NAO Robot: Connections, Sensors, and Best Practices

Eng. Ahmed Métwalli

25 December 2024

Contents

NAOqi Venv Setup Guide

1 Types of Connections with NAO

NAO robot supports multiple connection types for interaction and programming. Below are the primary connection types:

- **Wi-Fi Connection:** Used for wireless communication with NAO through a local network.
- **Ethernet Connection:** A direct wired connection for stable communication and programming.
- **USB Connection:** Allows for transferring files or firmware updates.
- **Socket Connections:** Used for low-level communication, often between Python 2.7 (for NAOqi) and Python 3.11 for advanced tasks.
- **HTTP/REST API:** Enables remote interaction through web services.

For tasks requiring updated Python versions (e.g., Python 3.11), socket or HTTP connections are essential to bridge the gap between NAOqi's Python 2.7 compatibility and modern environments.

2 Sensors in NAO

NAO is equipped with a range of sensors to perceive its environment and interact effectively. Below is an overview:

2.1 Sensor Types and Principles of Work

1. **Infrared Sensors:**
   - **Principle:** Detects infrared light to measure distances or recognize beacons.
   - **Range of Values:** Typically up to 2 meters.
2. **Sonar Sensors:**
   - **Principle:** Uses ultrasonic waves to measure distances by calculating the time of echo returns.
   - **Range of Values:** 15 cm to 1.2 meters.
3. **Touch Sensors:**
   - **Principle:** Detects pressure changes on specific areas (e.g., head, hands, feet).
   - **Range of Values:** Binary (pressed or not pressed).
4. **Cameras:**
   - **Principle:** Captures visual data for image processing and object recognition.

    - **Range of Values:** 640x480 resolution at 30 fps.

5. **Gyroscope and Accelerometer:**
    - **Principle:** Measures orientation and acceleration for balance and movement.
    - **Range of Values:** $-2g$ to $+2g$ for accelerometer.

6. **Microphones:**
    - **Principle:** Captures audio signals for voice recognition and environmental sound analysis.
    - **Range of Values:** Frequency range 50 Hz to 16 kHz.

2.2 Environmental Operating Range

- **Temperature:** Operates best between 10C to 35C.
- **Humidity:** 20% to 80% non-condensing.
- **Illumination:** Requires moderate light levels for camera-based tasks.

2.3 Limitations

- Infrared sensors may face interference in direct sunlight.
- Sonar sensors may provide inaccurate readings on soft or uneven surfaces.
- Cameras require proper lighting for accurate recognition.
- Microphones are susceptible to noise interference in loud environments.

3 Best Practices

3.1 What to Do

- Ensure stable Wi-Fi or Ethernet connection for optimal performance.
- Keep the robot in its specified temperature and humidity range.
- Regularly clean cameras and sensors for accurate readings.
- Use socket or HTTP connections for advanced integrations between Python 2.7 and Python 3.11 environments.

3.2 What Not to Do

- Avoid exposing the robot to extreme temperatures or humidity.
- Do not use sensors in conditions beyond their operating range.
- Do not use abrasive materials to clean the sensors or cameras.
- Avoid physical shocks that may damage internal sensors.

4 Python Code for NAO Robot Interaction

Listing 1: Python Code for Text-to-Speech and Audio Recording

```
import naoqi
from naoqi import ALProxy
import paramiko
import time

# Variables:
IP = "1.1.1.21" #'10.1.95.58' #1.1.1.21
soundRecordPath = "/home/nao/audio.wav" # Recording path stored in Nao
localPath = "audio.wav" # Our target path on our device

MODEL_TTS = "ALTextToSpeech"
PORT = 9559
MODEL_RECORDER = "ALAudioRecorder"

tts = ALProxy(MODEL_TTS, IP, PORT)
tts.say("what can I do for you Ahmed")

USER = 'nao'
PASS = 'nao'
AUDIO_FILE_TYPE = "wav"
SPEAKER = (0,0,1,0)
SAMPLING_FREQUENCY = 16000
DELAY = 10
audio_recorder = ALProxy(MODEL_RECORDER, IP, PORT)
audio_recorder.startMicrophonesRecording(soundRecordPath, AUDIO_FILE_TYPE,
time.sleep(DELAY)
audio_recorder.stopMicrophonesRecording()

ssh_client = paramiko.SSHClient() # SSH
ssh_client.set_missing_host_key_policy(paramiko.AutoAddPolicy())
ssh_client.connect(hostname=IP, port=22, username=USER, password=PASS)
ftp_client = ssh_client.open_sftp() # Secure File Transfer Protocol
ftp_client.get(soundRecordPath, localPath)

ftp_client.close()
ssh_client.close()
```

5 Conclusion

Understanding and respecting the connection options, sensor capabilities, and environmental requirements of NAO ensures optimal performance and longevity. Proper maintenance and adherence to best practices will maximize the robot's functionality and lifespan.