Pattern Recognition Letters 100 (2017) 83–88

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Detection of gestures without begin and end markers by fitting into Bézier curves with least squares method

Michał Słapek*, Szczepan Paszkiel*

Opole University of Technology, 76 Prószkowska Street, Opole, 45-758, Poland

ARTICLE INFO

Article history:
Received 24 April 2017
Available online 9 October 2017

Keywords:
Motion analysis
Gesture recognition system
Bezier curves
Human-computer interaction

ABSTRACT

Gesture recognition is used to obtain natural way of device control. This paper describes a method of extracting predefined gestures from series of points without a marker, when the gesture begins or ends. Presents a way to fit points to Bézier curve with constant memory usage. The procedure is based on least–squares method. Concludes with usage of algorithm to detect loops in series of points.

1. Introduction

The problem is to detect among an incoming sequence of points predefined set of gestures, which can be described with a cubic Bézier curve. The solution there discussed falls into *curve fitting* method among hand gesture recognition tools [6]. It is based on assumption, that input points are evenly spaced. However, in certain cases it can lead to invalid results (Section 3.2). The algorithm runs with constant memory space usage.

It was already proposed to use curve fitting to recognize gestures [8] with markers from user when gesture begins and ends. This paper considers variation of the solution – describes a method of extracting predefined gestures from series of points without a marker, when the gesture begins or ends (Section 4.1) and presents derivation of formula for fitting the curve with implementation discussion.

1.1. Bézier curve definition

Bézier curve is parametric formula describing path on the plane [5].

$$\mathbf{B} : [0, 1] \to \mathbb{R}^2, \quad \mathbf{B}(t) = (x, y) \tag{1}$$

Coordinates of points are described with equations:

$$x = \mathrm{B}_x(t) = \sum_{j=0}^{k} a_{xj} t^j, \qquad y = \mathrm{B}_y(t) = \sum_{j=0}^{k} a_{yj} t^j \tag{2}$$

for $t \in [0, 1]$ and $x, y \in \mathbb{R}$. Functions $\mathrm{B}_x(t)$ and $\mathrm{B}_y(t)$ from Eq. (2) are kth order polynomials when the curve is kth order Bézier curve. Fig. 1 presents example of Bézier curve. Notice, that this paper focuses on power basis form of curve.

The curve is chosen to minimize distance (on Euclidean plane) from each point $\mathbf{Q}_i$ to corresponding point on the curve $\mathbf{B}(\bar{t}_i)$, where $\bar{t}_i$ is chosen partition of the curve (Section 5.2). $\overline{\mathrm{R}}^2$ denotes sum of squares of these distances.

Some sections treat dimensions independently – calculations over $\mathrm{B} : [0, 1] \to \mathbb{R}$ will be valid for any dimension. Then in this context $\overline{\mathrm{R}}^2$ is sum of squares of distances on that dimension (Section 3.2).

Further sections describe detection of gesture beginning and end. The process is configured by parameters as q frame length and s gesture span (Section 5.2).

1.2. Solution overview

Problem is divided into following subproblems:

- Capture of input, which consists of sequence of points on plane (Section 2).
- Calculation of Bézier curve matching input (Section 3).
- Detection of gestures based on matched Bézier curves (Section 4).

2. Gathering input

As an input is given a sequence of n points:

$$\mathbf{Q}_i = (x_i, y_i), \quad i = 1, 2, 3, \ldots, n \tag{3}$$

* Corresponding authors.
E-mail addresses: m.slapek@student.po.edu.pl (M. Słapek), s.paszkiel@po.opole.pl (S. Paszkiel).

https://doi.org/10.1016/j.patrec.2017.10.006
0167-8655/

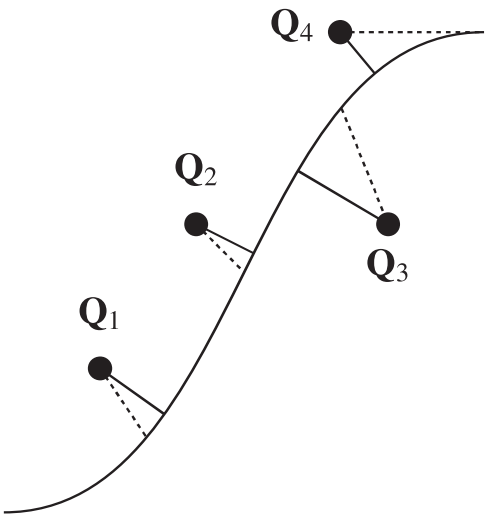

Fig. 1. Distances of points to Bézier curve. Solid line presents distance from point to the curve, according to Eq. (4). Dashed line presents distance to assigned point according to Section 3.2.

Points could be gathered from many sources, such as mouse cursor or touchscreen. There exist devices capturing hand position. Other source of points could be a path on image stored as pixel matrix.

The point index is expected to be correlated with time. For instance, points could be probed with constant frequency 15 Hz. Therefore, rest of paper will identify point index with time.

User can signal beginning of gesture with opened hand or with press of touchscreen. In this paper, we also consider case, when user does not signal beginning of gesture, i.e. the algorithm must decide, when begins and ends a gesture (Section 4.1).

3. Fitting points to Bézier curve

Fitting curve to *sequence* of points is the main problem of the paper. This section considers fitting to *one* Bézier curve.

Coefficients of polynomials $B_x(t)$ and $B_y(t)$ are output (compare with Eq. (2)).

3.1. Minimalizing sum of distances

Distance from point $\mathbf{Q}_i$ to curve $\mathbf{B} : [0, 1] \to \mathbb{R}^2$ can be described with formula:

$$d_i = \min_{0 \le t \le 1} \|\mathbf{B}(t) - \mathbf{Q}_i\| \tag{4}$$

Distances are presented in Fig. 1 with sample curve.

Let's assign to each point $\mathbf{Q}_i$ value t_i, where t_i is equal to parameter t from Eq. (4). Substituting (2), distance becomes

$$\begin{aligned} d_i &= \|\mathbf{B}(t_i) - \mathbf{Q}_i\| \\ &= \sqrt{\left(B_x(t_i) - x_i\right)^2 + \left(B_y(t_i) - y_i\right)^2} \end{aligned} \tag{5}$$

In order to fit curve to set of points, this algorithm will minimize sum of distances raised to square – residual R^2:

$$R^2 = \sum_{i=1}^{n} d_i^2. \tag{6}$$

3.2. Simplifying distance calculations

Instead of calculating distances from the whole curve, we can consider distance from *particular* points on the curve, as shown in Eq. (5). Assume that scanning returns points in such order that sequence t_i is increasing. Moreover, assume that points are evenly spaced.

One of the methods of t_i approximation is equal spacing [5].

$$\bar{t}_i = \frac{i}{n} \tag{7}$$

It is reasonable approximation of true sequence t_i under assumption that consecutive points in the sequence have equal distances.[1] Notice, that this method is not recommended for unevenly spaced data [5]. For such case, there are better methods of choosing $\bar{t}_i$ [3]. There exists algorithm choosing $\bar{t}_i$ with multiple Newton–Raphson iterations [7]. Such improvements are important in graphics editors – however, for purposes of gesture detection they are not crucial (Section 5).

Fig. 1 presents distances calculated with approximation from this section. From this point the approximation will be used for the rest of paper. Thus, the value to minimize from Eq. (6) becomes

$$\bar{R}^2 = \sum_{i=1}^{n} \left(B_x(\bar{t}_i) - x_i\right)^2 + \sum_{i=1}^{n} \left(B_y(\bar{t}_i) - y_i\right)^2. \tag{8}$$

Notice that summands for each dimension are independent.[2] Thus to minimize $\bar{R}^2$, it is sufficient to minimize $\sum_{i=1}^{n} \left(B_x(\bar{t}_i) - x_i\right)^2$ and $\sum_{i=1}^{n} \left(B_y(\bar{t}_i) - y_i\right)^2$. This conclusion would be invalid without approximation from Eq. (7).

3.3. Minimizing for single dimension

Calculations performed in this subsection are dimension independent. Following calculations are performed for each dimension individually – v must be replaced with dimension label. Let v_j be coordinate of jth point in given dimension. Single dimension in Bézier curve is described by polynomial $B(t)$ with coefficients (a_i). According to the Eq. (8), the value to minimize is

$$\bar{R}^2 = \sum_{i=1}^{n} \left(B(\bar{t}_i) - v_i\right)^2. \tag{9}$$

Describing with matrices gives

$$\mathbf{v} = X\mathbf{a} + \mathbf{R} \tag{10}$$

where

$$\mathbf{a} = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_k \end{bmatrix}, \qquad \mathbf{v} = \begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix}, \tag{11}$$

$$X = \begin{bmatrix} \bar{t}_1^{\,0} & \bar{t}_1^{\,1} & \dots & \bar{t}_1^{\,k} \\ \bar{t}_2^{\,0} & \bar{t}_2^{\,1} & \dots & \bar{t}_2^{\,k} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{t}_n^{\,0} & \bar{t}_n^{\,1} & \dots & \bar{t}_n^{\,k} \end{bmatrix}. \tag{12}$$

The residual (9) is equal to the square of the length of vector $\mathbf{R}$. Thus, minimizing of that value is instance of least squares problem, whose solution is described by the following system of linear equations:

$$X^T X\mathbf{a} = X^T\mathbf{v} \tag{13}$$

[1] There is a next simplification: not all Bézier curves can be described with natural parametrizations using polynomials. In other words, **B** is not function of distance nor it's linear transformation in general case.

[2] Calculations are valid for any number of dimensions.