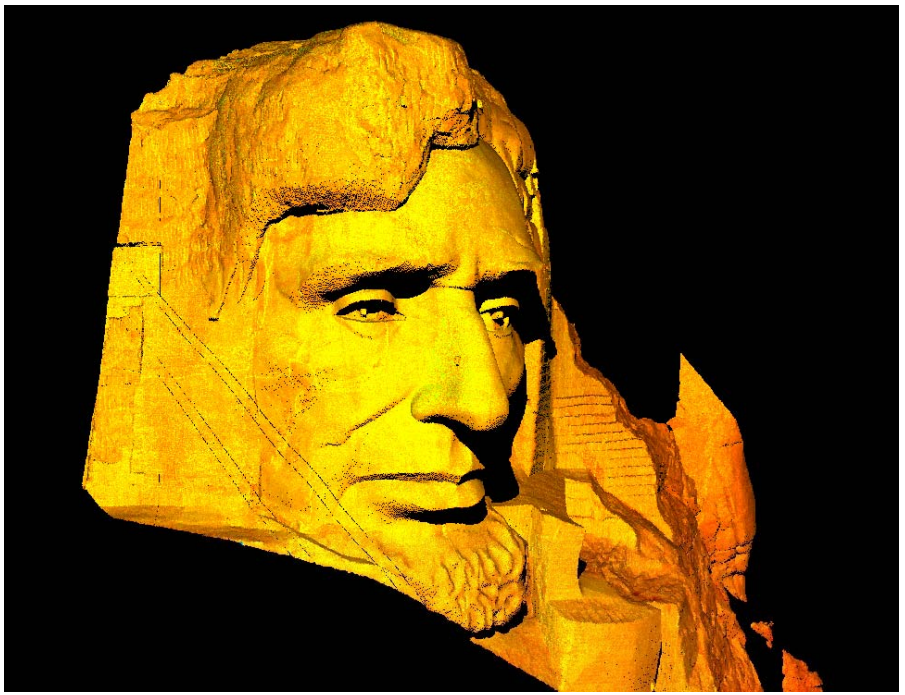
Surface **Reconstruction:** Part II

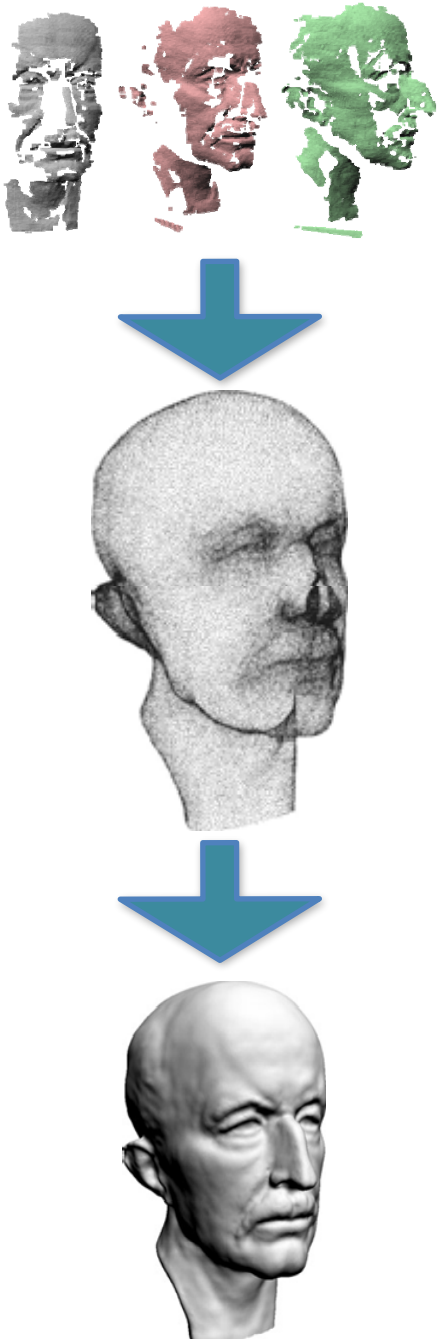


Input Data

- Set of range scans
  - Scans registered in common coordinate system

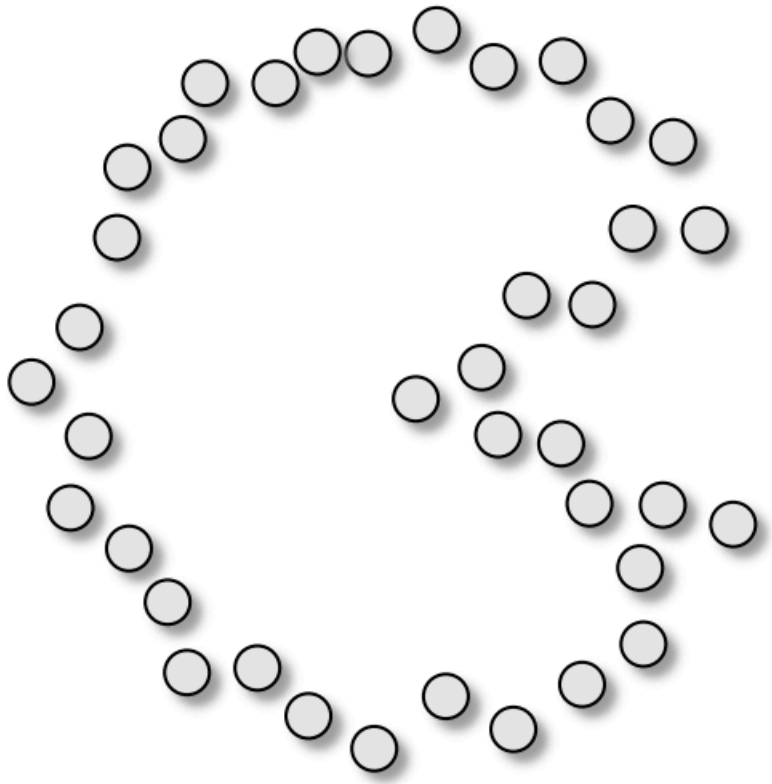
- Set of irregular sample points
  - Typically with normal vectors
  - More general

- Surface Extraction

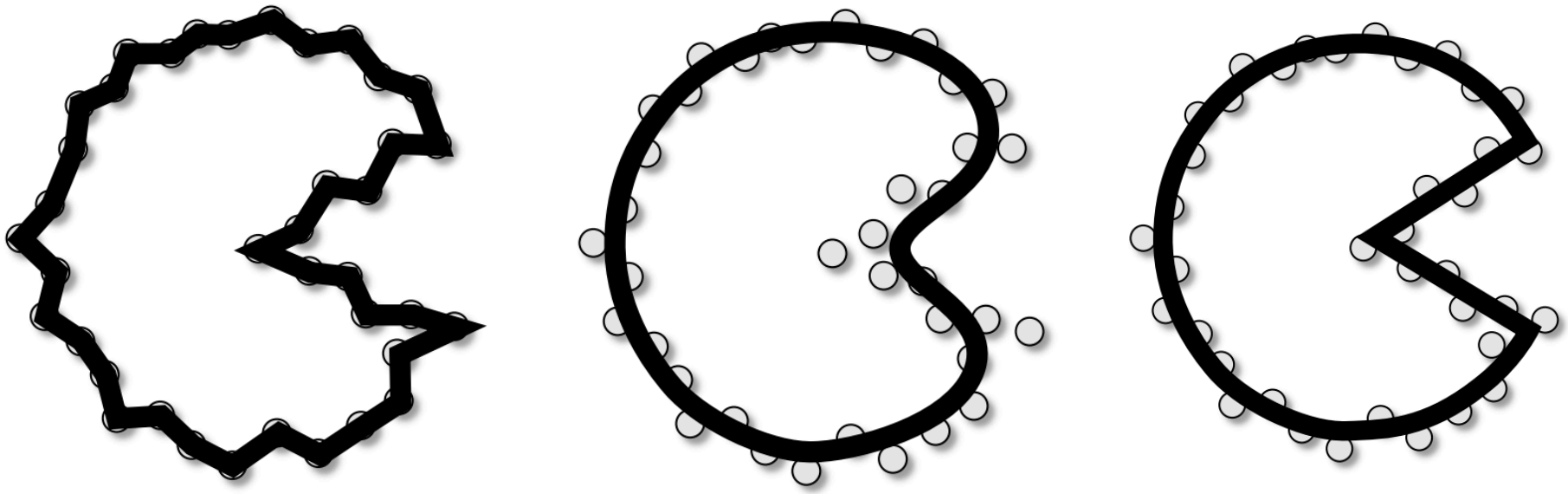


Point Cloud

- In how many ways can we connect the points?
    - Many candidate shapes
    - Ill Posed Problem!!!

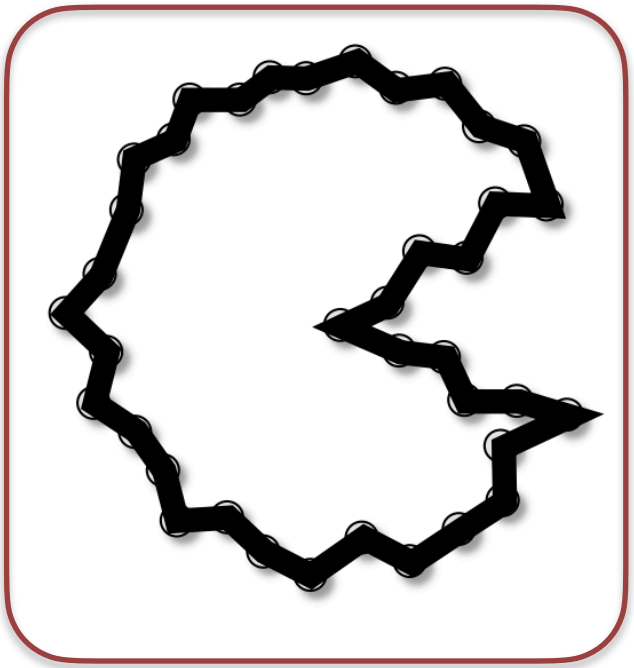


Ill Posed Problem

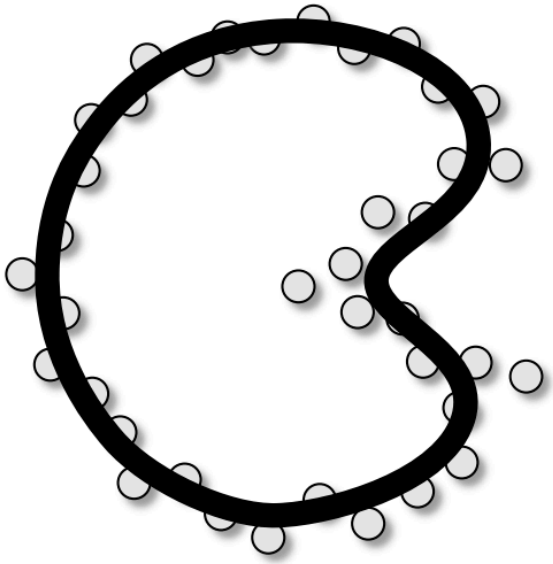


Many candidate shapes for the reconstruction problem

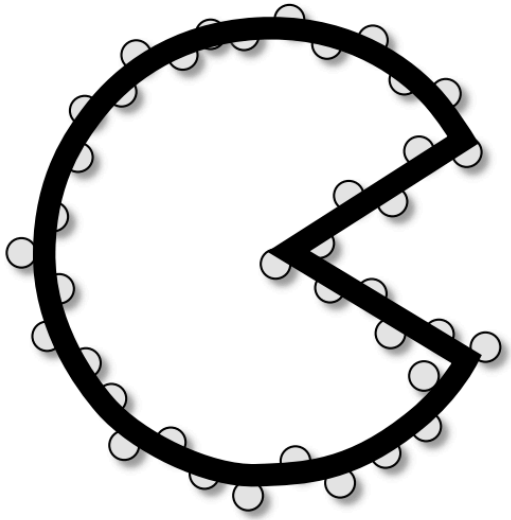
Priors



Exhaustive Connection



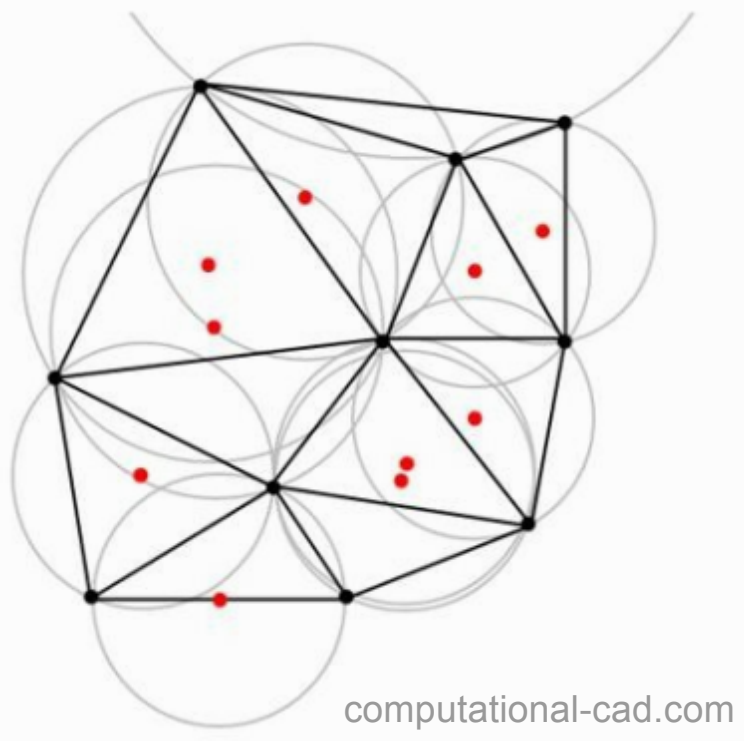
Smooth



Piece-wise Smooth

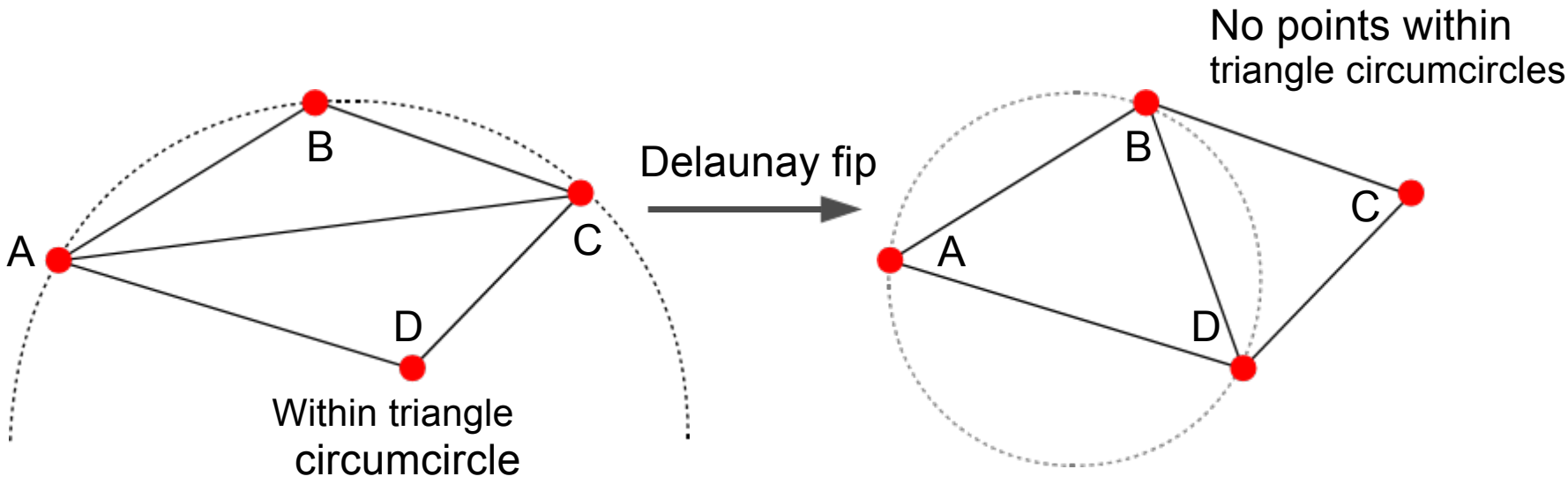
Delaunay Triangulation

- Used to generate a planar base mesh whose vertices are a set of points *P*
- A triangulation is **Delaunay** if no point in *P* lies within the circumcircle of any triangle
  - It also happens to maximize the minimum angle of any triangle, which is why it's useful

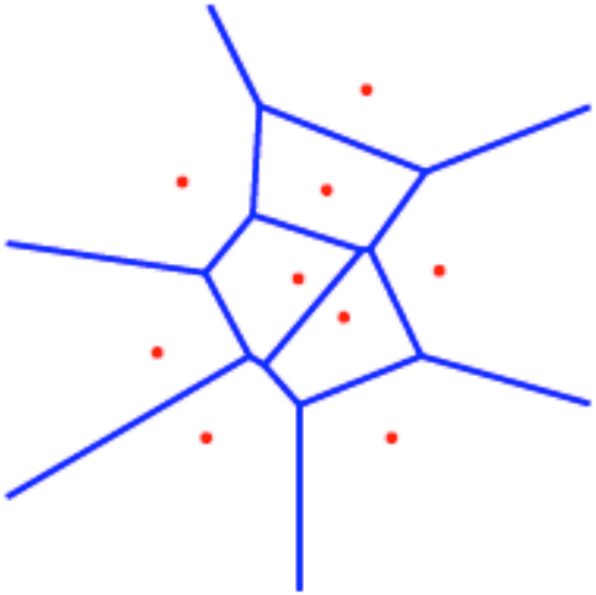


Lawson's Flip Algorithm

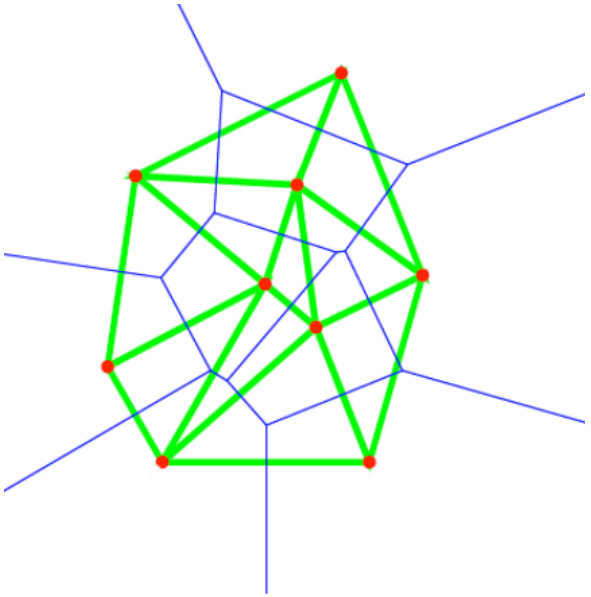
- Iteratively apply Delaunay flips to improve the triangulation
  - Identify two adjacent triangles
  - Flip the shared edge if the minimum of the 6 angles increases (i.e. if the circumcircle property can be restored)



Voronoi Tesselation and Delaunay Triangulation



Voronoi diagram – each cell consists of points nearer the cell centre than any other point in *P*

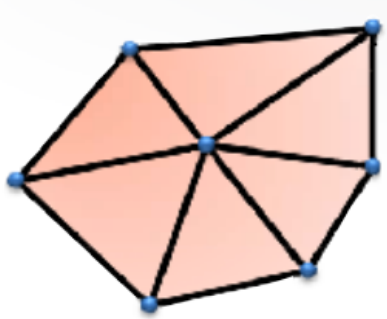


For each vertex of the Voronoi diagram, there is a Delaunay face. Two faces are adjacent if the corresponding vertices are connected by a Voronoi edge.

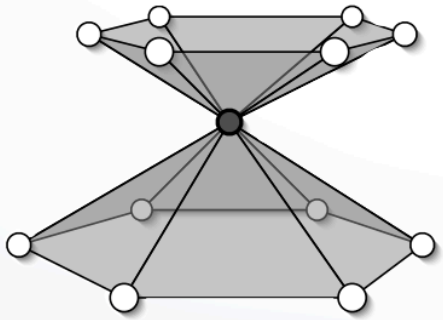
$$V(\mathbf{p_i}) = \{\mathbf{x} \in \mathbb{R}^d : \|\mathbf{x} - \mathbf{p_i}\| \leq \|\mathbf{x} - \mathbf{p_j}\|, \forall j \leq n\}.$$

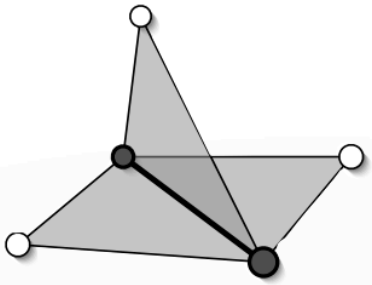
More on Triangle Meshes

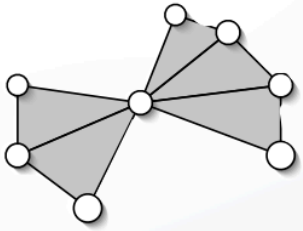
- Local neighbourhood is disk shaped and half-disk shaped at boundary



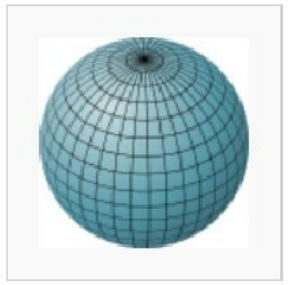
- Not a valid triangulation:







- Euler's formula: V-E+F = 2(1-g)

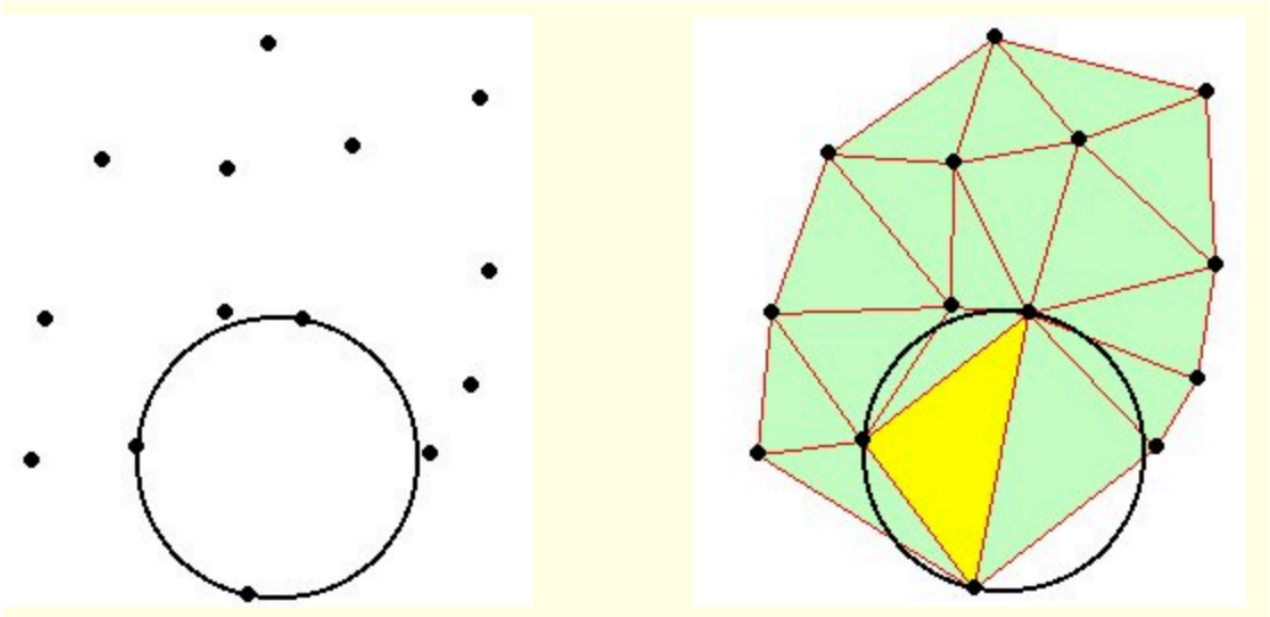


genus 0

genus 1

Comments…

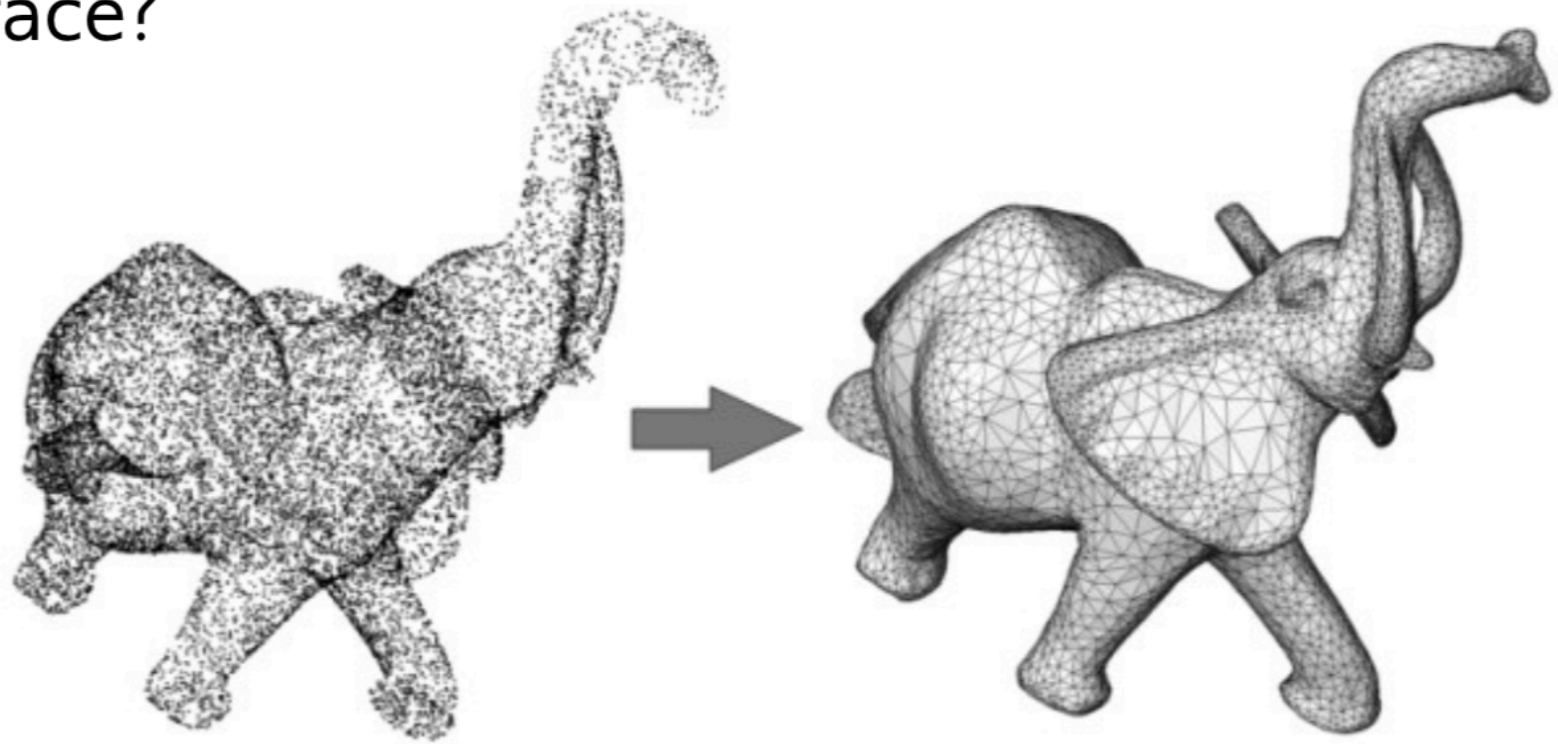
- Triangulations is sensitive to holes, sampling pattern, outlier noise



- Can we estimate underlying surface more robustly?

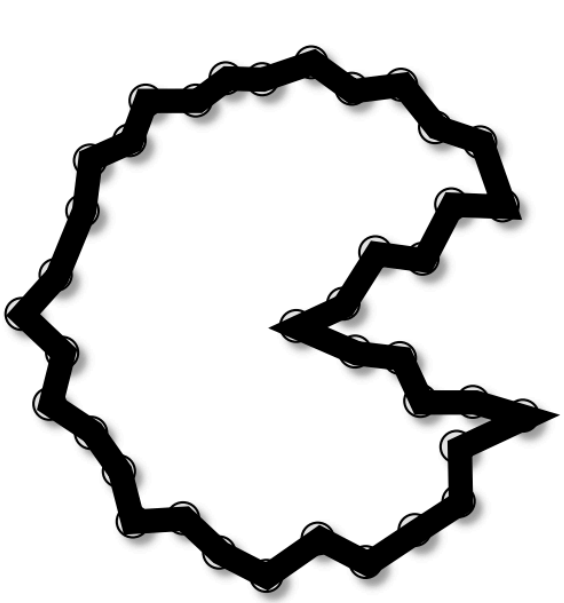
Comments

- Triangulation is sensitive to noise, sampling pattern, omissions etc
- Can we more robustly recover the underlying surface?

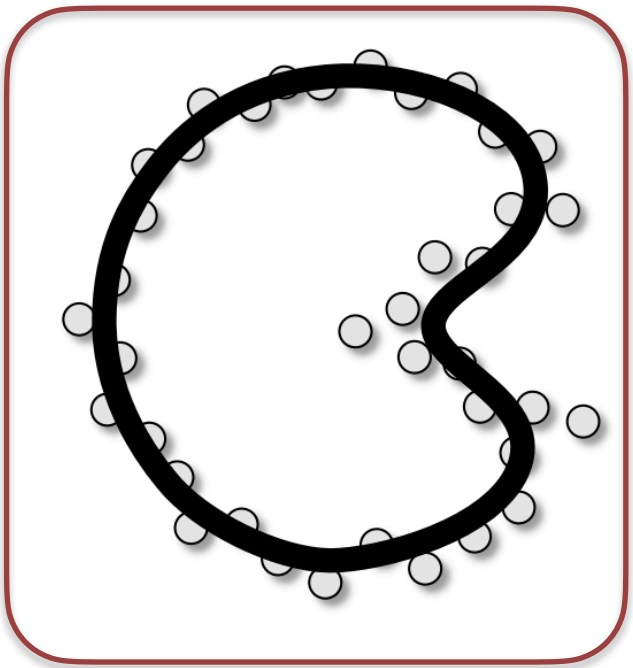


Slides adapted from Kazhdan, Bolitho and Hoppe

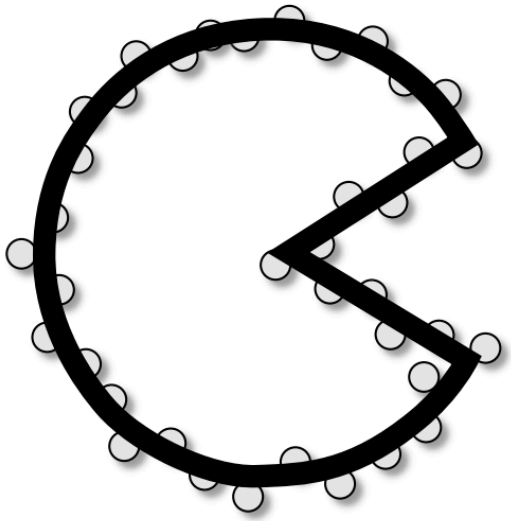
Priors



Exhaustive Connection



Smooth



Piece-wise Smooth

Surfaces: Representations

- Parametric Surfaces:
  - Vector-valued parameterization ‘f’
  - Maps 2D parameter domain $\Omega \subset \mathbb{R}^2$ to the surface $\mathcal{S} = \mathbf{f}(\Omega) \subset \mathbb{R}^3$

$$\mathbf{f} : \Omega \rightarrow \mathcal{S}$$

- Implicit Surfaces:
  - Defined by the zero level set of a scalar-valued function

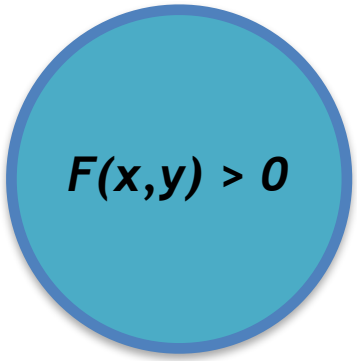
$$\mathcal{S} = \{\mathbf{x} \in \mathbb{R}^3 \mid F(\mathbf{x}) = 0\}$$

Shape Representations

- Explicit/parametric representation
  - Image of parametrization

$$F(x,y) = 0$$

$$f(t) = (x(t), y(t)) = (r\cos(t), r\sin(t))$$



$f([0,2\pi))$

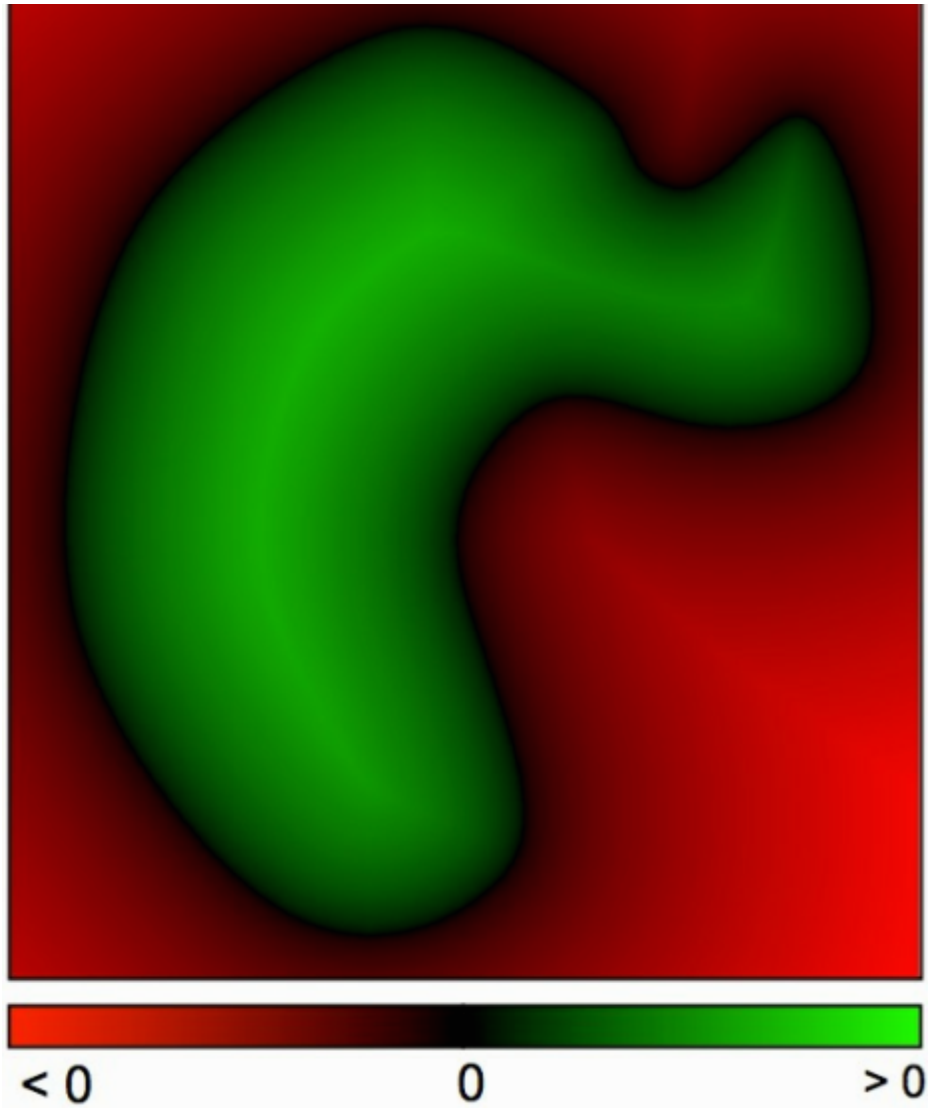
- Implicit representation
  - Zero set of distance function

F(x,y) < 0

$$F(x,y) = \sqrt{x^2 + y^2} - r$$

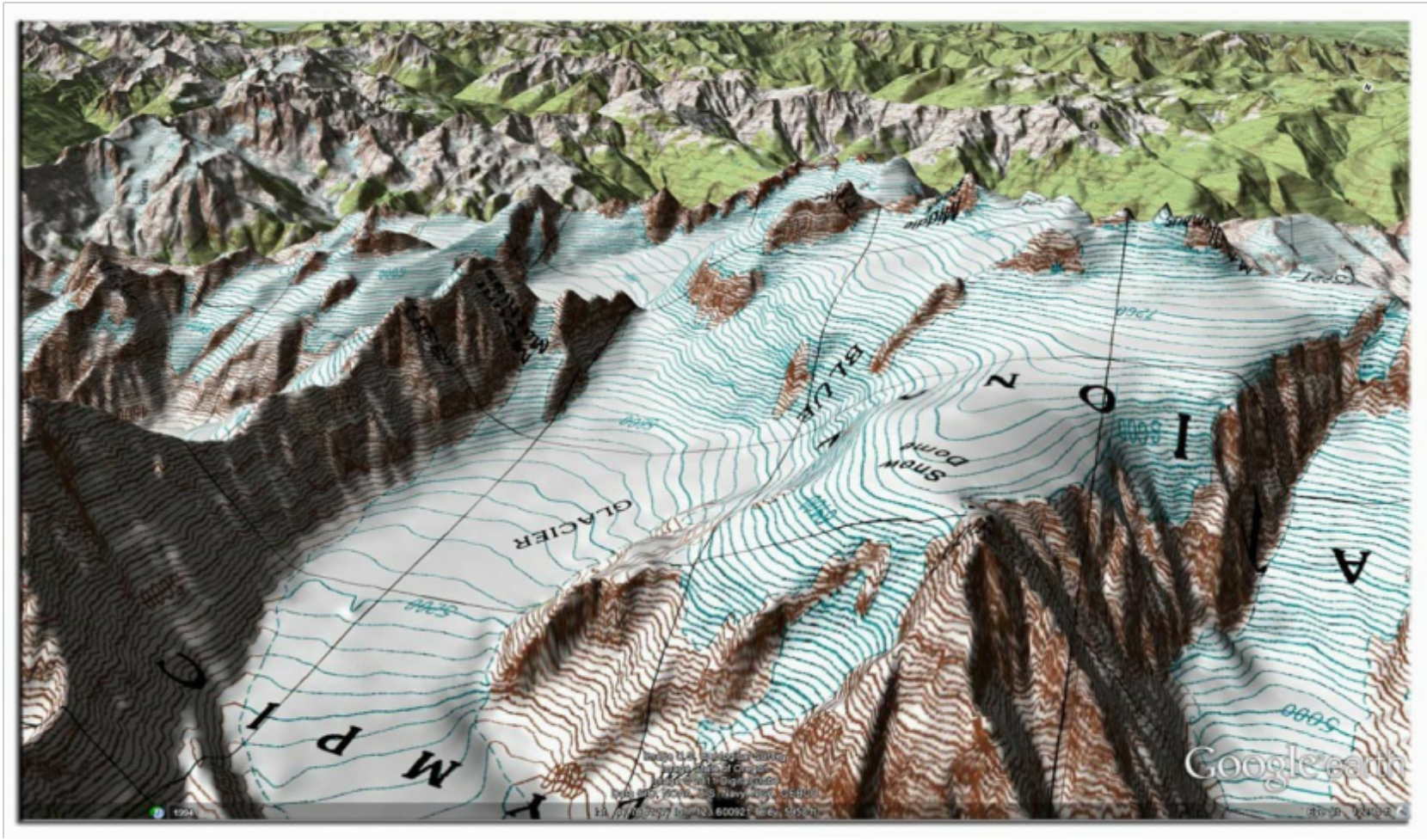
Implicit Function Approach

- Define a function with positive values inside the model and negative values outside



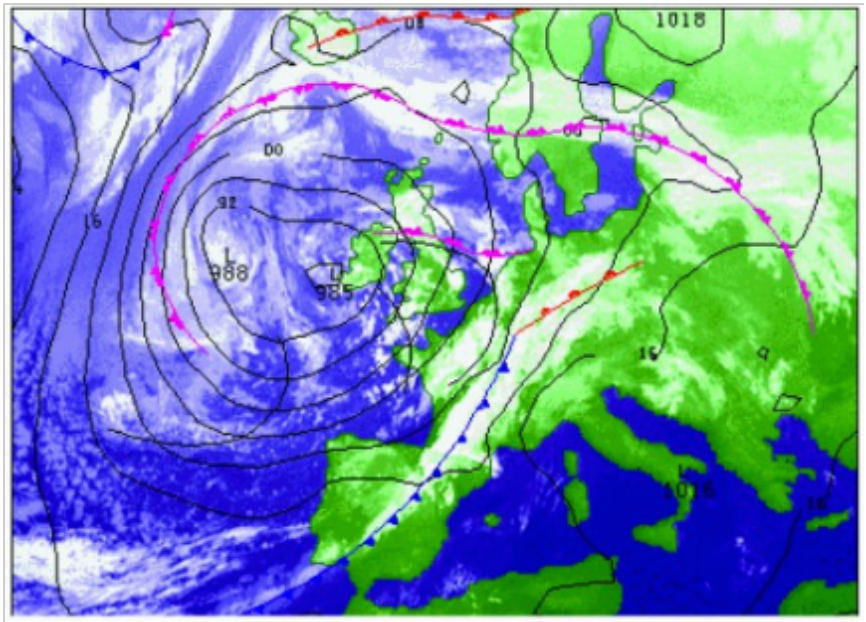
Level Set

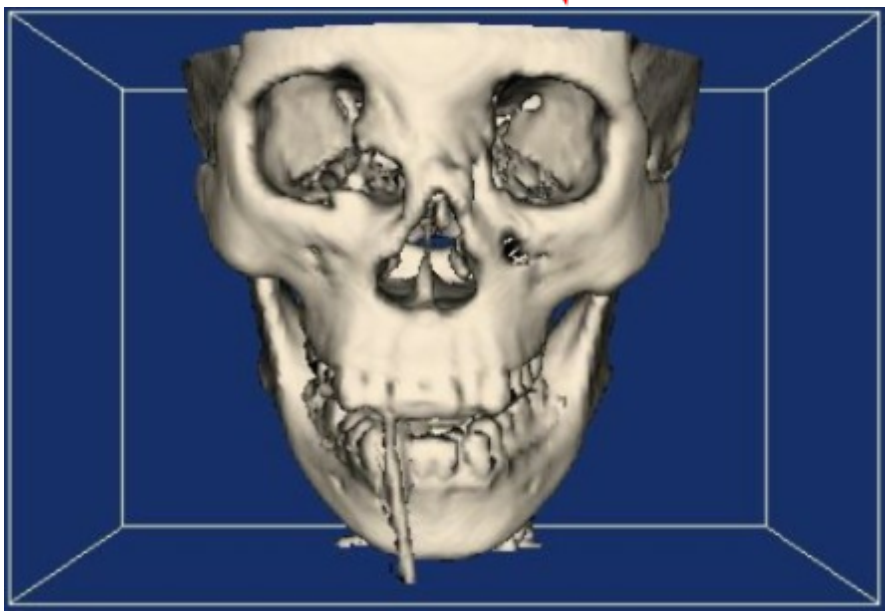
- ***c*-Level set**: The set of points where a function takes a constant value *c*



Level Set

- ***c*-Level set**: The set of points where a function takes a constant value *c*
  - **Isocontour**: Level set of a 2D function
  - **Isosurface**: Level set of a 3D function

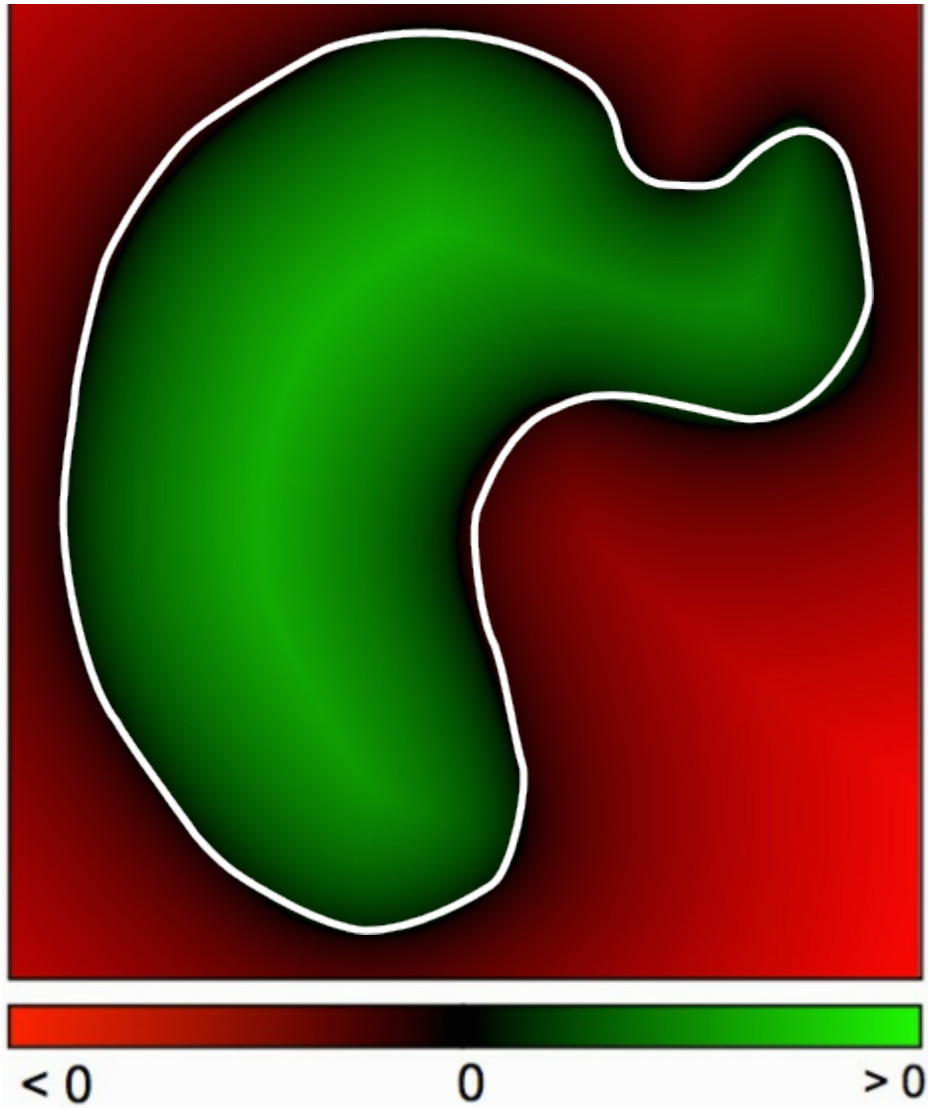




Implicit Function Approach

- Define a function with positive values inside the model and negative values outside

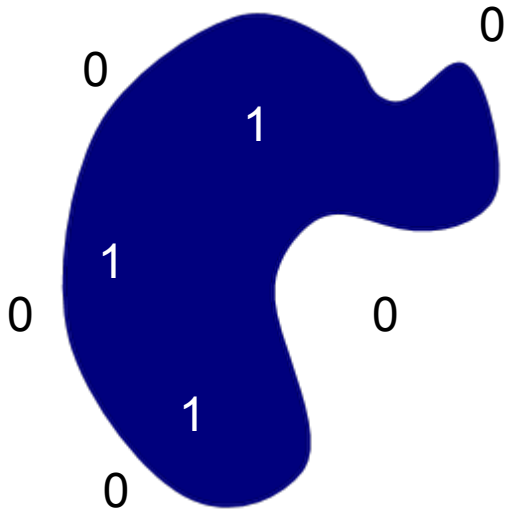
- Extract the zero-set



Key Idea

- Reconstruct the surface of the model by solving for the indicator function of the shape

$$\chi_M(p) = \begin{cases} 1 & \text{if } p \in M \\ 0 & \text{if } p \notin M \end{cases}$$



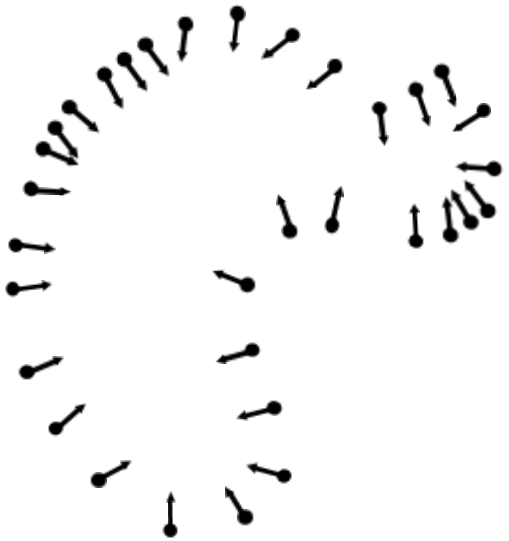
Indicator function

c_M

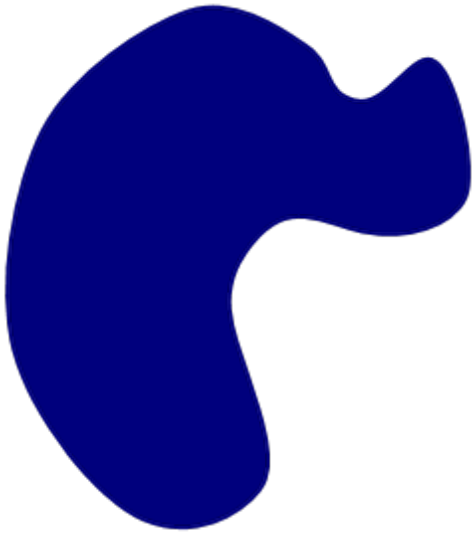
In practice, we define the indicator function to be -1/2 outside the shape and 1/2 inside, so that the surface is the zero level set. We also smooth the function a little, so that the zero set is well defined.

Challenge

- How to construct the indicator function?



Oriented points

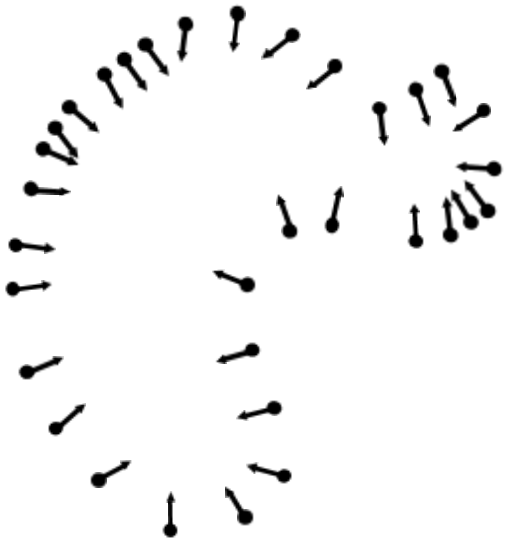


Indicator function

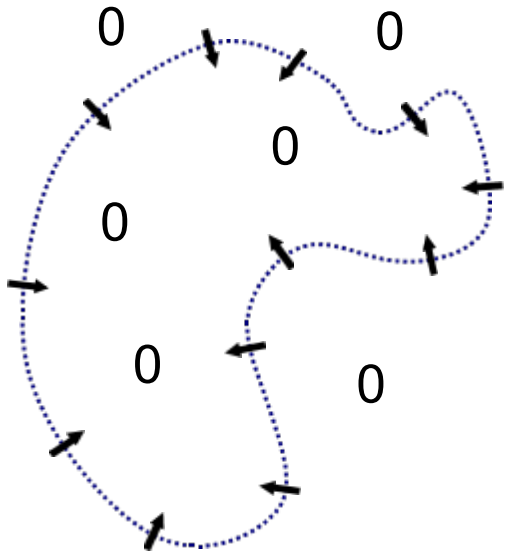
χ_M

Gradient Relationship

- There is a relationship between the normal field at the shape boundary, and the gradient of the (smoothed) indicator function



Oriented points



Indicator gradient

$\nabla \chi_M$

Linear Least Squares

- Need to solve set of equations $A\mathbf{x} = \mathbf{b}$ in a least squares
- Minimize $||\mathbf{r}|| = ||\mathbf{b} - A\mathbf{x}||^2$
- The minimum is achieved when

$$A^T A\mathbf{x} = A^T\mathbf{b}$$

$$x = \frac{A^T b}{A^T A}$$

Poisson Surface Reconstruction

- Represent the point normals by a vector field V
- Find the function χ whose gradient best approximates V

$$\min_{\chi} \|\nabla \chi - V\|^2$$

- Applying the divergence operator,

$$\nabla \cdot (\nabla \chi) = \nabla \cdot V \quad \Rightarrow \quad \Delta \chi = \nabla \cdot V$$

Laplacian $\Delta \chi = \frac{\partial^2 \chi}{\partial x^2} + \frac{\partial^2 \chi}{\partial y^2} + \frac{\partial^2 \chi}{\partial z^2}$

Divergence $\nabla \cdot V = \frac{\partial V_x}{\partial x} + \frac{\partial V_y}{\partial y} + \frac{\partial V_z}{\partial z}$

Discrete Derivative

Continuous

- Function: f
- Operator: $\frac{d}{dx}$
- Applying operator:

$$\frac{df}{dx} = f'$$

Discrete

- Vector: $\mathbf{f} = [\, f(x_1), f(x_2) \ldots f(x_n)]$
- Matrix:

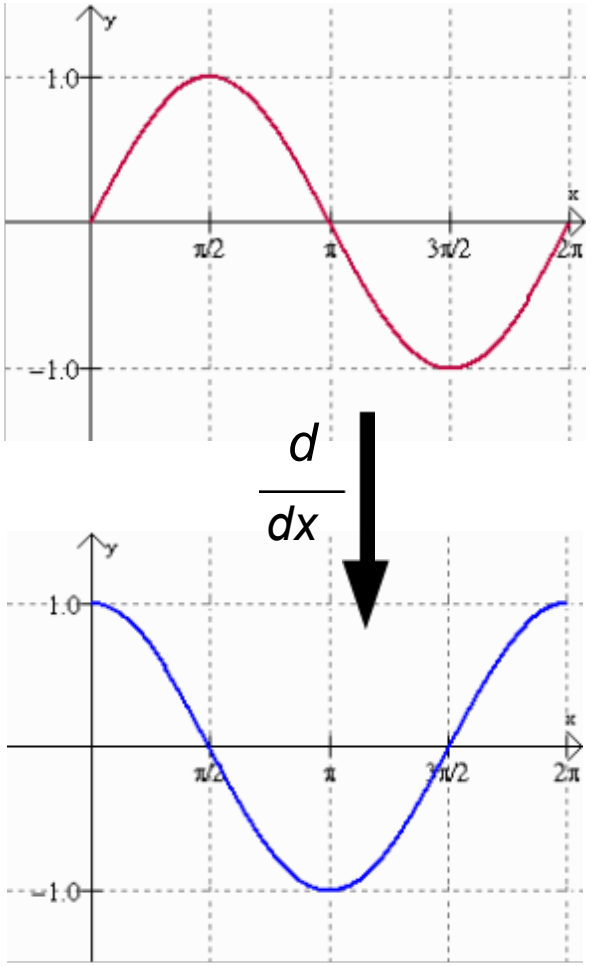
$$A = \begin{bmatrix} -1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & -1 & 1 & \cdots & 0 & 0 \\ & 0 & -1 & \cdots & 0 & 0 \\ \vdots & & & \ddots & & \vdots \\ & 0 & & & -1 & \\ 0 & 0 & \cdots & & 0 & -1 \end{bmatrix}$$

- Applying matrix:

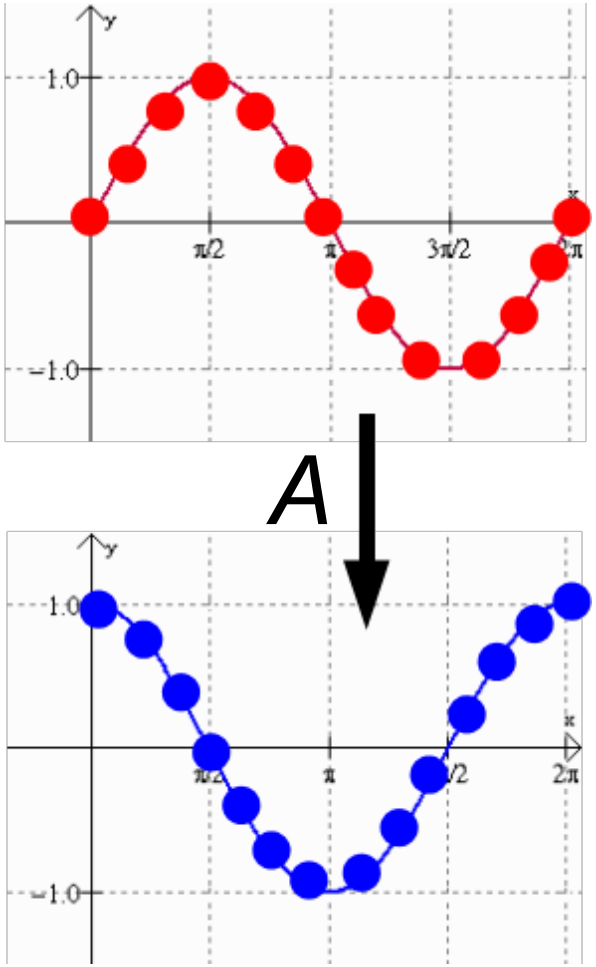
$$A\mathbf{f} = \mathbf{f'}$$

Example: Discrete Derivative

Continuous



Discrete



Discrete 2nd Derivative

Continuous

- Function: f
- Operator: $\frac{d^2}{dx^2}$
- Applying operator:

$$\frac{d^2 f}{dx^2} = f''$$

Discrete

- Vector: $\mathbf{f} = [\, f(x_1), f(x_2) \ldots f(x_n)]$
- Matrix:

$$L = \frac{1}{h^2}\begin{bmatrix} -2 & 1 & 0 & \cdots & 0 & 0 \\ 1 & -2 & 1 & \cdots & 0 & 0 \\ & 1 & -2 & \cdots & 0 & 0 \\ \vdots & & & \ddots & & \vdots \\ & & 0 & & -2 & \\ 0 & 0 & \cdots & & 1 & -2 \end{bmatrix}$$

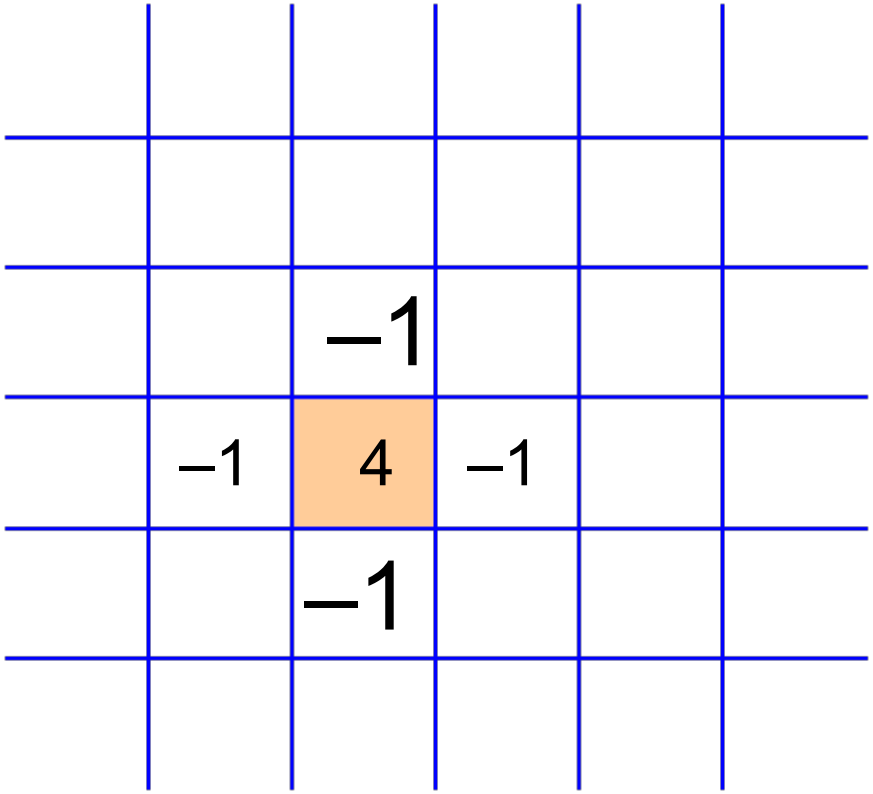
- Applying matrix:

$$L\mathbf{f} = \mathbf{f}''$$

$$f''(x) = \lim_{h \to 0} \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}$$

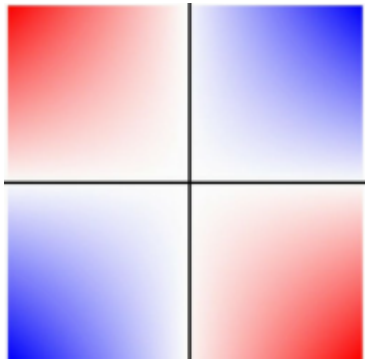
Discrete 2D Laplacian

- The Laplacian is computed via differences of a cell from its neighbours

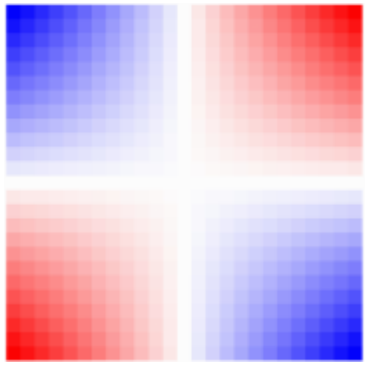


Operators in higher dimensions

- The underlying function space can have a higher-dimensional domain



Continuous function



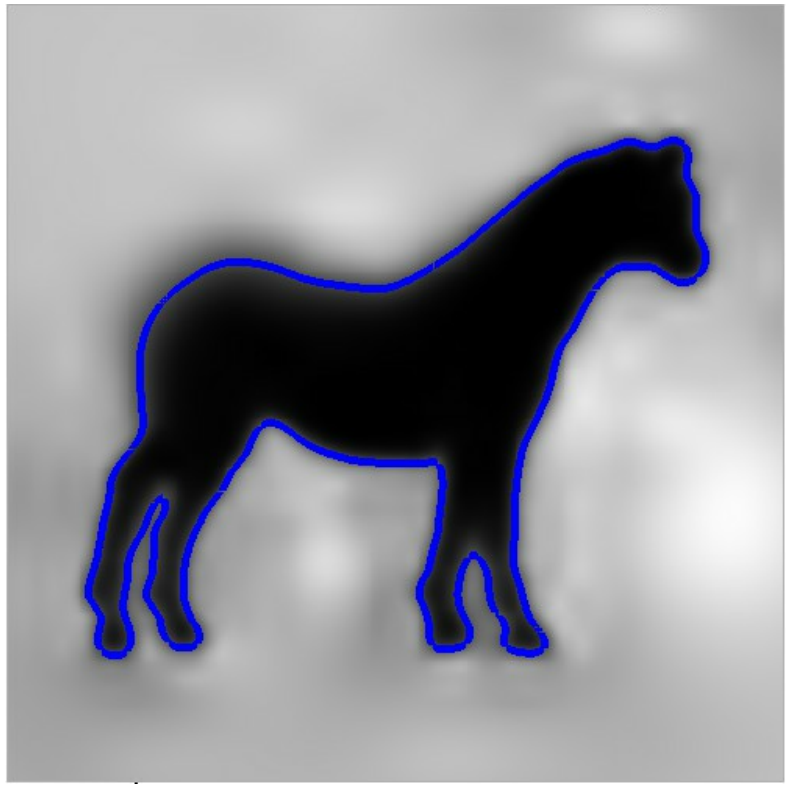
Discrete approximation

$$\begin{bmatrix} -4 & 1 & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & -4 & 1 & \cdot & 1 & \cdot & \cdot & \cdot & \cdot \\ \cdot & 1 & -4 & \cdot & \cdot & 1 & \cdot & \cdot & \cdot \\ 1 & \cdot & \cdot & -4 & 1 & \cdot & 1 & \cdot & \cdot \\ \cdot & 1 & \cdot & 1 & -4 & 1 & \cdot & 1 & \cdot \\ \cdot & \cdot & 1 & \cdot & 1 & -4 & \cdot & \cdot & 1 \\ \cdot & \cdot & \cdot & 1 & \cdot & \cdot & -4 & 1 & \cdot \\ \cdot & \cdot & \cdot & \cdot & 1 & \cdot & 1 & -4 & 1 \\ \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & 1 & -4 \end{bmatrix}$$

2D discrete Laplace operator

Final step of Poisson reconstruction

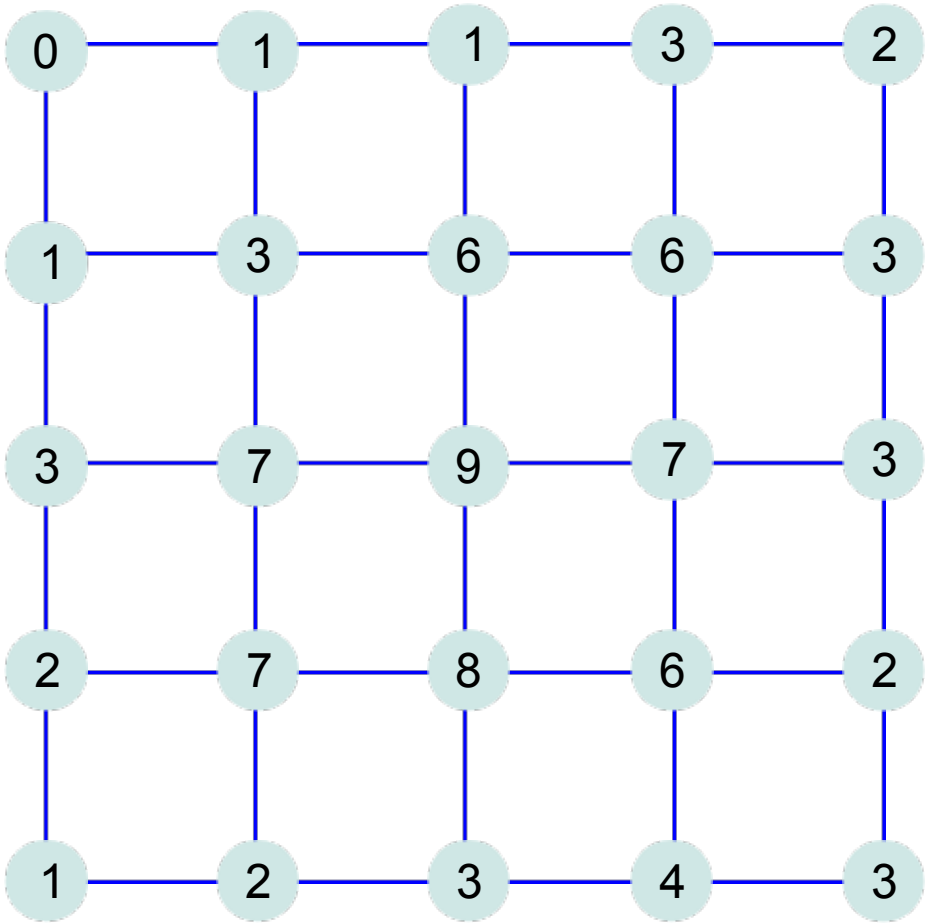
Output of Poisson Reconstruction



ToDo: Isosurface

Isocontours

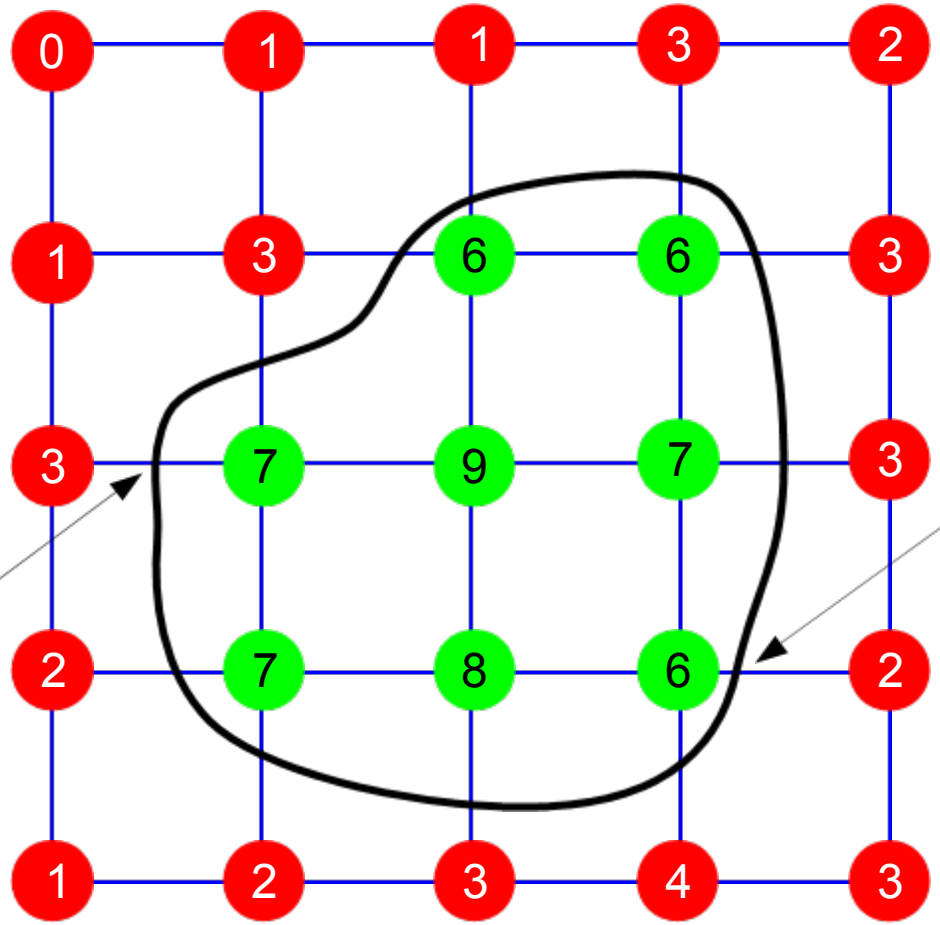
- **Data:** 2D structured grid of scalar values



Slides adapted from Toby Breckon

Isocontours

- The 5-level set:



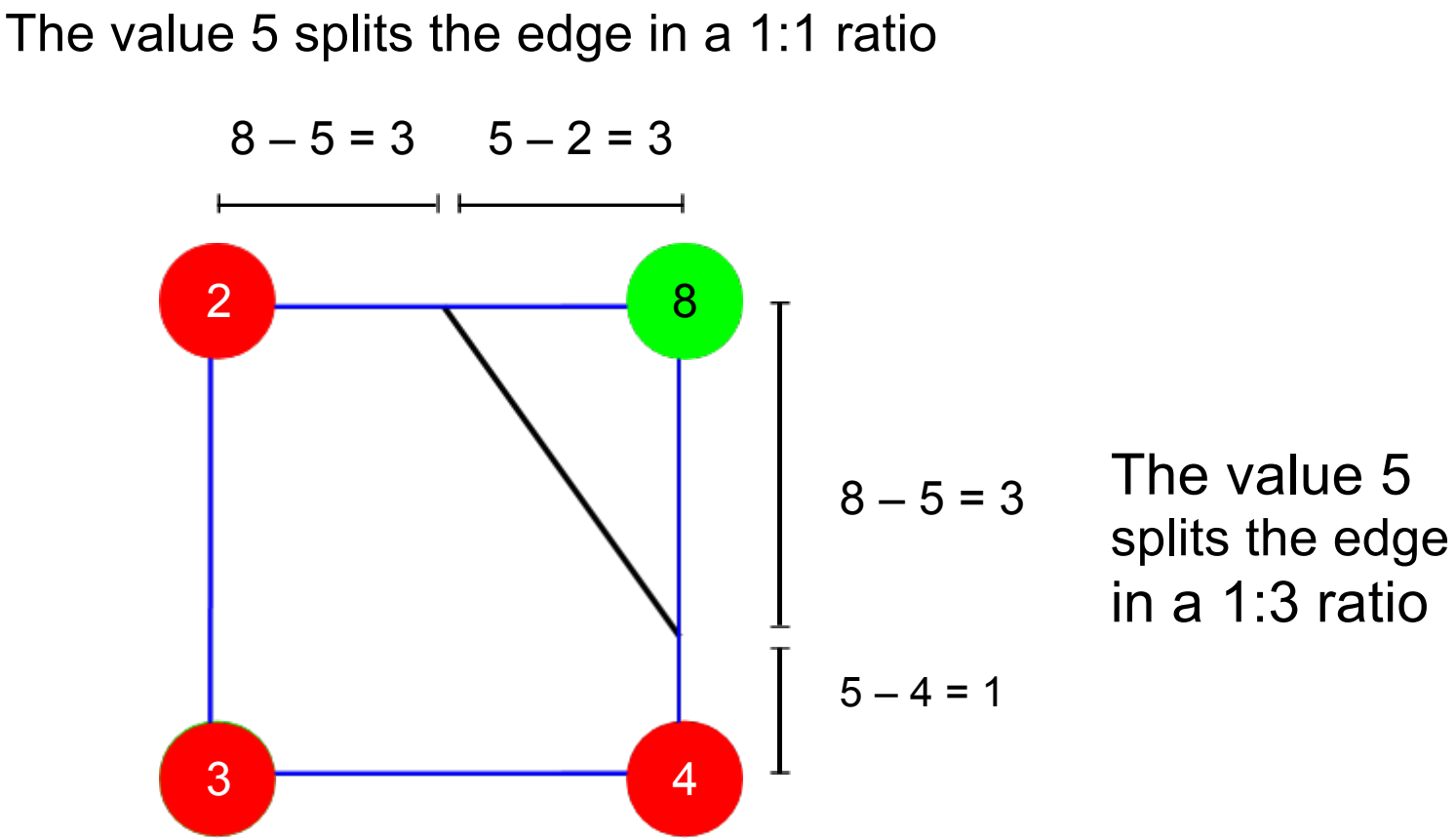
Bisects the edge, since 5 is equidistant from 3 and 7

Splits edge asymetrically, since 5 is closer to 6 than to 2

Slides adapted from Toby Breckon

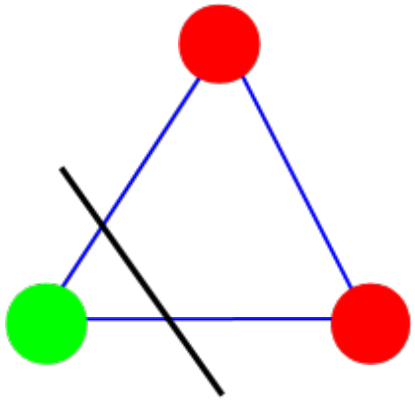
Where is the intersection?

- Find location of contour intersection with edge by interpolating vertex values

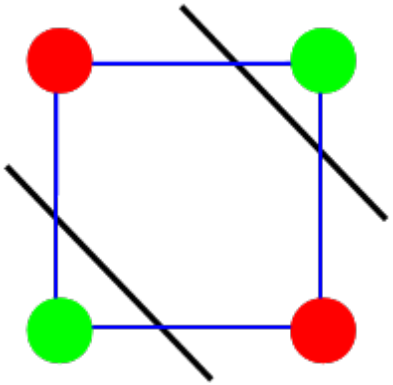


Isocontours: Ambiguity

- Where is the contour?

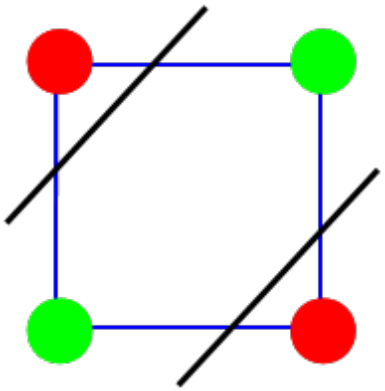


Triangular cell:
No ambiguities



"Split" green (inner) region

or

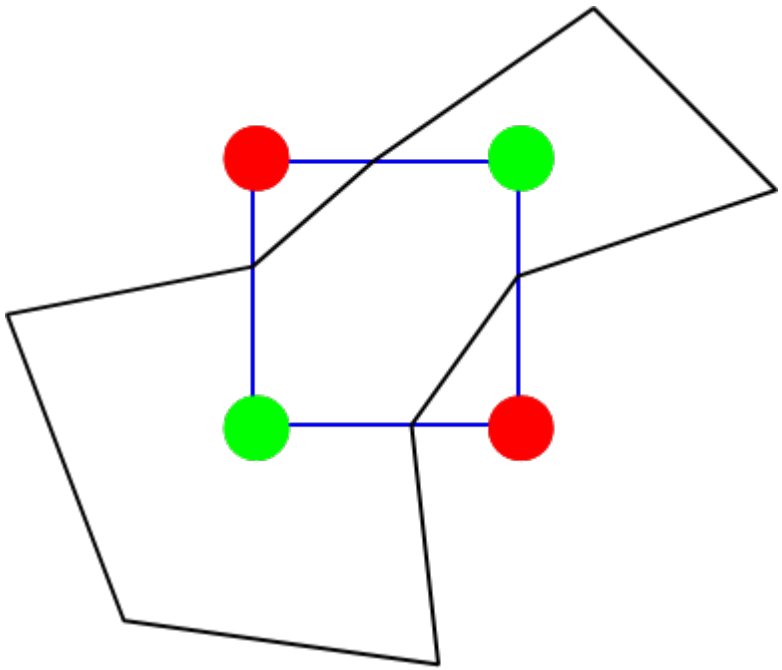


"Join" green (inner) region

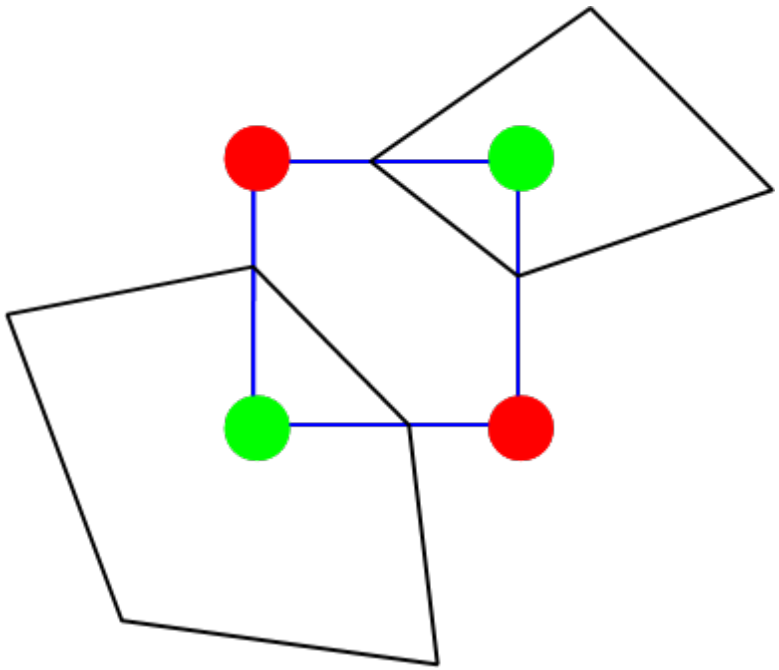
Square cell:
2 ambiguous cases

Isocontours: Ambiguity

- Where is the contour?



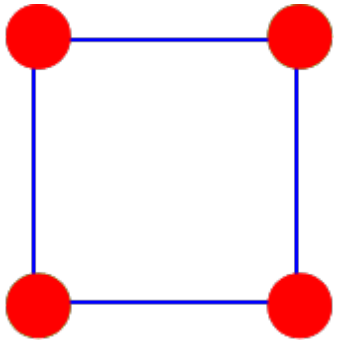
Join

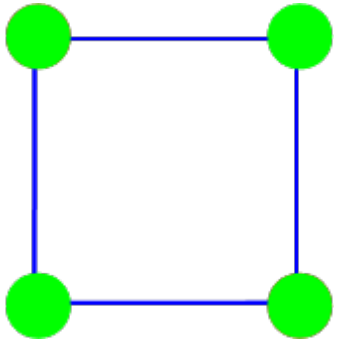


Split

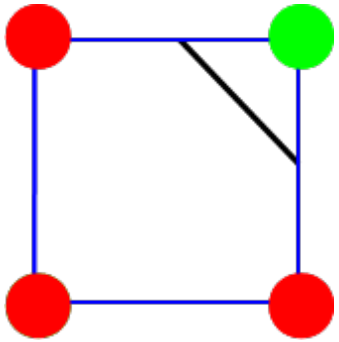
Isocontours: Cell Confgurations

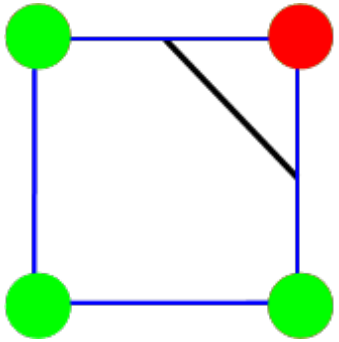
No intersections



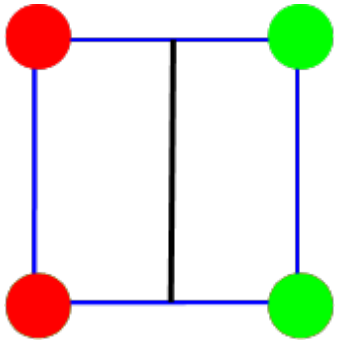


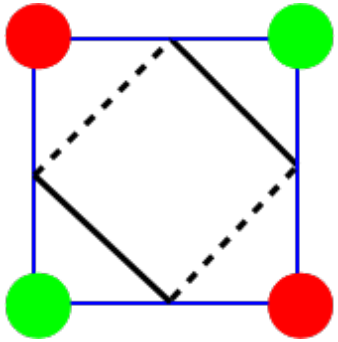
1 vertex diferent





2 vertices diferent





Ambiguous case

2^4 = 16 different possibilities, reducible to just 6 distinct cases after factoring out symmetries

Slides adapted from Toby Breckon

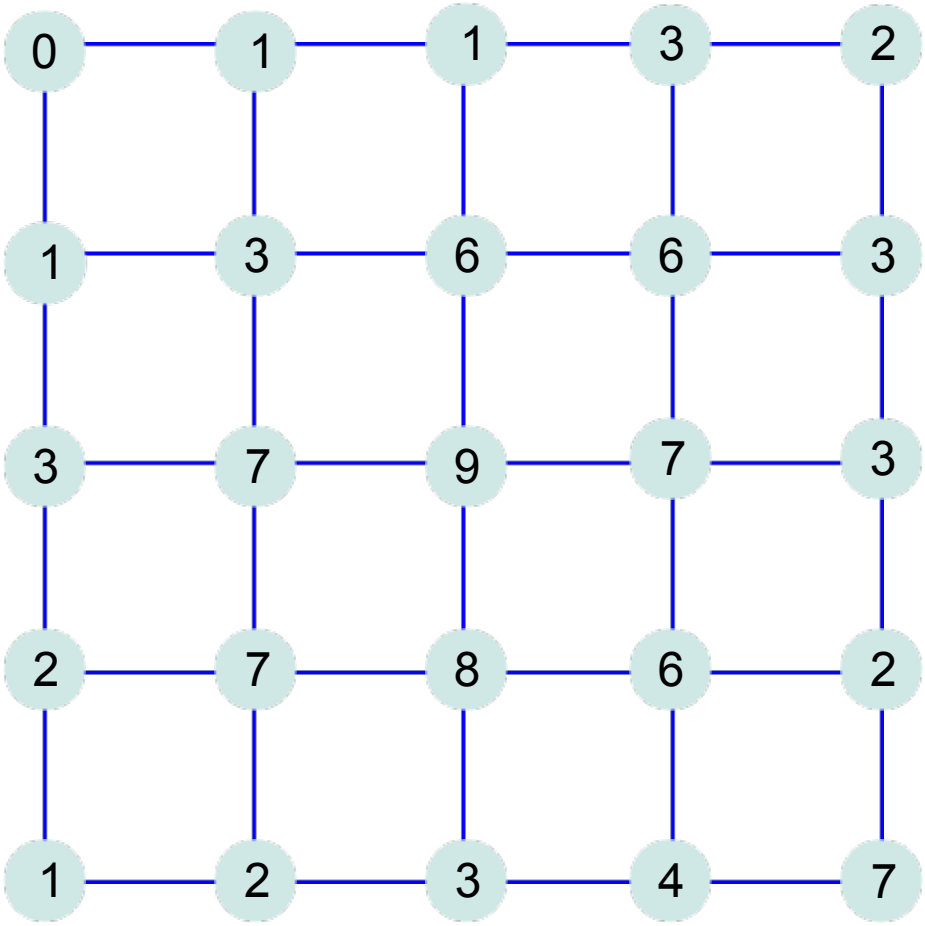
Marching Squares Algorithm

- Select a starting cell
- Calculate inside/outside state for each vertex
- Classify cell confguration
  - Determine which edges are intersected
- Find exact locations of edge intersections
- Link up intersections to produce contour segment(s)
- Move (or “march”) into next cell and repeat
  - … until all cells have been visited

Slides adapted from Toby Breckon

Example: Marching Squares

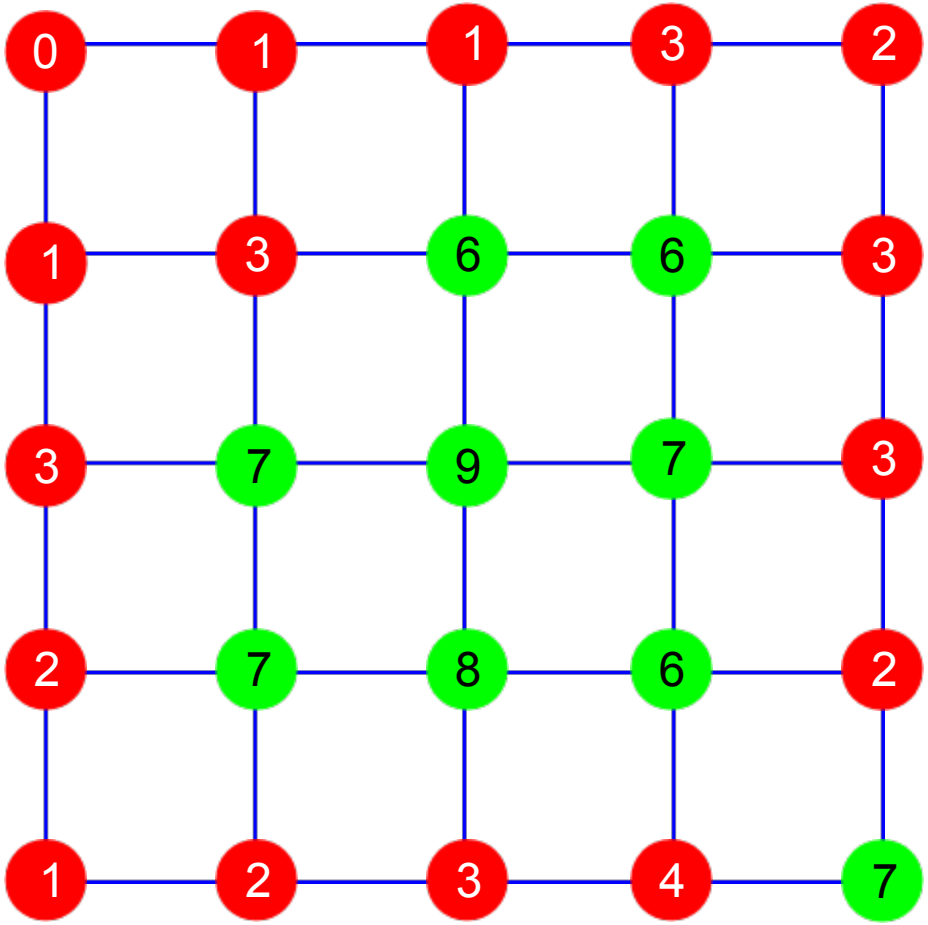
Find 5-contour of function represented by its values at vertices of a uniform grid



Slides adapted from Toby Breckon

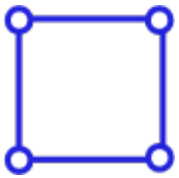
Step 1: Classify vertices

Green: inside
Red: outside



Slides adapted from Toby Breckon

Step 2: Classify cells

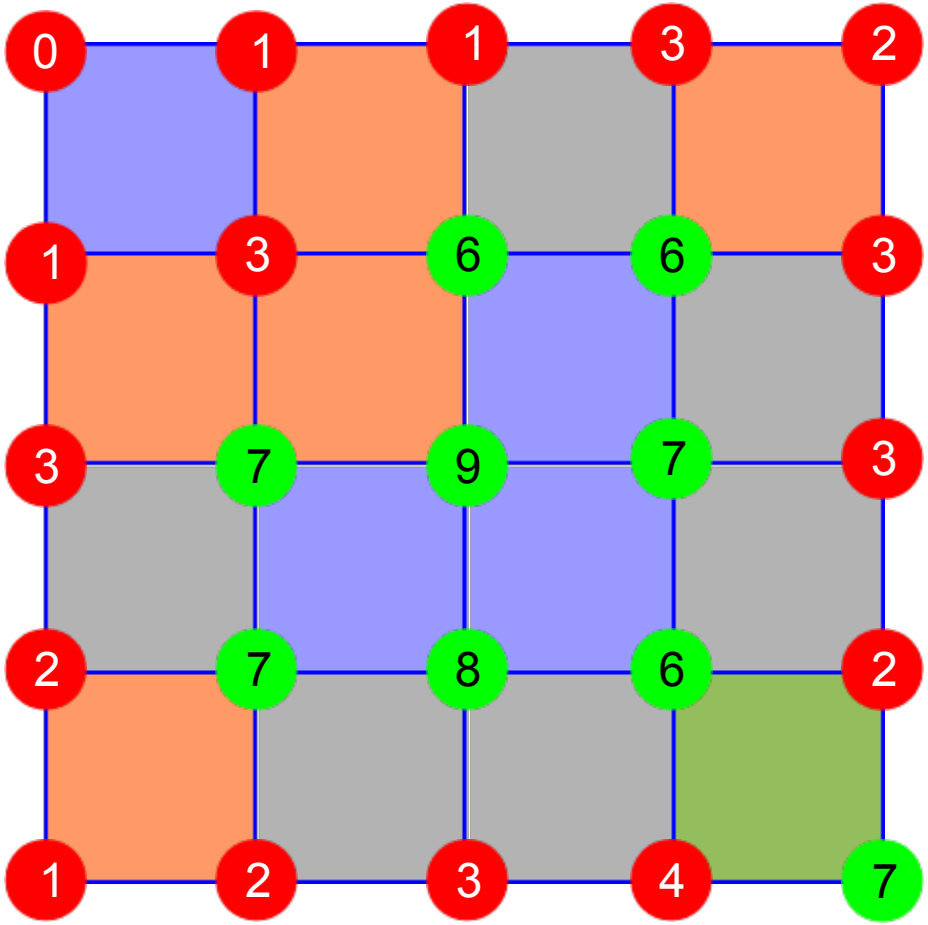


No intersections

Adjacent edges

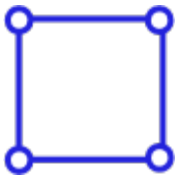
Opposite edges

Ambiguous



Slides adapted from Toby Breckon

Step 3: Interpolate contour intersections

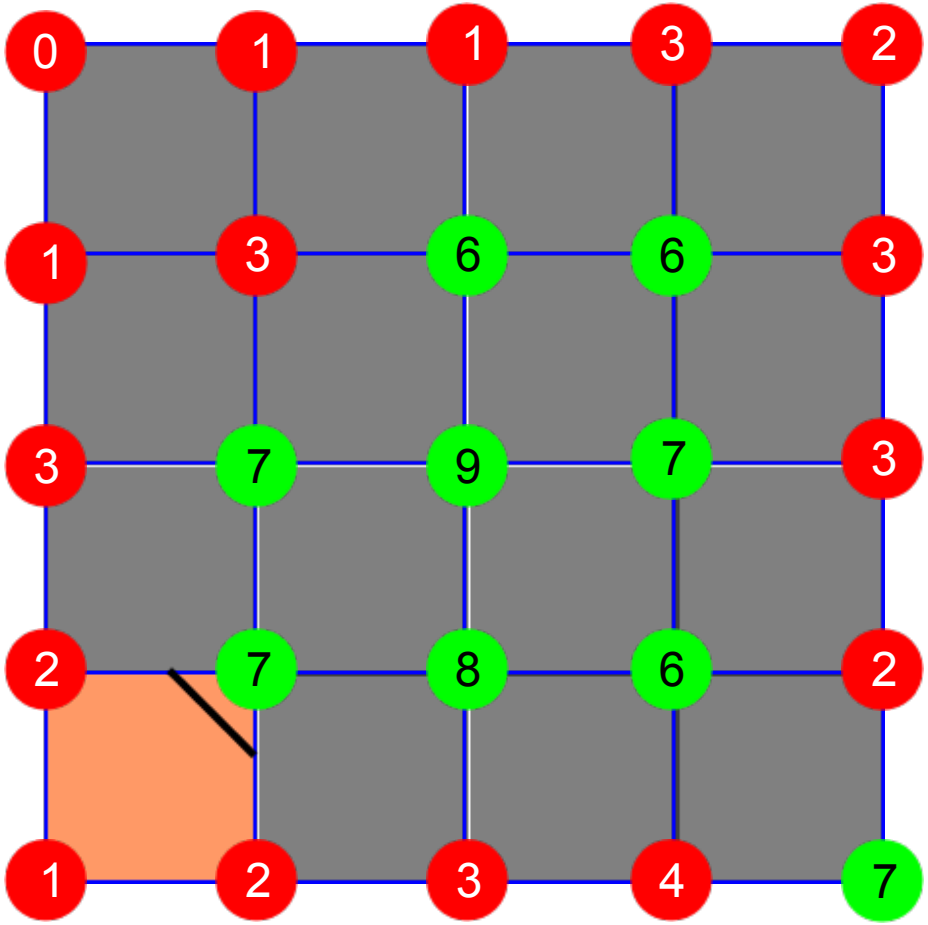


No intersections

Adjacent edges

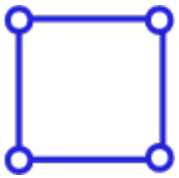
Opposite edges

Ambiguous



Slides adapted from Toby Breckon

Step 3: Interpolate contour intersections

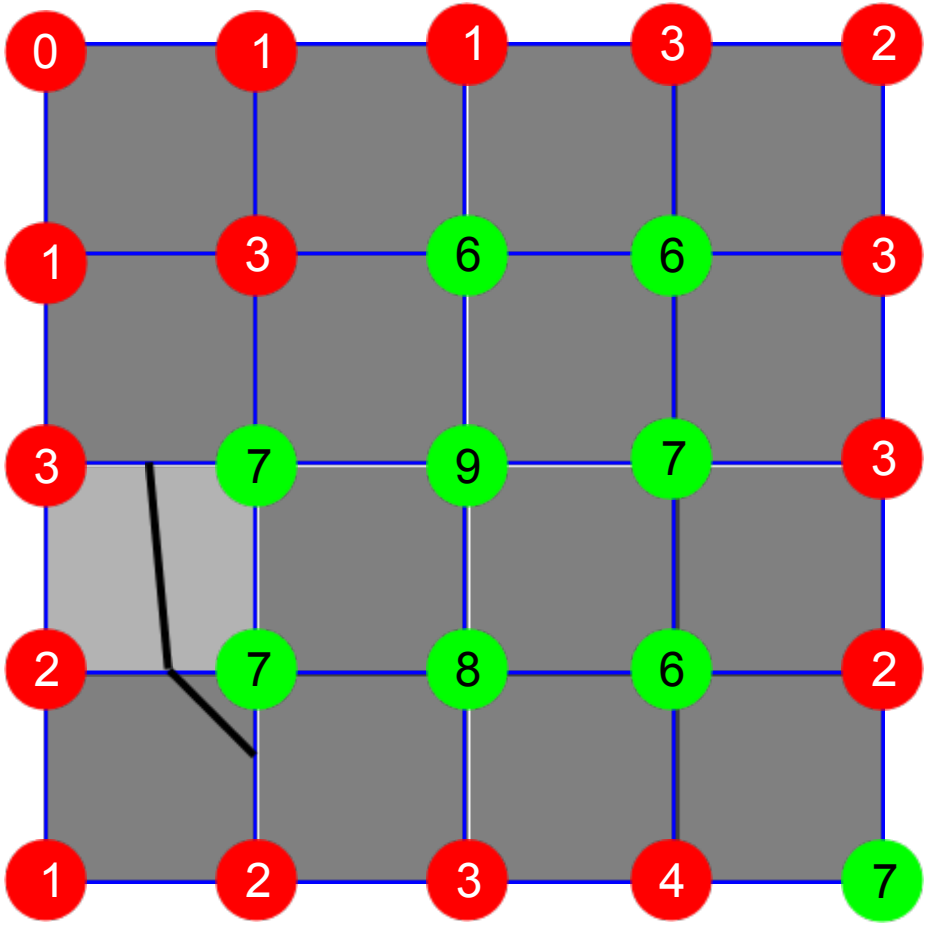


No intersections

Adjacent edges

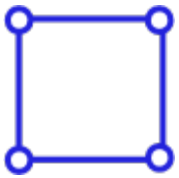
Opposite edges

Ambiguous



Slides adapted from Toby Breckon

Step 3: Interpolate contour intersections

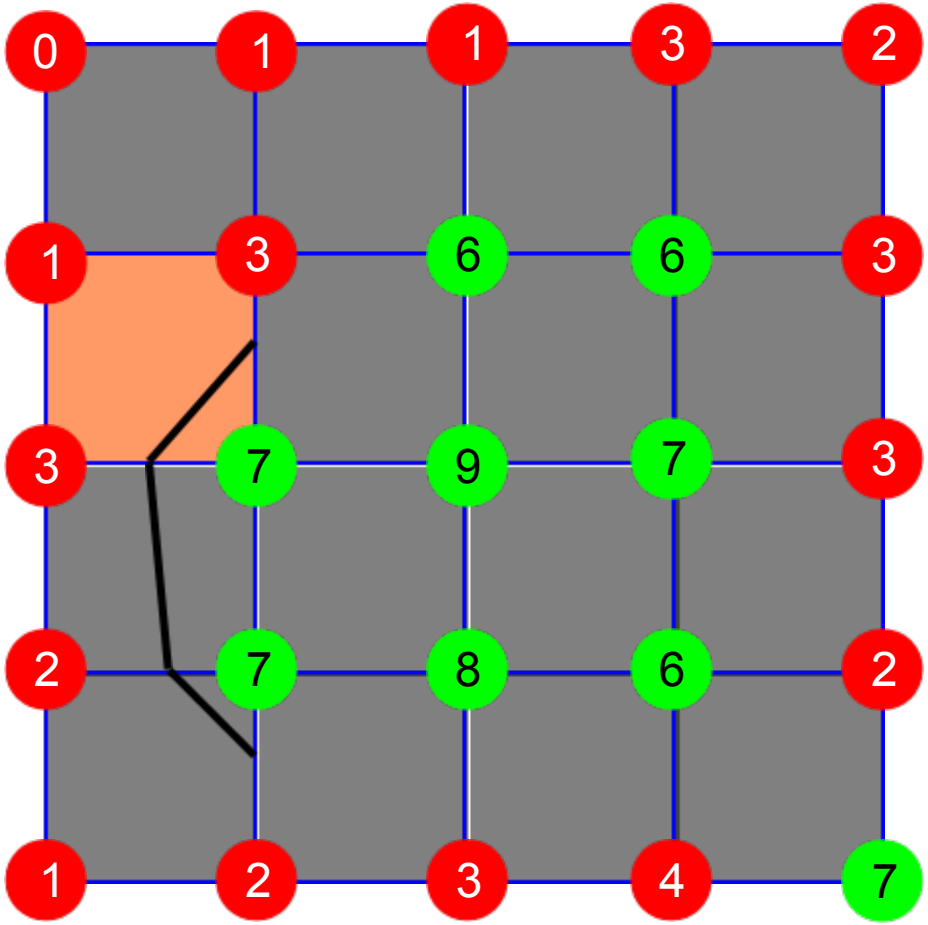


No intersections

Adjacent edges

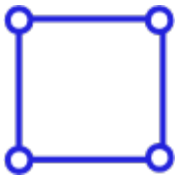
Opposite edges

Ambiguous



Slides adapted from Toby Breckon

Step 3: Interpolate contour intersections

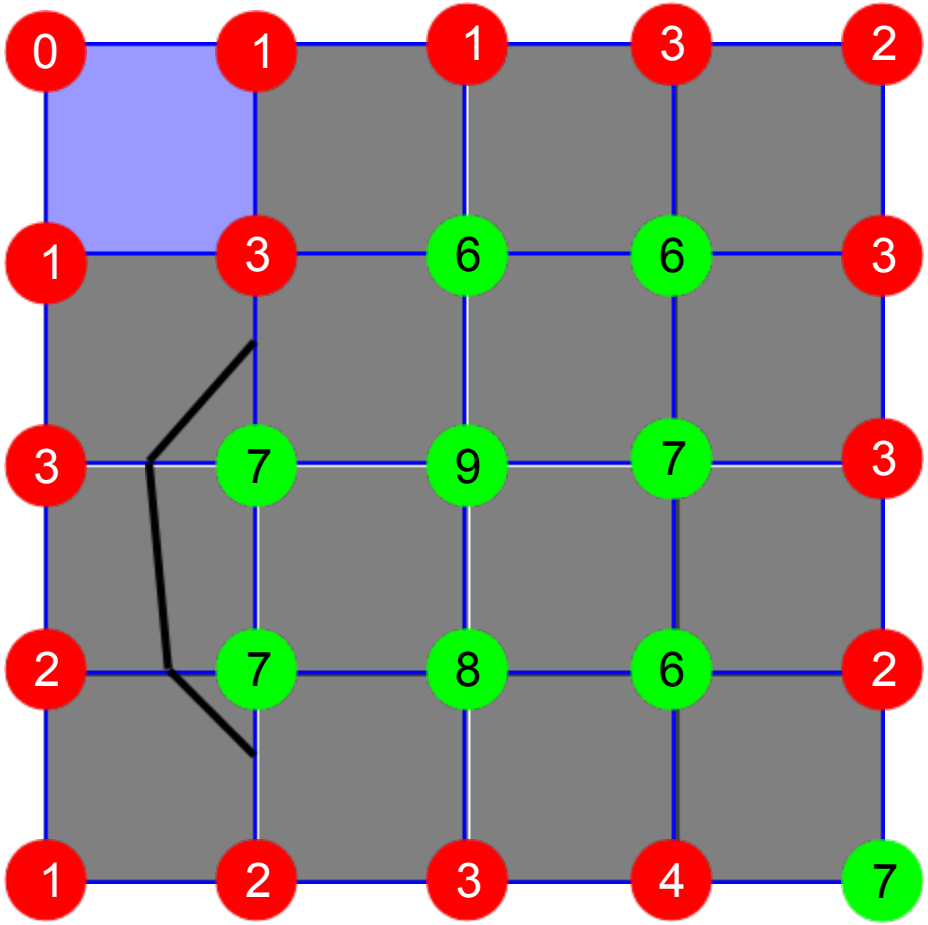


No intersections

Adjacent edges

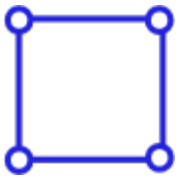
Opposite edges

Ambiguous



Slides adapted from Toby Breckon

Step 3: Interpolate contour intersections

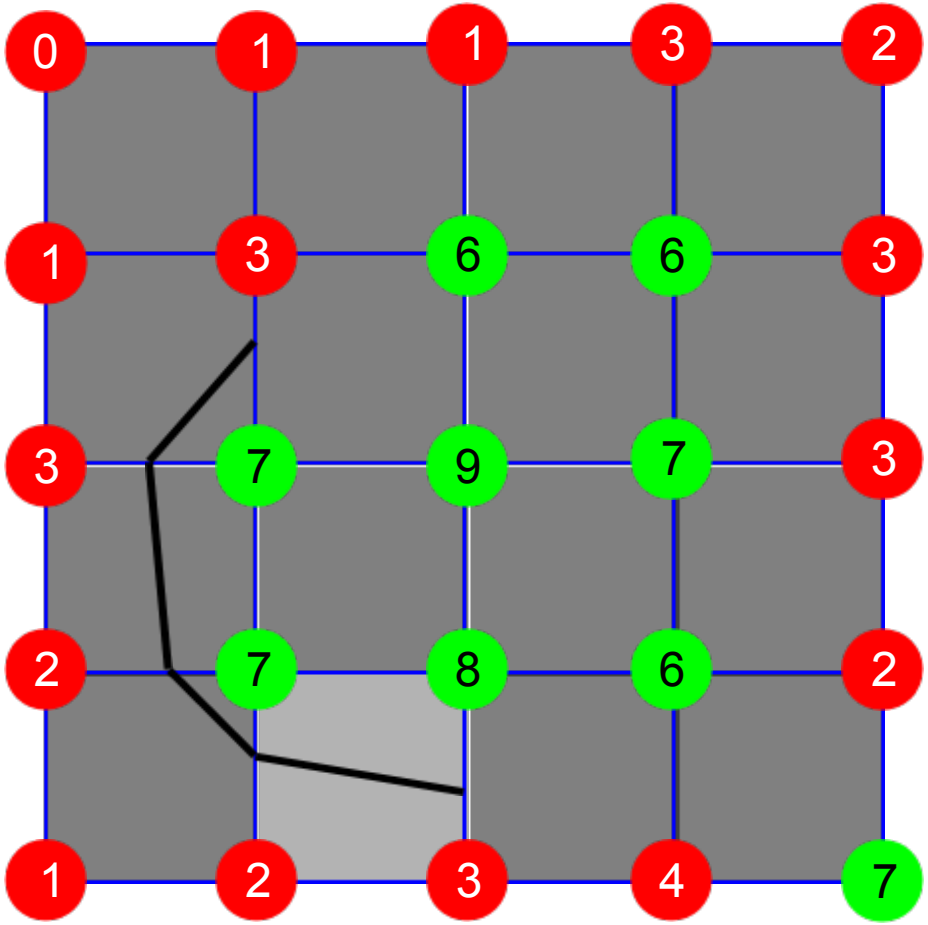


No intersections

Adjacent edges

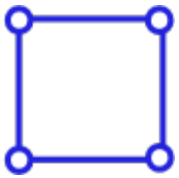
Opposite edges

Ambiguous



Slides adapted from Toby Breckon

Step 3: Interpolate contour intersections

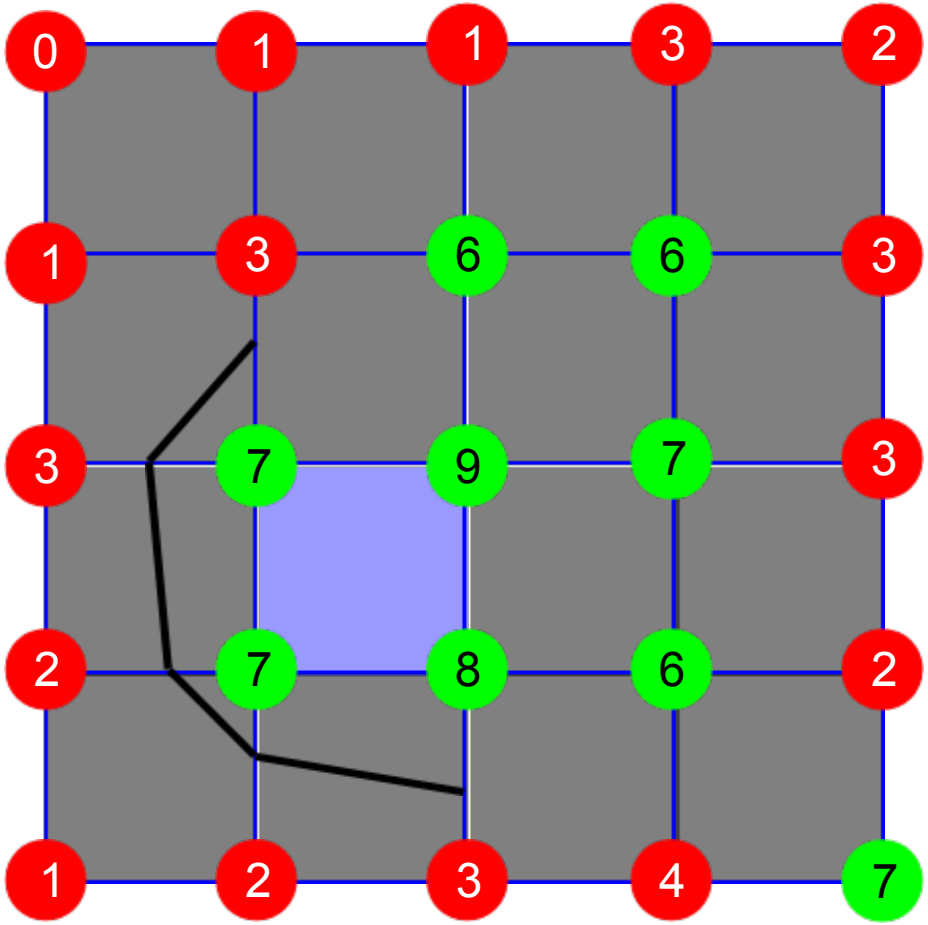


No intersections

Adjacent edges

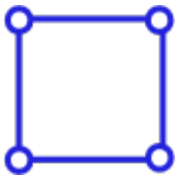
Opposite edges

Ambiguous



Slides adapted from Toby Breckon

Step 3: Interpolate contour intersections

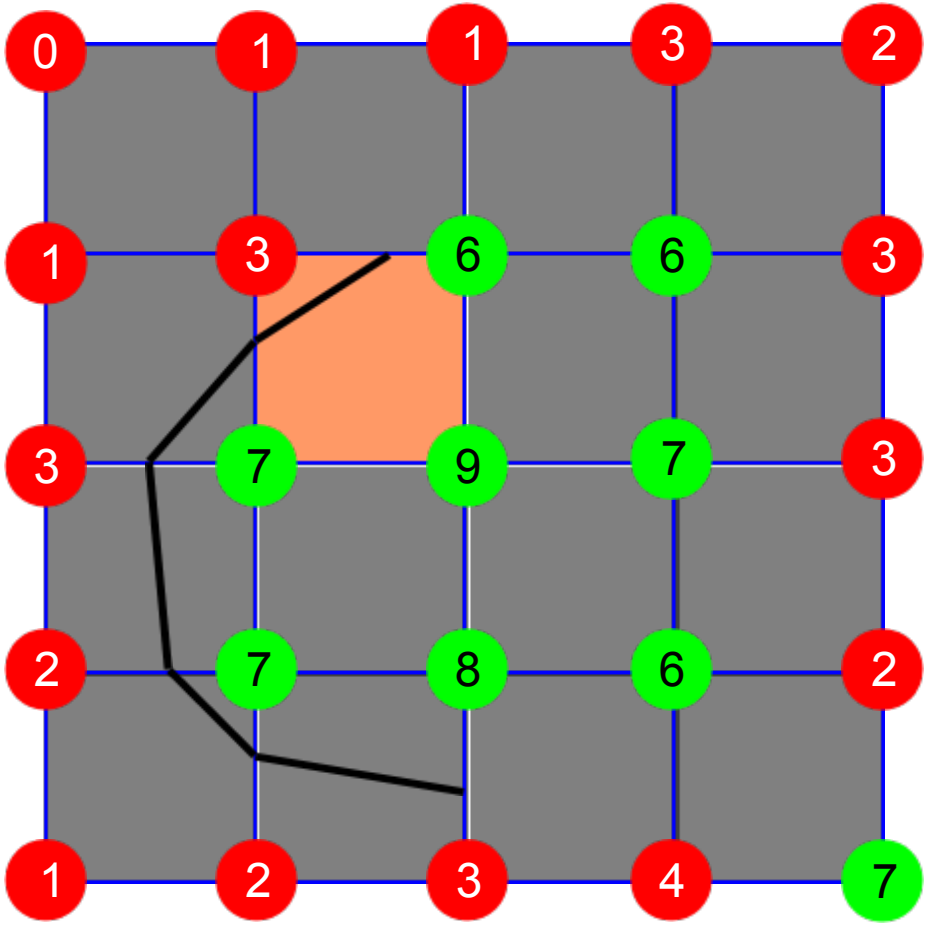


No intersections

Adjacent edges

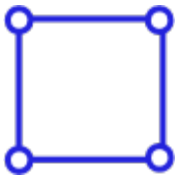
Opposite edges

Ambiguous



Slides adapted from Toby Breckon

Step 3: Interpolate contour intersections

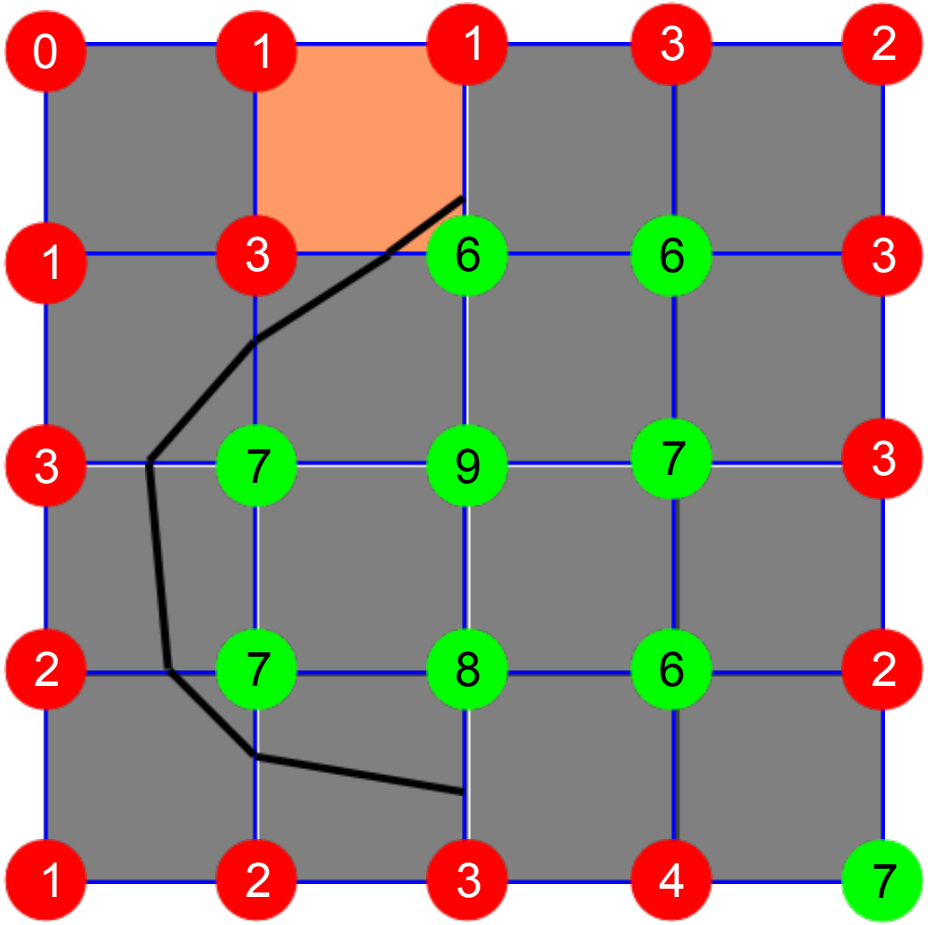


No intersections

Adjacent edges

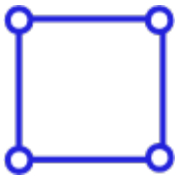
Opposite edges

Ambiguous



Slides adapted from Toby Breckon

Step 3: Interpolate contour intersections

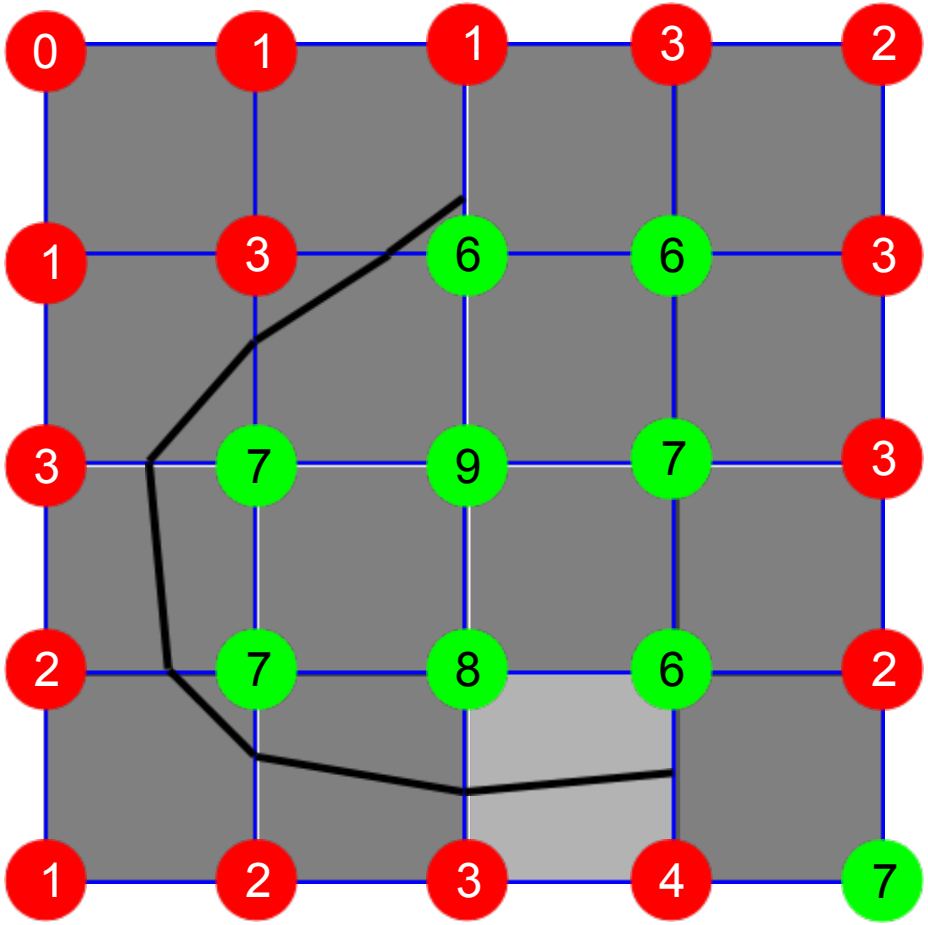


No intersections

Adjacent edges

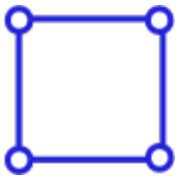
Opposite edges

Ambiguous



Slides adapted from Toby Breckon

Step 3: Interpolate contour intersections

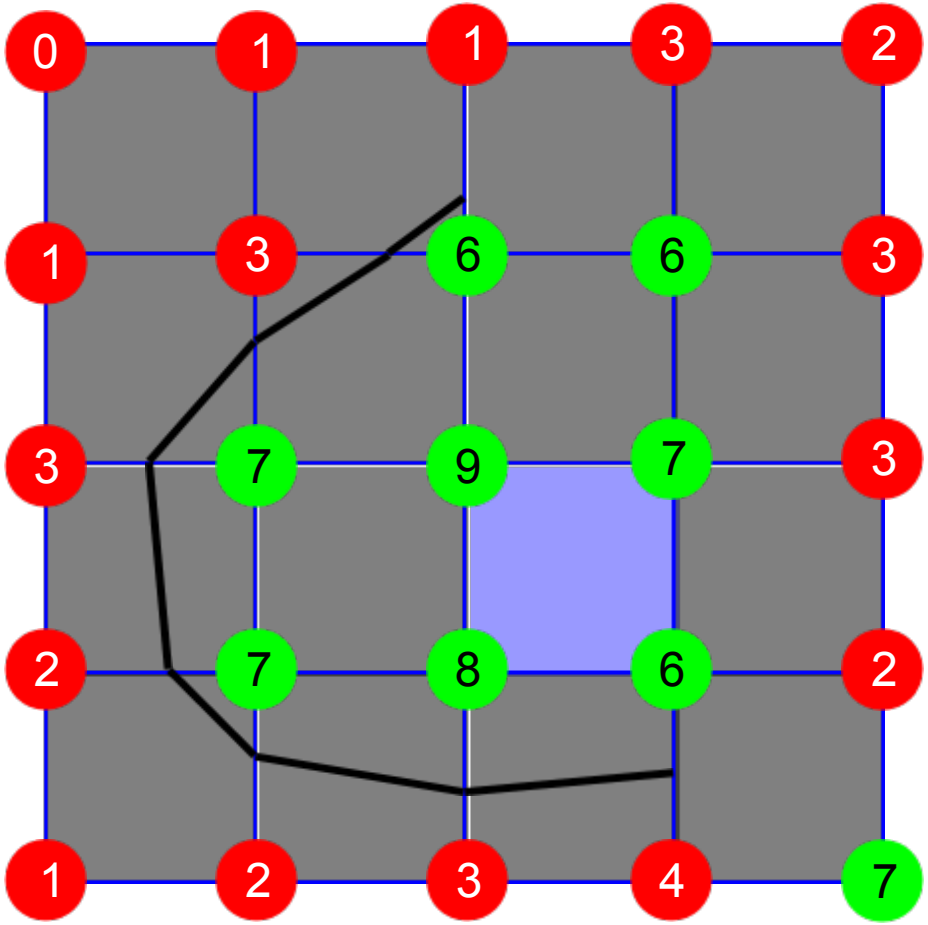


No intersections

Adjacent edges

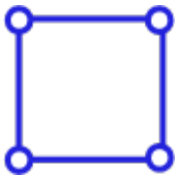
Opposite edges

Ambiguous



Slides adapted from Toby Breckon

Step 3: Interpolate contour intersections

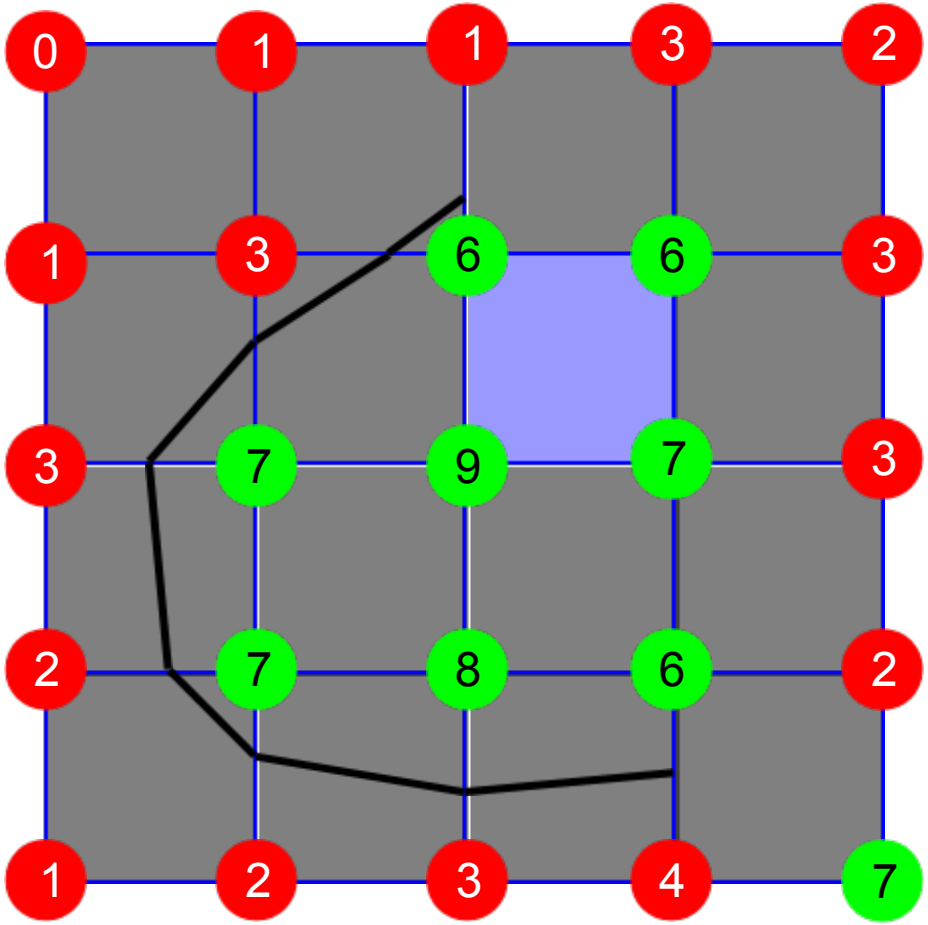


No intersections

Adjacent edges

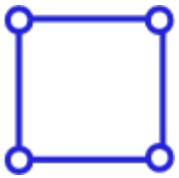
Opposite edges

Ambiguous



Slides adapted from Toby Breckon

Step 3: Interpolate contour intersections

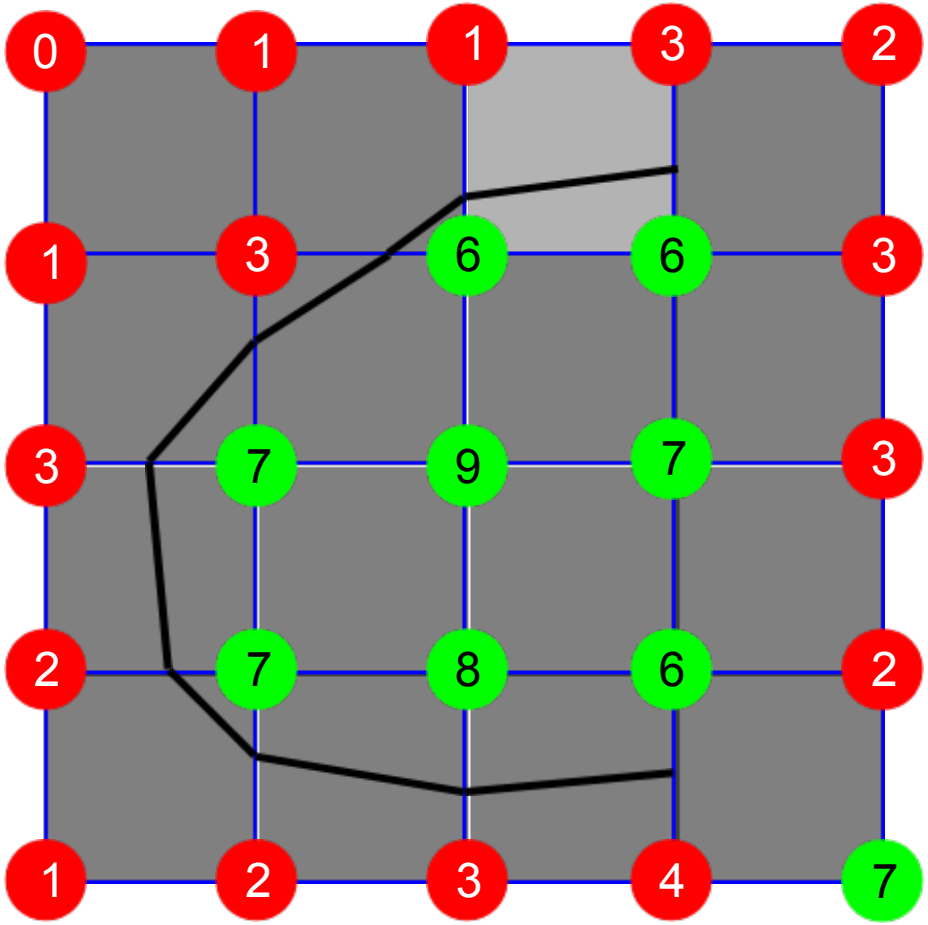


No intersections

Adjacent edges

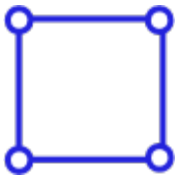
Opposite edges

Ambiguous



Slides adapted from Toby Breckon

Step 3: Interpolate contour intersections

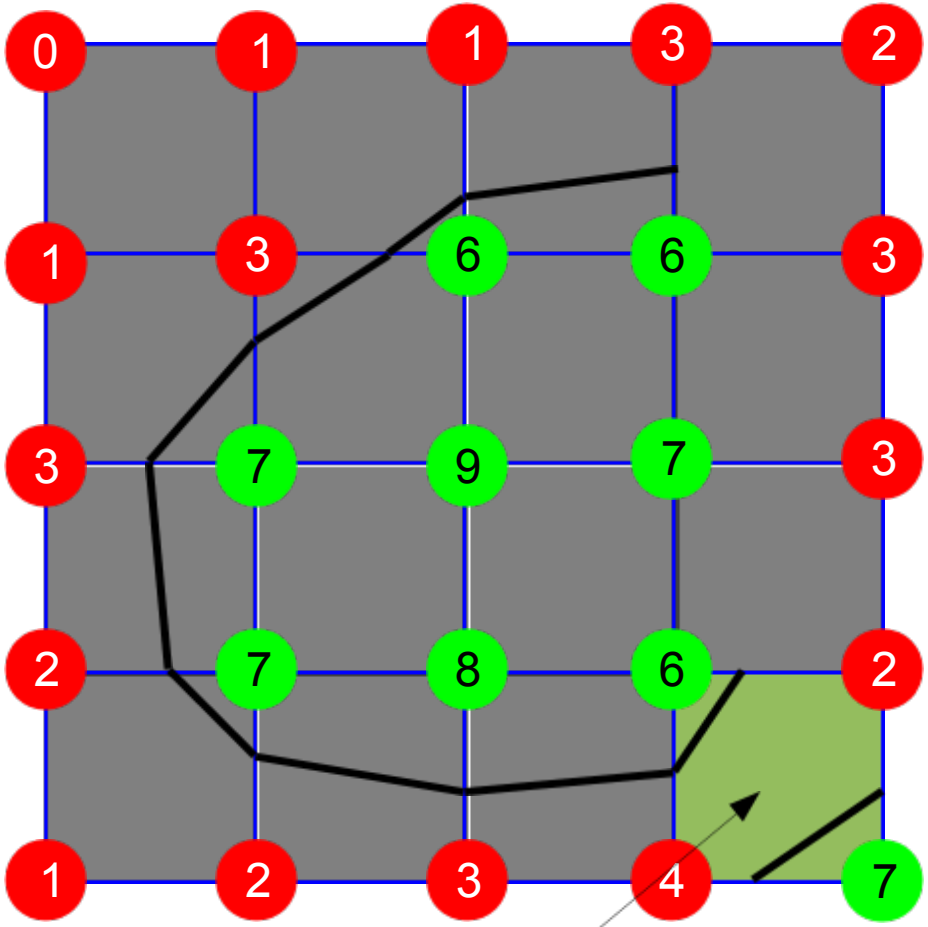


No intersections

Adjacent edges

Opposite edges

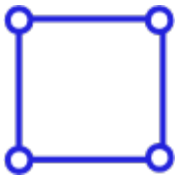
Ambiguous



Arbitrarily choose to split here, instead of join. We could also have gone the other way.

Slides adapted from Toby Breckon

Step 3: Interpolate contour intersections

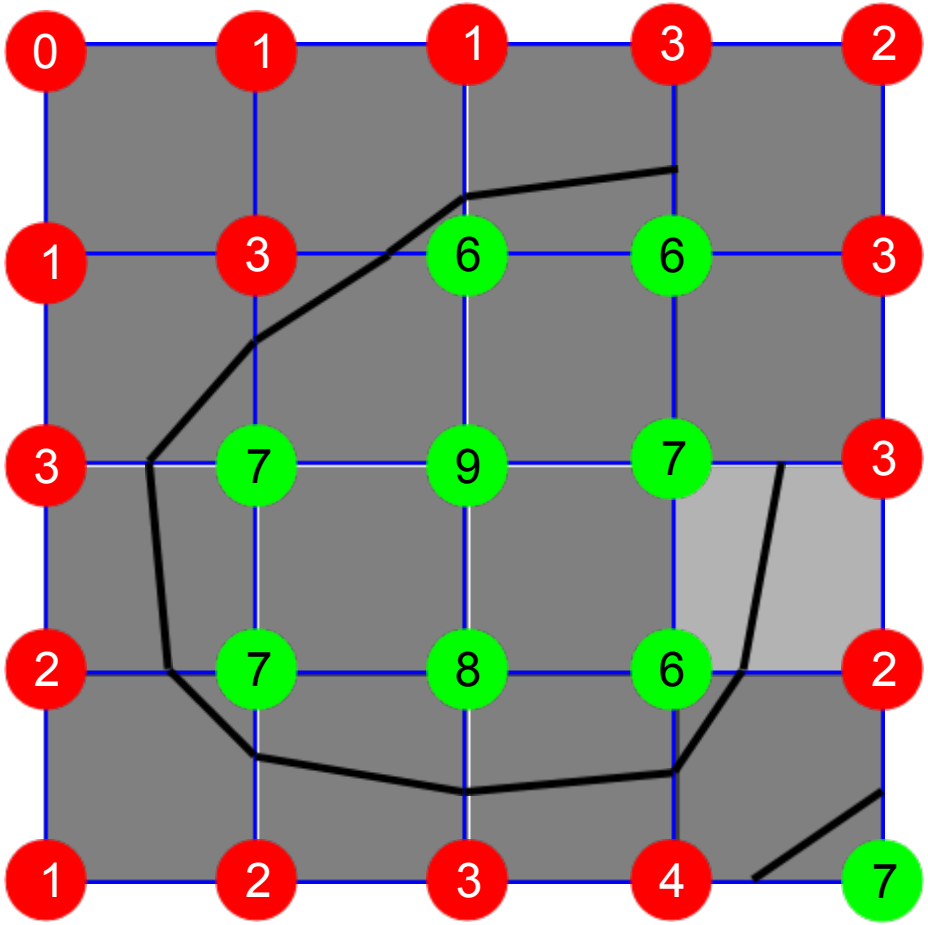


No intersections

Adjacent edges

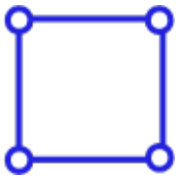
Opposite edges

Ambiguous



Slides adapted from Toby Breckon

Step 3: Interpolate contour intersections

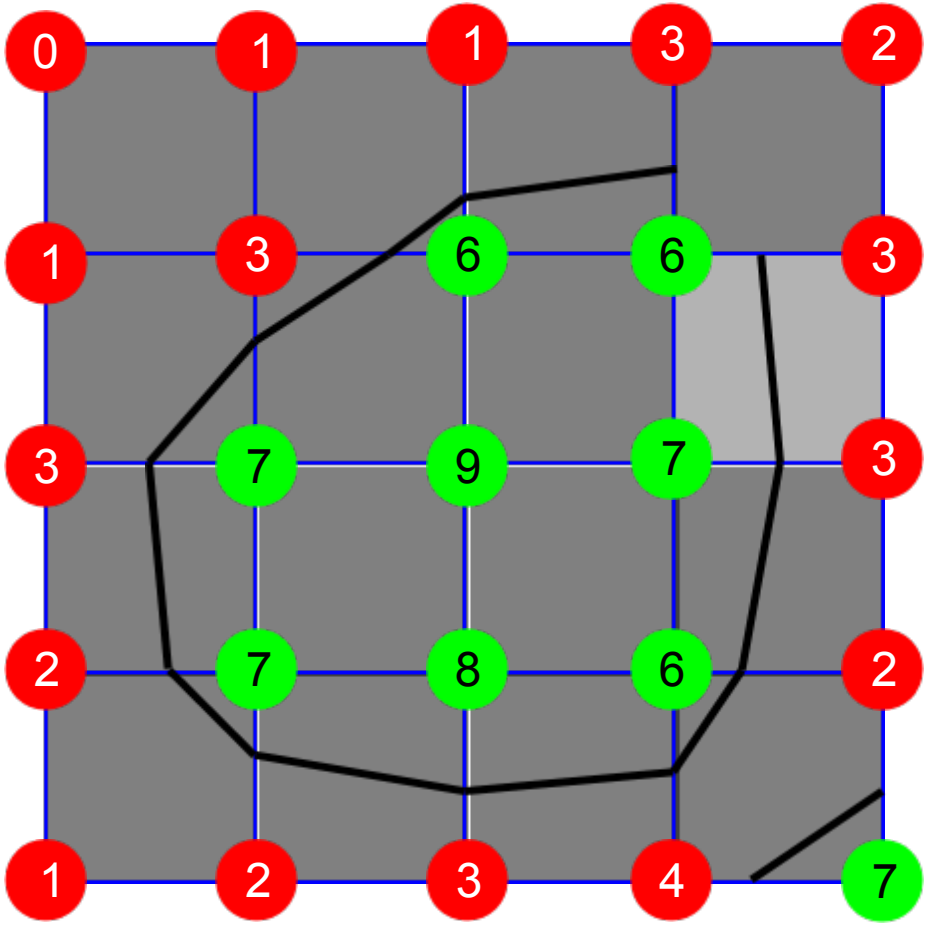


No intersections

Adjacent edges

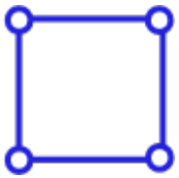
Opposite edges

Ambiguous



Slides adapted from Toby Breckon

Step 3: Interpolate contour intersections

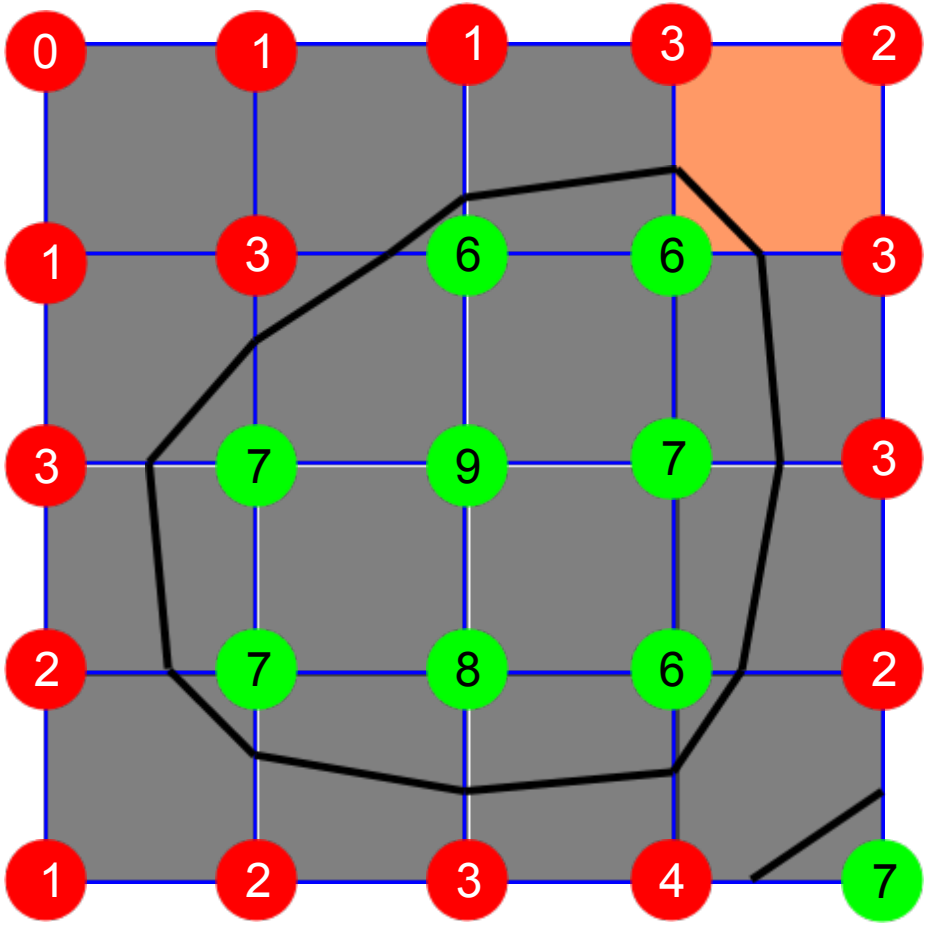


No intersections

Adjacent edges

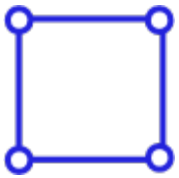
Opposite edges

Ambiguous



Slides adapted from Toby Breckon

Resolving ambiguities

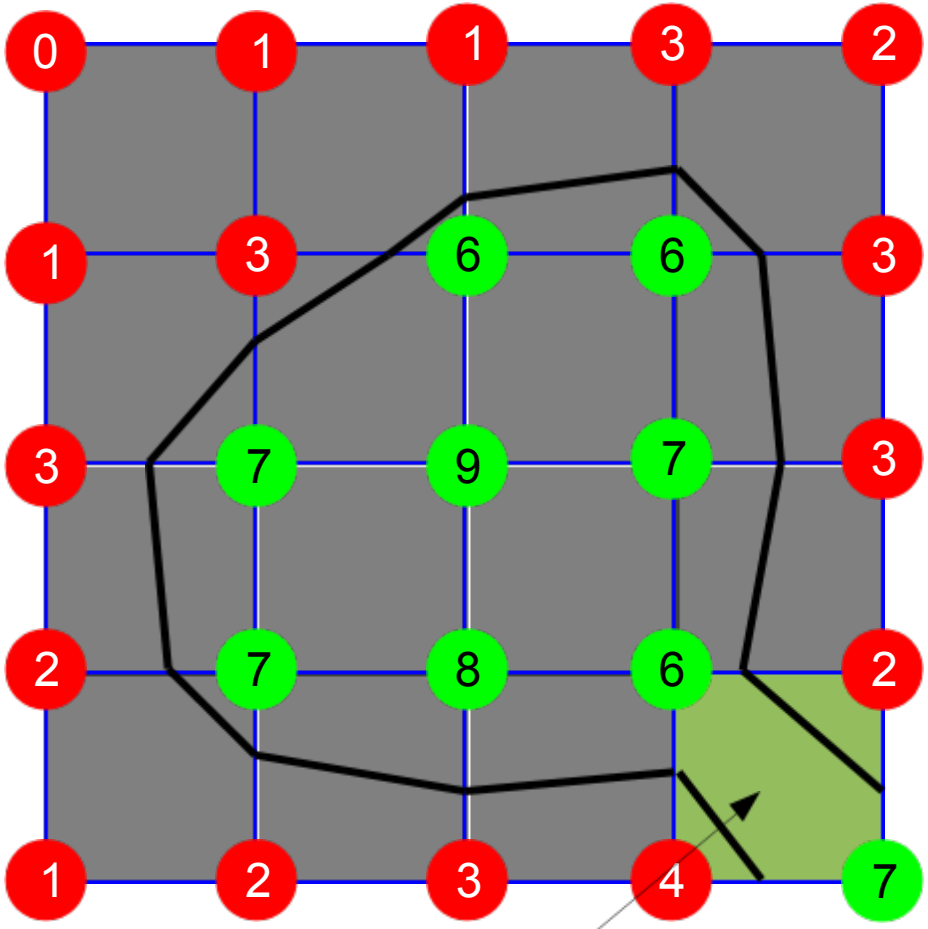


No intersections

Adjacent edges

Opposite edges

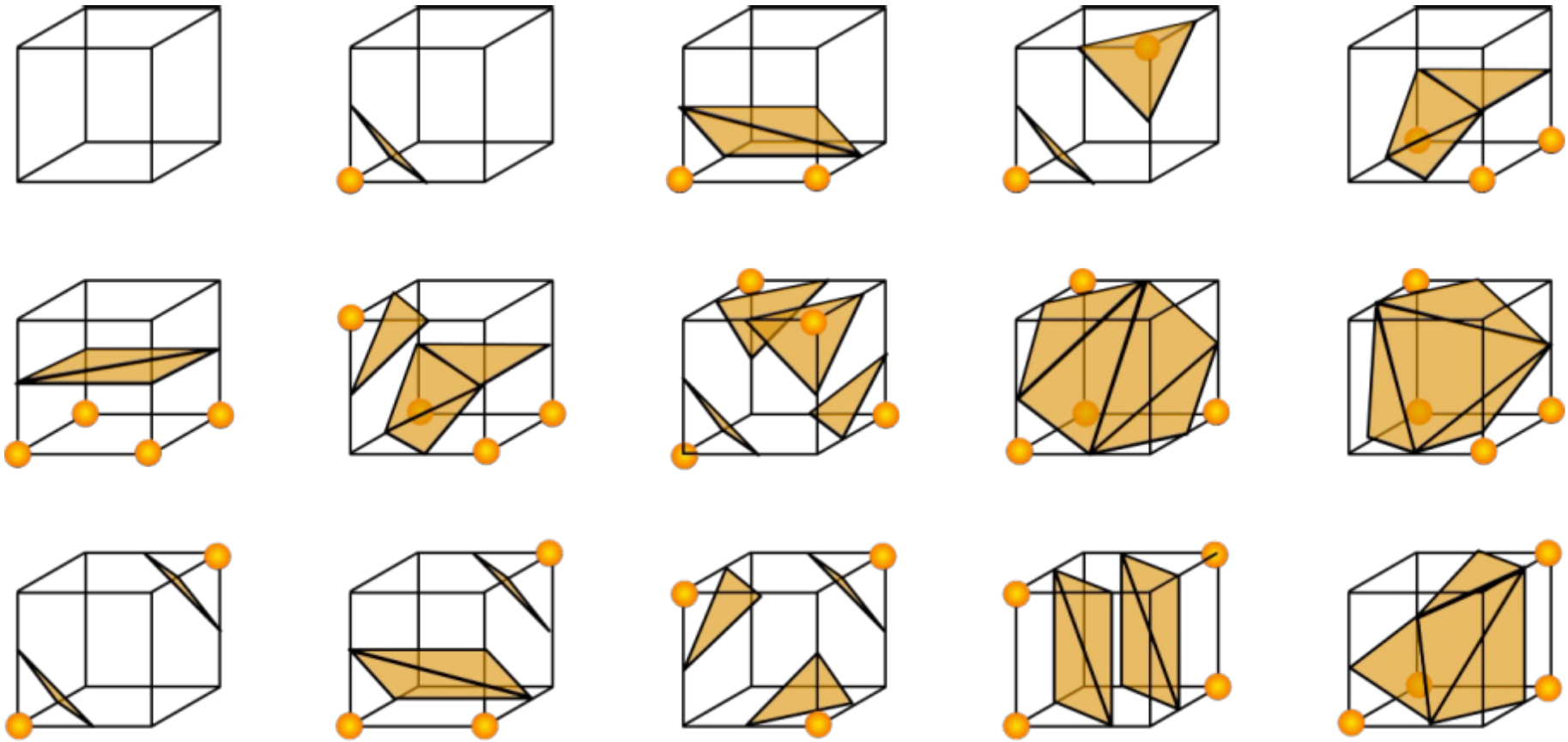
Ambiguous



Slides adapted from Toby Breckon

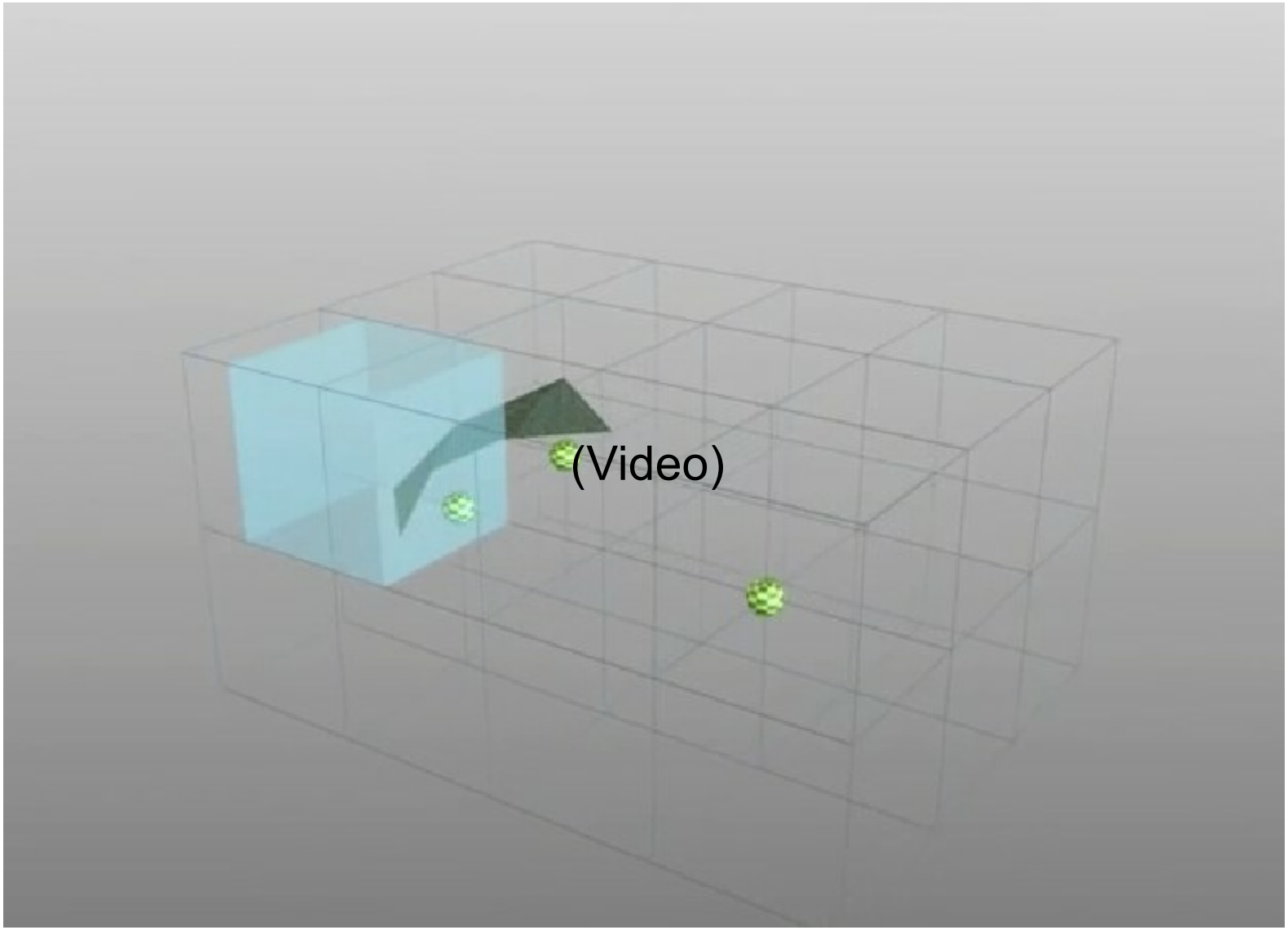
In 3D: Marching Cubes

Exactly the same algorithm, but cells are now cubes (15 distinct confgurations) and output is triangles (or a polygon mix)



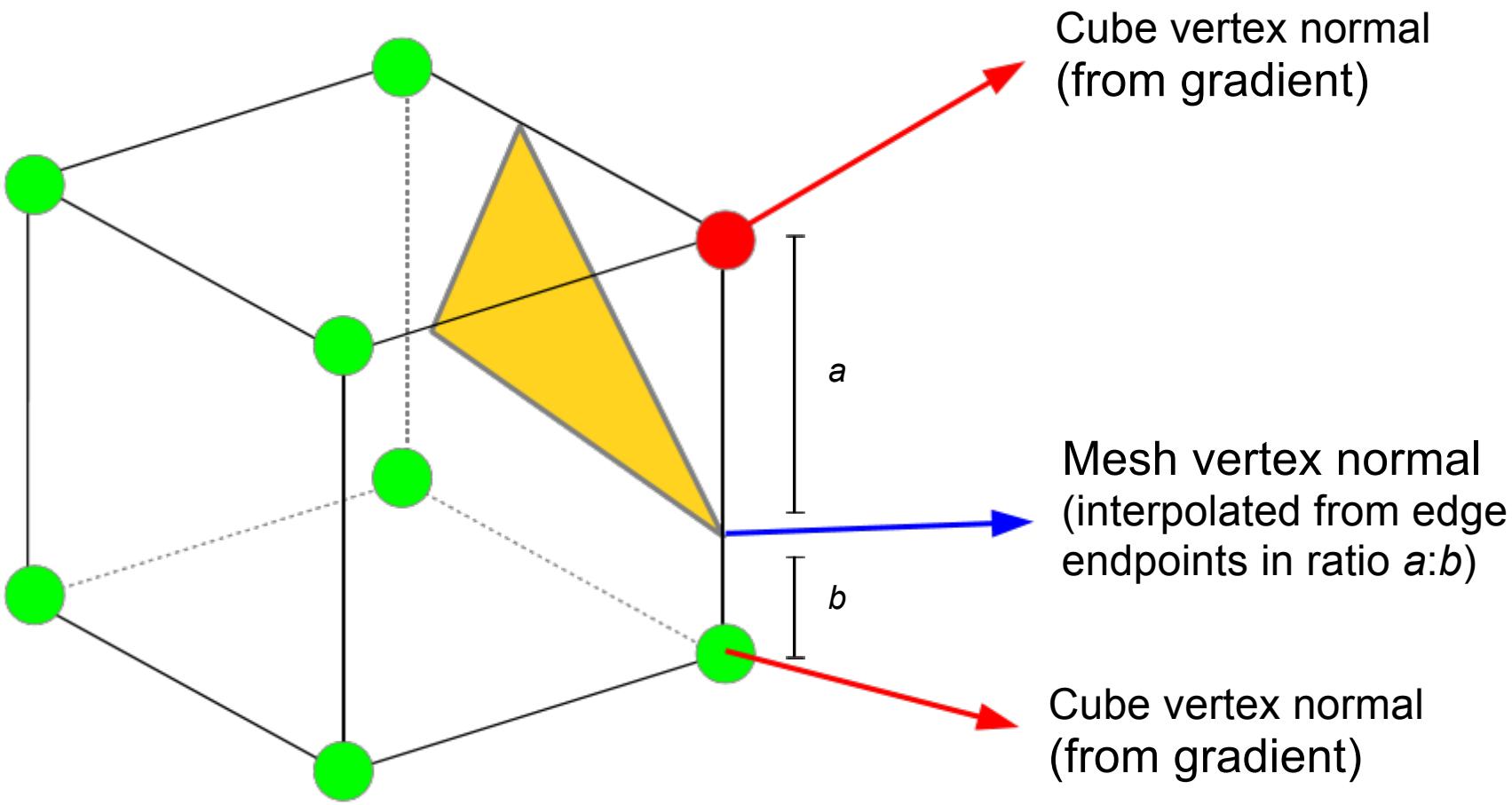
Wikipedia

In 3D: Marching Cubes

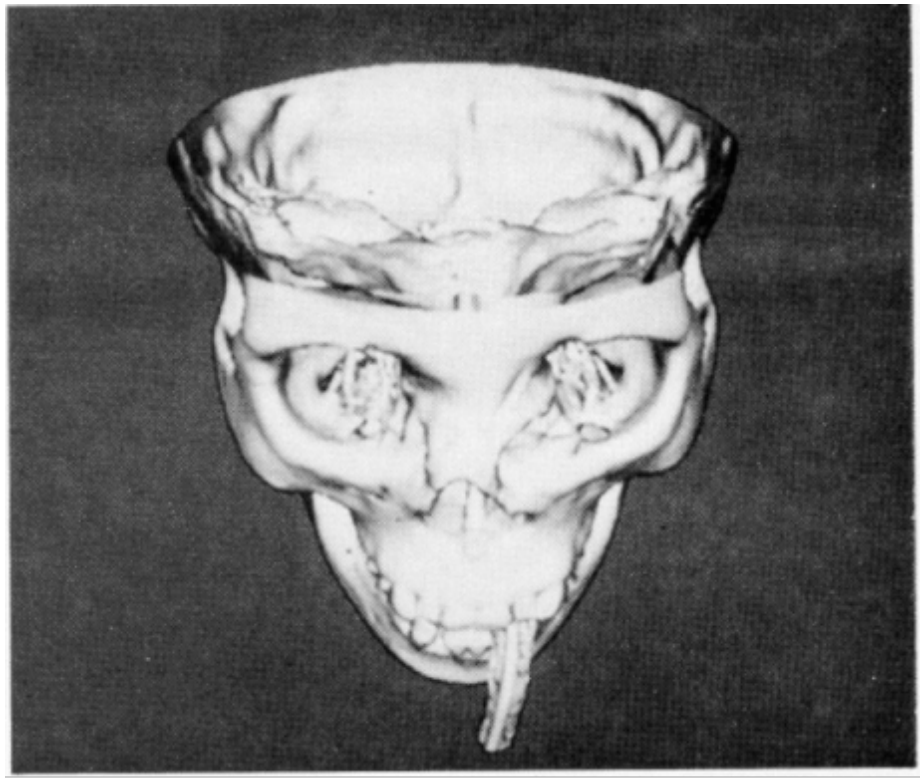


Koen Samyn, https://www.youtube.com/watch?v=LfttaAepYJ8

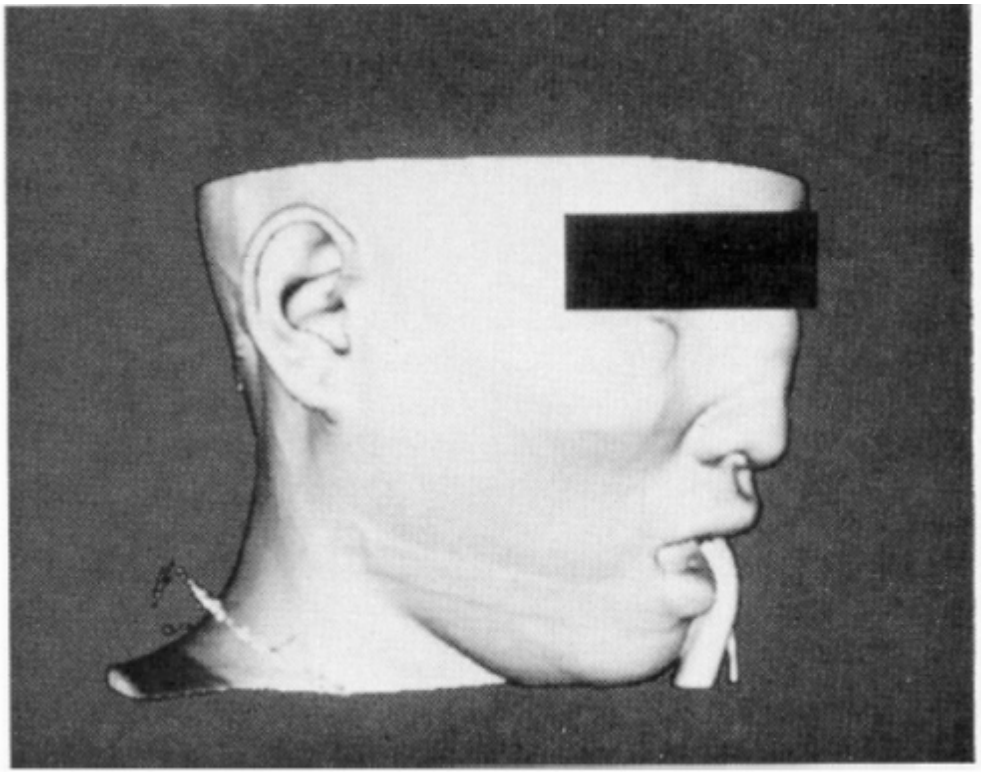
Normals at Mesh Vertices



Example: Diferent level sets of CT scan



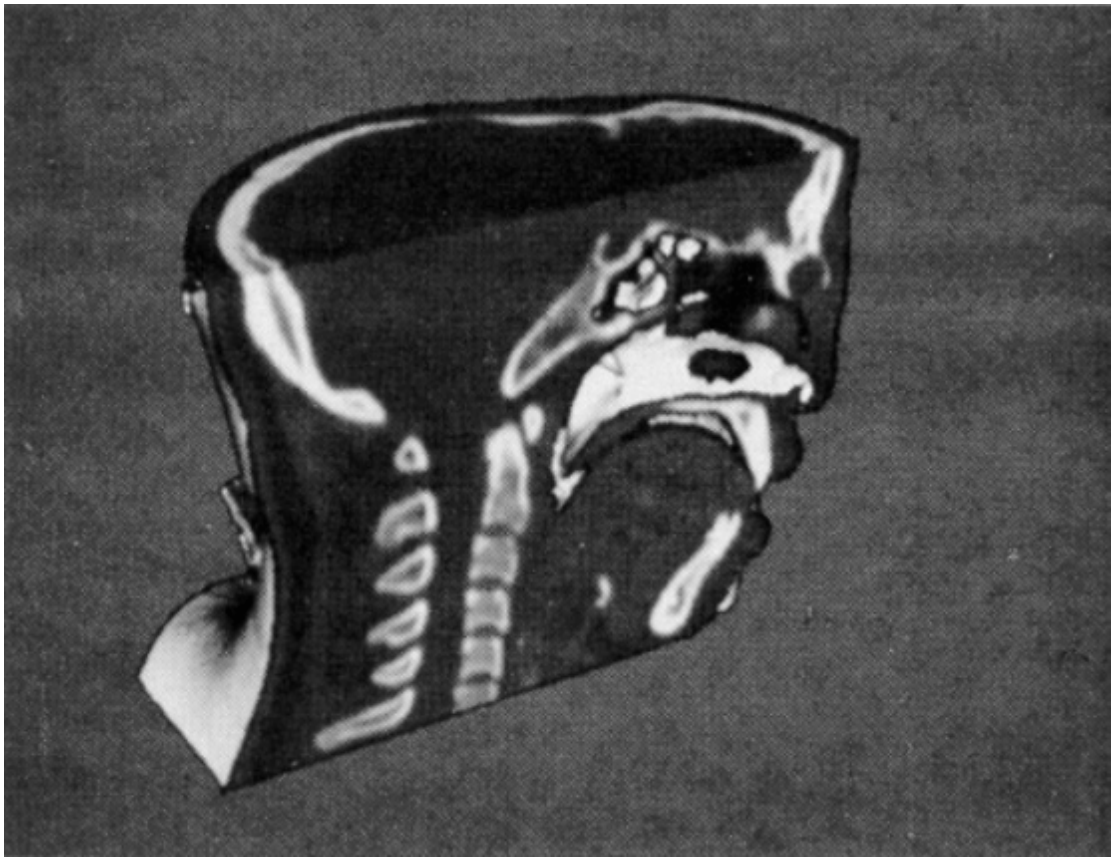
Bone surface



Soft tissue surface

Lorensen and Cline, “Marching Cubes: A High Resolution 3D Surface Reconstruction Algorithm”, SIGGRAPH '87

Example: Diferent level sets of CT scan



Alignment with original volumetric data

Lorensen and Cline, “Marching Cubes: A High Resolution 3D Surface Reconstruction Algorithm”, SIGGRAPH '87