DAYANANDA SAGAR COLLEGE OF ENGINEERING
(An Autonomous Institute affiliated to Visvesvaraya Technological University (VTU), Belagavi,
Approved by AICTE and UGC, Accredited by NAAC with 'A' grade & ISO 9001 – 2015 Certified Institution)
Shavige Malleshwara Hills, Kumaraswamy Layout, Bengaluru-560 111, India

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Skill Development Program on Signal & Image processing with Embedded hardware integration

(25/03/25 to 27/03/25)

Report on the topic

Step Counter using MATLAB

Submitted by:

Sl. No.	USN	NAME
1.	1DS23EC194	SHAIK SHAHEEN
2.	1DS23EC197	SHARON ROLLY
3.	1DS23EC203	SHREYA PRASHANTH SHETTY
4.	1DS23EC204	SHREYA RAO

Table of Contents:

1. Introduction

In today's world, where most of us carry smartphones everywhere, these devices have quietly become powerful tools for monitoring our daily habits — including how much we move. One of the simplest yet most commonly tracked metrics is the step count. Whether you're going for a walk, climbing stairs, or just moving around your home, your phone can keep track of your steps using its built-in sensors.

This project explores how we can take raw motion data from a phone and use it to count steps accurately. Most modern phones have an accelerometer — a sensor that detects movement along three directions: left-right (X), up-down (Y), and forward-backward (Z). By analyzing how much the phone moves over time, we can identify patterns that match the way people walk.

Instead of relying on any third-party fitness app or hardware like a smartwatch, we processed the phone's sensor data ourselves. We used techniques like computing the overall motion strength (magnitude), filtering out noise, and detecting peaks that represent steps. The goal was to build a method that is simple, reliable, and easy to implement in a basic MATLAB script.

2. Methodology

a. Data Acquisition

We began by loading a recorded log file, which contains motion data collected from a mobile phone. This file includes acceleration readings along the X, Y, and Z axes as well as corresponding timestamps. The data was captured using a phone app and saved in .mat format for analysis.
Sample Rate (fs): 10 Hz
Cutoff Frequency (fc): 3 Hz

b. Preprocessing

To analyze the motion signal, we first computed the overall acceleration magnitude at each time point using the Euclidean norm:

Acceleration Magnitude = $\text{sqrt}(x^2 + y^2 + z^2)$

This calculation reduces the three-axis data to a single time-series signal that represents the intensity of motion regardless of direction. To enhance visibility of periodic patterns like walking, we centred the signal around zero by subtracting its mean.

c. Signal Filtering

The raw acceleration magnitude often contains noise from small, irrelevant movements or vibrations. To remove this high-frequency noise and preserve only the low-frequency components that correspond to actual steps, we applied a second-order Butterworth low-pass filter with a cutoff frequency of 3 Hz. The filtfilt function was used to perform zero-phase filtering, which prevents distortion in the signal timing and shape.

d. Step Detection

With the filtered signal, we performed peak detection using MATLAB's findpeaks function. To ensure reliability, we used two constraints:

- Each detected peak must be higher than the mean plus one standard deviation of the filtered signal.
- The minimum time between successive peaks must be at least 0.5 seconds (based on a 10 Hz sample rate, this corresponds to a MinPeakDistance of 5 samples).

These conditions help filter out false positives due to jitter or noise, and make the algorithm suitable for detecting regular walking patterns.

3.Implementation in MATLAB

/MATLAB Drive/MobileSensorData/steps.m

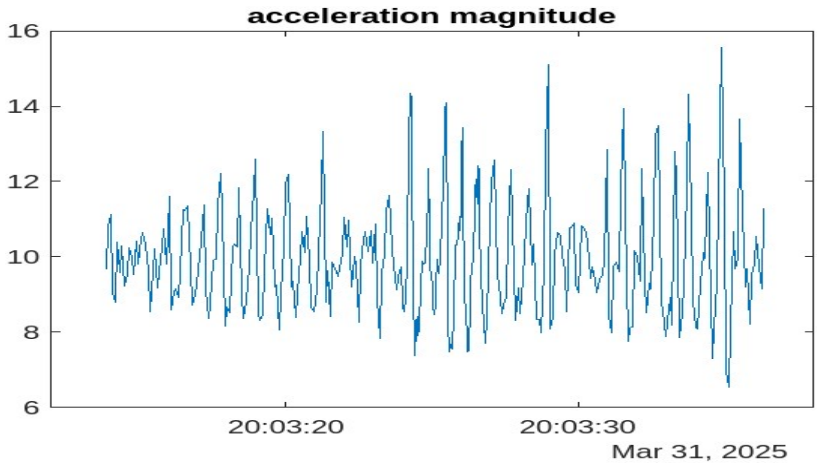
```
m=load("/MATLAB Drive/MobileSensorData/sensorlog_20250331_200313.mat");
x = m.Acceleration.X;
y = m.Acceleration.Y;
z = m.Acceleration.Z;
t_s=m.Acceleration.Timestamp;
mag = sqrt(sum(x.^2 + y.^2 + z.^2, 2)); % to find magnitude of acceleration
plot(t_s,mag)
mag_2 = mag-mean(mag); %to centre it around 0
fs = 10;
fc = 3; % Cutoff frequency for low-pass filter
[b, c] = butter(2, fc/(fs/2), 'low');
Acc_Mag_Filtered = filtfilt(b, c, mag);
[pks, locs] = findpeaks(Acc_Mag_Filtered, 'MinPeakHeight', mean(Acc_Mag_Filtered) + std(Acc_Mag_Filtered), 'MinPeakDistance', fs/2);
step_count = length(locs);
figure;
plot(Acc_Mag_Filtered);
hold on;
plot(locs, pks, 'ro');
title(['Step Count: ', num2str(step_count)]);
xlabel('Sample Index');
ylabel('Filtered Acceleration Magnitude');
grid on;
hold off;
```

4. Results and Visualization

From the filtered signal, we identified peaks that correspond to steps. The number of peaks gives us the total estimated step count. This value was stored in the variable step_count.

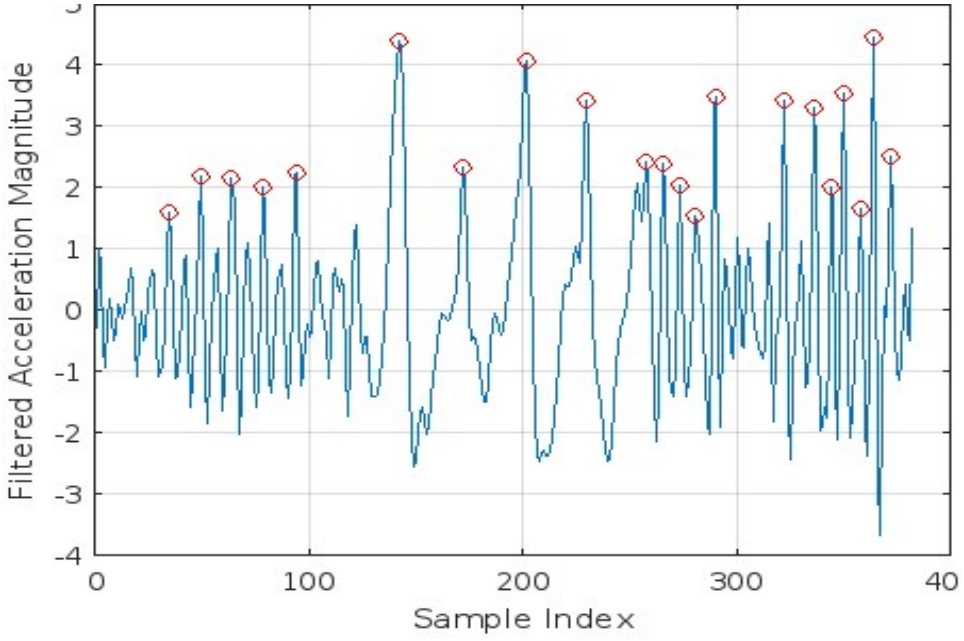
To verify the accuracy of the detection visually, two plots were generated:

a. Raw Acceleration Magnitude vs Time



This plot shows the unfiltered motion magnitude over time, which gives a broad view of how the phone was moved during the recording.

b. **Filtered Signal with Detected Steps**



c. This plot shows the filtered acceleration magnitude, with detected peaks marked using red circles. This provides a clear representation of how step events align with the smoothed motion data.

5. Conclusion

This report presented a straightforward approach to step counting using a phone's internal motion sensors. By processing acceleration data and applying basic signal analysis techniques, we were able to estimate steps with reasonable accuracy.

While this method is simple and easy to implement, it can serve as a solid foundation for future improvements. These might include handling different walking styles, using gyroscope data for better step detection, or implementing the algorithm for real-time use on mobile devices.

5. References

[1] MathWorks. (n.d.). *filtfilt function*. Retrieved from: https://www.mathworks.com/help/signal/ref/filtfilt.html

[2] MathWorks. (n.d.). *findpeaks function*. Retrieved from: https://www.mathworks.com/help/signal/ref/findpeaks.html

[3] Patel, S., Park, H., Bonato, P., Chan, L., & Rodgers, M. (2012). A review of wearable sensors and systems with application in rehabilitation. *Journal of NeuroEngineering and Rehabilitation*, 9(1), 21. https://doi.org/10.1186/1743-0003-9-21

[4] Wang, N., Ambikairajah, E., & Celler, B. G. (2010). Accelerometry based classification of walking patterns using time-frequency analysis. In *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology* (pp. 2230-2233). IEEE.