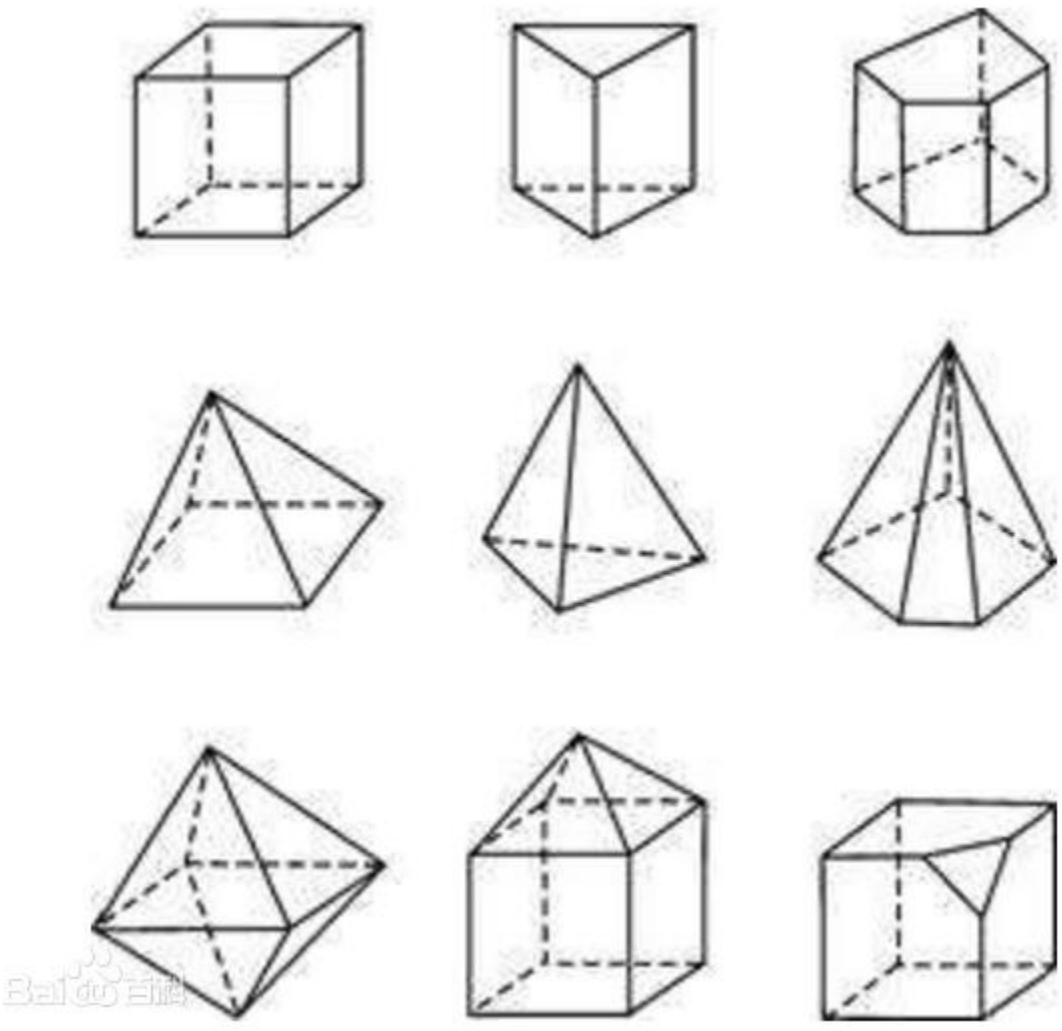
郭隆邦 2023-10-11

10. 凸多面体ConvexPolyhedron

对于不同形状的碰撞体Body，需要选择CannonJS不同的几何体表示，比如用球体 `Sphere` 表示一个乒乓球的形状，用长方体 `Box` 表示一个箱子的形状。

```js
const body = new CANNON.Body({
    shape: new CANNON.Box(new CANNON.Vec3(1, 2, 1))
});
```

这节课给大家介绍一个新的CannonJS几何类 `ConvexPolyhedron` ，`ConvexPolyhedron` 可以用来表示不同外形的凸多面体 。



查看gltf模型几何体结构

```js
const gltf = await loader.loadAsync("../凸多面体.glb")
const mesh = gltf.scene.getObjectByName('多面体');//获取凸多面体网格模型
mesh.position.y = 5;
console.log('mesh.geometry',mesh.geometry);
```

提醒：学下面内容之前，确保你已经掌握前面基础课程介绍的几何体BufferGeometry 。

凸多面体 `ConvexPolyhedron`

凸多面体 `ConvexPolyhedron` 三角形索引和顶点位置数据的格式如下

```js
//凸多面体数据格式：三角形顶点位置
const vertices = [
    new CANNON.Vec3(0, 0, 0),
    new CANNON.Vec3(0, 1, 0),
    ...
];
//凸多面体数据格式：三角形面的索引值
const faces = [
    [0,1,2],//三角形1顶点索引值
    [3,4,5] //三角形2顶点索引值
    ...
];
```

```js
const shape = new CANNON.ConvexPolyhedron({
    vertices: vertices,
    faces: faces
});
// 简化写法
const shape = new CANNON.ConvexPolyhedron({ vertices, faces });
const body = new CANNON.Body({
    shape: shape
});
```

gltf凸多面体转CannonJS凸多面体

获取gltf模型的三角形顶点数据，转化为CannonJS凸多面体`ConvexPolyhedron`的三角形顶点数据。

```js
const vertices = [];//所有三角形顶点位置数据
const faces = [];//所有三角形面的索引值
const pos = mesh.geometry.attributes.position;
for (let i = 0; i < pos.count; i++) {
    const x = pos.getX(i);
    const y = pos.getY(i);
    const z = pos.getZ(i);
    vertices.push(new CANNON.Vec3(x, y, z));
}
const index = mesh.geometry.index.array;
for (let i = 0; i < index.length; i += 3) {
    const a = index[i];
    const b = index[i + 1];
    const c = index[i + 2];
    faces.push([a, b, c]);
}
// CannonJS的凸多面体ConvexPolyhedron
const shape = new CANNON.ConvexPolyhedron({ vertices, faces });
// 物理凸多面体
const body = new CANNON.Body({
    shape: shape
});
```

注意：课程案例gltf模型有顶点索引数据`geometry.index`，不过有些模型`geometry.index`可能没有任何数据。如果没有index数据，转化代码和上面稍微有区别。

```js
const pos = geometry.attributes.position;
for (let i = 0; i < pos.count; i++) {
    const x = pos.getX(i);
    const y = pos.getY(i);
    const z = pos.getZ(i);
    vertices.push(new CANNON.Vec3(x, y, z));
}
for (let i = 0; i < pos.count; i = +3) {
    faces.push([i, i + 1, i + 2]);
}
```

注意：另一方面就是网格模型的位置、角度、缩放属性，也会对转化代码也会有影响，后面遇到具体问题具体分析。

← 9. 练习题-外部gltf箱子模型