



وزارة التعليم العالي والبحث العلمي
Ministry of Higher Education and Scientific Research
جامعة عبد الحميد بن باديس مستغانم

University Abdelhamid Ibn Badis Mostaganem

كلية العلوم والتكنولوجيا - قسم الهندسة الكهربائية

Faculty of Science and Technology - Department of Electrical Engineering



Mémoire de Master Spécialité : Systèmes des Télécommunications

Thème

Conception et simulation d'un système de géolocalisation et de détection
d'accident de voiture utilisant les modules GSM et GPS

Présenté par :

- BENDAHDJANE khadidja
- BOUZIANE Fatima Zohra

Soutenu le 21/06/2026

Membres du jury :

- | | |
|----------------------------|-----------|
| - Mme BENOUDNINE Hadjira | Président |
| - Mr OULD ALI Abdelaziz | Examineur |
| - Mr AIMOUCH Djamel Eddine | Encadreur |

Année universitaire : 2025/2026

Dédicace

« À mes chers parents, qui m'ont toujours soutenu et guidé avec amour et sagesse. A mes adorables sœurs, Radhia, Yousra, pour leur affection et leurs encouragements constants. A mes frères, Mohamed Redha et Miloud, pour leur fraternité. A ma belle sœur Rania pour son encouragement.

À mes petites anges, Ines, Lyna, Nayla et Lydia, qui apportent tant de joie et de lumière dans ma vie.

À ma chère grand-mère, dont je porte fièrement le prénom. Partie trop tôt, sa présence vit toujours dans mon cœur. Elle qui aimait tant nous voir réussir, j'espère qu'elle est fière de moi aujourd'hui.

Que Dieu l'accueille en Son vaste paradis. »

-BENDAHRMANE KHADIDJA-

« À mes parents, pour leur soutien éternel et leurs sacrifices. Ce succès est avant tout le vôtre. À mes sœurs, Chahrazed et Narimene, ainsi qu'à mon frère Abdelkader, pour leur présence et leur affection qui me sont indispensables.

À ma petite nièce, Ayline, la petite lumière de notre famille. Que ce travail soit pour toi un exemple de persévérance et que ton avenir soit aussi radieux que ton sourire. »

À la mémoire de ma chère grand-mère, qui nous a quittés physiquement mais qui reste présente dans mon cœur. Elle qui aimait tant nous voir réussir et gravir les échelons, j'espère que de là où elle est, elle est fière de moi aujourd'hui. Que Dieu l'accueille en Son vaste paradis.

-BOUZIANE FATIMA ZOHRA-

Remerciements

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

Nous tenons à exprimer premièrement notre profonde gratitude à Dieu, qui nous a donné la santé et la volonté pour le parachèvement de ce travail. Et nous voudrions exprimer toute notre gratitude à : Notre encadreur : Monsieur AIMOUCHE Djamel Eddine, pour la confiance qu'il nous a témoignée en acceptant de diriger ce travail et pour nous avoir mis à notre disposition ses compétences et ses conseils pour une meilleure maîtrise du sujet

Notre vif remerciement aux membres du jury pour l'honneur qu'ils nous font en acceptant de juger ce travail.

Notre remerciement aussi à nos professeurs, nos collègues, nos amies et toutes les personnes qui nous ont aidé durant nos études universitaires.

Résumé

La sécurité routière demeure une préoccupation majeure dans le domaine des transports publics en raison de l'impact significatif des accidents sur la performance globale du système de transport. Pour atténuer ces risques, notre projet propose un système de gestion des risques de sécurité routière basé sur l'intelligence artificielle. Notre approche consiste à développer des modèles d'intelligence artificielle capables de prédire les risques potentiels liés au comportement des conducteurs. Nous nous concentrons sur le style de conduite, la distraction et la somnolence. Nous avons obtenu des résultats prometteurs en termes de précision de prédiction. Nos modèles peuvent identifier avec précision les comportements à risque. L'efficacité de ce système réside dans l'utilisation de deux interfaces utilisateur distinctes, l'une dédiée aux superviseurs et l'autre aux conducteurs. Les superviseurs disposent ainsi d'un outil de supervision en temps réel des performances des conducteurs, tandis que ces derniers bénéficient d'alertes personnalisées visant à corriger leur comportement au volant. Cette approche basée sur le GPS et GSM contribue à réduire les risques d'accidents et à améliorer la sécurité sur nos routes.

ملخص

لا تزال السلامة المرورية تشكل هاجساً رئيسياً في قطاع النقل العام نظراً لتأثير الحوادث الكبير على الأداء العام لنظام النقل. وللمحد من هذه المخاطر، يقترح مشروعنا نظاماً لإدارة مخاطر السلامة المرورية قائماً على الذكاء الاصطناعي. يتضمن نهجنا تطوير نماذج ذكاء اصطناعي قادرة على التنبؤ بالمخاطر المحتملة المرتبطة بسلوك السائق، مع التركيز على أسلوب القيادة، وتشتت الانتباه، والنعاس. وقد حققنا نتائج واعدة من حيث دقة التنبؤ، حيث تستطيع نماذجنا تحديد السلوكيات الخطرة بدقة. تكمن فعالية النظام في استخدام واجهتي مستخدم منفصلتين: إحداهما للمشرفين والأخرى للسائقين. يمتلك المشرفون أداة لمراقبة أداء السائقين في الوقت الفعلي، بينما يتلقى السائقون تنبيهات مخصصة مصممة لتصحيح سلوكهم أثناء القيادة. يُسهم هذا النهج القائم على الذكاء الاصطناعي في تقليل مخاطر الحوادث وتحسين السلامة على طرقنا.

Abstract

Road safety remains a major concern in the public transport sector due to the significant impact of accidents on the overall performance of the transport system. To mitigate these risks, our project proposes a road safety risk management system based on artificial intelligence. Our approach involves developing AI models capable of predicting potential risks related to driver behavior. We focus on driving style, distraction, and drowsiness. We have obtained promising results in terms of predictive accuracy. Our models can accurately identify risky behaviors. The system's effectiveness lies in the use of two distinct user interfaces: one for supervisors and the other for drivers. Supervisors have a tool for real-time monitoring of driver performance, while drivers receive personalized alerts designed to correct their driving behavior. This AI-based approach helps reduce the risk of accidents and improve safety on our roads.

Sommaire :

Dédicace	1
Remerciements	2
Résumé	3
Sommaire	4
Liste des figures	7
Liste des tableaux	9
Glossaire	10
Introduction	17
CHAPITRE I Technologies GSM et GPS embarquées	20
1.1 Introduction aux systèmes embarqués dans la sécurité routière	21
1.1.1 Contexte Mondial et Enjeux de la Sécurité Routière	21
1.1.2 L'Évolution vers les Véhicules Intelligents (ITS)	21
1.1.3 Définition et Rôle des Systèmes Embarqués :	23
1.2 Technologies de Géolocalisation : Principes et Applications du GPS	23
1.2.1 Architecture du Système : Les Trois Segments	23
1.2.2 Le Calcul de la Position : La Trilatération	24
1.2.3 Sources d'Erreurs et Précision :	25
1.2.4 L'importance de la Synchronisation Temporelle et les Horloges Atomiques	26
1.2.5 Le Protocole NMEA 0183 : Langage du Module GPS	27
1.3 Technologies de Communication Mobile : Réseaux Télécoms	28
1.3.1 Comparaison Détaillée des Générations Mobiles (2G à 5G)	28
1.3.2 Protocoles de Transmission : SMS vs GPRS/Data	29
1.3.3 Architecture du Réseau GSM	30
1.4 Intellection Technique et Synergie du Système Embarqué	31
1.4.1 Architecture Matérielle Avancée (Hardware Intégration)	31
1.4.2 Logique de Traitement et "Parsing" des Données GPS	33
1.4.3 Dialogue par Commandes AT : Le protocole GSM	34
1.4.4 Contraintes Électromagnétiques et Environnementales	35
1.5 Études Approfondies des Solutions de Sécurité Existantes	36
1.5.1 Le Système eCall : Une Standardisation Institutionnelle	36
1.5.2 Les Systèmes Constructeurs Propriétaires (OnStar, BMW Assist)	37
1.5.3 Analyse Critique des Capteurs de Smartphones	37
1.6 Analyse Justificative du Choix Technologique : Pourquoi le couple GSM/GPS ?	38
1.6.1 Accessibilité Économique et Démocratisation (Le facteur Coût)	38

1.6.2 La Robustesse de la Couverture Mondiale (Le facteur Réseau)	39
1.6.3 Simplicité de Mise en Œuvre et Standardisation (Le facteur Ingénierie)	39
1.6.4 Fiabilité des Capteurs Dédiés face aux Smartphones	40
CHAPITRE II Matériel et environnement de développement	43
2.1 Présentation de la plateforme Arduino.....	44
2.1.1 Carte Arduino.....	44
2.1.2 L'alimentation.....	44
2.1.3 Les broche d'Arduino Uno	44
2.1.4 Domaines d'Application Arduino Uno	45
2.1.5 Avantages	46
2.2 Caractéristique techniques de la carte Arduino et du Microcontrôleur	46
2.2.1 La carte ARDUINO UNO	46
2.2.2 Caractéristiques techniques d'Arduino uno	49
2.2.3 Qu'est-ce qu'un microcontrôleur ?	51
2.3 L'environnement de Développement (IDE) :	53
2.3.1 Arduino IDE	53
2.3.2 Structure d'un programme	54
2.3.3 Le code minimal	55
2.3.4 La syntaxe du langage	55
2.4 Présentation et fonctionnement des modules GPS et GSM.....	60
2.4.1 GPS : Global Positioning System.....	60
2.4.1.1 Composants clés d'un module GPS	60
2.4.1.2 Comparaison du module GPS avec d'autres modules sans fil :	63
2.4.2 Présentation du module GSM Sim900	64
2.5 Caractéristique techniques de la carte ESP32	65
2.5.1 Description ESP32.....	65
2.5.2 Les caractéristiques techniques :	66
2.5.3 La liste des composants de la carte ESP32	67
2.5.4 Brochage ESP32	68
2.6 Présentation et fonctionnement du capteur MPU6050	70
2.6.1 MPU6050.....	70
2.6.2 Caractéristique	70
2.6.3 Applications	72
2.7 Description et fonctionnement du module sim800L.....	72
2.7.1 Module sim800L.....	72
2.7.2 Fonctionnalités :	73
2.7.3 LED clignotant :	73

2.7.4 Détails de la broche :	73
2.8 Étude Approfondie des Protocoles de Communication (Bus I2C, UART, SPI)	74
2.8.1 Le protocole UART (Universal Asynchronous Receiver-Transmitter)	75
2.8.2 Le Bus I2C (Inter-Integrated Circuit)	75
2.8.3 L'Interface SPI (Serial Peripheral Interface)	76
2.6 Modèle d'intelligence artificielle IA	76
2.6.1 Introduction à l'Intelligence Artificielle	76
2.6.3 Déploiement : L'Approche "TinyML"	76
CHAPITRE III Simulation et réalisation du système	80
3.1 Présentation de l'Environnement de Simulation Proteus ISIS	81
3.1.1 Justification du choix de l'outil	81
3.1.2 Configuration de l'environnement	81
Branchement du MPU6050 (Accéléromètre, Gyroscope)	84
Programme d'entraînement IA et de création de fichier PKL (joblib)	85
Programmation de la carte Arduino	87
Programme de Détection d'Accident et d'Alerte SMS/Appel (ESP32 - MPU6050 - GSM)	91
Synthèse :	93
Conclusion :	94
Références	95
Annexe	100

Liste des figures

Figure 1.1 Architecture globale du système d'assistance et de communication d'un véhicule intelligent.....	21
Figure 1.2 Processus de transmission de l'alerte eCall via le réseau mobile vers les centres de secours.....	22
Figure 1.3 Architecture de communication entre le véhicule connecté, l'infrastructure réseau et les services d'urgence.....	22
Figure 1.4 Illustration du principe de positionnement par satellites et les trois segments du système GPS.....	24
Figure 1.5 Architecture détaillée du réseau GSM présentant les interactions entre le sous-système radio (BSS) et le sous-système réseau (NSS).....	32
Figure 1.6 Conception du système de contrôle matériel sur cible Zynq-7000.....	32
Figure 1.7 Schema de câblage du prototype incluant l'Arduino uno le module GSM SIM800L et le récepteur GPS Neo-6M pour la transmission des coordonnées de l'accident.....	35
Figure 1.8 Référentiels tridimensionnels du capteur et de véhicule pour la synchronisation des données inertielles.....	37
Figure 1.9 Justification technico-économique de la solution : (1.10) étendue de la couverture réseau, (1.11) étude comparative des coûts entre systèmes propriétaires et solutions DIY, (1.12) mécanisme de robustesse de la transmission par SMS	40
Figure 2.1 : image qui représente quelque domaine d'utilisation Arduino Uno.....	46
Figure 2.2 : Schéma descriptif des composants de la carte Arduino Uno.....	47
Figure 2.3 : Architecture interne d'un microcontrôleur : Modèle de Von Neumann.....	51
Figure 2.4 : Schéma fonctionnel des entrées et sorties d'un système à microcontrôleur.....	52
Figure 2.5 : Cycle de développement et environnement de travail Arduino.....	54
Figure 2.6 : Déroulement d'un programme.....	55
Figure 2.7 : Description des zones fonctionnelles du logiciel de programmation.....	57
Figure 2.8 : L'interface du Moniteur Série (Serial Monitor) et ses fonctionnalités.....	58
Figure 2.9 : Mémento de la syntaxe et des fonctions du langage Arduino.....	59
Figure 2.10 : Présentation du module GPS (récepteur et antenne).....	60
Figure 2.11 : Principe de fonctionnement et communication du réseau GPS.....	61
Figure 2.12 : Module GSM GPRS SIM900.....	64
Figure 2.13 : Schéma descriptif des composants de l'ESP32.....	68

Figure 2.14 :	Diagramme de brochage (Pinout) de la carte ESP32 NodeMCU.....	68
Figure 2.15 :	une image qui représente MPU6050.....	70
Figure 2.16 :	Orientation des axes de sensibilité et polarité de rotation.....	71
Figure 2.17 :	Module GSM SIM800L.....	72
Figure 2.18 :	Brochage du module GSM SIM800L.....	74
Figure 3.1:	Schema global du system sous Proteus ISIS.....	82
Figure 3.2 :	Montage du système Arduino Uno–MPU6050	83
Figure 3.3 :	Résultat sur l’application web Streamlit	87
Figure 3.4 :	Montage expérimental ESP32–MPU6050 avec communication Bluetooth.....	88
Figure 3.5 :	Résultat de l’affichage après une détection d’accident.....	89
Figure 3.6 :	Affichage des résultats sur l’écran du téléphone via Bluetooth	89
Figure 3.7 :	Affichage des résultats sur l’écran du téléphone et l’écran LCD via le module SIM800L	90

Liste des tableaux :

Tableau 1.1 : Un tableau pour détailler chaque champ.....	28
Tableau 1.2 : Comparaison Détaillée des Générations Mobiles (2G à 5G).....	29
Tableau 2.1 : Tableau des correspondances entre les broches E/S et les fonctions alternatives du microcontrôleur.....	45
Tableau 2.2 : Tableau comparatif des technologies de communication sans fil pour systèmes embarqués	63
Tableau 2.3 : les composants de la carte ESP32 avec leurs descriptions	67
Tableau 2.4 : Tableau des broches de l'ESP32 et leurs fonctions	69

GLOSSAIRE :

BSC : *Base Station Controller* (Contrôleur de Stations de Base). Composant du réseau GSM qui gère les ressources radio et contrôle plusieurs antennes relais (BTS).

BSS : *Base Station Subsystem* (Sous-Système Radio). Partie du réseau cellulaire GSM qui assure la liaison radio directe avec le module embarqué.

BTS : *Base Transceiver Station* (Station de Base Émettrice-Réceptrice). L'antenne relais du réseau mobile qui reçoit et transmet les signaux radio du module SIM800L.

Buffer : Mémoire tampon. Zone de la mémoire RAM utilisée pour stocker temporairement des données (comme les phrases GPS) afin d'éviter leur perte pendant que le processeur exécute une autre tâche.

DD (Degrés Décimaux) : Format numérique d'affichage des coordonnées géographiques (ex: 48.1166), indispensable pour afficher directement une position sur Google Maps.

eCall : Système automatique d'appel d'urgence standardisé et obligatoire sur les véhicules neufs en Europe, combinant GPS et liaison cellulaire pour alerte les secours publics.

EMI : *Electromagnetic Interference* (Interférences Électromagnétiques). Perturbations causées par des ondes radioémises (comme celles du GSM) pouvant perturber les composants électroniques voisins (comme l'antenne GPS).

Fail-Safe : Concept d'ingénierie qui consiste à concevoir un système pour qu'il se place automatiquement dans un mode sécurisé et fonctionnel en cas de panne ou de dégradation du réseau.

Firmware : Micrologiciel interne programmé directement dans la mémoire du microcontrôleur pour gérer le comportement matériel du prototype.

FPGA: *Field-Programmable Gate Array*. Circuit logique que l'on peut programmer et configurer électroniquement après sa fabrication pour exécuter des calculs matériels lourds.

G (Force G) : Unité de mesure d'accélération correspondant à la gravité terrestre (environ 9.81 m/s^2). Le seuil de 17.0 m/s^2 dans le code équivaut à environ 1.73 G.

GPRMC: *Recommended Minimum Specific GNSS Data*. Type de trame textuelle standard du protocole NMEA, sélectionnée dans notre projet car elle regroupe le minimum d'informations nécessaires (latitude, longitude, heure, date, vitesse).

GSM: *Global System for Mobile Communications*. Standard mondial de téléphonie mobile de deuxième génération (2G), utilisé pour l'acheminement des SMS et des appels vocaux.

HLR : *Home Location Register (Registre des Abonnés Nominiaux)*. Base de données centrale du réseau GSM qui contient le profil, l'identité et les droits d'accès de l'abonné (lié à la carte SIM).

ISR : *Interrupt Service Routine (Routine de Service d'Interruption)*. Fonction logicielle prioritaire exécutée instantanément par le processeur lorsqu'un événement matériel critique survient (comme la détection d'un choc par l'accéléromètre), interrompant le programme principal.

ITS : *Intelligent Transport Systems (Systèmes de Transport Intelligents)*. Application des technologies de l'information et de la communication au domaine des transports routiers pour améliorer la sécurité et la gestion du trafic.

LTE : *Long Term Evolution*. Standard de réseau mobile de quatrième génération (4G) offrant de hauts débits de données.

MS : *Mobile Station (Station Mobile)*. Ensemble indissociable composé du terminal électronique (microcontrôleur + module GSM) et de la carte SIM de l'abonné.

MSC : *Mobile Switching Center* (Centre de Commutation Mobile). Le commutateur principal au cœur du réseau GSM qui gère le routage des appels et des SMS vers leur destination.

MSD : *Minimum Set of Data* (Ensemble Minimum de Données). Paquet de données standardisé de 140 octets envoyé par le système eCall contenant les informations vitales du crash.

NMEA 0183 : Norme d'échange de données maritimes et terrestres définissant le format des phrases textuelles (trames) envoyées par un récepteur GPS.

NSS: *Network and Switching Subsystem* (Sous-Système Réseau). Le cœur de réseau de l'opérateur télécom en charge de la commutation et de la gestion des abonnés.

Parsing : Analyse syntaxique logicielle. Action de découper et de filtrer une chaîne de caractères brute (comme une trame GPS NMEA) pour en extraire uniquement les valeurs utiles.

PCB : *Printed Circuit Board* (Circuit Imprimé). Support rigide en fibre de verre reliant mécaniquement et électriquement les composants électroniques du prototype.

PDOP : *Position Dilution of Precision* (Atténuation de la Précision de la Position). Indicateur numérique qualifiant la disposition géométrique des satellites GPS dans le ciel. Plus sa valeur est faible (proche de 1), meilleure est la précision du calcul.

PDU (Mode PDU) : *Protocol Data Unit*. Mode binaire industriel d'encodage et de transmission des SMS, plus complexe à programmer que le mode texte standard.

PSAP : *Public Safety Answering Point* (Centre de Réception des Appels d'Urgence). Centre de traitement des secours publics (ex: SAMU, Protection Civile) qui reçoit les alertes eCall ou SMS.

RAM : *Random Access Memory* (Mémoire Vive). Mémoire informatique volatile ultra-rapide utilisée par le microcontrôleur pour stocker ses variables et ses tampons de données pendant son fonctionnement.

SCL : *Serial Clock*. Ligne d'horloge dédiée à la synchronisation temporelle des composants sur le bus de communication I2C.

SDA : *Serial Data*. Ligne de données bidirectionnelle utilisée pour le transit des informations sur le bus I2C.

SIM : *Subscriber Identity Module*. Carte à puce amovible contenant l'identité de l'abonné et les clés de chiffrement pour accéder de façon sécurisée au réseau d'un opérateur mobile.

UTC : *Coordinated Universal Time (Temps Universel Coordonné)*. Échelle de temps de référence internationale diffusée de façon ultra-précise par les satellites GPS.

V2X : *Vehicle-to-Everything (Communication du véhicule avec tout son environnement)*. Concept regroupant les communications véhicule-à-véhicule (V2V), véhicule-à-infrastructure (V2I) et véhicule-au-réseau (V2N).

VIN : *Vehicle Identification Number*. Numéro de série unique à 17 caractères gravé sur le châssis d'un véhicule, transmis lors d'une alerte d'urgence pour identifier le modèle.

VLR : *Visitor Location Register (Registre des Abonnés Visiteurs)*. Base de données locale du réseau GSM qui enregistre temporairement l'identité des mobiles présents dans une zone géographique donnée pour permettre le routage rapide des SMS et appels.

ADC (*Analog-to-Digital Converter*) : Convertisseur analogique-numérique qui transforme un signal analogique en données numériques.

ASCII (*American Standard Code for Information Interchange*) : Norme de codage des caractères utilisée pour représenter les lettres, chiffres et symboles dans les systèmes informatiques. Le **Code 26 (Ctrl+Z)** est souvent utilisé pour indiquer la fin d'un message SMS avec les modules GSM.

AT Command (*Attention Command*) : Ensemble de commandes textuelles utilisées pour contrôler les modules GSM/GPRS, tels que le SIM800L.

Baud (Baud Rate) : Vitesse de transmission des données dans une communication série, exprimée en bauds (bits par seconde dans la plupart des cas).

CAN (Controller Area Network) : Bus de communication robuste utilisé principalement dans les systèmes automobiles et industriels.

DAC (Digital-to-Analog Converter) : Convertisseur numérique-analogique qui transforme des données numériques en signal analogique.

DMP (Digital Motion Processor) : Processeur intégré au MPU6050 permettant le traitement des données de mouvement et d'orientation.

ESP32 : Microcontrôleur développé par Espressif Systems intégrant le Wi-Fi, le Bluetooth et de nombreuses interfaces de communication.

Ethernet : Technologie de communication filaire utilisée pour les réseaux informatiques locaux et l'accès à Internet.

FlexRay : Protocole de communication haute vitesse utilisé dans les systèmes automobiles avancés nécessitant une grande fiabilité.

GND (Ground) : Masse électrique servant de référence commune dans un circuit électronique.

GPRS (General Packet Radio Service) : Technologie permettant la transmission de données sur les réseaux GSM.

GPS (Global Positioning System) : Système de géolocalisation par satellites permettant de déterminer une position géographique.

GPS NEO-8M : Module GPS de haute précision permettant de recevoir les coordonnées géographiques, la vitesse et l'heure à partir des satellites GPS.

GSM (Global System for Mobile Communications) : Norme mondiale de téléphonie mobile numérique.

I2C (Inter-Integrated Circuit) : Protocole de communication série utilisant deux lignes (SDA et SCL) pour relier plusieurs périphériques à un microcontrôleur.

Kbps (Kilobits per second) : Unité de mesure du débit de transmission des données, exprimée en milliers de bits par seconde.

LCD (Liquid Crystal Display) : Écran à cristaux liquides utilisé pour afficher du texte, des chiffres ou des informations graphiques.

MISO (Master In Slave Out) : Ligne de données SPI permettant à un périphérique esclave d'envoyer des données au maître.

MMS (Multimedia Messaging Service) : Service permettant l'envoi de messages contenant des images, du son ou des vidéos.

MOSI (Master Out Slave In) : Ligne de données SPI permettant au maître d'envoyer des données à l'esclave.

MPU6050 (Motion Processing Unit 6050) : Capteur combinant un accéléromètre à 3 axes et un gyroscope à 3 axes pour mesurer les mouvements et l'inclinaison.

RST (Reset) : Broche ou signal utilisé pour redémarrer ou réinitialiser un circuit électronique.

RXD (Receive Data) : Broche utilisée pour recevoir des données dans une communication série.

SCK (Serial Clock) : Ligne d'horloge utilisée dans le protocole SPI.

SCL (Serial Clock Line) : Ligne d'horloge du protocole I2C.

SDA (Serial Data Line) : Ligne de données du protocole I2C.

SIM800L : Module GSM/GPRS permettant l'envoi et la réception de SMS, les appels téléphoniques et la transmission de données mobiles.

SMS (Short Message Service) : Service de messagerie texte des réseaux mobiles.

SSP (Synchronous Serial Port) : Interface de communication série synchrone prenant en charge des protocoles comme SPI.

TXD (Transmit Data) : Broche utilisée pour transmettre des données dans une communication série.

UART (Universal Asynchronous Receiver Transmitter) : Interface de communication série asynchrone utilisée entre microcontrôleurs, capteurs et modules de communication.

USB (Universal Serial Bus) : Standard de connexion permettant le transfert de données et l'alimentation électrique des périphériques.

VCC (Voltage at Common Collector) : Broche d'alimentation positive d'un circuit électronique.

Introduction

Aujourd'hui, les véhicules jouent un rôle essentiel dans notre quotidien. Ils nous permettent d'aller travailler, de communiquer avec nos proches et de transporter des marchandises. Cependant, ils peuvent aussi être source de catastrophes, les accidents automobiles représentent l'une des principales causes de décès et de blessures dans le monde. Chaque année, des millions de personnes sont impliquées dans des collisions routières, entraînant des conséquences tragiques pour les victimes et leur entourage. Ces accidents peuvent survenir pour une multitude de raisons, allant de l'erreur humaine à des facteurs environnementaux ou mécaniques.

Les routes, malgré les progrès dans la conception des véhicules et les efforts de sensibilisation, demeurent des environnements où le risque d'accident persiste. Les facteurs contributifs incluent la vitesse excessive, la conduite sous l'influence de l'alcool ou de drogues, la distraction au volant, le non-respect des règles de circulation, les conditions météorologiques défavorables, ainsi que les défauts mécaniques des véhicules.

Outre les pertes en vies humaines, les accidents automobiles ont également un impact économique considérable, avec des coûts élevés en termes de soins médicaux, de réparations de véhicules, et de pertes de productivité. Ils affectent également les systèmes de santé publique, les services d'urgence, et les infrastructures routières. [1]

C'est pourquoi nous utilisons un système d'alerte afin d'améliorer les systèmes d'urgence et de gestion des accidents et de relever ce défi. L'équipe de secours est informée des coordonnées de l'accident (longitude et latitude) par le système dès sa détection. Le système qualifie un événement d'accident si l'accéléromètre enregistre une accélération inhabituelle et si le capteur de vibrations détecte un déplacement significatif. Une fois l'incident signalé, les modules GSM et GPS déterminent la position exacte et envoient un message contenant des informations sur l'accident et sa localisation précise. Cette étude contribue à proposer une solution efficace à la situation critique qu'elle engendre. L'approche actuelle privilégie la sécurité des passagers plutôt que l'assistance d'urgence en cas de collision. Le système que nous avons mis en place vise à identifier automatiquement les accidents et à notifier l'hôpital ou les services médicaux les plus proches de leur position précise. Ce dispositif transmet des informations essentielles à l'équipe de secours quelques secondes après l'accident. Grâce à cette technologie, des vies peuvent être sauvées en détectant rapidement les accidents et en alertant les secours. Le message d'alerte indique la position, l'heure et l'angle de l'incident. Un capteur active le dispositif dès qu'un accident est détecté. Le microcontrôleur reçoit le signal du capteur. L'alerte est émise par le

microcontrôleur. Les systèmes technologiques de suivi sont aujourd'hui largement répandus et utilisés dans presque tous les secteurs. Grâce à leurs performances supérieures, ces dispositifs sont conçus pour effectuer des manœuvres complexes et adaptées à chaque environnement. Leur capacité à accomplir des missions à un coût raisonnable leur confère une importance croissante dans les domaines civil et de la sécurité. Ils peuvent également servir au suivi des accidents de la route. [2-3]

CHAPITRE I

Technologies GSM et GPS embarquées

1.1 Introduction aux systèmes embarqués dans la sécurité routière :

1.1.1 Contexte Mondial et Enjeux de la Sécurité Routière

L'amélioration de la sécurité routière est une priorité. A l'échelle mondiale, les accidents de la route causent 1,3 million de décès et près de 50 millions de blessés par an. Ils constituent l'une des principales causes de mortalité en particulier chez les jeunes âgés de 15 à 29 ans.

C'est un véritable fléau qui pourrait d'ici 2030 devenir la cinquième cause de mortalité mondiale avec 2,4 millions de décès par an.

Attentifs à cette calamité, les pays développés ont réussi à coup de programmes préventifs et de mesures restrictives à améliorer leur bilan sécuritaire. Avec 90% des décès, les pays à revenu faible continuent, par contre, à occuper les premières places en termes de taux de mortalité. [4]

1.1.2 L'Évolution vers les Véhicules Intelligents (ITS)

Confrontés à cette situation, le secteur automobile et la recherche scientifique se sont orientés vers les Systèmes de Transport Intelligents (ITS). Le but est de convertir le véhicule passif en un agent dynamique capable de :

Observer son propre état mécanique. Examiner l'environnement proche.

Établir une communication en direct avec l'environnement externe (V2X - Vehicle to Everything).

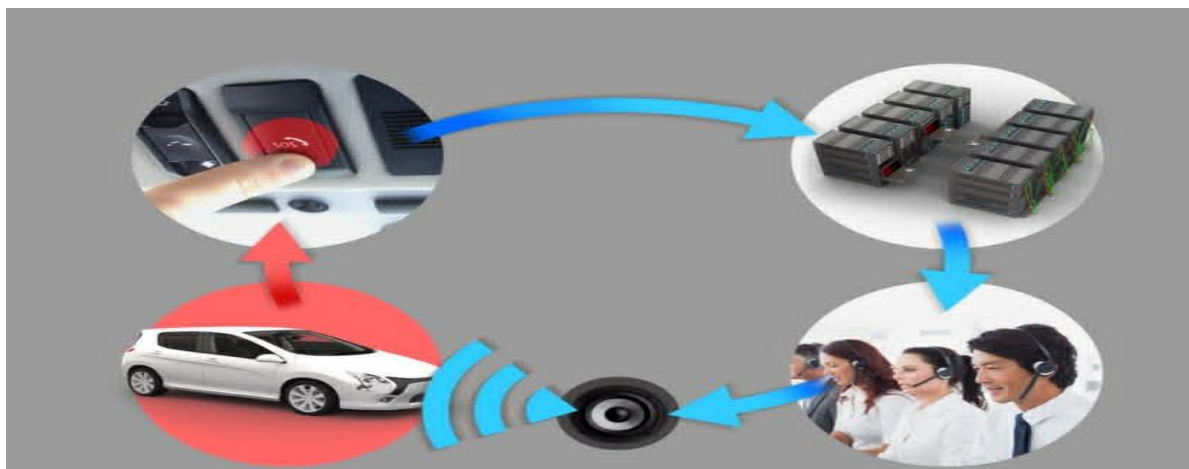


Figure 1.1 : Architecture globale du système d'assistance et de communication d'un véhicule intelligent [5]

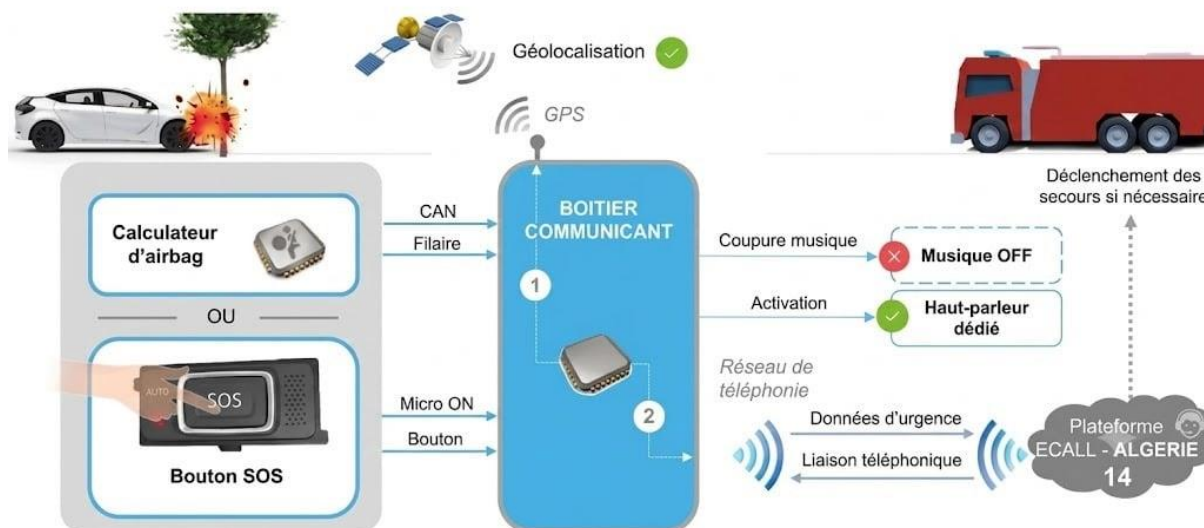


Figure 1.2 : Processus de transmission de l'alerte eCall via le réseau mobile vers les centres de secours [6]

Comment fonctionne l'appel d'urgence ?

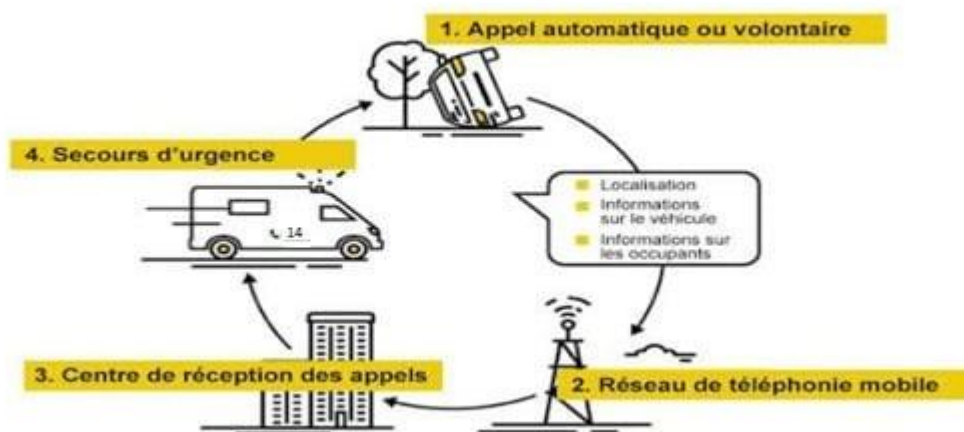


Figure 1.3 : Architecture de communication entre le véhicule connecté, l'infrastructure réseau et les services d'urgence [7]

1.1.3 Définition et Rôle des Systèmes Embarqués :

Un système embarqué est un système informatique et électronique autonome qui exécute une tâche précise au sein de l'appareil auquel il est intégré. La plupart du temps, cette tâche est réalisée en temps réel. Le terme de « système embarqué » recouvre à la fois le matériel et le logiciel utilisés.

Le logiciel est exécuté sur un microcontrôleur, un microprocesseur ou, plus rarement, sur un FPGA. Le choix de la plateforme dépend de la puissance de calcul nécessaire mais aussi des interfaces requises : bus de communication, écran, mesures analogiques... L'espace mémoire du système embarqué est généralement limité, de même que sa taille. Aussi, le système embarqué doit pouvoir effectuer sa tâche avec précision, dans des délais imposés, avec le meilleur ratio encombrement/mémoire/consommation d'énergie/coût. [8]

1.2 Technologies de Géolocalisation : Principes et Applications du GPS

Le GPS (Global Positioning System) est un système de positionnement par satellite permettant de déterminer la localisation GPS d'un appareil en captant des signaux provenant de plusieurs satellites. Chaque dispositif de suivi est équipé d'un récepteur GPS qui analyse ces signaux pour calculer son positionnement **GPS** en fonction des temps de transmission.

Pour garantir la précision des informations, toute la donnée est mise à jour et la position est recalculée continuellement grâce à des technologies de transmission des données qui permettent un système de localisation performant. [9]

1.2.1 Architecture du Système : Les Trois Segments

Le fonctionnement du GPS repose sur la coopération de trois entités distinctes :

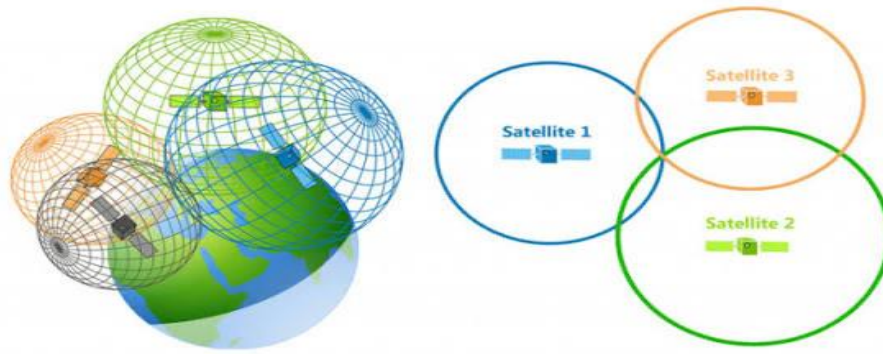


Figure1.4 : illustration du principe de positionnement par satellites et les trois segments du système GPS [10]

Le Segment Spatial : Ce segment comprend une constellation d'au moins 24 satellites en service (sans compter quelques satellites de réserve) qui gravitent autour de la Terre à approximativement 20 200 km d'altitude. Ils sont disposés sur six niveaux orbitaux pour assurer la visibilité d'au moins quatre satellites depuis n'importe quel endroit de la Terre en permanence.

.

Le Segment de Contrôle : Il s'agit des stations terrestres (la station principale se trouve au Colorado, États-Unis) qui assurent la surveillance des satellites, leur correction de trajectoire et leur synchronisation d'horloges atomiques. Sans cette composante, la dérive des horloges rendrait le système inopérant en quelques heures.

Le Segment Utilisateur : Il s'agit de destinataire. Il capte les signaux des satellites de manière passive sans leur transmettre quoi que ce soit. Il détermine sa position en examinant la durée nécessaire au signal pour faire le trajet du satellite à l'antenne.

1.2.2 Le Calcul de la Position : La Trilatération

La trilatération est une méthode de calcul qui s'apparente à la triangulation pour le calcul de positions. Au lieu de s'appuyer sur des mesures d'angles, comme l'ont longtemps fait les navigateurs en vue des côtes, la trilatération repose sur des mesures de distances.

Une fois connue la distance le séparant d'un satellite, le récepteur GPS « sait » qu'il se trouve

à la surface d'une sphère centrée sur ce satellite. Un satellite n'est donc pas suffisant pour qu'il puisse déterminer sa position. En fait, quatre satellites sont nécessaires pour mesurer une position en trois dimensions (latitude, longitude et altitude, ou encore X, Y, Z). L'un des quatre satellites a pour mission de synchroniser l'horloge interne du récepteur GPS [11]

1.2.3 Sources d'Erreurs et Précision :

1. Erreurs liées au système :

Ces erreurs sont inhérentes au système GPS. Hormis le brouillage volontaire des militaires américain (interrompu depuis le 1er mai 2000), ces erreurs sont au nombre de trois :

- **La propagation dans les différentes couches atmosphériques (l'ionosphère et la troposphère) :** les effets de cette propagation sur le signal sont d'autant plus dommageables que le satellite est bas sur l'horizon.
- **Les erreurs d'horloge :** malgré leur précision, les horloges atomiques embarquées sur les satellites n'ont pas une précision infinie...
- **Les erreurs d'éphémérides :** à cause de la proximité d'astres tels que la Terre, la Lune ou le Soleil, les satellites dérivent en permanence par rapport à leur trajectoire initiale. Leur trajectoire est donc recalculée quotidiennement par cinq postes de contrôle terrestres. Ceux-ci mettent ensuite à jour les trajectoires de chacun des satellites par une commande radio...

Le cumul de ces erreurs limite la précision des récepteurs GPS à des précisions comprises entre 5 et 10 mètres en horizontal avec 95% de probabilité, même si la précision « relative » d'une position à l'autre est beaucoup plus importante. La précision verticale (altitude) peut être jusqu'à deux fois moins bonne qu'en horizontal.

Plusieurs méthodes permettent d'améliorer la précision des mesures. Elles sont regroupées sous la dénomination de GPS Mode Différentiel (ou DGPS – cf § 3)

2. Erreurs liées à l'environnement

- **Les multi-trajets :** survient lorsque le signal diffusé par le satellite parvient au récepteur GPS après réflexion sur des obstacles proches, tels que des arbres, bâtiments, étang (pour les

satellites bas sur l'horizon), etc... Lorsque cela est possible, il est préférable de se tenir à distance de tels obstacles, qui diminuent potentiellement le nombre de satellites visibles.

• **Une mauvaise géométrie des satellites :** les satellites sont répartis sur 6 orbites, et effectuent environ deux fois le tour de la Terre par jour. Ils ne sont donc pas géostationnaires. Suivant l'heure de la mesure, les satellites peuvent être plus ou moins favorablement éclatés dans le ciel. La qualité de la géométrie des satellites dans le ciel est représentée par le PDOP. Plus la valeur du PDOP est faible, meilleure sera la précision de la mesure. Les systèmes GPS professionnels permettent à l'utilisateur de spécifier la valeur maximum du PDOP autorisée pendant la mesure. Un PDOP inférieur à 6 est généralement conseillé par les constructeurs de récepteur GPS. Suivant la précision recherchée, un PDOP maximum de 8 peut être toléré dans les environnements particulièrement difficiles tels que la forêt ou la ville [11]

1.2.4 L'importance de la Synchronisation Temporelle et les Horloges Atomiques :

Pour justifier pourquoi le GPS repose sur le principe du temps, il est nécessaire de recourir à la physique.

La formule essentielle :

En mesurant le délai de propagation (Δt) du signal radio, le récepteur détermine la distance (d) qui sépare l'antenne du satellite :

$$d = c \times \Delta t \quad (\text{I.1})$$

Où c est la vitesse de la lumière et Δt le décalage temporel

C'est à peu près la vitesse de la lumière, soit $3 \cdot 10^8$ m/s.

Exemple simple (Dans le vide) :

Imaginons qu'un satellite envoie un signal.

- Le signal indique qu'il a été émis à : **12:00:00.000 000**
- Le récepteur le reçoit à : **12:00:00.067 000**

Étape 1 : Calcul du délai de propagation (Δt)

$$\Delta t = 12:00:00.067000 - 12:00:00.000000 = 0,067 \text{ secondes (ou 67 ms)}$$

Étape 2 : Calcul de la distance (d)

En utilisant la vitesse de la lumière arrondie $c \approx 3 \cdot 10^8$ m/s :

$$d = c \times \Delta t$$

$$d = 3 \cdot 10^8 \text{ m/s} \times 0,067 \text{ s}$$

$$d = 20\,100\,000 \text{ mètres (soit } 20\,100 \text{ km)}$$

Pourquoi une telle précision ?

Si l'horloge du satellite ou celle du récepteur présente un décalage de seulement 1 microseconde (10^{-6}) :

300000000 m/s multiplié par 0,000001 s équivaut à une erreur de 300 mètres.

Pour atteindre une précision de 3 mètres, un intervalle temporel de 10 nanosecondes est nécessaire.

Le rôle des horloges atomiques :

Le principe de fonctionnement d'une horloge atomique repose sur l'asservissement de la fréquence d'un oscillateur sur une fréquence de référence absolue correspondant à la fréquence de transition entre deux états quantiques d'un atome, d'un ion ou d'une molécule. Le signal utile de l'horloge est généré par un oscillateur dont le type dépend de la gamme de fréquence considérée. Afin de comparer la fréquence ν de l'oscillateur à la fréquence de référence atomique ν_{at} , une partie de ce signal est prélevé. Une synthèse de fréquence permet d'adapter la fréquence de l'oscillateur au domaine de fréquence de la référence atomique en générant un signal d'interrogation de fréquence ν_{int} proche de ν_{at} . Cette synthèse est aussi utilisée pour contrôler finement les caractéristiques du signal qui va interagir avec les atomes : amplitude, ajout d'un décalage de fréquence, modulation d'amplitude et/ou de fréquence, [12]

1.2.5 Le Protocole NMEA 0183 : Langage du Module GPS

Le GPS interagit avec le microcontrôleur au moyen de chaînes de caractères nommées « trames NMEA ». La trame la plus complète pour notre projet est la \$GPRMC (Information de Navigation Minimum Recommandée).

Exemple de trame à analyser :

\$GPRMC,123519,A,4807.038,N,01131.000,E,022.4,084.4,230326,003.1,W*6A

Tableau 1.1 Un tableau pour détailler chaque champ :

Champ	Valeur exemple	Description
\$GPRMC	Identifiant	Type de trame (recommended minimum)
123519	UTC Time	Heure face à 12 :35 :19 UTC
A	Statu	A=Données valides, V=Alerte (pas de signal)
4807.038, N	Latitude	48 degrés, 07.038 minutes Nord
01131.000, E	Longitude	11 degrés, 31.000
022.4	Speed	Vitesse sur le sol en nœuds (à convertir en km/h)
230326	Date	Date du jour (23 mars 2026)
*6A	Checksum	Somme de contrôle pour vérifier l'intégrité des données

1.3 Technologies de Communication Mobile : Réseaux Télécoms

L'efficacité d'un système de détection de collision dépend de sa faculté à diffuser l'alerte de façon fiable, même en situation compromise. Cette partie examine les infrastructures mobiles ainsi que les protocoles de communication conçus pour la sécurité routière.

1.3.1 Comparaison Détaillée des Générations Mobiles (2G à 5G)

Une mise en parallèle de ces technologies

Depuis l'avènement de la téléphonie mobile, les réseaux ont connu des progrès pour fournir

une plus grande bande passante et une latence réduite. Voici une mise en parallèle de ces technologies :

Tableau 1.2 Comparaison Détaillée des Générations Mobiles (2G à 5G)

Génération	Nom Technique	Débit Max	Caractéristique pour le projet
2G	GSM/GPRS	9.6 à 114 kbps	Idéal : Couverture maximale, très fiable pour les SMS.
3G	UMTS/HSPA	2 à 42 mbps	Permet l'envoi de données plus lourdes mais consomme plus.
4G	LTE	100 à 300 mbps	Excellente pour la vidéo en temps réel, mais modules coûteux.
5G	NR (New Radio)	>1 Gbps	Latence ultra-faible, conçue pour les véhicules autonomes.

Analyse pour le projet : Malgré la domination de la 4G et de la 5G sur le marché des smartphones, la technologie 2G (GSM) demeure le standard privilégié pour les systèmes de sécurité embarqués simples. Son utilisation est plus étendue en milieu rural et son coût matériel est considérablement réduit pour des besoins de transmission de quelques octets (comme les coordonnées GPS).

1.3.2 Protocoles de Transmission : SMS vs GPRS/Data

Pour envoyer l'alerte, deux méthodes principales s'offrent au concepteur :

- **Le service de message court (SMS) :**

Bénéfices : Il exploite le canal de signalisation du réseau, ce qui permet l'envoi de SMS même lorsque le réseau est engorgé pour la voix ou les données. Il ne requiert pas une connexion internet en cours d'utilisation.

Inconvénients : Restreint à 160 caractères (ce qui est suffisant pour des coordonnées GPS).

- **GPRS / Données (Internet Mobile) :**

Atouts : Autorise la transmission des informations vers un serveur internet ou une base de données en direct pour un suivi sur une carte.

Inconvénients : Nécessite une couverture réseau stable. Si la connexion internet est coupée, l'alerte n'est pas envoyée

❖ **Option technologique :** Dans le cas d'un système de « détection d'accident », la solidité du SMS est fréquemment préférée en tant que solution de secours (fail-safe) car elle assure la transmission du message même dans les zones à faible signal.[13]

1.3.3 Architecture du Réseau GSM

Le système GSM est structuré en sous-systèmes facilitant la gestion des appels et des messages :

- **Le Mobile Station (MS) :**

L'unité mobile est constituée de deux composants essentiels et indissociables : l'équipement mobile (ME) — un système intégré basé sur un microcontrôleur et un module de communication GSM — et la carte SIM (Subscriber Identity Module). Ce dispositif sert d'interface entre l'utilisateur (ou les capteurs du véhicule) et le réseau mobile pour la transmission des données de secours.

- **Le Base Station Subsystem (BSS) :**

1. **(Base Transceiver Station) :** C'est l'antenne relais qui communique directement avec notre module.
2. **BSC (Base Station Controller) :** Il gère les ressources radio pour plusieurs BTS.
 - **Le Network and Switching Subsystem (NSS) :** C'est le cœur du réseau. Il contient le MSC (Mobile Switching Center) qui route les messages, ainsi que les bases de données HLR (Home Location Register) et VLR (Visitor Location Register) qui gèrent l'identité et la position de l'abonné.[14]

Note technique : En cas d'accident, le module GSM analyse les stations de base environnantes pour se connecter à la plus puissante. Dans le cas où le réseau de l'opérateur de la carte SIM est manquant, le système a la possibilité, en respectant les protocoles d'urgence, d'utiliser n'importe quel BTS disponible (itinérance d'urgence).

1.4 Intellection Technique et Synergie du Système Embarqué :

L'implémentation d'un dispositif de détection d'accident ne se limite pas à une simple combinaison de pièces. C'est une structure complexe où le microcontrôleur est chargé de gérer des flux de données asynchrones, de respecter des contraintes énergétiques strictes et d'assurer des priorités de traitement en temps réel. Cette partie examine en détail les processus internes de cette fusion technologique.[15]

1.4.1 Architecture Matérielle Avancée (Hardware Intégration)

Le cœur du système est une unité de traitement (Microcontrôleur) qui assure l'interface entre le monde analogique (les capteurs de choc) et le monde numérique (les modules de communication) [16].

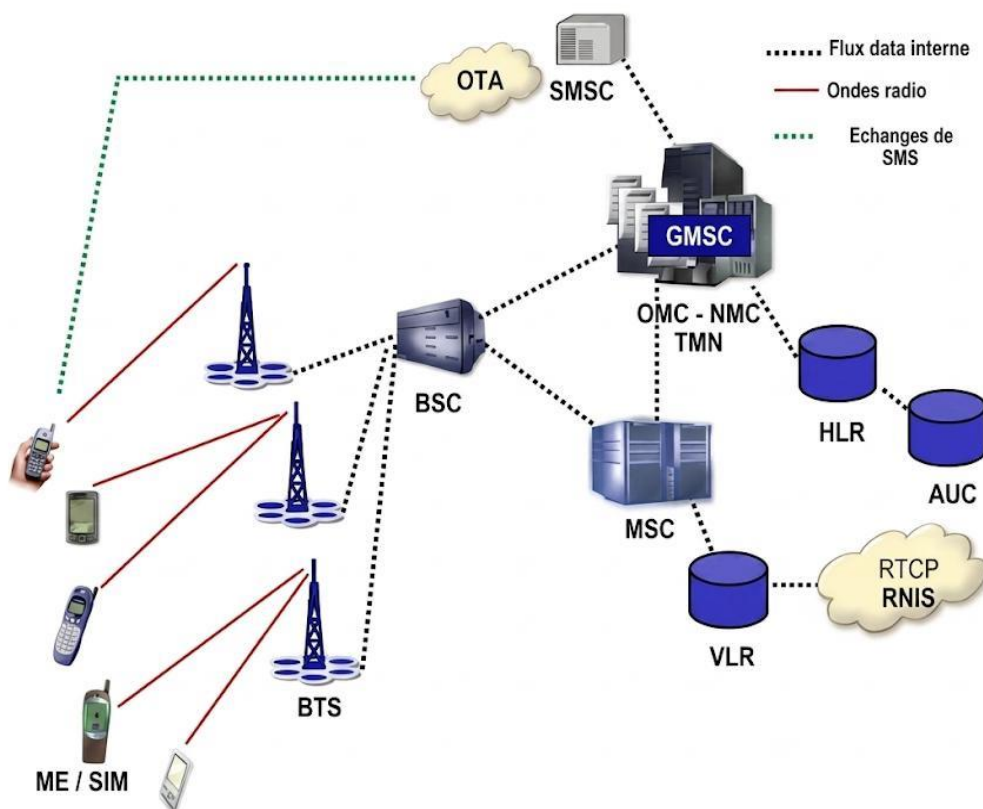


Figure 1.5 : architecture détaillée du réseau GSM présentant les interactions entre le sous-système radio (BSS) et le sous-système réseau (NSS)

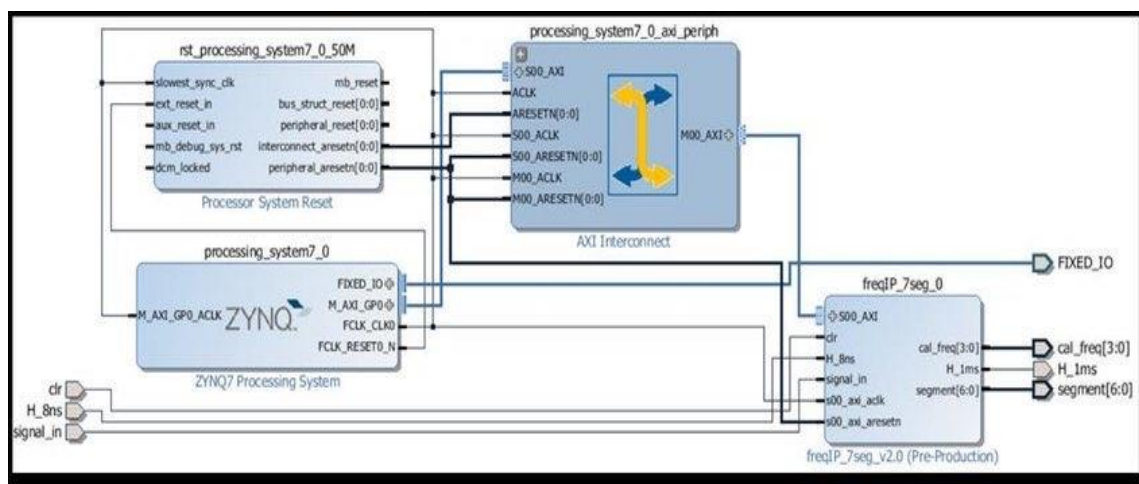


Figure 1.6 : conception du système de contrôle matériel sur cible [17]

1.4.1.1 Le Microcontrôleur : Chef d'orchestre du système

Le choix du microcontrôleur (qu'il soit basé sur une architecture AVR comme l'Arduino, ARM comme le STM32 ou PIC) se fait en fonction de sa capacité à gérer plusieurs ports de communication en parallèle.

- **La gestion des interruptions (ISR - Interrupt Service Routines) :** C'est l'aspect le plus critique. Le microcontrôleur ne doit pas "attendre" les données. Il doit être capable d'interrompre n'importe quelle tâche secondaire (comme l'affichage de l'heure) dès que l'accéléromètre détecte un dépassement de seuil (ex: > 4G).
- **La mémoire tampon (Buffer) :** Le CPU alloue une zone de mémoire RAM pour stocker temporairement les trames GPS. Cela permet de disposer de la position exacte même si le module GPS subit une coupure d'alimentation au moment précis du choc.

1.4.1.2 Communication Inter-modules via le protocole UART

L'échange de données entre le cerveau et les modules GPS/GSM s'effectue via l'interface UART (Universal Asynchronous Receiver-Transmitter).

- **Trame de données :** Chaque octet est encadré par un bit de "Start" et un bit de "Stop".
- **Niveaux logiques :** Une attention particulière doit être portée aux tensions. Si le microcontrôleur fonctionne en 5V et le module GPS en 3.3V, l'intégration nécessite un pont diviseur de tension ou un adaptateur de niveau logique (Logic Level Shifter) pour éviter la destruction des composants.[18]

1.4.2 Logique de Traitement et "Parsing" des Données GPS

Le module GPS envoie un flux continu de caractères au format NMEA 0183. Le microcontrôleur doit effectuer une opération de "Parsing" (analyse syntaxique).

1. **Extraction sélective** : Parmi toutes les phrases envoyées (\$GPGGA, \$GPGSV, \$GPRMC), le microcontrôleur est programmé pour ne capturer que la trame \$GPRMC (Recommended Minimum Navigation Information).
2. **Algorithme de découpage** : Le programme recherche le caractère de début '\$', puis utilise la virgule ',' comme délimiteur pour isoler les variables :

Variable 1 : Heure UTC.

Variable 2 : État (A=Validé, V=Invalidé).

Variables 3 & 5 : Latitude et Longitude brutes.

3. **Conversion Mathématique** : Les coordonnées GPS arrive souvent au format DD MM.MMM (Degrés Minutes). Pour être exploitables par les secours via une interface moderne, le microcontrôleur doit les convertir en Degrés Décimaux (DD) via la formule :

$$\text{Degrés décimaux} = \text{degrés} + \frac{\text{MINUTES}}{60}$$

1.4.3 Dialogue par Commandes AT : Le protocole GSM

Le pilotage du module GSM est une procédure séquentielle rigoureuse. Le microcontrôleur doit gérer les réponses du module pour s'assurer que l'alerte est bien partie.

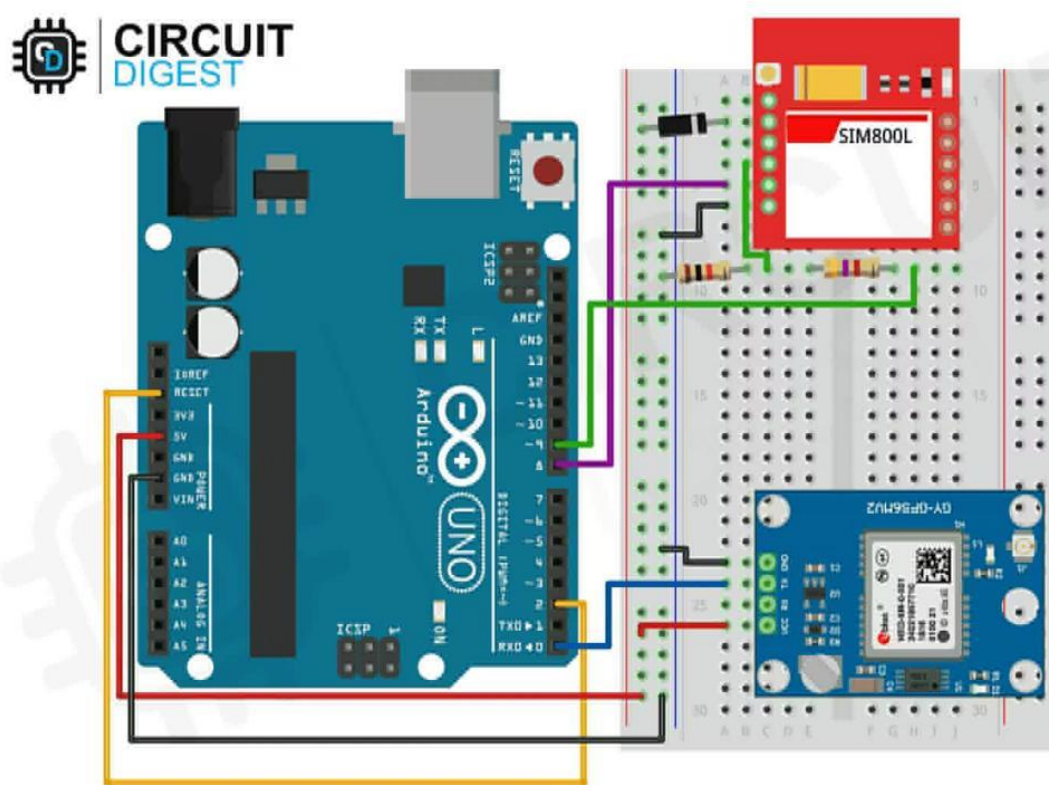


Figure 1.7 : Schema de câblage du prototype incluant l'Arduino uno le module GSM SIM800L et le récepteur GPS Neo-6M pour la transmission des coordonnées de l'accident [19]

- **La boucle de rétroaction (Feedback) :** Après chaque commande envoyée par le CPU (ex : AT+CMGS), le programme doit attendre la réponse OK ou >. Si le module répond ERROR (par exemple à cause d'un manque de crédit ou d'absence de réseau), le microcontrôleur doit exécuter une procédure de "Retry" (tentative de renvoi) programmée.
- **Encodage PDU vs Mode Texte :** Bien que le mode PDU (binaire) soit plus robuste, l'intégration logicielle utilise généralement le Mode Texte (AT+CMGF=1) pour faciliter l'insertion dynamique des liens de géolocalisation dans le corps du SMS.

1.4.4 Contraintes Électromagnétiques et Environnementales

Un aspect souvent oublié mais crucial pour un mémoire de 30 pages est l'impact de l'environnement automobile sur l'intégration :

1. **Les interférences (EMI) :** Le module GSM émet des ondes radio puissantes qui peuvent "éblouir" l'antenne GPS très sensible. L'intégration physique impose un blindage électromagnétique ou une distance minimale entre les deux antennes.
2. **Les vibrations et chocs :** Dans un véhicule, les soudures peuvent céder. L'intégration doit prévoir l'utilisation de connecteurs sécurisés et une fixation robuste sur un circuit imprimé (PCB) rigide.
3. **Gestion thermique :** En été, l'habitacle d'une voiture peut atteindre 70°C. Les composants choisis pour l'intégration doivent être de grade industriel (températures de -40°C à +85°C).

1.5 Études Approfondies des Solutions de Sécurité Existantes

L'émergence des technologies de l'information a donné naissance à plusieurs solutions de sécurité routière. Cette section analyse en profondeur les mécanismes de ces systèmes pour justifier la pertinence d'une solution autonome basée sur le GSM/GPS.

1.5.1 Le Système eCall : Une Standardisation Institutionnelle

Le projet eCall, initié par la Commission Européenne, représente la tentative la plus aboutie d'intégration de la télématique dans la sécurité publique.

- **L'Infrastructure PSAP (Public Safety Answering Point) :** Le système eCall ne se contente pas d'envoyer un SMS. Il établit une priorité sur le réseau mobile (eCall flag) qui permet d'écarter les autres appels pour laisser passer l'alerte. Cette priorité est gérée au niveau du cœur de réseau (MSC), une complexité que les systèmes artisanaux ne peuvent pas atteindre sans accords avec les opérateurs.
- **Le MSD (Minimum Set of Data) :** Le message envoyé pèse exactement 140 octets. Il inclut l'identification du véhicule (VIN), le type de propulsion (électrique, essence, gaz) pour prévenir les risques d'explosion pour les pompiers, et la direction de conduite (pour savoir sur quel côté de l'autoroute l'accident a eu lieu).

1.5.2 Les Systèmes Constructeurs Propriétaires (OnStar, BMW Assist)

Avant l'eCall, des constructeurs comme General Motors ou BMW avaient leurs propres services.

- **Le Modèle de Conciergerie** : Contrairement à notre projet qui est direct, ces systèmes passent par un centre d'appel privé. Un opérateur humain filtre l'alerte avant d'appeler les secours [20].

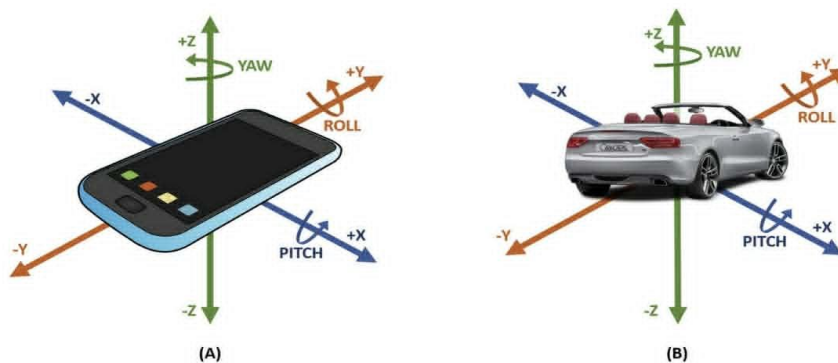


Figure 1.8 : référentiels tridimensionnels du capteur et de véhicule pour la synchronisation des données inertielles

- **Le coût du service** : Ces systèmes fonctionnent sur abonnement. Si l'abonnement n'est pas payé, la fonction de sécurité est souvent désactivée, ce qui pose un problème éthique majeur que notre solution "Open Source" et autonome cherche à résoudre.

1.5.3 Analyse Critique des Capteurs de Smartphones

L'utilisation d'applications mobiles (comme SOS Autoroute) est souvent proposée comme alternative. Cependant, une analyse technique révèle des failles critiques

1. **Le problème du "False Positive" (Faux Positifs)** : Un smartphone qui tombe d'un support peut générer une accélération de 10G, soit plus qu'un accident réel. Sans être fixé solidement au châssis du véhicule, l'algorithme ne peut pas distinguer un choc mineur d'un accident grave.

2. **L'atténuation du signal GPS :** Placés souvent près du levier de vitesse ou dans une poche, les smartphones subissent l'effet de "cage de Faraday" du toit métallique de la voiture, réduisant la précision de localisation de 5 à 50 mètres, ce qui est critique pour les secours.

- **Synthèse du Chapitre 1 et Positionnement Problématique :**

Ce premier chapitre a permis de poser les fondements théoriques et contextuels de notre étude. Nous avons démontré que :

- La rapidité d'alerte est le facteur numéro un de survie (L'Heure d'Or).
- Le GPS et le GSM forment un couple technologique mature, robuste et peu coûteux, capable de couvrir la quasi-totalité du réseau routier.
- L'unité centrale (Microcontrôleur) doit non seulement traiter des données, mais aussi gérer des priorités énergétiques et logicielles complexes.

1.6 Analyse Justificative du Choix Technologique : Pourquoi le couple GSM/GPS ?

Après avoir exploré les différentes technologies de positionnement et de transmission, il convient de justifier pourquoi l'architecture basée sur le GSM et le GPS reste la solution optimale pour un système de détection d'accident universel.

1.6.1 Accessibilité Économique et Démocratisation (Le facteur Coût)

L'un des piliers de notre projet est de proposer une solution "Low-Cost" sans sacrifier la fiabilité.

- **Comparaison avec les systèmes embarqués d'origine :** Alors qu'un système constructeur (comme l'eCall) a facturé plusieurs centaines d'euros en option ou inclus dans le prix d'achat d'un véhicule neuf, les modules SIM800L (GSM) et Neo-6M (GPS) représentent un investissement matériel inférieur à 20 euros.
- **Maintenance et Évolutivité :** En utilisant des composants standards, les coûts de réparation sont quasi nuls. De plus, cela permet de proposer un kit de "retrofit" pour les millions de véhicules anciens qui ne seront jamais équipés de série.

1.6.2 La Robustesse de la Couverture Mondiale (Le facteur Réseau)

Il existe d'autres technologies de transmission de données comme le LoRaWAN ou le Sigfox (réseaux basse consommation), mais elles ne sont pas adaptées à l'urgence routière :

- **Disponibilité du GSM :** Le réseau GSM (2G) est le réseau mobile le plus étendu au monde. Dans les zones montagneuses ou rurales (là où les accidents sont les plus dangereux car isolés), le signal 2G est souvent le seul à pénétrer les obstacles géographiques.
- **Roaming International :** Contrairement à d'autres protocoles, le GSM permet une itinérance (roaming) transparente. Un véhicule traversant une frontière continuera d'émettre son alerte sur n'importe quel réseau disponible, garantissant une continuité de service vitale.

1.6.3 Simplicité de Mise en Œuvre et Standardisation (Le facteur Ingénierie)

Le choix du GPS et du GSM facilite grandement l'intégration logicielle pour plusieurs raisons :

1. **Universalité des Commandes AT :** Presque tous les modules GSM du marché utilisent le même jeu de commandes AT. Cela rend le code portable d'un module à un autre.
2. **Standard NMEA :** Le format de données GPS est universel. Si l'on souhaite remplacer le module GPS par un modèle plus précis (Galileo), la structure des trames à analyser par le microcontrôleur reste identique.
3. **Indépendance vis-à-vis d'Internet :** En utilisant le SMS comme vecteur principal, le système ne dépend pas de la stabilité d'une connexion GPRS/4G ou de la validité d'une adresse IP, ce qui réduit drastiquement les risques de "bugs" au moment critique. [21]

1.6.4 Fiabilité des Capteurs Dédiés face aux Smartphones

Comme précédemment analysé, la solution matérielle fixe est supérieure aux applications mobiles. La présence d'un accéléromètre intégré à un circuit imprimé, qui est par ailleurs solidement ancré au châssis de la voiture, assure une mesure précise des vecteurs de décélération, sans aucune interférence (vibrations causées par un téléphone dans un compartiment de rangement). [22]

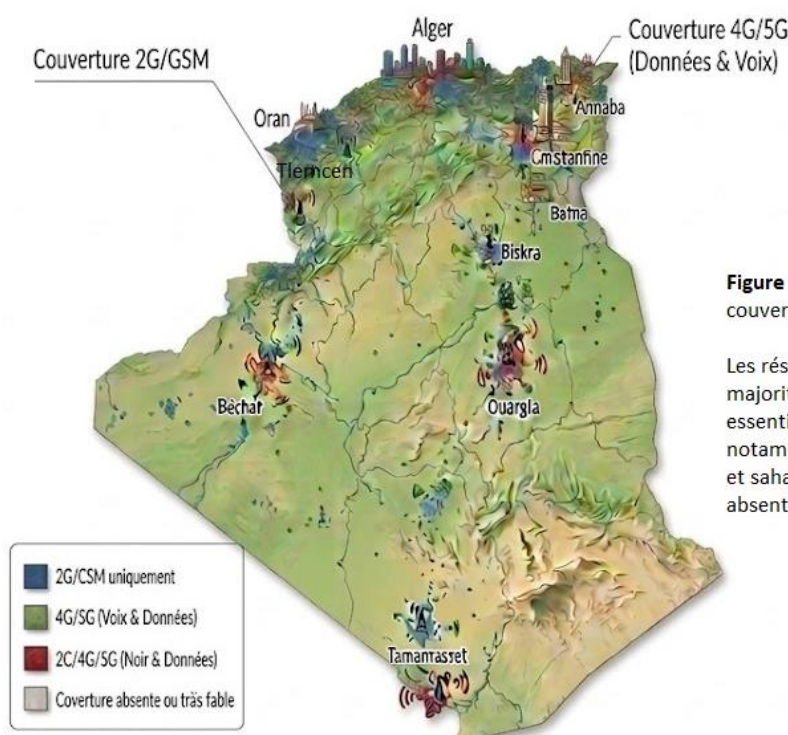


Figure 1.10 : Comparaison de la couverture réseau en Algérie

Les réseaux GSM couvrent la majorité du territoire et restent essentiels pour les SMS d'urgence, notamment dans les zones rurales et sahariennes où la 4G/5G est absente ou limitée.

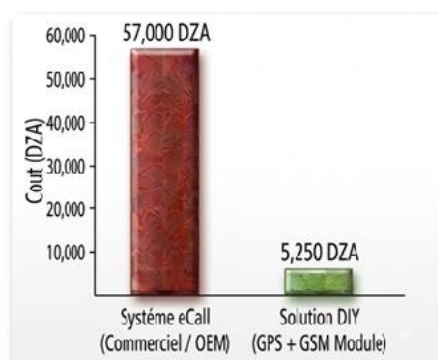


Figure 1.11 : Analyse des coûts matériels
Les solutions open-source et DIY permettent une réduction significative des coûts par rapport aux systèmes propriétaires, tout en assurant la souplesse et l'évolutivité du système.

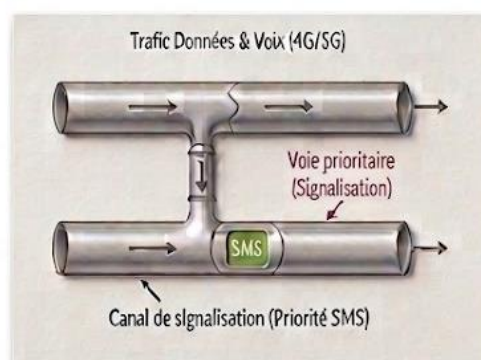


Figure 1.12 : Mécanisme de robustesse des SMS
Les SMS circulent sur un canal dédié, prioritaire et indépendamment des réseaux de données/voix. Ils sont ainsi délivrés même en cas de congestion du réseau ou de défaillance partielle durant une situation d'urgence.

Figure 1.9 : justification technico-économique de la solution : (1.10) étendue de la couverture réseau, (1.11) étude comparative des coûts entre systèmes propriétaires et solutions DIY, (1.12) mécanisme de robustesse de la transmission par SMS.

Afin de mieux illustrer le fonctionnement du système avec et sans intelligence artificielle, deux organigrammes ont été réalisés et sont présentés dans les figures 1 et 2 (ANNEXE)

Organigramme 1 : Diagramme de la méthode algorithmique traditionnelle (Seuils fixes)

Ce diagramme illustre le fonctionnement logique du système basé sur une méthode déterministe. Le déroulement suit une chaîne linéaire de traitement des informations :

Acquisition et Traitement : Suite à l'étalonnage et la mise en service, le dispositif recueille les informations primaires des capteurs (accéléromètre, gyroscope, GPS) afin d'évaluer des paramètres essentiels tels que la force de l'impact et le changement d'angle.

Décision basée sur des seuils : La détection s'effectue en comparant les valeurs calculées à des seuils préétablis. Si les valeurs excèdent ces seuils, le système identifie deux cas de figure : l'accident (situation critique) ou la conduite à risque (mesure préventive).

Actions d'urgence : Si un accident est détecté, le système déclenche immédiatement un protocole de protection (arrêt du moteur, alarme sonore continue) et de secours (transmission des coordonnées GPS via message texte).

Organigramme 2 : Structure de l'Intelligence Artificielle

Ce diagramme de flux dépeint la progression du système vers une méthode prédictive qui fait appel à un modèle d'apprentissage automatique. La complexité et la précision de l'architecture logicielle s'améliorent :

Intégration du Modèle : La procédure de calcul manuel est substituée par une étape d'inférence. Le modèle d'intelligence artificielle évalue simultanément toutes les données des capteurs afin de déterminer l'état de conduite.

Classification Intelligente : Plutôt que de s'appuyer sur des seuils stricts, l'IA divise les événements en trois catégories distinctes (Classe 1 : Accident, Classe 2 : Conduite à risque, Classe 3 : Normale). Cela facilite la différenciation entre un véritable choc et une irrégularité

de voie (nid-de-poule, freinage soudain).

Amélioration de la Réaction : La procédure d'alerte demeure inchangée pour assurer la sécurité, cependant le mécanisme de déclenchement est désormais plus fiable, diminuant ainsi les alertes erronées grâce à la précision du modèle d'intelligence artificielle qui distingue les chocs réels des simples freinages.

CHAPITRE II

Matériel et environnement de développement

2.1 Présentation de la plateforme Arduino

2.1.1 Carte Arduino

ARDUINO est une plateforme matérielle et logicielle de développement d'applications embarquées.

Une équipe de développeurs composée de Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino, David Mellis et Nicholas Zambetti a imaginé un projet répondant portant le nom d'Arduino et mettant en œuvre une petite carte électronique programmable et un logiciel multiplateforme, conçu pour être à la portée de tous, dans le but de créer facilement des systèmes électroniques. Étant donné qu'il y a des débutants parmi nous, commençons par voir un peu le vocabulaire commun propre au domaine de l'électronique et de l'informatique

- Côté matériel, une carte électronique est un support plan, flexible ou rigide, généralement composé d'époxy ou de fibre de verre. Elle possède des pistes électriques disposées sur une, deux ou plusieurs couches (en surface et/ou en interne) qui permettent la mise en relation électrique des composants électroniques. Chaque piste relie tel composant à tel autre, de façon à créer un système électronique qui fonctionne et qui réalise les opérations demandées.
- Le logiciel de programmation des modules ARDUINO est une application Java, libre et multi- plateformes, servant d'éditeur de code et de compilateur, et qui peut transférer le programme au travers de la liaison USB. Le langage de programmation utilisé est un mélange de C et de C++, restreint et adapté aux possibilités de la carte.[1]

2.1.2 L'alimentation

La carte ARDUINO UNO est alimentée via un port USB ou une source d'alimentation externe de 7 à 20 volts. La tension d'entrée est régulée par un régulateur de tension intégré, qui permet à la carte de fonctionner à une tension constante de 5 volts.

Arduino est compatible avec une grande variété des composants électroniques, tels que des écrans LCD, des capteurs, des relais et des moteurs. Elle dispose également d'une interface série pour la communication avec d'autres périphériques.

2.1.3 Les broche d'Arduino Uno

Arduino Uno est équipé de 14 broches d'E/S numériques et de 6 broches d'E/S analogiques.

Tableau 2.1 : Tableau des correspondances entre les broches E/S et les fonctions alternatives du microcontrôleur :

Elément	Fonction
VIN	La tension d'entrée de la carte lorsqu'elle utilise une source d'alimentation externe.
Alimentation 3.3V	Cette alimentation est utilisée pour l'alimentation de la carte.
Alimentation 5V	Cette alimentation est générée par un régulateur de tension.
GND	Utilisée pour la connexion à la masse.
Interface SPI	Interface de communication série synchrone utilisée pour communiquer avec des périphériques externes.
Interface PWM	Interface de modulation de largeur d'impulsion utilisée pour contrôler la vitesse des moteurs et des <u>LED</u> .
Connecteur USB	Connecteur de type B utilisé pour la programmation et l'alimentation de la carte.
RST	Il est utilisé pour la réinitialisation du microcontrôleur.
Broche série RX/TX	RX est utilisé pour recevoir des données et TX pour la transmission des données.
Interface I2C	Interface de communication série bidirectionnelle utilisée pour communiquer avec des périphériques externes.
Interruption externe	Elle sert à déclencher une interruption sur un changement de valeur.
Interface AREF	Elle est utilisée pour fournir une tension de référence pour les entrées analogiques.

2.1.4 Domaines d'Application Arduino Uno

L'Arduino Uno est largement utilisé dans de nombreux domaines, notamment :

- Automatisation domestique
- Robotique
- Domotique
- Internet des objets (IoT)
- Santé et dispositifs médicaux

- Acquisition et analyse de données
- Agriculture intelligente

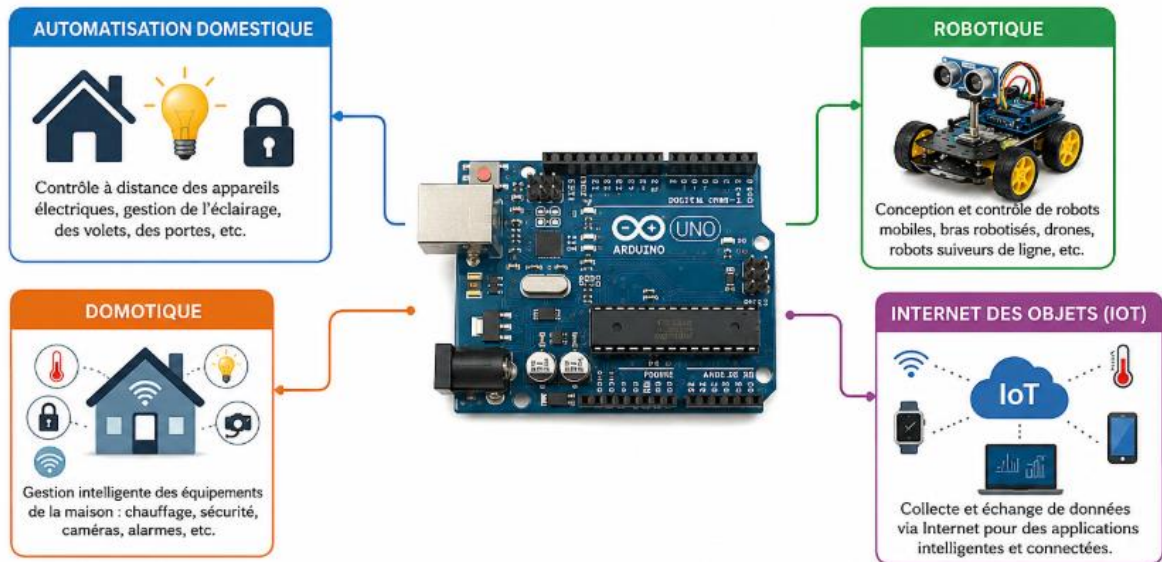


Figure 2.1 : image qui représente quelque domaine d'utilisation Arduino Uno [24]

2.1.5 Avantages

- Pas cher !
- Environnement de programmation clair et simple.
- Multiplate-forme : tourne sous Windows, Macintosh et Linux.
- Nombreuses bibliothèques disponibles avec diverses fonctions implémentées.
- Logiciel et matériel open source et extensible.
- Nombreux conseils, tutoriaux et exemples en ligne (forums, site perso, etc.)
- Existence de « shield » (boucliers en français).

2.2 Caractéristique techniques de la carte Arduino et du Microcontrôleur

2.2.1 La carte ARDUINO UNO

La carte Arduino UNO comprend plusieurs composants essentiels, avec un microcontrôleur jouant un rôle central. La figure 2.3 montre les différents éléments qui composent la carte Arduino UNO. [25]

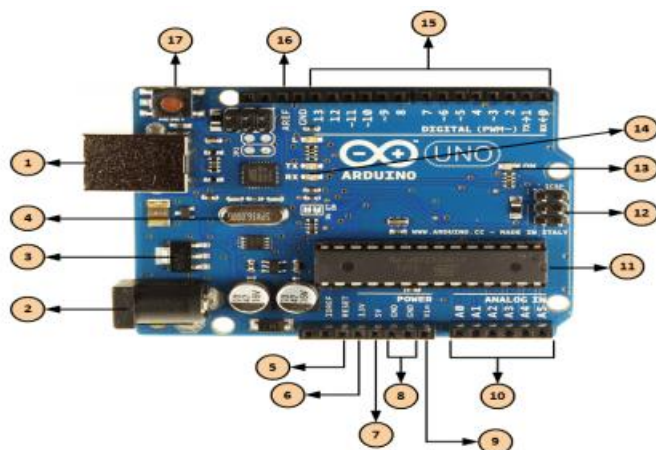


Figure 2.2 : Schéma descriptif des composants de la carte Arduino Uno [26]

La carte Arduino Uno est une plateforme de développement électronique open-source populaire, conçue pour créer des projets interactifs ou automatisés. Basée sur le microcontrôleur ATmega328P, elle offre des entrées/sorties numériques et analogiques ainsi que diverses 10 interfaces de communication. Idéale pour les débutants, elle est programmable via l'environnement de développement Arduino et est utilisée dans une gamme variée de projets, grâce à sa simplicité d'utilisation et à sa polyvalence. Elle est dotée de :

1. Il est possible d'alimenter la carte Arduino en utilisant un câble USB connecté à ordinateur afin de la programmer. Il vous faut simplement relier carte Arduino à ordinateur à l'aide du câble USB A/B. [27]

2. Alimentation à l'aide du connecteur Jack DC : En utilisant ce connecteur Jack DC, il est envisageable d'alimenter directement la carte Arduino. Ce câble est relié au dispositif de régulation de tension qui est inclus dans la carte. Il est nécessaire que l'alimentation via ce connecteur soit de 5 à 12 V.

3. Régulateur de tension : Le rôle du régulateur de tension consiste à Prendre en charge la tension d'alimentation de l'Arduino pour une tension appropriée pour le microcontrôleur et chaque élément de la carte. Les cartes Arduino UNO ont une tension de stabilisation de 5 Volts.

4. Oscillateur à quartz : L'oscillateur à quartz est un composant électronique contenant un quartz qui vibre grâce à l'effet piézoélectrique. Cette vibration, due aux propriétés électromécaniques du quartz, produit une fréquence extrêmement précise. Sur l'Arduino UNO, cet accessoire simplifie le calcul des données temporelles. En regardant le composant, on peut lire la mention "16.000". Cela indique que sa fréquence est de 16 millions de cycles par seconde, soit 16 MHz.

5. Arduino Reset : Il est possible de redémarrer programme depuis le début en connectant un bouton externe sur la broche de la carte Arduino indiquant "RESET".

6. 3.3V : Broche d'alimentation à 3.3 Volts de tension.

7. 5V : Broche d'alimentation à 5 Volts de tension.

8. GND : La carte Arduino contient plusieurs broches de ce genre, toutes sont communes et peuvent servir de masse (potentiel 0 Volts) pour vos circuits.

9. VIN : Grâce à cette broche, l'Arduino peut être alimenté depuis une source de tension externe. Elle est connectée au circuit principal d'alimentation de la carte Arduino.

10. Broches analogiques : Les 6 broches d'entrée analogiques de l'Arduino UNO sont numérotées de A0 à A5. Grâce à ces broches, il est possible de lire un signal analogique provenant d'un capteur tel que le capteur d'humidité ou chauffage. Un convertisseur analogique/numérique (convertisseur CAN) est utilisé par la carte Arduino afin de permettre la lecture du signal par le microcontrôleur. On convertira un signal en 10 bits.

On pourra lire la valeur sur une échelle de 1024 points (allant de 0 à 1023).

11. Microcontrôleur ATmega328p : Chaque carte Arduino est équipée d'un microcontrôleur distinct. Il peut être perçu comme le cœur de la carte Arduino. La configuration du microcontrôleur sur l'Arduino diffère légèrement d'une carte à l'autre.

En général, les microcontrôleurs sont fabriqués par la société ATMEL. Avant de charger un nouveau programme depuis l'IDE Arduino, il est nécessaire de connaître le microcontrôleur de notre carte. Le microcontrôleur de la carte Arduino UNO est équipé de 28 pins, d'une mémoire flash de 32 kilooctets, d'une mémoire RAM de 2 kilooctets et d'une mémoire EEPROM de 1 kilooctet.

12. Connecteur ICSP : Le connecteur ICSP (In-Circuit Serial Programming) est une interface essentielle sur les microcontrôleurs AVR, comprenant les broches MOSI, MISO, SCK, RESET, VCC et GND. Il est utilisé principalement pour la programmation du microcontrôleur. Ce connecteur permet de programmer directement le microcontrôleur à un niveau bas, incluant le bootloader et le code en langage assembleur. Il fonctionne également comme un port SPI (Serial Peripheral Interface), facilitant la communication avec d'autres périphériques SPI tels que des écrans et des capteurs.

13. Indicateur LED d'alimentation : Lorsque nous alimentons notre carte Arduino, le voyant correspondant devrait s'allumer pour indiquer que l'alimentation est correcte. Si ce voyant ne s'allume pas, cela signale un problème potentiel avec l'alimentation de la carte.

14. LED RX et TX : Sur la carte Arduino UNO, vous trouverez deux indicateurs : TX (pour transmission) et RX (pour réception). Ces indicateurs sont situés sur les broches numériques 0 et 1, qui sont dédiées à la communication série. Par la suite, on trouve les LED TX et RX.

Lorsque les données sont transmises en série, le voyant TX s'allume à une vitesse différente. La vitesse de transmission de la carte influence la fréquence de clignotement. Lors de la réception, le spectromètre clignote. La vitesse de transmission, en cas de signal binaire, est donnée en bauds, c'est-à-dire en bits/seconde.

15. Entrées/sorties numériques : Il y a 14 broches d'entrées/sorties numériques sur la carte Arduino UNO, dont 6 peuvent fournir une sortie PWM (Pulse Width Modulation). Il est possible de configurer ces broches pour exercer leur fonction de broches numériques d'entrée afin de lire des valeurs logiques (0 ou 1) ou numériques. Elles peuvent aussi servir de broches de sortie pour contrôler divers modules tels que des LEDs, des relais, etc. Les broches marquées « ~ » peuvent servir à produire des PWM, une méthode qui permet d'obtenir des résultats analogiques en utilisant des moyens numériques.

16. Broche AREF : En anglais, AREF signifie "Analog Reference". Cette broche est parfois utilisée pour définir une tension de référence externe (de 0 à 5 volts), servant de limite supérieure pour les broches d'entrée analogiques.

17. Arduino Reset : Un Arduino peut être redémarré en appuyant sur le bouton "Reset". Cela provoque le redémarrage du programme depuis le début.

2.2.2 Caractéristiques techniques d'Arduino uno:

- Microcontrôleur : ATmega328
- Modèle : Arduino Uno version R3
- Tension de fonctionnement : 5V
- Tension d'alimentation recommandée : 7-12V
- Tension d'alimentation limites : 4-20V
- Nombre de broches d'E/S : 14 (dont 6 broches PWM)
- Nombre de ports analogiques/numériques : 6
- Courant maximal par broche d'E/S : 40 mA
- Courant pour la broche 3.3V : 50 mA
- Mémoire Flash : 32 KB (ATmega328)
- SRAM : 2 KB (ATmega328)
- EEPROM : 1 KB (ATmega328)
- Vitesse d'horloge : 16 MHz
- Dimensions : 74 x 53 x 15 mm
- Interfaces de communication : Bus série, I2C et SPI
- Connecteur USB : Type B

Différents modèles Arduino :

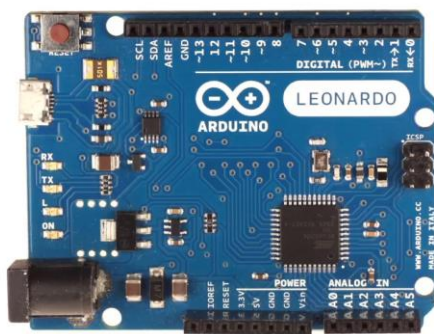
Arduino Uno



Arduino pro



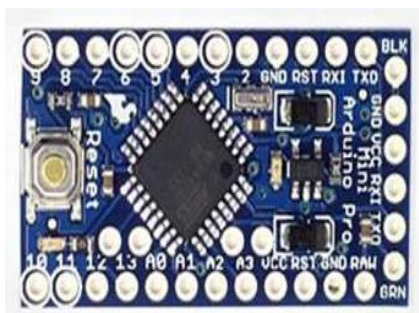
Arduino LEONARDO



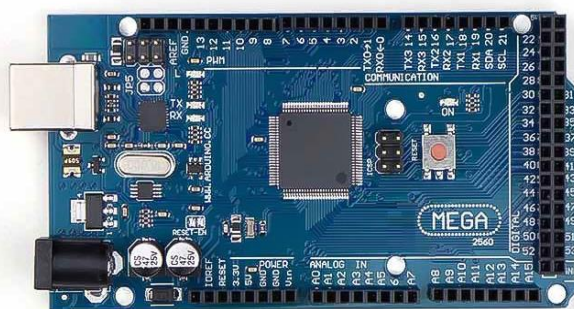
Arduino Uno R3



Arduino Mini



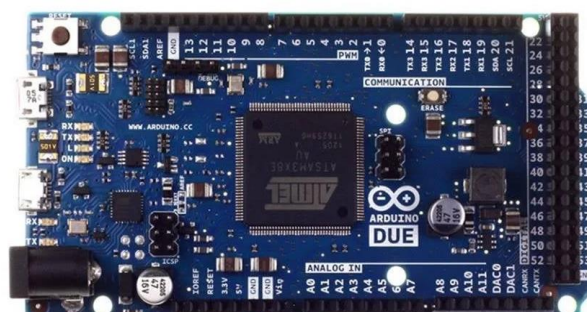
Arduino méga 2560



Arduino ADK Android



Arduino DUE



2.2.3 Qu'est-ce qu'un microcontrôleur ?

- Contrôleur : circuit intégré qui rassemble les éléments essentiels d'un ordinateur : processeur, mémoires (mémoire morte et/ou programmable pour le programme, mémoire vive pour les données), unités périphériques et interfaces d'entrées- sorties
- Ils sont fréquemment utilisés dans les systèmes embarqués, comme les contrôleurs des moteurs automobiles, les télécommandes, les appareils de bureau, l'électroménager, les jouets, la téléphonie mobile, etc.
- Un microcontrôleur intègre sur un unique die (circuit intégré) :
- Un processeur (CPU), avec une largeur du chemin de données allant de 4 bits pour les modèles les plus basiques à 32 ou 64 bits pour les modèles les plus évolués ; de la mémoire vive (RAM) pour stocker les données et variables ;
- de la mémoire pour stocker le programme : ROM (mémoire morte) et/ou EPROM,

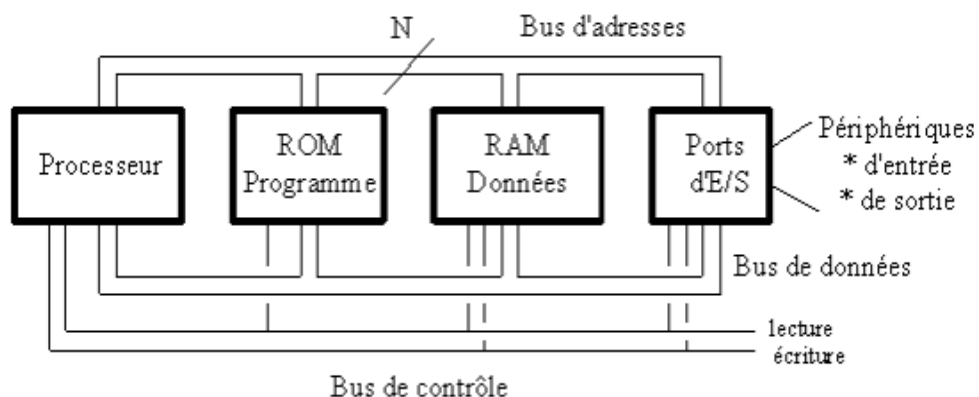


Figure 2.3 : Architecture interne d'un microcontrôleur : Modèle de Von Neumann [28]

EEPROM, Flash ;

- Souvent un oscillateur pour le cadencement. Il peut être réalisé avec un quartz, un circuit RC ou encore une PLL 1

Des périphériques, capables d'effectuer des tâches spécifiques. On peut mentionner entre autres :

- les convertisseurs analogiques-numériques (CAN) (donnent un nombre binaire à partir d'une tension électrique),
- les convertisseurs numériques-analogiques (CNA) (effectuent l'opération inverse),

- les générateurs de signaux à modulation de largeur d'impulsion (MLI, ou en anglais, PWM pour Pulse Width Modulation),
- les times/compteurs (compteurs d'impulsions d'horloge interne ou d'événements externes),
- les chiens de garde (watchdog),
- les comparateurs (comparent deux tensions électriques),
- les contrôleurs de bus de communication (UART, I2C, SSP, CAN, FlexRay, USB, Ethernet, etc.).[29]

CAN = ADC (Analog-to-Digital Converter) CNA = DAC (Digital-to-Analog Converter)

liaison série = UART (Universal Asynchronous Receiver Transmitter)

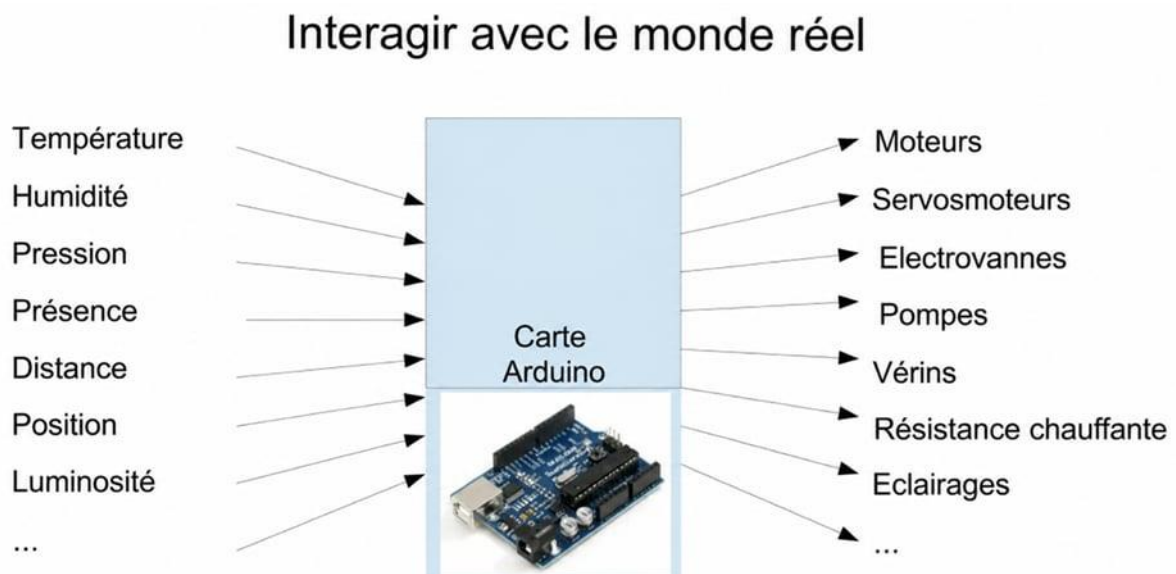


Figure 2.4 : Schéma fonctionnel des entrées et sorties d'un système à microcontrôleur [30]

Exemple : programme « Knock »

- Lecture en continu de la valeur d'un port analogique

- Lorsque cette valeur dépasse un certain seuil, la LED intégrée change d'état et un message est envoyé via le port série

partie déclarative	<pre>1 const int ledPin = 13; 2 const int knockSensor = A0; 3 const int threshold = 100; 4 5 int sensorReading = 0; 6 int ledState = LOW;</pre>
partie initialisation	<pre>7 8 void setup() { 9 pinMode(ledPin, OUTPUT); 10 Serial.begin(9600); 11 }</pre>
boucle sans fin	<pre>12 13 void loop() { 14 sensorReading = analogRead(knockSensor); 15 16 if (sensorReading >= threshold) { 17 ledState = !ledState; 18 digitalWrite(ledPin, ledState); 19 Serial.println("Knock!"); 20 } 21 delay(100); 22 }</pre>

2.3 L'environnement de Développement (IDE) :

2.3.1 Arduino IDE

L'IDE Arduino permet :

- **D'éditer un programme** : un programme est composé de croquis .
- **De compiler ce programme** dans le langage « machine » de l'Arduino, la compilation est une traduction du langage C vers le langage du microcontrôleur
 - la zone de sortie (en bas de la fenêtre) donne des informations sur le déroulement de la compilation et affiche les messages d'erreur et diverses notifications.
- **De téléverser** le programme dans la mémoire de l'Arduino, le téléversement (upload) se passe via le port USB de l'ordinateur une fois dans la mémoire de l'Arduino, le logiciel s'appelle un micrologiciel.
 - la zone de sortie (en bas de la fenêtre) donne des informations sur le déroulement du téléversement et affiche les messages d'erreur et diverses notifications.

- **De communiquer** avec la carte Arduino grâce au terminal (ou moniteur série). Pendant le fonctionnement du programme en mémoire sur l'Arduino, il peut communiquer avec l'ordinateur tant que la connexion est active (câble USB)

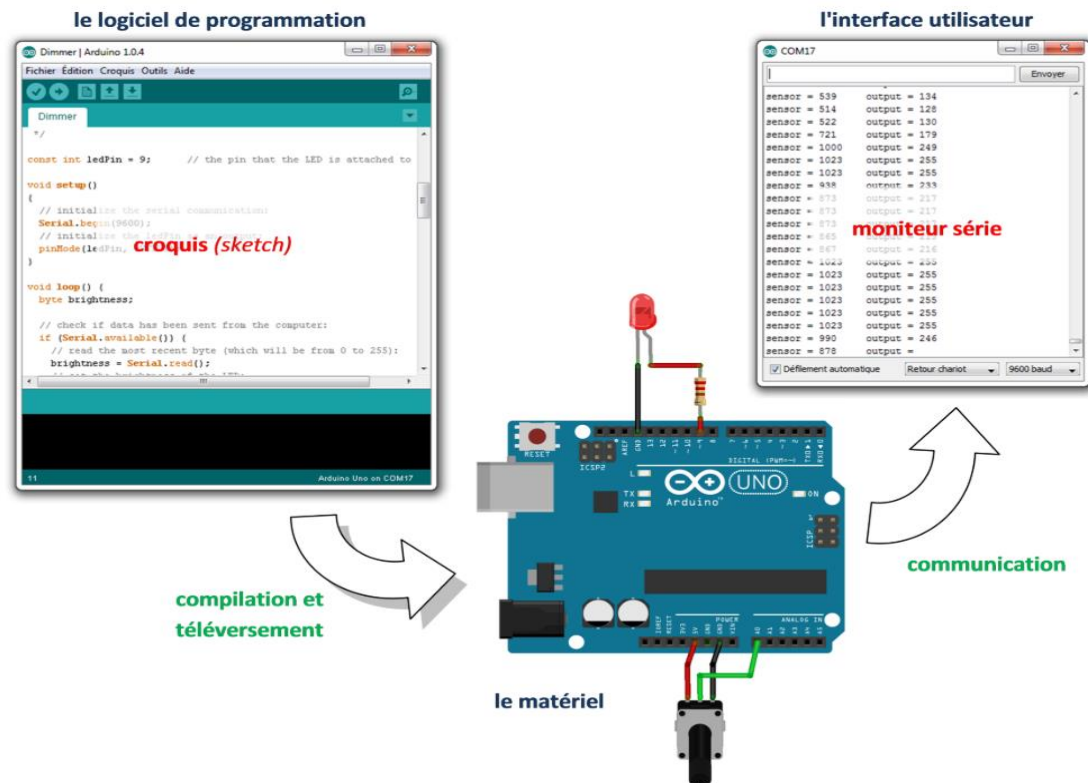


Figure 2.5 : Cycle de développement et environnement de travail Arduino [31]

- Le langage Arduino est basé sur le C/C++. – Le langage de programmation d'Arduino est en effet une implémentation de Wiring (une plate- forme open source similaire de physical computing qui proposait elle-même une bibliothèque appelée Wiring qui simplifie les opérations d'entrée/sortie).
- Un programme Arduino est aussi appelé un sketch

2.3.2 Structure d'un programme

- Prise en compte des instructions de la partie déclarative
- Exécution de la partie configuration (Fonction setup()),
- Exécution de la boucle sans fin (Fonction loop()): le code compris dans la boucle sans fin est exécuté indéfiniment.

2.3.3 Le code minimal

Avec Arduino, nous devons utiliser un code minimal lorsque l'on crée un programme. Ce code permet de diviser le programme que nous allons créer en deux grosses parties.

```
void setup() //fonction d'initialisation de la carte
```

```
{
```

```
    //contenu de l'initialisation
```

```
}
```

```
void loop() //fonction principale qui se répète (s'exécute) à l'infini
```

```
{
```

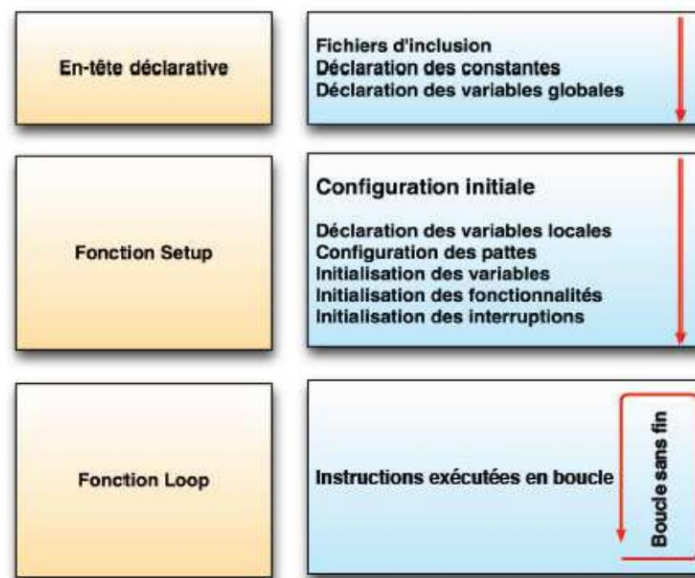


Figure 2. 6 : Déroulement d'un programme [32]

2.3.4 La syntaxe du langage :

Ce schéma illustre l'architecture fondamentale d'un programme informatique où la syntaxe repose sur une structure logique bien définie. Tout commence par l'en-tête déclarative qui utilise des directives comme `#include` et `#define` pour préparer l'environnement avec des variables globales et des constantes. La fonction `setup` prend ensuite le relais pour configurer le système et initialiser les variables locales à l'intérieur de blocs délimités par des accolades `{}`. Enfin, la fonction `loop` instaure une boucle sans fin où les instructions s'exécutent de manière répétitive en s'appuyant sur des structures de contrôle comme `if` ou `for` et des opérateurs arithmétiques pour traiter les données. L'ensemble du code respecte une ponctuation stricte

avec le point-virgule ; et peut être documenté par des commentaires // ou /* */ pour en faciliter la compréhension. [33]

Program structure

```
void setup() { ... } void loop() { ... }
```

Statements

```
;{}  
// /* */  
#define #include
```

Control structures

if, if ... else, for, switch case, while, do ... while
break, continue, return, goto

Control structures

```
type func(type param, ... )
```

Data Types

void, boolean, char, unsigned char, byte,
int, unsigned int, word, long, unsigned long
float, double, string (char[])

Constructor

```
[]
```

Variables

local, global, static local, volatile, const, sizeof()

Operators

Arithmetic = + - * / %
Comparison == != < > <= >=
Boolean && || !
Bitwise & | ^ ~ ""
Pointer Access * &
Compound = -- ++ -= *= /= &|=

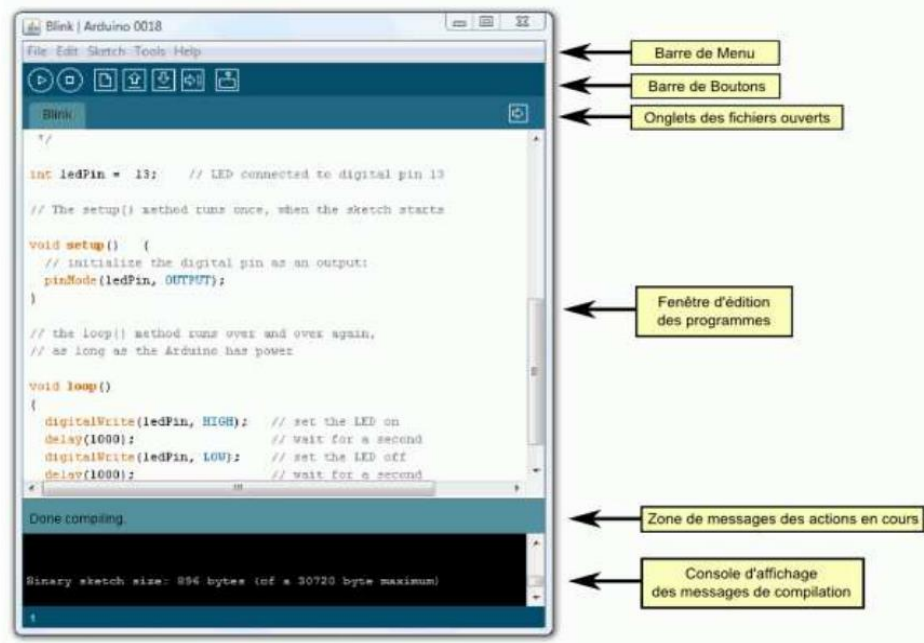


Figure 2.7 : Description des zones fonctionnelles du logiciel de programmation [34]

- **Les boutons :**



1. Vérifier (Verify) : vérifier les erreurs dans le code
2. Charge (Upload) : compiler le code et charge le programme sur la carte Arduino
3. Nouveau (New) : créer un nouveau sketch
4. Ouvrir (Open) : ouvrir un des sketchs déjà présents
5. Sauvegarder (Save) : sauvegarder le sketch
6. Serial Monitor : permet d'accéder au port série (en RX/TX)

- **Le terminal :** Il permet d'interagir avec la platine ARDUINO

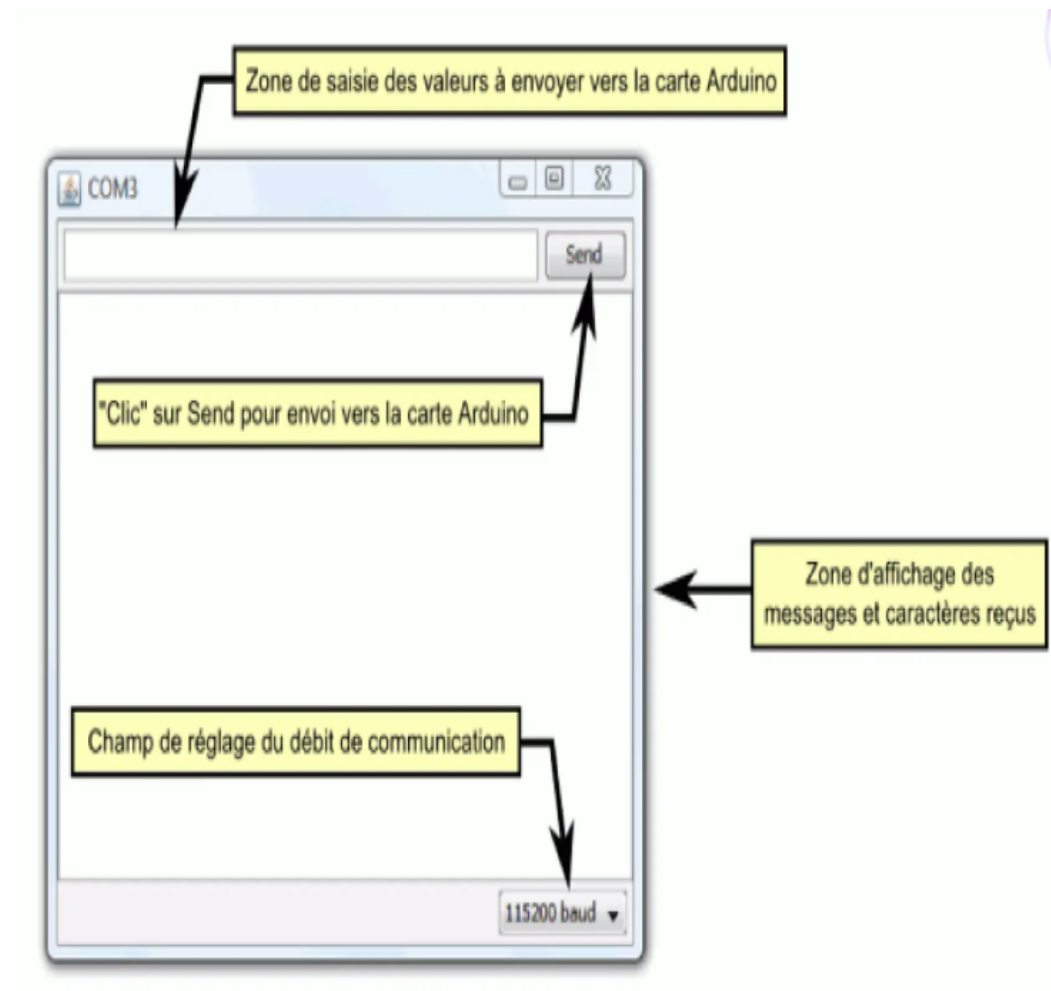


Figure 2.8 : L'interface du Moniteur Série (Serial Monitor) et ses fonctionnalités [34]

Les références :

Structure Fonctions de base Ces fleux fonctions sont obligatoires dans tout utiligramme en langage Arduino : void setup() void loop() Structures de contrôle •if •if, else •for •switch case •while HIGH LOW •break •continue •return Syntaxes de base * (point virgule) {} (accolades) // (commentaire sur une ligne) /* */ (commentaire sur plusieurs lignes) Opérateurs arithmétiques ++ (égalé) += (addition) -= (sustraction) *= (multiplication) /= (division) %= (modulo) Opérateurs de comparaison == (égal à) != (différent de) < (inférieur à) > (supérieur à) <= (inférieur ou égal à) >= (supérieur ou égal à) Opérateurs booléens && (ET booléen) (OU booléen) ! (NON booléen)	Variables et constantes Les variables sont des expressions que vous pouvez utiliser dans les programmes pour stocker des vitlises, telles que la tension de sortie d'un capteur présente sur une broche analogique. Constantes prédéfinies Les constantes prédéfinies du langage Arduino sont des valeurs particulières ayant une signification spécifique. •HIGH LOW •INPUT OUTPUT •true false A ajouter : constantes décimales prédéfinies Expressions numériques •Expressions numériques entières Types des données Les variables peuvent être de type variés qui sont décrits d-dessous. Synthies des types de données Arduino •boolean •byte •byte •int •unsigned int •long •unsigned long •float (nombres à virgules) •double (nombres à virgules) •Les chaines de caractères •Les tableaux de variables •void (functions) Operation des types de données <than •byrefl •initl •longl •floatl Référence	Fonctions Sortées/Sorties Numériques pinMode(broche, mode) digitalWrite(broche, valeur) int digitalWrite(broche) Entrées analogiques int analogRead(broche) Sorties "analogiques" (génération d'impulsion) AnalogWrite(broche, valeur) - PWM Entrées/Sorties Avancées tone() noTone() shiftOut(broche, procherortage, Ordrebit, valeur) unsigned long pulseIn(broche, valeur) Temps unsigned long millis() unsigned long millisl delay(ms) delayMicroseconds(us) Math min(x, y) max(x, y) abs(x) constrain(x, a, b) map(valeur, taieur, fromHigh, toLow, toHigh) pow(base, exposant) sq(x) sqrt(x) Trigonométrie sin(rad) cos(rad) tan(rad) Nombres randomisés (hasard) randomSeed(seed) long random(seed) long random(min, max) Communication Serial
---	---	--

Figure 2.9 : Mémento de la syntaxe et des fonctions du langage Arduino [35]

2.4 Présentation et fonctionnement des modules GPS et GSM

2.4.1 GPS : Global Positioning System.

Un module GPS est un appareil électronique qui reçoit des signaux du système mondial de navigation par satellite (GNSS) pour déterminer l'emplacement géographique précis de l'appareil. En règle générale, un module GPS se compose d'une antenne, d'un récepteur et d'un processeur pour traiter les signaux de plusieurs satellites afin de calculer des informations telles que la latitude, la longitude, l'altitude et la vitesse de l'appareil.



Figure 2.10 : Présentation du module GPS (récepteur et antenne) [36]

2.4.1.1 Composants clés d'un module GPS :

Antenne :

Antenne intégrée ou externe : utilisée pour recevoir les signaux des satellites GPS. Les antennes hautes performances peuvent maintenir une bonne réception du signal dans divers environnements.

Suppression des trajets multiples : certains modules haut de gamme sont équipés d'une technologie permettant de réduire les interférences par trajets multiples et d'améliorer la précision du positionnement.

Récepteur :

Amplification du signal : le récepteur est chargé d'amplifier les signaux satellites faibles pour un traitement ultérieur.

Filtre : utilisé pour éliminer les signaux d'interférence inutiles et garantir la clarté du signal.

Unité de traitement du signal

Processeur GPS : calcule la longitude, la latitude et l'altitude de l'emplacement actuel en décodant les signaux satellite.

Synchronisation de l'horloge : utilisez les signaux satellite pour calibrer l'horloge à l'intérieur du module afin de garantir l'exactitude des informations temporelles.

Interface

Interface de communication : les interfaces communes incluent UART, I2C, SPI, etc., qui sont utilisées pour communiquer avec le dispositif de contrôle principal.

Interface d'alimentation : fournit la tension de fonctionnement nécessaire pour le module.
[37]

2.4.1.2 Composition du GPS :

Composé de 3 parties :

1. Le segment de contrôle :

Formé de 6 stations de contrôle appartenant aux forces armées américaines de l'air (USAF), réparties tout autour du globe en fonction de la longitude.

- Le but de ces stations est de contrôler la santé du segment spatial et de maintenir le temps du système, le GPS-time (GPST).

De manière plus précise, ces stations permettent de :

- Contrôler l'état de santé des satellites ainsi que leur trajectoire,
- Prédire les éphémérides des satellites et les paramètres de l'horloge,
- Mettre à jour les messages de navigation des satellites,
- Commander de petites manœuvres afin de réinitialiser une orbite.

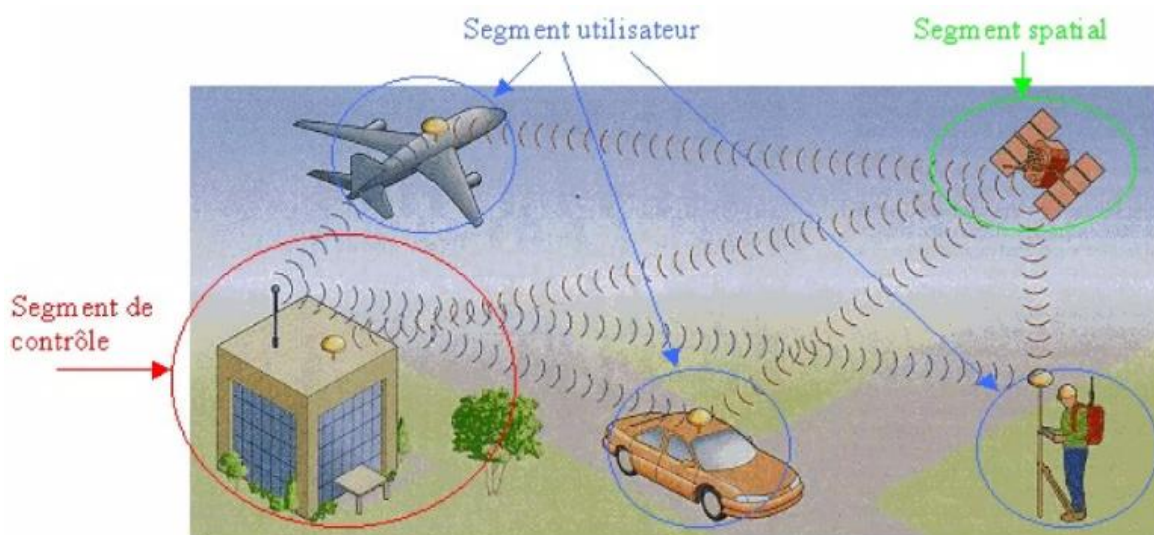


Figure 2.11 : Principe de fonctionnement et communication du réseau GPS

[38]

2. Le segment spatial :

- Le système GPS est formé de 31 satellites (situation au 27 août 2009) en orbite quasi circulaire (excentricité $<0,01$) à une altitude de 20'200 km.
- Leur période est de 11h58 minutes, soit un demi jour sidéral.
- Ces satellites sont répartis sur 6 plans orbitaux inclinés 55° par rapport au plan équatorial.
- Chaque satellite contient plusieurs horloges atomiques, certains ont 4 horloges (2 au rubidium et 2 au césium), certains en ont 3 en césium.
- Ce sont les stations au sol qui sélectionnent l'horloge la plus précise. En effet, ces horloges perdent ou gagnent moins d'une nanoseconde par jour !

3. Le segment utilisateur :

- Rassemble l'ensemble des utilisateurs du système, du simple utilisateur aux géomètres et aux militaires.
- L'ensemble de ces utilisateurs peut être séparé en deux catégories, selon la prestation du système qu'ils utilisent.
- Le système GPS offre deux types de services :
 - SPS (Standard Positioning Service) : service offert à tous les utilisateurs, mais qui peut malheureusement être dégradé volontairement, à l'exemple de la SA.

2.4.1.3 Principe de fonctionnement du GPS :

- Le GPS calcule la position par triangulation (Trilatération) :
- Le satellite émet une onde électromagnétique de vitesse connue
- Le récepteur calcule le temps mis par cette onde pour l'atteindre
- Le récepteur sait alors qu'il se trouve sur une sphère centrée sur le satellite. [39]

2.4.1.2 Comparaison du module GPS avec d'autres modules sans fil :

Tableau 2.2 : Tableau comparatif des technologies de communication sans fil pour systèmes embarqués :

Caractéristiques	Module GPS	Module Wi-Fi	Module Bluetooth	Module Zigbee
Fonctions principales	Positionnement, mesure de vitesse, synchronisation temporelle	Transmission de données à haut débit, accès Internet, mise en réseau d'appareils	Transmission de données à courte portée, couplage d'appareils	Réseau à faible consommation d'énergie, transmission de données à courte distance
Zone d'application	Équipements de navigation, positionnement IoT	Réseaux domestiques et professionnels, maisons intelligentes, appareils mobiles	Appareils audio, appareils portables, appareils domestiques intelligents	Maison intelligente, automatisation industrielle, surveillance environnementale
Avantages	Positionnement de haute précision, couverture mondiale, mises à jour en temps réel	Transmission à grande vitesse, compatibilité étendue, couverture longue distance	Faible consommation d'énergie, connexion simple, communication directe avec l'appareil	Faible consommation d'énergie, mise en réseau flexible, faible coût
Désavantages	Nécessite des signaux satellite, aucune capacité de communication de données	Consommation d'énergie élevée, prise en charge du point d'accès requise	Distance de transmission limitée, faible bande passante	Faible débit de données, courte distance de transmission
Distance de transmission	Portée mondiale (dépend des signaux satellite)	En intérieur, des dizaines à des centaines de mètres (selon l'environnement et les équipements)	Généralement 10 à 30 mètres	Généralement des dizaines de mètres à des centaines de mètres
Débit de données	Ne convient pas à la transmission de données	Haute vitesse (généralement des centaines de	Vitesse moyenne à faible (généralement 1	Faible vitesse (généralement 250 kbps)

Caractéristiques	Module GPS	Module Wi-Fi	Module Bluetooth	Module Zigbee
	(positionnement uniquement)	Mbps à des kiloMbps)	à 3 Mbps, BLE est inférieur)	
Consommation d'énergie	Moyen (dépend du module et du scénario d'utilisation)	Élevé (surtout lorsqu'il est connecté en continu)	Faible (surtout BLE)	Extrêmement faible (convient aux appareils alimentés par batterie à long terme)
Topologie du réseau	N'est pas applicable	Topologie en étoile ou maillée basée sur des points d'accès	Point à point ou étoile	Topologies flexibles telles que étoile, maillage, point à point, etc.

2.4.2 Présentation du module GSM Sim900 :

2.4.2.1 Description sur GSM Sim 900 :

Le SIM900 est un module sans fil fiable et ultra-compact. C'est un module quadri-bande GSM/GPRS complet de type SMT et conçu avec un processeur à puce unique très puissant intégrant une cour ARM926EJ-S, ce qui vous permet de bénéficier de solutions de petites dimensions et économiques. Doté d'une interface standard de l'industrie, le SIM900 offre des performances GSM/GPRS à 850/ 900/ 1 800/ 1 900 MHz pour la voix, les SMS, le fax et les données dans un petit facteur de forme et avec une faible consommation d'énergie. [40]



Figure 2.12 : Module GSM GPRS SIM900 [41]

Le module Sim 900 nous fournit un moyen de communiquer en utilisant le réseau de téléphonie cellulaire GSM à travers une liaison série, ainsi il nous permet d'atteindre les SMS, MMS, GPRS et audio via UART

2.4.2.2 caractéristiques

- Quadri-Bande 850 / 900/ 1800 / 1900 MHz (devrait fonctionner avec tous les réseaux GSM de tous les pays)
- GPRS multi-slot class 10/8
- GPRS mobile station class B
- Conform à GSM phase 2/2+
Class 4 (2 W @ 850 / 900 MHz)
Class 1 (1 W @ 1800 / 1900MHz)
- Contrôle à l'aide de commandes AT (Commandes standard : GSM 07.07 & 07.05)
- Messages SMS (Il est ainsi possible d'envoyer de petites quantités de données via le réseau mobile (ASCII ou données hexadécimales)).
- Embarquer un stack TCP/UDP (Permettant d'envoyer des données sur un serveur web. Support RTC).
- Port série sélectionnable.
- Prise pour micro et casque.
- Basse consommation (1.5mA (en mode veille))
- Supporte la norme de température industrielle (-40°C à +85 °C).
- Tension d'entrée 5V-12V DC [42]

2.5 Caractéristique techniques de la carte ESP32

2.5.1 Description ESP32

L'ESP32 est un microcontrôleur de la famille ESP8266 développé par Espressif Systems. Il s'agit d'une version améliorée de l'ESP8266, avec des performances supérieures et des fonctionnalités étendues.

L'ESP32 est basé sur un microcontrôleur à 32 bits de la famille Xtensa de Tensilica et dispose de capacités Wi-Fi dual-bande 2,4 GHz et 5 GHz. Il dispose également de Bluetooth Low Energy (BLE) intégré, ce qui en fait un choix populaire pour les projets IoT et de domotique.

En termes de mémoire, l'ESP32 dispose de 520 Ko de SRAM et de 448 Ko de ROM. Il dispose également de 16 Ko de mémoire cache et de 4 Mo de mémoire flash externe.

L'ESP32 peut être programmé à l'aide de l'environnement de développement intégré Arduino (IDE) ou d'autres langages de programmation tels que C ou Python. Il est disponible sous différentes formes, notamment modules, cartes de développement et puces, ce qui en fait une option polyvalente pour les projets IoT.

L'ESP32 est largement utilisé dans une gamme d'applications, notamment la domotique, l'éclairage intelligent, la surveillance environnementale et les systèmes de contrôle. [43]

2.5.2 Les caractéristiques techniques :

Microcontrôleur : Espressif ESP32 (monté sur puce ESP-WROOM-32)

Microprocesseur : Tensilica Xtensa Dual-Core 32-bit LX6

Fréquence d'horloge : 80 à 240 MHz (réglable)

Mémoire Flash : 4MB

ROM : 448 kB

SRAM : 520 kB

Bluetooth 4.0 (BLE/Bluetooth Smart) and Bluetooth Classic (BT)

802.11b/g/n HT40 Wi-Fi transceiver

Alimentation : 5 à 9V

Entrées/sorties digitales : 48 GPIO pins

Courant par broche E/S: 40 mA

Courant par broche 3.3V: 50 mA

Pilote : CP2102

Dimensions 55 x 28 x 13 mm

2.5.3 La liste des composants de la carte ESP32 :

Le tableau ci-dessous présente les différents composants de la carte ESP32

Tableau 2.3 : les composants de la carte ESP32 avec leurs descriptions :

Composants	Description
Microcontrôleur ESP32	Le cœur de la carte, responsable du traitement des instructions et de la gestion des périphériques.
Mémoire Flash	Utilisée pour le stockage du programme, peut être intégrée directement sur la carte.
Mémoire RAM	Pour le stockage temporaire des données pendant l'exécution du programme.
Module Wi-Fi et Bluetooth	Assure la connectivité sans fil, incluant le Wi-Fi et le Bluetooth.
Convertisseurs analogique-numérique (CAN)	Permet la lecture de signaux analogiques si nécessaire pour l'application.
Communication (UART, I2C, SPI)	Utilisées pour la communication avec d'autres composants.
Ports GPIO	Ports d'entrée/sortie généralistes pour connecter des capteurs, actionneurs, etc.
Alimentation	Circuit d'alimentation fournissant la tension nécessaire à la carte.
Connecteur USB	Utilisé pour la programmation et la communication série avec un ordinateur
Connecteurs d'antenne	Permettent la transmission et la réception des signaux Wi-Fi et Bluetooth.
Composants discrets	Résistances, condensateurs, inductances, etc., nécessaires pour le bon fonctionnement.
Circuits de régulation de tension	Maintiennent une tension stable pour le microcontrôleur et les autres composants.
Horloge	cristal fournissant une référence de temps stable.

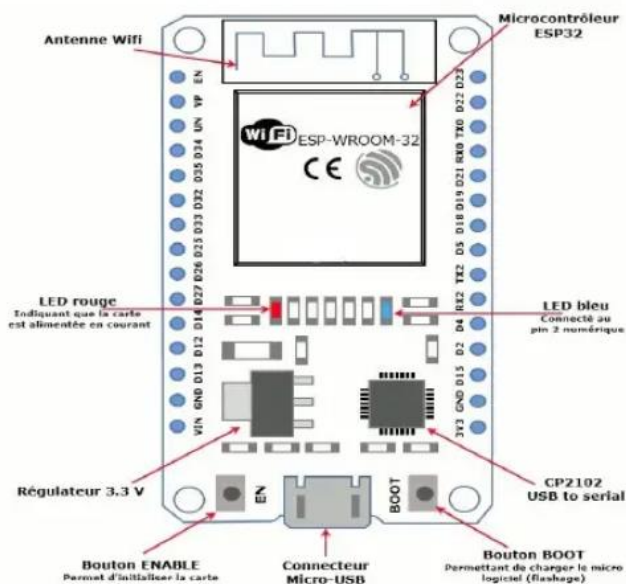


Figure 2.13 : Schéma descriptif des composants de l'ESP32 [44]

EN : est souvent utilisée pour activer ou désactiver le microcontrôleur.

BOOT : peut être utilisée pour sélectionner le mode de démarrage

2.5.4 Brochage ESP32

L'ESP32 est disponible sous différentes formes, notamment modules, cartes de développement et puces. Les broches de l'ESP32 varient en fonction de la forme choisie, mais voici une liste des broches les plus courantes : [45]

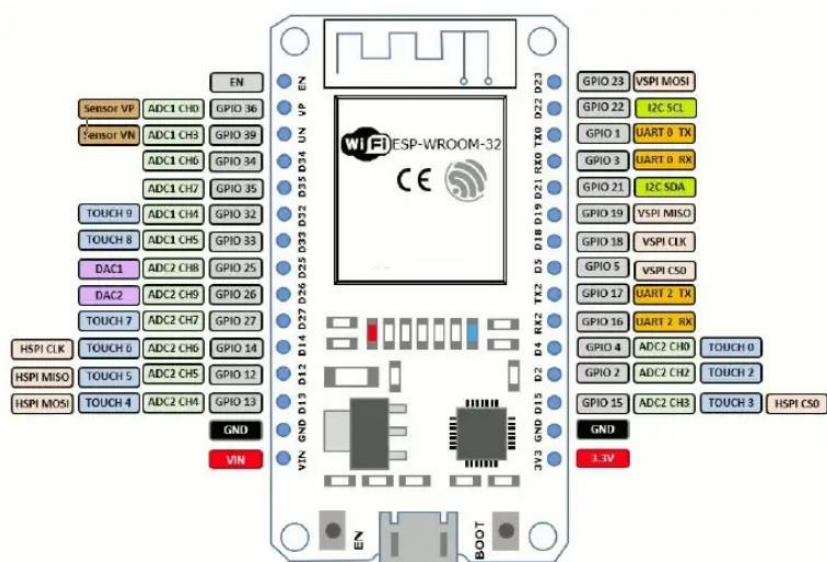


Figure 2.14 : Diagramme de brochage (Pinout) de la carte ESP32 NodeMCU [46]

Tableau 2.4 : Tableau des broches de l'ESP32 et leurs fonctions :

Nom	No.	Type	Fonction
GND	1	P	Ground
3V3	2	P	Power supply 3.3V
EN	3	I	High: On, enables the chip. Low: Off, the chip shuts down
SENSOR_VP	4	I	GPIO36, ADC1_CH0, RTC_GPIO0
SENSOR_VN	5	I	GPIO39, ADC1_CH3, RTC_GPIO3
IO34	6	I	GPIO34, ADC1_CH6, RTC_GPIO4
IO35	7	I	GPIO35, ADC1_CH7, RTC_GPIO5
IO32	8	I/O	GPIO32, XTAL_32K_P, ADC1_CH4
IO33	9	I/O	GPIO33, XTAL_32K_N, ADC1_CH5
IO25	10	I/O	GPIO25, DAC_1, ADC2_CH8
IO26	11	I/O	GPIO26, DAC_2, ADC2_CH9
IO27	12	I/O	GPIO27, ADC2_CH7, TOUCH7
IO14	13	I/O	GPIO14, ADC2_CH6, TOUCH6
IO12	14	I/O	GPIO12, ADC2_CH5, TOUCH5
GND	15	P	Ground
IO13	16	I/O	GPIO13, ADC2_CH4, TOUCH4
SD2	17	I/O	GPIO9, SD_DATA2
SD3	18	I/O	GPIO10, SD_DATA3
CMD	19	I/O	GPIO11, SD_CMD
CLK	20	I/O	GPIO6, SD_CLK
SD0	21	I/O	GPIO7, SD_DATA0
SD1	22	I/O	GPIO8, SD_DATA1
IO15	23	I/O	GPIO15, ADC2_CH3, TOUCH3
IO2	24	I/O	GPIO2, ADC2_CH2, TOUCH2
IO0	25	I/O	GPIO0, ADC2_CH1, TOUCH1
IO4	26	I/O	GPIO4, ADC2_CH0, TOUCH0
IO16	27	I/O	GPIO16, HS1_DATA4, U2RXD
IO17	28	I/O	GPIO17, HS1_DATA5, U2TXD
IO5	29	I/O	GPIO5, VSPI_CS0
IO18	30	I/O	GPIO18, VSPI_CLK

2.6 Présentation et fonctionnement du capteur MPU6050

2.6.1 MPU6050

Le MPU-6050 est un dispositif de suivi de mouvement intégrant un gyroscope et un accéléromètre triaxiaux sur une seule puce. Fabriqué par InvenSense (désormais intégré à TDK), il est couramment utilisé dans diverses applications électroniques pour la mesure du mouvement et de l'orientation.

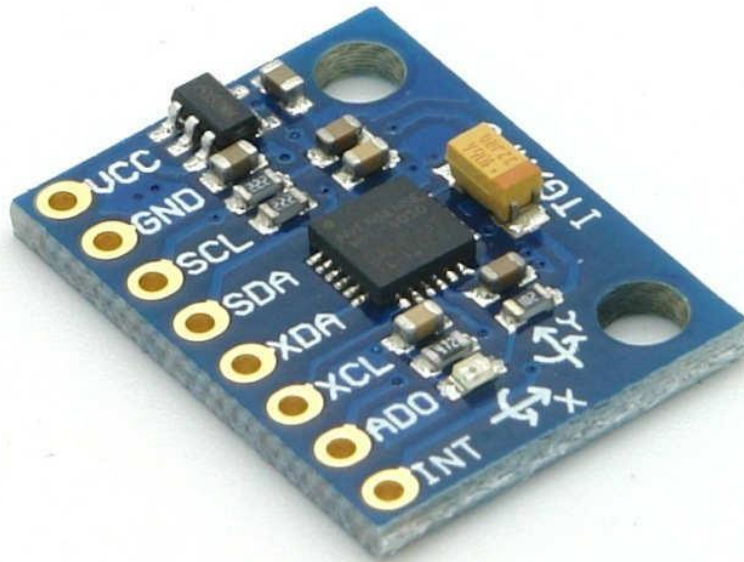


Figure 2.15 : une image qui représente MPU6050 [47]

2.6.2 Caractéristique

Vitesse angulaire : Mesure le mouvement de rotation en degrés par seconde (dps).

Sensibilité : La sensibilité est configurable selon la plage sélectionnée (par exemple, ± 250 , ± 500 , ± 1000 ou ± 2000 dps).

Accéléromètre : Accélération : Mesure l'accélération linéaire sur trois axes (X, Y, Z).

Sensibilité : La sensibilité est configurable selon la plage sélectionnée (par exemple, ± 2 g, ± 4 g, ± 8 g ou ± 16 g).

Unité de traitement de mouvement (MPU) : La MPU-6050 intègre un processeur de mouvement numérique (DMP) qui décharge le microcontrôleur principal des tâches

complexes de traitement de mouvement.

Le DMP fournit des données quaterniennes pour l'estimation d'orientation.

Interface de communication : La MPU-6050 communique généralement avec un microcontrôleur via le protocole I2C (Inter-Integrated Circuit). Certains modules peuvent également inclure des options de communication supplémentaires, telles que le SPI.

Capteur de température : La MPU-6050 est dotée d'un capteur de température intégré qui mesure la température de l'appareil.

Consommation d'énergie : La MPU-6050 est conçue pour une faible consommation d'énergie, avec des modes basse consommation.

Interruptions : La MPU-6050 peut générer des interruptions en fonction de certaines conditions, comme la détection de mouvement, la disponibilité des données ou le débordement de la FIFO.

Filtre passe-bas numérique (DLPF) : Le DLPF peut être configuré pour filtrer le bruit haute fréquence des données du capteur.[48].

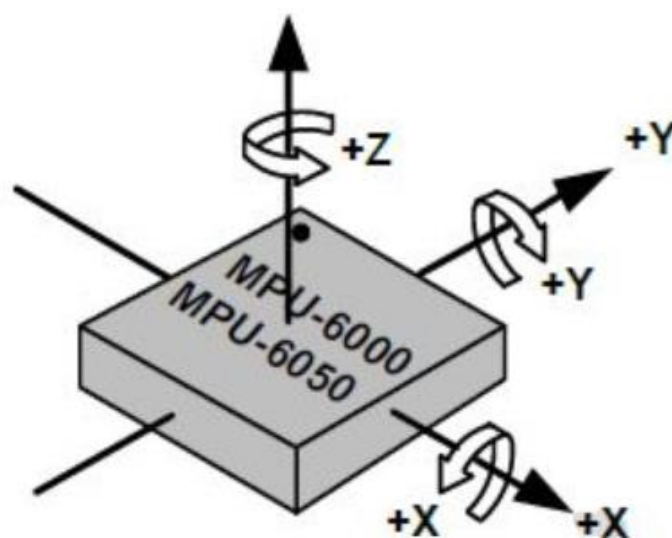


Figure 2.16 : Orientation des axes de sensibilité et polarité de rotation [49]

2.6.3 Applications :

- Centrales inertielles (IMU) pour le suivi d'orientation.
- Robotique et stabilisation de s.
- Manettes de jeu et applications de réalité virtuelle.
- Dispositifs d'entrée basés sur le mouvement.

2.7 Description et fonctionnement du module sim800L

2.7.1 Module sim800L

Le module SIM800L est un petit module GSM/GPRS mesurant 2.2 cm x 1.8 cm. Il s'agit d'un module robuste qui démarre automatiquement et recherche également automatiquement le réseau. Ce module supporte le réseau GSM/GPRS quadri-bande, permettant la transmission à distance de données via des messages SMS et GPRS. Le SIM800L communique avec le microcontrôleur via le port UART et prend en charge les commandes telles que 3GPP TS 27.007, 27.005 ainsi que les commandes AT améliorées SIMCOM. De plus, il intègre la technologie A-GPS pour le positionnement mobile, permettant de déterminer la position par réseau mobile. Cette fonctionnalité en fait également un module adapté au suivi d'objets. [50]

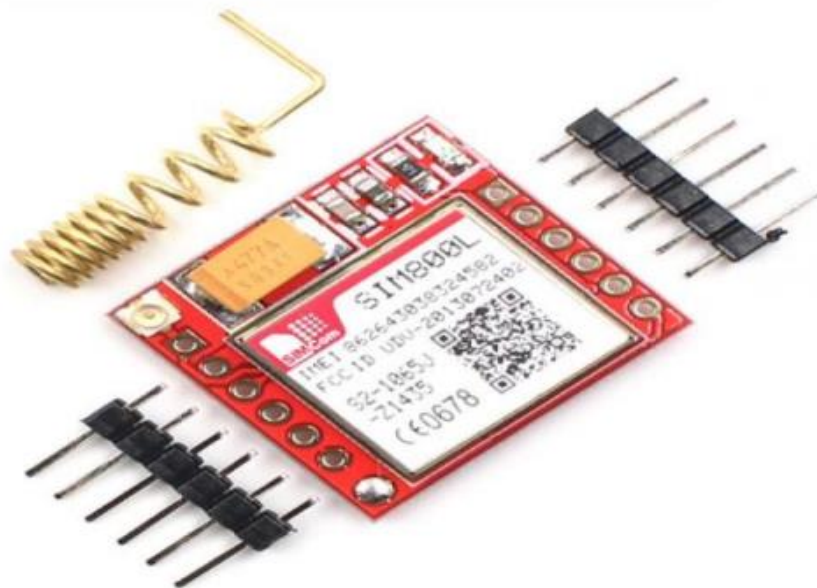


Figure 2.17 : Module GSM SIM800L [51]

Ce module requiert une alimentation comprise entre 3,4V et 4,4V, ce qui n'est pas compatible avec l'alimentation 5V fournie par l'Arduino. Pour résoudre cette contrainte d'alimentation, une diode 1N4007 est ajoutée entre le pin 5V de l'Arduino et le pin VCC du SIM800L. Le SIM800L demande un pic de courant d'environ 2A pour son fonctionnement optimal.

2.7.2 Fonctionnalités :

- 2G quadri-bande 850/900/1800 / 1900MHz.
- Recevoir et passer des appels à l'aide des sorties haut-parleur et microphone.
- Recevoir et envoyer des SMS.
- Écouter les émissions de radio FM
- Station mobile GPRS classe B.
- Contrôlé par AT Command (3GPP TS 27.007, 27.005 et SIMCOM Enhanced AT Commands)
- Prend en charge l'horloge en temps réel
- Plage de tension de fonctionnement 3,4 V ~ 4,4 V
- Prend en charge A-GPS
- Faible consommation d'énergie, 1mA en mode veille
- Carte micro SIM

2.7.3 LED clignotant :

Lorsque l'alimentation du SIM800L est adéquate, sa LED intégrée commence à clignoter pour refléter son état opérationnel. En cas d'alimentation insuffisante, le module peut recevoir un signal de qualité médiocre et pourrait persister dans la recherche de réseau de manière continue. Pour ajuster cette recherche persistante, il est recommandé d'ajuster légèrement le potentiomètre, sans dépasser 4,7 V. La fréquence du clignotement de la LED indique alors ce qui suit :

- Chaque seconde: recherche d'un réseau.
- Toutes les trois secondes: connectée à un réseau.
- Deux fois par seconde: connecté via GPRS.

2.7.4 Détails de la broche :

À l'arrière du module (figure I.10.), on peut trouver les détails de la broche imprimés dessus. Nous utiliserons 5 broches pour nous connecter à Arduino pour un fonctionnement de base. Ces broches sont :

NET - Broche d'antenne pour le module (type à ressort inclus). Vous pouvez utiliser une antenne externe via un connecteur uFL disponible sur le module pour une meilleure réception du signal. [52]

VCC - Alimentation 3,4V à 4,4V avec min 2 Amp. (LM2596 avec une entrée adaptateur 9 / 12V 1Amp et il doit être ajusté à 4.2V recommandé).

RST - Réinitialiser

RXD - Récepteur vers le module - Logique 3.3V (À connecter à la broche Arduino TX. Diviseur de tension recommandé).

TXD - Émetteur du module (à connecter à la broche Arduino RX).

GND - Terre.

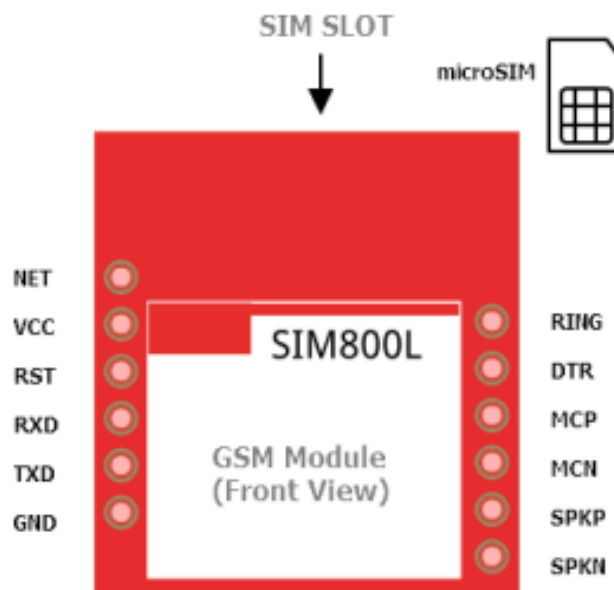


Figure 2.18 : Brochage du module GSM SIM800L [53]

2.8 Étude Approfondie des Protocoles de Communication (Bus I2C, UART, SPI)

Dans un système embarqué de sécurité routière, la fiabilité du transfert de données entre le microcontrôleur (Arduino Nano) et les périphériques est critique. Trois protocoles de série sont principalement exploités.

2.8.1 Le protocole UART (Universal Asynchronous Receiver-Transmitter)

L'UART est un protocole de communication asynchrone, ce qui signifie qu'il n'utilise pas de signal d'horloge commun. La synchronisation se fait par une configuration identique de la vitesse (Baud rate) sur les deux appareils.

- Fonctionnement technique : Les données sont envoyées bit par bit sous forme de trames comprenant un bit de départ (start bit), 8 bits de données, et un bit d'arrêt (stop bit).[54]
- Application au projet :
 - Module GPS : L'Arduino reçoit en continu des trames au format NMEA via la broche RX. Ces trames contiennent la latitude, la longitude et l'altitude.
 - Module GSM : L'Arduino communique via des Commandes AT (Attention Commands) pour demander au module d'envoyer un SMS ou d'établir une connexion GPRS.
- Contrainte : L'Arduino Nano ne possédant qu'un seul port UART matériel (utilisé pour le port USB), nous utilisons la bibliothèque <SoftwareSerial.h> pour créer des ports série virtuels sur d'autres broches numériques.

2.8.2 Le Bus I2C (Inter-Integrated Circuit)

Le bus I2C est une communication synchrone à deux fils, développée par Philips. Il repose sur une architecture Maître/Esclave.

Lignes de communication :

- ❖ SDA (Serial Data Line) : Ligne bidirectionnelle pour le transfert des données.
- ❖ SCL (Serial Clock Line) : Ligne d'horloge générée par l'Arduino pour cadencer les échanges.

- **Mécanisme d'adressage** : Chaque composant possède une adresse matérielle unique (7 bits). Cela permet à l'Arduino de discuter avec un capteur précis sans interférer avec les autres.
- **Application au projet** : Si un accéléromètre numérique, comme le MPU6050, est utilisé pour détecter les chocs brutaux, le protocole I²C est nécessaire pour transmettre les mesures d'accélération des axes X, Y et Z au microcontrôleur avec une bonne précision.[55]

2.8.3 L'Interface SPI (Serial Peripheral Interface)

Le protocole SPI est le plus rapide. Contrairement à l'I²C, il utilise quatre fils et fonctionne en mode Full-Duplex (émission et réception simultanées).

- **Signaux principaux** :
 - ❖ SCK (Serial Clock) : Horloge de synchronisation.
 - ❖ MOSI (Master Out Slave In) : Données envoyées par l'Arduino vers le module.
 - ❖ MISO (Master In Slave Out) : Données reçues par l'Arduino depuis le module.
 - ❖ SS/CS (Slave Select) : Ligne permettant d'activer le module avec lequel on souhaite communiquer.
- **Application au projet** : Ce protocole est privilégié si le système doit enregistrer des logs (historique des accidents) sur une carte micro-SD. La vitesse du SPI permet d'écrire de gros volumes de données sans ralentir le programme principal.

2.6 Modèle d'intelligence artificielle IA

2.6.1 Introduction à l'Intelligence Artificielle :

L'intelligence artificielle (IA) désigne un ensemble de techniques permettant à une machine d'imiter certaines capacités humaines, telles que l'apprentissage, la prise de décision et la reconnaissance de patterns. Dans le domaine de la sécurité routière, l'IA est utilisée pour analyser les données provenant des capteurs afin de détecter des situations anormales comme les accidents.

Architecture du Modèle :

Le Machine Learning Pour un système embarqué de type Arduino, on privilégie généralement des modèles de Machine Learning supervisé. Le processus suit un cycle précis :

- 1-Collecte de données : Récupération des signaux des capteurs (accélération a_x , a_y , a_z et inclinaison via gyroscope).
- 2-Prétraitement : Nettoyage du bruit et normalisation des valeurs.
- 3-Extraction de caractéristiques (Features) : Calcul de la magnitude du vecteur accélération : $A_{total} = \sqrt{a_x^2 + a_y^2 + a_z^2}$
- 4-Classification : Le modèle décide si les valeurs correspondent à un état "Normal" ou "Accident".

Objectif de l'IA dans le système proposé :

Dans notre projet, l'intégration d'un modèle d'IA vise à :

- Détection automatique d'un accident de véhicule
- Réduire et minimiser le risque de fausses alertes pour garantir la fiabilité du système.
- Améliorer la précision du système
- Déclencher l'envoi immédiat des coordonnées GPS via le réseau GSM sans intervention humaine.

Types de Modèles d'IA Utilisés :

Il existe 3 types de modèles d'IA

a) Modèle basé sur des règles (Rule-Based System) :

C'est le modèle le plus utilisé par les électriciens simple et adapté aux systèmes embarqués comme Arduino. Parmi ses avantages simple à implémenter, faible consommation, mémoire rapide.

Il fonctionne d'une manière simple Si vibration forte + arrêt brusque → accident probable

Exemple : Si (vibration > seuil) ET (vitesse = 0) Alors accident = vrai

b) Modèle d'Apprentissage Supervisé (Machine Learning) :

Ce modèle est plus précis, capable d'apprendre. Il utilise des données d'apprentissage pour prédire les accidents.

Algorithmes possibles :

Régression logistique Arbre de décision K-Nearest Neighbors (KNN)

Le modèle est entraîné avec des données :

Accidents réels

Situations normales

Puis il apprend à différencier les deux.

c) Réseaux de Neurones Artificiels (ANN) :

Inspirés du cerveau humain, très précis et adaptatif ils permettent une analyse plus complexe.

Mais consomme plus de ressource et très complexe pour microcontrôleur

Utilisation :

Détection intelligente d'accidents Analyse multi-capteurs (vibration, vitesse, position)

Choix du Modèle pour le Projet

C'est pour cela qu'on a choisi *Système basé sur des règles* pour notre projet pour plusieurs raisons :

- ✓ Compatible avec Arduino (ATmega328)
- ✓ Faible mémoire et puissance de calcul
- ✓ Facile à programmer et tester
- ✓ Suffisant pour une première implémentation

Choix de l'Algorithme

2.6.3 Déploiement : L'Approche "TinyML"

Puisque vous utilisez une carte Arduino (ATmega328), la mémoire est limitée (32 KB de Flash).

L'intégration de l'IA se fait via le concept de TinyML : Le modèle est entraîné sur un ordinateur puissant (Python/TensorFlow).

Il est ensuite compressé et converti en langage C++ pour être injecté dans l'Arduino IDE.

Cela permet au système de prendre des décisions localement (Edge Computing) sans dépendre d'une connexion internet, garantissant une réactivité immédiate.

2.6.4 Entraînement et Validation Le modèle doit être entraîné sur une base de données contenant :

Cas négatifs : Dos-d'âne, freinages d'urgence, vibrations de la route.

Cas positifs : Impacts frontaux, latéraux et tonneaux.

La performance est ensuite mesurée par une matrice de confusion pour minimiser les "faux positifs" (envoyer les secours alors qu'il n'y a pas d'accident).

Améliorations Futures

Pour améliorer le système, on peut :

Ajouter un modèle Machine Learning via un serveur externe Utiliser des cartes plus puissantes (ESP32, Raspberry Pi) Intégrer un modèle hybride (règles + apprentissage)

Conclusion :

L'intelligence artificielle permet d'améliorer considérablement la détection des accidents. Cependant, le choix du modèle dépend fortement des contraintes matérielles. Dans notre projet, un modèle simple basé sur des règles est adopté, offrant un bon compromis entre performance et faisabilité.

Chapitre III

Simulation et réalisation du système

3.1 Présentation de l'Environnement de Simulation Proteus ISIS :

Le choix de l'outil de simulation constitue une étape déterminante pour la validation fonctionnelle d'un système embarqué. Dans le cadre de notre projet, nous avons opté pour le logiciel Proteus VSM (Virtual System Modelling), développé par Labcenter Electronics. Cet environnement est particulièrement adapté aux projets intégrant des microcontrôleurs, car il permet une simulation hybride (analogique et numérique) en temps réel.

3.1.1 Justification du choix de l'outil :

L'utilisation de Proteus ISIS repose sur trois avantages majeurs :

- **Prototypage virtuel complet** : Il permet d'interconnecter la carte Arduino Nano avec des capteurs physiques (tel que le capteur de vibration SW-420) et des modules de communication (GPS, GSM SIM900D) sans risque de détérioration matérielle.
- **Outils d'analyse avancés** : Le logiciel met à disposition des instruments de mesure virtuels indispensables au diagnostic, notamment l'oscilloscope numérique, le terminal série (Virtual Terminal) pour le suivi des commandes AT, et les sondes de tension.
- **Validation du comportement logique** : La possibilité de simuler le code source (.hex) directement sur le microcontrôleur virtuel permet de valider le comportement du système dans des conditions proches du réel.

3.1.2 Configuration de l'environnement :

La configuration de la simulation s'est articulée autour de plusieurs étapes :

1. **Intégration des composants** : Importation des bibliothèques spécifiques nécessaires à la modélisation des modules de communication et du capteur de vibration.
2. **Paramétrage de la simulation** : Ajustement des options d'animation (CPU Load Limit) afin de garantir une exécution stable des modèles complexes comme le module SIM900D.

3.Configuration de l'affichage des données : Utilisation du terminal virtuel (Virtual Terminal) pour monitorer en temps réel les coordonnées GPS reçues et valider la bonne transmission des données du système.

Cet environnement de simulation constitue ainsi la plateforme de test primaire, garantissant la fiabilité du comportement global du système avant la phase de déploiement physique.

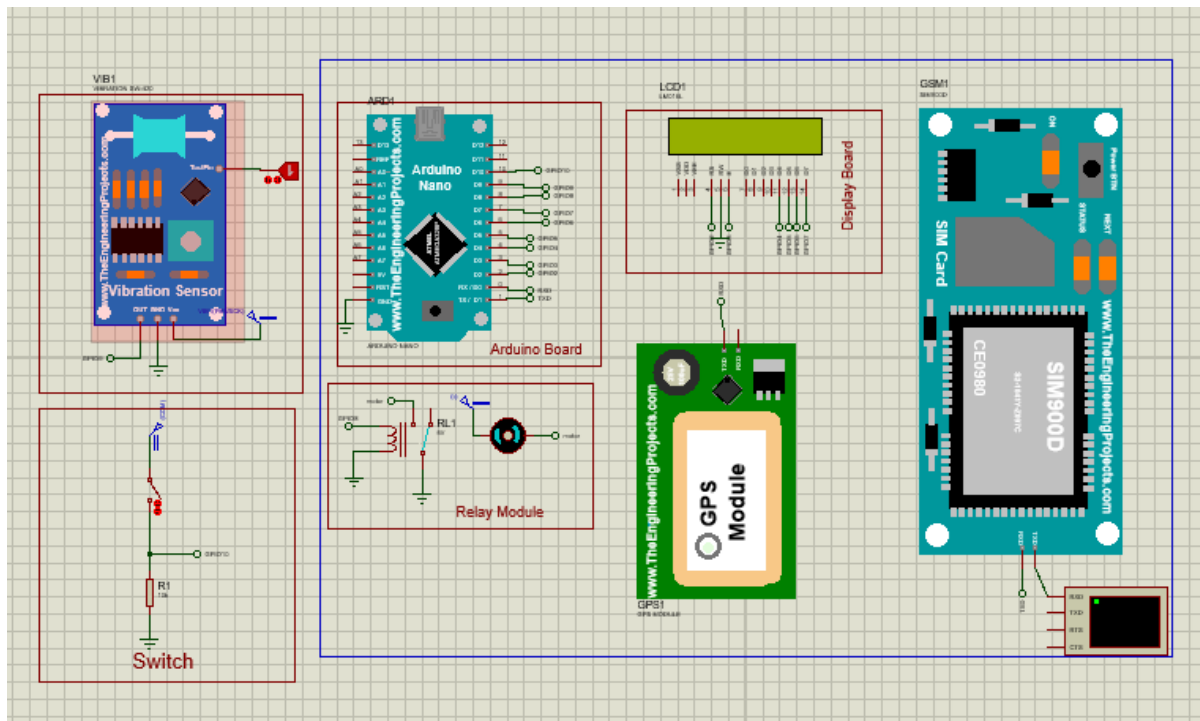


Figure 3.1: schéma global du système sous Proteus ISIS

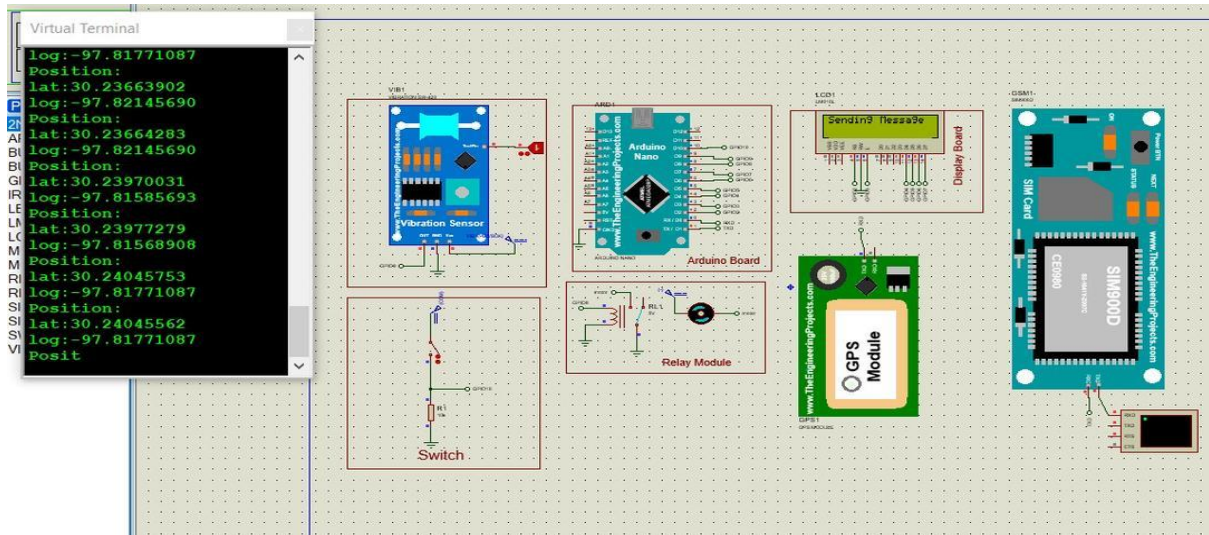
Ce diagramme illustre l'architecture électronique du système élaboré avec Proteus ISIS. Il représente la liaison entre l'unité de traitement et les dispositifs de détection et de communication :

Gestion : Un Arduino Nano coordonne la gestion des entrées et sorties.

Acquisition : L'événement est détecté par un capteur de vibrations et un interrupteur.

Localisation et Communication : L'implémentation d'un module GPS associé à un module GSM (SIM900D) offre la possibilité de géolocalisation ainsi que l'émission d'alertes à distance.

Interface et Action : Un terminal virtuel présente l'état en direct du système, pendant qu'un relais autorise la gestion d'un circuit de puissance (alarme ou coupure).



Ensuite, ce montage a été réalisé en pratique, d'abord à l'aide d'une carte Arduino Uno, puis avec un ESP32 relié au capteur MPU6050. Ce dernier a été calibré afin de fournir des valeurs précises d'accélération et de vitesse de rotation, comme l'illustrent les figures ci-dessous.

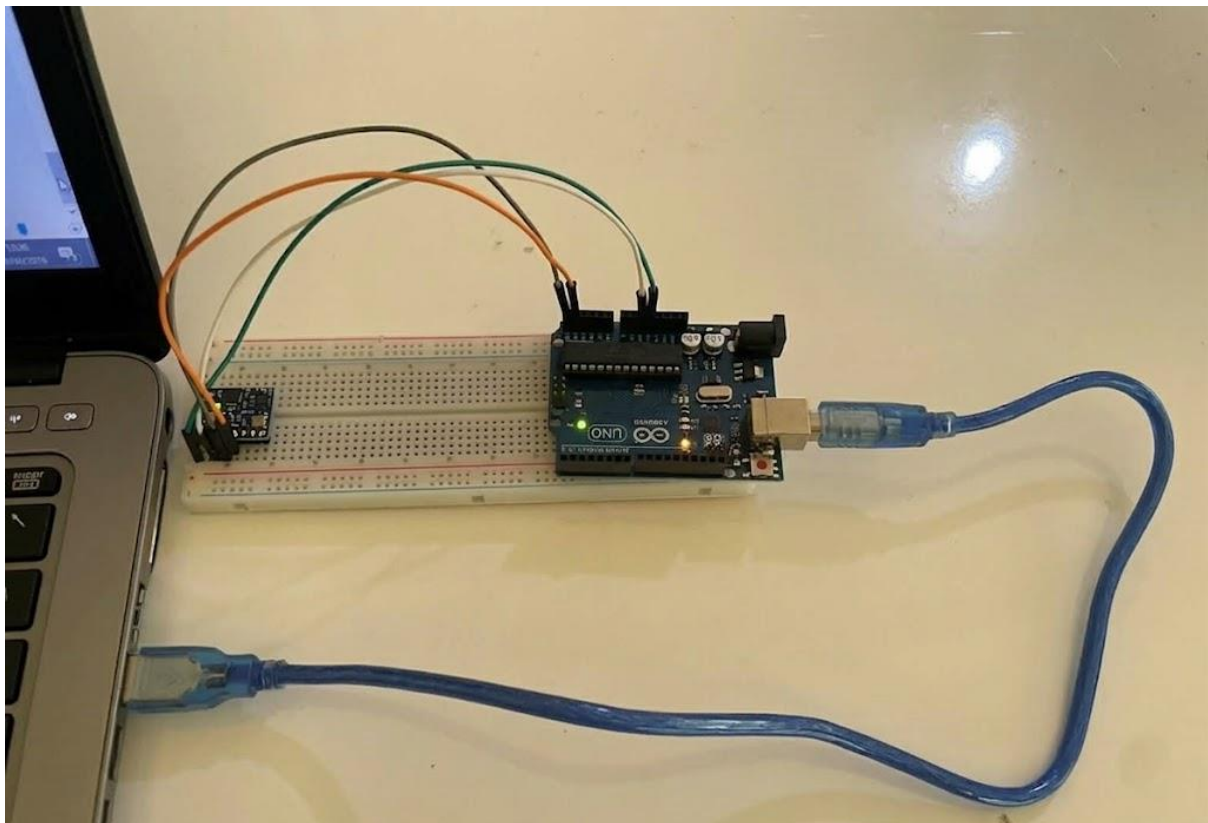


Figure 3.2 : Montage du système Arduino Uno–MPU6050

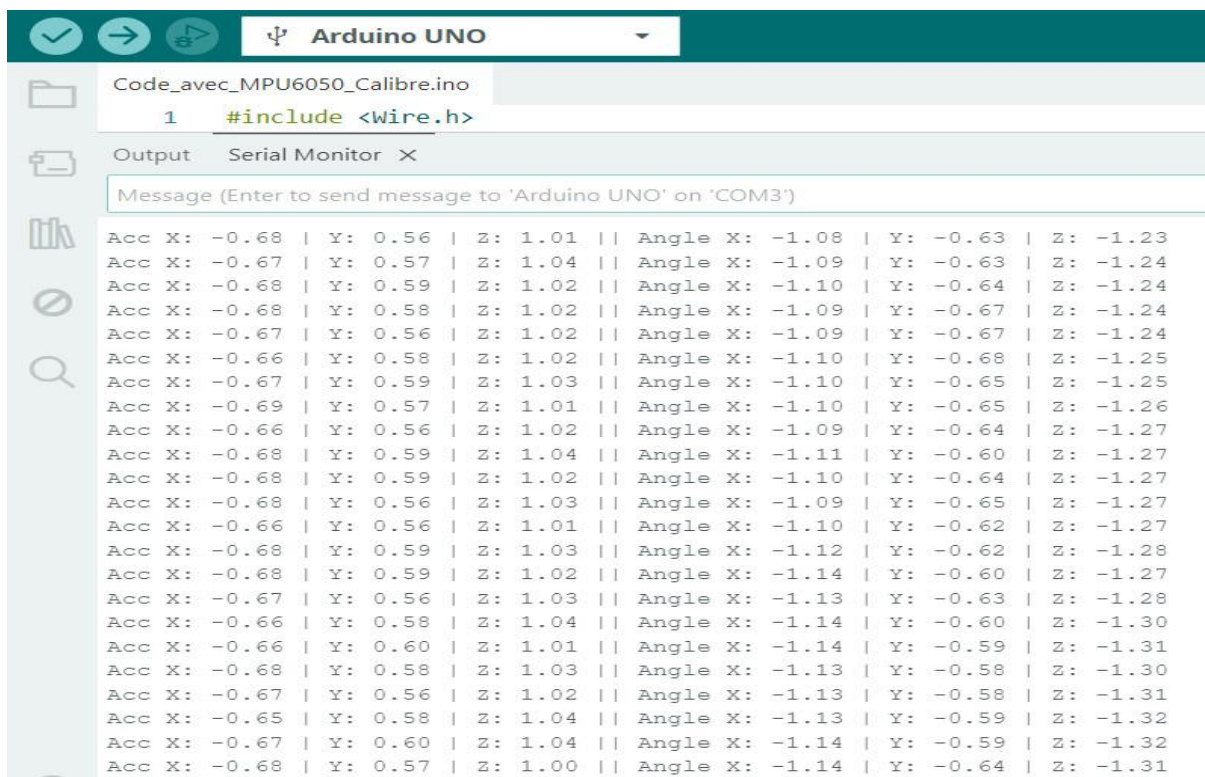
L'Arduino Uno a été connecté au capteur MPU6050 via l'interface I2C en reliant les broches d'alimentation (VCC et GND) ainsi que les lignes de communication (SDA et SCL).

Branchement du MPU6050 (Accéléromètre, Gyroscope)

- La broche VCC du MPU6050 à la broche 5V de l'Arduino.
- La broche GND du MPU6050 à la broche GND de l'Arduino.
- La broche SCL du MPU6050 à la broche A5 de l'Arduino.
- La broche SDA du MPU6050 à la broche A4 de l'Arduino.

Les valeurs du capteur calibré ont été affichées sur le moniteur série, comme le montre la figure ci-dessous.

Les données du capteur calibré



```

Code_avec_MPU6050_Calibre.ino
1 #include <Wire.h>

Output Serial Monitor X
Message (Enter to send message to 'Arduino UNO' on 'COM3')

Acc X: -0.68 | Y: 0.56 | Z: 1.01 || Angle X: -1.08 | Y: -0.63 | Z: -1.23
Acc X: -0.67 | Y: 0.57 | Z: 1.04 || Angle X: -1.09 | Y: -0.63 | Z: -1.24
Acc X: -0.68 | Y: 0.59 | Z: 1.02 || Angle X: -1.10 | Y: -0.64 | Z: -1.24
Acc X: -0.68 | Y: 0.58 | Z: 1.02 || Angle X: -1.09 | Y: -0.67 | Z: -1.24
Acc X: -0.67 | Y: 0.56 | Z: 1.02 || Angle X: -1.09 | Y: -0.67 | Z: -1.24
Acc X: -0.66 | Y: 0.58 | Z: 1.02 || Angle X: -1.10 | Y: -0.68 | Z: -1.25
Acc X: -0.67 | Y: 0.59 | Z: 1.03 || Angle X: -1.10 | Y: -0.65 | Z: -1.25
Acc X: -0.69 | Y: 0.57 | Z: 1.01 || Angle X: -1.10 | Y: -0.65 | Z: -1.26
Acc X: -0.66 | Y: 0.56 | Z: 1.02 || Angle X: -1.09 | Y: -0.64 | Z: -1.27
Acc X: -0.68 | Y: 0.59 | Z: 1.04 || Angle X: -1.11 | Y: -0.60 | Z: -1.27
Acc X: -0.68 | Y: 0.59 | Z: 1.02 || Angle X: -1.10 | Y: -0.64 | Z: -1.27
Acc X: -0.68 | Y: 0.56 | Z: 1.03 || Angle X: -1.09 | Y: -0.65 | Z: -1.27
Acc X: -0.66 | Y: 0.56 | Z: 1.01 || Angle X: -1.10 | Y: -0.62 | Z: -1.27
Acc X: -0.68 | Y: 0.59 | Z: 1.03 || Angle X: -1.12 | Y: -0.62 | Z: -1.28
Acc X: -0.68 | Y: 0.59 | Z: 1.02 || Angle X: -1.14 | Y: -0.60 | Z: -1.27
Acc X: -0.67 | Y: 0.56 | Z: 1.03 || Angle X: -1.13 | Y: -0.63 | Z: -1.28
Acc X: -0.66 | Y: 0.58 | Z: 1.04 || Angle X: -1.14 | Y: -0.60 | Z: -1.30
Acc X: -0.66 | Y: 0.60 | Z: 1.01 || Angle X: -1.14 | Y: -0.59 | Z: -1.31
Acc X: -0.68 | Y: 0.58 | Z: 1.03 || Angle X: -1.13 | Y: -0.58 | Z: -1.30
Acc X: -0.67 | Y: 0.56 | Z: 1.02 || Angle X: -1.13 | Y: -0.58 | Z: -1.31
Acc X: -0.65 | Y: 0.58 | Z: 1.04 || Angle X: -1.13 | Y: -0.59 | Z: -1.32
Acc X: -0.67 | Y: 0.60 | Z: 1.04 || Angle X: -1.14 | Y: -0.59 | Z: -1.32
Acc X: -0.68 | Y: 0.57 | Z: 1.00 || Angle X: -1.14 | Y: -0.64 | Z: -1.31
  
```

Les valeurs d'accélération ont ensuite été affichées en temps réel sur le moniteur série.

Les Valeurs d'accélération en temps réel (en continu)

```

sketch_apr7a.ino
13
14 void loop() {
15   Wire.beginTransmission(0x68);
16   Wire.write(0x3B); // début des registres accelerometer
17   Wire.endTransmission(false);
18   Wire.requestFrom(0x68, 6, true);
19
20   int16_t AcX = Wire.read() << 8 | Wire.read();
  
```

Output Serial Monitor X

Message (Enter to send message to 'Arduino UNO' on 'COM3')

```

X: -10140 | Y: 10032 | Z: 17240
X: -10288 | Y: 10384 | Z: 16864
X: -10056 | Y: 9920 | Z: 17116
X: -10264 | Y: 10332 | Z: 17012
X: -10172 | Y: 10136 | Z: 17268
X: -10164 | Y: 10204 | Z: 17224
X: -10220 | Y: 10216 | Z: 17000
X: -10108 | Y: 9908 | Z: 17052
X: -10332 | Y: 10280 | Z: 16980
X: -10124 | Y: 9976 | Z: 17444
X: -10404 | Y: 10260 | Z: 16908
X: -10184 | Y: 10100 | Z: 17192
X: -10068 | Y: 9872 | Z: 16976
X: -10280 | Y: 10300 | Z: 16952
X: -10088 | Y: 9912 | Z: 17168
X: -10316 | Y: 10264 | Z: 17200
X: -10108 | Y: 9952 | Z: 17436
  
```

Un modèle d'intelligence artificielle a été utilisé afin d'améliorer la fiabilité du système de détection des accidents et de réduire les fausses alertes. À cet effet, le modèle a été entraîné sur une base de données réelle. Les résultats ont ensuite été affichés via une application web développée avec Streamlit.

Le programme d'entraînement du modèle est présenté ci-dessous.

Programme d'entraînement IA et de création de fichier PKL (joblib)

```

import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import joblib

df = pd.read_csv("mpu6050_dataset_labeled.csv")
print("Aperçu des données :")
print(df.head())

df['acc_magnitude'] = (df['ax']**2 + df['ay']**2 + df['az']**2)**0.5
df['gyro_magnitude'] = (df['gx']**2 + df['gy']**2 + df['gz']**2)**0.5
  
```

```
df = df.drop(columns=['acc_magnitude', 'gyro_magnitude'], errors='ignore')
X = df[['ax', 'ay', 'az', 'gx', 'gy', 'gz']]
y = df['label']
print("\nRépartition des classes :")
print(y.value_counts())
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y)
model = Pipeline([
    ('scaler', StandardScaler()),
    ('classifier', RandomForestClassifier(
        n_estimators=150,
        max_depth=10,
        random_state=4)))
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

print("\n Accuracy :", accuracy_score(y_test, y_pred))
print("\n Classification Report :\n")
print(classification_report(y_test, y_pred))
print("\n Matrice de confusion :\n")
print(confusion_matrix(y_test, y_pred))

plt.figure(figsize=(6,4))
plt.hist(y_test, alpha=0.5, label="Vrai")
plt.hist(y_pred, alpha=0.5, label="Prédit")
plt.legend()
plt.title("Comparaison des classes")
plt.show()
joblib.dump(model, "mpu6050_model.pkl")
print("\n Modèle sauvegardé : mpu6050_model.pkl")
```

La figure 3.3 montre la détection d'accident par intelligence artificielle sur une application web Streamlit. La partie supérieure illustre une conduite normale, tandis que la partie inférieure montre que, lorsqu'un accident se produit, l'application Streamlit affiche une alerte d'accident.



Figure 3.3 : Résultat sur l'application web Streamlit

Programmation de la carte Arduino :

L'application proposée comporte à la fois un aspect matériel et logiciel. Le fonctionnement logiciel est basé sur l'organigramme. Le programme est composé de deux parties :

La première partie est l'initialisation du programme lecteur des variables et configuration des entrées et sorties. La deuxième partie est la partie principale fait lecteur des capteurs et s'exécute en boucle qui décrit le fonctionnement du système d'alerte.

Pour entraîner le modèle non linéaire de détection d'accident, nous utilisons un programme fondé sur l'intelligence artificielle afin de reproduire fidèlement la complexité des données.

Cette approche permet d'identifier avec efficacité les comportements atypiques et les anomalies sur la route :

Dans un second temps, nous avons utilisé la carte ESP32 pour recevoir l'alerte via Bluetooth. L'alerte est reçue sur le téléphone, comme le montre la figure 3.6.

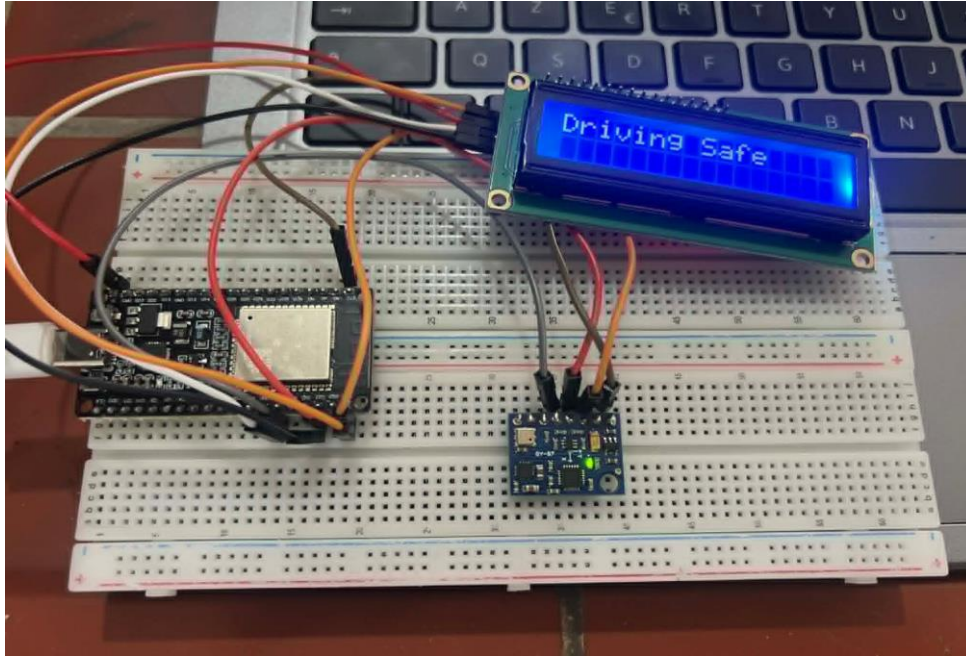


Figure 3.4 : Montage expérimental ESP32–MPU6050 avec communication Bluetooth

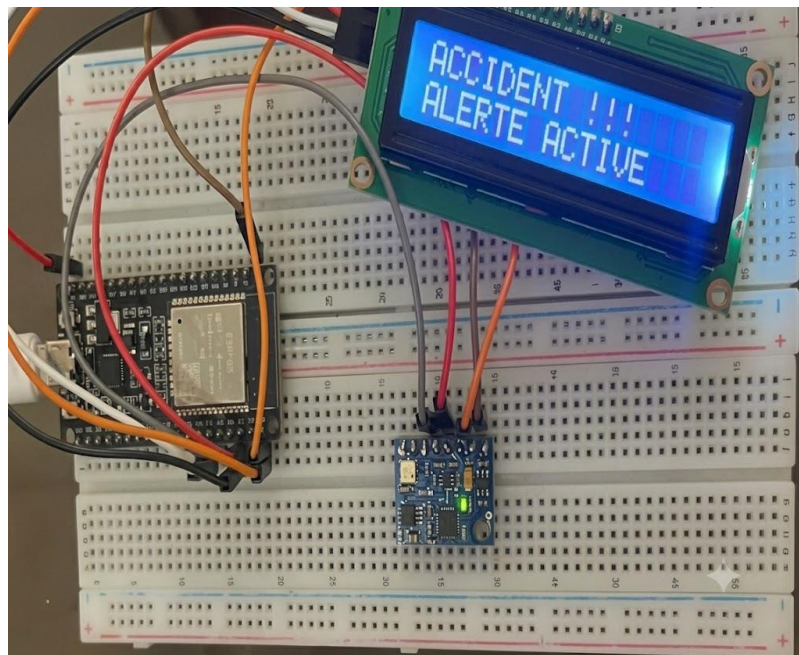




Figure 3.5 : Résultat de l’affichage après une détection d’accident

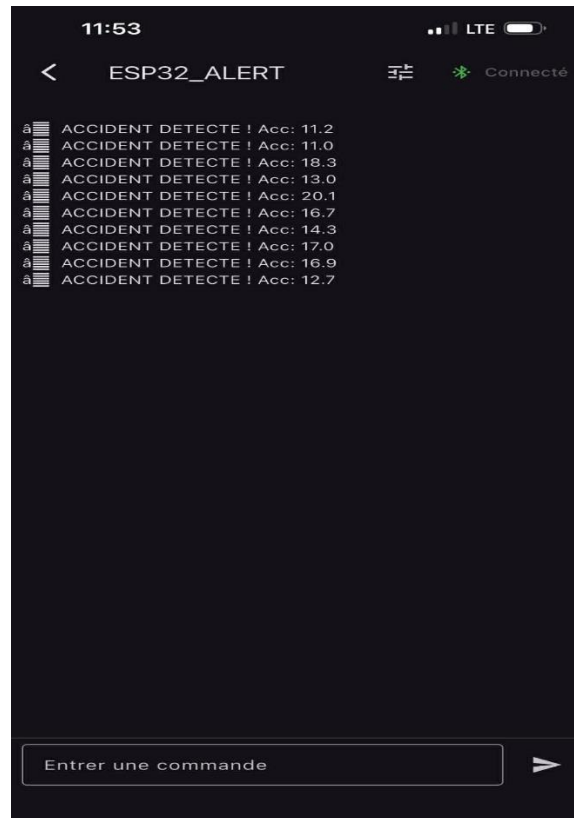


Figure 3.6 : Affichage de l’alerte sur l’écran du téléphone via Bluetooth

Enfin, nous avons réalisé le montage pratique en utilisant la carte ESP32, le capteur MPU6050, le module GSM SIM800L et le module GPS NEO-8M. (Figure 3.7)

Après la mise en fonctionnement du système, le module GSM s’est connecté au réseau, tandis que le module GPS a acquis les signaux des satellites après un certain temps. Ensuite, un accident a été simulé en agitant le système. Le capteur a détecté l’accident, puis le GPS a récupéré les coordonnées géographiques (latitude et longitude). Ces coordonnées ont été transmises par SMS via le module GSM à l’aide de commandes AT. Enfin, l’alerte, sous forme de SMS et d’appel téléphonique, a été reçue sur le téléphone.

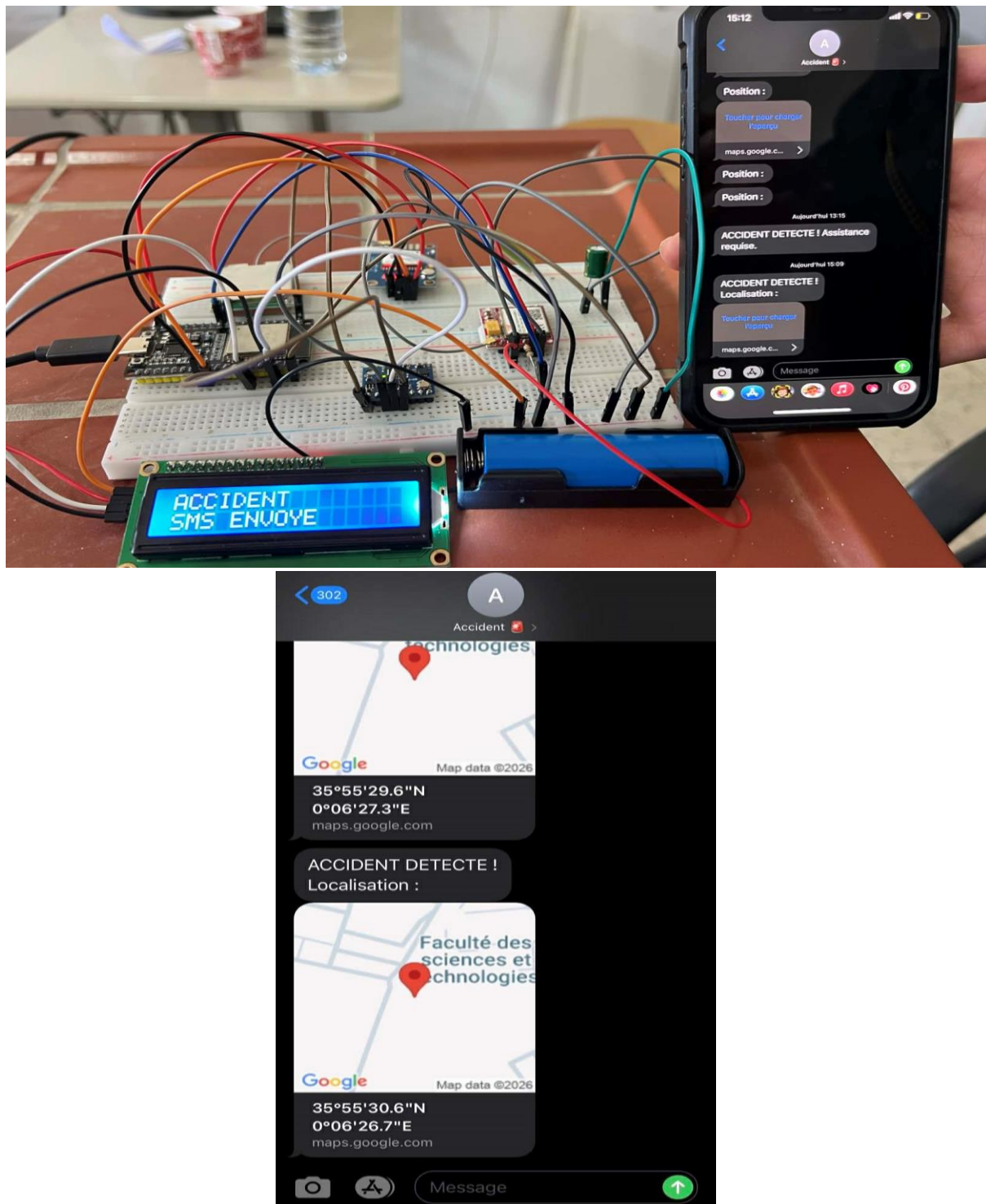


Figure 3.7 : Affichage des résultats sur l'écran du téléphone et l'écran LCD via le module SIM800L

Le code Arduino utilisé pour cette détection est présenté ci-dessous.

Lorsqu'un accident est détecté, les coordonnées GPS sont récupérées puis transmises par SMS et par appel téléphonique afin d'alerter les secours ou les proches.

Programme de Détection d'Accident et d'Alerte SMS/Appel (ESP32 - MPU6050 - GSM)

```
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>
#include <TinyGPSPlus.h>
#include <LiquidCrystal_I2C.h>
Adafruit_MPU6050 mpu;
TinyGPSPlus gps;
HardwareSerial gpsSerial(2);
HardwareSerial sim800(1);
LiquidCrystal_I2C lcd(0x27, 16, 2);
#define SDA_PIN 21
#define SCL_PIN 22
#define GPS_RX 16
#define GPS_TX 17
#define SIM_RX 26
#define SIM_TX 27

String numero = "0556014136";
bool accidentTraite = false;

void setup() {
  Serial.begin(115200);
  Wire.begin(SDA_PIN, SCL_PIN);
  lcd.init();
  lcd.backlight();
  lcd.setCursor(0, 0);
  lcd.print("Initialisation");

  if (!mpu.begin()) {
    lcd.clear();
    lcd.print("Erreur MPU6050");
    while (1);
  }
  gpsSerial.begin(9600, SERIAL_8N1, GPS_RX, GPS_TX);
  sim800.begin(9600, SERIAL_8N1, SIM_RX, SIM_TX);
  delay(3000);
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Systeme Pret");
}

void loop() {
  while (gpsSerial.available()) {
```

```

    gps.encode(gpsSerial.read());
}
sensors_event_t a, g, temp;
mpu.getEvent(&a, &g, &temp);
float accel = sqrt(
    a.acceleration.x * a.acceleration.x +
    a.acceleration.y * a.acceleration.y +
    a.acceleration.z * a.acceleration.z
);
Serial.print("Acceleration = ");
Serial.println(accel);

if (accel > 17.0 && !accidentTraite) {
    accidentTraite = true;
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("ACCIDENT !");
    lcd.setCursor(0, 1);
    lcd.print("Envoi SMS...");
    String sms = "ACCIDENT DETECTE !\n";

    if (gps.location.isValid()) {
        double lat = gps.location.lat();
        double lng = gps.location.lng();
        sms += "Localisation :\n";
        sms += "https://maps.google.com/?q=";
        sms += String(lat, 6);
        sms += ",";
        sms += String(lng, 6);
    } else {
        sms += "GPS non disponible";
    }
    envoyerSMS(sms);
    delay(10000);
    lcd.clear();
    lcd.print("Appel Urgence");
    appelerNumero();
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("ACCIDENT");
    lcd.setCursor(0, 1);
    lcd.print("SMS ENVOYE");

    while (true) {

```

```
    delay(1000);
  }
}
delay(100);
}

void envoyerSMS(String texte) {

  sim800.println("AT+CMGF=1");
  delay(1000);
  sim800.print("AT+CMGS=\"");
  sim800.print(numero);
  sim800.println("\"");
  delay(1000);
  sim800.print(texte);
  delay(500);
  sim800.write(26); // CTRL+Z
  delay(5000);
}

void appelerNumero() {
  sim800.print("ATD");
  sim800.print(numero);
  sim800.println(";");
  delay(3000);
  sim800.println("ATH");
}
```

Synthèse :

Ce montage, composé de trois modules (un capteur, un module GSM et un module GPS) contrôlé par une carte ESP32, a été réalisé avec succès. Le système permet de détecter efficacement les accidents de véhicules grâce au capteur MPU6050, puis de déterminer la position géographique du véhicule à l'aide du module GPS avant de la transmettre via le module GSM. Un programme Arduino a été téléversé sur la carte ESP32 afin d'assurer le bon fonctionnement et la coordination des différents modules. Par ailleurs, l'intégration d'un modèle d'intelligence artificielle constitue une approche efficace pour améliorer la fiabilité du système.

Conclusion :

Ce troisième chapitre a été consacré à la conception, à la simulation et à la réalisation pratique du système de détection d'accidents proposé.

Dans un premier temps, une simulation du montage a été réalisée sous Proteus ISIS. Celle-ci était constituée d'une carte Arduino Nano, d'un module GPS, d'un module GSM, d'un relais et d'un terminal virtuel permettant d'afficher les SMS envoyés contenant les coordonnées géographiques. Cette étape a permis de valider le schéma électronique ainsi que le fonctionnement global du système avant sa mise en œuvre pratique.

Dans un second temps, le système a été réalisé expérimentalement à l'aide d'une carte Arduino Uno puis d'une carte ESP32. Une communication Bluetooth a également été mise en place afin de visualiser les données acquises en temps réel. Afin d'améliorer les performances de détection et de réduire les fausses alertes, une approche basée sur l'intelligence artificielle a été adoptée. Un modèle a été entraîné à partir d'une base de données réelle, puis les résultats de détection ont été affichés à travers une application web développée avec Streamlit.

Enfin, le système final, composé du capteur MPU6050, des modules GPS et GSM ainsi que de la carte ESP32, a été réalisé avec succès. Les essais effectués ont montré qu'en cas d'accident simulé, le système est capable de détecter l'événement, de récupérer automatiquement la position géographique du véhicule et de transmettre une alerte sous forme de SMS et d'appel téléphonique vers un numéro prédéfini. Les résultats obtenus confirment l'efficacité et la fiabilité de la solution proposée. Un tel système peut contribuer à réduire le temps d'intervention des secours et ainsi participer à la préservation des vies humaines en cas d'accident de la circulation.

Références

- [1] Doshi, A., Shah, B., & Kamdar, J. "Accident Detection and Alert System." International Journal of All Research Education and Scientific Methods (IJARESM), vol. 9, no. 11, pp. 114–119.
- [2] Baballe, M. A., Bello, M. I., Alkali, A. U., Abdulkadir, Z., Muhammad, A. S., & Muhammad, F. "The Unmanned Aerial Vehicle (UAV): Its Impact and Challenges." 2022.
- [3] Baballe, M. A. "Accident Detection System with GPS, GSM, and Buzzer." TMP Universal Journal of Research and Review Archives, vol. 2, no. 1, pp. 28–36, 2023.
- [4] Atlas Magazine, "Sécurité routière : état des lieux dans le monde," 2013.
<https://www.atlas-mag.net/fr/articles/securite-routiere-etat-des-lieux-dans-le-monde-en-afrique-au-maghreb-et-au-moyen-orient>
- [5] Innovauto. "Systèmes embarqués." Innovauto. <https://www.innovauto.org/systemes-embarques>
- [6] Innovauto. "Les dispositifs eCall et bCall." Innovauto.
<https://www.innovauto.org/systemes-embarques/les-dispositifs-ecall-et-bcall>
- [7] Ouest-France, "Grâce à l'eCall : comment ça marche ?", 11 juillet 2023.
<https://www.ouest-france.fr/leditiondusoir/2023-07-11/grace-a-l-ecall-les-voitures-alertent-les-secours-automatiquement-mais-comment-ca-marche-vraiment-ac0df3b8-081d-43ea-a4dc-436ea35c6e18>
- [8] Cats Power Design. "Système embarqué." Cats Power Design.
<https://www.catspowerdesign.fr/actualites/systeme-embarque>
- [9] Geotab. "GPS." Geotab. <https://www.geotab.com/fr/glossaire/gps/>
- [10] Parlons Sciences, "Les mathématiques du GPS." <https://parlonssciences.ca/ressources-pedagogiques/documents-dinformation/les-mathematiques-qui-font-fonctionner-le-gps>
- [11] D3E. "Principe de fonctionnement du GPS." D3E.
https://www.d3e.fr/geolocalisation/geo_support_principe-gps.html
- [12] D3E. "Principe de fonctionnement du GPS." D3E.
https://www.d3e.fr/geolocalisation/geo_support_principe-gps.html
- [13] Mouly, M., & Pautet, M.-B. The GSM System for Mobile Communications. Cell & Sys, 1992.
- [14] IEEE Intelligent Transportation Systems Society. IEEE. <https://itss.ieee.org/>
- [15] "Cooperative RSU Based Detection and Prevention of Sybil Attacks in Routing Process of VANET."

- [16] Université Badji Mokhtar Annaba. "DJABOUR ABI Ines – Mémoire de fin d'études." 2021. <https://biblio.univ-annaba.dz/ingeniorat/wp-content/uploads/2021/03/DJABOURABI-Ines.pdf>
- [17] F. Aubepart, "Conception d'un système embarqué communicant," J3eA, 2017.
- [18] OnStar. "Automatic Crash Response." General Motors. <https://www.onstar.ca/fr/services/automatic-crash-response>
- [19] Instructables. "DIY Arduino GPS Tracker With Real-Time Location." <https://www.instructables.com/DIY-Arduino-GPS-Tracker-With-Real-Time-Location-on/>
- [20] Manchester Metropolitan University. "Harnessing Smartphone Sensors." <https://e-space.mmu.ac.uk/638940/1/Harnessing%20Smartphone%20Sensors-Published.pdf>
- [21] J. Ralyté, "Méthodes et Ingénierie de Méthodes," Encyclopédie de l'informatique et des systèmes d'information, Vuibert, 2006.
- [22] Cairn.info, "Sciences, technologies et médecine." <https://stm.cairn.info/?lang=fr>
- [23] Arduino CC, "Arduino Uno Rev3 Product Reference." <https://www.arduino.cc/en/Main/ArduinoBoardUno>
- [24] CCDMD, "Comparaison d'Arduino et de Raspberry Pi." <https://objets.ccdmd.qc.ca/manuel/1-3-comparaison-darduino-et-de-raspberry-pi/>
- [25] Atmel Corporation, "8-bit AVR Microcontroller with 32K Bytes In-System," 2015.
- [26] Conrad, "Arduino : guide officiel." <https://www.conrad.fr/fr/guides/materiel-educatif-kits-de-developpement/conrad-official-reseller-arduino.html>
- [27] M. Banzi and M. Shiloh, "Getting Started with Arduino: The Open Source Electronics Prototyping Platform," 2022.
- [28] M. Benhamida, "Microprocesseur," Université Abou Bekr Belkaïd – Tlemcen, 2023–2024.
- [29] E. Bartmann, Le grand livre d'Arduino, Eyrolles, 2014.
- [30] SlideShare, "Présentation Arduino pour les débutants." <https://fr.slideshare.net/slideshow/presentation-arduino-pour-les-debutant-pptx/274932432>
- [31] Blaise Pascal, "Présentation d'Arduino." <https://arduino.blaisepascal.fr/presentation/>
- [32] Tim Montmorency, "Le langage Arduino." <https://tim-montmorency.com/rel-electro-immersif/arduino/code.html>
- [33] Arduino, "Language Reference." <https://www.arduino.cc/reference/en/>

- [34] B. Kerrouche et B. Belabbas, "Conception et réalisation d'un système embarqué de surveillance et de sécurité automobile," Mémoire de Master, Université Kasdi Merbah Ouargla, 2022.
- [35] Leygues Couffignal, "Référence du langage Arduino."
<http://sin.lyceeleyguescouffignal.fr/arduino-langage-reference>
- [36] PowerLab DZ, "Arduino Uno R3." <https://powerlab.dz/product/6038/>
- [37] U.S. Department of Defense, "Global Positioning System Standard Positioning Service Performance Standard," 2008.
- [38] C. Vardon, "Introduction aux télécommunications."
<https://www.cvardon.fr/introduction%20aux%20telecommunications.pdf>
- [39] National Marine Electronics Association (NMEA), "NMEA 0183 Standard for Interfacing Marine Electronic Devices," 2002.
- [40] P. Horowitz and W. Hill, "The Art of Electronics," Cambridge University Press, 2015.
- [41] Dymadis, "Module GSM/GPRS SIM900 pour les projets électroniques, robotique et Arduino." <https://dymadis.com/module-gsm-gprs-sim900-pour-les-projet-electronique-robotique-arduino-iot>
- [42] M. Margolis, Arduino Cookbook, O'Reilly Media, 2011.
- [43] SIMCom Wireless Solutions, SIM900 Hardware Design, Version 1.06, 2010.
- [44] Robotique Tech, "Cahier d'activités pour la carte ESP32."
<https://www.robotique.tech/tutoriel/cahier-dactivites-pour-la-carte-esp32/>
- [45] SIMCom Wireless Solutions, SIM900 AT Command Manual, Version 1.03, 2012.
- [46] Algo TN, "Introduction à l'ESP32." <https://www.algo.tn/esp32/introduction/>
- [47] LED-Diode, "Module GY-521 – MPU-6050." <https://fr.led-diode.com/sensor-module/gy-521-mpu-6050-three-axis-gyroscope.html>
- [48] Analog Devices / Maxim Integrated, "Understanding UART Communication Protocol," 2021.
- [49] InvenSense Inc., MPU-6000/MPU-6050 Product Specification, Rev. 3.4, 2013.
- [50] NXP Semiconductors, I2C-bus Specification and User Manual (UM10204), 2014.
- [51] Otronic, "Module de données GSM/GPRS SIM800L." <https://www.otronic.nl/fr/module-de-donnees-gsm-gprs-sim800l.html>
- [52] Motorola Semiconductor, SPI Block Guide V03.06, 2003.
- [53] ElectroDuino, "SIM800L GSM Module Pinout and Description."
<https://www.electroduino.com/sim800l-gsm-module/>

- [54] M. Bondu et Arduino Team, "SoftwareSerial Library Reference."
- [55] J. Rowberg, I2Cdevlib: MPU-6050 Arduino Library, 2012

ANNEXE

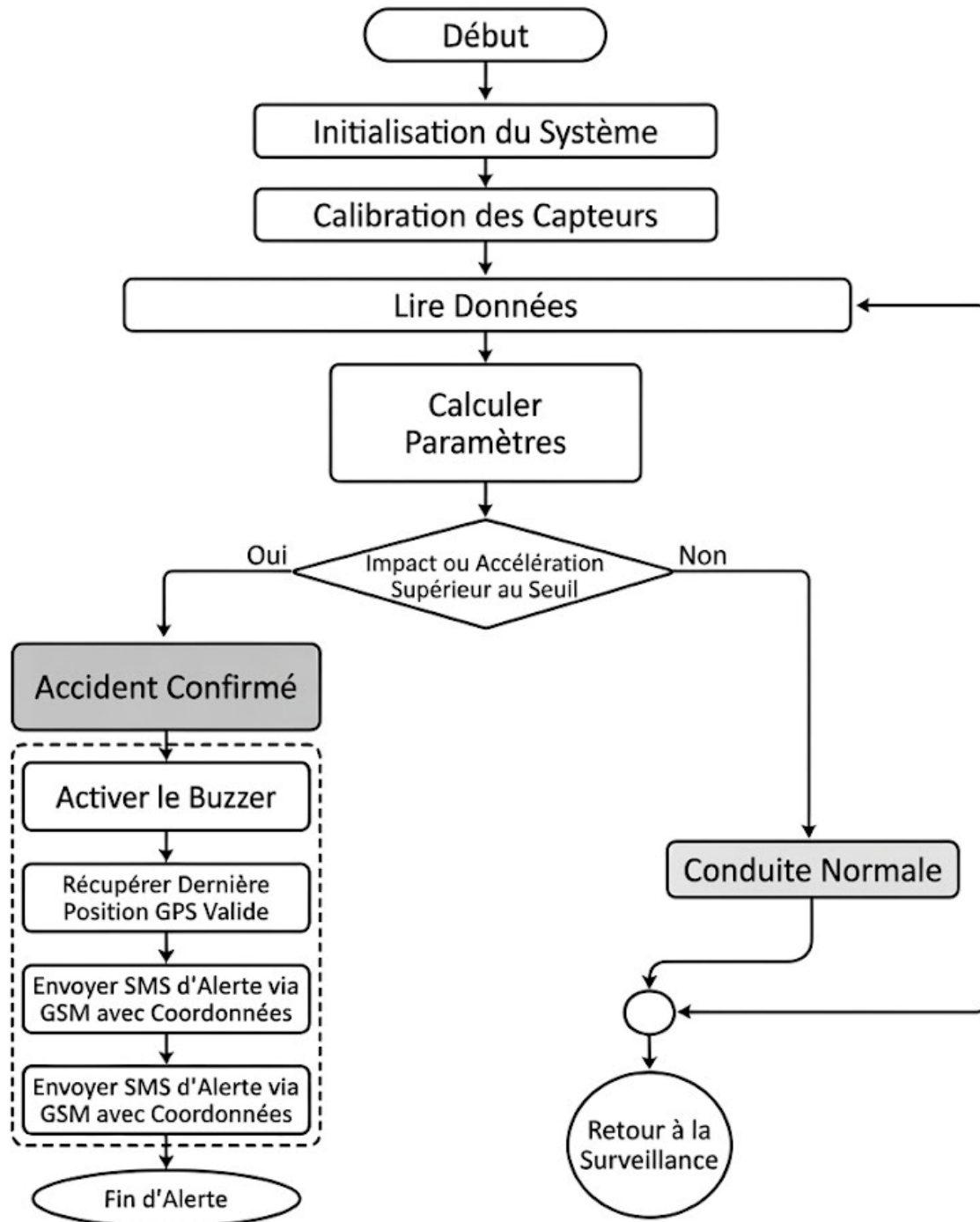


Figure 1 : Organigramme général du système de détection d'accident et de monitoring de conduire

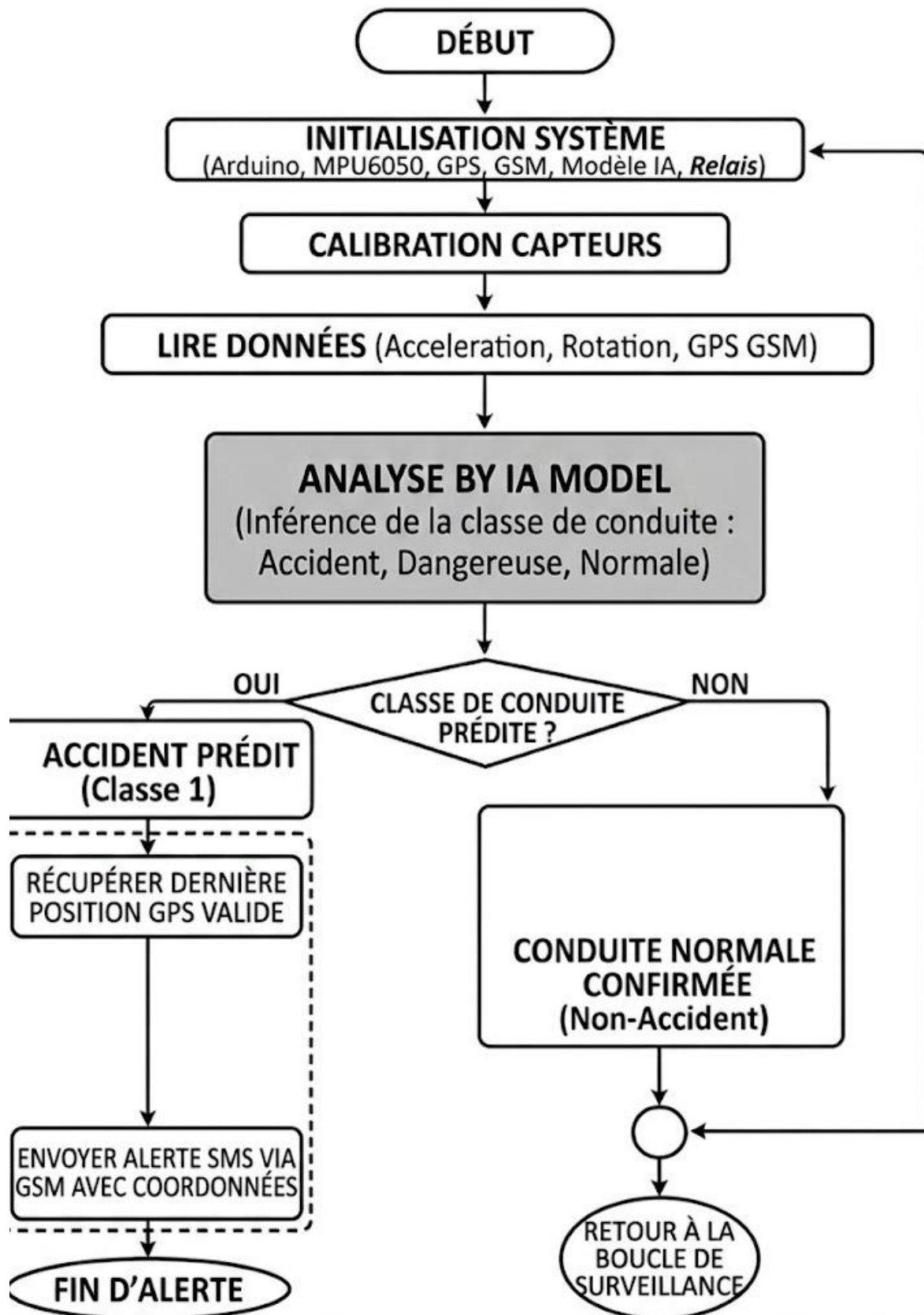


Figure 2 : Organigramme décisionnel basé sur un modèle d'intelligence Artificielle pour la classification de conduite