Rapport de séance :

Séance 4 – 17/01/2020

Pendant cette 4ème séance, la décision a été prise, après un entretien avec M. Masson, de mettre de côté l'avancement sur le module IMU pour se concentrer sur la mise en place du programme de simulation au sein de la carte Arduino. La partie sur le module de mesure de l'inertie étant plus complexe, nous avons préféré commencer dès maintenant la partie plus visuelle, pour rendre une éventuelle présentation à la journée portes ouvertes du 8 février plus attrayante. Il a donc fallu transformer le programme initialement codé en Java en une version utilisable sur l'IDE Arduino. Il semblait plus logique de conserver le fonctionnement orienté objet du programme et j'ai donc pris la décision d'en faire une bibliothèque en C++.

Nous avons donc maintenant une bibliothèque fonctionnelle, utilisable sur l'IDE Arduino, qui prend en argument les données relatives à la direction de l'accélération gravitationnelle et renvoies celles relatives aux déplacements d'un groupe de billes au sein de notre cube. Il s'agira, par la suite, de correctement utiliser ses données pour éclairer nos plaques de LEDs.

Header de la bibliothèque en C++ :

marbles.cpp | marbles.h

```
#ifndef marbles_h
#define marbles_h
#include <Arduino.h>

class Marble{
    public :
        Marble();
        Marble(float x, float y, float s);
        float* getCoords();
        float getSize();
        void update(float gX, float gY);
        void wallCollider();
        void marbleCollider(Marble &marble);
        bool isCollidingWith(Marble &marb);
        static float distance(float* c1, float* c2);
    private :
        float coords[2] = {0};
        float speed[2] = {0};
        float size;
};

class MarbleGroup{
    public :
        MarbleGroup();
        MarbleGroup(int nb, float s);
        Marble *getMarbles();
        void update(float gX, float gY);
    private :
        int nbMarbles;
        bool thereIsColliding();
        Marble marbles[9] = {Marble()};
};

class LedArray{
    public :
        LedArray(int nbLed, float size);
        void update1(float gX, float gY);
    private :
        float coords[2] = {0};
        float speed[2] = {0};
        float size;
        float ledArray[7][7] = {{0}};
        MarbleGroup marbles;
};

#endif
```

Utilisation sur l'IDE Arduino :

ledsTest

```
#include <marbles.h>

LedArray leds(9, 100);

void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:
  leds.update1(0,0);
}
```