10/01/2020

SYM Labo4

Loris Crüll, Julien Rod, Yoann Rohrbasser

# 1 Capteurs

## 1.2

L’accéléromètre est très sensible. De ce fait, il est constamment en train de recalculer la position du nord par rapport à la rotation de la Terre.

Pour le capteur électro-magnétique, il est à noter que tout autre champ électro-magnétique peut venir perturber son fonctionnement. Pour finir, la précision des capteurs du téléphone est très certainement peu fiable.

Pour remédier à cela, il faudrait faire usage du lissage exponentiel, une technique permettant de filtrer les valeurs récentes trop éloignées des valeurs actuelles afin de garder une certaine cohérence dans l’évolution des données au fil du temps pour que le mouvement de la flèche se fasse de manière fluide.

Source : <https://fr.wikipedia.org/wiki/Lissage_exponentiel>

# 2 Communication Bluetooth Low Energy

## 2.2

*a)* L’intérêt de récupérer la température ainsi est tout simplement d’économiser de la mémoire, qui est déjà très restreinte sur ce type de périphériques. Un entier non-signé de 16 bits est donc forcément plus léger qu’un float de 32 bits et couvre largement l’intervalle de températures pertinentes sur Terre (de −32’768 à 32'767 degrés).

La multiplication par 10 permet de tenir compte des dixièmes de degrés qui peuvent ensuite être récupérés lors de la division par 10 une fois la conversion effectuée lors de l’affichage.

*b)* Pour récupérer le niveau de batterie du périphérique, il faut s’assurer qu’il mette à disposition le service **battery\_service** correspondant au code 0x180F. Pour ce faire, il suffit de lister les services présents :

var gattServices = await bluetoothLeDevice.GetGattServicesAsync();  
foreach (var currService in gattServices)  
 Debug.WriteLine("Service: " + currService.Uuid);

Une fois ceci fait, on regarde si le service propose la caractéristique **battery\_level** correspondant au code 0x2A19 :

var gattCharacs = await currService.GetCharacteristicsAsync();  
foreach (var currCharac in gattCharacs)  
 Debug.WriteLine("Characteristic: " + currCharac.Uuid);

On peut ensuite lire sa valeur qui, dans notre cas, correspond au niveau de batterie en hexadécimal :

if (currCharac.CharacteristicProperties.HasFlag(GattCharacteristicProperties.Read)) {  
 var result = await currCharac.ReadValueAsync();  
 var reader = DataReader.FromBuffer(result.Value);  
 var input = new byte[reader.UnconsumedBufferLength];  
 reader.ReadBytes(input);  
 Debug.WriteLine(BitConverter.ToString(input));  
}

|  |  |  |  |
| --- | --- | --- | --- |
| Caractéristique | Opérations | Données échangées | Format |
| battery\_level | Lecture | Pourcentage de batterie restant | Hexadécimal |

Source : <https://www.andreasjakl.com/read-battery-level-bluetooth-le-devices/>