapping the visual sensitivity function to a specific codec. The potential provided by perceptual compression is high, especially for the scenarios where loop delay values are low.

References

[1] Jacob, R.J.K.: Eye tracking in advanced interface design, Virtual environments and advanced interface design. Oxford University Press, Inc., New York, NY (1995)

[2] Ware, C., Mikaelian, H.T.: An Evaluation of an Eye Tracker as a Device for Computer Input. In: Proc. ACM CHI+GI'87 Human Factors in Computing Systems Conference pp. 183–188 (1987)

[3] Komogortsev, O.: World of Warcraft Percept Interface http://www.cs.kent.edu/ okomogor/ wowpercept/wowpercept.htm

[4] Duchowski, A.T.: Eye Tracking Methodology: Theory and Practic. Springer, London, UK (2003)

[5] Grindinger, T.: Eye Movement Analysis and Prediction with the Kalman Filter, Masters thesis, Computer Science, Clemson University, Clemson, SC, USA (August 2006)

[6] Bahill, A.T.: Development, validation and sensitivity analyses of human eye movement models. CRC Critical Reviews in Bioengineering 4, 311–355 (1980)

[7] Komogortsev, O., Khan, J.: Perceptual Multimedia Compression based on the Predictive Kalman Filter Eye Movement Modeling. In: Proceedings of the Multimedia Computing and Networking Conference (MMCN'07), San Jose, pp. 1–12 (January 28 – February 1, 2007)

[8] Sauter, D., Martin, B.J., Di Renzo, N., Vomscheid, C.: Analysis of eyetracking movements using innovations generated by a Kalman filter. Med. Biol. Eng. Comput 29, 63–69 (1991)

[9] Norimichi, T., Chizuko, E., Hideaki, H., Yoichi, M.: Image compression and decompression based on gazing area. In: Human Vision and Electronic Imagin, SPIE (April 1996)

[10] Stelmach, L.B., Tam, W.J.: Processing image sequences based on eye movements. In: Proc. SPIE 2179, 90–98 (1994)

[11] Murphy, H., Duchowski, A.T.: Gaze-Contigent Level Of Detail Rendering. Eurographics 2001 (2001)

[12] Carpenter, R.H.S.: Movements of the Eyes, pp. 56–57. Pion, London (1977)

[13] Robinson, D.A.: Models of the saccadic eye movement control system. Kybernetic 14, 71 (1973)