The Application of Image Processing Techniques to Sign Language Recognition Using a Web Camera

Computer Systems Lab 2008

Byron Hood

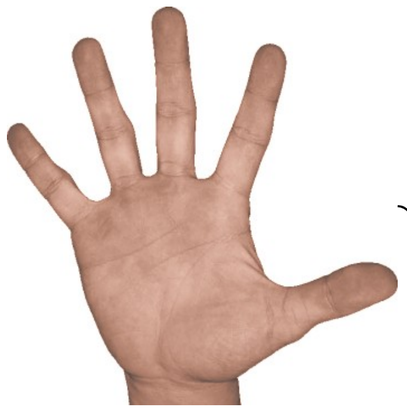
abstract

Sign language recognition is the first step in a long road towards natural language processing, or the ability for a computer to "understand" naturally spoken language. Such an invention would drastically lessen the amount of time require for computer input, maybe even by a factor of two. This project explores using image recognition techniques such as edge detection and line detection to identify sign language in real time, using input from an average web camera ("webcam"). When research is complete, it is expected that the program will be able to identify most, if not all alphanumeric characters with a high degree of accuracy.

background & introduction

In today's society, people with hearing and speaking disorders communicate using sign language. Through extensive practice and use (as people gain extensive practice speaking their native language), sign speakers are capable of "speaking" as fast as others speak orally, from 200-220 words per minute.

The average computer user types 33 words per minute when transcribing and a mere 19 when composing[1]. If the average sign speaker can communicate using finger spelling at as little as ¼ the pace of regular sign language, they sign 50 words per minute. If they could sign into a computer, this would be a significant speedup in computer input.

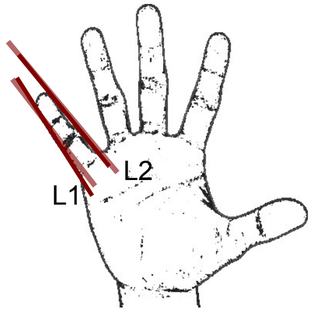


The image of a hand, to be captured from a webcam. Here we use the sign for "5."

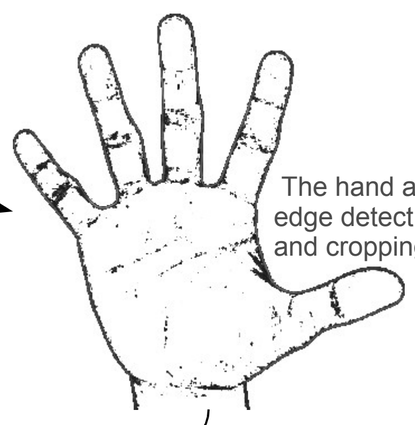
program procedures

Line-finding is the first step towards identifying complex shapes in an image.
You can try this yourself: look at your hand and try to find individual straight lines. Then, imagine what those lines would reveal about the position of your hand if nothing else about it was known.

What would *you* do to (mathematically) find your fingers in an image?



Edge detection



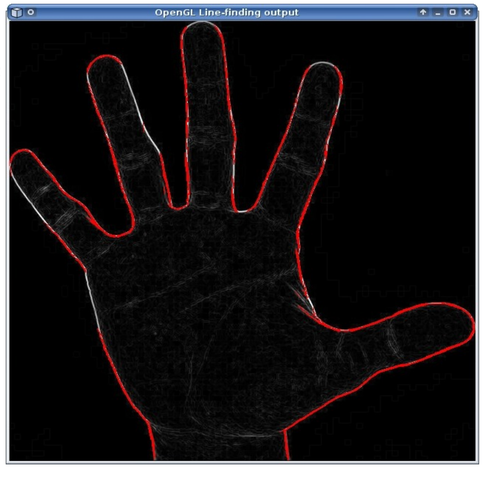
The hand after edge detection and cropping.

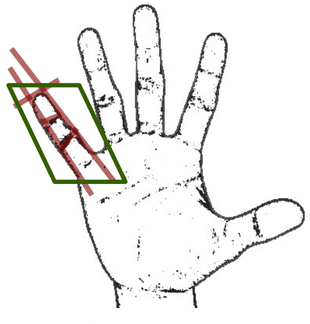
Line finding

Parameterized lines:
L1: (0, 50) → (30, 95)
L2: (10, 45) → (45, 89)
...

Endpoints of the lines found in the picture. A theoretical version is on the left, and the program output is seen on the right (red lines are detected lines)

Line interpretation:
Analyzes all of the lines in a given region and tries to match them to a finger.





The grouping is similar to a pinky

Final result:

```
> ./main
  ...
Detected char '5'
```

testing & timing

The program will be tested manually: automated testing would be highly impractical and would require complex image analysis (which is the object of my program). An example test would be running a program twenty times, and recording the time taken after each iteration, then manually viewing the results after all execution is complete. Below is sample timing, calculated by the program:

```
bhood@testing ~/syslab-tech/src $ ./main hand.png
Edge detection took 0.04 sec
Image cropping took 0.00 sec
Line detection took 0.17 sec (detected 1424 lines)
Line chaining took 0.25 sec (detected 130 chains)
Getting orientation took 0.08 sec (1 => ORIENTATION_FORWARD)
Getting pinky pos. took 0.00 sec (2 => FINGER_BENT)
Getting ring pos. took 0.00 sec (2 => FINGER_BENT)
Getting middle pos. took 0.01 sec (2 => FINGER_BENT)
Getting index pos. took 0.00 sec (4 => FINGER_TUCKED)
Overall process took 0.47 sec
[TOTALCOUNT] alloc: 10718901, free: 10364880; leak: 354021.
```

References

[1] Karat, C. M., et. al. "Patterns of entry and correction in large vocabulary continuous speech recognition systems." Chicago 1999 Conference Proceedings, 568-575.