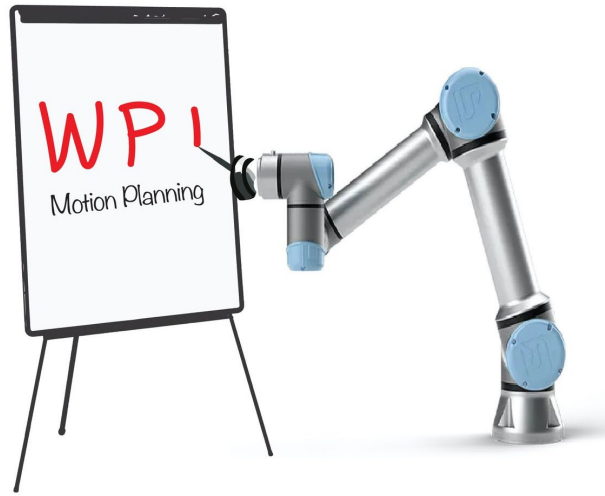
RBE550
Motion Planning
Collision Checking



Constantinos Chamzas
www.cchamzas.com
www.elpislab.org

Disclaimer and Acknowledgments

The slides are a compilation of work based on notes and slides from Constantinos Chamzas, Lydia Kavraki, Zak Kingston, Howie Choset, David Hsu, Greg Hager, Mark Moll, G. Ayorkor Mills-Tetty, Hyungpil Moon, Zack Dodds, Nancy Amato, Steven Lavalle, Seth Hutchinson, George Kantor, Dieter Fox, Vincent Lee-Shue Jr., Prasad Narendra Atkar, Kevin Tantiseviand, Bernice Ma, David Conner, Morteza Lahijanian, Erion Plaku, and students taking comp450/comp550 at Rice University.

Last Time: Recap

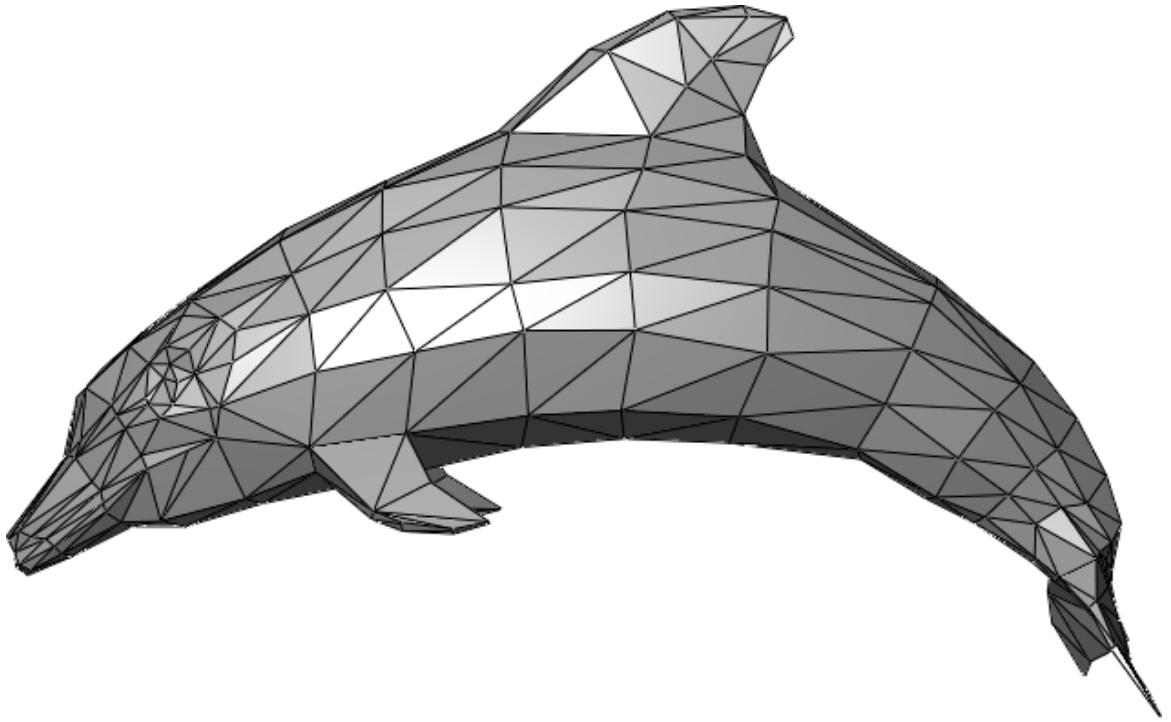
- Asymptotic Optimality
- RRT*
- RRT#
- BIT*, FMT*, IRRT*

Overview

- Motivation
- Bounding Volumes
- Hierarchies of Bounding Volumes
- Spatial Partitioning

Motivation

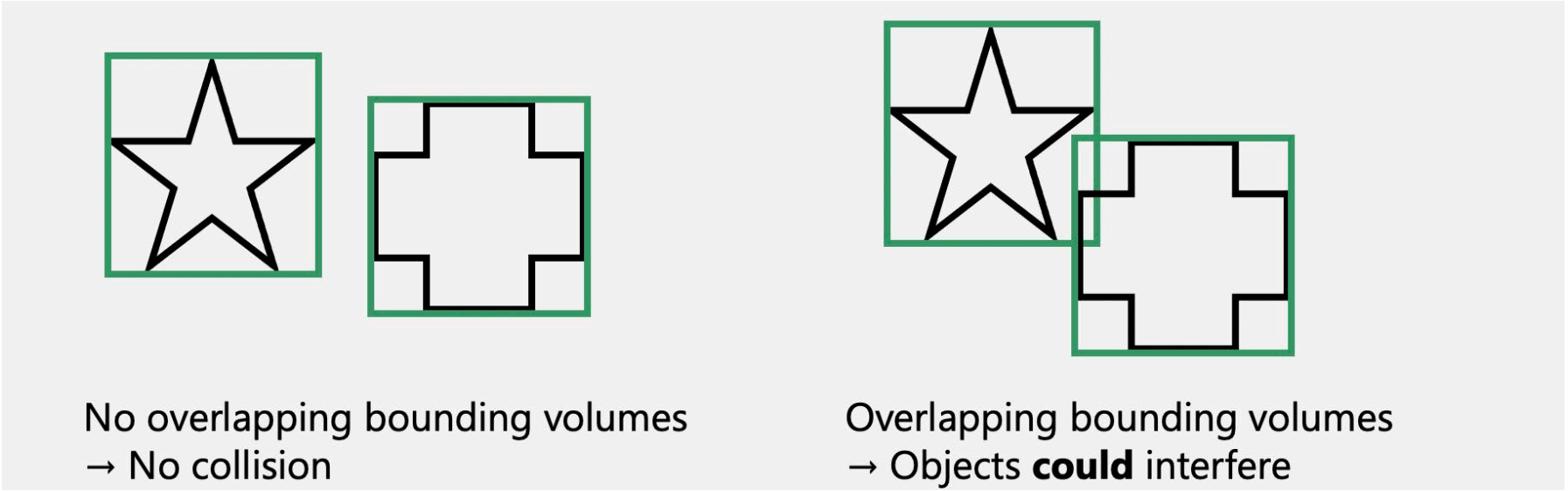
- Naïve collision checking for general polygons (polytopes) is $O(n^2)$
- State-of-the-art motion planners spend most of their time performing collision checks up to 90%



How can we quickly rule out collisions?

Motivation

- Collision detection for polygonal models is in $O(n^2)$
- Simple bounding volumes – encapsulating geometrically complex objects – can accelerate the detection of collisions

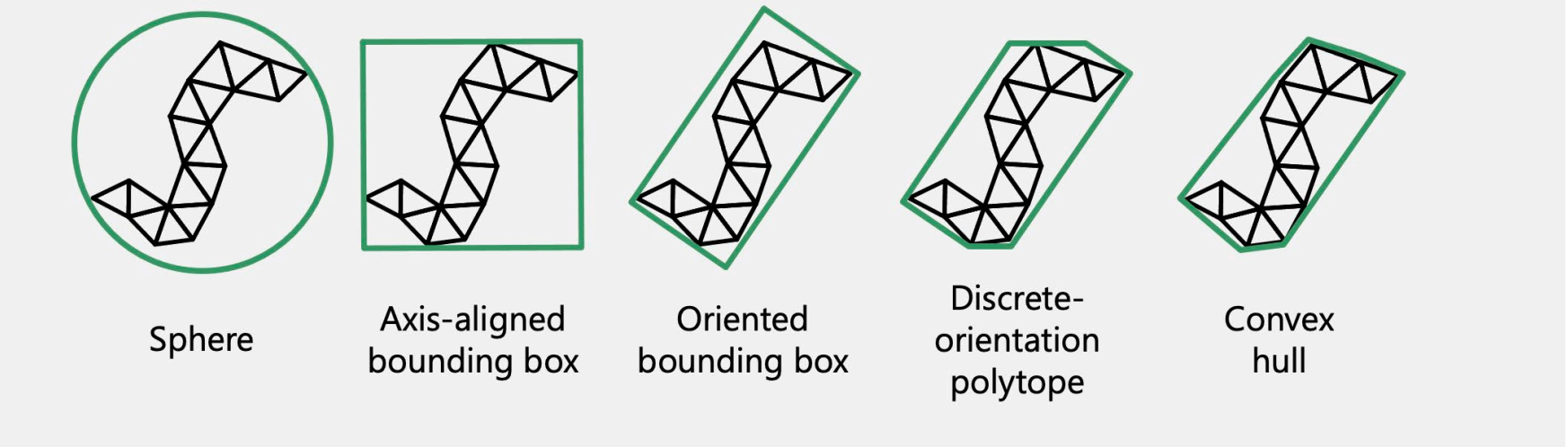


Use bounding volumes !

Motivation

- Efficient detection of collision-free state without checking all object primitives
- Overhead for overlapping bounding volumes (potential-collision)
  - However, often only few bounding volumes will overlap due to spatial coherence.
- For some applications, collision information on the bounding volumes might be sufficient
  - Approximate collision detection
  - Accuracy depends on the tight fitting of the bounding volume

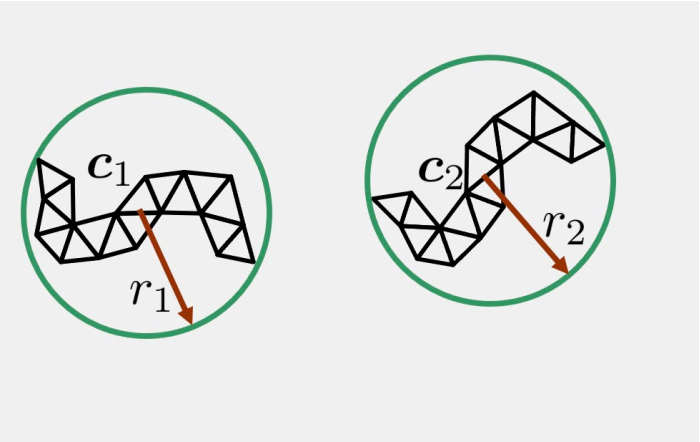
Bounding Volumes



- Desired characteristics:
  - Efficient intersection test, memory efficient
  - Efficient generation and update in case of transformations
  - Tight fitting

Spheres

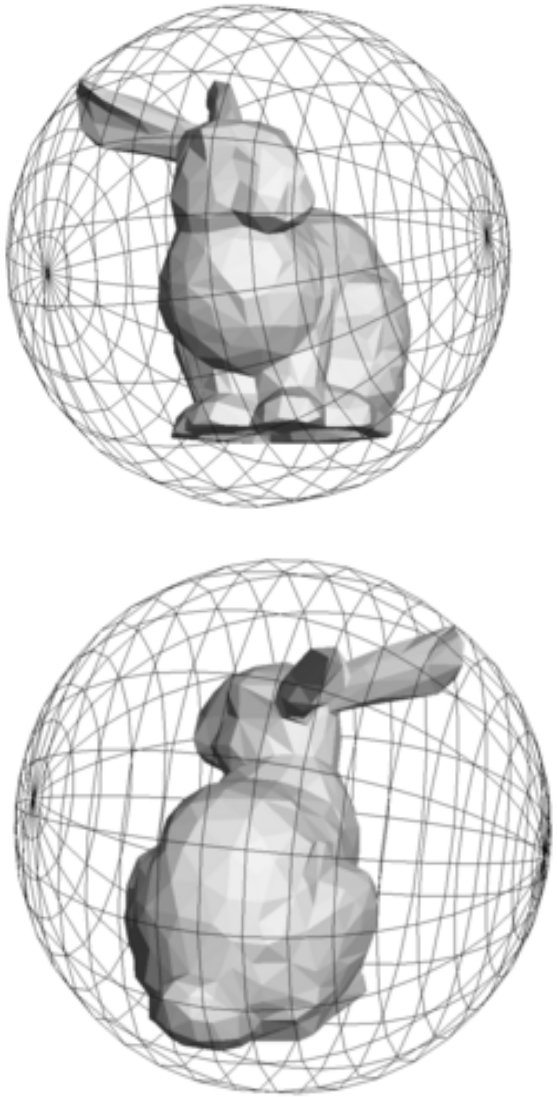
- Spheres are represented by
  - The center position
  - The radius



- Two spheres do not overlap if $(\boldsymbol{c}_1 - \boldsymbol{c}_2)(\boldsymbol{c}_1 - \boldsymbol{c}_2) > (r_1 + r_2)^2$

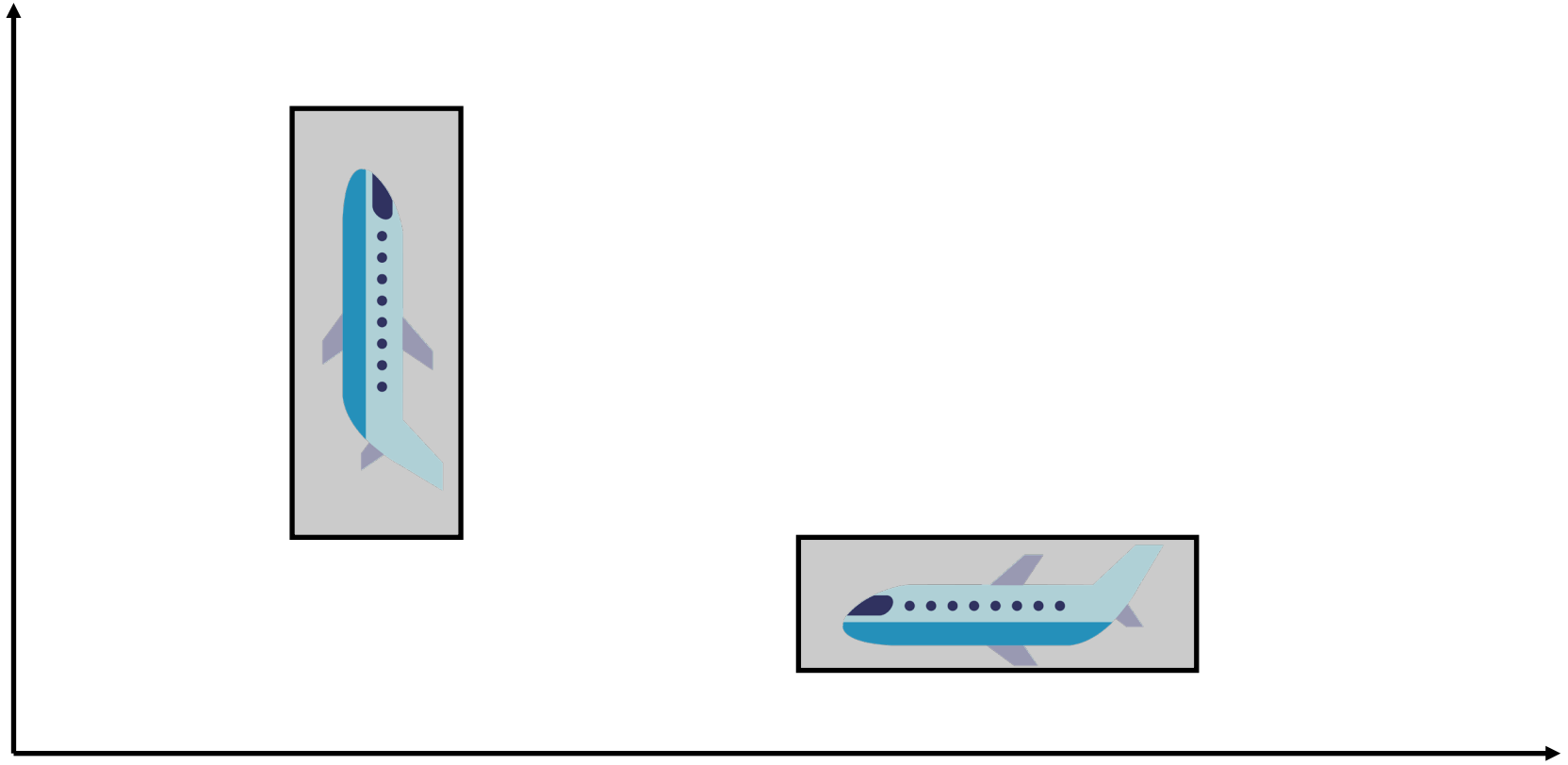
Bounding Spheres

- Sphere intersection is easy
- Algorithm?
  - Find distance between centers
  - If distance < sum of radii: collision
- *O(1)* time!
- Lots of false positives…
- Many other simple shapes are easy to intersect



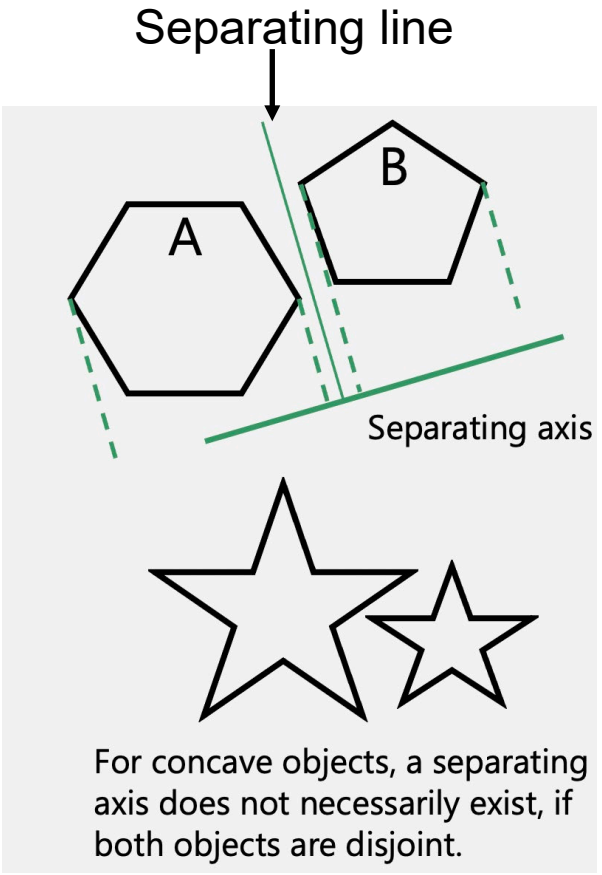
Axis-Aligned Bounding Boxes

- Commonly referred to as AABB (AKA: ABB)
- Approximate everything with a box aligned with the coordinate axes

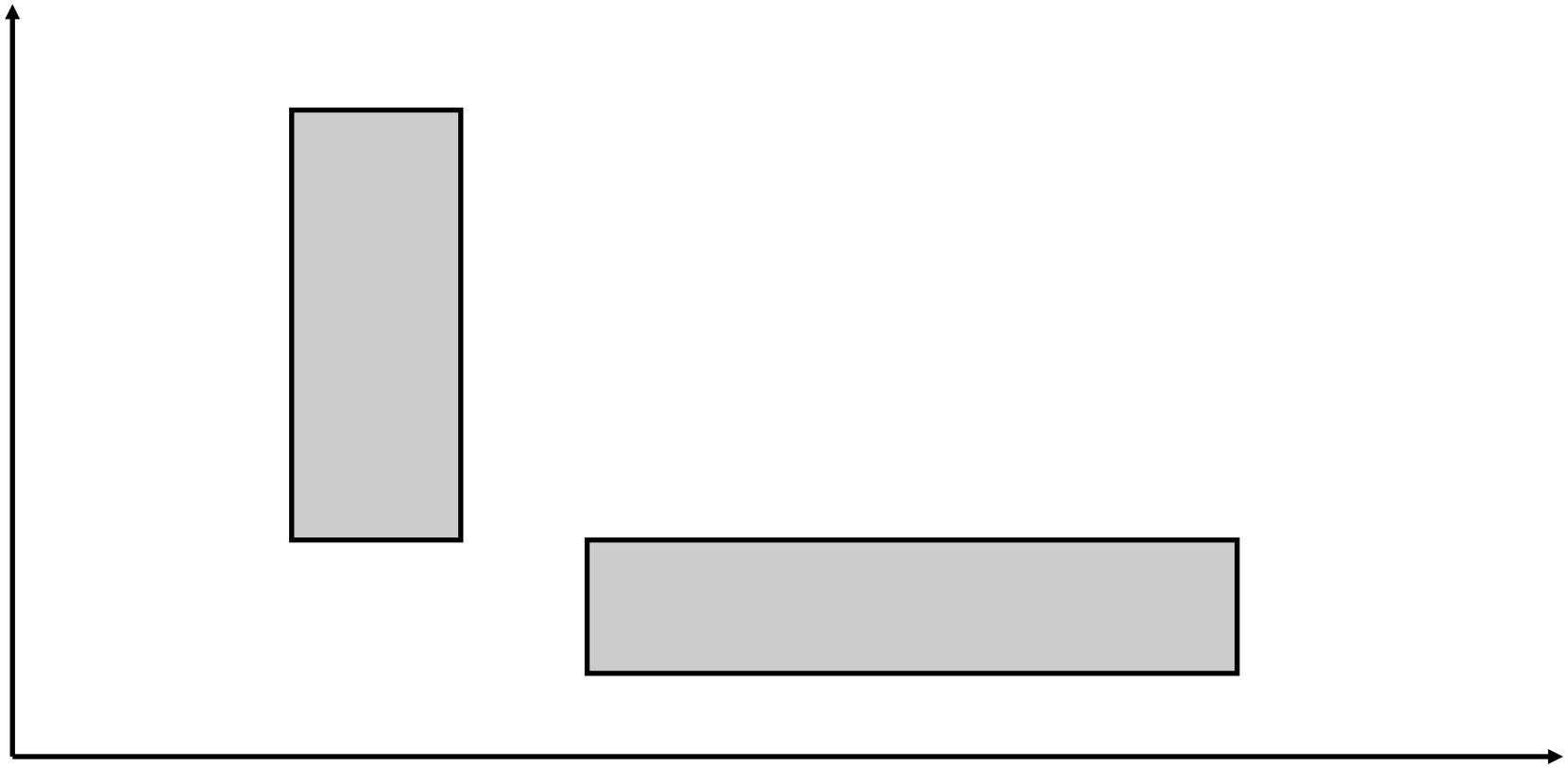


Separating Axis Theorem for Convex Objects

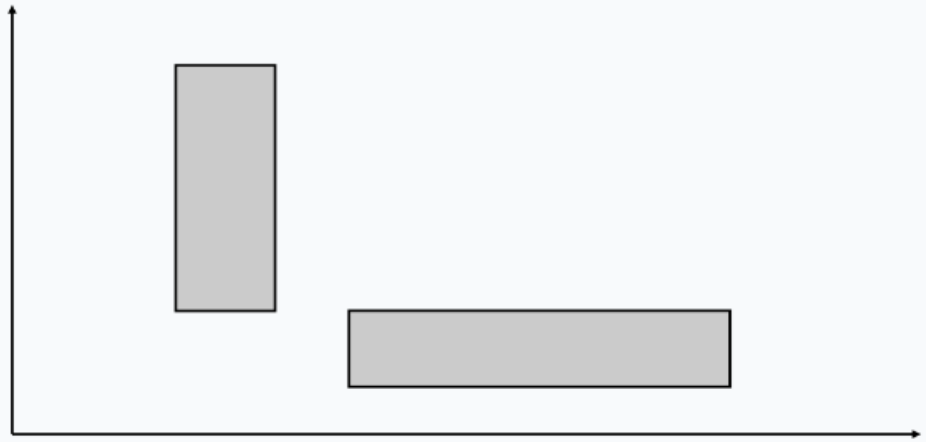
- Motivation
  - In 2D space, the Separating Axis Theorem states that:
  - Two convex polygons **A, B**, do not intersect iff there exists a line (separating axis) such that the projections of the two polygons onto the line do not intersect.
  - This is equivalent to stating that there exists a line (separating line) that completely divides **A** on one side and **B** on the other side. This line is perpendicular to the separating axis.



How many possible separating axes do we have to check?

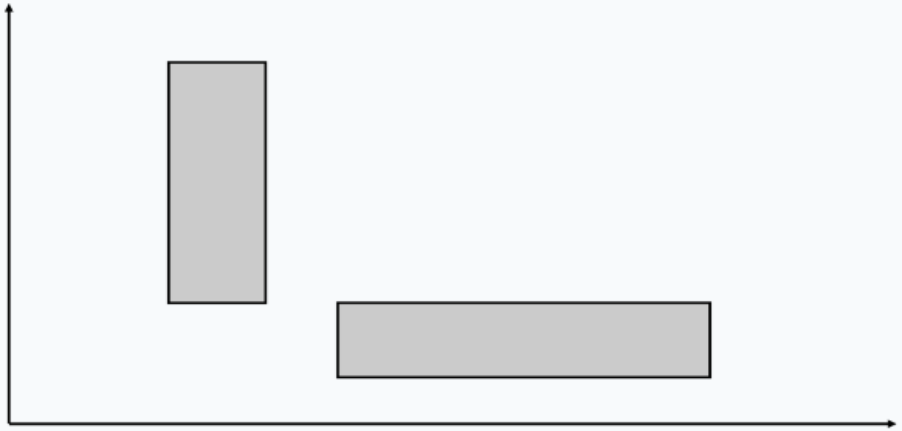


How many separating axes do we have to check to figure out if 2 AABBs are in collision?



- 1
- 2
- 4
- 8
- None of the above

How many separating axes do we have to check to figure out if 2 AABBs are in collision?



1
0%

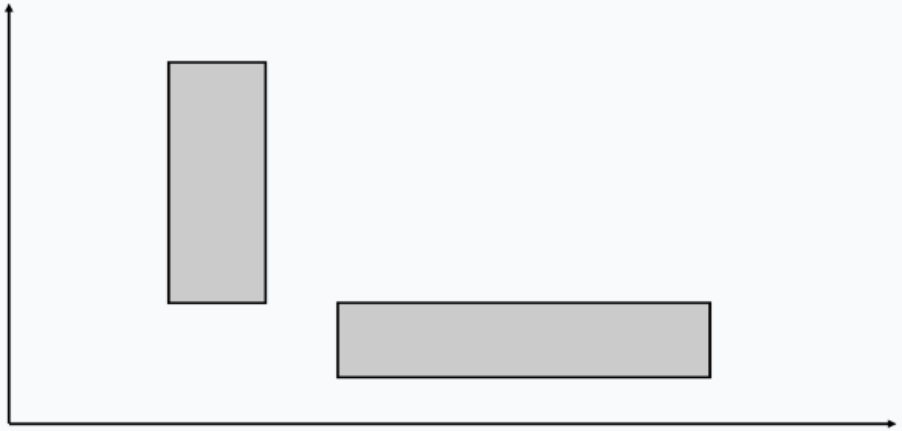
2
0%

4
0%

8
0%

None of the above
0%

How many separating axes do we have to check to figure out if 2 AABBs are in collision?



1
0%

2
0%

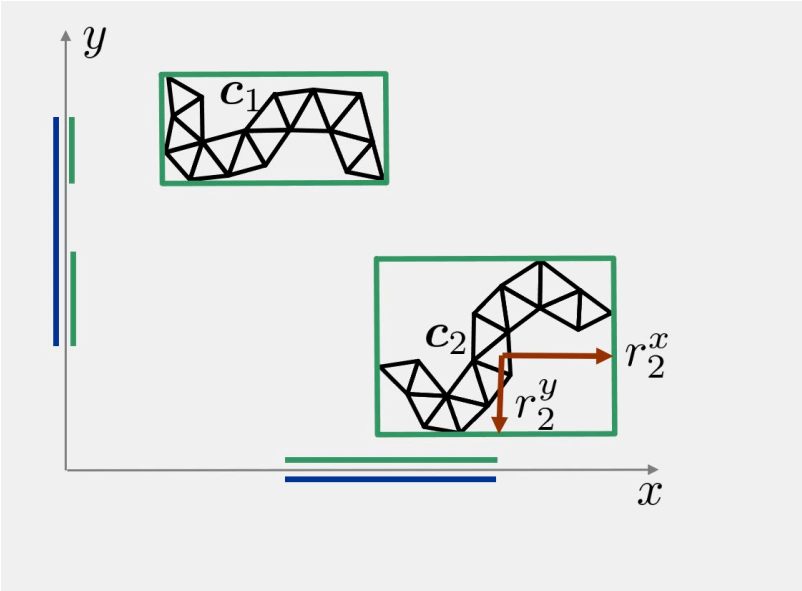
4
0%

8
0%

None of the above
0%

Axis-Aligned Bounding Boxes (2 possible separating axis)

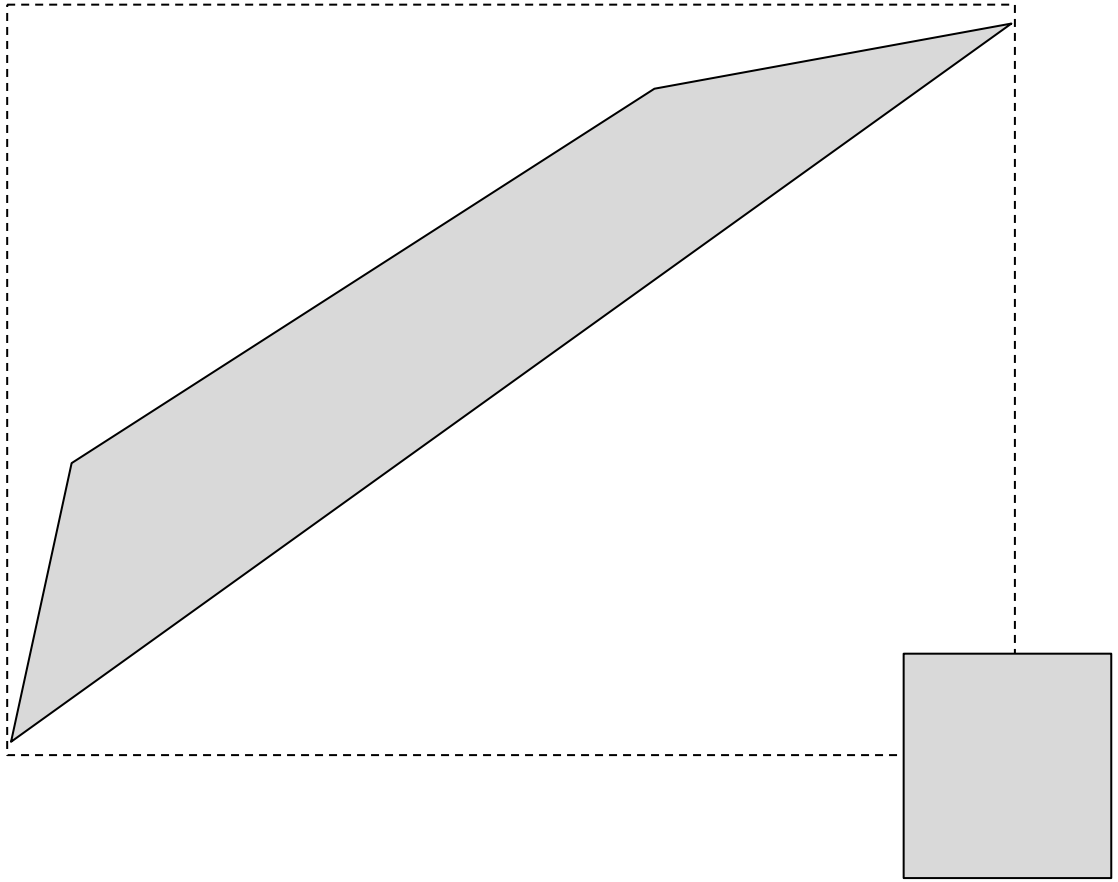
- AABBs are represented by
  - The center positions $\boldsymbol{c}$
  - The radii r^x, r^y



- Two AABBs in 2D do not overlap if $\left|(\boldsymbol{c}_1 - \boldsymbol{c}_2)\begin{pmatrix}1\\0\end{pmatrix}\right| > r_1^x + r_2^x$ OR $\left|(\boldsymbol{c}_1 - \boldsymbol{c}_2)\begin{pmatrix}0\\1\end{pmatrix}\right| > r_1^y + r_2^y$

Axis-Aligned Bounding Boxes Limitations

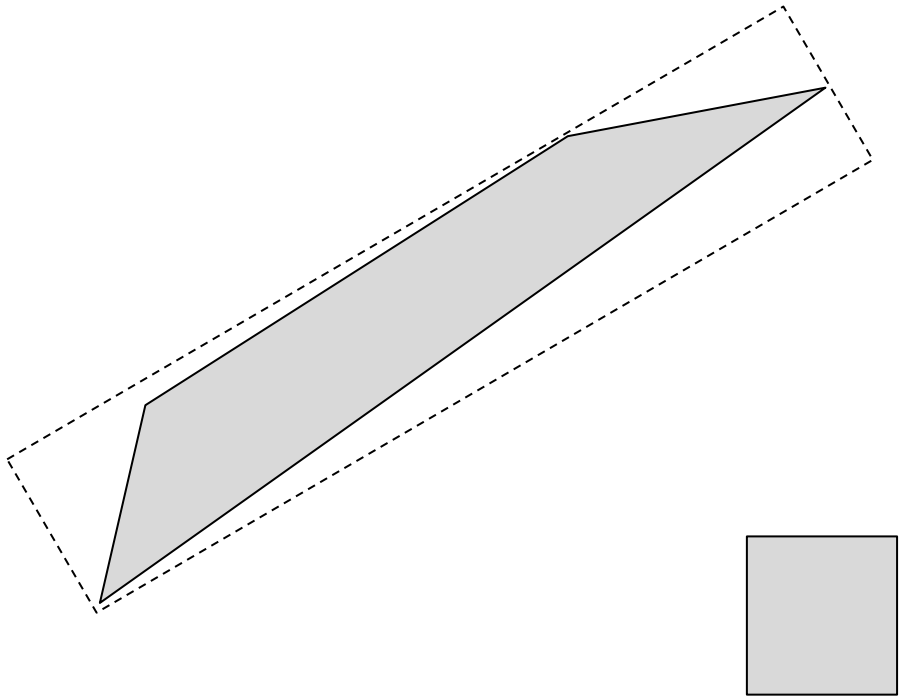
- AABBs can poorly approximate some objects



Are these polygons even close?

Oriented Bounding Boxes

- Do **not** align bounding box with axes
- Perform a *best* rectangular approximation: OBB



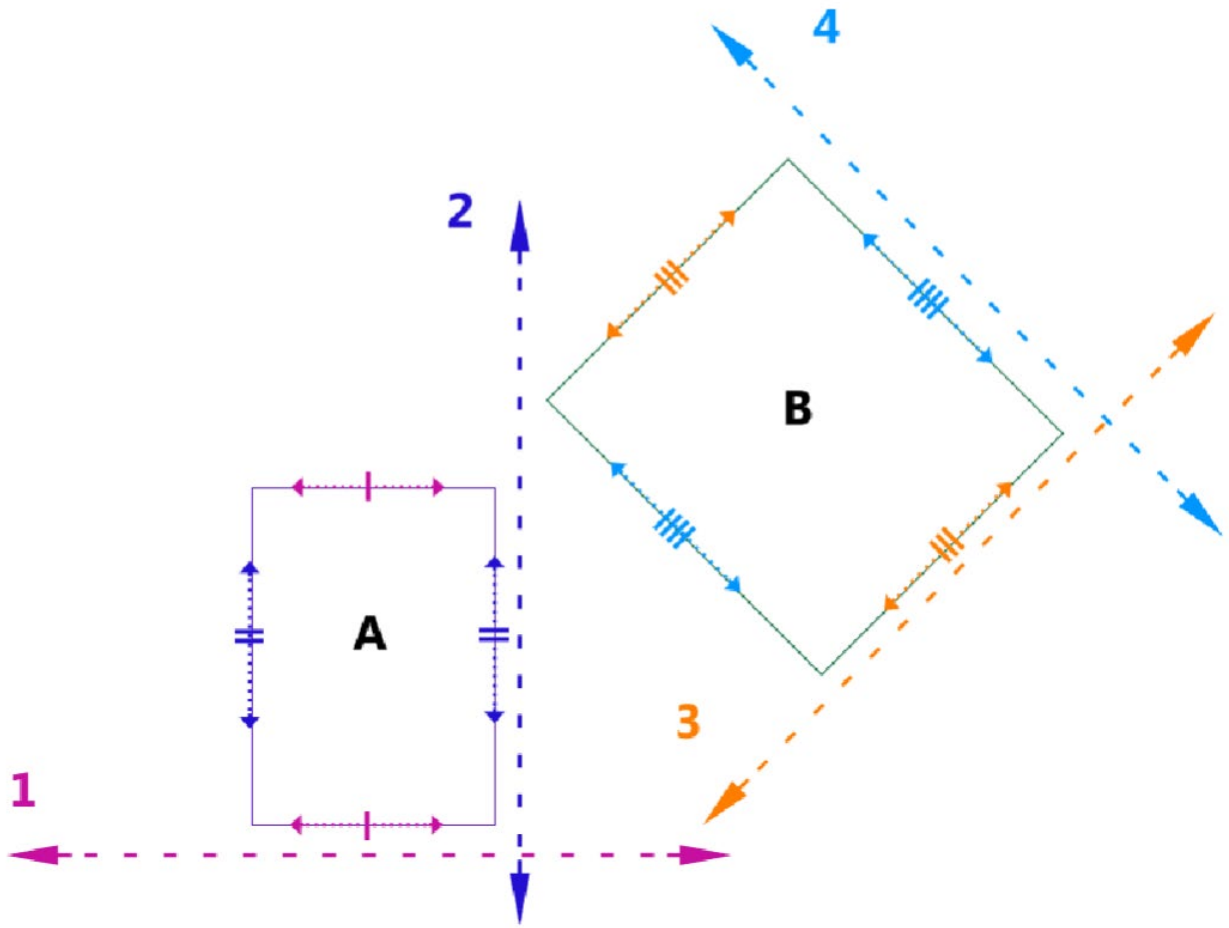
Bounding boxes no longer intersect

Oriented Bounding Boxes

- Similar to AABB, but with flexible orientations
- OBBs have not to be aligned with respect to each other or to a coordinate system
- In contrast to AABBs,
  - OBBs can be rotated with an object
  - OBBs are more expensive to check for overlap

How many possible separating axis?

OBB Overlap Test in 2D (4 potential separating axes)

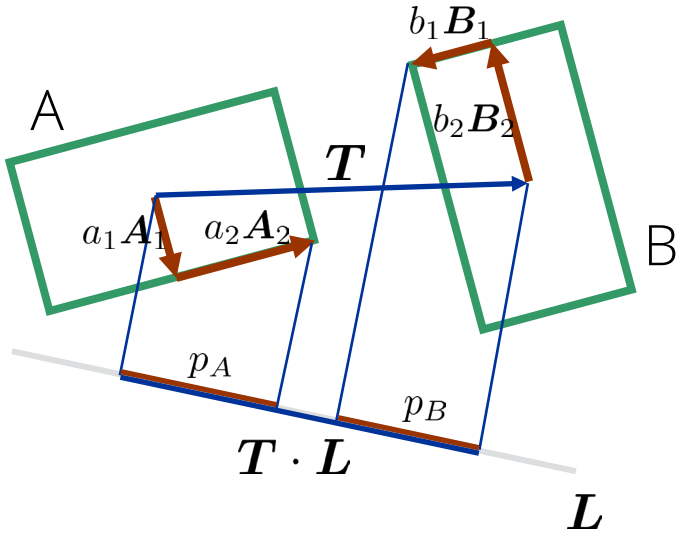


General Separating Axis Test in 2D

- $\boldsymbol{A}_1, \boldsymbol{A}_2, \boldsymbol{B}_1, \boldsymbol{B}_2$ are normalized axes of A and B
- a_1, a_2, b_1, b_2 are radii of A and B
- $\boldsymbol{L}$ is a normalized direction
- $\boldsymbol{T}$ is the distance of centers of A and B
- A and B do not overlap in 2D iff

$$p_A = a_1\boldsymbol{A}_1\boldsymbol{L} + a_2\boldsymbol{A}_2\boldsymbol{L}$$
$$p_B = b_1\boldsymbol{B}_1\boldsymbol{L} + b_2\boldsymbol{B}_2\boldsymbol{L}$$

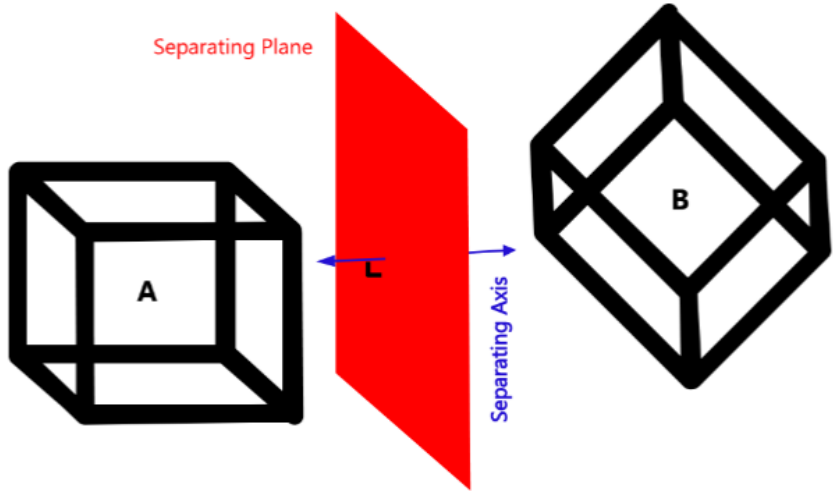
$$\exists \boldsymbol{L} : \boldsymbol{T} \cdot \boldsymbol{L} > p_A + p_B$$

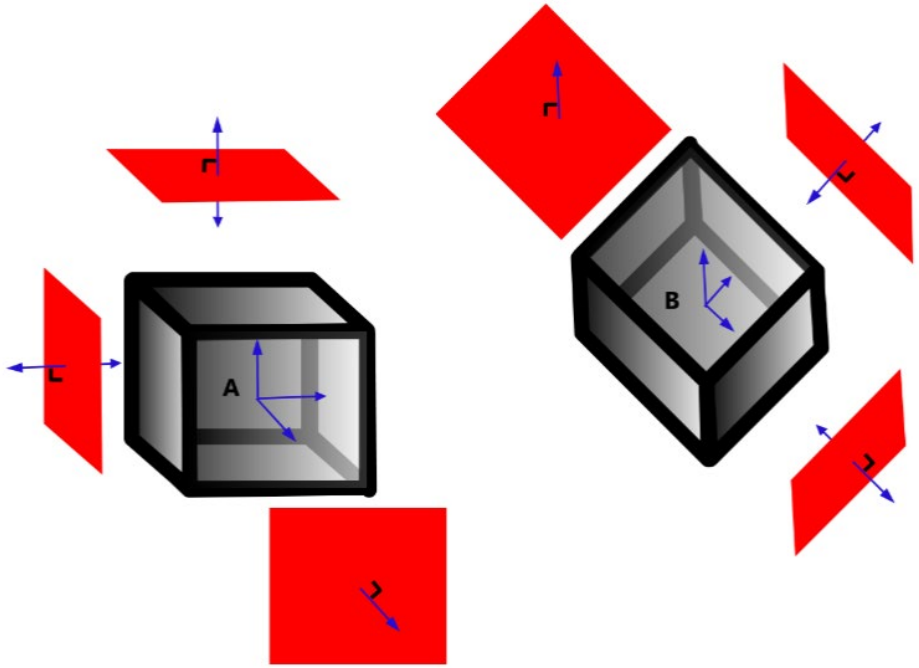


3 Dimensional OBBs

- The separating plane in 3D space contrasts with the separating line in 2D space.

- The separating plane separates box A from box B.



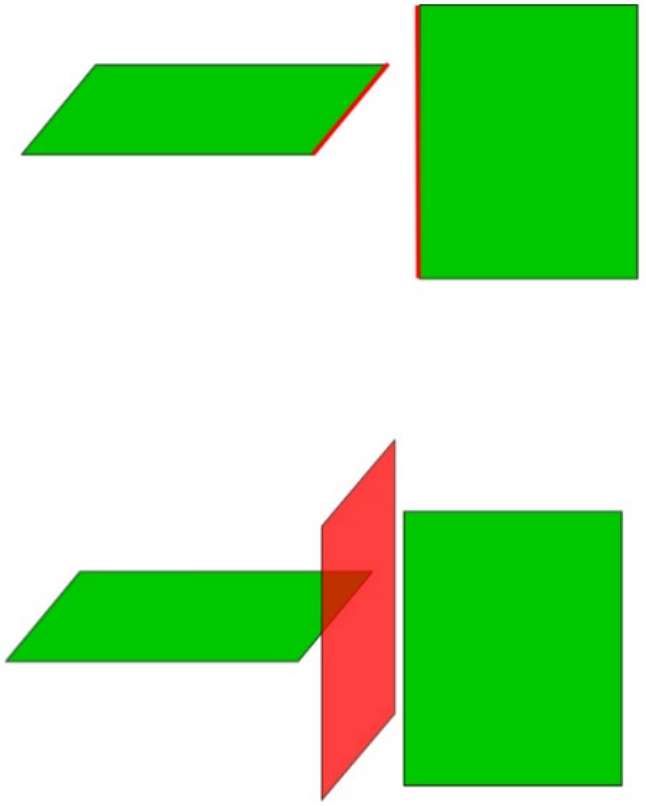


- 6 unique separating axes

Is this enough?

Just checking the 6 possible faces of each box is not enough

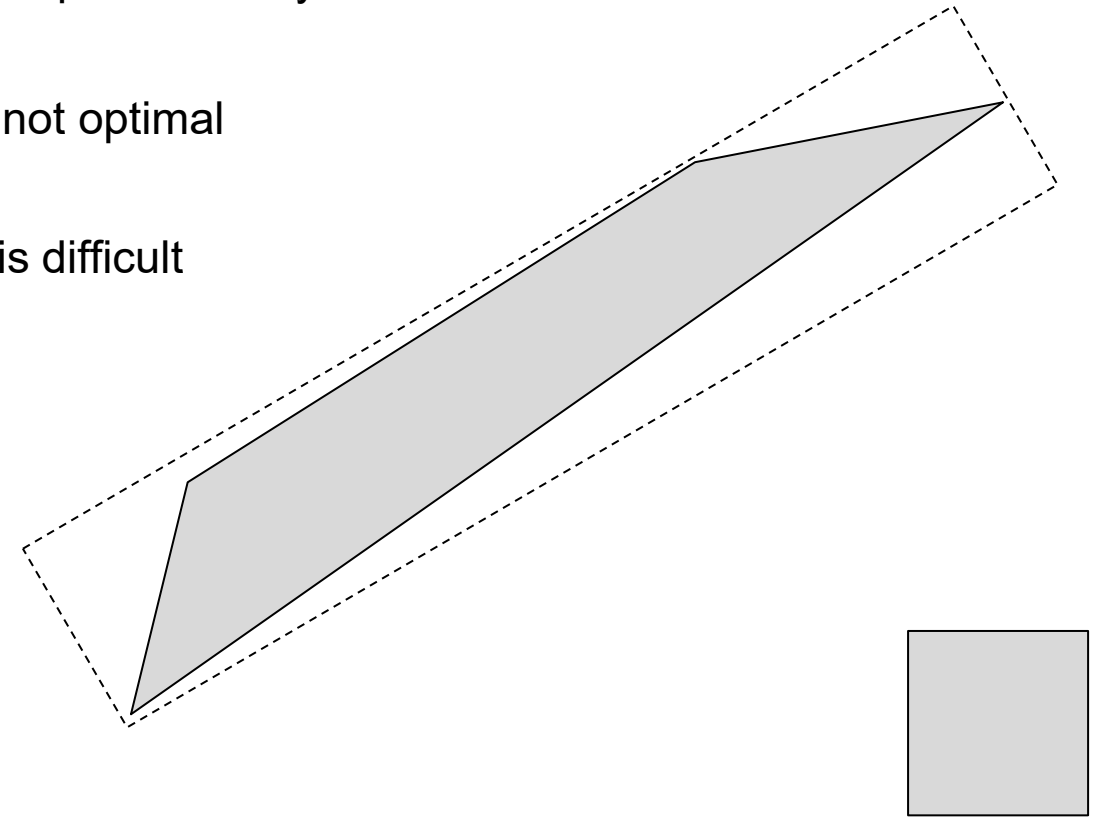
- The above image depicts two non-intersecting green faces.
- A separating plane that separates the two faces is spanned by the axes of the two edges highlighted in red.
- The bottom image depicts a separating plane



- We also have to consider the planes defined by each pair of edges. Each cube has 3 sets of parallel edges, so we need to check 3*3 = 9 extra separating axis for a total of 15 checks.

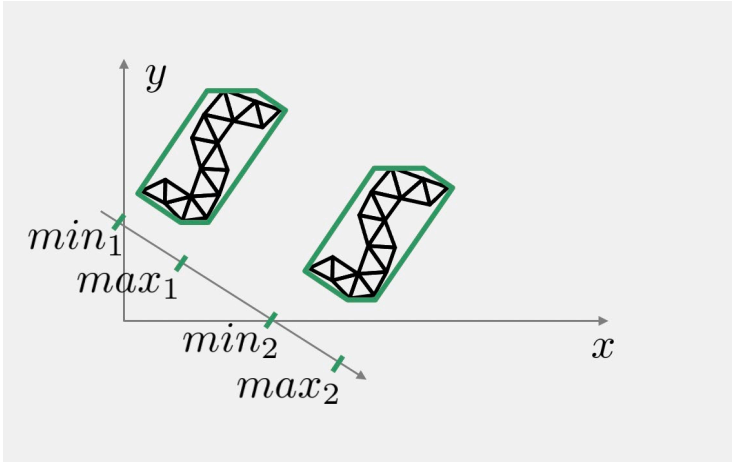
Oriented Bounding Boxes

- A good OBB is tough to compute
  - Principal component analysis over vertex distribution
  - Close, but not optimal
- Minimal OBB is difficult



Discrete-Orientation Polytops k-DOP

- Convex polytope whose faces are determined by a fixed set of normals
- *k*-DOPs are represented by
  - *k* / 2 normals
  - *k* / 2 min-max intervals



$$\exists \text{direction} : max_1 < min_2 \vee min_1 < max_2$$

- If any pair of intervals does not overlap, *k*-DOPs do not overlap

Discrete-Orientation Polytops k-DOP

- AABB is a 4-DOP. Are all 4-DOPs AABBs?
- All k-DOPs share the same pre-defined normal set
- Only min-max intervals are stored per *k*-DOP
- Larger *k* improves the approximation quality
- Intersection test is more expensive for larger *k*

Convex Hull approximation

- Approximate everything with its *convex hull*
  - *Graham's scan: O(n log n)*
  - *Giftwrap algorithm: O(nh)*
- Reduction to intersecting convex polyhedral

  *O(n log n)**

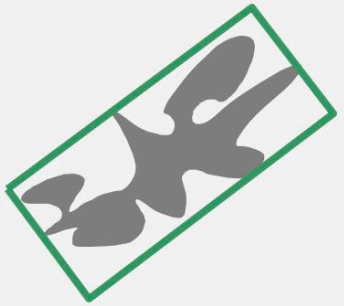
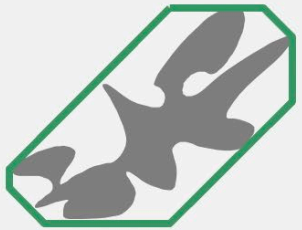
* Muller, David E., and Franco P. Preparata. "Finding the intersection of two convex polyhedra." *Theoretical Computer Science* 7.2 (1978): 217-236.

General Separating Axis Test

- For polyhedral objects, only a few axes have to be tested
  - Axes parallel to face normals of A
  - Axes parallel to face normals of B
  - Axes parallel to all cross products of edges of A and B

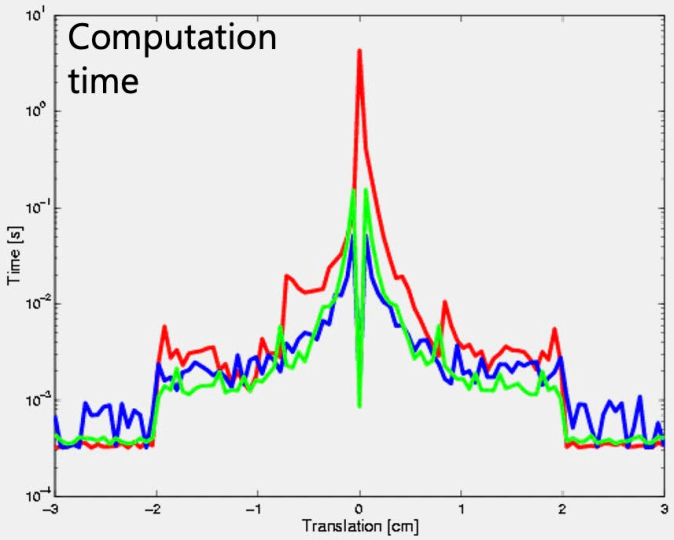
  (The proof is long and provided in readings)
- In case of 3D OBBs, $3 + 3 + 3 \cdot 3$ axes have to be tested
- Efficient and general overlap test
- Does not provide information on the intersection geometry

Collision Detection Libraries

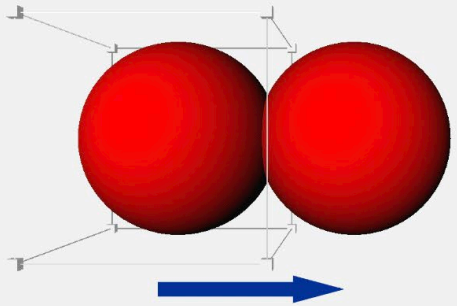
SOLID	***RAPID***	***QuickCD***
Axis-aligned bounding box	Object-oriented bounding box	*k* discrete orientation polytope
		
van den Bergen Eindhoven University 1997	Gottschalk et al. University of North Carolina 1995	Klosowski et al. University of New York 1998

Comparison

– Two spheres with radius 1 and 10000 triangles per sphere

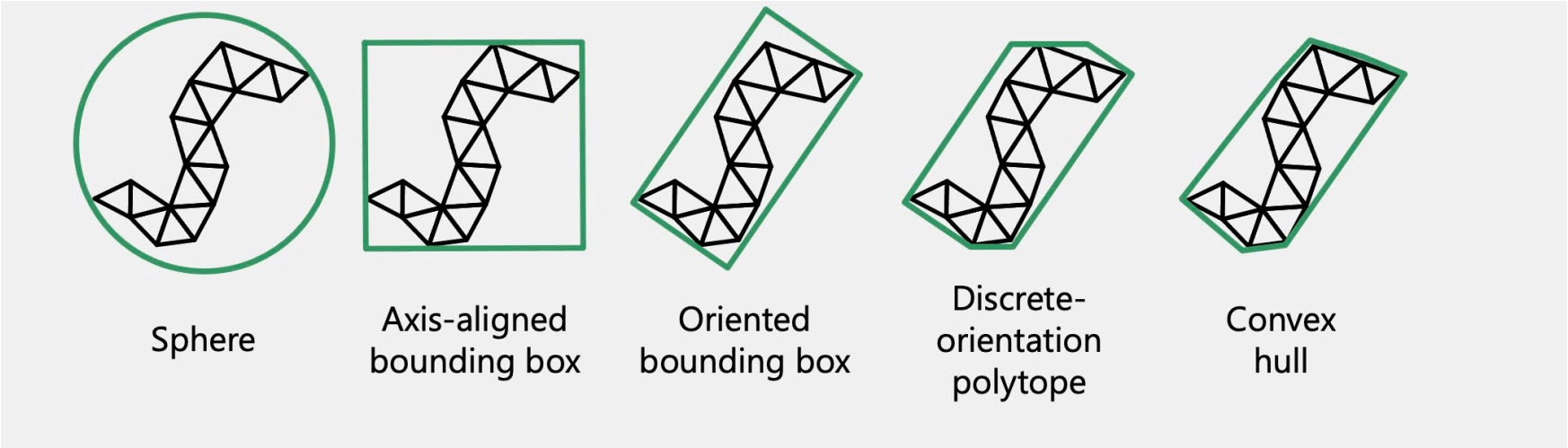


Distance between sphere centers



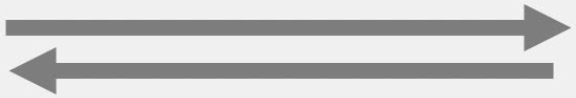
8-DOP
OBB
ABB

Optimal Bounding Volumes



It depends……

Tight approximation



Efficient overlap test

Summary: Bounding Volumes

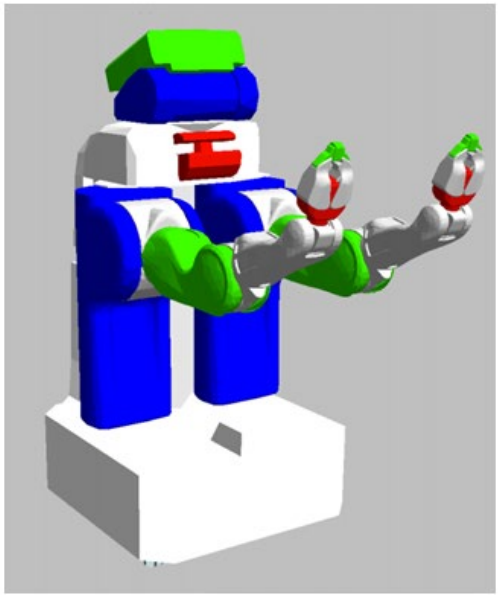
- Simple geometries that encapsulate complex objects
- Efficient overlap rejection test
- Tight object approximation, memory efficient, fast overlap test
- Spheres, AABBs, OBBs, k-DOPs, convex hulls

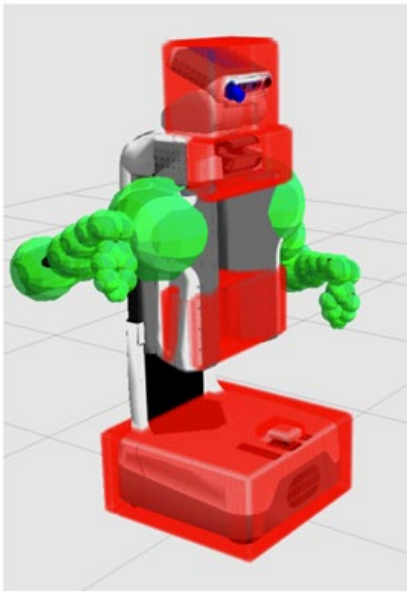
Overview

- Motivation
- Bounding Volumes
- Hierarchies of Bounding Volumes
- Spatial Partitioning

What happens with intersections?

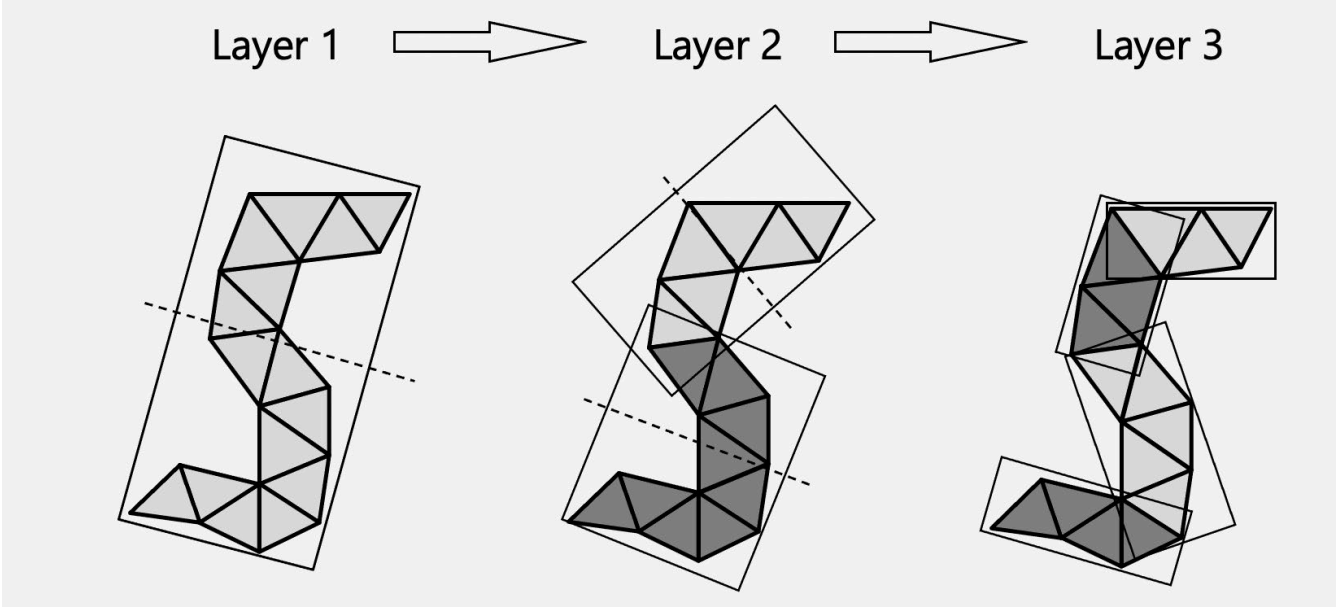
- If the bounding volume has an intersection, what do we do then?
  - Perform a naïve collision check
  - Use a hierarchical approximation





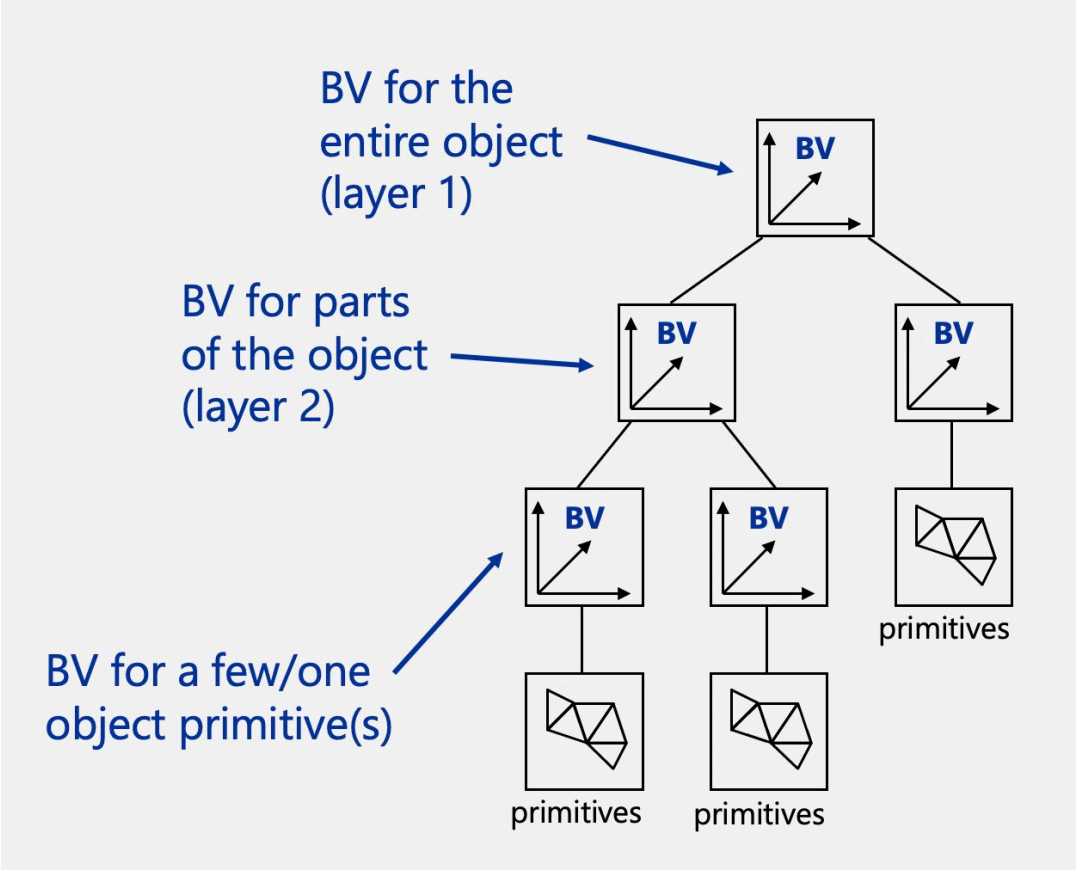
Bounding Volume Hierarchies

- Efficient overlap rejection test for parts of an objects
- E.g., object can be subdivided to build a BVH

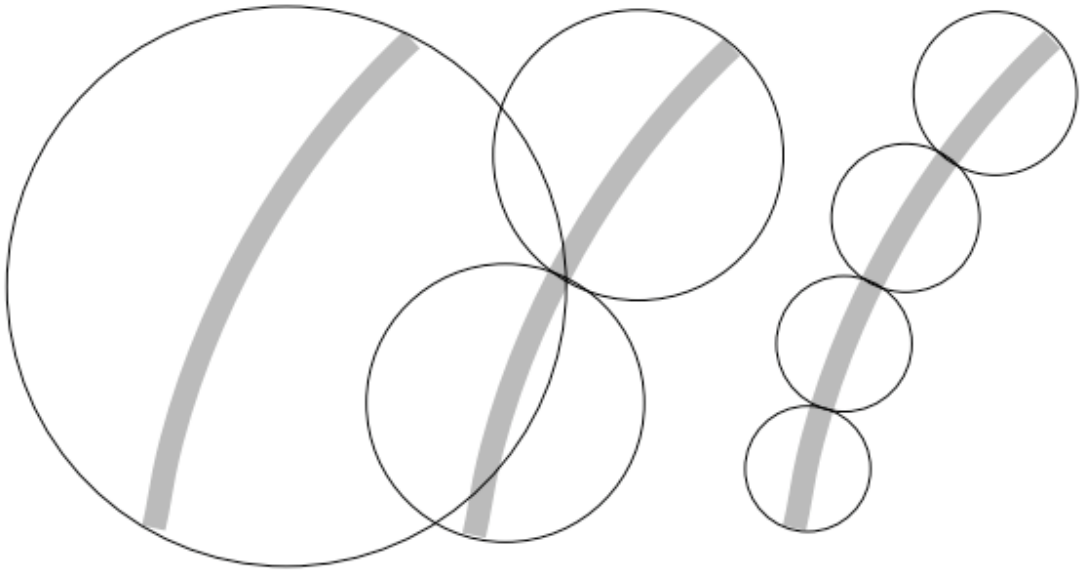


Data Structure

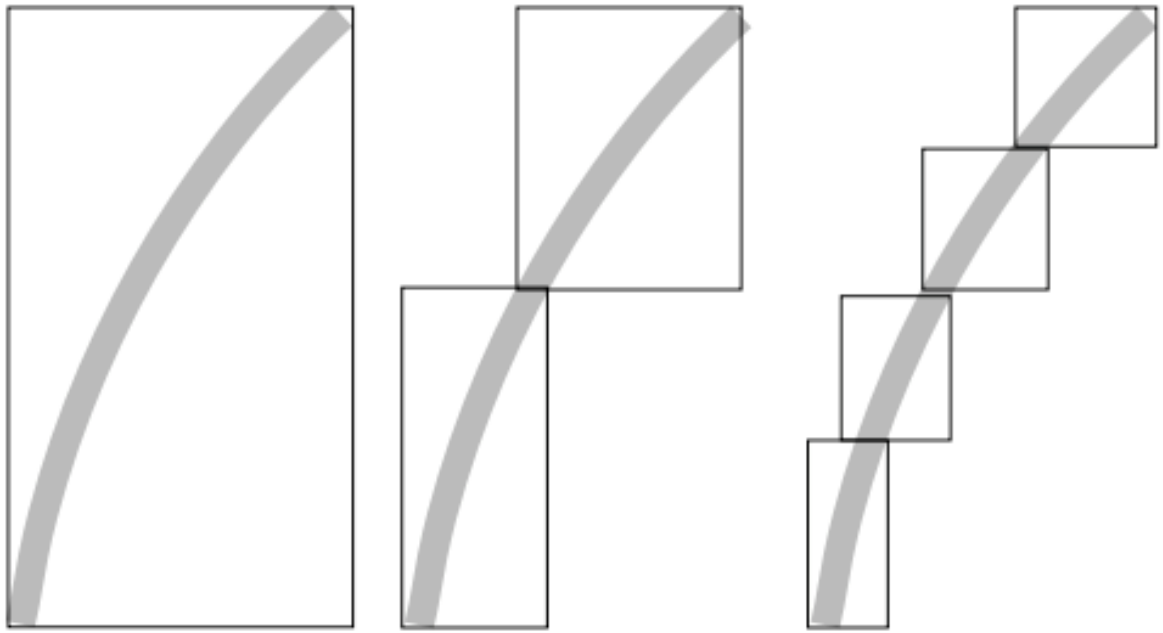
- Tree of bounding volumes
- Nodes contain BV information
- Leaf nodes contain object primitives



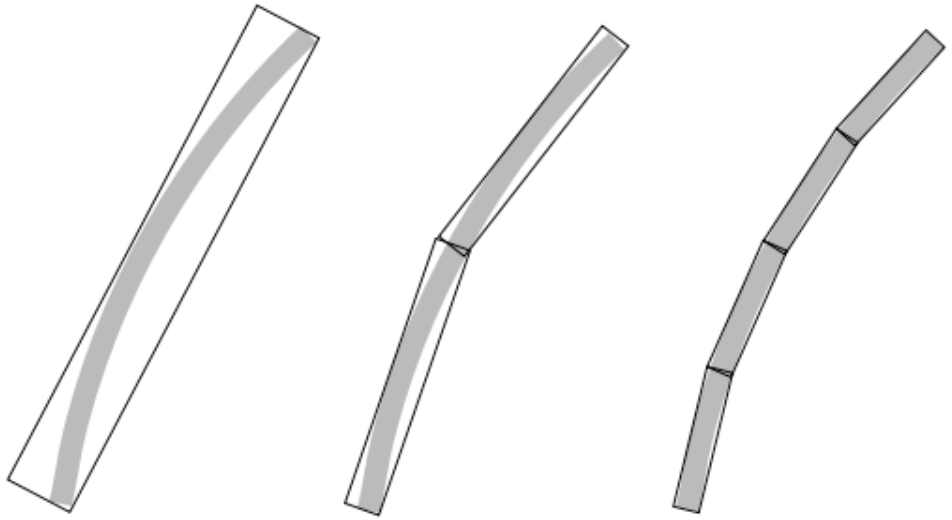
Sphere-trees



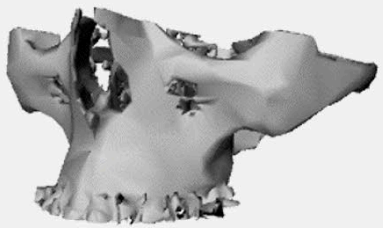
Example: AABB-trees



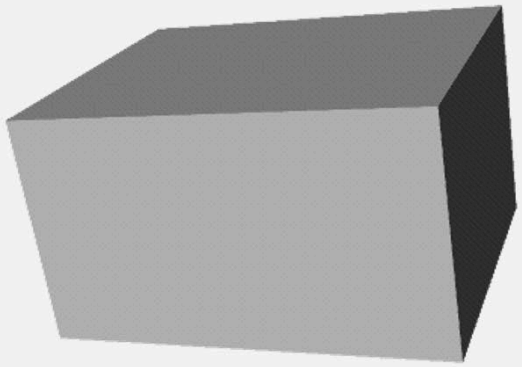
Example: OBB-trees



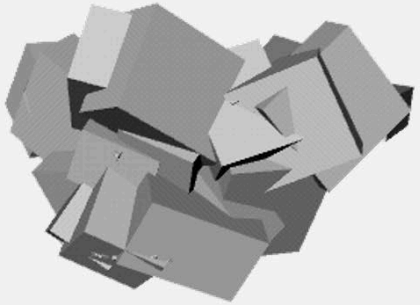
Example: OBB-trees



Object



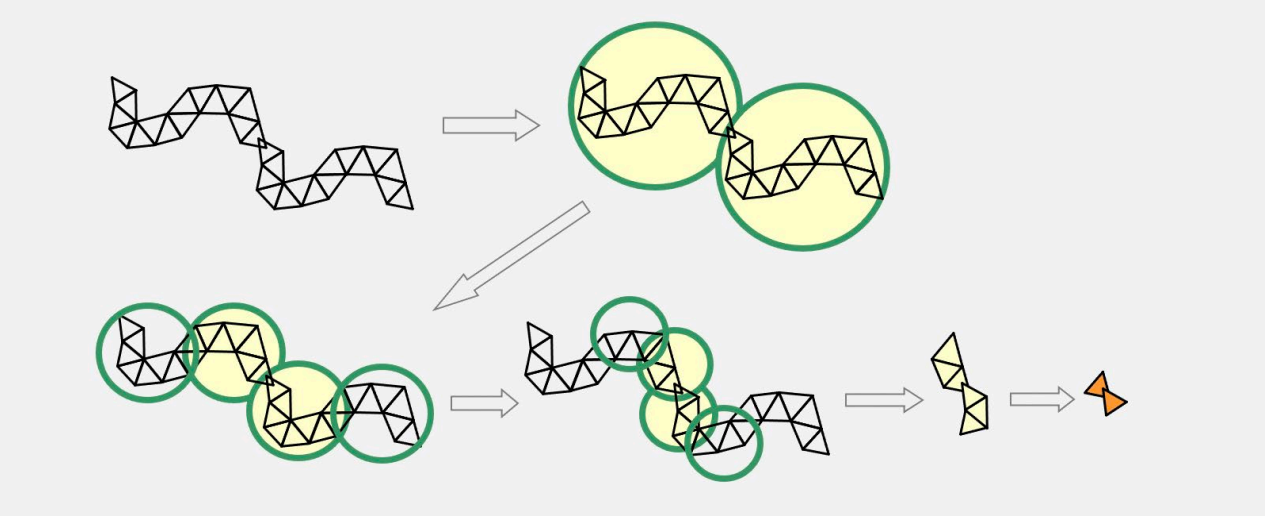
OBB
Layer 1



OBBs
Layer *n*

Overlap for BV Trees

- If BVs in a layer overlap, their children are checked
- At a leaf, primitives are tested with BVs and primitives
- Efficient culling of irrelevant object parts



Pseudo-code

1. Overlap test for two parent nodes (root)
2. If no overlap then "no collision" else
3. All children of one parent node are checked against children of the other parent node
4. If no overlap then "no collision" else
5. If at leaf nodes then "collision" else go to 3.

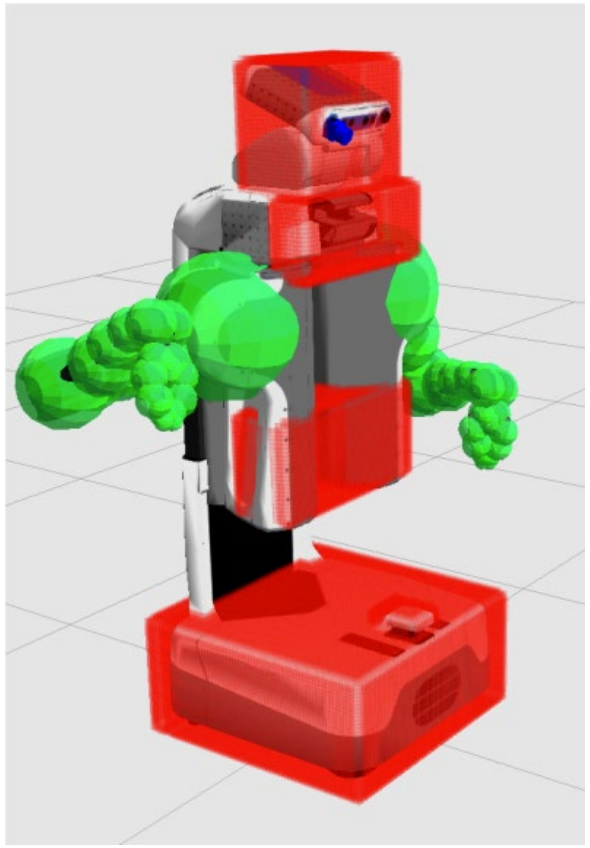
- Step 3. checks BVs or object primitives for intersection
- Required tests: BV-BV, (BV–primitive), primitive-primitive

Implementation - OBB tree, triangulated object

- **All tests based on separating axis test**
- Box-box
  - 3 + 3 + 3 · 3 axes have to be tested
- (Box-triangle)
  - 3 + 1 (face normals) + 3 · 3 (cross products of edges) tests
- Triangle-triangle
  - a) 1 + 1 (face normals) + 3 · 3 (cross products of edges) tests
  - b) Testing all edge-edge pairs → 6 edge-triangle tests → 5 tests sufficient (intersections occur in pairs)

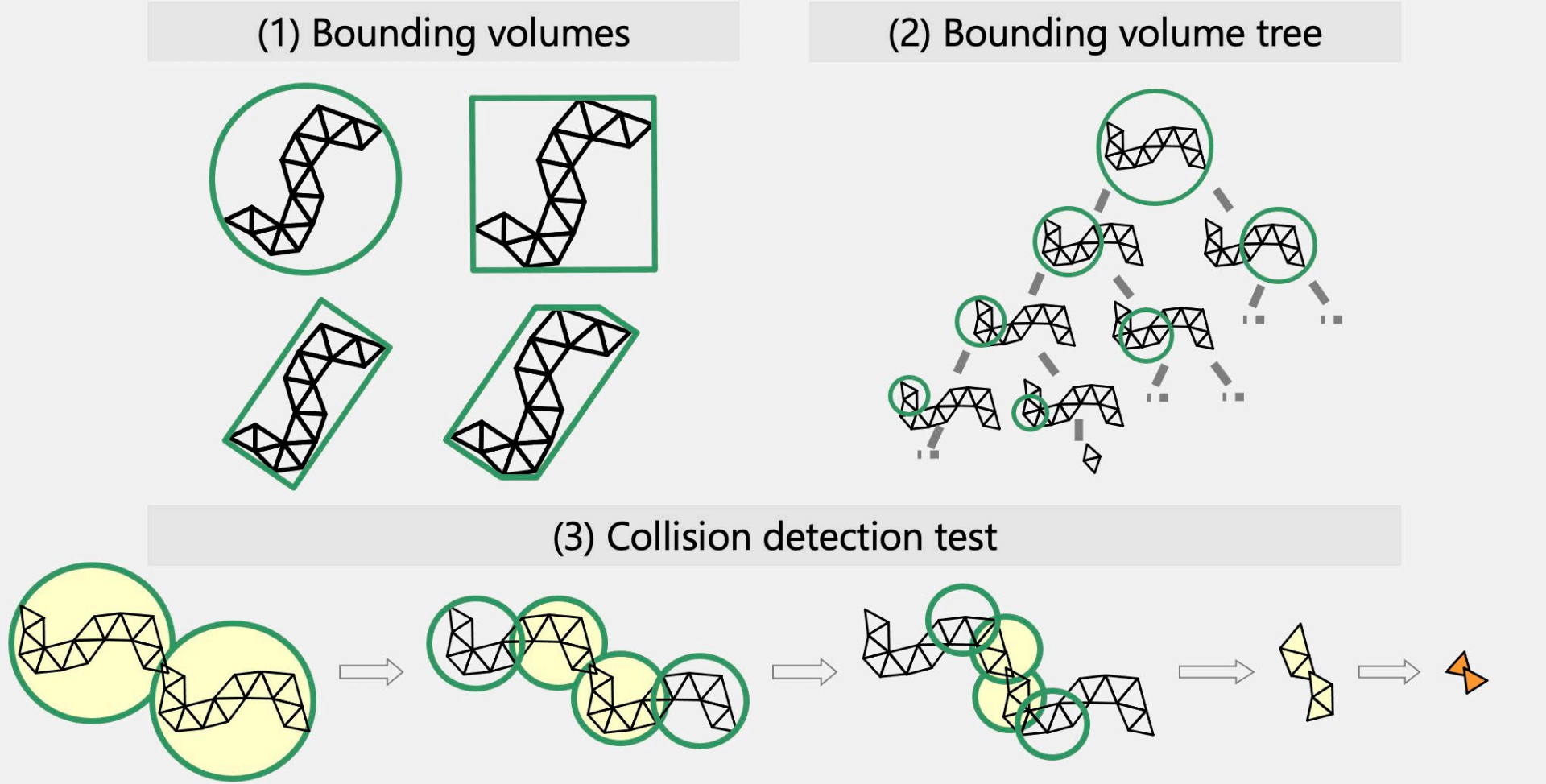
Example: Bounding Volume Hierarchies

- Use any combination of BVs!



- What about the environment?

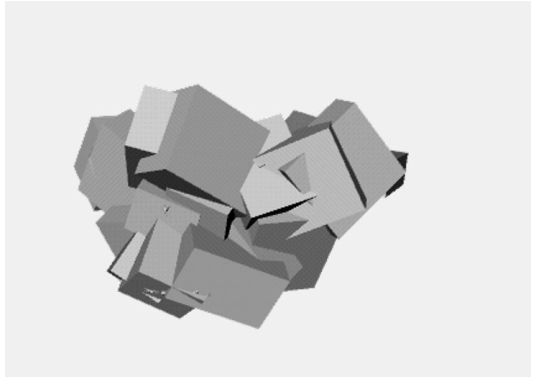
Summary - BVHs



Construction of BVHs

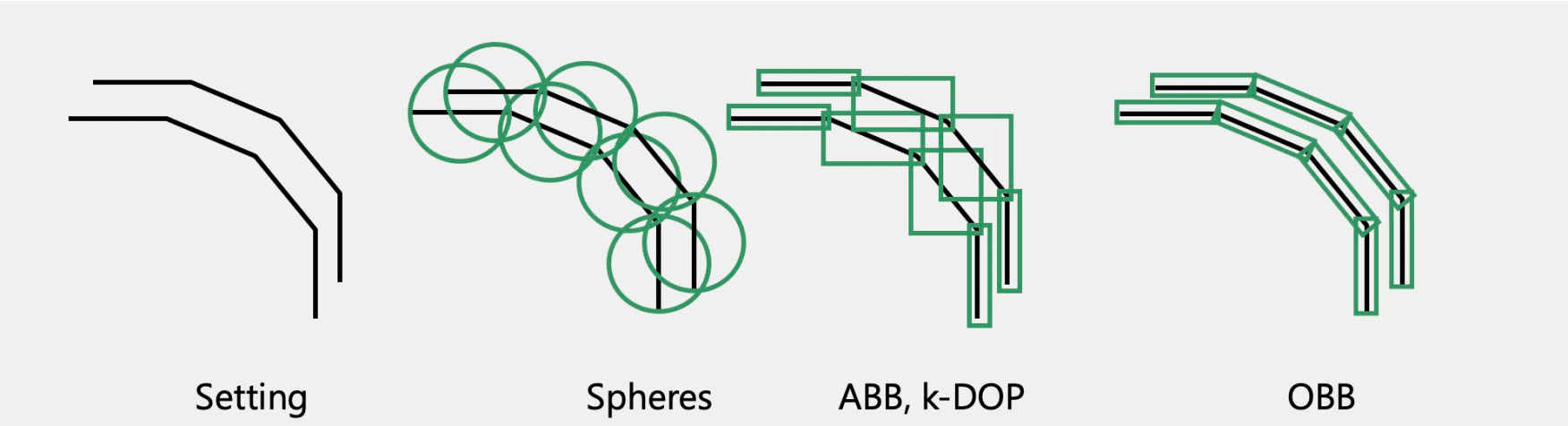
- Goals
  - Balanced tree
  - Tight-fitting bounding volumes
  - Minimal redundancy

    (primitives in more than one BV per level)
- Parameters
  - BV type
  - Top-down / bottom-up
  - What and how to subdivide or merge: primitives or BVs
  - How many primitives per leaf in the BV tree
  - Re-sampling of the object



Close Proximity

- In case of close proximity,
  - Quality of higher layers influences the collision detection performance
  - Quality of lower layers BV approximations is less critical



Hierarchical Bounding Volume (HBV) Tradeoffs

- The total cost of collision checking a bounding volume hierarchy:

$$N_{bv} \cdot C_{bv} + N_{ex} \cdot C_{ex}$$

- N_{bv}: number of bounding volume overlap checks
- C_{bv} : cost of a bounding volume overlap check
- N_{ex} : number of exact intersection checks
- C_{ex} : cost of an exact intersection check

Hierarchical Bounding Volume Tradeoffs

- The total cost of collision checking a bounding volume hierarchy:

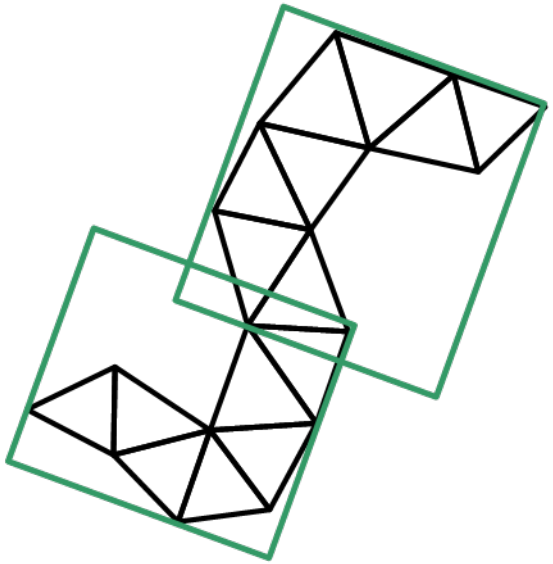
$$N_{bv} \cdot C_{bv} + N_{ex} \cdot C_{ex}$$

- A tight BV reduces N_{bv} and N_{ex}
- For complex geometries, tight BV increases C_{bv}

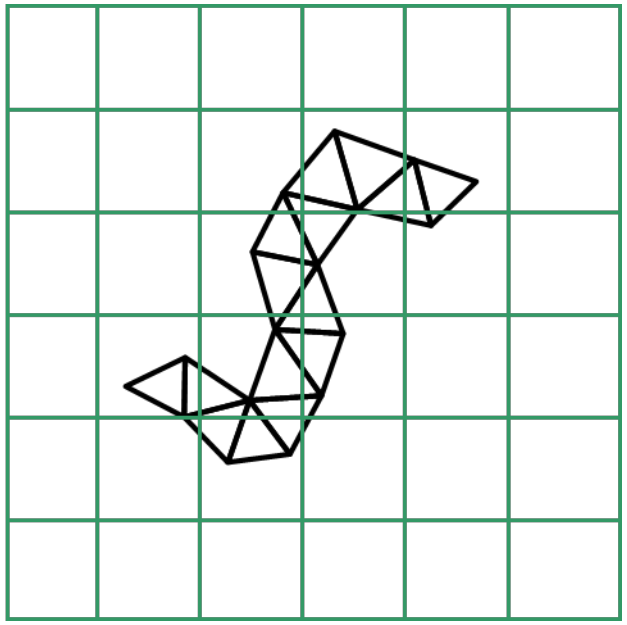
Overview

- Motivation
- Bounding Volumes
- Hierarchies of Bounding Volumes
- Spatial Partitioning

Model Vs Spatial Partitioning



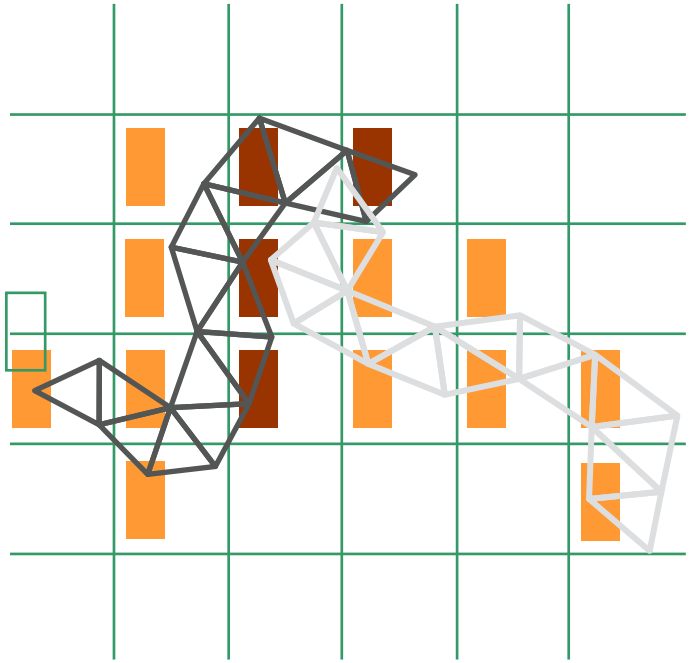
Model partitioning



Space partitioning

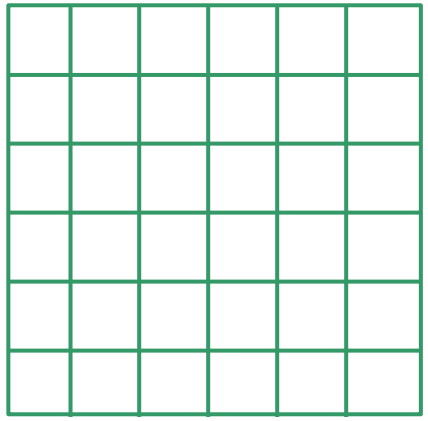
Motivation

- Restrict pairwise object tests to objects that are located in the same region of space
- Only objects or object primitives in the same region of space can overlap
- Efficient broad-phase approach for larger numbers of objects

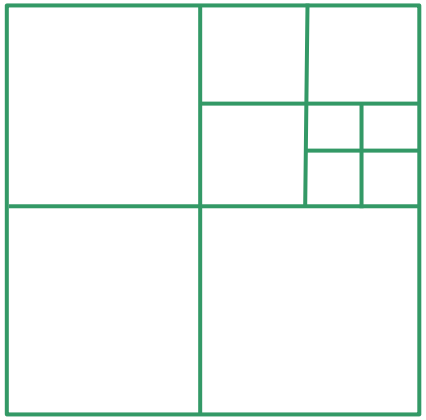


Spatial Data Structures

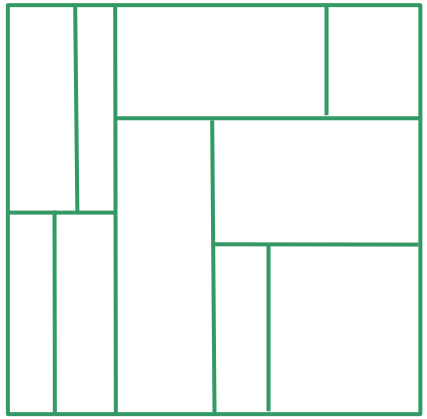
- Each cell contains information whether it is occupied by an object -> Possible collision!
- Information is updated for each object transformation
- quadtree, kd-tree, and BSP-tree are hybrid approaches
- They subdivide space, but are object-dependent



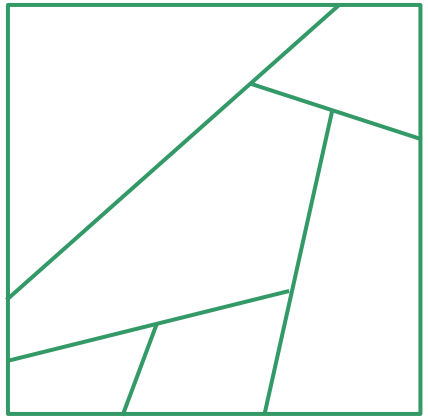
Uniform grid



Quadtree / Octree



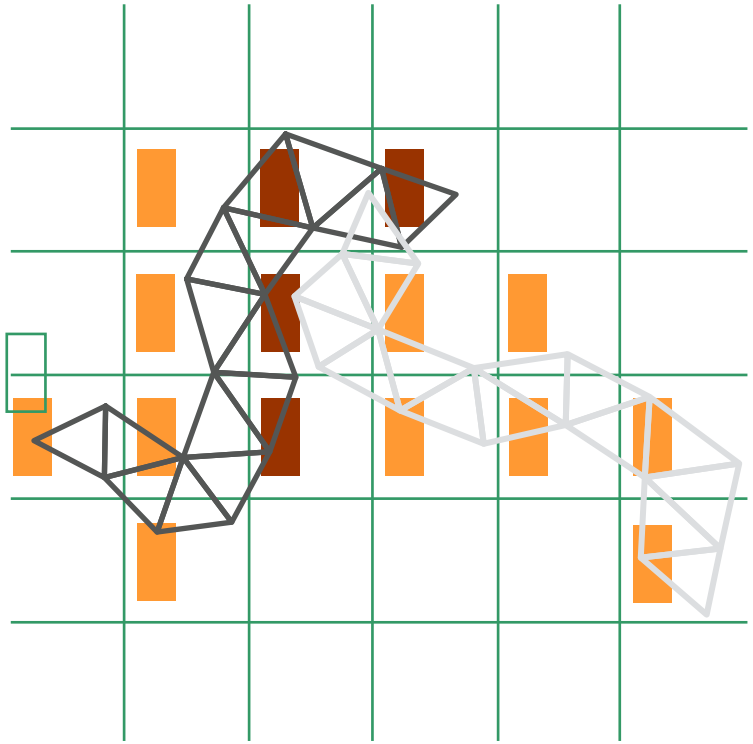
k-d tree



BSP tree

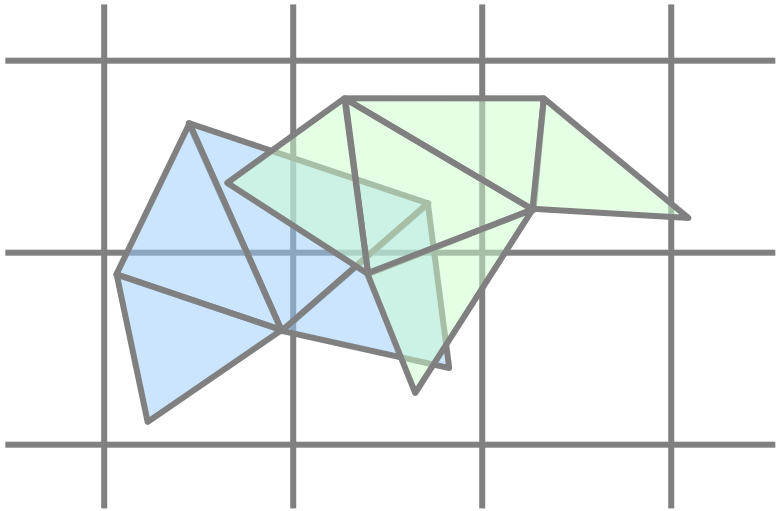
Uniform Grid: Basic idea

- Space is divided into cells
- Object primitives are placed into cells
- Object primitives in the same cell are checked for collision
- Pairs of primitives that do not share the same cell are not tested (trivial reject)



Setup

We can have an infinite uniform grid with hashing



Spatial data structure

Hash function:

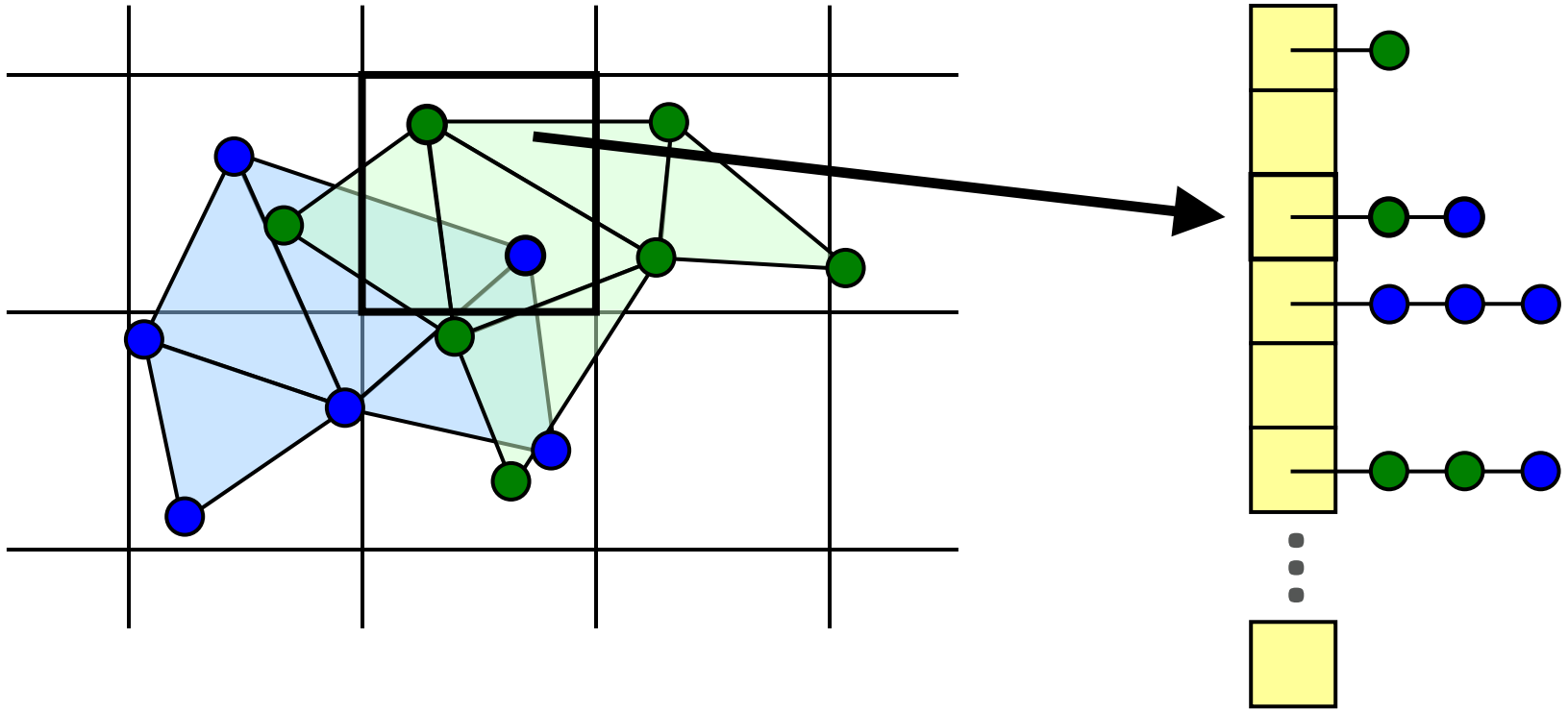
H(cell) → hash table index

Hash table

Representation / implementation

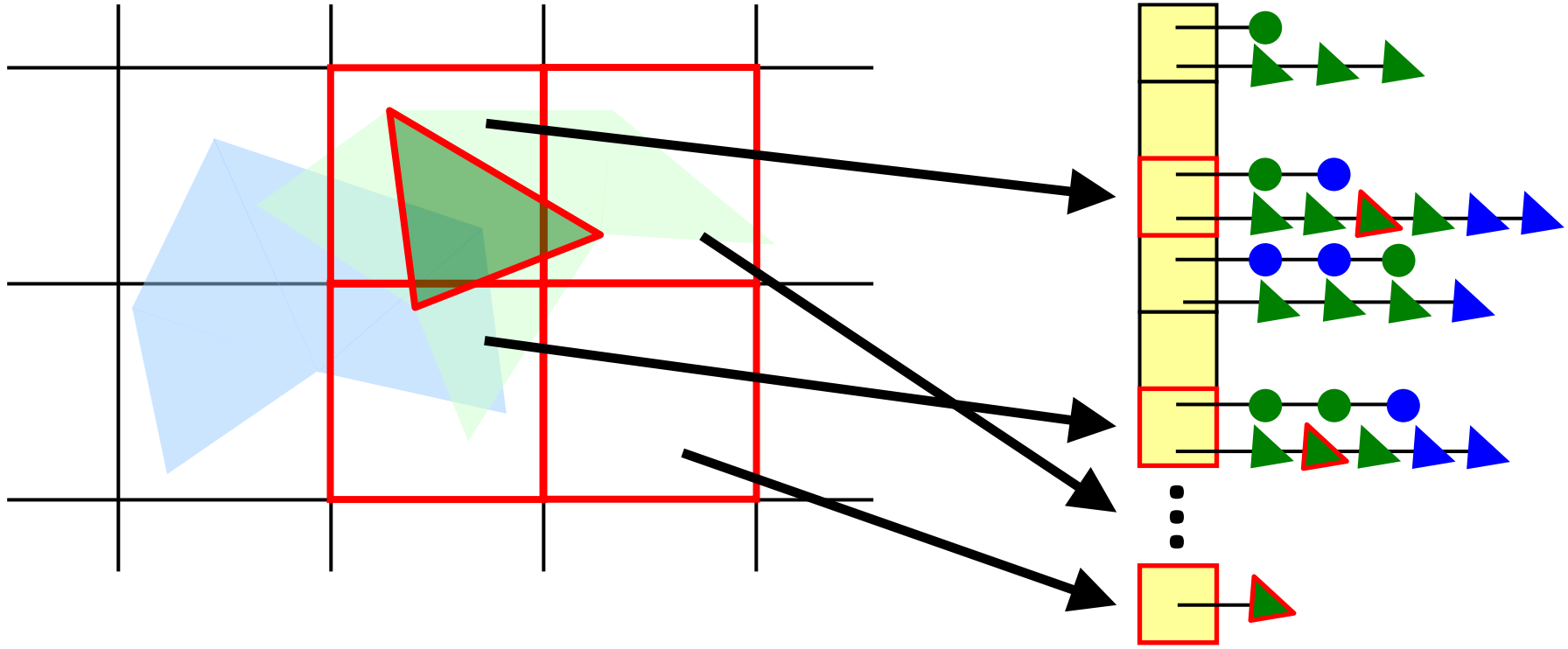
Stage 1

- All vertices are hashed according to their cell



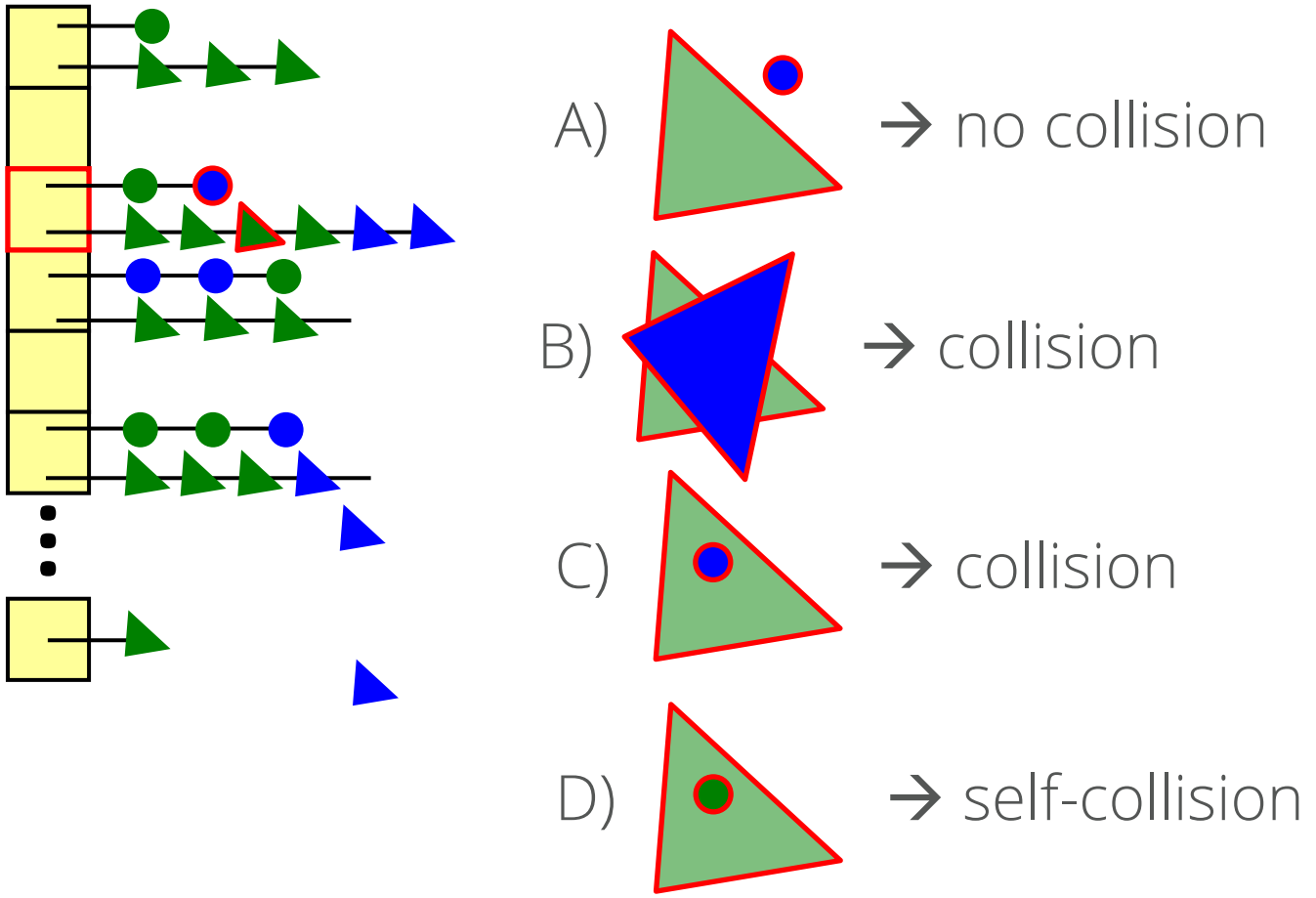
Stage 2

- All tetrahedrons are hashed according to the cells touched by their bounding box



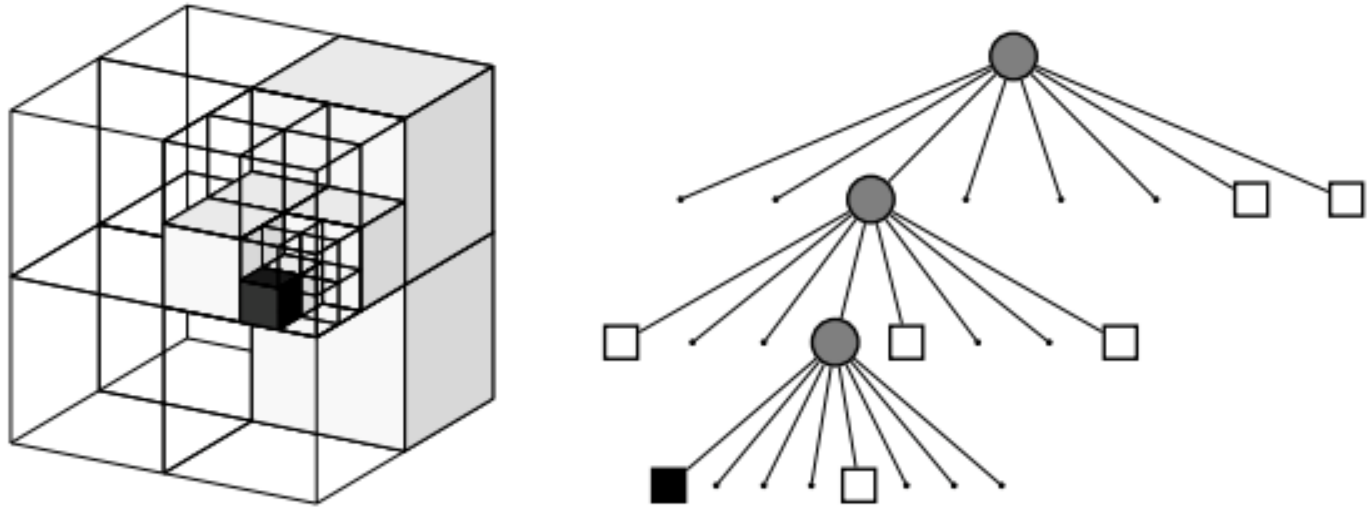
Stage 3

- Vertices and tetrahedrons in the same hash table entry are tested for intersection



Octree

- 3D spatial representation using voxels
  - 2D equivalent is a quadtree
- Voxels are empty (white), occupied (black), or mixed (gray)
- Recursively subdivides “gray” space down to a specific resolution

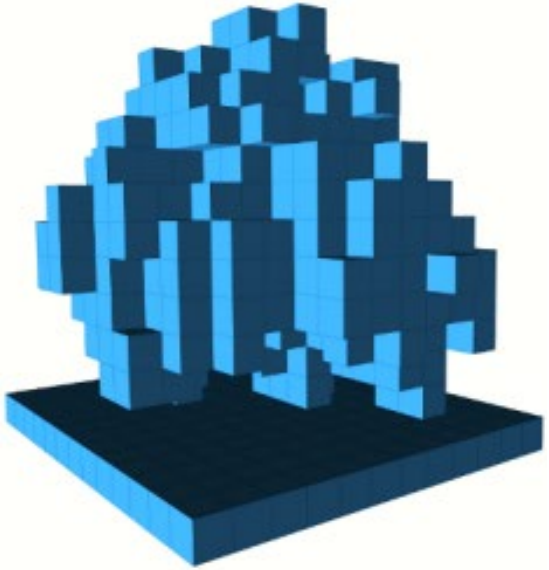


Octree

- Collisions are detected when object occupies an obstacle voxel in the octree
- Collision checks can be sped up by limiting depth in the tree

- Depth 5: Res = 0.08m

Depth 2: Res = 0.64m

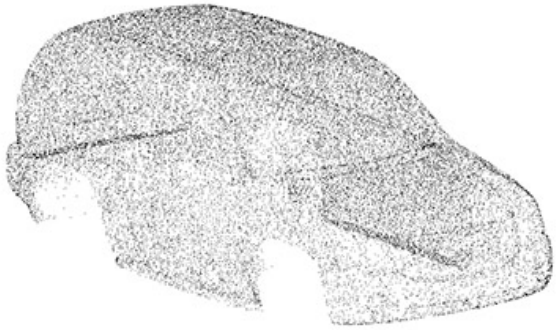


Depth 1: Res = 1.28m

Point clouds

- Clouds are all the rage
- Objects are approximated using thousands

  of discrete points
- Data is noisy (probabilistic)
- Easily encode info in an Octree or AABB-tree
  - Usually RGB+D

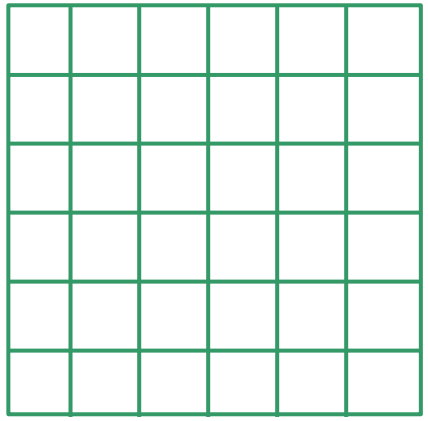


ROS pcl

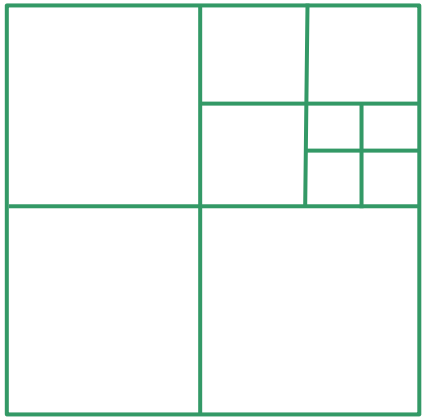
Pan, J., Sucan, I., Chitta, S., Manocha, D. *Real-time Collision Detection and Distance Computation on Point Cloud Sensor Data*. IEEE ICRA 2013.

Spatial Data Structures

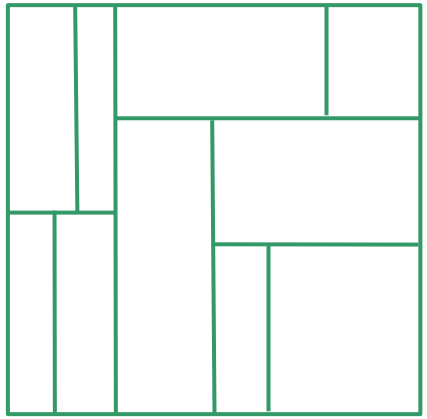
- Each cell contains information whether it is occupied by an object -> Possible collision!
- Information is updated for each object transformation
- quadtree, kd-tree, and BSP-tree are hybrid approaches
- They subdivide space, but are object-dependent



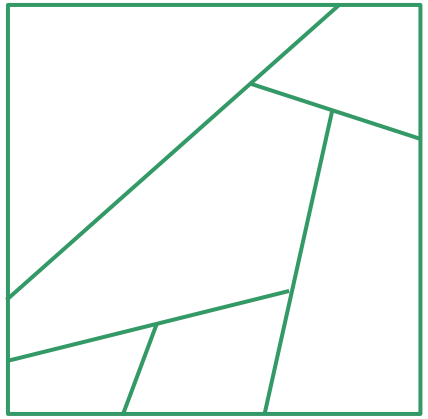
Uniform grid



Quadtree / Octree



k-d tree



BSP tree

Overview

- Motivation
- Bounding Volumes
- Hierarchies of Bounding Volumes
- Spatial Partitioning