Estimation of Hand Posture with Straight Line Detection for a Hand Pose Rally System

Ayumu MEIJI, Akira SUGANUMA *

National Institute of Technology, Ariake College,
150, Higashihagio, Ohmuta, Fukuoka 836-8585, Japan

*Corresponding Author: suga@ariake-nct.ac.jp

Abstract

There is an event called "stamp rally" that collects checkpoint stamps. The event needs some rally tools. If a participant loses his/her rally tools, it will be difficult for him/her to continue the rally. We are developing the hand pose rally system, which is one of the "gesture interface". Our system attempts to identify an individual by the posture of the participant's hand captured by the USB camera. By bending and stretching five fingers, 32 types of hand postures are achieved. This system estimates which of the 32 types of hand postures whom he/she has presented. We have been constructing a posture estimation method. However, the accuracy was not so good depending on the posture of the hand. In this paper, we focus only on the posture estimation part of the hand pose rally system, and consider a method to improve the estimation accuracy.

Keywords: Posture estimation, Hand pose rally system, Finger direction, Straight-line detection.

1. Introduction

Research Background

There is an event called "Stamp Rally" that aims to collect stamps at checkpoints. In recent years, stamp rally sometimes uses a FeliCa card⁽¹⁾ instead of the stamp card. Furthermore, today's stamp rally uses QR codes and GPS. These events require some tools such as stamp cards, FeliCa card and smartphone. Therefore, you may lose those tools. Loss of tools will make it difficult to continue the rally. On the other hand, a new user interface that has appeared in recent years is the "gesture interface". As the term suggests, it is a type of human interface that can be entered by gestures. In a survey on gesture interfaces⁽²⁾, 1,000 men and women

aged 15-49 years old were asked about their impressions of gesture recognition, and 74.7% of all respondents answered that it was "fun". Therefore, due to the high public interest in gesture recognition, the development of a hand-pause rally system using gestures was started in 2012.

Research objectives

In our laboratory, we have developed a prototype system of the hand pose rally system. Our system recognizes hand posture with the vectors connecting the interdigital space and the fingertips in the image of their right hand^(3, 4). We classified the postures of the hand into 32 patterns with the fingers bent and stretched. Our system estimates which pattern the posture presented by the participants is. In the operation of the system, however, the problem of participants not being able to pass the checkpoint if the posture estimation is not done correctly occurs.

Previous studies on hand shape estimation include those that estimate the shape three-dimensionally with multiple cameras⁽⁵⁾ and those that use machine learning⁽⁶⁾. These are highly accurate, but they request a large-scale system. As a hand pose rally system, we demanded to build a system with relatively simple equipment. Therefore, we have adopted a method of estimating by combining relatively easy image processing with only one camera.

In this study, we focus only on the posture estimation part of the system and consider how to improve the estimation accuracy. Until the year before last, methods of generating a vector from the wrist to the interdigital space and two vectors from the interdigital space to the fingertip and comparing them with representative postures taken beforehand, as well as methods of posture estimation using a CNN, were being investigated. In this paper, we describe a method to generate a vector from the wrist to the root of the finger and a vector from the base of the finger to the fingertip

using the Hough transform.

2. Overview of Our Method

2.1 Design Policy of Our Algorithm

In a study to improve the accuracy of conventional posture estimation methods, the ambiguity in vector generation was identified as a problem. One of the causes of this problem was that the position of the fingertips had changed because the range of movement of the fingers varied from person to person. That is, there needs to be a way to lessen the impact even if the coordinates of the fingertip points change significantly.

The lines next to the stretched fingers appear to be straight lines. Based on that line, get the coordinates of the base and fingertip of the stretched finger. Then, the coordinates of the wrist are obtained. Finally, the positional relationship between the "coordinates of the wrist" and the "coordinates of the base of the fingers or fingertips" allows us to determine which finger is stretching. We hypothesized that the "line from the fingertip to the base of the finger" and the "line next to the finger" were approximately parallel. Therefore, we expect that we are able to estimate posture more accurately than before.

2.2 Generating a Vector from the Posture

The method of generating a new attitude vectors is shown below:

1. Extracting the hand contour.
2. Creating a convex hull.
3. Detecting straight lines from the hand contours and drawing on the image.
4. Repeat from step 1 to step 3 for 30 times.
5. Thinning the lines in the image of step 4, and detecting straight lines by the Hough transform again.
6. Determining on which side of the retrieved line the fingers are.
7. Selecting only the fingertip points of a stretched finger from the points acquired in the convex hull.
8. Detecting the base of the fingers.
9. Generating vectors from the base of finger to the fingertip.

Fig. 1. Shooting Environment

After these processes, our system estimates the posture of the hand by comparing the vectors of the typical posture and the vector of the presented postures as before. The direction of the vectors from the wrist to the base of the fingers is aligned, and the direction of the vectors from the base of the fingers to the fingertips is compared to find the one with the closest value. Search for a typical posture that is closest to the presented posture using the method described above, and determine which posture is the presented posture.

2.3 Shooting Environment

Our system captures a right hand of the participants with a USB camera, which is a "C615" manufactured by Logitech. As shown in Fig. 1, the camera was fixed to a light stand. It is oriented at right angles to the desk. The distance from the camera to the desk is about 36 cm. Furthermore, a black cloth was placed underneath it for capturing. By putting down the black cloth, the background of the hand-posture image was unified to black. The lighting is a fluorescent lamp in the laboratory.

3. Posture Estimation Methods

3.1 Convex Hull

A convex hull is the smallest convex set that contains all of a given set. Our system finds the set of vertices of the smallest convex polygon that surrounds the hand.

3.1.1 Getting the hand contour

Our system detects the contour of the hand as part of the method getting the convex hull. Here, the program for acquiring the contour of the hand will be roughly described. Our system uses the library function in OpenCV to perform

Fig. 2. Binary Image

Fig. 3. Contour of the hand

the process below.

1. Converting the image obtained from the camera to a binary image (Fig. 2).
2. Getting the set of coordinates that make up the contour. (Fig. 3).

3.1.2 Getting the convex hull

The rough flow of the program of convex hull acquisition is as follows.

1. Input the set of coordinates of the contour obtained in the previous section into the function `convexHull()` in OpenCV and get the set of coordinates that make up the tentative convex hull.
2. Enter the set of coordinates that make up the tentative convex hull into the function `contourArea()` in OpenCV and get the area of the convex hull (Fig. 4).
3. Get the convex hull area if it is more than a certain amount (more than 1000 in this program), otherwise, don't get it.

The coordinates of the convex hull at this point may

Fig. 4. Convex Hull

include not only fingertips, but also the coordinates of the wrist at the edge of the screen and the coordinates of other protruding parts of the hand (e.g. the thumb and the base of the little finger). Also, multiple points are obtained at the end of the same finger. However, later in the program, only one coordinate of the extended fingertip is used for each finger. Therefore, unnecessary points must be removed.

3.2 Eliminating Unnecessary Points

Let "old_array" be the array of coordinates that make up the convex hull obtained in the second paragraph of the previous section, and "new_array" be the array of coordinates from which unnecessary points were removed in this operation. The "new_array" is assumed to have no elements in it yet (initialized).

3.2.1 Eliminating multiple points in a range

The general flow of the program is as follows. We use the "Point" type, a variable type that can be used in OpenCV. A "Point type" is a structure for pointing to a point on a two-dimensional plane. Its elements are "x (x coordinate)" and "y (y coordinate)".

1. Add the old array[0] (the first element of the old array) to the new array.
2. Prepare the variable "i" and set "i = 1".
3. Find the distance between the old array[i] and each element of the new array.
4. Check if there is an element with a value greater than 40 in the obtained distance array.
5. If not present, add old-array[i] into the new array.
6. If it exists, do not add the old array[i] to the new array.
7. Performing "i++" operations.

Fig. 5. Elimination of multiple points in a range

Fig. 6. Elimination of the wrist points

8. Perform operations from 3 to 7 until all elements of the old array have been examined (Fig. 5).

Before executing this process, there were multiple points at the fingertips. After execution, however, there is only one point on each finger.

3.2.2 Eliminating the wrist points

The general mechanics of the program are as follows. One of the following four conditions is satisfied.

- The y-coordinate of the point of interest is within 10 distance from the top edge of the image.
- The y-coordinate of the point of interest is within 10 distance from the bottom edge of the image.
- The x-coordinate of the point of interest is within 10 distance from the left side of the image.
- The x-coordinate of the point of interest is within 10 distance from the right side of the image.

The points of interest are removed from the elements of the new_array. This process removes points that appear at the wrist position (Fig. 6).

Fig. 7. Points appearing on the protruding part

3.2.3 Eliminating points appearing on the protruding part of the hand

Convex hull points may appear on the outer edge of the hand, such as when the little finger is bent as shown in Fig. 7. These points cannot be removed by the above process. At least one remains. In Fig. 7, there is one point each on the hypothenar of the hand and the base of the little finger. The remaining points may reduce the final accuracy of the estimation of hand posture. We are considering a special method to remove this point.

3.3 Detecting Straight Lines of the Stretched Fingers

Our new method detects straight lines from the edge image by using Hough transform⁽⁷⁾. Our conventional researches took a point between two stretched fingers and connected that point to a point on the fingertip to make a finger vector. The angle may be consequently different when compared to the actual finger orientation. On the other hand, using the straight lines detection, we can detect a straight line along the finger that has the same angle as the actual finger direction. Therefore, the accuracy of posture estimation can be expected to be higher than in previous studies. We have developed the method detecting straight lines by using the function HoughLinesP() in OpenCV.

3.4 Thinning Process

Thinning is the process of converting a binary image into a line image with a width of 1 pixel. If the straight line of the binary image is thick, an erroneous straight line may be detected when the straight line is detected by the Hough transform. The thinning is done by an ordinary method⁽⁸⁾.

3.4.1 Execution results

Fig. 8 shows the results of a straight-line detection with images of successive 30-frame by Hough transform. You can see the shape of the open hand. The straight line is thick

Fig. 8. The binary image in which 30 straight line detection images are superimposed

Fig. 9. The image after the thinning

because the detection results of 30 times are superimposed.

Fig. 9 shows the result of thinning Fig. 8. Although there are some branches, the image was output with fine lines while retaining sufficient features of the input image. After that, the Hough transform is performed by inputting the thinned image. Then, the point removal program described in the section on convex hull is performed.

3.5 Rotating the Hand Image

These programs work when the hand is presented from below the image. However, it is not always the case that the hand is presented from the bottom of the screen. Therefore, we wrote a program that rotates the image in which the hand is presented and converts it into an image in which the hand is presented from the bottom of the screen.

If the image is rotated as it is, there is a possibility that part of the hand will be missing from the image. Therefore, we added the process of adding black margins to the top, bottom, left and right of the image first.

3.5.1 Algorithms

1. Adding margins to the top, bottom, left and right of the image. The height and width of the image with

the margins added will be the maximum distance that the hand can be seen in the original image (the length of the diagonal of the original image).

2. Extracting the straight line located on the wrist from the straight line detected by the Hough transform. The two end points of the wrist line are denoted as P and Q, respectively, and the x and y coordinate values are denoted as P.x, Q.x, P.y, Q.y, respectively. Let point P be a point that appears near the edge of the image. If "P.x is near the edge of the image" and "Q.x is not near the edge of the image" and "either P.y or Q.y is not near the edge of the image" are satisfied, the line segment PQ is considered to be a line of the wrist. Also, if "P.y is near the edge of the image" and "Q.y is not near the edge of the image" and "Neither P.x nor Q.x is near the edge of the image" are satisfied, the line segment PQ is considered to be a line of the wrist.
3. Find the angle to rotate the image. The intersection of the straight lines drawn vertically from point B to the bottom edge of the screen and the bottom edge of the screen is point C. The angle of angle ABC at this point is the angle to be rotated.
4. Rotate the image. First, use the `getMatrix2D` (arguments: coordinates of the center of rotation, angle of rotation, degree of scaling) function to obtain the transformation coordinates. Then, perform affine transformation with `warpAffine` (arguments: input image name, output image name, transformation matrix, transformed image size) to rotate the image. (The rotated image is called `rotate_image`)
5. Decide the cropping area. To crop, we need the coordinates of the upper left edge, width, and height. Perform a raster scan from the bottom row of the image to the top edge. At the same time, count how many white pixels are in that row. When there are more than 150 white pixels in a row (about 120 is the thickness of the arm, and about 180 is the widest part of the palm), calculate the average x-coordinate (x_{wrist}) of the white pixels in that row. Also, let y be y_{wrist} at this time. This gives us the coordinates of the wrist (x_{wrist} , y_{wrist}). The coordinates of the upper left corner are (x_{wrist} - "half of the width of the cropped image", y_{wrist} - "half of the height of the cropped image" + 150).
6. The above process is achieved by using the

function Rect() in Opencv.

3.5.2 Execution results

In Fig. 10, image A is "an image of a hand captured by a camera," image B is "a binary image of image A," and image C is "an image with black margins added to image B." In Fig. 11, image D is "an image in which a straight line is detected from image A," image E is "a rotated image of image C", and image F is "an image of E cropped to the size of image A."

You can see from images B and C that margins have been added to the top, bottom, left and right. From images C and E, you can see that the wrist (arm) of image E has been able to be rotated so that it faces down. From Image E and Image F, you can see that the wrist of Image F is centered at the bottom edge of the screen.

4. Conclusions

4.1 Summary

In this research, we have developed a prototype system that estimates the posture of fingers using Hough transform. It may not be possible to detect an existing straight line by detecting a straight line with a single Hough transform. Our

Fig. 10. Rotation of the image (1)

Fig. 11. Rotation of the image (2)

method detects straight lines with images of successive 30-frame. After that process, the detected lines are thinned. It is possible to detect straight lines along the contours of both sides of the finger, and it is possible to detect the direction of the finger. We hope that this method will improve the accuracy of hand posture estimation.

4.2 Future Issues and Prospects

Future works include devising a method to remove the points that appear on the protruding part of the hand. We believe that this can be implemented by determining whether there is a point in a stretched finger using a line obtained by the Hough transform. If we solve the problem, we can get the point of the extended fingertip only. In addition, the pair of lines next to the stretched finger determined by the Hough transform is paralleled toward the center of the two lines to create a vector from the base of the finger to the fingertip. Then, the vector from the wrist to the base of the fingers is also created. We hope to complete the system for hand posture estimation from the direction of the final vectors created.

Also, in this system, changes in lighting intensity have a large effect on the results. The reason seems to be that our system uses the RGB color space. Therefore, changing the color space used in the system from RGB to HSV is expected to solve the problem.

References

- (1) T. Takahashi, S. Hori, R. Matsumoto, H. Kobayashi : "Development and Evaluation of FeliCa Stamp Rally Ver. 2 System," The Institute of Electronics, Information and Communication Engineer, IEICE technical report, pp23-28, 2012 (in Japanese).
- (2) Tokyo Polytechnic University : "Survey on Natural User Interface," <https://www.t-kougei.ac.jp/static/file/nui.pdf>, pp. 12-14, 2012 (in Japanese).
- (3) A. Suganuma, Y. Kunitomo : "A method estimating the postures of a hand with a single camera for Hand Posing Rally System," Research reports of the Ariake Technical College, Vol.50, pp.15-22, 2014 (in Japanese).
- (4) A. Suganuma, and Koki Nishi : "Development of a Hand Pose Rally System Based on Image Processing," IEIE Transactions on Smart Processing and Computing, vol.4, no.5, pp.340-348, 2015.
(DOI: <http://dx.doi.org/10.5573/IEIESPC.2015.4.5.340>).

- (5) Weiyang CHEN, Ryuji FUJIKI, Daisaku ARITA, and Rin-ichiro TANIGUCHI: "Real-Time 3D Hand Shape Estimation using Multiple Cameras," MIRU2006, pp.328-333, 2006 (in Japanese).
- (6) Ryousuke MUTOU, Kazutaka SIMADA, and Tutomu ENDOU: "Hand Shape Recognition based on SVM and online learning with HOG," TECHNICAL REPORT OF IEICE., HIP, Vol.109, No.471, pp.459-464, 2010 (in Japanese).
- (7) Math Works : "Hough Transform,"
<https://jp.mathworks.com/discovery/image-transform.html> (in Japanese).
- (8) "Thinning"
<https://imaging-solution.net/imaging/thinning/>
(in Japanese).