Introduction to Computer Graphics

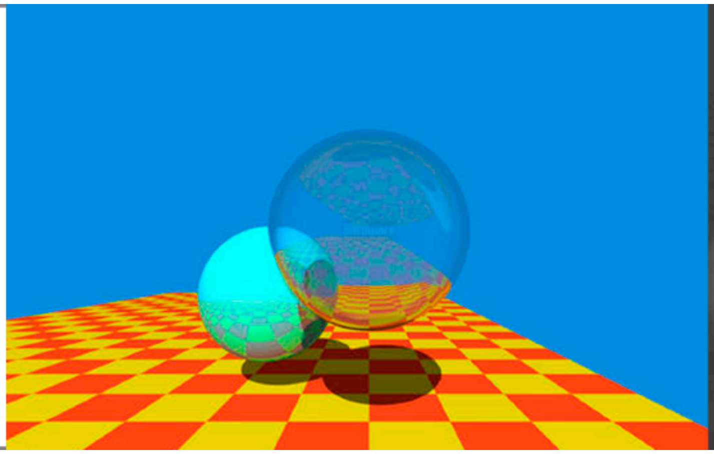
GAMES101, Lingqi Yan, UC Santa Barbara

Lecture 14:

Ray Tracing 2

(Acceleration & Radiometry)






http://www.cs.ucsb.edu/~lingqi/teaching/games101.html

Announcements

- Grading of resubmissions — we're working on that
- GTC news: DLSS 2.0
    - https://zhuanlan.zhihu.com/p/116211994
- GTC news: RTXGI
    - https://developer.nvidia.com/rtxgi
- Personal feeling
    - Offline rendering techniques will soon become real-time
    - Current real-time rendering techniques will still be useful
- Next lectures won't be easy

Last Lecture

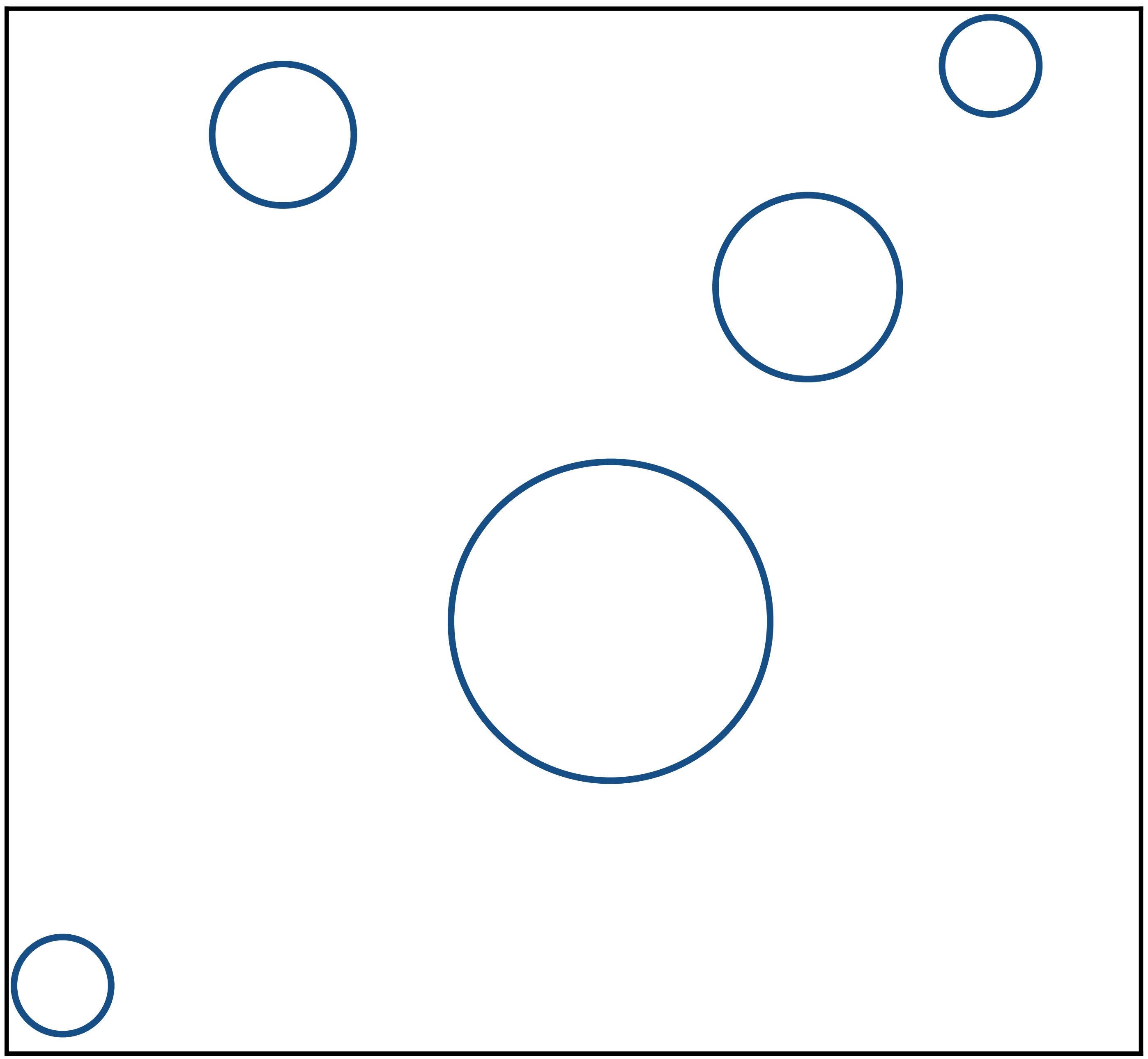
- Why ray tracing?
- Whitted-style ray tracing
- Ray-object intersections
  - Implicit surfaces
  - Triangles
- Axis-Aligned Bounding Boxes (AABBs)
  - Understanding — pairs of slabs
  - Ray-AABB intersection

Today

- Using AABBs to accelerate ray tracing
  - Uniform grids
  - Spatial partitions
- Basic radiometry (辐射度量学)

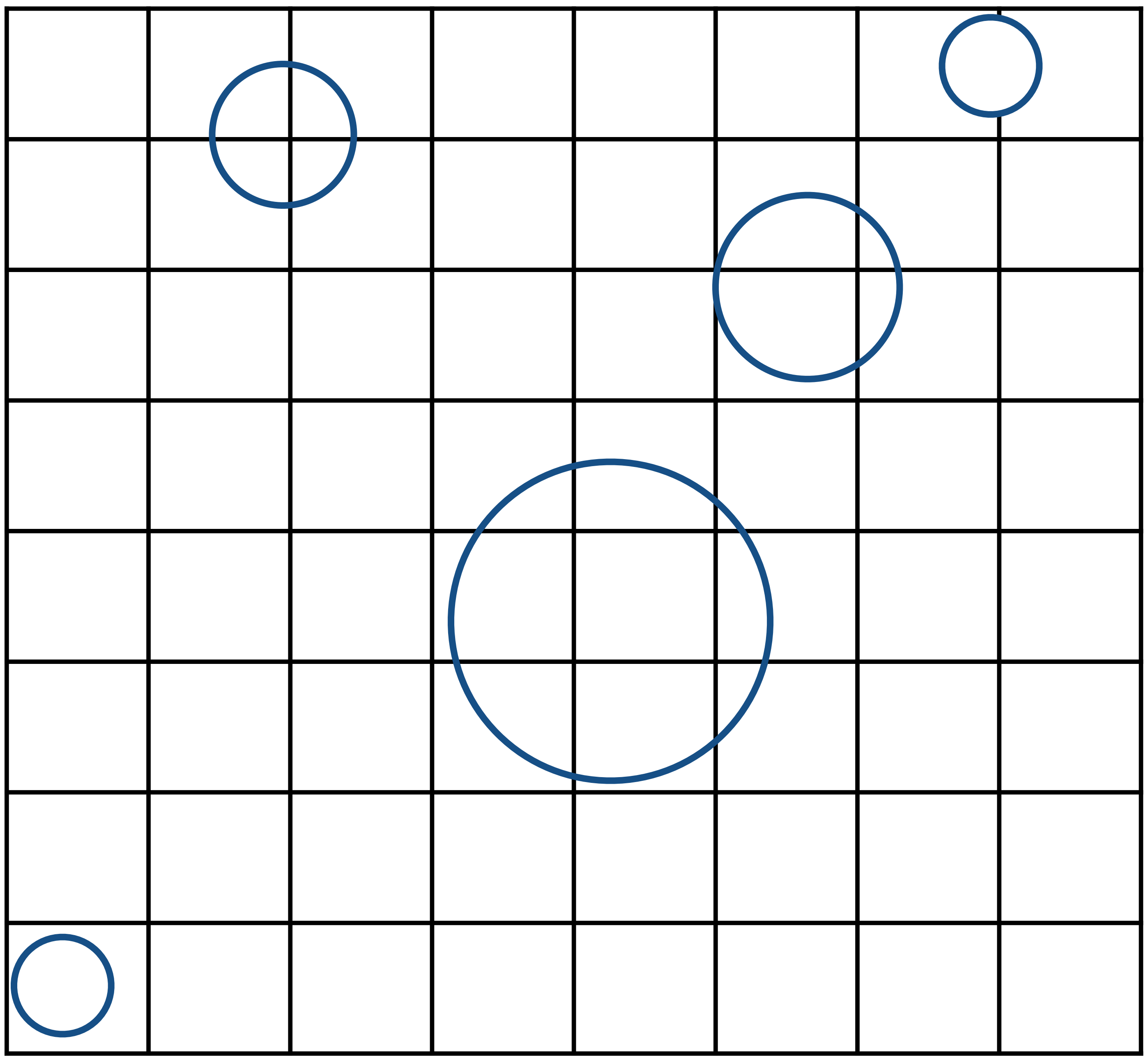
Uniform Spatial Partitions (Grids)

Preprocess – Build Acceleration Grid



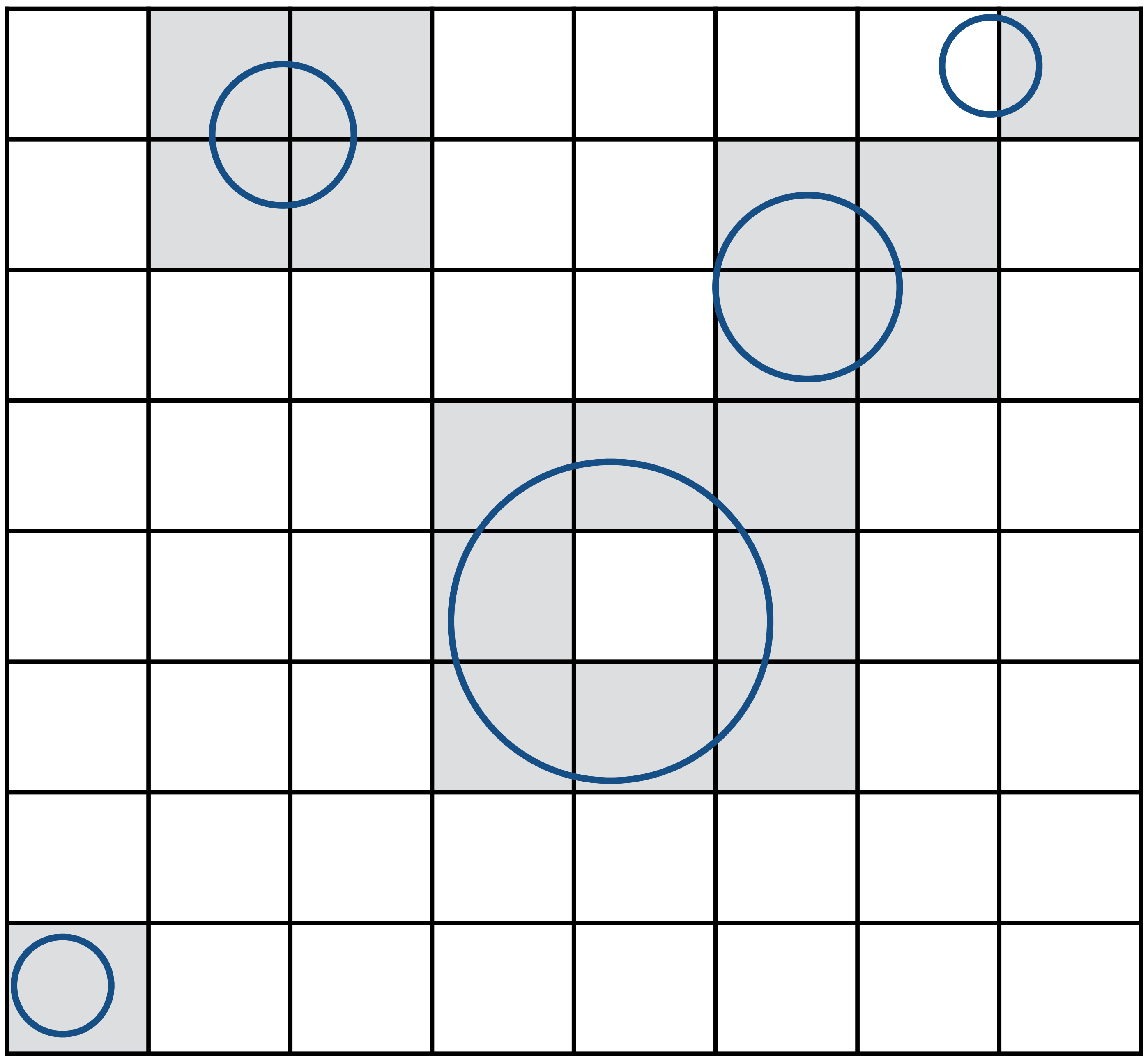
1. Find bounding box

Preprocess – Build Acceleration Grid



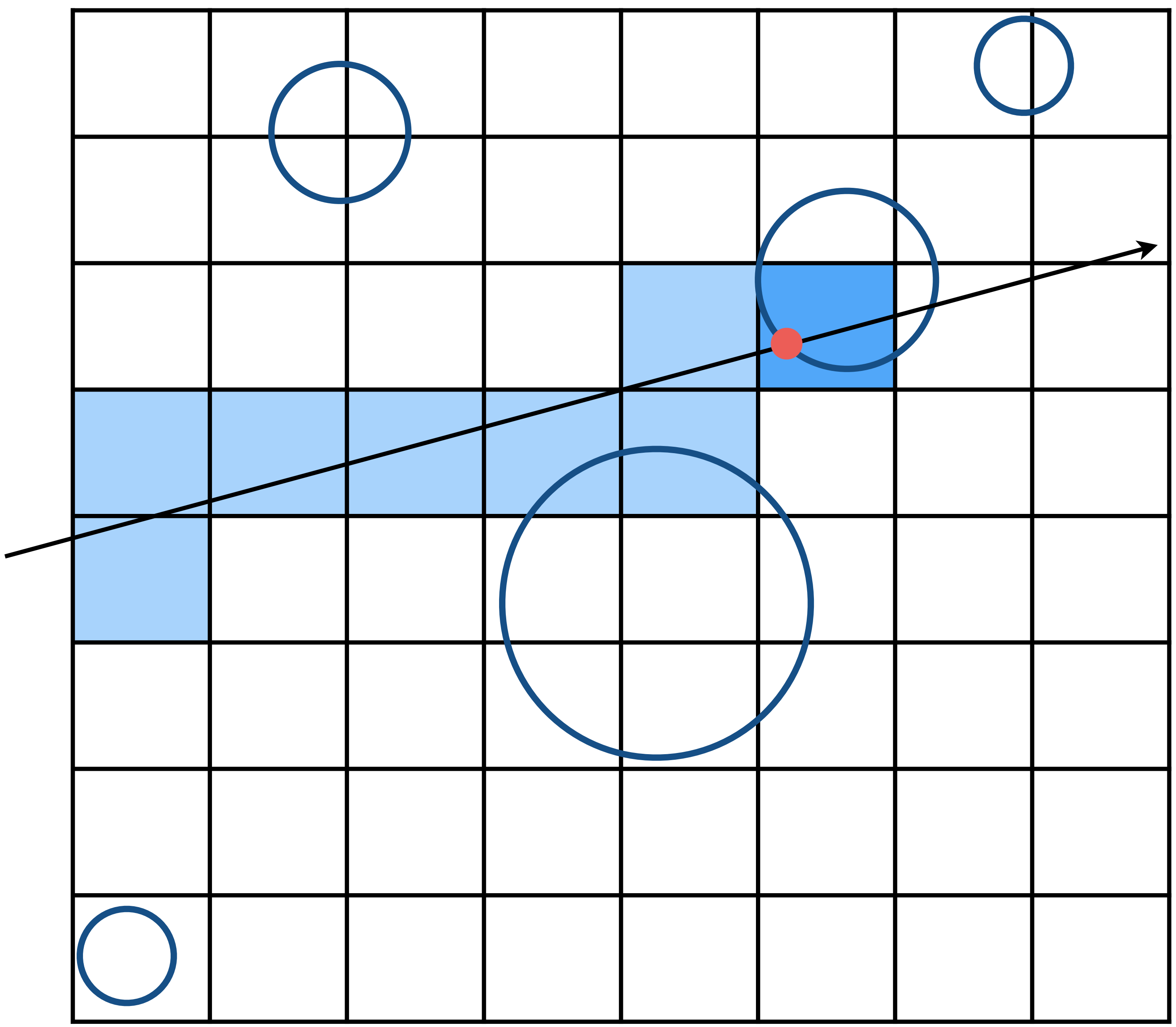
1. Find bounding box
2. Create grid

Preprocess – Build Acceleration Grid



1. Find bounding box
2. Create grid
3. Store each object in overlapping cells

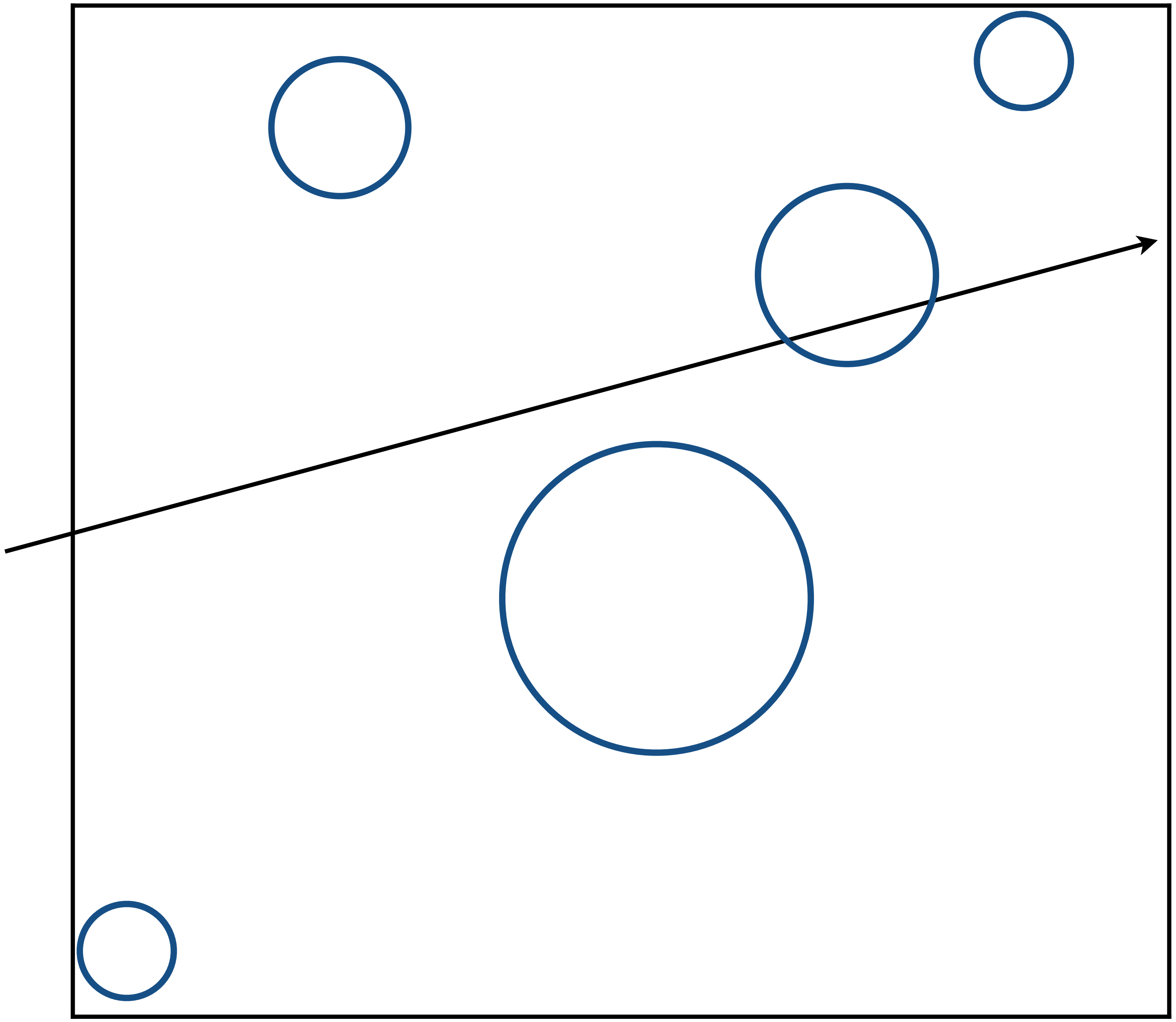
Ray-Scene Intersection



Step through grid in ray traversal order

For each grid cell
Test intersection
with all objects
stored at that cell

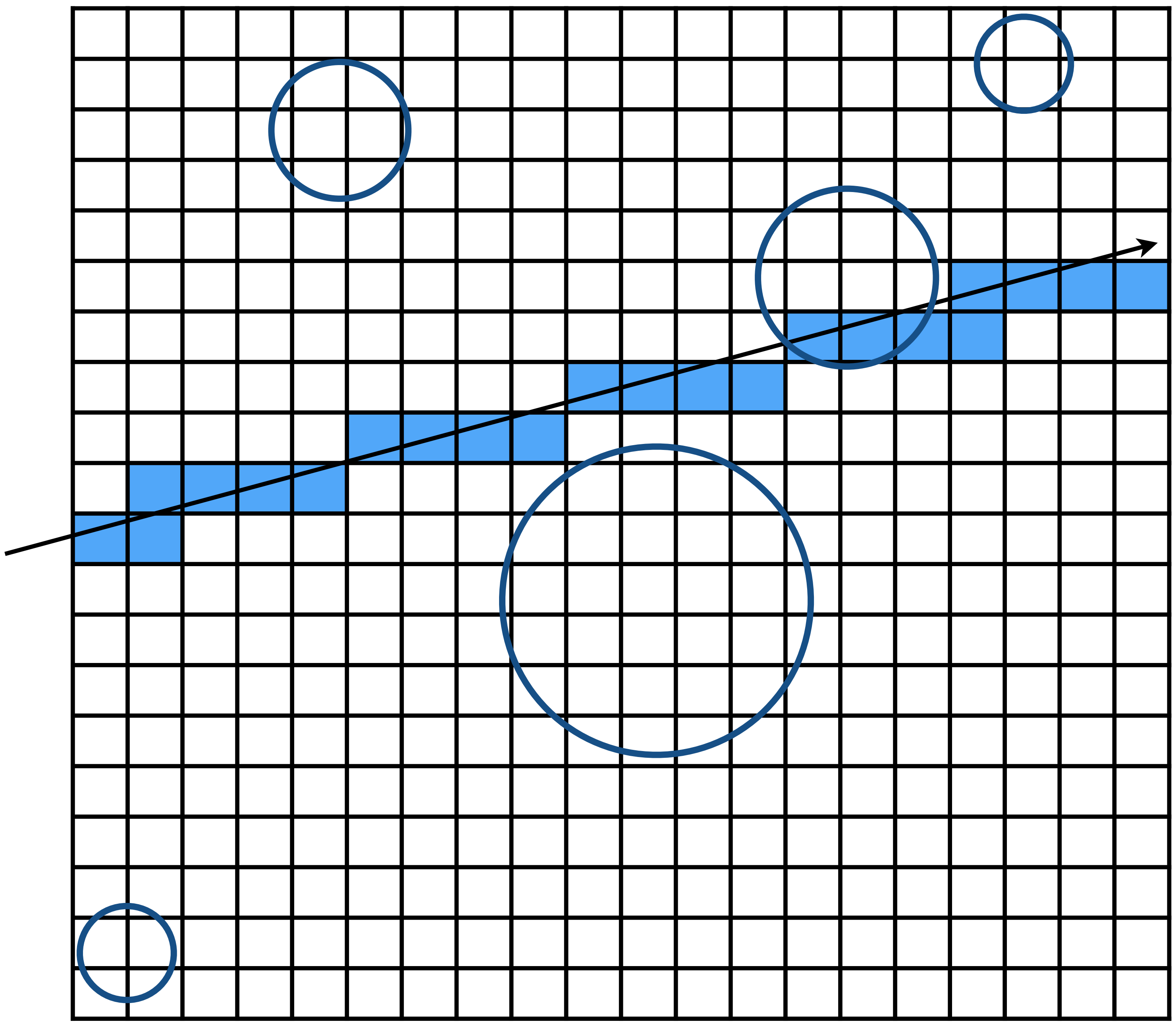
Grid Resolution?



One cell

- No speedup

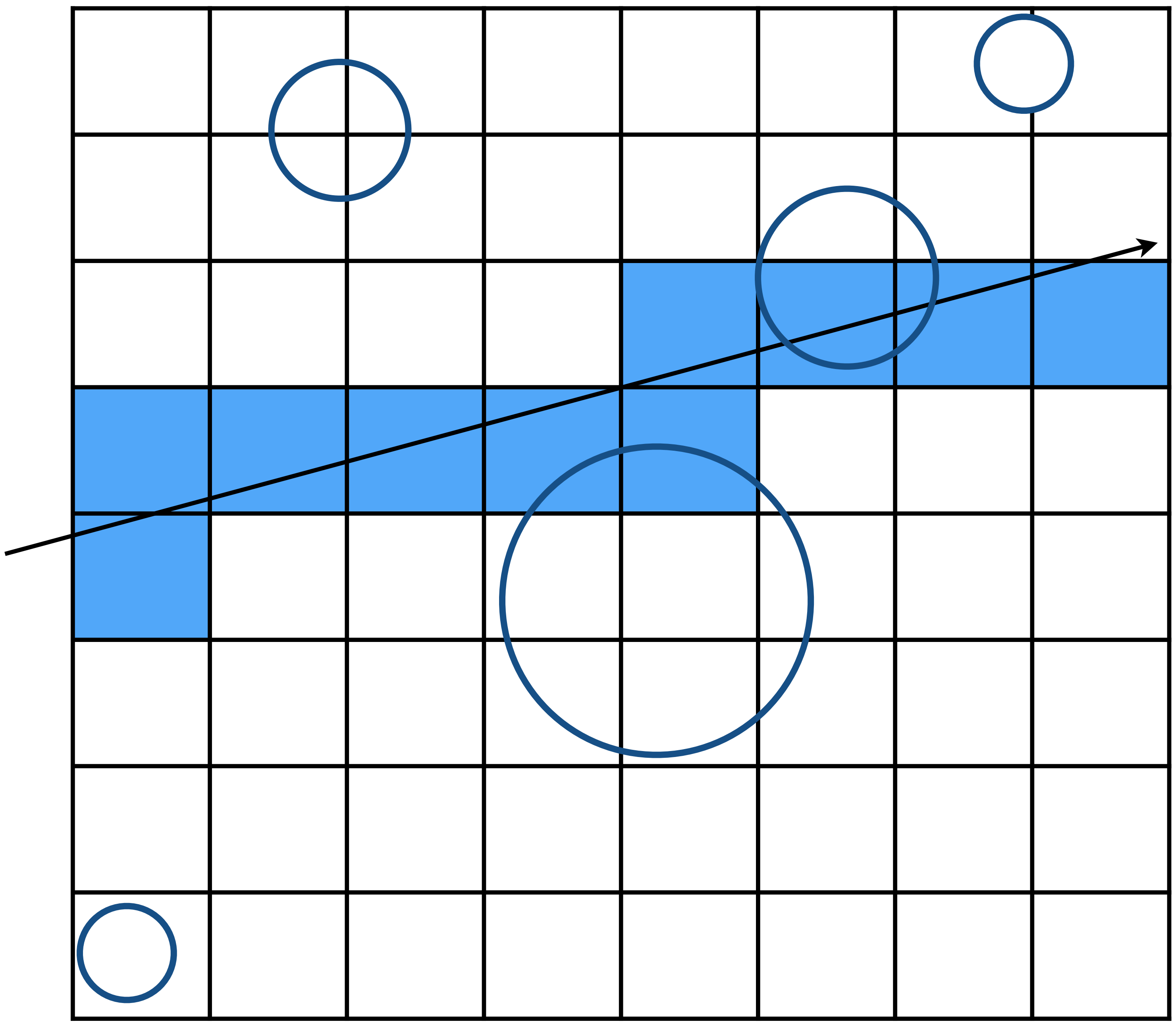
Grid Resolution?



Too many cells

- Inefficiency due to extraneous grid traversal

Grid Resolution?



Heuristic:

- #cells = C * #objs
- C ≈ 27 in 3D

Uniform Grids – When They Work Well

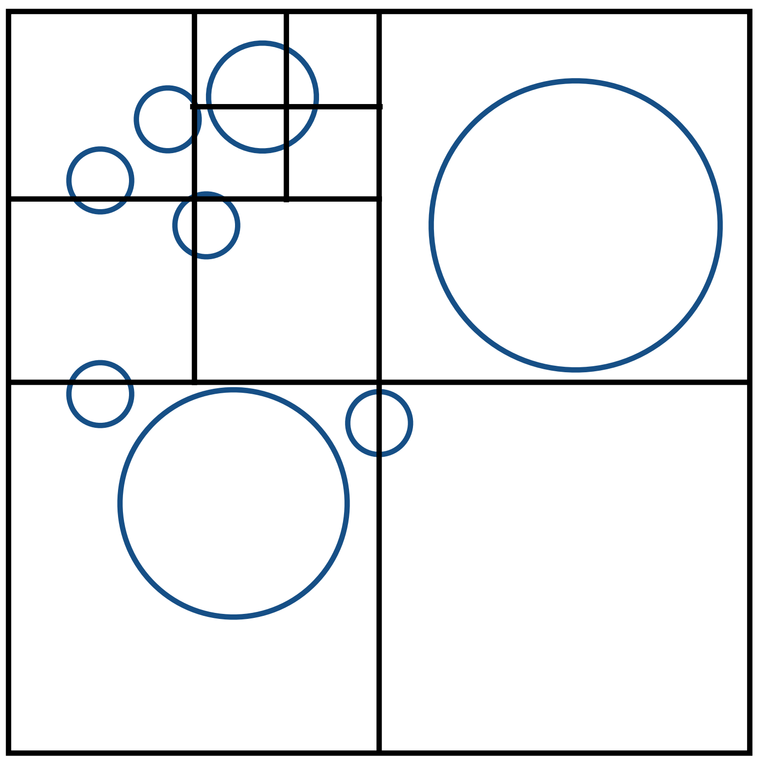
Grids work well on large collections of objects that are distributed evenly in size and space

Uniform Grids – When They Fail

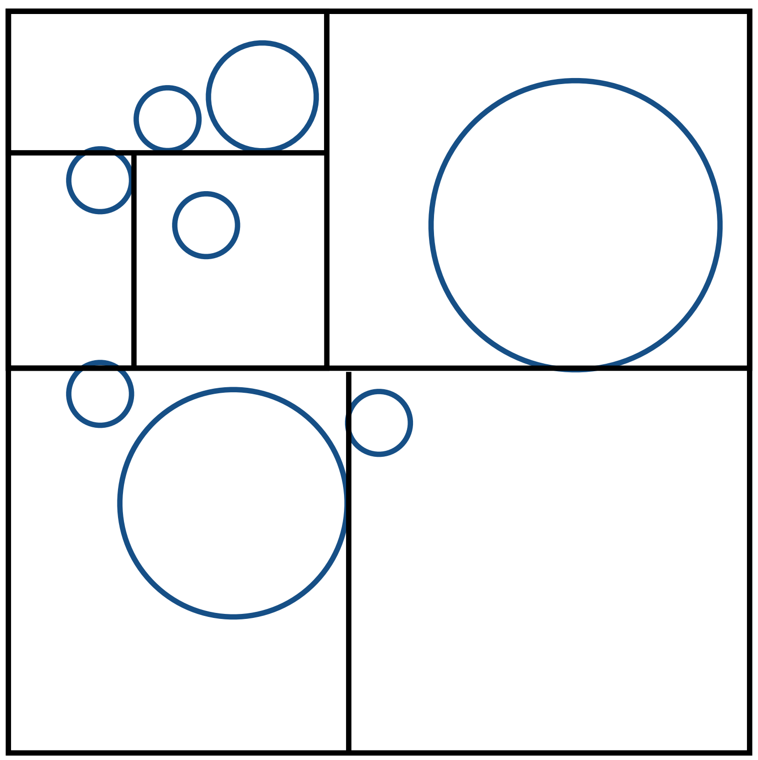
"Teapot in a stadium" problem

Spatial Partitions

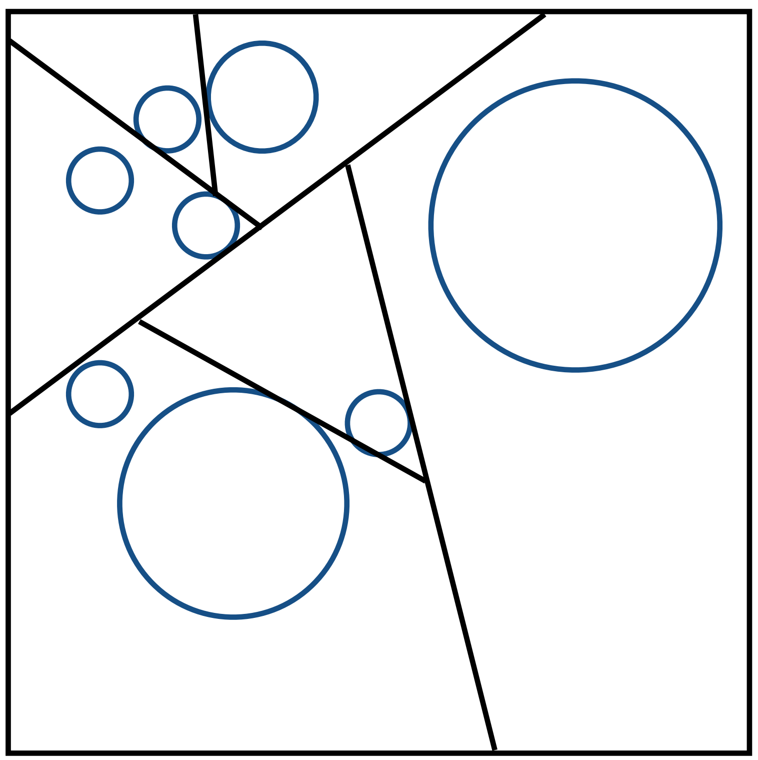
Spatial Partitioning Examples



Oct-Tree



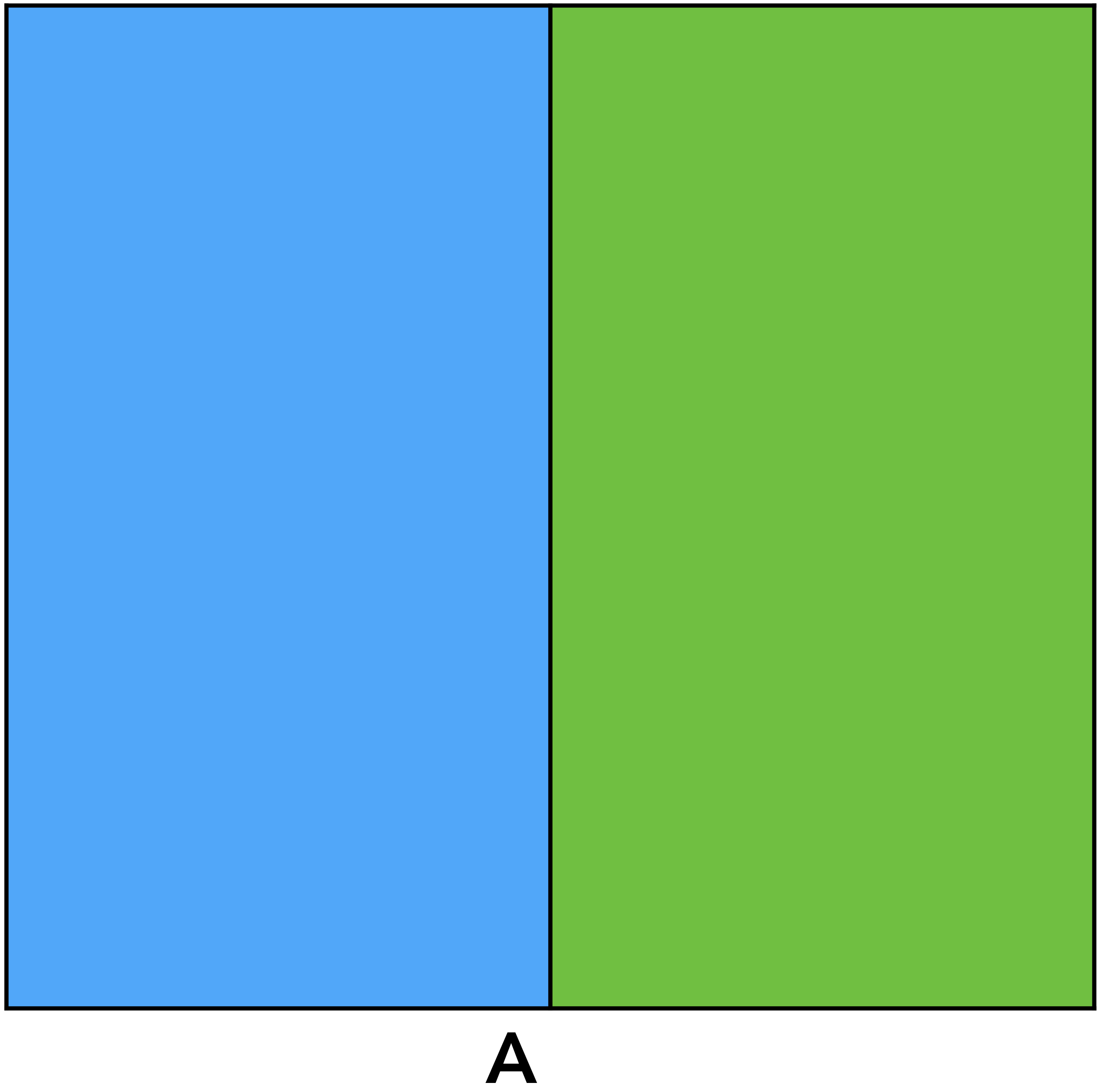
KD-Tree

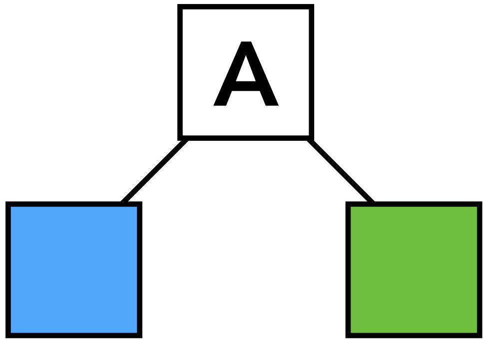


BSP-Tree

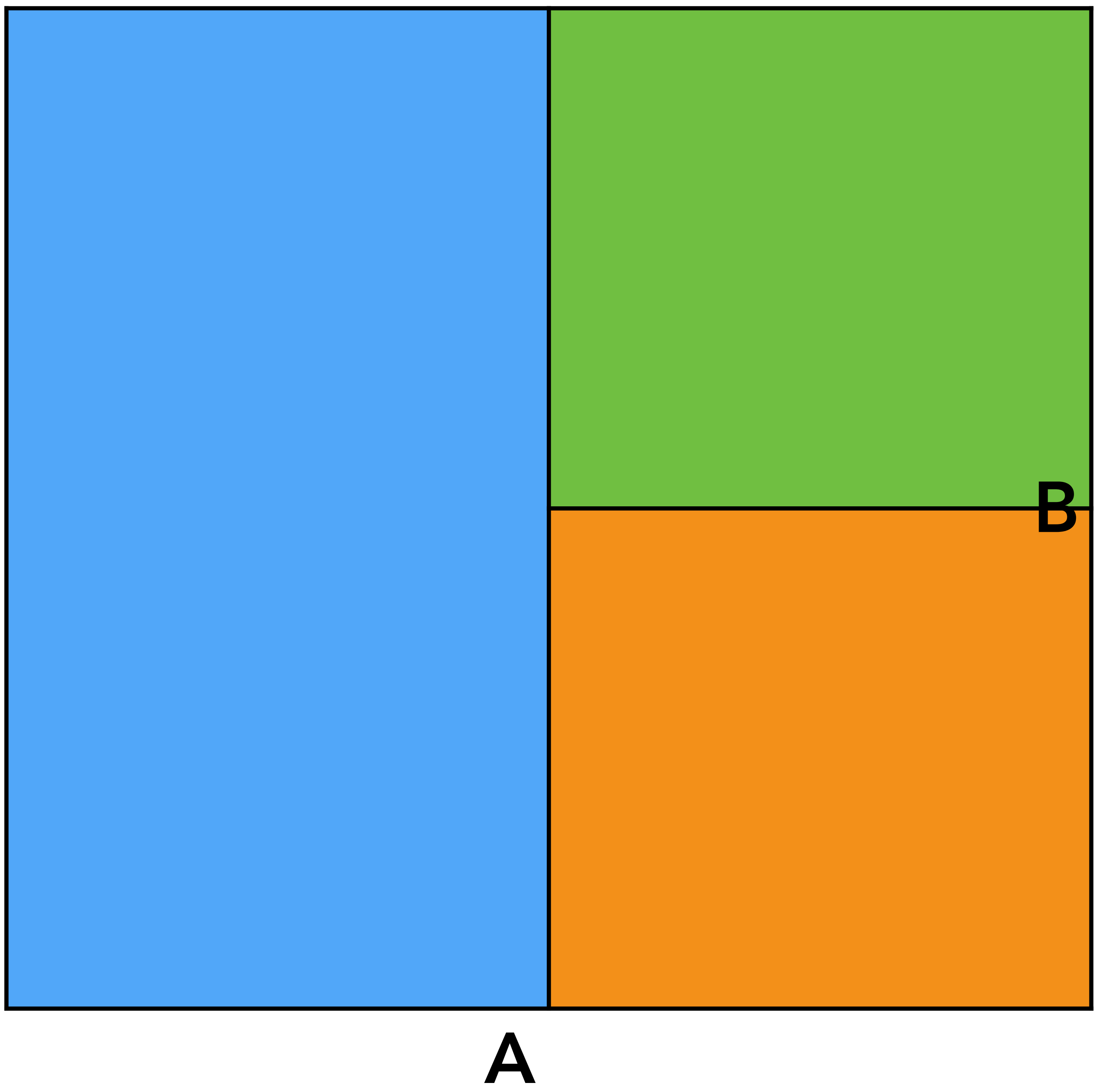
Note: you could have these in both 2D and 3D. In lecture we will illustrate principles in 2D.

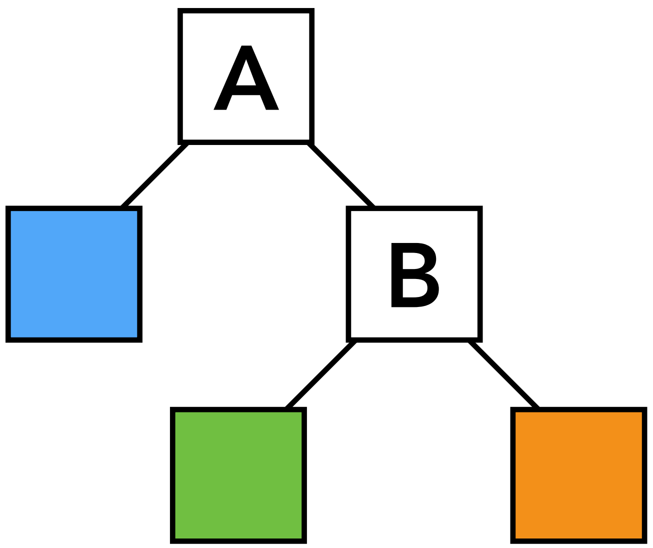
KD-Tree Pre-Processing



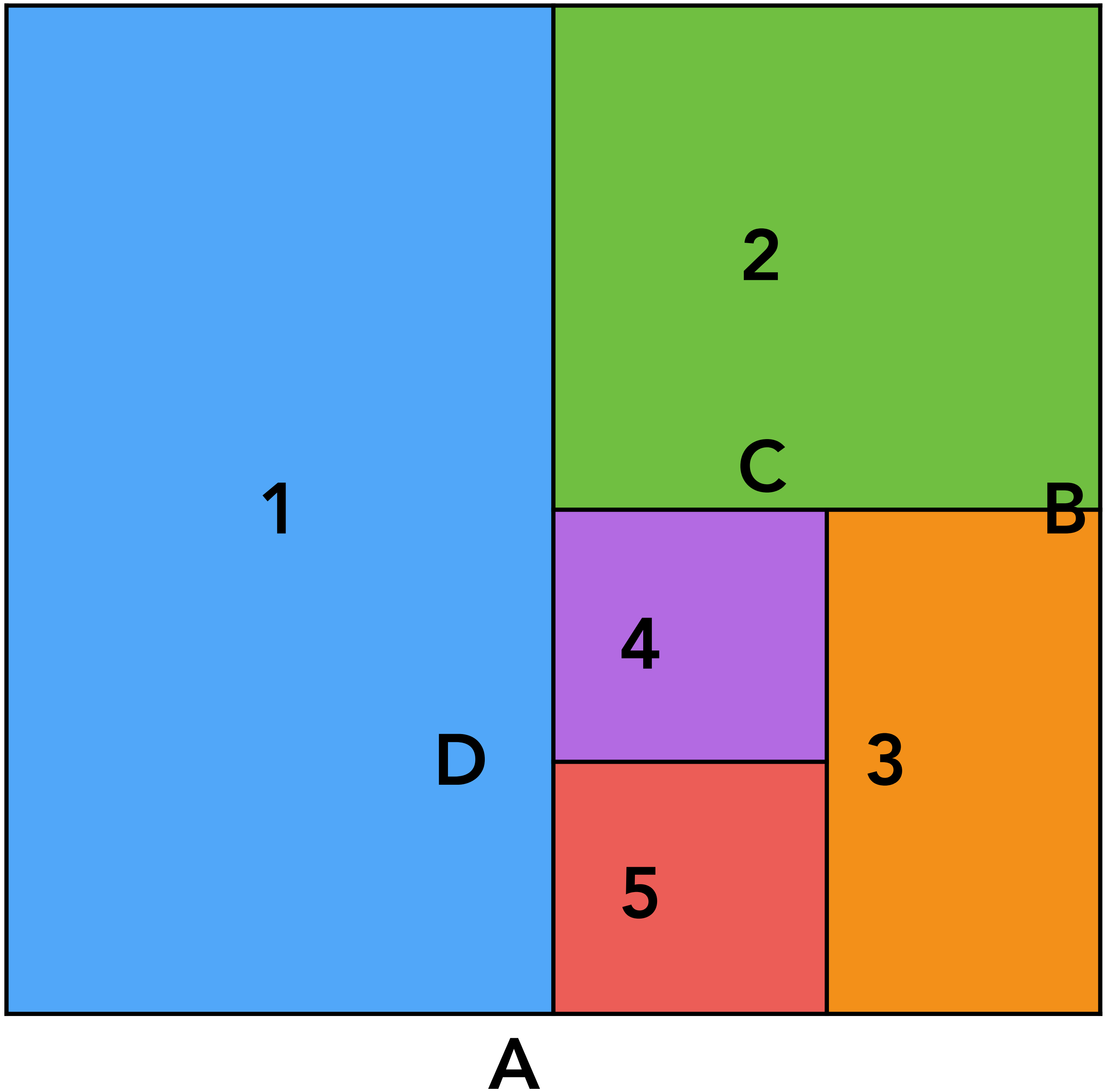


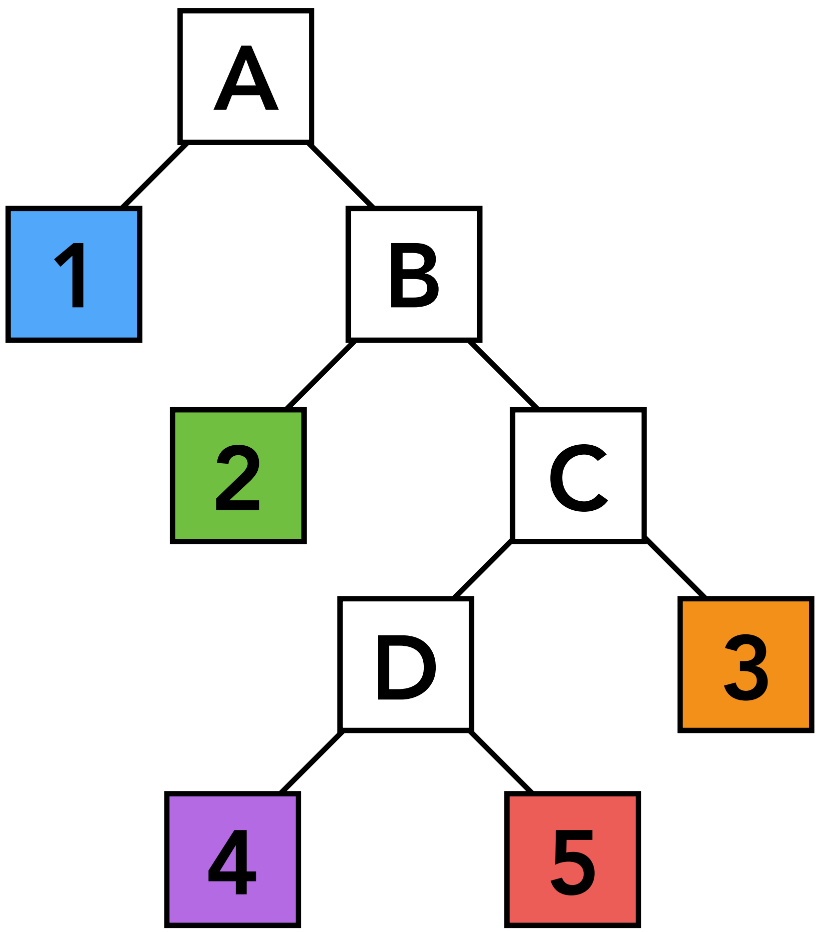
KD-Tree Pre-Processing





KD-Tree Pre-Processing





Note: also subdivide nodes 1 and 2, etc.

Data Structure for KD-Trees

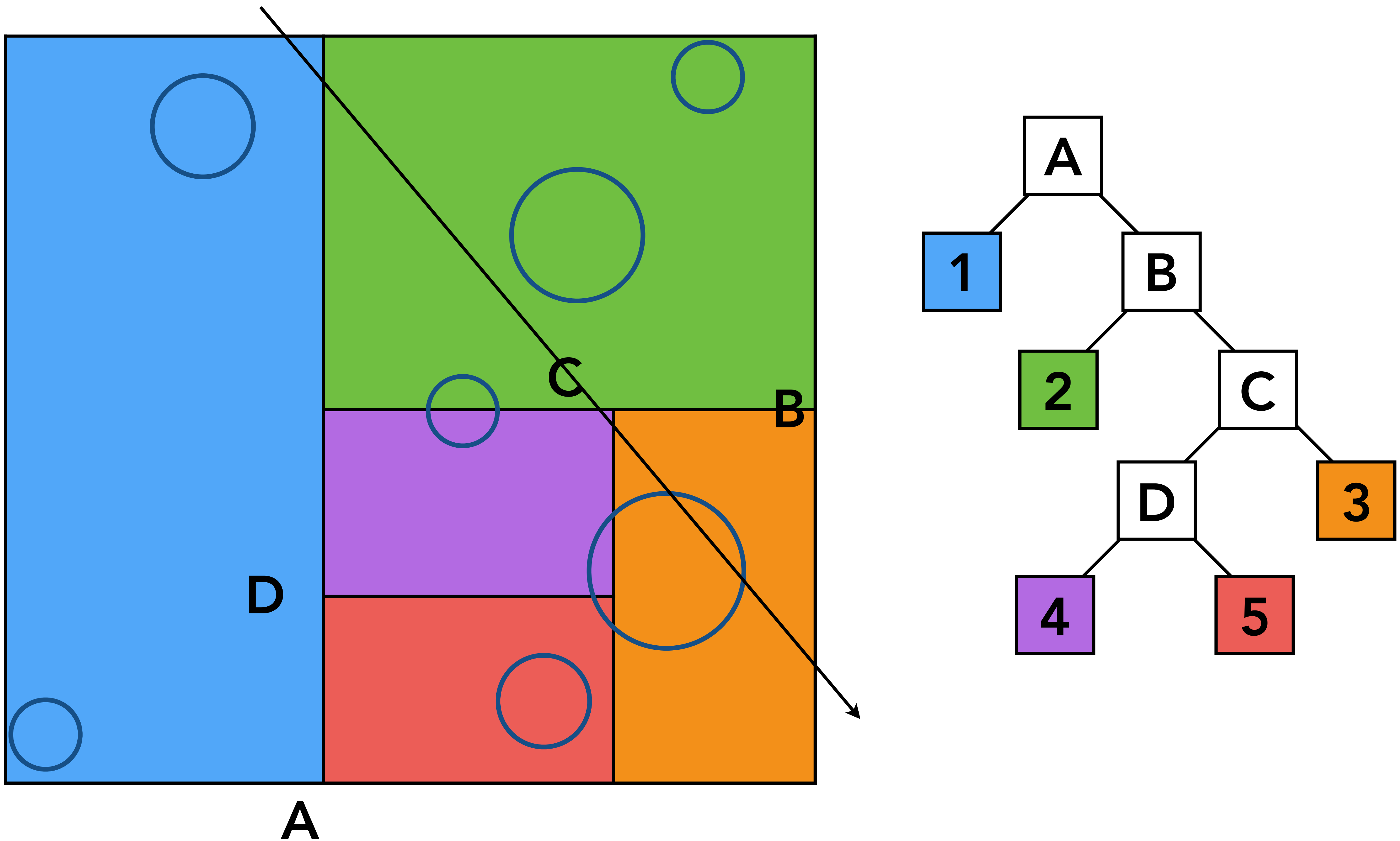
Internal nodes store

- split axis: x-, y-, or z-axis
- split position: coordinate of split plane along axis
- children: pointers to child nodes
- **No objects are stored in internal nodes**

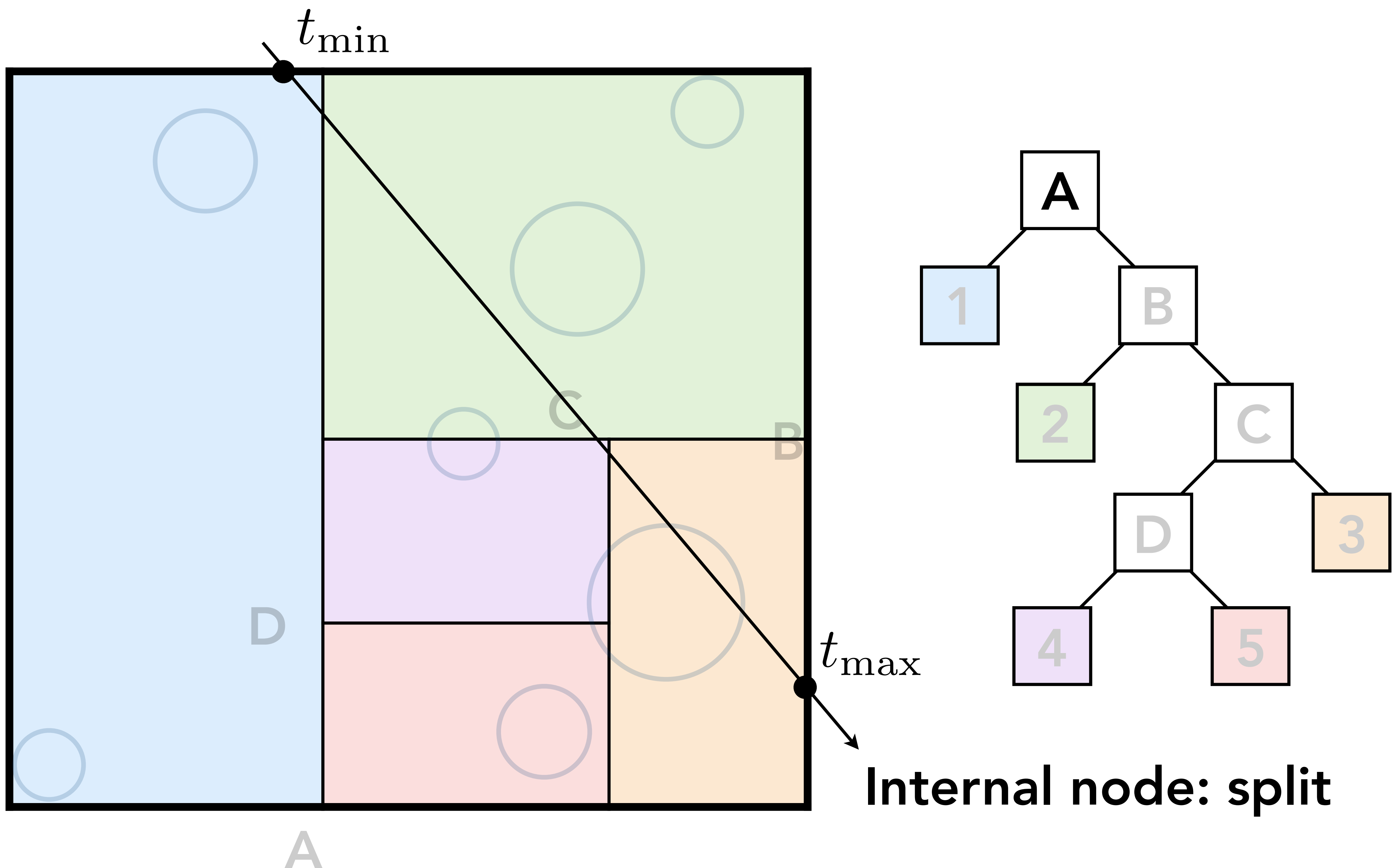
Leaf nodes store

- list of objects

Traversing a KD-Tree

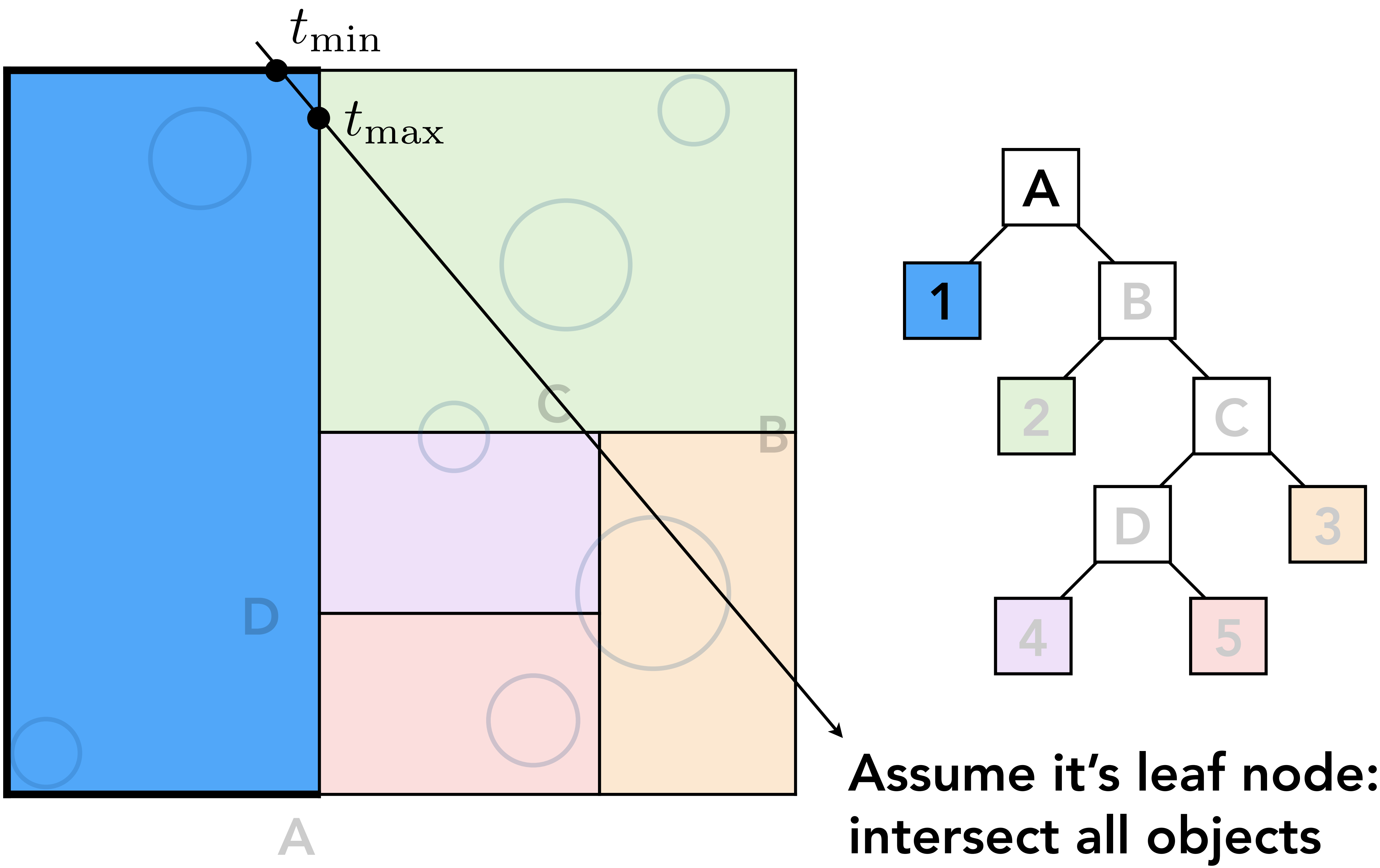


Traversing a KD-Tree



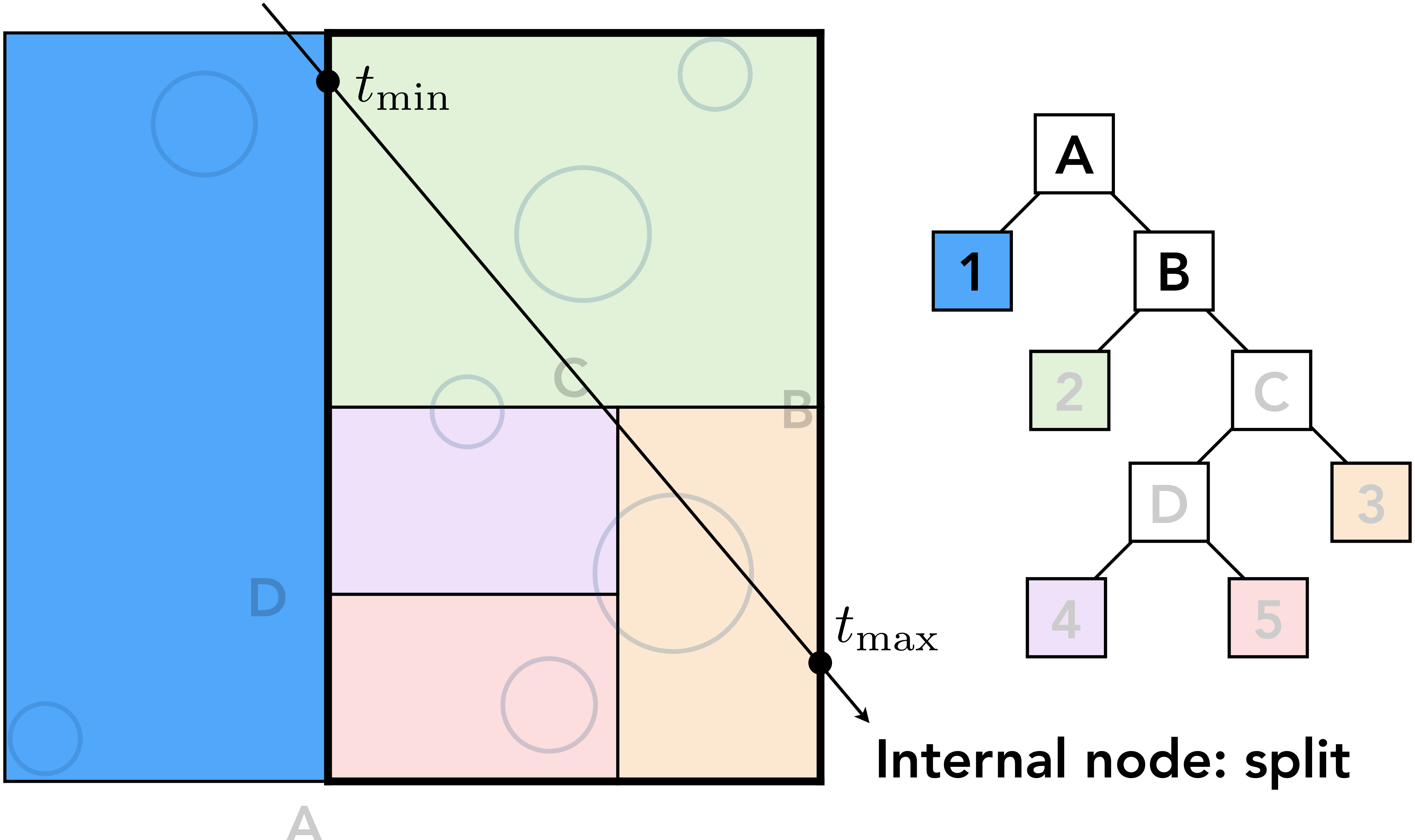
Internal node: split

Traversing a KD-Tree



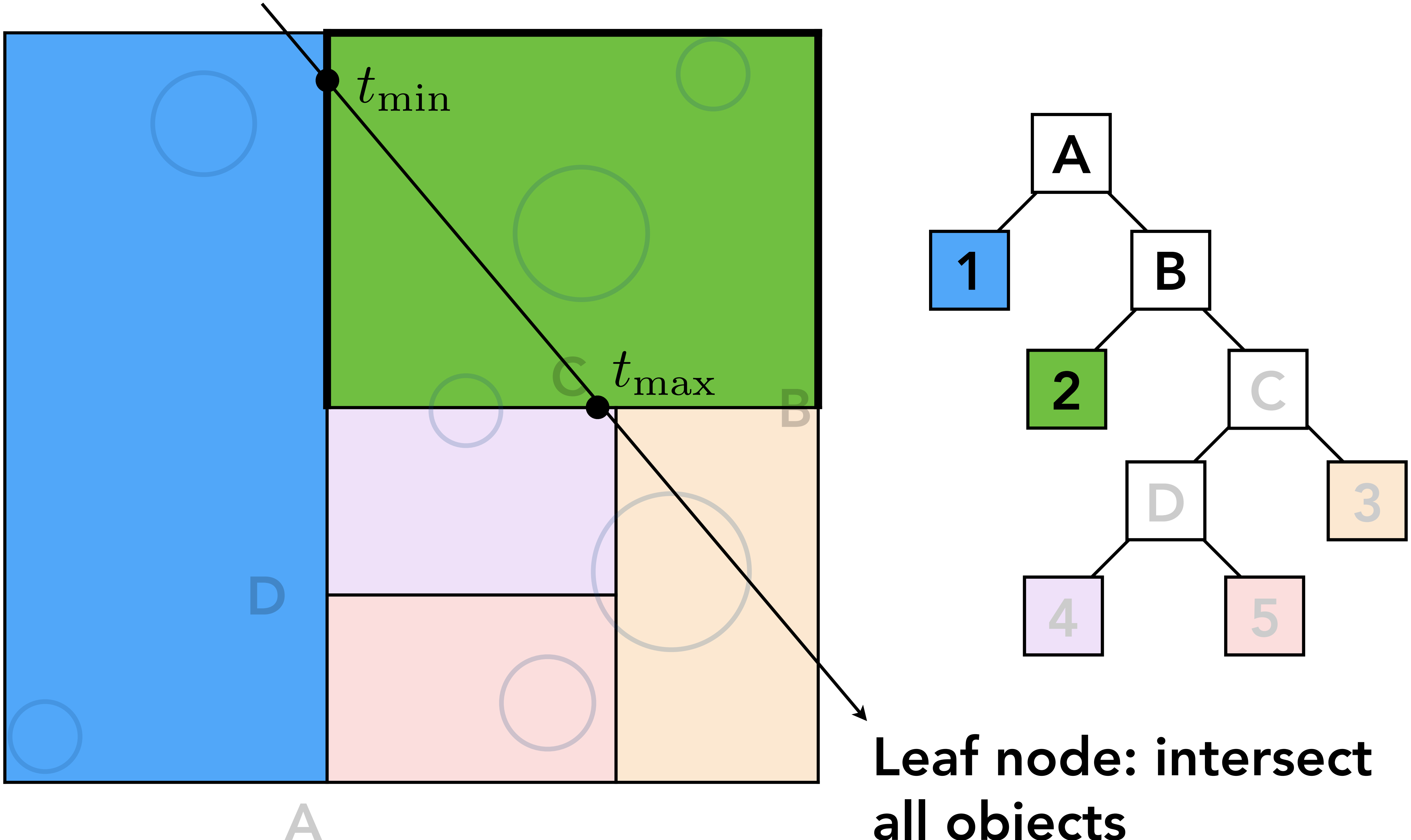
Assume it's leaf node: intersect all objects

Traversing a KD-Tree



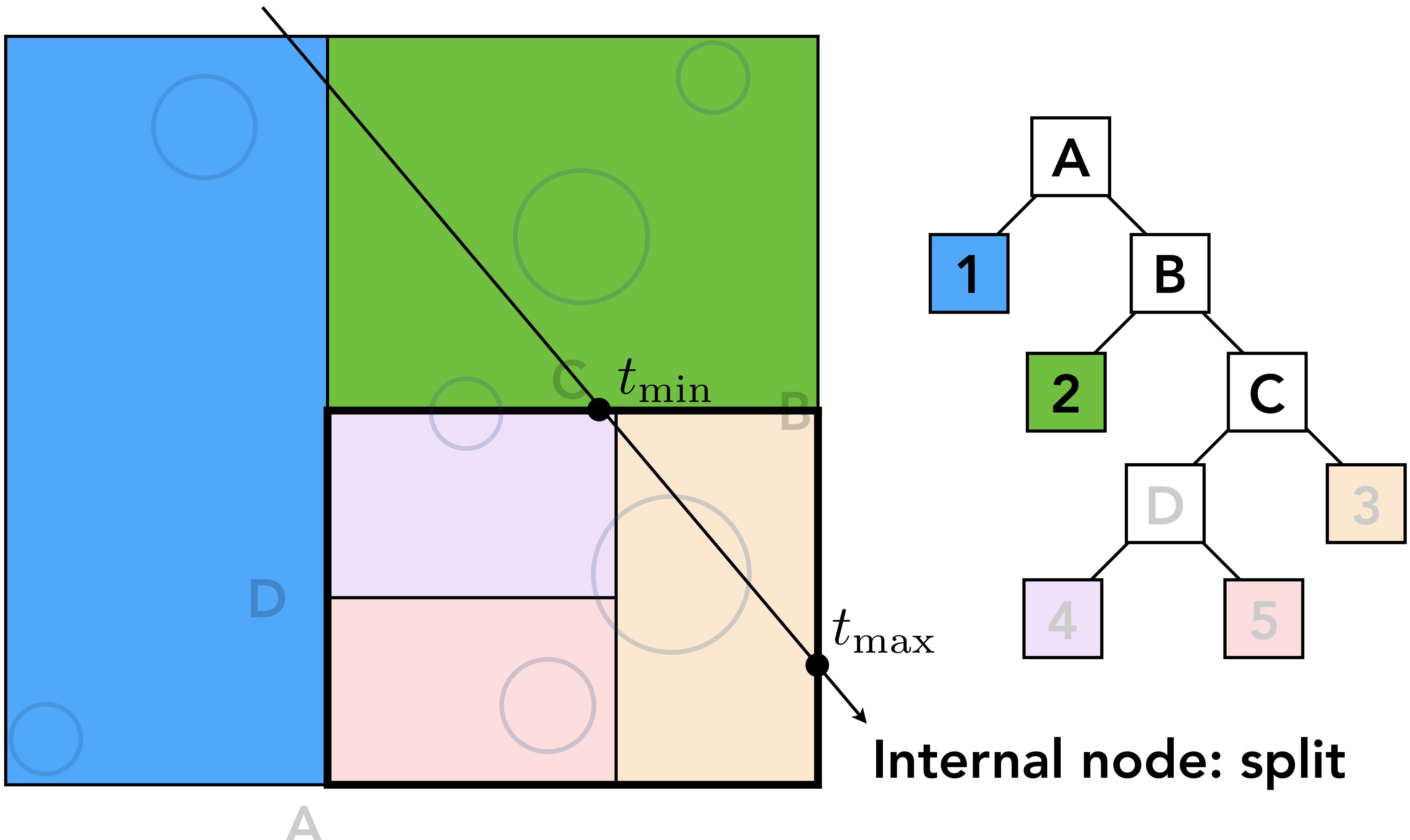
Internal node: split

Traversing a KD-Tree



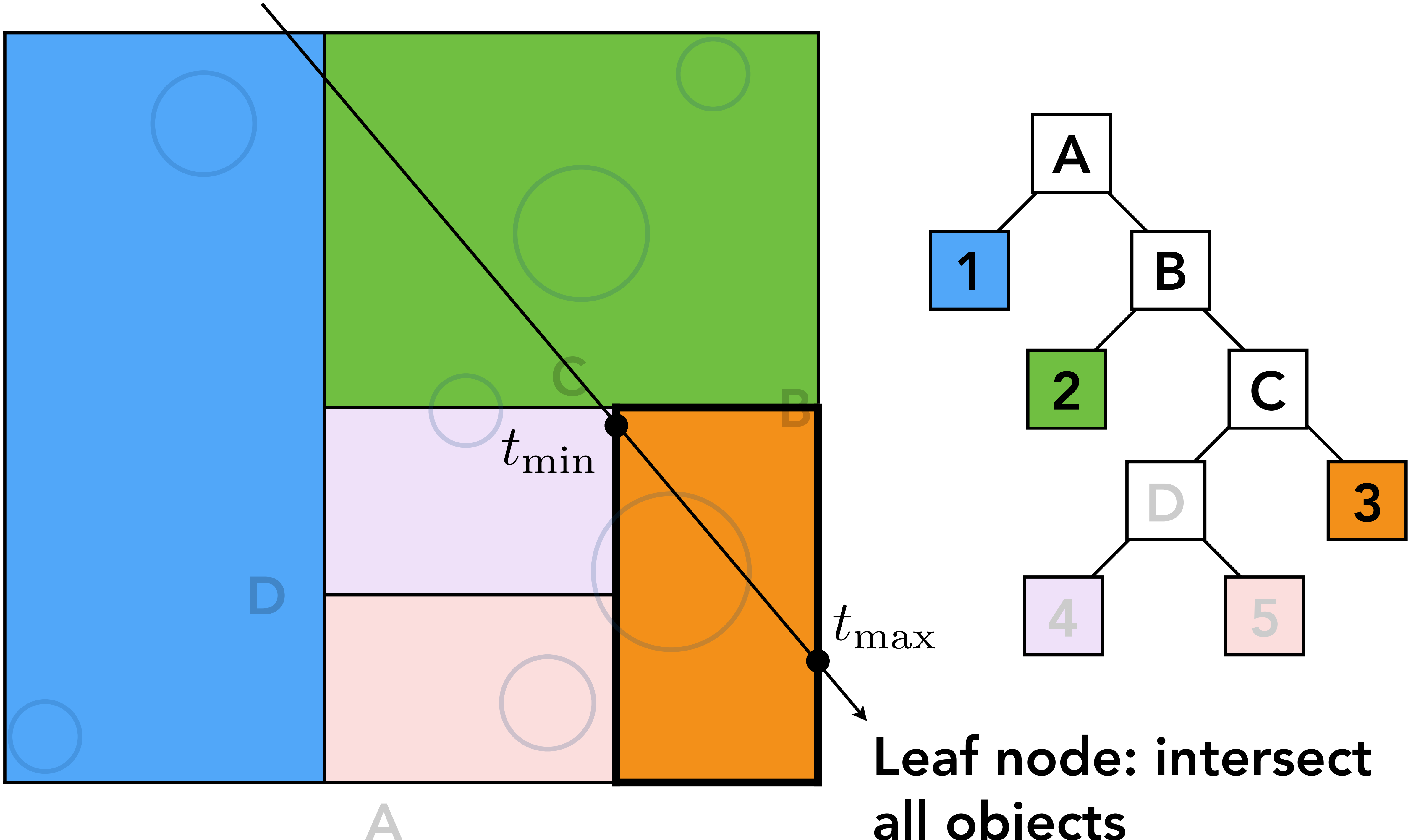
Leaf node: intersect all objects

Traversing a KD-Tree



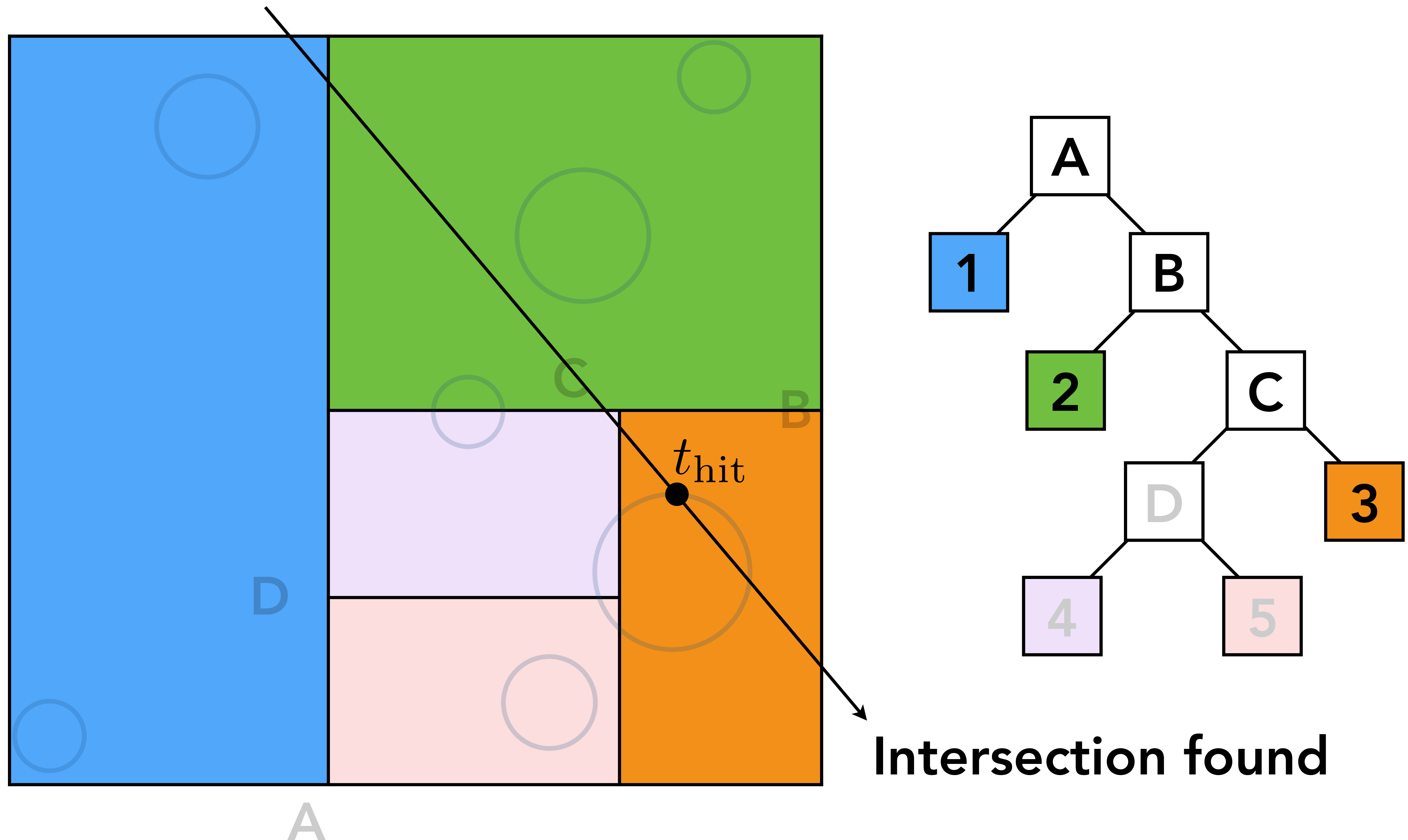
Internal node: split

Traversing a KD-Tree



Leaf node: intersect all objects

Traversing a KD-Tree

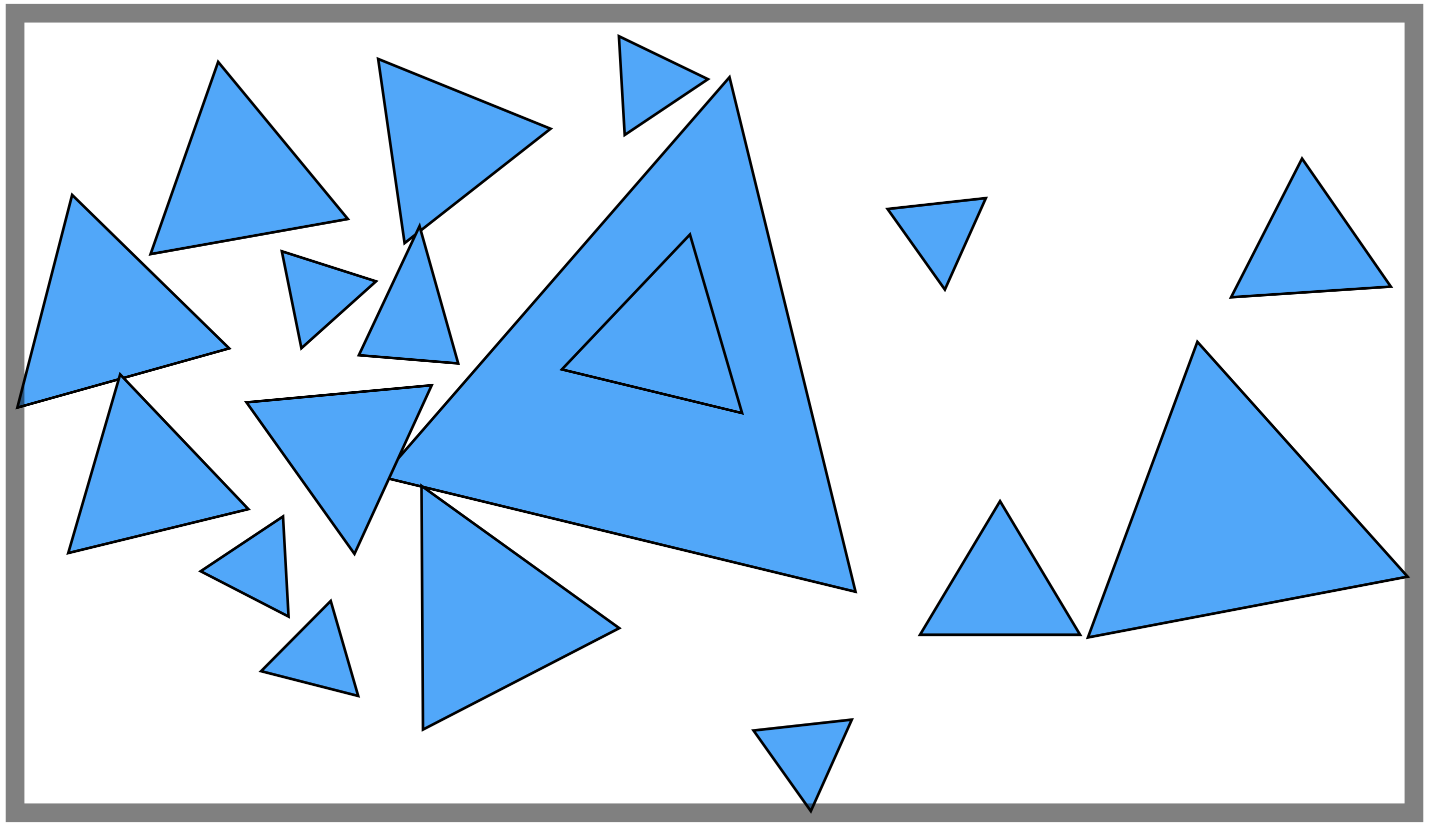


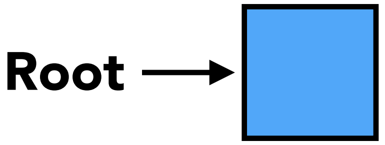
Intersection found

Object Partitions &

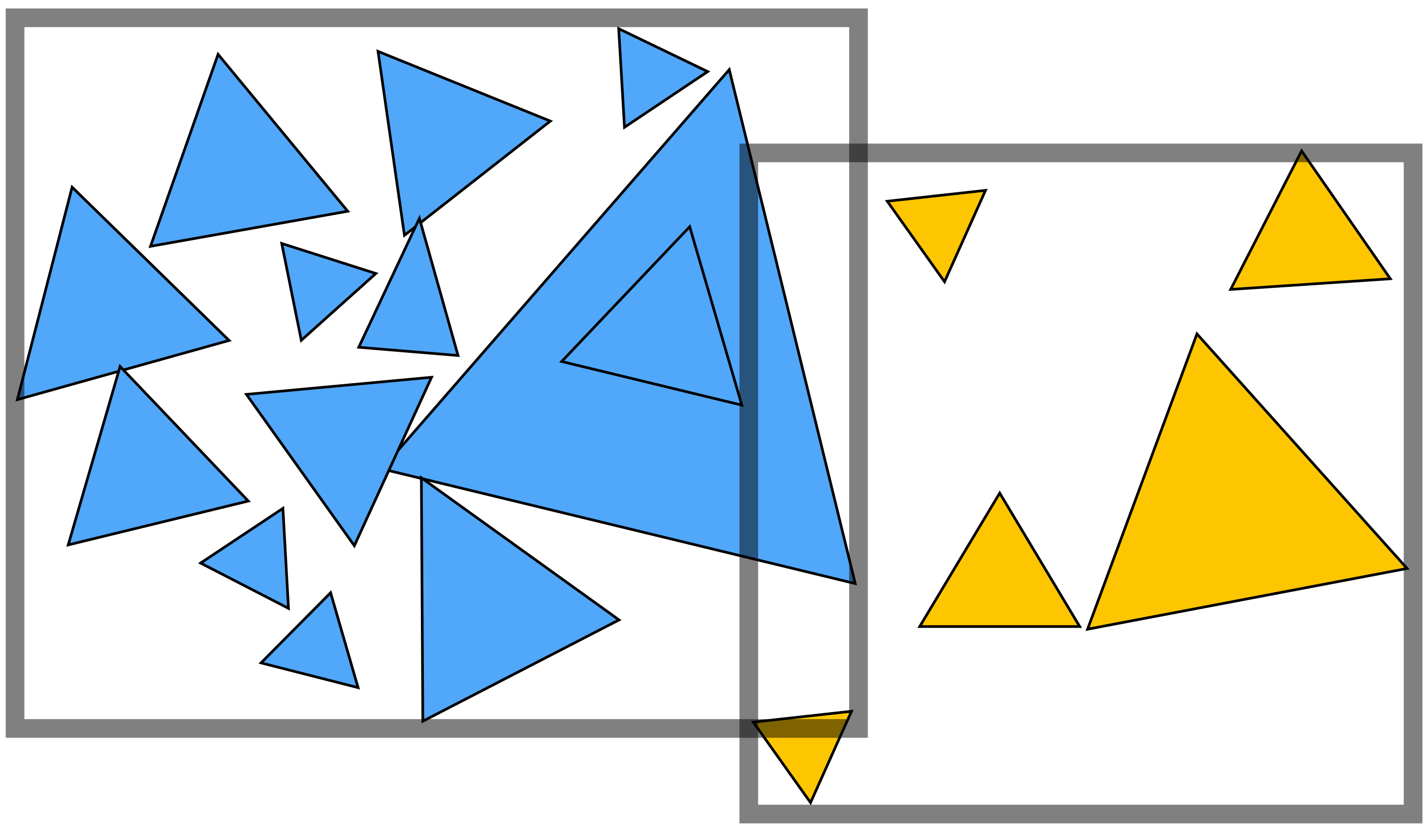
Bounding Volume Hierarchy (BVH)

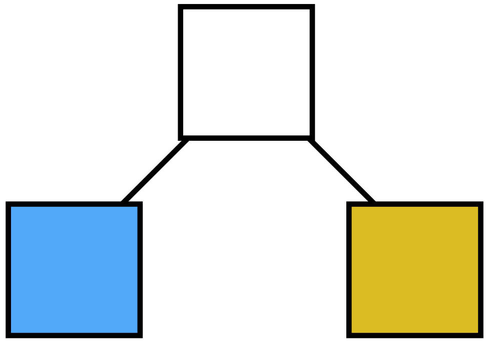
Bounding Volume Hierarchy (BVH)



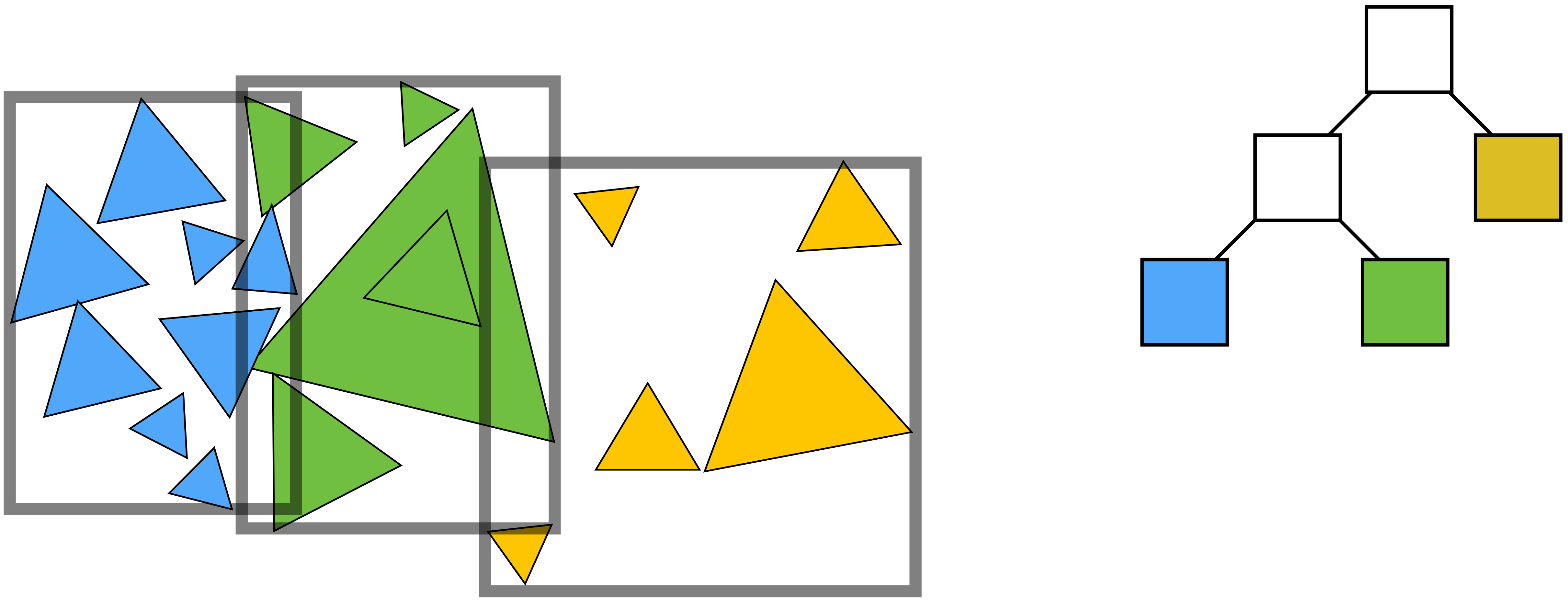


Bounding Volume Hierarchy (BVH)

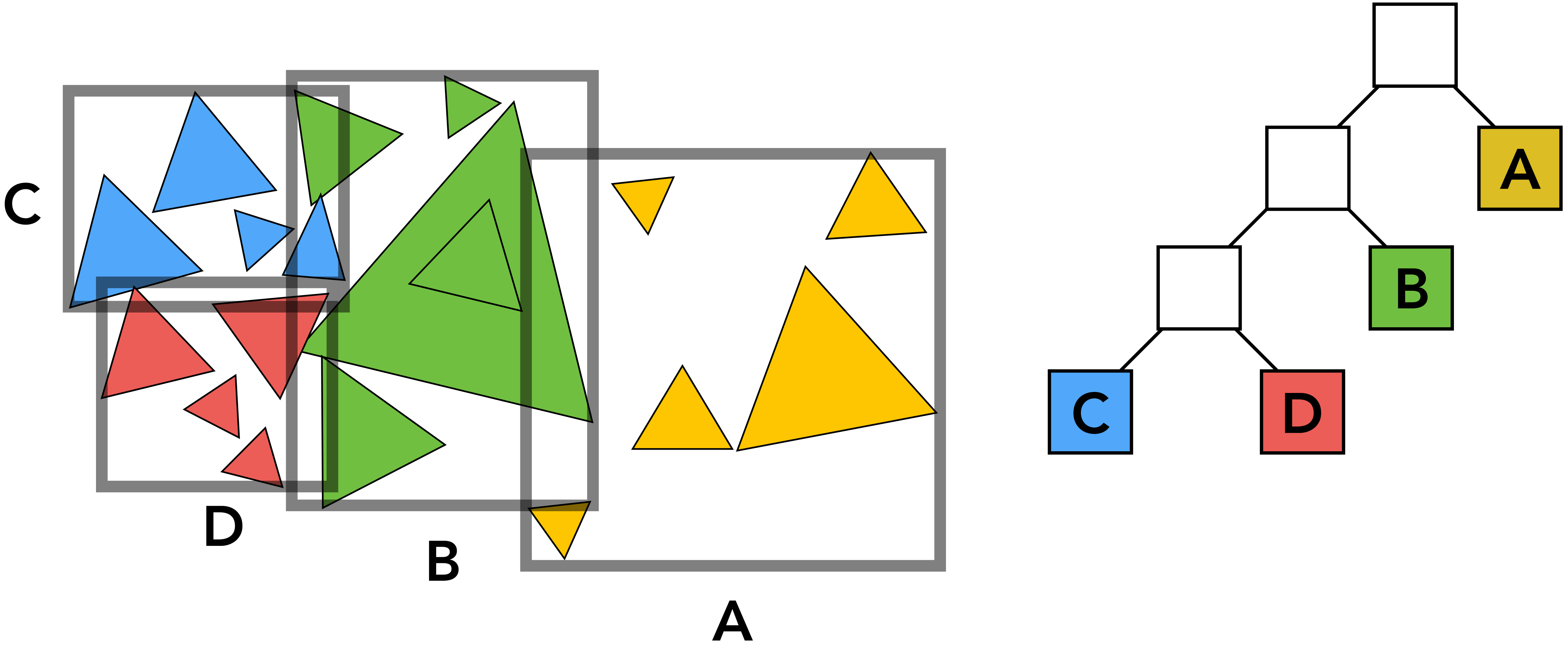




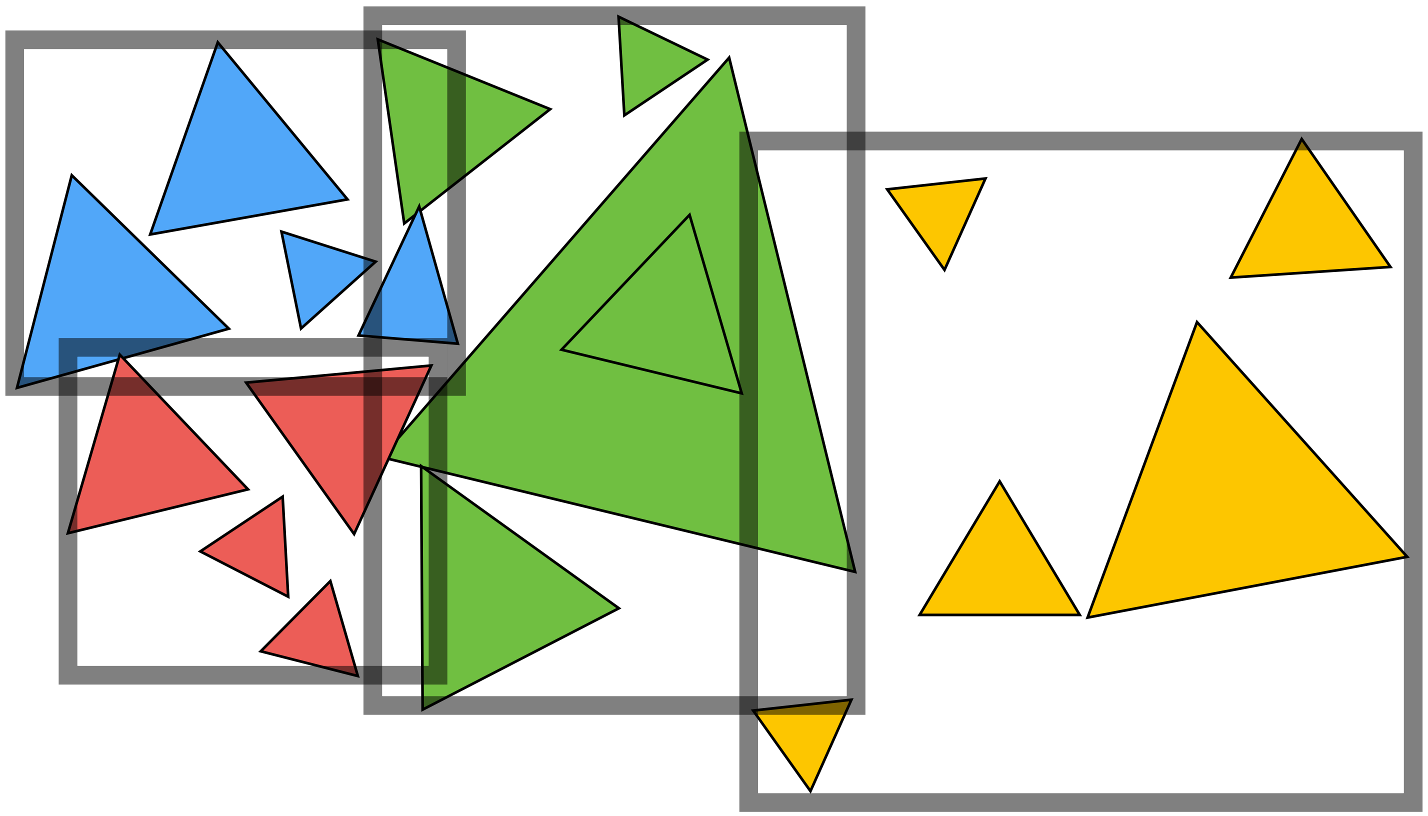
Bounding Volume Hierarchy (BVH)



Bounding Volume Hierarchy (BVH)



Summary: Building BVHs



- Find bounding box
- Recursively split set of objects in two subsets
- **Recompute** the bounding box of the subsets
- Stop when necessary
- Store objects in each leaf node

Building BVHs

How to subdivide a node?

- Choose a dimension to split
- Heuristic #1: Always choose the longest axis in node
- Heuristic #2: Split node at location of **median** object

Termination criteria?

- Heuristic: stop when node contains few elements (e.g. 5)

Data Structure for BVHs

Internal nodes store

- Bounding box
- Children: pointers to child nodes

Leaf nodes store

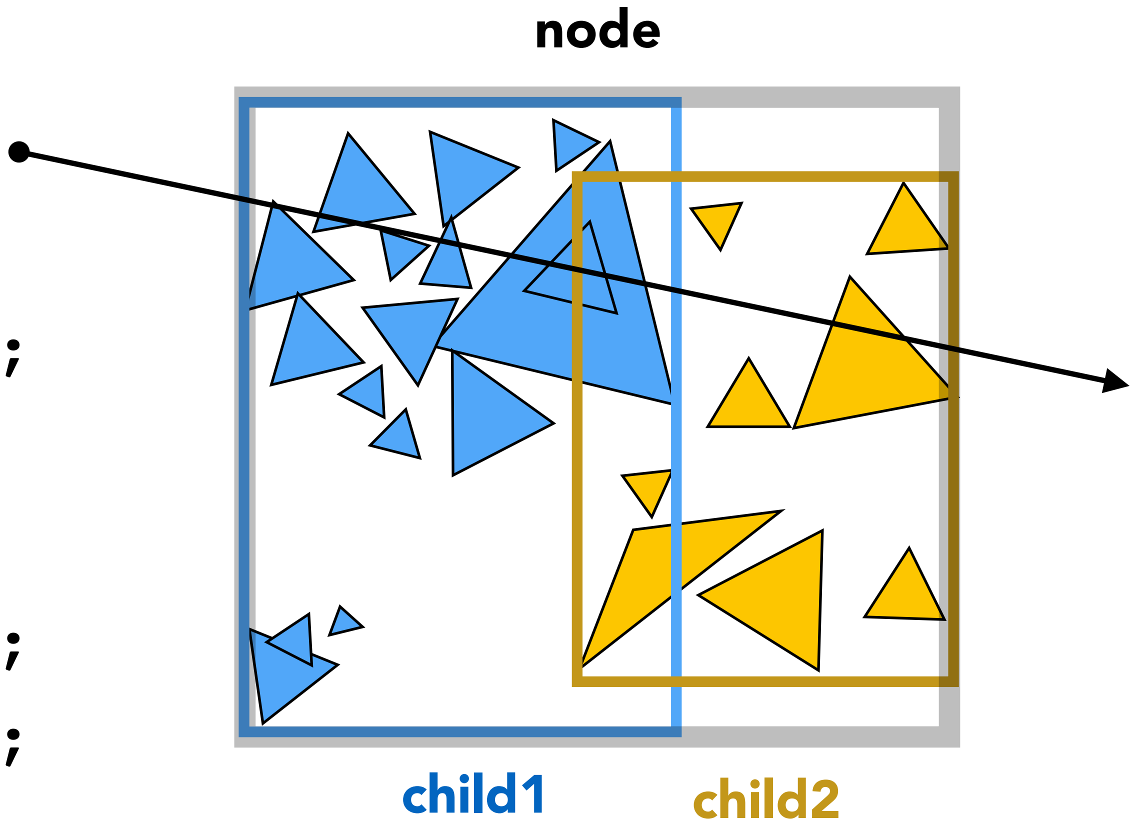
- Bounding box
- List of objects

Nodes represent subset of primitives in scene

- All objects in subtree

BVH Traversal

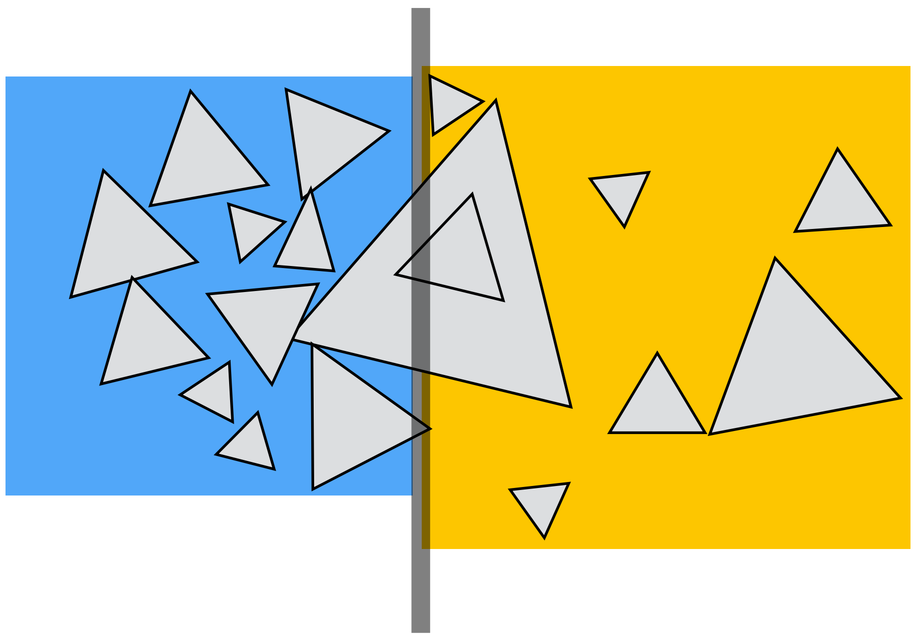
```
Intersect(Ray ray, BVH node) {
  if (ray misses node.bbox) return;

  if (node is a leaf node)
     test intersection with all objs;
     return closest intersection;

  hit1 = Intersect(ray, node.child1);
  hit2 = Intersect(ray, node.child2);

  return the closer of hit1, hit2;
}
```



Spatial vs Object Partitions

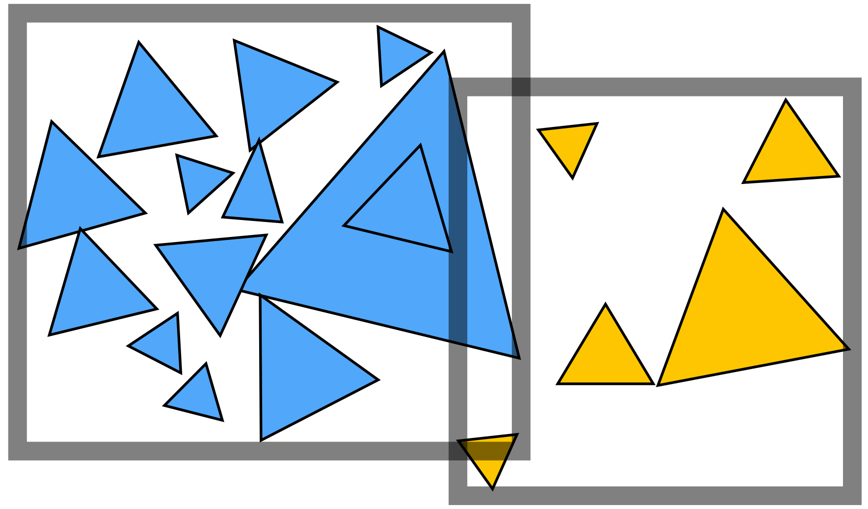
Spatial partition (e.g.KD-tree)

- Partition space into non-overlapping regions
- An object can be contained in multiple regions



Object partition (e.g. BVH)

- Partition set of objects into disjoint subsets
- Bounding boxes for each set may overlap in space



Today

- Using AABBs to accelerate ray tracing
  - Uniform grids
  - Spatial partitions
- Basic radiometry (辐射度量学)
  - Advertisement: new topics from now on, scarcely covered in other graphics courses

Radiometry — Motivation

Observation

- In assignment 3, we implement the Blinn-Phong model
- Light intensity I is 10, for example
- But 10 what?

Do you think Whitted style ray tracing gives you CORRECT results?

All the answers can be found in radiometry

- Also the basics of “Path Tracing”

Radiometry

Measurement system and units for illumination

Accurately measure the spatial properties of light

- New terms: Radiant flux, intensity, irradiance, radiance

Perform lighting calculations **in a physically correct manner**

My personal way of learning things:

- WHY, WHAT, then HOW

Radiant Energy and Flux (Power)

Radiant Energy and Flux (Power)

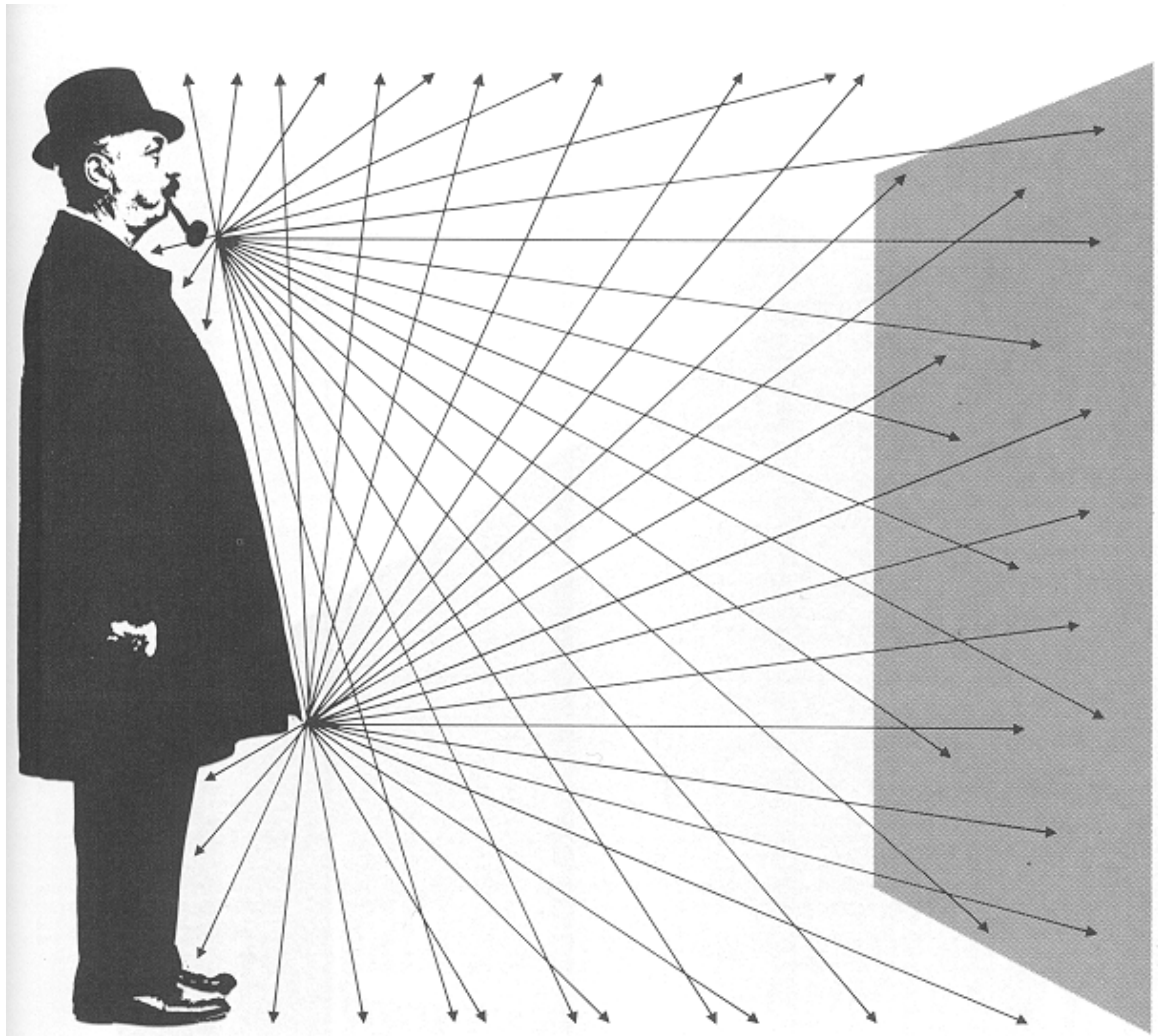
Definition: Radiant energy is the energy of electromagnetic radiation. It is measured in units of joules, and denoted by the symbol:

$$Q \ [\mathrm{J} = \mathrm{Joule}]$$

Definition: Radiant flux (power) is the energy emitted, reflected, transmitted or received, per unit time.

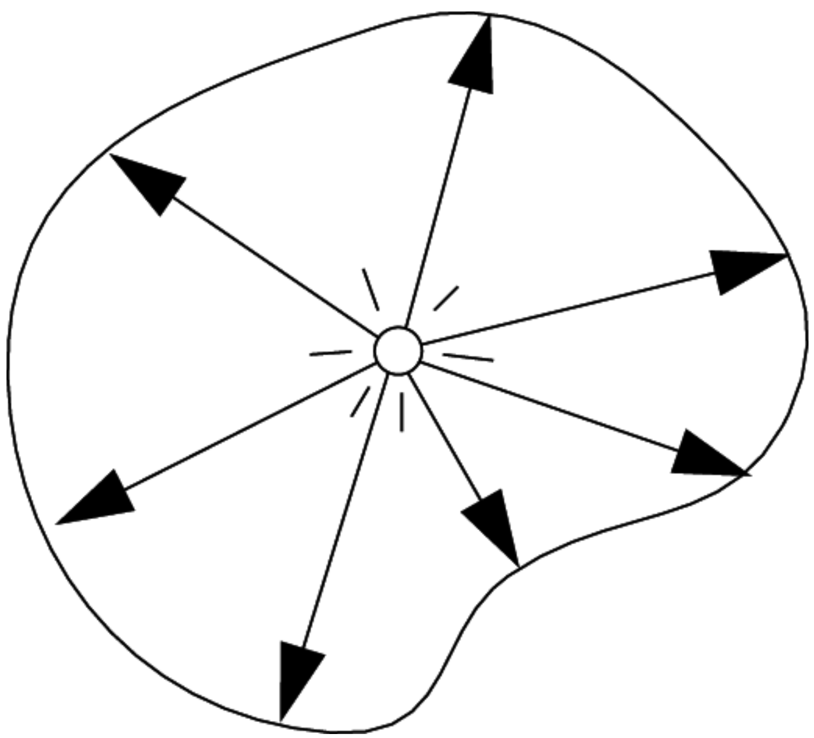
$$\Phi \equiv \frac{\mathrm{d}Q}{\mathrm{d}t} \ [\mathrm{W} = \mathrm{Watt}] \ [\mathrm{lm} = \mathrm{lumen}]^{*}$$

Flux – #photons flowing through a sensor in unit time



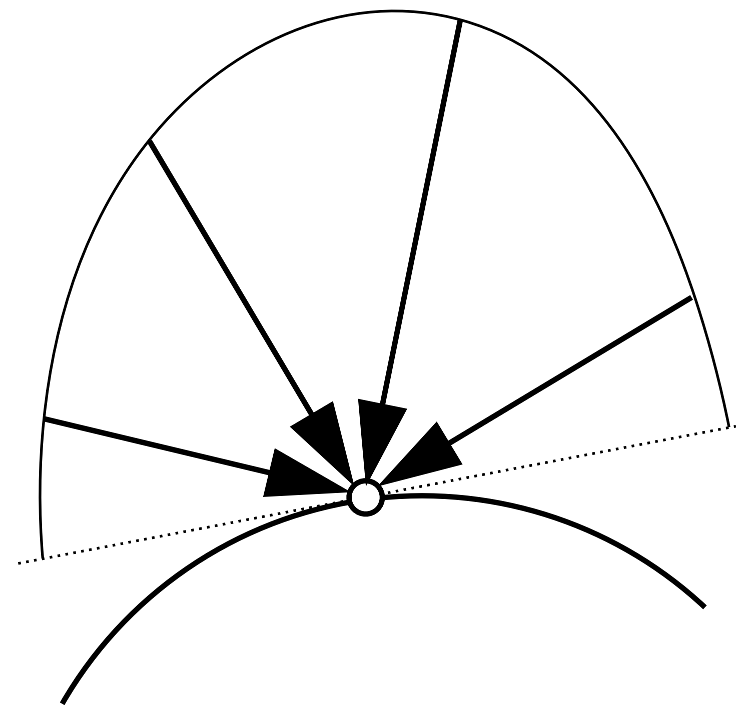
From London and Upton

Important Light Measurements of Interest



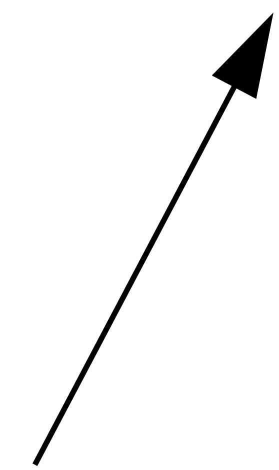
Light Emitted
From A Source

"Radiant Intensity"



Light Falling
On A Surface

"Irradiance"



Light Traveling
Along A Ray

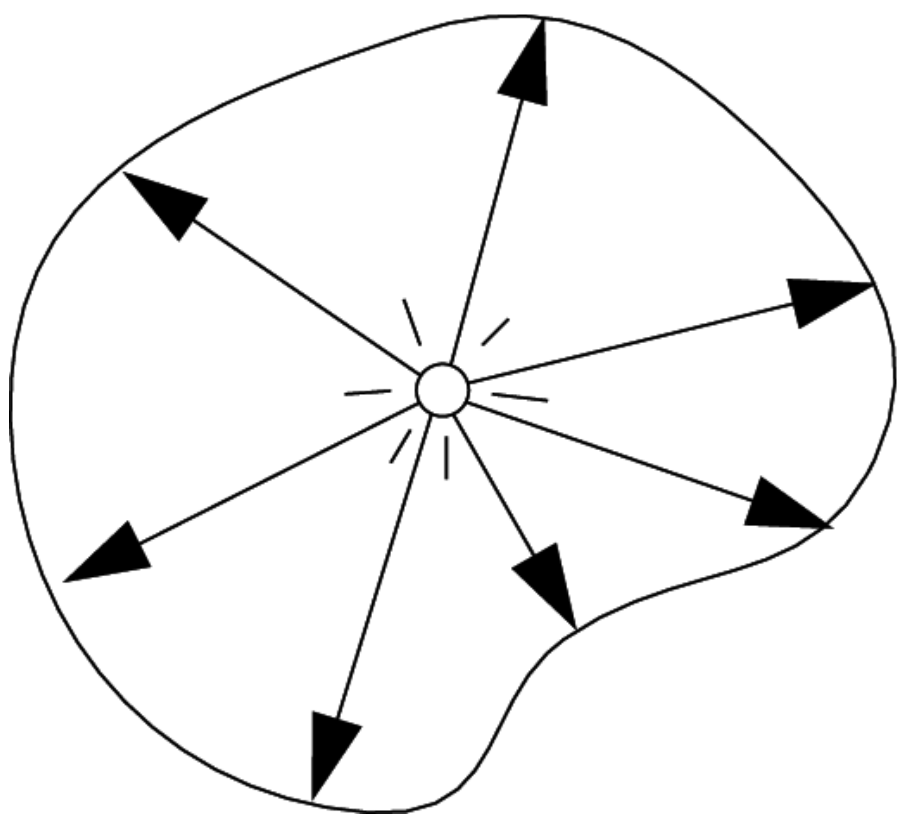
"Radiance"

Radiant Intensity

Radiant Intensity

Definition: The radiant (luminous) intensity is the power per unit **solid angle (?)** emitted by a point light source.

(立体角)



$$I(\omega) \equiv \frac{\mathrm{d}\Phi}{\mathrm{d}\omega}$$

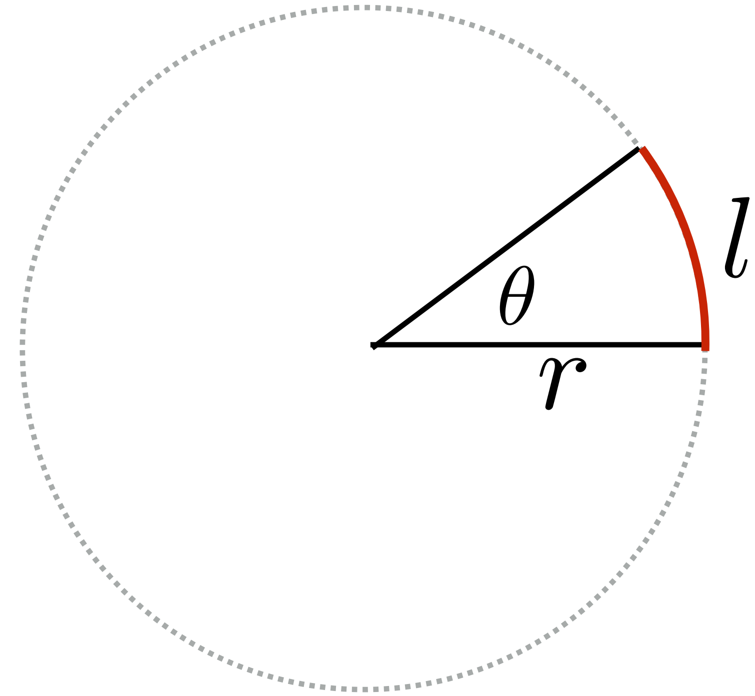
$$\left[\frac{\mathrm{W}}{\mathrm{sr}}\right] \left[\frac{\mathrm{lm}}{\mathrm{sr}} = \mathrm{cd} = \mathrm{candela}\right]$$

The candela is one of the seven SI base units.

Angles and Solid Angles

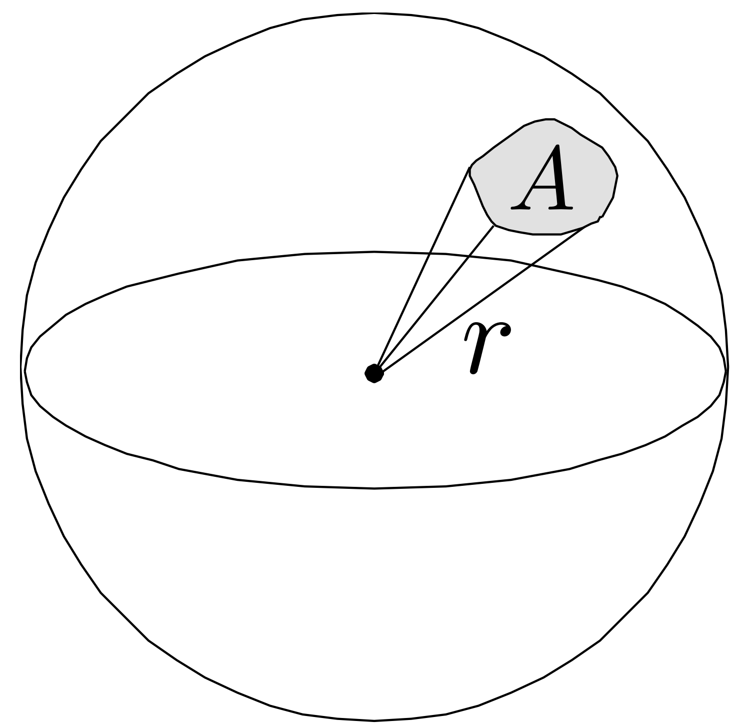
Angle: ratio of subtended arc length on circle to radius

- $\theta = \frac{l}{r}$
- **Circle has** 2π **radians**

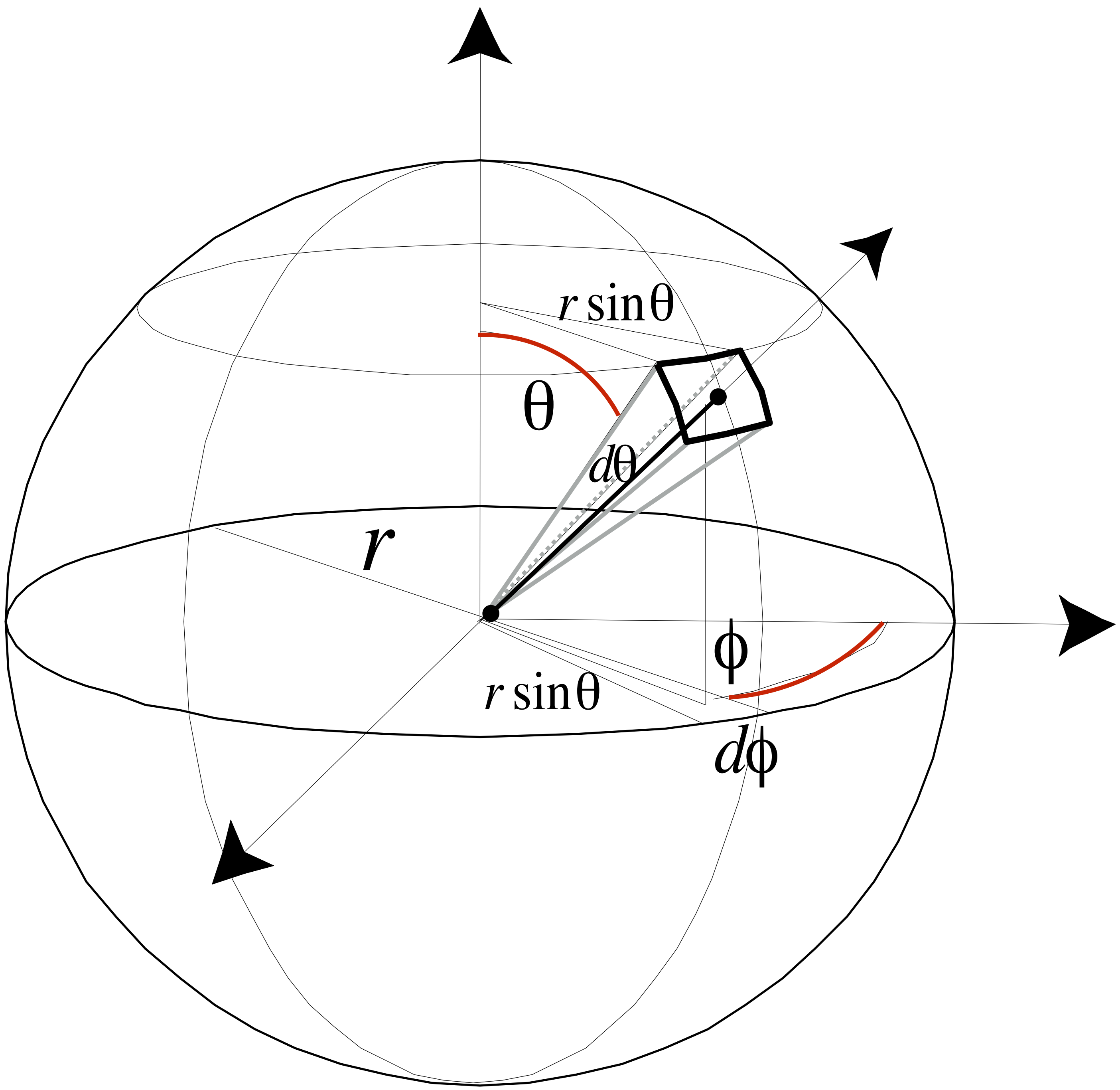


Solid angle: ratio of subtended area on sphere to radius squared

- $\Omega = \frac{A}{r^2}$
- **Sphere has** 4π **steradians**



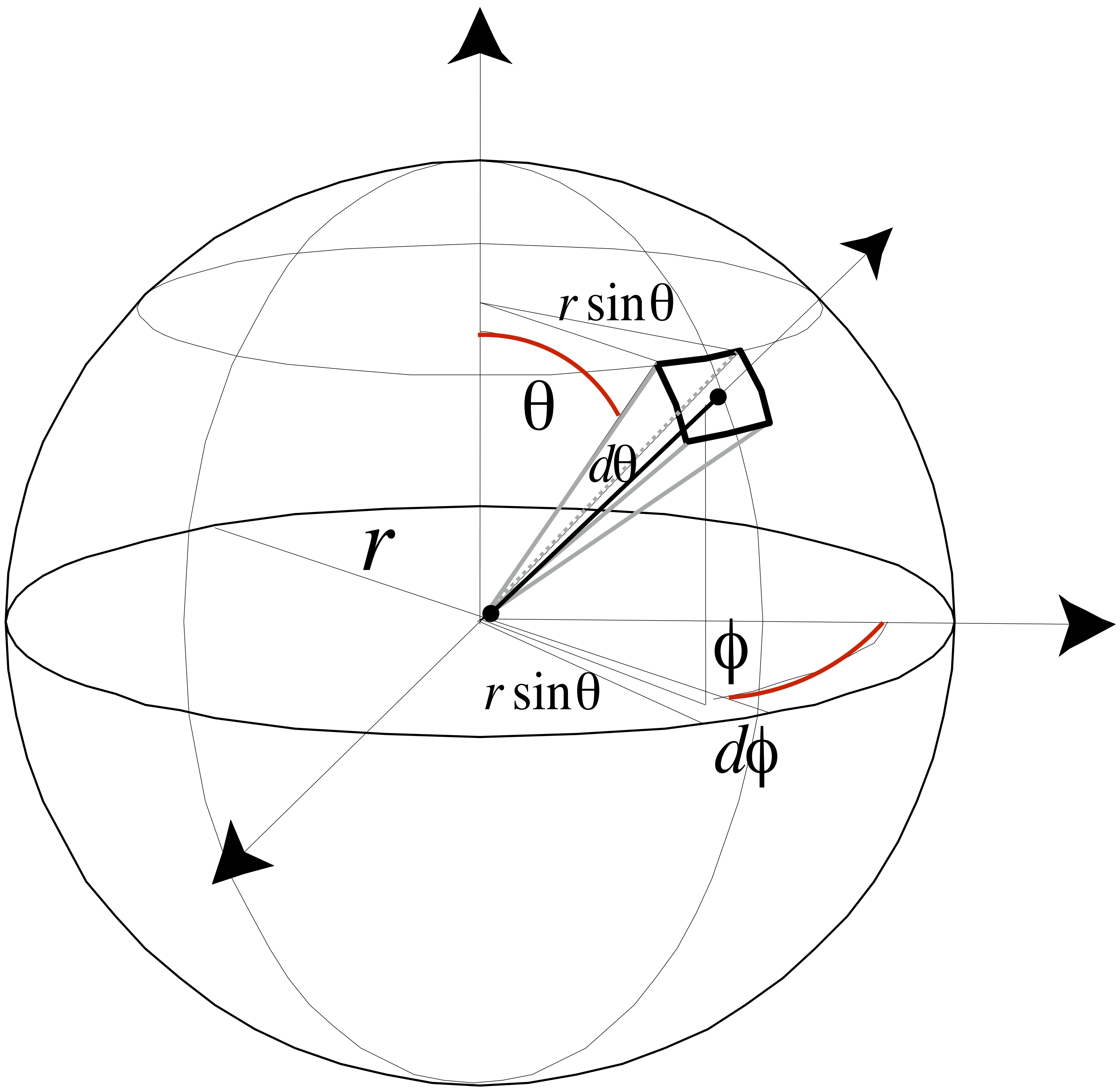
Differential Solid Angles



$$\begin{aligned} \mathrm{d}A &= (r\,\mathrm{d}\theta)(r\sin\theta\,\mathrm{d}\phi) \\ &= r^2\sin\theta\,\mathrm{d}\theta\,\mathrm{d}\phi \end{aligned}$$

$$\mathrm{d}\omega = \frac{\mathrm{d}A}{r^2} = \sin\theta\,\mathrm{d}\theta\,\mathrm{d}\phi$$

Differential Solid Angles



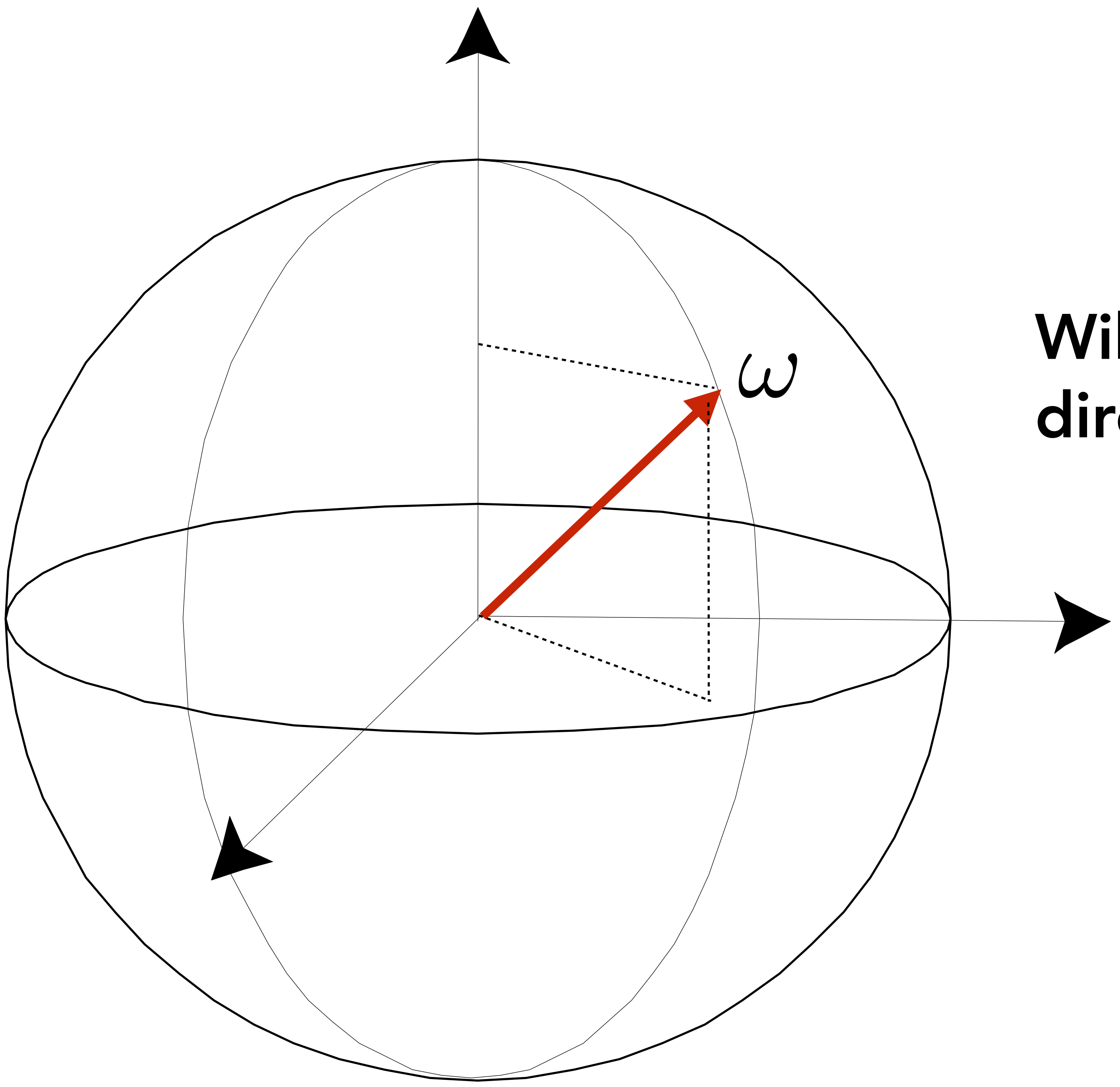
Sphere: S^2

$$\Omega = \int_{S^2} \mathrm{d}\omega$$

$$= \int_0^{2\pi} \int_0^{\pi} \sin\theta \, \mathrm{d}\theta \, \mathrm{d}\phi$$

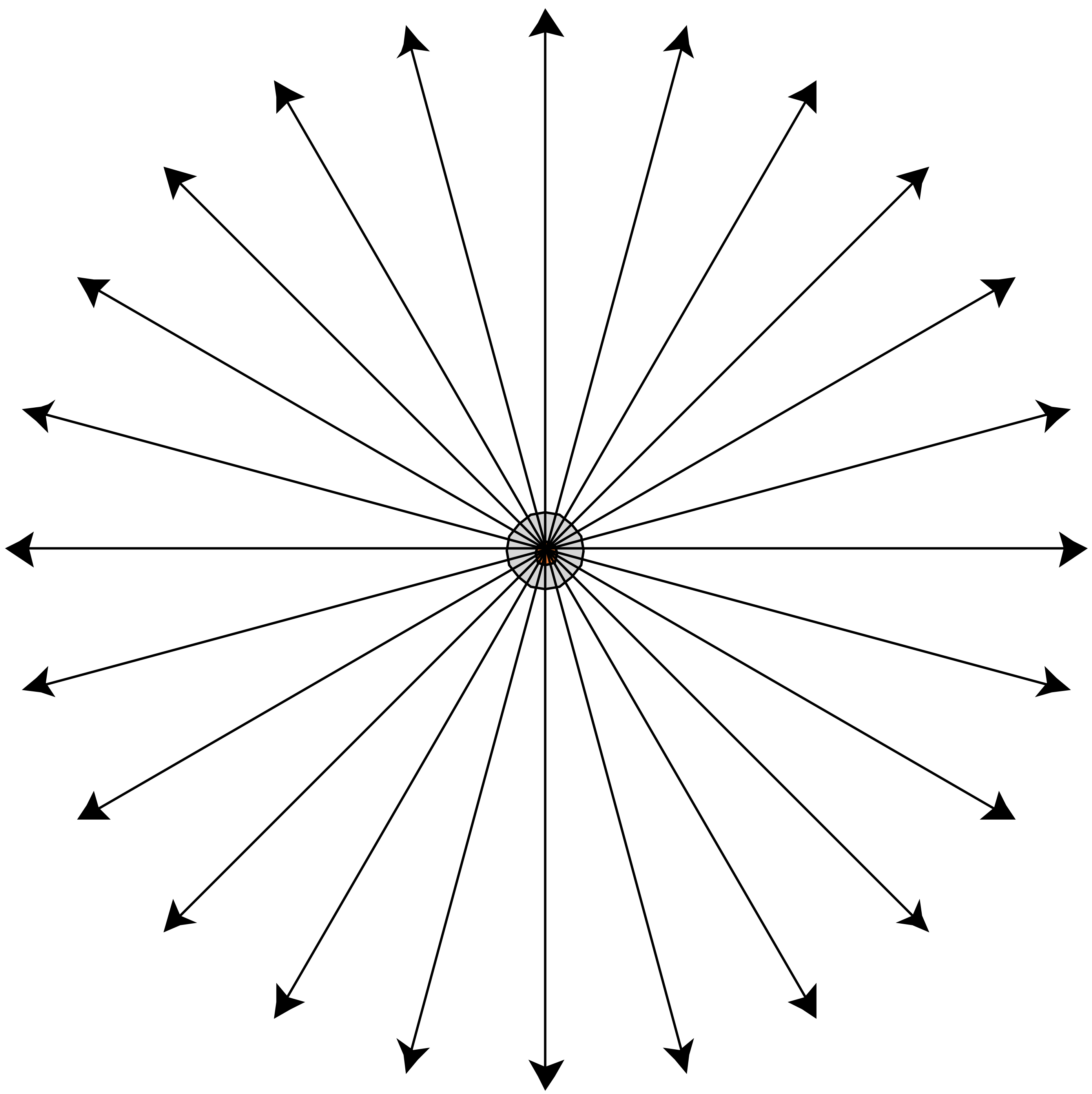
$$= 4\pi$$

ω as a direction vector



Will use ω to denote a direction vector (unit length)

Isotropic Point Source



$$\Phi = \int_{S^2} I \, \mathrm{d}\omega$$
$$= 4\pi I$$

$$I = \frac{\Phi}{4\pi}$$

Modern LED Light

Output: 815 lumens

(11W LED replacement
for 60W incandescent)

Radiant intensity?

Assume isotropic:
Intensity = 815 lumens / 4pi sr
= 65 candelas

Thank you!

(And thank Prof. Ravi Ramamoorthi and Prof. Ren Ng for many of the slides!)