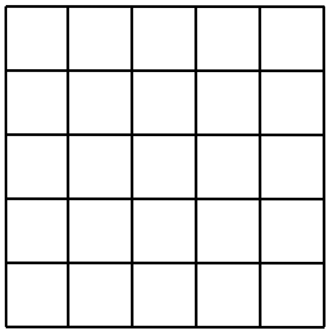
Unstructured Volume Rendering

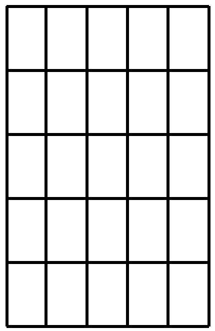
Ref) www.cse.ohio-state.edu/~hwshen/788/PT.ppt

Grid Types

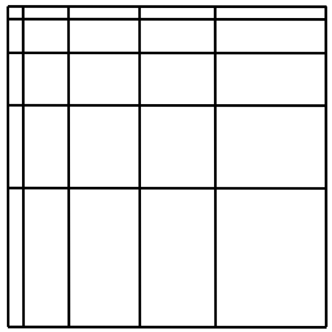
- Structured Grids



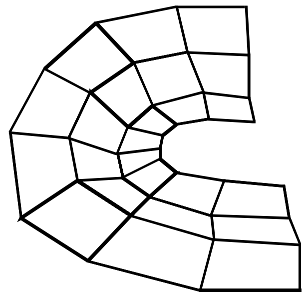
uniform



regular

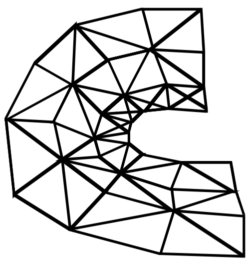


rectilinear

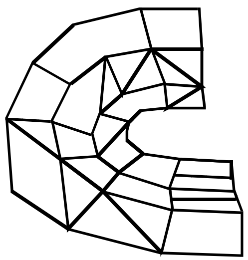


curvilinear

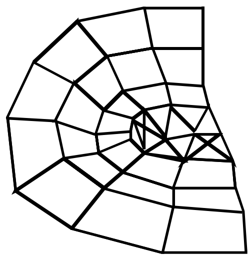
- Unstructured Grids



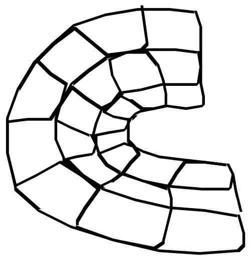
regular



irregular



hybrid



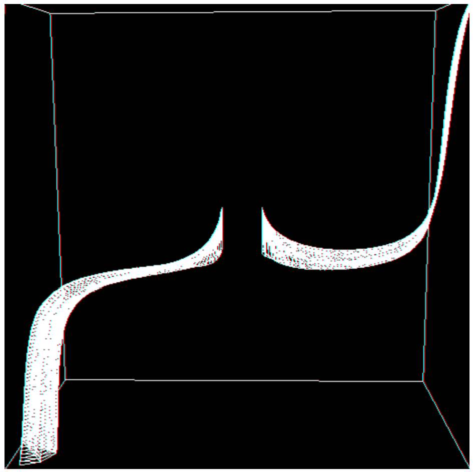
curved

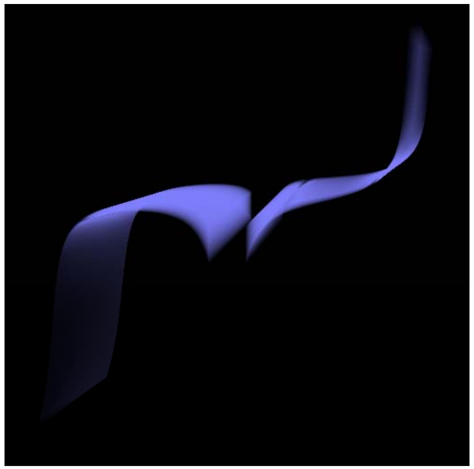
Unstructured Volume Rendering

- Given an irregular data set that consists of volumetric cells (typically from FEM simulation)
- How can the volume be displayed accurately?
- Numerous approaches:
  - Ray casting
  - Ray tracing
  - Projected Tetrahedra (PT) algorithm

tornado

1374 Tetrahedra



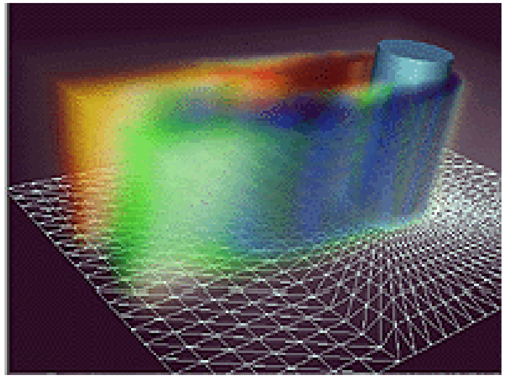


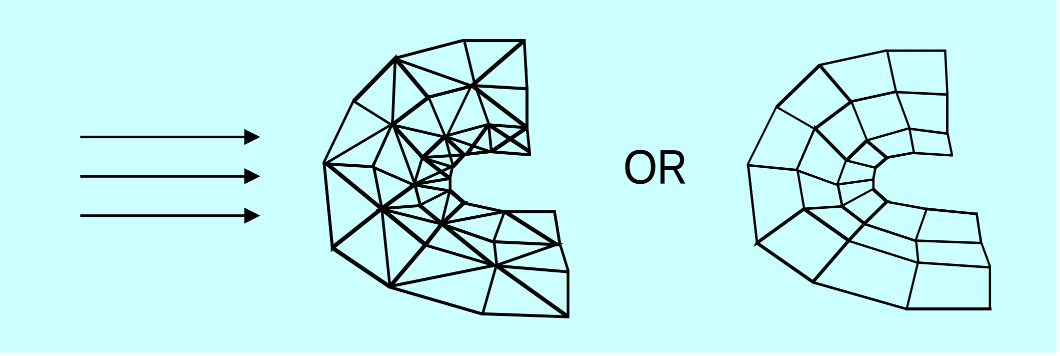
Ray-Casting Approach

- Garrity 1990
- Need to figure out, where the ray is going
- Need to figure out how to integrate along that ray
- Assume arbitrary polyhedra (as opposed to just tetrahedra as many other algorithm)
- Image order

Ray-Casting

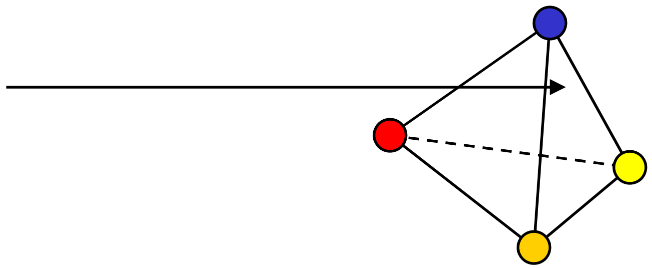
- For each ray
  - find intersection with first cell
  - interpolate a color, opacity value
  - while not outside the volume
    - find exit point of ray in cell
    - interpolate a color, opacity value
    - integrate opacity/color along ray within cell





Ray-Casting

- Computationally more expensive
  - Point location is much more complicated (adjacency information is needed)
  - Degeneracy in intersection test (hit vertex, edge etc)
  - Interpolation could be more expensive (cell type dependent)



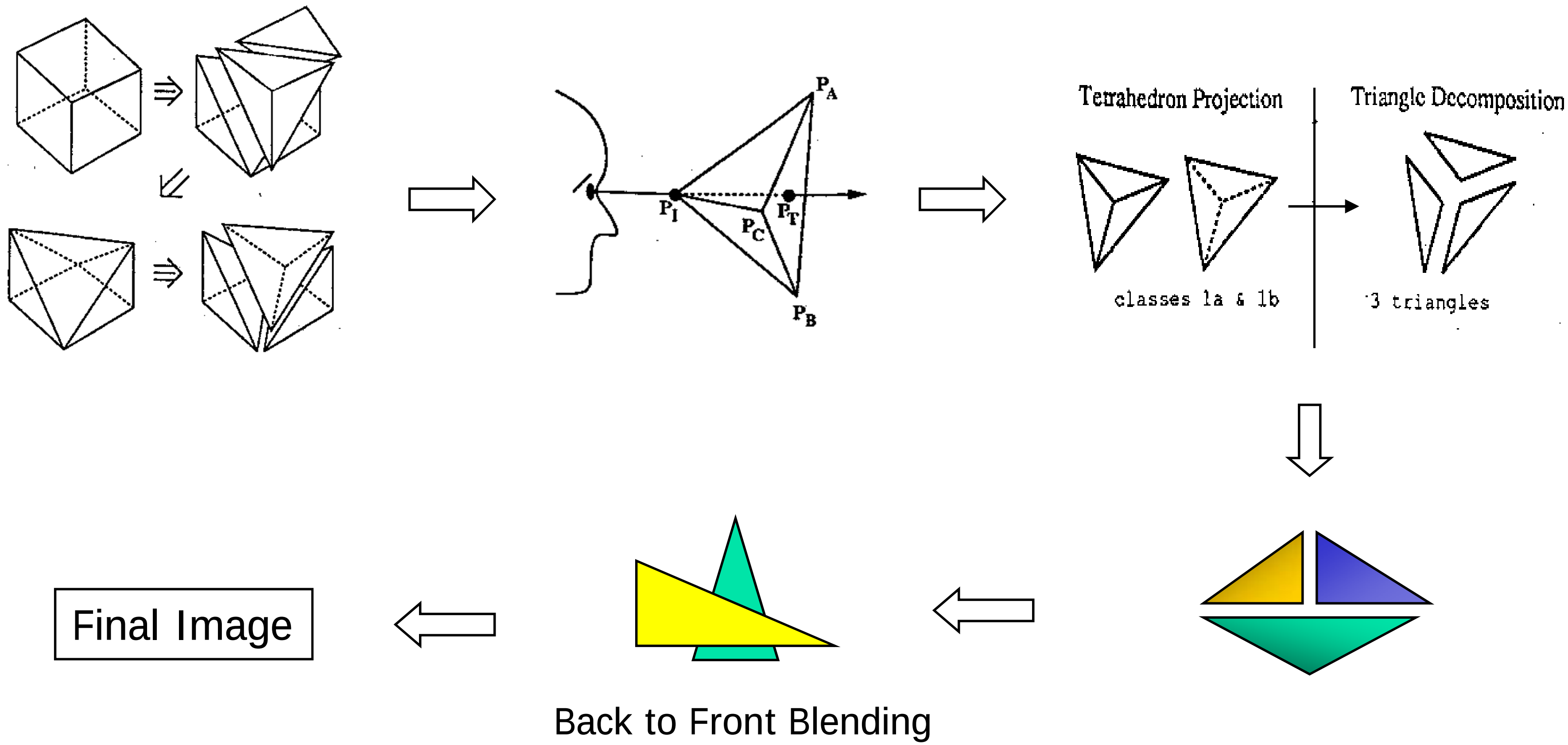
Ray-Casting - conclusion

- Bottleneck - size of image (# of pixels)
- results for 512^2 image:
  960 cells - 81.6 seconds
  1490 cells - 147.6 seconds
  16,896 cells - 149.2 seconds
  381,548 cells - 591.6 seconds
- may miss small polyhedra!!

Projected Tetrahedra (PT)

- Shirley and Tuchman 1990
- Max, Hanrahan, Crawfis 1990
- Object order (cell by cell)
- Utilize graphics hardware
- Makes sense if one cell projects to many pixels based on tetrahedral decomposition

PT – Basic Idea

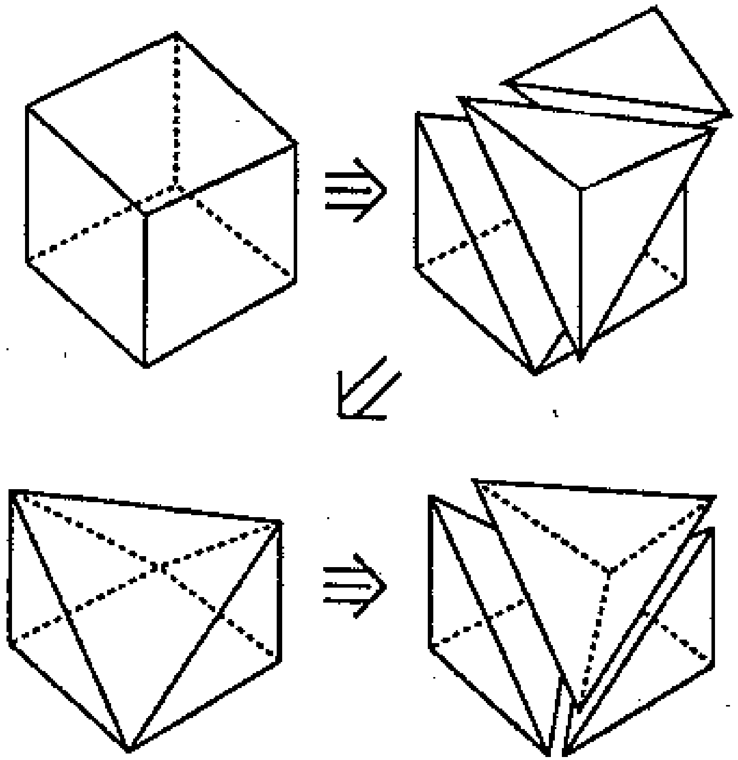


PT - Basic algorithm

1. Decompose volume into tetrahedral cells
2. Classify each tetrahedron according to its projected profile relative to a viewpoint
3. Find the positions of the tetrahedra vertices after the perspective transformation
4. Decompose projections of the tetrahedra into triangles
5. Find color and opacity values for the triangle vertices using ray integration in the original world coordinates
6. Scan convert the triangles on a graphics workstation

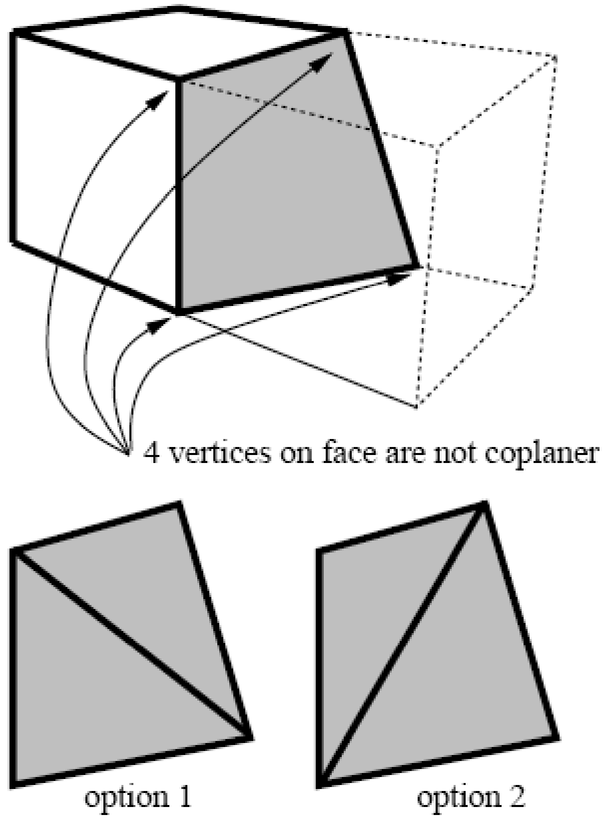
PT – Tetrahedra decomposition

- Decompose volume into tetrahedral cells
- Want least amount of computation



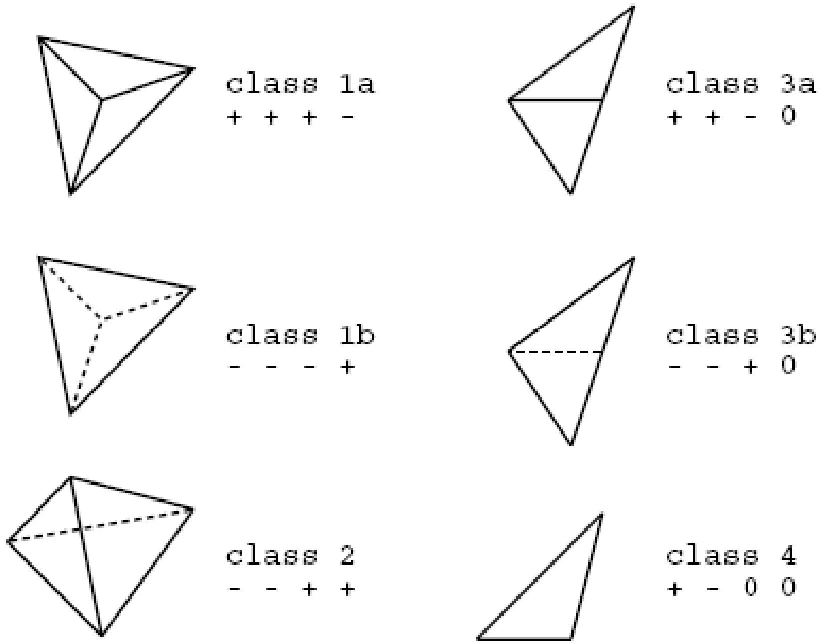
PT - Tetrahedra

- Need to make sure that there will be no “cracks”
- Make the same pair of triangles for adjacent cells



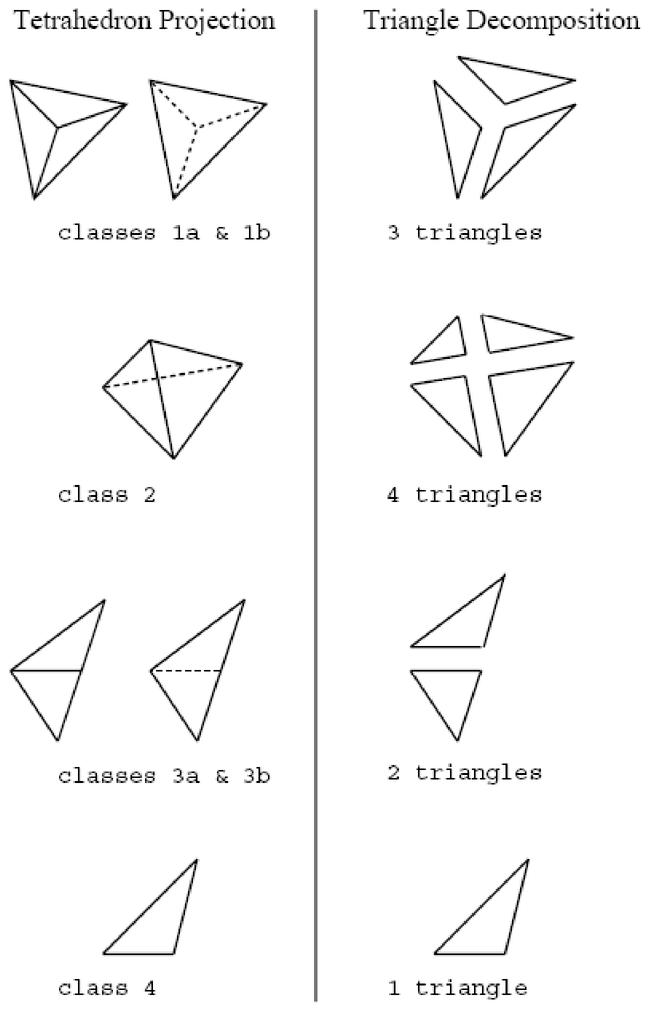
PT – Projection & Classification

- Classify each tetrahedron according to its projected profile relative to a viewpoint
- Based on the normal directions
  - Point toward (+)
  - Point away from (-)
  - Perpendicular to the eye (0)



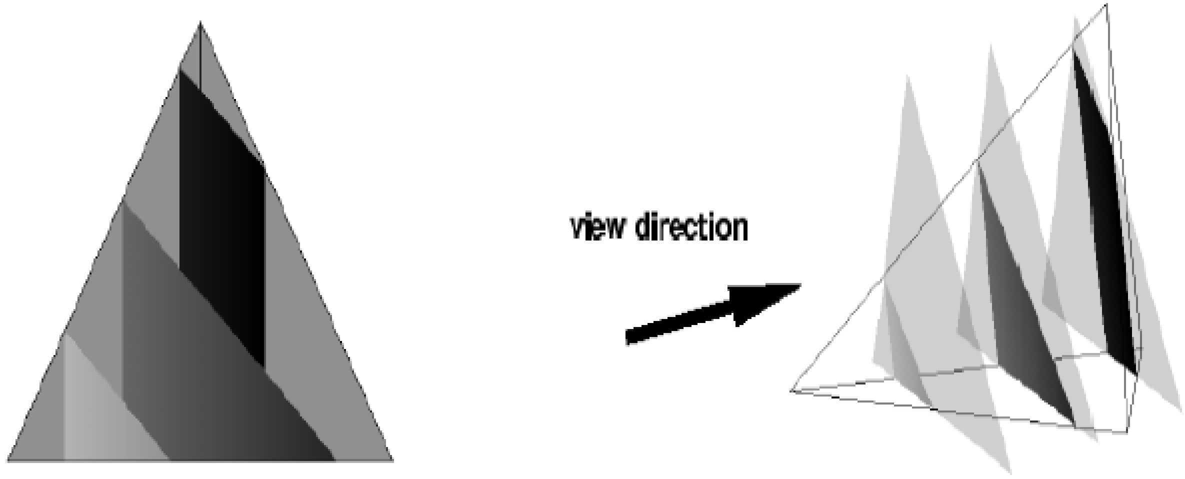
PT - Decompose

- Decompose projections of the tetrahedra into triangles
- Find the triangle vertex positions (tetrahedron vertex or edge intersection) after perspective projection



PT – Rendering

- Render the tetrahedra (decomposed triangles) back to front
- Need to calculate the color and opacity at each triangle vertex – *precomputed ray integration* is needed
- Graphics hardware for Gouraud shading



PT – Vertex Color/Opacity

- Zero opacity at "thin" vertices
- Data color at "thin" vertices
- Ray integration needed for "thick" vertices

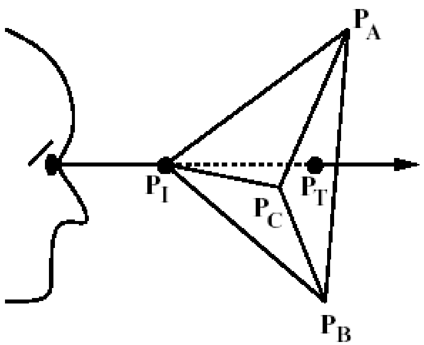
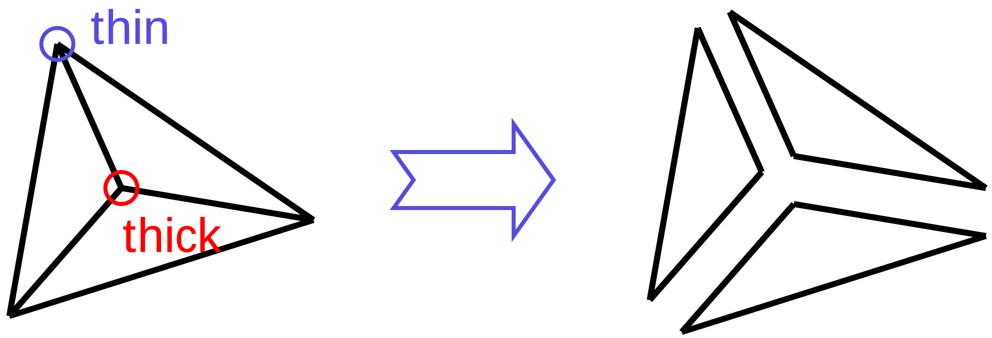
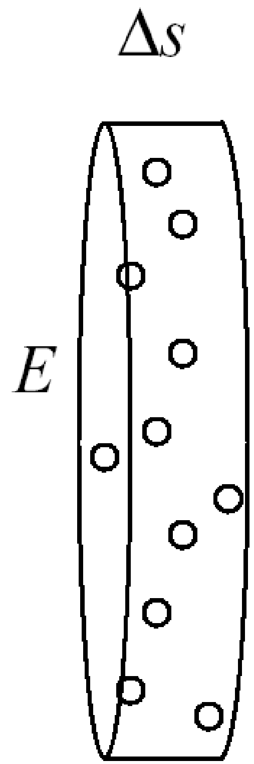


Figure 5: Example of Class 1 Decomposition.

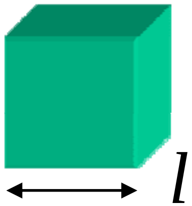


Optical model - Absorption Only

- Cold perfectly black particles (완전 불투명 입자)
  - Identical spheres, of radius r and projected area $A = \pi r^2$
  - ρ : the number of particles per unit volume
  - *# of particles :* $N = \rho E \Delta s$
  - *Absorption rate :* $NA/E = \rho A E \Delta s / E = \rho A \Delta s$
  - $\kappa = \rho A$ is called the *absorption coefficient* and defines the rate that light is occluded
    - In absorption and emission only model, extinction coefficient is approximated by absorption coefficient



Optical model – opacity



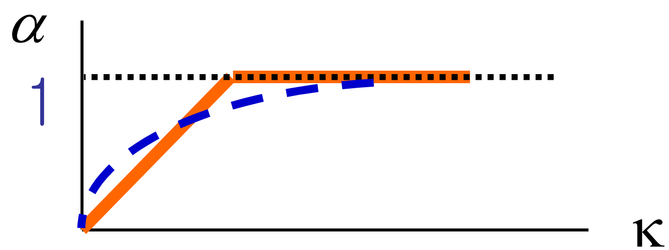
$$\alpha = 1 - \exp\left(-\int_0^l \kappa(t)\right)$$

When κ is constant,

$$\alpha = 1 - \exp(-\kappa l) = \kappa l - (\kappa l)^2/2 + \ldots\ldots$$

For small voxels, $\alpha = \min(1, \kappa l)$

When $l = 1$ and for small κ, $\alpha \approx \kappa$



For volume rendering, we use α instead of κ,

since α=f(s) is more intuitive than κ = f(s)

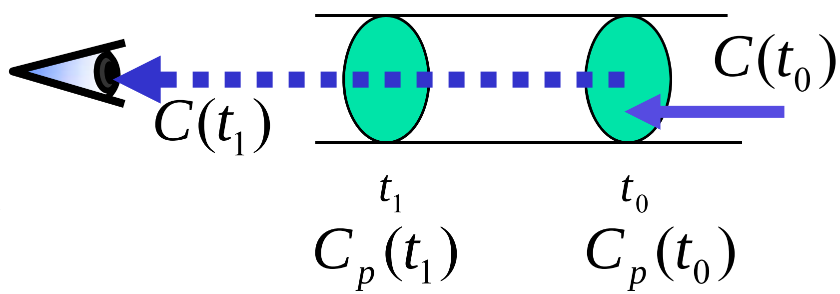
PT – Ray Integration

- Opacity of a thick vertex

$$\alpha \approx 1 - \exp\left(-A(t_1 - t_0)\frac{N_0 + N_1}{2} \right)$$

$$\approx A(t_1 - t_0)\frac{N_0 + N_1}{2}$$

N & color $\propto$ density



- Color through a thick vertex

$$C(t_1) = \alpha \frac{C_p(t_0) + C_p(t_1)}{2} + (1-\alpha) C(t_0)$$

Background color

Average of two vertex colors

Pros & Cons

- Pros
  - Hardware renderable triangles
  - Good for low density and smoothly changing data - fluid flows, stress analysis
- Cons
  - Visibility error
  - Numerical integration can take several minutes → interactive updates not possible yet
  - Not proper for dense data – large approximation error
  - Not good for large data sets because of the overhead of generating triangles.