Gesture-Based Interaction with a Pet Robot

Milyn C. Moy

MIT Artificial Intelligence Lab / Oracle Corporation
500 Oracle Parkway
Redwood Shores, CA 94065
milyn@ai.mit.edu

Abstract

Pet robots are autonomous robots capable of exhibiting animal-like behaviors, including emotional ones, as they interact with people and objects surrounding them. As pet robots become more integrated into our lives, a more natural way of communicating with them will become necessary. Similarly, they will need to understand human gestures in order to perceive our intentions and communicate with us more effectively. In this paper, we present an extensible, real-time, vision-based communication system that interprets 2D dynamic hand gestures in complex environments. Our strategy for interpreting hand gestures consists of: hand segmentation, feature extraction, and gesture recognition. To segment the hand from the cluttered background, this system uses both motion and color information. The location of the hand is subsequently tracked as the user makes the gesture and its trajectory information is stored in a feature vector. Finally, the gesture is interpreted using this vector and translated into a command that the robot understands. We implemented our system on Yuppy, a pet robot prototype. Currently, via an external microcamera, we can navigate Yuppy in unstructured environments using hand gestures.

Introduction

As pet robots become more integrated into our everyday lives, it will become essential for them to perceive and understand our intentions and actions. We will also want to communicate with them as we do with other human beings. Yet, to communicate and interact with robots, we are still required to use specialized input devices such as keyboards, mice, trackers, or data gloves (Zimmerman & Lanier 1987). Thus, a more natural, contact-less interface would be desirable to avoid the need for external devices.

An example of such an interface is speech (Huang, Ariki, & Jack 1990). However, when we communicate with each other, we also use gestures, facial expressions, and poses as supplements or substitutes for speech. Clearly, pet robots would perceive our intentions and communicate with us more effectively if they were able to interpret human gestures and body language.

One of the most expressive components of body language is hand gesture. We use our hands to explore objects and to express our ideas and our feelings. Thus, hand gestures provide a very natural interface for us to communicate with robots.

Recently, there has been a significant amount of research on hand gesture recognition. The two most common approaches are model-based and feature-based. Model-based approaches assume a physically valid model of the hand and attempt to recognize a gesture as a variation of the hand articulation, using techniques such as template matching (Heap & Hogg 1996; Rehg & Kanade 1994) and neural networks (Nowlan & Platt 1995). Similarly, a 3D model of a human has been used to guide stereo measurements of body parts for human-robot interaction (Kortenkamp, Huber, & Bonasso 1996). Model-based approaches generally suffer from being computationally expensive due to the need for high resolution images to ensure accurate segmentation, and the need to cope with variations in intensity, scale, orientation, and deformation.

Feature-based approaches do not assume a 3D model of the hand but instead use low-level image features. Some examples are the use of Hidden Markov Models (HMMs) (Yamato, Ohya, & Ishii 1992), a view-based approach with dynamic time warping (Darrell & Pentland 1993), and the trajectory analysis of hand gestures coupled with speech recognition for controlling a mobile robot (Perzanowski, Schultz, & Adams 1998) among others. These methods may be more suitable for real-time gesture recognition since low-level image features can be computed quickly, but they require an effective segmentation of the hand from the input images.

Accurate segmentation of the hand from the background is difficult. For this reason, some researchers have controlled the imaging conditions using special backgrounds (Darrell & Pentland 1993; Rehg & Kanade 1994), static backgrounds with background subtraction (Appenzeller, Lee, & Hashimoto 1997; Dang 1996), or special markers or colored gloves (Starner, Weaver, & Pentland 1998; Hienz, Grobel, & Offner 1996) to simplify the segmentation task. However, these systems are not flexible enough for most real world applications.

A combination of motion and color detection has been used (Appenzeller, Lee, & Hashimoto 1997; Kahn *et al.* 1996) to eliminate these constraints. Limitations still exist in the their use because they are prone to also detect shadows of moving objects and similarly colored objects. But, by using these two cues together, the false-positives are considerably reduced, making this a more flexible approach for hand segmentation.

This paper focuses on the real-time, visual interpretation of 2D dynamic hand gestures in complex environments. Our goal is to enable humans to communicate and interact with a pet robot in a more natural fashion. The scope of this work is not to create a full lexicon system that will facilitate this interaction in every possible way, but to provide an extensible mechanism to enable the recognition of new gestures.

The next section of this paper presents an overview of the system. The following sections explain the different modules of our system: hand segmentation, feature extraction, and gesture recognition. Then, the results of our experiments are discussed. Finally, the conclusion and the future work are provided in the last section.

Overview

The test bed for this system is an emotional pet robot called Yuppy (Figure 1), currently being developed at the MIT Artificial Intelligence Laboratory. This robot is built on top of a 12 inch synchrodrive base and is equipped with various sensors to increase its awareness of the environment and allow for better interaction with humans. Although this robot is meant to be fully autonomous, all perception and control code for this work temporarily run off-board on a 300 MHz Pentium Pro workstation. For development purposes an external and uncalibrated, wide-angle lens Chinon CX-062 color microcamera was used, similar to the ones on Yuppy.

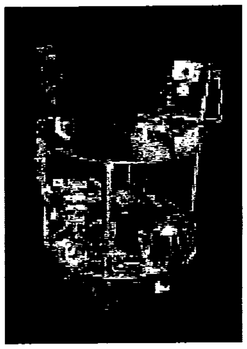

Figure 1: Yuppy: An emotional pet robot

A precise "language" with a grammar is needed in human to human communication and some researchers have worked on recognizing American Sign Languages (Starner, Weaver, & Pentland 1998; Dorner 1993). Nevertheless, a small set of gestures that is easy to use and understand will suffice to specify commands to pets. The initial goal of this system is to enable humans to direct the robot using hand gestures; thus, the lexicon chosen consists of a simple set of basic 2D gesture classes or primitive gestures (Figure 2) that can be done with a single hand in the direction of the arrow. They include linear (vertical, horizontal, and diagonal) as well as circular (clockwise and counterclockwise) gestures. Each gesture class allows for variations in the speed of the hand's motion, different ranges of hand motions, and small deviations (orientation, translation) in the trajectory of the hand's motion.

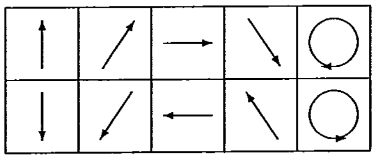

Figure 2: Primitive Gestures

Our strategy for interpreting hand gestures consists of the following three phases:

1. Segment the gesturing hand from the input images using motion and color information and track it in real-time.
2. Store the hand's trajectory information in a feature vector.
3. Interpret the user's gesture using the feature vector and issue a command for the robot to perform the desired task.

We begin with a detailed discussion of the hand segmentation, followed by the feature extraction and gesture recognition phases.

Hand Segmentation

Hand gesture recognition requires the ability to accurately segment the hand from the cluttered background. To achieve real-time performance and robustness to complex backgrounds, we use a segmentation algorithm based on motion and color information. We currently assume that users wear long-sleeved clothes and the robot is static while observing the gesture.

Motion Detection

The most common method for detecting motion is optical flow, which estimates the relative motion by means of temporal variations of the brightness pattern (Horn 1986). However, this method is slow in the absence of specialized hardware.

Another common alternative is to use image differencing. This method determines the edges of the moving components, and can be computed very fast by taking the difference between corresponding pixel values of two subsequent images and selecting the pixels that pass a certain threshold. However, it also includes in the result pixels of the background which were covered in the previous frame. To improve this method, we use the image difference of three sequential images (I_{t-2}, I_{t-1}, I_t) by taking the pixelwise difference between the RGB (Red, Green, Blue) representations of the last two and the first two images and selecting the pixels that satisfy the following condition:

$$|I_t - I_{t-1}| > \theta \text{ and } |I_{t-1} - I_{t-2}| > \theta$$

where I_t is an image frame taken at time t and θ is a threshold. This method can fail in the presence of shadows because they move with the entities that create them, but we solved this problem using color.

Color Detection

The use of color as a basis for skin segmentation has several advantages. First, human skin colors tend to cluster in color space (Yang, Lu, & Waibel 1997) due to their richness in red tones. Second, color is faster to process compared to other hand features. Third, it is orientation invariant in the 2D image plane, assuming that the image intensity is invariant between adjacent image frames. Despite these advantages, deviations in the color representation can occur due to changes in the lighting conditions, specular reflections, motion, and nonlinearities introduced by the camera and the frame grabber. However, by choosing the right color space, these deviations can be reduced.

Simple RGB and normalized RGB ($r = \frac{R}{R+G+B}$, $g = \frac{G}{R+G+B}$, $b = \frac{B}{R+G+B}$) thresholding are commonly used methods (Scheile & Waibel 1995), but they are not very robust to different skin tonalities and lighting conditions. We use the HLS (Hue, Lightness, and Saturation) color model, a linear transform of the RGB space. By disregarding the lightness component and using appropriately defined domains of hue and saturation, robustness to changes in illumination and shadows can be achieved (Saxe & Foulds 1996).

To obtain bounds for the hue and saturation of human skin color we used an a priori model of skin color. We plotted different skin tonalities under different lighting conditions in HS space. Under any given lighting condition, a skin-color distribution can be characterized by a multivariate normal distribution (Yang, Lu, & Waibel 1997). Thus, the thresholds for hue and saturations $(H_{min}, H_{max}, S_{min}, S_{max})$ were obtained by determining an area (typically two standard deviations) around the mean values of H and S. For efficiency in computation, a simple square representation in HS space is used; thus, a pixel or color pair (h, s) in this space is skin-colored if it passes the following condition:

$$(H_{min} < h < H_{max}) \text{ and } (S_{min} < s < S_{max})$$

As the gesture is made, a sequence of RGB images (100x100 pixels) is taken by the frame grabber and processed in real-time on a frame by frame basis. We currently run our algorithms in a user-specified window of attention which excludes the user's face.

In our system, motion detection is computationally less expensive than skin color detection, so it is applied to the images first, to constrain the search for skin color to moving regions only. Initially three images (I_{t-2}, I_{t-1}, I_t) are obtained from the frame grabber and processed independently by both the motion and the skin-color detectors. Motion is detected by taking the image difference of these three images. The result is stored in a binary image, I_m. To speed up the skin-color detection, we also compute a bounding box that surrounds all moving components.

The HLS representation of I_{t-1} (this image reflects the moving components as a result of the image differencing) is then used to determine the skin-colored patches inside the bounding box. The result is stored in a binary image, I_c, where 1 indicates skin color and 0 other colors. This binary image is smoothed and an 8-neighbor connected components algorithm is used to cluster each skin-colored region.

After the initial processing on the input images, the results of the motion and skin-color detection are combined by superimposing I_c on I_m. For each skin-colored region in I_c, a bounding box is placed around the moving pixels, to obtain a moving skin-colored region. The regions with very few moving pixels (noise) are discarded because they are usually similarly colored objects (e.g. a pink chair, a light-pink phone) in the image's background. The others are passed through function f which segments the hand based on the size of the skin-color region and the amount of motion within it:

$$f(moving\_pixels, skincolored\_pixels) =$$
$$c_1 * moving\_pixels + c_2 * skincolored\_pixels$$

where c_1 and c_2 are constants that determine the weight each detector is given in segmenting the hand. This function allows the flexibility of adding new detectors to the system as well as using probability instead of a simple score to segment the hand.

The hand is chosen to be the region with the highest score (i.e. the skin-colored region that has the largest area and the greatest number of displaced pixels). The hand's centroid is determined and its motion is tracked in real-time as new image frames are processed. If little motion is detected or the area of the moving skin-colored region with the highest score is very small, the previously computed centroid is reused. Similarly, if the Euclidean distance between the hand's centroid of the previous image frame and the current centroid does not exceed a certain threshold, the current centroid is considered the new location of the hand; otherwise, the previous centroid is kept.

An illustration of the hand segmentation is shown in Figure 3. Image (a) shows the last image frame taken by the camera as the person makes the gesture in front of the robot. The bounding box delimits the attention window where gestures are recognized. Image (b) shows the results of the image differencing and the bounding box surrounding the moving components. Image (c) shows skin-colored areas inside the bounding box, including the pink chair in the background. Image (d) shows all the moving skin-color regions. The gesturing hand is selected from among these regions and displayed in image (e). Finally, a crosshair is drawn on the centroid of the hand in image (a).

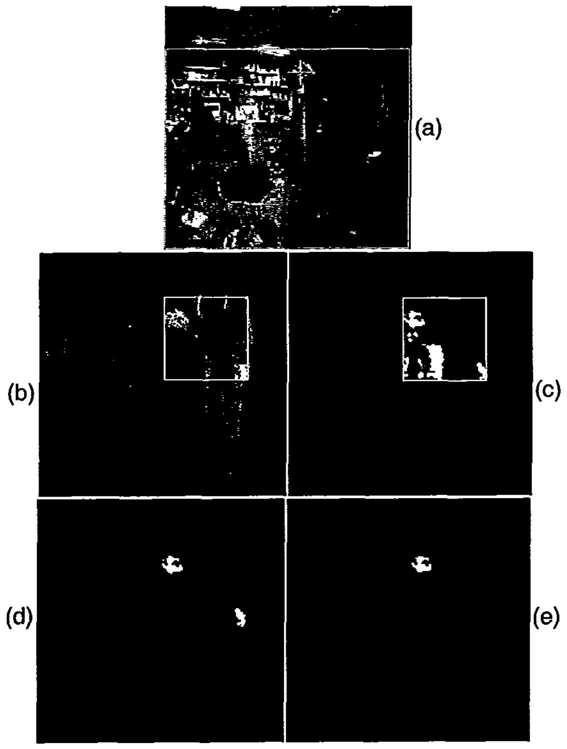

Figure 3: Hand segmentation

Feature Extraction

The information about the hand's trajectory needs to be encapsulated in a representation that can be used to identify the appropriate gesture. To achieve this, we need to first determine the beginning and the end of a gesture. The capability of determining when a gesture ends and a new one begins is called gesture segmentation and is a major problem in gesture recognition.

Different approaches have been taken by the research community to solve the segmentation problem. External devices have been used (e.g. clicking and releasing a mouse button (Rubine 1991)), but the need for these devices is cumbersome and unnatural. A fixed starting posture for each gesture (Baudel & Beaudouin-Lafon 1993) is not scalable because every gesture needs to have a different starting posture. The use of hand tension was proposed by (Harling & Edwards 1996), but it does not work on a sequence of dynamic gestures. This problem can be solved by using HMMs given that segmentation between gestures is done during the recognition process. Nevertheless, HMMs are very difficult to use because of their demanding training phase.

Our system assumes that once the hand's motion exceeds a certain velocity, the person has started a gesture. As the hand moves, the vertical and horizontal displacements (dx, dy) of the hand's centroid in the image space are calculated and stored in a feature vector until the hand stops moving for a short period of time (usually 2-3 seconds).

Gesture Recognition

In human communication, gestures are accompanied by attitudes and emotions. Thus, being able to recognize these attributes along with the type of gesture made would be desirable. We provide a gesture recognition system that is capable of interpreting a very important gesture attribute – its speed.

Gesture interpretation is achieved by analyzing the feature vector created from the hand's trajectory. In the case of linear gestures, the (dx, dy) displacements cluster around fixed axes in the *dx*-*dy* plane: vertical gestures around the *dy* axis, horizontal gestures around the *dx* axis, and diagonal gestures around the two bisecting axes (45° with respect to the *dx*-*dy* axes). The direction of motion is determined by the side of the axis (positive/negative) on which clustering occurs (e.g., vertical upward gestures have (dx, dy) clustering around the positive side of *dy*, and vertical downward gestures cluster around the negative side of the *dy* axis). Hence the (dx, dy) displacements for the 8 linear primitive gestures cluster in 8 distinct regions of the plane. The centroid of such a cluster, along with the origin of the *dx*-*dy* plane, determines a velocity vector whose magnitude indicates the speed of the gesture.

For circular gestures, however, the (dx, dy) displacements are spread in the plane. If the centroid of these displacements coincides with the origin, we conclude that the gesture is circular. The direction of motion (clockwise/counterclockwise) is deduced from the time sequence of the displacements in the feature vector.

First, the feature vector is scanned and any element that contains the same set of signs for both *dx* and *dy* as the previous element in the vector is discarded. The result of this operation is that neighboring elements in the vector have different signs for *dx* and *dy*. Second, the signs of every sequence of four (dx, dy) pairs are compared with a time-sequence model for the clockwise motion (– + , + + , + – , – –) and counterclockwise motion (+ + , – +, – – , + –). The sequences reflect the hand's trajectory in time on the *dx*-*dy* plane. The direction of the circular gesture results from the time-sequence model that matches the feature vector's information.

Each primitive is assigned a label that distinguishes it from other primitives. After determining the basic gesture made by the user through the analysis of the feature vector, this information is represented in a more compact way, using a descriptor. The descriptor of a gesture is an array that contains as its only element the label of the identified primitive.

Model gestures are stored in a database containing user-specified model gestures and their corresponding meanings or commands for the pet robot (e.g., a circular clockwise motion commands the robot to rotate in place clockwise). Gestures are stored as descriptors in the database. Once the descriptor produced from the user gesture is obtained, it is queried in the database. If such a gesture was found in the database, the command associated with the gesture is returned and issued to the robot; otherwise, the user gesture is ignored.

The simple gesture classes introduced above can be combined to create new ones, allowing for an extensible hand gesture recognition system. To segment these different primitives in one gesture, a hand pause is used between primitives. This pause is shorter than the one used for segmenting different gestures. When the gesture is composite, the feature vector contains the displacements of the hand's centroid for each primitive, separated by a marker. The descriptor for such a gesture will contain sequentially the different labels for each primitive in the composite gesture. This flexibility allows new gestures containing any number of primitives with new different meanings to be created and stored in the database as needed.

Experimental Results

To assess the accuracy of our system, we performed an initial experiment where 14 subjects with different skin tonalities (Asians, Caucasians, Hispanics, and Indians; Black subjects were not available) were asked to perform 5 times a sequence of 16 gestures shown in Figure 4. The environment was a crowded lab with a variety of objects spread throughout.

The accuracy rate obtained was above 90% for each primitive gesture class and slightly above 70% for composite gestures. These results demonstrate the viability of our system for unstructured environments and its robustness to different skin tonalities and varying lighting conditions. We used a commodity 300 MHz Pentium Pro system with a Matrox Meteor frame grabber, and achieved 15 frames per second (this includes processing and redrawing the frame on the computer screen). Although this speed is sufficient, higher performance can be obtained using faster hardware.

A variety of reasons explain why the obtained accuracy was not higher. First, we had slight distortions of diagonal motions due to the camera tilt, as well as tracking failures when the hand left the camera's field of view and when other bare parts of the user's body exhibited a lot of motion. Second, some subjects erred when making diagonal motions or composite gestures. In other cases, the beginning of gestures was not detected when subjects started the motion too slowly. Third, the recognition success for composite gestures is geometrically dependent on the recognition of each one of its primitives. Lastly, the lower accuracy for composite gestures is largely due to them being less natural for humans, making it difficult for subjects to execute them correctly.

Gesture		Accuracy (%)	Gesture		Accuracy (%)
↑	front	100.0	↻	rotate-right	90.0
→	right	97.1	↺	rotate-left	95.7
↙	left-back	100.0	↑↓	calm-down	91.4
↗	right-front	98.6	∞	dance	80.0
↓	back	100.0	⇄	hi	82.9
↖	left-front	97.1	□	make-square	72.9
↘	right-back	95.7	△	make-triangle	71.4
←	left	100.0	Ω	make-omega	80.0

Figure 4: Gestures accuracy$=\frac{\#\ of\ correct\ gestures}{\#\ of\ total\ gestures}$=(14×5)=70

Some of these problems are not practical concerns given that, when humans interact with a pet robot, they tend to be more cooperative. Humans position themselves close enough to the robot so that it can pay attention to them. They also make the gestures carefully, for the robot to understand them, and use the robot's behavior as feedback for improving the way they make the gestures. During the experiment, we observed how the gesture recognition's accuracy for most subjects improved as they obtained visual feedback.

Our system already allows people to interact with Yuppy in a more natural fashion. Yuppy's current interface consists of an off-board camera, via which any person can easily navigate the robot around the lab using each primitive gesture as a command (e.g., an upward hand motion commands the robot to go front, a clockwise circular motion commands the robot to rotate in place clockwise).

It is important for the robot to interpret correctly what the gestures mean and be able to respond to them accordingly, but 100% accuracy in gesture interpretation is not really necessary. Even communication between humans is sometimes ambiguous, and we expect the communication between humans and animals to be even more so. People expect objects to react immediately, predictably and consistently; however, they are more tolerant with humans and animals. Humans can accept that a pet robot might not have perceived or correctly interpreted the request; they expect and even prefer unpredictable behaviors from their pets.

Conclusion

We have presented a starting point toward understanding how vision can be used to recognize human gestures and provide a natural interface to enhance human-robot communication. Our system explored the use of fast algorithms and simple features to determine the hand's trajectory in real-time using commodity hardware.

Our system was tested on Yuppy, an emotional pet robot, and with the help of an off-board camera we were able to navigate the robot in unstructured environments using hand gestures. Initial evaluation of this system resulted in above 90% accuracy for recognition of single primitive gestures and above 70% for recognition of composite gestures, demonstrating the viability of our approach. But, we believe that better methods should be devised to further evaluate this behavioral system.

This work provides a basic interface for future behaviors implemented on Yuppy, such as approaching a person, searching for a bone, fetching the newspaper, etc. Future work will address additional competency in reliably differentiating the hand from other moving parts of the body and continuously tracking the motion of the hand, coping with simultaneous motion of both the robot and the human, and supporting simultaneous interaction of multiple people with the robot. A more general gesture recognition system would include the interpretation of hand poses and 3D gestures. We are also interested in gesture learning, the robot's reaction to both what it perceives and how it feels, and the interpretation of humans' attitudes and emotions implicit in their gestures.

Acknowledgments

Support for this research was provided by Yamaha Corporation under contract No. 65539. The author would like to thank Rodney Brooks for his continued support.

References

Appenzeller, G.; Lee, J.; and Hashimoto, H. 1997. Building topological maps by looking at people: An example of cooperation between intelligent spaces and robots. In *Proceedings of the IEEE-RSJ International Conference on Intelligent Robots and Systems.*

Baudel, T., and Beaudouin-Lafon, M. 1993. Charade: Remote control of objects using free-hand gestures. *Communications of the ACM* 36(7):28–35.

Dang, D. 1996. Template based gesture recognition. Master's thesis, MIT.

Darrell, T., and Pentland, A. 1993. Space-time gestures. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 335–340.

Dorner, B. 1993. Hand shape identification and tracking for sign language interpretation. In *Proceedings of the IJCAI Workshop on Looking at People*, 75–88.

Harling, P., and Edwards, A. 1996. Hand tension as a gesture segmentation cue. In *Proceedings of the Progress in Gestural Interaction.*

Heap, T., and Hogg, D. 1996. Towards 3D hand tracking using a deformable model. In *Proceedings of the International Conference on Automatic Face and Gesture Recognition*, 140–145.

Hienz, H.; Grobel, K.; and Offner, G. 1996. Real-time hand-arm motion analysis using a single video camera. In *Proceedings of the International Conference on Automatic Face and Gesture Recognition*, 323–327.

Horn, B. 1986. *Robot Vision.* MIT Press.

Huang, X.; Ariki, Y.; and Jack, M. 1990. *Hidden Markov Models for Speech Recognition.* Edinburgh University Press.

Kahn, R. E.; Swain, M. J.; Prokopowicz, P. N.; and Firby, R. J. 1996. Gesture recognition using the perseus architecture. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.*

Kortenkamp, D.; Huber, E.; and Bonasso, R. P. 1996. Recognizing and interpreting gestures on a mobile robot. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 915–921.

Nowlan, S., and Platt, J. C. 1995. A convolutional neural network hand tracker. In *Proceedings of Neural Information Processing Systems*, 901–908.

Perzanowski, D.; Schultz, A.; and Adams, W. 1998. Integrating natural language and gesture in a robotics domain. In *Proceedings of the IEEE International Symposium on Intelligent Control: ISIC/CIRA/ISIS Joint Conference*, 247–252.

Rehg, J., and Kanade, T. 1994. Visual tracking of high DOF articulated structures: An application to human hand tracking. In *Proceedings of the Third European Conference on Computer Vision*, 35–45.

Rubine, D. 1991. Specifying gestures by example. *Computer Graphics* 25(4):329–337.

Saxe, D., and Foulds, R. 1996. Toward robust skin identification in video images. In *Proceedings of the International Conference on Automatic Face and Gesture Recognition*, 379–384.

Scheile, B., and Waibel, A. 1995. Gaze tracking based on face color. In *Proceedings of the International Workshop on Automatic Face and Gesture Recognition.*

Starner, T.; Weaver, J.; and Pentland, A. 1998. A wearable computer based american sign language recognizer. In *Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence.*

Yamato, J.; Ohya, J.; and Ishii, K. 1992. Recognizing human action in time-sequential images using hidden markov models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recongition*, 379–385.

Yang, J.; Lu, W.; and Waibel, A. 1997. Skin-color modeling and adaptation. Technical Report CMU-CS-97-146, Carnegie Mellon.

Zimmerman, T., and Lanier, J. 1987. A hand gesture interface device. In *ACM SIGCHI/GI*, 189–192.