RoboCupJunior 2022 OnStage

Technical Description Paper

Team Information

Team Name:	**MariTeam**
Country / Region:	**France**
Do you need a translator? If yes, in what language? :	
☐Yes or ☒ No. Language:	
Has your team read the 2022 OnStage and RoboCupJunior rules and scoresheets?	
☒Yes or ☐ No.	
By selecting "Yes", you confirm that you have read the rules for competition, you understand them and agree to fully comply with them. These can be found at the official website (http://junior.robocup.org). If in doubt, please access the site and download the latest one.	
The Participants Name and their Technical Role: What are the roles of each team member? Please indicate each team member's name and their role. We would like to know how you contributed to the project as a team member.	
Member 1:	**Arthur: Artistic leader**
Member 2:	**Matteo: Hardware**
Member 3:	**Paul: Software**
Member 4:	**Etienne: Electronics**

Collaboration:

Please link any team websites or online depositories for open source learning and continuing development.

It is always important to share our expertise and learning. RoboCup is a great way to learn more, share your experience and aspire to new project goals. Learning opportunities are the ultimate goal of the RoboCup Community.

https://github.com/NehKowp/on-stage

Technical Information [500 Words Per Paragraph]

Overview:

What is the theme of your project? If you made multiple robots, please describe them below (the number, kinds of robots, etc.).

By our performance of interaction between robot/human, we denounce the pollution of the oceans, an emergency topic. Our performance takes place in an aquatic environment. This universe will be visible with colored corals, a jellyfish and a turtle which is our main robot. At first the corals light up and the turtle gets moving. But she will unfortunately confuse a jellyfish, her natural food, with our robotic garbage bag. It represents the aquatic pollution of Man (in general) and the marine ecosystem that we let "die". The corals light off, symbolizing their “oxidation”. Humans, terrified by this spectacle, help the turtle, notably by teaching it to differentiate between jellyfish and plastic waste. Everything finally returns to normal and the robots and the corals come back to life.
We have 3 robots: the turtle Marie, the jellyfish, the waste bag and luminous corals.
The turtle can move each of its legs, its tail thanks to 5 servo motors and can move in all directions with the help of mecanum wheels. The structure is made of plywood and the shell is molded from a real turtle shell. Moreover there is a camera in the head that detects faces and colors using AI.
We have a trash bag controlled with hand gesture recognition on raspberry 4 and an Arduino nano. The trash bag is equipped with a line sensor.
The jellyfish robot moves with an I2C motor and driver who are controlled with an arduino nano with leds.
Finally, our corals are lit by LEDs and controlled by an Arduino board and a xbee module.

Mechanical Design:

Detail all the ingenuity that the team has made to realize difficult movements of the robot, such as moving smoothly, keeping balance, grasping objects, and so on. Include photos of designs and models (CAD/CAM models etc.).

We used mecanum wheels to have a better freedom for the movements and drive in every direction. We use an I2C motor for the other robot.

Materials:

Detail any materials used in constructing robots, including the purpose of weight reduction, strength preservation, improvement of finish, etc.

We used paper mache and fence (paper + glue) for the shell of the turtle and the jellyfish, plywood for the legs, tail and the châssis of the turtle, metal for the chassis of the jellyfish and the trash bag, and carbon fiber for the trash bag too, to guarantee a better balance of the robot. We also used 3D printing with PLA to print the turtle’s head in 2 parts, the trash bag robot cover, and several other pieces. Finally, we used wood planks and branches to represent corals.

Microcontrollers:

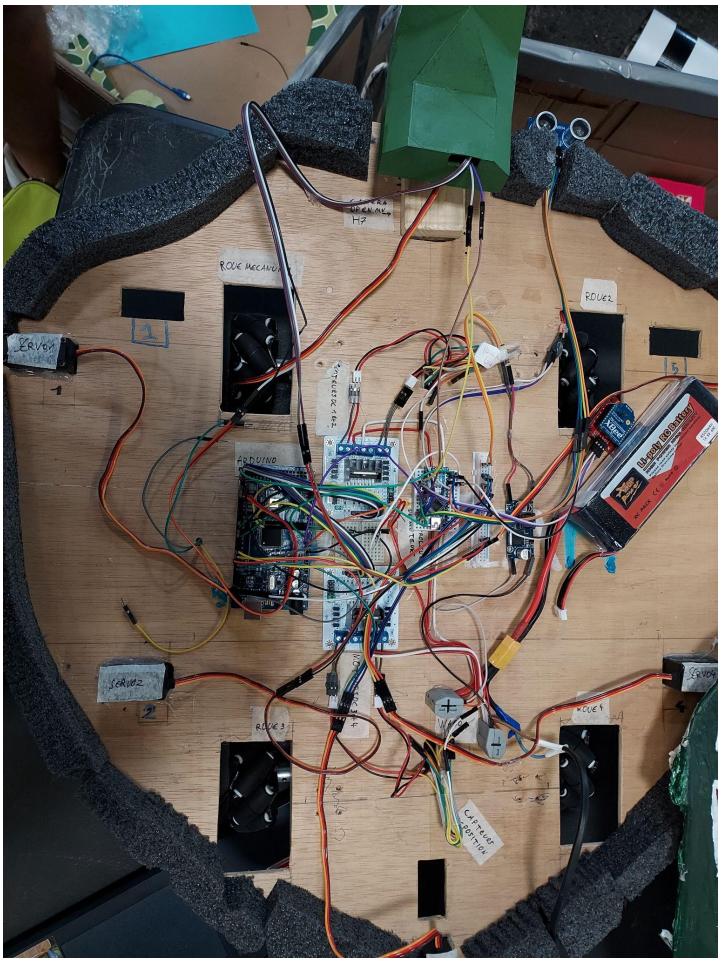
Indicate the name and kind of controller(s) you are using. For example: NXT, Arduino, Raspberry Pi, EV3, etc.

We use 3 Arduino nano old bootloader, 2 Arduino uno, a raspberry 4

Sensors: Which sensors are you using? For example: Touch, Light, Sound, Rotation, Shaft encoder, Compass, Proximity, Ultrasound, Color, Compass, etc.
We use UltraSound to stop the turtle when it's too close from an obstacle like the trash bag or the jellyfish. We use Color Sensor to prevent the risk of a robot out of the scene. We use a sound sensor to detect clap sound.

Electrical/Electronic Design:

Have you developed your own electronics? For example, motor controllers, voltage regulators, amplification circuits, etc. Include photos of custom board designs (schematics, board layouts etc.).



Wireless Communication:

Are you using wireless communication(s)? If so, what type? No team is allowed to use Wi-Fi. Please refer to the Official RoboCupJunior OnStage 2022 Rules.

We use an xbee communication between the turtle and the corals to change their colors when an object is detected.

Power Management:

What kind of battery is built into/used in your robot? Please clarify the name and type of battery, together with current and voltage. (Teams should comply with the Official RoboCupJunior OnStage 2022 Rules). What measures are you using to regulate your power supplies?

We use a lipo battery with a voltage of 14,8 and 4,5Ah for the turtle. We use 4 lgchem battery with a voltage of 3.5V for each secondary robot (jellyfish and plastic bag). We are using a resistance to regulate the electrical flow.

Programming Language:

What programming language(s) are you using? Are you using any libraries/datasets? You may wish to add a link to your GitHub repository.

We are using different languages, C++ for the Arduino part and Python with all the Rasberry parts. With the C++, we use the I2C motors library. With Python, we use many libraries like Mediapipe for hand gesture recognition, openCV for camera capture, csv to save all gestures. For the face and color recognition we use the micropython library of openMv.

Sources:

Please provide links to any manuals, documentation or open source repos used in the development of the project.

https://docs.openmv.io/library/index.html#python-standard-libraries-and-micro-libraries
https://docs.arduino.cc/

Performance Information [1000 Words Per Section]

Features:

An OnStage Performance must showcase the implementation and integration of robotic features in ways that visually enhance or add value and contribute to the theme or story being portrayed. Consequently, teams must present four of what they believe are their robotic features: for example system/sensor integration, electromechanical design, interaction or software solutions implemented on their robot(s). The aim should be to present the integration of the chosen features and how the features contribute to the progression of the performance.

In our main robots we use face recognition to express the interaction of a human with a turtle.The face recognition works with every face. Firstly it searches the shape of a face, an oval face with a mouth and a nose, but for better results we implement the search of eyes inside of the face.

We also use color recognition by the same camera as face recognition. By setting different thresholds we define a color and with a function that detects the colors in the image of the camera it will move the robot in the directions of the colors.

For our second robot we implement the recognition of hand gestures. We use a mediapie library in order to find a hand and attribute to each finger a number. Then we stock the position of every finger of a hand into another library to save a gesture. Afterwards when the recognition finds a gesture already save in our library its send a signal to an arduino nano to move the robot agreement to the gesture.

For the movement of our main robots, which symbolize a turtle, we choose to use 4 mecanum wheels to create a more natural movement. By our mecanum wheels we can go in every direction with a vectorial direction. It works by compensate the speed of each wheels to go in the direction chosen.

Interaction:

Do you interact with the robot? (i.e. human to robot, robot to robot interaction) If so, how?

We are interacting with the robot with the help of face detection, with an openmv camera placed in the turtle's head, hand gesture recognition with a camera placed in the trash bag robot, applause detection and distance detection with 2 ultrasound sensors, and a sound sensor placed in the turtle. The robots are interacting together with the help of xbee communication in the turtle and in the corals, and color recognition with the openmv camera.

Integration:

How do you use your sensors, actuators and robot(s) to create a cohesive performance? Are you using multisensor systems? Do the robots rely on each other during the performance?

Color detection is used for the turtle. At first, it searches for yellow, because it is the color of the trash bag robot, which we are controlling with hand gestures and the camera placed on it. Then a Human will be facing the turtle, and the camera will recognize it. Then it will search for blue, which is the color of the jellyfish. The ultrasound captors are used to avoid collisions. We are using xbee because the corals are linked by bluetooth with the turtle, because the corals' lights will change depending on the turtle's actions. Finally, the sound sensor is used to measure audience applause, as it will influence the coral's light color and intensity.

Challenges and Difficulties:

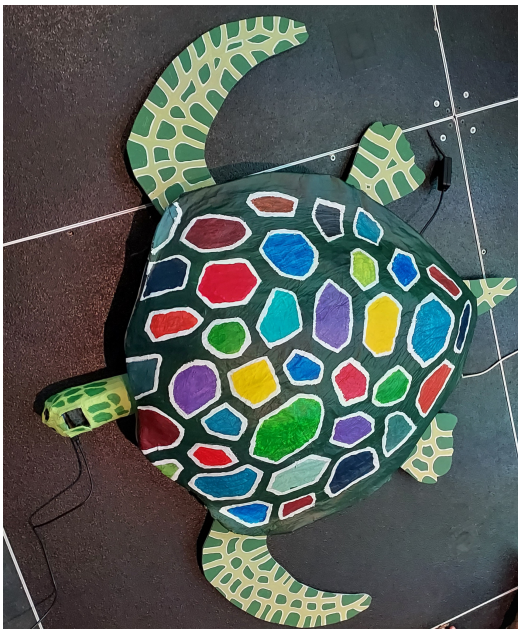
What challenges and difficulties has the team encountered? How did you overcome them? If you did not, what would you do if it happened again?

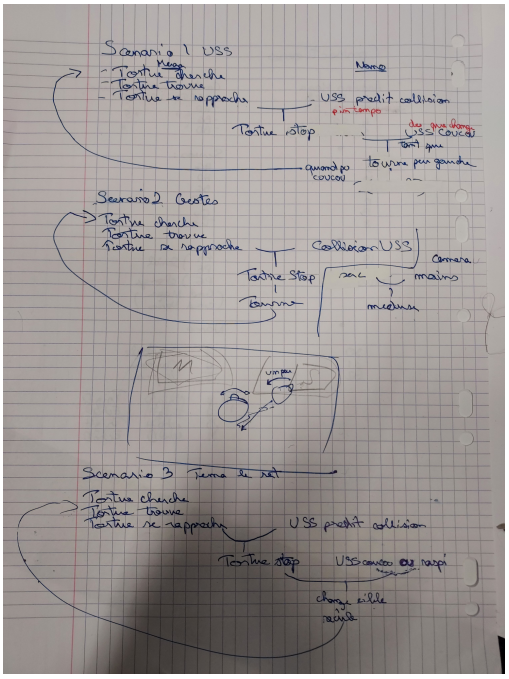
We had problems with the transportation. We had too much stuff and the turtle and corals were too big. We had to cut in half the corals and we had to create a custom box for the plane. We also had issues with our 3d printer because it had a lot of bugs, but now everything is fine. We had issues with color recognition. In fact, the camera detects differently the color depending on the ambient light. At first the turtle wasn't finding the color, but then we modified the thresholds and it was finally working. We had issues with the raspberry pi because of Linux, as it was quite hard to understand at first. But after some tutorials we finally figured out how it works.

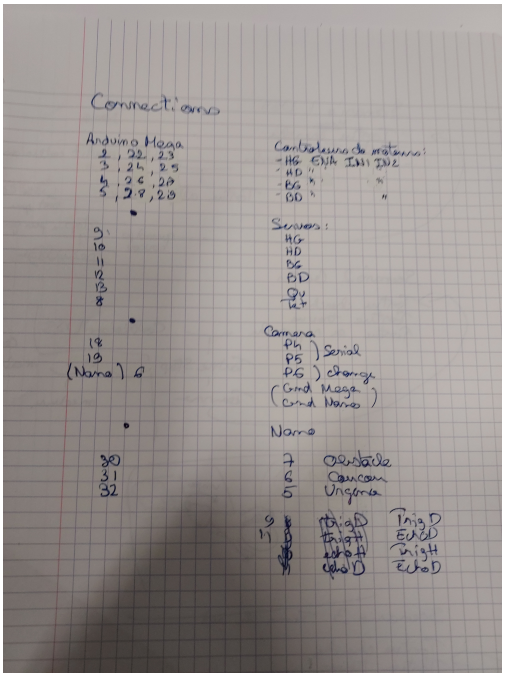
Appendix [Limit to 5 pages - excluding code]

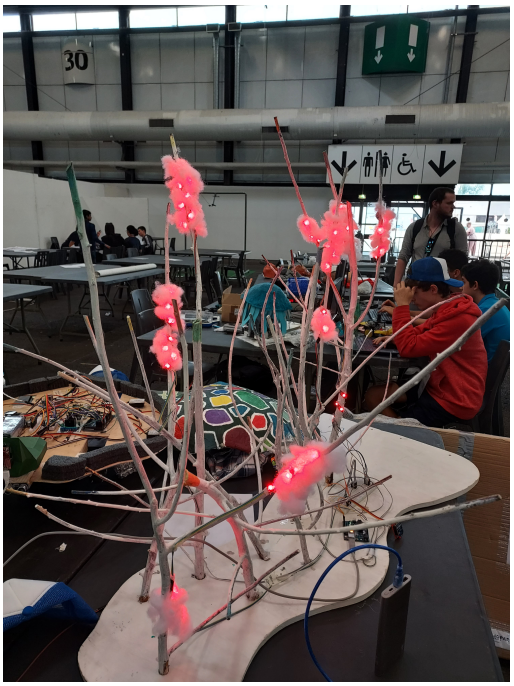
Photos and Images of the robot(s):

If there is a design drawing of the robot, or if you have photos or notes of the development process, please provide them. Those will be useful to show and prove that the team's robots and designs are their own. If you are including photos or documents, please ensure that they fit within five sheets of A4 size paper.




Scenario 1 USS
Tortue cherche
Tortue trouve
Tortue se rapproche
USS predit collision
Tortue stop
Scenario 2 Gestes
Tortue cherche
Tortue trouve
Tortue se rapproche
Collision USS
Tortue stop
Scenario 3
Tortue cherche
Tortue trouve
Tortue se rapproche
USS predit collision
Tortue stop


Connections
Arduino Mega
Servos:
HG
HD
BG
BD
Camera
Serial
Nano


30

Main code for robot(s):

Please attach the latest version of your code for each of the robot(s). The code is allowed to be modified after submission and will not be used during the judging process, only to inform the judges of the team's skill level and the programming language.

Code for the displacement of the robot in Arduino :

```
#include <Wire.h>
#include <Servo.h>

// Initialise le servomoteur de chaque patte et la queue de la tortue
Servo paAG, paAD, paBG, paBD, paQU;

// Pins moteurs            1   2   3   4
const int in1[] = {22, 24, 26, 28};
const int in2[] = {23, 25, 27, 29};
const int ena[] = {2,  3,  4,  5};

////////////////////////////// Fonctions supplémentaires ////////////////////////////////////////

// Fonction pour faire se déplacer la tortue
// Chaque moteur est controlé avec 3 connections :
// in1 et in2 controllent la direction du moteur (respectivement HIGH LOW pour avancer
// LOW HIGH pour reculer, HIGH HIGH pour freiner brutalement, LOW LOW pour ne rien faire)
// enfin la connection ena s'occupe de la vitesse, 0 étant l'arrêt, 255 la vitesse max

void moteurs(String dir, int vit){
  if(analogRead(A0) <= 500)
  {
        // Pour avancer
        if(dir == "av"){
        // Moteurs : + + + +
        for(int i=0; i<4;i++)
        {
        digitalWrite(in1[i], HIGH);
        digitalWrite(in2[i], LOW);
        // Vitesse des moteurs
        analogWrite(ena[i], vit);
        } return;
        }

        // Pour reculer
        if(dir == "re"){
        // Moteurs : - - - -
```

```
for(int i=0; i<4;i++)
{
digitalWrite(in1[i], LOW);
digitalWrite(in2[i], HIGH);
// Vitesse des moteurs
analogWrite(ena[i], vit);
} return;
}

// Pour tourner clockise (sens aiguille montre)
if(dir == "td"){
// Moteurs : + - + -
digitalWrite(in1[0], HIGH); // M1 +
digitalWrite(in2[0], LOW);
digitalWrite(in1[1], LOW);  // M2 -
digitalWrite(in2[1], HIGH);
digitalWrite(in1[2], HIGH); // M3 +
digitalWrite(in2[2], LOW);
digitalWrite(in1[3], LOW);  // M4 -
digitalWrite(in2[3], HIGH);
// Vitesse des moteurs
for(int i=0; i<4;i++) {
analogWrite(ena[i], vit);
} return;
}

// Pour tourner counter clockwise (sens inverse aiguille montre)
if(dir == "tg"){
// Moteurs : - + - +
digitalWrite(in1[0], LOW);  // M1 -
digitalWrite(in2[0], HIGH);
digitalWrite(in1[1], HIGH); // M2 +
digitalWrite(in2[1], LOW);
digitalWrite(in1[2], LOW);  // M3 -
digitalWrite(in2[2], HIGH);
digitalWrite(in1[3], HIGH); // M4 +
digitalWrite(in2[3], LOW);
// Vitesse des moteurs
for(int i=0; i<4;i++) {
analogWrite(ena[i], vit);
} return;
}

// Pour se déplacer à gauche
if(dir == "gg"){
// Moteurs : - + + -
digitalWrite(in1[0], LOW); // M1 -
digitalWrite(in2[0], HIGH);
digitalWrite(in1[1], HIGH); // M2 +
digitalWrite(in2[1], LOW);
digitalWrite(in1[2], HIGH); // M3 +
digitalWrite(in2[2], LOW);
digitalWrite(in1[3], LOW); // M4 -
digitalWrite(in2[3], HIGH);
// Vitesse des moteurs
for(int i=0; i<4;i++) {
analogWrite(ena[i], vit);
} return;
}

// Pour se déplacer à droite
if(dir == "dd"){
// Moteurs : + - - +
digitalWrite(in1[0], HIGH); // M1 +
digitalWrite(in2[0], LOW);

digitalWrite(in1[1], LOW); // M2 -
digitalWrite(in2[1], HIGH);

digitalWrite(in1[2], LOW); // M3 -
digitalWrite(in2[2], HIGH);

digitalWrite(in1[3], HIGH); // M4 +
digitalWrite(in2[3], LOW);
```

```
            // Vitesse des moteurs
            for(int i=0; i<4;i++) {
            analogWrite(ena[i], vit);
            } return;
            }

            // Pour avancer en diagonale haut gauche
            if(dir == "hg"){
            // Moteurs : 0 + + 0
            for(int i=0; i<4;i++)
            {
            digitalWrite(in1[i], HIGH);
            digitalWrite(in2[i], LOW);
            }
            analogWrite(ena[0], 0);
            analogWrite(ena[1], vit);
            analogWrite(ena[2], vit);
            analogWrite(ena[3], 0);
            return;
            }

            // Pour avancer en diagonale haut droit
            if(dir == "hd"){
            // Moteurs : + 0 0 +
            for(int i=0; i<4;i++)
            {
            digitalWrite(in1[i], HIGH);
            digitalWrite(in2[i], LOW);
            }
            analogWrite(ena[0], vit);
            analogWrite(ena[1], 0);
            analogWrite(ena[2], 0);
            analogWrite(ena[3], vit);
            return;
            }

            // Pour avancer en diagonale bas gauche
            if(dir == "bg"){
            // Moteurs : - 0 0 -
            for(int i=0; i<4;i++)
            {
            digitalWrite(in1[i], LOW);
            digitalWrite(in2[i], HIGH);
            }
            analogWrite(ena[0], vit);
            analogWrite(ena[1], 0);
            analogWrite(ena[2], 0);
            analogWrite(ena[3], vit);
            return;
            }

            // Pour avancer en diagonale bas droit
            if(dir == "bd"){
            // Moteurs : - 0 0 -
            for(int i=0; i<4;i++)
            {
            digitalWrite(in1[i], LOW);
            digitalWrite(in2[i], HIGH);
            }
            analogWrite(ena[0], 0);
            analogWrite(ena[1], vit);
            analogWrite(ena[2], vit);
            analogWrite(ena[3], 0);
            }
  } else {
            Serial.println("Stop");
  }
}


// Vérifie l'état du Serial1 (caméra) pendant forMillis millisecondes
int checkSerial(int forMillis, int moteur_active){
  long tzero = millis();
  int didrecieve = 0;
```

```
 // Tant que le délai n'est pas dépassé ou qu'un message est recu dans le Serial1 (caméra)
 while(millis()-tzero < forMillis && didrecieve == 0){
                // Si Serial1 (caméra) a envoyé des informations
                if(Serial1.available()>0){
                String message = "";
                //Récupère tous les caractères et les concatène dans "message"
                while(Serial1.available()>0){
                char c = Serial1.read();
                message.concat(c);
                }
                // Debug
                Serial.println(message);
                // Si on a demandé à la fonction de déplacer les moteurs
                if(moteur_active == 1){
                // 1 -> Tourne dans le sens inverse des aiguilles d'une
                // 2 -> Tourne dans le sens des aiguilles d'une
                // 3 -> Avance (l'objet détecté est au centre le l'image)
                if(message.indexOf("1")!=-1){
                moteurs("tg", 50);
                } else if(message.indexOf("2")!=-1){
                moteurs("td", 50);
                } else if(message.indexOf("3")!=-1){
                moteurs("av", 100);
                }
                }
                // Mesure de sécurité pour éviter les messages vides
                if(message.length()>0){
                didrecieve = 1;
                }
                message = "";
                }
                delay(10);
 }
 delay(50);
 // Check Debug
 Serial.println("rien");
 return didrecieve;
}


// Fonction qui permet à la caméra de balayer la scène pour trouver des objets (méduse ou sac)
int turnTime = 1000;
int waitTime = 400;
int ampliPat = 20;
int ampliAvan = 25;
void scanne(){
 if(digitalRead(A1) >= 800){
                turnTime = 2500;
                waitTime = 400;
 }
 int atrouve = 0;
 // Utilise while et des if pour réagir le plus vite et ne pas perdre de temps
 // Appelle la fonction "checkSerial" comme équivalent de "delay()" mais qui permet de se terminer
 // dès qu'on recoit un message dans le Serial1 pour finir "scanne" le plus vite possible
/*
 paAD.write(90+ampliAvan);
 paBD.write(90+ampliPat);
 moteurs("tg", 150);
 delay(turnTime);
 moteurs("tg", 0);
 paAD.write(90);
 paBD.write(90);
 delay(waitTime);

 paAG.write(90-ampliAvan);
 paBG.write(90-ampliPat);
 moteurs("td", 150);
 delay(turnTime);
 paAG.write(90);
 paBG.write(90);
 moteurs("td", 0);
 delay(waitTime);
 paAG.write(90-ampliAvan);
```

```
  paBG.write(90-ampliPat);
  moteurs("td", 150 );
  delay(turnTime);
  paAG.write(90);
  paBG.write(90);
  moteurs("td", 0);
  delay(waitTime);
  paAD.write(90-ampliAvan);
  paBD.write(90-ampliPat);
  moteurs("tg", 150 );
  delay(turnTime);
  paAD.write(90);
  paBD.write(90);
  moteurs("tg", 0);
  delay(waitTime);*/

  while(atrouve==0){
                // Regarde a gauche
                if(atrouve==0){
                paAD.write(90+ampliAvan);
                paBD.write(90+ampliPat);
                moteurs("tg", 150);
                atrouve = checkSerial(turnTime, 0);
                }
                if(atrouve==0){
                moteurs("tg", 0);
                paAD.write(90);
                paBD.write(90);
                atrouve = checkSerial(waitTime, 0);
                }
                // Reviens
                if(atrouve==0){
                paAG.write(90-ampliAvan);
                paBG.write(90-ampliPat);
                moteurs("td", 150);
                atrouve = checkSerial(turnTime, 0);
                }
                if(atrouve==0){
                paAG.write(90);
                paBG.write(90);
                moteurs("td", 0);
                atrouve = checkSerial(waitTime, 0);
                }
                // Regarde a droite
                if(atrouve==0){
                paAG.write(90-ampliAvan);
                paBG.write(90-ampliPat);
                moteurs("td", 150 );
                atrouve = checkSerial(turnTime, 0);
                }
                if(atrouve==0){
                paAG.write(90);
                paBG.write(90);
                moteurs("td", 0);
                atrouve = checkSerial(waitTime, 0);
                }
                // Reviens
                if(atrouve==0){
                paAD.write(90+ampliAvan);
                paBD.write(90+ampliPat);
                moteurs("tg", 150 );
                atrouve = checkSerial(turnTime, 0);
                }
                if(atrouve==0){
                paAD.write(90);
                paBD.write(90);
                moteurs("tg", 0);
                atrouve = checkSerial(waitTime, 0);
                }
                Serial.println("Recherche");
  }
  return;
}
```

```
// Similaire à checkSerial mais ne s'éxecute pas pour un temps donné, le test se fait jusqu'a ce
// qu'il n'y ait plus de messages durant un temps donné "threshold"
void suis_camera(){
  long compteur = 0;
  long threshold = 1000;
  while(compteur < threshold && analogRead(A2) <= 500){
                if(Serial1.available()>0){
                String message = "";
                while(Serial1.available()>0){
                char c = Serial1.read();
                message.concat(c);
                }

                Serial.println(message);
                // 1 -> Tourne dans le sens inverse des aiguilles d'une
                // 2 -> Tourne dans le sens des aiguilles d'une
                // 3 -> Avance (l'objet détecté est au centre le l'image)
                if(message.indexOf("1")!=-1){
                Serial.println("Gauche");
                moteurs("tg", 80);
                delay(100);
                } else if(message.indexOf("2")!=-1){
                Serial.println("Droite");
                moteurs("td", 80);
                delay(100);
                } else if(message.indexOf("3")!=-1){
                Serial.println("avance");
                moteurs("av", 200);
                delay(100);
                } else {
                Serial.println("Message pas compris");
                moteurs("av", 0);
                }
                compteur = 0;
                } else {
                delay(1);
                compteur ++;
                //Serial.println(compteur);
                }
  }
  if(analogRead(A2) >= 500){
                Serial.println("Obstacle");
  }
  moteurs("av", 0);
  Serial.println("finSuivicamera");
  return;
}

////////////////////////////////////// Fonctions principales  //////////////////////////////////////

void setup() {
  Serial.begin(38400); // Juste pour le débug

  Serial1.begin(9600); // Camera, TX en 18 et RX en 19
                // UART(3): (TX, RX) = (P4, P5)
                // Donc connecter 18-P5 et 19-P4
  Serial2.begin(9600); // Coraux, TX en 16, RX en 17
                // Xbee : Dout et Din
                // Donc connecter 16-Din et 17-Dout

  pinMode(A0, INPUT);
  pinMode(A1, INPUT);
  pinMode(A2, INPUT);

  // Un objet pour chaque servomoterus, simple etèfiable
  paAG.attach(12);
  paAD.attach(9);
  paBG.attach(11);
  paBD.attach(10);
  paQU.attach(13);

  for(int i=0;i<4;i++)
  {
```

```
            // Initialisation des pins pour chaque moteur,
            // pas besoin de modifier ça si on change les pins moteurs
            pinMode(in1[i], OUTPUT);
            pinMode(in2[i], OUTPUT);
            pinMode(ena[i], OUTPUT);
  }
  // Debug
  Serial.println("Pins ok");

  // Initialisation à 90 degrés de chaque servo
  paAG.write(90);
  delay(200);
  paAD.write(90);
  delay(200);
  paBG.write(90);
  delay(200);
  paBD.write(90);
  delay(200);
  paQU.write(90);
  delay(200);

  // Debug
  Serial.println("Lance loop");

  // Montre les mecanum



  //delay(30*1000);
}

void loop() {
  // Attend une détection de la caméra
  scanne();
  Serial.println("Fini scan");

  // Suis les instructions de la caméra
  suis_camera();
  Serial.print("Fini suivi");

  moteurs("re", 180);
  delay(500);
  moteurs("re", 0);
  delay(10);
  moteurs("re", 0);

  // Attend 10 secondes avant de recommencer un scan
  delay(10*1000);
  turnTime = 2500;
  waitTime = 400;
}
```

Code for the recognition of face and color in microPython:

```
import sensor, time, image, math,pyb
from pyb import UART
from machine import Pin
#Configuration arduino with 9600  baudrates
uart = UART(3, 9600)

#Camera settings
sensor.reset()
sensor.set_contrast(3)
sensor.set_gainceiling(16)
sensor.set_framesize(sensor.HQVGA)
sensor.set_pixformat(sensor.RGB565)

sensor.skip_frames(time = 2000)

#Face detection function
#return a 1 if a face is detected and a 0 if not
def visage_detect():
```

```
        global face_cascade, eyes_cascade, count
        img = sensor.snapshot()
        objects = img.find_features(face_cascade, threshold=0.75, scale_factor=1.25)
        eyes = None
        for face in objects:
        img.draw_rectangle(face)
        eyes = img.find_features(eyes_cascade, threshold=0.85, scale_factor=1.2, roi=face)
        for e in eyes:
        count = 1
        img.draw_rectangle(e)
        if count == 1:
        return 1
        return 0

#Function that detect if their is a tention send to the pin 6
def detect():
        global count
        p0 = Pin("P6", Pin.IN, Pin.PULL_UP)
        if p0.value() >= 0.6:
        count = 1
        return 1
        return 0

#Function that detect a color and return the location coordinates
def robot(x):
        thresholds = [(43, 89, -30, 11, 44, 78),(16, 63, -16, 4, -43, -31)]
        led[x].on()
        img = sensor.snapshot()
        # Analyse the color by their threshlds in the whole image (58, 96, -38, 15, 36, 81),(55, 100, -39, -12, -71, 1),(36, 75, -24, 0, -54, -21)
        for blob in img.find_blobs([thresholds[x]], pixels_threshold=80, area_threshold=40, merge=True):
        # Display the object detected on the IDE
        if blob.elongation() > 0.5:
        img.draw_edges(blob.min_corners(), color=(255,0,0))
        img.draw_line(blob.major_axis_line(), color=(0,255,0))
        img.draw_line(blob.minor_axis_line(), color=(0,0,255))
        img.draw_rectangle(blob.rect())
        img.draw_cross(blob.cx(), blob.cy())
        img.draw_keypoints([(blob.cx(), blob.cy(), int(math.degrees(blob.rotation())))], size=20)
        # Set the coordinates (0;0) in the center of the screen
        pos = (blob.cx() - img.width()//2, blob.cy() - img.height()//2)
        return pos

time.sleep(2)
level = 0 # 0=Initial state / 1=face is detected
count = 0
led = [pyb.LED(1),pyb.LED(2)] #pyb.LED(1) = red and pyb.LED(2) = yellow
face_cascade = image.HaarCascade("frontalface", stages=25)
eyes_cascade = image.HaarCascade("eye", stages=24)

# Program launch loop
while True:
        level = visage_detect()
        if level == 0:
        level = detect()
        pos = robot(level) # coordonates
        if pos : # If their is an object it send information to the Arduino for deplacement
        print(pos)
        if pos[0] < -40:
        print("Go__left")
        uart.write("1111")

        if pos[0] > 40 :
        print("Go_right")
        uart.write("2222")

        if pos[0] > -40 and pos[0] < 40:
        print("GOGOGOGO")
        uart.write("3333")
```