A Different Approach for Continuous Physics

Vincent ROBERT
vincent.robert@ubisoft.com
Physics Programmer at Ubisoft

GAME DEVELOPERS CONFERENCE
SAN FRANCISCO, CA
MARCH 5-9, 2012
EXPO DATES: MARCH 7-9
2012

A Different Approach for Continuous Physics

Existing approaches

Our method

Limitations

Performances

Conclusion

A Different Approach for Continuous Physics

Existing approaches

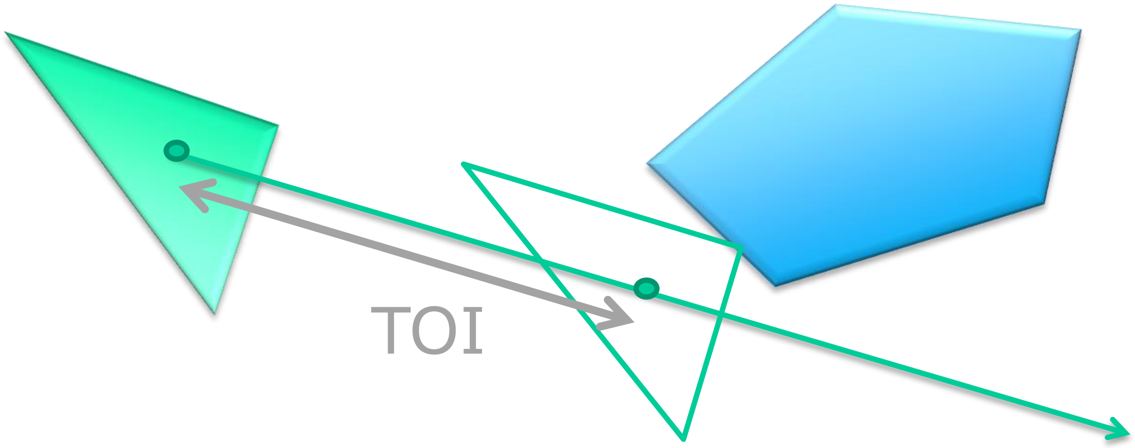
Our method

Limitations

Performances

Conclusion

Linear convex cast

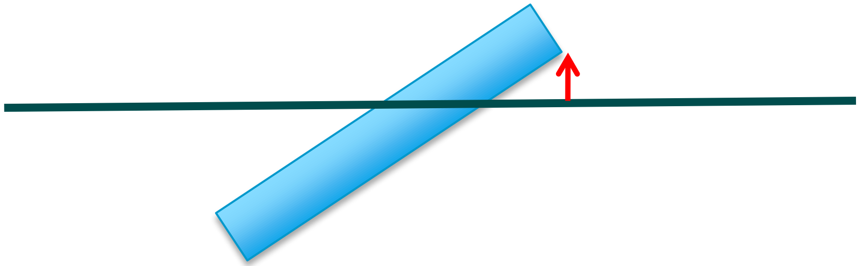


Compute the Time of Impact (TOI) between two convex shapes

An issue can still occur

With the Linear cast, a future collision can be detected.

Detecting the collision != handling it.



Static mesh

Dynamic box

Existing Continuous Physics method

while (TOI found)

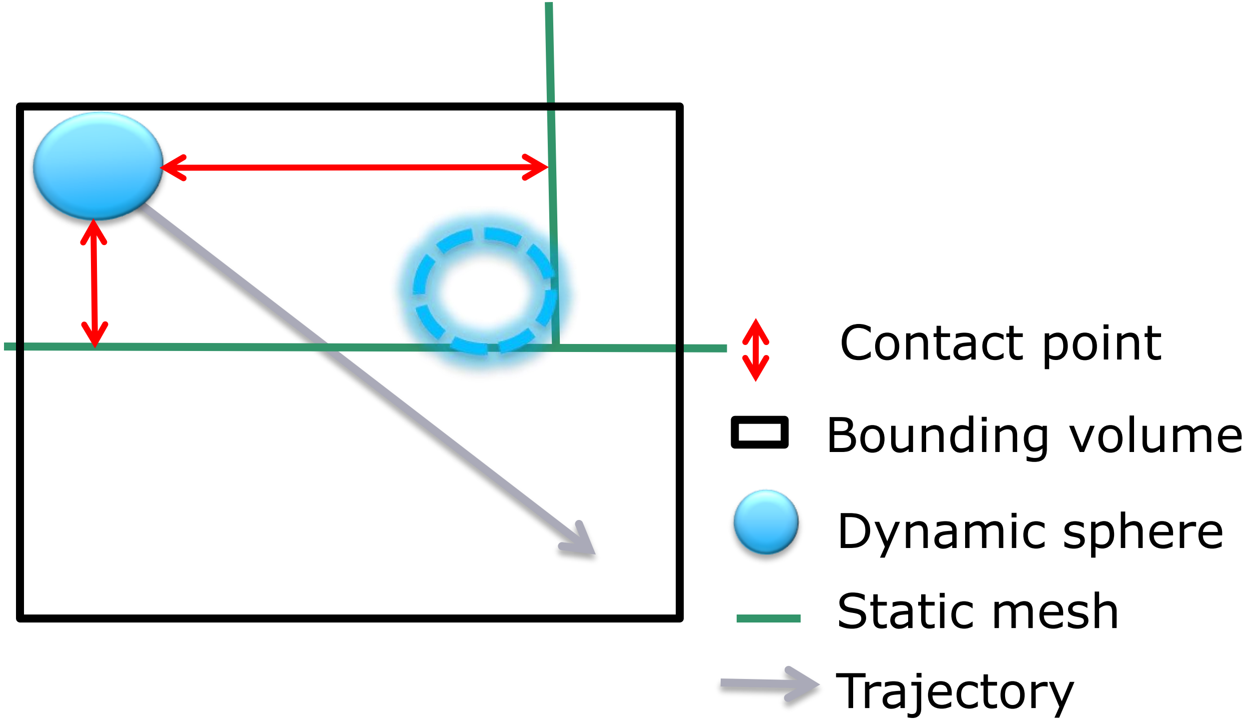
- Move at earliest time of impact
- Compute collision
- Solve

This method costs a lot of CPU.

Does not always prevent tunnelling of fast rotating bodies.

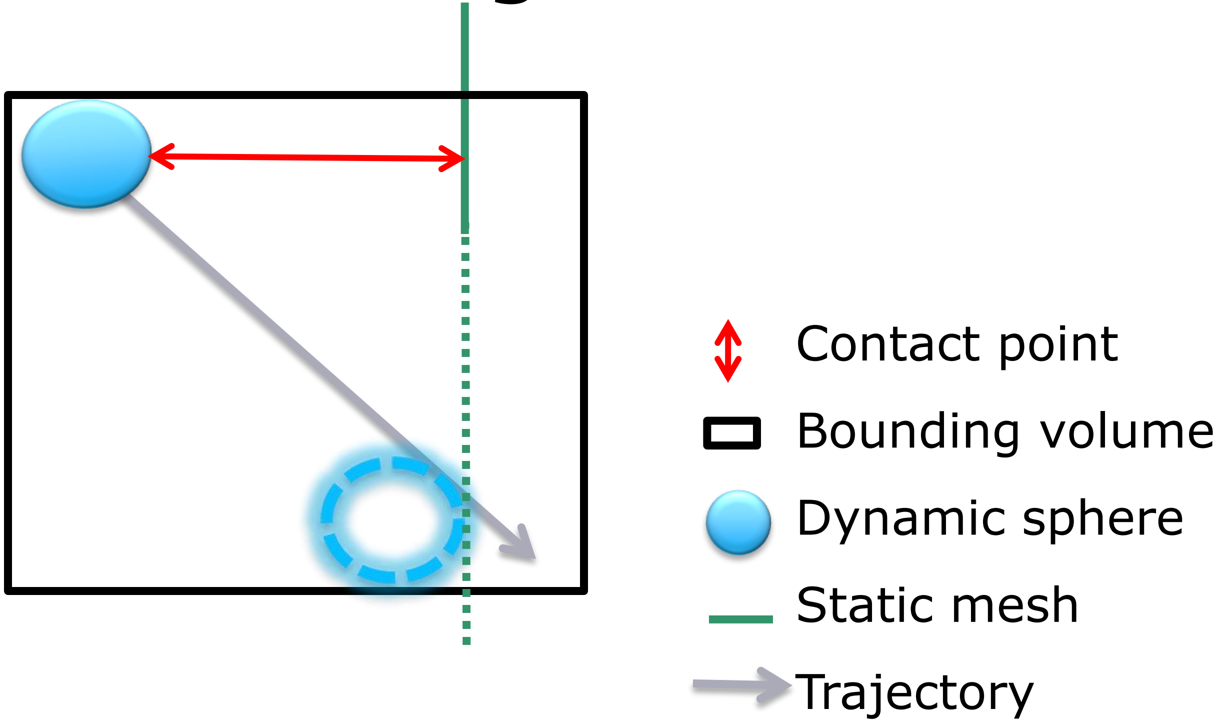
Speculative contact points

- Contact point with a positive distance
- Cheap and efficient solution
- Handles various impacts in one frame



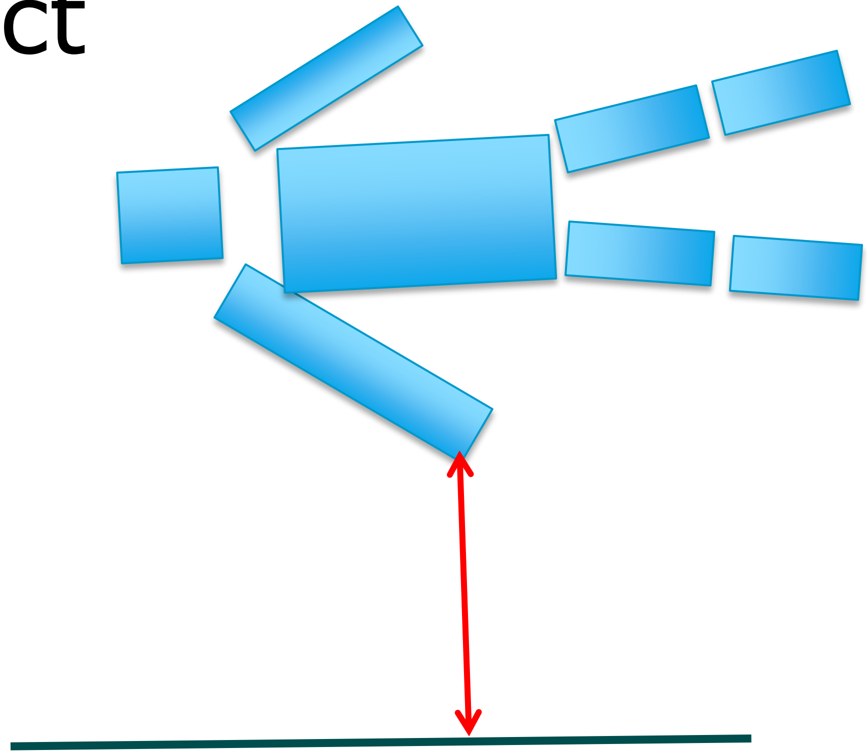
Speculative contact points: Ghost bug

Stops the dynamic rigid body even if it shouldn't.



Speculative contact

- This solution doesn't always prevent tunnelling issues.
- This issue can occur with ragdoll.



A Different Approach for Continuous Physics

Existing approaches

Our method

Limitations

Performances

Conclusion

Objective: No tunnelling issues

- No iterative algorithm that costs a lot of CPU:
  - Iteration of all the pipeline
- Robust:
  - Few solver iterations
  - Handling variable frame rate
  - Handling fast rotating bodies

Our method

Our approach involves some modifications at different stages of the physics pipeline:

- Broad phase
- Narrow phase
- Constraint creation
- Solver

Our method

Broad phase

Narrow phase

Constraint creation

Solver

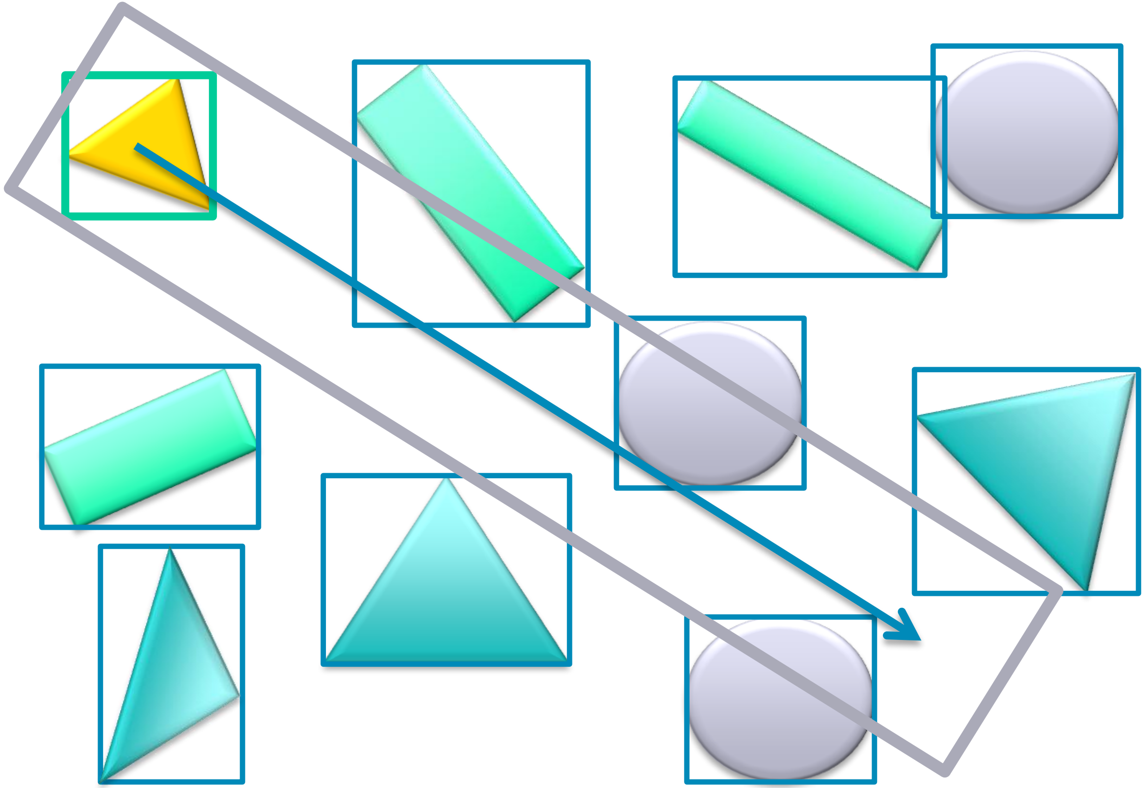
Moving body in the broad phase

Body's linear velocity is used to compute the trajectory:

Segment = velocity * deltaTime

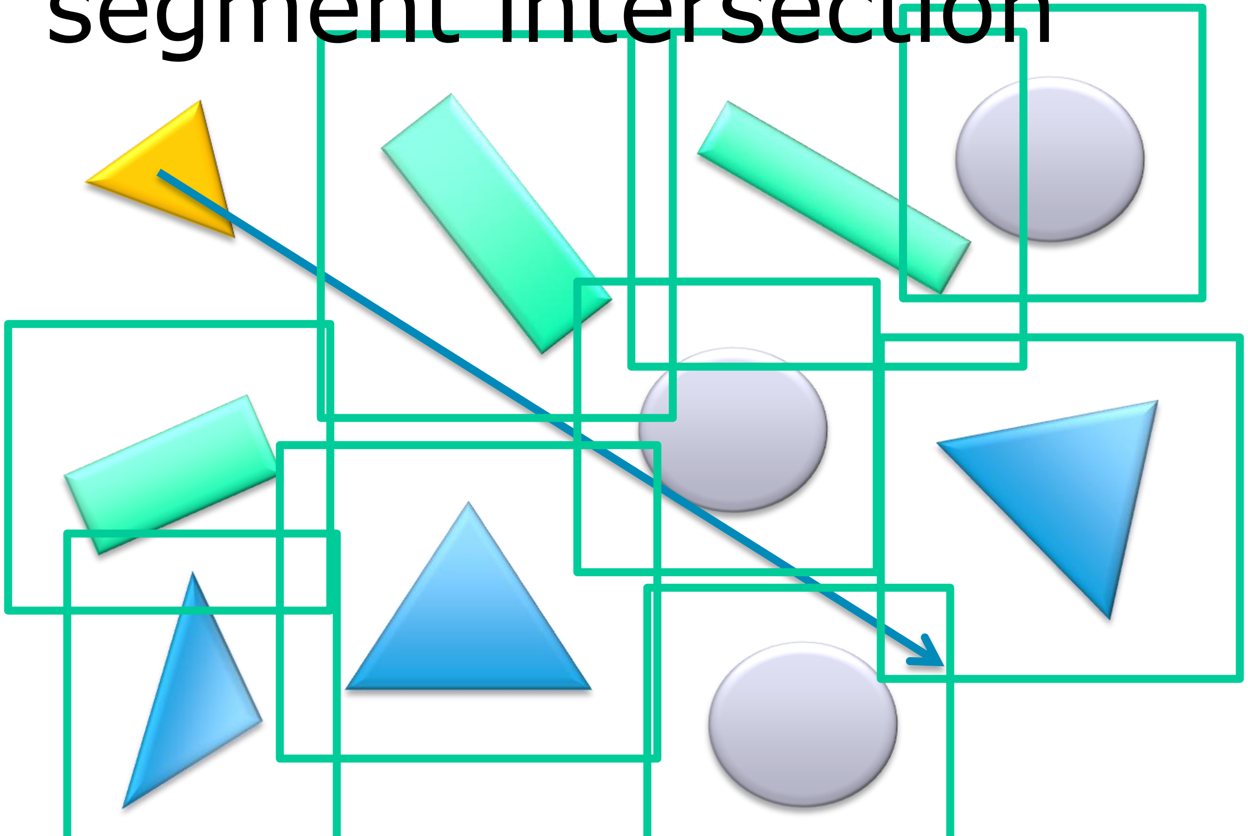
The segment is used to detect the potential collisions.

Detecting the potential colliding bodies



- Consider the trajectory.
- Use the bodies' Axis Aligned Bounding Box (AABB).

Transfer volume and compute a segment intersection



- Add AABB of the moving body to the other AABB
- Compute intersection between segment and AABB
- Generate the expected body pairs

Our method

Broad phase

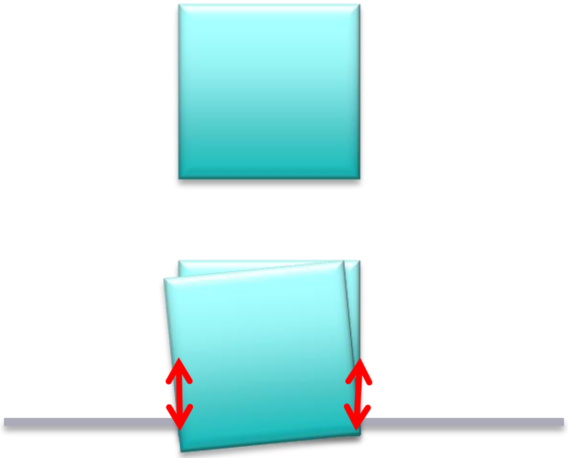
Narrow phase

Constraint creation

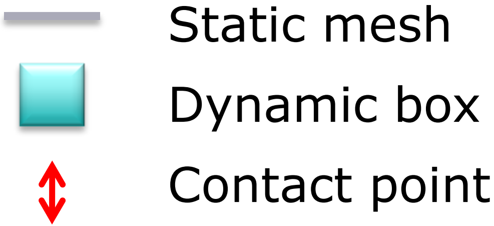
Solver

Incremental Manifold

Frame 0

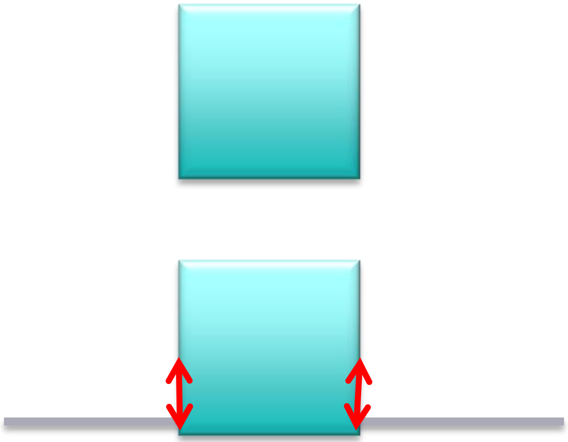


Incremental manifold provides one new contact point at each frame.

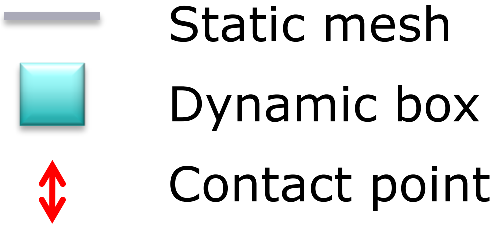


Full Manifold

Frame 0

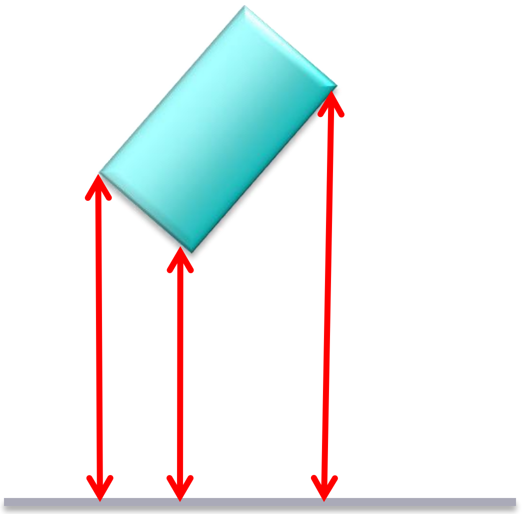


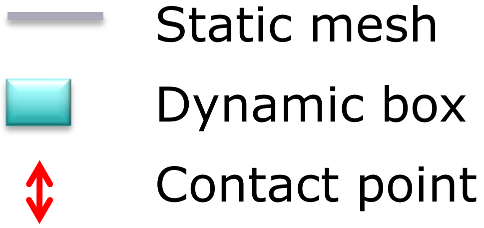
Full manifold provides all contact points in one frame.



Distance-based full manifold

Potential contact points in full manifold





Supported shapes

Shapes supported on all rigid bodies:

- Sphere (point + radius)
- Capsule (segment + radius)
- Box
- Convex

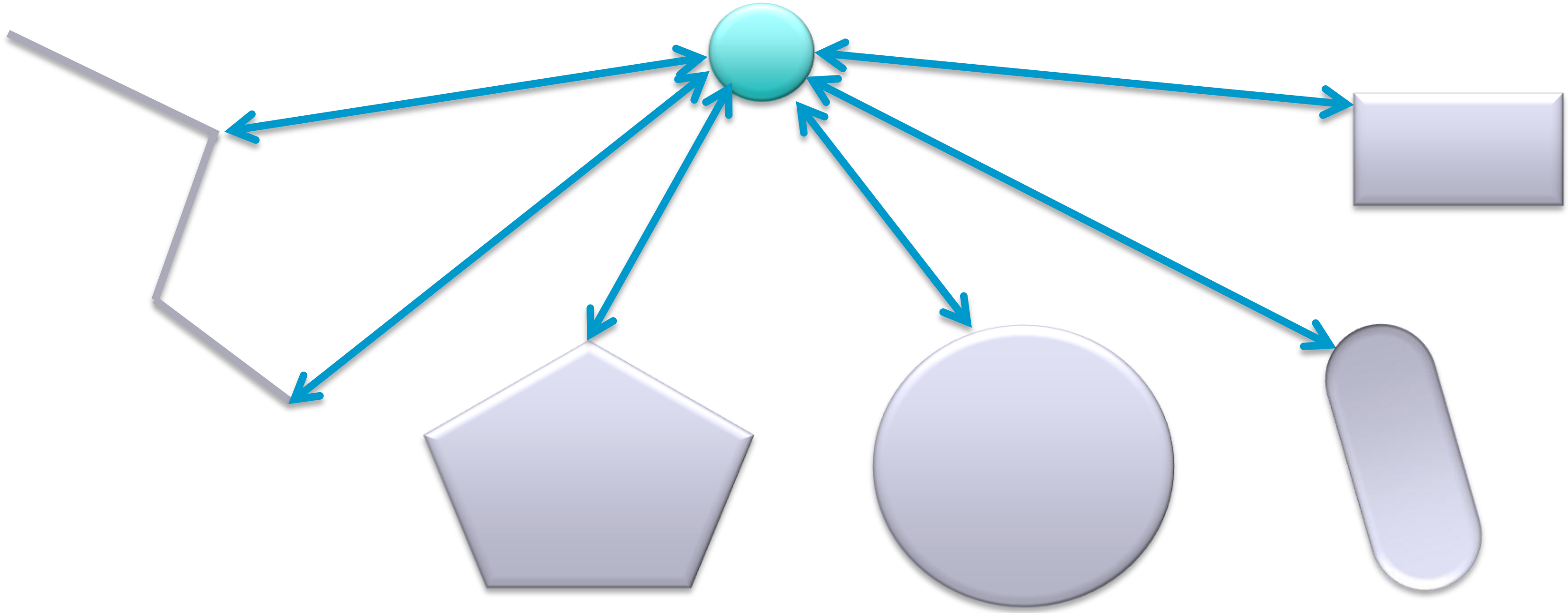
Shapes only supported on static rigid bodies:

- Mesh / Height map (collision with triangles)

Distance-based full manifold with a **sphere**

- One contact point = full manifold.
- Gilbert Johnson Keerthi (GJK) is a well known algorithm to compute the minimum distance between two convex shapes.
- Use GJK against any other shapes.

Distance-based full manifold with a sphere

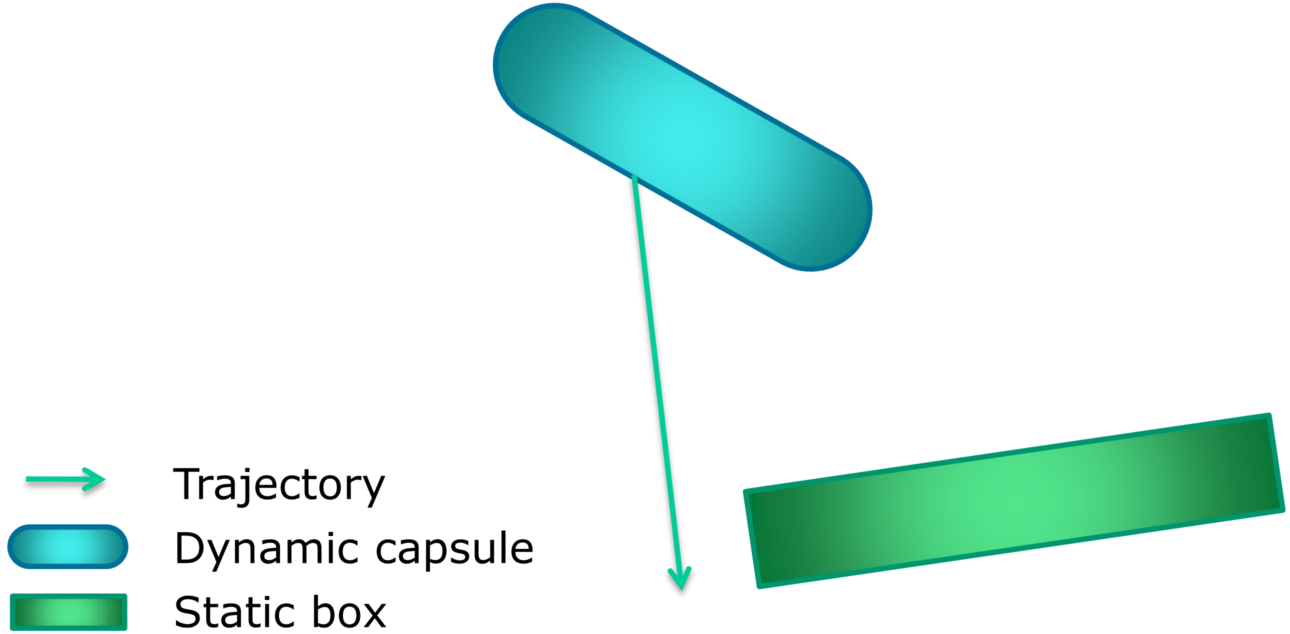



Contact point

Distance-based full manifold with a capsule

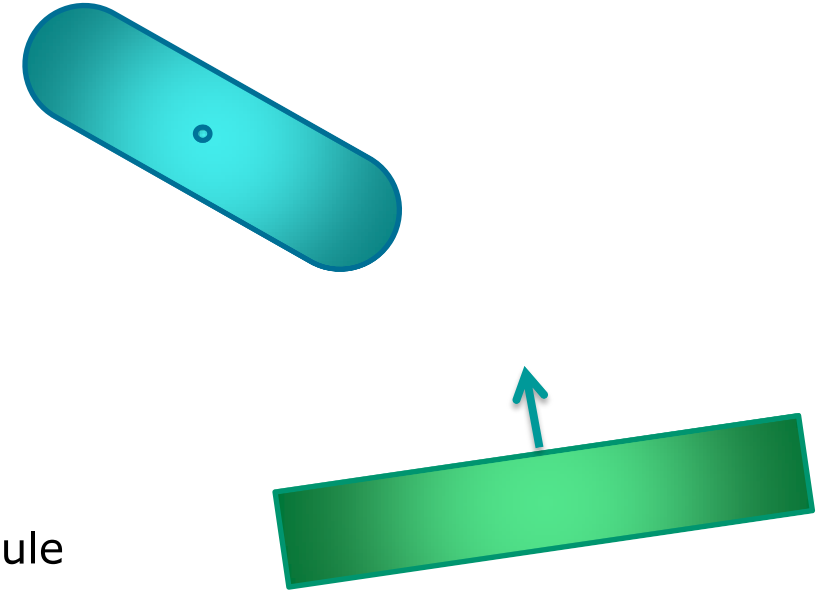
- Full manifold is required for a capsule.
- But, how do we calculate it?

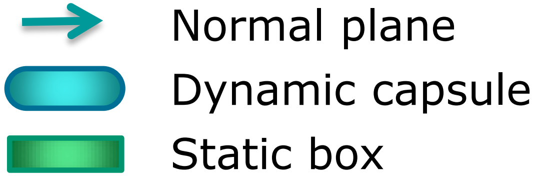
Distance-based full manifold between a capsule and a box



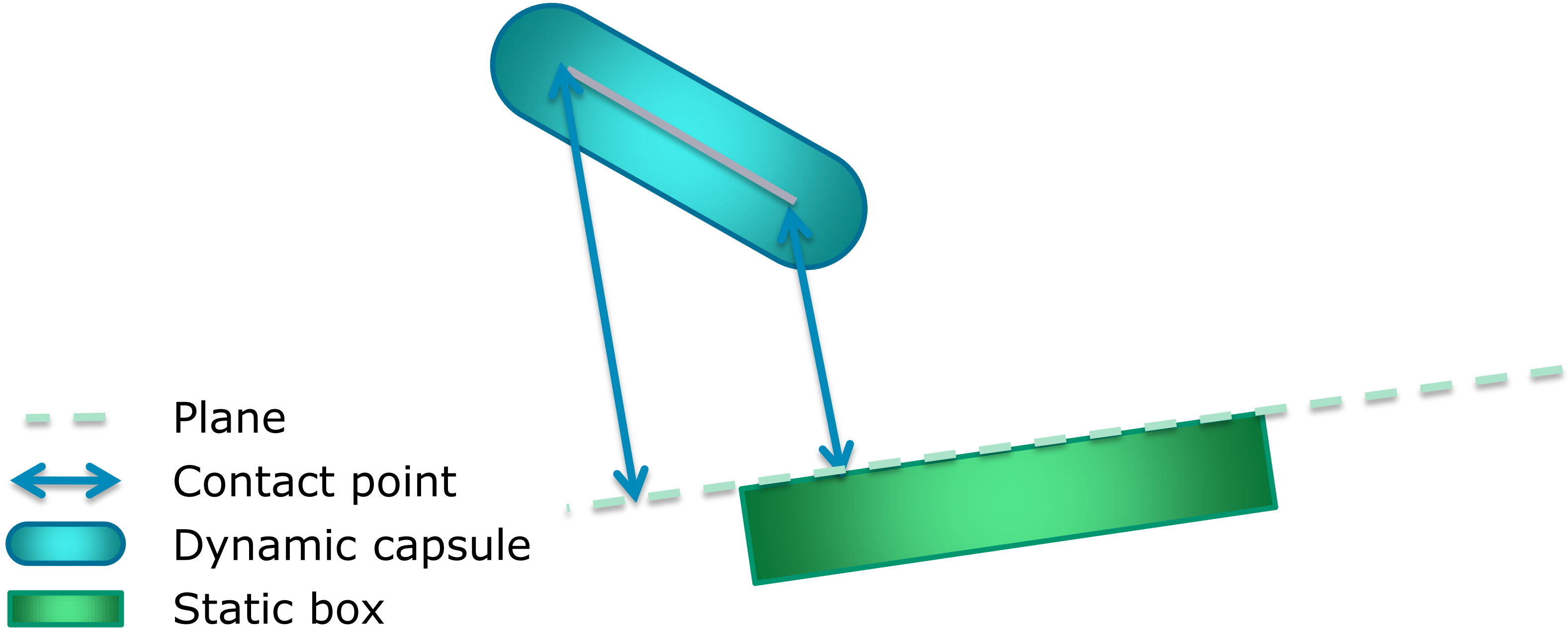
Trajectory
Dynamic capsule
Static box

Find the box's reference plane

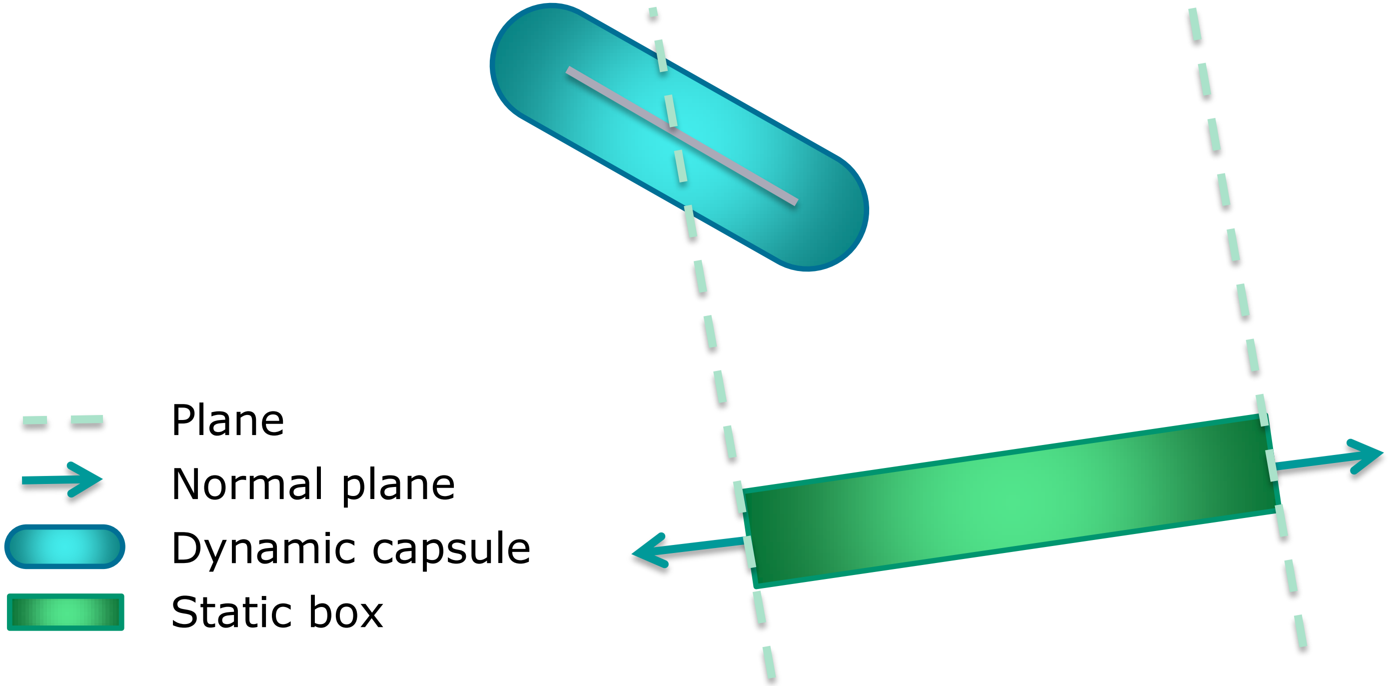




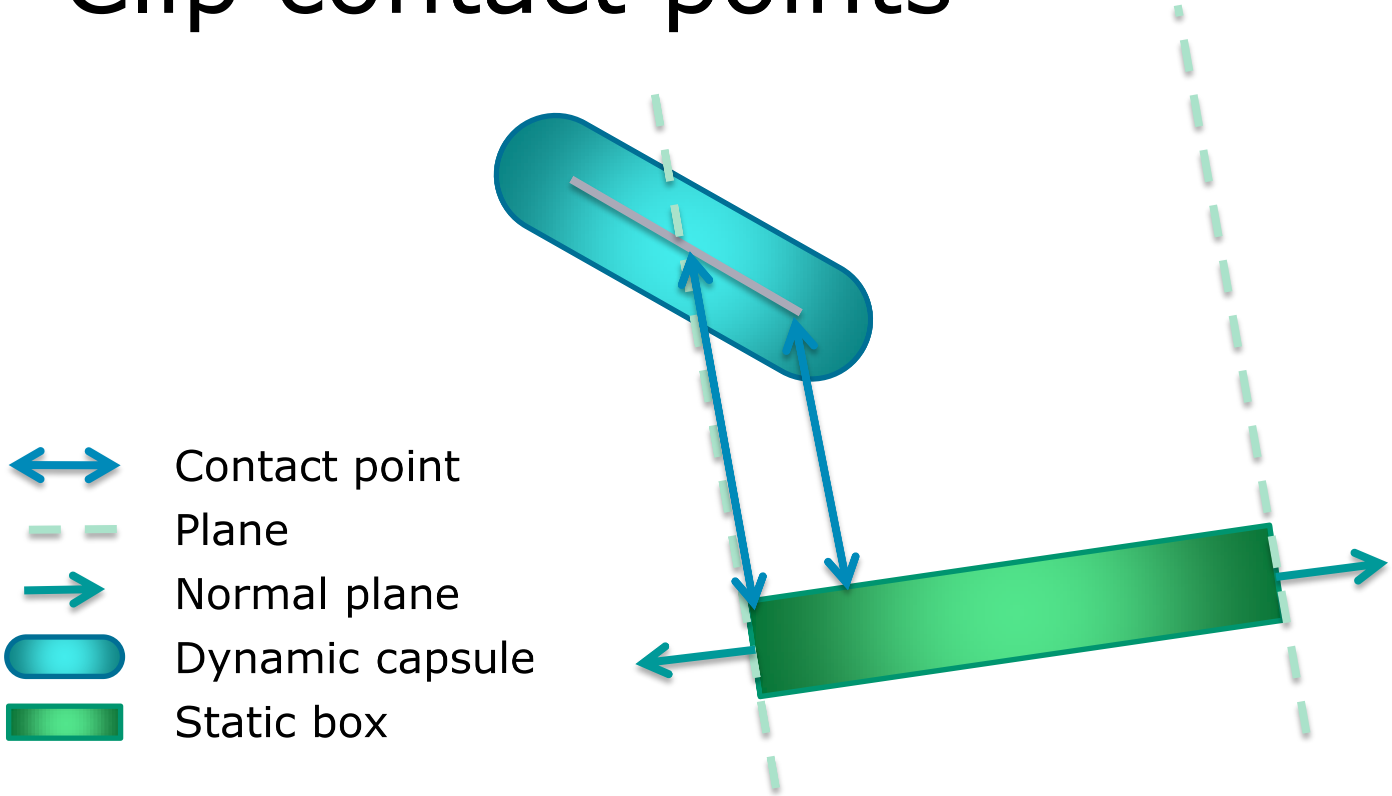
Project capsule extremities on the plane



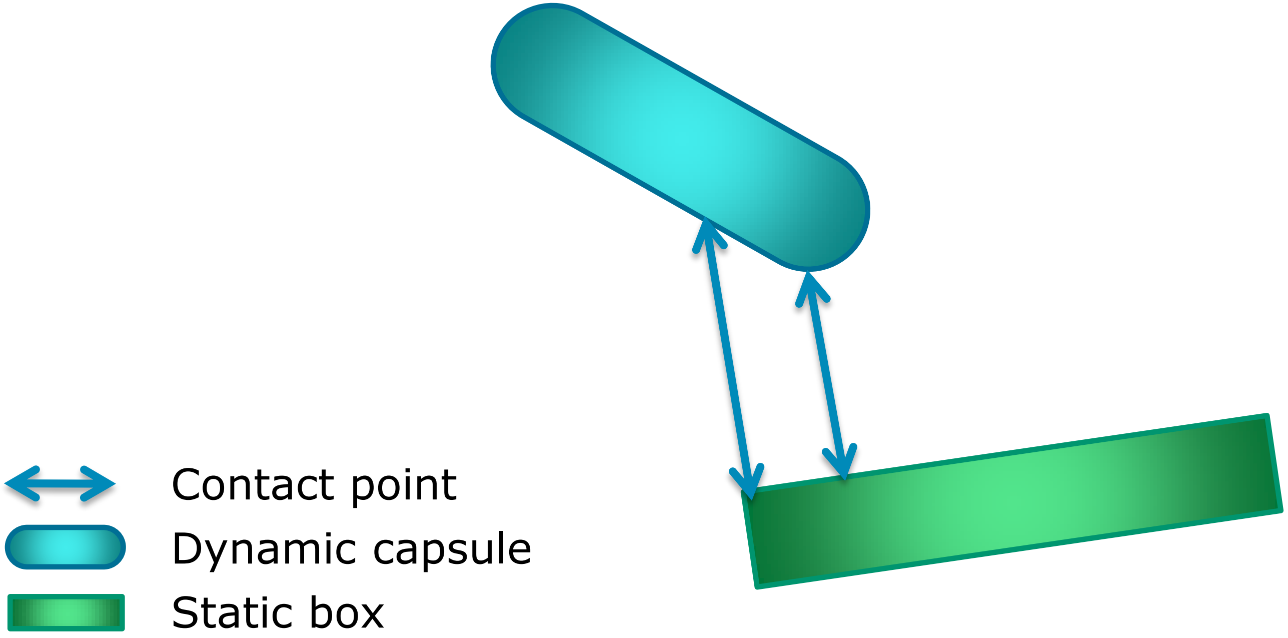
Find clipping planes: orthogonal to the reference plane with an edge in common



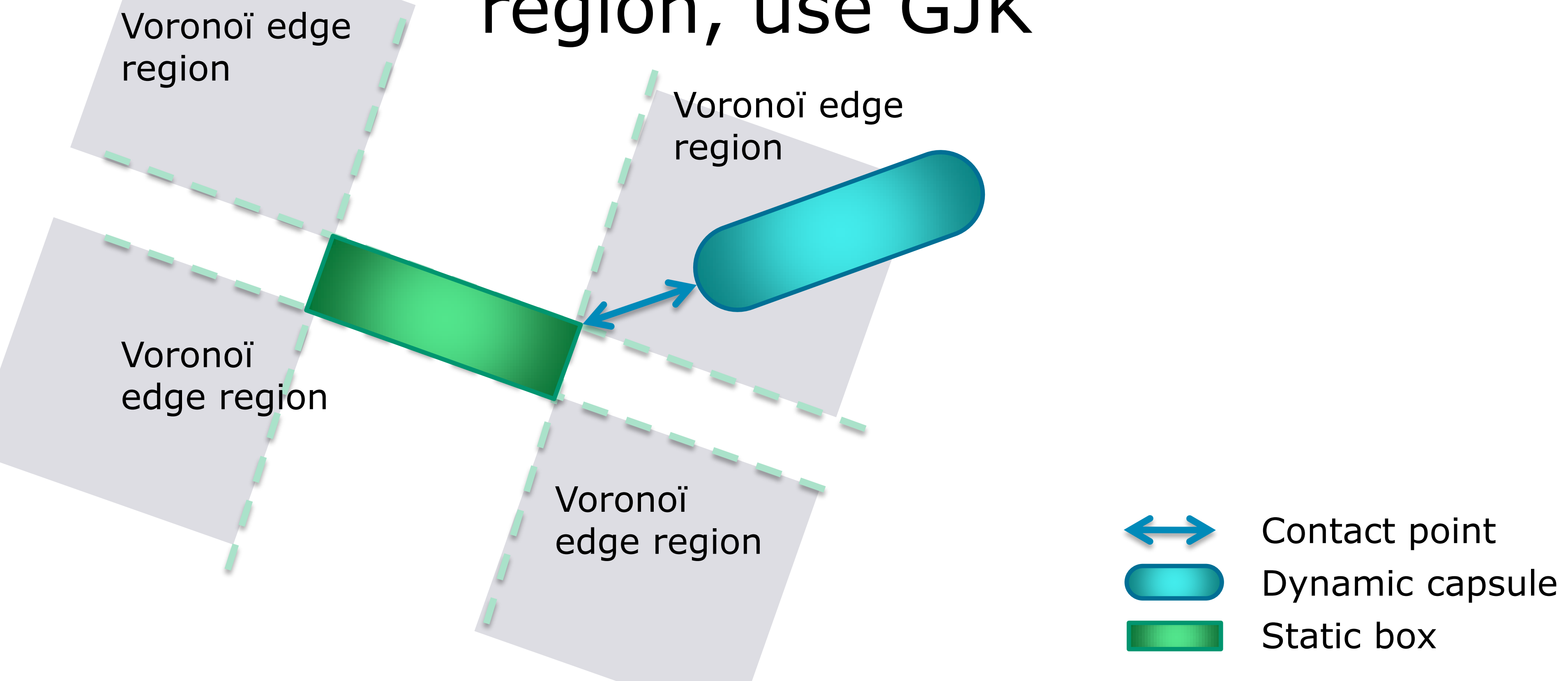
Clip contact points



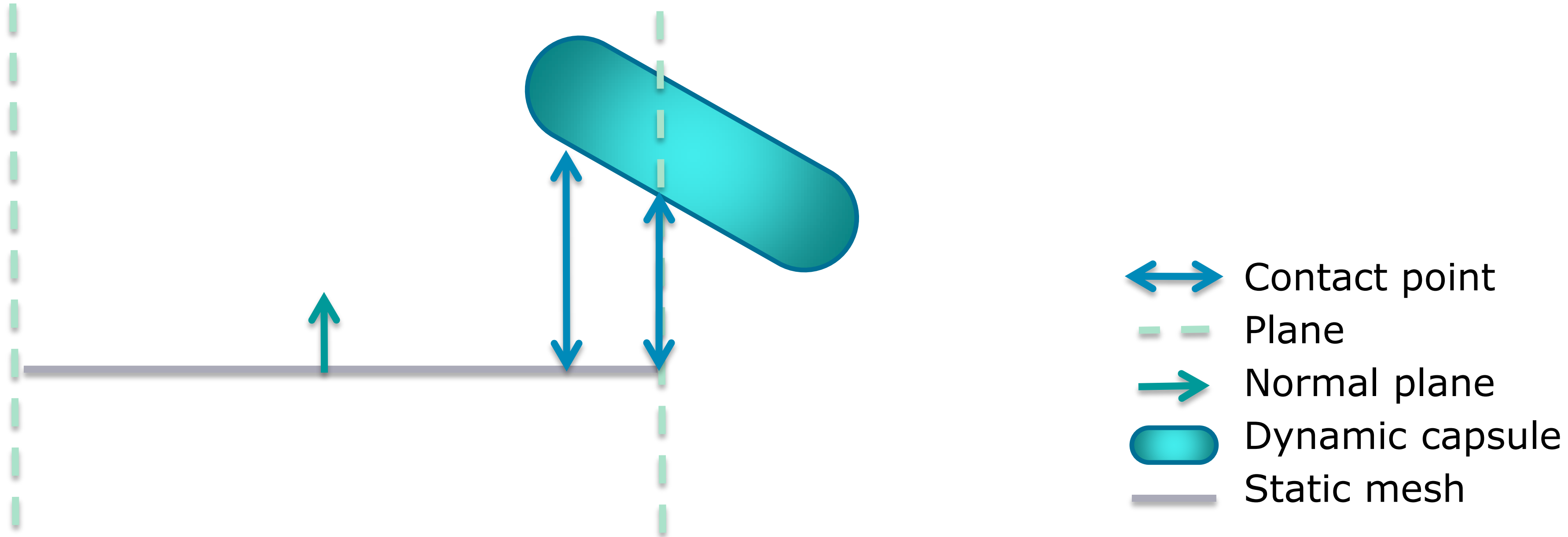
Compute contact points, considering the capsule radius



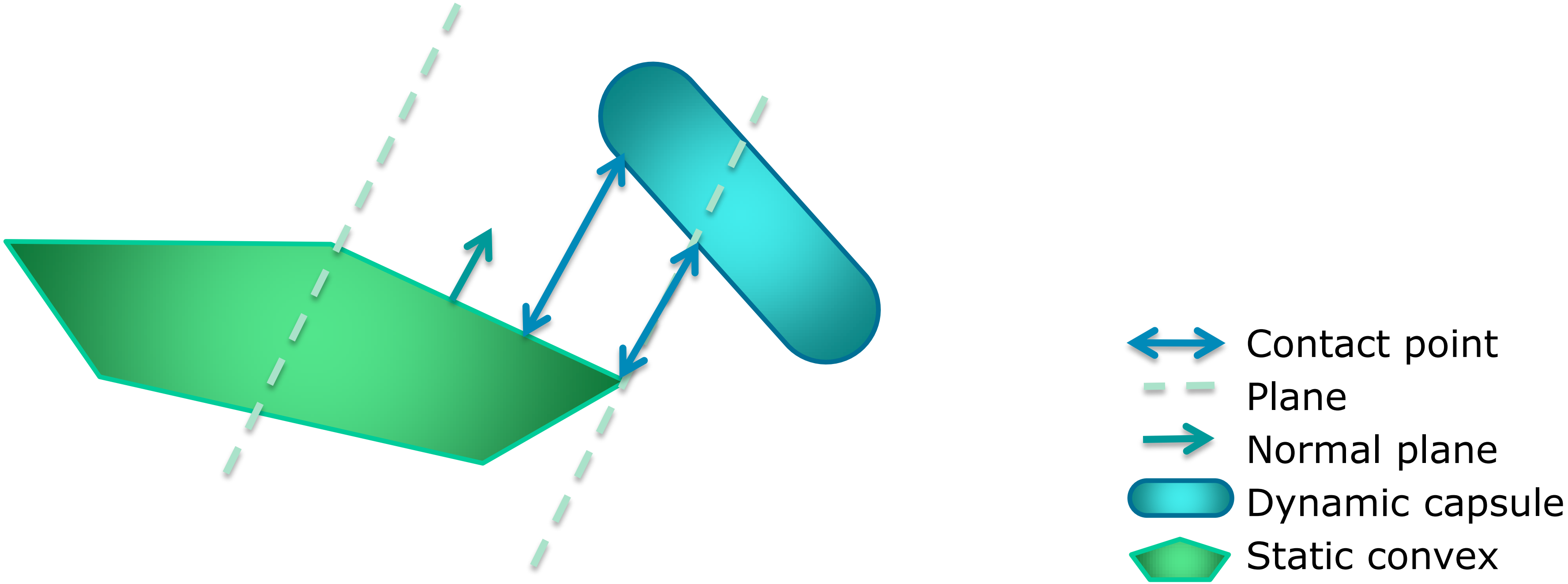
If the capsule is in the Voronoï edge region, use GJK



Distance-based full manifold between a capsule and a triangle



Generalizing the computation to convex



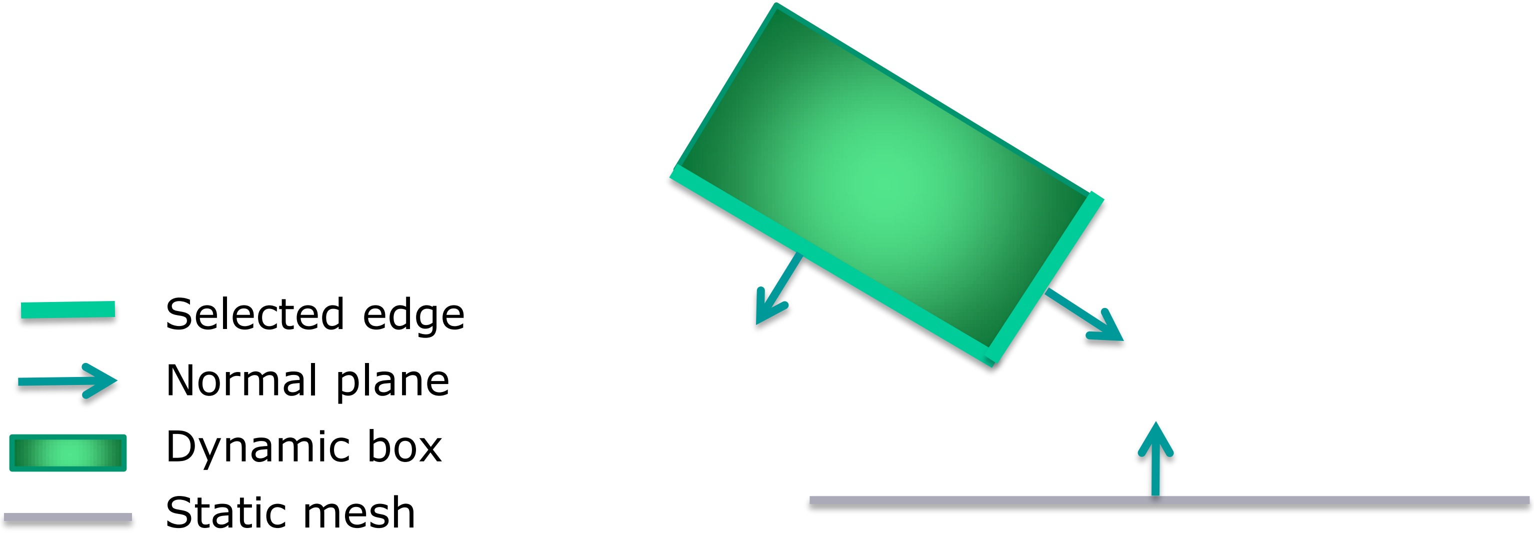
Distance-based full manifold with a box

- Full manifold is required for a box.
- Same technique:

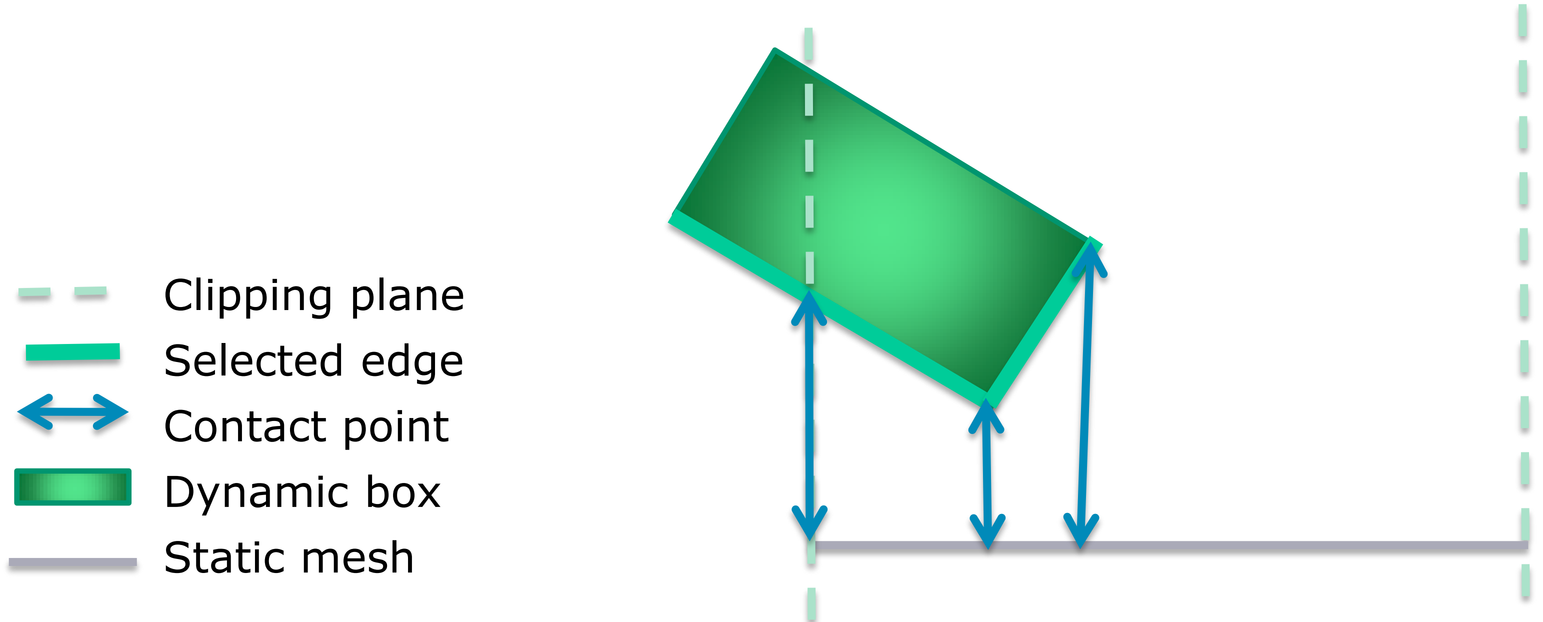
  Clip edges instead of segment.
- Don't clip all edges:

  Select the right ones.

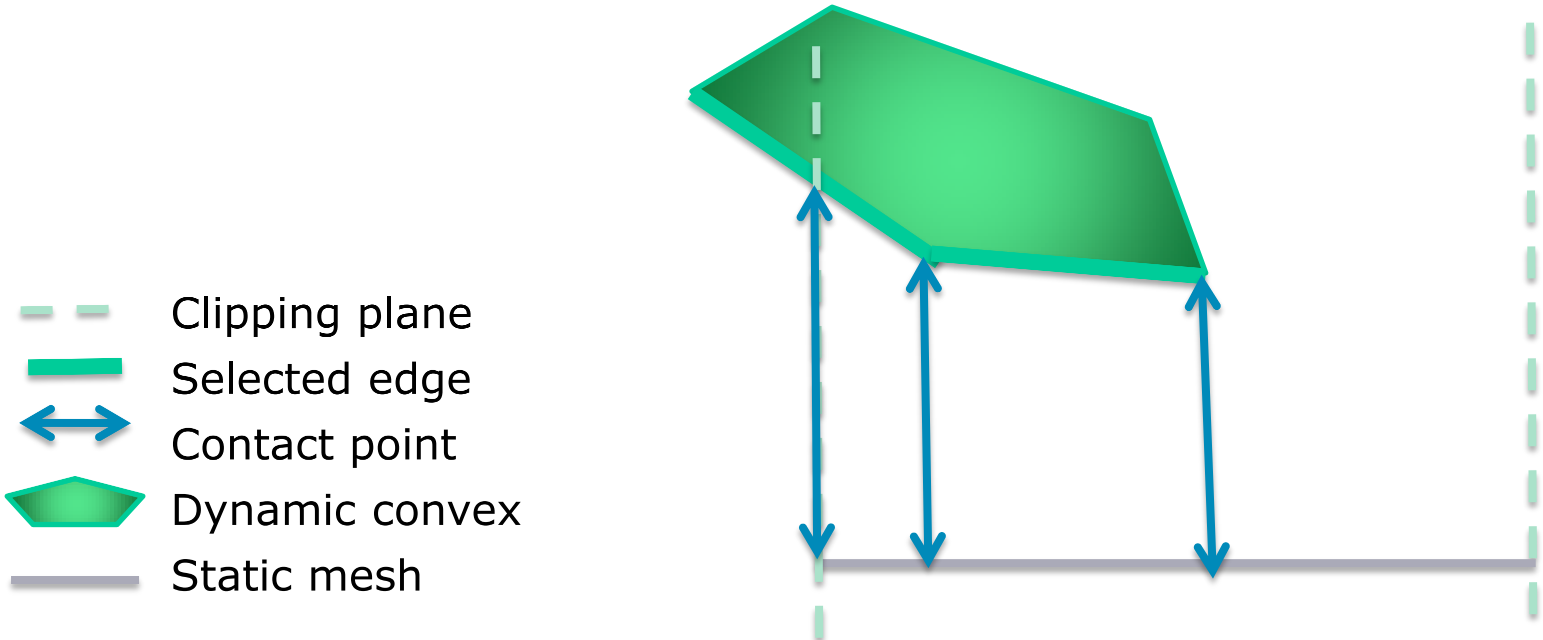
Use edges that face the reference plane



Distance-based full manifold between a box and a triangle

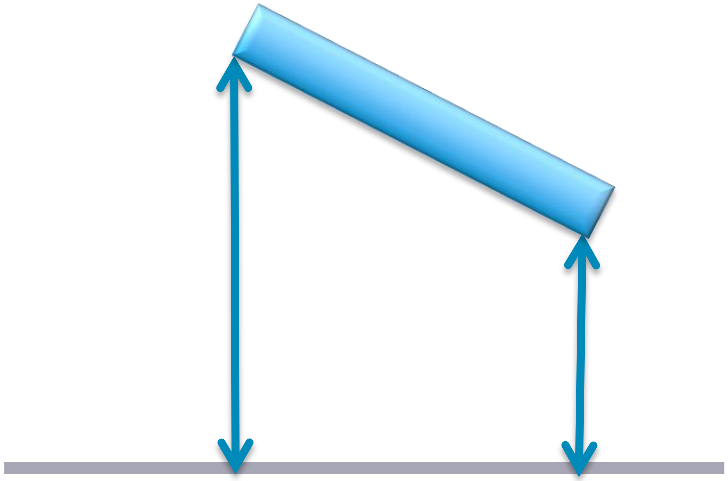


Distance-based full manifold: Generalization between two convexes

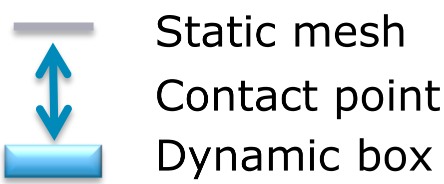


Handle potential and real contact points

Potential contact points

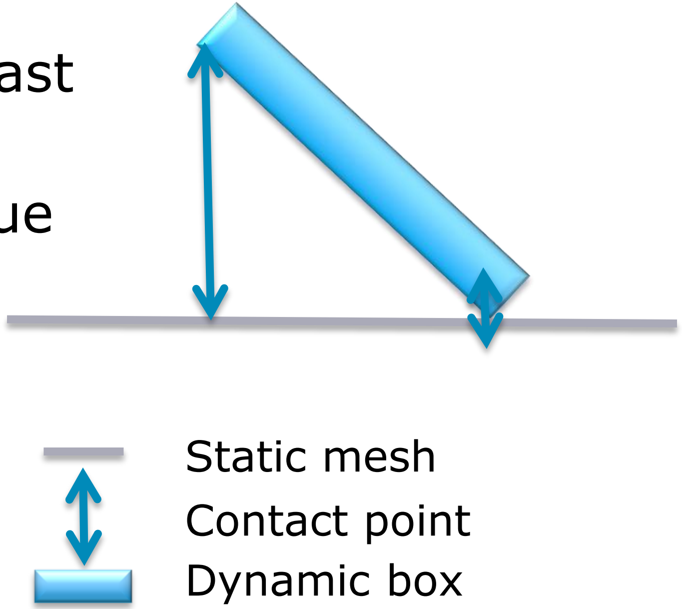


Real contact points



Handle potential and real contact points at the same time

- Same frame:
  - Real contact points can generate fast rotation.
  - Potential ones avoid tunnelling issue in the same frame.
- Same part of the code:
  - Reuse geometry information.
  - Maximize cache access.



Our method

Broad phase

Narrow phase

Constraint creation

Solver

Constraint creation for **real** contact points and **potential** ones

Real contact

- Restitution is computed.
- Friction is added.

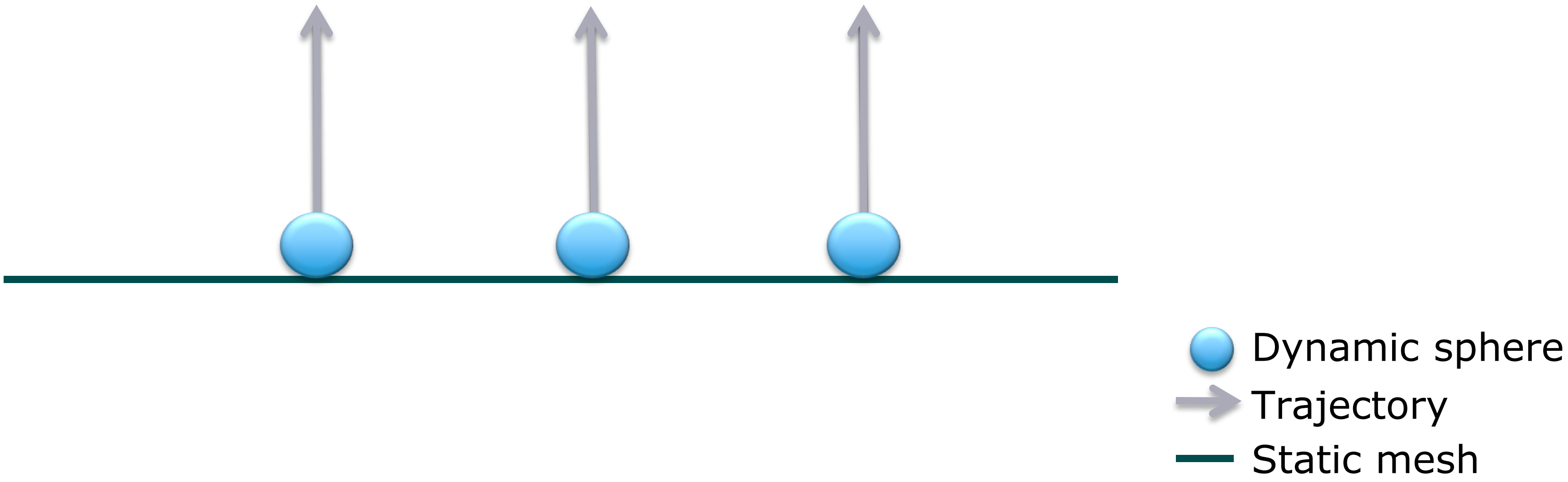
Potential Contact

- No restitution.
- No friction.
- Cheaper: no need to solve the friction.

Restitution

- Potential contact points reduce the velocities to reach the point of impact on the obstacle.
- At the next frame the body reaches the obstacle with reduced velocities.
- Don't use the current velocities to compute the restitution.

Restitution example using the current velocities

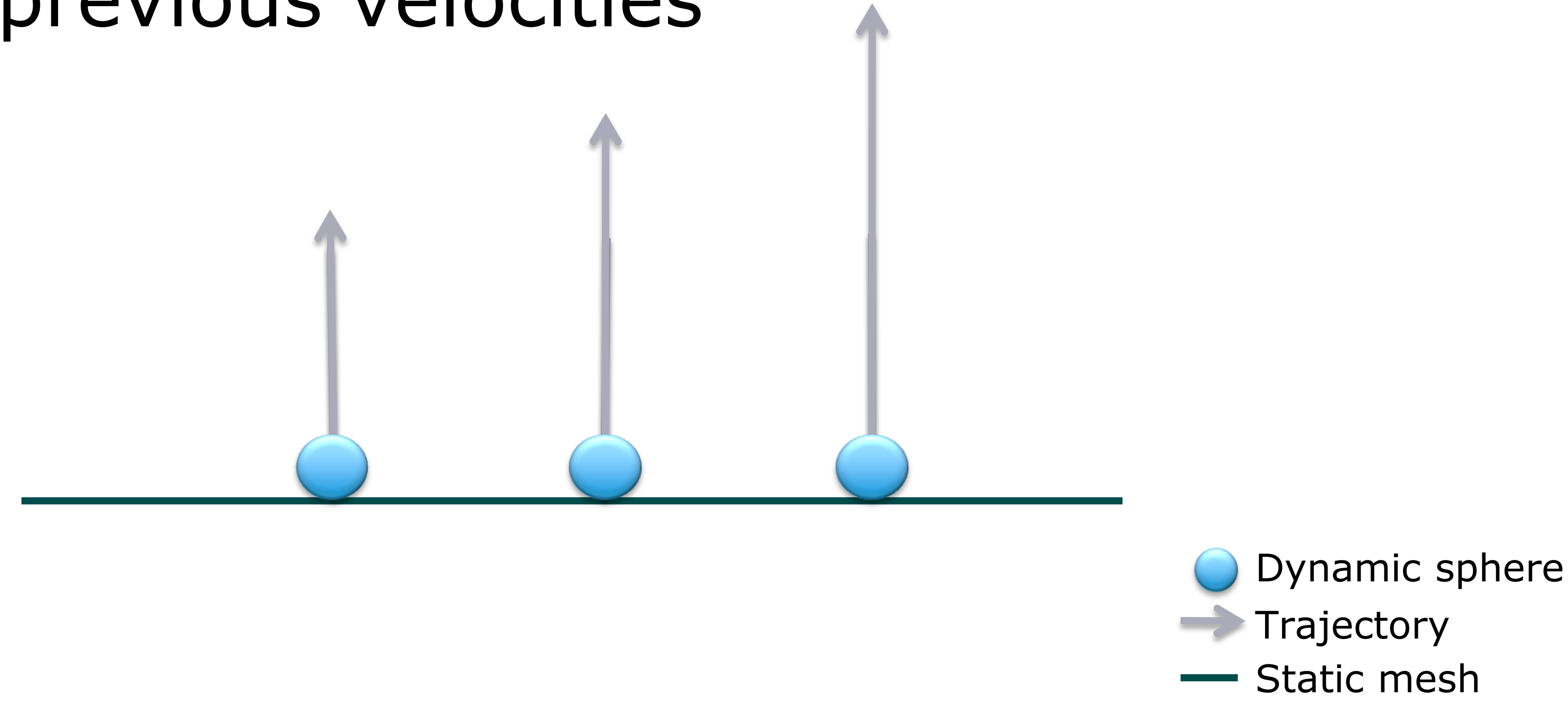


Handling restitution

Using current velocities results in a false restitution. Therefore, we must:

- Store the previous velocities; and
- Use them to compute restitution.

Restitution example using the previous velocities



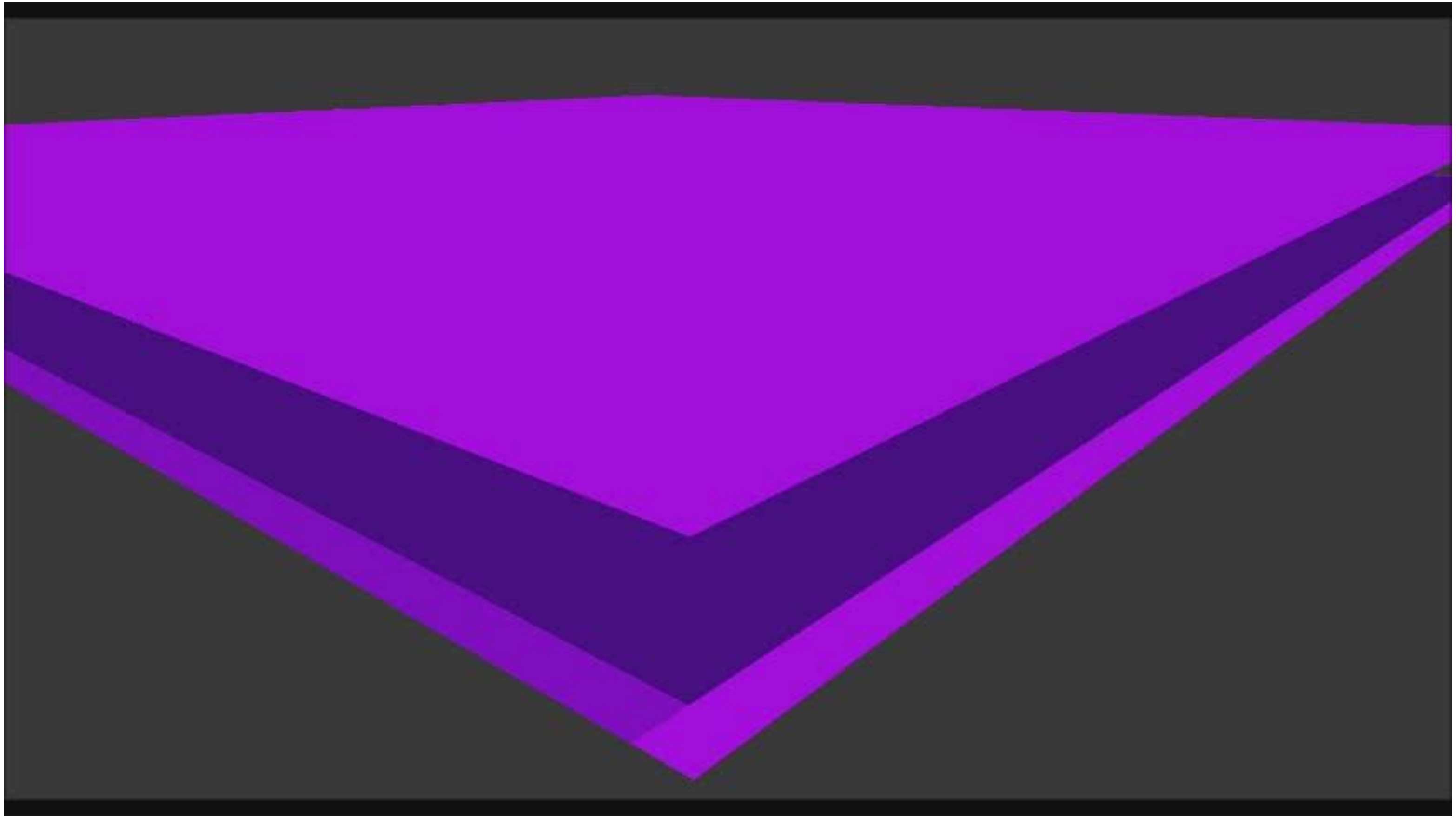
Restitution

Pro

- Restitution is correct, with no loss of energy.

Con

- Still a loss of distance during the frame of impact. This small loss is not visible in a video game.



Our method

Broad phase

Narrow phase

Constraint creation

Solver

Organize constraints in the solver

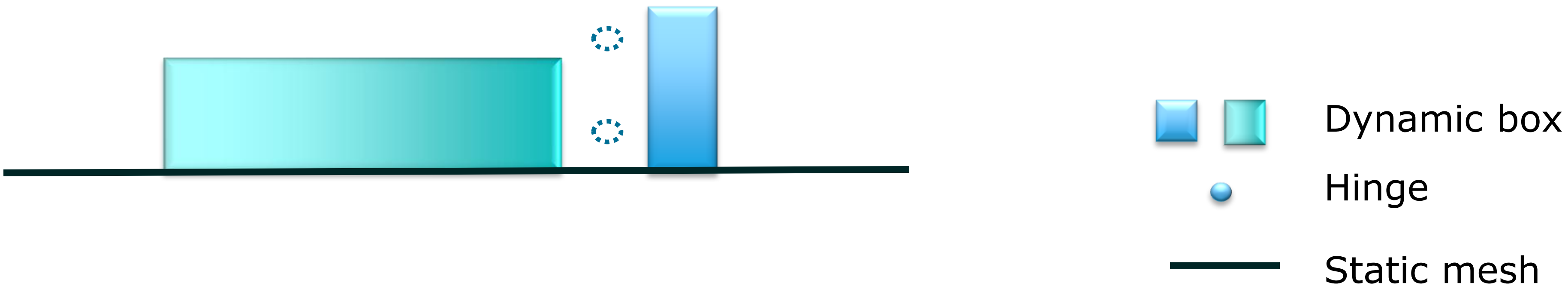
With a Gauss Seidel solver, each constraint changes the velocities of bodies.

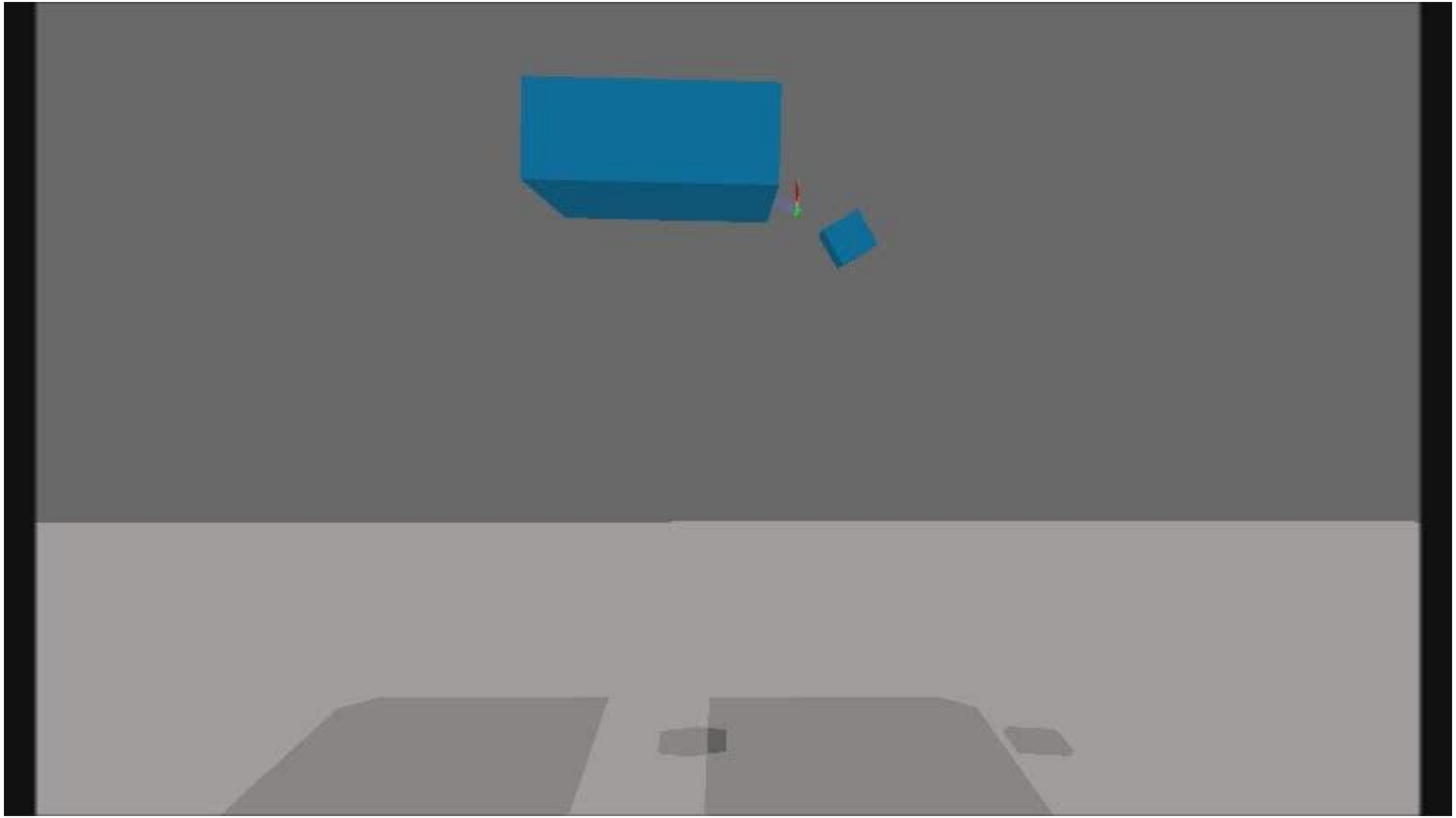
The latest solved constraints have more importance.

Group the constraints by type.

Sort them by importance.

Hinge vs. Contact: hinge solved first avoids tunnelling issues





A Different Approach for Continuous Physics

Existing approaches

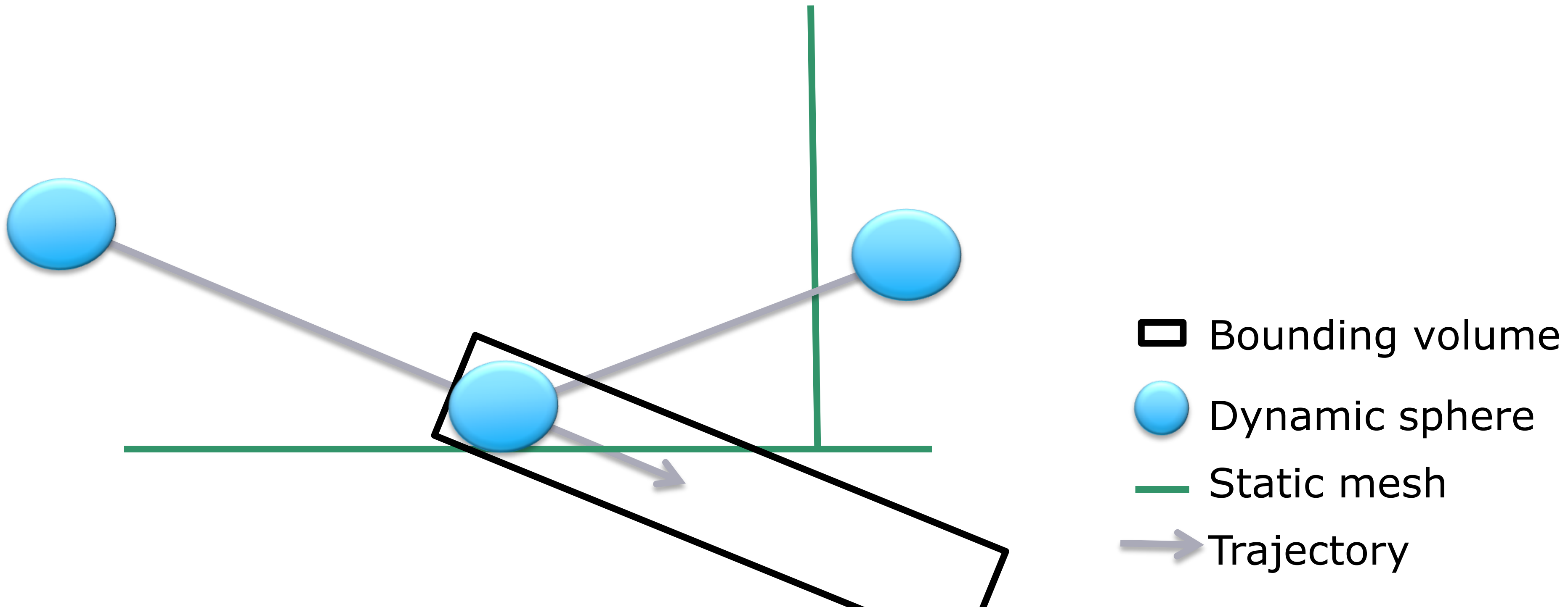
Our method

Limitations

Performances

Conclusion

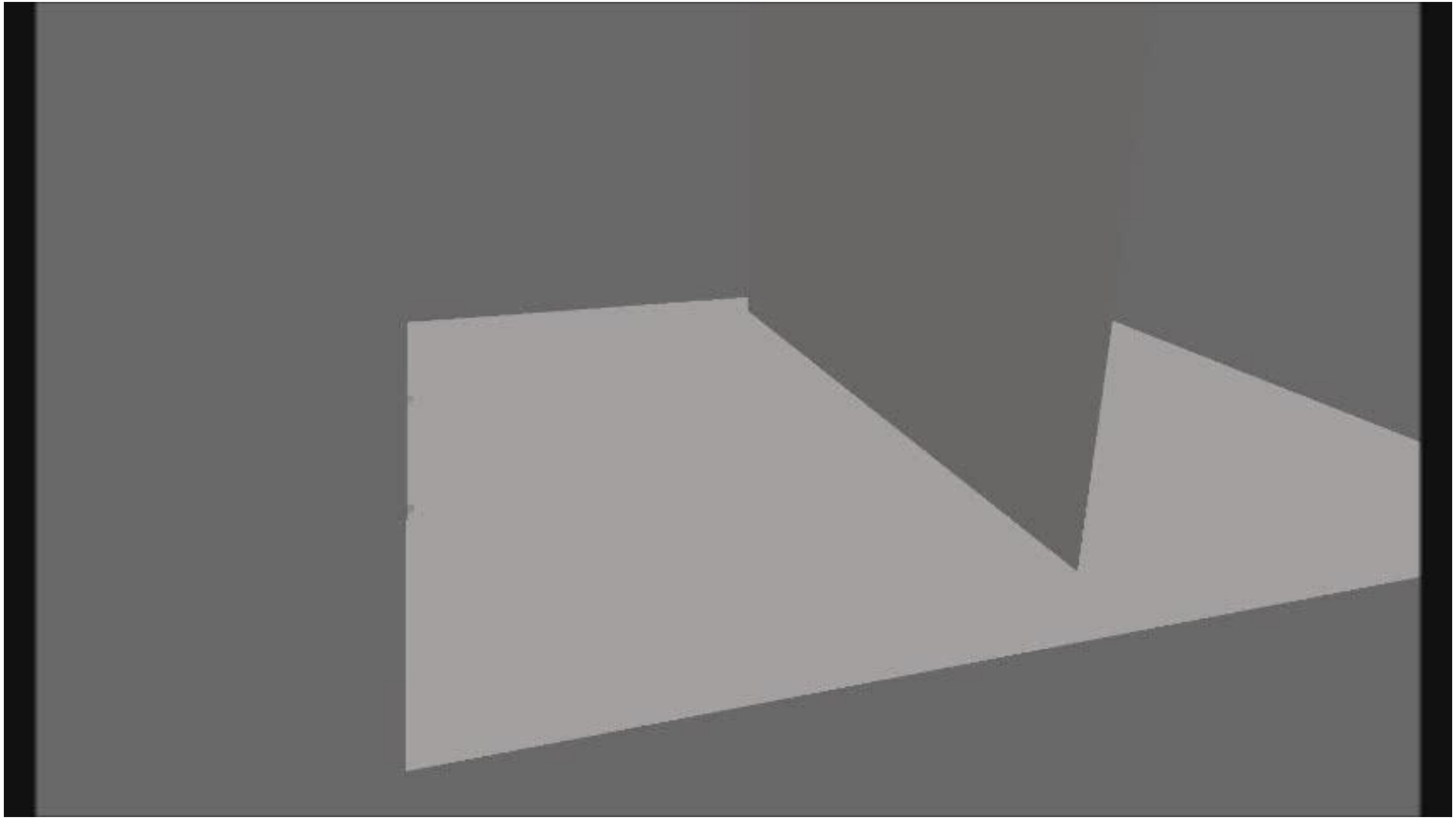
Limitation on the second impact



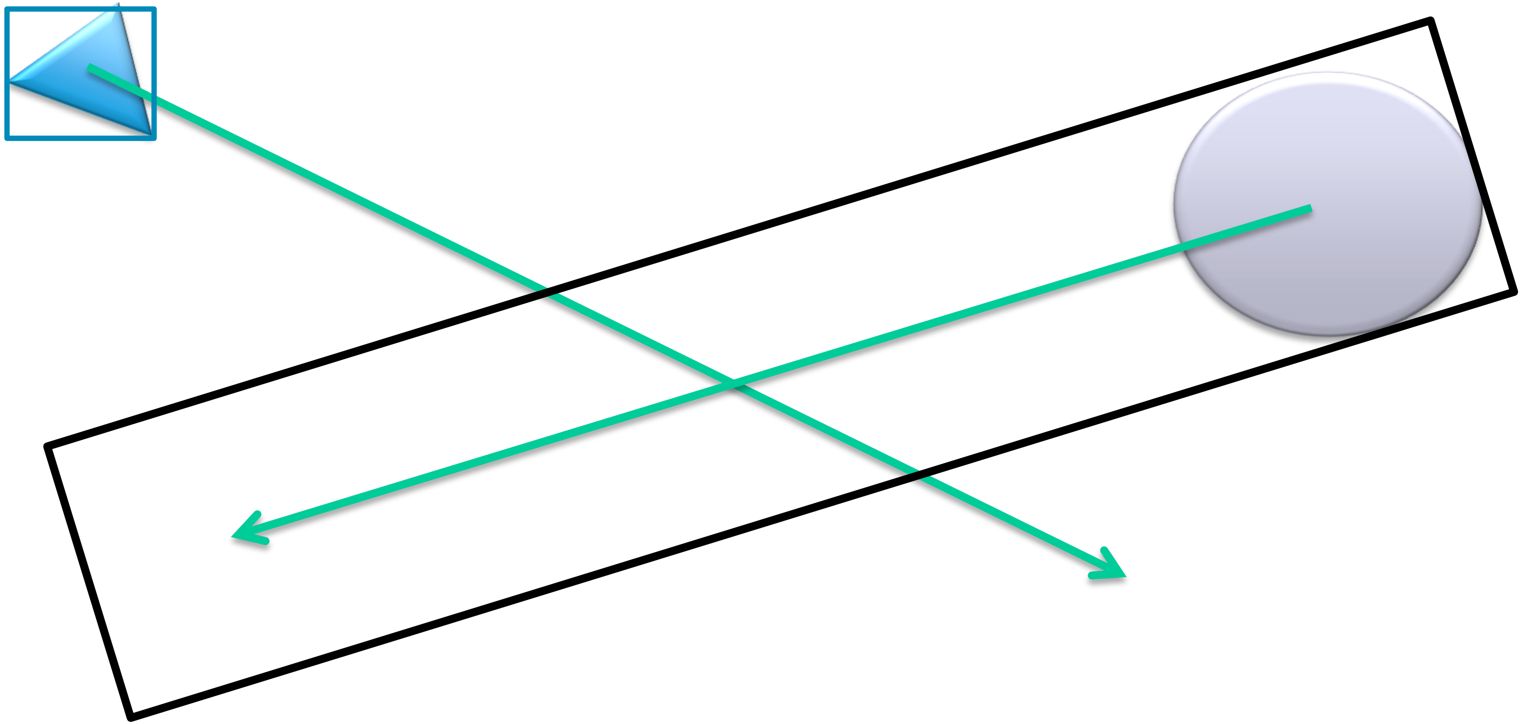
Limitation on the second impact

This issue will happen if the second obstacle is right after a first obstacle.

Solution wouldn't be suitable for some video games.

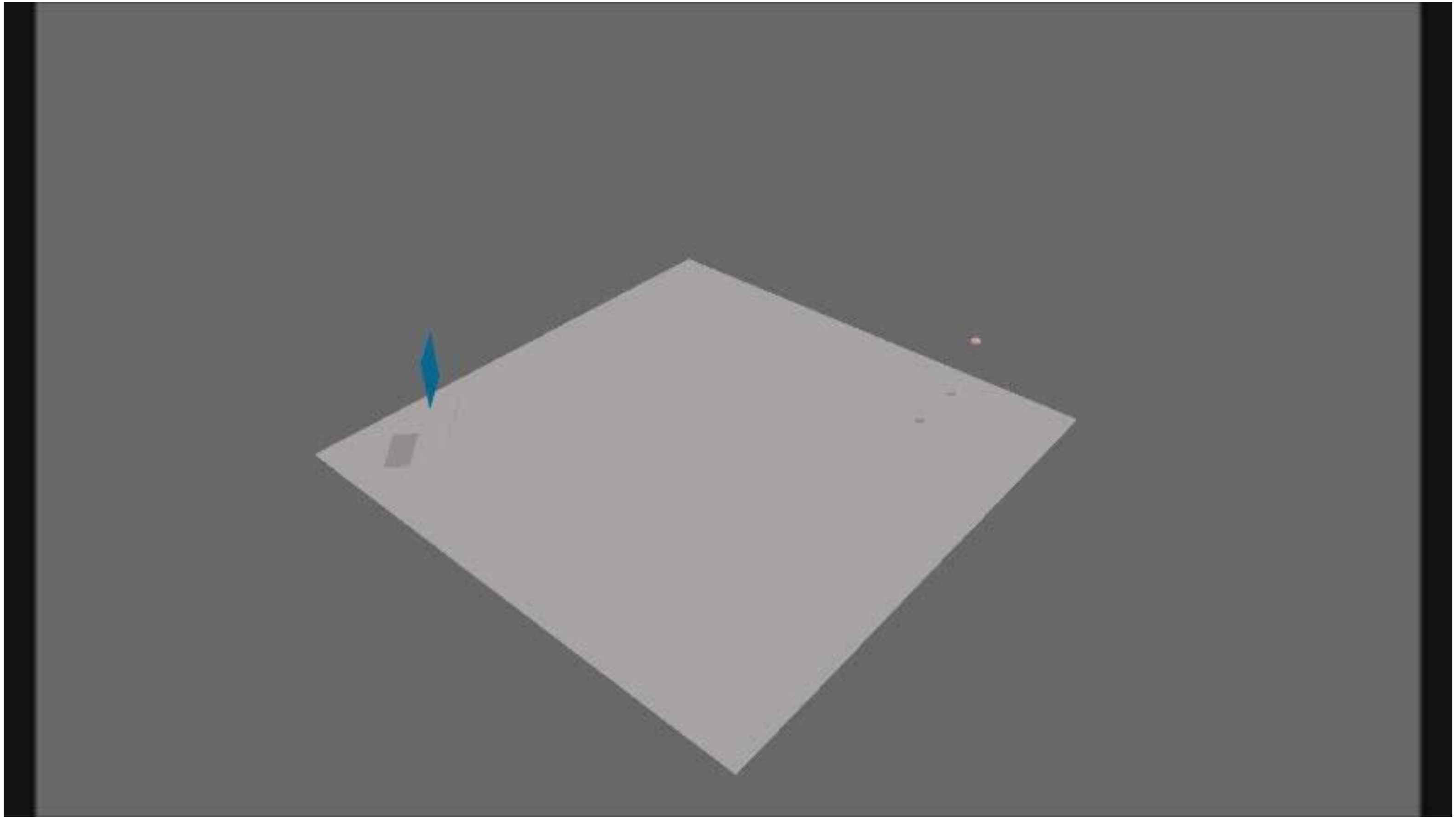


Handling several fast bodies



Handling several fast bodies

- We decided not to manage this case because it was not an issue for most of the games.
- If one body moves really fast and the other one moves slowly, the collision will be handled correctly.



Remove these limitations

- To handle these limitations, only a modification on the broad phase is needed.
- Use a bigger bounding volume, but this method:
  - Can generate unnecessary body pairs
  - Can increase CPU costs
  - Causes the ghost bug

A Different Approach for Continuous Physics

Existing approaches

Our method

Limitations

Performances

Conclusion

Continuous physics cost

Comparison between the discrete collision pipeline and the continuous physics pipeline.

Broad phase

- Segment intersection with an AABB:
  addition, min, max, cross product, select...
- More body pairs are generated.

Narrow Phase

- Distance-based full manifold collision algorithms cost about the same as traditional collision algorithms.
- More contact points are generated.
- Additional memory is used to store the contact points.

Continuous physics cost

Constraint creation

Additional data to store: previous velocities.

(For managing the restitution only.)

Solver

No additional process.

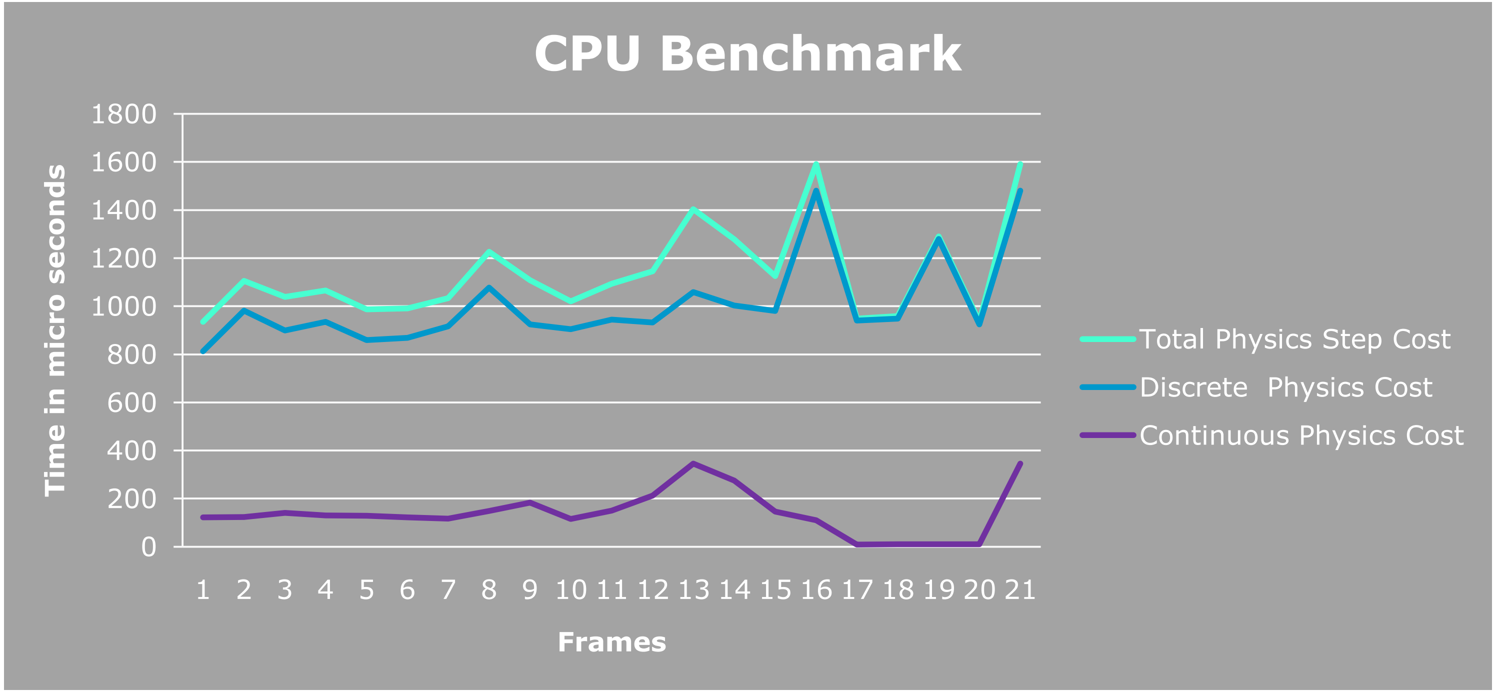
It takes more time because there are more contact point constraints to solve.

Profiling in Ghost Recon Future Soldier on Xbox 360

- Showing the profiling scene using Continuous physics
- CPU benchmark
- Memory consumption

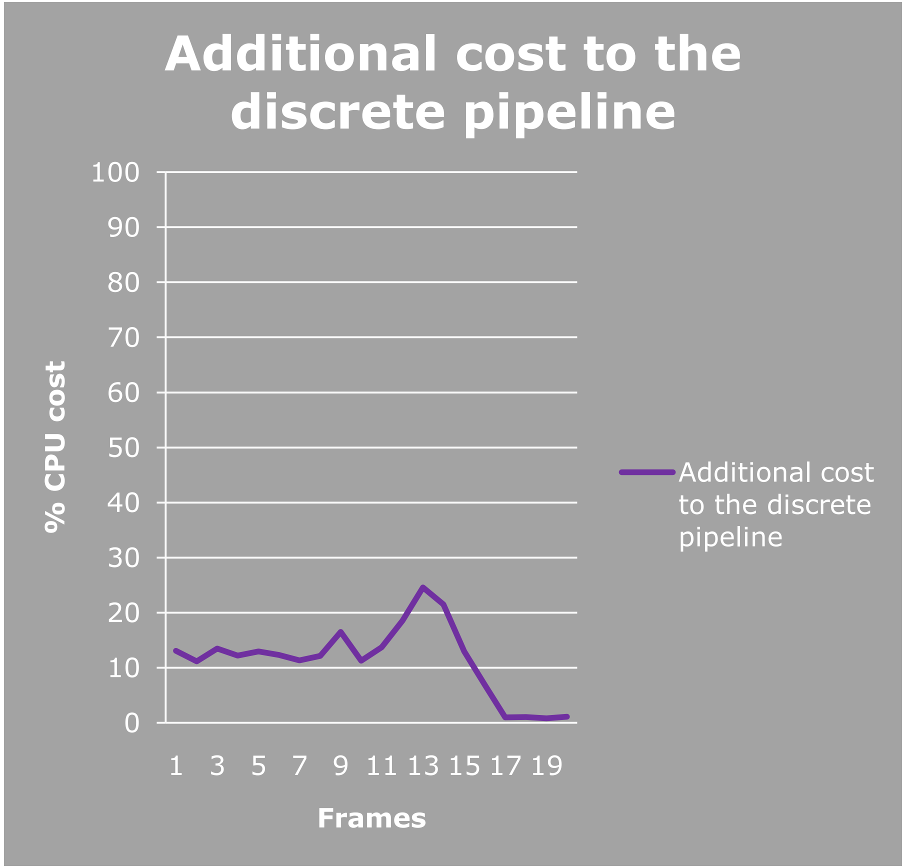

Reading KB/Sec
0.00
AI Main List Ok.
AI Secondary List Ok.
Free RAM (MB)
38.70
Frames/Sec
27
Delta Time (MS)
SENSOR
05

CPU benchmark

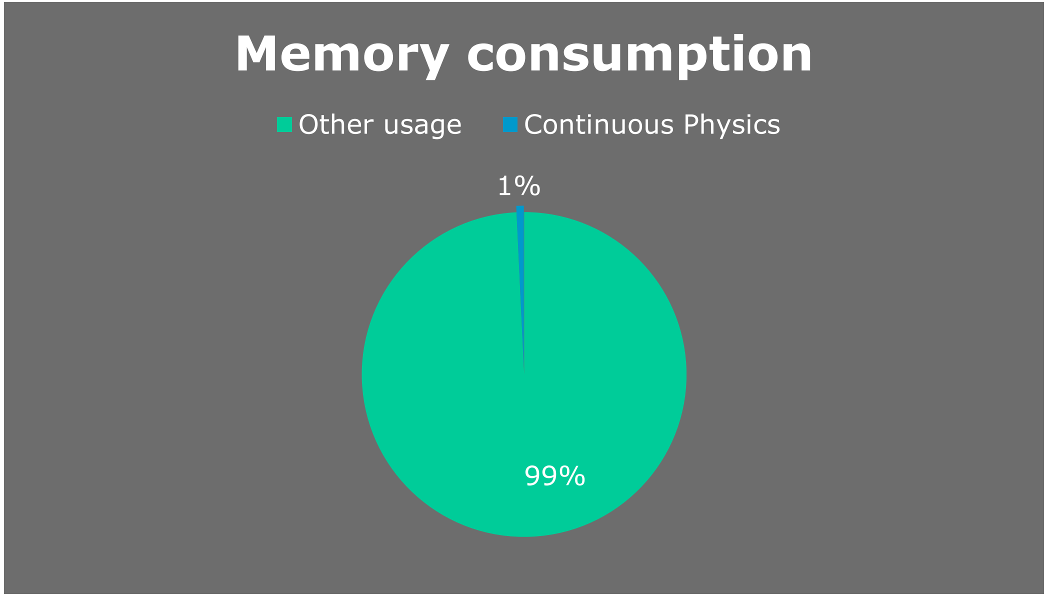


CPU benchmark

Average cost: 11.4%
Average only falling: 15.4%
Max cost: 24.5%
Min cost: 0.7%
Dynamic rigid bodies: 11
All using continuous physics



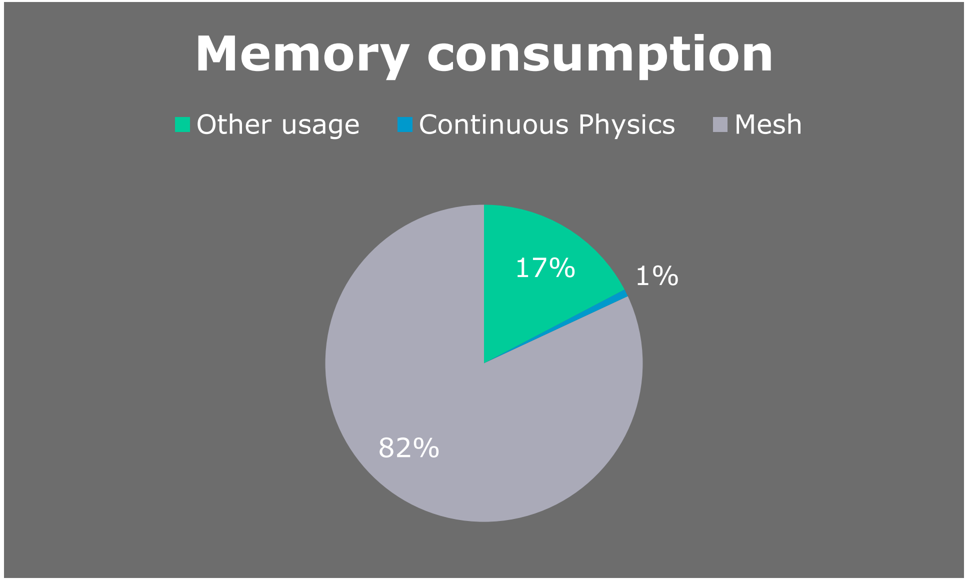
Memory consumption: compare to physics data



Other usage: 14.2 MB

Continuous Physics: 107 KB

Memory consumption: Mesh as most important data

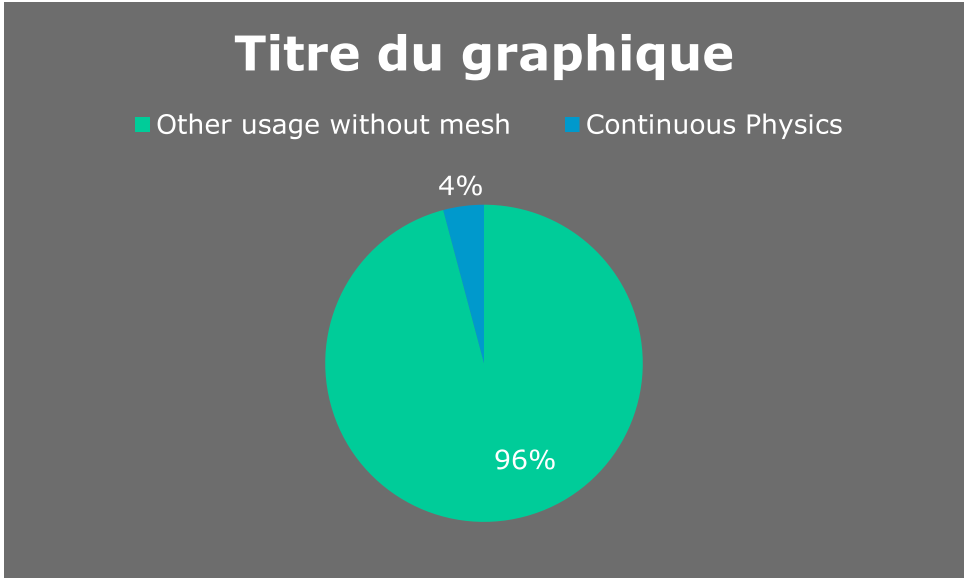


Other usage excluding meshes: 2.5 MB

Continuous Physics: 107 KB

Mesh data : 11.7 MB

Memory consumption: Comparison without mesh data



Other usage excluding meshes: 2.5 MB

Continuous Physics: 107 KB

A Different approach for Continuous Physics

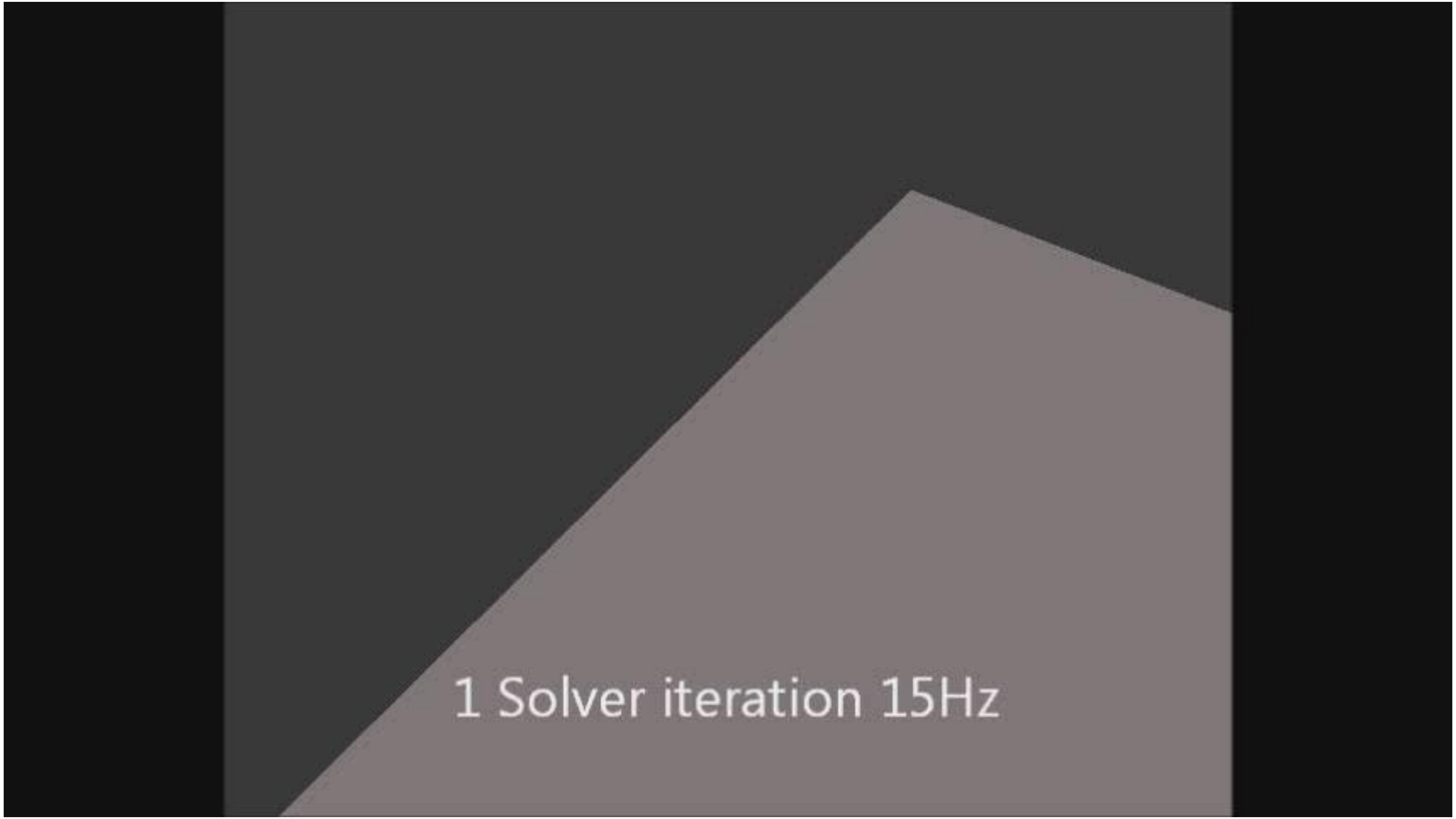
Existing approaches

Our method

Limitations

Performances

Conclusion


1 Solver iteration 15Hz

Conclusion

Low additional cost for the CPU

- No big additional process.
- Potential contact points less expensive than real contact points: no friction.
- The number of body pairs generated is more significant, so the cost increases in the entire pipeline.

Restitution is handled correctly

Conclusion

Robust: No tunnelling issues with fast rotating bodies

- Variable frame rate
- Few solver iterations

Limitations:

- Several fast bodies
- Second impact
- Solution can be improved

References

Erin Catto

Iterative Dynamics with Temporal Coherence
Box2D

Russell Smith

Constraints in Rigid Body Dynamics

Open Dynamics Engine (ODE)

Erwin Coumans

Continuous Collision Detection and Physics

Bullet

References

Gino van den Bergen

Ray Casting against General Convex Objects with Application to Continuous Collision

Dirk Gregorius

Game Physics Pearls (Gino van den Bergen)

Paul Firth

Speculative Contacts - A continuous collision engine approach

Special Thanks

UBISOFT®

MOTION Team

GHOST RECON FUTURE SOLDIER

engine room.ubi.com

powered by UBISOFT

YOUR ONE-STOP IN-DEPTH TECH BLOG