Do-It-Yourself Eye Tracker: Low-Cost Pupil-Based Eye Tracker for Computer Graphics Applications

Radosław Mantiuk, Michał Kowalik, Adam Nowosielski, and Bartosz Bazyluk

West Pomeranian University of Technology in Szczecin,
Faculty of Computer Science,
Żołnierska 49, 71-210, Szczecin, Poland
`rmantiuk@wi.zut.edu.pl`,
`http://rmantiuk.strony.wi.ps.pl`

Abstract. Eye tracking technologies offer sophisticated methods for capturing humans' gaze direction but their popularity in multimedia and computer graphics systems is still low. One of the main reasons for this are the high cost of commercial eye trackers that comes to 25,000 euros. Interestingly, this price seems to stem from the costs incurred in research rather than the value of used hardware components. In this work we show that an eye tracker of a satisfactory precision can be built in the budget of 30 euros. In the paper detailed instruction on how to construct a low cost pupil-based eye tracker and utilise open source software to control its behaviour is presented. We test the accuracy of our eye tracker and reveal that its precision is comparable to commercial video-based devices.

Keywords: eye tracking, human computer interfaces, eye tracker accuracy, computer graphics.

1 Introduction

Eye tracking is a technique of gathering real-time data concerning *gaze direction* of human eyes. In particular, position of the point, called *point-of-regard*, that a person is looking at is captured. Interestingly, eye tracking is not popular in imaging and computer visualisation applications despite its undeniable potential. A human is able to see details only by the *fovea* - a part of the eye located in the middle of the macula on the retina. Fovea covers about 2° of the human viewing angle, therefore, information about gaze direction can be very useful in many multimedia applications.

In the last decade much work has been done in the study of using eye tracking as a user interface to multimedia systems [5]. Capturing of the visual attention was considered in the context of supporting multimedia learning [6], understanding web page viewing behaviour [7], and many others [9]. Eye trackers are used in real time graphics systems, e.g. in serious games to support activity rehabilitation [8], to reduce computation time (e.g. render with non-uniform pixel

K. Schoeffmann et al. (Eds.): MMM 2012, LNCS 7131, pp. 115–125, 2012.

distribution [1], simplified scene geometry [2]) or support 3D rendering (e.g. to locate the accommodation plane during depth-of-field rendering [25]). Our goal is to popularise the eye tracking technology in multimedia systems, especially in applications that use the 3D visualisation techniques.

The main issue of the contemporary eye trackers is their high price. A precision below 0.5° of the visual angle (roughly 10 pixels on a 17" display observed from 60 cm distance) is possible to achieve only with the use of very expensive intruisive eye trackers. This type of devices requires that the observer would place her head on the chin rest or use the bite bar that further hinders the practical use of the device. Even less accurate devices with precision of about 1° cost over 20,000 euros [3]. The high cost of the commercial eye trackers seems to stem from the costs incurred in research rather than the price of the hardware components.

In this work we argue that eye tracker with sufficient accuracy can be built in the budget of 30 euros. As a proof of concept, we have designed and built a low-cost head-mounted eye tracker, called *Do-It-Yourself* (DIY) eye tracker. Its construction is based on a web camera and the 3rd party ITU Gaze Tracker software [4]. To estimate accuracy of DIY eye tracker we conduct subjective experiments measuring its precision for a number of observers. As a case of study we test DIY eye tracker in our virtual environment software in which the depth-of-field rendering is controlled by captured gaze direction (see Section 4.5).

The paper starts with a survey of eye tracking techniques (Section 2). In Section 3 we present the DIY eye tracker and describe details of its construction. Section 4 describes the conducted experiment and depicts its results. We conclude the paper in Section 5.

2 Eye Tracking Technologies

Tracking of humans' gaze direction is acquired in numerous ways encompassing intruisive and remote techniques.

Intruisive eye trackers require some equipment to be put in physical contact with the user. In early works a coil embedded into a contact lens was used [10]. The eye gaze was estimated from measuring the voltage induced in the coil by an external electro-magnetic field. In another electro-oculogram technique (EOG) [11] electrodes are placed around the eye. Eye movement is estimated by measuring small differences in the skin potential. In general, intruisive techniques are very accurate and often used in scientific experiments (accuracy reaches 0.08 *deg* of human visual angle) but due to intruisive nature are rather useless in the most computer graphics and imaging applications.

More suitable for vision systems are remote techniques that use cameras to capture image of the eye. Even if they require some intrusive head mounted devices [12, Sect. 6], they are still acceptable for many applications, e.g. for virtual environments and augmented reality.

The most common remote eye trackers apply the *corneal reflection* (CR) method. The eyes are exposed to direct invisible infra-red (IR) light, which results in appearance of Purkinje image with reflection in the cornea (see Fig. 1, left). The reflection is accompanied by image of the pupil. Captured by a video camera sensitive to the infra-red spectrum, the relative movement of both pupil and corneal reflections are measured, which enables the estimation of observer's gaze point. It is reported that commercial eye trackers can achieve the accuracy below 0.5° [3]. The CR eye trackers require calibration to establish a mapping between the reflection-pupil vector and the actual screen-space target point.

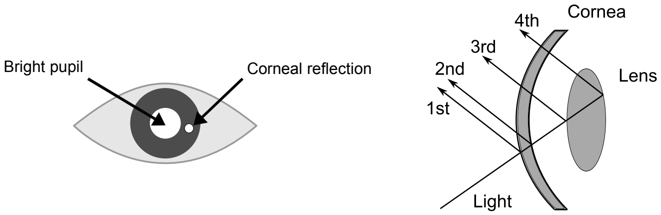

Fig. 1. Left: the corneal reflection in the infra-red light, relative location of the pupil and the corneal reflection are used to estimate observer's gaze point. Right: Purkinje images.

There are eye trackers that simultaneously process more than one corneal reflection. The first Purkinje image (used in CR eye tracking) corresponds to the reflection from the external surface of the cornea. The three remaining images are created by reflections from internal surface of the cornea, and both surfaces of the lens (see Fig. 1, right). In literature various eye tracking systems based on 1st and 4th Purkinje images [13], as well as 3rd and 4th [14] were proposed. The most popular are DPI (dual Purkinje image) eye trackers that estimate gaze point with very high accuracy of about 1 min of arc. Their drawback is the need of using a chin rest and/or a bite bar for head stabilisation [12, Sec. 5.4].

The sufficient eye tracking accuracy can be achieved detecting pupil's centre. In our project, we built the pupil-based eye tracker suitable for many computer graphics tasks including free-walking in virtual environments (if combined with the head tracker system). The detailed description of our eye tracker is presented in Section 3.

A similar low-cost head-mounted eye tracker was constructed by Li et al. [15] (openEyes project). The authors report that accuracy of this CR-based eye tracker is close to 0.6° (for the 4th generation of the device). EyeSecret project (continuation of openEye) presents auto-calibration eye tracker of accuracy about 1° [16]. The mentioned projects were inspired by Pelz et al. [17] work, in which analog camera and mini-DVD camcorder were used to record user's eye. Then, analysis of the video was performed off-line to capture points of regard. In contemporary solutions analog camera and a camcorder can be replaced with a digital camera and wireless data transfer techniques to allow remote connection between an eye tracker and a computer. Another low-cost solution was presented by Augustin et al. in [19]. The authors tested performance of target acquisition and eye typing of the developed webcam-based eye tracker. They assessed the ability of using the eye tracker as a user interface rather than measured its geometric accuracy. Their eye tracker must be held with observer's teeth what seems to be inconvenient for users.

Detailed reviews of eye tracking techniques are presented in [18,20] and [12].

3 Do-It-Yourself Eye Tracker

We designed and constructed a prototype eye tracker called Do-It-Yourself eye tracker (DIY). The main goal of this work was to develop an eye tracker suitable for computer graphics applications. We assumed that this device should base on remote gaze tracking technique, it should be cheap and possible to build with components available at consumer market.

We constructed the eye tracker which can be used for free-walking tasks in virtual environments. However, it would require the head tracker to capture the head position.

The DIY eye tracker operation is based on the detection of centre of the pupil. In our system, the accompanying ITU Gaze Tracker software (see Section 3.2) analyses an infrared image of the eye and locates position of the pupil. Coefficients gathered during the calibration phase are used to compute the gaze position in screen coordinates.

3.1 DIY Hardware

The DIY eye tracker consists of two main components: a modified safety goggles that act as a frame and a capture module attached to the goggles (see Figure 3, right).

The capture module is based on a typical web camera (we used Microsoft Lifecam VX-1000, working in 640x480 pixels resolution). This camera is placed in 5 cm distance from the left eye. The camera should be as small as possible to avoid occluding the observer's field of view. The original VX-1000 camera was modified by removing the chassis and replacing the infrared light blocking filter with the visible light blocking filter. For this purpose we used a fragment of the overexposed analog camera film which filters light in a similar way as infrared filter does, but this solution is much cheaper. In Figure 2 differences between images taken with various filters are presented.

The capture module is equipped with infrared photodiodes to additionally illuminate the eye in the infrared spectrum. Position of photodiodes was carefully chosen to assure correct illumination of the eye and avoid strong corneal reflection which could influence results of the software pupil detection algorithm. We found that three photodiodes (45 mW/sr) spread in the triangle topology around the camera lens give satisfactory results (see Figure 3 left).

The capture module is mounted on the safety goggles. A flexible connection based on aluminium rod allows to adjust position of the camera in relation to the eye. The plastic glass of goggles was removed to avoid image deformation

Fig. 2. Image of the human eye, from left: image captured by a regular web-camera, without the infrared light blocking filter, and with visible light blocking filter (with a fragment of the burned analog camera film). Notice that the dark pupil is very clearly visible in the rightmost image.

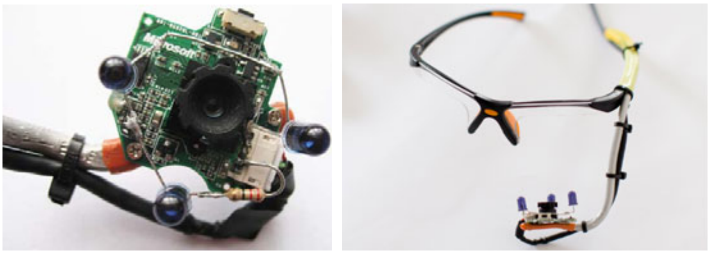

Fig. 3. Left: topology of the photodiodes used to illuminate the eye. Right: DIY eye tracker hardware.

and unwanted reflections. The capture module is connected to computer via the USB cable which acts also as a power source for the camera and photodiodes. Detailed description of the DIY construction is available on the project web site[1].

The total cost of all components needed for building the DIY eye tracker is under 30 euros. The eye tracker can be assembled by a student in a few hours. After installation of a typical USB driver for the camera module, DIY eye tracker is automatically detected by the ITU Gaze Tracker software (see Section 3.2) and there is no need for additional configuration of its software.

3.2 ITU Gaze Tracker Software

We use the ITU Gaze Tracker software [4] to control the communication between a PC computer and the DIY eye tracker and to execute the eye tracking functionalities. The ITU Gaze Tracker software is developed at the IT University of Copenhagen. It is delivered as a C# open source package under the GPLv3 license.

The ITU Gaze Tracker front-end allows to calibrate eye tracker and then computes a current position of the gaze point. The software captures images taken by the DIY camera module. The images are analysed to find the pupil

---

[1] `http://rmantiuk.strony.wi.ps.pl/projects/diy/index.html`

centre. Detection of pupil position is supported by the OpenCV package and the algorithm parameters can be adjusted with the ITU Gaze Tracker interface. Pupil detection implemented in ITU is based on image thresholding and points extraction in the contour between the pupil and iris. The points are then fitted to an ellipse using RANSAC technique [21].

Each eye tracking session starts with the calibration procedure. Observer is asked to watch at the target points that appear in different positions on the screen. The target points are displayed one by one in random order. After calibration, a current gaze position in the screen coordinates is computed and transfer using UDP protocol to an external application.

4 Evaluation of DIY Eye Tracker Accuracy

The main goal of the tests was to measure the accuracy of DIY eye tracker. We present detailed description of the measurement procedure and the way in which the achieved data were analysed.

4.1 Participants

Nine observers with an age from 21 to 27 participated in our experiment with an average of 22.8 years, standard deviation 2.044, all male. Eight participants had normal vision, one of them had corrected vision with contact lenses. We asked each participant to repeat the experiment twice. We have performed 18 measurement sessions in total. No session took longer than 4 minutes for one participant to avoid fatigue. Participants were aware that accuracy of the eye tracker is tested, however they were not informed about the details of the experiment.

4.2 Hardware and Software Setup

Our experimental setup is presented in Figure 4. It consists of DIY eye tracker controlled by the ITU Gaze Tracker software (version 2.0 beta) and PC with 2.8 GHz Intel i7 930 CPU equipped with NVIDIA GeForce 480 GTI 512MB graphics card and 8 GB of RAM (Windows 7 OS). The experiments were run on a 22” Dell E2210 LCD display with the screen dimensions of 47.5x30 cm, and native resolution of 1680x1050 pixels (60Hz). The second monitor was used to control the eye tracker through the ITU Gaze Tracker software. Observers sit in the front of the display in 63 cm distance and were asked to use the chin-rest (adopted from the ophthalmic slit lamp). The illumination in the laboratory was subdued by black curtains to minimise the effect of display glare and to focus observers’ attention on experiment tasks.

We developed a software which implements the validation procedure. This software is responsible for communication with external applications (in our case with ITU Gaze Tracker). It collects eye tracking data using the UDP protocol interface, renders graphics, supports user interactions required during experiment, and stores experiment results. The software was implemented in C++ and as Matlab scripts using Psychtoolbox.

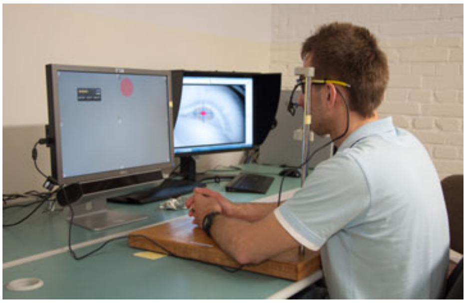

Fig. 4. Hardware setup used for the experiments

4.3 Stimuli and Experimental Procedure

Following [23] recommendation, the experiment started with a training session in which observers could familiarise themselves with the task, interface, chin rest, and how to wear DIY eye tracker. After that session, they could ask questions or start the main experiment.

The experiment started with DIY eye tracker calibration controlled by the ITU Gaze Tracker software. This procedure took about 20 seconds and consisted of observation of the markers displayed in different screen areas. In the next step, the actual validation of eye tracker accuracy was performed. During this procedure controlled by our validation software, participants were asked to look at a set of 25 target points that acted as known and imposed fixation points. As observers used the chin-rest, we knew estimated geometrical position of these points in relation to participants' eyes. The target points were displayed in random order for 2 seconds each. The location of the points on the screen is depicted in Figure 5 (yellow dots).

4.4 Results

Captured positions of gaze points together with positions of corresponding target points were transformed from screen coordinates (pixels) to degrees of the visual angle. We used geometrical dimensions of the hardware setup to compute the transformation, assuming perpendicular view direction at a half of the screen in the horizontal direction and 1/3rd from top in the vertical direction. The gaze direction error angle was computed as a difference between direction towards a target point and towards gaze point captured during observer's fixation on this target point.

The results of the experiment for the individual target points are depicted in Figure 5. Average error for all target points amounts to 1°. Before computation of the average error, we removed 10% of gaze point outliers for every target point. ANOVA analysis did not reveal dependence of the mean results on positions of target points (p=0.0632). However we noticed that the accuracy error is

Table 1. Average error angle in degrees of the visual angle (GP-gaze points)

	DIY eye tracker			RED250 eye tracker		
observer	no. of GP	mean error [°]	std [°]	no. of GP	mean error [°]	std [°]
observer A	5507	0.8125	0.4174	1478	1.2784	0.6363
observer B	5035	1.0866	0.5320	5229	1.2282	0.6177
observer C	3363	1.1619	0.4956	5438	1.0800	0.5968
observer D	5281	1.1492	0.4436	5357	1.2180	0.6147
observer E	5175	1.1365	0.5717	2728	1.4723	0.6794
observer F	4466	1.3932	0.6152	5590	0.9771	0.5242
observer G	4995	1.2167	0.5851	5303	1.2469	0.6350
observer H	5669	0.9424	0.4415	3302	1.4808	0.7697
observer I	5754	0.7510	0.3988	4718	1.2998	0.6247
all observers	45245	1.0557	0.5371	39143	1.2218	0.6425

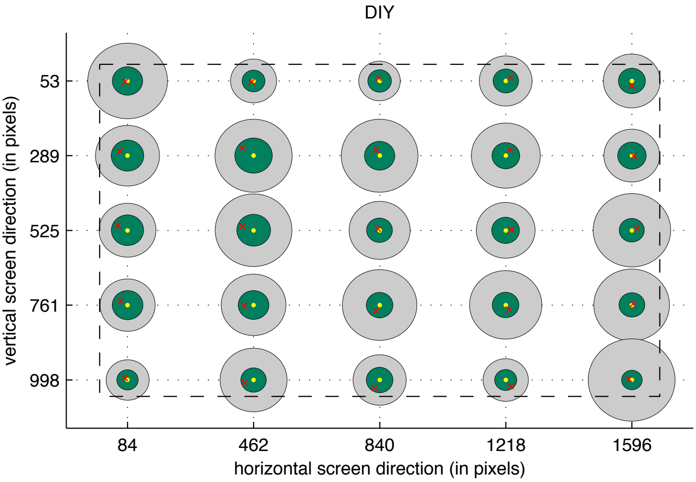

Fig. 5. Gaze direction error measured for 25 target points. The yellow dots denote positions of the target points, the red crosses - positions of median of gaze points (captured individually for every target point), the green circles - median of gaze direction error, and the grey circles - maximum value of gaze direction error (after outliers filtering).

observer dependent. Figure 6 depicts means of the error angle for every observer and their significant statistical difference. In Table 1 number of samples (captured gaze points), average error angles and their standard deviations for every individual observer are presented.

We conducted the same experiment for commercial RED250 eye tracker and achieved average error amounts to about 1.2° which favours our device (see Table 1).

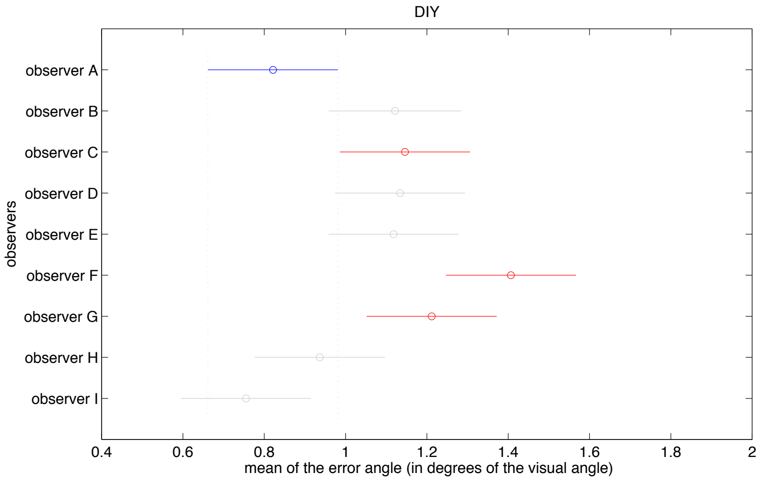

Fig. 6. The average error angle for each observer. The red circle denotes value of the average error angle and the horizontal line is the 95% confidence interval. Note that observers' average gaze directions are significantly different (red lines for observers C,F, and G denote observations significantly different than results for observer A).

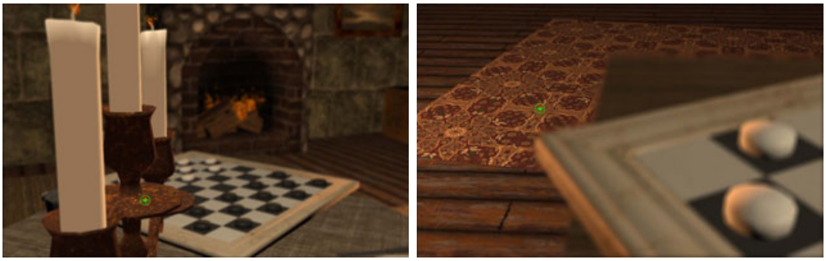

Fig. 7. Example screenshots from the virtual reality renderer

4.5 Application

We tested whether DIY eye tracker can be used to control the depth of field effect rendering in a computer-generated virtual environment. The gaze direction read from the eye tracking system's output can be used to estimate an exact 3D point in the displayed scene. By reading its virtual camera-space depth from the z-buffer, a physical model can be used to calculate the blurriness of different parts of the screen simulating an image viewed through a real optical lens [24]. The user can have a more realistic impression of the scene's depth (see Figure 7).

To measure the actual users' impressions, we have conducted a perceptual experiment (details are presented in [25]). The results show that the gaze-dependent simulation of a depth-of-field phenomenon affects the observer's immersion and has a significant advantage over the non-interactive version of this visual effect. During the experiment we noticed however that the gaze data's accuracy offered by DIY is still inadequate to provide a completely comfortable and realistic simulation comparable with the expected image. The methods for filtering data has to be improved for this use, so as the actual eye tracker's accuracy.

5 Conclusions and Future Work

A price of eye tracking devices inevitably determines the universality of this technology. In this work we describe how to build eye tracker within a very limited budget. We evaluate our eye tracker's accuracy conducting subjective experiments measuring the accuracy of DIY eye tracker for a number of observers. The resulting accuracy (1° of the visual angle) is acceptable for many applications and comparable with similar devices.

The main drawback of DIY eye tracker is the necessity of using a chin rest. We plan to reconstruct our device so that it also supported the head tracking. Interesting solution was proposed in [22] where four infrared photodiodes are located in the corners of a monitor screen. Infrared camera captures image of the eye and reflections of the photodiodes' light are detected in the image. The solution does not require calibration and combines the eye tracking with the head tracking. However, reported accuracy of about one degree of the visual angle could be increased.

Acknowledgements. This work was supported by the Polish Ministry of Science and Higher Education through the grants no. N N516 193537 and no. N N516 508539.

References

1. Peli, E., Yang, J., Goldstein, R.B.: Image invariance with changes in size: the role of peripheral contrast thresholds. JOSA A 8(11), 1762–1774 (1991)
2. Ohshima, T., Yamamoto, H., Tamura, H.: Gaze-directed Adaptive Rendering for Interacting with Virtual Space. In: Proceedings of the 1996 Virtual Reality Annual International Symposium (VRAIS 1996), p. 103 (1996)
3. RED250 Technical Specification. SensoMotoric Instruments GmbH (2009)
4. ITU Gaze Tracker software, IT University of Copenhagen, ITU GazeGroup, `http://www.gazegroup.org/home`
5. Jacob, R.J.K., Karn, K.S.: Eye tracking in human-computer interaction and usability research: Ready to deliver the promises. The Minds Eye: Cognitive and Applied Aspects of Eye Movement Research (2003)
6. Gog, T., Scheiter, K.: Eye tracking as a tool to study and enhance multimedia learning. Learning and Instructions 20(2), 95–99 (2010)
7. Pan, B., Hembrooke, H.A., Gay, G.K., Granka, L.A., Feusner, M.K., Newman, J.K.: The determinants of web page viewing behavior: an eye-tracking study. In: Proc. of the 2004 Symposium on Eye Tracking Research & Applications (ETRA 2004), pp. 147–154 (2004)

8. Lin, C.S., Huan, C.C., Chan, C.N., Yeh, M.S., Chiu, C.: Design of a computer game using an eye-tracking device for eye's activity rehabilitation. Optics and Lasers in Engineering 42(1), 91–108 (2004)
9. Duchowski, A.T.: A breadth-first survey of eye-tracking applications. Behavior Research Methods 34(4), 455–470 (2003)
10. Robinson, D.A.: A method of measuring eye movements using a scleral search coil in a magnetic field. IEEE Trans. Biomed. Eng. 10, 137–145 (1963)
11. Kaufman, A., Bandopadhay, A., Shaviv, B.: An eye tracking computer user interface. In: Proc. of the Research Frontier in Virtual Reality Workshop, pp. 78–84. IEEE Computer Society Press (1993)
12. Duchowski, A.T.: Eye Tracking Methodology: Theory and Practice, 2nd edn. Springer, London (2007)
13. Cornsweet, T., Crane, H.: Accurate two-dimensional eye tracker using first and fourth Purkinje images. J. Opt. Soc. Am. 63(8), 921–928 (1973)
14. Crane, H., Steele, C.: Accurate three-dimensional eye tracker. J. Opt. Soc. Am. 17(5), 691–705 (1978)
15. Li., D., Babcock, J., Parkhurst, D.J.: OpenEyes: a low-cost head-mounted eye-tracking solution. In: Proceedings of the 2006 Symposium on Eye Tracking Research & Applications, ETRA 2006, pp. 95–100 (2006)
16. Yun, Z., Xin-Bo, Z., Rong-Chun, Z., Yuan, Z., Xiao-Chun, Z.: EyeSecret: an inexpensive but high performance auto-calibration eye tracker. In: Proc. of ETRA 2008, pp. 103–106 (2008)
17. Pelz, J., Canosa, R., Babcock, J., Kucharczyk, D., Silver, A., Konno, D.: Portable eyetracking: A study of natural eye movements. In: Proc. of the SPIE, Human Vision and Electronic Imaging, pp. 566–582 (2000)
18. Morimoto, C.H., Mimica, M.: Eye gaze tracking techniques for interactive applications. Computer Vision and Image Understanding 98(1), 4–24 (2005)
19. San Agustin, J., Skovsgaard, H., Hansen, J.P., Hansen, D.W.: Low-cost gaze interaction: ready to deliver the promises. In: Proceedings of the 27th International Conference Extended Abstracts on Human Factors in Computing Systems, pp. 4453–4458 (2009)
20. Morimoto, C., Koons, D., Amir, A., Flickner, M., Zhai, S.: Keeping an Eye for HCI. In: Proc. of the XII Symposium on Computer Graphics and Image Processing, pp. 171–176 (1999)
21. Agustin, J.S., Mollenbach, E., Barret, M.: Evaluation of a Low-Cost Open-Source Gaze Tracker. In: Proc. of ETRA 2010, Austin, TX, March 22-24, pp. 77–80 (2010)
22. Yoo, D.H., Chung, M.J., Ju, D.B., Choi, I.H.: Non-intrusive Eye Gaze Estimation using a Projective Invariant under Head Movement. In: Proc. of the Internat. Conf. on Automatic Face and Gesture Recognition, Washington, DC, pp. 94–99 (2002)
23. ITU-R.REC.BT.500-11: Methodology for the subjective assessment of the quality for television pictures (2002)
24. Riguer, G., Tatarchuk, N., Isidoro, J.: Real-time depth of field simulation. ShaderX2: Shader Programming Tips and Tricks with DirectX 9.0, 529–579 (2002)
25. Mantiuk, R., Bazyluk, B., Tomaszewska, A.: Gaze-Dependent Depth-of-Field Effect Rendering in Virtual Environments. In: Ma, M. (ed.) SGDA 2011. LNCS, vol. 6944, pp. 1–12. Springer, Heidelberg (2011)