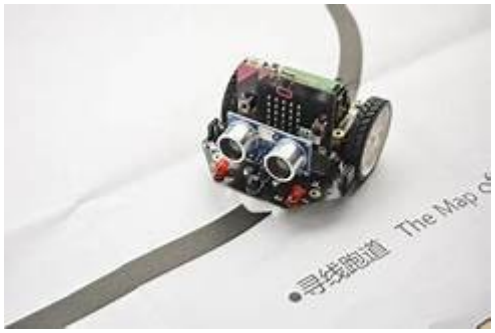
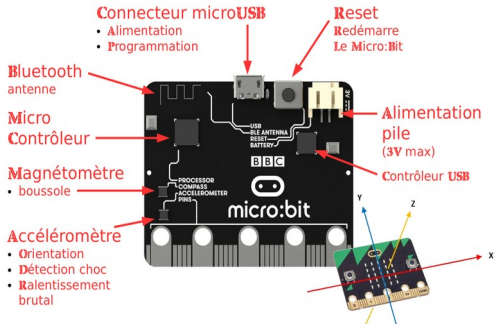
Projet de NSI : Nos différents programmes

Auteurs : FERREIRA COSTA Micael, DEMAY Mathieu et BAYNI Ilyas

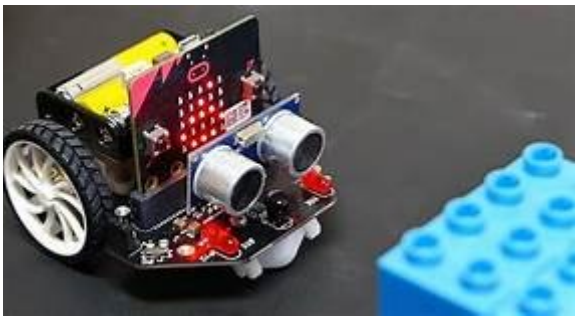
Description de nos projets :

- Notre premier projet est basé sur la programmation python micro:bit python d'un robot Maqueen qui consiste à lui faire suivre une ligne noir sur fond blanc. À chaque fois que notre robot détecte qu'il n'est plus ou qu'il s'éloigne de la ligne noire, il s'orientera de manière à revenir sur la ligne noire. Exemple : s'il s'éloigne à droite de la ligne noire, il reviendra automatiquement à droite. (1)

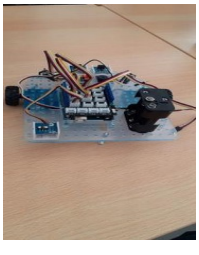




- Notre deuxième projet est basé sur la programmation micro:python d'un robot Maqueen qui consiste à lui faire éviter des obstacles. Si les ultrasons du robot détectent un obstacle, objet à moins de 35 cm, le robot décidera de tourner aléatoirement soit à droite soit à gauche afin d'éviter cet obstacle. (2)



- Notre troisième est dernier programme, est programmé en langage C avec Arduino. Le Projet est de réaliser un petit drone autonome. L'idée étant de faire en sorte qu'une fois lancé il puisse se déplacer de lui-même dans aucune intervention humaine. (3)



Liste du matériel utilisé : Programme 1 et 2 :

- Une carte Micro:bit V1.5
- Un robot Maqueen

Liste du matériel utilisé : Programme 3 :

- Carte informatique :
- Un Arduino Uno (Pièce n°1 de l’annexe)
- Un shield Grove (Pièce n°2)

Capteurs :
- Un Télémètre a ultrason (Pièce n°3)
- Deux potentiomètres rotatifs (Pièce n°4)
- Un interrupteur à glissière (Pièce n°5)

Moteurs :
- Deux moteurs à rotation continu (Pièce n°6)

Afficheur :
- Un Afficheur LCD I2C (Pièce n°7)

Autre :
- Câblage Grove (Pièce n°8)
- Boîtier 6 piles + piles + adaptateur (Pièces n° 9, 10 et 11)
- Kit de montage fourni par l’école

Fonctionnements capteurs, actionneurs et autres (programme 1 et 2) :

Moteurs : Moteurs à rotation continu : servomoteur à rotation continu qui a une plage de réglage proportionnelle à sa vitesse de 0 (éteint) à 255(vitesse max.), la vitesse max. en tours par minute n'est pas connue :

- Moteur droite : fait tourner la roue droite du Maqueen
- Moteur gauche : fait tourner la roue gauche du Maqueen

Le télémètre à ultrason : Calcul la distance en cm le séparant d'un objet devant lui dans un maximum d'environ 4m

Capteurs IR : Ce sont une classe d'appareils qui transmettent et/ou reçoivent un rayonnement infrarouge. Comme la couleur noire absorbe le rayonnement et que la couleur blanche le reflète, lorsque la LED IR émet un rayonnement, il ne sera réfléchi et détecté par le récepteur/photodiode IR que lorsque la surface est blanche ou de couleur claire.

Boîtier 3 piles : sers d'alimentation mobile pour le coté autonome, l'autonomie (en temps de fonctionnement) et la consommation électrique du Maqueen n'est pas connues.

Fonctionnements capteurs, actionneurs et autres (programme 3) :

Le télémètre à ultrason : Calcul la distance en cm le séparant d'un objet devant lui dans un maximum d'environ 4m

Potentiomètre rotatif : Résistance variable se branchant sur un port analogique et renvoi des valeurs de 0 (totalement fermer) à 1023 (totalement ouvert)

Interrupteur à glissière : Fonctionnement classique d'un interrupteur (on-off) qui renvoi des valeurs de 0 (off) ou 1 (on)

Moteurs à rotation continu : servomoteur à rotation continu qui a une plage de réglage proportionnelle à sa vitesse de 0 (éteint) à 1023 (vitesse max.), la vitesse max. en tours par minute n'est pas connue.

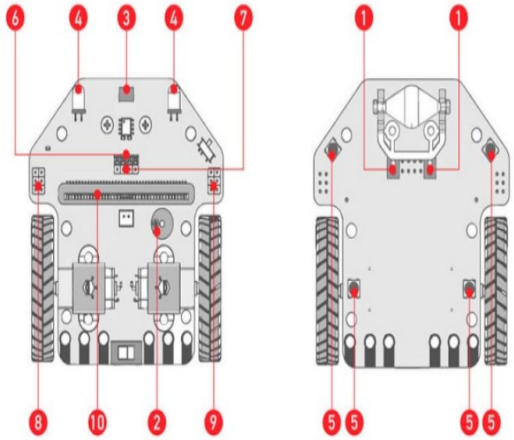
Afficheur LCD : Afficheur LCD paramétrable qui contient 2 lignes et dont on peut régler la couleur de l'arrière-plan.

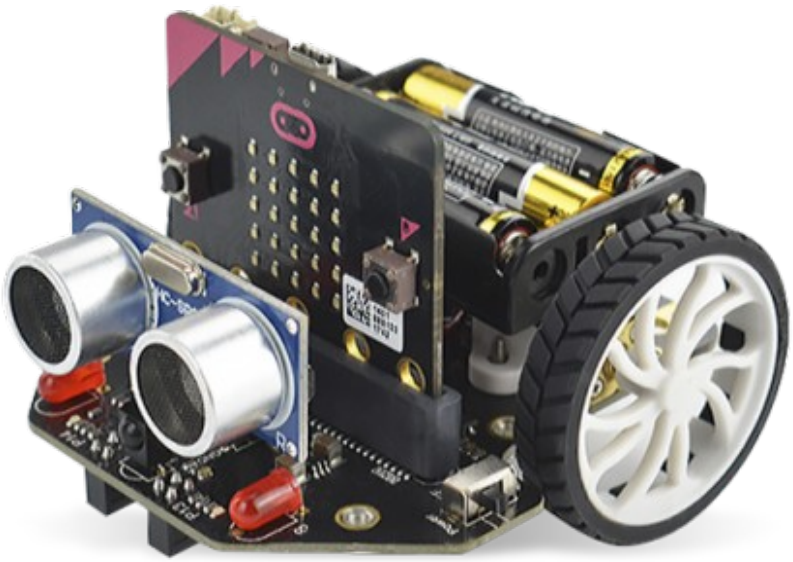
Boîtier 6 piles : sers d'alimentation mobile pour le coté autonome, l'autonomie (en temps de fonctionnement) et la consommation électrique du drone ne sont pas connues.

Maqueen :

1- Caractéristiques châssis

1-capteurs de ligne infrarouge X2
2-buzzer
3-récepteur infrarouge (télécommande)
4-LED rouge X2
5-LED RVB (d'ambiance) X4
6-connecteur capteur ultrason
7-connecteur I2C X2
8-connecteur servo X2
9-connecteur supp X2
10- Slot carte micro:bit
+moteur I2C +roueX2
+connecteur alimentation (pack 3 piles)
+interrupteur alimentation(on/off)
+interrupteur son (on/off)
+roue libre devant





Code Programme 1 :

```
from microbit import *

# motor control
i2c.init(freq=100000, sda=pin20, scl=pin19)

def MotorControl(motor, direction, speed):
  buf = bytearray(3)
  # contrôle du moteur (0 pour M1 (L) et 1 pour M2 (R))
  if motor == 0:
      buf[0] = 0x00
  else:
      buf[0] = 0x02
# contrôle de la direction (0 pour CW et 1 pour CCW)
  if direction == 0:
      buf[1] = 0
  else:
      buf[1] = 1
  # set speed, 0 - 255
  buf[2] = speed
  i2c.write(0x10, buf)

while True:
#Si Le capteur IR lit 1 si blanc (réfléchissant), 0 si noir (ou ne réfléchissant pas la lumière infrarouge)

#Si la broche 13 (capteur IR gauche) lit le blanc et la broche 14 (capteur IR droit) lit le noir, le véhicule tournera à droite.
    if pin13.read_digital() == 1 and pin14.read_digital() == 0:
        MotorControl(0,0,255)
        MotorControl(1,0,0)
#Si la broche 13 (capteur IR gauche) est noire et la broche 14 (capteur IR droit) est blanche, il tournera à gauche.
    elif pin13.read_digital() == 0 and pin14.read_digital() == 1:
        MotorControl(0,0,0)
        MotorControl(1,0,255)
# Si les deux capteurs IR sont noirs, il ira tout droit.
    elif pin13.read_digital() == 0 and pin14.read_digital() == 0:
        MotorControl(0,0,255)
        MotorControl(1,0,255)
# si les deux capteurs IR lisent tout blanc, il s'arrêtera.
    else:
        MotorControl(0,0,0)
        MotorControl(1,0,0)
```

Code programme 2 :

Si la distance entre l'ultrason est inférieure
à 35 cm et qu'elle est différente de 0
Alors nous établissons une variable item qui choisis au hasard « True » ou
« False »
Si item est égal « True » alors le moteur M1 continue de tourner et le moteur 2 arrête de tourner (Le robot tourne à gauche)
Fais un pause 800 ms
Si item est égal « False » alors le moteur M2 continue de tourner et le moteur M1 arrête de tourner (Le robot tourne à droite)
Fais une pause 800 ms
Sinon si le capteur ultrason est à une distance supérieure à 35 cm alors le moteur M1 et M2 tourne tous les deux au maximum (ce qui fait avancer le robot tout droit)

Code programme 3 :

```
#include <TSServo.h>
#include <rgb_lcd.h>
#include <Ultrasonic.h>

boolean __ardublockDigitalRead(int pinNumber)
{
  pinMode(pinNumber, INPUT);
  return digitalRead(pinNumber);
}


TSServo servo_pin_7;
TSServo servo_pin_8;
rgb_lcd rgbLcd;
Ultrasonic us_pin6(6);

void ecran();
void rota();

void setup()
{
  servo_pin_7.attach(7);
  servo_pin_8.attach(8);
  rgbLcd.begin(16,2);
}

void loop()
{
  ecran();
  if (__ardublockDigitalRead(4))
  {
    rota();
    servo_pin_7.write(( analogRead(0) + 150 ), false);
    servo_pin_8.write(( analogRead(0) + 0 ), true);
  }
  else
  {
    servo_pin_7.write(250, false);
    servo_pin_8.write(0, false);
  }
}

void ecran()
{
  rgbLcd.setCursor(0, 0);
  rgbLcd.print("Vitesse:" );
  rgbLcd.print(analogRead(0) );
  rgbLcd.print("              ");
  rgbLcd.setCursor(0, 1);
  rgbLcd.print("rota degre:" );
  rgbLcd.print(( ( analogRead(2) * 360 ) / 1024 ) );
  rgbLcd.print("              ");
  rgbLcd.setRGB(constrain(0,0,255),constrain(255,0,255),constrain(0,0,255));
}

void rota()
{
  while ( ( ( us_pin6.MeasureInCentimeters() ) <= ( 10 ) ) )
  {
    servo_pin_7.write(analogRead(0), false);
    servo_pin_8.write(analogRead(0), false);
    delay( ( ( analogRead(2) * 9 ) / 1024 ) * 1000);
  }

}
```

Explication générale du fonctionnement du programme : Dans la boucle principale, on détecte la position de l'interrupteur à glissière. Si l'interrupteur est en position off, rien ne se passe et les moteurs sont à l'arrêt. Si l'interrupteur est en position on, les moteurs seront activés vers l'avant et auront une vitesse de rotation proportionnelle à l'ouverture d'un premier potentiomètre rotatif. Deux sous programmes sont appelés dans la boucle principale : le sous-programme « ecran » qui lui fonctionne indépendamment de la position de l'interrupteur et le sous-programme « rota » qui lui est appelé uniquement si l'interrupteur est en position on.

Le sous-programme « ecran » lui va appliquer la couleur verte à l'arrière-plan de l'afficheur LCD et afficher sur la première ligne la vitesse en rapport à l'ouverture du potentiomètre contrôlant la vitesse (sur une base 1024 : 0 = moteur arrêter et 1023 = moteur à pleine vitesse) ; sur la seconde ligne est affiché l'ouverture du second potentiomètre rotatif, lui agissant sur le sous-programme « rota » explicité ci-dessous (toujours sur une base 1024 comme le premier potentiomètre).

Le sous-programme « rota » lui va gérer l'angle auquel le drone va tourner lorsqu'un obstacle est détecté à moins de 10cm. Un second potentiomètre est utilisé pour ce sous-programme. Des essais ont été menés pour arriver à la conclusion que le robot, à pleine vitesse fera une rotation de 360° en plus ou moins 9 secondes. Une formule mathématique va convertir la valeur de l'ouverture du potentiomètre en temps (potentiomètre fermé = 0s et potentiomètre ouvert à fond = 9s). L'idée est que lorsque le potentiomètre est fermé, le robot s'arrête de tourner instantanément à partir du moment où l'objet n'est plus détecté par le télémètre à ultrason. Et que lorsqu'il est ouvert à fond, dès que le télémètre détecte un objet trop proche, le drone fait une rotation de 360°. L'objectif est de pouvoir jouer avec l'ouverture du potentiomètre pour que le drone ne fasse pas trop de rotation mais qu'en même temps il ait assez tourner pour ne pas accrocher l'objet en passant trop proche : une ouverture nulle fera que le robot ne tournera que si le télémètre détecte un objet à moins de 10cm, une ouverture d'un quart du potentiomètre (la valeur affiché sera de 1023/4 ≈ 256) fera pivoter le robot de 90° (d'un quart) dès la détection d'un objet à moins de 10 cm, et ainsi de suite : une ouverture de la moitié fera faire demi-tour au drone et aux ¾ fera faire ¾ de tour soit 270° au drone. L'objectif est d'avoir une aisance dans les réglages et de pouvoir adapter au mieux possible le drone à son environnement.

Études linéaires d’un morceau de code :

```
void loop()
{
  ecran();
  if (__ardublockDigitalRead(4))
  {
    rota();
    servo_pin_7.write(( analogRead(0) + 150 ), false);
    servo_pin_8.write(( analogRead(0) + 0 ), true);
  }
  else
  {
    servo_pin_7.write(250, false);
    servo_pin_8.write(0, false);
  }
}
```

Ligne1 : fonction de type void (vide qui ne retourne rien) nommé loop qui sera la boucle principale du programme, elle ne prend pas d’arguments.
Ligne 3 : appel au sous-programme ecran
Ligne 4 : le programme test la valeur de l’entrée numérique numéro 4 (l’interrupteur à glissière) avec une condition (if)
Ligne 6 : Si la condition est remplie, que l’interrupteur est sur on alors : on fait appel au sous-programme rota
Ligne 7 : Si la condition est remplie, que l’interrupteur est sur on alors : on règle la vitesse du premier servomoteur à la valeur de l’ouverture du potentiomètre de vitesse + 150 points (les moteurs étants mal étalonnés, ils ne tourneraient pas à la même vitesse), le false signifie que le moteur ne tournera pas en sens inverse.
Ligne 8 : Si la condition est remplie, que l’interrupteur est sur on alors : on règle la vitesse du second servomoteur à la valeur de l’ouverture du potentiomètre de vitesse, le true signifie que le moteur tournera en sens inverse, les moteurs étants les même mais placé de façon opposée, l’un doit tourner en sens inverse pour que le drone aille vers l’avant.
Ligne 10 : Si la condition n’est pas remplie, que l’interrupteur est sur off alors : (suite du programme)
Ligne 12 : Si la condition n’est pas remplie, que l’interrupteur est sur off alors : le premier servomoteur se voie désactiver, il y a une valeur de 250 car le servomoteur étant mal étalonné, s’arrête a 250 et tourne à l’envers à 0.
Ligne 13 : Si la condition n’est pas remplie, que l’interrupteur est sur off alors : le second servomoteur se voie désactiver.
Lignes suivantes : fin de la boucle principale et retour au début pour recommencer un cycle.

Extensions (programme 1 et 2) :

On peut attribuer à nos programmes d'autres fonctions :
- un potentiomètre pour réguler la vitesse
- rajouter des fonctions avec LED

On peut imaginer aussi ajouter d'autre programmes :
- Télécommandé le Maqueen
- Programme pour de la musique

Ce prototype avec certaines améliorations faites peut devenir un jouet pour enfant

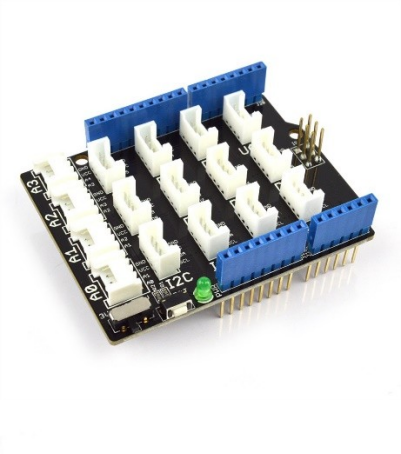
Extension (programme 3) :

Ce robot pourrait être utilisé pour la cartographie de lieux, il suffirait de coder un programme pour garder en mémoire les moments où il à rencontrer des obstacles et au bout d'un certain temps à naviguer dans une zone il l'aurait cartographier.

Annexe :

Pièce n°1 :

Pièce n°2 :



Pièce n°3 :

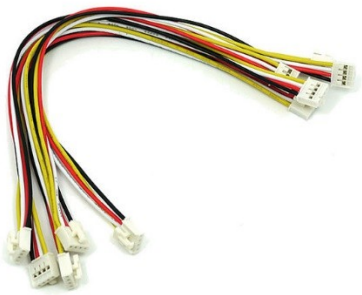
Pièce n°4 :

Pièce n°5 :

Pièce n°6 :

Pièce n°7 :

Pièce n°8 :



Pièce n°9 :

Pièce n°10 :

Pièce n°11 :