Lecture 9: Image Interpolation

Contents

M I A
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30

Motivation

What is Interpolation?

- recovery of continuous data from discrete data within the range given by the discrete data
- inverse step to sampling, where discrete data are created from continuous ones (see Lectures 1 and 3)
- involves assumptions on the data, e.g. smoothness, band limitation
- distinguish from
  - *extrapolation:* uses a model *outside* the given range;
    example: weather forecast
  - *approximation:* the model does not reproduce the given discrete data exactly;
    example: regression curve through noisy data
- important topic that is not treated adequately in most text books

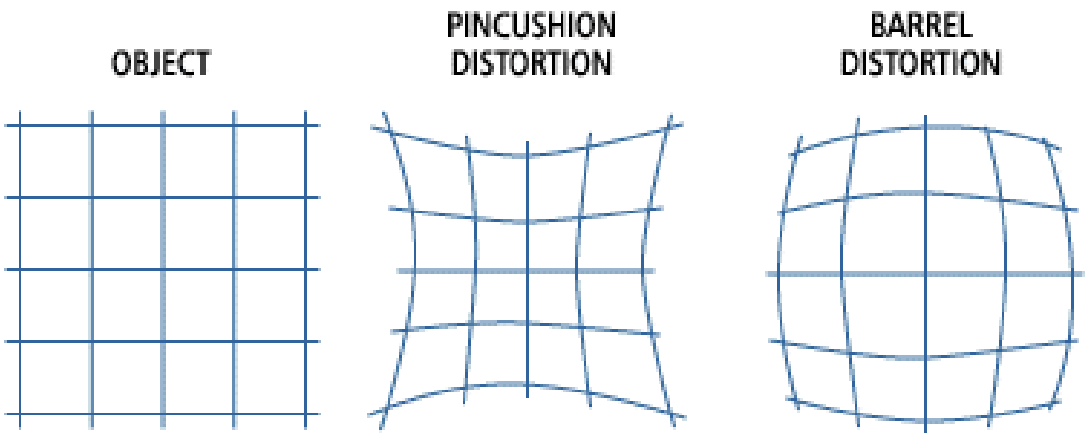
Motivation (2)

Where is Interpolation Necessary ?

- rescaling, zooming: modify the sampling rate of pixels (e.g. in so-called digital zooms in digital cameras)
- reslicing: display a (medical) 3-D data set along planes that do not coincide with planes along the axis directions
- computer graphics: volume rendering
- all kinds of geometric transformations, e.g.
  - image translation with subpixel precision
  - image rotation
  - warping in order to compensate for motion in image sequences
  - registration of medical images
  - compensation of pincushion and barrel distortions (kissen- und tonnenförmige Verzeichnungen) in lenses
  - ultrasound scan conversion from polar to cartesian coordinates

Motivation (3)

(a) Top: Original image. **(b) Bottom left:** Opical zoom with a factor 10. **(c) Bottom right:** "Digital zoom". Source: `http://www.cambridgeincolour.com/tutorials/image-interpolation.htm`.



Many optical lens systems – in particular so-called "super zooms" – suffer from pincushion and/or barrel distortions. Compensating these distortions in digital images requires interpolation methods. Source: `http://www.mellesgriot.com/products/optics/images/fig1_20.gif`.

Basic Structure of Classical Interpolation

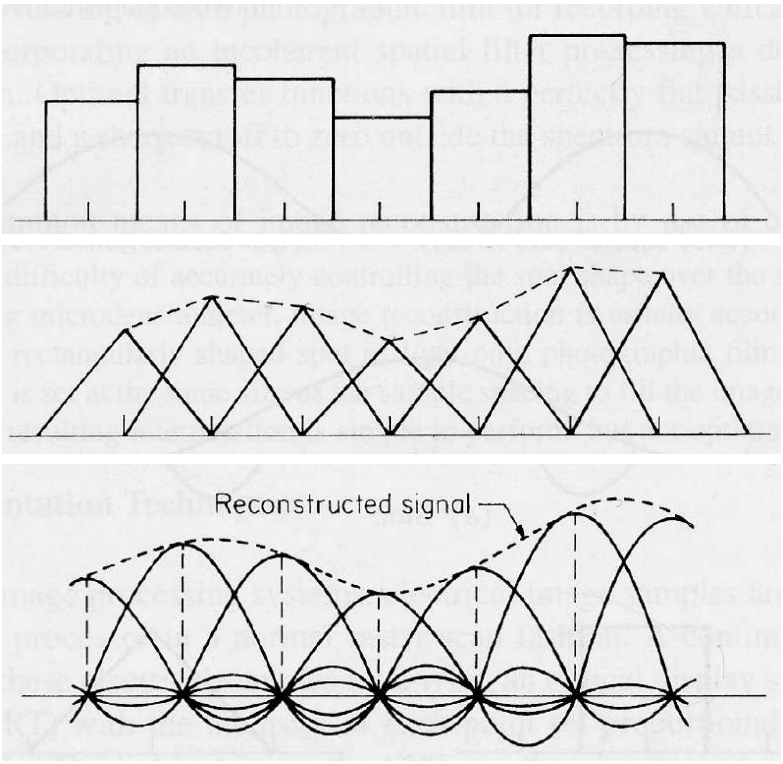
Classical Interpolation

- consider an infinitely extended (e.g. by reflection) m-dimensional discrete signal $f_{\mathbf{k}}$ with $\mathbf{k} = (k_1, k_2, ..., k_m)^\top \in \mathbb{Z}^m$
- recover a continuous signal $f(\mathbf{x})$ as a weighted average of the discrete samples $f_{\mathbf{k}}$, $\mathbf{k} \in \mathbb{Z}^m$:

$$f(\mathbf{x}) = \sum_{\mathbf{k} \in \mathbb{Z}^m} f_{\mathbf{k}} \, \varphi_{\text{int}}(\mathbf{x} - \mathbf{k})$$

  for all $\mathbf{x} = (x_1, x_2, ..., x_m)^\top \in \mathbb{R}^m$.
- The only remaining freedom lies in the *synthesis function* φ_{int} that determines the weight.

Basic Structure of Classical Interpolation (2)



(a) Top: Nearest neighbour interpolation of a signal uses pulse functions as synthesis functions. **(b) Middle:** Linear interpolation uses hat functions. **(c) Bottom:** Interpolation with sinc functions. Author: W. K. Pratt (2001).

Basic Structure of Classical Interpolation (3)

A Necessary Condition for Every Synthesis Function

- **Interpolation Condition**

  Requiring $f(\mathbf{k}) = f_{\mathbf{k}}$ for all $\mathbf{k} \in \mathbb{Z}^m$ implies that φ_{int} must satisfy

$$\varphi_{\mathrm{int}}(\mathbf{k}) = \begin{cases} 1 & \text{for } \mathbf{k} = (0,...,0)^\top, \\ 0 & \text{for } \mathbf{k} \in \mathbb{Z}^m \setminus \{(0,...,0)^\top\}. \end{cases}$$

  This fixes φ_{int} at integer arguments.
  For non-integer arguments, $\varphi_{\mathrm{int}}(\mathbf{x})$ can be tuned to specific applications.

- Synthesis functions that satisfy this condition are called *interpolating synthesis functions*.

Basic Structure of Classical Interpolation (4)

Desirable Properties of the Synthesis Function

- **Separability**

$$\varphi_{\text{int}}(\mathbf{x}) = \prod_{i=1}^{m} \tilde{\varphi}_{\text{int}}(x_i) \qquad \text{for all } \mathbf{x} = (x_1, x_2, ..., x_m)^\top \in \mathbb{R}^m$$

allows simple and efficient m-D implementations using 1-D interpolations

- **Symmetry**

$$\varphi_{\text{int}}(\mathbf{x}) = \varphi_{\text{int}}(-\mathbf{x}) \qquad \text{for all } \mathbf{x} \in \mathbb{R}^m$$

equal treatment of opposite sides;
weaker requirement than rotation invariance

- **Partition of Unity**

$$1 = \sum_{\mathbf{k} \in \mathbb{Z}^m} \varphi_{\text{int}}(\mathbf{x} - \mathbf{k}) \qquad \text{for all } \mathbf{x} \in \mathbb{R}^m$$

guarantees that a constant signal remains unaltered

Basic Structure of Classical Interpolation (5)

How Can One Judge the Quality of a Good Interpolant?

- **Small Support Region of the Synthesis Function**
  - size of the region where φ_{int} does not vanish *(support region)* should be small
  - allows fast computation
- **High Approximation Order**
  - Let the grid size h go to 0. If the (L^2) error between the interpolant and the initial continuous function (before sampling) behaves like $\mathcal{O}(h^L)$, then L is called *approximation order*.
  - large L indicates good approximation qualities
  - equivalent to the exact reproduction of all polynomials of degree $\leq L - 1$
- **High Regularity**
  - The interpolant should be as smooth as possible, i.e. it should belong to the set C^k of k-times continuously differentiable functions with a large k.
  - allows to compute high derivatives for feature detection

M I A
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30

Synthesis Functions for Classical Interpolation (in 1-D)

- **Nearest Neighbour Interpolation:**

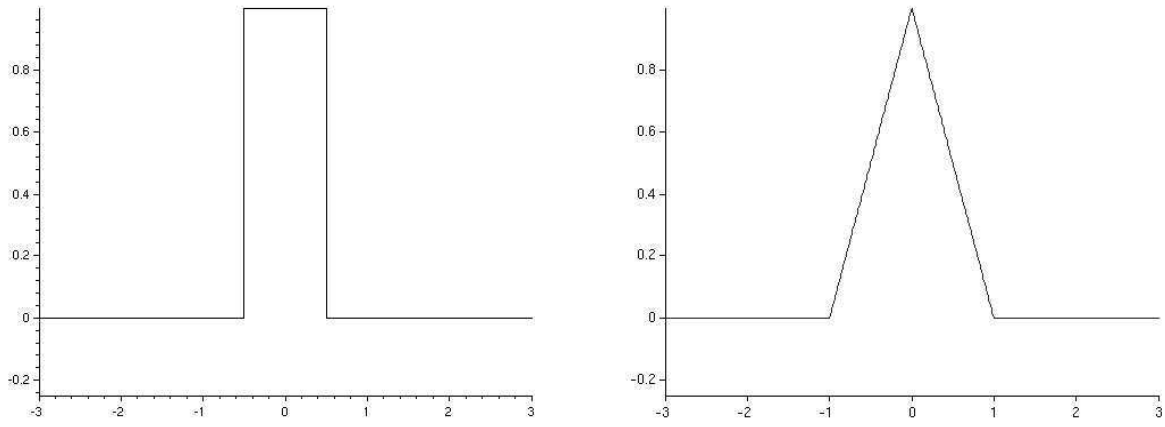
$$\varphi_{\text{int}}(x) = \begin{cases} 1 & \text{for } |x| < \frac{1}{2}, \\ \frac{1}{2} & \text{for } |x| = \frac{1}{2}, \\ 0 & \text{else} \end{cases}$$

  - very small support interval: $[-\frac{1}{2}, \frac{1}{2}]$
  - lowest approximation order: $L = 1$ (reproduces constant functions)
  - leads to a discontinuous interpolant

- **Linear Interpolation:**

$$\varphi_{\text{int}}(x) = \begin{cases} 1 - |x| & \text{for } |x| < 1, \\ 0 & \text{else} \end{cases}$$

  - small support interval: $[-1, 1]$
  - approximation order $L = 2$ (reproduces linear functions)
  - gives a continuous (C^0) interpolant



(a) Left: Synthesis function for nearest neighbour interpolation. **(b) Right:** Synthesis function for linear interpolation. Author: N. Khan (2005).

Synthesis Functions for Classical Interpolation (3)

- **Keys Interpolation (with $a = -\frac{1}{2}$):**

$$\varphi_{\text{int}}(x) = \begin{cases} \frac{3}{2}|x|^3 - \frac{5}{2}x^2 + 1 & \text{for } |x| < 1, \\ -\frac{1}{2}|x|^3 + \frac{5}{2}x^2 - 4|x| + 2 & \text{for } 1 \leq |x| < 2, \\ 0 & \text{else.} \end{cases}$$

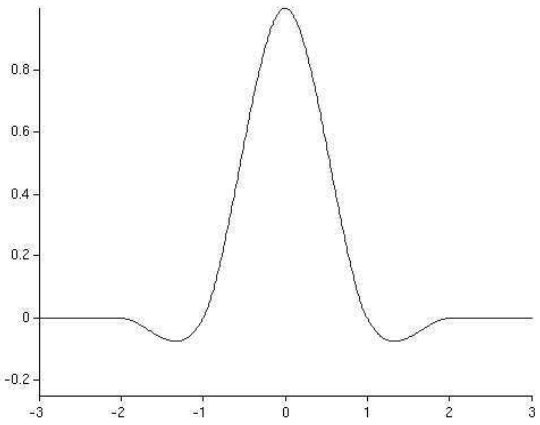
  - support interval $[-2, 2]$
  - approximation order $L = 3$ (reproduces quadratic functions)
  - creates a continuously differentiable (C^1) interpolant

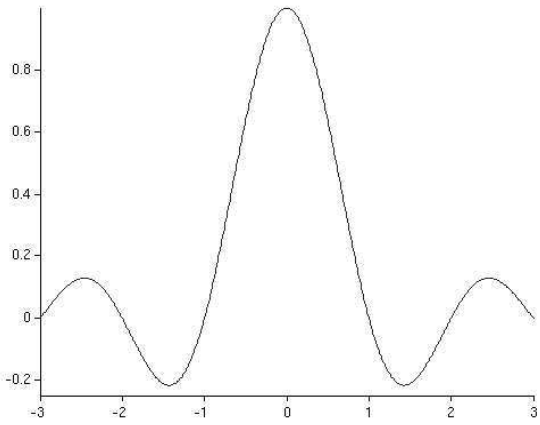
- **Sinc Interpolation**

$$\varphi_{\text{int}}(x) = \frac{\sin(\pi x)}{\pi x}$$

  - infinite support interval $(-\infty, \infty)$
  - can be shown to give the exact (!) reconstruction for a bandlimited signal that has been sampled according to the sampling theorem (Lecture 3)
  - yields an infinitely times differentiable (C^∞) interpolant
  - impossible to implement exactly due to its infinite support and its slow decay: approximated by truncated or windowed sinc functions

Synthesis Functions for Classical Interpolation (4)





(a) Left: Synthesis function for Keys interpolation ($a = -1/2$). **(b) Right:** Synthesis function for sinc interpolation. Author: N. Khan (2005).

M I A
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30

Synthesis Functions for Classical Interpolation (5)

Truncated Sinc Interpolation

- truncate the sinc function outside some interval $[-n, n]$, $n \in \mathbb{N}$
- gives the so-called *Dirichlet apodisation*

$$\varphi_{\text{int}}(x) = \begin{cases} \frac{\sin(\pi x)}{\pi x} & \text{for } |x| \leq n, \\ 0 & \text{else.} \end{cases}$$

- Unfortunately, this leads only to a C^0 interpolant.
- This does not even satisfy the partition of unity!
  May create severe shifts in the average grey level of the image.

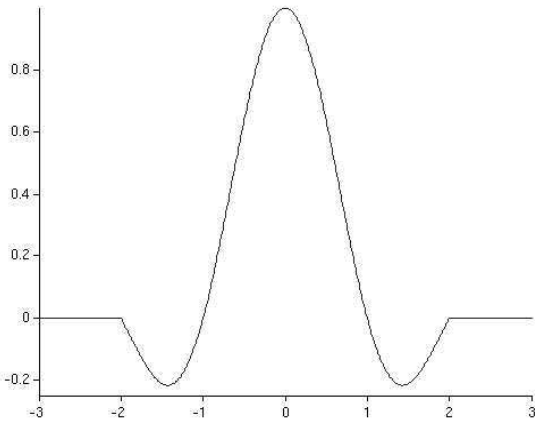
Windowed Sinc Interpolation

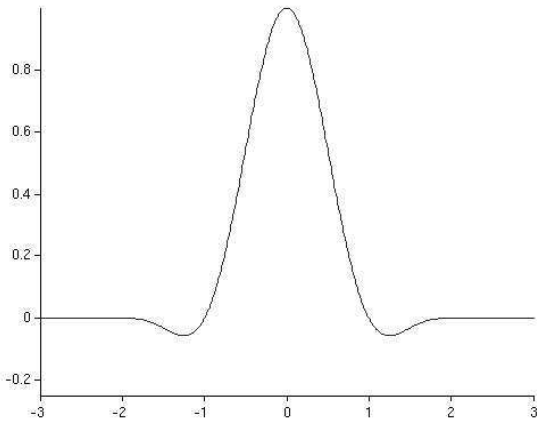
- multiplies the sinc function with a window of support $[-n, n]$, $n \in \mathbb{N}$
- using e.g. a so-called Hanning window gives the *Hanning apodisation*

$$\varphi_{\text{int}}(x) = \begin{cases} \frac{\sin(\pi x)}{\pi x} \left(\frac{1}{2} + \frac{1}{2}\cos\left(\frac{\pi x}{n}\right)\right) & \text{for } |x| \leq n, \\ 0 & \text{else.} \end{cases}$$

- creates a C^1 interpolant
- violates partition of unity as well, leading to greyscale shifts

Synthesis Functions for Classical Interpolation (6)





(a) Left: Synthesis function for truncated sinc interpolation ($n = 2$). **(b) Right:** Synthesis function for windowed sinc interpolation with a Hanning window ($n = 2$). Author: N. Khan (2005).

Generalised Interpolation

Motivation

- So far we have considered classical interpolation methods, where a weighted average of the function values $\{f_{\mathbf{k}} \,|\, \mathbf{k} \in \mathbb{Z}^m\}$ is computed:

$$f(\mathbf{x}) = \sum_{\mathbf{k} \in \mathbb{Z}^m} f_{\mathbf{k}} \, \varphi_{\text{int}}(\mathbf{x} - \mathbf{k}).$$

- This restricts the choice of suitable synthesis functions to interpolating synthesis functions that satisfy the interpolation condition

$$\varphi_{\text{int}}(\mathbf{k}) = \begin{cases} 1 & \text{for } \mathbf{k} = (0, ..., 0)^\top, \\ 0 & \text{for } \mathbf{k} \in \mathbb{Z}^m \setminus \{(0, ..., 0)^\top\}. \end{cases}$$

- It is possible to obtain novel, better methods when we do not insist on the interpolation condition and permit coefficients that do not coincide with the function values ?

Basic Idea Behind Generalised Interpolation

- Consider the *generalised interpolation* formula

$$f(\mathbf{x}) = \sum_{\mathbf{k} \in \mathbb{Z}^m} c_{\mathbf{k}} \, \varphi(\mathbf{x} - \mathbf{k})$$

  where

  - the synthesis function φ can be *noninterpolating*, i.e. it does not have to satisfy the interpolation condition,
  - the coefficients $\{c_{\mathbf{k}} \,|\, \mathbf{k} \in \mathbb{Z}^m\}$ are computed from the function values $\{f_{\mathbf{k}} \,|\, \mathbf{k} \in \mathbb{Z}^m\}$ and φ.

- Generalised interpolation proceeds in two steps:
  - compute the coefficients $\{c_{\mathbf{k}} \,|\, \mathbf{k} \in \mathbb{Z}^m\}$
  - interpolate with the preceding formula

- This two-step procedure can pay off in terms of interpolation quality: Noninterpolating synthesis functions with a finite support can be equivalent to interpolating synthesis functions with an infinite support.

Generalised Interpolation (3)

Desirable Properties of Noninterpolating Synthesis Functions

- Just as for interpolating synthesis functions, separability and symmetry are highly desirable.
- Partition of unity has to be modified. One can show that

$$\frac{1}{c_0} = \sum_{\mathbf{k}\in\mathbb{Z}^m} \varphi(\mathbf{x}-\mathbf{k}) \qquad \text{for all } \mathbf{x}\in\mathbb{R}^m$$

guarantees that a constant signal remains unaltered.

Quality Criteria of a Good Generalised Interpolant

- We have the same criteria as for a classical interpolant:
  - small support region of the synthesis function
  - high approximation order
  - high regularity
- Moreover, it should be easy to compute the coefficents $\{c_{\mathbf{k}} \,|\, \mathbf{k}\in\mathbb{Z}^m\}$.

Generalised Interpolation (4)

How Are the Coefficients Computed ?

- For the sake of simplicity, consider the 1-D case with finitely many equidistant interpolation data: f_1, f_2,..., f_N at the points $x_1 = 1$, $x_2 = 2$,...,$x_N = N$.
- The conditions

$$\sum_{k=1}^{N} c_k\, \varphi(i-k) \;=\; f_i \qquad \text{for } i = 1,...,N$$

describe a linear system of equations for the N unknowns c_1, c_2,..., c_N:

$$\begin{pmatrix} \varphi(0) & \varphi(1) & \dots & \varphi(N-1) \\ \varphi(1) & \varphi(0) & \dots & \varphi(N-2) \\ \vdots & \vdots & \dots & \vdots \\ \varphi(N-1) & \varphi(N-2) & \dots & \varphi(0) \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_N \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_N \end{pmatrix}$$

where we have used the symmetry condition $\varphi(x) = \varphi(-x)$.

- If φ has a small support, then we have a band matrix with a small band.

M I A
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30

Generalised Interpolation (5)

Which Synthesis Functions Are Used ?

- The most popular synthesis functions are the ones derived from B-splines.
- We start with the box function for nearest neighbour interpolation and define it as synthesis function β_0:

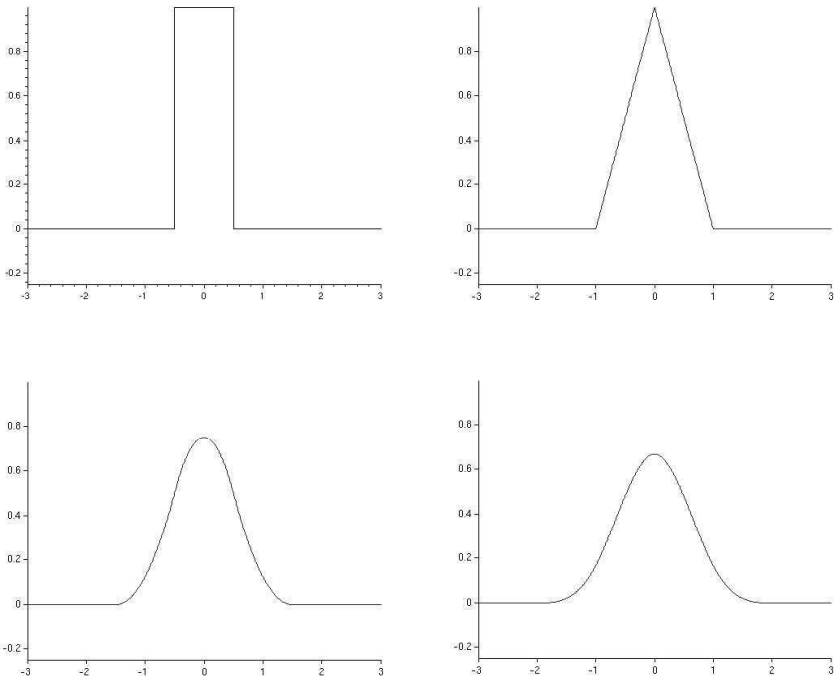
$$\beta_0(x) = \begin{cases} 1 & \text{for } |x| < \frac{1}{2}, \\ \frac{1}{2} & \text{for } |x| = \frac{1}{2}, \\ 0 & \text{else} \end{cases}$$

- The synthesis functions β_n with $n = 1, 2, ...$ are derived in an iterative manner by convolution with β_0 (see also Assignment T1, Problem 1):

$$\begin{aligned} \beta_1 &= \beta_0 * \beta_0, \\ \beta_2 &= \beta_1 * \beta_0, \\ \beta_3 &= \beta_2 * \beta_0. \end{aligned}$$

- β_n is a piecewise polynomial of degree n.
  It is symmetric and $n - 1$ times continuously differentiable.

Generalised Interpolation (6)



The first four synthesis functions derived from B-splines. **(a) Top left:** β_0. **(b) Top right:** β_1. **(c) Bottom left:** β_2. **(d) Bottom right:** β_3. Note that β_0 and β_1 satisfy the interpolation condition, while β_2 and β_3 are noninterpolating synthesis functions. Author: N. Khan (2005).

Generalised Interpolation (7)

Explicit Formulae for the Synthesis Functions Made of B-Splines

$$\beta_0(x) = \begin{cases} 1 & \text{for } |x| < \frac{1}{2}, \\ \frac{1}{2} & \text{for } |x| = \frac{1}{2}, \\ 0 & \text{else} \end{cases}$$

$$\beta_1(x) = \begin{cases} 1 - |x| & \text{for } |x| < 1, \\ 0 & \text{else} \end{cases}$$

$$\beta_2(x) = \begin{cases} \frac{3}{4} - x^2 & \text{for } |x| < \frac{1}{2}, \\ \frac{1}{2}(\frac{3}{2} - |x|)^2 & \text{for } \frac{1}{2} \le |x| < \frac{3}{2}, \\ 0 & \text{else} \end{cases}$$

$$\beta_3(x) = \begin{cases} \frac{2}{3} - x^2 + \frac{1}{2}|x|^3 & \text{for } |x| < 1, \\ \frac{1}{6}(2 - |x|)^3 & \text{for } 1 \le |x| < 2, \\ 0 & \text{else} \end{cases}$$

Generalised Interpolation (8)

Cubic B-Spline Interpolation

- most popular generalised interpolation
- support interval $[-2, 2]$.
- approximation order $L = 4$ (reproduces cubic functions)
- creates a twice continuously differentiable (C^2) interpolant
- one order better than Keys interpolation which has comparable costs
- Many evaluations have shown that it offers the best cost–performance ratio.

Finding the Coefficients for Cubic B-Spline Interpolation

- synthesis function β_3 at integer values gives

$$\beta_3(k) = \begin{cases} \frac{2}{3} & \text{for } k = 0, \\ \frac{1}{6} & \text{for } k = \pm 1, \\ 0 & \text{for other integer values } k. \end{cases}$$

- leads to the following tridiagonal system for the interpolation coefficients:

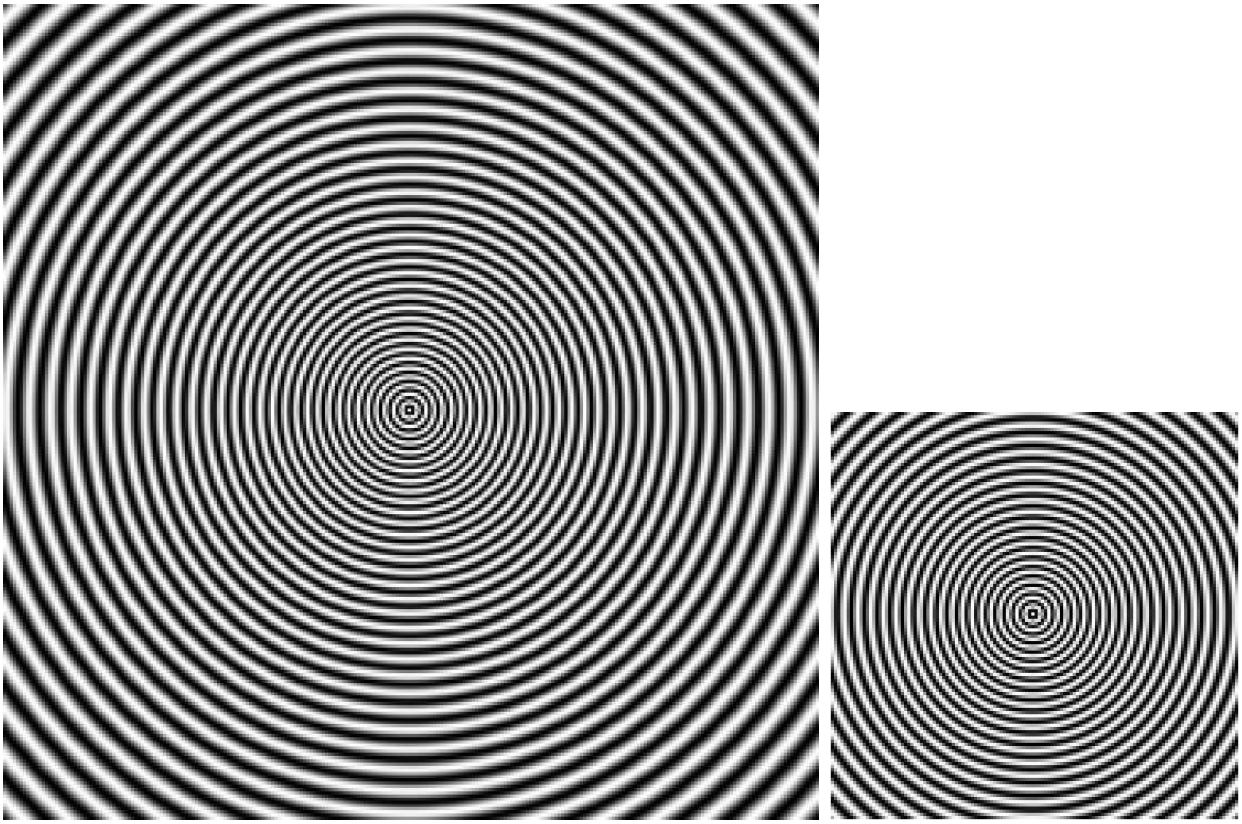
$$\begin{pmatrix} \frac{2}{3} & \frac{1}{6} & 0 & \dots & \dots & 0 \\ \frac{1}{6} & \frac{2}{3} & \frac{1}{6} & 0 & \dots & 0 \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ 0 & \dots & 0 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \\ 0 & \dots & \dots & 0 & \frac{1}{6} & \frac{2}{3} \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ \vdots \\ c_{N-1} \\ c_N \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ \vdots \\ f_{N-1} \\ f_N \end{pmatrix}$$

- can be efficiently solved with the Thomas algorithm from Lecture 13

Experiments

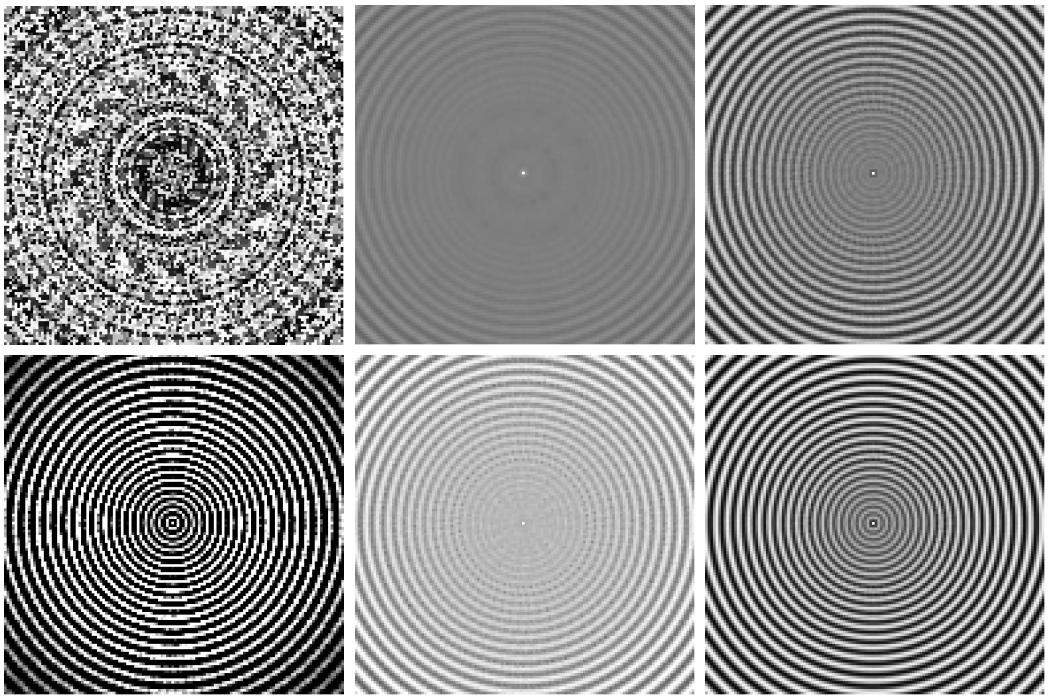
- Consider a rotationally invariant test image.
- Perform 15 rotations by 360/15 degrees where the output of any step is used as input of the next step.
- For an ideal interpolant, the resulting image should be identical with the original one.

Experiments (2)



(a) Left: Test image. **(b) Right:** Central square. Authors: P. Thévenaz, T. Blu, M. Unser (2000).

Experiments (3)



Comparison of different interpolants after 15 rotations with 360/15 degrees. **(a) Top left:** Nearest neighbour interpolation. **(b) Top middle:** Linear interpolation. **(c) Top right:** Keys interpolation ($a = -1/2$). **(d) Bottom left:** Truncated sinc interpolation ($n = 2$). Note the grey level shift. **(e) Bottom middle:** Windowed sinc interpolation ($n = 2$). Note the grey level shift. **(f) Bottom right:** Cubic B-spline interpolation. Authors: P. Thévenaz, T. Blu, M. Unser (2000).

M I A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26
27 28
29 30

Summary

- Interpolation recovers continuous data from discrete samples.
- Classical interpolation involves the function values and synthesis functions that satisfy the interpolation condition.
- Examples: nearest neighbour, linear, Keys, sinc interpolation
- Generalised interpolation renounces the interpolation condition and requires to compute coefficients.
- This extra effort is counterbalanced by better quality.
- Example: Cubic spline interpolation.
  It offers the best cost–performance ratio.

M I A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26
27 28
29 30

Literature

- P. Thévenaz, T. Blu, M. Unser: Image interpolation and resampling. In I. N. Bankman (Ed.): *Medical Imaging, Processing and Analysis*. Academic Press, San Diego, pp. 393–420, 2000.
  `http://bigwww.epfl.ch/publications/thevenaz9901.html`
  (excellent paper that formed the basis of the current lecture)
- H. S. Hou, H. C. Andrews: Cubic splines for image interpolation and digital filtering. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. 26, No. 6, pp. 508–517, Dec. 1978.
  (one of the first papers on spline interpolation in image analysis)
- T. M. Lehmann, C. Gönner, K. Spitzer: Survey: Interpolation methods in medical image processing. *IEEE Transactions on Medical Imaging*, Vol. 18, No. 11, pp. 1049–1075,Nov. 1999.
  `http://libra.imib.rwth-aachen.de/lehmann/paper.php`
  (experimental evaluation of numerous methods)
- E. H. W. Meijering, W. J. Niessen, M. A. Viergever: Quantitative evaluation of convolution-based methods for medical image interpolation. *Medical Image Analysis*, Vol. 5, No. 2, pp. 111–126, 2001.
  `http://imagescience.bigr.nl/meijering/publications/download/media2001.pdf`
  (another detailed experimental evaluation)
- E. Meijering: A chronology of interpolation: From ancient astronomy to modern signal and image processing. *Proceedings of the IEEE*, Vol. 90, No. 3, pp. 319–342, March 2002.
  `http://imagescience.bigr.nl/meijering/publications/abstracts/pieee2002.html`
  (provides interesting historical facts)