SIViP (2012) 6:19–24
DOI 10.1007/s11760-010-0165-9

ORIGINAL PAPER

A new method for video data compression by quadratic Bézier curve fitting

Murtaza Ali Khan

Received: 11 December 2008 / Revised: 24 May 2010 / Accepted: 25 May 2010 / Published online: 10 June 2010

Abstract This paper presents a new and efficient method for video data compression using quadratic Bézier curve fitting. The method treats the luminance or color variations of a spatial location in a sequence of frames as input points in Euclidean space R^1 or R^3. The input points are approximated using quadratic Bézier least square fitting. The output data consists of quadratic Bézier control points and difference between original and fitted data. Video data compression is the main application of proposed method. It is shown that entropy of output data is significantly less than the entropy of input data. The method can be applied to 1-D space like luminance and chrominance components separately or 3-D color spaces such as RGB and YC_bC_r.

Keywords Video data · Fitting · Approximation · Interpolation · Compression · Quadratic Bézier curve

1 Introduction

Spline and curve are widely used in computer-aided design and computer graphics because of the simplicity of their construction, accuracy of evaluation and their capability to approximate complex shapes through curve fitting and interactive curve design [1]. Spline can fit large number of data points with far less number of control points. Control points can be encoded by some appropriate encoding technique. During the decoding process, fitted data points are regenerated by spline interpolation of control points.

In order to understand how quadratic Bézier curve can be used to fit video data, let first understand the nature of video data. Digital video data consists of sequence of frames (images). Each frame consists of rectangular 2-D array of pixels. One-dimensional *luminance* or 3-D RGB values in a sequence of frames are associated with each pixel. *Luminance* or RGB values of a pixel can be considered as points in Euclidean space R^1 or R^3, respectively. Let a video consists of a sequence of n frames. Frame width and height are W and H, respectively. Then for each spatial location (x_i, y_j), $1 \leq i \leq W$, $1 \leq j \leq H$, we have temporal video data in n frames, $\{p_1, p_2, \ldots, p_n\}$, i.e., $p_j = I_j$ for *luminance* or $p_j = \left(R_j, G_j, B_j\right)$ for RGB, where $1 \leq j \leq n$. Figure1 shows RGB variation of a spatial location in 80 frames of a video.

We considered temporal variations of luminance or color values for each spatial location in a sequence of frames as input data. The input data is fitted with far less number of control points (output data) of quadratic Bézier curve. An important factor in fitting of data via quadratic Bézier curve is finding least number of control points. We achieved this goal by least square fitting.

Organization of the rest of the paper is as follows: Related work is discussed in Sect. 2, mathematical model of quadratic Bézier curve is briefly described in Sect. 3. Section 4 elaborates the fitting strategy. Application of our method is described in Sect. 5. Selected experiments and results are presented in Sect. 6. Section 7 analyzes results and gives insight view of the proposed method. Final concluding remarks are in Sect. 8.

2 Related work

Approximation and compression of data using parametric curves particularly cubic spline is explored by many authors [3,6,8–10,14,15] et al. The method presented by [6] approximates the video data using parametric line and

M. A. Khan (✉)
Faculty of Information Technology,
Royal University for Women, West Riffa, Bahrain
e-mail: mkhan@ruw.edu.bh

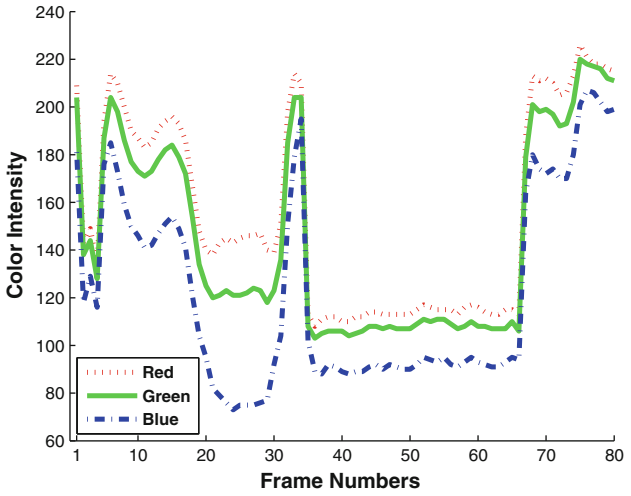

Fig. 1 *RGB* temporal variations of a spatial location (50, 50) in 80 frames of a video sequence

Natural cubic spline by combining the group of pixel together as a block and the applies fitting to block. The approach we adopted in this paper is based on quadratic Bézier curve fitting at pixel level. Pixel level fitting yields more precise control of accuracy compare to block level fitting. In contrast to [6], the scheme presented in this paper is lossless compression. A video object encoding algorithm based on the curve fitting trajectory of video object moving edges pixels is presented by [3]. The algorithm of [3] is suitable, if the objects in the video are not moving quickly such as in video conferencing or surveillance. Our method does not encode the edges of objects only; but it applies fitting to luminance/color variations of every pixel. Non-spline based temporal correlation reductions methods are based on motion estimation via translating block matching algorithms (BMAs) [2,7,12]. In a typical BMA, a frame is divided into rectangular blocks of pixels. Then the current (predicted) block is matched against blocks in the previous frame, for a maximum motion displacement of w pixels. The best match on the basis of a *mean absolute error* (MAE) criterion yields displacement relative to current block called motion vector. Predicted frame is obtained by blocks in reference frame and corresponding motion vectors [4,11,13]. In contrast to BMAs, we do not find matching pixel or matching block. We adopted different approach of fitting i.e., approximating the change in color or luminance values of each pixel in a sequence of frames, at the fixed spatial location (without translation of block/pixel), by quadratic Bézier curve fitting then finding the difference between original and approximating values. The approach used by [9] is based on EdgeBreaker and principal component analysis (PCA), and it exploits both spatial and temporal coherence. [14] presented a method of image data compression using cubic convolution spline interpolation. Another method of image compression (medical images) using cubic spline is presented by [10]. Cubic spline is more appropriate for image compression but less feasible for video data compression due to its computational cost. We used low degree quadratic Bézier curve, and therefore it is more efficient than cubic spline. Contour data compression method using Curvature Scale Space technique and Hermite curves is proposed by [15]. Proposed method of [8] uses multilevel B-Splines to approximate scattered data. [17,5,18] used spline for shape coding of objects in video data. Our method is not limited to shape coding of objects, but it provides a way to code every pixel in the video including background, foreground and objects. Due to large size of video data, it is also desirable that fitting process is automated. In our scheme, the user has just to initialize a single parameter, then the rest of the steps are fully automated.

3 Quadratic Bézier curve (QBC)

Quadratic Bézier curve (QBC) is a C^0 continuous curve. A QBC segment is defined by three control points, i.e., P_0, P_1 and P_2, as shown in Fig. 2. P_0 and P_2 are called *end control points* (ECP), while P_1 is called a *middle control point* (MCP). A QBC passes through its *end control points*, while a *middle control point* is used to control the shape of the curve. To generate continuous quadratic Bézier curves that interpolate $k+1$ points, k curve segments are used. Equation of a QBC segment can be written as follows:

$$Q(t_i) = (1-t_i)^2 P_0 + 2t_i (1-t_i) P_1 + t_i^2 P_2, \quad (1)$$

where t_i is a parameter of interpolation , $0 \leq t_i \leq 1$. In order to generate n points between P_0 and P_2 inclusive, the parameter t_i is divided into $(n-1)$ intervals between 0 and 1 inclusive, and $Q(t_i)$ is evaluated at n values of t_i. Since a QBC passes through its first and last control points, therefore $Q(0) = P_0$ and $Q(1) = P_2$. Figure 3 shows multi-segment quadratic Bézier curves, where the 2-D data, i.e., (x, y)

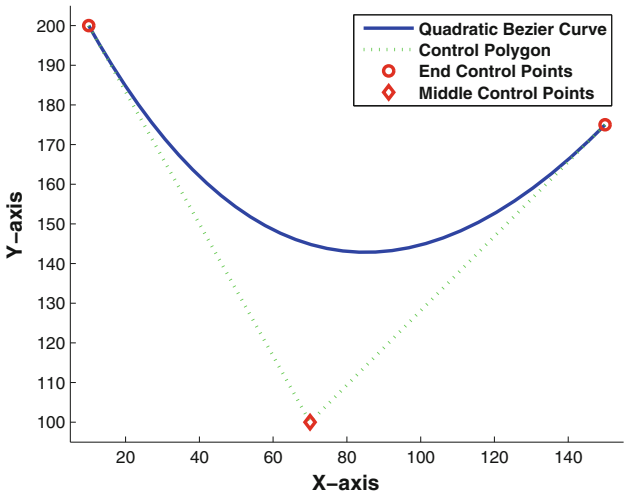

Fig. 2 A quadratic Bézier curve segment

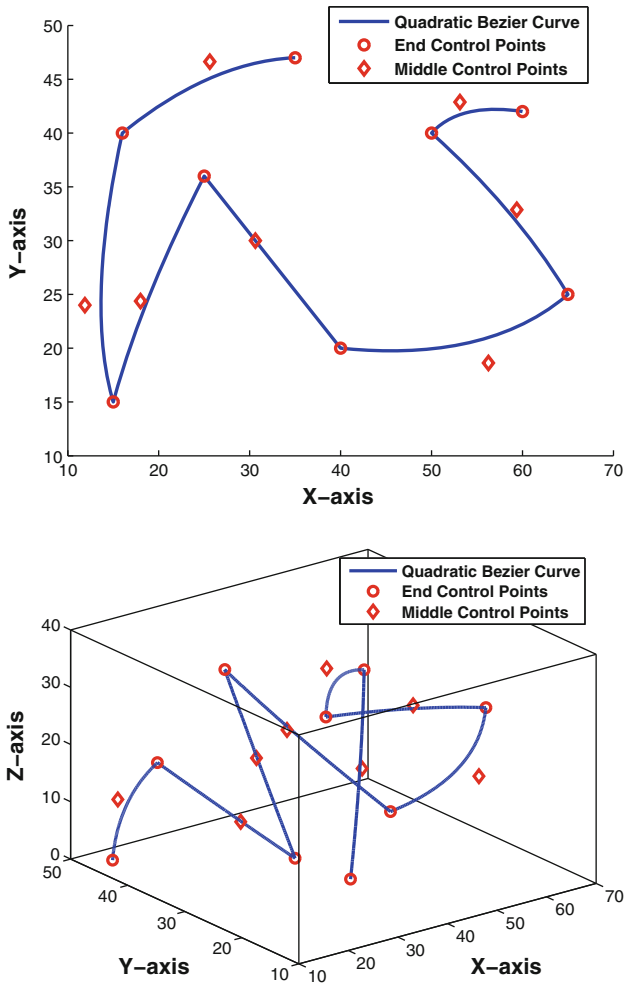

Fig. 3 Multi-segment quadratic Bézier curves in 2-D and 3-D space

values and 3-D data, i.e., (x, y, z) values are taken from non-video source.

4 Fitting strategy

This section describes the strategy of our algorithm to fit the video data by quadratic Bézier curve. Fitting process is applied to temporal data of each spatial location individually. Let n is the number of frames in a video sequence, let W and H are width and height of a frame respectively. Let luminance or color value of a spatial location (x, y), $1 \leq x \leq W$, $1 \leq y \leq H$, at frame i is p_i, where $0 \leq p_i \leq 255$ and $1 \leq i \leq n$. We have to approximate the n values of each spatial location by quadratic Bézier curve. Now we describe the fitting process of an arbitrary spatial location (x_i, y_i). The temporal data of (x_i, y_i) in n frames is $O = \{p_1, p_2, \ldots, p_n\}$. As an input to algorithm the user specifies *breakpoint interval* δ. Luminance or color values of a spatial location after every δ^{th} frames are taken as a breakpoint (control point), e.g., $\delta = 8$ then set of breakpoints is $BP = \{p_1, p_9, p_{17}, p_{25}, \ldots, p_n\}$ (luminance or color values of last frame is always taken as a breakpoint). The fitting process divides the data into segments based on breakpoints, i.e., $S = \{S_1, S_2, \ldots, S_{u-1}\}$. A segment is a set of all points (luminance or color values) between two adjacent breakpoints, e.g., $S_1 = \{p_1, p_2, \ldots, p_9\}$, $S_2 = \{p_9, p_{10}, \ldots, p_{17}\}$.

Each segment is fitted (approximated) by a quadratic Bézier curve. The first and the last breakpoints of a segment are taken as *end control points* (ECP) i.e., P_0 and P_2 of quadratic Bézier curve, while *middle control point* (MCP) i.e., P_1 is obtained by least square method. Least square method gives the *best* value of the *middle control point* that minimizes the squared distance between the original and the fitted data. If there are m data points in a segment, and O_i and $Q(t_i)$ are values of original and approximated points, respectively, then we can write the least square equation as follows:

$$U = \sum_{i=1}^{m} [O_i - Q(t_i)]^2. \tag{2}$$

Substituting the value of $Q(t_i)$ from Eq. (1) in Eq. (2) yields:

$$U = \sum_{i=1}^{m} [p_i - (1-t_i)^2 P_0 + 2t_i(1-t_i)P_1 + t_i^2 P_2]^2. \tag{3}$$

To find value of P_1 differentiating Eq. (3) partially with respect to P_1 yields:

$$\frac{\partial U}{\partial P_1} = 0. \tag{4}$$

Solving Eq. (4) for P_1 gives:

$$P_1 = \frac{\sum_{i=1}^{m} \left[p_i - (1-t_i)^2 P_0 - t_i^2 P_2\right]}{\sum_{i=1}^{n} 2t_i(1-t_i)}. \tag{5}$$

Once all the three control points, i.e., P_0, P_1 and P_2 are known, then approximated data of a segment using Bézier curve is obtained using Eq. (1). Same procedure is repeated for each segment. This yields n values of approximated data, $Q = \{q_1, q_2, \ldots, q_n\}$.

Let us take an example of a segment S_4, the input data is $O_{S_4} = \{221, 224, 222, 223, 226, 225, 225, 225, 224\}$ with $\delta = 8$. We take first and last point of O_{S_4} as end control points of quadratic Bézier curve, i.e., $P_0 = 221$ and $P_2 = 224$, while middle control point, i.e., $P_1 = 227$ is determined by least square method using Eq. 5. Next, we have to find the interpolated data using Eq. 1. The number of points in O_{S_4} is 9; therefore we divide the parameter t_i into 8 uniform space intervals or 9 values, i.e.,

$$t_i = \{0, 0.125, 0.25, 0.375, 0.5, 0.625, 0.75, 0.875, 1.0\}.$$

By substituting the values of P_0, P_1 and P_2 in Eq. 1 and evaluating it at each value of t_i we obtain the interpolated data, $Q_{S_4} = \{221, 222, 223, 224, 225, 225, 225, 225, 224\}$.

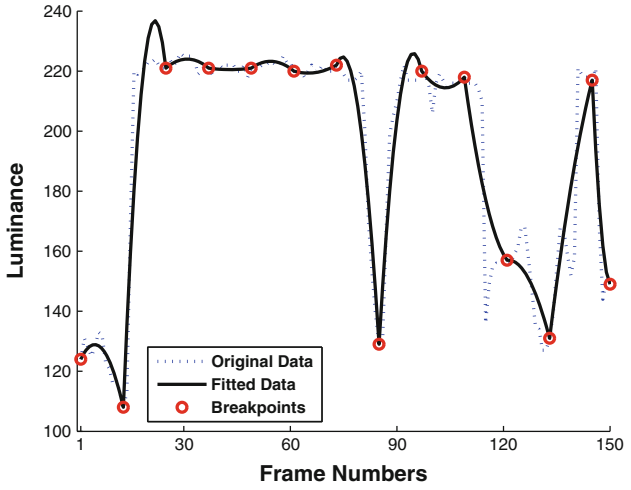

Fig. 4 Quadratic Bézier curve least square fitting to luminance values of spatial location (50, 50) in 150 frames of a video sequence

Note that the first and last points of input data and interpolated data are always same, because $Q(t_i = 0) = P_0$ and $Q(t_i = 1) = P_2$. Interpolated points other than first and last points may or may not have the same values as corresponding points of input data.

The above described fitting process is applied to luminance or color variations in temporal dimension of each spatial location separately. Figure 4 shows quadratic Bézier curve least square fitting to luminance values of a spatial location in 150 frames of a video sequence. Since breakpoint interval, i.e., δ is same for all segments of all spatial locations. Therefore, collectively all the *end control points* after every δ frame constitute a *keyframe of end control points* (KFE). Similarly, collectively all the *middle control points* between two adjacent KFE constitute a *keyframe of middle control points* (KFM).

The output data produced by our method that need to be store is: (1) *keyframes of end control points*, i.e., KFE, (2) *keyframes of middle control points*, i.e., KFM and (3) the difference between original and quadratic Bézier approximated (interpolated) frames other than keyframes, i.e., *frame difference* (FD). In order to reconstruct the original video data without any loss, first interpolated frames are generated using KFE and KFM, then adding FD to interpolated frames reproduces the original video frames.

5 Application

The most important application of our method is data compression. A fundamental approach of prevalent video data compression techniques such as MPEG-1, MPEG-2 and H.263. [4,16] is to reduce the entropy of data by applying Discrete Cosine Transform. Data with reduced entropy can be encoded with less numbers of bits. In our method, the overall entropy of KFE, KFM and FD is much less than the entropy of original video data; consequently, it can be encoded with less number of bits. This less entropy of output data is mainly due to the fact that quadratic Bézier curve approximates the original video data with quite good level of accuracy. Because least square technique gives the optimal value of middle control point that minimizes the squared distance between original video data and approximated data. If we take any arbitrary value of middle control point, then the fitting accuracy would be much less, and the pixel values of approximated data would spread to very large domain (range), this would increase the entropy. Consequently due to our least square fitting technique, the difference between original and quadratic Bézier approximated data has values confined in very small domain (range) compared to the values in original video data. Therefore, the method we presented can be used for lossless video compression for video archives. It is also possible to use our method for lossy video data compression by using it as a preprocessing step of existing lossy video coding methods.

6 Experiments and results

We have applied our method on several naturally recorded and synthetically created video sequences. Table 1 provides details of environment of experiments. We compared the entropy and encoding time of our method with full search (FS) and diamond search (DS) motion estimation methods. We did not apply any method to remove the spatial redundancy (such as DCT), since at this stage only temporal redundancy removal is considered. For motion estimation methods, we took reference frame interval 8, macro-block size 16×16 and range of search window ± 7 in both horizontal and vertical directions. We compute the entropy of original video data, entropy of motion estimation methods (reference frames, motion compensated residual frames and motion vectors) and the entropy of video data represented by our method (KFE, KFM and FD). The entropy of a single video frame is computed as follows:

$$Entropy = -\sum_{j=1}^{J} P(a_j) \log P(a_j), \tag{6}$$

where J is the unique number of symbols (pixel values) in the source (frame), $P(a_j)$ is the probability of the occurrence

Table 1 Environment of experiments

Hardware	Inter Core Duo 2.4 GHz
Operating system	Windows XP SP3
Programming language	MATLAB 7.0.4 SP2

Table 2 Details of input video sequences

Video name	Video no.	Format	Size	Frames
Hall and monitor	V1	Luminance	352 × 288	96
Foreman	V2	Luminance	352 × 288	96
Dinosaur	V3	RGB	352 × 288	96
Cloud	V4	RGB	352 × 240	96
Mobile and calendar	V5	RGB	352 × 240	96

Fig. 5 Hall and Monitor sequence

Fig. 6 Foreman sequence

of symbol a_j. The entropy of complete video sequence is the mean of entropies of all the frames.

Table 2 gives the details of selected input video sequences whose results are presented in this paper. Figures 5–9 show the 21th frame of input video sequences. Table 3 gives the details of entropy and encoding time comparison of our method with full search motion estimation method. In terms of entropy, our method performed better for three video sequences, while FS and DS methods preformed better for two video sequences. In terms of encoding time, our method performed better than FS but less than DS for all the video sequences.

Fig. 7 Dinosaur sequence

Fig. 8 Cloud sequence

Fig. 9 Mobile and Calendar sequence

7 Discussion

Dinosaur and *Cloud* are synthetically created video sequences, while *Hall & Monitor*, *Foreman* and *Mobile & Calendar* are naturally recorded video sequences. *Hall & Monitor*

Table 3 Entropy and encoding time comparison

Video no.	E_A (bpp)	E_B (bpp)	E_C (bpp)	E_D (bpp)	T_B/T_D	T_C/T_D
V1	7.233	4.182	4.184	3.918	1.430	0.171
V2	7.228	4.316	4.350	5.124	1.441	0.198
V3	7.163	3.085	3.086	2.736	3.270	0.278
V4	7.567	4.341	4.342	4.032	3.012	0.288
V5	7.627	6.137	6.140	6.653	2.842	0.272

E_A Entropy of original video data, E_B Entropy of video data by full search method, E_C Entropy of video data by diamond search method, for E_B and E_C macro-block size 16×16, search range=7, and $\delta = 8$, E_D Entropy of video data by our method, $\delta = 8$. Time ratios: T_B/T_D, T_C/T_D; T_B Encoding time by full search method, T_C Encoding time by diamond search method, T_D Encoding time by our method

and *Foreman* sequences have only luminance component, while the other three sequences have RGB components. Our method reduces the entropy of all types of video sequences. The reduction in entropy for synthetically created video sequences is higher than that in naturally recorded video sequences. Because, usually temporal variations of synthetic video sequences show lesser fluctuations and can be fitted with less number of control points. The method we presented can be applied to 3-D color spaces such as RGB, YC_bC_r or HSV or 1-D space like luminance or chrominance components separately. The only parameter that has to be set is δ. From experiments, we found that appropriate range of δ is from 8 to 12.

8 Conclusion

We presented an efficient method for video data compression using quadratic Bézier curve fitting. The method approximates the luminance or color variations of fixed spatial locations in a sequence of frames by quadratic Bézier curve. We described the least square technique to minimize the squared distance between the original and the fitted video data. Experimental results show that the method yields very good results both in terms of entropy reduction and encoding time for both naturally recorded and synthetically created video sequences.

9 Future work

H.264, which is one of the most modern compression techniques, has lossless macroblock coding features. Since the proposed method is also for lossless video compression, the future work includes incorporation of our method in H.264 coding and improving it.

References

1. Bartels, R.H., Beatty, J.C., Barsky, B.A.: An Introduction to Splines for use in Computer Graphics and Geometric Modeling. Morgan Kaufmann (1995)
2. Cheung, C.H., Po, L.M.: A novel cross-diamond search algorithm for fast block motion estimation. IEEE Trans. Circuits Syst. Video Technol. **12**(12), 1168–1177 (2002)
3. Fu, X., Liang, D., Wang, D.: A new video compression algorithm for very low bandwidth using curve fitting method. In: Advances in Visual Information Systems, pp. 223–229 (2007)
4. Ghanbari, M.: Standard Codecs: Image Compression to Advanced Video Coding, new edn. Institution Electrical Engineers (2003)
5. Katsaggelos, A.K., Kondi, L.P., Meier, F.W., Ostermann, J., Schuster, G.M.: Mpeg-4 and rate-distortion-based shape-coding techniques. IEEE Proc. Spec. Issue Multimed. Signal Process. **86**(6), 1126–1154 (1998)
6. Khan, M.A., Ohno, Y.: Compression of video data using parametric line and natural cubic spline block level approximation. IEICE Trans. Inf. Syst. **E90-D**(5), 844–850 (2007)
7. Koga, T., Iinuma, K., Hirano, A., Iijima, Y., Ishiguro, T.: Motion compensated interframe coding for video conferencing.In: Proceedings National Telecommunications Conference, New Orleans, LA, pp. G5.3.1–G5.3.5 (1981)
8. Lee, B.G., Lee, J.J., Yoo, J.: An efficient scattered data approximation using multilevel B-splines based on quasi-interpolants.In: 3DIM '05: Proceedings of the Fifth International Conference on 3-D Digital Imaging and Modeling (2005)
9. Libor, V., Václav, S.: Coddyac: Connectivity driven dynamic mesh compression. In: 3DTV Conference Proceedings (2007)
10. Lin, T.C., Chen, S.H., Truong, T.K.: Medical image compression using cubic spline interpolation with bit-plane compensation.In: Proceedings of the SPIE, Medical Imaging 2007: PACS and Imaging Informatics, San Diego, California, USA **6516** 65,160D (2007)
11. Sayood, K.: Introduction to Data Compression, third edn. Morgan Kaufmann (2005)
12. Soongsathitanon, S., Dlay, S.S.: A new orthogonal logarithmic search algorithm for fixed block-based motion estimation for video coding.In: Proceedings of Third International Symposium on Communication Systems Networks and Digital Signal Processing, Sheffield Hallam University Press Learning Centre, pp. 256–256 (2002)
13. Thyagarajan, K.: Digital Image Processing with Application to Digital Cinema. Focal Press (2005)
14. Truong, T.K., Wang, L.J., Reed, I.S., Hsieh, W.S.: Image data compression using cubic convolution spline interpolation. IEEE Trans. Image Process. **9**(11), 1988–1995 (2000)
15. Ung, Y.K., Mokhtarian, F.: Multi-scale spline-based contour data compression and reconstruction through curvature scale space.In: Proceedings ICASSP '00, IEEE International Conference on Acoustics, Speech, and Signal Processing **6**, vol. 4, pp. 2123–2126 (2000)
16. Wang, Y., Ostermann, J., Zhang, Y.Q.: Video Processing and Communications, first edn. Prentice Hall (2001)
17. Zaletelj, J., Pecci, R., Spaan, F., Hanjalic, A., Lagendijk, R.: Rate distortion optimal contour compression using cubic B-splines.In: Proceedings European Signal Processing Conference (1998)
18. Zaletelj, J., Tasic, J.F.: Curvature analysis approach to shape coding using B-splines.In: Proceedings of Visual Communications and Image Processing 2001, pp. 676–685 (2001)