Gesture Recognition based on 1-Dimensional Encoding of Motion Changes

Kengo KOARA Atsushi NISHIKAWA Kentaro SHINTO
Kaoru ISHII Yoichi YASUI Fumio MIYAZAKI
Department of Systems and Human Science
Graduate School of Engineering Science, Osaka University
1-3 Machikaneyama, Toyonaka, Osaka 560-8531, Japan
atsushi@me.es.osaka-u.ac.jp

Abstract

Gesture is a very intuitive way for human communication, and it is also very useful as interface for human-robot interaction. In this report, we propose a simple gesture recognition method which can be easily extended to handle multiple inputs. In the method we proposed here, at first, we calculate the position and velocity of the center of gravity of moving region in video image by using optical-flow field. Then, we consider that the maximum/minimum points of such motion information represent motion pattern well, we encode them into a 1-dimensional code sequence. Finally, we distinguish input pattern by comparing with template patterns which have been prepared in advance using a method based on string comparison algorithms. We apply the proposed method to recognition of hand gestures taken by single or 2 video cameras to show the extensibility of the method. We also show an example of gesture based mobile robot navigation.

1 Introduction

Gesture is a very intuitive way for human communication, and it is also useful as interface for human-robot interaction. Especially, research in vision-based human motion recognition has greatly increased in recent years (see [1],[6] for recent surveys), because vision-based methods can find human-motion contactlessly, that is, they do not restrict the user's motion.

Generally, we may say that visual motion-based gesture recognition contains the following 3 steps:

1. finding motion information of the target (e.g., hand position) from video sequence,
2. extracting and encoding motion pattern,
3. recognizing motion pattern.

For the first point, some proposed methods use correlation based optical-flow[2][3][5], because the methods using motion field are more robust to noise and/or illumination changes than color based or subtraction based methods. Correlation-based optical-flow can be calculated easily and quickly, which is an important point for real-time gesture recognition. However, the accuracy is not so good and is unreliable, especially in case of the target object moving quickly in the scene. Therefore, in this study, we do not use the direction information of optical-flow, but we use the center of gravity of the motion field as motion information of the target object. Then, we propose a method of 1-dimensional encoding of motion pattern, which is based on the consideration that motion changes represent characteristics of the movement pattern very well. We also propose a pattern recognition method based on string comparison algorithms. These methods can be extended to handle multiple inputs easily. In the following sections, we discuss details of these points, and finally, we show gesture-based mobile robot navigation as an application.

2 Finding motion of hand gesture

In this study, we consider single-hand gesture recognition. To detect the motion of hand gesture, we use a correlation-based optical-flow technique[1]. We calculate the optical-flow using a special hardware with correlation processor chip (Color Tracking Vision TRV-CPW5 by Fujitsu Co.,Ltd.) and a PC (CPU: Intel PentiumIII 700MHz, OS: Linux 2.0.36[4]) in real-time

[1]See [5] for details about the selection of local correlation parameters.

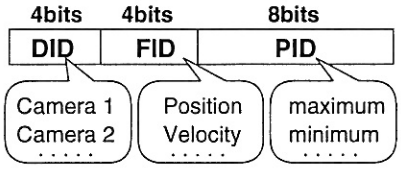


Figure 1: Assignment of IDs

(at the rate of 15Hz). However, it is very difficult for the correlation-based method to calculate the correct flow of the object moving quickly and/or around the boundary of the moving object. Hence, we do not use flow vector itself, but we just use it for finding moving region in the scene. We use the position and velocity of the center of gravity of such moving region as the motion information of the target object.

3 Encoding gesture pattern

Our approach is based on a consideration that the timing of motion changes is an important factor for describing motion pattern. Now we refer to the remarkable motion change as "event". We consider that such events occur at the time when the position and/or velocity of motion field in the image coordinate is maximal or minimal. Basically we can find the maximum/minimum points of the feature values such as position and velocity by selecting the point where the first differential is zero. In this paper, in order to avoid inappropriate results caused by noises, we use not only the values at the adjacent frames against the current time T_s, but also the values obtained a certain frames before/after T_s to calculate the first differential.

The event is encoded into 2 bytes code by the following rule. The code is composed of "Device ID (DID)" (4 bits), "Feature ID (FID)" (4 bits), and "Pattern ID (PID)" (8 bits). Fig. 1 shows how to assign each ID to the bit-field (total 2 bytes). DID indicates the type of input device (e.g., camera), FID means the type of the input feature (e.g., position, velocity), PID shows the type of event (e.g., max., min.). The code indicating each event is defined as the following equation: C_{event} = DID|FID|PID.

We align each event in order of the time of occurrence. When more than two events occur simultaneously, we align them in reverse order of the magnitude of the event ID (C_{event}). By doing this, we can generate a 1-dimensional code sequence. We refer to this as the "1-dimensional event code sequence (1DECS)".

4 Motion pattern recognition

For recognizing motion patterns, the system should have a kind of knowledge of the target motion patterns. In this study, we prepare the 1DECS of each motion itself as the motion pattern template, in advance. Generally, we have to consider temporal changes of input to recognize time sequential pattern. Most of the previous methods use the Hidden Markov Models (HMM) for gesture recognition to cope with this problem (e.g., [3]).

The 1DECS does not put emphasis on absolute occurring time of the events, but just considers the occurring order of them, that is, the encoding does not depend on absolute time, so that it is easy to cope with temporal changes. However, due to noise or motion difference, there may be variations in the same motion category. To cope with this, we prepare a number of 1DECSs for the template of each category.

Considering that the 1DECS is similar to simple string data, we apply the string comparison algorithms for computing the SED (shortest edit distance) and LCS (longest common substring). The SED problem is to find the shortest path of the edit operations from one string to the other. The edit means the operation of inserting/deleting letter (code) in/from the string (code sequence). The LCS problem is almost equivalent to that of SED. We use a SED algorithm for finding LCS. We do not discuss the SED and LCS algorithms for lack of space (See [7], for example).

4.1 Pattern similarity

Now, let us consider how to compare two 1DECSs, S and T. S is the 1DECS of input pattern, and T is one of the 1DECSs in the template. We can say that if the two sequences are similar, the LCS is almost the same as the input sequence. But when the length of two sequences is quite different and the longer one contains the shorter one in it, the LCS is almost the same as the shorter one, though these two codes are different. Considering this point, we define the criterion for evaluating the similarity of two sequences as follows:

$$P_{sim}(S,T) = \frac{\mathrm{len}(\mathrm{LCS}(S,T))}{\max(\mathrm{len}(S),\mathrm{len}(T))} \quad (1)$$

where $\mathrm{LCS}(S,T)$ is the LCS of S and T, $\mathrm{len}(X)$ is the length (the number of codes) in the code sequence X.

4.2 Finding best-match pattern

We show the procedure for finding the best match pattern of input 1DECS S below. Beforehand, we de-

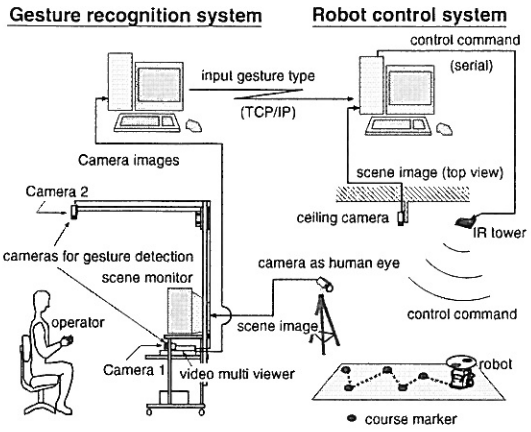


Figure 2: Experimental system

fine $\mathcal{M} = \{\text{All motion patterns}\}$ and $\mathcal{N} = \{1, 2, \ldots, Nt\}$ where Nt is the number of templates for each gesture type.

1. For $m \in \mathcal{M}$,
   calculate $P_{sim}(S, T(m)) = \max_{j \in \mathcal{N}}(P_{sim}(S, T_j(m)))$.

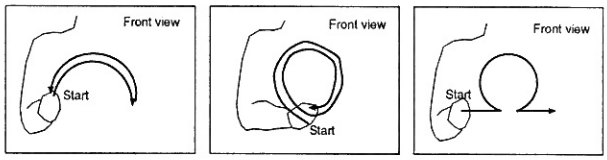
2. Find the pattern m of $\max_{m \in \mathcal{M}}(P_{sim}(S, T(m)))$ as the best fit motion pattern.

With this method, input pattern is always judged as one of the pattern in the template. However, in practical use, the user might input unknown gestures, or some input patterns might be unintended motion. We discuss how to judge these inappropriate inputs in Section 5.1.2.

5 Experiment

The system we use for the experiment consists of a PC with the color tracking vision (see Section 2), a 4-video signal multi viewer and 2 CCD cameras (see Fig. 2: gesture recognition system). One camera (camera 1) is placed in front of the hand of the operator (the person who inputs gesture), and the other one (camera 2) is placed above the hand.

To verify the proposed method, we made two experiments. In the first experiment, we use one camera placed in front of the user (see Fig. 2: camera 1), and verify the performance of the methods of motion pattern encoding and recognition. In the second experiment, to see the extensibility of the proposed method, we use 2 cameras to take the motion of the hand gesture (Fig. 2: camera 1 and 2).



(a) arch type (b) circle type (c) ohm type
Each gesture has two types according to the direction of the hand motion. Total 3 types, 6 categories.

Figure 3: Gestures for single camera input

5.1 Gesture recognition with 1 camera

In this experiment, we use only one camera placed in front of the hand of the operator (camera 1). The gestures we use in this experiment are "arch type", "circle type" and "ohm type" as shown in Fig. 3. Each type of the gesture has two types according to the direction of the hand motion so that the total number of the gesture categories is 6.

We recorded all of the input gestures on DV (digital video) tape so that we could reuse the input motion when the parameters of the recognition system were changed. We inputted the gestures by replaying the tape. The gestures were then spotted by evaluating the velocity and area of the moving region (we do not discuss the gesture spotting problem due to lack of space). The number of subjects was 4, and each gesture category was recorded 30 times respectively. The first 10 gestures in each gesture type were used for generating the template data, and the rest were used as the input of the recognition experiment.

Now we focus on the two experimental results: ① recognition rate and the relevancy of the gesturing person to the person who generates template (template generator), and ② the value of pattern similarity and its relevancy to the reliability of input gestures.

5.1.1 Recognition rate

We prepared the templates for each subject (the template holding number $Nt = 5$). We made experiments for each subject's templates. We calculated the recognition rate of

- gestures inputted only by the template generator,
- gestures inputted by all subjects

for the templates of each subject.

The result is shown in Fig. 4. As shown in this result, in case of gestures inputted by the template generator, the recognition rate was about 95%, but

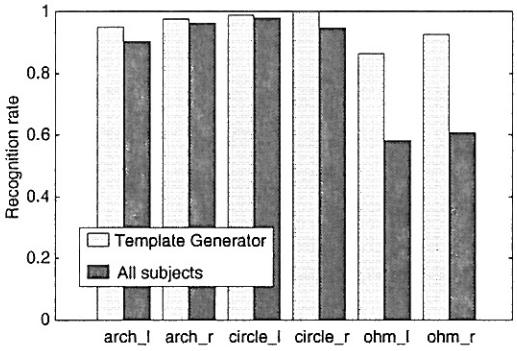


Figure 4: Recognition rate

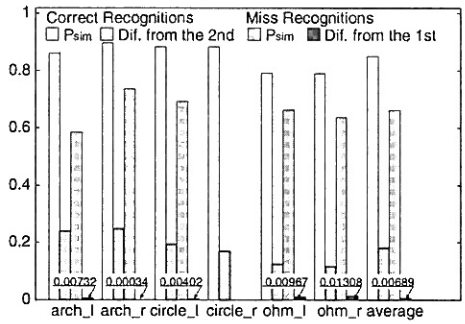


Figure 5: Evaluation of pattern similarity

in case of gestures inputted by all subjects, the rate was relatively low, especially ohm type was about 60%. The reason for this low recognition rate is that the pattern of ohm type is relatively complicated so that the motion patterns tend to be different between the subjects. That is, our motion pattern description method expresses such motion difference between each person.

5.1.2 Value of pattern similarity, and reliability of input

The proposed recognition method classifies input pattern into the known categories registered as the template. However, in practical use, it is desirable that the system can also recognize "unknown gesture" and "unintended input (or noise)" automatically. To accomplish this, we calculate the following 4 values:

1. when the input is correctly recognized
   - average value of P_{sim} for recognized pattern
   - difference between P_{sim} for correct pattern and P_{sim} for the 2nd candidate
2. when the input is wrongly recognized
   - average value of P_{sim} for recognized pattern
   - difference between P_{sim} for correct pattern (should have been recognized) and P_{sim} for wrongly selected pattern

P_{sim} indicates the value of pattern similarity defined by Eq. (1). In this experiment, we used the recognition results of the gestures inputted by the template generator.

The results are shown in Fig. 5. From the results, we can see that the average of P_{sim} for correctly recognized pattern was higher than wrongly recognized pattern (correctly recognized: around 0.8, wrongly recognized: around 0.6). And the more interesting point is that the difference between P_{sim} for correct pattern and P_{sim} for wrongly selected pattern was quite small (less than 0.01) when the input was wrongly recognized, on the other hand, when the input was correctly recognized, the difference between P_{sim} for correct pattern and the evaluation of the 2nd candidate was around 0.2. Hence, we consider we can make the method for evaluating the reliability of the recognized results by using the difference between P_{sim} for the 1st and the 2nd candidates.

Let m_1, m_2 be the patterns evaluated first and second for an input S, respectively. The difference between the evaluation of the 1st and the 2nd candidates is as follows:

$$\Delta P_{sim12} = |P_{sim}(S, T(m_1)) - P_{sim}(S, T(m_2))| \quad (2)$$

Then, we judge the reliability of the input by a threshold $Th_{ambiguous}$:

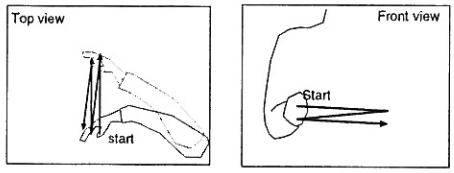
$$\begin{cases} \Delta P_{sim12} \geq Th_{ambiguous} : & \text{reliable} \\ \Delta P_{sim12} < Th_{ambiguous} : & \text{ambiguous} \end{cases} \quad (3)$$

The gesture judged "ambiguous" should be rejected as unreliable. By applying this, we can remove unknown or undesirable input as "ambiguous" at the gesture recognition stage, and miss-recognition rate can be decreased, but the number of rejected inputs tends to increase. The parameter $Th_{ambiguous}$ should be set depending on applications.

5.2 Gesture recognition with 2 cameras

We made another experiment of gesture recognition with 2 cameras for discussing the extensibility of the proposed method for inputs.

In this experiment, we added "mae type"and "yoko type"(yoko has two types according to the direction of the hand motion) to the gestures we used in the previous experiment with single camera (5 types, 9 categories, see Fig. 6).



mae type yoko type

yoko type has two types according to the direction of the hand motion. Total: 5 types, 9 categories

Figure 6: Added gestures for experiment with 2 cameras

As same as the previous experiment, we made the templates for each subject in advance ($Nt = 5$).

5.2.1 Experimental results

Recognition rates of gestures by template generators and by all subjects are shown in Fig. 7.

We can see that our method also worked when the number of the camera was extended 1 to 2. On the whole, the recognition rates were over 90%, however, in case of the gesturing person and the template generator were different, the recognition rates of ohm type, mae type, and yoko type were only about 60%. It seems that the number of simultaneously occurring events increased as the input devices increased, so that the patterns were much complicated. However, the tendency of P_{sim} for the 1st and 2nd candidates was similar to that of the single camera experiment.

6 Application: Mobile robot navigation with gesture

We applied the proposed method to mobile robot navigation as an example of application.

6.1 Experimental setup

We made a mobile robot using the LEGO Mindstorms[2](see Fig. 8). The robot has two wheels and it can "go forward", "go backward", "turn left", and "turn right" as the basic motion, and it is controlled by infra-red signal through the "IR tower" (IR communication port, belongings of the Mindstorms).

In this experiment, the operator observes the mobile robot at a remote place through the TV monitor screen indirectly, and navigates it with gestures(Fig. 2).

[2]Package of the LEGO blocks with programmable unit which can communicate with PC.

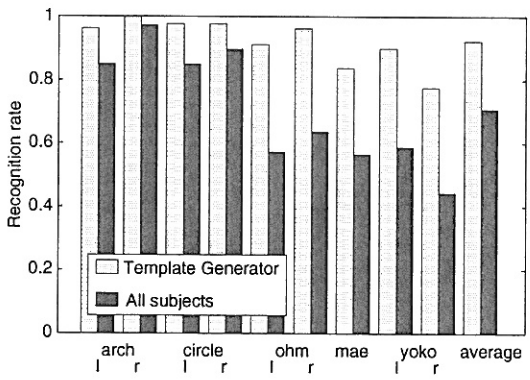


Figure 7: Recognition results of the experiment with 2 cameras

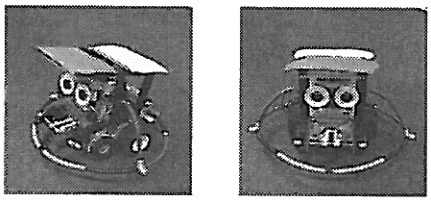


Figure 8: Mobile robot for experiment

We use 2 cameras for gesture recognition (the **gesture recognition system**). The results of gesture recognition are transmitted to the host computer of the **robot control system** through TCP/IP networks. The robot control system can trace the robot position and posture by the camera placed above the field.

The scene image for the operator is taken by the camera which looks down the field diagonally from a point, and the image is displayed on the monitor placed in front of the subject (Fig. 2: scene monitor).

The gestures we use for navigating the robot are the following 4 patterns: swinging the hand forward and backing to the standard position (front), swinging the hand backward and backing to the standard position (back), and rotating the hand clockwise / counterclockwise 2 times (circle_r and circle_l respectively). We prepare the template of these motions for each subject in advance ($Nt = 10$).

We assigned the gestures to robot actions like a radio-control car operation, for example, circle_r for the command "turn right", front for "go forward". But here, the robot does not move continuously, it stops after moving one step and waits for the next command. We set the moving step of robot to about 80[mm] for

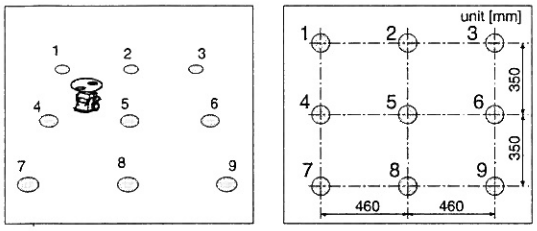
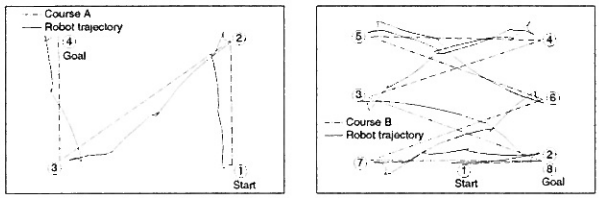


Figure 9: Placement of course markers



Course A Course B

Figure 10: Trajectories of the robot (Subject 1)

forward/backward, and 18[deg] for rotation.

In this experiment, we introduced the method described in Section 5.1.2, in order to cope with inappropriate inputs. In the method, we set the threshold $Th_{ambiguous} = 0.02$.

6.2 Results: Mobile robot navigation

We recorded the trajectory of the mobile robot navigated by each subject and the recognition rate during navigation. In this experiment, we prepared two courses tracing markers on the field as follows: $9 \rightarrow 3 \rightarrow 7 \rightarrow 1$ (course A), and $8 \rightarrow 9 \rightarrow 4 \rightarrow 3 \rightarrow 1 \rightarrow 6 \rightarrow 7 \rightarrow 9$ (course B) (see Fig. 9). The initial direction of the robot was set to be rightward toward the operator (camera as human-eye). The operator navigated the mobile robot along the course observing the robot position through the scene monitor. The arrival of the robot at each marker was checked by the computer of the robot control system using the ceiling camera. We tested 4 subjects for each course.

We show the trajectories of the robot navigated by a subject in Fig. 10. The robot was made of LEGO blocks so that the accuracy was not so high. In this experiment, our robot tended to shift leftward during straight motion, so that the operator had to adjust the way rightward sometimes. In spite of this undesirable condition, the robot were navigated along the course almost successfully.

The rejection rate (the rate of rejected gestures as ambiguous) and miss-recognition rate were about 2% and 3%, respectively. The main reason for miss-recognition was as follows. Sometimes the shape of the gesture was effected by the end position or posture of the previous gesture. In case of front type, when the starting motion was relatively small, the encoding started after the hand was swung forward, and the input was sometimes miss-judged as back type.

7 Conclusion

In this report, we proposed the method of finding gesturing hand motion from video sequence using the center of gravity of the optical-flow field, and the method of encoding motion pattern into the 1-dimensional event code sequence (1DECS) based on motion changes. We recognized the pattern based on string comparison algorithms. The proposed methods can describe and recognize the motion patterns of each person by simple rules whose input can be easily extended. We are now planning to use other recognition methods such as the Hidden Markov Models toward a gesture recognition which does not depend on users[8].

References

[1] J. K. Aggarwal and Q. Cai, "Human Motion Analysis: A Review", *Computer Vision and Image Understanding*, Vol. 73, No. 3, pp. 428–440, March 1999.

[2] R. Cutler and M. Turk, "View-based Interpretation of Real-time Optical Flow for Gesture Recognition", *Proc. the 3rd IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 416–421, 1998.

[3] Y. Iwai et al., "Gesture Recognition from Image Motion Based on Subspace Method and HMM", *Proc. the 3rd Asian Conference on Computer Vision*, Vol. II, pp. 639–646, Lecture Notes in Computer Science 1352, 1998.

[4] Y. Matsumoto et al., "PC-based Hypermachine–A Kernel System for Intelligent Robot Application", *Proc. the 15th Annual Conference of Robotics Society of Japan*, pp. 979–980, 1997 (in Japanese).

[5] A. Nishikawa et al., "Systematic Selection of Local Correlation Parameters for Optical Flow-based Gesture Recognition", *Proc. the 8th IEEE International Workshop on Robot and Human Interaction*, pp. 183–188, 1999.

[6] V. I. Pavlovic et al., "Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 7, pp. 677–695, 1997.

[7] Skiena, Steven S., *The Algorithm Design Manual*, Springer-Verlag, 1997.

[8] Y. Yasui et al., "Gesture Recognition with 1-Dimensional Encoding of Motion Change Events –Improvement of the Recognition Performance by Introducing the Hidden Markov Model–", *Proc. the 2001 IEICE General Conference (Information and Systems II)*, No. D-12-39, p. 206, 2001 (in Japanese).