REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE DE SAÏDA DR MOULAY TAHAR

Faculté de technologie

Département d'électronique

MEMOIRE DE FIN D'ETUDES EN VUE DE L'OBTENTION

DU DIPLOME DE MASTER EN ELECTRONIQUE

OPTION : INSTRUMENTATION

THEME :

SYSTÈME DE SURVEILLANCE UTILISANT UN ROBOT EQUIPÉ D'UNE CARTE ESP32-CAM

Présenté par :

KEBIR Walid

KHATIR Manouar

Soutenu le 19 juin 2023

Devant le jury composé de :

DINE Khaled	Maître de conférences à l'Université de Saida	Président
MOSTEFAI Lotfi	Maître de conférences à l'Université de Saida	Examinateur
MAACHOU Fatima	Maître de conférences à l'Université de Saida	Encadrante

Année Universitaire : 2022 - 2023

Abstract

This project aims to design a remote surveillance system based on the ESP32-CAM board integrated into a robot car. The system enables real-time video streaming through a web interface accessible from a smartphone. In addition to video streaming, the web interface also provides control functionalities. It allows for controlling the robot's movement using directional commands, adjusting the camera orientation using a servomotor, and adjusting the lighting intensity for low-light conditions using an LED flash. To ensure real-time bidirectional communication between the ESP32-CAM and the web interface, the system utilizes HTTP and WebSocket protocols. This remote surveillance system offers a practical and flexible solution for remote monitoring and control of the robot car.

Keywords : ESP32-CAM, Web Interface, OV2640 Camera, Robot car, HTTP protocol, WebSocket, motor driver, Wi-Fi.

Résumé

Ce projet vise à concevoir un système de surveillance à distance basé sur la carte ESP32-CAM intégrée à une voiture robot. Le système permet la diffusion en direct de vidéos via une interface Web accessible depuis un smartphone. En plus de la diffusion vidéo, l'interface Web offre également des fonctionnalités de contrôle. Il est possible de contrôler le mouvement du robot grâce à des commandes de direction, d'ajuster l'orientation de la caméra en utilisant un servomoteur, et de régler l'intensité lumineuse pour les conditions de faible luminosité à l'aide d'une LED flash. Pour assurer une communication bidirectionnelle en temps réel entre l'ESP32-CAM et l'interface Web, le système utilise les protocoles HTTP et les WebSockets. Ce système de surveillance à distance offre une solution pratique et flexible pour la surveillance et le contrôle à distance de la voiture robot.

Mots Clés : ESP32-CAM, Interface Web, Camera OV2640, Voiture robot, Protocole HTTP, WebSocket, pilote moteur, Wi-Fi.

الملخص :

يهدف هذا المشروع إلى تصميم نظام مراقبة عن بعد يعتمد على لوحة ESP32-CAM المدمجة في سيارة روبوت. يتيح النظام دفق الفيديو في الوقت الفعلي من خلال واجهة ويب يمكن الوصول إليها من هاتف ذكي. بالإضافة إلى دفق الفيديو ، توفر واجهة الويب أيضا وظائف التحكم. يسمح بالتحكم في حركة الروبوت باستخدام أوامر الاتجاه ، وضبط اتجاه الكاميرا باستخدام محرك مؤازر ، وضبط شدة الإضاءة لظروف الإضاءة المنخفضة باستخدام فلاش ليد. لضمان الاتصال ثنائي الاتجاه في الوقت الحقيقي بين ESP32-CAM وواجهة الويب ، يستخدم النظام بروتوكولات هتب و ويبسوكيت. يوفر نظام المراقبة عن بعد هذا حلا عمليا ومرنا للمراقبة والتحكم عن بعد في سيارة الروبوت.

كلمات البحث : ESP32-CAM ، واجهة ويب ، OV2640 كاميرا ، سيارة روبوت ، بروتوكول HTTP ، ويبسوكيت ، سائق المحرك ، واي فاي.

Remerciements

Nous tenons à remercier notre Dieu, le tout puissant, de nous avoir donné la santé et la volonté pour la réalisation de ce modeste travail.

Nous souhaitons exprimer nos sincères remerciements à notre encadrante, Dr. Fatima MAACHOU, pour sa disponibilité, son orientation et son soutien moral tout au long de notre travail. Sa présence bienveillante, ses conseils éclairés, son implication active dans notre projet a été essentielle pour nous guider dans nos recherches, développer nos compétences et approfondir notre compréhension du sujet.

Nous tenons à exprimer notre profonde gratitude envers le Dr. Khaled DINE pour avoir accepté de présider le jury de notre soutenance.

Nous souhaitons également remercier chaleureusement le Dr. Lotfi MOSTEFAI pour sa participation en tant que membre du jury.

Nous sommes conscients de l'importance du rôle du jury dans l'évaluation de notre travail de fin d'études, et nous sommes honorés de pouvoir bénéficier de l'expertise et de l'expérience du jury.

Tous nos infinis remerciements vont à tous les enseignants qui ont contribué à notre formation durant notre cursus universitaire.

Enfin, nous remercions tous ceux qui ont participé de près ou de loin à la réalisation de ce mémoire.

Table des matières

Introduction générale **1**

1 Etat de l'art **3**
1.1 Introduction 3
1.2 Un survol de la littérature 4
1.3 Conclusion 5

2 Matériel et méthodes **6**
2.1 Introduction 6
2.2 Description de la carte de développement l'ESP32-CAM 6
2.2.1 Le processeur ESP32-S 6
2.2.2 Les types de mémoire de l'ESP32-CAM 7
2.2.3 La caméra OV2640 8
2.2.4 Le stockage 8
2.2.5 Connectivité sans fil 8
2.2.6 Connexion d'une antenne 9
2.2.7 LEDs embarquées sur l'ESP32-CAM 9
2.2.8 Brochage ESP32-CAM 10
2.2.9 Programmation de l'ESP32-CAM 15
2.2.10 Utilisation de l'adaptateur FTDI 16
2.3 Module de puissance L298N 17
2.3.1 Les « Ponts en H » 18
2.3.2 Le pilotage PWM 19
2.3.3 Table de vérité du L298N (fonctionnement) 21
2.4 Moteur continu DC 21
2.5 Servo-moteur 23
2.5.1 Servomoteur type S90 23
2.5.2 Fonctionnement du servomoteur numérique : 25
2.5.3 L'asservissement des servo-moteurs 25
2.5.4 Servo-moteurs : Signal de commande 26
2.6 Serveur Web 27

2.6.1 Etapes pour afficher une page Web 28
2.6.2 Serveur Web synchrone 28
2.6.3 Serveur Web asynchrone 29
2.6.4 WebSocket 30
2.7 Bibliothèques utilisées dans le code Arduino du système de surveillance 31
2.7.1 Async TCP 31
2.7.2 io stream 31
2.7.3 sstream 32
2.7.4 esp camera.h 33
2.7.5 Arduino.h 34
2.7.6 WiFi.h 34
2.8 Technologies HTML, CCS et Javascript 35
2.9 Video et streaming 36
2.10 Conclusion 37

3 Conception du système de surveillance à base de l'ESP32-CAM 38
3.1 Introduction 38
3.2 Etapes de conception du système de surveillance 38
3.2.1 Commande des moteurs DC par l'ESP32-CAM 38
3.2.2 Commande du servo-moteur S90 par l'ESP32-CAM 41
3.2.3 Commande de la LED Flash caméra de l'ESP32-CAM 42
3.2.4 Acquisition de la vidéo par la caméra embarquée de l'ESP32-CAM 44
3.3 Assemblage de la voiture robot 44
3.4 Conception de l'interface Web pour le contrôle de la voiture robot . . 47
3.5 Tests et résultats 49
3.6 Conclusion 51

Conclusion générale 52

Références Bibliographiques 54

A Datasheet CI L298 56

B Datasheet Camera OV2460 60

Table des figures

2.1 Processeur ESP32-S. 7
2.2 Les mémoires de l'ESP32-CAM. 7
2.3 Caméra de l'ESP32-CAM 8
2.4 Connexion d'une antenne externe. 9
2.5 LED Flash caméra. 10
2.6 Brochage ESP32-CAM. 10
2.7 Broches GPIO. 11
2.8 Broches de carte MicroSD. 11
2.9 Broches du convertisseur ADC2. 12
2.10 Touches tactiles. 12
2.11 Broches SPI. 13
2.12 Broches UART. 13
2.13 Broches PWM. 14
2.14 Broches GPIO RTC. 14
2.15 Broches de puissance. 15
2.16 Connection du module FTDI à l'ESP32-cam. 16
2.17 Module pilote de moteur L298N. 17
2.18 Architecture interne du module L298N. 18
2.19 Pont en H. 18
2.20 Principe de fonctionnement du Pont en H. 19
2.21 Le pilotage PWM 20
2.22 Schéma d'un moteur à courant continu. 22
2.23 Ensemble moteur, réducteur et roue. 22
2.24 Les constituants du servomoteur. 24
2.25 Principe de l'asservissement des servomoteurs. 26
2.26 le temps séparant deux fronts. 26
2.27 LA DURÉE DE L'ÉTAT HAUT. 27
2.28 Serveur Web schema général. 27

3.1 Schéma de connexion de driver moteur à l'ESP32-cam 39

3.2 Logigramme de commande de vitesse et des directions de la voiture robot. 40
3.3 Schéma de connexion du servo-moteur à l'ESP32-CAM. 41
3.4 Logigramme de commande du servomoteur. 43
3.5 Montage des 04 moteurs TT sur le châssis inférieur de la voiture robot. 44
3.6 Montage du châssis supérieur et des 4 roues de la voiture robot. . . . 45
3.7 Mise en place du module L298N sur la voiture robot. 45
3.8 46
3.9 46
3.10 Schéma de montage des différents composants équipant la voiture robot. 47
3.11 Interface utilisateur de contrôle de la voiture robotisée. 48
3.12 Bloc diagramme du système de surveillance à distance basé sur l'ESP32-CAM. 49
3.13 Image extraite de la vidéo prise par la voiture robot. 51

Liste des tableaux

2.1 Table de correspondance des broches FTDI – ESP32-CAM. 16
2.2 Table de vérite du pilote L298N. 21

3.1 Table de correspondance des broches L298N- ESP32CAM. 39
3.2 Table de correspondance des broches servomoteur – ESP32-CAM. . . 41

Liste des abbréviations

ADC	Analog Digital Converter
BLE	Bluetooth Low Energy
BR/EDR	Basic Rate/Enhanced Data Rate
CSS	Cascading Style Sheets
DC	Direct Current
DDM	Direct Drive Mode
DVP	Digital Video Port
FPC	Flexible Printed Circuit
FTDI	Future Technology Devices International
GPIO	General Purpose Input Output
HTML	Hyper Text Markup Language
IDE	Integrated Developpment Environment
IoT	Internet of Things
LED	Light Emitting Diodes
MBM	Map-Based Mode
microSD	micro Secure Digital
PSRAM	Pseudo Static Random Access Memory
PWM	Pulse Width Modulation
RTC	Real Time Clock
SCCB	Serial Camera Control Bus
SoC	System on Chip
SSID	Service Set Identifier
SRAM	Static Random Access Memory
TCP	Transmission Control Protocol
UART	Universal Asynchronous Receiver Transmitter
ULP	Ultra Low Power

USB	Universal Serial Bus
VSPI	Virtual Serial Peripheral Interface
Wi-Fi	Wireless Fidelity

Introduction générale

La télésurveillance est devenue un domaine d'intérêt croissant en raison de son potentiel à fournir une surveillance à distance efficace et sécurisée. Elle offre la possibilité de surveiller des zones sensibles, telles que les domiciles, les bureaux ou les sites industriels, en temps réel, offrant ainsi une tranquillité d'esprit et une protection accrue.

L'utilisation des robots dans un système de surveillance constitue une avancée significative dans le domaine de la sécurité et de la surveillance. Cette technologie moderne offre des possibilités d'exploration et de surveillance des zones à distance parfois même difficile d'accès.

C'est dans cette thématique que s'inscrit ce projet de fin d'études portant sur l'étude et la réalisation d'un système de surveillance à distance utilisant une voiture robot équipée d'une carte de développement ESP32-CAM. Une interface Web a été développée pour contrôler à distance les différents mouvements de la voiture robot, ainsi qu'un servomoteur pour orienter la caméra embarquée dans différentes directions d'une zone de surveillance. Les images vidéos capturées sont diffusées sur l'interface en temps réel.

L'ESP32-CAM, avec sa caméra embarquée et sa connectivité Wi-Fi, est devenu un choix populaire pour les applications de surveillance. Les capacités de traitement embarquées dans l'ESP32 permettent de réaliser des tâches telles que la détection d'objets, la reconnaissance faciale et la détection de mouvement, ajoutant une couche supplémentaire de fonctionnalités intelligentes au système de surveillance.

L'avenir des systèmes de surveillance continue de s'améliorer avec les avancées technologiques constantes. On peut s'attendre à des améliorations dans les domaines de la vision par ordinateur, de l'automatisation et de l'intelligence artificielle, ce qui permettra d'accroître encore davantage les capacités de surveillance des robots ESP32-CAM.

Notre manuscrit s'articule essentiellement autour de 03 chapitres :

Le premier chapitre est un survol de littérature qui reporte quelques travaux et recherches sur le domaine de la surveillance basée sur l'ESP32 et les robots.

Le second chapitre est consacré à décrire les différents composants et langages utilisés pour la conception de notre système de surveillance.

Le dernier chapitre est destiné à la partie pratique dans laquelle nous aborderons la réalisation de notre système de surveillance, le développement de notre application Web smartphone et les différents tests réalisés.

Ce munuscrit se termine par une conclusion générale.

Chapitre 1

Etat de l'art

1.1 Introduction

La surveillance à distance joue un rôle crucial dans de nombreux secteurs tels que la sécurité, la domotique et l'industrie. Avec l'avancement de l'IoT et de la technologie des applications Android, de nombreuses études ont été réalisées pour concevoir des systèmes Android basés sur l'IoT, exploitant les fonctionnalités des modules ESP32 et ESP8266. Ces modules offrent des fonctionnalités avancées permettant la surveillance et le contrôle à distance dans les applications IoT.

L'ESP32-CAM, avec sa fonctionnalité de caméra intégrée, offre une plateforme idéale pour développer des solutions de surveillance à distance. elle se distingue notamment par ses avantages supplémentaires lorsqu'il s'agit de projets nécessitant une fonctionnalité de vidéo et de capture d'images en temps réel.

Ce module spécifique, ESP32-CAM, est un module de développement sans fil ESP32 dédié à la capture d'images dans les applications IoT. Il offre la possibilité de créer des projets tels que les systèmes de surveillance à distance, des systèmes de sécurité, des systèmes de verrouillage de porte basés sur la reconnaissance faciale, et bien d'autres encore. Grâce à sa fonctionnalité de capture d'images, il est possible de réaliser des projets nécessitant la reconnaissance faciale, la reconnaissance des yeux, des objets, etc. De plus, la présence d'un emplacement pour carte SD permet d'enregistrer facilement les images capturées.

Cela en fait une carte de développement polyvalente pour la réalisation de projets liés à la vision par ordinateur, à la surveillance, à la robotique et à de nombreux autres domaines de l'IoT.

Ce chapitre vise à fournir un aperçu de l'état de l'art dans le domaine de la télésurveillance, en mettant en évidence les développements récents et l'utilisation de l'ESP32-CAM.

1.2 Un survol de la littérature

Selvam Muthuraman a proposé une application pour contrôler un voiture robot à l'aide d'un module Bluetooth. Un Arduino, un pilote de moteur L293D, un module Wi-Fi et des servomoteurs ont été utilisés pour déplacer une voiture robot avec un bras manipulateur commandé par une application mobile ainsi qu'un serveur Web pour la surveillance.

Dans le contexte du respect de la distanciation sociale pendant la pandémie de coronavirus, une étude menée par V. Sai Nikhil et al. propose l'utilisation d'une voiture robot pour surveiller les infractions dans les lieux publics. Le robot est équipé d'un capteur ultrasonique permettant de mesurer la distance entre deux personnes, ainsi que d'une carte ESP32-CAM. Lorsque cette distance est inférieure à un seuil prédéfini, une alerte est envoyée aux autorités compétentes via une connexion Wi-Fi. Cette alerte inclut une photographie de l'infraction capturée par la caméra de l'ESP32-CAM, et est transmise par e-mail. Cette image peut servir de preuve pour prendre des mesures disciplinaires à l'encontre des contrevenants.

Dans un article rédigé de Band Saumya et al., un système de surveillance militaire basé sur l'IoT est conçu pour répondre aux exigences des soldats sur le champ de bataille. Il utilise un contrôleur ESP8266 interfacé avec divers capteurs et connecté à une application. Pour garantir une surveillance adéquate à la frontière, une caméra ESP32-CAM est également utilisée pour la diffusion en direct, la détection et la reconnaissance des visages.

Un article rédigé par Filantropi Yusuf et al. porte sur la sécurité à domicile contre les crimes et les accidents pouvant survenir dans les maisons tels que les incendies. Dans ce cas, un système a été conçu pour surveiller l'état du domicile à distance via un smartphone. Ce système de sécurité de est basé sur l'ESP32 caméra interfacé à un capteur PIR pour la détection mouvement et un capteur d'incendie. En fonction des données des capteurs, l'ESP32 caméra contrôle à distance une serrure pour ouvrir et fermer une porte. Les données provenant des capteur seront envoyées à un serveur par l'ESP32-CAM. Les images prises et les notifications d'incendie peuvent être envoyées à l'application Telegram.

V SHANKAR Robot de combat intelligent 2015 : Ce robot a été développé pour permettre une opération à distance en utilisant la technologie RF et une caméra sans fil à des fins de surveillance. Le robot et la caméra sont capables d'envoyer des images en temps réel avec des capacités de vision nocturne via un réseau sans fil. Ce type de robot pourrait être utile dans les zones de guerre pour des missions d'espionnage. Cependant, il convient de noter que dans cette technologie, le robot ne peut être contrôlé qu'à une distance maximale de dix mètres. Une amélioration de la

portée du robot a été présentée grâce à une nouvelle technique de classification, mais cette technologie a souvent entraîné des pertes de connexion avec la caméra.

Dans un article de Nakshtra Popli et al., il a été discuté de la manière de contrôler une voiture robotisée à l'aide d'un module Wi-Fi et d'un téléphone portable. Dans le cadre d'activités d'espionnage, une nouvelle version d'une voiture contrôlée sans fil a été proposée. Ce robot est un robot d'espionnage portable équipé d'un système de communication sans fil basé sur le module AIThinker. Les images sont capturées par la caméra de l'ESP32-CAM et sont stockées sur une carte microSD. Dans ce cas, le protocole de communication HTTP sera utilisé pour recevoir le flux vidéo de la caméra OV2640 via un navigateur web. Les signaux provenant du smartphone de l'utilisateur sont utilisés pour contrôler les mouvements du robot, et la vidéo est diffusée en direct via une caméra montée sur le robot.

Il a été rapporté dans un article de Hebah H.O. Nasereddin et Amjad Abdullah Abdelkarim, une étude sur l'utilisation de la technologie Bluetooth pour contrôler un robot à l'aide d'un smartphone. Dans cette étude, le robot peut être contrôlé en utilisant deux modes : le mode de pilotage direct (DDM) et le mode basé sur la carte (MBM). Dans le mode DDM, le robot peut être déplacé dans toutes les directions en fonction des besoins de l'utilisateur. En ce qui concerne le mode MBM, il permet à l'utilisateur de spécifier un point de départ, un point d'arrivée et des obstacles afin de calculer le chemin le plus court. Ainsi, l'utilisateur a la possibilité de choisir l'un de ces deux modes pour contrôler le robot via une communication sans fil.

1.3 Conclusion

Ce chapitre a permis d'appréhender les développements récents dans le domaine de la télésurveillance, les avancées et les possibilités offertes par l'ESP32-CAM dans le domaine de la télésurveillance. Avec ces avancées, l'IoT continue de révolutionner le domaine de la surveillance et du contrôle à distance, ouvrant la voie à de nouvelles possibilités d'applications intelligentes et efficaces.

Chapitre 2

Matériel et méthodes

2.1 Introduction

Ce chapitre présente une description détaillée de la carte de développement ESP32-CAM basé sur le système-sur-puce ESP32S d'Espressif qui intègre une caméra et prend en charge diverses fonctionnalités de connectivité. Nous examinerons également les composants matériels essentiels destinés à notre voiture robot, tels que le servomoteurs, les moteurs DC et le module L298N. En outre, nous aborderons les principales bibliothèques utilisées dans le code de notre application, ainsi que les technologies clés, notamment le Wi-Fi, le HTML, le CSS et JavaScript.

2.2 Description de la carte de développement l'ESP32-CAM

La carte de développement ESP32-CAM est une carte du fabricant AI-Thinker. Elle intègre le SoC ESP32-S d'Espressif et une caméra OV2640 embarquée. Le SoC intègre plusieurs composants sur une même puce. Cela comprend un microprocesseur ESP32, de la mémoire RAM, du stockage et divers périphériques. Grâce à ses fonctionnalités, l'ESP32-CAM offre une plateforme plus adaptée aux applications spécifiques, comme la surveillance vidéo, la reconnaissance d'images, les projets IoT et bien d'autres. Dans ce qui suit, nous allons décrire brievement les divers composants de l'ESP32-CAM.

2.2.1 Le processeur ESP32-S

Le processeur ESP32 est un dual core 32 bits, fonctionnant à une fréquence de 240 MHz. Il est basé sur l'architecture Tensilica Xtensa LX6, qui offre une puissance de calcul élevée et une efficacité énergétique optimisée. Il est donc adapté aux tâches

intensives telles que le traitement vidéo, la reconnaissance faciale et l'intelligence artificielle. La Figure 2.1 indique l'emplacement où se trouve intégré le processeur ESP32 sur le SoC ESP32-S.

FIGURE 2.1 – Processeur ESP32-S.

2.2.2 Les types de mémoire de l'ESP32-CAM

Le SoC ESP32-S dispose de 520 Ko de RAM interne pour le stockage de données et des instructions pendant l'exécution des programmes. En outre, il est équipé de 4 Mo de PSRAM externe pour augmenter la capacité de mémoire disponible, bénéfique pour les applications gourmandes en mémoire, comme les traitements d'audio et d'images. Les programmes et les données sont stockées d'une manière permanente dans une mémoire Flash intégrée au SoC de capacité 4Mo. La Figure 2.2 montre ces différents types de mémoire sur l'ESP32-CAM.

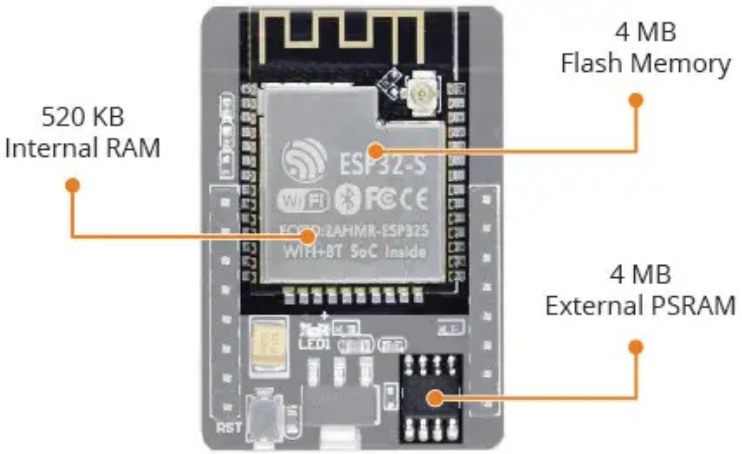


FIGURE 2.2 – Les mémoires de l'ESP32-CAM.

2.2.3 La caméra OV2640

Le capteur de caméra OV2640 sur l'ESP32-CAM est ce qui distingue l'ESP32-CAM des autres cartes de développement ESP32 et la rend idéale pour une utilisation dans des projets vidéo.

La caméra OV2640 a une résolution maximale de 2 mégapixels, ce qui équivaut à une résolution de 1600 × 1200 pixels. Cela permet de capturer des images détaillées et de qualité. Elle prend en charge le format JPEG. Le module de caméra OV2640 est connecté sur l'ESP32-CAM au moyen d'un connecteur FPC, comme le montre la Figure 2.3. Le connecteur FPC, ou connecteur à enfoncement, est couramment utilisé pour connecter des modules de caméra à des cartes de développement ou des circuits imprimés. Il offre une connexion fiable et compacte grâce à leur conception à enfoncement, où le câble plat flexible est inséré dans le connecteur et maintenu en place par une pression exercée par le connecteur lui-même.

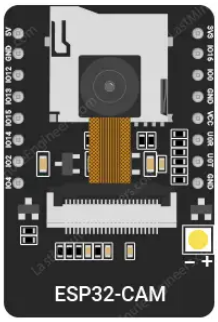


FIGURE 2.3 – Caméra de l'ESP32-CAM

L'OV2640 est compatible avec différentes interfaces de communication, notamment l'interface série SCCB (Serial Camera Control Bus) et l'interface parallèle DVP (Digital Video Port). Ces interfaces permettent une communication facile entre la caméra et le microcontrôleur, comme l'ESP32.

2.2.4 Le stockage

L'ESP32-CAM est équipé d'un lecteur de carte microSD, ce qui offre une grande flexibilité en termes de stockage de données et de capture d'images. Il faut configurer un programme pour pouvoir enregistrer directement les images sur la carte microSD, ce qui facilite la gestion et le transfert ultérieur des images.

2.2.5 Connectivité sans fil

l'ESP32 est doté d'un émetteur-récepteur Wi-Fi 802.11b/g/n HT40 intégré, ce qui lui confère une connectivité sans fil pour se connecter à un réseau Wi-Fi.

L'ESP32 peut fonctionner en mode Station, ce qui lui permet de se connecter à un réseau Wi-Fi existant, ou en mode Point d'accès, où il peut créer son propre réseau Wi-Fi auquel d'autres appareils peuvent se connecter.

L'ESP32-CAM est équipé également d'une connectivité Bluetooth de version 4.2, prenant en charge à la fois le mode de communication traditionnel BR/EDR (Basic Rate/Enhanced Data Rate) et le mode basse consommation BLE (Bluetooth Low Energy). Ce qui permet à l'ESP32-CAM de se connecter à d'autres appareils prenant en charge Bluetooth, tels que des smartphones, des écouteurs sans fil, des périphériques IoT.

2.2.6 Connexion d'une antenne

L'ESP32-CAM intègre une antenne de trace PCB embarquée ainsi qu'un connecteur IPEX pour connecter une antenne externe. IPEX est un type de connecteur d'antenne utilisé pour connecter l'antenne externe à la carte de développement. Il s'agit d'un type de connecteurs RF miniatures, également connus sous le nom de connecteurs u.fl, u.FL ou micro-coaxiaux. Il permet une connexion solide et stable entre l'antenne et la carte, assurant ainsi une transmission sans fil fiable et de qualité.

Un trio de pad est situé à côté du connecteur u.FL et entre l'antenne embarquée et le boîtier métallique de l'ESP32-S. On y trouve un cavalier de sélection d'antenne (résistance zéro ohm) illustré à la Figure 2.4 permettant de choisir entre les deux types d'antennes.

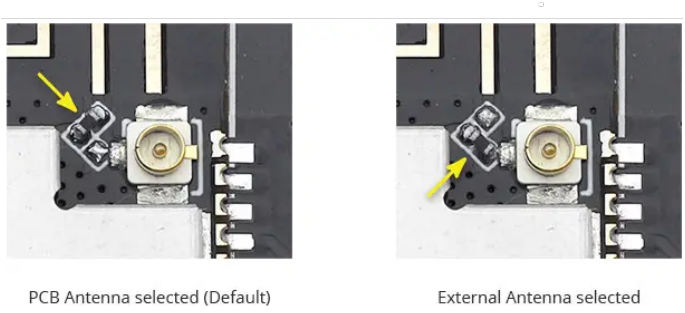


FIGURE 2.4 – Connexion d'une antenne externe.

2.2.7 LEDs embarquées sur l'ESP32-CAM

L'ESP32-CAM embarque une LED en forme de carré de couleur blanche située sur sa face avant comme le montre la Figure2.5 . Elle est destinée à être utilisé comme flash d'appareil photo. Une seconde LED rouge de petite taille se trouve à l'arrière de l'ESP32-CAM qui sert d'indicateur d'état. Elle est programmable par l'utilisateur et connectée au GPIO33.

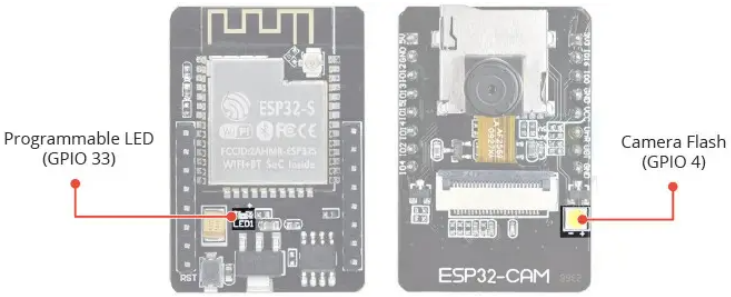


FIGURE 2.5 – LED Flash caméra.

2.2.8 Brochage ESP32-CAM

ESP32-CAM dispose de 16 broches comme le montre le brochage de la Figure 2.6. Ces broches sont polyvalentes et peuvent donc remplir différentes fonctions en fonction de la configuration.

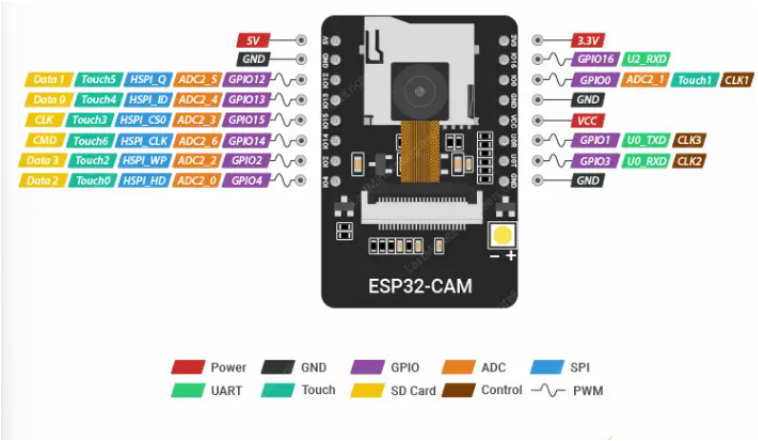


FIGURE 2.6 – Brochage ESP32-CAM.

1. **Broches GPIO :** La puce ESP32-S dispose au total de 32 broches GPIO, dont 10 sont disponibles pour une utilisation externe, comme illustréé à la Figure 2.7 tandis que le reste des broches est utilisé en interne pour la caméra et la PSRAM. Ces broches peuvent être assignées à différentes fonctions périphériques, telles que UART, SPI, ADC et Touch.

FIGURE 2.7 – Broches GPIO.

2. **Broches de carte MicroSD :** Les broches assignées à la carte microSD sont montrées à la Figure2.8 .

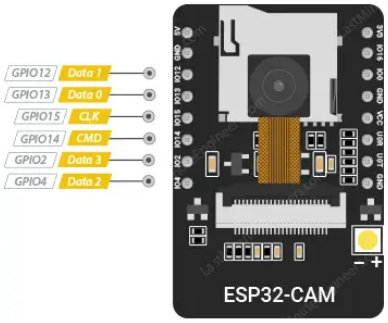


FIGURE 2.8 – Broches de carte MicroSD.

3. **Broches CAN :** Seules les broches du convertisseur ADC2 sont disponibles et accessibles sur la carte ESP32-CAM. Cependant, comme les broches ADC2 sont utilisées en interne par le Wi-Fi, elles ne peuvent pas être utilisées lorsque le Wi-Fi est activé. Les CAN offre une résolution de 12 bits, ce qui permet de convertir les signaux analogiques entre 0 à 4095. Cette plage de valeur correspond à une plage de tension entre 0 et 3,3V où 0 correspond à 0V et 4095 à 3,3V.

FIGURE 2.9 – Broches du convertisseur ADC2.

4. **Touches tactiles :** L'ESP32-CAM dispose de 7 broches GPIO capacitives de détection tactile, comme le montre la Figure2.10. Lorsqu'une charge capacitive (comme un doigt humain) se trouve à proximité de la broche GPIO, l'ESP32 détecte le changement de capacitance. Les broches de détection tactile de l'ESP32-CAM peuvent être utilisées dans diverses applications, notamment en interfaces utilisateur tactiles, le contrôle de la luminosité, le contrôle des gestes , la sécurité et accès basés sur la détection d'empreintes digitales.

FIGURE 2.10 – Touches tactiles.

5. **Broches SPI :** L'ESP32-CAM dispose d'un seul bus SPI (VSPI) pouvant fonctionner en mode maître ou esclave.Le bus SPI est un bus de communication série synchrone standard, utilisé pour connecter des microcontrôleurs à des périphériques externes. Il prend également en charge les fonctionnalités SPI à usage général suivantes :

   - 4 modes de synchronisation du transfert au format SPI
   - Jusqu'à 80 MHz et les horloges divisées de 80 MHz
   - Jusqu'à 64 octets FIFO

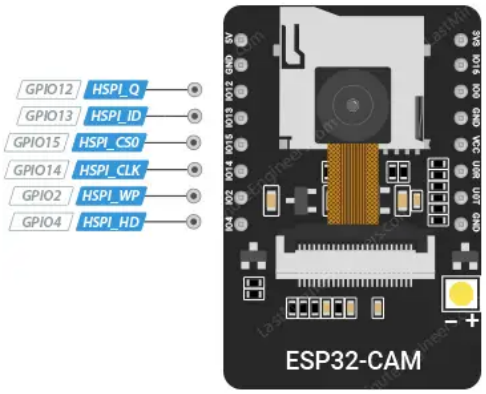


FIGURE 2.11 – Broches SPI.

6. **Broches UART :**

   Le module ESP32-CAM est équipé de deux interfaces UART, UART0 et UART2. L'UART0 est accessible via les broches GPIO 1 (TX) et GPIO 3 (RX), comme illustré à la Figure 2.12. L'UART2, en revanche, ne dispose que de la broche RX (GPIO 16) qui est disponible. Sa broche TX (GPIO 17) n'est pas accessible sur l'ESP32-CAM. Par conséquent, l'UART0 est le seul UART utilisable sur l'ESP32-CAM . De plus, étant donné que l'ESP32-CAM ne dispose pas d'un port USB, ces broches doivent être utilisées pour le flashage du firmware.

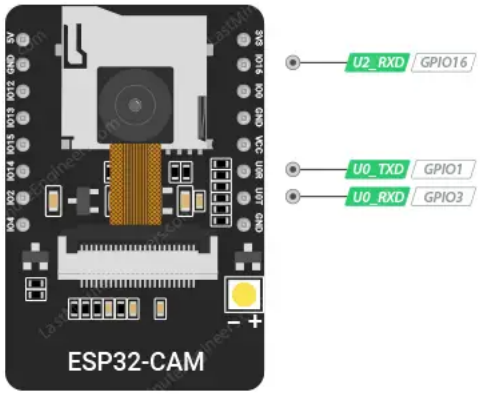


FIGURE 2.12 – Broches UART.

7. **Broches PWM :**

   L'ESP32-CAM dispose de 10 broches PWM (toutes les broches GPIO) comme illustré à la Figure 2.13. Ces broches peuvent être utilisées pour piloter divers périphériques tels que des moteurs DC, des moteurs pas à pas ou des servomoteurs, ainsi que des LED. Sur l'ESP32-CAM, il y a un total de 16 canaux PWM disponibles. Chaque canal PWM lui est associé une résolution (8 ou 16 bits) et une fréquence.

FIGURE 2.13 – Broches PWM.

8. **Broches GPIO RTC :**

- Les broches GPIO acheminées vers le sous-système RTC (Real-Time Clock) de faible puissance sont appelées broches GPIO RTC. Elles sont utilisées pour réveiller l'ESP32-CAM d'un état de sommeil profond lorsque le co-processeur Ultra Low Power (ULP) est en fonctionnement.
- Les broches GPIO RTC sont spécifiquement conçues pour permettre au co-processeur ULP de surveiller des événements externes, tels que des changements d'état de broche, des signaux d'horloge ou d'autres déclencheurs, même lorsque le processeur principal de l'ESP32-CAM est en mode de faible consommation d'énergie.
- L'utilisation des broches GPIO RTC permet de maintenir une certaine surveillance et une réactivité aux événements externes tout en conservant une efficacité énergétique élevée. Ces broches peuvent être configurées pour générer une interruption qui réveillera le processeur principal de l'ESP32-CAM lorsque l'ULP détecte un événement spécifique.

FIGURE 2.14 – Broches GPIO RTC.

9. **Broches de puissance :** L'ESP32-CAM peut être aliméntée avec une tension de 5V ou 3,3V comme le montre la Figure 2.15 . La broche VCC fournit normalement une tension de 3,3 V à partir du régulateur de tension intégré de l'ESP32-CAM. Cependant, il est possible de configurer cette broche pour fournir une tension de 5V en utilisant un cavalier de zéro ohm situé à proximité de la broche VCC. Cela permet de sélectionner la tension d'alimentation appropriée en fonction des besoins spécifiques de l'application.

FIGURE 2.15 – Broches de puissance.

2.2.9 Programmation de l'ESP32-CAM

La programmation de l'ESP32-CAM peut être réalisée en utilisant l'environnement de développement Arduino ou en utilisant l'IDE ESP-IDF (Espressif IoT Development Framework). La programmation de l'ESP32-CAM consiste en les étapes suivantes :

1. Installation du logiciel IDE Arduino ou l'IDE ESP-IDF sur votre ordinateur.
2. Configuration du logiciel : qui consiste à installer les bibliothèques nécessaires pour l'ESP32-CAM. Pour Arduino, on doit installer le support ESP32 et les bibliothèques spécifiques à la caméra. Pour ESP-IDF, on doit cofigurer l'environnement et installer les outils nécessaires.
3. Connexion de l'ESP32-CAM : l'esp32-CAM ne dispose pas de connecteur USB intégré. Pour connecter la carte à un ordinateur, on requiert un adaptateur série ou un module FTDI. Une fois connectée, il suffit de sélectionner le port correspondant et procéder au téléversement du code dans la mémoire Flash de la carte.
4. Écriture du code : on écrit le code dans votre environnement de développement en incluant les bibliothèques nécessaires pour l'ESP32-CAM, telles que la bibliothèque de la caméra, les bibliothèques GPIO et les bibliothèques de

communication sans fil (WiFi, Bluetooth, etc.). Il y aura lieu aussi d' installer d'autres bibliothèques à partir du gestionnaire de bibliothèque .

5. Compilation et téléversement : Une fois le code compilé, on peut téléverser vers la mémoire Flash de l'ESP32-CAM après avoir sélectionner le type de carte et le port correspondant.
6. Surveillance des résultats : Après le téléversement, on peut visualiser les sorties sur le moniteur série ou utiliser d'autres méthodes de débogage pour vérifier les résultats.

2.2.10 Utilisation de l'adaptateur FTDI

L'ESP32-CAM peut être connecter à un ordinateur via un module FTDI illustré à la Figure 2.16. Il s'agit d'un adaptateur USB vers série qui permet de communiquer avec l'ESP32-CAM via l'interface UART.

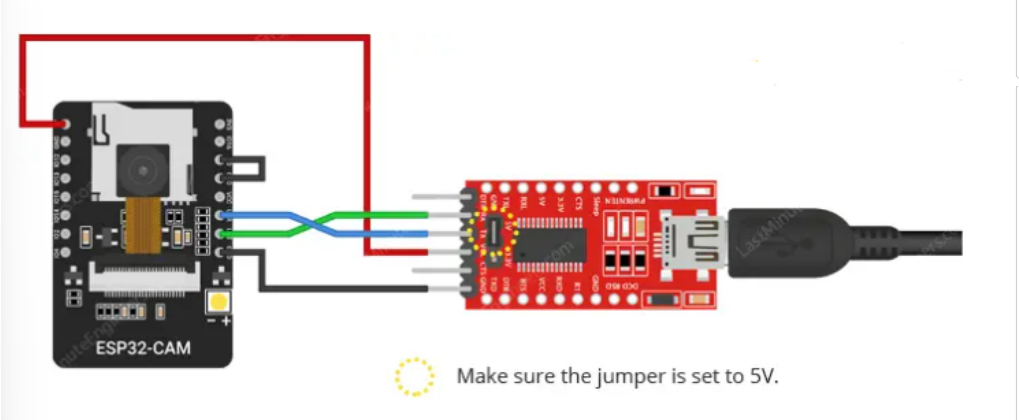


FIGURE 2.16 – Connection du module FTDI à l'ESP32-cam.

L'adaptateur FTDI dispose d'un cavalier qui permet de choisir entre 3,3 V et 5 V pour alimenter l'ESP32-CAM. Pour le flashage du code sur l'ESP32-CAM, il faut se placer en mode téléversement en reliant la broche GPIO 0 à la masse. Une fois le téléversement terminé, il faut revenir au mode normal en supprimant la mise à la masse de la broche GPIO 0.

FTDI	ESP32-CAM
RX	UOT
TX	UOR
Vcc	5V

UOT : U0_TXD (GPIO 3) **UOR** : U0_RXD (GPIO 1)

TABLEAU 2.1 – Table de correspondance des broches FTDI – ESP32-CAM.

2.3 Module de puissance L298N

Le module pilote L298N, montré à la Figure 2.17, est une breakbaoard utilisée pour le contrôle de moteur à courant continu (DC) et de moteur pas à pas. Il est couramment utilisé dans les projets de robotique et les véhicules télécommandés pour contrôler la direction et la vitesse des moteurs.

Il est basé sur le composant L298N qui est un double Pont-H conçu spécifiquement pour ce cas d'utilisation, est capable de contrôler deux moteurs à courant continu indépendamment, chacun pouvant fournir un courant maximum de 2 A avec une tension d'alimentation de 5 V à 35 V. Il peut également contrôler un moteur pas à pas unipolaire ou bipolaire avec une résolution allant jusqu'à 1/16 de pas. Le module L298N a également des fonctions de freinage dynamique, de protection thermique et de diagnostique de court-circuit.

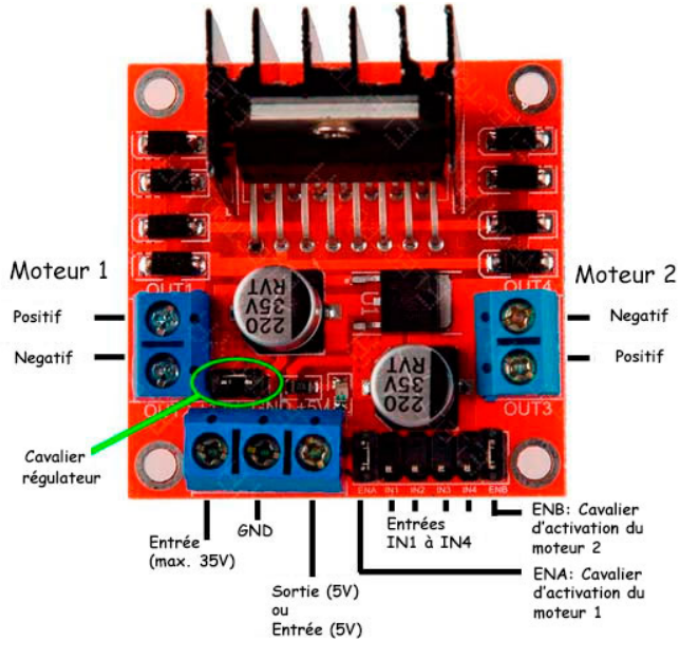


FIGURE 2.17 – Module pilote de moteur L298N.

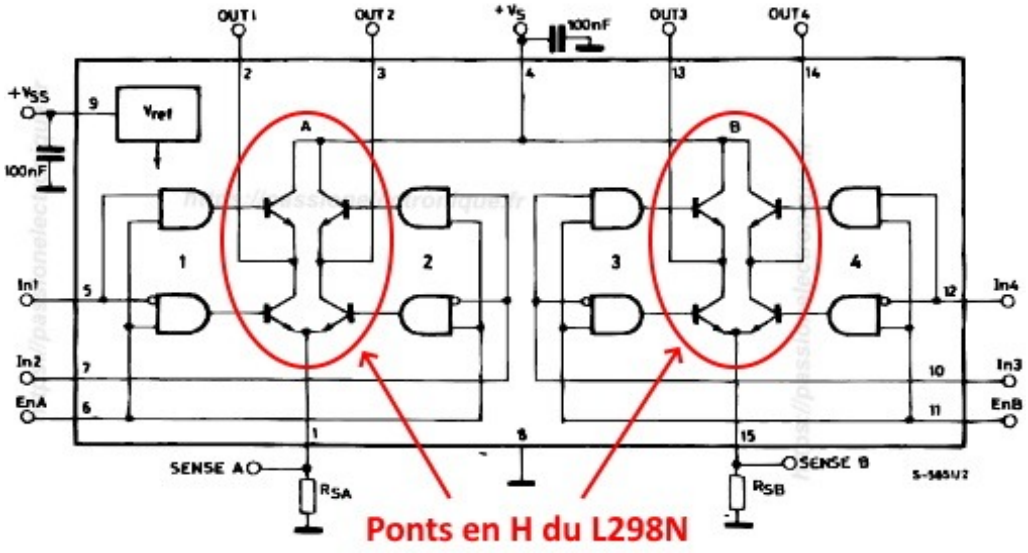


FIGURE 2.18 – Architecture interne du module L298N.

2.3.1 Les « Ponts en H »

Le pont en H est composant essentiel de l'électronique de puissance. Il s'agit d'une configuration à 4 transistors en commutation disposés en forme de lettre H qui permet d'inverser le sens de rotation des moteurs.

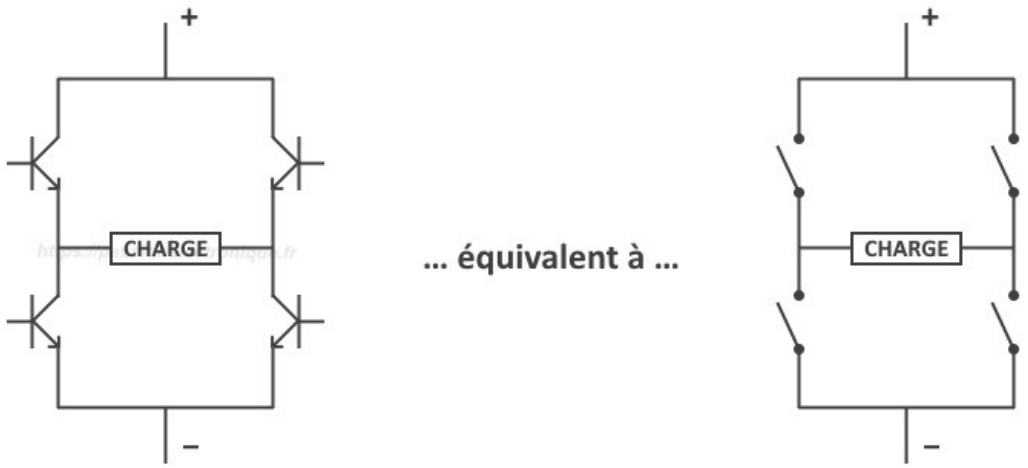


FIGURE 2.19 – Pont en H.

Son principe de fonctionnement repose sur l'activation des transistors deux par deux (ceux de sens opposés), comme illustré à la Figure 2.19. Ainsi, il est possible de contrôler le sens du passage du courant dans la charge, branchée « au milieu du H ». Ce changement de sens de courant permet aux moteurs à courant continu de pouvoir tourner dans de sens direct ou inverse.

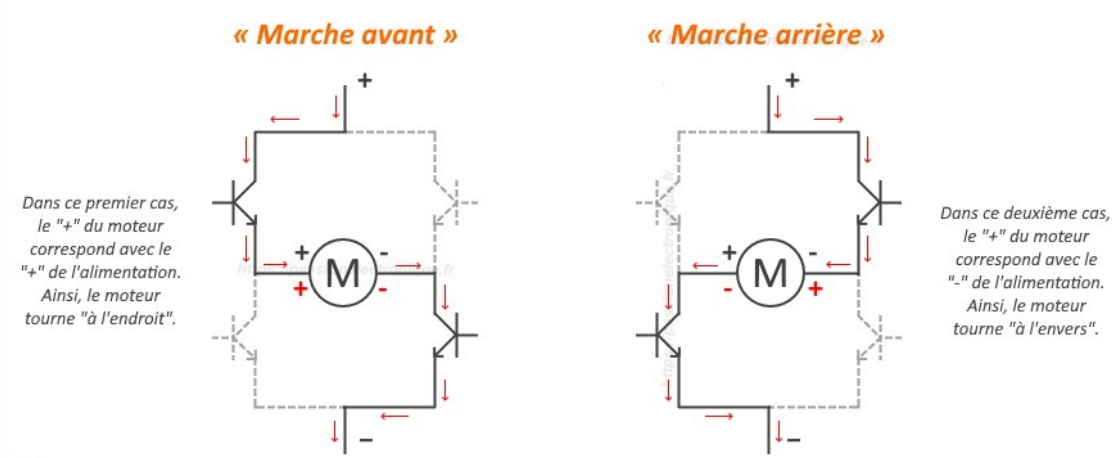


FIGURE 2.20 – Principe de fonctionnement du Pont en H.

2.3.2 Le pilotage PWM

Le pilotage du L298N se fait via un signal à rapport cyclique variable, pour pouvoir faire varier la vitesse des moteurs. En effet, la vitesse d'un moteur à courant continu est proportionnelle à sa tension d'alimentation, ou plus exactement, à sa tension moyenne. Ainsi, il suffit de faire varier la tension moyenne de l'alimentation fournie à un moteur, pour faire varier sa vitesse de rotation. Et pour faire varier une tension moyenne, on utilse la modulation en largeur d'impulsion ou PWM.

En effet, en PWM, on génère un signal de fréquence fixe, et de rapport cyclique variable. Et mathématiquement, la tension moyenne sera tout simplement égale à la tension max, multipliée par ce rapport cyclique. La Figure 2.21 donne une représentation imagée de ce concept de pilotage par PWM.

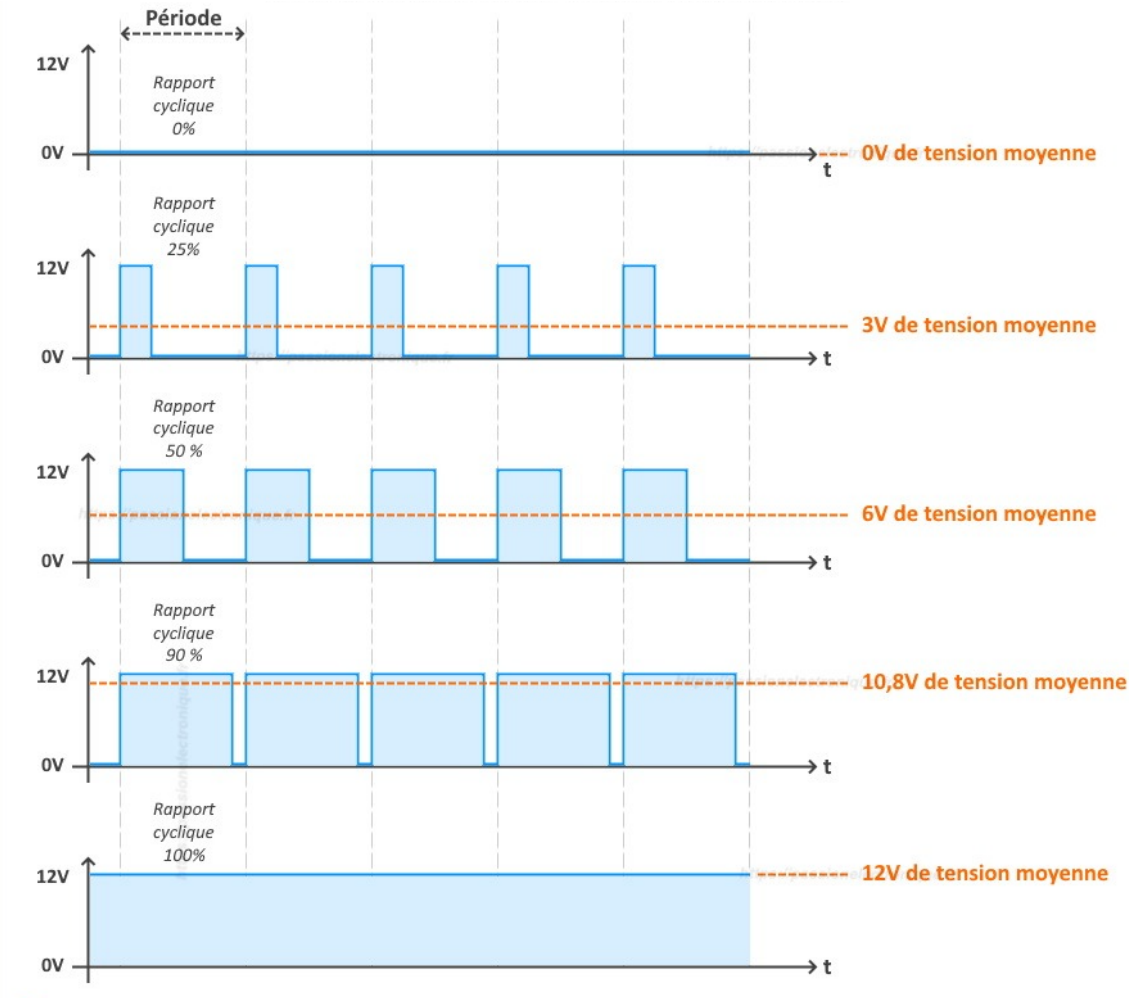


FIGURE 2.21 – Le pilotage PWM

2.3.3 Table de vérité du L298N (fonctionnement)

Le L298N dispose des entrées logiques suivantes :

- ENA et ENB : pour respectivement activer ou désactiver les pont A et B (les deux ponts en H de notre L298N, constitués de 4 transistors bipolaires chacun)
- IN1 et IN2 : pour définir le sens du courant en « sortie » dans le pont A pour le moteur A.
- IN3 et IN4 : pour définir le sens du courant en « sortie » dans le pont B pour le 2ème moteur B

Le fonctionnement du module L298N est basé sur sa table de vérité du Tableau 2.2, qui définit les états de ses entrées et sorties en fonction des signaux appliqués. L'état logique des signaux ENA et ENB détermine si les moteurs sont activés ou désactivés. Selon les combinaisons des signaux IN1, IN2, IN3 et IN4, les sorties OUT1, OUT2, OUT3 et OUT4 fournissent les signaux nécessaires pour contrôler la rotation et la direction des moteurs.

Tableau 2.2 – Table de vérite du pilote L298N.

ENA	IN1	EN2	Résultat en sortie
ENB	IN3	EN4	Résultat en sortie
L	X	X	Moteur en roue libre (à l'arrêt, sans frein)
H	L	L	Blocage du moteur (arrêt rapide, freinage fort)
	L	H	Marche arrière
	H	L	Marche avant
	H	H	Blocage du moteur (arrêt rapide, freinage fort)

X : peu importe L : état bas H : état bas

2.4 Moteur continu DC

Les moteurs à courant continu transforment l'énergie électrique en énergie mécanique de rotation, pour être précis. Mais ils peuvent également servir de générateur d'électricité en convertissant une énergie mécanique de rotation en énergie électrique.

Le moteur à courant continu est composé de deux parties principales : le rotor (induit) et le stator (inducteur), comme on peut le voir sur la partie gauche de la Figure 2.22.

Le collecteur, visible sur la partie droite de la Figure 2.22, est située sur l'axe de l'arbre du moteur à courant continu. Il se compose de deux pastilles métalliques auxquelles sont connectées les extrémités des bobines. C'est par le biais du collecteur et de petits éléments appelés charbons que le contact électrique est établi entre la pile d'alimentation et les bobines du moteur. Ces charbons ont pour rôle de permettre au courant de circuler dans les bobines en établissant un contact électrique direct.

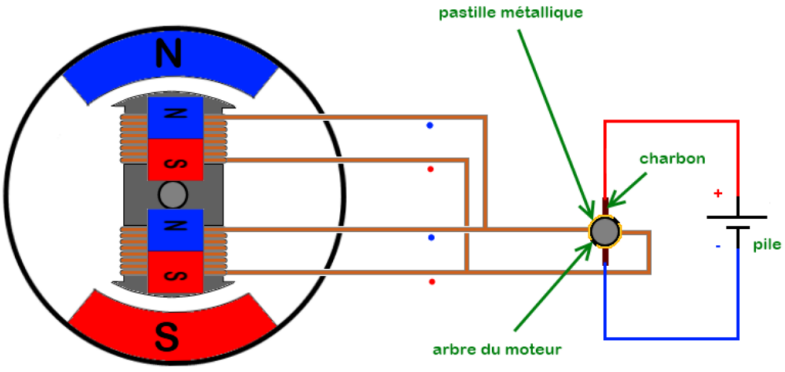


FIGURE 2.22 – Schéma d'un moteur à courant continu.

Cependant, lorsque nous avons besoin d'un moteur électrique pour un robot qui ne doit pas tourner rapidement, nous devons réduire sa vitesse de rotation. Cela peut être réalisé en utilisant un "frein" ou en le pilotant de manière appropriée. Cependant, même avec une vitesse réduite, le moteur ne pourra pas supporter des charges lourdes. Pour obtenir à la fois une réduction de vitesse et un couple plus élevé, nous utilisons un réducteur comme illustré sur la Figure 2.23. Le réducteur est composé d'engrenages qui réduisent la vitesse de rotation de l'axe du moteur tout en augmentant le couple de sortie. Ainsi, nous pouvons obtenir un moteur adapté aux besoins de notre robot.

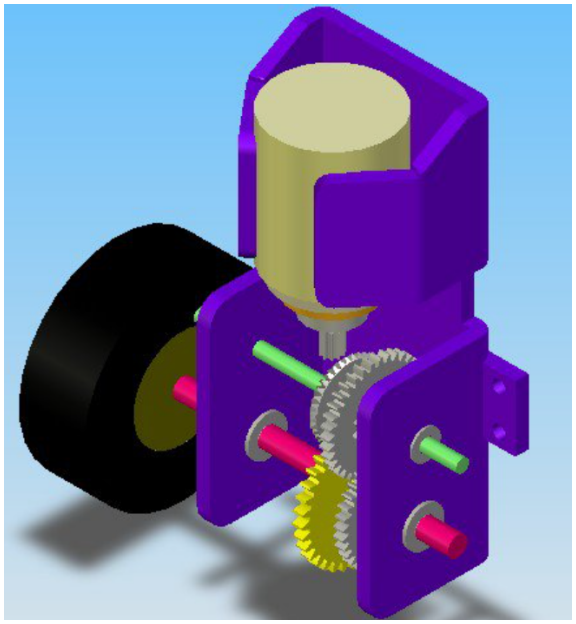


FIGURE 2.23 – Ensemble moteur, réducteur et roue.

2.5 Servo-moteur

Un servomoteur est un moteur capable de maintenir une opposition à un effort statique et dont la position est vérifiée en continu et corrigée en fonction de la mesure. C'est donc un système asservi. C'est un ensemble mécanique et électronique comprenant :

- un moteur à courant continu
- un réducteur en sortie de ce moteur diminuant la vitesse mais augmentant le couple .
- un potentiomètre (faisant fonction de diviseur résistif) qui génère une tension variable, proportionnelle à l'angle de l'axe de sortie .
- un dispositif électronique d'asservissement .
- un axe dépassant hors du boîtier avec différents bras ou roues de fixation. Les servos peuvent actionner les parties mobiles d 'un robot, drone, etc

2.5.1 Servomoteur type S90

Le servomoteur S90 est un servomoteur miniaturisé adapté aux systèmes de directions des petits hélicoptères, s voitures télécommandées ou autres petits véhicules télécommandés. Ces mini-servos peuvent aussi êtres utilisés pour des projets incluant des contrôles de tourelles ou des mouvements mécaniques (jusqu'à 180°) sur des pièces légères.

Un servomoteur est constitué de différents éléments comme illustré à la Figure 2.24
.

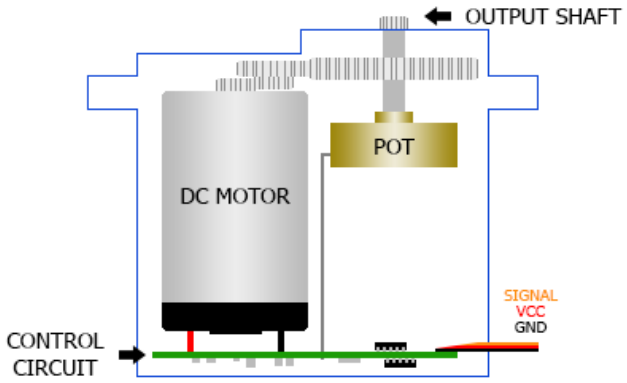


FIGURE 2.24 – Les constituants du servomoteur.

1. **Moteur électrique à courant continu (MCC) :** Le moteur à courant continu est alimenté par une tension constante et tourne à vitesse angulaire proportionnelle à sa tension d'alimentation, par exemple dans un servomoteur standard la tension est de 5 volts et donc le moteur tourne à vitesse élevée. Le problème c'est que comme le moteur est de petite taille, il est beaucoup moins efficace pour la production de couples. C'est la raison pour laquelle il est judicieux de placer un réducteur (train d'engrenages) sur l'axe du moteur pour fournir un bon couple.
2. **Train d'engrenages :**
   Il s'agit d'une cascade d'étages constitués chacun d'un couple de roues dentées. Chaque étage possède son rapport de réduction qui est égal au rapport du nombre de dents des deux roues ou pignons. Au total, le rapport de réduction est le produit de tous les rapports et peut atteindre typiquement une valeur de l'ordre de 400 ou 500.Sur un plan électromécanique, la puissance de sortie du moteur P est donnée l'équation 2.1 :

$$P = C_m.\Omega_m \tag{2.1}$$

   où :

   - C_m représente le couple sur l'axe du moteur ;
   - Ω_m est la vitesse angulaire de rotation de l'axe du moteur.

   Dans le cas où la puissance du moteur est entièrement transférée au palonnier, (mais en réalité, il y a toujours des pertes) alors la puissance est donnée par l'équation 2.2 :

$$P = C_p.\Omega_p \tag{2.2}$$

où :

- C_p représente le couple sur l'axe du palonnier ;
- Ω_p est la vitesse angulaire de rotation de l'axe du palonnier.

Aux pertes près, le produit $P = C.\Omega$ est constant. La réduction de vitesse est ainsi associée à la démultiplication du couple dans des rapports identiques.

3. **Un potentiomètre :**

   Le potentiomètre est relié au train d'engrenages par l'intermédiaire d'une roue dentée, pour tirer l'angle et la vitesse réelle du servomoteur c'est à dire la position réelle instantanée du servomoteur, il joue le rôle de capteur de position du palonnier. Il délivre une tension proportionnelle à la position angulaire, la tension exploitée par la carte électronique dans la boucle d'asservissement.

4. **Carte électronique :**

   La carte électronique a pour rôle de lire les impulsions électriques et lire aussi la position du moteur pour contrôler et corriger continuellement la position angulaire de l'axe de sortie. Ceci permet de fournir une loi de commande convenable et bien gérer l'asservissement.

2.5.2 Fonctionnement du servomoteur numérique :

Les servomoteurs numériques disposent aussi de trois fils. Deux fils d'alimentation, plus un fil de pilotage. Le fil de pilotage est utilisé pour transmettre des données numériques en série au servomoteur.

2.5.3 L'asservissement des servo-moteurs

C'est un moyen de gérer et corriger une commande en fonction d'une consigne et d'un capteur de position.

- on envoie au servomoteur un signal de commande qui défini l'angle désiré (consigne), et le moteur va corriger sans angle de départ pour tendre vers la consigne de l'utilisateur. Et il réalise cette opération en boucle.
- Pour pouvoir réaliser la correction de l'angle du bras, le servo utilise une électronique d'asservissement. Cette électronique est constituée d'un comparateur qui compare la position du bras du servo à la consigne.
- La position du bras est obtenue grâce à un potentiomètre couplé à l'axe du moteur.

- Après une rapide comparaison entre la consigne et valeur réelle de position du bras, le servomoteur (du moins son électronique de commande) va appliquer une correction si le bras n'est pas orienté à l'angle imposé par la consigne.

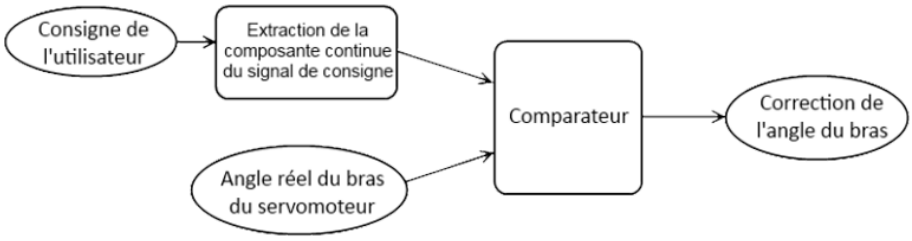


FIGURE 2.25 – Principe de l'asservissement des servomoteurs.

2.5.4 Servo-moteurs : Signal de commande

La consigne envoyée au servomoteur est un signal électronique de type PWM. Il dispose cependant de deux caractéristiques indispensables pour que le servo puisse fonctionner : la fréquence fixe et la durée de l'état haut

1. **Une fréquence fixe**
   - Le signal que nous allons devoir générer doit avoir une fréquence de 50 Hz. Autrement dit, le temps séparant deux fronts montants est de 20 ms.
   - La fonction analogWrite() de Arduino ne pourra donc pas utiliser cette fonction car on ne peut pas régler la fréquence. Par conséquent, il est nécessaire d'inclure la bibliothèque Servo pour contrôler la position du servomoteur. Elle gère automatiquement la génération du signal PWM avec la fréquence appropriée pour les servomoteurs, généralement autour de 50 Hz.

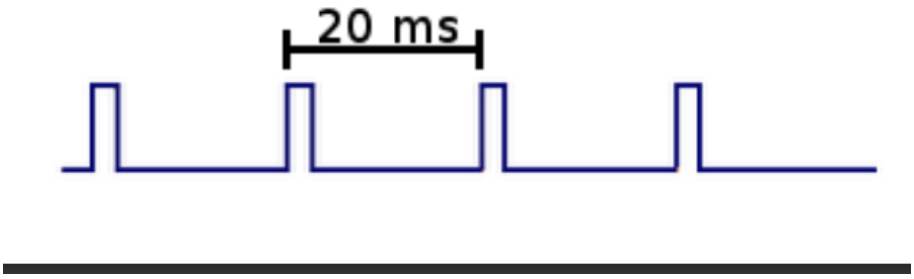


FIGURE 2.26 – le temps séparant deux fronts.

2. **La durée de l'état haut**
   - Cette durée indique au servomoteur l'angle précis qui est souhaité par l'utilisateur.

- Généralement, un signal PWM avec une durée d'état haut de 1 ms correspond à un angle de 0°, tandis qu'un signal PWM avec une durée d'état haut de 2 ms correspond à l'angle maximum que le servomoteur peut atteindre.

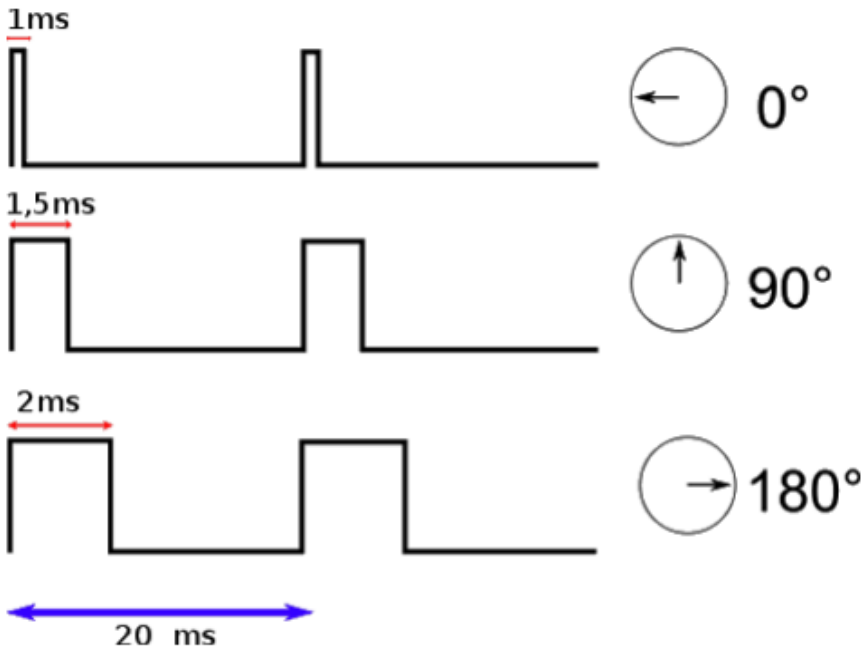


FIGURE 2.27 – LA DURÉE DE L'ÉTAT HAUT.

2.6 Serveur Web

Un serveur web est un logiciel permettant à des clients d'accéder à des pages web au format HTML à partir d'un navigateur installé sur un ordinateur distant, comme illustré à la Figure 2.28. Il est capable d'interpréter les requêtes HTTP arrivant sur le port associé au protocole HTTP (par défaut le port 80), et de fournir une réponse avec ce même protocole.

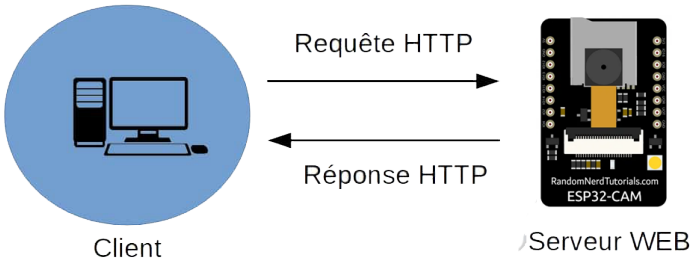


FIGURE 2.28 – Serveur Web schema général.

2.6.1 Etapes pour afficher une page Web

Voici les principales étapes pour obtenir une page Web provenant d'un serveur Web :

- L'utilisateur saisit l'adresse d'une page internet avec l'URL soit donc le nom de domaine + une page WEB.
- Puis le navigateur Web se connecte au serveur Web et demande le contenu de la page WEB avec la méthode GET du protocole HTTP.
- Le serveur Web retourne le code de la page WEB (HTML, CSS, JavaScript, etc).
- Le navigateur Web construit le contenu de la page Web qui s'affiche ensuite à l'écran de l'utilisateur.

2.6.2 Serveur Web synchrone

Un serveur web synchrone est un type de serveur Web qui gère les requêtes de manière synchrone, ce qui signifie qu'il traite chaque requête une par une et attend la fin du traitement d'une requête avant de passer à la suivante. Voici les étapes générales pour créer un serveur web synchrone :

1. Importation des bibliothèques : on commence par importer les bibliothèques nécessaires pour la gestion des requêtes HTTP. Dans le cas de l'ESP32-CAM, on peut utiliser les bibliothèques telle que WebServer.
2. Configuration du serveur : le serveur Web doit être initialisé en spécifiant le port sur lequel il écoutera les requêtes. Par exemple, on peut utilisé le port 80 pour les requêtes HTTP.
3. Définition des routes : on doit définir les routes ou les URL auxquelles le serveur doit répondre. Par exemple, on spécifier une route pour la page d'accueil ("/"), une autre pour une API ("/api") ou toute autre route qu'on souhaite gérer.
4. Gestion des requêtes : Pour chaque route définie, on doit spécifier la logique de traitement des requêtes entrantes. Cela peut inclure l'analyse des paramètres de requête, la récupération des données, l'exécution d'actions spécifiques et la génération de réponses appropriées.
5. Démarrage du serveur : Une fois le serveur configuré et toutes les routes définies, il démarre, écoute et gèreles requêtes entrantes.
6. Boucle principale : Dans le cas d'un serveur web synchrone, on doit exécuter une boucle principale pour gérer les requêtes de manière continue. Cela permet au serveur de rester actif et de traiter les requêtes entrantes au fur et à mesure qu'elles arrivent.

Il est important de noter que la gestion synchrone des requêtes signifie que le serveur ne peut traiter qu'une seule requête à la fois. Si une requête prend du temps à être traitée, cela peut affecter la capacité du serveur à répondre aux autres requêtes. Pour des performances optimales, il est recommandé d'utiliser un serveur web asynchrone tel que ESPAsyncWebServer, qui peut gérer plusieurs requêtes en parallèle.

2.6.3 Serveur Web asynchrone

Un serveur web asynchrone est un type de serveur web qui gère les requêtes de manière asynchrone, ce qui lui permet de traiter plusieurs requêtes en parallèle sans bloquer le flux d'exécution principal. Voici les étapes générales pour créer un serveur web asynchrone :

1. Importation des bibliothèques : on importer les bibliothèques nécessaires pour la gestion des requêtes HTTP asynchrones. Dans le cas de l'ESP32-CAM, on peut utiliser des bibliothèques telles que ESPAsyncWebServer ou AsyncWebServer.
2. Configuration du serveur : on initialisez le serveur web en spécifiant le port sur lequel il écoutera les requêtes, par exemple,le port 80 pour les requêtes HTTP.
3. Définition des routes : on définit les routes ou les URL auxquelles le serveur doit répondre. Par exemple, on peut spécifier une route pour la page d'accueil ("/"), une autre pour une API ("/api") ou toute autre route qu'on souhaite gérer.
4. Gestion des requêtes : Pour chaque route définie, on spécifie la logique de traitement des requêtes entrantes. On utilise des fonctions de rappel (callbacks) ou des fonctions asynchrones pour gérer le traitement des requêtes de manière non bloquante. On peut effectuer des opérations telles que l'analyse des paramètres de requête, la récupération des données, l'exécution d'actions spécifiques et la génération de réponses appropriées.
5. Démarrage du serveur : Une fois qu'on configuré le serveur et défini toutes les routes, on démarre le serveur pour qu'il écoute et gère les requêtes entrantes.
6. Boucle d'événements : Dans le cas d'un serveur Web asynchrone, on doit exécuter une boucle d'événements pour gérer les requêtes de manière continue. Cette boucle permet au serveur de traiter les requêtes entrantes de manière asynchrone, sans bloquer le flux d'exécution principal.

2.6.4 WebSocket

Le WebSocket est un protocole de communication bidirectionnel et en temps réel, qui permet l'établissement d'une connexion persistante entre un client et un serveur web. Contrairement au protocole HTTP traditionnel, qui suit le modèle requête-réponse, WebSocket permet une communication continue et bidirectionnelle, ce qui en fait une excellente option pour les applications nécessitant des mises à jour en temps réel.

Voici les étapes générales pour utiliser WebSocket dans une application :

1. Configuration du serveur : Sur le serveur, on met en place une infrastructure pour gérer les connexions WebSocket. Cela peut être réalisé en utilisant des bibliothèques spécifiques pour le langage de programmation utilisé, telles que Socket.IO, ws ou asyncio en Python.
2. Établissement de la connexion : Dans votre application cliente, on établit une connexion WebSocket avec le serveur en utilisant l'URL appropriée. Cela peut être fait en utilisant des bibliothèques clientes WebSocket telles que WebSocket API en JavaScript.
3. Gestion des événements : on définit des écouteurs d'événements pour gérer les événements liés à la connexion WebSocket, tels que l'ouverture de la connexion, la réception de messages du serveur et la fermeture de la connexion. On peut implémenter des fonctions de rappel pour traiter ces événements.
4. Échange de messages : Une fois la connexion établie, on peut envoyer et recevoir des messages entre le client et le serveur via la connexion WebSocket. Les messages peuvent être de différents types, tels que des chaînes de texte ou des données binaires. III faut veiller à définir un protocole de communication clair pour l'échange de messages entre les deux extrémités.
5. Gestion de la déconnexion : la gestion de la fermeture de la connexion WebSocket doit prise en compte, que ce soit de manière explicite ou en gérant les déconnexions inattendues. On doit s'assurer de nettoyer les ressources et d'informer les parties concernées de la fermeture de la connexion.

WebSocket offre une flexibilité et une facilité d'utilisation pour les applications en temps réel, telles que les applications de chat en direct, les tableaux de bord en temps réel et les jeux en ligne.

2.7 Bibliothèques utilisées dans le code Arduino du système de surveillance

2.7.1 Async TCP

Il s'agiy d'une une approche de communication asynchrone utilisant le protocole TCP. Il permet d'établir des connexions TCP de manière asynchrone, ce qui signifie que le flux d'exécution principal du programme n'est pas bloqué lors de l'envoi ou de la réception de données sur la connexion TCP. Cela permet de gérer efficacement de multiples connexions TCP en parallèle.

Voici les étapes générales pour utiliser Async TCP dans une application :

1. Importation des bibliothèques : on commence par importer les bibliothèques nécessaires pour la gestion des connexions TCP asynchrones. Selon le langage de programmation utilisé, des bibliothèques spécifiques peuvent être disponibles, telles que asyncio en Python ou libuv en C++.
2. Configuration du serveur ou du client : Selon les besoins de l'application, on peut configurer un serveur ou un client utilisant Async TCP. Pour un serveur, on doit spécifier le port sur lequel il écoutera les connexions entrantes, tandis pour un client, on doit fournir l'adresse IP et le port du serveur auquel on souhaite se connecter.
3. Gestion des événements : on utilise des mécanismes asynchrones tels que les boucles d'événements (event loops) pour gérer les événements liés à la communication TCP. Ces événements peuvent inclure l'ouverture de la connexion, la réception de données, l'envoi de données, la fermeture de la connexion, etc. Des rappels (callbacks) ou des coroutines asynchrones doivent être définis pour gérer ces événements.

2.7.2 io stream

IO stream est un concept utilisé en programmation pour représenter un flux de données entrant ou sortant. Il s'agit d'une abstraction qui permet de manipuler des flux de données de manière uniforme, indépendamment de leur source ou de leur destination réelle.

Les IO Streams sont couramment utilisés pour lire des données à partir de sources externes (comme des fichiers, des sockets réseau, des connexions série, etc.) ou pour écrire des données vers des destinations externes. Ils offrent une interface cohérente pour effectuer des opérations de lecture et d'écriture, quel que soit le type de données ou le support utilisé.

Les IO Streams peuvent être divisés essentiellement en deux catégories :

- Les streams d'entrée (input streams) : Ils permettent de lire des données à partir d'une source externe. Par exemple, un FileInputStream peut être utilisé pour lire des données à partir d'un fichier, ou un ObjectInputStream peut être utilisé pour lire des objets sérialisés à partir d'un flux de données.
- Les streams de sortie (output streams) : Ils permettent d'écrire des données vers une destination externe. Par exemple, un FileOutputStream peut être utilisé pour écrire des données dans un fichier, ou un ObjectOutputStream peut être utilisé pour écrire des objets sérialisés vers un flux de données.

Les IO Streams fournissent des méthodes de lecture et d'écriture de base, ainsi que des fonctionnalités supplémentaires telles que la mise en mémoire tampon (buffering) pour améliorer les performances, la compression/décompression de données et le chiffrement/déchiffrement.

2.7.3 sstream

La classe sstream fait référence à la bibliothèque "sstream" en C++, qui fournit des fonctionnalités pour la manipulation de chaînes de caractères en mémoire. Cette bibliothèque permet de lire et d'écrire des données dans des objets de flux de chaînes de caractères, appelés stringstream.

La classe sstream permet de traiter une chaîne de caractères comme un flux d'entrée ou de sortie, ce qui facilite la conversion de données entre des types différents, la création de chaînes de caractères formatées et bien d'autres opérations. Elle est incluse dans la bibliothèque standard C++ <sstream>.

Voici quelques fonctionnalités clés de la classe sstream :

- Conversion de données : La classe sstream permet de convertir des données de différents types en chaînes de caractères et vice versa. Par exemple, on peut convertir un entier en une chaîne de caractères à l'aide de l'opérateur d'insertion («) ou convertir une chaîne de caractères en un entier à l'aide de l'opérateur d'extraction (»).
- Formatage de chaînes de caractères : La classe sstream permet de créer des chaînes de caractères formatées en utilisant des opérations similaires à celles utilisées pour l'entrée/sortie sur les flux classiques. On peut insérer des variables, des valeurs constantes et des spécificateurs de format pour formater les données selon nos besoins.

- Lecture et écriture dans les flux : La classe sstream fournit des fonctionnalités pour lire et écrire des données dans les flux de chaînes de caractères. On peut utiliser également les opérateurs d'insertion («) et d'extraction (») pour écrire et lire des données respectivement, tout comme on le ferait avec les flux d'entrée/sortie standard.

L'utilisation de la classe sstream est très utile lorsqu'on a besoin de manipuler des chaînes de caractères et de travailler avec des données dans un format spécifique. Elle offre une flexibilité et une facilité d'utilisation pour la conversion de données et le formatage des chaînes de caractères en C++.

2.7.4 esp camera.h

Le fichier "esp camera.h" est un fichier d'en-tête de la bibliothèque "esp32-camera" utilisée pour la gestion de la caméra intégrée de l'ESP32-CAM. Cette bibliothèque facilite l'initialisation, la configuration et l'utilisation de la caméra pour capturer des images ou des vidéos.

Voici quelques fonctionnalités clés fournies par la bibliothèque "esp32-camera" via le fichier "esp camera.h" :

- Initialisation de la caméra : on peut utiliser la fonction esp err t esp camera init(const camera config t* config) pour initialiser la caméra avec la configuration spécifiée. La structure camera config t contient les paramètres tels que la résolution, la fréquence d'images, la balance des blancs, etc.
- Capture d'images : La bibliothèque "esp32-camera" permet de capturer des images à partir de la caméra. La fonction esp err t camera capture(), permet d'enregistre une image dans le buffer de capture.
- Capture de vidéos : En utilisant les fonctions fournies, on peut capturer des vidéos en configurant les paramètres appropriés, tels que la résolution, la fréquence d'images, etc. La bibliothèque offre également des fonctions pour démarrer et arrêter l'enregistrement vidéo.
- Gestion des paramètres de la caméra : on peut peut accéder et configurer les paramètres de la caméra, tels que la résolution, la fréquence d'images, l'exposition, la balance des blancs, etc., en utilisant les fonctions appropriées. Par exemple, l' esp err t camera set framesize(framesize t size) permet définir la résolution de la caméra.
- Configuration des broches de la caméra : La bibliothèque permet de spécifier les broches utilisées pour la communication avec la caméra. Les broches de données (data pins), les broches de synchronisation (clock pins), etc., peuvent être configurées en utilisant les fonctions appropriées.

2.7.5 Arduino.h

Le fichier Arduino.h est un fichier d'en-tête inclus dans l'IDE Arduino qui contient les définitions et les fonctions essentielles pour le développement de projets Arduino. Il est inclus automatiquement dans chaque sketch Arduino lors de la compilation.

Le fichier "Arduino.h" fournit des fonctionnalités de base, telles que la gestion des broches d'E/S, les fonctions de temporisation, les fonctions de communication série, les fonctions mathématiques, etc. Il facilite la programmation des cartes Arduino en fournissant une interface simple et cohérente.

Voici quelques fonctionnalités fournies par le fichier "Arduino.h" :

- Gestion des broches d'E/S : Le fichier "Arduino.h" définit des macros et des fonctions pour configurer et manipuler les broches d'E/S numériques et analogiques. Par exemple, On peut utiliser les fonctions pinMode(), digitalWrite() et digitalRead() pour configurer et interagir avec les broches numériques.
- Gestion des temporisations : Le fichier "Arduino.h" fournit des fonctions pour créer des retards et des temporisations dans le code. On peut utiliser les fonctions delay() et millis() pour introduire des pauses dans l'exécution du programme.
- Communication série : Le fichier "Arduino.h" offre des fonctions pour la communication série avec d'autres périphériques. On peut utiliser les fonctions Serial.begin(), Serial.print(), Serial.read(), etc., pour configurer et communiquer via les ports série (UART) des cartes Arduino.
- Fonctions mathématiques : Le fichier "Arduino.h" inclut des fonctions mathématiques courantes, telles que abs(), sqrt(), map(), etc., pour effectuer des opérations mathématiques dans nos projets.
- Fonctions d'interruption : Le fichier "Arduino.h" propose des fonctions pour gérer les interruptions matérielles. On peut utiliser les fonctions attachInterrupt() et detachInterrupt() pour configurer des interruptions sur les broches d'E/S.

En plus de ces fonctionnalités, le fichier "Arduino.h" inclut également d'autres bibliothèques et définitions nécessaires pour le fonctionnement des cartes Arduino, telles que la gestion de la mémoire, la gestion des librairies, etc.

2.7.6 WiFi.h

est une bibliothèque intégrée dans l'IDE Arduino qui permet de gérer les fonctionnalités WiFi sur les cartes Arduino compatibles avec la connectivité WiFi, telles que l'ESP8266 et l'ESP32.

Cette bibliothèque offre des fonctionnalités pour configurer et gérer les connexions WiFi, se connecter à des réseaux WiFi, créer des points d'accès WiFi, effectuer des opérations de communication réseau, etc.

Voici quelques fonctionnalités clés de la bibliothèque "WiFi.h" dans l'IDE Arduino :

- Connexion à un réseau WiFi : La bibliothèque permet de se connecter à un réseau WiFi en spécifiant le nom du réseau (SSID) et le mot de passe associé. On peut utiliser la fonction WiFi.begin(ssid, password) pour initier la connexion et vérifier l'état de la connexion avec les méthodes WiFi.status() et WiFi.isConnected().
- Configuration du mode d'opération : Vous pouvez configurer le mode d'opération WiFi en tant que client (station) ou point d'accès (access point). Pour configurer l'ESP8266 ou l'ESP32 en tant que point d'accès, On peut utiliser les méthodes WiFi.softAP(ssid, password) et WiFi.softAPConfig(local ip, gateway, subnet).
- Gestion des événements WiFi : La bibliothèque "WiFi.h" offre des fonctions de rappel (callbacks) pour gérer les événements WiFi, tels que la connexion réussie, la déconnexion, la détection de nouveaux réseaux disponibles, etc. On peut utiliser les fonctions WiFi.onEvent() et WiFi.onStationModeConnected() pour définir des fonctions de rappel personnalisées.
- Communication réseau : La bibliothèque "WiFi.h" fournit des méthodes pour effectuer des opérations de communication réseau, telles que l'ouverture de sockets TCP ou UDP, l'envoi et la réception de données, etc. On peut utiliser les classes WiFiClient et WiFiServer pour établir des connexions réseau et communiquer avec d'autres périphériques.

2.8 Technologies HTML, CCS et Javascript

Les technologies HTML, CSS et JavaScript sont les piliers fondamentaux du développement web. Voici une brève description de chacune de ces technologies :

- HTML (HyperText Markup Language) : HTML est le langage de balisage standard utilisé pour structurer le contenu d'une page web. Il définit la structure logique des éléments tels que les titres, les paragraphes, les liens, les images et les tableaux. HTML utilise des balises pour marquer et organiser le contenu, permettant aux navigateurs web de l'interpréter correctement.
- CSS (Cascading Style Sheets) : CSS est un langage de feuilles de style utilisé pour décrire la présentation et l'apparence visuelle d'une page web. Il permet de définir des règles pour contrôler le positionnement, la mise en page, les couleurs, les polices de caractères et d'autres aspects de l'aspect graphique d'un site web. CSS est utilisé en conjonction avec HTML pour styliser les éléments marqués avec des balises HTML.

- JavaScript : JavaScript est un langage de programmation polyvalent utilisé pour ajouter des fonctionnalités interactives à une page web. Il permet de manipuler le contenu HTML et CSS, de réagir aux actions de l'utilisateur, de valider les formulaires, de créer des animations, de charger du contenu dynamiquement et de communiquer avec des serveurs pour obtenir des données en temps réel. JavaScript est exécuté côté client, ce qui signifie qu'il s'exécute dans le navigateur web de l'utilisateur.

  En combinant ces trois technologies, on peut créer des sites web interactifs et attrayants. HTML fournit la structure du contenu, CSS gère la présentation visuelle, et JavaScript ajoute la logique et l'interactivité. Il est important de noter que ces technologies sont souvent utilisées en tandem avec d'autres frameworks et bibliothèques, tels que Bootstrap, React, Angular et Vue.js, pour faciliter et accélérer le développement web.

2.9 Video et streaming

L'ESP32-CAM est un module basé sur l'ESP32 qui est capable de capturer des images et des vidéos à partir de sa caméra intégrée. Il peut être utilisé pour réaliser des projets de streaming vidéo en temps réel.

Pour réaliser un streaming vidéo avec l'ESP32-CAM, voici les étapes générales à suivre :

1. Configuration de la caméra : Utilisez les bibliothèques appropriées pour initialiser et configurer la caméra intégrée de l'ESP32-CAM. Par exemple, On peut utiliser la bibliothèque "CameraWebServer" ou "ESP32 Camera" pour faciliter la configuration de la caméra.
2. Capture d'images ou de vidéos : Utilisez les fonctions fournies par les bibliothèques de la caméra pour capturer des images ou des vidéos à partir de la caméra. Vous pouvez définir la résolution, la qualité, la fréquence d'images, etc., en fonction de nos besoins.
3. Diffusion du flux vidéo : Vous avez plusieurs options pour diffuser le flux vidéo en temps réel :
   - Serveur Web : On peut utiliser les bibliothèques de serveur Web intégrées dans l'IDE Arduino, telles que "ESPAsyncWebServer" ou "WebServer", pour créer un serveur web sur l'ESP32-CAM. Vous pouvez ensuite envoyer les images ou les vidéos capturées aux clients via des requêtes HTTP.

- Protocole RTSP : Le protocole Real-Time Streaming Protocol (RTSP) est couramment utilisé pour le streaming vidéo en temps réel. On peut utiliser des bibliothèques telles que "ESP32-RTSP" pour mettre en place un serveur RTSP sur l'ESP32-CAM et permettre aux clients de se connecter et de recevoir le flux vidéo.

4. Affichage du flux vidéo côté client : Les clients peuvent afficher le flux vidéo en utilisant des lecteurs vidéo compatibles avec les protocoles HTTP ou RTSP. On peut utiliser des lecteurs vidéo HTML5 pour le streaming via HTTP, ou des lecteurs RTSP pour le streaming via RTSP.

Il est important de noter que la mise en place d'un streaming vidéo en temps réel peut être complexe et nécessite une bonne compréhension des concepts et des protocoles impliqués. Assurez-vous de consulter les exemples, les tutoriels et la documentation spécifiques à l'ESP32-CAM pour obtenir des instructions détaillées et adaptées à votre configuration.

2.10 Conclusion

Ce chapitre a fourni une description détaillée de l'ESP32-CAM, des composants matériels de notre voiture robot et des bibliothèques logicielles utilisées dans notre application. Nous avons exploré les caractéristiques de l'ESP32-CAM, telles que sa caméra intégrée, ses capacités de connectivité Wi-Fi et ses interfaces GPIO. Nous avons également examiné les composants matériels essentiels pour la voiture robot, tels que les servomoteurs, les moteurs DC et le module de commande L298N.

D'autre part, nous avons présenterons les technologies clés utilisées pour créer une interface utilisateur Web, notamment le HTML, le CSS et JavaScript. Ces technologies permettent de concevoir une page web interactive pour contrôler notre voiture robot à distance. La compréhension de ces différents parties du projet est essentielle pour la réalisation de notre système de surveillance à distance utilisant une voiture robot équipée d'une carte ESP32-CAM.

Chapitre 3

Conception du système de surveillance à base de l'ESP32-CAM

3.1 Introduction

La conception d'un système de télésurveillance basé sur la carte ESP32-CAM équipant une voiture robot implique plusieurs étapes allant de la mise en place et le contrôle des différents composants à l'aide de l'ESP32-CAM tels que les moteurs, le servo-moteur, la LED embarquée à la programmation de l'interface utilisateur en langage HTML. Dans ce chapitre, nous allons détailler ces différentes phases de conception, en expliquant les choix de conception et les méthodes utilisées pour réaliser un système fonctionnel de télésurveillance à distance.

3.2 Etapes de conception du système de surveillance

3.2.1 Commande des moteurs DC par l'ESP32-CAM

Pour contrôler les moteurs à courant continu avec l'ESP32-CAM, nous avons interfacé celle-ci avec le module L298N comme le montre la Figure 3.1.

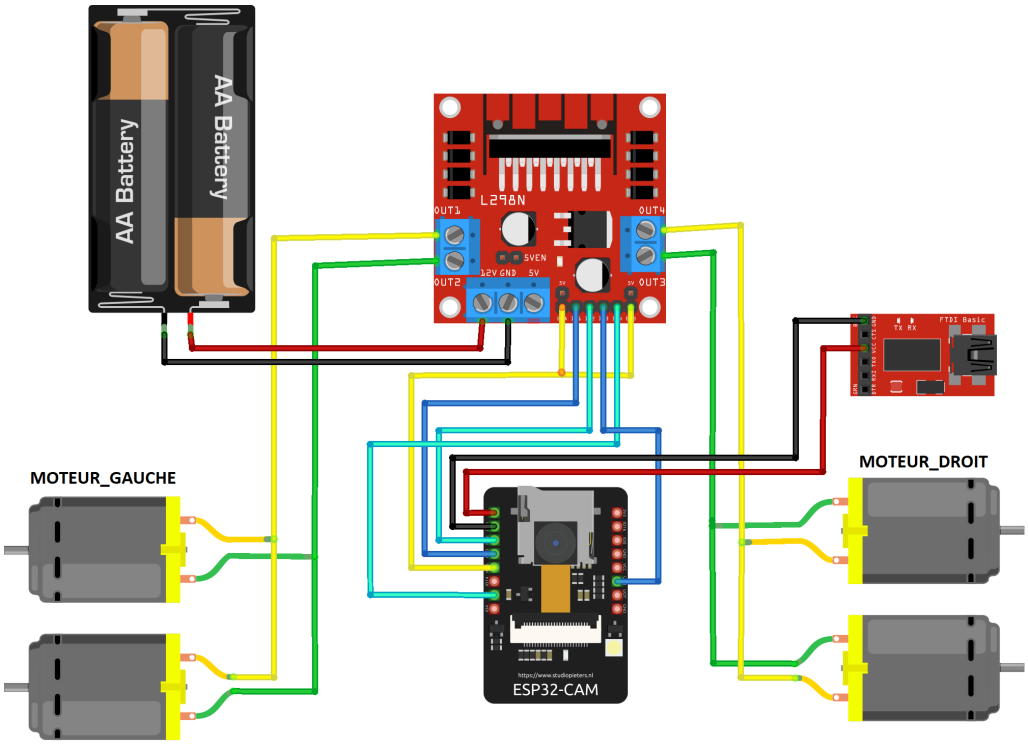


FIGURE 3.1 – Schéma de connexion de driver moteur à l'ESP32-cam .

Le Tableau 3.1 présente la correspondance des broches de la carte ESP32-CAM avec celles du module L298N. Il est à noter que nous avons affecté une seule et même broche Enable pour commander la vitesse des deux moteurs.

Module L298N	ESP32-CAM
ENA - ENB	15
IN1	13
IN2	12
IN3	14
IN4	02

TABLEAU 3.1 – Table de correspondance des broches L298N- ESP32CAM.

Les moteurs gauches et droits sont connectés aux borniers OUT1/OUT2 et OUT3/OUT4 respectivement.

Le logigramme présenté à la Figure 3.2 représente le code permettant de contrôler la vitesse et les différents mouvements de la voiture robot. Ce code est téleversé vers l'ESP32-CAM au moyen module FTDI connecté au port USB de l'ordinateur. La vitesse des moteurs est contrôlée à l'aide d'un signal PWM dont la fréquence est de 1 Khz avec une résolution de 8 bits. Par conséquent, la plage des valeurs de vitesse est comprise entre 0 et 255. Le signal PWM est envoyé aux pins ENA et ENB du module L298N.

Pour la commande des différentes directions de la voiture robot, les signaux appropriés sont envoyés aux pins IN1, IN2, IN3 et IN4 du driver L298N.

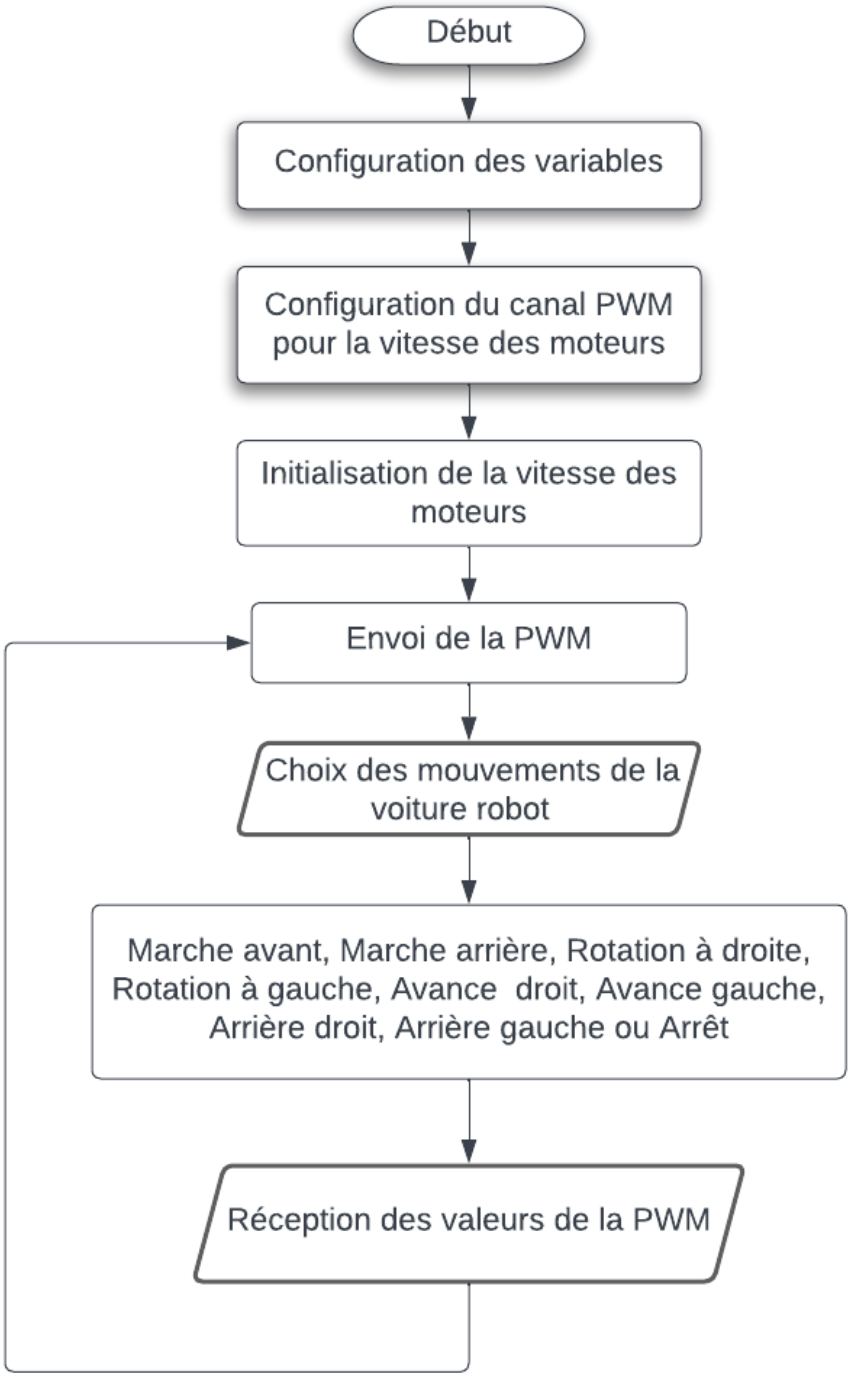


FIGURE 3.2 – Logigramme de commande de vitesse et des directions de la voiture robot.

3.2.2 Commande du servo-moteur S90 par l'ESP32-CAM

Le contrôle de la position du servomoteur permettra d'orienter la camera embarquée de l'ESP32-CAM dans différentes directions pour couvrir la zone surveillée souhaitée.

Le servomoteur S90 est interfacé à la carte ESP32-CAM comme illustré à la Figure 3.3 et selon la table de correspondance du Tableau 3.2

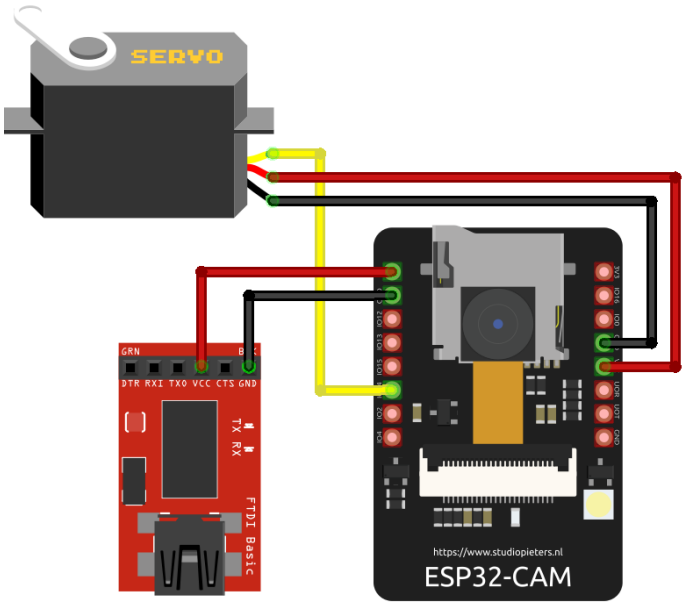


FIGURE 3.3 – Schéma de connexion du servo-moteur à l'ESP32-CAM.

Servomoteur	ESP32-CAM
Signal	GPIO 14
Gnd	Gnd
Vcc	Vcc

TABLEAU 3.2 – Table de correspondance des broches servomoteur – ESP32-CAM.

Le code téléversé dans l'ESP32-CAM permet d'envoyer les signaux de commande appropriés à la broche de signal du servomoteur. Le code est réprésenté par le logigramme de la Figure 3.4.

Le servomoteur est contrôlé en utilisant un signal PWM dont la fréquence est de 50 Hz et la résolution est de 16 bits.

Dans le code, la position actuelle du servomoteur est initialisé à un angle de 4000. Cet angle est progressivement ajusté pour atteindre un angle cible donné. Si l'angle cible est supérieur à l'angle actuel, le servomoteur tourne dans le sens des aiguilles d'une montre jusqu'à atteindre la valeur l'angle cible désirée.

Si l'angle cible est inférieur à l'angle actuel, le servomoteur tourne dans le sens trigonométrique jusqu'à atteindre la valeur l'angle cible désirée.

Il convient de noter que dans cette partie, la valeur de l'angle cible a été introduite manuellement dans le code pour les besoins de test. Nous verrons par la suite que le contrôle de la position du servomoteur se fait depuis l'interface Web de notre application. Pour cela, l'utilisateur déplace un slider dédié au servomoteur et la valeur de l'angle cible correspondante est envoyée vers la voiture robot via un websocket .

3.2.3 Commande de la LED Flash caméra de l'ESP32-CAM

La broche 4 de l'ESP32-CAM est configurée pour contrôler la LED Flash intégrée à la caméra. Un signal PWM de fréquence 1 kHz et de résolution 8 bits est appliqué à cette broche pour réguler l'intensité lumineuse de la LED Flash, offrant ainsi un contrôle sur la luminosité des prises de vue de la caméra. La plage des valeurs pour l'éclairage est également de 0 à 255, où 0 représente l'extinction complète de l'éclairage et 255 correspond à l'intensité lumineuse maximale.

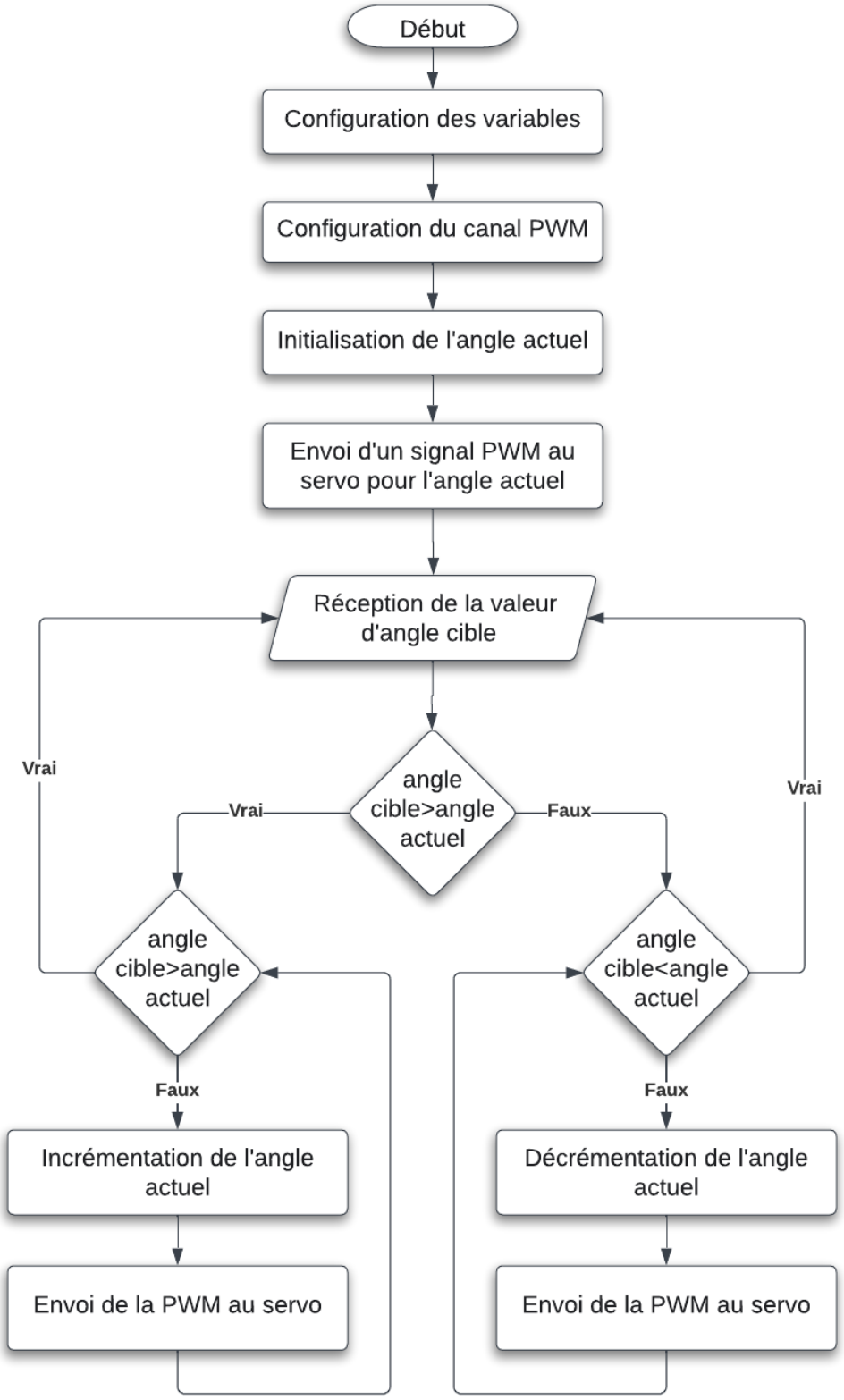


FIGURE 3.4 – Logigramme de commande du servomoteur.

3.2.4 Acquisition de la vidéo par la caméra embarquée de l'ESP32-CAM

Dans le code,les paramètres de configuration sont spécifiés pour initialiser la caméra ESP32-CAM. Les images sont capturées par la caméra puis envoyées aux clients WebSocket connectés. La caméra diffuse en direct un flux vidéo sur l'interface Web.

3.3 Assemblage de la voiture robot

A présent, nous allons décrire les différentes étapes d'assemblage de la voiture robot et cela après avoir testé séparément les différentes parties qui constituent notre système de surveillance à distance.

1. Soudage des fils de connexion aux bornes des 04 moteurs DC TT.
2. Fixation les moteurs DC TT sur le châssis inférieur de la voiture robot selon la configuration illustrée à la Figure 3.5.

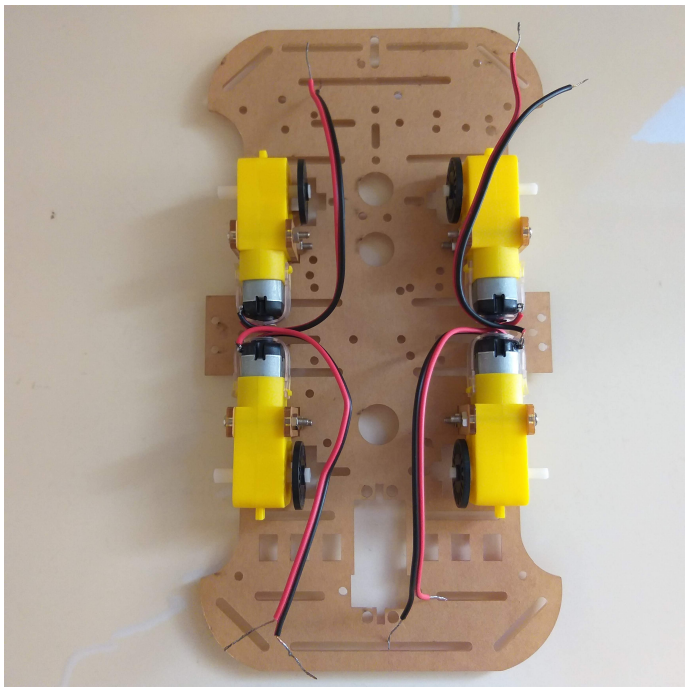


FIGURE 3.5 – Montage des 04 moteurs TT sur le châssis inférieur de la voiture robot.

3. Montage des roues dans l'axe des moteurs puis installation du châssis supérieur. Les fils de connexion des moteurs droits et ceux des moteurs gauches sont acheminés vers l'extérieur à travers les orifices du chassis supérieur, comme montré à la Figure 3.6.

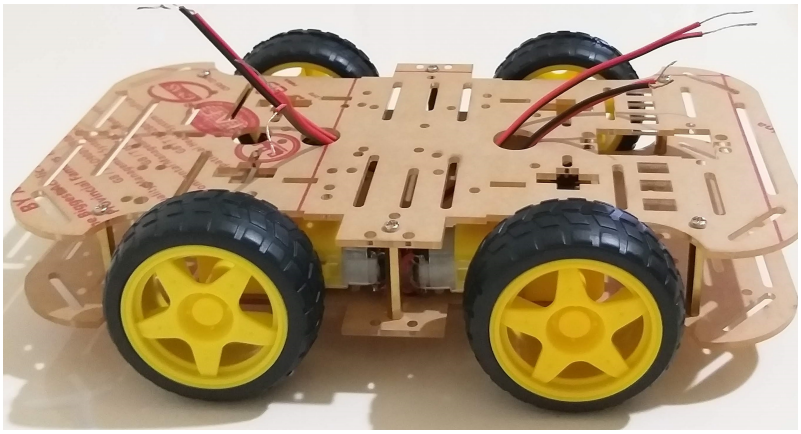


FIGURE 3.6 – Montage du châssis supérieur et des 4 roues de la voiture robot.

4. Mise en place du module de puissance L298N sur le chassis supérieur puis raccordement des moteurs droits et des moteurs gauches aux borniers moteurs correspondants (M1 et M2), comme illustrée à la Figure 3.7. L'alimentation du module de puissance est fournit par deux piles de 3,7V logées dans un compartiment à pile.

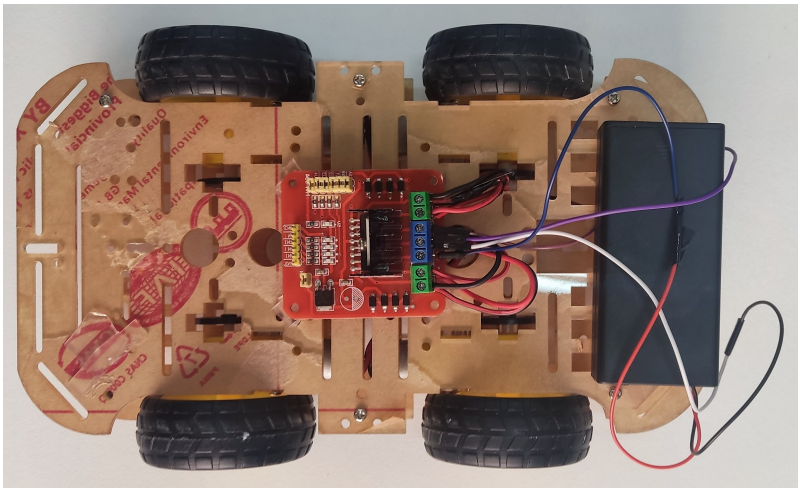


FIGURE 3.7 – Mise en place du module L298N sur la voiture robot.

5. Installation du servomoteur et de l'ESP32-CAM sur le chassis supérieur de la voiture robot comme le montre la Figure 3.8. Une alimentation disctincte a été utilisée pour fournir la tension et le courant nécessaire au bon fonctionnement du servomoteur. Dans notre cas, il s'agit d'une pile de 9V et un régulateur de tension 5V. L'idéal serait d'utiliser un convertisseur UBEC 5V/2A.

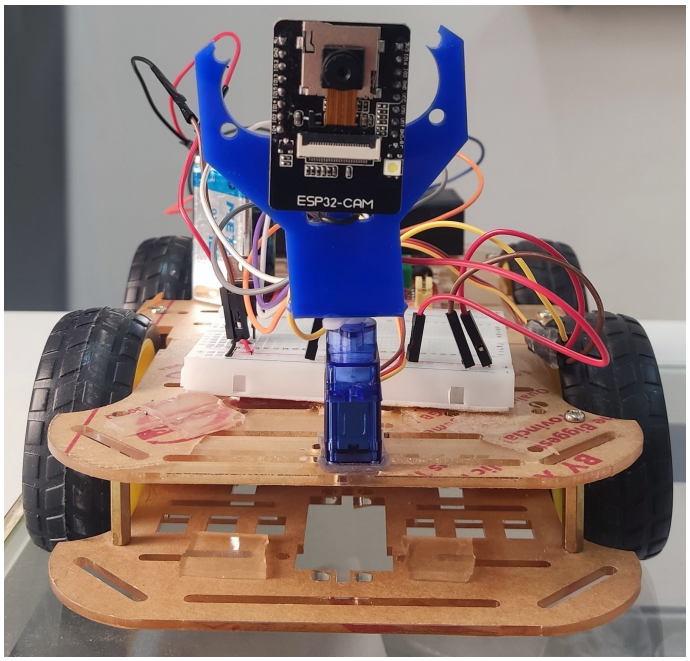


FIGURE 3.8

La Figure 3.9 offre une vue de dessus de la voiture robot assemblée.

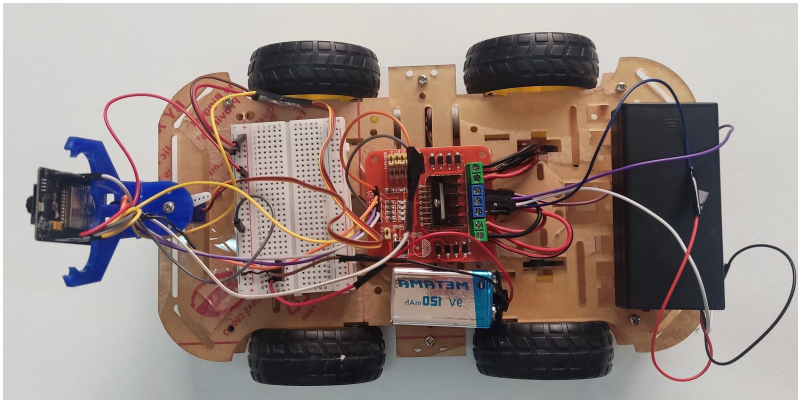


FIGURE 3.9

La Figure 3.10 illustre le schéma de connexion qui représente la manière dont les différents composants de la voiture robot ont été interconnectés.

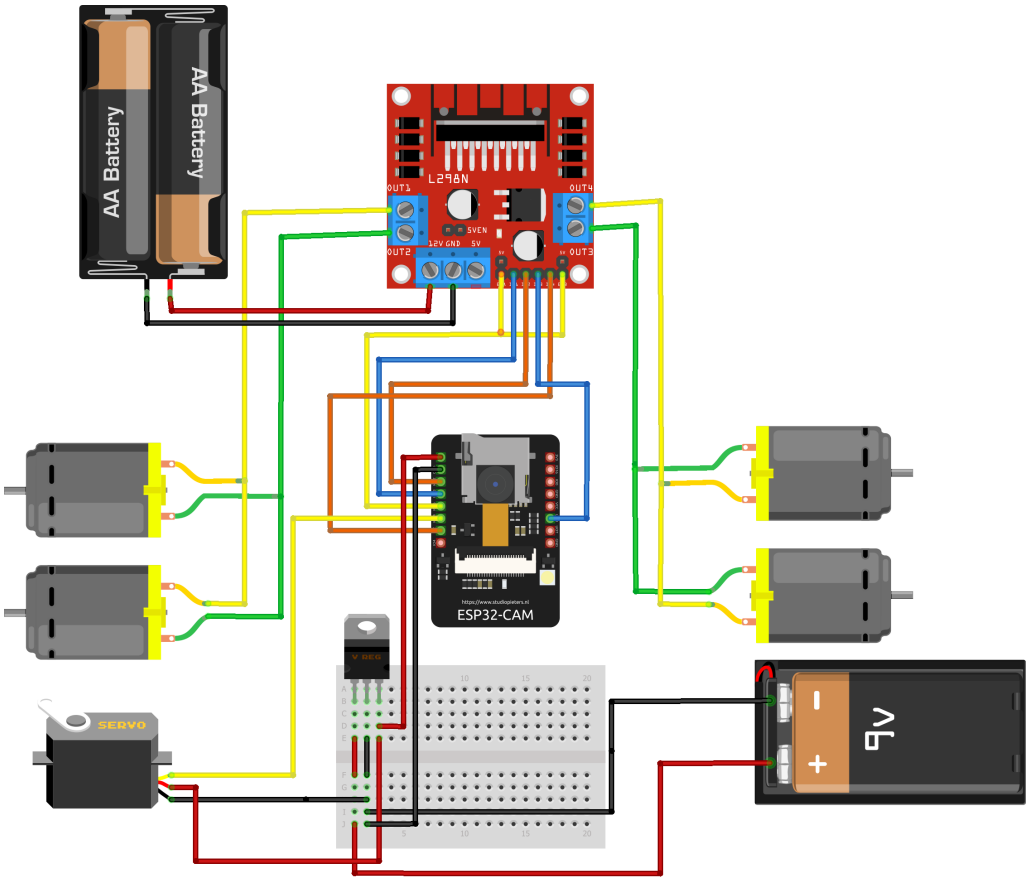


FIGURE 3.10 – Schéma de montage des différents composants équipant la voiture robot.

3.4 Conception de l'interface Web pour le contrôle de la voiture robot

A présent, nous allons décrire les différentes étapes de conception de l'interface Web pour notre application de surveillance basée sur l'ESP32-CAM.
Notre interface Web, présentée à la Figure 3.11, a été développée spécifiquement pour contrôler la voiture robotisée via un smartphone et offre les fonctionnalités suivantes :

- la navigation de la voiture robot et le contrôle de la position du servomoteur
- l'acquisition de la vidéo en temps réel
- l'éclairage nécessaire fournit par la LED de flash caméra pour l'obtention d'une meilleure qualité d'images vidéo même dans les conditions de faible luminosité.

L'interface graphique a été crée à l'aide des langages HTML, CSS et JavaScript et le code correspondant est inclut dans notre code Arduino. Nous y avons implémenté les différentes fonctionnalités de contrôle en utilisant les éléments interactifs tels que les boutons et les sliders (ou curseurs) . Les boutons servent à contrôler la navigation de la voiture robotisée et les sliders à contrôler le pivotement du servomoteur pour diriger la position de la camera embarquée dans différentes directions, la vitesse de la voiture et l'intensité de la lumière émise par la LED flash.Ces éléments de contrôle réagissent au touché sur un écran tactile, comme celui d'un smartphone dans notre cas.

FIGURE 3.11 – Interface utilisateur de contrôle de la voiture robotisée.

Le logigramme de la Figure 3.12 offre une vue d'ensemble claire du fonctionnement global de votre système.

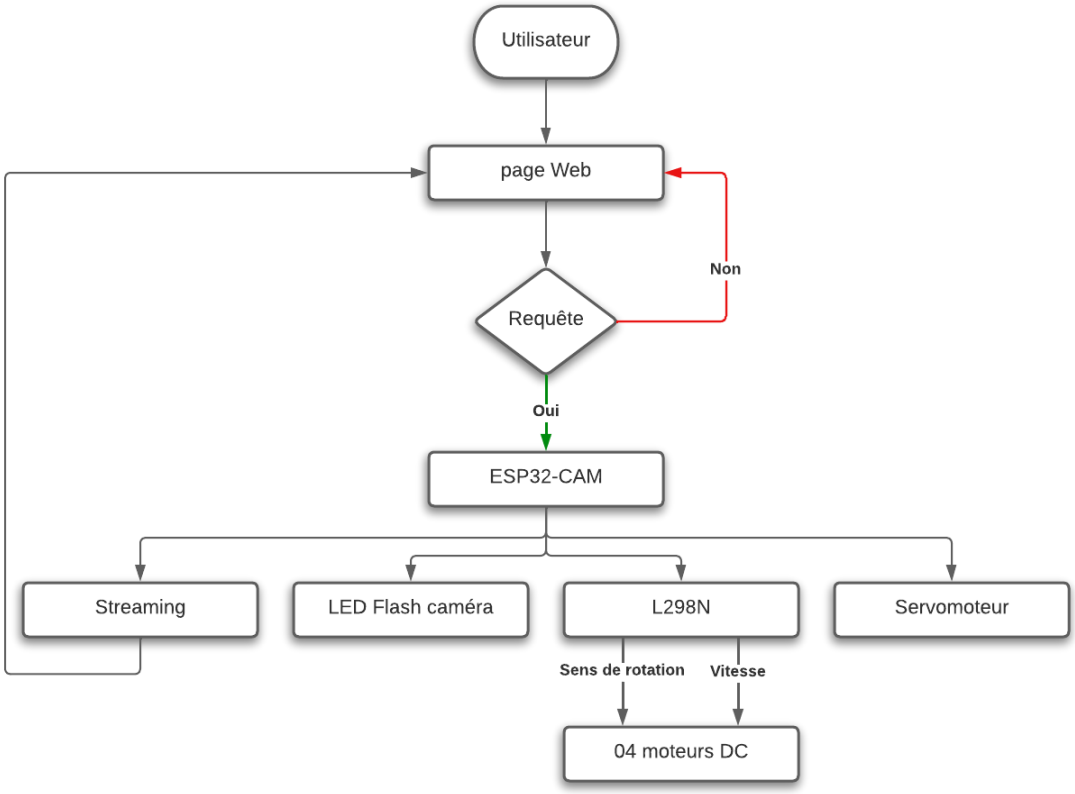


FIGURE 3.12 – Bloc diagramme du système de surveillance à distance basé sur l'ESP32-CAM.

Nous avons créer un serveur asynchrone qui permet à l'utilisateur d'accéder à l'interface de contrôle de la voiture robotisée via un navigateur web. Le serveur asynchrone permet de traiter efficacement les requêtes HTTP de manière asynchrone, ce qui signifie qu'il peut traiter plusieurs requêtes simultanément sans bloquer l'exécution du programme.

Pour établir une communication entre l'interface Web et la voiture robotisée, nous avons eu recours à la technologie WebSocket. Les WebSockets permettent d'établir une communication en temps réel et bidirectionnelle entre un navigateur Web et un serveur. Dans notre code, deux connexions WebSocket ont été établies, l'une pour la caméra pour réception des images en provenance de la caméra embarquée de l'ESP32-CAM pour les afficher sur la page HTML, et l'autre pour l'evoie des commandes liées au contrôle de la voiture robotisée.

3.5 Tests et résultats

Après avoir effectué une série de test et d'ajustements , nous sommes parvenus aux résultats prévus pour notre application de surveillance à distance utilisant une voiture robotisée à base de la carte ESP32-CAM.

Dans ce qui suit, nous allons décrire les différentes étapes pour lancer et tester notre système de surveillance.

- Le code Arduino écrit pour notre application est compilé et téléversé dans l'ESP32-CAM. Il permet de configurer celle-ci en tant que point d'accès Wi-Fi avec un nom du réseau SSID : **"MyWiFiCar"** et un mot de passe **"12345678"**. Il permet de récupérer l'adresse IP locale **"192.168.4.1"** de l'ESP32-CAM qui sera affichée sur le moniteur série de l'IDE Arduino lors de son démarrage.
- Une fois le réseau Wi-Fi **"MyWiFiCar"** détecté sur le smartphone, on peut s'y connecter et ensuite ouvrir un navigateur Web pour saisir l'adresse IP locale de l'ESP32-CAM dans la barre d'adresse, ce qui permet de charger notre interface Web.
- On peut voir alors un aperçu en direct de la caméra embarquée sur la voiture robotisée, ce qui permet d'avoir une surveillance à distance en temps réel de la zone couverte. Voici sur la Figure 3.13 une capture d'image extraite de la vidéo diffusée en direct sur notre interface, prise dans le parking de la faculté de Technologie de l'université de Saida.
- Les boutons de contrôle de la navigation, tels que les flèches directionnelles permettent de déplacer la voiture dans la direction souhaitée : avance, arrière, à droite, à gauche et les directions obliques.
- Le slider du servomoteur permet de régler la position de la caméra embarquée, offrant ainsi la possibilité de surveiller différentes zones avec précision. Le slider de vitesse permet d'ajuster la vitesse de déplacement de la voiture robot, offrant une plus grande flexibilité dans les mouvements. Enfin, le slider de lumière permet de contrôler l'intensité de l'éclairage fourni par la LED FLASH, permettant ainsi d'adapter l'éclairage aux besoins spécifiques de l'environnement de surveillance.

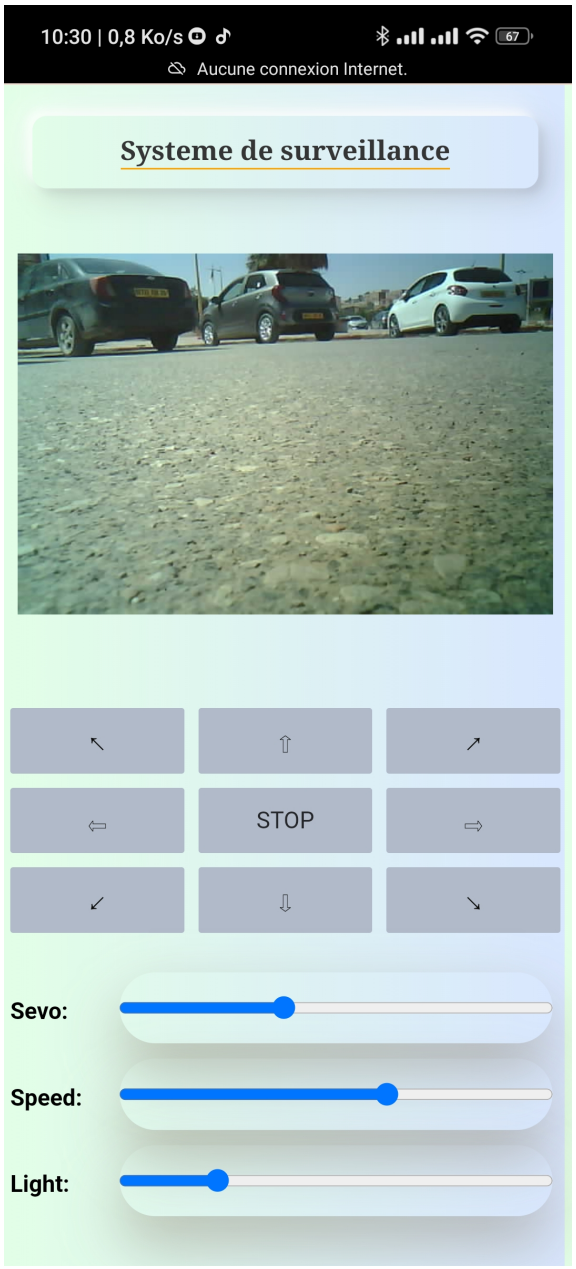


FIGURE 3.13 – Image extraite de la vidéo prise par la voiture robot.

3.6 Conclusion

Ce chapitre a abordé en détail les différentes parties du système de surveillance basé sur l'ESP32-CAM, l'interface web développée ainsi que les tests réalisés. En ce qui concerne les tests, nous avons effectué des essais pour vérifier le bon fonctionnement des moteurs DC, du servomoteur, de l'éclairage, l'acquisition de la vidéo en temps réel ainsi que la communication entre l'ESP32-CAM et l'interface web. Les résultats ont confirmé que le système était opérationnel et que la voiture robot pouvait être contrôlée à distance avec succès à l'aide de l'interface web.

Conclusion générale

Dans le cadre de ce projet, nous avons exploré divers aspects de la conception et du développement d'une voiture robot utilisant l'ESP32-CAM. Nous avons examiné les divers composants matériels essentiels d'une voiture robot, tels que les moteurs, les servomoteurs et le module pilote L298N, qui permet le mouvement et la direction de la voiture robot. Nous avons appris à intégrer ces composants de manière à créer un système fonctionnel et réactif.

Du point de vue logiciel, nous avons utilisé des bibliothèques spécifiques pour le contrôle des périphériques, la gestion du Wi-Fi et la création d'interfaces utilisateur Web interactives. Nous avons utilisé des langages et des technologies tels que le C++, l'HTML, le CSS et JavaScript, pour offrir des fonctionnalités avancées à notre voiture robot.

L'utilisation d'un robot équipé d'une carte ESP32-CAM dans un système de surveillance représente une évolution significative dans le domaine de la sécurité. Cette technologie offre des fonctionnalités de surveillance à distance, de mobilité et d'interactivité, contribuant ainsi à assurer la sécurité et la protection des environnements surveillés de manière efficace et innovante.

Dans la continuité de ce projet, les vidéos capturées peuvent être soit stockées localement sur une carte SD, soit transférées vers un serveur distant en utilisant une connexion Wi-Fi.Il est également possible d'appliquer des traitements d'image ou de vidéo sur les données acquises. Cela inclut des fonctionnalités telles que la détection de mouvement, la reconnaissance d'objets, le suivi d'objets. Ces fonctionnalités offrent de nombreuses possibilités pour la surveillance à distance, la sécurité, la vision par ordinateur et d'autres domaines et bien d'autres domaines où l'analyse d'images et de vidéos en temps réel est requise.

Voici quelques pistes pour les futurs projets :

- Intégration de capteurs supplémentaires : L'utilisation de capteurs supplémentaires tels que des capteurs de température, des capteurs de gaz, des capteurs de qualité de l'air ou des capteurs de mouvement permettrait d'étendre les capacités de surveillance de la voiture robot. Ces capteurs pourraient détecter les variations environnementales et contribuer à des applications telles que la surveillance de la qualité de l'air, la détection d'incendies ou la surveillance environnementale.
- Intégration de l'intelligence artificielle pour l'analyse des données : l'utilisation de techniques d'intelligence artificielle pour l'analyse des données pourrait permettre d'extraire des informations précieuses à partir des flux de données en temps réel. Les travaux futurs pourraient se concentrer sur l'intégration de modèles d'apprentissage automatique ou de réseaux neuronaux pour une analyse plus avancée des données de surveillance.

Notre projet a été une expérience enrichissante et stimulante. Il nous a ouvert les portes de l'univers passionnant de la robotique et des objets connectés, et nous a donné les compétences nécessaires pour poursuivre notre exploration dans ce domaine en constante évolution.

Références Bibliographiques

1. Sunitha, M., Meghana, A., Jayendra, D., Ashritha, C. & Lalitha, B. Advance Surveillance Robot with Robotic Arm Control Over Iot. **7** (2022).
2. Reddy, V. S. N., Kumar, S. P., Venkat, B. & Priyanka, J. S. IoT based social distance checking robot using Esp32-Cam. *AIP Conference Proceedings* **2407,** 020011. issn : 0094-243X. https://doi.org/10.1063/5.0074056 (2023) (1er déc. 2021).
3. Band, S., Javeri, R., Kale, V., Morope, A. & Rudrawar, S. *Military Surveillance System Based on IOT* in *2022 Fourth International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT)* 2022 Fourth International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT) (déc. 2022), 1-6.
4. Cahyono, F. Y. A., Suharto, N. & Mustafa, L. D. Design and Build a Home Security System based on an ESP32 Cam Microcontroller with Telegram Notification. *Journal of Telecommunication Network (Jurnal Jaringan Telekomunikasi)* **12.** Number : 2, 58-64. issn : 2654-6531. http://jartel.polinema.ac.id/index.php/jartel/article/view/296 (2023) (29 juin 2022).
5. Shankar, V. Design of an Intelligent Combat Robot for War Fields.
6. Legrand. *Déclenchement caméra* https://www.f-legrand.fr/scidoc/docmml/sciphys/arduino/synchrobasler/synchrobasler.html (2023).
7. *ESP32 Wi-Fi & Bluetooth MCU I Espressif Systems* https://www.espressif.com/en/products/socs/esp32 (2023).
8. *ESP32-CAM Video Streaming and Face Recognition with Arduino IDE | Random Nerd Tutorials* https://randomnerdtutorials.com/esp32-cam-video-streaming-face-recognition-arduino-ide/ (2023).
9. Ferrero, F. Moteurs et transistors MOS.
10. Venu, D. N. IOT Surveillance Robot Using ESP-32 Wi-Fi CAM & Arduino. **11.**
11. *Présentation du module ESP32-CAM* https://sigmdel.ca/michel/ha/esp8266/ESP32-CAM_01_fr.html (2023).

12. *The Internet of Things with ESP32* http://esp32.net/ (2023).

13. *Caméra Wi-Fi ESP32-CAM avec Arduino* Tropratik. Section : Arduino. https://tropratik.fr/camera-wi-fi-esp32-cam-avec-arduino (2023).

14. *Qu'est-ce que le WebSocket ?* IONOS Digital Guide. https://www.ionos.fr/digitalguide/sites-internet/developpement-web/quest-ce-que-le-websocket/ (2023).

15. Tibermacine, C. & Tibermacine, C. Introduction aux Websockets.

16. Eskimon. *Le moteur à courant continu* Le blog d'Eskimon. Publisher : Le blog d'Eskimon. https://eskimon.fr/tuto-arduino-601-le-moteur-%C3%A0-courant-continu (2023).

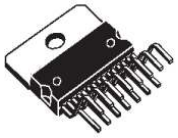
Annexe A

Datasheet CI L298

L298

DUAL FULL-BRIDGE DRIVER

- OPERATING SUPPLY VOLTAGE UP TO 46 V
- TOTAL DC CURRENT UP TO 4 A
- LOW SATURATION VOLTAGE
- OVERTEMPERATURE PROTECTION
- LOGICAL "0" INPUT VOLTAGE UP TO 1.5 V (HIGH NOISE IMMUNITY)



Multiwatt15

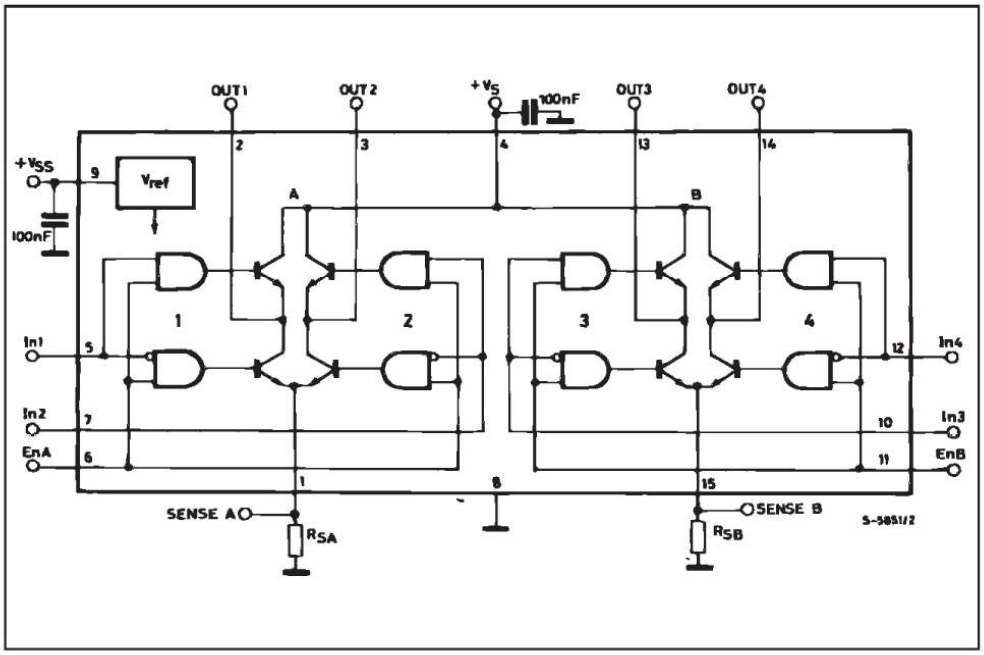
PowerSO20

ORDERING NUMBERS : L298N (Multiwatt Vert.)
L298HN (Multiwatt Horiz.)
L298P (PowerSO20)

DESCRIPTION

The L298 is an integrated monolithic circuit in a 15-lead Multiwatt and PowerSO20 packages. It is a high voltage, high current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the input signals. The emitters of the lower transistors of each bridge are connected together and the corresponding external terminal can be used for the connection of an external sensing resistor. An additional supply input is provided so that the logic works at a lower voltage.

BLOCK DIAGRAM



PIN FUNCTIONS (refer to the block diagram)

MW.15	PowerSO	Name	Function
1;15	2;19	Sense A; Sense B	Between this pin and ground is connected the sense resistor to control the current of the load.
2;3	4;5	Out 1; Out 2	Outputs of the Bridge A; the current that flows through the load connected between these two pins is monitored at pin 1.
4	6	V_S	Supply Voltage for the Power Output Stages. A non-inductive 100nF capacitor must be connected between this pin and ground.
5;7	7;9	Input 1; Input 2	TTL Compatible Inputs of the Bridge A.
6;11	8;14	Enable A; Enable B	TTL Compatible Enable Input: the L state disables the bridge A (enable A) and/or the bridge B (enable B).
8	1,10,11,20	GND	Ground.
9	12	VSS	Supply Voltage for the Logic Blocks. A100nF capacitor must be connected between this pin and ground.
10; 12	13;15	Input 3; Input 4	TTL Compatible Inputs of the Bridge B.
13; 14	16;17	Out 3; Out 4	Outputs of the Bridge B. The current that flows through the load connected between these two pins is monitored at pin 15.
–	3;18	N.C.	Not Connected

ELECTRICAL CHARACTERISTICS (V_S = 42V; V_{SS} = 5V, T_j = 25°C; unless otherwise specified)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V_S	Supply Voltage (pin 4)	Operative Condition	V_{IH} +2.5		46	V
V_{SS}	Logic Supply Voltage (pin 9)		4.5	5	7	V
I_S	Quiescent Supply Current (pin 4)	V_{en} = H; I_L = 0 V_i = L V_i = H		13 50	22 70	mA mA
		V_{en} = L V_i = X			4	mA
I_{SS}	Quiescent Current from V_{SS} (pin 9)	V_{en} = H; I_L = 0 V_i = L V_i = H		24 7	36 12	mA mA
		V_{en} = L V_i = X			6	mA
V_{iL}	Input Low Voltage (pins 5, 7, 10, 12)		–0.3		1.5	V
V_{iH}	Input High Voltage (pins 5, 7, 10, 12)		2.3		VSS	V
I_{iL}	Low Voltage Input Current (pins 5, 7, 10, 12)	V_i = L			–10	µA
I_{iH}	High Voltage Input Current (pins 5, 7, 10, 12)	Vi = H ≤ V_{SS} –0.6V		30	100	µA
V_{en} = L	Enable Low Voltage (pins 6, 11)		–0.3		1.5	V
V_{en} = H	Enable High Voltage (pins 6, 11)		2.3		V_{SS}	V
I_{en} = L	Low Voltage Enable Current (pins 6, 11)	V_{en} = L			–10	µA
I_{cn} = H	High Voltage Enable Current (pins 6, 11)	V_{en} = H ≤ V_{3S} –0.6V		30	100	µA
$V_{CEsat(H)}$	Source Saturation Voltage	I_L = 1A I_L = 2A	0.95	1.35 2	1.7 2.7	V V
$V_{CEsat(L)}$	Sink Saturation Voltage	I_L = 1A (5) I_L = 2A (5)	0.85	1.2 1.7	1.6 2.3	V V
V_{CEsat}	Total Drop	I_L = 1A (5) I_L = 2A (5)	1.80		3.2 4.9	V V
V_{sens}	Sensing Voltage (pins 1, 15)		–1 (1)		2	V

ELECTRICAL CHARACTERISTICS (continued)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
T_1 (V_i)	Source Current Turn-off Delay	0.5 V_i to 0.9 I_L (2); (4)		1.5		µs
T_2 (V_i)	Source Current Fall Time	0.9 I_L to 0.1 I_L (2); (4)		0.2		µs
T_3 (V_i)	Source Current Turn-on Delay	0.5 V_i to 0.1 I_L (2); (4)		2		µs
T_4 (V_i)	Source Current Rise Time	0.1 I_L to 0.9 I_L (2); (4)		0.7		µs
T_5 (V_i)	Sink Current Turn-off Delay	0.5 V_i to 0.9 I_L (3); (4)		0.7		µs
T_6 (V_i)	Sink Current Fall Time	0.9 I_L to 0.1 I_L (3); (4)		0.25		µs
T_7 (V_i)	Sink Current Turn-on Delay	0.5 V_i to 0.9 I_L (3); (4)		1.6		µs
T_8 (V_i)	Sink Current Rise Time	0.1 I_L to 0.9 I_L (3); (4)		0.2		µs
fc (V_i)	Commutation Frequency	I_L = 2A		25	40	KHz
T_1 (V_{en})	Source Current Turn-off Delay	0.5 V_{en} to 0.9 I_L (2); (4)		3		µs
T_2 (V_{en})	Source Current Fall Time	0.9 I_L to 0.1 I_L (2); (4)		1		µs
T_3 (V_{en})	Source Current Turn-on Delay	0.5 V_{en} to 0.1 I_L (2); (4)		0.3		µs
T_4 (V_{en})	Source Current Rise Time	0.1 I_L to 0.9 I_L (2); (4)		0.4		µs
T_5 (V_{en})	Sink Current Turn-off Delay	0.5 V_{en} to 0.9 I_L (3); (4)		2.2		µs
T_6 (V_{en})	Sink Current Fall Time	0.9 I_L to 0.1 I_L (3); (4)		0.35		µs
T_7 (V_{en})	Sink Current Turn-on Delay	0.5 V_{en} to 0.9 I_L (3); (4)		0.25		µs
T_8 (V_{en})	Sink Current Rise Time	0.1 I_L to 0.9 I_L (3); (4)		0.1		µs

1) 1)Sensing voltage can be –1 V for t ≤ 50 µsec; in steady state V_{sens} min ≥ –0.5 V.
2) See fig. 2.
3) See fig. 4.
4) The load must be a pure resistor.

Annexe B

Datasheet Camera OV2460

Advanced Information
Preliminary Datasheet

OV2640 Color CMOS UXGA (2.0 MegaPixel) CAMERACHIP™ with OmniPixel2™ Technology

General Description

The OV2640 CAMERACHIP™ is a low voltage CMOS image sensor that provides the full functionality of a single-chip UXGA (1632x1232) camera and image processor in a small footprint package. The OV2640 provides full-frame, sub-sampled, scaled or windowed 8-bit/10-bit images in a wide range of formats, controlled through the Serial Camera Control Bus (SCCB) interface.

This product has an image array capable of operating at up to 15 frames per second (fps) in UXGA resolution with complete user control over image quality, formatting and output data transfer. All required image processing functions, including exposure control, gamma, white balance, color saturation, hue control, white pixel canceling, noise canceling, and more, are also programmable through the SCCB interface. The OV2640 also includes a compression engine for increased processing power. In addition, OmniVision CAMERACHIPS use proprietary sensor technology to improve image quality by reducing or eliminating common lighting/electrical sources of image contamination, such as fixed pattern noise, smearing, etc., to produce a clean, fully stable color image.

Note: The OV2640 uses a lead-free package.

Features

- High sensitivity for low-light operation
- Low operating voltage for embedded portable apps
- Standard SCCB interface
- Output support for Raw RGB, RGB (RGB565/555), GRB422, YUV (422/420) and YCbCr (4:2:2) formats
- Supports image sizes: UXGA, SXGA, SVGA, and any size scaling down from SXGA to 40x30
- VarioPixel® method for sub-sampling
- Automatic image control functions including Automatic Exposure Control (AEC), Automatic Gain Control (AGC), Automatic White Balance (AWB), Automatic Band Filter (ABF), and Automatic Black-Level Calibration (ABLC)
- Image quality controls including color saturation, gamma, sharpness (edge enhancement), lens correction, white pixel canceling, noise canceling, and 50/60 Hz luminance detection
- Line optical black level output capability
- Video or snapshot operation
- Zooming, panning, and windowing functions
- Internal/external frame synchronization
- Variable frame rate control
- Supports LED and flash strobe mode
- Supports scaling
- Supports compression
- Embedded microcontroller

Ordering Information

Product	Package
OV02640-VL9A (Color, lead-free)	38-pin CSP2

Applications

- Cellular and Camera Phones
- Toys
- PC Multimedia
- Digital Still Cameras

Key Specifications

Array Size	**UXGA**	1600 x 1200
Power Supply	**Core**	1.2VDC ± 5%
	Analog	2.5 ~ 3.0VDC
	I/O	1.7V to 3.3V
Power Requirements	**Active**	125 mW (for 15 fps, UXGA YUV mode)
		140 mW (for 15 fps, UXGA compressed mode)
	Standby	600 µA
Temperature Range	**Operation**	-30°C to 70°C
	Stable Image	0°C to 50°C
Output Formats (8-bit)		• YUV(422/420)/YCbCr422 • RGB565/555 • 8-bit compressed data • 8-/10-bit Raw RGB data
Lens Size		1/4"
Chief Ray Angle		25° non-linear
Maximum Image Transfer Rate	**UXGA/SXGA**	15 fps
	SVGA	30 fps
	CIF	60 fps
Sensitivity		0.6 V/Lux-sec
S/N Ratio		40 dB
Dynamic Range		50 dB
Scan Mode		Progressive
Maximum Exposure Interval		1247 x t_{ROW}
Gamma Correction		Programmable
Pixel Size		2.2 µm x 2.2 µm
Dark Current		15 mV/s at 60°C
Well Capacity		12 Ke
Fixed Pattern Noise		<1% of $V_{PEAK-TO-PEAK}$
Image Area		3590 µm x 2684 µm
Package Dimensions		5725 µm x 6285 µm

Figure 1 OV2640 Pin Diagram (Top View)