Lecture 13

Planning - V - Collision Detection

Fishman, Adam, Adithyavairavan Murali, Clemens Eppner, Bryan Peele, Byron Boots, and Dieter Fox. "Motion policy networks." In *Conference on Robot Learning* 2023.

Course Logistics

- **Quiz 6 was posted yesterday and was due at noon today.**
- Project 4 is due tomorrow (extended) instead of today.
- Project 5 will be posted today 03/05 and will be due on 03/24.

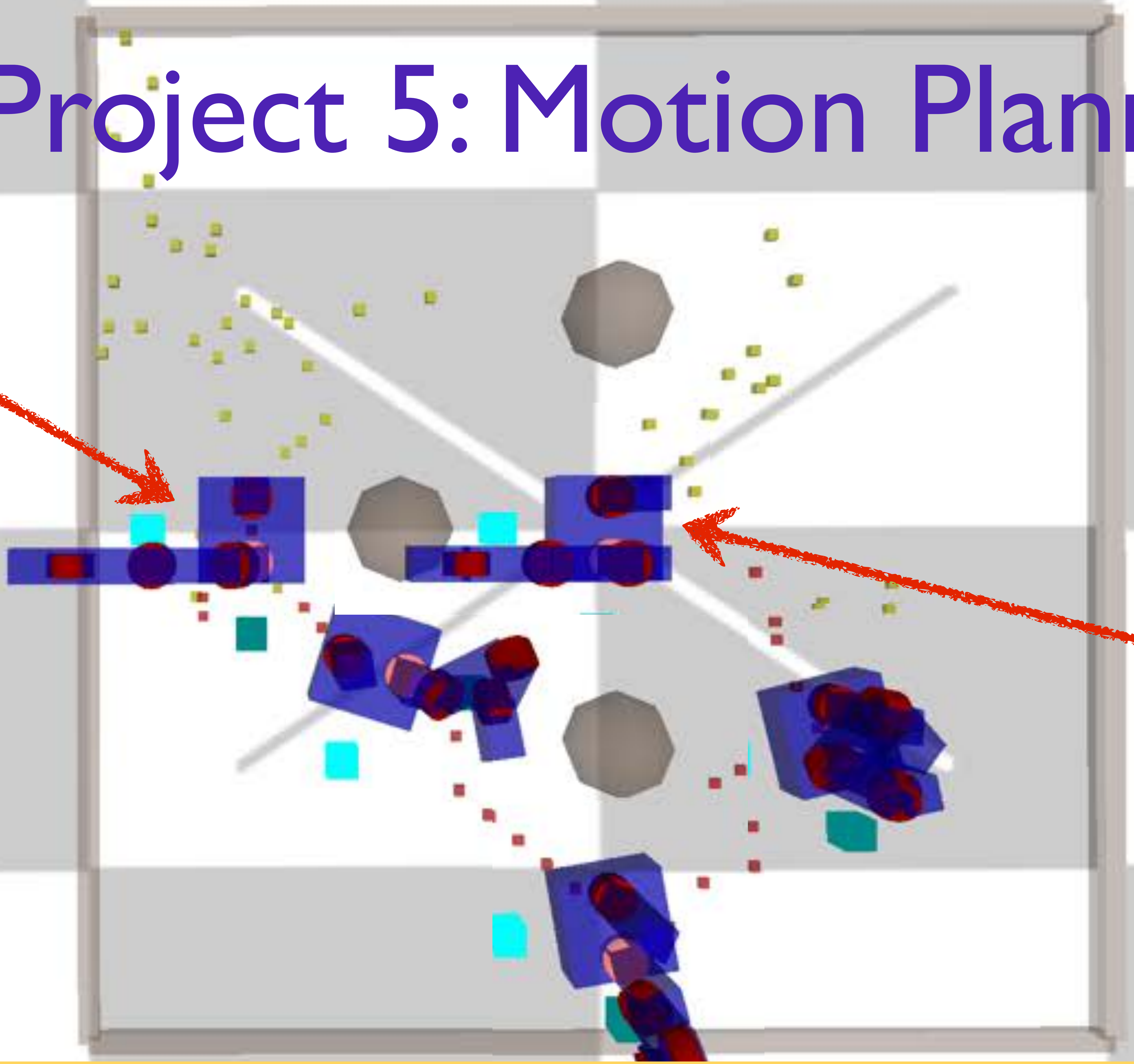
Project 5: Motion Planning

- Generate a collision free motion plan to the world origin and zero joint angle configuration

Project 5: Motion Planning

Project 5: Motion Planning

Project 5: Motion Planning



Start: random non-colliding configuration

Goal: zero configuration at world origin

Generate collision-free motion plan

home.html?world=worlds/world_barrier.js?robot=robots/robot_urdf_example.js

Stencil code for KinEval (Kinematic Ev

2 commits

Branch: master | New pull request

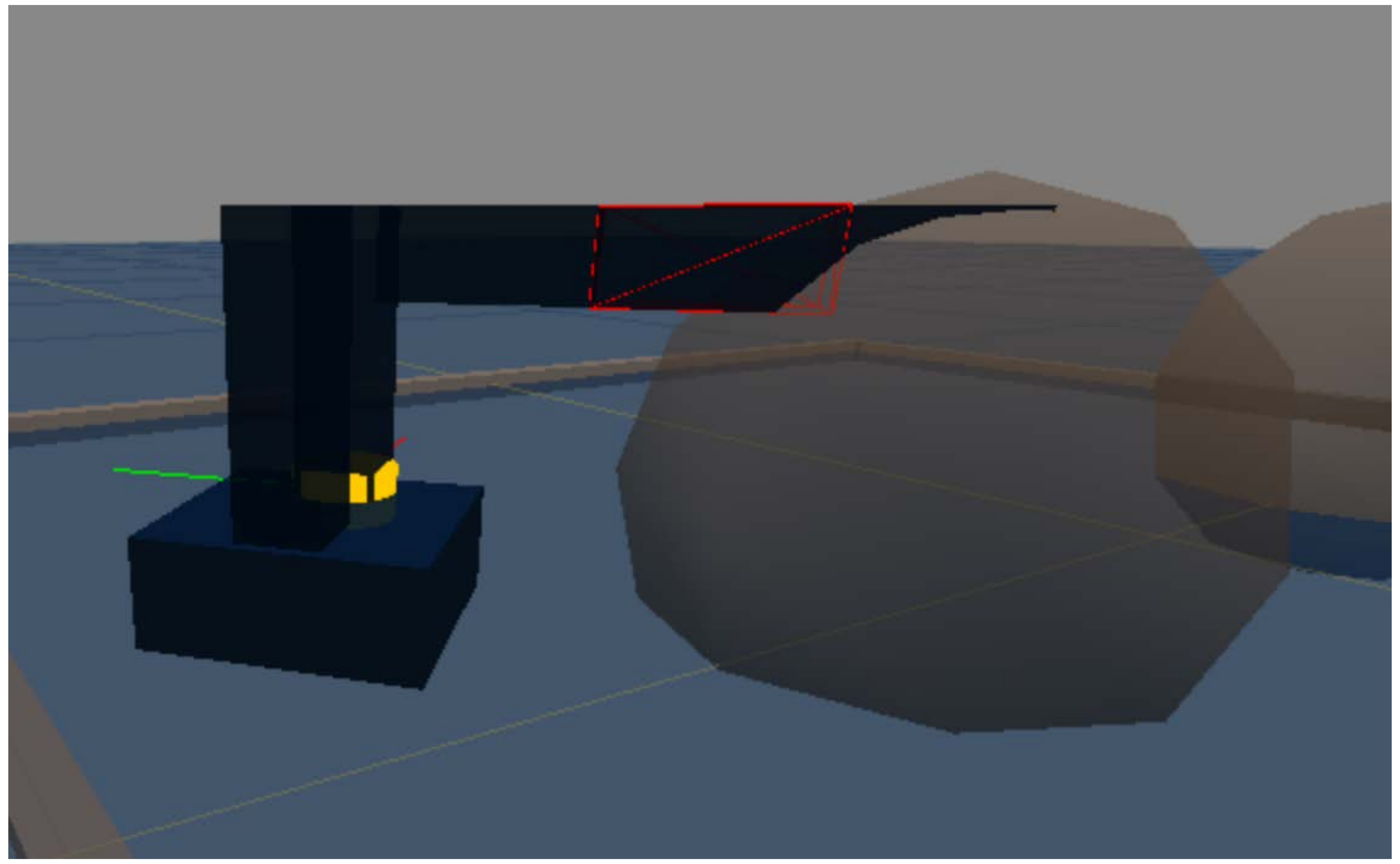
js		
kineval		
project_pathplan		
project_pendularm		
robots		
tutorial_heapsort		
tutorial_js		
worlds	initial commit	26 days ago
README.md	initial commit	26 days ago
home.html	initial commit	26 days ago

home.html

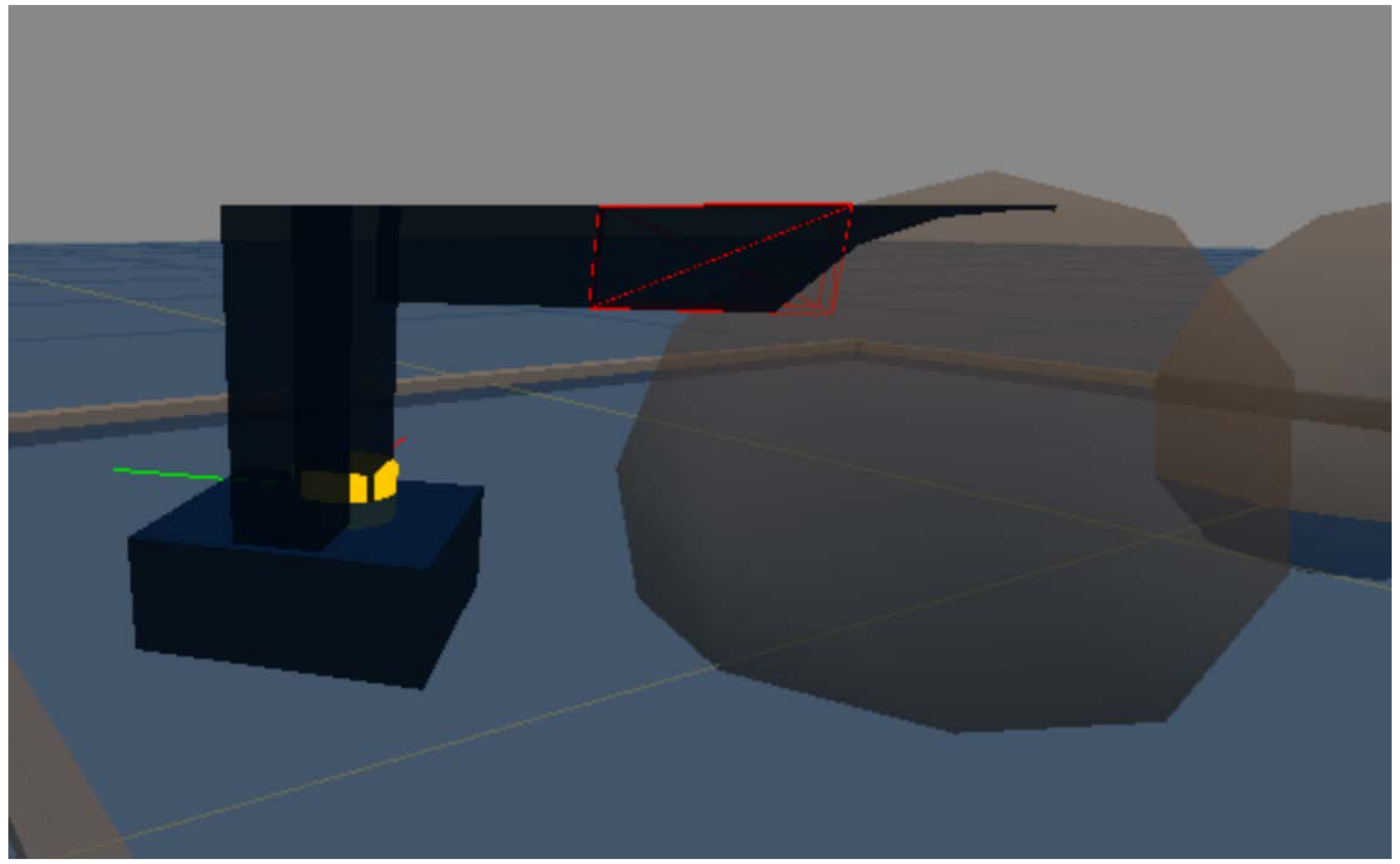
```
<script src="worlds/world_basic.js"></script>
...

function my_animate() {
    ...
    // detect robot collisions
    kineval.robotIsCollision();
    ...
    // if requested, perform configuration space motion planning to home pose
    kineval.planMotionRRTConnect();
}
```

home.html

```
<script src="worlds/world_basic.js"></script>
...

function my_animate() {
     ...
     // detect robot collisions
     kineval.robotIsCollision();
     ...
     // if requested, perform configuration space
motion planning to home pose
     kineval.planMotionRRTConnect();
}
}
```



world file can be alternatively loaded by a script tag (avoid doing this)

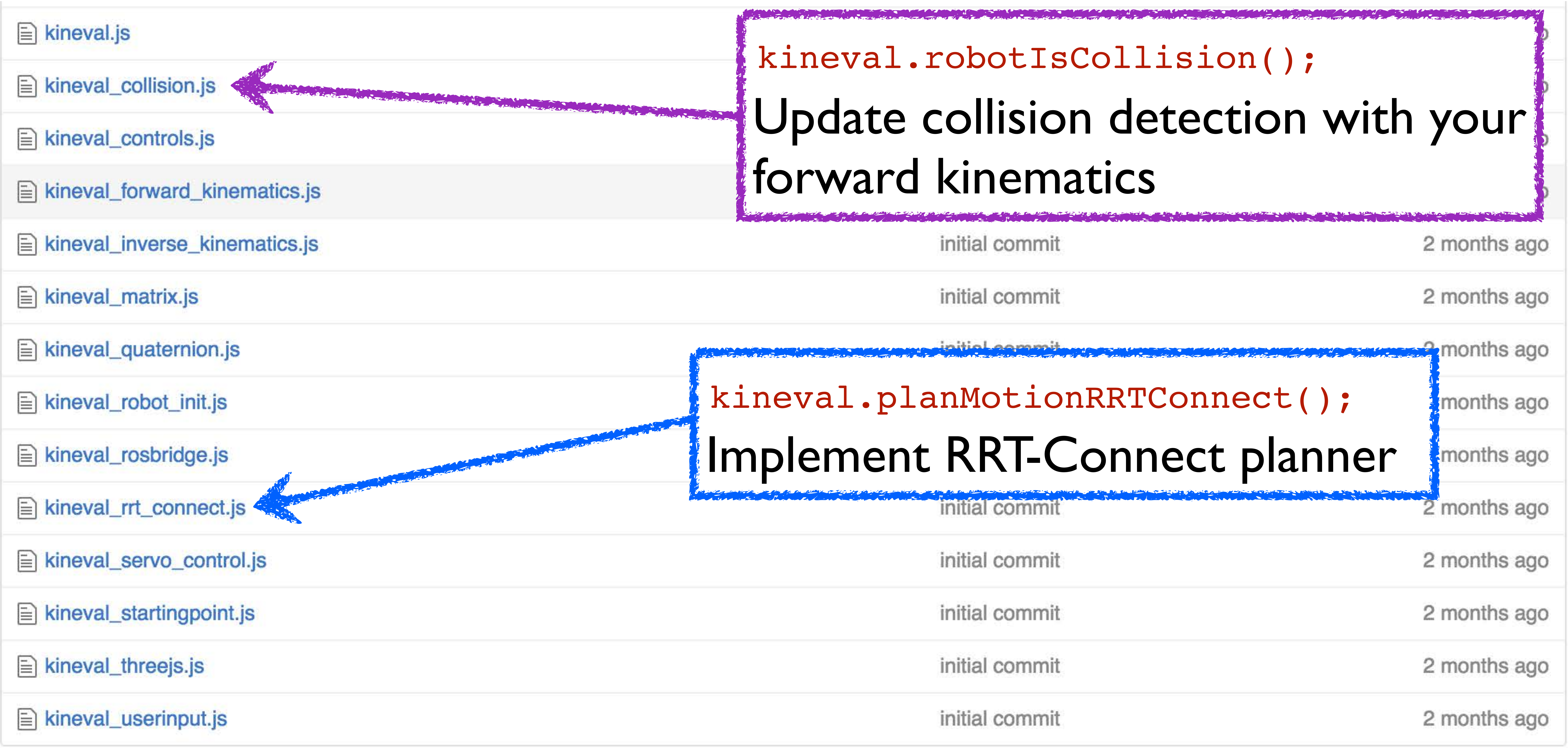


home.html

```
<script src="worlds/world_basic.js"></script>
...

function my_animate() {
    ...
    // detect robot collisions
    kineval.robotIsCollision();
    ...
    // if requested, perform configuration space
motion planning to home pose
    kineval.planMotionRRTConnect();

}
```

detect if current configuration is in collision (colliding link turns red)

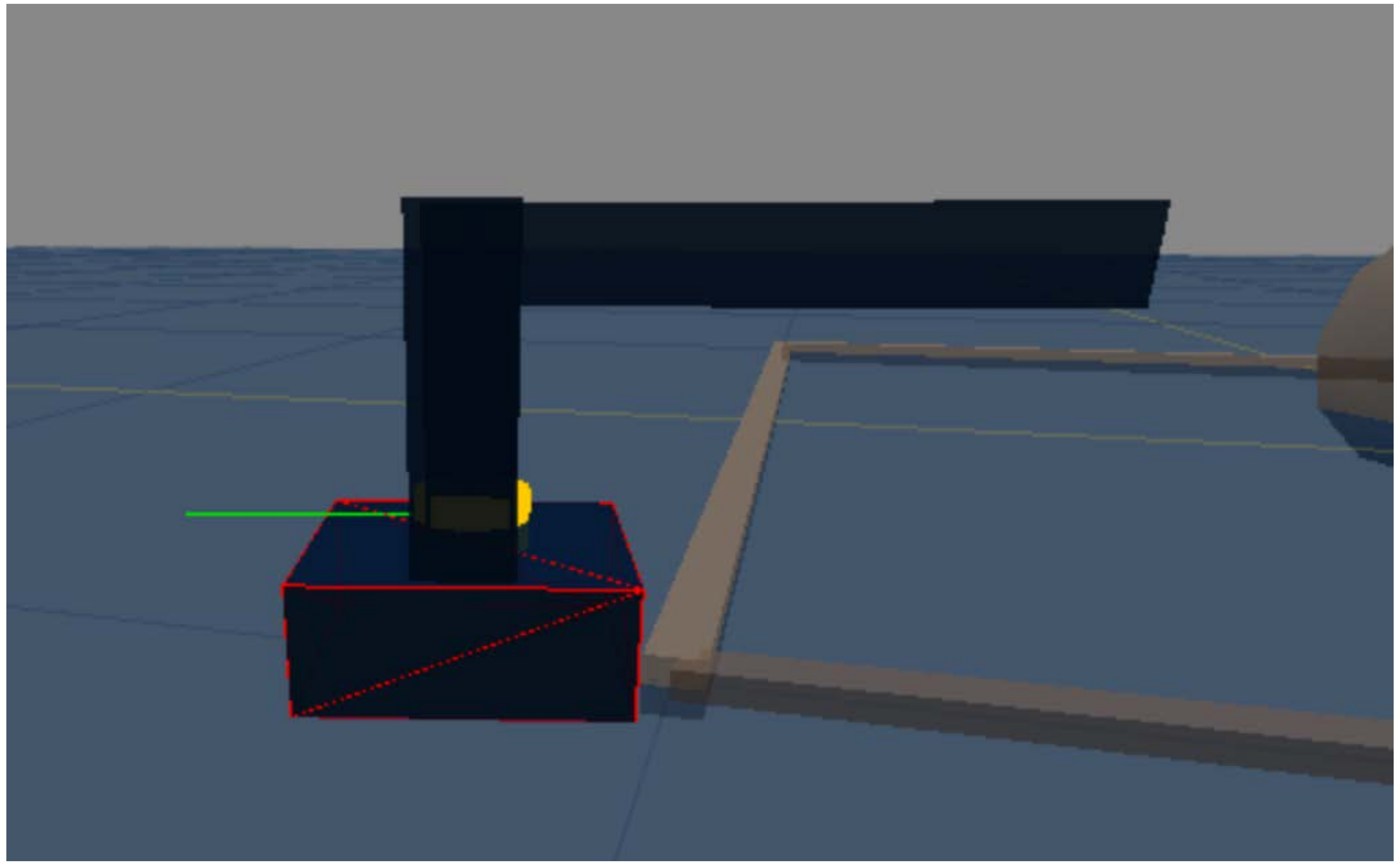
iterate motion planner


kineval.js
kineval_collision.js
kineval_controls.js
kineval_forward_kinematics.js
kineval_inverse_kinematics.js
kineval_matrix.js
kineval_quaternion.js
kineval_robot_init.js
kineval_rosbridge.js
kineval_rrt_connect.js
kineval_servo_control.js
kineval_startingpoint.js
kineval_threejs.js
kineval_userinput.js
initial commit
2 months ago
kineval.robotIsCollision();
Update collision detection with your forward kinematics
kineval.planMotionRRTConnect();
Implement RRT-Connect planner

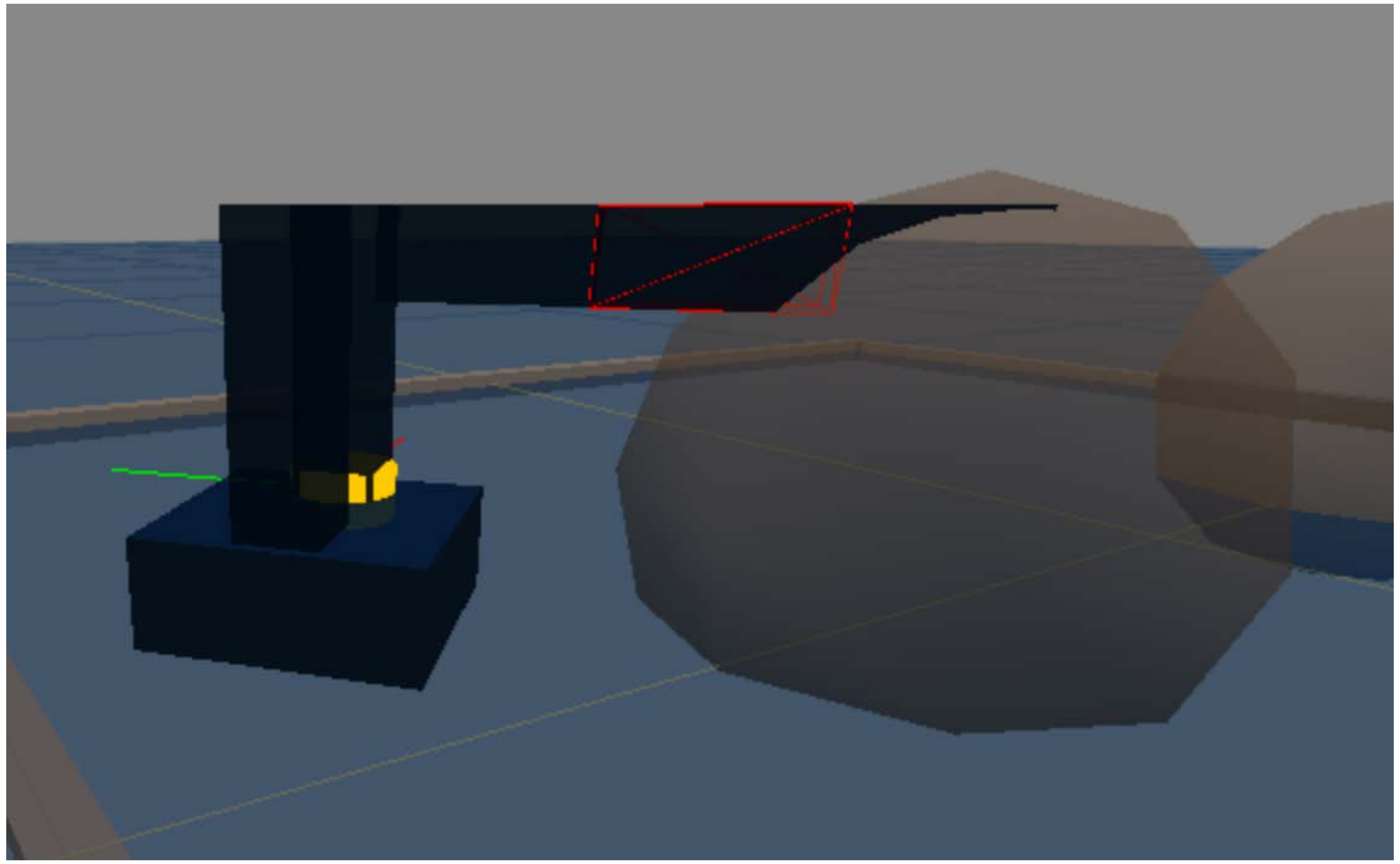
```
kineval.robotIsCollision();
```

Project 5 collision detection

Boundary Collision
(provided by default)



Link Collision
(requires your FK)



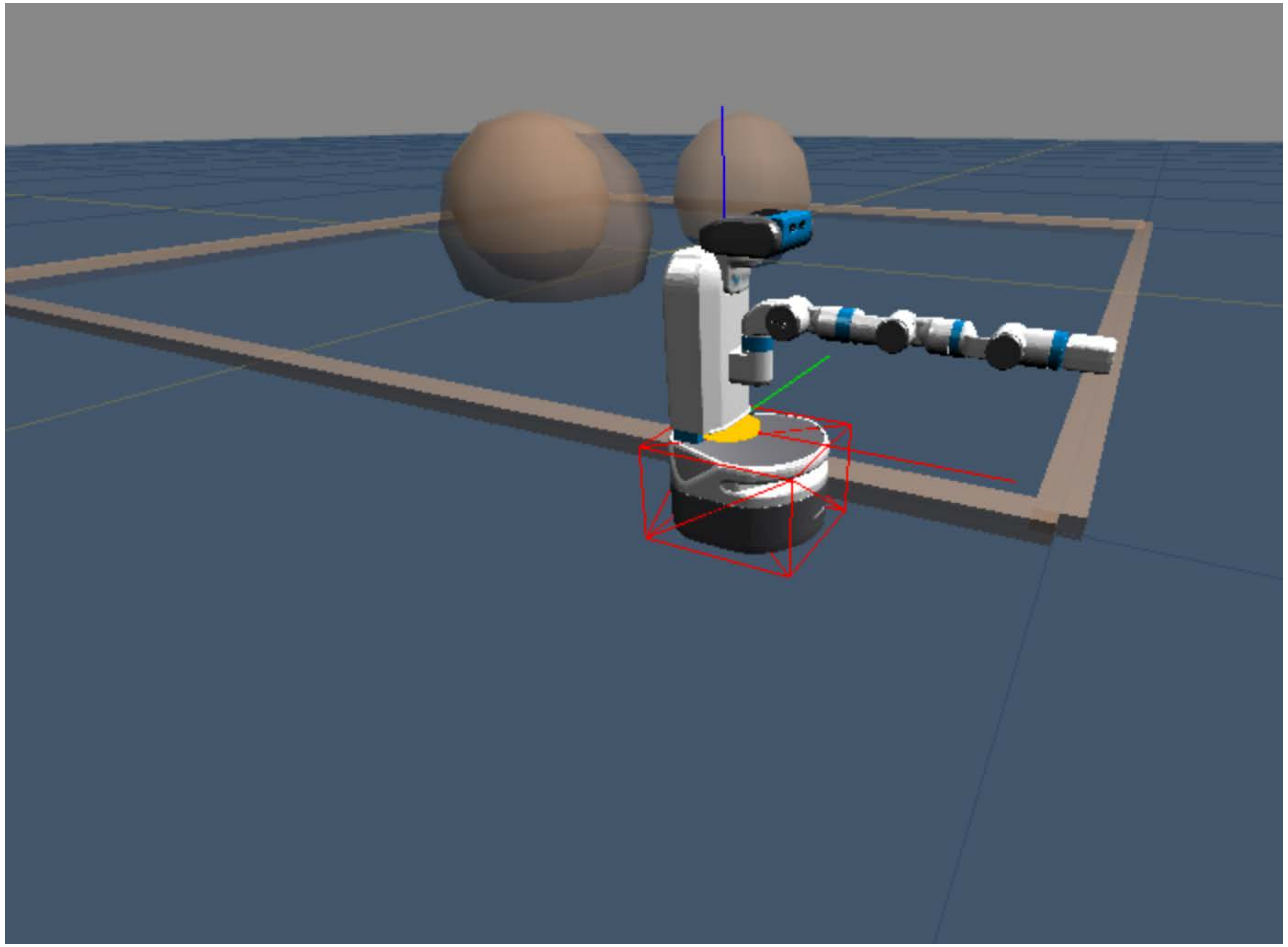
input: q (robot configuration)
output: false (for no collision) or name of link in collision

kineval_collision.js

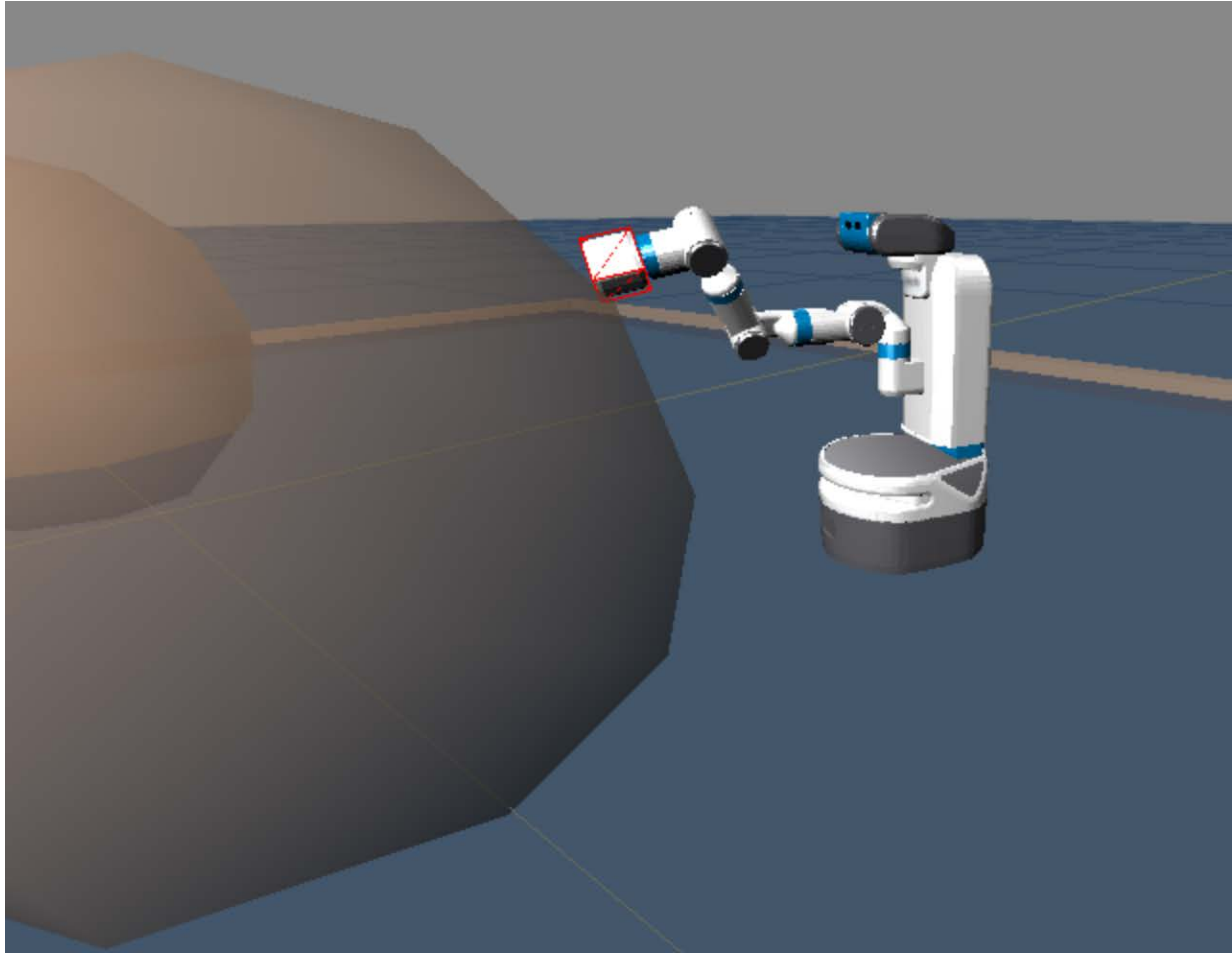
```
kineval.poseIsCollision = function robot_collision_test(q) {
    // perform collision test of robot geometry against planning world

    // test base origin (not extents) against world boundary extents
    if ((q[0]<robot_boundary[0][0])||(q[0]>robot_boundary[1][0])||
        (q[2]<robot_boundary[0][2])||(q[2]>robot_boundary[1][2]))
        return robot.base;

    // traverse robot kinematics to test each body for collision
    // STENCIL: implement forward kinematics for collision detection
    return robot_collision_forward_kinematics(q);

}
```

world boundary detection is provided

Uncomment this call;
Implement this function with FK transforms to test for collisions;
Use provided link collision function to test bounding box of each link



```
// test base origin (not extents) against world boundary extents
if ((q[0]<robot_boundary[0][0])||(q[0]>robot_boundary[1][0])||
   (q[2]<robot_boundary[0][2])||(q[2]>robot_boundary[1][2]))
   return robot.base;
```

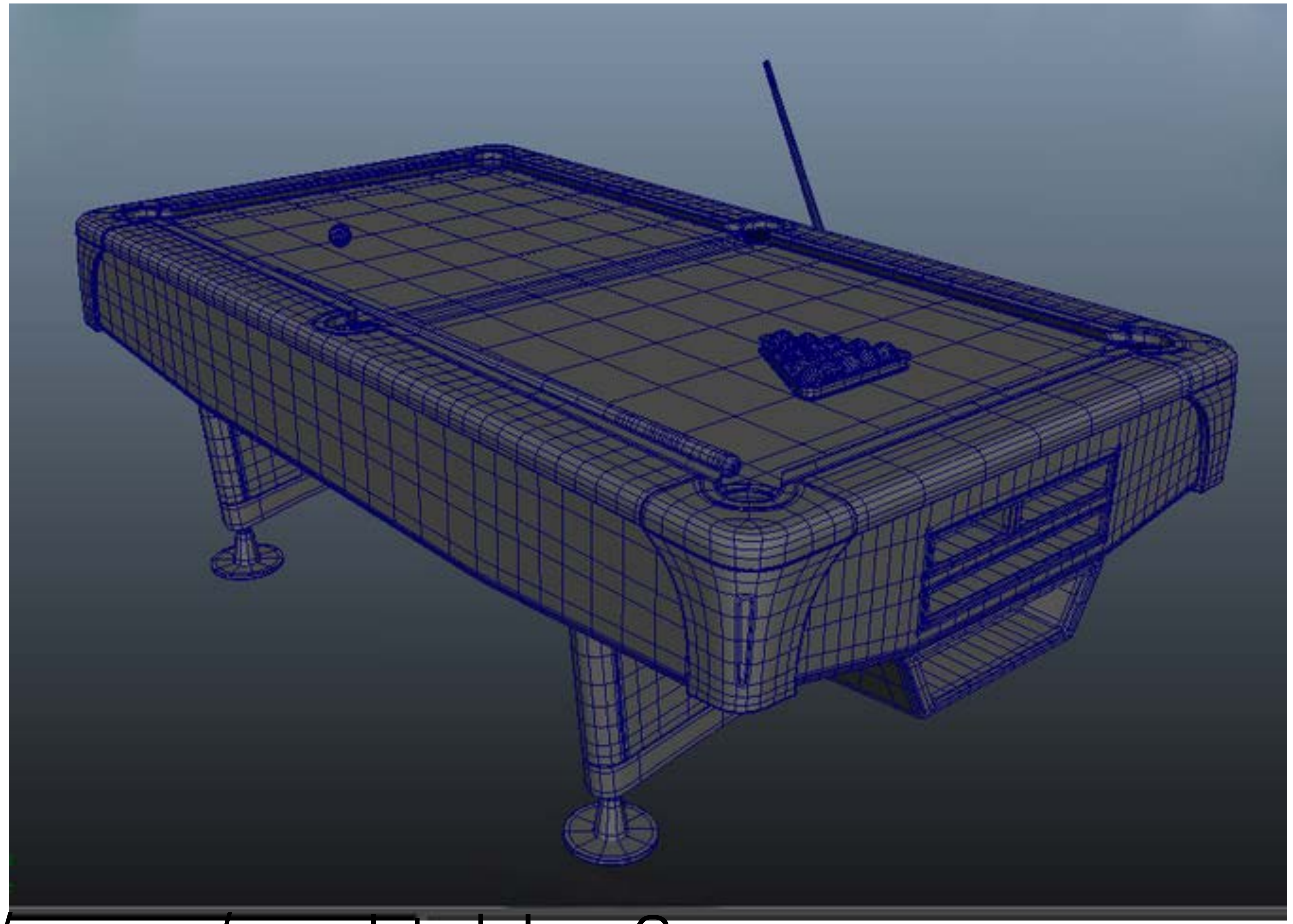


```
// traverse robot kinematics to test each body for collision
// STENCIL: implement forward kinematics for collision detection
return robot_collision_forward_kinematics(q);
```

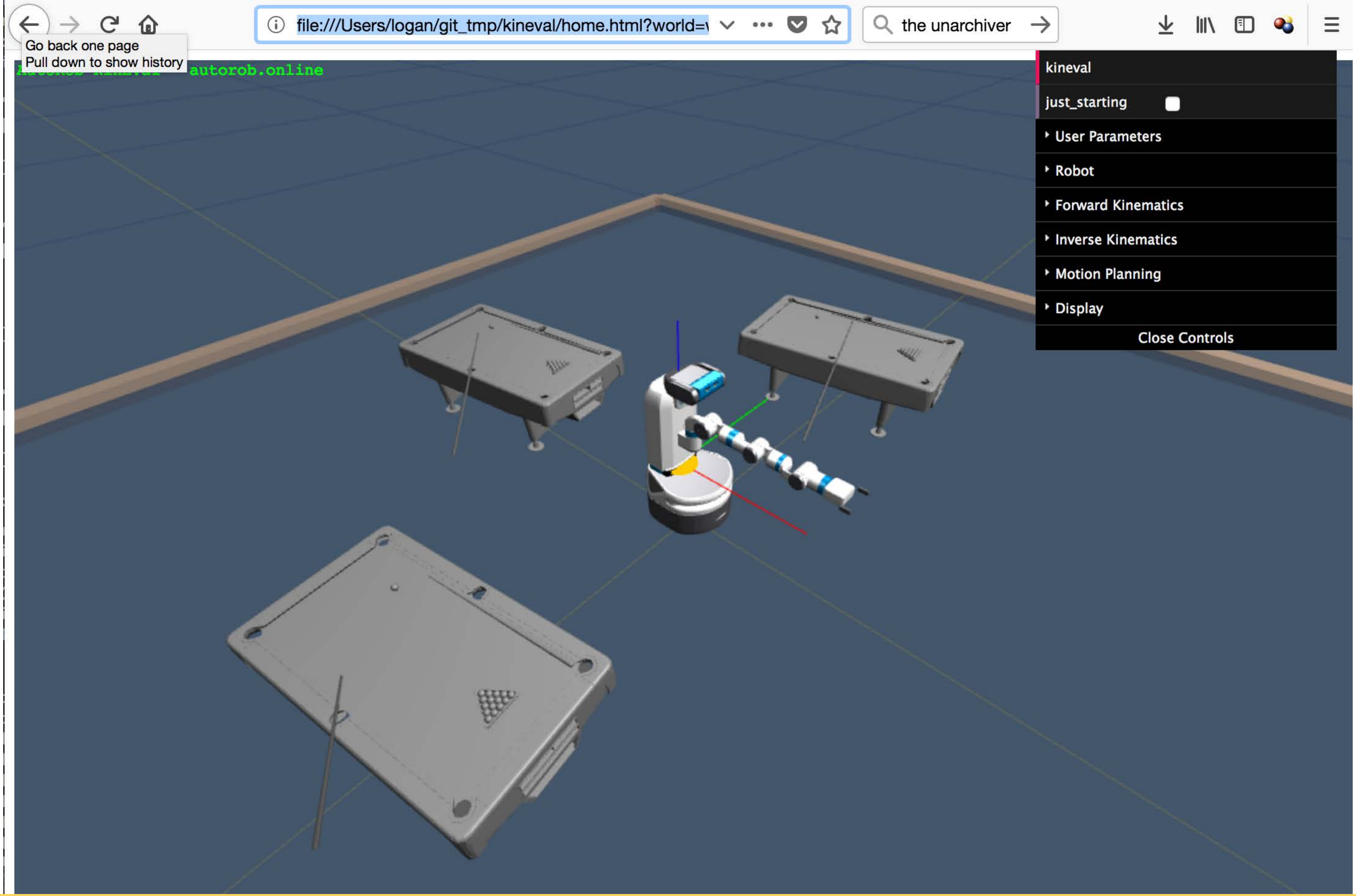

Slide borrowed from Michigan Robotics autorob.org

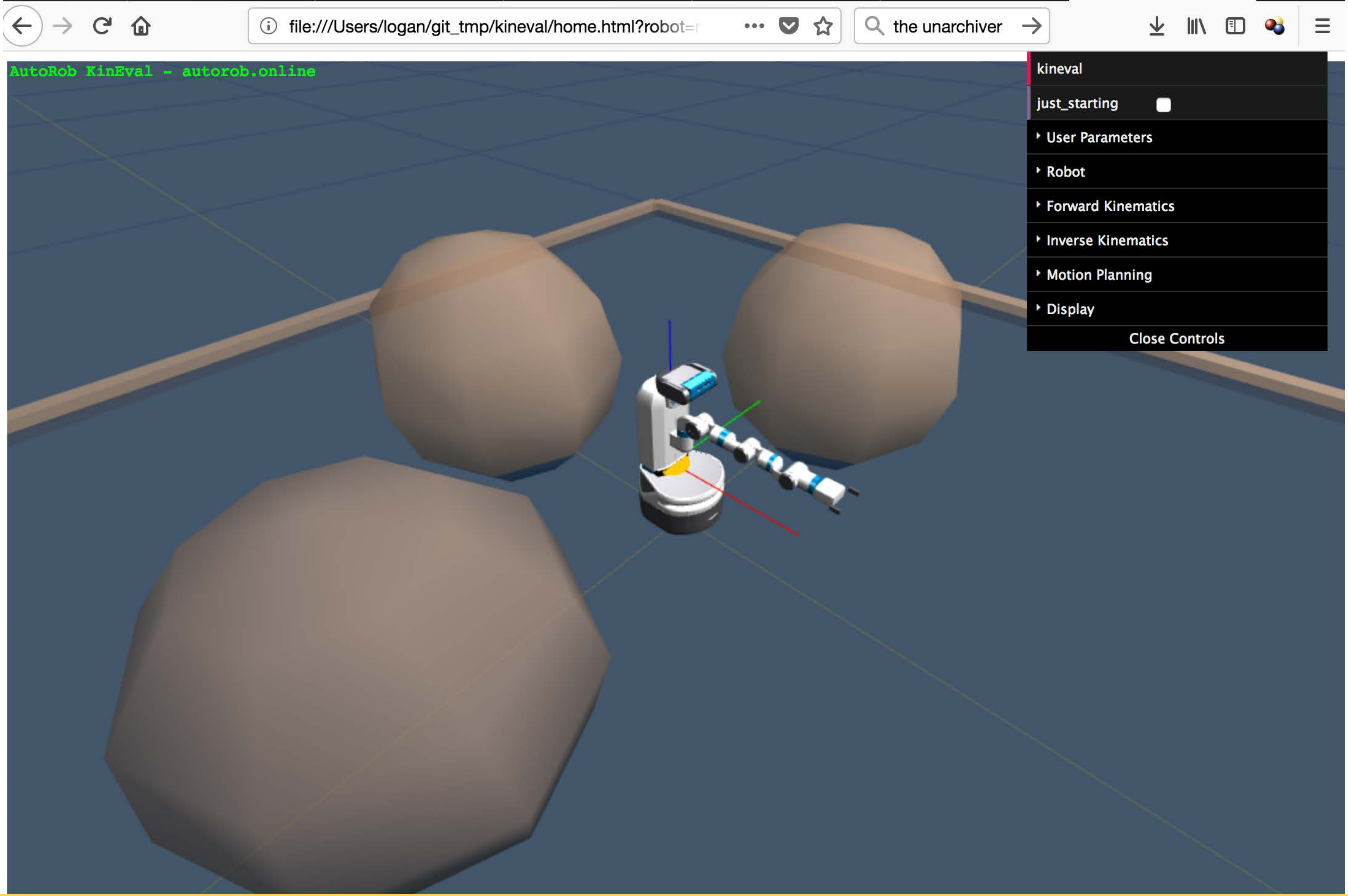
What item is not real in this picture?

Link geometries are represented as triangular geometric meshes



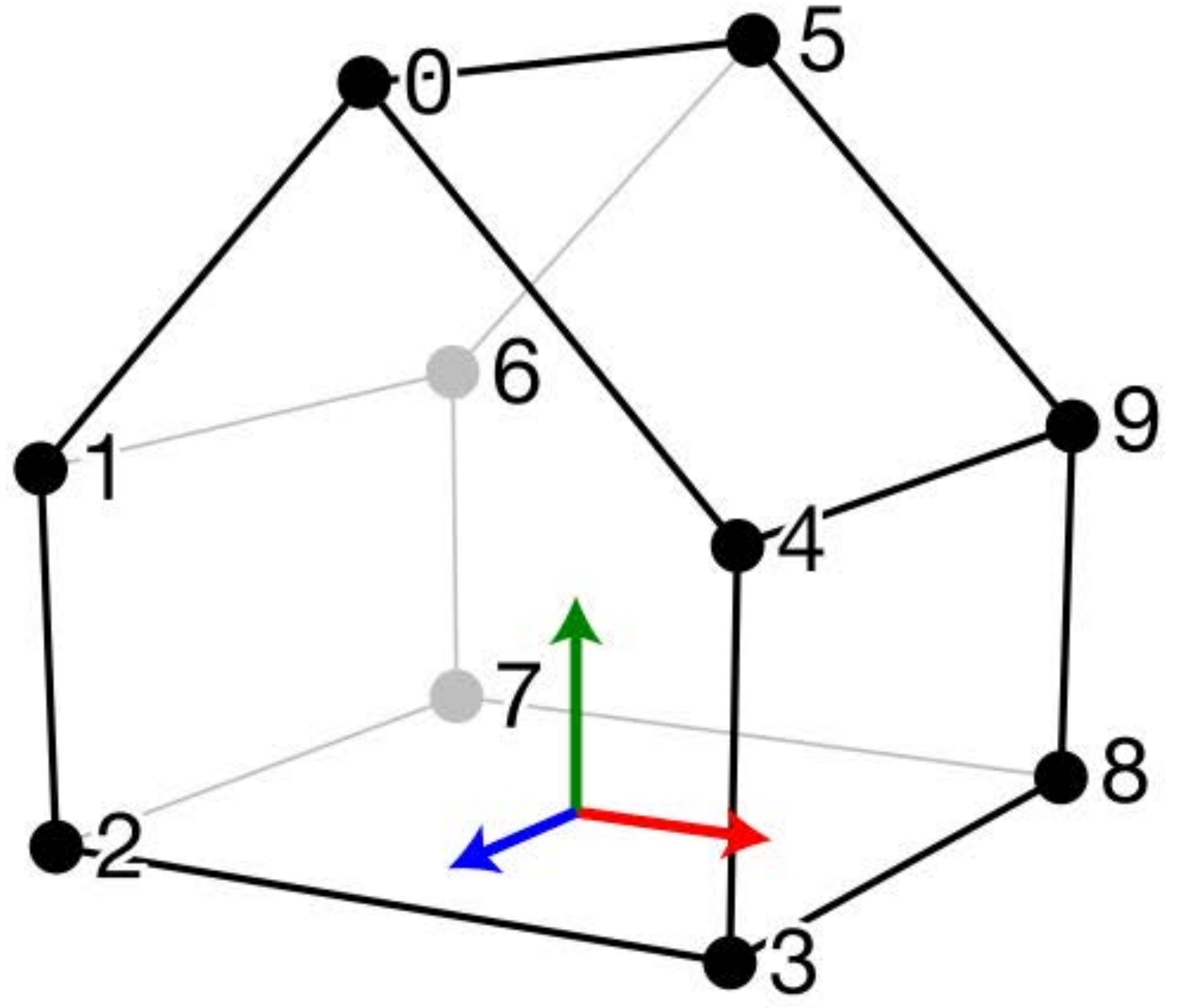
https://www.cgtrader.com/3d-models/sports/game/pool-table--3


file:///Users/logan/git_tmp/kineval/home.html?world=
the unarchiver
Go back one page
Pull down to show history
autorob.online
kineval
just_starting
User Parameters
Robot
Forward Kinematics
Inverse Kinematics
Motion Planning
Display
Close Controls


file:///Users/logan/git_tmp/kineval/home.html?robot=
the unarchiver
AutoRob KinEval - autorob.online
kineval
just_starting
User Parameters
Robot
Forward Kinematics
Inverse Kinematics
Motion Planning
Display
Close Controls

Remember:

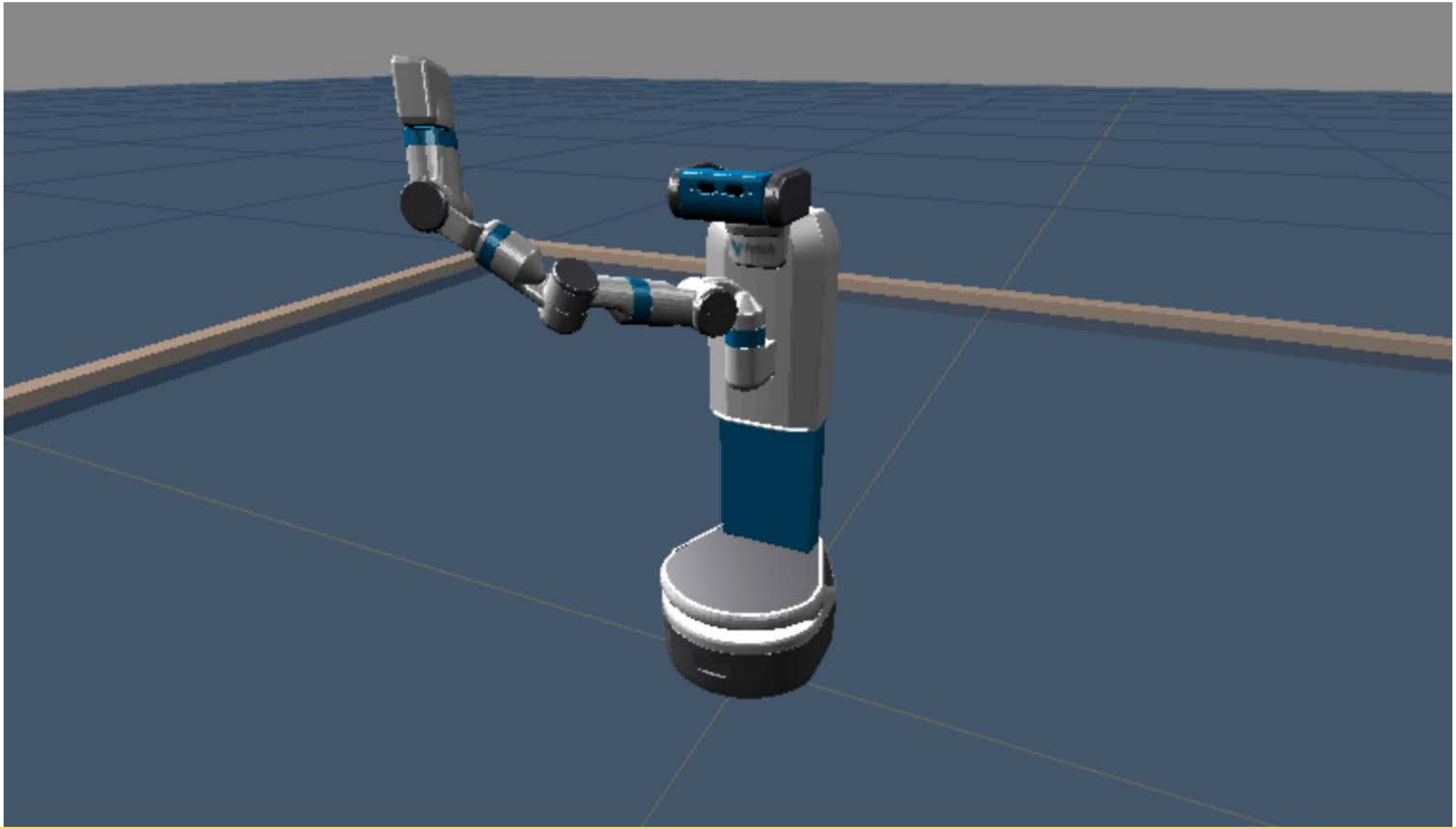
Link Geometry



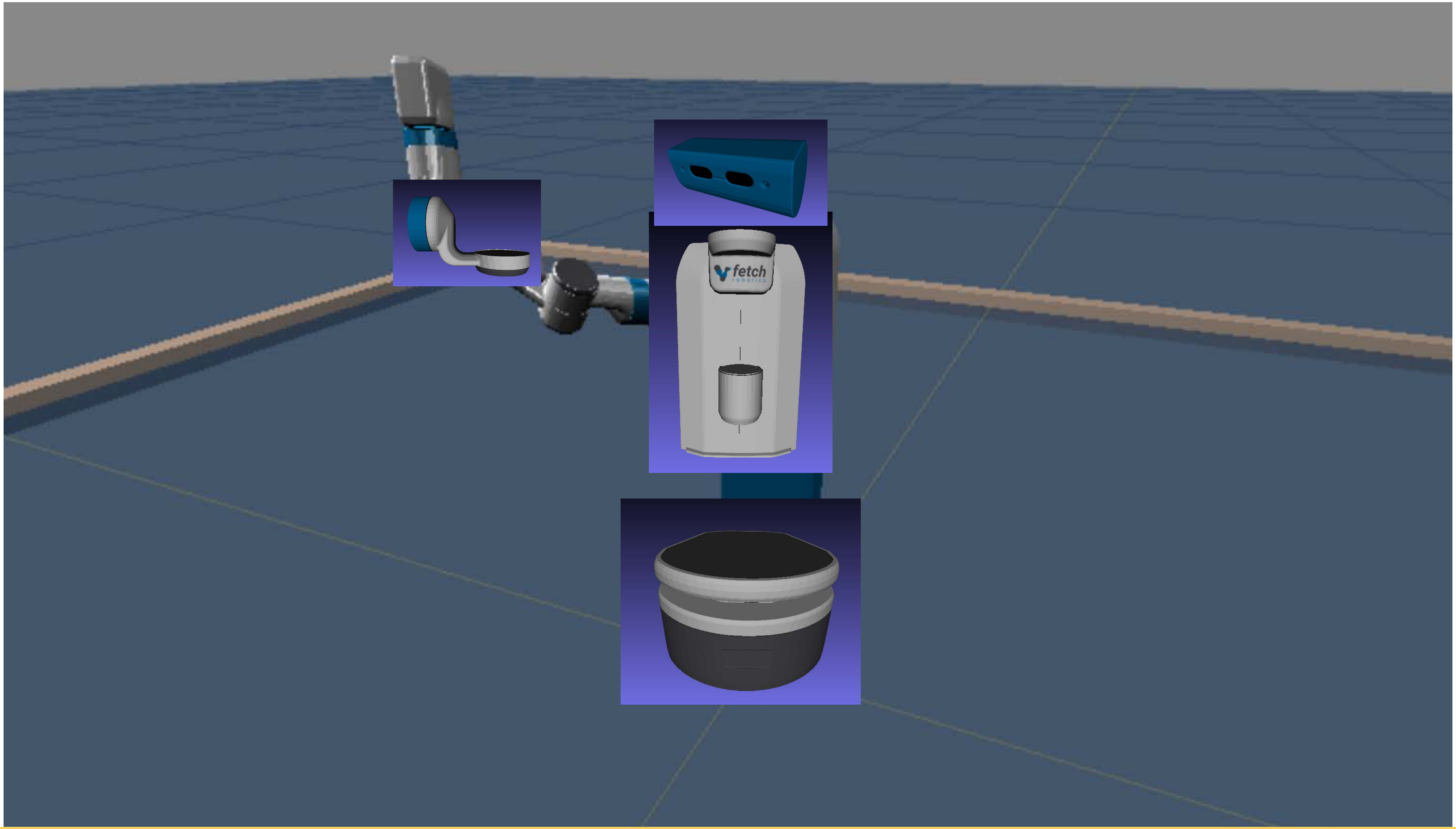
Each link has a geometry specified as 3D vertices in the frame of the link connected into faces of its surface

i	x	y	z
0	0.0	1.0	0.5
1	–0.5	0.5	0.5
2	–0.5	0.0	0.5
3	0.5	0.0	0.5
4	0.5	0.5	0.5
5	0.0	1.0	–0.5
6	–0.5	0.5	–0.5
7	–0.5	0.0	–0.5
8	0.5	0.0	–0.5
9	0.5	0.5	–0.5

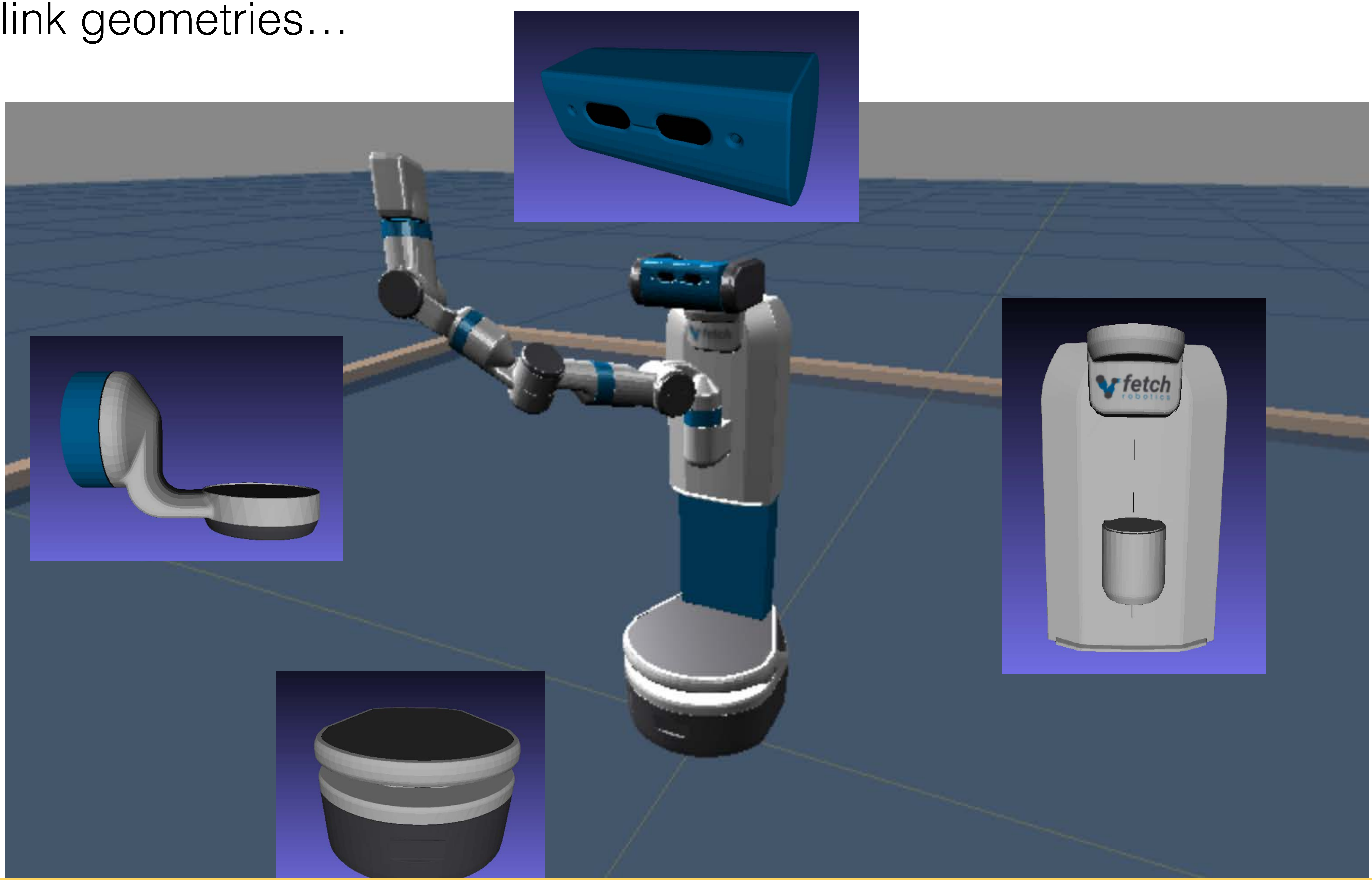
Individual link geometries…



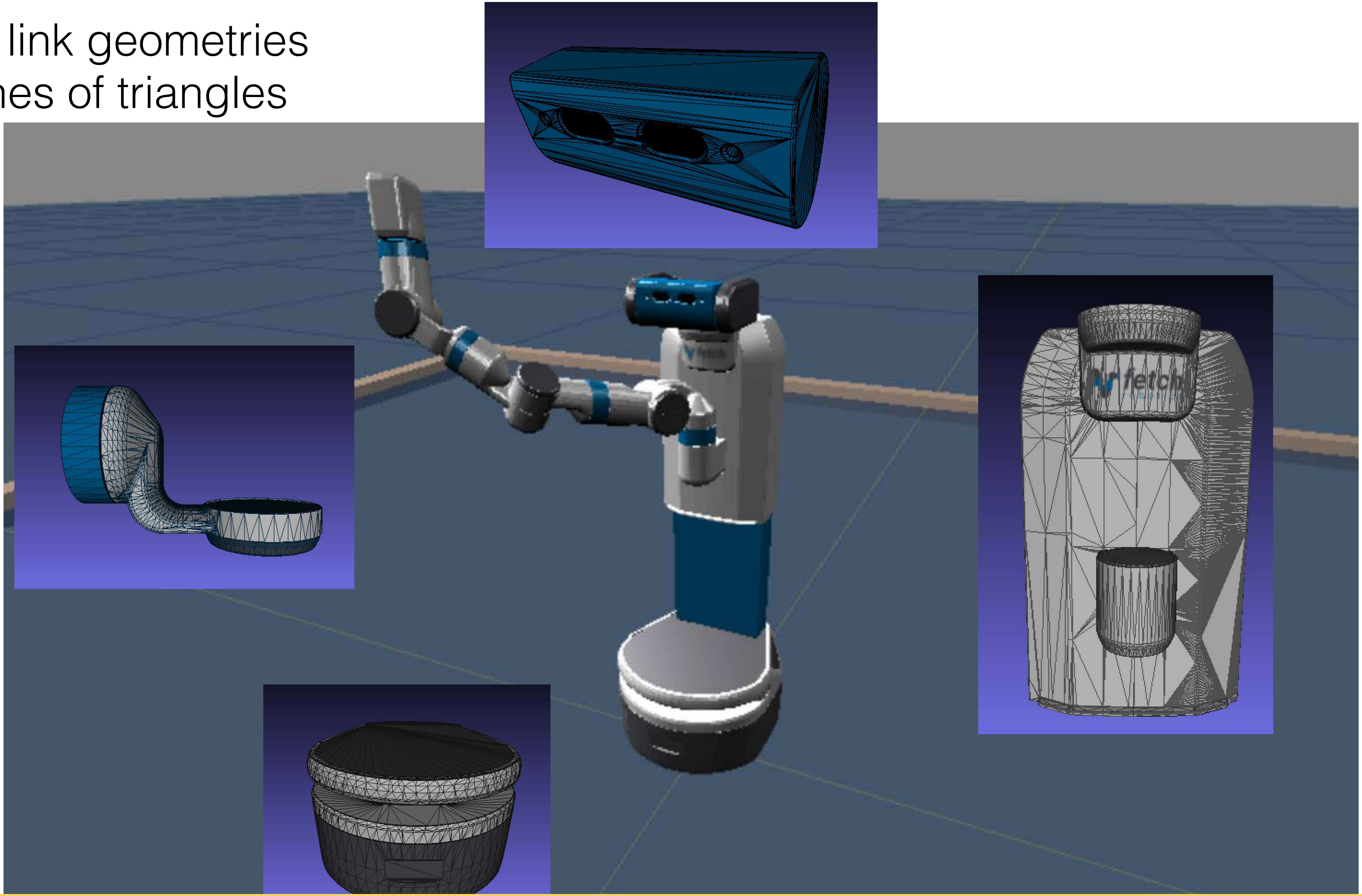
Individual link geometries…



Individual link geometries…



Individual link geometries are meshes of triangles



GitHub, Inc. (US) https://github.com/fetchrobotics/fetch_ros/tree/indigo-devel/fetch_description/meshes

marketing vs accomplishment

This repository | Search | Pull requests | Issues | Marketplace | Explore

fetchrobotics / **fetch_ros**

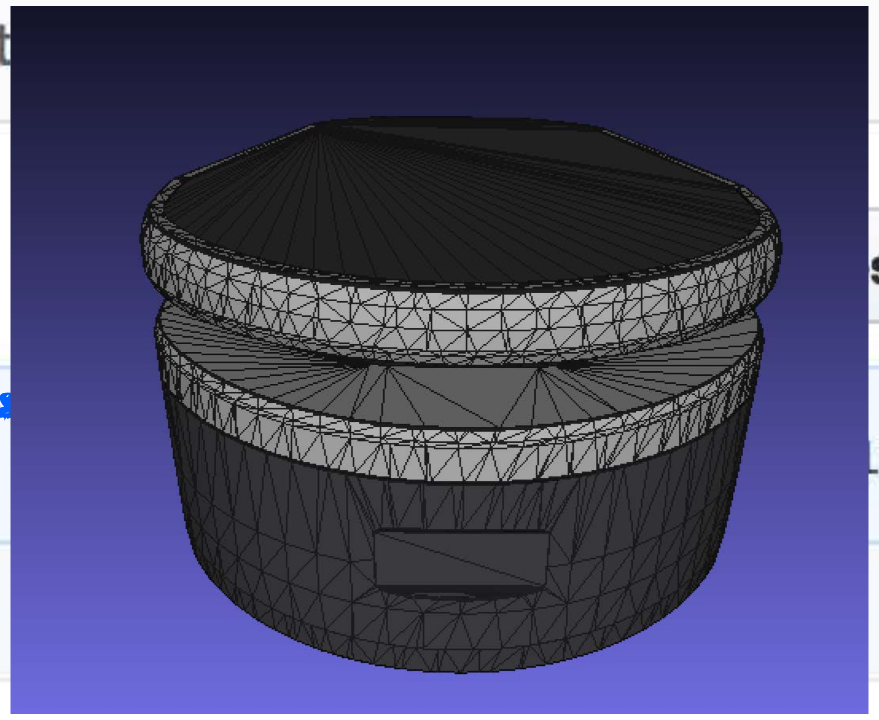
Watch 18 | Star 35 | Fork 57

Code | Issues 3 | Pull requests 1 | Projects 0 | Wiki | Insight

Branch: **indigo-devel**

fetch_ros / fetch_description / **meshes** /

Find file | History



mikeferguson update gripper model		19857 on Oct 10, 2015
..		
base_link.dae	add fetch description package	3 years ago
base_link_collision.STL	remove laser opening from collision mesh	3 years ago
base_link_uv.png	add fetch description package	3 years ago
bellows_link.STL	add fetch description package	3 years ago
bellows_link_collision.STL	add fetch description package	3 years ago
elbow_flex_link.dae	add fetch description package	3 years ago
elbow_flex_link_collision.STL	add fetch description package	3 years ago
elbow_flex_uv.png	add fetch description package	3 years ago
estop_link.STL	add fetch description package	3 years ago
forearm_roll_link.dae	add fetch description package	3 years ago

https://github.com/fetchrobotics/fetch_ros/tree/indigo-devel/fetch_description/meshes

fetchrobotics / fetch_ros

Watch 18 | Star 35 | Fork 57

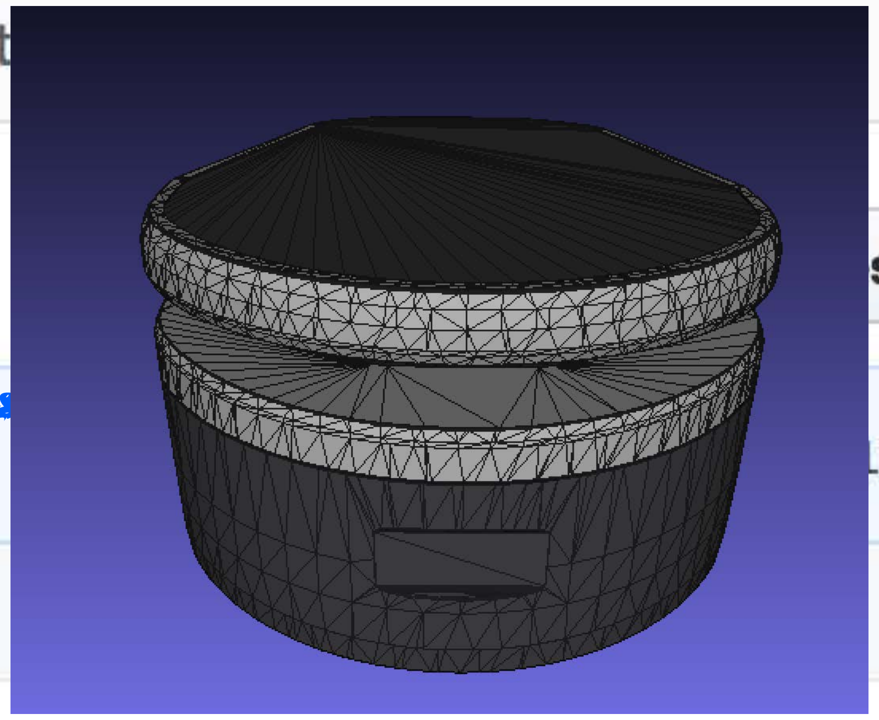
Code | Issues 3 | Pull requests 1 | Projects 0 | Wiki | Insights

Branch: indigo-devel | fetch_ros / fetch_description / meshes /

Find file | History

mikeferguson update gripper model ...19857 on Oct 10, 2015

..		
base_link.dae	add fetch description package	3 years ago
base_link_collision.STL	remove laser opening from collision mesh	3 years ago
base_li...		
bellows...		
bellows...		
elbow_...		
elbow_...		
elbow_...		
estop_l...		
forearm...		



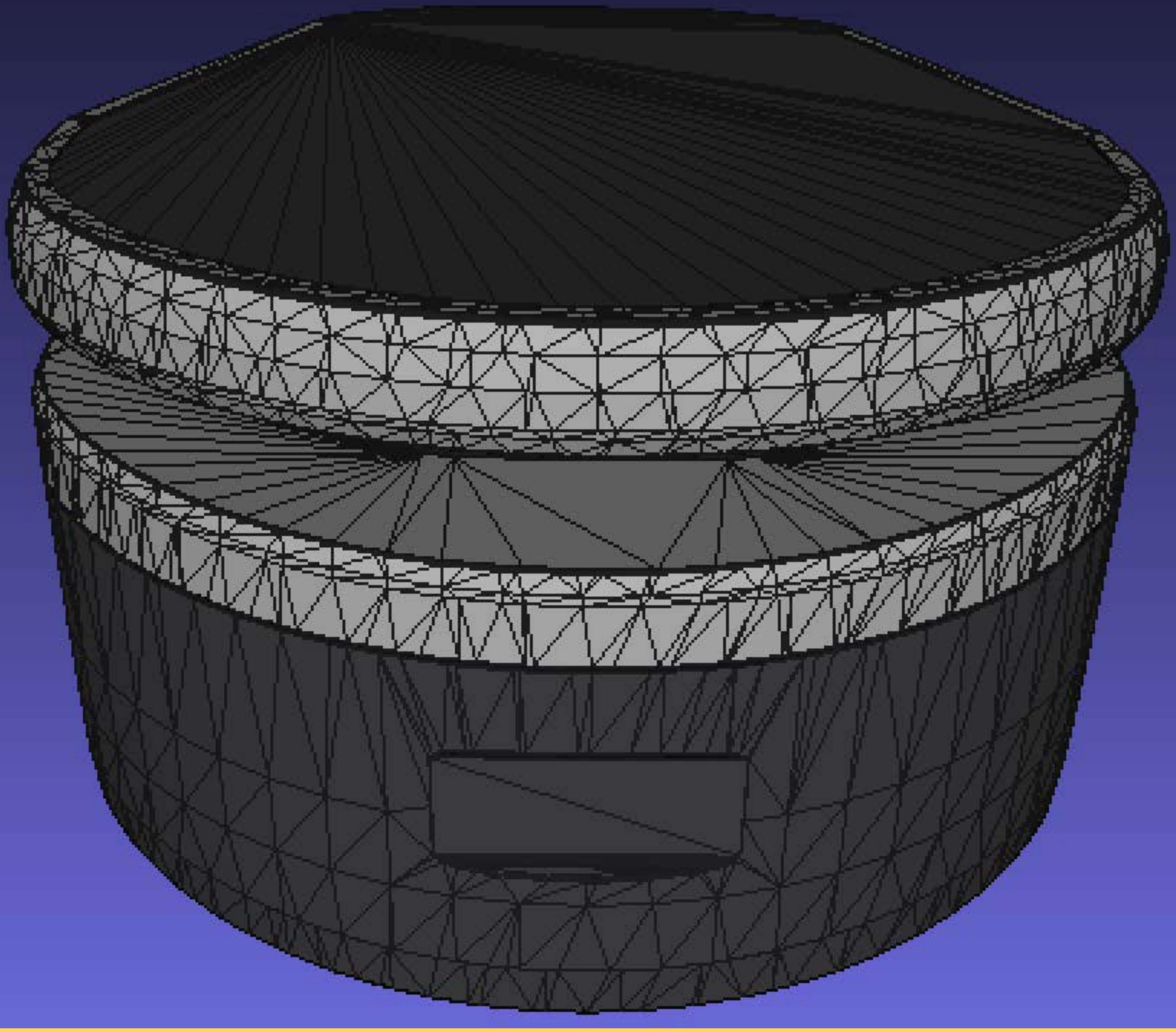
COLLADA

COLLADA (***COLLA****borative* ***D****esign* ***A****ctivity*) is an interchange file format for interactive 3D applications. It is managed by the nonprofit technology consortium, the Khronos Group, and has been adopted by ISO as a publicly available specification, ISO/PAS 17506.[1]

COLLADA defines an open standard XML schema for exchanging digital assets among various graphics software applications that might otherwise store their assets in incompatible file formats. COLLADA documents that describe digital assets are XML files, usually identified with a **.dae** (digital asset exchange) filename extension.

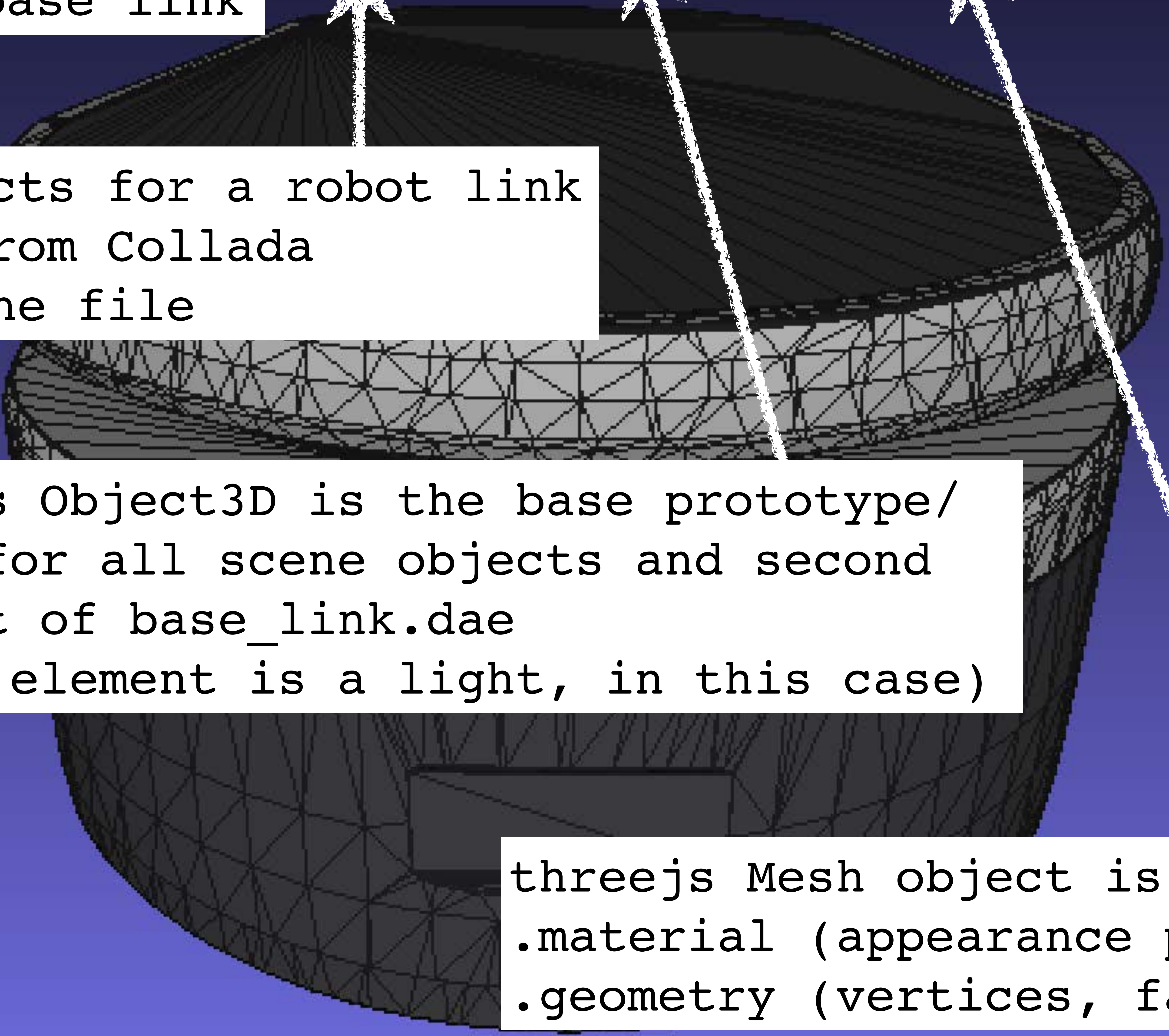
Vertices: robot.links[robot.base].geom.children[1].children[0].geometry.vertices

Faces: robot.links[robot.base].geom.children[1].children[0].geometry.faces



Vertices: robot.links[robot.base].geom.children[1].children[0].geometry.vertices

KinEval robot base link



.geom: threejs objects for a robot link (or joint) loaded from Collada (base_link.dae) scene file

threejs Object3D is the base prototype/class for all scene objects and second element of base_link.dae
(first element is a light, in this case)

threejs Mesh object is consists of:
.material (appearance properties)
.geometry (vertices, faces, normals)

Vertices: robot.links[robot.base].geom.geometry.vertices
Faces: robot.links[robot.base].geom.geometry.faces

Collision detection: ensure triangles of robot links do not intersect triangles of scene objects

Collision detection: ensure triangles of robot links do not intersect triangles of scene objects

Collision detection: ensure triangles of robot links do not intersect triangles of scene objects

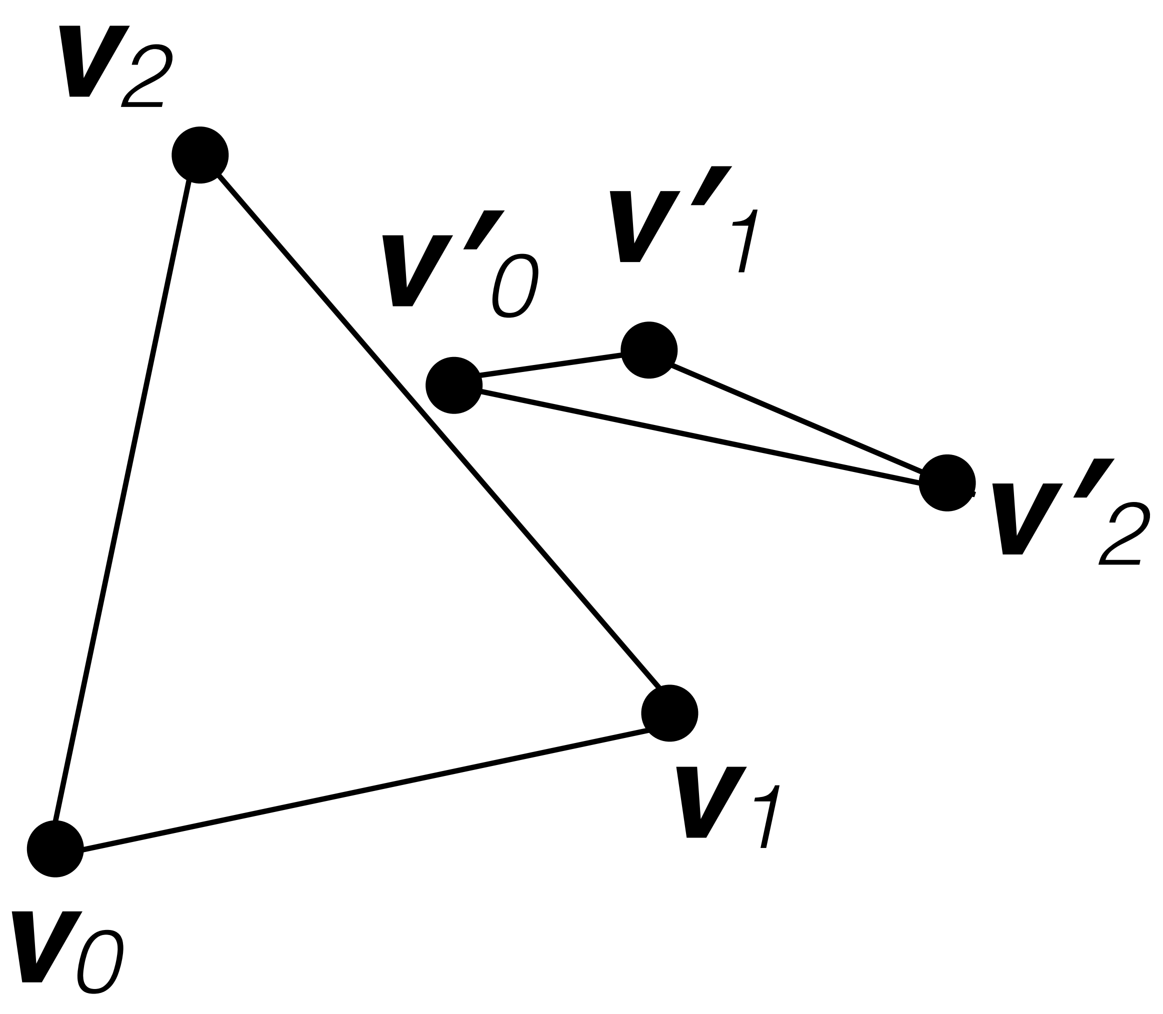
Collision detection: ensure triangles of robot links do not intersect triangles of scene objects

Slide borrowed from Michigan Robotics autorob.org

How do we test whether two triangles intersect?

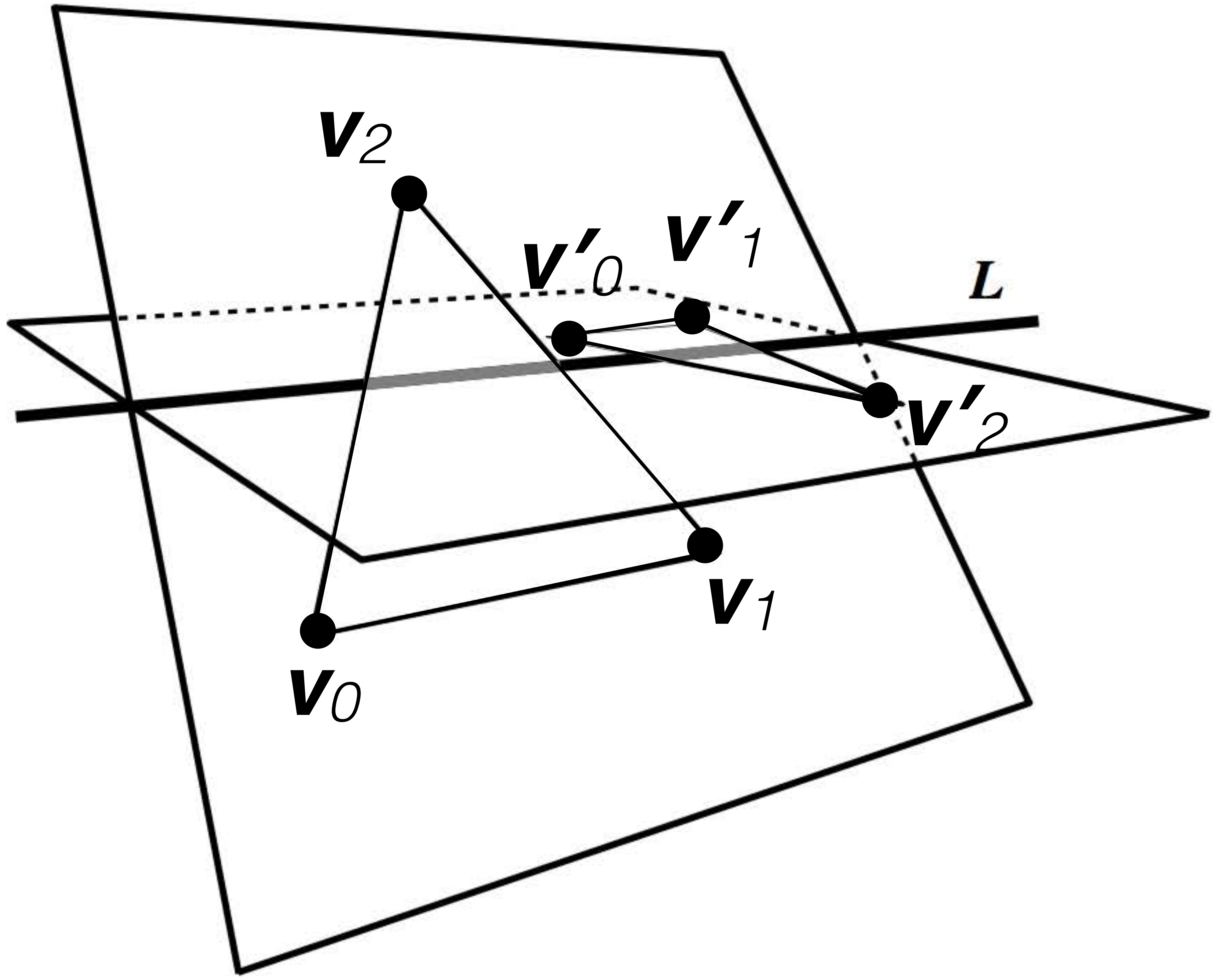
3D Triangle-Triangle Test

- Given two triangles each with three vertices
  - $T = \{\mathbf{v}_0, \mathbf{v}_1, \mathbf{v}_2\}$
  - $T' = \{\mathbf{v'}_0, \mathbf{v'}_1, \mathbf{v'}_2\}$
- Return true if T and T' intersect



3D Triangle-Triangle Test

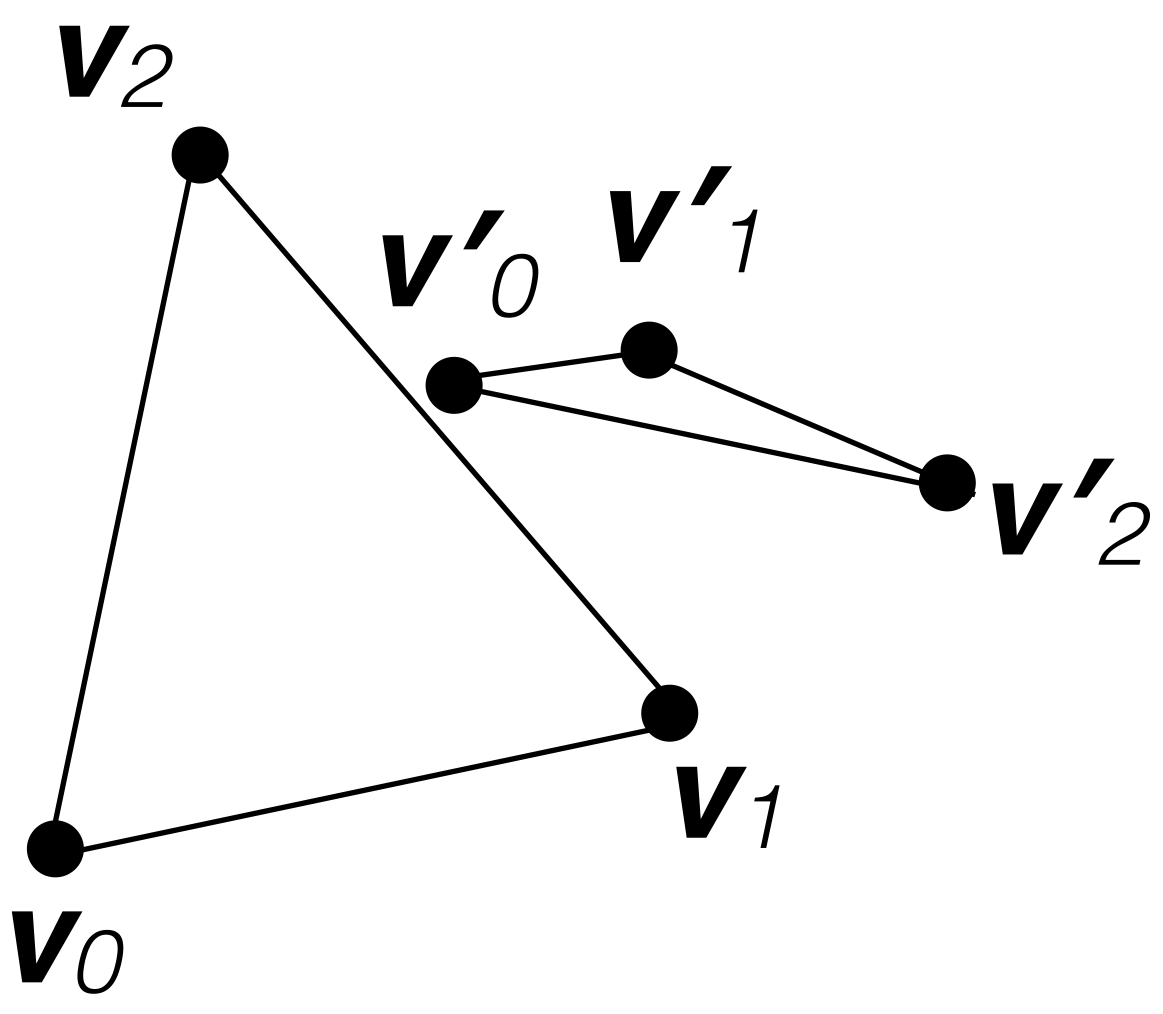
1. Compute plane equation of triangle 2.
2. Reject as trivial if all points of triangle 1 are on same side.
3. Compute plane equation of triangle 1.
4. Reject as trivial if all points of triangle 2 are on same side.
5. Compute intersection line and project onto largest axis.
6. Compute the intervals for each triangle.
7. Intersect the intervals.



Möller 1997

3D Triangle-Triangle Test

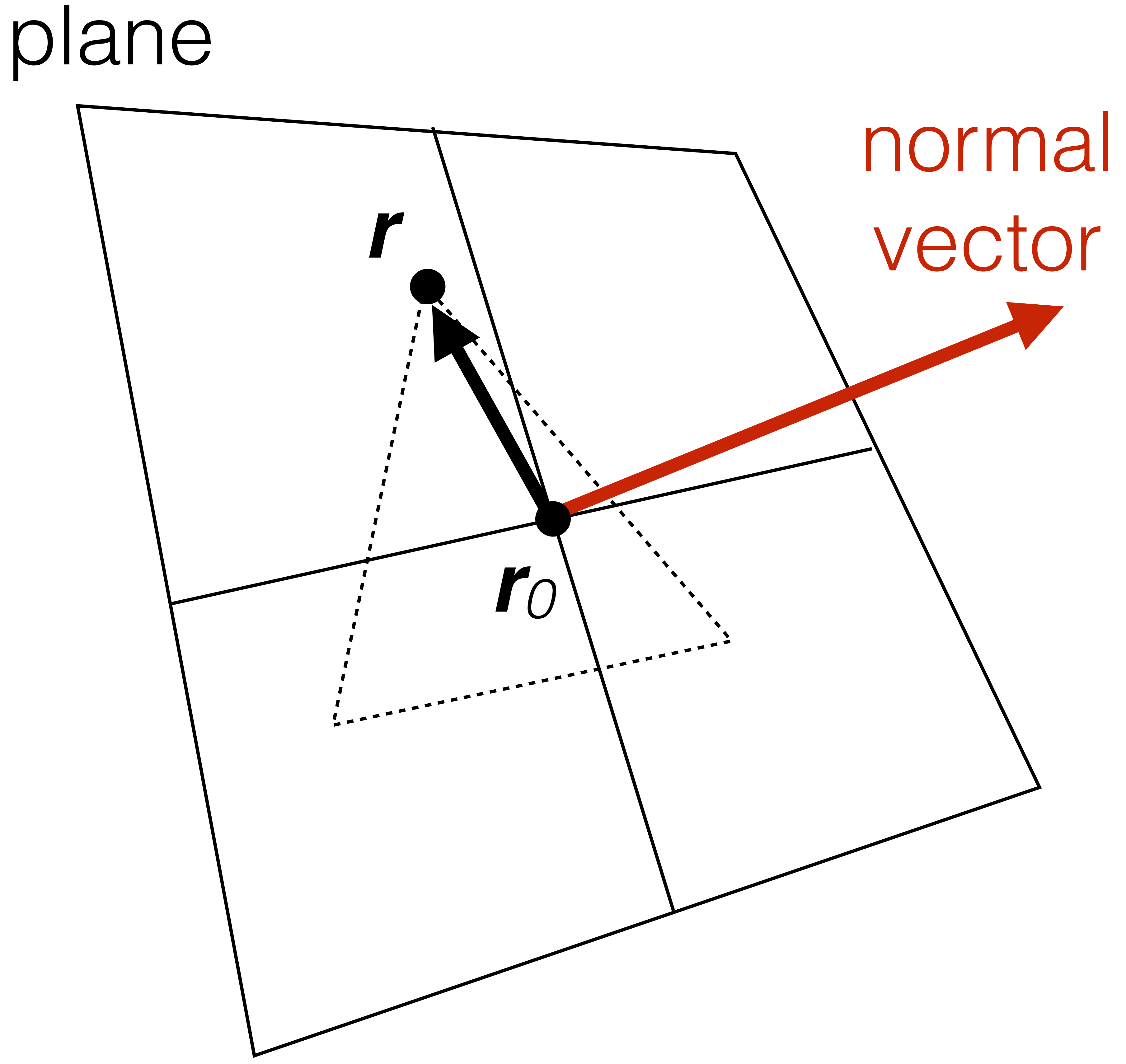
1. Compute plane equation of triangle 2.
2. Reject as trivial if all points of triangle 1 are on same side.
3. Compute plane equation of triangle 1.
4. Reject as trivial if all points of triangle 2 are on same side.
5. Compute intersection line and project onto largest axis.
6. Compute the intervals for each triangle.
7. Intersect the intervals.



Möller 1997

3D Triangle-Triangle Test

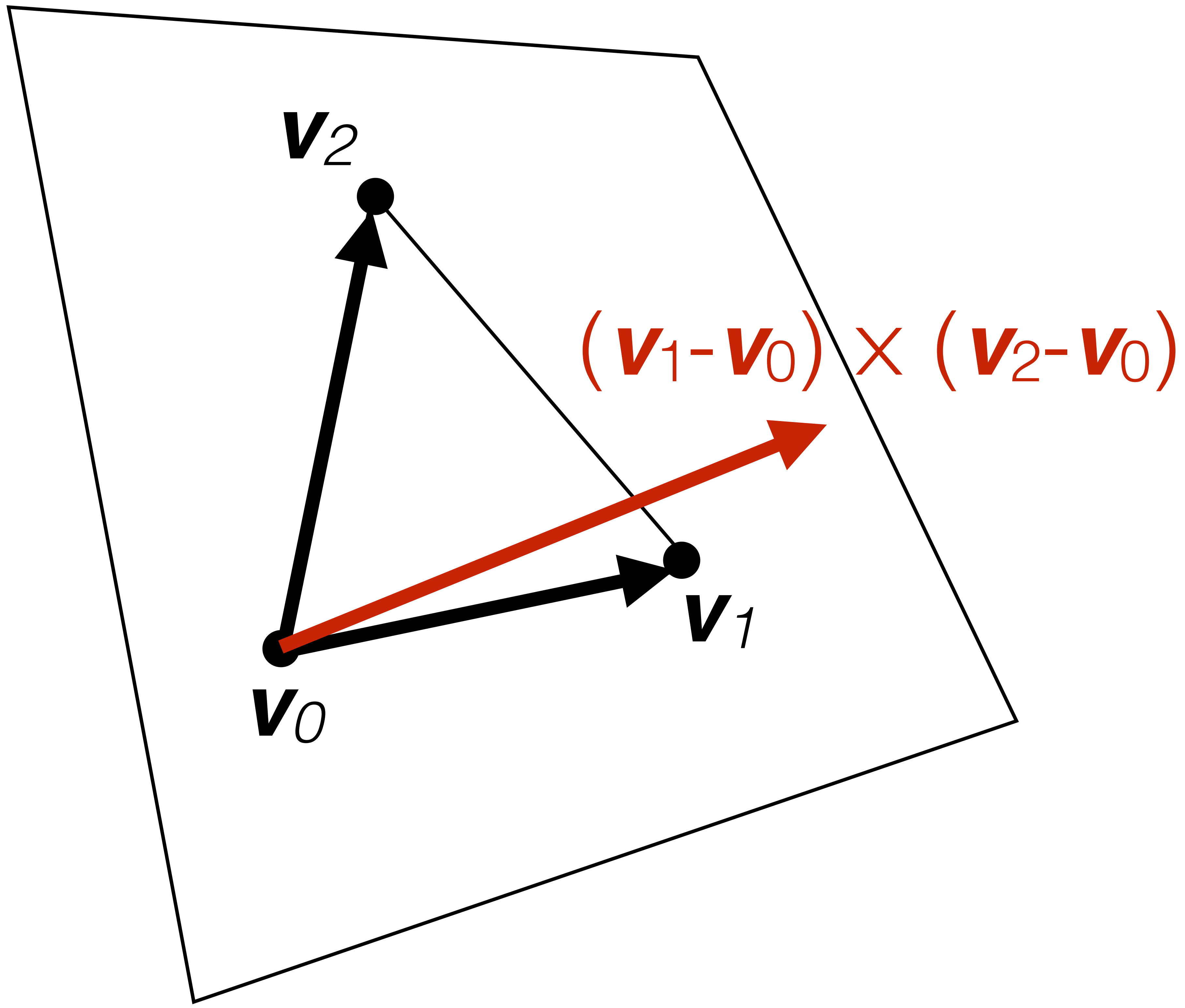
1. Compute plane equation of triangle 2.
2. Reject as trivial if all points of triangle 1 are on same side.
3. Compute plane equation of triangle 1.
4. Reject as trivial if all points of triangle 2 are on same side.
5. Compute intersection line and project onto largest axis.
6. Compute the intervals for each triangle.
7. Intersect the intervals.



Möller 1997

3D Plane Definition

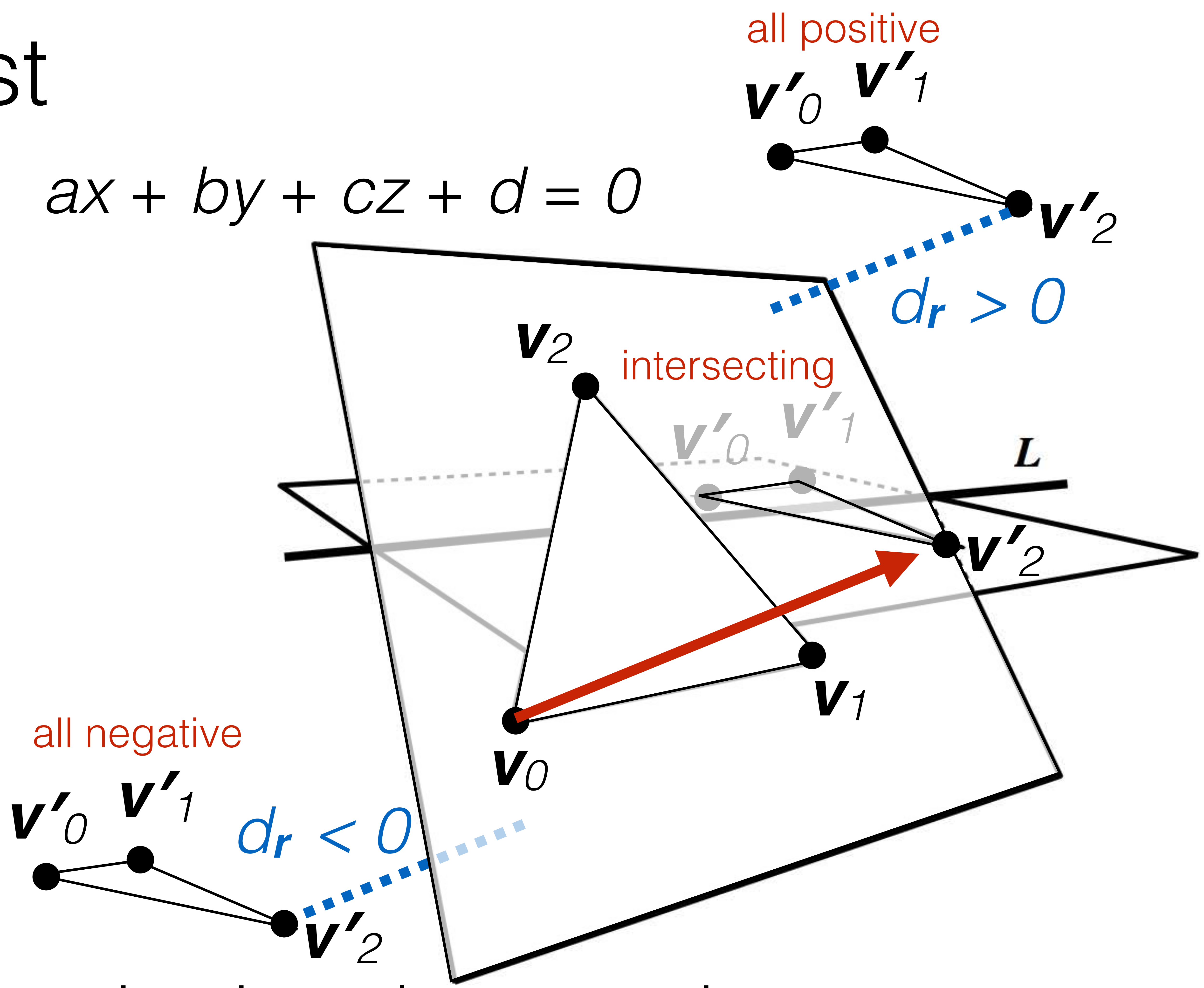
$ax + by + cz + d = 0$



- Plane coefficients can be computed from points of triangle
  - $\mathbf{n} = [a,b,c] = (\mathbf{v}_1 - \mathbf{v}_0) \times (\mathbf{v}_2 - \mathbf{v}_0)$
  - $d = - \mathbf{v}_2 \cdot \mathbf{n}$

3D Triangle-Triangle Test

1. Compute plane equation of triangle 2.
2. Reject as trivial if all points of triangle 1 are on same side.
3. Compute plane equation of triangle 1.
4. Reject as trivial if all points of triangle 2 are on same side.
5. Compute intersection line and project onto largest axis.
6. Compute the intervals for each triangle.
7. Intersect the intervals.



Input points into plane equation.

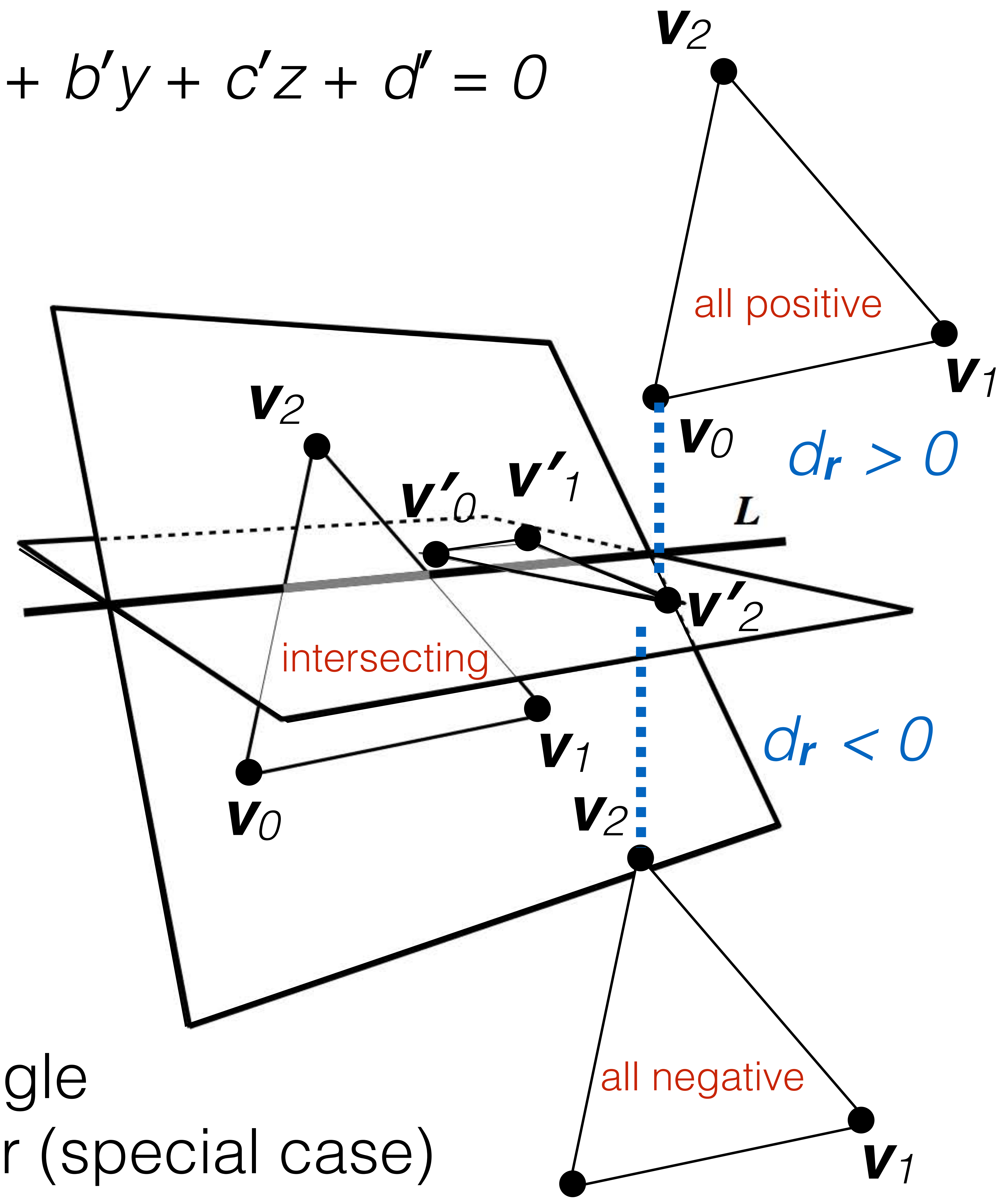
If all have the same sign, planes of triangles do not intersect

Möller 1997

3D Triangle-Triangle Test

$a'x + b'y + c'z + d' = 0$

1. Compute plane equation of triangle 2.
2. Reject as trivial if all points of triangle 1 are on same side.
3. Compute plane equation of triangle 1.
4. Reject as trivial if all points of triangle 2 are on same side.
5. Compute intersection line and project onto largest axis.
6. Compute the intervals for each triangle.
7. Intersect the intervals.



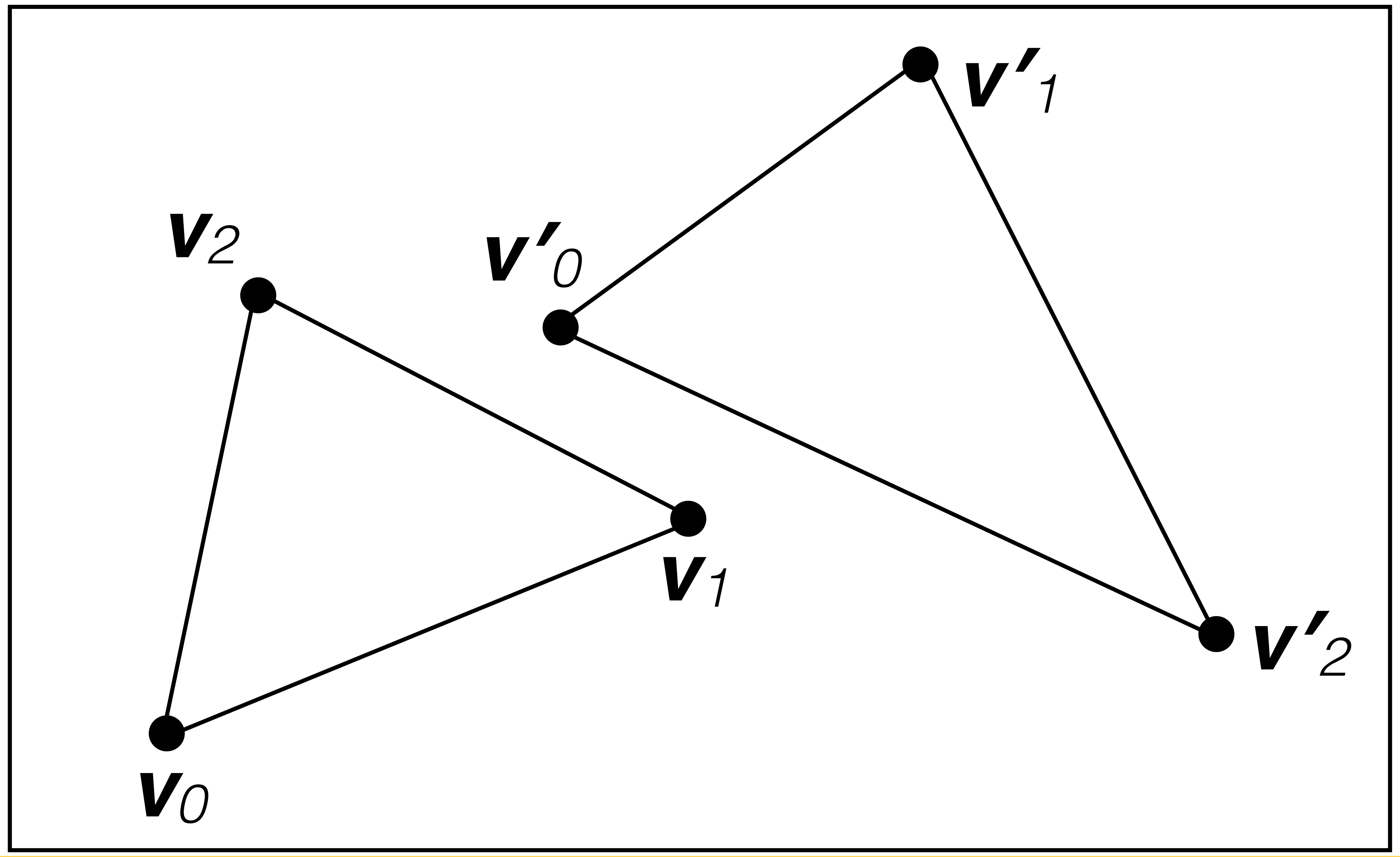
Repeat for other plane of other triangle

If all evaluations are zero, triangles are co-planar (special case)

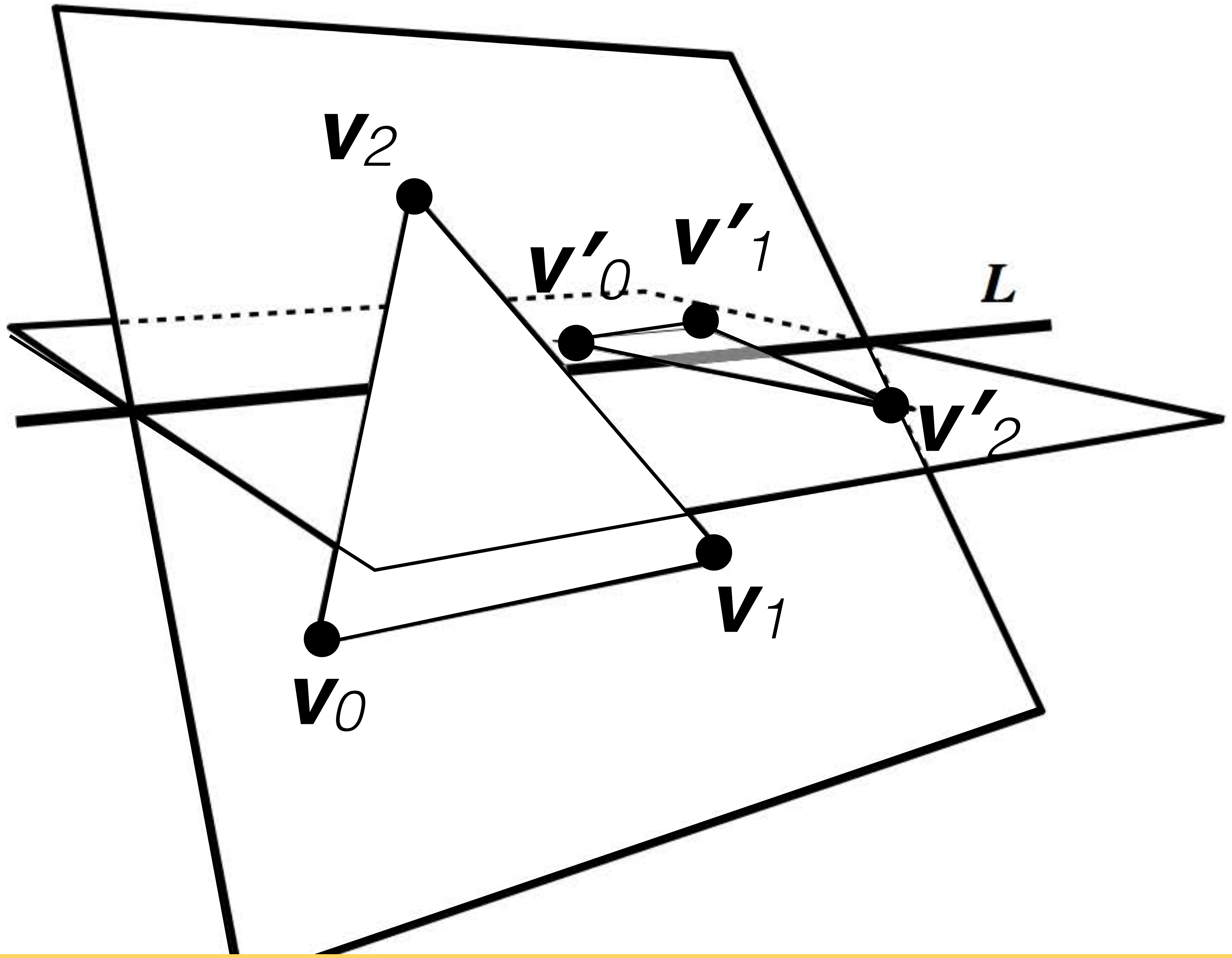
Three possible cases can occur based on evaluation of vertices of one triangle against the plane of the other triangle

1. Triangle does not intersect plane
(all positive or all negative evaluations)
return non-collision

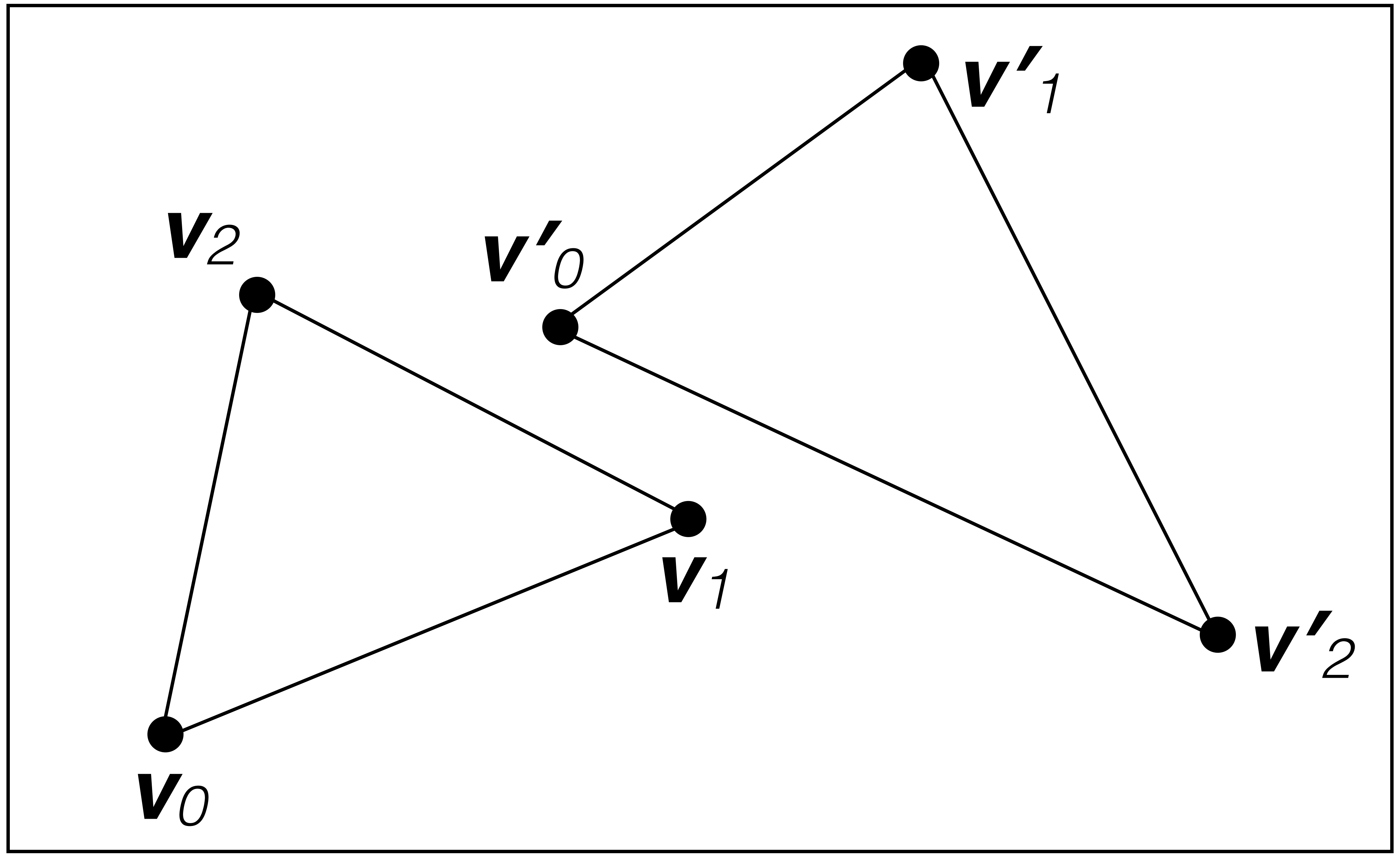
2. Triangles are coplanar
(all evaluations are zero)

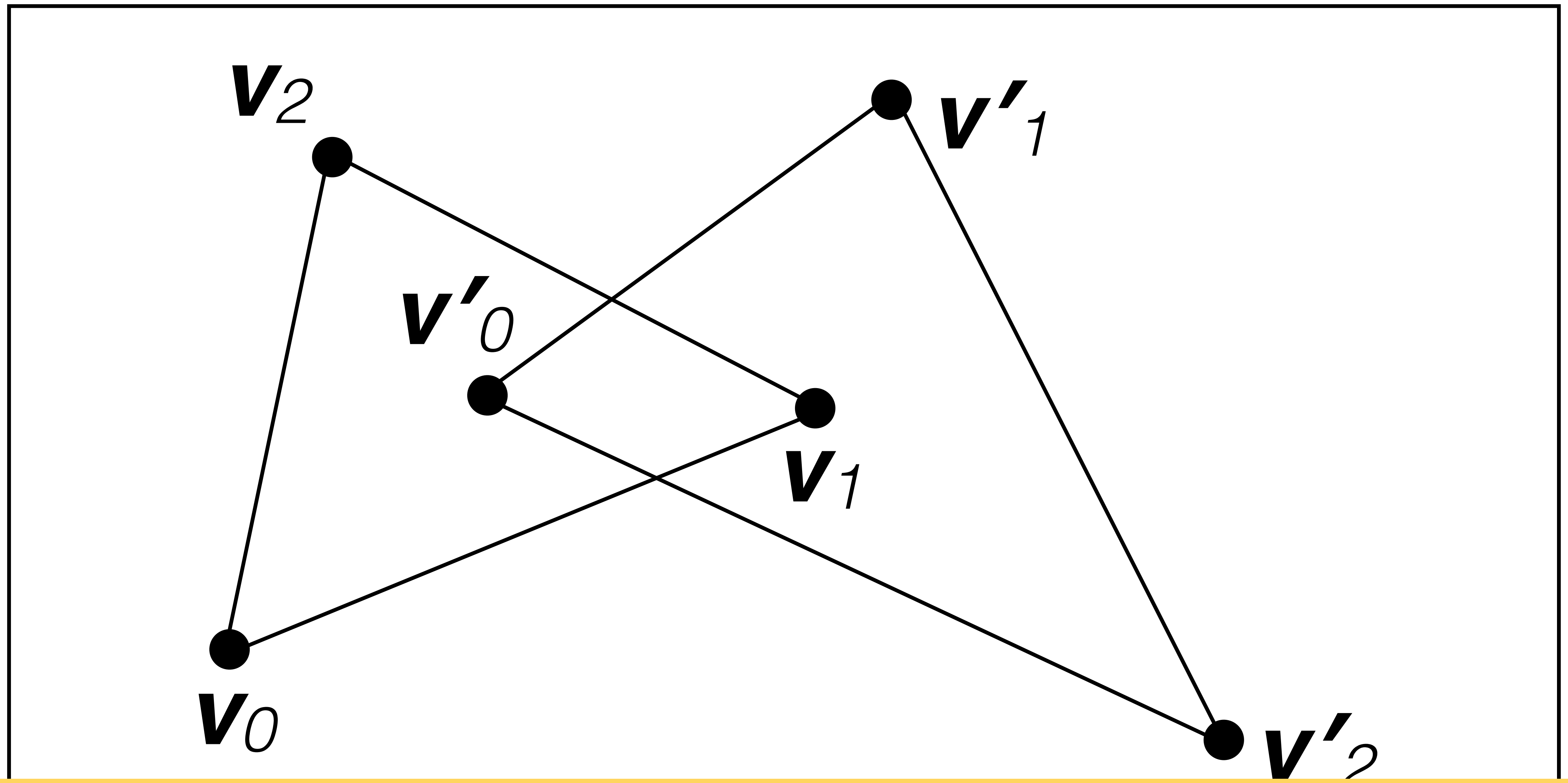


3. Triangles are not coplanar
(positive and negative evaluations)



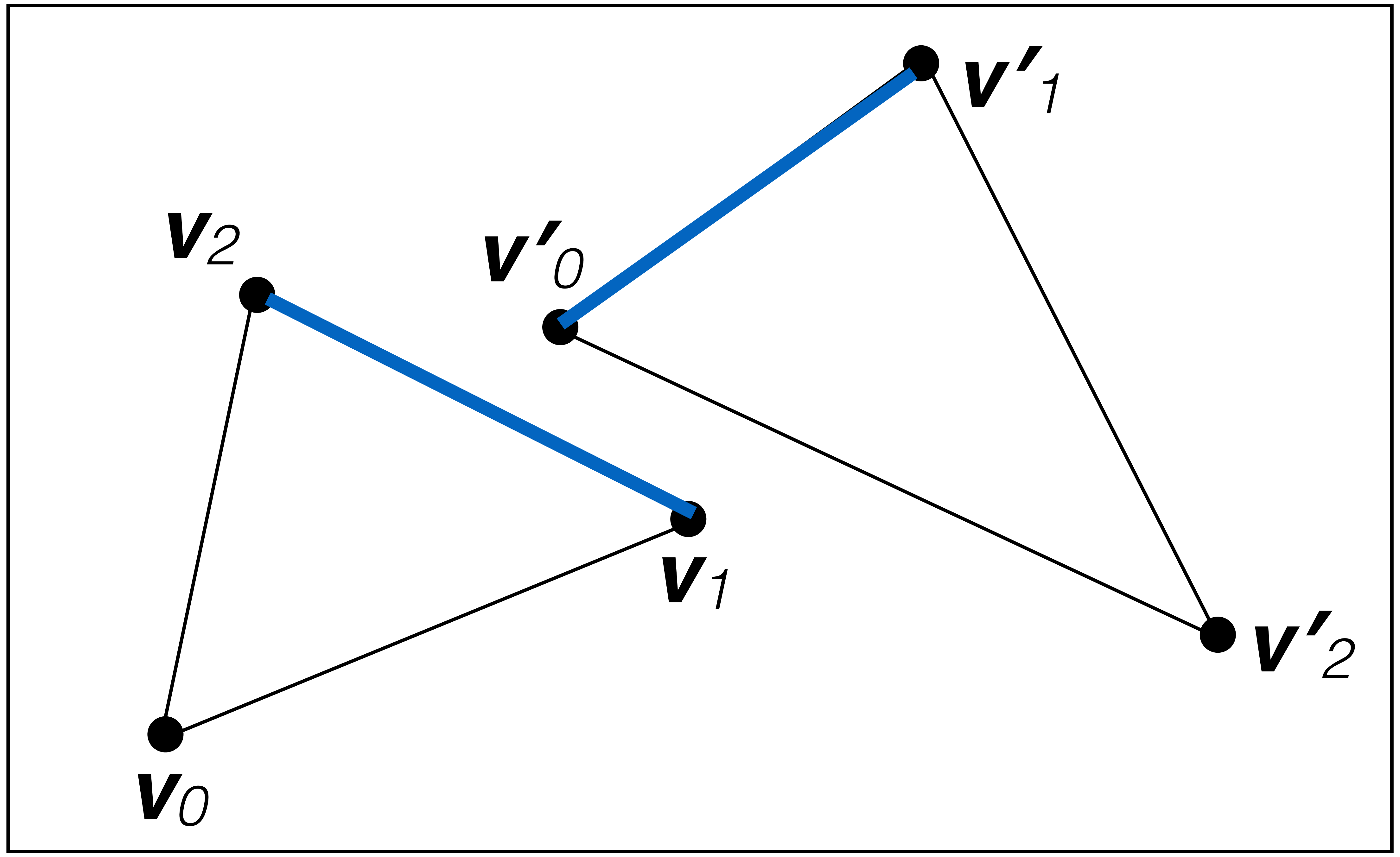
Suppose triangles are coplanar



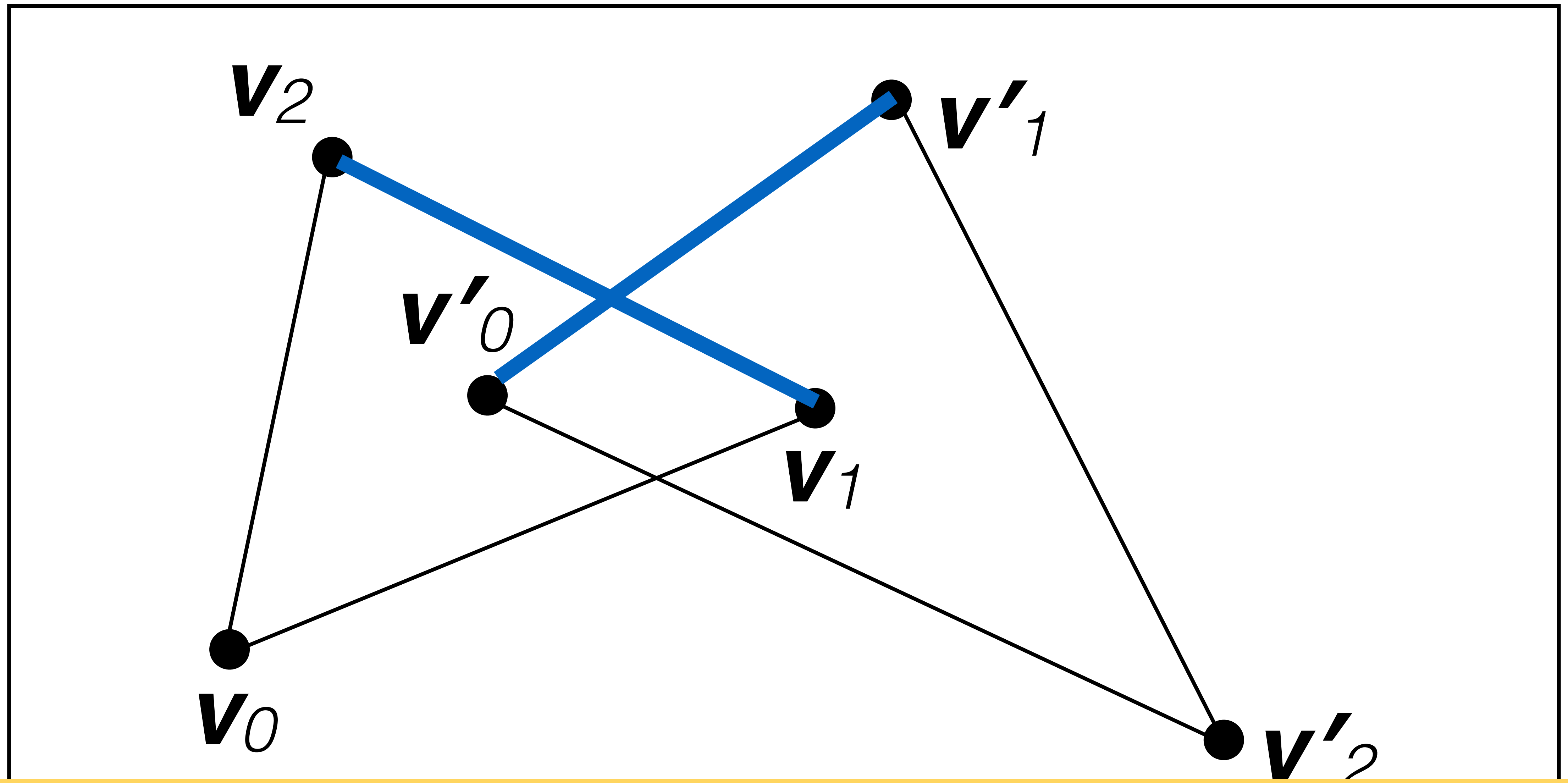


Slide borrowed from Michigan Robotics autorob.org

Suppose triangles are coplanar



Compare each pair of line segments

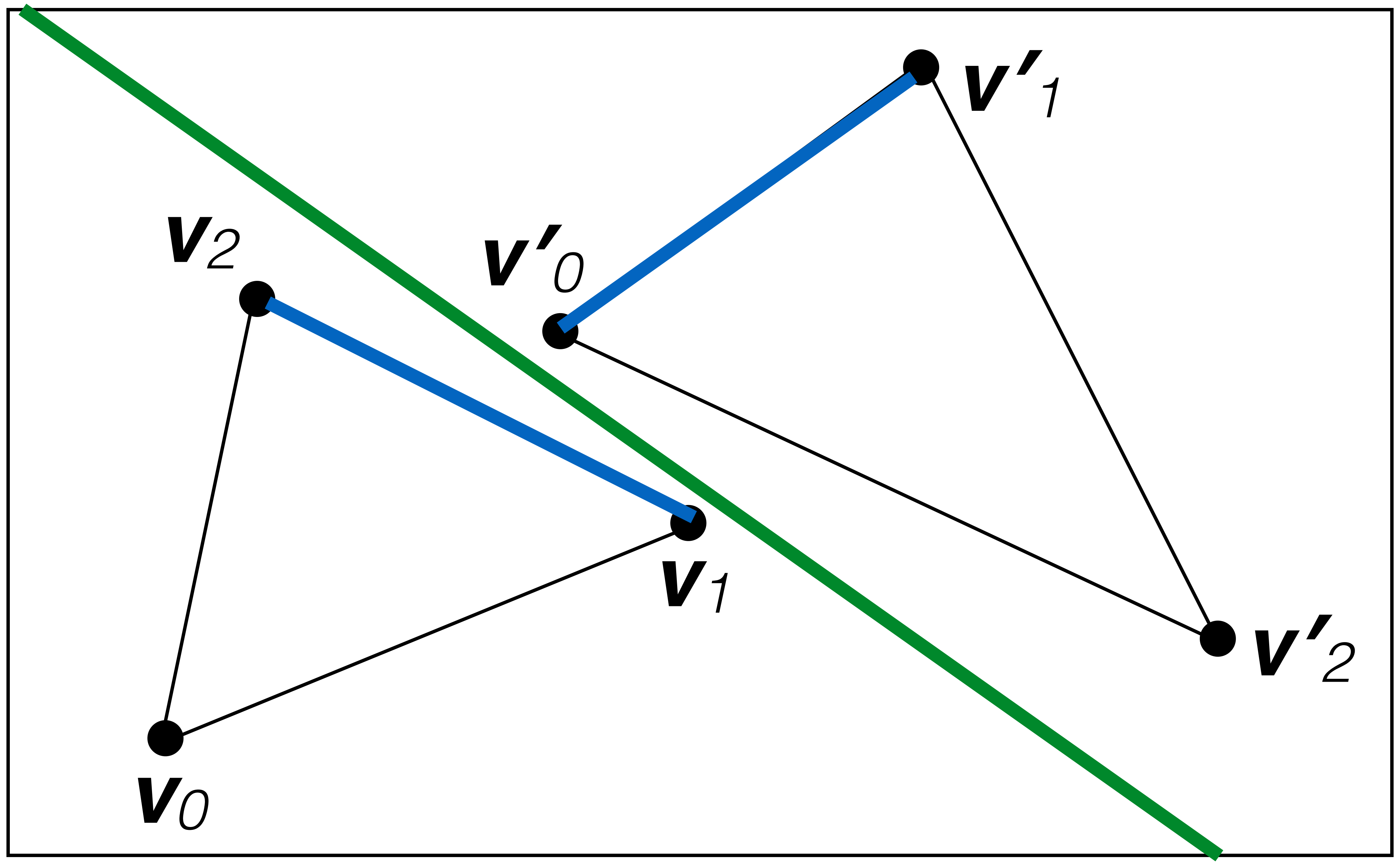


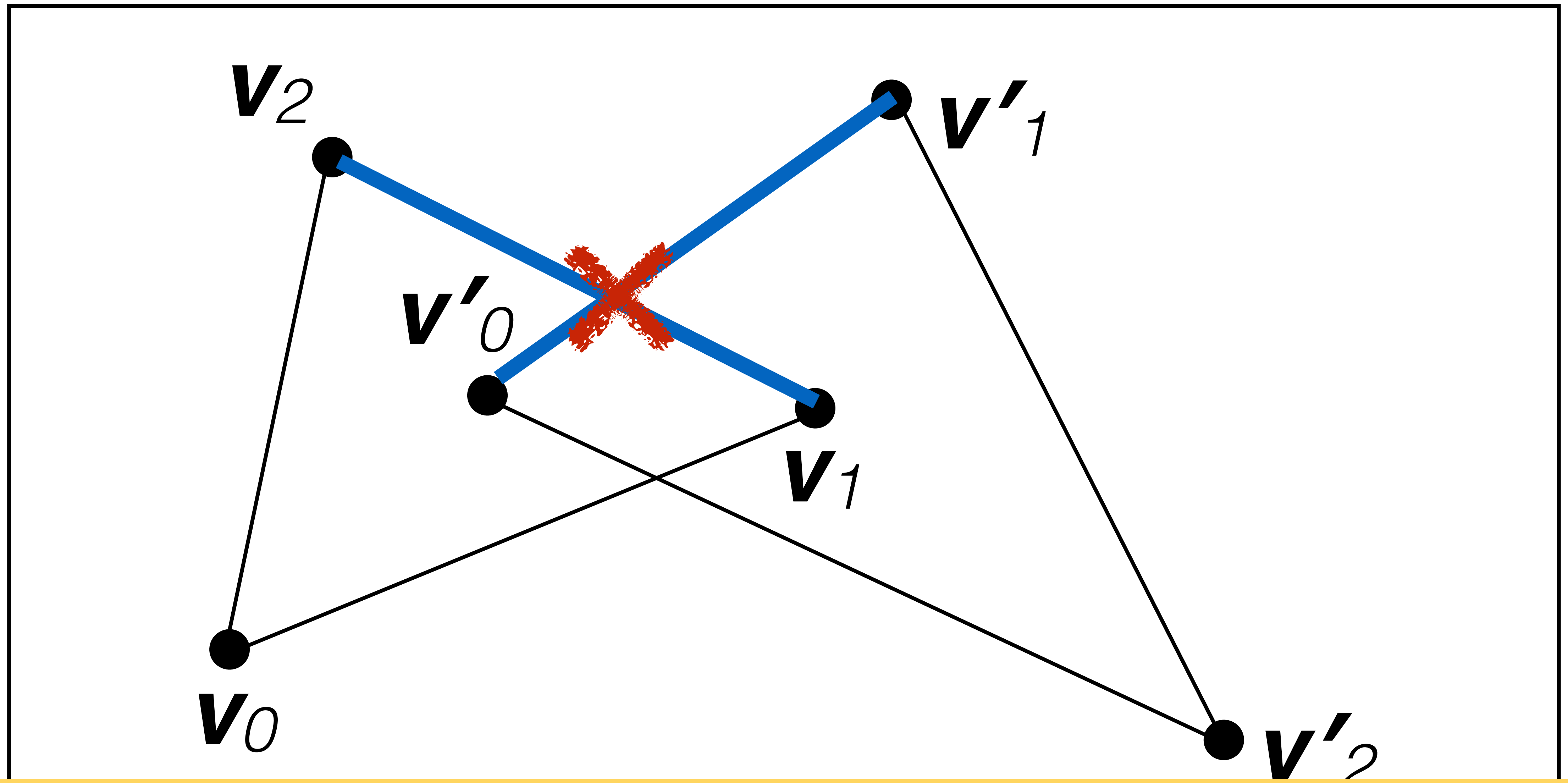
Slide borrowed from Michigan Robotics autorob.org

Suppose triangles are coplanar

Compare each pair of line segments

Find intersection point as solution to linear system

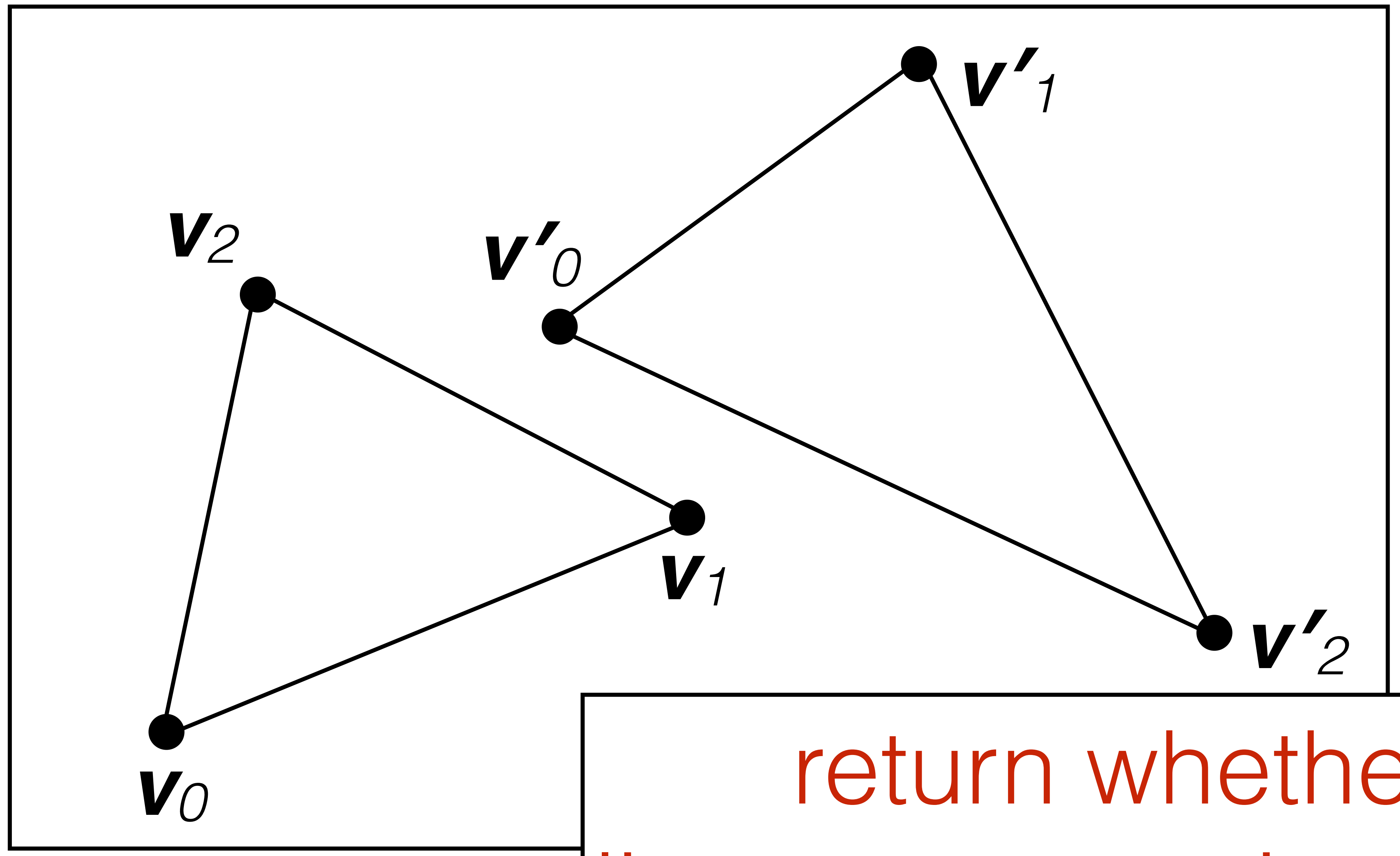




Three possible cases can occur based on
evaluation of vertices of one triangle against the plane of the other triangle

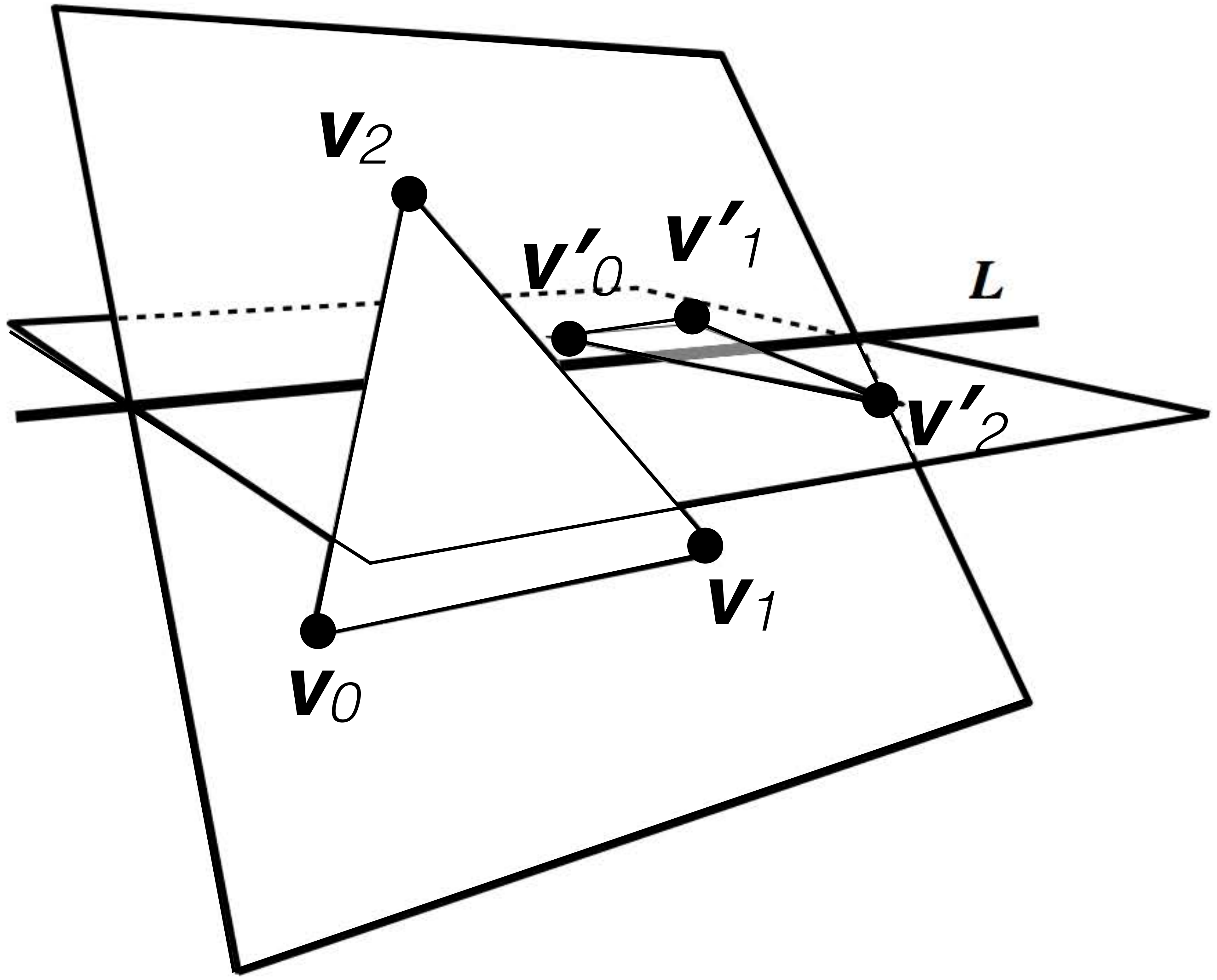
1. Triangle does not intersect plane
(all positive or all negative evaluations)
return non-collision

2. Triangles are coplanar
(all evaluations are zero)



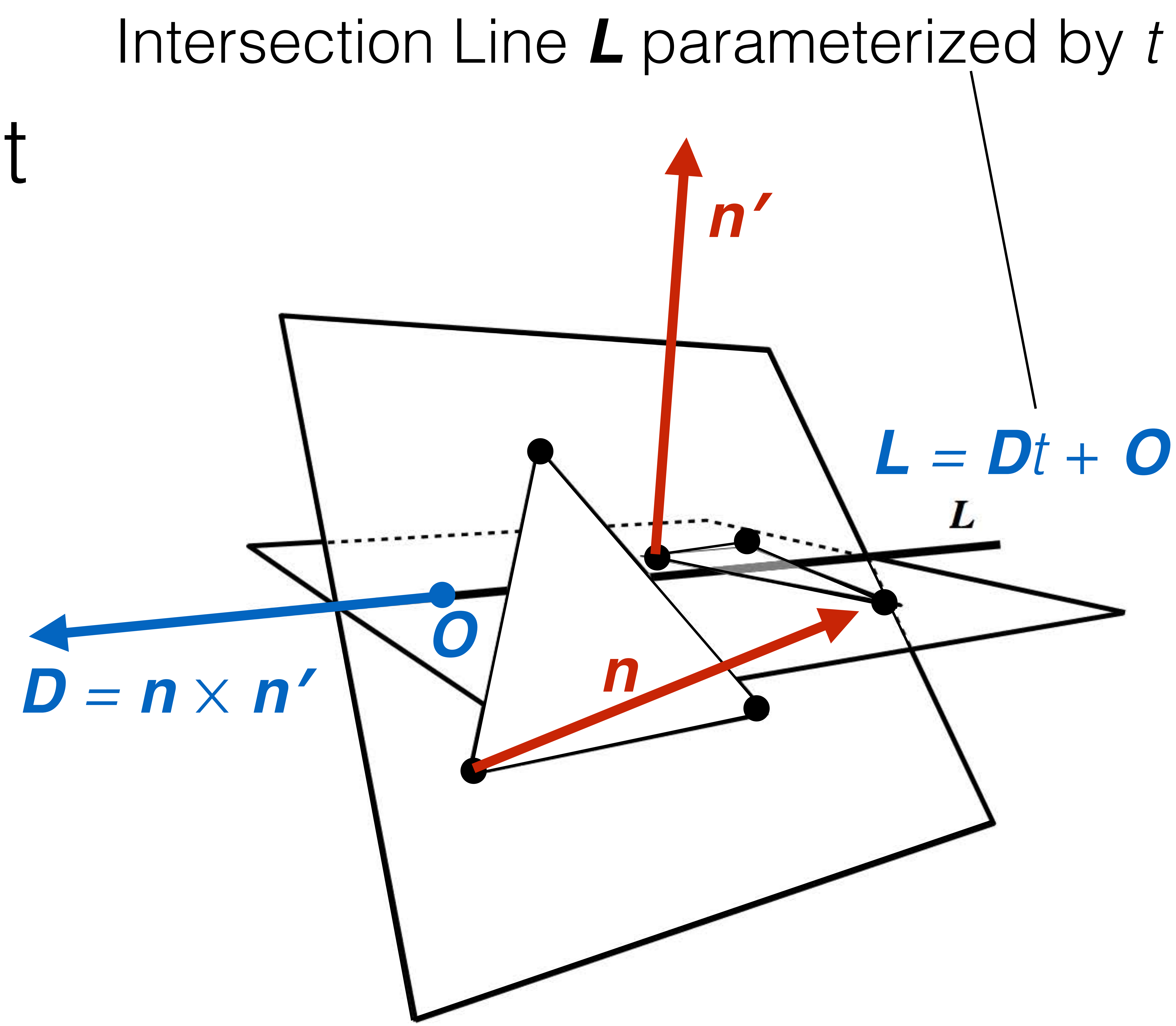
return whether
line segments intersect

3. Triangles are not coplanar
(positive and negative evaluations)



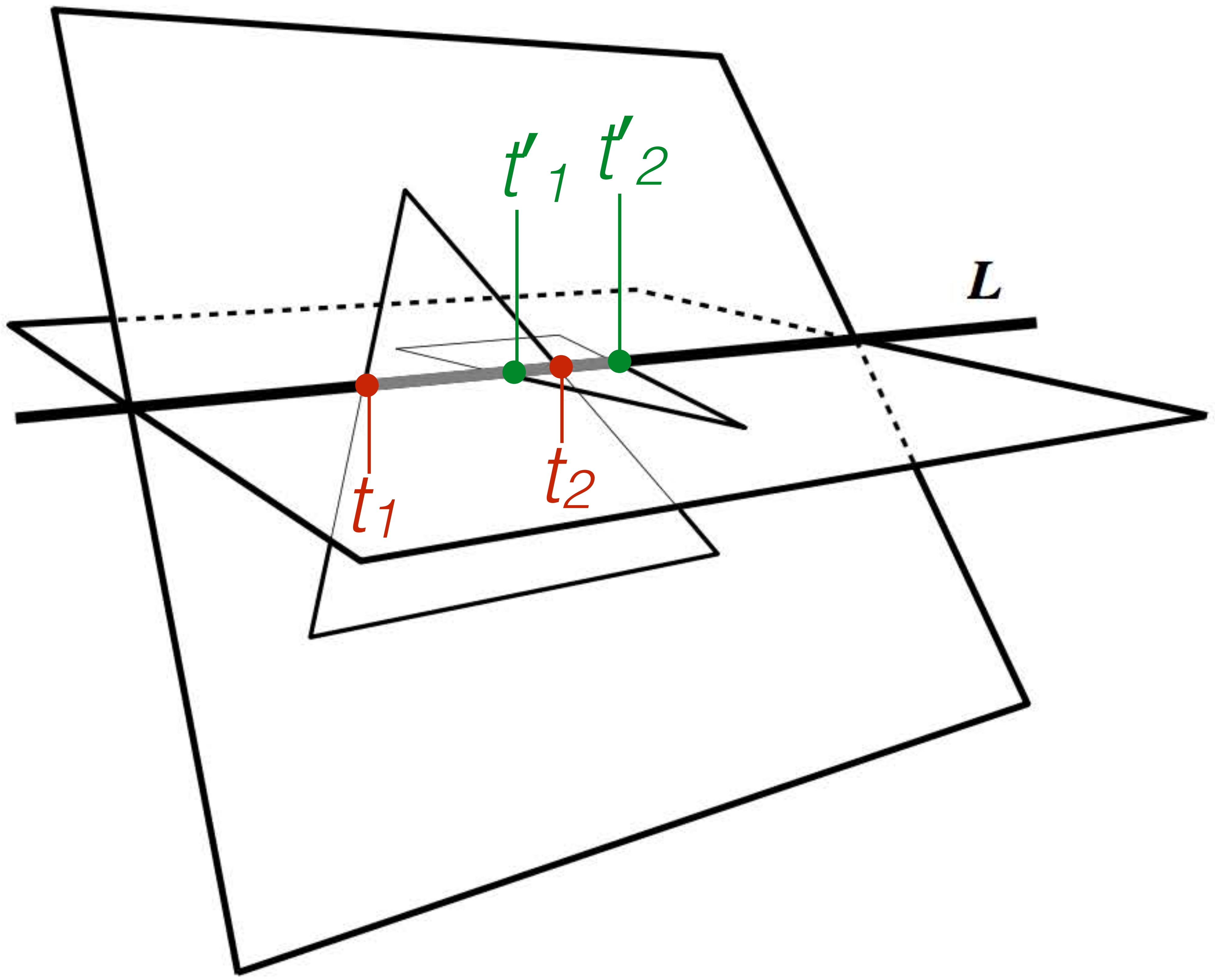
3D Triangle-Triangle Test

1. Compute plane equation of triangle 2.
2. Reject as trivial if all points of triangle 1 are on same side.
3. Compute plane equation of triangle 1.
4. Reject as trivial if all points of triangle 2 are on same side.
5. Compute intersection line and project onto largest axis.
6. Compute the intervals for each triangle.
7. Intersect the intervals.

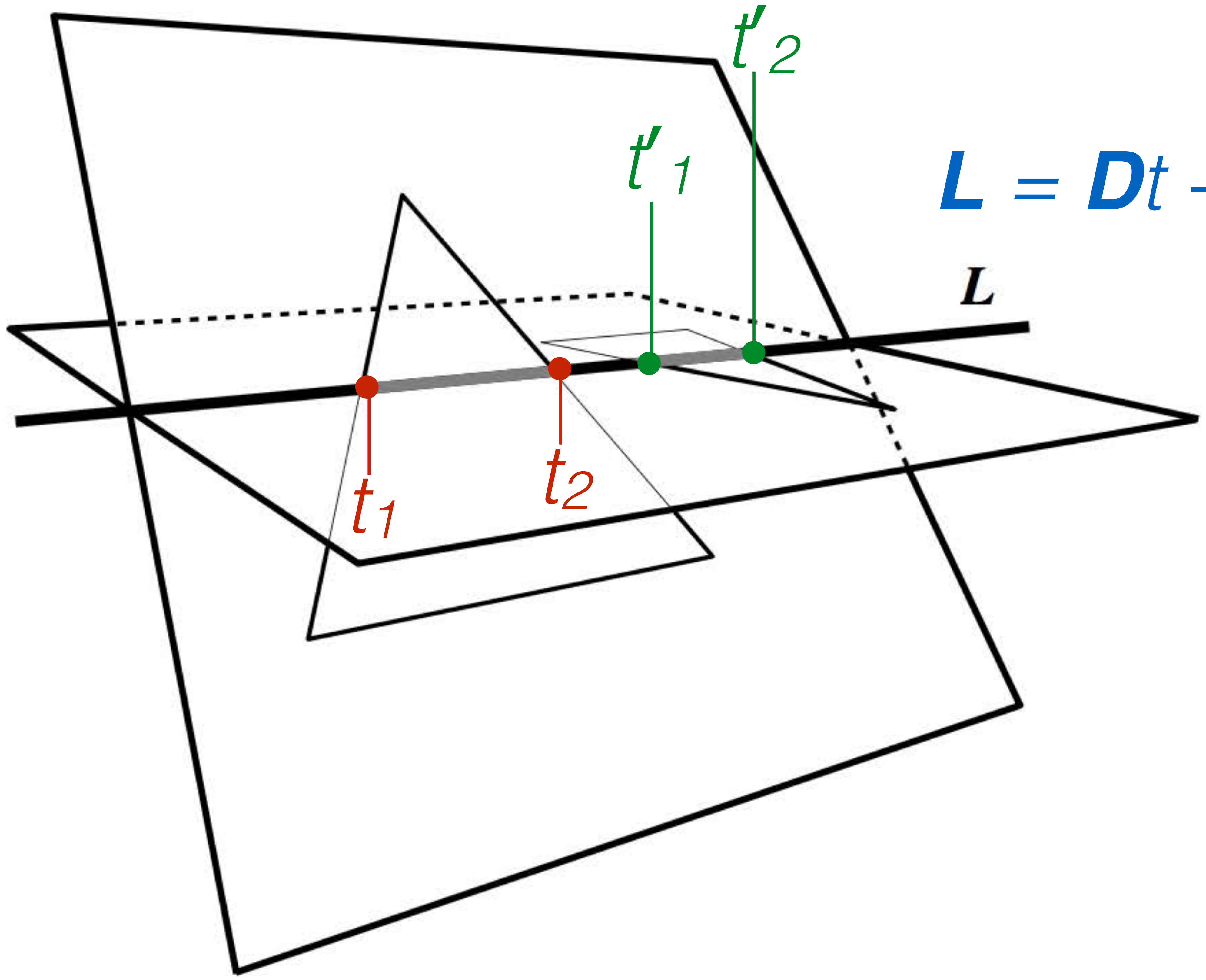


Two possible cases can occur based on interval spans ($[t_1,t_2]$, $[t'_1,t'_2]$) of triangle intersections with ***L***

Collision, if intervals overlap



No Collision, if intervals do not overlap

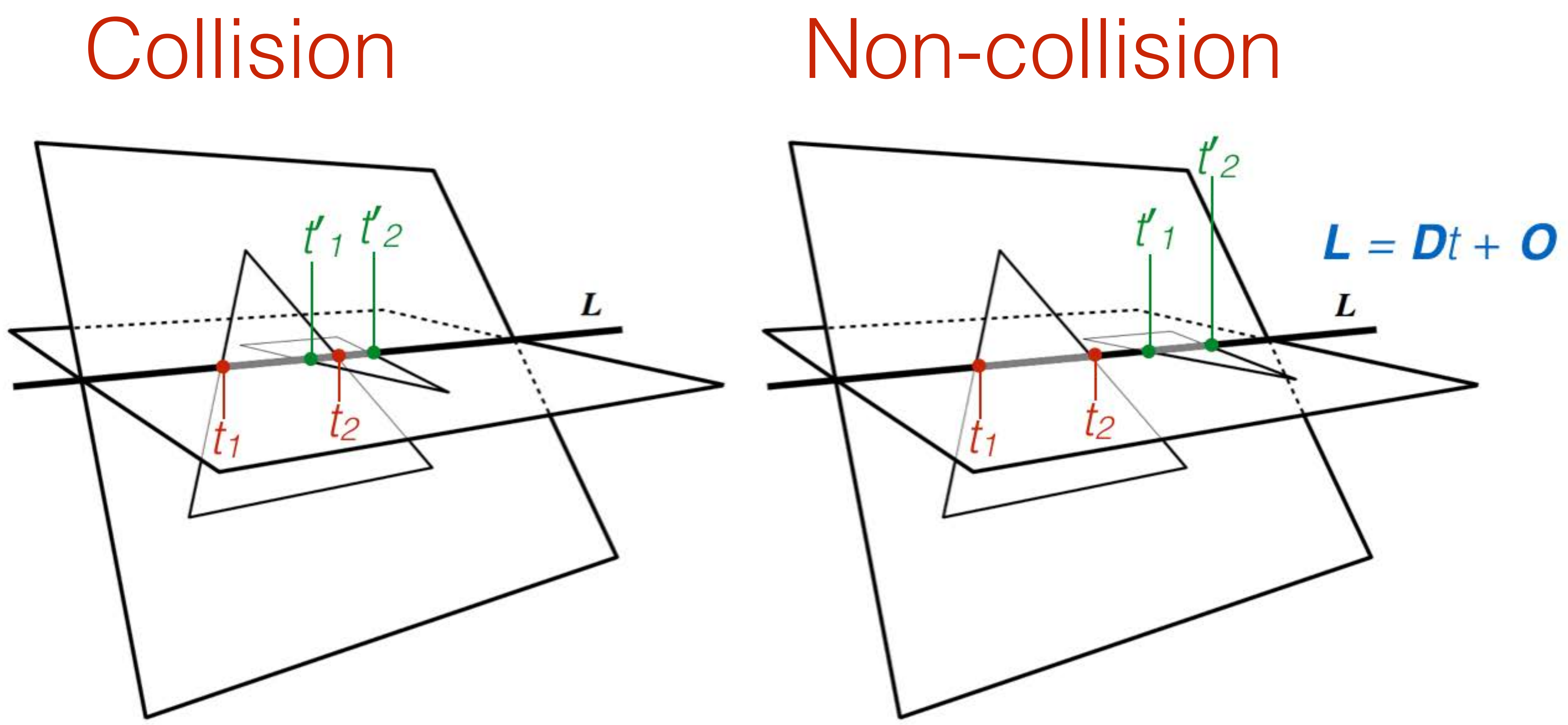


$\boldsymbol{L} = \boldsymbol{D}t + \boldsymbol{O}$

Slide borrowed from Michigan Robotics autorob.org

3D Triangle-Triangle Test

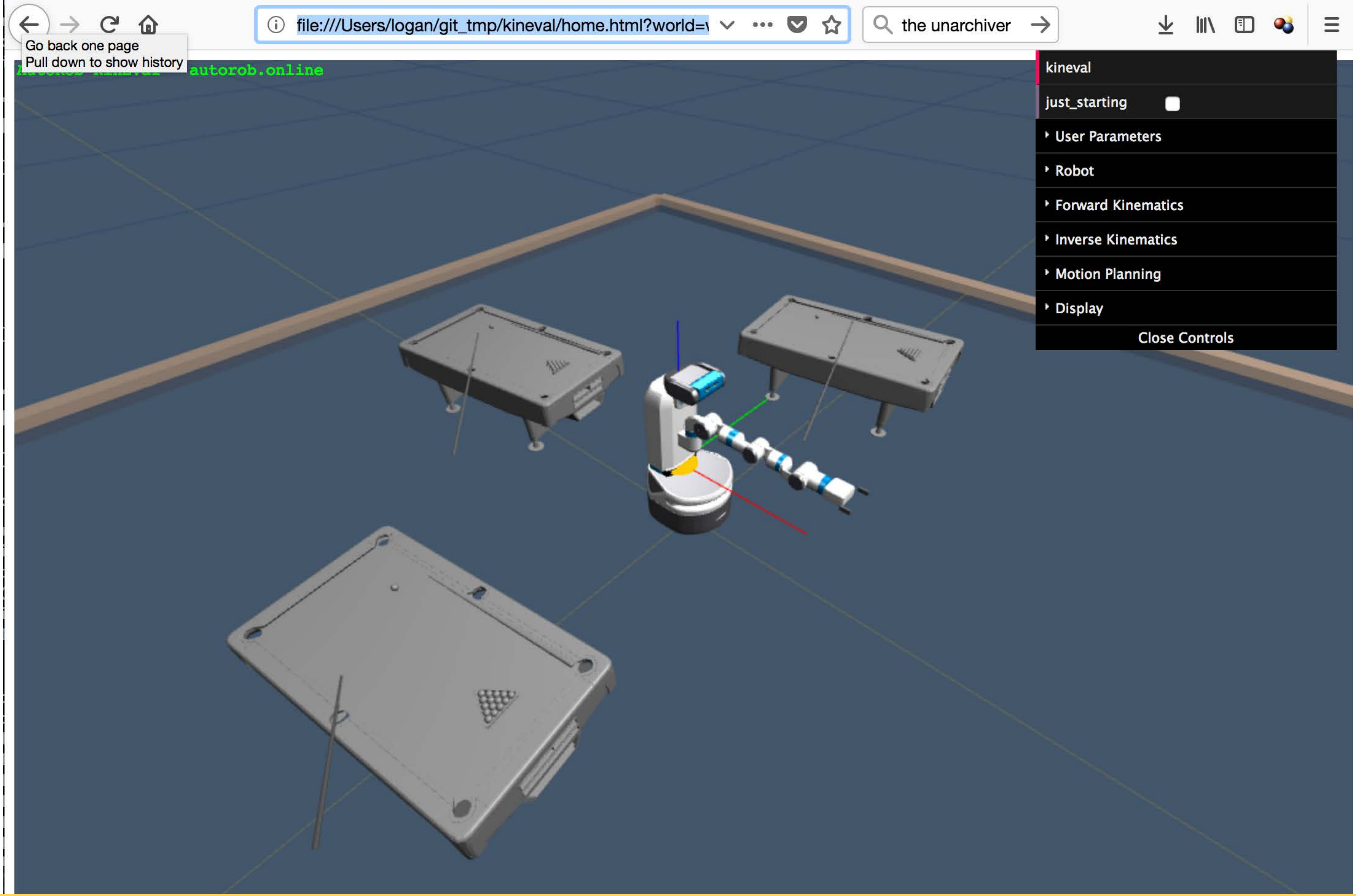
1. Compute plane equation of triangle 2.
2. Reject as trivial if all points of triangle 1 are on same side.
3. Compute plane equation of triangle 1.
4. Reject as trivial if all points of triangle 2 are on same side.
5. Compute intersection line and project onto largest axis.
6. Compute the intervals for each triangle.
7. Intersect the intervals.

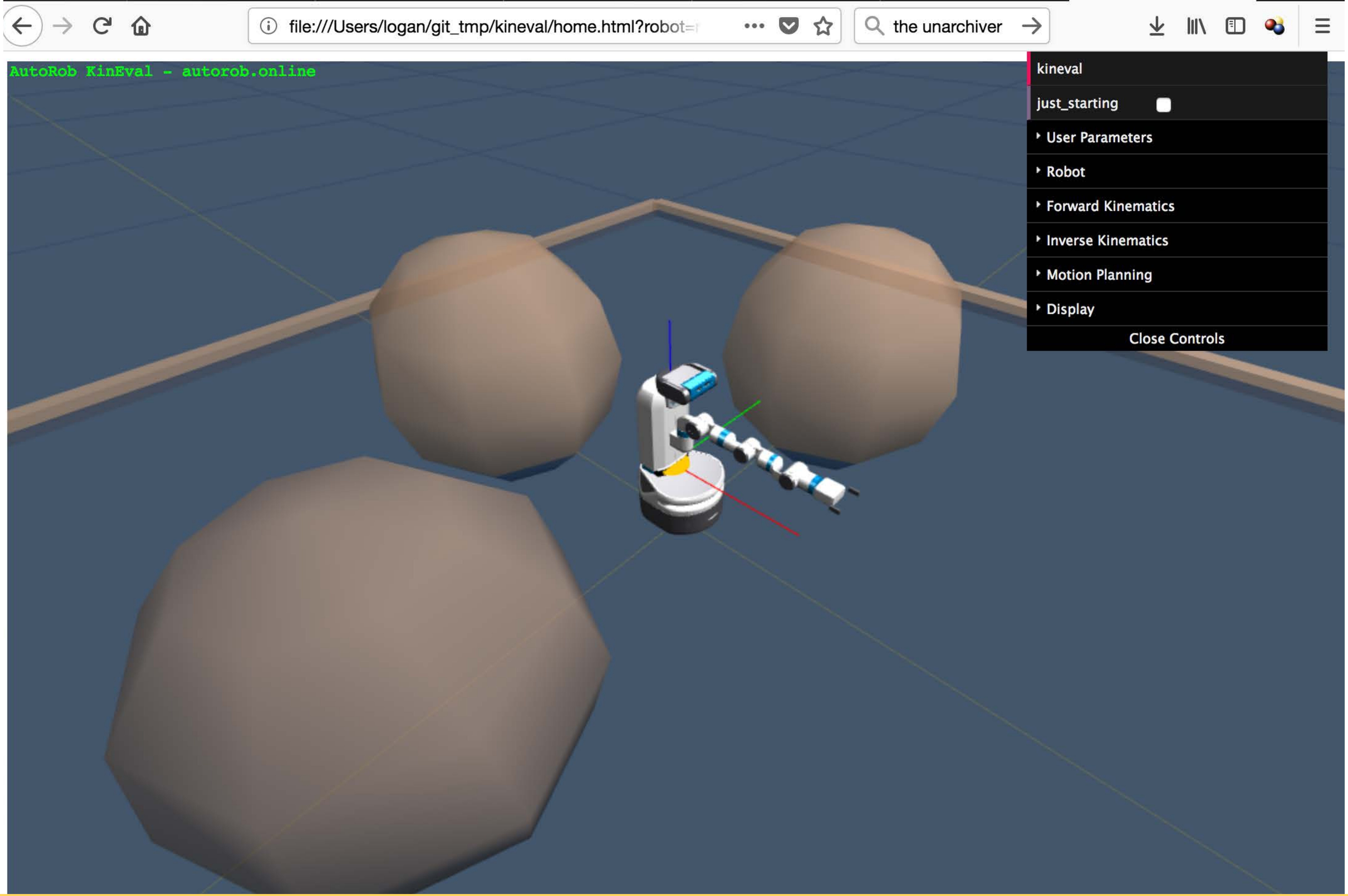


How many triangle tests must be performed?

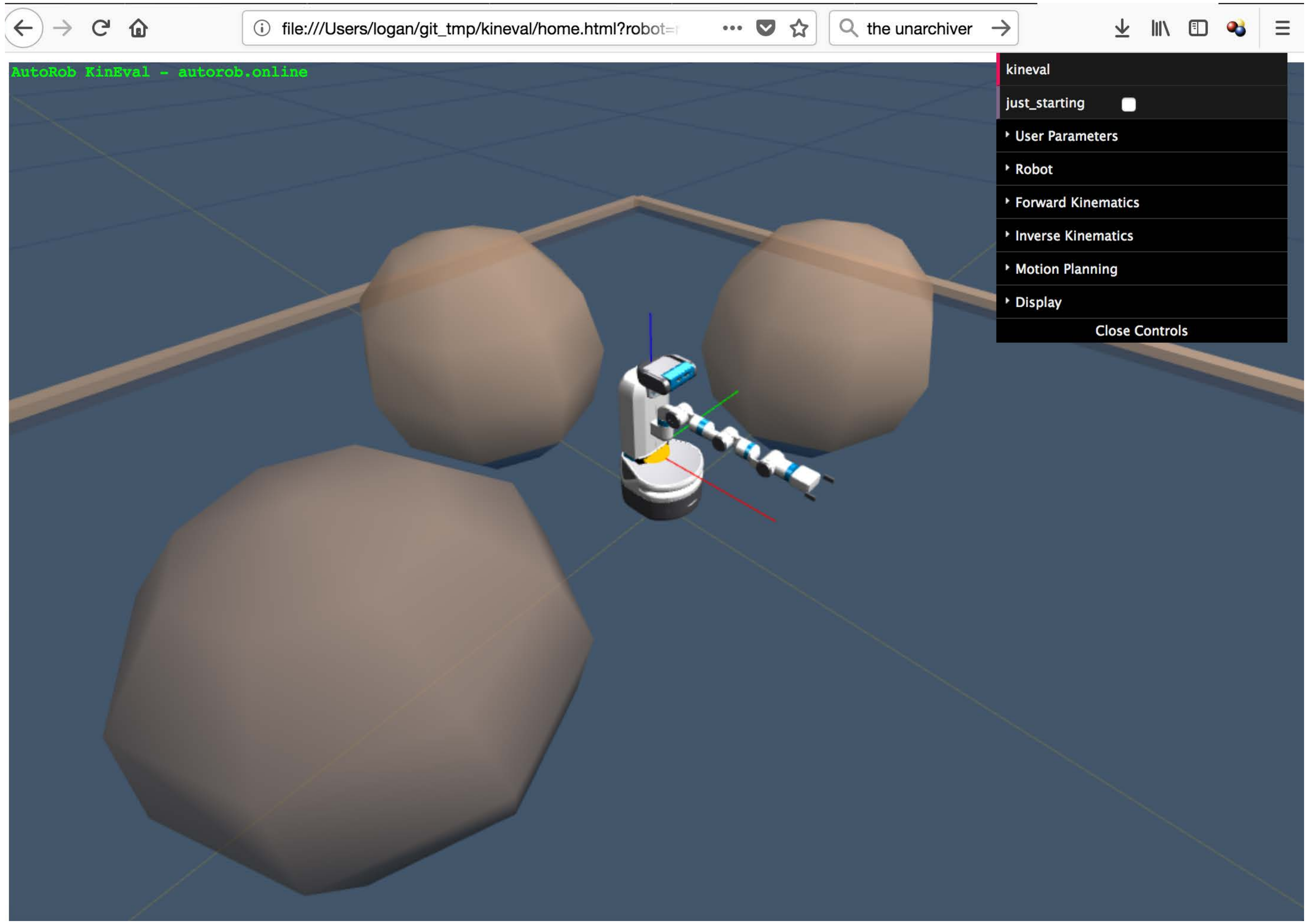
How many triangle tests must be performed?

Can we reduce the number of tests to evaluate?


file:///Users/logan/git_tmp/kineval/home.html?world=
the unarchiver
Go back one page
Pull down to show history
autorob.online
kineval
just_starting
User Parameters
Robot
Forward Kinematics
Inverse Kinematics
Motion Planning
Display
Close Controls

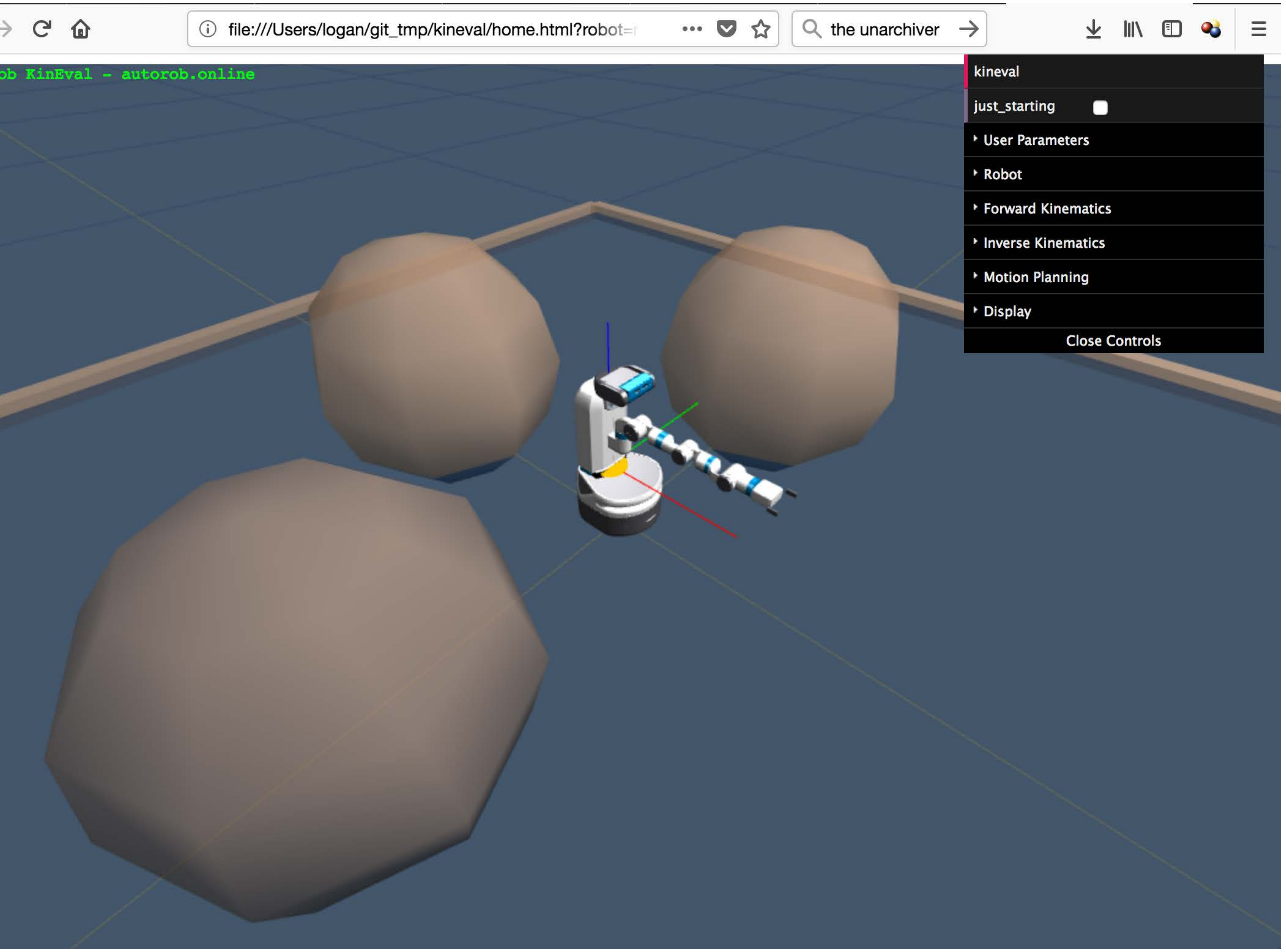

file:///Users/logan/git_tmp/kineval/home.html?robot=
the unarchiver
AutoRob KinEval - autorob.online
kineval
just_starting
User Parameters
Robot
Forward Kinematics
Inverse Kinematics
Motion Planning
Display
Close Controls

KinEval approximates obstacles with bounding spheres

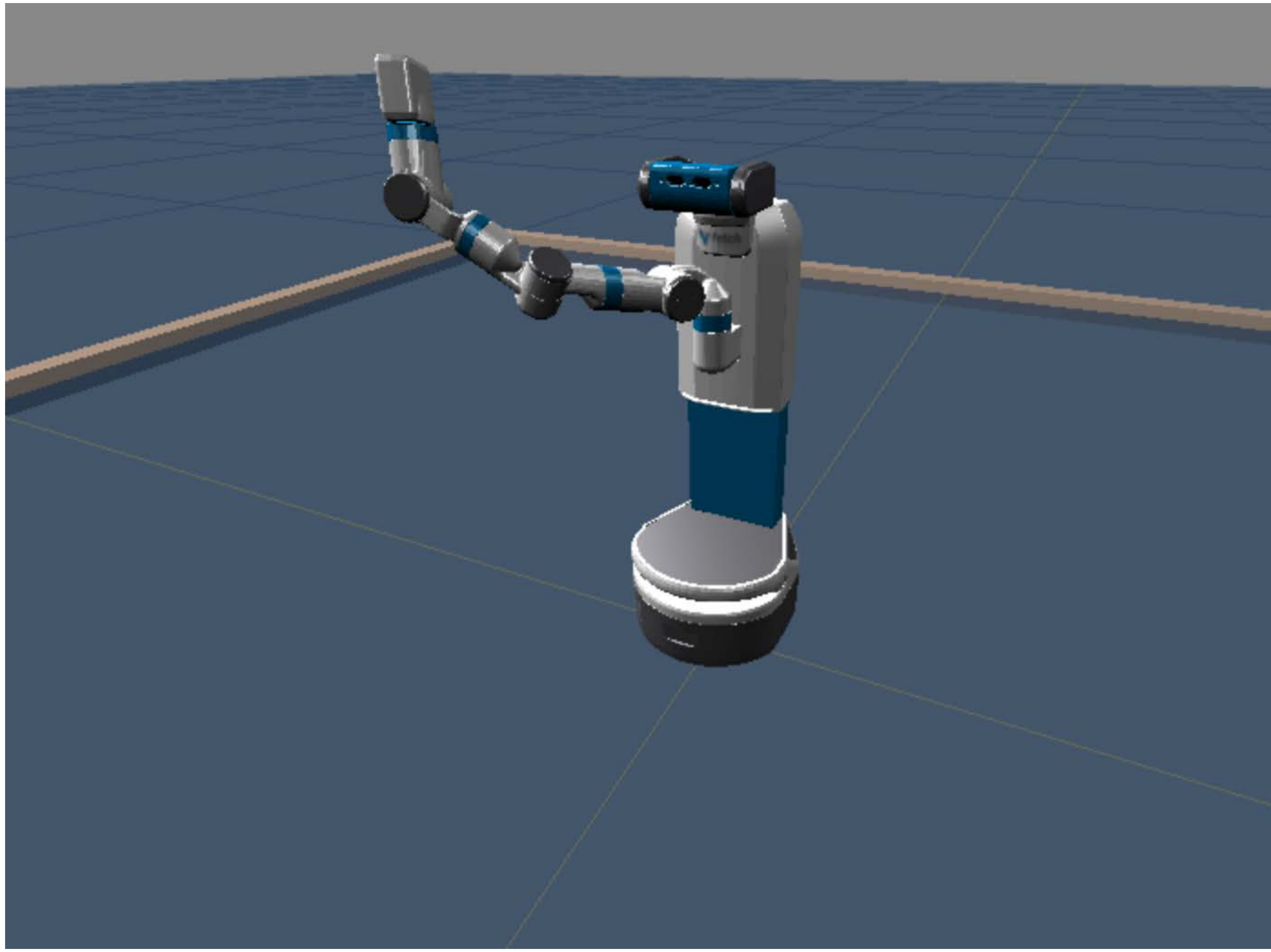


KinEval approximates

obstacles with bounding spheres



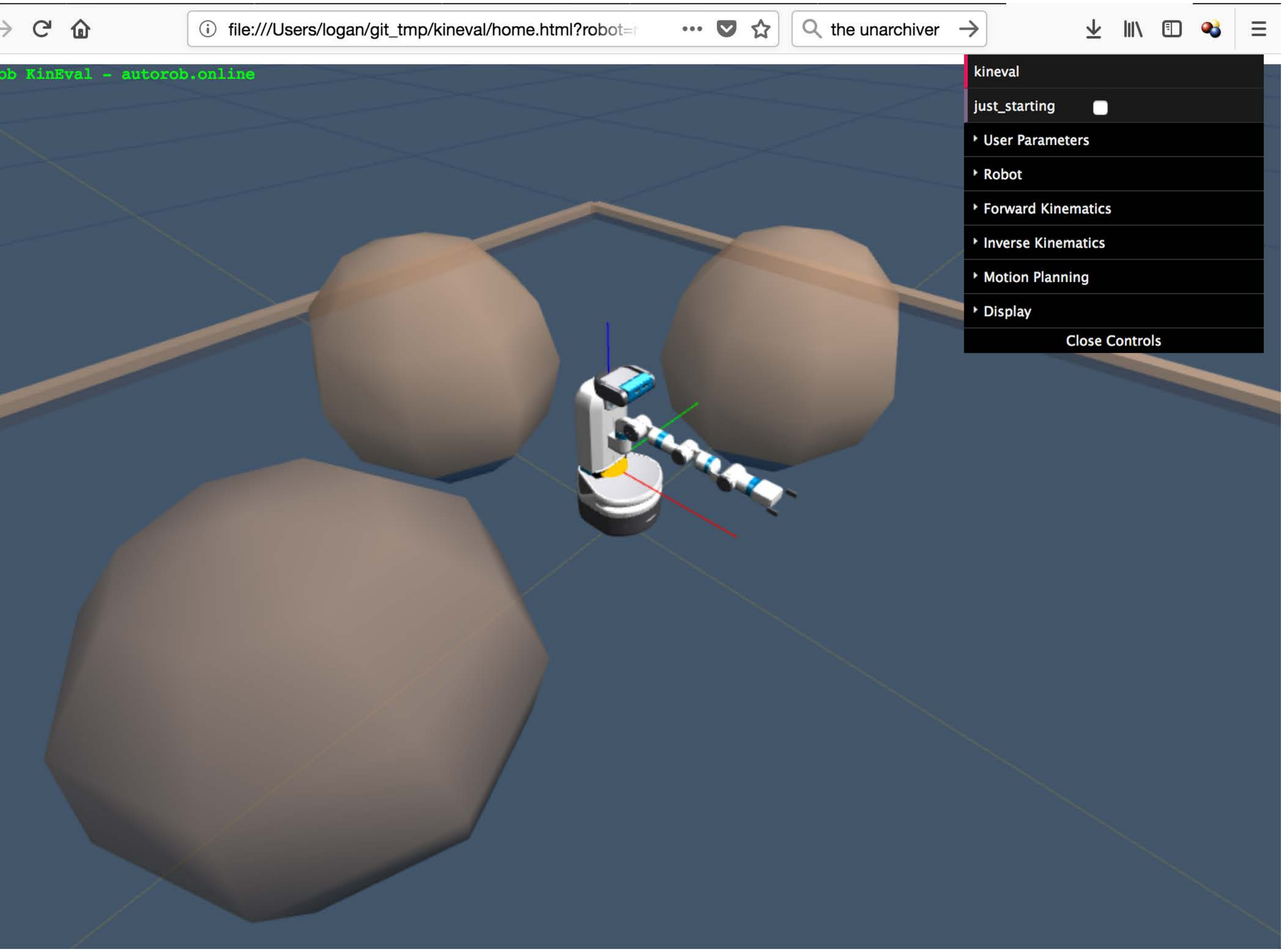
and the robot?



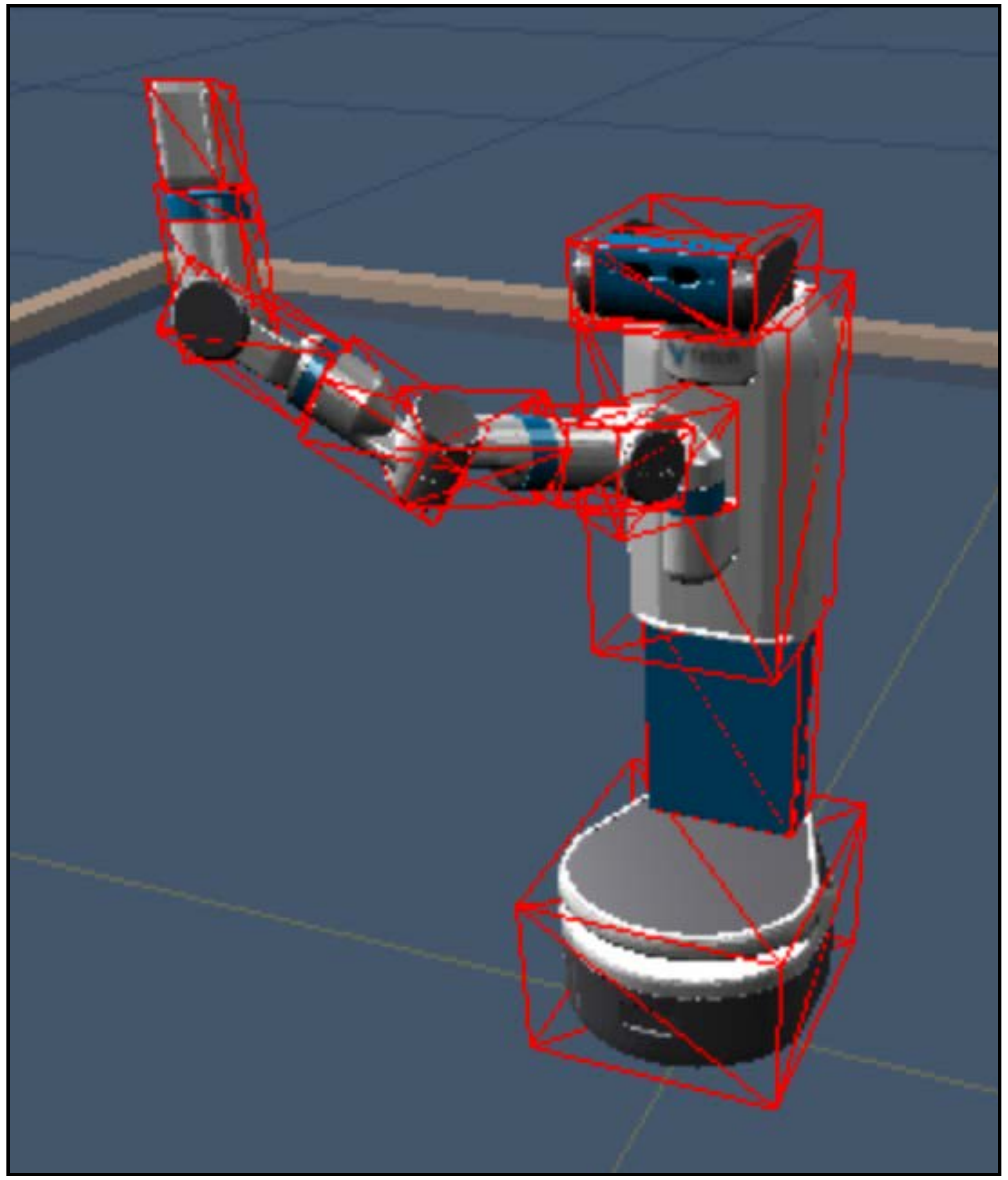
Slide borrowed from Michigan Robotics autorob.org

KinEval approximates

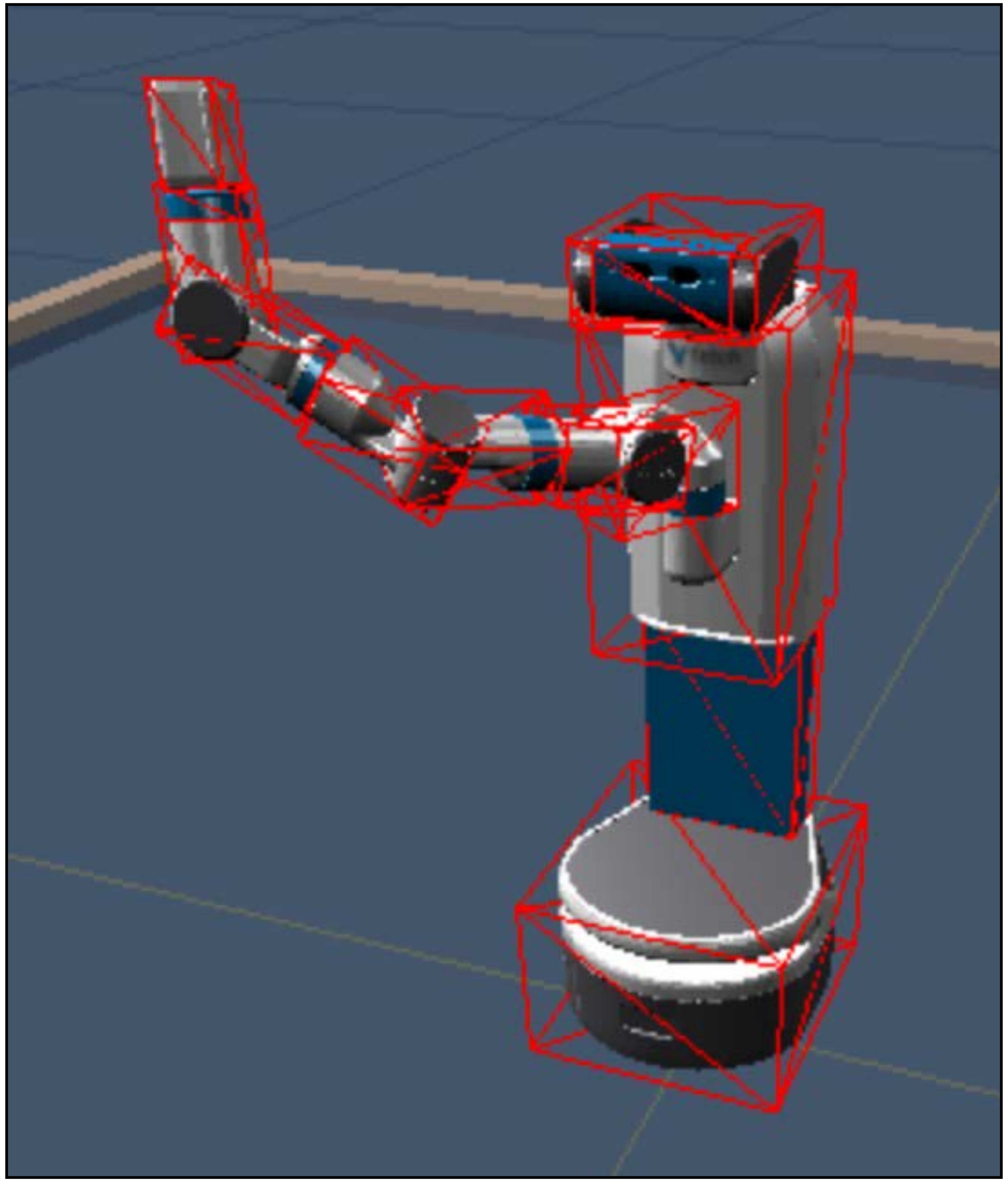
obstacles with bounding spheres



robot as bounding boxes

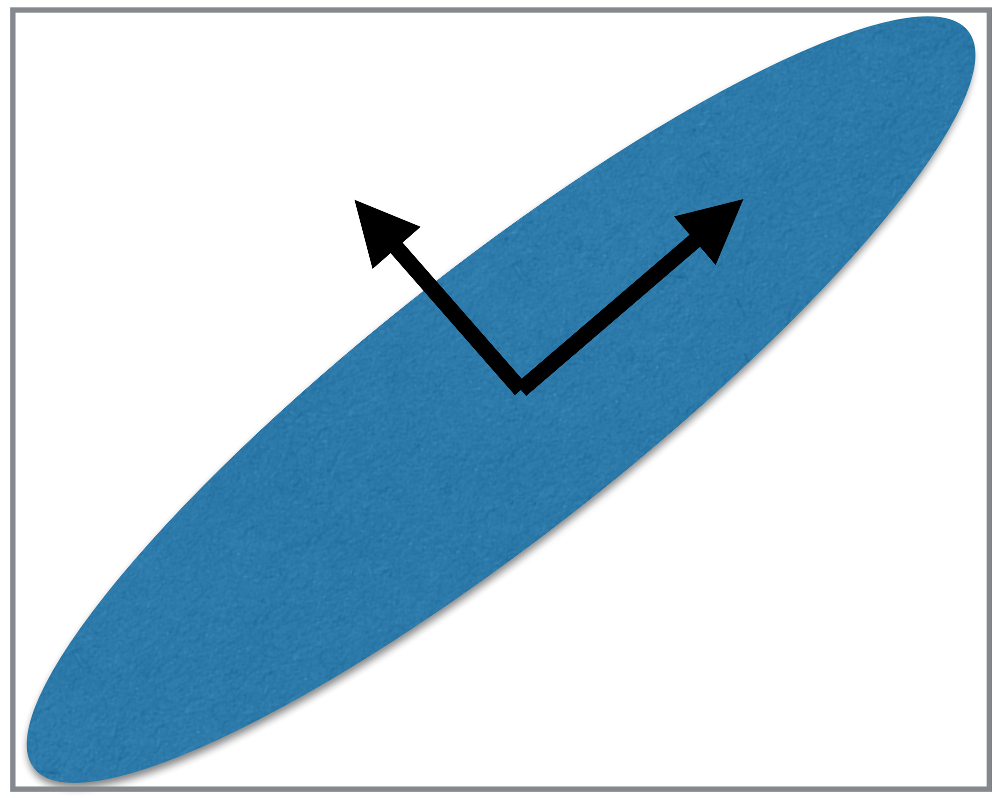


KinEval approximates link geometries with bounding boxes



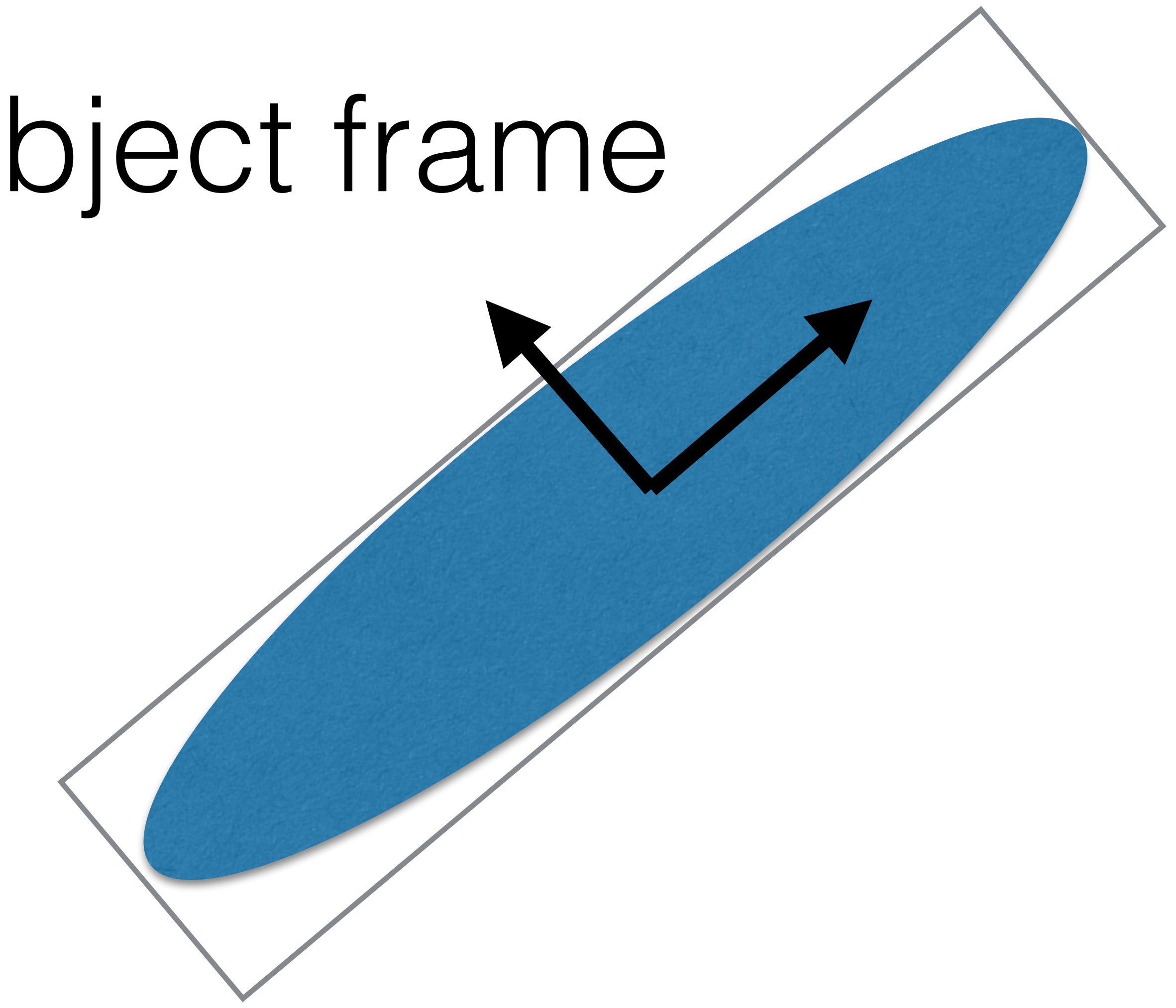
Slide borrowed from Michigan Robotics autorob.org

Bounding Boxes

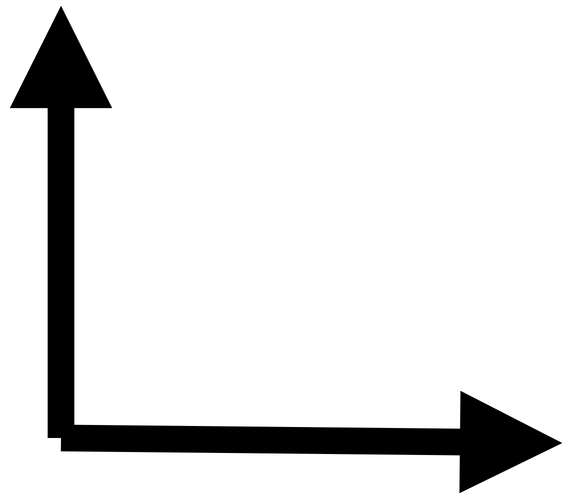


Axis-aligned Bounding Box
(AABB)

object frame



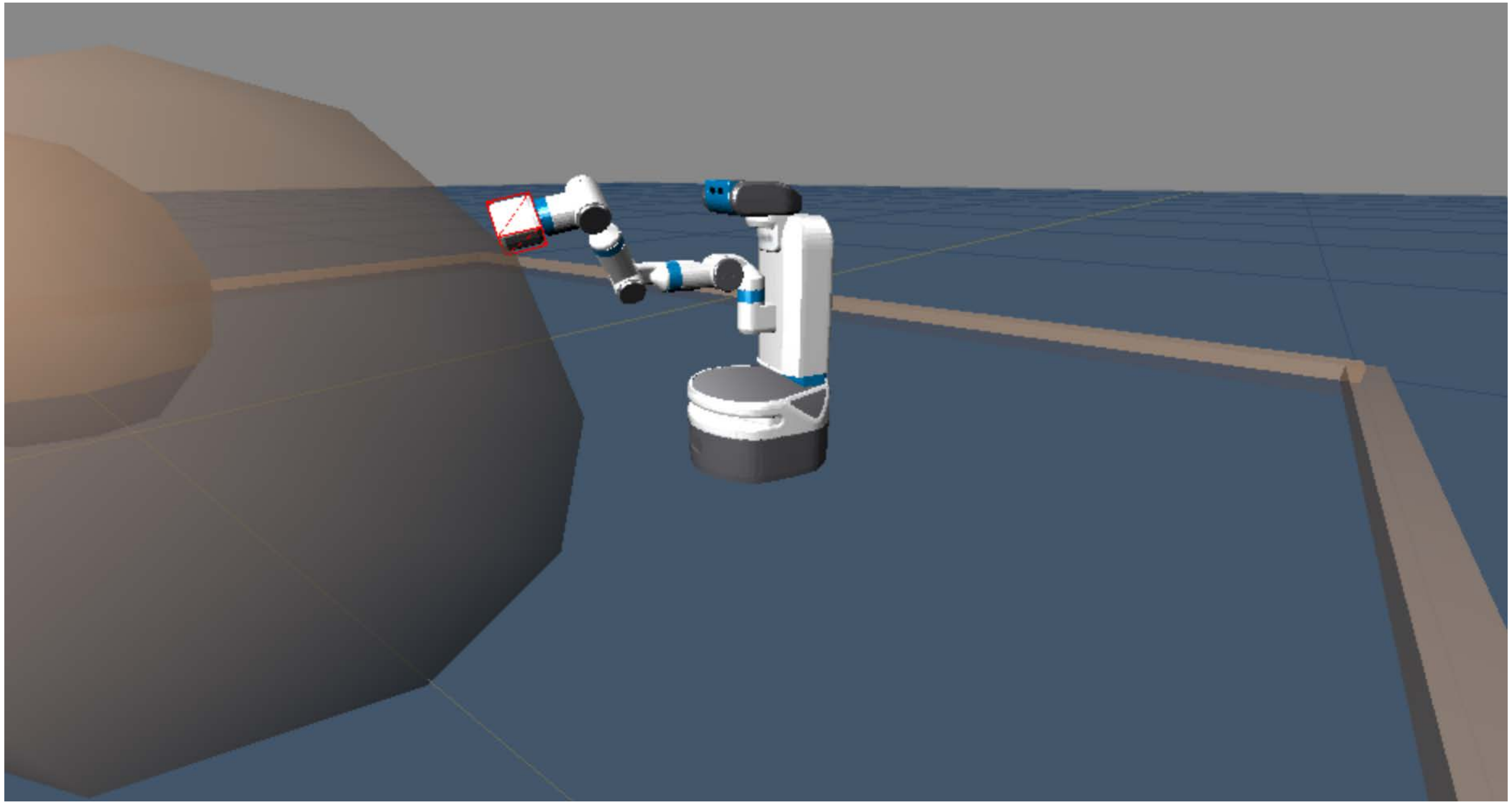
Oriented Bounding Box
(OBB)



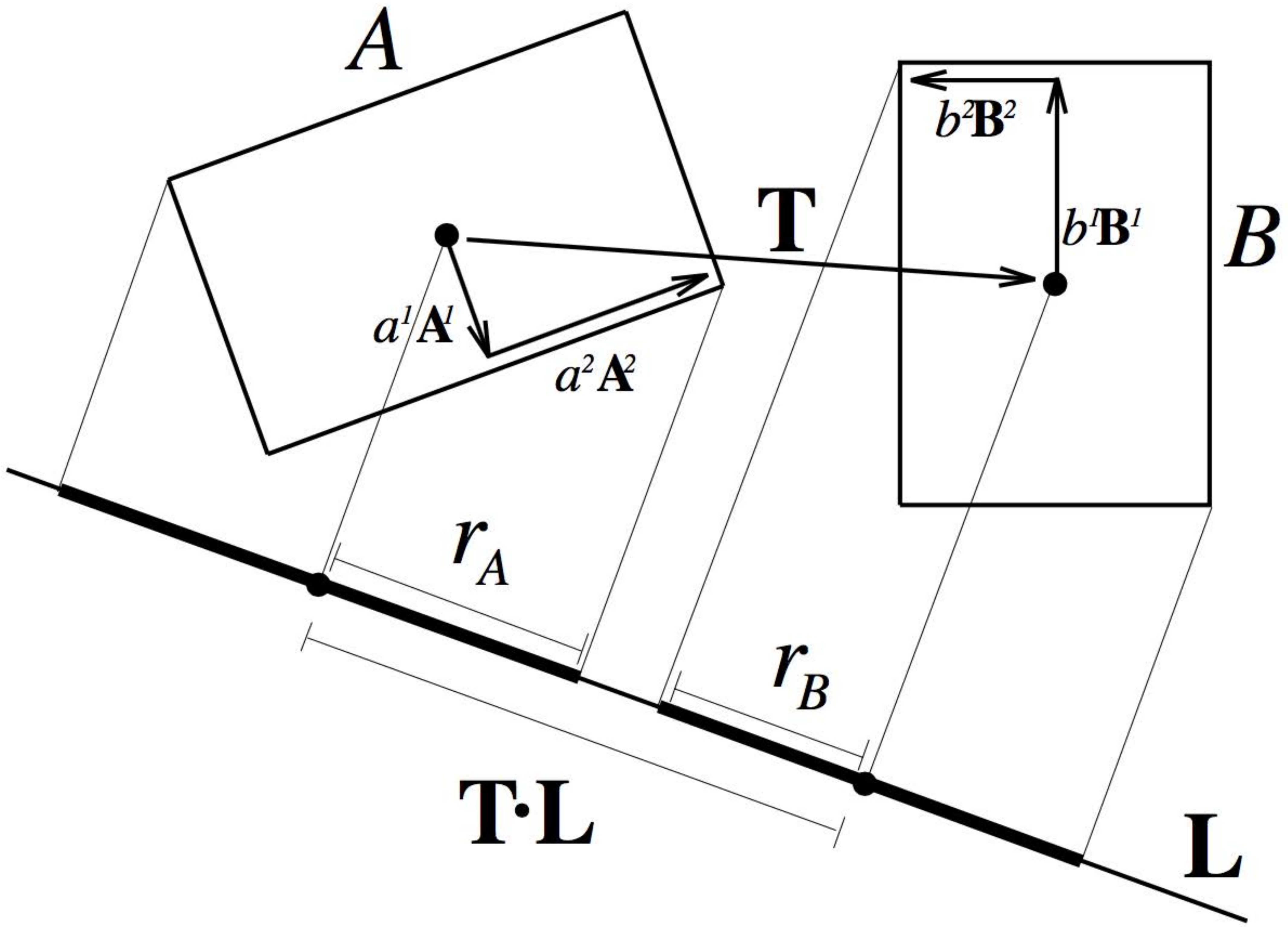
world frame

Gottschalk et al. 1996

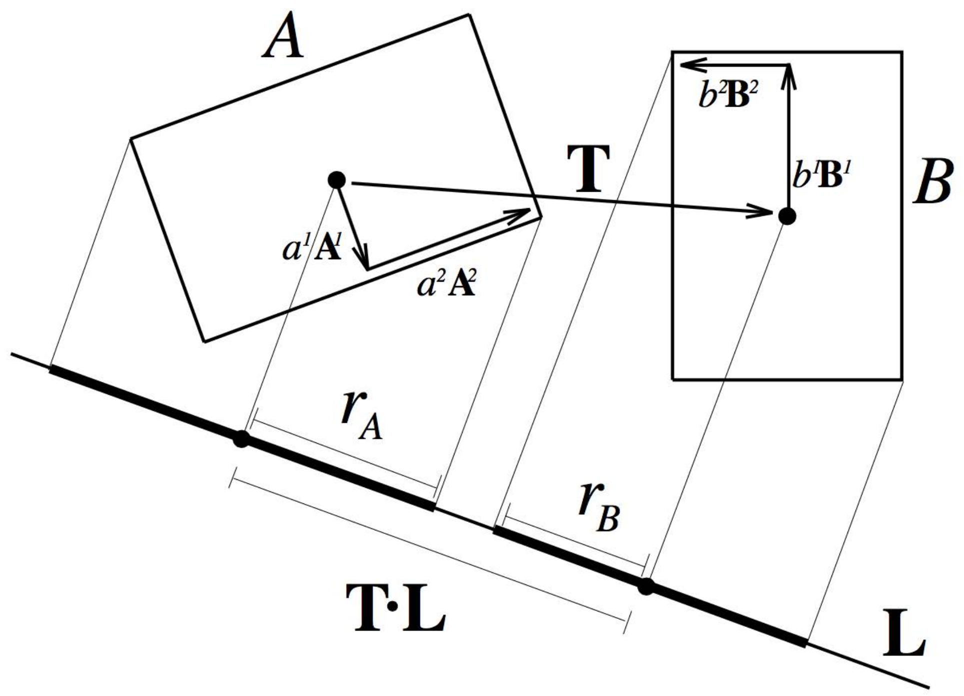
Only a “separating axis” needs to be found



Separating Axis Theorem



Hyperplane separation theorem



From Wikipedia, the free encyclopedia
(Redirected from Separating axis theorem)

In geometry, the **hyperplane separation theorem** is a theorem about disjoint convex sets in *n*-dimensional Euclidean space. There are several rather similar versions. In one version of the theorem, if both these sets are closed and at least one of them is compact, then there is a hyperplane in between them and even two parallel hyperplanes in between them separated by a gap. In another version, if both disjoint convex sets are open, then there is a hyperplane in between them, but not necessarily any gap. An axis which is orthogonal to a separating hyperplane is a **separating axis**, because the orthogonal projections of the convex bodies onto the axis are disjoint.

The hyperplane separation theorem is due to Hermann Minkowski. The Hahn–Banach separation theorem generalizes the result to topological vector spaces.

A related result is the supporting hyperplane theorem.

In geometry, a **maximum-margin hyperplane** is a hyperplane which separates two 'clouds' of points and is at equal distance from the two. The margin between the hyperplane and the clouds is maximal. See the article on Support Vector Machines for more details.

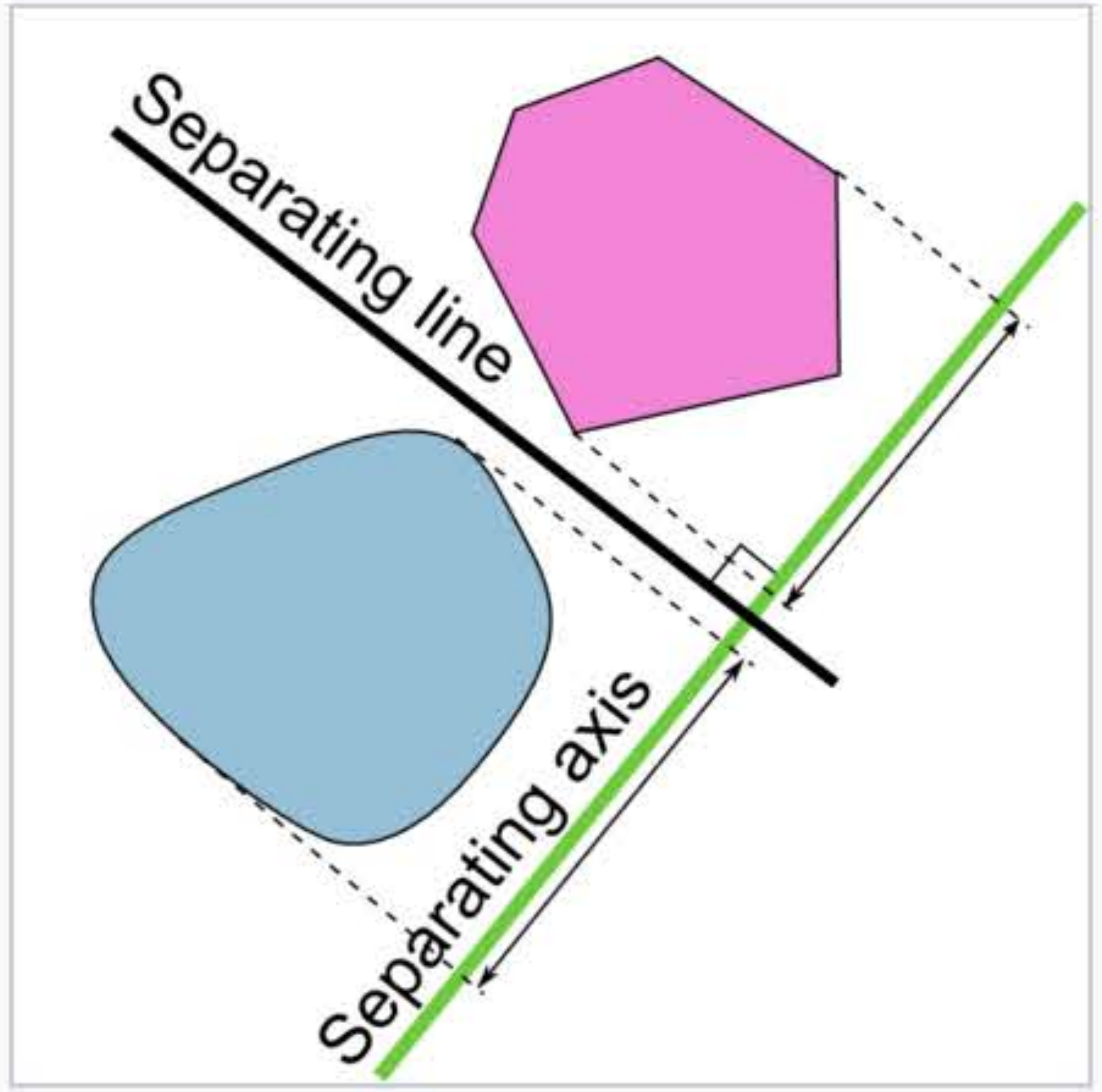
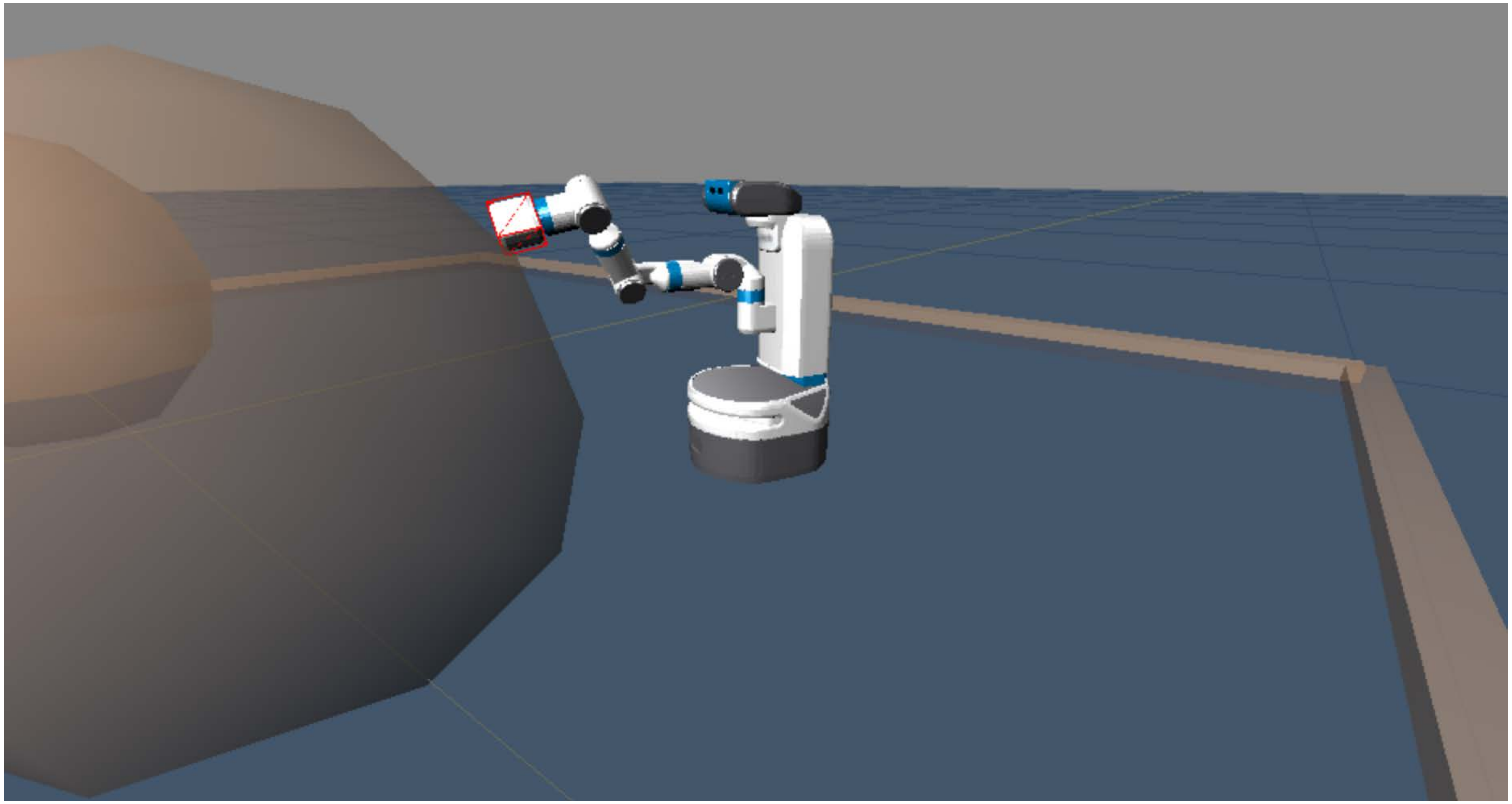


Illustration of the hyperplane separation theorem.

Consider AABB link tested against spherical obstacles in link frame

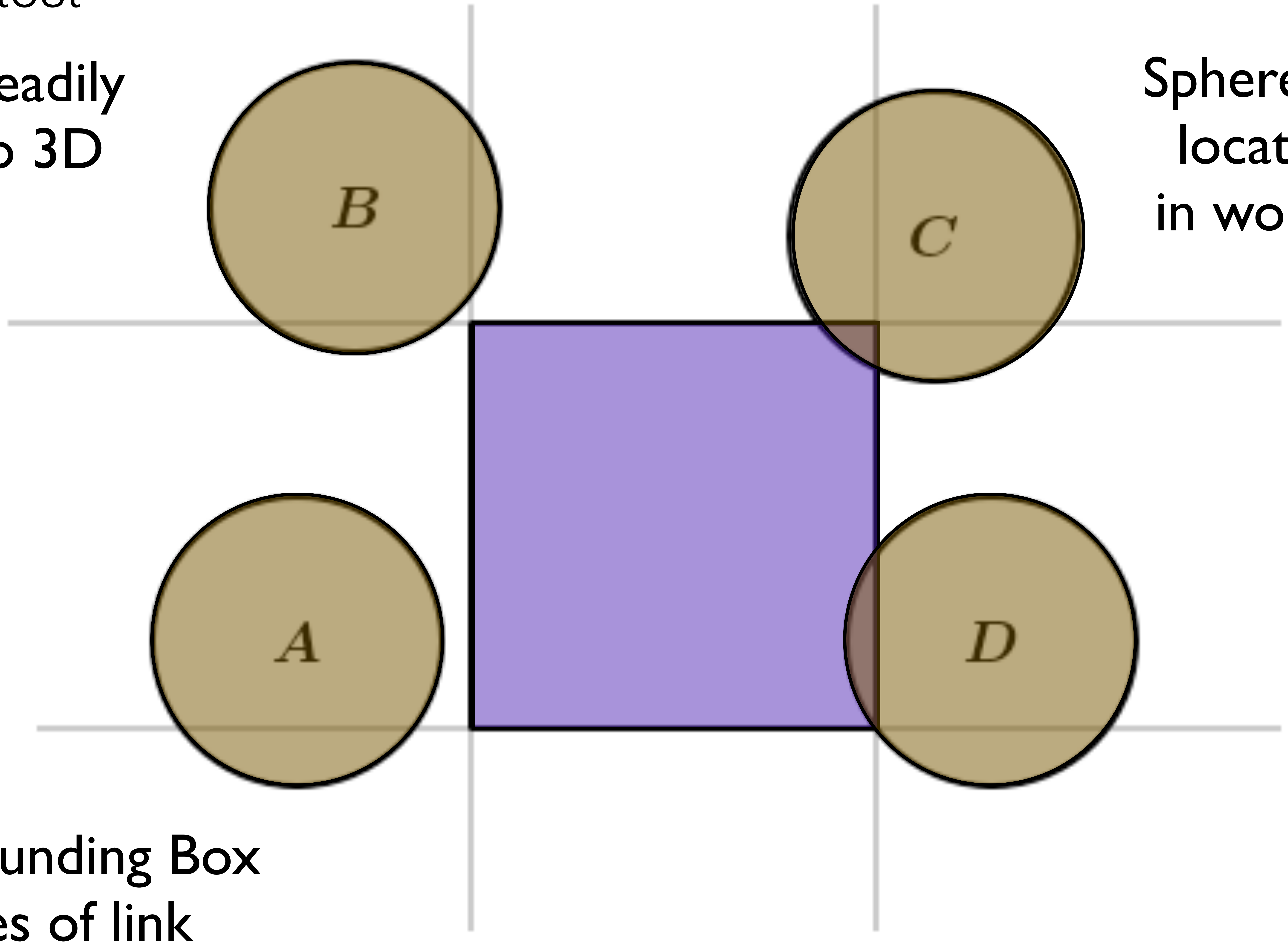


Sphere-bbox test

`robot_obstacles[i]`

2D example readily generalizes to 3D

Sphere obstacles with location and radius in world coordinates



Axis Aligned Bounding Box in coordinates of link

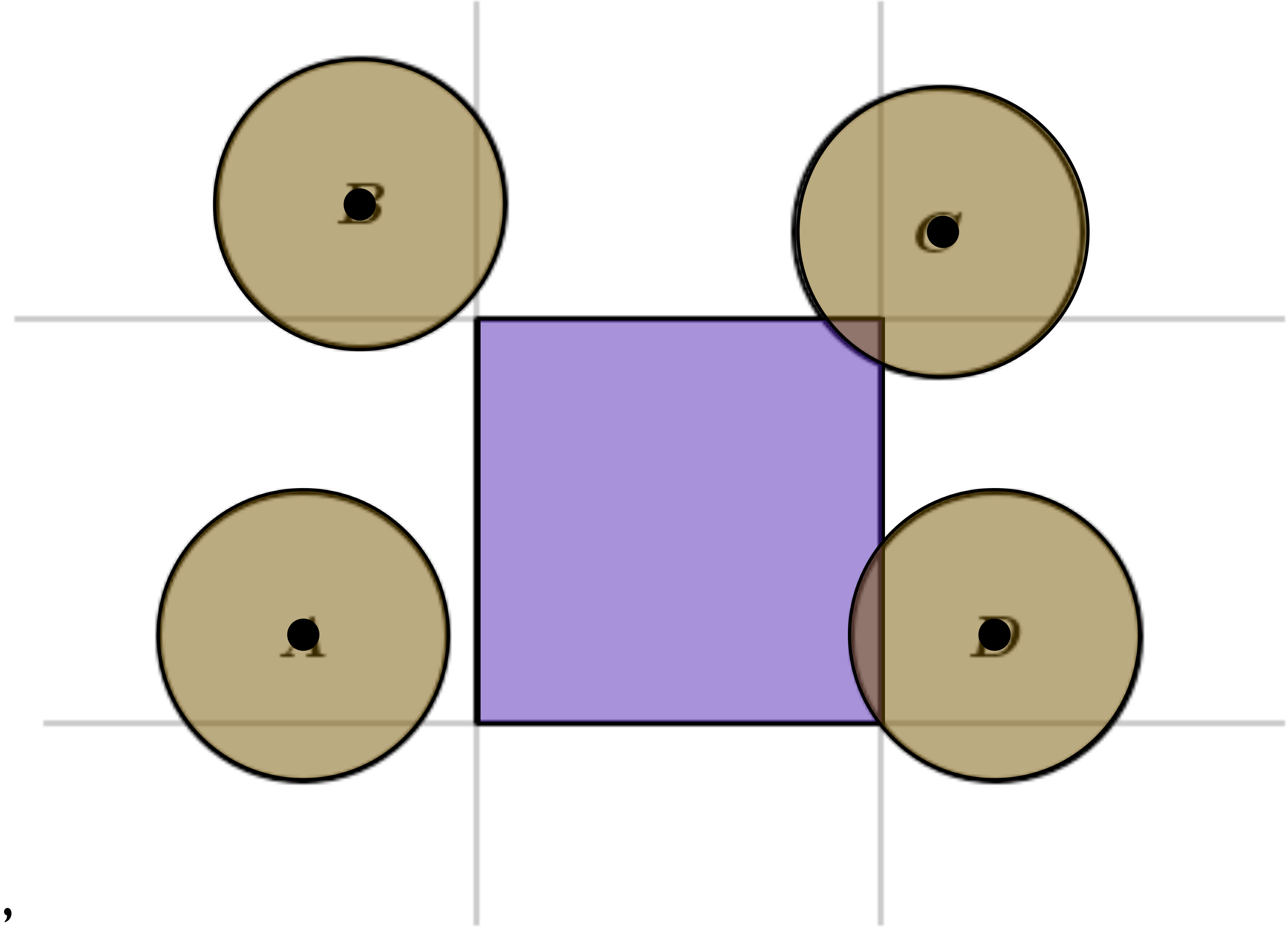
`robot.links[x].bbox = [[x_min,y_min,z_min], [x_max,y_max,z_max]]`

Sphere-bbox test

If sphere separable from AABB in any dimension, return no collision

`loc_y-radius>y_max?`

`loc_y+radius<y_min?`

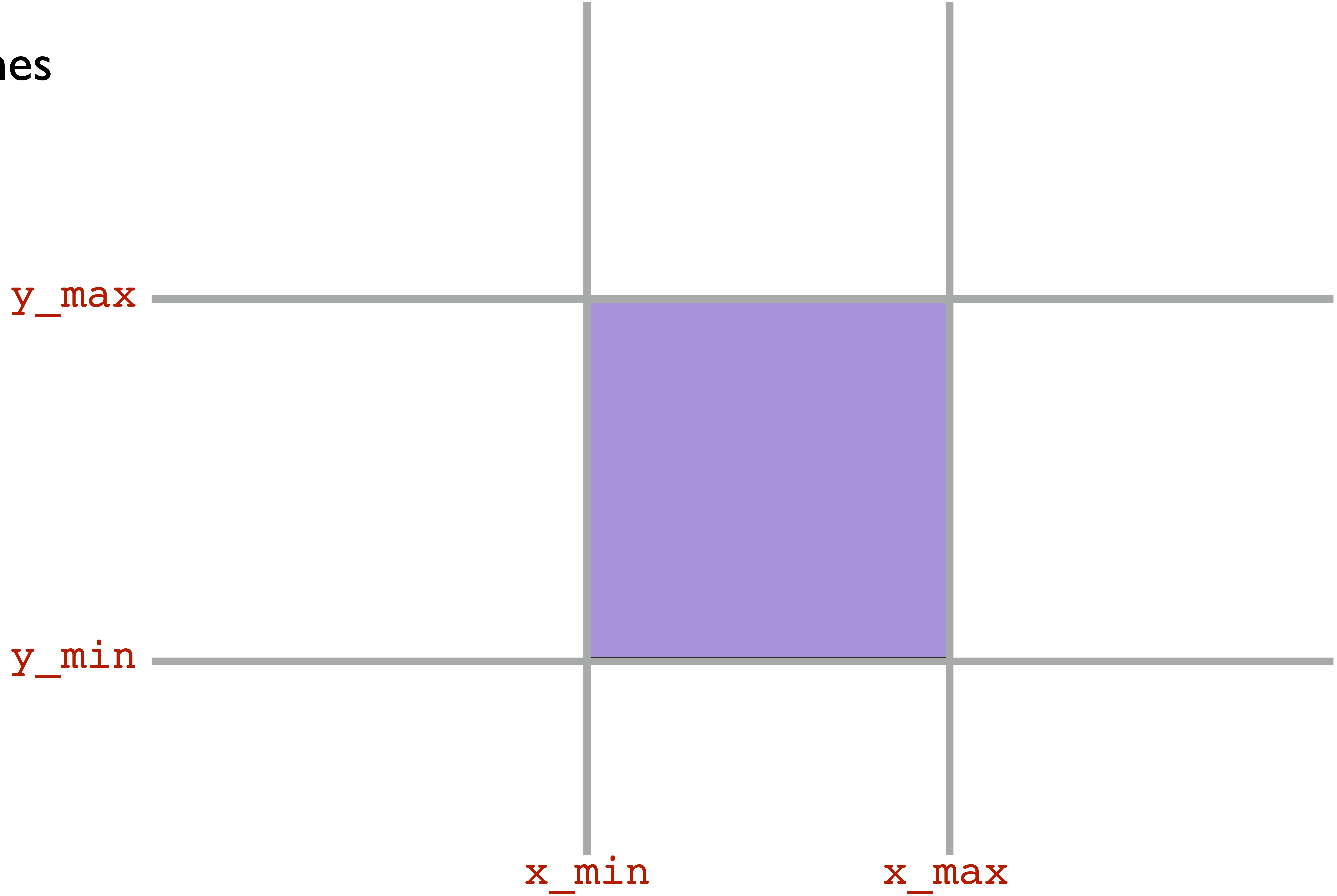


`loc_x+radius<x_min?`

`loc_x-radius>x_max?`

If sphere collides on all tests, return collision

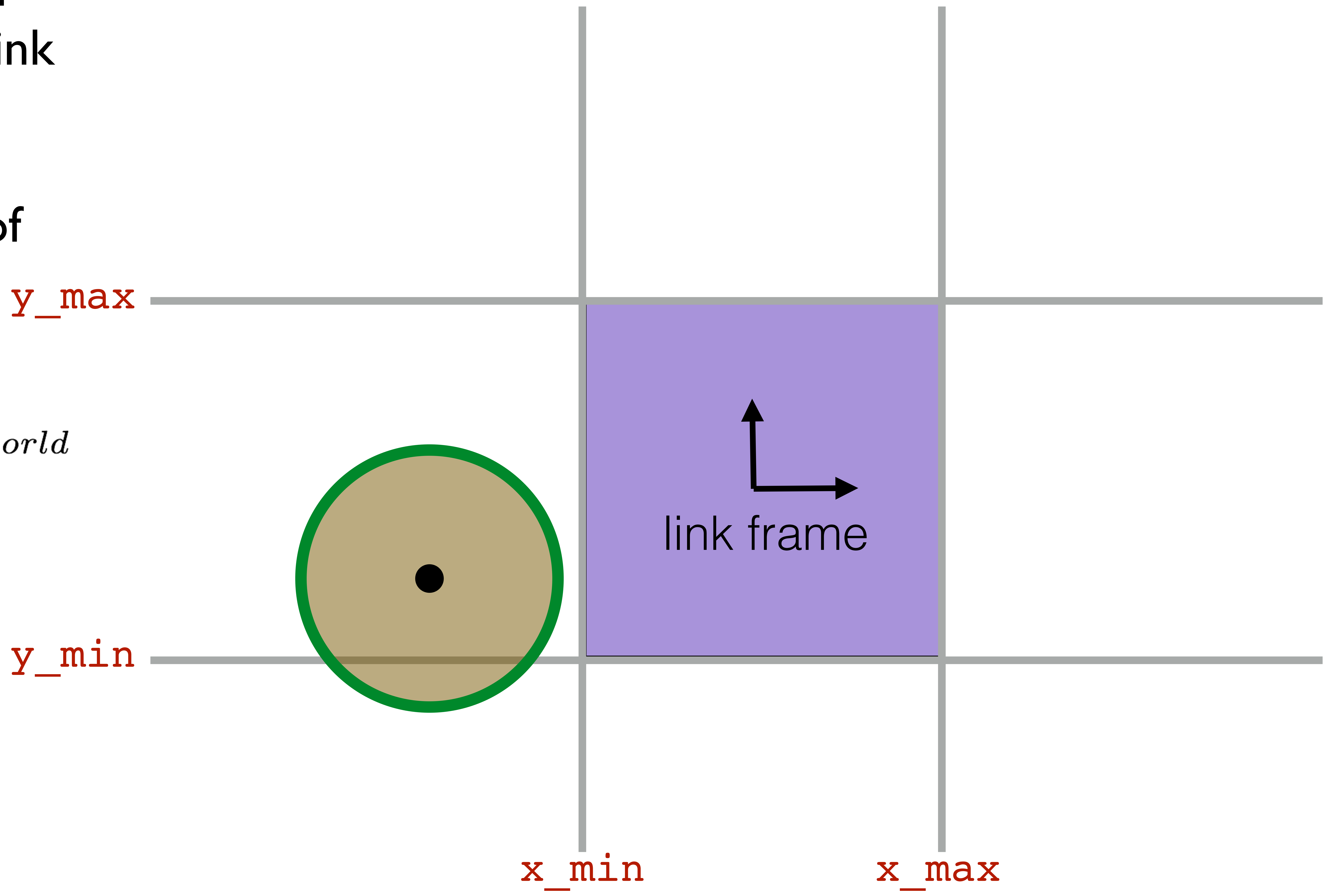
Separating planes

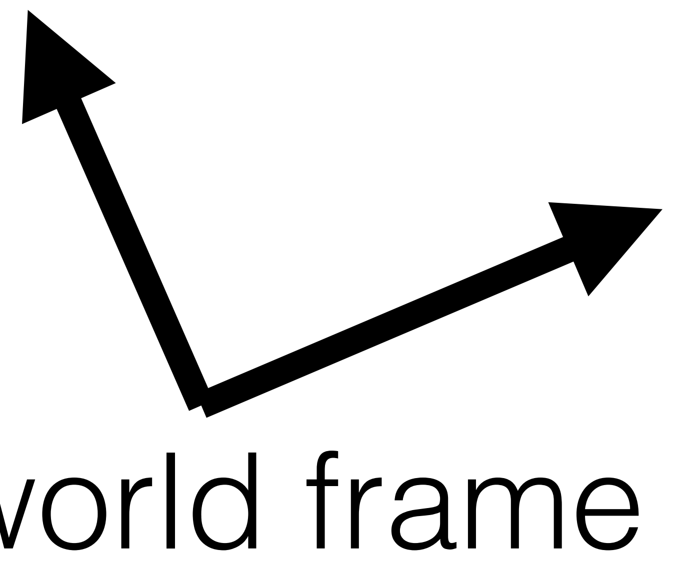


Transform centers of sphere obstacles into link coordinates

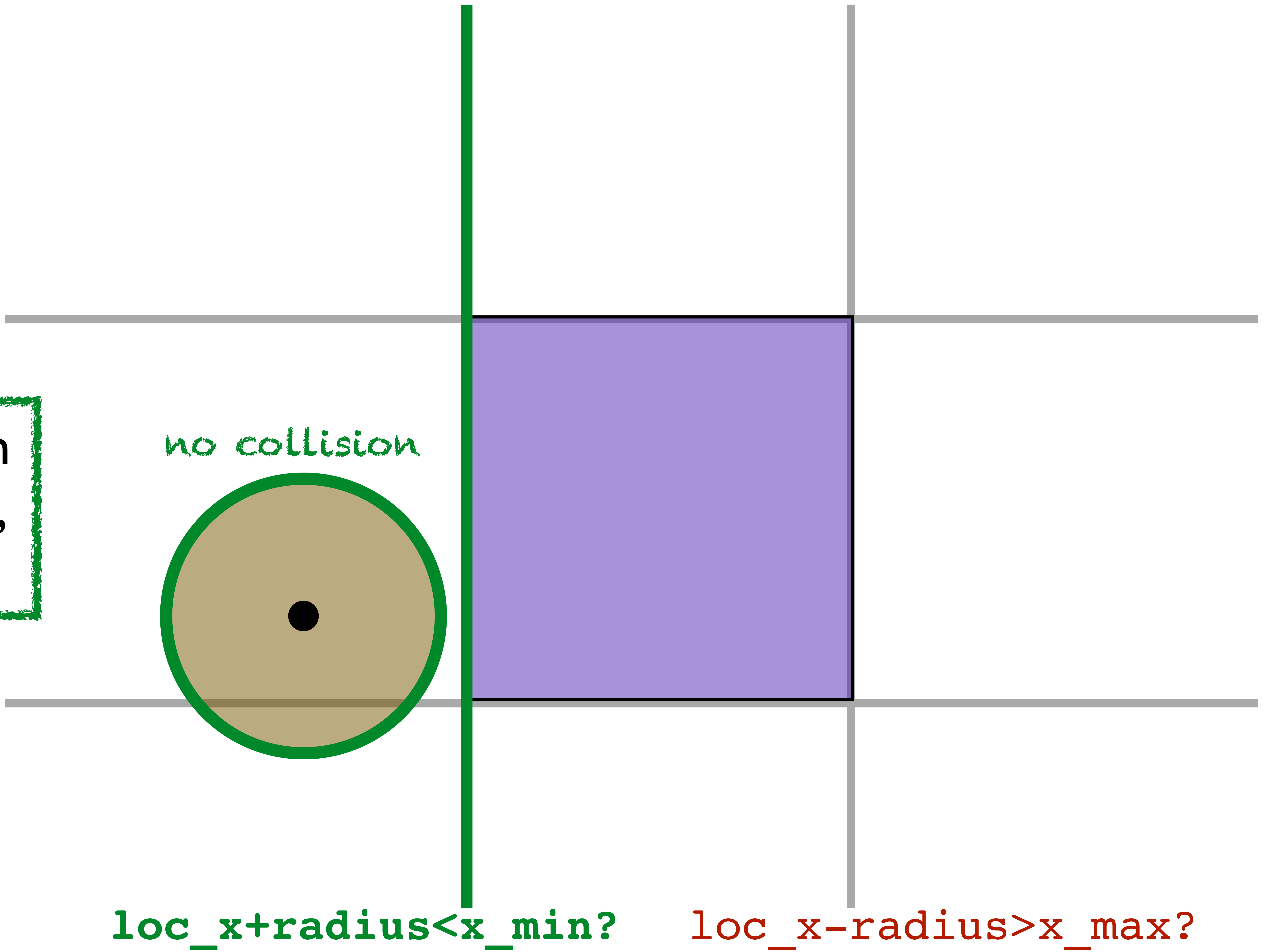
(Remember inverse of homogeneous transform?)

$$p^{link} = (T^{world}_{link})^{-1} p^{world}$$

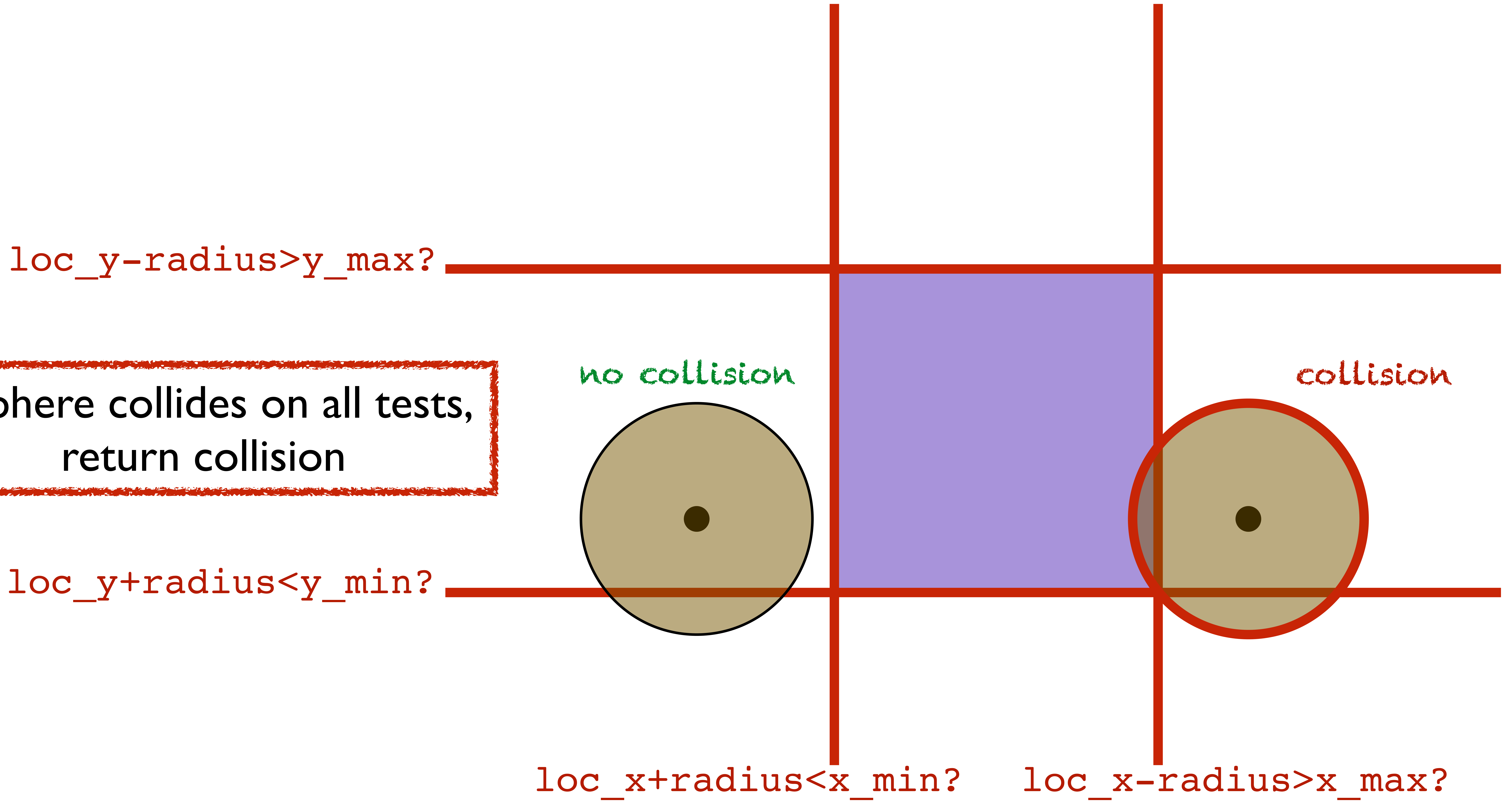


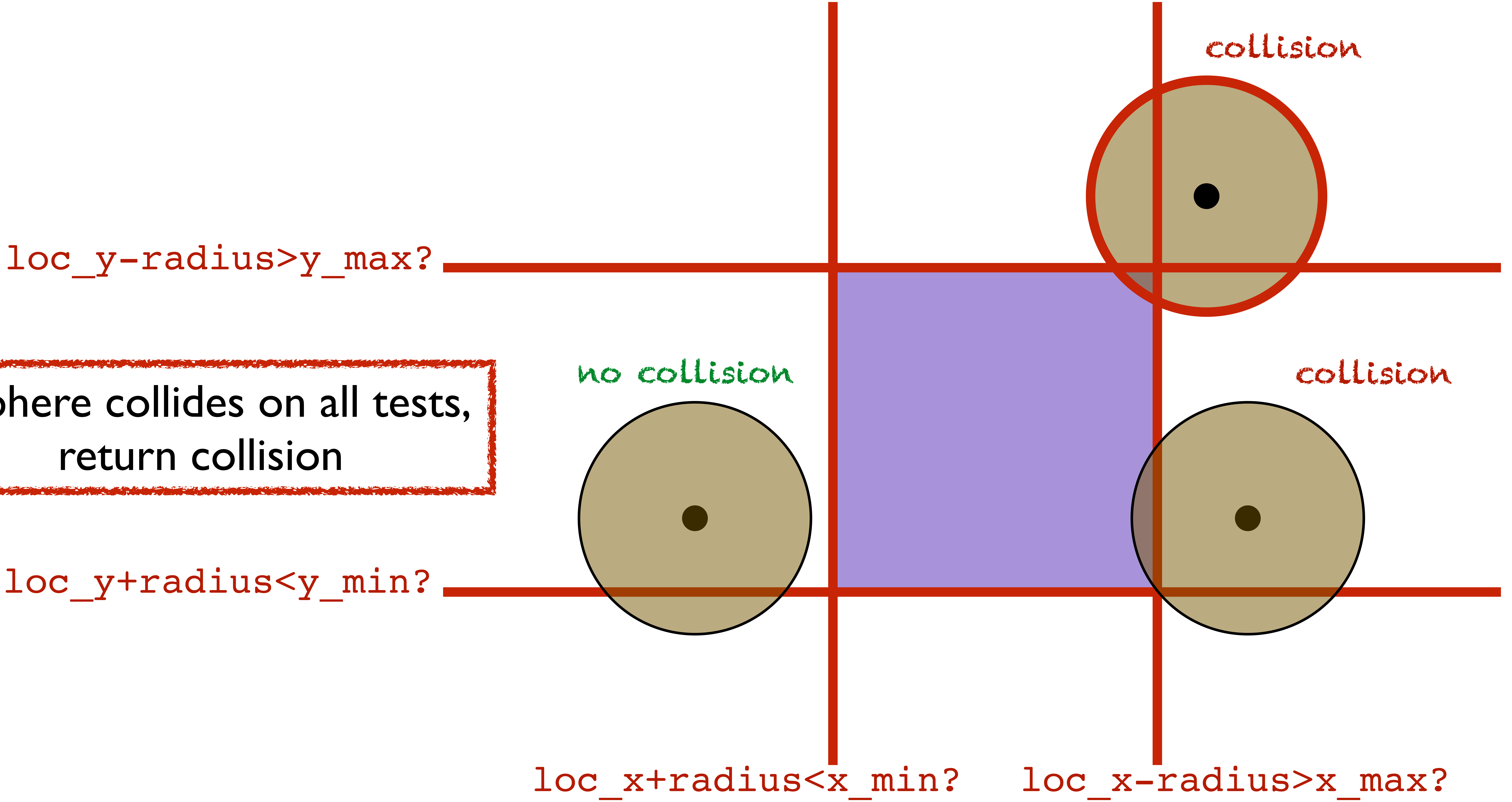


If sphere separable from AABB in any dimension, return no collision



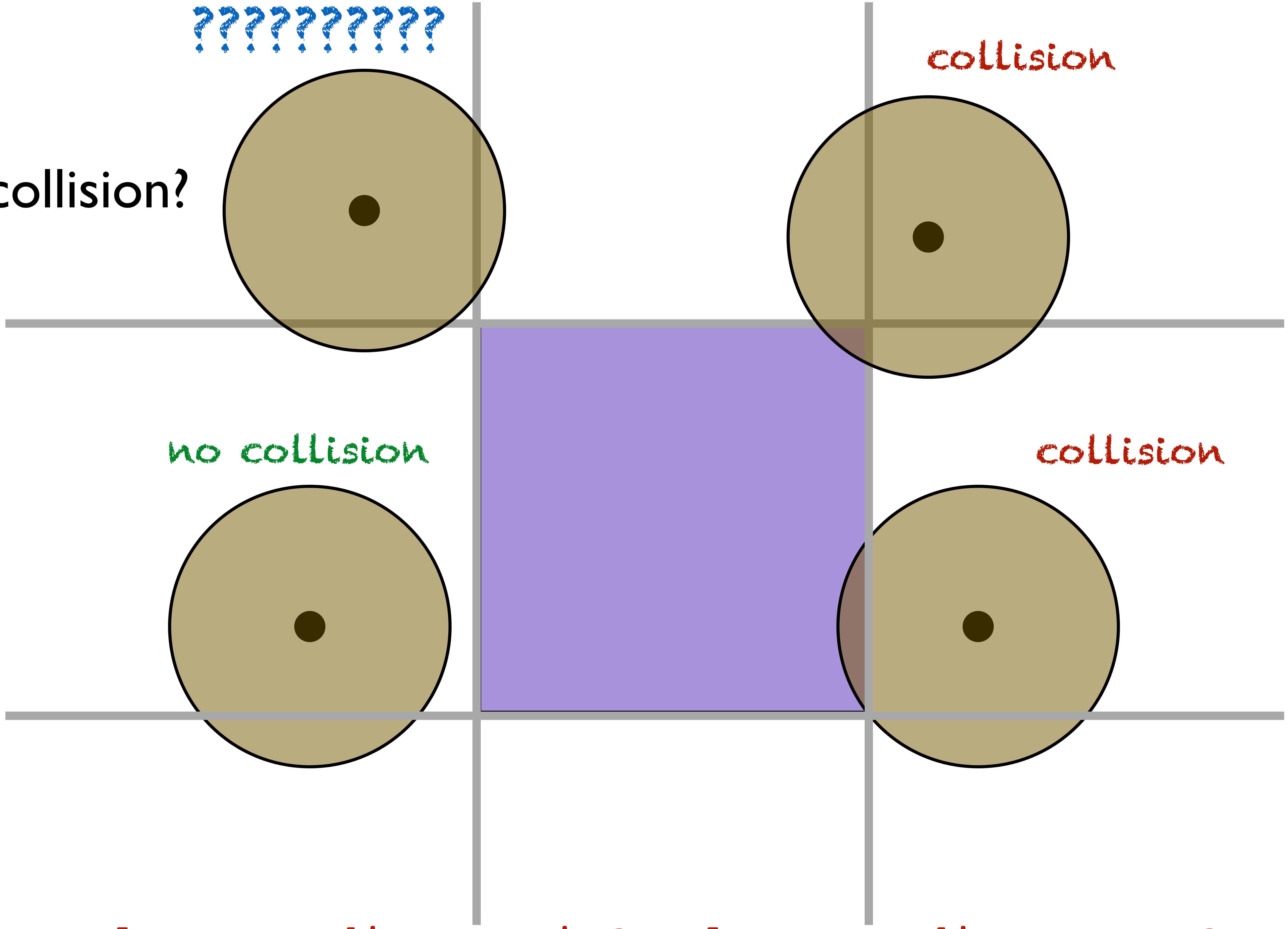
If sphere collides on all tests, return collision





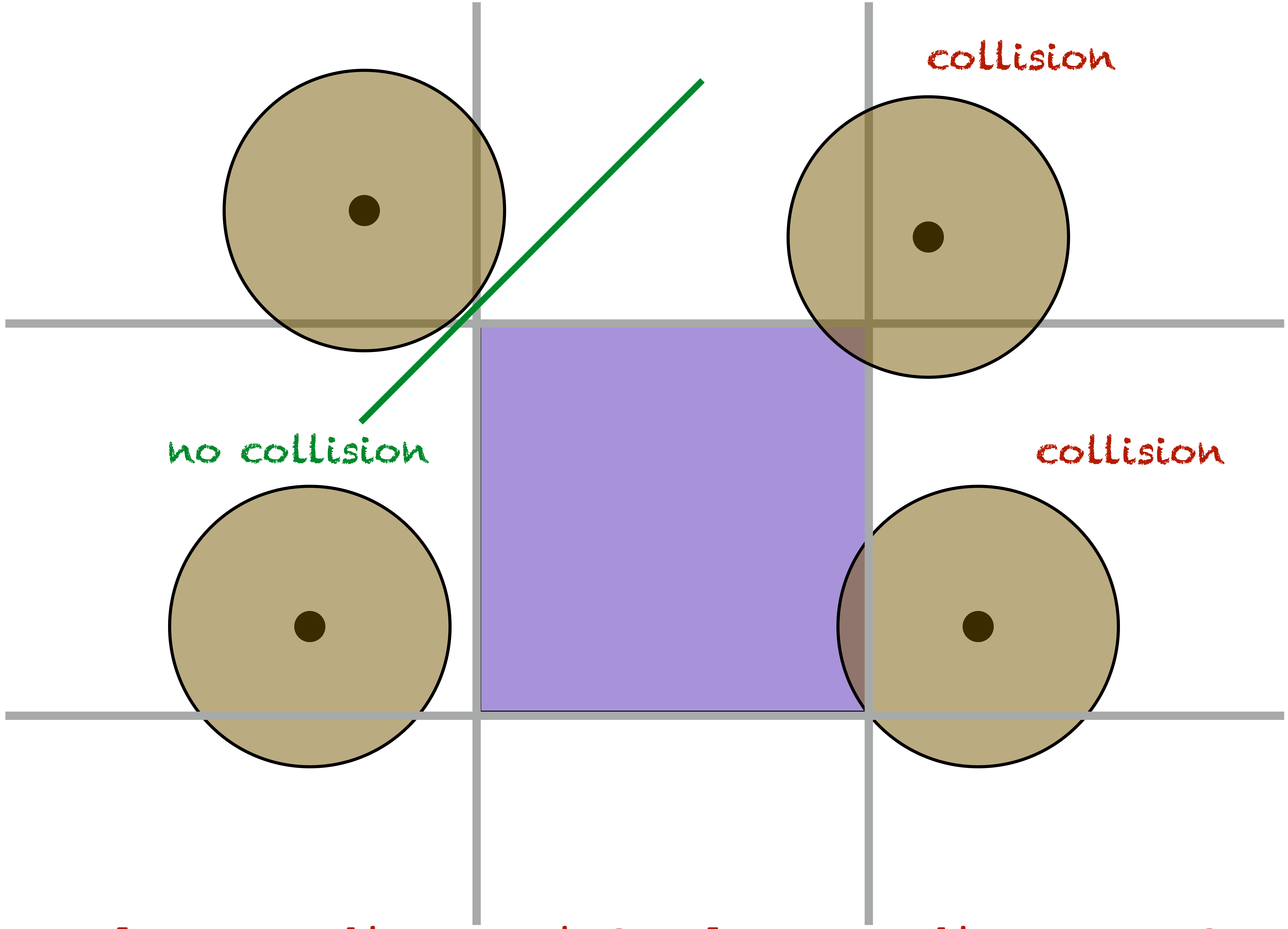
If sphere collides on all tests, return collision

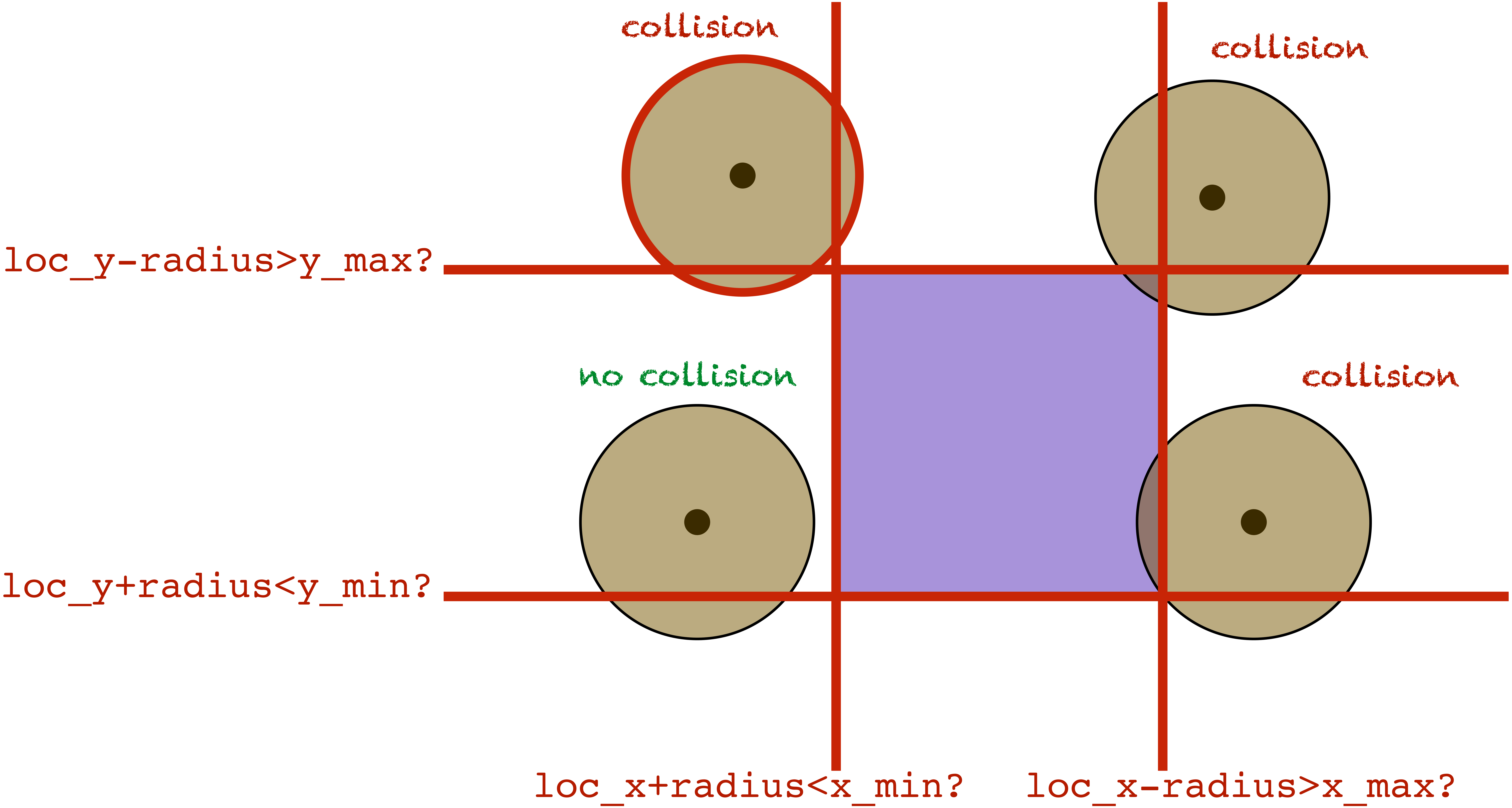
Is this obstacle in collision?

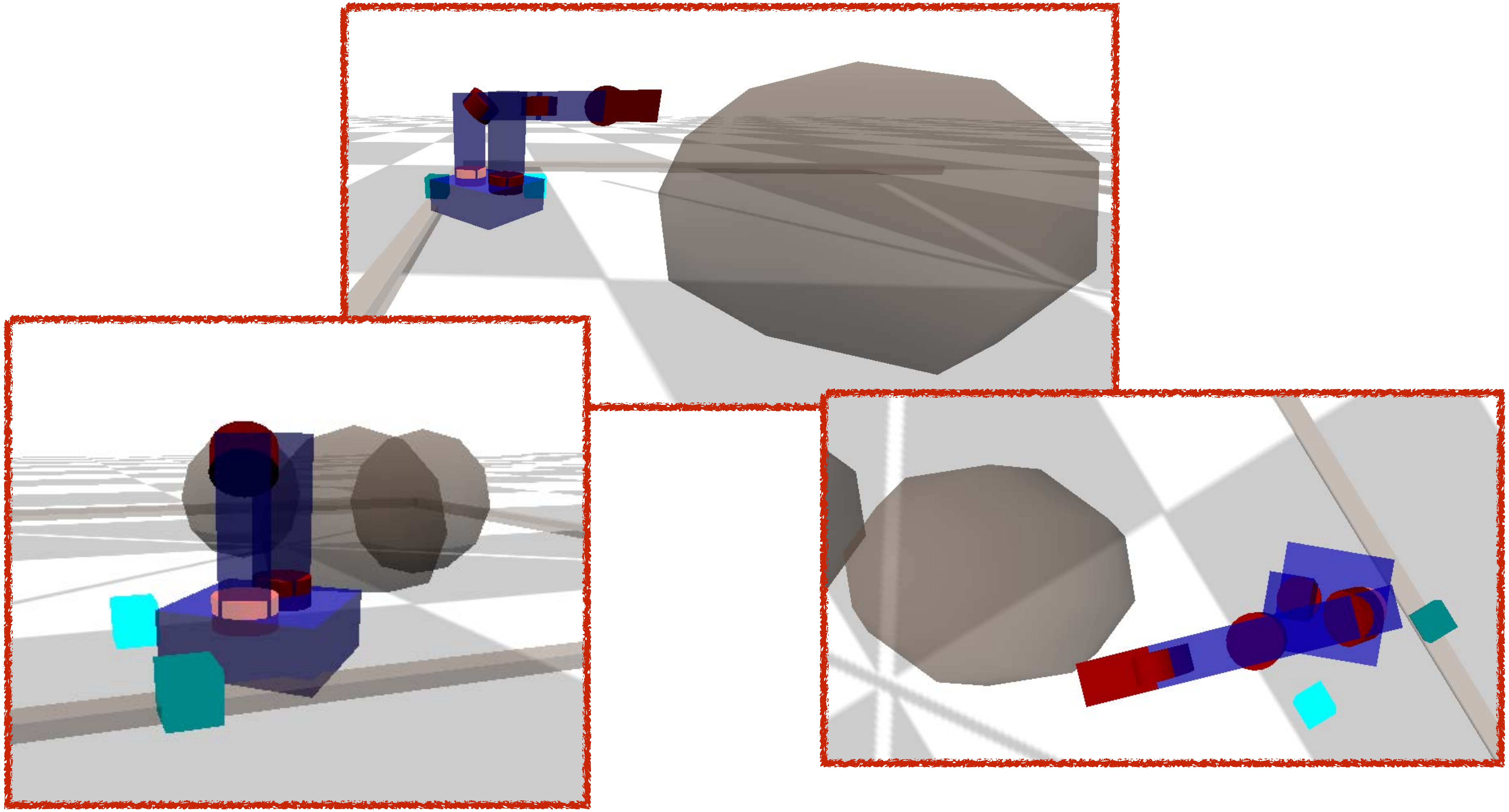


Slide borrowed from Michigan Robotics autorob.org

True separating axis not tested



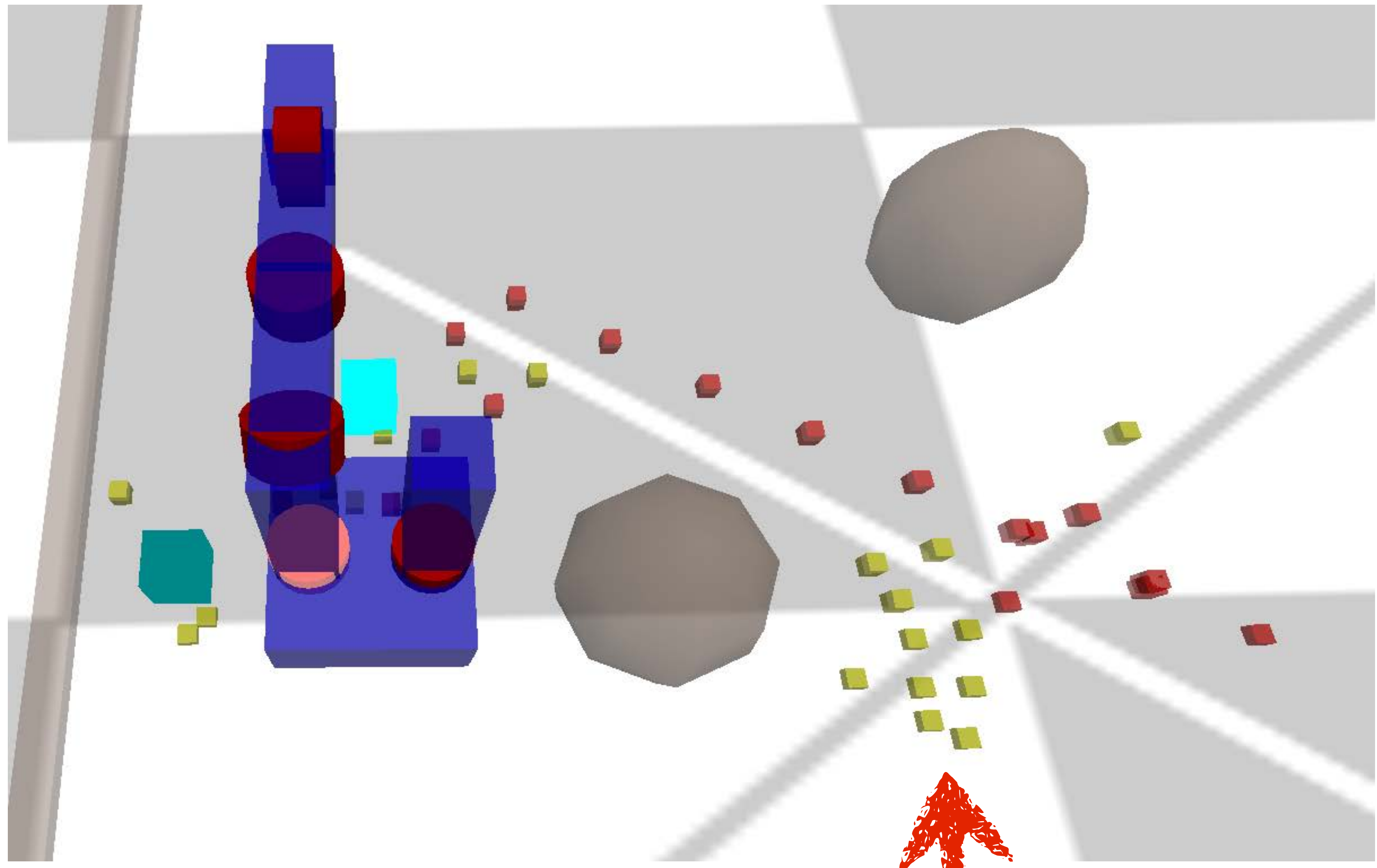

collision
collision
loc_y-radius>y_max?
no collision
collision
loc_y+radius<y_min?
loc_x+radius<x_min?
loc_x-radius>x_max?



Last notes about planning visualization

kineval_rrt_connect.js

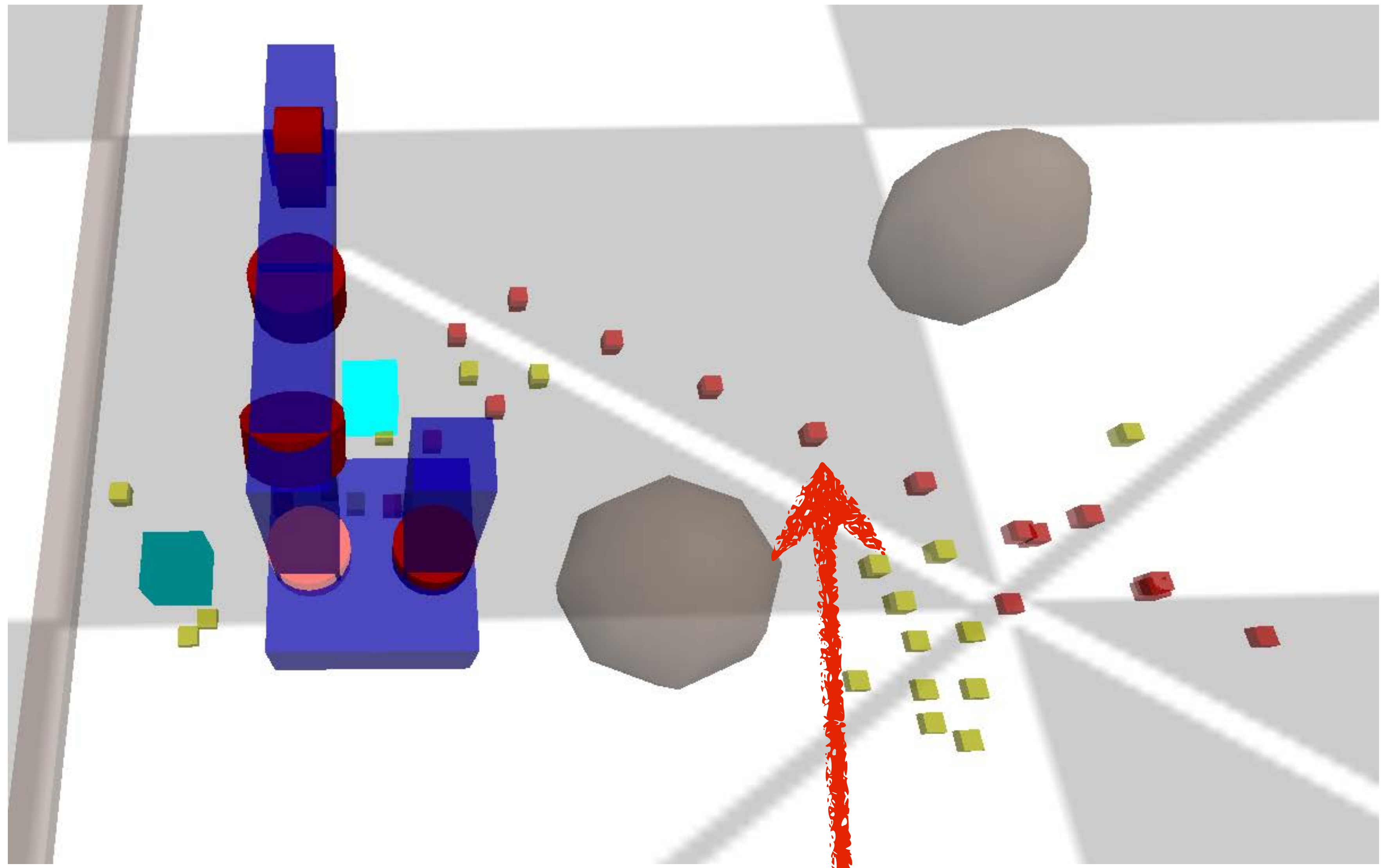
```
function tree_init(q) {

    // create tree object
    var tree = {};

    // initialize with vertex for given configuration
    tree.vertices = [];
    tree.vertices[0] = {};
    tree.vertices[0].vertex = q;
    tree.vertices[0].edges = [];

    // create rendering geometry for base location of vertex configuration
    add_config_origin_indicator_geom(tree.vertices[0]);

    // maintain index of newest vertex added to tree
    tree.newest = 0;

    return tree;
}
```



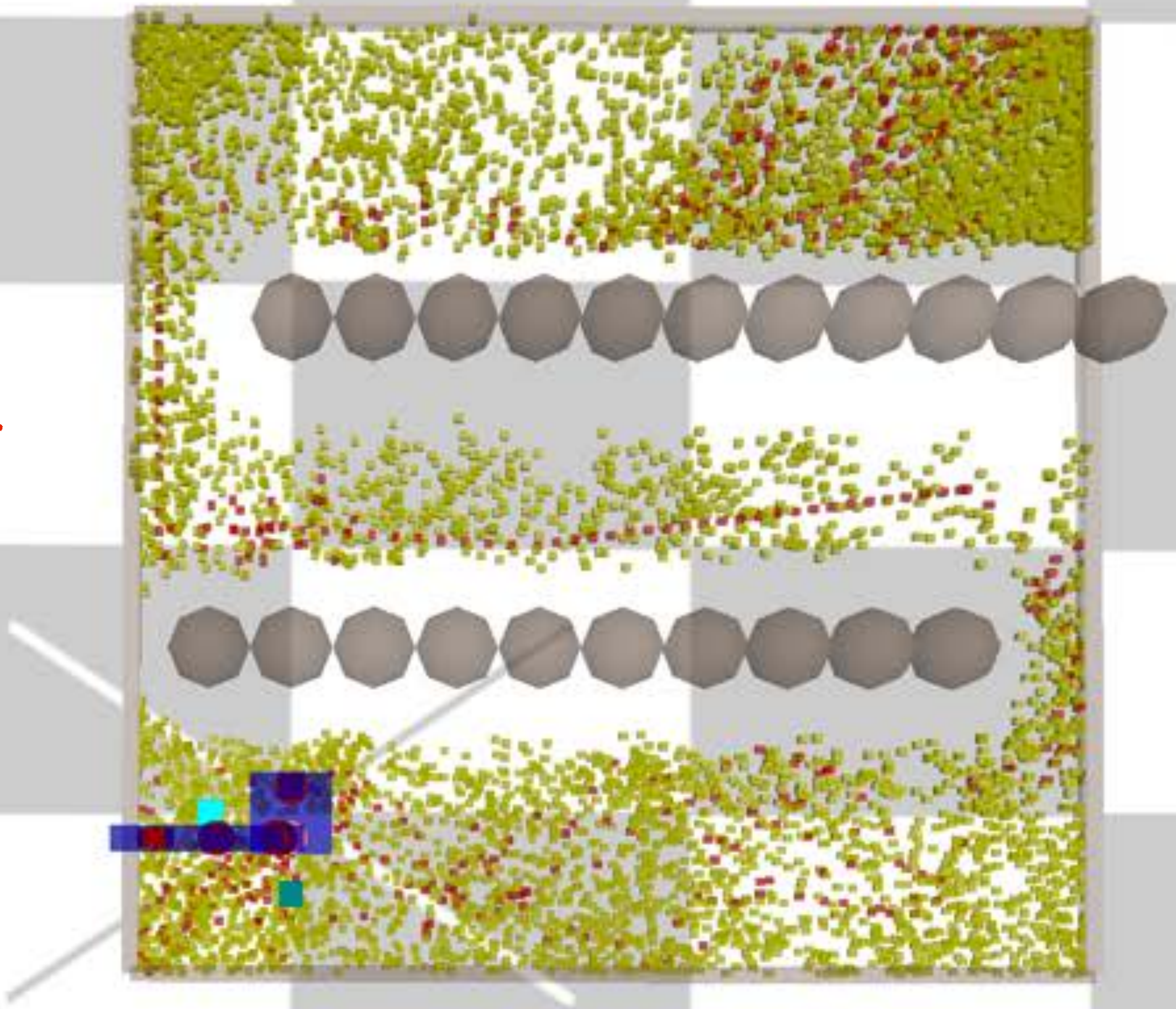
creates "geom" property of tree vertex with cube at base location for explored tree configuration

kineval_rrt_connect.js

```
for (i=0;i<robot_path.length;i++) {
     robot_path[i].geom.material.color = {r:1,g:0,b:0};
 }
```



found motion path highlighted in red with this code make sure to test against all provided worlds!

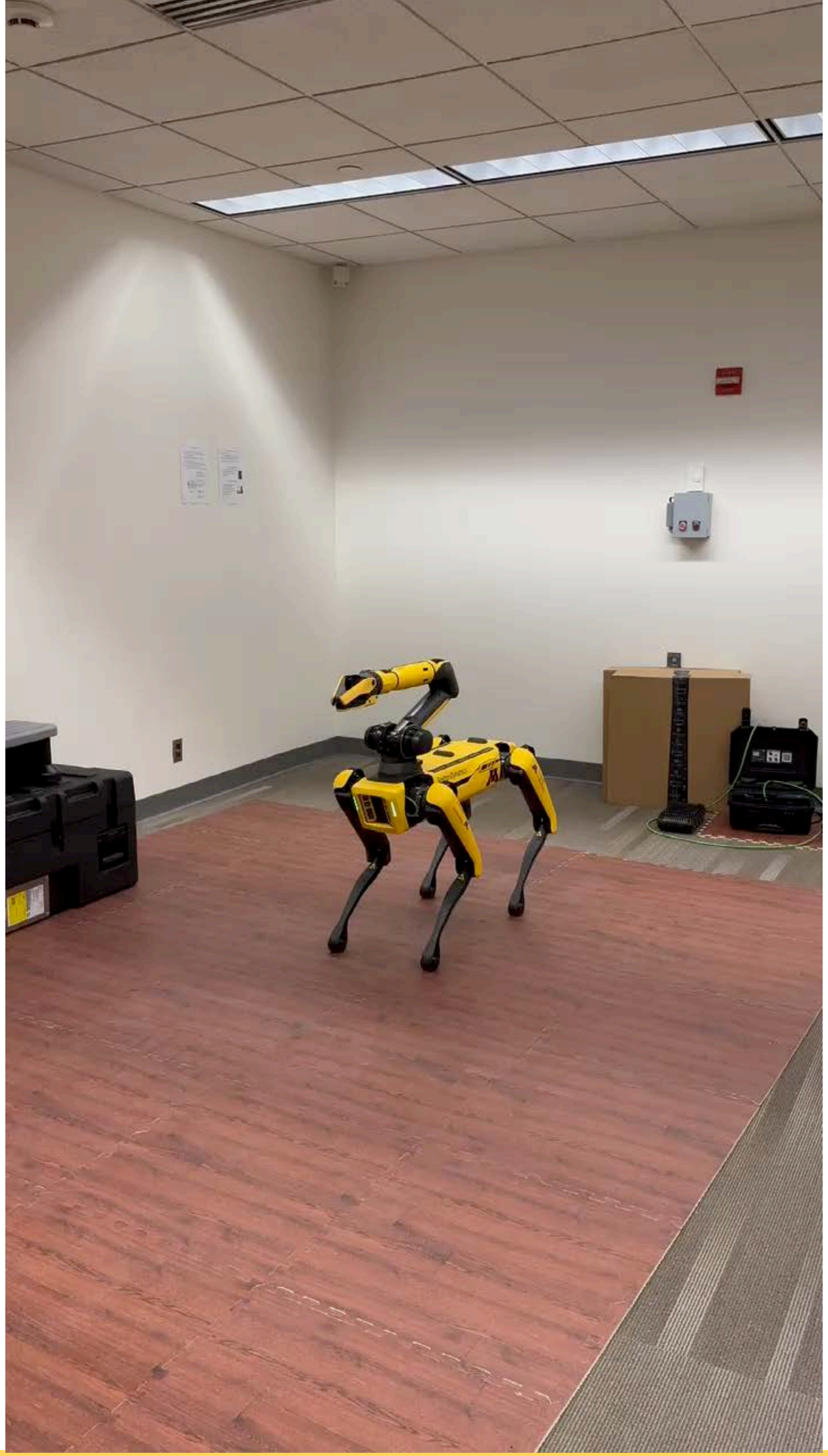


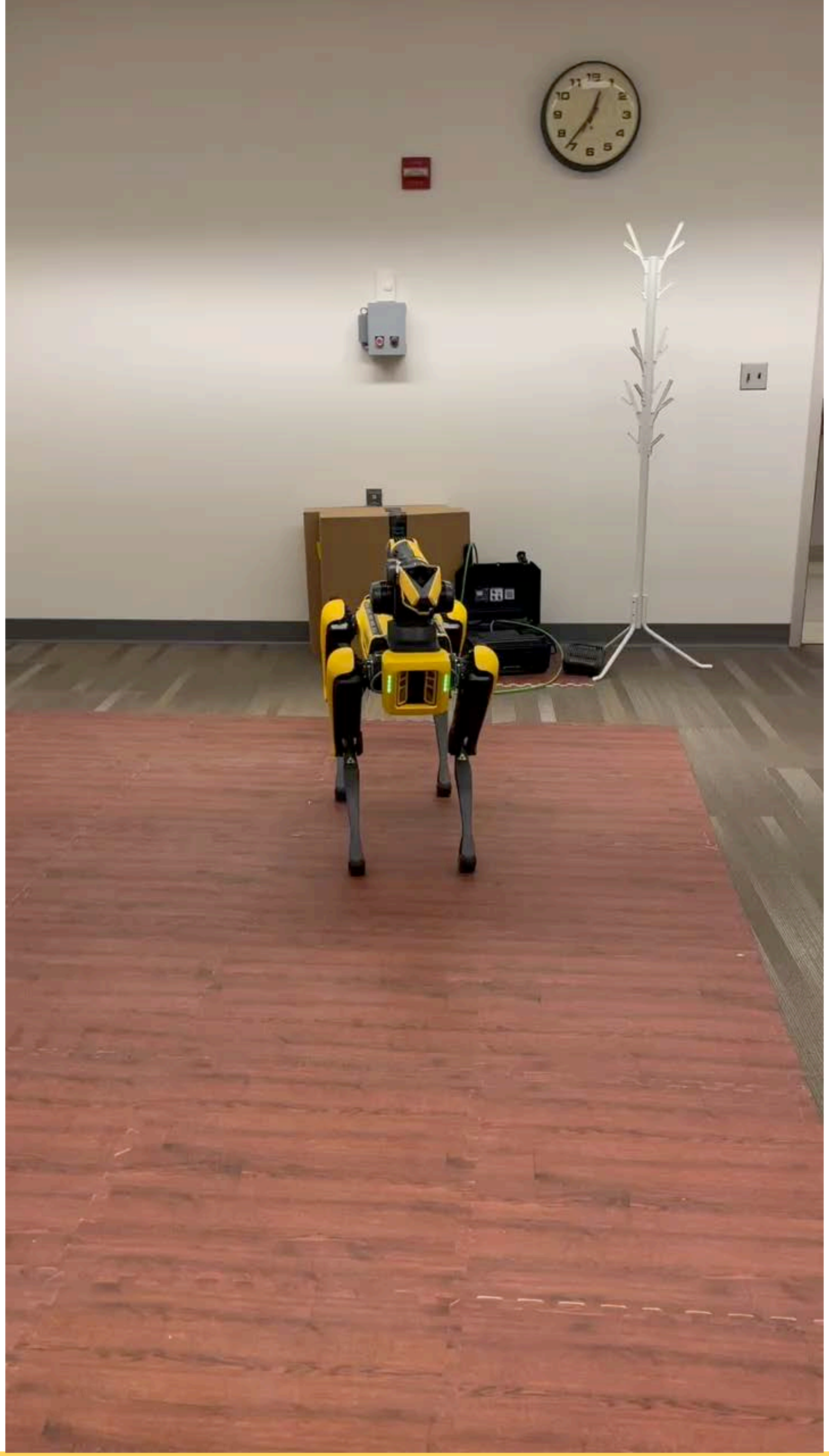
make sure to test against all provided worlds!

Thanks to all the robot dance videos

Brown University - https://www.youtube.com/watch?v=8VNlgN58hbg

Hachi Dances Too!





Next Lecture

Planning - VI - Potential Fields