Data Structures for Computer Graphics

INTRODUCTION

Vlastimil Havran

Czech Technical University in Prague

Lectured by Vlastimil Havran

Lectures Overview

1. Rules of the game, lectures overview, review of sorting and searching, intro to computer graphics algorithms, questions to the course.
2. Introduction to hierarchical and regular data structures used in CG
3. Incidence operations used in computer graphics
4. Point based representations and data structures
5. Object based and imaged based representations in 2D and 3D
6. Proximity search and its applications I
7. Proximity search and its applications II
8. Proximity search in high-dimensional spaces, data structures for sampling

Lectures Overview

9. Ray shooting and its applications I
10. Ray shooting and its applications II
11. Visibility culling for large scale scenes
12. Static and advanced collision detection
13 and 14. Reserved + Easter Holidays

Data Structures & Algorithms

- Data structures - inherent part of algorithms
- Large data, many queries

  Efficiency is crucial!
- Example:
  - Ray tracing ~10M rays per frame
  - Real time (50FPS): 2ns per ray
  - Scene with 10M triangles
    - naïve alg. 2.5×10^{15} intersections/second

Organizing Data Gives Advantage

- Improve searching performance
  - Naïve search: O(n) time
  - With sorting: O(log n)!
  - In special cases even O(1)

Basic Data Structures

- 1D arrays
- Multidimensional arrays

  (1D array X 1D array X ….)
- Linked lists (unidirectional, bidirectional, circular)
- Stacks and Queues
- Hash tables

Data Structures Properties (Complexity)

- Preprocessing (construction) time T_C
- Space (storage) S
- Running time T_R

Properties

- Asymptotic bounds (O, Ω, Θ)

  (worst case complexity and average case complexity)
- Instance based measurements (running time in seconds, used memory in Bytes, …)

At the Core of Algorithms we have:

Sorting and Searching

Recall Sorting and Searching Methods

- Sequential / Binary search
- Quicksort
- Merge sort
- Insertion sort
- Radix sort
- Asymptotic bounds

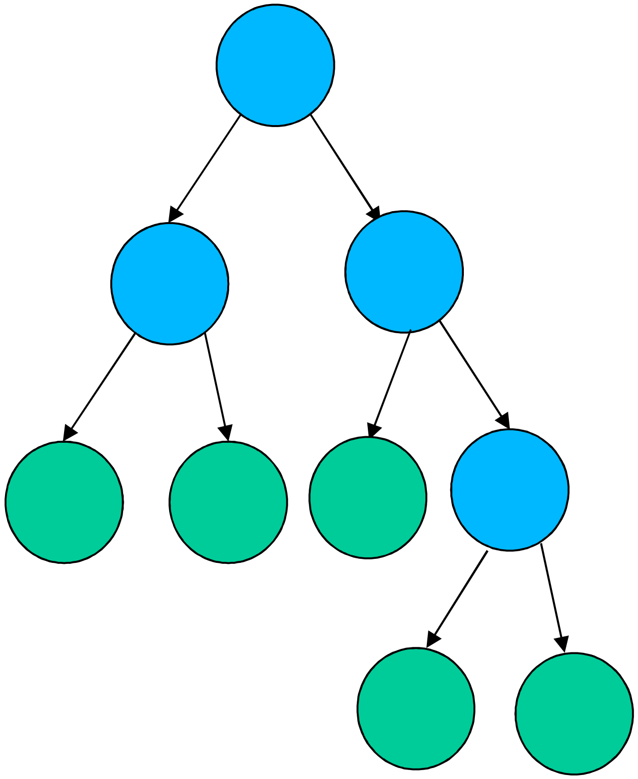
Searching in 1D Arrays

1D Sorted array: O(N) space, O(N log N) time

- Sequential search … in 1D
- Binary search ... in 1D sorted arrays O(log N)
- Interpolation search (using tangent) O(log log N) for some instances
- Binary-Interpolation search – one step binary search one step interpolation search
- Binary-Sequential search – at first binary search, at the end sequential search

Searching in (balanced) binary trees

- Balanced/unbalanced trees
- Huffman encoding (digital trees)
- Construction O(N log N)
- Space O(N)
- Search O(log N)
- Inorder, preorder,
  postorder traversal



Basic Sorting Algorithms

Algorithm	Method	Best	Average	Worst
Heapsort	Selection	O(n log n)	O(n log n)	O(n log n)
Selection sort	Selection	$O(n^2)$	$O(n^2)$	$O(n^2)$
Quicksort	Partitioning	O(n log n)	O(n log n)	$O(n^2)$
Bucket sort	Distribution	O(n)	O(n)	$O(n^2)$
Merge sort	Merging	O(n log n)	O(n log n)	O(n log n)
Bubble sort	Exchanging	O(n)	$O(n^2)$	$O(n^2)$
Insertion sort	Insertion	O(n)	$O(n^2)$	$O(n^2)$

Space complexity: O(n)

Quicksort

- Pick up a pivot Q
- Reorder the data into two subarrays
  - left part ≤ Q
  - right part > Q
- Recurse in both subarrays
- $O(N^2)$, but with high probability O(N log N)
- Simple & usually fast – commonly

  Relevant to partitioning techniques

Merge Sort

- From single numbers to tuples, from tuples to 4-tuples (1->2->4->8->16->32->64 ...)
- Auxiliary array needed
- O(N) space
- O(N log N) running time

Relevant to clustering techniques

Insertion Sort

- Keep partially sorted array
- Sorting by insertion
- O(N) space
- $O(N^2)$ complexity

Relevant to incremental DS construction

Other Sorting Techniques for Limited Precision and/or Data Distribution

- Bucket sort – only for unique keys with limited distribution
- Radix sort
  - bottom up (from LSB to MSB)
  - top-down (distribution sort to many equivalence classes and recurse)

Relevant to rasterization

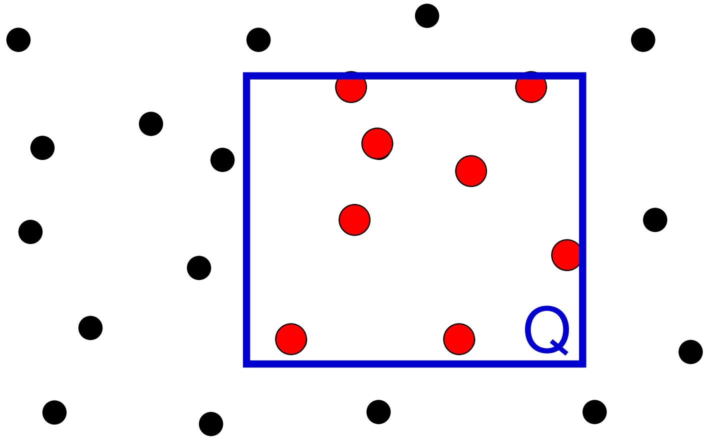
Sorting & Searching in Computer Graphics Algorithms

Data Entities in Computer Graphics

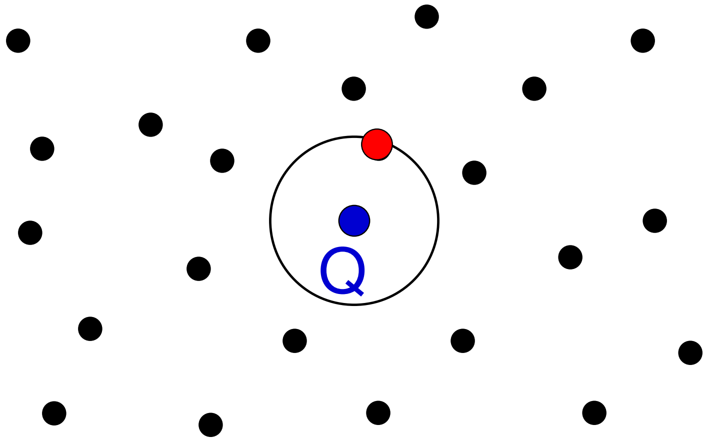
- Not just numbers !
- Multidimensional data or objects (at least 2D)
- Examples: points, lines, oriented half-lines, disks, oriented hemispheres, triangles, polygons, spheres, general objects in 2D, 3D etc., images, contours, multi-layer data structures, extension to temporal domain
- **Consequence:** Instead of 1D problem we solve problems in multidimensional spaces!

Geometric Search Problems

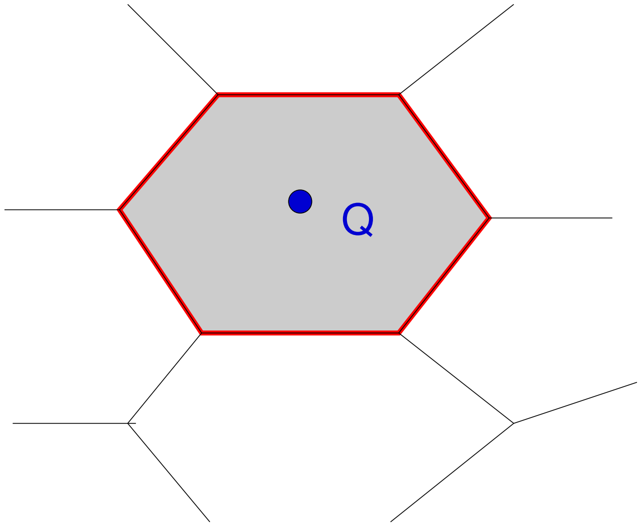
Range search



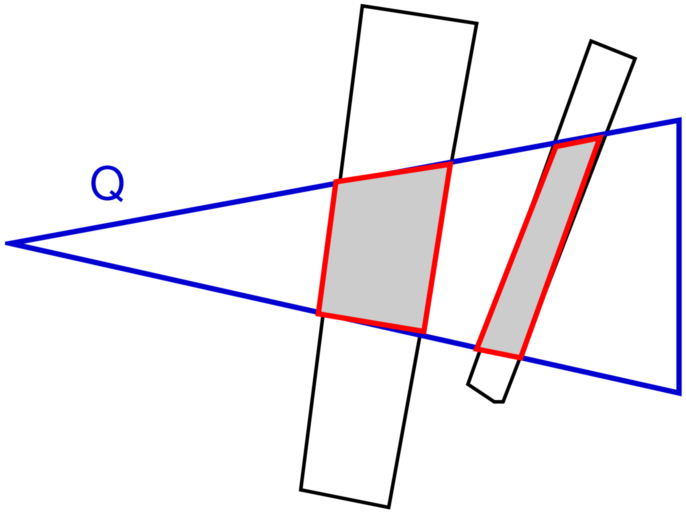
Nearest Neighbors



Point location



Intersection detection



Search Problems in Rendering

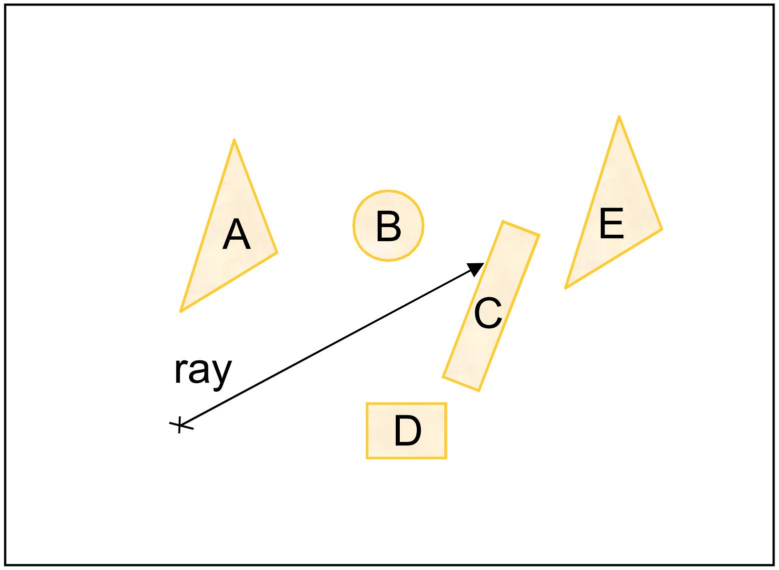
Q x S → A, Q query domain, S search space, A domain of answers

Problem	Q	S	A
Ray shooting	ray	{objects}	point
Hidden Surface Removal	{rays}	{objects}	{points}
Visibility culling	{rays}	{objects}	{objects}
Photon maps	point	{points}	{points}
Ray maps	point	{rays}	{rays}
Irradiance caching	point	{spheres}	{spheres}

90% of Computer Graphics is about Sorting and Searching…

Example 1: Ray Shooting

Task: Given a ray, find out the first object intersected.

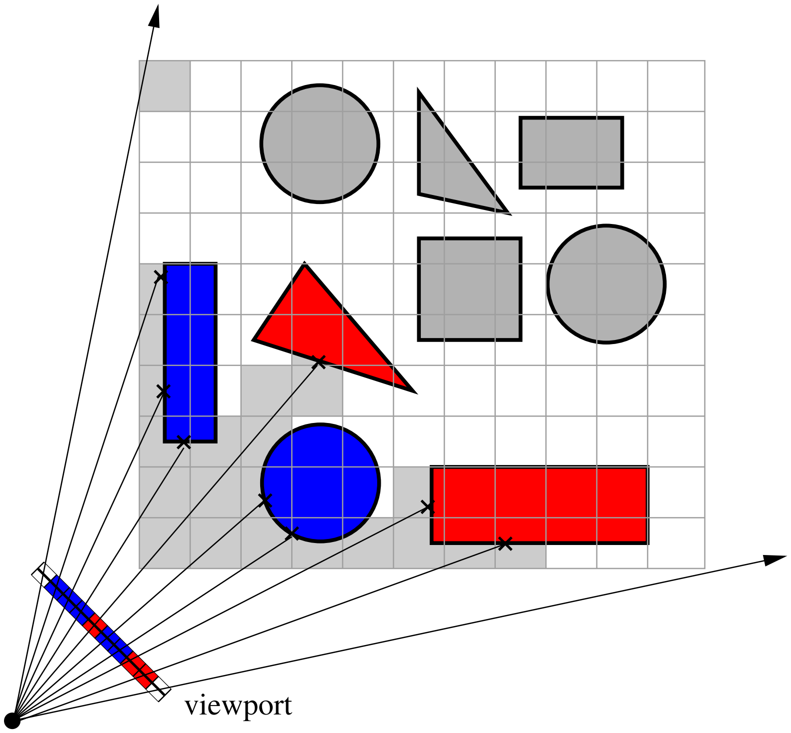


Input: a *scene* and a *ray*

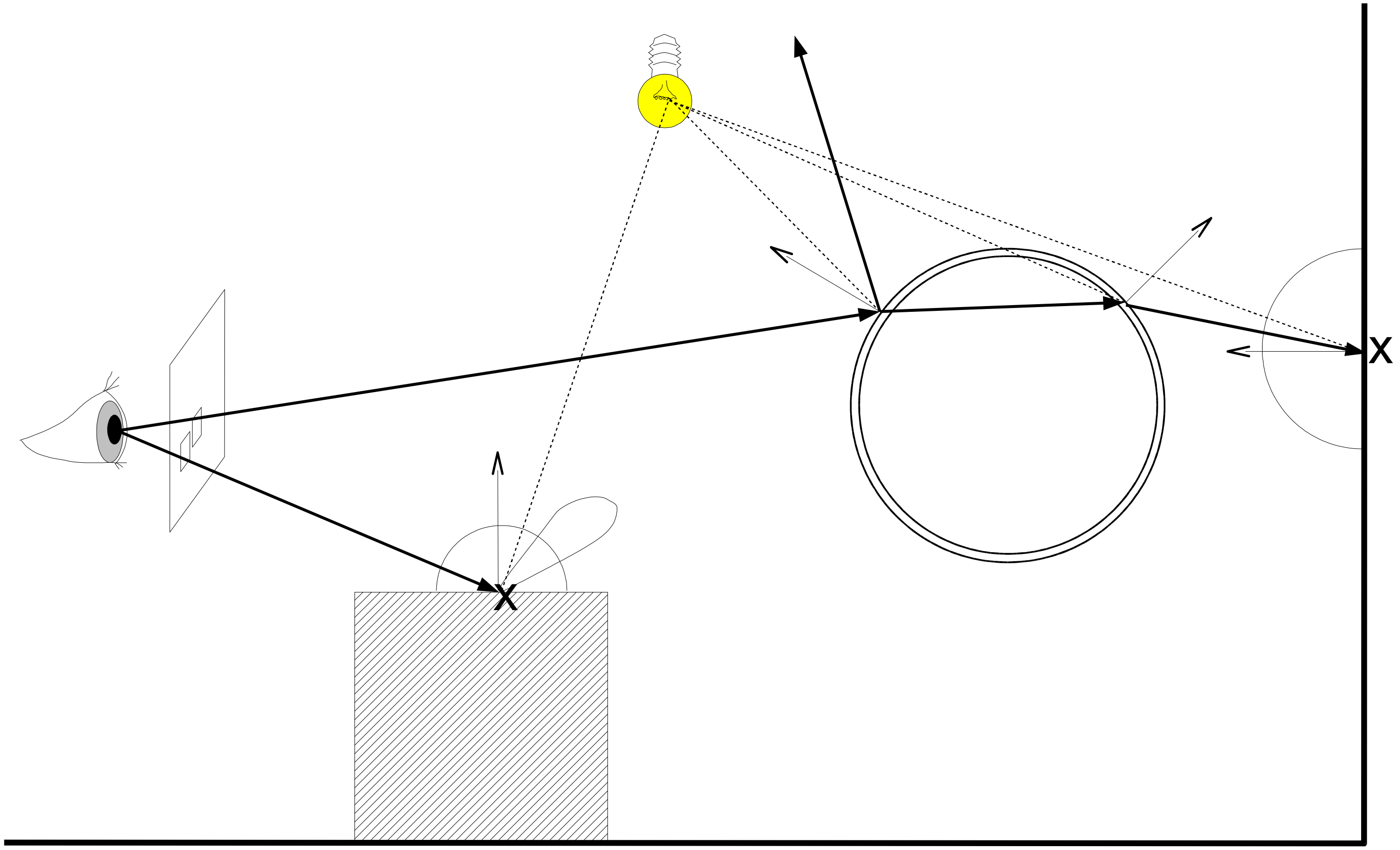
Output: the *object C*

Example 2: Ray Casting for Image

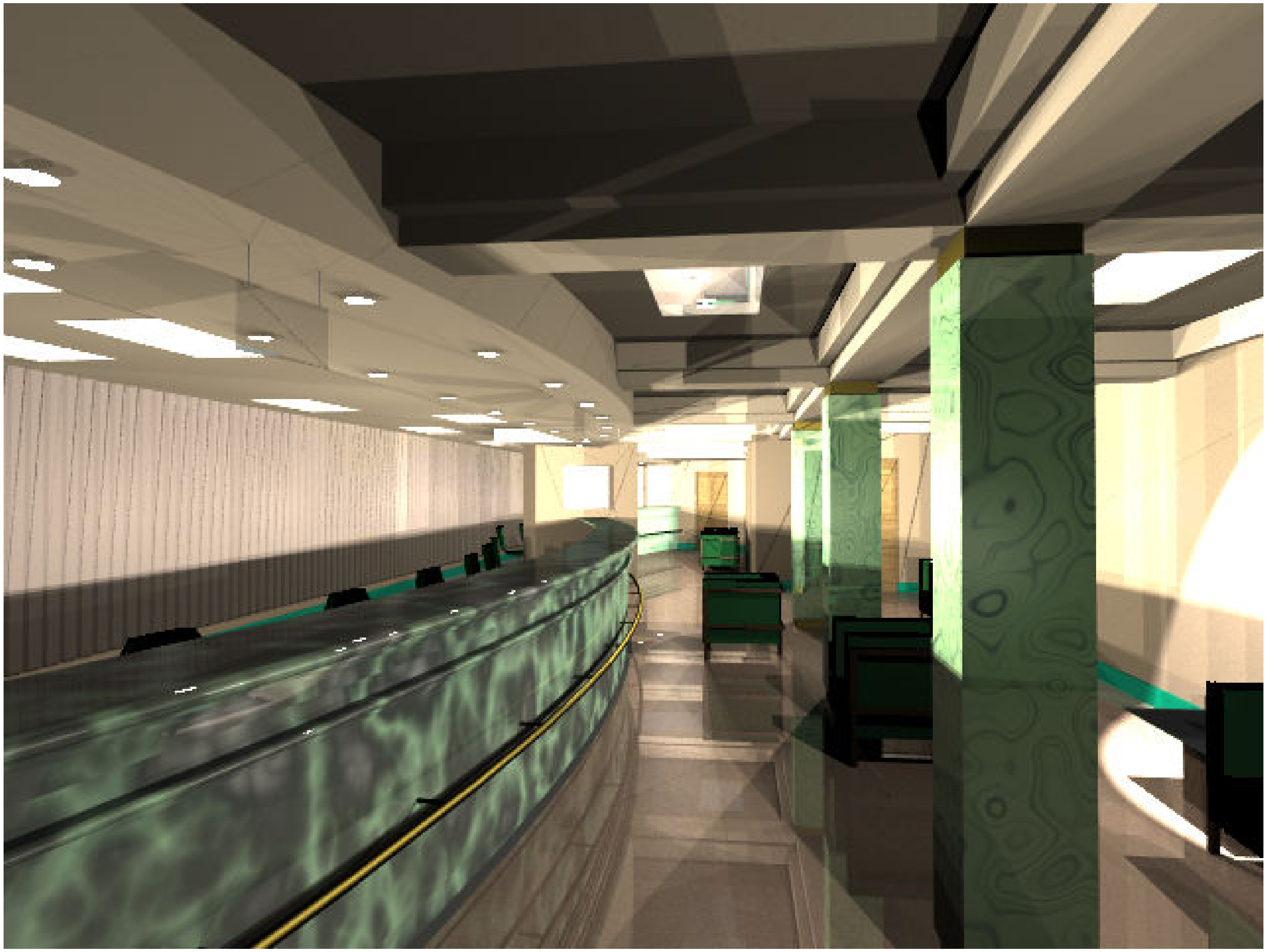
- Cast ray for each pixel (primary ray)
- Step 1: spatial data structure (XYZ)
  - Preprocess
  - Trees ~ quick sort
  - Grid ~ distribution sort
- Step 2: search for nearest intersection
  - Min selection with early termination



Example 3: Recursive Ray Tracing



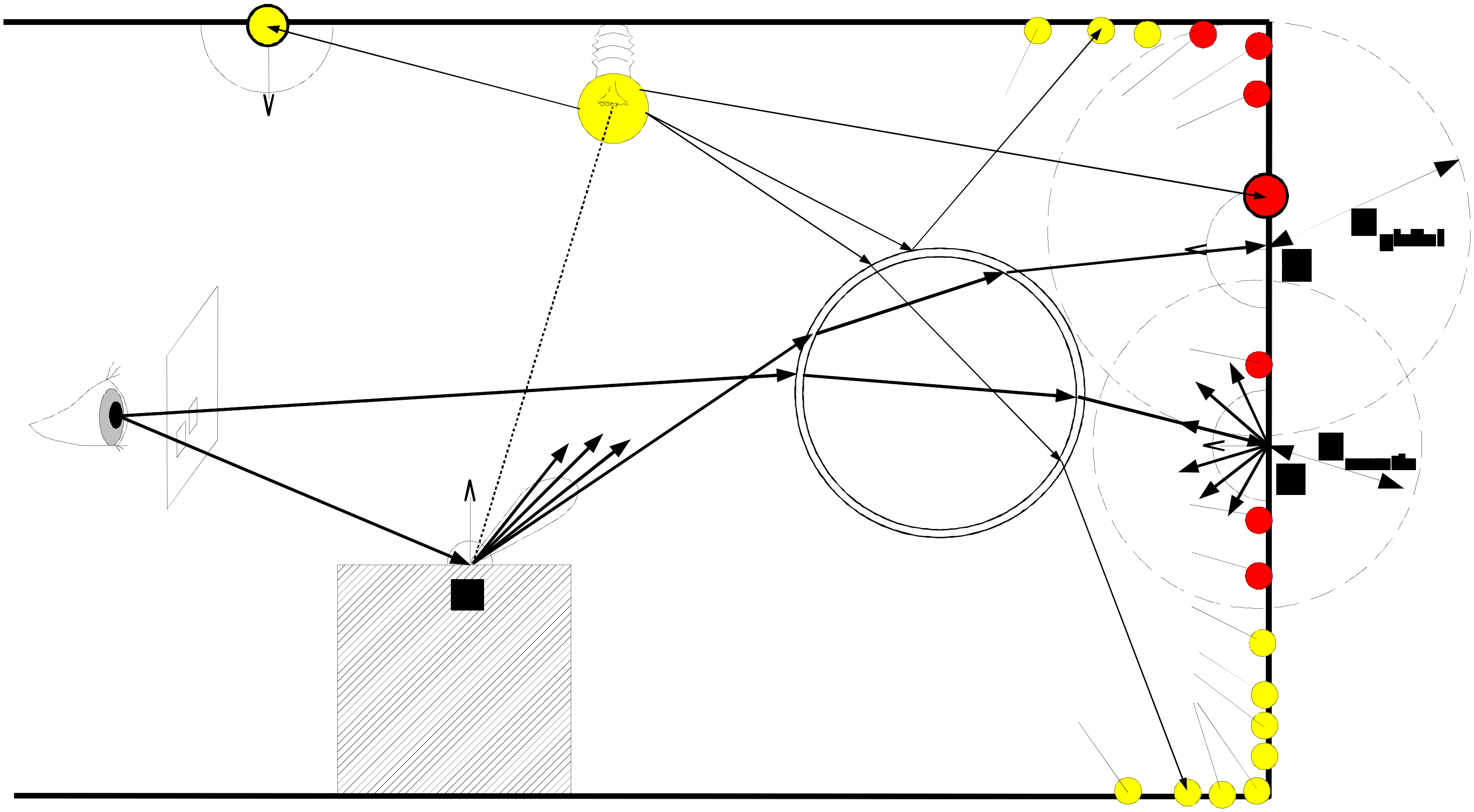
Sample Ray Traced Image



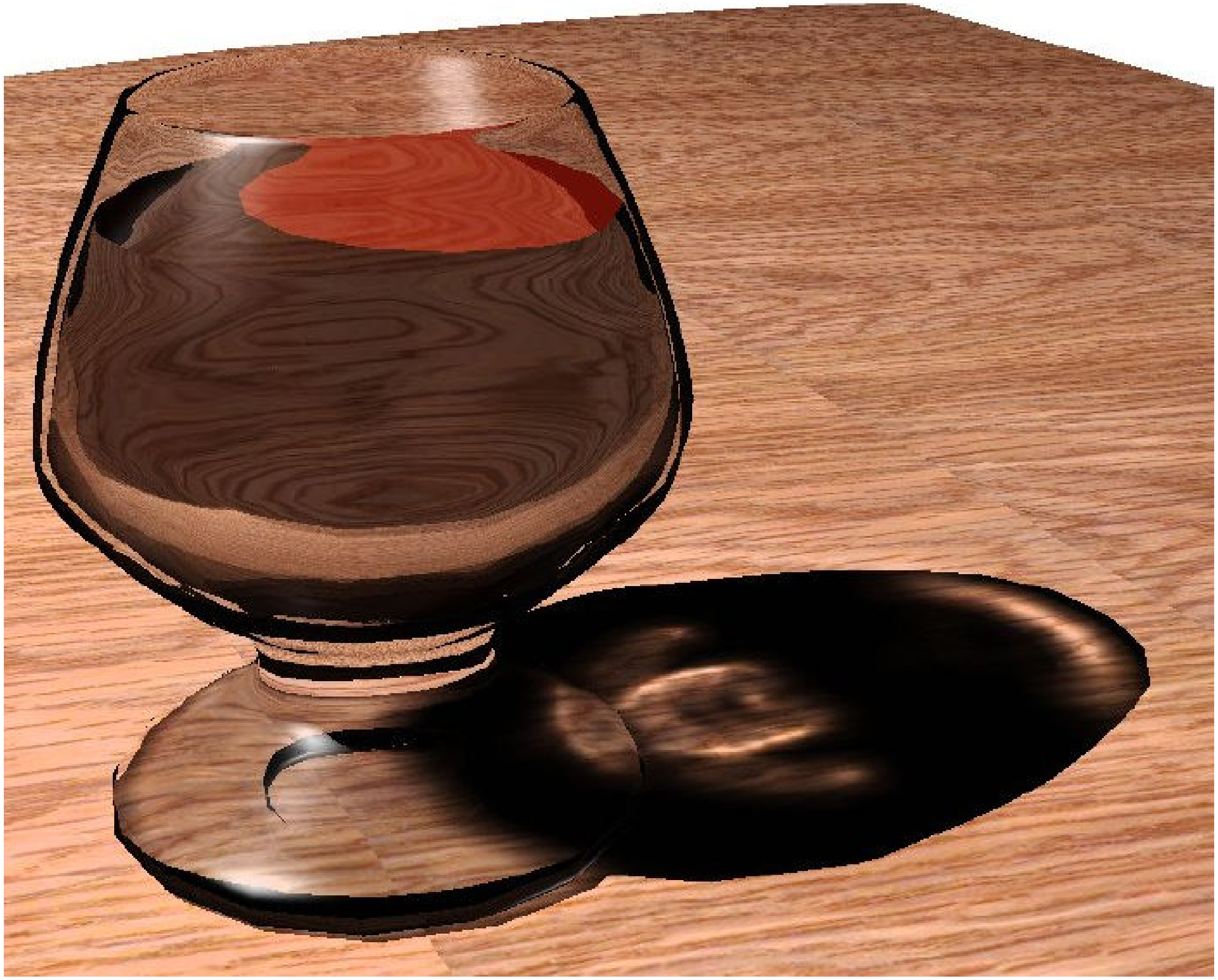
Example 4: Photon Mapping

Phase I: photon shooting

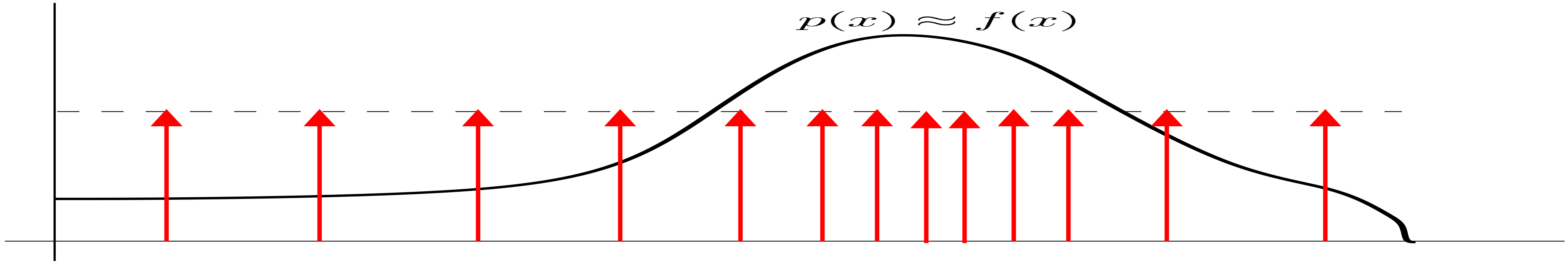
Phase II: density estimation



Sample Image by Photon Mapping



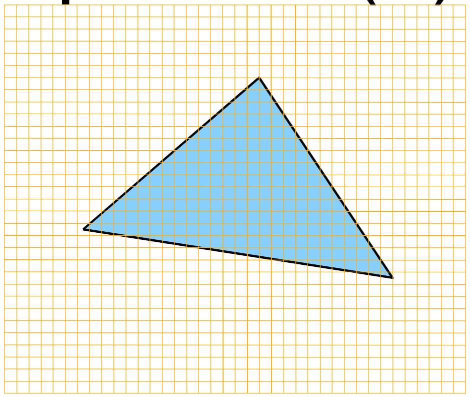
Example 5: Density Estimation in 1D



Density Estimation: given the samples, estimate probability density function *p(x)*

Example 6: Z-buffer based Rendering

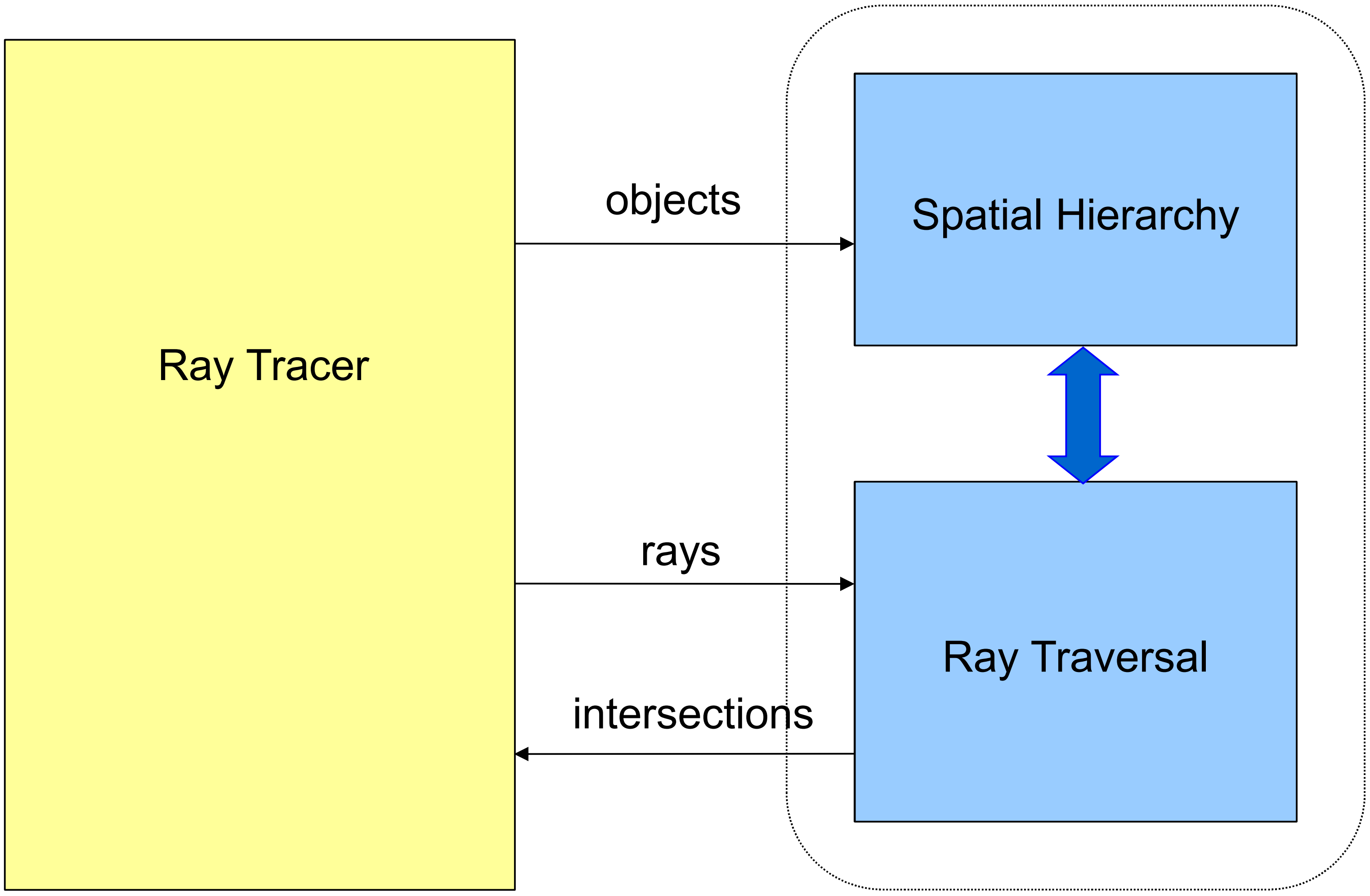
- Rasterize polygons in arbitrary order
- Maintain per pixel depths
- Step 1: rasterization (YX)
  - Bucket sort like
- Step 2: per pixel depth comparison (Z)
  - Min selection



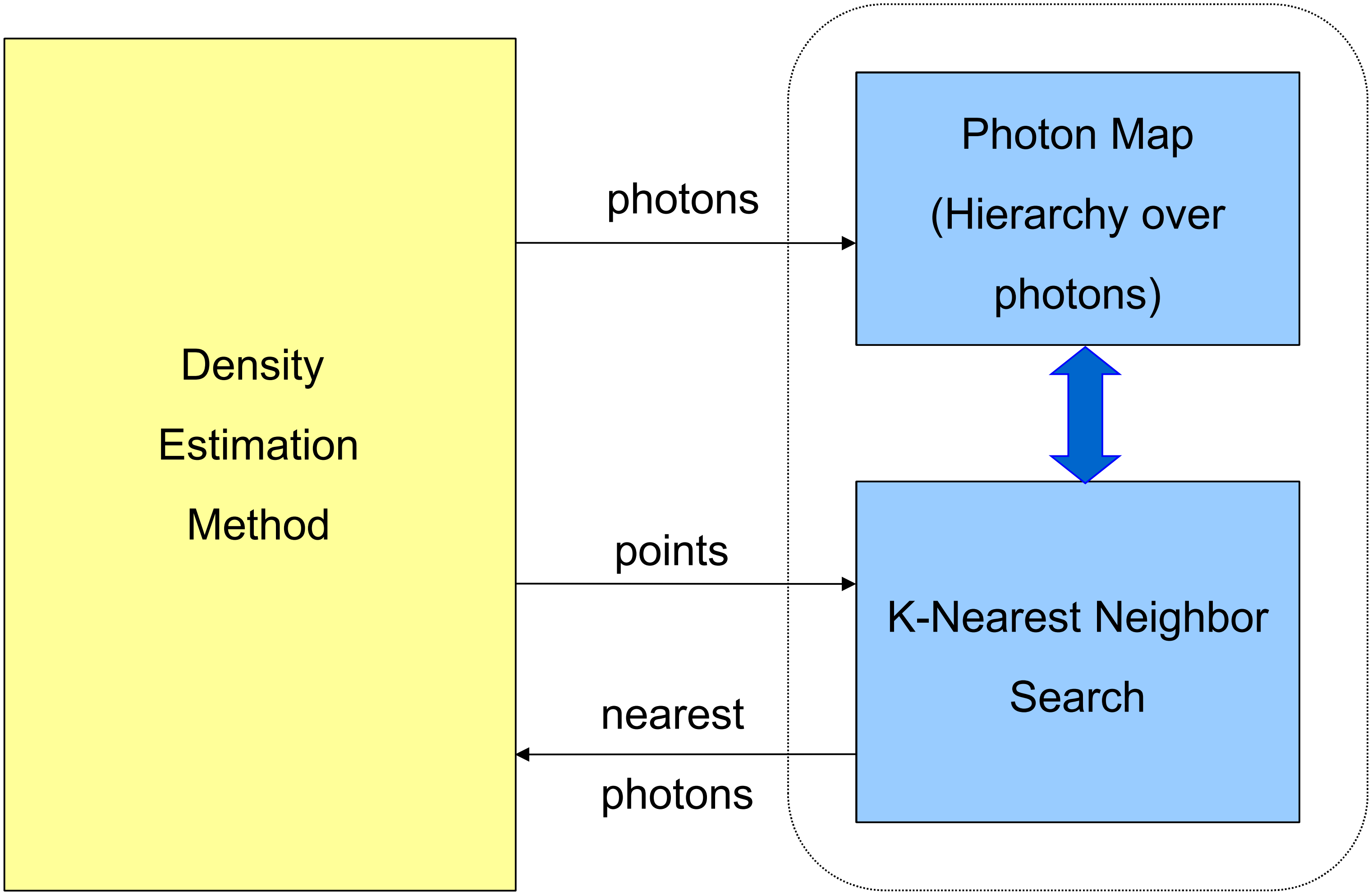
Example 7: Collision Detection

- Incidence among simple/complex objects
- Real-time demands
- Application: games, motion planning for industry (car assembly), etc.

Example I: Ray Tracing



Example II: Photon Density Estimation



Density Estimation Method

photons

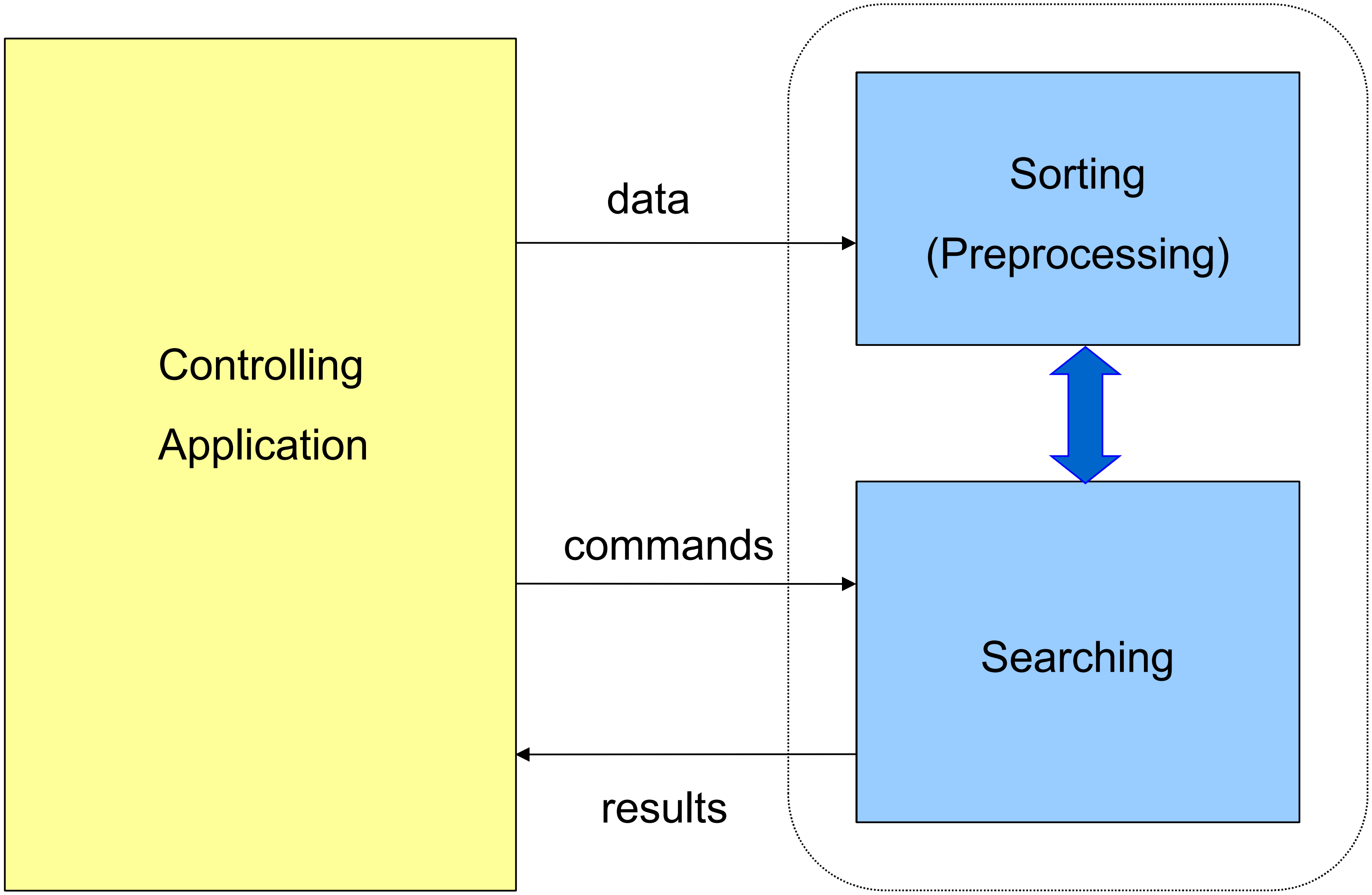
Photon Map (Hierarchy over photons)

points

K-Nearest Neighbor Search

nearest photons

Sorting and Searching – Typical usage in CG



Controlling Application

data

commands

results

Sorting (Preprocessing)

Searching

Thank you for your attention!