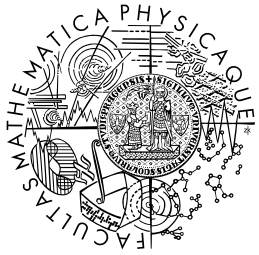

FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

MASTER THESIS

Adam Dominec

Software-based eye tracking

Department of Software and Computer Science Education

Supervisor of the master thesis:	RNDr. Barbara Zitová, Ph.D.
Study programme:	Computer Science
Study branch:	Software Systems

Prague 2016

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In ..... date ..... signature of the author

Title: Software-based eye tracking

Author: Adam Dominec

Department: Department of Software and Computer Science Education

Supervisor: RNDr. Barbara Zitová, Ph.D., Institute of Information Theory and Automation, Czech Academy of Science

Abstract: This thesis presents a software library for eye gaze tracking. The typical use case is a person watching their computer screen. All data is obtained from a single video camera and is processed in real time. The resulting software is freely available including source code.

Keywords: face tracking gaze tracking image analysis

I could not finish this work without the help of many people. But most of all, Samy is my hero!

Contents

1. Introduction

The main idea behind this program is to view human eye as a means of communication. It is rather intuitive for us to recognize other people's gaze just by looking at their face thanks to high contrast coloring of the eye itself. The corresponding brain circuitry develops early in infants (@cite) and is equivalent among individuals. This seems to be an evidence of evolutionary purpose, that the human eye is built in a way so as to display the gaze clearly. Therefore it should be possible for a computer to estimate gaze from the data us humans have available, that is, a color image.

Thanks to the fact that almost every laptop computer or cell phone is now equipped with a video camera, the hardware requirements of this software should be easy to satisfy. If applications are built on top of this program, minimal hardware requirements like this allow them to be used almost immediately upon download and completely for free.

Eye trackers can actually be used for communication, namely for human-computer interaction. Wide variety of algorithms have been proposed (@cite) for on-screen keyboard and general desktop control. The user's center of attention can also serve as a cue for adaptive rendering in video games because the rest of the screen need not be displayed in full detail. Possible applications in gaze tracking in cell phones are a vast topic, such as locking the screen when not looked at or displaying a note when somebody looks over your shoulder.

1.1 History

Perhaps a more appropriate heading of this section would be *Unrelated Work*. Indeed, before we get to the overview of our software competitors, we should shortly review past research of gaze tracking in general. Its history spans half a century before the computer era.

The interest in eye tracking originates in the field of psychology. Gaze designates the focus of attention of the subject, which in turn can tell much about ongoing cognitive processes. Furthermore, eye movement itself is the result of a rather complex neural system; nowadays, it is perhaps the best studied part of human cognition.

In order to record data with reasonable precision, sophisticated mechanical structures were often built around the subject's head. Eye movement was then measured either directly from an object attached to the eye, or indirectly from small displacements in the eye region. An especially precise technique was developed by Yarbus in 1954 and requires to stick a mirror to the surface of the eye using a small suction cup.[1] Eye movement can then be recorded directly on a piece of photographic film via the reflection of a point light source shining at the eyes.

A method known as *electrooculography* provides a less invasive alternative. There is a constant voltage gradient across the eye from back to front, in magnitude of about 1 mV. Rotation of this small dipole is measurable, so it is possible to

[1] The article is not cited here because it has been published only in Russian and does not seem generally available. However, Yarbus provides details on the method in [1].

almost directly measure the speed at which the eye moves. The advantage gained is temporal resolution: this method allows to draw a graph of angle against time. However, the actual direction is an integral of the measured value, and therefore tends to deteriorate and spatial resolution is generally poor.

Details on other purely analog methods such as this can be found in a 1967 book by Yarbus [1]. The various methods suggested for eye tracking since the end of 19th century are a story of human curiosity and can be seen as a proof of the effort directed towards this area.

1.2 Related Work

Rapid development of computers and digital video cameras has removed most of the hardware constraints mentioned so far. The demand for non-intrusive gaze tracking is high in many branches of science; for example, it is obvious that the results of a psychological experiment can greatly vary with emotional influence of the environment. Letting the subject move their head without constraints, invariance against head pose becomes a serious challenge. Head pose estimation and gaze tracking are typically handled as two separate tasks to be performed in series. Few approaches are able to encompass the both tasks within a single model.

Many substantially different approaches have been suggested for head pose estimation, and there seems to be no consensus so far. The tracker by Kanade, Lucas and Tomasi(@cite) is based on matching a template image using gradient descent optimization and can be considered the cornerstone of object tracking. The original paper describes tracking a rectangular grayscale template by horizontal and vertical shift. We should note here that such a problem can efficiently be solved on global scale by normalized cross-correlation and the Fast Fourier Transform. However, there have been numerous generalizations of this concept to a broader class of motion models such as affine(@cite) or perspective(@cite) transformations where global optimization is not feasible.

An especially sophisticated motion model are the Adaptive Appearance Models(@cite) (AAMs). A planar mesh is overlaid on the object, subdividing the template image into polygon-shaped cells. Each of the cells is able to stretch its cut-out image part when its vertices are displaced. All vertices of the mesh can be displaced separately, and they are essentially the parameter models to be optimized. The degrees of freedom of this model can be customized to application needs, so that the method will handle either rigid or soft-body transformations gracefully.

Purely geometric image transformations have poor invariance to changes in lighting, so they are well combined with element-wise image transformations. A simple option is to acquire multiple templates of the object in question, and either just select the best candidate for each input image, or allow their arbitrary linear combinations.(@cite) If the amount of training data grows large, more efficient and robust schemes are necessary. The template images can be arranged in a search structure (...) Instead of single images, linear subspaces can be taken into account using Principal Component Analysis—an approach widely known as Eigenfaces(@cite). (...)

Eye tracking is rather a simple task once the head pose is known. Much research relies on the fact that both human iris and pupil are circles, so their camera projection is always an ellipse. When the gaze direction is reasonably bounded, these projected shapes can safely be considered to remain circular(@cite). The generalized Hough transform(@cite) provides a global method of searching for circles with one-pixel precision. In many applications(@cite), this is enough.

It is possible and sometimes more robust to use an appearance-based method (. . .)

Methods with partially controlled lighting are especially efficient for eye tracking because the mammalian eye is reflective both on the outside and from the inside. In near infrared light, (...)

2. Modeling

In this section, design decisions behind all the necessary concepts are presented. Many of them are deliberate, but all are based on solid theory or measurements.

2.1 Eye

2.1.1 Eye Shape

Throughout the literature, eye is modeled either as a sphere, or with an extra spherical section for the cornea. Some sources also model the eyelids.

Sclera is white and pupil is black. Cornea is something in between.

2.1.2 Eye Movement

Eye can rotate in all three degrees of freedom, in fact.

Donder's Law is important here because of the view axis offset from the optical axis. Theoretically, position of the pupil might on its own not be sufficient to calculate gaze precisely. That is not the case.

2.2 Face

2.2.1 Face Appearance

2.2.2 Face Movement

Human face consists of many muscles. Like the pupil, they mainly serve for communication. Of special interest for us are regions that remain mostly fixed to the skull in normal conditions.

2.3 Gaze

Under the assumption of small angle divergence, we can model movement of the pupil as simple translation.

Given the eye–face offset, onscreen gaze center is assumed to be a homography. This saves us from explicitly modeling the viewed scene. The non-linear factor is quite huge, unfortunately.

2.4 Image

The image as acquired from the camera is assumed to be a rectangular grid of colored points. However, certain parts of the computation require a continuous image model in order to obtain sub-pixel precision. This problem has three solutions in general:

- Use simple interpolation and ignore the inaccuracy induced. This is the approach applied in this thesis.
- Use simple interpolation on a blurred image so that the inaccuracy disappears. This is the solution implicitly used by many software libraries.
- Interpolate using a sophisticated function. Surprisingly enough, commonly used interpolation functions are not suitable for a precise model.

3. Solving

This chapter provides mathematical methods for solving the problems presented so far.

3.1 Projective fitting

The correspondence between measured values and the appropriate results is assumed to be a homography. There are, loosely speaking, two difficulties in homography fitting as compared to affine transformations. Firstly, it has a nonlinear component so an algebraic least-squares solution is generally different from a geometric least-squares solution. The former can be obtained in a much more stable way than the latter, so we start with algebraic least squares fitting and use the result for initialization of a nonlinear optimization method. Secondly, the measured data points are defined only up to scale, so it is not possible to formulate a linear system straight away. A technique called Direct Linear Transformation is widely used to overcome this issue.

3.1.1 Direct Linear Transformation

We are presented with a set projective correspondences of the form

$$\mathbf{H}\mathbf{x}^k = \alpha_k \mathbf{p}^k, 1 \leq k \leq K.$$

Unfortunately, these equations do not form a linear system with respect to the unknown matrix $\mathbf{H}$ because there is an unknown scale factor α_i involved in each equation. Differences among the scale factors make up the perspective part of the homography, so although they can make the problem numerically unstable, we cannot impose any limits on these values. The core idea of Direct Linear Transformation is to find a set of vectors that must be orthogonal to the solution, and then solve the resulting homogeneous system.

Let us define a set of antisymmetric matrices $\{\mathbf{M}^{i,j} \in \mathbb{R}^{n \times n}\}$, $1 \leq i < j \leq n$, each having only two nonzero elements:

$$\mathbf{M}^{i,j}_{i,j} = 1, \mathbf{M}^{i,j}_{j,i} = -1.$$

From this definition it follows that for any $\mathbf{p} \in \mathbb{R}^n$, the product $\mathbf{p}^\mathsf{T}\mathbf{M}^{i,j}\mathbf{p} = \mathbf{p}_i\mathbf{p}_j - \mathbf{p}_j\mathbf{p}_i = 0$. The vectors $\{\mathbf{M}^{i,j}\mathbf{p}\}, 1 \leq i < j \leq n$, are all orthogonal to a given $\mathbf{p} \in \mathbb{R}^n$.

With increasing dimension, this set of vectors becomes heavily redundant. Clearly, a minimal solution would form a basis of the subspace $\mathbb{R}^n/\mathbf{p}$ orthogonal to $\mathbf{p}$ and thus would consist of $n-1$ vectors, whereas this approach generates $\frac{1}{2}n \cdot (n-1)$ vectors. Some sources[2] suggest to pick an arbitrary subset of size $n-1$ from the matrices defined above and use these for the whole data set. A more proper solution is to select these matrices specifically for each correspondence pair $\mathbf{x}^k \leftrightarrow \mathbf{p}^k$ so as to avoid possible degeneracies. In particular, we find the largest vector element $\mathbf{p}_l$ (in absolute value) and then select all matrices $\mathbf{M}^{i,j}$ such that $i = l$ or $j = l$. Each of the vectors generated this way contains the value of $\mathbf{p}_l$ at

a different position (i.e., different vector element), therefore the vectors are linear independent and they form a basis of $\mathbb{R}^n/\mathbf{p}$.

Now we can formulate the linear system. It will consist of $K \cdot (n-1)$ equations, K is the number of correspondence pairs $\mathbf{x}^k \leftrightarrow \mathbf{p}^k$. Its unknowns will be precisely the elements of the matrix $\mathbf{H}$. Each equation of the system represents a constraint of the form $\mathbf{p}^\mathsf{T}\mathbf{M}\mathbf{H}\mathbf{x} = 0$, which is the orthogonality constraint as proposed above. This constraint can be reformulated in terms of the Frobenius inner product $\langle \mathbf{A}, \mathbf{B} \rangle_F = \sum_{i,j} \mathbf{A}_{i,j} \cdot \mathbf{B}_{i,j}$ and the vector outer product as follows:

$$\mathbf{p}^\mathsf{T}\mathbf{M}\mathbf{H}\mathbf{x} = \sum_{i,j} (\mathbf{p}^\mathsf{T}\mathbf{M})_i \mathbf{H}_{i,j} \mathbf{x}_j = \langle \mathbf{p}^\mathsf{T}\mathbf{M}\mathbf{x}^\mathsf{T}, \mathbf{H} \rangle_F = 0.$$

Viewing the matrices as vectors of their elements, this finally leads to an equation in the suitable form:

$$\mathrm{vec}(\mathbf{p}^\mathsf{T}\mathbf{M}\mathbf{x}^\mathsf{T})^\mathsf{T} \cdot \mathrm{vec}(\mathbf{H}) = 0,$$

where $\mathrm{vec} : \mathbb{R}^{m \times n} \to \mathbb{R}^{mn}$ stacks all matrix elements to a vector, in row-major order. Each correspondence pair and each selected antisymmetric matrix $\mathbf{M}^{i,j}$ for that pair provide one such equation, and the row vectors $\mathrm{vec}(\mathbf{p}^{k\mathsf{T}}\mathbf{M}^{i,j}\mathbf{x}^{k\mathsf{T}})^\mathsf{T}$ can be stacked to form the system matrix.

3.1.2 Singular Value Decomposition

Homogeneous linear systems such as this one can be solved using the Singular Value Decomposition (SVD).

However, if the input vectors are degenerate and only span a subspace of $\mathbb{R}^m$, there are multiple solutions and not all of them are proper solutions of the original equation $\mathbf{H}\mathbf{x}^k = \alpha_k \mathbf{p}^k$. In particular, it is possible that a solution will produce points at infinity, meaning that $\mathbf{p}_n \approx 0$ and that the Cartesian counterpart of $\mathbf{p}$ is undefined. Instead of avoiding such degeneracies explicitly, we loop through the resultant singular vectors and choose the one with the minimal reprojection error.

The system matrix in our case is of shape $K \cdot (n-1) \times m$, which leads to an overall time complexity of $O(m^2 K(n-1))$.

3.1.3 Nonlinear optimization

3.1.4 Four-point homography

The following text are notes from my own research and do not probably fit into the flow of the thesis so far.

A minimal case of a 2-dimensional homography estimation is a correspondence of 4 point pairs. The general formula as generated by DLT would be needlessly bloated for this setting.

Let us write the point sets as columns of a 3×4 matrix, and let us denote the equivalence of two point sets up to scale by the $\sim$ (tilde) character:

$$\mathbf{A} \sim \mathbf{B} \Leftrightarrow \exists \alpha_1 \dots \alpha_4 : \mathbf{A} \cdot \begin{pmatrix} \alpha_1 & & \\ & \ddots & \\ & & \alpha_4 \end{pmatrix} = \mathbf{B}.$$

Also, let us declare the following set as the *canonical configuration*:

$$\mathbf{C} = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}.$$

We will now present the formulas to convert an arbitrary 4-point set to and from the canonical configuration.

For a given point set $\mathbf{A} = (\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d})$, the homography $\mathrm{can}(\mathbf{A})$ is defined as:

$$\mathrm{can}(\mathbf{A}) = \begin{pmatrix} \alpha_1(\mathbf{b} \times \mathbf{c})^\mathsf{T} \\ \alpha_2(\mathbf{c} \times \mathbf{a})^\mathsf{T} \\ \alpha_3(\mathbf{a} \times \mathbf{b})^\mathsf{T} \end{pmatrix},$$

where $\alpha_1 \dots \alpha_3$ are chosen so that $\mathrm{can}(\mathbf{A}) \cdot \mathbf{d} = (1, 1, 1)^\mathsf{T}$. Note that this requires no linear solving, just an element-wise division. If the points $\mathbf{A}$ are affine independent, it clearly follows that $\mathrm{can}(\mathbf{A}) \cdot \mathbf{A} \sim \mathbf{C}$.

The homography $\mathrm{dec}(\mathbf{A})$ is, in turn, defined as:

$$\mathrm{dec}(\mathbf{A}) = (\alpha_1\mathbf{a}, \alpha_2\mathbf{b}, \alpha_3\mathbf{c}),$$

where $\alpha_1 \dots \alpha_3$ are chosen so that $\mathrm{dec}(\mathbf{A}) \cdot (1, 1, 1)^\mathsf{T} = \mathbf{d}$. This requires us to solve the 3×3 linear system $(\mathbf{a}, \mathbf{b}, \mathbf{c}) \cdot (\alpha_1, \alpha_2, \alpha_3)^\mathsf{T} = \mathbf{d}$. As a result it follows that $\mathrm{dec}(\mathbf{A}) \cdot \mathbf{C} \sim \mathbf{A}$. Note that the vector $\mathbf{d}$ is exactly the right-hand side of this system. A neat side-effect is that the inverse matrix $\mathrm{dec}(\mathbf{A})^{-\mathsf{I}} = \mathrm{can}(\mathbf{A})$ has correct scale. More importantly, it means that the mapping dec is linear with respect to all elements of $\mathbf{d}$.

These two mappings can be composed to provide any four-point homography as necessary:

$$\mathrm{dec}(\mathbf{B}) \cdot \mathrm{can}(\mathbf{A}) \cdot \mathbf{A} \sim \mathbf{B}.$$

The resulting mapping is linear with respect to the last column of $\mathbf{B}$. To obtain alternate formulations that are linear with respect to the remaining three columns of $\mathbf{B}$, we can use the following homography:

$$\mathbf{R} = \begin{pmatrix} 0 & 0 & 1 \\ -1 & 0 & 1 \\ 0 & -1 & 1 \end{pmatrix}.$$

It has the effect of a cyclic permutation upon the points of $\mathbf{C}$, namely:

$$\mathbf{R} \cdot \mathbf{C} \sim \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}.$$

This particular choice is also pleasant from a numeric point of view, because $\det(\mathbf{R}) = 1$ and therefore also $\mathbf{R}^4 = \mathbf{I}_3$, the identity matrix.

We observe that $\mathrm{dec}(\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}) \cdot \mathbf{C} \sim \mathrm{dec}(\mathbf{d}, \mathbf{a}, \mathbf{b}, \mathbf{c}) \cdot \mathbf{R} \cdot \mathbf{C}$ and because four points define the homography matrix (up to scale), we can safely conclude that $\mathrm{dec}(\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}) = \mathrm{dec}(\mathbf{d}, \mathbf{a}, \mathbf{b}, \mathbf{c}) \cdot \mathbf{R}$.

3.1.5 Derivatives of a homography

We wish to calculate the derivative of a four-point homography with respect to the coordinates of its four control points. The linearity of $\mathrm{dec}(\mathbf{A})$ implies that the derivative $\frac{\partial}{\partial \mathbf{d}_i}\mathrm{dec}\,(\mathbf{a},\mathbf{b},\mathbf{c},\mathbf{d}) = \mathrm{dec}\,(\mathbf{a},\mathbf{b},\mathbf{c},\mathbf{e}_i)$, where $\mathbf{e}_i$ is the unit vector for the coordinate axis i.

Derivative with respect to $\mathbf{c}$ can be obtained by a permutation of the points:

$$\mathrm{dec}(\mathbf{A}) = \mathrm{dec}(\mathbf{A}) \cdot \mathbf{R}^{-\mathsf{I}} \cdot \mathbf{R} \sim \mathrm{dec}(\mathbf{d},\mathbf{a},\mathbf{b},\mathbf{c}) \cdot \mathbf{R},$$

$$\frac{\partial}{\partial \mathbf{c}_i}\mathrm{dec}(\mathbf{A}) \sim \frac{\partial}{\partial \mathbf{c}_i}\mathrm{dec}(\mathbf{d},\mathbf{a},\mathbf{b},\mathbf{c}) \cdot \mathbf{R} = \mathrm{dec}\,(\mathbf{d},\mathbf{a},\mathbf{b},\mathbf{e}_i) \cdot \mathbf{R}.$$

(*TODO: I'm not exactly sure if the permutation works this way or the other.*) The remaining two points $\mathbf{a},\mathbf{b}$ are obtained by iteration of this scheme. The scale factors are important and need to be calculated, too.

4. Implementation

4.1 Gaze Estimation

4.1.1 Face Tracking

4.1.2 Eye Tracking

4.2 Calibration

Conclusion

Bibliography

[1] Alfred L Yarbus. *Eye movements during perception of complex objects*. Springer, 1967.

[2] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Second Edition. Cambridge university press, 2003.

List of Abbreviations

Attachments