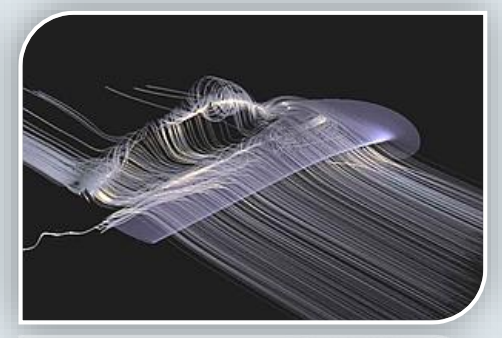
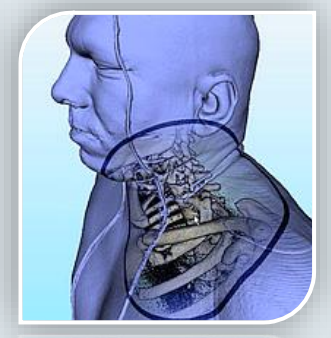
SIEMENS
Ingenuity for life


Visual Data Analytics
Isolines & Isosurfaces

Dr. Johannes Kehrer

Disclaimer

SIEMENS
Ingenuity for life

These lecture slides and materials are for personal use by students and are intended for educational purposes only. Distribution or reproduction of the material is prohibited.

The views and opinions expressed by any individual during the course of these lecturers are their own and do not necessarily represent the views, opinions, or positions of Siemens or the Technical University of Munich.

Although reasonable efforts have been made to ensure the information in these lecture slides is current and accurate, we do not assume any responsibility for the content of the posted material.

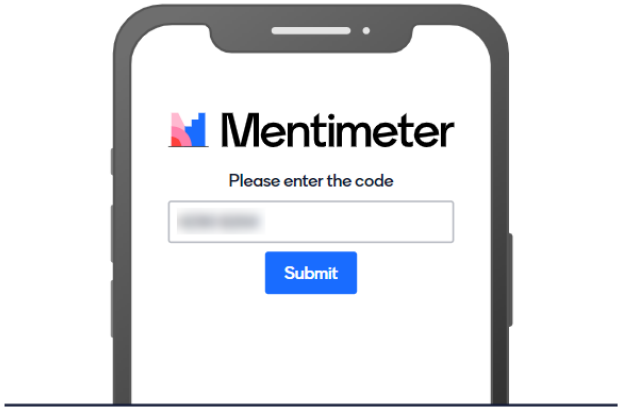
These slides also contain links to websites of third parties. As the content of these websites is not under our control, we cannot assume any liability for such external content.

Any comments/questions?

SIEMENS
Ingenuity for life

Go to

www.menti.com



Enter the code

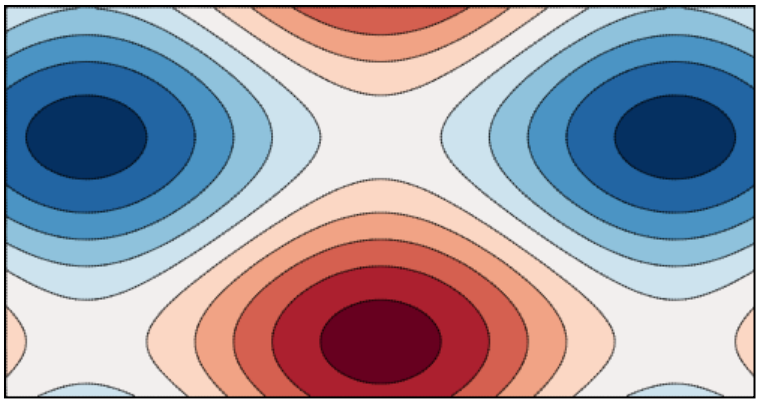
59 72 03 7

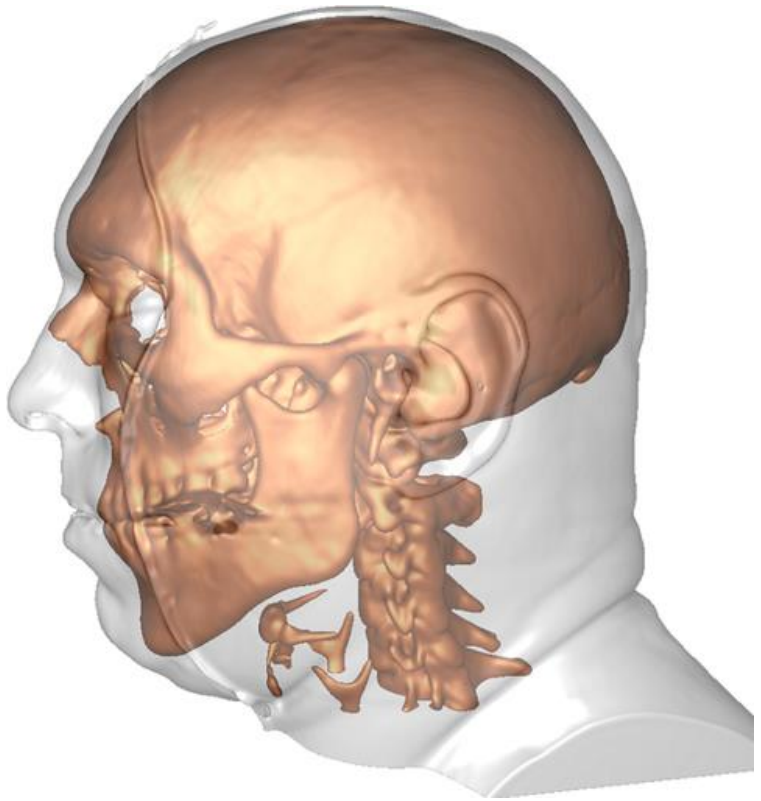
Or use QR code

Overview

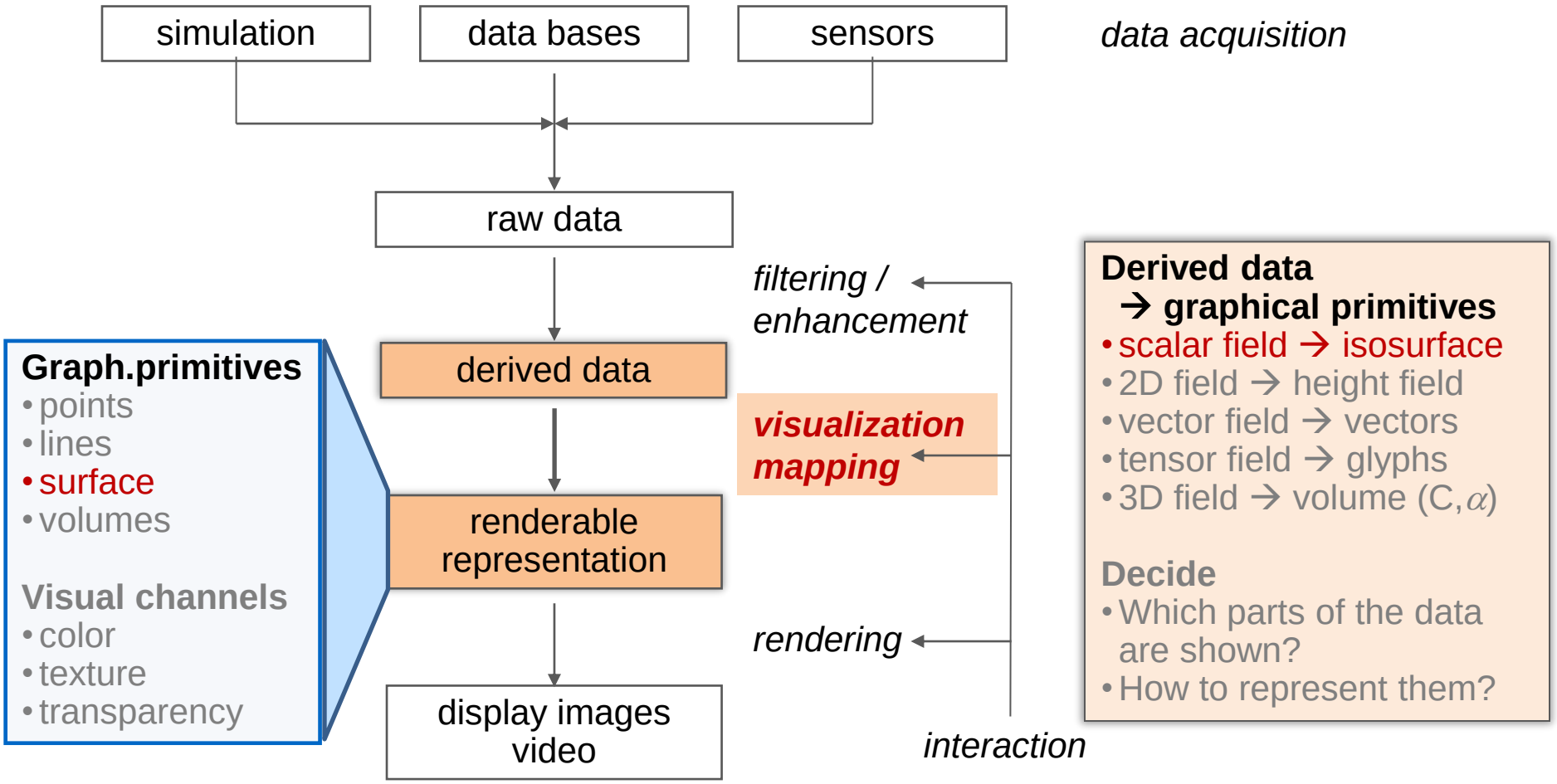
SIEMENS
Ingenuity for life

- Isolines & Isosurfaces
  - Marching squares
  - Marching cubes
- Lighting
  - Phong illumination model
- Gradient approximation





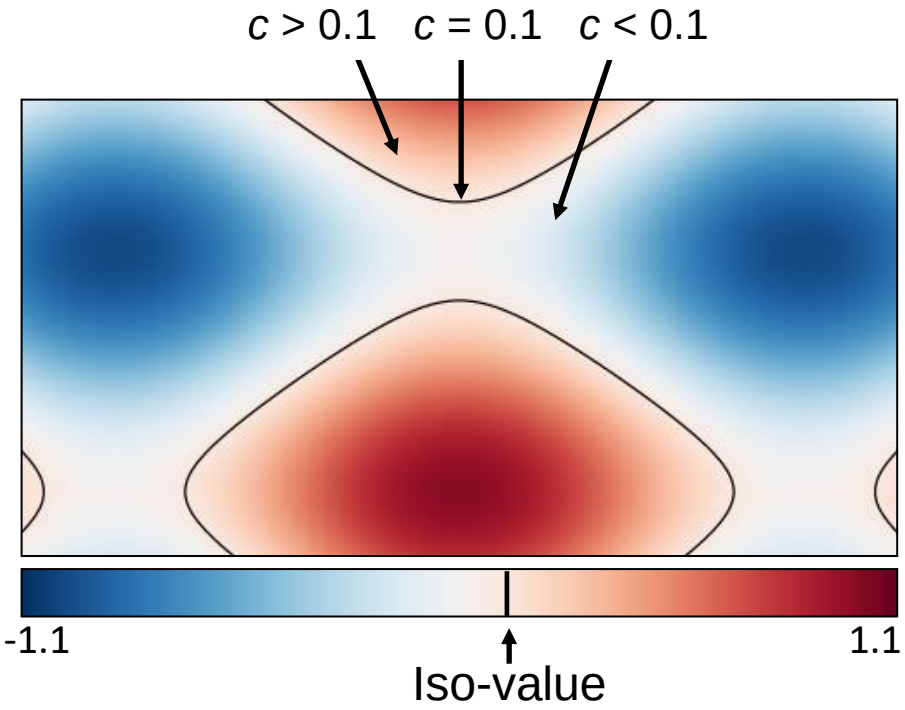
Isolines & Isosurfaces



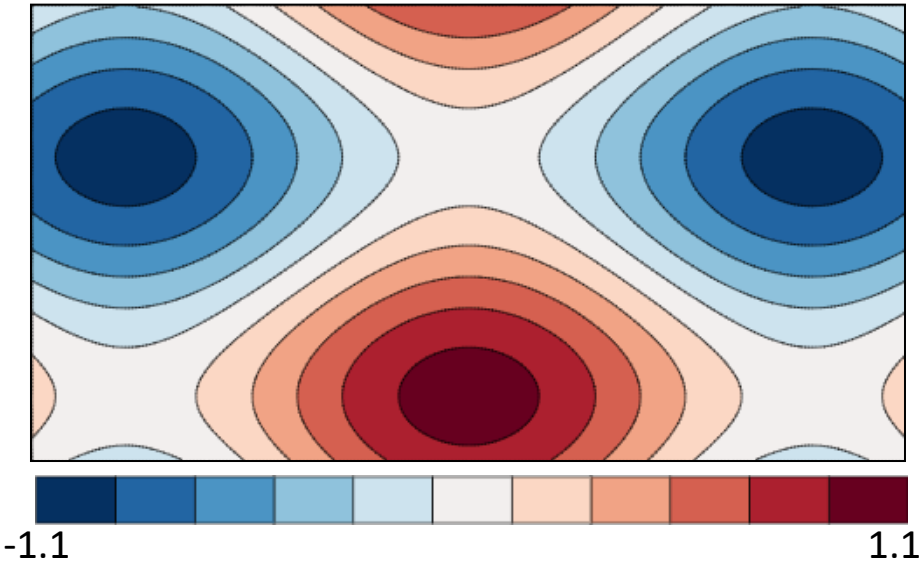
Isolines & Isosurfaces

SIEMENS
Ingenuity for life

- How to see where different values appear?



Isoline = all points having the scalar value $c = 0.1$

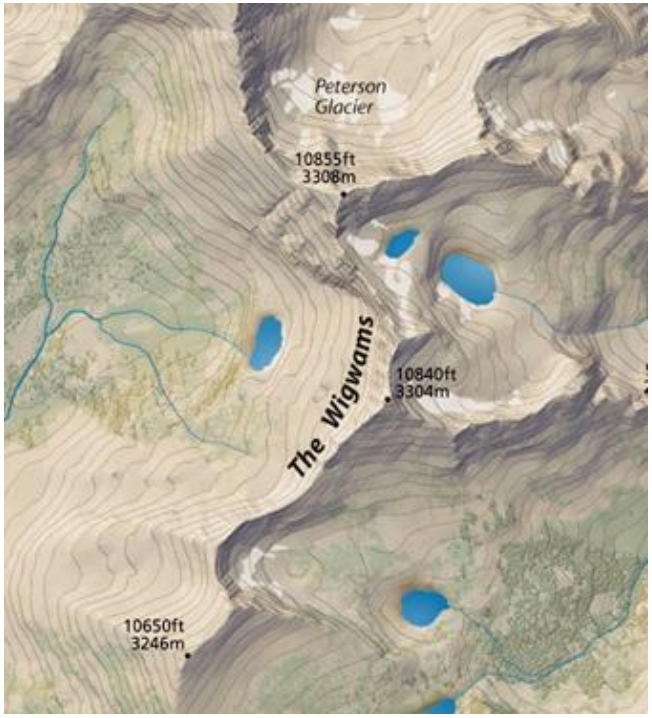


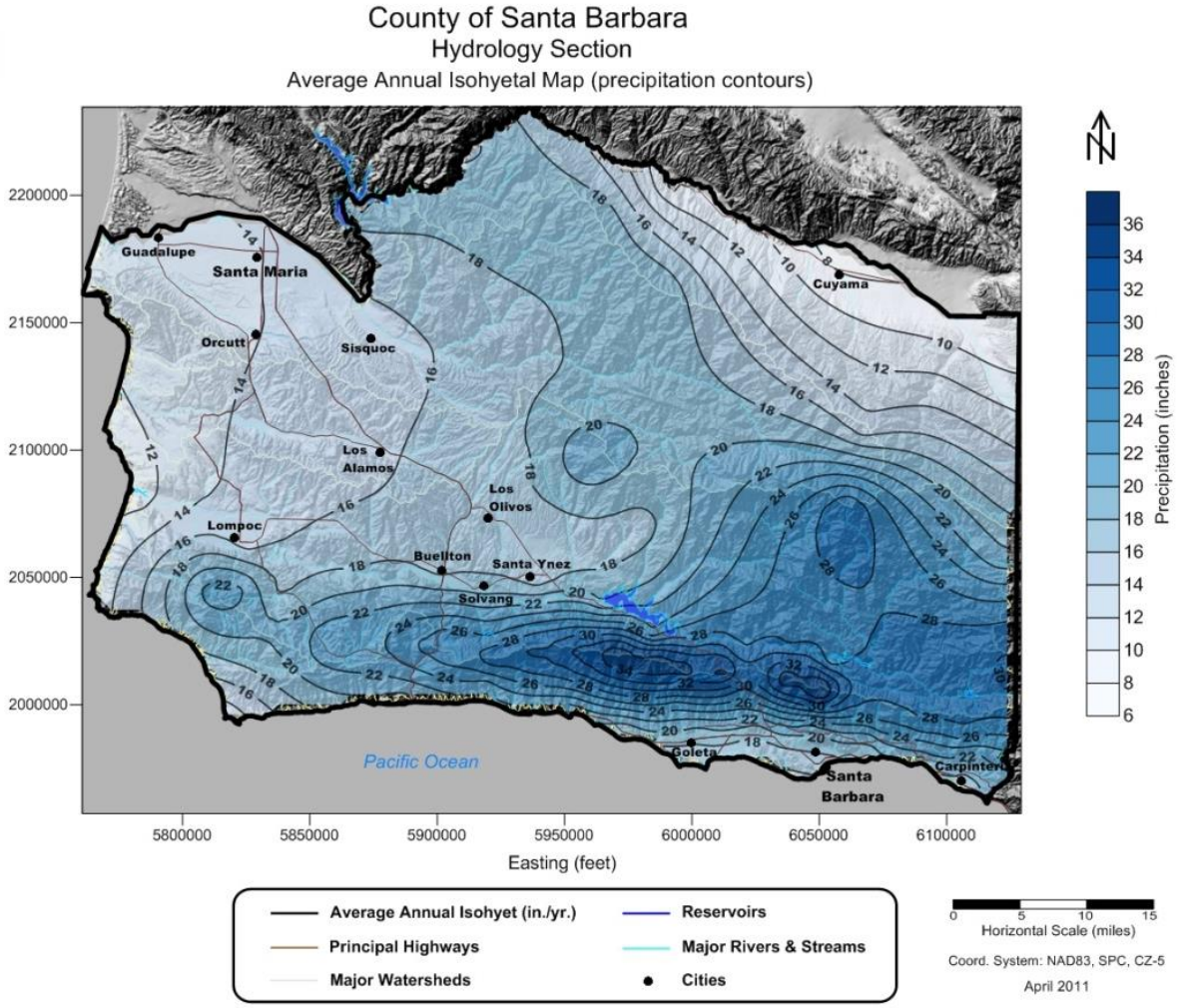
Ten different isolines, equidistant in value space

Isolines & Isosurfaces

SIEMENS
Ingenuity for life

- Known for hundreds of years in cartography

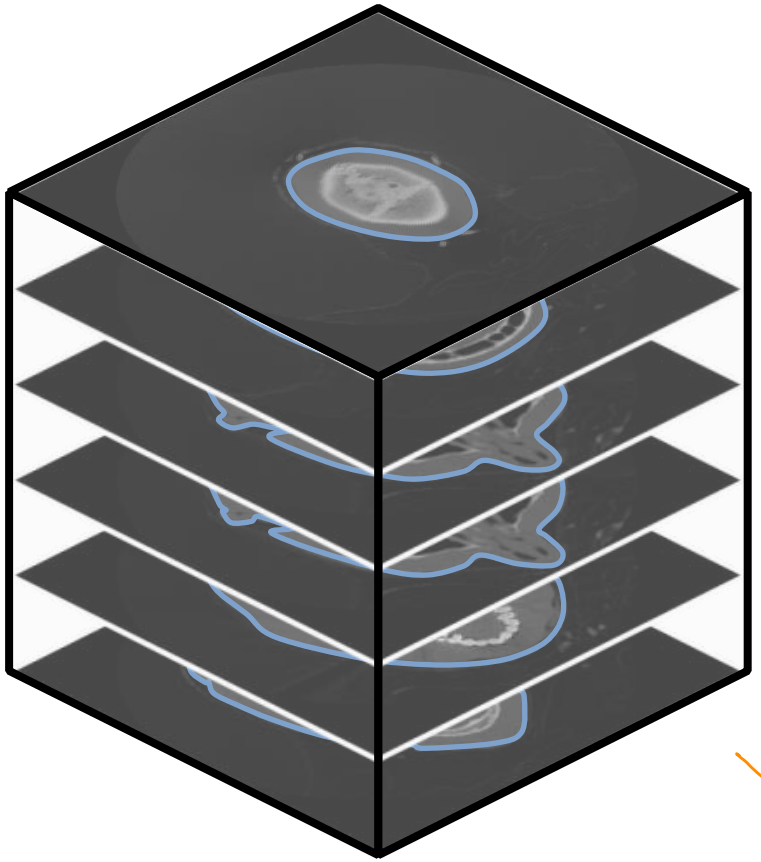


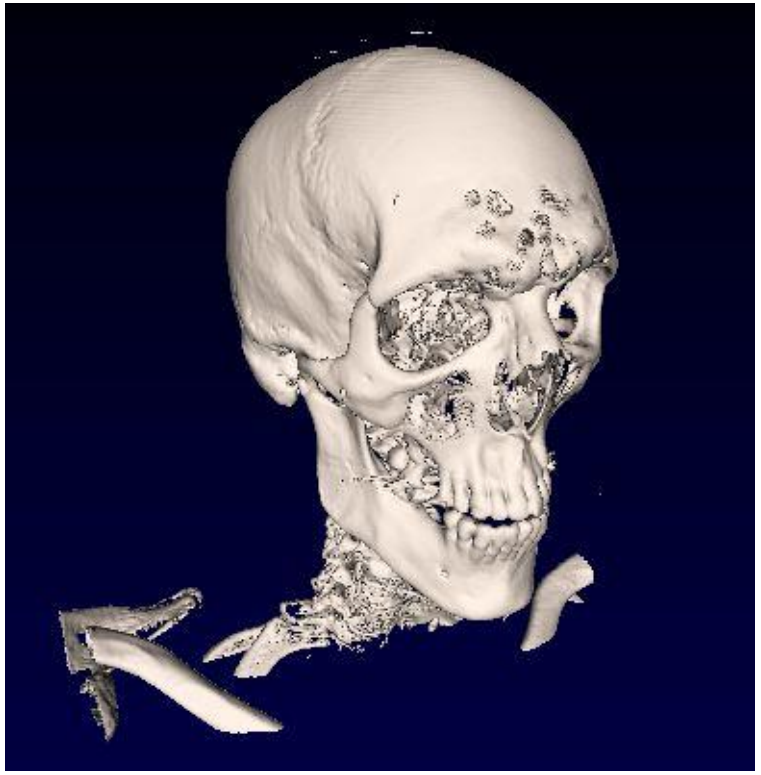


Isolines & Isosurfaces

SIEMENS
Ingenuity for life

- Isosurfaces in continuous data fields



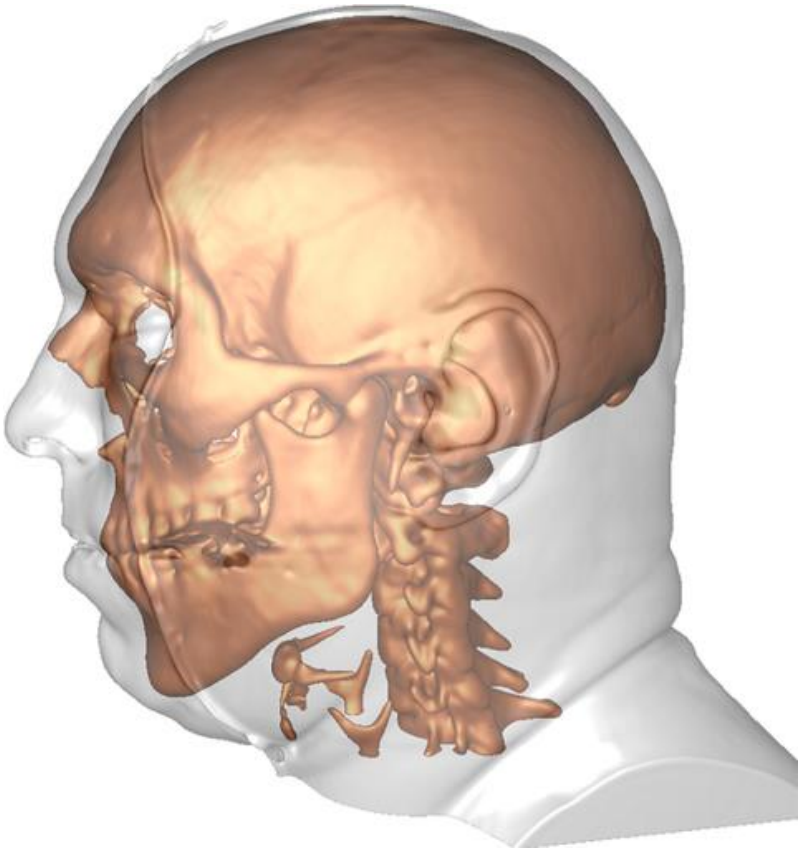


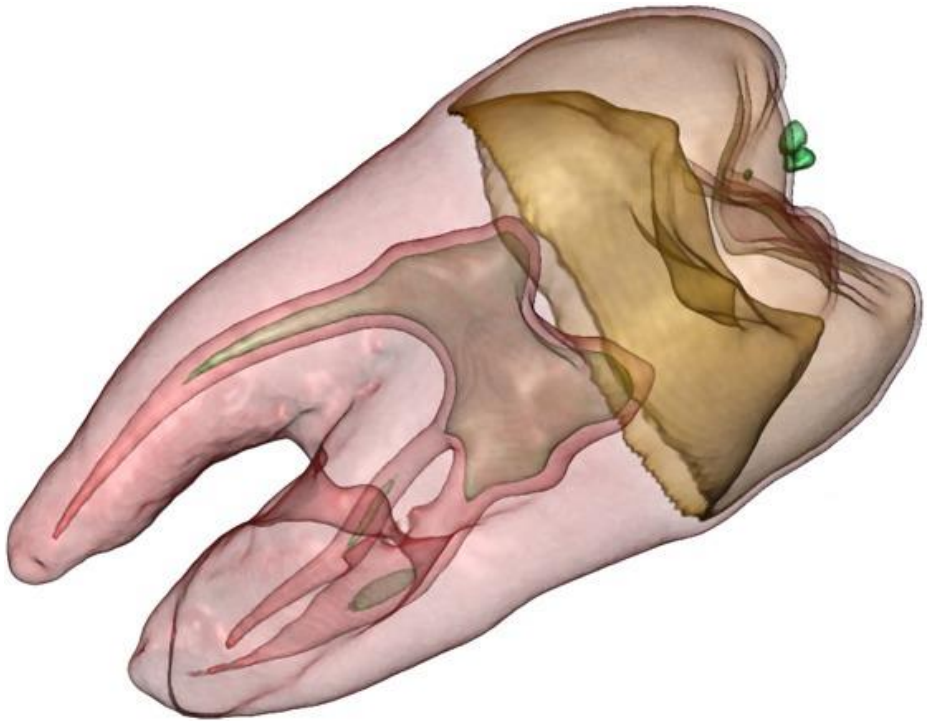
You can construct from isosurfaces to 3D object

Isolines & Isosurfaces

SIEMENS
Ingenuity for life

- Isosurfaces in continuous data fields





Semitransparent isosurfaces
for different iso-values [Kniss 02]

Isolines (2D)

SIEMENS
Ingenuity for life

- Given a 2D scalar function $f: \Omega \subset \mathbb{R}^2 \rightarrow \mathbb{R}$, and a scalar iso-value $c \in \mathbb{R}$
- Map the data to specific isolines
  - An isoline (iso-contour) consists of all points at which the data has a specific value c

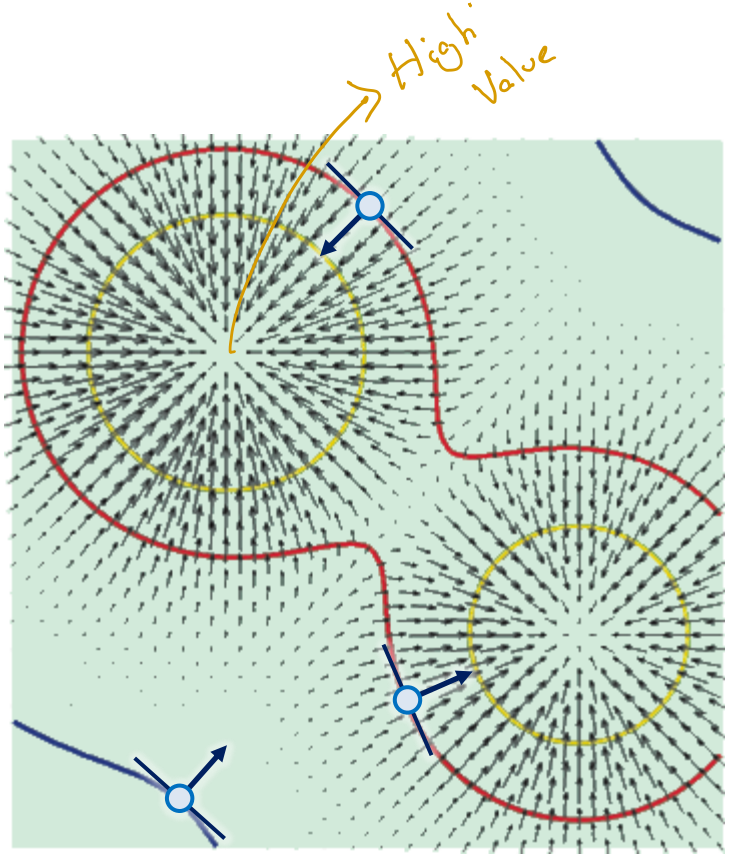
$$\{(x, y) \mid f(x, y) = c\}$$

  - Can be seen as a special kind of data condensation

Isolines

SIEMENS
Ingenuity for life

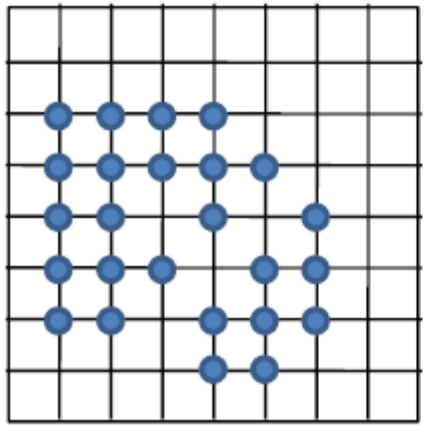
- Properties
  - Isolines are always closed curves (except when they exit the domain)
  - Isolines never (self-)intersect, thus they are nested
    - What would happen if a point belonged to 2 different isolines?
  - Isolines are always orthogonal to the scalar field's gradient

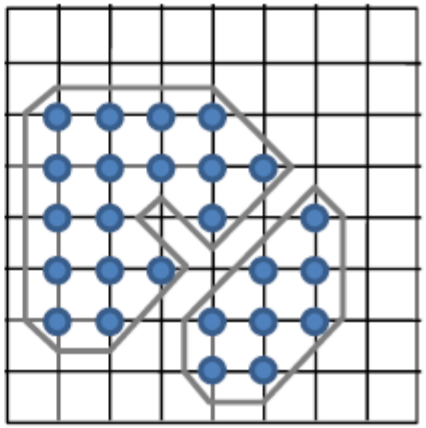


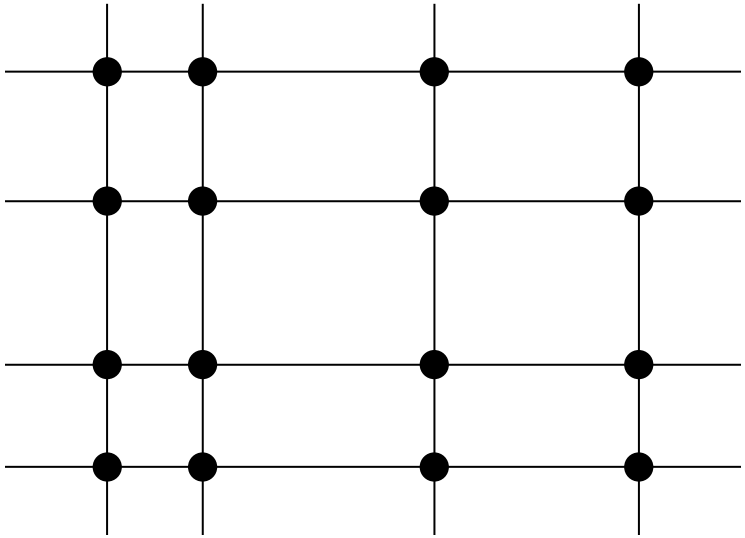
Isolines

SIEMENS
Ingenuity for life

- Computing isolines using marching squares
  - Assumes scalar values given on a rectangular grid
    - Scalar values are given at each vertex $f \leftrightarrow f_{ij}$
  - Divide and conquer: considers each cell independently
  - Takes into account the interpolation along edges
    - Isolines cannot be missed



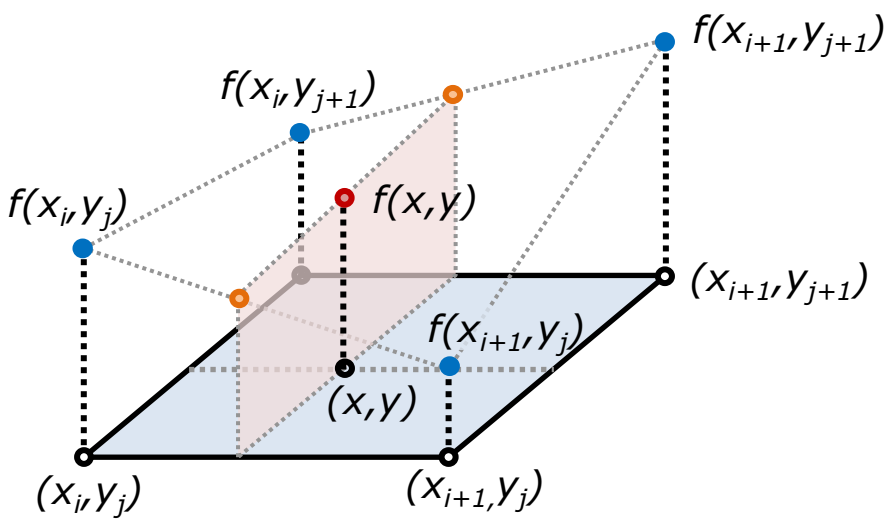


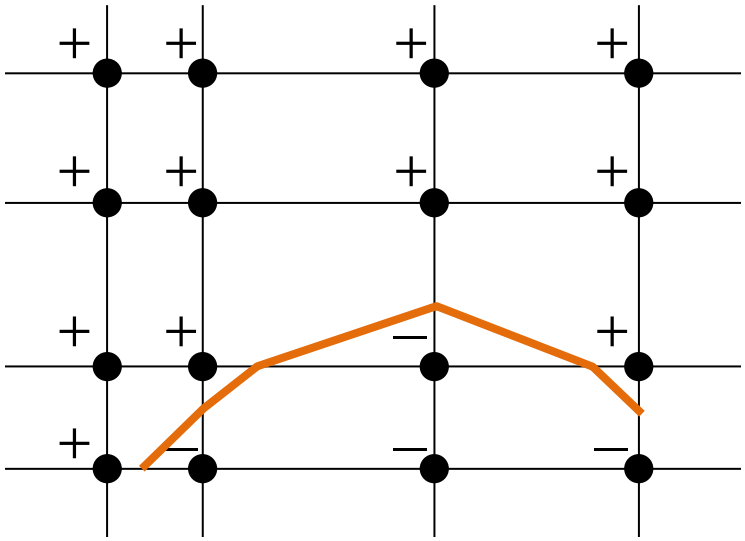


Isolines

SIEMENS
Ingenuity for life

- Which cells have an intersection with the isoline?
  - Initially mark all vertices by – or +, depending on the conditions $f_{i,j} < c$ or $c \leq f_{i,j}$
  - We assume bilinear interpolation inside a cell, thus isoline can not pass through cells with same sign at all 4 vertices
  - Only consider cells having edges with different signs

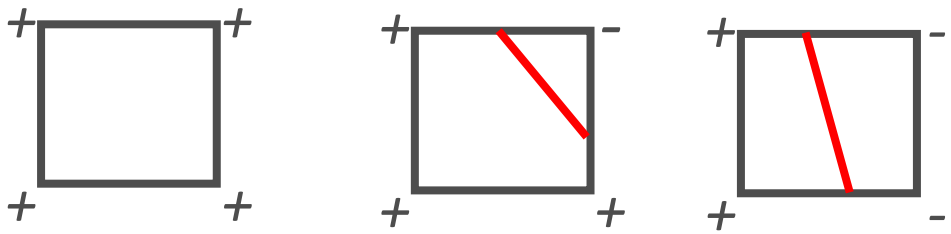


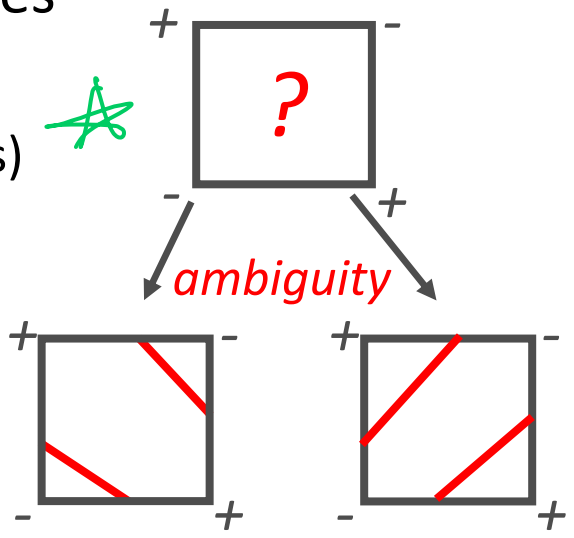


Isolines

SIEMENS
Ingenuity for life

- 16 different sign configurations in 2D
  - Only 4 different base cases
  - Symmetries: rotation, reflection, change + to – and – to +
- Compute intersections between isoline and cell edge
  - Use linear interpolation along the cell edges
  - Connect intersection points via lines
    (even though we know isocontours are hyperbolas)

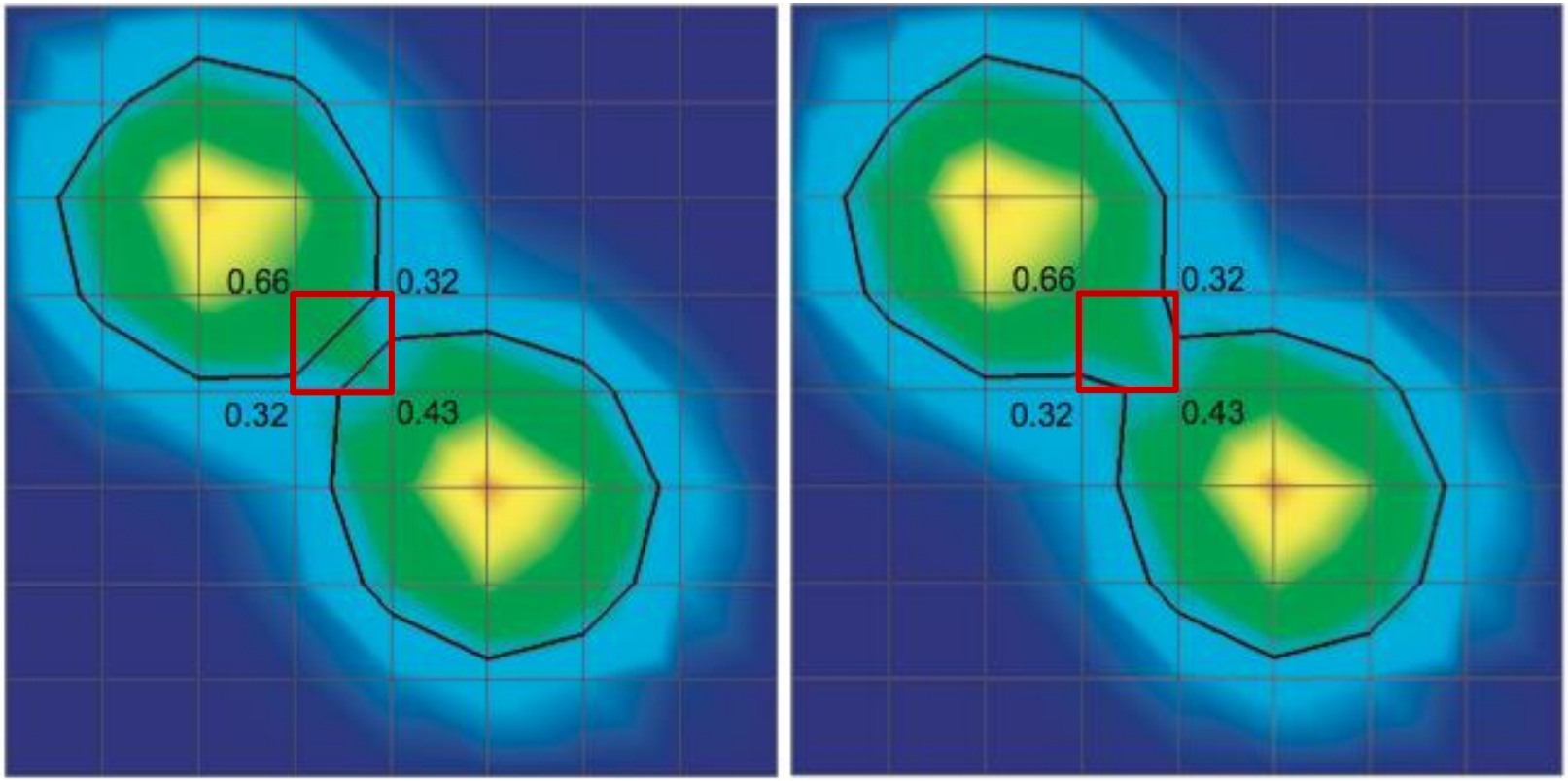




Isolines

SIEMENS
Ingenuity for life

- Ambiguities in surface topology



Which is the right isoline result?

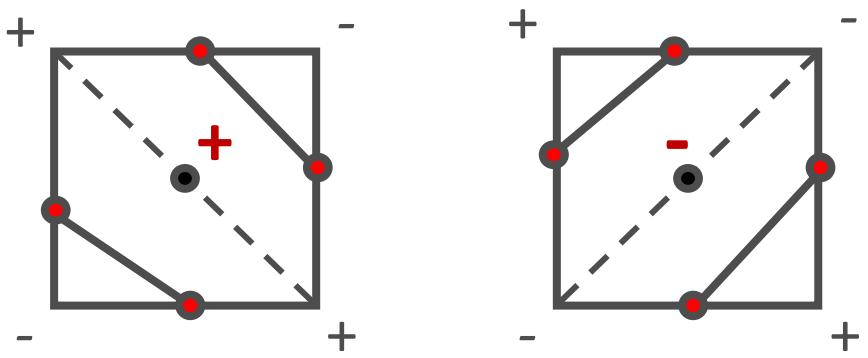
Isolines

SIEMENS
Ingenuity for life

- We can distinguish (not in all cases) the ambiguous cases by a mid point decider
  - Interpolate the function value in the center

$$f_{\text{center}} = \frac{1}{4}\left(f_{i,j} + f_{i+1,j} + f_{i,\,j+1} + f_{\text{i+1, j+1}}\right)$$

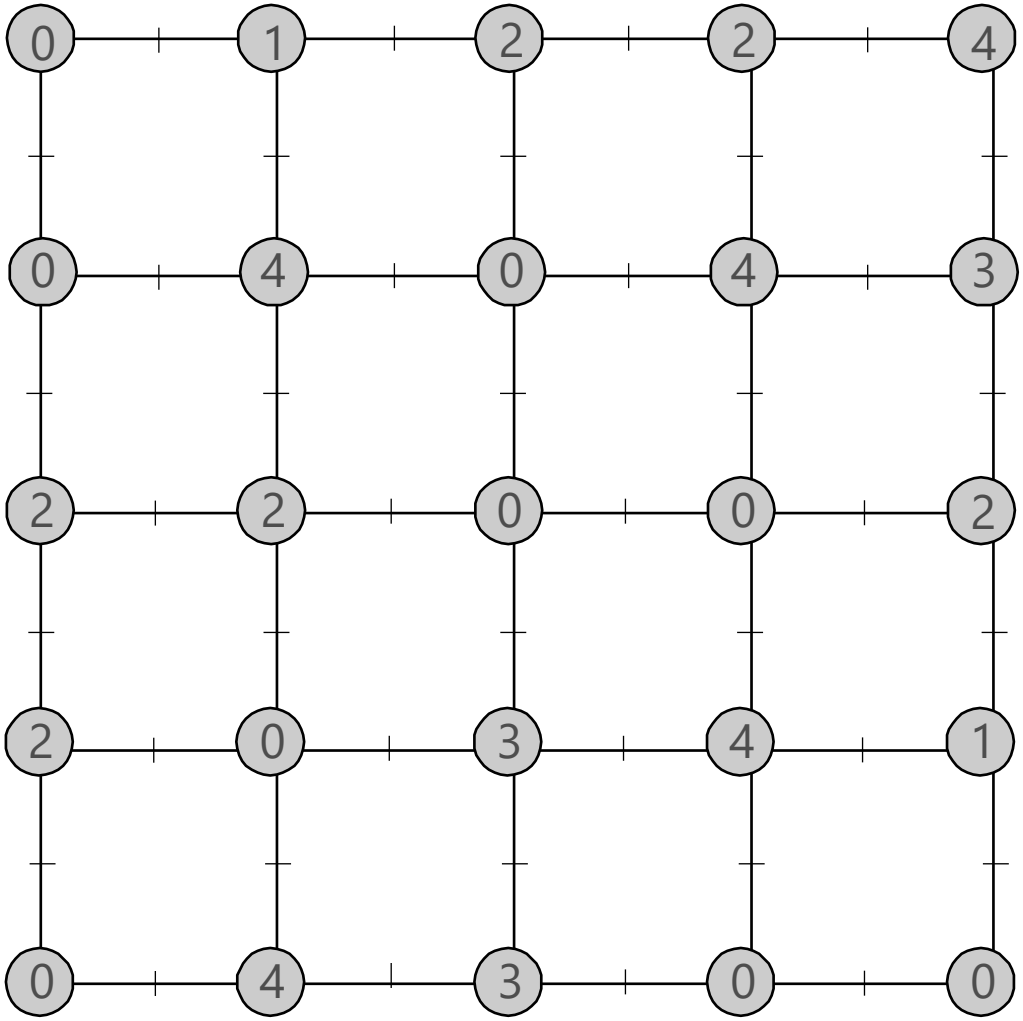
  - If $f_{\text{center}} < c$ we choose the right case, otherwise we choose the left case



Example

SIEMENS
Ingenuity for life

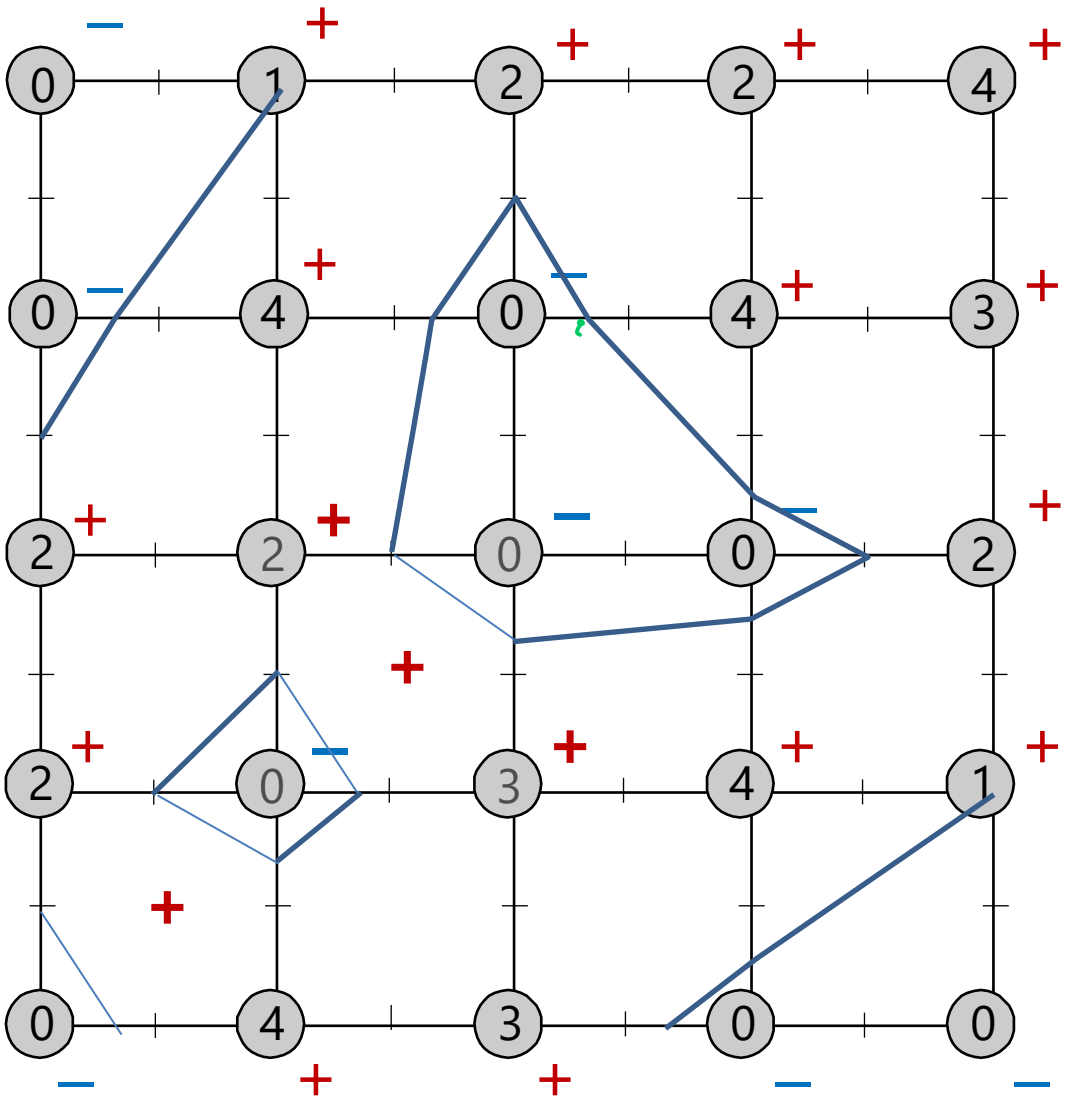
iso-value 1



Example

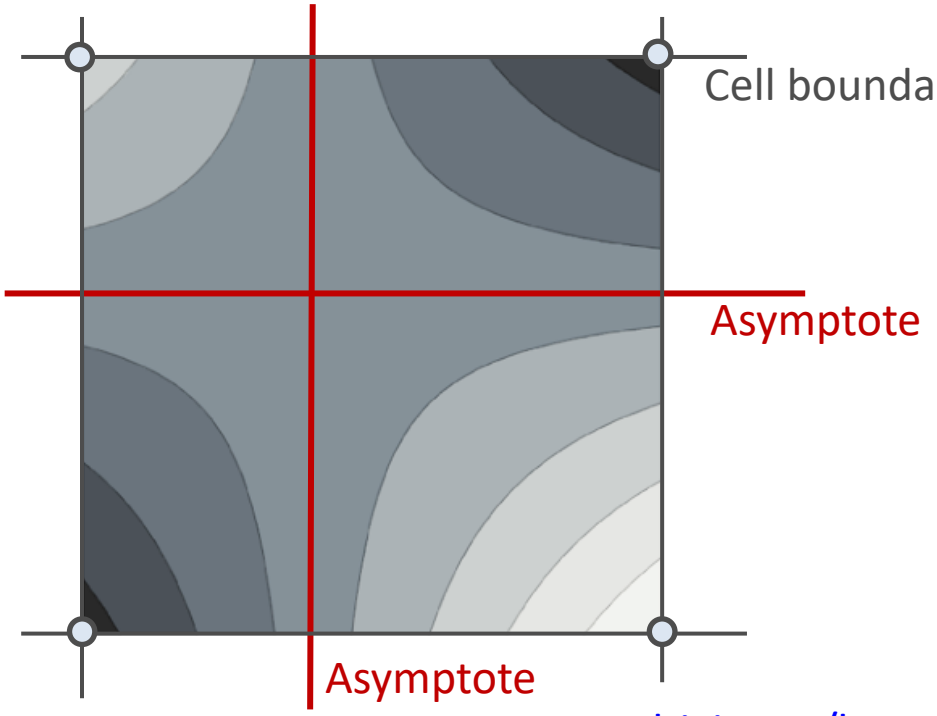
SIEMENS
Ingenuity for life

iso-value 1



Isolines

- An exact decider is the asymptotic decider
  - Considers bilinear interpolation within a cell
  - The true isolines within a cell are hyperbolas
  - Value at intersection of asymptotes decides connectivity



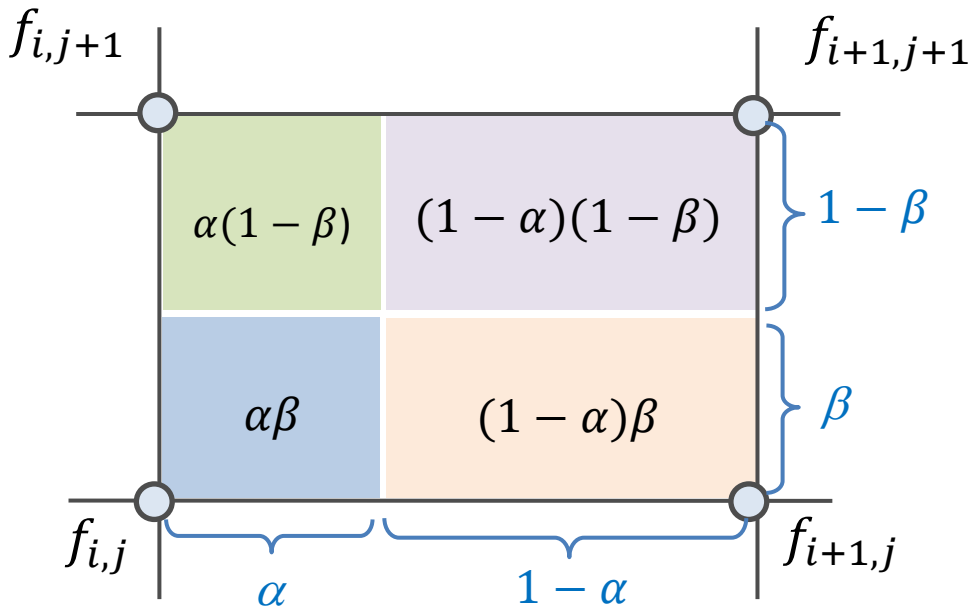
multivis.net/lecture/bilinear.html

Isolines

- How to evaluate the asymptotic decider?
  - Consider bilinear interpolation within a grid cell

$$f(\alpha,\beta) = (1-\alpha)(1-\beta)f_{i,j} + \alpha(1-\beta)f_{i+1,j} + (1-\alpha)\beta f_{i,\,j+1} + \alpha\beta f_{\mathrm{i+1,\,j+1}}$$

$$\alpha, \beta \in [0,1]$$



multivis.net/lecture/bilinear.html

Isolines

SIEMENS
Ingenuity for life

- How to evaluate the asymptotic decider?
  - Consider bilinear interpolation within a grid cell

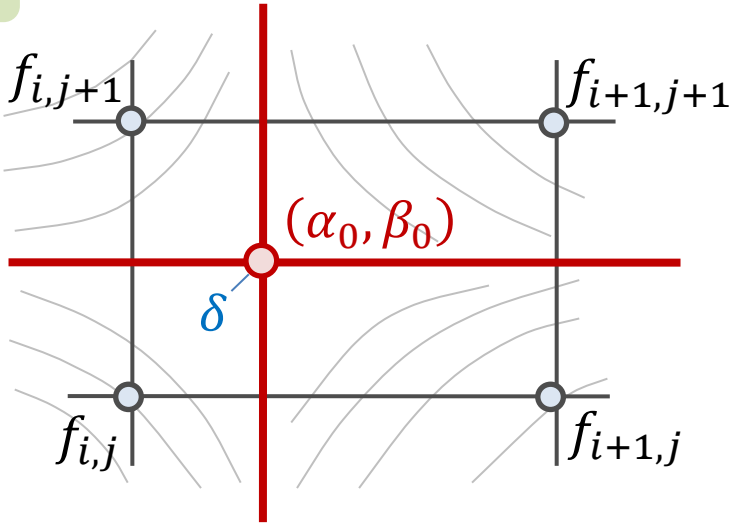
$$f(\alpha,\beta) = (1-\alpha)(1-\beta)f_{i,j} + \alpha(1-\beta)f_{i+1,j} + (1-\alpha)\beta f_{i,\,j+1} + \alpha\beta f_{\mathrm{i+1,\,j+1}} \qquad \alpha,\beta \in [0,1]$$

  - Given the values at cell corners, transform f to

$$f(\alpha,\beta) = \gamma(\alpha-\alpha_0)(\beta-\beta_0) + \delta$$

Function of a hyperbola

  - δ is the function value at the intersection point (α_0,β_0) of the asymptotes

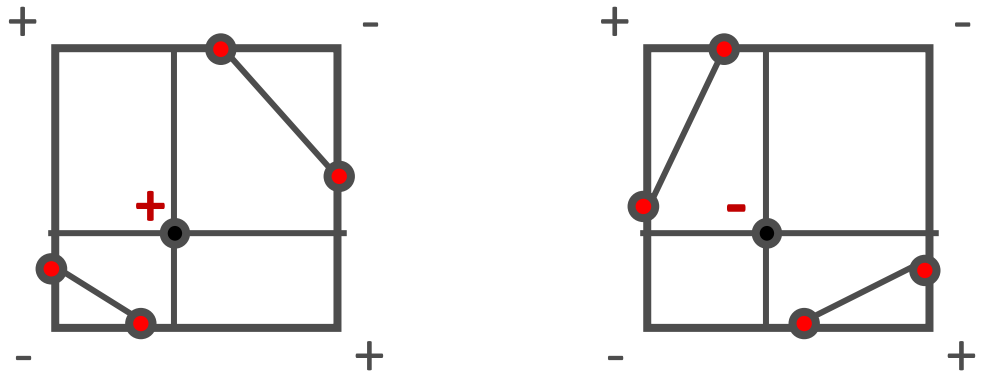


multivis.net/lecture/bilinear.html

Isolines

SIEMENS
Ingenuity for life

- How to evaluate the asymptotic decider?
  - If $\delta < c$ we chose the right case, otherwise we chose the left one



Isolines

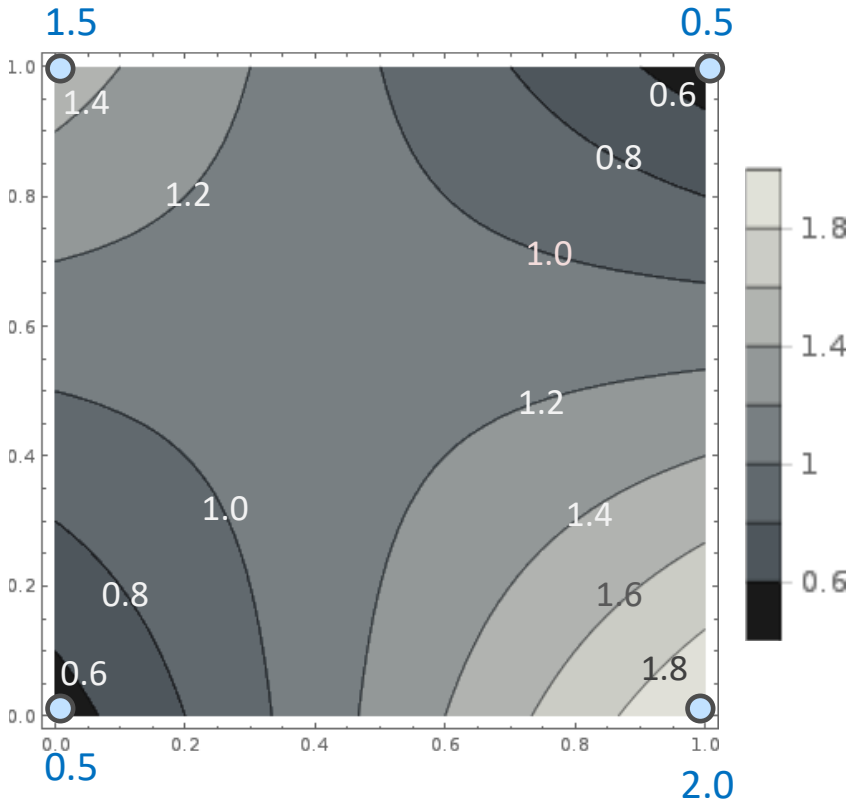
- How to evaluate the asymptotes?

  Compute asymptotes from

  $$f(\alpha,\beta) = 0.5 + 1.5\alpha + \beta - 2.5\alpha\beta$$

  Get into form of hyperbola

  $$f(\alpha,\beta) = \gamma(\alpha-\alpha_0)(\beta-\beta_0) + \delta$$



  - Where do the asymptotes intersect?
  - What is the value at the intersection?

multivis.net/lecture/bilinear.html

Isolines

- How to evaluate the asymptotes?

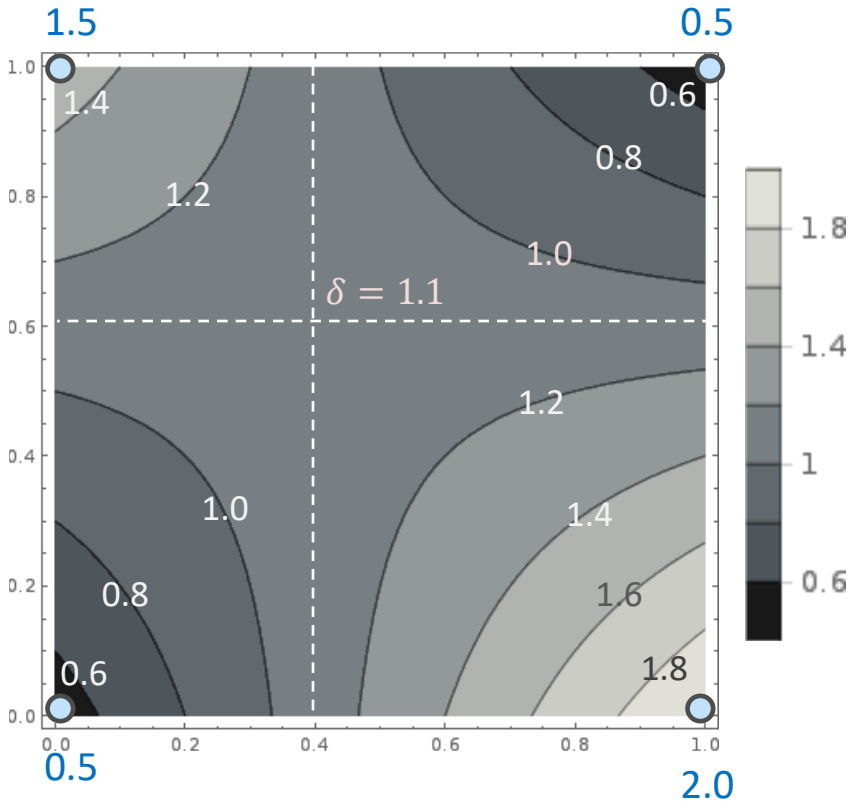
  Compute asymptotes from

  $$f(\alpha,\beta) = 0.5 + 1.5\alpha + \beta - 2.5\alpha\beta$$

  Get into form of hyperbola

  $$f(\alpha,\beta) = \gamma(\alpha-\alpha_0)(\beta-\beta_0)+\delta$$
  $$= \gamma(\alpha\beta - \beta_0\alpha - \alpha_0\beta + \alpha_0\beta_0) + \delta$$

  $$\rightarrow f(\alpha,\beta) =$$
  $$-2.5\left(\alpha - \frac{1}{2.5}\right)\left(\beta - \frac{1.5}{2.5}\right) + 0.5 + 0.6$$
  $$= -2.5(\alpha - 0.4)(\beta - 0.6) + 1.1$$



  – Asymptotes intersect at $(0.4, 0.6)$

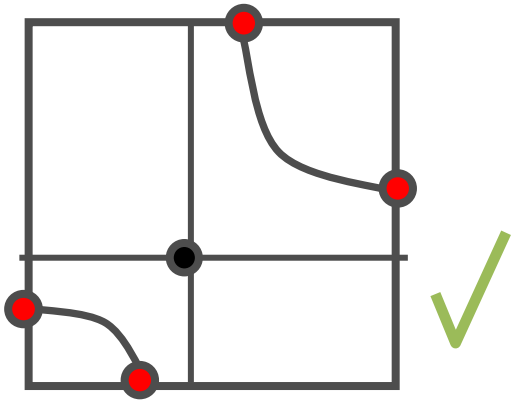
  – Value at intersection: $\delta = 1.1$

multivis.net/lecture/bilinear.html

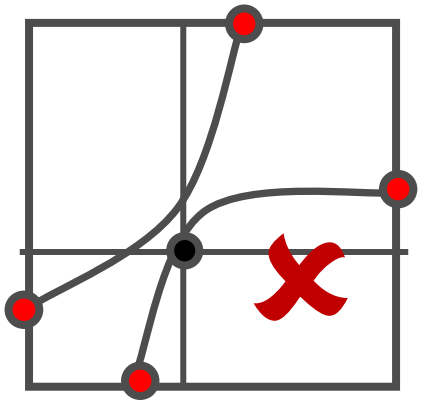
Isolines

SIEMENS
Ingenuity for life

- How to evaluate the asymptotic decider?
  - We can avoid the evaluation of δ by investigating the order of intersection points along either axis
  - Build pairs of first two and last two intersections and connect these pairs



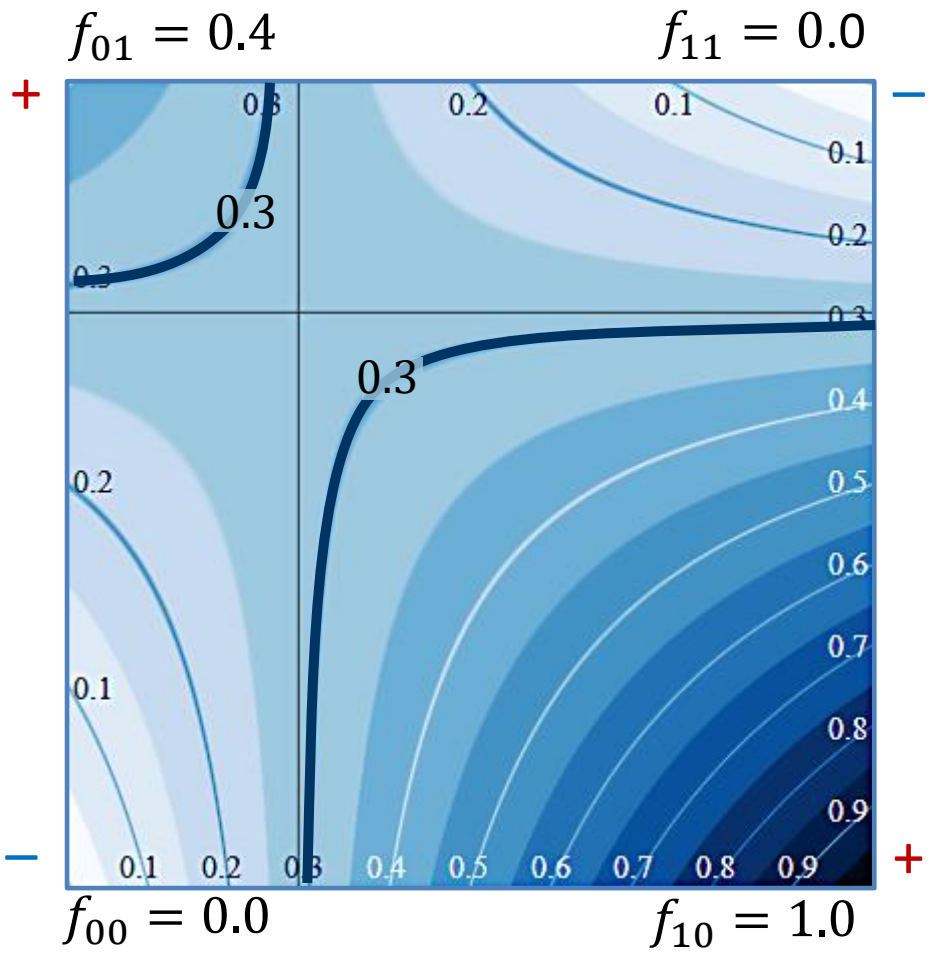
valid



not possible

Isolines

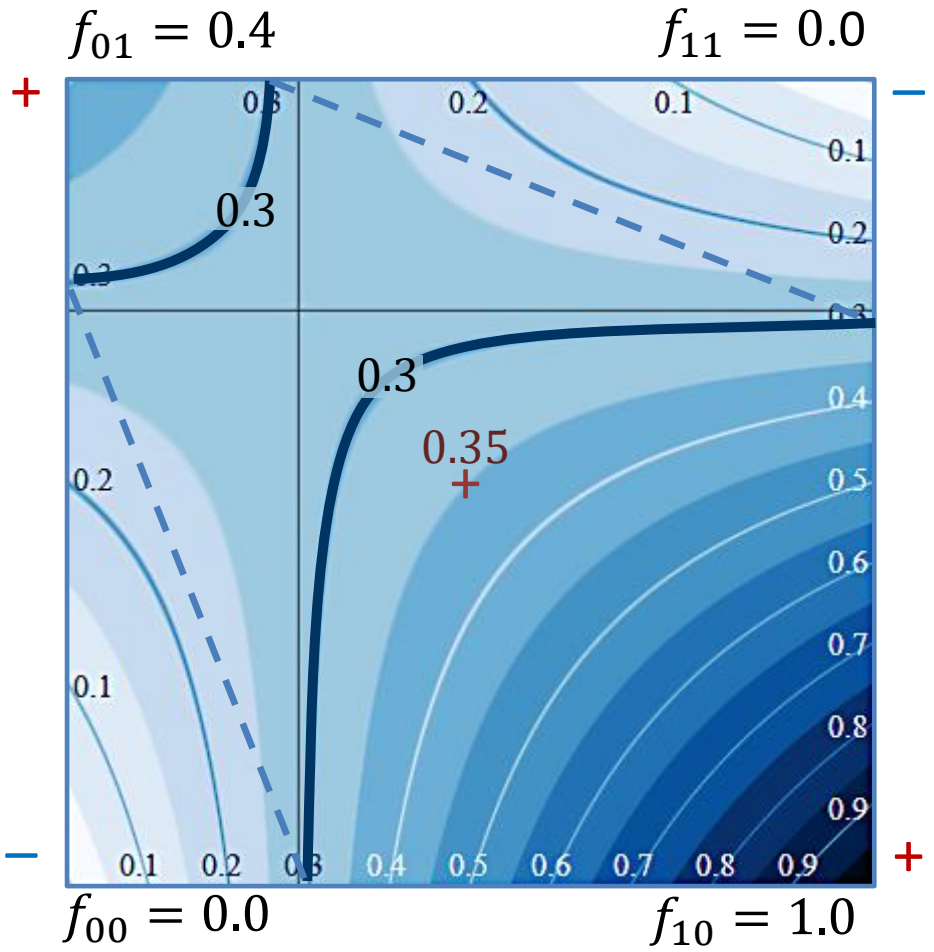
- When does the midpoint decider **not** work
  - **Example:** Iso-line for $c = 0.3$



multivis.net/lecture/bilinear.html

Isolines

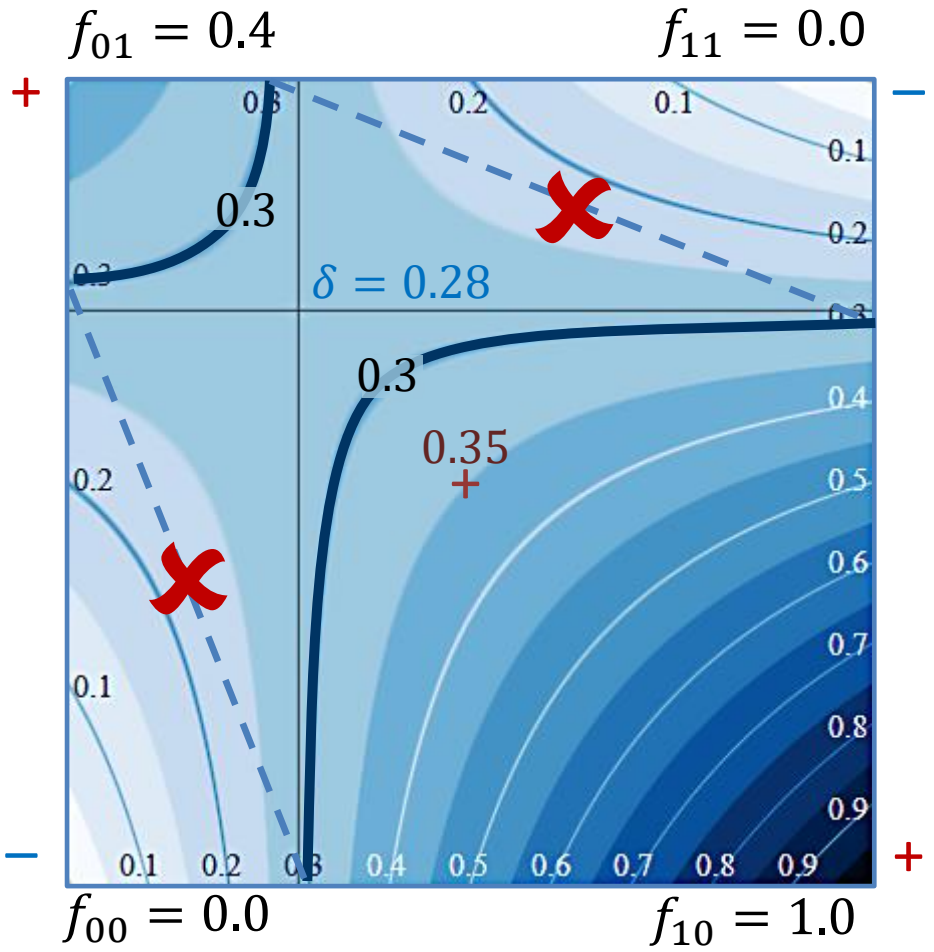
- When does the midpoint decider **not** work
  - **Example:** Iso-line for $c = 0.3$
  - Value at midpoint: 0.35



multivis.net/lecture/bilinear.html

Isolines

- When does the midpoint decider **not** work
  - **Example:** Iso-line for $c = 0.3$
  - Value at midpoint: 0.35
  - Evaluating the asymptotic decider, we get $\delta = 0.28$ at intersection of asymptotes
  - The midpoint decider would give us wrong isolines



multivis.net/lecture/bilinear.html

Isolines

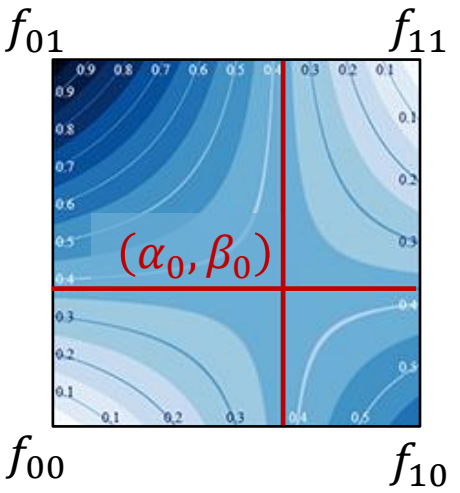
SIEMENS
Ingenuity for life

- Summary: Bilinear Interpolation
  - The value at each point (α, β) within the cell can be obtained by

$$f(\alpha,\beta) = (1-\alpha)(1-\beta)f_{00} + \alpha(1-\beta)f_{10} + (1-\alpha)\beta f_{01} + \alpha\beta f_{11}$$

$$= A\alpha + B\beta + C\alpha\beta + D$$

where $A = f_{10} - f_{00}$, $B = f_{01} - f_{00}$, $C = f_{00} - f_{01} - f_{10} + f_{11}$, $D = f_{00}$



  - We can evaluate the isoline for an iso-value c by setting $f(\alpha,\beta) = c$
  - The asymptotes of the hyperbolas can be computed by

$$f(\alpha,\beta) = \gamma(\alpha-\alpha_0)(\beta-\beta_0) + \delta$$

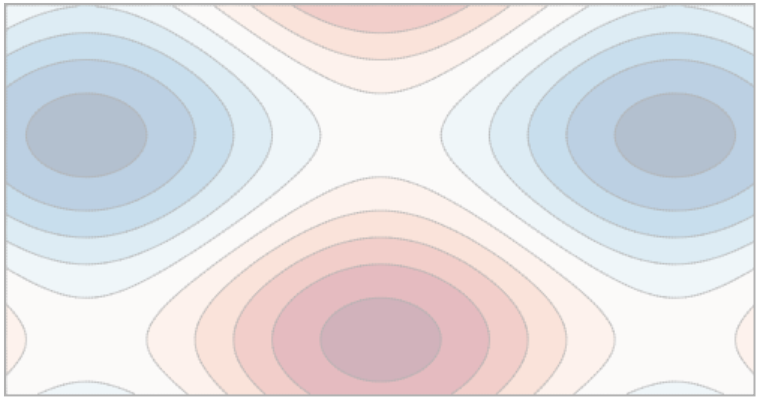
where $\gamma = C$ and $\delta = (f_{00}\,f_{11} - f_{01}\,f_{10})/C$ is the value at the intersection point (α_0, β_0) of the asymptotes with $\alpha_0 = -B/C$ and $\beta_0 = -A/C$

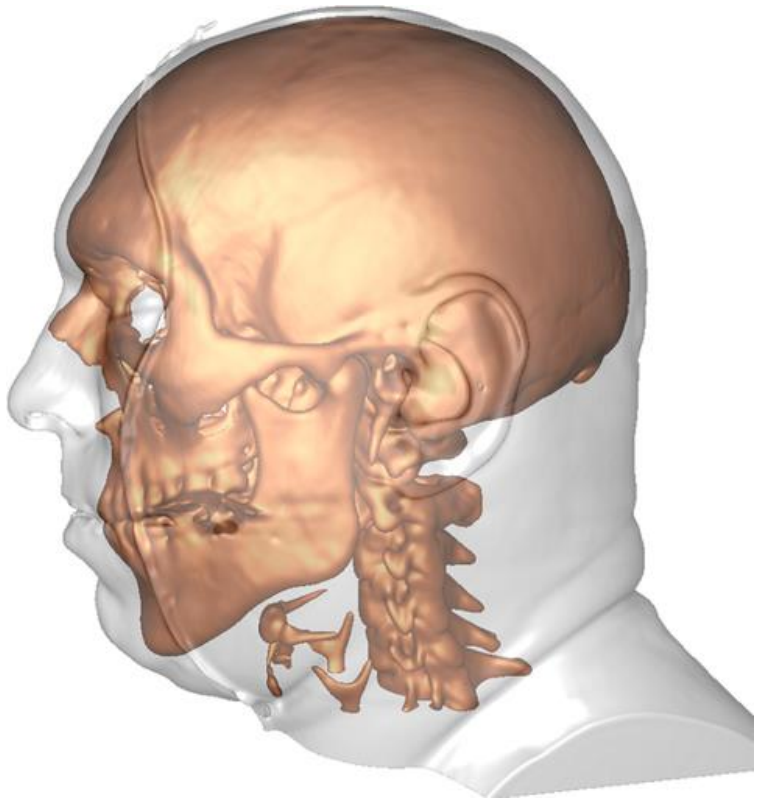
multivis.net/lecture/bilinear.html

Overview

SIEMENS
Ingenuity for life

- Isolines & Isosurfaces
  - Marching squares
  - Marching cubes
- Lighting
  - Phong illumination model
- Gradient approximation

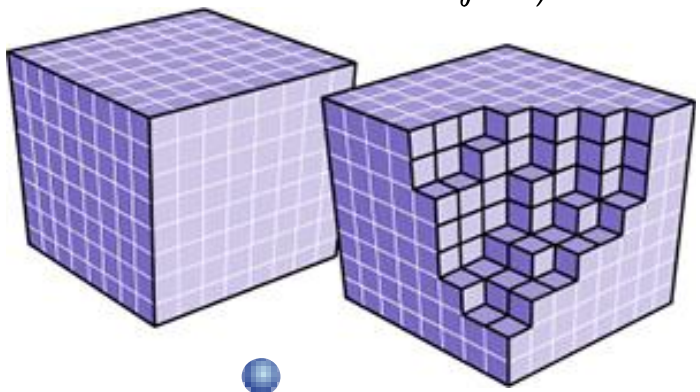


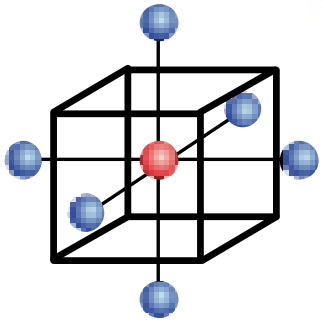


Volume visualization

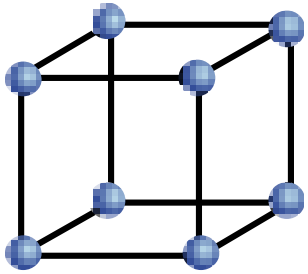
SIEMENS
Ingenuity for life

- Data values are initially given at vertices of a 3D grid
  - These are called voxels (volume elements)
  - Voxel = point sample in 3D





- "Adjacent" grid vertices make up a cell
  - Use interpolation for data reconstruction in a cell
  - Corners: 8 voxel



Surface reconstruction in 3D

SIEMENS
Ingenuity for life

- Marching-Cubes (MC) algorithm
  - Invented by Lorensen and Cline 1987
  - Approximates the surface by a triangle mesh
  - Surface vertices are found by linear interpolation along cell edges
  - Efficient triangulation by means of lookup tables

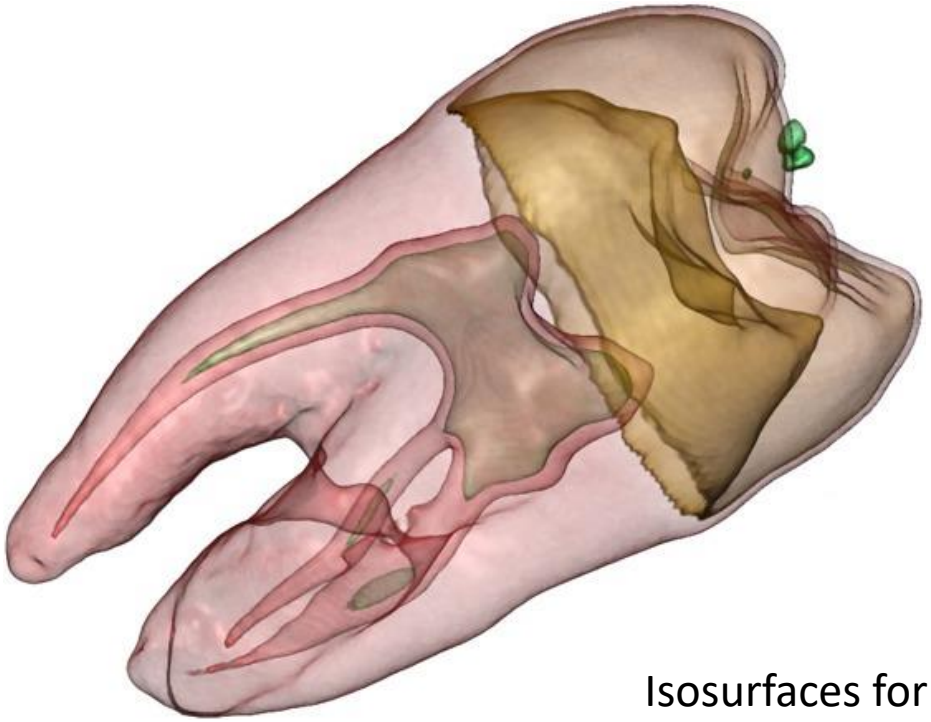
- The standard geometry-based surface extraction algorithm for 3D scalar field

Marching Cubes

SIEMENS
Ingenuity for life

- Computes isosurface for specific iso-value

- Tasks
  - Shape understanding
  - Spatial relationships



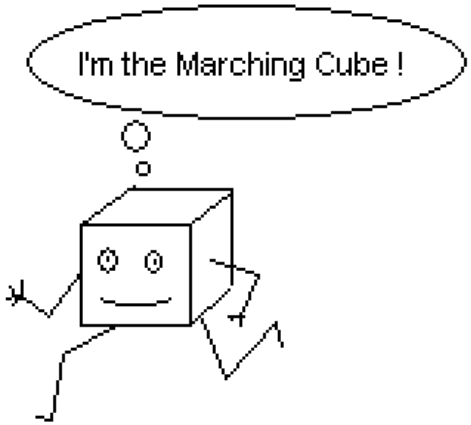
Isosurfaces for different iso-values [Kniss 02]

Marching Cubes

SIEMENS
Ingenuity for life

- The MC algorithm
  - Cell consists of 8 vertices
  - indices: (i+[0,1], j+[0,1], k+[0,1])

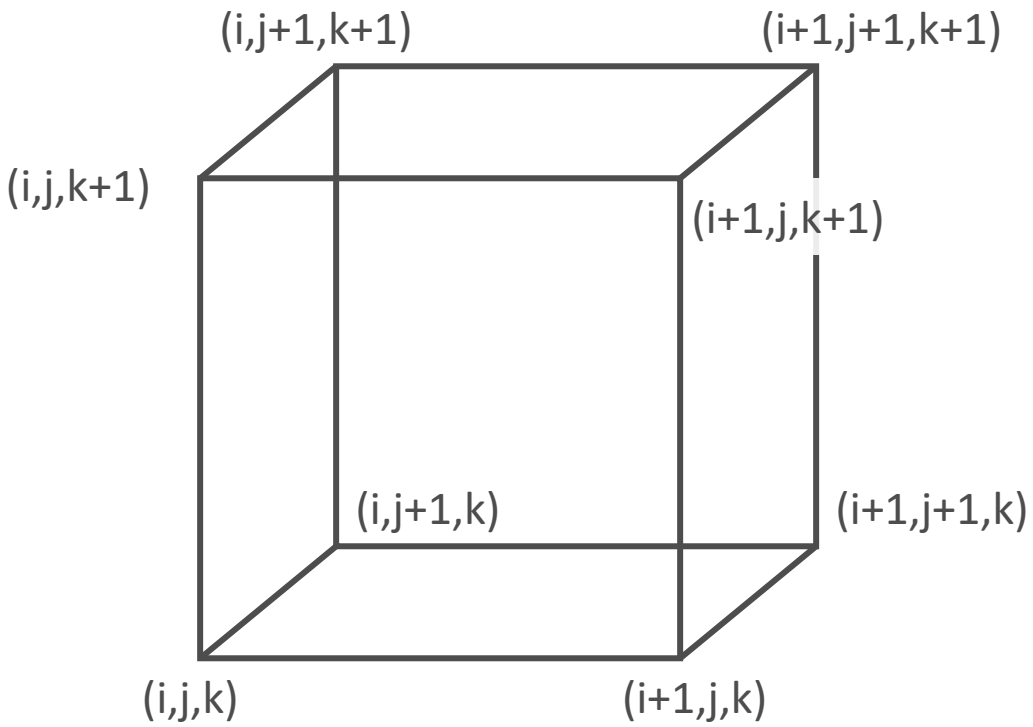
1. Consider a cell independently
2. Classify each vertex as inside or outside
3. Build an index
4. Get per-cell triangulation from index
5. Interpolate the edge location
6. Compute gradients
7. Consider ambiguous cases
8. Go to next cell



Marching Cubes

SIEMENS
Ingenuity for life

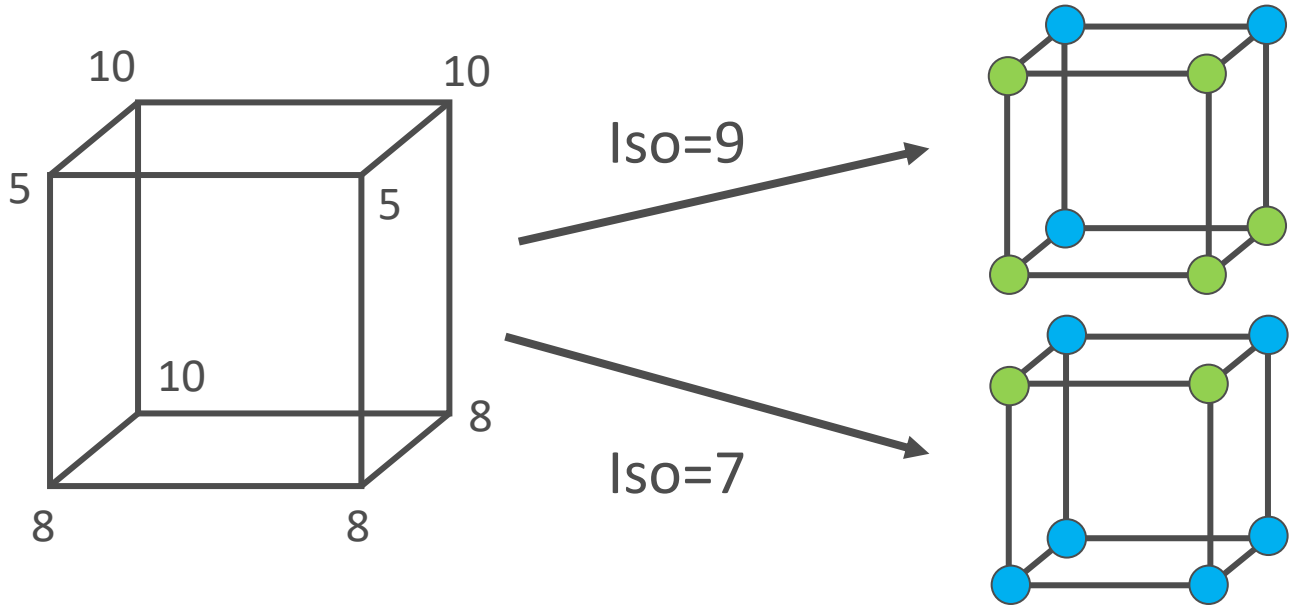
- Step 1: Consider a cell defined by eight vertices with associated data values



Marching Cubes

SIEMENS
Ingenuity for life

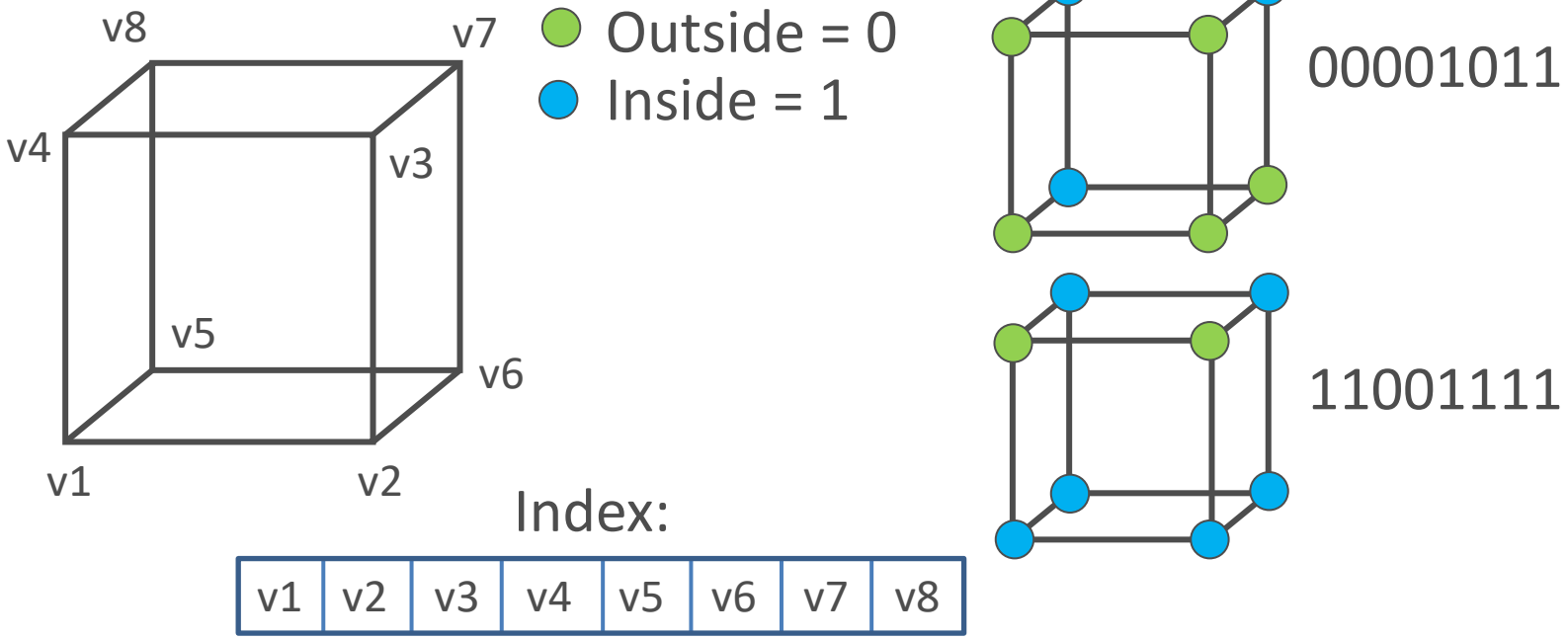
- Step 2: Classify each cell according to whether it lies
  - outside the surface (value < iso-value)
  - inside the surface (value ≥ iso-value)



Marching Cubes

SIEMENS
Ingenuity for life

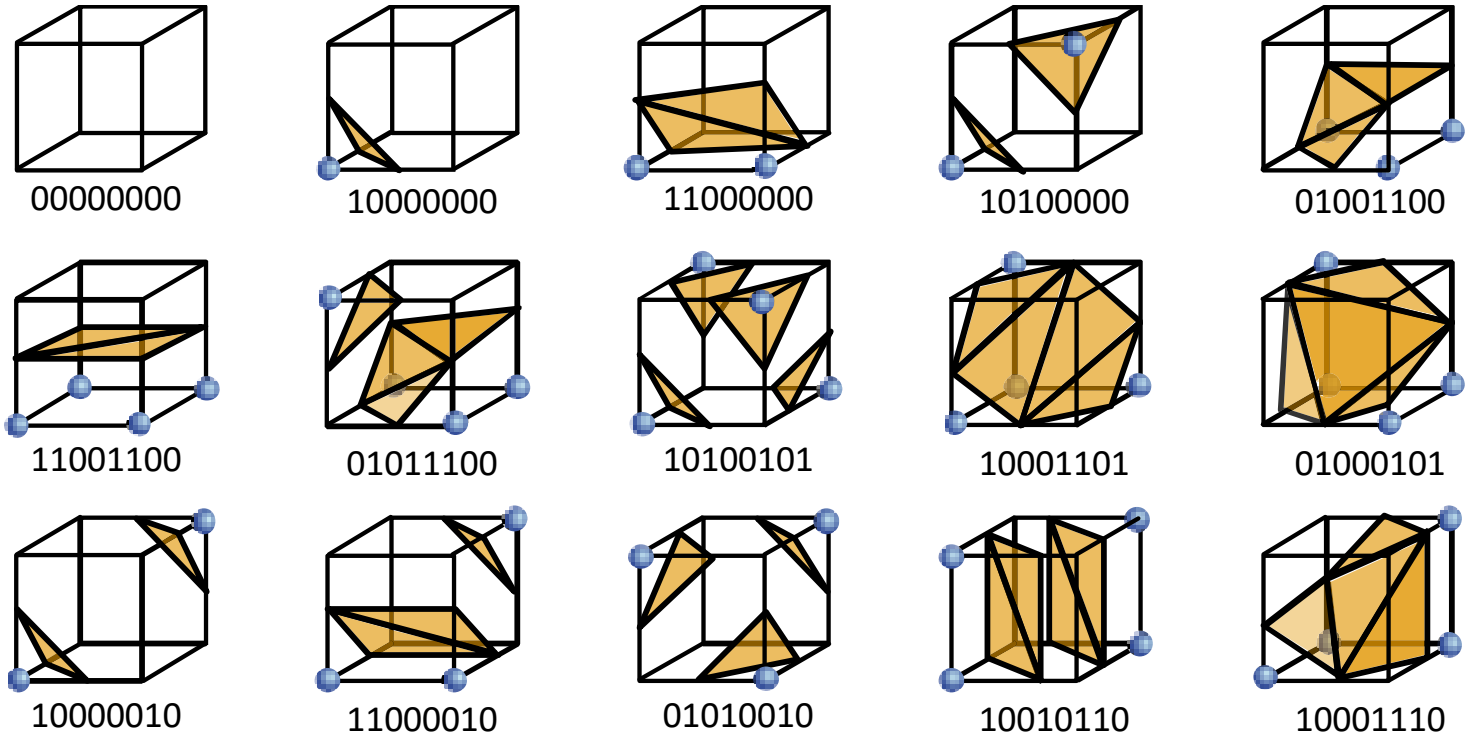
- Step 3: Use the binary labeling of each cell to compute an index



Marching Cubes

SIEMENS
Ingenuity for life

- Step 4: Per index: look up the triangulation for this index from a pre-computed table
  - All 256 cases can be derived from 15 base cases due to symmetries

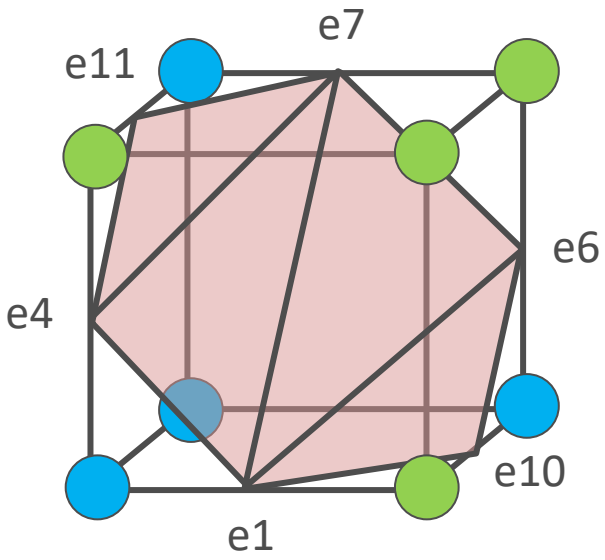


Marching Cubes

SIEMENS
Ingenuity for life

- Step 4 cont.: Get triangulation from table
  - Example for index = 10001101
  - Table at entry 10001101 stores:

    #triangles = 4
    triangle 1 = e4, e7, e11
    triangle 2 = e1, e7, e4
    triangle 3 = e1, e6, e7
    triangle 4 = e1, e10, e6

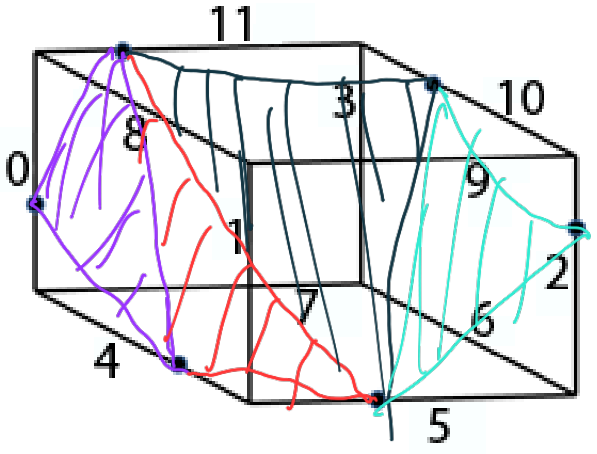


Marching Cubes

SIEMENS
Ingenuity for life

- Exercise

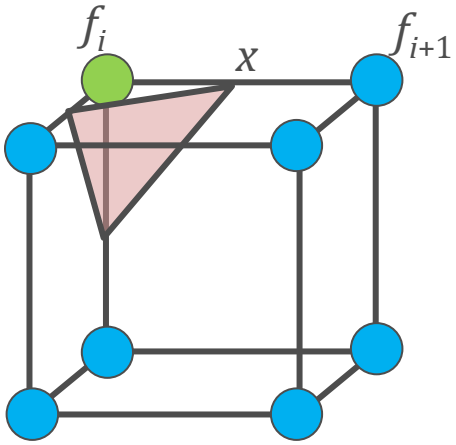
Cubic cell with intersection points with iso-surface. Intersection points are marked by dots. Edges are numbered as shown. Write down all the information that would be stored in the pre-computed Marching Cubes table.



(Note: multiple solutions possible, based on triangulation)

Marching Cubes

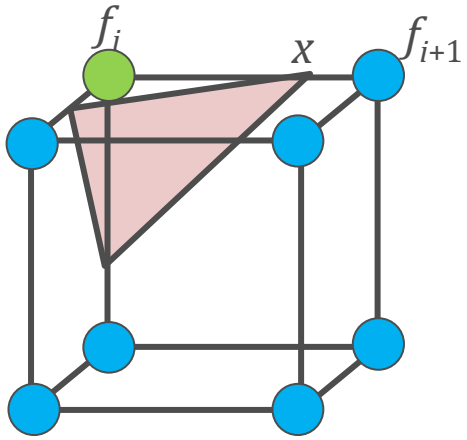
- Step 5: For each triangle edge, find the vertex location along the edge using linear interpolation of the vertex values



Iso=5

= 0

= 10



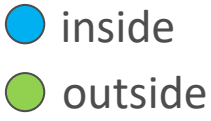
Iso=8

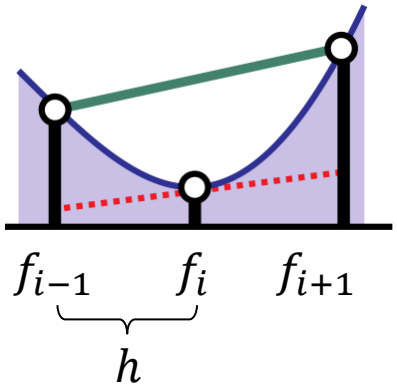
$$x = i + \left(\frac{Iso - f_i}{f_{i+1} - f_i} \right)$$

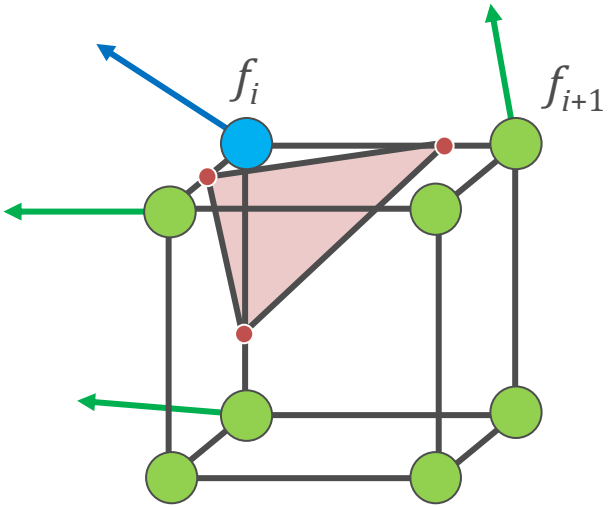
Marching Cubes

- Step 6: Calculate normals at each cube vertex (via finite differences), and interpolate along the edges
  - Normal vector at vertex (i, j, k):

$$\nabla\rho(\boldsymbol{x}) \approx \begin{pmatrix} (f_{i+1,j,k} - f_{i-1,j,k})/2h \\ (f_{i,j+1,k} - f_{i,j-1,k})/2h \\ (f_{i,j,k+1} - f_{i,j,k-1})/2h \end{pmatrix}$$







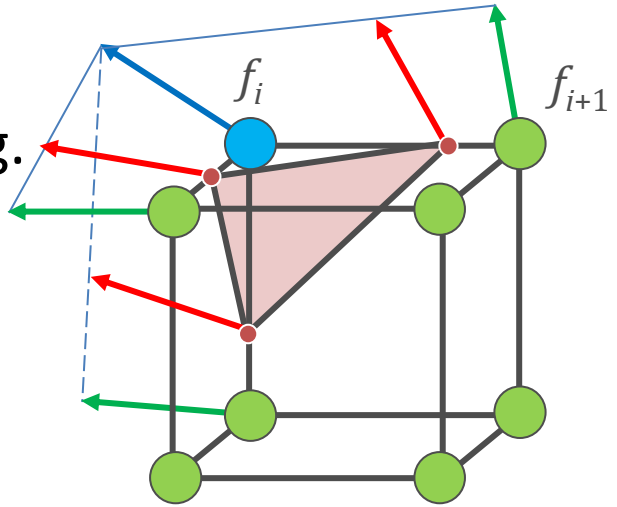
Marching Cubes

SIEMENS
Ingenuity for life

- Step 6: Calculate normals at each cube vertex (via finite differences), and interpolate along the edges
  - Normal vector at vertex (i, j, k):

$$\nabla\rho(\boldsymbol{x}) \approx \begin{pmatrix} (f_{i+1,j,k} - f_{i-1,j,k})/2h \\ (f_{i,j+1,k} - f_{i,j-1,k})/2h \\ (f_{i,j,k+1} - f_{i,j,k-1})/2h \end{pmatrix}$$

  - Linear interpolation along edges, e.g. along x: $\alpha \cdot \nabla\rho_{i+1,j,k} + (1-\alpha) \cdot \nabla\rho_{i,j,k}$

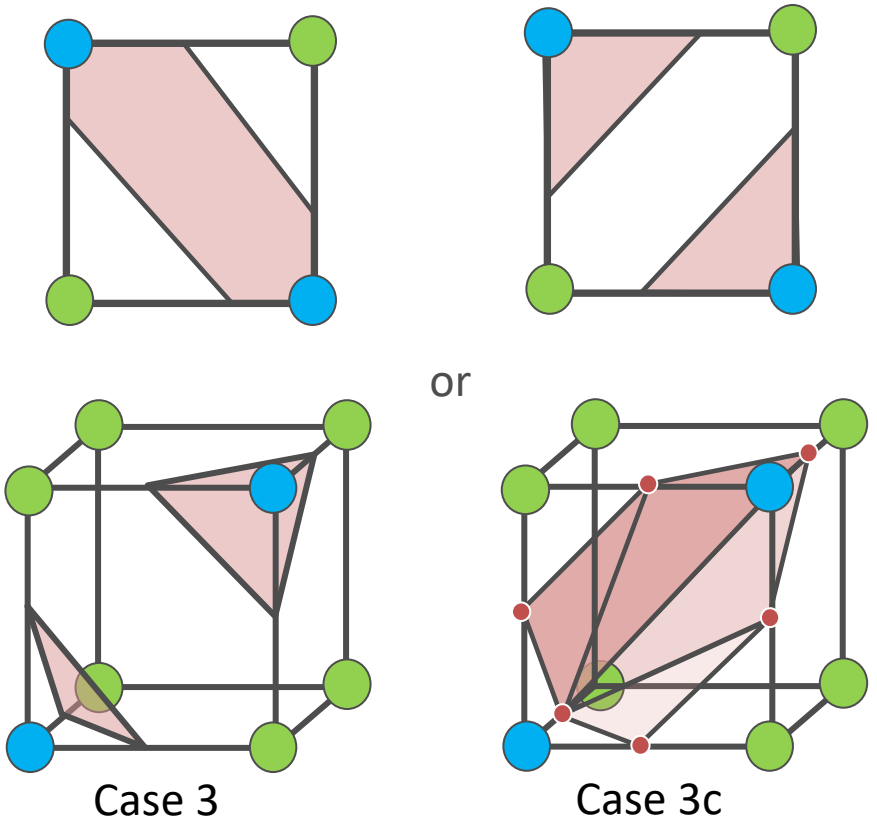


inside
outside

Marching Cubes

SIEMENS
Ingenuity for life

- Step 7: Consider ambiguous cases
  - Ambiguous cases:
    3, 6, 7, 10, 12, 13
  - Use decider as in 2D

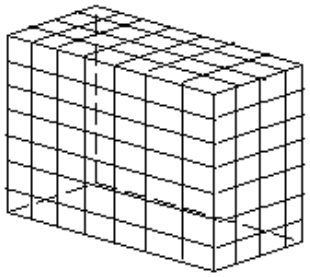


inside
outside

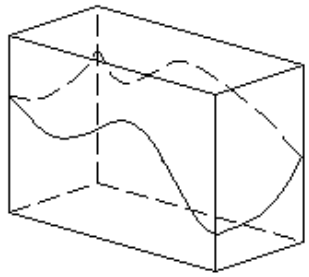
Marching Cubes

SIEMENS
Ingenuity for life

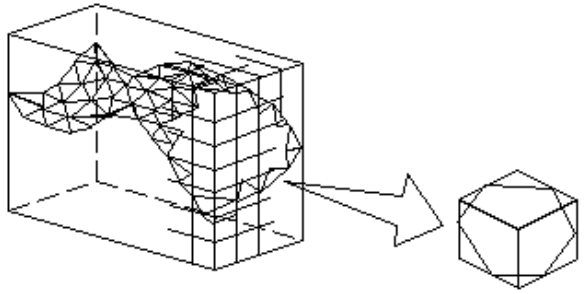
- Summary
  - 256 Cases
    - Reduce to 15 cases by symmetry
  - Ambiguity resides in cases 3, 6, 7, 10, 12, 13
  - Causes holes if arbitrary choices are made
- Up to 5 triangles per cube
- Large datasets can result in several millions of triangles



(a) Volume data



(b) Isosurface
S = f(x,y,z)

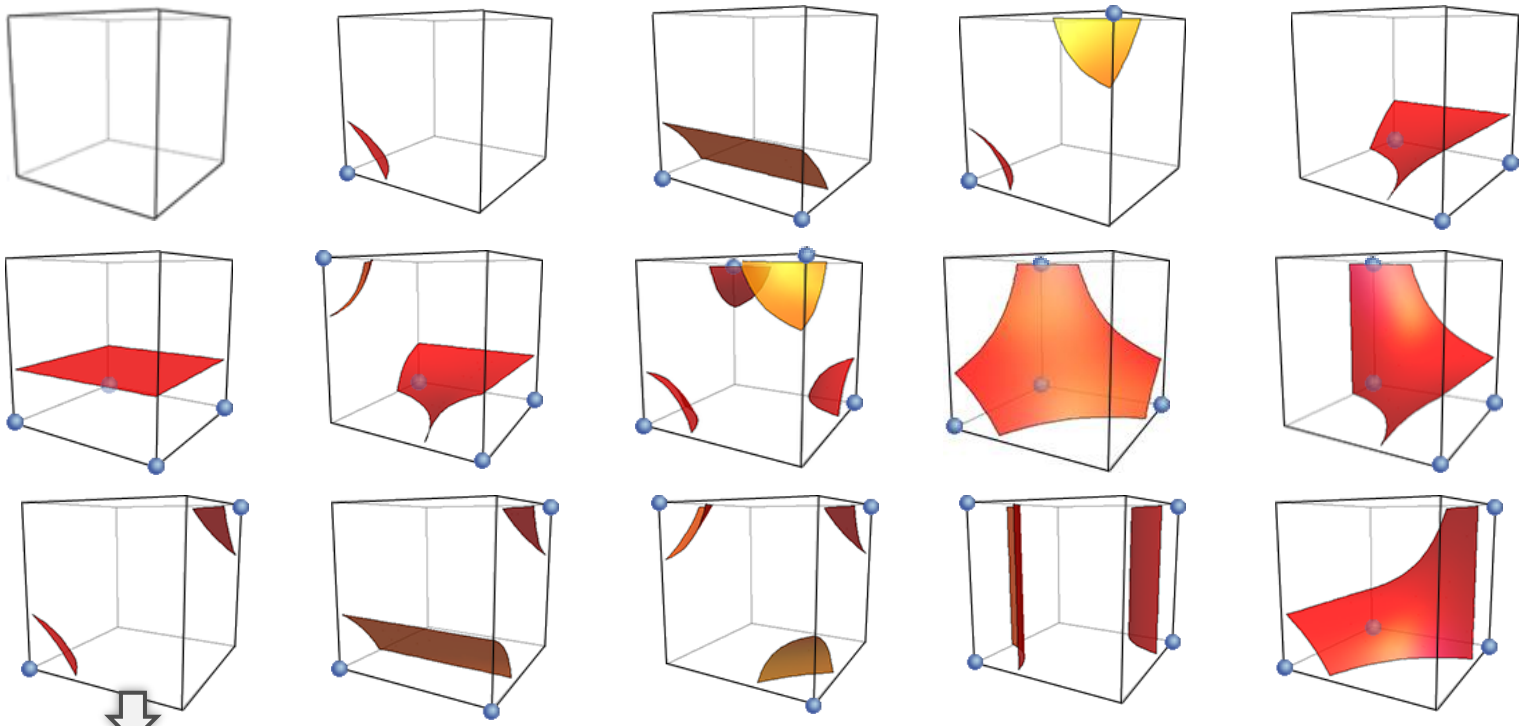


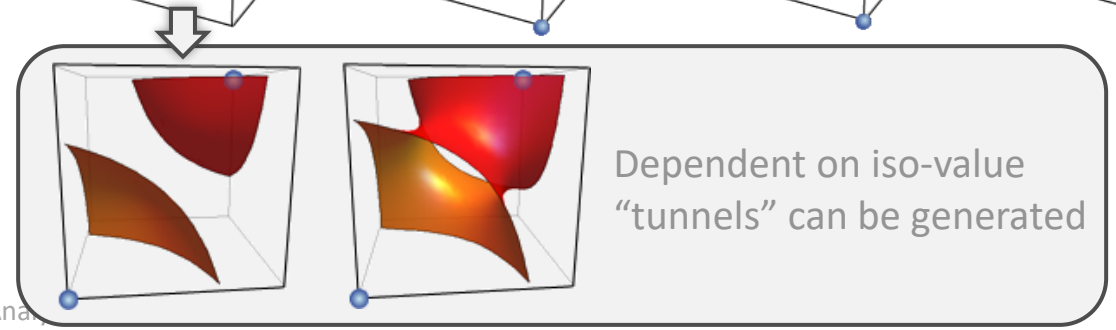
(c) Polygonal Approximation

Marching Cubes

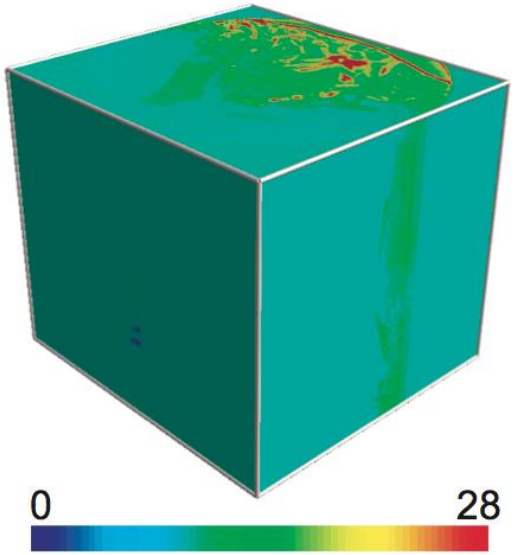
SIEMENS
Ingenuity for life

- Summary: Note that triangulation is only approximation of true isosurfaces produced by trilinear interpolation



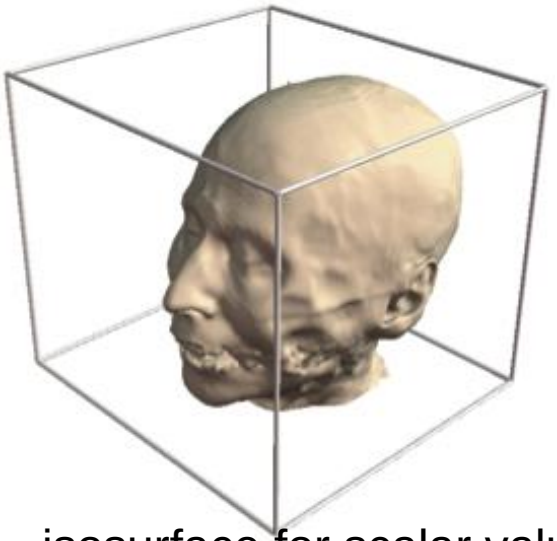


Marching Cubes

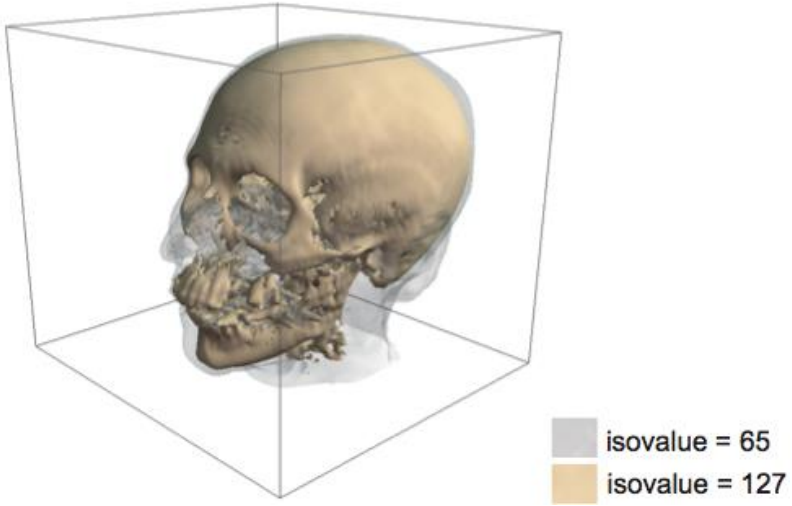


scalar CT volume
(tissue density)

isosurfaces



isosurface for scalar value
corresponding to skin



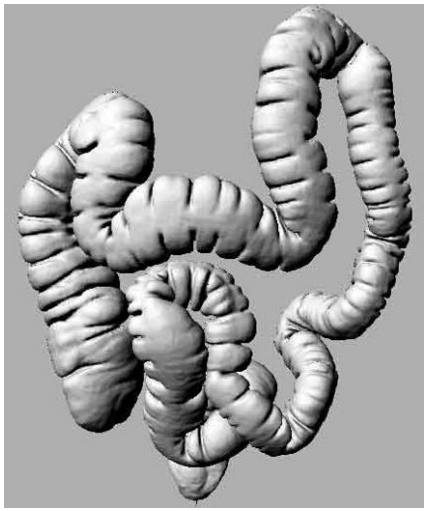
isosurfaces for skin and bone

- extremely simple to use tool
- insightful results

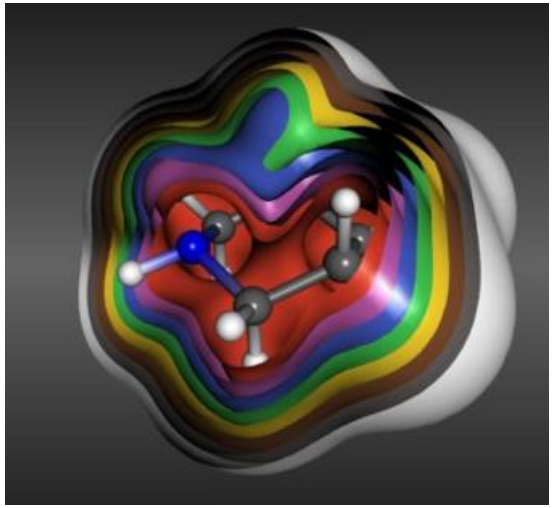
[A. Telea]

Marching Cubes

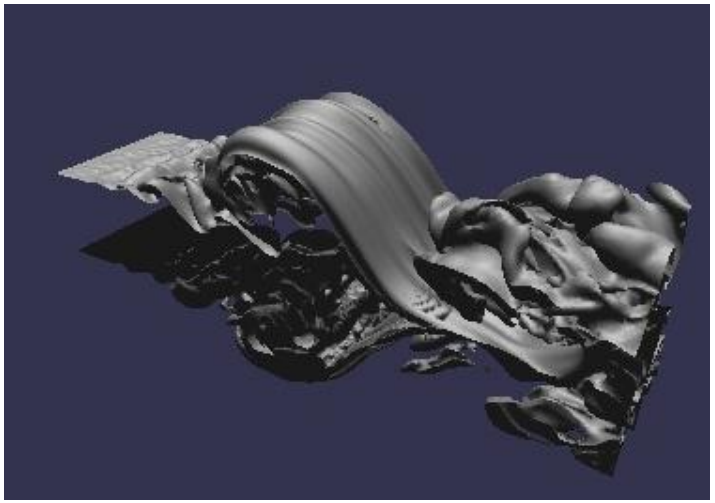
SIEMENS
Ingenuity for life



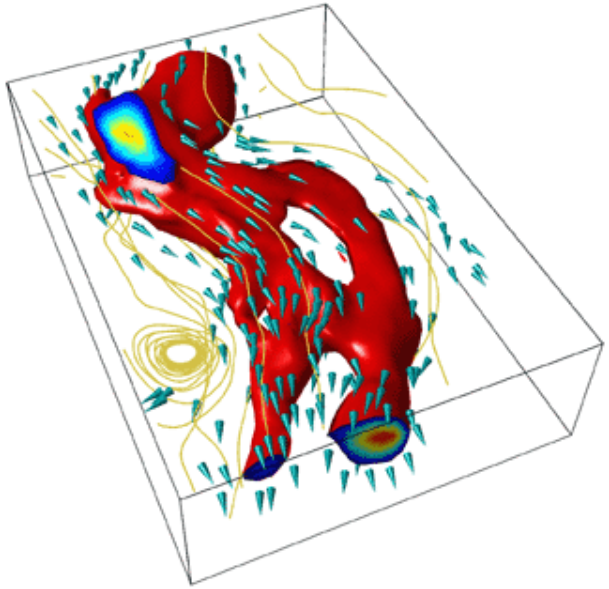
Colon (CT dataset)



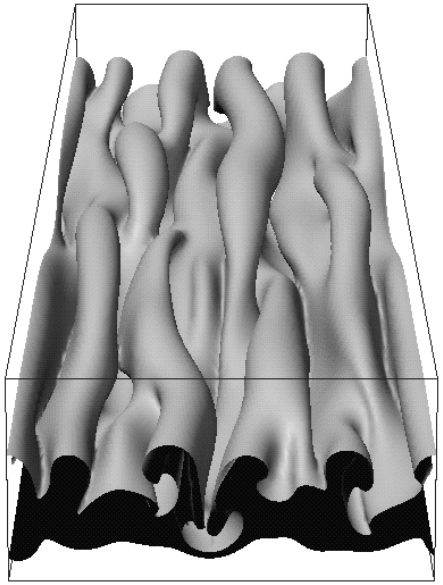
Electron density in molecule



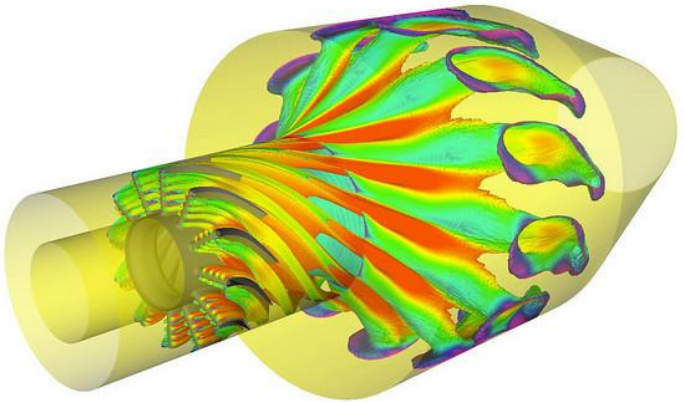
Velocity in 3D fluid flow



Velocity in 3D fluid flow



Magnetic field in sunspots



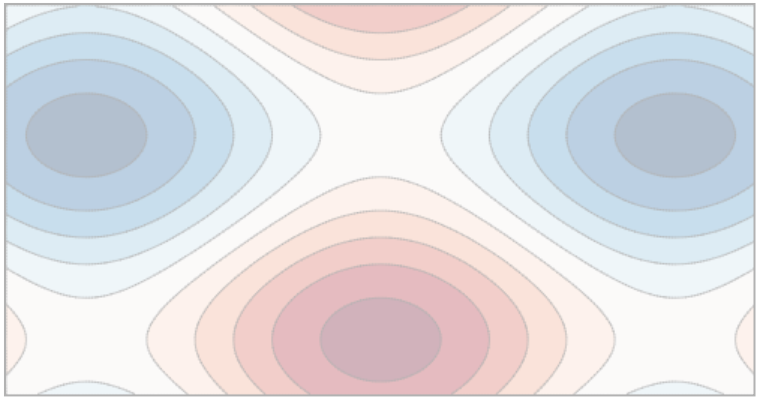
Fuel concentration, colored by temperature in jet engine

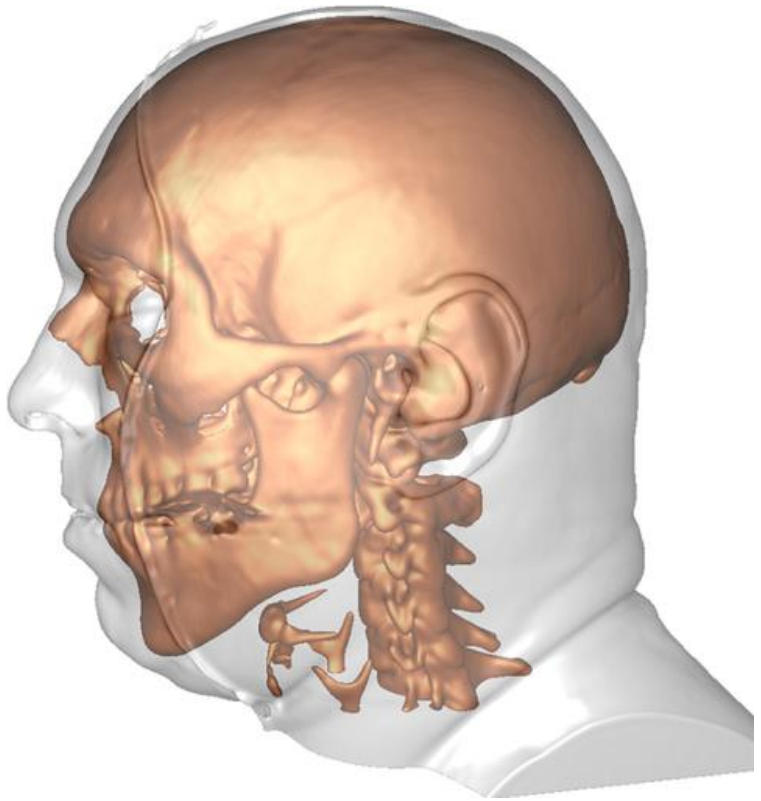
[A. Telea]

Overview

SIEMENS
Ingenuity for life

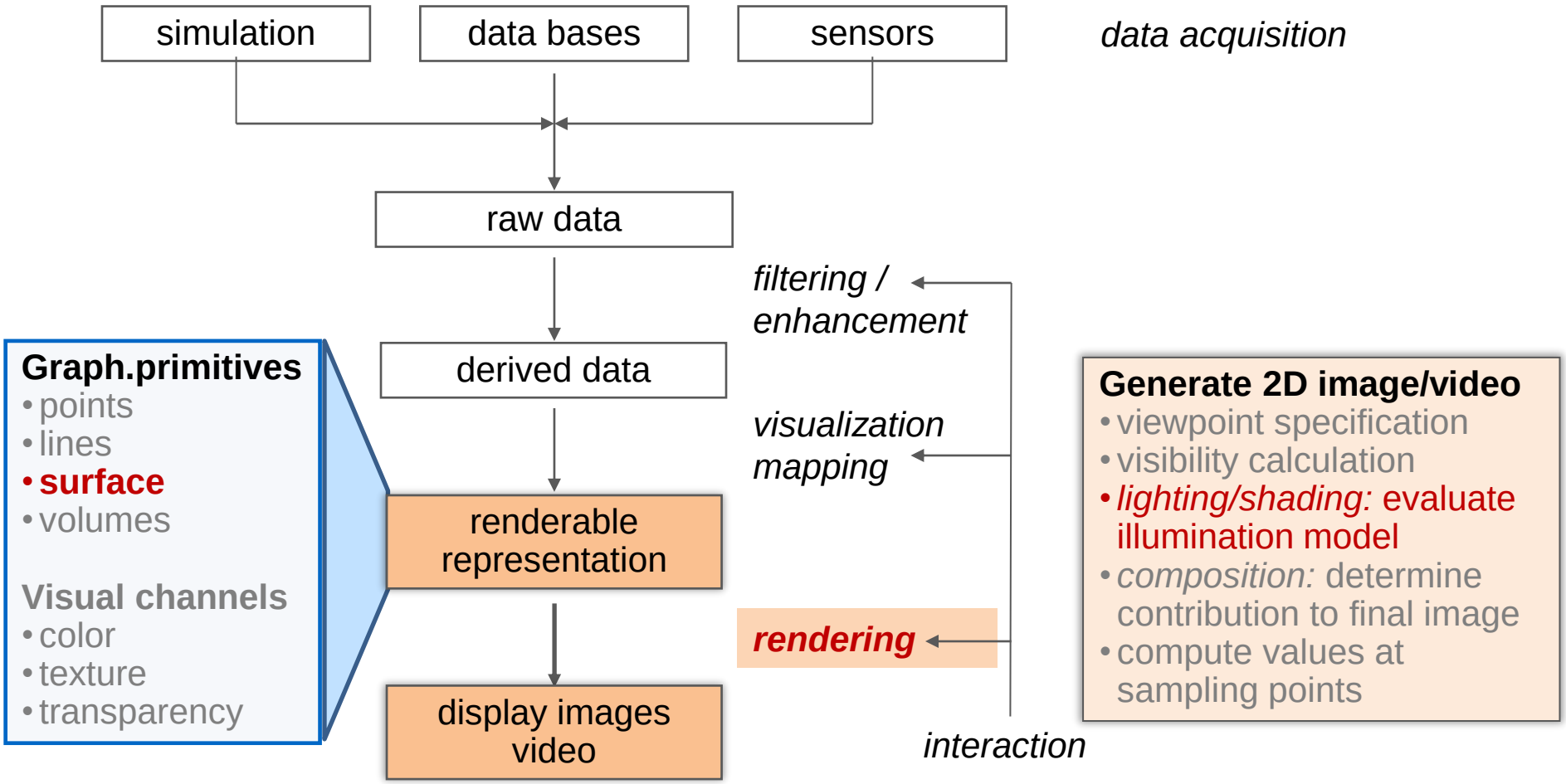
- Isolines & Isosurfaces
  - Marching squares
  - Marching cubes
- Lighting
  - Phong illumination model
- Gradient approximation





Lighting

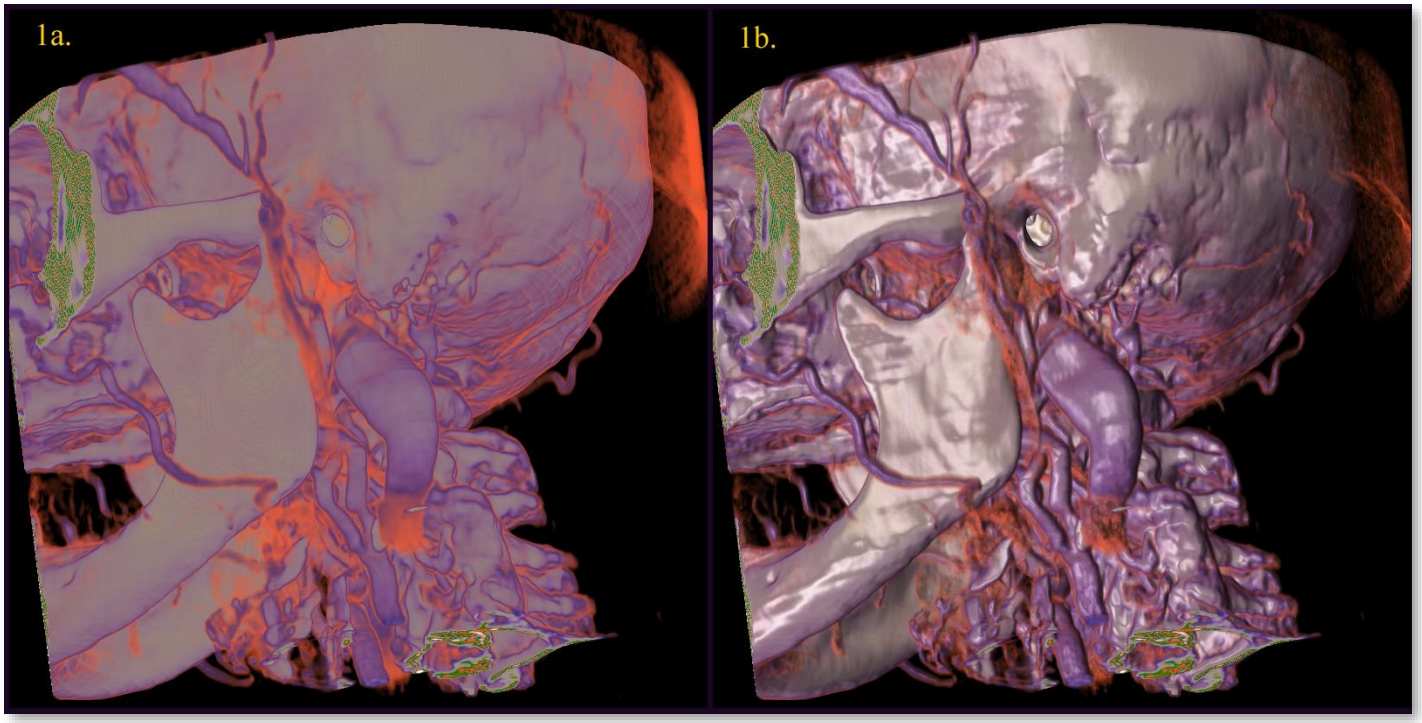
SIEMENS
Ingenuity for life



Lighting

SIEMENS
Ingenuity for life

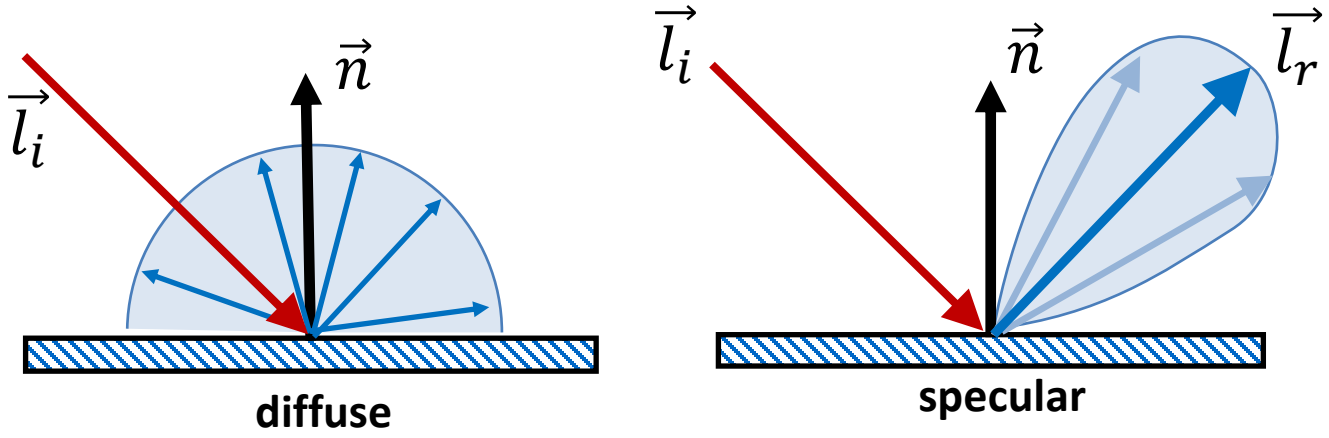
- Necessary to emphasize iso-surface shape
  - Simulate reflection of light and effect on color



Lighting

SIEMENS
Ingenuity for life

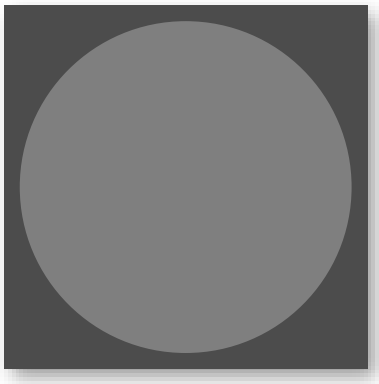
- Phong's illumination model
  - Considers ambient light and point lights as well as the material color and reflection properties
  - Ambient light: background light, constant everywhere
  - Diffuse reflector reflects equally into all directions
  - Specular reflector reflects mostly into the mirror direction



Lighting

SIEMENS
Ingenuity for life

- Ambient light: $C = k_a C_a O_d$
  - Background light, constant everywhere
  - k_a … ambient reflection coefficient $\in [0, 1]$
  - C_a … color of the ambient light
  - O_d … object color

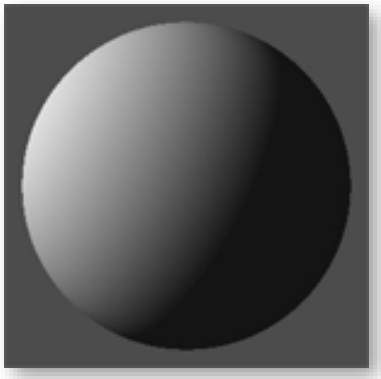


We know that light does't come directly always.

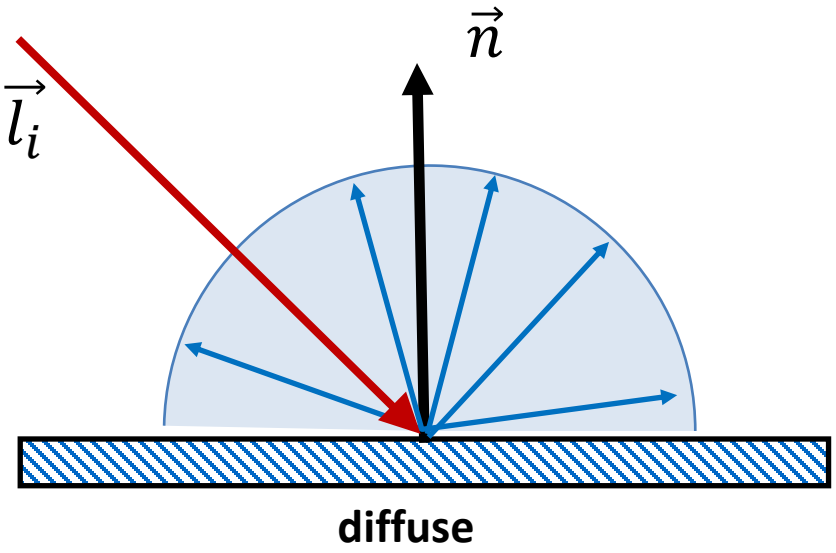
Generally, Light reflect from objects and comes our object. Ambient light is the basic color of objet. it is constant.

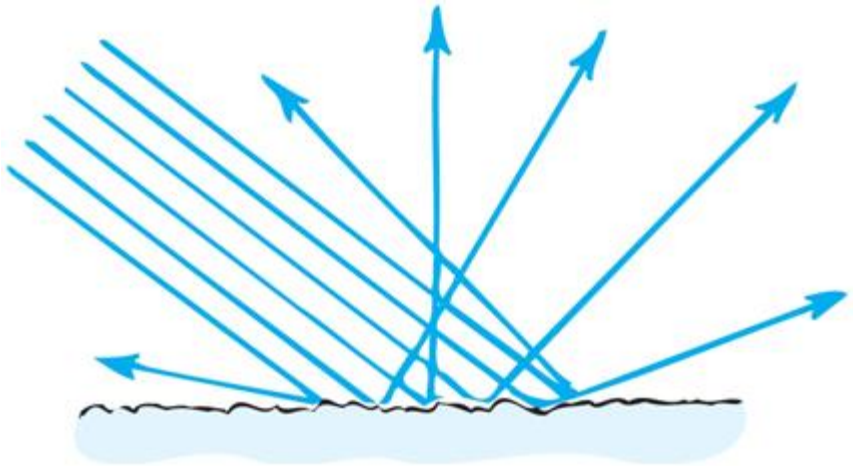
Lighting

SIEMENS
Ingenuity for life



- Diffuse reflection (Lambert term)
  - Connected to physical reality (Lambert's law)
  - Approximates rough surfaces
  - Scatters light equally in all directions

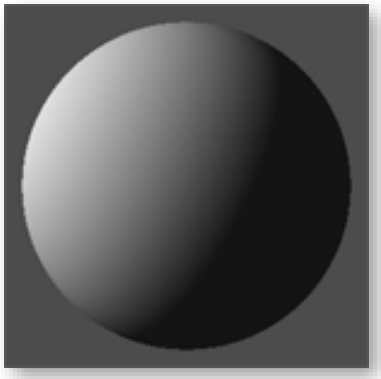




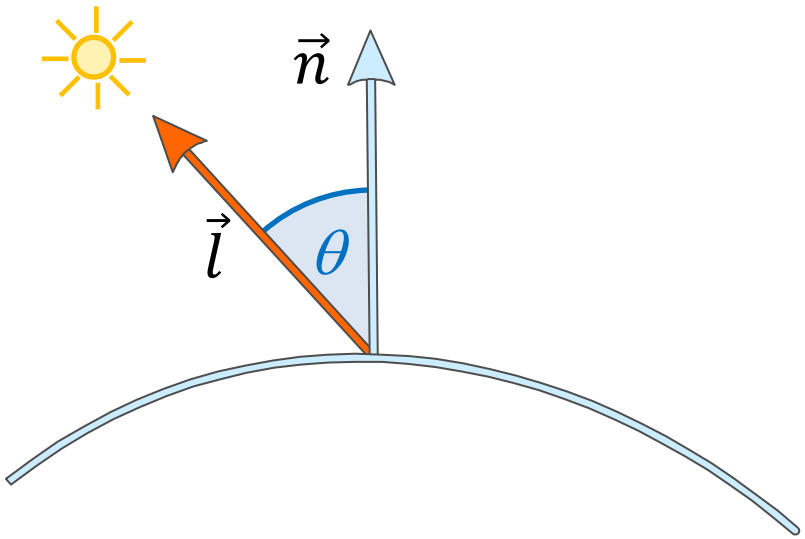
http://multivis.net/lecture/phong.html

Lighting

SIEMENS
Ingenuity for Life



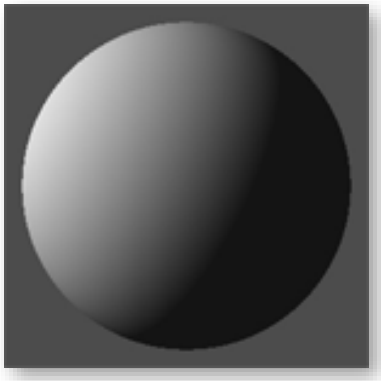
- Diffuse reflection: $C = k_d C_p O_d \cos\theta$
  - k_d … diffuse reflection coefficient $\in [0, 1]$
  - C_p … color of the point light
  - O_d … object color
  - $\cos\theta$ … angle between light vector $\vec{l}$ and normal $\vec{n}$



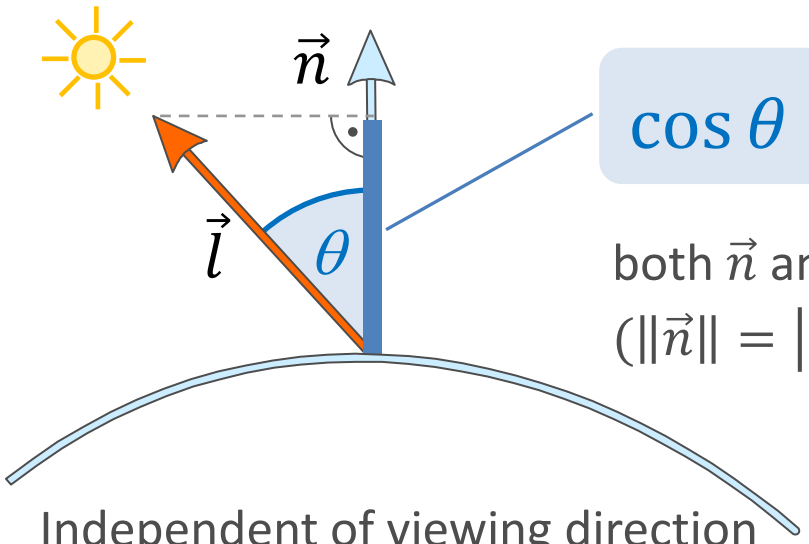
http://multivis.net/lecture/phong.html

Lighting

SIEMENS
Ingenuity for life



- Diffuse reflection: $C = k_d C_p O_d \cos\theta$
  - k_d … diffuse reflection coefficient $\in [0, 1]$
  - C_p … color of the point light
  - O_d … object color
  - $\cos\theta$ … angle between light vector $\vec{l}$ and normal $\vec{n}$



Independent of viewing direction

$$\cos\theta = \vec{n} \cdot \vec{l}$$

→ dot product.

both $\vec{n}$ and $\vec{l}$ have unit length
$(\|\vec{n}\| = \|\vec{l}\| = 1)$

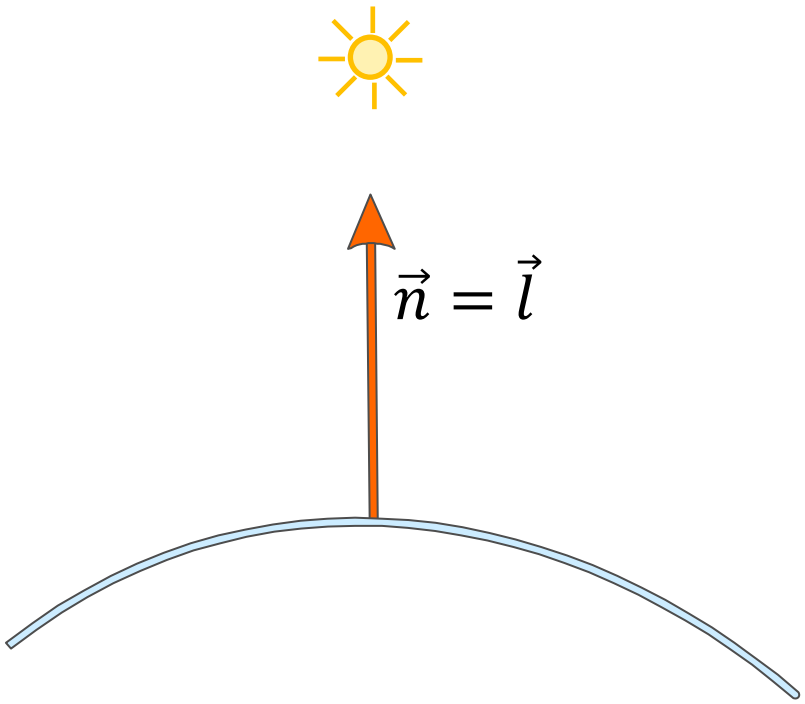
Bir nesneye yukarıdan ışık kaynağını tutarsak en parlak görüntüyü alırız. Bunun nedeni gelen ışıkla normalin çarpımlarının 1 olması.

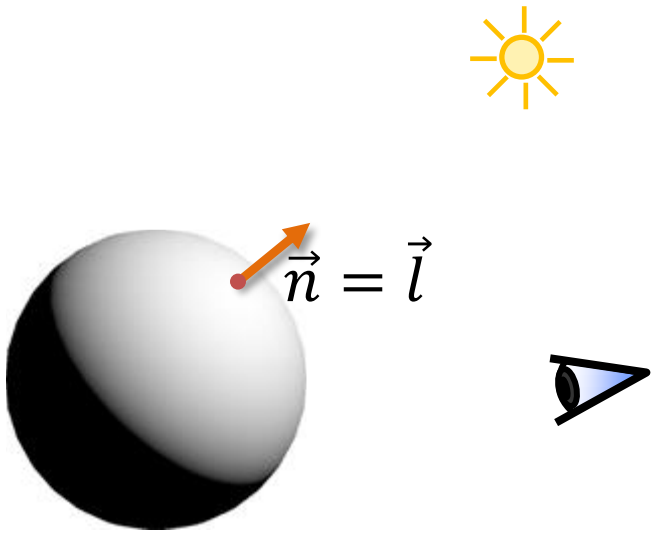
http://multivis.net/lecture/phong.html

Lighting

SIEMENS
Ingenuity for life

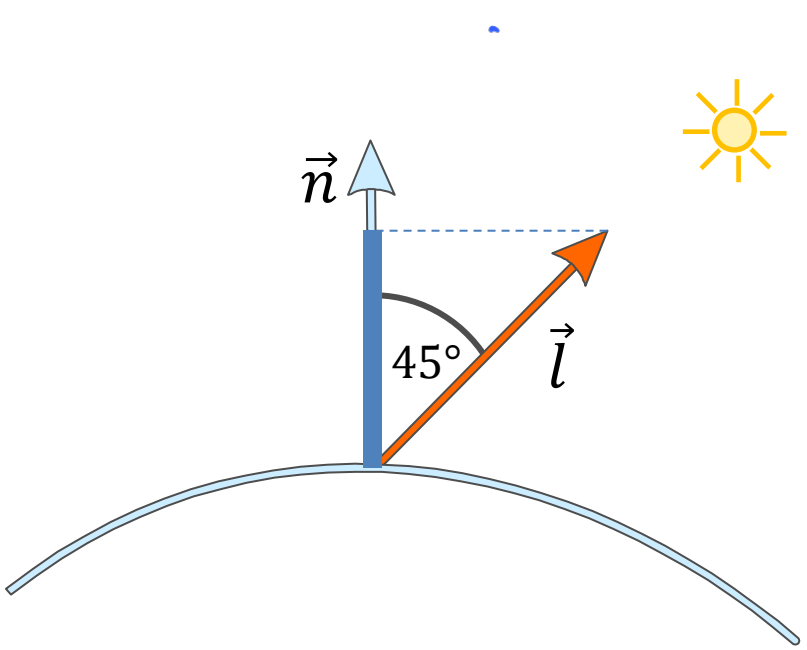
- Diffuse reflection: $C = k_d C_p O_d \cos\theta$
  - The light is precisely above the point ($\theta = 0$)
  - $\cos 0° = 1$ → 100% intensity

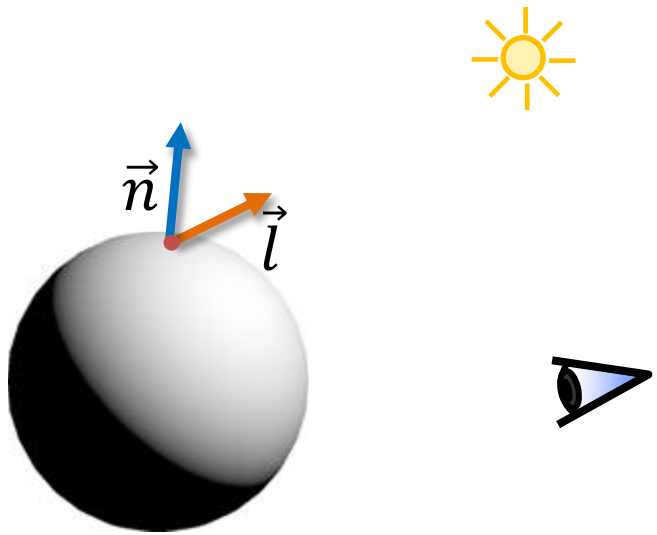




Lighting

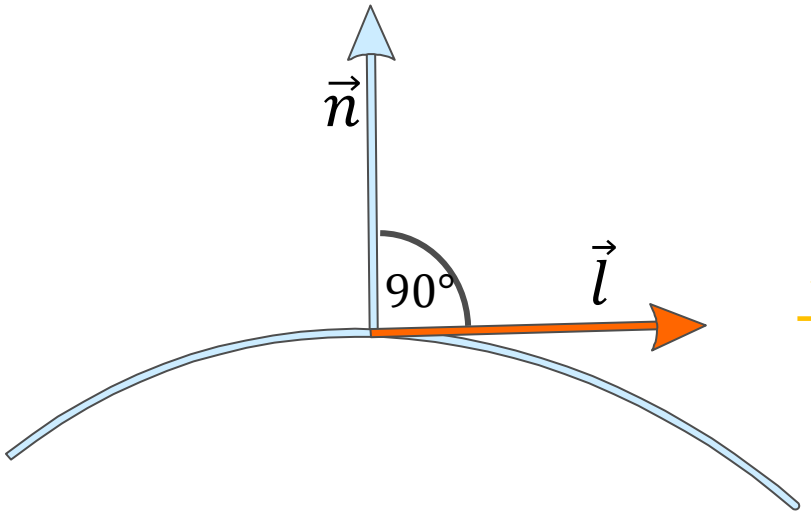
- Diffuse reflection: $C = k_d C_p O_d \cos\theta$
  - The light shines onto a point in a 45° angle
  - $\cos 45° \approx 0.7$ → 70% intensity

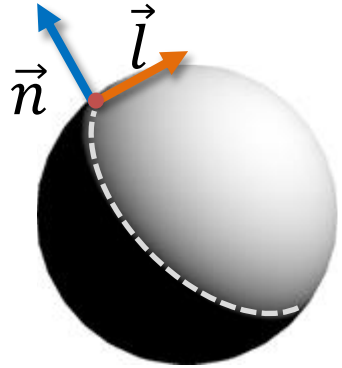




Lighting

- Diffuse reflection: $C = k_d C_p O_d \cos\theta$
  - The light shines onto a point in a 90° angle
  - $\cos 90° = 0$ → 0% intensity



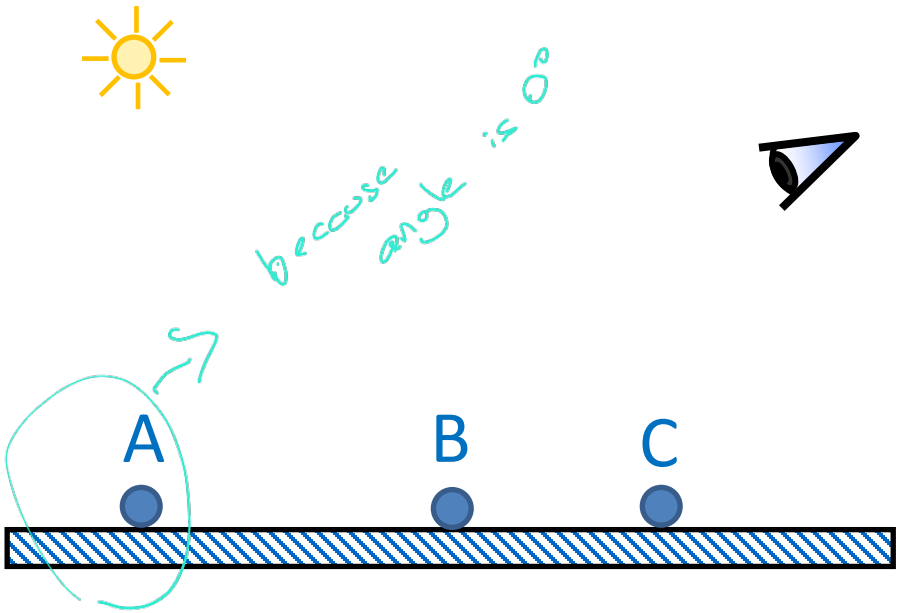


The flatter light falls on a surface, the darker it will appear

Lighting

SIEMENS
Ingenuity for life

- Plane with only diffuse reflection
  - At which point does the viewer see highest diffuse reflection?

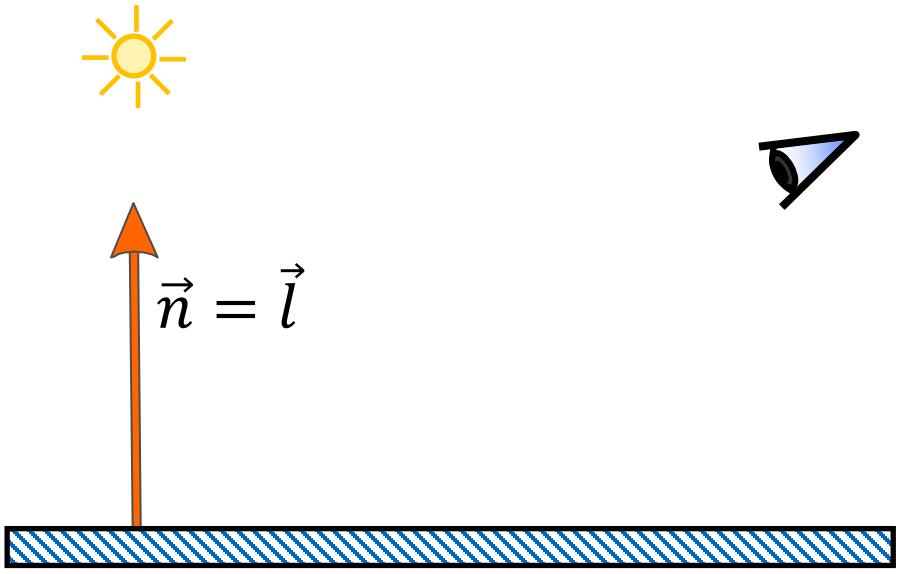


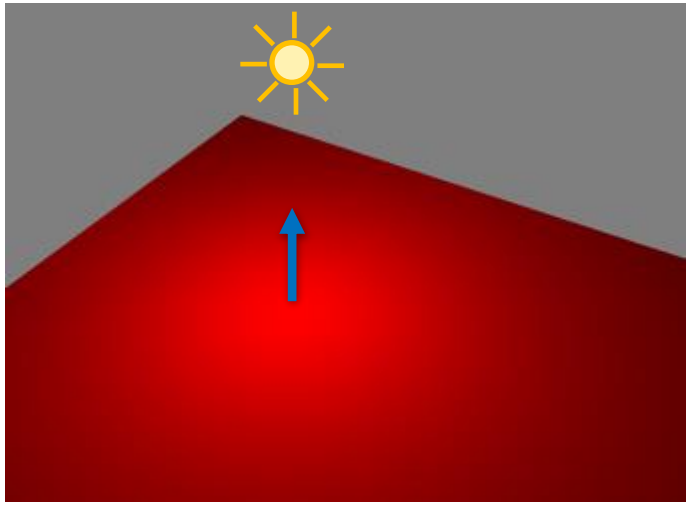
http://multivis.net/lecture/phong.html

Lighting

SIEMENS
Ingenuity for life

- Plane with only diffuse reflection
  - At which point does the viewer see highest diffuse reflection?
  - Max. intensity → $\cos\theta = 1$
  - Normal vector = light vector
  - Point precisely below light source



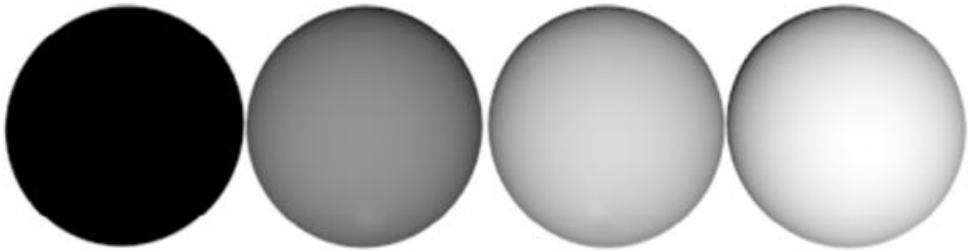


http://multivis.net/lecture/phong.html

Lighting

SIEMENS
Ingenuity for life

- Diffuse reflection: $C = k_d C_p O_d \cos\theta$
  - Varying k_d (diffuse reflection coefficient)

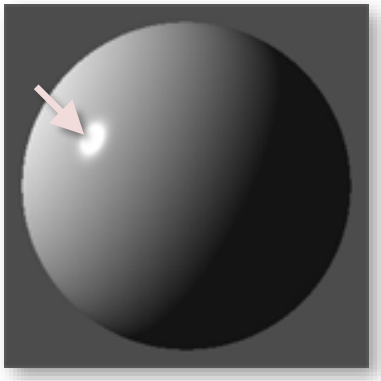


$k_d = 0$ $k_d = 0.3$ $k_d = 0.7$ $k_d = 1$

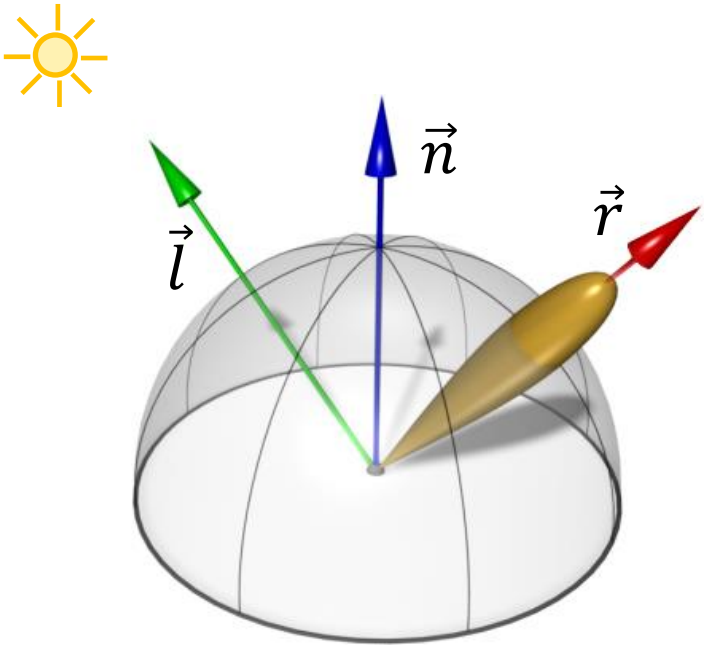
http://multivis.net/lecture/phong.html

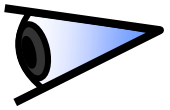
Lighting

SIEMENS
Ingenuity for life



- Specular reflection
  - Highlight = reflection of light source
  - Glossy surfaces
  - Reflects mostly into the mirror direction
  - View dependent!

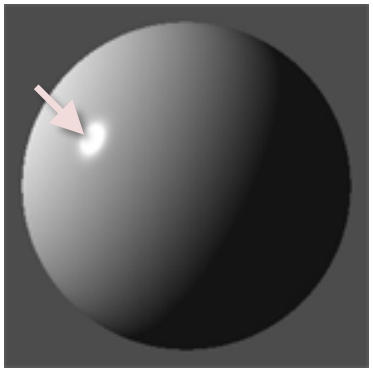


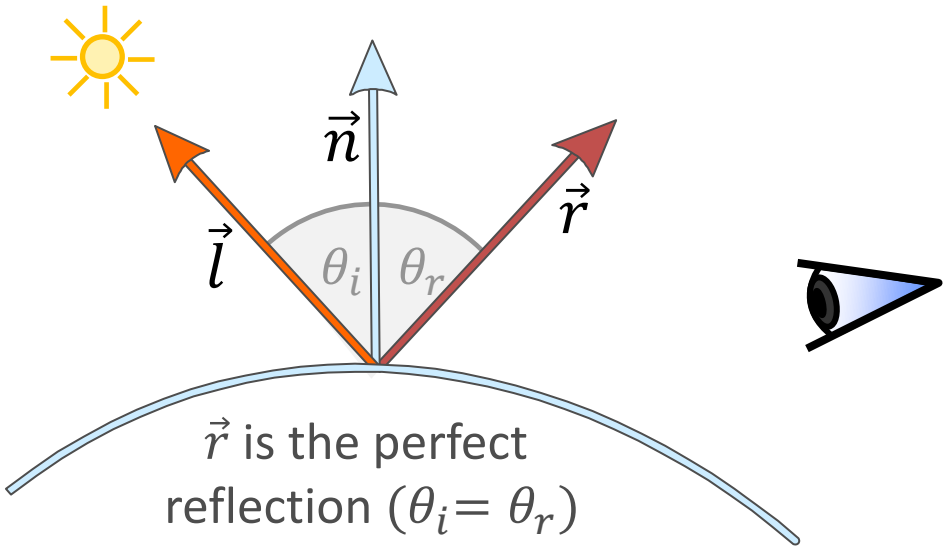


Lighting

SIEMENS
Ingenuity for life

- Specular reflection: $C = k_s C_p O_d \cos^n \varphi$
  - k_s … specular reflection coefficient $\in [0, 1]$
  - $\cos \varphi$ … angle between the reflected light ray $\vec{r}$ …

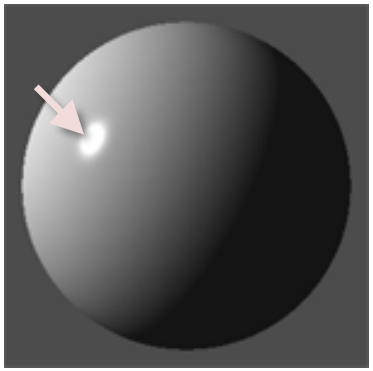




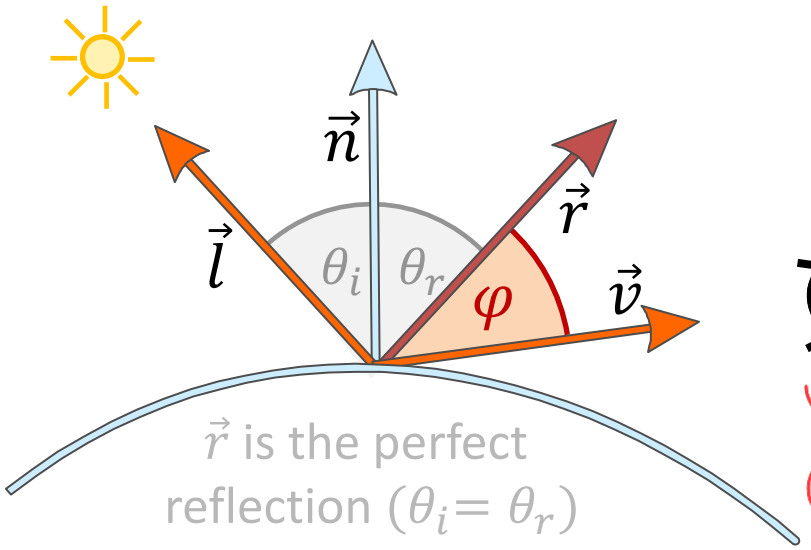
http://multivis.net/lecture/phong.html

Lighting

SIEMENS
Ingenuity for life

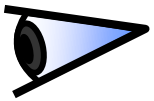


- Specular reflection: $C = k_s C_p O_d \cos^n \varphi$
  - k_s … specular reflection coefficient $\in [0, 1]$
  - $\cos \varphi$ … angle between the reflected light ray $\vec{r}$ and the vector to the viewer $\vec{v}$
  - $(\quad)^n$ … shininess factor (controls extend of highlight)



$$\cos^n \varphi = (\vec{r} \cdot \vec{v})^n$$

$\vec{r}$ and $\vec{v}$ have unit length



Very similar to diffusion

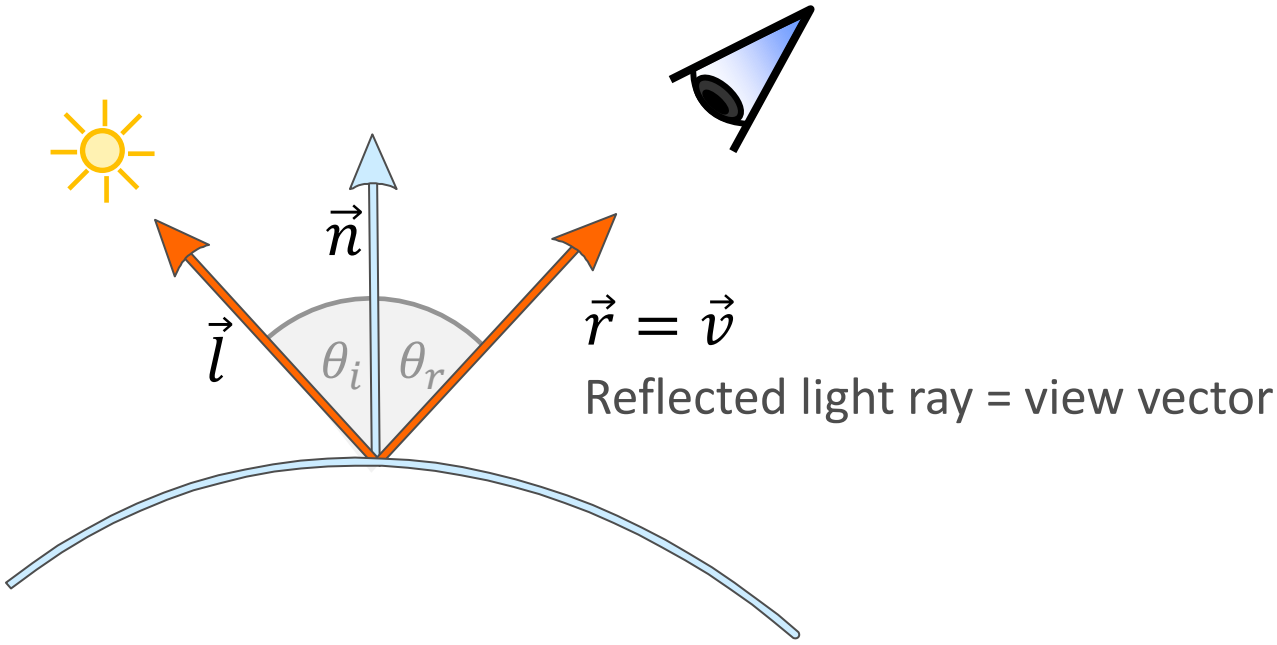
but this time we consider receiver position. We take reflection of L by using n and calculate φ

The view vector $\vec{v}$ is in the formula, so it should look differently if we look from a different angle

Lighting

SIEMENS
Ingenuity for life

- Specular reflection: $C = k_s C_p O_d \cos^n \varphi$
  - We are looking directly into the reflection ($\varphi = 0°$)
  - $(\cos 0°)^n = 1^n$ → 100% intensity



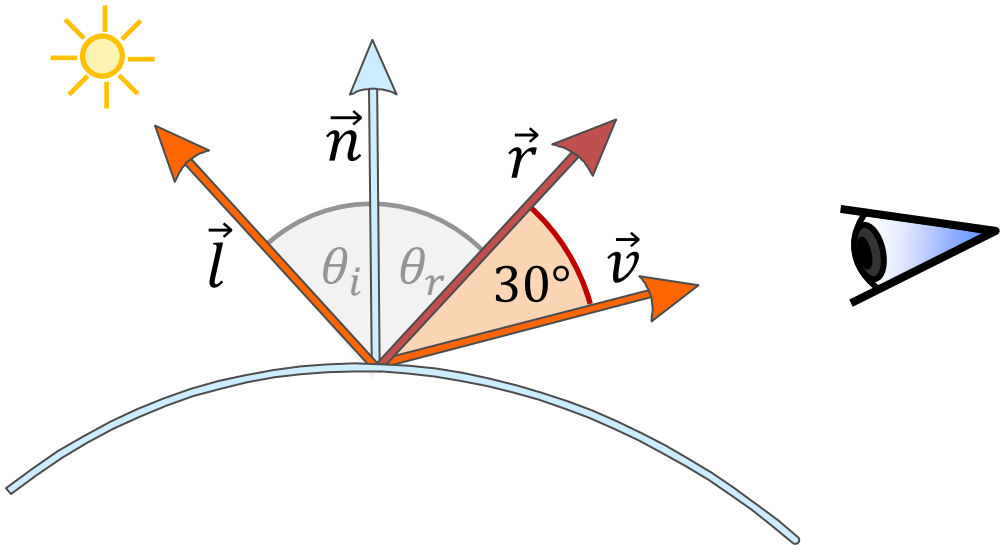
Lighting

SIEMENS
Ingenuity for life

- Specular reflection: $C = k_s C_p O_d \cos^n \varphi$

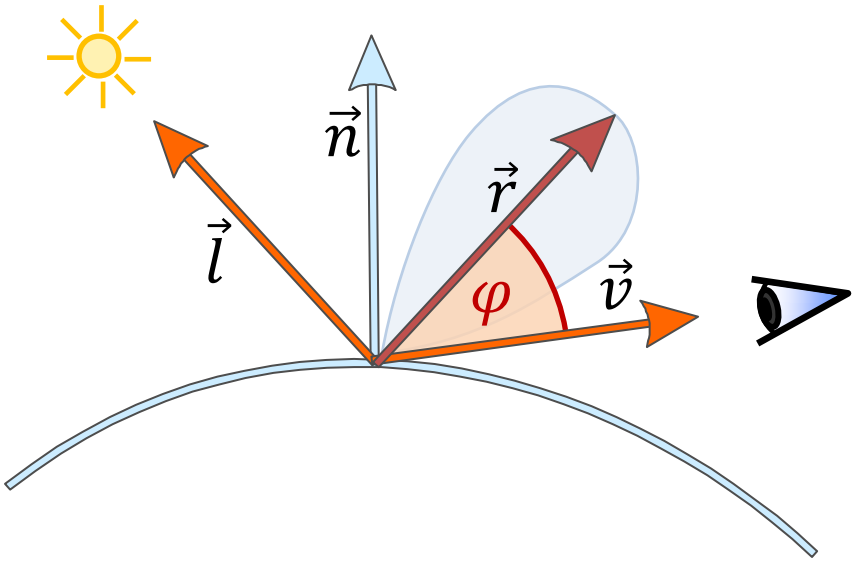
$$\varphi = 30° \rightarrow (\cos 30°)^n = \left(\frac{\sqrt{3}}{2}\right)^n \approx 0.87^n$$

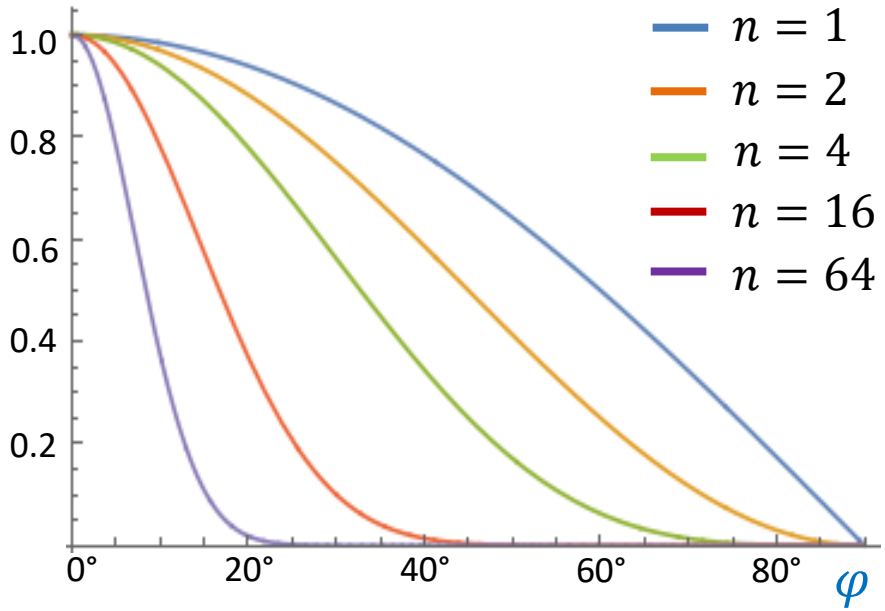
- Case $n = 2$ $\rightarrow \left(\frac{\sqrt{3}}{2}\right)^2 = \frac{3}{4}$ → 75% intensity
- Case $n = 4$ $\rightarrow \left(\frac{\sqrt{3}}{2}\right)^4 = \frac{9}{16} \approx 0.56$ → 56% intensity



Lighting

- Effect of the exponent n of highlight





$$\cos^n \varphi = (\vec{r} \cdot \vec{v})^n$$

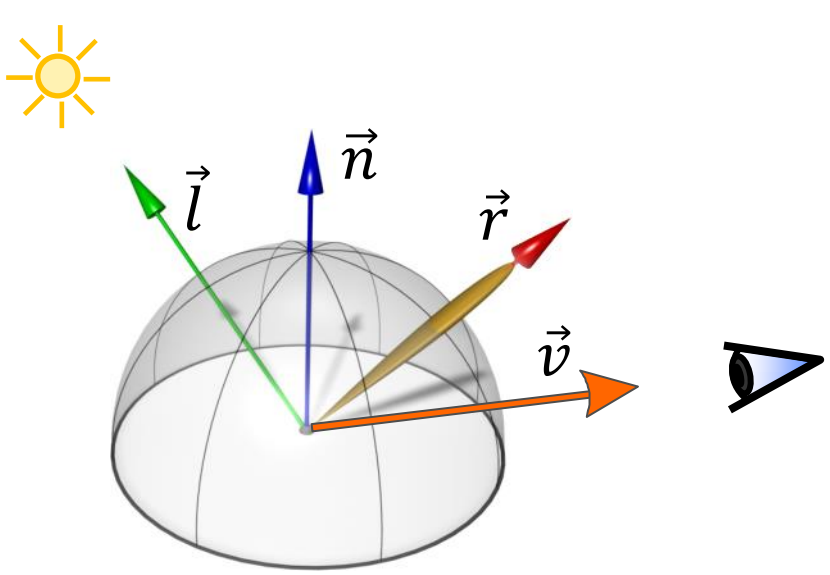
$n = 1$ $n = 2$ $n = 4$ $n = 16$ $n = 64$

Lighting

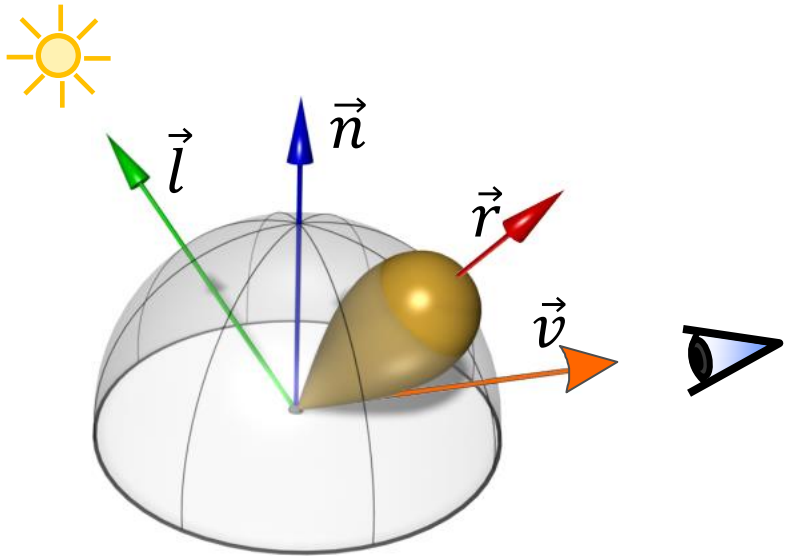
SIEMENS
Ingenuity for life

- Effect of the exponent n of highlight

$$C = k_s C_p O_d \cos^n \varphi$$



Shiny surface (large n)



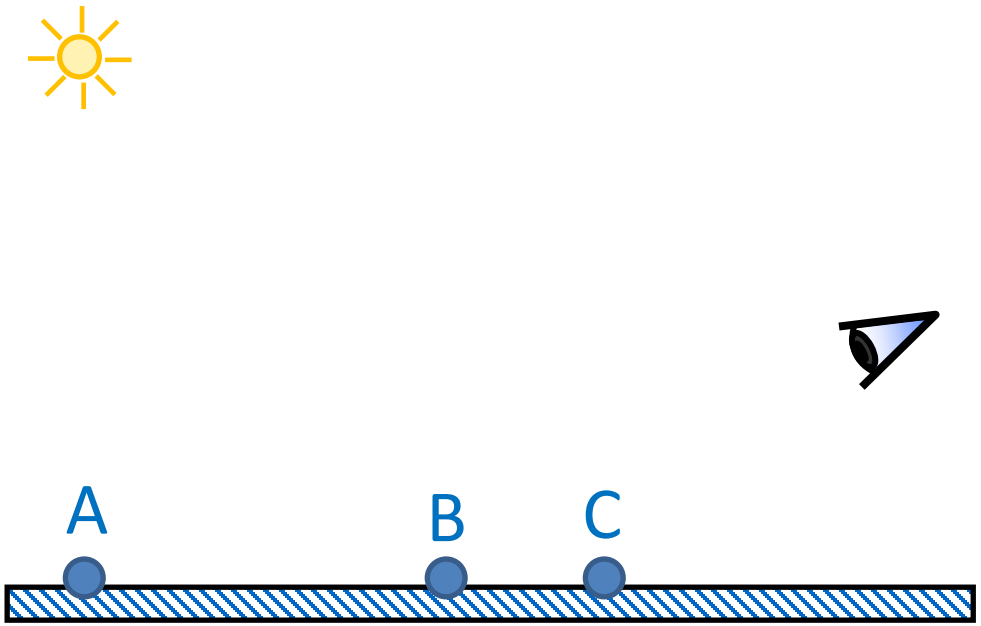
Dull surface (small n)

http://multivis.net/lecture/phong.html

Lighting

SIEMENS
Ingenuity for life

- Plane with diffuse and specular reflection
  - At which point is the highest specular reflection?

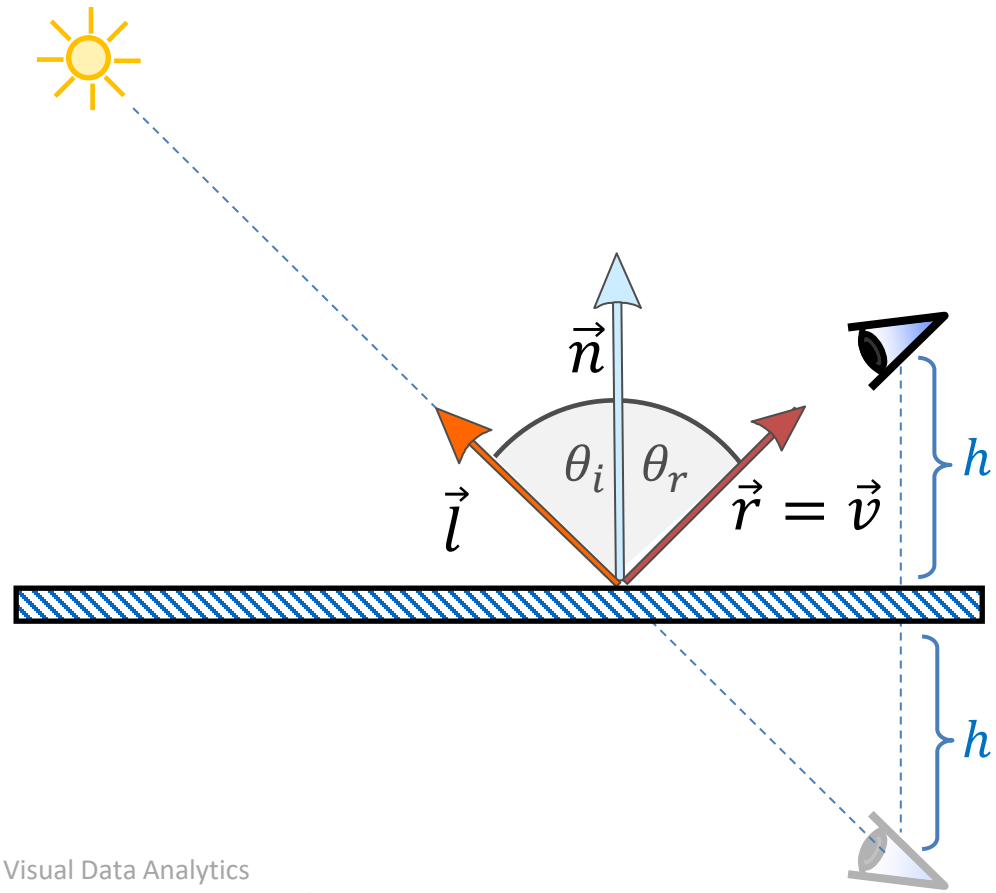


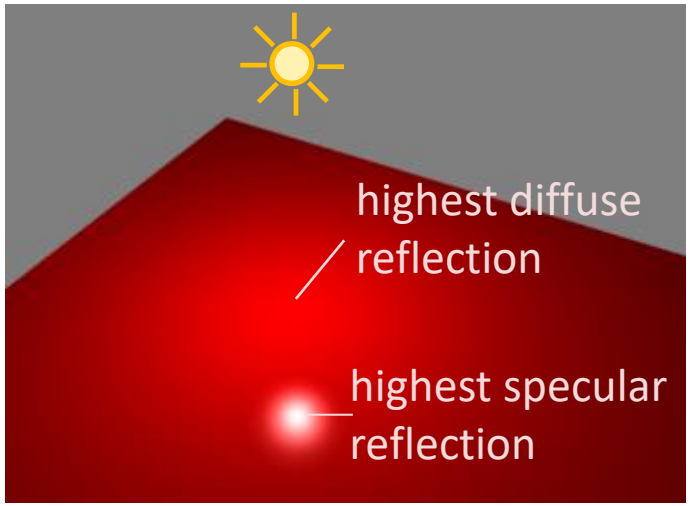
http://multivis.net/lecture/phong.html

Lighting

- Plane with diffuse and specular reflection
  - At which point is the highest specular reflection?
  - Reflected vector = view vector

Two of them are in different position

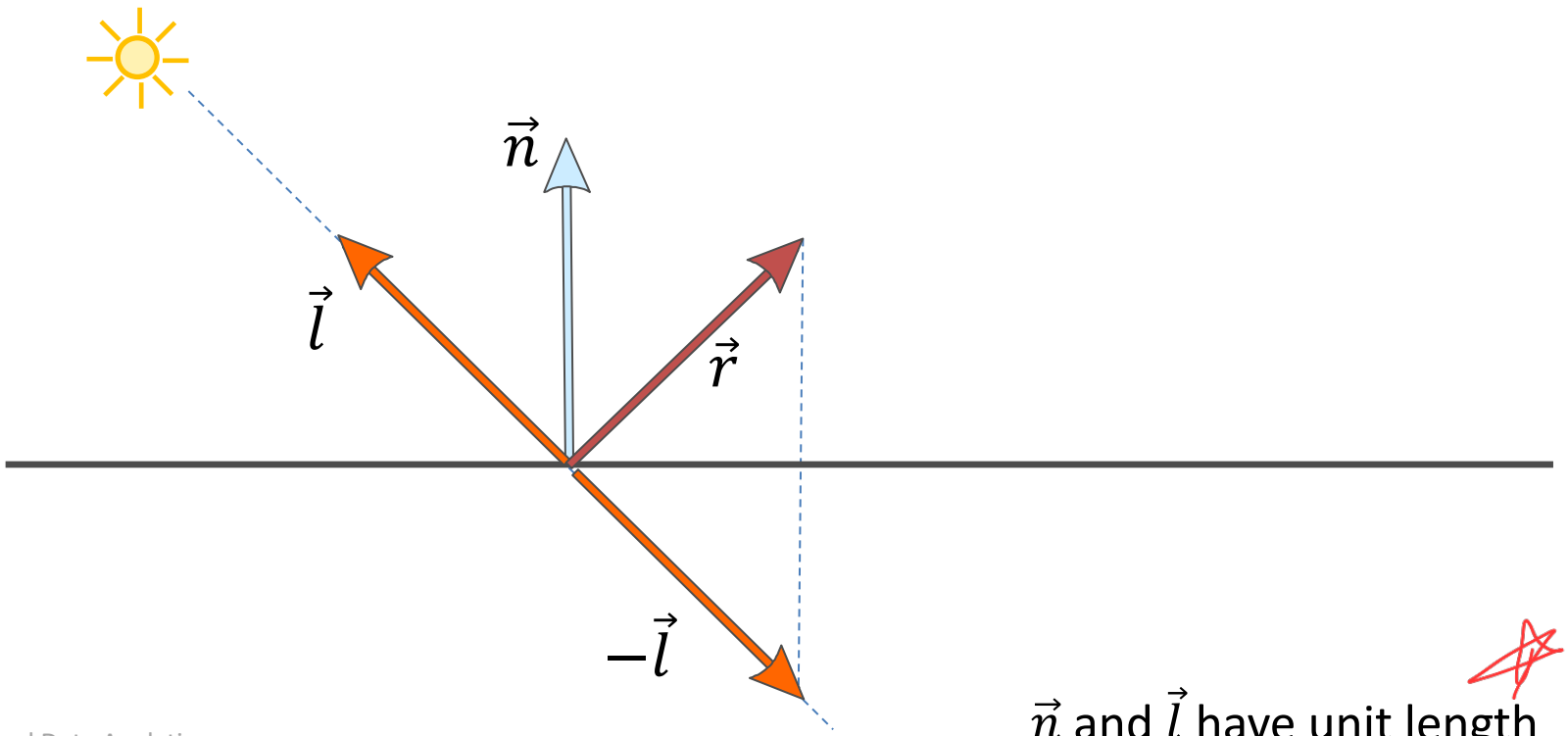




http://multivis.net/lecture/phong.html

Lighting

- Calculation of reflected light ray $\vec{r}$



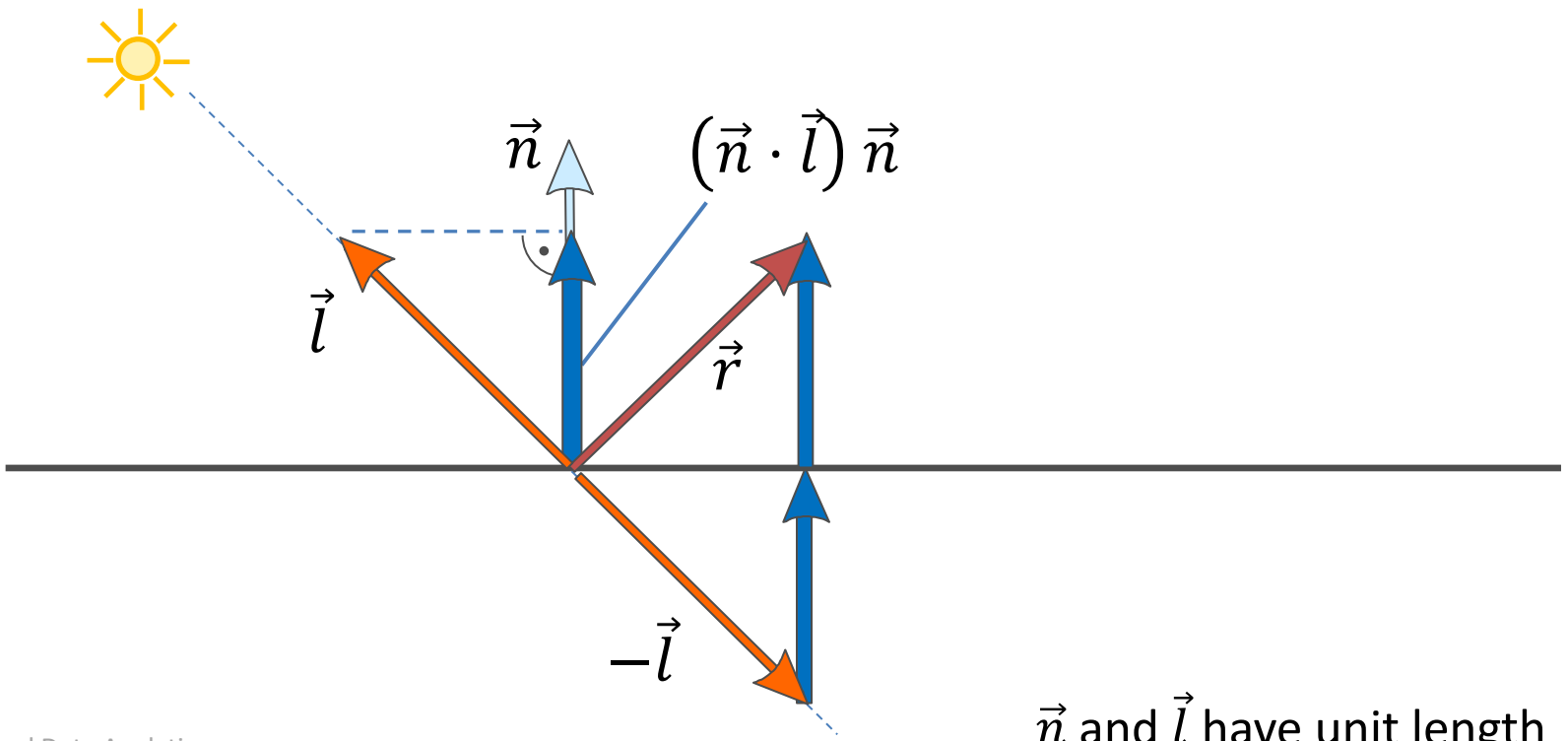
$\vec{n}$ and $\vec{l}$ have unit length

Lighting

SIEMENS
Ingenuity for life

- Calculation of reflected light ray $\vec{r}$

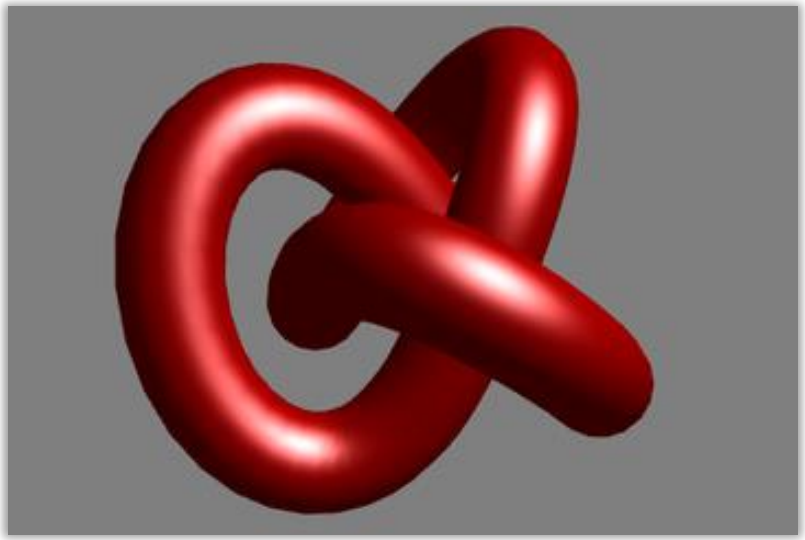
$$\vec{r} = 2(\vec{n} \cdot \vec{l})\,\vec{n} - \vec{l}$$



$\vec{n}$ and $\vec{l}$ have unit length

Lighting

SIEMENS
Ingenuity for life



```
varying vec3 normalInterp;  // Surface normal
varying vec3 vertexPos;     // Vertex position
uniform vec3 lightPos;      // Light position

uniform float Ka;   // Ambient reflection coefficient
uniform float Kd;   // Diffuse reflection coefficient
uniform float Ks;   // Specular reflection coefficient
uniform float shininess; // Shininess

// Material color
uniform vec3 ambientColor;
uniform vec3 diffuseColor;
uniform vec3 specularColor;

void main() {
   vec3 N = normalize(normalInterp);
   vec3 L = normalize(lightPos - vertexPos);

   // Lambert's cosine law
   float lambertian = max(dot(N, L), 0.0);      // use max-function to avoid negative values

   float specular = 0.0;
   if (lambertian > 0.0) {
      vec3 R = reflect(-L, N);        // Reflected light vector
      vec3 V = normalize(-vertexPos); // Vector to viewer with eye position (0, 0, 0)

      // Compute the specular term
      float specAngle = max(dot(R, V), 0.0);
      specular = pow(specAngle, shininess);
   }
   gl_FragColor = vec4(Ka * ambientColor +
                       Kd * lambertian * diffuseColor +
                       Ks * specular * specularColor, 1.0);
}
```


http://multivis.net/lecture/phong.html

Lighting

- Example
  - Ambient reflection coefficient $k_a = \frac{1}{10}$
  - Diffuse reflection coefficient $k_d = \frac{1}{2}$
  - Specular reflection coefficient $k_s = \frac{5}{6}$
  - Shininess factor $n = 2$
  - Light & object color white

$C = k_s c_p O_d (\vec{r} \cdot \vec{v})^n$

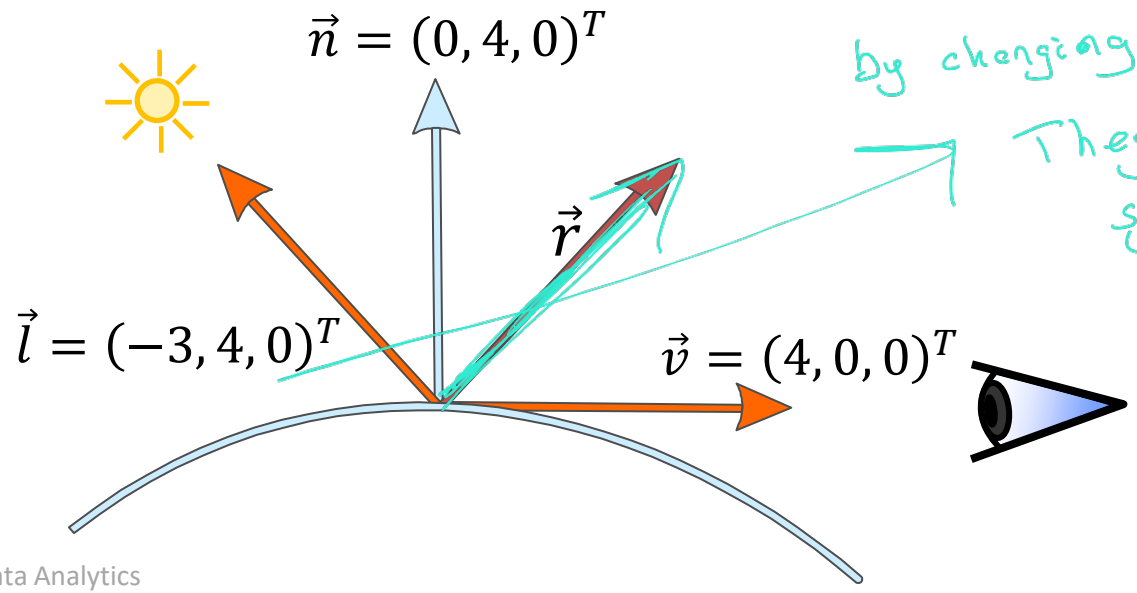
$\frac{5}{6} c_p O_d \left(\begin{pmatrix} 3 \\ 4 \\ 0 \end{pmatrix} \begin{pmatrix} 4 \\ 0 \\ 0 \end{pmatrix} \right)^2$

$5 \cdot 4$

$= \frac{3}{10} c_p O_d$

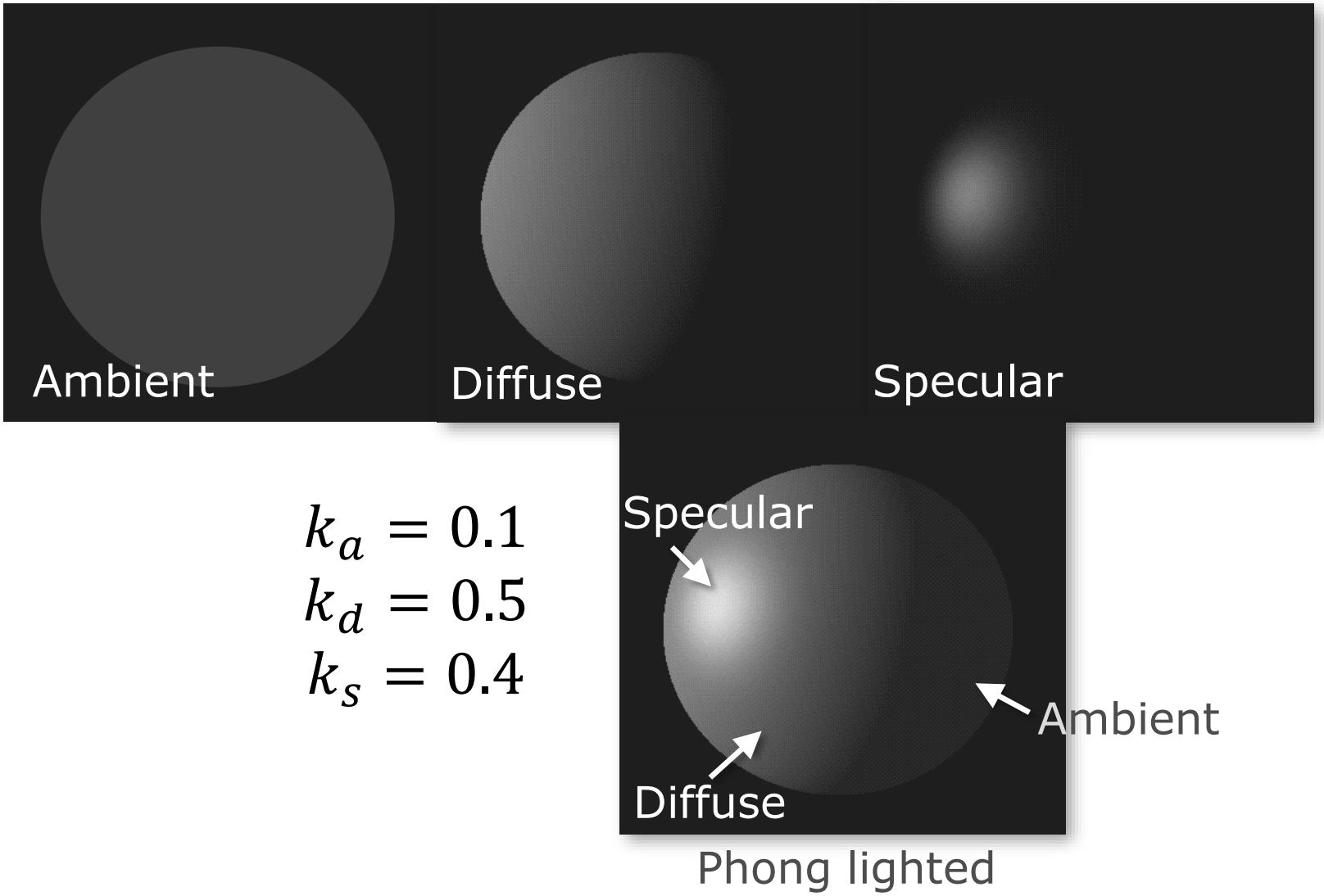
We took inverse because we calculate r by changing x value

They are symetric



Lighting

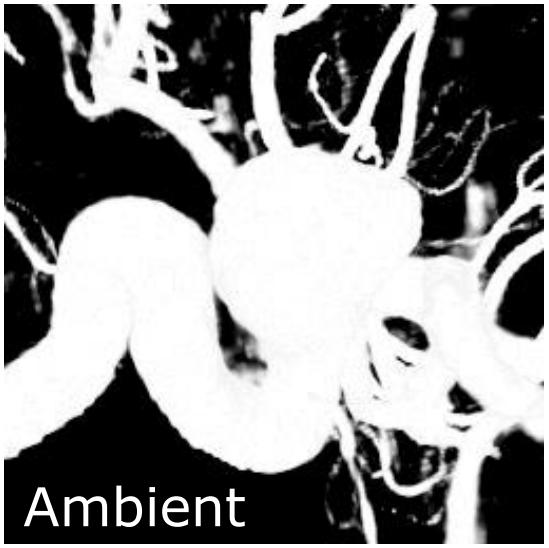
SIEMENS
Ingenuity for life

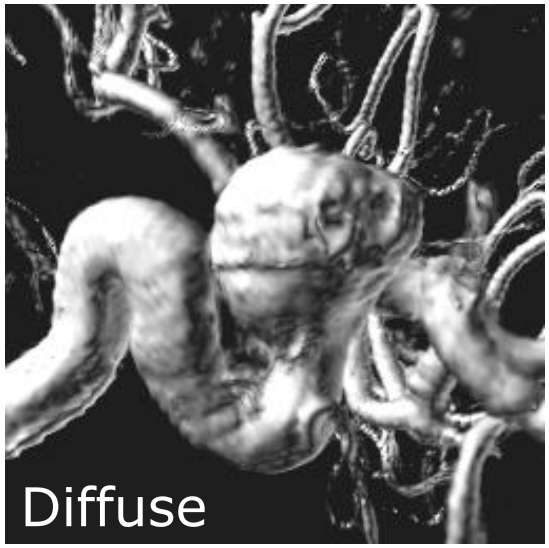


$$k_a = 0.1$$
$$k_d = 0.5$$
$$k_s = 0.4$$

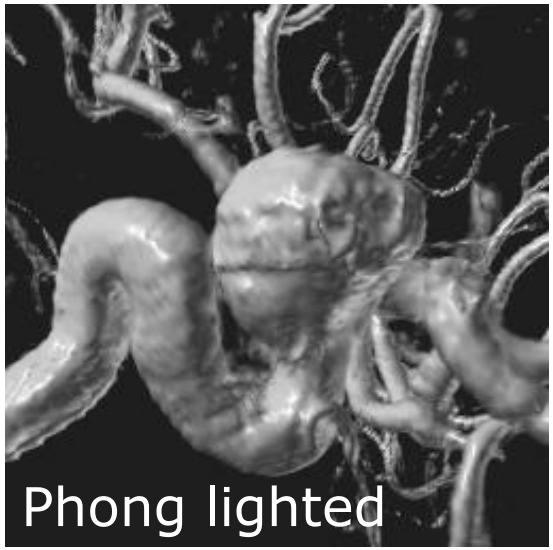
Lighting

SIEMENS
Ingenuity for life





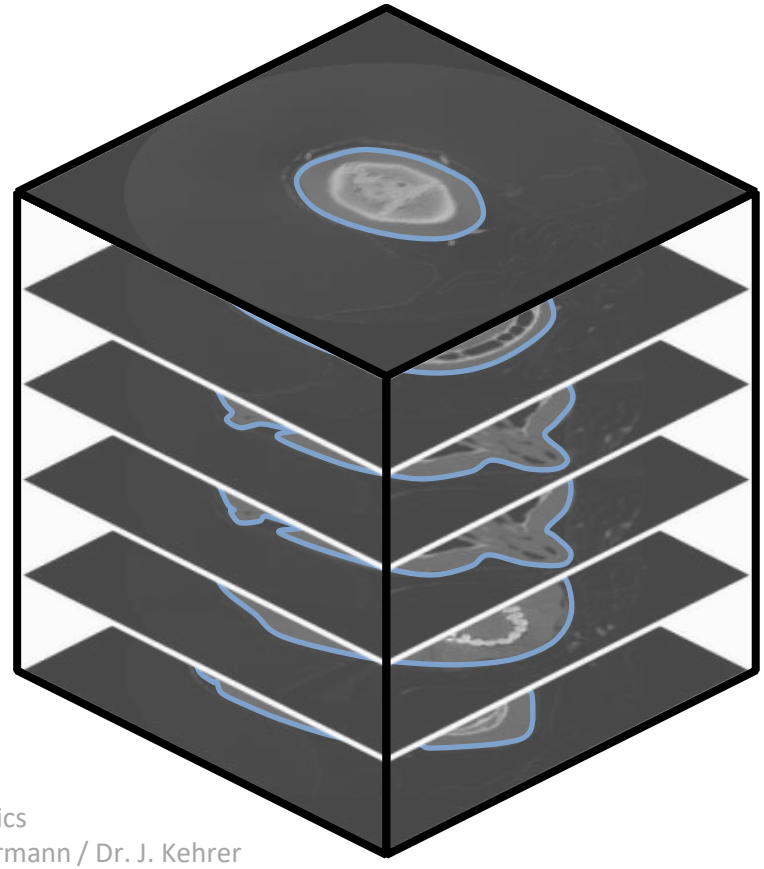
$$k_a = 0.1$$
$$k_d = 0.5$$
$$k_s = 0.4$$

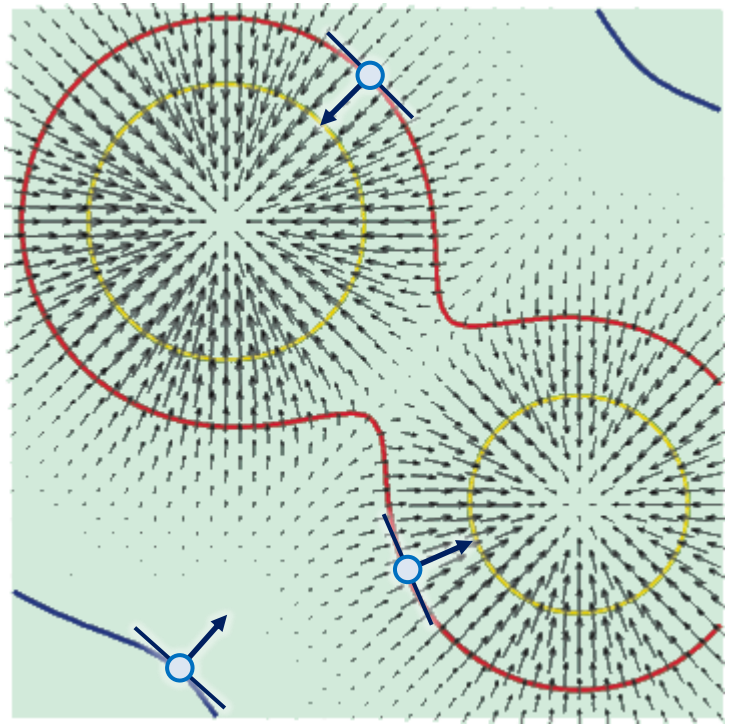


Lighting

SIEMENS
Ingenuity for life

- What is the normal vector at a point on an iso-surface in a scalar field?
- It is the gradient at this point, which is perpendicular to the iso-surface





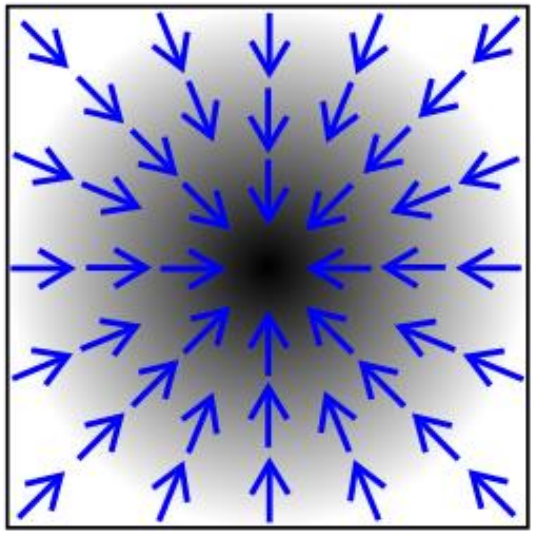
Lighting

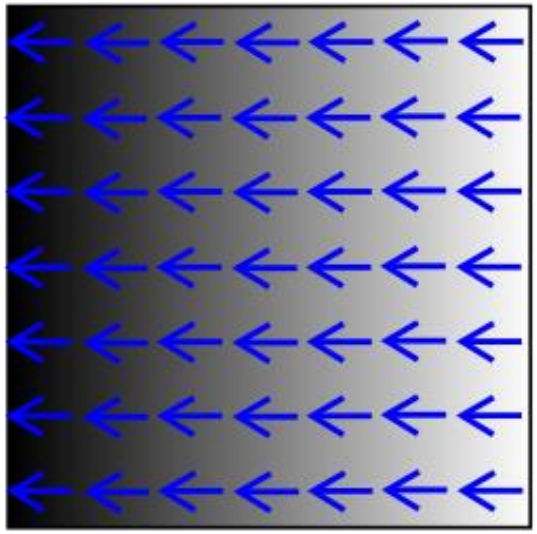
SIEMENS
Ingenuity for life

- The gradient in a scalar field $f(\boldsymbol{x})$

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

  - Partial derivatives of scalar field
  - The vector pointing into the direction of steepest ascent of f, with magnitude indicating the slope of the ascent
  - ∇ is the Nabla operator



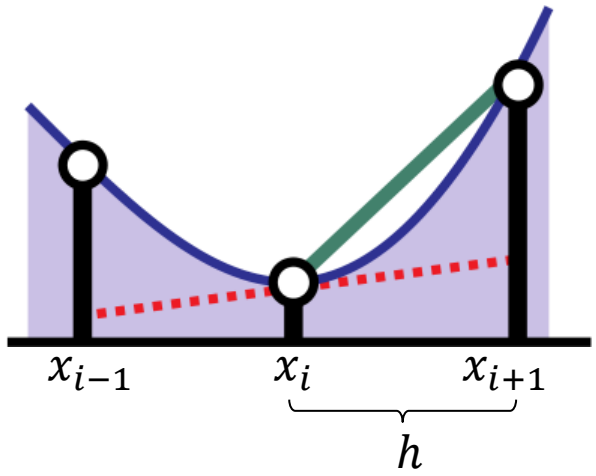


Lighting

SIEMENS
Ingenuity for life

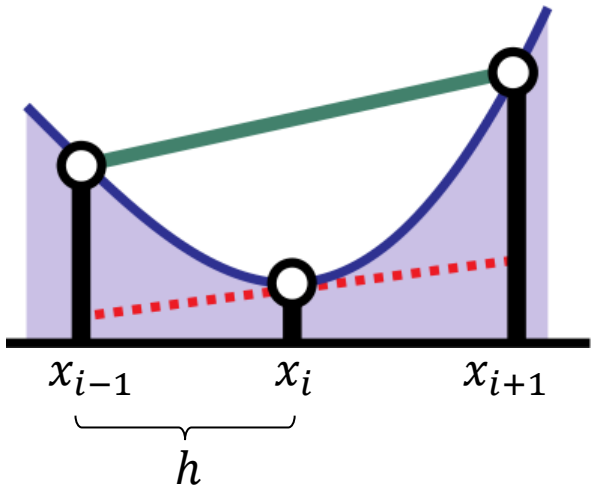
- Gradient approximation (1D)
  - Using forward differences

$$\frac{df}{dx}(x_i) \approx \frac{f(x_{i+1}) - f(x_i)}{h}$$



  - Using central differences

$$\frac{df}{dx}(x_i) \approx \frac{f(x_{i+1}) - f(x_{i-1})}{2h}$$



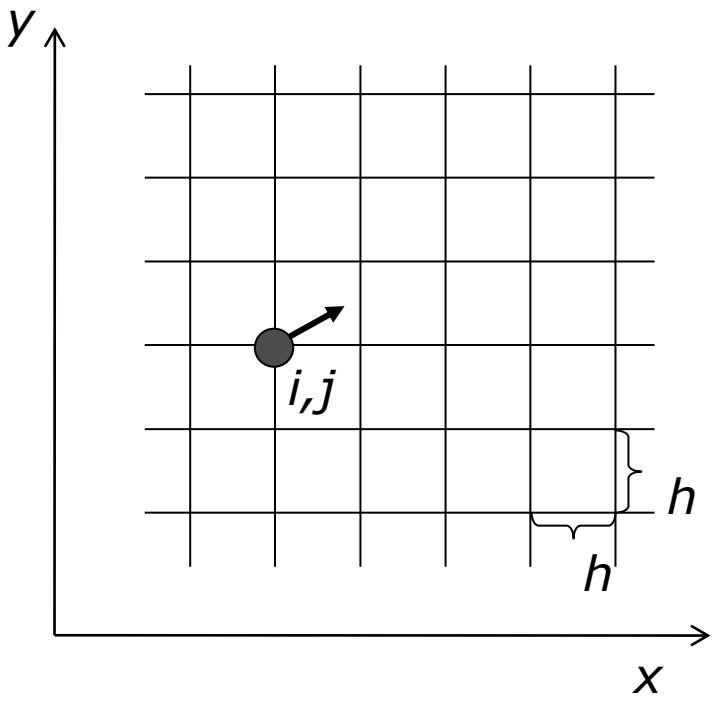
[Engel et al. 06]

Lighting

SIEMENS
Ingenuity for life

- Approximation of partial derivatives on uniform grids
  - Using forward differences

$$grad\ f = \nabla f = \begin{pmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \\ \frac{\partial f}{\partial z} \end{pmatrix} \approx \begin{pmatrix} \frac{f_{i+1,j,k} - f_{i,j,k}}{h} \\ \frac{f_{i,j+1,k} - f_{i,j,k}}{h} \\ \frac{f_{i,j,k+1} - f_{i,j,k}}{h} \end{pmatrix}$$



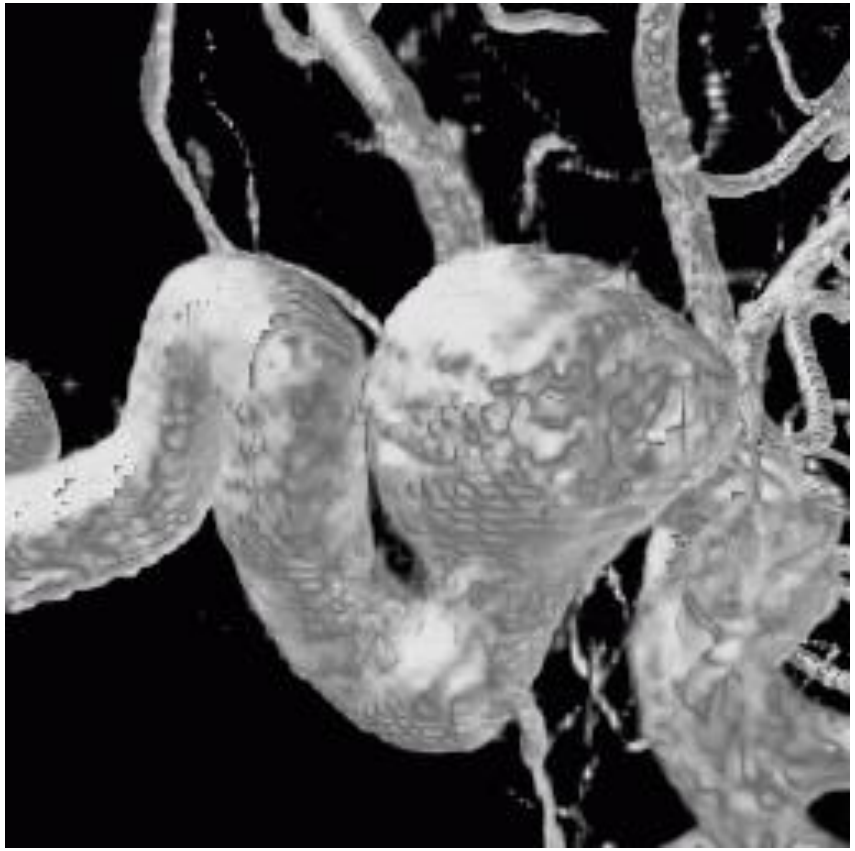
  - Using central differences

$$grad\ f = \nabla f = \begin{pmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \\ \frac{\partial f}{\partial z} \end{pmatrix} \approx \begin{pmatrix} \frac{f_{i+1,j,k} - f_{i-1,j,k}}{2h} \\ \frac{f_{i,j+1,k} - f_{i,j-1,k}}{2h} \\ \frac{f_{i,j,k+1} - f_{i,j,k-1}}{2h} \end{pmatrix}$$

Lighting

SIEMENS
Ingenuity for life

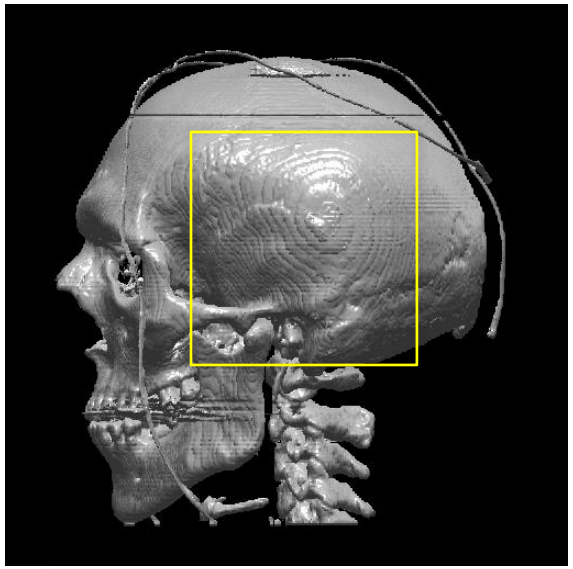
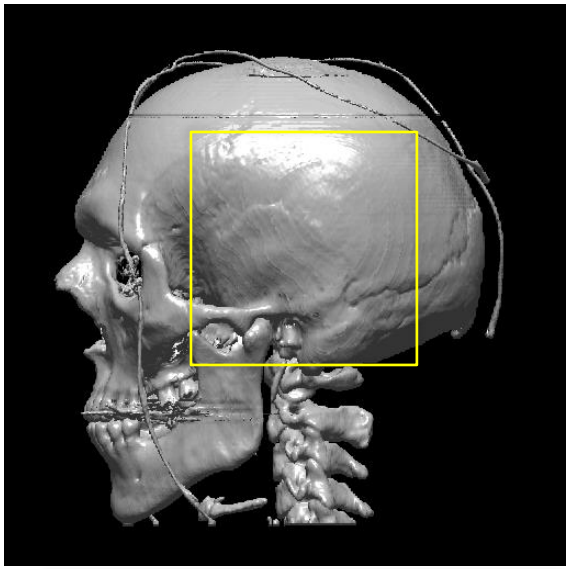
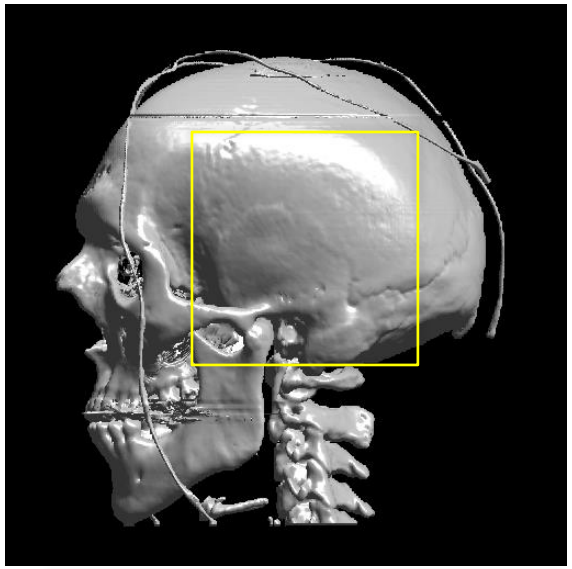
Central differences

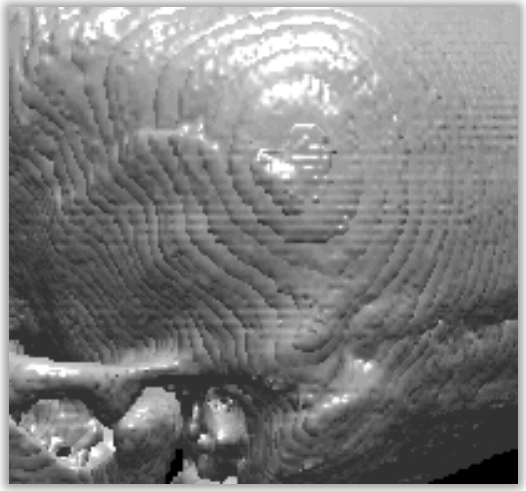


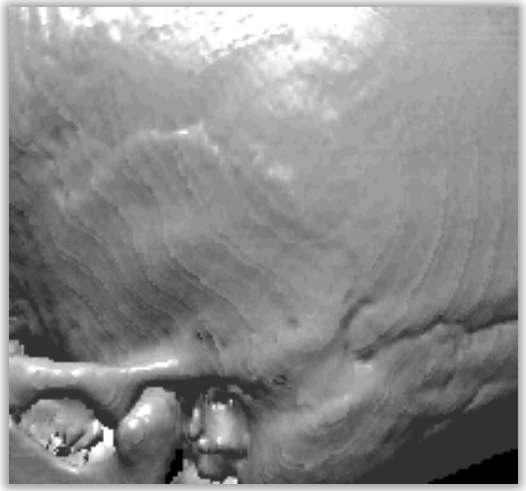
Forward differences

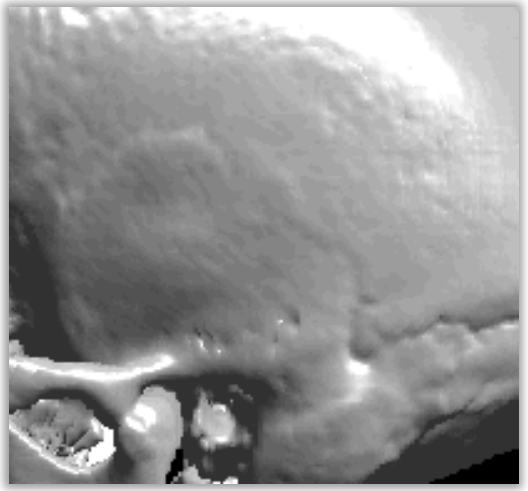
Lighting

SIEMENS
Ingenuity for life

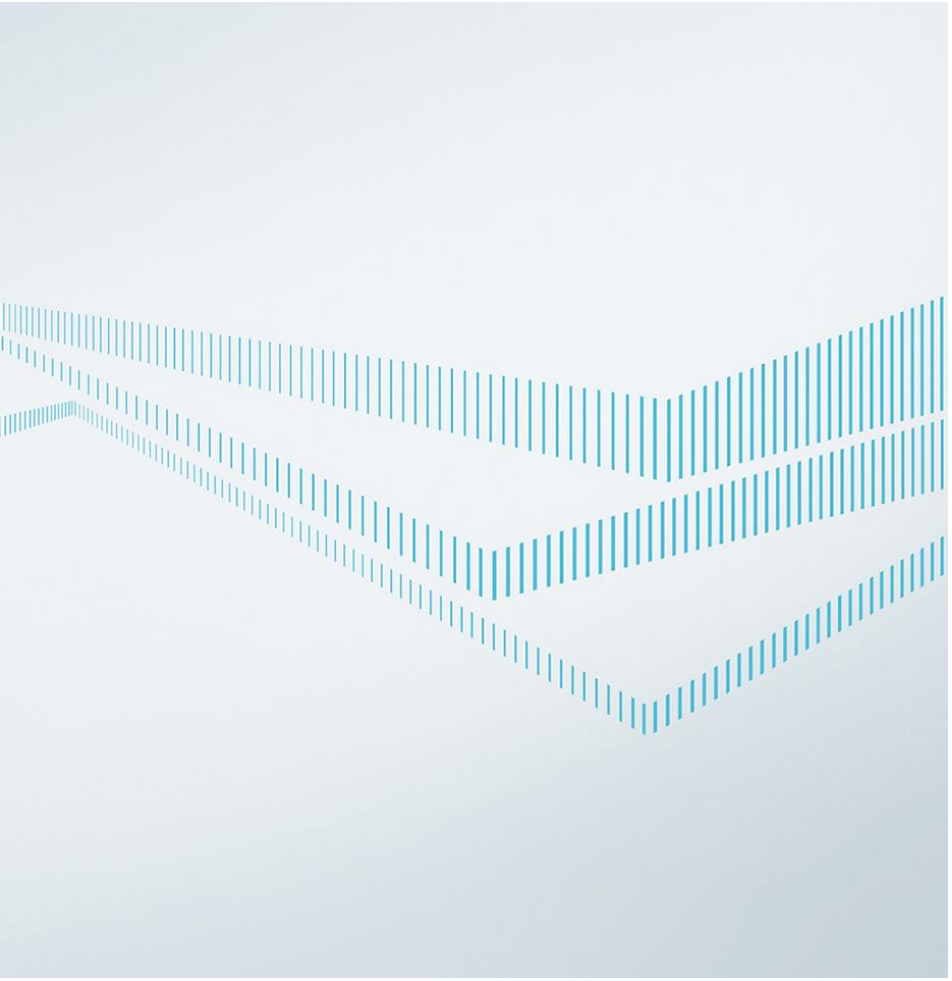

Forward differences


Central differences


Sobel operator

Contact information

SIEMENS
Ingenuity for life



Dr. Johannes Kehrer

Siemens Technology
T DAI HCA-DE
Otto-Hahn-Ring 6
81739 München, Deutschland

E-mail:
kehrer.johannes@siemens.com

Internet
siemens.com/innovation

Intranet
intranet.ct.siemens.com