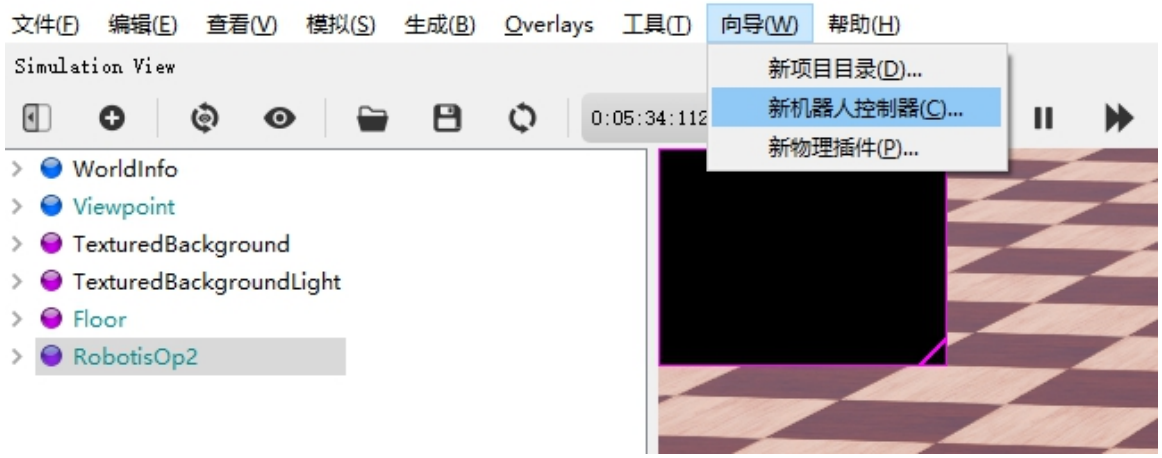
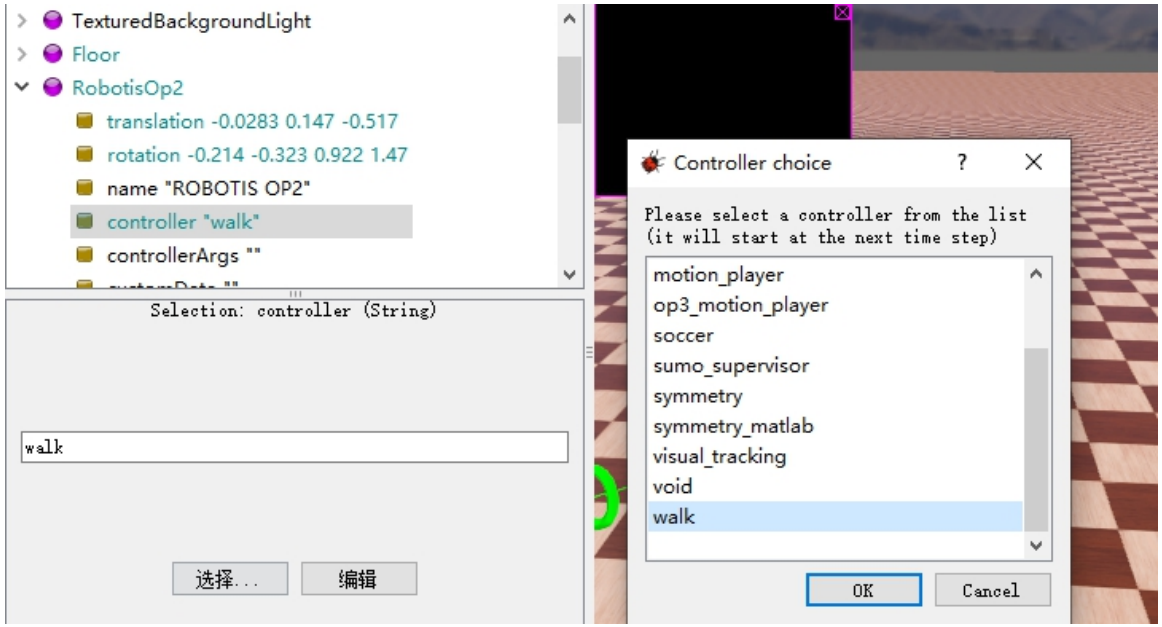
用 Python 代码控制机器人的运动

1 对控制器代码的分析及说明

打开 walk.wbt 文件。点击工具栏-向导-新机器人控制器，选择 Python 语言，完成创建，并在文本编辑器中打开。复制附件代码，覆盖默认代码，保存。



将机器人控制器设置为新建控制器，保存世界。



在 walk.py 中，为了完成对机器人的控制，需要引入几个 Python 库。

```
1. from controller import Robot
2. import os
3. import sys
4.
5. libraryPath = os.path.join(os.environ.get("WEBOTS_HOME"), 'projects', 'robots', 'robotis', 'darwin-op', 'libraries',
6.                            'python37')
7. libraryPath = libraryPath.replace('/', os.sep)
```

```
sys.path.append(libraryPath)
from managers import RobotisOp2GaitManager, RobotisOp2MotionManager
```

其中，对机器人环境进行控制的 Robot 库包含了对各种机器人元件的控制。RobotisOp2GaitManager 与 RobotisOp2MotionManager 库分别为机器人步态控制库与动作组库，分别实现机器人的行走控制与预编动作执行。为了能在 Pycharm 等外部 IDE 中正常运行，导入这两个库前需要添加路径信息。

注意：本段代码适用于 Python 3.7 版本，其他版本的 Python 可能存在兼容性问题。

定义 Walk 类，实现对机器人的控制，初始函数如下。

```
class Walk():
    def __init__(self):
        self.robot = Robot()  # 初始化 Robot 类以控制机器人
        self.mTimeStep = int(self.robot.getBasicTimeStep())  # 获取当前每一个仿真步所仿真时间 mTimeStep
        self.HeadLed = self.robot.getLED('HeadLed')  # 获取头部 LED 灯
        self.EyeLed = self.robot.getLED('EyeLed')  # 获取眼部 LED 灯
        self.HeadLed.set(0xff0000)  # 点亮头部 LED 灯并设置一个颜色
        self.EyeLed.set(0xa0a0ff)  # 点亮眼部 LED 灯并设置一个颜色
        self.mAccelerometer = self.robot.getAccelerometer('Accelerometer')  # 获取加速度传感器
        self.mAccelerometer.enable(self.mTimeSt ep)  # 激活传感器，并以 mTimeStep 为周期更新数值
        self.fup = 0
        self.fdown = 0  # 定义两个类变量，用于之后判断机器人是否摔倒

        self.mGyro = self.robot.getGyro('Gyro')  # 获取陀螺仪
        self.mGyro.enable(self.mTimeStep)  # 激活陀螺仪，并以 mTimeStep 为周期更新数值
```

```
16.
17.         self.positionSensors = []  # 初始化关节角度传感器
18.         self.positionSensorNames = ('ShoulderR', 'ShoulderL', 'ArmUpperR', 'ArmUpperL',
19.                                     'ArmLowerR', 'ArmLowerL', 'PelvYR', 'PelvYL',
20.                                     'PelvR', 'PelvL', 'LegUpperR', 'LegUpperL',
21.                                     'LegLowerR', 'LegLowerL', 'AnkleR', 'AnkleL',
22.                                     'FootR', 'FootL', 'Neck', 'Head')  # 初始化各传感器名
23.
24.         # 获取各传感器并激活，以 mTimeStep 为周期更新数值
25.         for i in range(0, len(self.positionSensorNames)):
26.             self.positionSensors.append(self.robot.getPositionSensor(self.positionSensorNames[i] + 'S'))
27.             self.positionSensors[i].enable(self.mTimeStep)
28.
29.         self.mKeyboard = self.robot.getKeyboard()  # 初始化键盘读入类
30.         self.mKeyboard.enable(self.mTimeStep)  # 以 mTimeStep 为周期从键盘读取
31.
32.         self.mMotionManager = RobotisOp2MotionManager(self.robot)  # 初始化机器人动作组控制器
33.         self.mGaitManager = RobotisOp2GaitManager(self.robot, " Head ")  # 初始化机器人步态控制器
```

定义 Walk 类中的 myStep()函数，完成一个仿真步长的仿真。

```
1.  def myStep(self):
2.      ret = self.robot.step(self.mTimeStep)
3.      if ret == -1:
```

```
        exit(0)
```

定义 Walk 类中的 wait()函数，输入为等待毫秒数，使机器人等待一段时间。

```
def wait(self, ms):
    startTime = self.robot.getTime()
    s = ms / 1000.0
    while (s + startTime >= self.robot.getTime()):
        self.myStep()
```

run()函数通过键盘输入来控制机器人的运动：

```
def run(self):
    print("-------Walk example of ROBOTIS OP2-------")
    print("This example illustrates Gait Manager")
    print("Press the space bar to start/stop walking")
    print("Use the arrow keys to move the robot while walking")
    self.myStep()  # 仿真一个步长，刷新传感器读数

    self.mMotionManager.playPage(9)  # 执行动作组 9 号动作，初始化站立姿势，准备行走
    self.wait(200)  # 等待 200ms

    self.isWalking = False  # 初始时机器人未进入行走状态
```

```
12.
13.     while True:
14.         self.checkIfFallen()  # 判断是否摔倒
15.         self.mGaitManager.setXAmplitude(0.0)  # 前进为0
16.         self.mGaitManager.setAAmplitude(0.0)  # 转体为0
17.         key = 0  # 初始键盘读入默认为0
18.         key = self.mKeyboard.getKey()  # 从键盘读取输入
19.         if key == 32:  # 如果读取到空格，则改变行走状态
20.             if (self.isWalking):  # 如果当前机器人正在走路，则使机器人停止
21.                 self.mGaitManager.stop()
22.                 self.isWalking = False
23.                 self.wait(200)
24.             else:  # 如果机器人当前停止，则开始走路
25.                 self.mGaitManager.start()
26.                 self.isWalking = True
27.                 self.wait(200)
28.         elif key == 315:  # 如果读取到‘↑’，则前进
29.             self.mGaitManager.setXAmplitude(1.0)
30.         elif key == 317:  # 如果读取到‘↓’，则后退
31.             self.mGaitManager.setXAmplitude(-1.0)
32.         elif key == 316:  # 如果读取到‘←’，则左转
33.             self.mGaitManager.setAAmplitude(-0.5)
34.         elif key == 314:  # 如果读取到‘→’，则右转
35.             self.mGaitManager.setAAmplitude(0.5)
36.         self.mGaitManager.step(self.mTimeStep)  # 步态生成器生成一个步长的动作
37.         self.myStep()  # 仿真一个步长
```

checkIfFallen 函数用于检测机器人是否倒地，并完成倒地起身动作：

```
1.      def checkIfFallen(self):
2.          acc_tolerance = 60.0
3.          acc_step = 100  # 计数器上限
4.          acc = self.mAccelerometer.getValues()  # 通过加速度传感器获取三轴的对应值
5.          if acc[1] < 512.0 - acc_tolerance :  # 面朝下倒地时 y 轴的值会变小
6.              self.fup += 1  # 计数器加 1
7.          else :
8.              self.fup = 0  # 计数器清零
9.          if acc[1] > 512.0 + acc_tolerance : # 背朝下倒地时 y 轴的值会变大
10.             self.fdown += 1 # 计数器加 1
11.         else :
12.             self.fdown = 0 # 计数器清零
13.
14.         if self.fup > acc_step :   # 计数器超出上限，即倒地时间超过 acc_step 个仿真步长
15.             self.mMotionManager.playPage(10) # 执行面朝下倒地起身动作
16.             self.mMotionManager.playPage(9) # 恢复准备行走姿势
17.             self.fup = 0 # 计数器清零
18.         elif self.fdown > acc_step :
19.             self.mMotionManager.playPage(11) # 执行背朝下倒地起身动作
20.             self.mMotionManager.playPage(9) # 恢复准备行走姿势
21.             self.fdown = 0 # 计数器清零
```

通过加速度传感器获取机器人 y 轴的加速度值，当其值大于/小于某值一段时间后，判断机器人背部朝下/面部朝下摔倒，然后执行对应的起身动作。

主函数定义如下。

```
if __name__ == '__main__':
    walk = Walk()  # 初始化 Walk 类
    walk.run()  # 运行控制器
```

2. 对代码进行修改来控制机器人的运动

修改 run()函数如下，可以使机器人自动行走，不借助键盘输入。

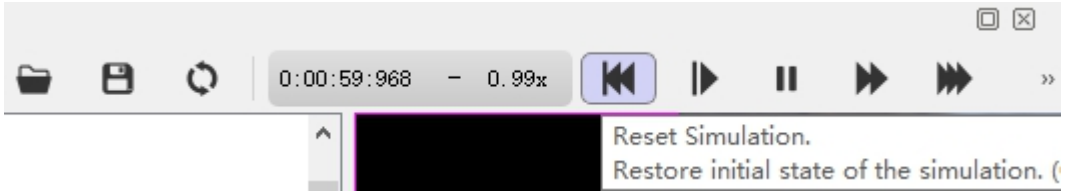
```
def run(self):
    self.myStep()  # 仿真一个步长，刷新传感器读数

    self.mMotionManager.playPage(9)  # 执行动作组 9 号动作，初始化站立姿势，准备行走
    self.wait(200)  # 等待 200ms

    self.isWalking = False  # 初始时机器人未进入行走状态
    while True:
        self.checkIfFallen()  # 判断是否摔倒
        self.mGaitManager.start()  # 步态生成器进入行走状态
        self.mGaitManager.setXAmplitude(1.0)  # 设置机器人前进
        self.mGaitManager.step(self.mTimeStep)  # 步态生成器生成一个步长的动作
        self.myStep()  # 仿真一个步长
```

修改后保存代码文本，如下图。

之后重新运行仿真，就可以看到机器人开始自动前进。



While 循环中的 `self.mGaitManager.setXAmplitude(1.0)`语句使机器人持续向前走动，若想让机器人在适当位置停止，可以修改 while 循环的条件。

3 常用函数列表

函数名称	用途	备注
mGaitManager.start ()	启动	
mGaitManager.stop ()	停止	
mGaitManager.step (t)	生成一段时间的步态规划	t 的单位为毫秒
mGaitManager.setXAmplitude (double X)	前进/后退	X 影响脚步向前的长

		度，它可以取-1 到 1 之间的任何值
mGaitManager.setYAmplitude (double Y)	左移/右移	Y 影响脚步在侧面方向上的长度，它可以取-1 到 1 之间的任何值
mGaitManager.setAAmplitude (double A)	左转/右转	A 影响步态的角度并允许机器人在行走过程中旋转，它可以取 0 到 1 之间的任何值
mKeyboard.getKey()	获取键盘输入	获取的值为输入字符对应的 ASCII 码
wait (int t)	等待	t 的单位为毫秒
mMotionManager.playPage()	执行对应动作	playPage 为一系列封装好的机器人动作文件

4 playPage 部分动作列表

动作组号码	名称	描述	初始姿态

9	walkready	准备行走	直立
10	f up	起身	面部朝下倒地
11	b up	起身	背部朝下倒地

更多的动作可以查询 Webots User Guide 中 Robots- Robotis OP2 的相关文档。

5 修改配置文件来调整机器人的站姿与步态

从 Webots\projects\robots\robotis\darwin-op\controllers\walk 路径下找到 config.ini，复制一份到控制器代码所在路径下。之后在代码初始化部分通过 `self.mGaitManager = RobotisOp2GaitManager(self.robot, "config.ini")` 语句加载配置文件。可以使用记事本打开，如下图：

config.ini - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
[Walking Config]
x_offset                    = -10.0;
y_offset                    = 5.0;
z_offset                    = 20.0;
roll_offset                 = 0.0;
pitch_offset                = 0.0;
yaw_offset                  = 0.0;
hip_pitch_offset            = 13.0;
period_time                 = 600.0;
dsp_ratio                   = 0.1;
step_forward_back_ratio     = 0.28;
foot_height                 = 40.0;
swing_right_left            = 20.0;
swing_top_down              = 5.0;
pelvis_offset               = 3.0;
arm_swing_gain              = 1.5;
balance_knee_gain           = 0.3;
balance_ankle_pitch_gain    = 0.9;
balance_hip_roll_gain       = 0.0;
balance_ankle_roll_gain     = 0.0;

[Robot Config]
time_step                   = 16.0;
camera_width                = 320.0;
camera_height               = 240.0;
```

可以修改配置文件中的各项数值来调整机器人的站姿、运动时的步态。各参数的具体解释可以查询 Webots User Guide 中 Robots- Robotis OP2 的附录部分。