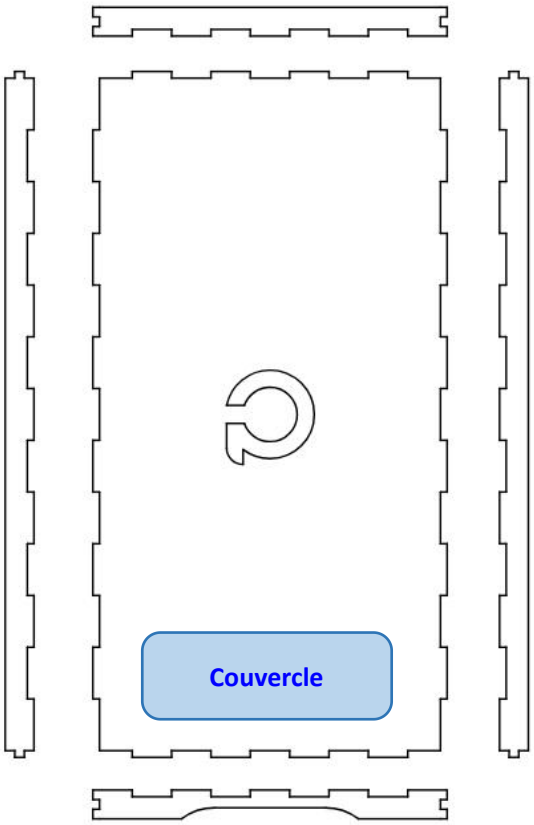
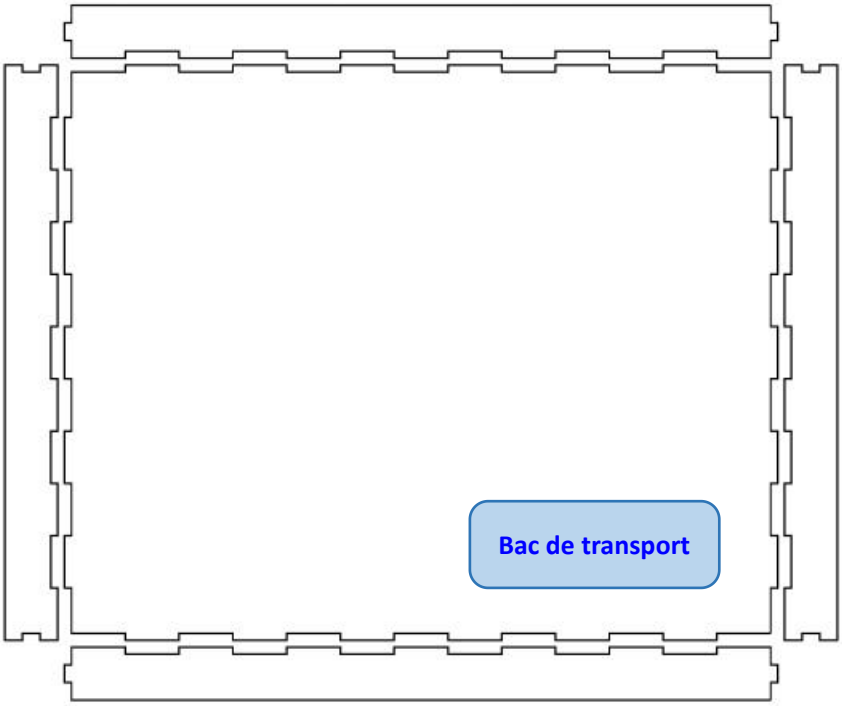
# RAPPORT DE SEANCE 5 :

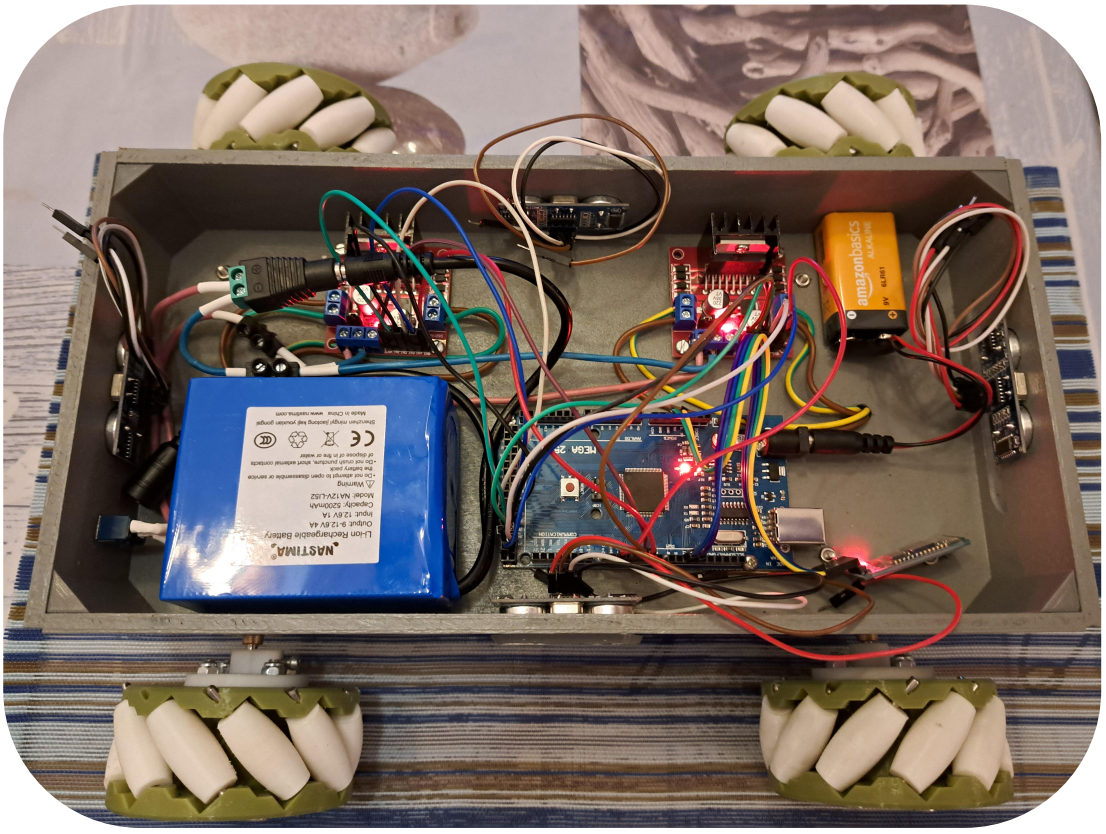
Durant la **cinquième séance** nous avons réalisé la **structure du couvercle et du bac** de la plate-forme. Pour cela nous avons utilisé un **générateur de boite** ou nous avons du entrer les dimensions (*310x160x13mm* ***et*** *310x250x23mm*). Nous avons récupéré un fichier **SVG** puis nous l'avons converti en **DXF** afin de le **modifier** sur ***OnShape*** (*encoche capteur ultrason*).

**Ensuite, nous l'avons reconverti en SVG afin de le découper au laser.**
**Nous avons finalement choisi du *bois* car c'est moins cher que le plexiglas et d'une résistance équivalente.**

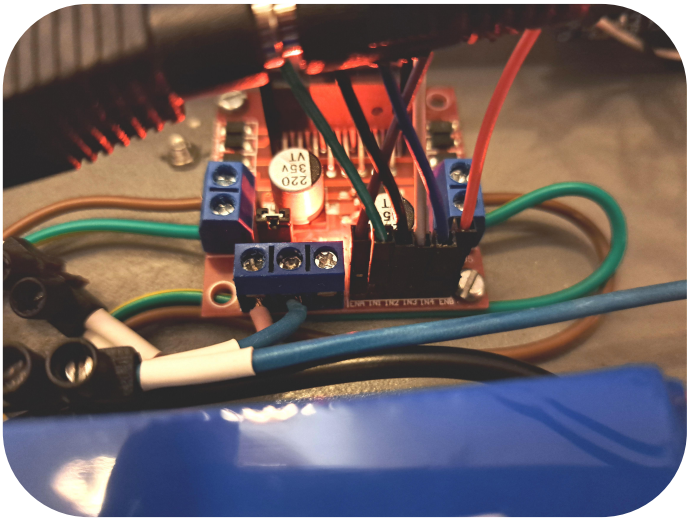



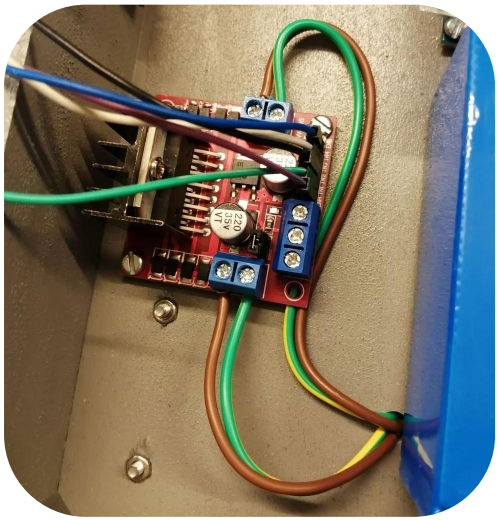
Ensuite, nous avons **mis en place les moteurs** et nous avons **assemblé** une bonne partie de la **plate-forme** avec les **deux ponts en H** *(vis 2x8)*, la **carte arduino mega**, les **capteurs ultrasons** et une **pile 9V** pour l'**alimentation des cartes et des capteurs**.
J'ai amené ma **batterie 12V 4A** afin de pouvoir faire des tests.




**Nous avons peint le bois en gris pour plus de style et nous avons fait le câblage des ponts en H avec les moteurs, l'alimentation et la carte arduino.**



**«Cable Management»**



Enfin nous avons **réalisé le code arduino** qui contient la **partie gestion des caractères reçus en *Bluetooth***, le **fonctionnement des capteurs ultrasons** et des **moteurs**.

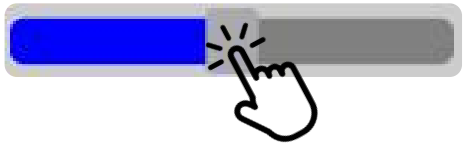
**Exemple pour avancer, si le caractère «a» est reçu alors la plate-forme avance.**

```
else if (message == "a"){
  Serial.println("Avancer");

  // Sens des moteurs
  digitalWrite(INAVG1, LOW);
  digitalWrite(INAVG2, HIGH);
  digitalWrite(INAVD1, HIGH);
  digitalWrite(INAVD2, LOW);
  digitalWrite(INARG1, HIGH);
  digitalWrite(INARG2, LOW);
  digitalWrite(INARD1, LOW);
  digitalWrite(INARD2, HIGH);

  // Vitesse des moteurs
  analogWrite(ENAVG,vitesse);
  analogWrite(ENAVD,vitesse);
  analogWrite(ENARG,vitesse);
  analogWrite(ENARD,vitesse);
}
```

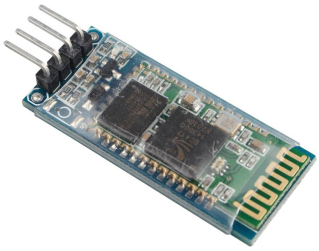
**Partie du code pour déterminer la distance des 4 côtés et ensuite arrêter les moteurs.**

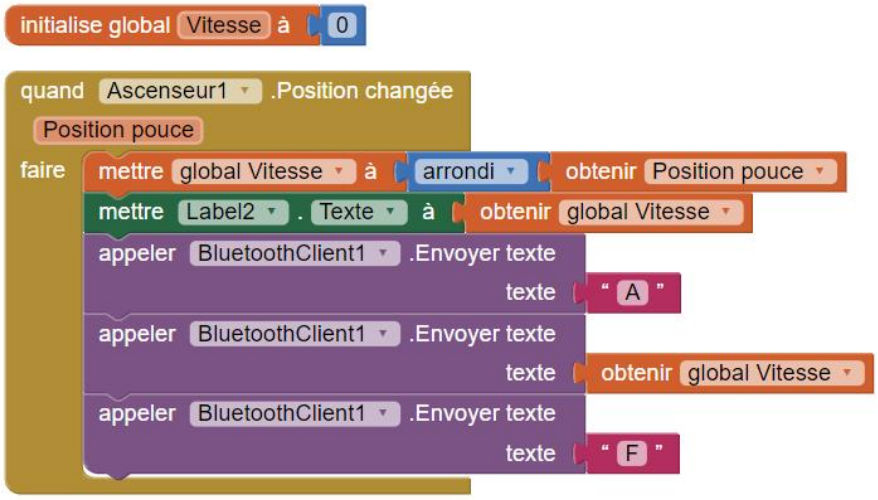
```
// Recupere la distance obstacles
distAV = sonarAV.ping_cm();
distAR = sonarAR.ping_cm();
distG = sonarG.ping_cm();
distD = sonarD.ping_cm();
Serial.println("##########");
Serial.println(distAV);
Serial.println(distAR);
Serial.println(distG);
Serial.println(distD);

if (distAV <= distMIN and distAR <= distMIN and distG <= distMIN and distD <= distMIN){
  analogWrite(ENAVG,0);
  analogWrite(ENAVD,0);
  analogWrite(ENARG,0);
  analogWrite(ENARD,0);
}
```

**J'ai rajouté un curseur pour ajuster la vitesse de la plate-forme.**



Ici, lorsque le curseur bouge, l'application envoie le caractère *'A'* (attention ce qui suit est un nombre à traiter) suivi du nombre (compris entre 0 et 255) puis le caractère *'F'* (attention fin du nombre) au module Bluetooth.






```
else if(BlueT.available()){
  message = (char)BlueT.read(); // On recupere le message sous forme de caractere
  delay(6);

  if (message == "A"){ // Sert à indiquer que les caracteres qui vont suivre "A" (attention) est un nombre
    String newVitesseSring = "";
    while (message != "F"){ // Tant qu'on a pas recu de "F" (fin du nombre)
      delay(6);
      message = (char)BlueT.read(); // Recupere 1 caractere
      delay(6);
      // Verifie si le caractere est un chiffre pour former le nombre correctement
      if (message == "0" or message == "1" or message == "2" or message == "3" or message == "4"
      or message == "5" or message == "6" or message == "7" or message == "8" or message == "9"){
        newVitesseSring = newVitesseSring + message;
      }
    }
    Serial.println("Vitesse changee a : "+newVitesseSring+"/255");

    int newVitesse = newVitesseSring.toInt(); // Recupere la valeur (int) de la vitesse
    if (newVitesse >= 0 and newVitesse <= 255){ // Verifie si la vitesse correspond
      vitesse = newVitesse; // Change la vitesse
    }
  }
```