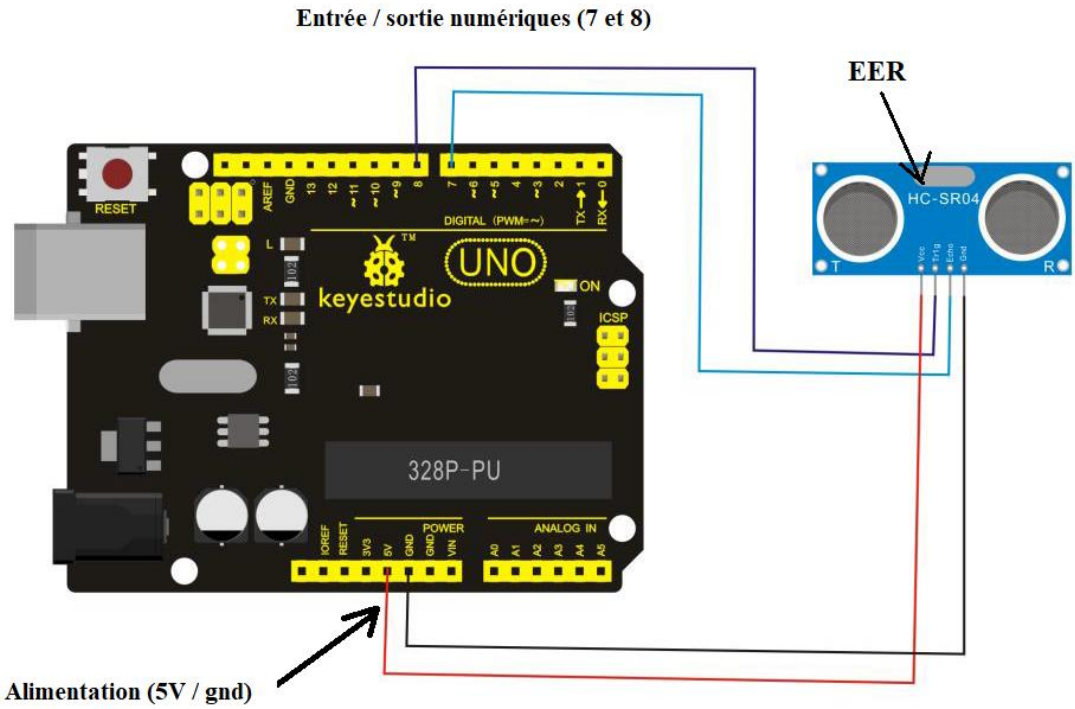
TP ultrason : partie 3 avec aide et explications détaillées…
3ème partie : principe d'une échographie

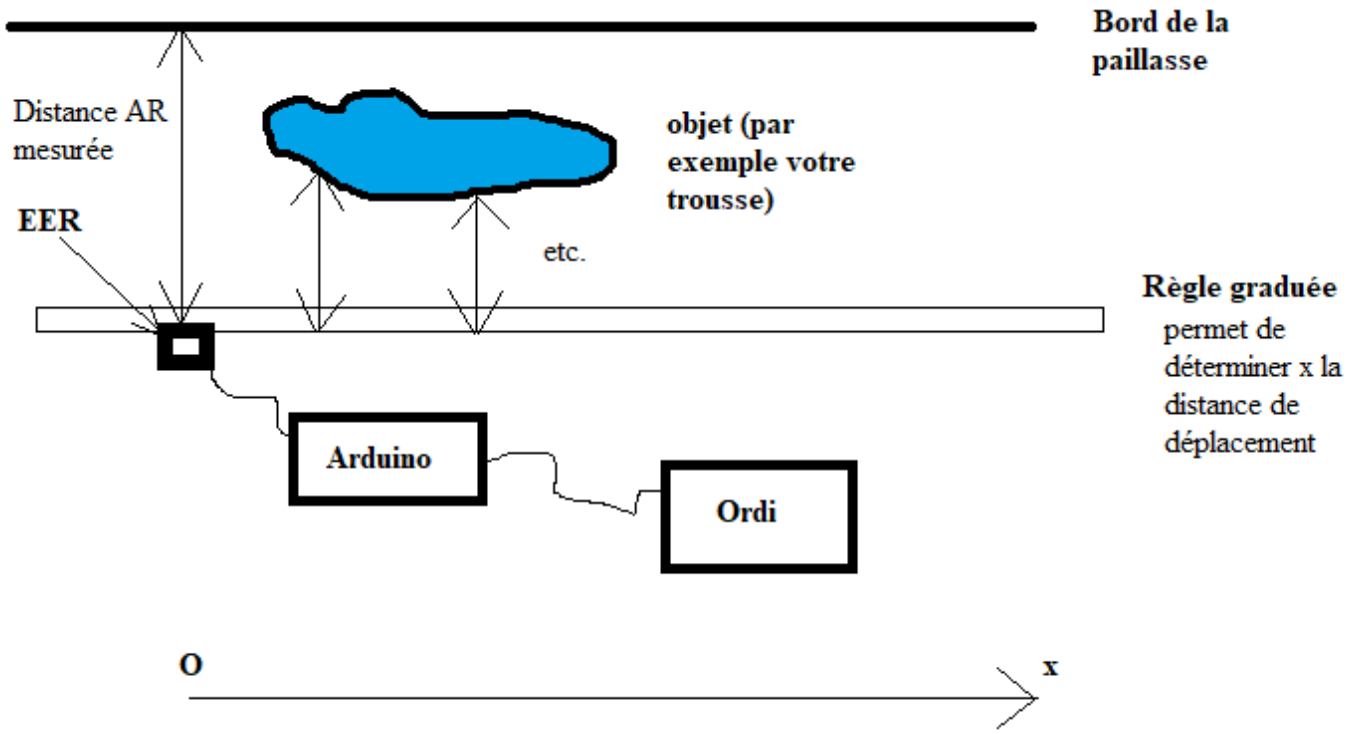
Le dispositif expérimental :

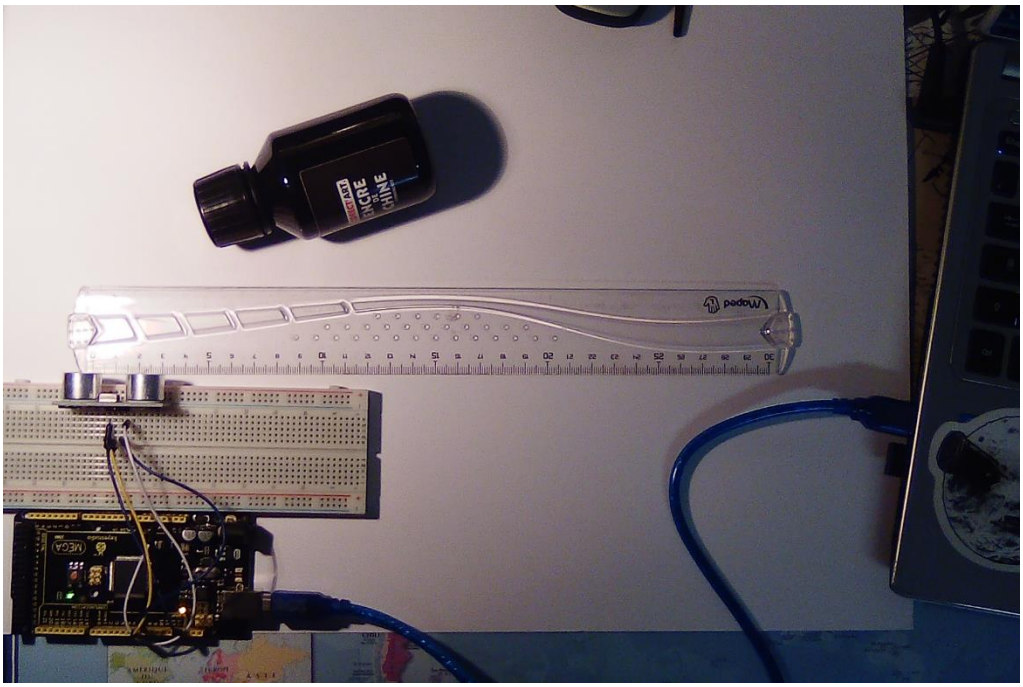


Objectif :

Disposer un objet de forme quelconque à une vingtaine de centimètres de l'ensemble émetteur/récepteur U.S. L'émetteur est noté « E », le récepteur est noté « R » et l'ensemble est noté « EER ».
Déplacer progressivement l'EER dans la direction de votre choix tout en envoyant automatiquement des salves ultrasonores (E) qui sont réfléchies par l'objet et détectées (R). Récupérer les données de mesure et présenter un graphe qui restitue de manière cohérente la forme de l'objet.

Schéma :





(la bouteille d'encre couchée, c'est l'objet)

Le programme informatique (le code) qui permet d'envoyer automatiquement des salves d'U.S. et de mesurer la durée de leur aller-retour entre E et R est donné à la fin du document. Il peut être modifié (nombre de mesures, intervalle de temps entre deux mesures successives, etc.).

L'aide qui suit doit être lue avec le code (le programme en langage « arduino ») fourni à la fin du document sous les yeux…
Bien entendu le code (le programme qui est à la dernière page) ne se contente pas de générer l'enregistrement de durées d'allers-retours de l'onde ultra-sonore, il va plus loin :

- **Une formule contenant la célérité des U.S. (340 $m.s^{-1}$) permet de calculer la distance à laquelle l'obstacle se trouve du dispositif émetteur/récepteur d'U.S. (noté « EER » on le rappelle).**

- **Une instruction désigne une pause de 5 s entre chaque mesure (par défaut l'unité de temps dans arduino est la ms, vous trouvez donc une instruction « delay 5000 »…**

- **Pendant le délai de 5 s (que l'on peut rallonger si on n'a pas le temps de réaliser les modifications envisagées), on déplace l'EER d'une distance précise, par exemple 1 cm, selon la direction choisie (parallèlement à la paillasse semble être le choix le plus raisonnable).**

- **Un compteur permet de réaliser seulement 20 mesures (on peut changer cette valeur si on le souhaite).**

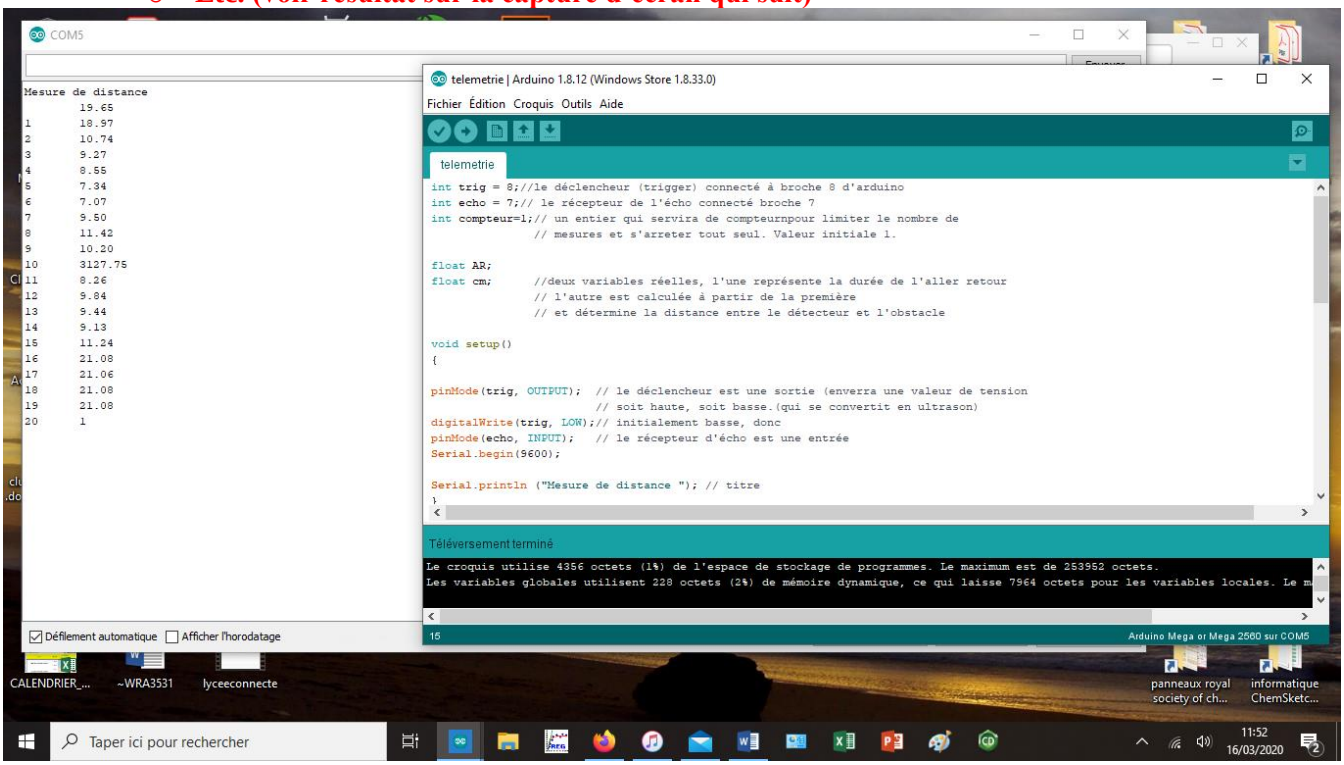
- **On récupère quand c'est terminé un ensemble lignes/colonnes de valeur sur le moniteur qui a été ouvert (voir image 2), on le copie et on le colle sur excel ou regressi (image 3). *J'ai eu un problème à ce niveau, voir plus loin…***

- **C'est là que la consigne est ouverte et qu'il faut avoir l'idée suivante : nous allons pouvoir tracer un graphe distance = f(temps), mais nous savons qu'à chaque durée supplémentaire de 5000 ms nous pouvons associer un déplacement supplémentaire de l'EER de 1 cm. Nous pouvons donc créer une grandeur fictive, notée x, par exemple avec regressi, qui transforme le temps écoulé en distance (en cm). Comme le délai entre deux mesures (ici 5000 en ms) correspond à un déplacement de 1 cm, comme le temps écoulé est noté t, on crée x = $\frac{t}{5000}$.**

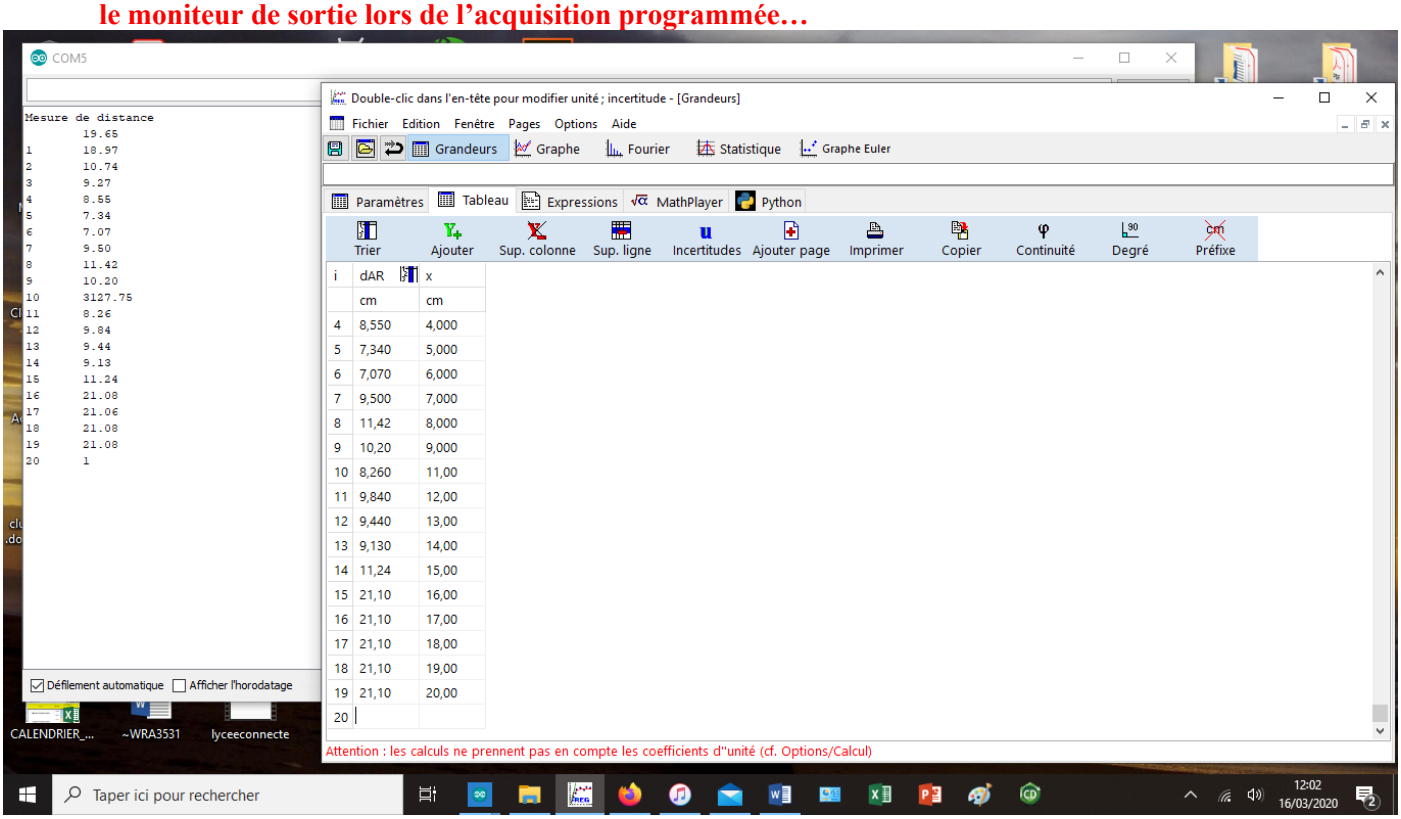
- **Mais : pourquoi ne pas intégrer cette création de grandeur x directement dans le code Arduino et récupérer directement les données sous la forme d'un graphe distance aller-retour = f(x) ?**
-

En fait je me suis servi du compteur !

- **Lorsque la première mesure se fait le compteur est à zéro… et x = 0**
- **Lors de la deuxième mesure le compteur affiche 1 et la distance mesurée ; x = 1 cm**
- **Lors de la troisième mesure le compteur affiche 2 et la distance mesurée ; x = 2 cm**
- **Etc. (voir résultat sur la capture d'écran qui suit)**



- **J'avoue j'ai eu un problème et j'ai copié à la main, sur un fichier regressi, les données récupérées dans le moniteur de sortie lors de l'acquisition programmée…**



- **On peut tracer le graphe d = f(x) C'est ce graphe qui illustrera la forme de l'objet testé.**

Je n'ose pas vous montrer le résultat qui est très mauvais… Sans doute parce que :

- **Mon EER est trop près de mon objet**
- **Mon objet est un peu compliqué (il va parfois réfléchir l'onde dans une direction inadaptée et le résultat pour la distance AR sera absurde).**

Il faut accepter que nous n'en sommes qu'à un dispositif très sommaire et qu'il faut se confronter à un objet très simple, une boite rectangulaire par exemple…

- **Si nous avions en plus la possibilité de faire se déplacer l'EER à vitesse constante et connue (sur un rail avec un dispositif motorisé), nous pourrions réaliser des mesures beaucoup plus souvent et obtenir beaucoup plus de données traduisant avec beaucoup plus de précision la forme de l'objet.**

Les instructions d'utilisation du logiciel Arduino sont proposées pendant la séance (ou bien ont donné lieu à une présentation préliminaire).

Si vous n'avez pas commandé de kit arduino au père Noël, vous êtes condamnés à imaginer ce que l'on pouvait mettre en œuvre avec les quelques croquis disponibles…

Je choisis de ne pas vous décrire l'utilisation du micro-contrôleur et du logiciel arduino, il faudrait vraiment être en situation de TP. De nombreux sites, de très nombreux tutoriels vidéo sont disponibles sur ce sujet.

Le code est à la page suivante…

Le code ci-dessous peut-être copié/collé dans une fenêtre du logiciel Arduino préalablement ouvert :

```
int trig = 8;//le déclencheur (trigger) connecté à broche 8 d'arduino
int echo = 7;// le récepteur de l'écho connecté broche 7
int compteur=1;// un entier qui servira de compteurnpour limiter le nombre de
        // mesures et s'arreter tout seul. Valeur initiale 1.

float AR;
float cm;      //deux variables réelles, l'une représente la durée de l'aller retour
        // l'autre est calculée à partir de la première
        // et détermine la distance entre le détecteur et l'obstacle

void setup()
{

pinMode(trig, OUTPUT);  // le déclencheur est une sortie (enverra une valeur de tension
                // soit haute, soit basse.(qui se convertit en ultrason)
digitalWrite(trig, LOW);// initialement basse, donc
pinMode(echo, INPUT);   // le récepteur d'écho est une entrée
Serial.begin(9600);

Serial.println ("Mesure de distance "); // titre
}

void loop(){


digitalWrite(trig, HIGH);  // Le déclencheur a envoyé une impulsion de tension
delayMicroseconds(10);   // pendant 10 microsecondes
digitalWrite(trig, LOW); // il faut que la durée de l'impulsion soit négligeable par rapport à
                // la durée de l'aller retour des US (pour AR 4 cm dans l'air, durée prévue : 120 microsecondes)
                // voir pourquoi dans la description qui suit de l'instruction Pulsein
                // Ou alors on retranche 10 microsecondes de la formule du temps

AR = pulseIn(echo,HIGH);    // ce pourrait être "AR = pulseIn(echo,LOW)-10", voir ci-dessous
 //PulseIn : lit la durée d'une impulsion (soit niveau HAUT, soit niveau BAS) appliquée sur une broche (configurée en ENTREE).
 //Par exemple, si le paramètre valeur est HAUT, l'instruction pulseIn() attend que la broche émettrice (trig) passe à HAUT,
 //commence alors le chronométrage, attend que la broche réceptrice (echo)passe (ici) au niveau HAUT et stoppe alors le chronométrage.
// si c'était un déclenchement au niveau bas (réception fin de la salve) on pourrait retrancher la durée de l'impulsion envoyée
cm = AR*0.0170; //  d = vt... comme "AR" est la durée de l'AR, il y a une division par 2 à effectuer.
        // la vitesse du son (340 m/s) est ici en cm/microseconde : 0,034. Divisé par 2 : 0,017

// Remarque: si l'obstacle est près, la distance est "oblique" et un peu plus grande que perpendiculairement
// à l'obstacle et au couple trig/echo que l'on a positionnés parallèlement.
// on peut rajouter 0,5 cm (capteurs légèrement à l'intérieur des émetteur/récepteur US)


Serial.print("\t");
Serial.print(cm);
Serial.println("\t");
Serial.print(compteur);// la serie d'instructins "print" : pour affichage en lignes et colonnes exportables (excel, ...)
if(compteur>20)exit(0);  // Pour s'arrêter après 10 mesures
compteur=compteur+1;

delay(5000);            // Durée (par défaut en ms) entre chaque mesure
}
```