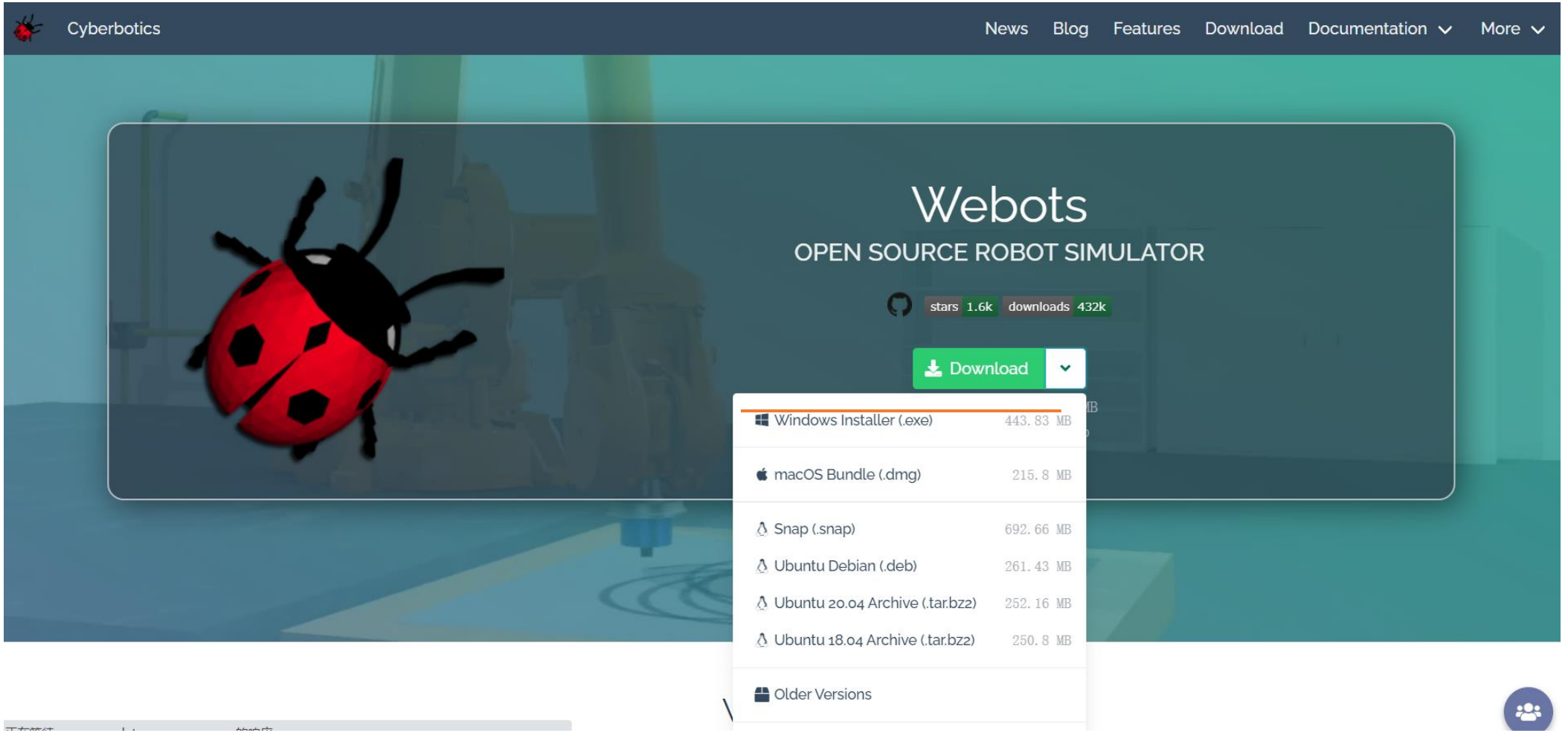
使用webots搭建麦轮小车

2021.9.22

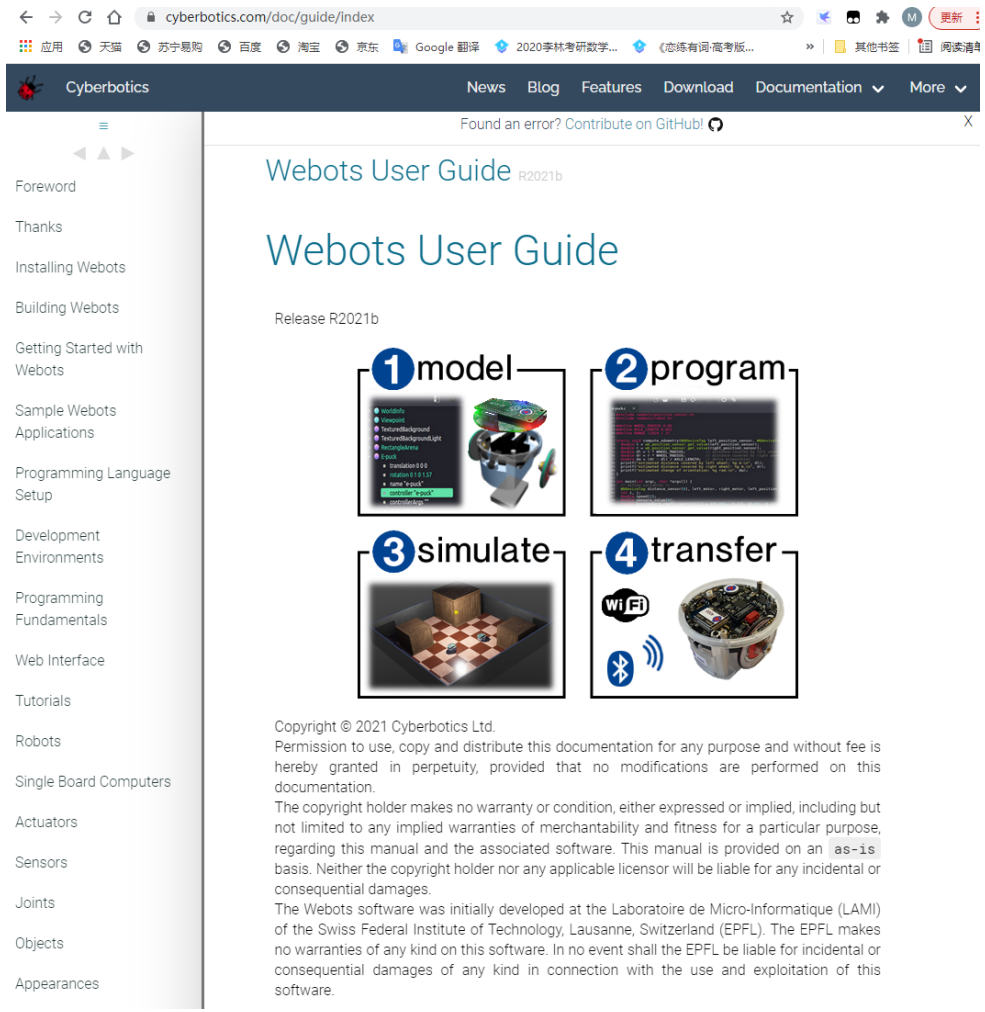
1.1 安装webots



官网：https://www.cyberbotics.com/

安装路径一定不能有中文和空格，因为之后编程要调用webots的函数，makefile认不出来

1.2 参考——官方文档



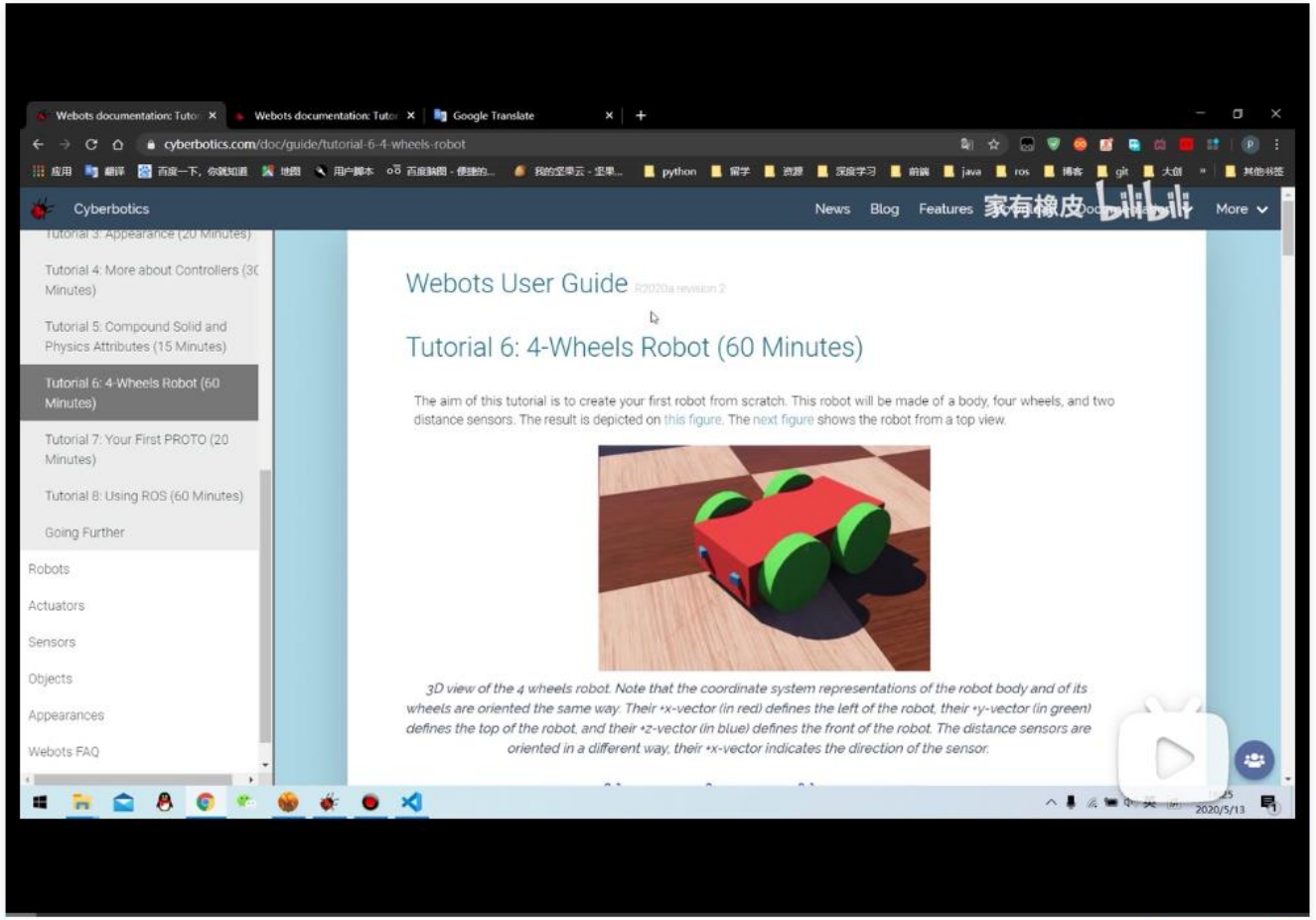
官方文档：https://www.cyberbotics.com/doc/guide/index

要科学上网才能看，可以开个自动翻译，方便多了。
我们统一使用c++ API

1.2 参考——制作小车

活动作品 webots-超详细入门教程（2020）

1.7万播放 · 98弹幕 2020-04-05 02:05:35



元宵不眨眼 发消息

所求皆如愿，所行化坦途

＋ 关注 356

弹幕列表 展开

视频选集 (9/10) 自动连播

P1 0 introduction 01:04

P2 1p 软件界面和简单模型仿真 08:09

P3 2p nodes简介 02:46

P4 3p tutorial1 Your First Simulation in We… 04:51

P5 4p tutorial 2: Modification of the Enviro… 06:13

P6 5p Tutorial 3 Appearance 01:14

P7 6p Tutorial 4 More about Controllers 12:52

P8 7p Tutorial 5 Compound Solid and Phy… 07:06

P9 8p Tutorial 6 4-Wheels Robot 17:45

P10 9p Tutorial 7 Your First PROTO 04:25

B站教程：https://www.bilibili.com/video/BV11V411f7ko?p=9

1.2 参考——制作小车

Webots学习笔记（二）---创建四轮小车模型

原创 Exploer_ 2020-04-21 23:52:35 1643 收藏 7 版权

分类专栏： Webots学习 文章标签： webos

Webots... 专栏收录该内容 1 订阅 4 篇文章 订阅专栏

四轮小车模型

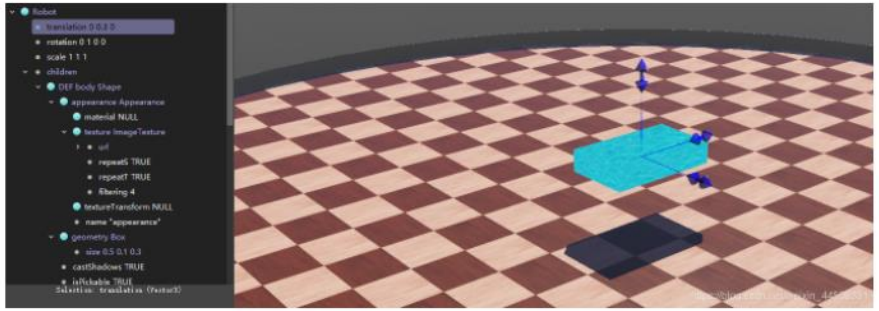
本文将创建如下一个四轮小车模型

模型建立

（一）新建模型

（1）建立车体

- 在场景树中创建Robot节点
- 在children下创建第一个Shape节点，命名为body,设置颜色纹理，并在geometry节点添加几何属性，这里我们选择Box，并且设置车体大小，我这里设置（x=0.5 y=0.1 z=0.3）m,将其在y上坐标设置高一些，方便后面加轮子。
  现在我们得到这样一个物体：



（2）建立第一个轮子

- 在children下创建一个HingeJoints节点,命名为left_front_wheel，该节点下会出现三个节点-jointParameters device 以及endPoint
- 在jointParameters 节点下创建子节点 HingeJointParameters，该节点将定义旋转原点(anchor)以及旋转轴(axis)。
  (点击查看–>可选显示–>show joint Axes ，可以显示出旋转轴，方便我们调整旋转轴及旋转原点)
  axis是表示电机旋转轴的平行向量，anchor表示旋转轴相对于父节点的坐标偏移量

https://blog.csdn.net/weixin_44508381/article/details/105669173

1.2 参考——麦克纳姆轮的制作

ROS联合Webots之麦克纳姆轮篇-搭建麦轮底盘

原创 锡城筱凯 2021-06-10 00:24:04 282 收藏 5 版权

分类专栏：▶机器人仿真 ▶机器人操作系统ROS 文章标签：ROS Webots

▶... 同时被 2 个专栏收录 2 订阅 27 篇文章 订阅专栏

ROS联合Webots之麦克纳姆轮篇-搭建麦轮底盘

ubuntu版本：20.04
webots版本：2021a
ros版本：noetic

0.前言

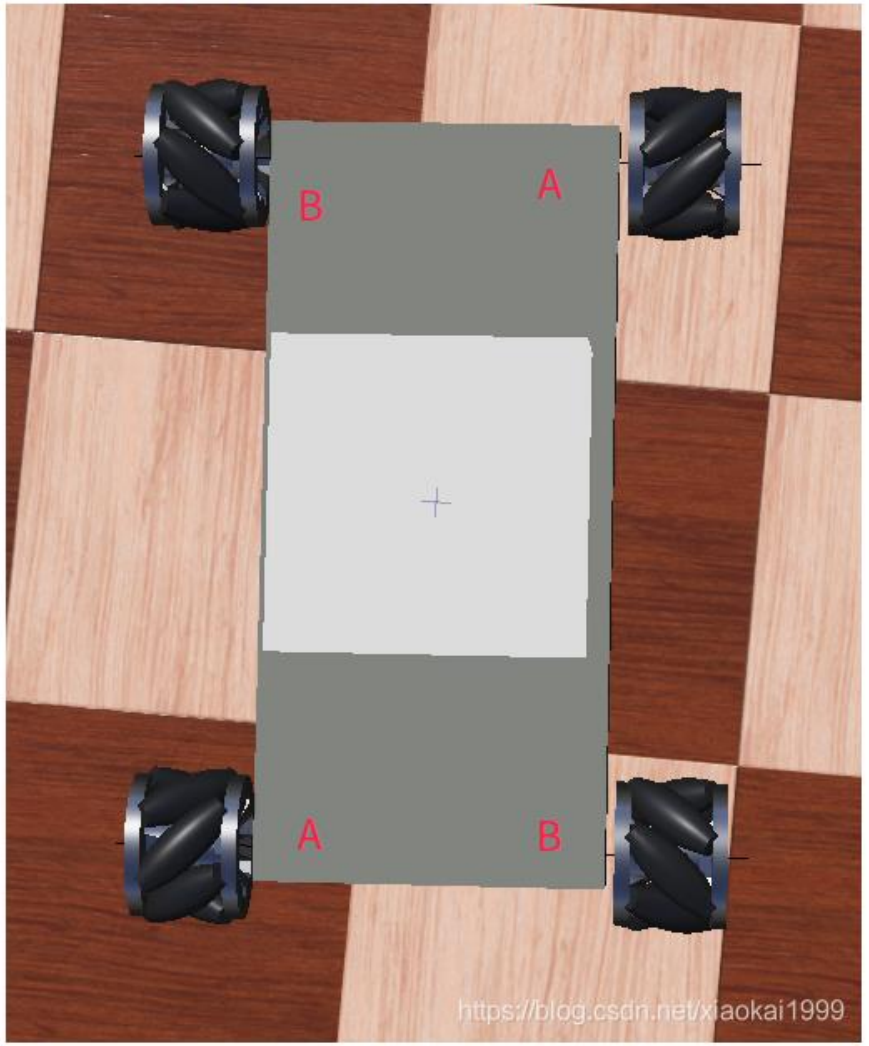
之前笔者出过ROS联合webots开发教程，在教程中使用的是双轮差动底盘模型，今天笔者将带给笔者麦克纳姆轮的使用教程。

1.模型选取

在Webots中，官方提供了kuka公司推出的youbot机器人作为麦克纳姆轮的案例

- 实物机器人如下所示：

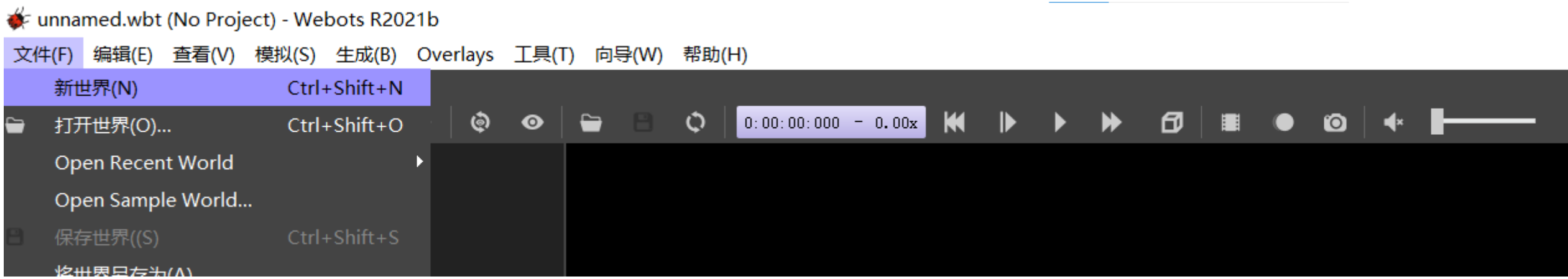
youbot机器人底盘上的麦克纳姆轮为A-B-A-B配置，如下图所示：



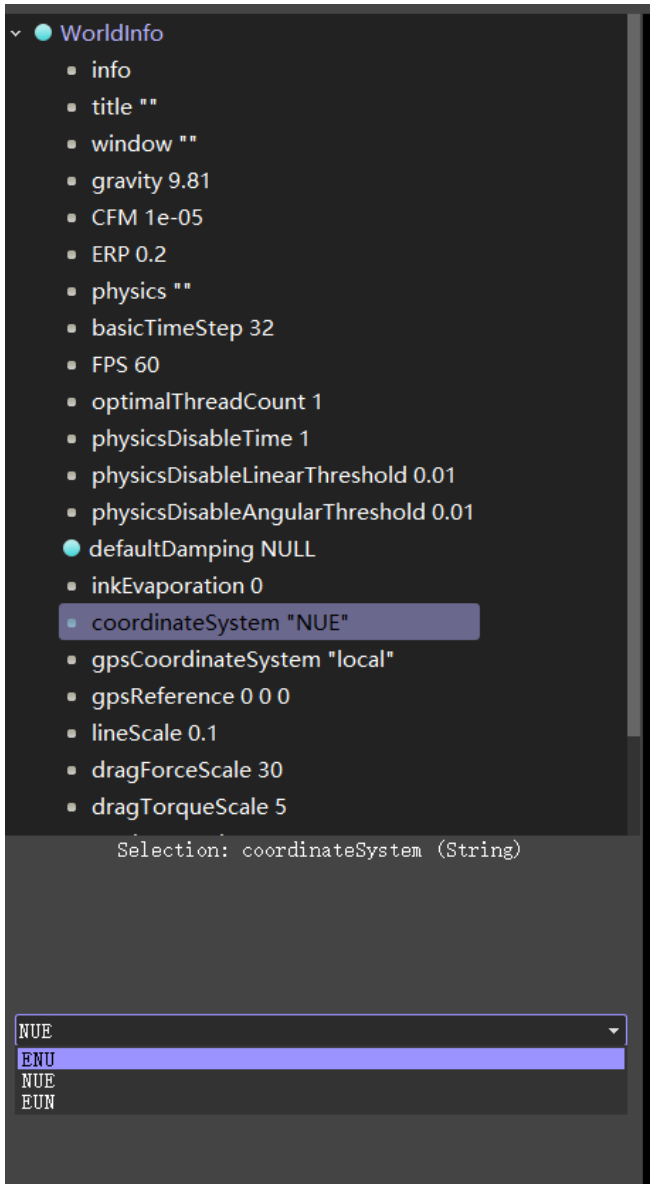
将四个轮子复制进自己的机器人中。

https://blog.csdn.net/xiaokai1999/article/details/117757052

2.1打开新世界

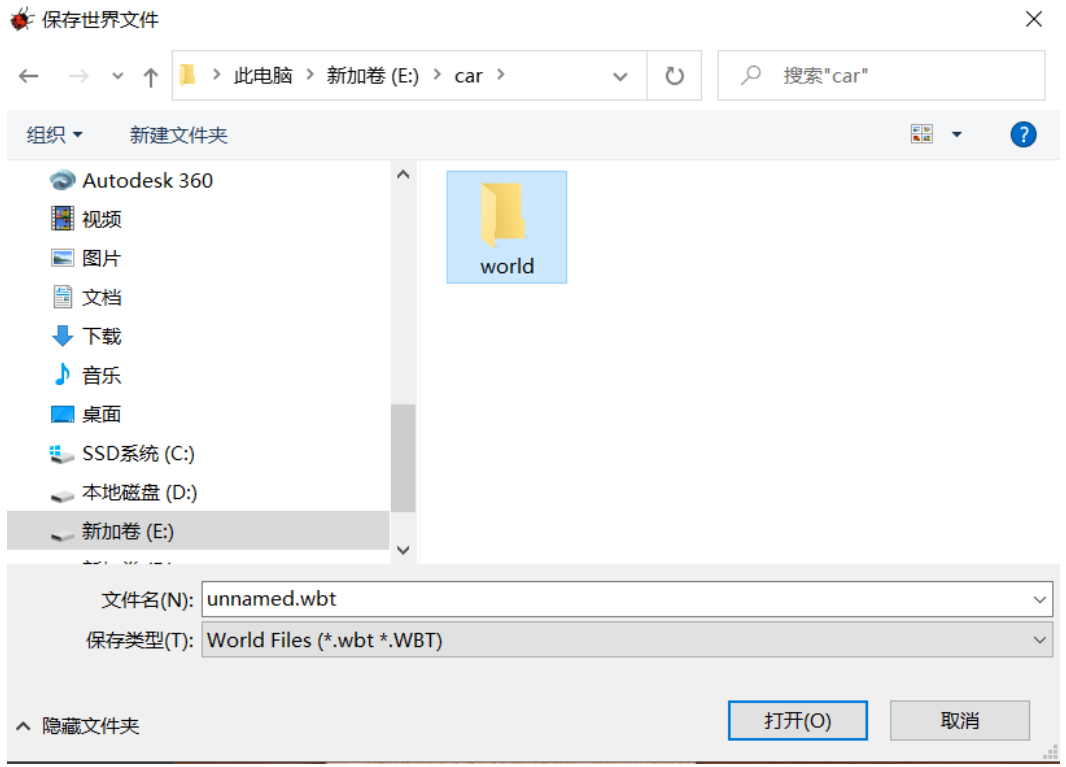

unnamed.wbt (No Project) - Webots R2021b
文件(F)
编辑(E)
查看(V)
模拟(S)
生成(B)
Overlays
工具(T)
向导(W)
帮助(H)
新世界(N) Ctrl+Shift+N
打开世界(O)... Ctrl+Shift+O
Open Recent World
Open Sample World...
保存世界((S) Ctrl+Shift+S
0:00:00:000 - 0.00x

2.2 修改坐标系



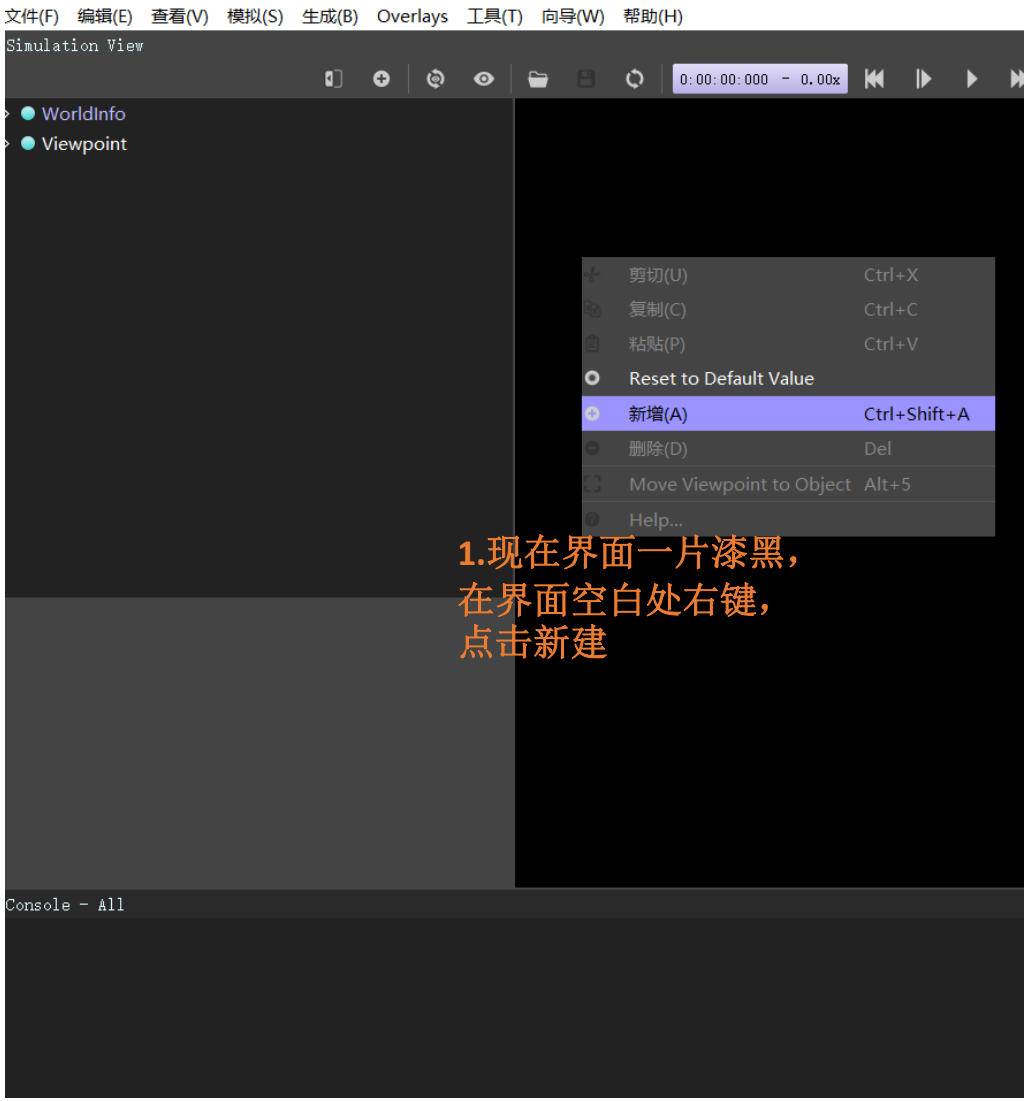
为了更加符合直觉我们将坐标系改成ENU，就是z轴指向天空的右手坐标系（就是重力是z轴负方向了，原来是y轴负方向。。。。）。

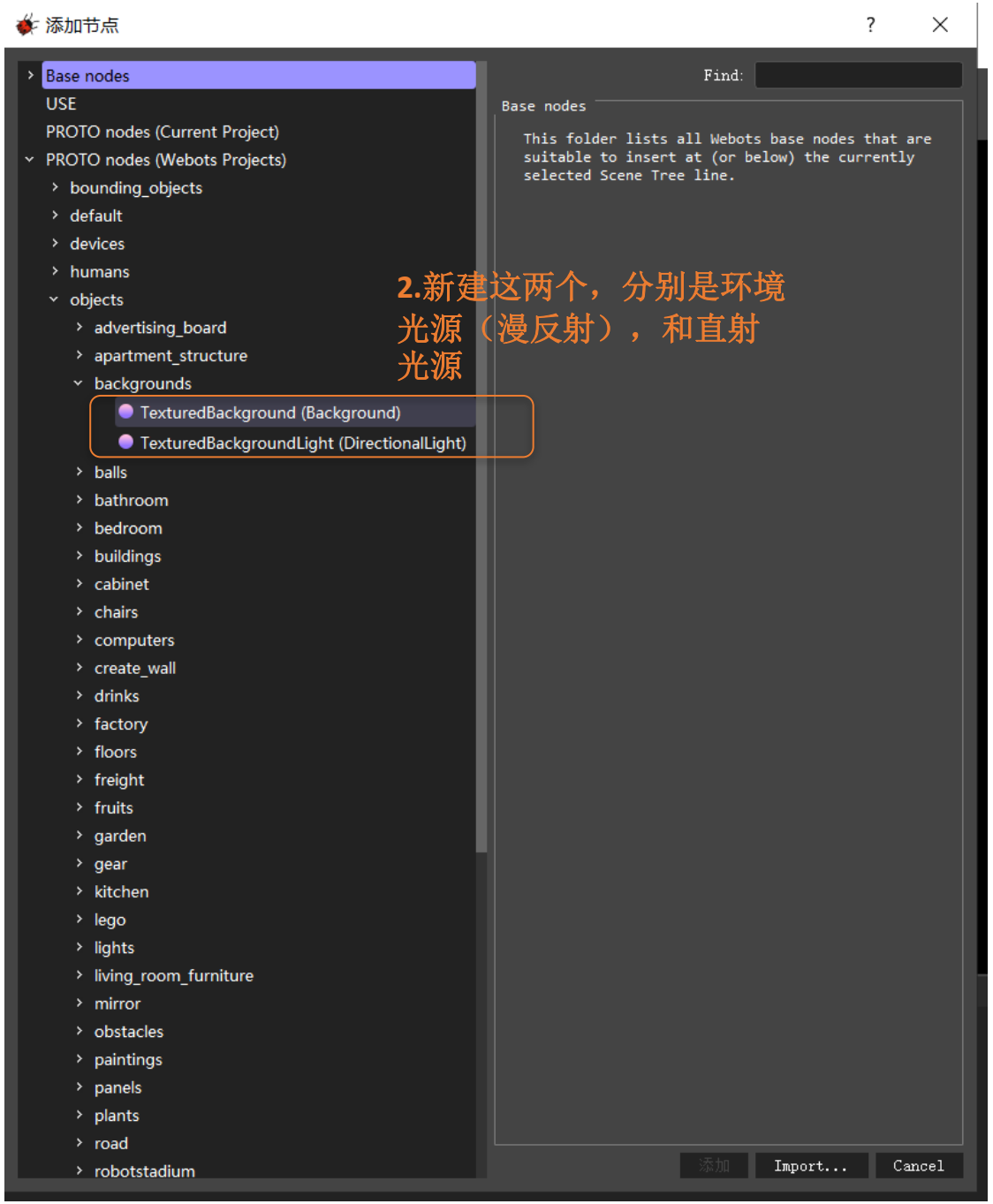
需要保存后重启一下软件(一定要记住你保存的位置，最好是新建一个文件夹叫car的文件夹，下面再建个world，把这个wbt文件放进去，这样条理清晰，尤其是之后加入控制代码的时候)



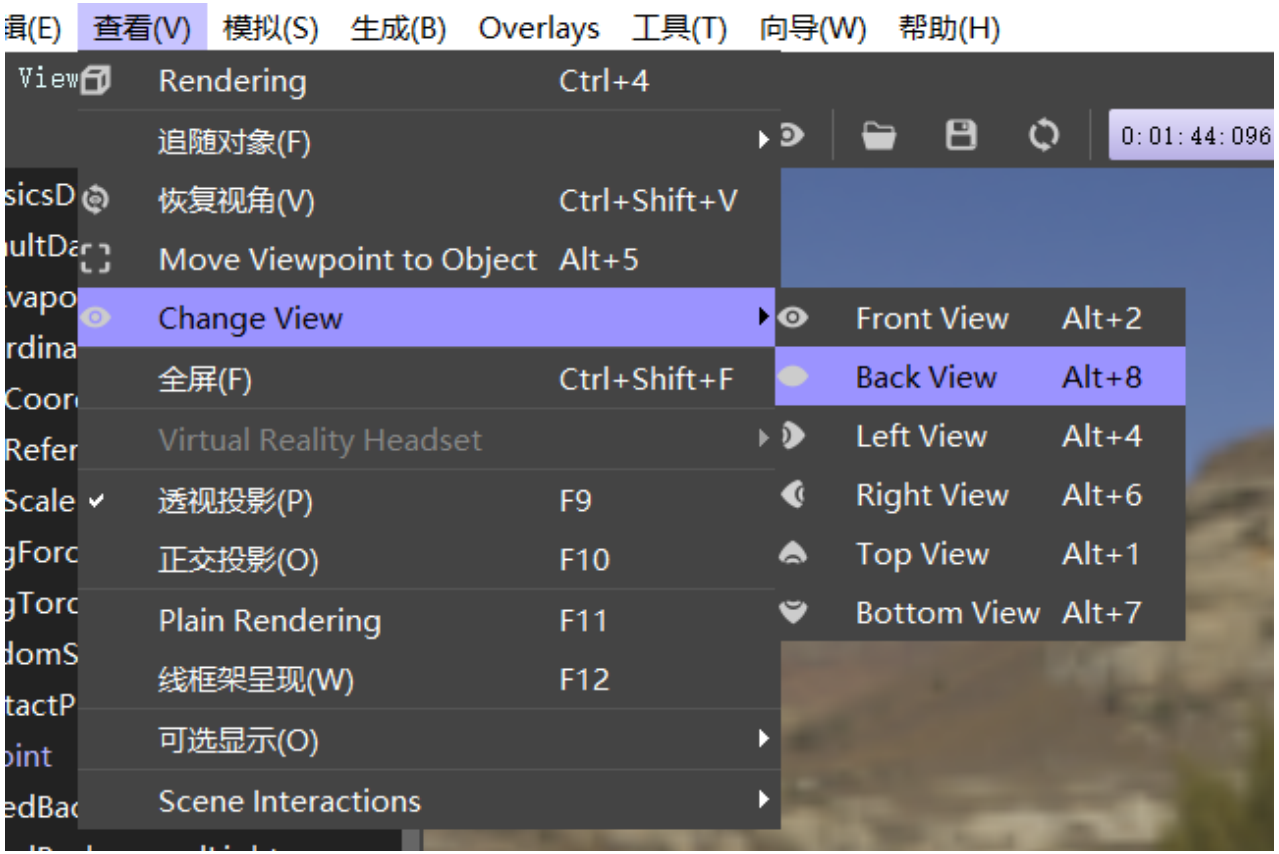
Ps:只有Webots2021才能使世界坐标系z轴向上

2.3 添加直射和漫反射光源



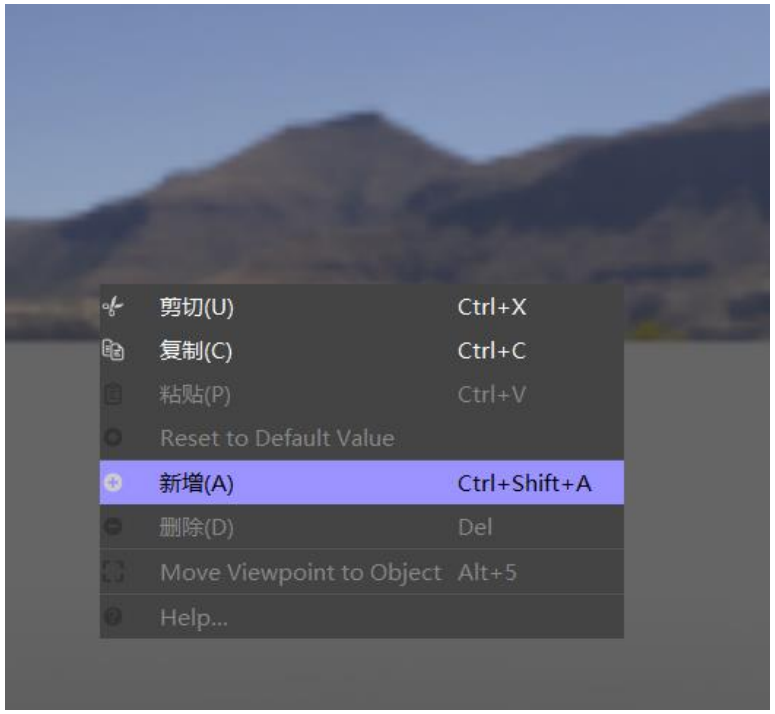


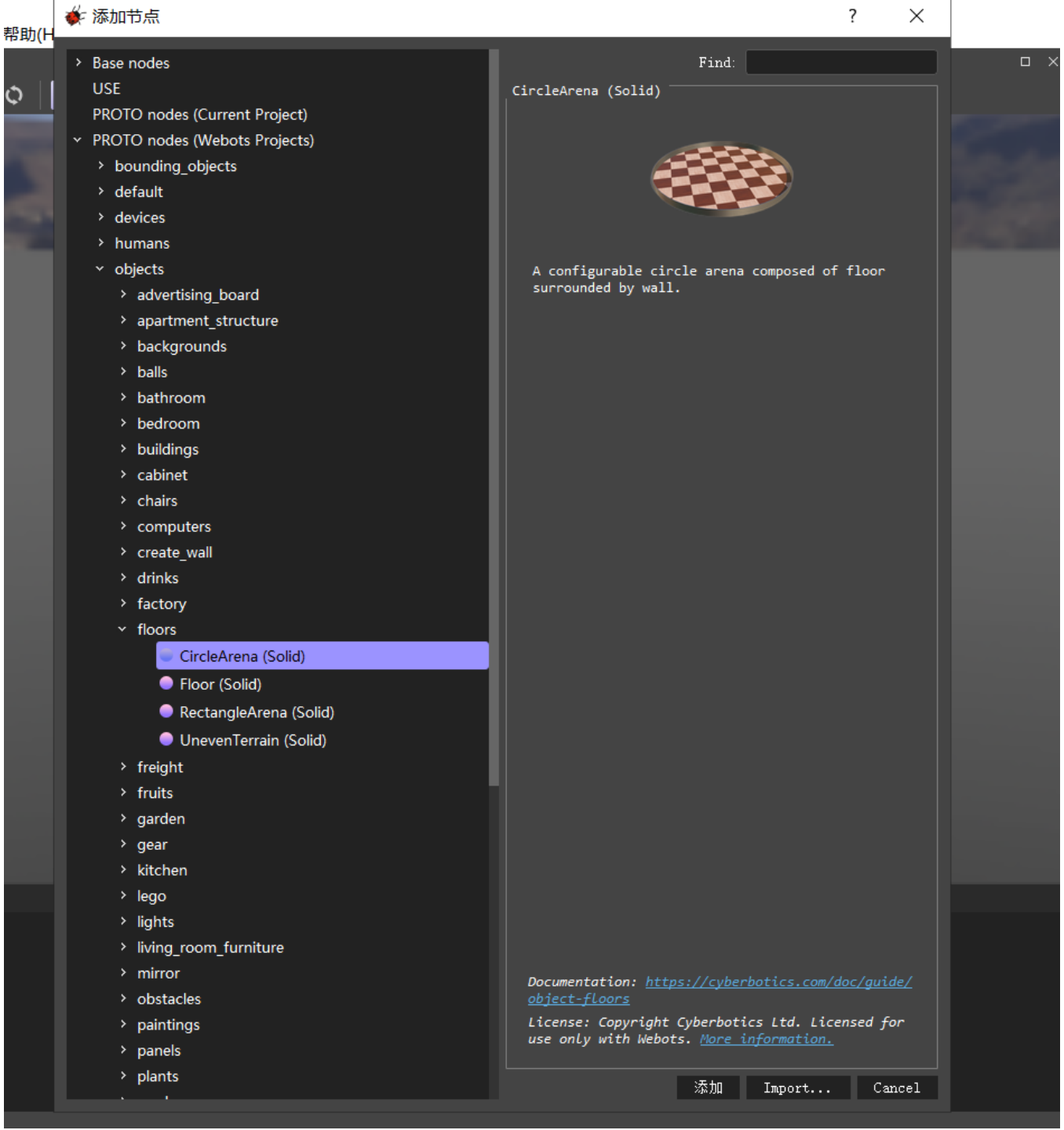
上帝说：“要有光！”

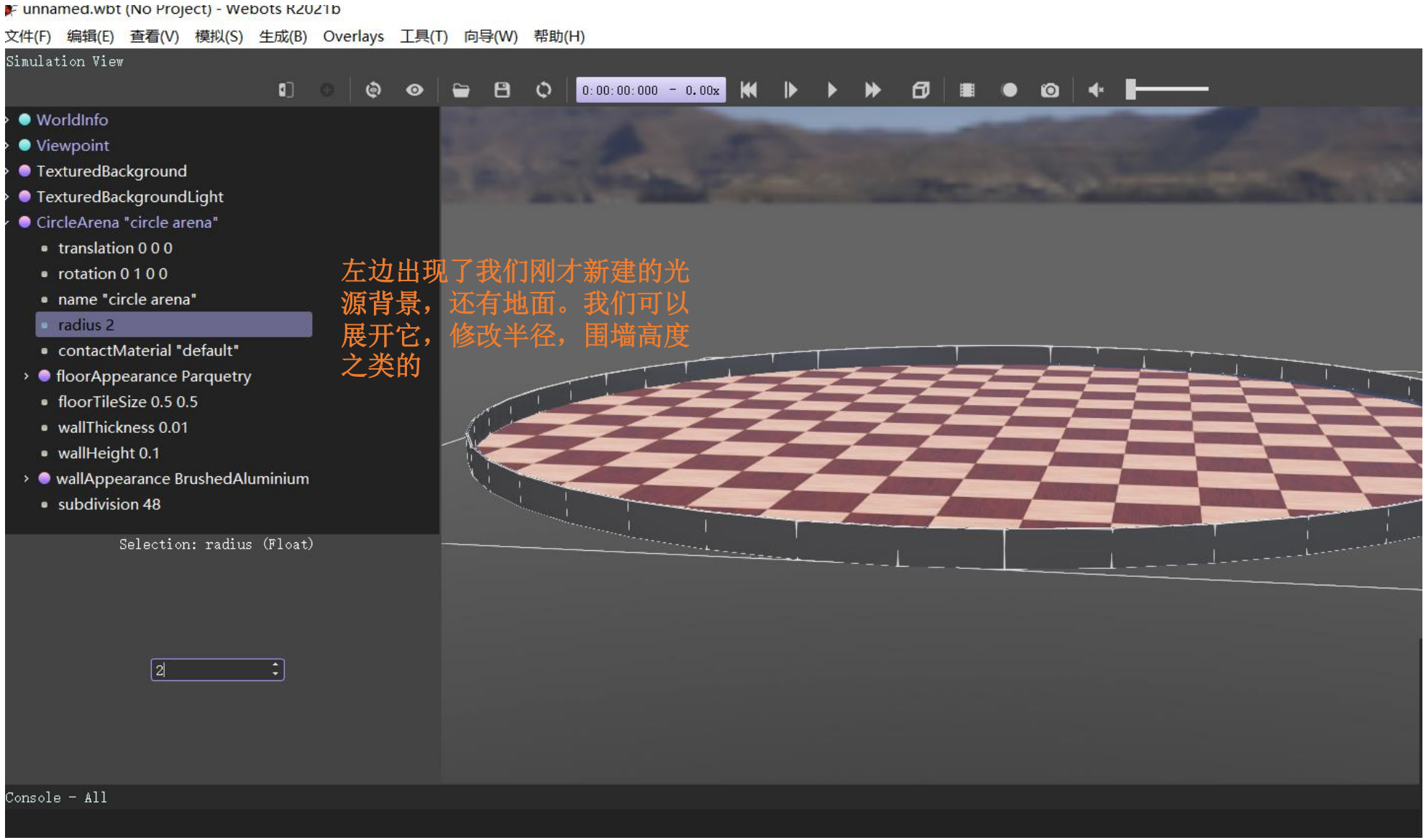


如果还是一片漆黑，大概率是视角的问题，可以这样调一下

2.4 添加地面

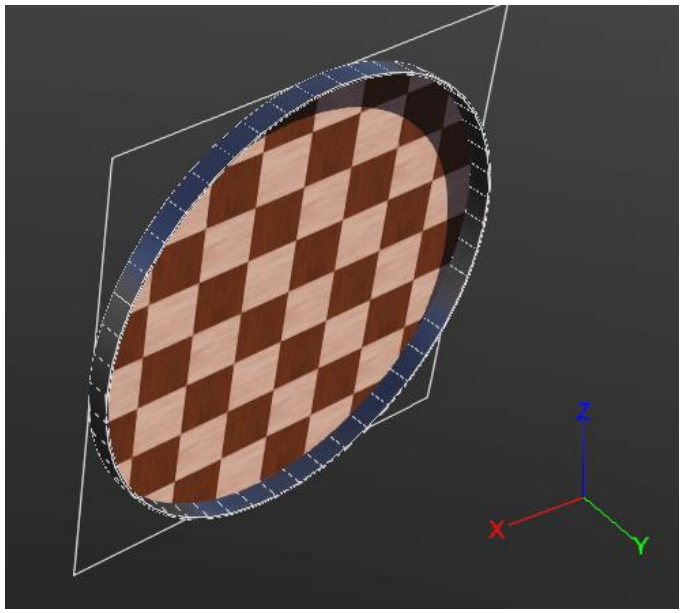

剪切(U) Ctrl+X
复制(C) Ctrl+C
粘贴(P) Ctrl+V
Reset to Default Value
新增(A) Ctrl+Shift+A
删除(D) Del
Move Viewpoint to Object Alt+5
Help...

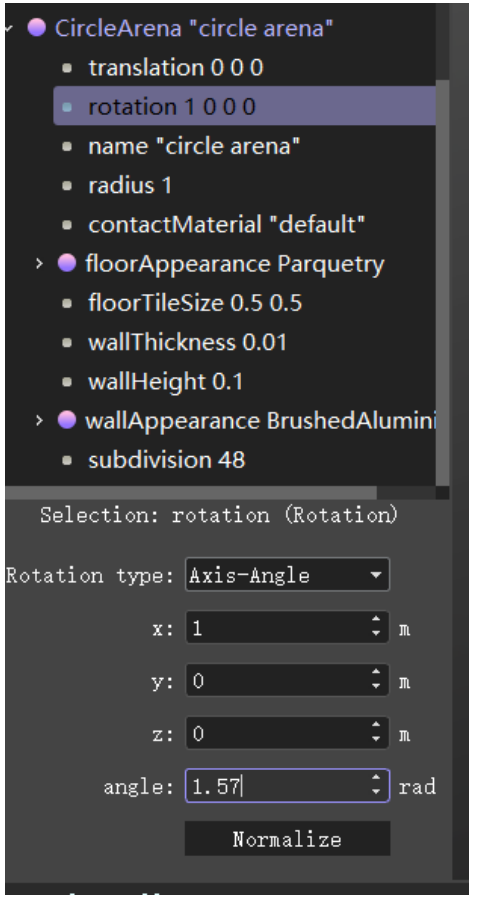

帮助(H
添加节点
Base nodes
USE
PROTO nodes (Current Project)
PROTO nodes (Webots Projects)
bounding_objects
default
devices
humans
objects
advertising_board
apartment_structure
backgrounds
balls
bathroom
bedroom
buildings
cabinet
chairs
computers
create_wall
drinks
factory
floors
CircleArena (Solid)
Floor (Solid)
RectangleArena (Solid)
UnevenTerrain (Solid)
freight
fruits
garden
gear
kitchen
lego
lights
living_room_furniture
mirror
obstacles
paintings
panels
plants
Find:
CircleArena (Solid)
A configurable circle arena composed of floor surrounded by wall.
Documentation: https://cyberbotics.com/doc/guide/object-floors
License: Copyright Cyberbotics Ltd. Licensed for use only with Webots. More information.
添加
Import...
Cancel



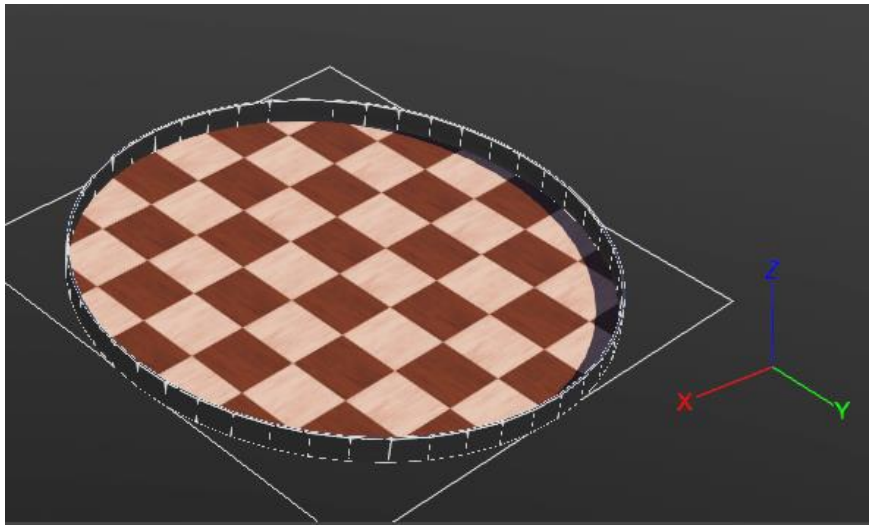
左边出现了我们刚才新建的光源背景，还有地面。我们可以展开它，修改半径，围墙高度之类的

我们可以看一下世界坐标系

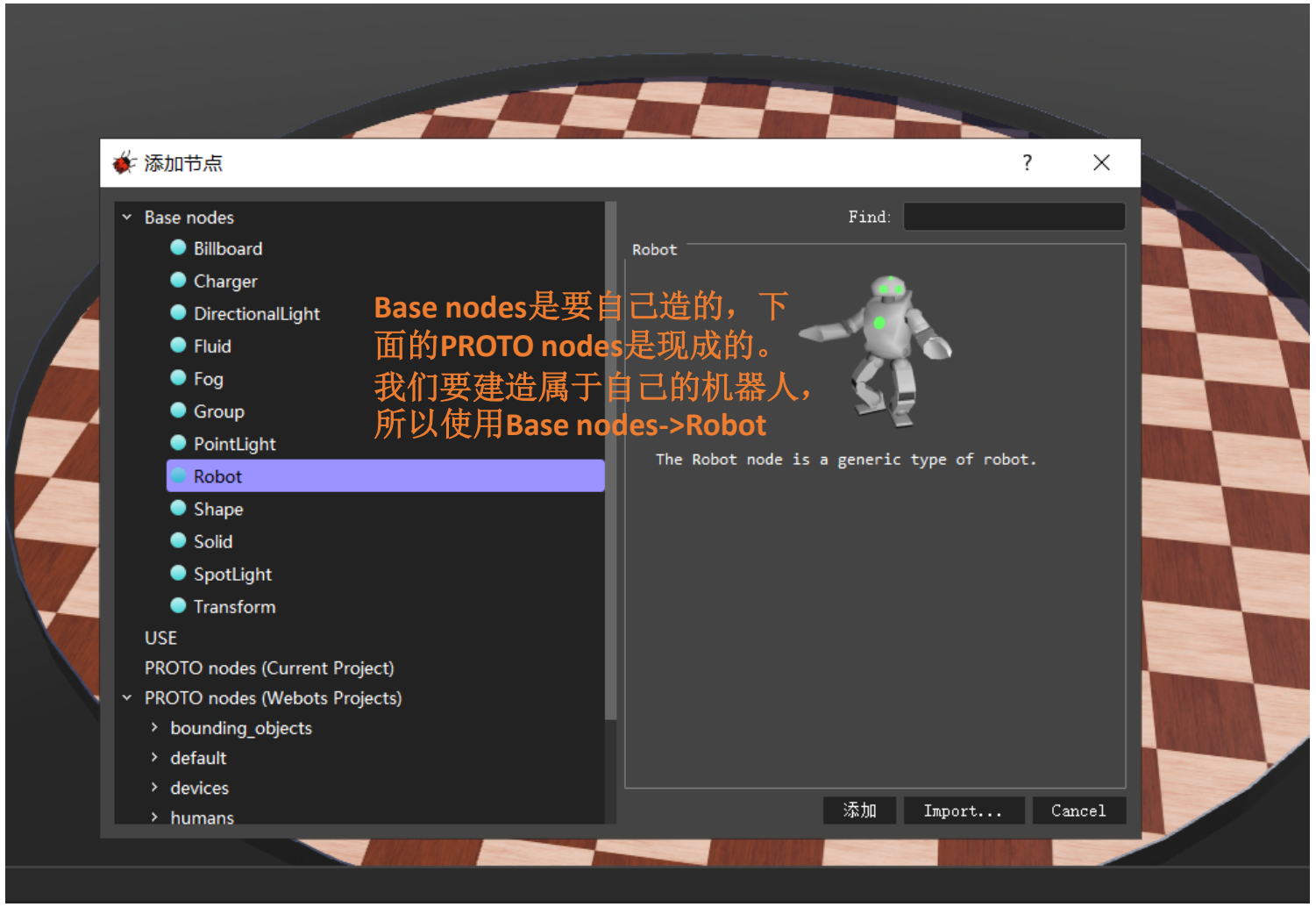




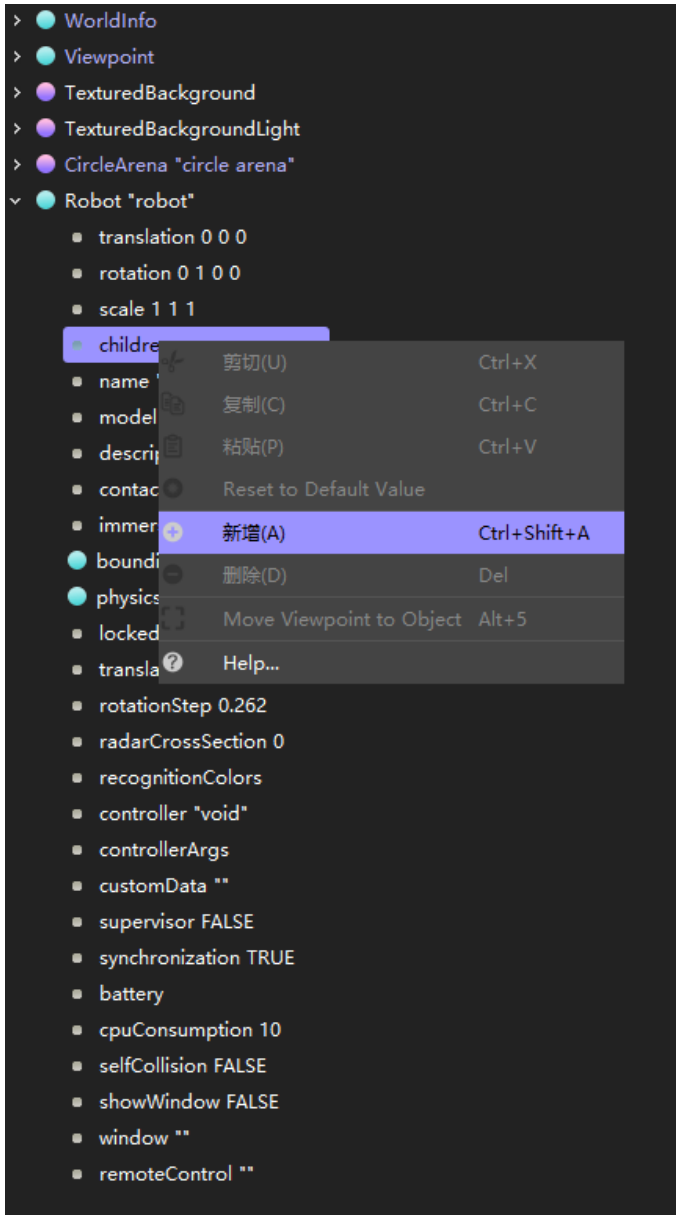
因为刚才改了坐标系，新建的地面还以为y轴负方向才是重力方向，我们把它转一下转 1.5707963267949

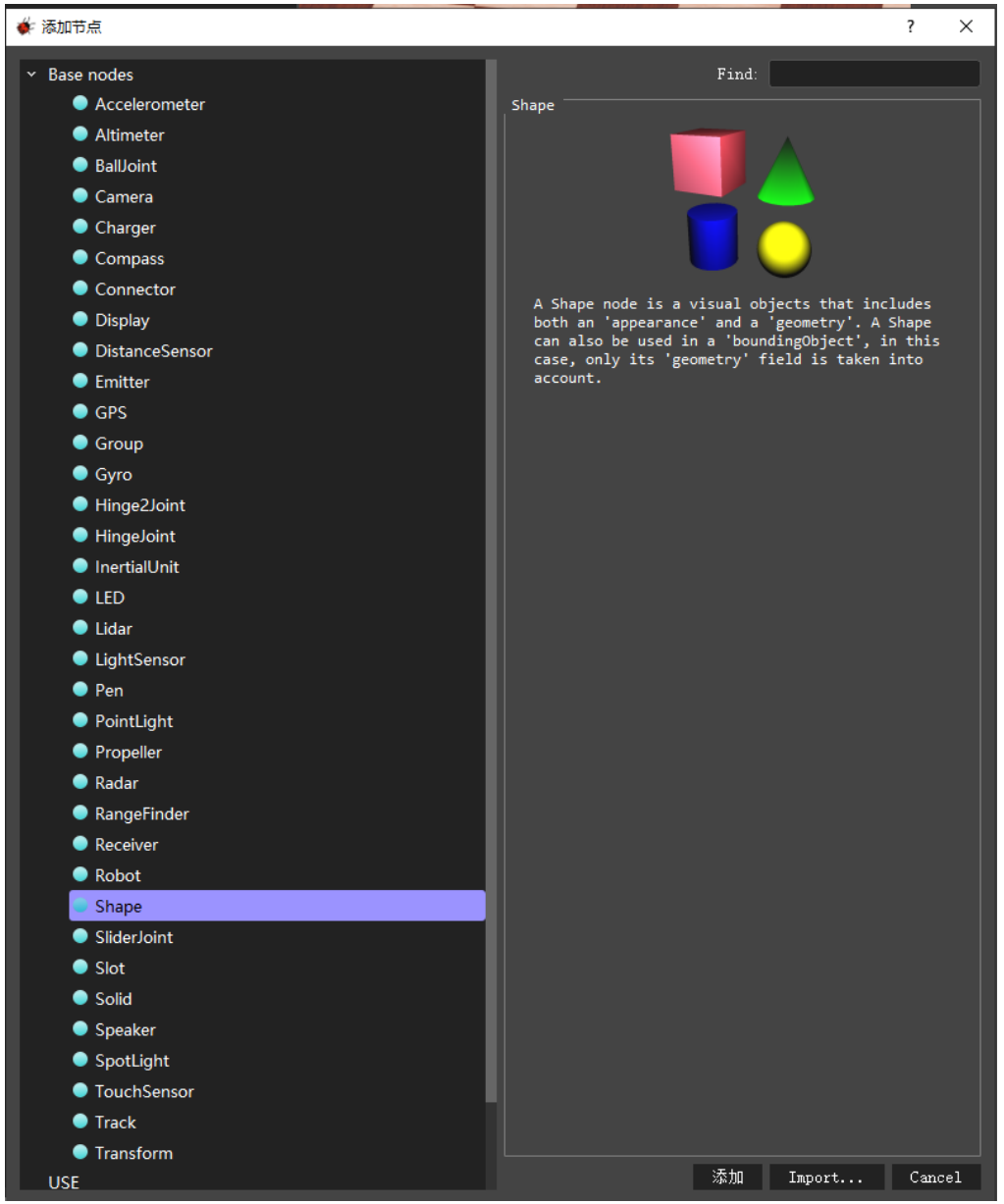


3.1 添加机器人节点

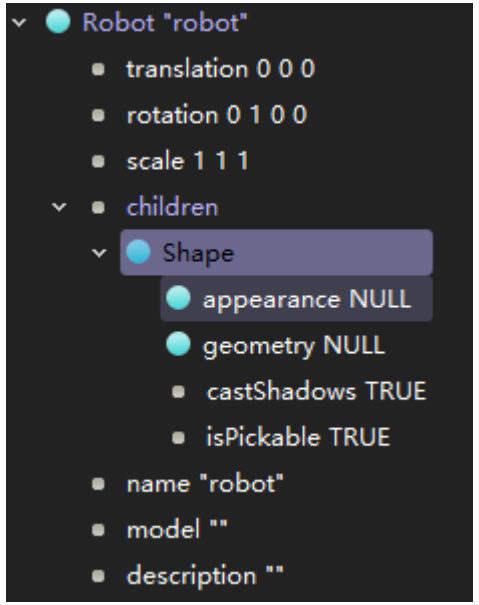

添加节点
?
Base nodes
Billboard
Charger
DirectionalLight
Fluid
Fog
Group
PointLight
Robot
Shape
Solid
SpotLight
Transform
USE
PROTO nodes (Current Project)
PROTO nodes (Webots Projects)
bounding_objects
default
devices
humans
Find:
Robot
The Robot node is a generic type of robot.
添加
Import...
Cancel
Base nodes是要自己造的，下面的PROTO nodes是现成的。
我们要建造属于自己的机器人，所以使用Base nodes->Robot

3.2 添加机器人刚体

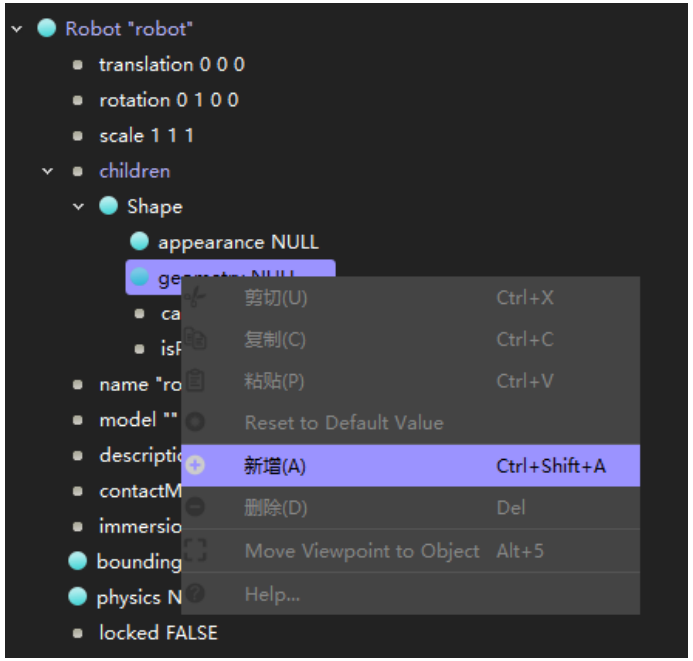


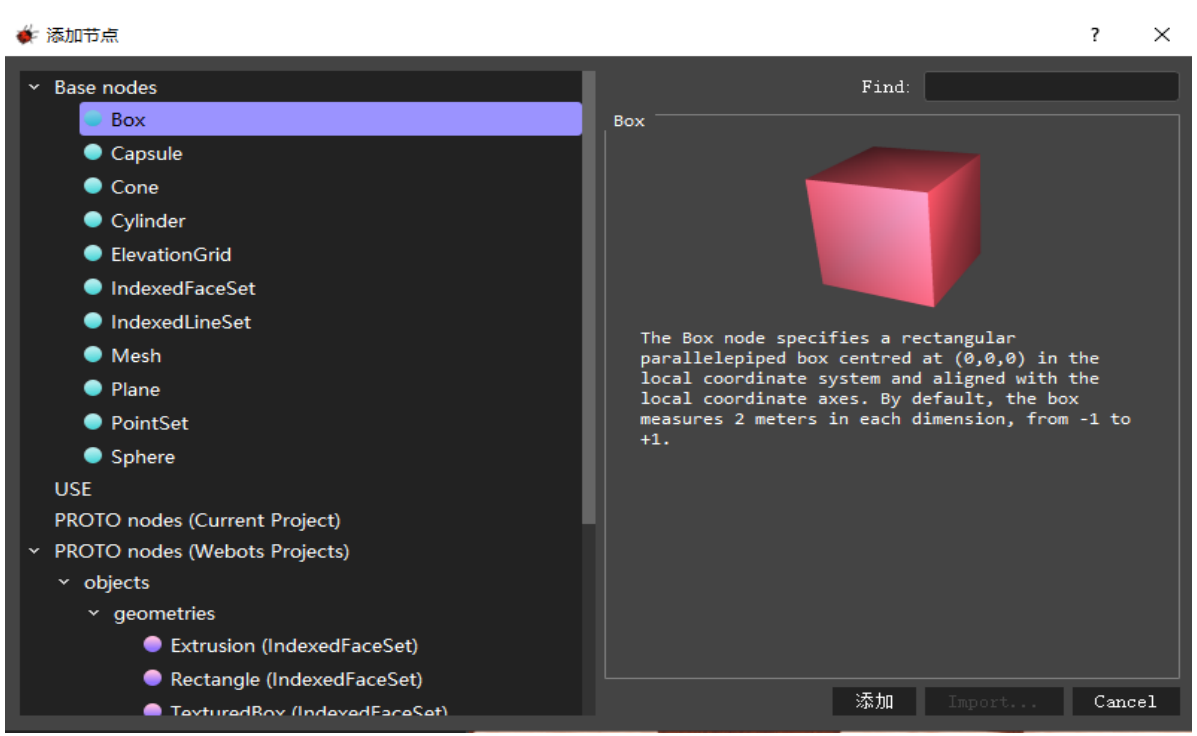


在children下创建第一个Shape节点，改名叫Body

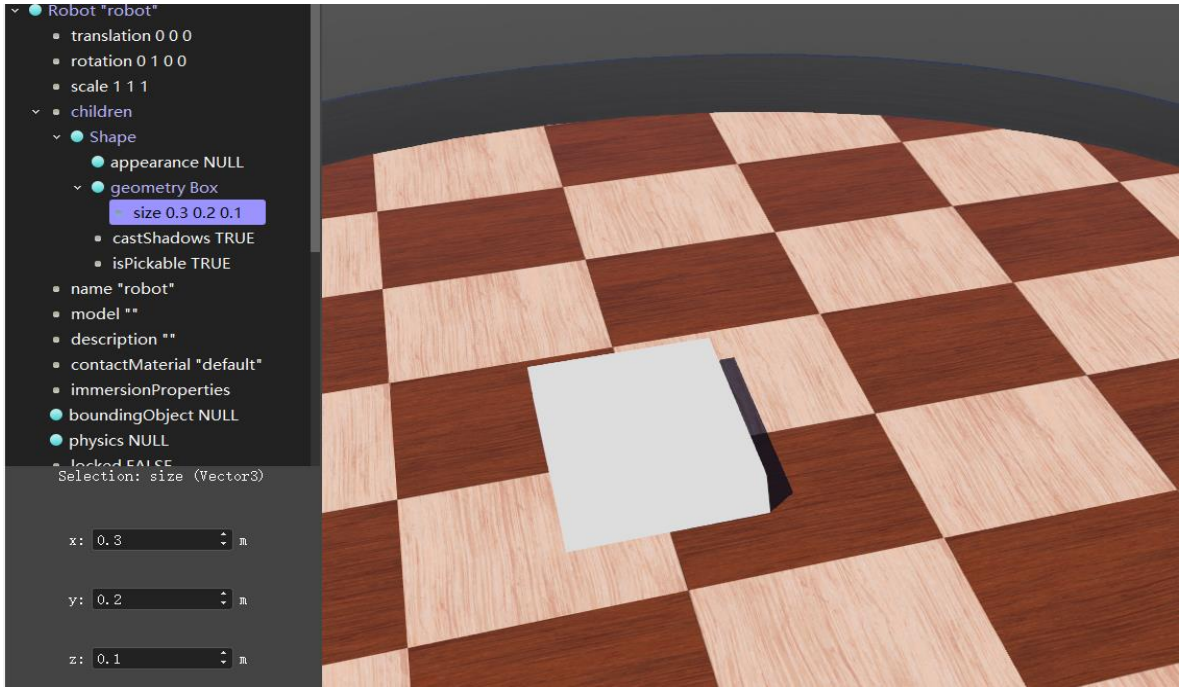


3.3 修改外观形状

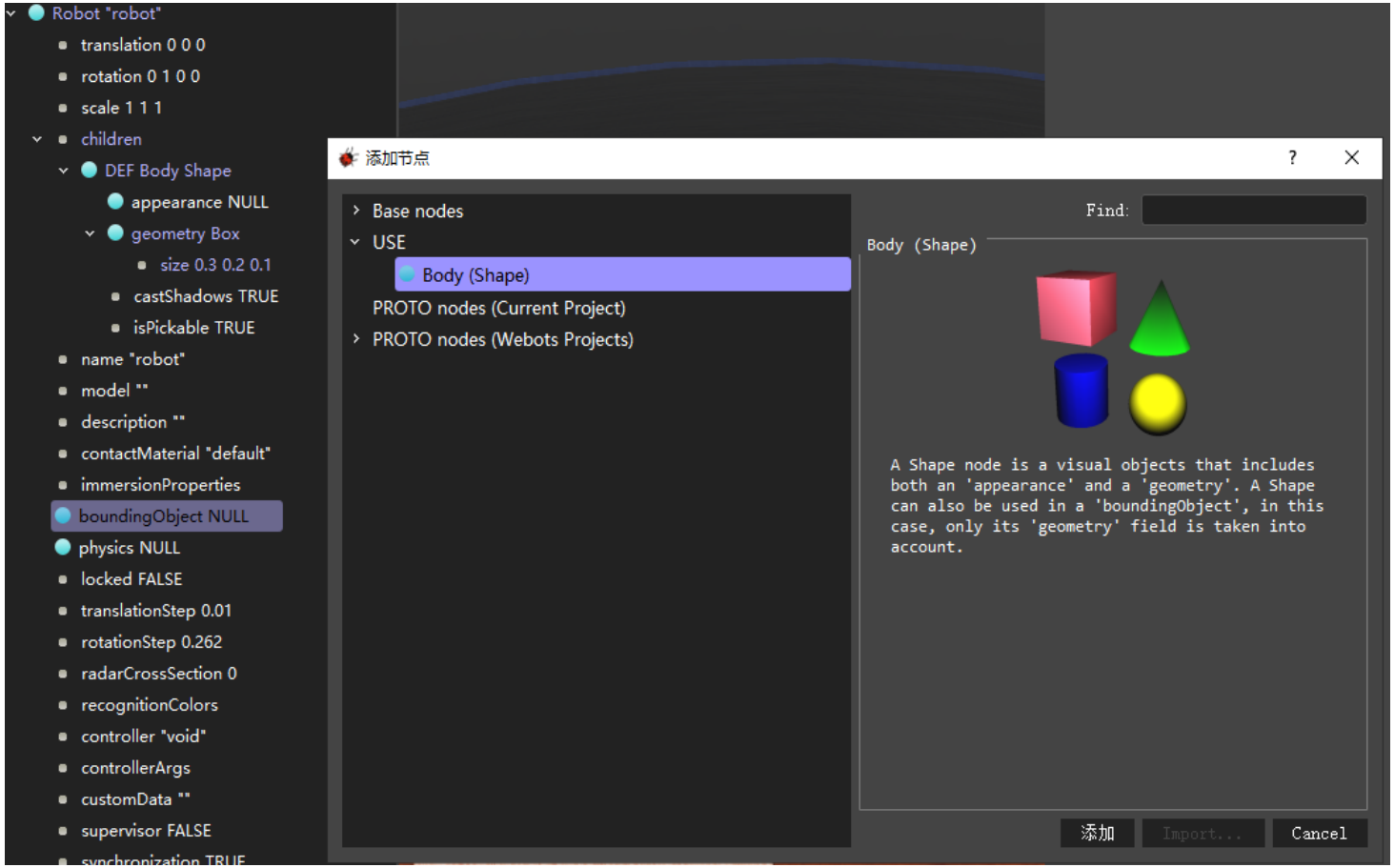




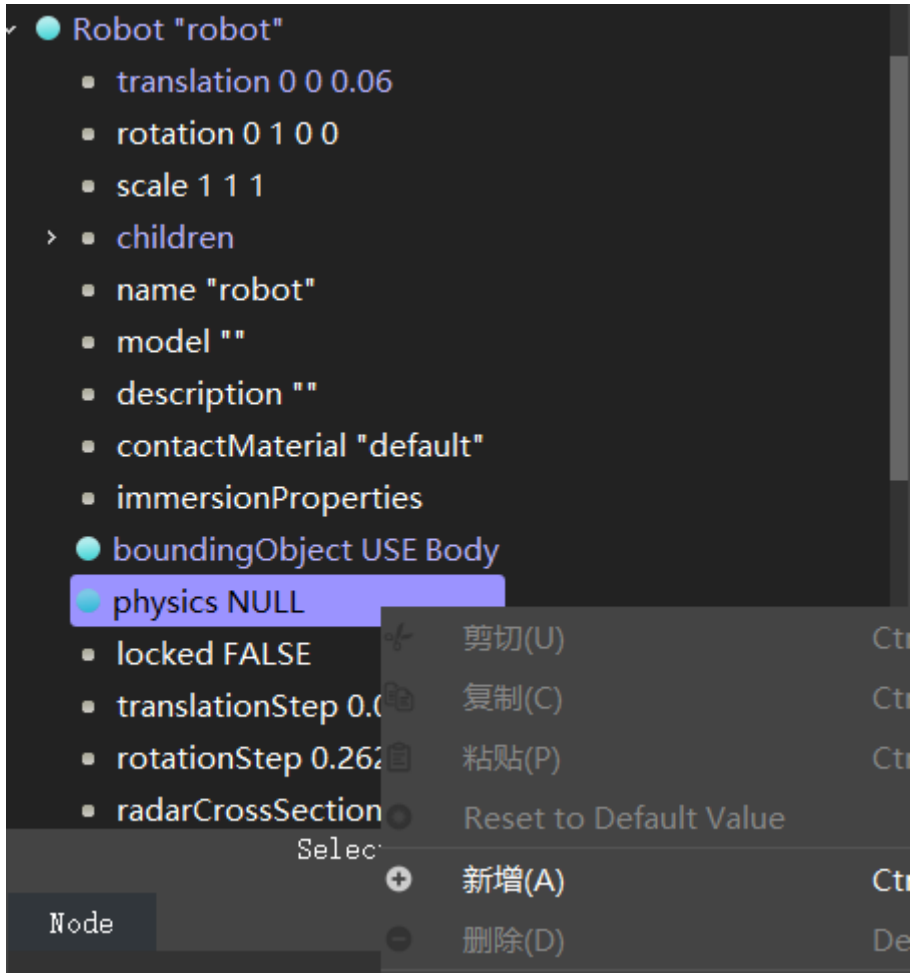
并在geometry节点添加几何属性，这里我们选择Box，并且设置车体大小，我这里设置（x=0.3 y=0.2 z=0.08）m



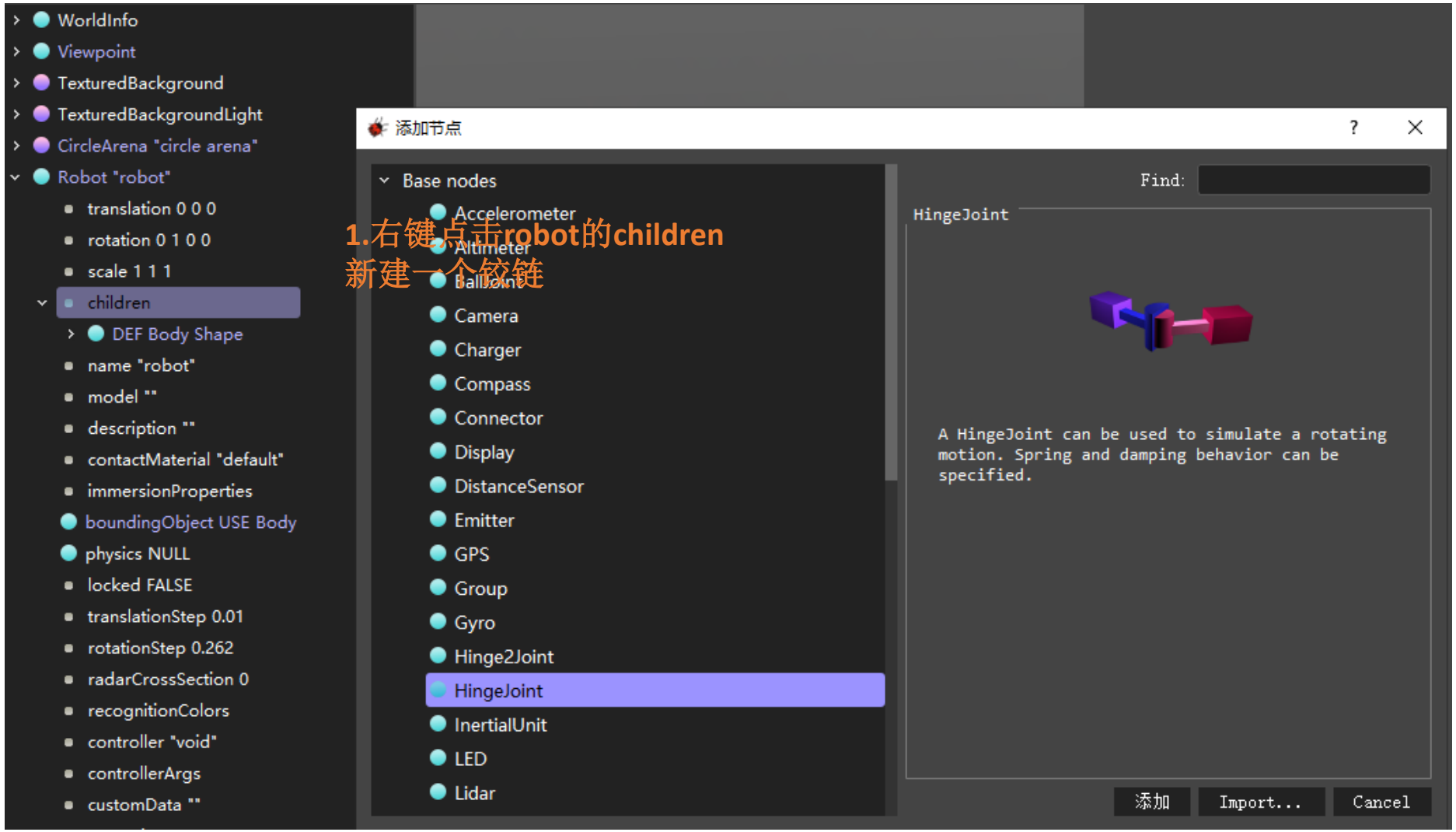
3.4 修改碰撞边界形状



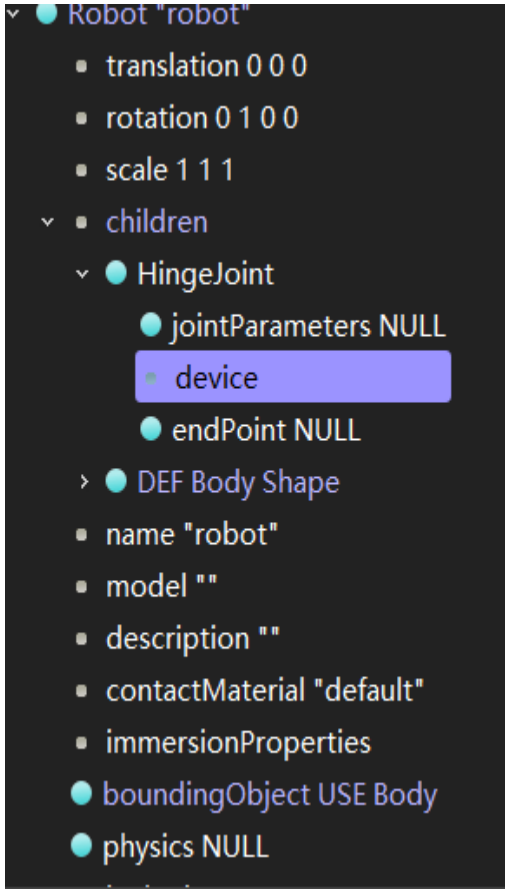
刚才的那个只是外观边界而已，我们要定义它物理碰撞边界。同时给它物理属性，否则它不受重力影响

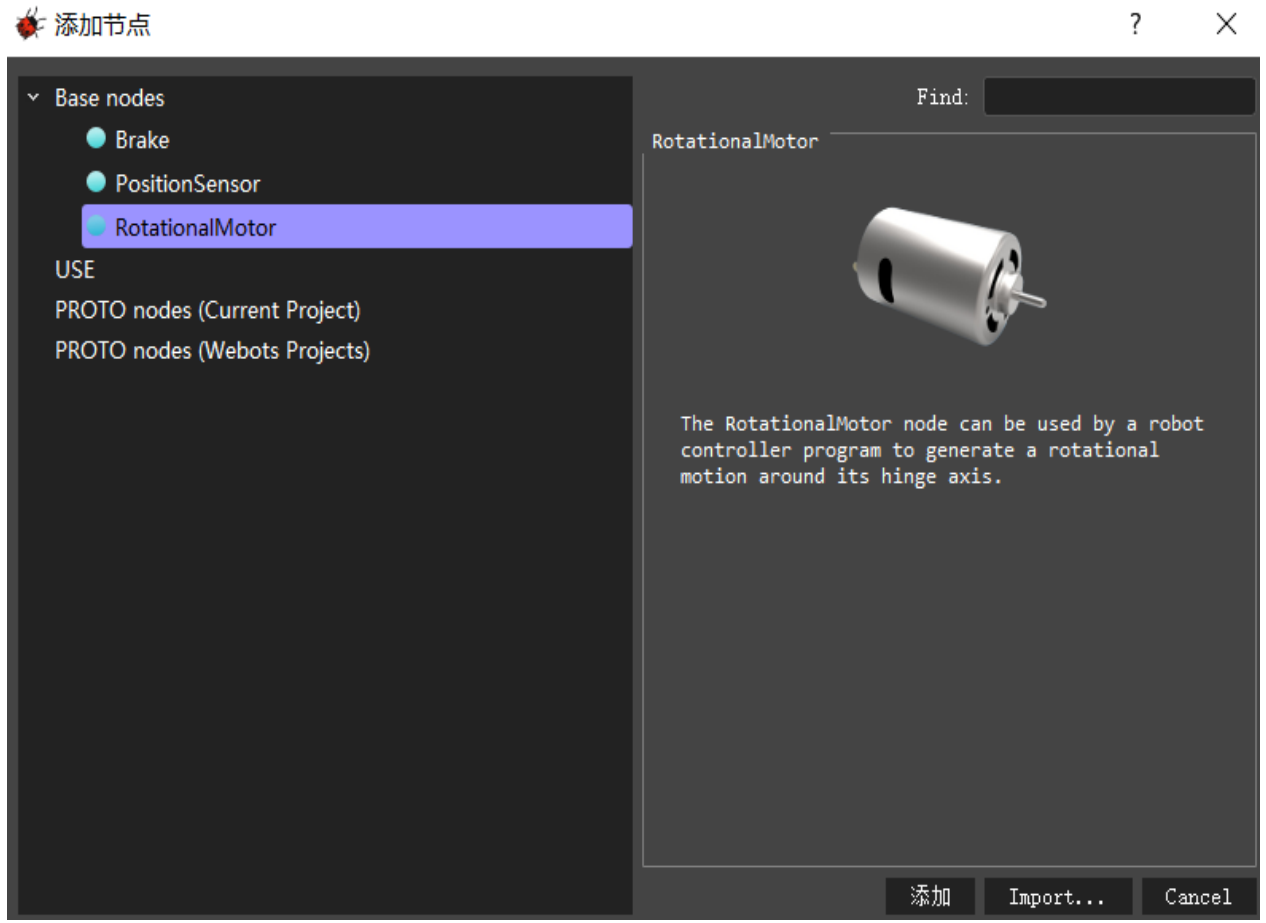


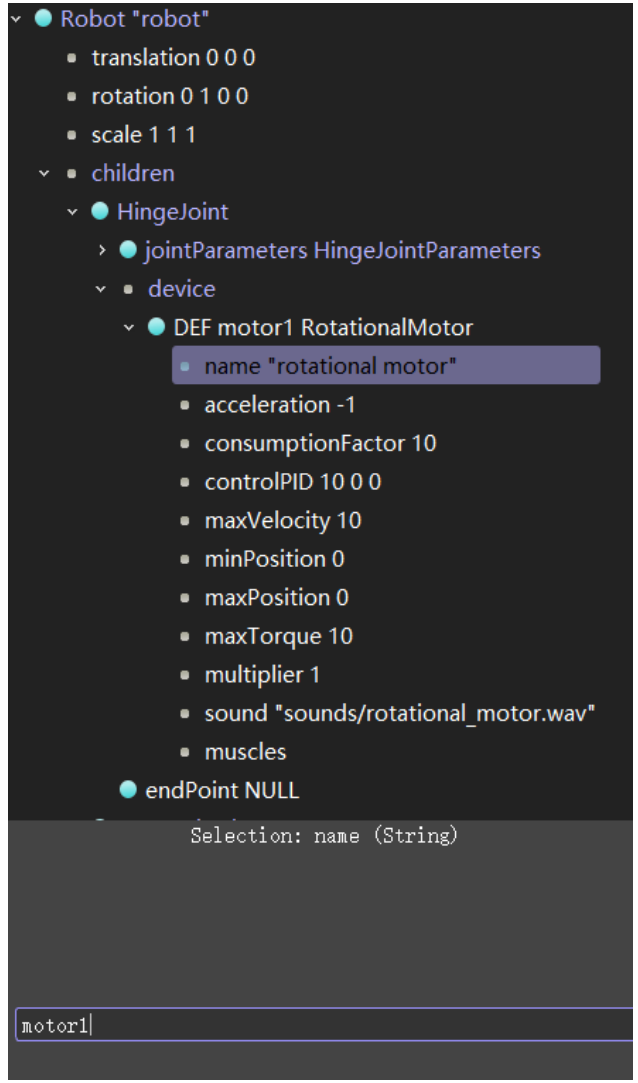
3.5 添加铰链


WorldInfo
Viewpoint
TexturedBackground
TexturedBackgroundLight
CircleArena "circle arena"
Robot "robot"
translation 0 0 0
rotation 0 1 0 0
scale 1 1 1
children
DEF Body Shape
name "robot"
model ""
description ""
contactMaterial "default"
immersionProperties
boundingObject USE Body
physics NULL
locked FALSE
translationStep 0.01
rotationStep 0.262
radarCrossSection 0
recognitionColors
controller "void"
controllerArgs
customData ""
1.右键点击robot的children
新建一个铰链
添加节点
Base nodes
Accelerometer
Altimeter
BallJoint
Camera
Charger
Compass
Connector
Display
DistanceSensor
Emitter
GPS
Group
Gyro
Hinge2Joint
HingeJoint
InertialUnit
LED
Lidar
Find:
HingeJoint
A HingeJoint can be used to simulate a rotating motion. Spring and damping behavior can be specified.
添加
Import...
Cancel

3.51 为铰链一端添加电机

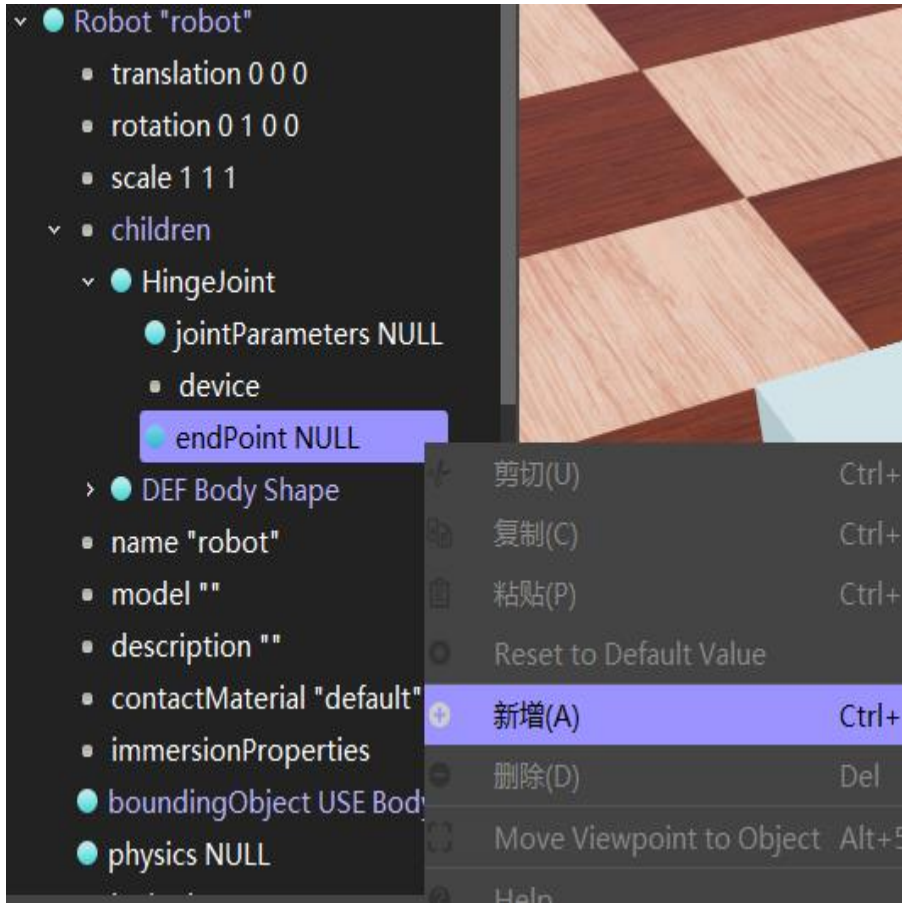


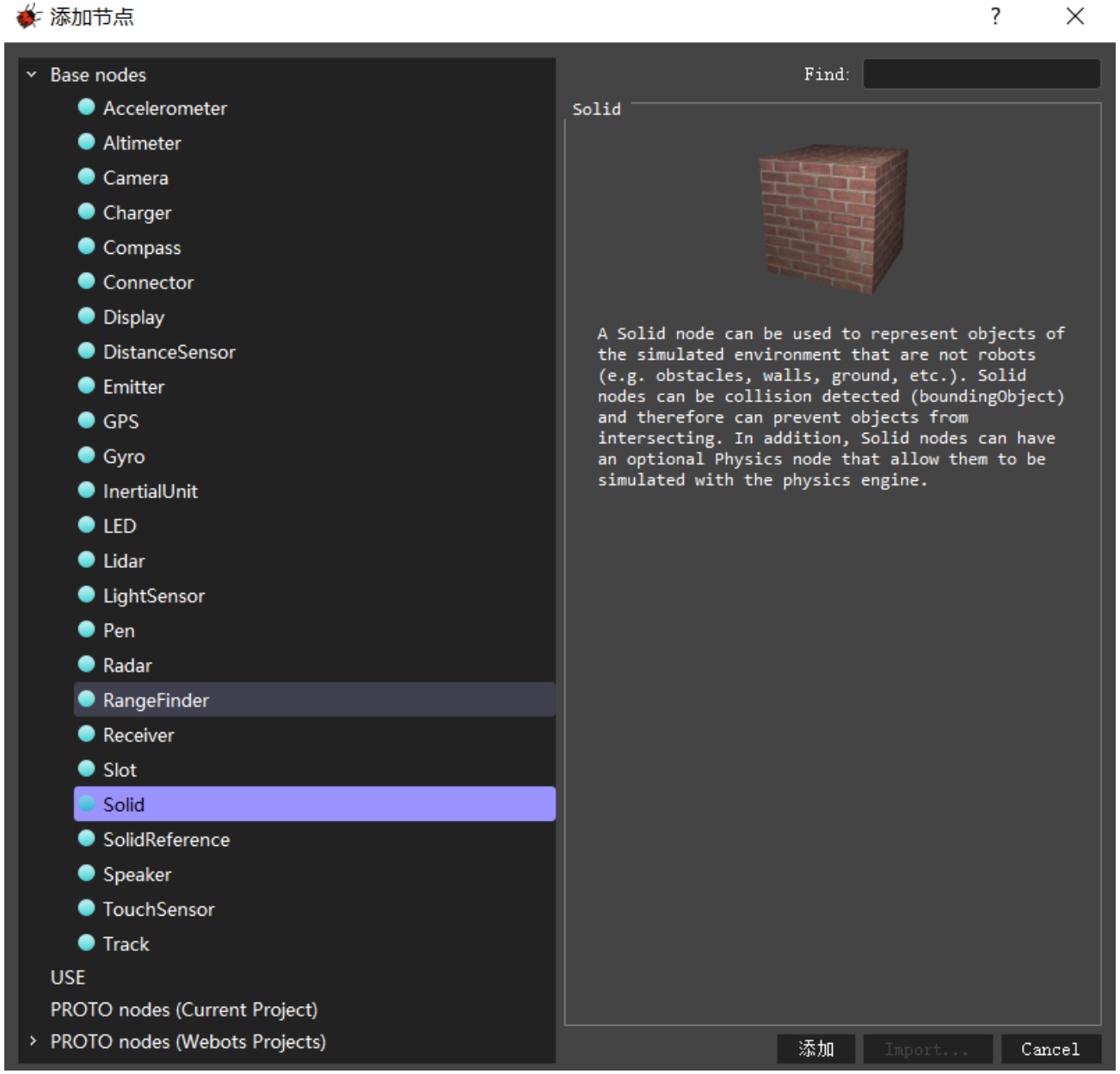




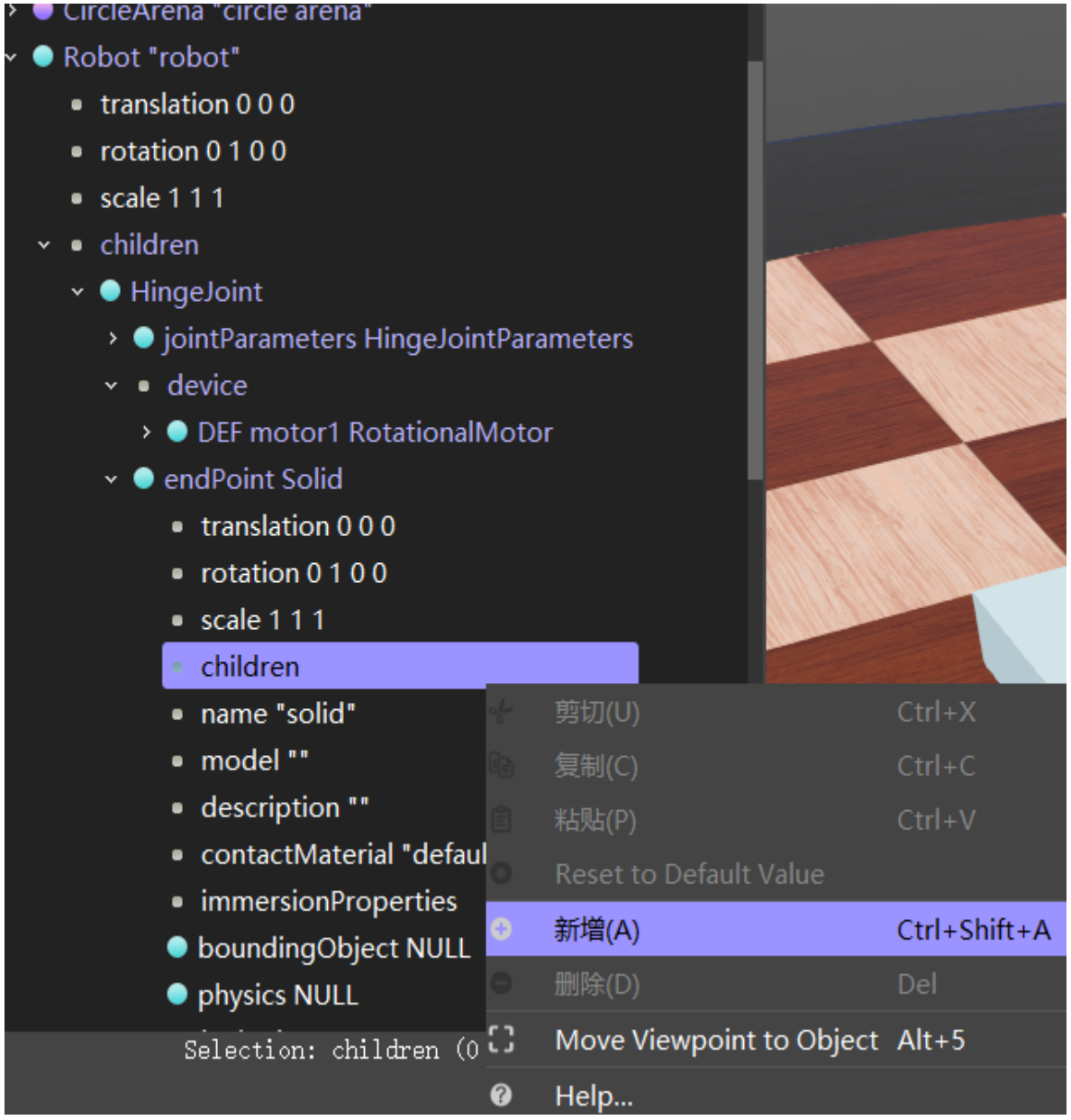
Device表示使用什么驱动，我们新建一个电机。
改名字，这个很重要，因为程序中想调用电机，要先通过名字获取它的句柄

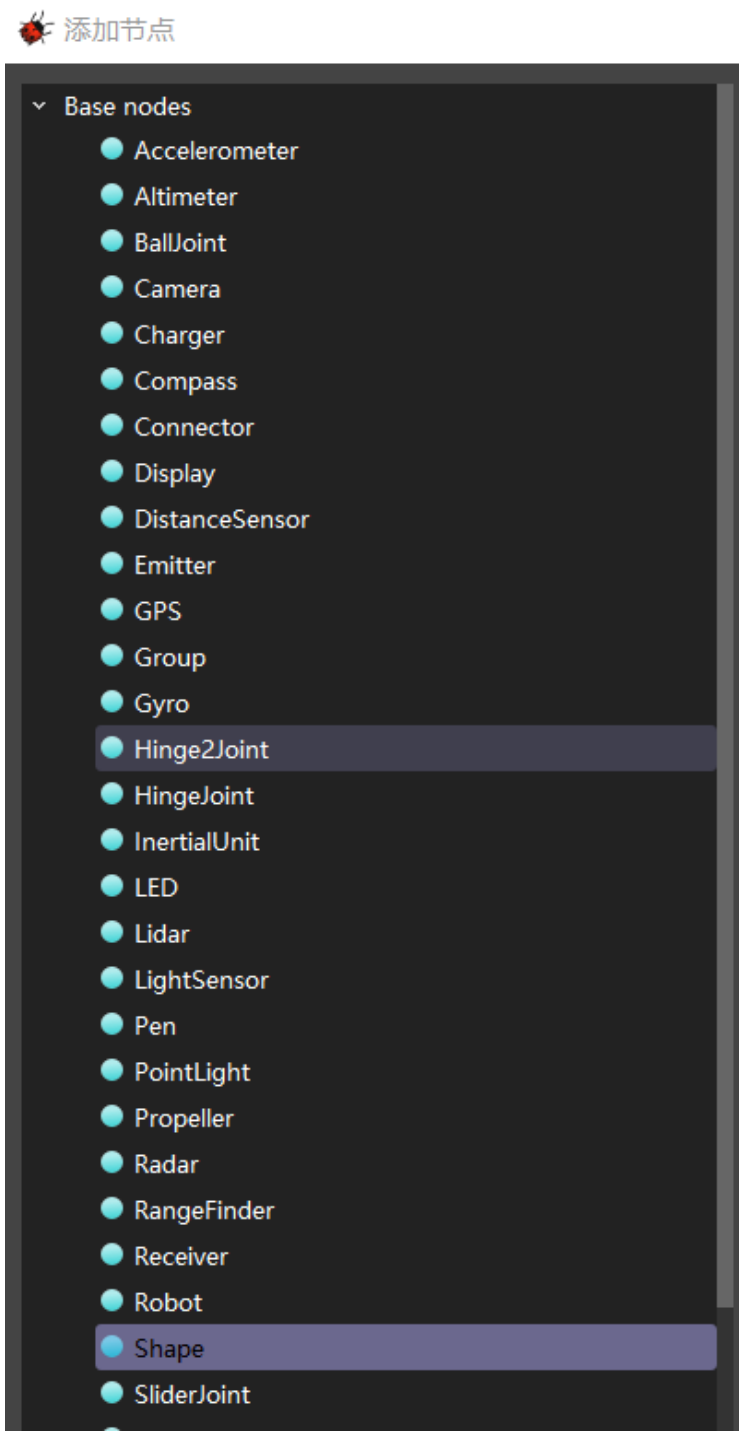
3.52 为铰链另一端添加轮子



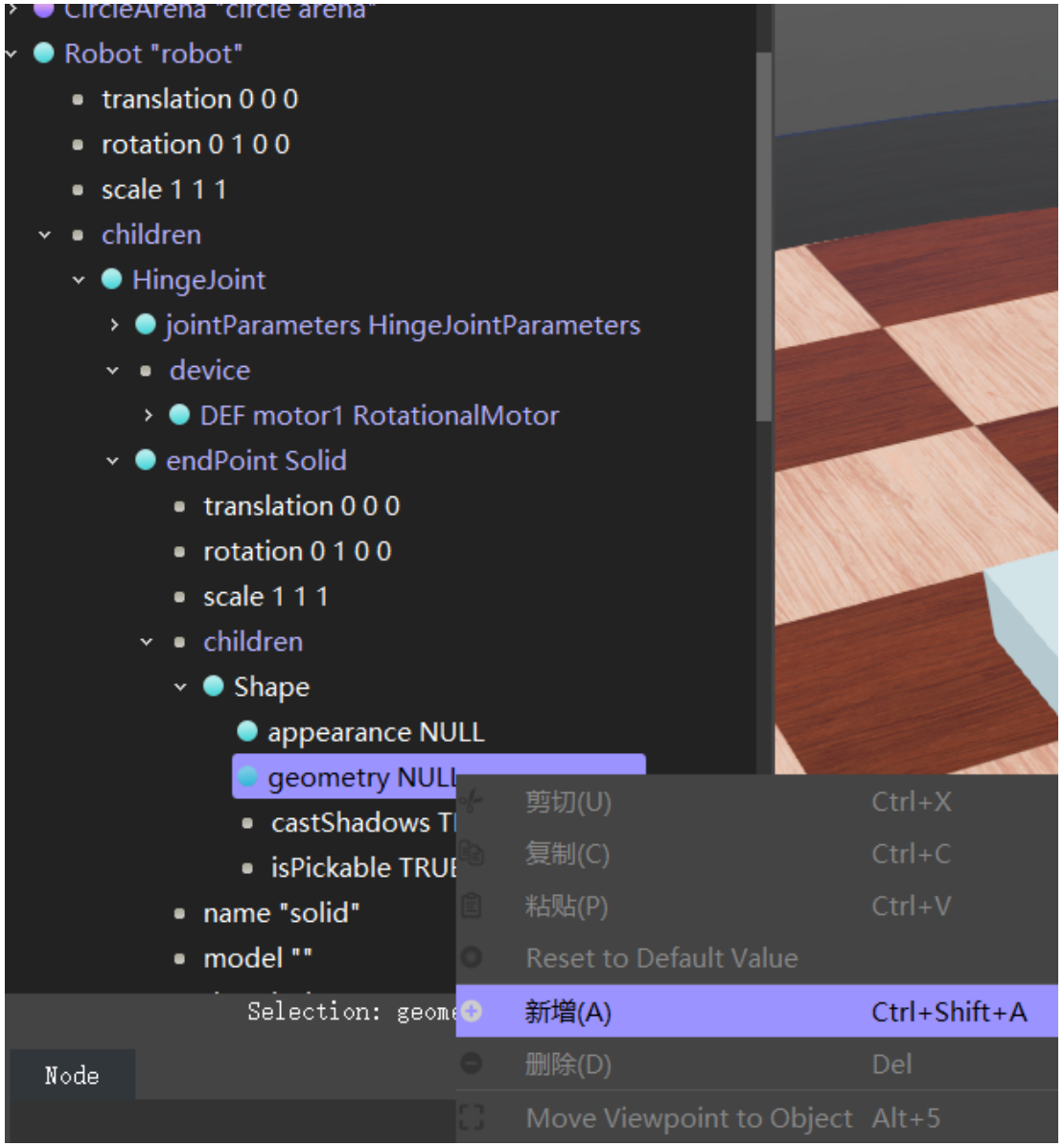


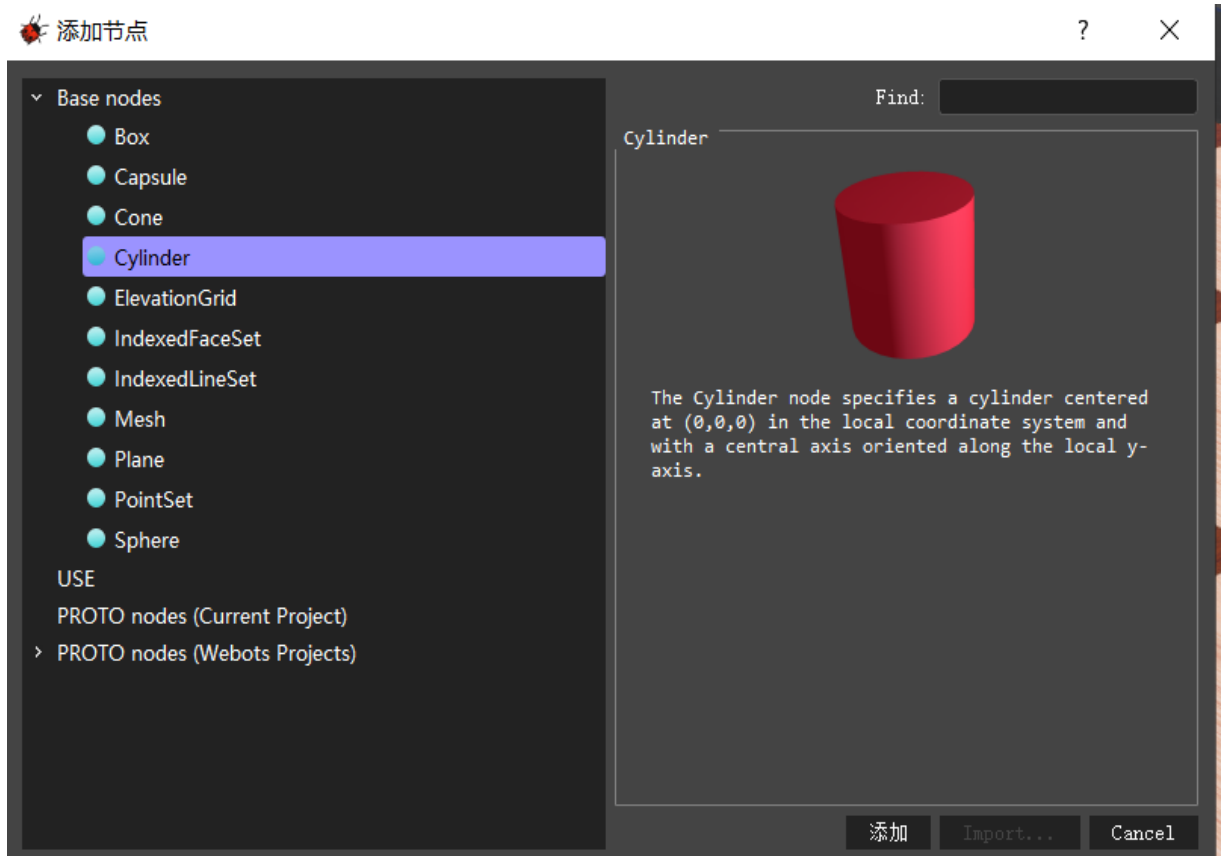
3.52 为铰链另一端添加轮子



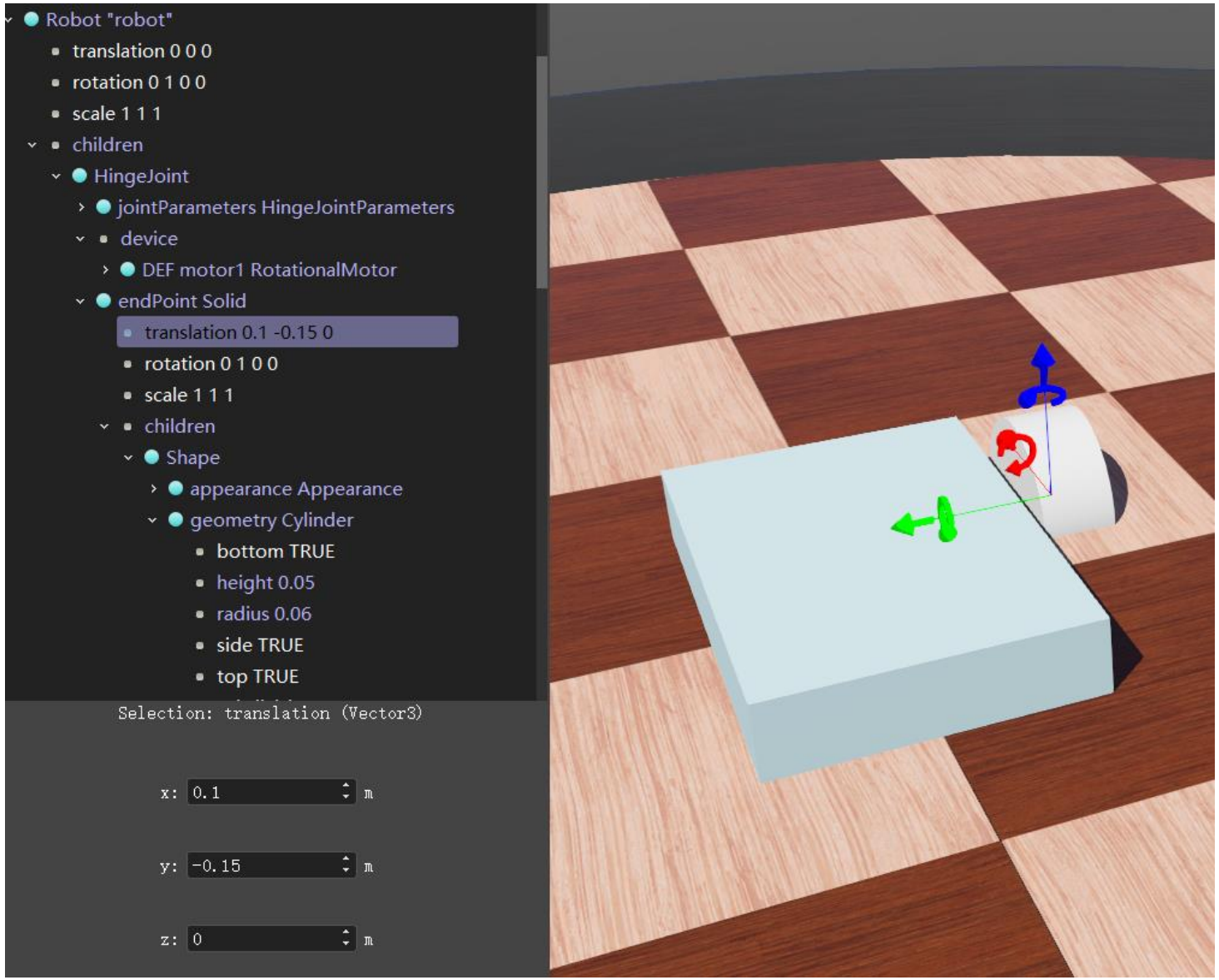


3.52 为铰链另一端添加轮子



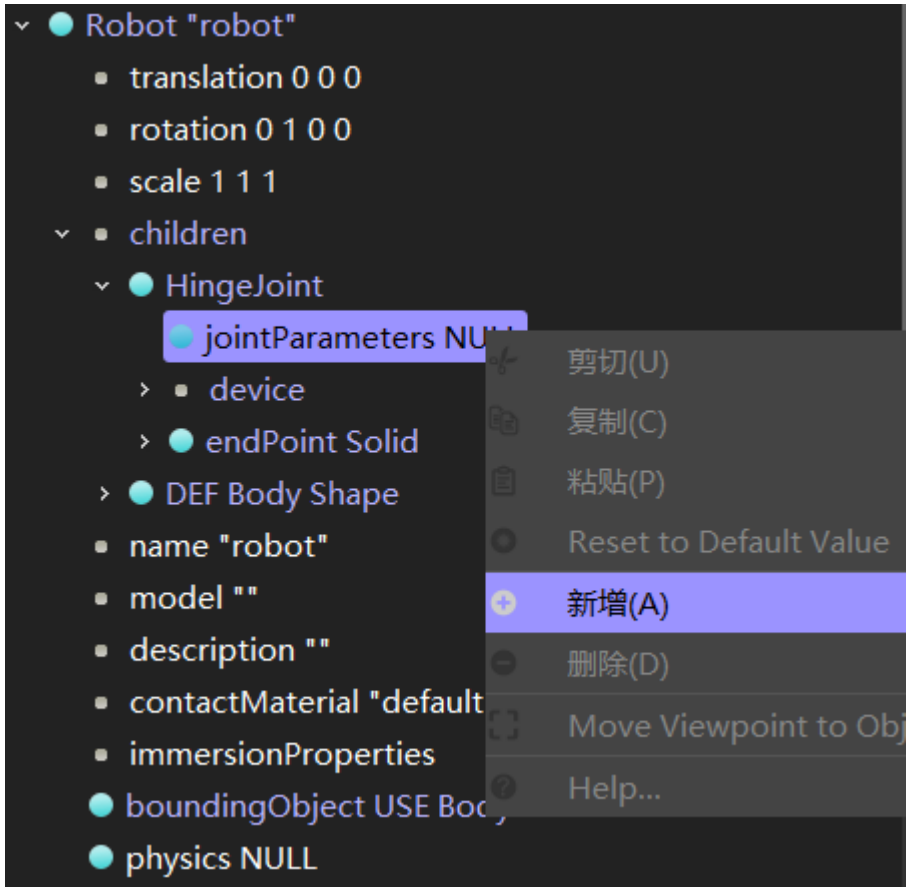


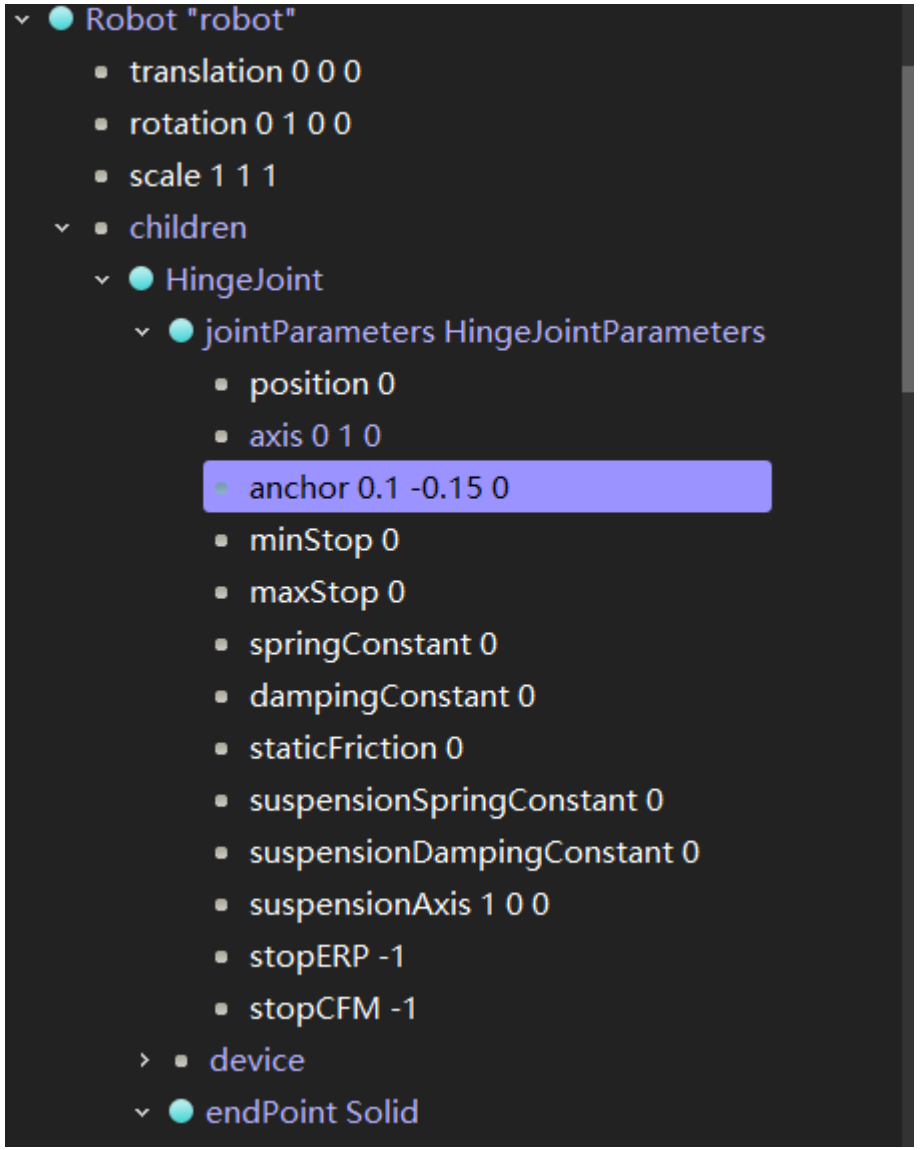
轮子也要给碰撞边界和物理属性，做法和身体类似



把轮子大小改成半径0.06m，高0.05m
把轮子位置改成（x=0.1 y=-0.15 z=0）m

3.53 把铰链装到机器人上

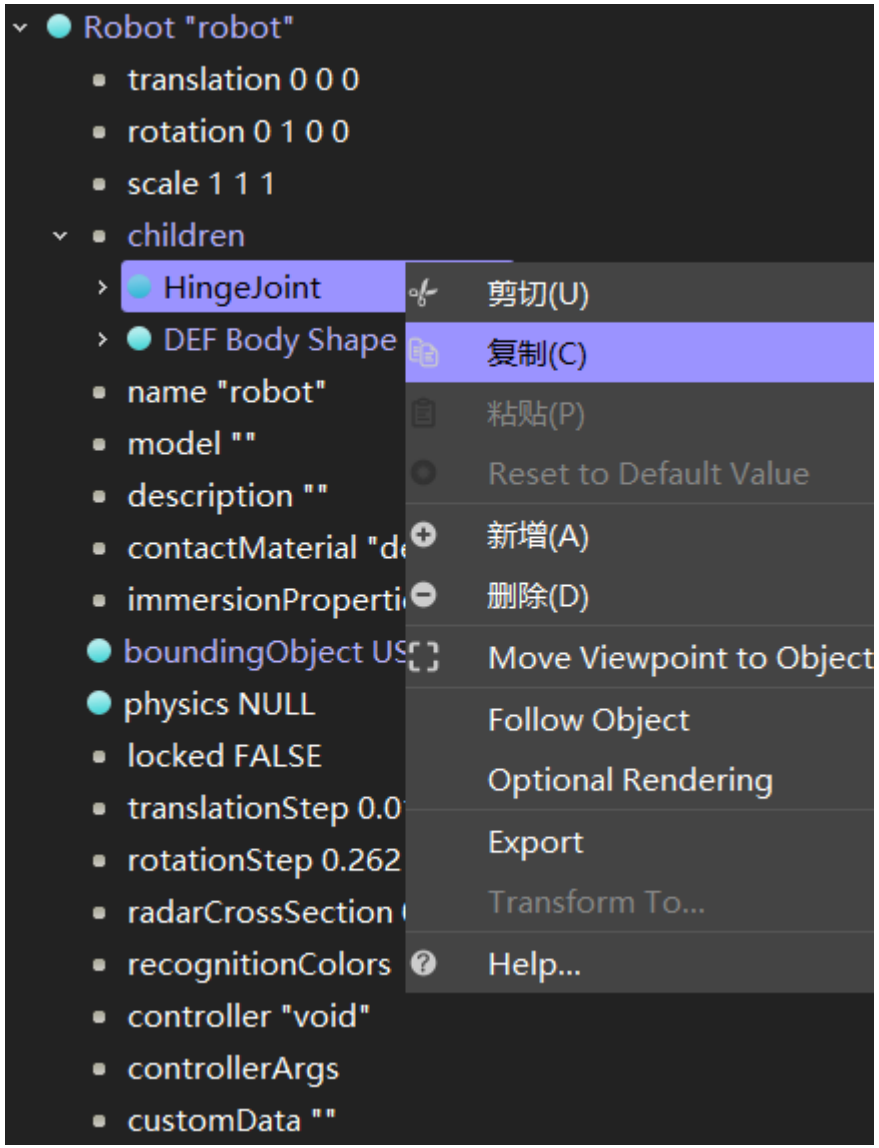


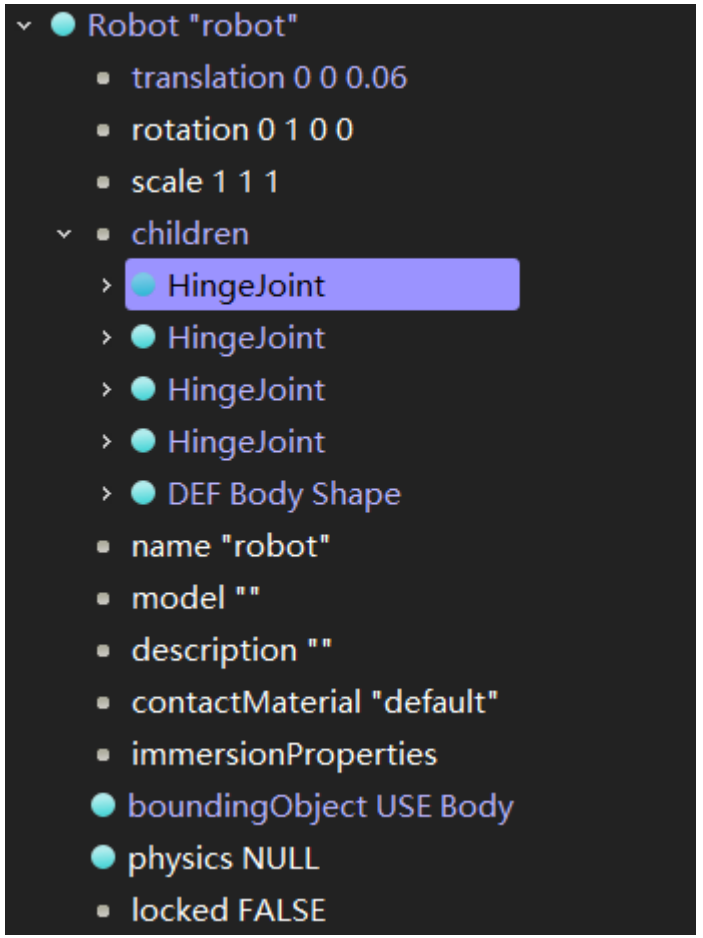


铰链
轴改成（x=0 y=1 z=0）

安装在（x=0.1 y=-0.15 z=0）
m处

3.54 复制出另外三个轮子

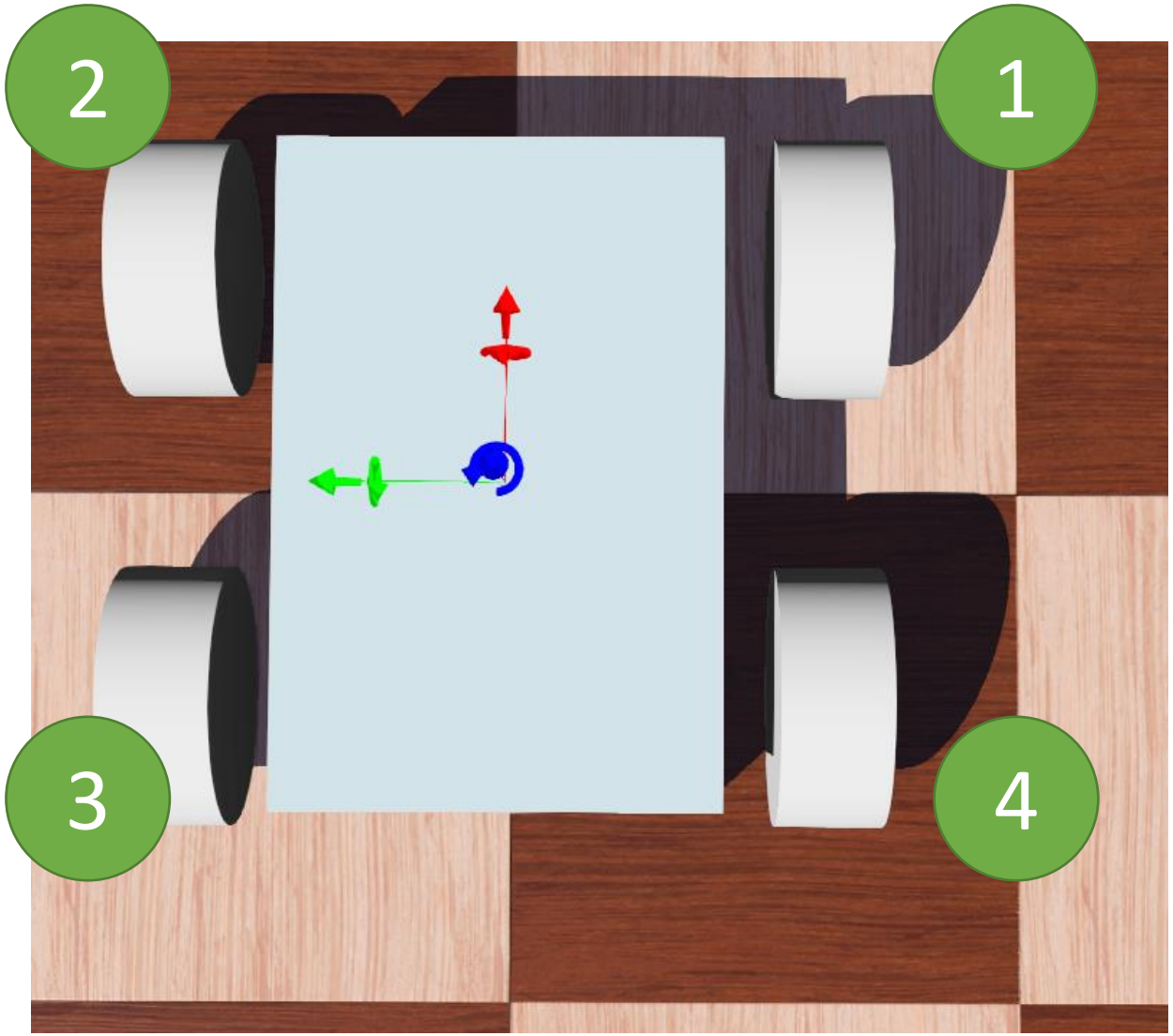




修改每个轮子的anchor
（x=0.1 y=-0.15 z=0）
（x=0.1 y=0.15 z=0）
（x=-0.1 y=0.15 z=0）
（x=-0.1 y=-0.15 z=0）

修改每个电机的名字
motor1，motor2，
motor3，motor4

（x=0.1 y=0.15 z=0）
motor2



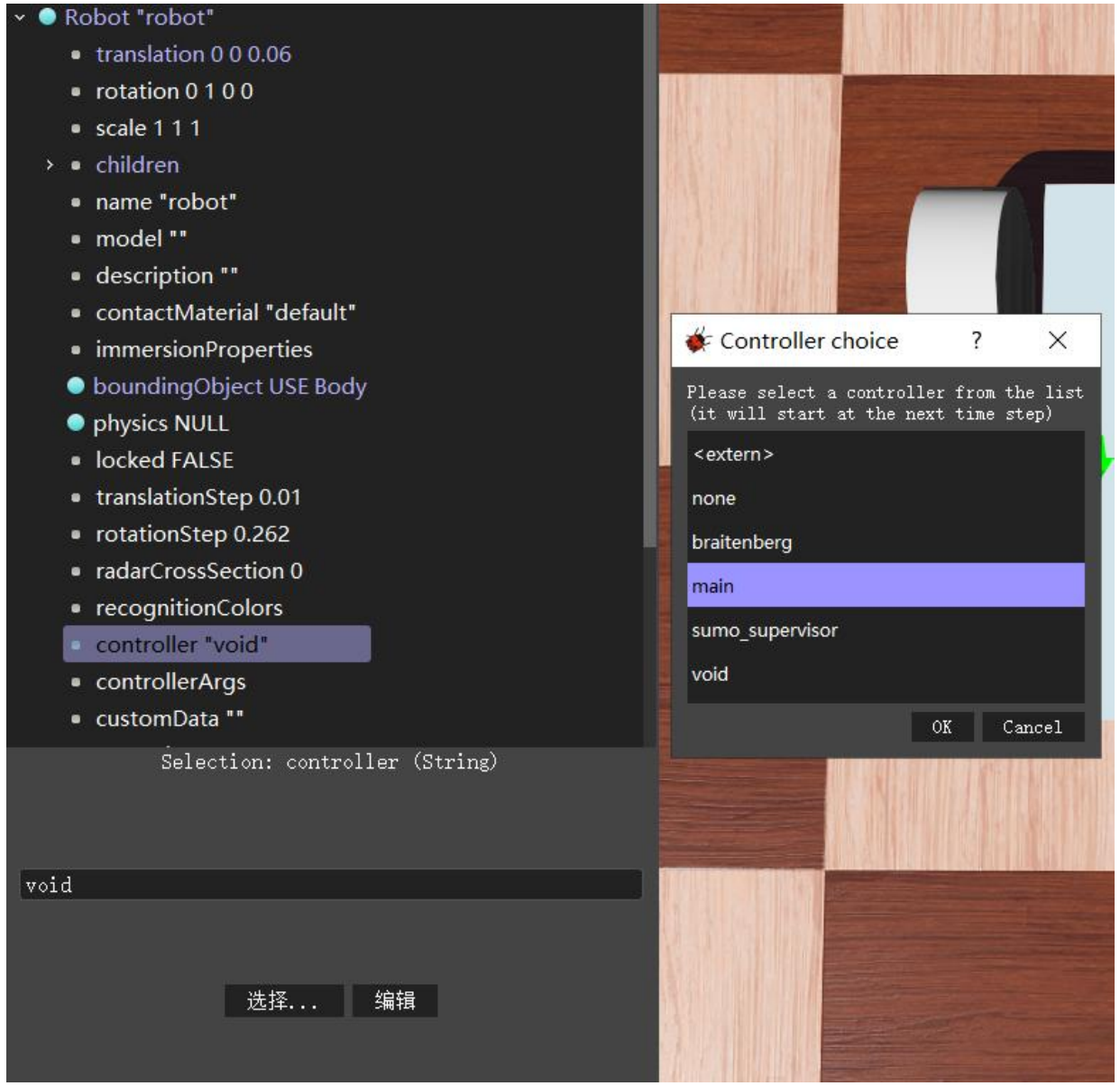
（x=0.1 y=-0.15 z=0）
motor1

（x=-0.1 y=0.15 z=0）
motor3

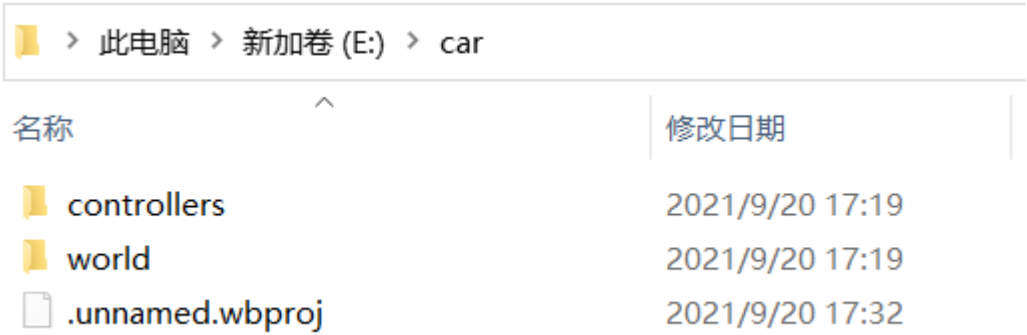
（x=-0.1 y=-0.15 z=0）
motor4

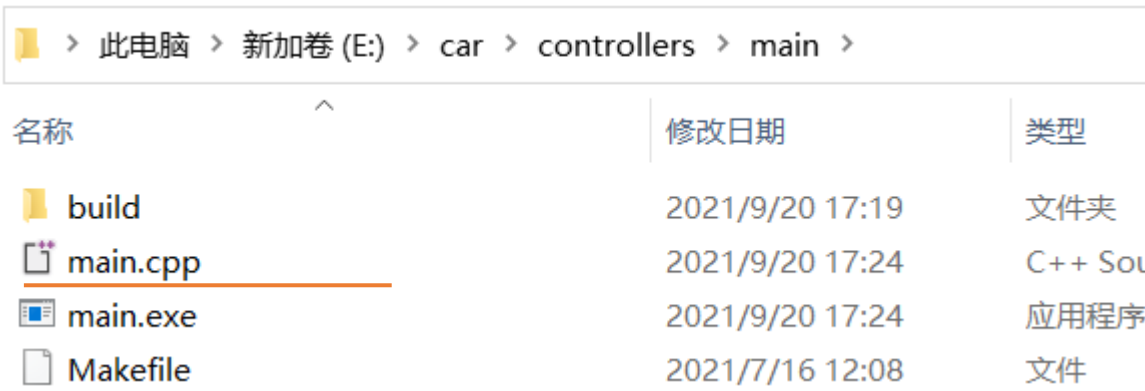
4.1 添加机器控制器

使用C++，控制器名字叫main，这时候你打开原来的那个文件夹，会发现里面多了一个controller文件夹



选择刚才的那个，点编辑就可以编辑了，你也可以直接打开对应目录下的文件直接改





4.3 键盘的使用

```
#include <webots/Robot.hpp>
#include <webots/Motor.hpp>
#include <webots/Keyboard.hpp>

using namespace std;
using namespace webots;
webots::Keyboard keyboard;
keyboard.enable(1);//运行键盘输入设置频率是1ms读取


int keyValue1 = keyboard.getKey();
int keyValue2 = keyboard.getKey();

if (keyValue1 == 'W')
{
    。。。。。。。。。。。。
}
```

所有传感器都类似与多线程，你只需要管开启和设置时间间隔，之后它会自己调用

4.2 电机的使用

```
#include <webots/Robot.hpp>
#include <webots/Motor.hpp>
#include <webots/Keyboard.hpp>

using namespace std;
using namespace webots;

Motor *motors[4];
char wheels_names[4][8] = { “motor1”, “motor2”, “motor3”, “motor4” };
Robot *robot = new Robot();//使用webots的机器人主体
for (int i = 0; i < 4; i++)
{
        motors[i] = robot->getMotor(wheels_names[i]);
        motors[i]->setPosition(std::numeric_limits<double>::infinity());
        motors[i]->setVelocity(0.0);
}

motors[0]->setVelocity(1.0);
```

4.4与仿真器同步

```
while (robot->step(timeStep) != -1) //仿真运行一帧
{

    //获取键盘输入，这样写可以获得同时按下的按键（最多支持7个）
    int keyValue1 = keyboard.getKey();
    int keyValue2 = keyboard.getKey();
    cout << keyValue1 << ":" << keyValue2 << endl;

    //根据按键决定电机怎么样转动
    。。。。。。。。。。。。。。。。。。。

    //让电机执行
    for (int i = 0; i < 4; i++)
   {
        motors[i]->setVelocity(speed1[i] + speed2[i]);
   }

}
```

5.1 作业——麦克纳姆轮的制作

ROS联合Webots之麦克纳姆轮篇-搭建麦轮底盘

原创 锡城筱凯 2021-06-10 00:24:04 282 收藏 5 版权

分类专栏：▶机器人仿真 ▶机器人操作系统ROS 文章标签：ROS Webots

▶... 同时被 2 个专栏收录 2 订阅 27 篇文章 订阅专栏

ROS联合Webots之麦克纳姆轮篇-搭建麦轮底盘

ubuntu版本：20.04
webots版本：2021a
ros版本：noetic

0.前言

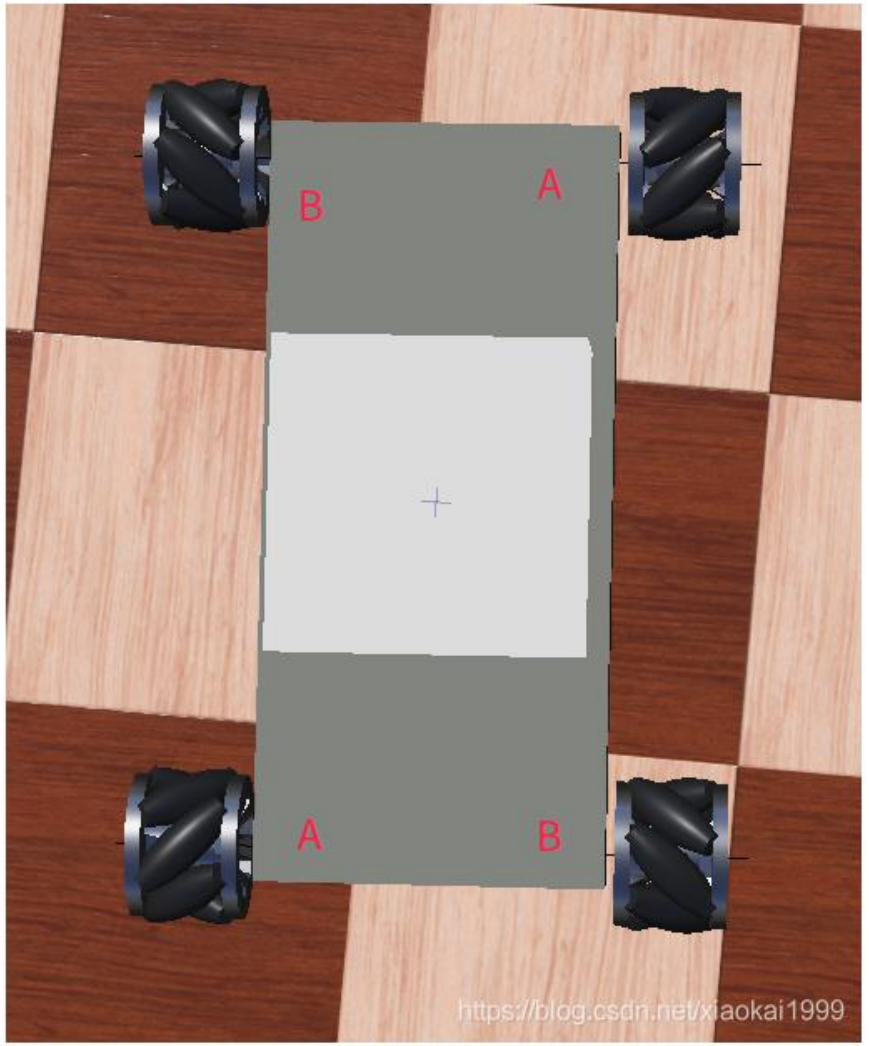
之前笔者出过ROS联合webots开发教程，在教程中使用的是双轮差动底盘模型，今天笔者将带给笔者麦克纳姆轮的使用教程。

1.模型选取

在Webots中，官方提供了kuka公司推出的youbot机器人作为麦克纳姆轮的案例

- 实物机器人如下所示：

youbot机器人底盘上的麦克纳姆轮为A-B-A-B配置，如下图所示：

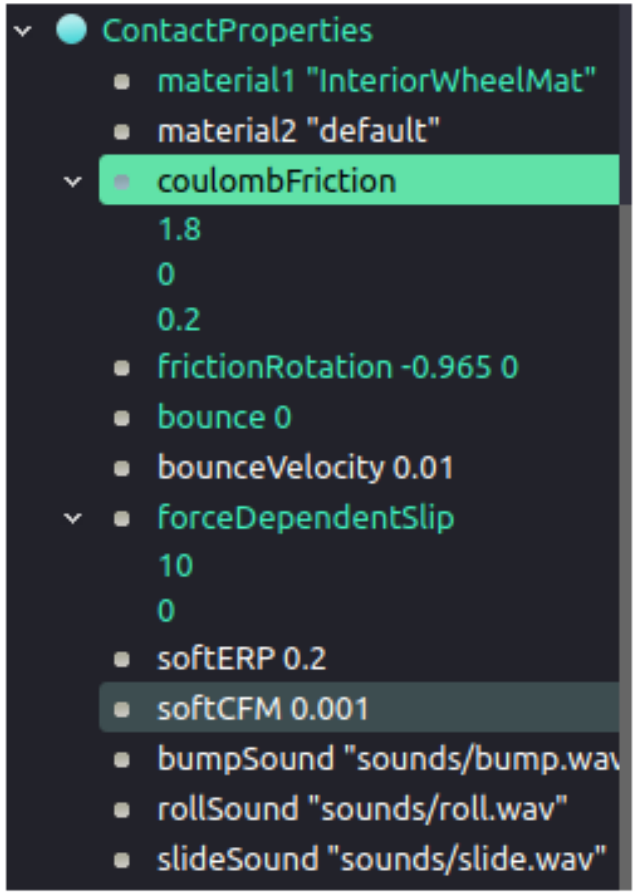


将四个轮子复制进自己的机器人中。

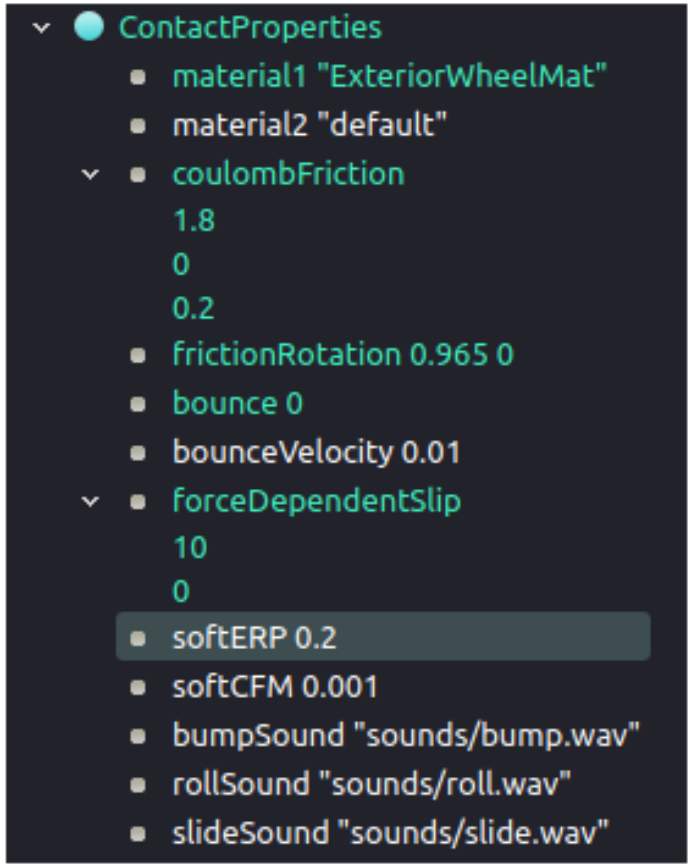
https://blog.csdn.net/xiaokai1999/article/details/117757052

5.2 设置摩擦材质

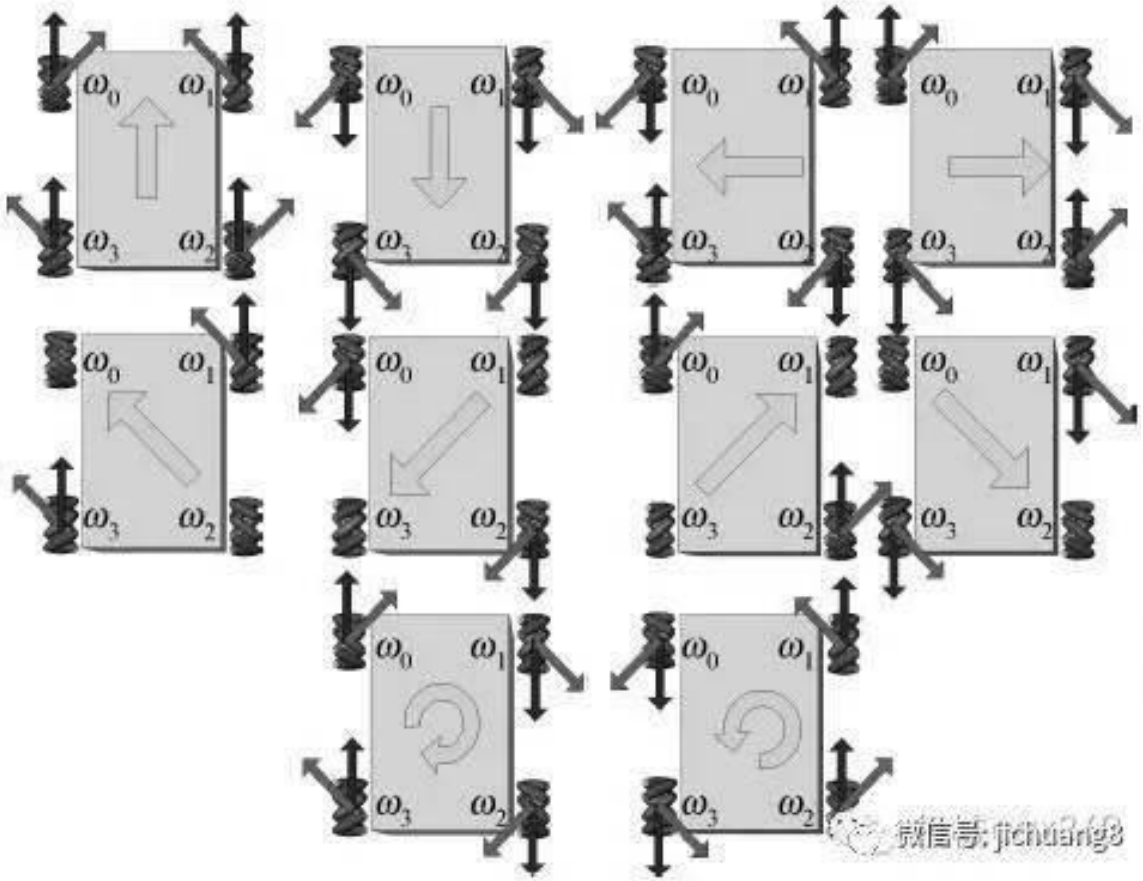
第一个`ContactProperties`节点如下图所示：

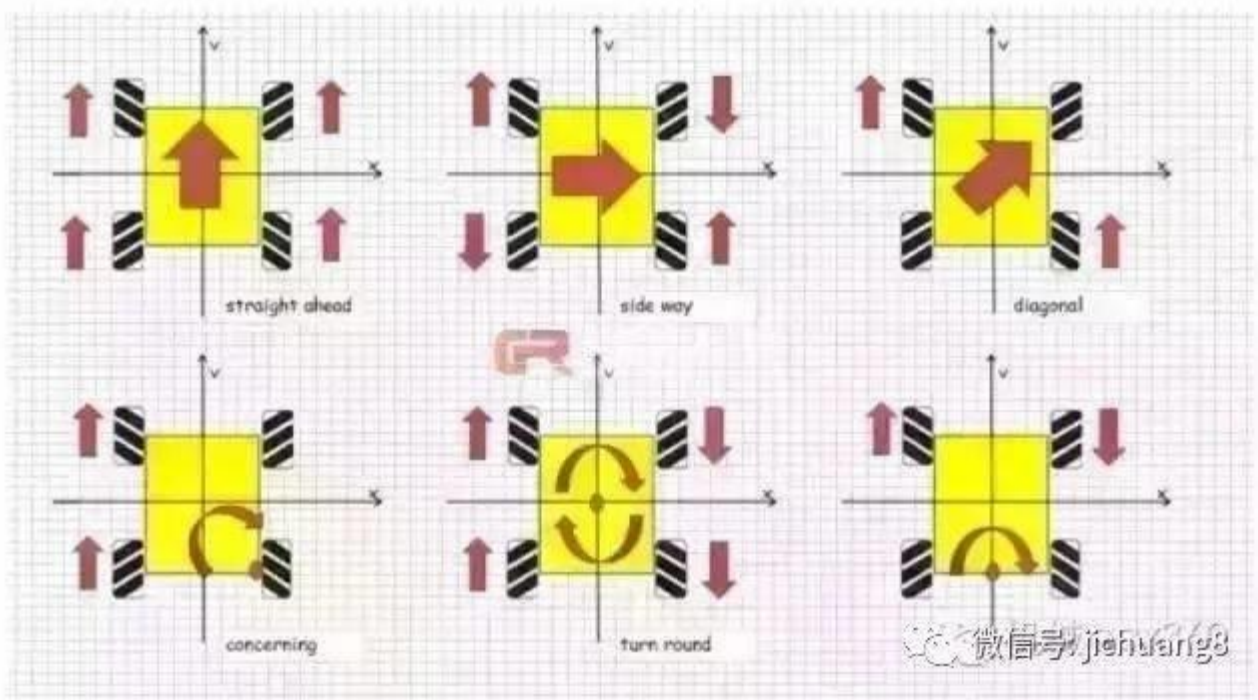


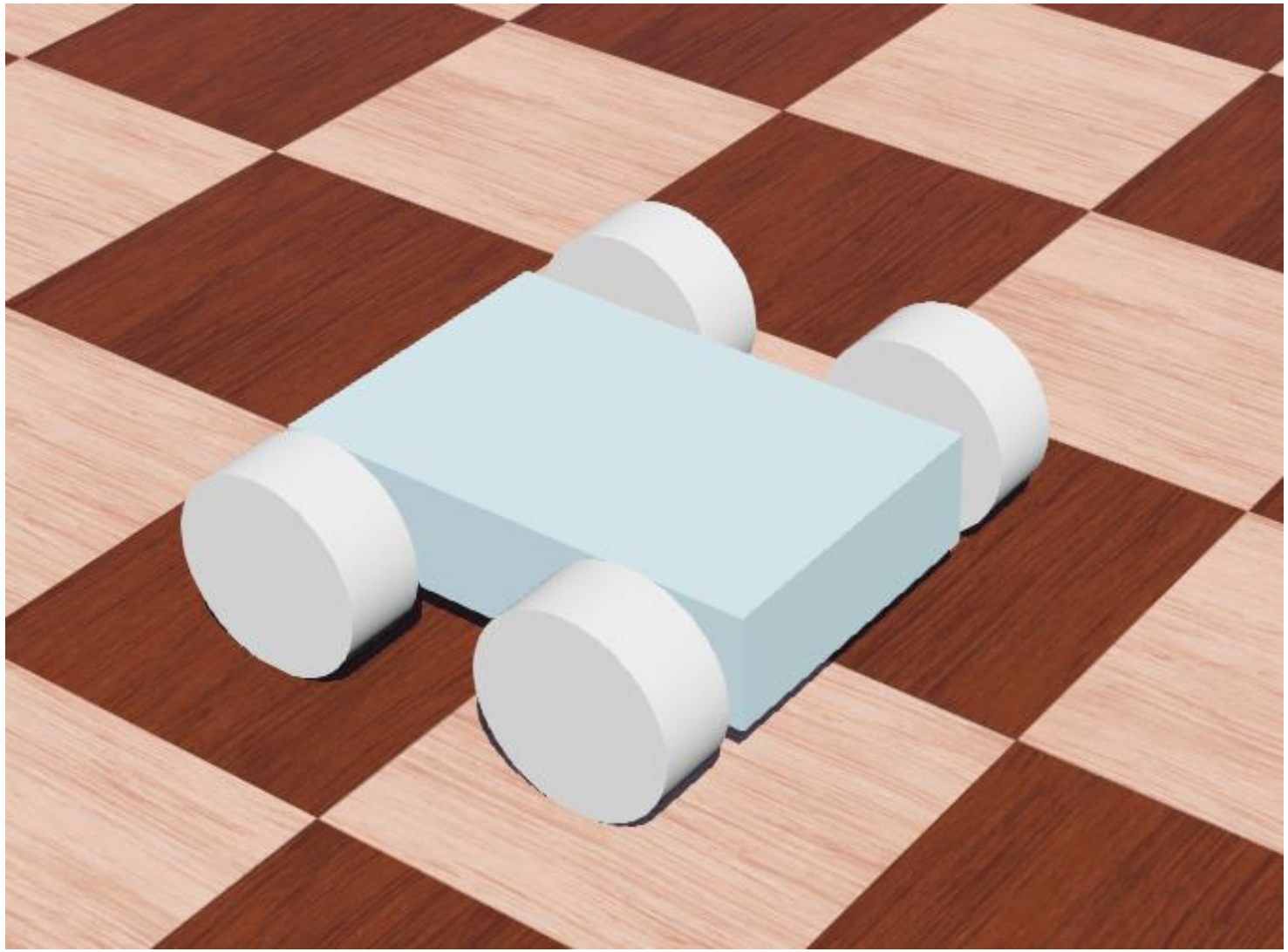
第二个`ContactProperties`节点如下图所示：



5.3麦克纳姆轮原理


ω0
ω1
ω3
ω2
微信号: jichuang8


straight ahead
side way
diagonal
concerning
turn round
微信号: jichuang8




Simulation View
Text Editor
0:26:32:928 - 0.96x
Robot "youBot"
translation 0.757 0.0993 0.303
rotation -1 0.0137 0.0133 1.57
scale 1 1 1
children
VelodyneVLP-16 "Velodyne VLP-16"
translation 0 3.67e-08 0.09
rotation -1 -2.16e-08 7.17e-07 -1.57
name "Velodyne VLP-16"
enablePhysics TRUE
GPS "gps"
InertialUnit "inertial unit"
Group
children
Shape
appearance PBRAppearance
baseColor 1 0.4 0
baseColorMap NULL
transparency 0
roughness 0.4
roughnessMap NULL
metalness 0
Selection: appearance PBRAppearance
Node
DEF:
KUKA
KUKA
Y
Z
X
Console - All

文件(F) 编辑(E) 查看(V) 模拟(S) 生成(B) Overlays 工具(T) 向导(W) 帮助(H)

Simulation View

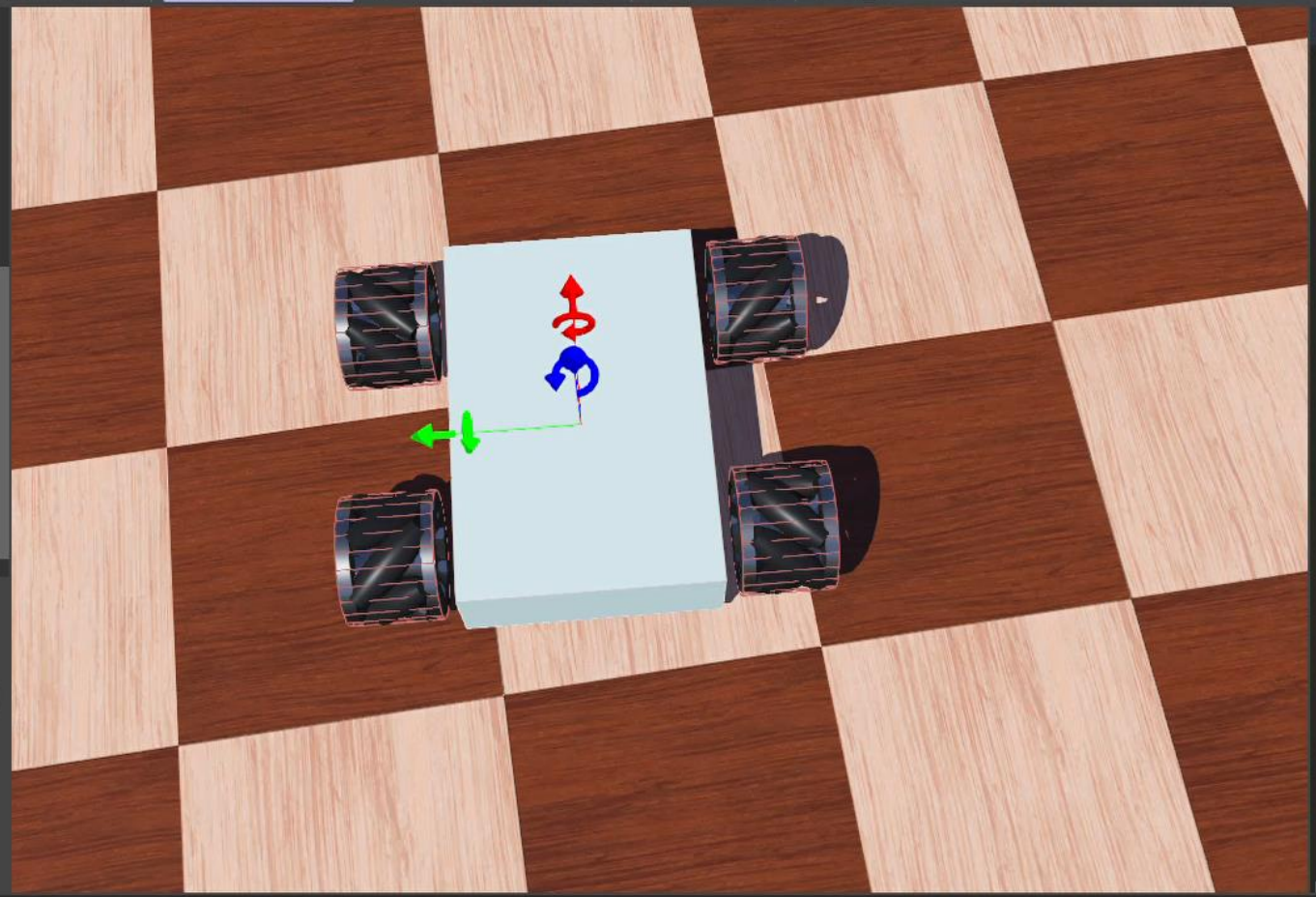
0:00:00:000 - 0.00x

- boundingObject USE Body
- physics Physics
- locked FALSE
- translationStep 0.01
- rotationStep 0.262
- radarCrossSection 0
- recognitionColors
- controller "main"
- controllerArgs
- customData ""
- supervisor FALSE
- synchronization TRUE
- battery
- cpuConsumption 10
- selfCollision FALSE
- showWindow FALSE
- window ""

Selection: controller (String)

main

选择... 编辑



E:\car\controllers\main\main.cpp

main.cpp

```
1
2  #include <webots/Robot.hpp>
3  #include <webots/Motor.hpp>
4  #include <webots/Keyboard.hpp>
5  #include <iostream>
6
7  #include <algorithm>
8  #include <iostream>
9  #include <limits>
10 #include <string>
11
12 using namespace std;
13 using namespace webots;
14
15
16
17 int main() {
18           Motor *motors[4];
19           webots::Keyboard keyboard;
20           char wheels_names[4][8] = { "m
21
22           double speed1[4];
23           double speed2[4];
24           double velocity = 10;
25           Robot *robot = new Robot();//'
26           keyboard.enable(1);//
27           for (int i = 0; i < 4; i++)
28           {
29                     motors[i] = robot->
30                     motors[i]->setPosit
31                     motors[i]->setVeloc
32
33                     speed1[i] = 0;
```

Console - All

```
-1:-1
-1:-1
-1:-1
-1:-1
-1:-1
-1:-1
-1:-1
-1:-1
```