

Extensions of the Zwart-Powell Box Spline for Volumetric Data Reconstruction on the Cartesian Lattice

Alireza Entezari and Torsten Möller

`aentezar@cs.sfu.ca`

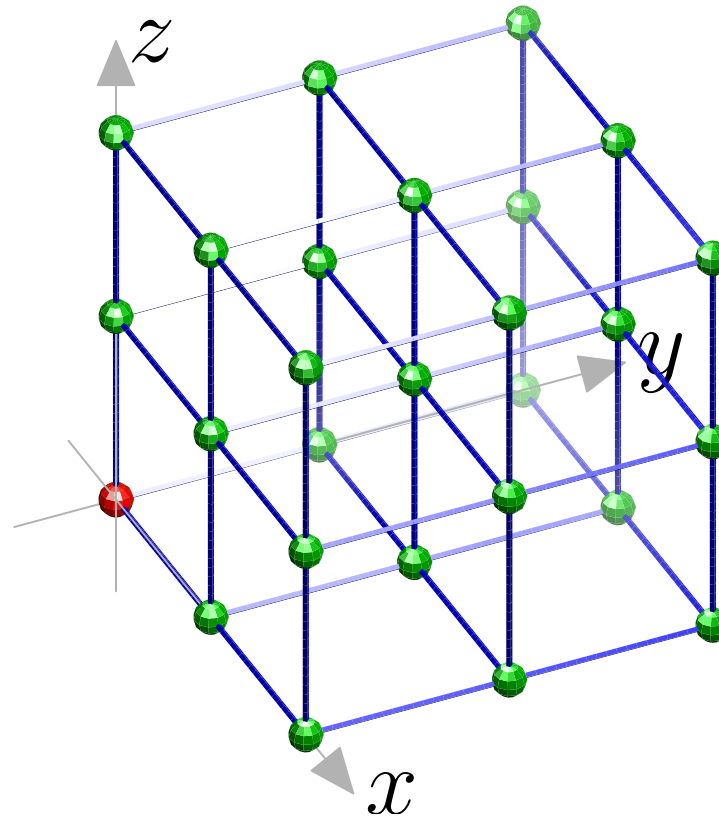
GrUVi Lab, School of Computing Science, Simon Fraser University



Reconstruction on the Cartesian Lattice

A ubiquitous problem in many computing areas:

- Scientific computing
- Visualization
- Computer Graphics
- ...



Interpolation/Reconstruction

- Studied in Numerical Analysis and Signal Processing
- 1-D interpolation/reconstruction, rich literature.



Interpolation/Reconstruction

- Studied in Numerical Analysis and Signal Processing
- 1-D interpolation/reconstruction, rich literature.
- Volume Data / Image Processing \Rightarrow 3-D/2-D



Interpolation/Reconstruction

- Studied in Numerical Analysis and Signal Processing
- 1-D interpolation/reconstruction, rich literature.
- Volume Data / Image Processing \Rightarrow 3-D/2-D
 - Radial Basis Functions: apply 1-D methods on the *norm* of vectors



Interpolation/Reconstruction

- Studied in Numerical Analysis and Signal Processing
- 1-D interpolation/reconstruction, rich literature.
- Volume Data / Image Processing \Rightarrow 3-D/2-D
 - Radial Basis Functions: apply 1-D methods on the *norm* of vectors
 - Tensor Product Solution: apply 1-D methods along *rows, columns, ...*



Interpolation/Reconstruction

- Studied in Numerical Analysis and Signal Processing
- 1-D interpolation/reconstruction, rich literature.
- Volume Data / Image Processing \Rightarrow 3-D/2-D
 - Radial Basis Functions: apply 1-D methods on the *norm* of vectors
 - Tensor Product Solution: apply 1-D methods along *rows, columns, ...*

essentially 1-D solutions, not true multi-D solutions. Our approach: true 3-D



Our Reconstruction Algorithm



tri-linear B-spline



tri-cubic B-spline



box spline Ξ



Box Splines

- True multi-dimensional basis functions



Box Splines

- True multi-dimensional basis functions
- Convenient piecewise polynomial approximation



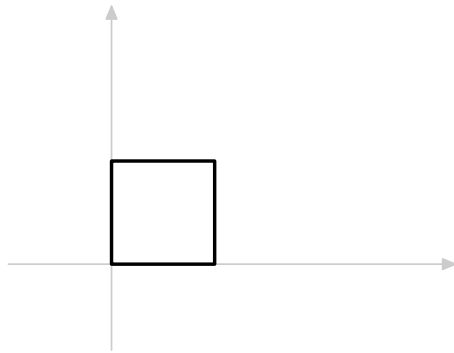
Box Splines

- True multi-dimensional basis functions
- Convenient piecewise polynomial approximation
- Not restricted to axis-aligned tensor product B-splines

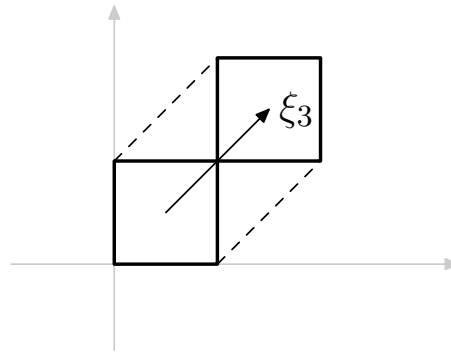


Box Splines

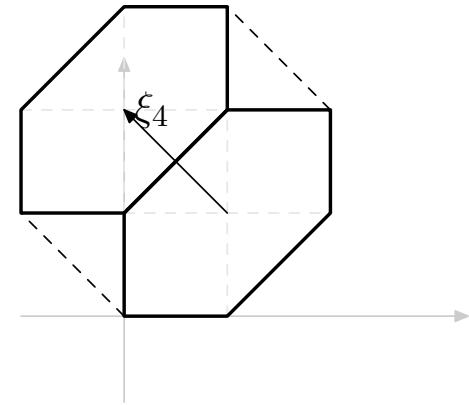
- True multi-dimensional basis functions
- Convenient piecewise polynomial approximation
- Not restricted to axis-aligned tensor product B-splines
- Obtained by *directional convolutions*



box, discontinuous



linear, C^0

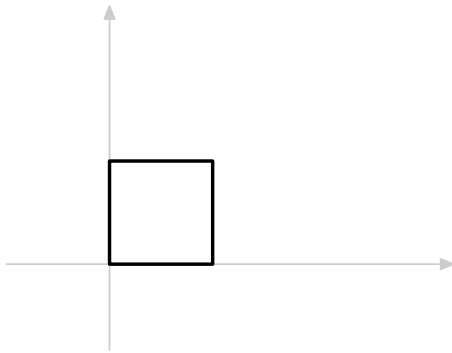


quadratic, C^1

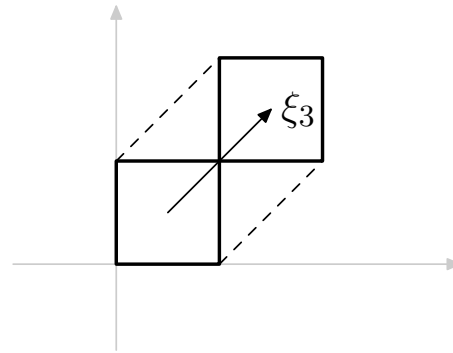


Box Splines

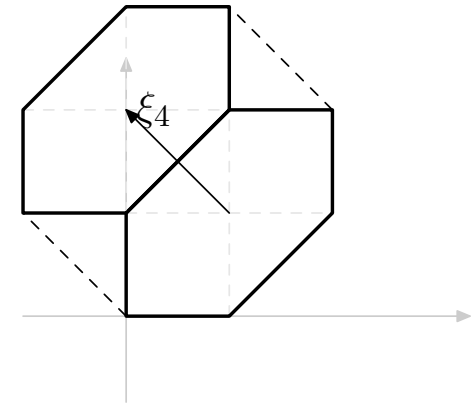
- True multi-dimensional basis functions
- Convenient piecewise polynomial approximation
- Not restricted to axis-aligned tensor product B-splines
- Obtained by *directional convolutions*



box, discontinuous



linear, C^0



quadratic, C^1

 *Zwart-Powell element*

Tri-variate Box Splines

- Despite attractive 2D examples, not many 3D box splines



Tri-variate Box Splines

- Despite attractive 2D examples, not many 3D box splines
- Reconstruction on Body Centered Cubic with 4-dir box splines.



Tri-variate Box Splines

- Despite attractive 2D examples, not many 3D box splines
- Reconstruction on Body Centered Cubic with 4-dir box splines.
- Extend the Zwart-Powell element to tri-variate setting



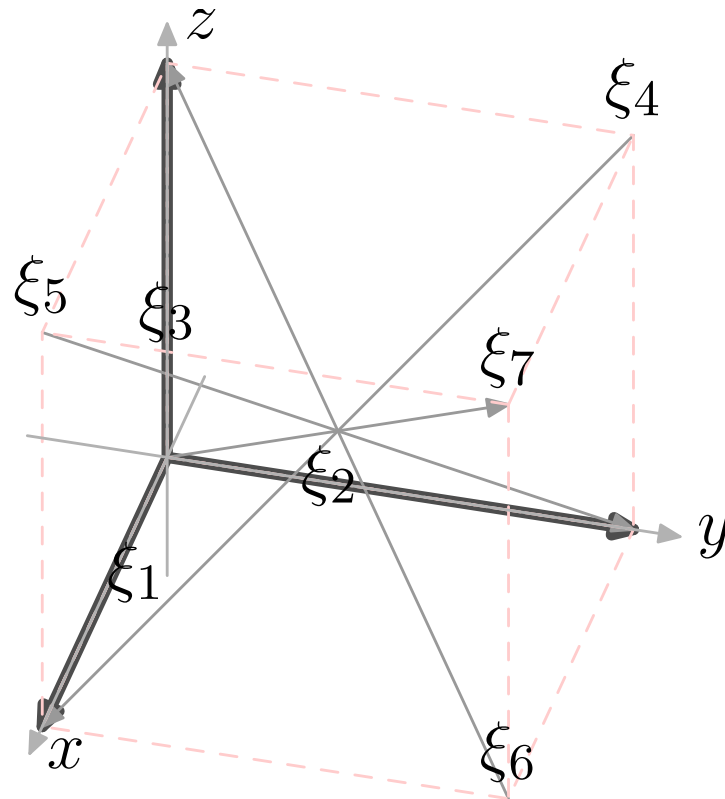
Tri-variate Box Splines

- Despite attractive 2D examples, not many 3D box splines
- Reconstruction on Body Centered Cubic with 4-dir box splines.
- Extend the Zwart-Powell element to tri-variate setting
- 7-directional box spline



Seven Directional Box Spline

- 3-axis aligned directions
- 4-diagonal directions



Seven Directional Box Spline

- Matrix of directions:

$$\mathbf{E} = \begin{bmatrix} 1 & 0 & 0 & 1 & -1 & -1 & 1 \\ 0 & 1 & 0 & -1 & 1 & -1 & 1 \\ 0 & 0 & 1 & -1 & -1 & 1 & 1 \end{bmatrix} \quad (1)$$

- $\rho = \min\{Z \in \mathbf{E}, \mathbf{E} \setminus Z \text{ does not span } \mathbb{R}^3\} \therefore \rho = 4.$
- $M_{\mathbf{E}} \in C^{\rho-2} = C^2$



Approximation Power

- Fourier Transform:

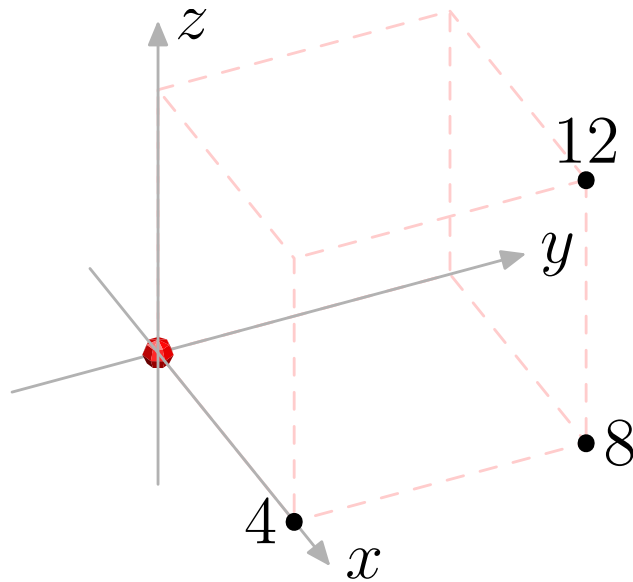
$$\hat{M}_{\Xi}(\omega) = \prod_{\xi \in \Xi} \text{sinc}(\xi \cdot \omega) \quad (2)$$

- Essentially product of *directional* sinc's
- One can verify zero-crossing of \hat{M}_{Ξ}

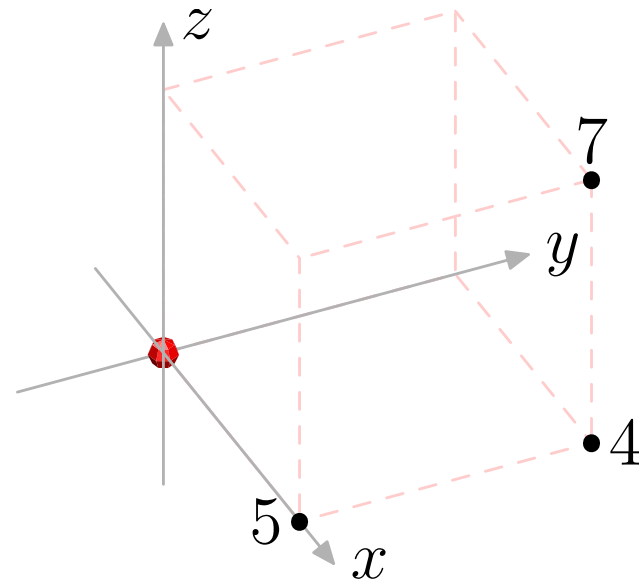


Approximation Power

- Number of Vanishing moments



tri-cubic B-spline



box spline M_{Ξ}

- A minimum of *four* vanishing moments.
- Approximation Power = 4



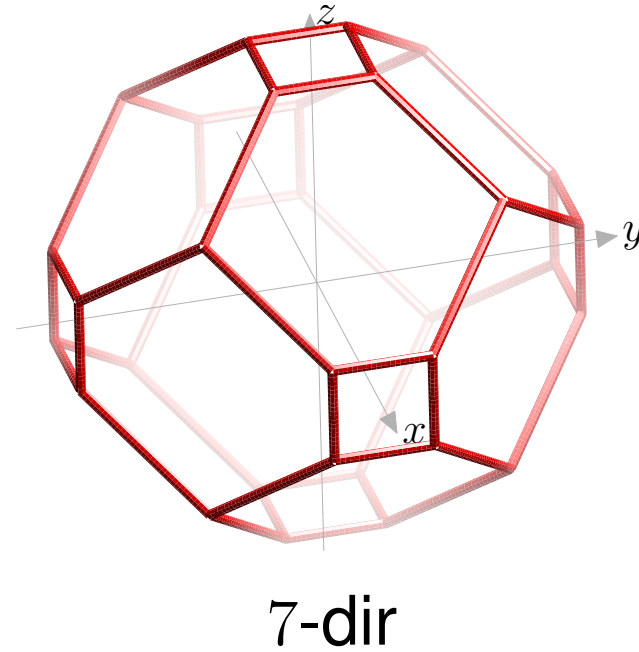
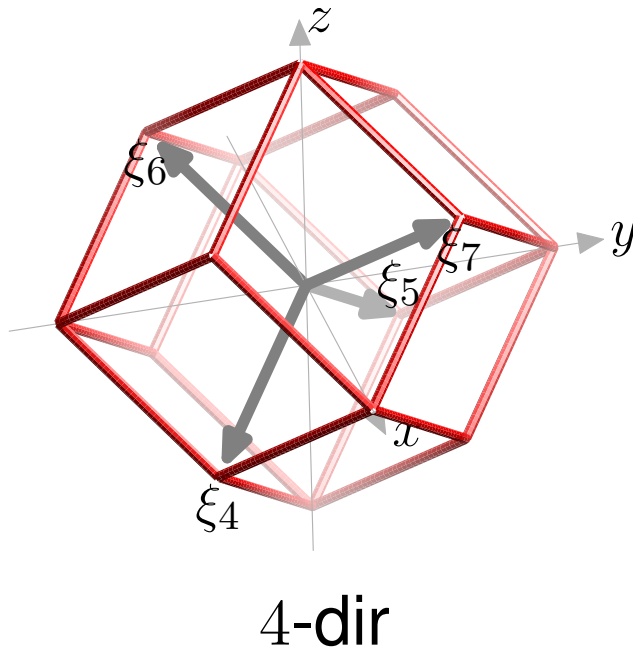
B-splines?

The smoothness and approximation power matches that of the *tri-cubic B-spline*.



Support

- Convolution of a 4-directional box spline with a cube



Computational Efficiency

- The support of the tri-cubic B-spline is a $4 \times 4 \times 4$ neighborhood



Computational Efficiency

- The support of the tri-cubic B-spline is a $4 \times 4 \times 4$ neighborhood
- The support of this box spline is *contained* in $5 \times 5 \times 5$ neighborhood.



Computational Efficiency

- The support of the tri-cubic B-spline is a $4 \times 4 \times 4$ neighborhood
- The support of this box spline is *contained* in $5 \times 5 \times 5$ neighborhood.
- Originally we considered this box spline to be slower



Computational Efficiency

- The support of the tri-cubic B-spline is a $4 \times 4 \times 4$ neighborhood
- The support of this box spline is *contained* in $5 \times 5 \times 5$ neighborhood.
- Originally we considered this box spline to be slower
- It turns out that only **53 points** fall inside the support.



Computational Efficiency

- The support of the tri-cubic B-spline is a $4 \times 4 \times 4$ neighborhood
- The support of this box spline is *contained* in $5 \times 5 \times 5$ neighborhood.
- Originally we considered this box spline to be slower
- It turns out that only **53 points** fall inside the support.
- 20% *faster than tri-cubic B-spline*



Numerical Implementation

- Table look up for box spline values
- The box spline M_{Ξ} can be decomposed into

$$M_{\Xi}(\mathbf{x}) = (M_{\Xi_1} * M_{\Xi_2})(\mathbf{x})$$

where

$$\Xi_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \Xi_2 = \begin{bmatrix} 1 & -1 & -1 & 1 \\ -1 & 1 & -1 & 1 \\ -1 & -1 & 1 & 1 \end{bmatrix}$$



Numerical Implementation

- the 3-directional M_{Ξ_1} is simply the box function over $[0..1)^3$



Numerical Implementation

- the 3-directional M_{Ξ_1} is simply the box function over $[0..1)^3$
- the 4-directional M_{Ξ_2} has been derived previously to be

$$M_{\Xi_2}(x, y, z) = 2 \max(0, 1 - \max(|x| + |y|, |x| + |z|, |y| + |z|))$$



Numerical Implementation

- the 3-directional M_{Ξ_1} is simply the box function over $[0..1)^3$
- the 4-directional M_{Ξ_2} has been derived previously to be
$$M_{\Xi_2}(x, y, z) = 2 \max(0, 1 - \max(|x| + |y|, |x| + |z|, |y| + |z|))$$
- Sample M_{Ξ_1} and M_{Ξ_2} on finite volume dataset



Numerical Implementation

- the 3-directional M_{Ξ_1} is simply the box function over $[0..1)^3$
- the 4-directional M_{Ξ_2} has been derived previously to be

$$M_{\Xi_2}(x, y, z) = 2 \max(0, 1 - \max(|x| + |y|, |x| + |z|, |y| + |z|))$$

- Sample M_{Ξ_1} and M_{Ξ_2} on finite volume dataset
- Perform discrete convolution on these two volumes

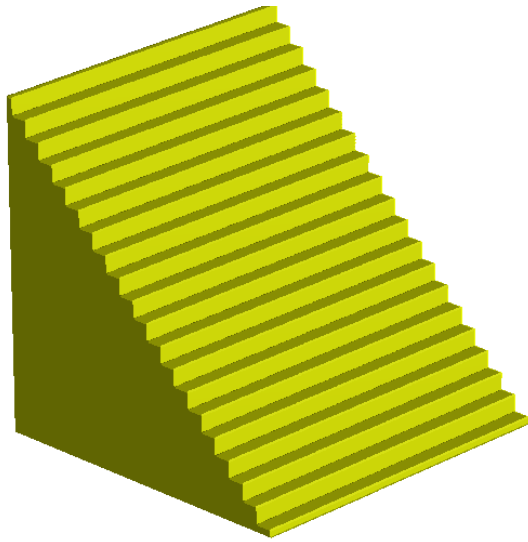


Numerical Implementation

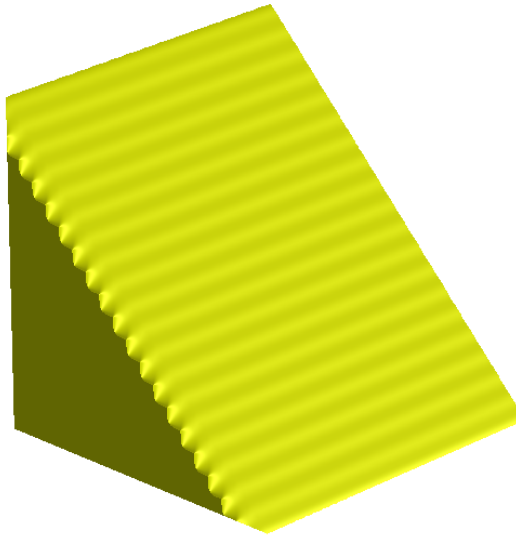
- the 3-directional M_{Ξ_1} is simply the box function over $[0..1)^3$
- the 4-directional M_{Ξ_2} has been derived previously to be
$$M_{\Xi_2}(x, y, z) = 2 \max(0, 1 - \max(|x| + |y|, |x| + |z|, |y| + |z|))$$
- Sample M_{Ξ_1} and M_{Ξ_2} on finite volume dataset
- Perform discrete convolution on these two volumes
- Can be efficiently performed by multiplication in Fourier domain



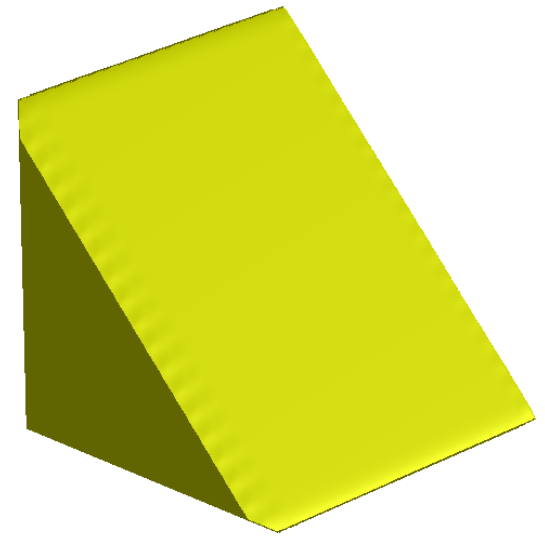
Stair-casing in Reconstruction



Voxelized surface



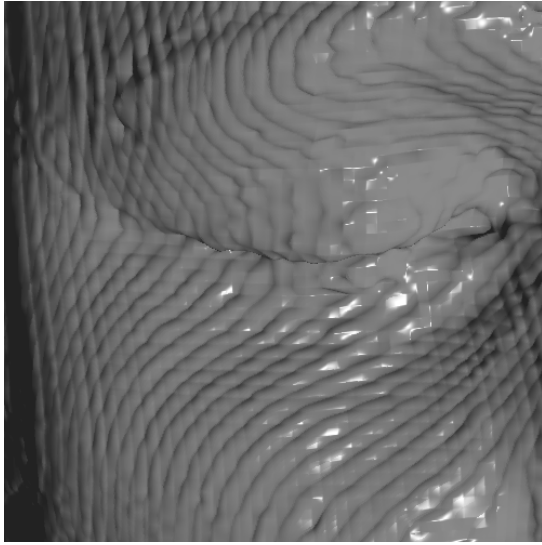
tri-cubic B-spline



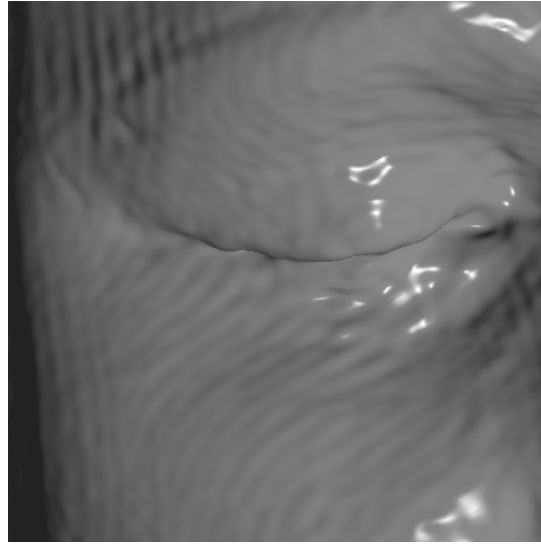
box spline Ξ



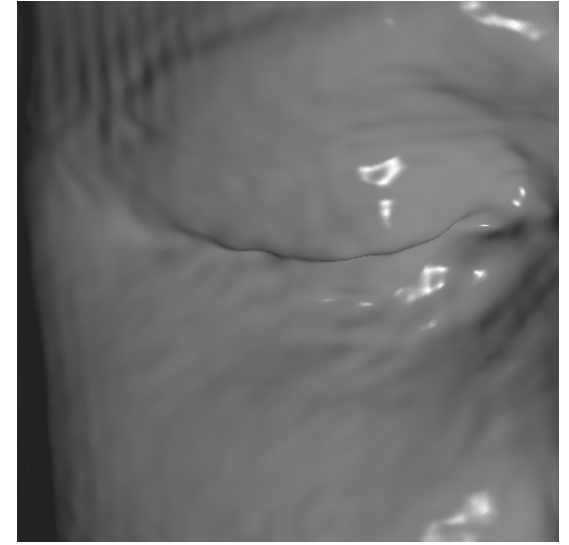
Grid-Aligned Artifacts



tri-linear B-spline



tri-cubic B-spline



box spline M_E



Our Reconstruction Algorithm



tri-linear B-spline



tri-cubic B-spline



box spline Ξ



Conclusion

The 7-directional box spline

- has the same smoothness and approximation power of the tri-cubic B-spline
- offers a more isotropic treatment of data and reduces axis-aligned artifacts
- is computationally more efficient than tri-cubic B-spline (20%)

