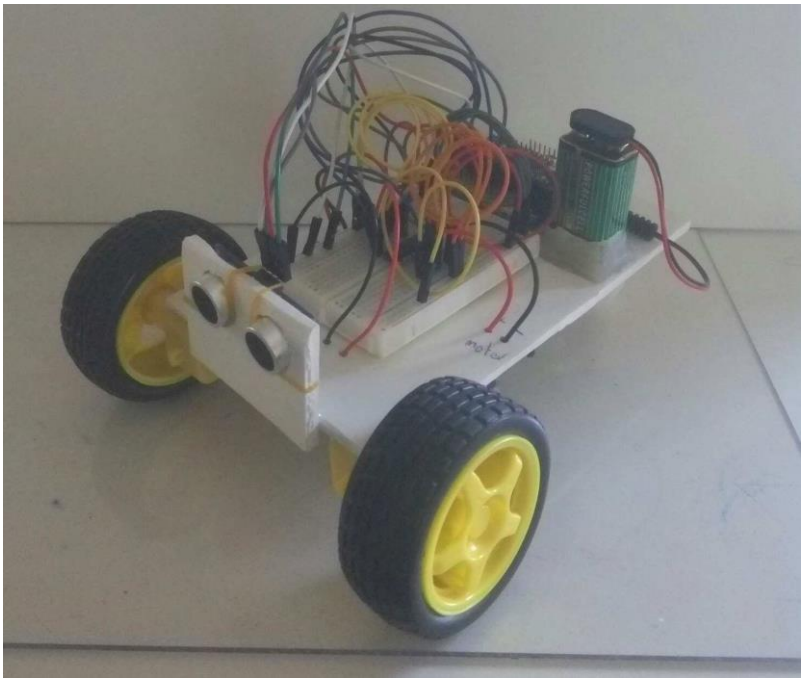
Projet Internet des Objets
Voiture Anti-Collision

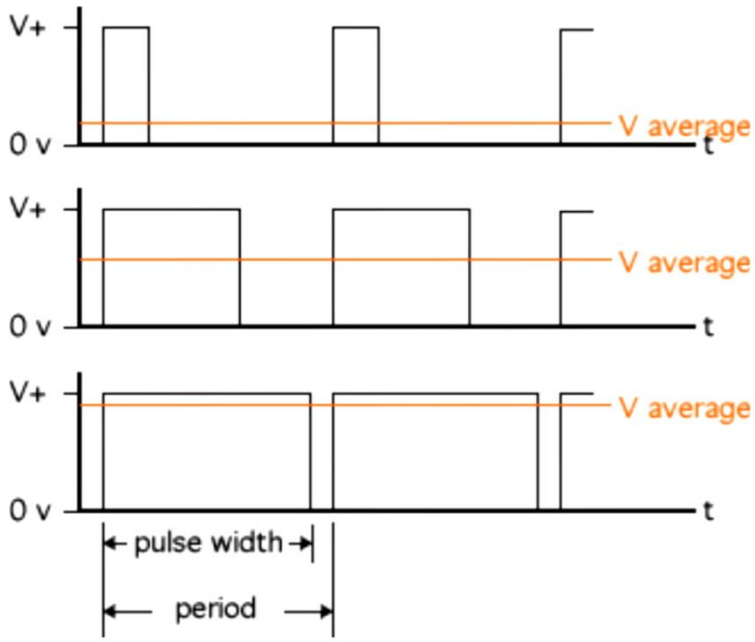


Introduction

Le but de ce projet est de réaliser un premier robot radar qui va être capable de cartographier une zone. De plus le radar doit pouvoir se déplacer. En effet, si le radar peut se déplacer alors il pourra recouvrir une plus grande surface. Pour le véhicule, je voulais réaliser un drone car il s'agit du meilleur moyen de déplacement. Construire un drone de A à Z représente un vrai défi. Malheureusement, cette idée a été abandonnée car le transducteur consomme bien trop d'énergie donc le drone ne pourra pas décoller. De plus pour contrôler les moteurs du drone il est nécessaire d'avoir un contrôleur de vol. Comme je ne possède pas de contrôleur de vol il est impossible de construire un drone. Je décide de créer un véhicule qui pourra se déplacer dans un premier et esquiver les obstacles qui se présentent devant lui.

P.V.M (Pulse Width Modulation)

La P.V.M ou Modulation par Largeur d'Impulsions(M.L.I) en français permet de modifier l'intensité du courant en sortie. Pour se faire, on alterne rapidement deux états et en modifiant leur durée on arrive à avoir un état intermédiaire.

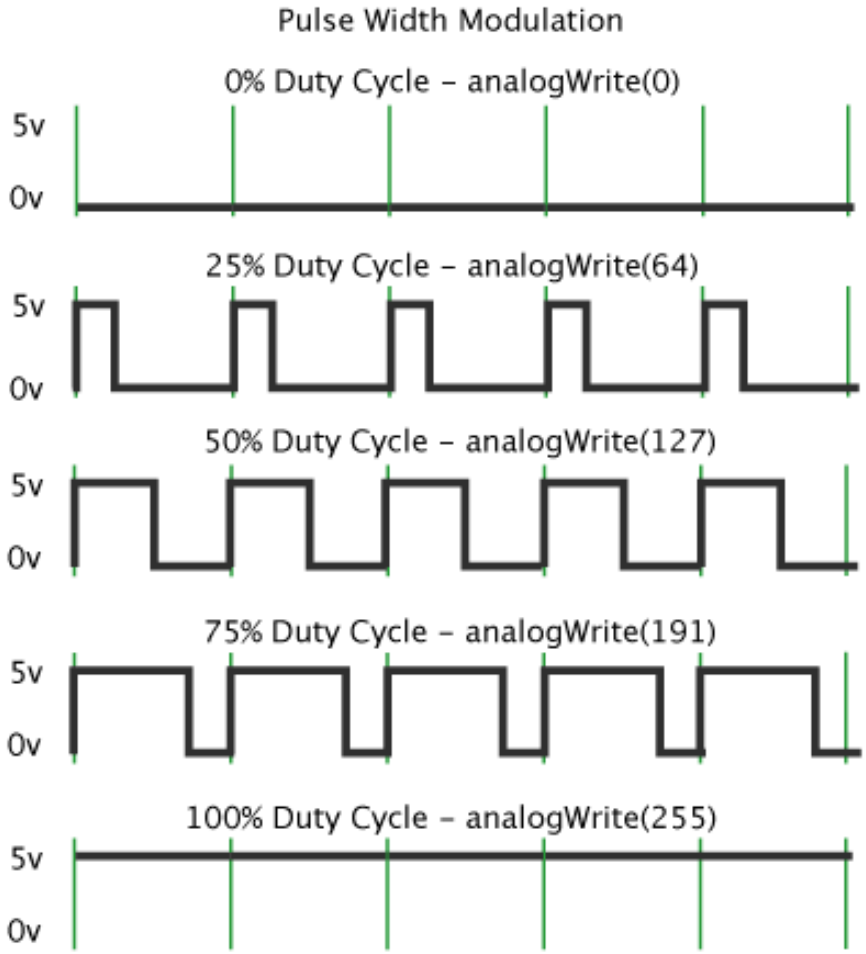


On a alors le rapport cyclique qui est :

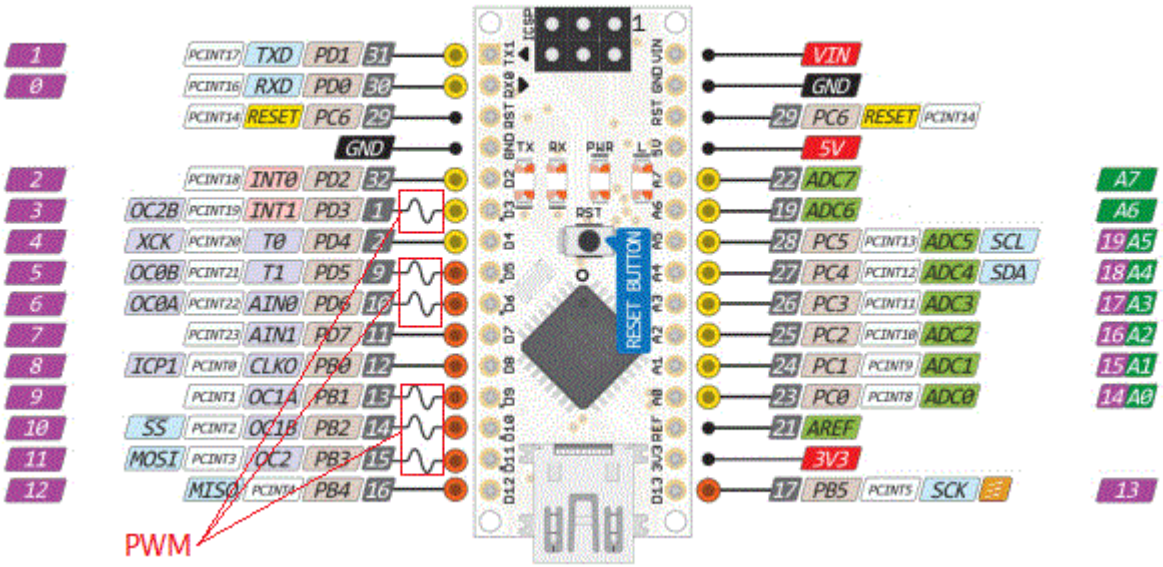
$$Rapport\ Cyclique = \frac{Durée\ de\ l'état\ haut}{Période\ du\ cycle}$$

La fréquence du cycle doit être très élevée car l'œil humain fonctionne comme un filtre passe bas c'est-à-dire que l'on peut voir les basses fréquences, inférieur à 30Hz et que les fréquences supérieures à 30Hz ne sont pas visibles par l'œil humain. Donc pour que cette illusion fonctionne il faut une fréquence de cycle largement supérieur à 30Hz.

Pour créer cette fréquence, la carte Arduino va alterner la tension au bord d'une de ses broches entre 0V et 5V. On aura donc une intensité proportionnelle au rapport cyclique. La fréquence de la PWM d'une carte Arduino est d'environ 500Hz avec un rapport cyclique sur 8 bits car nous pouvons écrire sur une sortie au maximum la valeur 255. Nous avons alors 256 possibilités (de 0 à 255) de rapport cyclique.



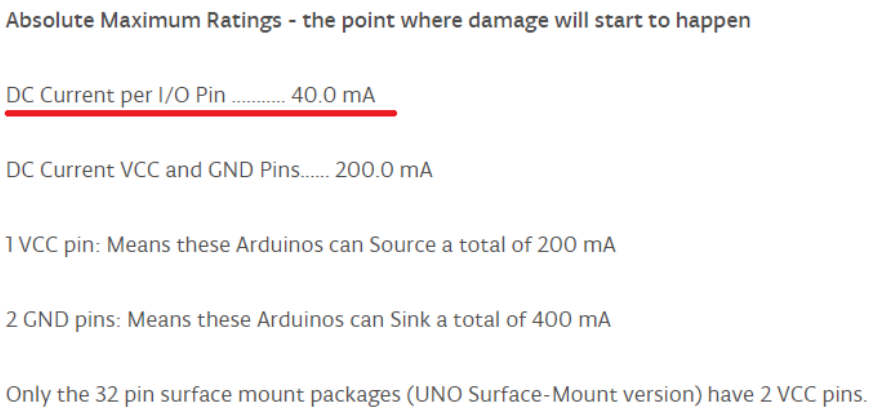
Les broches PWM se reconnaissent sur la carte car à côté du numéro il y a un tilde "~". On peut aussi lire la datasheet de la carte Arduino. Les broches PVM sur les datasheet sont représentées généralement par une petite ondulation au niveau des broches.



On utilise la PWM pour contrôler des moteurs ou l'intensité des diodes. Dans notre cas, on utilisera une sortie PWM pour notre capteur à ultrason.

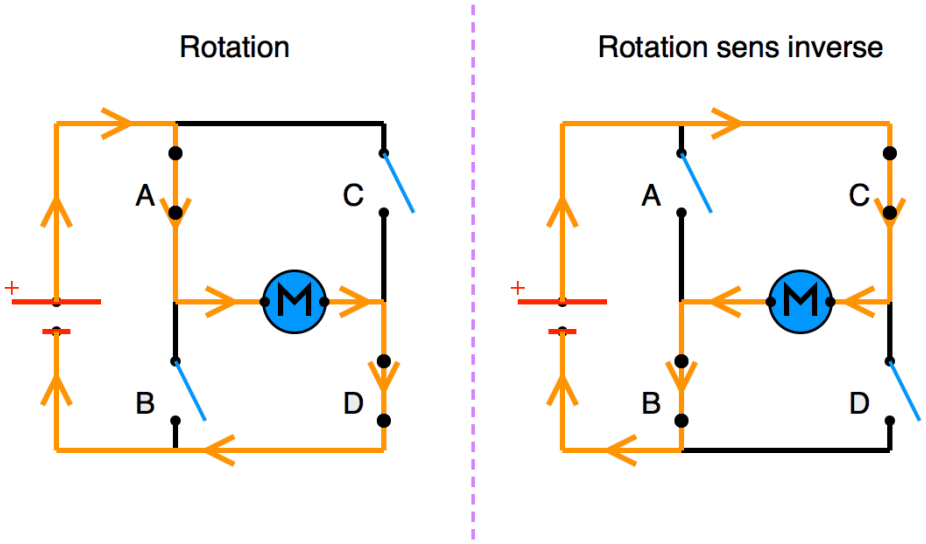
Moteur à courant continu et Pont en H

Un moteur à courant continu est un composant de puissance c'est-à-dire qu'il faut de la puissance pour pouvoir entraîner la bobine de cuivre qu'il y a dans le moteur. Sachant que la sortie d'une carte Arduino ne délivre que 40 mA cela est insuffisant pour faire tourner le moteur.



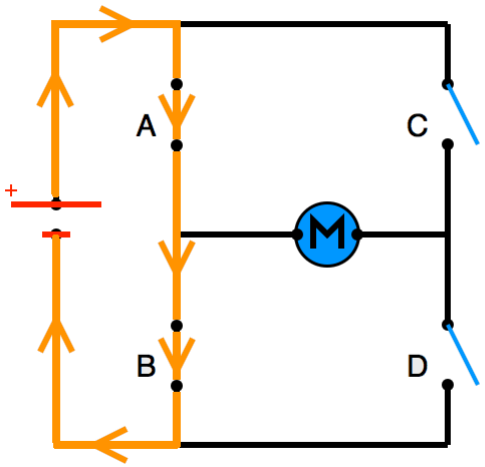
De plus nous rencontrons un autre problème. Si l'on veut changer le sens de rotation de notre moteur, nous devons le faire manuellement. On doit inverser les broches VCC et GND de notre moteur. Notre véhicule possède deux moteurs ce qui rend la tâche plus compliquée.

Pour résoudre ces problèmes nous allons utiliser un pont en H. Un pont en H est une structure en électronique qui regroupe 4 interrupteurs, généralement des transistors de puissance. Grace au pont en H et aux différentes combinaisons des interrupteurs, nous pouvons contrôler la polarité du courant.



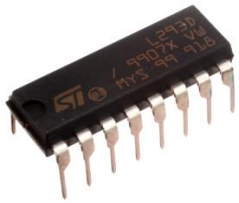
Comme on peut le voir sur le schéma ci-dessus, dans la première partie, l'interrupteur A et D ont été activé. Le moteur trouve dans le sens horaire. Dans la seconde partie, l'interrupteur C et B ont été activés et le moteur tourne dans le sens anti-horaire.

Les ponts en H sont utilisés avec un driver pour faciliter leurs utilisations. Mais aussi pour éviter de faire des courts-circuits.

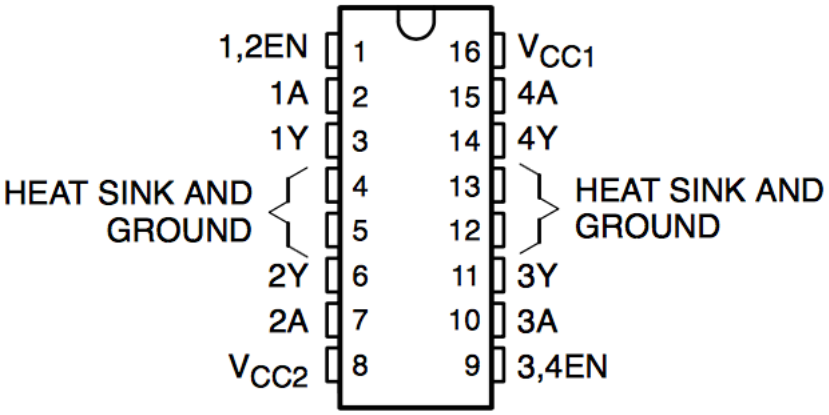


Si l'alimentation provient d'une pile alors elle se déchargera extrêmement vite. Et si l'alimentation provient d'une carte Arduino alors la carte peut subir des dégradations aux niveaux de ses composants voire de ne plus fonctionner.

Il existe un composant qui regroupe toutes ces fonctionnalités c'est-à-dire le driver le pont et des transistors de puissance. Ce composant est le L293D.

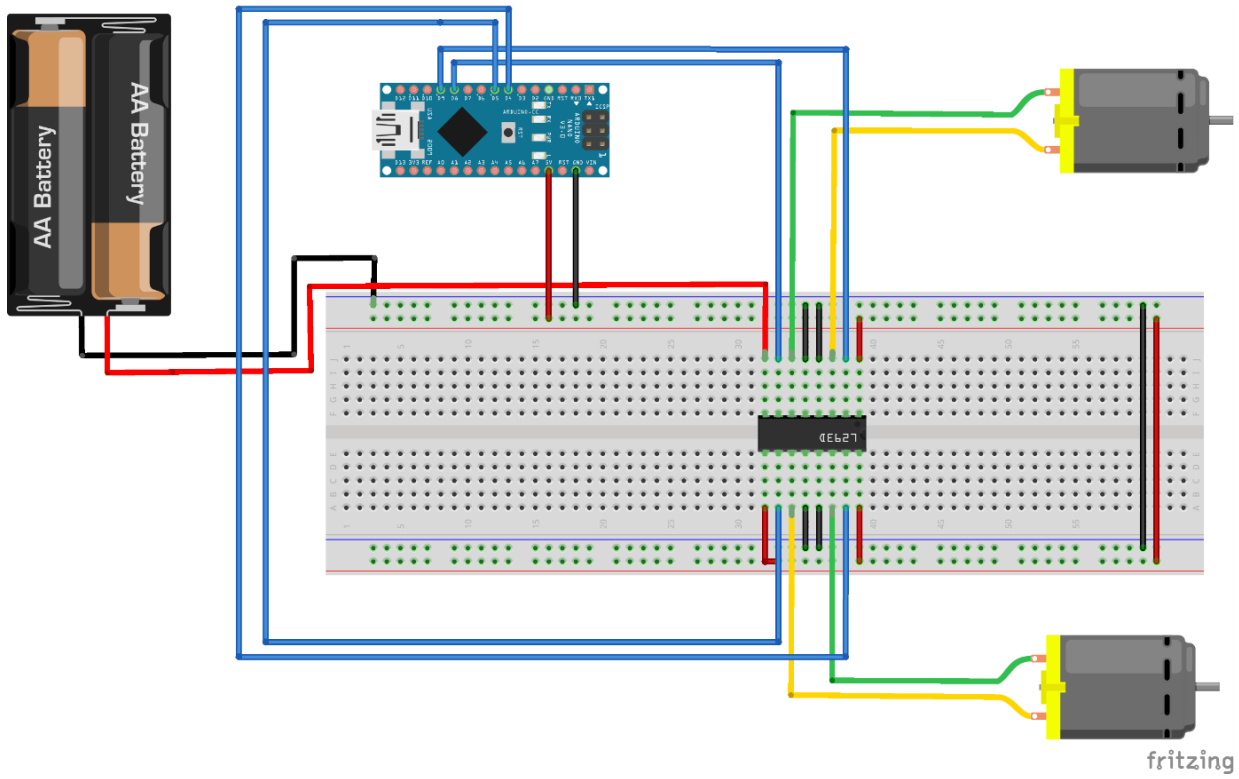


Nous allons nous intéresser aux différentes broches du L293D. Etudions le schéma du L293D.



Il faut placer l'encoche vers le haut car elle permette de se repérer et de donner un sens au composant.
Les broches 4, 5, 13 et 12 correspondent à la masse des différentes composantes du L293D. Il suffit de les relier à la masse. Il est conseillé de les relier si l'on manque de fils ou pour des raisons d'encombrement.
Les broches 3 et 6 sont les broches qui vont alimenter notre moteur gauche.
Les broches 14 et 11 sont les broches qui vont alimenter notre moteur droit.
Les broches 2 et 7 vont permettre de donner le sens au moteur gauche. L'une de ses deux broches doit être à l'état haut ainsi elle doit recevoir une tension de 5V et l'autre à l'état bas soit une tension de 0V. Cela va déterminer le sens de rotation du moteur.
Le principe s'applique aussi pour les broches 15 et 10 sauf que cela permet de contrôler le sens de rotation du moteur droit.
La broche 1 permet d'activer le premier pont donc toute la partie gauche.
La broche 9 permet d'activer le premier pont donc toute la partie droite.
La broche 16 permet d'alimenter le L293D. On appliquera une tension de 5V à celui-ci.
La broche 8 permet d'alimenter nos moteurs. Ici, l'alimentation de cette broche ne doit pas être la même que celle qui alimente notre L293D car cela ne fonctionnera pas. C'est pourquoi on utilisera un bloc de 4 piles. Il ne faut pas oublier de relier la masse de ce bloc de pile à la masse générale.

Après avoir pris en compte les différents détails concernant les broches, nous pouvons passer au schéma. Voici le schéma proposé avec le composant L293D.



Nous utiliserons les broches 9 et 8 pour contrôler le moteur gauche et les broches 5 et 4 pour le moteur de droite. Une alimentation externe (bloc de pile) est utilisée pour alimenter la partie puissance du L293D.

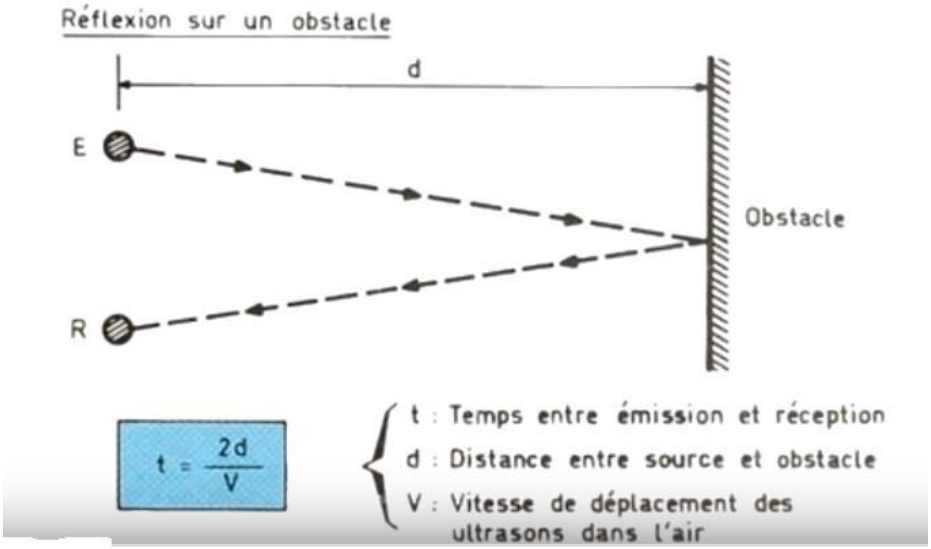
Voici le code associé à ce câblage.

```
/*Declaration variable*/
int devantGauche = 9;                    // Broche 9
int derriereGauche = 8;                  // Broche 8
int devantDroite = 5;                    // Broche 5
int derriereDroite = 4;                  // Broche 4

/*Configuration*/
void setup(){
  pinMode(devantGauche, OUTPUT);         // Mode Sortie
  pinMode(derriereGauche, OUTPUT);       // Mode Sortie
  pinMode(devantDroite, OUTPUT);         // Mode Sortie
  pinMode(derriereDroite, OUTPUT);       // Mode Sortie
}

/*Boucle infini*/
void loop(){
  digitalWrite(devantGauche, LOW);       // Envoie 0V sur la Broche 9
  digitalWrite(derriereGauche, HIGH);    // Envoie 5V sur la Broche 8
  digitalWrite(devantDroite, LOW);       // Envoie 0V sur la Broche 5
  digitalWrite(derriereDroite, HIGH);    // Envoie 5V sur la Broche 4
  delay(10);                             // Pause 10 millisecondes
}
```

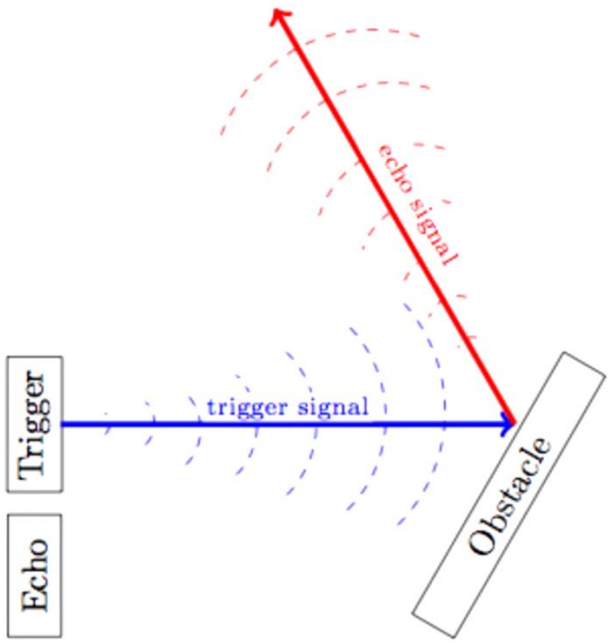
Capteur à ultrason (Transducteur)

Un capteur à ultrason est un capteur capable d'émettre et de recevoir des ondes. Il possède alors un émetteur et un récepteur d'ondes.

Pour calculer la distance entre le capteur à ultrason et l'obstacle il faut connaitre la vitesse de propagation du son et calculer l'intervalle de temps entre l'émission de l'onde créé par l'émetteur et la réception de l'onde par le récepteur.

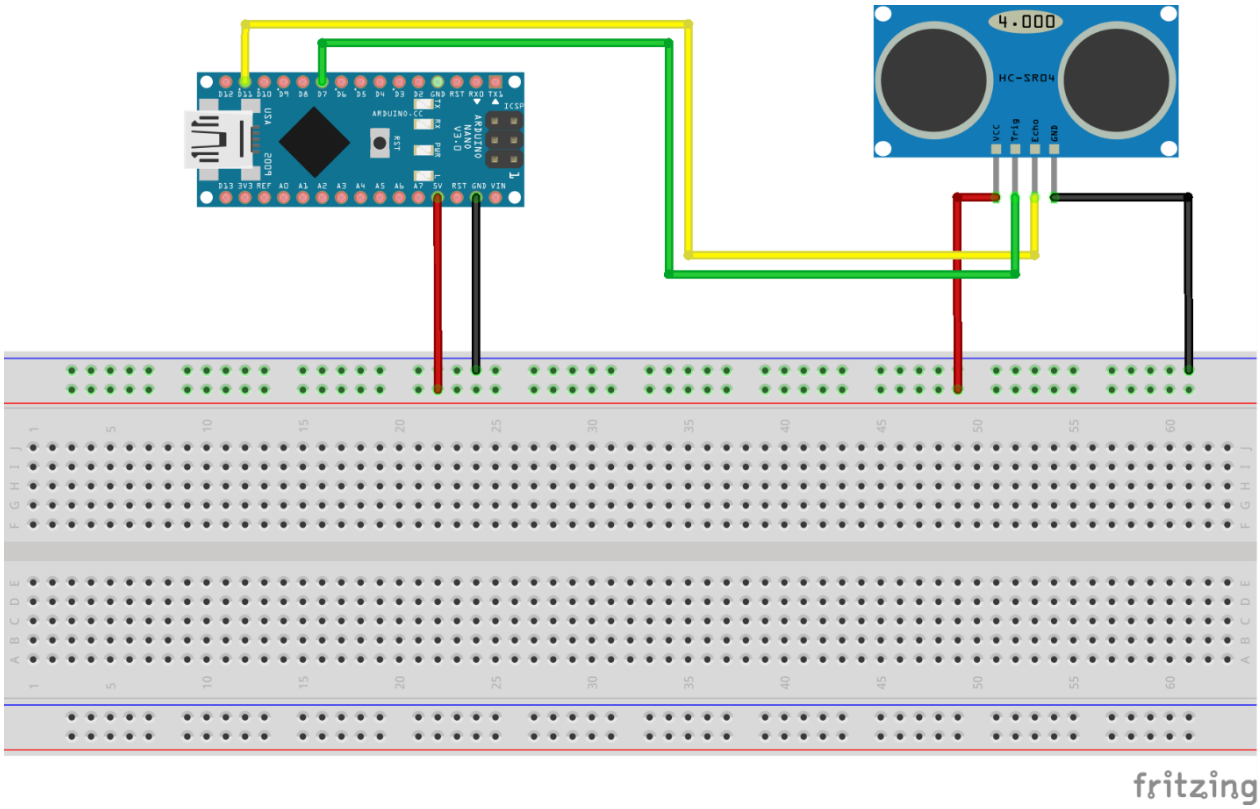


Plus le délai est court plus l'obstacle est proche du capteur.
Cependant il y a des cas ou l'onde peut être perdue. Comme on peut le voir sur le schéma qui ci-dessous, si l'obstacle a une inclinaison, l'onde sera dévier et le récepteur ne pourra pas réception l'onde émise par l'émetteur.



Nous allons utiliser un transducteur pour détecter s'il y a un obstacle devant le véhicule.

Voici un schéma de câblage pour utiliser un traducteur.



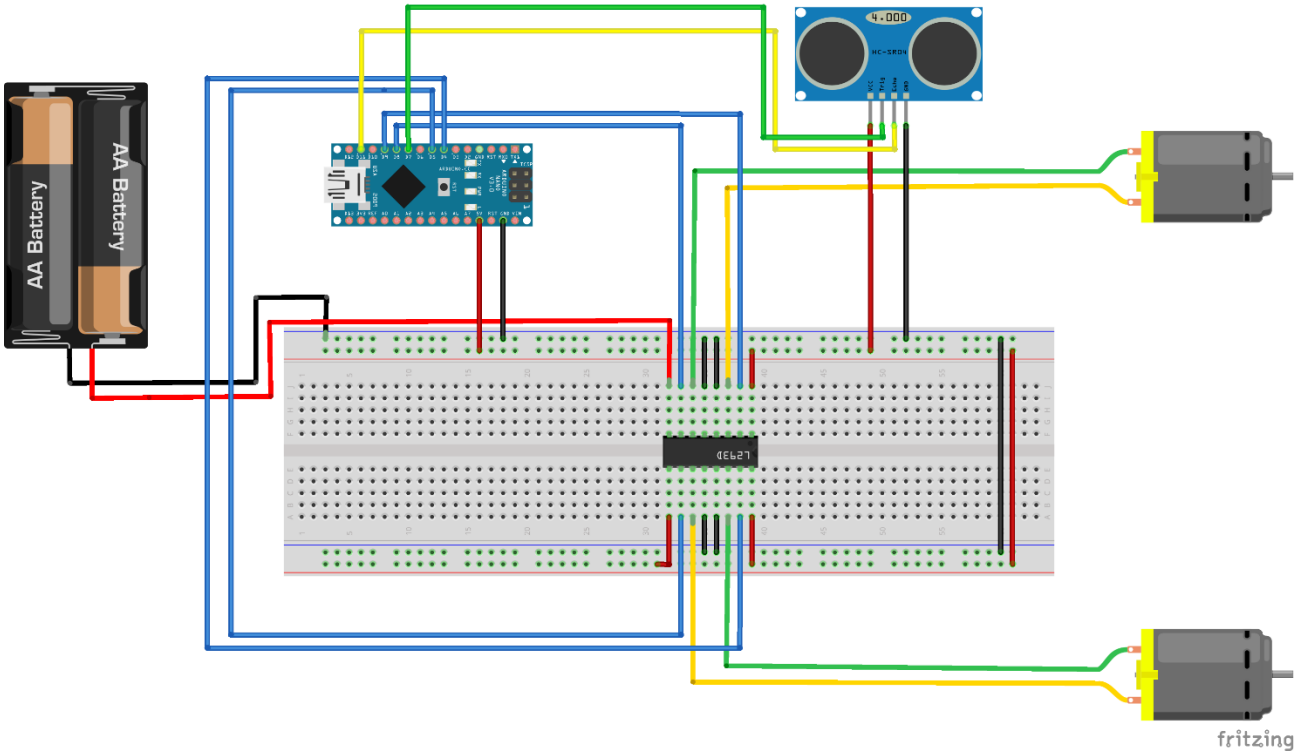
On branche la broche trigger du transducteur à la broche 7 de l'Arduino et la broche echo du transducteur a une sortie PWM. Nous prendrons la broche 11.

Le programme associé à ce schéma est le suivant :

```
/*Declaraction variable*/
int trigger = 7;                        // Broche 7
int echo = 11;                          // Broche PWM 11
float distance;

/*Configuration*/
void setup(){
  pinMode(trigger, OUTPUT);             // Mode Sortie
  pinMode(echo, INPUT);                 // Mode Entrée
  Serial.begin(9600);
}

/*Calcul la distance*/
float calculDistance(){
  digitalWrite(trigger, LOW);           //  Envoie 0V sur la Broche 7
  delayMicroseconds(2);                 //  Attente 2 microsecondes
  digitalWrite(trigger, HIGH);          //  Envoie 5V sur la Broche PWM 11
  delayMicroseconds(10);                //  Attente 10 microsecondes
  digitalWrite(trigger, LOW);           //  Envoie 0V sur la Broche 7
  return pulseIn(echo, HIGH) / 58.0;    //  Retourne la distance calculer
}

/*Boucle infini*/
void loop(){
  distance = calculDistance();          // Affectation de la distance
  Serial.println(distance);             // Affichage de la distance
  delay(15);                            // Pause 15 millisecondes
}
```

Le véhicule anti collision

Maintenant que nous avons vu comment utiliser un capteur à ultrason et les moteurs, on combine les deux parties.



Le schéma ci-dessus correspond au câblage de notre véhicule. Nous avons le capteur à ultrason sur les broches 7 et 11 de l'Arduino. Noté que la broche 11 est une sortie PWM. Le moteur gauche est branché sur la sortie 9 et 8 et le moteur droit sur les broches 5 et 4.

Voici le programme correspondant au déplacement du véhicule.

```
/*Declaraction variable*/
int trigger = 7;                        // Broche 7
int echo = 11;                          // Broche PWM 11
float distance;
int devantGauche = 9;                   // Broche 9
int derriereGauche = 8;                 // Broche 8
int devantDroite = 5;                   // Broche 5
int derriereDroite = 4;                 // Broche 4
/*Configuration*/
void setup(){
  pinMode(devantGauche, OUTPUT);        // Mode Sortie
  pinMode(derriereGauche, OUTPUT);      // Mode Sortie
  pinMode(devantDroite, OUTPUT);        // Mode Sortie
  pinMode(derriereDroite, OUTPUT);      // Mode Sortie
  pinMode(trigger, OUTPUT);             // Mode Sortie
  pinMode(echo, INPUT);                 // Mode Entrée
  Serial.begin(9600);
}
/*Boucle infini*/
void loop(){
  distance = calculDistance();          // Affectation de la distance
  Serial.println(distance);             // Affichage de la distance
  if(distance > 40){                    // Si la distance est supérieur a 40 cm
    avancer();                          //    le robot avance
  }
  else{                                 // Sinon
    arret();                            //    Le vehicule s'arrete
    delay(500);                         //    Pause 500 millisecondes
    reculer();                          //    Le vehicule recule
    delay(500);                         //    Pause 500 millisecondes
    droite();                           //    Le vehicule troune a droite
    delay(500);                         //    Pause 500 millisecondes
  }
  delay(500);                           // Pause 500 millisecondes
}
```

```
/*Fonction qui permet de faire avancer le vehicule*/
void avancer(){
  digitalWrite(devantGauche, LOW);
  digitalWrite(derriereGauche, HIGH);
  digitalWrite(devantDroite, LOW);
  digitalWrite(derriereDroite, HIGH);
}

/*Fonction qui fait reculer le vehicule*/
void reculer(){
  digitalWrite(devantGauche, HIGH);
  digitalWrite(derriereGauche, LOW);
  digitalWrite(devantDroite, HIGH);
  digitalWrite(derriereDroite, LOW);
}

/*Fonction qui fait reculer le vehicule a gauche*/
void gauche(){
  digitalWrite(devantGauche, LOW);
  digitalWrite(derriereGauche, HIGH);
  digitalWrite(devantDroite, HIGH);
  digitalWrite(derriereDroite, LOW);
}

/*Fonction qui fait tourner le vehicule a droite*/
void droite(){
  digitalWrite(devantGauche, HIGH);
  digitalWrite(derriereGauche, LOW);
  digitalWrite(devantDroite, LOW);
  digitalWrite(derriereDroite, HIGH);
}

/*Fonction qui fait arreter le vehicule*/
void arret(){
  digitalWrite(devantGauche, LOW);
  digitalWrite(derriereGauche, LOW);
  digitalWrite(devantDroite, LOW);
  digitalWrite(derriereDroite, LOW);
}

/*Fonction qui calcul la distance*/
float calculDistance(){
  digitalWrite(trigger, LOW);          //  Envoie 0V sur la Broche 7
  delayMicroseconds(2);                //  Attente 2 microsecondes
  digitalWrite(trigger, HIGH);         //  Envoie 5V sur la Broche PWM 11
  delayMicroseconds(10);               //  Attente 10 microsecondes
  digitalWrite(trigger, LOW);          //  Envoie 0V sur la Broche 7
  return pulseIn(echo, HIGH) / 58.0;   //  Retourne la distance calculer
}
```

Conclusion

Le but initial de ce projet été de réaliser un véhicule qui permet de cartographier une pièce. Au début, je voulais réaliser un drone mais malheureusement sans contrôleur de vol il est difficile de réaliser un drone. De plus, je ne voulais pas acheter des contrôleurs de vol. Hormis le fait que ce soit un peu cher, je pense que j'aurais beaucoup moins appris que de faire mon véhicule comme je l'ai fait. Je ne voulais surtout pas juste acheter des composants, en l'occurrence un drone, et le monter. Je pense que suivre un guide de montage est à la portée de tous et je ne trouvais pas intéressant de le faire. C'est pourquoi j'ai décidé de changer de projet.

Bien que j'ai changé de type de véhicule, la difficulté n'en reste pas moindre. Il fallait comprendre si ce que j'essayais de réaliser était possible ou non. C'est-à-dire que les composants que je disposais me permettrait de réaliser ou non mon véhicule. C'était la principale difficulté du projet voire la seule car je ne savais pas toujours si ce que je faisais était correct. S'il s'agissait d'allumer une LED, j'avais tout de suite le rendu car on peut voir la LED s'allumer. Et si elle ne s'allume pas d'où vient le problème ? La LED ne marche-t-elle plus ? Le branchement est-il correct ? Est-ce que la tension aux bords de la LED est adaptée ? Il y avait à chaque beaucoup de possibilités d'erreur.

D'un point vu technique, j'ai appris à recherche les informations dont j'avais besoin sur internet et je pense que n'importe qui peut trouver ce qu'il veut sur internet car il y a vraiment beaucoup de ressources. Que ce soit des vidéos, des tutoriels, des forums, il y a absolument tout sur internet. Combiné avec le travail des camarades de classe on peut vite fait le tour d'un composant.

D'un point de vue plus générale, sur l'Internet des Objets est un domaine vraiment très intéressant et je comprends maintenant pourquoi. J'ai vraiment beaucoup aimé ce module. N'importe qui avec seulement une idée peut réaliser ses projets. C'est accessible au grand public. La seule limite est l'imagination on peut vraiment tout faire. Après, pour ma part, je pense que pour les projets un peu plus sérieux comme faire un essaim de drones ou la cartographie d'une zone avec pose de balise, etc… il faudrait peut-être utiliser des composants de qualités et réaliser des objets, structures et composantes sur-mesure. L'utilisation d'une imprimante 3D par exemple semble est une bonne idée car durant mon projet je me suis rendu compte plusieurs fois que si je pouvais construire un véhicule sur mesure avec des compartiments spécifiques pour mes composants comme la carte Arduino, le bloc de pile, etc… ça aurait était plus simple. Ça revient à acheter un kit de montage sauf que le kit de montage est fait par moi.

Démonstration

Voici une vidéo commentée du projet

Composant utilisé

Voici un descriptif des composants utilisés pour réaliser le projet

Nom	Quantité	Prix en euros	Lien
Arduino Nano V3	1	7	Amazon
Shield Nano	1	3	Amazon, Letmeknow
L293D	1	1	Amazon
L293D Motor Drive Shield pour Arduino UNO MEGA Nano	1	10	Amazon
Roue pivotante	1	3	Amazon
Moteur pour voiture	2	10	Amazon, LetmeKnow
Roue	2	(fourni avec les moteurs)	Amazon
Plaque de PVC	1	8	Amazon
Capteur à ultrason	1	3	Amazon, Letmeknow
Bloc 4 pile 1,5V	1	2	Amazon

Source

Arduino PWM : Lien vers arduino.cc
Arduino DC Curent per I/O Pin : Lien vers playground.arduino.cc
Pont en H explication : Lien vers openclassroom.com
Tuto Arduino : Liens vers openclassroom.com
Arduino Documentation : Liens vers arduino.cc