Royaume du Maroc
UNIVERSITÉ HASSAN II DE CASABLANCA
ÉCOLE NATIONALE SUPÉRIEURE D'ÉLECTRICITÉ ET DE MÉCANIQUE

ENSEM

المملكة المغربية
جامعة الحسن الثاني بالدار البيضاء
المدرسة الوطنية العليا للكهرباء والميكانيك

Département de Génie Electrique

Filière : *Génie Electrique, Systèmes Embarqués et Télécommunications*

Niveau : Deuxième année Cycle Ingénieur

Rapport de Projet de Fin d'Année

Robot intelligent de détection des débris étrangers (FOD) sur piste d'atterrissage

Soutenu le 18/06/2025

Réalisé par :

ADA Ouèssè Amédé

JORFAOUI Taha

Membres du Jury :

Mme Soumia BAKKALI	Encadrante
Mme Bahia ALAOUI	Examinatrice
Pr Ahmed ERRAMI	Examinateur

Année Universitaire 2024-2025

Remerciements

Nous exprimons notre profonde reconnaissance à Mme Soumia BAKKALI, notre encadrante, pour son accompagnement remarquable tout au long de ce projet passionant. Par son écoute active, ses conseils avisés et son encadrement pédagogique rigoureux, elle a su nous guider avec bienveillance à chaque étape du projet, nous encourager et nous pousser à donner le meilleur de nous-mêmes.
Au-delà de son rôle académique, nous tenons à saluer son geste généreux et décisif, en prenant à sa charge l'achat de plusieurs composants essentiels, dont le coût aurait représenté un obstacle majeur pour nous. Grâce à son soutien intellectuel, humain et matériel, nous avons pu concrétiser ce projet ambitieux dans les meilleures conditions.

Nous adressons nos plus chaleureux remerciements à l'ensemble des enseignants de la filière Génie Électrique, Systèmes Embarqués et Télécommunications (GESET) pour la richesse des enseignements qu'ils nous ont transmis au cours de notre formation.

Nos remerciements vont également à l'endroit des membres du jury pour le temps qu'ils nous ont consacré, pour l'attention portée à notre travail et pour leurs suggestions constructives.

Résumé du Projet

Les débris étrangers présents sur les pistes d'atterrissage, communément appelés FOD (Foreign Object Debris), représentent une menace importante pouvant causer des dommages aux avions lors des phases critiques de décollage et d'atterrissage. Leur détection rapide et précise est donc cruciale pour assurer la sécurité des opérations aériennes, un enjeu majeur dans le domaine aéronautique.
Traditionnellement, la surveillance des pistes repose sur des inspections humaines ou des systèmes peu automatisés ou très souvent coûteux.

Pour répondre à ces limites, notre projet propose un robot autonome intelligent, capable de parcourir une piste d'atterrissage, de détecter les FOD, et de générer une cartographie visuelle précise des zones potentiellement contaminées. Ce robot innovant allie plusieurs technologies accessibles et complémentaires :

- **Système embarqué double plateforme :** Le déplacement et la navigation du robot sont assurés par un microcontrôleur Arduino, qui contrôle les moteurs, traite les informations issues des capteurs de position et d'évitement d'obstacles. Ce choix garantit une gestion efficace des déplacements sur la piste, avec un contrôle fin et en temps réel.
- **Traitement d'images avancé :** Une Raspberry Pi 5, équipée d'une caméra haute définition, est dédiée à la détection visuelle. Le robot exploite un modèle d'intelligence artificielle basé sur YOLO (You Only Look Once), entraîné spécifiquement avec une base de données d'images de FODs capturées sur site. Cette approche permet une détection rapide, précise et robuste des objets étrangers.
- **Géolocalisation et cartographie :** Le système rapporte en continu la position des FODs détectés, ce qui permet de construire une carte visuelle dynamique de la piste, facilitant l'intervention rapide des équipes de nettoyage.

L'originalité de notre solution réside dans la combinaison d'une architecture modulaire simple à mettre en œuvre, avec des composants facilement accessibles et un algorithme IA performant entraîné sur des données réelles et personalisées. Ce design assure non seulement une bonne efficacité opérationnelle, mais aussi une grande facilité d'usage et de maintenance, essentielle pour une adoption pratique dans les aéroports.
Ainsi, notre robot autonome se positionne comme une solution innovante, économique et évolutive, capable d'améliorer significativement la sécurité des pistes d'atterrissage tout en réduisant les coûts et les efforts humains.

MOTS CLES :
FOD, piste d'atterrissage, sécurité aéroportuaire, avion, intelligence artificielle, navigation, inspection

Executive Abstract

Foreign Object Debris (FOD) on runways represents a serious safety hazard, capable of causing significant damage to aircraft during the critical phases of takeoff and landing. Ensuring flight safety therefore requires rapid, accurate, and reliable detection of such debris—an ongoing challenge in the aviation sector.
Traditionally, runway inspections have relied on manual procedures or costly, partially automated systems, limiting both their frequency and effectiveness.

To address these limitations, our project proposes an intelligent autonomous robot capable of navigating a runway, detecting FOD, and generating a precise visual map of potentially contaminated areas. This innovative robot combines several accessible and complementary technologies :

- **Dual-platform embedded system :** The robot's movement and navigation are provided by an Arduino microcontroller, which controls the motors and processes information from the position and obstacle avoidance sensors. This choice guarantees efficient management of movements on the track, with fine, real-time control.
- **Advanced Image Processing :** A Raspberry Pi 5, equipped with a high-definition camera, is dedicated to visual detection. The robot uses an artificial intelligence model based on YOLO (You Only Look Once), specifically trained with a database of FOD images captured on site. This approach enables fast, accurate, and robust foreign object detection.
- **Geolocation and Mapping :** The system continuously reports the position of detected FOD, allowing for the construction of a dynamic visual map of the runway, facilitating rapid intervention by cleaning teams.

The originality of our solution lies in the combination of a modular architecture that is easy to implement, with easily accessible components and a powerful AI algorithm trained on real and personalized data. This design ensures not only good operational efficiency, but also ease of use and maintenance, essential for practical adoption in airports.
Thus, our autonomous robot is positioned as an innovative, economical, and scalable solution, capable of significantly improving runway safety while reducing costs and human effort.

KEYWORDS :
FOD, runway, airport security, aircraft, artificial intelligence, navigation, inspection

Table des matières

Table des figures

Liste des tableaux

Liste des Abbréviations

Termes techniques et Sigles aéronautiques utilisés dans le présent rapport :

ASDA Acceleration Stop Distance Available
CNN Convolutional Neural Network
EASA European Union Aviation Safety Agency
FAA Federal Aviation Administration
FO Foreign Object
FOD Foreign Object Debris
FODEX FOD EXplorer : Nom donné au robot objet du présent projet
LDA Landing Distance Available
mAP mean Average Precision
NTSB National Transportation Safety Board
OACI Organisation de l'Aviation Civile Internationale
ONDA Office National des Aéroports
RESA Runway End Safety Area
TODA Take Off Distance Available
TORA Take Off Run Available
YOLO You Only Look Once

Introduction générale

La sécurité aérienne demeure l'un des piliers fondamentaux de l'aviation civile, un secteur où la moindre défaillance peut avoir des conséquences humaines, matérielles et économiques considérables. Parmi les nombreux facteurs susceptibles de compromettre cette sécurité, la présence de débris étrangers sur les pistes d'atterrissage – FOD (Foreign Object Debris). Ils représentent une menace réelle et documentée. De nombreux accidents tragiques ont démontré qu'un simple fragment métallique, à peine visible à l'œil nu, pouvait entraîner une catastrophe aérienne majeure. Au-delà des vies humaines, les FODs coûtent à l'industrie aéronautique mondiale plusieurs milliards de dollars chaque année, en raison des dommages aux aéronefs, des retards d'exploitation, et des opérations de maintenance non prévues.
Face à cet enjeu critique, la détection et l'élimination rapide des FOD sont devenues des priorités absolues pour les autorités aéroportuaires et les organismes aéronautiques internationaux. Pourtant, les systèmes actuellement en place restent limités par leur coût, ou leur complexité de déploiement. De plus, leur efficacité repose souvent sur des inspections manuelles ou des technologies propriétaires difficilement accessibles pour les aéroports de petite et moyenne taille, notamment dans les pays en développement.

Le présent projet s'inscrit dans une démarche à la fois technologique, pédagogique et sociétale : la conception et la réalisation d'un robot autonome de détection des FODs, combinant intelligence artificielle, navigation autonome et cartographie dynamique. Ce robot, nommé FODEX, repose sur une architecture double plateforme : une plateforme Arduino dédiée au pilotage et à la navigation, et une plateforme Raspberry Pi5 assurant le traitement d'images pour identifier les objets suspects sur piste.
L'objectif de ce projet est double. Sur le plan technique, il s'agit de concevoir un système embarqué fiable, évolutif et peu coûteux, capable de détecter et localiser les FOD avec précision. Sur le plan applicatif, il vise à offrir aux exploitants d'aérodromes une alternative pratique et économique aux solutions industrielles existantes.

Ce rapport présente l'ensemble de la démarche suivie, depuis l'analyse du problème jusqu'à l'implémentation et la validation du prototype, mettant en lumière les choix technologiques, les défis rencontrés, et les perspectives d'amélioration d'un projet qui ambitionne de contribuer activement à la modernisation des outils de sûreté aéroportuaire.

Chapitre 1

Cadre général et Cahier des charges

1.1 Mise en contexte

1.1.1 Définition et origine des FOD

Il n'existe pas de définition standardisée de ce que sont les débris étrangers appelés en anglais Foreign Objects Debris ou FOD. Il faut surtout retenir qu'il s'agit d'un "objet inanimé présent sur la chaussée ou ses abords immédiats et dont la présence est inappropriée et inattendue".

La nature exacte des FOD varie énormément. Ils peuvent être composés de n'importe quel matériau et être de n'importes quelles couleur et taille. Les FOD typiques sont constitués entre autres d'**éléments de fixations d'aéronefs et de moteurs** (écrous, boulons, rondelles, fils de sécurité, etc.) ; des **pièces d'aéronefs** (bouchons de réservoir, fragments de train d'atterrissage, bâtonnets d'huile, tôles, trappes et fragments de pneus) ; des **outils de mécanicien** ; des **fournitures de restauration** ; des **articles de vol** (clous, badges du personnel, stylos, crayons, étiquettes de bagages, canettes de soda, etc.) ; des **articles de tablier** (débris de papier et de plastique provenant de palettes de restauration et de fret, morceaux de bagages et débris d'équipement de piste) ; des **matériaux de piste et de voie de circulation** (morceaux de béton et d'asphalte, joints en caoutchouc et éclats de peinture) ; des **débris de construction** (morceaux de bois, pierres, fixations et divers objets métalliques) ; des **matériaux en plastique et/ou polyéthylène** ; des **matériaux naturels** (fragments de plantes, animaux sauvages et cendres volcaniques) ; des **contaminants hivernaux** (neige, glace)...

En plus de leurs natures très variées, les FOD proviennent également de sources multiples, ce qui complique les efforts visant à assurer la sécurité des opérations sur les aérodromes. Ils peuvent provenir du personnel, des infrastructures aéroportuaires (trottoirs, éclairage et signalisation), de l'environnement et des équipements utilisés sur l'aérodrome (aéronefs, véhicules d'exploitation aéroportuaire, matériel de maintenance,

camions-citernes, autres équipements d'entretien des aéronefs et engins de chantier).

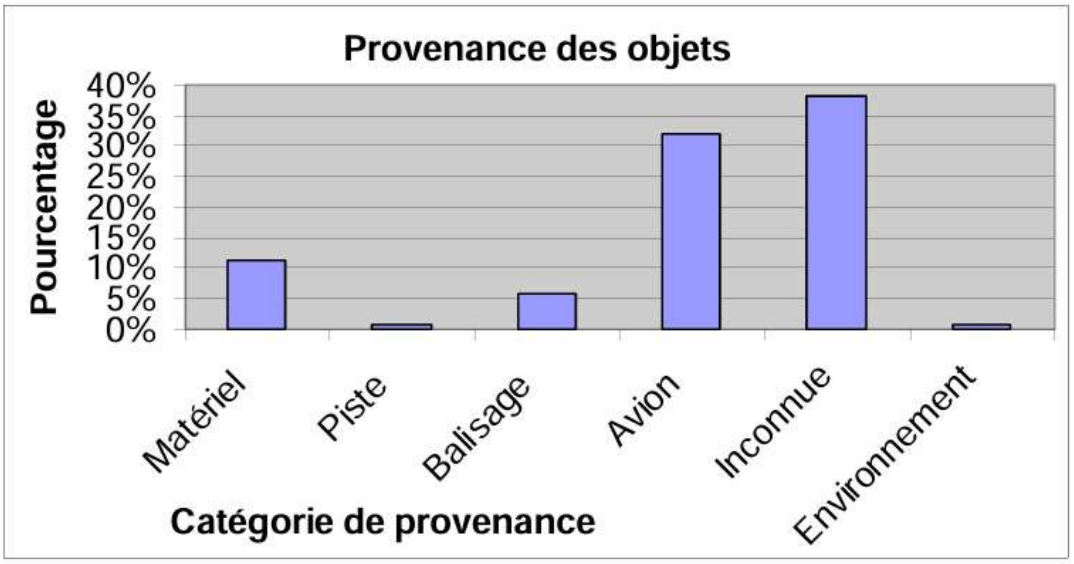


FIGURE 1.1 – Différentes provenances des FOD

Les FOD peuvent s'accumuler sur et sous le matériel de soutien au sol stocké ou installé sur l'aire de trafic de l'aéroport, en particulier dans les zones d'aire de trafic. Le souffle des réacteurs peut ensuite créer des FOD sur la piste lorsqu'un aéronef passe d'une piste relativement large à une voie de circulation plus petite. Les moteurs proches du bord projettent sur la piste les débris et les matériaux non adhérents des accotements vers le centre de la piste ou de la voie de circulation ; surtout dans le cas des quadrimoteurs.

FIGURE 1.2 – Décomposition de la piste en plusieurs zones pour faciliter le mapage

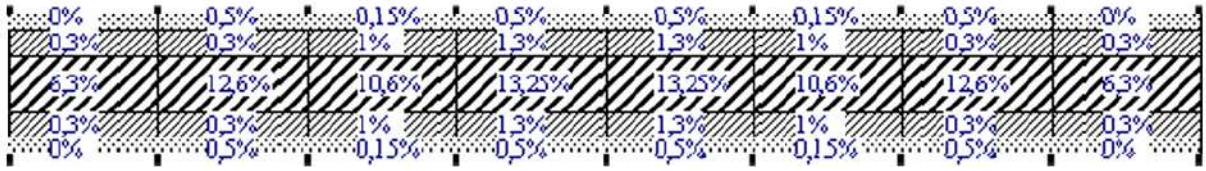


FIGURE 1.3 – Statistique de la répartition des FODs collectées sur les chaussées

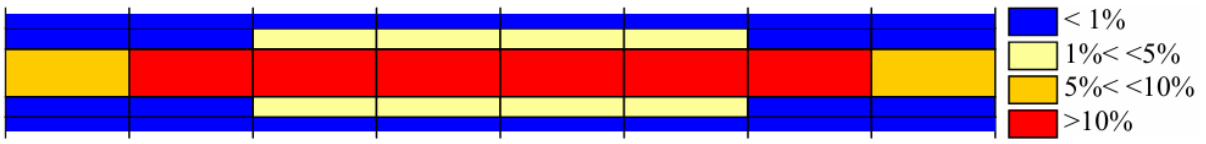


FIGURE 1.4 – Repartition plus visuelle de l'emplacement des FOD sur les pistes

1.1.2 Problématiques et accidents en aviation liés aux FOD

La présence de FOD sur les zone d'opérations aériennes des aéroports constitue une menace importante pour la sécurité du transport aérien. Les dommages causés par des FOD constituent une préoccupation majeure dans les secteurs de l'aéronautique, de l'espace et de la défense. Ils peuvent endommager les avions pendant les phases critiques du vol, entraîner des pertes humaines et des pertes catastrophiques de structures d'avion, et, à tout le moins, une augmentation des coûts de maintenance et d'exploitation.

Les FOD coûtent à l'industrie aéronautique mondiale entre 4 et 13 milliards de dollars par an, en dommages directs, retards, maintenance et pertes d'exploitation. Ils sont impliqués dans 1 à 2% des incidents et accidents d'aviation commerciale documentés, malgré le fait que les bases de données officielles (comme celles de l'OACI, du NTSB, ou de l'EASA) ne catégorisent pas toujours les FOD comme cause principale d'un accident, surtout s'ils ont contribué indirectement.
Sur les pistes ou au sol, les FOD représentent jusqu'à 10% des incidents (roulage, décollage, atterrissage). Les types de dommages potentiels comprennent la perforation des pneus d'avion, l'ingestion par les moteurs, ou le blocage dans les mécanismes affectant les opérations de vol. Des blessures, voire la mort, peuvent survenir lorsque le souffle des réacteurs propulse les FOD à grande vitesse dans l'environnement aéroportuaire.

Dommages aux pneus ou trains d'atterrissage

FIGURE 1.5 – Pneu endommagé par un débris

Les débris sur les pistes (graviers, fragments métalliques) peuvent crever les pneus ou endommager les trains d'atterrissage durant le décollage ou l'atterrissage, entraînant des sorties de piste ou des crash durant ces phases reconnues les plus critiques du vol.
Le **crash du Vol 4590 de Air France Concorde,** est l'un des exemples tragiques illustrant l'ampleur des risques engendrés par un simple débris sur une piste d'atterrissage. Le 25 juillet 2000 à Gonesse, près de Paris, un morceau de titane resté sur la piste a perforé un pneu, provoquant une chaîne d'événements (éclats perçant un réservoir de carburant, incendie) menant à l'écrasement de l'appareil peu après le décollage à Paris, tuant 113 personnes dont 109 à bord et 4 au sol. Ce moreceau, proviendrait d'un avion DC-10 ayant décollé avant le concorde, et mesurait à peine une cinquantaine de cm de long sur 1mm d'épaisseur, négligeable face aux 26m d'envergure et 62m de long du Concorde.

FIGURE 1.6 – Crash du Concorde (Cause : Morceau de Titane)

Cet accident fut parmi les éléments majeurs ayant précipité le retrait du concorde, un projet qui a coûté plus de 2.8 milliards de dollars.

FIGURE 1.7 – Le Concorde

En rappel, le Concorde est un avion de ligne supersonique développé conjointement par la France et le Royaume-Uni. Il était un symbole d'innovation technologique et de prestige, l'un des deux seuls avions de ligne supersoniques ayant opéré commercialement (avec le Tupolev Tu-144 soviétique, lui aussi victime d'accident lié aux FOD). Capable de voler à Mach 2.04 (environ 2 180 km/h), soit plus de deux fois la vitesse du son, il permettait de relier Paris à New York en environ 3h30.

Dommages aux moteurs

L'ingestion de débris (objets métalliques, pierres) dans les réacteurs peut endommager les pales, les compresseurs ou d'autres composants, provoquant une panne partielle ou totale des moteurs, entraînant une perte de poussée, l'incapacité à maintenir l'altitude, un atterrissage d'urgence ou pire, un crash.
Un exemple notable de crash dû à des FOD dans les moteurs, est le **crash du Trans World Airlines (TWA) Flight 2, un Lockheed L-1049 Super Constellation,** survenu le 30 juin 1956 près de l'aéroport international de Kansas City. Le crash a causé la mort de 70 des 74 personnes à bord (passagers et équipage).

Dommages structurels critiques

Les FOD peuvent endommager des éléments structurels essentiels (ailes, fuselage, réservoirs de carburant) occasionnant fuites de carburant, incendies, rupture structurelle, perte d'aérodynamisme. En conséquence, perte de l'intégrité de l'aéronef, des crashs par perte de contrôle ou incendie.
C'est le cas par exemple du **crash du Tupolev Tu-144** (avion supersonique comme le Concorde) en 1973 au Salon du Bourget lors d'une démonstration, causant 14 morts. (La cause exacte reste débattue, des débris supposés sur la piste ont contribué à un crash.)

FIGURE 1.8 – Crash du Tupolev

Obstruction des instruments de vol

Les FOD peuvent obstruer des capteurs critiques comme les tubes de Pitot ou les sondes d'incidence, faussant les données de vol (vitesse, altitude) et provoquant une désorientation des pilotes.

FIGURE 1.9 – Un tube de Pitot mis hors service par un FOD

C'est le triste cas du **crash de Aeroperú Flight 603** en 1996 dans l'océan Pacifique après que du ruban adhésif oublié sur les tubes de Pitot a faussé les indications d'altitude et de vitesse, occasionnant la mort de 70 personnes.

FIGURE 1.10 – crash Aeroperu

Autres accidents

De nombreux autres exemples similaires foisonnent dans l'histoire de l'aviation, des pertes énormes, aussi humainement qu'économiquement. Le tableau ci-dessous liste quelques accidents liés aux FOD et les dégâts engendrés.

Cause du FOD	**Avion impliqué**	**Compagnie**	**Année**	**Dégâts (morts, blessures, dommages)**	**Coût estimé par rapport au prix de l'avion**
Outil oublié sur piste, ingestion moteur au décollage	Boeing 777-200	United Airlines (SFO)	2015	Aucun mort, moteur endommagé, vol avorté	$4M sur $280M
Débris métalliques après maintenance, ingestion par moteur	Airbus A330	Lufthansa	2007	Aucun mort, moteur gravement endommagé	$3.5M sur $230M
Pneu éclaté à cause de FOD sur piste (bouteille)	Boeing 737	Ryanair FR 4102	2008	Aéronef endommagé, 10 blessés	$10M sur $90M
Débris de matériau composite sur piste, ingestion moteur	Boeing 747	British Airways	2010	Moteur endommagé, annulation du vol	$5M sur $350M
Vis oubliée lors maintenance, ingérée par moteur au taxi	Airbus A320	JetBlue	2014	N/A, vol annulé, maintenance lourde	$2M sur $100M

TABLE 1.1 – Autres accidents ou incidents causés par des FODs

1.1.3 Règlementation Internationale en Aéronautique encadrant la gestion des FOD

Structure d'une piste d'atterrissage

Les pistes sont les surfaces d'un aérodrome, réservées au décollage et à l'atterrissage des aéronefs (avions, planeurs, hélicoptères, etc.). Elles peuvent être en béton, en bitume, en asphalte, en herbe, en plaques en acier perforées ou Pierced Steel Planking (PSP), juste en terre battue ou recouverte de neige (altiports).

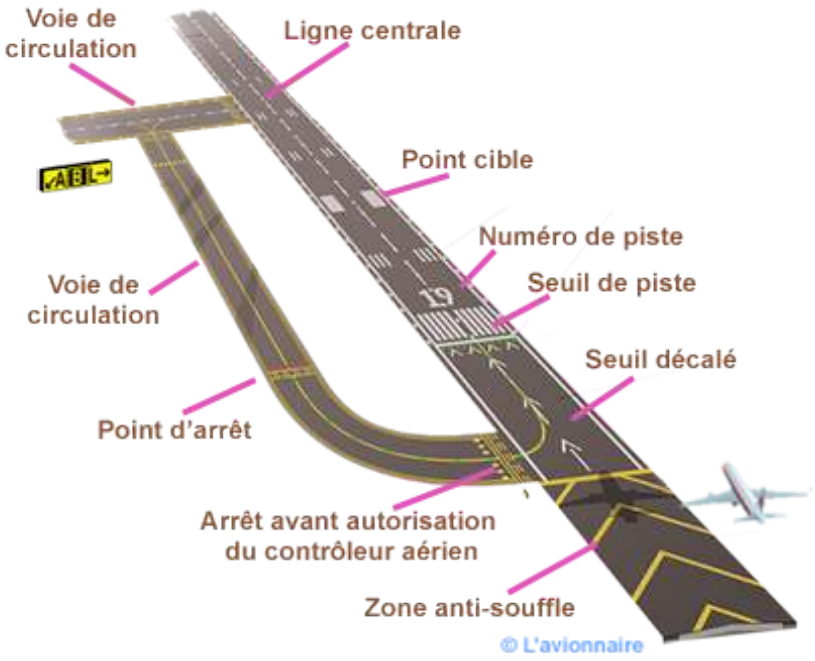


FIGURE 1.11 – Piste Aerodrome

L'aire d'atterrissage comprend la ou les pistes et leurs prolongements éventuels. La piste est l'aire aménagée servant au roulement des aéronefs au décollage et à l'atterrissage.
L'accotement est la partie des abords de piste traités de façon à offrir une surface de raccordement entre cette chaussée et le terrain environnant afin de limiter le risque d'ingestion de corps étranger par des tétra-moteurs.
La bande aménagée sert à limiter les conséquences d'une sortie de piste afin de réduire les dommages sur un aéronef sortant accidentellement de la piste.
La Bande (dégagée) est une aire débarrassée de tout obstacle pouvant présenter un danger pour un aéronef volant à faible hauteur.
L'aire de sécurité d'extrémité de piste RESA est destinée à réduire les risques matériels d'un aéronef se posant trop court ou trop long.
Quant au prolongement d'arrêt (SWY) situé en fin de piste dans le sens du écollage (StopWaY), il est aménagé pour augmenter la distance d'accélération-arrêt pour permettre à un aéronef de terminer sa manœuvre de décollage interrompu dite d'accélération-arrêt.

Les pistes peuvent avoir des dimensions très variables, de 200 mètres à 5 000 mètres de long et de 13 mètres à 60 mètres de large. La fiche d'aérodrome donne pour chaque piste différentes longueurs disponibles, parmi lesquelles la LDA, la TORA, l'ASDA, et la TODA. Le seuil de piste est identifié par une série de bandes rectangulaires. Le nombre de bandes indique la largeur de la piste.

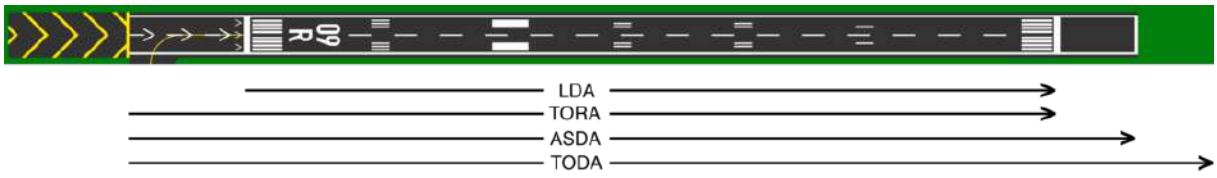


FIGURE 1.12 – Différentes distances dans le marquage d'une piste

Nombre de bandes	Largeur en mètres
4	18 m
6	23 m
8	30 m
12	45 m
16	60 m

FIGURE 1.13 – Largeur de piste en fonction du nombre de bandes du seuil

Dans le cadre de notre projet, nous ne nous intéresserons pas aux voies de circulations, aux taxiways, à l'apron, mais uniquement à la piste d'atterrissage.

FIGURE 1.14 – Vue aérienne de la structure d'un aéroport et des pistes de circulation

Traitement des FOD

Dans le chapitre 05 (CONTRÔLE DES OBJETS INTRUS -FOD) du document 9981 de l'OACI, il est stipulé que les FOD devraient être enlevés dès que possible après leur détection, ceci pouvant être fait de diverses manières, manuellement ou en utilisant des moyens mécaniques. Plus loin, il est rappelé que tous les FOD trouvés sur l'aérodrome et enlevés devraient être enregistrés, analysés et évalués. Au besoin, une enquête devrait être effectuée pour en déterminer les sources. De plus, les sources des FOD, y compris les endroits où elles sont, ainsi que les activités génératrices de FOD sur l'aérodrome devraient être trouvées et enregistrées. L'information recueillie devrait être analysée afin de déceler les tendances et les sources de problèmes et de focaliser les efforts du programme de contrôle des FOD.

Pour consigner l'emplacement de FOD, il conviendrait d'utiliser une carte adéquate de l'aérodrome. Il importe de décrire les FOD correctement afin de permettre la détermination de leur provenance et l'adoption de mesures d'atténuation appropriées.

1.2 Cahier des charges

1.2.1 Introduction au cahier des charges

La sûreté et la sécurité des opérations aéroportuaires sont des enjeux cruciaux dans l'industrie aéronautique.Les FOD, aussi anodins qu'ils puissent paraître peuvent provoquer des dommages matériels considérables.
Dans ce contexte, l'utilisation de systèmes robotisés autonomes pour leur détection représente une solution innovante, efficace et moins dépendante de l'intervention humaine. Le robot que nous nous proposons de concevoir devra être capable de parcourir automatiquement une piste ou une zone prédéfinie, d'identifier la présence de FOD, d'enregistrer leur position géographique, et de transmettre ces informations à un centre de contrôle via une interface utilisateur.

Le présent cahier des charges vise à définir précisément les exigences du système robotisé, en tenant compte des contraintes techniques, environnementales et temporelles du projet. Il constitue une étape essentielle pour garantir une conception cohérente, performante et adaptée aux réalités du terrain.

1.2.2 Spécifications fonctionnelles

SF-01 : Navigation sur la piste

Le robot doit être capable de se déplacer de manière autonome dans un environnement défini sans intervention humaine directe.
Il doit pouvoir suivre une trajectoire composée de segments définis à l'avance, que l'utilisateur peut créer à l'aide d'une interface ou d'une méthode interactive.
À chaque instant, le robot doit être en mesure d'estimer sa position et son orientation pour assurer le suivi de la trajectoire.

SF-02 : Acquisition d'images

Le robot doit capturer des images du sol sur son trajet afin de détecter les objets étrangers (FOD). Cette acquisition est essentielle pour permettre une analyse automatique des anomalies. Le déclenchement est programmé et conditionné par le bon positionnement du robot sur la piste (le placement n'est pas autonome) et le début de la mission d'inspection. Elle prend fin lorsque la piste est complètement sillonnée. En sortie, nous avons des images numériques enregistrées localement.
Exigences :
Le robot doit être en mesure de capturer en continu des images de la piste pendant son déplacement.
La prise d'images doit couvrir efficacement la zone devant lui pour permettre l'analyse de la présence d'objets ou de débris.
Les images doivent être de qualité suffisante pour permettre un traitement fiable, même en conditions de luminosité variables.

SF-03 : Détection des débris

Le robot doit analyser les images captées pour identifier les objets étrangers (FOD) présents sur la surface inspectée, à l'aide d'un algorithme embarqué. L'objectif est d'automatiser la détection des FOD sans intervention humaine, en utilisant des techniques d'intelligence artificielle et de vision par ordinateur.
Traitement à effectuer :
Prétraitement des images (redimensionnement)
Exécution d'un modèle de détection
Détection des objets de taille de plus de 5 cm
Précision de détection $> 85\%$ sur des objets communs (plastiques, métal, outils...)
La fiabilité de détection doit être assurée quel que soit le type de débris (forme, couleur, taille...)

Aucune fausse alerte persistante

SF-04 : Localisation des débris

Lorsqu'un débris est détecté, le robot doit enregistrer la position exacte de ce dernier. Ceci permet une traçabilité spatiale des données d'inspection et faciliter l'intervention rapide en cas de détection de FOD.

Exigences :

La localisation doit être suffisamment précise pour permettre une intervention rapide et efficace.

Le système doit stocker ou transmettre ces coordonnées à un centre de supervision ou à une base de données locale.

Erreur de position inférieure à 5cm dans 90% des cas

SF-05 : Alertes

À chaque fin de mission d'inspection de piste, le robot doit générer une alerte contenant les informations suivantes :

Heure de détection

Position du débris

Image correspondante

Ces alertes doivent être transmises à un opérateur ou affichées sur une interface de gestion. Un système de gestion des alertes doit permettre de visualiser, trier et exploiter les événements enregistrés.

SF-06 : Enregistrement

Le robot doit sauvegarder automatiquement, en local, toutes les données liées aux détections effectuées (images, coordonnées, probabilité). Cela permettra de conserver un historique exploitable des opérations d'inspection, permettant une analyse a posteriori, une traçabilité et une relecture en cas de besoin, conformément aux directives de l'OACI.

1.2.3 Spécifications techniques

ST-01 : Matériel

Le robot est conçu autour d'une architecture embarquée intégrant deux unités principales :

Une unité dédiée à la gestion de la mobilité, à la navigation autonome et à l'exploitation des capteurs de base.

Une unité dédiée au traitement de données complexes telles que l'analyse d'images, la détection de débris et la communication avec l'interface utilisateur.
Les principaux sous-systèmes matériels comprennent :

Un système de propulsion motorisé adapté.

Des capteurs d'orientation et de mouvement pour assurer le maintien du cap et la correction de trajectoire.

Des capteurs de proximité permettant la détection d'obstacles pour garantir la sécurité de navigation.

Un module de capture d'images assurant l'acquisition visuelle de la piste.

Un système de communication sans fil pour la transmission des données vers un dispositif externe.
Tous les éléments matériels sont issus de technologies facilement accessibles et modulables, permettant une réduction des coûts et une évolutivité simplifiée du système.

ST-02 : Autonomie

L'autonomie du robot est un critère central pour assurer des missions de surveillance efficaces sur des pistes de grandes dimensions. Elle dépend de l'efficacité énergétique du système embarqué, du poids total du robot, et du choix des sources d'alimentation.
Le système d'alimentation doit être conçu pour assurer un fonctionnement autonome sans dépendance au réseau électrique :

Un bloc d'alimentation indépendant dédié aux moteurs et à l'unité de commande de navigation.

Un second bloc d'alimentation dédié à l'unité de traitement et de communication, isolé pour éviter toute instabilité liée aux fluctuations de puissance.
Cette séparation permet d'éviter les interférences entre les circuits de puissance et les circuits logiques.

Autonomie estimée :
De 1 à 2 heures selon l'intensité des déplacements et la charge du traitement embarqué.
Système de recharge rapide et simple.
Conception favorisant la stabilité électrique globale du système.

ST-03 : Interface

L'interface utilisateur est déployée sur un poste externe, accessible via une connexion sans fil sécurisée. Elle remplit plusieurs fonctions clés :

Affichage des images capturées, en mettant en évidence celles contenant des éléments considérés comme des débris.

Localisation des objets détectés sur une carte simplifiée de la zone surveillée.

Historisation des événements avec possibilité de consulter les données enregistrées (images, coordonnées, horodatage).

Interface intuitive, accessible depuis un navigateur, ne nécessitant pas de compétences techniques avancées de la part de l'utilisateur.

1.2.4 Contraintes

Il est essentiel, dans l'utilisation du robot, de prendre en compte certaines contraintes éventuelles, susceptibles d'entraver le respect des spécifications techniques et fonctionnelles du cahier des charges :

- Fragilité des capteurs ultrasons ou de la caméra face à l'humidité, poussière ou aux projections (piste mouillée, hydrocarbures).
- Fiabilité de la connexion Wi-Fi : Latence réseau lors de l'envoi des images ou alertes.
- Une mauvaise luminosité peut dégrader la qualité des images.
- Flou de mouvement si le robot roule trop vite sans stabilisation mécanique ou logicielle.
- Difficile d'obtenir une précision $> 85\%$ sans base de données adaptée pour l'apprentissage.

1.2.5 Livrables attendus

Prototype fonctionnel : Le prototype fonctionnel attendu doit démontrer la faisabilité technique et opérationnelle du système développé. Il inclura :
Un robot mobile autonome capable de se déplacer dans un environnement simulé (piste test) en évitant les obstacles.
Un système de détection des FOD basé sur un modèle de vision par ordinateur.

Rapport final : Le rapport final devra documenter l'ensemble du projet selon une démarche rigoureuse.

Chapitre 2

Etat de l'art : Systèmes de Détection des FOD déjà existants

Une bonne connaissance des débris est essentielle à la réussite de tout programme FOD, mais leur détection reste encore l'une des plus critiques des opérations d'un aéroport. Ce processus implique non seulement l'identification des causes et des emplacements potentiels des FOD, mais aussi la détection rapide de tout FOD sur les surfaces de l'aéroport. Que la détection soit effectuée manuellement, par le biais d'inspections régulières, à la suite de rapports de pilotes ou grâce à l'utilisation de technologies de détection avancées, le résultat est tout aussi important.
Si l'emplacement ou les caractéristiques du FOD ne présentent aucun danger immédiat pour la sécurité, l'objet doit être retiré dès que le calendrier opérationnel le permet. Si par contre il présente un danger immédiat pour la sécurité, les dispositions du programme de gestion des FOD doivent clairement indiquer l'existence d'un danger et permettre au superviseur de l'aéroport de prendre des mesures et de cesser temporairement les opérations. Si le FOD provient d'un aéronef ou d'un équipement aéroportuaire, d'en informer l'exploitant. Il convient, par exemple, que le comité FOD d'un aéroport étudie cette question et fournisse des orientations supplémentaires au personnel de gestion et d'exploitation de l'aéroport.

Les récentes avancées technologiques ont considérablement accru les capacités de détection des FOD grâce à l'automatisation. Des technologies avancées permettent désormais d'améliorer la détection des FOD, notamment la détection continue sur les pistes et autres aires de mouvement des avions, ainsi que des dispositifs de détection mobiles pour compléter les capacités du personnel aéroportuaire. Cependant, si un aéroport choisit de mettre en œuvre ces nouvelles technologies de détection des FOD, il doit s'assurer que le personnel chargé de la surveillance de ces systèmes dispose de l'autorité (ou de la capacité de contacter rapidement les personnes compétentes) pour prendre les mesures appropriées et rapides en cas de détection de FOD.

Il existe plusieurs systèmes et techniques de détection des FODs utilisés dans les aéroports à travers le monde, certains plus simples et plus accessibles, d'autres plus sophistiqués et onéreux.

2.1 Les systèmes de détection actuellement utilisés

2.1.1 Méthodes traditionnelles de détection

Une inspection de piste implique un passage sur toute sa longueur afin d'observer et d'éliminer les FOD. La méthode la plus efficace consiste à effectuer deux passages ou plus afin de réduire la largeur de la zone d'inspection. Lorsqu'il est possible de n'effectuer qu'un seul passage sur la piste, le personnel d'inspection doit, dans la mesure du possible, circuler dans le sens inverse de l'atterrissage des avions, en utilisant un gyrophare à haute intensité et des phares allumés en permanence. Cette pratique permettra au personnel d'auto-inspection de voir, au cas où, les avions en approche et d'améliorer la visibilité du véhicule pour les pilotes.

FIGURE 2.1 – inspection visuelle de la piste 24 de l'aéroport d'orly

Les inspections sont assurées par des agents circulant à bord d'un véhicule, et la sécurité impose que le trafic aérien soit suspendu le temps de l'intervention. A titre d'illustration, sur l'aérodrome de Paris Charles de Gaulle, la durée d'une inspection est de l'ordre de 10 à 15 minutes pendant lesquelles aucun aéronef ne peut atterrir ou décoller.

FIGURE 2.2 – Elimination manuelle des FODs (Equipe Airside Operation)

Les inspections sont visuelles et l'efficacité des inspections dépend de la visibilité au moment de l'inspection, un facteur supplémentaire à prendre en compte concerne les inspections qui peuvent être annulées pendant les opérations aériennes en condition de basse visibilité.
Les aéroports organisent souvent des opérations appelées "Fod Walk" impliquant plusieurs personnes, alignées sur la piste à la recherche d'éventuels FOD. Ces opérations sont plus courantes lors des travaux, ou après des renovations, mais surtout pas regulières car très lentes, très couteuses en personnel, et nécessite une fermeture de la piste pendant un long moment.

FIGURE 2.3 – FOD Prevention de EASA Safety Promotion à l'aéroport de Dusseldorf

Cette technique est plutôt courante chez les militaires et sur les portes-avions.

FIGURE 2.4 – Fod Walk des militaires : opération de dépollution de la piste

2.1.2 Technologies fixes de surveillance

Le système QinetiQ (Tarsier)

Le système Tarsier, développé par la société QinetiQ (Royaume Uni), a été spécifiquement conçu pour la détection d'objet sur les chaussées aéronautiques. Le système Tarsier s'appuie sur la technologie radar. Il repose sur une détection active utilisant un radar primaire. Le mode de détection utilisé est la détection à partir d'ondes millimétriques. Le détecteur utilisé est un radar FMCW (Frequency Modulated Continuous Waves) dont la fréquence centrale est à 94,5 GHz, de largeur de bande 600 MHz, à double antenne. L'unité de détection est placée sur une plate-forme au sommet d'un pylône en treillis (tour), à une hauteur s'échelonnant entre 7 et 9 m en fonction de la topographie du terrain d'implantation et de celle de la piste. Les détecteurs sont placés à égale distance les uns des autres parallèlement à l'axe de la piste qu'ils surveillent, et éloignés d'une distance approximative de 200 à 250 m de celle-ci. Un détecteur peut porter à plus de 1 km (toutefois, le fabricant recommande que chaque point de la zone surveillée par un radar soit situé à une distance inférieure à 1 km), rendant généralement l'installation de deux têtes suffisante. Néanmoins pour améliorer la détection, un nombre plus important de têtes radar peut être jmplanté.

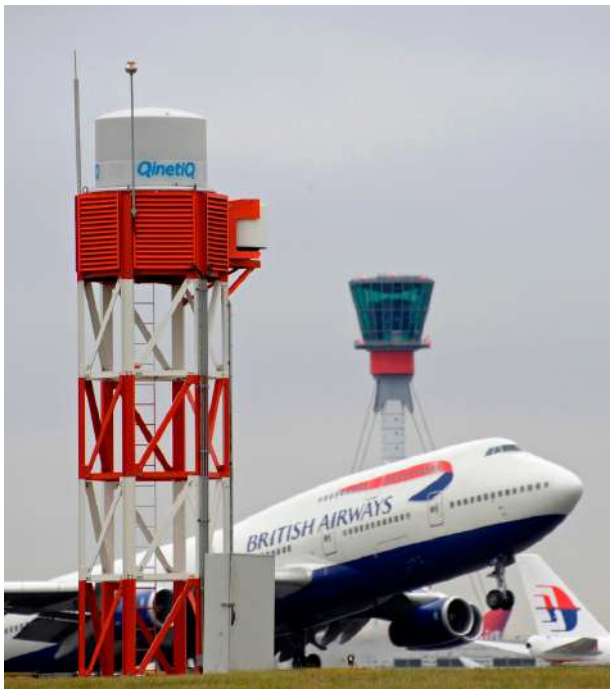


FIGURE 2.5 – Système QinetiQ installé à l'aéroport de Heathrow au Royaume Uni

Aucun prix constructeur explicite n'est publié par QinetiQ, car les installations sont sur devis, adaptées aux infrastructures de chaque aéroport. Cependant, il a un coût estimé entre 1 et 5M$ par piste, dépendant des options choisies.

Le système Stratech (iFerret)

La technique de détection du système iFerret fait référence au domaine optique du spectre électromagnétique. Dans ce domaine, la technique utilisée pour détecter des objets est l'analyse d'images vidéo. La détection est dite passive ; la source du rayonnement reçu par la caméra vidéo est la lumière naturelle réfléchie à la surface des objets. Le pylône est conçu pour limiter les risques de vibration de la caméra (dues notamment au vent : il résiste à un vent de 120 km/h). Les détecteurs sont implantés à une distance égale les uns des autres d'environ 350 mètres, et à une distance d'environ 250 mètres de l'axe de la piste. Le pylône et le détecteur sont ainsi placés sous les surfaces de dégagement de la piste.
Le système localise, catégorise et mesure la taille des objets détectés en temps réel, avec une fiabilité rapportée supérieure à 95%. Il a une architecture modulaire et évolutive avec une armoire de tours optiques installées le long des zones à surveiller. Chaque module peut être ajouté pour couvrir des zones plus larges (piste, taxiway, aire de stationnement). Pour une configuration type, il y a 12 caméras pour un tour complet (piste + taxiway).

FIGURE 2.6 – Disposition des détecteurs iFerret le long de la piste

Le système est installé dans des aéroports comme Hong Kong International Airport (HKIA) avec un contrat global (conception, fourniture, installation, maintenance 60 mois, option 24 mois) d'environ 36,9 millions $, ou à l'aéroport de Changi (Singapour) où un déploiement initial sur deux pistes a nécessité un contrat de l'ordre de 12 millions $.

Le système iFerret se positionne comme un équipement premium de détection FOD automatisée, conçu pour les grands aéroports mondiaux. Le coût élevé de l'installation s'accompagne d'une maintenance longue durée, ce qui le rend adapté aux infrastructures ayant besoin d'une solution de surveillance continue et fiable.

Le système Xsight (FODetect)

Le système est composé de détecteurs couplés radar et optiques, l'ensemble a une hauteur de 13 pouces (soit environ 33 cm). La caméra est de type CCD travaillant dans le domaine optique et dans le proche infrarouge avec une capacité de zoom. Le radar est de type FMCW (Frequency Modulated Continuous Waves) dont la fréquence centrale est 76,5 GHz.

FIGURE 2.7 – Détecteurs Xsight

Il fournit une couverture à 360° le long de la piste, sans angles morts, avec un temps de scan inférieur à 60 s. Cependant, il requiert une maintenance (1 h au plus) tous les 15 jours, et le prix du système complet varie entre 1,7 M$ et 4,6 M$.

2.1.3 Systèmes mobiles de détection des FODs

Trex Enterprises FODFinder

Utilisé dans des aéroports comme Boston Logan, c'est un système basé sur un radar à ondes millimétriques (78 à 81 GHz), couplé à un module d'imagerie monté sur véhicule, pour détecter des débris de petite taille (dès 2 cm) à longue portée (jusqu'à 400 m). Il fournit les coordonnées GPS des FODs.
En plus d'avoir une portée longue, il est efficace dans le brouillard, la pluie ou la neige. Cependant, il est moins performant pour les débris non réfléchissants (plastique, tissu) sans capteurs optiques. Le prix estimé est de 250000 à 500000 dollars par unité, selon les options (caméra, intégration logicielle). Les coûts varient en fonction de la personnalisation et des contrats de maintenance.

FIGURE 2.8 – Véhicule fod finder de Trex Enterprises

Smart Airport FOD Detection Rover

Ce sont des systèmes LiDAR mobiles (montés sur rovers ou véhicules) utilisant des lasers pour créer des cartes 3D des pistes, détectant les FOD par des variations de surface. On les retrouve surtout dans les aéroports de petite taille, pour des inspections périodiques.
Ils sont précis pour les petits objets, fonctionnent bien de nuit, avec un coût initial bas pour les prototypes (10 000 à 50 000 dollars), mais les versions commerciales pourraient coûter entre 100 000 et 200 000 dollars avec intégration logicielle et robustesse accrue. La portée est limitée (comparée au radar), et le traitement des données complexe.

IEEE Drone-Based FOD Detection System

Il est basé sur des drones avec IA et vision par ordinateur. Des drones équipés de caméras haute résolution et d'algorithmes d'IA (ex. : YOLOv8, Masked Auto-Encoder) pour détecter et classer les FODs survolent les pistes et transmettent des données en temps réel.
Ce système est pratique pour des aéroports sans infrastructure fixe, et pour des inspections d'urgence. Le bémol est qu'il a une forte sensibilité aux conditions météorologiques (vent, pluie), et nécessite des pilotes ou une automatisation avancée.

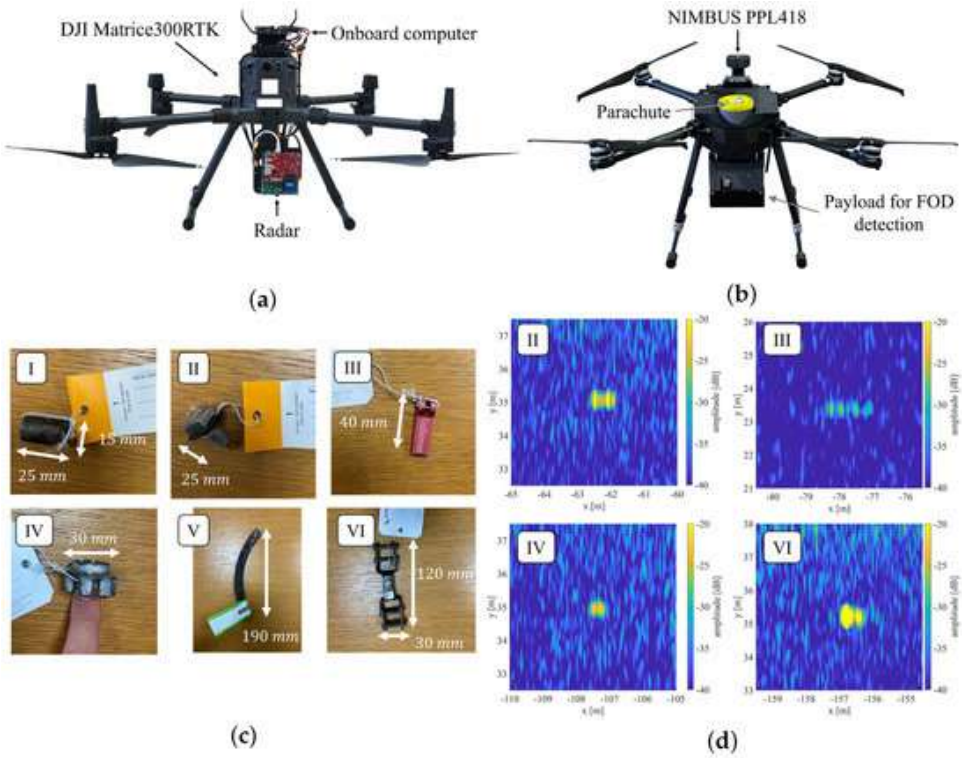


FIGURE 2.9 – drone de détection des FODs

2.1.4 Méthodes d'élimination des FODs

Une fois les FOD détectés, l'étape suivante majeure d'un programme de gestion des FOD est l'élimination des FOD de l'environnement aéroportuaire. La solution la plus efficace pour éliminer les FOD est l'utilisation d'équipements de retrait, en particulier dans les zones où des FOD sont susceptibles d'être détectés, comme à proximité de chantiers. Pour l'élimination d'un FOD isolé détecté sur une piste, l'élimination manuelle est la plus efficace bien que des équipements spécialisés soient disponibles et parfaitement adaptés à certaines opérations aéroportuaires, des méthodes et technologies de retrait des FOD sont disponibles pour les aéroports.
Ces techniques, les unes meilleures et/ou plus recommandables que les autres incluent des Balayeuses électriques, y compris les remorques à poils remorqués ; des balayeuses à tapis de friction ; des souffleurs d'airs ; des systèmes de vide ; des Barres magnétiques (fixées aux véhicules) ; des bandes rugueuses (également appelées "FOD Shakers").

FIGURE 2.10 – Une balayeuse d'élimination de débris

2.2 Limites des systèmes de détection actuels

2.2.1 Performance et flexibilité

L'efficacité des inspections visuelles est fortement réduite. Un seul passage sur la piste peut manquer des débris, surtout sur des zones larges. Même avec plusieurs passages, certaines zones intermédiaires (ex. : voies de circulation) sont souvent négligées, d'autant

plus que la détection dépend de l'attention et de la compétence des inspecteurs, ce qui introduit une variabilité dans les résultats. De plus, les inspections nécessitent la suspension du trafic aérien (ex : 10 à 15 minutes à Paris Charles de Gaulle), entraînant une perte de capacité, particulièrement critique dans les aéroports à fort trafic.

Les systèmes à radar à ondes millimétriques sont moins efficaces pour détecter des débris non métalliques. Les systèmes fixes eux sont des détecteurs statiques, limitant leur flexibilité pour surveiller des zones spécifiques ou temporaires (ex. : taxiways secondaires).

Les systèmes mobiles sont rares, coûteux et souvent encombrants. Les systèmes de détection de FOD (Foreign Object Debris) les plus avancés à ce jour, tels que FODFinder MK2 de Trex Enterprises, FODetect d'Xsight Systems ou Tarsier de QinetiQ, offrent une couverture automatisée des pistes et des aires de roulage. Toutefois, leur performance reste limitée, car les objets non métalliques (comme le plastique ou le bois) sont moins bien détectés, bien que la FAA rapporte que ces objets représentent près de 30% des débris responsables d'incidents.

En termes de flexibilité, la plupart de ces systèmes sont installés de manière fixe en bordure de piste. Ils nécessitent des infrastructures spécifiques (mâts, alimentations, réseaux de communication), ce qui limite leur déploiement à d'autres zones telles que les parkings avions, taxiways éloignés ou zones de maintenance. Le système Tarsier, par exemple, exige l'installation de radars tous les 300 mètres, ce qui rigidifie son implantation. Dans le cas d'une extension de piste ou de reconfiguration d'un aéroport, ces systèmes doivent être entièrement réinstallés ou reprogrammés, ce qui augmente leur inertie opérationnelle. Leur capacité à fonctionner de manière autonome sur de longues distances sans infrastructure fixe est également limitée, rendant leur usage difficile dans les aéroports régionaux ou militaires en zones isolées.

2.2.2 Coûts d'installation et d'exploitation

Les coûts liés aux systèmes actuels de détection de FOD représentent un frein majeur à leur généralisation, notamment pour les petits et moyens aéroports. À titre d'exemple, le système FODFinder MK2 de Trex Enterprises est estimé entre 1,2 et 1,5 million USD pour équiper une seule piste de 3 000 mètres. Ce coût n'inclut pas les infrastructures nécessaires (fondations, alimentations électriques, réseaux de données), qui peuvent ajouter 300 000 à 500 000 USD supplémentaires selon la configuration de l'aéroport.

À cela s'ajoutent les coûts d'exploitation annuels, qui comprennent la maintenance préventive et corrective, le remplacement de pièces sensibles (lentilles optiques, cartes électroniques), les mises à jour logicielles et la surveillance technique. Ces coûts peuvent atteindre 80 000 à 120 000 USD par an pour des systèmes comme Tarsier ou FODetect. Par ailleurs, la consommation électrique de ces systèmes varie entre 1,5 et 2,5 kWh par

heure, ce qui génère des charges supplémentaires, surtout pour les aéroports en zones non interconnectées au réseau national. Il faut aussi tenir compte de la nécessité d'un personnel spécialisé pour l'interprétation des alertes et le maintien opérationnel du système.

Enfin, l'absence de modularité et de mutualisation des composants rend leur évolution coûteuse. Par exemple, l'ajout d'un second système mobile pour les taxiways nécessite souvent l'achat d'un nouveau système complet, sans possibilité de réutilisation partielle. Ces éléments rendent ces solutions peu accessibles pour plus de 80% des aéroports secondaires dans les pays en développement, alors même que ceux-ci sont aussi confrontés au risque FOD.

Chapitre 3

Conception et développement de l'Architecture de notre système Robot FODEX

3.1 Conception du Robot

3.1.1 Positionnement du projet

Problème ciblé

Il n'existe pas encore des systèmes avancés de détection de FOD dans les aéroports marocains (comme Casablanca Mohammed V, Rabat-Salé, Marrakech-Menara...). Du moins, il n'en existe aucune mention explicite dans les rapports publics de l'ONDA. Les inspections manuelles ou visuelles avec patrouilles régulières sont encore la norme. Dans un contexte où le secteur aéronautique est en pleine expansion au Maroc, et le pays se préparant à accueillir la coupe du Monde 2030 qui va sans doute multiplier les vols vers le pays et l'exploitation des infrastructures aéroporturaires, il est très important de se pencher sur les questions de sécurité.
Au delà du Maroc, le problème reste le même dans la plupart des petits et moyens aéroports à travers le monde.
Les FOD par nature sont très discrets, mais leurs effets lorsqu'ils se manifestent sont très énormes. Le prototype que nous proposons est un système de détection FOD qui présente de nombreux avantages par rapport aux technologies existantes, surtout financièrement parlant. Nous reviendrons sur plus de détails de comparaison dans la section 5.3.4.

Approche proposée pour la conception

Dans un programme FOD en aéronautique, quatre (04) dimensions cruciales sont à considérer : **la détection, l'élimination, l'évaluation et la prévention.** C'est un cycle

qui permet de diminuer au maximum l'occurrence des FODs sur les pistes de circulation.

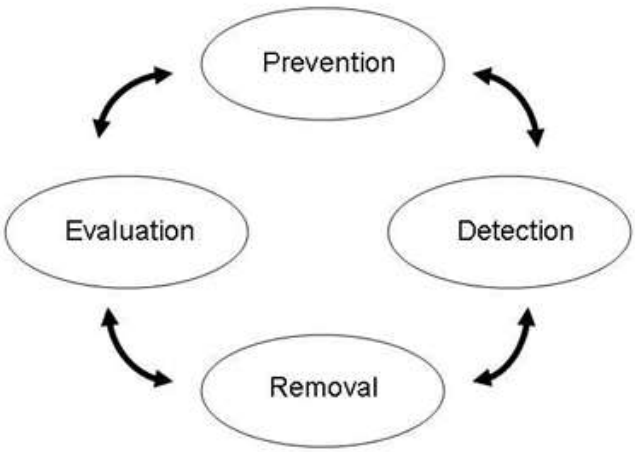


FIGURE 3.1 – Les quatre (04) principaux axes d'un programme FOD en aéronautique

Dans ce diagramme, notre robot FODEX couvre principalement la partie détection, et offre en plus, après chaque opération, des informations cruciales pouvant former une base de données solide pour la prévention.

Nous avons développé un prototype de robot mobile à quatre roues motrices adapté à des pistes aux dimensions prédéfinies.
Le robot navigue de manière autonome, cartographie la piste et localise précisément les débris détectés. Les données collectées sont transmises à une interface web, accessible par l'équipe d'intervention permettant une réponse rapide et ciblée pour l'élimination des débris, réduisant ainsi les risques pour la sécurité aérienne et minimisant les interruptions des opérations aéroportuaires.
Il présente nettement des avantages supérieurs aux méthodes visuelles d'inspection, lesquels sont résumés dans le tableau ci-dessous :

TABLE 3.1 – Avantages des systèmes comme FODX par rapport aux inspections manuelles

Critère	**Inspection visuelle (manuelle)**	**Système robotisé comme FODX**
Fréquence d'inspection	1 à 2 fois par jour (souvent limitée)	Possibilité d'inspection fréquente ou continue
Fiabilité	Dépend de l'attention humaine, risque d'oubli ou de fatigue	Détection systématique et régulière grâce à l'IA
Taux de détection	70–80 % (peut descendre selon conditions)	Jusqu'à 90–95 % sur les objets courants
Traçabilité des inspections	Très limitée, peu de preuves enregistrées	Enregistrement des détections (images, temps, position)
Accessibilité pour petits aéroports	Main d'œuvre nécessaire, résultats variables	Solution économique, autonome, déployable localement
Équipements requis	Véhicule, personnel formé, communication sol	Arduino, Raspberry Pi, caméra et capteurs simples
Risque pour les agents	Présence humaine sur la piste (danger)	Aucun, inspection totalement autonome
Portée de la solution	Détection, évaluation et retrait faits simultanément par les agents	Notre système assure la détection et la collecte de données pertinentes ; l'évaluation et le retrait sont laissés aux équipes aéroportuaires compétentes.

Conformité du projet avec le cadre réglementaire (OACI)

Conformément à l'appendice 03 au Chapitre 05 du document 9981 publié par l'OACI, notre système avertit l'équipe au sol une fois des débris détectés, afin de leur permettre de prendre les dispositions nécessaires pour une élimination rapide et efficace.
Par ailleurs, la carte que nous proposons constitue une base solide pour faciliter l'enregistrement des endroits où les FODs ont été trouvés. Cela permettra de savoir les zones à risque dans un aéroport donné.

3.2 Architecture du Système

3.2.1 Architecture globale

L'architecture globale du système de détection des FOD repose sur une conception modulaire et intégrée, combinant des composants matériels et logiciels pour assurer une

détection autonome des débris sur une piste pour répondre aux exigences de sécurité aéronautique.

Le châssis robuste, alimenté par un pack de batteries, garantit une mobilité stable sur les surfaces de piste. Le robot est équipé de deux capteurs principaux : un capteur à ultrasons pour la détection d'obstacles et une caméra pour la capture d'images destinées à l'analyse visuelle des débris. Ces capteurs sont orchestrés par deux microcontrôleurs complémentaires : Arduino pour la gestion des entrées/sorties et Raspberry Pi5 pour le traitement des données complexes et la communication réseau.

Les données collectées sont transmises à une interface web via une connexion Wi-Fi. Cette interface affiche une carte visuelle dynamique de la piste, indiquant les positions exactes des débris détectés. L'interface permet aux équipes d'intervention de localiser rapidement les FOD, minimisant les interruptions des opérations aéroportuaires.

L'architecture globale est conçue pour être modulaire, permettant des mises à jour ou des remplacements de composants sans modifier l'ensemble du système. Cette modularité, combinée à l'utilisation de microcontrôleurs standards réduit les coûts de développement. Les tests en environnement simulé ont validé l'intégration des composants.

Cette architecture a été modélisée pour illustrer les interactions entre le robot et ses composants, et fournir une vue formelle des flux de données et des comportements du système, facilitant son analyse et son optimisation future.

3.2.2 Modélisation SYSML

Diagramme des cas d'utilisation (Use Case Diagram)

Dans le cadre de la modélisation SysML, un diagramme de cas d'utilisation (Use Case Diagram) décrit les interactions entre un système et ses acteurs externes (utilisateurs, autres systèmes, ou entités environnementales) en mettant en évidence les fonctionnalités principales que le système doit fournir. Pour élaborer un diagramme de cas d'utilisation, je vais supposer un système robotique générique, comme un robot mobile autonome (par exemple, un rover ou un robot aspirateur), car le contexte n'est pas précisé. Si vous avez un système spécifique en tête, merci de le préciser pour une réponse plus ciblée.

Prenons par exemple un cas typique de Détection + Notification :

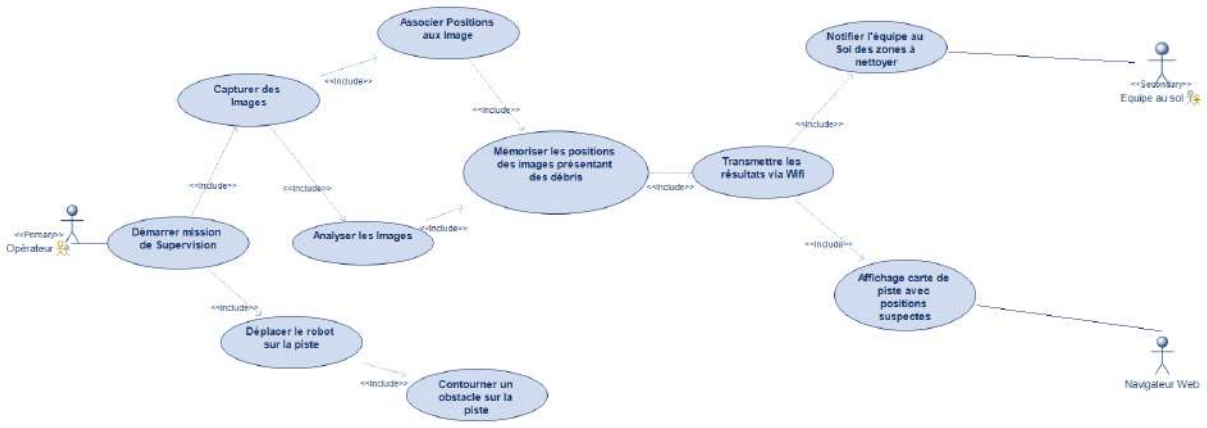


FIGURE 3.2 – Diagramme de Cas d'Utilisation

Lorsque l'opérateur lance la mission de supervision (Acteur Principal, agent habilité par l'aéroport pour cette opération : il peut être activé à distance, ce qui n'est pas encore pris en compte par notre prototype, ou manuellement) le robot démarre la mission de supervision sur la piste, connaissant déjà ses dimensions. Cela suppose donc que le robot est initialement placé en début de piste. Pendant que le robot se déplace, la caméra scanne la piste et capture des images en continu. Chaque image est associée à sa position (l'emplacement sur la piste où elle est prise) et traitée par la raspberry Pi5. Seulement les images présentant des FOD seront sauvegardées dans la mémoire et transmises via Wifi au centre de contrôle pour valoir notification. De même, les positions de ces images "polluées" seront utilisées pour la construction de la carte visuelle qui servira de guide à l'équipe au sol chargée de la dépollution de la piste.

Diagramme de définition de blocs (Block Definition Diagram – BDD)

Le diagramme de définition de blocs est utilisé pour décrire la structure statique du système, en identifiant ses composants principaux (blocs), leurs relations, et leurs propriétés. Il représente l'architecture globale du système sous forme hiérarchique, en mettant en évidence les sous-systèmes, leurs interfaces, et leurs attributs.

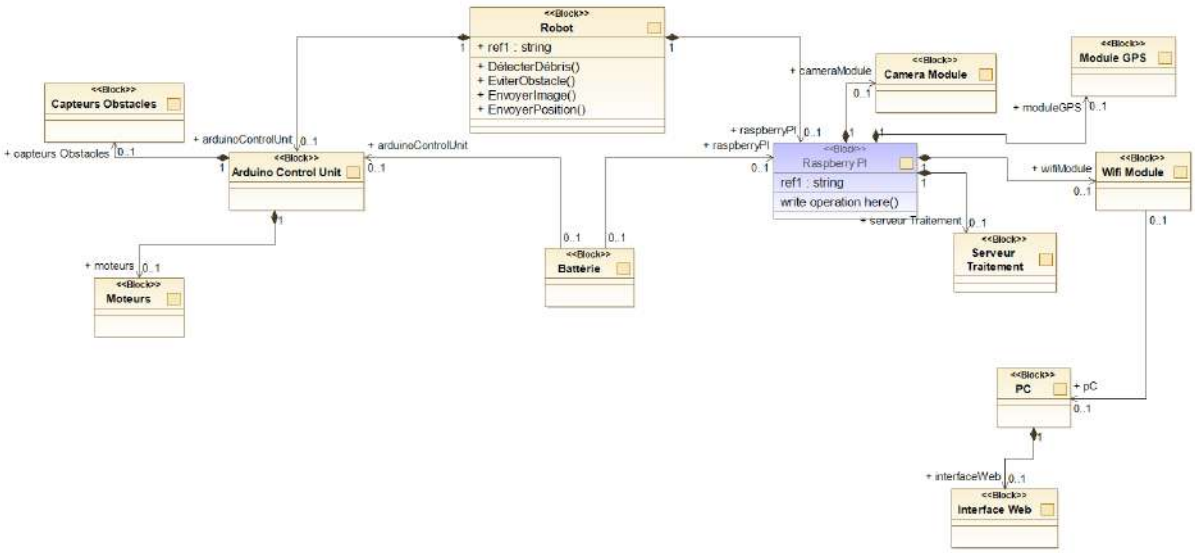


FIGURE 3.3 – Diagramme de Définition de Blocs

Il y a onze (11) blocs principaux :

— **Arduino Control Unit :** C'est le microncontroleur en charge du bon déplacement du robot sur la piste. Il est solidaire au robot, et interagit directement avec les capteurs d'obstacles et les moteurs.

— **Capteurs Obstacles :** Il s'agit de tous les composants qui permettent au robot d'avoir des informations de l'environnement extérieur pour son déplacement. Ce module inclut aussi bien le MPU6050, le LM393 et le HC-SR04.

— **Moteurs :** Ce sont les 04 moteurs assurant le déplacement du robot. Ils sont contrôlés par l'arduino via un driver L298N.

— **Battérie :** Il s'agit du module d'alimentation du robot. Il inclut un pack de deux battéries Li-ion 3.7v alimentant le driver et l'arduino, et d'une source d'alimentation appropriée pour la raspberry Pi5.

— **Raspberry Pi5 :** C'est le coeur du traitement des données pour la détection. Il analyse les images capturées par la caméra, et communique avec l'arduino via par une communication série.

— **Camera Module :** C'est une caméra raspberry module V3 adaptée avec le serveur de traitement et offrant une très bonne résolution et un angle d'ouverture suffisant.

— **Serveur Traitement :** Il est en réalité intègre à la Raspberry. Il matérialise juste le programme d'IA chargé de la détection.

— **Wifi Module :** Il est également intégré à la raspberry, et sert de communication avec l'équipe au sol.

— **PC :** Ce module matérialise l'équipement de supervision dont dispose l'équipe au sol et qui est chargé de recevoir les données émises par la Raspberry après détection.

— **Interface Web :** Il s'agit de la page qui servira d'affichage de la carte visuelle

contenant les éventuelles positions des FODs sur la piste.

— **Robot :** Il matérialise l'ensemble du système et permet de relier les deux microncontroleurs entre eux, et d'assurer la protection des composants.

Un douzième bloc **(Module GPS)** a été précisé mais n'interviendra pas directement dans le cadre de la présentation de ce projet dû aux dimensions réduites de l'espace de démonstration et la résolution du module. A une échelle plus grande, il pourrait être intégré sans problème.

Diagramme de séquence (Sequence Diagram)

Le diagramme de séquences est utilisé pour modéliser les interactions dynamiques entre les différents composants du système ou entre le système et des acteurs externes dans un scénario spécifique. Il met en évidence l'ordre des messages ou des appels échangés, ainsi que leur chronologie, pour accomplir une fonction particulière.

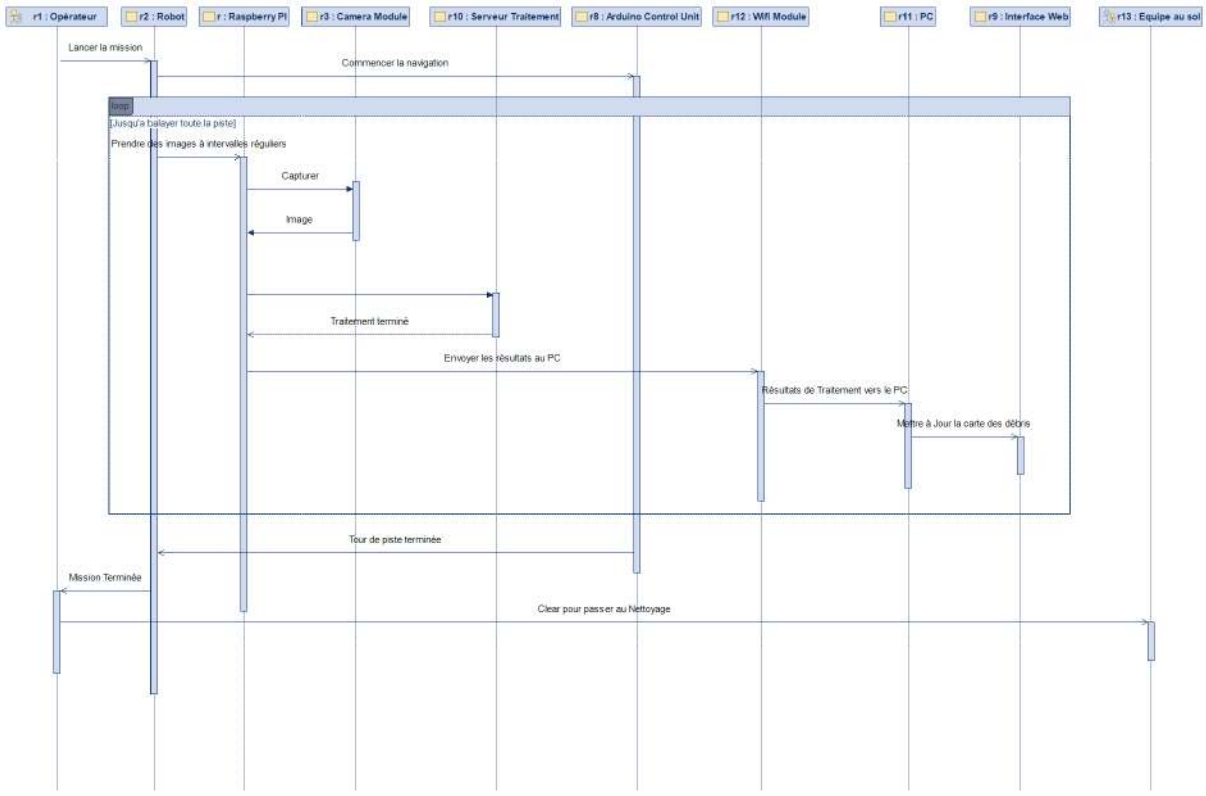


FIGURE 3.4 – Diagramme de séquence

Diagramme de machine à états (State Machine Diagram)

Le diagramme de machine à état fini décrit le comportement dynamique du système ou d'un composant en représentant les différents états qu'il peut prendre, les transitions entre ces états, et les événements ou conditions qui déclenchent ces transitions.
Chaque état correspond à un mode ou une condition spécifique du système (ex. : "En

attente", "Navigation", "Évitement d'obstacle"). Les changements d'état sont déclenchés par des événements ou des conditions.

Dans ce cas, nous présentons les différents états du robot en mode déplacement et en mode Traitement.

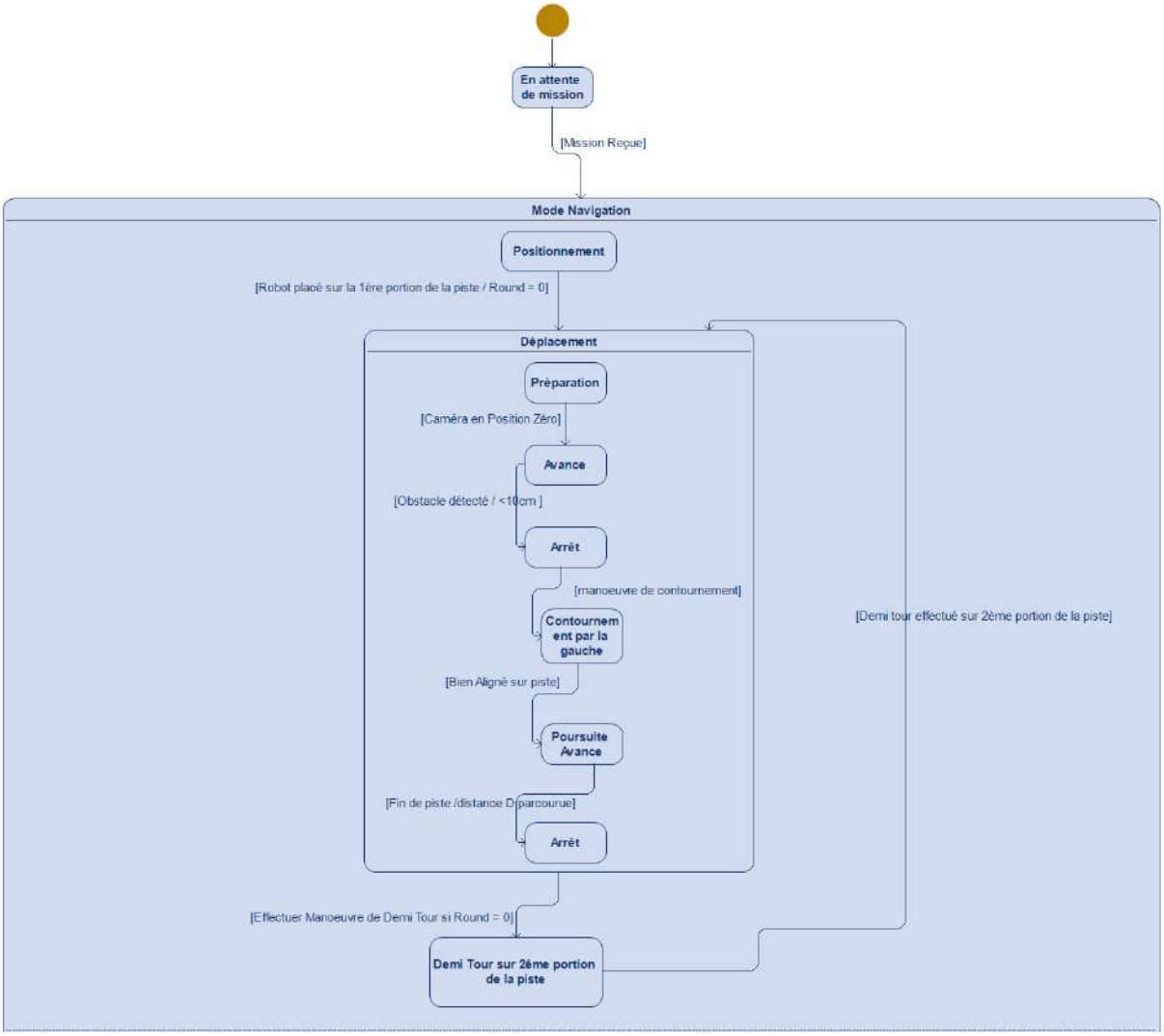


FIGURE 3.5 – Etats du robot en mode navigation

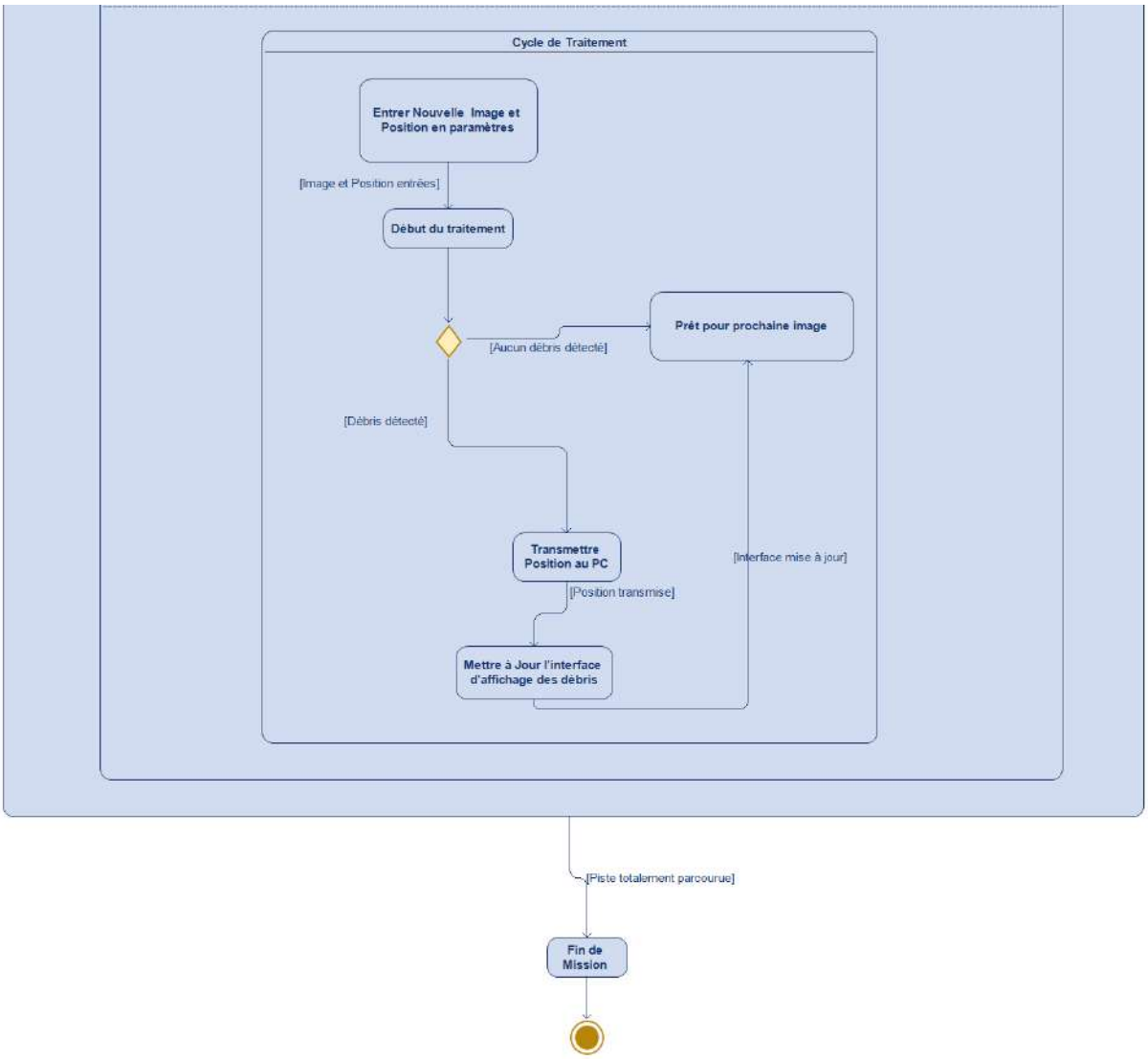


FIGURE 3.6 – Etats du robot en mode traitement

3.2.3 Simulation du Déplacement du Robot sous Matlab/Simulink

Objectif

La modélisation du déplacement du robot vise à anticiper et valider le comportement cinématique du système mobile avant la mise en œuvre physique. Elle permet de simuler les trajectoires sur une piste virtuelle, évaluer la précision de la navigation, tester les algorithmes de contrôle (PID, suiveur de ligne, évitement d'obstacles...), et réduire les erreurs potentielles lors de l'intégration réelle sur le robot.

Approche de modélisation

Le modèle du robot a été conçu selon une architecture fonctionnelle en boucle fermée, simulant l'échange d'informations entre le robot et son contrôleur. L'objectif est de représenter

fidèlement le fonctionnement réel du système embarqué, avec des interactions bidirectionnelles entre les capteurs, la logique de commande, et les actionneurs.
L'architecture est décomposée en deux blocs principaux :

Un bloc de commande qui reçoit, en entrée, un signal de mise en marche et les vitesses mesurées des roues issues des encodeurs gauche et droit. Il traite ces informations pour générer, en sortie, les signaux PWM destinés aux moteurs gauche et droit. La logique de commande inclut un régulateur permettant d'adapter dynamiquement les signaux de puissance pour maintenir le cap du robot.

Le bloc de robot reçoit les PWM en entrée, qu'il convertit en vitesses de rotation des roues à l'aide d'un modèle simplifié de moteur à courant continu. Ces vitesses sont ensuite utilisées pour simuler le déplacement global du robot dans l'espace, en calculant sa position et son orientation en fonction du temps. En sortie, les vitesses mesurées sont renvoyées vers le bloc de commande pour former une boucle de régulation.

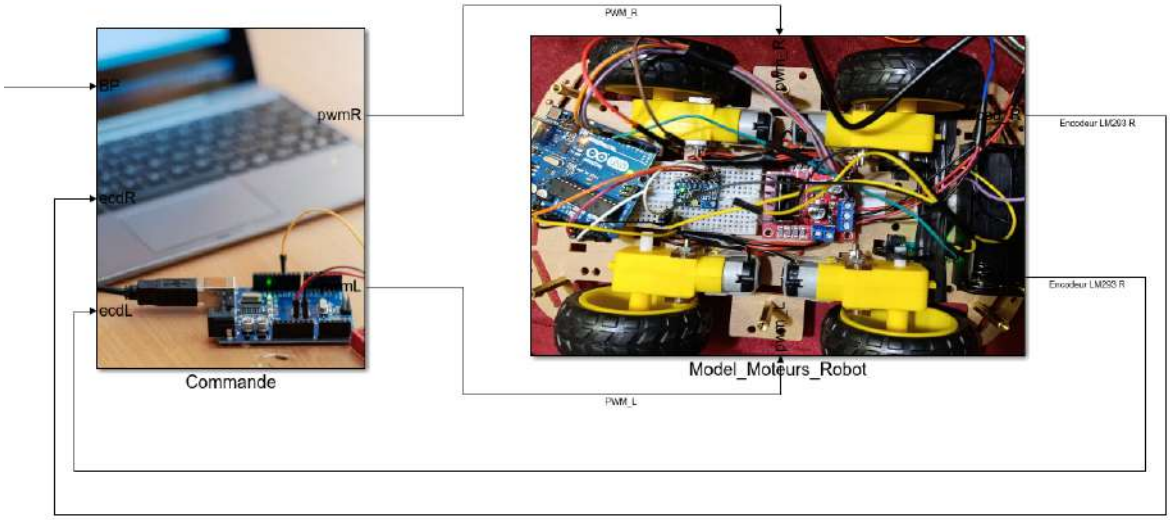


FIGURE 3.7 – Modèle de Simulation Matlab

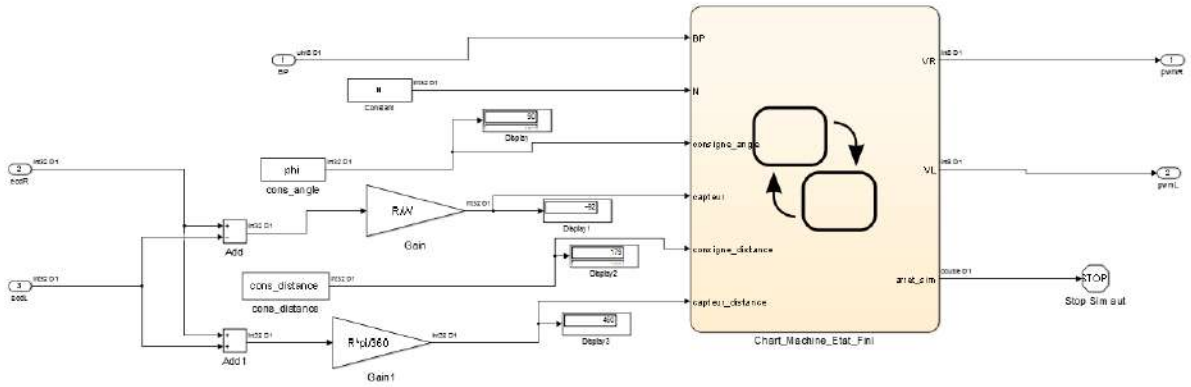


FIGURE 3.8 – Bloc de Commande

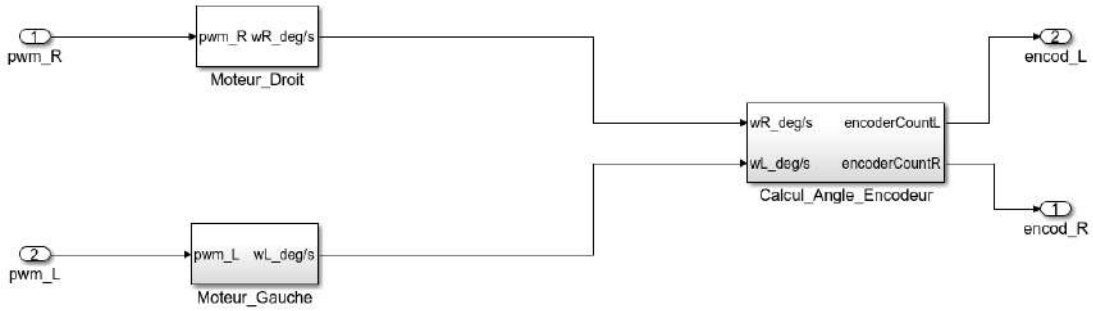


FIGURE 3.9 – Modèle utilisé pour les roues motrices et leurs signaux de commande

Ce modèle en boucle fermée permet de tester le comportement du robot sur différentes trajectoires, de valider les réglages du contrôleur, et de prévoir les réactions du système à des perturbations ou erreurs de cap. Il constitue une étape clé dans la transition vers l'implémentation physique sur le robot réel.

Environnement MATLAB/Simulink

La simulation a été développée dans MATLAB, avec l'utilisation de Simulink et Stateflow.
Une interface de sélection de trajectoire (par clic souris) permet de définir la piste. En rappel, les dimensions de la piste sont connues d'avance dans le cas réel, cette approche vise donc à donner au robot les informations de la piste sur laquelle l'inspection aura lieu. La simulation prend en compte l'affichage de la trajectoire réelle vs trajectoire de consigne, la simulation des changements d'orientation (courbes, virages).

3.2.4 Résultats de simulation

Cas 1 : Aucun obstacle empêchant le déplacement

Pour l'inspection, nous avons divisé la piste prototype en deux. Le robot fait un aller retour où il scanne la portion de piste sur laquelle il se déplace. La largeur est estimée en fonction de l'angle d'ouverture de la caméra.

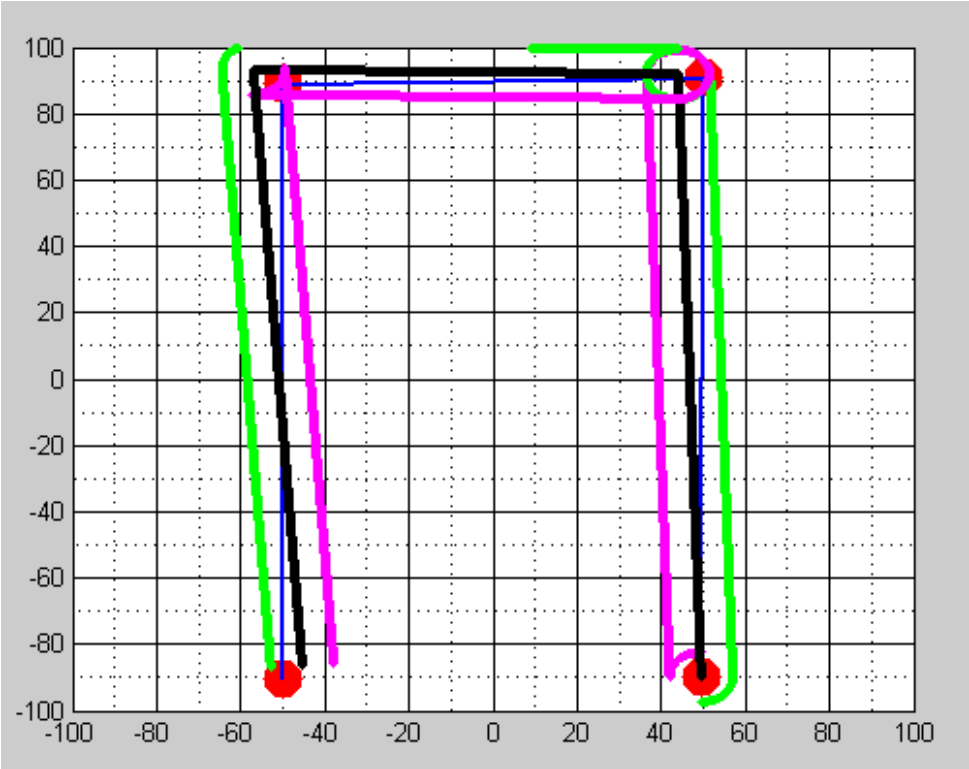


FIGURE 3.10 – Déplacement aller retour sans obstacle sur le chemin

Le robot est initialement placé au début de la piste.
Il commence l'inspection dans la première moitié de la piste. Lorsqu'il a parcouru la distance consigne (correspondant à la longueur de la piste qui lui est fournie), il effectue une manoeuvre de demi-tour pour se replacer sur la deuxième moitié de la piste. De là, il continue la navigation jusqu'à atteindre la fin de la piste avant de sortir par la voie de circulation.

Cas 2 : Présence d'un obstacle sur le chemin

Attention : Un débris sur la piste n'est pas nécessairement un obstacle pour le robot. L'obstacle représente juste un débris, qui, de par sa taille particulière, empêcherait l'avancée du robot.

Si un obstacle est détecté, le robot effectue une manoeuvre d'évitement rectangulaire avant de continuer son chemin. Il le contourne toujours pas la gauche, où il a plus de

marge de mannoeuvre. Dans l'exemple ci-dessous, les obstacles simulés sont marqués en points noirs.

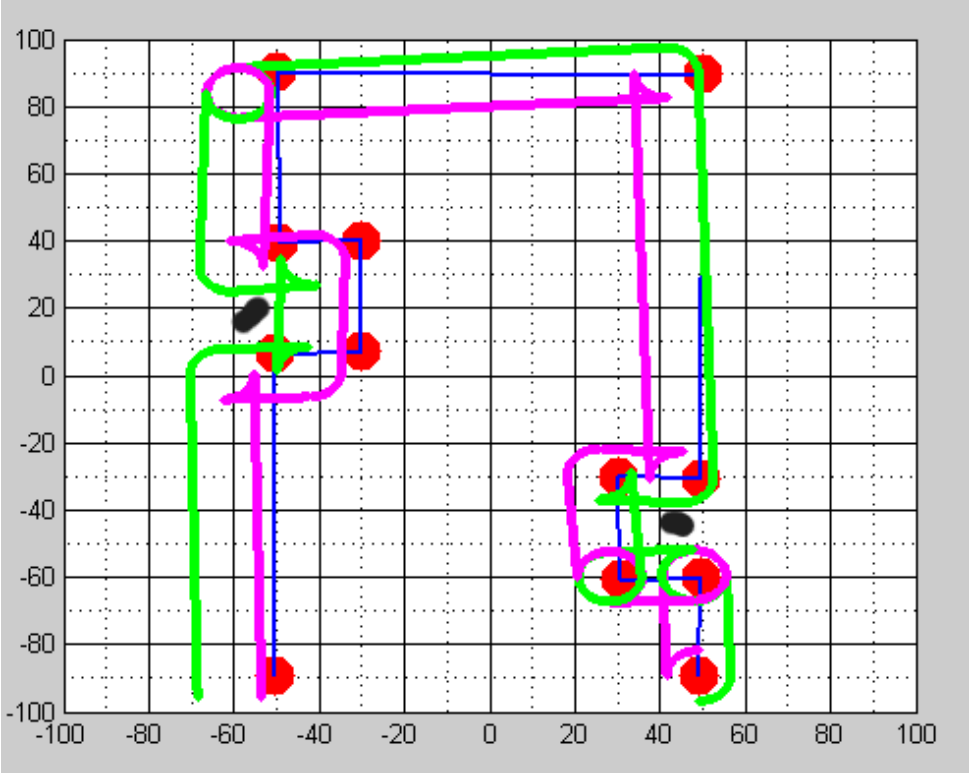


FIGURE 3.11 – Déplacement aller retour avec obstacles (en noir) sur le chemin

Chapitre 4

Technologies utilisées dans la conception du système

4.1 Objectifs du Chapitre

Ce chapitre vise à décrire les technologies matérielles et logicielles utilisées dans le prototype de robot pour la détection des FODs sur une piste. Il explique leur rôle dans la détection, la navigation, et la cartographie des débris, tout en justifiant leur choix par rapport aux besoins du projet (sécurité, coût, efficacité).

4.2 Composants matériels

4.2.1 Microcontroleurs

Il s'agit d'un système embarqué double plateforme. Il est muni de deux (02) microcontroleurs distincts assignés à des tâches de natures différentes, à savoir Arduino Uno et Raspberry Pi 5.

L'Arduino Uno est un microcontrôleur open-source basé sur le microprocesseur ATmega328P. Il est largement utilisé pour des projets d'électronique et de robotique en raison de sa simplicité, de sa robustesse, et de sa communauté active. Ses caractéristiques sont resumées dans le tableau ci-dessous :

TABLE 4.1 – Caractéristiques techniques de la carte Arduino Uno

Paramètre	**Valeur / Description**
Microcontrôleur	ATmega328P
Tension de fonctionnement	5 V
Tension d'alimentation recommandée (via jack ou USB)	7 – 12 V
Nombre de broches d'E/S numériques	14 (dont 6 utilisables en PWM)
Entrées analogiques	6
Courant max par broche E/S	20 mA
Mémoire flash	32 KB (dont 0.5 KB utilisés par le bootloader)
SRAM	2 KB
EEPROM	1 KB
Vitesse d'horloge (fréquence)	16 MHz
Connectivité USB	Type-B, pour alimentation et programmation
Dimensions physiques	68.6 mm × 53.4 mm
Poids approximatif	25 g

Sur le plan technique, l'Arduino Uno offre des performances largement suffisantes pour le contrôle des moteurs, la lecture de capteurs, et la communication série avec la Raspberry Pi. Cette répartition des tâches permet de déléguer les fonctions de bas niveau à l'Arduino, laissant à la Raspberry Pi le soin de traiter l'intelligence artificielle embarquée (vision et détection FOD).

L'accessibilité de la plateforme est un autre point fort : l'Arduino est l'une des cartes les plus documentées dans la communauté open source, avec une très large base d'exemples, de bibliothèques, de tutoriels, et de forums d'entraide. Cela a grandement facilité la mise en œuvre, les tests, et les ajustements rapides lors du développement du robot.

Enfin, l'Arduino Uno est un excellent choix rapport qualité/prix pour un prototype fonctionnel et polyvalent pour les projets embarqués. Le starter kit complet, à environ 420 MAD, inclut la carte, les câbles, et une variété de capteurs de base.

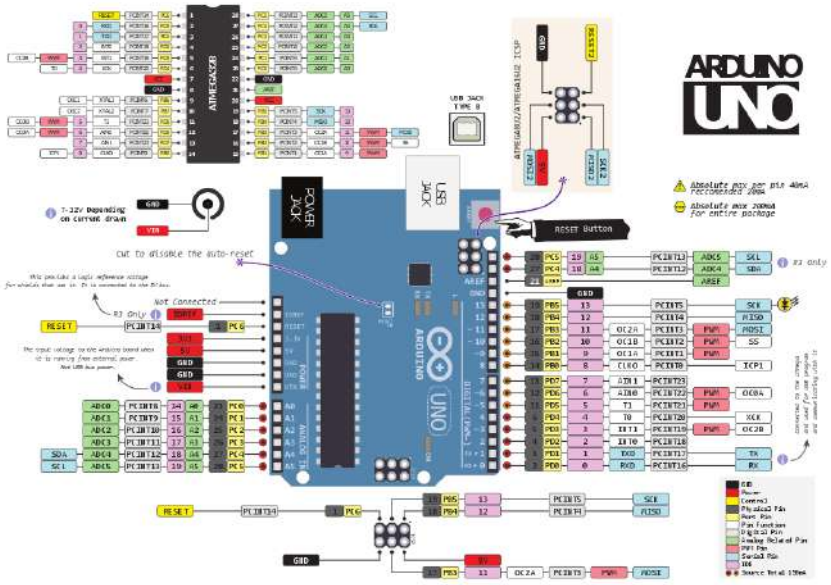


FIGURE 4.1 – Pinout de la carte Arduino Uno

La Raspberry Pi5 offre des performances significativement améliorées par rapport à ses prédécesseurs.

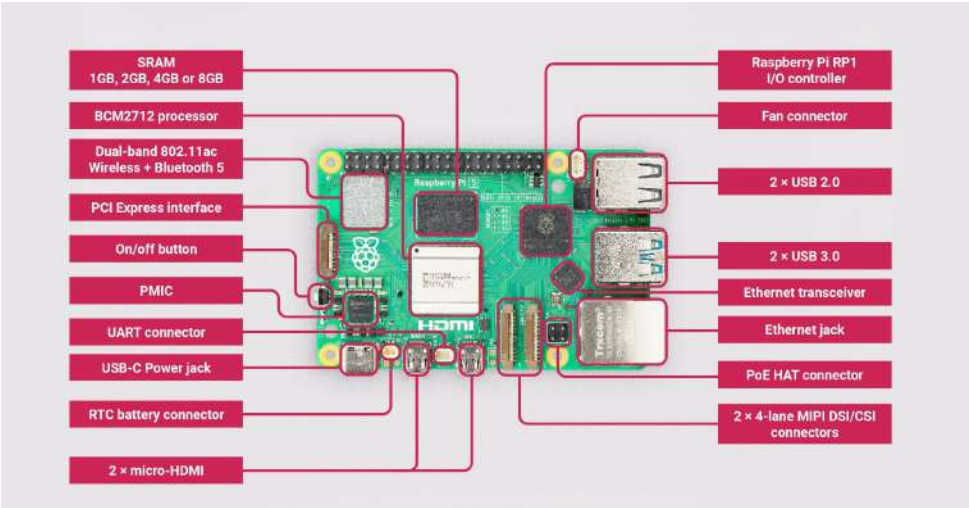


FIGURE 4.2 – Pinout de la Carte Raspberry Pi5

Ses caractéristiques incluent un processeur Broadcom BCM2712, quad-core Cortex-A76 64 bits cadencé à 2,4 GHz ; un stockage via Port microSD pour le système d'exploitation par exemple (Raspberry Pi OS) et autres données. Son alimentation est très sensible (5.1V, 5A), mais ses dimensions compactes sont idéales pour une intégration dans le robot.

TABLE 4.2 – Caractéristiques techniques de la Raspberry Pi 5

Paramètre	**Valeur / Description**
Processeur	Broadcom BCM2712, Quad-core ARM Cortex-A76 à 2.4 GHz
Mémoire RAM	4 Go, 8 Go ou 16 Go LPDDR4X (selon modèle)
Stockage	Carte microSD, ou SSD via port PCIe ou USB 3.0
GPU	VideoCore VII (prise en charge de la 4K à 60 fps)
Connectivité sans fil	Wi-Fi 802.11ac, Bluetooth 5.0
Ports USB	2 × USB 3.0, 2 × USB 2.0
Ports vidéo	2 × micro-HDMI (4K à 60 Hz)
Audio	Sortie audio via HDMI ou jack 3.5 mm (via adaptateur)
Ethernet	Gigabit Ethernet (1 Gbps)
Ports GPIO	40 broches GPIO compatibles HAT
Caméras et écrans	2 ports MIPI (1 caméra + 1 écran)
Alimentation	USB-C, 5V/5A
Dimensions	85.6 mm × 56.5 mm
Système d'exploitation	Raspberry Pi OS (Linux), Ubuntu, etc.

La Raspberry Pi5 occupe un rôle central dans la gestion de l'intelligence embarquée, notamment pour le traitement des images, l'exécution du modèle d'intelligence artificielle (YOLO), et la transmission des résultats vers l'interface utilisateur pour la supervision. La maîtrise de son environnement, de la gestion des ressources système, et de la configuration des outils logiciels a constitué un défi, notamment dans ce contexte de projet à durée limitée. De plus, son prix exhorbitant, 1000 MAD, n'est pas donné pour un projet étudiant, mais ce coût est pleinement justifié par ses capacités de calcul nettement supérieures, des performances qui ont pleinement justifié l'effort initial d'apprentissage, et ont permis d'implémenter une solution embarquée puissante et évolutive.

4.2.2 Caméra embarquée

Le Raspberry Pi Camera Module V3 est une caméra compacte conçue pour les ordinateurs monocarte Raspberry Pi, comme le Raspberry Pi 5 utilisé dans le prototype. Ce module offre des performances améliorées pour les applications de vision par ordinateur, ce qui en fait un choix idéal pour la détection des FOD. Il est muni d'un connecteur MIPI CSI-2 pour une connexion directe au Raspberry Pi.
Avec 12 mégapixels, la caméra capture des détails fins, essentiels pour détecter de petits débris comme des boulons ou des fragments de pneu. Conçue pour le Raspberry Pi, elle s'interface directement via le port MIPI, réduisant la latence et simplifiant le câblage.

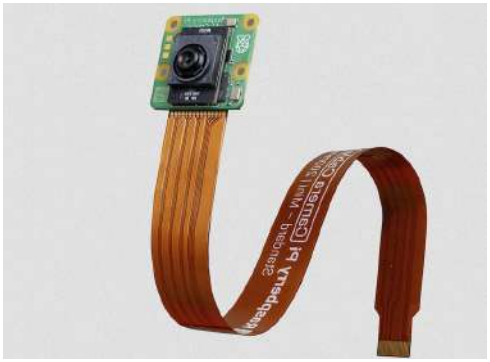


FIGURE 4.3 – Camera Raspberry Module V3

4.2.3 Moteurs et Driver

Moteurs

Nous avons utilisé 04 moteurs à courant continu avec réducteur couplé à un train d'engrenages. Chaque moteur est contrôlé par deux signaux depuis l'Arduino : un signal PWM pour ajuster la vitesse, et un signal logique pour le sens de rotation. Un module L298N permet de contrôler deux moteurs à la fois, donc parfait pour un robot 4WD avec un double H-Bridge.

FIGURE 4.4 – moteur robot 4WD

Driver

Le L298N est basé sur un circuit intégré de puissance capable de fournir jusqu'à 2A par canal (avec un dissipateur thermique). Il possède deux canaux de contrôle (A et B), chacun permettant de contrôler un moteur DC. Chaque canal contient un pont en H, une configuration électronique qui permet d'inverser la polarité appliquée au moteur.

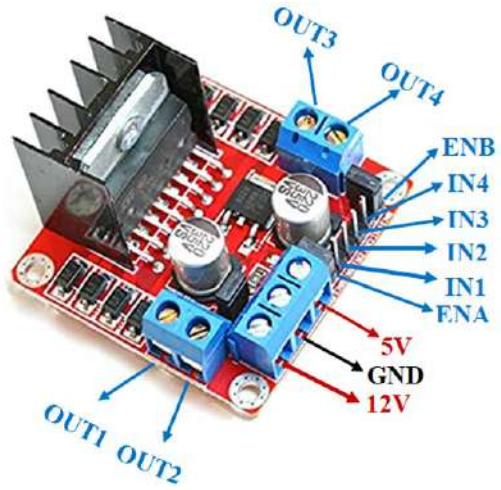


FIGURE 4.5 – Pinout du Module driver L298N

Les deux tableaux ci-dessous présentent l'un les connexions typiques du L298N, et l'autre le principe de fonctionnement logique d'un moteur.

TABLE 4.3 – Connexions typiques du module L298N

Broche	**Fonction**
`IN1`, `IN2`	Commandes logiques pour le moteur A (sens de rotation et arrêt)
`IN3`, `IN4`	Commandes logiques pour le moteur B
`ENA`, `ENB`	Contrôle de l'activation des moteurs (via broche PWM)
`OUT1`, `OUT2`	Connexion au moteur A
`OUT3`, `OUT4`	Connexion au moteur B
`VCC`	Alimentation des moteurs (jusqu'à 35V)
`5V`	Sortie régulée (si cavalier présent) ou alimentation logique (si cavalier retiré)
`GND`	Masse commune (à relier à celle de l'Arduino)

TABLE 4.4 – Logique de commande d'un moteur DC via le pont en H

IN1	IN2	Action du moteur
0	0	Arrêt (haute impédance ou roue libre)
1	0	Rotation dans un sens (ex. horaire)
0	1	Rotation dans l'autre sens (ex. antihoraire)
1	1	Arrêt avec frein actif

4.2.4 Capteurs

Les capteurs sont surtout utilisés pour la fonction de navigation sur la piste, assurant le bon déplacement du robot. Il s'agit principalement de capteurs de vitesse à fente optique LM393, de capteur à ultrasons HC-SR04, un module de capteur inertiel 6 axes intégrant un accéléromètre et un gyroscope MPU6050.

Le capteur à ultrasons HC-SR04 est un module économique et assez fiable. Il fonctionne en émettant des ondes ultrasonores (40 kHz) et en mesurant le temps de retour de l'écho réfléchi par un objet. Il a une portée de 2cm à 4m avec une précision de ±3 mm. Le HC-SR04 est monté à l'avant du châssis, pour détecter les obstacles.

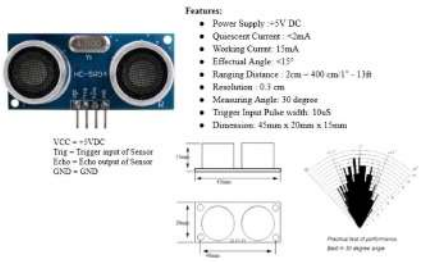


FIGURE 4.6 – Capteur Ultrason HC-SR04

Les encodeurs optiques à fente LM393 sont des capteurs utilisés pour mesurer la vitesse et la distance parcourue par les roues du robot. Chaque encodeur se compose d'une LED infrarouge et d'un phototransistor séparés par une fente, où passe un disque codeur fixé à l'axe du moteur.
Le disque codeur, percé de fentes régulières, interrompt le faisceau infrarouge, générant des impulsions numériques proportionnelles à la rotation. Ceux que nous avons choisi ont 20 fentes pour une résolution de 18° par impulsion, avec une sortie numérique connectée à l'Arduino Uno.

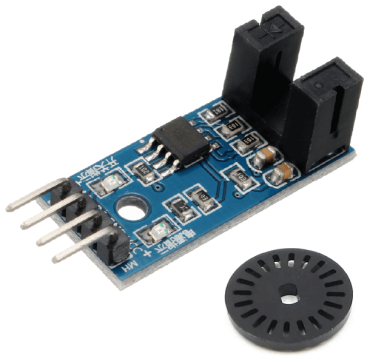


FIGURE 4.7 – Capteur Vitesse LM393

Le MPU6050 est un module de capteur inertiel 6 axes intégrant un accéléromètre et un gyroscope, utilisé pour mesurer l'orientation et les mouvements du robot. Dans notre cas, nous nous intéressons juste au Yaw, c'est-à-dire la rotation dans le plan horizontal de la piste (suivant l'axe Z) permettant de mieux surveiller les virages du robot.

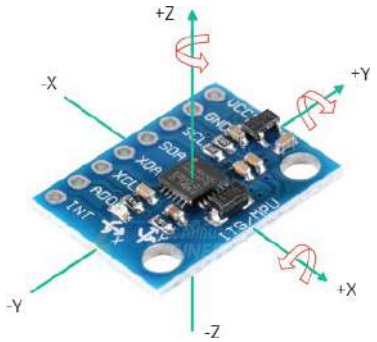


FIGURE 4.8 – Gyroscope MPU6050

4.3 Composants logiciels

4.3.1 L'Intelligence Artificielle pour la détection

L'algorithme de détection repose sur YOLOv5 Nano, une architecture de détection d'objets en temps réel développée par Ultralytics. Elle est basée sur PyTorch, un framework de deep learning open source.

Fonctionnement :

YOLOv5 Nano repose sur les fondements de l'apprentissage profond, en particulier sur les réseaux de neurones convolutifs (CNN), qui sont capables d'extraire automatiquement des caractéristiques visuelles pertinentes à partir d'images d'entrée.

YOLO adopte une stratégie end-to-end à une seule passe : l'image entière est traitée en une seule fois par le réseau, qui divise l'image en une grille et prédit pour chaque cellule les boîtes englobantes (bounding boxes), les classes probables des objets présents, et leur

score de confiance. Cette approche unifiée permet une vitesse de traitement élevée, essentielle pour des applications en temps réel comme la détection de FODs sur une piste d'atterrissage.

Étapes clés :

Prétraitement de l'image : redimensionnement, normalisation.

Passage dans le modèle YOLOv5 Nano : traitement de l'image par CNN.

Post-traitement : filtrage par seuil de confiance et suppression des détections redondantes (non-max suppression).

Annotation et affichage : superposition des boîtes sur l'image et export des résultats.

Le choix de YOLOv5 Nano

En plus de sa compatibilité avec la Raspberry Pi5, de sa documentation riche et communauté active, ce modèle présente une facilité d'entraînement et d'adaptation, ce qui nous a permis de créer notre propre base de données annotée de FODs, de l'entraîner localement, et d'obtenir un modèle ajusté à notre cas d'usage spécifique.

Le modèle a été reformé ou adapté (fine-tuning) sur un ensemble de données spécifiques aux objets FOD courants dans les zones aéroportuaires (clous, outils, plastiques...).

Labellisation et entraînement du modèle de détection

Nous avons constitué manuellement une base d'images en photographiant différents objets posés au sol dans des conditions variées : angles de vue, luminosité, fond plus ou moins homogène. Chaque image représente une situation réaliste de débris abandonné sur une piste ou dans ses abords. Au total, environ 203 images ont été collectées, redimensionnées, et organisées selon les bonnes pratiques YOLO.

FIGURE 4.9 – Base de données : Images d'entraînement

Pour annoter notre base d'images, nous avons utilisé Label Studio, un outil open source très complet et ergonomique. Son interface web permet de créer facilement des bounding boxes (boîtes englobantes) autour des objets présents dans chaque image, puis d'y associer une classe prédéfinie.

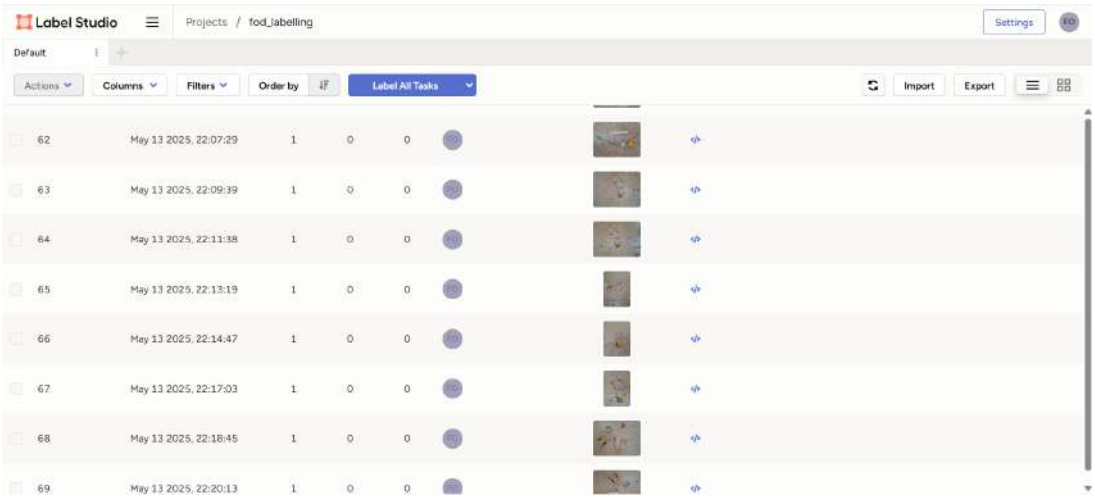


FIGURE 4.10 – Environnement de Label-Studio

L'entraînement du modèle YOLOv5 a été réalisé sur Google Colab, qui offre un environnement Python avec GPU gratuit. Après avoir préparé et annoté notre base de données au format YOLO, celle-ci a été téléversée sur Google Drive et montée dans Colab.

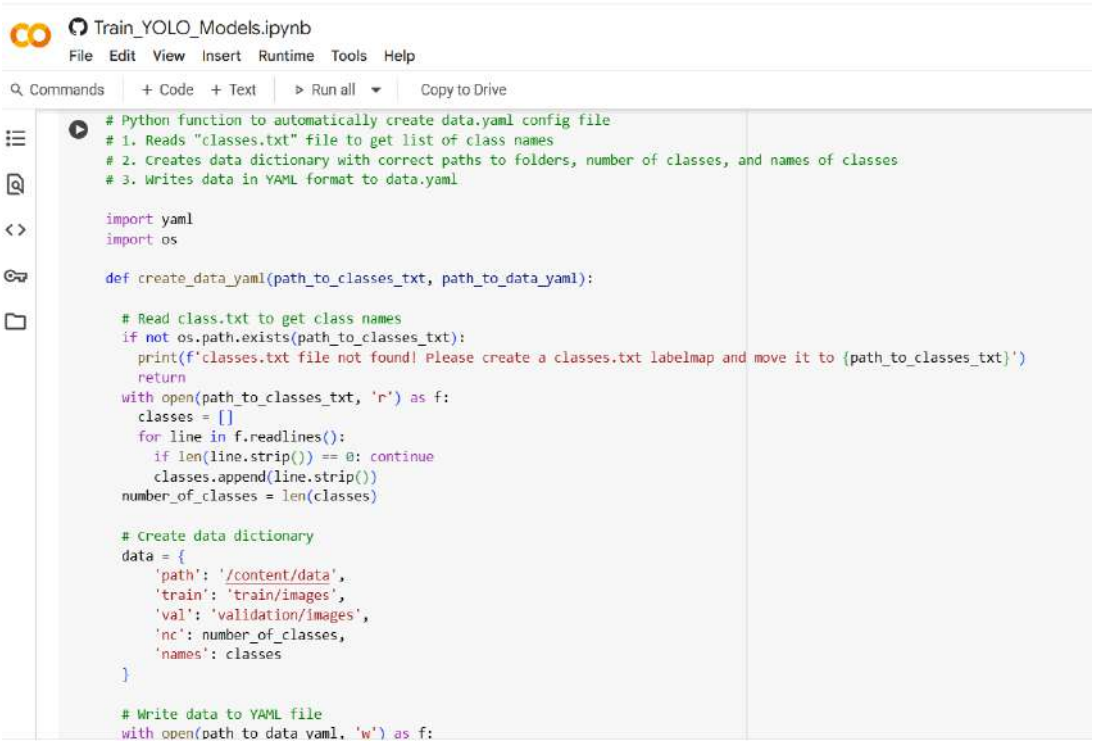


FIGURE 4.11 – Environnement Google Colab

4.3.2 Algorithme de navigation autonome

L'algorithme de navigation repose sur la méthode dite de Dead Reckoning, combinée avec des capteurs d'orientation (MPU6050) et des encodeurs.

Fonctionnement :

Dead Reckoning consiste à estimer la position du robot à partir de sa vitesse, de son orientation et du temps écoulé. Les encodeurs montés sur les roues mesurent le déplacement linéaire. Le capteur MPU6050 fournit l'angle de rotation (yaw) et l'accélération.

Un calcul vectoriel est effectué à chaque instant pour estimer la position :

$$x(t+1) = x(t) + v \cdot \cos(\theta) \cdot \Delta t$$
$$y(t+1) = y(t) + v \cdot \sin(\theta) \cdot \Delta t$$

Fonctions intégrées :

PID pour la correction de trajectoire selon le cap actuel vs. le cap désiré.

Suivi de trajectoire segmentée : le robot passe d'un segment à l'autre dès qu'il atteint une distance seuil.

La détection d'obstacle (capteur ultrason) déclenchant une manœuvre d'évitement spécifique avant de revenir à un cap 0°.

Environnement informatique :

Langages : C++ pour Arduino (contrôle des moteurs et capteurs), Python pour le module de gestion des données (sur Raspberry Pi)

Bibliothèques : Wire.h, MPU6050.h, Encoder.h

4.3.3 Interface web

L'interface web permet la visualisation des détections après l'inspection du robot, et offre une interaction simple avec l'utilisateur.

Fonctionnalités principales :

Liste des objets identifiés avec leur type, position estimée

Affichage de l'image avec la ou les objets détectés

Chapitre 5

Réalisation et Implémentation

Ce chapitre présente la concrétisation du projet à travers la mise en œuvre du système complet, tant sur le plan matériel que logiciel. Il détaille la structure physique du robot, l'architecture logicielle mise en place ainsi que les tests de validation réalisés.

L'ensemble de cette phase de réalisation a permis de transformer les choix de conception établis auparavant en un prototype fonctionnel, répondant aux exigences du cahier des charges.

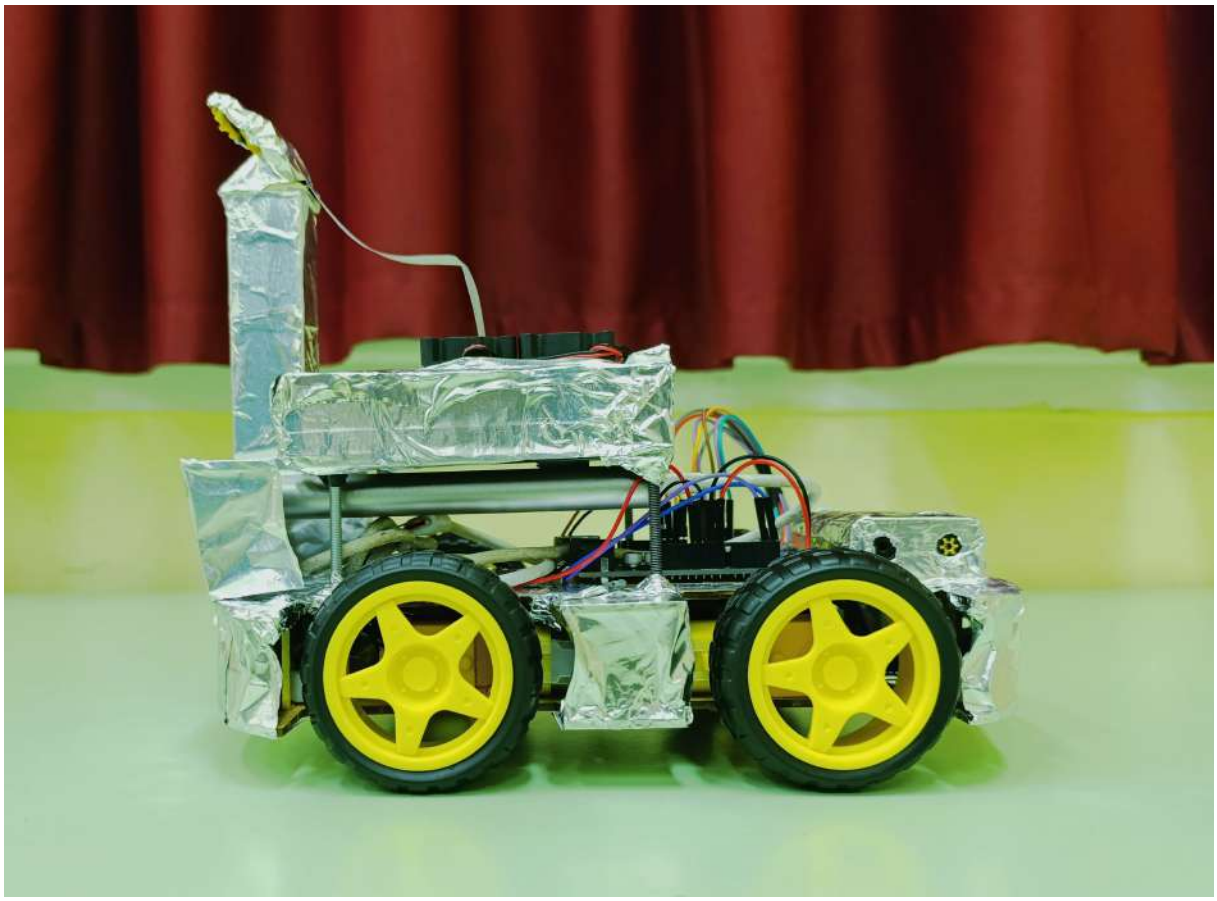


FIGURE 5.1 – Le robot FODEX

5.1 Partie Hardware

La partie matérielle constitue la base physique du système, regroupant les différents composants électroniques, mécaniques et d'alimentation.

5.1.1 Structure du robot

Le robot repose sur un châssis à quatre roues motrices qui supporte l'ensemble des modules électroniques embarqués. Les moteurs DC sont pilotés par un pont L298N, permettant de contrôler indépendamment la vitesse et le sens de rotation de chaque roue.

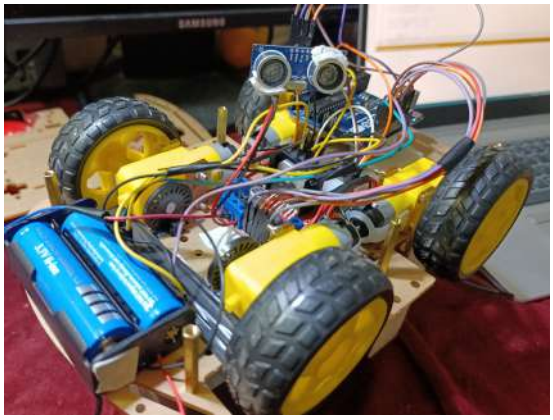


FIGURE 5.2 – Assemblage des composants

5.1.2 Cartes de contrôle embarquée

Nous avons opté pour une architecture double carte : une Arduino Uno, responsable du contrôle bas-niveau des moteurs, de la lecture des capteurs, et de la gestion de la navigation locale et une Raspberry Pi 5 utilisée pour le traitement d'images et l'exécution du modèle de détection de FOD. Les deux cartes communiquent via un port série.

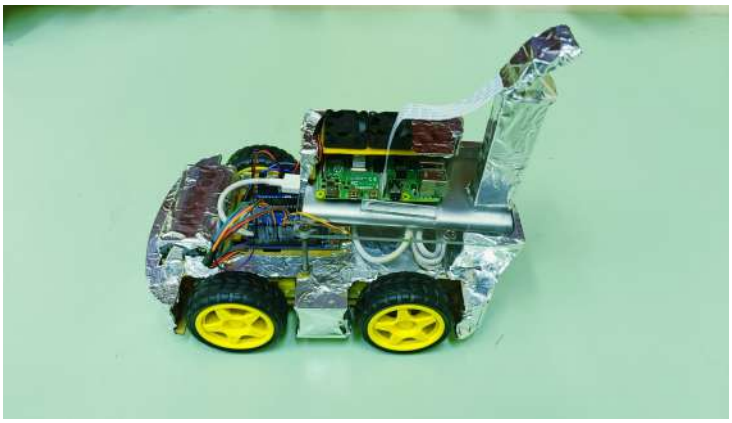


FIGURE 5.3 – Cartes de contrôle du robot

5.1.3 Alimentation

L'un des principaux défis dans la réalisation du robot autonome était de garantir une alimentation électrique stable et suffisante, notamment pour la Raspberry Pi 5, qui est plus gourmande en énergie que ses versions précédentes. Une mauvaise gestion de l'alimentation peut entraîner des redémarrages, des pertes de données ou des ralentissements critiques dans le traitement d'image ou la navigation.
Pour ce faire, deux sources d'alimentation ont été utilisées : un pack de batteries LiPo 3,7V (x2) pour alimenter les moteurs et l'Arduino, une power bank pour l'alimentation stable de la Raspberry Pi 5, garantissant un courant suffisant pour les calculs intensifs liés au deep learning. Cette séparation évite les chutes de tension, les perturbations dues aux pics de courant et les plantages.

5.1.4 Intégration capteurs-contrôleur

L'un des premiers défis techniques a concerné la calibration et l'intégration correcte des capteurs avec la carte Arduino. La précision de la navigation dépendait fortement de la calibration des capteurs, notamment du MPU6050 pour l'orientation (yaw). Une mauvaise calibration entraînait une dérive importante de la trajectoire. le capteur dérive au fil du temps, affectant la précision de l'angle utilisé pour corriger la trajectoire. De plus, les mouvements brusques du robot génèrent des données bruitées difficiles à exploiter.
Nous avons dû opter pour une calibration fréquente pour fusionner les données gyroscopiques et accélérométriques, ce qui n'a pas manqué d'alourdir les tâches pour l'Arduino.

Le capteur à ultrasons, bien qu'utile pour la détection d'obstacles, a posé plusieurs difficultés lors des phases de test. Environ 1 fois sur 10, il renvoyait des valeurs erronées, soit très éloignées de la réalité, soit nulles, ce qui trompait le système de décision du robot. Ces erreurs étaient principalement dues à des réflexions parasites sur certaines surfaces (bords inclinés, objets absorbants ou très lisses), à la présence de multiples obstacles proches, créant des interférences, ou encore des zones trop étroites où l'onde sonore se dispersait mal. Ces lectures incorrectes ont parfois provoqué des manœuvres d'évitement injustifiées, compromettant temporairement la robustesse du système.
Pour limiter ces effets, des techniques de filtrage simples (comme la moyenne glissante) ont été envisagées.

5.2 Partie software

Le développement logiciel a été réparti entre le contrôle embarqué, le traitement d'images et l'interface utilisateur de supervision. L'ensemble du code a été développé principalement en C++ pour Arduino et en Python pour la Raspberry Pi.
La structure de la partie Software est résumée dans ce diagramme.

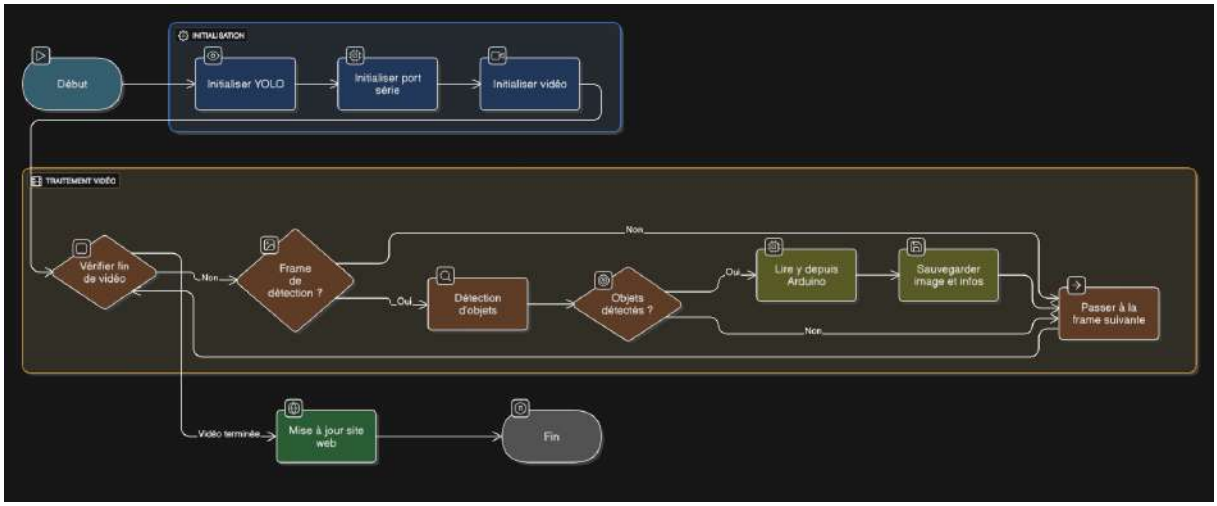


FIGURE 5.4 – Diagramme recapitulatif Partie Software

5.2.1 Arduino IDE

Le programme de l'Arduino est responsable de La lecture continue des capteurs, la commande des moteurs en fonction des consignes de direction et de la correction de trajectoire (PID sur le yaw), et la gestion des manœuvres d'obstacles : si un obstacle est détecté à moins de 15 cm, le robot effectue une séquence d'évitement avec changement d'orientation à ±90°, puis retourne à l'angle initial (0°).

L'utilisation du GPS a été initialement envisagée pour faciliter la localisation du robot, notamment dans la détection et la cartographie de la position des débris. Cependant, ce choix technologique a été rapidement écarté en raison de deux contraintes majeures. D'une part, nous évoluons dans un environnement réduit (inférieur à 5 mètres), bien en deçà de la résolution typique du GPS standard, qui varie entre 2 à 5 mètres. À cette échelle, le déplacement du robot ne serait même pas perceptible avec le GPS, rendant impossible toute exploitation fiable des données.
D'autre part, même si le GPS aurait considérablement simplifié la gestion de la position absolue du robot dans un espace étendu, son inadéquation aux environnements restreints et son imprécision à petite échelle ont représenté un obstacle majeur. Cela nous a poussés à recourir à des méthodes de localisation relative, telles que le Dead Reckoning, combinant les données des encodeurs et du gyroscope, pour estimer la position du robot.

5.2.2 Détection d'objets par vision (Raspberry Pi 5

La Raspberry Pi 5 est chargée de la détection des FODs à l'aide d'un modèle d'intelligence artificielle entraîné spécifiquement. Le processus s'est déroulé en plusieurs étapes :

Constitution de la base de données

Nous avons collecté des centaines d'images de débris typiques (fragments métalliques, bouteilles, outils, vis, papiers, déchets plastiques, etc.), prises dans différents environnements simulant une piste d'atterrissage. Ces images ont ensuite été triées et renommées pour un traitement unifié.

Labellisation avec Label Studio

Pour annoter ces images, nous avons utilisé l'outil open source Label Studio, qui permet de dessiner des boîtes englobantes (bounding boxes) autour des objets à détecter, et de leur attribuer une classe (ex : FOD). Le projet a été configuré au format YOLO, ce qui a généré un fichier .txt par image, contenant les coordonnées normalisées des objets annotés.

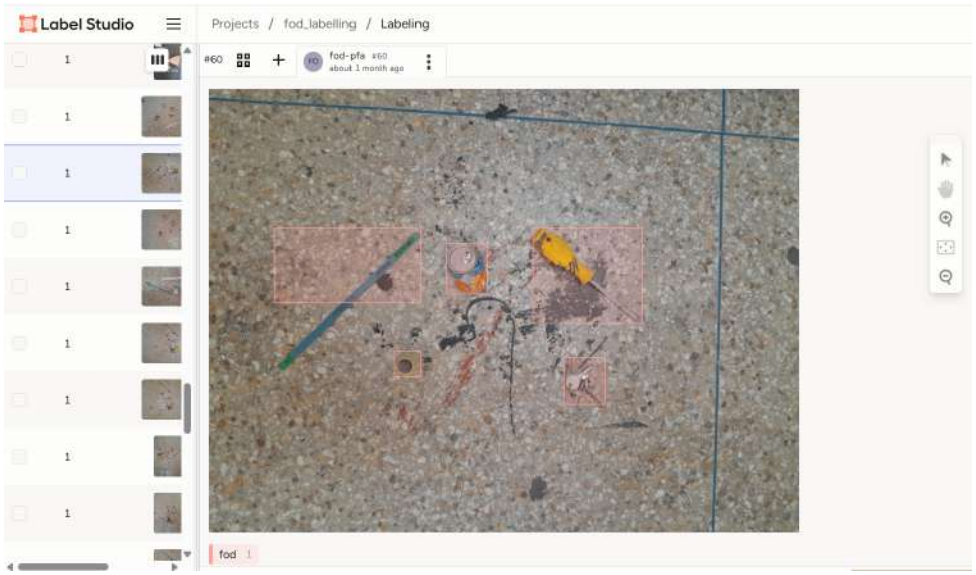


FIGURE 5.5 – Processus de labélisation

Entraînement du modèle sur Google Colab

Google Colab nous a permis de bénéficier gratuitement d'un GPU (NVIDIA Tesla T4) pour accélérer l'apprentissage. Les étapes ont été les suivantes :
Téléversement des images et des annotations,
Configuration du fichier data.yaml (chemins, classes),

Entraînement du modèle yolov5.pt,
Surveillance de la précision (mAP) et de la perte (loss) via les courbes de suivi.

À l'issue de l'entraînement, nous avons obtenu un modèle final atteignant une précision de détection supérieure à 90% sur les objets les plus fréquents.

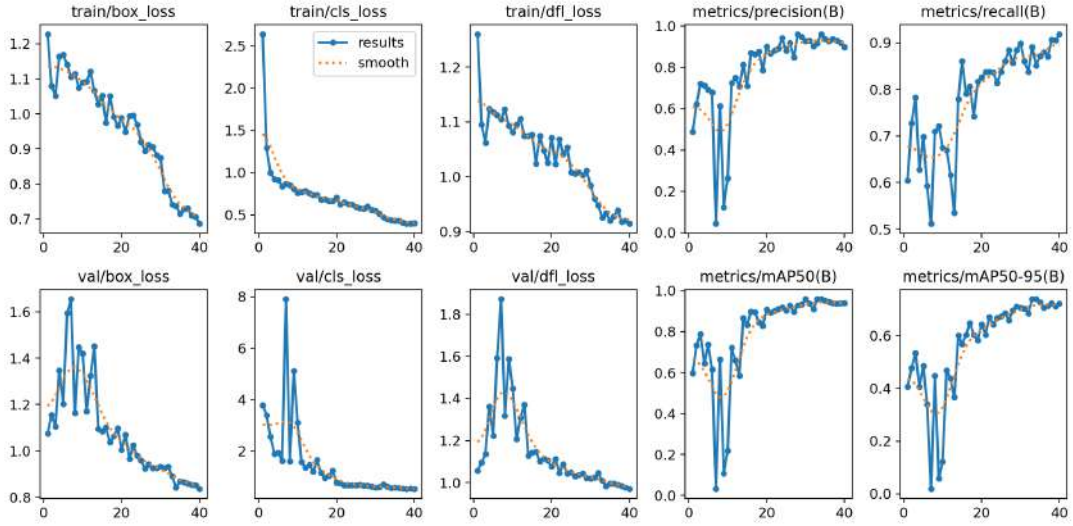


FIGURE 5.6 – Courbes de suivi

Exportation du modèle vers la Raspberry Pi

Le modèle final (best.pt) a été converti et optimisé pour un usage embarqué, puis transféré sur la Raspberry Pi. Une fois installé, il est exécuté via un script Python exploitant PyTorch. Ce script permet :
La capture en temps réel depuis la caméra Pi,
Le traitement image par image avec le modèle YOLOv5,
L'affichage des détections (classes, scores de confiance, boîtes),
L'enregistrement local des résultats (image annotée, classe).

5.2.3 Interface et organisation logicielle

Une interface web légère a été développée, permettant de visualiser les images capturées et annotées, ainsi qu'un journal de détections FOD.

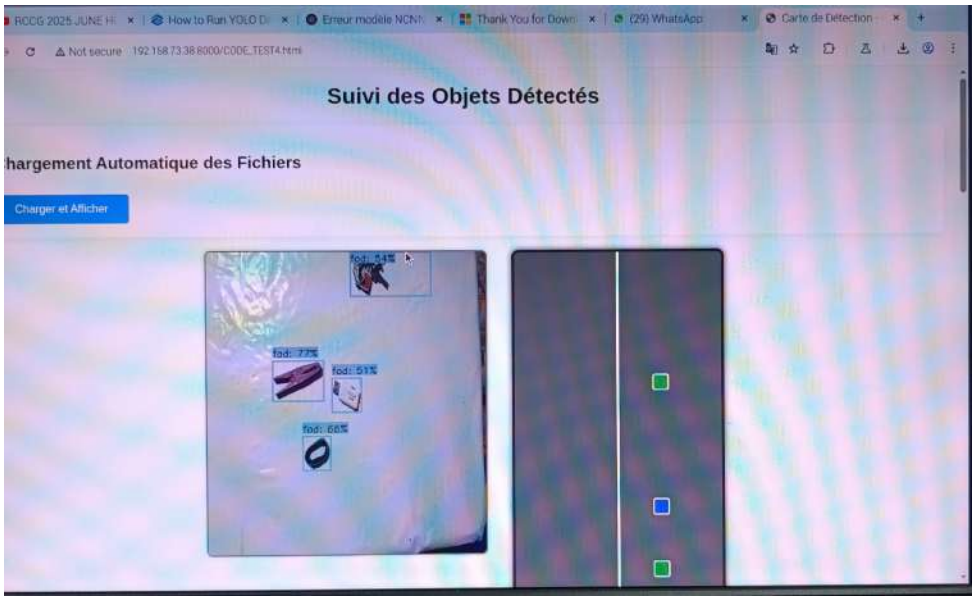


FIGURE 5.7 – Interface de visualisation des résultats de détection avec carte de la piste

5.3 Résultats et Validation du fonctionnement du Robot FODEX

Afin d'évaluer les performances du prototype, plusieurs tests fonctionnels ont été menés dans des environnements simulant une piste d'atterrissage.

5.3.1 Scénario de test

Les tests ont été réalisés sur une piste rectiligne d'environ 5 mètres, avec des objets représentant des FOD placés aléatoirement (bouteilles, pièces métalliques, outils). Les tests ont été effectués dans diverses conditions.

5.3.2 Navigation et évitement

Le robot s'est déplacé avec stabilité, en maintenant un cap stable grâce à la correction en yaw issue du gyroscope. Lorsqu'un obstacle était détecté à moins de 15 cm, la manœuvre d'évitement en trois phases était bien exécutée. Après la dernière phase, le robot reprenait son cap initial (0°) avec une erreur angulaire moyenne inférieure à 3°.

5.3.3 Résultats de détection

Le modèle YOLOv5 a correctement détecté 95% des objets dans des conditions normales. Il est même parvenu à détecter des objets de très petite taille (2 à 3cm), bien mieux

que la limite minimale 5cm que l'on s'était fixée au départ, bien que la probalité n'est pas très élevée dans ce cas.

FIGURE 5.8 – Tests de Validation du modèle (Visualisation sur Interface après mission)

Par ailleurs, même dans des conditions de faible luminosité, le robot arrive à détecter les débris. Les erreurs de détection ont été principalement dues à des objets de très petite taille, ou un flou de mouvement.

FIGURE 5.9 – Test de détection en environnement de faible luminosité

5.3.4 Analyse technico-économique de notre solution

Estimation du Prix

Le développement du robot s'est inscrit dans un cadre académique, avec une gestion parallèle des cours, des examens et du projet lui-même. L'équipe a investi environ 7 semaines, ce qui représente un volume estimé à 140 à 150 heures de travail individuel, réparti entre la conception, l'implémentation technique, les tests sur piste et la préparation des livrables.

Sur le plan financier, le coût global du projet s'élève à environ 3 000 MAD, couvrant l'ensemble des composants matériels nécessaires à la réalisation du prototype. Ce budget

a été entièrement optimisé dans une logique pédagogique, en privilégiant les ressources accessibles et compatibles avec un environnement d'apprentissage. Aucun coût de main-d'œuvre n'a été comptabilisé, le projet ayant été réalisé dans le cadre d'un travail étudiant. Ce montant, relativement modeste au regard des solutions industrielles de détection de FOD dont les coûts atteignent plusieurs millions de dirhams, illustre l'intérêt des solutions low-cost développées dans les milieux universitaires pour explorer des alternatives viables, flexibles et pédagogiques à haute valeur ajoutée.

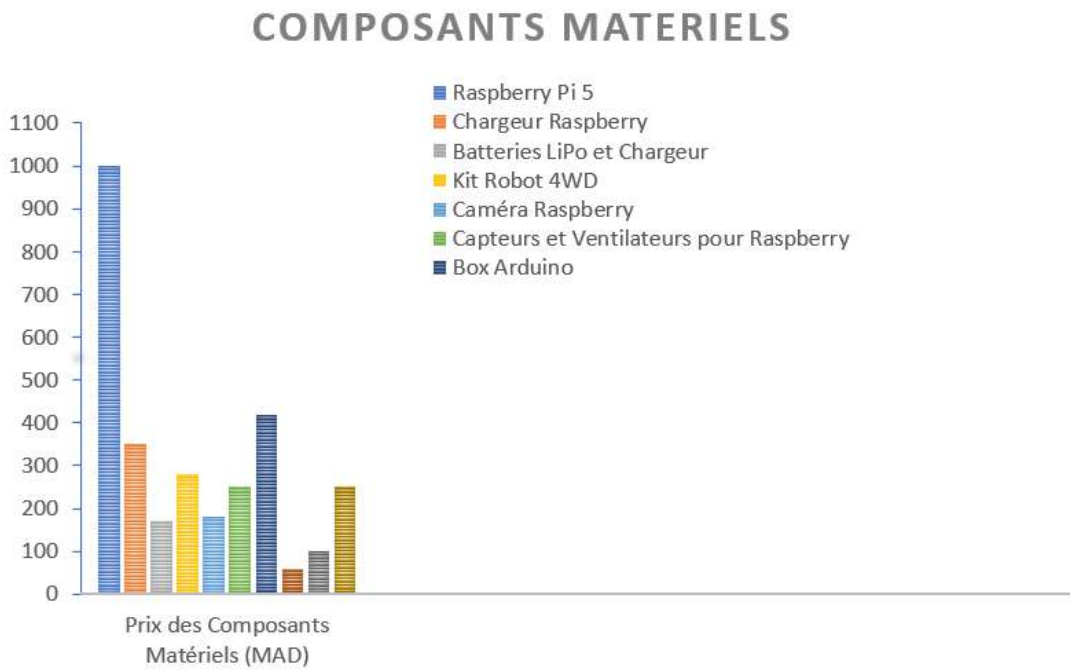


FIGURE 5.10 – Estimation du prix brut du Robot FODEX

Comparaison avec les systèmes existants

Originalité du projet

Ce projet se distingue par son approche novatrice de la problématique critique des débris sur les pistes aéroportuaires. Alors que la majorité des solutions existantes reposent sur des infrastructures fixes coûteuses (telles que des radars millimétriques ou des caméras intelligentes implantées sur toute la longueur des pistes) notre système propose une alternative radicalement différente : un robot mobile autonome, modulaire et économique, capable de détecter les FOD de manière flexible et évolutive.
Son originalité réside dans plusieurs aspects clés :

— **Mobilité :** contrairement aux systèmes fixes, notre robot peut parcourir dynamiquement différentes zones critiques, y compris les accotements et zones secondaires souvent négligées.

- **Accessibilité :** conçu à partir de composants open-source (Arduino, Raspberry Pi, capteurs abordables), il offre une solution viable pour les aéroports de petite ou moyenne taille, ou pour des pays à ressources limitées.
- **Scalabilité :** le système peut évoluer vers une flotte coordonnée de robots collaboratifs pour couvrir de larges surfaces.
- **Intelligence embarquée :** grâce à l'intégration possible d'algorithmes de vision par ordinateur et de capteurs inertiels, le robot ne se limite pas à la détection passive, mais peut analyser en temps réel et prendre des décisions autonomes
- **Potentiel éducatif et expérimental :** au-delà de son application opérationnelle, ce robot constitue également une plateforme pédagogique idéale pour expérimenter l'intégration de capteurs, la robotique mobile, l'IA embarquée et la sécurité aérienne.

Ce projet ouvre ainsi de nouvelles perspectives dans la surveillance aéroportuaire de sécurité, en s'inscrivant dans une démarche à la fois pragmatique, durable et technologiquement inclusive.

Valeur ajoutée

Afin de mieux situer la valeur ajoutée de notre projet, il est pertinent de le comparer aux systèmes de détection de FOD actuellement déployés dans certains grands aéroports internationaux. Ces systèmes, bien que technologiquement avancés, présentent des limitations en termes de coût, de flexibilité et d'accessibilité, notamment pour les aéroports régionaux ou les infrastructures situées dans des pays en développement.
Notre solution robotique, pensée pour être mobile, économique et facilement déployable, répond à ces enjeux avec une approche plus modulaire et inclusive.

Le tableau suivant présente une comparaison synthétique entre notre robot et les solutions conventionnelles, mettant en évidence les avantages concurrentiels et l'intérêt stratégique de notre démarche.

TABLE 5.1 – Comparaison entre notre robot mobile de détection de FOD et les systèmes existants

Critères	**Systèmes existants (radar, optique)**	**Notre robot mobile de détection de FOD**
Coût d'installation	Très élevé (en Millions de Dollars), infrastructure fixe, capteurs haut de gamme	Faible coût (environ 300 dollars), matériel open-source, composants réutilisables
Facilité de déploiement	Moins flexible, nécessite travaux d'installation	Rapide à mettre en œuvre, installation quasi plug-and-play
Mobilité	Fixes, limités à certaines zones de la piste	Entièrement mobile, possibilité d'explorer plusieurs zones dynamiquement
Maintenance	Nécessite des techniciens spécialisés, interruption possible du service	Facile à entretenir, modules interchangeables
Portée de détection	Longue portée (jusqu'à 1 km avec radar)	Portée plus limitée (dépend du capteur utilisé), mais plus proche du sol
Flexibilité d'usage	Faible (configuration spécifique à l'aéroport)	Élevée (adaptable à plusieurs types de terrains ou d'aéroports)
Analyse intelligente	Oui, avec IA intégrée (détection automatique via caméra ou radar)	Oui, possibilité d'intégrer du traitement embarqué (ex. YOLO, TensorFlow Lite)
Évolutivité et flexibilité	Faible (mise à jour complexe)	Élevée (mise à jour logicielle, ajout de capteurs facile)
Adaptabilité aux petits aéroports	Souvent non rentable pour les petits sites	Solution idéale pour les petits aéroports ou aérodromes locaux

Conclusion générale et Perspectives

Bilan du projet

Ce projet de fin d'année a constitué une aventure technique, humaine et intellectuelle marquante, à l'intersection de l'aéronautique, des systèmes embarqués et de l'intelligence artificielle. Nous avons conçu, modélisé et implémenté un robot autonome capable de détecter les FODs (Foreign Object Debris) sur une piste, en combinant des outils modernes comme l'Arduino Uno, la Raspberry Pi 5, le deep learning avec YOLOv5, et des techniques de navigation inertielle.

En tant qu'élèves ingénieurs en systèmes embarqués et télécommunications, ce travail nous a permis de développer des compétences concrètes dans la conception de systèmes cyber-physiques, la gestion de projet, l'intégration capteur-contrôleur, ainsi que dans la communication machine-machine et le traitement d'images embarqué. La rigueur imposée par le cadre aéronautique, conjuguée aux contraintes de temps, de ressources et de fiabilité, a renforcé notre sens de l'analyse, de l'optimisation et de la prise de décision technique. Au-delà des compétences techniques, ce projet nous a également formés à la collaboration interdisciplinaire, à la documentation rigoureuse, et à la communication scientifique.

Perspectives

Le projet a dû être conçu, développé, testé et documenté dans un laps de temps restreint (quelques semaines), ce qui a limité le temps alloué à la phase d'optimisation ou à l'intégration avancée de technologies comme le GPS ou la navigation assistée par caméra. Malgré ces contraintes, le projet a pu atteindre ses objectifs fonctionnels grâce à une bonne coordination de l'équipe, une organisation rigoureuse et des choix techniques réalistes et adaptés au temps disponible.

Si ce prototype a démontré la faisabilité technique d'un robot de détection des FODs à bas coût, son évolution vers un système réellement déployable en environnement aéroportuaire réel ouvre des perspectives prometteuses.

Parmi les pistes identifiées, l'intégration d'un GPS de précision centimétrique (RTK-GPS) ou l'usage de la fusion de capteurs (encodeurs, gyroscope, IA visuelle) permettrait une localisation fiable à grande échelle. Le remplacement du capteur ultrason par un LiDAR améliorerait significativement la perception de l'environnement, notamment en

cas d'obstacles multiples, et ouvrirait la voie à une cartographie 2D/3D en temps réel. En complément, l'ajout d'un module de classification automatique des FODs offrirait une capacité d'analyse contextuelle, permettant non seulement de détecter, mais de prioriser les interventions selon le type de débris (métallique, plastique, textile, etc.).

D'autres améliorations telles qu'un bras robotisé embarqué pour la collecte automatique des petits objets, une gestion énergétique intelligente (retour automatique à la base, recharge solaire), ou encore une interface web de supervision en direct pourraient être intégrées dans une version industrielle.

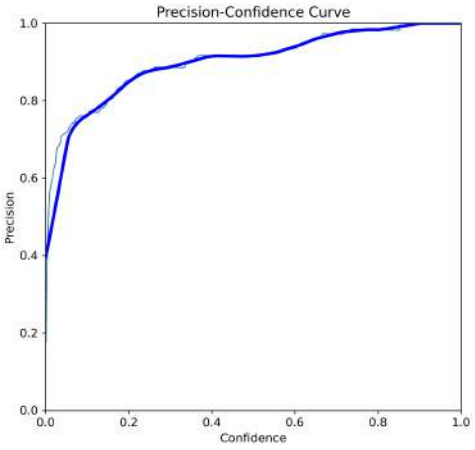
Enfin, une phase de validation en conditions réelles – en collaboration avec des autorités telles que l'ONDA ou des écoles d'aviation – serait nécessaire pour évaluer la robustesse, la conformité aux normes OACI, et la valeur ajoutée opérationnelle du système. Ce projet, s'il est poursuivi, pourrait ainsi devenir une solution innovante de surveillance aéroportuaire, particulièrement adaptée aux aéroports secondaires ou aux pays émergents recherchant des alternatives efficaces et accessibles.

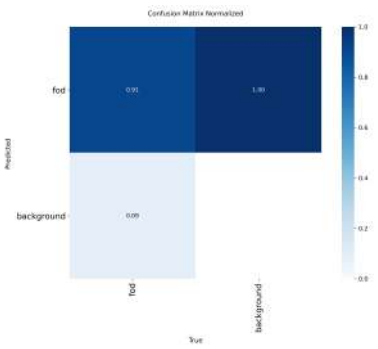
Annexes

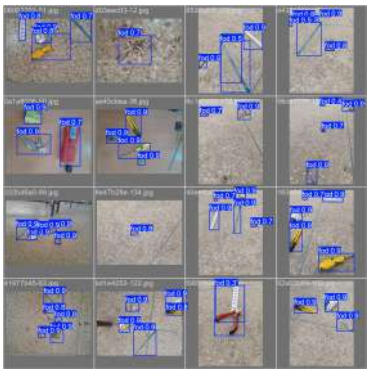
Arborescence du dossier d'entraînement Yolo

- yolo model/
  - fod images/ (base de données)
  - data/ (de label-studio)
    - notes.json
    - classes.txt
    - images/
    - labels/ (labels des images)
  - my model/ (de Google Colab)
    - yolo-detect.py (algorithme de détection)
    - my-model.pt
    - train/ (dossier d'entrainement)
      - args.yaml
      - confusion-matrix
      - val-batch-labels
      - results
      - results.py
      - weights/ (best.pt (utilisé pour le déploiement) et last.pt)

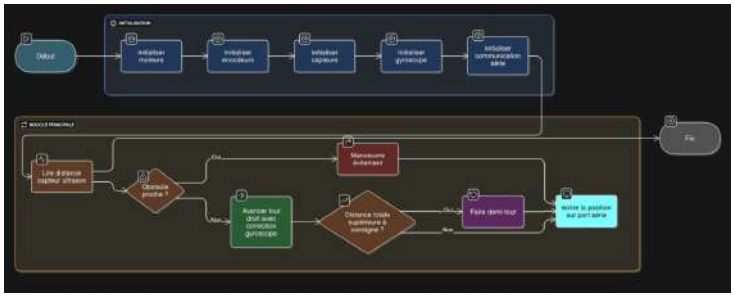
Courbes de performance et Batch de prédiction







Organigramme de déplacement



Code source de déplacement du robot (Arduino)

```
/*-----------------------------------------SETUP-----------------------------------------*/
void setup() {
  // Moteurs
  pinMode(moteur1, OUTPUT);
  pinMode(moteur2, OUTPUT);
  pinMode(moteur3, OUTPUT);
  pinMode(moteur4, OUTPUT);
  pinMode(ENA, OUTPUT);
  pinMode(ENB, OUTPUT);

  // Encodeurs
  pinMode(encodeurGauche, INPUT_PULLUP);
  pinMode(encodeurDroit, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(encodeurGauche), tickGauche, RISING);
  attachInterrupt(digitalPinToInterrupt(encodeurDroit), tickDroit, RISING);

  // Capteur ultrason
  pinMode(triggerPin, OUTPUT);
  pinMode(echoPin, INPUT);

  // MPU6050
  Wire.begin();
  Wire.beginTransmission(MPU_ADDR);
  Wire.write(0x68);
  Wire.write(0);
  Wire.endTransmission(true);
  delay(1000);
  calibrateGyro();
  last_time = millis();

  Serial.begin(9600);
}
```

```
/*-----------------------------------------LOOP-----------------------------------------*/
void loop() {
  naviguer();
}

/*-----------------------------------FONCTION NAVIGUER-----------------------------------*/
void naviguer() {
  long distanceObstacle = lireDistanceUltrason();
  if (distanceObstacle > 0 && distanceObstacle < 15) {
    manoeuvreEvitement();

  } else {
    avancerToutDroitCorrige(400);  // continuer à avancer normalement
    if(DemiTour){
      faireDemiTour(600);}
      DemiTour = false;
  }
}

/*-----------------------------FONCTION TOURNER D'UN ANGLE-----------------------------*/

// Fonction pour tourner jusqu'à un angle précis en degrés
// sensGauche = true pour tourner à gauche (angle décroissant)
// sensGauche = false pour tourner à droite (angle croissant)
void tournerJusquA(float angleCible, bool sensGauche) {
  mettreAJourYaw();
  float angleDepart = angle_z;
  float angleActuel = angle_z;

  // Ajustement angle pour tourner dans la bonne direction
  while (true) {
    mettreAJourYaw();
    angleActuel = angle_z;

    // Calcul de l'erreur en tenant compte du sens
    float erreur = angleCible - angleActuel;
```

```
  // Tolérance +/- 1 degrés
  if (sensGauche && erreur <= 1) break;
  if (!sensGauche && erreur >= -1) break;

  // Commande moteurs pour tourner
  if (sensGauche) {
    // Tourner à gauche
    digitalWrite(moteur1, LOW); //gauche arret
    digitalWrite(moteur2, LOW);
    digitalWrite(moteur3, HIGH); //droit avance
    digitalWrite(moteur4, LOW);
    analogWrite(ENA, VITESSE_PWM_TURN);
    analogWrite(ENB, VITESSE_PWM_TURN);
  } else {
    // Tourner à droite
    digitalWrite(moteur1, LOW);
    digitalWrite(moteur2, HIGH); //gauche avance
    digitalWrite(moteur3, LOW);
    digitalWrite(moteur4, LOW); //droit arret
    analogWrite(ENA, VITESSE_PWM_TURN);
    analogWrite(ENB, VITESSE_PWM_TURN);
  }
  delay(10);
}

arreter();
delay(10);
}
```

```
/*-----------------------------FONCTION FAIRE DEMI TOUR-----------------------------*/
void faireDemiTour(int duree){
    tournerJusquA(90, true);   // tourner à gauche de 90°

    //Avancer légèrement
    digitalWrite(moteur1, LOW);
    digitalWrite(moteur2, HIGH); //gauche avance
    digitalWrite(moteur3, HIGH); //droit avance
    digitalWrite(moteur4, LOW);
    analogWrite(ENA, VITESSE_PWM);
    analogWrite(ENB, VITESSE_PWM);
    delay(duree);
    tournerJusquA(180, true);   // encore à gauche de 90°
    delay(400);
    distanceTotale = 0;
    arreter();
    return; // Quitte directement la fonction
}

/*-----------------------------FONCTION AVANCER TOUT DROIT ET CORRIGER-----------------------------*/
void avancerToutDroitCorrige(int duree) {
  mettreAJourYaw();
  float angleDepart = 0.0;
  if(ChangerOrientationGyro){
    angleDepart = 180;
  }

  unsigned long debut = millis();
  unsigned long ticksGaucheStart = ticksGauche;
  unsigned long ticksDroitStart = ticksDroit;

  while (millis() - debut < duree) {
    long distanceObstacle = lireDistanceUltrason();
    if (distanceObstacle > 0 && distanceObstacle < 15) {
      manoeuvreEvitement();
    }
```

```
  mettreAJourYaw();
  float erreur = angle_z - angleDepart;

  float kp = 2;
  int correction = kp * erreur;
  int pwmG = constrain(VITESSE_PWM - correction, 0, 255);
  int pwmD = constrain(VITESSE_PWM + correction, 0, 255);

  digitalWrite(moteur1, LOW);
  digitalWrite(moteur2, HIGH);
  digitalWrite(moteur3, HIGH);
  digitalWrite(moteur4, LOW);
  analogWrite(ENA, pwmG);
  analogWrite(ENB, pwmD);

  delay(10);

  if (distanceTotale > consigneDistance) {
    DemiTour = true;
    ChangerOrientationGyro = true;
    break;
  }
}

unsigned long deltaGauche = ticksGauche - ticksGaucheStart;
unsigned long deltaDroit  = ticksDroit  - ticksDroitStart;
float distanceMoyenne = (deltaDroit / impulsionsParTour) * circonferenceRoue;
distanceTotale += distanceMoyenne;

arreter();
delay(400);
}
```

```
/*-----------------------------------FONCTION ARRETER-----------------------------------*/
void arreter() {
  analogWrite(ENA, 0);
  analogWrite(ENB, 0);
  digitalWrite(moteur1, LOW);
  digitalWrite(moteur2, LOW);
  digitalWrite(moteur3, LOW);
  digitalWrite(moteur4, LOW);
}

/*-----------------------------------MANOEUVRE D'EVITEMENT-----------------------------------*/
void manoeuvreEvitement(){
    arreter();
    delay(200);

    int angle1 = 90;
    int angle2 = 0;
    int angle3 = -90;

    if(ChangerOrientationGyro){
      angle1 = 270;
      angle2 = 180;
      angle3 = 90;
    }

    // Séquence de manœuvre d'évitement
    tournerJusquA(angle1, true);   // tourner à gauche
    tournerJusquA(angle2, false);    // tourner à droite
    avancerToutDroitCorrige(300);  //Avancer un peu pour dépasser l'obstacle
    tournerJusquA(angle3, false);   // tourner à droite
    tournerJusquA(angle2, true);     // tourner à gauche

    arreter();
    delay(200);
```

```
    // Reprendre l'avance tout droit avec correction gyroscope (angle 0°)
    avancerToutDroitCorrige(400);
}

/*-----------------------------------ENCODEURS-----------------------------------*/
void tickGauche() {
  ticksGauche++;
}

void tickDroit() {
  ticksDroit++;
}

/*-----------------------------------LECTURE CAPTEUR ULTRASON-----------------------------------*/
long lireDistanceUltrason() {
  digitalWrite(triggerPin, LOW);
  delayMicroseconds(2);
  digitalWrite(triggerPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(triggerPin, LOW);
  long duree = pulseIn(echoPin, HIGH, 20000); // Timeout 20ms
  long distance = duree * 0.034 / 2;
  return distance;
}
```

```
/*-----------------------------------GYROSCOPE-----------------------------------*/
void mettreAJourYaw() {
  Wire.beginTransmission(MPU_ADDR);
  Wire.write(0x47);
  Wire.endTransmission(false);
  Wire.requestFrom(MPU_ADDR, 2, true);
  gyro_z_raw = Wire.read() << 8 | Wire.read();

  unsigned long current_time = millis();
  float dt = (current_time - last_time) / 1000.0;
  last_time = current_time;

  float gz = (gyro_z_raw / 131.0) - gyro_offset_z;
  if (abs(gz) > threshold) {
    angle_z += gz * dt;
  }
}

/*-----------------------------------CALIBRAGE GYROSCOPE-----------------------------------*/
void calibrateGyro() {
  long sum = 0;
  const int samples = 500;
  for (int i = 0; i < samples; i++) {
    Wire.beginTransmission(MPU_ADDR);
    Wire.write(0x47);
    Wire.endTransmission(false);
    Wire.requestFrom(MPU_ADDR, 2, true);
    int16_t raw = Wire.read() << 8 | Wire.read();
    sum += raw;
    delay(3);
  }
  gyro_offset_z = sum / (float)samples / 131.0;
}
```

Bibliographie

[1] **Doc 9981, PROCÉDURES POUR LES SERVICES DE NAVIGATION AÉRIENNE**,
érodromes, Troisième édition, 2020, publié par l'OACI (Organisation de l'Aviation Civile Internationale)

[2] **Systèmes de détection des objets sur les pistes d'aérodrome,**
Étude exploratoire, Rapport d'étude, Version : V1 du 22/04/2010 du service technique de l'Aviation Civile du Ministère de l'Écologie, du Développement durable, des Transports et du Logement français
Rédacteur : Pierre THERY ; Vérificateur : Frédéric FUSO ; Approbateur : Jean-Philippe DUFOUR

[3] **Advisory Circular, Airport Foreign Object Debris (FOD) Management**,
2/8/2024, AC No : 150/5210-24A, publié par U.S. Department of Transportation Federal Aviation Administration

[4] **9146 :2016 Ballot Draft FOREIGN OBJECT DAMAGE (FOD) PREVENTION PROGRAM Guidance Material**,
March 2016, publié par International Aerospace Quality Group (IAQG)

[5] *Pistes et Balisages, Article de lavionnaire.fr*
`https://www.lavionnaire.fr/PistesNormes`, consulté en Juin 2025.

[6] *Comment sont construits les aéroports, Article de ADA Ouèssè Amédé publié sur le blog SpaceAirMasters*
`https://spaceairmasters.com/blog/aeroport_working-35`, consulté en Juin 2025.

[7] FAA, *Foreign Object Debris (FOD) Program*,
`https://www.faa.gov/airports/runway_safety/fod`, consulté en Avril 2025.

[8] Glenn Jocher, *YOLOv5 by Ultralytics*,
`https://github.com/ultralytics/yolov5`, consulté en Mai 2025.