



# FILIÈRE LPH3

PROJET D'ACTIVITÉ  
PRATIQUE

**EAGLESIGHT**

RÉALISER PAR

**BENAMOR BRAHIM**

ANNEE  
UNIVERSITAIRE

**2022 - 2023**

# Table de matière

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <i>Hardware</i>   | <b>4</b>  |
| 1.1      | <i>le Matériel</i>  | 5         |
| 1.1.1    | Carte Arduino Uno Dip REV3  | 5         |
| 1.1.2    | Module ESP32 S-CAM : Carte Développement Bluetooth WiFi muni<br>d'une Camera OV2640 avec port micro USB | 6         |
| 1.1.3    | Module capteur ultrason HC-SR04   | 7         |
| 1.1.4    | Pont H double L298N   | 8         |
| 1.1.5    | Moteur plat Sankyo  | 9         |
| 1.1.6    | Buzzer passif   | 9         |
| 1.1.7    | Module feu tricolore à LED  | 10        |
| 1.1.8    | Pile Li-ion 18650   | 11        |
| 1.2      | <i>Construction</i>   | 12        |
| <b>2</b> | <i>Software</i>   | <b>14</b> |
| 2.1      | <i>Structure générale et principe de fonctionnement</i>   | 14        |
| 2.2      | <i>Le Programme</i>   | 15        |
| <b>3</b> | <i>Les perspectives</i>   | <b>25</b> |
| 3.1      | <i>La reconnaissance faciale</i>  | 25        |
| 3.2      | <i>Intégration des polycapteurs</i>   | 26        |
| 3.3      | <i>Stockage du données</i>  | 26        |

# List of Figures

|     |  |    |
|-----|--|----|
| 1.1 | Carte Arduino Uno Dip REV3 . . . . .         | 5  |
| 1.2 | Module ESP32 S-CAM . . . . .                 | 6  |
| 1.3 | Module capteur ultrason HC-SR04 . . . . .    | 7  |
| 1.4 | Pont H double L298N . . . . .                | 8  |
| 1.5 | Le moteur plat Sankyo . . . . .              | 9  |
| 1.6 | Buzzer passif . . . . .                      | 10 |
| 1.7 | Module feu tricolore à LED . . . . .         | 10 |
| 1.8 | Pile Li-ion 18650 . . . . .                  | 11 |
| 1.9 | Le montage du système "EagleSight" . . . . . | 13 |
| 3.1 | Capteur HC-SR510 . . . . .                   | 26 |
| 3.2 | Capteur détection d'obstacle . . . . .       | 26 |

# *Introduction*

Dans le monde d'aujourd'hui , la sécurité à domicile est une aspect essentiel de la propriété immobilière qui ne peut être ignoré . C'est une priorité absolue pour les propriétaires et les locataires .

L'une des façons les plus efficaces d'assurer la sécurité à domicile est d'avoir un système d'alarme de sécurité fiable en place . Investir dans l'un de haute qualité est une étape essentielle pour maintenir la sécurité et le bien-être de votre famille , ainsi que pour assurer que les biens de valeur sont protégés contre le vol .

De plus , les systèmes d'alarme de sécurité peuvent potentiellement réduire les primes d'assurance des propriétaires , offrir un accès à distance pour la surveillance et le contrôle via des applications pour smartphone ou d'autres appareils , et offrir des fonctionnalités avancées telles que la vidéo surveillance , la détection de mouvement et l'intégration de la maison intelligente .

Par conséquent , si vous souhaitez sécuriser votre domicile et protéger votre famille contre les intrusions potentielles , un système d'alarme de sécurité fiable est indispensable .

le rapport ci-dessous examinera les différentes étapes principales du fonctionnement de système d'alarme , y compris la conception et la construction ainsi l'installation



# Chapitre 1

## *Hardware*

On s'intéresse dans cette chapitre sur le matériel qu'on est utilisé : on donne la définition de chaque composant , leurs caractéristiques et leurs utilisation dans le cadre général ; puis on va discuter le montage à réaliser .



## 1.1 le Matériel

Soit le matériel utilisé dans notre projet ainsi leurs caractéristiques

### 1.1.1 Carte Arduino Uno Dip REV3

#### Définition

La carte Arduino Uno Dip REV3 est une carte de développement électronique open-source basée sur le microcontrôleur **ATmega328P**.

Elle est conçue pour faciliter la création de projets électroniques interactifs en offrant un environnement de développement convivial et facile à utiliser.

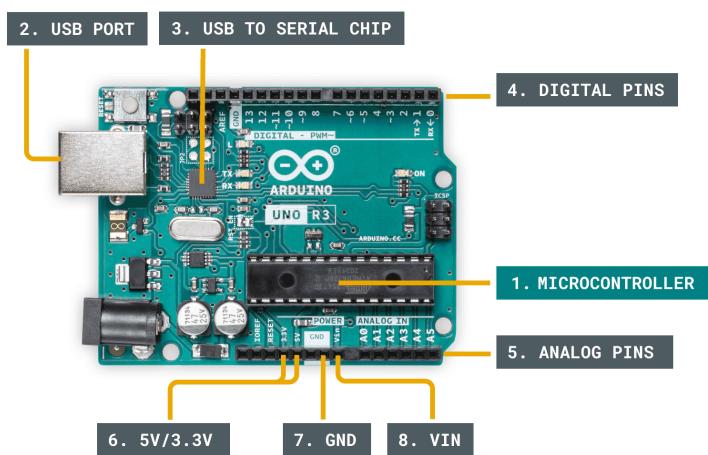


Figure 1.1: Carte Arduino Uno Dip REV3

#### Caractéristiques

Soit les principales caractéristiques de la carte Arduino Uno Dip REV3 :

- ♠ Microcontrôleur : ATmega328P
- ♠ Tension de fonctionnement : 5V
- ♠ Tension d'entrée (recommandée) : 7-12V
- ♠ Tension d'entrée (limite) : 6-20V
- ♠ Broches d'entrée/sortie numériques : 14 (dont 6 peuvent être utilisées en PWM)
- ♠ Broches d'entrée analogique : 6
- ♠ Courant continu par broche de sortie : 40 mA
- ♠ Courant continu pour 3.3V : 50 mA
- ♠ Mémoire flash : 32 KB (ATmega328P) dont 0.5 KB utilisés par le bootloader
- ♠ RAM : 2 KB (ATmega328P)
- ♠ EEPROM : 1 KB (ATmega328P)
- ♠ Fréquence d'horloge : 16 MHz

## L'utilité

Les cartes Arduino offrent une plateforme flexible et peu coûteuse pour les projets électroniques ; allant des projets simples de domotique et de capteurs à des projets plus complexes de robotique et d'automatisation industrielle .

### 1.1.2 Module ESP32 S-CAM : Carte Développement Bluetooth WiFi muni d'une Camera OV2640 avec port micro USB

#### Définition

L'ESP32 S-CAM est une carte de développement équipée du module ESP32 de la société Espressif . Elle est dotée de fonctionnalités Bluetooth et Wi-Fi pour permettre la communication sans fil avec d'autres appareils .

La caméra OV2640 est un module de caméra **CMOS** d'OmniVision Technologies qui est utilisé dans diverses applications d'imagerie . Elle est dotée d'un port micro-USB pour l'alimentation et la transmission de données.

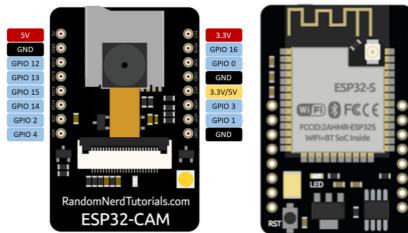


Figure 1.2: Module ESP32 S-CAM

#### Caractéristiques

Soit les principales caractéristiques de base du module ESP32 S-CAM :

- ♠ Module SOC WiFi 802.11b/g/n ultra petit + BT/BLE
- ♠ Processeur 32 bits double cœur basse consommation pour les processeurs d'application
- ♠ Jusqu'à 240 MHz, jusqu'à 600 DMIPS
- ♠ SRAM 520 Ko intégrée, externe 4M PSRAM
- ♠ Puissance de sortie : jusqu'à 19,5 dBm
- ♠ Prend en charge les interfaces telles que UART/SPI/I2C/PWM/ADC/DAC
- ♠ Compatible avec les caméras OV2640 et OV7670 avec flash intégré
- ♠ Prend en charge les images de téléchargement WiFi . Prise en charge de la carte TF . Prend en charge plusieurs modes de veille
- ♠ Lwip et FreeRTOS intégrés
- ♠ Compatible avec le mode de fonctionnement STA/AP/STA+AP
- ♠ Compatible avec le réseau de distribution Smart Config/AirKiss en un clic

♠ Prise en charge de la mise à niveau série locale et de la mise à niveau du micrologiciel à distance (FOTA)

♠ Prise en charge du développement secondaire

Soit les principales caractéristiques de base du Camera OV2640 avec port micro USB :

♠ Type de capteur : CMOS

♠ Résolution 2 mégapixels capable de prendre des photo et des vidéo avec une résolution de 1600x1200 pixels.

### L'utilité

Le module ESP32 S-CAM peut être programmé avec des plusieurs langages tels que **C** , **Lua** ou **Arduino IDE** , pour l'utiliser dans plusieurs domaines et projets : contrôle à distance ; robotique ; développement de prototypes ; surveillance à distance

### 1.1.3 Module capteur ultrason HC-SR04

#### Définition

Le module capteur ultrason **HC-SR04** est un capteur de distance qui utilise des ondes sonores pour mesurer la distance entre lui-même et un objet .



Figure 1.3: Module capteur ultrason HC-SR04

#### Caractéristiques

Spit les principales caractéristiques du module capteur ultrason HC-SR04 sont les suivantes :

♠ Tension d'alimentation : 5V DC

♠ Courant statique : moins de 2 mA

♠ Plage de mesure : de 2 cm à 450 cm

♠ Angle de détection : 15 degrés

♠ Signal de déclenchement d'entrée : impulsion TTL 10us

♠ Signal d'écho : Signal TTL de sortie TTL

- ♠ Fréquence de fonctionnement : 40 kHz
- ♠ Temps de réponse : 500 ms
- ♠ Consommation électrique : 15 mA

### L'utilité

Grâce à ses broches de connexion standard , le module capteur ultrason HC-SR04 est facile à utiliser et à intégrer dans des projets électroniques et robotiques pour détecter des obstacles ; mesurer des distances ou contrôler des mouvements .

#### 1.1.4 Pont H double L298N

##### Définition

Un pont H est un circuit électronique qui permet de contrôler la direction et la vitesse d'un moteur à courant continu en inversant le sens du courant électrique qui y circule .

Le double L298N est un circuit intégré très couramment utilisé pour construire des ponts H .

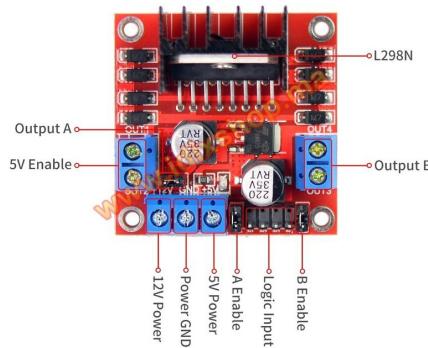


Figure 1.4: Pont H double L298N

##### Caractéristiques

Soit les caractéristiques principales du la pont H double L298N sont les suivantes :

- ♠ Tension logique : 5 V
- ♠ Tension d' entraînement : entre 5 V et 12 V
- ♠ Courant logique : entre 0 mA et 36 ma
- ♠ Courant d' entraînement : 2A (MAX single bridge)
- ♠ Puissance maximale : 25 W

### L'utilité

La pont H double L298N est largement utilisé dans les projets de robotique , car il offre une solution simple et efficace pour contrôler la vitesse et la direction des moteurs à courant continu .

### 1.1.5 Moteur plat Sankyo

#### Définition

Le moteur plat Sankyo est un type de moteur électrique qui se caractérise par un rotor plat et une stator plat .



Figure 1.5: Le moteur plat Sankyo

#### Caractéristiques

Soit les caractéristiques principales du Le moteur plat Sankyo sont les suivantes :

- ♠ Tension d'entrée entre 5 V et 12 V
- ♠ faible encombrement axial et une grande densité de puissance.

### L'utilité

Les moteurs plats Sankyo sont utilisés dans une variété d'applications , notamment dans les robots , les machines-outils , les équipements médicaux et les systèmes d'automatisation industrielle .

### 1.1.6 Buzzer passif

#### Définition

Un buzzer actif est un type de dispositif de **signalisation acoustique** qui produit un son lorsqu'il est alimenté en courant .

Contrairement à un buzzer passif qui nécessite une oscillation externe pour produire un son , un buzzer actif contient un oscillateur interne qui génère un signal audio lorsqu'il est alimenté en courant .



Figure 1.6: Buzzer passif

### Caractéristiques

Soit les caractéristiques principales du buzzer sont les suivantes :

- ♠ Tension : 5 v
- ♠ Courant: < 250mA

### L'utilité

Les buzzers actifs sont couramment utilisés dans les dispositifs électroniques pour fournir des alertes sonores , tels que les alarmes de sécurité , les avertissements de dysfonctionnements , les signaux sonores dans les jeux , les signaux sonores dans les jouets , etc.

### 1.1.7 Module feu tricolore à LED

#### Définition

Un module feu tricolore à LED est un dispositif électronique utilisé pour créer un signal de feu tricolore (rouge, jaune et vert) .



Figure 1.7: Module feu tricolore à LED

### Caractéristiques

Soit les caractéristiques principales du module feu tricolore à LED sont les suivantes :

- ♠ Faible consommation d'énergie
- ♠ Contrôle programmable

#### L'utilité

Généralement , il est utilisé pour réguler la circulation routière , ferroviaire ou piétonne .

#### 1.1.8 Pile Li-ion 18650

##### Définition

Une pile Li-ion 18650 rechargeable 1500mAh est une batterie rechargeable au **lithium-ion** avec une taille de 18mm x 65mm , qui est couramment utilisée dans les appareils électroniques tels que les ordinateurs portables , les lampes de poche , les drones , les caméras , etc .



Figure 1.8: Pile Li-ion 18650

##### Caractéristiques

Soit les caractéristiques principales du la pile Li-ion 18650 sont les suivantes :

- ♠ Capacité : 1500 mAh
- ♠ Voltage : 3.7 V
- ♠ Plage de température de fonctionnement : Décharge -20°C à +60°C ; Charge 0°C à 45°C

## 1.2 Construction

Pour obtenir une fonctionnement efficace du système , on suivi le démarche suivant du montage :

♠ Pour la carte Arduino :

- La pine 5V soit relier avec la ligne 1 du plaque d'essai .
- La pine GND soit relier avec la ligne 2 du plaque d'essai .
- La pine GND soit relier avec la pine RESET du carte Arduino .

♠ Pour le module ESP32-CAM :

- La pine 5V soit relier avec la ligne 1 du plaque d'essai .
- La pine GND soit relier avec la ligne 2 du plaque d'essai .
- La pine GPIO15 soit relier avec la ligne 5 du plaque d'essai .
- La pine GPIO14 soit relier avec la ligne 4 du plaque d'essai .
- La pine UNR soit relier avec la pine RX du carte Arduino .
- La pine UNT soit relier avec la pine TX du carte Arduino .
- La pine GND soit relier avec la ligne 2 du plaque d'essai .

♠ Pour la pente HC-SR04 :

- Elle est alimenté par une tension via 11.7 V , tel que les bornes d'entrée et du sortie soit respectivement : 12V relier avec le borne plus du support qui contient les trois piles d'alimentation et GND relier avec le borne négatif du support d'alimentation .
- La pine ENA soit relier avec la ligne 3 du plaque d'essai .
- la pine IN1 soit relier avec la ligne 4 du plaque d'essai .
- La pine IN2 soit relier avec la ligne 5 du plaque d'essai .

♠ module feu tricolore :

- La pine Y soit relier avec la ligne 4 du plaque d'essai .
- La pine R soit relier avec la ligne 5 du plaque d'essai .
- La pine GND soit relier avec la ligne 2 du plaque d'essai .

♠ Capteur ultrasons HC-SR04 : - La pine VCC 5V soit relier avec la ligne 1 du plaque d'essai .

- La pine GND soit relier avec la ligne 2 du plaque d'essai .
- La pine Echo soit relier avec la pine 6 du carte Arduino .
- La pine Trigger soit relier avec la pine 7 du carte Arduino .

♠ Le buzzer : - La pine positive soit relier avec la pine 8 du carte arduino .

- La pine négative soit relier avec la ligne 2 du plaque d'essai .

Soit le figure ci-dessous caractérise approximativement le montage résulte

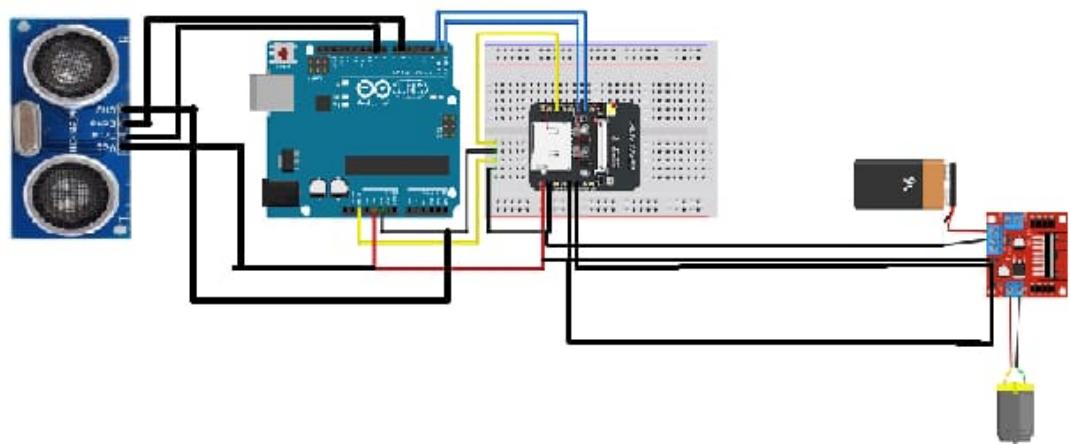


Figure 1.9: Le montage du système "EagleSight"

# Chapitre 2

## *Software*

### **2.1 Structure générale et principe de fonctionnement**

♣Ce code est un programme pour un dispositif qui utilise une caméra pour capturer des images et les envoyer via un serveur web . Le programme utilise une carte ESP32 et une caméra , et il est écrit en utilisant l'IDE Arduino .

♣Le programme commence par inclure les bibliothèques nécessaires pour la caméra , le Wi-Fi , le serveur HTTP , le convertisseur d'image et les bibliothèques Arduino . Ensuite , les identifiants du réseau Wi-Fi sont définis avec les constantes ssid et password .

♣La partie suivante du code définit les broches GPIO pour la caméra en fonction du modèle de caméra utilisé . Cette section utilise des directives préprocesseur #if defined pour sélectionner les broches en fonction du modèle de caméra .

♣Ensuite , les broches GPIO pour deux moteurs sont définies avec les constantes MOTOR\_1\_PIN\_1 et MOTOR\_1\_PIN\_2 .

♣La fonction setup() initialise le Wi-Fi et la caméra , en se connectant au réseau Wi-Fi et en initialisant la caméra avec les broches GPIO appropriées .

♣La fonction loop() est une boucle qui capture une image à partir de la caméra , la convertit en JPEG , l'envoie au serveur HTTP en tant que flux de données multiparties et fait tourner les moteurs .

♣La capture d'image est effectuée en appelant la fonction esp\_camera\_fb\_get() qui renvoie un pointeur vers un objet fb\_data\_t qui contient les données d'image . Ensuite , la fonction fb\_gfx\_convert() est appelée pour convertir l'image en JPEG .

♣La partie suivante du code envoie les données d'image converties au serveur HTTP en tant que flux multipartie . La fonction httpd\_resp\_send\_chunk() est utilisée pour envoyer chaque partie du flux multipartie .

♣Enfin , la fonction digitalWrite() est utilisée pour faire tourner les moteurs en appelant les broches GPIO appropriées .

♣En résumé , ce programme utilise une caméra pour capturer des images et les envoie en tant que flux de données multiparties via un serveur HTTP . Il contrôle également deux moteurs à l'aide de broches GPIO pour effectuer des actions .

## 2.2 Le Programme

```
#include "esp_camera.h"
#include <WiFi.h>
#include "esp_timer.h"
#include "img_converters.h"
#include "Arduino.h"
#include "fb_gfx.h"
#include "soc/soc.h"          // disable brownout problems
#include "soc/rtc_cntl_reg.h"  // disable brownout problems
#include "esp_http_server.h"

// Replace with your network credentials
const char* ssid = "A51";
const char* password = "12345678.';

#define PART_BOUNDARY "123456789000000000000987654321"

#define CAMERA_MODEL_AI_THINKER
//#define CAMERA_MODEL_M5STACK_PSRAM
//#define CAMERA_MODEL_M5STACK_WITHOUT_PSRAM
//#define CAMERA_MODEL_M5STACK_PSRAM_B
//#define CAMERA_MODEL_WROVER_KIT

#if defined(CAMERA_MODEL_WROVER_KIT)
    #define PWDN_GPIO_NUM -1
    #define RESET_GPIO_NUM -1
    #define XCLK_GPIO_NUM 21
    #define SIOD_GPIO_NUM 26
    #define SIOC_GPIO_NUM 27

    #define Y9_GPIO_NUM 35
    #define Y8_GPIO_NUM 34
    #define Y7_GPIO_NUM 39
    #define Y6_GPIO_NUM 36
    #define Y5_GPIO_NUM 19
    #define Y4_GPIO_NUM 18
    #define Y3_GPIO_NUM 5
    #define Y2_GPIO_NUM 4
```

```

#define VSYNC_GPIO_NUM 25
#define HREF_GPIO_NUM 23
#define PCLK_GPIO_NUM 22

#elif defined(CAMERA_MODEL_M5STACK_PSRAM)
#define PWDN_GPIO_NUM -1
#define RESET_GPIO_NUM 15
#define XCLK_GPIO_NUM 27
#define SIOD_GPIO_NUM 25
#define SIOC_GPIO_NUM 23

#define Y9_GPIO_NUM 19
#define Y8_GPIO_NUM 36
#define Y7_GPIO_NUM 18
#define Y6_GPIO_NUM 39
#define Y5_GPIO_NUM 5
#define Y4_GPIO_NUM 34
#define Y3_GPIO_NUM 35
#define Y2_GPIO_NUM 32
#define VSYNC_GPIO_NUM 22
#define HREF_GPIO_NUM 26
#define PCLK_GPIO_NUM 21

#elif defined(CAMERA_MODEL_M5STACK_WITHOUT_PSRAM)
#define PWDN_GPIO_NUM -1
#define RESET_GPIO_NUM 15
#define XCLK_GPIO_NUM 27
#define SIOD_GPIO_NUM 25
#define SIOC_GPIO_NUM 23

#define Y9_GPIO_NUM 19
#define Y8_GPIO_NUM 36
#define Y7_GPIO_NUM 18
#define Y6_GPIO_NUM 39
#define Y5_GPIO_NUM 5
#define Y4_GPIO_NUM 34
#define Y3_GPIO_NUM 35
#define Y2_GPIO_NUM 17
#define VSYNC_GPIO_NUM 22
#define HREF_GPIO_NUM 26
#define PCLK_GPIO_NUM 21

#elif defined(CAMERA_MODEL_AI_THINKER)
#define PWDN_GPIO_NUM 32
#define RESET_GPIO_NUM -1
#define XCLK_GPIO_NUM 0

```

```

#define SIOD_GPIO_NUM 26
#define SIOC_GPIO_NUM 27

#define Y9_GPIO_NUM 35
#define Y8_GPIO_NUM 34
#define Y7_GPIO_NUM 39
#define Y6_GPIO_NUM 36
#define Y5_GPIO_NUM 21
#define Y4_GPIO_NUM 19
#define Y3_GPIO_NUM 18
#define Y2_GPIO_NUM 5
#define VSYNC_GPIO_NUM 25
#define HREF_GPIO_NUM 23
#define PCLK_GPIO_NUM 22

#elif defined(CAMERA_MODEL_M5STACK_PSRAM_B)
#define PWDN_GPIO_NUM -1
#define RESET_GPIO_NUM 15
#define XCLK_GPIO_NUM 27
#define SIOD_GPIO_NUM 22
#define SIOC_GPIO_NUM 23

#define Y9_GPIO_NUM 19
#define Y8_GPIO_NUM 36
#define Y7_GPIO_NUM 18
#define Y6_GPIO_NUM 39
#define Y5_GPIO_NUM 5
#define Y4_GPIO_NUM 34
#define Y3_GPIO_NUM 35
#define Y2_GPIO_NUM 32
#define VSYNC_GPIO_NUM 25
#define HREF_GPIO_NUM 26
#define PCLK_GPIO_NUM 21

#else
#error "Camera model not selected"
#endif

#define MOTOR_1_PIN_1 14
#define MOTOR_1_PIN_2 15

static const char* _STREAM_CONTENT_TYPE = "multipart/x-mixed-
replace;boundary=" PART_BOUNDARY;
static const char* _STREAM_BOUNDARY = "\r\n--" PART_BOUNDARY "\r\n";
static const char* _STREAM_PART = "Content-Type: image/jpeg \r\nContent-
Length :%u\r\n\r\n";

```

```

httpd_handle_t camera_httpd = NULL;
httpd_handle_t stream_httpd = NULL;

static const char PROGMEM INDEX_HTML[] = R"rawliteral(
< html >
  < head >
    < title >ESP32-CAM Robot< /title >
    < metaname = "viewport" content = "width = device-width, initial-
scale = 1" >
  < style >
    body font-family: Arial; text-align: center; margin:0px auto; padding- top:
30px;
    table margin-left: auto; margin-right: auto; }
    td padding: 8 px; }
    .button {
      background-color: 2f4468;
      border: none;
      color: white;
      padding: 10px 20px;
      text-align: center;
      text-decoration: none;
      display: inline-block;
      font-size: 18px;
      margin: 6px 3px;
      cursor: pointer;
      -webkit-touch-callout: none;
      -webkit-user-select: none;
      -khtml-user-select: none;
      -moz-user-select: none;
      -ms-user-select: none;
      user-select: none;
      -webkit-tap-highlight-color: rgba(0,0,0,0);
    }
    img { width: auto ;
      max-width: 100% ;
      height: auto ;
    }
  < /style >
< /head >
< body >
  < h1 >ESP32-CAM Robot< /h1 >
  < img src="" id = "photo" >
  < table >
    < tr >< tdcolspan = "3" align = "center" >< buttonclass = "button"
onmousedown="toggleCheckbox('lock');"
```

```

ontouchstart="toggleCheckbox('lock');"
onmouseup="toggleCheckbox('stop');"
ontouchend="toggleCheckbox('stop');"> Forward </button></td></tr>

<tr><td colspan="3" align="center"><button class="button"
onmousedown="toggleCheckbox('unlock');"
ontouchstart="toggleCheckbox('unlock');"
onmouseup="toggleCheckbox('stop');"
ontouchend="toggleCheckbox('stop');"> Backward </button></td></tr>

</table>
<script>
function toggleCheckbox(x) {
    var xhr = new XMLHttpRequest();
    xhr.open("GET", "/action?go=" + x, true);
    xhr.send();
}
window.onload = document.getElementById("photo").src =
window.location.href.slice(0, -1) + ":81/stream";
</script>
</body>
</html>
)rawliteral';

static esp_err_tindex_handler(httpd_req_t *req){
    httpd_resp_set_type(req, "text/html");
    return httpd_resp_send(req, (const char *)INDEX_HTML, strlen(INDEX_HTML));
}

static esp_err_tstream_handler(httpd_req_t *req){
    camera_fb_t * fb = NULL;
    esp_err_tres = ESP_OK;
    size_t_jpg_buf_len = 0;
    uint8_t * _jpg_buf = NULL;
    char * part_buf[64];

    res = httpd_resp_set_type(req, _STREAM_CONTENT_TYPE);
    if(res != ESP_OK){
        return res;
    }

    while(true){
        fb = esp_camera_fb_get();
        if (!fb) {
            Serial.println("Camera capture failed");
            res = ESP_FAIL;

```

```

} else {
    if(fb-> width > 400){
        if(fb-> format != PIXFORMAT_JPEG){
            bool jpeg_converted = frame2jpg(fb, 80, _jpg_buf, _jpg_buf_len);
            esp_camera_fb_return(fb);
            fb = NULL;
            if(!jpeg_converted){
                Serial.println("JPEG compression failed");
                res = ESP_FAIL;
            }
        } else {
            _jpg_buf_len = fb-> len;
            _jpg_buf = fb-> buf;
        }
    }
}
if(res == ESP_OK){
    size_t hlen = snprintf((char *)part_buf, 64, _STREAM_PART,
_jpg_buf_len);
    res = httpd_resp_send_chunk(req, (const char *)part_buf, hlen);
}
if(res == ESP_OK){
    res = httpd_resp_send_chunk(req, (const char *)_jpg_buf, _jpg_buf_len);
}
if(res == ESP_OK){
    res = httpd_resp_send_chunk(req,_STREAM_BOUNDARY,
strlen(_STREAM_BOUNDARY));
}
if(fb){
    esp_camera_fb_return(fb);
    fb = NULL;
    _jpg_buf = NULL;
} else if(!_jpg_buf){
    free(_jpg_buf);
    _jpg_buf = NULL;
}
if(res != ESP_OK){
    break;
}
//Serial.printf("MJPEG: %uB \n",(uint32_t)(_jpg_buf_len));
}
return res;
}

static esp_err_t cmd_handler(httpd_req_t *req){
    char* buf;

```

```

size_t buf_len;
char variable[32] = 0;

buf_len = httpd_req_get_url_query_len(req) + 1;
if (buf_len > 1) {
    buf = (char*)malloc(buf_len);
    if(!buf){
        httpd_resp_send_500(req);
        return ESP_FAIL;
    }
    if (httpd_req_get_url_query_str(req, buf, buf_len) == ESP_OK) {
        if (httpd_query_key_value(buf, "go", variable, sizeof(variable)) == ESP_OK)
    {
        } else {
            free(buf);
            httpd_resp_send_404(req);
            return ESP_FAIL;
        }
    } else {
        free(buf);
        httpd_resp_send_404(req);
        return ESP_FAIL;
    }
    free(buf);
} else {
    httpd_resp_send_404(req);
    return ESP_FAIL; }
```

sensor\_t \* s = esp\_camera\_sensor\_get();  
int res = 0;

```

if(!strcmp(variable, "lock")) {
    Serial.println("lock");
    digitalWrite(MOTOR_1_PIN_1, 1);
    digitalWrite(MOTOR_1_PIN_2, 0);

}
else if(!strcmp(variable, "unlock")) {
    Serial.println("unlock");
    digitalWrite(MOTOR_1_PIN_1, 0);
    digitalWrite(MOTOR_1_PIN_2, 1);

}
else if(!strcmp(variable, "stop")) {
    Serial.println("Stop");
    digitalWrite(MOTOR_1_PIN_1, 0);
    digitalWrite(MOTOR_1_PIN_2, 0);
```

```

    }
else {
    res = -1;
}

if(res){
    return httpd_resp_send_500(req);
}

httpd_resp_set_hdr(req, "Access-Control-Allow-Origin", "*");
return httpd_resp_send(req, NULL, 0);
}

void startCameraServer(){
    httpd_config_t config = HTTPD_DEFAULT_CONFIG();
    config.server_port = 80;
    httpd_uri_t index_uri = {
        .uri = "/",
        .method = HTTP_GET,
        .handler = index_handler,
        .user_ctx = NULL
    };

    httpd_uri_t cmd_uri = {
        .uri = "/action",
        .method = HTTP_GET,
        .handler = cmd_handler,
        .user_ctx = NULL
    };
    httpd_uri_t stream_uri = {
        .uri = "/stream",
        .method = HTTP_GET,
        .handler = stream_handler,
        .user_ctx = NULL
    };
    if (httpd_start(camera_httpd, config) == ESP_OK) {
        httpd_register_uri_handler(camera_httpd, index_uri);
        httpd_register_uri_handler(camera_httpd, cmd_uri);
    }
    config.server_port += 1;
    config.ctrl_port += 1;
    if (httpd_start(stream_httpd, config) == ESP_OK) {
        httpd_register_uri_handler(stream_httpd, stream_uri);
    }
}

```

```

void setup() {
    WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0); //disable brownout
detector

pinMode(MOTOR_1_PIN_1, OUTPUT);
pinMode(MOTOR_1_PIN_2, OUTPUT);

Serial.begin(115200);
Serial.setDebugOutput(false);

camera_config_t config;
config.ledc_channel = LEDC_CHANNEL_0;
config.ledc_timer = LEDC_TIMER_0;
config.pin_d0 = Y2_GPIO_NUM;
config.pin_d1 = Y3_GPIO_NUM;
config.pin_d2 = Y4_GPIO_NUM;
config.pin_d3 = Y5_GPIO_NUM;
config.pin_d4 = Y6_GPIO_NUM;
config.pin_d5 = Y7_GPIO_NUM;
config.pin_d6 = Y8_GPIO_NUM;
config.pin_d7 = Y9_GPIO_NUM;
config.pin_xclk = XCLK_GPIO_NUM;
config.pin_pclk = PCLK_GPIO_NUM;
config.pin_vsync = VSYNC_GPIO_NUM;
config.pin_href = HREF_GPIO_NUM;
config.pin_sscb_sda = SIOD_GPIO_NUM;
config.pin_sscb_scl = SIOC_GPIO_NUM;
config.pin_pwdn = PWDN_GPIO_NUM;
config.pin_reset = RESET_GPIO_NUM;
config.xclk_freq_hz = 20000000;
config.pixel_format = PIXFORMAT_JPEG;

if(psramFound()){
    config.frame_size = FRAMESIZE_VGA;
    config.jpeg_quality = 10;
    config.fb_count = 2;
} else {
    config.frame_size = FRAMESIZE_SVGA;
    config.jpeg_quality = 12;
    config.fb_count = 1;
}

// Camera init
esp_err_t err = esp_camera_init(&config);

```

```
if (err != ESP_OK) {
    Serial.printf("Camera init failed with error 0x%x", err);
    return;
}
// Wi-Fi connection
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
}
Serial.println("");
Serial.println("WiFi connected");

Serial.print("Camera Stream Ready! Go to: http://");
Serial.println(WiFi.localIP());

// Start streaming web server
startCameraServer();
}

void loop() {

}
```

# Chapitre 3

## *Les perspectives*

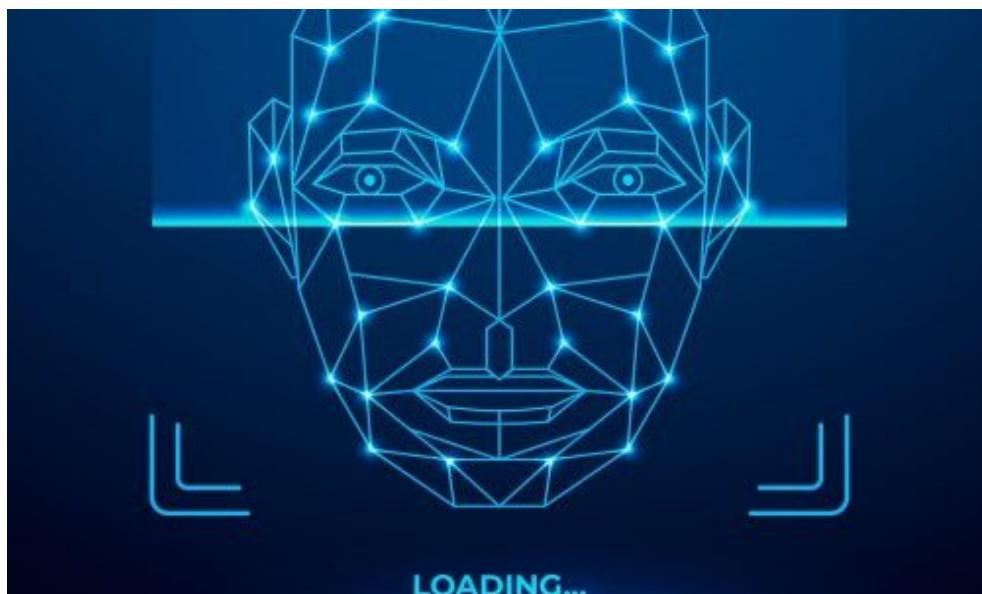
EagleSight est un système de sécurité basé sur l'utilisation d'une ESP32-CAM et une carte Arduino . Ce projet offre plusieurs perspectives potentielles pour l'amélioration et l'extension .

Dans la suite , on aura donné trois perspectives pour améliorer notre système et devient plus en plus performant .

### **3.1 La reconnaissance faciale**

Parfois le contrôleur du système était occupé , d'où il ne voit pas les notification du joindre , d'où l'intégration d'un système de reconnaissance faciale pouvez résoudre cette problème .

L'intégrer d'un système de **reconnaissance faciale** peut améliorer la sécurité de EagleSight . Cela nous permettrait d'identifier les personnes autorisées à entrer et d'ouvrir la porte uniquement pour elles . Nous pouvons utiliser une bibliothèque de reconnaissance faciale telle que **OpenCV** pour cela .



### **3.2 Intégration des polycapteurs**

Intégrer des capteurs supplémentaires peut être une excellente idée pour améliorer la précision et la fiabilité du EagleSight .

♠ On peut intégrer **capteur du mouvement HC-SR50** pour améliorer la détection les actions du personne . Si le contrôleur refuse d'ouvrir la porte au Mr X , et Mr X veut ouvrir la porte par force , le capteur du mouvement permet la détection de cette violence



Figure 3.1: Capteur HC-SR510

♠ Intégrer dans les systèmes un **capteur détection d'obstacle** pour la connaître la position du porte : soit qu'il est ouvert ou non .



Figure 3.2: Capteur détection d'obstacle

### **3.3 Stockage du données**

L'idée du projet est gardé la sécurité . Puisque on a un stockage etéchange du données , il ya toujours le risque du spam .

L'utilisation de la technologie **blockchain** pour sécuriser et protéger les données de système EagleSight de sécurité peut être une excellente idée .

La blockchain est une technologie de registre distribué qui permet de stocker les données relatives à l'authentification et à l'autorisation d'accès de manière

décentralisée , sécurisée et inviolable .

Cela ajouterait une couche de sécurité supplémentaire et garantirait que les données de votre système de sécurité sont résistantes aux attaques malveillantes .

# *Conclusion*

La fabrication d'un système de sécurité sous le nom " EagleSight " qui permet d'ouvrir la porte en prenant une photo et en l'envoyant au contrôleur est impressionnant , car il combine plusieurs technologies telles que l'ESP32-CAM ; la carte Arduino ; le capteur ultrason ; les diodes ; le buzzer ; le moteur et le Wi-Fi pour contrôler à distance l'ouverture de la porte .

Il est clair que nous avons travaillé dur pour concevoir et construire ce système de sécurité , et nous avons fourni des détails sur les composants matériels et le programme utilisés , ainsi que sur les fonctionnalités du système .

Notre projet présente plusieurs avantages , notamment la sécurité améliorée de la porte grâce à l'utilisation d'une photo pour l'ouvrir , la commodité offerte par le contrôle à distance via le Wi-Fi .

Cependant , il est important de prendre en compte les limitations et les défis éventuels de notre projet , tels que la nécessité de garantir la stabilité de la connexion Wi-Fi , la compatibilité de votre système avec d'autres technologies de sécurité et la fiabilité du matériel utilisé que l'on peut les traiter au futur ainsi que les perspectives .

En conclusion , EagleSight basé sur la prise de photo pour ouvrir la porte est impressionnant et peut être bénéfique pour les utilisateurs qui cherchent à améliorer la sécurité de leur domicile .

EagleSight est une solution pour " "Private Life and Peaceful Mind"

---

**ESP32-CAM Robot**

