Point Lattices in Computer Graphics and Visualization
how signal processing may help computer graphics

Dimitri Van De Ville
Ecole Polytechnique Fédérale de Lausanne
Biomedical Imaging Group
dimitri.vandeville@epfl.ch

IEEE Visualization 2005 - Tutorial (Image Processing 2D)

Overview

B-splines: the right tool for interpolation
- fundamental properties
- spline fitting
  - interpolation; smoothing; least-squares
  - quantitative approximation quality

A primer to the wavelet transform
- multi-resolution, semi-orthogonal wavelets

Multi-dimensional extensions: hex-splines
- hexagonal versus Cartesian lattice

Notations

- one-sided
  \( (x)_+ = \max(0, x) \)

- Fourier transform
  \( \hat{f}(\omega) = \int f(x)e^{-j\omega x}dx \)

- Z-transform
  \( C(z) = \sum_{k \in \mathbb{Z}} c[k]z^{-k} \)

Polynomial B-splines

- B-spline of degree \( n \)
  \[ \beta_n^\alpha(x) = \underbrace{\beta_n^0 * \beta_n^1 * \cdots * \beta_n^0}_{(n+1) \text{ times}}(x) \]

- Symmetric B-spline
  \[ \beta_n^0(x) = \beta_n^0 \left( x + \frac{n+1}{2} \right) \]

- Key properties
  - compact support
  - piecewise polynomial
  - positivity
  - smoothness (continuity)

[Schoenberg, 1946]
The link between continuous and discrete

\[ \mathbf{\beta}(x) = \sum_{k \in \mathbb{Z}} \beta^0_+(x - k) \]

analog signal in the continuous domain

B-spline coefficients in the discrete domain

“Cardinal” setting = unit spacing and \( \infty \) points

Fundamental B-spline properties

- Partition of unity
  - reproduction of the constant
- Riesz basis
  - stability: small perturbation of coefficients results into small change of spline signal
  - unambiguity: each representation is unique
- \( m \)-scale relation

\[ \beta^m_n(x/m) = \sum_{k \in \mathbb{Z}} h^m_n[k] \beta^p(x - k) \quad \text{with} \quad H^m_n(z) = \frac{1}{m^n} \left( \sum_{k=0}^{m-1} z^{-k} \right)^{n+1} \]

B-spline representation

B-spline Fourier expression

\[ \beta^m_n(x) = \beta^0_n \ast \cdots \ast \beta^0_n(x) \]

Fourier transform of basic element:

\[ \beta^m_n(\omega) \leftrightarrow \tilde{\beta}^0_+(\omega) = \frac{\sin(\omega/2)}{\omega/2} e^{-j\omega/2} = \frac{1 - e^{-j\omega}}{j\omega} \]

\[ \tilde{\beta}^m_n(\omega) = \left( \frac{1 - e^{-j\omega}}{j\omega} \right)^{n+1} \]

“poor man’s derivative” (finite difference)

\[ \Delta f = f(x) - f(x - 1) \leftrightarrow (1 - e^{-j\omega}) \tilde{f} \]

exact derivative

\[ D f \leftrightarrow (j\omega) \tilde{f} \]

Link between discrete and exact derivatives

\[ D^m s = D^m \{ c * \beta^m_+ \} \]

discrete filtering

spline degree reduction
Generalized fractional B-splines

Definition in the Fourier domain
\[ \hat{\beta}_\tau^\alpha(\omega) = \left( \frac{1 - e^{j\omega}}{-j\omega} \right)^{\frac{\alpha+1}{2}} \left( \frac{1 - e^{-j\omega}}{j\omega} \right)^{\frac{\alpha+1}{2} + \tau} \]

Degree \( \alpha \in \mathbb{R}_+ \)
Shift \( \tau \in \mathbb{R} \)

Spline fitting

How to find the spline coefficients?
\[ s(x) = \sum_{k \in \mathbb{Z}} \beta^+_n(x - k) \]

Spline fitting: (1) spline interpolation

- Spline interpolation (exact, reversible)
  - Discrete input \( f[k] \)
  - Filtering \( c[k] \)
  - Such that \( s(x)|_{x=k} = f[k] \)

- Smoothing spline
- Least square splines (approximation between spline spaces)

Spline interpolation

- Discrete B-spline kernels
  \[ b^n_l[k] = \beta^n(x)|_{x=k} \quad \overset{z}{\longrightarrow} \quad B^n_l(z) = \sum_{k=-[n/2]}^{[n/2]} \beta^n(k)z^{-k} \]

- Satisfying interpolation condition: inverse filter!
  \[ f[k] = \sum_{l \in \mathbb{Z}} c[l] \beta^n(x - l)|_{x=k} = (b^n_l * c)[k] \Rightarrow c[k] = (b^n_l)^{-1} * f[k] \]

- Efficient recursive implementation:
  - Cascade of causal and anti-causal filters
  - E.g., cubic spline interpolation
    \[ (b^n_l)^{-1}[k] \overset{z}{\longrightarrow} \frac{6}{z + 4 + z^{-1}} = (1 - \alpha)^2 \]
    - Causal
    - Anti-causal
Spline interpolation

- Generic C-code
  - main recursion
    ```c
    void ConvertToInterpolationCoefficients ( double *c[, ], long nDataLength, double *z[, ], long nbInterp, double tolerance)
    {
        double lambda = 1.0; long n, k;
        if (nDataLength == 1L) return;
        for (k = 0L; k < nbInterp; k++) {
            lambda = fmax( (z[k] - k) * (1.0 - lambda) );
            for (n = nL, n < nDataLength; n++) c[n] = lambda * (fmax( 0.0, (n - k) * (1.0 - lambda) ));
        }
    }
    ```
  - initialization
    ```c
    double InitialChaosCoefficient ( double *c[, ], long nDataLength, double z[, ], double tolerance)
    {
        double Sum = 1.0; long n, k;
        double GetTen5Length (long GetTen5Length) GetTen5Length; (nL, n < nDataLength; n++) Sum * GetTen5Length; n *= k;
        return(Sum);
    }
    ```

Spline interpolation

- The fundamental spline converges to sinc as the degree goes to infinity
  - \[
  \lim_{n \to \infty} \varphi_n^{\text{int}}(x) = \sin(x)
  \]
  - \[
  \lim_{n \to \infty} \left( \frac{\sin(\omega/2)}{\omega/2} \right)^{n+1} \frac{1}{B_n^0(x, 2\pi)} = \text{rect} \left( \frac{\omega}{2\pi} \right)
  \]
- Shannon’s theory appears as a particular case

Spline fitting: (2) smoothing spline

- Smoothing spline
  - discrete and noisy input
  - filtering
  - subject to regularization

- Least square splines (approximation between spline spaces)
Smoothing spline

- The solution (among all functions) of the smoothing spline problem
  \[ \min_{s(x)} \left\{ \sum_{k \in \mathbb{Z}} |f[k] - s(k)|^2 + \lambda \int_{-\infty}^{+\infty} |D^m s(x)|^2 \, dx \right\} \]
  is a cardinal spline of degree \(2m-1\). Its coefficients can be obtained by suitable digital filtering of the input samples:
  \[ c[k] = h_\lambda \ast f[k] \]

- Related to: MMSE (Wiener filtering); splines form optimal space!!!

Special case: the draftman’s spline

Minimum curvature interpolant is obtained for \( m = 2, \lambda \to 0 \)
= cubic spline!

\[ [Ursen\&Miu2005] \]

Spline fitting: (3) least-square spline

- Least-square spline (approximation between spline spaces)
  \[ s_1(x) \to \text{resampling \\ \\ & filtering} \to s_2(x) \]
  such that \( \min_{s_2} ||s_1 - s_2||_{L_2} \)

Least-square spline

- Minimize quadratic error between splines
  \[ \{c_\alpha[k]\} = \arg \min_{\{c_\alpha[k]\}} ||s_1 - s_\alpha||_{L_2} \quad \text{with} \quad s_1(x) = \sum_{k \in \mathbb{Z}} c_1[k] \beta^n(x-k) \\
  s_\alpha(x) = \sum_{k \in \mathbb{Z}} c_\alpha[k] \beta^n(x/\kappa - k) \]
  - determine \( c_1[k] \); e.g., by spline interpolation \((b_1^n)^{-1}\)
  - resample using
  \[ d_\alpha[k] = \sum_{l \in \mathbb{Z}} c_1[l] \xi^n_\kappa (k \kappa - l) \quad \text{with} \quad \xi^n_\kappa(x) = \frac{1}{\kappa} (\beta^n(\cdot) * \beta^n(\cdot/\kappa))(x) \]
  - obtain samples of new spline representation
  \[ s_\alpha[k] = (d_\alpha \ast (b_1^{2n+1})^{-1})[k] \]

Least-square spline

- Special case: “surface projection”
  - first-order B-splines on source and target grid
  - weight of sample = overlap between B-splines’ support

\[ [\text{Unser et al. 1995}] \]
Quantitative approximation quality

- Best approximation in a space?
  \[
  \text{analog input } f(x) \quad \text{sampling & filtering} \quad c[k] = \left( \begin{array}{c} f(0) \\ f(1) \end{array} \right) \\
  s(x) = \sum_{k \in \mathbb{Z}} c[k] \varphi(x/a - k)
  \]

- Orthogonal projection
  \[
  \min_{s \in V_a} \| f - s \|_{L^2}^2 = \frac{1}{2\pi} \int_{-\infty}^{\infty} \left| \tilde{f}(\omega) \right|^2 E(\omega) d\omega
  \]
  with error kernel
  \[
  E(\omega) = 1 - \frac{\left| \tilde{\varphi}(\omega) \right|^2}{\sum_{n \in \mathbb{Z}} |\tilde{\varphi}(\omega + 2\pi n)|^2}
  \]
  [Bluherer,1999]

B-spline interpolation in 2D

- 2D separable model
  \[
  f(x, y) = \sum_{k=k_1}^{k_1+n+1} \sum_{l=l_1}^{l_1+n+1} c[k, l] \beta^n(x - l) \beta^n(y - l)
  \]

- Geometric transformations

- Applications
  - zooming, rotation, resizing, warping

High-quality image interpolation

- 2D separable model
  \[
  f(x, y) = \sum_{k=k_1}^{k_1+n+1} \sum_{l=l_1}^{l_1+n+1} c[k, l] \beta^n(x - l) \beta^n(y - l)
  \]

- Geometric transformations

- Applications
  - zooming, rotation, resizing, warping

[Thévenaz et al., 2000]
Interpolation benchmark

- Cumulative interpolation experiment: the best algorithm wins...

- bilinear
- windowed sinc
- cubic spline

High-quality isosurface rendering

- 3D B-spline representation of volume data
- Isosurface
  - analytical knowledge of normal vectors

Wavelets

- Admissible scaling function (“father wavelet”)
  - Riesz basis conditions
  - partition of unity
  - two-scale relation
- B-splines are perfect candidates
- Then there exists a wavelet \( \psi(x/2) = \sum_{k \in \mathbb{Z}} g[k] \varphi(x - k) \)
  such that \( \left\{ 2^{-n/2} \psi \left( \frac{x - 2^k}{2^n} \right) \right\}_{k \in \mathbb{Z}, n \in \mathbb{Z}} \) forms a Riesz basis of \( L^2 \)

Multi-resolution approximation

- \( m \)-scale relation
  \[ \beta^m(x/m) = \sum_{k \in \mathbb{Z}} h_m[k] \beta^n(x - k) \quad \text{with} \quad H^n_m(z) = \frac{1}{m^n} \left( \sum_{k=0}^{m-1} z^{-k} \right)^{n+1} \]

- Pyramid or tree algorithms \( (m = 2^i) \)
  - fast evaluation of \( f(\cdot) * \beta^m(\cdot/2^i) \)

  \[ H^n_m(z) \rightarrow 2 \rightarrow \text{binomial filter} \]
  - for high \( n \sim \) Gaussian filter

[Thévenaz et al., 2000]

[Mallat-Meyer, 1989]
Haar wavelet transform revisited

- Signal representation
  \[ s_0(x) = \sum_k c_k \varphi(x - k) \]
  basis function:
  \[ \varphi(x) \]

- Multi-scale signal representation
  \[ s_i(x) = \sum c_{i,k} \varphi_{i,k}(x) \]
  multi-scale basis function:
  \[ \varphi_{i,k}(x) = \varphi\left(\frac{x - 2^i k}{2^i}\right) \]

Semi-orthogonal wavelets

- Scaling and wavelet spaces
  \[ \mathcal{V}_i = \text{span}_{n \in \mathbb{Z}} \{ \varphi\left(\frac{x}{2^i} - n\right) \} \]
  \[ \mathcal{W}_i = \text{span}_{n \in \mathbb{Z}} \{ \psi\left(\frac{x}{2^i} - n\right) \} \]

- Semi-orthogonality conditions
  1. \[ \mathcal{W}_i \subset \mathcal{V}_{i-1} \]
  2. \[ \mathcal{W}_i \perp \mathcal{V}_i \]

Wavelets act as differentiators

Effect on transient features:
1) locality
2) sparsity (vanishing moments)
**Wavelets and differentiation**

- Fundamental property: multiscale differentiator
  \[ \hat{\psi}(\omega) \propto |\omega|^\gamma \quad \text{when} \ \omega \to 0 \]
- Responsible for
  - vanishing moments
  - decorrelation
- Very successful for coding applications
  - JPEG2000

**Hexagonal lattices**

- Voronoi cell = “best” tessellation:
  - Six equivalent neighbours
  - Twelve-fold symmetry
  - High isotropy
- Lattice matrix: \( R = [r_1 \ r_2] \)

**Hex-splines**

- Basis functions for hexagonal grids

**First order**

- Basis functions for hexagonal grids

**Second order**
Hex-splines

- Basis functions for hexagonal grids

Third order

- B-spline-like construction algorithm:
  - generating functions
  - localization operators
- B-spline-like properties:
  - convolution property (by construction)
  - positivity, partition of unity, compact support
  - convergence to Gaussian
- But...
  - no two-scale relation

Hex-splines versus B-splines

- Keep sampling density equal: $\det(R) = \Omega$
Hex-splines versus B-splines

- Extra samples so approximation quality B-splines equals that one of hex-splines

![Plot showing sampling gain vs. surface area comparison between hex-splines and B-splines.](image)

© Annette Unser

Conclusions

- B-splines are a great tool for interpolation and approximation
  - short support; analytical expression; tunable degree
  - fundamentally linked to differential operators
- Shift-invariant spaces due to uniform sampling brings along
  - fast algorithms (filtering, FFT-based, etc.)
  - powerful theoretical results (error kernel)
- Multi-resolution
  - m-scale relation for pyramids and wavelets
- Multi-dimensional extensions and variations
  - tensor-product, hex-splines, box-splines (see later)

And finally

- Many thanks go to
  - Michael Unser
  - Thierry Blu
  - Philippe Thévenaz

- Papers, demonstrations, source code:
  - [http://bigwww.epfl.ch/](http://bigwww.epfl.ch/)
- The Wavelet Digest: