The WristCam as Input Device

Andrew Vardy, John Robinson, Li-Te Cheng
Multimedia Communications Laboratory, Faculty of Engineering and Applied Science
Memorial University of Newfoundland, St. John's, NF, Canada
vardy@engr.mun.ca, john@engr.mun.ca, lcheng@engr.mun.ca

Abstract

We show how images of a user's hand from a video camera attached to the underside of the wrist can be processed to yield finger movement information. Discrete (and discreet) movements of the fingers away from a rest position are translated into a small set of base symbols. These are interpreted as input to a wearable computer, providing unobtrusive control.

1 Introduction

We are investigating the use of a wrist- or forearm-mounted camera for wearable computer applications. As shown in Figure 1, the WristCam attaches to the underside of the forearm and views only the front of the hand and fingers. Typically it is invisible to other people, being concealed by the wearer's sleeve and connected up the sleeve to the remainder of the system.

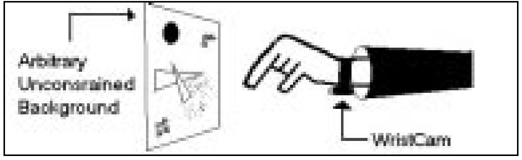

Figure 1. Camera position and expected background conditions

Foreseen applications for the WristCam include a hand's-eye-view key frame recorder (useful for storing sequences of gestures like keystrokes on a dial pad), hand scanning of documents, and active vision tasks that can combine input from a static head camera and a dynamic wrist camera. However, our main interest is to develop a virtual input device that allows chorded character input by tracking certain coarse finger movements or gestures. To do this, we need to track and interpret finger motions as seen by the WristCam, in the wide range of illumination conditions and backgrounds encountered as the user moves their hand around. Typically, the user will tap fingers against a handy table top or their clothing, or simply move their fingers freely. Recognizing these motions and converting them efficiently to system input makes it possible for WristCam users to interact with their wearable computers unobtrusively. Although head-mounted displays are becoming less encumbering and obvious to others (non-wearers), existing input devices such as chord keypads are very visible and distracting to others. The WristCam will allow input to be subtle (or even covert) and therefore not distracting. Also, when the system is not in "input" mode the user's hands will be instantly free for other uses.

The basic system operation is as follows: A process on the wearable computer is dedicated to interpreting the movements of the user's fingers to determine if the user is attempting to communicate. If the process determines that the user's fingers are in a predefined "start" position then the communication will begin. From this point on, movements of particular fingers and combinations of fingers away from the start position will correspond to individual symbols. These symbols may themselves comprise direct commands, or subsequent symbols may be aggregated together to form "words". In this manner, the user can send not just rudimentary commands such as "UP", "DOWN", and "CONTINUE", but also regular text.

The same system could also be employed to receive a continuous range of input values from the user's hand to use as a control mechanism for some continuous-motion type virtual device such as a scroll bar. This, however, is not the focus of this paper.

2 Methodology

We first identified a set of general requirements for a WristCam input device. Then, for an initial prototype, we established a simple syntax for communication using finger gestures. This syntax drove the detailed design and was used for the testing reported here.

General requirements for a WristCam input device are: robustness, speed, ease-of-use, and covertness.

The system must operate with a very high degree of reliability in a wide variety of environments. Thus, the background and illumination remain unconstrained. A wearable computer user is likely to be found in virtually any environment and should not be expected to hold themselves frozen in front of a featureless background. Thus, the system must work when presented with an arbitrary moving background.

Clearly, an input device must be able to accept user input in real-time. A slight delay may be acceptable, but if the system is not capable of almost immediately responding to a rapid chain of gesture inputs then it will fail as a general-purpose input device.

The system must be easily usable. Gestures corresponding to “inputs” to the wearable computer must be comprised of simple comfortable gestures. This constraint, in conjunction with the robustness constraint, also implies a certain degree of latitude in terms of camera positioning. The system should be resilient to changes in camera position and orientation. It is unrealistic to expect a wrist-mounted camera to stay perfectly rigid and stable.

The covert operation requirement is borne largely from the desire to utilize this system in a social context. Finger movements corresponding to system input symbols must be fairly subtle and unobtrusive.

A simple base alphabet of seven possible gestures was derived from the eight possible combinations of three fingers with each finger being in a state of movement from the rest position. Figure 2 illustrates this alphabet.

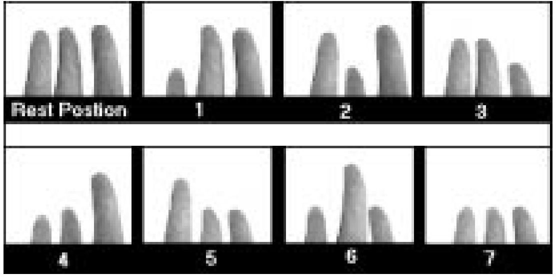

Figure 2. Space of possible 3-finger gesture positions

The static gesture positions shown above do not themselves constitute gestures. We define a gesture as a dynamic change from the rest position to one of the gesture positions, and then back to the rest position. Note that the rest position has the fingers extended, and that the actual gestures are finger lifts. This allows the hand to be rested on a flat surface and taps to be interpreted as gestures. The three-finger alphabet was used because of the relatively high dexterity and close physical proximity of the first three fingers. Also, these three fingers tend to appear parallel to a camera mounted on the underside of the wrist. The system was designed to sense this small set of gestures.

3 System Operation

The system as a whole is a state machine. The following sequence of steps provides a simplified description of normal system operation:

1. The user initiates finger tracking by turning the wrist inwards to blacken the camera’s input.
2. The user now places their hand motionless in clear view of the camera. An audible tone will notify the user if the current hand position is unsuitable.
3. The user then begins gesturing. The system is returned to idle by blackening the camera's input.

4 High-Level Design

Two key features of the expected hand-viewing conditions have been exploited in this system. The first is that the hand normally takes up most of the bottom of the image area. Due to the often complex distribution of skin color in color space [1-3], we will find that selection of the “most common color” will often not yield good segmentation of the hand image. However, the assumption that “skin color”, however it is defined, is relatively frequent in the source image underpins the approach discussed below.

The second feature to exploit is the parallel nature of the fingers. From the camera’s vantage point at the underside of the user’s wrist, the fingers, having been bent inwards, all appear approximately parallel in the source image. This suggests converting the source image into a one-dimensional representation according to the average finger angle.

The two motivational assumptions given above provide the basis for our high-level design.

We map the color coordinates of each pixel of the incoming video stream into a 3-dimensional (3 color components) quantized color histogram. We find points of maximum density in the histogram and treat these as candidate skin colors. The learned skin color (the prototype) from previous frames is compared with the candidates and the prototype is incrementally updated towards the most likely current skin candidate. The rate at which the prototype changes is under control of a parameter, which is set so that movement, shading and illumination effects, such as movement of a single finger, that cause transient changes in the histogram, do not cause the prototype to jump. The consequence of this would be a sudden difference in segmentation not only for the finger that moved, but for all of the other fingers as well, thus masking the appearance of a moving finger.

The method used to divide the image into skin and non-skin pixels is a linear color segmentation algorithm developed specifically for real-time skin finding applications. The segmented image is then filtered. Figure 3 illustrates the result of the segmentation process.

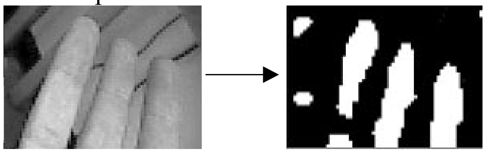

Figure 3. Image segmentation

We now return to the original image and attempt to determine in which direction the fingers are pointing. The parallel assumption allows us to take this as a single direction. We find the angle-to-vertical of the single line which is most closely parallel to the fingers in the source image. We first obtain a thinned contour image from the original bitmap. Next, a sequence of bins is created where each bin corresponds to a range of angles between –45° and +45°. Each contour point with an angle in this range is placed in the appropriate bin. The largest bin is found and the average of all its contour points is taken. This average is taken as the desired angle. Some inertia is required in this quantity in order to stabilize the process. Clearly, this is a simple process. It is also robust and very fast.

Once the angle of fingers is found, we can reduce the source image to the more compact representation desired. The 2-D segmented image is transformed into a 1-D histogram by taking each point in the histogram as the number of skin-color pixels in the segmented image counted by summing along slanted columns of the segmented image parallel to the finger angle line. This histogram is then smoothed. Figure 4 illustrates the histogram constructed from the image above.

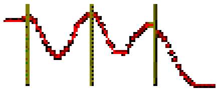

Figure 4. Smoothed image histogram

The histogram is analyzed to determine the heights and locations of the peaks. Each peak should ideally correspond to a single finger. However, in order to combat against temporary periods of poor image segmentation, and thus a poor histogram, three horizontal points along the histogram are assigned to each of the three fingers. If the correct number of peaks are present (i.e. three) then the three horizontal points will be moved incrementally towards those positions. If the number of peaks in the histogram is not correct then the three horizontal points will stay stationary unless they are in the immediate vicinity of a peak in the histogram, in which case they will be moved incrementally towards such a peak.

At this point we have three horizontal points that correspond to three of the slanted columns used to transform the segmented image into the image histogram. These three columns are parallel lines that should roughly bisect each of the fingers in the original image. Thus, three lines are extended from the top of the image down along these three slanted columns. The lines extend down to the point where the fingertips are encountered. This point is determined when a threshold number of skin-colored image pixels are found on a short line (just a few pixels long) called the "test" line which is perpendicular to the downwards extended line and which is fixed at its center to the bottom of that line. Figure 5 shows a close-up of the point at which a fingertip is "found".

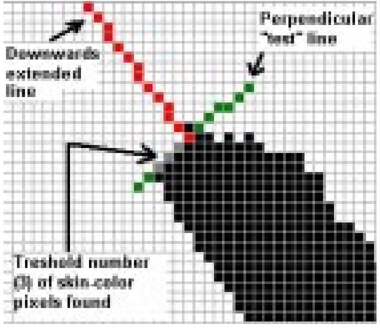
Downwards extended line
Perpendicular "test" line
Treshold number (3) of skin-color pixels found

Figure 5. Close-up of "test" line encountering fingertip

The quantity produced by this process is simply the length of the downwards extended line from the top of the image until the point where it encounters the fingertip. Tracking these three quantities (one for each finger) over time generates a third set of histograms. Each of these histograms corresponds to a single finger. To stabilize the process these histograms are continuously smoothed. Also, they are inverted so that peaks become valleys. Finally, by looking for peaks in these histograms we can determine if a finger has moved significantly from the rest position. At this point the finger(s) being moved describe the input symbol. Note that it is necessary to introduce a small delay upon finding a peak in one of the histograms. This delay allows time for another peak in a different histogram to "arrive". That is, in a multi-finger gesture we allow the movements of each moving finger to peak slightly out of phase.

5 Results

We present some numerical results of trials of this system attempted in a variety of environments. The first two environments shown are relatively ideal. A plain background was used and care was taken in the positioning and orientation of the fingers. The other environments employed arbitrary backgrounds and less constrained finger positioning. It should be stated that the results for each new environment were taken after the new position had stabilized (i.e. the camera was allowed to self-adjust its brightness, the finger orientation angle was allowed to settle, and the three histogram peaks were reliably detected). In each environment each of the seven gestures were attempted (see Figure 2). The following table gives the system response to these seven gestures:

Table 1. Tabulated results in seven settings

Setting	Correct	Right time – wrong gesture	Gestures missed entirely	Non-existent gestures
Ideal	7	0	0	0
Ideal	7	0	0	1
Non-Ideal	7	0	0	1
Non-Ideal	5	2	0	1
Non-Ideal	6	1*	0	0
Non-Ideal	7	0	0	0
Non-Ideal	6	1*	0	1
Average	6.43	1.33	0	0.57

* - In these cases a multiple-finger gesture was intended, yet the system interpreted these as two single-finger gestures.

The results for each setting are single trials. No system parameters were adjusted between the tests shown above.

The average delay between the actual completion of a gesture and the time that the system indicates that a gesture has occurred is approximately one second. The average rate of processing is approximately 9 frames per second on a Pentium II 400 MHz PC with 64MB of RAM and a Vista ViCam Digital Camera.

6 Discussion

The results given above are for a single user making single, well-separated, gestures. Although they do not characterize the performance of the system in decoding continuous natural movement, they demonstrate the reliability of the underlying image processing algorithms. They show that correct operation can be reliably achieved if the user is careful about the background seen by the camera. However, in the case of an arbitrary and moving background, i.e. in the context specified by the design requirements, the system exhibits errors. Primarily, these occur when portions of the background are misclassified as skin. Such areas often display frequent alternation of classification, causing spurious gesture detection.

Errors from color misclassification illustrate the persistent difficulty of discriminating fingers from arbitrary backgrounds. The method used to determine skin color by adaptive clustering is robust and fast, but does not disambiguate well against skin-toned backgrounds. We attempted to fix this problem by attaching red illuminants to the camera pointing at the hand. Unfortunately this causes mutual illumination from the hand onto white surfaces (e.g. table tops) resulting in more non-finger areas matching skin color.

The overall frame rate of the system is quite high (9 fps) but the high delay time (one second) makes interactive usage difficult. This delay is incurred through a combination of processing lag and the window of time allocated to wait between the peaks of movement for each finger involved in a multi-finger gesture. We may improve the accuracy of detecting multi-finger gestures, but only at the cost of increased delay. A partial solution to the delay problem, as well as an increase in accuracy, could be achieved by eliminating multi-finger gestures from the alphabet.

7 Conclusions

This paper has described a new kind of input device for wearable computing. Its design gears it towards unobtrusive use in a variety of environments.

The system is completely adaptive and requires no prior training in terms of skin color or expected background conditions. This adaptability is the key strength of this design. It allows the system its robustness and ability to operate in real-time. However, there may be reliability advantages to be gained by equipping the system with some pre-defined notions of what colors to define as skin colors and what colors to define as background.

The next step in development is to develop a more sophisticated input language. This will be designed to be partially error correcting.

The prototype system described here proves the potential for using a WristCam as an input device. By freeing the user from the bonds of a tactile interface and affording them the benefit of discreet operation, this system allows invisible wearability to extend to input and control.

8 References

[1] Pavlovic, V.I., Sharma, R., Huang, T.S., “Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, July 1997, Vol. 19, No. 7, IEEE Computer Society, Los Alamitos, pp. 677-696.

[2] Kjeldsen, R., Kender, J., “Finding skin in color images” *Proceedings of the International Conference on Automatic Face and Gesture Recognition*, Killington, VT, October 14-16, 1996, pp 312-317.

[3] Saxe, D., Foulds, R., “Towards Robust Skin Identification in Video Images”, *Proceedings of the International Conference on Automatic Face and Gesture Recognition*, Killington, VT, October 14-16, 1996, pp 379-384.