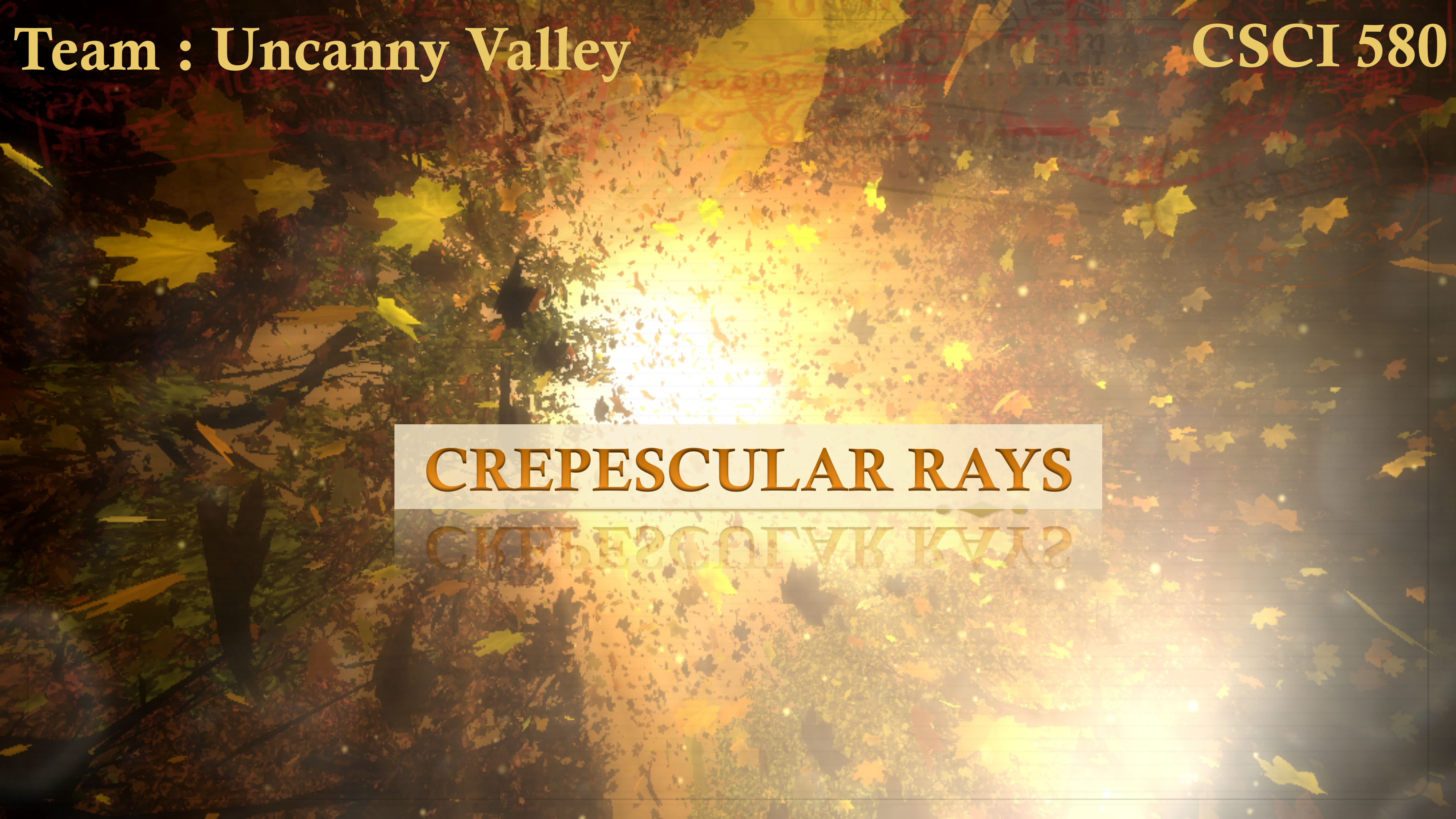


Team : Uncanny Valley

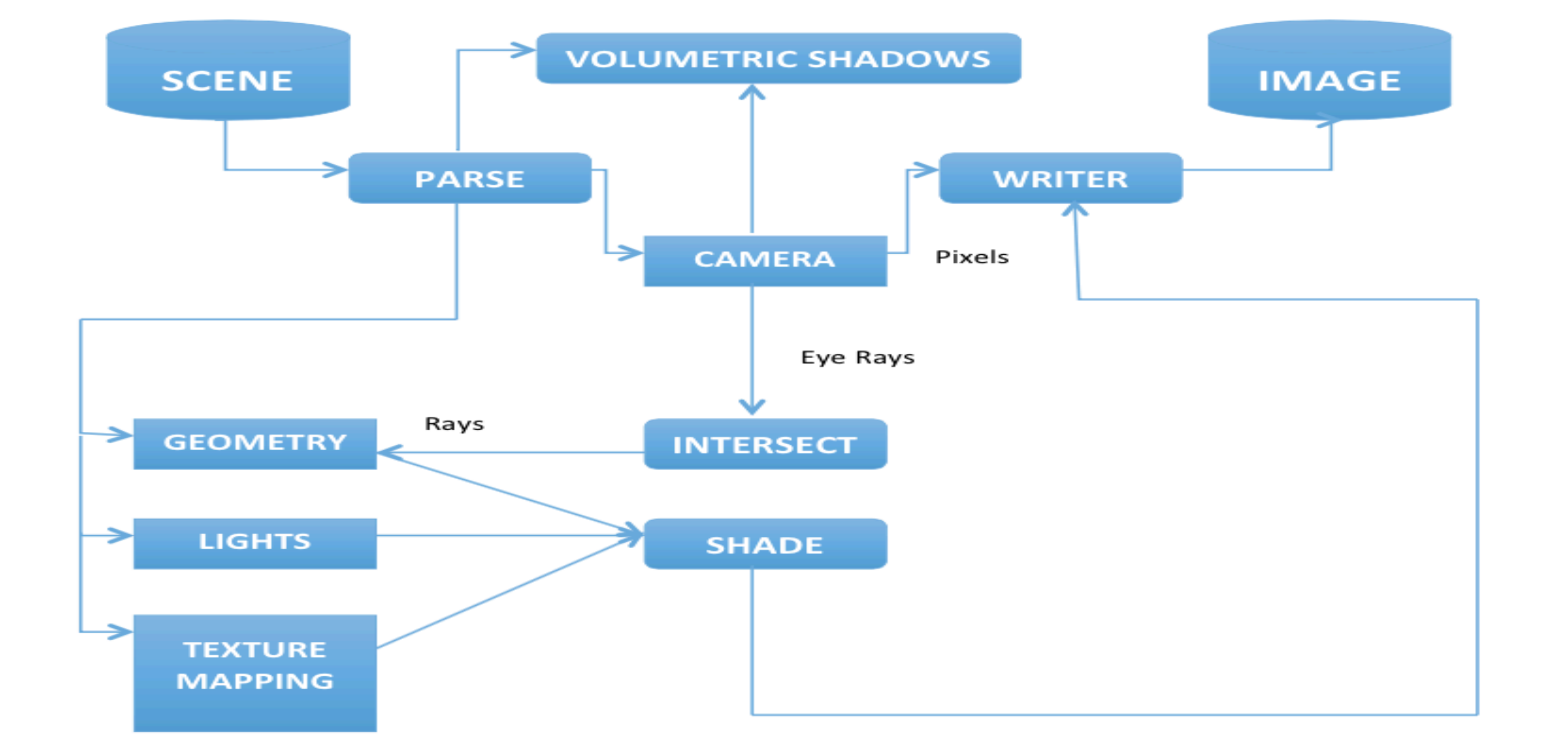
CSCI 580

CREPESCULAR RAYS

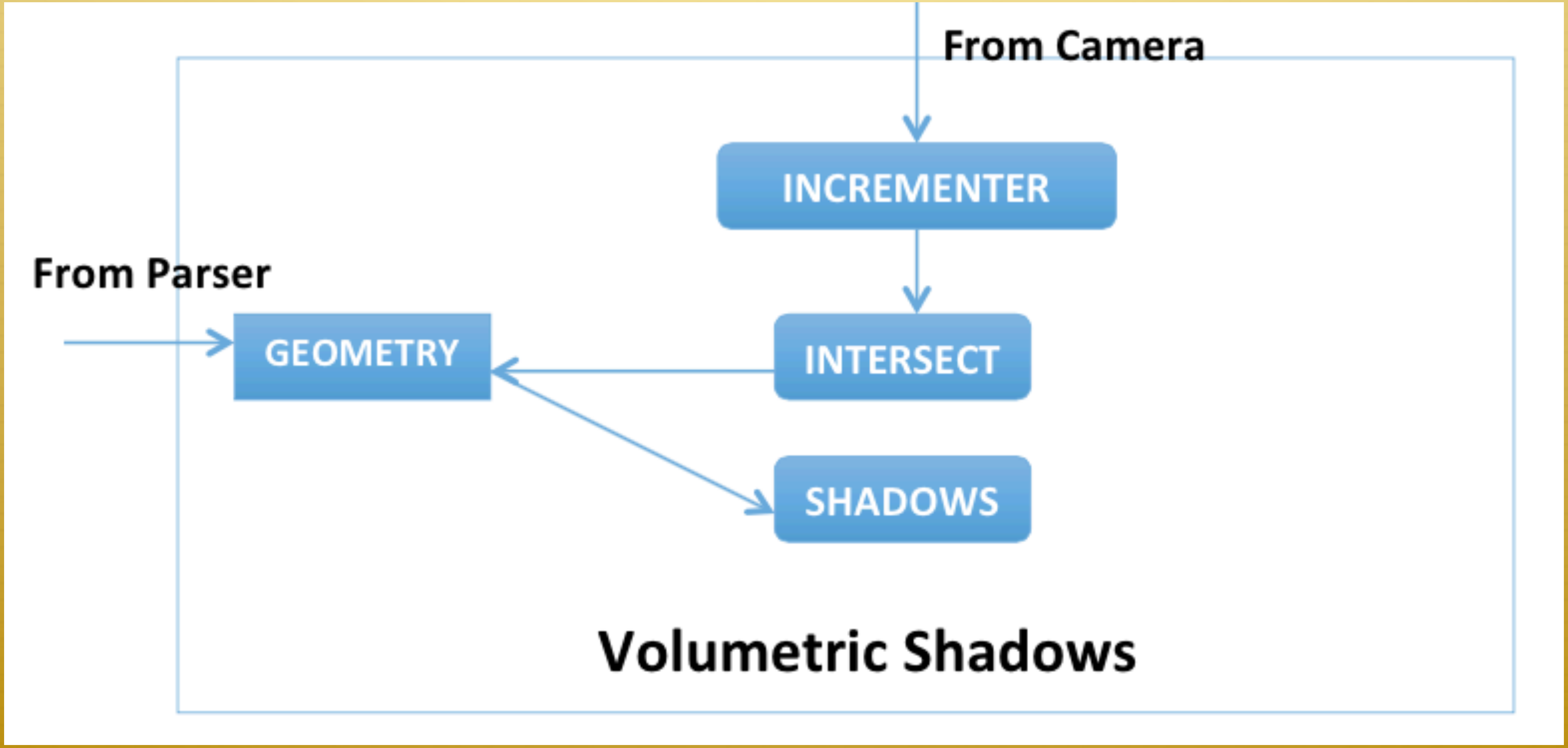
Brief about the title of our project

- Crepuscular rays are sun rays radiating from a point in the sky where the sun is located.
- Crepuscular as they occur during the crepuscular hours i.e. dawn and dusk.
- Inspite of its converging appearance they are merely parallel and apparently appear to converge owing to perspective effect.
- Ray Tracing for leaves, shadow rays for volumetric shadow calculation, procedural texture (Julia Set) for sun, LEE's scan line algorithm for tree.

Ray Tracing System Architecture



Volumetric Shadow block



From Camera

INCREMENTER

From Parser

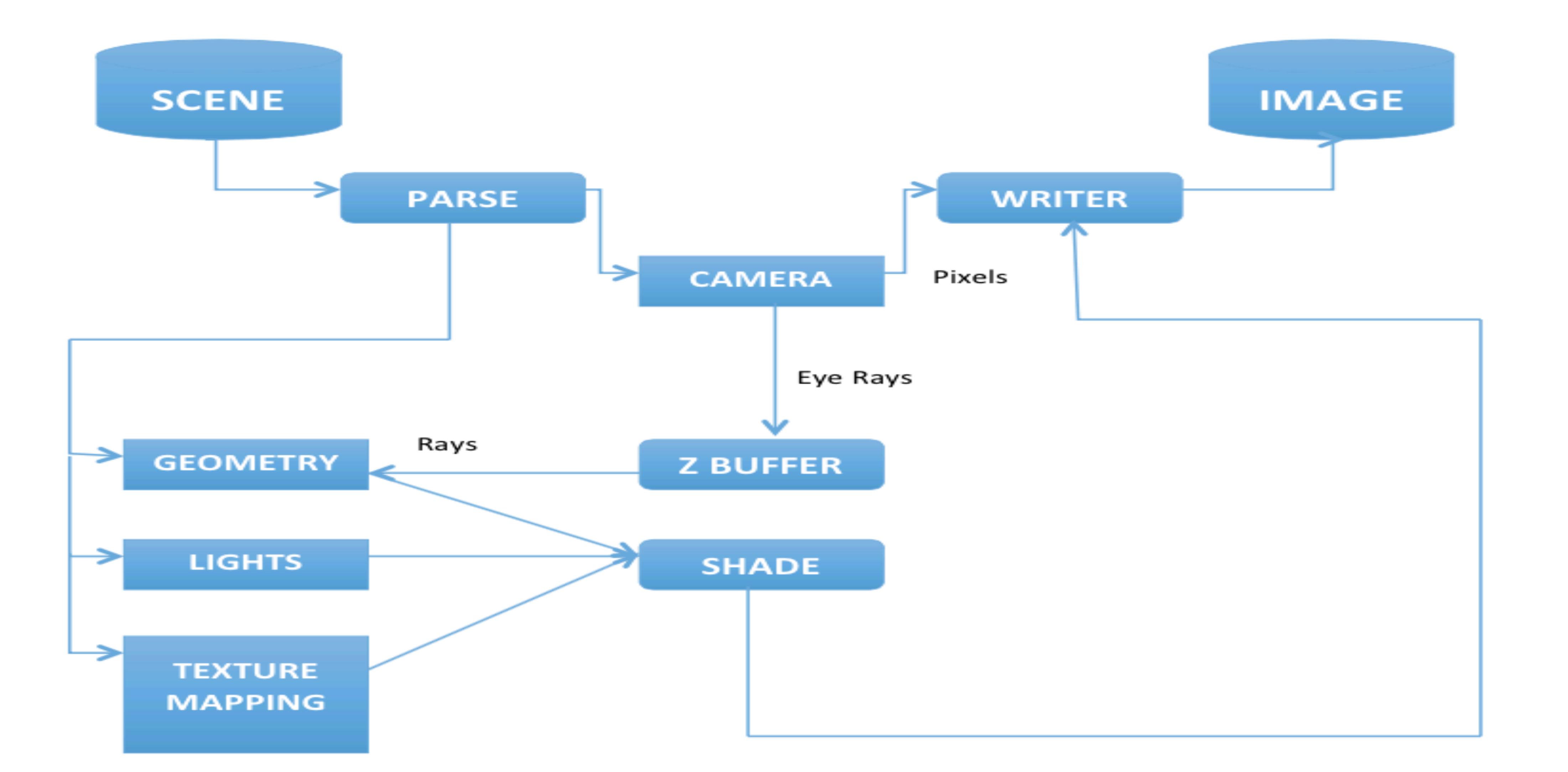
GEOMETRY

INTERSECT

SHADOWS

Volumetric Shadows

Scan Line System Architecture



Ray Tracing

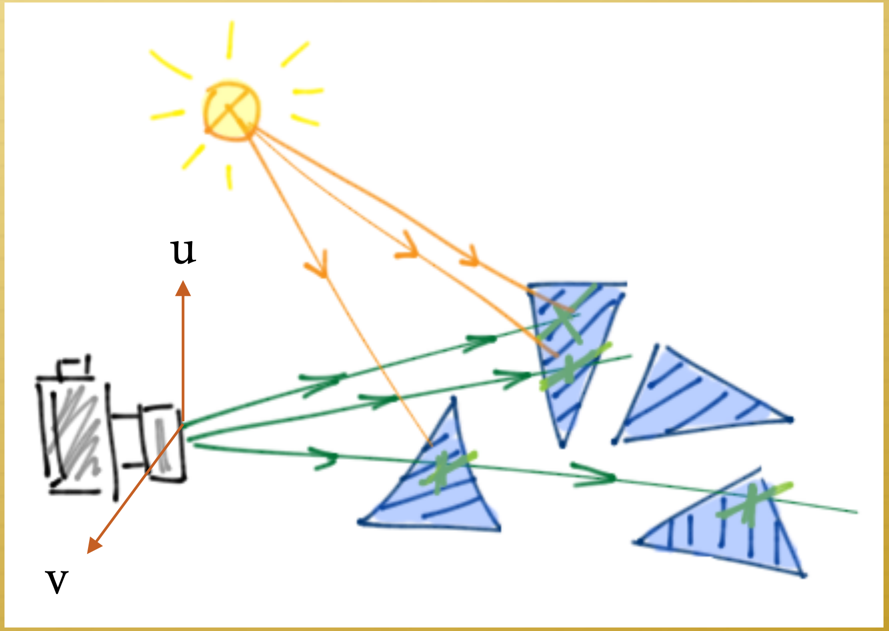
Ray Tracing Implementation – Eye to Object

For every pixel value

- Map the pixel value to the vector space

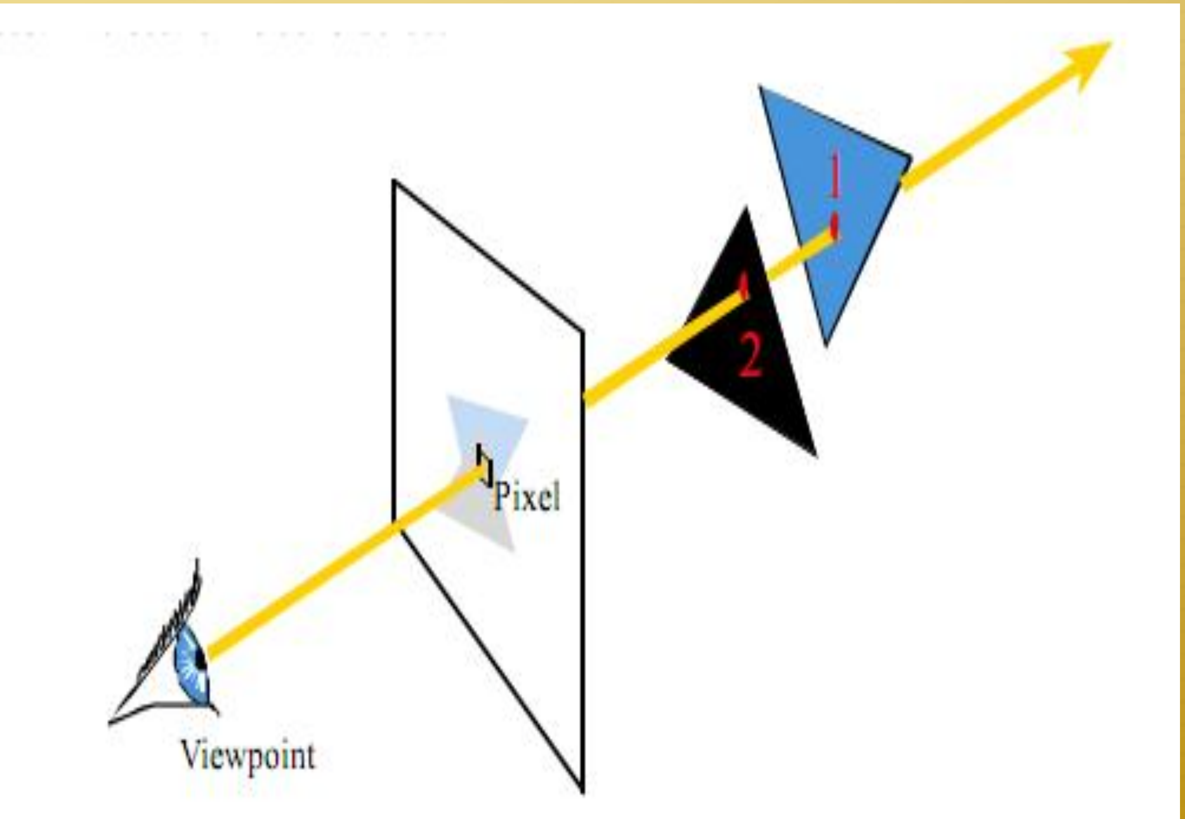
- Calculate the ray’s position and direction

- Calculate the point on the ray using the rays parametric equation



Ray Tracing cont’d

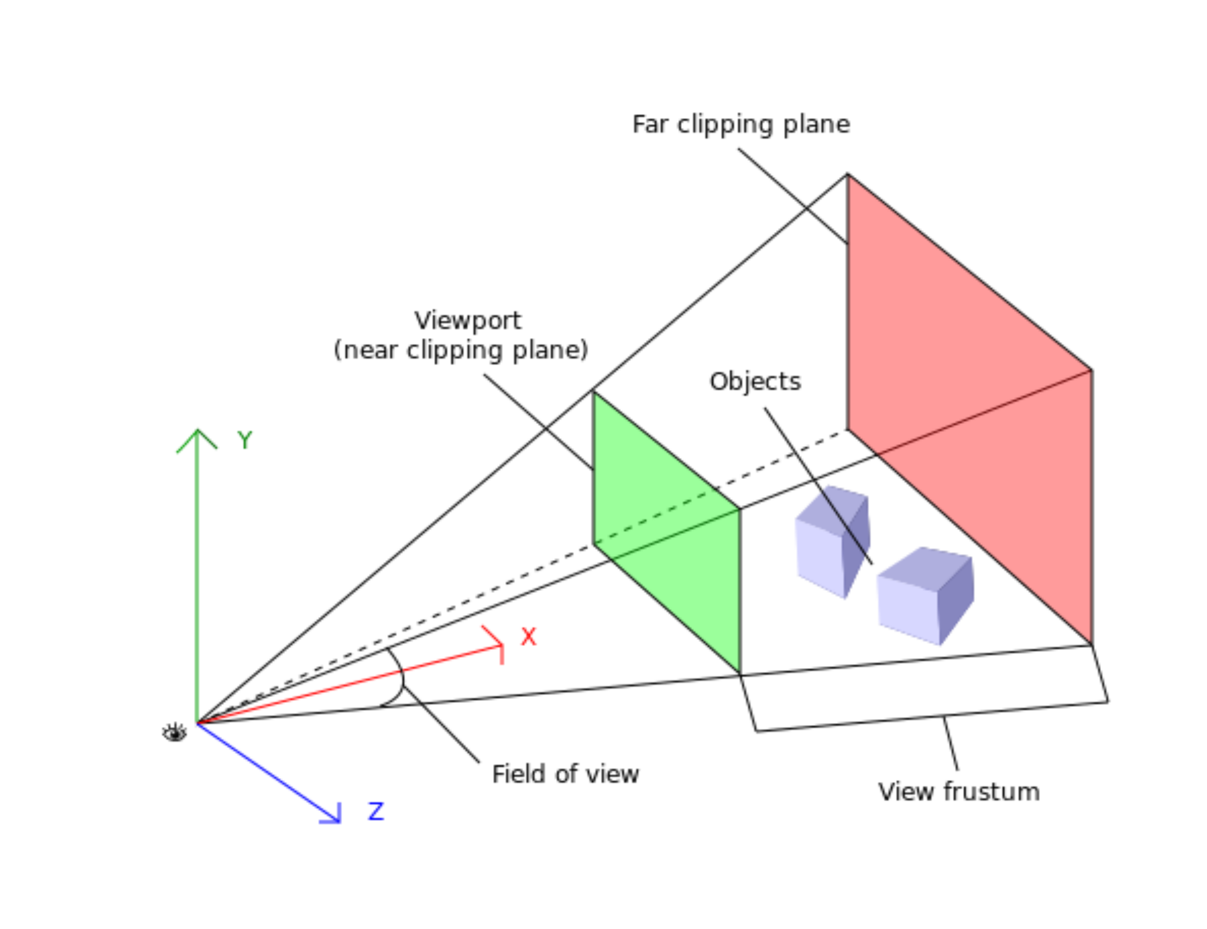
- For every triangle in the geometry
- Calculate the normal and check if the dot product of the normal and the direction is not equal to 0
- Calculate the intersection of the ray with the triangle plane and check whether it is the minimum positive intersection and whether the point lies inside the triangle or not.
- For the minimum intersection value that lies within the triangle we compute the color of that pixel.



Volumetric Shadow Rendering

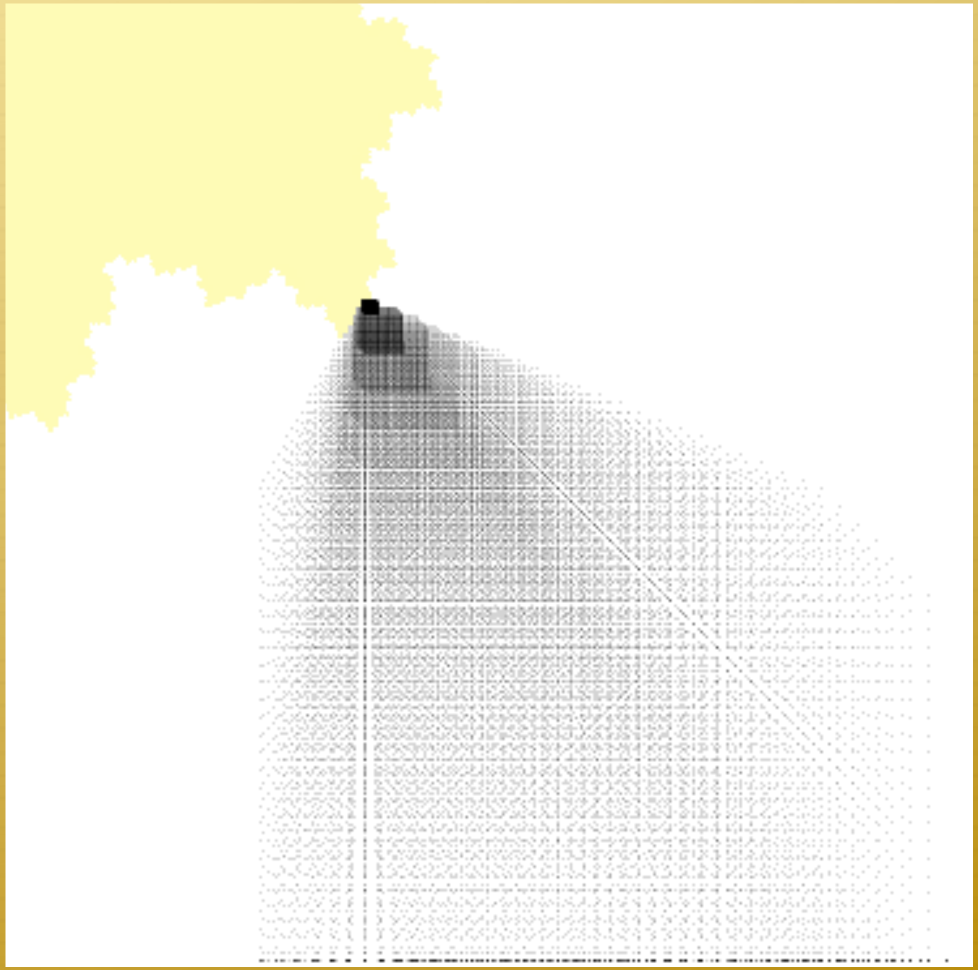
Shadow Volume

- Extension of shadow ray using ray tracing.
- Calculated aggregate shadow of each subsequent image plane between the original image plane and the closest object to light.



Shadow volume steps

- For each pixel:
  - For each image plane
    - Calculate ray equation from pixel to light
    - Find any intersections between the pixel and light
    - If any intersections: color the pixel as shadow
  - End for
- End for

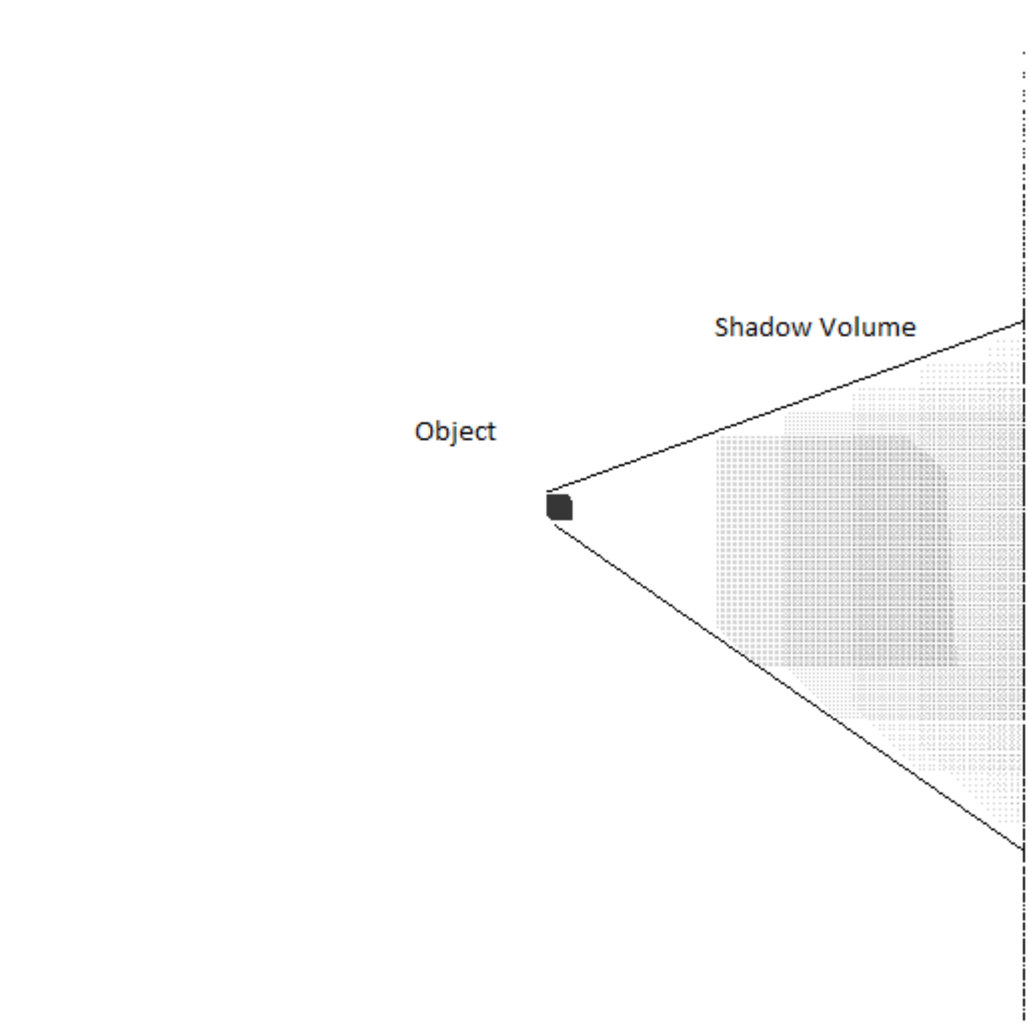


How the Shadow Volume Grows ?

Previous slide showed a compactly spaced Shadow Volume.

The figure along side gives a better understanding of the consideration of multiple plane usage for shadow Rendering.

As the distance from the object increases the Shadow becomes larger and fades out. The shadow plan here is sparsely spaced. Hence the glitchy spaced shadow volume render.



Challenges

- Ray Tracing calculations. More the number of lights more the computational complexity.
- Each pixel is accounted for all the geometry present in the object space.
- Visualization of every object in the object space in contrast to the image space
- Light positions to get shadow volume rendering.
- Separate ray equations per intermediate shadow plane led to increased computational complexity and time complexity.

Possible solutions

- Multi threaded application to reduce computation time.
- Application dependent selection of ray equation per ‘n x n’ pixel where n can be any positive number.
- Again the project was done without using any external libraries which accounts for a slow renderer. Use of external libraries can considerably reduce the time complexity.

Thank you
for your attention