Track-based Gesture Recognition Method Based on Kinect

Ying Wang*
Computer Information Center, Beijing Institute of Fashion Technology, Beijing, China

ABSTRACT: This paper presents a new gesture track recognition method based on the depth image information received from the Kinect sensor. First, a Kinect sensor is used to obtain the coordinates of a moving arm. Then, the gesture tracks corresponding to these coordinates are analyzed. Matching and recognition of gesture tracks are implemented by performing golden section search. The results show that this track-based method is highly effective in gesture recognition.

Keywords: gesture track recognition; Kinect

1 INTRODUCTION

Kinect [1, 2] is a new game controller technology introduced by Microsoft in November, 2010. Since its launch date, it was evident that Microsoft’s device is transformed not only in computer game but also in many other applications like robotics and virtual reality due to its ability to track movements and voices and even identify faces, all without the need for any additional devices [3]. Kinect (Figure 1) interprets 3D scenes from a continuously-projected infrared structure. It has a webcam like structure and allows users to control and interact with a virtual world through a natural user interface, using gestures, spoken commands or presented objects and images. The device includes a RGB color camera, a depth sensor and a multi-array microphone. It provides the full-body 3D motion capture and the facial and gesture recognition. The depth sensor consists of an infrared laser projector combined with a monochrome CMOS sensor, and allows the Kinect sensor to process 3D scenes in any ambient light condition [2]. The depth sensor technology was developed by Israeli Prime Sense [4]. It interprets 3D scene information from a continuously-projected infrared structured light. A variant of image-based 3D reconstruction was used to recover the depth of the observed points in the 3D scene [3]. Kinect is able to simultaneously recognize up to six people, including two moving players, for motion analysis with a feature extraction of 20 joints per active player.

As a simple and natural way of communication, gesture has become an important means of man-machine interaction [6-9]. A gesture track is to recognize the meaning expressed by a moving arm according to certain rules. The application of gesture track information has good portability and expansibility. However, the segmentation method of the vision-based gesture track recognition is susceptible to the influence of light, background and camera performance, which may result in low recognition rate. Kinect, the motion sensing device designed by Microsoft uses sensors to obtain the position coordinates of human joints and depth data, providing a sound basis for further gesture track recognition. This algorithm performs segmentation and recognition on depth images using the coordinate information and tracks obtained by Kinect sensors. This method is not affected by such factors as light and background, which has enhanced the consistency and robustness of gesture track recognition.

*Corresponding author: jsjwy@bift.edu.cn
2  POSE AND GESTURE

Movement is dynamic while posture is static. Gesture and pose can be respectively understood as movement and posture. Gesture is made up of a series of continuous movement of a body part in the space, such as the act of waving and of swiping. The gesture of swiping will get the response in speed and direction in a Kinect system. To be specific, hands swiping overhead, the rolling direction and speed on the screen will be consistent with that of the hands. Pose is a static relationship of spatial layout of bodies’ different parts. The “universal pause” in the Xbox game which can be achieved by raising hands 45 degrees is the most well-known case.

For simple games, hands and fingers are the body parts which are used most frequently to control scene other than motion sensing games. Hand movement is of the greatest variety, of the easiest identification and of the highest recognition rate in all the body movements, thus becoming the first choice of interactive measure. Hand movement is characterized by nature, simplicity and richness. Gesture recognition can be divided into following categories:

- **Algorism matching:**
  Match simple gesture by tracking the spatial relationship between the body parts with the knowledge of geometry and trigonometry.

- **Template matching:**
  Make a comparison between the sequence of gesture movement and the proposed movement template, and conduct recognition work by measuring their similarity.

- **Neutral network and support vector machine:**
  Neutral network recognition technique is characterized by self-organizing ability, self-learning ability, the uniformity of the distribution and the mode generalization ability. It can effectivelly counter noise and solve incomplete mode. Support vector machine is developed based on statistical theory, having better inference ability than neutral network.

- **Statistical analysis and machine learning:**
  It is one of classification methods to train computer vision technology by collecting eigenvectors. It is also a classification method based on probability and Bayesian maximum similarity theory. This technology cannot be used for the direct recognition of original data. Instead, it can be used to extract specific eigenvectors from a large amount of data.

There are seven universal gestures in Kinect games and applications: wave, hover button, magnet hover button, swiping, slide unlock and universal pause. A train of thoughts on how these gestures’ recognition is realized are summed up.

- **Wave:**
  Wave is usually a method to say hello in the real world. In Kinect games and applications, wave can be the cursor control to activate a screen.
along the negative direction of the Z-axis. A reference point, such as the coordinates of users’ heads, can be chosen for the calculation of the distance of the movement. When the distance of hand movement exceeds the preset threshold, an event like “mouse click” will be triggered.

- **Universal pause:**
As is known to Windows users, the ESC key in the keyboard performs a function of universal pause. The gesture of universal pause is one of the guiding gestures offered by Microsoft. And this gesture is realized by raising one’s left arms at 45 degrees off his body. The algorithm of universal pause is relatively simple. It can be determined based on the angle of one’s left arm off his body and the angle threshold.

### 3 MOTION DETECTION BASED ON SAMPLES

Suppose that movements can be recorded and serialized. If the ongoing movement is one of those that happened before, it can be detected. The main purpose is to quickly determine whether the data of the ongoing movement matches that of any previous movement \([10, 11]\). All movements are in the form of sets of coordinates. However, the coordinates only represent the distance from the sensor. Therefore, there is a need to propose an algorithm for data comparison. In addition, the data should be standardized and converted so that they are in the same reference space.

Suppose that the coordinate series is shown as follows:

\[
P = \{p_0, p_1, \ldots, p_{n-1}\}
\]

- **Generate the movement with a specific number of coordinate points**

Suppose that the number of samples is \(M\). The distance between two samples is defined as \(L_{avg} = (n - 1)/(M - 1)\). The distance between \(p_{i-1}\) and \(p_i\) is defined as \(L_{i,i-1}\). In the sampling process of coordinate series according to the number of samples, the new coordinates of sampled points need to be recalculated. The recalculation is implemented by traversing points corresponding to coordinate series. If the distance between the current point and the last sampling point cannot reach the sampling point, the traversal continues. If the distance reaches or exceeds the sampling point, the coordinate point needs to be recalculated. So a variable \(L_{cur}\) representing the accumulative distance during the traversal needs to be defined. When \(L_{cur} + L_{i,i-1} \geq L_{avg}\), the new coordinate point is shown as follows:

\[
p'_i = p_{i-1} + (L_{avg} - L_{cur}/L_{i,i-1}) \times (p_i - p_{i-1})
\]

The new coordinate series is shown as follows:

\[
P' = \{p'_0, p'_1, \ldots, p'_{n-1}\}
\]

- **Rotate the coordinate point so that the first point is at 0 degree**

In order that the two matched series have the same starting point, the coordinate series are rotated until the first coordinate point is at 0 degree. Then the center coordinates of the coordinate series are calculated. The direction angle of the first coordinate point relative to center coordinates is obtained. A new coordinate series is generated by setting this angle as the direction angle and using the rotation matrix in the two-dimensional space.

- **Map the coordinates to the specified coordinate level \([0, 1]\)**

As the value range of the coordinate series has not been unified, inaccuracy may arise. So, the coordinates need to be set within the interval \([0, 1]\).

- **Standardize the origin of the series of movement points**

After this standardization, the matching will be performed. The coordinate series with the same length are resized in the unified scale, direction and centers. Start the gesture:

\[\text{Figure 2. Standardization of the gesture track.}\]
Golden section search\textsuperscript{[15]} is a technique for finding the extrema by successively narrowing the range of values of a monotone continuous function inside which the extremum is known to exist. The technique derives its name from the fact that the algorithm maintains the function values for triples of points whose distances form a golden ratio. Golden section search is more real-time and convenient than derivatives in finding the extremum of a function.

Consider the function in the interval \([a, b]\). Suppose that \(f(x)\) is a continuous unimodal function in \([a, b]\), then \(f(x)\) has only one minimum or maximum in \([a, b]\). The best way to find this extremum is to use the bisection method. Similar to the bisection method, GSS algorithm narrows down the interval in which the extremum exists by comparing function values. Two objectives must be fulfilled in designing the algorithm. First, the optimal interval narrowing factor can be chosen to make \([a, b]\) as the new region of search. However, two calculations need to be performed to solve the function in each narrowing operation. This is not the optimal method.

Enlightened by the bisection method, one choice is to calculate the intermediate point \(m = (a + b)/2\). Then find the values of \(x_1\) and \(x_2\) and, define \(x_1 = m - \delta/2\) and \(x_2 = m + \delta/2\). \(\delta\) is a small parameter that makes \(f(x_1) \neq f(x_2)\). If \((x_1) < f(x_2)\), we reserve \([a, x_1]\), otherwise we set \([x_2, b]\) as the new region of search. However, two calculations need to be performed to solve the function in each narrowing operation. This is not the optimal method.

Figure 3 shows one step for finding the minimum. The function value of \(f(x)\) is on the vertical coordinate axis. The parameter \(x\) is on the horizontal coordinate axis. The values of three points on the function \(f(x)\) have been obtained. They are \(x_1, x_2\) and \(x_3\). It’s thus clear that \(f(x_2)\) is smaller than \(f(x_1)\) and \(f(x_3)\). Obviously, the minimum exists between \(x_1\) and \(x_3\).

The next step is to calculate the value of the function at another point \(x_4\). It is more efficient to choose \(x_4\) in the largest interval, for example, between \(x_2\) and \(x_3\). It can be seen from the figure that if the function value exists in \(f_{da}\), the minimum will exist between \(x_2\) and \(x_4\), and a new group of points will be \(x_1, x_2\) and \(x_4\). If the function value is \(f_{db}\), the new group of points will be \(x_2, x_3\) and \(x_3\). So, in both cases, we can build a new narrower search interval that contains the function’s minimum.

Consider a function \(f\) over the interval \([a, b]\). We assume that \(f(x)\) is continuous over \([a, b]\); and that \(f(x)\) is unimodal over \([a, b]\), i.e. \(f(x)\) has only one minimum in \([a, b]\). Note that the method applies as well to find the maximum\textsuperscript{[13, 14, 15]}.

The conditions above remind us to the bisection method and we will apply a similar idea: Narrow the interval that contains the minimum comparing function values. In designing the method, we seek to satisfy two goals. We want an optimal reduction factor for the search interval and minimal number of function calls. With these goals in mind, we are going to determine the location to evaluate the function.

One choice inspired by the bisection method would be to compute the midpoint \(m = (a + b)/2\) and to evaluate at \(x_1 = m - \delta/2\) and \(x_2 = m + \delta/2\), for some small value \(\delta\) for which \(f(x_1) \neq f(x_2)\). If \(f(x_1) < f(x_2)\), then we are left with \([a, x_1]\), otherwise \([x_2, b]\) is our new search interval. While this halves the search interval in each step, we must take two new function evaluation in each step. Therefore, this is not optimal.

We only need to perform a new function evaluation in each step. Furthermore, we need a constant reduction factor, say, \(c\), for the size of the interval.

In order to increase the efficiency of the algorithm, the function solving process operation should be performed only once in each step. Be relative to the current search interval, a constant factor \(c\) is defined. And there are two cases for \(x_1\) and \(x_2\) in \([a, b]\).

If \((x_1) < f(x_2)\), we make \([a, b] := [a, x_1]\) and make the following adjustment in the interval:

\[
x_2 - a = c(b - a) \Rightarrow x_2 = a + cb - ca \Rightarrow x_2 = (1 - c)a + cb(2)
\]

If \((x_1) > f(x_2)\), we make \([a, b] := [x_1, b]\) and make the following adjustment in the interval:

\[
b - x_1 = c(b - a) \Rightarrow -x_1 = cb - ca - b \Rightarrow x_1 = ca + (1 - c)b(3)
\]

So long as \(C\) is known, position \(x_1\) and \(x_2\) can be obtained. Without loss of generality, we consider the case \((x_1) < f(x_2)\). To simplify the calculation, we suppose that \([a, b] = [0, 1]\).

If \((x_1) < f(x_2)\), we set \(x_1 = 1 - c\) repeatedly and determine whether the next value should be selected from the left or right of \(1 - c\).

Suppose that a function value is obtained from the left of \(x_1 = 1 - c\), then \(x_1\) is the left boundary of

\[
1 - c = (1 - c)0 + cc \Rightarrow c^2 + c - 1 = 0 \quad (4)
\]

The positive root \(c = (-1 + \sqrt{5})/2\) is approximately equal to 0.6180.

Suppose that a function value is obtained from the right of \(x_1 = 1 - c\), then \(x_1\) is the left boundary of

Figure 3. Standardization of the gesture track.
The expression of \( x_1 \) can be written in two forms. \((a = 0 \text{ and } b = c \text{ in the expression where } x_1 \text{ is used})\):

\[
1 - c = c_0 + (1 - c)c \Rightarrow (1 - c)^2 = 0
\] (5)

The root of equation (5) is 1. The interval is not narrowed down. This possibility is thus ruled out. Two rules are obtained: If \( f(x_1) < f(x_2) \), we reserve \( x_1 \), make \( x_2 \) and calculate a new value for \( x_1 \) through using equation (3). If \( f(x_1) > f(x_2) \), we reserve \( x_2 \), retain it as \( x_1 \) and calculate a new value for \( x_2 \) through using equation (2).

4 RESULTS

Two experiments are conducted using this algorithm to recognize two kinds of gestures, namely, left-right sway of hand and circular gesture. The gesture tracks are collected from five people. Each person repeats each gesture for ten times. As shown in Figure 4, the hand moving leftwards in the left picture represents a leftward sway and the red dots represent the moving track of the hand. The set of points is used as the sample for recognition. The left picture shows a circular gesture. It can be seen that the sample highly matches the gesture.

![Leftwards sway of hand](image)

(a) Leftwards sway of hand

![Circular gesture recognition](image)

(b) Circular gesture recognition

Figure 4. Recognition performance

Table 1 gives out the recognition rate of the two gestures by the five participants. The results show that this algorithm has a high recognition rate which is able to meet demands of practical use.

<table>
<thead>
<tr>
<th>Gesture</th>
<th>Leftwards sway</th>
<th>Rightwards sway</th>
<th>Circular gesture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Times of gesture</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>Times of recognition</td>
<td>50</td>
<td>49</td>
<td>46</td>
</tr>
<tr>
<td>Recognition rate</td>
<td>100%</td>
<td>98%</td>
<td>92%</td>
</tr>
</tbody>
</table>

5 CONCLUSION

In recent years, gesture has increasingly become an important way of man-machine interaction for its ease of use. Based on certain rules, the meaning carried by a gesture track can be recognized. Therefore, the application of gesture track information is highly portable and expansible. The traditional gesture track recognition which is vision-based contains a variety of algorithms. This paper uses the Kinect sensors to obtain depth data and perform segmentation and recognition over depth images. As this method is free from the influence of light and background, it has increased the consistency and robustness of gesture track recognition.

ACKNOWLEDGEMENT

This paper is sponsored by the Youth Innovation Program of Beijing Institute of Fashion Technology (GN: QTXM02140201/054) and the Beijing Education Commission Scientific Research Support Program (KYJH02150201).

REFERENCES


