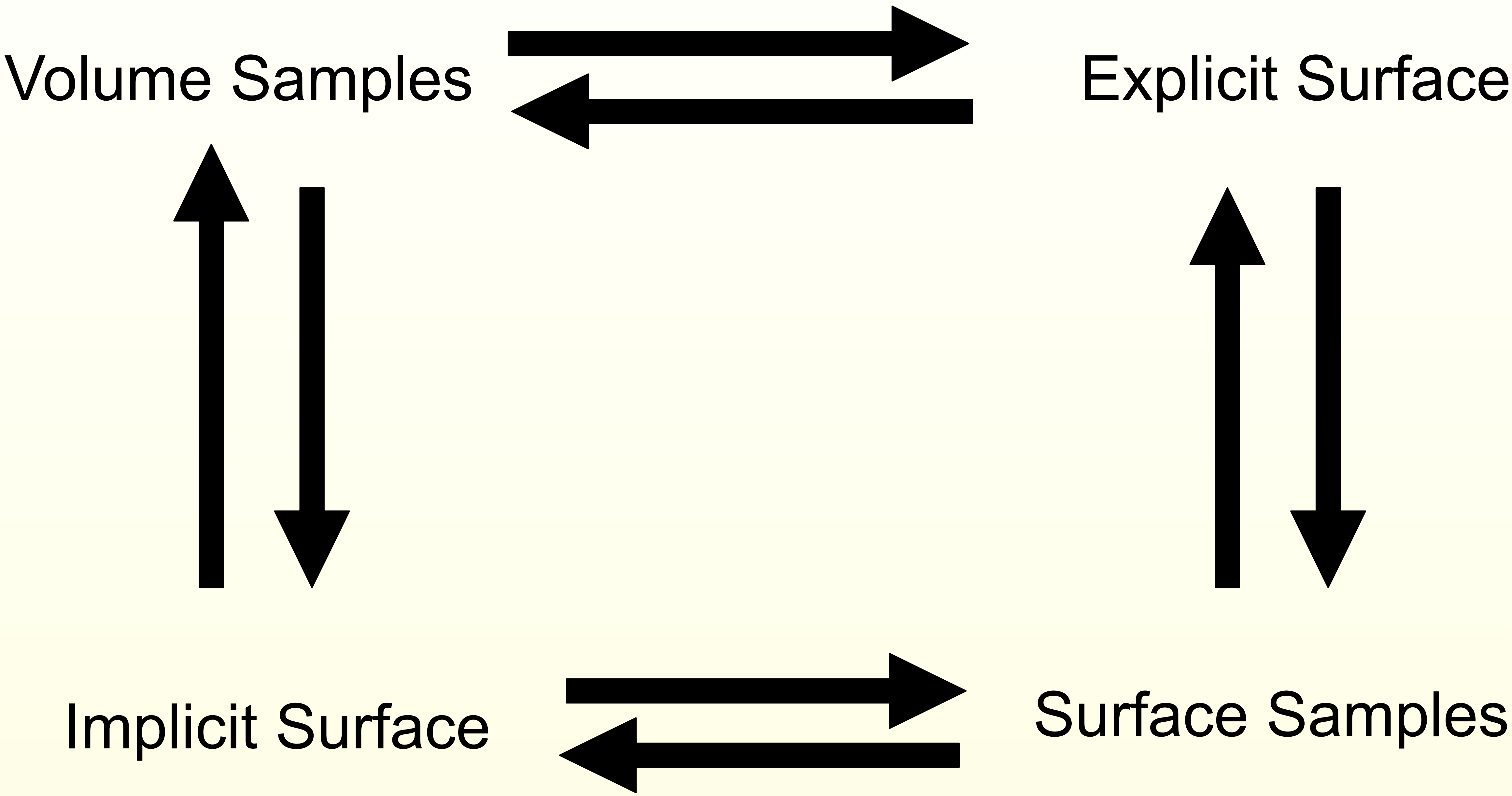
CS164: Surface Reconstruction, Marching Cubes

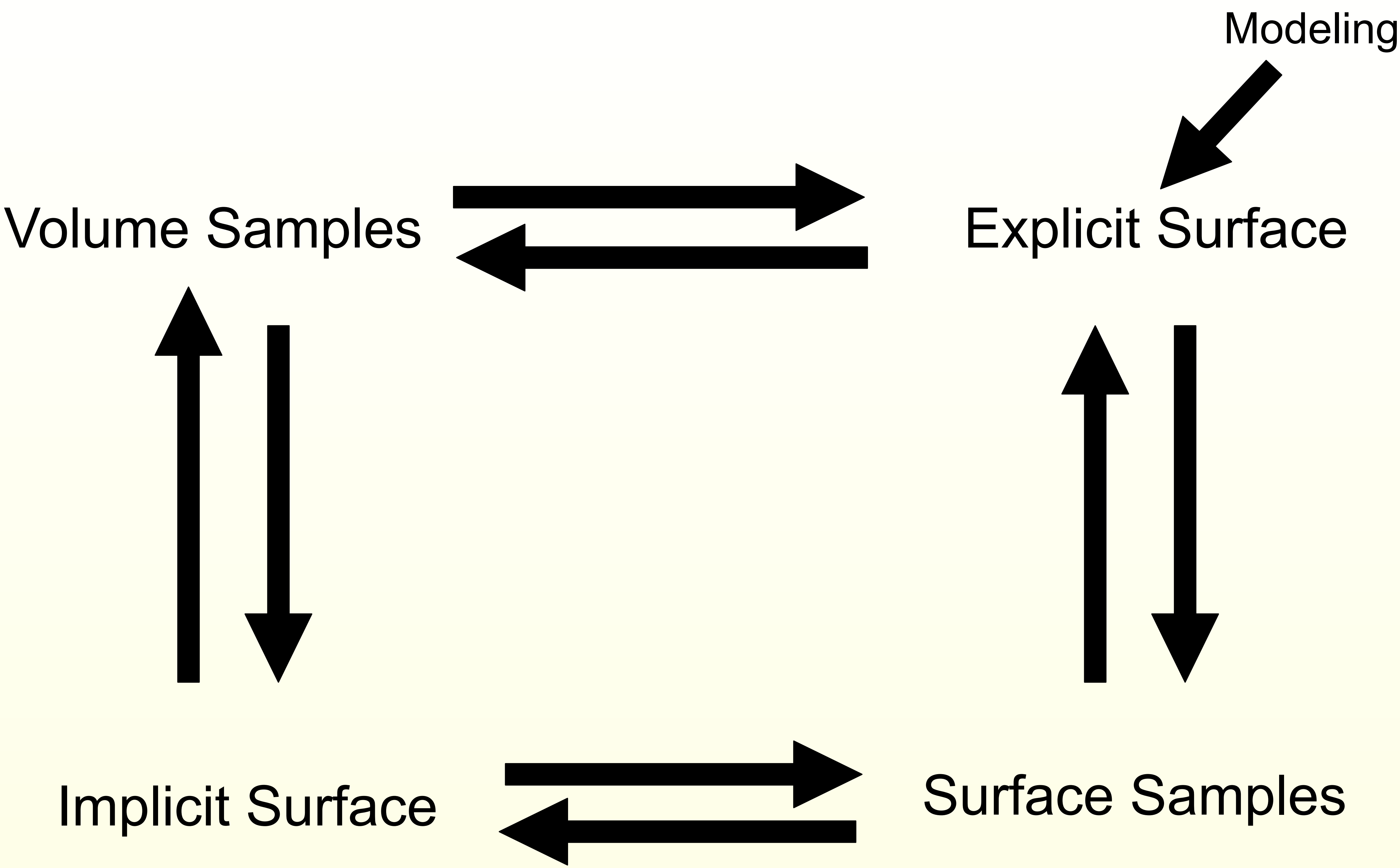
Leonidas Guibas
Computer Science Dept.
Stanford University

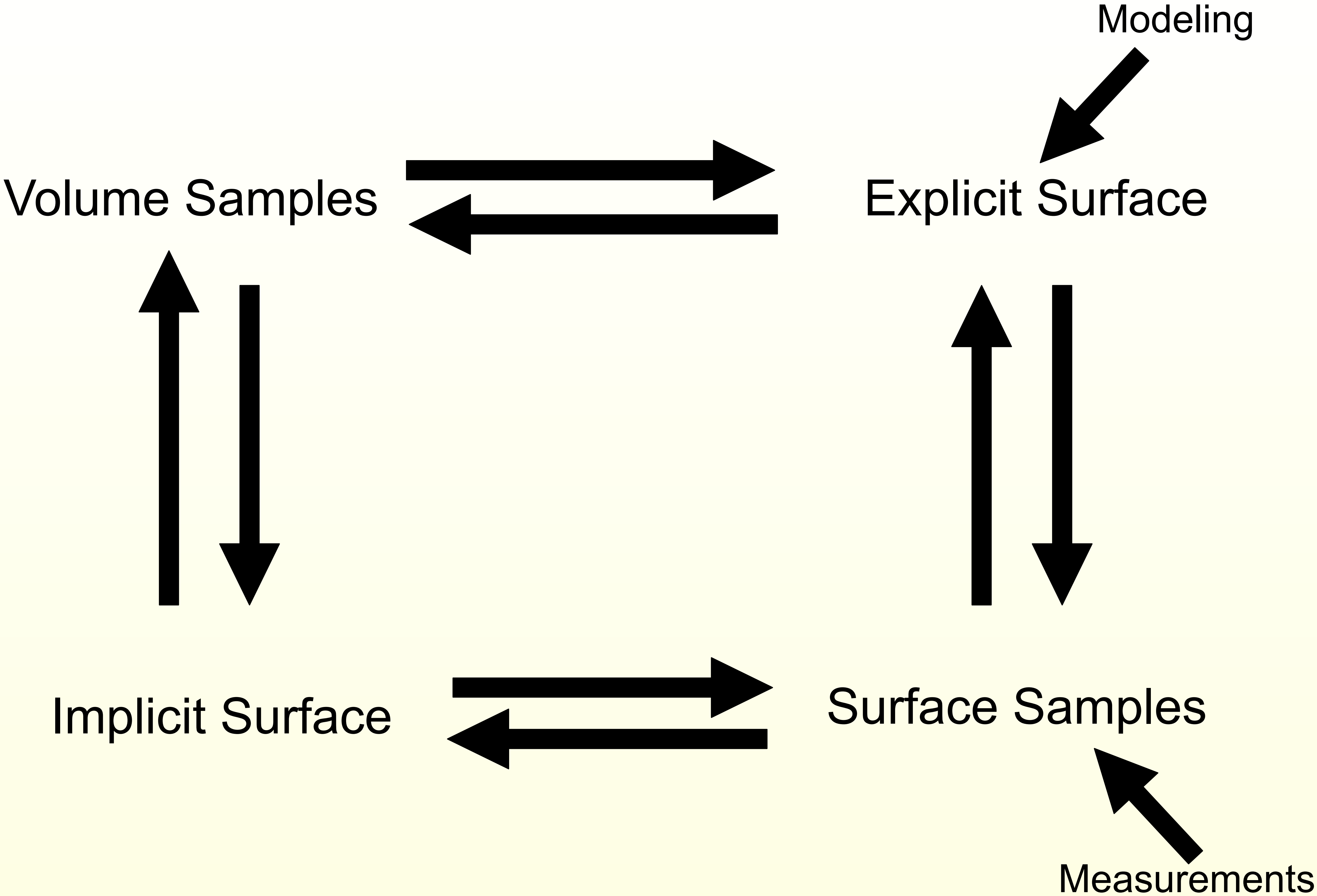
Overview

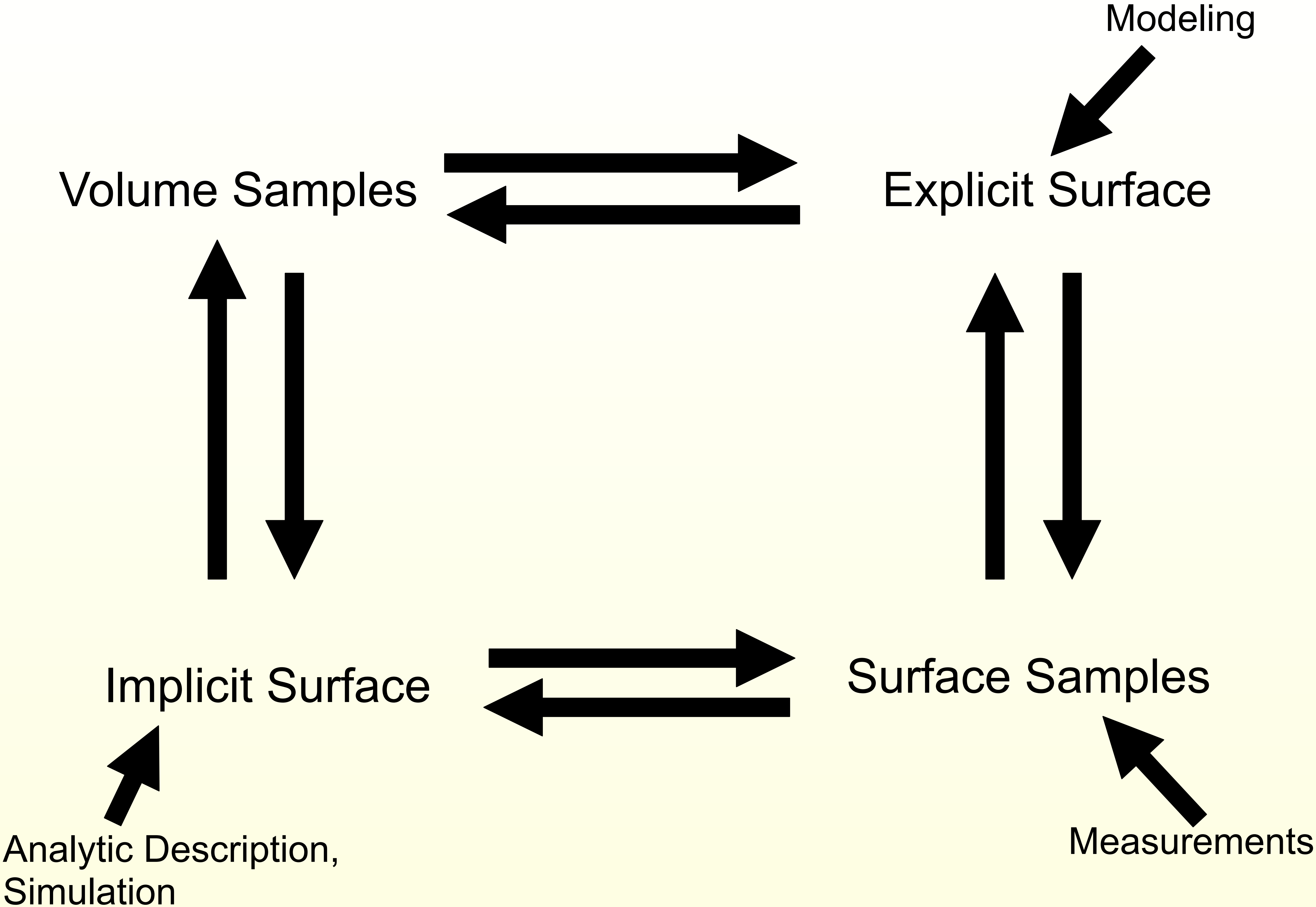
- Surface Representations
  - Explicit Surfaces
  - Implicit Surfaces
- Marching Cubes
  - Hermite Data/Extended Marching Cubes
  - Dual Contouring
  - Topological Guarantees

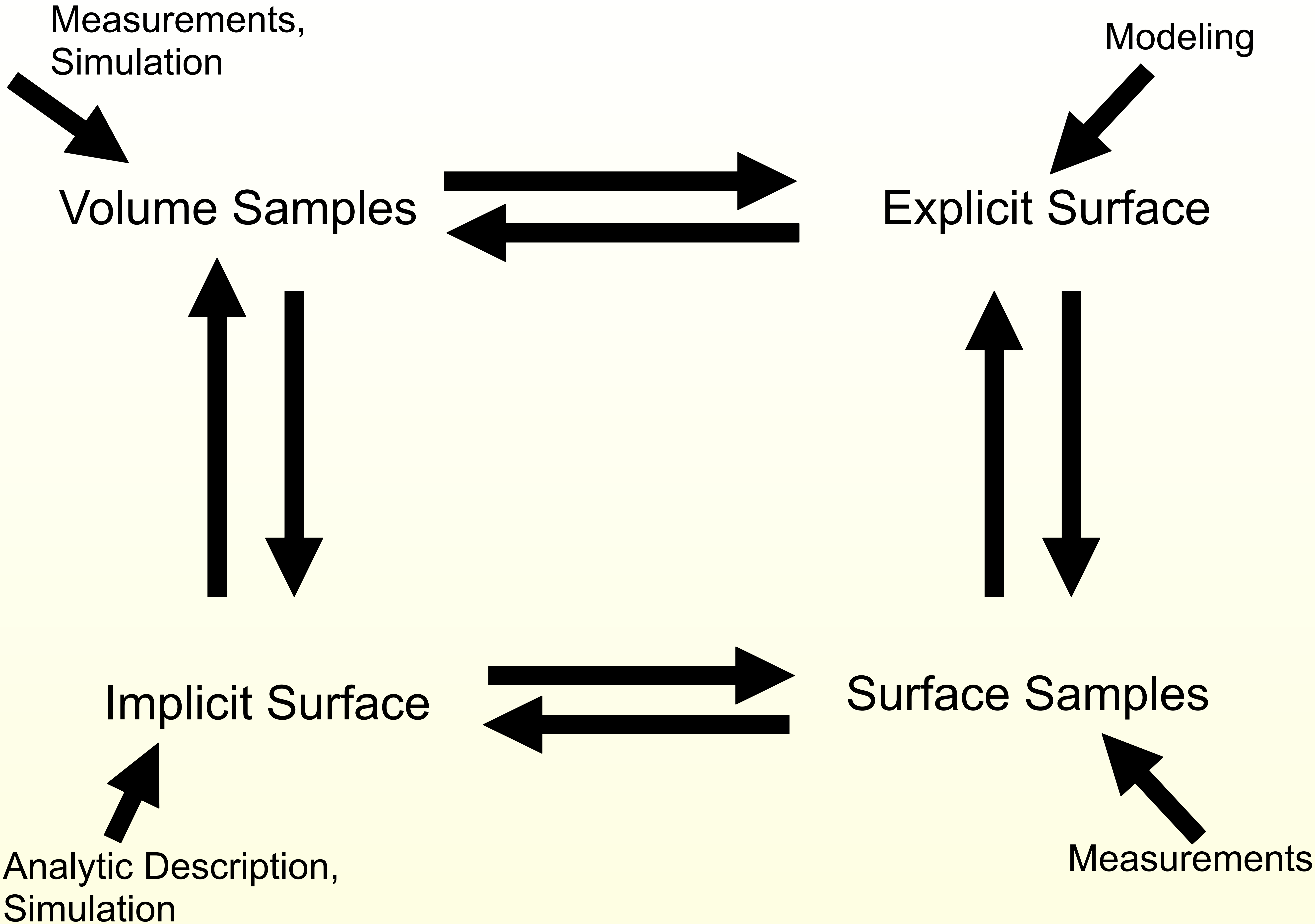
Surface Representations

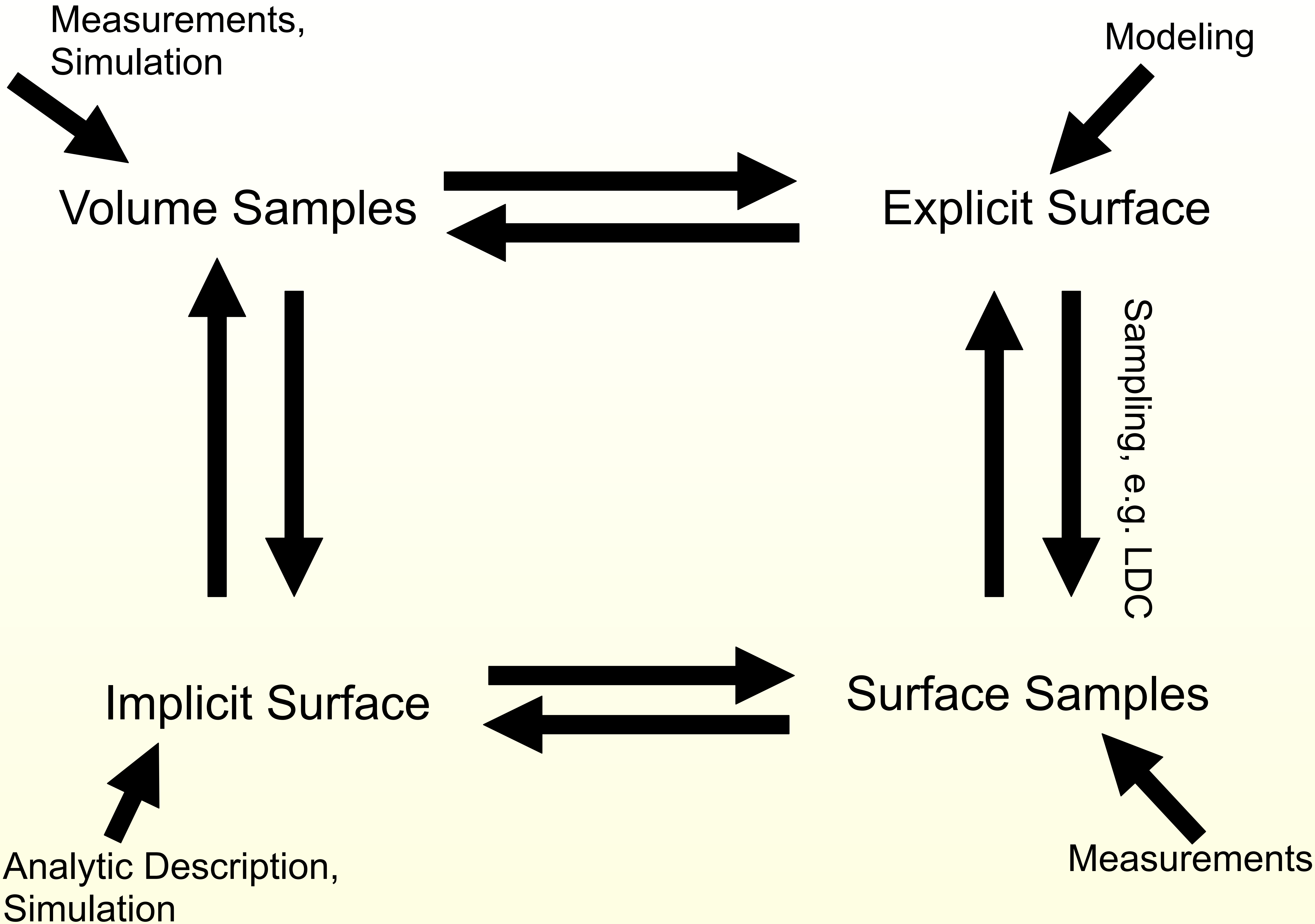

Volume Samples
Explicit Surface
Implicit Surface
Surface Samples


Modeling
Volume Samples
Explicit Surface
Implicit Surface
Surface Samples

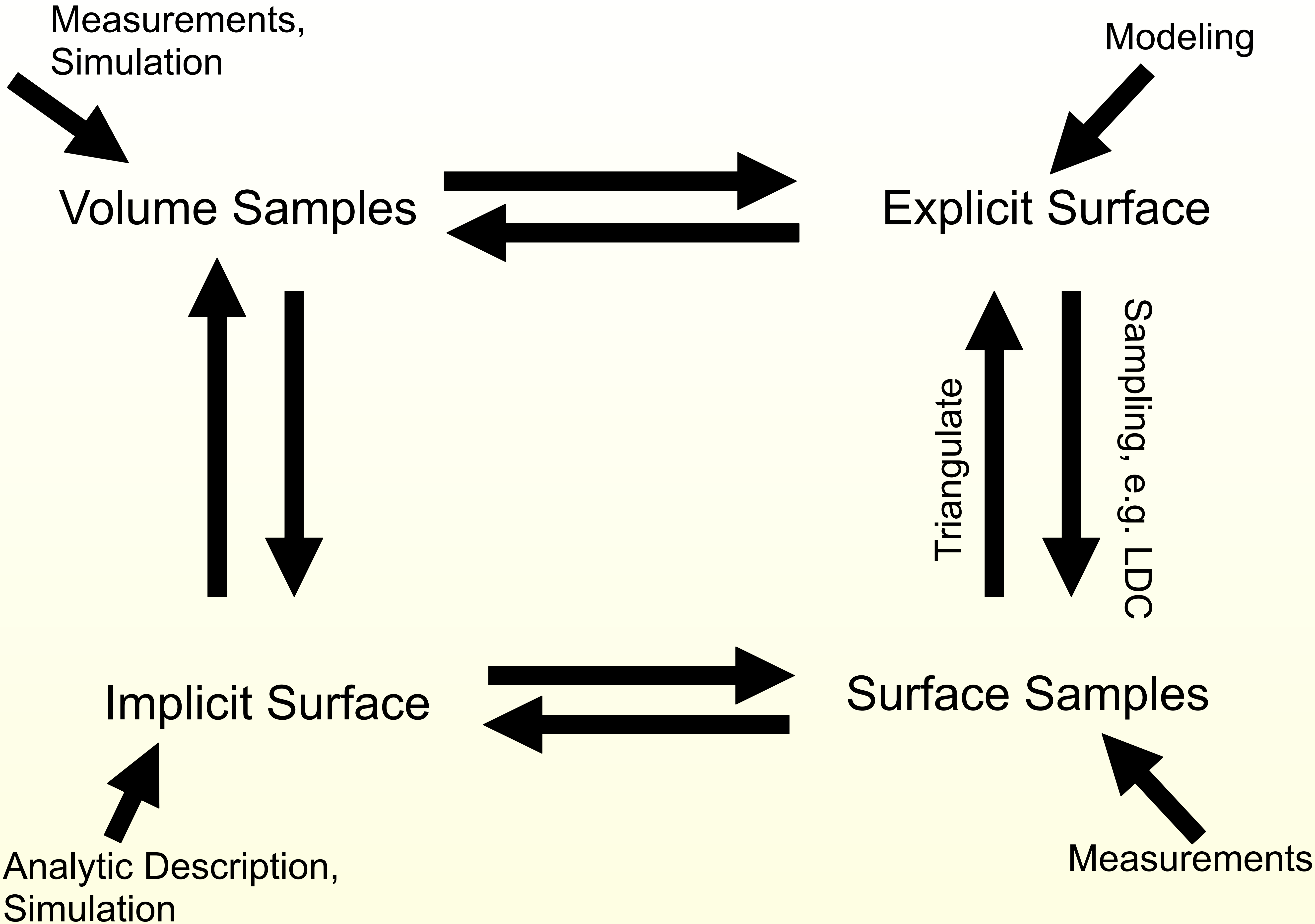

Modeling
Volume Samples
Explicit Surface
Implicit Surface
Surface Samples
Measurements

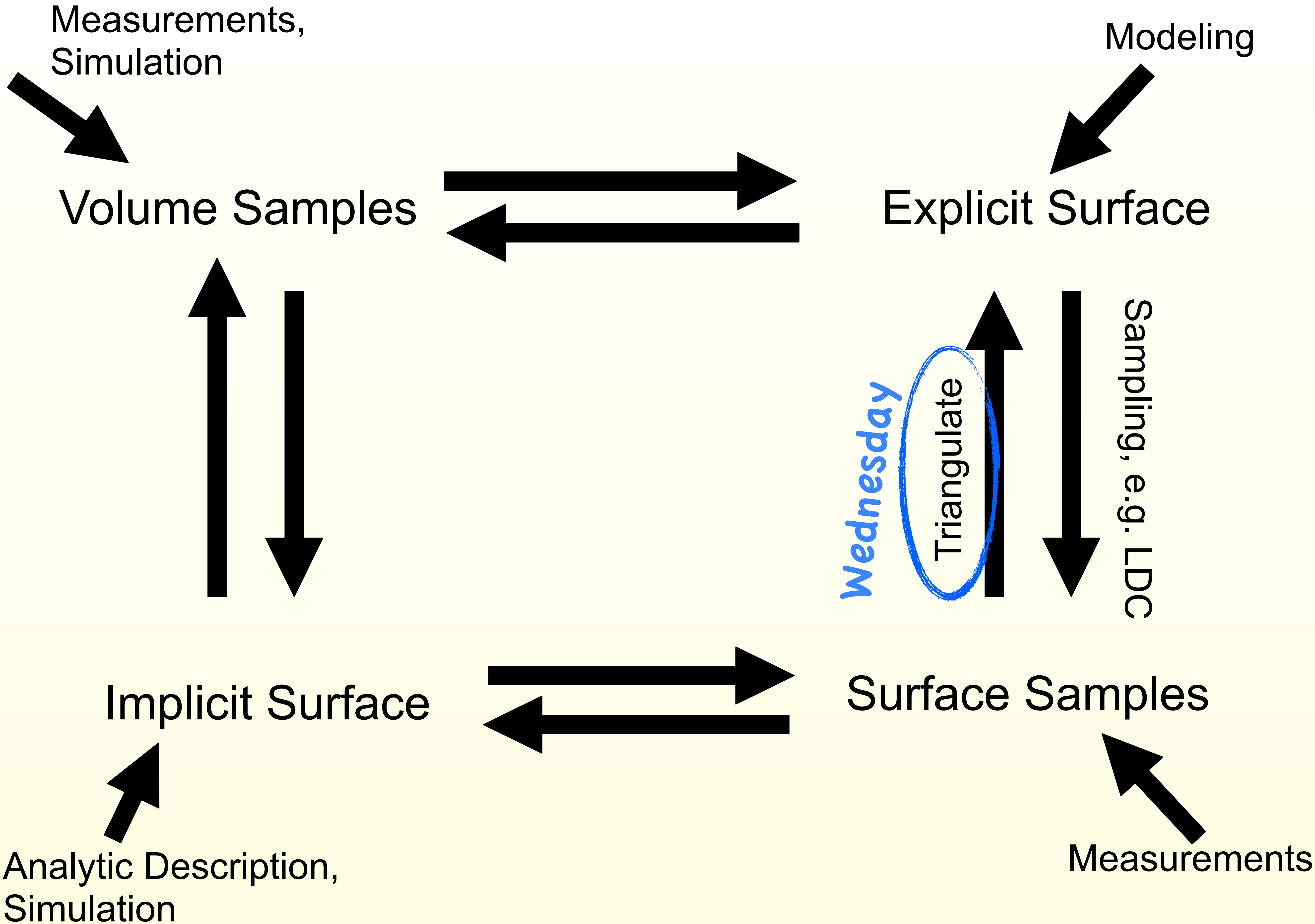

Modeling
Volume Samples
Explicit Surface
Implicit Surface
Surface Samples
Analytic Description,
Simulation
Measurements

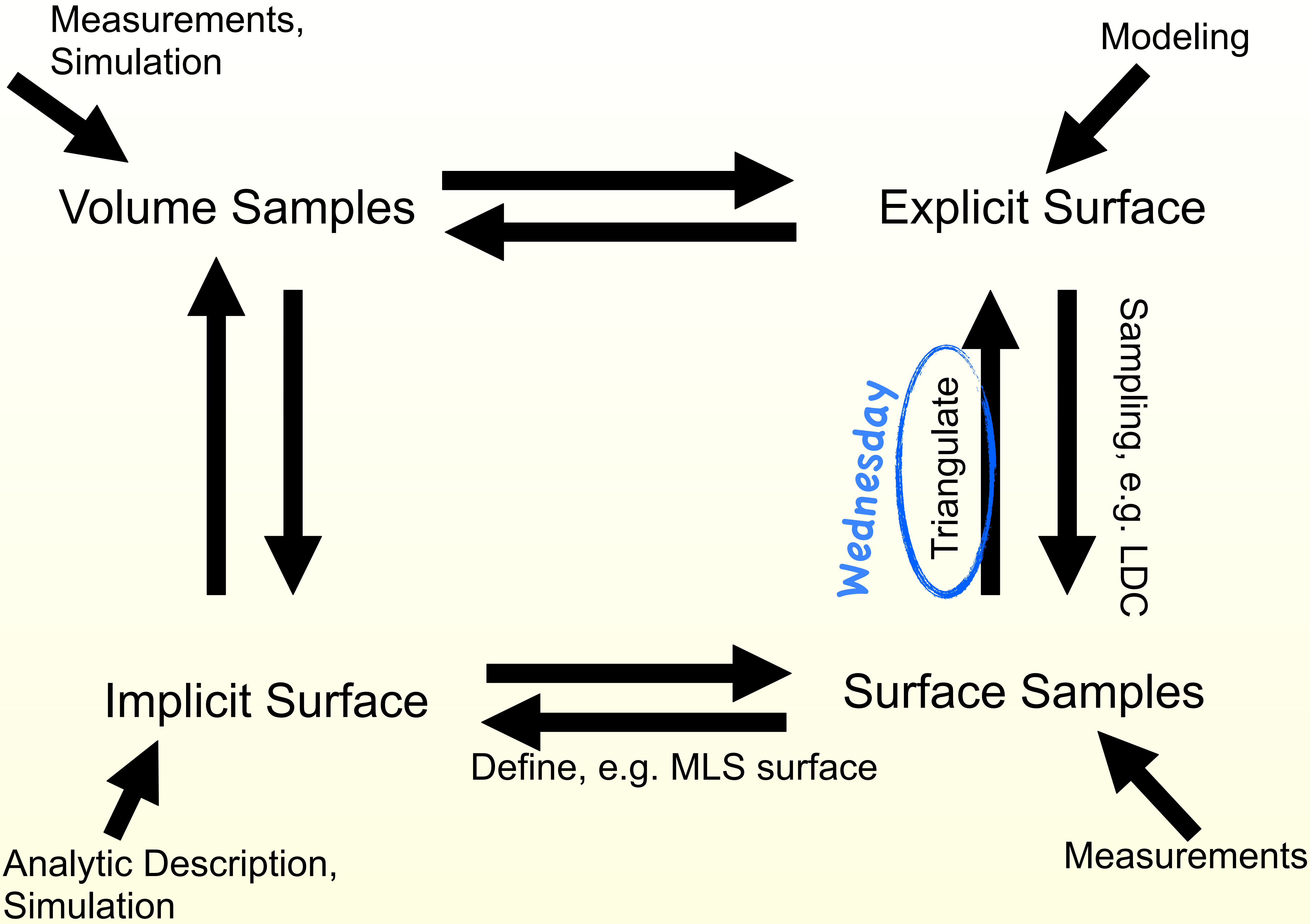

Measurements,
Simulation
Modeling
Volume Samples
Explicit Surface
Implicit Surface
Surface Samples
Analytic Description,
Simulation
Measurements

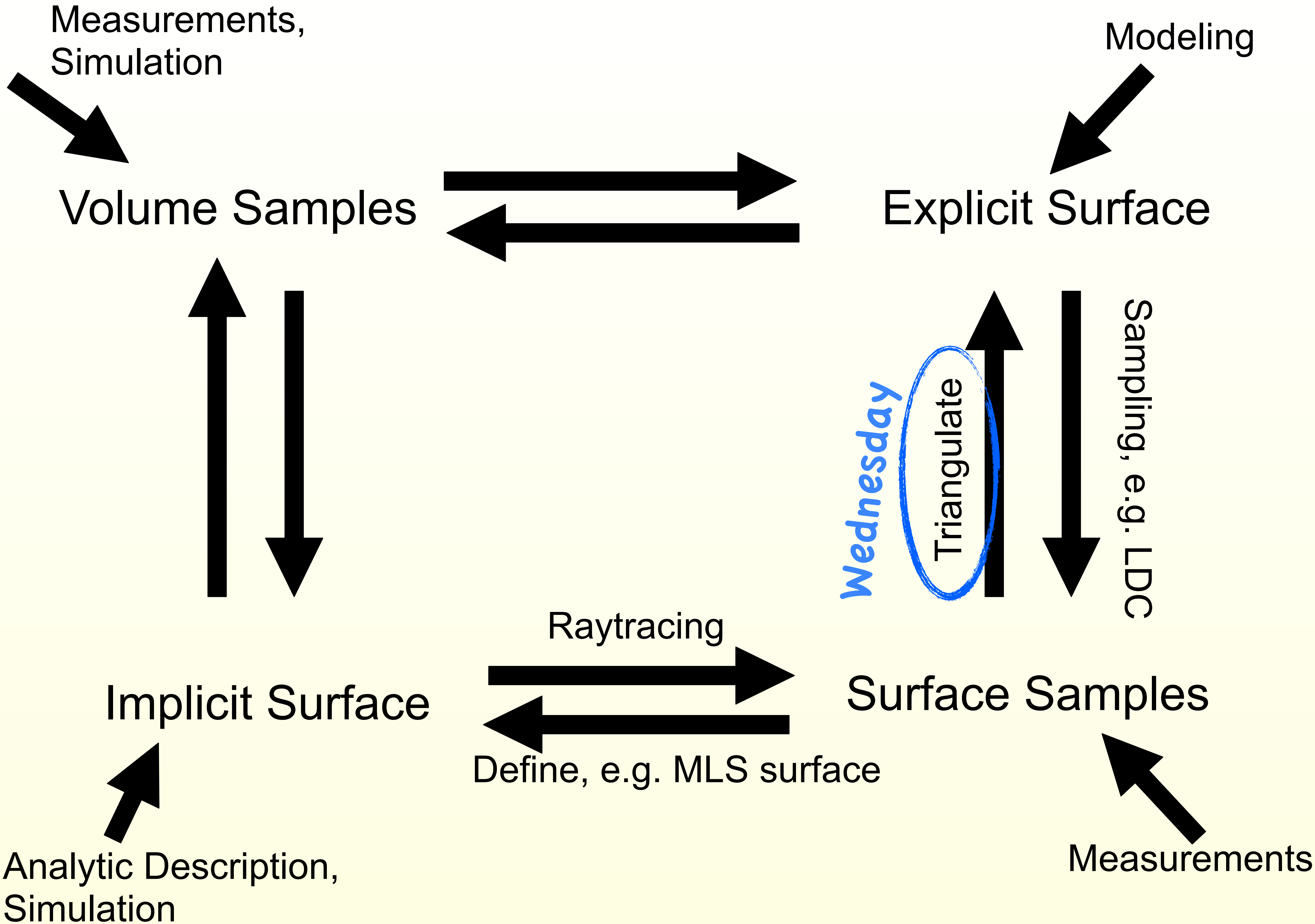


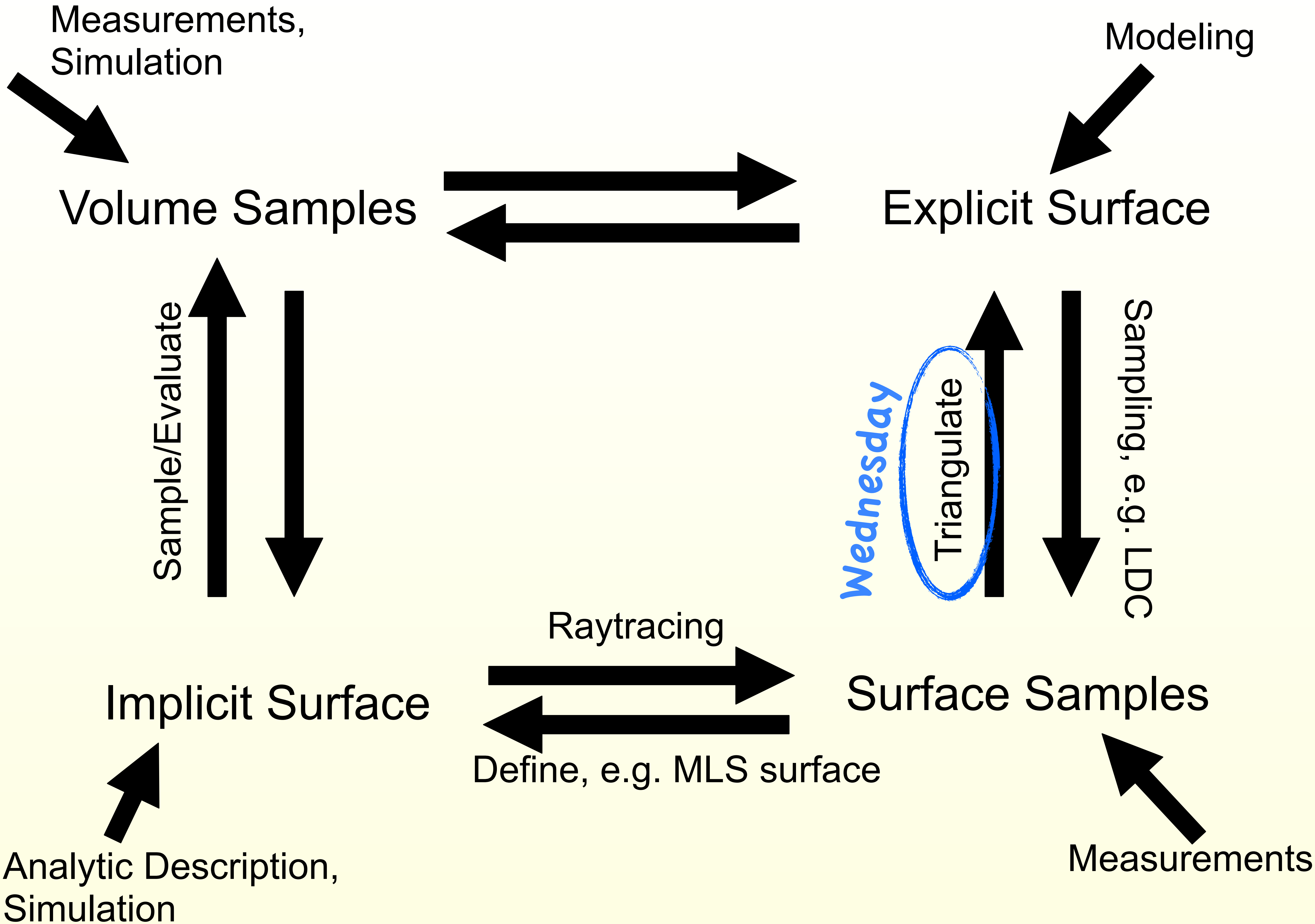
Measurements,
Simulation
Modeling
Volume Samples
Explicit Surface
Sampling, e.g. LDC
Implicit Surface
Surface Samples
Analytic Description,
Simulation
Measurements

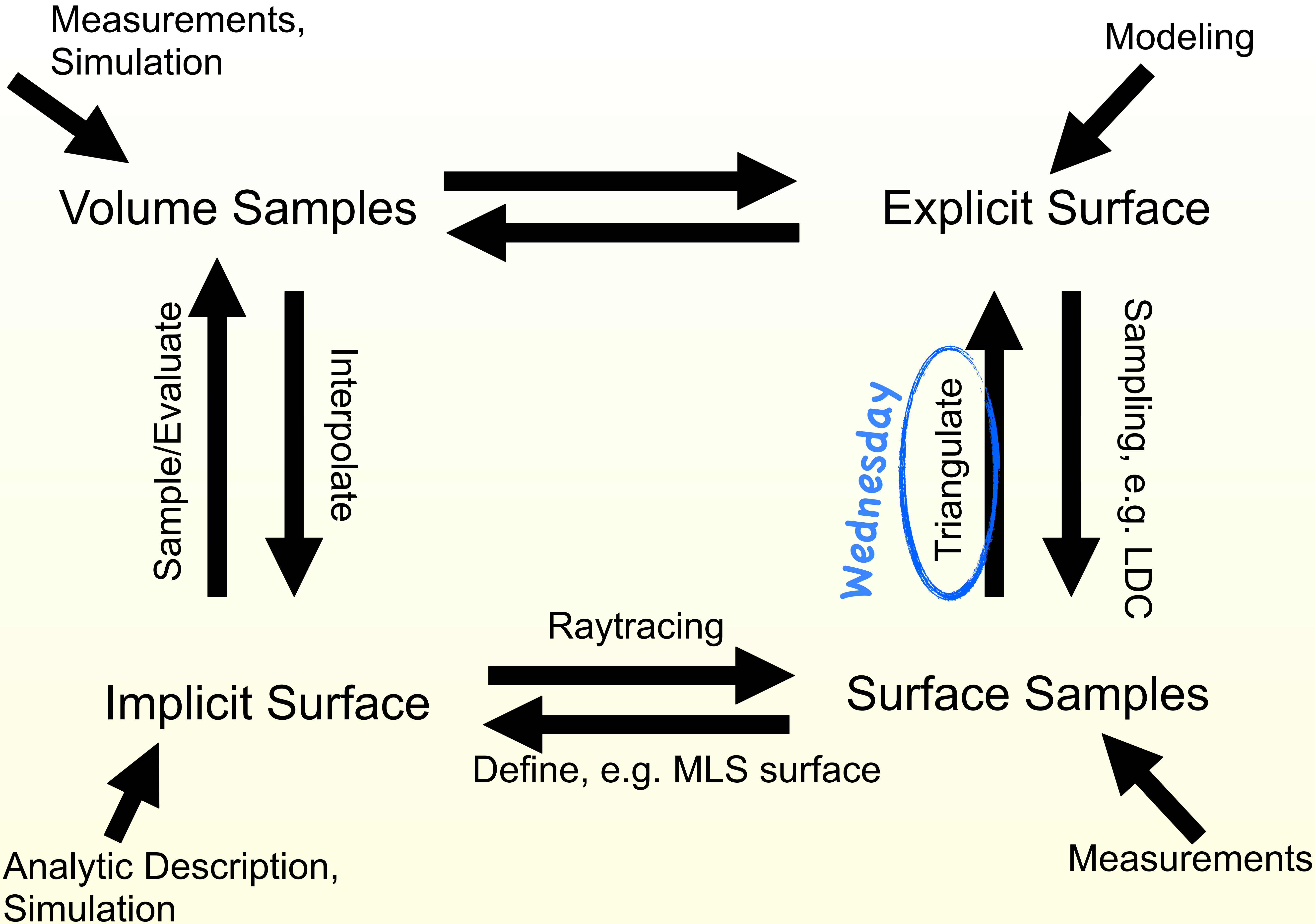

Measurements,
Simulation
Modeling
Volume Samples
Explicit Surface
Triangulate
Sampling, e.g. LDC
Implicit Surface
Surface Samples
Analytic Description,
Simulation
Measurements

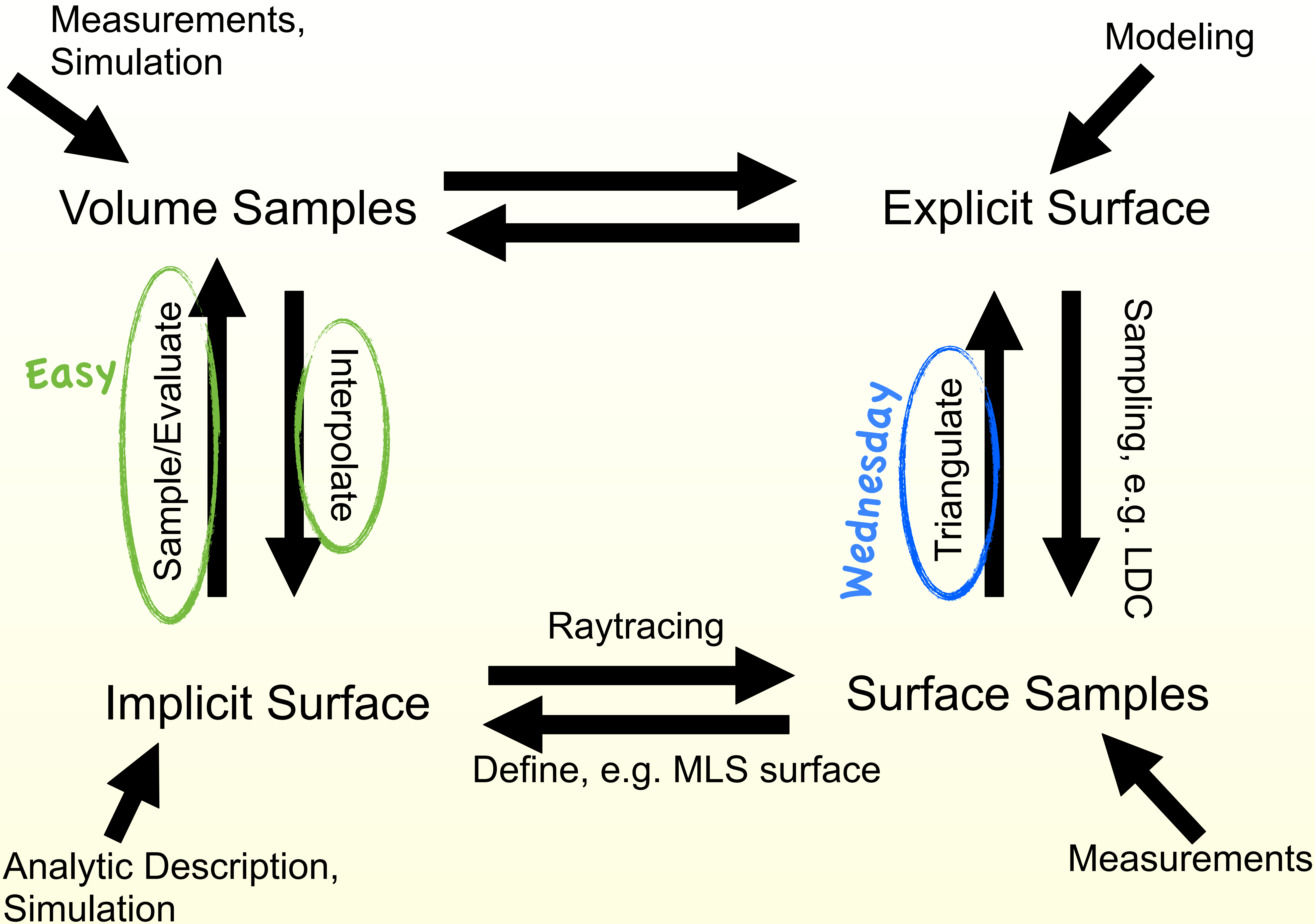

Measurements, Simulation
Modeling
Volume Samples
Explicit Surface
Wednesday
Triangulate
Sampling, e.g. LDC
Implicit Surface
Surface Samples
Analytic Description, Simulation
Measurements

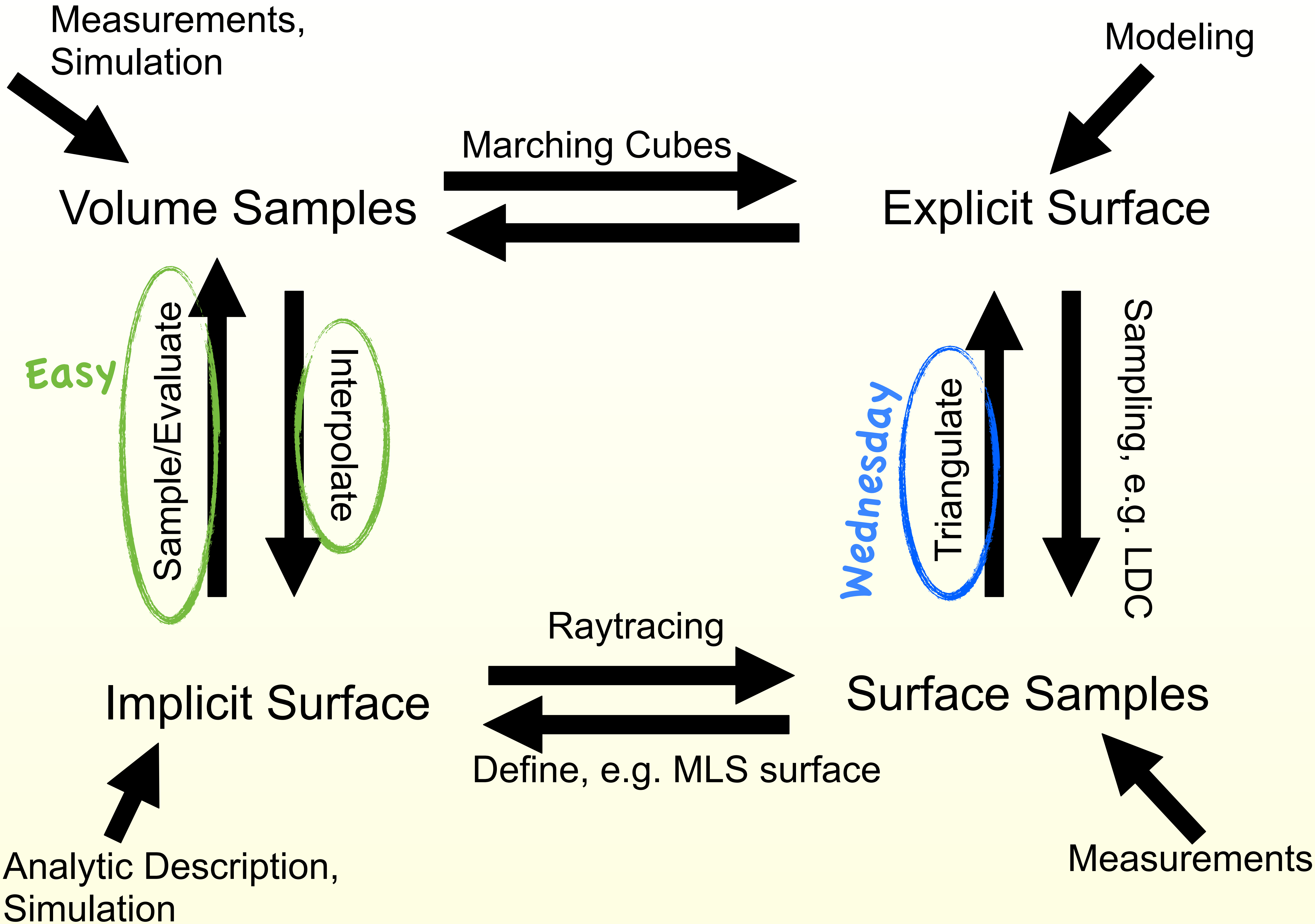

Measurements,
Simulation
Modeling
Volume Samples
Explicit Surface
Wednesday
Triangulate
Sampling, e.g. LDC
Implicit Surface
Surface Samples
Define, e.g. MLS surface
Analytic Description,
Simulation
Measurements

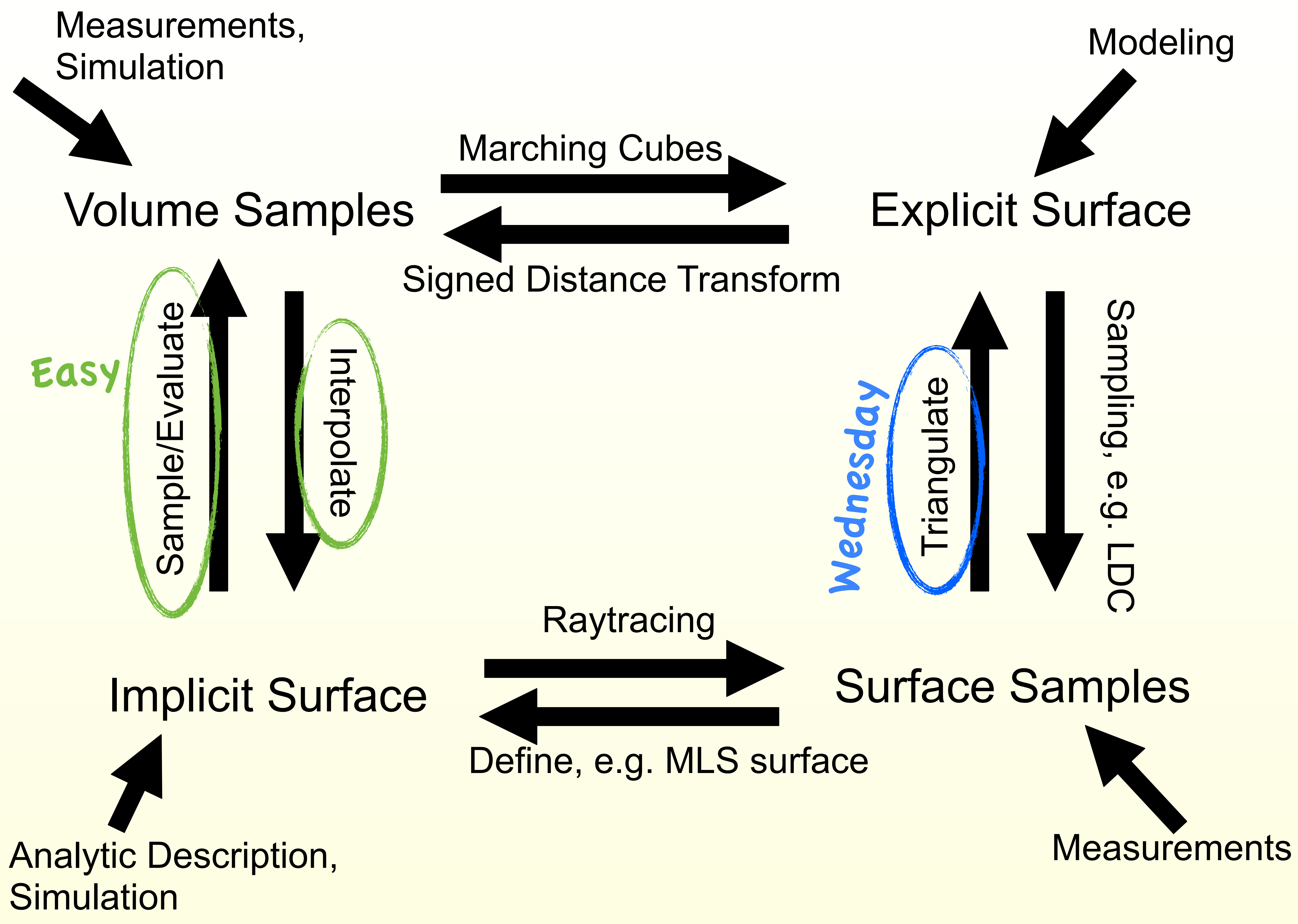

Measurements,
Simulation
Modeling
Volume Samples
Explicit Surface
Wednesday
Triangulate
Sampling, e.g. LDC
Raytracing
Implicit Surface
Surface Samples
Define, e.g. MLS surface
Analytic Description,
Simulation
Measurements


Measurements, Simulation
Modeling
Volume Samples
Explicit Surface
Sample/Evaluate
Wednesday
Triangulate
Sampling, e.g. LDC
Raytracing
Implicit Surface
Surface Samples
Define, e.g. MLS surface
Analytic Description, Simulation
Measurements

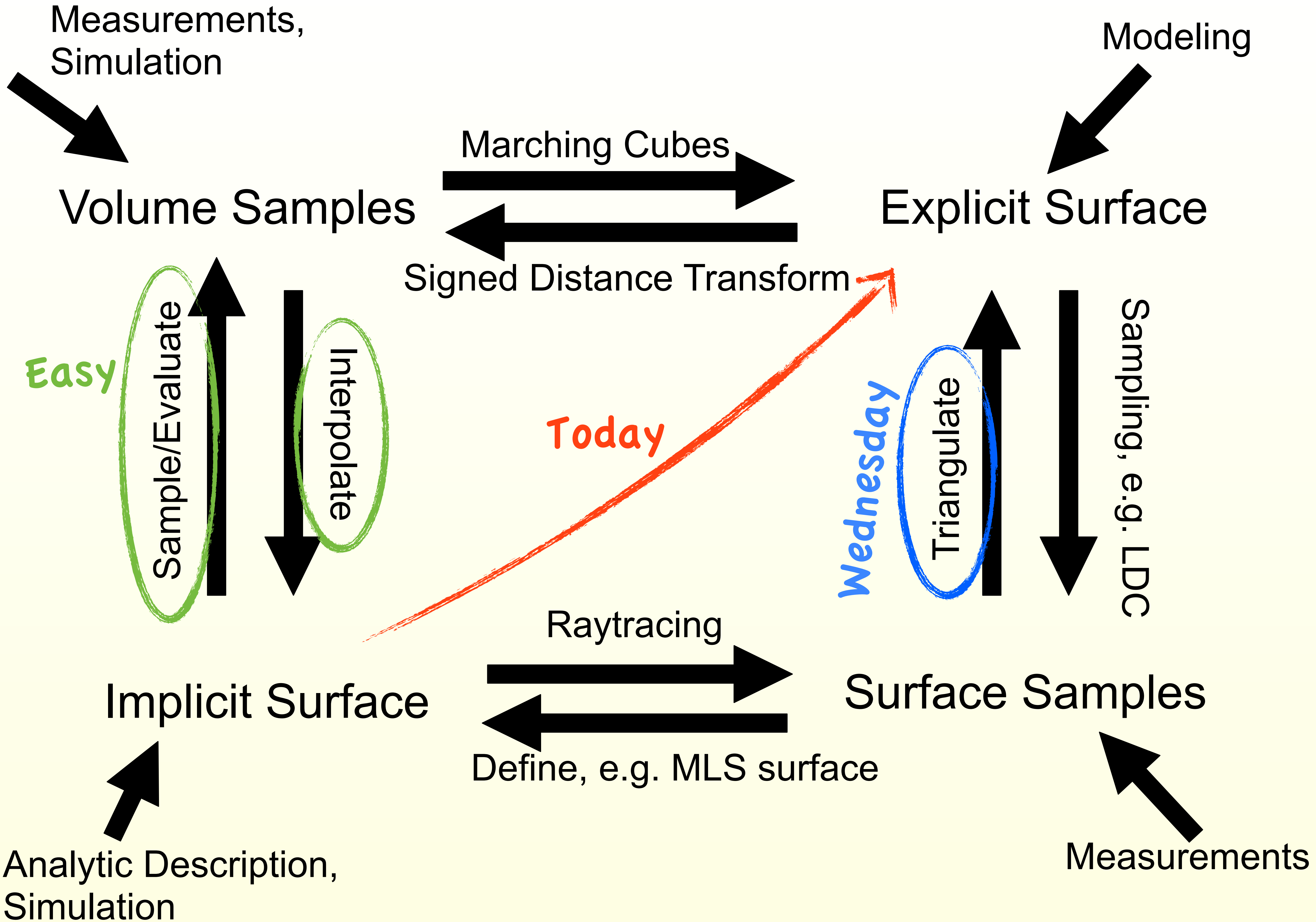

Measurements,
Simulation
Modeling
Volume Samples
Explicit Surface
Sample/Evaluate
Interpolate
Wednesday
Triangulate
Sampling, e.g. LDC
Raytracing
Implicit Surface
Surface Samples
Define, e.g. MLS surface
Analytic Description,
Simulation
Measurements


Measurements,
Simulation
Modeling
Volume Samples
Explicit Surface
Easy
Sample/Evaluate
Interpolate
Wednesday
Triangulate
Sampling, e.g. LDC
Raytracing
Implicit Surface
Surface Samples
Define, e.g. MLS surface
Analytic Description,
Simulation
Measurements

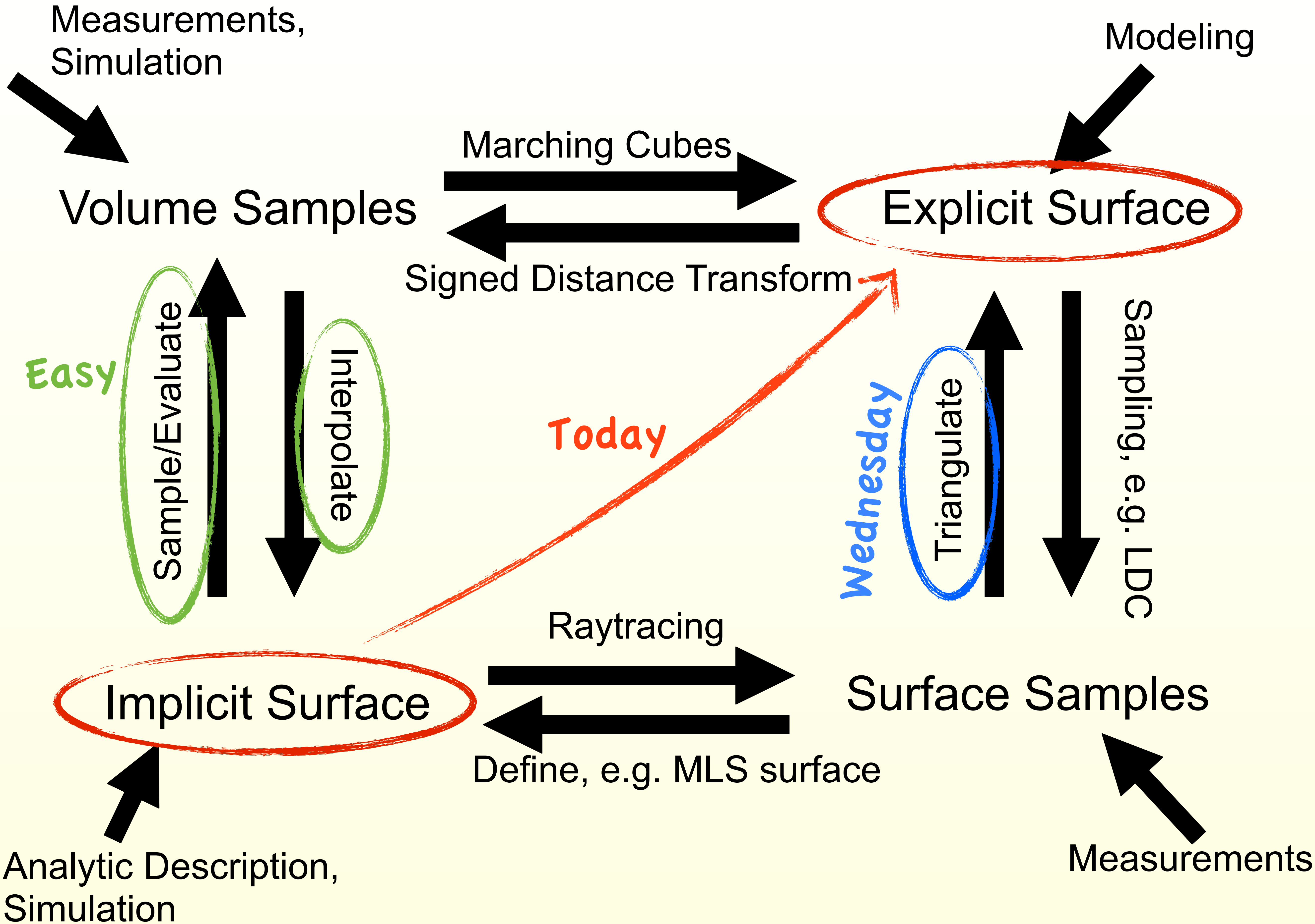

Measurements,
Simulation
Modeling
Marching Cubes
Volume Samples
Explicit Surface
Easy
Sample/Evaluate
Interpolate
Wednesday
Triangulate
Sampling, e.g. LDC
Raytracing
Implicit Surface
Surface Samples
Define, e.g. MLS surface
Analytic Description,
Simulation
Measurements

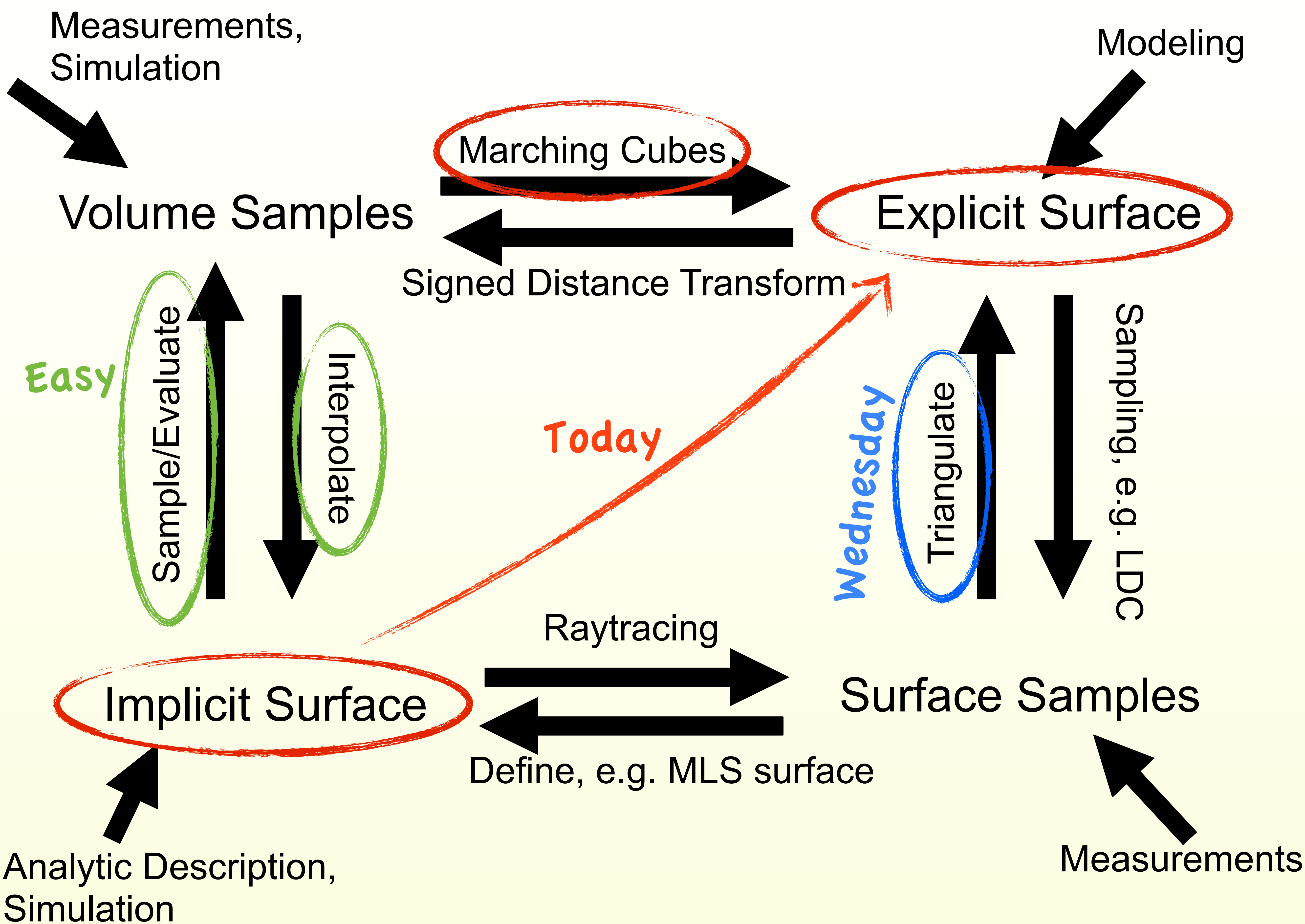


Measurements, Simulation
Modeling
Marching Cubes
Volume Samples
Explicit Surface
Signed Distance Transform
Easy
Sample/Evaluate
Interpolate
Wednesday
Triangulate
Sampling, e.g. LDC
Raytracing
Implicit Surface
Surface Samples
Define, e.g. MLS surface
Analytic Description, Simulation
Measurements



Measurements,
Simulation
Modeling
Marching Cubes
Volume Samples
Explicit Surface
Signed Distance Transform
Easy
Sample/Evaluate
Interpolate
Today
Wednesday
Triangulate
Sampling, e.g. LDC
Raytracing
Implicit Surface
Surface Samples
Define, e.g. MLS surface
Analytic Description,
Simulation
Measurements


Measurements,
Simulation
Modeling
Marching Cubes
Volume Samples
Explicit Surface
Signed Distance Transform
Easy
Sample/Evaluate
Interpolate
Today
Wednesday
Triangulate
Sampling, e.g. LDC
Raytracing
Implicit Surface
Surface Samples
Define, e.g. MLS surface
Analytic Description,
Simulation
Measurements



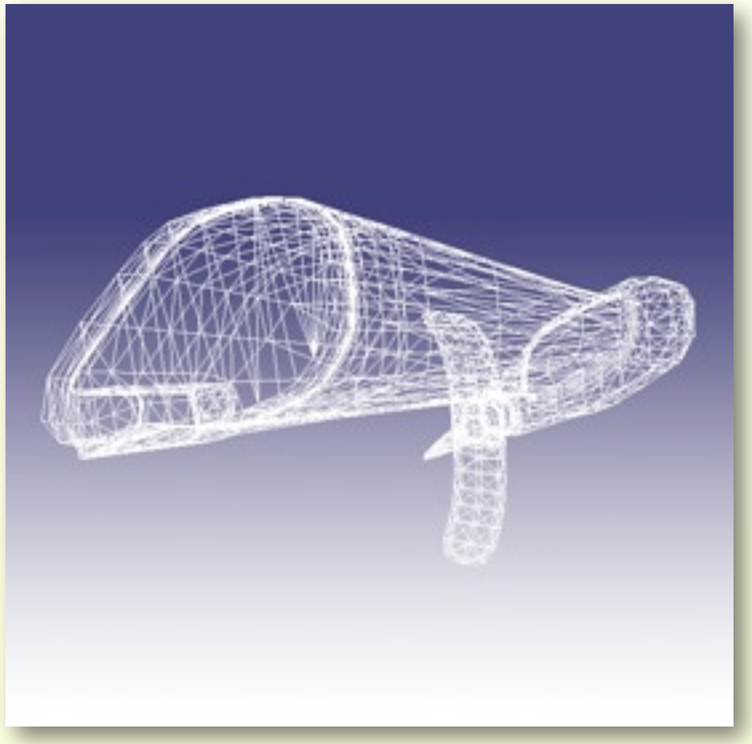
Measurements, Simulation
Modeling
Marching Cubes
Volume Samples
Explicit Surface
Signed Distance Transform
Easy
Sample/Evaluate
Interpolate
Today
Wednesday
Triangulate
Sampling, e.g. LDC
Raytracing
Implicit Surface
Surface Samples
Define, e.g. MLS surface
Analytic Description, Simulation
Measurements

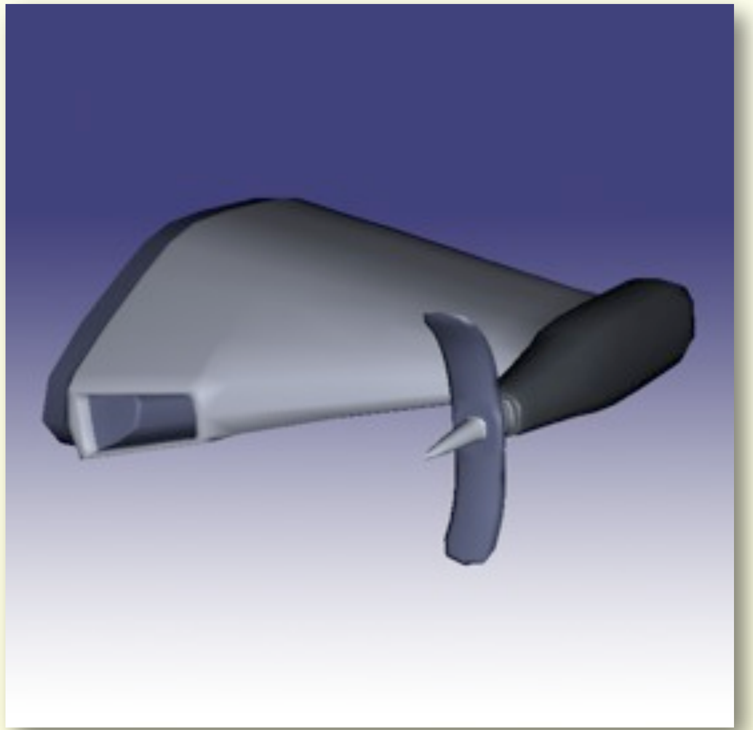
Explicit (Parametric) Surfaces

- "The surface consists of these points: ..."

$$\{f(\mathbf{u})|\mathbf{u} \in \mathbb{R}^2\}$$

- Splines (treated earlier)
- Piecewise-linear surfaces (polygonal meshes)
- Most common: triangle meshes

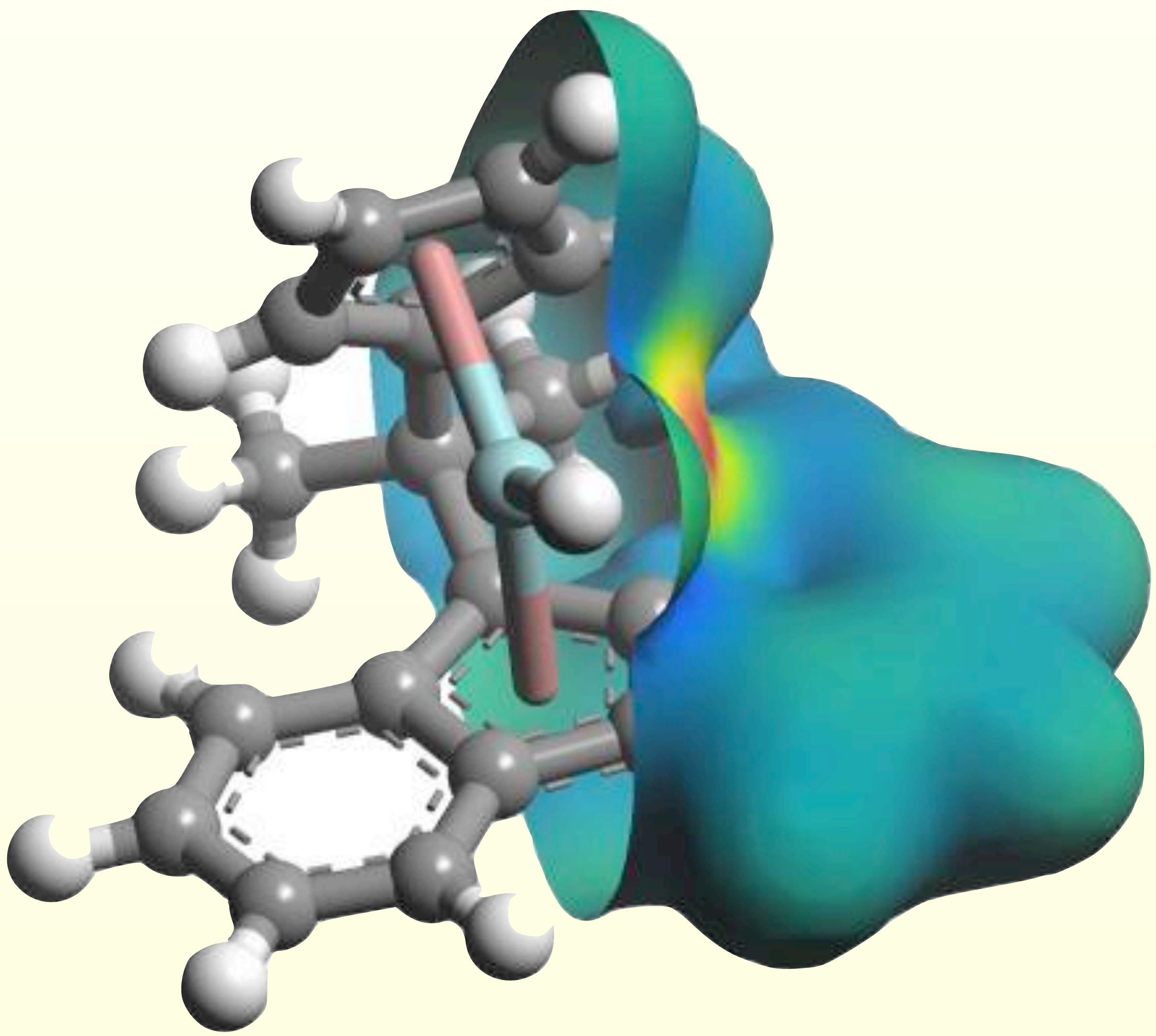




Implicit Surfaces

- "The surface consists of all points, which..."

$$\{\mathbf{x} | f(\mathbf{x}) = 0\}$$



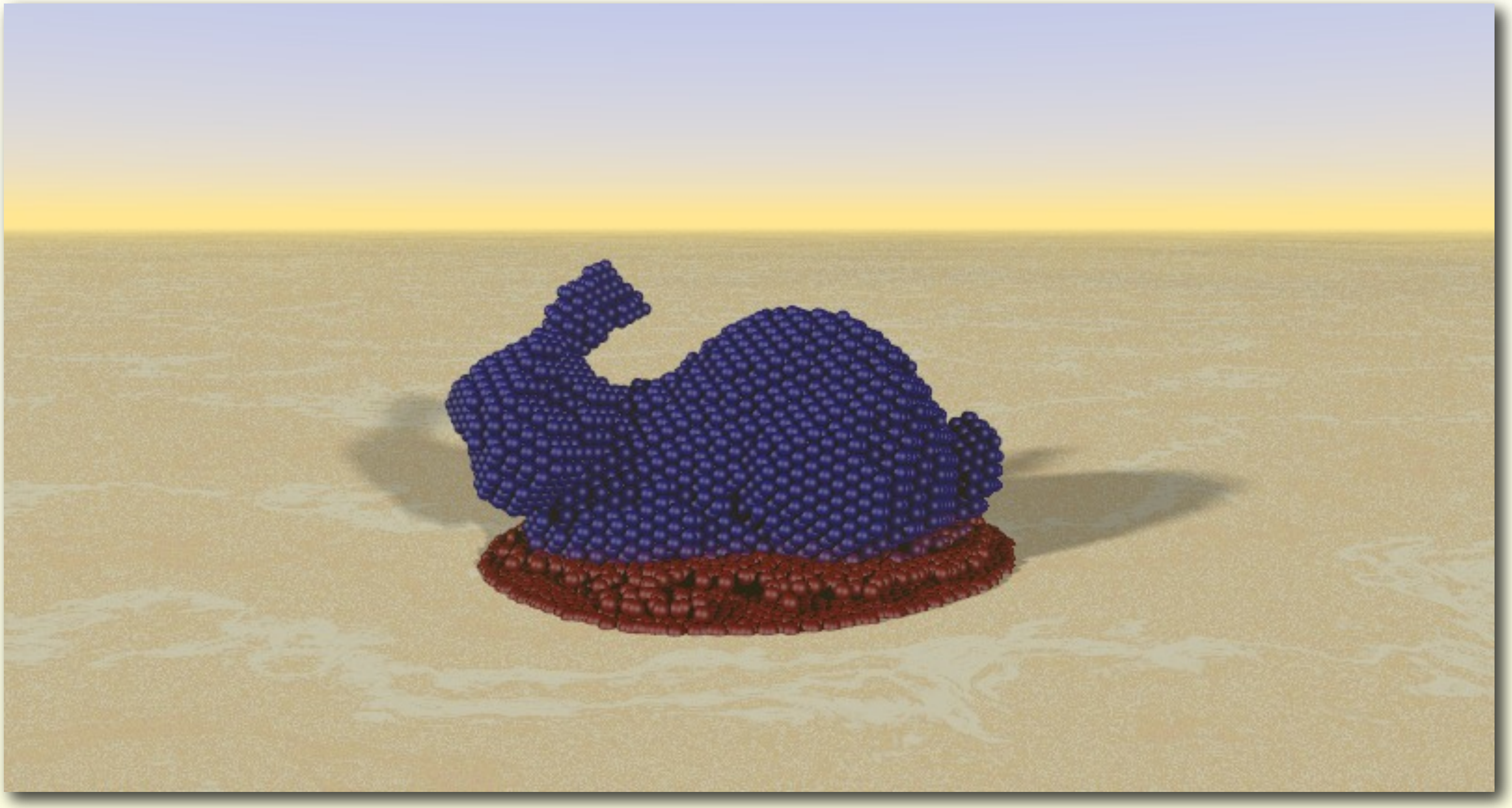
Isosurface around Zirconocene molecule [Accelrys]

Implicit vs. Explicit

- Different sources
- Explicit: $\{f(\mathbf{u})|\mathbf{u} \in \mathbb{R}^2\}$
  - Image of a function
  - Easy to enumerate points
  - Hard to check whether a given point is on the surface
- Implicit: $\{\mathbf{x}|f(\mathbf{x}) = 0\}$
  - Kernel of a function
  - Hard to enumerate points
  - Easy to check whether a given point is on the surface

Iso-Surface of a Density Field

- Example: $f(\mathbf{x}) = \sum_i w(\mathbf{x}, \mathbf{x}_i) - \rho_0$

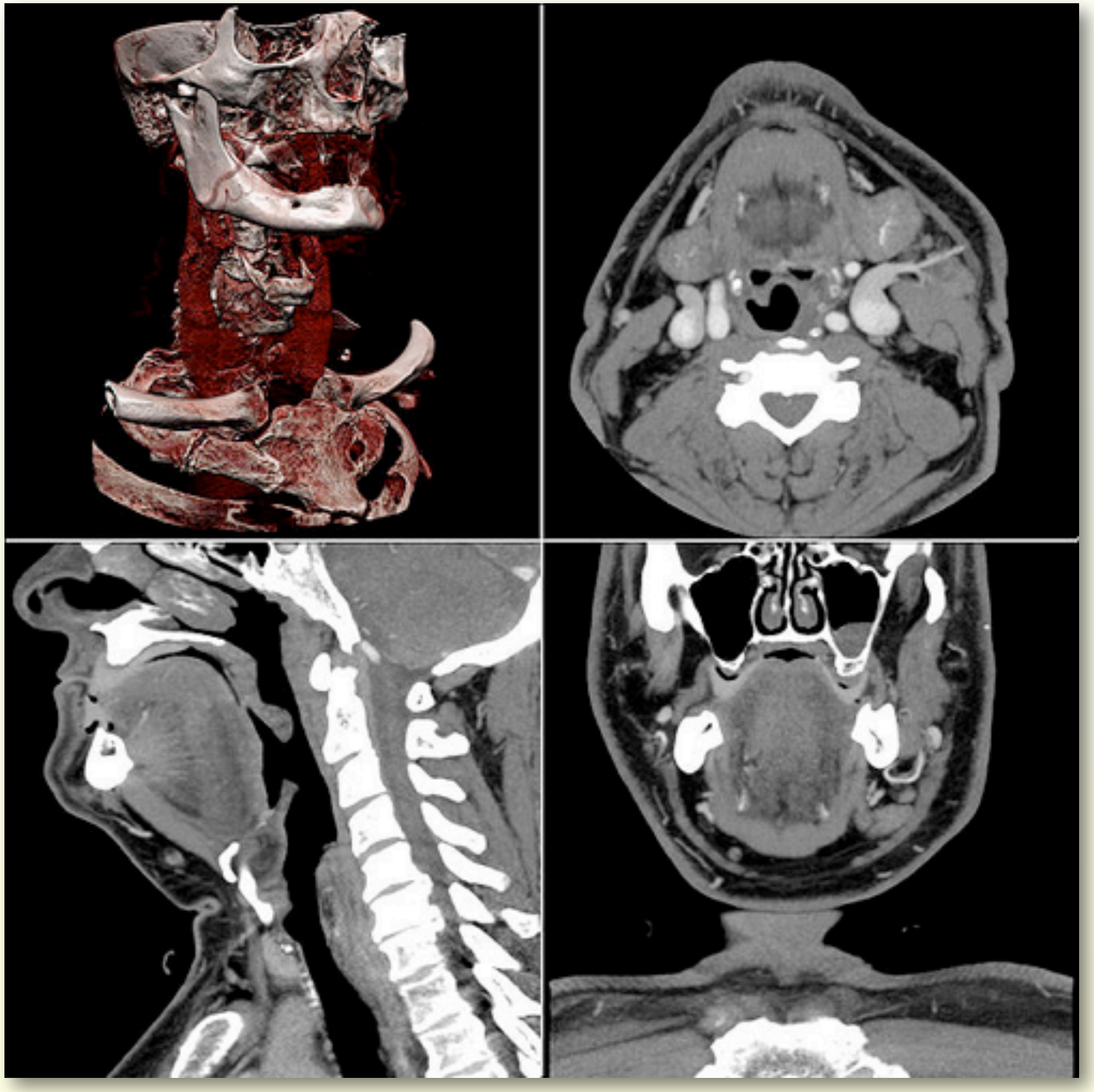


Iso-Surface of a Density Field

- Example: $f(\mathbf{x}) = \sum_i w(\mathbf{x}, \mathbf{x}_i) - \rho_0$

Iso-Surface in a CAT Scan

- $f(\mathbf{x}) = I(\mathbf{x}) - q$
- Samples in a regular grid
- Trilinear within voxels
- How can we make an isosurface explicit?



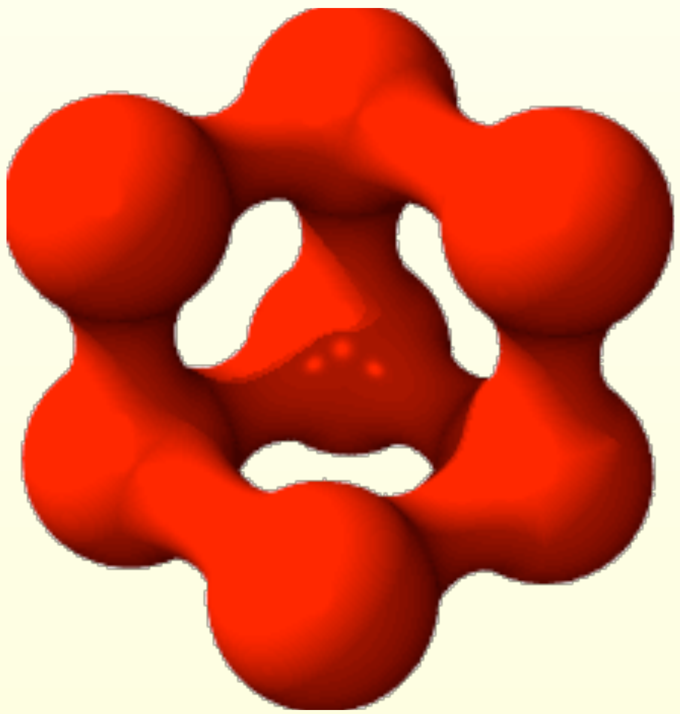
Marching Cubes
(and variants)

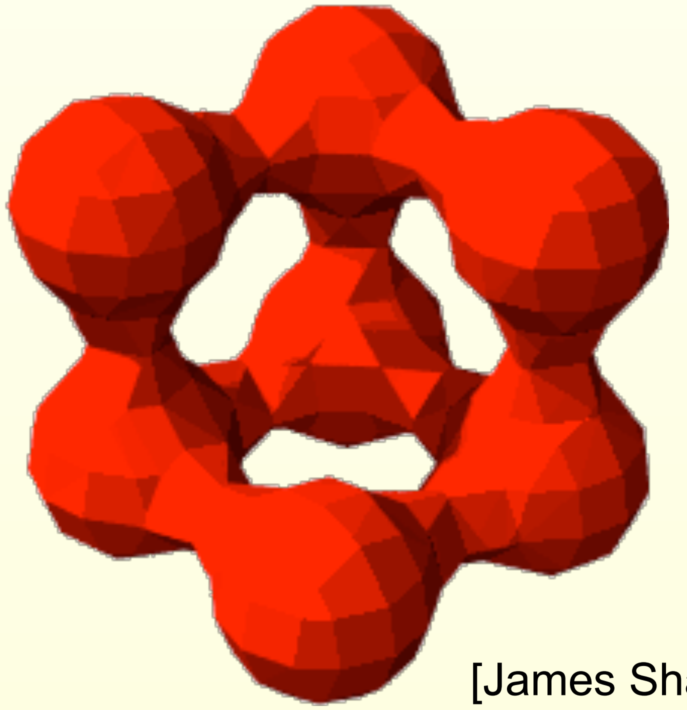
Problem Statement

Given a function $I(\mathbf{x})$ defining an implicit surface

$$\mathcal{S} = \{\mathbf{x} | f(\mathbf{x}) = I(\mathbf{x}) - q = 0\},$$

create a triangle mesh that approximates the surface S.





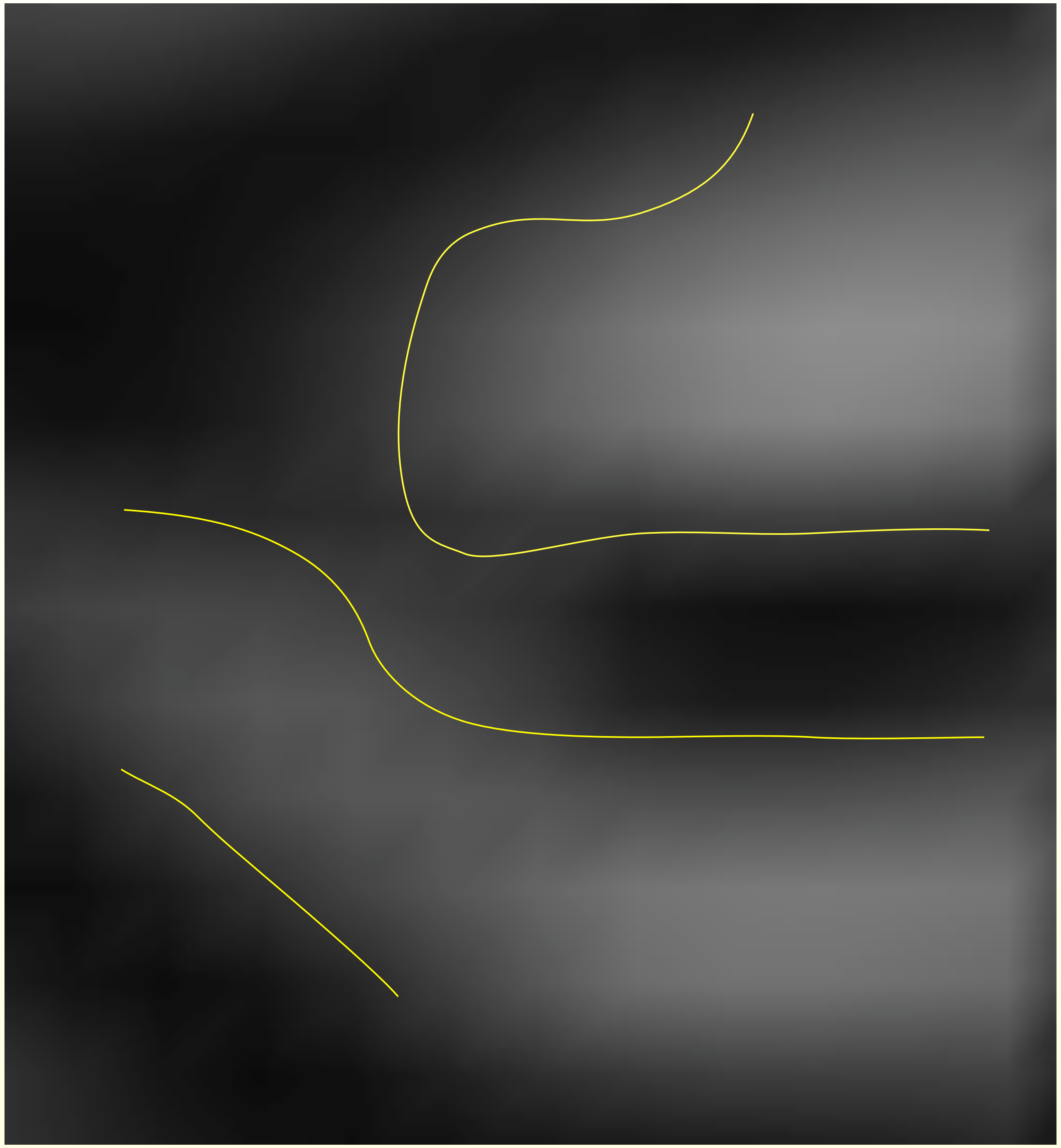
[James Sharman]

Overview

- Marching Cubes
  - 2D case: Marching Squares
  - 3D case: Marching Cubes
  - Marching Tetrahedra
- Extended Marching Cubes
- Dual Contouring

Marching Squares

- Given $I(\mathbf{x})$
  - $I(\mathbf{x}) < q$: $\mathbf{x}$ outside
  - $I(\mathbf{x}) > q$: $\mathbf{x}$ inside
- Discretize space
- Evaluate on the grid

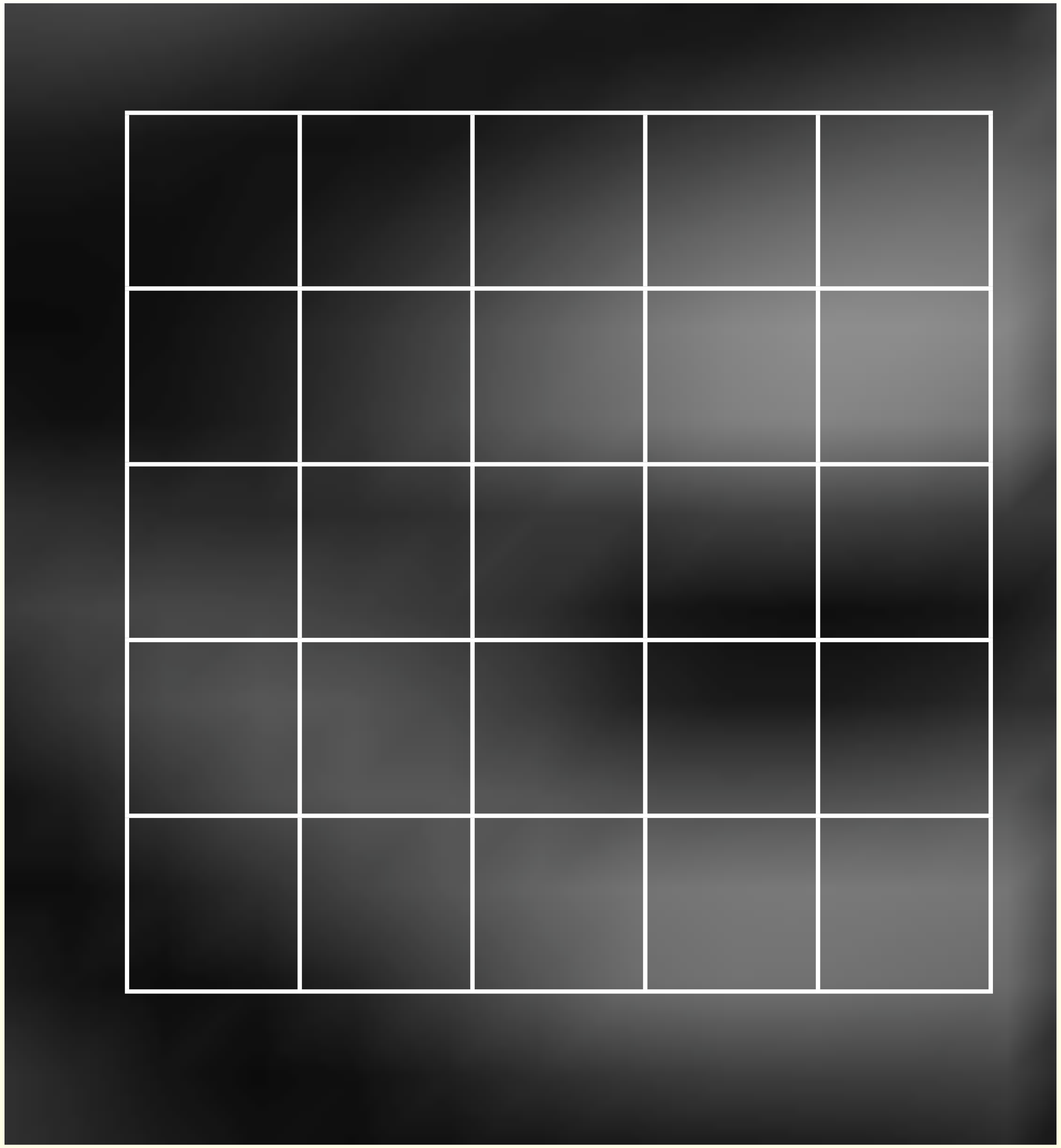


Marching Squares

- Given $I(\mathbf{x})$
  - $I(\mathbf{x}) < q$: $\mathbf{x}$ outside
  - $I(\mathbf{x}) > q$: $\mathbf{x}$ inside
- Discretize space
- Evaluate on the grid

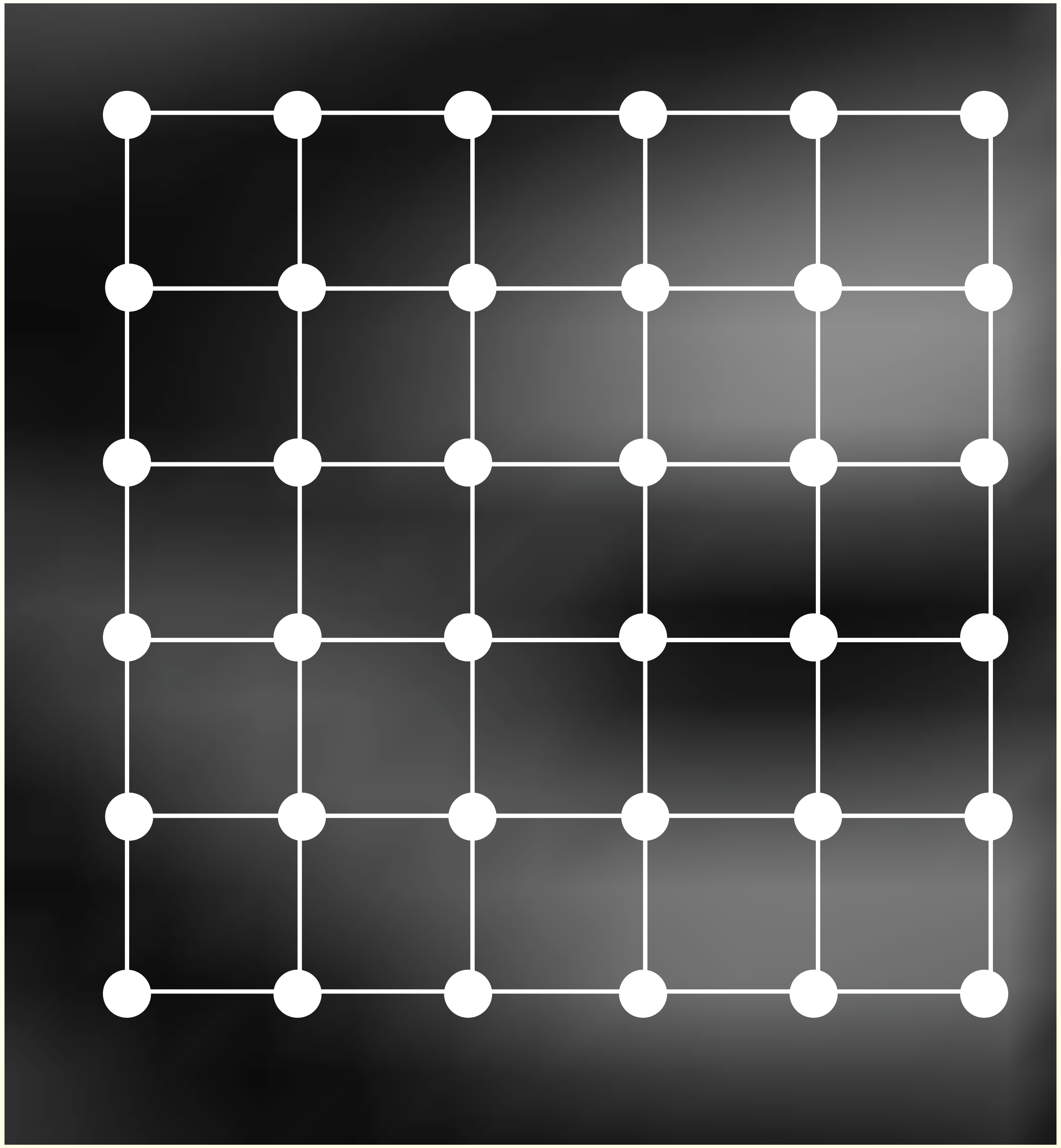
Marching Squares

- Given $I(\mathbf{x})$
  - $I(\mathbf{x}) < q$: $\mathbf{x}$ outside
  - $I(\mathbf{x}) > q$: $\mathbf{x}$ inside
- Discretize space



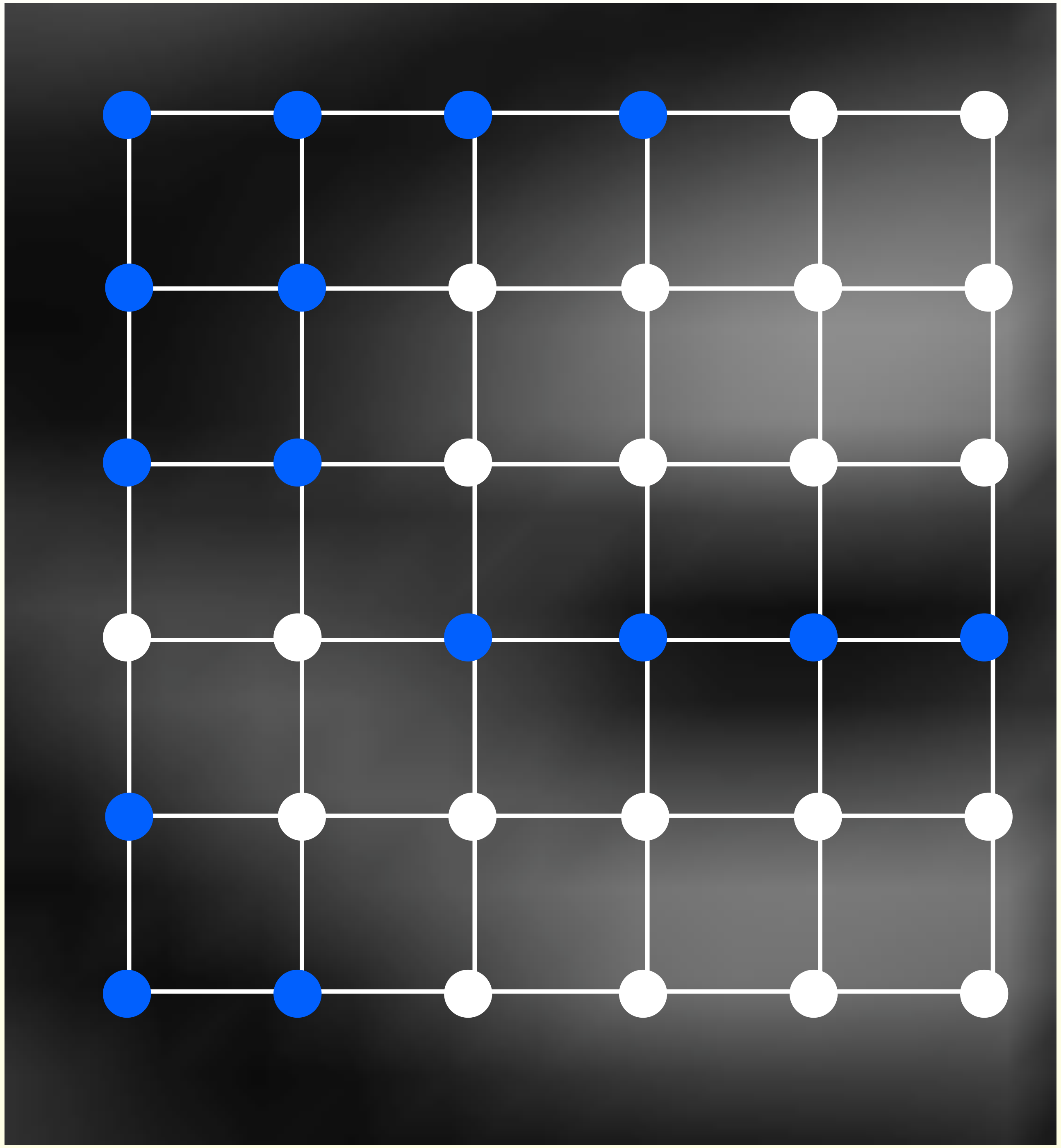
Marching Squares

- Given $I(\mathbf{x})$
  - $I(\mathbf{x}) < q$: $\mathbf{x}$ outside
  - $I(\mathbf{x}) > q$: $\mathbf{x}$ inside
- Discretize space
- Evaluate $f(\mathbf{x})$ on the grid



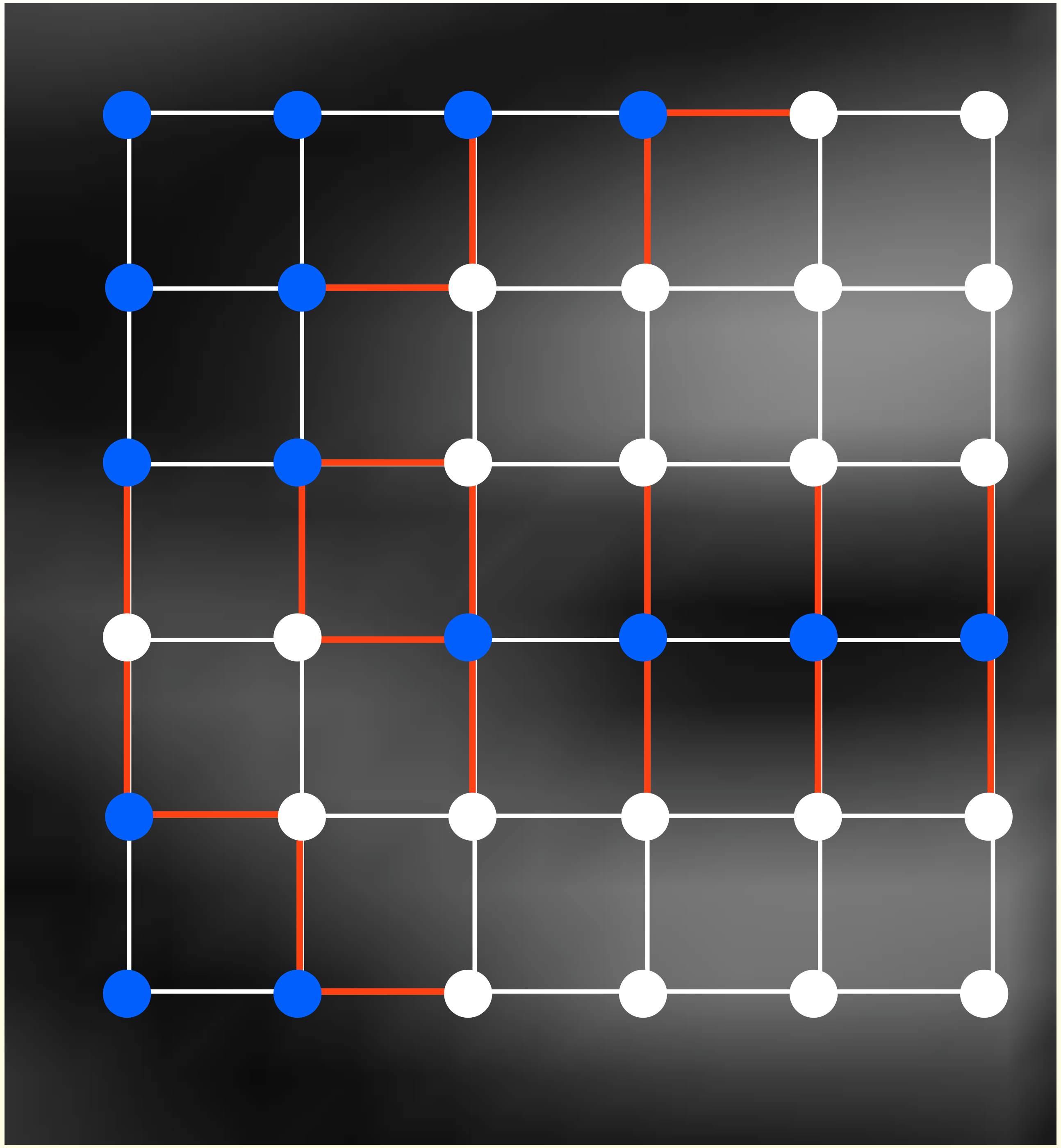
Marching Squares

- Given $I(\mathbf{x})$
  - $I(\mathbf{x}) < q$: $\mathbf{x}$ outside
  - $I(\mathbf{x}) > q$: $\mathbf{x}$ inside
- Discretize space
- Evaluate $f(\mathbf{x})$ on the grid
- Classify grid points



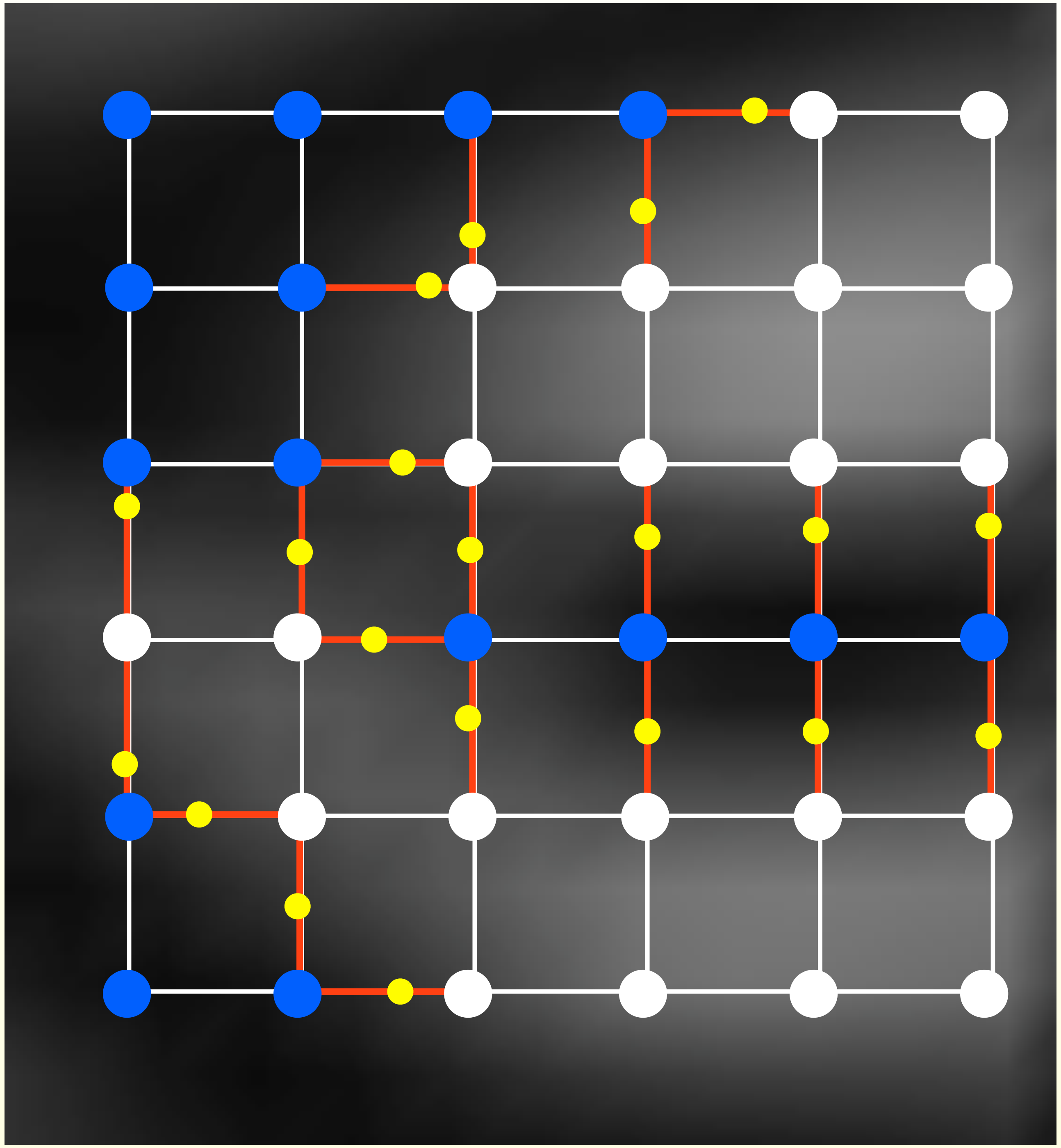
Marching Squares

- Given $I(\mathbf{x})$
  - $I(\mathbf{x}) < q : \mathbf{x}$ outside
  - $I(\mathbf{x}) > q : \mathbf{x}$ inside
- Discretize space
- Evaluate $f(\mathbf{x})$ on the grid
- Classify grid points
- Classify grid edges



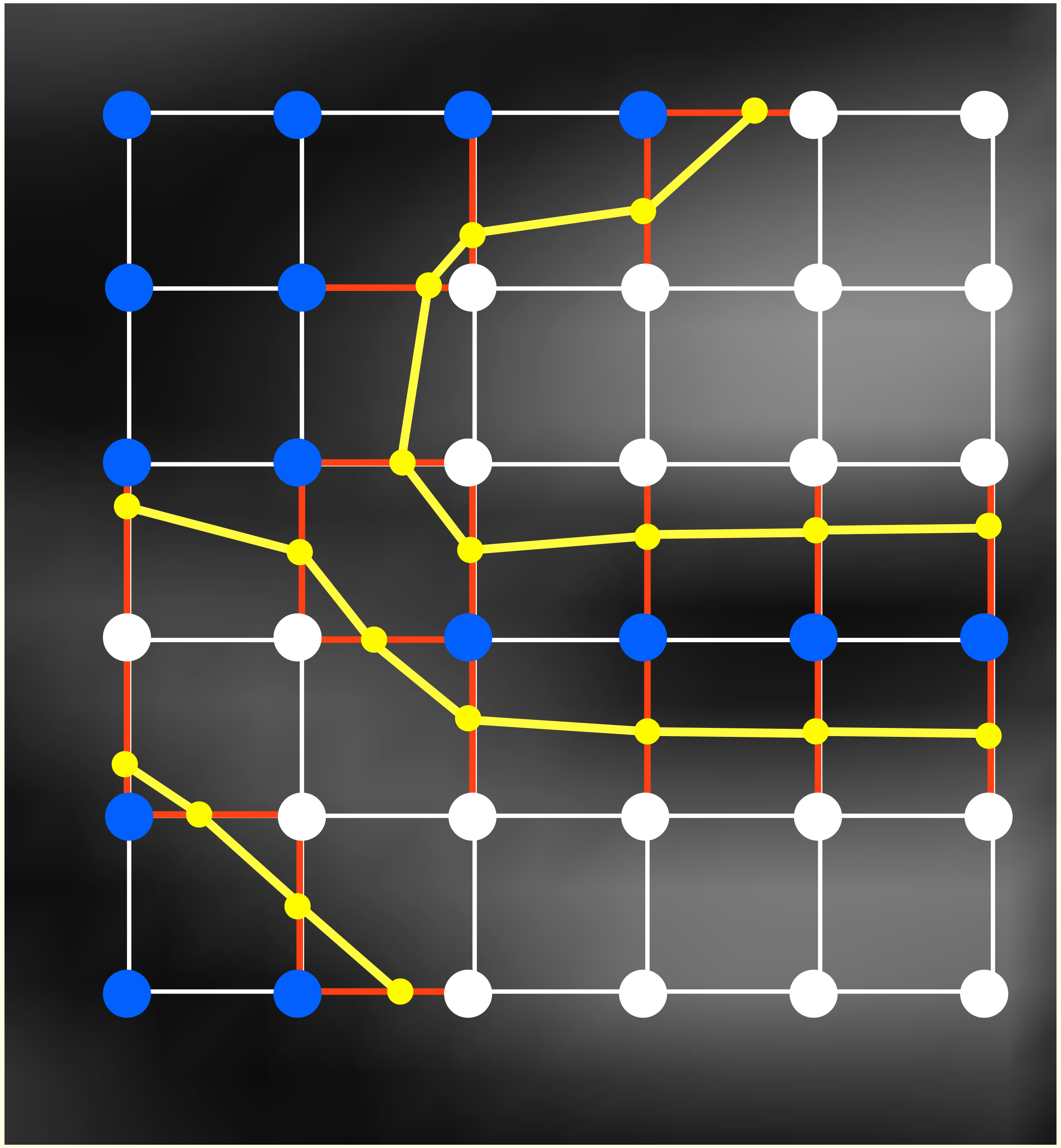
Marching Squares

- Given $I(\mathbf{x})$
  - $I(\mathbf{x}) < q : \mathbf{x}$ outside
  - $I(\mathbf{x}) > q : \mathbf{x}$ inside
- Discretize space
- Evaluate $f(\mathbf{x})$ on the grid
- Classify grid points
- Classify grid edges
- Compute intersections



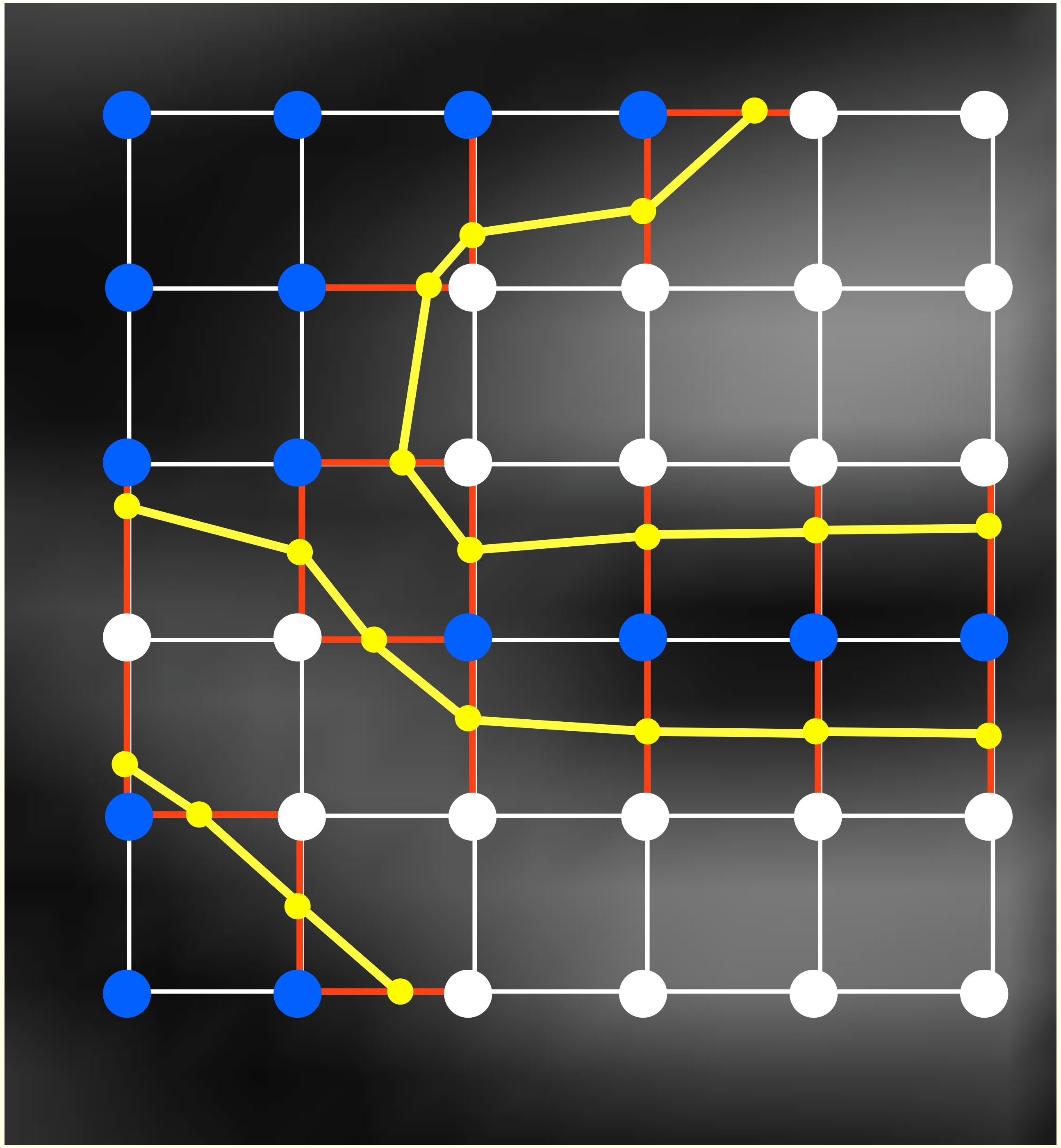
Marching Squares

- Given $I(\mathbf{x})$
  - $I(\mathbf{x}) < q$: $\mathbf{x}$ outside
  - $I(\mathbf{x}) > q$: $\mathbf{x}$ inside
- Discretize space
- Evaluate $f(\mathbf{x})$ on the grid
- Classify grid points
- Classify grid edges
- Compute intersections
- Connect intersections



Marching Squares

- Given $I(\mathbf{x})$
  - $I(\mathbf{x}) < q$: $\mathbf{x}$ outside
  - $I(\mathbf{x}) > q$: $\mathbf{x}$ inside
- Discretize space
- Evaluate $f(\mathbf{x})$ on the grid
- Classify grid points
- Classify grid edges
- Compute intersections
- Connect intersections



Computing Intersections

- Edges with a sign switch contain intersections

$$
\begin{aligned}
& f(\mathbf{x}_1) < 0 \text{ and } f(\mathbf{x}_2) \geq 0 \\
\Rightarrow \quad & f(\mathbf{x}_1 + t(\mathbf{x}_2 - \mathbf{x}_1)) = 0 \text{ for some } 0 < t \leq 1
\end{aligned}
$$

- Nonlinear equation, use raycasting to find root

Computing Intersections

- Edges with a sign switch contain intersections

$$f(\mathbf{x}_1) < 0 \text{ and } f(\mathbf{x}_2) \geq 0$$
$$\Rightarrow \quad f(\mathbf{x}_1 + t(\mathbf{x}_2 - \mathbf{x}_1)) = 0 \text{ for some } 0 < t \leq 1$$

- Nonlinear equation, use raycasting to find root
- Sampled data
  - f is trilinear
  - f is linear along $\mathbf{x}_2 - \mathbf{x}_1$

$$f(\mathbf{x}_1 + t(\mathbf{x}_2 - \mathbf{x}_1)) = 0$$

Computing Intersections

- Edges with a sign switch contain intersections

$$f(\mathbf{x}_1) < 0 \text{ and } f(\mathbf{x}_2) \geq 0$$
$$\Rightarrow \quad f(\mathbf{x}_1 + t(\mathbf{x}_2 - \mathbf{x}_1)) = 0 \text{ for some } 0 < t \leq 1$$

- Nonlinear equation, use raycasting to find root
- Sampled data
  - f is trilinear
  - f is linear along $\mathbf{x}_2 - \mathbf{x}_1$

$$f(\mathbf{x}_1) + t(f(\mathbf{x}_2) - f(\mathbf{x}_1)) = 0$$

Computing Intersections

- Edges with a sign switch contain intersections

$$
\begin{aligned}
& f(\mathbf{x}_1) < 0 \text{ and } f(\mathbf{x}_2) \geq 0 \\
\Rightarrow \quad & f(\mathbf{x}_1 + t(\mathbf{x}_2 - \mathbf{x}_1)) = 0 \text{ for some } 0 < t \leq 1
\end{aligned}
$$

- Nonlinear equation, use raycasting to find root
- Sampled data
  - f is trilinear
  - f is linear along $\mathbf{x}_2 - \mathbf{x}_1$

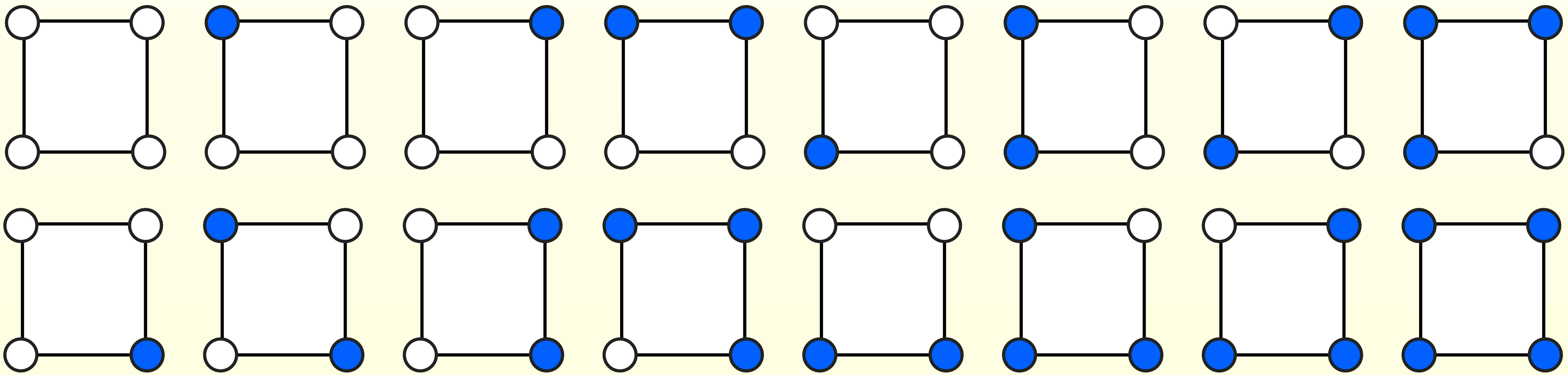
$$t = -f(\mathbf{x}_1)/(f(\mathbf{x}_2) - f(\mathbf{x}_1))$$

Connecting Intersections

- Treat each cell separately

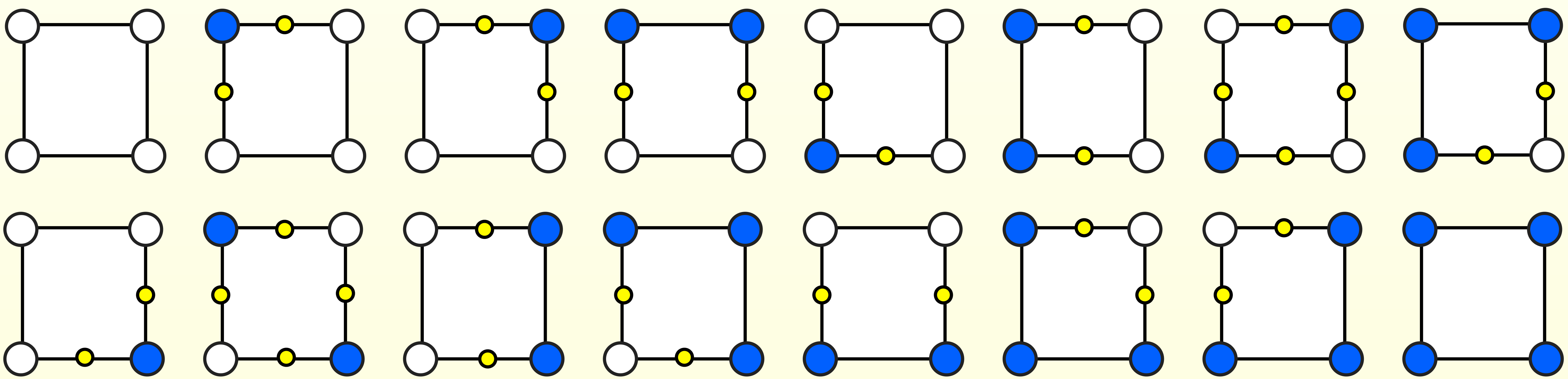
Connecting Intersections

- Treat each cell separately
- Enumerate all possible inside/outside combinations



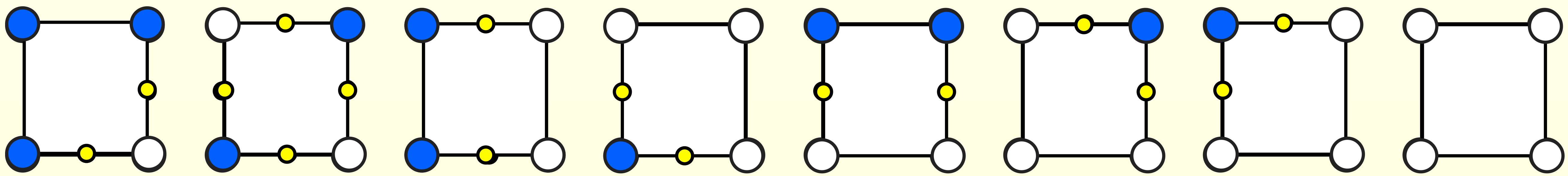
Connecting Intersections

- Treat each cell separately
- Enumerate all possible inside/outside combinations
- Group those leading to the same intersections



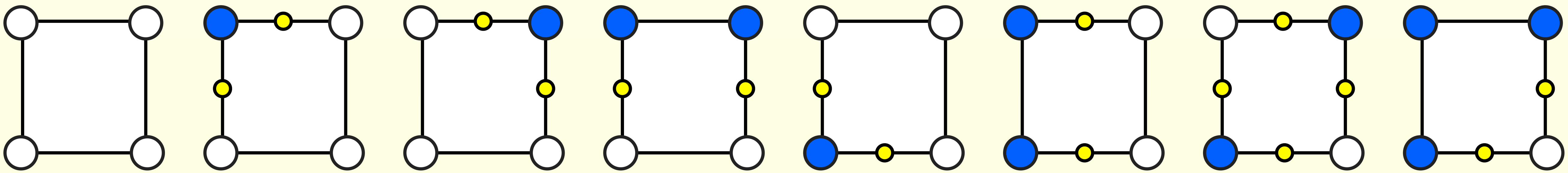
Connecting Intersections

- Treat each cell separately
- Enumerate all possible inside/outside combinations
- Group those leading to the same intersections



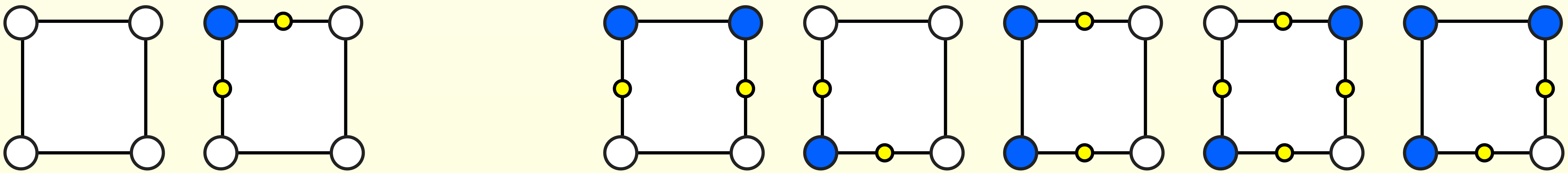
Connecting Intersections

- Treat each cell separately
- Enumerate all possible inside/outside combinations
- Group those leading to the same intersections
- Group those equivalent after rotations



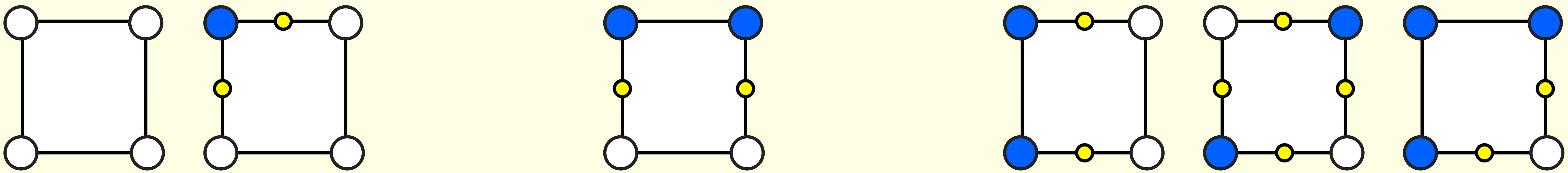
Connecting Intersections

- Treat each cell separately
- Enumerate all possible inside/outside combinations
- Group those leading to the same intersections
- Group those equivalent after rotations



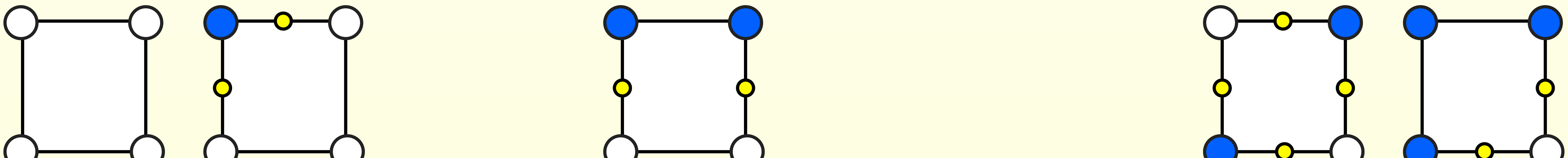
Connecting Intersections

- Treat each cell separately
- Enumerate all possible inside/outside combinations
- Group those leading to the same intersections
- Group those equivalent after rotations



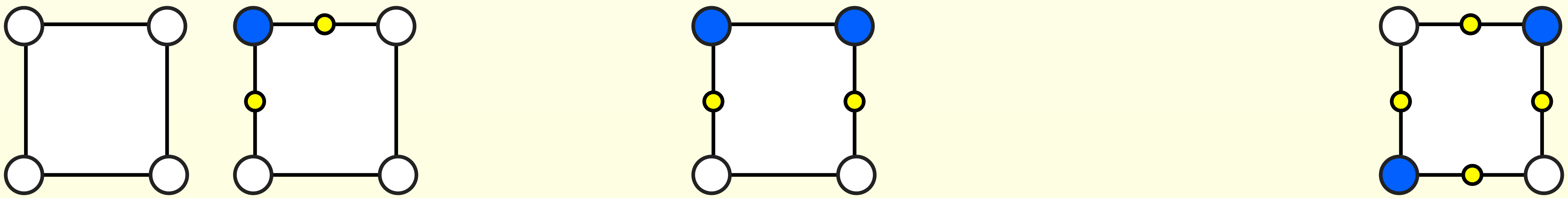
Connecting Intersections

- Treat each cell separately
- Enumerate all possible inside/outside combinations
- Group those leading to the same intersections
- Group those equivalent after rotations



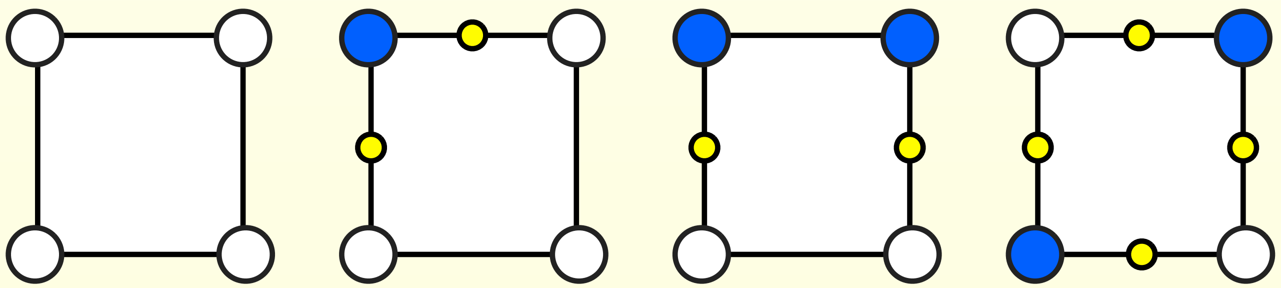
Connecting Intersections

- Treat each cell separately
- Enumerate all possible inside/outside combinations
- Group those leading to the same intersections
- Group those equivalent after rotations



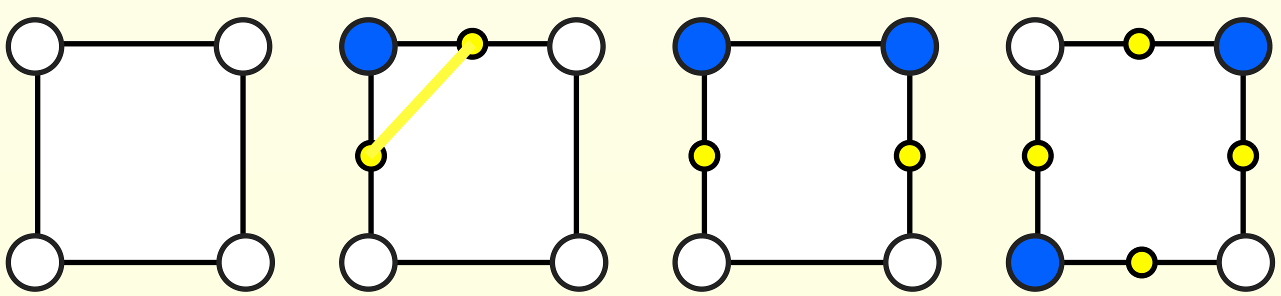
Connecting Intersections

- Treat each cell separately
- Enumerate all possible inside/outside combinations
- Group those leading to the same intersections
- Group those equivalent after rotations
- Connect intersections



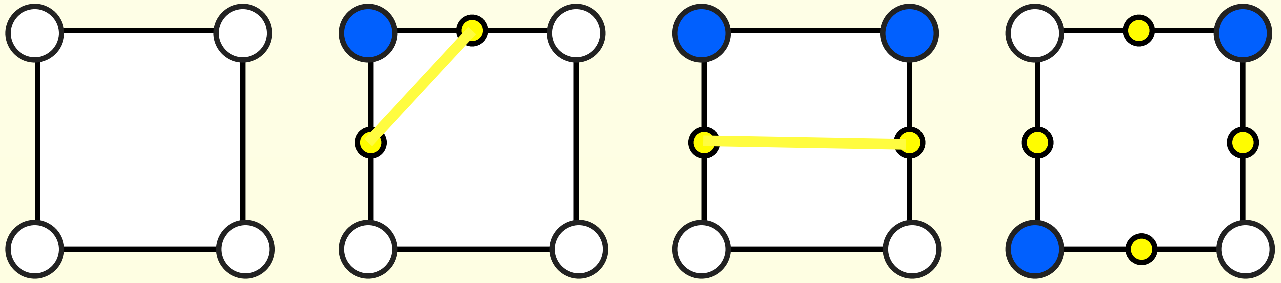
Connecting Intersections

- Treat each cell separately
- Enumerate all possible inside/outside combinations
- Group those leading to the same intersections
- Group those equivalent after rotations
- Connect intersections



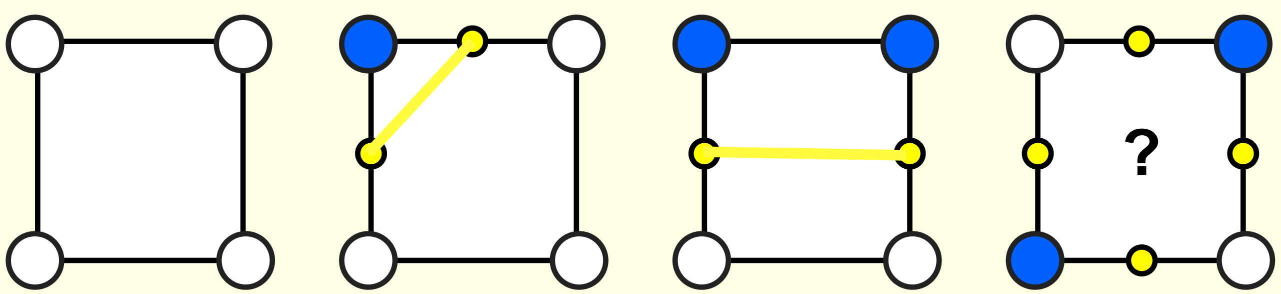
Connecting Intersections

- Treat each cell separately
- Enumerate all possible inside/outside combinations
- Group those leading to the same intersections
- Group those equivalent after rotations
- Connect intersections

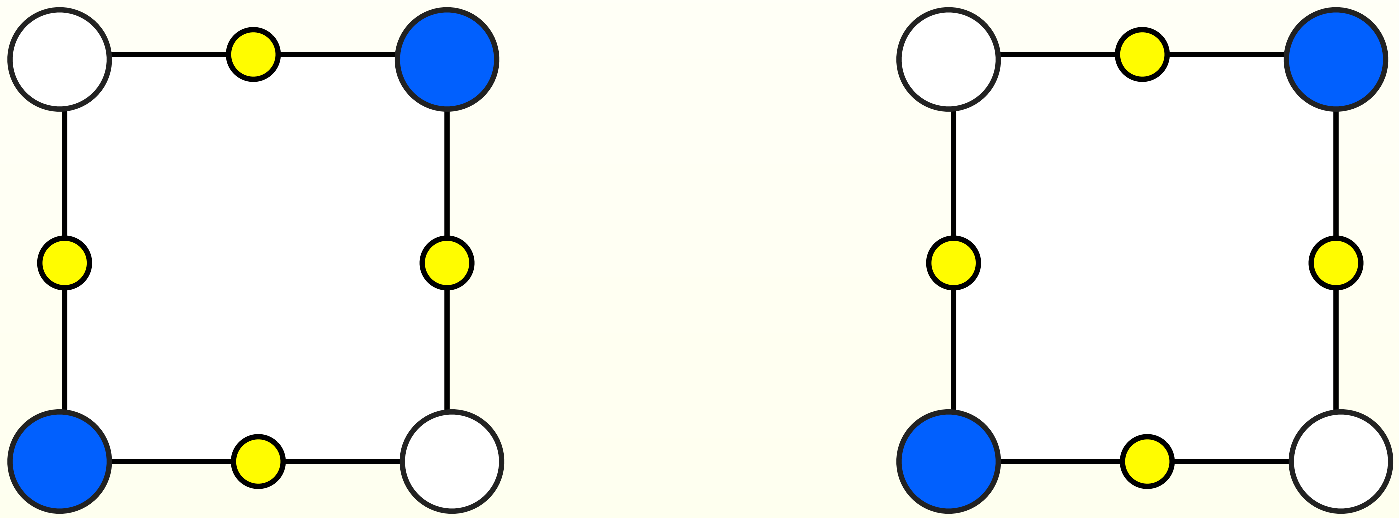


Connecting Intersections

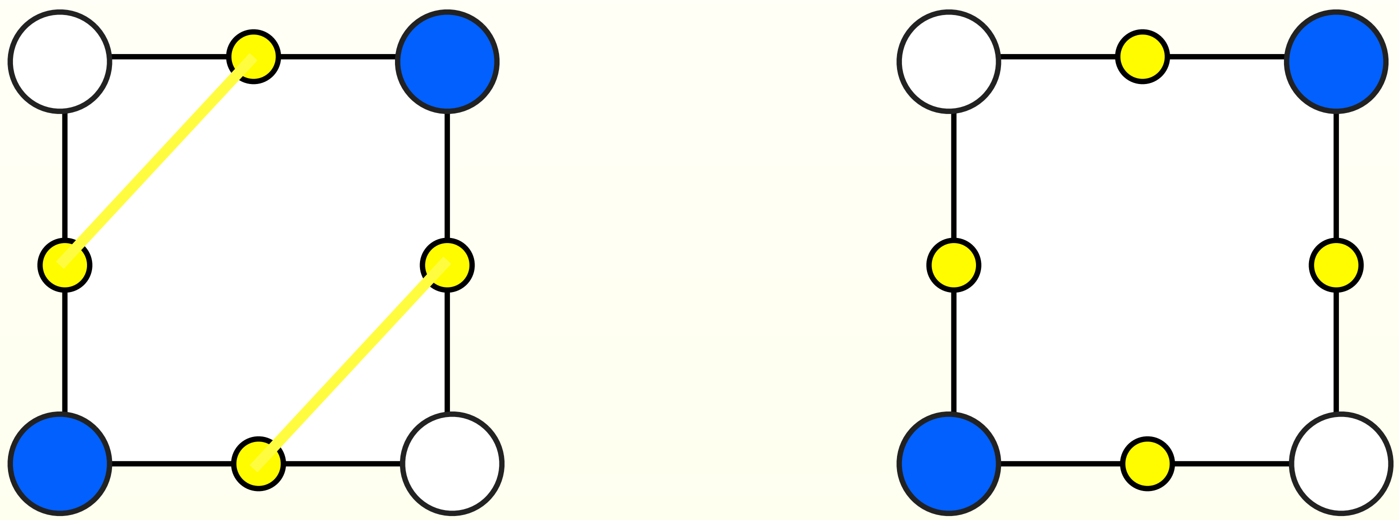
- Treat each cell separately
- Enumerate all possible inside/outside combinations
- Group those leading to the same intersections
- Group those equivalent after rotations
- Connect intersections



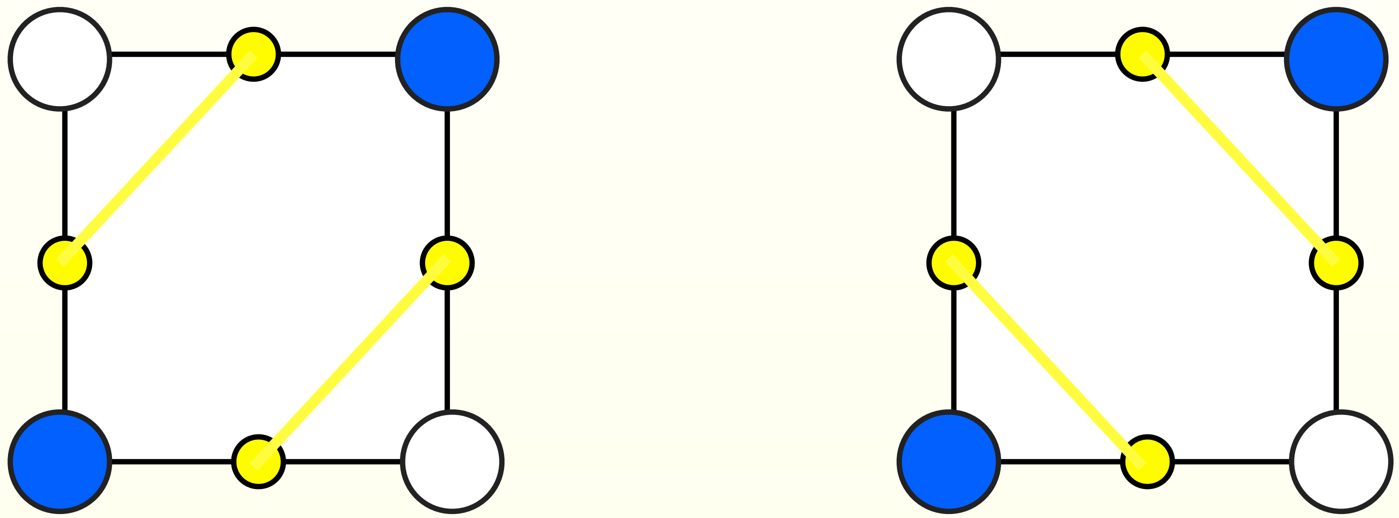
Ambiguous case



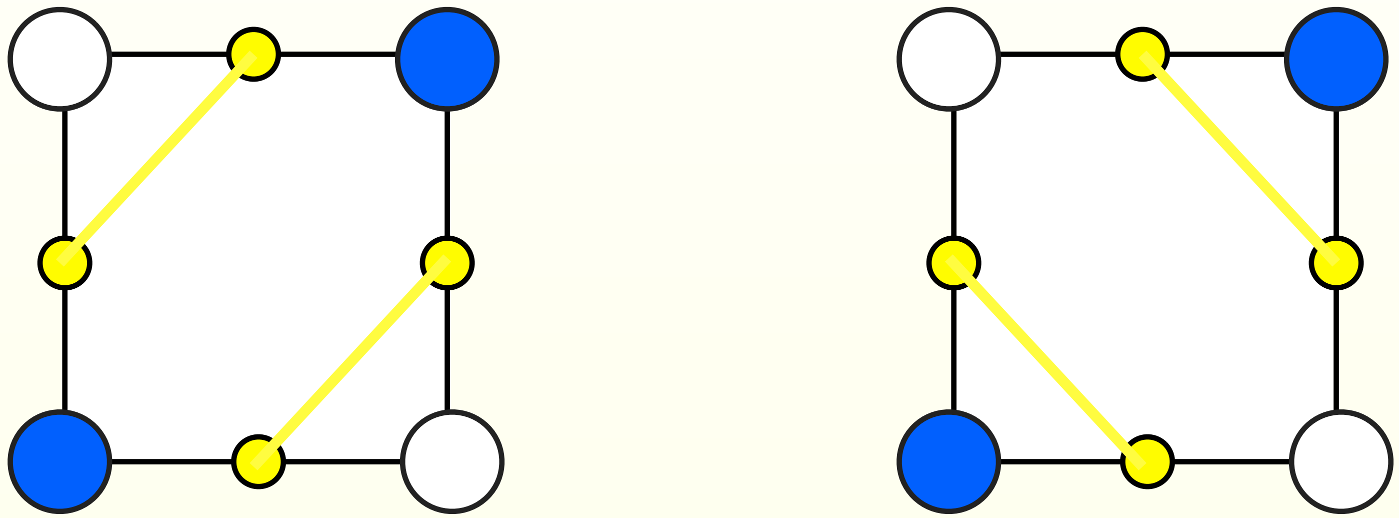
Ambiguous case



Ambiguous case

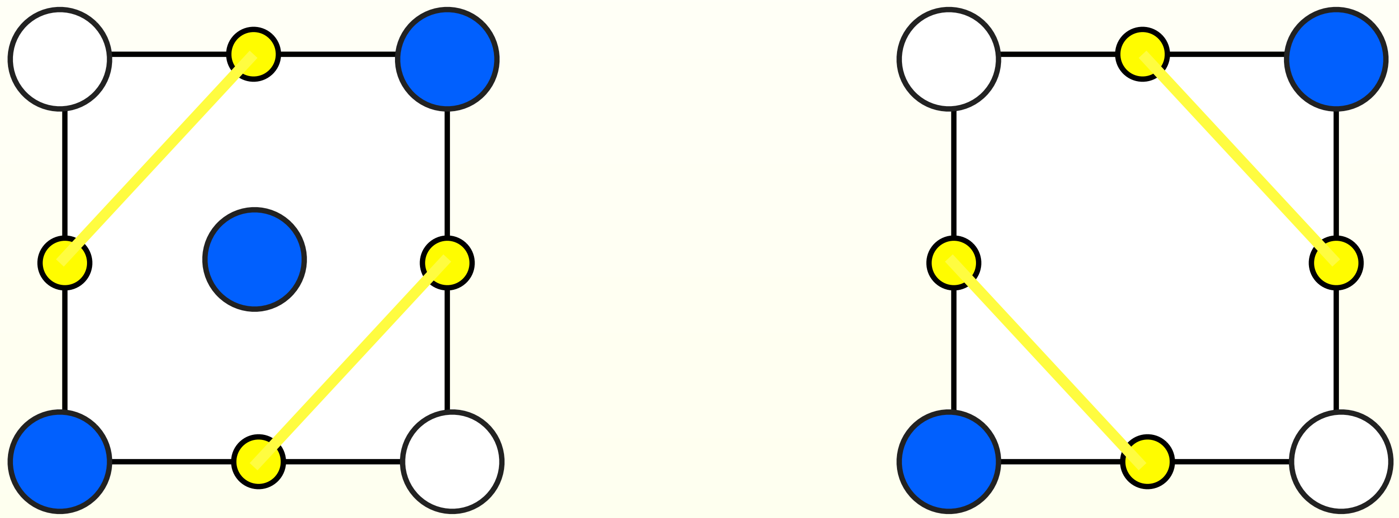


Ambiguous case



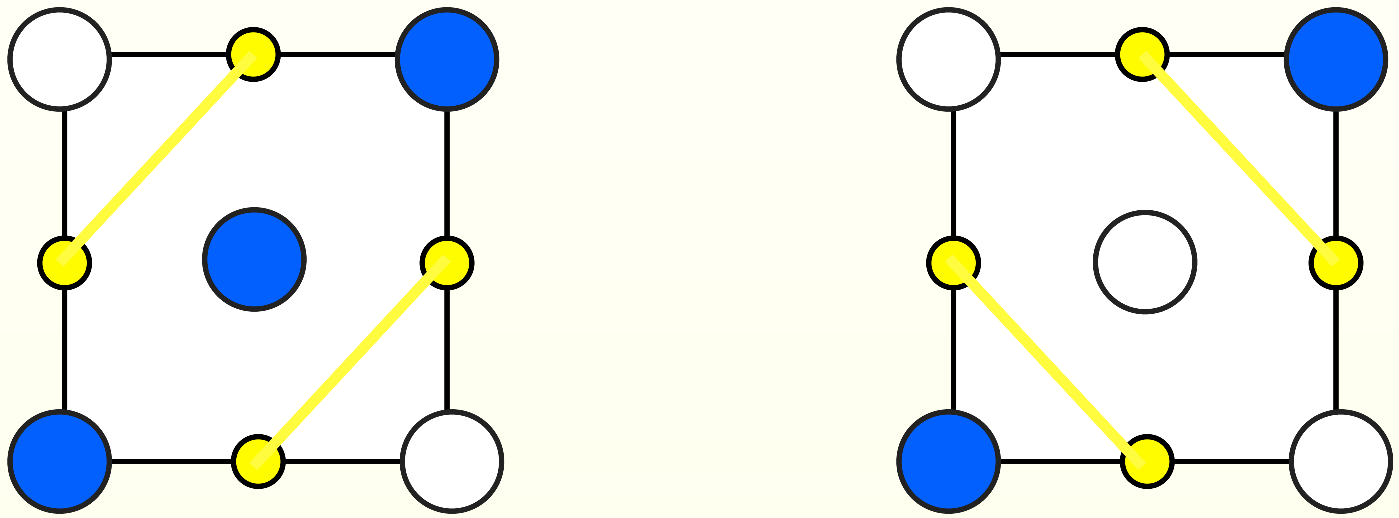
- No way to decide without further samples

Ambiguous case



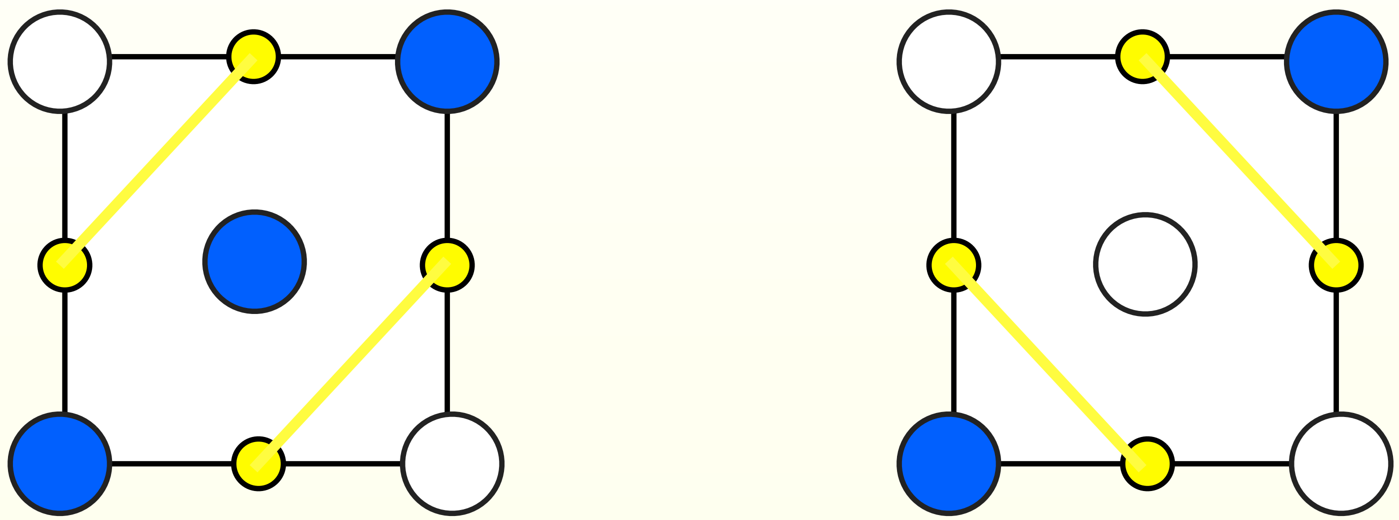
- No way to decide without further samples

Ambiguous case



- No way to decide without further samples

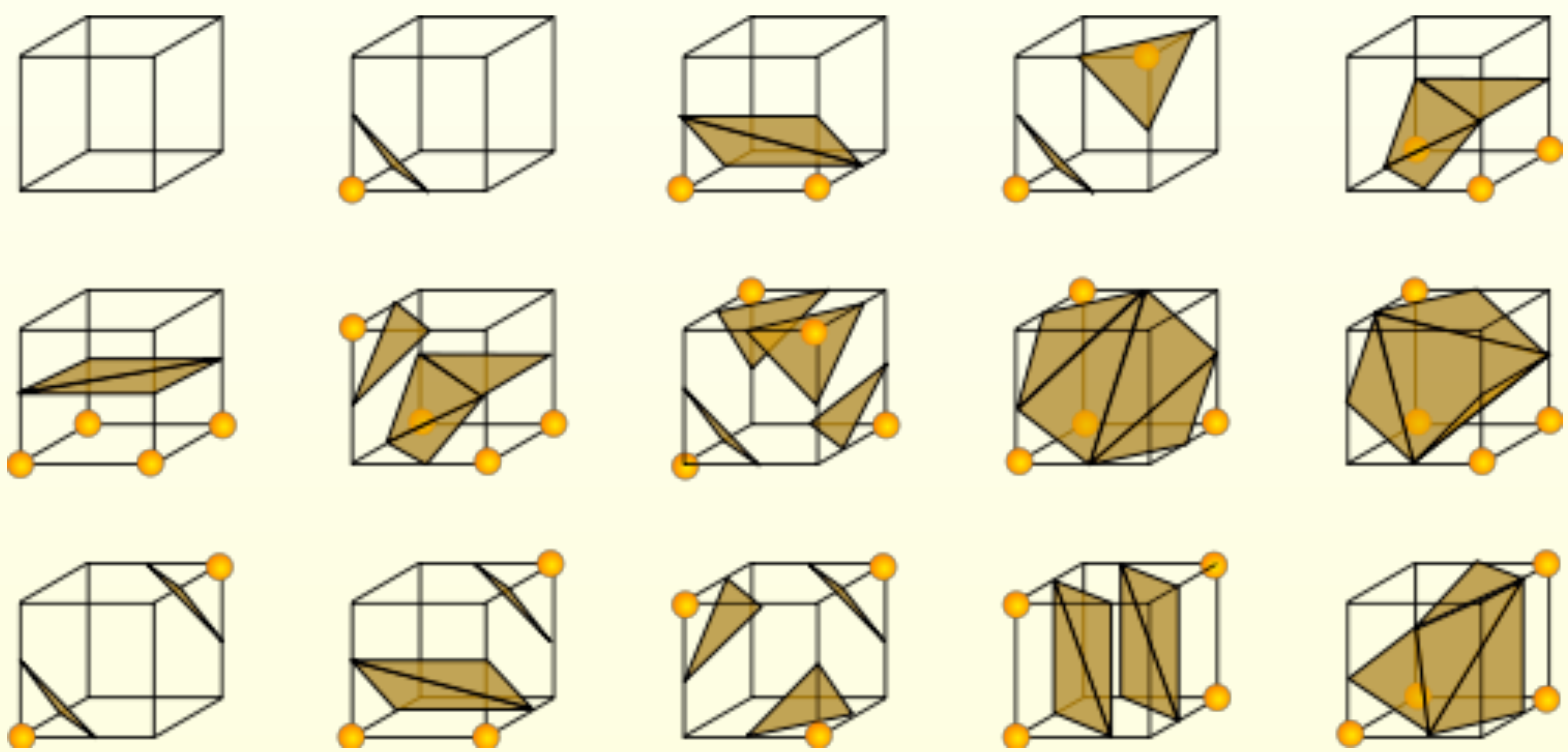
Ambiguous case



- No way to decide without further samples
- No samples available: Just choose one

Marching Cubes

- Same basic principle in 3D
- Lines become surface patches
  - Up to 4 triangles per voxel
- 256 different cases, 15 after symmetries



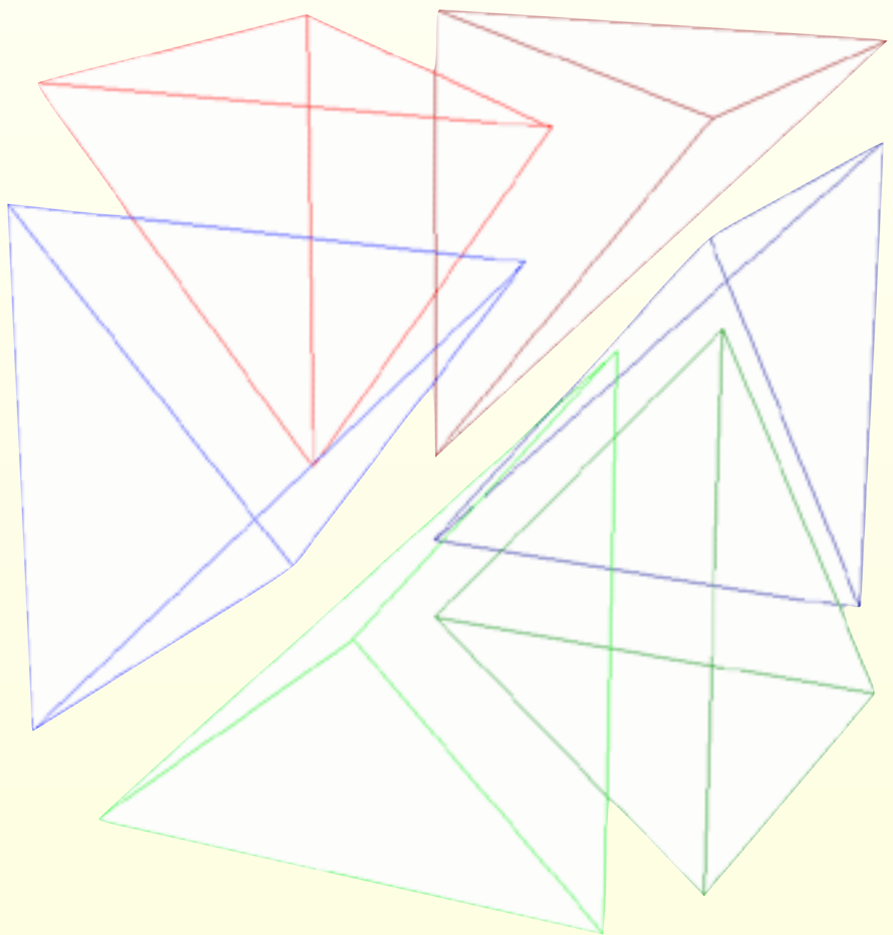
Marching Tetrahedra

- Different discretization: Tetrahedra

[Paul Bourke]

Marching Tetrahedra

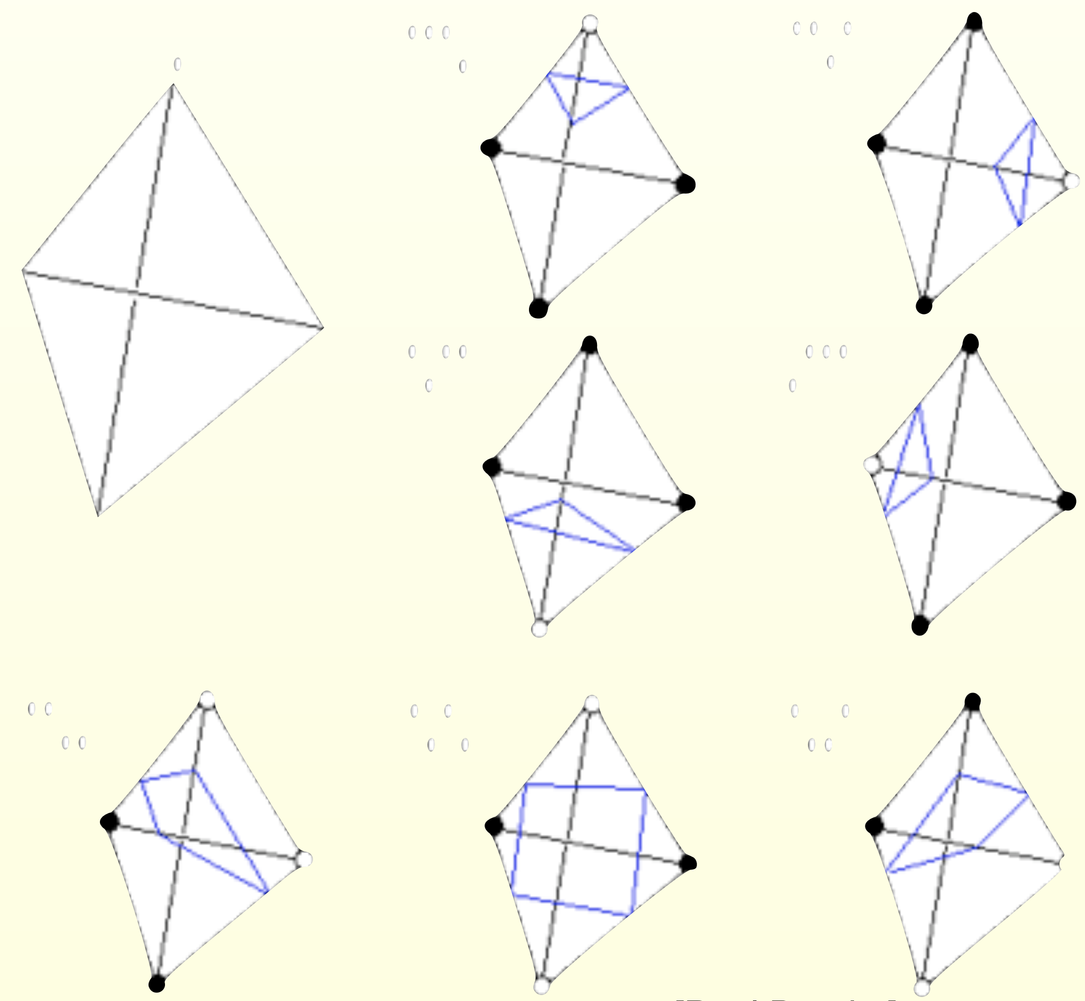
- Different discretization: Tetrahedra
  - 6 tetrahedra per voxel (if we start from cubes)



[Paul Bourke]

Marching Tetrahedra

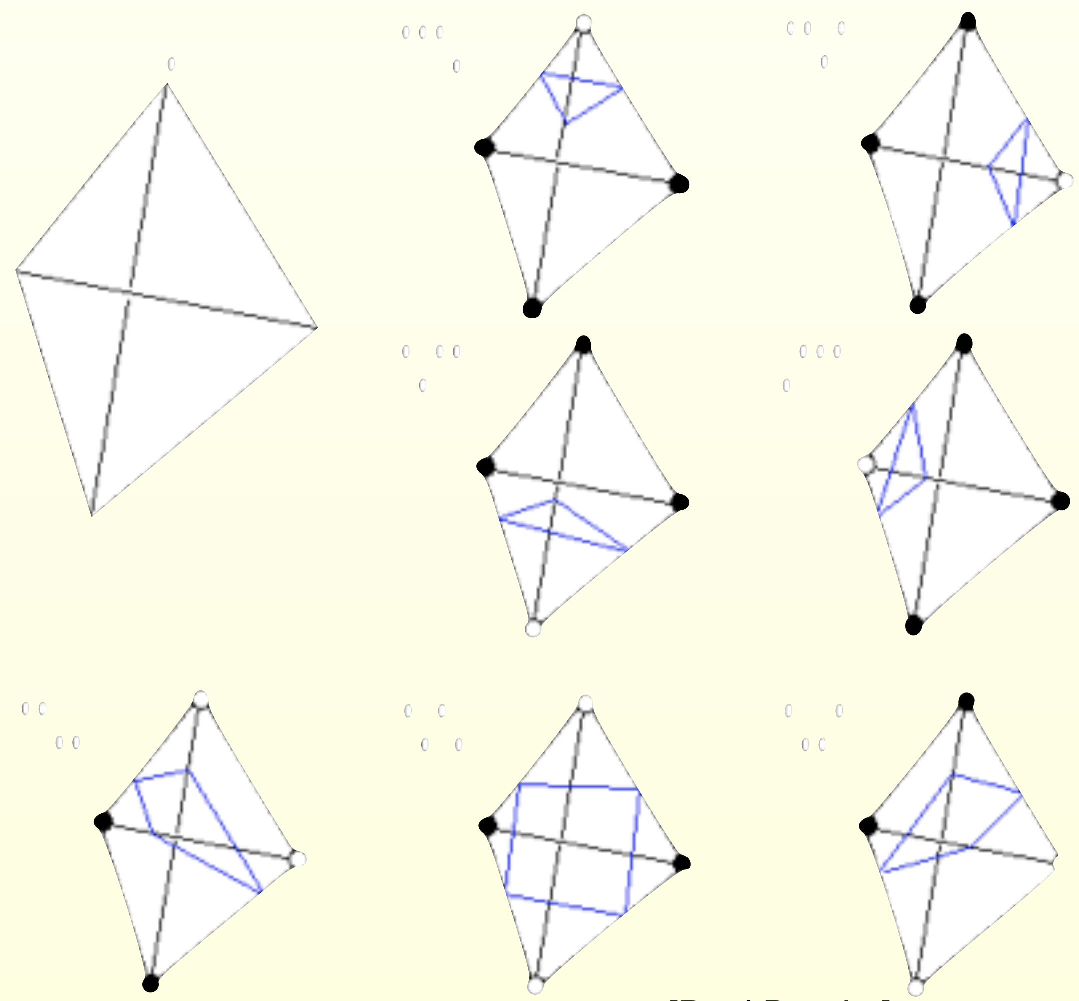
- Different discretization: Tetrahedra
  - 6 tetrahedra per voxel (if we start from cubes)
  - 16 cases, 8 after symmetry



[Paul Bourke]

Marching Tetrahedra

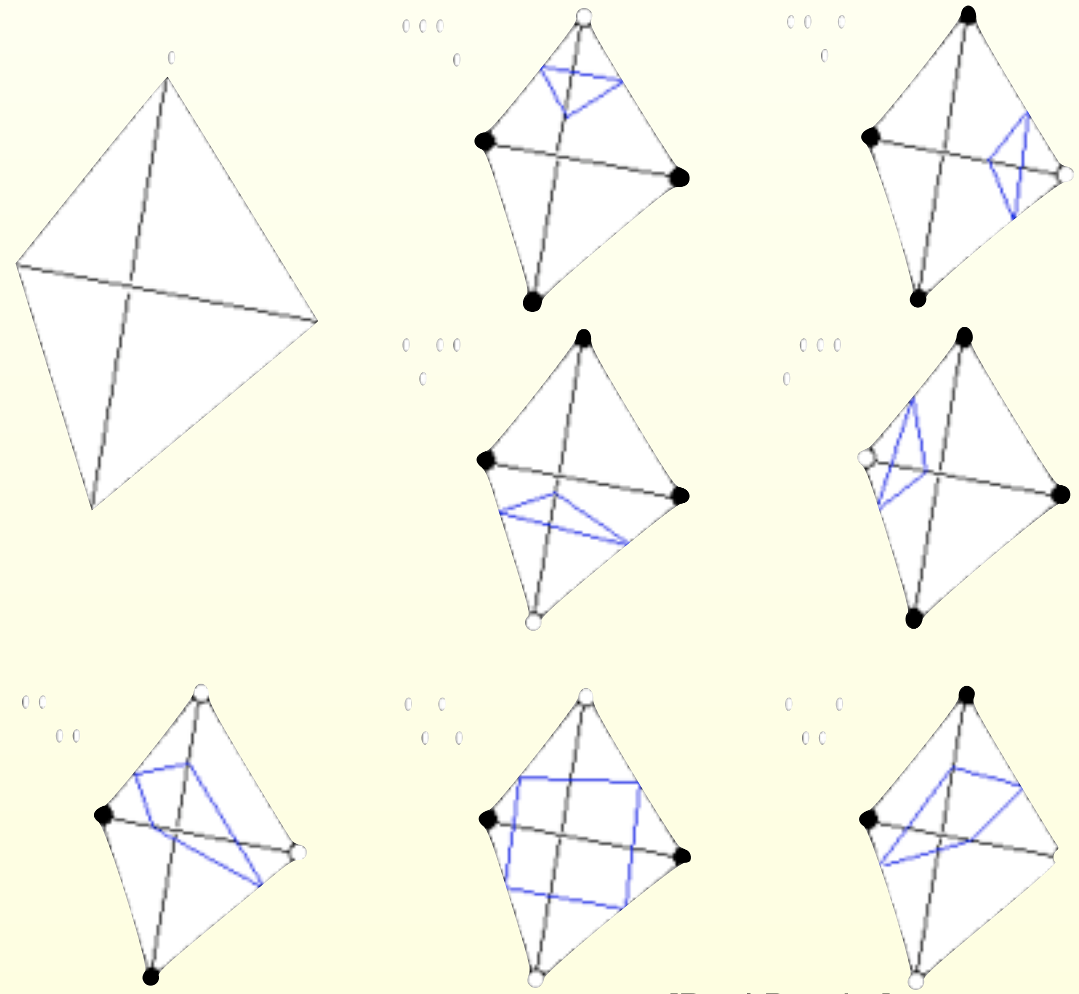
- Different discretization: Tetrahedra
  - 6 tetrahedra per voxel (if we start from cubes)
  - 16 cases, 8 after symmetry
  - Up to 2 triangles per tet



[Paul Bourke]

Marching Tetrahedra

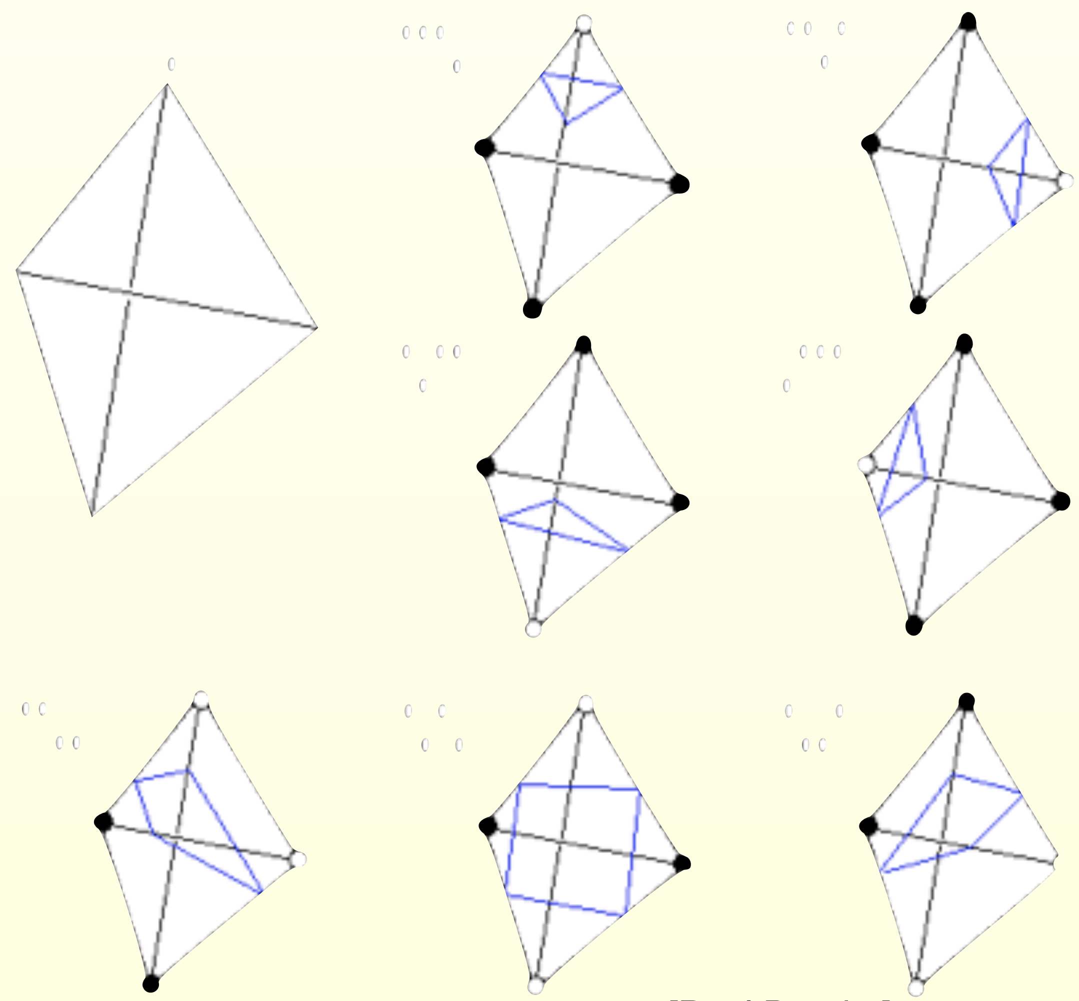
- Different discretization: Tetrahedra
  - 6 tetrahedra per voxel (if we start from cubes)
  - 16 cases, 8 after symmetry
  - Up to 2 triangles per tet
  - No ambiguities



[Paul Bourke]

Marching Tetrahedra

- Different discretization: Tetrahedra
  - 6 tetrahedra per voxel (if we start from cubes)
  - 16 cases, 8 after symmetry
  - Up to 2 triangles per tet
  - No ambiguities
- Used when input data discretized as tetrahedra



[Paul Bourke]

Implementation

- Big lookup tables

```
int edgeTable[256]={
0x0  , 0x109, 0x203, 0x30a, 0x406, 0x50f, 0x605, 0x70c,
0x80c, 0x905, 0xa0f, 0xb06, 0xc0a, 0xd03, 0xe09, 0xf00,
0x190, 0x99 , 0x393, 0x29a, 0x596, 0x49f, 0x795, 0x69c,
0x99c, 0x895, 0xb9f, 0xa96, 0xd9a, 0xc93, 0xf99, 0xe90,
0x230, 0x339, 0x33 , 0x13a, 0x636, 0x73f, 0x435, 0x53c,
0xa3c, 0xb35, 0x83f, 0x936, 0xe3a, 0xf33, 0xc39, 0xd30,
0x3a0, 0x2a9, 0x1a3, 0xaa , 0x7a6, 0x6af, 0x5a5, 0x4ac,
0xbac, 0xaa5, 0x9af, 0x8a6, 0xfaa, 0xea3, 0xda9, 0xca0,
0x460, 0x569, 0x663, 0x76a, 0x66 , 0x16f, 0x265, 0x36c,
0xc6c, 0xd65, 0xe6f, 0xf66, 0x86a, 0x963, 0xa69, 0xb60,
0x5f0, 0x4f9, 0x7f3, 0x6fa, 0x1f6, 0xff , 0x3f5, 0x2fc,
0xdfc, 0xcf5, 0xfff, 0xef6, 0x9fa, 0x8f3, 0xbf9, 0xaf0,
0x650, 0x759, 0x453, 0x55a, 0x256, 0x35f, 0x55 , 0x15c,
0xe5c, 0xf55, 0xc5f, 0xd56, 0xa5a, 0xb53, 0x859, 0x950,
0x7c0, 0x6c9, 0x5c3, 0x4ca, 0x3c6, 0x2cf, 0x1c5, 0xcc ,
0xfcc, 0xec5, 0xdcf, 0xcc6, 0xbca, 0xac3, 0x9c9, 0x8c0,
0x8c0, 0x9c9, 0xac3, 0xbca, 0xcc6, 0xdcf, 0xec5, 0xfcc,
0xcc , 0x1c5, 0x2cf, 0x3c6, 0x4ca, 0x5c3, 0x6c9, 0x7c0,
0x950, 0x859, 0xb53, 0xa5a, 0xd56, 0xc5f, 0xf55, 0xe5c,
0x15c, 0x55 , 0x35f, 0x256, 0x55a, 0x453, 0x759, 0x650,
0xaf0, 0xbf9, 0x8f3, 0x9fa, 0xef6, 0xfff, 0xcf5, 0xdfc,
0x2fc, 0x3f5, 0xff , 0x1f6, 0x6fa, 0x7f3, 0x4f9, 0x5f0,
0xb60, 0xa69, 0x963, 0x86a, 0xf66, 0xe6f, 0xd65, 0xc6c,
0x36c, 0x265, 0x16f, 0x66 , 0x76a, 0x663, 0x569, 0x460,
0xca0, 0xda9, 0xea3, 0xfaa, 0x8a6, 0x9af, 0xaa5, 0xbac,
0x4ac, 0x5a5, 0x6af, 0x7a6, 0xaa , 0x1a3, 0x2a9, 0x3a0,
0xd30, 0xc39, 0xf33, 0xe3a, 0x936, 0x83f, 0xb35, 0xa3c,
0x53c, 0x435, 0x73f, 0x636, 0x13a, 0x33 , 0x339, 0x230,
0xe90, 0xf99, 0xc93, 0xd9a, 0xa96, 0xb9f, 0x895, 0x99c,
0x69c, 0x795, 0x49f, 0x596, 0x29a, 0x393, 0x99 , 0x190,
0xf00, 0xe09, 0xd03, 0xc0a, 0xb06, 0xa0f, 0x905, 0x80c,
0x70c, 0x605, 0x50f, 0x406, 0x30a, 0x203, 0x109, 0x0   };
```

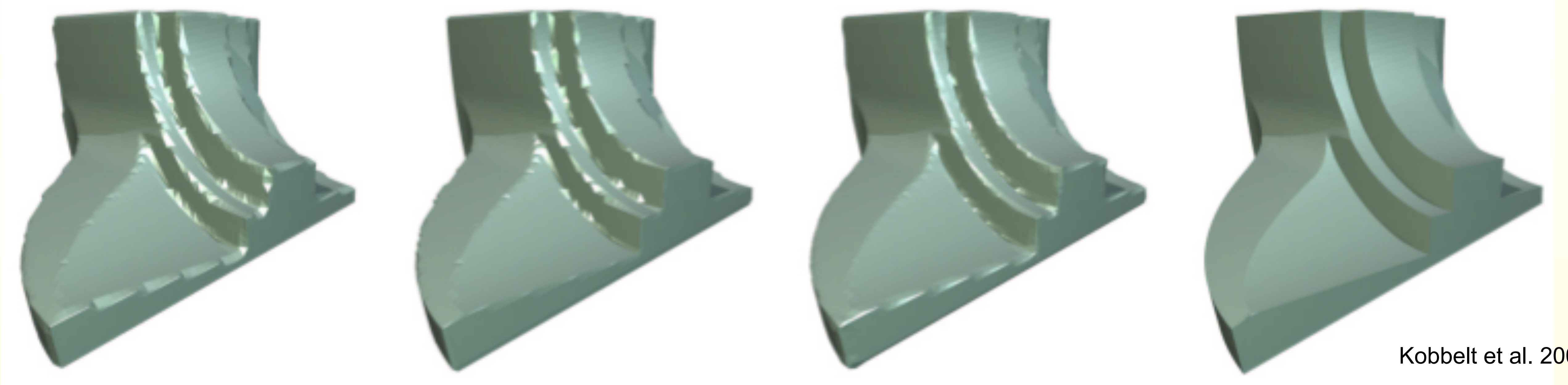
```
int triTable[256][16] =
{{-1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{0, 8, 3, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{0, 1, 9, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{1, 8, 3, 9, 8, 1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{1, 2, 10, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{0, 8, 3, 1, 2, 10, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{9, 2, 10, 0, 2, 9, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{2, 8, 3, 2, 10, 8, 10, 9, 8, -1, -1, -1, -1, -1, -1, -1},
{3, 11, 2, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{0, 11, 2, 8, 11, 0, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{1, 9, 0, 2, 3, 11, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{1, 11, 2, 1, 9, 11, 9, 8, 11, -1, -1, -1, -1, -1, -1, -1},
{3, 10, 1, 11, 10, 3, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{0, 10, 1, 0, 8, 10, 8, 11, 10, -1, -1, -1, -1, -1, -1, -1},
{3, 9, 0, 3, 11, 9, 11, 10, 9, -1, -1, -1, -1, -1, -1, -1},
{9, 8, 10, 10, 8, 11, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{4, 7, 8, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{4, 3, 0, 7, 3, 4, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{0, 1, 9, 8, 4, 7, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{4, 1, 9, 4, 7, 1, 7, 3, 1, -1, -1, -1, -1, -1, -1, -1},
{1, 2, 10, 8, 4, 7, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{3, 4, 7, 3, 0, 4, 1, 2, 10, -1, -1, -1, -1, -1, -1, -1},
{9, 2, 10, 9, 0, 2, 8, 4, 7, -1, -1, -1, -1, -1, -1, -1},
{2, 10, 9, 2, 9, 7, 2, 7, 3, 7, 9, 4, -1, -1, -1, -1},
{8, 4, 7, 3, 11, 2, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{11, 4, 7, 11, 2, 4, 2, 0, 4, -1, -1, -1, -1, -1, -1, -1},
{9, 0, 1, 8, 4, 7, 2, 3, 11, -1, -1, -1, -1, -1, -1, -1},
{4, 7, 11, 9, 4, 11, 9, 11, 2, 9, 2, 1, -1, -1, -1, -1},
{3, 10, 1, 3, 11, 10, 7, 8, 4, -1, -1, -1, -1, -1, -1, -1},
{1, 11, 10, 1, 4, 11, 1, 0, 4, 7, 11, 4, -1, -1, -1, -1},
{4, 7, 8, 9, 0, 11, 9, 11, 10, 11, 0, 3, -1, -1, -1, -1},
{4, 7, 11, 4, 11, 9, 9, 11, 10, -1, -1, -1, -1, -1, -1, -1},
{9, 5, 4, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{9, 5, 4, 0, 8, 3, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{0, 5, 4, 1, 5, 0, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{8, 5, 4, 8, 3, 5, 3, 1, 5, -1, -1, -1, -1, -1, -1, -1},
{1, 2, 10, 9, 5, 4, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{3, 0, 8, 1, 2, 10, 4, 9, 5, -1, -1, -1, -1, -1, -1, -1},
{5, 2, 10, 5, 4, 2, 4, 0, 2, -1, -1, -1, -1, -1, -1, -1},
{2, 10, 5, 3, 2, 5, 3, 5, 4, 3, 4, 8, -1, -1, -1, -1},
{9, 5, 4, 2, 3, 11, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1},
{0, 11, 2, 0, 8, 11, 4, 9, 5, -1, -1, -1, -1, -1, -1, -1},
```

[Paul Bourke]

Problems & Solutions

No Sharp Features

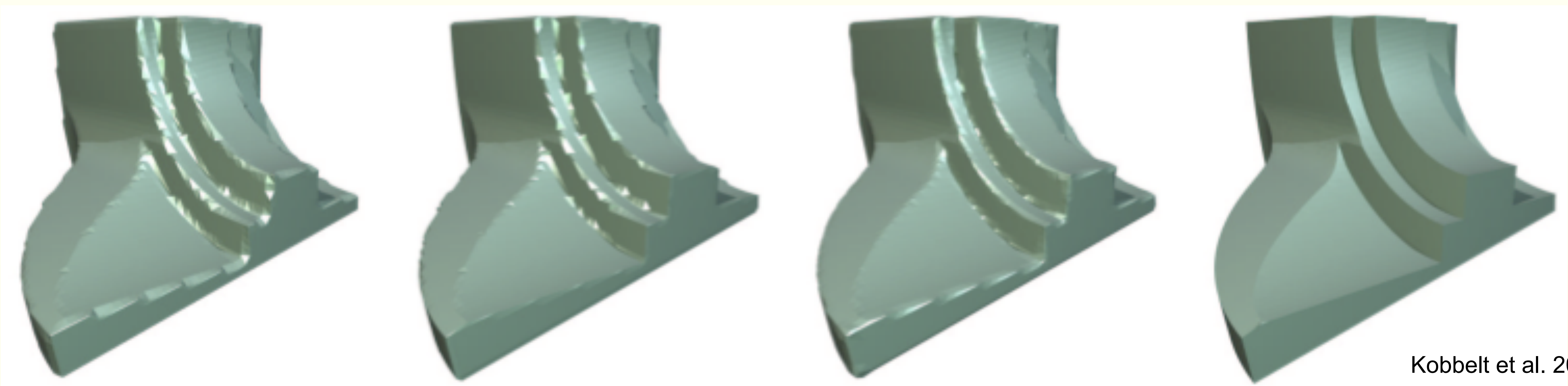
- Increasing grid resolution does not help
- Normals do not converge



Kobbelt et al. 2001

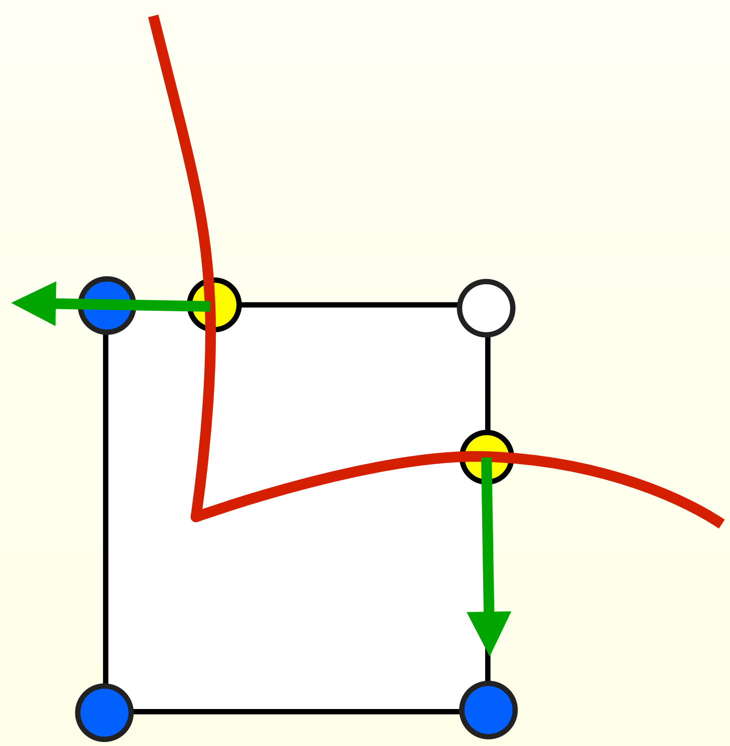
No Sharp Features

- Increasing grid resolution does not help
- Normals do not converge



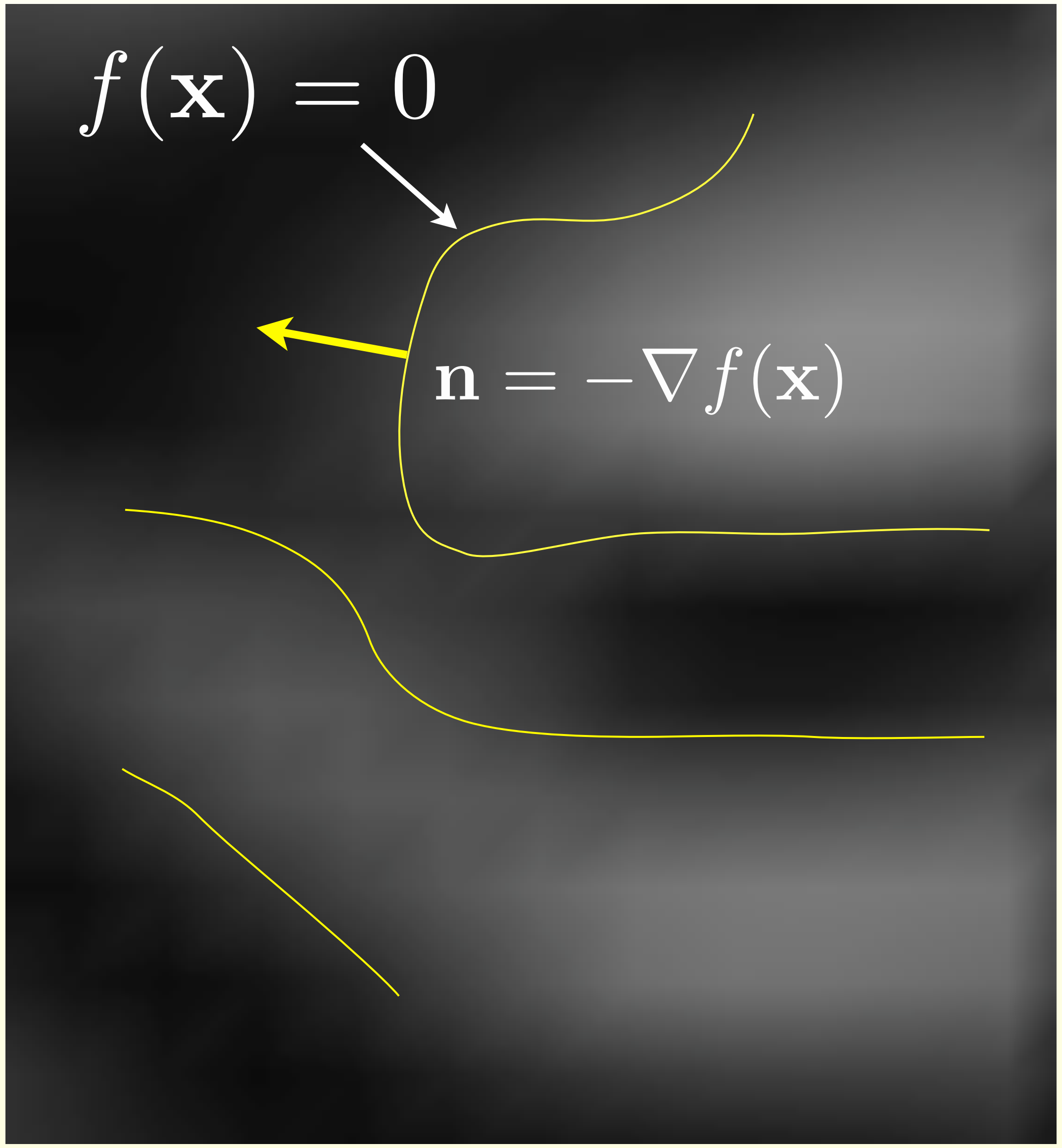
- Use normal information to find edges and corners

Using Hermite Data



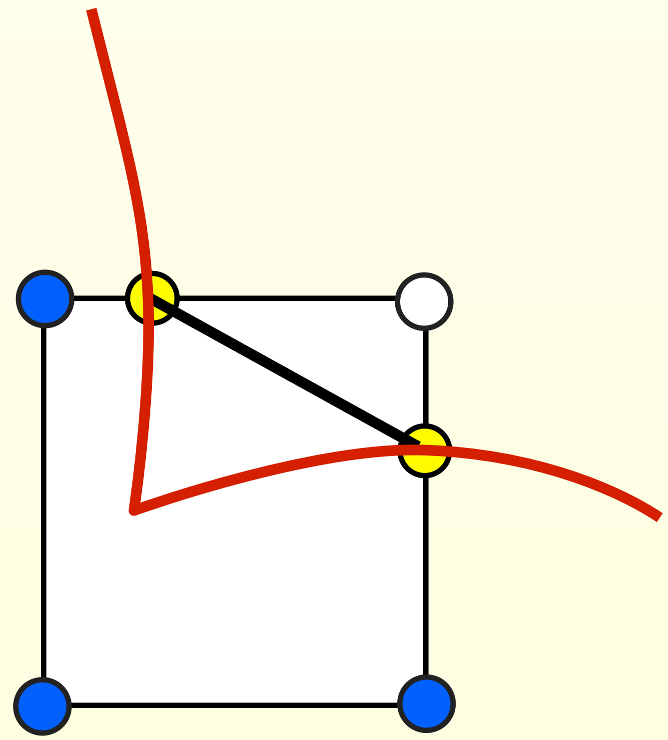
Where Do Normals Come From?

- Gradient of the function $f(\mathbf{x})$ defining the surface



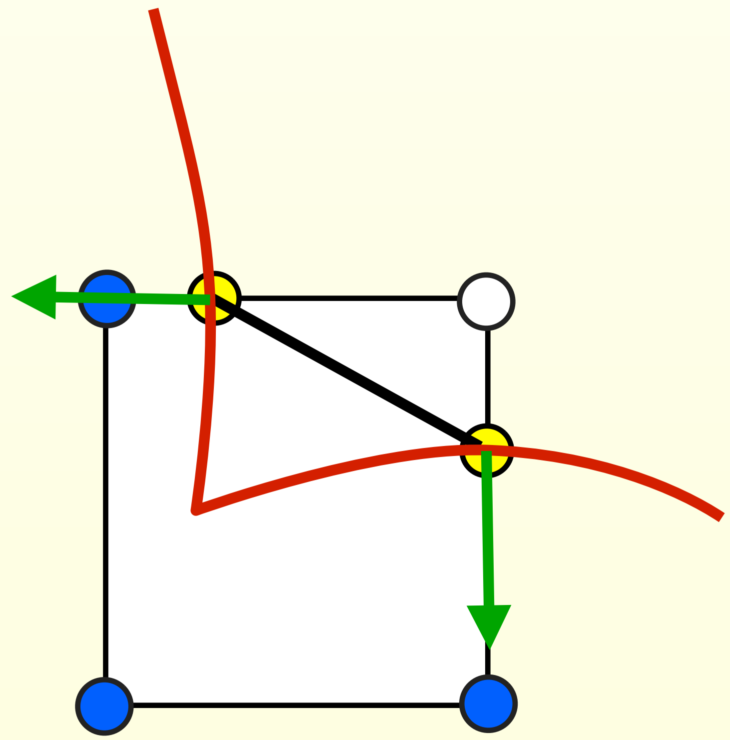
Extended Marching Cubes

- Sharp features are not well approximated



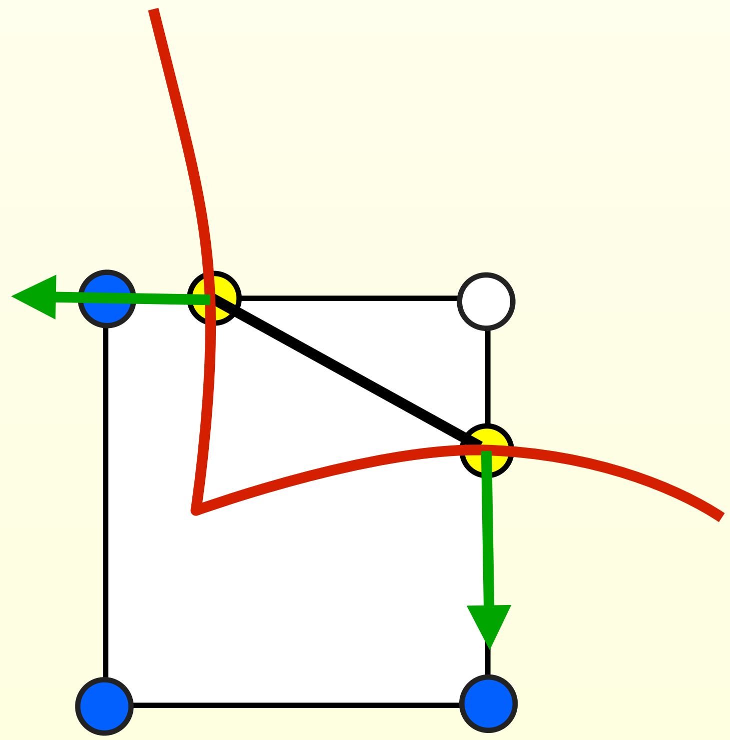
Extended Marching Cubes

- Sharp features are not well approximated
- Use normal information



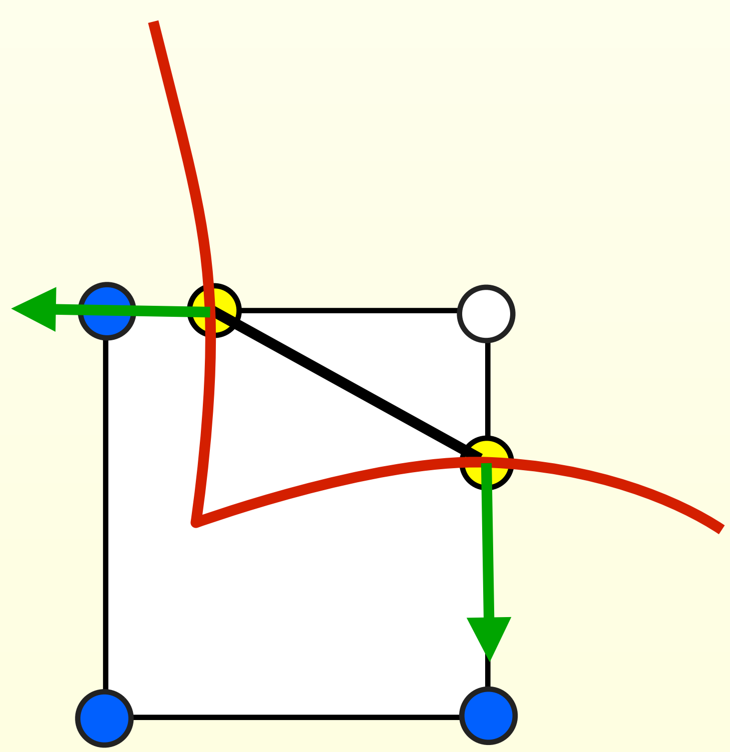
Extended Marching Cubes

- Sharp features are not well approximated
- Use normal information
- Special treatment for corners and edges



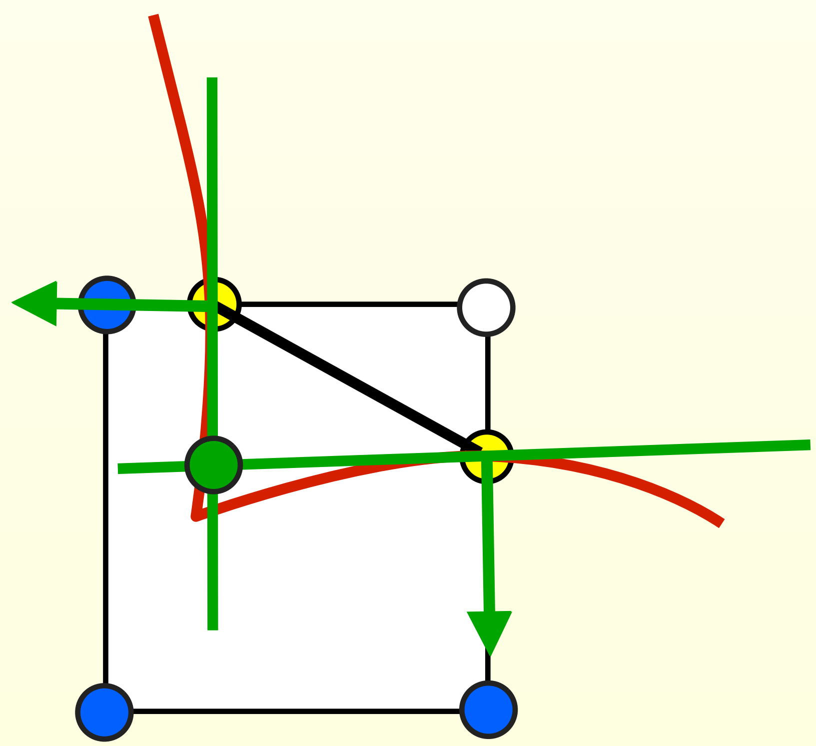
Extended Marching Cubes

- Sharp features are not well approximated
- Use normal information
- Special treatment for corners and edges
  - Corner/edge if large angle between normals



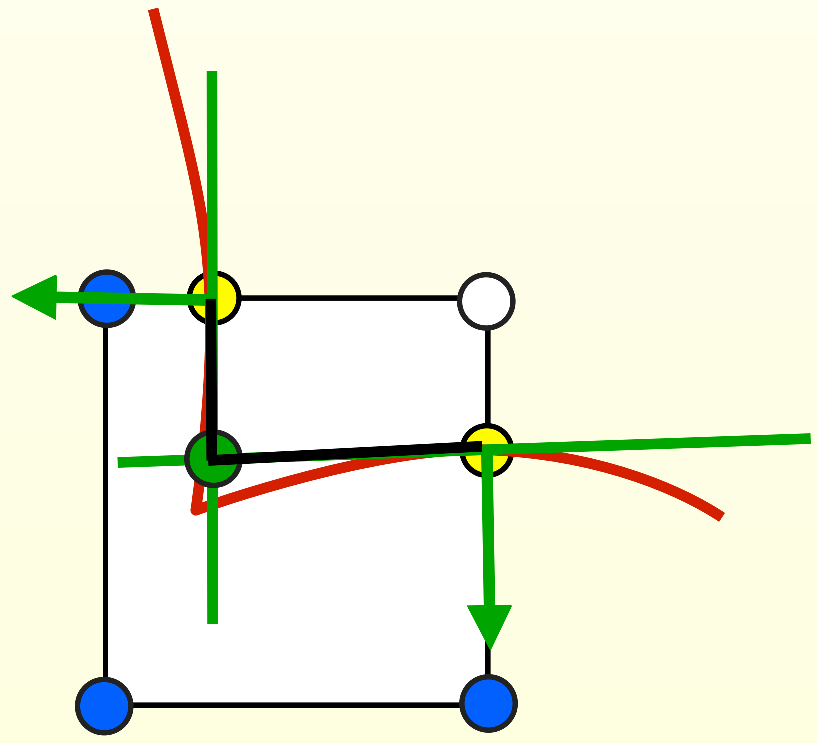
Extended Marching Cubes

- Sharp features are not well approximated
- Use normal information
- Special treatment for corners and edges
  - Corner/edge if large angle between normals
  - Add a vertex at the intersection of tangent planes



Extended Marching Cubes

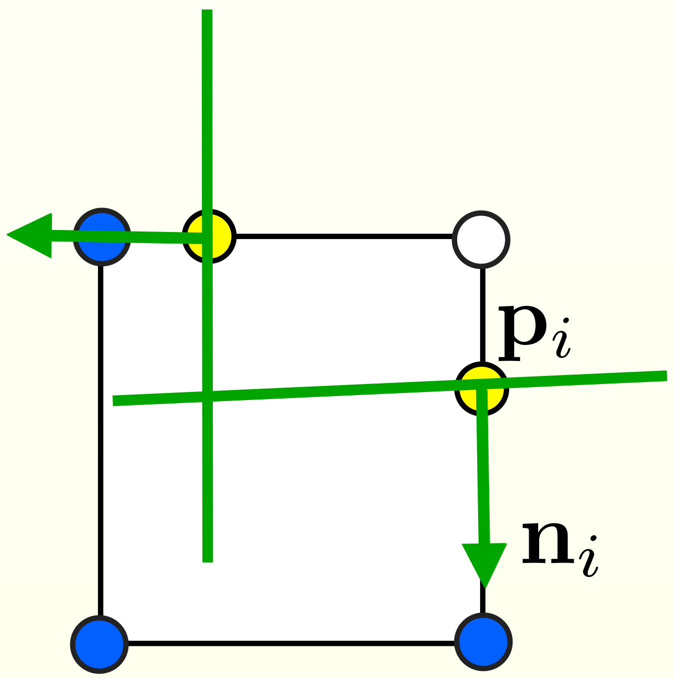
- Sharp features are not well approximated
- Use normal information
- Special treatment for corners and edges
  - Corner/edge if large angle between normals
  - Add a vertex at the intersection of tangent planes



Extended Marching Cubes

- Computing the intersection

$$\mathbf{n}_i \mathbf{x} = \mathbf{n}_i \mathbf{p}_i$$

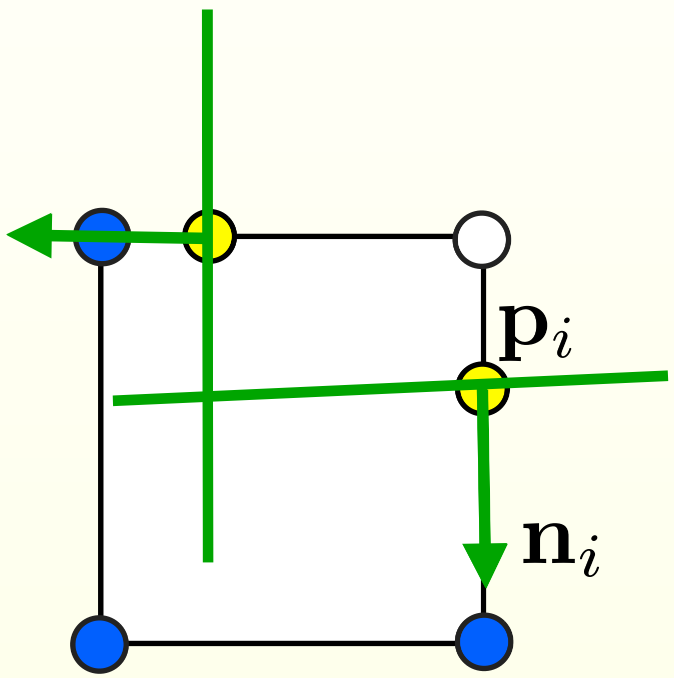


- Sometimes underdetermined
  - When planes are (almost) parallel
- Sometimes overdetermined
  - When too many planes

Extended Marching Cubes

- Computing the intersection

$$\begin{bmatrix} \mathbf{n}_1^T \\ \vdots \\ \mathbf{n}_k^T \end{bmatrix} \mathbf{x} = \begin{bmatrix} \mathbf{n}_1^T \mathbf{p}_1 \\ \vdots \\ \mathbf{n}_k^T \mathbf{p}_k \end{bmatrix}$$

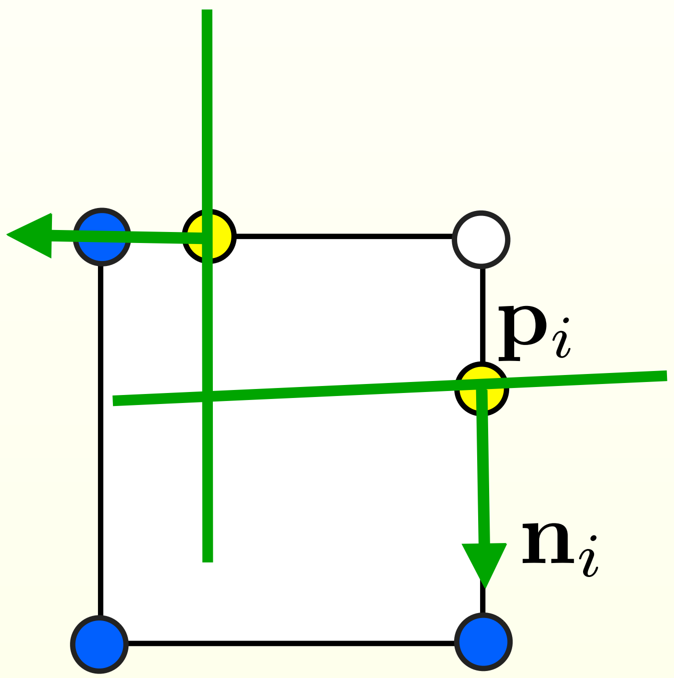


- Sometimes underdetermined
  - When planes are (almost) parallel
- Sometimes overdetermined
  - When too many planes

Extended Marching Cubes

- Computing the intersection

$$\mathbf{A}\mathbf{x} = \begin{bmatrix} \mathbf{n}_1^T \\ \vdots \\ \mathbf{n}_k^T \end{bmatrix} \mathbf{x} = \begin{bmatrix} \mathbf{n}_1^T \mathbf{p}_1 \\ \vdots \\ \mathbf{n}_k^T \mathbf{p}_k \end{bmatrix} = \mathbf{b}$$

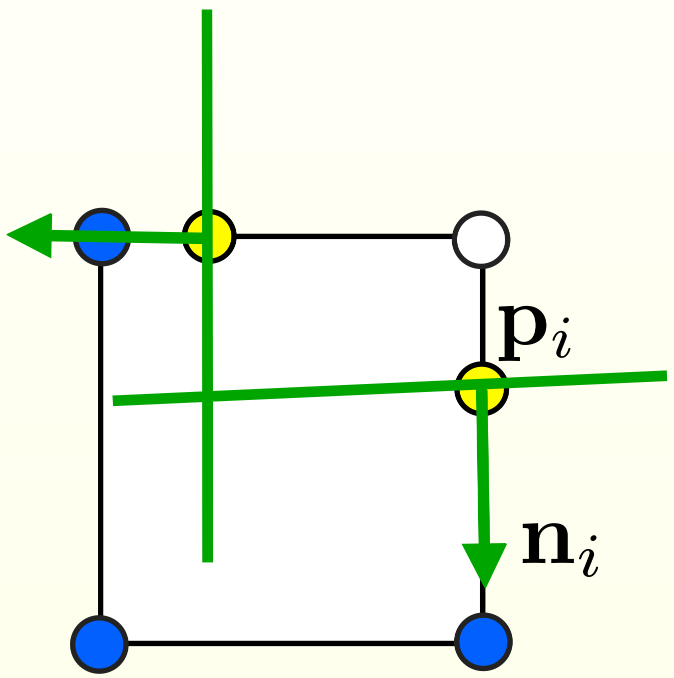


- Sometimes underdetermined
  - When planes are (almost) parallel
- Sometimes overdetermined
  - When too many planes

Extended Marching Cubes

- Computing the intersection

$$\mathbf{A}\mathbf{x} = \begin{bmatrix} \mathbf{n}_1^T \\ \vdots \\ \mathbf{n}_k^T \end{bmatrix} \mathbf{x} = \begin{bmatrix} \mathbf{n}_1^T \mathbf{p}_1 \\ \vdots \\ \mathbf{n}_k^T \mathbf{p}_k \end{bmatrix} = \mathbf{b}$$

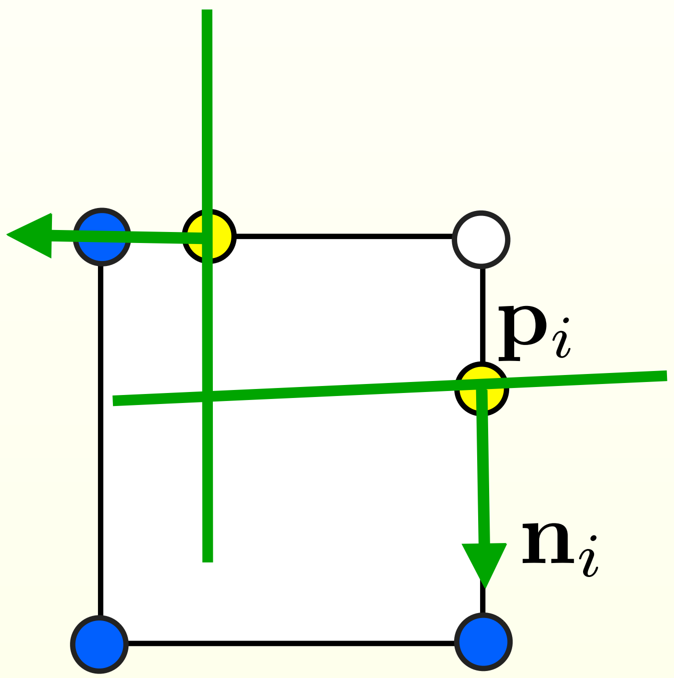


- Sometimes underdetermined
  - When planes are (almost) parallel

Extended Marching Cubes

- Computing the intersection

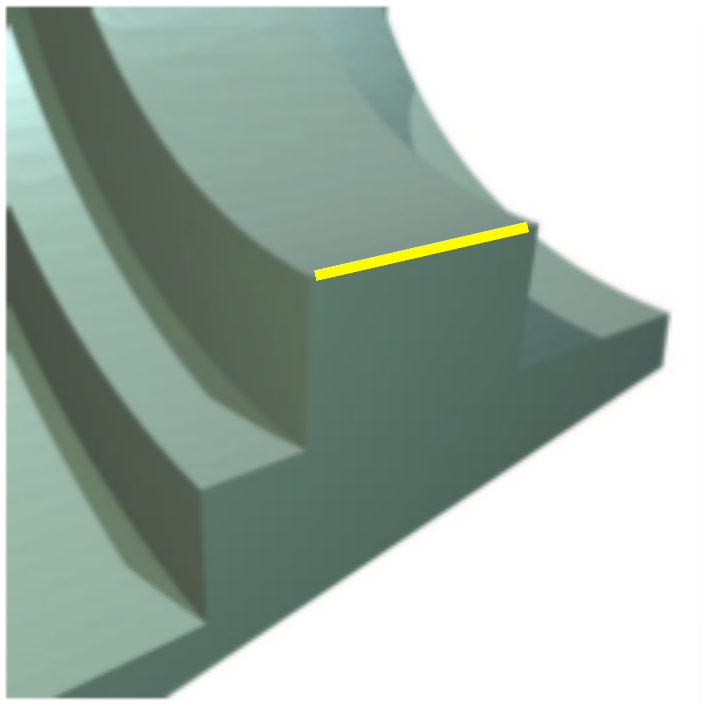
$$\mathbf{A}\mathbf{x} = \begin{bmatrix} \mathbf{n}_1^T \\ \vdots \\ \mathbf{n}_k^T \end{bmatrix} \mathbf{x} = \begin{bmatrix} \mathbf{n}_1^T \mathbf{p}_1 \\ \vdots \\ \mathbf{n}_k^T \mathbf{p}_k \end{bmatrix} = \mathbf{b}$$



- Sometimes underdetermined
  - When planes are (almost) parallel
- Sometimes overdetermined
  - When too many planes

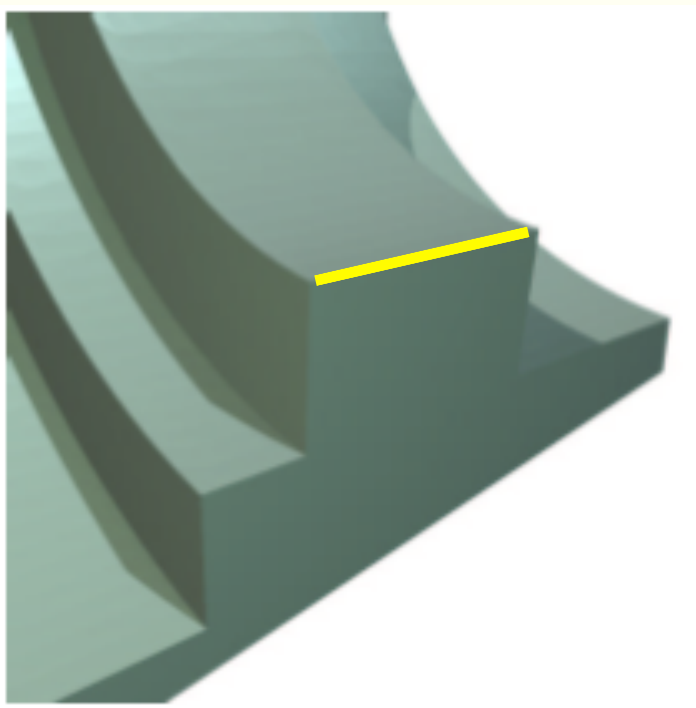
Underdetermined

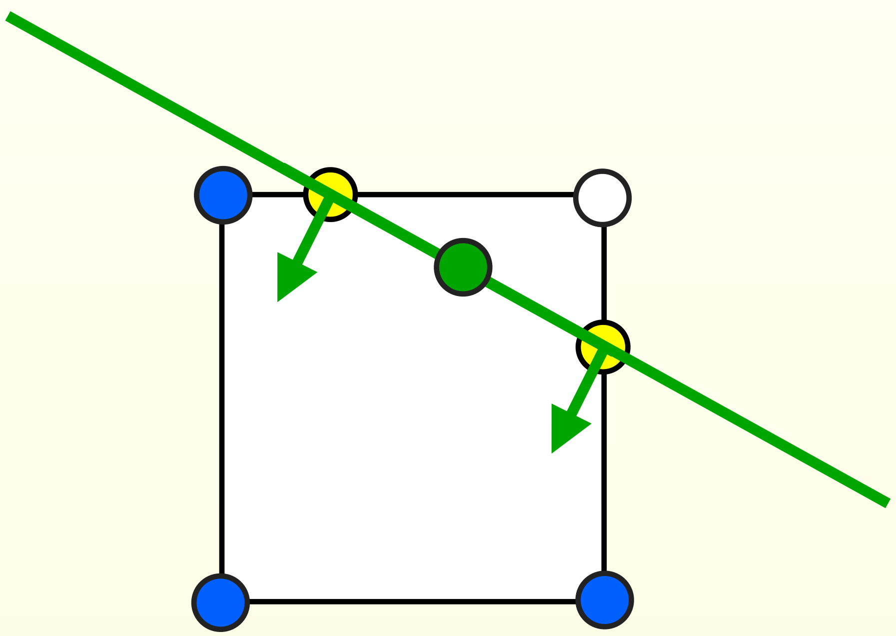
- Many solutions to $\mathbf{Ax} = \mathbf{b}$
- Standard edge case in 3D



Underdetermined

- Many solutions to $\mathbf{Ax} = \mathbf{b}$
- Standard edge case in 3D





- Choose the solution closest to the center of gravity of all intersections

Overdetermined

- Find least-squares solution: minimize $\|\mathbf{Ax} - \mathbf{b}\|^2$

Overdetermined

- Find least-squares solution: minimize $\|\mathbf{Ax} - \mathbf{b}\|^2$

- Compute the pseudo-inverse $\mathbf{A}^+$ using SVD

$$\mathbf{x} = \mathbf{A}^+\mathbf{b} = \mathbf{A}^+[\mathbf{n}_1^T\mathbf{p}_1 \ldots \mathbf{n}_k^T\mathbf{p}_k]^T$$

Overdetermined

- Find least-squares solution: minimize $\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2$
- Compute the pseudo-inverse $\mathbf{A}^+$ using SVD

$$\mathbf{x} = \mathbf{A}^+\mathbf{b} = \mathbf{A}^+[\mathbf{n}_1^T\mathbf{p}_1 \ldots \mathbf{n}_k^T\mathbf{p}_k]^T$$

- Solution for under- and over-determined system

$$\mathbf{A}\mathbf{y} = [\ldots \mathbf{n}_i^T(\mathbf{p}_i - \sum_i \mathbf{p}_i/k) \ldots]$$

$$\mathbf{x} = \mathbf{A}^+([\ldots \mathbf{n}_i(\mathbf{p}_i - \sum_i \mathbf{p}_i/k) \ldots]) + \sum_i \mathbf{p}_i/k$$

Overdetermined

- Find least-squares solution: minimize $\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2$
- Compute the pseudo-inverse $\mathbf{A}^+$ using SVD

$$\mathbf{x} = \mathbf{A}^+\mathbf{b} = \mathbf{A}^+[\mathbf{n}_1^T\mathbf{p}_1 \ldots \mathbf{n}_k^T\mathbf{p}_k]^T$$

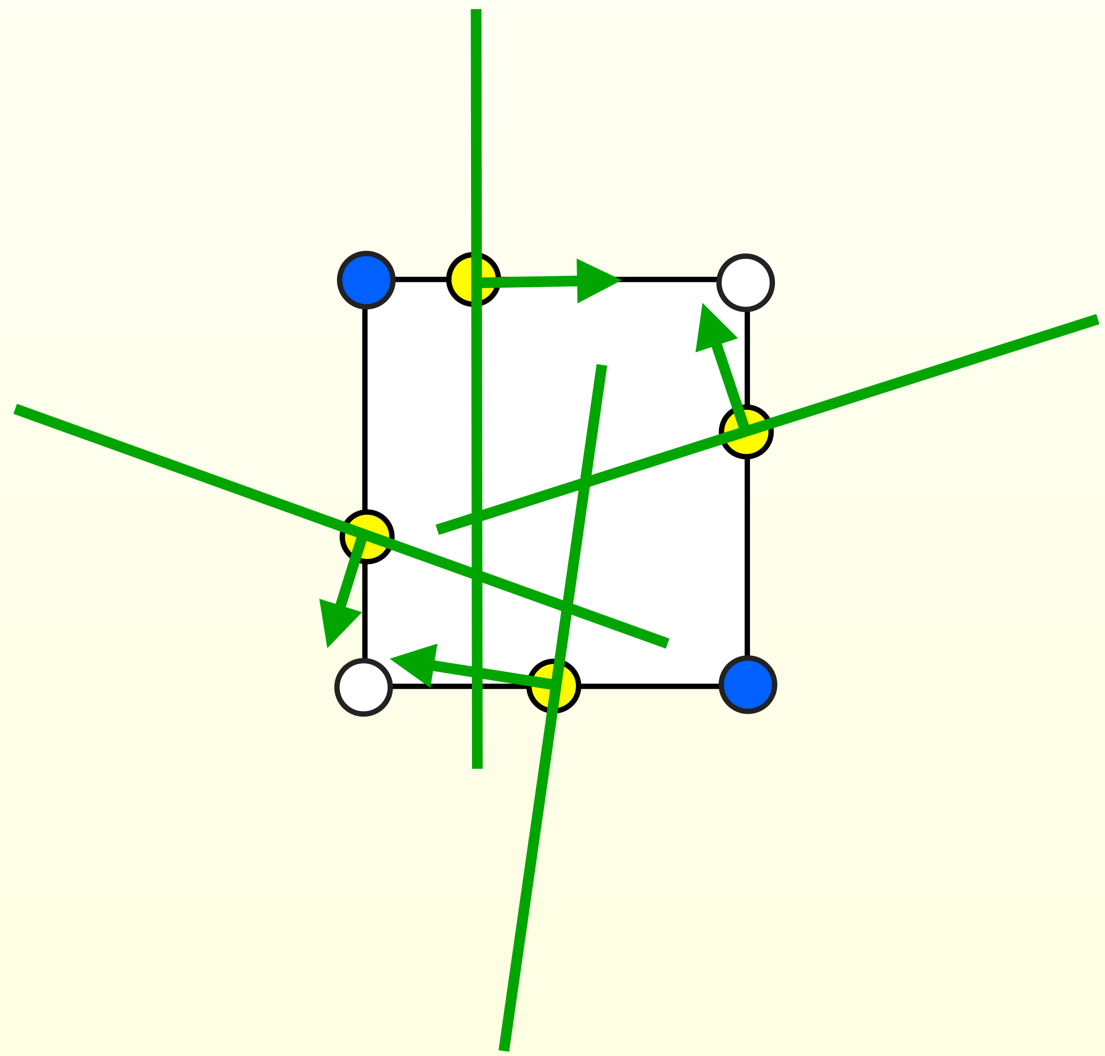
- Solution for under- and over-determined system

$$\mathbf{A}\mathbf{y} = [\ldots \mathbf{n}_i^T(\mathbf{p}_i - \sum_i \mathbf{p}_i/k) \ldots]$$

$$\mathbf{x} = \mathbf{A}^+([\ldots \mathbf{n}_i(\mathbf{p}_i - \sum_i \mathbf{p}_i/k) \ldots]) + \sum_i \mathbf{p}_i/k$$

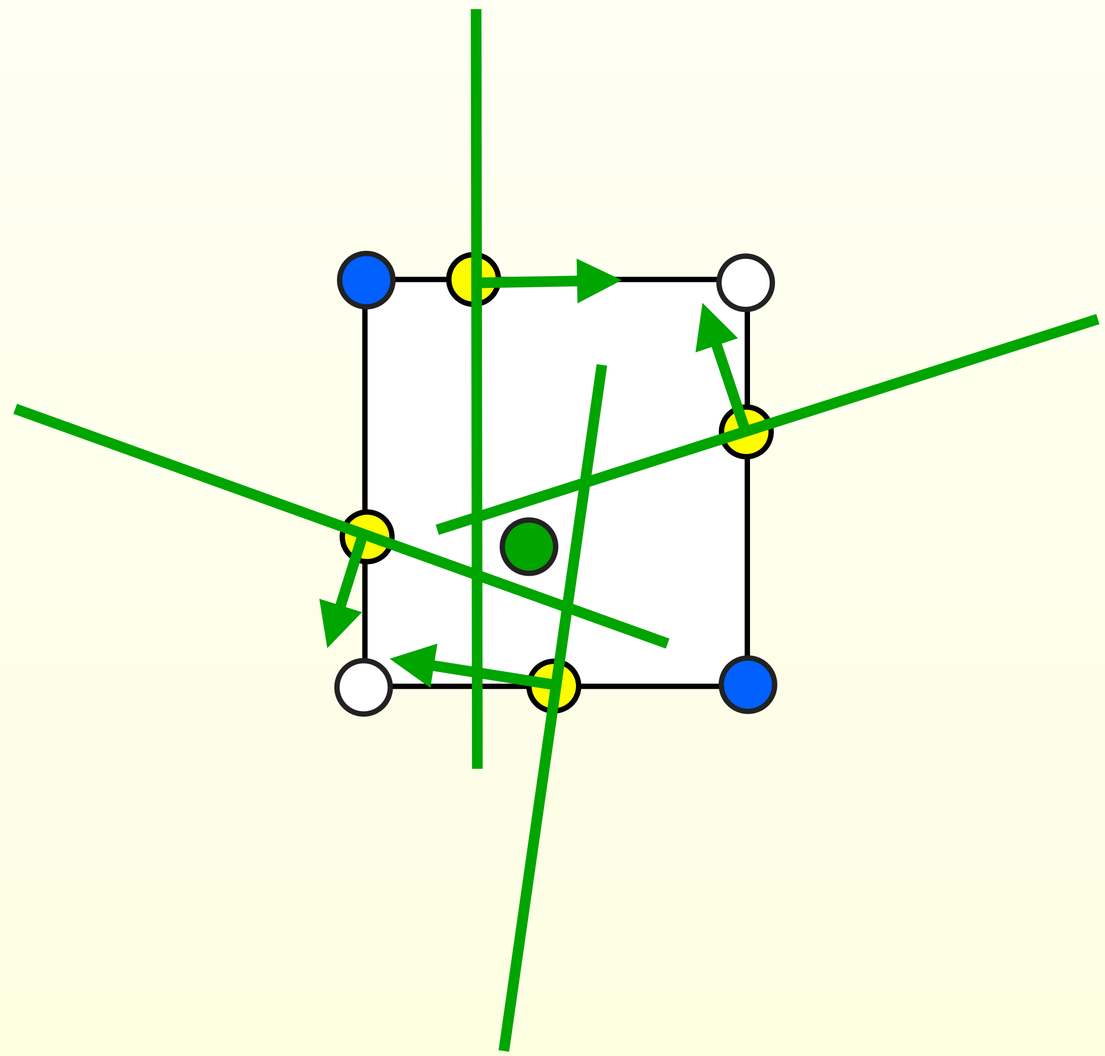
Extended Marching Cubes

- No ambiguous cases



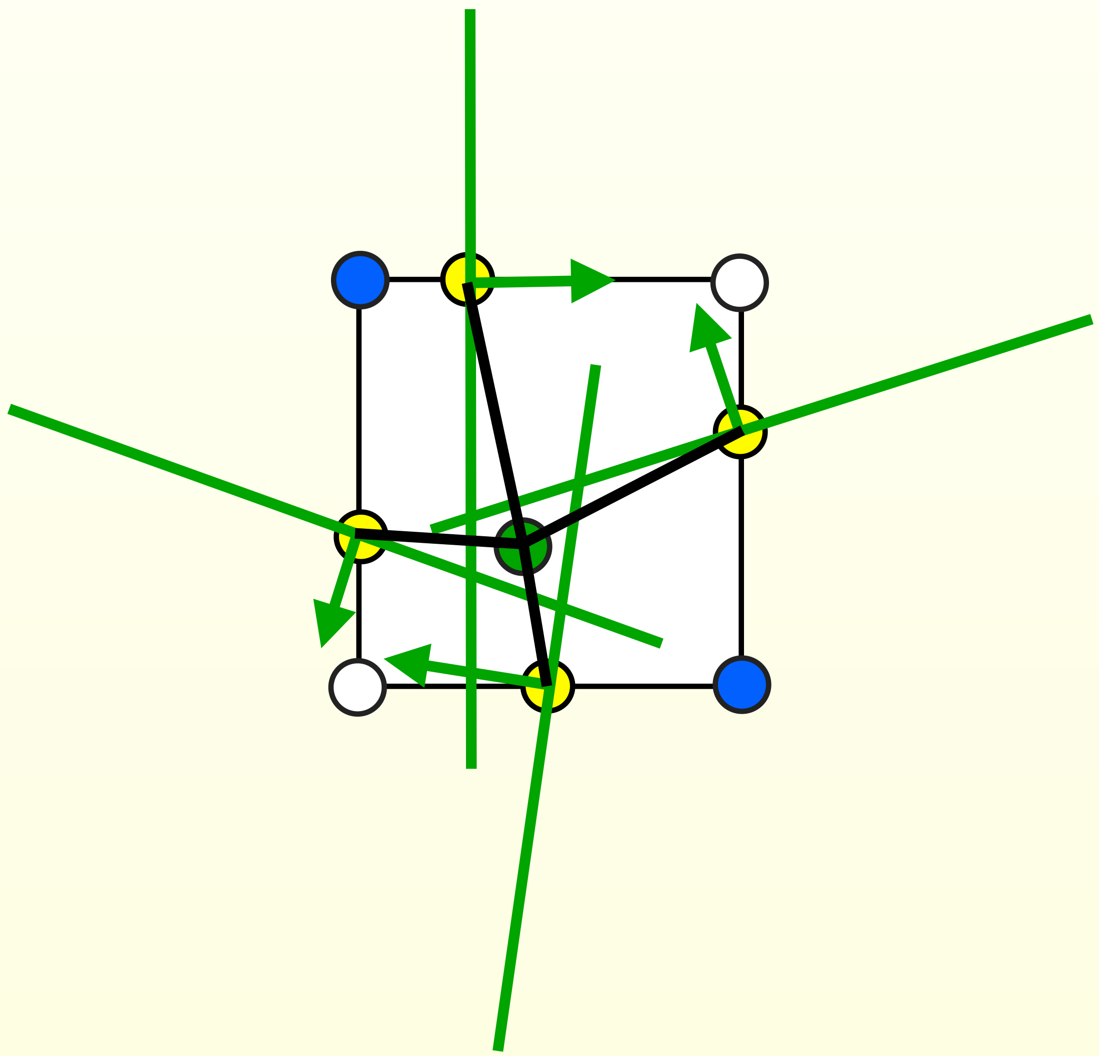
Extended Marching Cubes

- No ambiguous cases



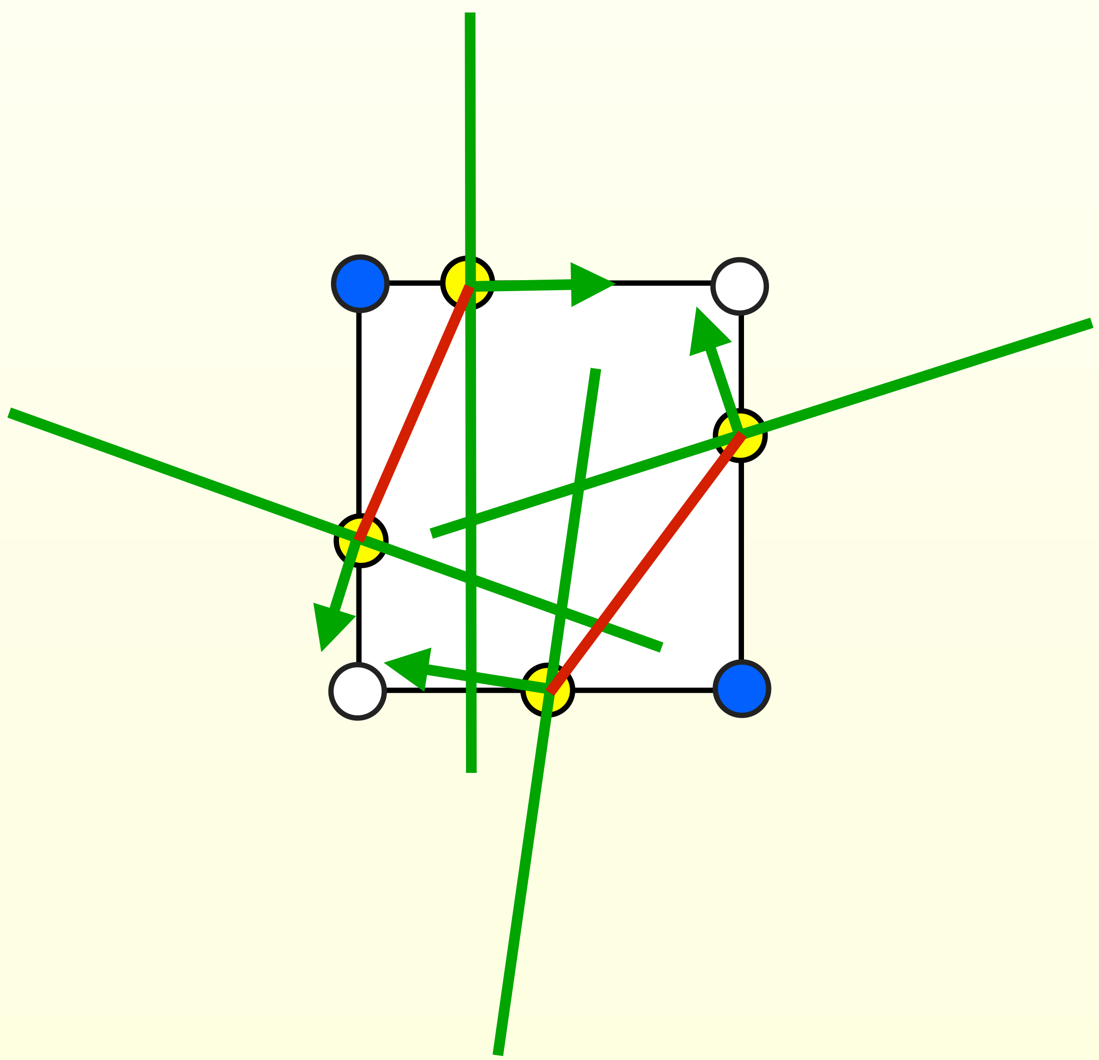
Extended Marching Cubes

- No ambiguous cases



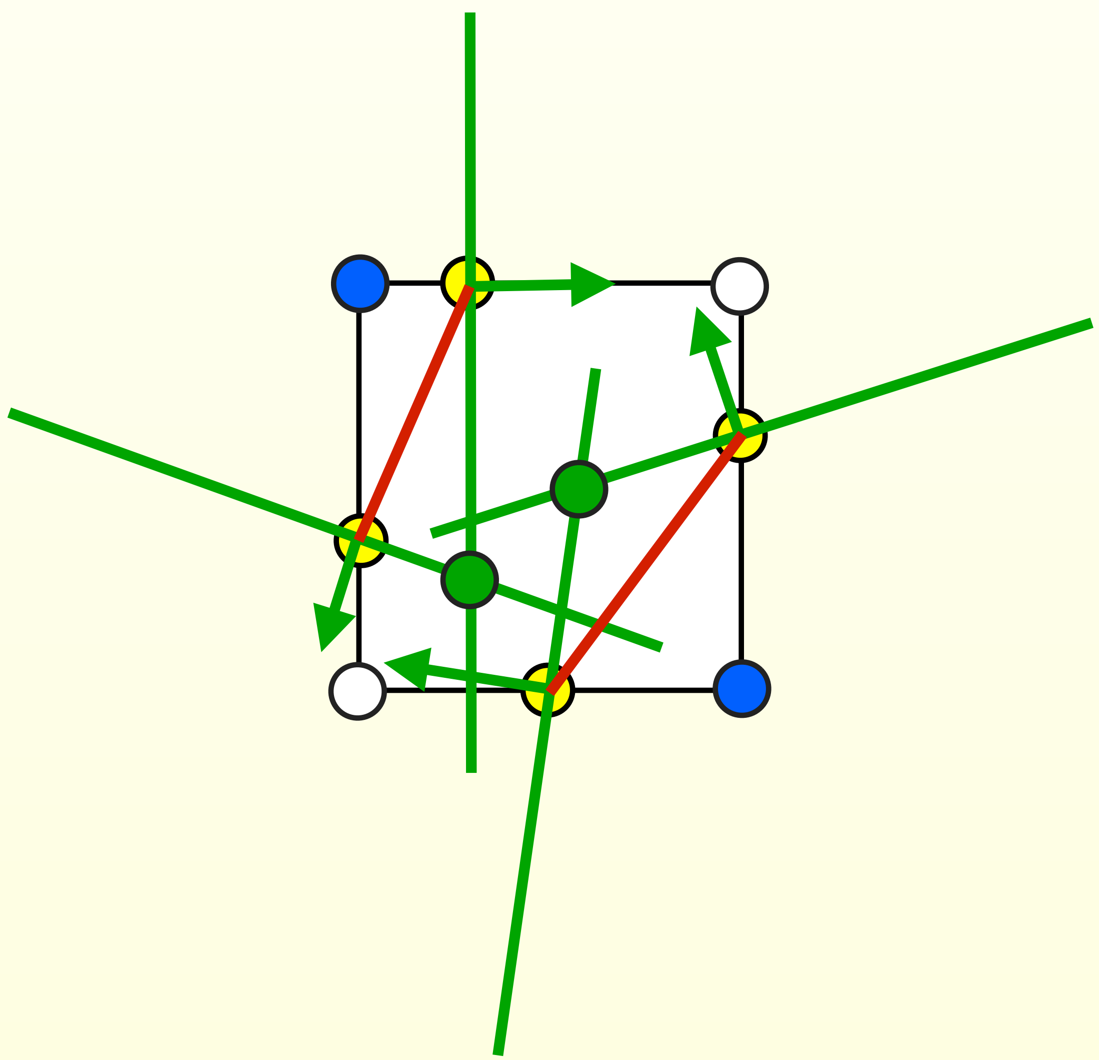
Extended Marching Cubes

- Reintroduce ambiguity:
  - Use the regular marching cubes connectivity
  - Compute several points



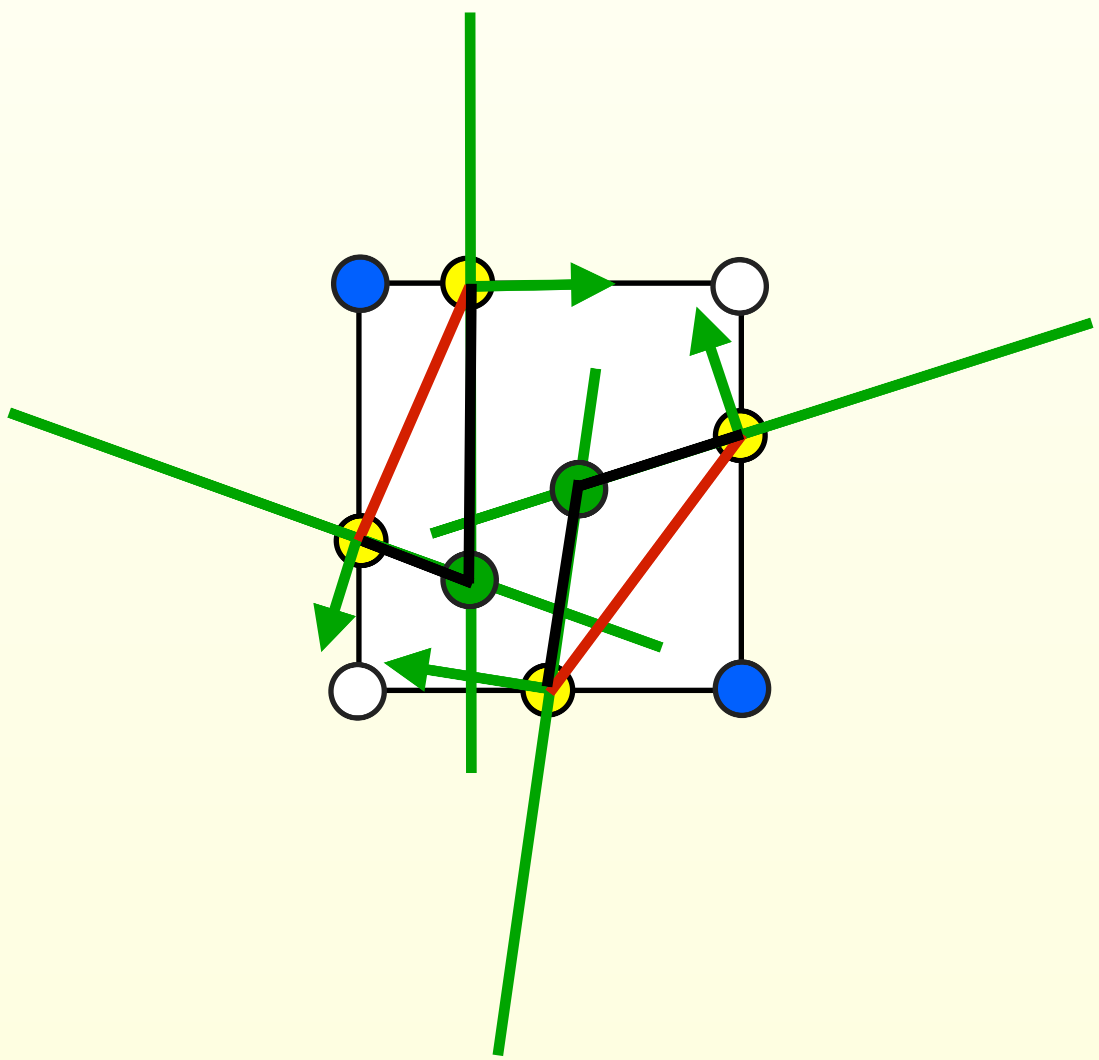
Extended Marching Cubes

- Reintroduce ambiguity:
  - Use the regular marching cubes connectivity
  - Compute several points

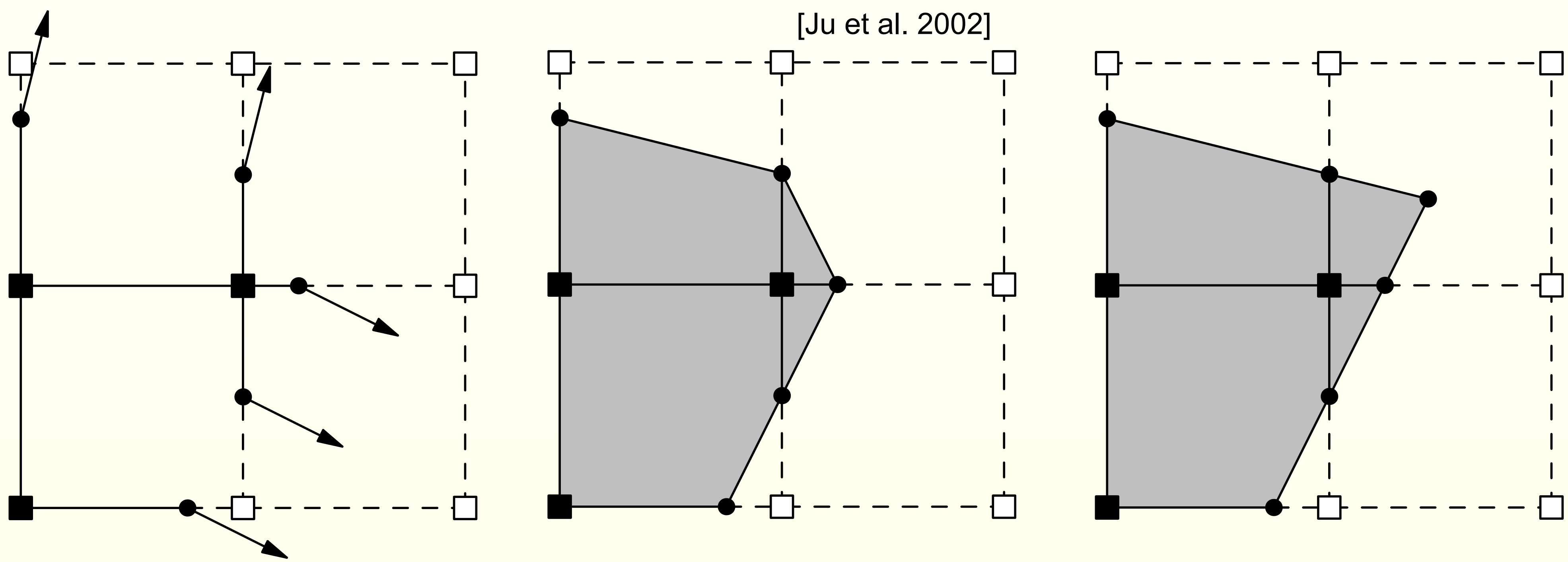


Extended Marching Cubes

- Reintroduce ambiguity:
  - Use the regular marching cubes connectivity
  - Compute several points



Comparison



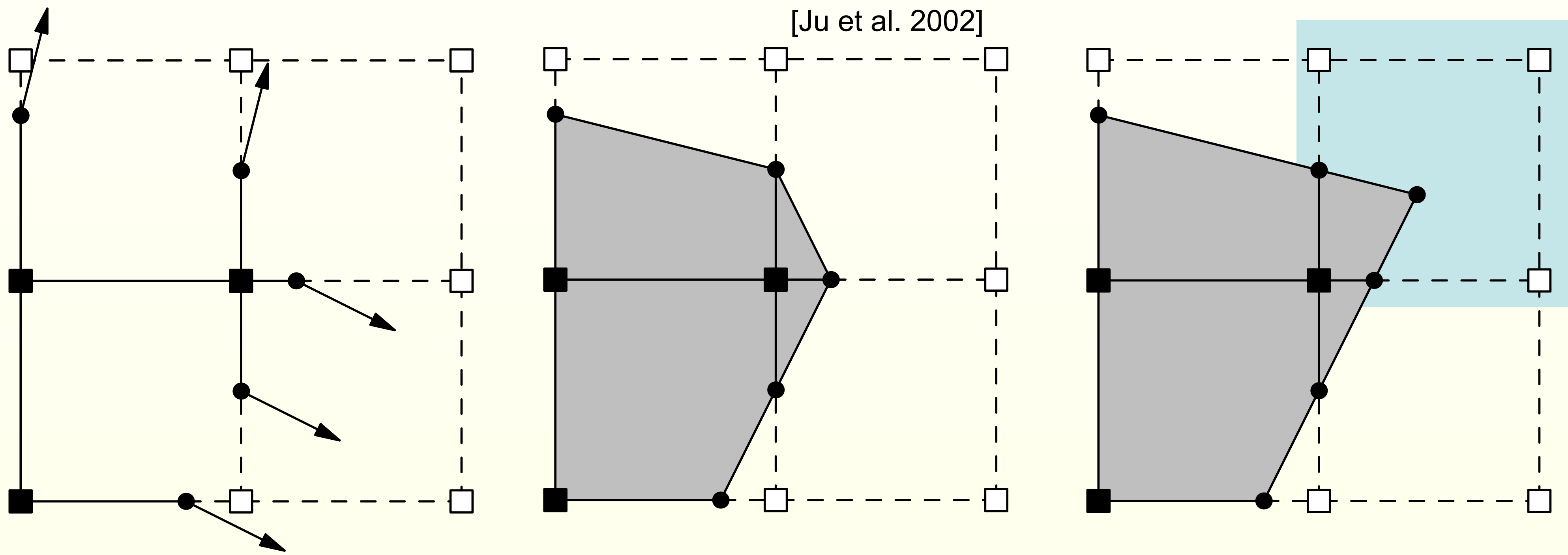
Input data

Marching Cubes

Extended Marching Cubes

Comparison

[Ju et al. 2002]



Input data

Marching Cubes

Extended Marching Cubes

Dual Methods

- Marching Cubes (Primal)
  - 1 point per edge
  - Connecting primitives (triangles) per voxel

- Dual Contouring (Dual)
  - 1 point per voxel
  - Connecting primitives (quads) per edge

Dual Methods

- Marching Cubes (Primal)
  - 1 point per edge
  - Connecting primitives (triangles) per voxel
- Dual Contouring (Dual)
  - 1 point per voxel
  - Connecting primitives (quads) per edge

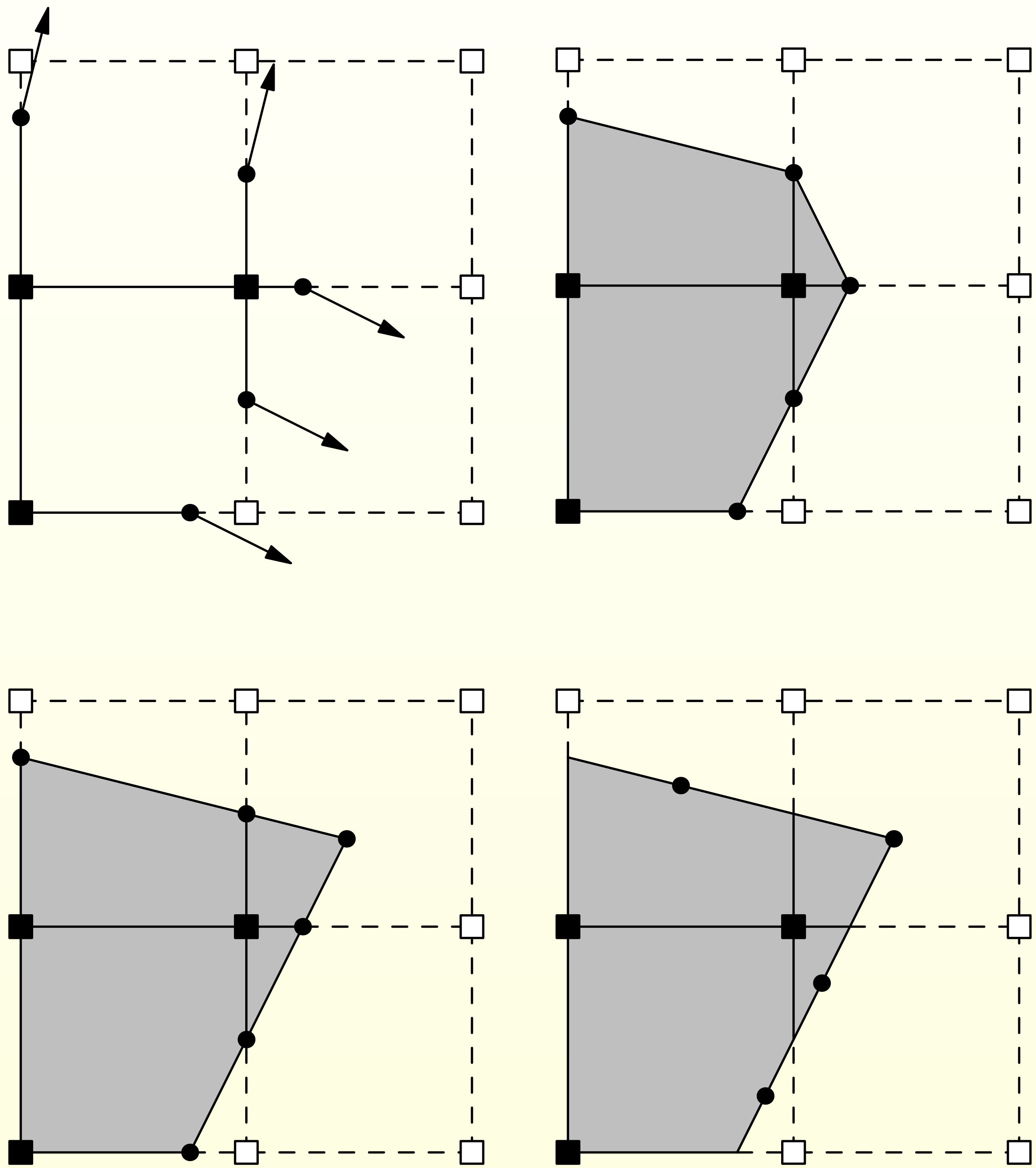
Dual Methods

- Marching Cubes (Primal)
  - 1 point per edge
  - Connecting primitives (triangles) per voxel
- Dual Contouring (Dual)
  - 1 point per voxel
  - Connecting primitives (quads) per edge

Comparison

Input data

Marching Cubes



Extended Marching Cubes

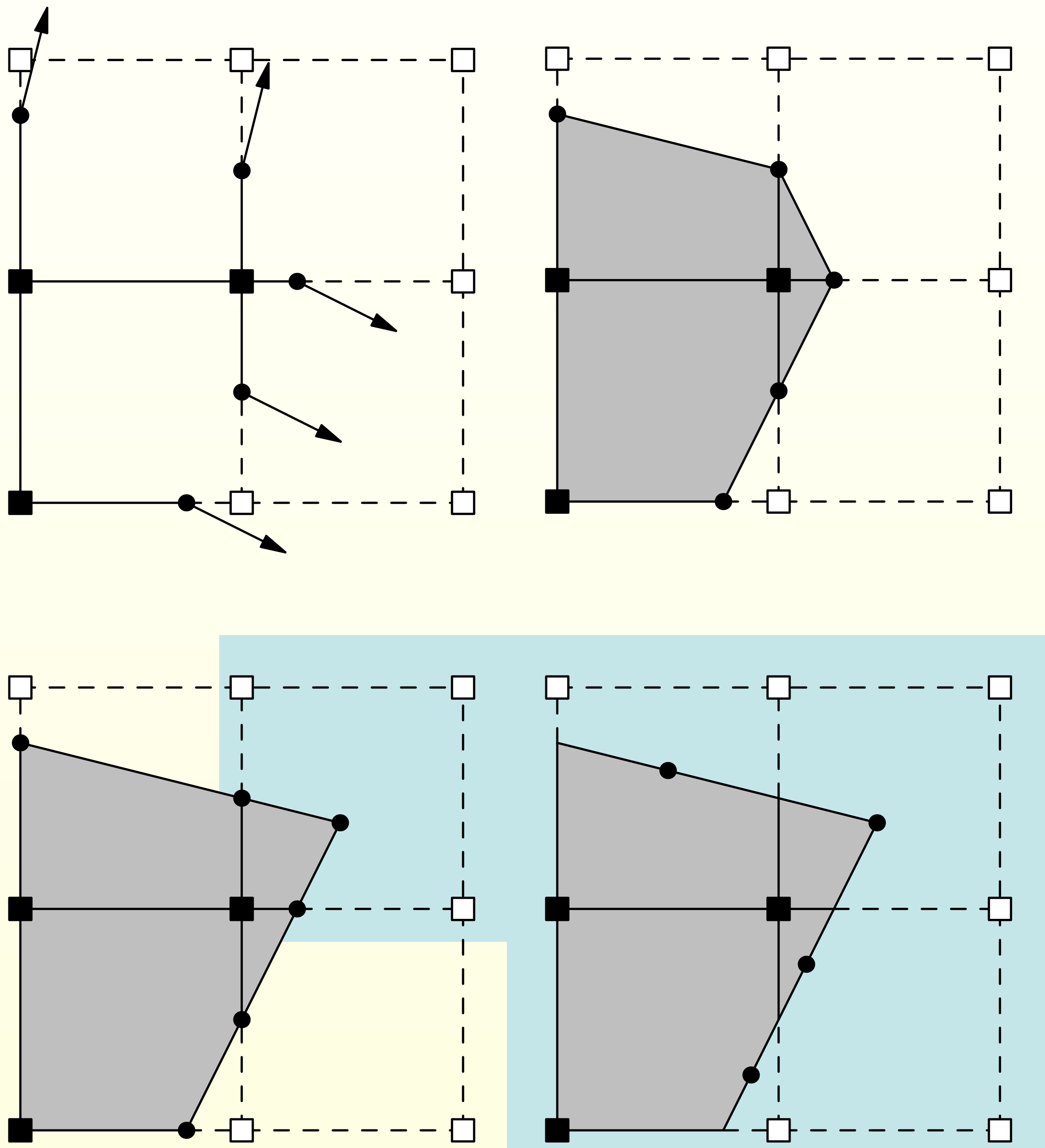
Dual Contouring

[Ju et al. 2002]

Comparison

Input data

Marching Cubes



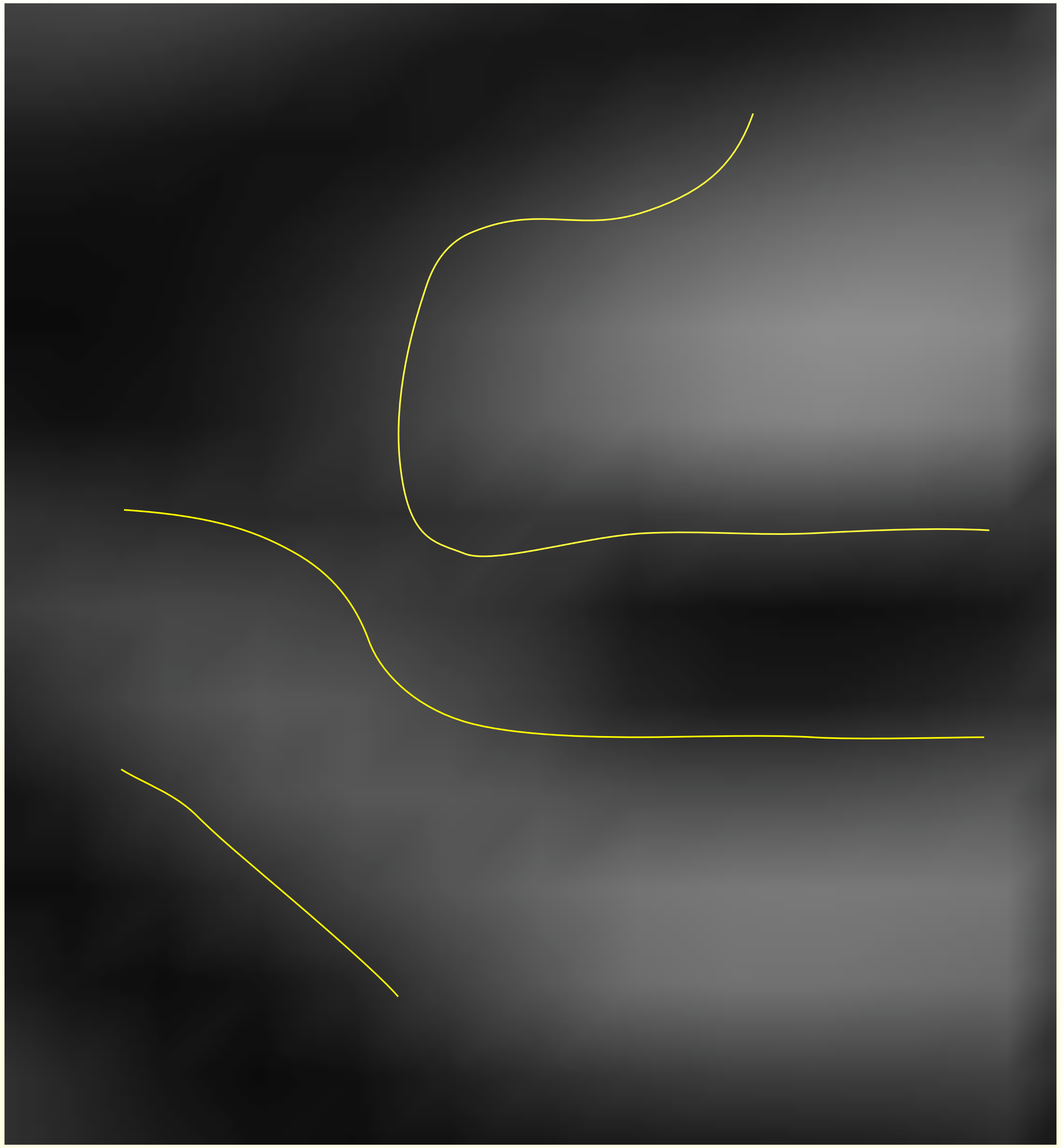
Extended Marching Cubes

Dual Contouring

[Ju et al. 2002]

Marching Squares

- Given $I(\mathbf{x})$
  - $I(\mathbf{x}) < q$: $\mathbf{x}$ outside
  - $I(\mathbf{x}) > q$: $\mathbf{x}$ inside
- Discretize space
- Evaluate on the grid

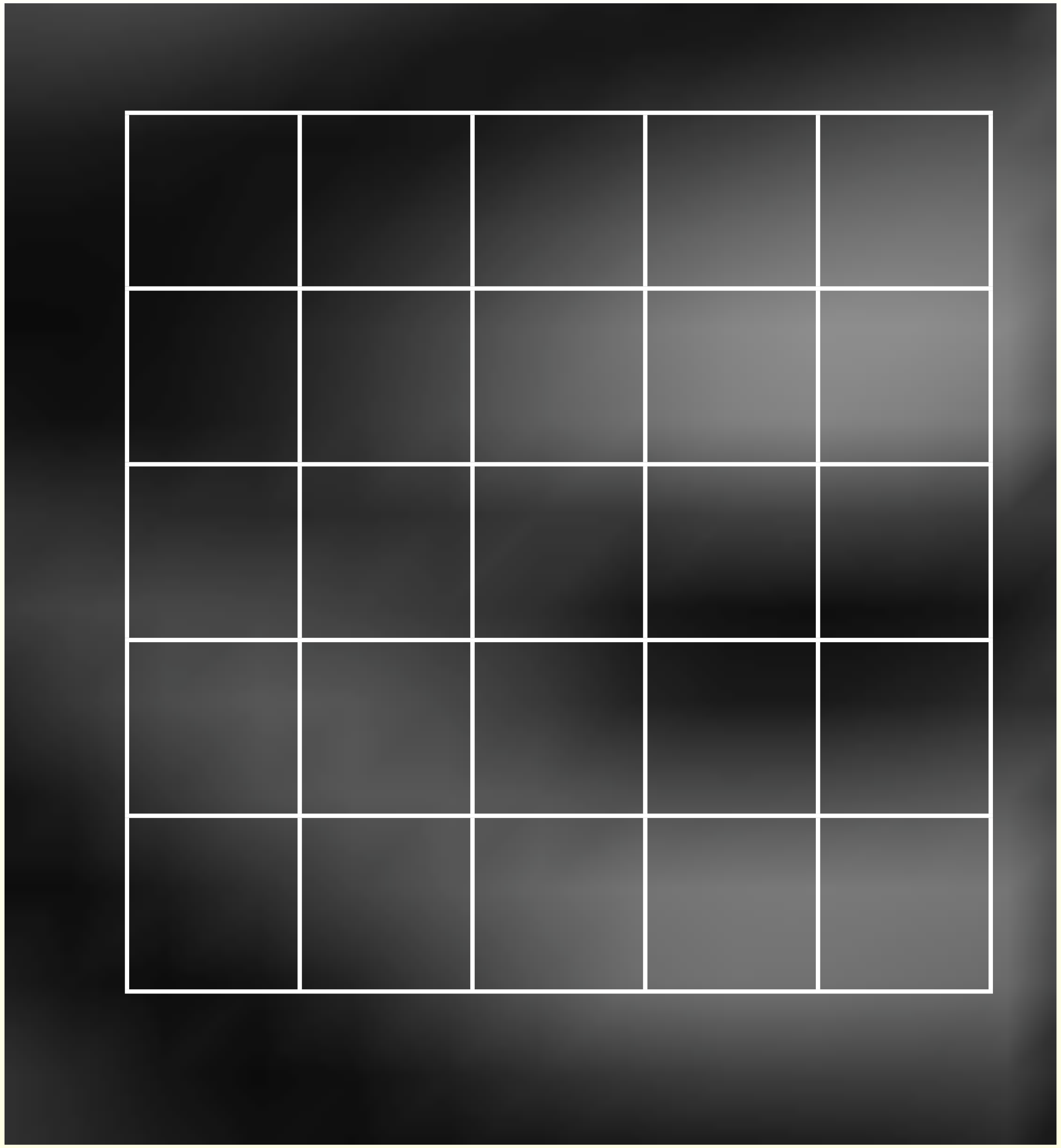


Marching Squares

- Given $I(\mathbf{x})$
  - $I(\mathbf{x}) < q$: $\mathbf{x}$ outside
  - $I(\mathbf{x}) > q$: $\mathbf{x}$ inside
- Discretize space
- Evaluate on the grid

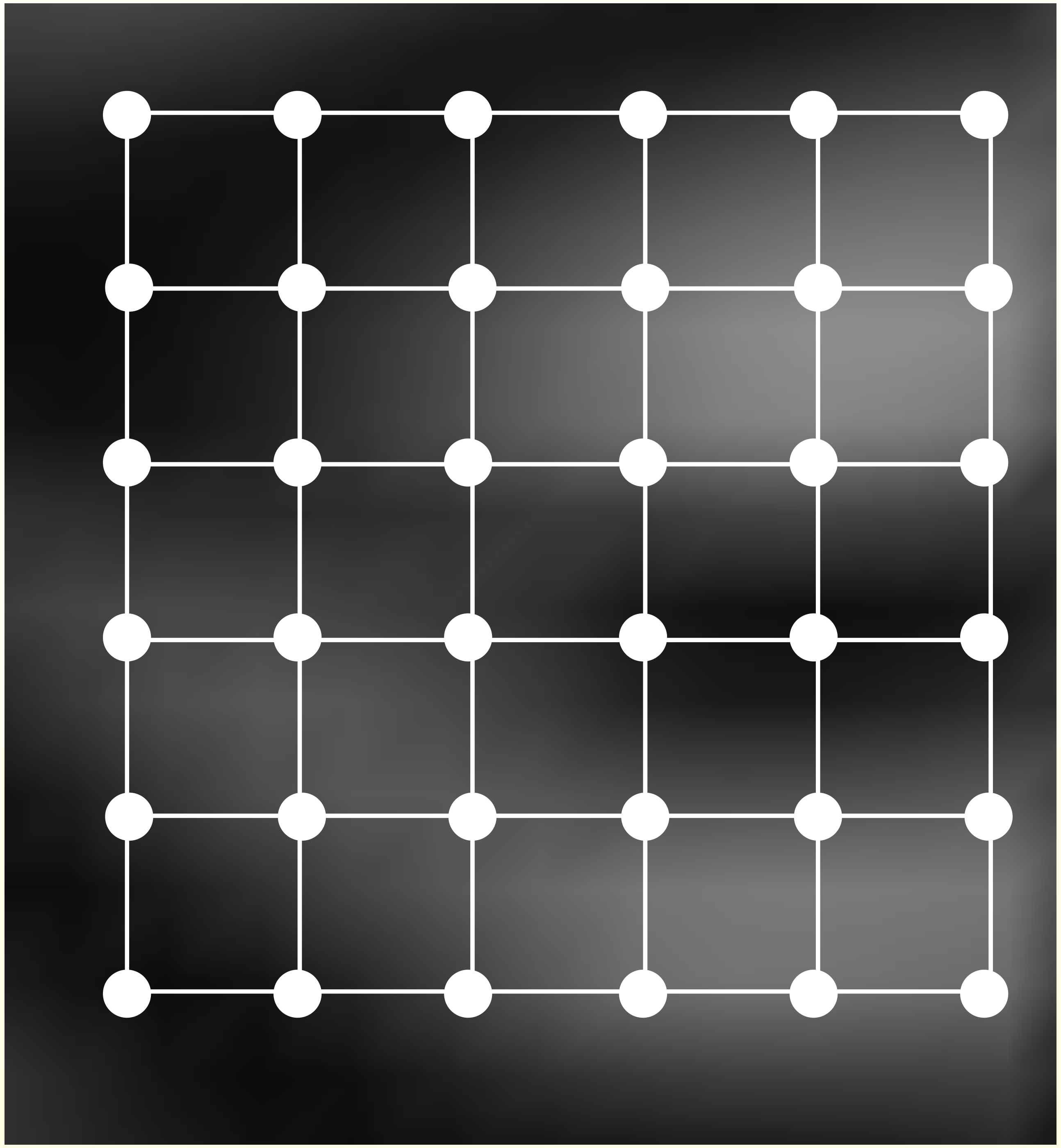
Marching Squares

- Given $I(\mathbf{x})$
  - $I(\mathbf{x}) < q$: $\mathbf{x}$ outside
  - $I(\mathbf{x}) > q$: $\mathbf{x}$ inside
- Discretize space



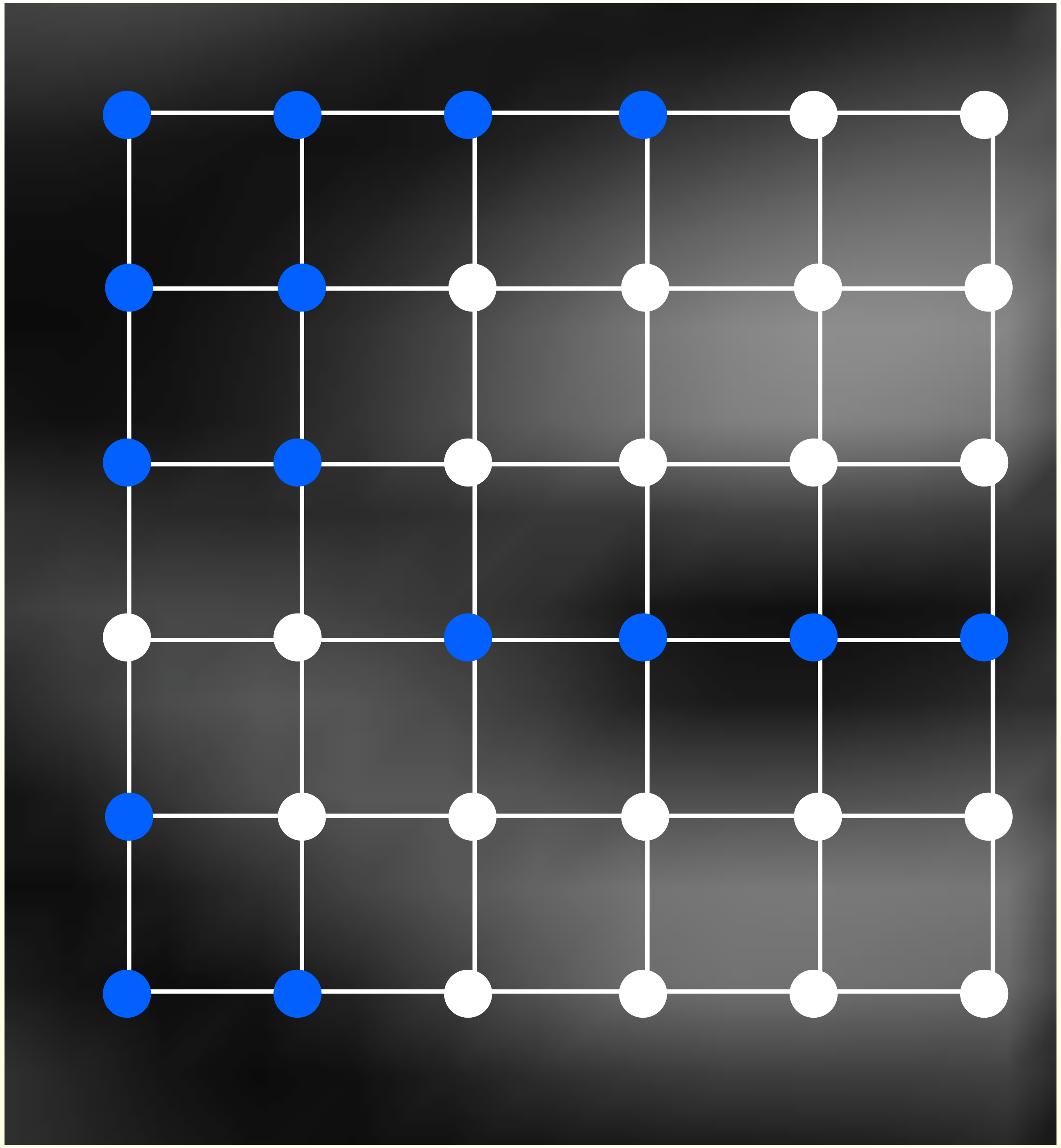
Marching Squares

- Given $I(\mathbf{x})$
  - $I(\mathbf{x}) < q : \mathbf{x}$ outside
  - $I(\mathbf{x}) > q : \mathbf{x}$ inside
- Discretize space
- Evaluate $f(\mathbf{x})$ on the grid



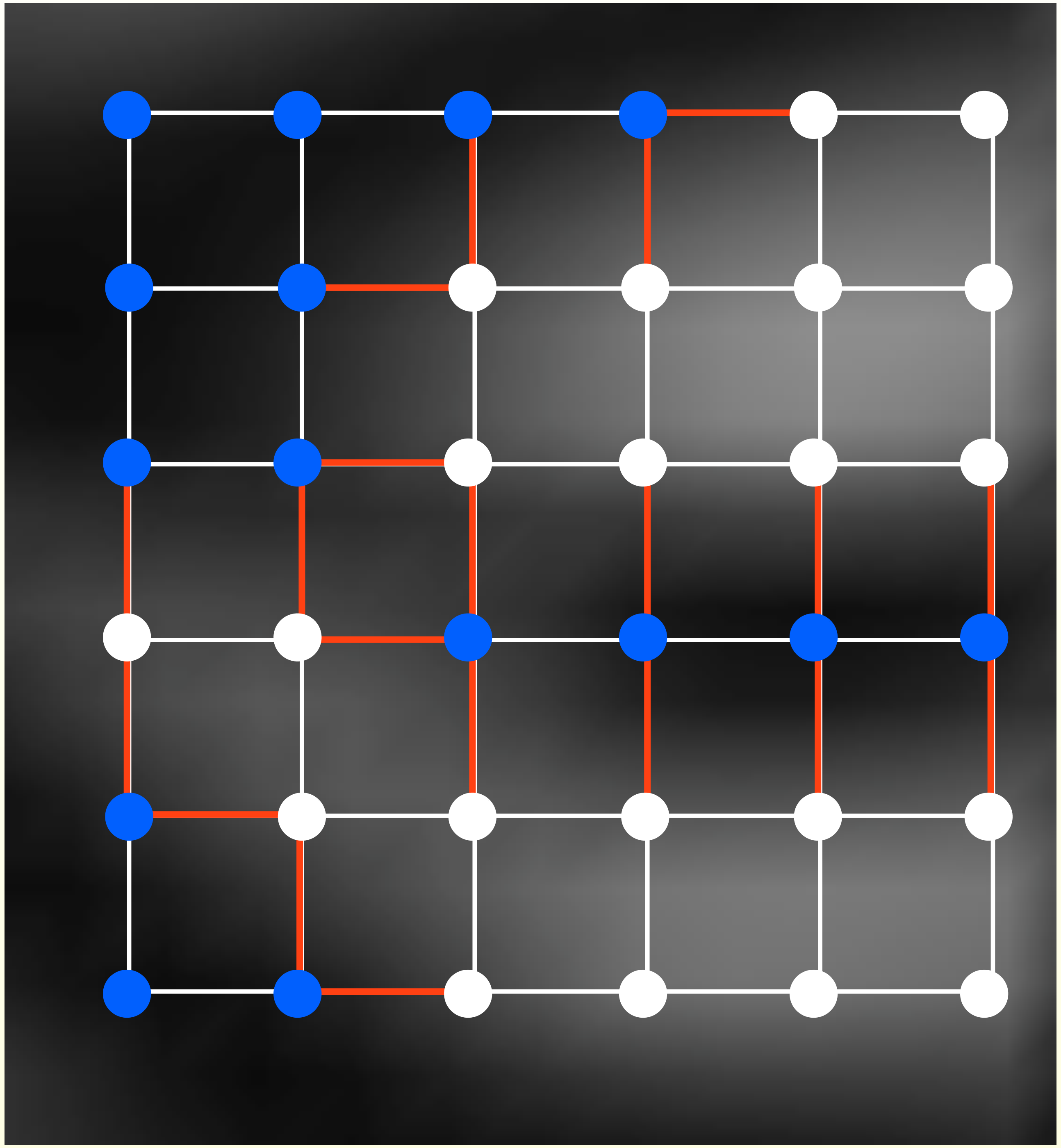
2D Dual Contouring

- Given $I(\mathbf{x})$
  - $I(\mathbf{x}) < q$: $\mathbf{x}$ outside
  - $I(\mathbf{x}) > q$: $\mathbf{x}$ inside
- Discretize space
- Evaluate $f(\mathbf{x})$ on the grid
- Classify grid points



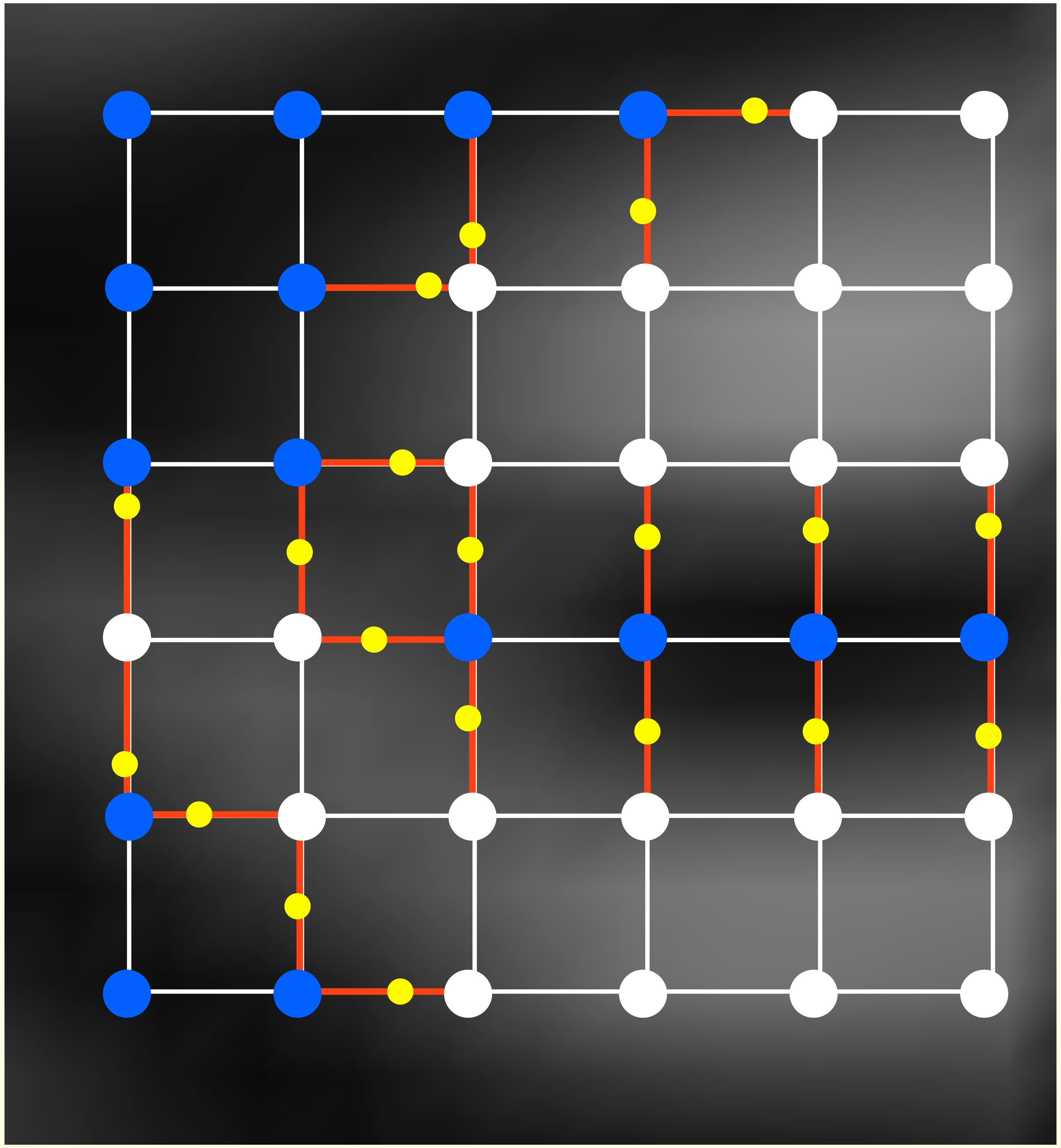
2D Dual Contouring

- Given $I(\mathbf{x})$
  - $I(\mathbf{x}) < q$: $\mathbf{x}$ outside
  - $I(\mathbf{x}) > q$: $\mathbf{x}$ inside
- Discretize space
- Evaluate $f(\mathbf{x})$ on the grid
- Classify grid points
- Classify grid edges



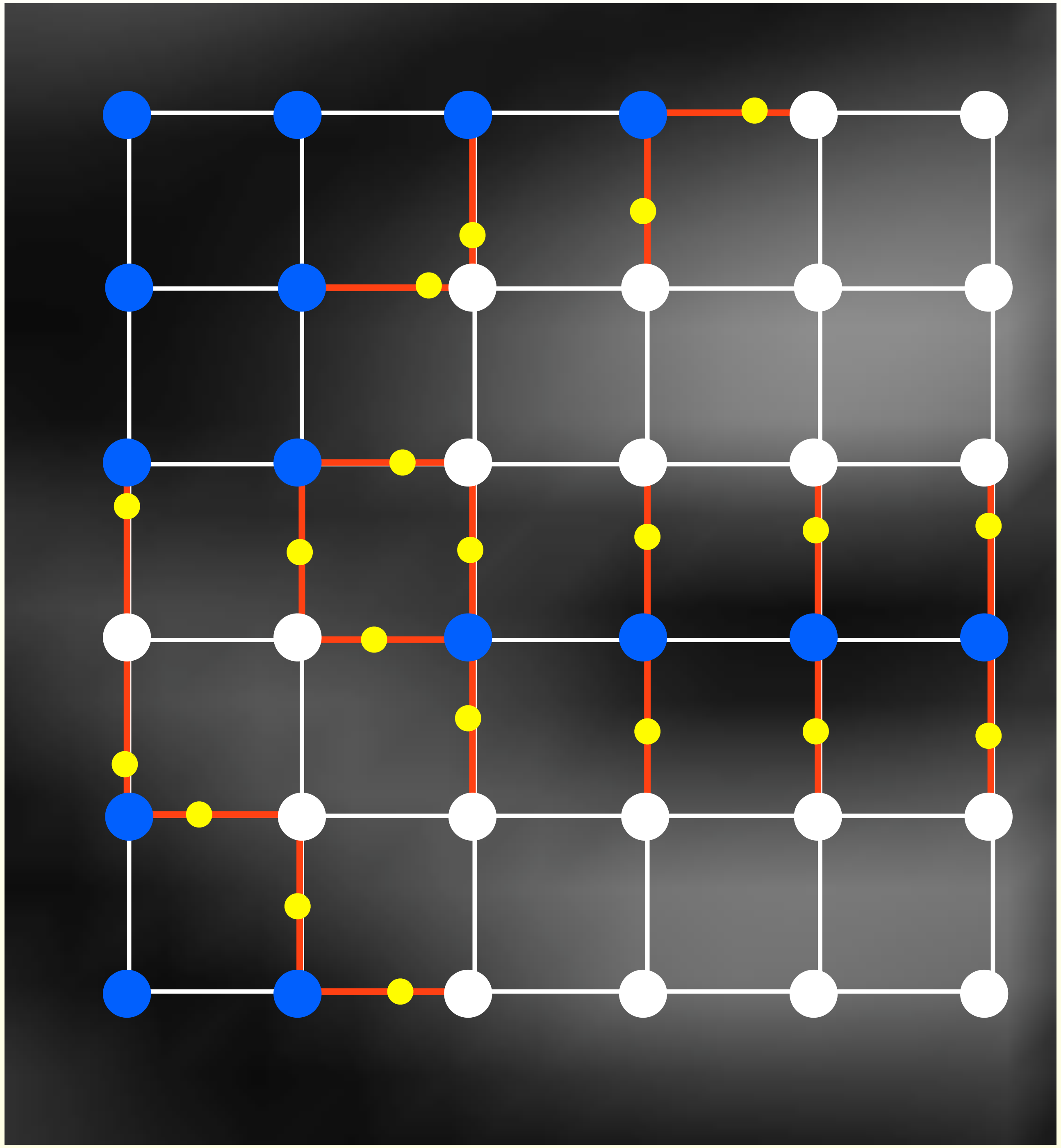
2D Dual Contouring

- Given $I(\mathbf{x})$
  - $I(\mathbf{x}) < q$: $\mathbf{x}$ outside
  - $I(\mathbf{x}) > q$: $\mathbf{x}$ inside
- Discretize space
- Evaluate $f(\mathbf{x})$ on the grid
- Classify grid points
- Classify grid edges
- Compute intersections



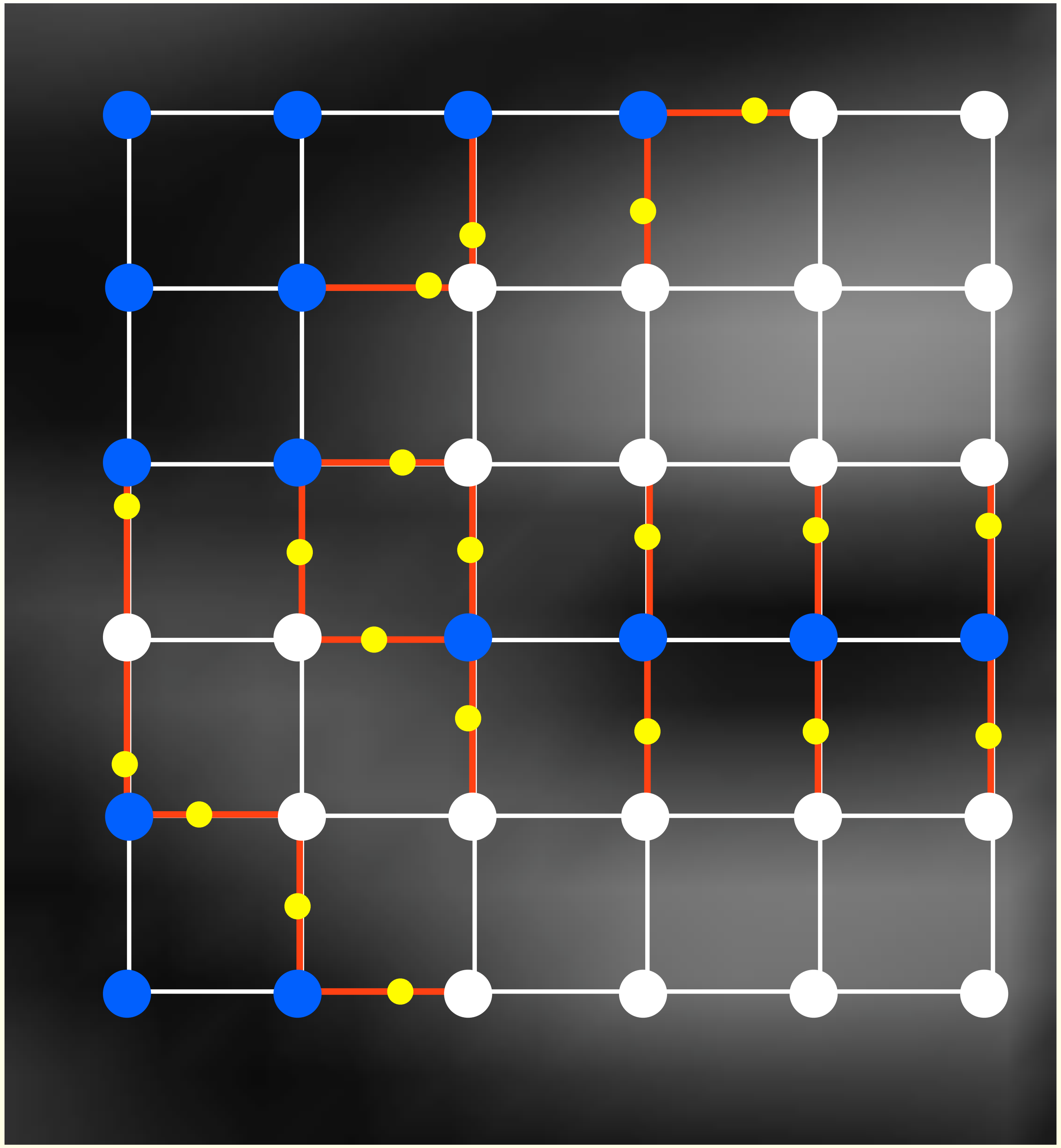
2D Dual Contouring

- Compute intersections



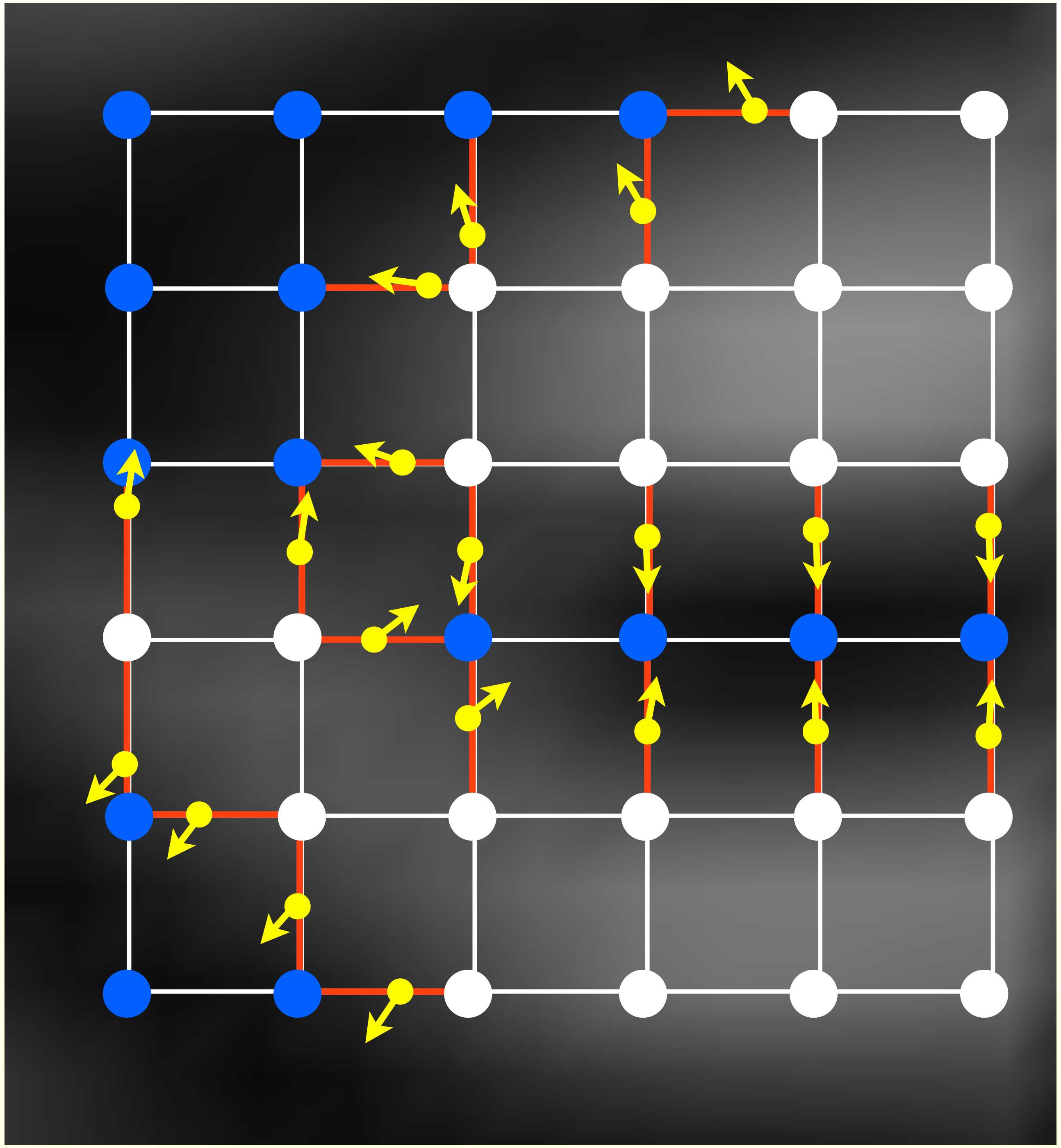
2D Dual Contouring

- Compute intersections



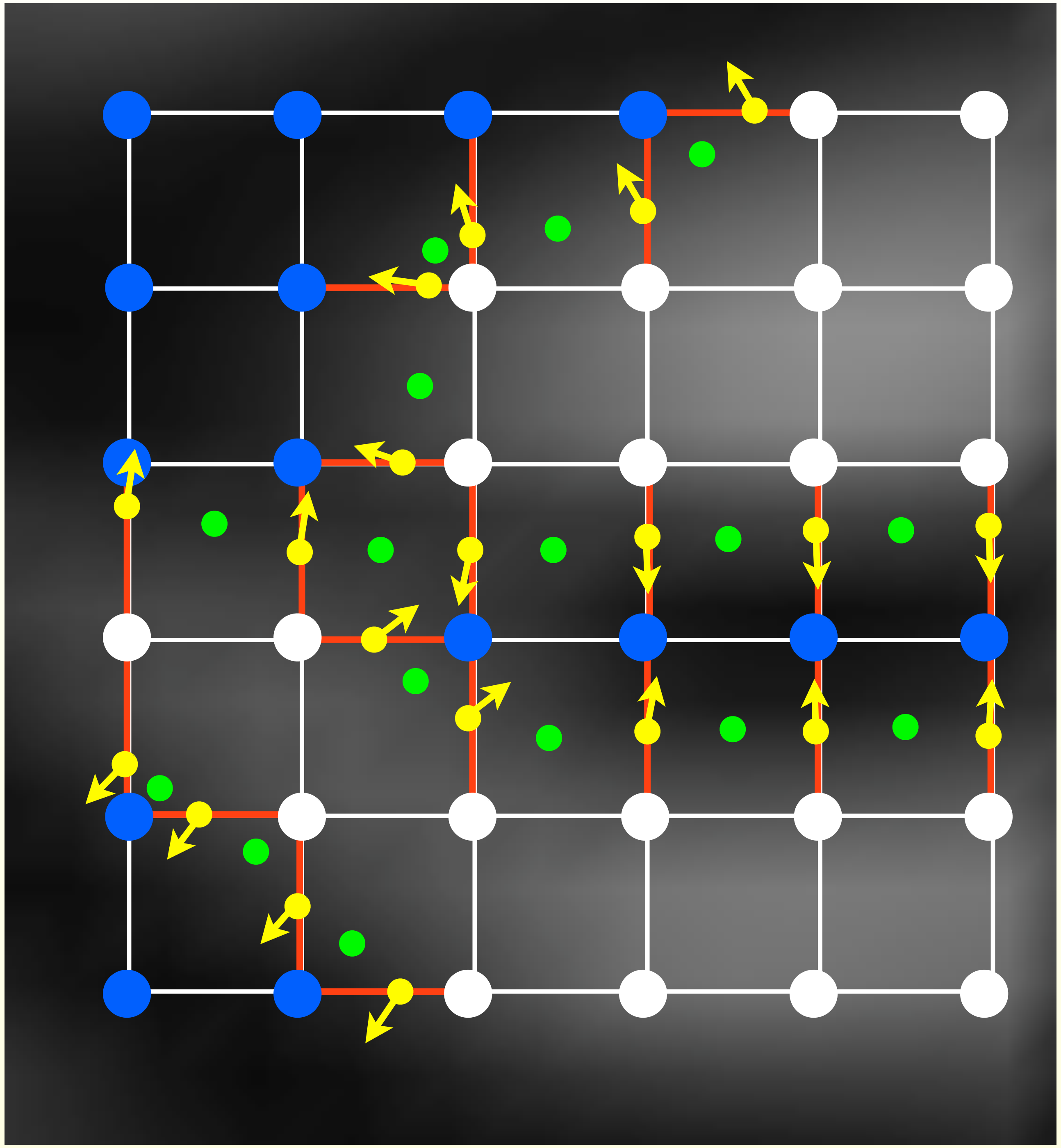
2D Dual Contouring

- Compute intersections
- Compute normals



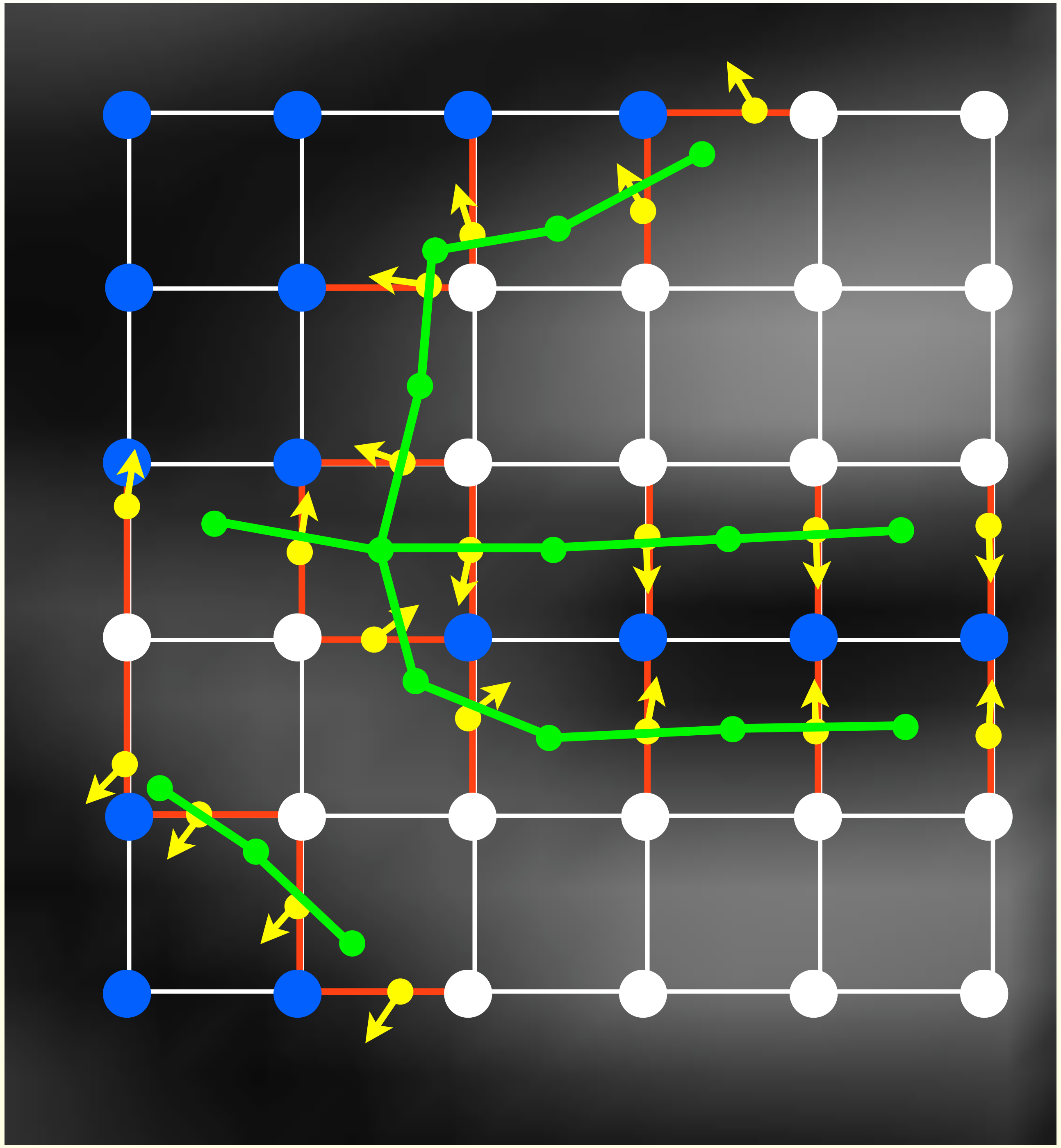
2D Dual Contouring

- Compute intersections
- Compute normals
- Compute dual points
  - for each voxel with sign change



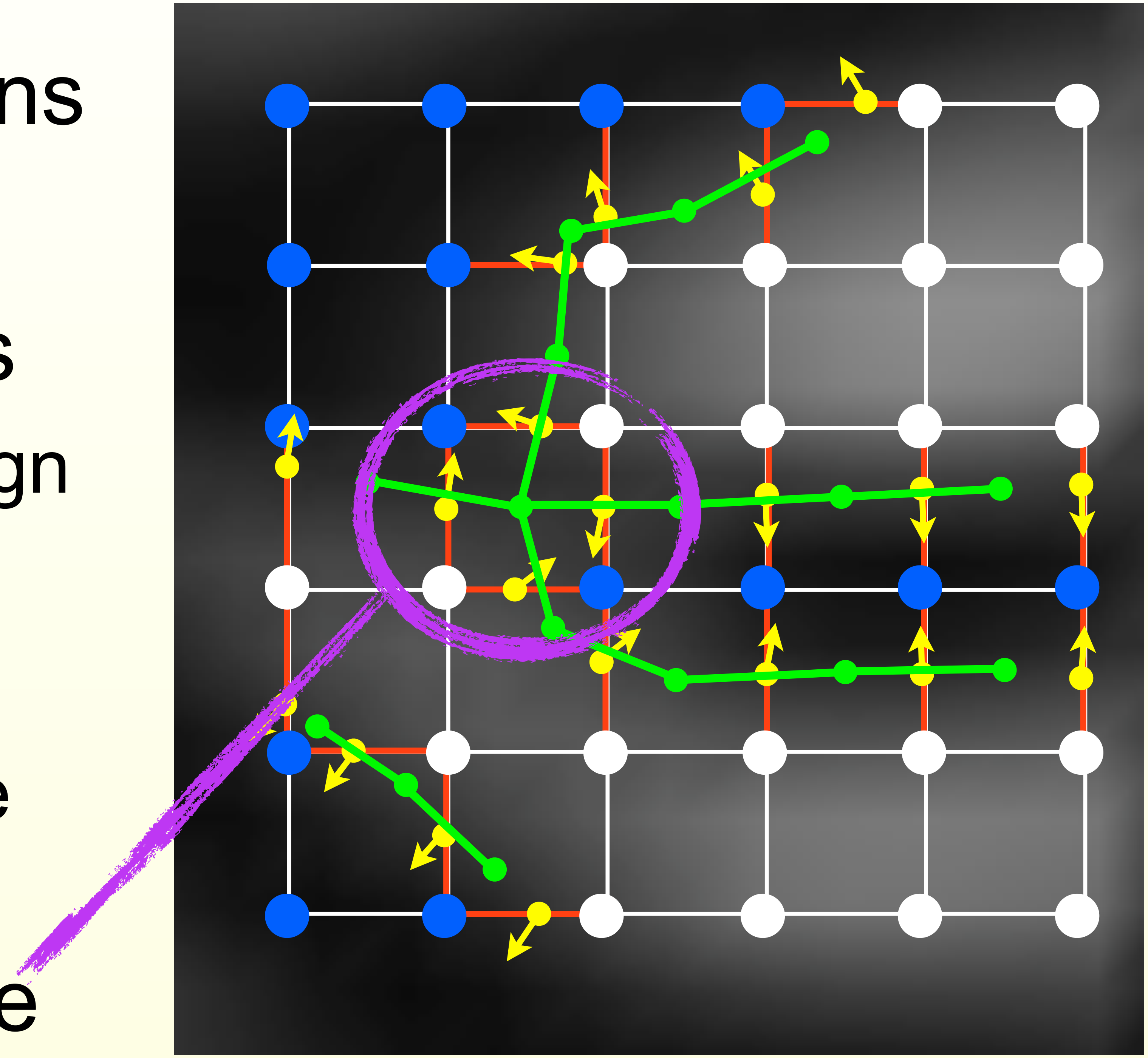
2D Dual Contouring

- Compute intersections
- Compute normals
- Compute dual points
  - for each voxel with sign change
- Connect dual points
  - across each red edge



2D Dual Contouring

- Compute intersections
- Compute normals
- Compute dual points
  - for each voxel with sign change
- Connect dual points
  - across each red edge
- We could reintroduce ambiguity

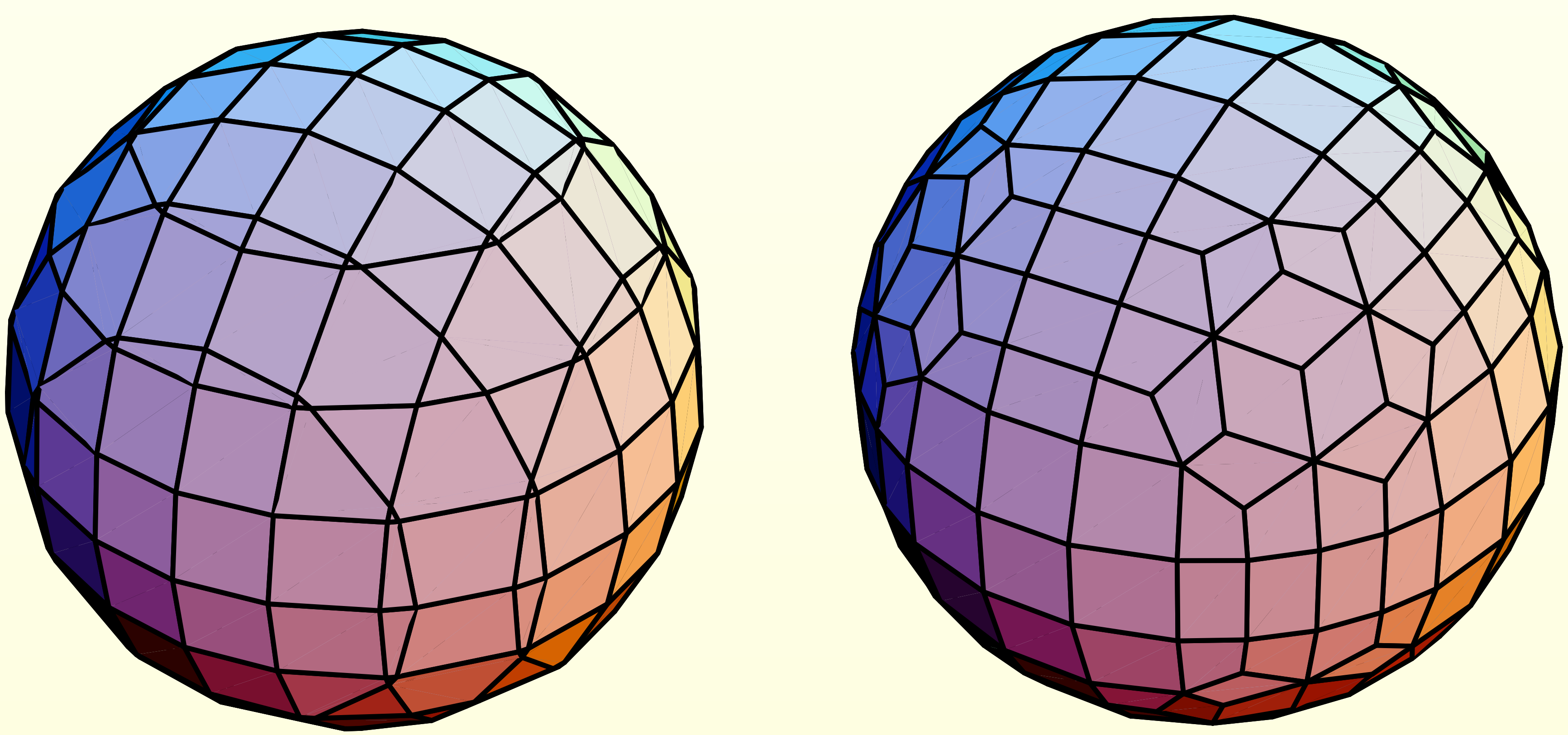


Dual Contouring

- No ambiguities
- Does not interpolate known surface points/ normals
- No special cases for features
- No lookup tables
- Some trickiness in matrix pseudo-inverse
- Naturally produces quads, not triangles (in 3D)
  - What would dual marching tetrahedra produce?

Mesh quality

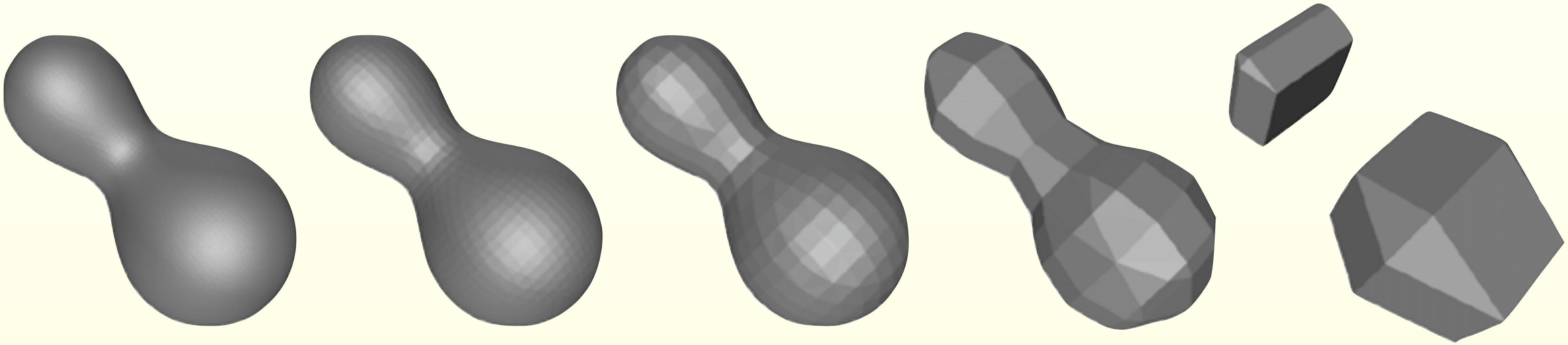
- Dual contouring produces higher-quality meshes
  - Flat surfaces never have bad triangles/quads
- Control over placement of samples



[Ju et al. 2002]

No Topological Guarantees

- Discrete Sampling: Expect Resolution Issues



[Paul Bourke]

Isotopic Meshing

- Reduce mesh size until all features resolved
- Within each voxel that contains the surface:

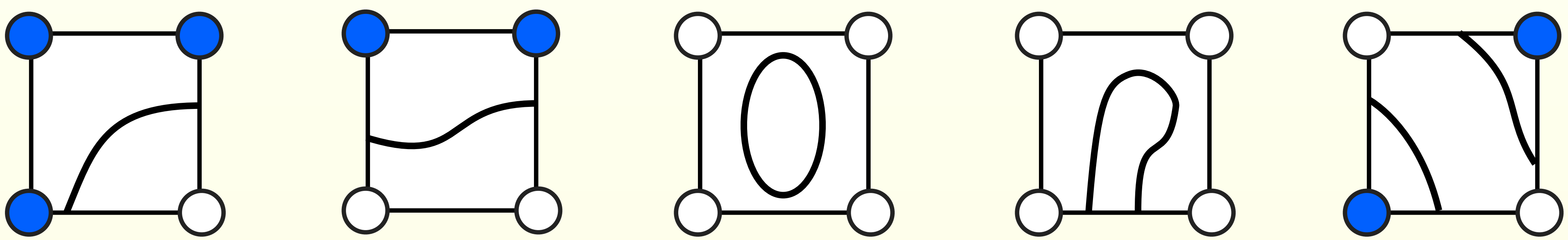
$$\nabla f(\mathbf{x}) \cdot \nabla f(\mathbf{y}) > 0 \quad \forall \mathbf{x}, \mathbf{y}$$

- Normals in a -cone

Isotopic Meshing

- Reduce mesh size until all features resolved
- Within each voxel that contains the surface:

$$\nabla f(\mathbf{x}) \cdot \nabla f(\mathbf{y}) > 0 \quad \forall \mathbf{x}, \mathbf{y}$$

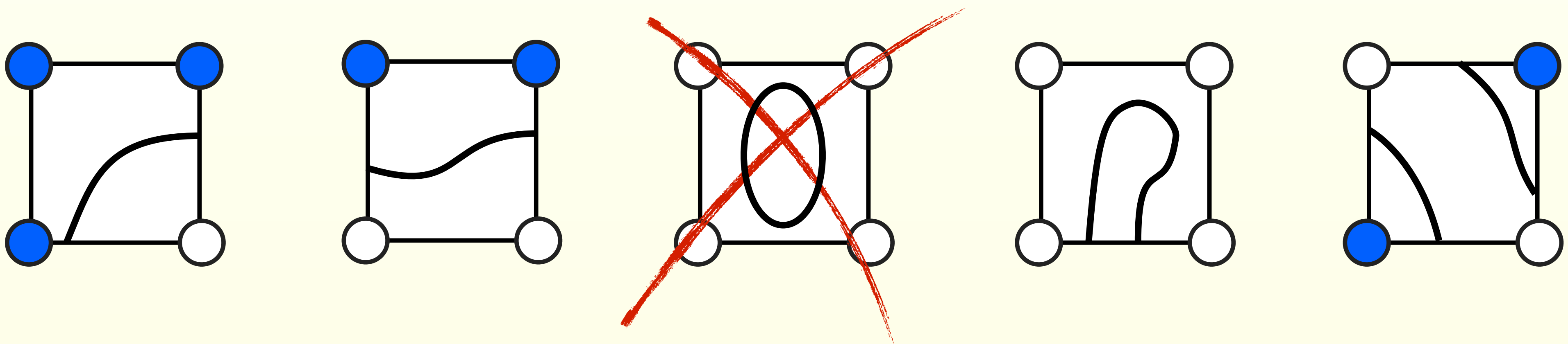


- Normals in a   -cone

Isotopic Meshing

- Reduce mesh size until all features resolved
- Within each voxel that contains the surface:

$$\nabla f(\mathbf{x}) \cdot \nabla f(\mathbf{y}) > 0 \quad \forall \mathbf{x}, \mathbf{y}$$

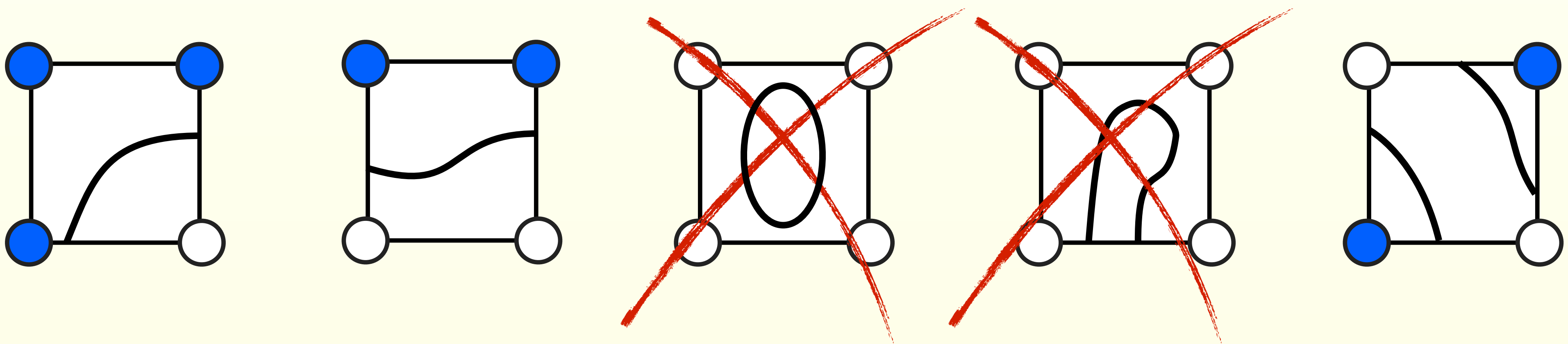


- Normals in a -cone

Isotopic Meshing

- Reduce mesh size until all features resolved
- Within each voxel that contains the surface:

$$\nabla f(\mathbf{x}) \cdot \nabla f(\mathbf{y}) > 0 \quad \forall \mathbf{x}, \mathbf{y}$$

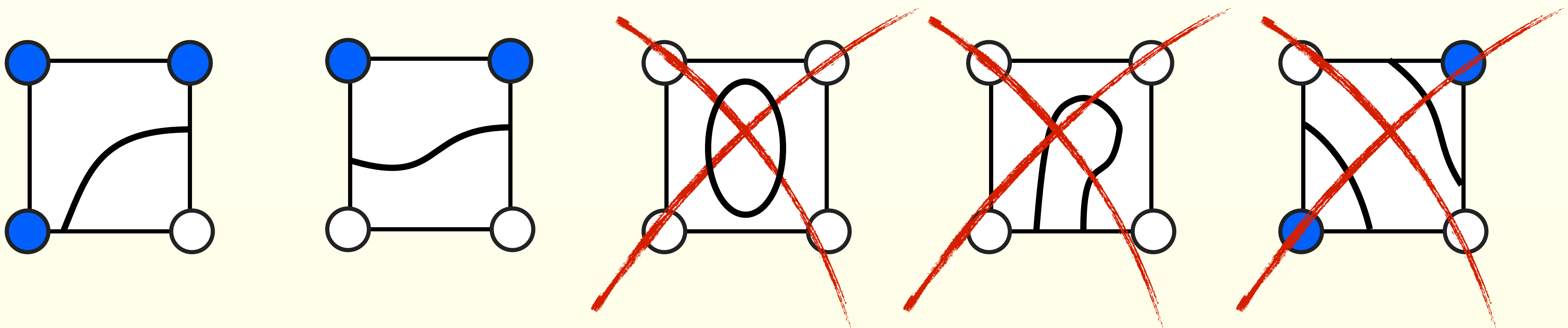


- Normals in a -cone

Isotopic Meshing

- Reduce mesh size until all features resolved
- Within each voxel that contains the surface:

$$\nabla f(\mathbf{x}) \cdot \nabla f(\mathbf{y}) > 0 \quad \forall \mathbf{x}, \mathbf{y}$$

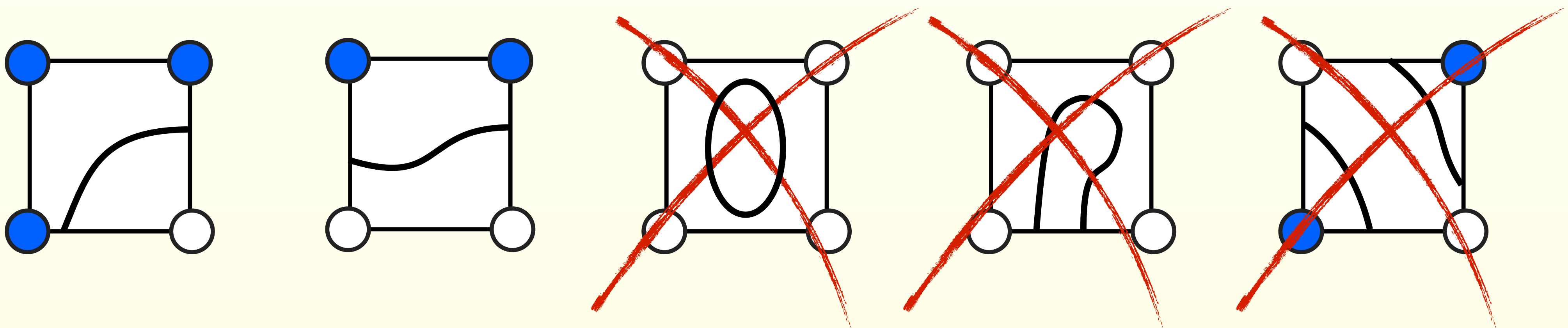


- Normals in a -cone

Isotopic Meshing

- Reduce mesh size until all features resolved
- Within each voxel that contains the surface:

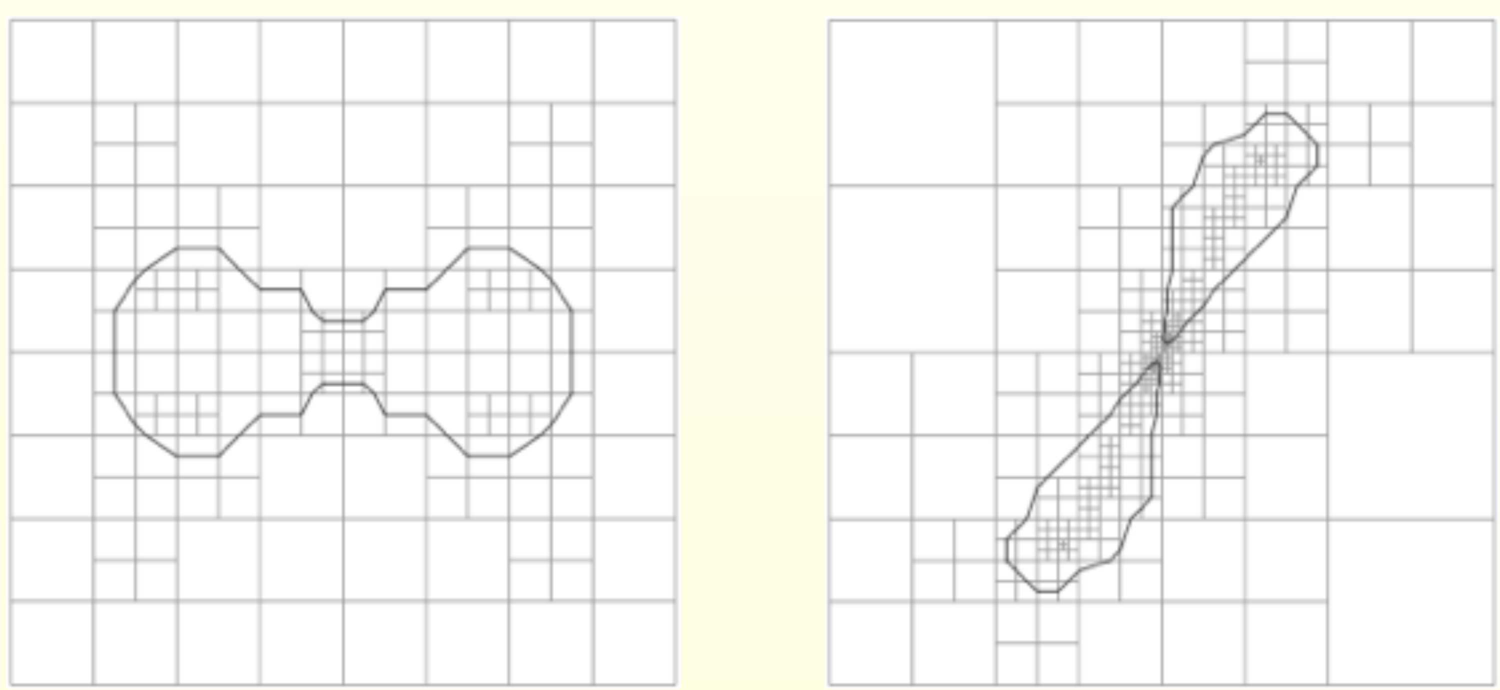
$$\nabla f(\mathbf{x}) \cdot \nabla f(\mathbf{y}) > 0 \quad \forall \mathbf{x}, \mathbf{y}$$



- Normals in a $\frac{\pi}{2}$ -cone

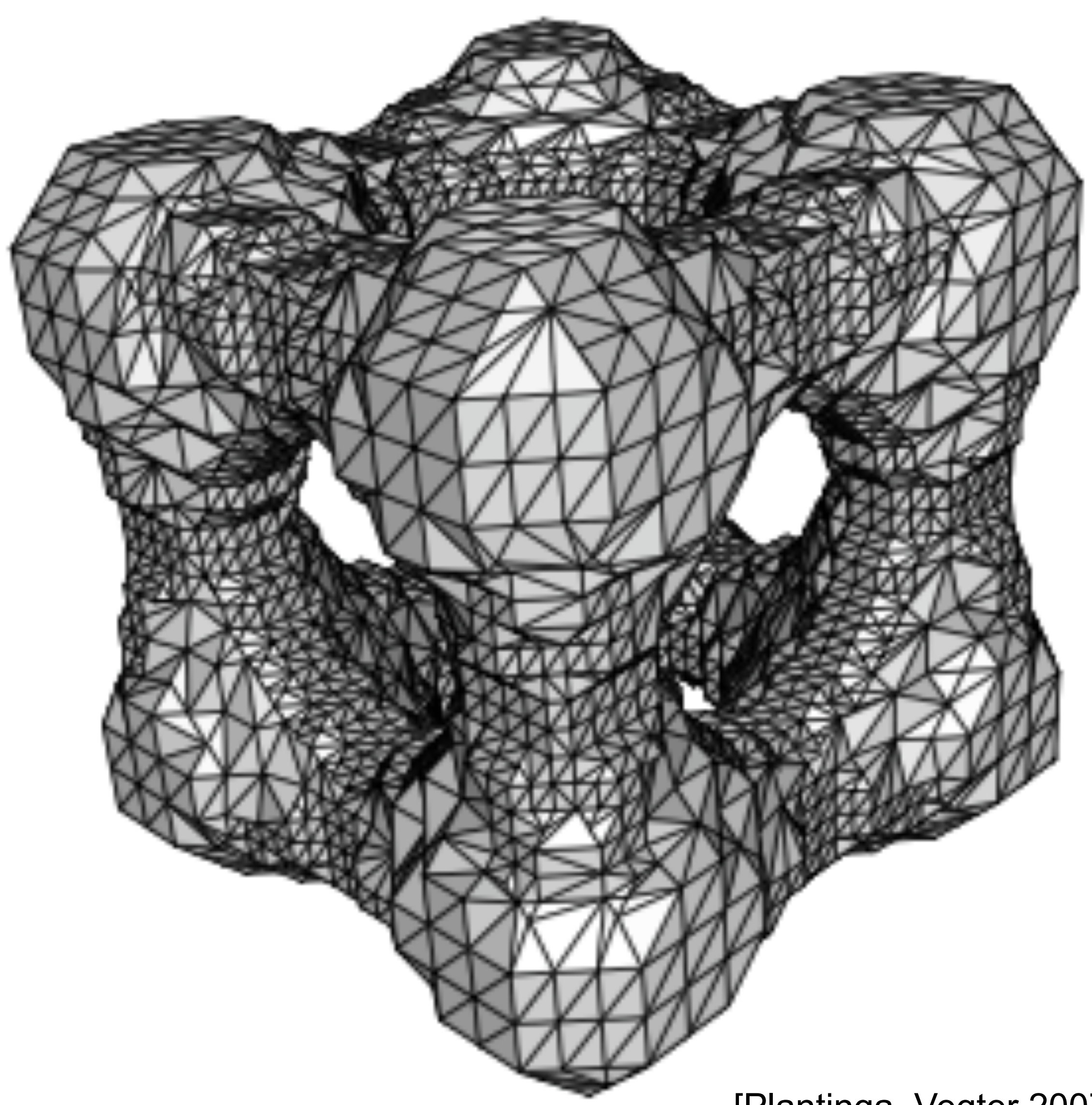
Isotopic Meshing

- Check the condition using interval arithmetic
  - Good for analytic surfaces
  - Expensive for sampled volume data
- Refine hierarchically: Use quadtree/octree
  - Less painful: Use tetrahedral subdivision



[Plantinga, Vegter 2007]

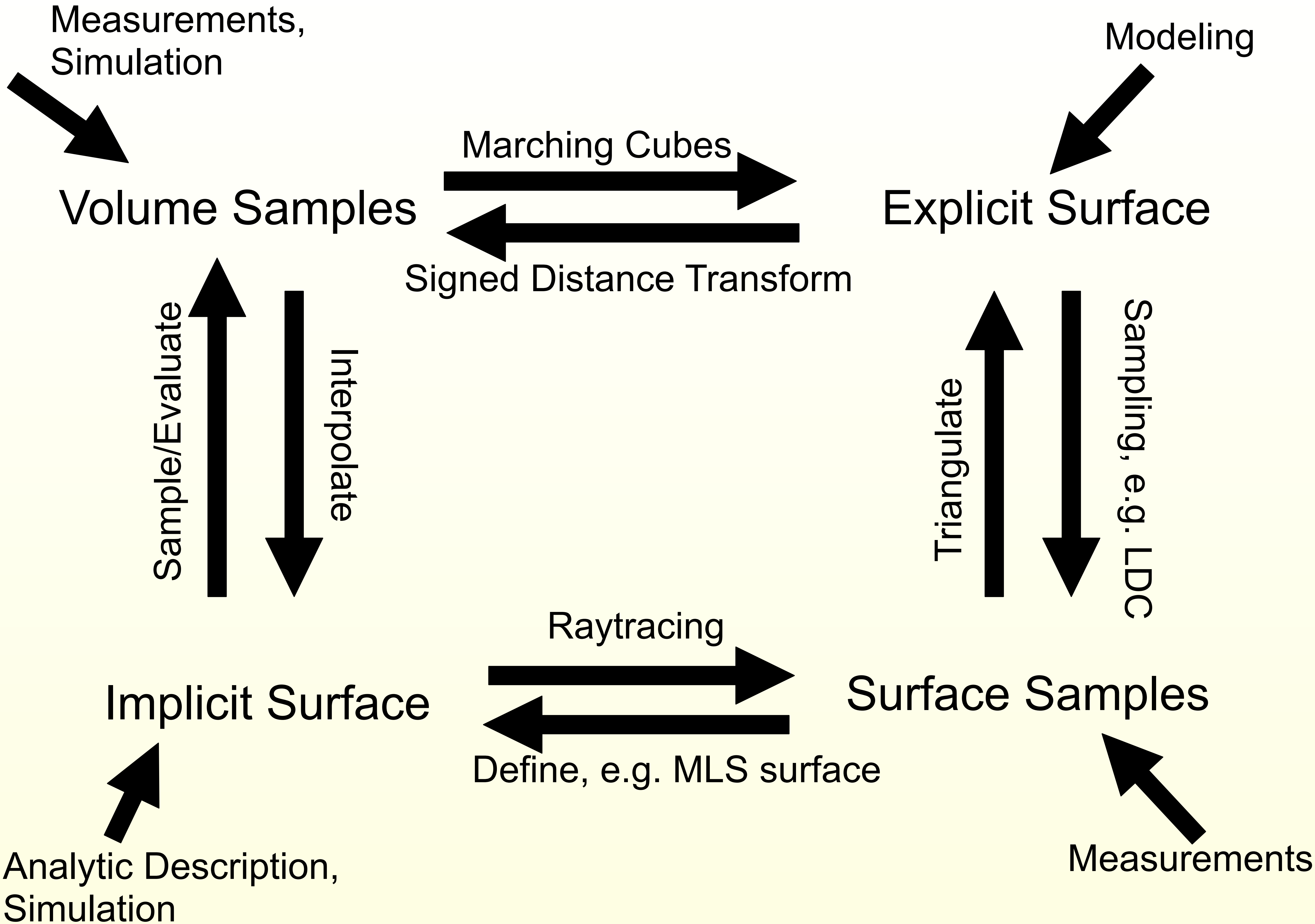
Isotopic Meshing

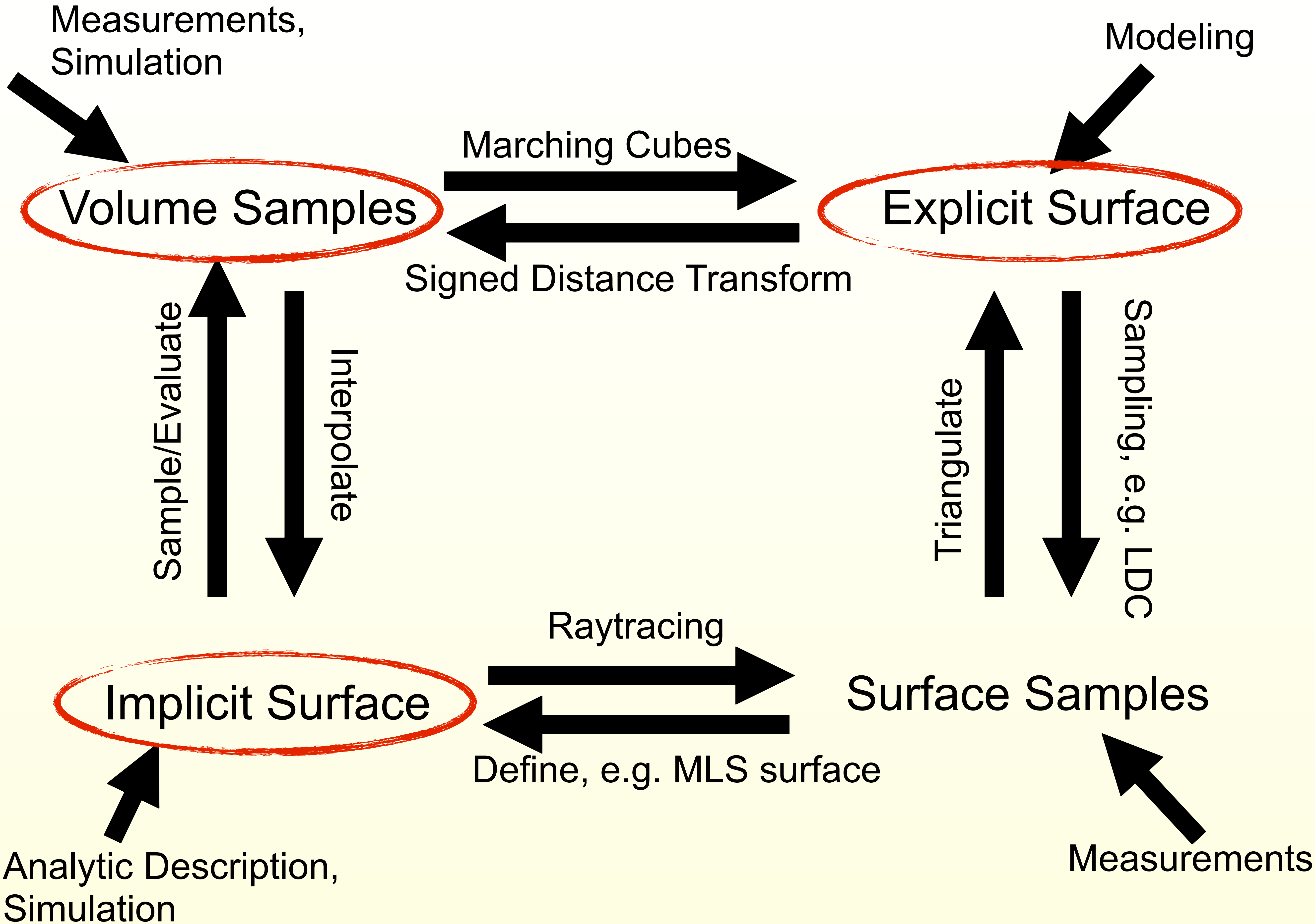


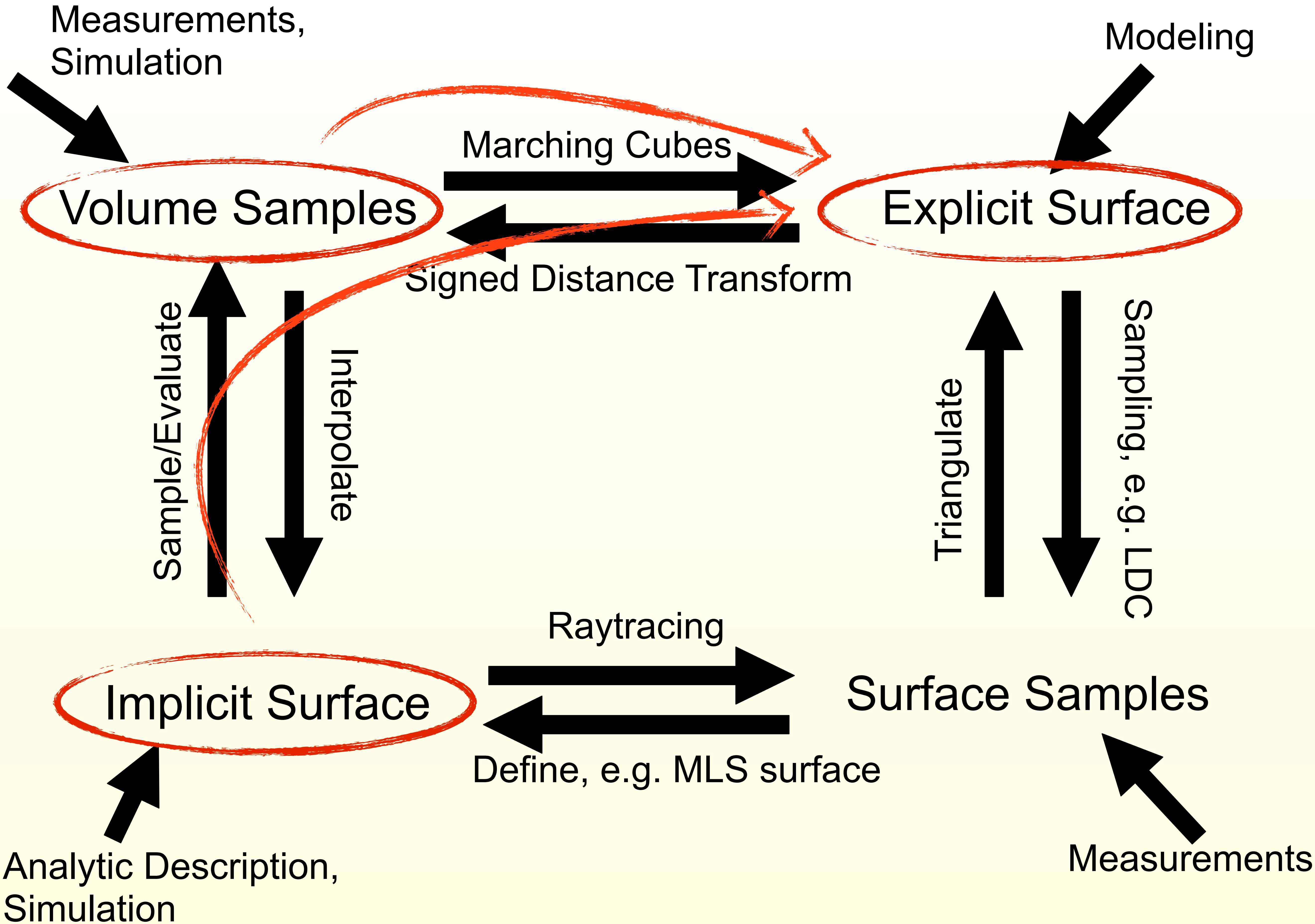
[Plantinga, Vegter 2007]

$$f(x, y, z) = x^4 - 5x^2 + y^4 - 5y^2 + z^4 - 5z^2 + 10$$

Summary


Measurements,
Simulation
Modeling
Marching Cubes
Volume Samples
Explicit Surface
Signed Distance Transform
Sample/Evaluate
Interpolate
Triangulate
Sampling, e.g. LDC
Raytracing
Implicit Surface
Surface Samples
Define, e.g. MLS surface
Analytic Description,
Simulation
Measurements

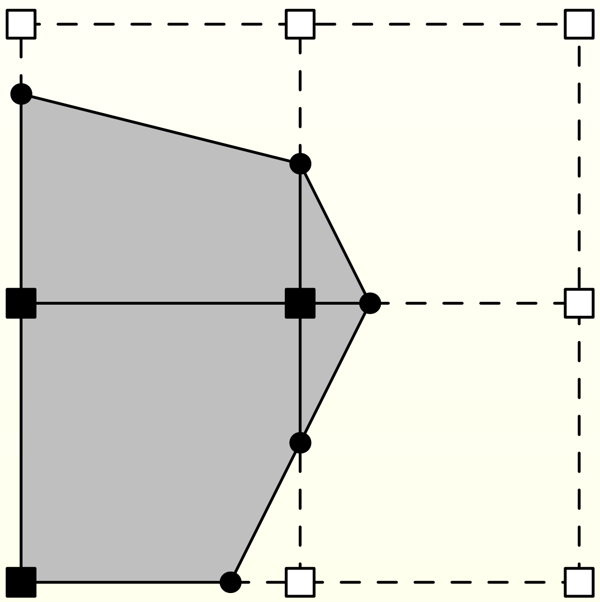

Measurements,
Simulation
Modeling
Marching Cubes
Volume Samples
Explicit Surface
Signed Distance Transform
Sample/Evaluate
Interpolate
Triangulate
Sampling, e.g. LDC
Raytracing
Implicit Surface
Surface Samples
Define, e.g. MLS surface
Analytic Description,
Simulation
Measurements



Measurements,
Simulation
Modeling
Marching Cubes
Volume Samples
Explicit Surface
Signed Distance Transform
Sample/Evaluate
Interpolate
Triangulate
Sampling, e.g. LDC
Raytracing
Implicit Surface
Surface Samples
Define, e.g. MLS surface
Analytic Description,
Simulation
Measurements

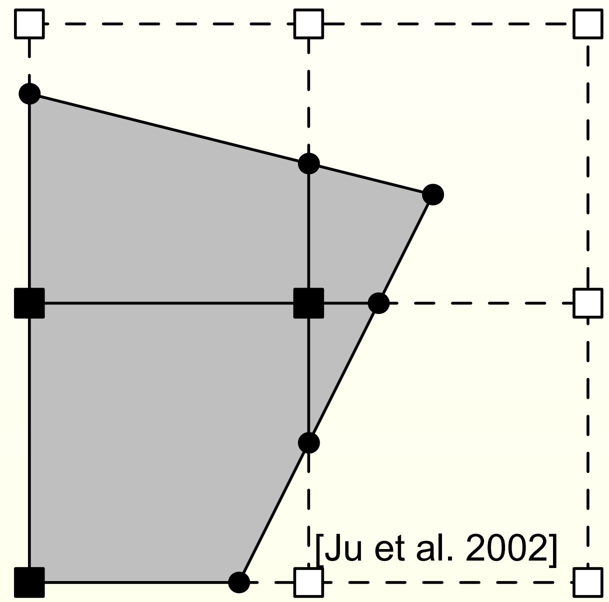
Marching Cubes Variants

Marching Cubes



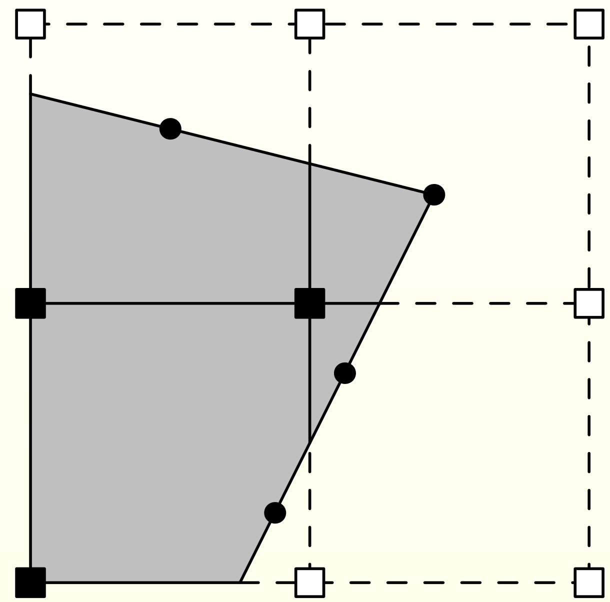
- Lookup tables
- No sharp features

Extended Marching Cubes



- Hybrid method
- Edges/ Corners are special cases
- Need feature threshold

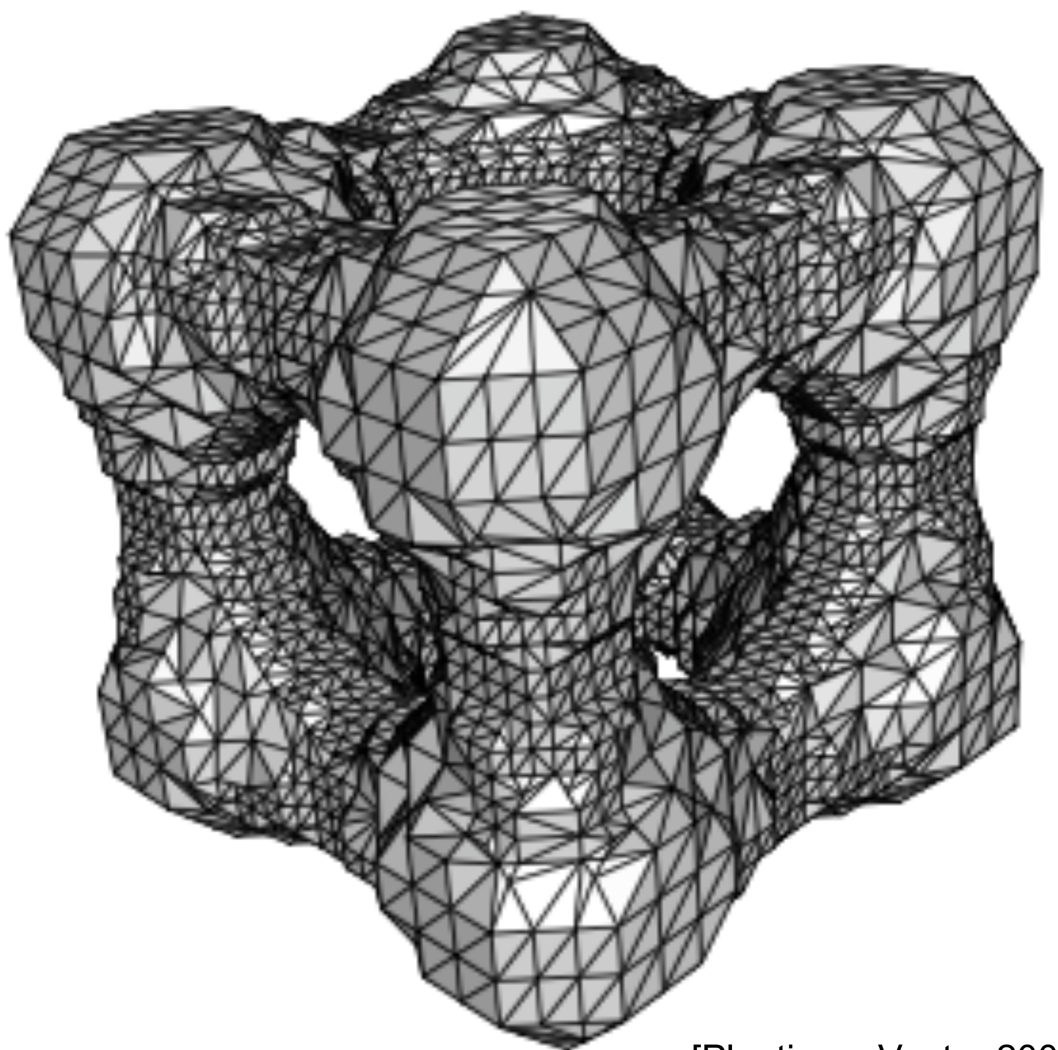
Dual Contouring



- Dual method
- No special cases
- No feature threshold

Topological Guarantees

- Enforced by subdivision
- Best for analytic surfaces
- Hierarchical refinement



[Plantinga, Vegter 2007]

Literature

- W. Lorensen, H. Cline: "**Marching Cubes: A High Resolution 3D Surface Construction Algorithm**", SIGGRAPH '87
- J. Bloomenthal: "**Polygonisation of Implicit Surfaces**". Computer-Aided Geometric Design 5(4), 1988
- Foley, van Dam, Feiner, Hughes: "**Computer Graphics: Principles and Practice**", Addison Wesley, 1995
- S. Gibson: "**Using distance maps for accurate surface reconstruction in sampled volumes**", IEEE Volume Visualization Symposium, 1998
- L. Kobbelt, M. Botsch, U. Schwanecke, H.-P. Seidel: "**Feature Sensitive Surface Extraction from Volume Data**", SIGGRAPH '01
- T. Ju, F. Losasso, S. Schaeffer, J. Warren: "**Dual Contouring of Hermite Data**", SIGGRAPH '02
- S. Plantinga and G. Vegter: "**Isotopic Meshing of Implicit Surfaces**", The Visual Computer 23, 2007
- James Sharman: http://www.exaflop.org/docs/marchcubes/
- Paul Bourke: http://local.wasp.uwa.edu.au/~pbourke/geometry/polygonise/