Compression of Temporal Video Data by Catmull-Rom Spline and Quadratic Bézier Curve Fitting

Murtaza Khan  
Graduate School of Science & Technology  
Keio University, Japan  
murtaza@on.ics.keio.ac.jp  

Yoshio Ohno  
Faculty of Science & Technology  
Keio University, Japan  
ohno@on.ics.keio.ac.jp  

ABSTRACT

This paper presents a new method for lossy compression of temporal data of both naturally recorded and synthetically created videos by Catmull-Rom spline and quadratic Bézier curve fitting. The proposed method approximates the luminance or color variations in a sequence of frames by spline fitting in Euclidean space. Precise control of accuracy at pixel level is achieved by a specified tolerance of error. A break and fit criterion is employed to minimize the number of curve segments required to fit the data. Experimental results show that the described method yields very good results, both in terms of objective and subjective quality measurement, i.e., bit-rate/PSNR and human visual acceptance, without causing any blocking artifacts.

Keywords: Video data, sequence of images, approximation, compression, fitting, Catmull-Rom spline, Bézier curve.

1 INTRODUCTION

Digital video data consists of sequence of frames (images). Each frame consists of rectangular 2-D array of pixels. 3-D RGB or 1-D luminance values in a sequence of frames are associated with each pixel. RGB or luminance value(s) of a pixel can be considered as a point in Euclidean space $R^3$ or $R^1$ 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., $p_j = (R_j, G_j, B_j)$ for RGB or $p_j = l_j$ for luminance, where $1 \leq j \leq n$. Figure 1 shows RGB variation of a pixel in 80 frames of a video. Video data contains temporal and spatial correlation. In our proposed method focus is on temporal compression of video data by approximating it using quadratic Bézier curve and Catmull-Rom spline fitting.

Splines and curves 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 [BBB95]. Spline can compress the data by approximating 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 approximated data points are generated by spline interpolation of control points.

In our method, we considered temporal variations of color or luminance values of pixels in a sequence of frames as input data and approximated it with far less number of control points (output data). An important factor in spline approximation of data is finding least number of control points. We achieved this goal by selecting optimal set of control points.

Approximation and compression of data using parametric curves particularly cubic splines are explored by many authors [LCT07, UM00, IO93] et al. A method of dynamic mesh compression of animated sequences is described by [LV07]. The approach used by [LV07] is based on EdgeBreaker and Principal Component Analysis (PCA) and it exploits both spatial and temporal coherence. [LCT07] presented a medical image compression algorithm using cubic spline. Contour data compression method using Curvature Scale Space technique and Hermite curves is proposed by [UM00]. Proposed method of [IO93] uses cubic Bézier curves and is suitable for compressed representation of outline of

Figure 1: RGB temporal variation of a pixel in 80 frames of a video.
fonts. Our method uses quadratic Bézier curve and Catmull–Rom spline that are computationally more efficient than cubic splines. Cubic splines are more appropriate for image compression but less feasible for video data compression. Non-spline based temporal correlation reductions methods are based on motion estimation via translating block matching algorithms (BMAs) [CP02, SD02, KIH+81]. 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 approximated by blocks in reference frame and corresponding motion vectors [Gha03, Thy05, Say05]. Due to pixel level fitting by BMAs, we do not find matching pixel or matching block. We adopts 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 and Catmull–Rom spline fitting. BMA works at block level and may cause blocking artifacts [WOZ01, IM00]. Due to pixel level fitting by proposed method, precise control of accuracy and immunity from blocking artifacts is achieved. Due to large size of video data it is also desirable that fitting process is automated. In our proposed scheme, the user has just to initialize a few parameters, then the rest of the steps i.e., fitting, encoding/decoding are fully automated.

2 CATMULL-ROM SPLINE AND QUADRATIC BÉZIER CURVE

The following two subsections i.e., 2.1 and 2.2 describe the theory and mathematical models of Catmull–Rom spline and quadratic Bézier curve, respectively.

2.1 Catmull–Rom Splines (CRS)

Catmull–Rom spline (CRS) [CR74] is a piecewise \( C^1 \) continuous curve with specified endpoint tangents. A

\[ P_j \quad P_{j-1} \quad P_{j+2} \quad P_{j+1} \]

Catmull–Rom spline segment.

\[ P_j \quad P_{j-1} \quad P_{j+2} \quad P_{j+1} \]

Catmull–Rom spline segment.

\[ P_j \quad P_{j-1} \quad P_{j+2} \quad P_{j+1} \]

Catmull–Rom spline segment.

\[ P_j \quad P_{j-1} \quad P_{j+2} \quad P_{j+1} \]

Figure 2: The \( j^{th} \) Catmull–Rom spline segment.

\[ P_j \quad P_{j-1} \quad P_{j+2} \quad P_{j+1} \]

Figure 3: Multi-segment Catmull–Rom splines in 2-D and 3-D space.

CRS segment is defined by four control points, i.e., \( P_{j-1}, P_j, P_{j+1}, \) and \( P_{j+2} \), as shown in Figure 2. The \( j^{th} \) segment of CRS interpolates between two middle control points, i.e., \( P_j \) and \( P_{j+1} \). The end control points, i.e., \( P_{j-1} \) and \( P_{j+2} \) are used to calculate the tangents of \( P_j \) and \( P_{j+1} \). Equations for boundary conditions of \( j^{th} \) segment are written as:

\[
P_j' = \frac{1}{2} (P_{j+1} - P_{j-1}) \quad (1)
\]

\[
P_{j+1}' = \frac{1}{2} (P_{j+2} - P_j) \quad (2)
\]

For \( l \) joined segments, there are 2\( l \) conditions for continuity of functions and 2\( l \) conditions for continuity of slopes. Finally the equation of CRS for \( j^{th} \) segment is written as follows:

\[
Q_j(t_i) = \frac{1}{2} \left[ (-t_i^3 + 2t_i^2 - t_i)P_{j-1} + [3t_i^2 - 5t_i^2 + 2]P_j + [-3t_i^3 + 4t_i^2 - t_i]P_{j+1} + (-t_i^3 + t_i^2)P_{j+2}}{2} \right]
\]

where \( t_i \) is parameter of interpolation , \( 0 \leq t_i \leq 1 \). In order to generate \( n \) points between \( P_j \) and \( P_{j+1} \) inclusive, the parameter \( t_i \) is divided into \( (n - 1) \) intervals...
between 0 and 1 inclusive, and \( Q_i(t_i) \) is evaluated at \( n \) values of \( t_i \). Since the CRS passes through its middle control points, therefore \( Q_i(0) = P_i \) and \( Q_i(1) = P_{i+1} \).

Figure 3 shows multi-segment Catmull-Rom splines.

### 2.2 Quadratic Bézier Curve (QBC)

Quadratic Bézier curve (QBC) is a \( C^1 \) continuous curve. A QBC segment, is defined by three control points, i.e., \( P_0, P_1, \) and \( P_2 \), as shown in Figure 4. \( P_0 \) and \( P_2 \) are called end control points (ECP), while \( P_1 \) called a middle control point (MCP). QBC passes through its end control points, while a middle control point is used to control the shape of 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:

\[
Q(t) = (1-t)^2P_0 + 2t(1-t)P_1 + t^2P_2, \quad (4)
\]

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 the QBC passes through its first and last control points, therefore \( Q(0) = P_0 \) and \( Q(1) = P_2 \). Figure 5 shows multi-segment quadratic Bézier curves.

### 3 FITTING STRATEGY

This section describes the strategy of fitting Catmull-Rom spline and quadratic Bézier curve to video data. Fitting process is applied to temporal data of each spatial location individually. Number of spatial locations are \( W \times H \), where \( W \) and \( H \) are width and height of a frame respectively. Let color or luminance value of a spatial location \( (x_i, y_i) \), \( 1 \leq x \leq W, 1 \leq y \leq H \), 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 and Catmull-Rom spline fitting. 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 two parameters: (1) upper limit of error \( \xi^\text{lim} \), i.e., maximum allowed squared distance between original and fitted data, e.g., \( \xi^\text{lim} = 100 \), and (2) initial breakpoint interval \( \delta \), i.e., color or luminance value of pixel after every 8th frames is taken as a breakpoint (control point), e.g., \( \delta = 12 \) sets the initial breakpoints as \( BP = \{p_1, p_{13}, p_{25}, p_{37}, \ldots, p_n\} \) (color or luminance value of pixel in last frame is always taken as a breakpoint). The fitting process divides the data into segments based on breakpoints, i.e., \( S = \{S_1, S_2, \ldots, S_{n-1}\} \). A segment is a set of all points (color or luminance values) between two consecutive breakpoints, e.g., \( S_1 = \{p_1, p_2, \ldots, p_{13}\}, S_2 = \{p_{13}, p_{14}, \ldots, p_{25}\} \).

In addition to control points of the current segment, Catmull-Rom spline needs control points of previous and next segments. Therefore we used breakpoints of previous, current and next segments as control points of a current Catmull-Rom spline segment and obtained the approximated data of current segment using Eq. (3). An interesting question is how to obtain the breakpoints of first and last segments because they do not have previous and next segments respectively. Unlike [KB84] taking arbitrary breakpoints at both ends, we opted to take \( P_{j-1} = P_j \), for the first segment and \( P_{j+1} = P_{j+2} \), for the last segment. This way we are able to get the control
points of all segments automatically. For a quadratic Bézier curve breakpoints of current segment are taken as end control points (ECP), while middle control point (MCP) i.e., \( P_i \) is obtained by least square method. Least square method gives the best value of 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 \( O_t \) and \( Q(t) \) are values of original and approximated points respectively then we can write the least square equation as follows:

\[
U = \sum_{i=1}^{m} [O_t - Q(t)]^2. \tag{5}
\]

Substituting the value of \( Q(t) \) from Eq. (4) in Eq. (5) yields:

\[
U = \sum_{i=1}^{m} [p_i - (1 - t_i)^2 p_0 + 2t_i(1 - t_i)p_1 + t_i^2 p_2]^2. \tag{6}
\]

To find value of \( P_i \) differentiating Eq. (6) partially with respect to \( P_i \) yields:

\[
\frac{\partial U}{\partial P_i} = 0. \tag{7}
\]

Solving Eq. (7) for \( P_i \) gives:

\[
P_i = \frac{\sum_{i=1}^{m} [p_i - (1 - t_i)^2 p_0 + 2t_i(1 - t_i)p_1 + t_i^2 p_2]^2}{\sum_{i=1}^{m} 2t_i(1 - t_i)}. \tag{8}
\]

Once all three control points are in hand then approximated data of current Bézier segment is obtained using Eq. (4).

Same procedure is repeated for each segment for both Catmull-Rom spline and quadratic Bézier curve. This yields \( n \) values of approximated data, \( Q = \{q_1, q_2, \ldots, q_n\} \). Then we compute error of fitting, i.e., squared distance of each point between the original data and the approximated data, \( d_i^2 = |p_i - q_i|^2, 0 \leq i \leq n \). From the maximum distance \( d_{\text{max}} \) we compute maximum squared distance, \( \xi_{\text{max}} = \text{Max}(d_1^2, d_2^2, \ldots, d_n^2) \). If maximum squared distance of any \( j^{th} \) segment is greater than \( \xi_{\text{lim}} \) then this segment is split (replaced) with two new segments, \( j_1^{th} \) and \( j_2^{th} \), i.e., \( S = \{S_1 \cup \{S_j_1, S_j_2\}\} \).

A new breakpoint \( b_{\text{new}} \) from original data is added in the set of breakpoints where the error is maximum, i.e., \( BP = \{BP\} \cup \{b_{\text{new}}\} \). For example, if segment \( S_1 \) splits at \( p_6 \) then a new breakpoint \( b_{\text{new}} \) is inserted between breakpoints \( p_1 \) and \( p_{13} \) \( BP = \{p_1, \overbrace{p_6, p_{13}, p_{25, 37, \ldots, p_n}}^{b_{\text{new}}}, p_{13}\} \) and two new segments \( \{p_1, p_2, \ldots, p_6\} \) and \( \{p_6, p_7, \ldots, p_{13}\} \) replace \( S_1 \). The fitting process is repeated with new set of breakpoints until mean square error of each segment is equal to or less than \( \xi_{\text{lim}} \).

The above described fitting process is applied to color or luminance variations in temporal dimension of each spatial location separately. It is obvious that each spatial locations have different number of breakpoints (control points) selected from different frames, in other words rectangular shape of video frames is no longer intact.

Section 4 describes the proposed algorithm formally for a single spatial location. In order to have a better comparative look of fitting process between Catmull-Rom spline (CRS) and quadratic Bézier curve (QBC), we marked steps in curly braces where CRS and QBC differs in fitting. Figure 6 and Figure 7 show fitting of luminance values of in 80 frames of a video by Catmull-Rom spline and quadratic Bézier curve, respectively.

4 ALGORITHM

procedure Fitting algorithm for luminance variations of single spatial location.

spatial location \((x, y)\), \(1 \leq x \leq W, 1 \leq y \leq H\).

Luminance of \((x, y)\) in frame \(1\) to \(n\) is \(\{p_1, p_2, \ldots, p_n\}\).

Require:

Max. allowed squared distance i.e., \(\xi_{\text{lim}}\),

Breakpoints interval i.e., \(\delta\),

Points of original data: \(O = \{p_1, p_2, \ldots, p_n\}\),

Breakpoints: \(BP = \{p_1, p_{\delta}, \ldots, p_{n+\delta}, \ldots, p_n\}\).

1: Get indices of \(BP\): \(V = \{1, 1+\delta, 1+2\delta, \ldots, n\} = \{v_1, v_2, \ldots, v_n\}\).

2: Divide \(O\) into segments \(S = \{S_1, S_2, \ldots, S_n\}\).

3: Not required \(\quad\quad\quad\text{CRS}\)

4: Find MCP of each segment \(\text{QBC}\)

5: \(\quad\text{QBC}\) \(\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\text{Not required}\)

6: \(\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\text{CRS}\)

7: \(\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\text{QBC}\)

while \(\xi_{\text{max}} > \xi_{\text{lim}}\) do

8: \(bp_{\text{new}} = b_k\)

9: \(V = V \cup \{k\}\)

10: Split \(j^{th}\) segment into \(j_1^{th}\) and \(j_2^{th}\) segments

11: \(BP = \{BP\} \cup \{bp_{\text{new}}\}\)

12: \(S = \{S \cup \{j_1, j_2\}\} \cup \{S_{j_1}, S_{j_2}\}\)

13: \(\quad\text{QBC}\)

14: Using updated \(BP\) fit spline/curve to:

15: \(\quad\text{QBC}\)

16: \(\quad\text{QBC}\)
Update $\xi_{\text{max}} = \begin{cases} \max \left( \xi_{\text{max}}, d_{j-1}^2, d_{j-1}^2, d_{j-1}^2, d_{j+1}^2 \right) & \text{CRS} \\ \max \left( \xi_{\text{max}}, d_{j-1}^2, d_{j-1}^2 \right) & \text{QBC} \end{cases}$

17: Find count of interpolating points (C) from $V = \{ v_1, v_2, \ldots, v_n \}, u \ll n$:

$C = \{ c_1, c_2, \ldots, c_{u-1} \}, c_i = v_{i+1} - v_i + 1$

18: Encode BP and C CRS

18: Encode BP, MCP and C QBC

---

Figure 6: Catmull-Rom spline fitting to luminance values in 80 frames of a video, $\xi_{\text{int}} = 21, \delta = 79$, PSNR=43.845-dB.

Figure 7: Quadratic Bézier curve fitting to luminance values in 80 frames of a video, $\xi_{\text{int}} = 21, \delta = 79$, PSNR=45.115-dB.

### 5 EXPERIMENTS AND RESULTS

We have applied the proposed algorithm on various naturally recorded and synthetically created video sequences. Output data is entropy encoded. The performance of proposed method is evaluated in terms of bit-rate, measured in bits per pixel (bpp) and PSNR, measured in decibel (dB). Table 1 gives the details of selected input video sequences whose results are presented in this paper. Salesman and Foreman are naturally recorded luminance videos at 8-bpp, while Darius is a synthetically created RGB video at 24-bpp. Table 2 compares the performance of CRS and QBC with Three Step Search method (TSS) [KIH + 81], a well known and widely used block matching algorithm for temporal compression of video data. For TSS every alternative frame is predicted from previous frame (reference frame), macro block size is $8 \times 8$, and range of search window is $\pm 8$ in both horizontal and vertical directions. Reference frames are first differentially then entropy coded along with motion-vectors (MVs). Predicted frames are not coded because they are approximated from reference frames. The fitting methods achieved better compression performance than TSS, because the fitting work of our algorithm is at pixel level and they approximate the variations of luminance with good level of accuracy. We did not compare the performance for RGB video, because block matching algorithm convert the RGB data to $Y C_b C_r$ data, and block matching is performed for only luminance data ($Y$), while chrominance data ($C_b C_r$) is sub-sampled (4:2:0), and predicted chrominance frames are obtained from MVs of luminance data. Therefore comparison of CRS and QBC with TSS for RGB video is not appropriate.

Figures 8, 9 and 10 show R-D curves of CRS and QBC for Salesman, Foreman and Darius sequences respectively. Note that these statistics are only for temporal compression of video data; neither spatial compression nor quantization is used. The statistics show that the proposed method yields good objective performance for naturally recorded videos (Salesman and Foreman) and very good performance for synthetically created video (Darius). For the Salesman sequence, around 35-dB PSNR is achieved at bit-rate 0.59-bpp and 0.64-bpp by CRS and QBC respectively. For the Foreman sequence, around 35-dB PSNR is achieved at bit-rate 1.98-bpp and 2.1-bpp by CRS and QBC respectively. Darius is a color sequence and bit-rate of original Darius sequence is 24-bpp. We achieved around 35-bpp PSNR at bit-rate as low as 1.5-bpp by both CRS and QBC.

Now look at the comparative performance of CRS and QBC. For the Salesman sequences at low bit-rate CRS performs better but at bit-rate 0.82-bpp QBC leads CRS. For the Foreman sequences CRS performs better than QBC at all values of bit-rate. For the Darius sequence the curve is quite interesting, at bit-rate 1.6-bpp CRS performs better, 1.6-bpp-bit-rate<3.5-bpp QBC performs better, and bit-rate<3.5-bpp, again CRS takes leads over QBC.

Figures 11, 12 and 13 show 30-th frames of CRS and QBC encoded videos for Salesman, Foreman and Darius video sequences respectively. It can be seen from these figures that subjective performance i.e., human vi-
suual acceptance of CRS and QBC approximated video frames is very good and no blocking artifacts are produced.

6 DISCUSSION

Split of a segment: Lines 8 to 16 are important steps of our fitting algorithm. For the correct and efficient implementation of our algorithm it is very important to find which other segments are affected by splitting of a segment. Let us assume that an arbitrary segment $j$ splits into two segments $j_1$ and $j_2$. For the quadratic Bezier curve we have to recompute middle control points, approximating points and squared distances of $j_1$ and $j_2$ segments only. But for Catmull-Rom spline the situation is not so simple. For three consecutive segments $j-1$, $j$ and $j+1$, the last three control points of $(j-1)^{th}$ segment i.e., $P_{j-2}$, $P_j$ and $P_{j+1}$ are same to first three control point of $(j)^{th}$ segment and first control point of $(j-1)^{th}$ segment i.e $P_{j-2}$ is not shared with $(j)^{th}$ segment. Similarly the first three control points of $(j+1)^{th}$ segment i.e., $P_j$, $P_{j+1}$ and $P_{j+2}$ are same to last three control points of $(j)^{th}$ segment and last control point of $(j+1)^{th}$ segment i.e $P_{j+3}$ is not shared with $(j)^{th}$ segment. This means that a segment shares control points with its previous and next segments. Consequently adding a new breakpoint (splitting) at $(j)^{th}$ segment requires to recompute approximating values and squared distances for two new segments, i.e., $j_1$ and $j_2$, obtained from splitting of $(j)^{th}$ segment, in addition to that approximating values and squared distances of previous and next segments of $(j)^{th}$ segments, i.e., $(j-1)^{th}$ and $(j+1)^{th}$ segments are also need to be recomputed. Fortunately Catmull-Rom spline saves some computation by not requiring least square solution as required by quadratic Bezier curve to find the value of its middle control point. Lines 3 and 13 of the fitting algorithm describe these recomputation steps for quadratic Bezier curve and Catmull-Rom spline.

Rate Control: Rate of the output data is controlled by varying the value of $\xi^{imt}$. Increasing the value of $\xi^{imt}$ decreases the bit-rate. The default value of $\xi^{imt}$ is 100.

<table>
<thead>
<tr>
<th>Video Name</th>
<th>Format</th>
<th>Number of Frames</th>
<th>Bit-rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Salesman</td>
<td>CIF</td>
<td>45</td>
<td>8-bpp</td>
</tr>
<tr>
<td>Foreman</td>
<td>SIF</td>
<td>44</td>
<td>8-bpp</td>
</tr>
<tr>
<td>Darius</td>
<td>SIF</td>
<td>44</td>
<td>24-bpp</td>
</tr>
</tbody>
</table>

Table 1: Details of input video sequences.

<table>
<thead>
<tr>
<th>Method</th>
<th>Salesman</th>
<th>Foreman</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>PSNR</td>
<td>Bit-rate</td>
</tr>
<tr>
<td>TSS</td>
<td>38.132</td>
<td>1.7768</td>
</tr>
<tr>
<td>CRS</td>
<td>38.244</td>
<td>0.8574</td>
</tr>
<tr>
<td>QBC</td>
<td>38.291</td>
<td>0.8578</td>
</tr>
</tbody>
</table>

Table 2: Performance comparison of TSS [KIHt 81], CRS and QBC.
**Output data encoding requirement:** For CRS we need to encode (1) breakpoints (BP) and (2) count of interpolating points (C). For QBC, in addition to (1) and (2), we also need to encode *middle control points* (MCP). If video data is fitted using \( m \) segments then for CRS, \( m + 1 \) values of BP and \( m \) values of C are need to be encoded. For QBC with equal number of segments, \( m + 1 \) values of BP, \( m + 1 \) values of MCP and \( m \) values of C are need to be encoded. Apparently CRS has less output data to be encoded. But it also depends on how much splitting of segments occurs. Usually splitting of a segment by CRS fitting is more often than splitting of a segment by QBC fitting.

**Reasons to choose quadratic Bézier curve and Catmull-Rom spline:** (1) Both QBC and CRS are computationally efficient than other curves e.g., Natural cubic spline, B-spline, cubic Bézier curve etc. In fact we also used Natural cubic spline and cubic Bézier curve and found them less feasible than QBC and CRS. (2) Breaking of a segment into two segments keeps the computation cost within acceptable limit for both QBC and CRS.

**Computational cost:** QBC is computationally more efficient than CRS, because, (1) QBC is a quadratic function, while CRS is cubic a function. (2) Due to the least square fitting, QBC approximates longer segments, which means it causes lesser splitting and needs lesser computation than CRS.

**Naturally recorded vs synthetically created videos:** The latter has less noise and more uniform distribution of luminance/color. Therefore, both CRS and QBC perform extremely well for synthetically created videos.

**Choosing between CRS and QBC:** CRS and QBC behave quite similar, but if higher compression is desired then CRS is slightly better than QBC. If computational efficiency is of more importance then QBC is more appropriate choice.

**Limitations:** The performance of fitting process degrades for spatial locations where changes in luminance in temporal dimension is very sharp. Because such spa-
tial locations cause lot of splitting of segments. Due to non-rectangle shape of output data, combining spatial compression with temporal compression needs special care and shape-adaptive wavelet transform can be used for spatial compression.

7 CONCLUSION

We presented a new method for compression of temporal video data by Catmull-Rom spline and quadratic Bézier curve fitting. Detail of fitting strategy and pseudo code of the algorithm are presented. We tested the proposed method using luminance and color (RGB) temporal data of naturally recorded and synthetically created video sequences. Experimental results show that the proposed method yields very good results both in terms of objective and subjective quality measurement parameters, i.e. bit-rate/PSNR and human visual acceptance, without causing any blocking artifacts. The method is suitable for compression of both naturally recorded and synthetically created videos such as animations and cartoons.

8 FUTURE WORK

Compression of spatial video data by spline/curve fitting is under investigation.

REFERENCES


