Surface and Gradient Estimation

Leonid I. Dimitrov and Miloš Šrámek
Commission for Scientific Visualization
Austrian Academy of Sciences

Introduction

- Another view/model of volumetric data
- Reduces volume visualization problem to conventional CG-methods
- Poses new problems:
  - How to find the surfaces?
  - How to find their normals?

Surface Extraction

- Contour tracking (Keppel 75, Fuchs 77)
- Opaque cubes (Herman and Liu 1979)
- Marching cubes (Lorensen and Cline 1987)
- Dividing cubes (- ,, -)
- Marching tetrahedra (- ,, -)
- Marching triangles (Hilton et al, 1996)
- Surface Detection by Ray Casting (Höhne et al, 1988)

Surface Extraction Fundamentals

- Assume volume contains surfaces
- Traverse in object order
- Classify cells as inside/outside („on“/„off“)
- Fit constant value surfaces („isosurfacing“)
- Render surfaces

Contour Tracking

- Define a threshold value
- Trace a closed contour in each slice
- Connect contours in adjacent slices
- Perform a tesselation, e.g. triangulation
- Render polygons

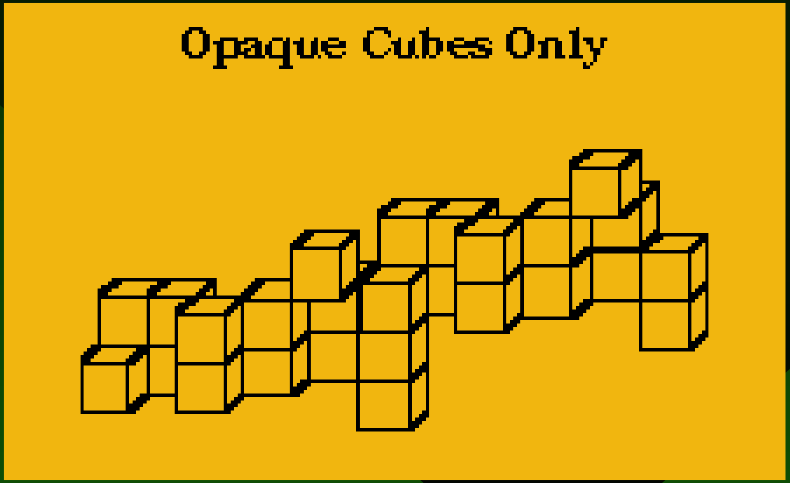
Contour Tracking

Contour Tracking

- Advantages
  - Simple to understand AND implement
  - Known surface rendering methods exist
  - Parallelizable
- Disadvantages
  - Not suited for high detail
  - Not applicable for amorphous data

Opaque Cubes

- Herman and Liu: „Cuberille“
- Represent „on“-cells as hexahedra
- Render six faces with traditional methods



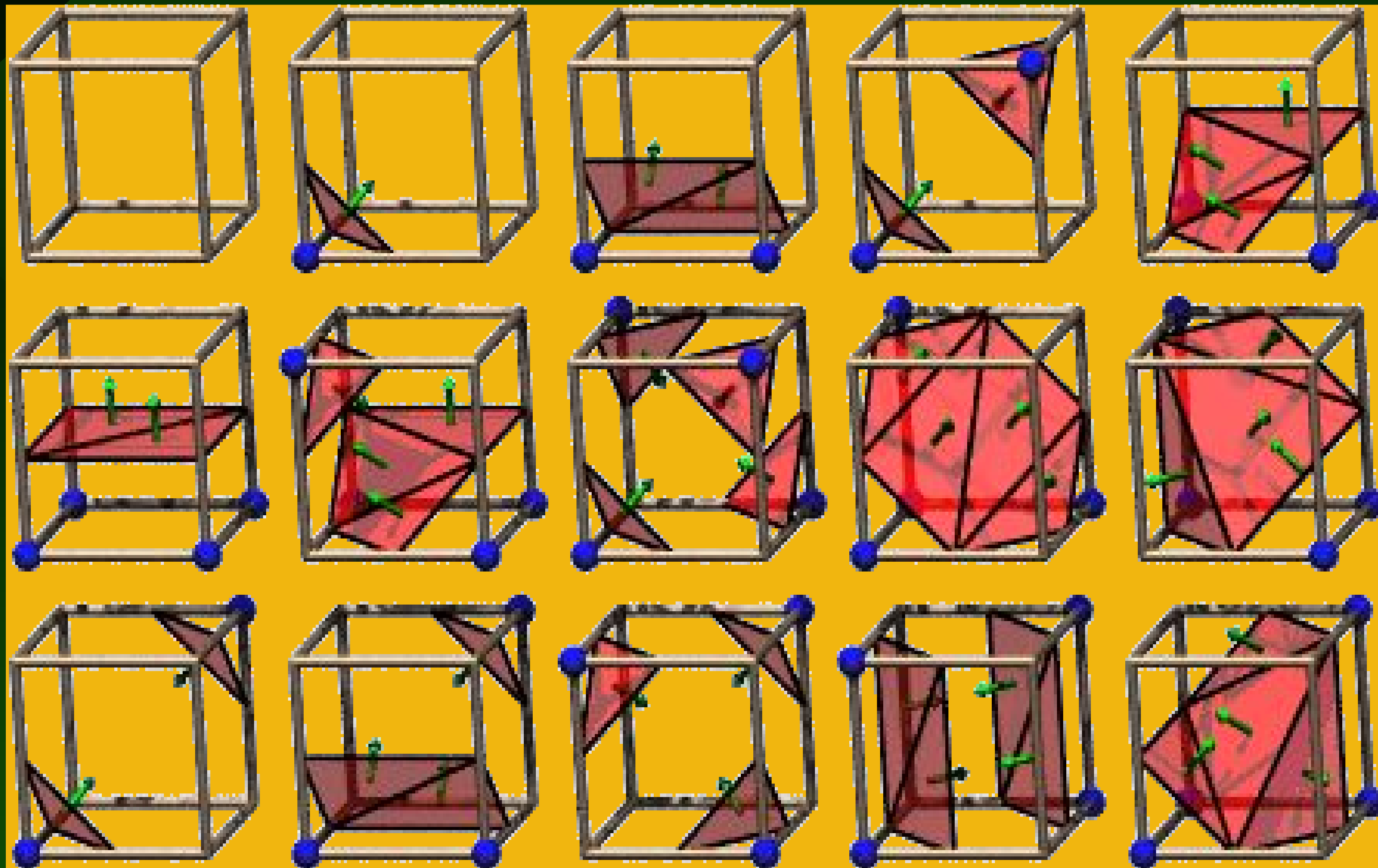
Opaque Cubes

- Advantages
  - Straight-forward implementation
  - Fast
  - Easy classification
- Disadvantages
  - Binary classification
  - Jaggy images
  - Throws away data between surfaces

Marching Cubes

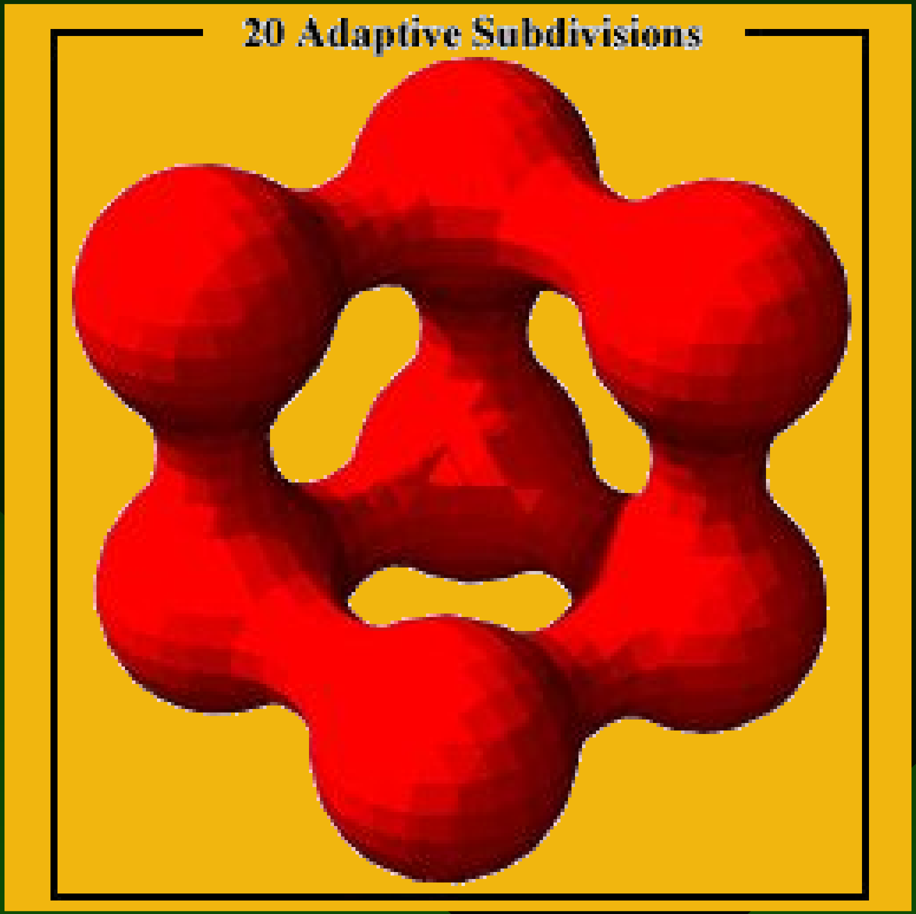
- Specify a threshold value
- Identify voxels bracketing the isosurface
- Examine the voxels and produce a set of polygons for each one
- Use a table lookup to reduce intersection tests
- Use linear interpolation along the edges

Marching Cubes



The 15 Cube Combinations

Marching Cubes



Dividing Cubes

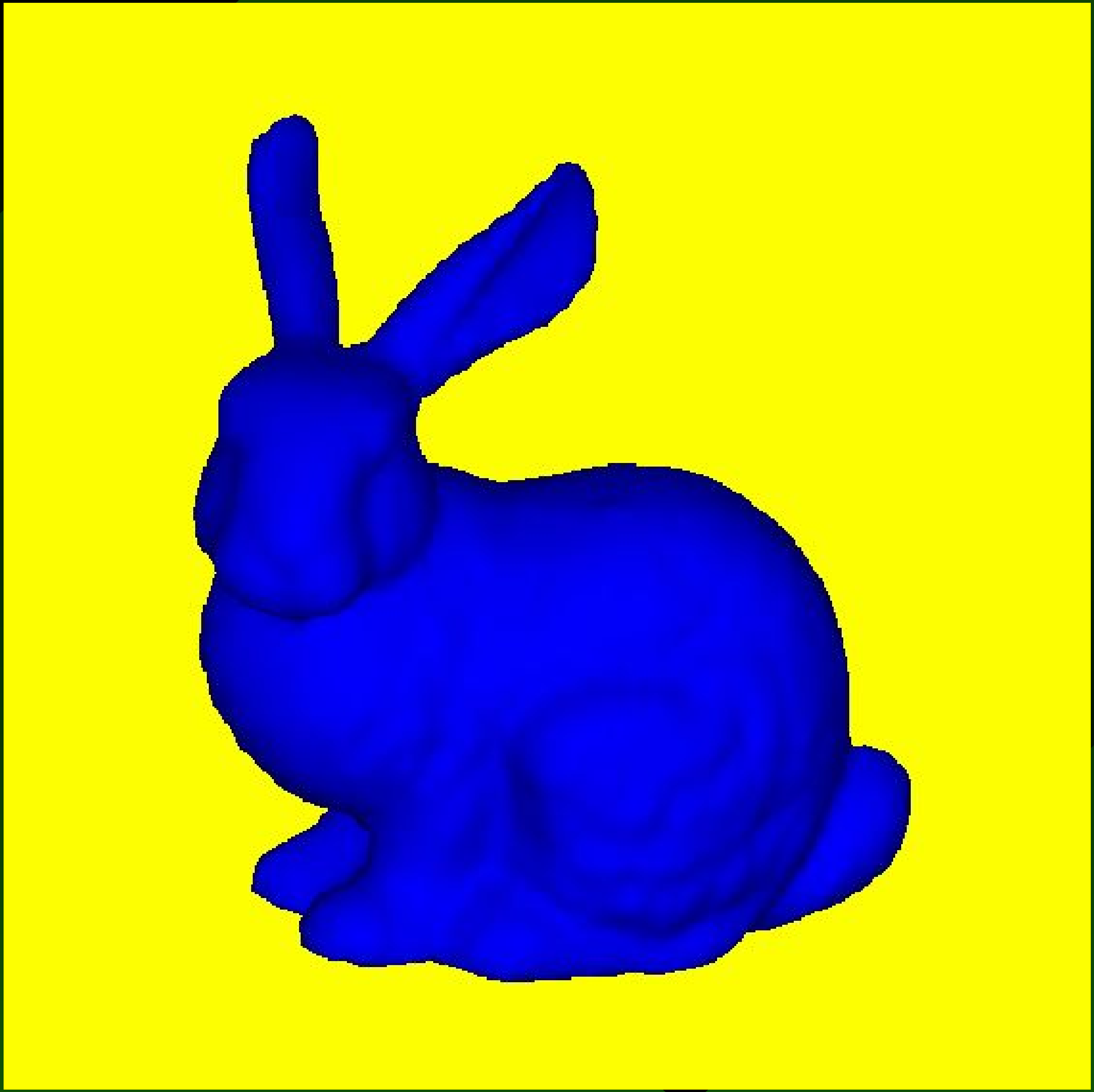
- Specify a threshold value
- Identify voxels bracketing the isosurface
- Project voxel to image plane
- If voxel projection covers less than a pixel render it
- Otherwise subdivide voxel and continue

Marching Tetrahedra

- Subdivide cells into tetrahedra (5, 6, or 24)
- Only 3 unique entries in table
- Maximum of 2 triangles/tetrahedra

Marching Triangles

- Tries to overcome restrictions imposed by previous volume-based approaches
- Generates a Delaunay triangulation using a local 3D constraint
- Results in uniform triangle shape
- Claims factor 3 improvement in representation efficiency and computational cost



Surface Extraction Advantages

- Uses known rendering methods
- Can take advantages of hardware
- View/light changes require only re-rendering (no pre-processing)
- Compact storage and transmission
- Good spatial coherence

Surface Extraction Disadvantages

- Requires binary classification
- Throws away data
- False positives and negatives
- Handles small features poorly
- Requires user intervention sometimes
- Amorphous data has no thin surfaces

Surface Extraction Comparison

- Contour tracking basically obsolete
- Dividing cubes is best (without hardware)
  - No clipping
  - Polygons/triangles are pixelsized anyway
- Tetrahedra reduce ambiguity

Gradient Estimation

- Ideal gradient

$$\nabla f(x, y, z) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z} \right)$$

- Discrete estimators
  - Central difference
  - Intermediate difference
- Derivative filters

Frequency Analysis

- Ideal interpolation filter:

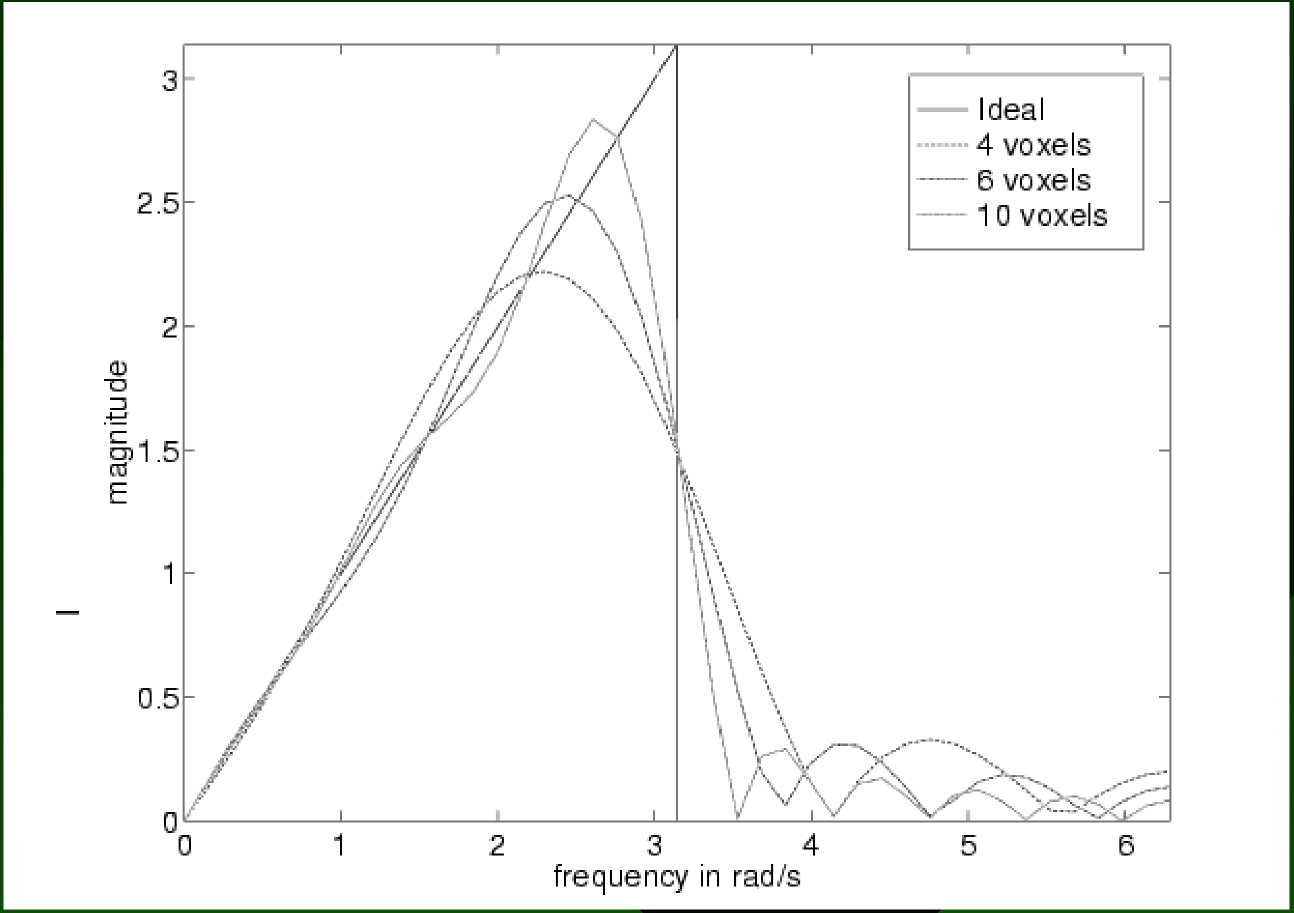
$$f(x) = f(x) * \mathrm{sinc}(x) = \sum_k f[k]\,\mathrm{sinc}(\mathrm{x} - \mathrm{k})$$

- Ideal derivative filter:

$$f'(x) = f(x) * \mathrm{sinc}'(x) = \sum_k f[k]\,\mathrm{sinc}'(\mathrm{x} - \mathrm{k})$$

$$\frac{d}{dx}\left(\frac{\sin \pi x}{\pi x}\right) = \frac{\cos \pi x - \mathrm{sinc}\, \pi x}{x}$$

Frequency Domain



Commonly Used Gradient Estimators

- Central difference

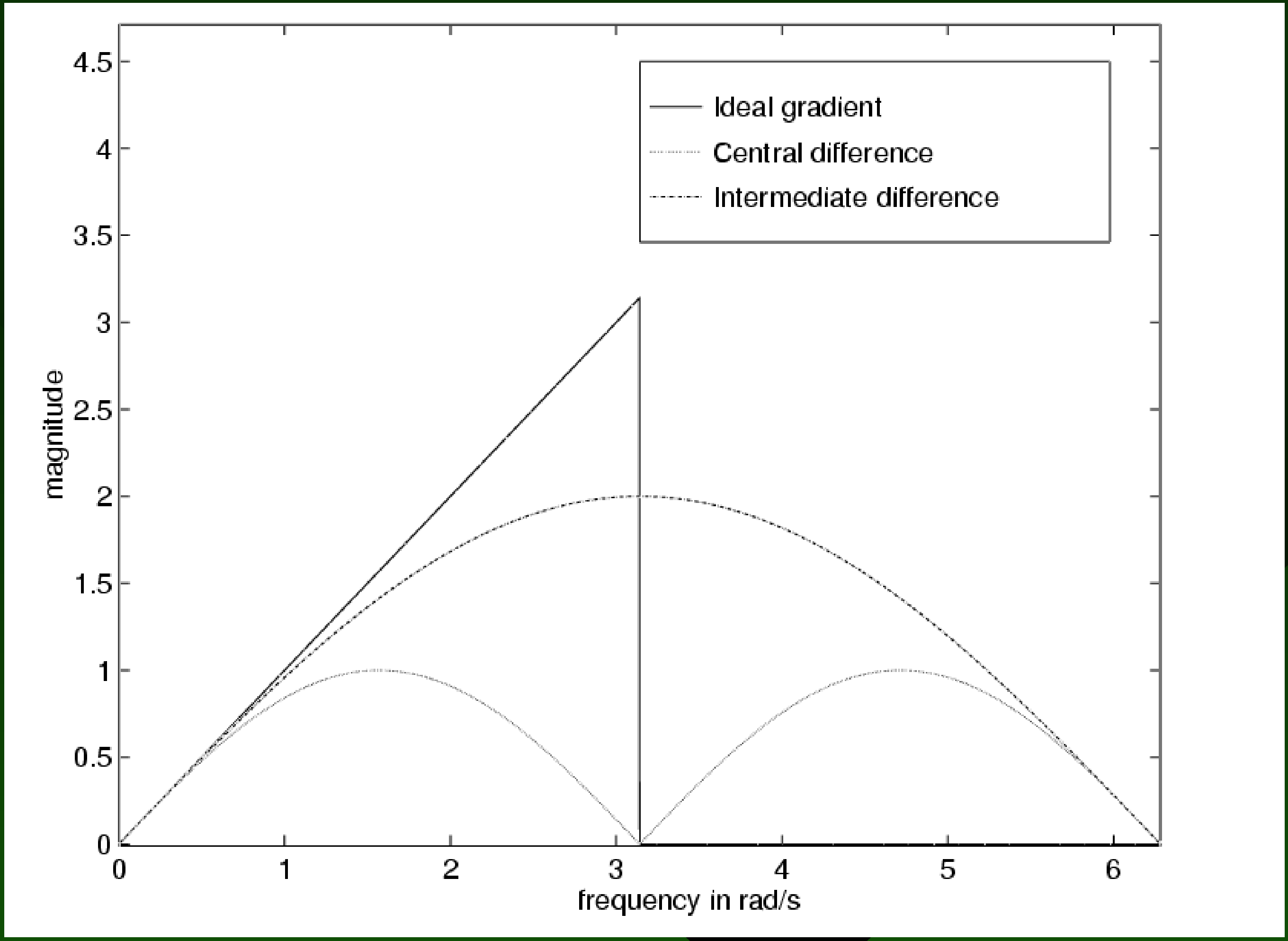
$$\nabla f\left(x_i, y_j, z_k\right) = \begin{pmatrix} \dfrac{f\left(x_{i+1}, y_j, z_k\right) - f\left(x_{i-1}, y_j, z_k\right)}{2} \\ \dfrac{f\left(x_i, y_{j+1}, z_k\right) - f\left(x_i, y_{j-1}, z_k\right)}{2} \\ \dfrac{f\left(x_i, y_j, z_{k+1}\right) - f\left(x_i, y_j, z_{k-1}\right)}{2} \end{pmatrix}$$

Commonly Used Gradient Estimators

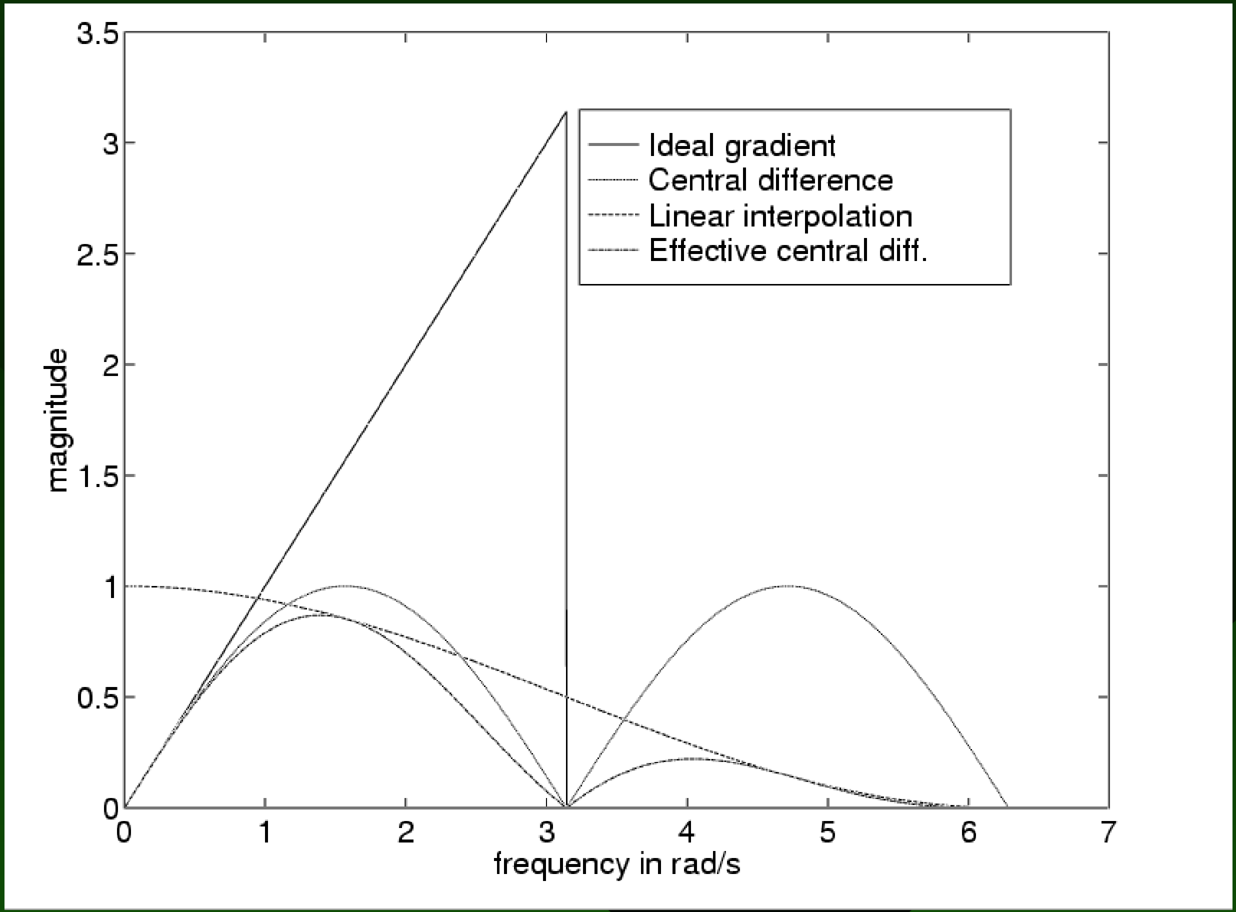
- Intermediate difference

$$\nabla f\left(x_{i/2}, y_{j/2}, z_{k/2}\right) = \begin{pmatrix} f\left(x_{i+1}, y_j, z_k\right) - f\left(x_i, y_j, z_k\right) \\ f\left(x_i, y_{j+1}, z_k\right) - f\left(x_i, y_j, z_k\right) \\ f\left(x_i, y_j, z_{k+1}\right) - f\left(x_i, y_j, z_k\right) \end{pmatrix}$$

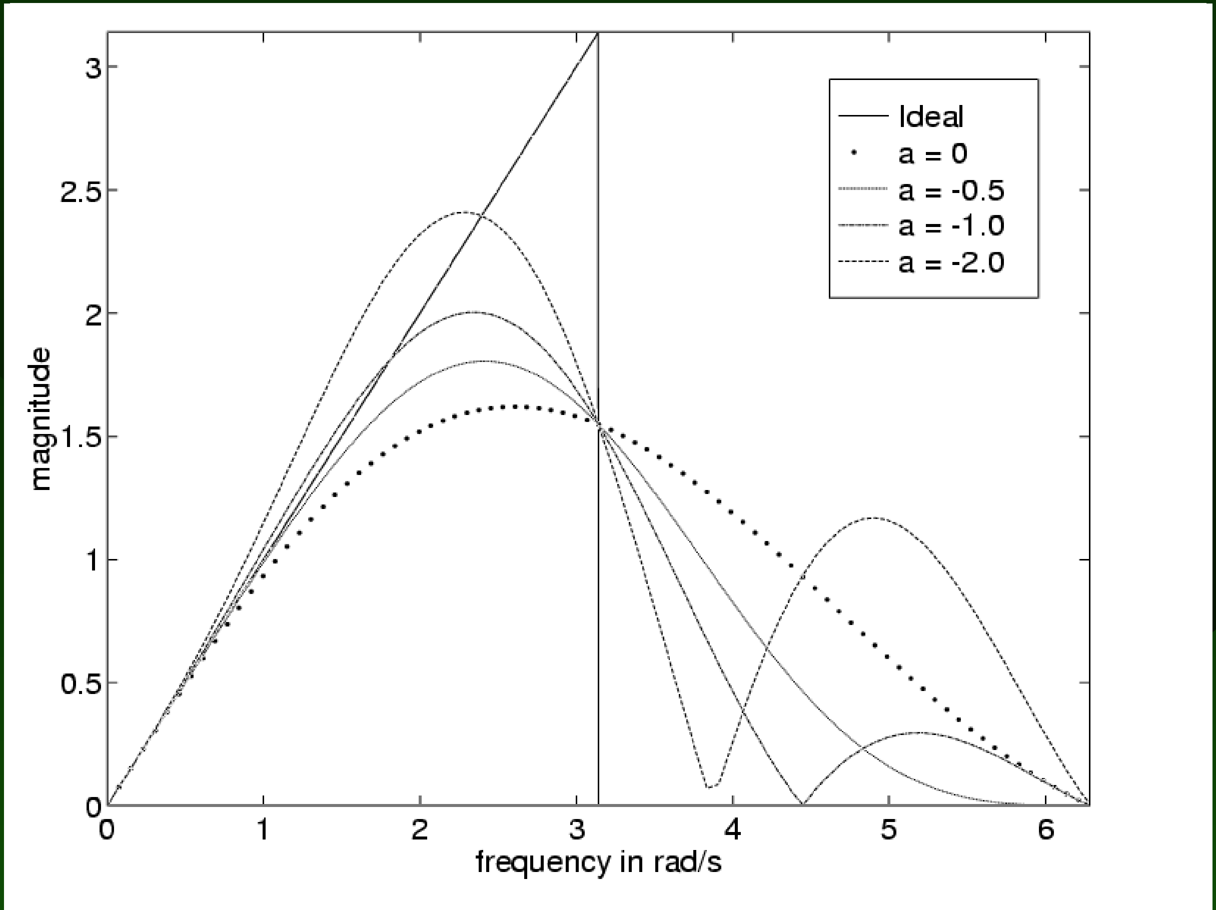
Commonly Used Gradient Estimators



Combining Gradient Estimation and Interpolation



Derivative Filter Evaluation by Frequency Analysis



Derivative Filter Design by Taylor Series Expansion

- Design is based on spatial smoothness and accuracy
- Frequency based methods control global errors
- Taylor series based methods control local errors
- Design generates filters of arbitrary smoothness and accuracy

Discrete Convolution

$$f_r^w(t) = \sum_{k=-\infty}^{\infty} f[k] \cdot w\left(\frac{t}{T} - k\right)$$

Taylor Series Expansion

$$f[k] = \sum_{n=0}^{N} \frac{f^{(n)}(t)}{n!} (kT - t)^{n} + \frac{f^{(N+1)}(\xi_k)}{(N+1)!} (kT - t)^{(N+1)}$$

Substitution of Taylor Series Expansion in Convolution

$$f_r^w(t) = \sum_{n=0}^{N} a_n^w(\tau) f^{(n)}(t) + r_{N,i}^w(\tau)$$

$$a_n^w(\tau) = \frac{T^n}{n!} \sum_{k=-\infty}^{\infty} (k-\tau)^n w(\tau - k)$$

$$r_{N,i}^w(\tau) \le \left(\max_{\xi \in [(i-M)T,\ (i+M)T]} (f^{(N+1)}(\xi)) \right) \left| a_{N+1}^w(\tau) \right|$$

or

$$r_N^w(\tau) \approx a_{N+1}^w(\tau) f^{(N+1)}(t)$$

Design Criteria: Accuracy

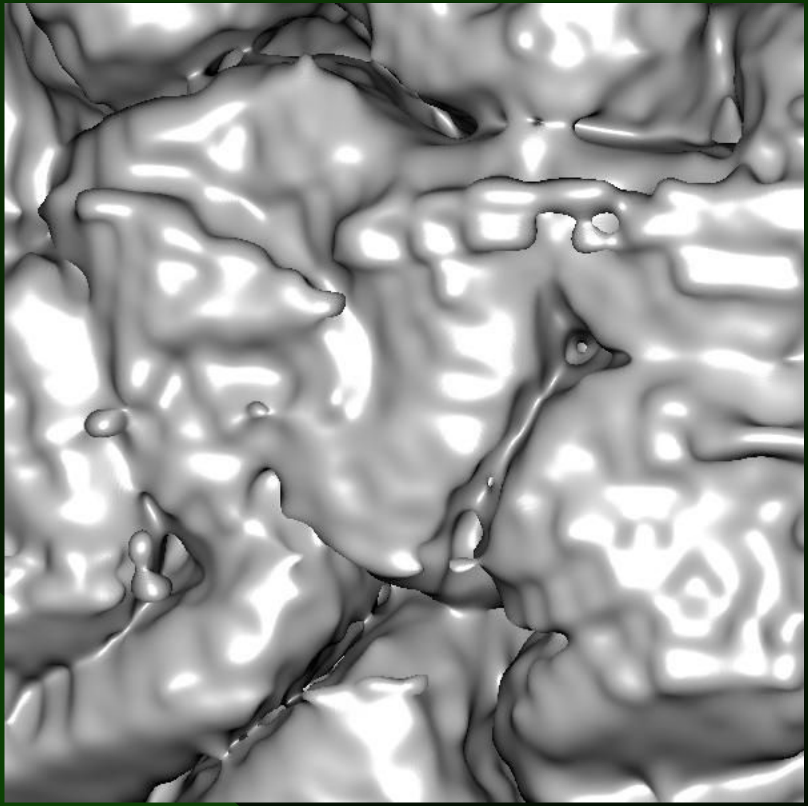
<u>*Condition 1*</u>: $a_n^w = 0$ for all $n < k$ and $a_k^w = 1$

<u>*Condition 2:*</u> $a_n^w = 0$ for all $k < n < N + k - 1$

Design Criteria: Smoothness

Condition 3: $w_k(\tau) \in C^M$ and $w_k^{(m)}(1) = w_{k+1}^{(m)}(0)$ for all k and all $m < M$, where $w_k^{(m)}$ denotes the m-th derivative of w_k.

Design Results: MRI rendering



Design Results: Pattern Mapping