Gaze Tracking Based on Face-Color

Bernt Schiele and Alex Waibel
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213-3890
E-mail: Bernt.Schiele@imag.fr and waibel@cs.cmu.edu

Abstract

In many practical situations, a desirable user interface to a computer system should have a model of where a person is looking at and what he/she is paying attention to. This is particularly important if a system is providing multimodal communication cues, speech, gesture, lipreading, etc., [2, 3, 8] and the system must identify, whether the cues are aimed at it, or at someone else in the room. This paper describes a system that identifies user focus of attention by visually determining where a person is looking. While other attempts at gaze tracking usually assume a fixed or limited location of a person's face, the approach presented here allows for complete freedom of movement in a room. The gaze-tracking system, uses several connectionist modules, that track a person's face using a software controlled pan-tilt camera with zoom and identifies the focus of attention from the orientation and direction of the face.

1 Introduction

One major impediment to user acceptance of speech interfaces in many potential applications is the fact that people need to wear head-sets, gloves, etc. and operate push-buttons to control the system. A commonly proposed solution to this problem in speech is for the recognizer always to listen to all of the sounds within a room. Typically keywords or phrases, and loudness are used to identify the onset of speech for recognition. This in turn leads to very fragile user interfaces, that "turn-on" at odd places and times: An always-listening recognizer, typically cannot identify if an utterance or phrase is directed at it or at someone else in the room, or even worse, if the putative utterance was in fact a false recognition to begin with (a noise or similar sounding word for example). For robust and useful interfaces, a computer system must identify user intent and focus of attention so that it can recognize when it is being addressed, and/or who a person in the room might be addressing or interacting with.

In this paper, we propose a solution to this problem by visually identifying where a person is looking and the direction of their attention. Unlike systems that require considerable limitation of movement to identify location or to track gaze, our system allows for freedom of movement around a room by tracking a persons face and identifying the orientation and gaze from the image of the whole face.

1.1 Overview of the Gaze Tracking System

The gaze tracking system consists of three steps. In the first step a human face is located and the camera zooms on this face. This first step is achieved with our Facetracker [5] which is introduced in section 2. The Facetracker works at approximately 10Hz. The output of the Facetracker is a normalized face or more precisely the region of the image, which contains the face. The subsequent steps in the function of the Gazetracker use only this region of the image.

The second step of the Gazetracker consists of two parallel procedures. The first procedure "intensifies" all objects with face-color. Figure 1(b) shows such a color-intensified image. In parallel the second procedure adjusts the face-color to the face in the image. Even though the face-colors of many humans are relatively similar we have to adjust the face-color, if we want to distinguish this face properly from the background. This important part of the Gazetracker is described in section 3.

In the third step the color-intensified image is projected into the input-units of our artificial neural network and a forward pass of the neural network is calculated. The output of the neural network gives directly the orientation of the head. This third step, the design of the network and the training of the network are described in section 4.

Section 5 summarizes some results from experiments with the Gazetracker.

2 Facetracker

This section briefly introduces the Facetracker, considerably more detail has been published in [6]. We also refer to [5] which describes the Facetracker in detail and can be easily obtained via WWW[1].

The camera used in the system (Sony CCD-TR 101) is mounted on two stepper-motors, allowing horizontal and vertical turns. The stepper-motors are controlled in a serial port. The remote control of the zoom lens of the camera has been engineered to allow control through a second serial port. The camera images are obtained by a frame-grabber, which digitizes the video-signal into RGB-values. The entire computation is performed on a single HP 9000/735 workstation.

To find a face the Facetracker searches for the largest moving object which is skin-colored. As soon as a face is found the system tracks it (only based on skin-color) at approximately 10Hz. This processing is the average time needed by the system and includes camera zooming and movement. This is possible, since the Facetracker works on a very low resolution and we can define a virtual camera (e.g. half the size of the whole camera-image) which can be "moved" and "zoomed" much faster than the real camera. Neglecting the time for zooming and moving the real camera, the Facetracker operates at to 25Hz.

The output of the Facetracker is a normalized face or more precisely the region of the image, which contains the face. The precision of the Facetracker varies strongly with the velocity of the head movements. It is usually in the range between 10% and 20% (variation in position and size). The subsequent steps of the Gazetracker have to cope with this imprecision, since they use only the region provided by the Facetracker.

It should be pointed out, that the Gazetracker uses a second frame-grabber (connected with the same camera) and works on a second HP 9000/735 workstation. This enables the Gazetracker to work at a higher resolution and independent of the Facetracker.

3 Self-adjustable Face-color Intensifier

The intensification of face-color (of a particular person) in an image (as in figure 1(b)) has three main advantages: First of all we can distinguish the face and the background. Secondly we can find features within the face in the color-intensified image. Such features are for example the eyes and the mouth, these features are differentiated because they don't have face-color. The third advantage is that we obtain the shape of the face in the image. These three advantages together make it possible to use the face-color-intensified image as input to neural network. We have that the network generalises very well with respect to different backgrounds and different individual faces (namely for white faces, faces of Asian people and also for most Indian people).

The first procedure of the Face-Color-Intensifier calculates and adjusts a color-map to a particular person. This procedure is described in the first part of the section and is similar to the Face-Color-Classifier which is used for the Facetracker. Based on color-maps we obtain a binary image (see figure 2(c)) where we can distinguish regions with face/skin-color from the rest of the image. The second part of the section explains in more detail how to use such a color-map in order to intensify face-color in an image. The result of the Intensifier is an image with different grey-levels, where higher values correspond to frequent colors in the face and low values to infrequent colors. Such an image is shown in figure 1(b).

3.1 Color-maps

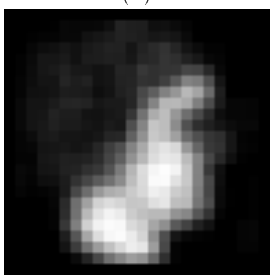
The red (R), green (G) and blue (B) values of a pixel of the video-camera signal can be transformed into different representations. Since the intensity or brightness of a given pixel value doesn't contain information about the color of the pixel, we want explicitly to ignore intensity. Chromatic colors provide such a representation by normalizing the RGB-value by its intensity. The chromatic colors are defined as [1]:

$$\begin{aligned} r &= \frac{R}{R+G+B} \\ g &= \frac{G}{R+G+B} \\ b &= \frac{B}{R+G+B} \end{aligned}$$

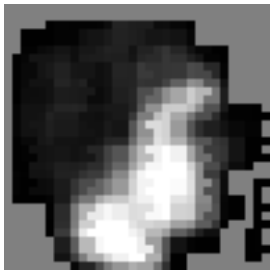
All subsequent steps of the Gazetracker are based on these two-dimensional representations (r, g) of chromatic colors (The third value b can be neglected, since the values always sum to one: $r + g + b = 1$). Using this (r, g)-representation we calculate a probability density function of the chromatic color of an image (or only part of an

[1] http://www.cs.cmu.edu:8001/afs/cs.cmu.edu/user/clamen/mosaic/reports/1994.html

(a)



(b)



(c)

Figure 1: Face-Color-Intensifier: (a) grey-scale image, (b) face-color-intensified image, (c) ANN-input

(a)

(b)

(c)

Figure 2: Applying the general color-map

image):

$$p(r,g) = \frac{h(r,g)}{\sum_{r,g} h(r,g)}$$

where $h(r,g)$ represents the histogram of the number of pixels in the image with the chromatic color (r,g). In the following we will call $p(r,g)$ *the color-map*.

By analyzing the color-maps of many human faces all possible face-colors (or better skin-colors of all available faces: faces of Asian people, Indian people and white people) are located in a relatively small bandwidth of the (r,g)-values [5]. Therefore we can construct a general color-map which contains most of the possible face-colors. This is done basically by averaging the face-colors of the available people. Figure 2(b) shows the application of the general color-map to an image. In this image a certain pixel with the chromatic color (r,g) has the value $p(r,g)$ of the general color-map. Brighter regions therefore correspond to chromatic colors with a high value $p(r,g)$ in the general color-map. Darker regions correspond to low value $p(r,g)$ in the general color-map. This application of a color-map to an image is the basic procedure of the Face-Color-Intensifier.

By smoothing and thresholding we obtain a binary image (see figure 2(c)). This binary image is used (at the beginning together with a motion detector) to find the largest object in the image with face-color. This object is then used to adjust the general color-map to a particular human face. The adjustment can be formulated as:

$$adj_{n+1}(r,g) = \begin{cases} \frac{1}{3}(obj_n(r,g) + 2adj_n(r,g)) & \text{if } gen(r,g) \geq min \\ 0 & \text{otherwise} \end{cases}$$

where gen is the general color-map, adj_n is the adjusted color-map at time n and obj_n is the color-map of the largest object in the image with face-color (initialization: $adj_0 = gen$).

3.2 Face-Color-Intensifier

The described Color-maps are also used to intensify the face-color in an image. Since the color in a human face varies considerably between different regions of the face, we increase the values of the color-map relatively to their histogram. This is a change in the shape of the face-color-histogram rather than in the region of the face-color. This mainly increases the color-values which occur with higher frequency in the face and strengthen the distinction from the background colors. An image with this enhanced Color-Intensifier is shown in figure 1(b). One can see the quality of this color-intensified image.

Figure 1(c) shows the final color-intensified image. It is obtained by normalizing the image of figure 1(b). This normalization procedure projects the highest 5% of values onto the highest grey-value and projects the lowest 5% of values in the image onto the lowest grey-value. The rest of the values are then linearly distributed between these two extremes. This normalization ensures that the Color-Intensifier is independent of different humans and different lightening conditions. It also distributes the values of the image more uniformly onto the input of the neural network.

4 Network architecture

Before we can use an artificial neural network we have to decide which architecture and which training-algorithm to use. In this case we have found that our problem could be solved with a Multi-Layer-Perceptron (MLP) trained by standard back-propagation [4]. The architecture, the input and output representation and the interpretation of the output are described in the following sections.

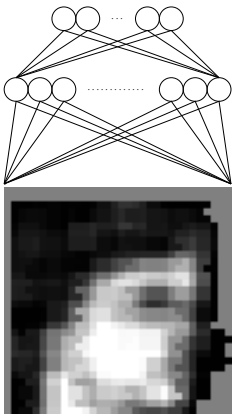


Figure 3: Network architecture

The MLP-architecture is summarized in figure 3. The input consists of 32×32 neurons. The color-intensified image is projected onto these input-neurons. We are using one hidden layer with 50 hidden units (the number of hidden units was determined empirically). For output we used on the one hand side 3 output-units (namely for the directions *left*,*straight* and *right*) and on the other hand 15 output-units which correspond to the possible head directions $-70, -60, \ldots, +60, +70$ degree.

4.1 Input representation

The input to the MLP is a 32 x 32 – color-intensified image (Fig. 1(c)). The relatively low resolution of the input-image makes it possible to apply the Gaze-tracker in real-time. It also reduces the training-time and the amount of training data needed (since the size of the training-data should be at least in the range of the free parameters of the neural network [7]). Unfortunately the low resolution has the disadvantage that features such as the eyes and the mouth are difficult to find in the image. But the training-results show that the neural network is still able to determine the direction of the head with sufficient accuracy (see section 5).

4.2 Output representation and interpretation

We used two different configurations for the output-units. The first configuration consisted of 3 output-units corresponding to the three discrete directions of the head: *left* for the angles of the interval $[-70, -30]$, *straight* for the angles of the interval $[-20, +20]$ and *right* for the angles of the interval $[+30, +70]$. The second representation used 15 output-units where each unit corresponds to one of the angles $-70, -60, \ldots, +60, +70$.

An important question concerning the output-representation was how to project a desired output (here the direction of the head) onto the output-units. The 3 output-units indicate whether the person is looking *straight*, *left* or *right*. The projection therefore is a choice of one from three possibilities. Only one unit is supposed to be "on" and the other two are supposed to be "off".

The second solution has 15 output-units. Here we tried two different representations. The first is, as for 3 outputs, an 1 from 15 decision. But the better results were obtained with a gaussian representation (see figure 4). Here not only the unit, which corresponded to the desired output is "on", the units close to this desired unit are to a certain degree "on". The main advantage of the gaussian representation is that the output-unit learning is not only based on the input-images which correspond exactly to the angle of the output. Instead a output-unit learns also from images which are similar and which correspond to nearby units. Nevertheless a necessary condition for this representation is that similar input-images correspond to similar outputs.

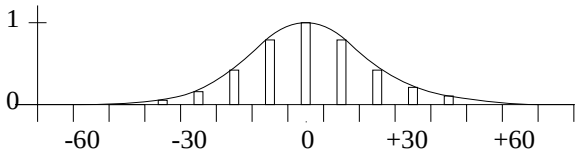


Figure 4: the gaussian output representation

5 Results

This section gives an overview of the performance of our Gazetracker. In order to give the training and test results we first describe our image-database and the production of the sample-sets. The following sections gives the off-line results on the training and test-set.

5.1 Training and Test data

As training data we use four sets of 15 images of 7 different people. The 15 images correspond to the directions $-70, -60, \ldots + 60, +70$ degrees of a person's head. The people were asked to sit in front of the camera and to look in the specified directions $-70, -60, \ldots + 60, +70$ degrees. The images were taken with a blue-screen–background. The blue-screen enabled us to project an arbitrary background behind the heads of the people. During the production of the training and test sets we replaced the blue-screen with a randomly chosen background from our lab. This forces the neural network to learn independent of the background.

In order to learn the neural network shift-invariant we shift each of the images artificially by $\pm 15\%$ in x and y–direction with a step-size of 3%. The training-set contains therefore $4 \times 15 \times 7 \times 11^2 = 50820$ images. The test data contains four sets of 15 images of 2 people. The test data is independent of the training data. That means that neither of the two people are included in the training data. The test data contained then $4 \times 15 \times 2 \times 11^2 = 14520$ images.

5.2 Results

Training the network was time-consuming activity due to the size of the training set. But the number of iterations was been always within the range 100–150. Table 1 and figure 5 show the training- and test-results of the neural networks described in the previous section.

Table 1 corresponds to the neural network with 3 output-units and the 1 from 3 representation (see section 4). This table shows the capability of the network architecture to distinguish between the three head-directions *right*, *straight* and *left*.

99.72% of the training data and 99.65% of the independent test set were classified correctly.

Error [1/units]	0	1	2	av.
% training	99.72	0.27	0.01	0.0028
% test	95.65	4.35	0.0	0.044

Table 1: Training results with 3 output-units

The figure 5 correspond to the neural network with 15 output-units and the gaussian output-representation (see section 4). The performance of this neural network is encouraging especially if you consider that the average error less than 10 degrees on the training data and 12 on the test data. There is some inaccuracy in the training data, since the gaze in a certain direction is only partially correlated with orientation of the head. Considering this we can conclude that the Gazetracker is able to determine the head-direction with a high accuracy.

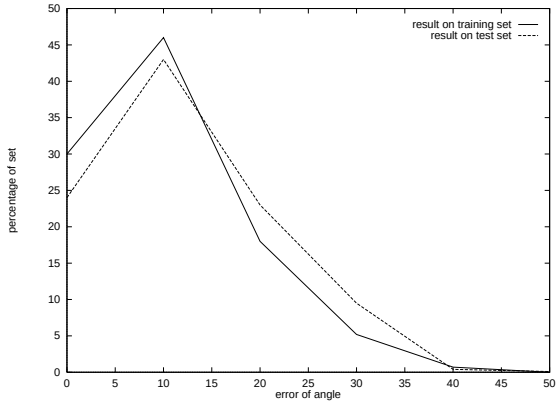


Figure 5: Training results with 15 output-units

5.3 Runtime results

At the moment we have two setups for the Gazetracker. The first setup works without the Facetracker. This system can be used, when a person is sitting in a certain place, as e.g. in front of a computer. This setup has a typical cycle time of 93ms (on a workstation HP 9000/735) so that it works at approximately 10Hz.

The second setup, as initially described, uses the Face-tracker as pre-processing step. This setup is used when a person is walking or moving around in the room. The Facetracker then finds and tracks the person and provides the Gazetracker with a "normalized face" (see section 2). Both processes run independently on two workstations. This setup has a typical cycle time of 134ms, due to the communication needed with the Facetracker.

To measure the precision of the first setup we analysed the results of image sequences, where one person looked every five seconds at different predefined positions in the room. The precision for this setup is the same as for the test-set, i.e. about 12 degrees. To measure the performance of the second setup we defined 5 different positions in the room with 4 different points to look at (with arbitrary angles). One person was asked to move to this locations and to look at one of the predefined points. As long as the person was standing at this location, we recorded the output of the Gazetracker. In this setup the Gazetracker works with a slightly lower precision (about 15 degrees), due to the delay of about 200ms which is needed for the Facetracker and the Gazetracker for calculation (after the movement to a new location).

6 Conclusion

We described one further step towards a multimodal human-to-computer interface. We introduced a component of our vision system, namely the real-time connectionist Gazetracker which determines the angle of a human head relatively to the camera. Training and runtime results show the capability of the Gazetracker.

7 Acknowledgements

The authors wish to thank the members of the INTERACT project for help in this research. Thanks to Martin Hunke and Ravi Desai for help with the implementation of components of the Face-tracker and Gaze-tracker.

This research was sponsored by the Department of the Navy, Office of Naval Research under Grant No. N00014-93-1-0806.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government.

References

[1] D.H. Ballard and C.M. Brown. *Computer Vision*. Englewood Cliffs, N.Y., 1982.

[2] U. Bub, M. Hunke, and A. Waibel. Knowing who to listen to in speech recognition: Visually guided beamfroming. In *International Conference on Acoustics, Speech, and Sig-*

nal Processing (ICASSP), Detroit, Michigan, 1995.

[3] P. Duchnowski, M. Hunke, D. Busching, U. Meier, and A. Waibel. Toward movement-invariant automatic lip-reading and speech recognition. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Detroit, Michigan, 1995.

[4] J. Hertz, A. Krogh, and R. Palmer. *Introduction to the Theory of Neural Networks*. Addison Wesley, 1991.

[5] M. Hunke. Locating and tracking of human faces with neural networks. Technical Report CMU–CS–94–155, Carnegie Mellon University, August 1994.

[6] M. Hunke and A. Waibel. Face locating and tracking for human–computer interaction. In *Twenty-Eight Asilomar Conference on Signals, Systems & Computers*, California, November 1994.

[7] D. Pomerleau. Neural network perception for mobile robot guidance. Technical Report CMU–CS–92–115, Carnegie Mellon University, February 1992.

[8] M. Tue Vo and A. Waibel. A multi-modal human-computer interface: Combination of gesture and speech recognition. In *CHI 1993*, 1993.