GAZE TRACKING FOR MULTIMODAL HUMAN-COMPUTER INTERACTION

Rainer Stiefelhagen and Jie Yang

Interactive Systems Laboratories
University of Karlsruhe — Germany, Carnegie Mellon University — USA*
stiefel@ira.uka.de, yang+@cs.cmu.edu

ABSTRACT

This paper discusses the problem of gaze tracking and its applications to multimodal human-computer interaction. The function of a gaze tracking system can be either passive or active. For example, a system can identify user's message target by monitoring the user's gaze, or the user could use his gaze to directly control an application or launch actions. We have developed a real-time gaze tracking system that estimates the 3D position and rotation (Pose) of a user's head. We demonstrate the applications of the gaze tracker to human-computer interaction by two examples. The first example shows that gaze tracker can help speech recognition systems by switching language model and grammar based on user's gaze information. The second example illustrates the combination of the gaze tracker and a speech recognizer to view a panorama image.

1. INTRODUCTION

Multimodal human-computer interaction has received much attention recently. Several researchers have studied the effectiveness of multimodal human-computer interaction [1, 2, 3]. Multimodal interfaces benefit from the redundancy, naturalness and flexibility that arise from exploiting alternate and complementary communication cues. Our research efforts at the Interactive Systems Laboratories (Carnegie Mellon University and University of Karlsruhe) are focused on producing a sensible and useful user interface to support the multimodal human-computer interaction. Some of our initial works along this line have been reported in previous publications [4, 5]. While multimodal interfaces offer greater flexibility and robustness, they have still been largely pen- or voice-based, user activated, and operate in settings where headsets, helmets, suits, buttons or other constraining devices are required. If more freedom is to be provided to users, some important parameters of the communicative situation have to be identified. For example, who or what is the target and object of the message (focus of attention). This information provides communication cues to a multi-modal interface. One way to obtain such information is through gaze tracking.

In this paper, we address the problem of gaze tracking and its application to multimodal human-computer interaction. A person's gaze direction is determined by two factors: the orientation of the head, and the orientation of the eyes. We limit our discussion to the head orientation in this paper. A real-time gaze tracker is a prerequisite for tracking user's gaze. There have been several approaches to compute the gaze of a person. Hardware-intensive and/or intrusive methods, where the user has to wear special headgear, or methods that use expensive hardware such as radar-range-finder [6]. Recently, there have been proposed non intrusive gaze trackers using mainly software. For example, Cipolla & Gee [7] developed a system to track the rotation and position of the head by finding correspondences between facial feature points and corresponding points in a model of the head, using a weak perspective projection. However, the system has to be initialized manually because the system cannot locate the face and the facial feature points automatically.

We have developed a non-intrusive model-based gaze-tracking system [8, 9]. The system estimates the 3-D pose of a user's head by tracking as few as six facial feature points. The system locates a human face using a statistical color-model without any mark on the face. It is able to find and track facial feature points automatically, as soon as a person appears in the field of view of the camera, and turns his face toward the camera. The system then finds and tracks the facial features, such as eyes, nostrils and lip-corners. The system is also able to recover from tracking failures.

In a multimodal interface the function of a gaze tracking system can be either passive or active. For example, a system can identify user's message target by monitoring the user's gaze, or launch an action by user's gaze. Furthermore, a gaze tracking system can be used alone, or/and combined with other system such as a speech recognition system. We demonstrate the applications of the gaze tracker to human-computer interaction by two examples. The first example shows that gaze tracker can be used to enhance the performance of a multimodal interface. The second example illustrates the combination of the gaze tracker and a speech recognizer to view a panorama image.

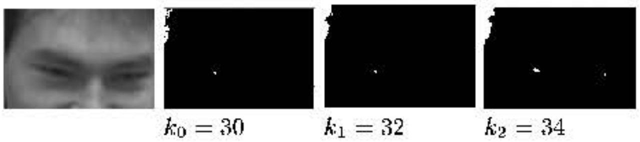
2. A REAL-TIME GAZE TRACKER

In this section we briefly describe how to track gaze in real-time [8, 9].

In our system we are estimating the gaze of the user by computing the pose of his head. This is done by finding correspondences between five to six model points such as eyes, nostrils and lip corners in a simple 3D model of a head, and their corresponding locations in a camera image. To compute the pose from these 3D to 2D correspondences we used the POSIT algorithm, recently proposed by DeMenthon and Davis [10].

In order to compute the pose, the facial features must

*This work is supported by ARPA under grant number N00014-93-1-0806.



$k_0 = 30$ $k_1 = 32$ $k_2 = 34$

Figure 1. Iterative thresholding of the search window

be searched and tracked in the camera image. To search the facial features we use a top-down approach: First we search the facial area in the image, using a statistical color model, then the search of the facial features is restricted to certain areas inside the face. Once the features have been found, the search for these features can be restricted to small search windows around their previous positions, and faster search stratgies can be applied. Furthermore, these local search windows can be predicted using linear extrapolation over previous positions and their size can be adjusted to the actual size of the face in the image.

Searching the Face

To find and track the face, we use a statistical color-model consisting of a two-dimensional Gaussian distribution of normalized face colors [11]. The input image is searched for pixels with face colors and the largest connected region of face-colored pixels in the camera-image is considered as the region of the face. The color-distribution is initialized so as to find a variety of face-colors and is gradually adapted to the actual found face.

Searching the Pupils

Assuming a frontal view of the face initially, we can search the pupils by looking for two dark regions that satisfy certain geometric constraints and lie within a certain area of the face.

For a given situation, these dark regions can be located by applying a fixed threshold to the grayscale image. However, the threshold value may change for different people and lighting conditions. To use the thresholding method under changing lighting conditions, we developed an iterative thresholding algorithm. The algorithm iteratively thresholds the image until a pair of regions that satisfies the geometric constraints can be found. Figure 1 shows the iterative thresholding of the search window for the eyes with thresholds k_i. After three iterations, both pupils are found.

Because the thresholding value is adjustable, this method is able to apply to various lighting conditions and to find the pupils in very differently illuminated faces robustly.

Searching the Lip Corners

First, the approximate positions of the lip corners are predicted, using the positions of the eyes, the face-model and the assumption, that we have a near-frontal view. A generously big area around those points is extracted and used for further search.

Finding the vertical position of the line between the lips is done by using a horizontal integral projection P_h of the grey-scale-image in the search-region. Because lip line is the darkest horizontally extended structure in the search area, its vertical position can be located where P_h has its global minimum.

The horizontal boundaries of the lips can be found by applying a horizontal edge detector to the refined search area and regarding the vertical integral projection of this horziontal edge image. The positions of the lip corners can be found by looking for the darkest pixel along the two columns in the search area located at the horizontal boundaries.

Searching the Nostrils

Similar to searching the eyes, the nostrils can be found by searching for two dark regions, that satisfy certain geometric constraints. Here the search-region is restricted to an area below the eyes and above the lips. Again, iterative thresholding is used to find a pair of legal dark regions, that are considered as the nostrils.

Tracking the Eyes

For tracking the eyes, simple darkest pixel finding in the predicted search-windows around the last positions is used.

Tracking the Lip Corners

Tracking the lip-corners consists of the following steps:

1. Search the darkest pixel in a search-region right of the predicted position of the left corner and left of the predicted position of the right corner. The found points will lie on the line between the lips
2. Search the darkest path along the lip-line for a certain distance d to the left and right respectively, and choose positions with maximum contrast along the search-path as lip-corners

Because the shadow between upper and lower lip is the darkest region in the lip-area, the search for the darkest pixel in the search windows near the predicted lip corners ensures that even with a bad prediction of the lip corners, a point on the line between the lips is found. Then the true positions of the lip corners can be found in the next step. Figure 2 shows the two search windows for the points on the line between the lips. The two white lines mark the search paths along the darkest paths, starting from where the darkest pixel in the search windows have been found. The found corners are marked with small boxes.

Figure 2. Search along the line between the lips

Tracking the Nostrils

Tracking the nostrils is also done by iteratively thresholding the search-region and looking for 'legal' blobs. But whereas we have to search a relatively big area in the initial search, during tracking, the search-window can be positioned around the previous positions of the nostrils, and can be chosen much smaller. Furthermore, the initial threshold can be initialized with a value that is a little lower than the intensity of the nostrils in the previous frame. This limits the number of necessary iterations to be very small.

However, not always both nostrils are visible in the image. For example, when the head is rotated strongly to the right, the right nostril will disappear, and only the left one will remain visible. To deal with this problem, the search for two nostrils is done only for a certain number of iterations. If no nostril-pair is found, then only one nostril is searched by looking for the darkest pixel in the search window for the nostrils. To decide which of the two nostrils was found, we choose the nostril, that leads to the pose which implies smoother motion of the head compared to the pose obtained choosing the other nostril. The position of the other nostril can be predicted using the current estimated pose, as shown in Figure 3.

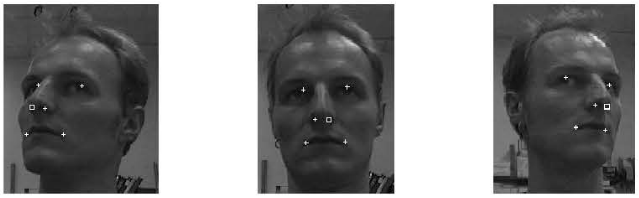


Figure 3. Predicted nostrils (marked with box)

Rejection and Prediction of Outliers

To increase the robustness as well as the accuracy of the system, we try to find outliers in the set of found feature points, and predict their true position in the next frame. At the same time, we use a most consistent subset of 2D to 3D point-correspondences to compute the pose, instead of using all found points. To find a best subset we investigated two methods proposed by Gee & Cipolla [12]: Sample consensus tracking and temporal continuity tracking. Using the first method, the subset is chosen that leads to the best back-projection of model-points into the image-plane. Using the second method, the subset that leads to the pose implying the smoothest motion is chosen as the best subset.

Once the best subset of features is found, the true position of an outlier can be easily predicted by projecting its model point into the image, using the computed pose. This prediction allows the system to recover from tracking errors and leads to a more robust tracking of the feature points.

Recovery from Tracking Failure

In order to build a robust usable tracking system, the system has to be able to detect tracking failure and to recover from it.

To detect tracking failure, the average distance between the back projected model points and their actual found locations in the image can serve as a measure of confidence. Once this average distance exceeds a certain threshold, tracking failure is considered. The system then searches the features again. However, if failure occurs during tracking, we cannot assume a frontal view of the face anymore, because failure could have occurred at any possible rotation of the head, and the initial search might not work anymore. This problem can be solved by initializing the search-windows and the geometric restrictions according to the previously found pose. For example, if failure occured, while the person was looking to the right, we then shift the search window for the eyes more to the right in the facial area, and more to the left, if the person was looking to the left.

Experimental Results

To evaluate the system we compared the output of the gaze tracker on some pre-recorded image sequences to the results obtained by labelling the facial features manually. The best results were obtained using the temporal continuity method [12], where we achieved rotation errors as low as 5 degrees for rotation around the x- and y-axis and as low as 1 degree for rotation around the z-axi. The average distance in x- and in y-direction of manually marked feature locations and the automatically found locations was between two and three pixels. The system runs with around 20 frames per second. See [8, 9] for complete results.

3. APPLICATIONS TO MULTIMODAL INTERFACES

Although gaze tracking techniques have existed for a long time, most applications of these techniques have been in psychological research for probing into subjects' perceptual or cognitive processes. Tracking the locations of the users and their gaze direction can provide additional helpful visual information to a user interface. In this section we discuss the applications of a gaze tracking system to multimodal human-computer interaction.

A gaze tracking system can be used in interfaces to create a faster and simpler communication between human and computer. A gaze tracking system can be used either actively or passively in an interface. An application of eye-gaze tracking would be for activating a window on a screen or directing inquiries. This application is similar to those techniques that have been in assistant devices for disabled people. An interesting research issue is how to improve the reliability of the gaze information. Even if a gaze tracker could provide high accuracy gaze information, gaze information alone is not reliable. For example, when a user sits in the front of a screen, he/she may look around randomly even though his/her focus of attention is at a certain window. A solution is to combine the gaze with other modalities to increase reliability. Another area where eye-gaze techniques potentially can be applied are in virtual reality and games. By using gaze tracking the user can view different scenes as he/she looks at different directions. Passive applications of gaze tracking system include monitoring users' eye-gaze pattern, blink rate and pupil size. The system can send an alert signal if an abnormal pattern would be detected.

Switching Language Model and Grammar

The performance of a speech recognition system depends on specific applications which reflect the constraints on the task. Different technologies are sometimes appropriate for different tasks. By limiting the vocabulary size and developing the language model and grammar for the task, we can obtain a high quality speech recognizer. However, a user may work on many different tasks at the same time. It is desirable to switch the language model and grammar automatically. This requires the system to have a method to detect user's status. One way to do this is to find out the user's focus of attention. Suppose that different tasks are running in different windows on a computer. User's gaze can reflect his/her attention. In order to increase reliability, we can use voice commands to confirm selections.

We have developed an interface to demonstrate the concept. We use gaze and voice to switch language models for Janus III recognition engine [13]. The Janus III system is at present specific to discourse domains of common interest, and supports spontaneously uttered human-to-human speech. Janus III was designed to be a speech recognition research tool. It has the ability to dynamically switch language models and grammar.It has its own object-oriented programming language implemented on top

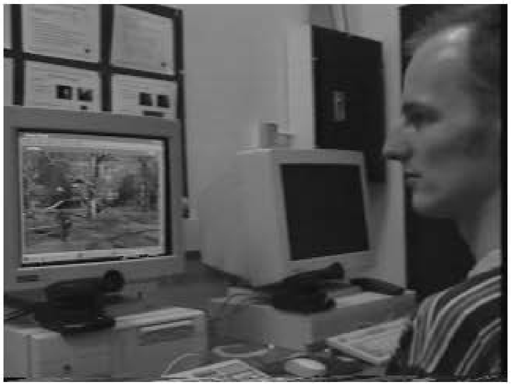


Figure 4. Controlling a panorama image viewer

of Tcl/Tk. This programming language allows researchers to do both, trigger powerful huge training processes with one single command and to control very low level features (down to single acoustic parameters) with simple commands. Tcl/Tk offers a user friendly environment with easy to implement GUIs. The gaze tracker is used to detect the user's focus of attention. When the user is looking at a window, the window will be highlighted. No action is taken unless the user uses a voice command to confirm the selection. The voice commands could be "select this window" or "close window", etc.. Once the selection is confirmed, the interface will send a command to Janus system to change the language model and grammar.

Viewing a Panorama Image

As another application, we developed a multimodal interface to control a panorama image viewer. A panorama image is made from photographs, video stills, or computer renderings. Most panoramas are made from photographs as they provide the most realistic images. The QTVR Player is a stand-alone application for Mac (or a component file for Windows) that lets you experience virtual reality scenes and objects from your desktop. The QTVR Player allows the user to scroll through 360 degree panorama images by using the mouse or keyboard, and to zoom in and out using the keyboard. In oder to make the user hands free, we have developed an interface that uses gaze to control scrolling through the panorama images, and voice-commands to control the zoom.

The interface receives parameters describing the rotation of the users' head from the gaze tracker and parameters for the spoken commands from a speech-recognizer. It then sends messages which simulate mouse- or key-events to the image viewer in order to control scrolling and zooming. The interface and the image viewer are running on a PC, gaze tracker and speech recognizer are running on workstations. Communication with the interface is done via sockets.

With such an interface, a user can fully control the panorama image viewer without using his/her hands as shown in Figure 4. He/she can scroll through the panorama images in a natural way by looking to the left and right or up and down, and he can control the zoom by speaking commands such as "zoom in", "zoom out" or "zoom in three times". The basic concept of this interface can be extended to navigate in a virtual environment where the surrounding then can be rendered according to the users' gaze.

4. CONCLUSION

We have addressed the problem of gaze tracking for multimodal human-computer interaction. A gaze tracking system can enhance human computer communication in many ways. We have demonstrated that a gaze tracker can be used to detect a user's focus of attention and driven an interface. The gaze information can improve the performance of the interaction made by other modalities. And other modalities can be used to increase the reliability of the gaze. The concepts developed in this paper can be applied to other applications such as virtual reality simulations.

REFERENCES

[1] H. Matsu'ura, Y. Masai, J. Iwasaki, S. Tanaka, S., H. Kamio, and T. Nitta, "A multimodal, keyword-based spoken dialogue system MultiksDial," Proc. ICASSP'94 (Adelaide, Australia, April 1994), Vol. 2, pp. II/33-36.

[2] S. Nakagawa and J.X. Zhang, "An input interface with speech and touch screen," Trans. Inst. Elec. Eng. Jpn. C (Japan), Vol. 114-C, No. 10, Oct. 1994, pp. 1009-1017.

[3] H. Ando, Y. Kitahara, and N. Hataoka, "Evaluation of multimodal interface using spoken language and pointing gesture on interior design system," Proc. ICSLP'94 (Yokohama, Japan, Sept. 1994), Vol. 2, pp. 567-570.

[4] M.T. Vo, R. Houghton, J. Yang, U. Bub, U. Meier, A. Waibel, and P. Duchnowski, "Multimodal Learning Interfaces," Proc. ARPA SLT Workshop 95 (Austin, Texas, Jan. 1995).

[5] A. Waibel, M.T. Vo, P. Duchnowski, and S. Manke, "Multimodal Interfaces," Artificial Intelligence Review, Special Volume on Integration of Natural Language and Vision Processing, McKevitt, P. (Ed.), Vol. 10, Nos. 3-4, 1995.

[6] D. A. Simon, M. Hebert, T. Kanade. Real-time 3-D Pose Estimation Using a High-Speed Range Sensor. International Conference of Robotics and Automation Proceedings, May '94, San Diego.

[7] Andrew Gee and Robert Cipolla. Non-Intrusive Gaze Tracking for Human-Computer Interaction. *Proc. Mechatronics and Machine Vision in Practise, p. 112-117, Toowoomba, Australia, 1994*

[8] Stiefelhagen, R., Yang, J., Waibel, A.: A Model-Based Gaze Tracking System. *Proc. of the IEEE International Symposia on Intelligence and Systems, p. 304-310, Nov. 1996.*

[9] Stiefelhagen, R. Gaze Tracking for Multimodal Human Computer Interaction. University of Karlsruhe, 1996. Available at http://werner.ira.uka.de/ISL.multimodal.publications.html.

[10] D. F. DeMenthon and L. S. Davis. Model based object pose in 25 lines of code. In G. Sandini, editor, *Computer Vision - ECCV 92, Proceedings Second European Conference on Computer Vision, Santa Margherita Ligure, May 1992*, pages 335 - 343. Springer Verlag, May 1992.

[11] J. Yang and A. Waibel, "A real-time face tracker," Proceedings of WACV'96 (Sarasota, Florida, USA), pp. 142-147, 1996.

[12] A. H. Gee and R. Cipolla. Fast Visual Tracking by Temporal Consensus. *Technical Report CUED/F-INFENG/TR-207, University of Cambridge, February 1995*

[13] T. Zeppenfeld, M. Finke, K. Ries, and A. Waibel, "Recognition of conversitional telephone speech using the janus speech engine," Proc. of ICASSP'97.