Collision Detection Based on Fuzzy Scene Subdivision

David Mainzer[1] and Gabriel Zachmann[2]

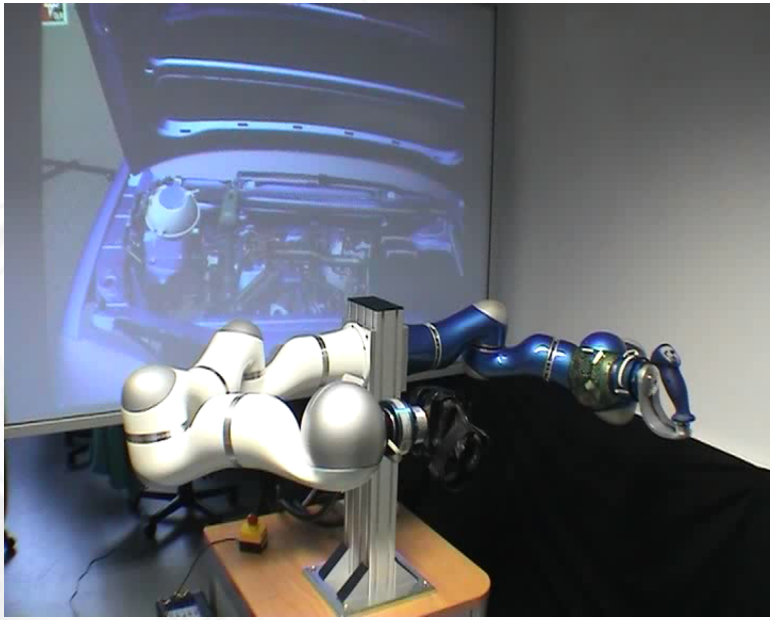
[1] Clausthal University, Germany

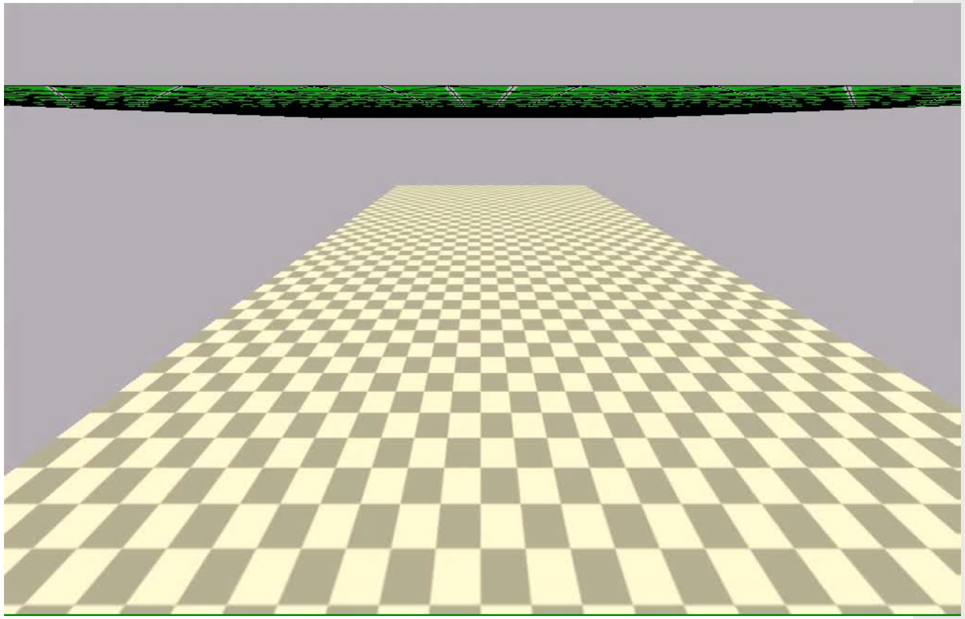
[2] University of Bremen, Germany

Symposium on GPU Computing and Application

Oct 2013, Singapore

Motivation for Collision Detection



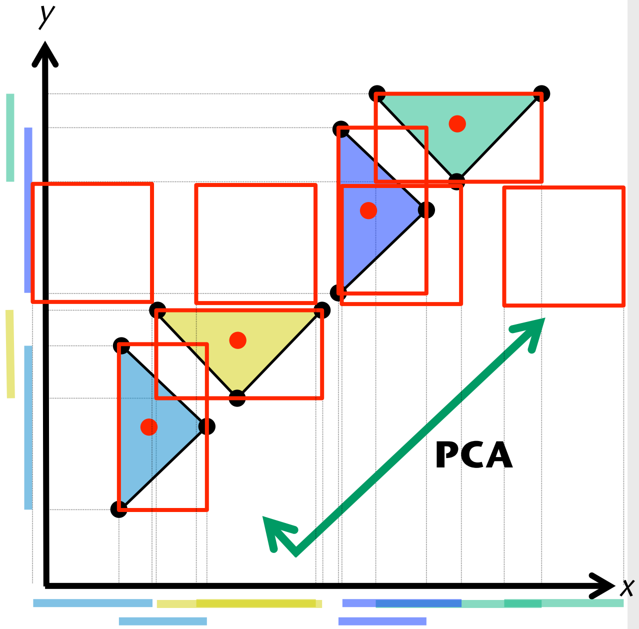


- Make virtual environments realistic
- Basic component of video games, robotics, medical applications, etc.
- Collision detection is bottleneck in many applications

Previous Work

- Algorithm for proximity queries between a closed rigid object and an arbitrary mesh [Morvan 2008]
  - Sampled distance field of rigid object over arbitrary mesh
  - Drawback: one object has to be a rigid body
- Hybrid CPU/GPU technique based on spatial subdivision [Pabst 2010]
  - Prune away non-colliding parts of scene by using an adapted spatial subdivision method
- GPU-based streaming algorithm for collision detection [Tang 2011]
  - Use BVH as culling technique
  - Cannot be easily extended to use more than one GPU

PCA-based Sweep-Plane Technique

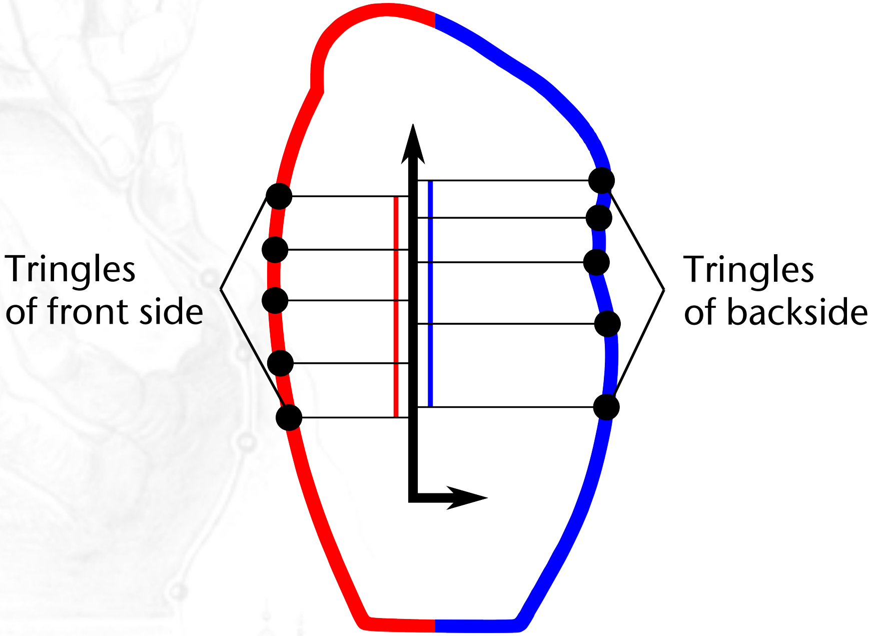


- 1D sweep-plane approach
  - BBox spans an interval on axis
- Sorting intervals
  - Identify possible colliding BBoxes
- Minimize potentially colliding BBoxes
- Best sweep direction separate primitives as much as possible
- Best sweep direction:
  - Compute the Principal Component Analysis **(PCA)**

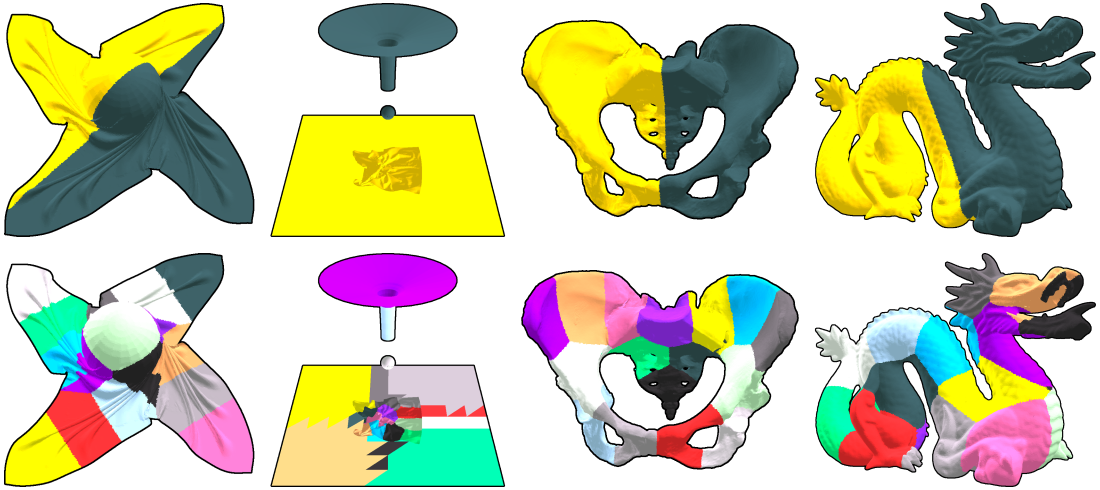
Scene Subdivision Using Fuzzy C-Means

Problem:

- Sweep-plane step → dimensional reduction → false positives
  - Primitives of *front side* and primitives of *backside* identified as potentially colliding pairs



Scene Subdivision Using Fuzzy C-Means



Solution:

- Subdivide scene into components using *clustering* algorithm
- Primitives on border need to be on two (or more) clusters → fuzzy clustering
- Run incrementally → reuse results from previous step → Fuzzy C-Means

Our Algorithm

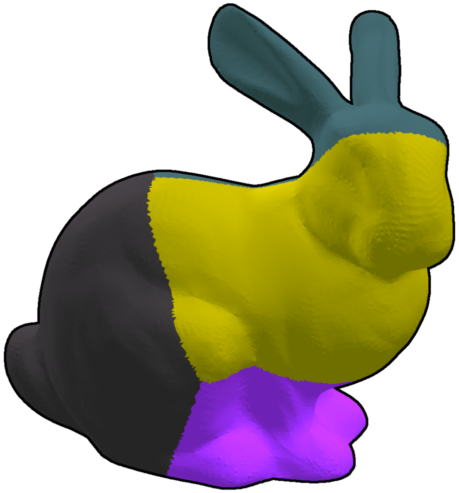
Fuzzy C-means GPU-based Collision Detection

Input: primitives of all objects

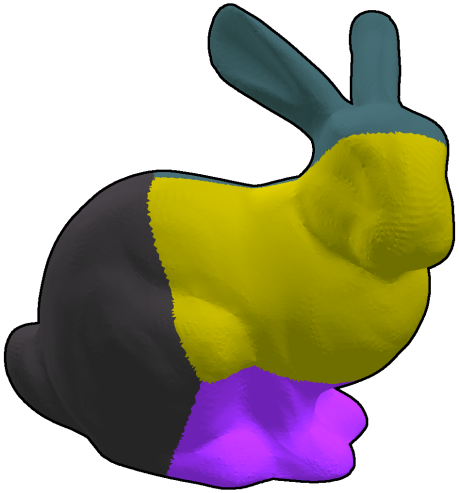
Output: intersecting pairs of primitives

Subdivide scene into c clusters using fuzzy C-means

for all clusters **do in parallel**



Our Algorithm

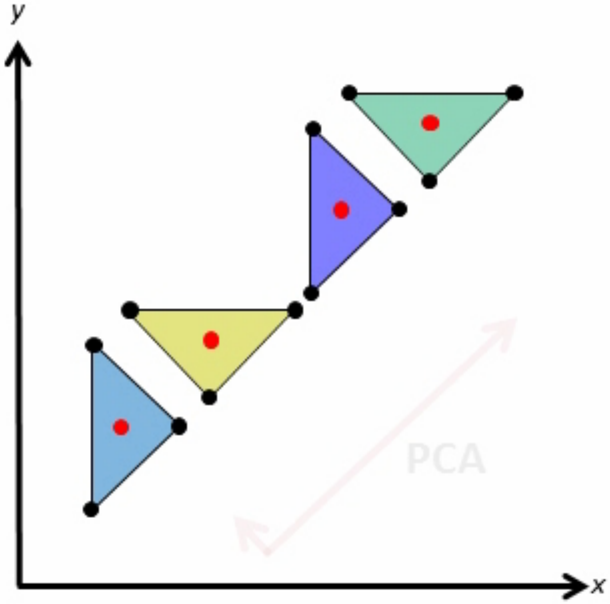


Fuzzy C-means GPU-based Collision Detection

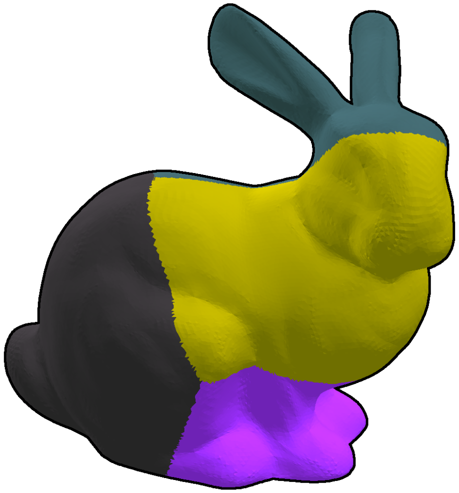
Input: primitives of all objects

Output: intersecting pairs of primitives

```
Subdivide scene into c clusters using fuzzy C-means
for all clusters do in parallel
    compute and apply PCA
    sort AABBs along longest principle axis
    collect all overlapping intervals
```

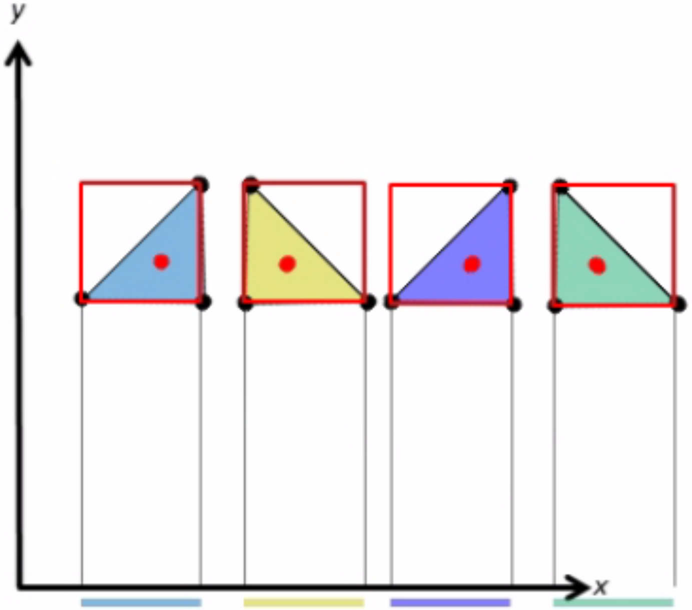


Our Algorithm



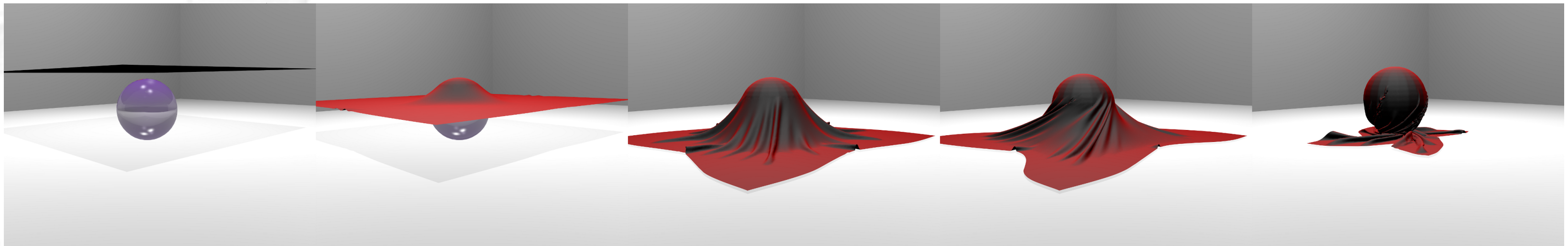
Fuzzy C-means GPU-based Collision Detection

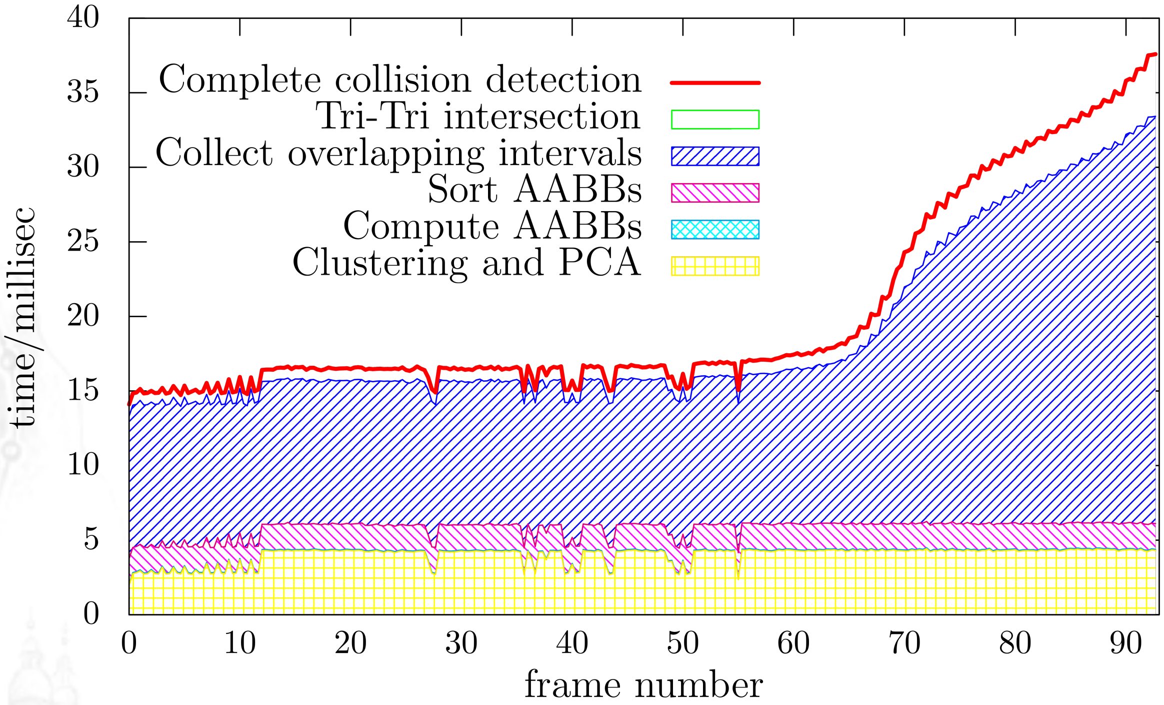
```
Input: primitives of all objects
Output: intersecting pairs of primitives
Subdivide scene into c clusters using fuzzy C-means
for all clusters do in parallel
    compute and apply PCA
    sort AABBs along longest principle axis
    collect all overlapping intervals
    for all overlapping intervals do in parallel
        if AABB intersect along y-axis then
            do primitive-primitive intersection test
```



Results: Cloth on Ball Benchmark

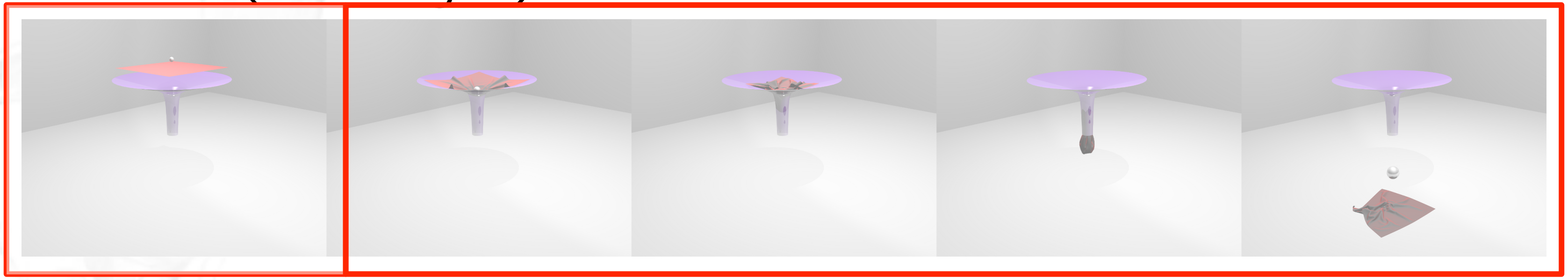
- Cloth (92k triangles) drops down on a rotating ball (760

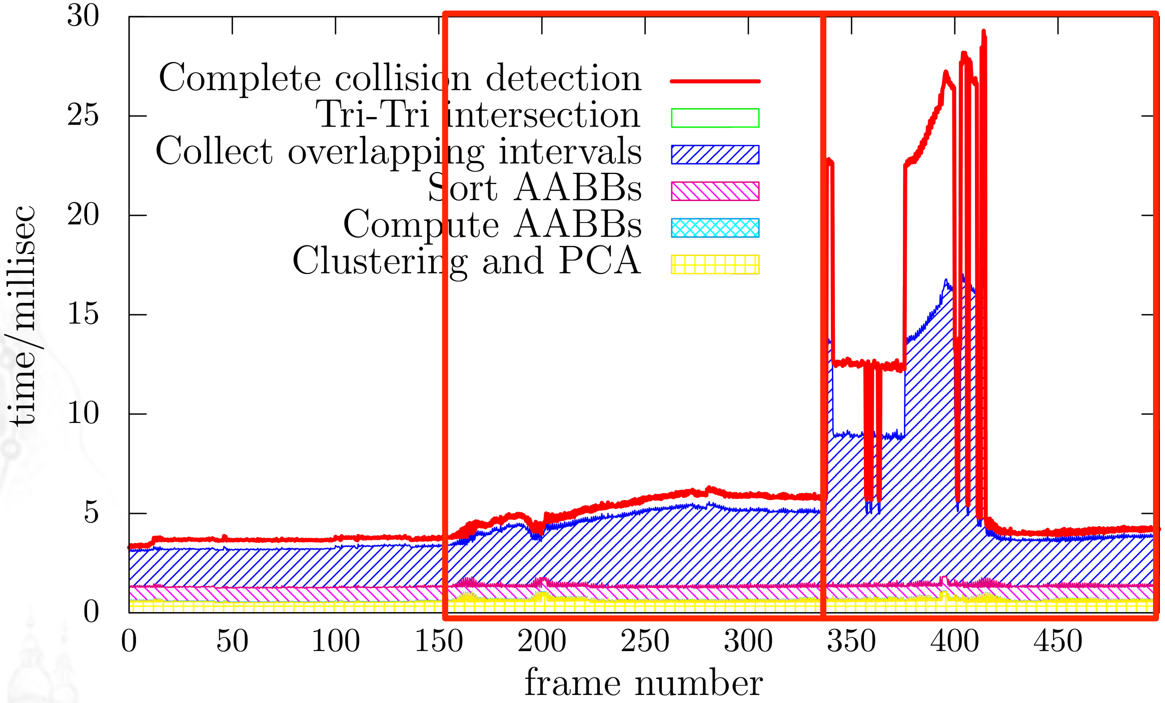




Results: Funnel Benchmark

- Ball (1.7k triangles) passes a cloth (14.4k triangles) through a funnel (2k triangles)





Conclusions and Future Work

- Completely GPU-based fast and accurate collision detection algorithm
- Broad phase and narrow phase within one single framework
- Our approach is easier to implement than many other collision detection algorithm
- We can compute external and self-collisions within one computation step
- We can handle scenes with 95k triangles in ~22*ms*

Conclusions and Future Work

- Completely GPU-based fast and accurate collision detection algorithm
- Broad phase and narrow phase within one single framework
- Our approach is easier to implement than many other collision detection algorithm
- We can compute external and self-collisions within one computation step
- We can handle scenes with 95k triangles in ~22*ms*
- Improving the PCA process → reduce number of false positives
- Virtual subdivision for extremely large triangles
- Extend to perform other proximity queries