

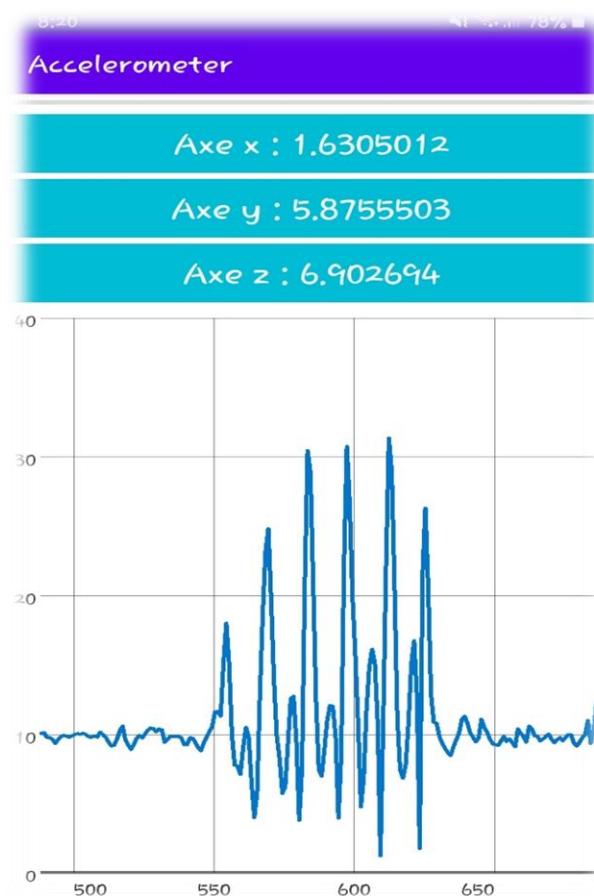
2021/2022

AP

DETECTION DES CHUTES

Département : Génie électrique

2-FI-Energie Electrique Et Industrie Numérique



Réalisé par :

RAFYA HAMZA

DARRAK ASMAA

Encadré par :

Pr.JILBAB

Table des matières

I.	Etude d'un accéléromètre :	4
1.	Introduction :	4
a.	Définition :	4
b.	À quoi sert un accéléromètre :	4
c.	Comment fonctionne un accéléromètre :	5
2.	Les applications de l'accéléromètre :	5
a.	Les chocs :	5
a.	L'accélération vibratoire :	5
b.	L'accélération de mobile :	6
3.	Etude des accéléromètres intégrés dans les smartphones :	6
a.	Fonctionnement de l'accéléromètre du smartphone :	6
b.	L'utilité des accéléromètres utilisés dans les appareils mobiles :	7
➤	Détermination des mouvements de nos téléphones :	7
➤	Détection d'orientation :	7
4.	Etat de l'art de l'application :	7
II.	Mesure des vibrations :	9
1.	Les étapes pour utiliser l'accéléromètre sous Android :	9
a.	Construisez la mise en page de l'application avec textView, barre de progression :	10
b.	Créer des références variables aux éléments de mise en page à l'aide de findViewById : ..	11
c.	Code de démarrage SensorManager :	11
d.	Méthodes onResume et onPause :	11
e.	nouvelles méthodes onSensorChanged onAccuracyChanged :	12
f.	Normaliser le mouvement de l'axe z y z en une seule valeur :	12
g.	Mise à jour de l'interface utilisateur avec de nouvelles valeurs de capteur :	12
h.	Exécutez l'accélération sur un téléphone :	13
i.	Installez GraphView à l'aide du gestionnaire de dépendances Gradle :	13
j.	Ajouter des exemples de données pour le graphique :	14
k.	Suivre les événements de secousse sur l'historique des graphiques :	14
2.	Un programme permettant de mesurer les vibrations du Smartphone :	15
III.	Réalisation de l'application :	18
1.	Exploitation des vibrations pour l'application de détection des chutes :	18
a.	Système complet :	18
b.	Analyse de la méthode proposée :	18
c.	Algorithme proposé de détection de chute :	20

2. Programme de la réalisation de l'application :	20
a. Le programme d'application :	21
IV. Conclusion :	23

Liste des figures :

Figure 1: l'accéléromètre	4
Figure 2 : Principe de fonctionnement d'accéléromètre du smartphone.	6
Figure 3: Analyse temporelle d'une chute.....	8
Figure 4: Cadre général d'un système de détection des chutes.	9
Figure 5: Page de l'application avec textView.	10
Figure 6: l'accélération sur un téléphone.....	13
Figure 7: GraphView sur l'application	13
Figure 8: Graph par des exemples de données	14
Figure 9: Capture d'écran de l'application réalisée.....	15
Figure 10: : SVM mesuré d'accélération avec et sans événement de chute.....	19
Figure 11: L'algorithme proposé de détection de chute.....	20
Figure 12: Account SID et Auth Token.....	21

I. Etude d'un accéléromètre :

1. Introduction :

a. Définition :

Un accéléromètre est un capteur qui mesure l'accélération non gravitationnelle linéaire de l'appareil sur lequel il est fixé. L'accélération peut être statique, comme la gravité ou dynamique, comme un choc contre un autre objet. Tout comme la gravité, l'accélération s'exprime en g : 1 g = 9,81 m/s². Contrairement à un gyroscope, il ne peut pas mesurer une vitesse rotative. Pour mesurer l'accélération en 3 dimensions, on utilise un ensemble de 3 accéléromètres formant un accéléromètre triaxial.

Il existe de nombreux types d'accéléromètres, répondant à des besoins différents. Il ne s'agit pas d'une invention récente puisque c'est le mathématicien George Atwood qui inventa le premier accéléromètre au XVIII^e siècle. C'est avec l'essor de l'industrie automobile que les accéléromètres ont commencé à être vraiment utilisés pour rendre les véhicules plus efficaces et plus sûrs. Au fil des années, la technologie a évolué pour aboutir à l'utilisation d'accéléromètres piézoélectriques et d'accéléromètres Mems, les accéléromètres les plus répandus aujourd'hui.

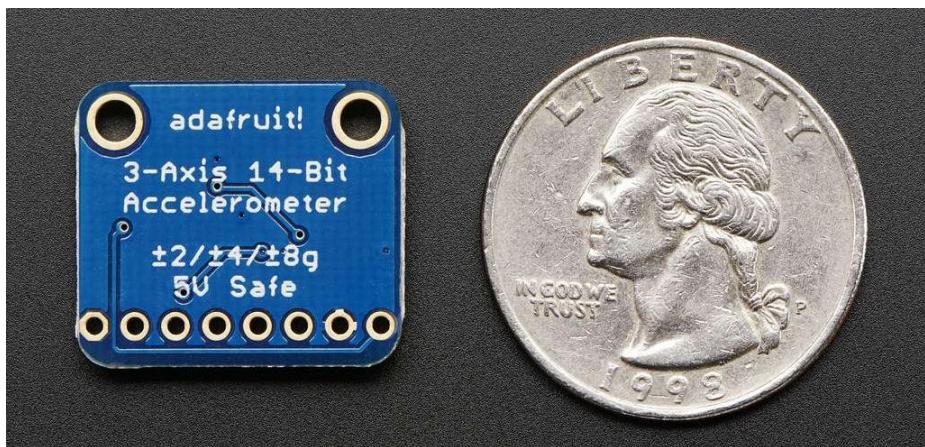


Figure 1: l'accéléromètre

b. À quoi sert un accéléromètre :

Un accéléromètre est utilisé dès lors que l'on souhaite effectuer et utiliser des mesures d'accélération sur un appareil. Mesurer l'accélération permet notamment de calculer la vitesse, le déplacement, les vibrations et les chocs de l'appareil, quelle que soit sa taille. On trouve des accéléromètres dans :

- Les smartphones.
- Les coussins de sécurité dans les voitures.
- Les manettes de consoles de jeux vidéo.
- Les montres connectées.
- Les appareils photo.
- Les ordinateurs portables.
- Divers véhicules et machines.
-

c. Comment fonctionne un accéléromètre :

Un accéléromètre utilise le principe fondamental de la dynamique, appelé aussi seconde loi de Newton. Il stipule que la somme des forces extérieures appliquées à un solide est égale à la dérivée de la quantité de mouvement par rapport au temps. Autrement dit, la somme des forces extérieures est égale à la masse multipliée par l'accélération exprimée en m/s^2 . Un accéléromètre est formé d'une masse attachée à une surface fixe grâce à un ressort. Lorsque la masse se déplace, on en déduit l'accélération en fonction du coefficient du ressort et de la distance parcourue par la masse. Il existe plusieurs méthodes pour mesurer cette distance. Elles peuvent être résistives, capacitifs ou inductives.

2. Les applications de l'accéléromètre :

Les applications de ce capteur sont très diverses :

- La mesure de vitesse (par intégration).
- La mesure de déplacement (par double intégration).
- Le diagnostic de machine (par analyse vibratoire).
- La détection de défaut dans les matériaux (en mesurant la propagation d'une vibration à travers les matériaux).
- La détection des chutes (**notre application**).

Elles sont généralement classées en trois grandes catégories :

a. Les chocs :

Les chocs sont des accélérations de très forte amplitude. Par exemple, un accéléromètre qui tombe d'une hauteur de 20 cm sur une tôle d'acier de 5 cm d'épaisseur sera soumis à une accélération de 8000 g lors de l'impact, et sur un cahier de 50 pages d'épaisseur, il sera soumis à une accélération de 90 g. Ceux sont des accélérations très rapides et donc qui nécessitent un capteur de bande de passante allant généralement de 0 à 100 kHz. La précision requise pour ces mesures est de l'ordre de 1% de l'échelle de mesure du capteur.

Exemples :

- Déclenchement des coussins de sécurité dans les voitures.

a. L'accélération vibratoire :

Les accélérations vibratoires sont considérées comme des accélérations de niveau moyen (généralement une centaine de g). Elles nécessitent un capteur de bande de passante allant jusqu'à 10kHz et de précision de l'ordre de 1% de l'échelle de mesure du capteur.

Exemples :

- Le contrôle vibratoire pour la R&D.
- Le contrôle industriel.

b. L'accélération de mobile :

Les accélérations de mobiles sont de faible niveau. Par exemple, l'accélération maximum retenue pour le "Rafale" est de 9g. Ces accélérations n'excèdent pas quelques dizaines de Hertz. En revanche, la précision requise peut être importante. Elle varie de 0,01% à 2% de l'échelle de mesure du capteur.

Exemple :

- Les stations inertielles des avions
- L'aide à détermination dynamique de la position d'un train sur une ligne

3. Etude des accéléromètres intégrés dans les smartphones :

a. Fonctionnement de l'accéléromètre du smartphone :

L'accéléromètre d'un smartphone est rendu possible grâce à un type particulier de puces appelées « MEMS », de l'acronyme anglais pour « microsystème électromécanique ».

Ce sont des constructions en semi-conducteurs, très fins et très petits, qui fonctionnent comme des petits ressorts. Chargés électriquement, la déviation du petit ressort produit le déplacement de la charge électrique, qui est alors captée.

L'accéléromètre de votre téléphone est constitué d'une très fine tige de silicium mobile. Cette tige est suffisamment petite pour rester solide et ne pas se briser, mais également assez longue (une centaine de micromètres) pour avoir une inertie propre et pouvoir bouger librement.

De cette façon, quand le téléphone bouge, l'inertie de la tige retarde sa propre mise en mouvement et elle se retrouve décalée d'un côté. Cette déviation ne dépasse pas une distance de l'ordre du dixième de micromètre.

Quand cette tige de silicium est chargée électriquement et la cage dans laquelle elle se trouve également, le déplacement de charges portée sur la tige est détectable et on en déduit le sens du déplacement :

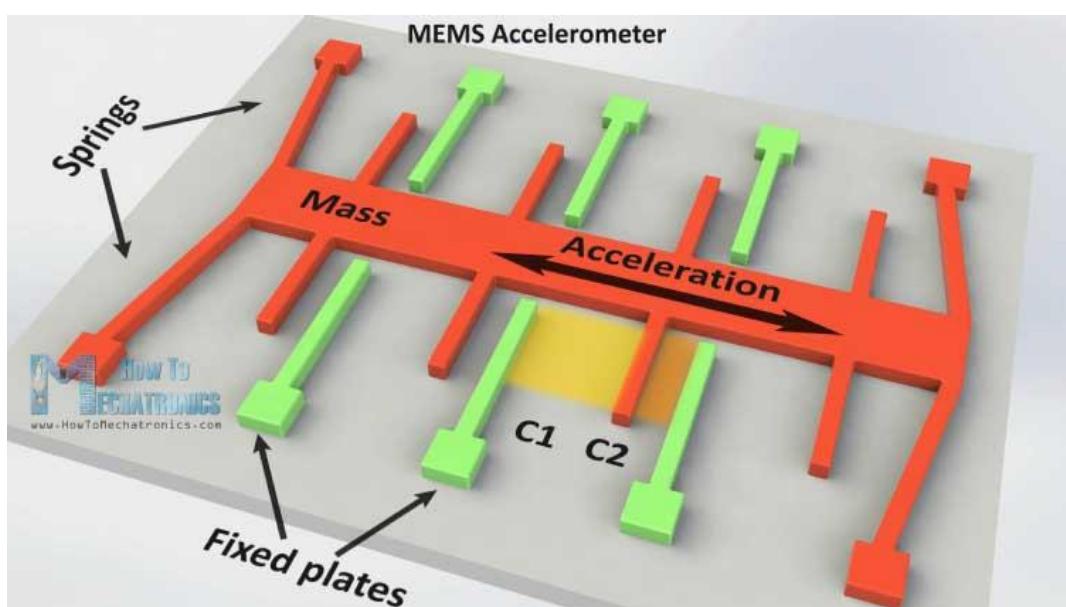


Figure 2 : Principe de fonctionnement d'accéléromètre du smartphone.

Lorsqu'on bouge le téléphone, la partie mobile (en rouge) se déplace sur un plan fixe. La variation de capacité électrique entre les deux tiges fixes est analysée pour reconstituer l'ampleur et le sens du déplacement.

Bien-sûr, avec un seul de ces systèmes, on arrive à mesurer l'accélération sur un seul axe seulement. C'est donc pourquoi que les accéléromètres sont constitués en réalité d'un minimum de 3 de ces tiges, mesurant chacune l'accélération dans un plan (X, Y, Z). C'est ensuite une unité de calcul dédiée qui envoie les informations relatives au déplacement vers le processeur central (CPU).

b. L'utilité des accéléromètres utilisés dans les appareils mobiles :

➤ Détermination des mouvements de nos téléphones :

Le capteur est disponible dans les téléphones mobiles haut de gamme. Si vous possédez un téléphone mobile basé sur clavier, alors vous ne serez pas en mesure de faire l'expérience de ce capteur. Vous devez avoir un smartphone pour voir comment accéléromètre aide à changer l'orientation de votre application mobile.

Les assistants numériques personnels et les lecteurs audiois numériques contrôlent l'interface utilisateur de l'application mobile à l'aide d'un accéléromètre. Il ajuste l'orientation du contenu et la présentation de l'application mobile pour la rendre conviviale. Comme accéléromètre dans smartphone peut suivre le mouvement, il est utilisé comme un podomètre pour compter les étapes, et en fonction de l'étape, il permet aux utilisateurs avec l'analyse détaillée du nombre de calories brûlées, combien de kilomètres ils marchaient, et plus encore. Ainsi, le capteur largement utilisé dans les applications de santé et de remise en forme et les applications sportives.

➤ Détection d'orientation :

La plupart des smartphones utilisent accéléromètre de nos jours pour aligner l'orientation de l'écran en fonction de la direction de l'appareil est tenue. Avec l'accéléromètre intégré de téléphone mobile, les utilisateurs peuvent obtenir une meilleure expérience de visualisation tout en tournant la page, en jouant à des jeux basés sur le geste, en s'ajustant du paysage à l'orientation du portrait, et le zoom-out et zoom-in sur les images.

L'accéléromètre du smartphone comprend au moins un capteur d'inclinaison pour gérer la vue des images. Parfois, il est utilisé dans le but de corriger le mouvement de main tout en prenant des photos, autorotation des images, joué à des mini-jeux sensibles au mouvement.

4. Etat de l'art de l'application :

L'application choisie est la détection des chutes, le thème de la détection des chutes attire de plus en plus l'attention et les contributions scientifiques dans la recherche communauté. La chute est un problème de santé important. Environ 30% des personnes de plus de 65 ans d'âge vivant dans la communauté chutent chaque année. Entre 5 % et 10 % de toutes les chutes entraînent une fracture, et jusqu'à 90 % de toutes les fractures sont causées par une chute.

Les chutes sont également la cause d'autres événements indésirables, comme longtemps réside avec l'incapacité de récupérer. Par conséquent, une détection automatique efficace des chutes, alertées pour obtenir des soins médicaux, est d'une importance primordiale.

Une quantité importante de recherches utilisant la technologie portable pour la détection automatique des chutes a été réalisée ces dernières années. Le sujet porte généralement des

capteurs portables dans une position fixe (par exemple, ceinture) ou position semi-fixe (par exemple, pendentif). Les signaux enregistrés sont généralement des signaux cinématiques (accélérations et les vitesses angulaires).

Pour réduire les conséquences des chutes et améliorer la qualité de vie des personnes âgées. Il y a un dispositif communicant qui s'intègre dans la vie de la personne dans l'objectif d'alerter en cas de chute. À chaque événement anormal, une alarme, contenant l'état de la personne, est envoyée aux soignants. Cette alarme est générée à l'aide d'un système de détection de chute automatique, et on peut utiliser notre smartphone comme le dispositif qui permet de détecter les chutes et envoyé une alerte. Dans le cas d'un évènement critique, une décision sera envoyée aux personnes concernées (infirmier, proches) sous forme d'une alarme.

Nous avons ensuite calculé la norme d'accélération (également appelée amplitude ou vecteur somme dans le littérature scientifique), qui a été obtenue par les signaux d'accélération selon les trois axes de détection par la formule suivante :

$$Acc = \sqrt{Ax^2 + Ay^2 + Az^2}$$

La Figure suivante illustre :

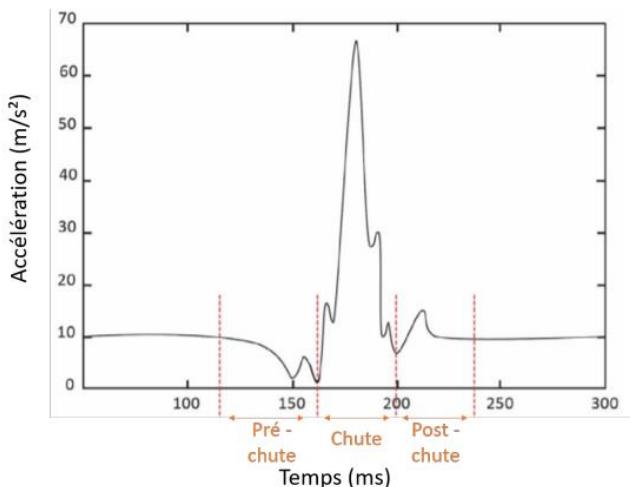


Figure 3: Analyse temporelle d'une chute.

Les différents éléments de détection des chutes sont présentés dans la figure suivante :

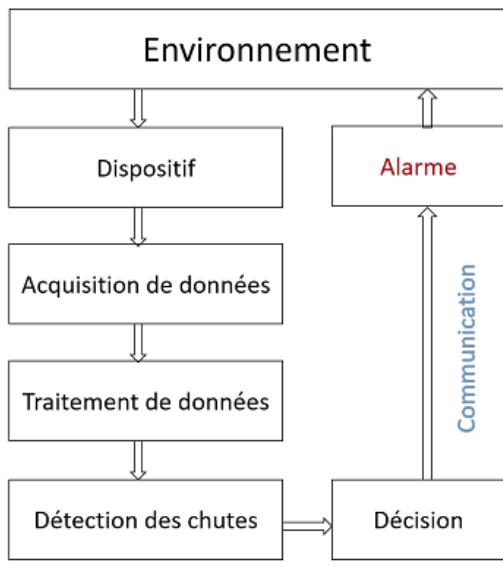


Figure 4: Cadre général d'un système de détection des chutes.

II. Mesure des vibrations :

1. Les étapes pour utiliser l'accéléromètre sous Android :

La plupart des appareils fonctionnant sous Android ont des capteurs intégrés qui mesurent le mouvement, l'orientation et diverses conditions environnementales. Ces capteurs sont capables de fournir des données brutes avec une précision et une exactitude élevées et sont utiles si vous souhaitez surveiller le mouvement ou le positionnement d'un appareil en trois dimensions, ou si vous souhaitez surveiller les changements dans l'environnement ambiant à proximité d'un appareil. Par exemple, un jeu peut suivre les lectures du capteur de gravité d'un appareil pour déduire des gestes et des mouvements complexes de l'utilisateur, tels que l'inclinaison, le tremblement, la rotation ou le balancement. De même, une application météorologique peut utiliser le capteur de température et le capteur d'humidité d'un appareil pour calculer et signaler le point de rosée, ou une application de voyage peut utiliser le capteur de champ géomagnétique et l'accéléromètre pour signaler un relèvement au compas.

La plate-forme Android prend en charge trois grandes catégories de capteurs :

Détecteurs de mouvement

Ces capteurs mesurent les forces d'accélération et les forces de rotation selon trois axes. Cette catégorie comprend les accéléromètres, les capteurs de gravité, les gyroscopes et les capteurs de vecteurs de rotation.

Capteurs environnementaux

Ces capteurs mesurent divers paramètres environnementaux, tels que la température et la pression de l'air ambiant, l'éclairage et l'humidité. Cette catégorie comprend les baromètres, les photomètres et les thermomètres.

Capteurs de position

Ces capteurs mesurent la position physique d'un appareil. Cette catégorie comprend les capteurs d'orientation et les magnétomètres.

Vous pouvez accéder aux capteurs disponibles sur l'appareil et acquérir des données de capteur brutes à l'aide de la structure de capteur Android. L'infrastructure de capteurs fournit plusieurs classes et interfaces qui vous aident à effectuer une grande variété de tâches liées aux capteurs. Par exemple, vous pouvez utiliser la structure de capteur pour effectuer les opérations suivantes :

Déterminez quels capteurs sont disponibles sur un appareil.

Déterminez les capacités d'un capteur individuel, telles que sa portée maximale, son fabricant, ses besoins en alimentation et sa résolution.

Acquérir des données de capteur brutes et définir la vitesse minimale à laquelle vous acquérez des données de capteur.

Enregistrez et réenregistrez les écouteurs d'événements de capteur qui surveillent les changements de capteur.

Dans notre application on est concerné par la mesure de vibrations, donc l'accéléromètre. On accède à

La bibliothèque des capteurs sous Android Studio par la commande SensorManager.

La référence de l'accéléromètre dans cette bibliothèque est TYPE_ACCELEROMETER.

a. Construisez la mise en page de l'application avec textView, barre de progression :

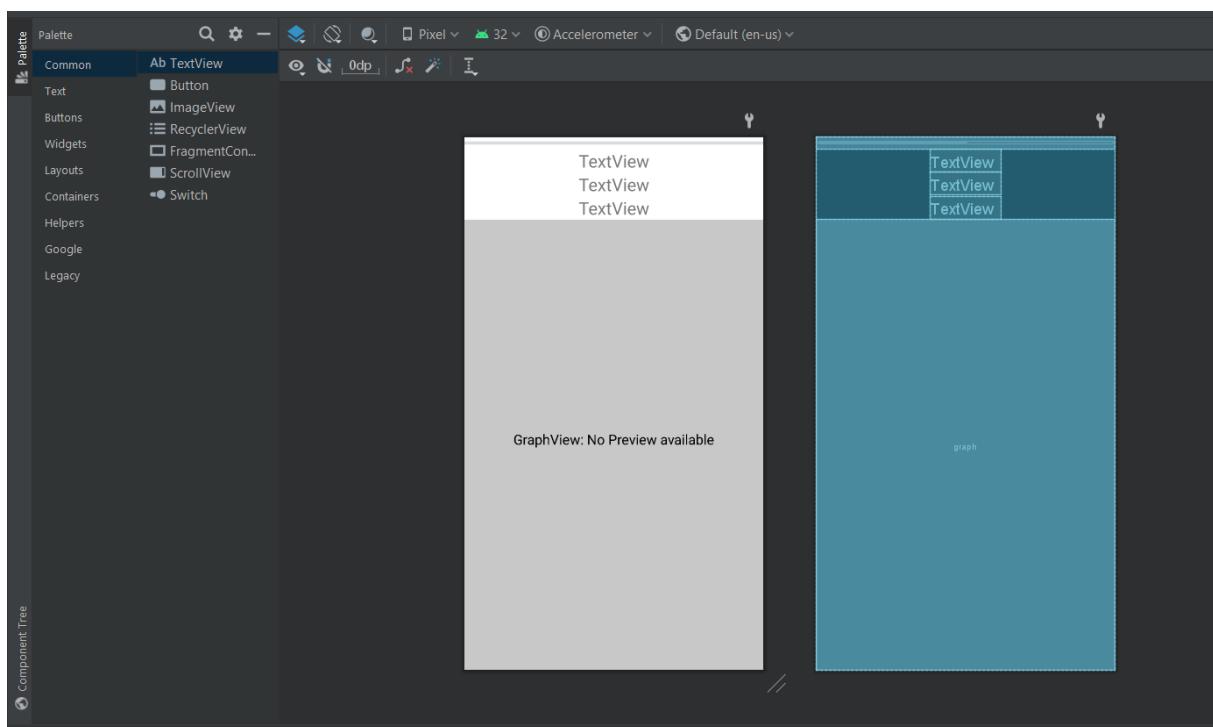


Figure 5: Page de l'application avec textView.

b. Créer des références variables aux éléments de mise en page à l'aide de findViewByld :

Lorsque nous passons au code java, la première étape que nous devons toujours faire lorsque vous créez une nouvelle mise en page et que vous l'attachez au code java est de fournir des références, créons-nous des " textview variables "

```
TextView txt_AccelX, txt_AccelY, txt_AccelZ;  
ProgressBar prog_Accel;
```

```
txt_AccelX = findViewById(R.id.txt_AccelX);  
txt_AccelY = findViewById(R.id.txt_AccelY);  
txt_AccelZ = findViewById(R.id.txt_AccelZ);
```

c. Code de démarrage SensorManager :

Tout d'abord on vérifie que notre appareil (téléphone, tablette) a-t-il des accéléromètres
Nous allons interroger l'ensemble des senseurs de notre device
On commence par création de « sensor manager »

```
private SensorManager mSensorManager;  
private Sensor mAccelerometer;  
  
mSensorManager = (SensorManager) getSystemService(SENSOR_SERVICE);  
mAccelerometer=  
mSensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
```

The sensorManager, nous l'obtenons d'un service intégré à Android, puis Accelerometer va être le capteur par défaut qui recherche donc, le téléphone doit avoir un accéléromètre ici, sinon l'application ne fonctionnera pas

d. Méthodes onResume et onPause :

Un écouteur prévu pour recevoir des informations venant de capteurs est une classe devant implémenter l'interface SensorEventListener. Il sera appelé par le système à chaque fois qu'une nouvelle valeur sera disponible.

```
protected void onResume() {  
    super.onResume();  
    mSensorManager.registerListener(sensorEventListener,  
mAccelerometer, SensorManager.SENSOR_DELAY_NORMAL);  
}  
  
protected void onPause() {  
    super.onPause();  
    mSensorManager.unregisterListener(sensorEventListener);  
}
```

```
}
```

e. nouvelles méthodes *onSensorChanged* *onAccuracyChanged* :

```
public void onSensorChanged(SensorEvent sensorEvent) {  
    // variable declaration  
    float x = sensorEvent.values[0];
```

Cette méthode sera appelée par le système à chaque fois que la précision de la mesure change

```
    float y = sensorEvent.values[1];  
    float z = sensorEvent.values[2];  
public void onAccuracyChanged(Sensor sensor, int i) {  
    }
```

f. Normaliser le mouvement de l'axe z y z en une seule valeur :

Puisque l'accéléromètre est un capteur multidimensionnel, il aura une accélération dans un espace à 3 dimensions ,3 axes X, Y et Z. Donc la valeur reçue sera dans un tableau à 3 dimensions.

```
float x = sensorEvent.values[0];  
float y = sensorEvent.values[1];  
float z = sensorEvent.values[2];
```

g. Mise à jour de l'interface utilisateur avec de nouvelles valeurs de capteur :

```
txt_AccelX.setText("Axe x : " + x);  
txt_AccelY.setText("Axe y : " + y);  
txt_AccelZ.setText("Axe z : " + z);
```

Nous devrions donc voir un changement sur ces trois champs de texte lorsque nous déplaçons le téléphone

h. Exécutez l'accélération sur un téléphone :

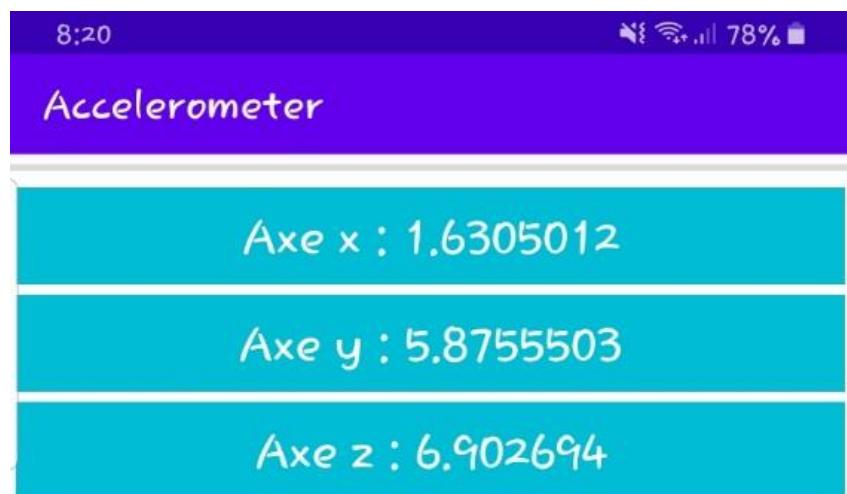


Figure 6: l'accélération sur un téléphone

i. Installez GraphView à l'aide du gestionnaire de dépendances Gradle :

```
GraphView graph = (GraphView) findViewById(R.id.graph);
viewport = graph.getViewport();
viewport.setScrollable(true);
viewport.setXAxisBoundsManual(true);
graph.addSeries(series);
```

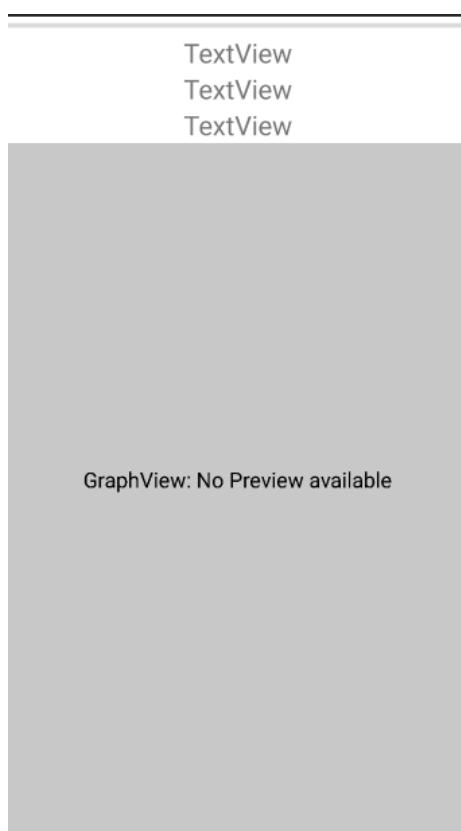


Figure 7: GraphView sur l'application

j. Ajouter des exemples de données pour le graphique :

```
LineGraphSeries<DataPoint> series = new  
LineGraphSeries<DataPoint>(new DataPoint[] {  
    new DataPoint(0, 1),  
    new DataPoint(1, 5),  
    new DataPoint(2, 3),  
    new DataPoint(3, 2),  
    new DataPoint(4, 6)  
});
```



Figure 8: Graph par des exemples de données

Nous avons des points de données qui donneront cinq points de données sur le graphique et au moins nous pourrons voir s'il est chargé correctement même s'il n'affiche aucune donnée utile à notre accéléromètre.

k. Suivre les événements de secousse sur l'historique des graphiques :

Ces deux fonctions vont nous aider à déterminer la fréquence de mise à jour du graphique.

```
// define the number of points plot  
private int pointsPlotted = 5;  
  
// define the variable for th graph view  
private Viewport viewport;
```

Maintenant, si nous déplaçons le téléphone, les petits éléments sur le graphique s'affichent également, il enregistre donc bien les résultats.

```
series.appendData(new DataPoint(pointsPlotted,  
accelerationValue), true, pointsPlotted);  
viewport.setMaxX(pointsPlotted);  
viewport.setMinX(pointsPlotted - 200);
```

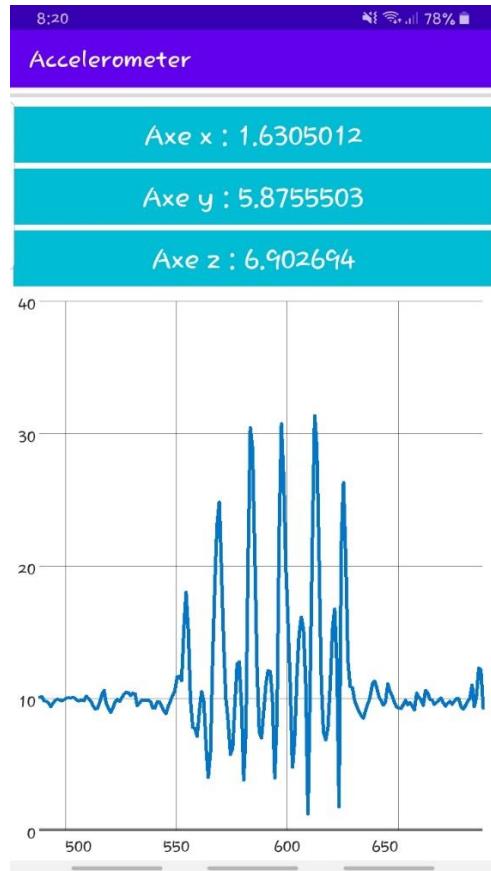


Figure 9: Capture d'écran de l'application réalisée

2. Un programme permettant de mesurer les vibrations du Smartphone :

```

package edu.gcu.shadsluiter.accelerometer;

import androidx.appcompat.app.AppCompatActivity;

import android.graphics.Color;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;
import android.os.Bundle;
import android.widget.ProgressBar;
import android.widget.TextView;

import com.jjoe64.graphview.GraphView;
import com.jjoe64.graphview.Viewport;
import com.jjoe64.graphview.series.DataPoint;
import com.jjoe64.graphview.series.LineGraphSeries;

public class MainActivity extends AppCompatActivity {

```

```
TextView txt_AccelX, txt_AccelY, txt_AccelZ;
ProgressBar prog_Accel;

// define the sensor variables
private SensorManager mSensorManager;
private Sensor mAccelerometer;

// define the variable declaration to calculate the acceleration
value
private double accelerationValue;

// define the number of points plot
private int pointsPlotted = 5;

// define the variable for th graph view
private Viewport viewport;

// initialize graph view
LineGraphSeries<DataPoint> series = new
LineGraphSeries<DataPoint>(new DataPoint[] {
    new DataPoint(0, 1),
    new DataPoint(1, 5),
    new DataPoint(2, 3),
    new DataPoint(3, 2),
    new DataPoint(4, 6)
});

private SensorEventListener sensorEventListener = new
SensorEventListener() {
    @Override
    public void onSensorChanged(SensorEvent sensorEvent) {

        // variable declaration
        float x = sensorEvent.values[0];
        float y = sensorEvent.values[1];
        float z = sensorEvent.values[2];

        // calculate the acceleration value
        accelerationValue = Math.sqrt((x * x + y * y + z * z));

        // update text views
        txt_AccelX.setText("Axe x : " + x);
        txt_AccelY.setText("Axe y : " + y);
        txt_AccelZ.setText("Axe z : " + z);

        // update the graph
        pointsPlotted++;
    }
}
```

```
// reset the data after 1000
if(pointsPlotted > 1000){
    pointsPlotted = 1;
    series.resetData(new DataPoint[] {new DataPoint(1,
0)}));
}

// set the data
series.appendData(new DataPoint(pointsPlotted,
accelerationValue), true, pointsPlotted);
viewport.setMaxX(pointsPlotted);
viewport.setMinX(pointsPlotted - 200);
}

@Override
public void onAccuracyChanged(Sensor sensor, int i) {

}

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    // find the text view by your IDs
    txt_AccelX = findViewById(R.id.txt_AccelX);
    txt_AccelY = findViewById(R.id.txt_AccelY);
    txt_AccelZ = findViewById(R.id.txt_AccelZ);

    // initialize sensor objects
    mSensorManager =
(SensorManager) getSystemService(SENSOR_SERVICE);
    mAccelerometer=
mSensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);

    // sample graph code
    GraphView graph = (GraphView) findViewById(R.id.graph);
    viewport = graph.getViewport();
    viewport.setScrollable(true);
    viewport.setXAxisBoundsManual(true);
    graph.addSeries(series);

}

protected void onResume() {
    super.onResume();
```

```

        mSensorManager.registerListener(sensorEventListener,
mAccelerometer, SensorManager.SENSOR_DELAY_NORMAL);
    }

    protected void onPause() {
        super.onPause();
        mSensorManager.unregisterListener(sensorEventListener);
    }
}

```

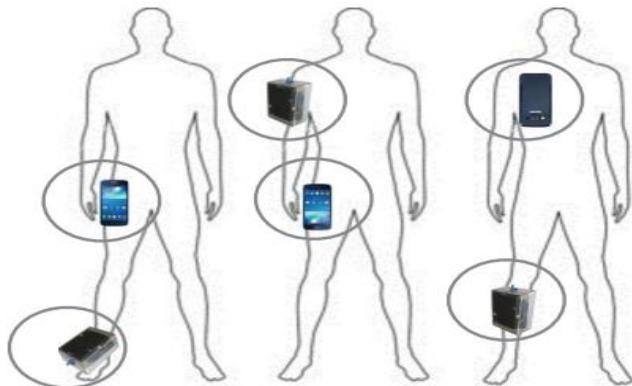
III. Réalisation de l'application :

1. Exploitation des vibrations pour l'application de détection des chutes :

a. Système complet :

Smartphone et dispositif ont été placés dans différentes positions pour identifier la meilleure solution pour la détection de chute.

- Les accéléromètres mesurent les 3 composants de l'accélération : Ax, Ay, Az.
- Si la personne ne se déplace pas toute l'accélération est l'accélération de pesanteur (autour de $9,81 \text{ m/s}^2$).



b. Analyse de la méthode proposée :

Pour s'assurer que le système peut évaluer correctement les chutes, l'amplitude du vecteur de signal (SVM), exprimée comme suit, est utilisée comme base pour évaluer la valeur de seuil.

$$SVM = \sqrt{Ax^2 + Ay^2 + Az^2}$$

Où X, Y et Z sont trois variables d'accélération des axes x, y et z, respectivement. En état normal, la SVM d'accélération est inférieure à 15 m/s^2 (autour de **9,81 m/s²**), alors qu'en cas de chute, la SVM d'accélération calculée est supérieure à seuil donnée qu'on peut noter par Se1. Ou bien dans le deuxième cas c'est que la SVM d'accélération calculée est inférieure à seuil donnée qu'on peut noter par Se2.

Donc en utilisant le trois axes intégré accéléromètre de smartphone, la fonction de détection des chutes peut être réalisée facilement en réglant les deux valeurs seuils (Se1 et Se2).

La figure suivante représente la mesure SVM d'accélération avec et sans événement de chute. D'après cette figure, on peut juger que l'événement de chute se produit à une SVM de 28 m/s^2 .

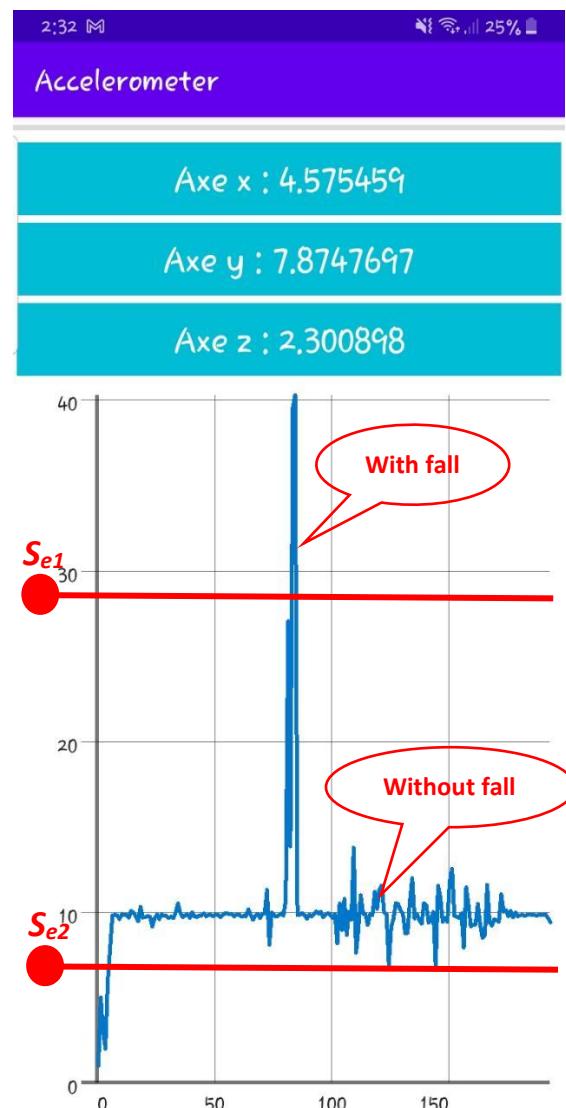


Figure 10: : SVM mesuré d'accélération avec et sans événement de chute.

c. Algorithme proposé de détection de chute :

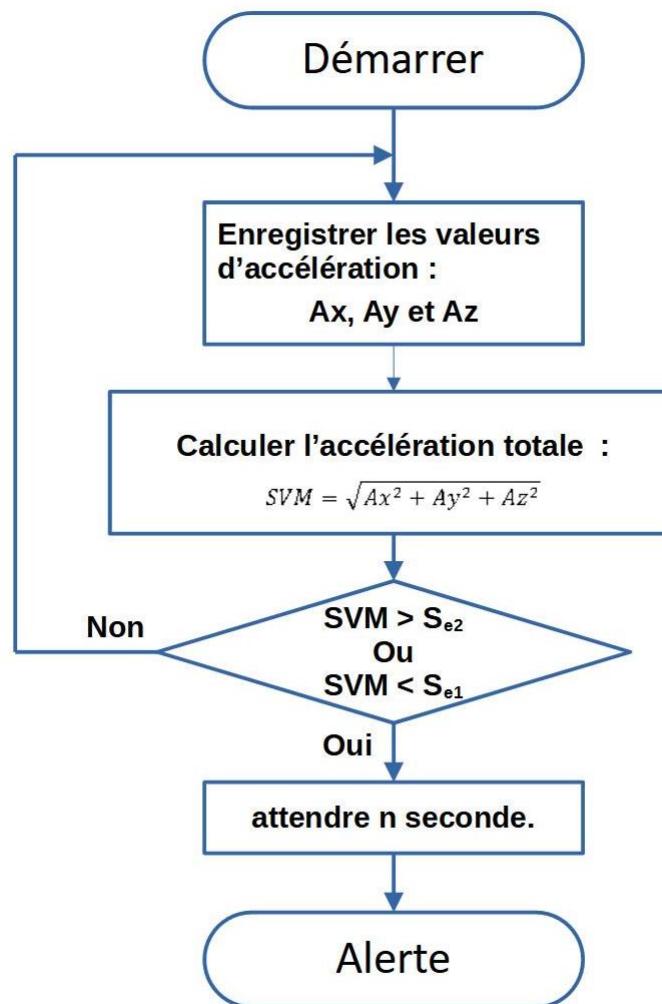


Figure 11: L'algorithme proposé de détection de chute

2. Programme de la réalisation de l'application :

D'après l'algorithme qu'on a proposé et se basons sur le programme précédent, on va ajouter un test pour avoir est ce qu'on a dépassé la valeur des deux seuils, si on a dépassé les seuils on va envoyer une alerte sous forme d'un sms.

Dans notre application on va utiliser la librairie **twilio** pour envoyer le message, tout d'abord on a besoin de crée un compte car on besoin de **Account SID** et **Auth Token**.

Après la création du compte les deux paramètres sont donnés comme suite :

Account SID

AC5c4e9abc89e04f4ea09cf9dd5735a53b



Auth Token

4fdf765e344b51a641596f18733bb308



Hide

Figure 12: Account SID et Auth Token

a. Le programme d'application :

```
package edu.gcu.shadsluiter.accelerometer;

import androidx.appcompat.app.AppCompatActivity;

import android.graphics.Color;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;
import android.os.Bundle;
import android.widget.ProgressBar;
import android.widget.TextView;

import com.jjoe64.graphview.GraphView;
import com.jjoe64.graphview.Viewport;
import com.jjoe64.graphview.series.DataPoint;
import com.jjoe64.graphview.series.LineGraphSeries;

import com.twilio.Twilio;
import com.twilio.rest.api.v2010.account.Message;
import com.twilio.type.PhoneNumber;

public class MainActivity extends AppCompatActivity {

    TextView txt_AccelX, txt_Accely, txt_AccelZ;
    ProgressBar prog_Accel;

    // define the sensor variables
    private SensorManager mSensorManager;
    private Sensor mAccelerometer;

    // define the variable declaration to calculate the acceleration value
    private double accelerationValue;

    // define the number of points plot
    private int pointsPlotted = 5;

    // define the variable for th graph view
    private Viewport viewport;
```

```

// variables Account SID and Auth Token
private static final String ACCOUNT_SID =
"AC5c4e9abc89e04f4ea09cf9dd5735a53b";
private static final String AUTH_TOKEN =
"4fdf765e344b51a641596f18733bb308";

// define the variables of thresholds ;
private float threshold_1 ; // you must enter the numeric value
threshold_1
private float threshold_2 ; // you must enter the numeric value
threshold_2

// initialize graph view
LineGraphSeries<DataPoint> series = new LineGraphSeries<DataPoint>(new
DataPoint[] {
    new DataPoint(0, 1),
    new DataPoint(1, 5),
    new DataPoint(2, 3),
    new DataPoint(3, 2),
    new DataPoint(4, 6)
});

private SensorEventListener sensorEventListener = new SensorEventListener()
{
    @Override
    public void onSensorChanged(SensorEvent sensorEvent) {

        // variable declaration
        float x = sensorEvent.values[0];
        float y = sensorEvent.values[1];
        float z = sensorEvent.values[2];

        // calculate the acceleration value
        accelerationValue = Math.sqrt((x * x + y * y + z * z));

        // update text views
        txt_AccelX.setText("Axe x : " + x);
        txt_AccelY.setText("Axe y : " + y);
        txt_AccelZ.setText("Axe z : " + z);

        // update the graph
        pointsPlotted++;

        // reset the data after 1000
        if(pointsPlotted > 1000){
            pointsPlotted = 1;
            series.resetData(new DataPoint[] {new DataPoint(1, 0)});
        }

        // set the data
        series.appendData(new DataPoint(pointsPlotted, accelerationValue),
true, pointsPlotted);
        viewport.setMaxX(pointsPlotted);
        viewport.setMinX(pointsPlotted - 200);

        // test if we have a fall and send sms
        if(accelerationValue < threshold_1 || accelerationValue >
threshold_1){
            public static void String[] args {
                Twilio.init(ACCOUNT_SID, AUTH_TOKEN);

```

```

        Message message = Message.creator(
            new com.twilio.type.PhoneNumber("MY_Number_Phone"),
            new com.twilio.type.PhoneNumber("+2126XXXXXXX"),
            "Alert for fall detection!!!")
            .create();
    }
}
@Override
public void onAccuracyChanged(Sensor sensor, int i) {
}
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    // find the text view by your IDs
    txt_AccelX = findViewById(R.id.txt_AccelX);
    txt_AccelY = findViewById(R.id.txt_AccelY);
    txt_AccelZ = findViewById(R.id.txt_AccelZ);

    // initialize sensor objects
    mSensorManager = (SensorManager) getSystemService(SENSOR_SERVICE);
    mAccelerometer =
    mSensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);

    // sample graph code
    GraphView graph = (GraphView) findViewById(R.id.graph);
    viewport = graph.getViewport();
    viewport.setScrollable(true);
    viewport.setXAxisBoundsManual(true);
    graph.addSeries(series);

}

protected void onResume() {
    super.onResume();
    mSensorManager.registerListener(sensorEventListener,
mAccelerometer, SensorManager.SENSOR_DELAY_NORMAL);
}

protected void onPause() {
    super.onPause();
    mSensorManager.unregisterListener(sensorEventListener);
}
}

```

IV. Conclusion :

L'objectif de cette activité pratique est de réaliser une application sous le système d'exploitation Android pour exploiter les vibrations mesurées par l'accéléromètre intégré dans un smartphone, et de développer ces vibrations mesurées pour alimenter l'application détection des chutes.

L'élaboration de ce projet nous a aidé à apprendre des connaissances sur le développement d'application, l'usage du logiciel Android Studios et langage JAVA qui est beaucoup utilisé et demandé.