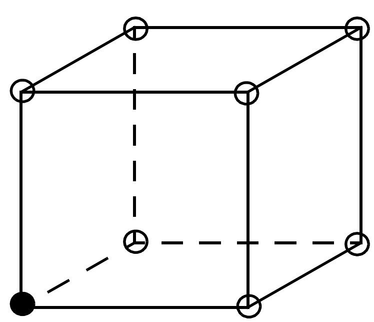
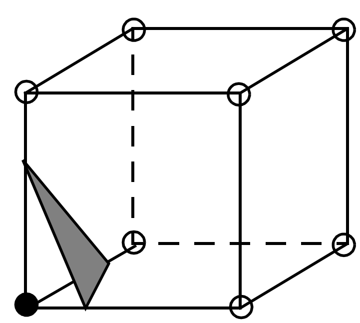
The Marching Cubes Polygonization Algorithm

- The *Marching Cubes (MC)* algorithm converts a volume into a polygonal model
  - this model *approximates* a chosen iso-surface by a mesh of polygons
  - the polygonal model can then be rendered, for example, using a fast z-buffer algorithm
  - if another iso-surface is desired, then MC has to be run again
- Steps:
  - imagine all voxels above the iso-value are set to 1, all others are set to 0
  - the goal is to find a polygonal surface that includes all 1-voxels and excludes all 0-voxels
  - look at one volume cell (a cube) at a time → hence the term *Marching Cubes*
  - here are 2 of 256 possible configurations:

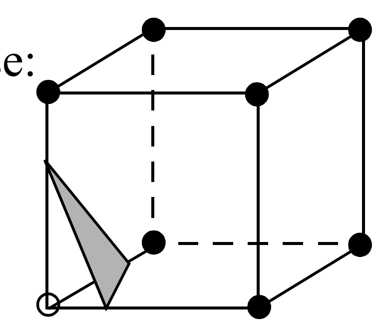


only 1 voxel > iso-value



the polygon that separates inside from outside

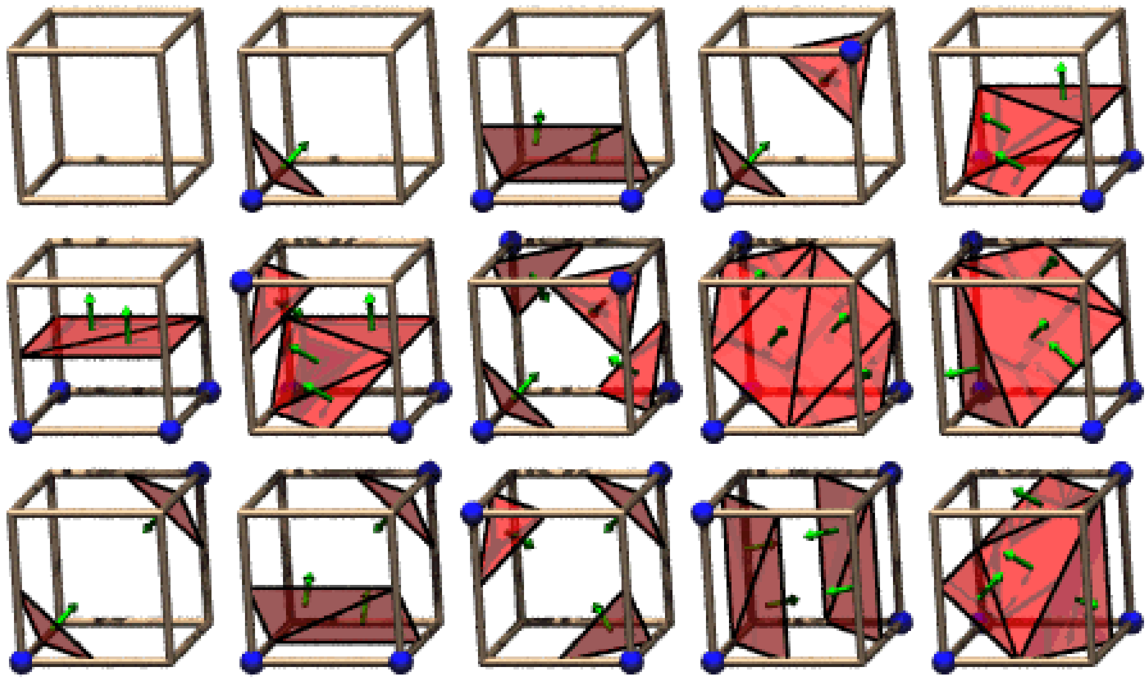
the reverse case:



7 voxels > iso-value
the same polygon results

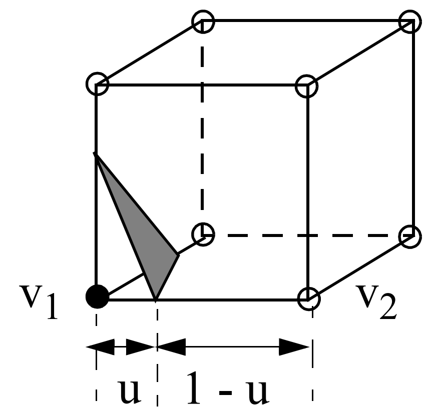
Marching Cubes (2)

- One can identify 15 base cases
  - Use symmetry and reverses to get the other 241 cases


The 15 Cube Combinations

- The exact position of the polygon vertex on a cube edge is found by linear interpolation:

$$iso = v_1 \cdot (1-u) + v_2 \cdot u \longrightarrow u = \frac{v_1 - iso}{v_1 - v_2}$$

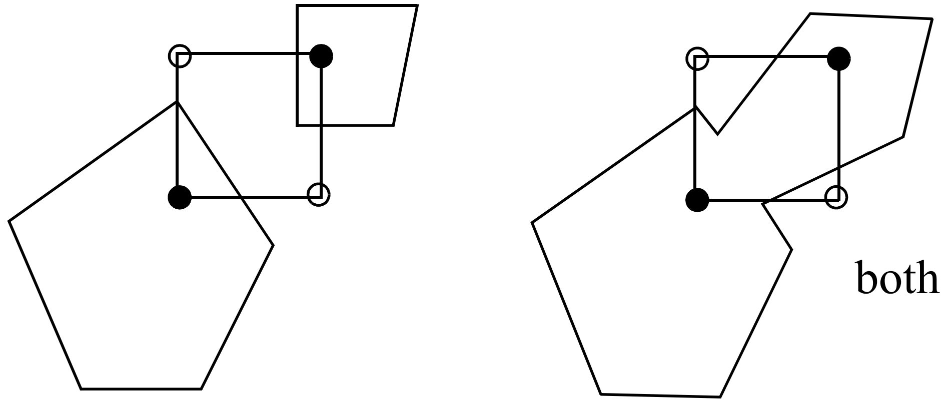


- Now interpolate the vertex color by: $c_1 = uc_2 + (1-u)c_1$
- Interpolate the vertex normal by: $\boldsymbol{n}_1 = u\boldsymbol{g}_{\mathbf{2}} + (1-u)\boldsymbol{g}_{\mathbf{1}}$

(the **g1** and **g2** are the gradient vectors at v1 and v2 obtained by central differencing)

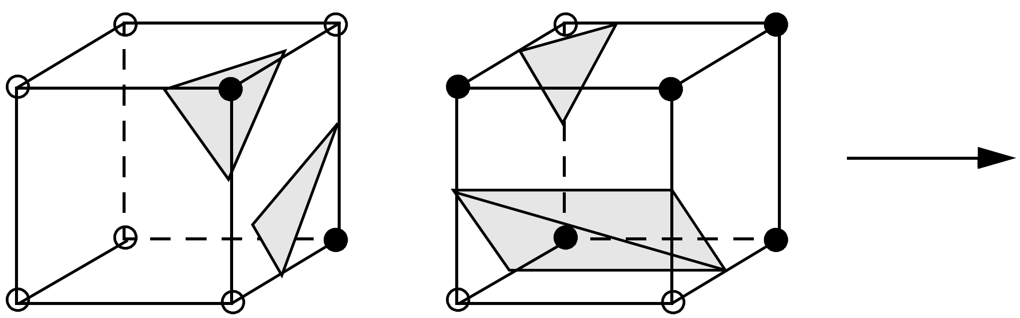
Ambiguous Cases

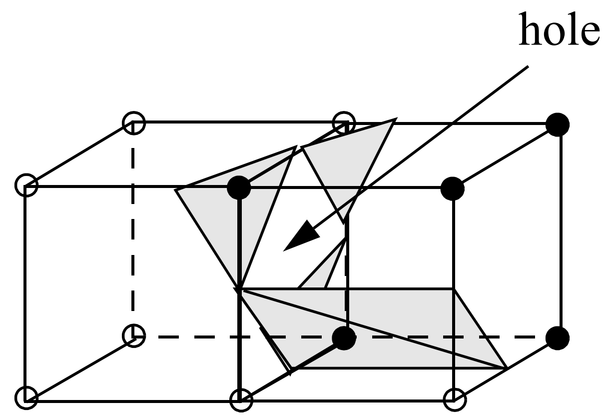
2D: ambiguous case



both versions are plausible

3D: what happens when cases are arbitrarily chosen

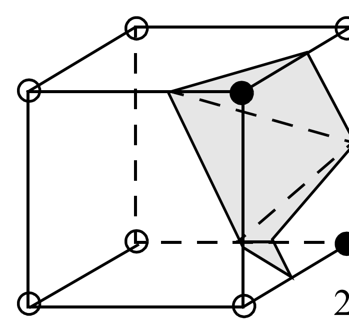




case 3 case 6 (complementary) connected

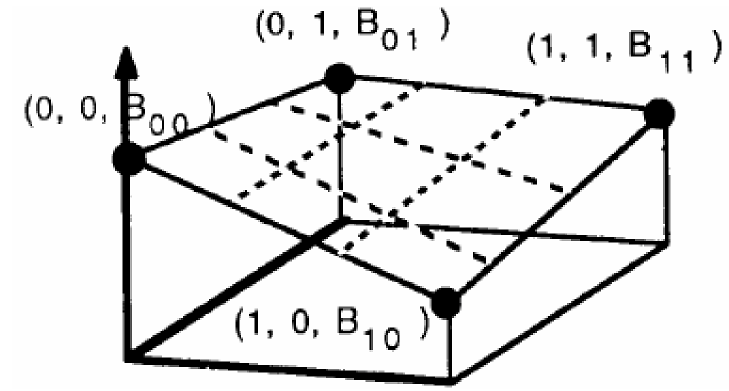
Remedy: add 6 alternative cases for 3, 6, 7, 10, 12, 13 to prevent holes

Example: case 3c

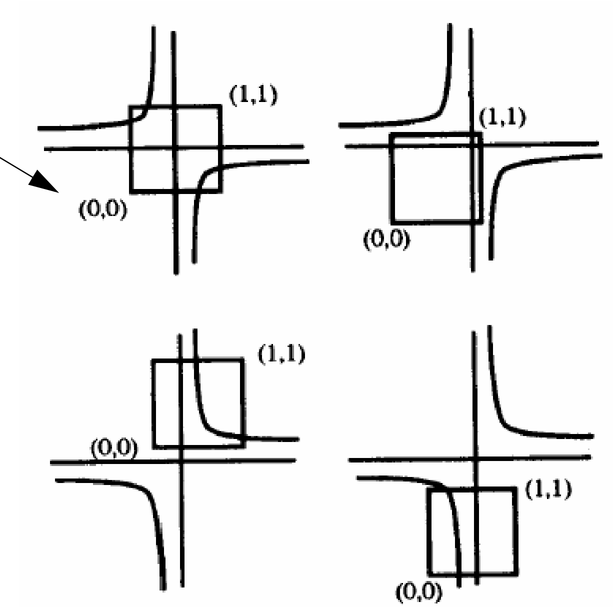


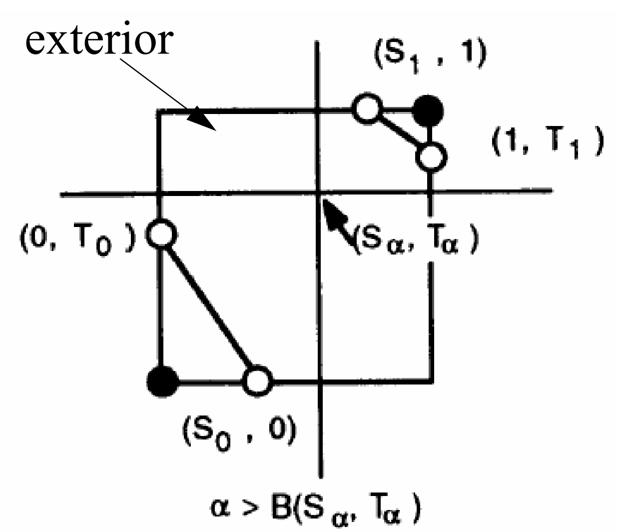
Remove Ambiguities Using the *Asymptotic Decider* Method

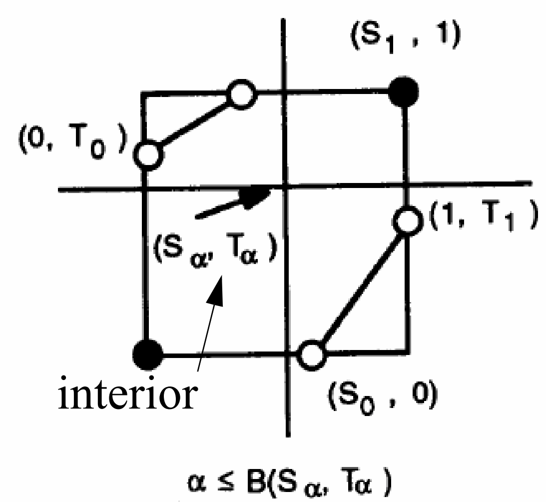
- Explain in 2D:
  - surface created by bilinear interpolation



  - function $(1-s, s)\begin{bmatrix} B_{00} & B_{01} \\ B_{10} & B_{11} \end{bmatrix}(1-t, t)$
  - gives rise to two hyperbolas $B(s,t) = \alpha$ (isovalue)
  - ambiguity: both hyperbolas intersect domain (0,0), (1,1)
  - resolve ambiguity by comparing $B(S_\alpha, T_\alpha)$ with α







$$S_\alpha = \frac{B_{00} - B_{01}}{B_{00} + B_{11} - B_{01} - B_{10}}$$

$$T_\alpha = \frac{B_{00} - B_{10}}{B_{00} + B_{11} - B_{01} - B_{10}}$$

  - similar cases in 3D