References

[1] S. Fels and G. Hinton. “Glove-talk: A neural network interface between data-glove and a speech synthesizer.” *IEEE Transactions on Neural Networks*, 1995\

[2] W. Freeman and M. Roth. “Orientation Histograms for Hand Gesture Recognition.” *International Workshop On Automatic Face And Gesture Recognition*, June 1995. Zurich, Switzerland.

[3] E. Fosler and P. Neuhaus. “Using Orientation Histograms for Hand Gesture Recognition.” *CS280 Class Project*, May 1995. University of California, Berkeley.

[4] T. Starner and A. Pentland. “Visual Recognition of American Sign Language Using Hidden Markov Models.” *Media Lab Technical Report*, Media Lab, Massachusetts Institute of Technology, 1995. MIT Media Lab TR#306.

orientation histogram by measuring the local gradient direction at each pixel. Their method is implemented using 2 filters (measuring derivatives at x and y direction), and the orientation histogram can vary continuously, instead of having values at discrete angles. The shortcoming of this system is that at an intersection, the gradient direction would lie in an intermediate direction. In our system, we measure orientation at distinct angles through convolution, so our system works correctly at an intersection.

7 Conclusion

Now, we will conclude with statistical results we obtained from our system. This will introduce our views on what the limitations were and what we can do in the future to improve our recognition application.

7.1 Result

Table 3: Accuracy Measurement

Hand type vs. Command	Hand A 1st Set of Data	Hand A 2nd Set of Data	Hand A w/ Less Light	Hand B w/ data Hand A
Forward	90.0%	70.0%	100.0%	74.0%
Right	96.0%	100.0%	72.0%	88.0%
Left	60.0%	92.0%	50.0%	82.0%
Open	86.0%	80.0%	72.0%	82.0%
Close	98.0%	100.0%	100.0%	96.0%
Average	**84.0%**	**88.4%**	**78.8%**	**84.4%**

As shown on table 3, our system works remarkably well given the speed constraint. Our system worked at 86.2 percent accuracy when the hand in the template image is the same as the hand in the sample image. Thus, we tried some different setups to test its robustness.

Surprisingly, our system gives approximately 78.8 percent accuracy even when the sample images are captured under less lighting compared to the template images. (The recognition algorithm performed very poorly when the template images were taken with less lighting. This shows that getting good template images is very important for the gesture recognition algorithm.)

Then images of different hand is used. Again, our system gives consistent accuracy of 84.4 percent. This shows that using different hands makes relatively little difference to the final orientation histogram, as we desired.

7.2 Limitations

The biggest factor that effect our project turns out to be hardest to control. Small camera we used is of low quality. Therefore, captured images are also in poor quality. These poor pictures are then mis-recognized and produce erroneous response. We believe higher quality camera will drastically raise the accuracy of our recognition system.

Also, we believe the algorithm we used needs to be replaced with another. The orientation histogram algorithm we used, although accurate, is highly computation intensive. Thus, even with 6 by 6 strides that we take, our recognition system is quite slow. Although there are many algorithms that we haven't tested, we believe that combination of other faster algorithms may produce more desirable result.

Lastly, since most algorithms are, in fact, computation intensive, result will definitely improve if faster computers are used. In most algorithm, each sets of calculations are independent of each other. Thus, using either vector computers or parallel processor will improve its performance.

7.3 Future Work

One of the most exciting improvement we can easily add is learning capability. When image matches are made, we can check for accuracy of system. If system finds that accuracy of its match is not satisfactory (e.g. within certain threshold) it can then ask user to enter in right match. Then this new vector can be either stored in the memory in a neural networking fashion or integrate itself with its original template. After some iterations of these learning processes, our system will have higher accuracy in recognizing a gesture.

For each application, we believe that there are set of algorithms that will work better than others. Therefore different algorithm needs to be used for different application. In an interactive game, speed maybe more important than accuracy. Then an algorithm optimized for speed (such as finding the center of mass in each strips of picture) would be more suitable. Whereas in face recognition program, more accurate algorithms are preferable. Thus, in future, we will examine many recognition algorithms available. Then for each application, we will compare its need with what each algorithms can provide to give best results.

pixel in the composite image is marked as a background. This threshold eliminates the background noise. In the composite image shown above, different orientations are colored as shown below:

Table 1: Composite Image Description

Color	Orientation Description
Grey	Undetected Area - background
White	Local Orientation at 0 degrees
Red	Local Orientation at 45 degrees
Green	Local Orientation at 90 degrees
Blue	Local Orientation at 135 degrees

In order to achieve an acceptable speed, our system strides through the image and calculates the local orientation for every 6 x 6 squre.

4.2 Classification

The orientation histogram is derived from the composite image by counting the number of pixels oriented at 0, 45, 90 and 135 degrees. We do not normalize the orientation histogram, because in practice, we found that normalized histograms produced less accurate results compared to an unnormalized histogram.

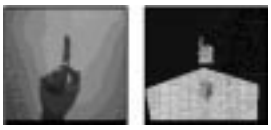
The classification is performed using the nearest-neighbor technique. The template histograms can be described as a point in 4-D space. Each dimension corresponds to 0, 45, 90 and 135 degrees. Given a sample point in 4-D, the nearest neighbor is found by finding the template point with the smallest Euclidean distance to the sample point.

5 Virtual Reality Application

The hand gesture recognition system offers an intuitive interface to the virtual environment. The virtual hand matches the gesture of the user's hand, and the virtual hand changes position according to the shape of the hand.

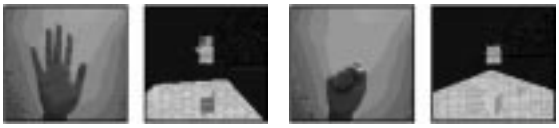
Table 2: Recognition and command

Hand Shape	Command Function
Pointing Up	Moves virtual hand forward.
Pointing Slant	Changes direction of virtual hand.
Closed Hand	Virtual hand grabs a selected object
Open Hand	Virtual hand releases selected object



(a) UP - Forward

(b) RIGHT - Turn Right (c) LEFT - Turn left



(d) OPEN - Release (e) CLOSE - Grab

Figure 3: Hand gesture and VR

As shown on figure 3, our virtual reality environment will move forward upon recognition of index finger pointing upward. And change directions when the finger slants either left or right. Then once virtual hand is above an object, it is highlighted. This indicates that an object can be grabbed and dragged by gesturing closed hand and move. Once an object is moved to a desired position, it can be released by gesturing an open hand.

Initially, a user supplies the system with a set of template images. Then in recognition phase, the system matches captured images against template images. As a result, our virtual reality application currently runs at the rate of 0.4 frames per second.

6 Other Systems

Fosler and Neuhaus's system uses the same algorithm for deriving the orientation histogram. Their system works off-line, so speed is not a factor in their design. Their system derived the orientations at 8 distinct directions, and calculates the local orientation at every pixel. (Our system calculates the local orientation at 4 distinct directions, once every 36 pixels.) Given their advantages, their system performed around 90 percent accuracy, which compares favorably with our system (at around 85 percent accuracy.) Also, their system matches the sample histogram against a class of template images, where a class may contain many images with the same gesture. To match against these images, they implemented a variety of matching techniques, including neural network. Our system uses one template image for each gesture, so we do not such elaborate matching scheme.

The main difference between Freeman and Roth's system and our system is that their system calculates the

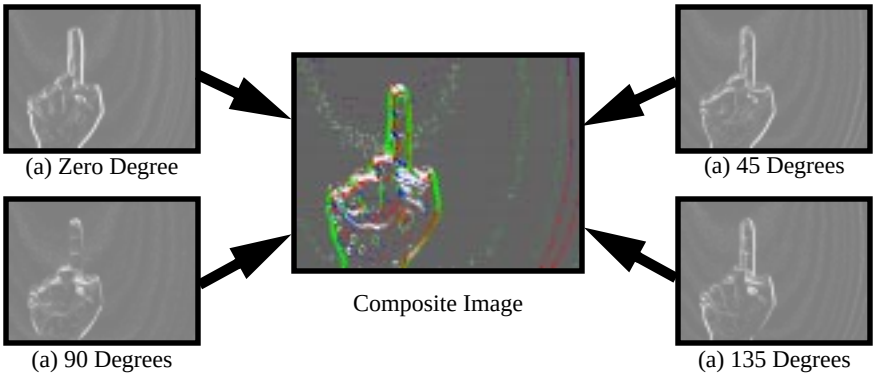


Figure 2: Making a composite Image

(2) The recognition algorithm should be translation invariant. When the system is trying to recognize a human hand, it is impossible for the user to hold his hand perfectly still. The position of the hand may change significantly, and the position of the fingers may even change. The recognition algorithm should be robust enough to handle these changes.

(3) We want an algorithm that is robust against changing lighting conditions. The lighting conditions may change between the time when the template images and the sample images are taken (due to changing background, different time of day, etc). Therefore, it is highly desirable that the recognition algorithm is robust against such changes.

(4) It is cumbersome for users to input the template hand gestures every time system is used. It would be desirable to store generic hand gesture templates (this may or may not be the user's hand) for the matches. Therefore algorithm need to be able to adapt to such difference. This is an attainable goal because all human hands share common characteristics (same number of fingers, same basic shape, etc.)

Given this list of desired characteristics, we decided to use "orientation histogram" as the feature vector for gesture classification, since it exhibits all of the desired characteristics.

Orientation histogram works as follows: Given an image, the system calculates the local orientation at every pixel. After this step, one can construct an orientation histogram which shows how many pixels exhibited a given orientation. For example, the X-axis can show the orientations (from 0 to 180 degrees) and the Y-axis can show how many pixels were orientated in the corresponding direction.

The orientation histogram is relatively unaffected by changing lighting conditions, since it detects how the edges are oriented. A changing lighting conditions may reveal more of these edges, but would not change the relative ratio among the number of edges with different orientations. Therefore the orientation histogram would maintain its basic shape under different lighting conditions, and the orientation histogram can be normalized to help the matching process.

It is also relatively translation invariant, since the ratio among different local orientations should not change when the hand moves to a different position. The orientation histogram may change its shape if the hand shifts to reveal a portion which was previously occluded. For our project we assume that such effects are negligible. We use a blank white background so that such occlusions are minimized.

It is robust even when different hands are used. This is because it measures the relative ratio among different local orientations. Since all human hands have same basic shape(e.g., the middle finger is longer than the index finger, etc.), the orientation histogram should be relatively robust even when different hands are used. Of course, the sample hand may have a significantly different shape from the template hand (e.g., different race, different age, etc.) Our (limited) experiments shows, however, that our recognition algorithm works well even when a different hand is used. The result of our exhaustive experiment shows less than 2 percent degradation in accuracy.

4 Image Processing

The orientation histogram can be calculated relatively efficiently by convolving the image with appropriate 2-D Gaussian derivative filters. These directional images are then combined to form a composite image.

4.1 Composite Image

As shown on figure 2, composite image is derived from the four directional images as follows: at each pixel, the directional image with the largest intensity is found, and the corresponding pixel in the composite image is marked with the appropriate orientation. If the largest intensity falls below the threshold, the corresponding

Virtual Reality Simulation UsingHand Gesture Recognition

Franklin Cho and Young Cho
University of California at Berkeley
Department of Computer Science
Berkeley, Ca 94704

Computer Vision - CS280
Professor Jitendra Malik
May 10, 1996

Abstract

Our purpose of this project is first, present a hand gesture recognition techniques introduced by Freeman and Roth[2] then make unique design decisions on the algorithm to integrate it into real-time applications such as virtual reality with minimal hardware setup.

We use second derivative gaussian filters for horizontal, 45 degrees slant, vertical, and 135 degrees slant orientations. All four filters are convoluted with the captured images to produce corresponding directional images. These directional images are weighed and compared to produce composite image which then can be abstracted as a vector. These vectors are then used to compare with the template of original image vectors.

However, there are limitations in implementing such algorithm without use of special camera and other hardwares. These calculations need to occur in real-time for satisfactory performance in virtuality reality applications. Thus fast and efficient method for calculation is necessary. To accelerate image processing rates, sampling technique and thresholds are used. We will discuss such adaptations and limitations in this method.

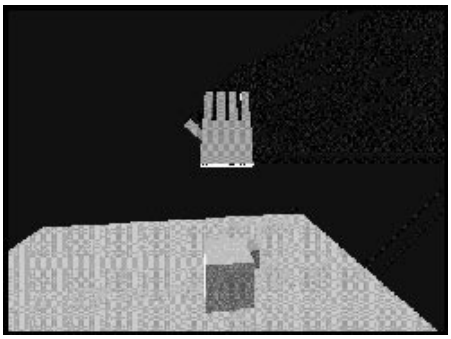
1 Introduction

Computer recognition camera may soon become the next generation I/O device that can provide more natural interface between human and computer. Using hand gestures or facial expression, one may control pointing icons, play interactive games, or maybe further down the line, drive a car.

We focus our recognition algorithm for what Freeman and Roth refers to as a static gesture recognition. This type of recognition is based on still images captured by camera. Our implementation involves extracting recognition data from image and comparing it with templates of pre-recorded images.

For robustness of our design, we make certain adjustments to algorithm based on mathematical results and experimentations. This results in satisfactory recognition for integration of the system into an application.

Figure 1: Virtual Reality Application



2 Motivation

We built our recognition system so that the user can interact with the virtual environment using hand gesture. Other than a digitizing camera, our system does not require any special hardware except or hand markings. Therefore, our system is much more natural way for a user to interface with computers unlike other systems that require special hardware devices (such as Fels and Hinton's system [1] and other systems that use data gloves) or systems which require body markings (such as Starner and Pentland's system[4]). Also, while Fosler and Neuhaus' system[3] which does computation off-line using MATLAB, our system does computations on the fly which then interfaces with real-time application.

Our system is intuitive to use, since the virtual hand mimics the gesture of the user's hand. Our system works relatively fast, achieving a frame rate of 0.4 frames per second.

3 Recognition

We wanted a recognition algorithm which exhibited following behavior:

(1) We wanted an algorithm which is relatively simple and fast, which can run in real-time on a workstation. Efficiency is critical in the recognition algorithm for real-time application.