CSE4203: Computer Graphics
Chapter – 4 (part - B)
Ray Tracing

Mohammad Imrul Jubair

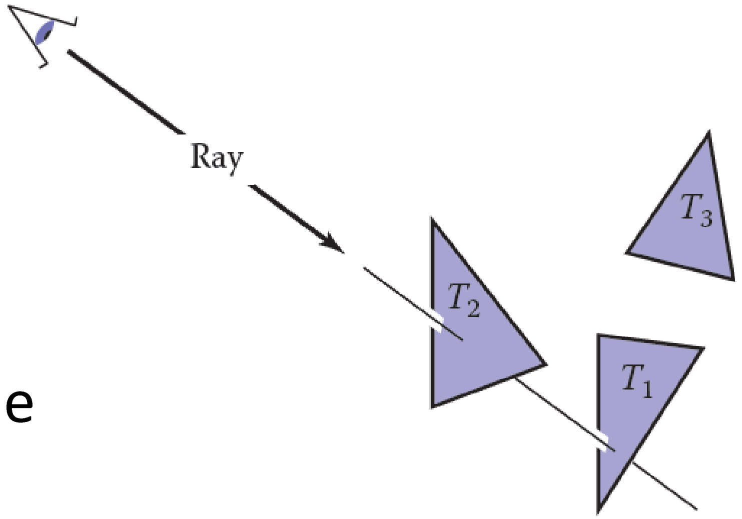
Outline

– Ray-tracing

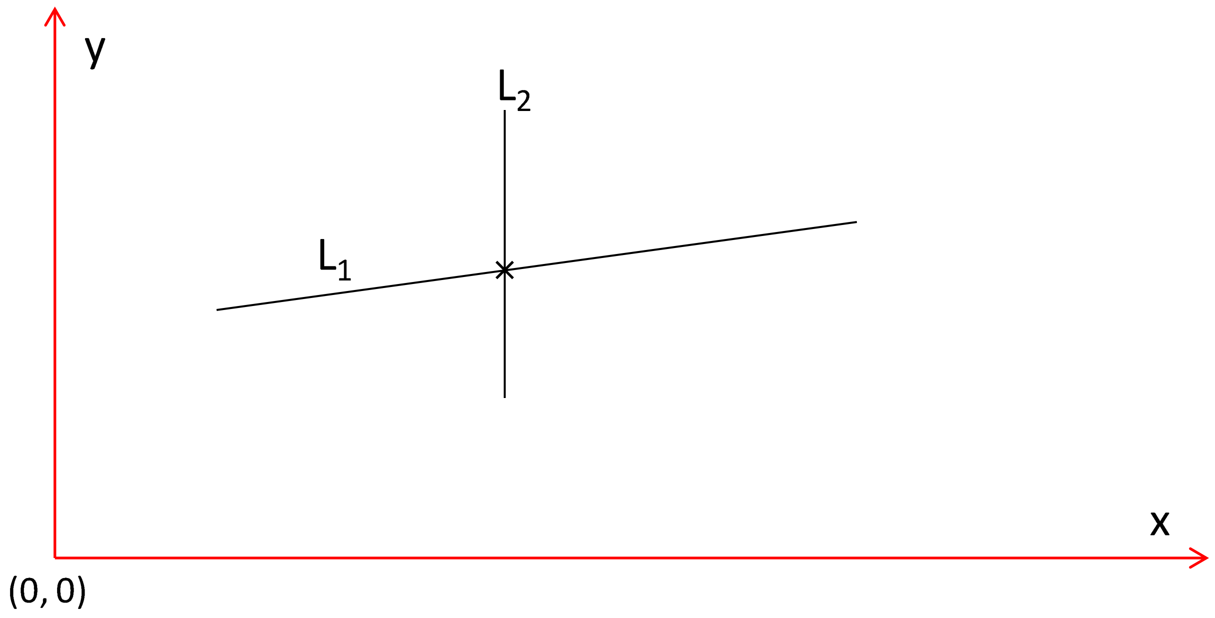
3D→2D

- Implementing projection: (3D → 2D)
  - Ray-tracing technique

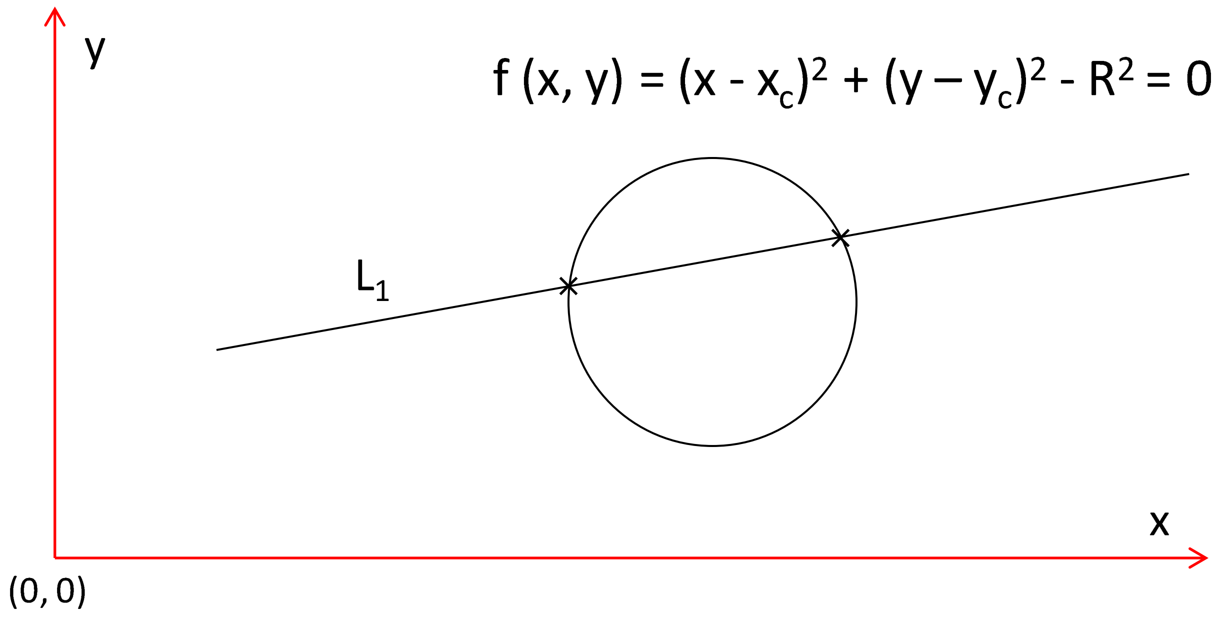
- Motivation:
  - From how we see!
  - The **ray** is "**traced**" into the **scene**
  - the first **object** hit is the one seen.
    - In this case, the triangle *T2 is returned.*



Warm-up (1/9)

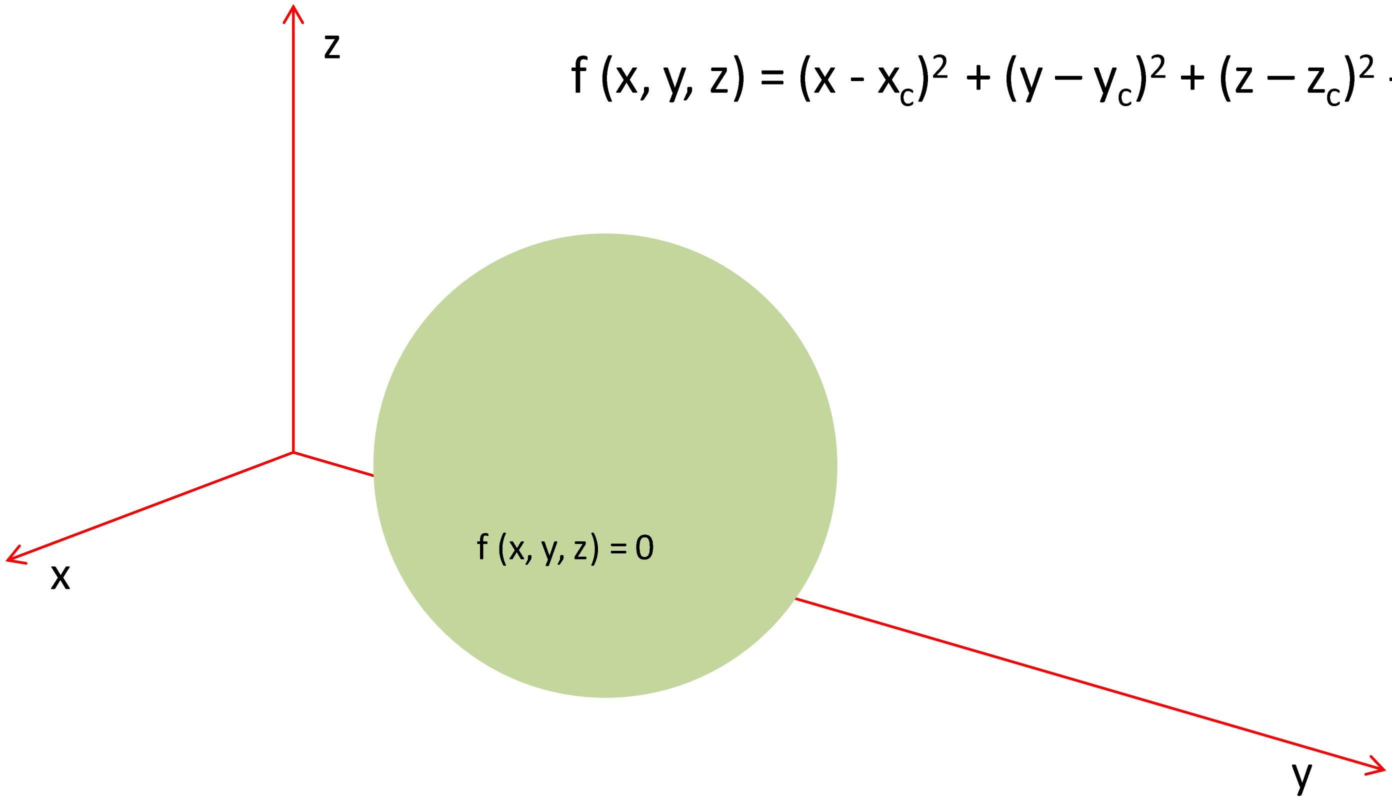


Warm-up (2/9)

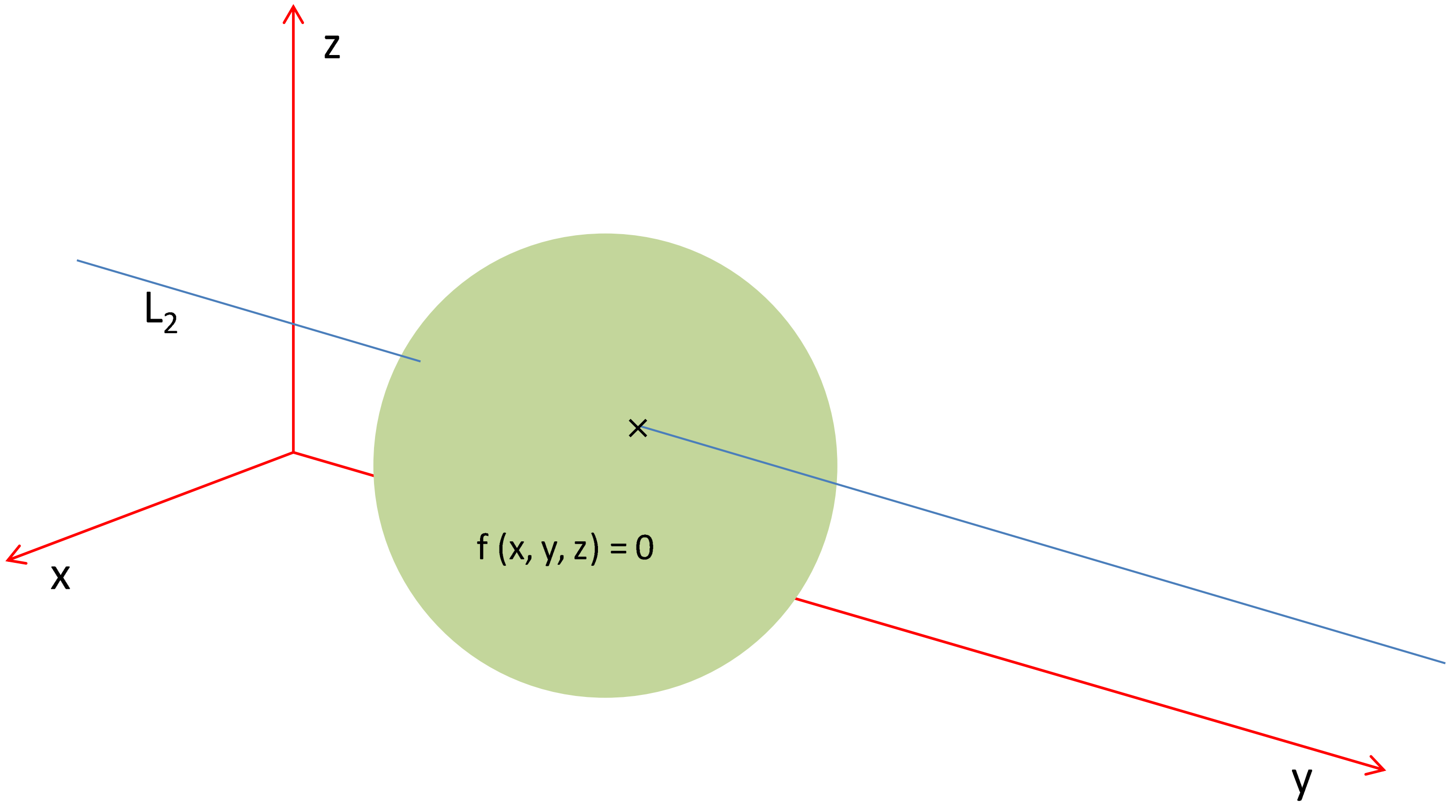


Warm-up (3/9)

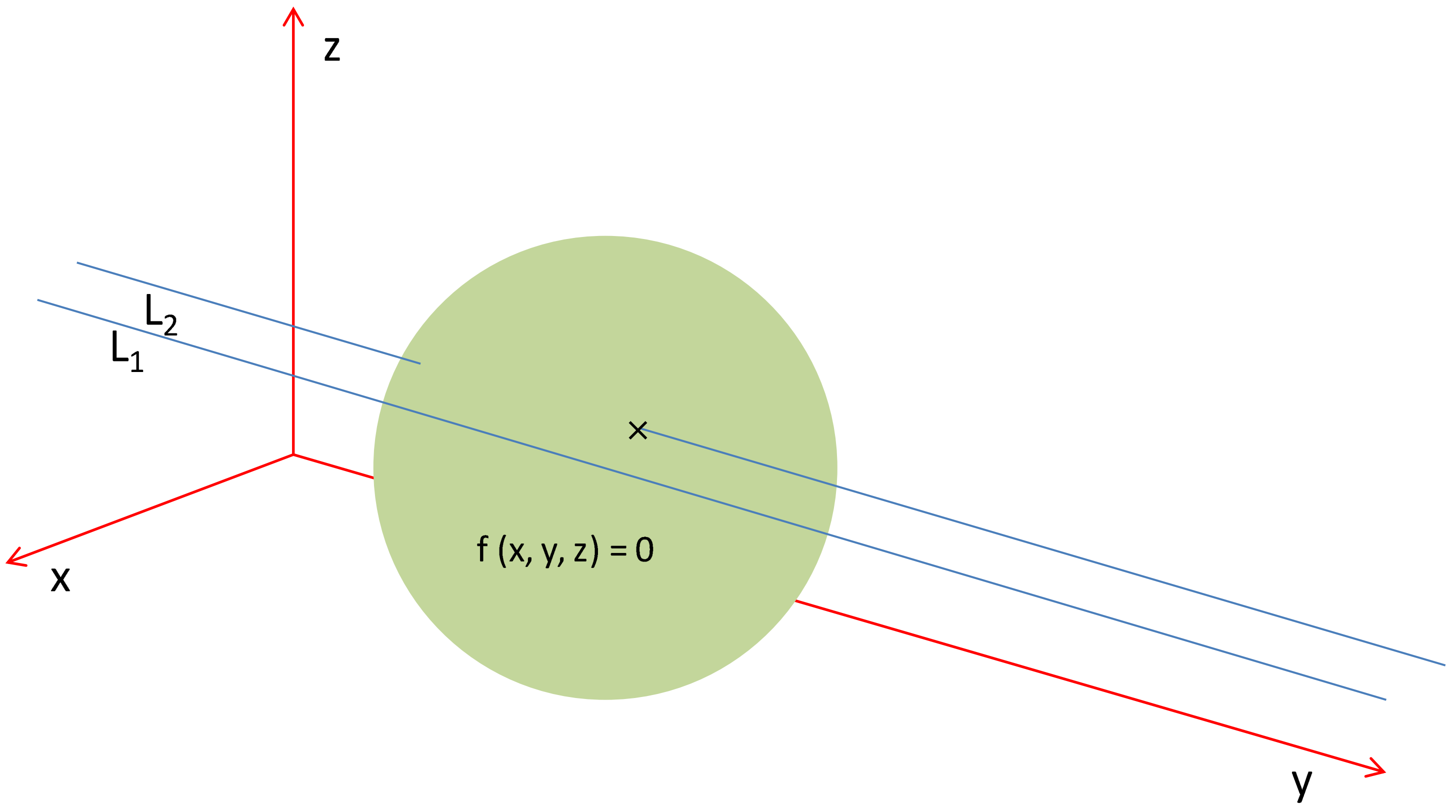
$$f(x, y, z) = (x - x_c)^2 + (y - y_c)^2 + (z - z_c)^2 - R^2 = 0$$



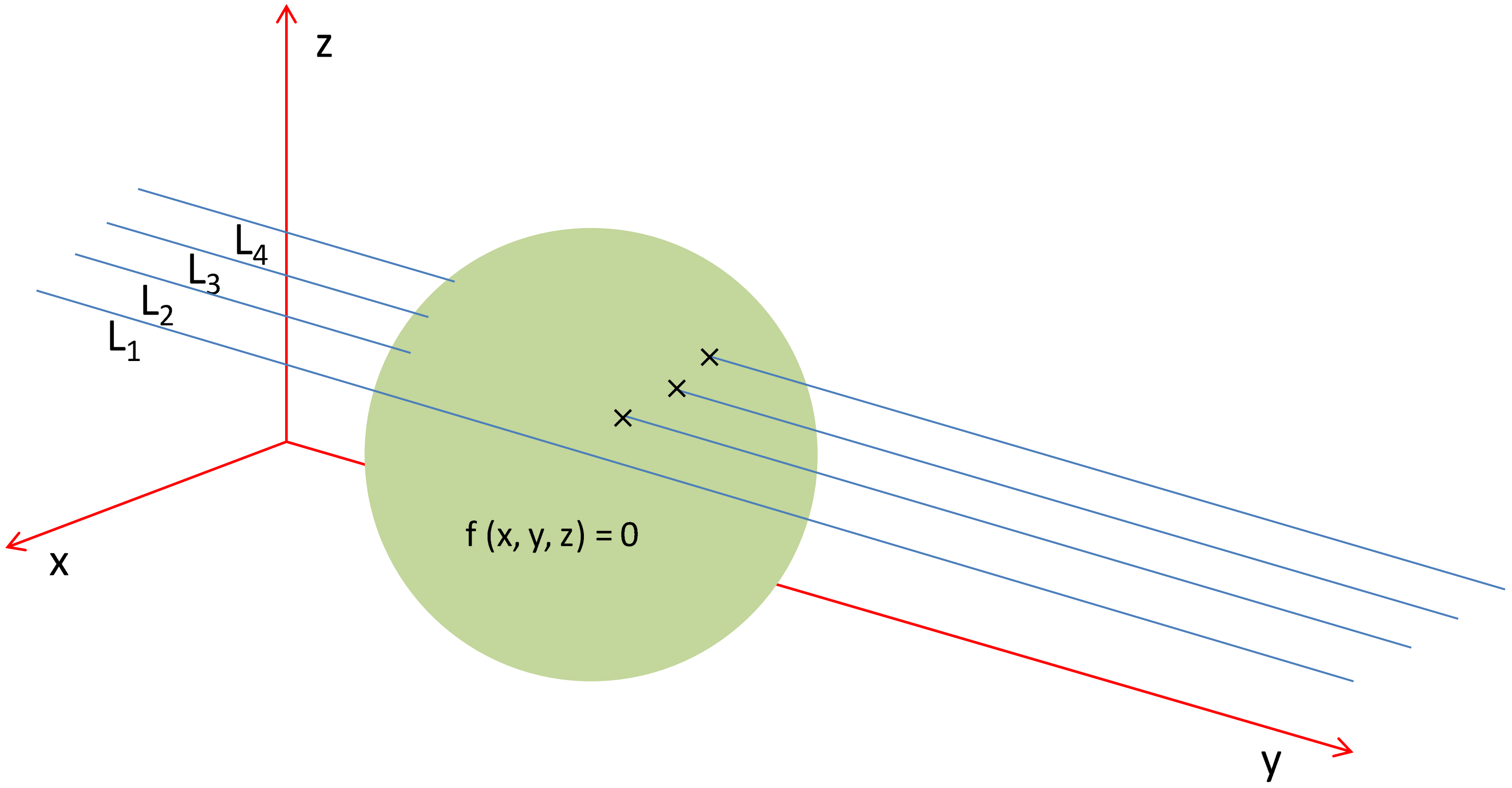
Warm-up (4/9)



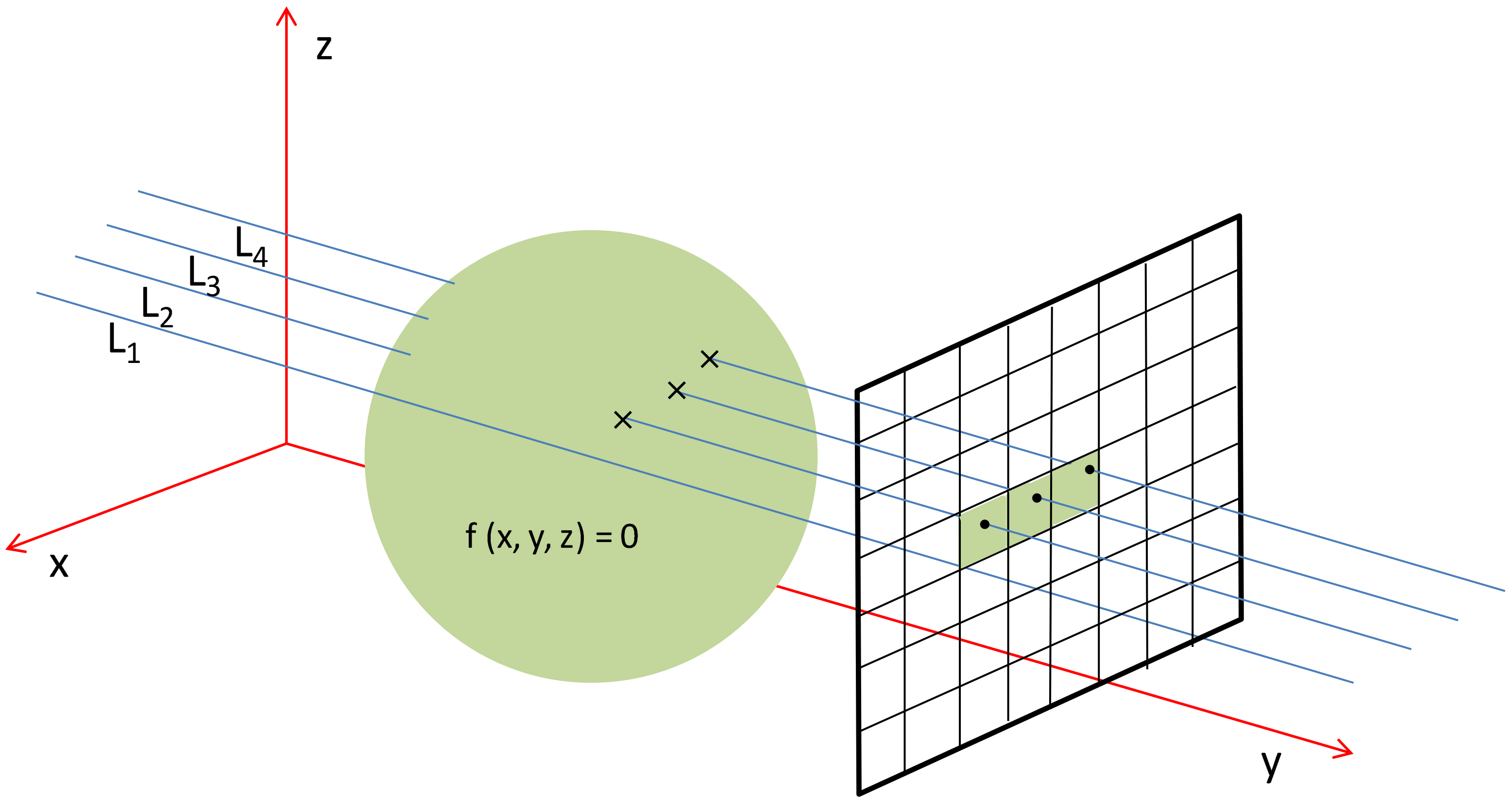
Warm-up (5/9)



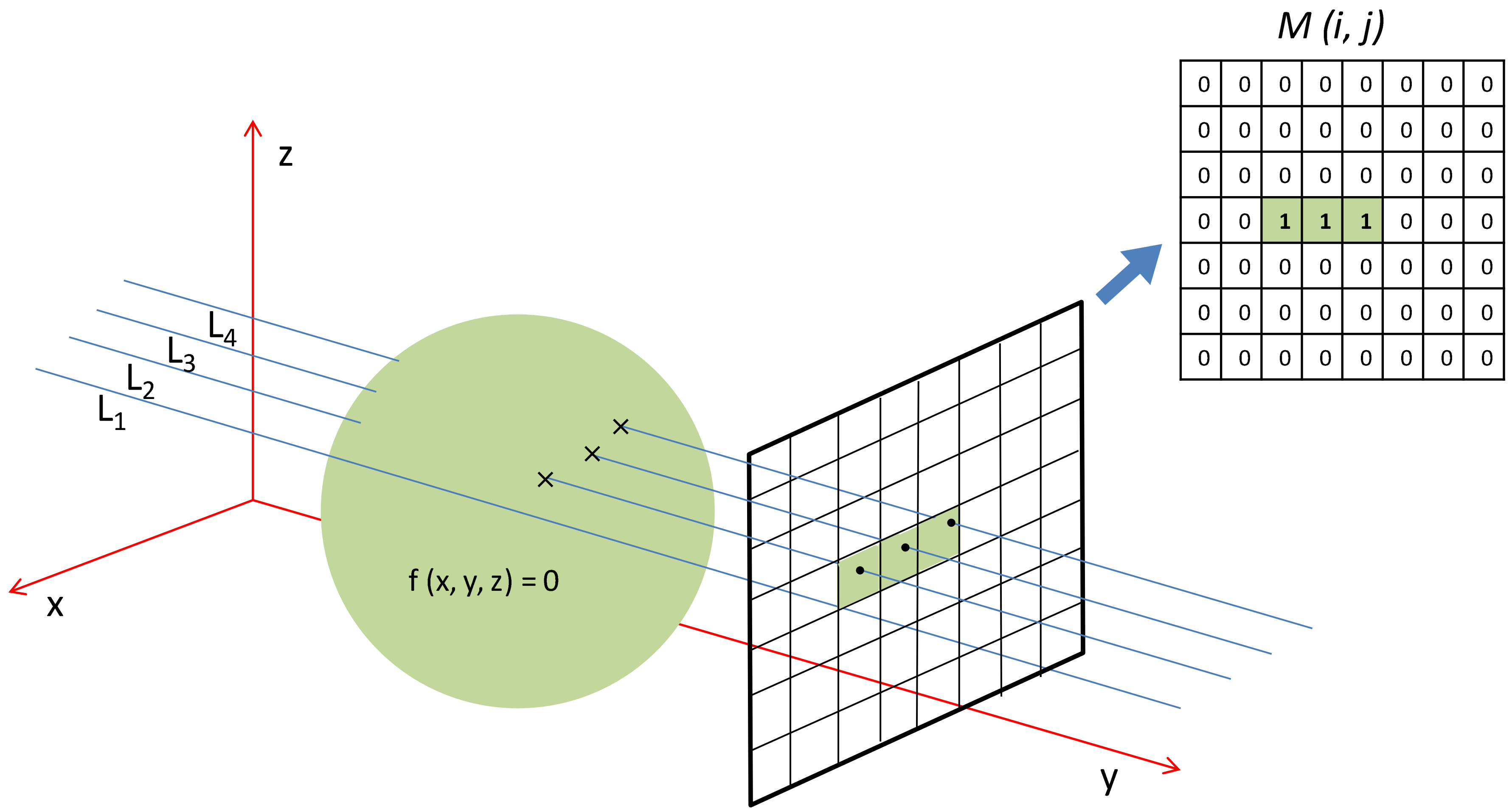
Warm-up (6/9)



Warm-up (7/9)



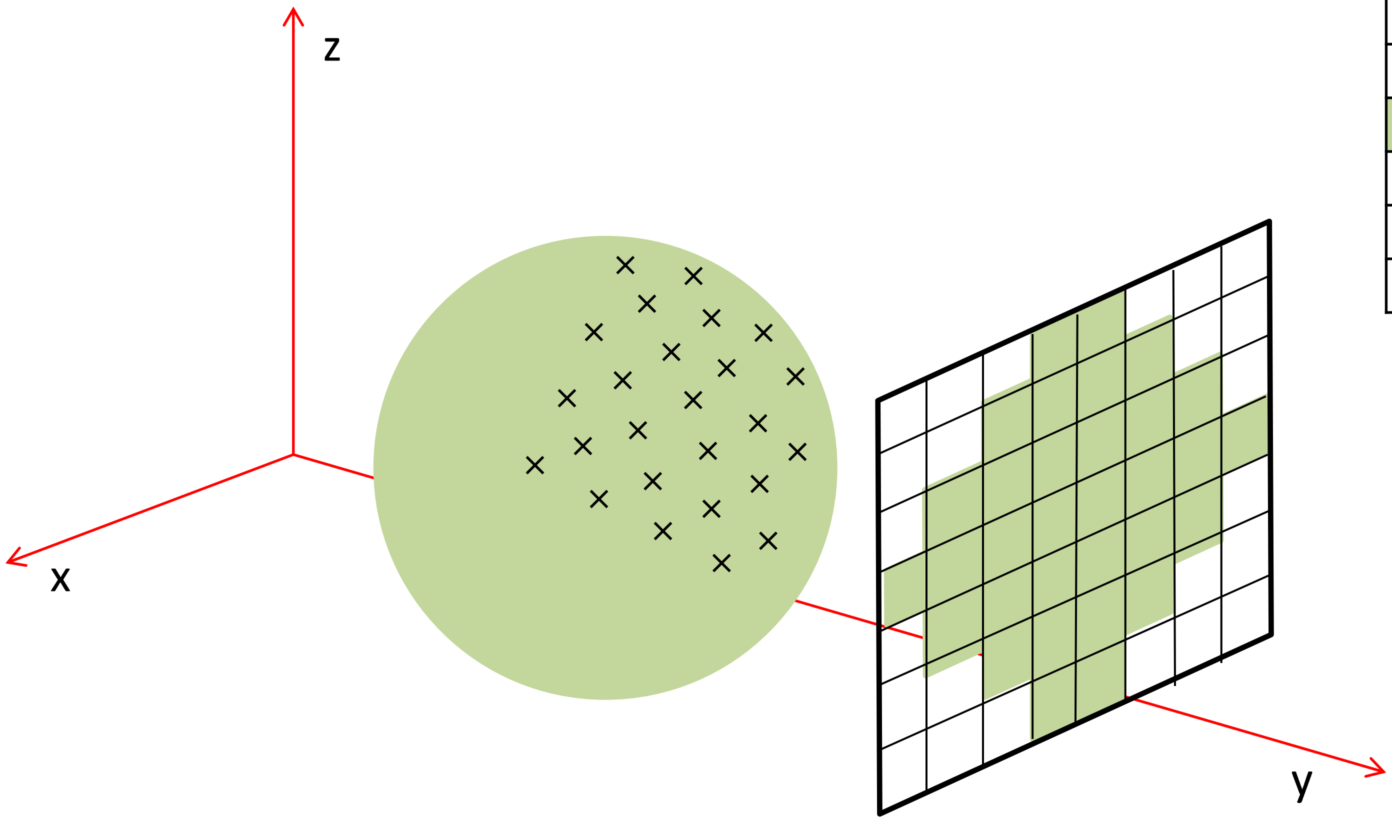
Warm-up (8/9)



M (i, j)

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	**1**	**1**	**1**	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

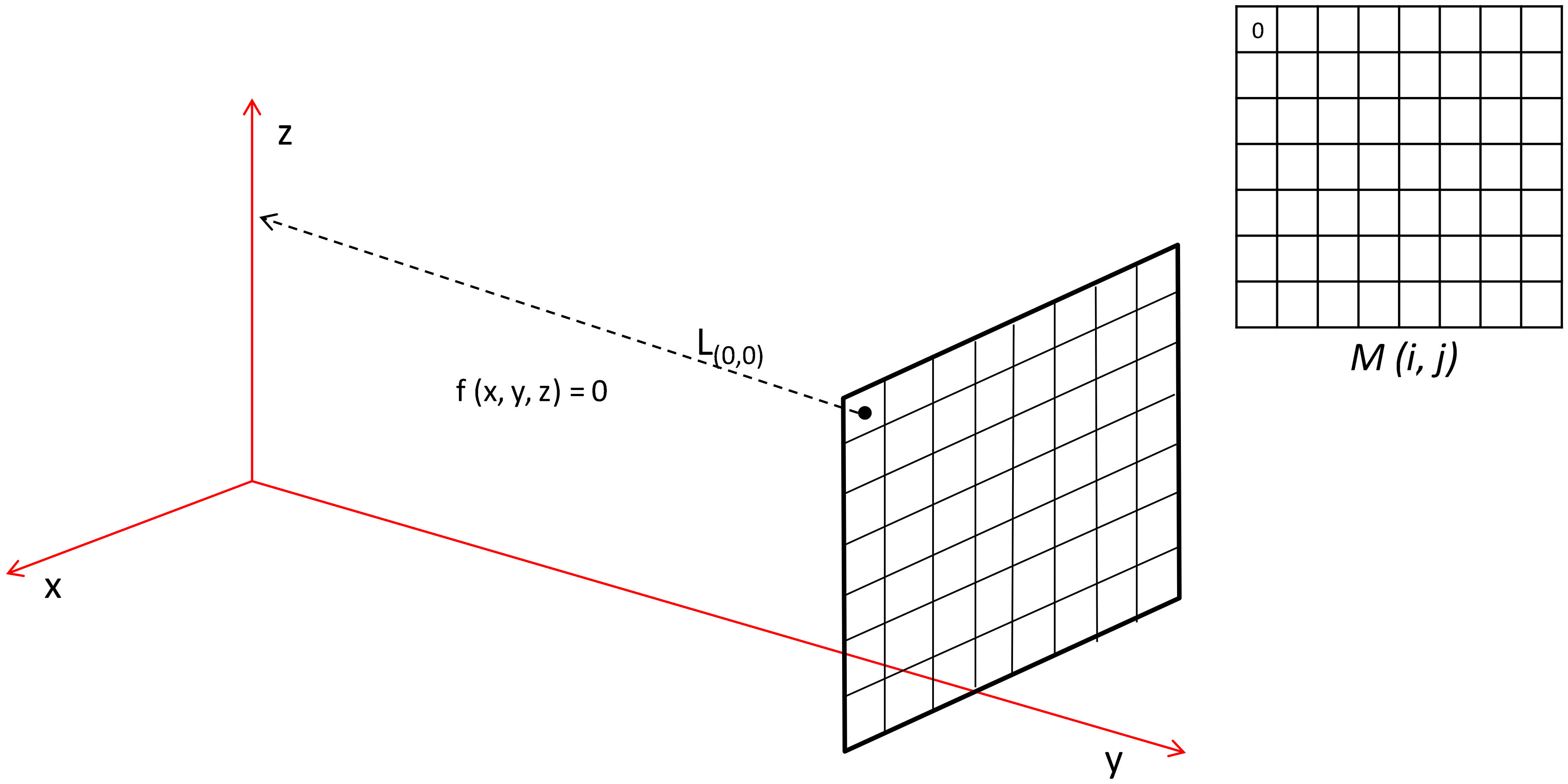
Warm-up (9/9)



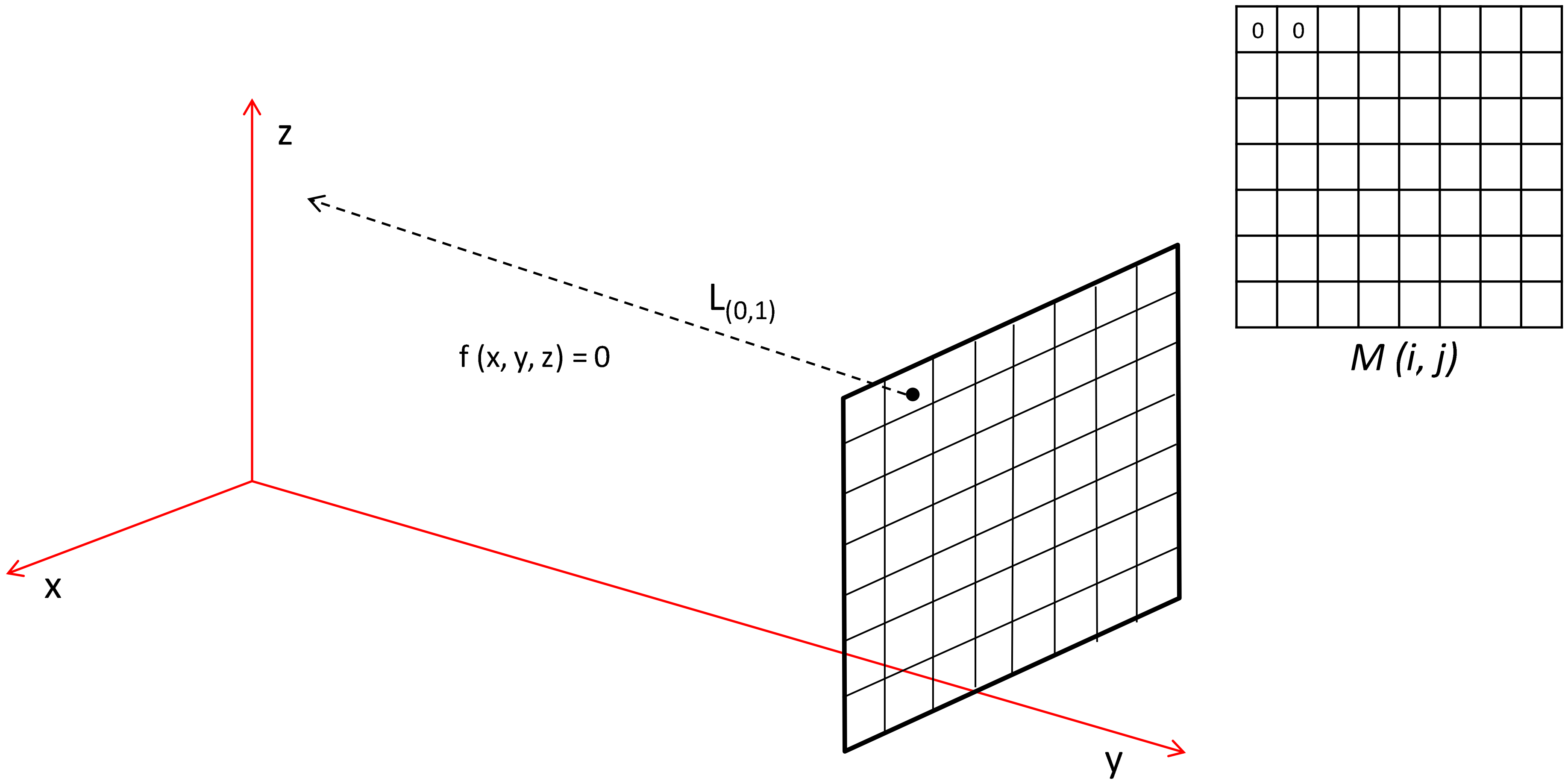
M (i, j)

0	0	0	1	1	0	0	0
0	0	1	1	1	1	0	0
0	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1
0	1	1	1	1	1	1	0
0	0	1	1	1	1	0	0
0	0	0	1	1	0	0	0

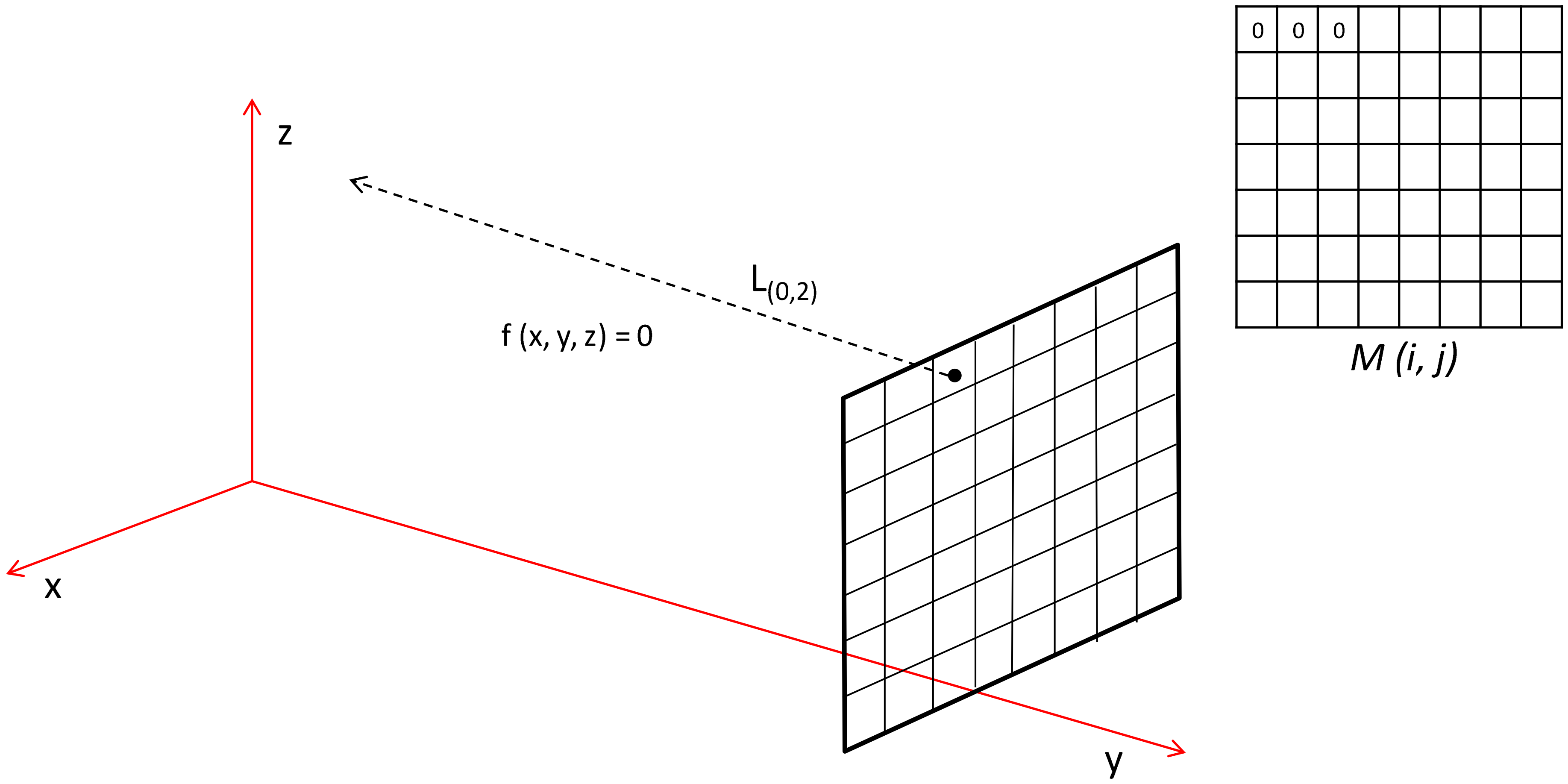
Ray-tracing Basics (1/15)



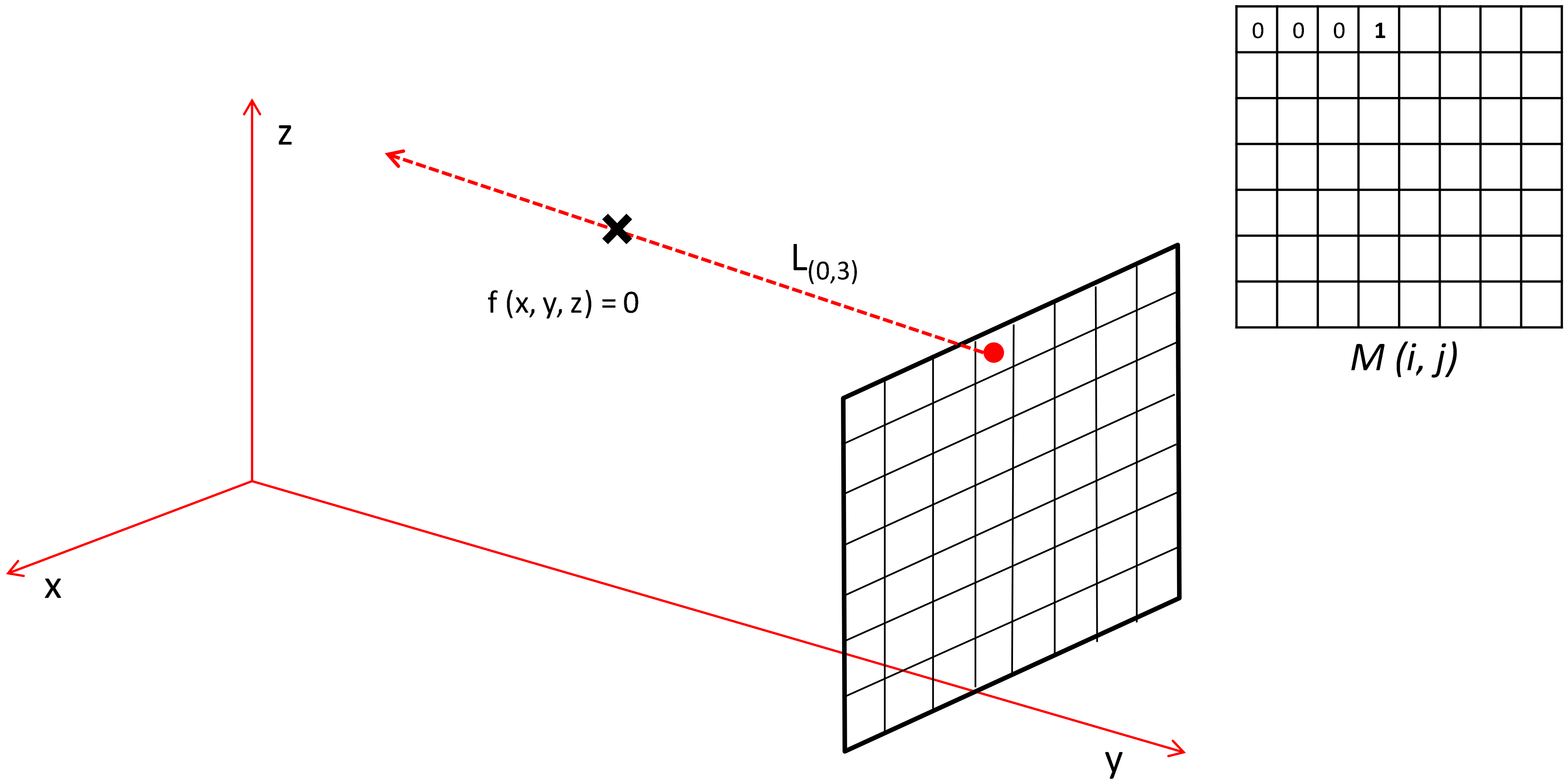
Ray-tracing Basics (2/15)



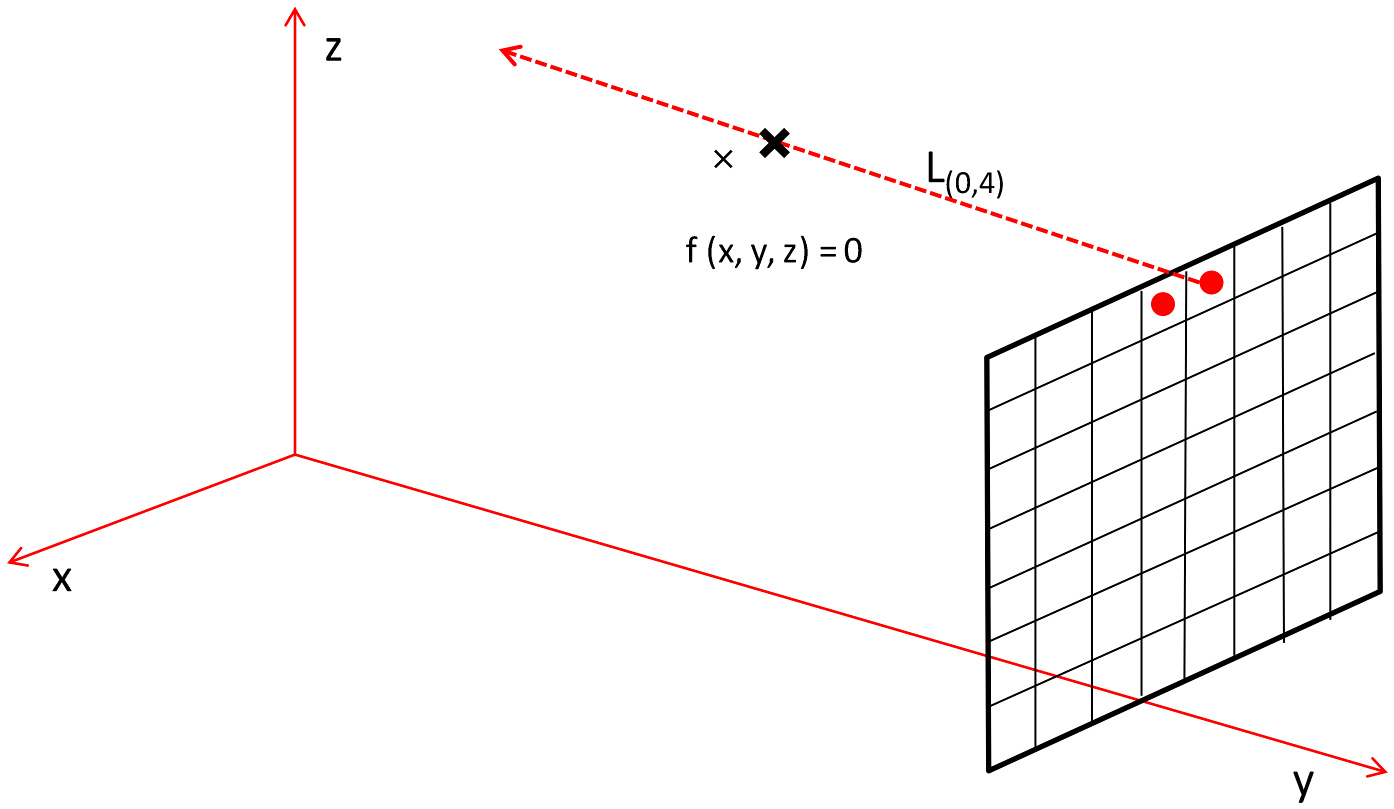
Ray-tracing Basics (3/15)



Ray-tracing Basics (4/15)



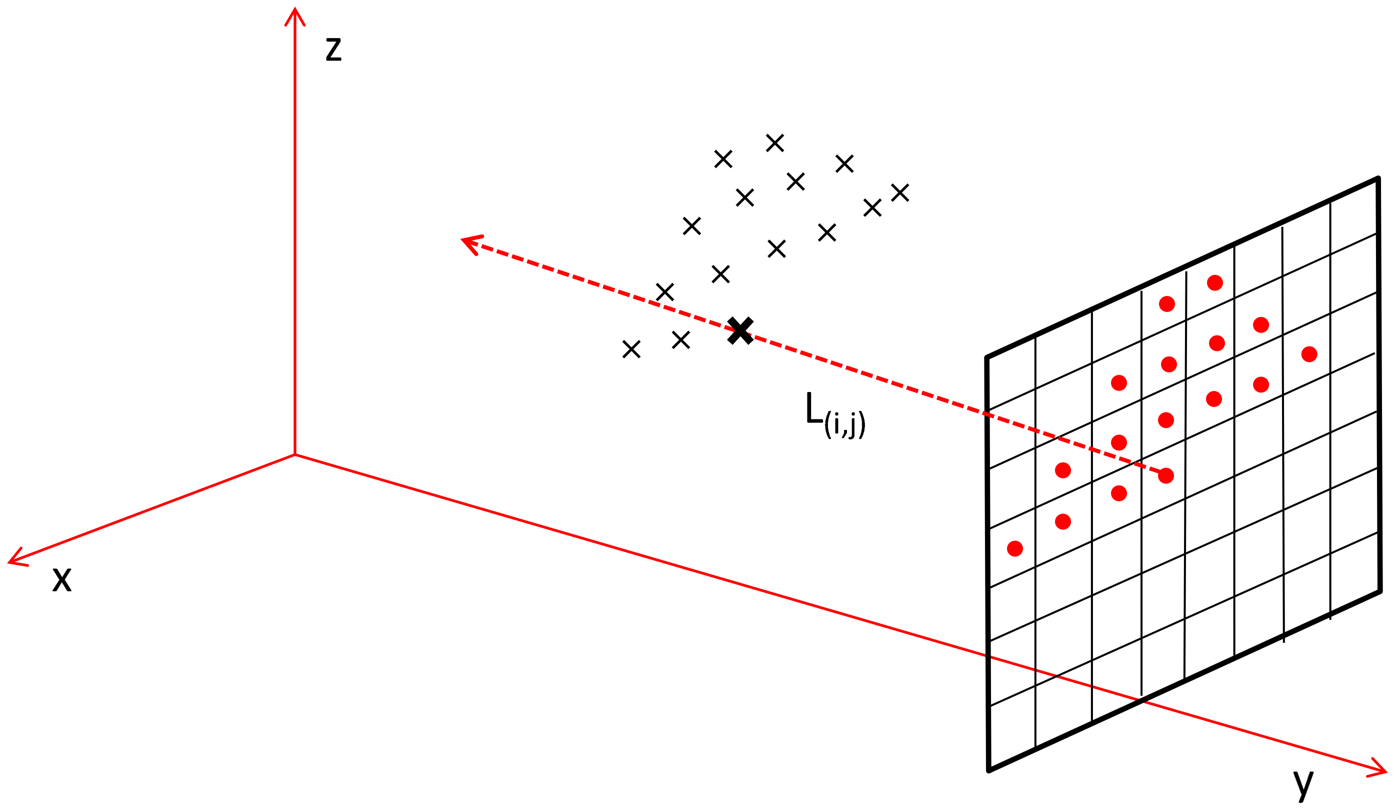
Ray-tracing Basics (5/15)



0	0	0	**1**	**1**			

M (i, j)

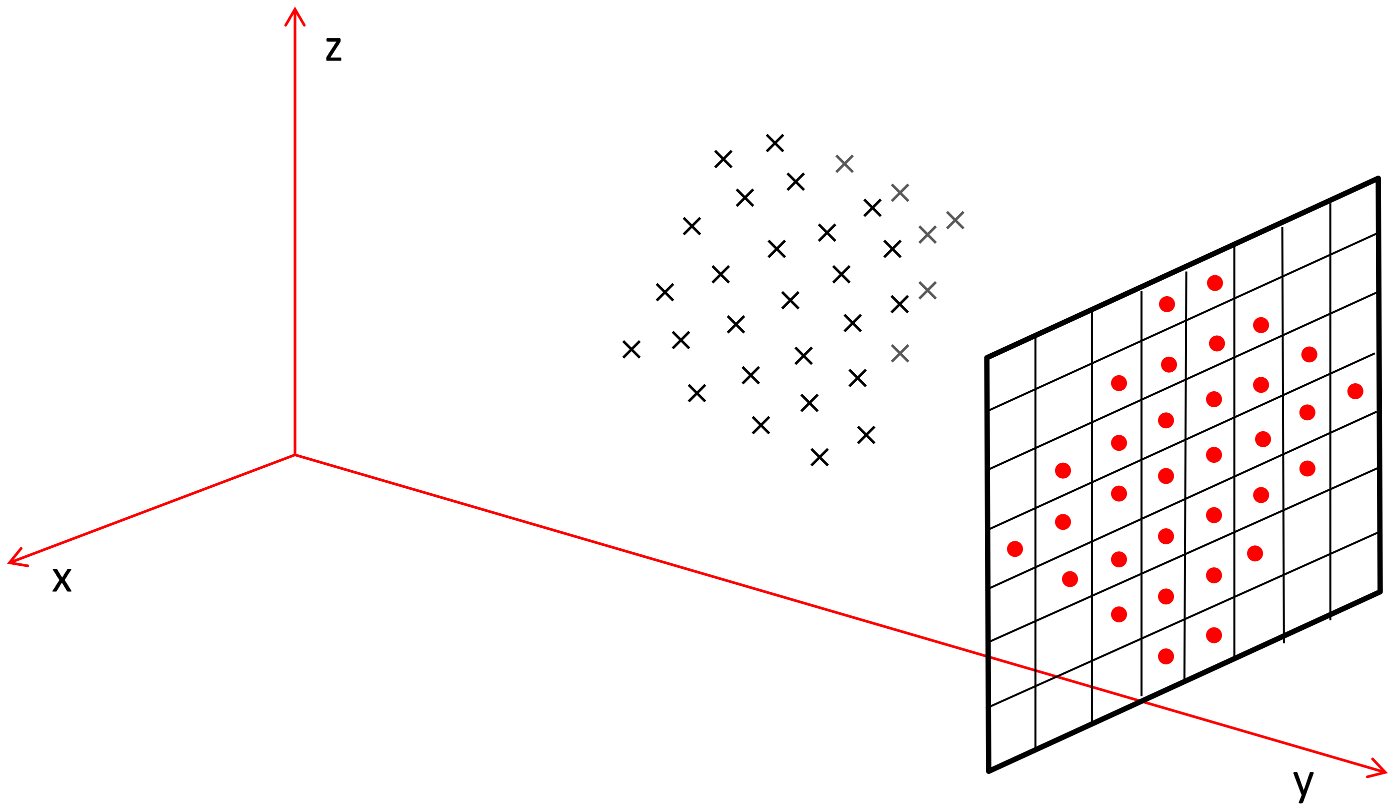
Ray-tracing Basics (6/15)



0	0	0	**1**	**1**	0	0	0
0	0	**1**	**1**	**1**	**1**	0	0
0	**1**	**1**	**1**	**1**	**1**	**1**	0
1	**1**	**1**	**1**				

M (i, j)

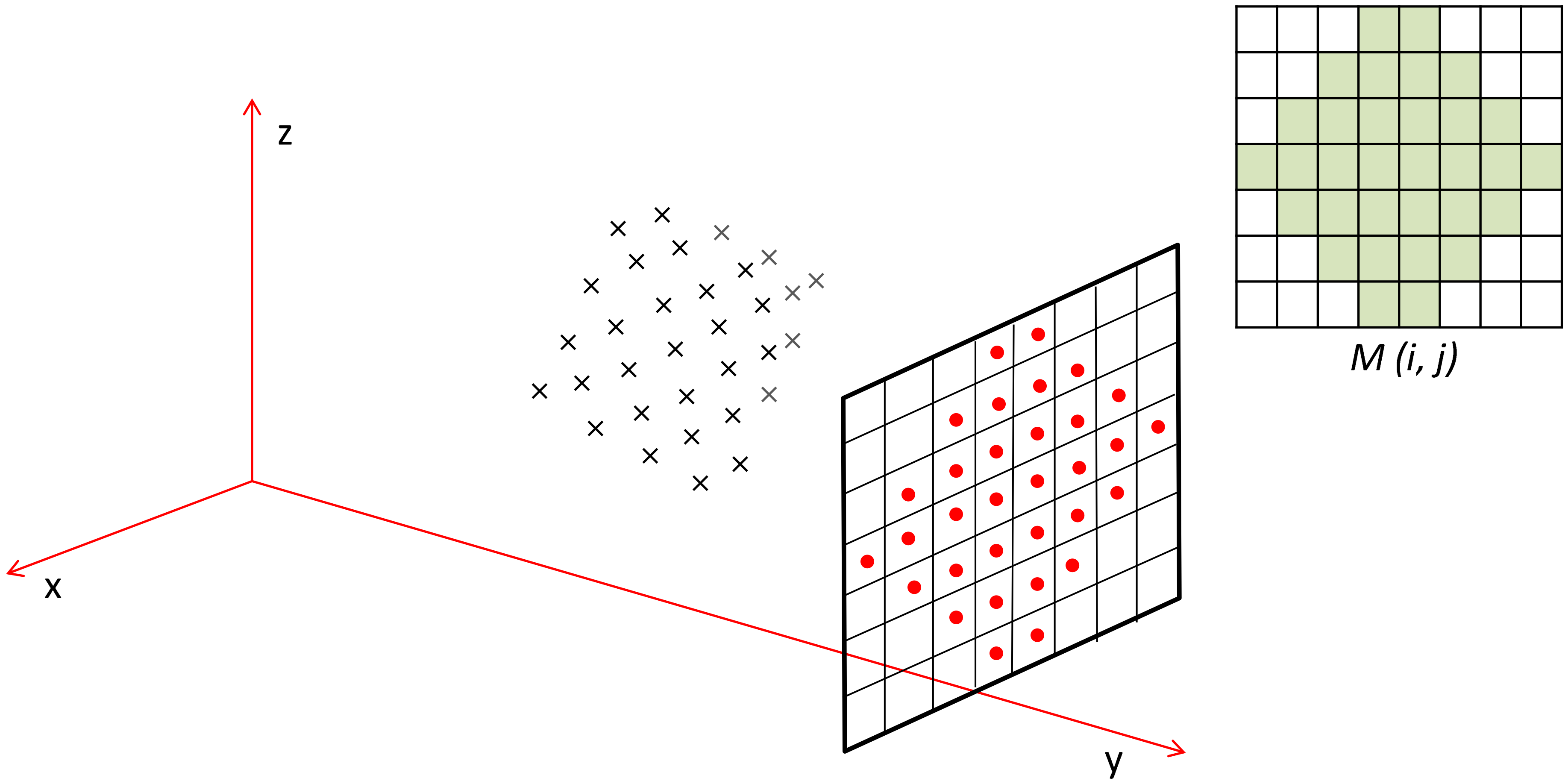
Ray-tracing Basics (7/15)



0	0	0	**1**	**1**	0	0	0
0	0	**1**	**1**	**1**	**1**	0	0
0	**1**	**1**	**1**	**1**	**1**	**1**	0
1	**1**	**1**	**1**	**1**	**1**	**1**	**1**
0	**1**	**1**	**1**	**1**	**1**	**1**	0
0	0	**1**	**1**	**1**	**1**	0	0
0	0	0	**1**	**1**	0	0	0

M (i, j)

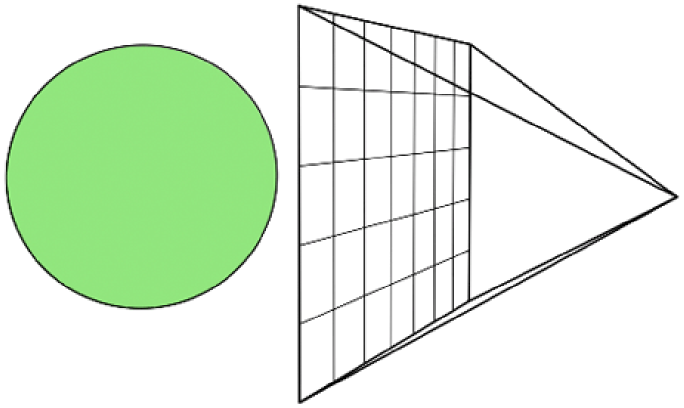
Ray-tracing Basics (8/15)



Ray-tracing Basics (9/15)

- Computing one pixel at a time
  - Each pixel "**looks**" in a **direction**

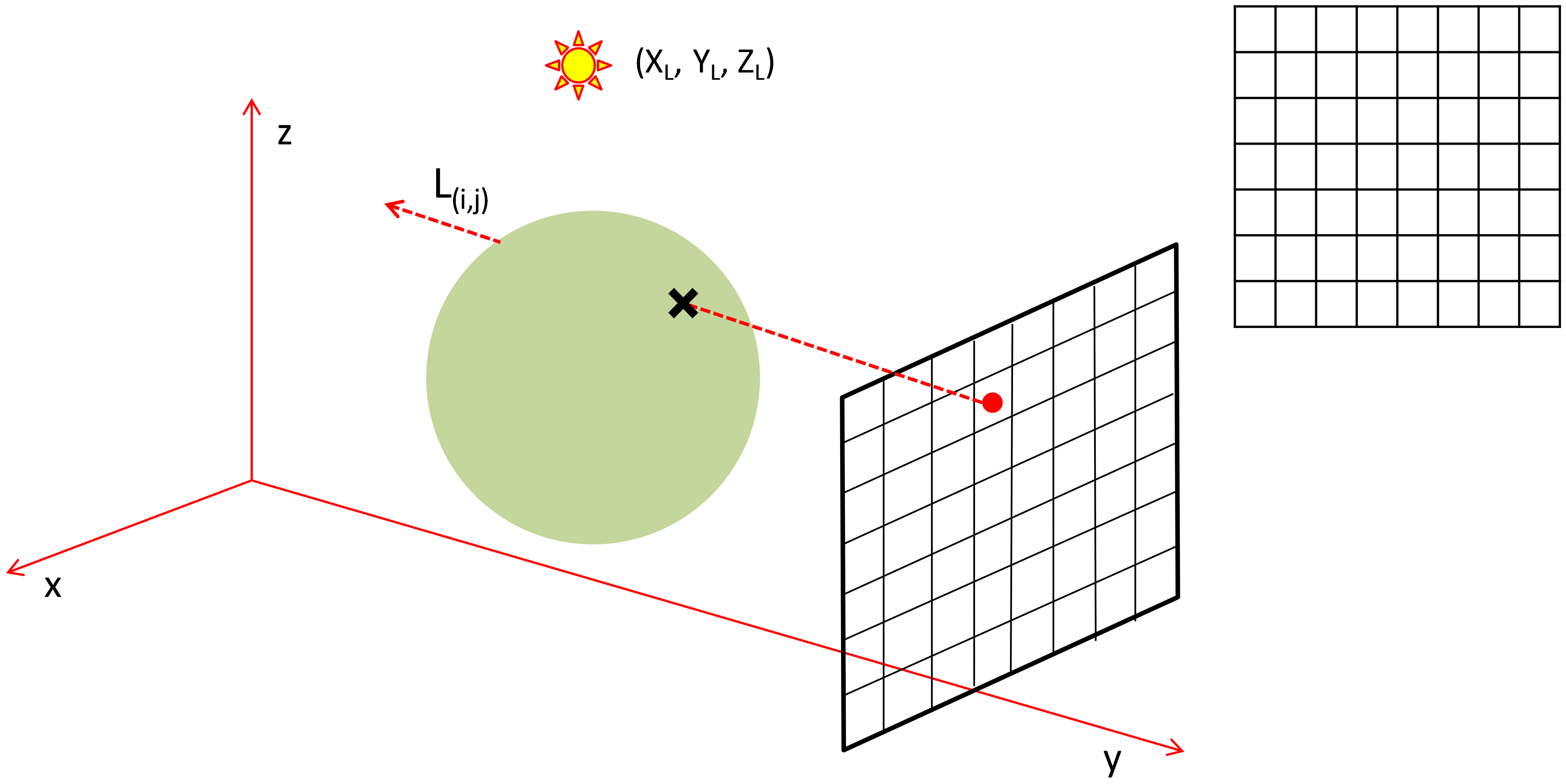
- Any object that is seen by a pixel
  - intersect "***viewing ray***"
  - viewing ray: line through that pixel is looking



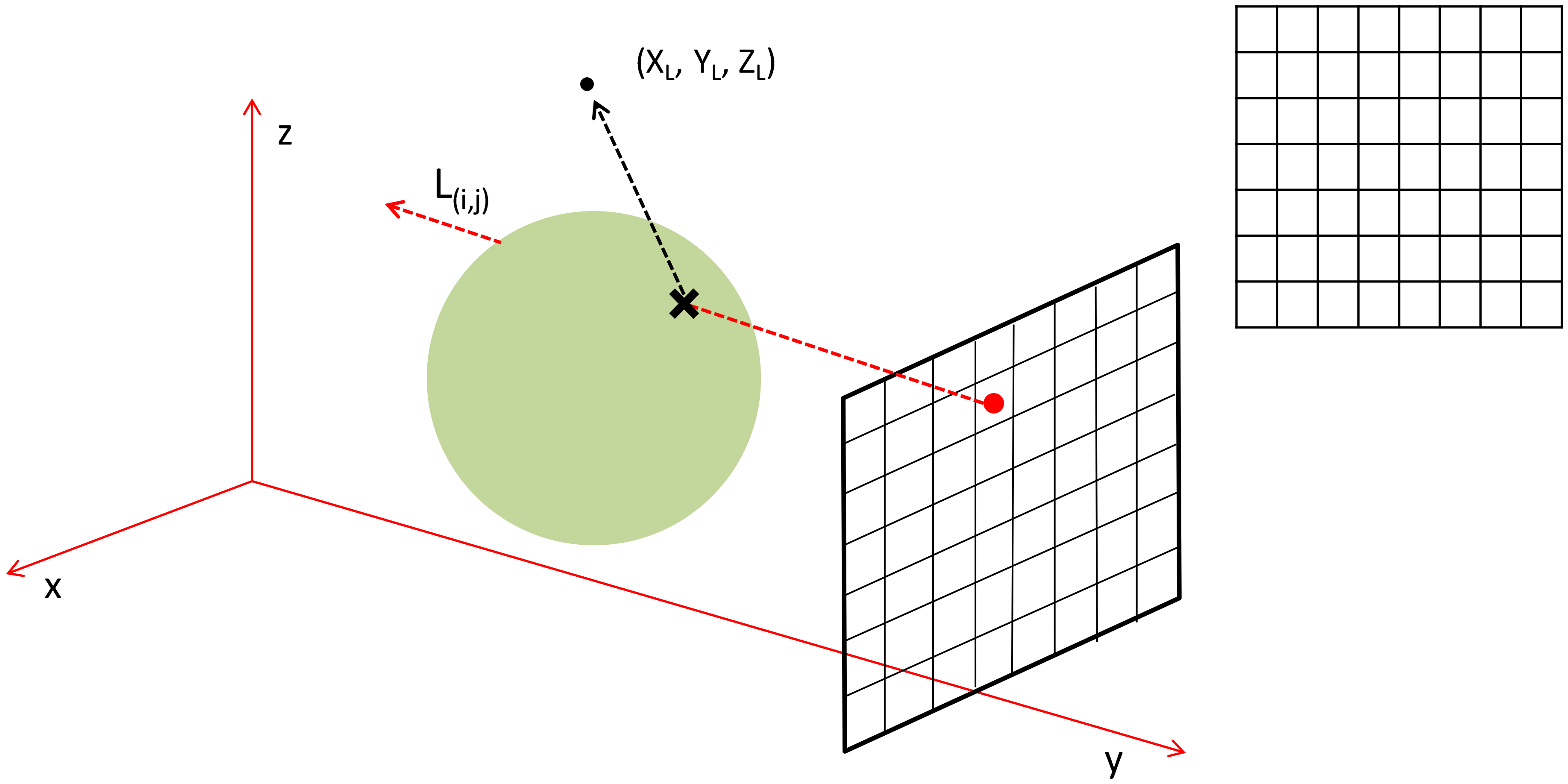
Ray-tracing Basics (10/15)

- Once that object is found, determine the color of the pixel.
  - a **shading computation** is need, that uses
    - the intersection point
    - surface normal (*n*)
    - other information

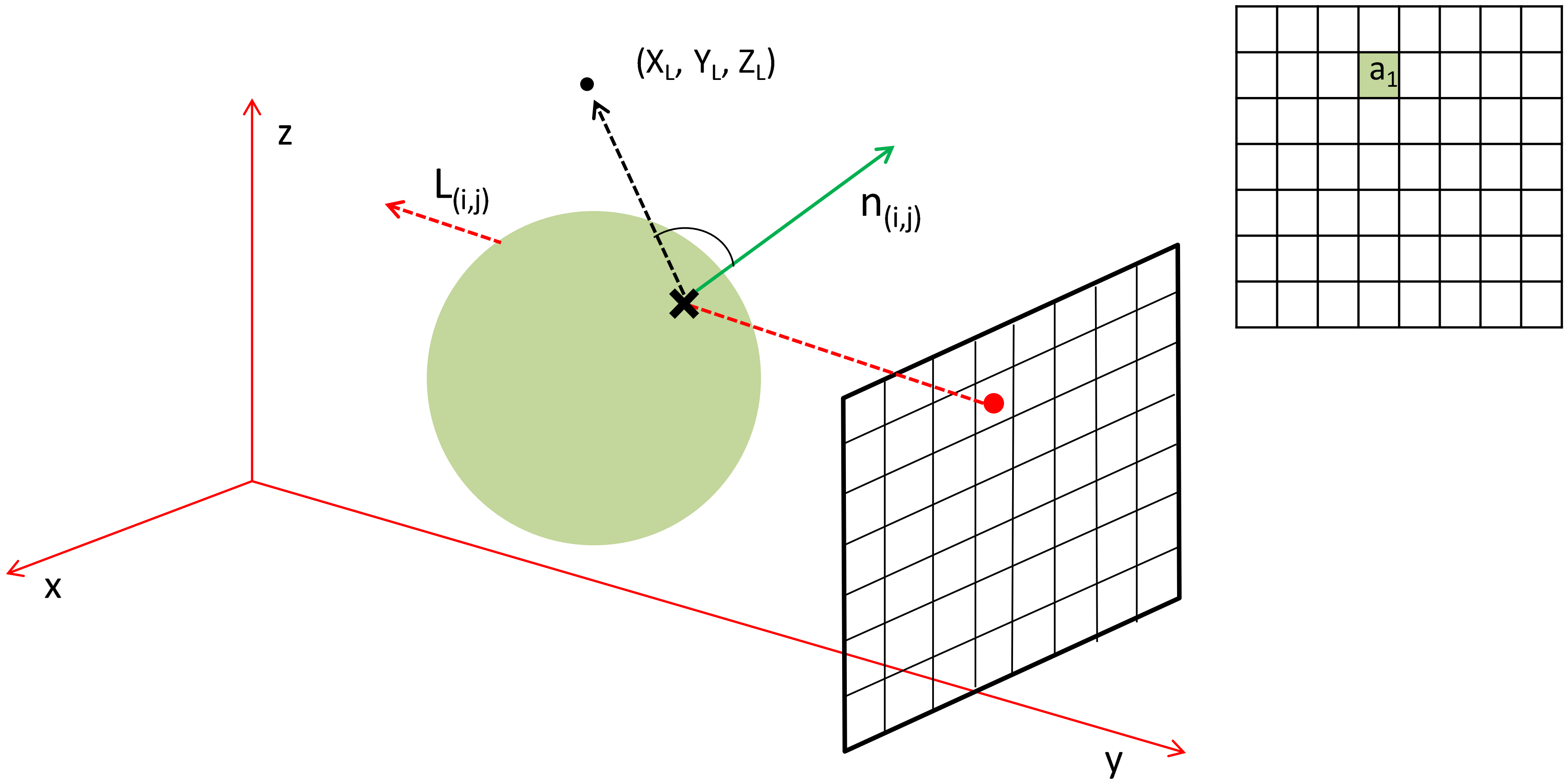
Ray-tracing Basics (11/15)



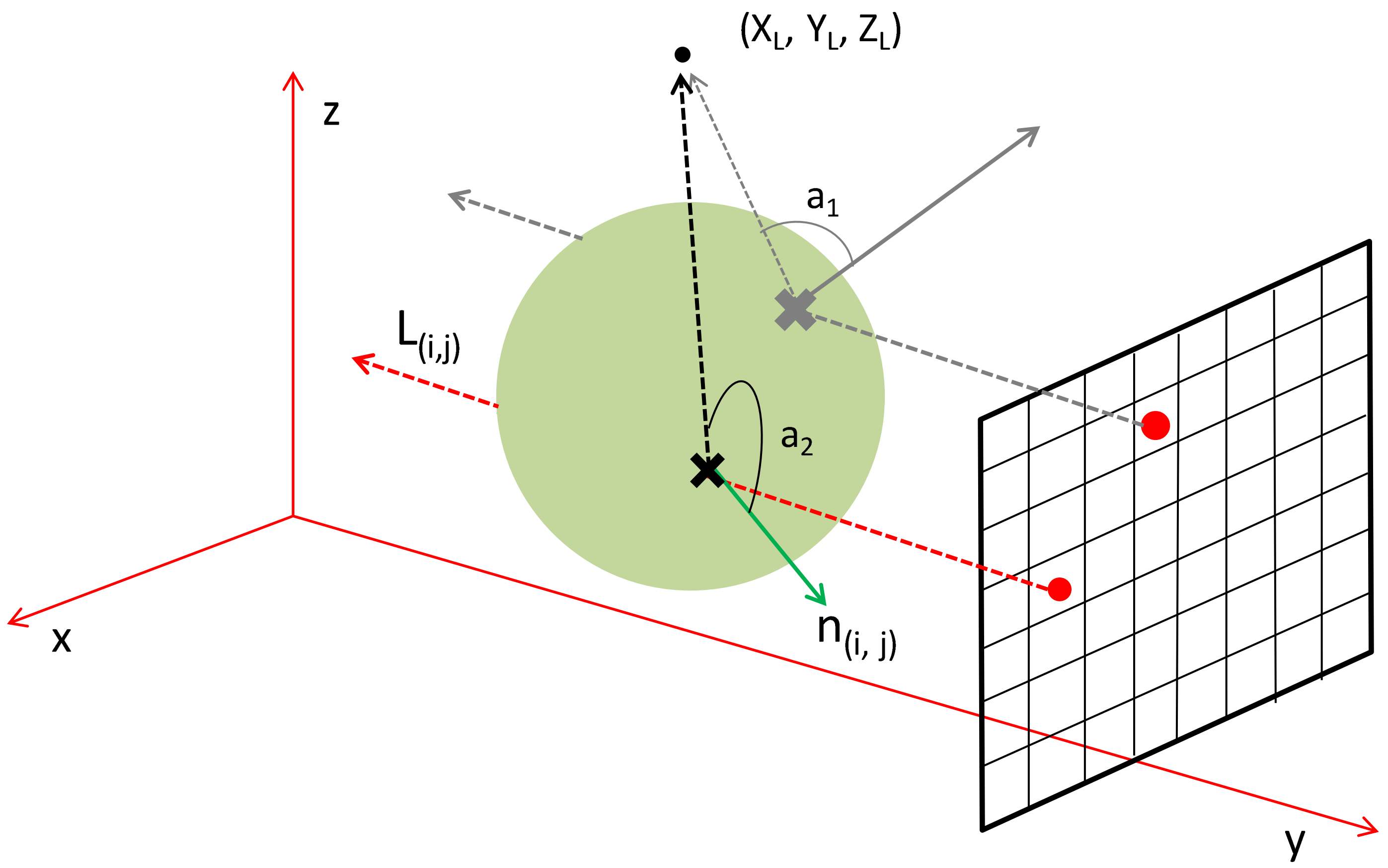
Ray-tracing Basics (12/15)

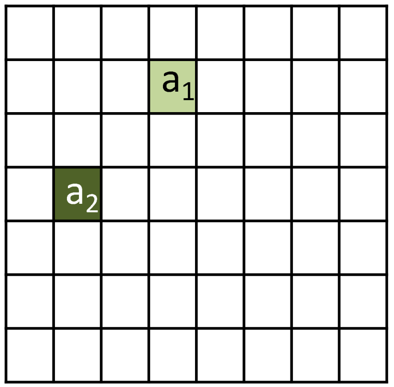


Ray-tracing Basics (13/15)

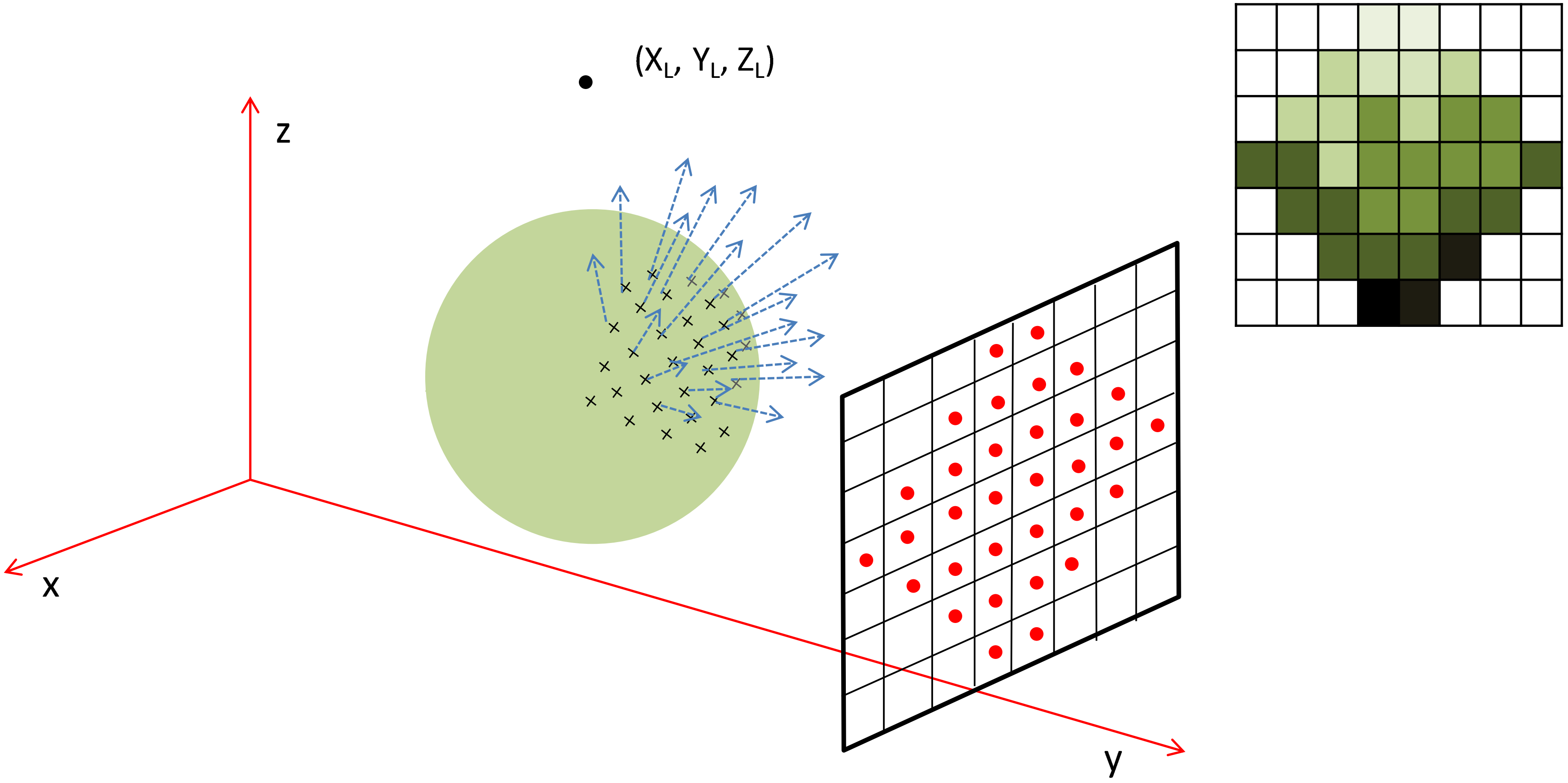


Ray-tracing Basics (14/15)





Ray-tracing Basics (15/15)

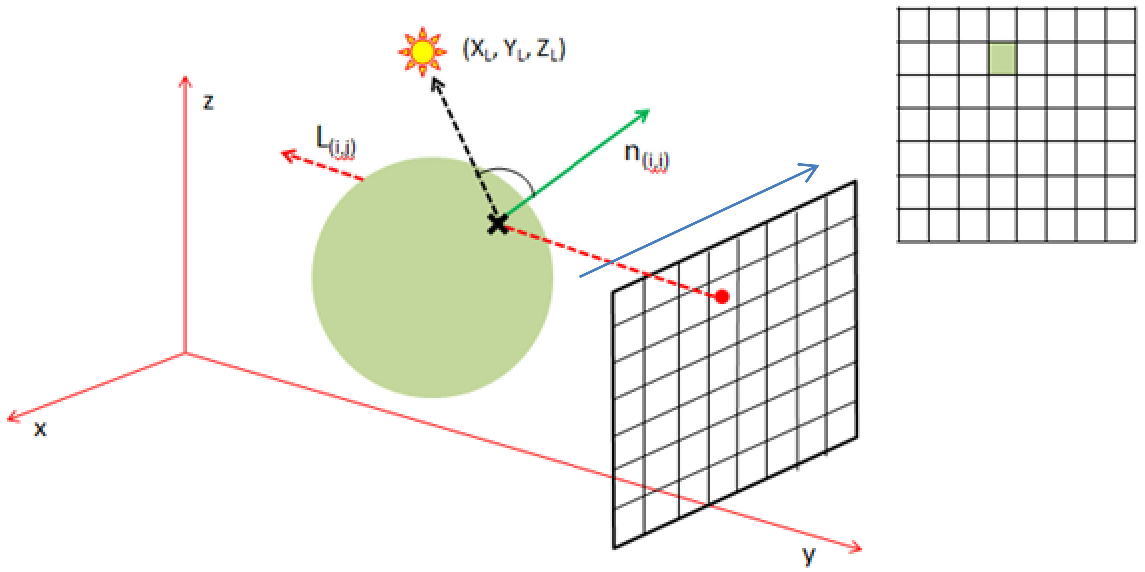


Ray-Tracing Algorithm (1/3)

- A basic ray tracer therefore has three parts:
  - **ray generation:**
    - computes the origin and direction of each pixel's viewing ray.
  - **ray intersection:**
    - finds the closest object intersecting the viewing ray.
  - **shading:**
    - computes the pixel color based on the results of ray intersection.

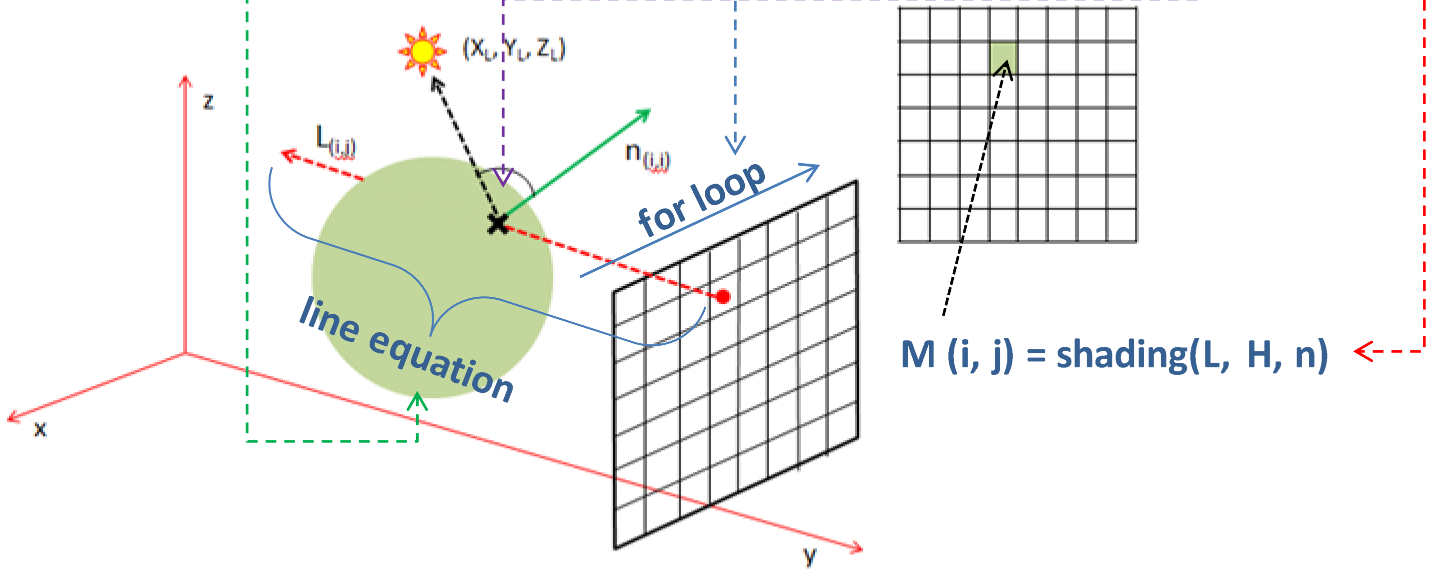
Ray-Tracing Algorithm (2/3)

- **for each pixel do:**
  - compute viewing ray
  - find first object hit by ray and its surface normal **n**
  - set pixel color computed from hit point, light, and **n**



Ray-Tracing Algorithm (3/3)

- **for each pixel do:**
  - compute viewing ray
  - find first object hit by ray and its surface normal **n**
  - set pixel color computed from hit point, light, and **n**



Practice Problems

1. Is the projected image on the image plane in the given example perspective?
2. Consider the following setup*:
   - Image plane: Situated at *y = 13*, parallel to *ZX* plane, (Resolution: *11 × 11*), *M* is the corresponding array and *Y-axis* goes through (*6, 6*).
   - Sphere: Center at (*0, 0, 0*), *Radius = 6*.
   - Light: at *(0, 15, 0)*.

   Now –

   a) Draw the ray-tracing setup showing two viewing rays (one hitting, another missing).

   b) Fill up the array (pixel) with 1 (for hitting) and 0 (for missing). Show the hitting/ missing mathematically for at least one pixel.

   c) Fill up the array (pixel) with angles between surface normal and viewing ray. Show the angle calculation for at least one pixel.

* This problem can be helpful for understanding basic ray-tracing algorithm from the scratch.

Thank You