Introduction

The script ExactConvexDecomp is a MonoBehaviour script in Unity that allows you to create a convex decomposition of a 3D mesh. It takes a 3D mesh (attached to the same GameObject as the script) and generates a set of convex colliders to approximate the shape of the original mesh. It supports multiple collider types, including Oriented Box, Axis-Aligned Box, Tetrahedron, and Triangular Prism. The script can be added to a game object with a MeshFilter component to generate colliders.

Collider Types

BoxOBB (Oriented Bounding Box)

A BoxOBB is a type of bounding volume that is represented by a rectangular box that is oriented in 3D space.

Pros:

- Better representation of 3D objects compared to AABBs (Axis-Aligned Bounding Boxes) as it can be rotated to fit the shape of an object, providing a tighter fit and improved collision detection accuracy.
- Can be efficiently transformed and tested for collisions.
- Uses fast BoxColliders

Cons:

- More complex to implement than AABBs as it requires additional calculations to determine orientation.
- May not always provide the tightest fit, especially if the object has a complex shape.

Vars:

-Uses Width Variable
-Can use Centered
-Can use Inverted

BoxAABB (Axis-Aligned Bounding Box)

A BoxAABB is a type of bounding volume that is represented by a rectangular box that is aligned with the coordinate axis.

Pros:

- Simple to implement and fast to calculate collisions.
- Provides an efficient way to group and organize objects in 3D space.
- Uses fast BoxColliders

Cons:
- May not provide an accurate representation of 3D objects if the object has a complex shape.
- Can result in a larger volume than is necessary, leading to false collisions.

Tetrahedron

A Tetrahedron is a type of bounding volume that is represented by a triangular pyramid.

Pros:
- Can provide a tight fit for objects with triangular faces.
- Simple to implement and fast to calculate collisions.

Cons:
- Not suitable for objects with a large number of faces or complex shapes.
- Can result in “Bumpy”, “Jagged” collisions
- Uses slow MeshCollider

Vars:

-Uses Width Variable

Prism

A Prism is a type of bounding volume that is represented by a parallelogram with a height.

Pros:
- Can provide a tight fit for objects with rectangular faces.
- Simple to implement and fast to calculate collisions.

Cons:
- Not suitable for objects with a large number of faces or complex shapes.
- Can result in a larger volume than is necessary, leading to false collisions.
- Uses slow MeshCollider

Vars:

-Uses Width Variable
-Can Use Centered
-Can Use Inverted

Properties

The following properties can be set in the Unity Editor:

ColliderType: An enumeration that determines the type of collider to generate. The options are BoxOBB, BoxAABB, Tetrahedron, and Prism.

Width: A float that determines the width of the colliders.

InvertColliders: A boolean that inverts the direction of the colliders if set to true.

Centered: A boolean that determines if the colliders should be centered on the mesh or not.

AddRigidBody: A boolean that adds a rigid body component to the target game object if set to true.

Gravity: A boolean that enables gravity for the rigid body component if set to true.

Methods

The following methods are available in the script:

Initialize(): A method that initializes the script by setting up the mesh, vertices, and triangles.

AddColliders(): A method that generates the convex colliders on the mesh.

ResetMesh(): A method that removes the convex colliders and resets the mesh to its original state.

Usage

Add the ExactConvexDecomp script to a game object with a MeshFilter component.

In the Unity Editor, set the properties as desired.

Call the AddColliders method to generate the convex colliders on the mesh.

Optionally, call the ResetMesh method to remove the convex colliders and reset the mesh to its original state.

Variables

- ColliderType: An enumeration that allows you to choose between four different types of colliders: BoxOBB, BoxAABB, Tetrahedron, and Prism.
- width: The width of the colliders, used as the thickness of some colliders. (see collider types). The value is a float and is defined within the range of 0.001f and 0.01f.
- invertColliders: A boolean that, if set to true, inverts the direction of some colliders. (see collider types)
- centered: A boolean that, if set to true, centers the colliders around the centroid of the original mesh.
- addRigidBody: A boolean that, if set to true, adds a Rigidbody component to the target GameObject.
- gravity: A boolean that, if set to true, sets the useGravity property of the added Rigidbody component to true.
- target: The target GameObject, which should have a MeshFilter component attached to it.
- vertices: An array of the vertices of the original mesh.
- triangles: An array of the triangles of the original mesh.
- convexColliders: A GameObject that acts as a parent for the set of generated colliders.
- unwantedColliders: An array of MeshColliders that were present on the target GameObject before the script was run.