الأقسام التحضيرية للمدارس العليا

Classes préparatoires aux grandes écoles

Royaume du Maroc
Ministère de l'Education Nationale
de la Formation Professionnelle
de l'Enseignement Supérieur &
de la Recherche Scientifique

Thème de l'année : santé préventive

Présenté par :

Hassan abdoun

Canne de guidage des malvoyants

2021/2022

Plan

Introduction

Problématique

Malgré le développement de la technologie , on trouve encore des lacunes en certains domaines ; en ce qui concerne les malvoyants, ils affrontent des difficultés en se déplaçant d'un endroit à l'autre ! Et même ils risquent leur vie à l'extérieur, le pire exemple est quand un véhicule l'heurte

Solution proposée

Notre système ; apportera aux personnes aveugles et malvoyantes une aide supplémentaire à la mobilité . Les signaux transmis à l'utilisateur permettent de lui fournir une protection à 100% . Chacun des émetteurs fournissent des données sur le plus proche danger potentiel .

Objectifs

- Eviter les obstacles qui gênent les malvoyants ;
- Guider le malvoyant dans son déplacement ;
- Assurer la sécurité du malvoyant et lui faciliter le déplacement ;
- Connaitre la position de malvoyant

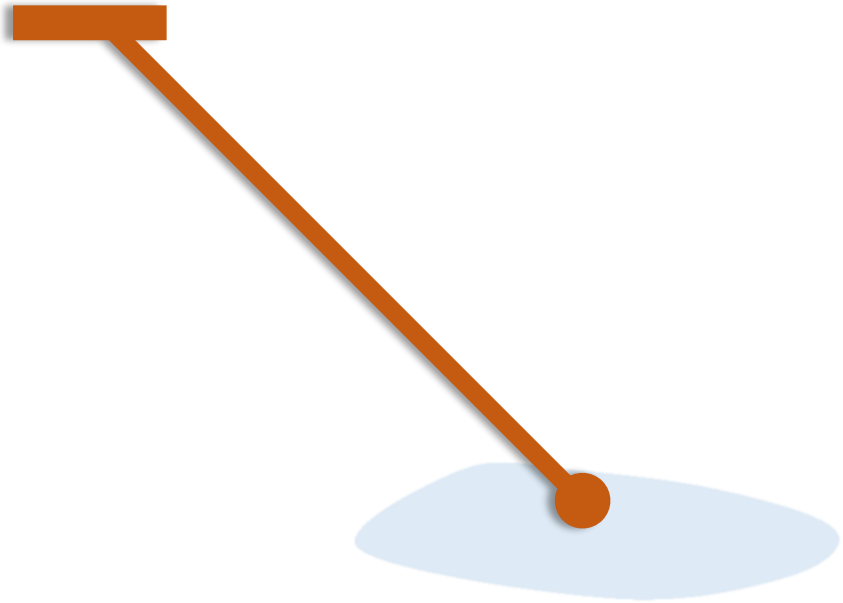


Diagramme de cas d'utilisation

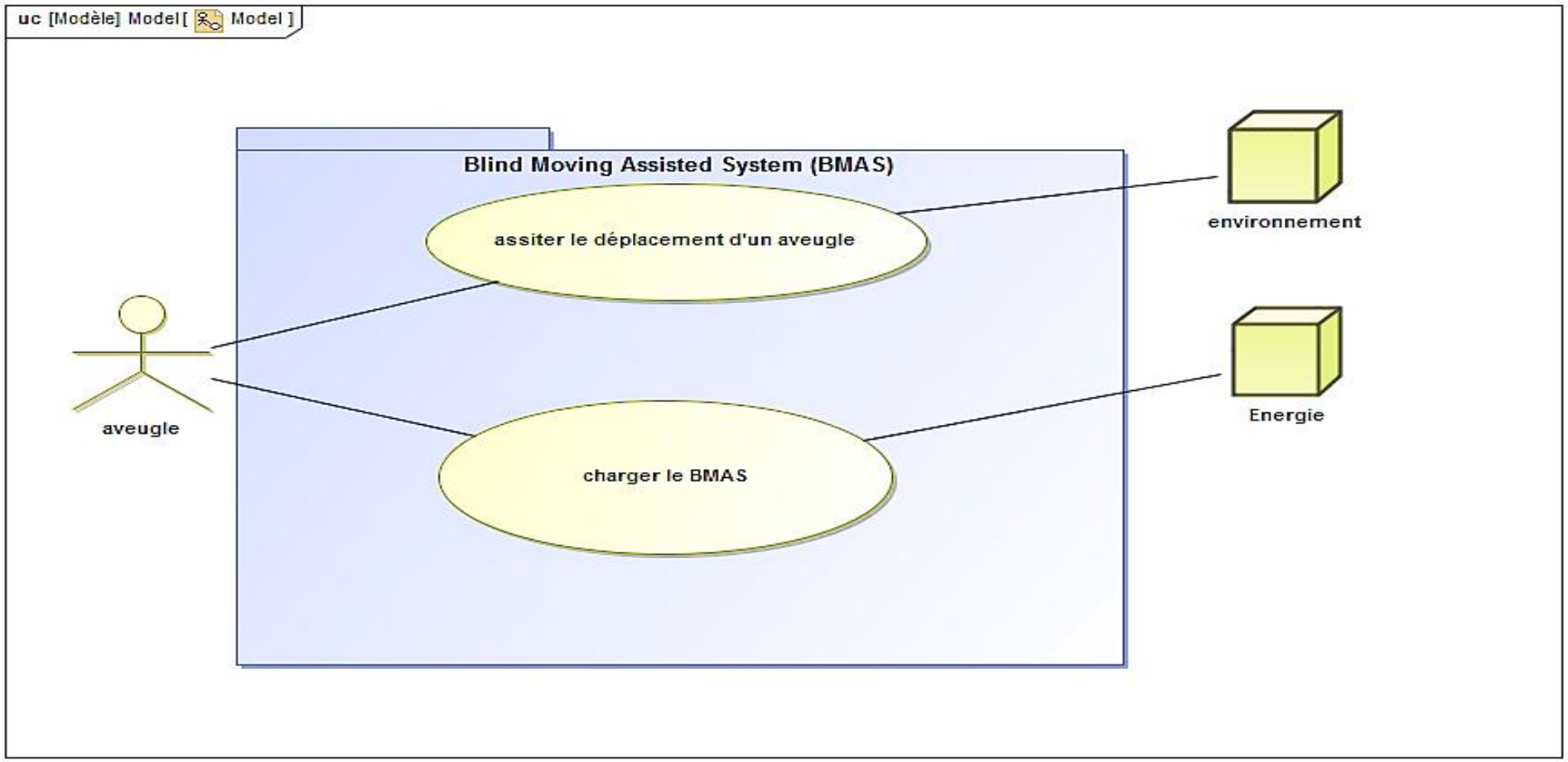


Diagramme d'exigence

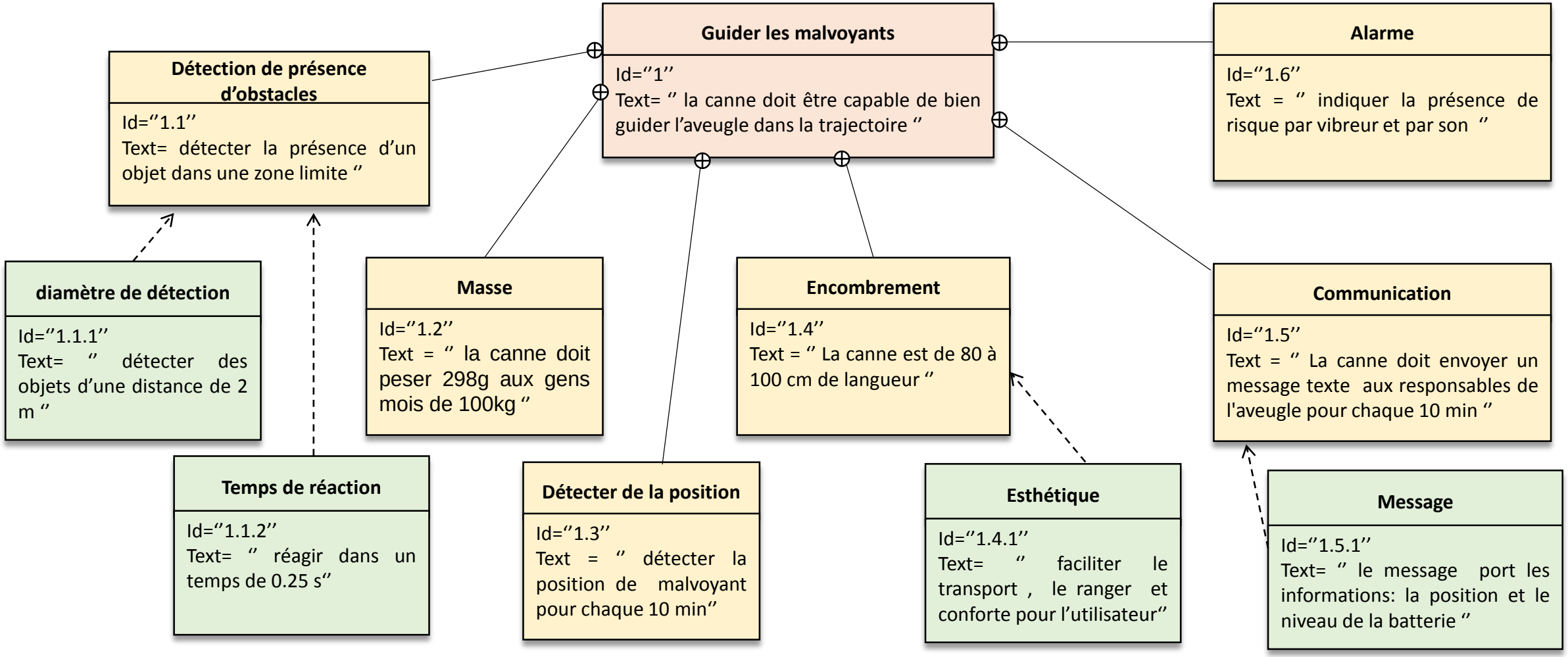
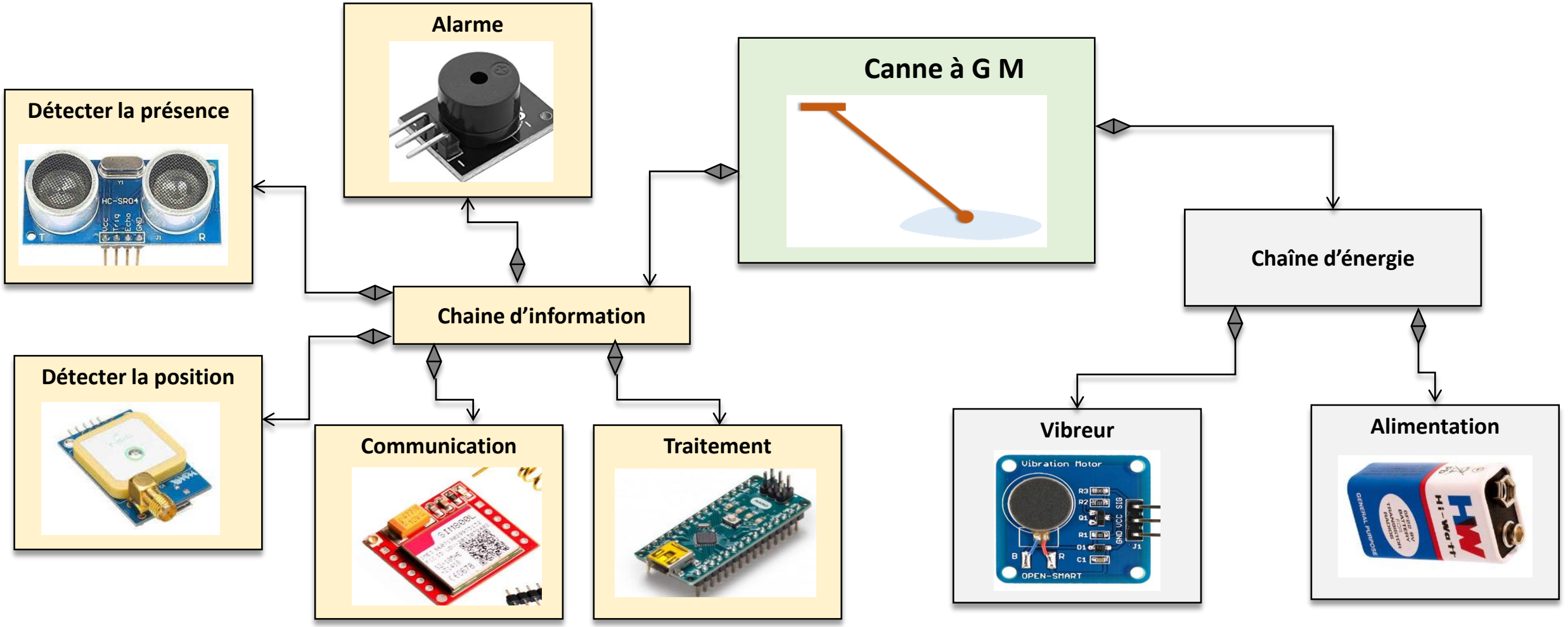


Diagramme de bloc interne



Exigence sur la détection d'obstacle

Exigence 1.1: détecter la présence d'un objet dans une zone limite

- ✓ Détecter des objets d'une distance de 2 m
- ✓ Réagir dans un temps de 0.25 s

Validation le choix de capteur

Choix : Capteur ultrason **SR-04**

- **Caractéristiques de capteur**
  - ✓ Distance maximale : **4 m**
  - ✓ Distance minimale : 2 cm
  - ✓ Tension d'alimentation : 5V
  - ✓ La vitesse de son : 340 m/s

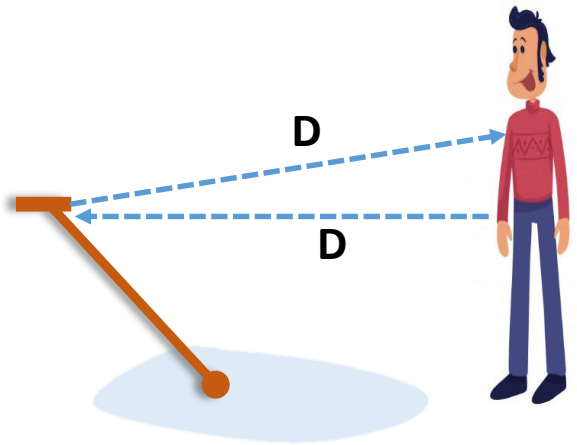
Ce capteur est validé car la distance maximale détectée par le capteur est supérieur à la distance imposée (**2m**).

Exigence sur la détection d'obstacle

Exigence 1.1: détecter la présence d'un objet dans une zone limite

- ✓ Détecter des objets d'une distance de 2 m
- ✓ Réagir dans un temps de 0.25 s

Méthode de calcul de la distance



- **La relation de distance V:**

$$D = \frac{\Delta t}{2} . v$$

Δt: temps de propagation en s

v: vitesse du son dans l'air : 340 m/s

Au niveau de la carte de traitement (ARDUINO)

Mesure le temps entre l'envoi de l'impulsion ultrasonique et son écho Δt

Algorithme de calcul :

```
t = pulseIn(ECHO_PIN, HIGH, MEASURE_TIMEOUT);
D= (t / 2.0) * 340 ;
delay(250)
```

Temps de réagir (0.25s)

Exigence sur la détection de localisation de position

Exigence 1.3: détecter la position de malvoyant pour chaque 10 min

Validation de choix de capteur

La **géolocalisation** est un procédé permettant de positionner un objet, un véhicule, ou une personne sur un plan ou une carte à l'aide de ses coordonnées géographiques. Il y a plusieurs technique sont :

- **Géolocalisation par satellite GPS**
- **Géolocalisation par GSM (précision 100 m)**
- **Géolocalisation par Wi-Fi**
- **Géolocalisation par adresse IP … etc**

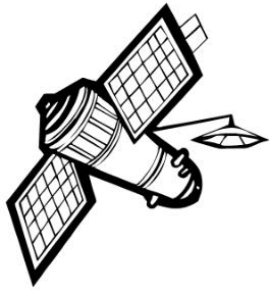
Choix de capteur est basé sur la précision, la meilleur solution technologique actuelle est la **géolocalisation par satellite GPS;**

Analyse des solutions

Exigence sur la détection de localisation de position

Exigence 1.3: détecter la position de malvoyant pour chaque 10 min

Fonctionnement de récepteur GPS



La trame GGA :

$GPGGA,064036.289,4836.5375,N,00740.9373,E,...,0000*0E .

Latitude
48°36'32.25'' Nord

Longitude
7°40'56.238'' Est

Au niveau de la carte de traitement (ARDUINO)

```
#include <TinyGPS.h>;
```
Bibliothèque de GPS

```
float lat = 00.000,lon = 00.000;
```
Déclaration

```
SoftwareSerial gpsSerial(3,4);
```
Connexion

```
gpsSerial.begin(9600);
```
Vitesse de communication

```
gps.f_get_position(&lat,&lon);
```
Extraire à partir du trame , données de localisation

Exigence sur la communication

Exigence 1.3: La canne doit envoyer un message texte aux responsables de l'aveugle pour chaque 10 min

Choix du module de communication

Pour répondre à cet exigence, nous avons travaillé avec le module GSM SIM900 qui permet d'émettre ou de recevoir des appels téléphoniques, des SMS, et des MMS avec des images stockées dans une carte SD ou micro-SD.

Bonjour
Position :
Lat : 00.00 / **Lon** : 00.00
Niveau de batterie :
75%

❑ Caractéristiques

- Control via AT commands
- Supply voltage range : 3.2 ... 4.8V
- Low power consumption: 1.0mA

Analyse des solutions

Exigence sur la communication

Exigence 1.3: La canne doit envoyer un message texte aux responsables de l'aveugle pour chaque 10 min

Fonctionnement de module SIM900

Tension : 4.2 V
Courant : 1.2 A

LM2596

SIM900L

```
#include <GSM.h>
char remoteNum[20];
char txtMsg[150];

sms.beginSMS(remoteNum);
sms.print(txtMsg);
sms.endSMS();
```

Bibliothèque de GSM

Numéro destination

Le message à envoyer

Définir le numéro tel

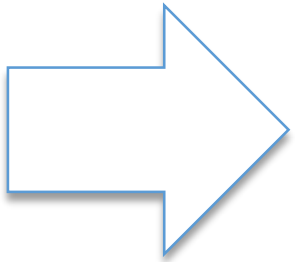
Envoyer le message

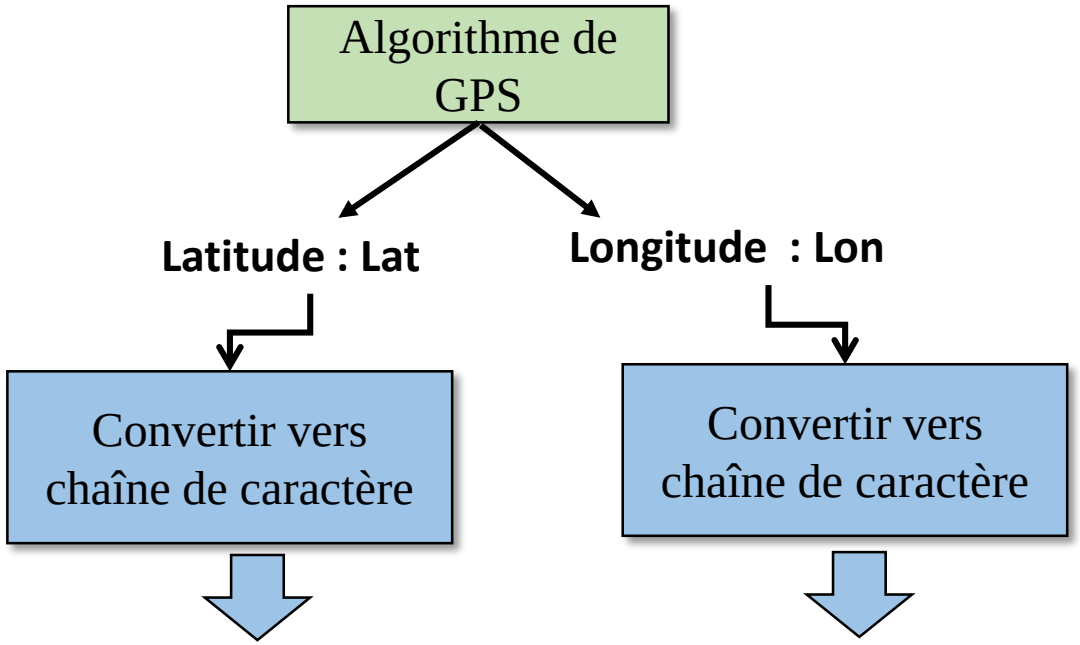
Exigence sur la communication

Exigence 1.3: La canne doit envoyer un message texte aux responsables de l'aveugle pour chaque 10 min

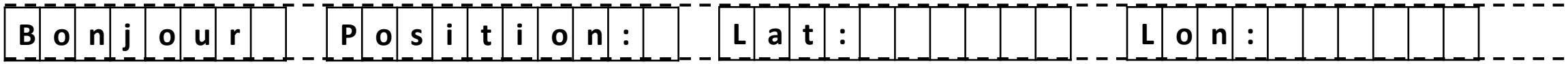
Construction de message

Bonjour
Position :
Lat : 00.00 / **Lon** : 00.00
Niveau de batterie :
75%





Message est composé de 150 lettres (txtMsg[150])



Exigence sur l'alarme

Exigence 1.3: indiquer la présence de risque par vibreur et par son

Validation de choix

Vibreur	Buzzer
	
Mcc 5V	Tension : 5V

L'alarme est déclenché lorsque la distance mesurée est inférieur à 2 m

Organigramme de fonctionnement

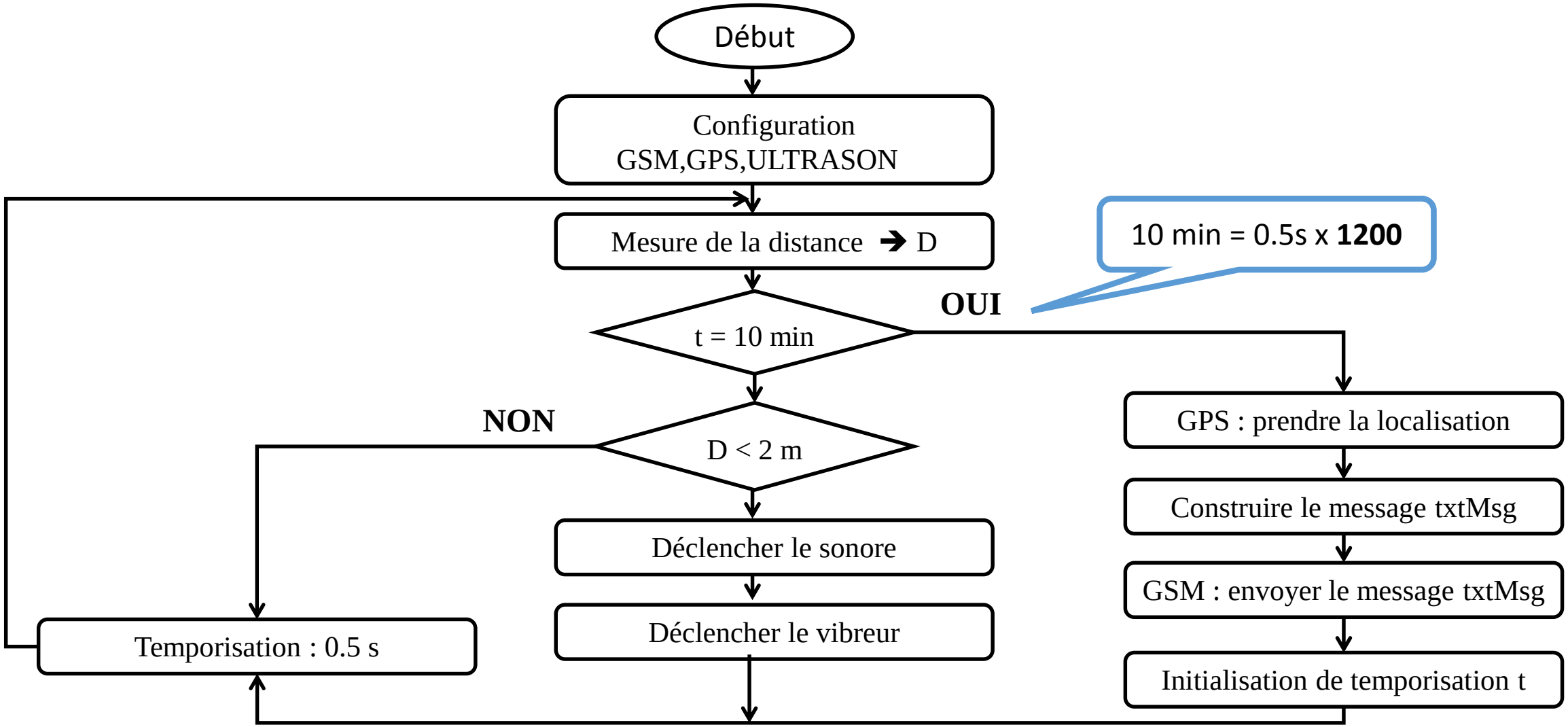
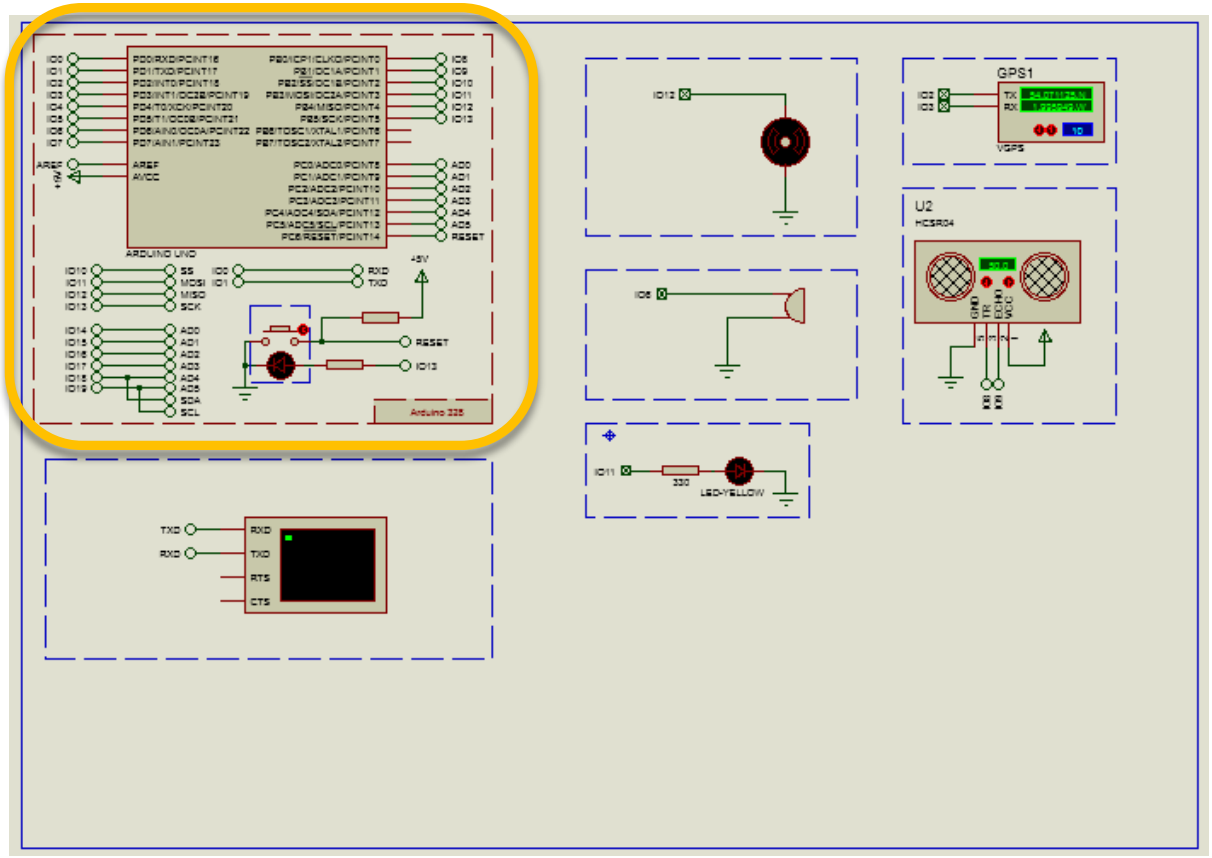
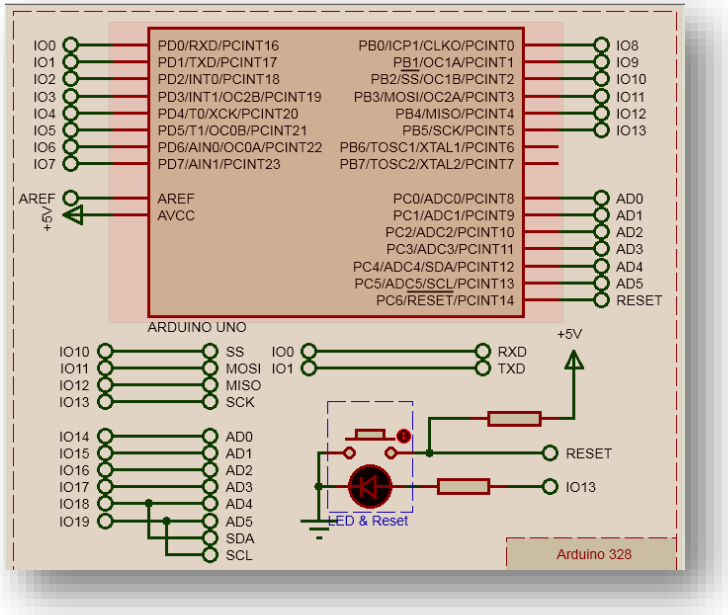


Schéma de simulation

La simulation est réalisé dans **Proteus ISIS version 8.12**





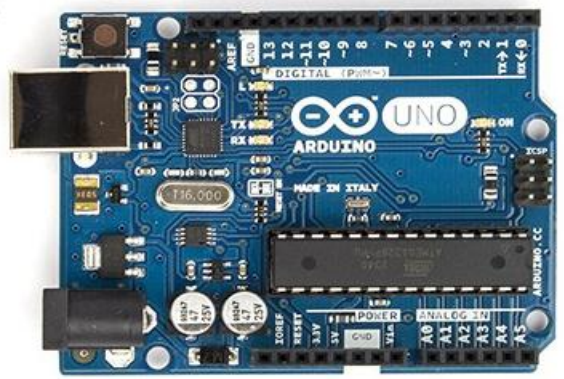
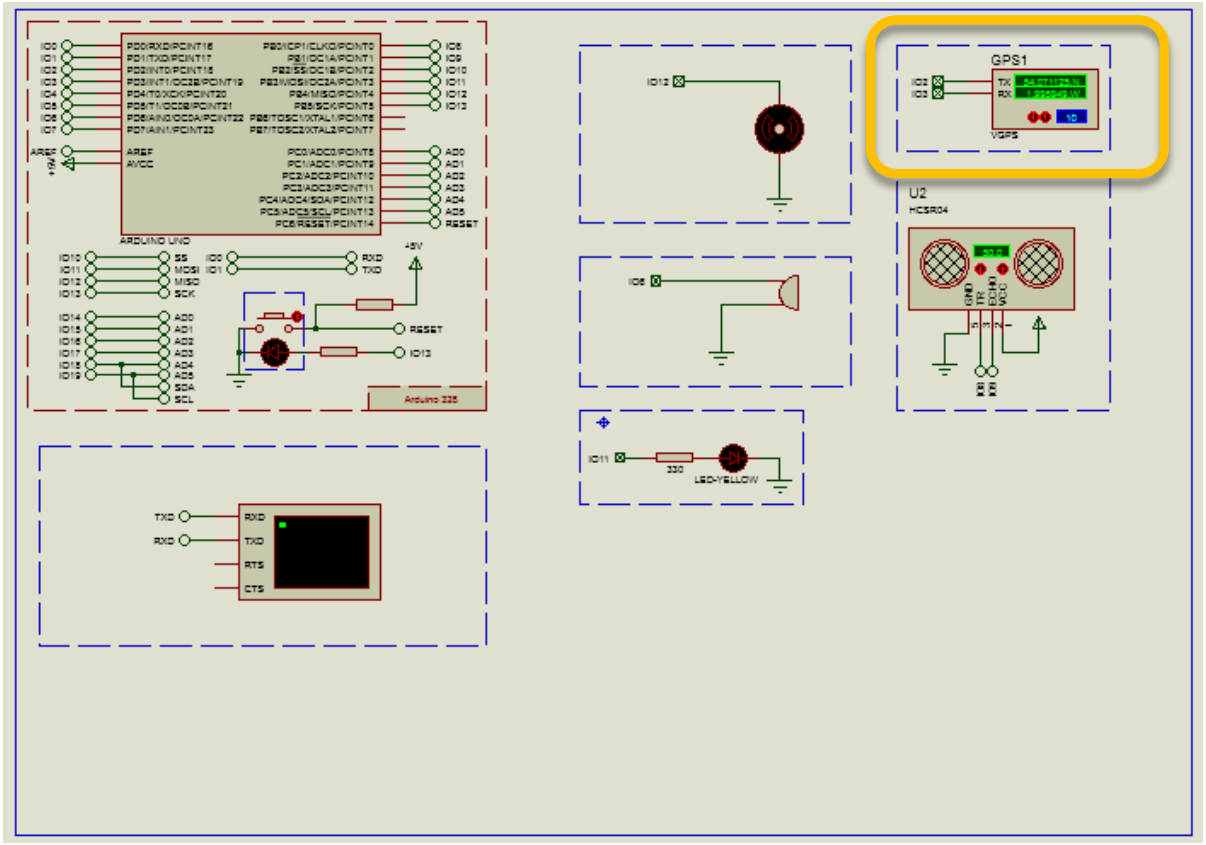


Schéma de simulation

La simulation est réalisé dans **Proteus ISIS version 8.12**



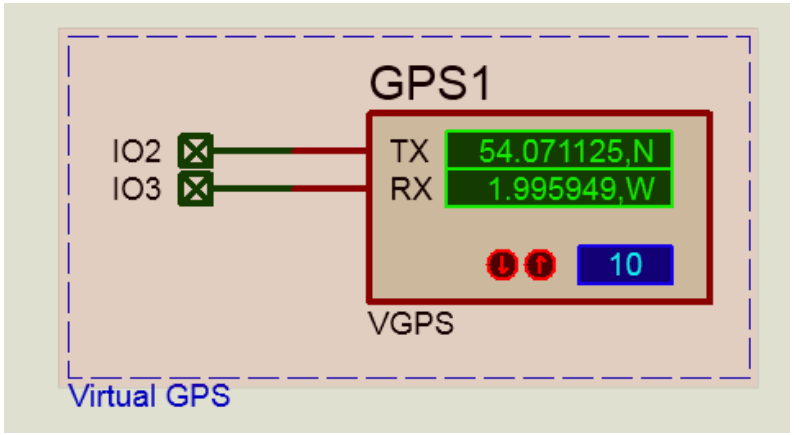
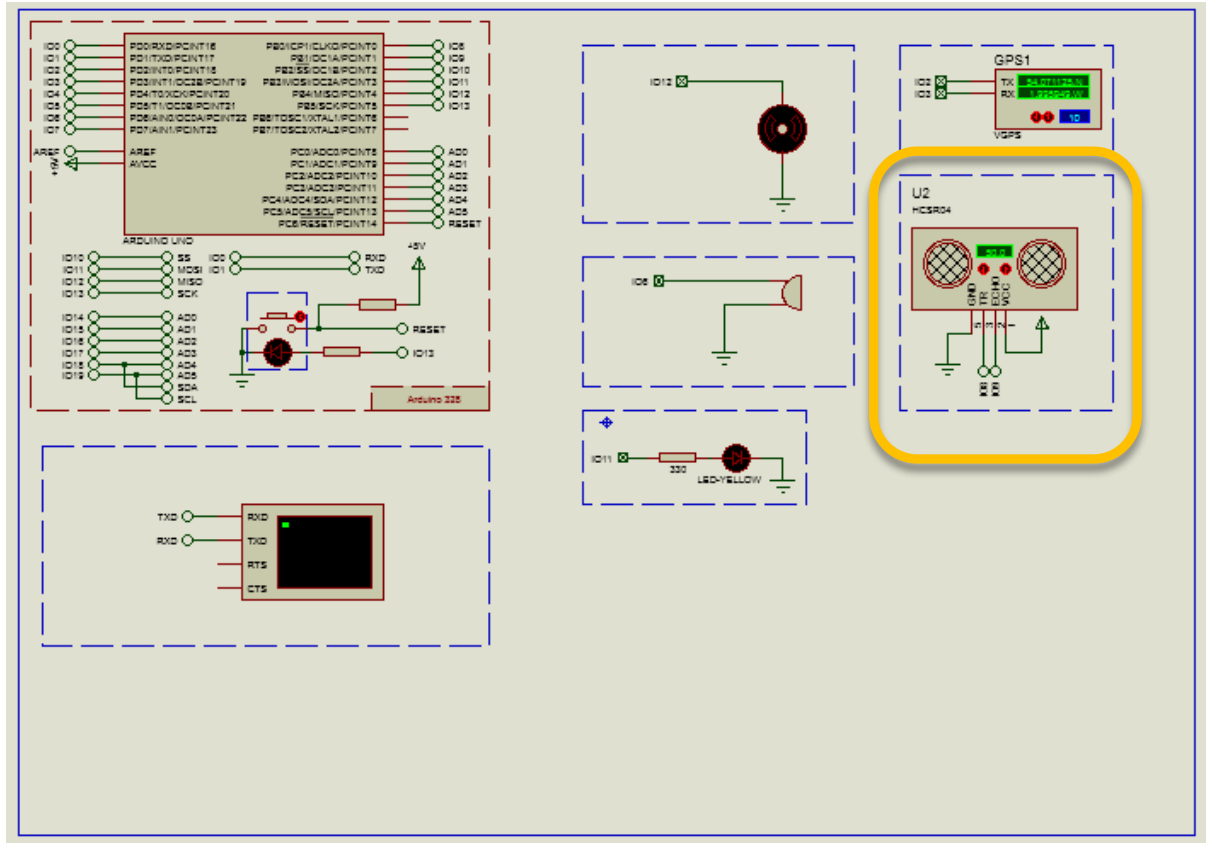


Schéma de simulation

La simulation est réalisé dans **Proteus ISIS version 8.12**



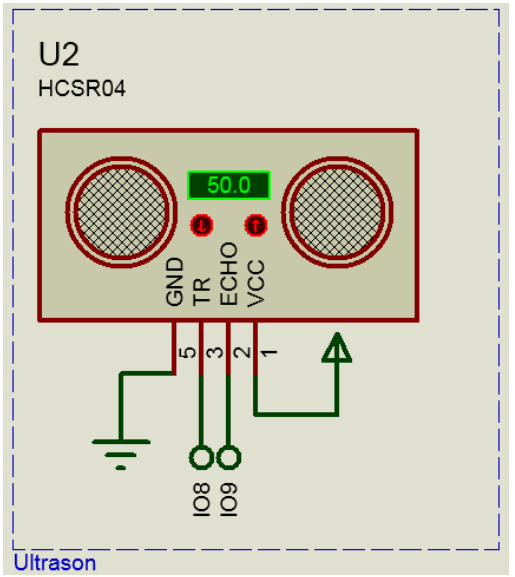
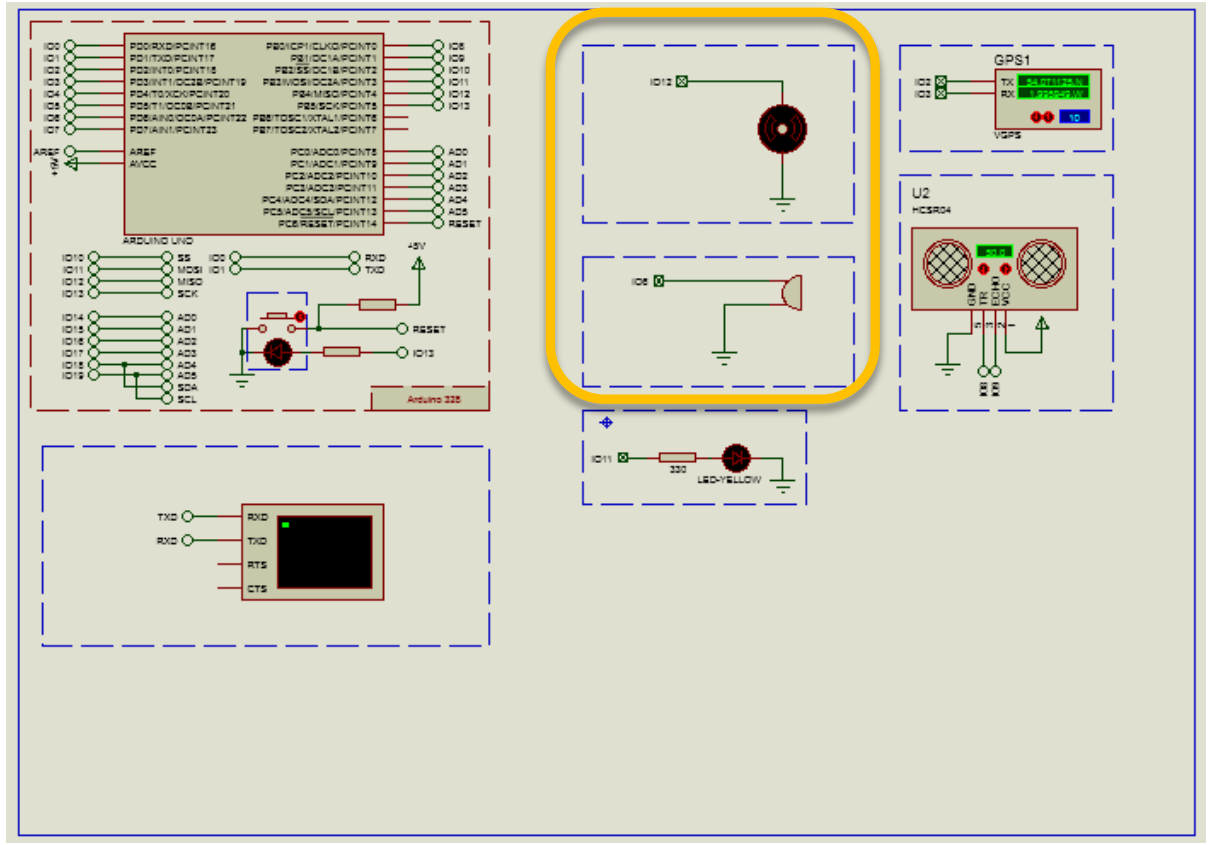
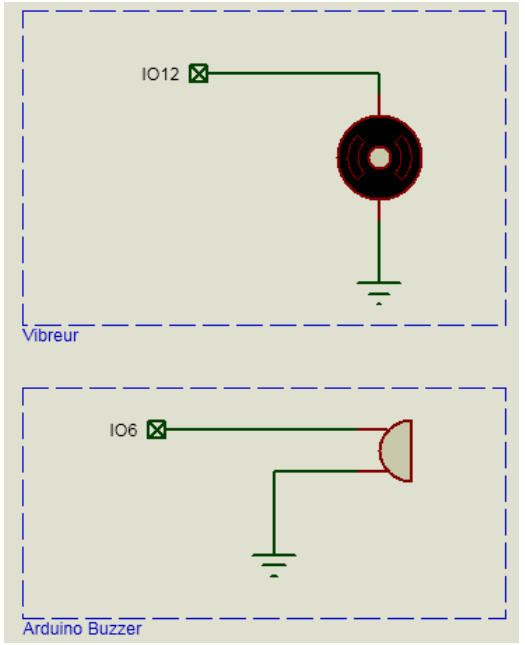


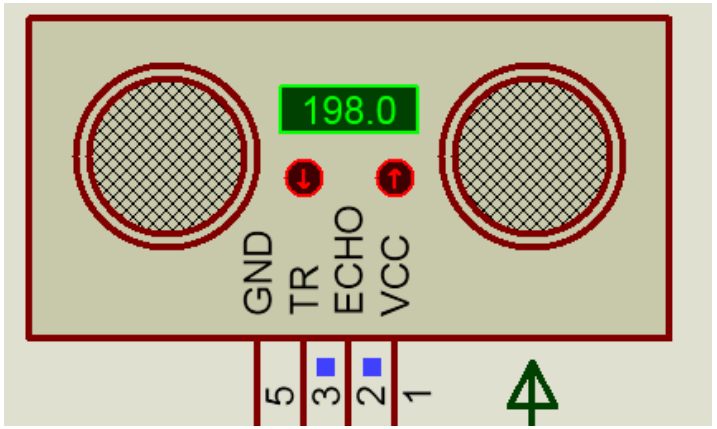
Schéma de simulation

La simulation est réalisé dans **Proteus ISIS version 8.12**

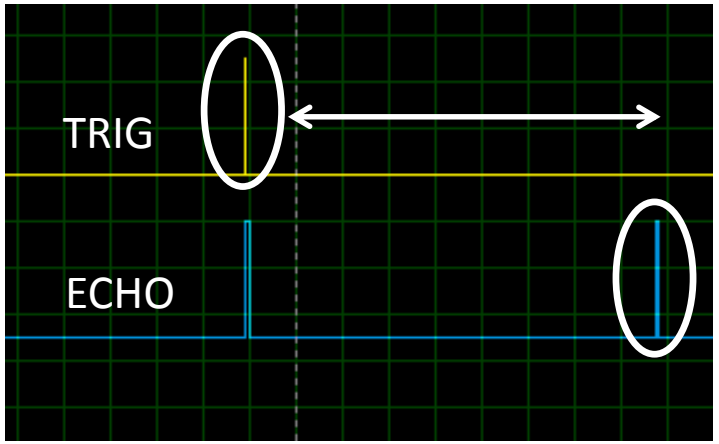




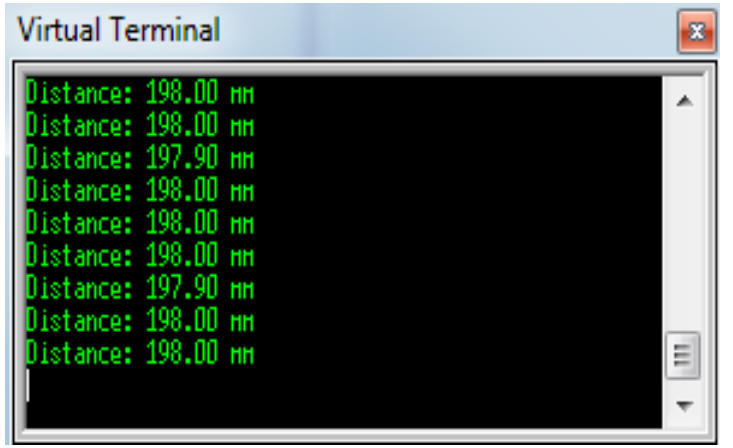
Résultats de simulation : capteur ultrason



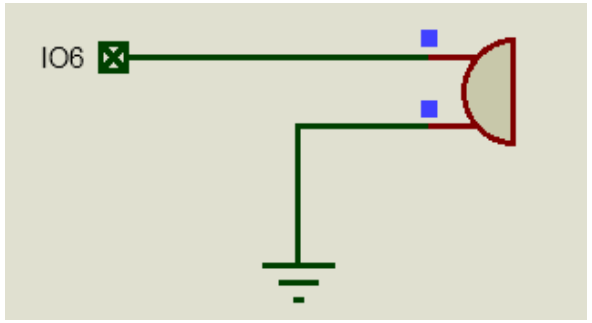
D= 1.98 m < 2m

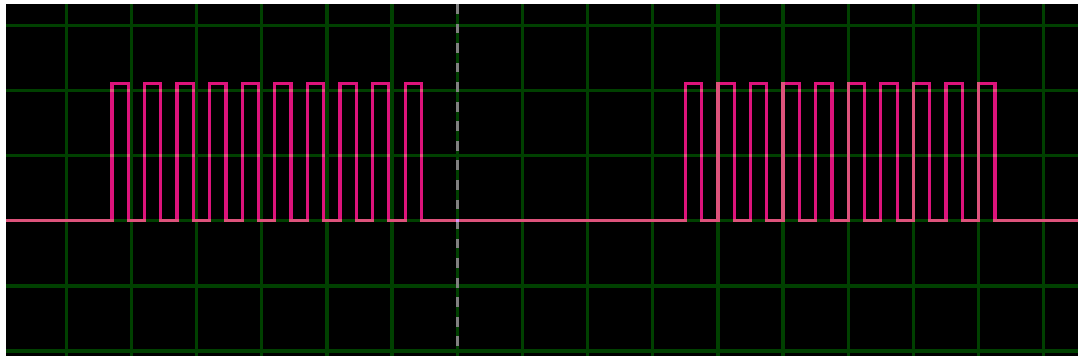


Signal émis et signal écho



La distance mesurée (avec erreur de 1mm)

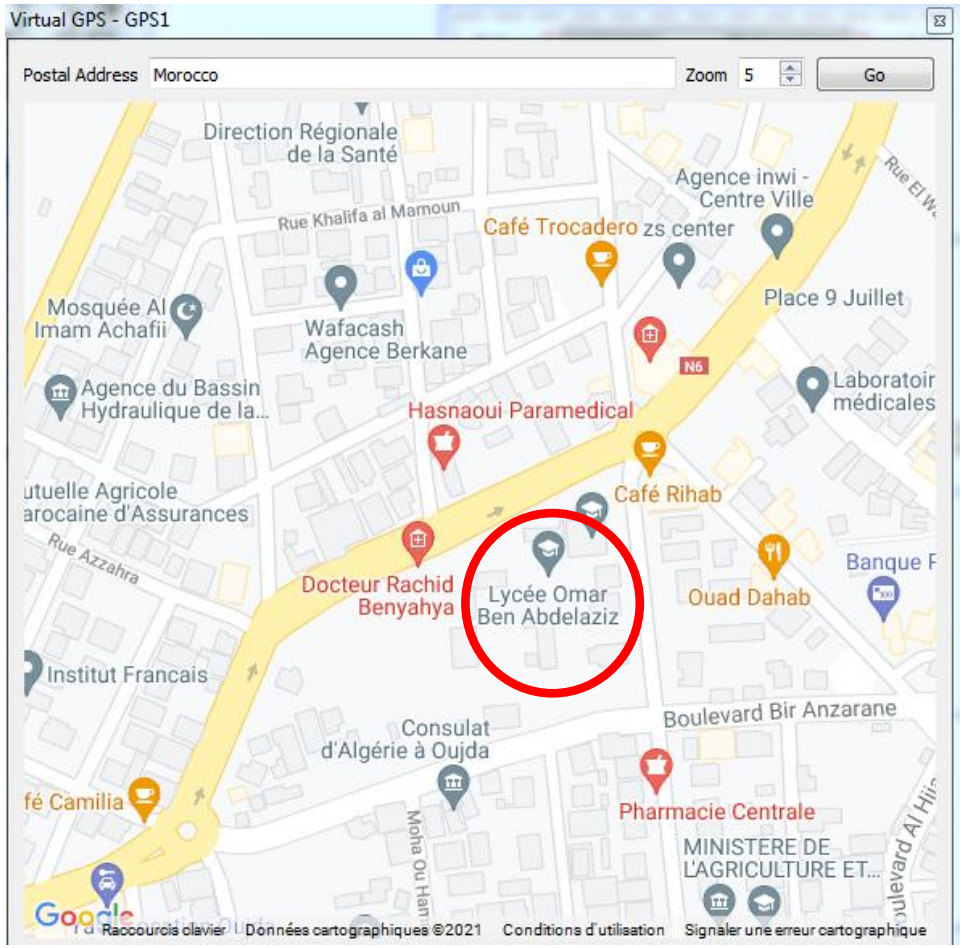




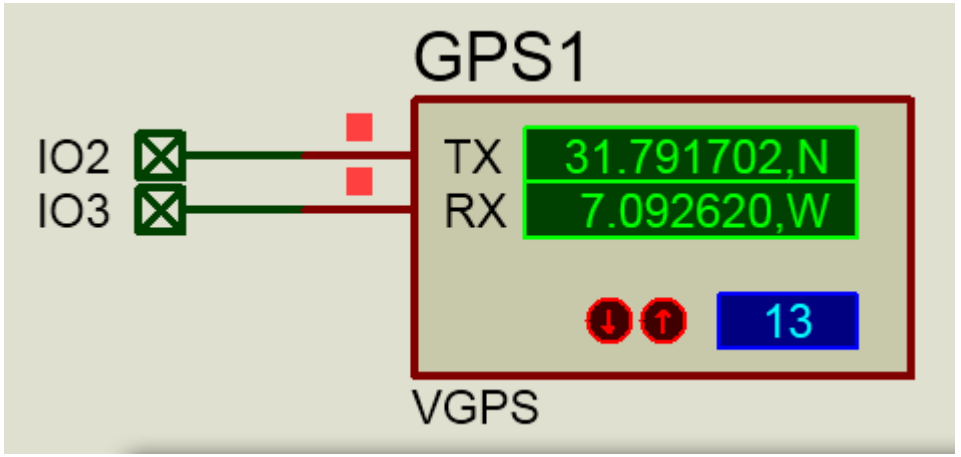
Le signal sonore type « PIP » (la détection d'obstacle)

Simulation et résultats

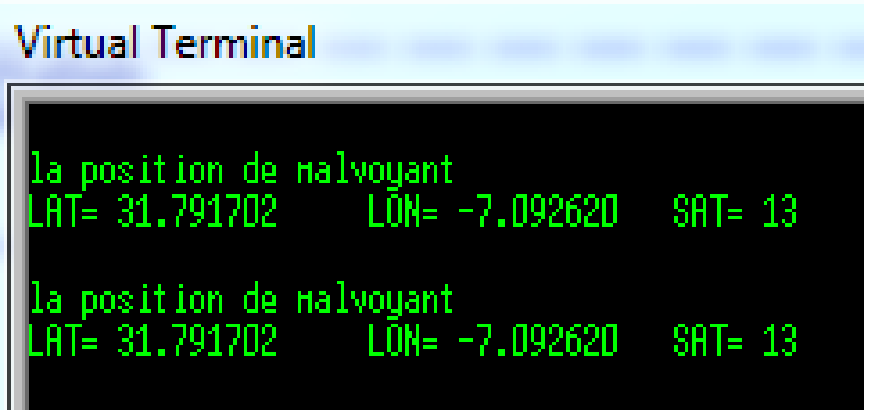
Résultats de simulation : GPS



La carte de OUJDA



GPS virtuel



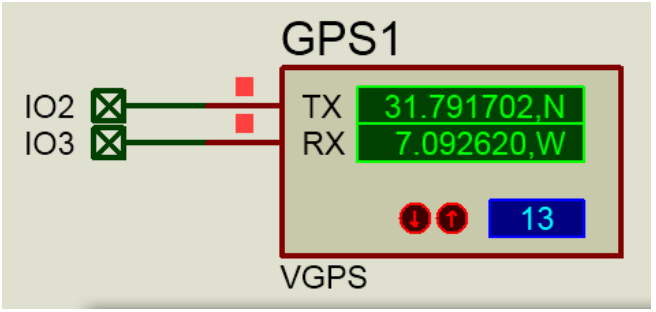
Les information récupérées (latitude , longitude et nombre de satellites)

Résultats de simulation : GPS

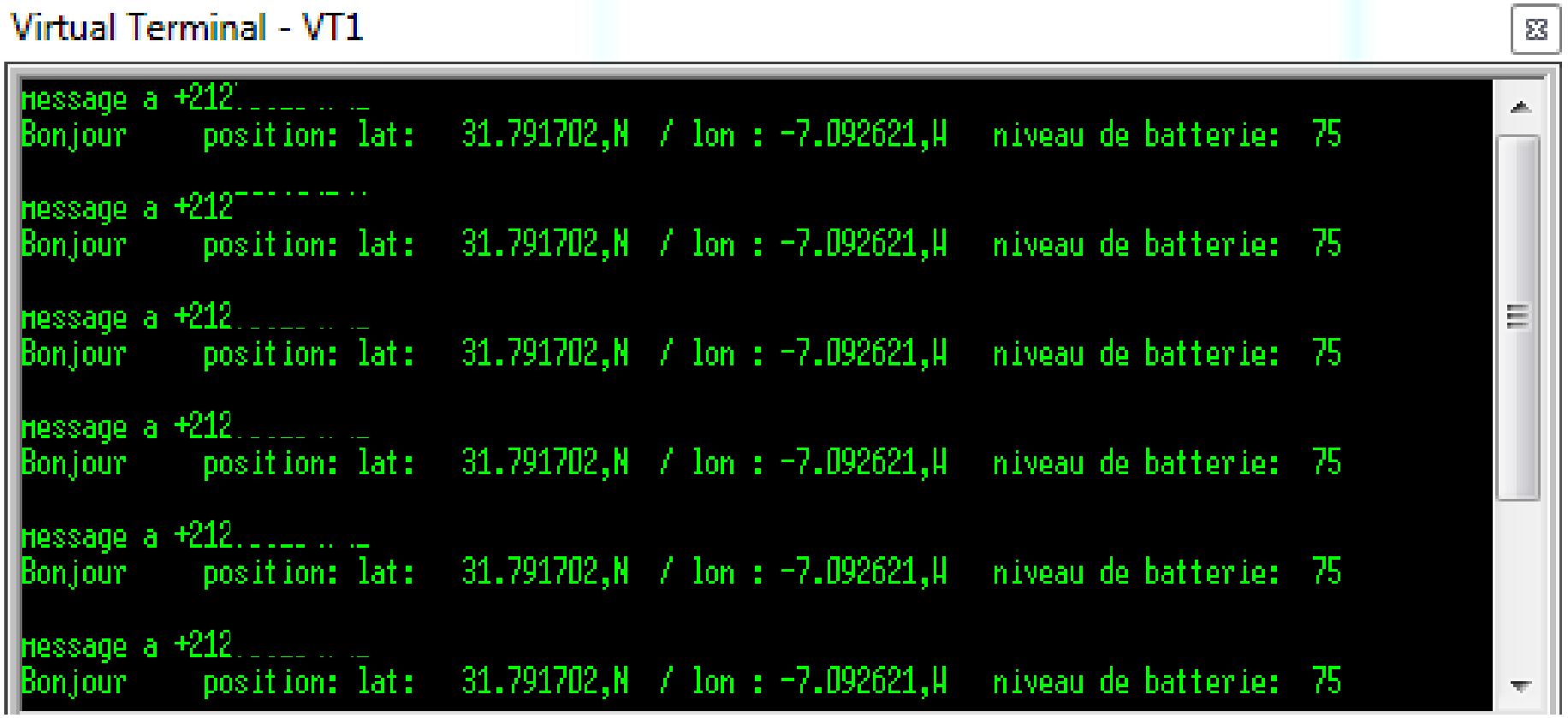
Virtual Terminal

```
$GPRMC,215456.050,A,3147.5021,N,00705.5572,W,000.0,090.0,300621,,,A*77
$GPGGA,215456.050,3147.5021,N,00705.5572,W,1,13,1.50,100.0,M,50.0,M,,*78
$GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,23,02,1.50,1.20,0.90*05
$GPRMC,215457.065,A,3147.5021,N,00705.5572,W,000.0,090.0,300621,,,A*70
$GPGGA,215457.065,3147.5021,N,00705.5572,W,1,13,1.50,100.0,M,50.0,M,,*7F
$GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,23,02,1.50,1.20,0.90*05
$GPRMC,215458.298,A,3147.5021,N,00705.5572,W,000.0,090.0,300621,,,A*7F
$GPGGA,215458.298,3147.5021,N,00705.5572,W,1,13,1.50,100.0,M,50.0,M,,*70
$GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,23,02,1.50,1.20,0.90*05
$GPRMC,215459.596,A,3147.5021,N,00705.5572,W,000.0,090.0,300621,,,A*77
$GPGGA,215459.596,3147.5021,N,00705.5572,W,1,13,1.50,100.0,M,50.0,M,,*78
$GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,23,02,1.50,1.20,0.90*05
$GPRMC,215501.176,A,3147.5021,N,00705.5572,W,000.0,090.0,300621,,,A*71
$GPGGA,215501.176,3147.5021,N,00705.5572,W,1,13,1.50,100.0,M,50.0,M,,*7E
$GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,23,02,1.50,1.20,0.90*05
$GPGGA,215502.360,3147.5021,N,00705.5572,W,1,13,1.50,100.0,M,50.0,M,,*78
$GPGSA,A,3,13,11,20,28,14,18,16,21,22,19,23,02,1.50,1.20,0.90*05
```

La trame GGA :

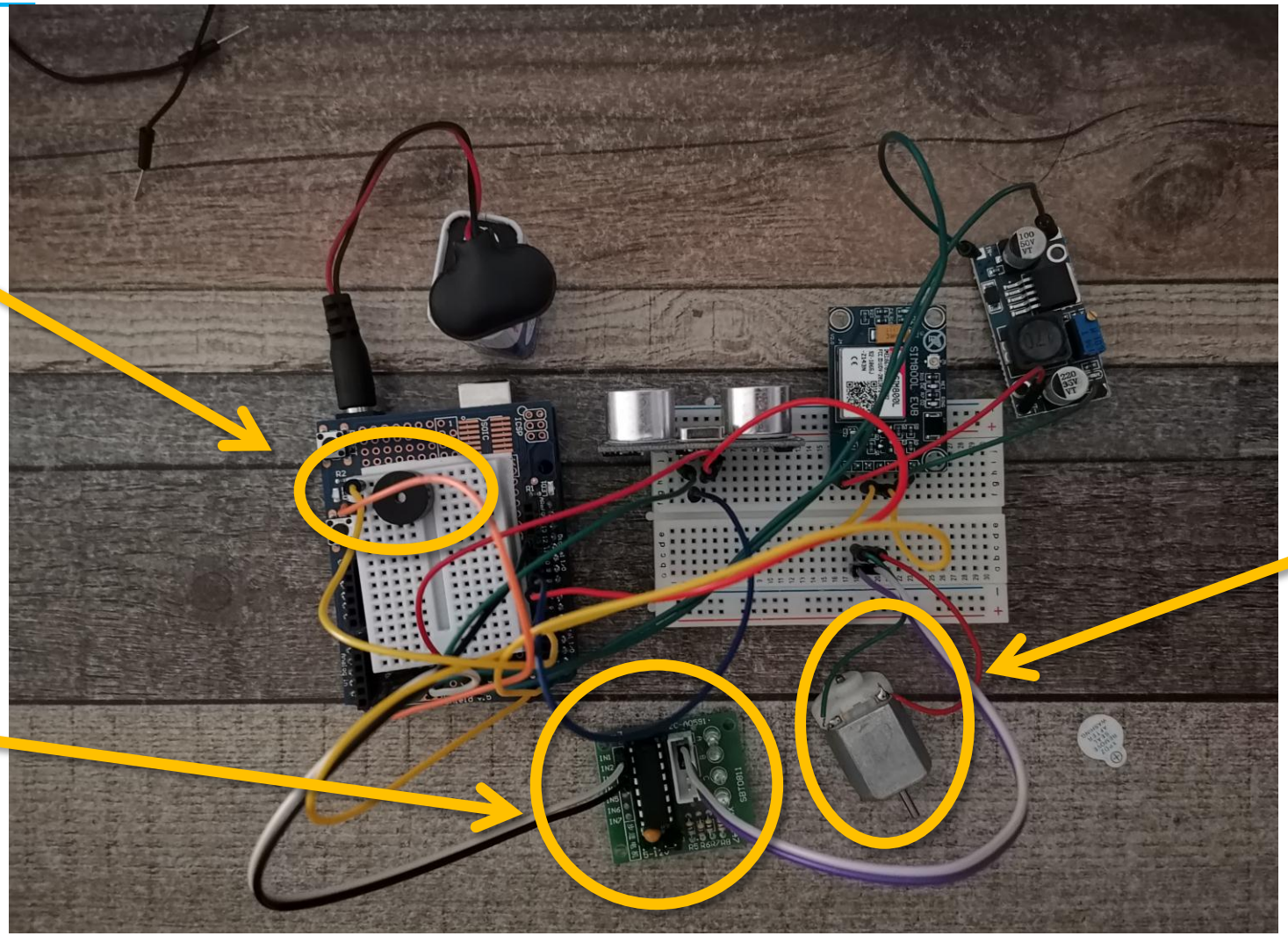


Résultats de simulation : la construction de message à envoyer



Message à envoyer

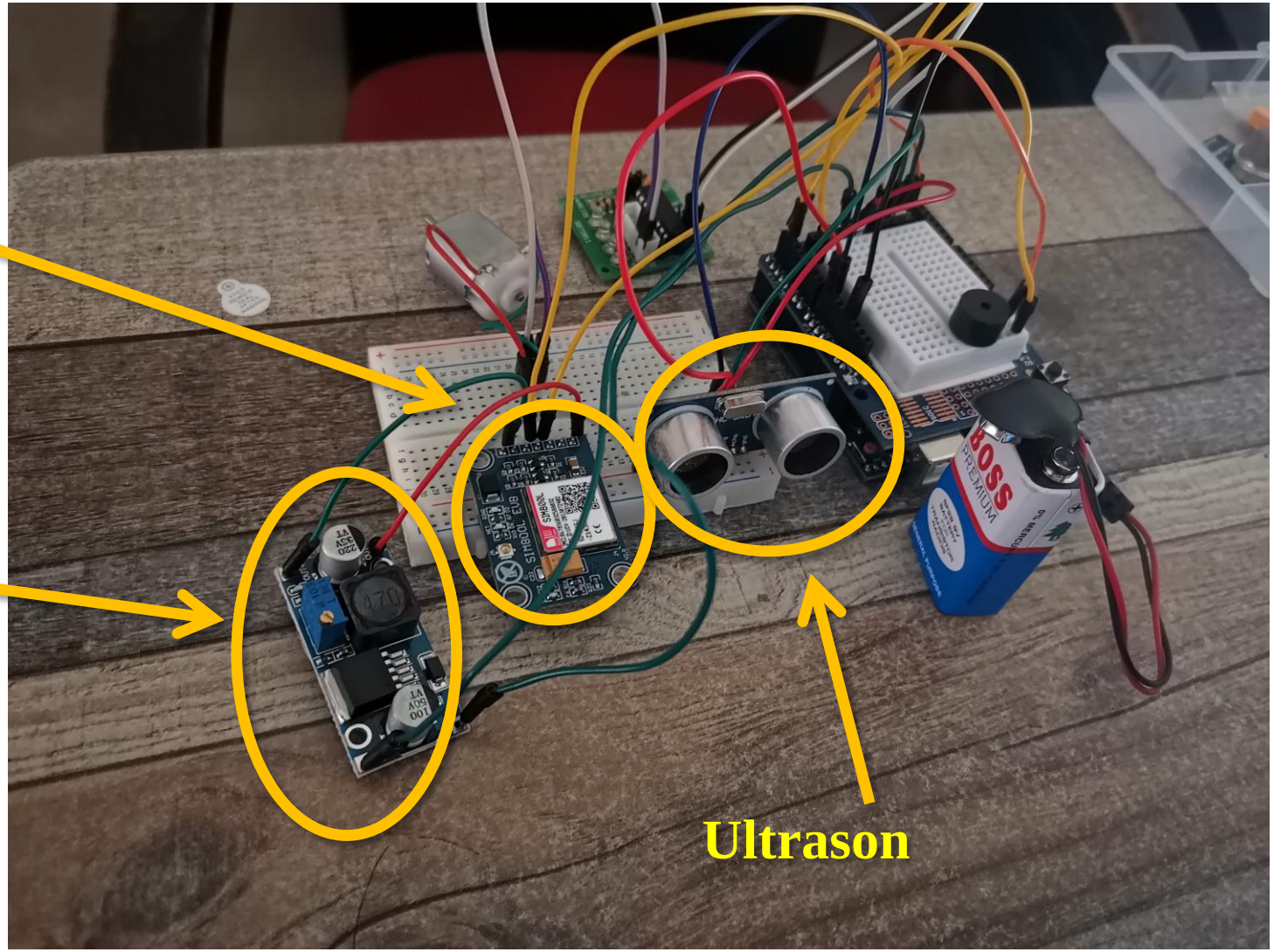
Montage réel



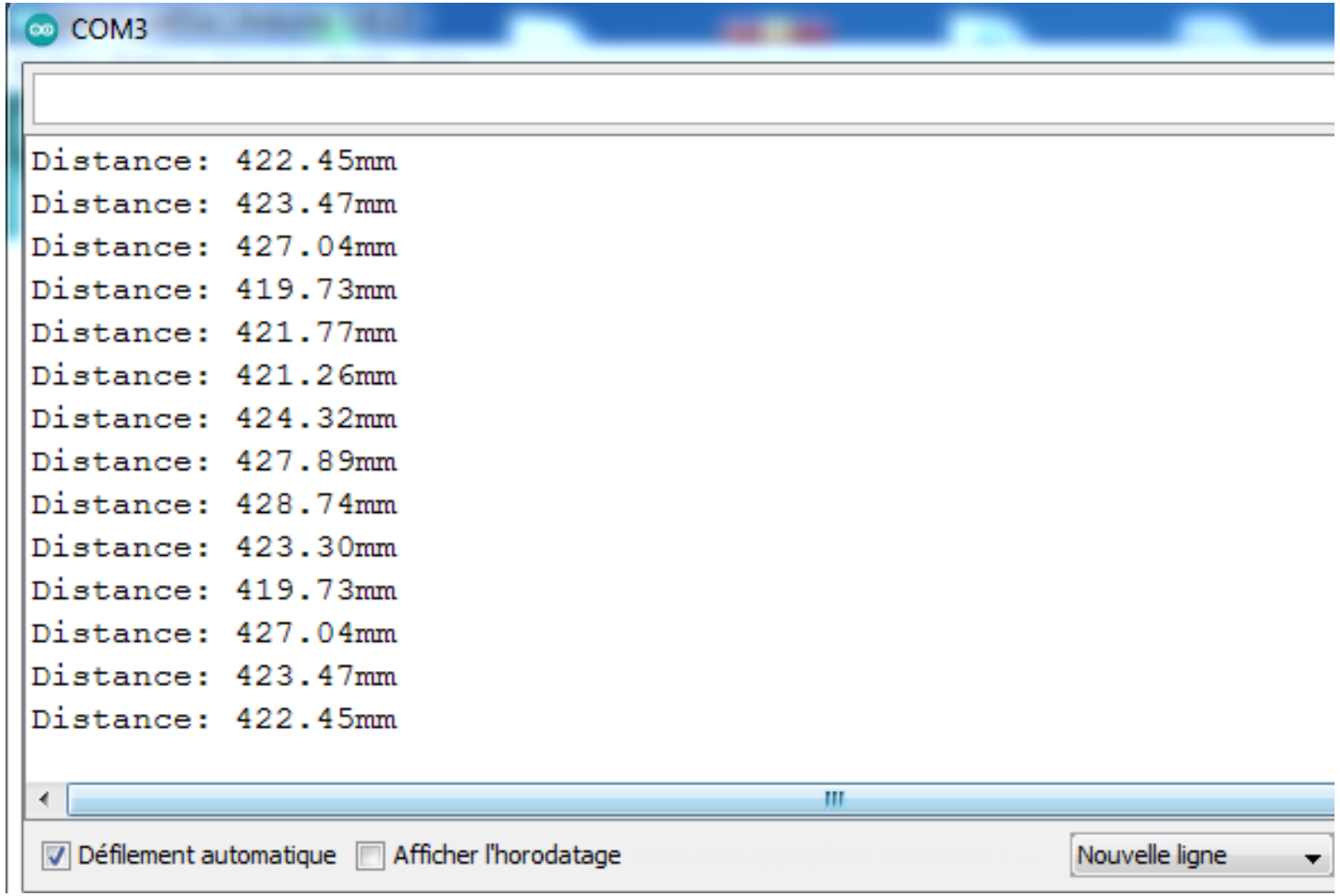
Résultats: ultrason

Module GSM

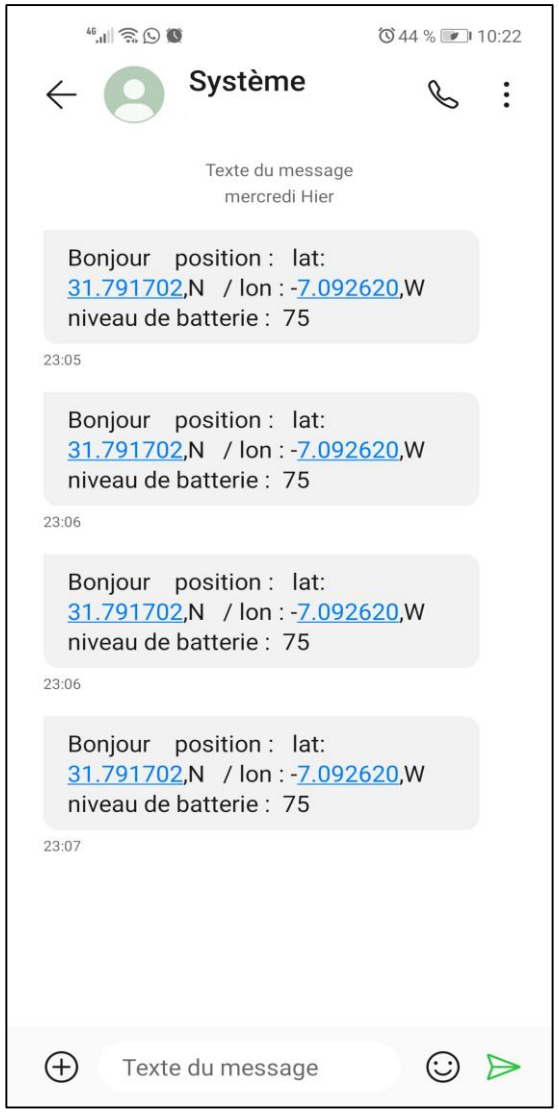
Régulateur de tension
LM2596



Résultats: distance en mm



Résultats: GSM



conclusion

Dans le présent travail de TIPE , il nous a été confié d'analyser des lacunes en certains domaines ; en ce qui concerne les malvoyants, ils affrontent des difficultés en se déplaçant d'un endroit à l'autre, ce qui nous a poussé à faire une analyse de l'existant afin de chercher les causes et proposer par la suite les remèdes à ces causes. L'analyse, le choix et la conception de tel problème nécessite une démarche rigoureuse qui s'appuie sur des savoirs scientifiques et techniques solides.

Comme perspective de ce travail, l'angle de détection de capteur ultrason est courte (15°), ce qui exige d'installer une caméra, donc une traitement d'image pour détecter les objets fixes et/ou les objets en mouvement.

Programme principale

```
/*****************************************************************************
*                              Programme pricipal
*****************************************************************************/
void loop() {
 int j
 float D_mm;

     D_mm=  distance();           // lire la distance

           if (j==1200)           // si j=1200, cette valeur équivalente que t=10 min. si oui
             {
               GPS_Pro();         // prendre la localisation par  GPS (lat et lon)
               Message ();        // construire le message à envoyer
               GSM_Pr();          // evoyer le message via le module GSM
               j=0;               // initialisation de temporisation t=0;
             }
           else if( D<2000)       // si D<2000 mm , donc il y a un obstacle
               alarme(D_mm);      // lancer le Buzzer et le vibreur

     delay(500);                   // temps de réagir 0.5s
     i++;                          // incrémeter la temporisation ix0.5s

}
```

Sous programme de mesure de distance

```
/*************************************************************************
 *                        Mesure de distance D
 *************************************************************************/
int      distance()            // Distance
{

 digitalWrite(TRIGGER_PIN, HIGH);
 delayMicroseconds(10);                    //  générer une impulsion
 digitalWrite(TRIGGER_PIN, LOW);

 long measure = pulseIn(ECHO_PIN, HIGH, MEASURE_TIMEOUT);  // mesure de temps propagation
 float distance_mm = measure / 2.0 * SOUND_SPEED;    // calcul la distance D en mm
 return distance_mm;
}
```

Sous programme vibreur et buzzer

```
/*****************************************************************************
 *                          alarme
 *****************************************************************************/
 void            alarme(float D )        // fonction l'alarme
 {
   int i;
       if (D<2000){                        // si D < 2m
        digitalWrite(VIBR, HIGH);          // lancer le vibreur
  //      digitalWrite(LED, HIGH);
        for (i=0;i<10;i++){
           digitalWrite(sonor, HIGH);
           delay(50);
           digitalWrite(sonor, LOW);       // construire un signal sonor de periode 0.5 s
           delay(50); }
        delay(250);
       }
       else
           {digitalWrite(LED, LOW);digitalWrite(VIBR, LOW);}    //si non, rein à faire
 }
```

Sous programme de l'envoi de message GSM

```
 /*************************************************************************
 *                          programme GSM
 *************************************************************************/
void   GSM_Pr()
{

  sms.beginSMS(remoteNum);        // composé le numéro de téléphone de destination
  sms.print(txtMsg);              // envoyer le message
  sms.endSMS();
}
```

Sous programme de construire le message à envoyer

```
/*************************************************************************
*                          construction de message
*************************************************************************/
void                Message ()
{
char BJ[7]="bonjour";
char P[10]="position: ";
char NB[18]="niveau de batterie: 75";
char latt[9],lontt[9];

 latt = String (flat, 6);
 lontt = String (flon, 6);

for(i=0;i<7;i++)
   txtMsg[i]=BJ[i];                    // coupier "bonjour" dans le tableau txtMsg
for(i=7;i<17;i++)
   txtMsg[i]=P[i];                      // coupier "position " dans le tableau txtMsg
for(i=17;i<26;i++){
   txtMsg[i]=latt[9];                   // coupier latitude dans le tableau txtMsg
   txtMsg[27]="/"                       // separation
   txtMsg[i]=lontt[i+11]; }             // coupier longitude dans le tableau txtMsg

 for(i=47;i<70;i++)
   txtMsg[i]=P[i];                      // coupier le niveau de batterie dans
                                         // le tableau txtMsg (partie n'est pas traitée)
```

Sous programme de localisation par GPS

1

```
/*******************************************************************
*                          GPS
*******************************************************************/
void  GPS_Pro()
{
   bool newData = false;
 unsigned long chars;
 unsigned short sentences, failed;

 // For one second we parse GPS data and report some key values
 for (unsigned long start = millis(); millis() - start < 1000;)
 {
   while (ss.available())
   {
     char c = ss.read();
     if (gps.encode(c))
       newData = true;
   }
 }
```

2

```
if (newData)
{

  unsigned long age;
  gps.f_get_position(&flat, &flon, &age);       // lire la trame et extraire l'altitude et longitude
  Serial.print("LAT=");
  Serial.print(flat == TinyGPS::GPS_INVALID_F_ANGLE ? 0.0 : flat, 6);
  Serial.print(" LON=");
  Serial.print(flon == TinyGPS::GPS_INVALID_F_ANGLE ? 0.0 : flon, 6);
  Serial.print(" SAT=");
  Serial.println(gps.satellites() == TinyGPS::GPS_INVALID_SATELLITES ? 0 : gps.satellites());
                    // lire nombre de satellite qui réalise la localisatio

 }
 gps.stats(&chars, &sentences, &failed);
 if (chars == 0)
   Serial.println("** No characters received from GPS: check wiring **");
}
```

Déclaration des variables et des bibliothèques

tipe_pro §

```
#include <SoftwareSerial.h>           // Bus série programmé
#include <TinyGPS.h>                  // bibliothèque GPS
#include <GSM.h>                      // bibliothèque GSM
#define PINNUMBER "0000"              // Code PIN


const byte  sonor=6;                        // Bezzer dans la brouche 6
const byte  VIBR=12;                        // Bezzer dans la brouche 12
const byte  LED=11;                         // Bezzer dans la brouche 11
const byte TRIGGER_PIN = 8;                 // Brouche TRIGGER
const byte ECHO_PIN = 9;                    // Brouche ECHO
const unsigned long MEASURE_TIMEOUT = 25000UL; // 25ms = ~8m à 340m/s
const float SOUND_SPEED = 340.0 / 1000;         // la distance en mm
float flat, flon;                           // déclaration lat ( latitude) lon (longitude)
char remoteNum[20]="+212'         ";        // définir le numéro de téléphone de destination
char txtMsg[150];                           // déclaration du message à envoyer

TinyGPS gps;                                // class gps
SoftwareSerial ss(2, 3);                    // le bus série programme 2: TX et 3: RX

GSM gsmAccess;
GSM_SMS sms;
```