

# Modélisation d'un dispositif alternatif à une canne pour malvoyants

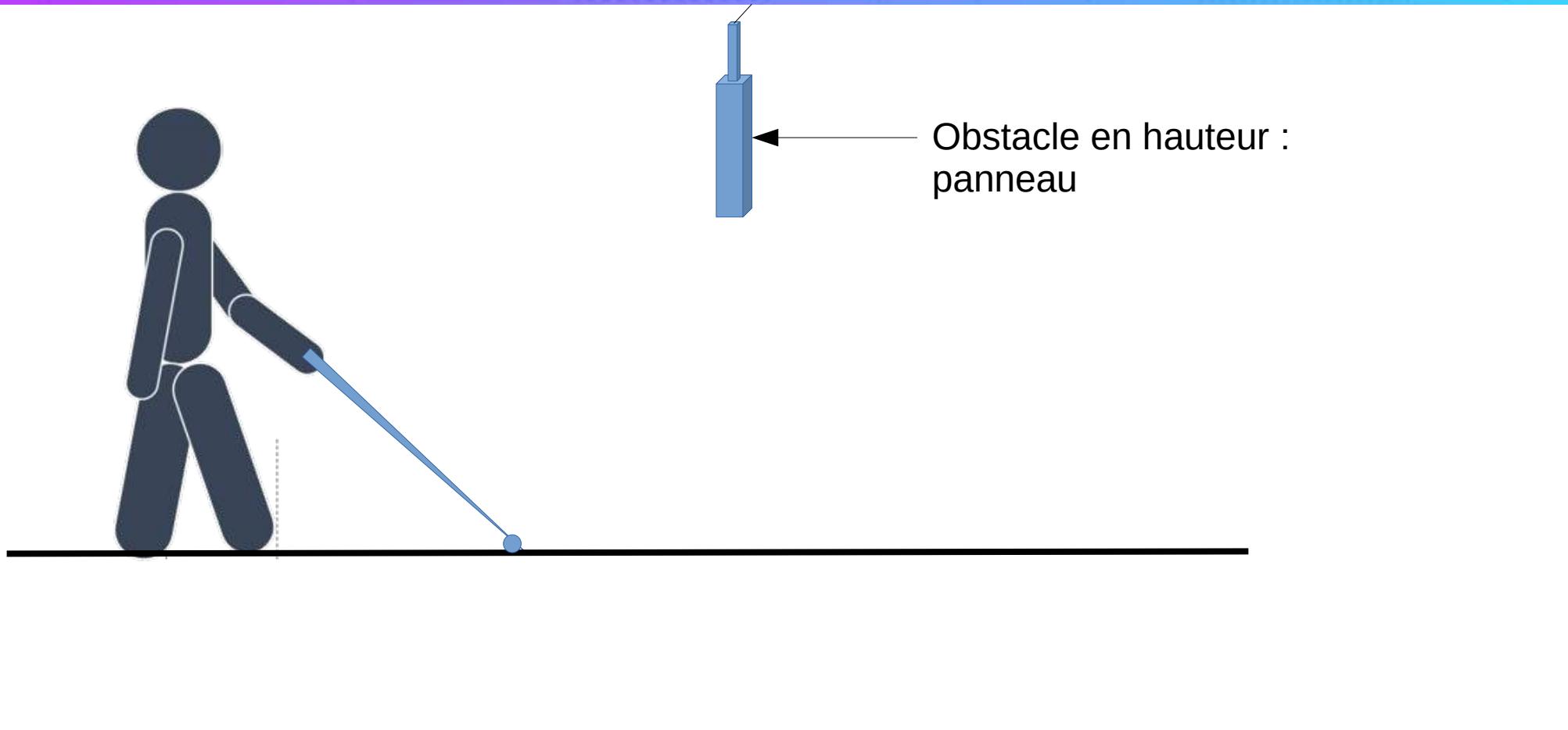
Comment permettre aux personnes malvoyantes d'éviter les obstacles en gardant les mains libres ?



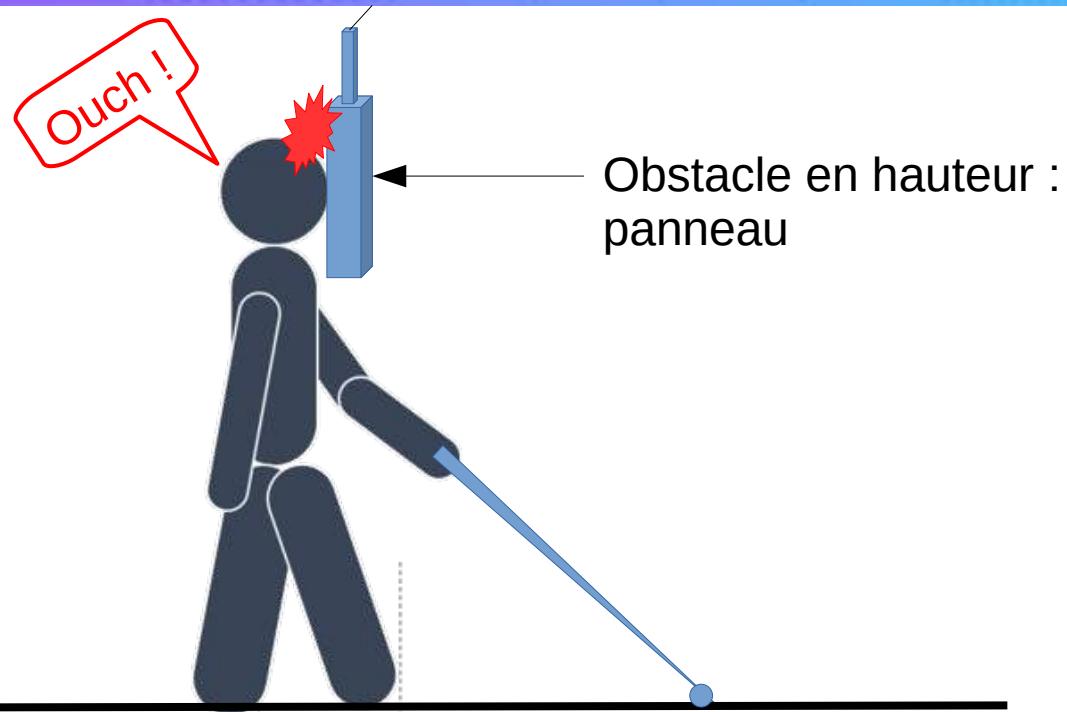
# Introduction

- En 2015, 36M de personnes non voyantes
- 217M de déficients visuels
- x3 d'ici 2050
- 1 seule main libre
- Obstacles en hauteur
- Cannes intelligentes chères (~1000€)

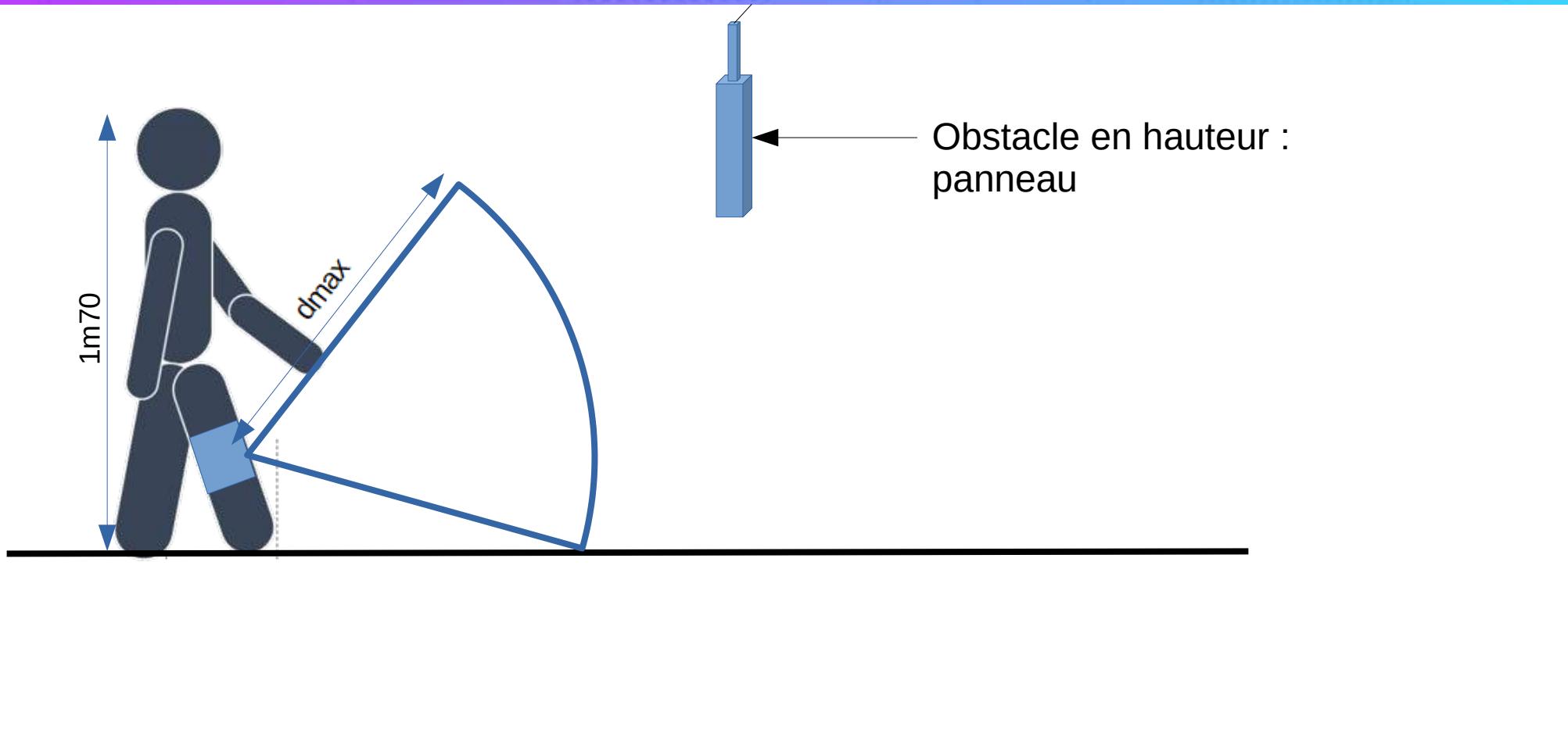
# Avec une canne



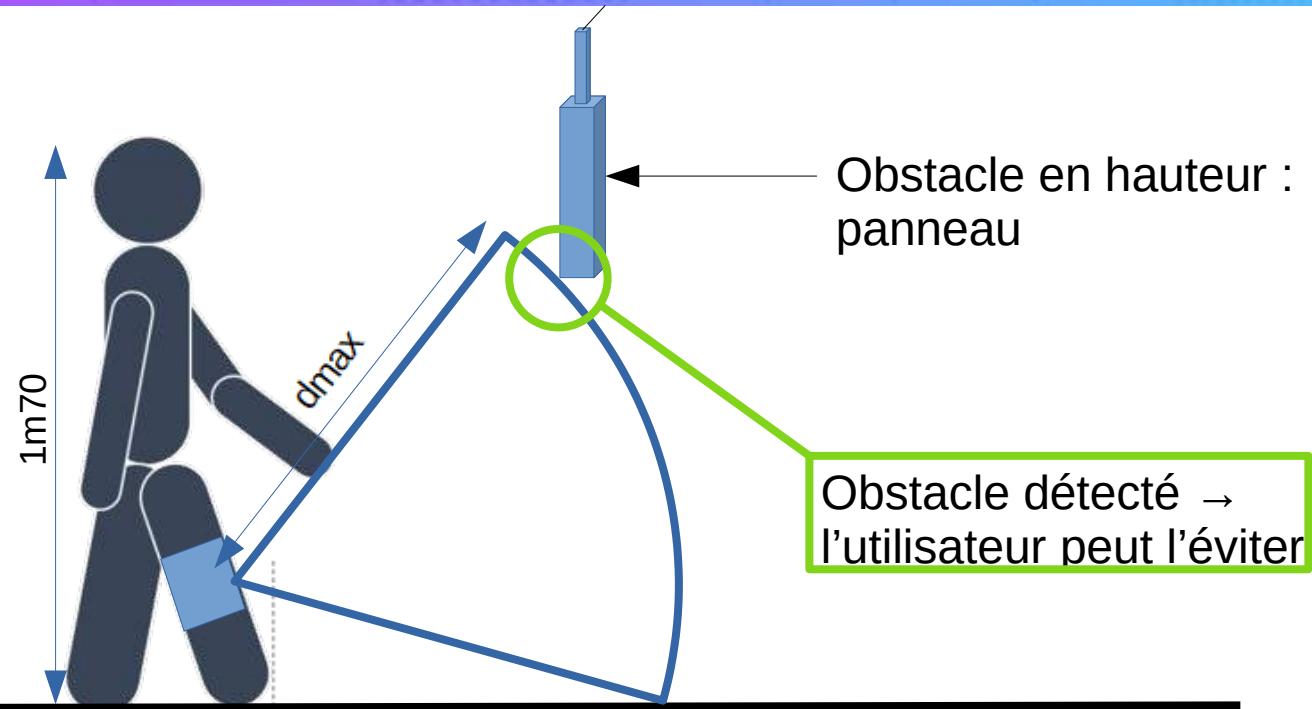
# Avec une canne



# Avec la genouillère



# Avec la genouillère

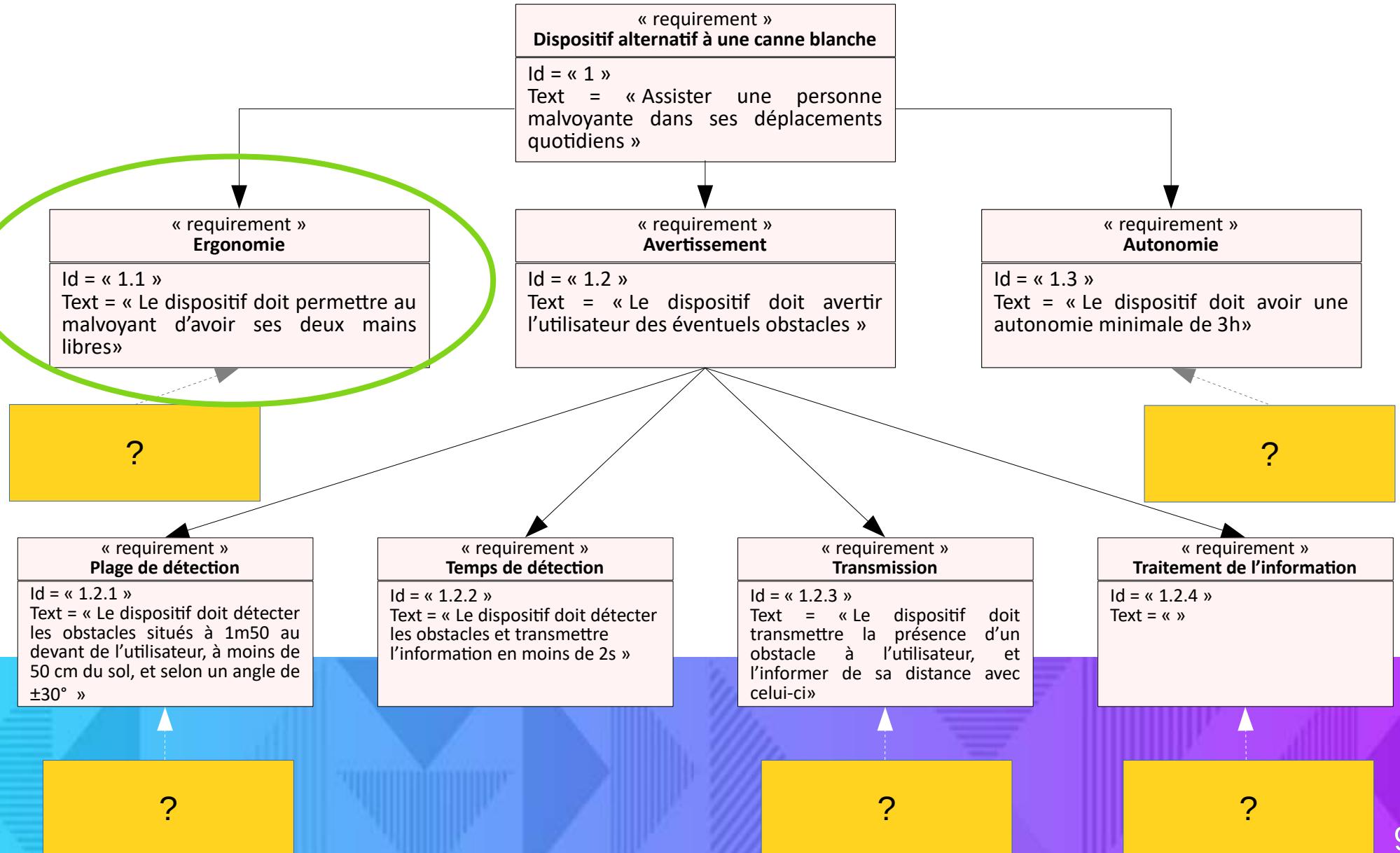


# Sommaire

- Présentation du projet
- Choix des composants
- Programmation de la carte arduino
- Modélisation et impression du support
- Résultat final
- Amélioration du dispositif

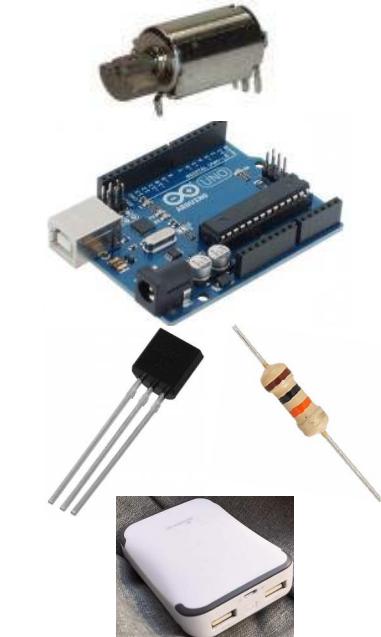
# Présentation du projet

# Diagramme des exigences

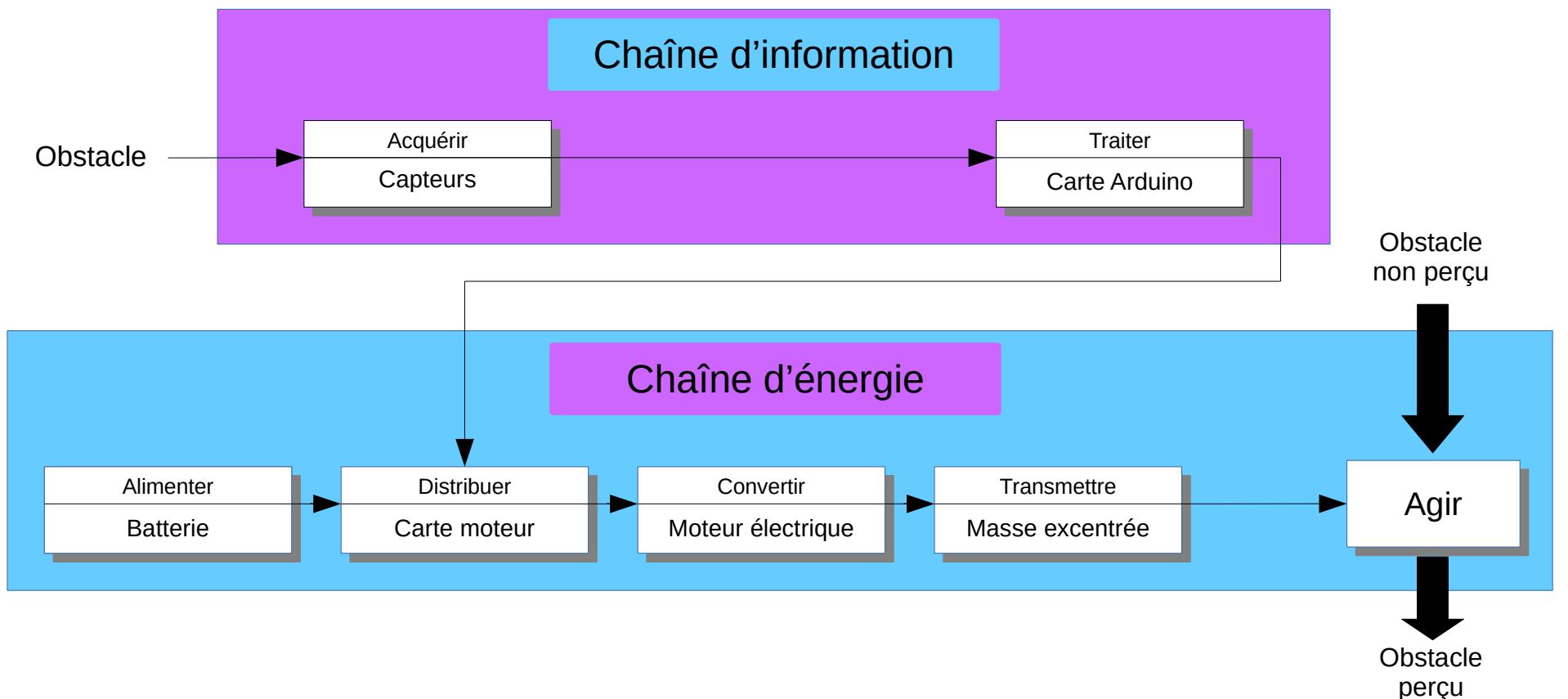


# Description du dispositif

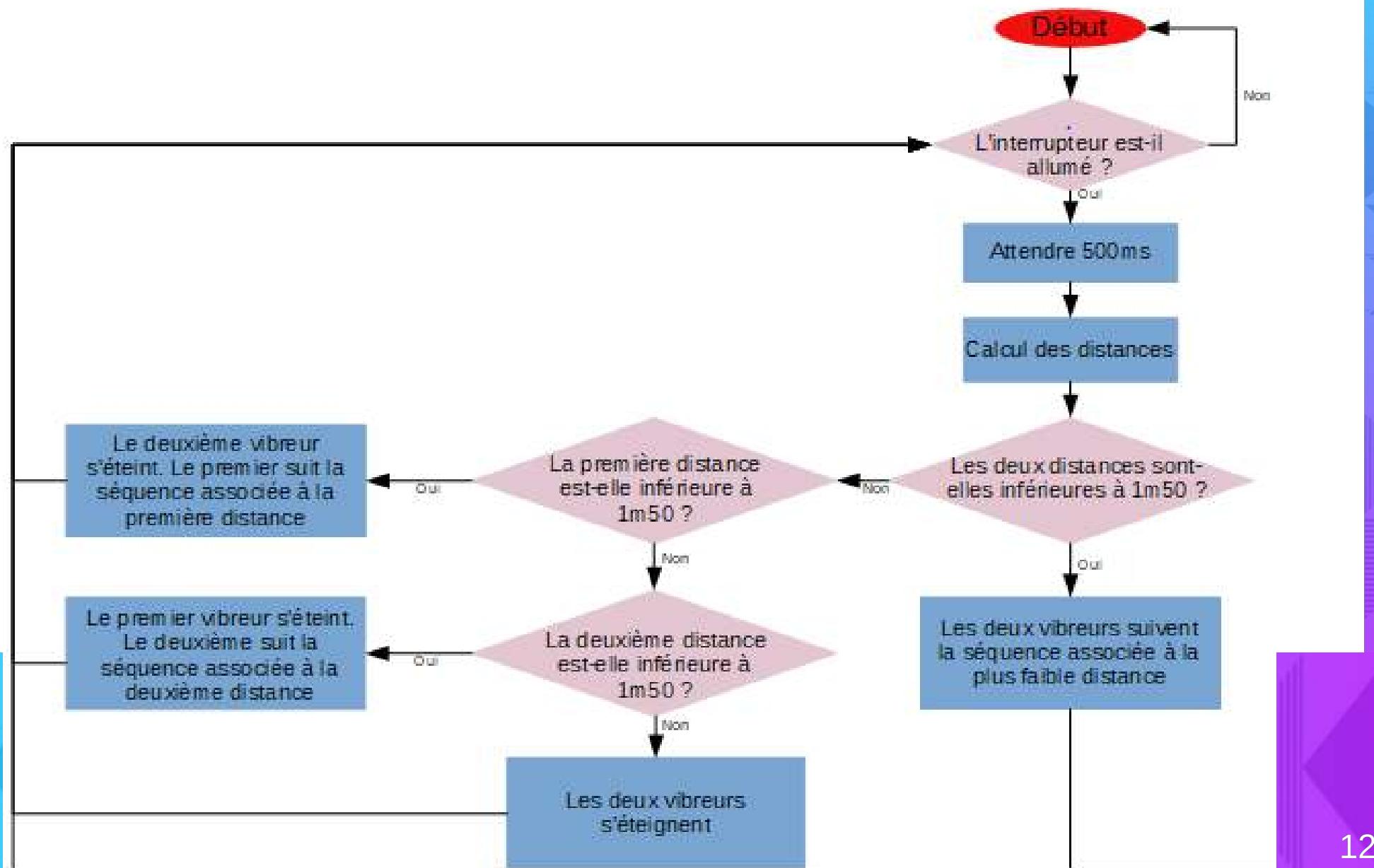
- Genouillère
- Capteurs
- Moteurs à masse excentrée (C6070)
- Carte arduino uno
- Amplificateur d'amplitude (transistor+résistance)
- Batterie



# Chaîne fonctionnelle



# Algorithme



# Choix des composants

# Choix de la batterie

## MediaRange MR742

- à disposition
- forme pratique
- $U_{charge} \sim 6V$  ;  $U_{alimentation arduino} \geq 6V$
- environ 500 cycles décharge-charge



### ➤ autonomie suffisante

$$C = 6600\text{mAh}$$

$$I < 1000\text{mA}$$

$$A = C / I : A > 6h36min$$

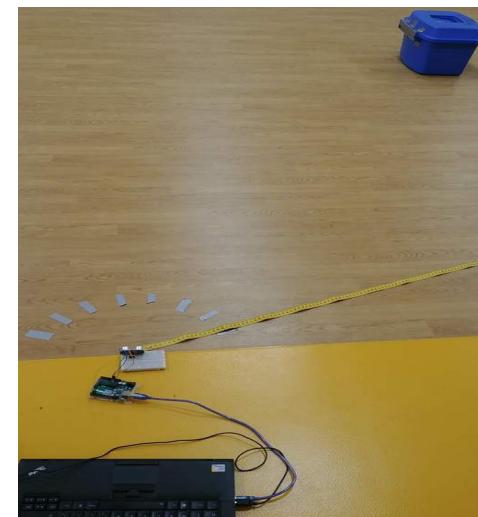
« requirement »  
**Autonomie**

Id = « 1.3 »  
Text = « Le dispositif doit avoir une autonomie minimale de 3h»

# Choix des capteurs

Comment détecter efficacement les obstacles ?

	Capteur ultrasonore	Capteur laser
Distance de détection frontale maximale	3m70	>5m
Plage de détection angulaire	$\pm 35^\circ$	$\pm 5^\circ$
Précision pour un obstacle à moins de 1m50	$\leq 4\text{cm}$	$< 1\text{cm}$



# Choix des capteurs

Comment détecter efficacement les obstacles ?

$\pm 35^\circ \rightarrow [115.5 ; 137]$

$\pm 30^\circ \rightarrow [105 ; 169]$

$0^\circ \rightarrow [18 ; 370]$

➤ tourner avant d'avancer

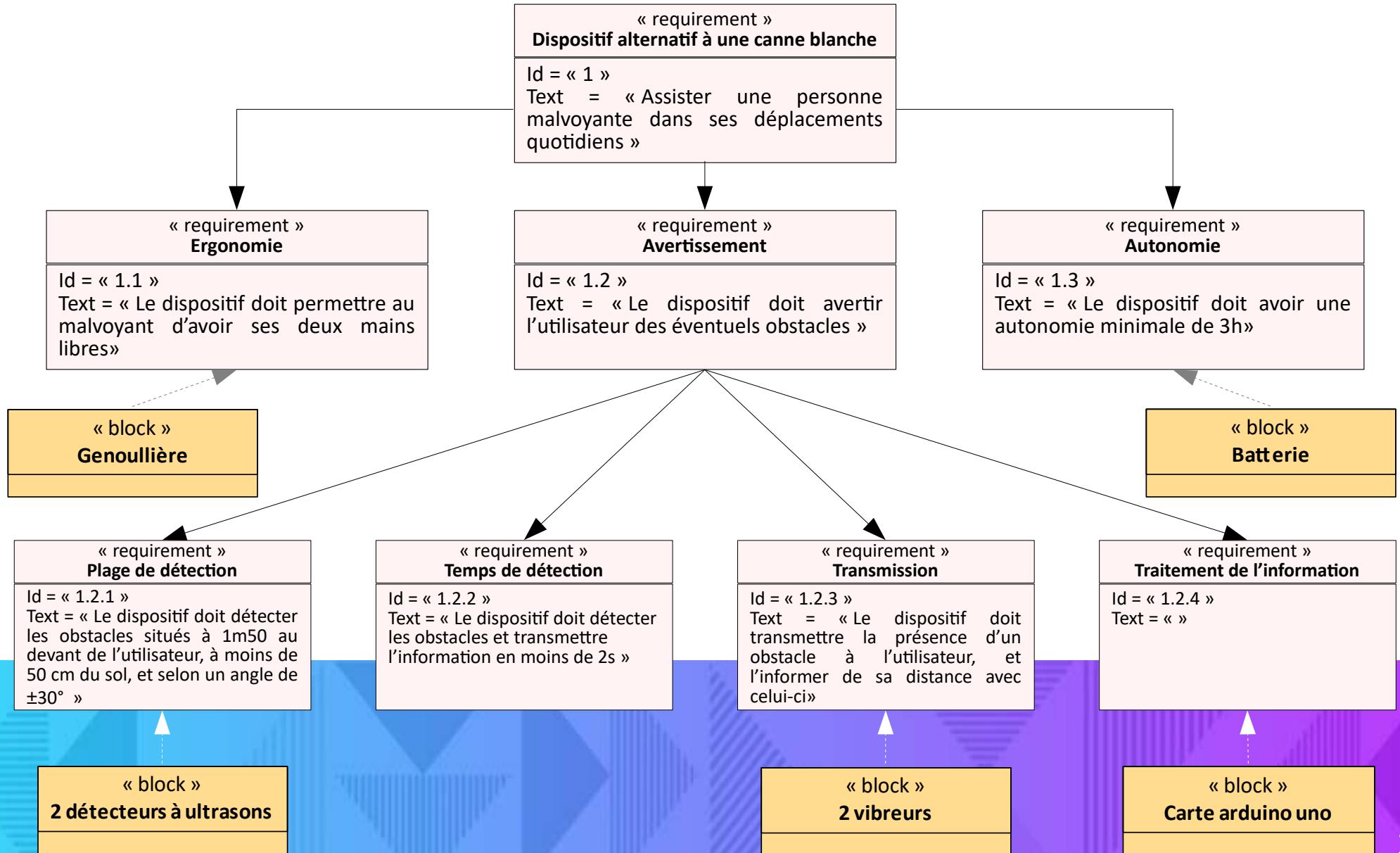
	Capteur ultrasonore	Capteur laser	« requirement » Plage de détection
Distance de détection frontale maximale	3m70	>5m	Id = « 1.2.1 » Text = « Le dispositif doit détecter les obstacles situés à 1m50 au devant de l'utilisateur, à moins de 50 cm du sol, et selon un angle de $\pm 30^\circ$ »
Plage de détection angulaire	$\pm 35^\circ$	$\pm 5^\circ$	
Précision pour un obstacle à moins de 1m50	$\leq 4\text{cm}$	$< 1\text{cm}$	



Module de détection US HC-SR04



# Diagramme des exigences



# Programmation de la carte arduino

# Informer l'utilisateur

## Calcul des distances

- $d_{(\text{cm})} = 0,017 * \Delta t_{(\mu\text{s})}$  (voir annexes)

## Comment prévenir l'utilisateur de la position des obstacles ?

- 2 moteurs, chacun associé à un capteur

Durant chaque boucle :

- **Calcul des distances**
- Pendant  $\Delta t = \text{fct}(\text{distance})$ ,  $\omega \sim 0 \text{ rad/s}$
- Puis  $\omega = \text{fct}(\text{distance})$  ; si distance > 1m50,  $\omega = 0 \text{ rad/s}$

Variation :  
de la **fréquence** de vibration  
de l'**intensité** de vibration

# Branchements

```

int trighaut = 2;
int echohaut = 3;
const byte vibreuhaut = 6;

int trigbas = 4;
int echobas= 5;
const byte vibreurbas = 9;

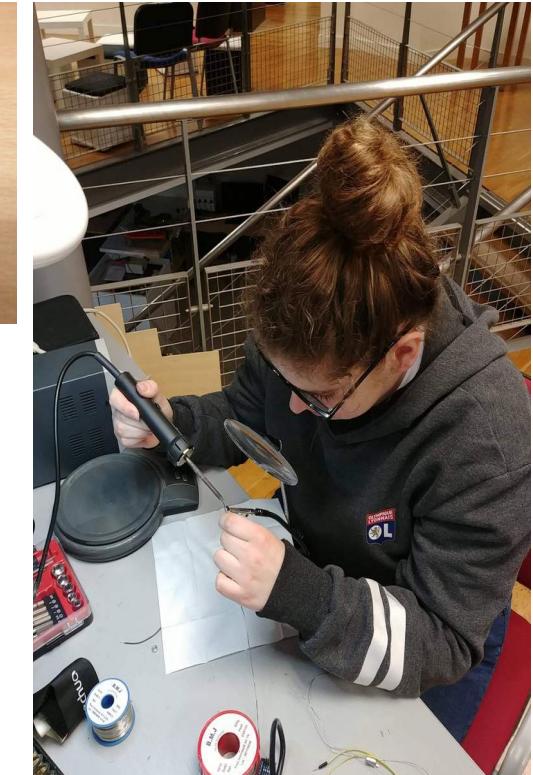
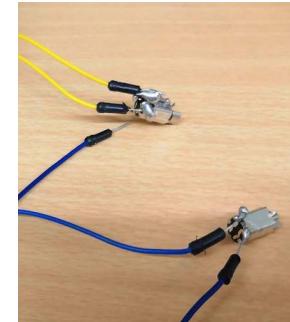
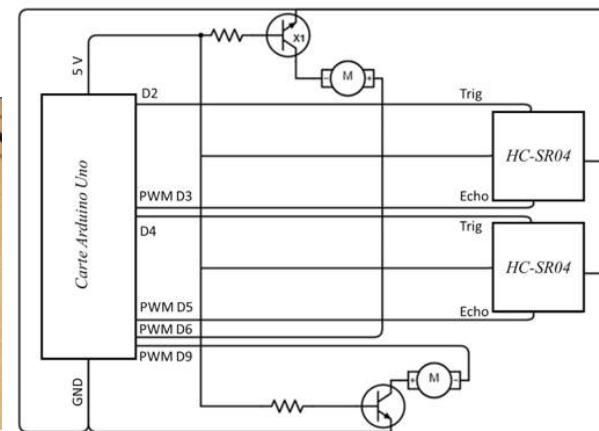
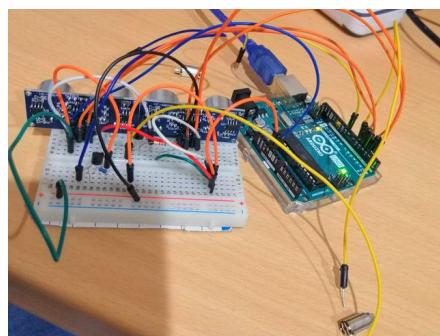
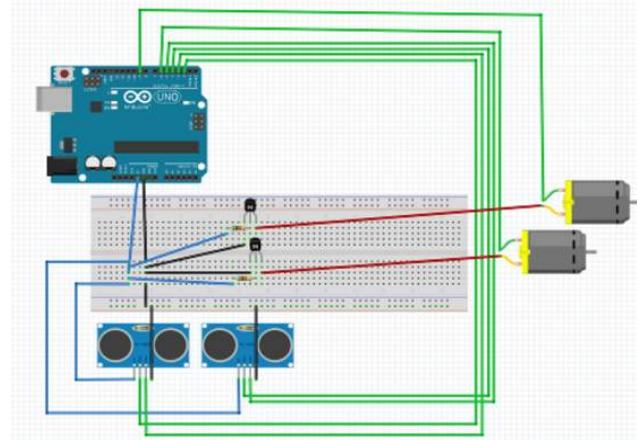
float lecture_echohaut,lecture_echobas;
float cmhaut,cmbas,cm;
int vibration = 0, intensite = 0;

void setup()
{
    pinMode(vibreuhaut, OUTPUT);
    pinMode(trighaut, OUTPUT);
    digitalWrite(trighaut, LOW);
    pinMode(echohaut, INPUT);

    pinMode(vibreurbas, OUTPUT);
    pinMode(trigbas, OUTPUT);
    digitalWrite(trigbas, LOW);
    pinMode(echobas, INPUT);

    Serial.begin(9600);
}

```



# Boucle principale

```
void loop()
{
    distancehaut();
    distancebas();

    if (cmhaut<150 && cmbas<150) {
        vibrations(cmhaut,cmbas);
    }
    else if (cmhaut>=150) {
        analogWrite(vibreurhaut,0);
        if (cmbas>=150){
            analogWrite(vibreurbas,0);
        }
        else {
            vibrationsseul(cmbas,vibreurbas);
        }
    }
    else {
        analogWrite(vibreurbas,0);
        vibrationsseul(cmhaut,vibreurhaut);
    }
}
```

# Définition des fonctions annexes

```
void distancehaut()
{
//on envoie une impulsion au premier capteur durant 10ms
digitalWrite(trighaut, HIGH);
delayMicroseconds(10);
digitalWrite(trighaut, LOW);

//mesure le temps entre la fin de l'impulsion et le
//retour des ultrasons au capteur (passage de LOW à HIGH)
lecture_echohaut = pulseIn(echohaut, HIGH);
//conversion du temps mesuré pour obtenir la distance
//séparant le capteur de l'obstacle
cmhaut = lecture_echohaut*0.017;
}
```

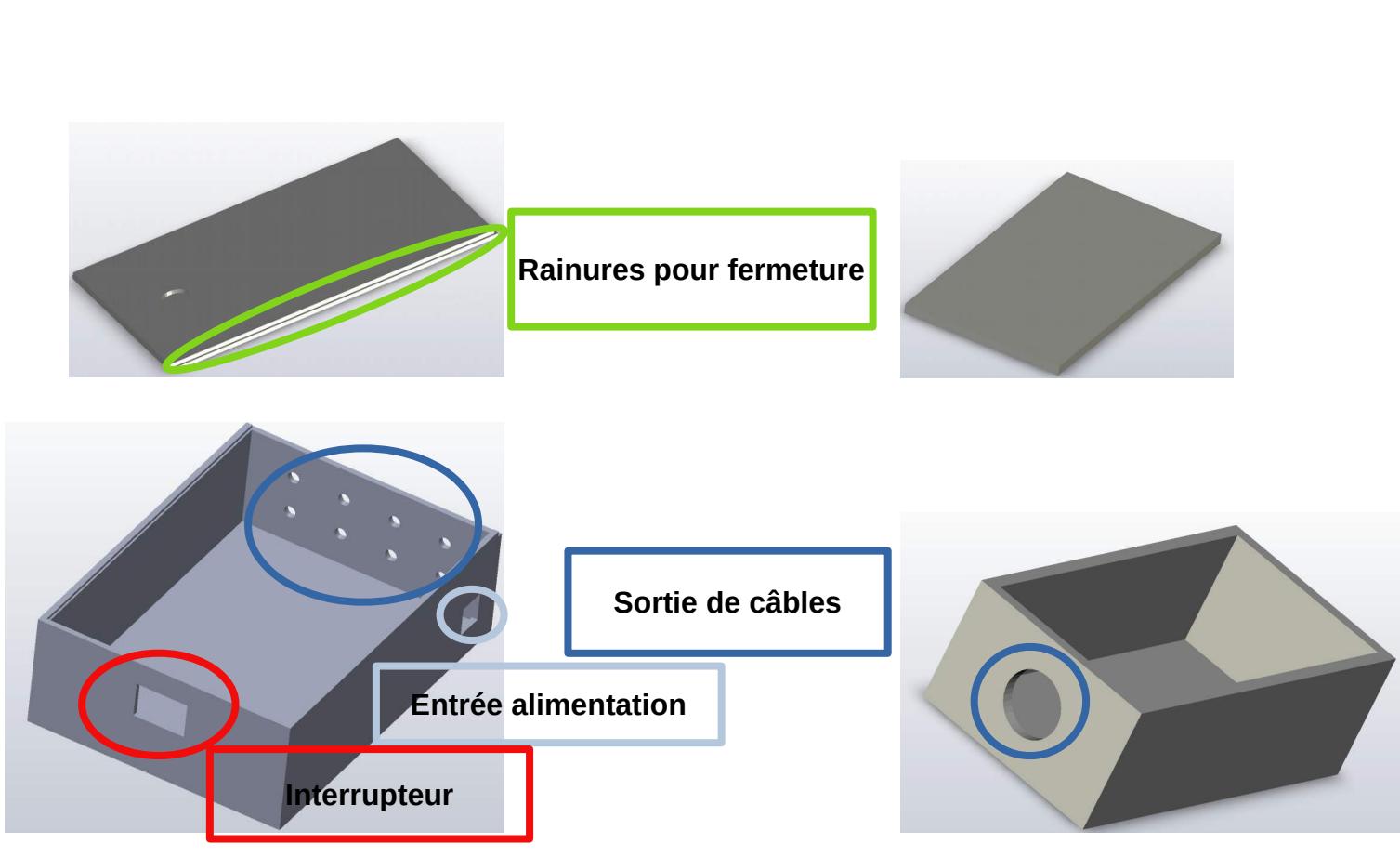
```
void distancebas()
{
digitalWrite(trigbas, HIGH);
delayMicroseconds(10);
digitalWrite(trigbas, LOW);
lecture_echobas = pulseIn(echobas, HIGH);
cmbas = lecture_echobas*0.017;
}
```

```
void vibrations(int cmh, int cmb)
{
cm=min(cmh,cmb)
//fonction affine (distance, borneinf, bornesup, viborneinf, vibbornesup)
vibrationh = map(cm, 0, 150, 255, 100);
duree = map(cm,0,150,10,1430);
analogWrite(vibreuhaut,70);
analogWrite(vibreurbas,70);
delay(duree);
analogWrite(vibreuhaut, vibration);
analogWrite(vibreurbas, vibration);
delay(500);
}

void vibrationsseul(int cm,int vibreur)
{
vibration = map(cm, 0, 150, 255, 100);
duree = map(cm,0,150,10,1430);
analogWrite(vibreur,70);
delay(duree);
analogWrite(vibreur, vibration);
delay(500);
}
```

# Modélisation et impression du support

# Maquette en carton et modèles solidworks

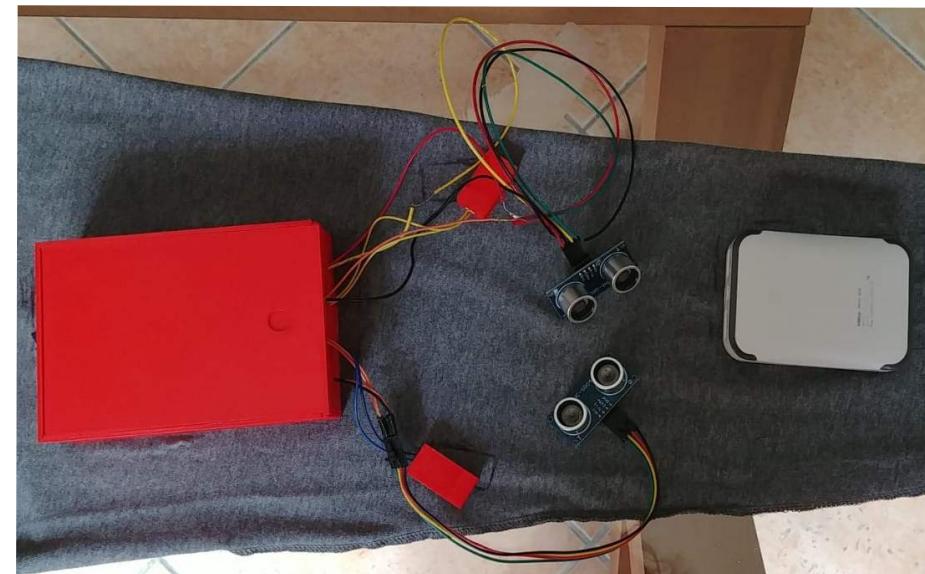
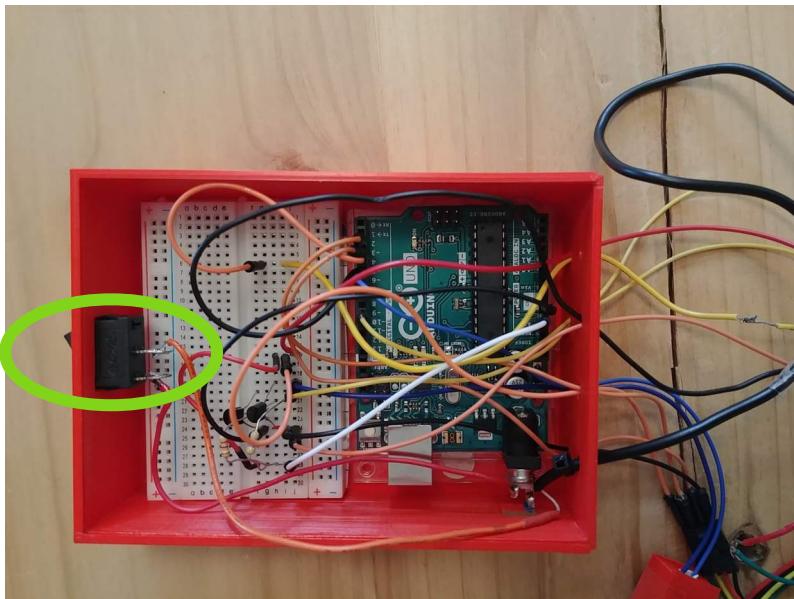


# Impression des pièces



# Résultat final

# Montage



- **Interrupteur à l'extérieur de la boîte**
  - meilleure autonomie
  - accessible, pratique

- **Morceau de legging**

- **Genouillère :**  
maintient vibrateurs

# Assemblage



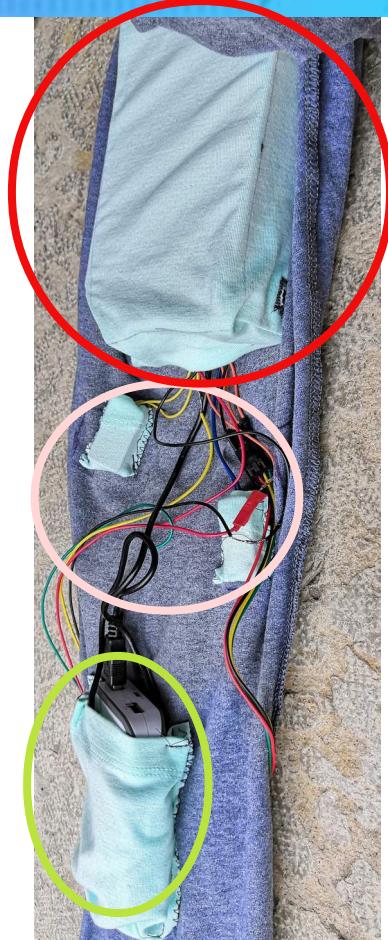
Boîte  
(carte arduino)

Vibreurs

Capteurs  
Hauteur : 35cm



Batterie



Vue de devant

Vue de derrière

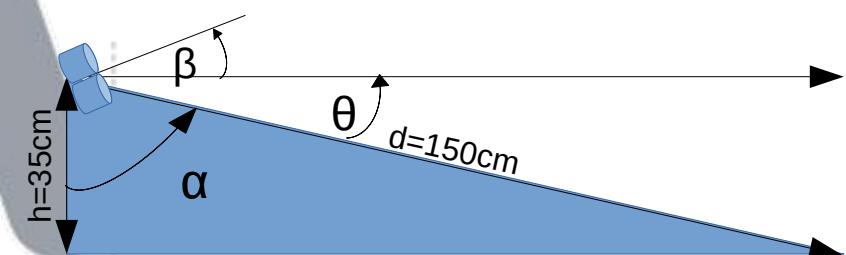
# Essai du dispositif



- ✓ Bonne réactivité
- ✓ Ne gêne pas pour s'asseoir
- ✗ Distance pas précisément perceptible (nécessité d'adaptation)
- ✗ Détection d'obstacles inexistant
- ✗ Détection permanente tant que la jambe n'est pas assez penchée
- ✗ Peu discret (couleur « jean »)?

# Amélioration du dispositif

# Origine des problèmes

- Détection d'obstacles inexistant → valeur de la **distance bloquée à 0**  
Recherches sur internet : absence d'obstacle, occultation du récepteur (doigt), absorption du rayon (vêtement en laine)
  - Détection permanente pour  $\beta \leq \beta_{\min}$  → **détection du sol** car  $\theta_{\text{perception}}$  et  $d_{\text{autorisée}}$  trop importants
  - Distance pas précisément perceptible → faible différence de puissances de vibrations
- 
- Calculs ►
- $h=35\text{cm}$
- $\beta$
- $\alpha$
- $\theta$
- $d=150\text{cm}$
- $\cos\alpha = 35/150 \rightarrow \alpha = 76,5^\circ = 90^\circ - \theta$
- $\theta = 13,5^\circ$
- Détection du sol pour  $\beta \leq 21,5^\circ$

# Solutions aux problèmes

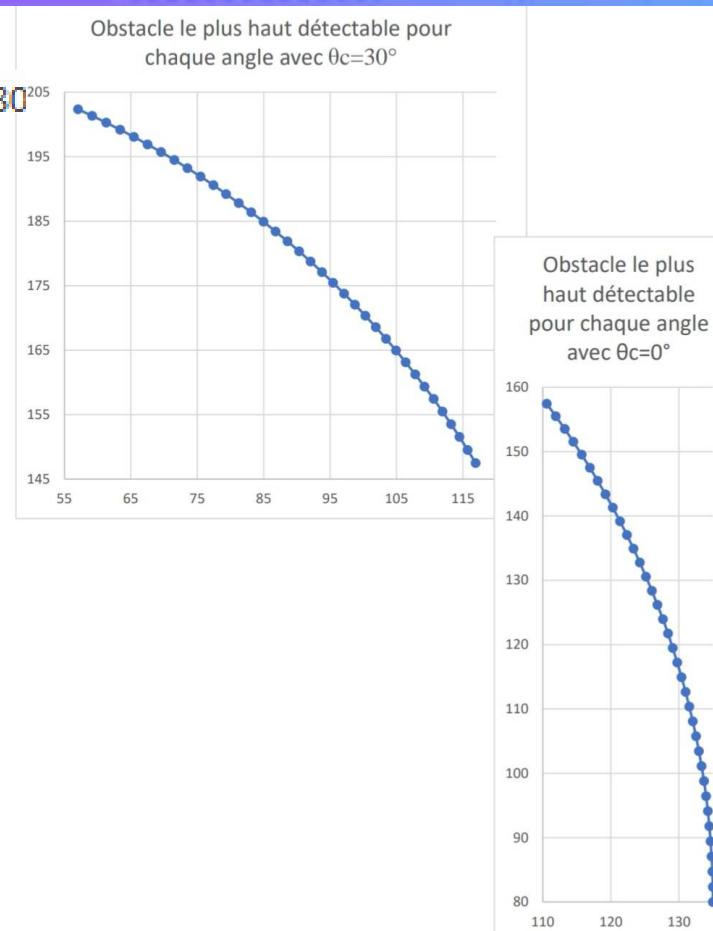
- Distance bloquée à 0 → signal bas pendant 100ms sur ECHO
- Détection du sol → capteur horizontal à 35cm,  $d_{\text{prog}}=55\text{cm}$  ; capteur vertical à 80cm,  $d_{\text{prog}}=135\text{cm}$  (voir calculs en annexe)
- Distance pas précisément perceptible → association de 1 vibreur bas + 3 hauts  
Variation de distance haute → variation du nombre de vibrateurs en action
  - voir le programme amélioré en annexe

# Plage de détection en hauteur du capteur haut

$$\text{Hauteur}(\theta) = 135 * \text{SIN}(\text{PI}() / 180 * (\theta_c + \theta)) + 80$$

Pour la cuisse inclinée

$$\Delta d = d_{\text{mesurée}} - d_{\text{horizontale}} < 80\text{cm}$$



Pour la cuisse droite

$$\Delta d = d_{\text{mesurée}} - d_{\text{horizontale}} < 25\text{cm}$$

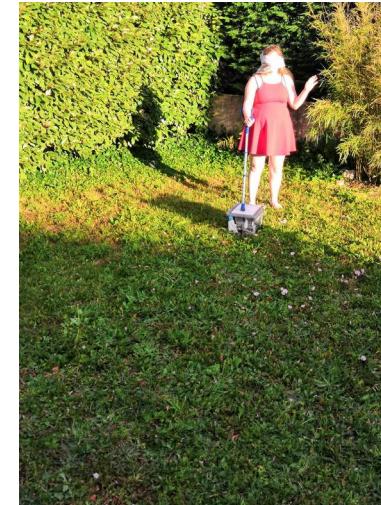
$$\text{Distance}(\theta) = 135 * \text{COS}(\text{PI}() / 180 * (\theta_c + \theta))$$

# Batterie de tests avec une canne pour non voyant puis avec notre dispositif

- Legging non modifié : capteurs à 35cm, un vertical avec  $d_{\text{prog}} = 85\text{cm}$  (calcul en annexes)  
un horizontal avec  $d_{\text{prog}} = 55\text{cm}$
- 1 parcours à travers 6 obstacles
- 4 chasses à l'obstacle



# Batterie de tests avec une canne pour non voyant puis avec notre dispositif



	Parcours	Chasse n°1	Chasse n°2	Chasse n°3	Chasse n°4
<b>Canne</b>	<b>50s</b> (3 obstacles touchés avec le corps)	<b>1min45</b>	2min30 (problème avec arbre)	<b>1min36</b>	<b>2min34</b>
<b>Dispositif</b>	1min48 ( <b>1 obstacle</b> touché avec le corps)	1min58	<b>2min02</b>	8min (capteurs débranchés)	2min54

# Conclusion

	Coût	Mains-libres	Contact avec l'obstacle	Nécessité d'adaptation	Poids	Autonomie	Distance détectable
<b>Canne blanche classique</b>	40€	Non	Oui	Faible	200g	$\infty$	Taille de la canne ie <1m
<b>Notre dispositif</b>	50€ / 60€ avec dynamo	Oui	Non	Élevée	550g	> 6h36min $\infty$ avec dynamo	1m35







# Annexes

- Fond d'écran :

<https://fr.venngage.com/blog/fonds-diaporama/>

- Photo diapositive n°1 :

<http://integration-aveugles.e-monsite.com/pages/temoignage-olivier/difference-des-cannes.html>

- Photos diapositive n°10 :

<https://www.gotronic.fr/art-vibreur-c6070-794.htm>

<https://www.gotronic.fr/art-carte-arduino-uno-12420.htm>

<https://www.amazon.fr/Kit-Transistor-TOOGOO-92-dAssortiment/dp/B01EHQ2PKU>

<https://www.cdiscount.com/bricolage/electricite/50-x-1-4w-250v-10k-ohm-resistances-axiales-de-fi/f-1661416-sod4894560281896.html>

- Photo diapositive n°16 :

<https://www.gotronic.fr/art-module-de-detection-us-hc-sr04-20912.htm>

# Annexes

Distance réelle D (cm)	Angle $\beta$ ( $^{\circ}$ )	Mesure (cm)
Quelconque	$\pm 70$	832 (pas de détection, distance du mur)
Quelconque	$\pm 60$	832
Quelconque	$\pm 40$	832
D < 115	$\pm 35$	832
115	$\pm 35$	114 ↔ 832 (oscillations)
115.5	$\pm 35$	114
120	$\pm 35$	120
125	$\pm 35$	124
126	$\pm 35$	125
127	$\pm 35$	127
128	$\pm 35$	128
129	$\pm 35$	129
130	$\pm 35$	128
132	$\pm 35$	132
134	$\pm 35$	134
136	$\pm 35$	135
137	$\pm 35$	137
137.5	$\pm 35$	137.7 ↔ 832
D > 137.5	$\pm 35$	832
D < 105	$\pm 30$	832
105	$\pm 30$	105
120	$\pm 30$	117
135	$\pm 30$	133
150	$\pm 30$	149
165	$\pm 30$	165
168	$\pm 30$	172
169	$\pm 30$	174
170	$\pm 30$	170 ↔ 832
D > 170	$\pm 30$	832
D < 18	0	832
18	0	18
20	0	20
25	0	25
50	0	49
75	0	74
100	0	99
125	0	122
140	0	140
150	0	146
280	0	280
300	0	295
350	0	347
370	0	367
D > 370	0	832

Pour  $|\beta|>40^{\circ}$ , l'obstacle n'est pas détecté peu importe sa distance : l'angle limite de détection est  $\beta_{\text{lim}} = \pm 35^{\circ}$ .

Pour  $\beta_{\text{lim}}$ , l'obstacle est détecté pour les distances D dans [115.5 ; 137] :  $\Delta D = 21.5\text{cm} \rightarrow$  l'obstacle est détecté pour une faible plage de distances assez éloignées du capteur.

Pour  $\beta = \pm 30^{\circ}$ , l'obstacle est détecté pour les distances D dans [105 ; 169] :  $\Delta D = 64\text{cm} \rightarrow$  l'obstacle est détecté pour des distances plus importantes et moins importantes qu'avec  $\beta_{\text{lim}}$ .

Pour  $\beta = 0^{\circ}$ , l'obstacle est détecté pour les distances D dans [18 ; 370] :  $\Delta D = 352\text{cm} \rightarrow$  l'obstacle est détecté pour des distances très proches du capteur (18cm) mais également pour des distances assez éloignées (3m70).

Entre les deux, plus on se rapproche de  $\beta = 0^{\circ}$ , plus la plage de détection de l'obstacle sera importante.

**Conclusion** : ces résultats imposent donc à l'utilisateur de se tourner avant d'avancer dans une direction, pour éviter les obstacles présents dans les « angles morts ».

# Annexes

## Calcul des distances

- Émission de 8 impulsions ultrasoniques à 40kHz (Trig)
- Réception du signal réfléchi
- Envoi d'un signal « High » (Echo) :  $\Delta t = K * \text{distance} (\mu\text{s})$

$$d = v * t ; v \sim 340 \text{ m/s} ; t = \Delta t * 0,5 \text{ (aller-retour)}$$

$$d_{(\text{cm})} = 34'000_{(\text{cm/s})} * \Delta t_{(\mu\text{s})} * 10^{-6} * 0,5$$

$$\mathbf{d_{(\text{cm})} = 0,017 * \Delta t_{(\mu\text{s})}}$$

```

int trighaut = 2;
int echohaut = 3;
const byte vibreuhaut = 6;

int trigbas = 4;
int echobas= 5;
const byte vibreurbas = 9;

float lecture_echohaut,lecture_echobas;
float cmhaut,cmbas,cm;
int vibration = 0, duree = 0;

void setup()
{
  pinMode(vibreuhaut, OUTPUT);
  pinMode(trighaut, OUTPUT);
  digitalWrite(trighaut, LOW);
  pinMode(echohaut, INPUT);

  pinMode(vibreurbas, OUTPUT);
  pinMode(trigbas, OUTPUT);
  digitalWrite(trigbas, LOW);
  pinMode(echobas, INPUT);

  Serial.begin(9600);
}

```

```

void distancehaut(){
  digitalWrite(trighaut, HIGH);
  delayMicroseconds(10);
  digitalWrite(trighaut,LOW);
  lecture_echohaut = pulseIn(echohaut, HIGH);
  cmhaut = lecture_echohaut*0.017;
}

void distancebas(){
  digitalWrite(trigbas, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigbas,LOW);
  lecture_echobas = pulseIn(echobas, HIGH);
  cmbas = lecture_echobas*0.017;
}

```

# Annexes

```

void loop()
{
  distancehaut();
  distancebas();

  if (cmhaut<150 && cmbas<150){
    vibrations(cmhaut,cmbas);
  }
  else if (cmhaut>=150)
  {
    analogWrite(vibreuhaut,0);
    if (cmbas>=150){
      analogWrite(vibreurbas,0);
    }
    else
    {
      vibrationsseul(cmbas,vibreurbas);;
    }
  }
  else
  {
    analogWrite(vibreurbas,0);
    vibrationsseul(cmhaut,vibreuhaut);
  }
}

```

```

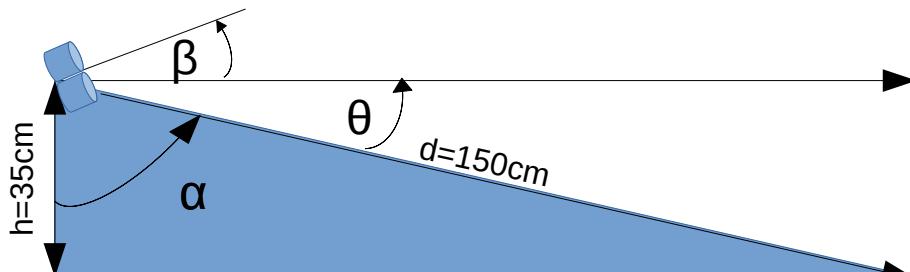
void vibrations(int cmh, int cmb){
  cm=min(cmh,cmb)
  //fonction affine (distance, borneinf, bornesup, vibborneinf, vibbornesup)
  vibrationh = map(cm, 0, 150, 255, 100);
  duree = map(cm,0,150,10,1430);
  analogWrite(vibreuhaut,70);
  analogWrite(vibreurbas,70);
  delay(duree);
  analogWrite(vibreuhaut, vibration);
  analogWrite(vibreurbas, vibration);
  delay(500);
}

void vibrationsseul(int cm,int vibreur){
  vibration = map(cm, 0, 150, 255, 100);
  duree = map(cm,0,150,10,1430);
  analogWrite(vibreur,70);
  delay(duree);
  analogWrite(vibreur, vibration);
  delay(500);
}

```

# Annexes

## ➤ Détection du sol pour le capteur vertical



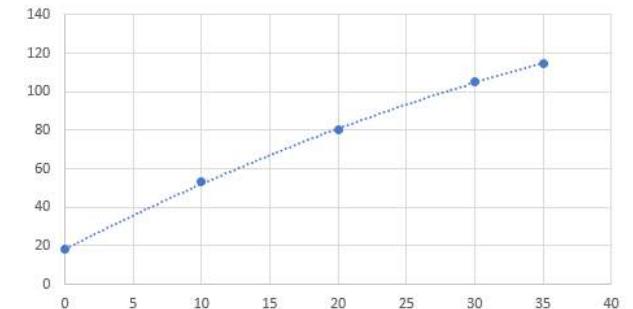
➤ Si  $\beta_{\min}=0$ ,  $\theta=35^\circ \rightarrow h=86\text{cm}$  car  $h=d*\cos(\alpha)=150*\cos(55)$

$\rightarrow d_{\min-\text{erreur}}=85\text{cm}$  car  $d=h/\cos(\alpha)=35/\cos(90-23,17)=88,95\text{cm}$  puisque  $d_{\min}=f(\theta)$  (sol détectable pour  $\theta_{\text{détexion}}>23,17^\circ$ , or  $d_{\min}$  croissante et  $d_{\min}(23,17^\circ)=89,01\text{cm}>88,95\text{cm}$ )

$\rightarrow h=80\text{cm}$  et  $d=135\text{cm}$  :  $\alpha=\arccos(80/139)\approx54,9^\circ \rightarrow \theta_{\text{détexion}}\approx35,01^\circ$

▲ Attention à la précision des capteurs

$$d_{\min}=f(\theta)$$

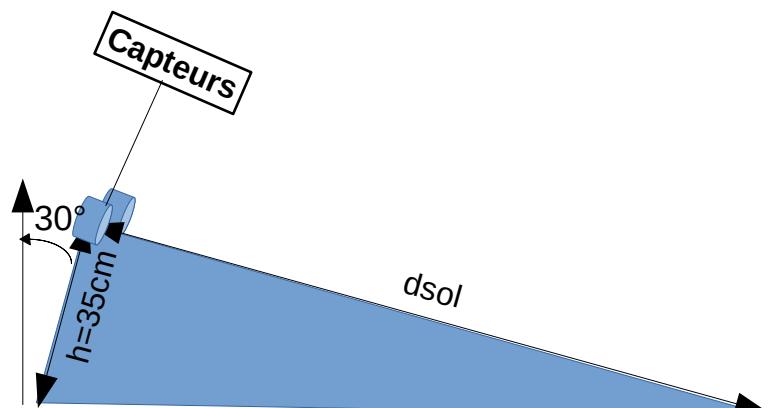


## ➤ Détection du sol pour le capteur horizontal

Angle maximal d'inclinaison de la jambe vers le bas :  $30^\circ$

$$dsol_{\min}=\tan(60)*35=60\text{cm} \rightarrow d_{\text{prog}}=55\text{cm}$$

▲ Attention à la précision des capteurs



➤ Solution choisie : capteur horizontal à 35cm avec  $d_{\text{prog}}=55\text{cm}$  pour obstacles bas, capteur vertical à 80cm avec  $d_{\text{prog}}=135\text{cm}$  pour obstacles éloignés et hauts

# Annexes

```
int trighaut = 2;
int echohaut = 3;
const byte vibreuhaut1 = 6;
const byte vibreuhaut2 = 9;
const byte vibreuhaut3 = 10;

int trigbas = 4;
int echobas = 5;
const byte vibreurbas = 11;

float lecture_echohaut, lecture_echobas;
float cmhaut,cmbas,cm;
float vibration, vibrationh, vibrationb, duree;

void setup()
{
pinMode(vibreuhaut1, OUTPUT);
pinMode(vibreuhaut2, OUTPUT);
pinMode(vibreuhaut3, OUTPUT);
pinMode(trighaut, OUTPUT);
digitalWrite(trighaut, LOW);
pinMode(echohaut, INPUT);

pinMode(vibreurbas, OUTPUT);
pinMode(trigbas, OUTPUT);
digitalWrite(trigbas, LOW);
pinMode(echobas, INPUT);

Serial.begin(9600);
}
```

```
void loop()
{
distancehaut();
distancebas();

if (cmhaut==0 or cmbas==0) {
pinMode(echohaut, OUTPUT);
pinMode(echobas, OUTPUT);
digitalWrite(echohaut, LOW);
digitalWrite(echobas, LOW);
delay(100);
pinMode(echohaut, INPUT);
pinMode(echobas, INPUT);
distancehaut();
distancebas();
}

if (cmhaut<135 && cmbas<55){
vibrations();
}
else if (cmhaut>=135)
{
analogWrite(vibreuhaut1,0);
analogWrite(vibreuhaut2,0);
analogWrite(vibreuhaut3,0);
if (cmbas>=55){
analogWrite(vibreurbas,0);
}
else
{
vibrationsbas();
}
}
else
{
analogWrite(vibreurbas,0);
vibrationshaut();
}
}
```

# Annexes

```
void distancehaut()
{
    digitalWrite(trighaut, HIGH);
    delayMicroseconds(10);
    digitalWrite(trighaut, LOW);
    lecture_echohaut = pulseIn(echohaut, HIGH);
    cmhaut = lecture_echohaut*0.017;
}
```

```
void distancebas()
{
    digitalWrite(trigbas, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigbas, LOW);
    lecture_echobas = pulseIn(echobas, HIGH);
    cmbas = lecture_echobas*0.017;
}
```

```
void vibrations()
{
    vibrationh = map(cmhaut, 0, 135, 255, 100);
    vibrationb = map(cmbas, 0, 135, 255, 100);
    cm = min(cmhaut,cmbas);
    duree = map(cm,0,135,10,1430);
    analogWrite(vibreurhaut1,70);
    analogWrite(vibreurhaut2,70);
    analogWrite(vibreurhaut3,70);
    analogWrite(vibreurbas,70);
    delay(duree);
    analogWrite(vibreurhaut1, vibrationh);
    analogWrite(vibreurhaut2, vibrationh);
    analogWrite(vibreurhaut3, vibrationh);
    analogWrite(vibreurbas, vibrationb);
    delay(500);
}
```

```
void vibrationshaut()
{
    vibrationh = map(cmhaut, 0, 135, 255, 100);
    duree = map(cmhaut,0,135,10,1430);
    analogWrite(vibreurhaut1,70);
    analogWrite(vibreurhaut2,70);
    analogWrite(vibreurhaut3,70);
    delay(duree);
    analogWrite(vibreurhaut1, vibrationh);
    analogWrite(vibreurhaut2, vibrationh);
    analogWrite(vibreurhaut3, vibrationh);
    delay(500);
}
```

```
void vibrationsbas()
{
    vibrationb = map(cmbas, 0, 135, 255, 100);
    duree = map(cmbas,0,135,10,1430);
    analogWrite(vibreurbas,70);
    delay(duree);
    analogWrite(vibreurbas, vibrationb);
    delay(500);
}
```