Activity Recognition System

Overview

This project describes the implementation of an activity recognition system using ESP8266 microcontroller. It uses data from an accelerometer and a gyroscope to recognize three types of activities: none, walking, and jumping. A complementary filter is used to obtain the board orientation and extract the vertical and horizontal components from accelerometer data. Based on the standard deviation of each component, all 3 activites can be recognized. The system uses an HTTP server to display the recognized activity as well as sensor measurements. On top of that, a 3D visualization of the board's orientation is displayed using WebGL.

Methods

Calibration

Due to sensor errors such as drift, the board first needs to be calibrated. Over 3 seconds, the system gathers 300 samples of accelerometer and gyroscope data. During that time, the board needs to be completely still. Furthermore, the board must be oriented horizontally, because the direction of gravity has to be accounted for. The samples are then averaged to obtain an expected error, which is used to correct further gyroscope and accelerometer readings.

Main loop

The first step is to obtain the board's orientation in the form of a roll and pitch angle. For this, we combine accelerometer and gyrosope readings. An accelerometer gives a good indication of orientation in static, slowly changing conditions. Conversely, a gyroscope is good for determining tilt in dynamic, rapidly changing conditions. Therefore, we can pass the accelerometer readings through a low-pass filter and gyroscope through a high-pass filter, then combine the results to obtain accurate orientation data. The roll angle φ and pitch angle θ are obtained from accelerometer data using the forumla as described in [2]:

$$\tan\varphi = \frac{a_y}{a_z} \qquad \tan\theta = \frac{-a_x}{\sqrt{{a_y}^2 + {a_z}^2}}$$

Accelerometer and gyroscope readings are then combined using the following equation as described in [1]:

$$angle_t = \alpha * \left(angle_{t-1} + \omega_{gyro} * \Delta t\right) + (1-\alpha) * angle_{acc}$$

$$\alpha = \frac{\tau}{\tau + \Delta t}$$

Using this method, we obtain a roll angle φ and a pitch angle θ of the board with respect to the direction of gravity.

The next step is to extract vertical and horizontal components from the accelerometer readings. This is done by multiplying the acceleration vector by inverse rotation matrices of roll φ and pitch θ:

$$a_{abs} = {R_\theta}^{-1} * {R_\varphi}^{-1} * a_{acc} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi & \cos\varphi \end{bmatrix} * a_{acc}$$

Here, a_{abs_z} represents the vertical (in the same/opposite direction as gravity) component and $\sqrt{{a_{abs_x}}^2 + {a_{abs_y}}^2}$ represents the horizontal (perpendicular to gravity) component of acceleration.

Finally, the standard deviation of acceleration values from the last 2 seconds is computed. If the standard deviation of the vertical acceleration component is above 4.0 (and larger than the horizontal component), the current activity is interpreted as jumping. Similarly, walking is detected if the standard deviation of the horizontal acceleration component is above 1.5.

Server

The server has 4 HTTP endpoints. "/" is the home page and sends the user HTML code. "/script.js" returns the JavaScript code that is used on the website. "/data" and "/history" both return JSON objects for orientation and acceleration data, respectively.

Frontend

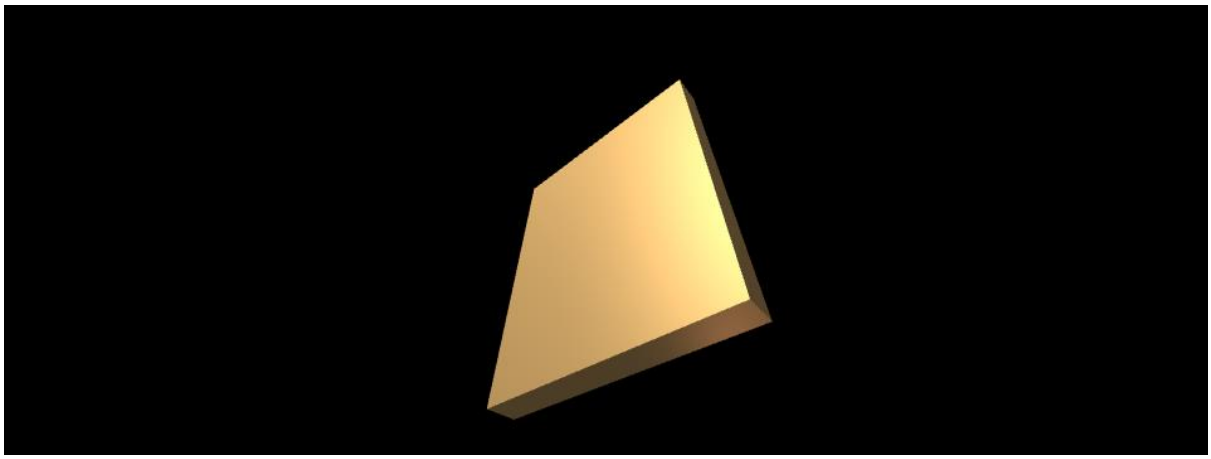
The frontend periodically requests data from "/data" and "/history" in order to display information in real time. Chart.js, a JavaScript library is used to draw graphs of acceleration values. The library is not included locally due to its size, so internet access is required in order to use it. WebGL 2 is used to display the orientation of the board in 3D. A modern web browser such as Firefox or Chrome is required to correctly display WebGL 2 content.

Results

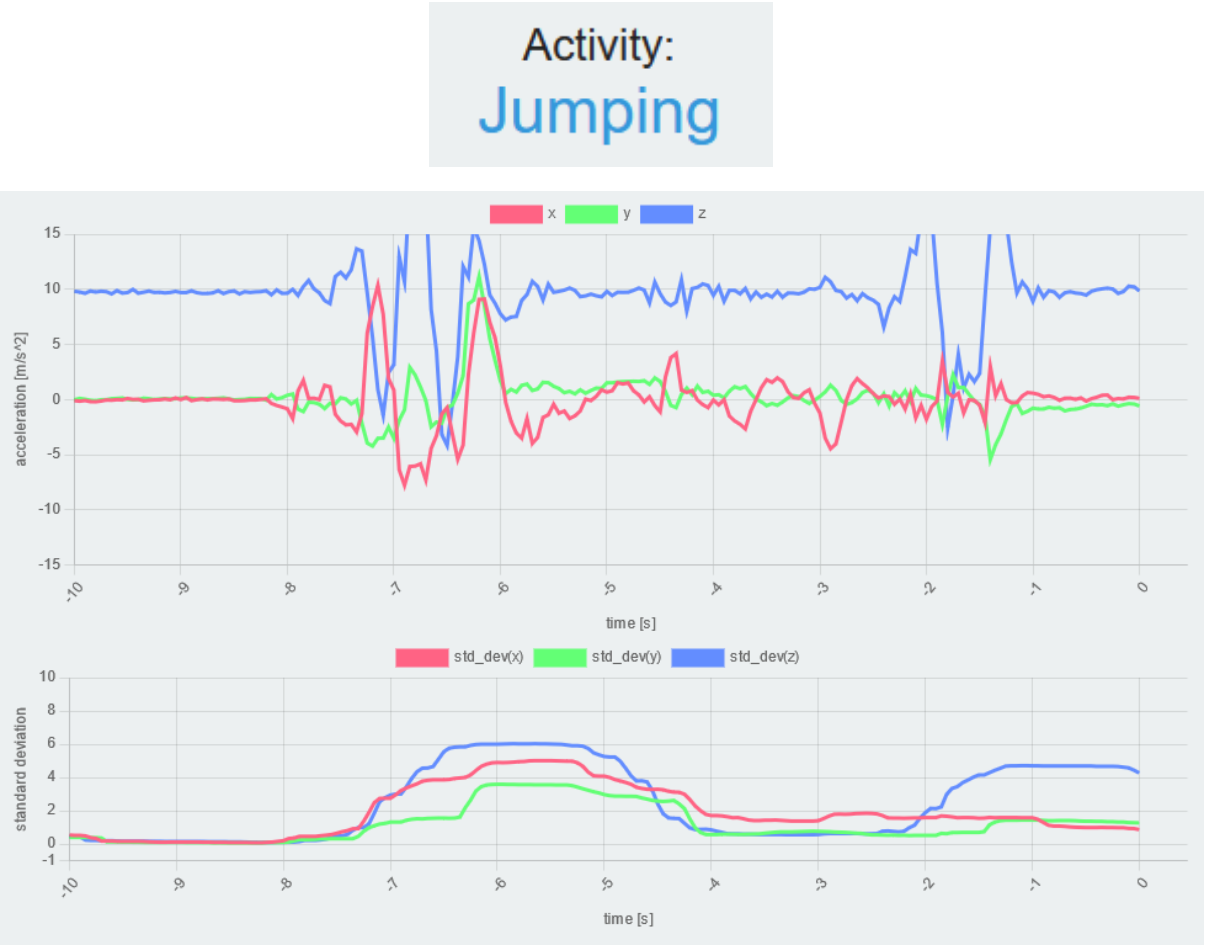
Example serial monitor output

```
Connecting to access point... Connected!
Local IP: 192.168.100.33
HTTP server started
Make sure that the board is in a horizontal position during calibration!
Calibrating, please don't move the board for a few seconds... Done!
Activity: Walking
Activity: Jumping
Activity: Walking
Activity: None
Activity: Walking
```

Example orientation output



Example acceleration output



References

[1] Complementary filter https://sites.google.com/site/myimuestimationexperience/filters/complementary-filter

[2] Tilt Sensing Using a Three-Axis Accelerometer https://www.nxp.com/files-static/sensors/doc/app_note/AN3461.pdf