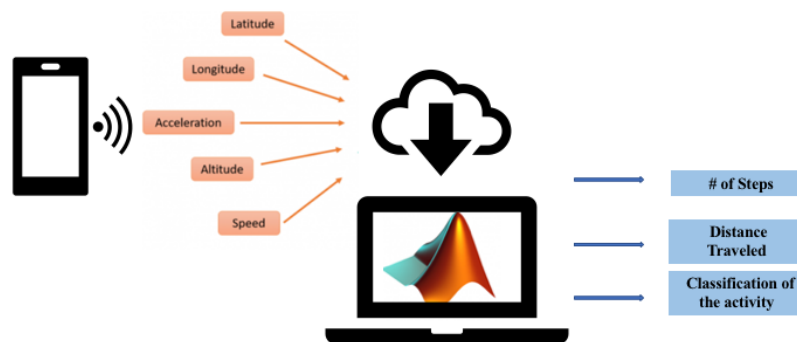


## Project 3

### Building Your First IoT Application: Physical Activity Monitoring System Via a Mobile Device

Fitness trackers, also known as physical activity trackers, are Internet of Things (IoT) applications that use sensors to collect data and software algorithms to process the information gathered. These devices have several functions, including counting steps, calculating distances, estimating burnt calories, and monitoring heart rate and sleep cycles. Physical activity data can be collected through sensors embedded in mobile devices or via smart bands and watches, which are wearable devices that can be connected to mobile phones or other devices.

In this project, you are expected to use your mobile devices and MATLAB Mobile to gather physical activity data, use MATLAB to process the data, and report your work, following the instructions in the following sections of this document. You are tasked with designing an application that not only counts the number of steps and distance traveled but also classifies and records the type of movement being performed.



#### 1 MATLAB Mobile and Data Collection from the Sensors

Download MATLAB to your computer and MATLAB Mobile to your mobile device. You can find MATLAB Mobile at Google Play for Android and App Store for IOS.

Open your MATLAB Drive account (with your METU email address) on MATLAB Mobile.

Create mobiledev object on MATLAB Mobile to acquire data from sensors.

```
% Create mobiledev object.
m=mobiledev;
```

 From which sensors can MATLAB Mobile obtain data? Briefly describe the outputs of each of those sensors.

From "Sensors-Settings-Stream to", select "Log" option to save the activity data as a log file. From "Sensors-Settings-More-MATLAB Drive Upload-Auto Upload", select "Wifi" or "Wifi and Cellular".

Turn on acceleration and position sensors. You may need to give permissions to the MATLAB Mobile application on your mobile device.

Under the "Sensors" menu, click on the "START" button to start collecting data and click on the "STOP" button to finish collecting data and save it as a log file. The file is saved on your device and your MATLAB Drive account. You will use that file to extract information on your physical activity.

Check [1, 2] for further information.

For this project, you should record the sensor data from three minutes of activity in the open air. These three minutes should consist of one minute of standing, one minute of slow walking, and one minute of fast walking/running in arbitrary order. State the order of your selection in your report.

## 2 Data Processing

Load your own physical activity log data.

```
% Load your log data.  
load('activity_data.mat');
```

Extract position and acceleration sensor data.

```
% Required position data.  
lat=Position.latitude;  
lon=Position.longitude;  
positionDatetime=Position.Timestamp;  
spd = Position.speed;
```

```
% Required acceleration data.  
Xacc = Acceleration.X;  
Yacc = Acceleration.Y;  
Zacc = Acceleration.Z;  
accelDatetime=Acceleration.Timestamp;
```

Convert type of data from datetime to double using "timeElapsed" function provided for the project.

```
% Time conversion.  
positionTime=timeElapsed(positionDatetime);  
accelTime=timeElapsed(accelDatetime);
```

Calculate the total distance during physical activity.

```
% Distance.
earthCirc = 40175017; % The equatorial circumference of Earth, m
totaldis = 0; % m
```

```
for i = 1: (length(lat)-1)
    lat1 = lat(i);
    lat2 = lat(i+1);
    lon1 = lon(i);
    lon2 = lon(i+1);
    diff_ = distance(lat1, lon1, lat2, lon2);
    dis = (diff_/360)*earthCirc;
    totaldis = totaldis + dis;
end
```

```
disp(['The total distance traveled is: ', num2str(totaldis), ' meters']);
```

Estimate and display the number of steps during the physical activity using calculated distance and your average step length.

```
% Number of steps, based on position data.
```

```
disp(['You took ', num2str(step_number) ' steps. (based on position data)']);
```

Plot the acceleration with respect to time for different axes (x, y, z).


```
% Plot of acceleration in different axis.
figure('units','normalized','outerposition',[0 0 1 1]);
plot(accelTime,Xacc);
hold on;
plot(accelTime,Yacc);
plot(accelTime,Zacc);
xlim([0 50])
legend('X Acceleration','Y Acceleration','Z Acceleration');
xlabel('Time (s)')
ylabel('Acceleration (m/s^2)');
title('Acceleration Data Vs. Time');
grid;
hold off
```

To visualize the general changes in acceleration regardless of device orientation, calculate and plot the magnitude of the 3D acceleration with respect to time.

```
% Calculate the magnitude of acceleration here as a variable called acc_mag.
```

```
acc_mag = % Your code here
```

```
figure('units','normalized','outerposition',[0 0 1 1]);
plot(accelTime, acc_mag);
xlabel('Time (s)');
ylabel('Acceleration (m/s^2)');
title('Magnitude of Acceleration Vs. Time');
grid;
```

 Can you observe a constant bias on the data from accelerometer even while you are standing still? Estimate its value and comment on the reason behind it.

Remove the effect of the bias on acceleration. Then, find the peak positions and show them on the acceleration plot. Estimate the number of steps during physical activity using the number of peaks in the acceleration data.

**% Remove the bias on acceleration using your code.**

```
constant_effect = % Your code here
mag_0 = % Your code here
```

**% Peak positions in acceleration.**

```
minPeakHeight = std(mag_0); % standard deviation of acceleration.
```

```
[pks,locs] = findpeaks(mag_0,'MINPEAKHEIGHT',minPeakHeight);
```

**% Plot acceleration and peak positions without constant effect.**

```
figure('units','normalized','outerposition',[0 0 1 1]);
```

```
plot(accelTime,mag_0);
```

```
hold on;
```

```
plot(accelTime(locs), pks, 'r', 'Marker', 'v', 'LineStyle', 'none');
```

```
xlabel('Time (s)');
```

```
title('Magnitude of Acceleration Vs. Time, Constant Effects are Excluded');
```

```
grid;
```

```
hold off;
```

**% Step number.**

```
numSteps = numel(pks);
```

```
disp(['You took ', num2str(numSteps) ' steps. (based on acceleration data)']);
```

📖 The data obtained via electronic sensors can be noisy and distorted. Using a noise filter can enhance the output signal quality by reducing the effects of noise and distortion. The moving average is one of the most common methods for filtering. The expression for  $2k+1$  moving average filter is given in Equation 1, where  $N$  is the total length of the data array,  $a_q$  is the  $q^{\text{th}}$  element of the data array and  $\widetilde{a}_n$  is the average.

$$\widetilde{a}_n = \left( \sum_{q=n-k}^{n+k} a_q \right) / (2k + 1), \quad k < n < N - k, \quad \text{Eq. 1}$$

Repeat the last step using 5-point and 11-point moving average filters and estimate the number of steps during physical activity using the number of peaks in the acceleration data.

**% Moving average filter, with  $2k+1$  points.**

```
mag_0 = movmean(mag_0,11);
```

✍ Present the estimated number of steps during this project as a table. Note that there should be three different estimations. Compare and comment on the accuracy of these five estimations.


Plot the speed data. To classify the movement (standing, walking, or running), you can simply use some thresholds on the speed data. For example: speed < 0.3 m/s → no

movement,  $0.3 \text{ m/s} < \text{speed} < 1.8 \text{ m/s} \rightarrow$  walking,  $1.8 \text{ m/s} < \text{speed} \rightarrow$  fast walking and running.

```
% Classification of activity
% Find the time and speed data for different activities
% Hint: You can use logical indexing to extract data
% e.g. data=array1(array2<=threshold)
% Include your code here to find positionTime and spd values for different
% activities
```

```
positionTimeStanding = % Your code here
spdStanding = % Your code here
positionTimeWalking = % Your code here
spdWalking = % Your code here
positionTimeRunning = % Your code here
spdRunning = % Your code here
```


```
figure('units','normalized','outerposition',[0 0 1 1]);
plot(positionTimeStanding, spdStanding, "*");
hold on;
plot(positionTimeWalking, spdWalking, "*");
hold on;
plot(positionTimeRunning, spdRunning, "*");
xlabel('Time (s)');
ylabel('Speed (m/s)');
title("Speed vs Time");
grid;
hold off;
```

 Comment on speed plot and accuracy of physical activity classification based on speed. How can the classification labels be extended, and what are the alternative classification methods?

Visualize the physical activity on a map using the “mapping” function provided to you as the final step.

```
% Mapping
mapping(Position); % Input the position data.
```

 What is the meaning of different colors on the route?

 What are the possible applications of this project work in real world? What are the possible extensions/developments on the project as the next step? (Answer these questions in the conclusion part of the report.)

### 3 Submission

Important dates:

- Assignment date: Monday (12:30), 24<sup>th</sup> of April 2023
- Due date: Sunday (23:59), 7<sup>th</sup> of May 2023

You can work in groups of up to three students.

Details on the evaluation of the project will be shared on ODTUCLASS.

Your report should consist of well-developed sentences and coherent paragraphs. You should add figure captions and spell-check your report before submission.

Your report should have the following subsections:

- *Introduction:* Briefly provide background information and the purpose of this project/report.
- *Data Collection/Processing and Results:* Provide details on your work, present your results as tables and figures, and answer the related questions.
- *Conclusion:* Draw all main points together and discuss possible developments on the project as the next steps.

You should submit one MATLAB script as your code, one ".mat" file as your activity data and one pdf file as your report.

Avoid plagiarism. "The members of the METU community are reliable, responsible and honourable people who embrace only the success and recognition they deserve, and act with integrity in their use, evaluation and presentation of facts, data and documents." [3].

## 4 References

[1] "Create mobiledev object to acquire data from Android", 2022. [Online]. Available: [https://www.mathworks.com/help/matlabmobile\\_android/ref/mobiledev.html](https://www.mathworks.com/help/matlabmobile_android/ref/mobiledev.html).

[2] "Sensor Data Collection with MATLAB Mobile or MATLAB Online", 2022. [Online]. Available: [https://www.mathworks.com/help/matlabmobile\\_android/ug/sensor-data-collection-with-matlab-mobile.html](https://www.mathworks.com/help/matlabmobile_android/ug/sensor-data-collection-with-matlab-mobile.html).

[3] "ACADEMIC INTEGRITY GUIDE FOR STUDENTS", Oidb.metu.edu.tr. [Online]. Available: <https://oidb.metu.edu.tr/sites/oidb.metu.edu.tr/files/Academic%20Integrity%20Guide%20for%20Students.pdf>.