

```

1 #include <AFMotor.h>
2 AF_DCMotor moteur3(1);
3 AF_DCMotor moteur2(2);
4
5 #define trigPinAvant 9 // defini les broches du capteur central Avant
6 #define echoPinAvant 10
7
8 #define trigPinDroit 45 // defini les broches du capteur DROIT
9 #define echoPinDroit 44
10
11 #define trigPinGauche 43 // defini les broches du capteur GAUCHE
12 #define echoPinGauche 42
13
14 void setup() {
15   Serial.begin(9600); // pour afficher des données à l'écran si besoin
16
17   pinMode(trigPinAvant, OUTPUT); // envoie les ultra-sons AVANT CENTRAL
18   pinMode(echoPinAvant, INPUT); // reçoit l'écho AVANT CENTRAL
19
20   pinMode(trigPinDroit, OUTPUT); // envoie les ultra-sons DROIT
21   pinMode(echoPinDroit, INPUT); // reçoit l'écho DROIT
22
23   pinMode(trigPinGauche, OUTPUT); // envoie les ultra-sons GAUCHE
24   pinMode(echoPinGauche, INPUT); // reçoit l'écho GAUCHE
25
26   moteur3.run(RELEASE);
27   moteur2.run(RELEASE);
28
29   // initialise les broches Analog In A0 en pin OUTPUT pour faire du son
30   pinMode(A0, OUTPUT);
31
32   delay(3000); // POUR PAS QUE LE ROBOT DEBARASSE DES LA MISE sous
33   tension
34 }
35
36 for (int compteur=0; compteur<100; compteur=compteur+1) // ALARME
37 {
38   // SONDRE de démarrage
39   digitalWrite(A0, 255);
40   delay(1);
41   digitalWrite(A0, LOW);
42   delay(1);
43   digitalWrite(A0, 255);
44   delay(1);
45   digitalWrite(A0, LOW);
46   delay(1);
47 }
48
49 void loop() {
50   long duration, distance; // MESURE OBSTACLE FRONTAL AVANT
51   digitalWrite(trigPinAvant, LOW);
52   delayMicroseconds(2);
53   digitalWrite(trigPinAvant, HIGH);
54   delayMicroseconds(10);
55   digitalWrite(trigPinAvant, LOW);
56   duration = pulseIn(echoPinAvant, HIGH);
57   distance = (duration/2) / 29.1; // converti la distance en cm
58
59   if (distance < 40) /* TESTE SI OBSTACLE FRONTAL AVANT A MOINS DE 10CM */
60   {
61     Serial.println("DETECTION OBSTACLE FRONTAL"); // si besoin d'afficher
62     infos
63     Serial.print("DISTANCE ");
64     Serial.print ( distance);
65     Serial.print ( " CM");
66     Serial.println ("CORRIGER TRAJECTOIRE");
67     moteur3.run(RELEASE);
68     moteur2.run(RELEASE);
69
70   }
71
72   // SONDRE OBSTACLE
73   digitalWrite(A0, 255);
74   delay(1);
75   digitalWrite(A0, LOW);
76   delay(1);
77   digitalWrite(A0, 255);
78   delay(1);
79   digitalWrite(A0, LOW);
80   delay(1);
81 }
82
83 delay(5);
84 long duration, distance; // MESURE SI OBSTACLE A DROITE
85 digitalWrite(trigPinDroit, LOW);
86 delayMicroseconds(2);
87 digitalWrite(trigPinDroit, HIGH);
88 delayMicroseconds(10);
89 digitalWrite(trigPinDroit, LOW);
90 duration = pulseIn(echoPinDroit, HIGH);
91 distance = (duration/2) / 29.1; // converti la distance en cm
92
93 if (distance < 40) /* TESTE SI OBSTACLE A DROITE A MOINS DE 10CM */
94 {
95   Serial.println("DETECTION OBSTACLE à droite");
96   Serial.print ("DISTANCE ");
97   Serial.print ( distance);
98   Serial.print ( " CM");
99   Serial.println ("NE PAS TOURNER A DROITE");
100   delay(500);
101
102   long duration, distance; // MESURE SI OBSTACLE A GAUCHE
103   digitalWrite(trigPinGauche, LOW);
104   delayMicroseconds(2);
105   digitalWrite(trigPinGauche, HIGH);
106   delayMicroseconds(10);
107 }
108
109 // MESURE OBSTACLE A GAUCHE AVEC AFFECTATION COEFF MOTEUR DROITS
110 if (CH0)
111 {
112   delay(5);
113   //long duration, distance; // MESURE OBSTACLE A GAUCHE
114   digitalWrite(trigPinGauche, LOW);
115   delayMicroseconds(2);
116   digitalWrite(trigPinGauche, HIGH);
117   delayMicroseconds(10);
118   digitalWrite(trigPinGauche, LOW);
119   duration = pulseIn(echoPinGauche, HIGH);
120   distance = (duration/2) / 29.1; // converti la distance en cm
121   if (distance > 13) {
122     moteur3.run(FORWARD);
123     moteur2.setSpeed(200);
124   }
125   else {
126     float CH0 = ((4+(distance-7))/10);
127     moteur3.run(FORWARD);
128     moteur2.setSpeed(200*CH0);
129   }
130 }
131
132 // MESURE OBSTACLE A GAUCHE AVEC AFFECTATION COEFF MOTEUR GAUCHE
133 if (CH1)
134 {
135   delay(5);
136   //long duration, distance; // MESURE OBSTACLE A GAUCHE
137   digitalWrite(trigPinGauche, LOW);
138   delayMicroseconds(2);
139   digitalWrite(trigPinGauche, HIGH);
140   delayMicroseconds(10);
141   digitalWrite(trigPinGauche, LOW);
142   duration = pulseIn(echoPinGauche, HIGH);
143   distance = (duration/2) / 29.1; // converti la distance en cm
144   if (distance > 13) {
145     moteur3.run(FORWARD);
146     moteur2.setSpeed(200);
147   }
148   else {
149     float CH1 = ((4+(distance-7))/10);
150     moteur3.run(FORWARD);
151     moteur2.setSpeed(200*CH1);
152   }
153 }
154
155 // MESURE OBSTACLE A DROITE AVEC AFFECTATION COEFF MOTEUR GAUCHE
156 if (CH2)
157 {
158   delay(5);
159   //long duration, distance; // MESURE OBSTACLE A DROITE
160   digitalWrite(trigPinDroit, LOW);
161   delayMicroseconds(2);
162   digitalWrite(trigPinDroit, HIGH);
163   delayMicroseconds(10);
164   digitalWrite(trigPinDroit, LOW);
165   duration = pulseIn(echoPinDroit, HIGH);
166   distance = (duration/2) / 29.1; // converti la distance en cm
167   if (distance > 13) {
168     moteur3.run(FORWARD);
169     moteur2.setSpeed(200);
170   }
171   else {
172     float CH2 = ((4+(distance-7))/10);
173     moteur3.run(FORWARD);
174     moteur2.setSpeed(200*CH2);
175   }
176 }
177
178 // MESURE OBSTACLE A DROITE AVEC AFFECTATION COEFF MOTEUR DROITS
179 if (CH3)
180 {
181   delay(5);
182   //long duration, distance; // MESURE OBSTACLE A DROITE
183   digitalWrite(trigPinDroit, LOW);
184   delayMicroseconds(2);
185   digitalWrite(trigPinDroit, HIGH);
186   delayMicroseconds(10);
187   digitalWrite(trigPinDroit, LOW);
188   duration = pulseIn(echoPinDroit, HIGH);
189   distance = (duration/2) / 29.1; // converti la distance en cm
190   if (distance > 13) {
191     moteur3.run(FORWARD);
192     moteur2.setSpeed(200);
193   }
194   else {
195     float CH3 = ((4+(distance-7))/10);
196     moteur3.run(FORWARD);
197     moteur2.setSpeed(200*CH3);
198   }
199 }
200
201 // MESURE OBSTACLE A DROITE AVEC AFFECTATION COEFF MOTEUR GAUCHE
202 if (CH4)
203 {
204   delay(5);
205   //long duration, distance; // MESURE OBSTACLE A DROITE
206   digitalWrite(trigPinDroit, LOW);
207   delayMicroseconds(2);
208   digitalWrite(trigPinDroit, HIGH);
209   delayMicroseconds(10);
210   digitalWrite(trigPinDroit, LOW);
211   duration = pulseIn(echoPinDroit, HIGH);
212   distance = (duration/2) / 29.1; // converti la distance en cm
213   if (distance > 13) {
214     moteur3.run(FORWARD);
215     moteur2.setSpeed(200);
216   }
217   else {
218     float CH4 = ((4+(distance-7))/10);
219     moteur3.run(FORWARD);
220     moteur2.setSpeed(200*CH4);
221   }
222 }
223
224 // MESURE OBSTACLE A DROITE AVEC AFFECTATION COEFF MOTEUR DROITS
225 if (CH5)
226 {
227   delay(5);
228   //long duration, distance; // MESURE OBSTACLE A DROITE
229   digitalWrite(trigPinDroit, LOW);
230   delayMicroseconds(2);
231   digitalWrite(trigPinDroit, HIGH);
232   delayMicroseconds(10);
233   digitalWrite(trigPinDroit, LOW);
234   duration = pulseIn(echoPinDroit, HIGH);
235   distance = (duration/2) / 29.1; // converti la distance en cm
236   if (distance > 13) {
237     moteur3.run(FORWARD);
238     moteur2.setSpeed(200);
239   }
240   else {
241     float CH5 = ((4+(distance-7))/10);
242     moteur3.run(FORWARD);
243     moteur2.setSpeed(200*CH5);
244   }
245 }
246
247 // MESURE OBSTACLE A DROITE AVEC AFFECTATION COEFF MOTEUR GAUCHE
248 if (CH6)
249 {
250   delay(5);
251   //long duration, distance; // MESURE OBSTACLE A DROITE
252   digitalWrite(trigPinDroit, LOW);
253   delayMicroseconds(2);
254   digitalWrite(trigPinDroit, HIGH);
255   delayMicroseconds(10);
256   digitalWrite(trigPinDroit, LOW);
257   duration = pulseIn(echoPinDroit, HIGH);
258   distance = (duration/2) / 29.1; // converti la distance en cm
259   if (distance > 13) {
260     moteur3.run(FORWARD);
261     moteur2.setSpeed(200);
262   }
263   else {
264     float CH6 = ((4+(distance-7))/10);
265     moteur3.run(FORWARD);
266     moteur2.setSpeed(200*CH6);
267   }
268 }
269
270 // MESURE OBSTACLE A DROITE AVEC AFFECTATION COEFF MOTEUR DROITS
271 if (CH7)
272 {
273   delay(5);
274   //long duration, distance; // MESURE OBSTACLE A DROITE
275   digitalWrite(trigPinDroit, LOW);
276   delayMicroseconds(2);
277   digitalWrite(trigPinDroit, HIGH);
278   delayMicroseconds(10);
279   digitalWrite(trigPinDroit, LOW);
280   duration = pulseIn(echoPinDroit, HIGH);
281   distance = (duration/2) / 29.1; // converti la distance en cm
282   if (distance > 13) {
283     moteur3.run(FORWARD);
284     moteur2.setSpeed(200);
285   }
286   else {
287     float CH7 = ((4+(distance-7))/10);
288     moteur3.run(FORWARD);
289     moteur2.setSpeed(200*CH7);
290   }
291 }
292
293 // MESURE OBSTACLE A DROITE AVEC AFFECTATION COEFF MOTEUR GAUCHE
294 if (CH8)
295 {
296   delay(5);
297   //long duration, distance; // MESURE OBSTACLE A DROITE
298   digitalWrite(trigPinDroit, LOW);
299   delayMicroseconds(2);
300   digitalWrite(trigPinDroit, HIGH);
301   delayMicroseconds(10);
302   digitalWrite(trigPinDroit, LOW);
303   duration = pulseIn(echoPinDroit, HIGH);
304   distance = (duration/2) / 29.1; // converti la distance en cm
305   if (distance > 13) {
306     moteur3.run(FORWARD);
307     moteur2.setSpeed(200);
308   }
309   else {
310     float CH8 = ((4+(distance-7))/10);
311     moteur3.run(FORWARD);
312     moteur2.setSpeed(200*CH8);
313   }
314 }
315
316 // MESURE OBSTACLE A DROITE AVEC AFFECTATION COEFF MOTEUR DROITS
317 if (CH9)
318 {
319   delay(5);
320   //long duration, distance; // MESURE OBSTACLE A DROITE
321   digitalWrite(trigPinDroit, LOW);
322   delayMicroseconds(2);
323   digitalWrite(trigPinDroit, HIGH);
324   delayMicroseconds(10);
325   digitalWrite(trigPinDroit, LOW);
326   duration = pulseIn(echoPinDroit, HIGH);
327   distance = (duration/2) /
```

# CODE DE RECONNAISSANCE FACIALE EN PYTHON CORRELE AVEC ARDUINO

(Le code résulte d'un travail en commun avec le robot Militech.)

```
1 import numpy as np
2 import serial
3 import time
4 import cv2
5
6 # Setup Communication path for arduino (in place of 'COM')
7 arduino = serial.Serial('COM', 9600)
8 time.sleep(1)
9 print("Connected to arduino...")
10
11 # Importing the HaarCascade for face detection
12 face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
13
14 # To capture the video stream from webcam.
15 cap = cv2.VideoCapture(0)
16 cv2.namedWindow('img', cv2.WND_PROP_FULLSCREEN)
17
18 # Read the captured image, convert it to Gray image and find faces
19 while 1:
20     ret, img = cap.read()
21     cv2.resizeWindow('img', 500, 300)
22     cv2.line(img, (250, 250), (0, 250), (0, 255, 0), 1)
23     cv2.line(img, (250, 0), (250, 250), (0, 255, 0), 1)
24     cv2.circle(img, (250, 250), 5, (255, 255, 255), -1)
25     gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
26     faces = face_cascade.detectMultiScale(gray, 1.3)
```

```
30 # Detect the face and make a rectangle around it.
31 for (x,y,w,h) in faces:
32     cv2.rectangle(img,(x,y),(x+w,y+h),(0,255,0),5)
33     roi_gray = gray[y:y+h, x:x+w]
34     roi_color = img[y:y+h, x:x+w]
35
36     arr = (y,y+h, x,x+w)
37     print (arr)
38
39     print ('x : ' +str(x))
40     print ('y : ' +str(y))
41     print ('w : ' +str(w))
42     print ('h : ' +str(h))
43
44 # Center of roi (Rectangle)
45 xx = int((x+w)/2)
46 yy = int((y+h)/2)
47 print (xx)
48 print (yy)
49 center = (xx,yy)
50
51 # sending data to arduino
52 print("Center of Rectangle is :", center)
53 data = "X[0:0]Y[1:0]".format(xx, yy)
54 print ("Output = " +data)
55 arduino.write(data)
56
57 # Display the stream.
58 cv2.imshow('img',img)
59
60 # Wait 'fac' to terminate execution
61 k = cv2.waitKey(30) & 0xff
62 if k == 27:
63     break
```

```
1 #include<Servo.h>
2
3 Servo servoVer; //Vertical Servo
4 Servo servoHor; //Horizontal Servo
5
6 int x;
7 int y;
8
9 int prevX;
10 int prevY;
11
12 void setup()
13 {
14     Serial.begin(9600);
15     servoVer.attach(5); //Attach Vertical Servo to Pin 5
16     servoHor.attach(6); //Attach Horizontal Servo to Pin 6
17     servoVer.write(90);
18     servoHor.write(90);
19 }
20
21 void Pos()
22 {
23     if(prevX != x || prevY != y)
24     {
25         int servoX = map(x, 600, 0, 70, 179);
26         int servoY = map(y, 450, 0, 179, 95);
27
28         servoX = min(servoX, 179);
29         servoX = max(servoX, 70);
30         servoY = min(servoY, 179);
31         servoY = max(servoY, 95);
```

```
32
33     servoHor.write(servoX);
34     servoVer.write(servoY);
35 }
36
37 void loop()
38 {
39     if(Serial.available() > 0)
40     {
41         if(Serial.read() == 'X')
42         {
43             x = Serial.parseInt();
44             if(Serial.read() == 'Y')
45             {
46                 y = Serial.parseInt();
47                 Pos();
48             }
49         }
50     }
51     while(Serial.available() > 0)
52     {
53         Serial.read();
54     }
55 }
56
57
58
```