Capteur à ultrasons - Arduino

1. Objectifs :

- Mesurer la position en fonction du temps d'un objet à l'aide d'un capteur ultrasons.

2. Matériel :

- Arduino UNO ou l'équivalent
- Ordinateur + fil de branchement pour la carte
- 4 fils
- Capteur de distance à ultrasons HR-S04 avec support
- Petit bolide ou objet assez gros pour en mesurer la position en fonction du temps
-

3. À quoi sert ce projet

Ce projet sert à s'initier à l'utilisation du capteur à ultrasons avec une carte Arduino pour mesurer la distance entre le capteur et un objet situé devant lui.

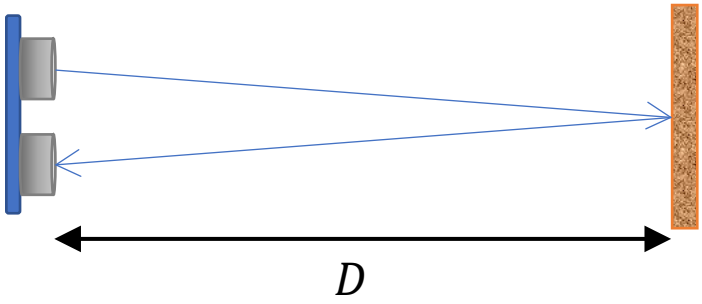
4. Fonctionnement du capteur ultrasonique

Le capteur utilise des ultrasons pour déterminer la distance des objets, un peu à la manière des chauves-souris. Les ultrasons sont des ondes sonores qui ont une fréquence plus élevée (plus aiguës) que ce que l'oreille humaine peut percevoir.

Ces ondes voyages à vitesse constante à la vitesse du son. Celle-ci, à la température de la pièce, est approximativement de 340 m/s. Le son va parcourir une distance Δx en un intervalle de temps Δt selon l'équation :

$$\Delta x = v \cdot \Delta t = 340 \cdot \Delta t$$

Le capteur fonctionne en envoyant un signal qui se réfléchit sur une surface en face de lui et revient vers le capteur. Le capteur envoie à la carte Arduino la valeur de l'intervalle de temps entre l'envoi et la réception su signal.



La distance parcourue (en mètres) par le son est donc (on peut négliger l'angle et considérer que l'onde fait un aller-retour sur elle-même) :

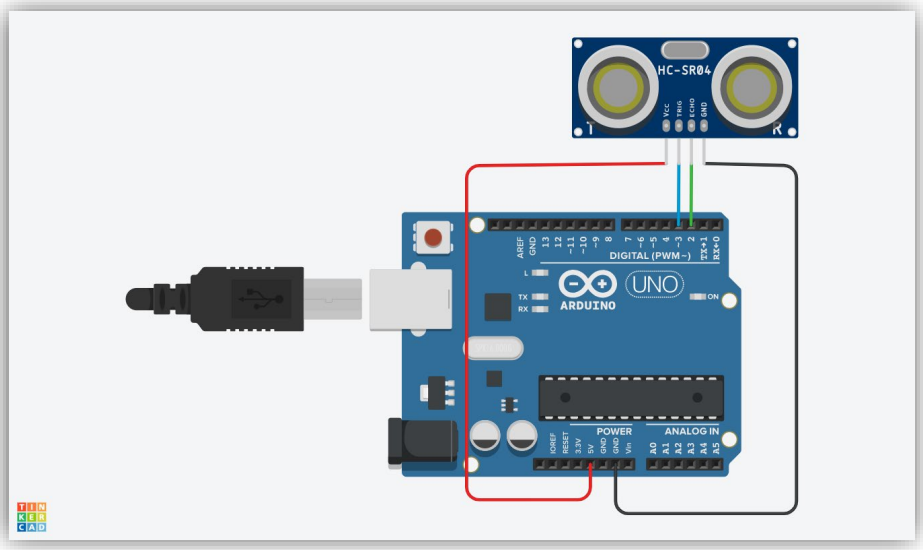
$$\Delta x = 2D$$

et

$$D = \frac{340 \cdot \Delta t}{2}$$

5. Étapes de réalisation

1. Réaliser le circuit suivant :



2. Brancher la carte Arduino UNO dans l'ordinateur avec le fil approprié. La DEL (verte) d'alimentation de la carte devrait être allumée.
3. Ouvrir l'application Arduino IDE.
4. Copier le code suivant et le téléverser dans la carte Arduino.
5. Ouvrir le moniteur série.
6. Modifier la position de l'objet devant le capteur ultrason et observer les résultats sur le moniteur série.

6. Code

```
const int trigPin = 3;
const int echoPin = 2;
long duree;
int distance;

void setup() {
      pinMode(trigPin, OUTPUT);
      pinMode(echoPin, INPUT);
      Serial.begin(9600);
}
void loop() {
      distance = calculeDistance();
      Serial.println(distance);
      delay(10);
}

int calculeDistance(){

      digitalWrite(trigPin, LOW);
      delayMicroseconds(2);
      digitalWrite(trigPin, HIGH);
      delayMicroseconds(10);
      digitalWrite(trigPin, LOW);
      duree = pulseIn(echoPin, HIGH);
      distance= duree*0.034/2;  //calcul de la distance en cm
      return distance;
}
```

Fonctions utilisées

La fonction calculeDistance

Lorsque cette fonction est appelée, dans la boucle *loop()*, plusieurs choses vont se passer. Tout d'abord, notons que la première ligne de la définition de la fonction, soit :

```
int calculateDistance()
```

nous indique, tout d'abord, que la fonction va retourner un nombre entier (*int*). C'est pourquoi, plus haut, on a pu écrire :

```
distance = calculateDistance();
```

La variable distance sera alors égale à la valeur retournée lors de la ligne :

```
return distance;
```

que l'on retrouve à la fin de la fonction.

Pour les autres lignes de code que l'on retrouve dans la fonction :

```
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
```

Cette série de commandes permet d'envoyer un signal à partir de la broche TRIG. Tout d'abord, on met la broche à LOW pendant 2 microsecondes. Ensuite, on la met à HIGH pendant 10 microsecondes et on remet sa valeur à LOW. On envoie donc une sorte de pic carré à la broche, qui émettra des ultrasons. En fait, 8 impulsions d'une fréquence de 40 000 Hz seront envoyées. La broche echoPin sera alors automatiquement mise à HIGH par le module. Cette broche tombera ensuite à une valeur LOW lorsque le capteur recevra l'écho des impulsions ultrasoniques envoyées.

La fonction delayMicrosecondes() est une fonction similaire à la fonction delay() plus couramment utilisée. Comme son nom l'indique, le temps que l'on met entre parenthèses est en microsecondes ($1 \times 10^{-6} s$) au lieu d'être en millisecondes (0,001 s).

```
duree = pulseIn(echoPin, HIGH);
```

La valeur de *duree*, ici, est le nombre de microsecondes entre le moment où la broche echoPin a été mise automatiquement à HIGH par le module (tout de suite après l'envoi des impulsions ultrasoniques) et le moment où elle retombe à une valeur LOW lorsque le capteur recevra l'écho des impulsions ultrasoniques envoyées. La fonction pulseIn calcule le temps entre le moment où la broche passe de LOW à HIGH (envoi de l'onde) et où elle repasse de HIGH à LOW (réception de l'écho).

```
distance= duree*0.034/2;  //calcul de la distance en cm
```

On calcule ici la distance en cm (le son se propage à 340 m/s, ce qui correspond à 0,034 cm/microseconde) à partir de la vitesse du son et de la durée du trajet de l'onde ultrasonique. On doit diviser la valeur par deux puisque l'onde fait l'aller-retour entre le capteur et l'objet qui est devant lui!

Défis :

- Tester différents objets et différentes distances pour voir les capacités et les limitations du capteur (dimensions minimales et maximales de l'objet et en fonction de la distance, distances maximales et minimales mesurées,…)
- Remplacer le moniteur série par le traceur série pour voir la différence entre les deux
- Ajouter une variable chronomètre et afficher la valeur du temps en plus de la distance. Y a-t-il toujours le même intervalle de temps entre deux mesures pour toutes les distances?
- Ajouter le contrôle de DEL dont l'éclairage est modifié en fonction de la distance, selon ce que l'on souhaite