Wolfgang Robot Model Description

Jasper Güldenstein

May 24, 2021

This document presents the Webots simulation model for the virtual RoboCup competition. The model is based on the real robot we use in the physical competition.

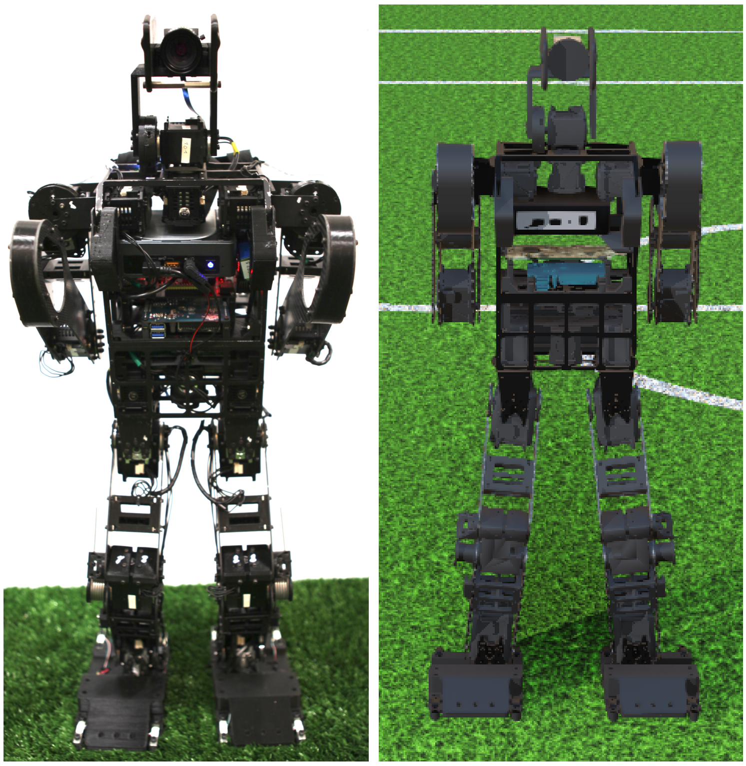

Figure 1: Real robot and robot model in simulation

1 Sensors

1.1 Position Sensors

All used Dynamixel motors (see 2) have a 12 bit hall effect sensor. Therefore the resolution of all position sensors is:

$$\frac{2\pi}{2^{12}} = 0.00153398078$$

1.2 IMU

The real robot has 2 MPU6500 6-axis IMUs[1] installed. One is in the torso and the other below the camera on the head. The position of the IMUs is the same in the model. The method to derive the Look Up Table for the model is described in the sections.

1.2.1 Accelerometer

The Accelerometer noise is defined on page 8 of the datasheet. The *Power Spectral Density* is defined as $300\mu g/\sqrt{\text{Hz}}$ from the sampling rate of the simulation of 125Hz we get a value of:

$$\frac{300\mu g}{\sqrt{Hz}} \cdot \sqrt{125Hz} = 3354.10196625\mu g = 0.03290374028m/s^2$$

The full scale of the device is $\pm 16g = 156.96m/s^2$.

The feature of mapping from one range of input values to sensor readings is not used since it does not provide any benefit.

Since the noise is defined as a fraction of the output, the values are interpolated in the definition. The script we used to calculate this interpolation can be found attached in 3.1.

The resolution is defined by the range divided by the 2^{16} since the device has a 16 bit ADC.

$$\frac{2 \cdot 156.96m/s^2}{2^{16}} = 0.00479003906m/s^2$$

1.2.2 Gyroscope

The gyroscope noise is defined on page 7 of the datasheet. The *Rate Noise Spectral Density* is defined as $0.01°/\text{s}/\sqrt{\text{Hz}}$ from the sampling rate of the

[1] `https://invensense.tdk.com/download-pdf/mpu-6500-datasheet/`

simulation of 125Hz we get a value of:

$$\frac{\frac{0.01^\circ}{\text{s}}}{\sqrt{Hz}} \cdot \sqrt{125Hz} = 0.11180339887^\circ/s = 0.001951337425197167 rad/s$$

The full scale range of the device is $\pm 2000^\circ/s = 34.90659 rad/s$.

The feature of mapping from one range of input values to sensor readings is not used since it does not provide any benefit.

Since the noise is defined as a fraction of the output, the values are interpolated in the definition. The script we used to calculate this interpolation can be found attached in 3.1.

The resolution is defined by the range divided by the 2^{16} since the device has a 16 bit ADC.

$$\frac{2 \cdot 34.90659 rad/s}{2^{16}} = 0.00106526458 rad/s$$

1.3 Touch Sensor

Just like the real robot, the model has four 1D force sensors on each foot.0.000003 We experimentally verified that we have a resolution of between $0.000003N$ and $0.000023N$ in our feet sensors. We assume a conservative estimate of $0.00005N$.

1.4 Camera

We use a single RGB camera on the robot's head with a resolution of 1440x1080 pixel, color depth of 1 byte with 3 color channels, and request the image every 64ms leading to a bandwidth used by 4 robots of:

$$1440 \cdot 1080 \cdot 3B \cdot 4 \cdot \frac{1}{0.064s} = 291,600,000 B/s = 291.6 MB/s$$

2 Actuators

The robot uses the Dynamixel MX64[2] in arms and its head, MX106[3] in its legs and XH540-W270[4] in its knees motors. The provided values for `maxTorque`, `maxVelocity`, `staticFriction`, and `dampingConstant` were used.

`backlash` was set to 0.01 and gearMass was set to 0.001 as in the provided Darwin robot[5].

3 Appendix

3.1 Python script to generate Lookup Tables

```
def noise_table(my_min, noise, iters, indent):
        print(indent * " ", end="")
        print("lookupTable [")
        for i in range(iters):
                val = my_min/2 ** i
                print(indent * " ", end="")
                print(f"  {val:f} {val:f} {noise/-val:f},")
        print(indent * " ", end="")
        print(f"  0 0 0")
        for i in range(iters):
                val = -my_min/2 ** (iters-i-1)
                print(indent * " ", end="")
                print(f"  {val:f} {val:f} {noise/val:f},")
        print(indent * " ", end="")
        print("]")

accel_min = -156.96
accel_noise =  0.03290374028
gyro_min = -34.90659
gyro_noise = 0.00106526458

noise_table(accel_min, accel_noise, 12, 12)
noise_table(gyro_min, gyro_noise, 12, 12)
```

[2] https://emanual.robotis.com/docs/en/dxl/mx/mx-64-2/

[3] https://emanual.robotis.com/docs/en/dxl/mx/mx-106-2/

[4] https://emanual.robotis.com/docs/en/dxl/x/xh540-w270/

[5] https://github.com/cyberbotics/webots/blob/fc4f769b69300530e181feea8b157f039e2a4017/projects/robots/robotis/darwin-op/protos/Darwin-opHinge2.proto