



Mise en garde

La bibliothèque du Cégep de l'Abitibi-Témiscamingue et de l'Université du Québec en Abitibi-Témiscamingue (UQAT) a obtenu l'autorisation de la personne autrice de ce document afin de diffuser, dans un but non lucratif, une copie de son œuvre dans [Depositum](#), site d'archives numériques, gratuit et accessible à tous. L'autrice ou l'auteur conserve néanmoins ses droits de propriété intellectuelle, dont son droit d'auteur, sur cette œuvre.

Warning

The library of the Cégep de l'Abitibi-Témiscamingue and the Université du Québec en Abitibi-Témiscamingue (UQAT) obtained the permission of the author to use a copy of this document for nonprofit purposes in order to put it in the open archives [Depositum](#), which is free and accessible to all. The author retains ownership of the copyright on this document.

Université Du Québec En Abitibi Témiscamingue

SYSTÈME INTELLIGENT DE SURVEILLANCE MINIÈRE
COMBINANT CAPTEUR ULTRASONIQUE ET YOLOV10 AVEC
NOTIFICATIONS EN TEMPS RÉEL ET INTERVENTION À DISTANCE

Mémoire
Présenté
Comme exigence partielle
de la maîtrise en ingénierie

Par
Hellas Abdenour

Juin 2025

REMERCIEMENTS

Je tiens avant tout à remercier Dieu le Tout-Puissant pour Sa grâce et Sa présence constante à mes côtés tout au long de ce parcours.

Ma profonde gratitude va à mon directeur de recherche, le Professeur **FOUAD SLAOUI HASNAOUI**, pour son orientation précieuse, sa motivation inlassable et le temps considérable qu'il m'a consacré. Sans son soutien et ses conseils avisés, la réalisation de ce mémoire n'aurait pu aboutir.

Je souhaite exprimer ma gratitude aux Professeurs Yassine Kali et Mourad Nedil pour avoir accepté d'évaluer ce travail. Mes remerciements s'adressent également à tous les professeurs du programme de maîtrise en ingénierie, concentration génie électrique.

Je suis reconnaissant envers la direction et l'ensemble du personnel de l'UQAT pour m'avoir offert cette opportunité d'apprentissage et d'épanouissement académique.

Un hommage particulier va à mes chers parents, **Nabile** et **Fahima** (Rahimaha Allah), ma source d'inspiration inépuisable. Leur soutien indéfectible, leur compréhension et leurs sacrifices ont été le moteur de ma persévérance. Aucun mot ne saurait traduire l'immensité de ma reconnaissance et de mon amour à leur égard. Je n'oublie pas non plus ma famille en Algérie (Zaki, Ayoub, Amira, Amel) pour son encouragement constant.

TABLE DE MATIÈRES

REMERCIEMENTS.....	II
TABLE DE MATIÈRES	III
LISTE DES FIGURES.....	VIII
LISTE DES TABLEAUX	XI
LISTE DES ABRÉVIATIONS	XII
RÉSUMÉ.....	XIII
ABSTRACT	XIII
INTRODUCTION GÉNÉRALE	1
PROBLÉMATIQUE	2
MOTIVATION ET OBJECTIFS	3
A.OBJECTIF GÉNÉRAL	3
B.OBJECTIFS SPÉCIFIQUES	3
ORIGINALITÉS	4
MÉTHODOLOGIE DE RECHERCHE	4
CHAPITRE 1 GÉNÉRALITÉ SUR LES RADARS.....	5
I INTRODUCTION	5
1 GÉNÉRALITÉS SUR LES RADARS	5
1.1. Historique	5
1.2. Définition du radar	6
1.3. Définition de la cible	6
2.PRINCIPE FONCTIONNEMENT D'UN RADAR :.....	7
3.LES ÉLÉMENTS DU RADAR	7
4.LES PRINCIPE MESURES.....	10
4.1. Mesure de distance.....	10
4.2. Mesure de l'angle	10
4.3. Mesure de la vitesse radiale	11
4.4. L'effet doppler.....	11
CHAPITRE 2 GÉNÉRALITÉ SUR LES RÉSEAUX DES NEURONES	12
1. GÉNÉRALITÉS SUR LES RÉSEAUX DE NEURONES	12
2. L'HISTOIRE DES RÉSEAUX DE NEURONES	12
3. APPLICATIONS.....	13
4. RÉSEAUX DE NEURONES CONVOLUTIFS (CNN).....	13
5. ARCHITECTURE DE CNN	14
6. FONCTIONNEMENT D'UN CNN	15
6.1. Couches convolutionnelles	15

6.2. Couche de Pooling (POOL)	16
6.3. La couche Fully Connected (FC)	17
6.4. Couche de perte (LOSS)	17
7. FONCTIONS D'ACTIVATION.....	17
7.1. Unité linéaire rectifiée (ReLU)	18
7.2. Unité linéaire rectifiée Leaky (Leaky ReLU)	18
7.3. La fonction sigmoïde	18
7.4. La fonction tangente hyperbolique	19
8. MÉTHODES CNN APPLIQUÉES À L'ANALYSE D'IMAGES ET À LA LOCALISATION D'OBJETS	19
8.1. Utilisation des CNN pour la reconnaissance d'images	19
8.2. Application des réseaux convolutifs pour l'identification d'objets.....	21
8.2.1. Modèles à deux étapes (two-stage detectors).....	21
8.2.2. Modèles à une étape	22
9. TECHNIQUES D'AMÉLIORATION DES CNN	23
9.1. Data augmentation	23
9.2. Dropout	23
9.3. La normalisation par lot (Batch normalization).....	24
9.4. Apprentissage par transfert	24
9.5. L'apprentissage par un seul exemple (one-shot) et sans exemple (zero-shot)	24
CHAPITRE 3 GÉNÉRALITÉ SUR L'ARCHITECTEUR YOLO	26
I. INTRODUCTION DE YOLO.....	26
1. DÉFINITION DE YOLOv10	27
2. ALGORITHME YOLO v10	27
2.1. Méthodologie	27
2.1.1. Formation et inférence NMS gratuites	27
2.1.2. Conception de modèles axée sur l'efficacité	32
2.1.3. Conception de modèle axée sur la précision	35
3. L'ARCHITECTURE	36
3.1. Variantes du modèle	38
3.2. Expériences et résultats.....	38
3.3. Comparaison des performances de YOLOv10	38
4. MISE EN ŒUVRE DE YOLOv10.....	40
4.1. Inférence YOLOv10-HuggingFace.....	40
4.2. Inférence - Interface de ligne de commande (CLI).....	41
4.3. Inférence-SDK Python.....	42
5. APPLICATIONS CONCRÈTES DE YOLOv10	43
6. COMPARAISON DES MÉTHODES DE DÉTECTION D'OBJETS (SSD, FASTER-RCNN)	44
CHAPITRE 4 GÉNÉRALITÉ SUR LE FILTRE DE KALMAN	47
I. INTRODUCTION AU FILTRE DE KALMAN.....	47
1. CONCEPTS DES BASES	47
1.1. Systèmes dynamiques	47
1.2. Un bruit blanc	48

2. DÉFINITION DE FILTRE DE KALMAN	48
3. MODÈLE DU FILTRE DE KALMAN	49
4. ALGORITHME DU FILTRE DE KALMAN	50
4.1. Cycle du filtre de kalman.....	50
4.1.1. Initialisation	50
4.1.2. Étape de prédiction.....	51
4.1.3. Étape de correction (Prise en compte de la mesure)	51
4.2. Expression des paramètres du filtre de Kalman.....	52
5. LES AVANTAGES DE FILTRE DE KALMAN	54
6. LES LIMITES DE FILTRE DE KALMAN	55
7. EXTENSION DE FILTRE DE KALMAN	56
7.1. Filtre de Kalman Etendu (EKF)	56
7.2. Filtre de Kalman sans parfum (UKF)	58
CHAPITRE 5 MATÉRIELLES ET LOGICIELS	60
1. SERVOMOTEURS PAS À PAS (MOTEUR PAS À PAS 28BYJ-48)	60
1.1. Définition	60
1.2. Les caractéristiques Moteur Pas à Pas 28BYJ-48	60
1.3. Fonctionnement du moteur pas à pas 28BYJ-48	61
1.4. Applications du moteur pas à pas 28BYJ-48	63
2. DRIVER ULN2003	63
2.1. Définition ULN2003	63
2.3. Fonctionnement.....	65
3. LE RASPBERRY PI 4 MODEL B	65
3.1. Les composants d'un Raspberry Pi 4 B	66
3.2. Les caractéristiques Raspberry Pi 4 Model B	67
4. CAMÉRA	68
4.1. types de camera.....	68
4.2. Les caractéristiques des caméras Raspberry Pi.....	68
4.2.1. Camera Module standard et NoIR	68
4.2.2. High Quality Camera (HQ Camera)	69
4.3. Fonctionnements du Camera Raspberry Pi 4.....	69
4.3.1. Installation du Camera Module sur Raspberry Pi 4	69
4.3.2. Tester la caméra sur Raspberry Pi	71
4.3.3. Contrôler le module caméra avec du code Python (picamera).....	72
4.3.4. Capture d'images fixes avec la caméra Raspberry Pi.....	73
4.3.5. Capture de vidéos avec la caméra Raspberry Pi	73
5. GPS DE LA CARTE RASPBERRY PI.....	74
5.1. Les caractéristique de GPS	75
5.2. brochage du module gps neo-6m (pinout)	76
5.3. Connexion du récepteur GPS et du Raspberry Pi	77
5.4. Configuration du Raspberry Pi 4 pour le GPS.....	77
5.5.1. Utilisation de Python.....	79

5.5.2. Utilisation de C.....	80
7. CAPTEUR ULTRASONS HC-SR04	80
7.1. Caractéristiques et spécifications du capteur HC-SR04	81
7.2. Broches de connexion.....	81
7.3. Fonctionnement du capteur Ultrason HC-SR04	81
7.3.1. Mesure de la distance de la cible.....	82
7.3.2. Le diagramme temporel	83
8.1. Types de modules de buzzer	85
8.1.1. Buzzer passif	85
8.1.2. Buzzer actif	87
CHAPITRE 6 RÉALISATION DU SYSTÈME	90
I. INTRODUCTION	90
1. PRINCIPE DE FONCTIONNEMENT	90
2. LES ÉTAPES DU PROJET	90
3. SCHÉMA SYNOPTIQUE DE RADAR	91
4. ALGORITHME PRINCIPALE DE DÉTECTION ET CLASSIFICATION D'UN OBJET.....	92
5. SIMULATION DU PROJET SOUS PROTEUS.....	93
5.1. Bibliothèque carte la carte Raspberry pi pour Proteus 8.....	93
5.2. Bibliothèque de capteurs à ultrasons pour PROTEUS 8	93
5.3. Bibliothèque de GPS pour Proteus 8	95
5.4. Configurer la caméra pour PROTEUS 8	96
6. RÉSULTATS DE LA SIMULATION.....	99
<u>RÉALISATION PRATIQUE DU PROJET.....</u>	103
1. ALIMENTATION ÉLECTRIQUE	104
2. DÉMARREZ MEDIA	105
3. CLAVIER.....	106
4. SOURIS.....	107
5. RÉSEAUTAGE	107
6. INSTALLER UN SYSTÈME D'EXPLOITATION	108
7. INSTALLATION D'IMAGE	108
8. PERSONNALISATION DE L'OS	111
9. CONNECTER LA CARTE RASPBERRY PI 4 AVEC LE RÉSEAU	115
10. CONFIGURATION AU PREMIER DÉMARRAGE.....	115
11. CONFIGUREZ NOTRE RASPBERRY PI	116
12. RASPBERRY PI CONNECT	118
13. CONTRÔLE À DISTANCE SUR LE RÉSEAU LOCAL.....	118
14. ÉTAPES DE LA PROCÉDURE :.....	122
14.1. Choix de l'Objet :	122
14.2. Cas d'Objet Fixe :	123
14.2.1. Erreurs	124
14.2.2. Calcul des Pourcentages.....	124
14.3. Cas d'Objet Mobile :	125

15.MISE EN ŒUVRE DE L'ARCHITECTURE YOLO POUR LA DÉTECTION ET LA CLASSIFICATION D'OBJETS.....	127
RÉSULTATS ET DISCUSSION	132
CONCLUSION GÉNÉRALE ET PERSPECTIVE	139
RÉFÉRENCE	141
ANNEXE	147

LISTE DES FIGURES

Figure1	Principe fondamental de l'opération radar.	7
Figure2	Les éléments du radar	9
Figure3	Technique d'évaluation des distances radars.....	10
Figure 4	Mécanisme de mesure des angles.	11
Figure 5	Chronologie des avancées majeures en réseaux neuronaux	12
Figure 6	Structure type d'un réseau neuronal convolutif (5 couches)	14
Figure 7	Les Réseaux de Neurones Convolutifs (CNN)	15
Figure 8	Résultat de l'application d'un filtre convolutif	16
Figure 9	Comparaison des techniques de pooling : maximal vs moyen	17
Figure 10	Allure de la fonction ReLU.....	18
Figure 11	différents fonctions d'activation et leurs graphes.	19
Figure 12	Flux du travail NMS.	28
Figure 13	Flux de travail du modèle YOLOv10.	28
Figure 14	Attribution d'une double étiquette.	29
Figure 15	Flux de travail de correspondance des mesures.	30
Figure 16	Présentation de l'architecture de YOLO.....	32
Figure 17	Processus de convolution.....	33
Figure 18	Conversion en profondeur et en point (source-milieu).	34
Figure 19	Bloc CIB.	35
Figure 20	Différentes formes de noyau utilisées dans la convulsion des modèles YOLO.	35
Figure 21	Bloc PSA.....	36
Figure 22	L'architecteur de YOLO.	37
Figure 23	Variations du modèle YOLOv10.....	38
Figure 24	YOLOv10 hugging espace visage.	40
Figure 25	Inférence CLI YOLOv10 avec le notebook colab.	41
Figure 26	Le résultat de l'inférence CLI.....	42
Figure 27	Inférence du SDK python YOLOv10.	42
Figure 28	Une architecture standard des détecteurs d'objets à une seule étape.	45
Figure 29	Représentation de l'autocorrélation du bruit blanc : domaine temporel (gauche) et fréquentiel (droite)	48
Figure 30	Processus itératif du filtre de Kalman.....	50
Figure 31	Moteur Pas à Pas 28BYJ-48.	60
Figure 32	The pinot.	62
Figure 33	Fonctionnement du moteur pas à pas 28BYJ-48	62
Figure 34	Les composants de driver ULN2003	64
Figure 35	La carte Raspberry Pi 4 type B	65
Figure 36	Les composants Raspberry Pi 4 type B	66
Figure 37	Camera Module standard	69
Figure 38	High Quality Camera	69
Figure 39	Trouve le port du module caméra.	70
Figure 40	(a) Vérifie que ton Raspberry Pi est éteint, (b) Insère la nappe caméra	70
Figure 41	Remets le clip en plastique en place	71

Figure 42	Configuration du Raspberry Pi.	71
Figure 43	GPS Module.....	74
Figure 44	Pinout de GPS	77
Figure 45	Connexions entre le récepteur GPS et le Pi 4 .	77
Figure 46	Capteur Ultrason HC-SR04.	80
Tableau 6	Spécifications techniques du capteur HC-SR04.	81
Figure 47	Mécanisme de détection du module HC-SR04.	82
Figure 48	Diagramme chronologique des signaux.	83
Figure 49	La piézoélectricité.....	84
Figure 50	Buzzer passif	85
Figure 51	Des images du buzzer passif.	85
Figure 52	Un circuit inverseur à push-pull	86
Figure 53	Implémentation d'un buzzer piézoélectrique dans un dispositif porte-clé	86
Figure 54	Le principe d'un buzzer actif .	87
Figure 55	Le buzzer piézoélectrique actif .	87
Figure 56	Des images du buzzer actif	88
Figure 57	Connexion buzzer actif au Raspberry Pi	89
Figure 58	Configuration minimale à 2 fils pour connexion GPIO	89
Figure 59	Schéma-bloc du projet radar expérimental	91
Figure 60	Logique algorithmique centrale du système	92
Figure 61	L'intégration de la carte Raspberry pi 4 sous Proteus.....	93
Figure 62	Simulation du capteur HC-SR04 dans logiciel Proteus 8.	94
Figure 63	Fichier hexadécimal UltrasonicTEP dans Proteus 8.	94
Figure 64	Le GPS sous Proteus 8.....	95
Figure 65	Fichier GpsTSEP.HEX sous Proteus 8.	95
Figure 66	La caméra sous Proteus 8.....	96
Figure 67	Control camera sous Proteus 8.....	96
Figure 68	Circuit électronique globale de simulation.	97
Figure 69	(a),(b)et(c) Circuit électronique globale de simulation.....	101
Figure 70	les conditions de fonctionnement les LEDs.....	101
Figure 71	Représentation des impulsions Trigger et Echo du capteur à ultrasons.....	101
Figure 72	Teste fonctionnement de caméra.....	102
Figure 73	Le teste pour le fonctionnement du GPS	103
Figure 74	Alimentation de carte Raspberry pi	105
Figure 75	Démarrer le media	105
Figure 76	Installation d'un clavier	106
Figure 77	Installation d'une souris.....	107
Figure 78	Installation d'un réseau	107
Figure 79	les étapes d'installation d'image (a,b,c,d,e,f,i,g,k,l,m,n) .	108
Figure 80	La configuration du la carte Raspberry pi	116
Figure 81	Insertion de la carte SD dans Raspberry pi	116
Figure 82	Circuit de système.....	117
Figure 83	Connection a wifi.....	119

Figure 84	Terminal python de la programmation.....	119
Figure 85	Connexion a application RealVNC Viewer.	120
Figure 86	Interface de radar	121
Figure 87	Le système prêt à détecter.....	122
Figure 88	Test capture sur un objet fixe et mobil.....	122
Figure 89	Les résultats de détection d'un objet fixe.	123
Figure 90	Les résultats avant et après filtrage de détection d'un objet fixe.....	123
Figure 91	Les résultats de détection d'un objet Mobile.	125
Figure 92	Les résultats avant et après filtrage de détection d'un objet mobil.....	126
Figure 93	Teste l'envoi des messages via Telegram	127
Figure 94	Teste de l'architecteur yolov10n sur la Raspberry pi.....	128
Figure 95	L'environnement COLAB.	129
Figure 96	Modèle pré entraîné.	130
Figure 97	L'entraînement de modèle.....	131
Figure 98	Training Results	132
Figure 99	Évolution des pertes pendant l'entraînement	133
Figure 100	Évolution de la précision pendant l'entraînement	134
Figure 101	Évolution du mAP50 pendant l'entraînement	135
Figure 102	Évolution du rappel pendant l'entraînement	136
Figure 103	Évolution du mAP50-95 pendant l'entraînement.....	137
Figure 104	Résultat d'une prédiction sur des image téléchargées	138
Figure 105	différents types des radars.....	147
Figure 106	Angles caractérisant une cible radar	148
Figure 107	Chronogramme de l'émission d'une impulsion radar	148
Figure 108	Évaluation des distances avec radar FMCW.	150
Figure 109	FMCW radar système.	151
Figure 110	Synthèse du faisceau total.	153
Figure 111	Représentations graphiques de différents modes RSO.	154
Figure 112	Schéma fonctionnel d'un radar secondaire	155
Figure 113	Équations différentielles de puissance PSR vs SSR	155
Figure 114	Principe de l'antenne réseau et quelques réalisations phased arrays.....	161
Figure 115	Composants essentiels d'un système de communication radar.	165
Figure 116	Classification des interférences dans un récepteur radar.	168
Figure 117	Analyse spectrale : signal impulsionnel et bruit blanc	168
Figure 118	Évolution du rapport signal/bruit selon la bande passante.....	169
Figure 119	Les applications radar.	170

LISTE DES TABLEAUX

Tableau 1	Comparaison des détecteurs d'objets en temps réel sur MS COCO (latence, précision et taille des modèles)	39
Tableau 2	Tableau Les caractéristiques Moteur Pas à Pas 28BYJ-48	61
Tableau 3	Les caractéristiques Raspberry Pi 4 Model B	67
Tableau 4	Les caractéristiques d'un module GPS.....	76
Tableau 5	Correspondance des broches et connexions du circuit	98

LISTE DES ABRÉVIATIONS

Radar	Radio-Detection-and-Ranging.
IEEE	Institute-of-Electrical-and-Electronics-Engineers.
MTI	Moving-Target-Indicator.
CW	Continuous-Wave.
FMCW	Frequency-Modulated-Continuous-Wave.
PRF	Pulse-Repetition-Frequency.
SNR	Signal-Noise-Ratio.
LED	Light-Emitting-Diode
USB	Universal-Asynchronous-Receiver-Transmitter
PWM	Pulse-width-modulation
RAM	Random-access-memory
EEPROM	Electrically-erasable-programmable-read-only-memory
Gnd	Ground
RX	Receive/Receiver/Reception
SPI	Serial-Peripheral-Interface
LCD	Liquid-Crystal-Display
RCO	Rapport-Cyclique-d'Ouverture
PC	Personal-Computer
PD	Power-Down
DC	Digit-Carry
YOLO	You Only Look Once (Architecture de détection en un seul passage)
v10	Version 10 (2024, dernière évolution officielle)
N/S/M/L/X	Tailles de modèles (Nano, Small, Medium, Large, XLarge)
AP	Average Precision (Précision moyenne sur COCO)
Map	mean Average Precision (mAP@0.5:0.95)
FLOPs	Floating Point Operations (Complexité calcul)
FPS	Frames Per Second (Débit en temps réel)
NMS	Non-Maximum Suppression (Suppression des détections redondantes)
CSP	Cross-Stage Partial (Optimisation du backbone)
PAN	Path Aggregation Network (Fusion multi-échelle)
COCO	Common Objects in Context (Dataset de référence)
IoU	Intersection over Union (Précision des boîtes)
TTA	Test-Time Augmentation (Boost de précision)

RÉSUMÉ

L'évolution technologique récente a favorisé le développement de systèmes de détection intelligents, notamment dans des domaines où la sécurité et la surveillance sont critiques. Ce projet met en œuvre un radar à ultrasons contrôlé par une carte Raspberry Pi B+, intégrant un système de détection basé sur YOLO (You Only Look Once) pour la classification d'objets. Le système envoie des notifications via Telegram, avec des alertes sonores (bips) et visuelles (LED) activées selon la distance de détection. Un filtre de Kalman améliore la précision des données en éliminant le bruit et en optimisant la détection. Le système permet également une intervention à distance, couplée à une caméra infrarouge pour une surveillance efficace dans des conditions complexes. Cette solution propose une détection et une classification d'objets en temps réel, avec une interface utilisateur intuitive et réactive.

Mots-clés : Radar ultrasonique, Raspberry Pi, YOLO, classification d'objets, contrôle à distance, caméra infrarouge, notifications sonores et visuelles, détection intelligente.

ABSTRACT

Recent technological advances have enabled the development of smart detection systems, particularly in security and surveillance applications. This project implements an ultrasonic radar controlled by a Raspberry Pi B+, integrating YOLO (You Only Look Once) object detection. The system sends Telegram notifications with audio (beeps) and visual (LED) alerts triggered by detection distance. A Kalman filter improves data accuracy by reducing noise. The solution features remote intervention capability and an infrared camera for effective monitoring in complex conditions, offering real-time object detection/classification with an intuitive user interface.

Keywords: Ultrasonic radar, Raspberry Pi, YOLO, object classification, remote control, infrared camera, sound and visual notifications, intelligent detection.

INTRODUCTION GÉNÉRALE

Les dispositifs radar, dont le nom provient de l'anglais Radio Detection and Ranging, sont des technologies de détection et de mesure qui utilisent des ondes électromagnétiques pour identifier des objets, mesurer des distances et surveiller des mouvements. Depuis leur création dans les premières décennies du 20ème siècle, les radars ont connu une expansion multidisciplinaire, s'intégrant aussi bien dans l'aviation et l'observation climatique que dans les systèmes de défense et les véhicules modernes. Leur capacité à fonctionner dans des conditions de faible visibilité, comme le brouillard, la pluie ou l'obscurité, en fait des outils indispensables pour la surveillance et la sécurité dans des environnements complexes [1].

Dans le contexte industriel, les radars ont évolué pour répondre à des besoins spécifiques, notamment dans des secteurs comme l'exploitation minière. Parmi les différentes variantes de radars, les radars ultrasoniques se distinguent par leur utilisation d'ondes sonores à haute fréquence plutôt que d'ondes radio. Ces systèmes sont particulièrement adaptés aux environnements confinés ou difficiles, où la précision et la fiabilité sont essentielles. Par exemple, dans les mines, où la poussière, l'humidité et les espaces restreints rendent la visibilité limitée, Les systèmes à ultrasons radar constituent une alternative performante pour le suivi des déplacements géologiques, l'identification d'obstacles et la protection du personnel [2].

Leur mécanisme repose sur l'émission d'ondes ultrasonores qui, en se réfléchissant sur les obstacles, permettent de déterminer avec exactitude leur position spatiale, leur distance et leur vitesse. Cette capacité à fournir des données précises en temps réel les rend particulièrement utiles dans des environnements dynamiques et à haut risque, comme les mines à ciel ouvert ou souterraines. Par exemple, ils peuvent détecter les instabilités des parois rocheuses, surveiller les déplacements des engins miniers ou identifier la présence de travailleurs dans des zones dangereuses [3, 4].

Cependant, pour maximiser leur efficacité, les radars ultrasoniques sont souvent couplés à des technologies d'intelligence artificielle, Similaire au modèle YOLO (You Only Look Once), offrant simultanément détection instantanée et classification précise des éléments visuels. Cette combinaison de technologies ouvre de nouvelles perspectives pour l'automatisation et la sécurité dans l'industrie minière, en fournissant des outils puissants pour prévenir les accidents, optimiser les opérations et améliorer la productivité [5].

Ce mémoire de fin de cycle explore l'application des radars, et plus spécifiquement des radars ultrasoniques, dans le domaine minier. Il met en lumière leur intégration avec des systèmes d'intelligence artificielle comme YOLO pour améliorer la sécurité au travail, automatiser les processus et offrir des bénéfices tangibles aux employeurs. Notre analyse s'appuie sur des travaux récents concernant l'utilisation combinée des technologies radar et d'intelligence artificielle dans le secteur minier.

PROBLÉMATIQUE

Le secteur minier, notamment les exploitations en surface, doit relever d'importants défis concernant la protection du personnel et l'optimisation opérationnelle. Les incidents comme les éboulements, les projections rocheuses et les accidents d'engins surviennent régulièrement, engendrant des impacts humains dramatiques, des pertes de rendement et des surcoûts opérationnels. Malgré les progrès technologiques, les méthodes traditionnelles de surveillance et de détection des dangers restent limitées : elles reposent souvent sur des inspections manuelles, des systèmes de détection peu précis et une intervention humaine intensive, ce qui entraîne des retards dans la détection des risques et une dépendance excessive à l'égard de l'opérateur.

Cette situation soulève plusieurs problématiques clés :

- Comment améliorer la détection des dangers dans les mines à ciel ouvert pour prévenir les accidents et protéger les travailleurs?
- Quelles solutions technologiques offrent une supervision immédiate des zones sensibles, notamment les versants instables, les chemins miniers et les espaces de travail?
- Comment automatiser les processus de surveillance pour réduire les erreurs humaines et améliorer l'efficacité opérationnelle?
- Quel est le potentiel des radars ultrasoniques et de l'architecture YOLO pour répondre à ces défis, en combinant précision, rapidité et adaptabilité aux conditions difficiles des mines ?

MOTIVATION ET OBJECTIFS

Dans l'industrie minière contemporaine, la sécurité des travailleurs et l'efficacité des opérations sont des enjeux majeurs. Les accidents, notamment les instabilités géologiques, les détachements de masses rocheuses ou les accidents d'engins miniers, peuvent sérieusement affecter la vie humaine et la productivité. Par conséquent, il est impératif d'adopter des méthodes plus efficaces pour repérer les dangers, surveiller en temps réel et prévenir les accidents. La motivation principale de cette étude réside dans l'impact significatif que pourrait avoir un système de surveillance intelligent, basé sur les radars ultrasoniques et l'architecture YOLO (You Only Look Once), en termes de réduction des risques, d'optimisation des opérations et de diminution des coûts liés aux accidents. L'une des caractéristiques essentielles de cette recherche est l'analyse des données de surveillance en temps réel, et leur utilisation dans la détection, la classification et la localisation des dangers dans les mines à ciel ouvert.

a. Objectif général

Cette étude vise à développer et déployer une solution innovante de monitoring intelligent, combinant des capteurs ultrasonores et la structure algorithmique YOLO, capable de détecter, classer et localiser les dangers dans les mines à ciel ouvert, afin d'améliorer la sécurité des travailleurs et l'efficacité des opérations.

b. Objectifs spécifiques

1. Mesurer l'efficacité du système à travers sa justesse de détection, sa réactivité et sa robustesse lors d'implémentations in situ dans des mines actives.
2. Intégrer un système d'alerte en temps réel pour notifier les travailleurs et les superviseurs en cas de détection de danger.
3. Créer une interface ergonomique permettant la visualisation instantanée des informations de monitoring et des notifications d'alerte.
4. Analyser le retour sur investissement du dispositif, notamment ses effets sur la diminution des dépenses accidentelles et l'amélioration du rendement opérationnel.
5. Proposer des recommandations pour l'intégration du système dans d'autres types de mines (souterraines, par exemple) ou dans d'autres industries à haut risque.
6. Analyser les limites du système et proposer des pistes d'amélioration pour de futures recherches.
7. Améliorer la robustesse du système pour qu'il fonctionne dans des conditions extrêmes (poussière, humidité, températures variables).

ORIGINALITÉS

La valeur ajoutée de cette recherche repose sur l'intégration inédite de technologies ultrasonores et d'architecteur YOLO, spécialement adaptée aux problématiques complexes des exploitations minières de surface (à ciel ouvert). Alors que ces technologies sont souvent utilisées séparément dans d'autres domaines, leur intégration dans un système de surveillance intelligent pour les mines représente une approche novatrice. Ce projet se démarque également par son approche multidisciplinaire, qui intègre l'ingénierie, l'intelligence artificielle et la sécurité industrielle pour fournir une solution pratique et efficace. Enfin, en proposant une méthodologie basée sur la validation sur un labo de l'Université UQAT et en visant un impact concret sur la sécurité et la productivité, ce projet apporte une contribution originale à la littérature scientifique et à l'industrie minière.

MÉTHODOLOGIE DE RECHERCHE

Notre approche principale pour atteindre le but et les objectifs du ce projet est constitué de cinq phases.

La première phase est la revue de littérature et la collecte des données : Analyser les recherches existantes et rassembler les informations nécessaires pour définir le cadre du projet.

Ensuite, Acquisition de matériel et prototypage : Sélectionner et assembler les composants matériels et logiciels nécessaires à la mise en place du prototype initial.

Programmation et développement du prototype : Coder les fonctionnalités principales du système radar, y compris l'affichage et l'interface utilisateur.

Intégration des algorithmes et entraînement du modèle : Intégrer des modèles d'IA pour l'identification d'objets et optimiser leur apprentissage par des scénarios virtuels (simulations).

Enfin, tester en temps réel et la validation : Tester le système dans des environnements réels, ajuster les paramètres et valider les résultats en comparant les performances obtenues aux attentes.

CHAPITRE 1 GÉNÉRALITÉ SUR LES RADARS

I Introduction

Le système radar est un cadre automatisé capable de détecter des objets dans son ensemble d'émission et de réception. Les systèmes radar se présentent sous différentes formes et tailles, chacun ayant son propre ensemble de capacités[6], Cette section présentera une synthèse globale du concept radar, abordant son évolution historique, ses mécanismes opérationnels, ses composants structurels et ses typologies.

1 Généralités sur les radars

1.1. Historique :

L'évolution des radars trouve son origine dans les avancées en électronique. Dès 1864, les travaux fondateurs de James Clerk Maxwell sur les principes électromagnétiques ont ouvert la voie au développement des premières sources d'ondes [7]. Puis en 1886, L'histoire du radar a commencé grâce aux les expériences du physicien allemand "Heinrich Hertz", Ces expériences ont prouvé que les objets métalliques étaient capables de réfléchir les ondes radioélectriques.

L'essor des transmissions sans fil (TSF) au début des années 1900 a conduit au développement d'antennes adaptées à la radiodétection. Cette période a vu de nombreux chercheurs et ingénieurs participer à l'évolution des systèmes radar. Dès 1904, Christian Hülsmeyer démontra la faisabilité de repérer des navires dans un épais brouillard grâce aux ondes hertziennes[8] .

Durant l'entre-deux-guerres (1920-1939), les essais de détection par antennes radioélectriques connaissent une accélération marquée. La France innove en 1934 avec un brevet pionnier sur la détection en ondes courtes, conduisant à l'installation des premiers radars décimétriques sur des navires - l'Oregon (1934) puis le Normandie (1935). Ces avancées permettent à Robert Watson-Watt d'établir le premier système radar opérationnel [9].

L'année 1939 marque une révolution technologique avec la création du magnétron, un dispositif à cavité résonante permettant la génération d'ondes centimétriques et millimétriques pour les systèmes radar. À l'aube du conflit mondial, le Royaume-Uni dispose déjà d'un système de surveillance radar opérationnel, assurant la protection de son territoire contre les menaces aéronavales.[9] .

C'est durant la deuxième guerre mondiale que le système radar a véritablement gagné en crédibilité. Cette époque a vu naître des avancées radars fondamentaux dont plusieurs applications restent pertinentes actuellement, notamment les systèmes de surveillance aéroportée et les dispositifs de suivi des cibles mobiles (technologie MTI - Moving Target Indicator). L'après-guerre a accéléré le perfectionnement des radars, avec l'apparition dans les années 1950 d'une innovation majeure : le radar à synthèse d'ouverture (SAR). Cette évolution a représenté un bond technologique majeur par rapport aux premiers concepts de Hülsmeyer. Dans les années 1950, le radar à synthèse d'ouverture (SAR) révolutionne la détection, offrant une précision inédite. Les années 1960 voient naître les radars à réseaux phasés, dynamisant la recherche radar. Les avancées en traitement numérique du signal (années 1970) marquent le début d'une ère moderne, avec des algorithmes adaptatifs. Aujourd'hui, les radars multi-émetteurs, inspirés des télécommunications, ouvrent de nouvelles pistes d'innovation[5].

1.2. Définition du radar :

La technologie RADAR (pour Radio Detection And Ranging) s'appuie conceptuellement sur trois caractéristiques essentielles qui définissent ce dispositif de sondage par ondes radioélectriques :

- ❖ **Radio** : Electromagnétique ou ondes radio.
- ❖ **Detection** : Cible.
- ❖ **Ranging** : Localisation en quatre dimensions.

Dispositif exploitant des ondes électromagnétiques, le radar permet d'identifier, positionner et tracer des cibles. Son principe repose sur l'émission d'ondes radio qui, après réflexion sur les objets, fournissent des données précises sur leur distance, vitesse et trajectoire, notamment pour le suivi aérien et maritime. [10]

1.3 Définition de la cible :

Un objet détectable par un radar est appelé une cible, en raison de sa capacité à réfléchir les ondes électromagnétiques. L'intensité de l'énergie renvoyée vers le radar varie considérablement et cette détection dépend significativement de l'orientation relative entre la cible et le radar. Les objets surveillés présentent des caractéristiques physiques hétérogènes (taille, géométrie, matériaux) et des états cinématiques variables (statiques ou mobiles). Une analyse approfondie de leurs propriétés est essentielle pour optimiser la performance et la configuration des systèmes radar.[11]

2.Principe fonctionnement d'un radar :

Le radar fonctionne sur un principe comparable à celui de l'écho sonore. Par exemple, lorsque vous criez en direction d'une surface réfléchissante, comme une paroi rocheuse, vous entendez un écho. En connaissant La mesure du délai de propagation sonore permet une estimation précise de la distance et de la localisation. Selon un principe comparable, les radars déterminent l'intervalle temporel entre l'émission et la réception d'une OEM réfléchi par une cible. Ce temps, combiné à la vitesse connue de l'OEM (environ 300 000 km/s), permet de calculer la distance de l'objet.

Le radar remplit ainsi deux fonctions principales : la détection d'obstacles (Détection) et la mesure de distance (Télémétrie). Ces fonctionnalités constituent le fondement opérationnel du système RADAR (Détection et Télémétrie par Radio), comme le démontre schématiquement la Figure 1[12].

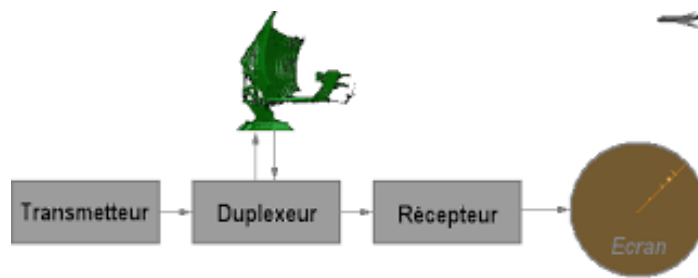


Figure 1
Principe fondamental de l'opération radar.[12]

3.Les éléments du radar

L'architecture dans figure 2, [13] Expose les constituants clés d'un radar. Pour son opération normale, l'appareil requiert sept modules fonctionnels principaux, précisément :

❖ Emetteur :

L'émetteur se compose de 2 parties fondamentales le :

- Le module d'émission, chargé de générer ou d'amplifier l'onde radar ;
- L'unité de modulation, alimentant l'étage émetteur en énergie et signaux de contrôle.[14]

❖ Le duplexeur :

Ce système de commutation électronique assure un routage efficace de l'onde émise vers l'antenne (faibles pertes) tout en garantissant l'isolation du module de réception. Inversement, il guide

intégralement le signal reçu vers le récepteur, sans déviation vers l'émetteur et en limitant les pertes. La conception du système s'adapte aux exigences énergétiques (échelle kW-MW) selon la portée radar. L'optimisation critique porte sur l'isolation du récepteur, vulnérable au-delà de 100 mW avec risques de saturation et détérioration permanente.[14]

❖ L'antenne :

Élément le plus caractéristique d'un radar, l'antenne assure la focalisation du rayonnement électromagnétique selon un diagramme de directivité précis, défini par ses paramètres géométriques et électriques. La direction de cet angle peut être modifiée mécaniquement ou électroniquement, permettant ainsi au radar d'explorer son environnement. [14]

Dans certains systèmes, une antenne double est utilisée pour séparer les fonctions d'émission et de réception. Ces deux éléments doivent être parfaitement alignés en permanence, soit par une liaison mécanique, soit par une synchronisation précise. De plus, il est crucial de minimiser les interférences radioélectriques pour éviter que le signal émis, de forte puissance, ne perturbe le récepteur. [14]

Cependant, la complexité et le coût élevé des antennes doubles ont conduit à privilégier, dans la plupart des cas, une antenne unique associée à un duplexeur. Cette approche optimise conjointement les phases d'émission et de réception, offrant un rapport performance/coût amélioré.[14]

❖ Le récepteur :

Le récepteur radar est l'un des composants les plus sensibles et complexes du système. Sa fonction consiste à amplifier les signaux reçus, éliminer les interférences par filtrage, et effectuer le traitement numérique des données radar. Sa sensibilité est extrêmement élevée, pouvant détecter des signaux de l'ordre de 10^{-15} Watts, tout en offrant une amplification considérable (de 10^{10} à 10^{14}). En plus du filtrage, il effectue des traitements spécifiques pour extraire des informations telles que la vitesse ou la position angulaire. Sa conception exige une grande précision pour limiter les distorsions et réduire les interférences causées par le bruit. [14]

Le récepteur opère suivant une architecture superhétérodyne, typiquement couplée à un préamplificateur RF. Sa configuration finale peut intégrer soit un détecteur d'enveloppe basique,

soit des circuits avancés. En phase terminale, un amplificateur BF élève le signal traité à une tension opérationnelle (quelques volts) via un étage vidéo.[14]

- ❖ Un étage de traitement (Exploitation et synchronisation) :
- ❖ Ce module assure le traitement des signaux bruts pour en extraire les informations pertinentes destinées à l'utilisateur. L'ensemble des composants radar est interconnecté via divers canaux de transmission (signaux) [13]:

- Signaux de synchronisation.
- Signaux de commande émission.
- Signaux hyperfréquence émis.
- Liaison duplexeur antenne.
- Signaux hyperfréquences reçus.
- Signaux traités par le récepteur.
- Information angulaire de pointe.

- ❖ Le modulateur :

Le modulateur assure une fonction vitale en alimentant la section active de l'émetteur. Il accumule l'énergie durant les phases inter-pulsées avant de la restituer instantanément lors de l'émission radar. Les puissances crêtes atteintes sont considérables (typiquement 10 MW pour une émission utile de 3 MW à 30% de rendement), nécessitant des courants de plusieurs dizaines d'ampères sous des tensions de dizaines de kV. Un contrôle rigoureux des signaux modulés est impératif pour préserver l'intégrité du signal radar.[14]

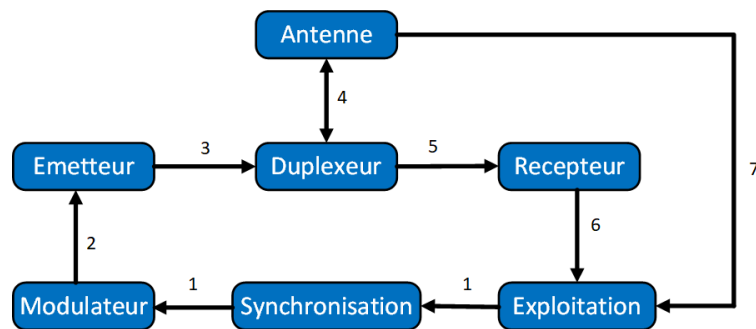


Figure2
Les éléments du radar [16]

4. Les principes mesures

Détecter une cible est peu utile si l'on ne peut pas obtenir d'informations la concernant. Inversement, disposer d'informations sur une cible sans être capable de la détecter n'a aucune valeur.

4.1. Mesure de distance

L'onde radar se propage à la célérité lumineuse sur la distance R_0 séparant l'émetteur de la cible. Au contact de l'objet, elle subit une diffusion omnidirectionnelle. La fraction rétrodiffusée effectue le trajet inverse (R_0) jusqu'au récepteur radar, comme schématisé figure 2. L'examen de cet écho révèle la présence de la cible et détermine sa distance via le temps de vol électromagnétique. Le signal retour correspond à une réplique atténuée et temporellement décalée de l'impulsion originale[15, 16].

$$\tau = \frac{R_0}{c} \quad (1)$$

Avec $C = 3.10^8 \text{ m/s}$ L'évaluation de ce délai de propagation fournit directement la distance radiale R_0 de la cible détectée.

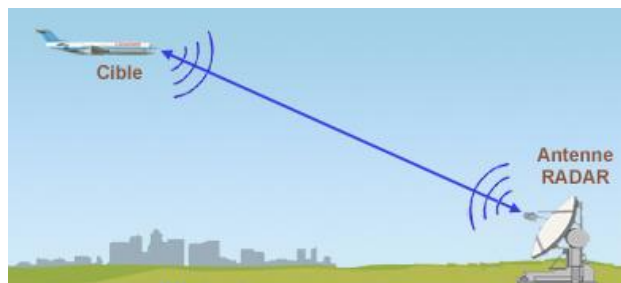


Figure 3
Technique d'évaluation des distances radars[17]

4.2. Mesure de l'angle

L'azimut cible est obtenu par calcul de l'angle entre l'axe nord et sa position, utilisant la directivité antennaire comme référence d'orientation lors de la réception d'écho. Cette technique permet aussi d'estimer l'élévation (site). La finesse de mesure angulaire est directement liée au diagramme de rayonnement de l'antenne, déterminé par ses dimensions physiques pour une longueur d'onde donnée.[17] Comme la figure 4 présente.

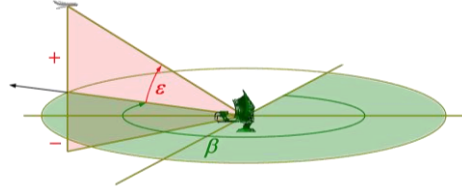


Figure 4
Mécanisme de mesure des angles. [17]

4.3. Mesure de la vitesse radiale

La détermination de la composante radiale de vitesse repose sur deux méthodes principales : soit en utilisant le décalage causé par l'effet Doppler, soit en mesurant successivement la distance. Bien que la seconde méthode nécessite un temps de mesure plus long pour atteindre une précision satisfaisante, elle reste la solution privilégiée [17].

4.4. L'effet doppler

Dans le cas d'un déplacement cible-radar, le temps de transit des ondes présente une évolution temporelle

$$\tau = \frac{2R_0(t)}{c} = \frac{2}{c}(R_0 + V_{rad}t) \quad (2)$$

$V_{(red)}$ quantifie la vitesse relative selon l'axe radar-cible, indépendamment du sens de déplacement. Les cibles mobiles induisent une modulation temporelle différentielle des échos. Quand $V_{(red)} \ll c$, cette modulation apparaît comme un shift fréquentiel de $V_{(red)}$ [16]. L'effet Doppler se manifeste alors par une fréquence résiduelle f_{D} calculable via [16]:

$$f_D = \frac{2V_{rad}}{\lambda} \quad (3)$$

Où $\lambda = \frac{c}{f_0}$ pour la longueur d'onde du système radar. La mesure f_D d'autoriser de déterminer la vitesse radiale de la cible.

Donc le système radar joue un rôle clé dans la sécurité et la surveillance en offrant une détection fiable des cibles mobiles ou statiques, même dans des conditions environnementales. Son couplage avec des algorithmes de traitement du signal et l'IoT en fait un outil indispensable pour les domaines complexes, et pour plus de détails sur les systèmes radars voire l'Annex page 146

CHAPITRE 2 GÉNÉRALITÉ SUR LES RÉSEAUX DES NEURONES

1 Généralités sur les réseaux de neurones

Les réseaux neuronaux, composés d'unités artificielles interconnectées, proposent une approche innovante pour résoudre des défis cognitifs comme la perception ou l'apprentissage. Leur architecture parallèle et bio-inspirée leur confère des capacités émergentes, en faisant des solutions pertinentes là où les systèmes classiques montrent leurs limites.[18]

Plusieurs appellations coexistent pour désigner les réseaux neuronaux artificiels, notamment connexionnisme et neuromimétique. Une clarification s'impose : si les réseaux de neurones artificiels correspondent aux modèles concrets, ils ne constituent pas une discipline à part entière. Le connexionnisme et la neuromimétique, en revanche, forment deux champs de recherche distincts analysant ces architectures, mais avec des finalités différentes. Les spécialistes du connexionnisme visent principalement à augmenter les performances computationnelles via des systèmes hautement interconnectés [19].

2 L'histoire des réseaux de neurones

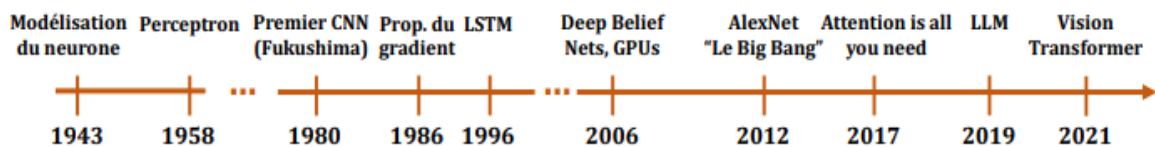


Figure 5
Chronologie des avancées majeures en réseaux neuronaux [20].

Les réseaux de neurones (NN) trouvent leurs origines dans un croisement pluridisciplinaire, mêlant neurosciences, intelligence artificielle, analyse du signal et reconnaissance visuelle.[20]. En 1943, McCulloch et Pitts introduisent un modèle de neurone formel imitant les neurones biologiques, capable de mémoriser des fonctions booléennes simples, conçu pour réaliser des opérations arithmétiques et logiques. La connexion de ces neurones permet la création de réseaux de neurones. En 1949, Donald Hebb propose une règle d'apprentissage, fondée sur le renforcement des connexions neuronales [21]

Frank Rosenblatt présente, en 1957, le perceptron, un modèle simplifié de vision reconnu comme le premier à identifier des formes. En 1960, Widrow et Hoff développent le modèle ADALINE (pour Adaptive Linear Neuron), qui devient la base des réseaux multicouches. Cependant, Minsky

et Papert ont souligné les limites du perceptron, notamment son inaptitude à résoudre des problèmes non linéaires. Parallèlement, T. Kohonen introduit le concept de carte auto-organisée, favorisant l'auto-organisation pour la reconnaissance de formes en classification non supervisée. J. Hopfield explore ces concepts en développant un réseau dynamique atteignant un état de stabilité de manière autonome[21].

L'ère moderne de l'apprentissage profond commence vers 2006, avec l'introduction du concept d'attention en 2017, permettant un modèle unifié pour divers problèmes.[20] Les nouveaux algorithmes d'apprentissage, comme la rétropropagation de l'erreur, émergent dans les années 1980, répondant aux critiques de Minsky et Papert et améliorant l'efficacité des réseaux multicouches. Ces avancées démontrent que les RN peuvent résoudre des problèmes d'optimisation complexes dans divers domaines, illustrant ainsi l'évolution continue de cette discipline.[21]

3 Applications

Les réseaux neuronaux trouvent désormais des applications dans de nombreux secteurs. Ils équipent les avions et voitures autonomes, décryptent chèques et adresses postales, et améliorent le traitement du signal militaire. Ces algorithmes génèrent également de la parole synthétique, développent la vision artificielle, prévoient les tendances boursières et calculent les risques assurantiels[18].

Leur usage s'étend aux chaînes de production, aux diagnostics médicaux, à la prospection énergétique, ainsi qu'aux domaines robotiques et télécoms. Cette technologie transforme déjà nos systèmes et son influence devrait s'amplifier dans les années à venir.[18]

4 Réseaux de Neurones Convolutifs (CNN)

Les CNN sont l'une des formes les plus exceptionnelles de réseaux de neurones artificiels (ANN). Ces CNN jouent un rôle crucial dans l'apprentissage profond, principalement utilisé pour relever les défis complexes liés à la reconnaissance d'images et de formes[22].

Les CNN s'inspirent des champs récepteurs du cortex visuel, qui détectent des motifs spécifiques comme les contours. En imagerie numérique, ce principe est reproduit grâce aux opérations de convolution [23].

Un CNN classique intègre trois couches principales : convolution, pooling et fully connected. En empilant ces couches, le réseau parvient à détecter et reconnaître des motifs. Prenons l'exemple d'un CNN simple appliqué à la classification des chiffres MNIST, illustré figure 21[23]

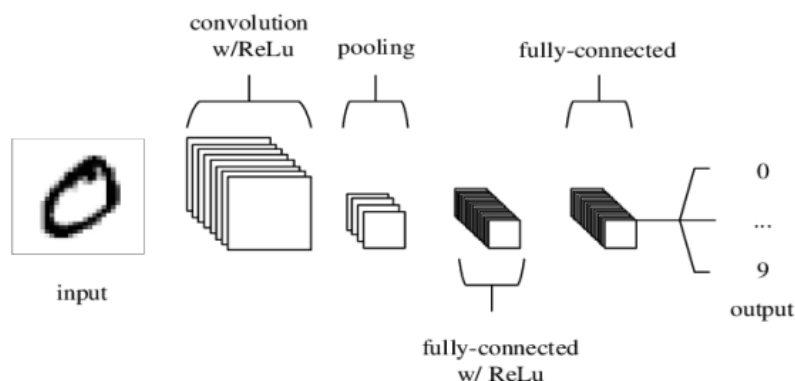


Figure 6
Structure type d'un réseau neuronal convolutif (5 couches) [22].

Les réseaux CNN, bien que très populaires aujourd'hui, trouvent leurs racines dans les années 1980. Rumelhart a introduit en 1986 le premier apprentissage par gradient [24] , perfectionné par LeCun pour la reconnaissance de chiffres [25]. Mais cette méthode présentait des limites, notamment une faible tolérance aux variations de position et de forme. Pour y remédier, les CNN ont émergé, intégrant naturellement l'invariance spatiale grâce à leur structure hiérarchique [26]. Malgré leur potentiel, les contraintes matérielles de l'époque ont conduit à leur abandon au profit des SVM. Leur renaissance récente est directement liée à l'avènement des GPU performants, faisant désormais des CNN la référence incontournable en vision par ordinateur [23].

5 Architecture de CNN

Les CNN s'appuient sur des perceptrons multicouches tout en mimant le cortex visuel. Contrairement aux MLP qui deviennent inefficaces avec les grandes images à cause de l'explosion du nombre de paramètres, les CNN évitent ce problème. Par exemple, une image 32x32x3 nécessiterait 3072 connexions par neurone dans un MLP. Avec une résolution de 200x200, ce nombre atteindrait 120 000 connexions par neurone, rendant le modèle rapidement ingérable[27]. (Figure 23).

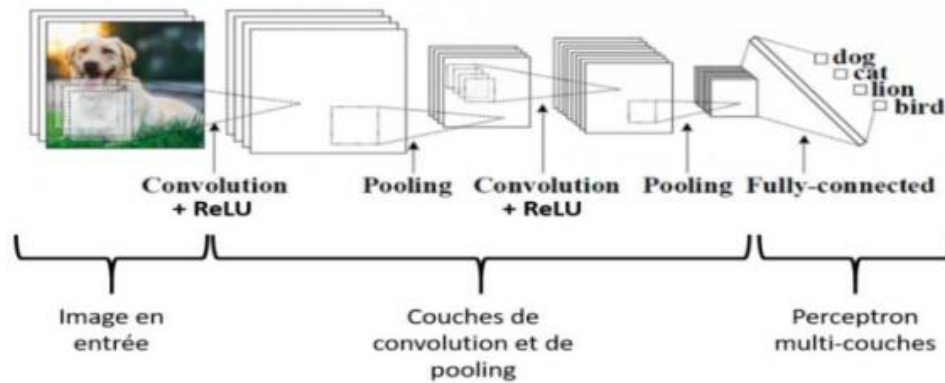


Figure 7
Les Réseaux de Neurones Convolutifs (CNN) [28].

Un réseau CNN se compose de plusieurs couches de calcul superposées [29]

- La couche de convolution (CONV) extrait les caractéristiques via un champ récepteur local.
- La couche de pooling (POOL), réduit la dimension des features par sous-échantillonnage.
- La couche entièrement connectée (FC), équivalente à un réseau de neurones classique
- La couche de perte (LOSS). calcule l'erreur de prédiction

6 Fonctionnement d'un CNN :

Un réseau CNN comporte différentes couches spécialisées. Il traite une image en entrée via sa première couche, puis transmet progressivement les caractéristiques extraites à travers les couches internes. Chaque niveau applique une fonction d'activation sur ses entrées. Le résultat final est comparé à la valeur attendue, suivant le principe de propagation avant [23].

6.1 Couches convolutionnelles :

Au cœur des CNN, la couche convolutionnelle utilise des filtres de taille $n \times n$ pour analyser localement les données d'entrée. Par balayage spatial, ces noyaux génèrent des cartes d'activation après transformation par une fonction non linéaire.[23].

Mathématiquement, la convolution s'exprime par une intégrale mesurant le recouvrement entre une fonction g translatée et une fonction f . Elle se définit par l'expression [22, 23]:

$$(f * g)(t) = \int_{-\infty}^{+\infty} f(\tau)g(t - \tau) d\tau. \quad (04)$$

Où f et g sont les fonctions arbitraires continues dans le domaine t . Dans le cas d'une image, l'opération de convolution est définie comme une fonction discrète. L'opération de convolution discrète entre une image f et une matrice de filtre g est définie comme [22, 23] :

$$f[n] * g[n] = \sum_{m=-\infty}^{+\infty} f[m] g[n - m] \quad (05)$$

La feature map résulte de la convolution entre $f[n] * g[n]$ obtenue en appliquant le filtre g à chaque sous-région de f de taille identique, centrée en (x, y) [23].

Un réseau de neurones utilise des couches de convolution composées de multiples filtres, dont les matrices de poids sont apprises durant l'entraînement. L'empilement alterné de ces couches avec des opérations de pooling forme finalement l'architecture CNN. [22].

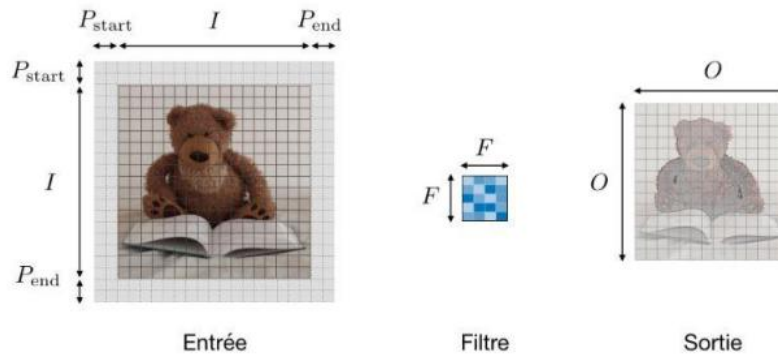


Figure 8
Résultat de l'application d'un filtre convolutif [23].

6.2 Couche de Pooling (POOL) :

Pooling layer (en anglais) permet d'introduire une invariance spatiale et simplifie la gestion des réseaux de traitement d'images en réduisant progressivement les cartes d'activation. Appliquée après une couche convolutive, cette opération diminue la taille des données. Les deux principales variantes sont :

- Max pooling : conserve la valeur maximale d'une zone
- Average pooling : calcule la moyenne d'une région

Cette réduction diminue la complexité calculatoire et le nombre de paramètres [22]. Le max pooling, plus couramment utilisé [30], opère sur des fenêtres glissantes (avec un stride typique de 2 [23]), comme illustré figure 24.

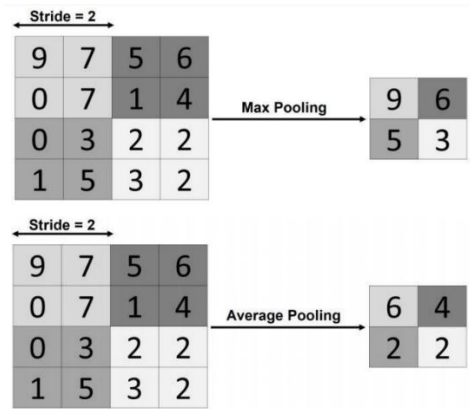


Figure 9
Comparaison des techniques de pooling : maximal vs moyen [23].

6.3 La couche Fully Connected (FC) :

La couche dense (ou FC pour fully connected) traite des données préalablement mises à plat, établissant des connexions complètes entre chaque neurone et l'ensemble des entrées. Positionnée en fin de réseau dans les CNN, elle sert principalement à affiner les résultats finaux comme les probabilités de classification.[23].

6.4 Couche de perte (LOSS) :

La fonction de perte mesure l'écart entre les prédictions du modèle et les valeurs attendues durant l'apprentissage. Elle se trouve généralement en dernière position dans le réseau. On peut y appliquer différentes fonctions de perte en fonction des tâches à accomplir. Par exemple, La fonction Softmax permet de sélectionner une seule classe parmi K options mutuellement exclusives, alors que l'entropie croisée sigmoïde prédit K probabilités indépendantes comprises entre 0 et 1. Quant à la perte euclidienne, elle est dédiée à la régression de valeurs continues[31].

7 Fonctions d'activation

La fonction d'activation ϕ transforme les résultats d'une couche convolutive, limitant les valeurs de sortie des neurones et ajoutant des non-linéarités aux sorties linéaires. Différentes fonctions ϕ sont couramment employées dans les réseaux neuronaux pour des problèmes de classification et de détection[23]

7.1 Unité linéaire rectifiée (ReLU)

La couche ReLU (Rectified Linear Unit) représente une fonction d'activation très répandue dans les réseaux neuronaux profonds. Son calcul s'effectue comme suit [23] :

$$f(x) = \max(x, 0).$$

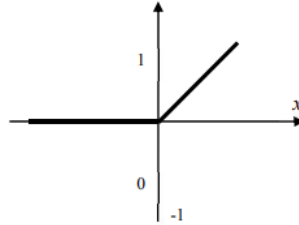


Figure 10
Allure de la fonction ReLU[32].

Les fonctions ReLU sont très prisées pour construire des réseaux non linéaires, car elles facilitent le calcul des gradients pour la rétropropagation. Bien que la fonction ReLU ne soit pas différentiable en zéro, inversement à la fonction sigmoïde, présente des dérivées continues, elle permet généralement une convergence plus rapide que les fonctions tangentes sigmoïdes et hyperboliques[23].

7.2 Unité linéaire rectifiée Leaky (Leaky ReLU)

Contrairement à la ReLU classique, la variante Leaky ReLU introduit une légère pente positive pour les valeurs négatives. Son calcul s'effectue selon la formule [23]:

$$f(x) = \max(x, ax). \quad (06)$$

7.3 La fonction sigmoïde

Avant l'émergence de la fonction ReLU, la sigmoïde dominait comme fonction d'activation. Cette fonction transforme toute valeur d'entrée en un résultat normalisé entre 0 et 1. Son expression mathématique est donnée par[23]:

$$\sigma(x) = \frac{1}{1+e^{-x}} \quad (07)$$

Les fortes valeurs négatives en entrée tendent vers zéro, alors que les valeurs positives importantes s'approchent de 1. Aujourd'hui, la sigmoïde est rarement employée, principalement à cause de ses

inconvenients : saturation du gradient, temps de calcul plus long, et le fait qu'elle ne soit pas centrée sur zéro.[23].

7.4 La fonction tangente hyperbolique

La tangente hyperbolique (tanh), apparentée à la sigmoïde, génère des outputs normalisés sur $[-1,1]$ plutôt que $[0,1]$ [23]. Elle est définie comme suit :

$$\tanh(x) = \frac{2}{1 + e^{-2x}}$$

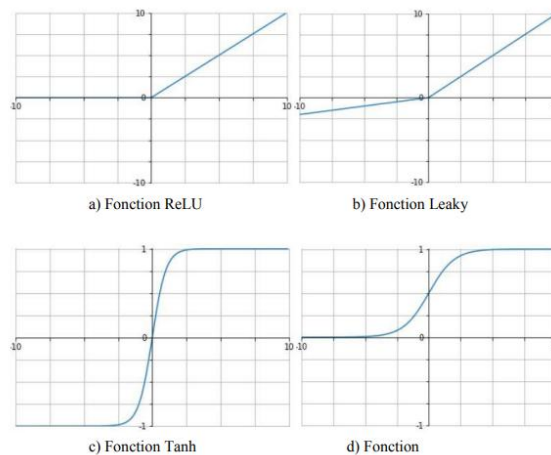


Figure 11
différents fonctions d'activation et leurs graphes[23].

8 Méthodes CNN appliquées à l'analyse d'images et à la localisation d'objets

Les CNN ont révolutionné notre manière d'aborder l'analyse d'images et l'identification d'objets. Leur architecture basée sur la convolution, les CNN peuvent analyser et comprendre les structures visuelles complexes présentes dans les données d'entrée. Ces modèles sont particulièrement réputés pour leurs performances exceptionnelles en classification d'images. Leur puissance, flexibilité et efficacité ouvrent la voie à des applications innovantes dans divers secteurs. Les recherches en cours laissent entrevoir des avancées encore plus prometteuses dans ces technologies.

8.1 Utilisation des CNN pour la reconnaissance d'images

a. Architecture [26] :

Les CNN sont composés de couches convolutives qui détectent les motifs et les caractéristiques locales des images, suivies de couches de pooling qui simplifient les données tout en conservant l'information essentielle

b. Prétraitement:

- Normalisation : Les images sont fréquemment normalisées pour améliorer la convergence.
- Augmentation de Données : L'enrichissement du jeu d'apprentissage s'effectue par transformations numériques (rotation, changement d'échelle, symétrie) pour renforcer la robustesse du modèle [33].

c. Entraînement:

- Les jeux de données tels que CIFAR-10, ImageNet ou MNIST sont couramment utilisés pour entraîner les CNN.
- Pendant l'entraînement, on utilise la fonction de perte, qui est similaire à l'entropie croisée, pour évaluer la performance du modèle.
- L'optimisation implique l'utilisation d'algorithmes tels qu'Adam ou SGD (Stochastic Gradient Descent) pour ajuster les poids du réseau[34].

d. Applications:

- Classification d'Images : Les CNN sont utilisés pour classer des images dans différentes catégories.
- Détection d'Objets : Ils peuvent localiser ainsi identifier différents objets dans une seule image (ex : YOLO, Faster R-CNN).
- Segmentation d'images : Des techniques telles que U-Net utilisent des CNN pour segmenter des images en différentes régions[35].

e. Évaluation :

- Métriques : La précision et le rappel ainsi que la F-mesure sont couramment utilisées pour évaluer la performance des modèles.
- Matrice de Confusion : Sa Permet de visualiser les caractéristiques du modèle sur différentes classes[36].

f.Défis et Améliorations :

- Surapprentissage : L'utilisation de le dropout est essentielle pour éviter le surapprentissage.
- Interprétabilité : Des méthodes telles que les cartes de chaleur (Grad-CAM) aident à comprendre ce que le modèle a appris[37].

8.2 Application des réseaux convolutifs pour l'identification d'objets

Les CNN offrent d'excellentes performances pour la détection d'objets. Alors que les premiers modèles se limitaient à quelques couches, des architectures plus complexes et plus profondes les ont rapidement dépassés. Généralement, ces réseaux se divisent en deux sous-réseaux : une première phase qui identifie des régions d'intérêt, suivie d'une seconde phase qui analyse chaque région proposée[35, 38]. Cette architecture en deux étapes (two-stage detectors) augmente significativement le nombre de couches et de paramètres, ce qui la rend peu adaptée aux applications embarquées disposant de ressources limitées en mémoire ainsi que en puissance de calcul, Pour pallier ces limitations, des réseaux plus compacts, appelés one-stage detectors, ont été développés. Ces derniers permettent de localiser et d'identifier les objets en une seule étape, réduisant ainsi considérablement le temps d'inférence [39, 40].

8.2.1 Modèles à deux étapes (two-stage detectors)

Inspirés par la classification d'images, il est naturel pour les chercheurs d'améliorer une approche en 2 étapes pour la détection d'objets. Dans la plupart des modèles à 2 étapes, la 1^{er} étape base à proposer des régions, suivie d'une classification des images dans ces régions proposées. Cette modification permet d'effectuer un travail supplémentaire sur la localisation, avec peu de changements par rapport aux modèles de classification précédents[41].

Pour la partie proposition de régions, certains systèmes utilisent une technique de fenêtre glissante, comme les Deformable Parts Models (DPM) [42] et OverFeat [43]. Ces méthodes appliquent généralement des classificateurs sur des fenêtres à différents emplacements et échelles. Une autre méthode de proposition de régions est le Selective Search[44] , adopté par le R-CNN[45]. Le R-CNN extrait sélectivement environ 2 000 propositions de régions de manière ascendante[45], ce qui réduit considérablement le nombre de régions nécessaires par rapport aux méthodes de fenêtre glissante[41]En général, les modèles à deux étapes offrent une précision plus élevée. Chaque étape effectuant une tâche spécifique, ces modèles se montrent plus performants pour détecter des objets

de tailles variées. Cependant, face aux exigences de détection en temps réel, les modèles à deux étapes révèlent des lacunes en termes de rapidité de traitement[41].

8.2.2 Modèles à une étape

Grâce aux capacités croissantes des processeurs, les architectures CNN gagnent en profondeur. Cela permet désormais l'utilisation d'approches unifiées comme YOLO (You Only Look Once)[40] ainsi les SSD (Single Shot Multibox Detector)[39], qui offrent une détection plus rapide tout en maintenant une précision similaire, voire supérieure. Les modèles à une étape divisent généralement l'image en grilles de $N \times N$. Chaque case de la grille prend en charge l'objet dont le point central est situé dans ses limites, générant ainsi un tenseur de sortie aux dimensions $N \times N \times S$. Chaque carte de caractéristiques S est une matrice de taille $N \times N$, où chaque élément décrit une caractéristique liée à sa cellule correspondante[41].

Un design typique de ces cartes de caractéristiques pour un objet se compose des $S = (5 + C)$ valeurs suivantes [41]:

- Quatre paramètres définissent le rectangle englobant : position (x,y) et dimensions (largeur, hauteur).
- Une valeur indiquant la probabilité que cette cellule de grille contienne un objet.
- C valeurs précisant à quelle classe appartient cet objet.

Grâce à leur architecture simplifiée, les modèles à une étape nécessitent généralement moins de ressources computationnelles. Certains réseaux récents peuvent atteindre plus de 150 images par seconde (fps) [46]. Cependant, cette rapidité s'accompagne souvent d'une perte de précision particulièrement pour les cibles de faibles dimensions, comme en témoignent les données du tableau 1.2 dans le chapitre 1.5.1. Un autre inconvénient est le déséquilibre de classe lors de l'entraînement. Pour détecter B objets dans une grille, le tenseur de sortie inclut des informations pour $N \times N \times B$ ancres[41]. Toutefois, parmi celles-ci, seules quelques-unes contiennent effectivement des objets, avec un ratio souvent autour de 1000:1[47]. Cela entraîne une faible efficacité lors de l'entraînement.[41]

9 Techniques d'amélioration des CNN

9.1 Data augmentation

L'enrichissement des données d'apprentissage permet d'augmenter les performances des réseaux neuronaux profonds et de minimiser leur sur-adaptation. En élargissant la taille et la diversité des ensembles d'entraînement, Cette approche génère un jeu de données plus riche, réduisant ainsi l'écart entre les données d'entraînement et de validation. Cette méthode génère des variations des données existantes, comme des transformations géométriques et des modifications de couleur, ce qui optimise l'extraction d'informations et améliore la généralisation des modèles[33, 48].

En associant des techniques traditionnelles et des approches neuronales, telles que les GAN, l'augmentation des données peut également renforcer la précision des tâches de classification, même si ces approches nécessitent plus de temps de calcul. De plus, des applications comme le transfert de style peuvent générer des données dans des contextes déséquilibrés, par exemple en produisant des vidéos de conduite nocturne à partir de séquences diurnes, ce qui est crucial pour la sécurité des véhicules autonomes. Si ces méthodes s'avèrent efficaces, elles pourraient simplifier la collecte de données tout en réduisant les efforts nécessaires.[48]

9.2 Dropout

Le « Dropout » [37, 49] est une méthode visant à résoudre le problème de surapprentissage dans les réseaux de neurones, en désactivant aléatoirement certains neurones dans les couches cachées pendant l'apprentissage, avec une probabilité définie par l'utilisateur, souvent de 50 %[37, 50]. Cela prévient la « coadaptation » des activations neuronales, qui peut conduire à un ajustement excessif aux données d'entraînement[34, 37].

Les couches entièrement connectées (FC) occupent une grande part de la mémoire dans un CNN et peuvent également entraîner des problèmes d'overfitting, Ce qui peut ralentir l'analyse des données. Pour y remédier, le dropout désactive aléatoirement certaines unités, améliorant ainsi la réactivité du réseau et optimisant son temps d'entraînement. Après l'apprentissage, les neurones désactivés sont réactivés avec leurs poids initiaux. Plus la couche FC est proche de l'entrée, moins elle subit de désactivation[50].

Cette méthode accélère l'apprentissage tout en désactivant aléatoirement certaines connexions neuronales, elle renforce également la robustesse du réseau et sa capacité à généraliser les concepts

appris [18]. En résumé, Le dropout améliore la robustesse du modèle en maintenant des classifications précises malgré la désactivation aléatoire de neurones [37, 50], tout en optimisant sa performance globale.

9.3 La normalisation par lot (Batch normalization)

La normalisation est une autre technique de régularisation qui normalise l'ensemble des activations dans une couche. Le processus de normalisation consiste à centrer-réduire les activations : on retire leur moyenne et on les divise par leur écart-type au niveau du lot. Cette technique de normalisation, avec la standardisation, est couramment utilisée dans le prétraitement des valeurs des pixels.[33]

9.4 Apprentissage par transfert

Grâce aux progrès des CNN, l'entraînement à partir de zéro est devenu rare, nécessitant d'importantes ressources en données et puissance de calcul. La pratique courante consiste désormais à exploiter des modèles pré-entraînés comme extracteurs de features. Cette méthode, appelée transfer learning, permet d'adapter des connaissances acquises dans un contexte à un nouveau domaine[23].

Le modèle initial est généralement pré-entraîné sur des bases de données standards comme MS COCO. Ce réseau sert ensuite de base pour un transfert d'apprentissage. Lorsqu'on l'utilise comme extracteur de features, on supprime sa dernière couche dense pour la remplacer par deux couches adaptatives. Durant l'entraînement, seuls les paramètres de ces nouvelles couches sont mis à jour, tandis que le reste de l'architecture reste figé[23].

9.5 L'apprentissage par un seul exemple (one-shot) et sans exemple (zero-shot)

Constituent des approches innovantes pour développer des modèles avec des ensembles de données très limités. L'apprentissage par un seul exemple est fréquemment appliqué dans le domaine de la reconnaissance faciale. Une méthode courante dans ce contexte est l'utilisation de réseaux siamois, qui apprennent une fonction de distance, permettant ainsi la classification d'images même lorsque le réseau n'a été entraîné que sur un ou quelques exemples. Une autre technique populaire est l'emploi de réseaux enrichis par mémoire.[33]

D'autre part, l'apprentissage sans exemple représente une approche plus radicale où un réseau utilise des vecteurs d'entrée et de sortie, comme ceux fournis par Word2Vec ou GloVe, pour classer des images en se basant sur des attributs descriptifs.[33]

Dans ce chapitre, nous avons abordé le contexte conceptuel des réseaux de neurones, leur histoire et leurs applications. Par la suite, nous avons exploré les CNN, leur architecture et leur fonctionnement, en mettant l'accent sur leur utilisation pour la reconnaissance d'images (des photos) et la détection d'objets. Enfin, nous avons discuté des techniques d'amélioration des CNN. Ce chapitre a exposé les bases théoriques du deep learning, détaillé les principes clés de la détection d'objets, et présenté des méthodes d'optimisation des réseaux neuronaux convolutifs.

Dans la 4^{ème} chapitre, nous examinerons plus détailler la détection d'objets avec YOLO v10.

CHAPITRE 3 GÉNÉRALITÉ SUR L'ARCHITECTEUR YOLO

I. Introduction de Yolo

La reconnaissance instantanée d'objets constitue un défi clé en vision artificielle, visant à identifier simultanément leur nature et leur localisation dans une scène avec un temps de traitement minimal[51]. Cette technologie est largement utilisée dans des divers applications variées comme la conduite autonome[52], la navigation robotique et le suivi d'objets. Récemment, l'accent a été mis sur les améliorations de détecteurs d'objets basés sur des réseaux CNN pour atteindre des performances en temps réel[51]. Parmi ces modèles, You Only Look Once (YOLO) a gagné en popularité grâce à son équilibre entre performance et efficacité[51].

Le pipeline de détection de YOLO se divise en deux étapes principales : le processus de passage du modèle et le post-traitement par Non-Maximum Suppression (NMS)[51]. Bien que Bien qu'offrant des atouts indéniables, cette approche comporte néanmoins des contraintes en précision et temps de réponse. YOLO se distingue par son architecture simple, sa haute performance et sa rapidité d'inférence[53]. La première version YOLOv1, utilise un réseau de convolution à 24 couches suivi de deux couches entièrement connectées, tandis que sa variante simplifiée, Fast YOLO, ne comporte que 9 couches de convolution, les deux étant basées sur Darknet[53].

Les itérations suivantes, YOLOv2 et YOLOv3, ont apporté des améliorations significatives. YOLOv2 a introduit des techniques telles que la normalisation par lots et des boîtes d'ancrage, permettant une meilleure détection d'objets plus petits grâce à l'utilisation de Darknet-19[40, 54]. YOLOv3 a élargi l'architecture avec Darknet-53 et des réseaux résiduels, tout en remplaçant la fonction SoftMax par l'entropie croisée binaire, offrant ainsi une plus grande flexibilité dans l'étiquetage des boîtes[46].

Les versions suivantes, YOLOv4 et YOLOv5, ont continué cette évolution en améliorant la vitesse et la précision. YOLOv4 a intégré des avancées comme le CSPDarknet53 et un module de cou innovant pour une meilleure intégration des caractéristiques[53]. YOLOv5, construit sur PyTorch, a simplifié le modèle tout en augmentant les performances avec un nouveau backbone CSP-Darknet53 et des techniques comme SPPF et CSP-PAN[53].

Les développements récents, YOLOv6 et YOLOv7, se concentrent sur l'optimisation des ressources computationnelles et l'amélioration de la précision, avec YOLOv7 introduisant des têtes

auxiliaires pour enrichir l'entraînement et les résultats[55, 56]. Enfin, YOLOv8 élargit le cadre des applications en prenant en charge divers domaines tels que la détection, la segmentation et le suivi, tout en intégrant un backbone CSPDarknet53 modifié et un cou PAN-FPN[53]. Les dernières versions, YOLOv9 et YOLOv10, introduisent des techniques avancées telles que l'information de gradient programmable (PGI) et le réseau d'agrégation de couches efficace généralisé (GELAN), avec YOLOv10 éliminant la nécessité de NMS via une conception intégrée de la chaîne de traitement, permettant une détection instantanée des cibles [53, 57].

L'évolution des versions YOLO illustre une quête permanente d'optimisation de rapidité, de justesse et d'adaptabilité, consolidant leur position clé en vision artificielle.

1 Définition de YOLOv10 :

YOLOv10 représente la dernière avancée de la famille de modèles de détection d'objets par YOLO, renommés pour leur capacité à effectuer des détections en temps réel. Ce modèle repousse les limites de l'efficacité et des performances, s'appuyant sur les succès de ses prédécesseurs. Les nouvelles améliorations promises par YOLOv10 pourraient révolutionner la reconnaissance instantanée d'objets dans divers domaines d'application[58].

2 Algorithme YOLO v10 :

2.1 Méthodologie

YOLOv10 se compose de deux composants principaux [59]:

1. Formation et inférence NMS gratuites
2. Conception de modèles axée sur l'efficacité et la précision

Explorons-les tous les deux un par un :

2.1.1 Formation et inférence NMS gratuites

2.1.1.1 Affectations doubles cohérentes pour une formation sans NMS

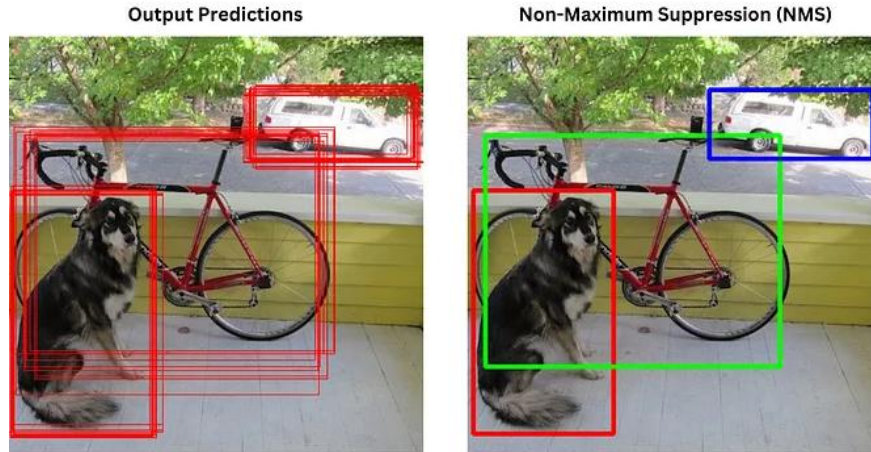


Figure 12
Flux du travail NMS[59].

Dans les modèles YOLO traditionnels, l'entraînement utilise généralement le TAL (Task Alignment Learning) [60] pour allouer plusieurs échantillons positifs à chaque instance, tout en s'appuyant sur la suppression non maximale (NMS) pour éliminer les cadres de délimitation redondants correspondant au même objet en post-traitement[59]. Cependant, le NMS peut parfois se révéler insuffisant, risquant d'éliminer des prédictions utiles ou de ne pas supprimer efficacement tous les doublons, tout en engendrant des coûts de calcul supplémentaires pendant l'entraînement et l'inférence[59]. Pour remédier à ces limitations, le modèle DETR (Detection transformer)[61] propose une méthode élégante, utilisant l'algorithme hongrois pour la correspondance un-à-un, ce qui élimine le besoin de NMS lors de l'inférence. Cette approche met en lumière les défis des techniques traditionnelles tout en ouvrant la voie à des méthodes plus efficaces pour la détection d'objets[59]

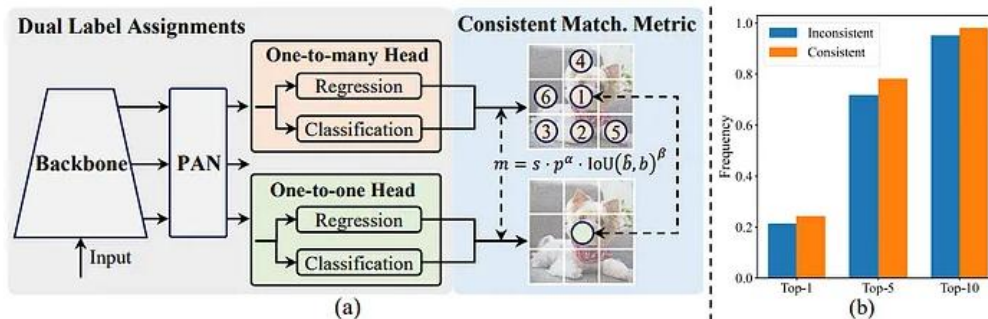


Figure 13
Flux de travail du modèle YOLOv10[51].

Les auteurs de YOLOv10 ont présenté une nouvelle architecture pour aborder le post-traitement NMS à l'aide de l'architecture DETR. Elle se compose de deux composants principaux : une attribution d'étiquette double utilisant deux têtes et une métrique de correspondance cohérente pour faire correspondre les deux prédictions de tête pour une prédiction parfaite[59].

Passons maintenant à la section suivante pour comprendre les deux méthodes en détail

2.1.1.2 Affectations à double étiquette

YOLOv10 combine les avantages des stratégies de correspondance un-à-un et un-à-plusieurs pour l'attribution de labels lors de la formation des modèles de détection d'objets [59]. La méthode un à un simplifie le post-traitement en n'attribuant qu'une seule prédiction à chaque vérité fondamentale, ce qui élimine le besoin de NMS. Cependant, cela peut réduire la supervision, entraînant une précision et une convergence moins optimales. Pour y remédier, YOLOv10 introduit une stratégie d'attribution à double étiquette, utilisant à la fois des têtes un à plusieurs et un à un durant la formation.[59]

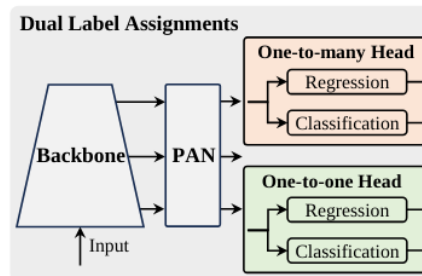


Figure 14
Attribution d'une double étiquette[51].

La correspondance un à un attribue une prédiction unique à chaque instance de vérité fondamentale, éliminant ainsi la nécessité d'utiliser le NMS et permettant un déploiement complet. Toutefois, cette méthode conduit souvent à une supervision réduite, entraînant une précision sous optimale et un ralentissement de la convergence[62].

L'affectation un-à-plusieurs offre des signaux de supervision plus riches, mais nécessite un NMS lors de l'inférence[62].

2.1.1.3 Métrique de correspondance cohérente

Une métrique de correspondance cohérente est employée pour optimiser les attributions de double étiquette dans YOLOv10. Cette métrique assure que les têtes un à un et un à plusieurs s'harmonisent durant le processus d'entraînement[62]. La définition de cette métrique de correspondance est la suivante :

$$m(\alpha, \beta) = s \cdot p^\alpha \cdot \text{IoU}(\hat{b}, b)^\beta \quad (08)$$

Équation 1 Équation métrique[51].

Où (p^α) représente le score de classification, et (b) fait référence aux boîtes englobantes de prédiction et d'instance, tandis que (s) indique si le point d'ancrage de la prédiction se trouve dans l'instance [60, 63, 64]. Les paramètres (α) et (β) harmonisent l'importance entre classification et localisation. Les métriques un-à-plusieurs et un-à-un sont respectivement désignées par $m_{o2m} = m(\alpha_{o2m}, \beta_{o2m})$ et $m_{o2o} = m(\alpha_{o2o}, \beta_{o2o})$. En utilisant la même métrique pour les deux têtes, le modèle s'assure que les meilleurs échantillons sélectionnés par la tête un-à-plusieurs sont également optimaux pour la tête un-à-un[51].

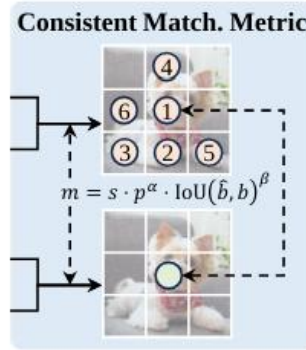


Figure 15
Flux de travail de correspondance des mesures[51].

Dans les assignations doubles de labels, la branche un-à-plusieurs fournit des signaux de supervision plus riches que la branche un-à-un. Pour harmoniser la supervision des deux têtes et optimiser la tête un-à-un vers l'objectif de la tête un-à-plusieurs, nous analysons l'écart de supervision entre elles[51].

Lors de l'examen initial, les deux têtes sont initialisées avec les mêmes valeurs et génèrent les mêmes prédictions, c'est-à-dire que la tête un-à-un et la tête un-à-plusieurs ont le même ppp et IoU. Les cibles de régression des deux branches ne sont pas en conflit, tandis que l'écart de supervision réside dans les cibles de classification[51].

Pour une instance donnée, nous notons le plus grand IoU avec les prédictions comme u^* , et les scores d'appariement un-à-plusieurs et un-à-un comme m_{o2m}^* et m_{o2o}^* [51] respectivement. Si la branche un-à-plusieurs génère des échantillons positifs Ω et que la branche un-à-un sélectionne la i -ème prédiction avec la métrique $m_{o2o,i} = m_{o2o}^*$, nous dérivons les cibles de classification[60, 63-65] :

$$t_{o2m,j} = u^* \cdot \frac{m_{o2o,i}}{m_{o2o}^*} \leq u^* \quad \text{pour } j \in \Omega \quad (9)$$

$$t_{o2o,i} = u^* \cdot \frac{m_{o2o,i}}{m_{o2o}^*} = u^* \quad (10)$$

L'écart de supervision est déterminé par la distance de wassertein de premier ordre :

C'est-à-dire :

$$A = t_{o2o,i} - I(i \in \Omega)t_{o2m,i} + t_{o2m,k} \quad (10)$$

$$k \in \Omega \setminus \{i\}$$

Cet écart diminue lorsque $t_{o2m,i}$ augmente, atteignant un minimum lorsque $t_{o2m,1} = u^*$, c'est-à-dire que i est le meilleur échantillon positif dans Ω . Pour ce faire, nous présentons la métrique de correspondance cohérente, c'est-à-dire : $\alpha_{o2o} = r \cdot \alpha_{o2m}$ et $\beta_{o2o} = r \cdot \beta_{o2m}$, ce qui implique: $m_{o2o} = m_{r \cdot o2m}$. [51] [66]

Ainsi, le meilleur échantillon positif pour la tête un-à-plusieurs est également le meilleur pour la tête un-à-un, permettant une optimisation cohérente. Pour vérifier l'amélioration de l'alignement de la supervision, nous comptons le nombre de paires d'appariement un-à-un parmi les top-1, top-5 et top-10 des résultats un-à-plusieurs après l'entraînement.[64, 65]

2.1.1.4 Modèle axé sur l'efficacité et la précision

En plus du post-traitement, les modèles YOLO rencontrent des défis majeurs pour optimiser l'équilibre entre efficacité computationnelle et précision de détection. Bien que diverses

architectures et stratégies d'optimisation aient été mises en œuvre, une analyse approfondie des composants des modèles YOLO fait défaut [46, 67]. Cette lacune peut entraîner une surcharge de calcul inutile, limitant ainsi les performances globales. YOLOv10 s'engage à optimiser simultanément l'efficacité et la précision dans la détection d'objets. Cela nécessite des optimisations architecturales avancées et des ajustements des algorithmes d'entraînement pour maximiser les performances tout en minimisant les coûts computationnels. Analysons ces éléments techniques en détail[59].

2.1.2 Conception de modèles axée sur l'efficacité

Pour améliorer l'efficacité, YOLOv10 propose plusieurs innovations qui diminuent la charge de calcul tout en préservant les performances : une tête de classification allégée, un sous-échantillonnage spatial des canaux et une conception de blocs basée sur le rang. Analysons chaque composant individuellement[59].

2.1.2.1 Tête de classification légère

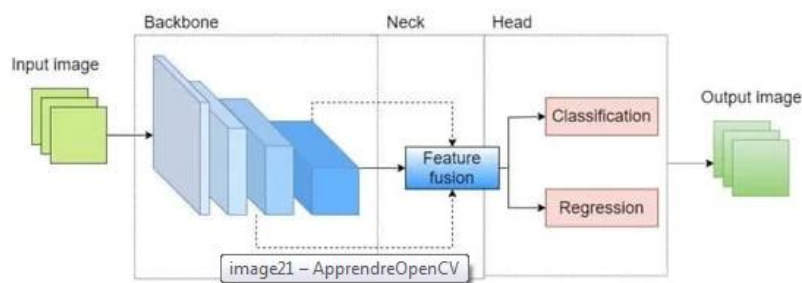


Figure 16
Présentation de l'architecture de YOLO[59].

Dans les modèles YOLO classiques, les têtes de classification et de régression ont une architecture similaire, impliquant une complexité algorithmique accrue, principalement pour l'étape de classification [59].

2.1.2.1.1 Tête de classification : Ce module attribue une catégorie spécifique à chaque cible identifiée ("piéton", "véhicule", etc.), tout en évaluant la vraisemblance de chaque classification. Le système garantit par ailleurs que la somme totale des probabilités reste égale à l'unité[59].

2.1.2.1.2 Tête de régression : cette composante prédit les coordonnées de la boîte englobante pour les objets détectés, y compris les coordonnées centrales, la largeur et la hauteur. De plus, elle fournit un score de confiance pour chaque prédiction, ce qui permet de filtrer les résultats peu fiables.[59]

Après avoir étudié l'impact des erreurs, les chercheurs ont conclu que la tête de régression joue un rôle important dans les performances globales. C'est pourquoi YOLOv10 adopte une tête de classification optimisée, constituée de deux convolutions séparables en profondeur suivies d'une convolution 1×1 , permettant ainsi de diminuer considérablement la charge computationnelle.[67, 68]

2.1.2.2 Sous-échantillonnage découplé des canaux spatiaux

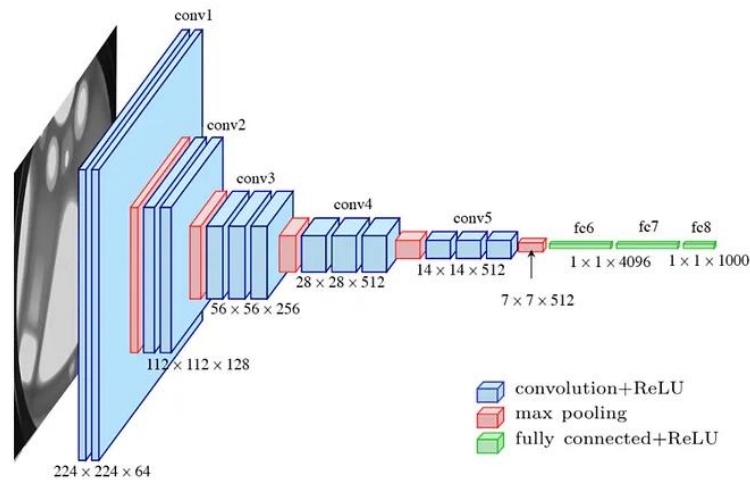


Figure 17
Processus de convolution[59].

Les modèles YOLO classiques appliquent des convolutions 3×3 avec un pas de 2 pour le sous-échantillonnage spatial (passant de $H \times W$ à $H/2 \times W/2$) et pour la transformation des canaux (de C à $2C$), ce qui engendre une charge computationnelle élevée. En revanche, YOLOv10 dissocie ces opérations pour améliorer l'efficacité. D'abord, une convolution ponctuelle ajuste les dimensions des canaux, puis une convolution en profondeur réduit les dimensions spatiales[51].

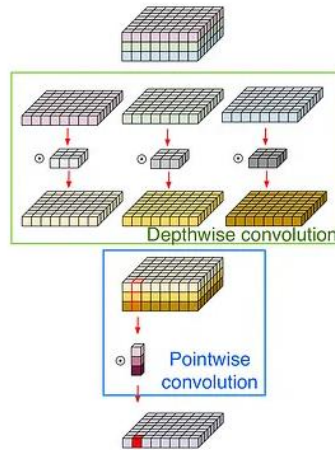


Figure 18
Conversion en profondeur et en point (source-milieu)[59].

En résumé, YOLOv10 recourt à la convolution ponctuelle (filtre 1-1) pour augmenter la quantité de canaux, tandis que la convolution en profondeur est utilisée pour réduire les dimensions spatiales, évitant ainsi de combiner ces deux opérations dans une seule couche. Ce découplage diminue les coûts de calcul tout en permettant de conserver davantage d'informations lors du sous-échantillonnage. Plus précisément, le coût de calcul diminue de $O\left(\frac{9}{2}HWC^2\right)$ à $O\left(2HWC^2 + \frac{9}{2}HWC\right)$, et la réduction du nombre de paramètres de $O(18C^2)$ à $O(2C^2 + 18C)$. Cette approche maximise la rétention des informations et entraîne de meilleures performances avec une latence réduite[51].

2.1.2.3 Conception de blocs guidée par rang

Les modèles YOLO utilisent souvent la même structure de bloc à chaque étape, ce qui peut engendrer des inefficacités et créer des goulots d'étranglement. En plus de YOLOv9 PGI et GELAN, YOLOv10 propose une conception de bloc guidée par le rang pour remédier à la redondance[59]. Il calcule le rang intrinsèque de chaque étape (la dernière convolution du bloc de base final de chaque étape), identifie les étapes redondantes, puis remplace leurs blocs de base par une structure de bloc inversé compact (CIB)[51].

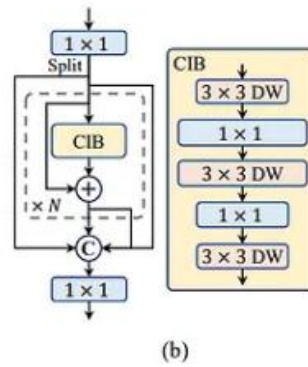


Figure 19
Bloc CIB[51].

Le CIB utilise des convolutions en profondeur pour le mélange spatial et des convolutions ponctuelles pour le mélange des canaux[51]. Ce principe constitue un élément fondamental optimal, semblable à celui intégré dans la structure ELAN (et dans le GELAN de YOLOv9). Cette stratégie adaptative commence par trier toutes les étapes selon leurs rangs intrinsèques dans l'ordre croissant. Ensuite, elle remplace les blocs de base de rang inférieur par le bloc CIB le plus efficace. Les étapes sont ainsi triées par rang, et ces remplacements sont réalisés de manière progressive pour optimiser les performances[51].

2.1.3 Conception de modèle axée sur la précision

Pour améliorer la précision, YOLOv10 introduit deux approches novatrices : la convolution à gros noyau dans le CIB et l'auto-attention partielle. Analysons ces deux méthodes dans la section suivante.

2.1.3.1 Convolution à gros noyau

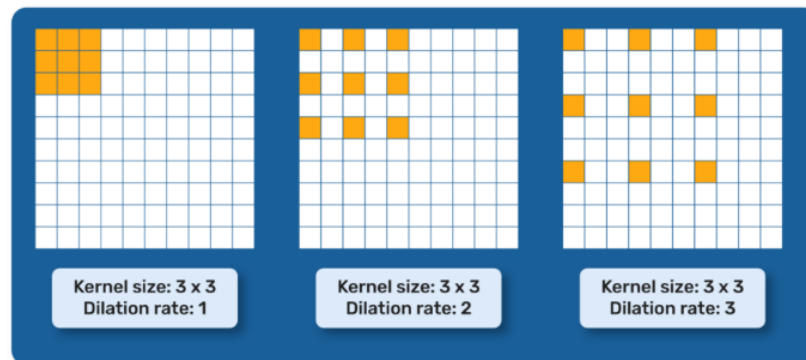


Figure 20
Différentes formes de noyau utilisées dans la convolution des modèles YOLO[59].

Les convolutions en profondeur à gros noyau sont employées pour étendre la taille de la zone de convolution, renforçant ainsi la capacité du modèle à détecter des caractéristiques fines. YOLOv10 intègre ces convolutions à gros noyau dans le CIB aux étapes les plus profondes du modèle, en augmentant spécifiquement le volume du noyau de la deuxième convolution en profondeur 3×3 à 7×7 [52]. De plus, les techniques de ré-paramétrisation structurel [69, 70] ajoutent une branche supplémentaire de convolution en profondeur de 3×3 pour atténuer les problèmes d'optimisation, sans engendrer de surcharge d'inférence. Cette approche améliore les performances de détection, notamment pour les petits objets, tout en maintenant la charge de calcul à un niveau gérable.[51]

2.1.3.2 Auto-attention partielle (APS)

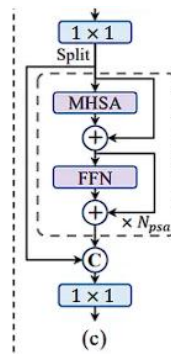


Figure 21
Bloc PSA[51].

L'auto-attention [4], bien qu'efficace, exige une importante puissance de calcul. YOLOv10 introduit un module d'auto-attention partielle (PSA) conçu pour intégrer l'apprentissage de représentations globales tout en minimisant les coûts computationnels[59]. Après une convolution 1×1 , les caractéristiques sont divisées en deux segments ; seule l'une de ces parties est traitée dans les blocs N_PSA , qui se composent d'une auto-attention multi-têtes (MHSA) et d'un réseau de neurones feedforward (FFN). Les deux branches sont ensuite assemblées puis combinées via une convolution 1×1 supplémentaire[51]. L'auto-attention est appliquée uniquement à des étapes ultérieures avec une résolution inférieure, ce qui évite une surcharge excessive due à la complexité quadratique de cette technique. Cette approche renforce à la fois la capacité et les performances globales du modèle, tout en conservant une charge de calcul réduite[59].

3 L'architecture

Les chercheurs de YOLOv10 n'ont pas encore fourni le diagramme d'architecture complet, mais l'architecture s'appuie sur les points forts des précédents modèles YOLO tout en introduisant

plusieurs innovations clés. Nous pouvons avoir un aperçu en comprenant la base de code et en explorant les éléments suivants [59]:

```

1  # Parameters
2  nc: 80 # number of classes
3  scales: # model compound scaling constants, i.e. 'model=yolov8n.yaml' will call
4  # [depth, width, max_channels]
5  b: [0.67, 1.00, 512]
6
7  # YOLOv8.0n backbone
8  backbone:
9  # [from, repeats, module, args]
10 - [-1, 1, Conv, [64, 3, 2]] # 0-P1/2
11 - [-1, 1, Conv, [128, 3, 2]] # 1-P2/4
12 - [-1, 3, C2f, [128, True]]
13 - [-1, 1, Conv, [256, 3, 2]] # 3-P3/8
14 - [-1, 6, C2f, [256, True]]
15 - [-1, 1, SCDow, [128, 3, 2]] # 5-P4/16
16 - [-1, 6, C2f, [128, True]]
17 - [-1, 1, SCDow, [64, 3, 2]] # 7-P5/32
18 - [-1, 3, C2fCIB, [64, True]]
19 - [-1, 1, SPPF, [1024, 5]] # 9
20 - [-1, 1, PSA, [1024]] # 10
21
22 # YOLOv8.0n head
23 head:
24 - [-1, 1, nn.Upsample, [None, 2, "nearest"]]
25 - [[-1, 6], 1, Concat, [1]] # cat backbone P4
26 - [-1, 3, C2fCIB, [128, True]] # 13
27
28 - [-1, 1, nn.Upsample, [None, 2, "nearest"]]
29 - [[-1, 4], 1, Concat, [1]] # cat backbone P3
30 - [-1, 3, C2f, [256]] # 16 (P3/8-small)
31
32 - [-1, 1, Conv, [256, 3, 2]]
33 - [[-1, 13], 1, Concat, [1]] # cat head P4
34 - [-1, 3, C2fCIB, [128, True]] # 19 (P4/16-medium)
35
36 - [-1, 1, SCDow, [128, 3, 2]]
37 - [[-1, 10], 1, Concat, [1]] # cat head P5
38 - [-1, 3, C2fCIB, [1024, True]] # 22 (P5/32-large)
39
40 - [[16, 19, 22], 1, v10Detect, [nc]] # Detect(P3, P4, P5)

```

Figure 22
L'architecteur de YOLO[59].

a) Backbone

Le backbone extrait les caractéristiques des images. YOLOv10 utilise une version améliorée de CSPNet (Cross Stage Partial Network) pour améliorer le flux d'informations et utiliser moins de puissance de calcul[59].

b) Neck

Le cou combine des caractéristiques de différents niveaux de la colonne vertébrale et les transmet à la tête. Il mélange efficacement des caractéristiques de plusieurs échelles (couches) à l'aide de couches PAN (Path Aggregation Network)[59].

c) One-to-Many Head

Lors de l'entraînement, cette tête fait plusieurs prédictions pour chaque objet. Cela donne au modèle beaucoup d'informations (caractéristiques), l'aidant à mieux apprendre[59]

d) One-to-One Head

Lors de l'inférence, cette tête effectue une seule meilleure prédiction pour chaque objet. Cela élimine le besoin de NMS (Suppression Non Maximale), ce qui rend le processus plus rapide[59]

3.1 Variantes du modèle

YOLOv10 est proposé en plusieurs variantes afin de s'adapter aux divers besoins d'application[71]

- YOLOv10-N: version nano pour les environnements à ressources extrêmement limitées.
- YOLOv10-S: Petite version équilibrant vitesse et précision.
- YOLOv10-M: version moyenne pour un usage général.
- YOLOv10-B: version équilibrée avec une largeur accrue pour une plus grande précision.
- YOLOv10-L: Grande version pour une plus grande précision au prix d'une augmentation des ressources informatiques.
- YOLOv10-X: Version extra-large pour une précision et des performances maximales.

Model	Input Size	AP ^{Val}	FLOPS (G)	Latency (ms)
YOLOv10-N	640	38.5	6.7	1.84
YOLOv10-S	640	46.3	21.6	2.49
YOLOv10-M	640	51.1	59.1	4.74
YOLOv10-B	640	52.5	92.0	5.74
YOLOv10-L	640	53.2	120.3	7.28
YOLOv10-X	640	54.4	160.4	10.70

Figure 23
Variations du modèle YOLOv10[59].

3.2 Expériences et résultats

Nous avons choisi YOLOv8 [72] comme modèle de référence en raison de son équilibre remarquable entre latence et précision, ainsi que de sa disponibilité en plusieurs tailles. En optimisant l'efficacité et la précision grâce à des attributions doubles cohérentes pour un entraînement sans suppression non maximale (NMS), nous avons développé nos modèles YOLOv10[51]. YOLOv10 présente les mêmes variantes que YOLOv8, à savoir N / S / M / L / X. De plus, nous avons dérivé une nouvelle variante, YOLOv10-B, en augmentant simplement le facteur d'échelle de largeur de YOLOv10-M[51]. Nous avons vérifié le détecteur proposé sur COCO [73] dans le même cadre d'entraînement à partir de zéro [56, 64, 72]. De plus, les latences de tous les modèles ont été testées sur un GPU T4 avec TensorRT FP16 conformément à,[74]

3.3 Comparaison des performances de YOLOv10

Comme le montre le Tab 2, notre YOLOv10 offre des performances de pointe et une latence de bout en bout sur diverses tailles de modèle. En comparaison avec notre modèle de référence, YOLOv8, YOLOv10 affiche des améliorations de 1,2 % / 1,4 % / 0,5 % / 0,3 % / 0,5 % en précision

moyenne (AP), avec une réduction du nombre de paramètres 28 % / 36 % / 41 % / 44 % / 57 %. De plus, il diminue les calculs de 23 % / 24 % / 25 % / 27 % / 38 % et les latences de 70 % / 65 % / 50 % / 41 % / 37 % pour les variantes N / S / M / L / X, respectivement[51].

Tableau 1

Comparaison des détecteurs d'objets en temps réel sur MS COCO : compromis entre latence, précision et taille des modèles[51].

Model	#Param.(M)	FLOPs(G)	AP ^{val} (%)	Latency(ms)	Latency ^f (ms)
YOLOv6-3.0-N [27]	4.7	11.4	37.0	2.69	1.76
Gold-YOLO-N [54]	5.6	12.1	39.6	2.92	1.82
YOLOv8-N [20]	3.2	8.7	37.3	6.16	1.77
YOLOv10-N (Ours)	2.3	6.7	38.5 / 39.5[†]	1.84	1.79
YOLOv6-3.0-S [27]	18.5	45.3	44.3	3.42	2.35
Gold-YOLO-S [54]	21.5	46.0	45.4	3.82	2.73
YOLO-MS-XS [7]	4.5	17.4	43.4	8.23	2.80
YOLO-MS-S [7]	8.1	31.2	46.2	10.12	4.83
YOLOv8-S [20]	11.2	28.6	44.9	7.07	2.33
YOLOv9-S [59]	7.1	26.4	46.7	-	-
RT-DETR-R18 [71]	20.0	60.0	46.5	4.58	4.49
YOLOv10-S (Ours)	7.2	21.6	46.3 / 46.8[†]	2.49	2.39
YOLOv6-3.0-M [27]	34.9	85.8	49.1	5.63	4.56
Gold-YOLO-M [54]	41.3	87.5	49.8	6.38	5.45
YOLO-MS [7]	22.2	80.2	51.0	12.41	7.30
YOLOv8-M [20]	25.9	78.9	50.6	9.50	5.09
YOLOv9-M [59]	20.0	76.3	51.1	-	-
RT-DETR-R34 [71]	31.0	92.0	48.9	6.32	6.21
RT-DETR-R50m [71]	36.0	100.0	51.3	6.90	6.84
YOLOv10-M (Ours)	15.4	59.1	51.1 / 51.3[†]	4.74	4.63
YOLOv6-3.0-L [27]	59.6	150.7	51.8	9.02	7.90
Gold-YOLO-L [54]	75.1	151.7	51.8	10.65	9.78
YOLOv9-C [59]	25.3	102.1	52.5	10.57	6.13
YOLOv10-B (Ours)	19.1	92.0	52.5 / 52.7[†]	5.74	5.67
YOLOv8-L [20]	43.7	165.2	52.9	12.39	8.06
RT-DETR-R50 [71]	42.0	136.0	53.1	9.20	9.07
YOLOv10-L (Ours)	24.4	120.3	53.2 / 53.4[†]	7.28	7.21
YOLOv8-X [20]	68.2	257.8	53.9	16.86	12.83
RT-DETR-R101 [71]	76.0	259.0	54.3	13.71	13.58
YOLOv10-X (Ours)	29.5	160.4	54.4 / 54.4[†]	10.70	10.60

Performances comparées des détecteurs temps réel sur MS COCO : compromis précision-latence (gauche) et précision-taille (droite), mesurées à l'aide des modèles officiels pré-entraînés pour la latence de bout en bout [51].

De plus, YOLOv10 présente des compromis supérieurs entre précision et coût de calcul par rapport aux autres modèles YOLO. Pour les modèles légers et de petite taille (YOLOv10-N/S contre YOLOv6-3.0-N/S), YOLOv10 atteint des améliorations de 1,5 AP et 2,0 AP avec respectivement 51 %/61 % de paramètres en moins et 41 %/52 % de calcul en moins. Les modèles de taille moyenne YOLOv10-B/M bénéficient également d'une latence inférieure de 46 %/62 % par rapport à YOLOv9-C/YOLO-MS, avec des performances similaires ou supérieures [51].

Pour les modèles de grande taille, YOLOv10-L surpasse Gold-YOLO-L avec 68 % de paramètres en moins, 32 % de latence en moins et une amélioration de 1,4 % de l'AP. Par rapport à RT-DETR, YOLOv10-S / X atteint des vitesses d'inférence 1,8× et 1,3× plus rapides sous des performances similaires[51].

Nous comparons également YOLOv10 avec d'autres modèles YOLO en utilisant l'approche d'entraînement original one-to-many [51]. Dans ce contexte, nous prenons en compte les performances et la latence du processus de prédiction du modèle (Latence), conformément aux études [56, 75, 76]. Comme indiqué dans le tableau 2, YOLOv10 présente également des performances et une efficacité à la pointe de la technologie sur différentes échelles de modèle, soulignant ainsi l'efficacité de nos conceptions architecturales [51]

4 Mise en œuvre de YOLOv10

La détection instantanément d'objets a vu le jour avec Faster R-CNN, et la minimisation de la latence a toujours été un objectif central. La latence d'un modèle est un facteur déterminant pour ses applications pratiques. Pour les applications nécessitant une grande intégrité, il est essentiel d'atteindre des performances optimales en termes d'efficacité et de précision, ce que propose YOLOv10[58].

Nous examinerons le code de YOLOv10, puis nous étudierons comment il peut transformer des applications dans le monde réel.

4.1 Inférence YOLOv10-HuggingFace

La plupart des modèles YOLO sont facilement implémentables en Python grâce à la bibliothèque Ultralytics. Cette bibliothèque permet d'entraîner et d'affiner les modèles YOLO sur nos propres données ou simplement de réaliser des inférences. Cependant, YOLOv10 n'est pas encore entièrement intégré à cette bibliothèque. Néanmoins, nous pouvons expérimenter avec YOLOv10 en utilisant son code via le notebook Colab disponible ou les espaces Hugging Face[58].

Commençons par explorer l'espace Hugging Face ;

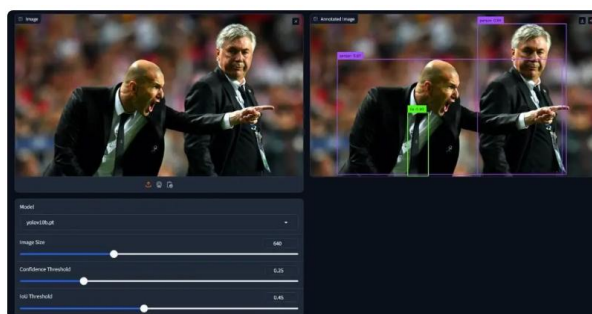


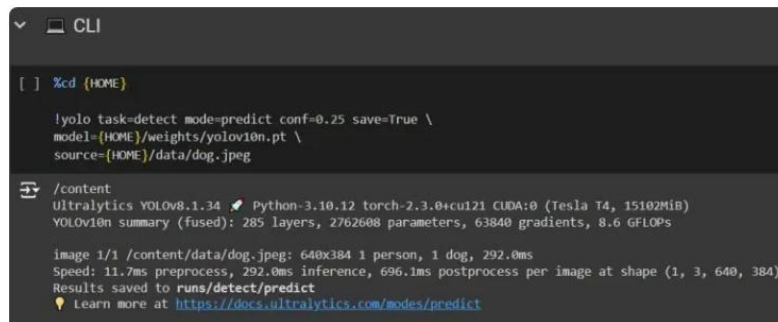
Figure 24
YOLOv10 Hugging Espace Visage[58].

En utilisant l'un des exemples disponibles, nous pouvons constater à quel point YOLOv10 génère rapidement des prédictions. Nous avons également la possibilité d'ajuster différents paramètres et de tester leurs effets. Dans l'exemple précédent, nous utilisons le modèle de base YOLOv10 avec une taille d'image de 640×640 . De plus, nous avons des seuils de confiance et d'IoU [58].

Bien que le seuil IoU n'apporte pas beaucoup d'avantages lors de l'inférence, son importance a été mise en évidence lors de l'entraînement. En revanche, le seuil de confiance est essentiel pendant l'inférence, surtout pour les images complexes : une valeur plus élevée permet des prédictions plus précises, mais cela peut réduire le nombre total de prédictions, et vice versa[58].

4.2 Inférence - Interface de ligne de commande (CLI)

De plus, nous pouvons explorer le code de YOLOv10 à travers le notebook Colab. Le didacticiel proposé est clair et offre diverses options, telles que l'exécution d'inférences via l'interface de ligne de commande (CLI) ou le SDK Python, ainsi qu'une possibilité d'entraînement sur des données personnalisées[58].



```

CLI

[ ] %cd {HOME}

lyolo task=detect mode=predict conf=0.25 save=True \
model={HOME}/weights/yolov10n.pt \
source={HOME}/data/dog.jpeg

/content
Ultralytics YOLOv8.1.34 Python-3.10.12 torch-2.3.0+cu121 CUDA:0 (Tesla T4, 15102MiB)
YOLOv10n summary (fused): 285 layers, 2762608 parameters, 63840 gradients, 8.6 GFLOPs

image 1/1 /content/data/dog.jpeg: 640x384 1 person, 1 dog, 292.0ms
Speed: 11.7ms preprocess, 292.0ms inference, 696.1ms postprocess per image at shape (1, 3, 640, 384)
Results saved to runs/detect/predict
Learn more at https://docs.ultralytics.com/modes/predict
  
```

Figure 25
Inférence CLI YOLOv10 avec le notebook Colab [58].

Après avoir exécuté tous les blocs de code précédents, il est important de les exécuter tels quels, car ils fournissent la configuration nécessaire pour utiliser YOLOv10. Vous pouvez maintenant essayer l'inférence via la CLI ; le code ci-dessus utilise le modèle yolov10-nano, avec un seuil de confiance fixé à 0,25, et télécharge une image à partir des données fournies par le notebook[58].

Si nous souhaitons réaliser des inférences avec différentes tailles de modèles, sur une image personnalisée ou ajuster le seuil de confiance, nous pouvons simplement procéder comme suit :

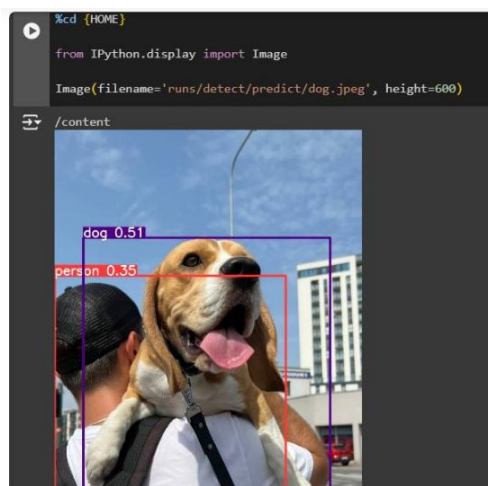


Figure 26
Le résultat de l'inférence CLI[58].

Dans le bloc de code suivant, nous pouvons visualiser les résultats de la prédiction en utilisant la bibliothèque d'affichage Python. La variable « filename » indique l'emplacement où les images résultantes sont enregistrées (notez que nous utilisons l'option Save=True dans la commande CLI)[58].

4.3 Inférence-SDK Python

Le bloc de code suivant montre l'utilisation de YOLOv10 à l'aide du SDK Python :

```
Python SDK

[ ] from ultralytics import YOLOv10

model = YOLOv10(f'{HOME}/weights/yolov10n.pt')
results = model(source=f'{HOME}/data/dog.jpeg', conf=0.25)

image 1/1 /content/data/dog.jpeg: 640x384 1 person, 1 dog, 98.5ms
Speed: 3.1ms preprocess, 98.5ms inference, 149.0ms postprocess per image at shape (1, 3, 640, 384)

[ ] results[0].boxes.xyxy
tensor([[6.7041e+01, 2.5264e+02, 6.4549e+02, 9.6621e+02],
        [7.4646e-01, 3.4156e+02, 5.4170e+02, 1.2773e+03]], device='cuda:0')

[ ] results[0].boxes.conf
tensor([0.5090, 0.3536], device='cuda:0')

[ ] results[0].boxes.cls
tensor([16., 0.], device='cuda:0')
```

Figure 27
Inférence du SDK python YOLOv10[58].

L'inférence via le SDK nous fournit des informations supplémentaires sur les prédictions. Nous pouvons observer les coordonnées des boîtes englobantes, le niveau de confiance, ainsi que « cases.cls », qui représente le numéro de la catégorie (classe) détectée[58].

Ce code est également modifiable, ce qui nous permet d'utiliser la taille du modèle et l'image de votre choix. Le bloc de code suivant montre comment afficher les prédictions en utilisant la bibliothèque « supervision », qui fournira également des informations telles que la vitesse de post-traitement et de prétraitement, la vitesse d'inférence et les noms des catégories[58].

Ainsi, nous avons terminé l'utilisation de YOLOv10 via le code et Hugging Face. Le notebook fourni sur le GitHub officiel de YOLOv10 est très utile, et le tutoriel qui l'accompagne vous guidera tout au long du processus. Cependant, l'entraînement de YOLOv10 nécessite un effort supplémentaire pour créer votre propre ensemble de données et itérer sur le processus d'entraînement[58].

Voyons maintenant comment nous pouvons intégrer ces améliorations de YOLOv10 dans des applications concrètes.

5 Applications concrètes de YOLOv10

L'efficacité, la précision et la légèreté de YOLOv10 en font un outil idéal pour de nombreuses applications, remplaçant potentiellement les modèles YOLO précédents dans la plupart des scénarios de détection en temps réel[58]. Ces nouvelles capacités repoussent les limites de ce qui est réalisable en matière de vision par ordinateur.

- **Suivi d'objets** : L'amélioration de la latence de YOLOv10 le rend particulièrement adapté aux cas d'utilisation nécessitant un suivi d'objets dans des flux vidéo. Les applications varient de l'analyse sportive (suivi des joueurs et des mouvements du ballon) à la surveillance de sécurité (identification des comportements suspects)[58].
- **Conduite autonome** : La détection instantanée d'objets est essentielle pour les autos autonomes. La capacité d'un modèle à détecter et classer les objets sur la route est cruciale pour cette utilisation. La vitesse et la précision de YOLOv10 en font un choix de premier plan pour les systèmes de perception en temps réel dans les voitures autonomes[58].
- **Navigation robotique** : Les robots équipés de YOLOv10 peuvent naviguer dans des environnements complexes en identifiant avec précision les objets et les obstacles sur leur chemin. Cela permet des applications dans la fabrication, les entrepôts et même les tâches ménagères[58].

- **Agriculture** : La détection d'objets peut jouer un rôle important dans la surveillance des cultures et dans la récolte automatisée. La précision et la légèreté de YOLOv10 le rendent particulièrement adapté à ces applications[58].

Bien que ces exemples ne soient qu'un aperçu des nombreuses possibilités offertes par YOLOv10, il ouvre la voie à une nouvelle ère de détection instantané d'objets, qui pourrait bien commencer avec lui.

6 Comparaison des méthodes de détection d'objets (SSD, Faster-RCNN)

La détection d'objets (OD) peut être largement classée en 2 types : les détecteurs à deux étapes ainsi les détecteurs à une seule étape. Les premiers sont définis comme une classe de CNN qui proposent d'abord des régions candidates dans l'image donnée pouvant contenir des objets, puis effectuent la classification et la localisation au sein de ces régions proposées[57]. Parmi les détecteurs à deux étapes les plus connus, on trouve le RCNN (réseau de neurones convolutionnels basé sur régions)[77], Fast R-CNN[35], et Faster R-CNN[38], qui intègrent des mécanismes tels que le ROI =(Region of Interest) pooling et le FPN= (feature pyramid network)[78].

a. Le RCNN introduit en 2014, utilisait une recherche sélective pour proposer des régions candidates potentielles. Après la génération de ces candidates, un réseau CNN était utilisé pour l'extraction des caractéristiques, suivi d'un classificateur SVM pour la classification et la localisation finales. Bien qu'il ait offert des performances satisfaisantes en termes de précision, son efficacité computationnelle était limitée en raison du processus en deux étapes [77].

b. Fast R-CNN a abordé les problèmes d'efficacité de son prédécesseur (RCNN) en introduisant Le ROI pooling. Au lieu de traiter chaque proposition de région individuellement, Fast R-CNN a utilisé le ROI pooling pour extraire des cartes de performances de taille fixe pour chaque région à partir des cartes de caractéristiques d'origine. Cela a entraîné une augmentation remarquable de vitesse calcul, puisque le processus d'extraction des caractéristiques est devenu commun à toutes les propositions de région.[35]

c. Faster R-CNN a encore développé Fast R-CNN en introduisant le Réseau de Proposition de Régions (RPN). Le RPN faisait partie intégrante du réseau, générant des propositions de régions directement à partir des cartes de caractéristiques convolutionnelles, éliminant ainsi la demande

d'une étape secondaire de proposition de régions. En intégrant le RPN dans Fast R-CNN, Faster R-CNN a obtenu des résultats de détection plus rapides et bien précis. Le ROI pooling était également un composant crucial dans Fast R-CNN et Faster R-CNN, permettant une extraction efficace des caractéristiques basées sur les régions et permettant aux réseaux de gérer efficacement des propositions candidates de dimensions et de formes spatiales variées. [38]

d. Le réseau de pyramide de caractéristiques (FPN) améliore les éléments détecteurs à 2 étapes en facilitant la détection d'objets à multiples échelles grâce à l'intégration de cartes de caractéristiques de différentes résolutions. Bien que ces détecteurs offrent une grande précision, leur exigence computationnelle élevée limite leur utilisation. En revanche, les détecteurs à une seule étape, comme le SSD et les variantes YOLO, permettent de prédire directement les boîtes englobantes et les classes en un seul passage, ce qui les rend plus efficaces sur le plan computationnel[78].

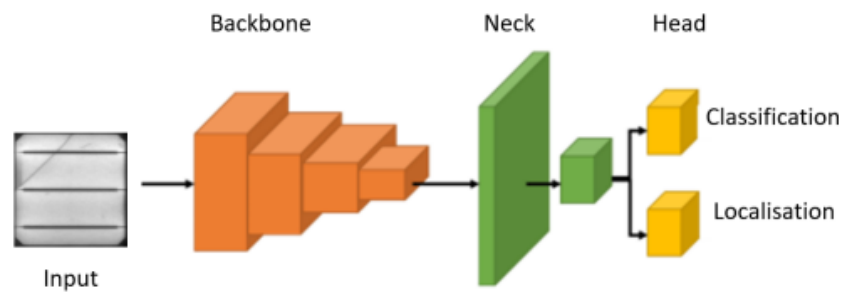


Figure 28
Une architecture standard des détecteurs d'objets à une seule étape[57].

e. Le SSD utilise diverses cartes de performances convolutionnelles à différentes échelles pour prédire les boîtes englobantes et les scores de probabilité de classe. En intégrant des boîtes d'ancrage par défaut à divers rapports d'aspect et échelles, le SSD parvient à détecter précisément des objets de différents tailles et de formes variées en une seule passe à travers le réseau. [39]

f. RefineDet++ est une extension de l'architecture RefineDet d'origine, conçue pour affiner les propositions de cibles de manière itérative à travers plusieurs étapes. RefineDet++ améliore encore la précision en déployant des mécanismes de fusion de caractéristiques améliorés et en affinant les contours des cibles pour une meilleure localisation.[34]

g. Le DSSD intègre des couches de déconvolution pour récupérer les informations spatiales perdues lors de la réduction de caractéristiques. Cela contribue à maintenir la résolution spatiale des cartes de caractéristiques et permet au DSSD de capturer des détails fins, essentiels pour une localisation précise des objets.[57]

h. RetinaNet Cette approche résout le déséquilibre de classes via la Focal Loss, un mécanisme qui accentue le poids des exemples difficiles et mal classés. Cette pondération améliore la gestion des classes minoritaires tout en optimisant les performances globales de détection [35].

Ce chapitre a abordé les principes essentiels de YOLO v10. Nous avons commencé par définir ce modèle et explorer son algorithme, ainsi que son architecture unique qui permet une détection rapide et précise des objets. Ensuite, nous avons étudié de la mise en œuvre de YOLO v10, mettant en lumière ses étapes et considérations pratiques.

Nous avons également examiné les applications concrètes de YOLO v10, illustrant son utilité dans divers domaines, tels que la surveillance, la conduite autonome. Enfin, Nous avons comparé les approches de détection d'objets existantes, soulignant les avantages et inconvénients de chacune, Cette analyse éclaire l'approche distinctive de YOLO v10 comparé aux autres méthodes.

Dans l'ensemble, YOLO v10 allie vitesse et précision de manière optimale, se positionnant comme une référence pour les tâches de détection instantané d'objets

Dans la dernière partie, nous avons abordé le filtre de Kalman

CHAPITRE 4 GÉNÉRALITÉ SUR LE FILTRE DE KALMAN

I. Introduction au Filtre de Kalman

Conçu par Rudolf Kalman (né à Budapest en 1930) [79], le filtre éponyme fut initialement développé pour optimiser le suivi de trajectoire lors des missions Apollo dans les années 1960 , [80]. Ce filtre représente une méthode d'estimation optimale pour systèmes dynamiques soumis à des perturbations stochastiques. Les approches d'estimation varient selon deux critères : le niveau d'information accessible et le moment d'analyse.[80]. En considérant un système dynamique avec des mesures $Y(I_0, I_f)$ entre un instant initial I_0 et un instant final I_f , nous pouvons estimer le cas et l'état du système X à un moment donné r noté $X(r | Y(I_0, I_f))$. Selon la valeur de r [79]:

1. Si $r < I_f$, Ceci relève d'une problématique de régularisation des données (smoothing).
2. Si $r = I_f$, nous faisons face à un problème lié au filtrage (filtering).
3. Si $r > I_f$, c'est un problème lié au prédiction (prediction).

Chaque observation Y_k est reliée à l'état inconnu X_k par la relation $Y_k = H_k X_k + V_k$, où H_k est la matrice de mesure et V_k modélise l'erreur d'observation. L'objectif est d'obtenir le maximum d'information sur l'état du système, qu'il s'agisse de lissage, de filtrage ou de prédiction.[79] Le filtre de Kalman propose un algorithme récursif optimal pour l'estimation en temps réel d'états dynamiques à partir de mesures bruitées, offrant des solutions linéaires sans biais et à variance minimale. Depuis son introduction, cette méthode a connu un essor considérable en robotique, navigation, économie et ingénierie, avec un succès notable dans le programme Apollo. Notre étude se concentrera sur l'application de ce filtre pour résoudre des problèmes de estimation[79].

1 Concepts des bases

1.1 Systèmes dynamiques

Un système dynamique est défini par un espace de phase, qui représente l'ensemble des états possibles d'un système, accompagné d'une équation de développement décrivant la variation de l'état au fil du temps. Cette équation d'évolution peut être formulée comme une équation différentielle ou bien une équation aux dérivées partielles quand le temps est traité comme une variable continue, mais elle peut également être exprimée sous forme d'une application discrète. On distingue deux types de systèmes dynamiques [81] :

- Déterministe : Un système est déterministe lorsque son état présent fixe de manière unique tous ses états ultérieurs.
- Aléatoire : leur avenir ne dépend que de phénomènes passés ou présents.

1.2 Un bruit blanc

Un bruit blanc représente un processus aléatoire caractérisé par une densité spectrale uniforme sur toutes les fréquences. Sa fonction d'autocorrélation correspond à une impulsion de Dirac, indiquant une absence totale de corrélation entre ses échantillons, même pour des intervalles temporels infinitésimaux.[82].

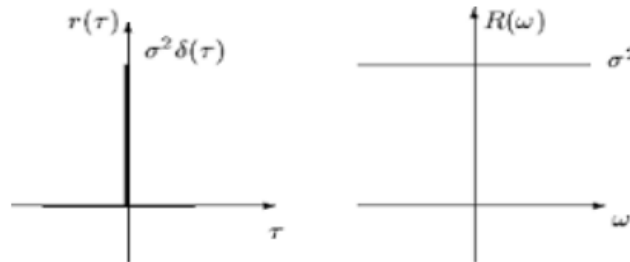


Figure 29

Représentation de l'autocorrélation du bruit blanc : domaine temporel (gauche) et fréquentiel (droite) [82].

Et également, il s'agit d'un processus aléatoire $\{X_n\}_{n \in \mathbb{N}}$ tels que $E(X_n)$ existe et ; [79]

$$\left\{ \begin{array}{l} \text{Var}(X_n) < \infty \text{ pour tout } n \in \mathbb{N} \\ \text{Cov}(X_n, X_i) = 0 \text{ si } n \neq i. \end{array} \right.$$

2 Définition de filtre de kalman

L'estimateur de Kalman est un algorithme récursif permettant de reconstruire l'état d'un système dynamique à partir d'observations partielles ou perturbées par du bruit [83]. Son principe s'appuie sur des formulations mathématiques optimisant la prédiction du système, malgré les incertitudes de mesure et les approximations du modèle [84]. Ce filtre fonctionne comme un reconstituteur d'état en milieu aléatoire. Avec des variances de bruit connues, il opère comme un estimateur optimal minimisant l'erreur quadratique. Les solutions algorithmiques à ce problème furent initialement proposées par Kalman (cas discret) puis étendues par Kalman et Bucy aux systèmes continus [85].

3 Modèle du filtre de kalman

L'algorithme de Kalman s'articule autour de deux relations fondamentales [80] :

- L'équation d'état X_k décrit la dynamique temporelle du système [1] et l'équation de l'observation Y_k qui nous fournit des informations sur l'état [2]

$$\mathbf{S}^{(n,m)} \quad \left\{ \begin{array}{ll} y_{k+1} = A_k y_k + B_k u_k + \Gamma_k \xi_k & 1 \end{array} \right. \quad (11)$$

$$\left\{ \begin{array}{ll} Y_k = C_k y_k + D_k u_k + \eta_k & 2 \end{array} \right. \quad (12)$$

Tel que $k = 0, 1, \dots$. Donc dans ces équations ;

y_k : Représente-le n-vecteur d'état du processus ;

A_k : Représente-la (n, n) matrice déterministe de transition de l'état ;

u_k : Représente-le m-vecteur déterministe de commande ;

B_k : Représente-la (n, m) matrice déterministe qui relie la commande u_k à l'état y_{k+1} ;

ξ_k : Représente-le p-vecteur de bruit blanc, avec une (p, p) matrice de variance covariance Q_k connues ;

Γ_k : Représente-la (n, p) matrice déterministe qui relie le bruit blanc ξ_k à l'état y_{k+1} ;

W_k : Représente-le q-vecteur d'observation ;

C_k : Représente-la (q, n) matrice déterministe de mesure ;

D_k : Représente-la (q, m) matrice déterministe qui relie la commande u_k à l'observation W_k ;

η_k : Représente-le q-vecteur de bruit blanc, qui modélise l'erreur du processus, avec une moyenne et une

(q, q): matrice de variance covariance R_k connues ;

Cette architecture peut se décomposer en deux sous-systèmes linéaires superposés. En considérant [80] ;

$$\left\{ \begin{array}{l} y_{k+1} = z_{k+1} + x_{k+1} \\ w_{k+1} = s_{k+1} + v_{k+1} \end{array} \right. \quad (13)$$

$$\left\{ \begin{array}{l} w_{k+1} = s_{k+1} + v_{k+1} \end{array} \right. \quad (14)$$

Nous avons le droit d'écrire

1. La déterministe d'un système linéaire

$$\left\{ \begin{array}{l} z_{k+1} = A_k z_k + B_k u_k \\ s_k = C_k z_k + D_k u_k \end{array} \right. \quad (15)$$

$$\left\{ \begin{array}{l} s_k = C_k z_k + D_k u_k \end{array} \right. \quad (16)$$

2. La stochastique d'un système linéaire

$$\left\{ \begin{array}{l} x_{k+1} = A_k x_k + \Gamma_k \xi_k \\ u_k = C_k x_k + \eta_k \end{array} \right. \quad (17)$$

$$\left\{ \begin{array}{l} u_k = C_k x_k + \eta_k \end{array} \right. \quad (18)$$

L'objectif consiste à déterminer une estimation optimale récursive des états y_k à partir des mesures w_k . [80]

L'objectif sera de trouver une estimation optimale et récursive des états à partir des observations

4 Algorithme du filtre de Kalman

Le filtre de Kalman représente l'outil d'estimation optimal pour suivre l'évolution d'un système, garantissant une erreur moyenne nulle tout en minimisant l'écart-type des estimations [79, 86].

Le calcul de cet estimateur s'effectue en deux phases après l'initialisation:

- L'étape de prédiction ;
- L'étape de correction.

4.1 Cycle du filtre de kalman

Le filtre de Kalman suit une séquence opérationnelle en trois phases, comme schématisé dans la figure 45 [87].

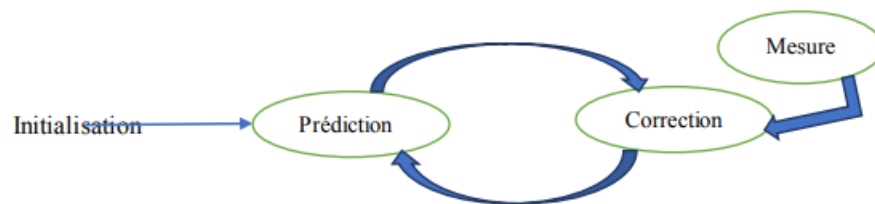


Figure 30
Processus itératif du filtre de Kalman [87].

4.1.1 Initialisation :

Initialement ($k=0$), aucune mesure n'est disponible. La première observation intervient à $k=1$. Dans ce contexte, X_0 (état initial du système) étant une variable aléatoire, son estimation par l'espérance mathématique constitue l'approche la plus rationnelle en l'absence de données. On obtient ainsi [79]:

$$\hat{X}_0 = E(X_0)$$

La matrice de variance-covariance des résidus s'exprime comme :

$$P_0 = E \left[(X_0 - \hat{X}_0)(X_0 - \hat{X}_0)^T \right]. \quad (19)$$

Notons que lorsque l'état initial du système est parfaitement déterminé, nous établissons alors : $\hat{X}_0 = X_0$, et dans ce cas, P_0 est nul [79].

4.1.2 Étape de prédiction

Cette méthode calcule l'état présent à partir de l'état antérieur.

❖ L'équation de prédiction

Au temps discret k , on dispose d'une estimation initiale calculée à partir des données et mesures disponibles jusqu'à l'instant $k-1$. L'objectif est de déterminer un estimateur $\hat{X}_{k|k-1}$ de X_k en s'appuyant uniquement sur l'équation (1) du système $S^{(n,m)}$ sans utiliser la mesure Y_k courante. On obtient alors:

$$\hat{X}_{k|k-1} = A_{k-1}\hat{X}_{k-1|k-1} + B_{k-1}U_{k-1} \quad (20)$$

L'écart d'estimation est caractérisé par :

$$\hat{X}_{k|k-1} = X_k - \hat{X}_{k|k-1}, \quad (21)$$

Sa matrice de covariance s'exprime par:

$$P_{k|k-1} = E \left[(X_k - \hat{X}_{k|k-1})(X_k - \hat{X}_{k|k-1})^T \right] = A_{k-1}P_{k-1|k-1}A_{k-1}^T + Q_{k-1} \quad (22)$$

4.1.3 Étape de correction (Prise en compte de la mesure)

Nous utilisons maintenant la mesure Y_k pour corriger l'estimation prédictive $\hat{X}_{k|k-1}$ et calculer l'estimation mise à jour $\hat{X}_{k|k}$ de X_k . [79]

❖ L'équation de correction [79]

$$\hat{X}_{k|k} = \hat{X}_{k|k-1} + E^*(\check{X}_{k|k-1} | \check{Y}_{k|k-1}) \quad (23)$$

$$= \hat{X}_{k|k-1} + P_{\check{X}_{k|k-1}, \check{Y}_{k|k-1}} P_{\check{Y}_{k|k-1}, \check{Y}_{k|k-1}}^{-1} (Y_k - \hat{Y}_{k|k-1}) \quad (24)$$

$$= \hat{X}_{k|k-1} + K_k (Y_k - \hat{Y}_{k|k-1}) \quad (25)$$

Où $K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + R_k)^{-1}$ représente l'amplification (le gain) apportée par le filtre de Kalman. Mais, puisque les observations $Y_0, \dots, Y_{k-1}$ dépendent linéairement que de X_0 et $\{V_j\}_{j=0}^{k-1}$, nous avons alors :

$$\hat{Y}_{k|k-1} = E^*(Y_k | Y_0, \dots, Y_{k-1}) = E^*(H_k X_k + V_k | \widehat{Y_0, \dots, Y_{k-1}}) = H_k \hat{X}_{k|k-1}, \quad (26)$$

Ce qui implique que :

$$\hat{X}_{k|k} = \hat{X}_{k|k-1} + K_k (Y_k - H_k \hat{X}_{k|k-1}). \quad (27)$$

L'écart de cette estimation peut ainsi être caractérisé par:

$$\tilde{X}_{k|k} = X_k - \hat{X}_{k|k} \quad (28)$$

De même pour la matrice de variance-covariance des écarts :

$$P_{k|k} = E \left[(X_k - \hat{X}_{k|k})(X_k - \hat{X}_{k|k})^T \right]. \quad (29)$$

4.2 Expression des paramètres du filtre de Kalman

$$\hat{X}(j/k) = E[X(j)/Z(1) \dots Z(k)] \quad (30)$$

L'écart d'estimation se caractérise par :

$$\tilde{X}(j/k) = X(j/k) - \hat{X}(j/k) \quad (31)$$

La matrice de variance-covariance de l'erreur conditionnelle correspondante s'exprime comme :

$$P(k/k) = E\{[X(k) - \hat{X}(k/k)][X(k) - \hat{X}(k/k)]' / Z^k\} \quad (32)$$

Le but consiste à déterminer une estimation récursive du vecteur d'état, calculée à partir des relations d'approximation stochastique :

$$\hat{X} = E[X/Z] = \bar{X} + P_{xz} \cdot P_{zz}^{-1} \cdot (Z - \bar{Z}) \quad (33)$$

$$\text{Avec } \bar{X} = \hat{X}(k+1/k) \quad (33.1)$$

$$\bar{Z} = \hat{Z}(k+1/k) \quad (33.2)$$

$$P_{xz} = \text{cov}[X(k+1), Z(k+1)/Z^k] \quad (33.3)$$

$$P_{zz} = \text{cov}[Z(k+1)/Z^k] \quad (33.4)$$

$$P_{xx/z} = E \left[(X - \hat{X})(X - \hat{X})' / Z \right] = P_{xx} - P_{xz} P_{zz}^{-1} P_{zx} \quad (34)$$

$$\text{Avec} \quad P_{zx} = \text{cov}[Z(k+1), X(k+1)/Z^k] \quad (35)$$

$$P_{xx} = \text{cov}[X(k+1)/Z^k] \quad (36)$$

La formulation de la matrice de covariance s'établit comme suit :

$$P(j/k) = E \left[(X(j) - \hat{X}(j/k)) \cdot (X(j) - \hat{X}(j/k))' \right] = E[\tilde{X}(j/k) \cdot \tilde{X}(j/k)']$$

$$\text{Avec} \quad \bar{X} = \hat{X}(k+1/k) \quad (37)$$

$$\text{où} \quad \hat{X}(k) = \hat{X}(k+1/k) + G(k)[Z - \bar{Z}] \quad (38)$$

Le coefficient $G(k)$ caractérise l'amplification du filtre de Kalman. La mise à jour de l'état estimé à l'itération k s'effectue en appliquant une correction à l'estimation précédente, basée sur l'écart entre l'observation mesurée et la prédiction théorique. L'écart d'estimation se formule par[88] :

$$\hat{X}(k/k) = X(k)\hat{X}(k/k) \quad (39)$$

On doit minimiser la quantité $E \left[X(t) \left(\hat{X}(t) \right) \right]$ qui correspond à la matrice de covariance des erreurs d'estimation, définie par :

$$P(k/k) = E \left[(X(k) - \hat{X}(k/k)) \cdot (X(k) - X - \hat{X}(k/k))' \right] \quad (40)$$

Le vecteur d'état à l'instant $k+1$ permet d'établir l'équation d'estimation dynamique par substitution directe.

$$X \rightarrow X(K+1) \quad (41)$$

$$\bar{X} \rightarrow \hat{X}(k+1/k) = E[X(k+1)/Z^k] \quad (42)$$

De même pour le vecteur de mesure :

$$Z \rightarrow Z(k+1) \quad (43)$$

$$Z \rightarrow \hat{Z}(k+1/k) = E[Z(k+1)/k] \quad (44)$$

L'évaluation a posteriori du vecteur d'état au temps $k+1$:

$$\hat{X} \rightarrow \hat{X}(k+1/k+1) \quad (42)$$

En appliquant la relation (37), on obtient la matrice de covariance de la variable estimée au temps $k+1$:

$$P_{xx} \rightarrow P(k+1/k) = \text{cov}[X(k+1)/Z^k] = E[\tilde{X}(k+1/k) \cdot \tilde{X}(k+1/k)' / Z^k] \quad (43)$$

Donc la covariance de la mesure est :

$$P_{zz} \rightarrow S(k+1) = \text{cov}[Z(k+1)/Z^k] = E[\tilde{Z}(k+1) \cdot \tilde{Z}(k+1/k)' / Z^k] \quad (44)$$

La corrélation statistique entre l'état estimé $X(k+1)$ et la mesure $Z(k+1)$:

$$P_{xz} \rightarrow \text{cov}[X(k+1), Z(k+1)/Z^k] = E[\tilde{X}(k+1/k) \cdot \tilde{Z}(k+1/k)' / Z^k] \quad (45)$$

Actualisation de la matrice de covariance :

$$P_{xx/z} \rightarrow P(k+1/k) = \text{cov}[X(k+1)/Z^k] = E[\tilde{X}(k+1/k+1) \cdot \tilde{X}(k+1/k+1)' / Z^{k+1}] \quad (46)$$

5 Les avantages de filtre de kalman

- 1) Le filtre est implémentable sous la forme d'un algorithme pour ordinateurs numériques, qui remplaçait les circuits analogiques pour l'estimation et le contrôle à l'époque où le filtre de Kalman a été introduit. Cette implémentation peut être plus lente, mais elle est capable d'une précision beaucoup plus grande que ce qui avait pu être atteint avec des filtres analogiques [89].
- 2) Le filtre de Kalman ne nécessite pas que la dynamique déterministe ou les processus aléatoires aient des distributions stationnaires.
- 3) Le filtre de Kalman est compatible avec la formulation en espace d'état des contrôleurs optimaux pour les systèmes dynamiques, et Kalman a pu prouver des propriétés duales utiles d'estimation et de contrôle pour ces systèmes.
- 4) Pour l'ingénierie moderne des contrôles, le filtre de Kalman nécessite moins de mathématiques supplémentaires. En conséquence, le filtre de Kalman peut être enseigné au niveau de premier cycle en ingénierie.
- 5) Le filtre de Kalman fournit les informations nécessaires pour des méthodes de décision mathématiquement solides et statistiquement basées pour détecter et rejeter des mesures anormales.
- 6) Ce filtre est couramment employé dans divers champs du traitement du signal, notamment en radar et en analyse d'images.

6. Les limites de filtre de Kalman

a. Modèle Linéaire

Le filtre de Kalman classique s'applique exclusivement aux systèmes linéaires. Pour traiter les cas non linéaires, des versions adaptées comme le filtre de Kalman étendu (EKF) ou le filtre à estimation non parfumée (UKF) sont nécessaires, malgré leur complexité accrue [90].

b. Hypothèse de Bruit Gaussien :

Ce filtre suppose une distribution gaussienne des bruits de mesure et de processus. Si cette condition n'est pas respectée, la qualité de l'estimation peut considérablement se dégrader [91].

c. Estimation de l'État Initial :

La précision de l'estimation est largement influencée par la qualité d'estimation initiale. Une estimation initiale incorrecte peut conduire à une propagation des erreurs au fil du temps, rendant les résultats peu fiables [92].

d. Complexité Computationnelle :

Les opérations matricielles requises par le filtre peuvent s'avérer coûteuses, en particulier pour des systèmes comportant de nombreux états ou lorsque les matrices de covariance sont de grande taille [93].

e. Sensibilité aux Erreurs de Modélisation :

Le filtre s'appuie sur un modèle mathématique précis. Si le modèle n'est pas correctement défini ou si des dynamiques importantes sont négligées, cela peut engendrer des erreurs considérables dans les estimations [94].

f. Stabilité :

La stabilité du filtre peut être affectée si les conditions de bruit et de modélisation ne sont pas respectées, ce qui peut conduire à des estimations divergentes [95].

g. Limitations Pratiques :

Le filtre de Kalman peut éprouver des difficultés face à des données manquantes ou corrompues, et il peut ne pas s'ajuster assez rapidement à des systèmes dynamiques très rapides [96].

7 Extension de filtre de kalman

7.1 Filtre de Kalman Etendu (EKF) :

Considérons cette fois un système dynamique non linéaire, où des bruits blancs gaussiens additifs sont présents. Ces bruits sont indépendants les uns des autres et ne dépendent pas de la loi initiale [97].

$$\begin{cases} X_k = f_k(X_{k-1}) + W_k & (47) \\ Y_k = h_k(X_k) + V_k & (48) \end{cases}$$

Dans ce contexte, les fonctions f_k et h_k sont non linéaires. Contrairement au cas linéaire gaussien, le filtre adapté à ce système ne suit pas une distribution gaussienne et n'admet pas de solution analytique directe. Une approche naturelle consiste à linéariser le système dynamique autour de l'état prédit $X_{k|k-1}$ et de l'état actuel X_k , puis d'appliquer la méthode du filtre de Kalman [97].

Pour l'initialisation on pose:

$$\hat{X}_0 = E[X_0], \text{ et } P_0 = \text{cov}(X_0) \quad (49)$$

Où X_0 n'est pas nécessairement gaussien.

Pour la phase de prédiction, on linéarise l'équation d'état autour de $\hat{X}_{k-1}$:

$$X_k = f_k(X_{k-1}) + W_k \quad (50)$$

$$\cong F_k X_{k-1} + f_k(\hat{X}_{k-1}) - F_k \hat{X}_{k-1} + W_k \quad (51)$$

$$\text{Où } F_k = \nabla f_k(\hat{X}_{k-1}) \quad (51.1)$$

Ainsi

$$\hat{X}_{k|k-1} = E[X_k | Y_{1:k-1}] \quad (52)$$

$$\cong E[f_k(\hat{X}_{k-1}) | Y_{1:k-1}] + F_k E[X_{k-1} - \hat{X}_{k-1} | Y_{1:k-1}] + E[W_k | Y_{1:k-1}] \quad (53)$$

$$= f_k(\hat{X}_{k-1}) \quad (54)$$

La covariance prédite est donnée par :

$$P_{k|k-1} = E[(X_k - \hat{X}_{k|k-1})(X_k - \hat{X}_{k|k-1})^T] \quad (55)$$

$$\cong F_{k-1}^T P_{k-1} F_{k-1} + Q_k. \quad (56)$$

Pour la phase de correction, on linéarise l'équation d'observation autour de l'estimée courant c'est-à-dire $X_{k|k-1}$ on obtient :

$$Y_k = h_k(X_k) + V_k \quad (57)$$

$$\cong H_k X_k + h_k(\hat{X}_{k|k-1}) - H_k \hat{X}_{k|k-1} + V_k \quad (58)$$

Où $H_k = \nabla h_k(\hat{X}_{k|k-1})$. On peut appliquer le filtration par kalman on obtient :

$$K_k = P_{k|k-1} H_k^T [H_k P_{k|k-1} H_k^T + R_k]^{-1} \quad (59)$$

$$\hat{X}_k = \hat{X}_{k|k-1} + K_k [Y_k - h_k(\hat{X}_{k|k-1})] \quad (60)$$

$$P_k = [I - K_k H_k] P_{k|k-1} \quad (61)$$

En résumé, l'algorithme de filtre étendu s'écrit comme suivant :

% Initialisation

$$\hat{X}_0 \leftarrow \bar{X}_0$$

$$P_0 \leftarrow Q_0$$

% Itérations

Pour k=1,2,3....faire

%prédiction

$$F_k \leftarrow \nabla f_k(\hat{X}_{k-1}) \quad (62)$$

$$\hat{X}_{k|k-1} \leftarrow f_k(\hat{X}_{k-1}) \quad (63)$$

$$P_{k|k-1} \leftarrow F_k P_{k-1} F_k^T + Q_k \quad (64)$$

% correction

$$H_k \leftarrow \nabla h_k(\hat{X}_{k|k-1}) \quad (65)$$

$$K_k \leftarrow P_{k|k-1} H_k^T [H_k P_{k|k-1} H_k^T + R_k]^{-1} \quad (66)$$

$$\hat{X}_k \leftarrow \hat{X}_{k|k-1} + K_k [Y_k - (H_k \hat{X}_{k|k-1})] \quad (67)$$

$$P_k \leftarrow [I - K_k H_k] P_{k|k-1} \quad (68)$$

Fin de l'algorithme : Filtre étendu pour un système non linéaire.

L'application du filtre de Kalman à ce système linéarisé conduit aux équations du filtre de Kalman étendu (EKF), basées sur une approximation du premier ordre. La qualité des résultats de l'EKF dépend directement de la précision de cette linéarisation.

En pratique, si les non-linéarités sont trop marquées pour qu'une approximation linéaire reste valable, le filtre de Kalman étendu finit généralement par diverger, car il accorde trop de confiance au modèle linéaire. Pour y remédier, on propose d'augmenter artificiellement la covariance du bruit d'état pour compenser le manque d'adéquation du modèle de prédiction, ou d'augmenter la covariance $P_{k|k-1}$, ce qui réduit l'importance accordée aux états passés.

Bien que l'EKF soit utilisé depuis environ trente ans, des études théoriques pour le justifier n'ont émergé que récemment, notamment grâce aux travaux de Picard. Ces recherches, axées sur le modèle d'état en temps continu et l'observation en temps continu, soutiennent l'utilisation de l'algorithme EKF lorsque les observations sont de haute qualité et que le bruit d'observation ainsi que l'erreur initiale sont faibles. Les résultats ont ensuite été élargis par Getout-Petit à des situations où les observations sont partiellement fiables, par exemple lorsqu'une coordonnée est complètement observée [97].

Cependant, l'EKF ne s'applique pas dans les cas où la densité conditionnelle est multimodale.

7.2 Filtre de Kalman sans parfum (UKF)

Nous évoquons enfin le filtre de Kalman sans parfum (Unscented Kalman Filter, UKF), développé par Julier [98]. Ce filtre repose sur le principe suivant : il approxime la loi conditionnelle a posteriori $P(X_{0:1}|Y_{0:1})$ selon une distribution normale caractérisée par son espérance et sa matrice de variance-covariance, estimées par le filtre et désignées par $X_{k|k}$ et P_k [97]

À l'inverse du filtre de Kalman étendu, cette méthode évite toute linéarisation des équations. Elle repose sur des points sigma représentant parfaitement la distribution gaussienne (moyenne et covariance), auxquels sont directement appliqués les modèles non linéaires [97].

L'algorithme de l'UKF est le suivant[97] :

- **Initialisation** : La 1^{er} étape du filtre distingue à générer un nuage des points répartis dans un volume défini par $\bar{X}_0$ et P_0 , servant d'approximation discrète de la loi $N(\bar{X}_0, P_0)$.

- **Prédiction** : À l'instant $t=k-1$, chaque point du nuage est prédit selon la loi d'évolution non linéaire du système. Les estimations prédites $X_{1:k|0:k-1}$ et $P_{1:k|0:k-1}$ sont calculées à partir de l'échantillon de la loi $P(X_{0:1}|Y_{0:1})$ constitué par les points du nuage prédit. À chaque point est associée une mesure prédite, pour laquelle on calcule également la moyenne $\hat{Y}_{k|k-1}$

- **Mise à jour (correction)** : Les estimations prédites $\hat{X}_{1:k|0:k-1}$ et $P_{1:k|0:k-1}$ sont mises à jour proportionnellement à l'innovation $Y_k - \hat{Y}_{k|k-1}$ et à la matrice de covariance des mesures prédites.

Ce filtre part du principe qu'approcher une distribution gaussienne via un ensemble de points s'avère plus simple que de procéder à une linéarisation fonctionnelle. Les performances de l'UKF seraient donc supérieures à celles de l'EKF, tout en ayant une complexité comparable [98] grâce à une meilleure estimation des moments. Cependant, il montre des limites face à des phénomènes fortement non linéaires et/ou non gaussiens. Les filtres analytiques mentionnés précédemment ne sont pas adaptés lorsque la densité conditionnelle est multimodale [97].

Ce chapitre a présenté le filtre type Kalman, une moyenne essentielle pour l'estimation aussi la prédiction dans les systèmes dynamiques. Nous avons expliqué son modèle, son algorithme en deux étapes (prédiction et correction), ainsi que ses avantages (précision en temps réel) et ses limites (sensibilité aux erreurs de modélisation). Les variantes avancées telles que le filtre de Kalman étendu (EKF) et sa version sans parfum (UKF) ont également été abordées pour traiter les systèmes non linéaires. Ces concepts constituent une base solide pour intégrer le filtre de Kalman dans notre système de surveillance intelligent.

CHAPITRE 5 MATÉRIELLES ET LOGICIELS

1 Servomoteurs pas à pas (Moteur Pas à Pas 28BYJ-48)

1.1 Définition :

Le 28BYJ-48 est un moteur pas à pas unipolaire à 5 fils qui fonctionne sous 5V. Il est parfait pour les projets nécessitant un positionnement précis, comme l'ouverture et la fermeture d'une grille. Puisque le moteur fonctionne sans balais, il offre un mouvement relativement précis et est assez fiable. Malgré sa petite taille, le moteur fournit un couple raisonnable de 34,3 mN.m à une vitesse d'environ 15 RPM. Il offre un bon couple même à l'arrêt et le maintien tant que le moteur reçoit de l'énergie. Le seul inconvénient est qu'il est quelque peu gourmand en énergie et la consommation d'énergie même s'il est à l'arrêt [99].

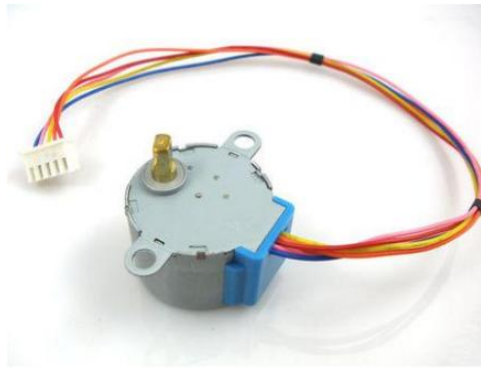


Figure 31
Moteur Pas à Pas 28BYJ-48[100].

1.2 Les caractéristiques Moteur Pas à Pas 28BYJ-48 :

Le moteur 28BYJ-48 se définit par son couple, sa tension d'alimentation, son nombre de phases, son réducteur intégré, son angle par pas, sa plage de fréquences opérationnelles, ses pertes mécaniques et son isolation électrique. Ces éléments influencent la précision, la réactivité, l'efficacité et la sécurité du moteur, le rendant adapté aux applications nécessitant un contrôle précis du mouvement [119].

Les caractéristiques du moteur **28BYJ-48** sont les suivantes :

Tableau 2

Tableau Les caractéristiques Moteur Pas à Pas 28BYJ-48[100].

Modèle	28BYJ-48
Tension nominale	5V DC
Nombre de phases	4
Rapport de variation de vitesse	1/64
Angle de pas	5.625° /64
Fréquence	100Hz
Résistance DC	50Ω±7%(25°C)
Fréquence de traction à vide	>600Hz
Fréquence de traction sortante à vide	>1000Hz
Couple de traction	>34.3mN.m (120Hz)
Couple de positionnement	>34.3mN.m
Couple de friction	600-1200 gf.cm
Couple d'attraction	300 gf.cm
Résistance d'isolation	>10MΩ (500V)
Puissance d'isolation	600VAC/1mA/1s
Classe d'isolation	A
Élévation de température	<40K (120Hz)
Niveau sonore	<35dB(120Hz,No load,10cm)

1.3Fonctionnement du moteur pas à pas 28BYJ-48 :

Ces moteurs pas à pas consomment un courant élevé, il est donc nécessaire d'utiliser un circuit intégré (IC) **driver** comme le **ULN2003**. Pour comprendre comment faire tourner ce moteur, nous devons examiner :

Comme nous pouvons le voir, Le moteur dispose de quatre bobines, dont une extrémité de toutes les bobines est reliée à **+5V** (fil rouge), tandis que les autres extrémités (orange, rose, jaune et bleu) sont sorties sous forme de fils. Le fil **rouge** est toujours alimenté par une source constante de **+5V**, et cette **+5V** sera appliquée (et donc la bobine sera activée) uniquement si l'autre extrémité de la bobine est mise à la masse.[101]

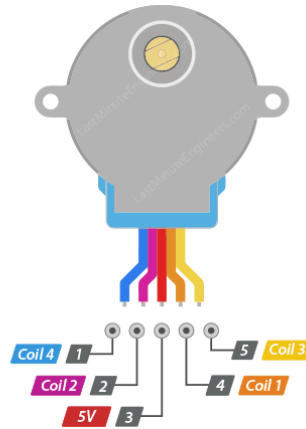


Figure 32
The pinot[99].

Ensemble, une extrémité de la bobine et le centre de prise forment une phase.[99]

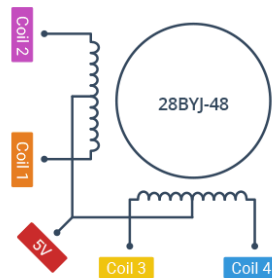


Figure 33
Fonctionnement du moteur pas à pas 28BYJ-48 [99].

Le fonctionnement du moteur pas à pas repose sur l'activation séquentielle des phases selon un ordre prédéfini, appelé cycle de marche.[99]

Cette séquence peut être programmée à l'aide d'un **microcontrôleur** ou par la conception d'un **circuit numérique**. La séquence dans laquelle chaque bobine doit être activée est montrée dans le tableau ci-dessous. Ici, "1" signifie que la bobine est maintenue à +5V, et puisque les deux extrémités de la bobine sont à +5V (fil rouge et l'autre extrémité), la bobine ne sera pas activée. En revanche, "0" signifie que la bobine est mise à la **masse** : une extrémité est à +5V et l'autre est reliée à la masse, ce qui permet d'**activer** la bobine.[101]

Motor Wire Color	Sequence to Rotate in clockwise Direction							
Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Step 7	Step 8	
Orange	0	0	1	1	1	1	1	0
Yellow	1	0	0	0	1	1	1	1
Pink	1	1	1	0	0	0	1	1
Blue	1	1	1	1	1	0	0	0
Red	1	1	1	1	1	1	1	1

1.4 Applications du moteur pas à pas 28BYJ-48

Les moteurs 28BYJ-48, de type pas à pas, équipent fréquemment divers dispositifs comme les lecteurs DVD ou les caméras motorisées. Ce moteur possède une configuration unipolaire à **4 bobines**, chaque bobine étant alimentée en **5V**, ce qui le rend relativement facile à contrôler avec des microcontrôleurs de base. Le moteur a un **angle de pas** de **5,625°/64**, ce qui signifie qu'il effectue **64 pas** pour accomplir une rotation complète, et à chaque pas, il parcourt **5,625°**. Cela permet un contrôle assez précis du mouvement.[101]

Cependant, ce moteur fonctionne uniquement à **5V**, ce qui limite sa capacité à fournir un **couple élevé**. Pour des applications nécessitant un couple plus important, il est recommandé d'envisager des moteurs comme les **Nema17**. En résumé, si vous recherchez un moteur pas à pas **compact, facile à utiliser** et offrant un **couple raisonnable**, le moteur **28BYJ-48** est un excellent choix.[101]

2Driver ULN2003

Parce que le moteur pas à pas **28BYJ-48** consomme une grande quantité d'énergie, ce qui fait il ne pourrait pas être contrôlé directement par un microcontrôleur comme un **Ardin**. Pour contrôler ce moteur, un circuit intégré **driver** comme l'ULN2003 est nécessaire ; c'est pourquoi ce moteur est généralement accompagné d'une carte driver basée sur l'ULN2003.[99]

2.1 Définition ULN2003:

connu pour sa capacité à gérer des courants et des tensions élevés, fournit un gain de courant supérieur à celui d'un transistor unique, permettant ainsi à la faible sortie en tension et courant d'un microcontrôleur de piloter un moteur pas à pas à fort courant.[99]

Le **ULN2003**, est composé d'un ensemble de sept paires de transistors **Darlington**, chacun pouvant piloter une charge allant jusqu'à **500 mA et 50 V**. Cette carte utilise quatre de ces sept paires pour contrôler le moteur pas à pas.[99]

2.2 Les composants de driver ULN2003 :

Le circuit imprimé ULN2003 sert d'interface directe pour commander un moteur pas-à-pas depuis un microcontrôleur. Elle dispose de 4 entrées pour se connecter au microcontrôleur, d'une borne d'alimentation pour la tension du moteur, d'un cavalier ON/OFF, d'un connecteur pour le moteur pas à pas et de 4 LED indiquant l'état des pas..[100]

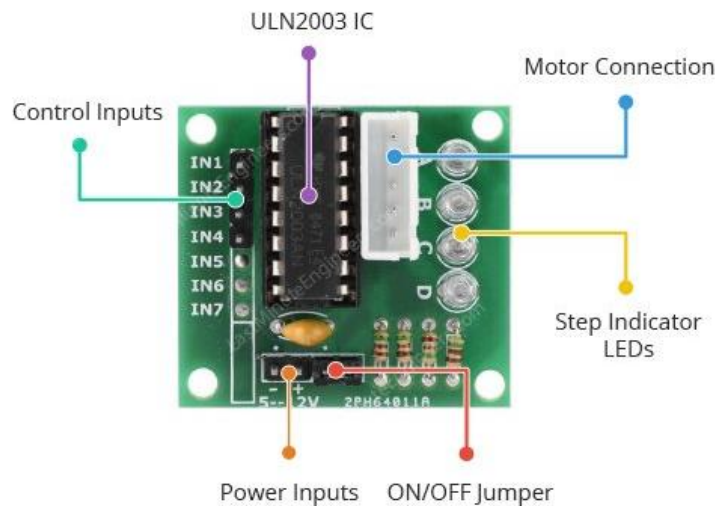


Figure 34
Les composants de driver ULN2003[99]

a) Connexion du moteur pas à pas (motor connection) :

Connectez le moteur pas à pas à ce connecteur. Si vous utilisez l'un de nos moteurs pas à pas, il se branchera directement sur le connecteur. Si vous avez une autre marque sans le connecteur approprié, vous pouvez souder les fils directement à l'arrière de la carte PCB[100].

b- LEDs d'état des étapes (step indicator) :

Indiquent quel canal est actif.

c- Entrée de contrôle (control input) :

Connectez les 4 sorties de votre microcontrôleur aux quatre entrées.

j- Jumper ON/OFF :

Isoler l'alimentation du moteur pas à pas.

e- Entrée d'alimentation (power inputs) :

Alimentation pour faire fonctionner le moteur pas à pas. Assurez-vous qu'elle correspond aux spécifications du moteur pas à pas que vous utilisez[100].

f- Tension d'alimentation : 5-12VCC**g- Courant maximum par sortie : 500mA***2.3 Fonctionnement*

La carte de commande accepte une commande de 4 bits provenant de n'importe quel microcontrôleur et applique en retour les impulsions nécessaires pour faire tourner le moteur. Au cœur du pilote se trouve un circuit intégré ULN2003AN. La carte peut fournir entre 5V et 12V au moteur à partir d'une alimentation indépendante. Elle dispose également d'un ensemble de LEDs qui correspondent aux signaux d'entrée reçus du contrôleur. Ces LEDs offrent une bonne indication visuelle lors des déplacements du moteur.[100]

3 Le Raspberry Pi 4 Model B :

Le Raspberry Pi 4 type B représente la dernière évolution de cette série d'ordinateurs miniatures. Plus performant que son prédécesseur (le Pi 3 B+), il se distingue par son processeur plus rapide, ses capacités multimédias améliorées, sa mémoire accrue et ses options de connexion étendues, tout en conservant une compatibilité ascendante et une efficacité énergétique comparable. Ses performances le rapprochent désormais des ordinateurs x86 d'entrée de gamme[102].



Figure 35
La carte Raspberry Pi 4 type B[103]

3.1 Les composants d'un Raspberry Pi 4 B :

À l'instar de tout système informatique, le Raspberry Pi intègre divers éléments spécialisés qui assurent collectivement son bon fonctionnement[104].

Les composants du Raspberry Pi sont les suivants[104] :

A - Alimentation_USB_Type-C

B - Connecteur_d'affichage_DSI

C- Sans_fil/_Bluetooth

D - Micro-HDMI_0_E_Micro-HDMI_1

F - Système_sur_puce

G- GPIO

H -RAM

I - Port_caméra_CSI

J- Prise_AV_3,5_mm

K- PoE_(Alimentation_par_Ethernet)

L- USB_2.0

M- USB_3.0

N- Port_Etherne



Figure 36
Les composants Raspberry Pi 4 type B[104]

3.2 Les caractéristiques Raspberry Pi 4 Model B

Ce produit se démarque par ses performances et sa polyvalence, intégrant un processeur quad-core 64 bits particulièrement performant. Il permet de piloter simultanément deux écrans en résolution 4K grâce à ses deux ports micro-HDMI, tandis que son décodeur matériel dédié assure un traitement fluide des flux vidéo 4K à 60 images par seconde. La configuration propose jusqu'à 4 Go de mémoire vive pour une réactivité optimale. Côté connectivité, il offre une solution complète avec Wi-Fi double bande (2,4/5 GHz), Bluetooth 5.0, un port Ethernet Gigabit et des interfaces USB 3.0. En option, un module PoE HAT permet en outre de l'alimenter via le réseau Ethernet[102]

Le Wi-Fi dual-band (réseaux) et le Bluetooth bénéficient d'une certification modulaire, simplifiant leur intégration dans les appareils finaux. Cette approche réduit les tests de conformité nécessaires, abaissant les coûts et accélérant la commercialisation [102]

Tableau 3
Les caractéristiques Raspberry Pi 4 Model B [103]

Processeur	Broadcom BCM2711, quadruple âme Cortex-A72 (ARM v8) SoC 64 bits @ 1,5 GHz
Mémoire	1 Go, 2 Go ou 4 Go LPDDR4 (selon le modèle)
Connectivité	2,4 GHz et 5 GHz IEEE 802.11b/g/n/ac sans fil Réseau local, Bluetooth 5.0, BLE Gigabit Ethernet 2 ports USB 3.0 2 ports USB 2.0
GPIO	Embase GPIO 40 broches standard (entièrement rétrocompatible avec les cartes précédentes)
Vidéo et son	2 ports micro HDMI (jusqu'à 4 Kp60 pris en charge) Port d'affichage MIPI DSI 2 voies Port d'appareil photo MIPI CSI 2 voies Port audio stéréo 4 pôles et port vidéo composite
Multimédia	H.265 (décodeur 4 Kp60) H.264 (décodeur 1080p60, encodeur 1080p30); graphiques OpenGL ES 3.0
Support de carte SD	Slot de carte microSD pour le chargement du système d'exploitation et le stockage des données
Puissance d'entrée	5 V c.c. via connecteur USB-C (minimum 3 A*) 5 Vcc. via embase GPIO (minimum 3 A*) Activé pour PoE (Power-over-Ethernet) (requiert un HAT PoE séparé)
Environnement	Température d'utilisation 0 - 50 °C
Durée de vie de production	Le Raspberry Pi 4 modèle B restera en production au moins jusqu'à janvier 2034.

4 Caméra :

Pour concevoir un système de vision artificielle, le module caméra du Raspberry Pi ou la version haute qualité récente constituent des solutions idéales. Ces petits capteurs carrés, équipés d'un objectif, se branchent sur l'interface CSI (Camera_Serial_Interface) du nano-ordinateur. Ils capturent aussi bien des photos haute résolution que des flux vidéo, utilisables directement ou intégrés à divers projets.[104].

4.1 TYPES DE CAMERA

Il existe **trois types de caméras Raspberry Pi [104]**:

a-Camera Module standard : Parfait pour capturer photos et vidéos en pleine lumière. Le choix évident pour un usage quotidien.

b-HQ Camera (High Quality Camera) : Ce modèle offre la meilleure qualité d'image et permet l'utilisation d'objectifs interchangeables. Il est parfait pour des applications où la qualité visuelle est essentielle.

c-Camera Module NoIR : Ce modèle, optimisé pour les captures photo/vidéo en faible luminosité ou nuit noire, utilise directement des éclairages infrarouges sans filtre IR intégré. Il est souvent utilisé pour des projets de vision nocturne, comme les caméras de sécurité ou les caméras de nichoirs. Il nécessite une **source de lumière infrarouge** pour fonctionner correctement dans des conditions sombres.

4.2 Les caractéristiques des caméras Raspberry Pi

4.2.1 Camera Module standard et NoIR [104]:

a-Capteur Sony IMX219 (8 mégapixels) : Ces caméras utilisent un capteur peut capturer des images en 8Mpx, soit 3280 x 2464 pixels.

b-Capture vidéo Full HD : Elles peuvent enregistrer des vidéos en Full HD (1080p) à 30 images par seconde (ips).

c- Modes à haute cadence : Pour des vidéos plus fluides ou en ralenti, elles offrent des options à des cadences plus élevées avec une résolution réduite :

-60 ips en 720p.

-90 ips en 480p (VGA).



Figure 37
Camera Module standard[105]

4.2.2 High Quality Camera (HQ Camera) :

a-Capteur Sony IMX477 (12,3 mégapixels) : La HQ Camera dispose d'un capteur plus grand (12,3 mégapixels) optimisant la luminosité pour des clichés plus détaillés, même de nuit [104].

b-Objectif interchangeable : À la différence des autres composants, la HQ Camera fonctionne sans objectif intégré. Elle nécessite l'ajout d'un objectif compatible monture C/CS, bien que d'autres formats puissent être adaptés à l'aide d'un raccord [104].

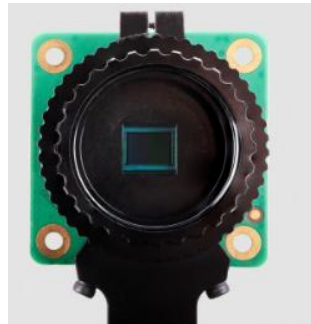


Figure 38
High Quality Camera[105]

4.3Fonctionnements du Camera Raspberry Pi 4

4.3.1 Installation du Camera Module sur Raspberry Pi 4

a-Sécurité : Avant de connecter ou déconnecter le Camera Module ou la HQ Camera, Avant toute manipulation, vérifiez que le Raspberry Pi est hors tension et non connecté au secteur.. Si le Raspberry Pi est sous tension, utilisez le menu "Shutdown" pour l'éteindre et débranchez-le ensuite.[104]

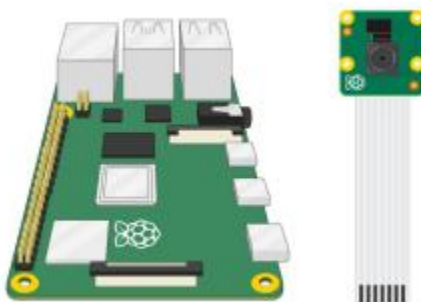


Figure 39
Trouve le port du module caméra[105].

b-Connexion du câble[104] :

Assurez-vous que le câble plat est bien relié au module caméra. Positionnez-le avec la face argentée vers le bas et la partie bleue en haut.

Insérez délicatement le câble dans le connecteur en plastique plat du Camera Module, puis verrouillez-le en repliant le rabat du connecteur.

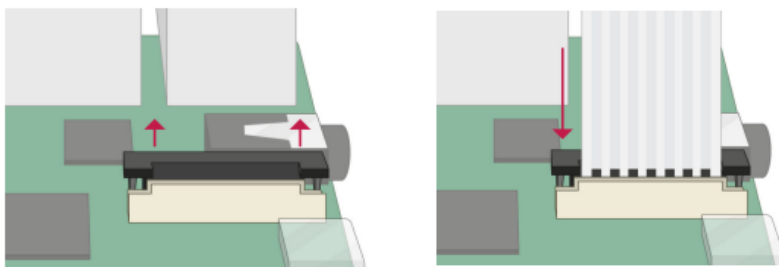


Figure 40
(a) Vérifie que ton Raspberry Pi est éteint, (b) Insère la nappe caméra [105]

c- Connexion à Raspberry Pi[104] :

Repérez le connecteur CSI (Camera Serial Interface) sur le Raspberry Pi et soulevez le volet.

Branchez le câble limande dans le port CSI, en veillant à ce que le côté argenté soit à gauche et le plastique bleu à droite.

Remettez le rabat en place pour maintenir le câble.

d- Retrait du protecteur de l'objectif [104]:

Le module caméra inclut une protection bleue en plastique sur la lentille. Ôtez-la avec précaution.

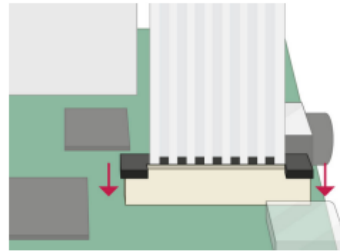


Figure 41
Remets le clip en plastique en place [105].

e-Activation de la caméra[104]:

Allumez la carte Raspberry Pi ainsi chargez **Raspberry Pi OS**.

Ouvrez le menu "Préférences", puis "Configuration du Raspberry Pi".

Allez dans l'onglet "Interfaces", activez l'option **Caméra** et redémarrez le Raspberry Pi.

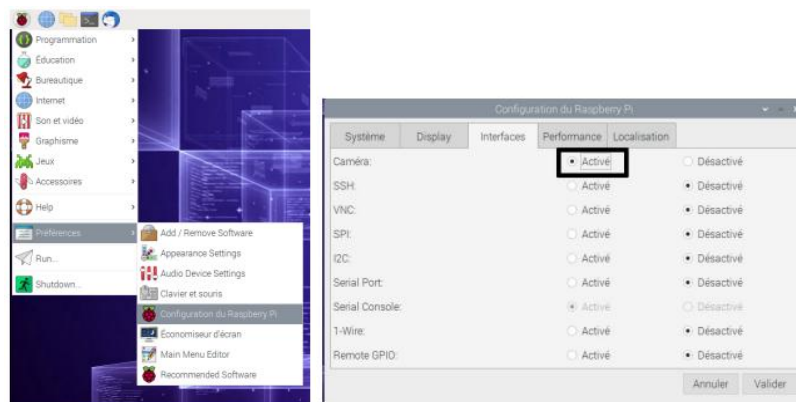


Figure 42
Configuration du Raspberry Pi[105].

4.3.2 Tester la caméra sur Raspberry Pi[104] :

a-Ouvrir le terminal :

Cliquez sur l'icône **Framboise**, puis allez dans **Accessoires** et ouvrez **LXTerminal**



b-Prendre une photo avec la caméra :

Pour capter des photos, vous utilisez une commande simple dans le terminal. Après avoir exécuté cette commande, un aperçu de ce que voit la caméra s'affichera pendant 5 secondes, et une photo sera enregistrée sous le nom **test.jpg**. La photo sera stockée dans le répertoire de l'utilisateur (/home/pi).

c-Changer le nom du fichier :

Pour prendre une autre photo sans écraser la première, vous pouvez modifier le nom du fichier directement dans la commande.

d-Rotation de l'image :

Si l'image est prise dans une mauvaise orientation (par exemple, à l'envers), vous pouvez ajuster l'orientation en utilisant une option de rotation dans la commande. L'orientation de l'image s'adapte à la connexion du câble caméra (dessus, droite ou gauche).

e-Visualiser la photo :

Une fois la photo prise, vous pouvez ouvrir le **Gestionnaire de fichiers** et aller dans le répertoire /home/pi. Là, vous trouverez la photo enregistrée sous le nom **test.jpg**. Double-cliquez pour visualiser le fichier.

4.3.3 Contrôler le module caméra avec du code Python (picamera)

La librairie picamera simplifie la gestion des modules Camera et HQ Camera sur Raspberry Pi en utilisant Python. Elle permet de visualiser, capturer des images et des vidéos, et d'intégrer ces fonctionnalités dans des programmes Python. **[104]**

a-Ouvrir Thonny :

Cliquez sur l'icône **Framboise**, allez dans **Programmation** et ouvrez **Thonny**.

b-Créer un projet Python :

Une fois dans Thonny, Créez un nouveau projet et sauvegardez-le, comme 'Caméra' par exemple.

c-Démarrer un aperçu avec la caméra :

Activez un aperçu en direct de la caméra et paramétrez sa durée, comme 10 secondes. L'image que vous voyez changera en fonction de ce que vous faites devant la caméra.

d-Rotation de l'image :

Si l'aperçu est affiché à l'envers, vous pouvez ajuster l'orientation de l'image en la faisant pivoter selon l'orientation du câble de la caméra (par exemple, pour une caméra montée avec le câble sur le côté droit.

4.3.4 Capture d'images fixes avec la caméra Raspberry Pi

Pour capturer une image fixe avec la caméra Raspberry Pi, vous devez simplement ajuster le programme pour enregistrer une photo à un moment donné, après avoir affiché un aperçu[104].

a-Réduire la durée de l'aperçu :

Diminuez le temps d'affichage de l'aperçu pour qu'il dure seulement 5 secondes, plutôt que 10.

b-Capturer l'image :

Ajoutez la commande pour enregistrer l'image, en indiquant le nom du fichier ainsi que son chemin d'enregistrement. Par exemple, vous pouvez choisir de l'enregistrer sur le bureau sous le nom **image.jpg**.

c-Enregistrer l'image:

Une fois l'aperçu terminé, la caméra capture une photo et l'enregistre dans le dossier spécifié.

4.3.5 Capture de vidéos avec la caméra Raspberry Pi

Pour enregistrer une vidéo avec la caméra de votre Raspberry Pi, voici les étapes à suivre :

a-Démarrer l'aperçu :

Tout comme pour la capture d'image, vous commencez par afficher un aperçu de ce que voit la caméra.

b- Lancer l'enregistrement vidéo :

Vous utilisez une fonction spécifique pour démarrer l'enregistrement de la vidéo. Cela permettra de sauvegarder l'enregistrement dans un fichier, par exemple sur le bureau, avec un nom donné comme **video.h264**.

c- Durée de l'enregistrement :

Spécifiez la durée d'enregistrement souhaitée, comme 10 secondes, durant lesquelles vous pouvez effectuer des mouvements ou interagir avec la caméra pour rendre la vidéo plus dynamique.

d- Arrêter l'enregistrement :

Une fois la durée écoulée, vous arrêtez l'enregistrement, et la vidéo sera sauvegardée automatiquement à l'emplacement choisi.

5 Gps de la carte Raspberry Pi

L'unité de système de positionnement global (GPS) utilise la dernière technologie pour fournir les meilleures informations de position possibles et comprend une antenne GPS active intégrée de 25 x 25 mm avec une prise UART TTL. Une batterie est également incluse afin de permettre une acquisition plus rapide du verrouillage GPS. Ce module GPS offre les meilleures informations de position possibles, permettant ainsi de meilleures performances avec votre Ardupilot ou d'autres plateformes de contrôle de multicopters.[106]



Figure 43
GPS Module[126]

5.1 Les caractéristique de GPS

Un récepteur GPS est un système électronique qui calcule avec précision la localisation d'un objet ou d'un individu en analysant les signaux provenant d'une constellation de satellites. Ses fonctionnalités techniques avancées lui permettent de délivrer des données géospatiales exactes et instantanées[107].

Les caractéristiques d'un module GPS sont les suivantes :

- a.) **Récepteur GPS autonome.**
- b.) **Basé sur le module U-BLOX NEO-6M**, il est compact et offre des performances excellentes.
- c.) **Antenne céramique intégrée et puce LNA à gain élevé MAXIM 20,5 dB**, offrant de solides capacités de recherche de satellites.
- d.) **Paramètres réglables via le port série** et possibilité de les sauvegarder dans la mémoire EEPROM.
- e.) **Interface IPX incluse**, permettant de connecter une variété d'antennes actives, offrant une grande adaptabilité.
- f.) **Compatible avec un niveau de 3,3V**, facile à connecter à n'importe quel microcontrôleur.
- g.) **Batterie rechargeable intégrée** pour stocker les données de recherche des satellites.
- h.) **Interface du module RS232 (niveau TTL, 3,3V pour 'H', 0V pour 'L')**, supporte les débits en bauds suivants : 4800, 9600, 19200, 38400 (par défaut), 57600, 115200, 230400.

Note : Après l'extinction de l'alimentation principale, la batterie de secours peut maintenir les données de recherche des satellites pendant une demi-heure, Réduisant ainsi le temps d'initialisation lors d'un démarrage à chaud [107].

Tableau 4
Les caractéristiques d'un module GPS[107].

Éléments	Description	
Interface	Niveau TTL, 3,3V	
Type de récepteur	Moteur u-blox 6 à 50 canaux, Code C/A GPS L1 SBAS : WAAS, EGNOS, MSAS	
Taux de la navigation	5Hz	
Précision	Position Accuracy	2.5 m CEP
	SBAS	2.0 m CEP
Acquisition	Démarrage à froid	27s
	Démarrage assisté	< 3s
	Démarrage à chaud	1s
Sensibilité	Sensibilité de suivi	-162 dBm
	Sensibilité au démarrage à froid	-147 dBm
	Sensibilité au démarrage à chaud	-156 dBm
Tension de fonctionnement	DC3.3V~5.0V	
Courant de fonctionnement	45mA	
Impédance TXD/RXD	510 ohms	

5.2 BROCHAGE DU MODULE GPS NEO-6M (pinout)[108]

Le module GPS NEO-6M possède quatre broches

Broche VCC : elle doit être connectée au VCC (3,3 V ou 5 V)

Broche GND : elle doit être connectée à GND (0 V)

Broche TX : elle est utilisée pour la communication série et doit être connectée à la broche série RX sur Raspberry Pi.

Broche RX : elle est utilisée pour la communication série et doit être connectée à la broche série TX sur Raspberry Pi.

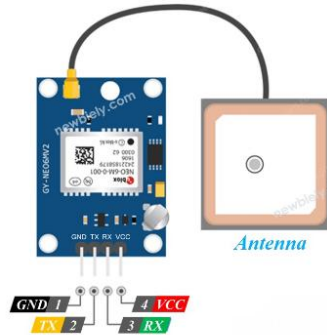


Figure 44
Pinout de GPS [108]

5.3 Connexion du récepteur GPS et du Raspberry Pi[109]

Le module GPS utilise une connexion série simple et doit être connecté aux broches TX et RX du Raspberry Pi (TX à RX et vice versa). Il est également relié aux broches d'alimentation du Raspberry Pi, en veillant à utiliser la tension appropriée.

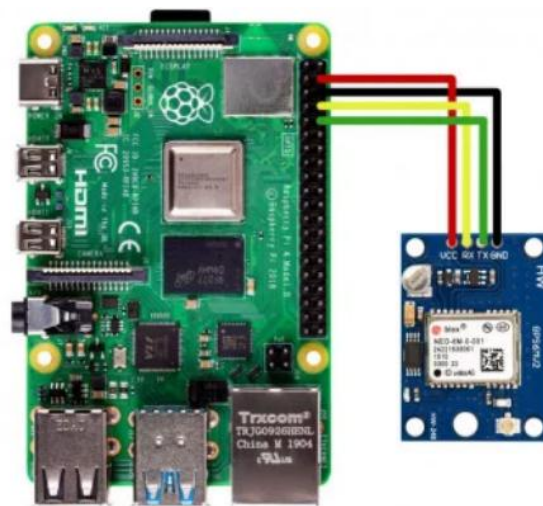


Figure 45
Connexions entre le récepteur GPS et le Pi 4 [109].

5.4 Configuration du Raspberry Pi 4 pour le GPS[109]

a) Exécution de raspi-config :

- Lancez la commande suivante pour configurer les interfaces série :

```
sudo raspi-config
```

- Allez dans « Options d'interface » puis « Série ».
- Désactivez l'accès au login-shell via la connexion série.
- Acceptez l'activation des ports série.
- Retournez au menu principal et redémarrez le Raspberry Pi.

b) Installation des logiciels nécessaires

- Installez les outils pour interagir avec le GPS :

```
sudo apt-get install gpsd gpsd-clients
```

« gpsd » est un démon qui gère la communication avec les récepteurs GPS.

c) Vérification de la réception des données

- Vérifiez que des données sont envoyées par le GPS via le port série :

```
cat /dev/serial0
```

-Si aucune donnée n'apparaît, vérifiez les connexions. Si vous rencontrez des problèmes d'accès, ajoutez l'utilisateur « pi » au groupe « dialout » :

```
sudo adduser pi dialout
```

d) Arrêt du service gpsd

- Le service gpsd peut démarrer automatiquement après installation, mais vous devez l'arrêter pour configurer correctement les ports :

```
sudo systemctl stop gpsd.socket
```

-Vous pouvez également le désactiver pour éviter qu'il ne redémarre à chaque démarrage :

```
sudo systemctl disable gpsd.socket
```

e) Démarrage de gpsd avec le port série correct

- Démarrez gpsd avec la commande suivante pour diriger les données du GPS vers un socket

```
sudo gpsd /dev/serial0 -F /var/run/gpsd.sock
```

f) Affichage des données GPS

- Utilisez l'une des commandes suivantes pour afficher les données GPS :

```
sudo gpsmon  
sudo cgps -s
```

- Vous devriez commencer à voir des données, un temps de convergence initial d'environ 30 minutes est nécessaire pour atteindre la pleine précision.

g) Problèmes de port série

- Si vous rencontrez des erreurs, essayez d'arrêter le service qui accède au port série avec cette commande :

```
sudo systemctl stop serial-getty@serial0.service
```

5.5 Obtenir la position GPS avec le Raspberry Pi

Connectons le module GPS au Raspberry Pi pour extraire les informations GPS. Il est possible d'interfacer le module GPS avec le Raspberry Pi en utilisant Python ou C (WiringPi). Pour ce faire, connectez le module GPS au Raspberry Pi comme indiqué tutoriel « www.electronicwings.com » explique comment connecter un module GPS à un Raspberry Pi. Le guide couvre le branchement matériel et la configuration logicielle nécessaire pour récupérer les données de positionnement.

5.5.1 Utilisation de Python

Nous allons extraire les informations de latitude, de longitude et d'heure à partir de la chaîne NMEA GPGGA envoyée par le module GPS, en utilisant Python. Ces données seront affichées dans la console (terminal). En utilisant les valeurs de latitude et de longitude, vous pourrez localiser la position actuelle sur Google Maps.

5.5.2 Utilisation de C

Nous allons extraire la chaîne NMEA GPGGA et l'afficher dans la fenêtre de sortie. Pour cela, nous utilisons la bibliothèque WiringPi écrite en C afin de lire les données du module GPS.

7 Capteur Ultrasons HC-SR04

Le HC-SR04 est un capteur ultrasonique fonctionnant sur le principe du sonar pour évaluer des distances. Son circuit électronique intégré restitue cette mesure (Fig. 60). Compact et précis, il combine un émetteur et un récepteur d'ultrasons pour des détections stables et non contact. Ses applications couvrent divers domaines : télémétrie, contrôle de mouvement, sécurité, imagerie médicale, tests industriels et systèmes à ultrasons[110].

Ce module ultrasonique détecte des objets de 2 à 500 cm avec une précision de 3 mm. Son angle de détection est de 15° environ. Insensible à la lumière ambiante et aux surfaces sombres, il peut néanmoins rencontrer des difficultés avec certains matériaux absorbants comme les textiles.[111]

a-Le principe de fonctionnement :

1. Générer un signal numérique haut sur l'émetteur pendant 10 μ s
2. Le capteur émet automatiquement 8 impulsions ultrasoniques à 40 kHz et capte les échos
3. Si un écho est détecté, la durée du niveau haut correspond au temps de parcours aller-retour.
Calcul :

$$\text{Distance} = (\text{Durée_écho} \times 340)/2 \text{ (en mètres)}$$

b-Connecteur du capteur :

- Tension requise : 5V
- Port d'entrée pour signal ultrasonique émis
- Port de sortie pour signal ultrasonique capté
- Référence terre (0V)



Figure 46
Capteur Ultrason HC-SR04[111].

7.1 Caractéristiques et spécifications du capteur HC-SR04

Le capteur ultrasonique HC-SR04 présente ces spécifications techniques principales [12] :

Tableau 5

Spécifications techniques du capteur HC-SR04[12].

Distance de captation	2 cm à 5 m
Résolution (précision)	3mm
Tension d'exploitation (Voltage d'entrée)	5 V
Courant (Ampérage d'entrée)	15 mA
Fréquence d'opération	40Hz
Angle de mesure	30 degrés
Angle efficace	15 degrés
Signal d'entrée trigger	10 μ s TTL impulsion
Dimensions L x W x H	45 mm x 20 mm x 15mm
Poids	8.5g

7.2 Broches de connexion

Les broches du capteur sonar à ultrasons HC-SR04 sont les suivantes [112]:

- **Vcc** : pour l'alimentation +5 V DC.
- **Trig** : pour lancer la mesure entrée de déclenchement (Trigger input).
- **Echo** : Sortie de la mesure (Echo output), renvoie le signal de retour de l'ultrason.
- **GND** : la masse (Ground).

7.3 Fonctionnement du capteur Ultrason HC-SR04

Pour activer la mesure, une impulsion d'au moins 10 μ s doit être envoyée sur l'entrée 'Trig'. Le capteur émet alors 8 impulsions ultrasoniques à 40 kHz avant d'attendre l'écho. Dès que le signal réfléchi est capté, la sortie 'Echo' passe à l'état haut avec une durée proportionnelle à la distance détectée[113].

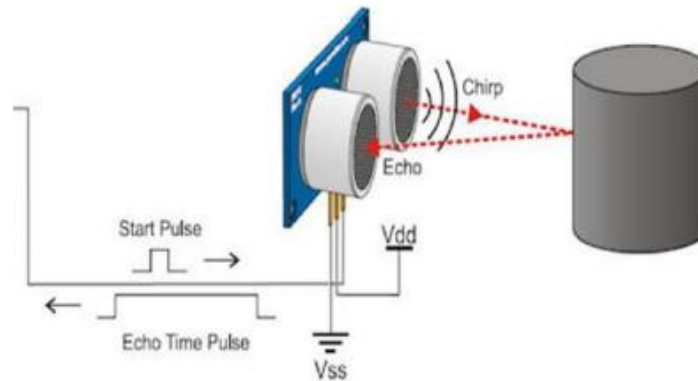


Figure 47
Mécanisme de détection du module HC-SR04[114].

7.3.1 Mesure de la distance de la cible

La propagation acoustique couvre une distance égale au produit de sa célérité (~ 340 m/s ou 34 000 cm par μ s) et de sa durée de trajet[12].

$$d = v \cdot t$$

Variables :

d : distance (m)

v : célérité (vitesse) du son (m/s)

t : durée (s)

Le module HC-SR04 mesure des intervalles en dizaines de μ s. La durée réelle s'obtient en multipliant la lecture par 10 μ s. Comme l'onde acoustique effectue un trajet aller-retour (émetteur $\rightarrow$ cible $\rightarrow$ récepteur), la distance effective correspond à la moitié du résultat[12].

$$d = \left(\frac{34000}{1000000} \cdot 10Valeur \right) \cdot \frac{1}{2} \quad (cm)$$

En résumé :

$$d = \frac{17000}{1000000} \cdot Valeur \quad (cm)$$

En conclusion :

$$d = \frac{17}{100} \cdot \text{Valeur} \quad (cm)$$

$$d = \frac{\text{durée}}{58} \quad (cm) \quad (2.5)$$

La formule (2.5), présente également dans l'utilisation physique du HC-SR04, utilise une fraction approximative de 17/1000, qui est équivalente à 1/58,8235. Bien que cette approximation permette de simplifier les calculs, elle peut donner des résultats légèrement moins précis.[12]

Remarque :

À des distances importantes, il est recommandé que la surface de la cible à détecter soit minimum **0,5 m²** pour garantir une détection fiable.[113]

7.3.2 Le diagramme temporel

Le diagramme temporel du fonctionnement du capteur HC-SR04 est illustré ci-dessous (figure 63). Pour déclencher la salve d'ultrasons, il faut avoir un signal à l'état **8** pendant **10 µs** dans l'entrée de l'émetteur (**Trig**). Une fois ce signal reçu, le module émet **8 cycles automatiquement d'ultrasons (40 kHz)** puis attend la réception du signal réfléchi.[12]

À la réception de l'écho, le système produit une impulsion dont la durée varie selon la distance de la cible. La mesure du délai entre l'émission et la réception permet ainsi de déterminer cette distance[12].

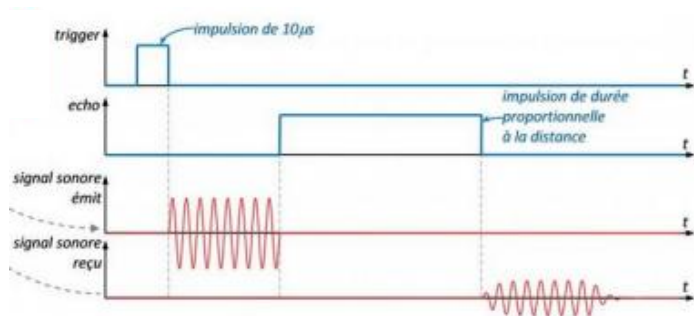


Figure 48
Diagramme chronologique des signaux[114].

8 Buzzer piézo-électrique



Figure 49
La piézoélectricité

La piézoélectricité désigne la propriété de certains matériaux de générer une charge électrique sous contrainte mécanique, et réciproquement, de se déformer en réponse à un champ électrique. Ces deux effets sont indissociables. Cette technologie trouve des applications variées, tant industrielles que domestiques. Par exemple, un allume-gaz convertit une pression mécanique en électricité, produisant ainsi une étincelle. L'effet direct (mécanique $\rightarrow$ électrique) sert notamment à fabriquer des capteurs (pression, balances), tandis que l'effet inverse (électrique $\rightarrow$ mécanique) permet de réaliser des actionneurs haute précision, comme les injecteurs automobiles, les nano-manipulateurs ou les vibreurs de téléphone[115].

Les matériaux piézoélectriques sont variés, le quartz étant le plus connu, encore employé dans les montres pour générer des impulsions. Aujourd'hui, les céramiques synthétiques, notamment les PZT, sont toutefois les plus couramment utilisées dans l'industrie.[115]

8.1 Types de modules de buzzer

8.1.1 Buzzer passif

8.1.1.1 Principe

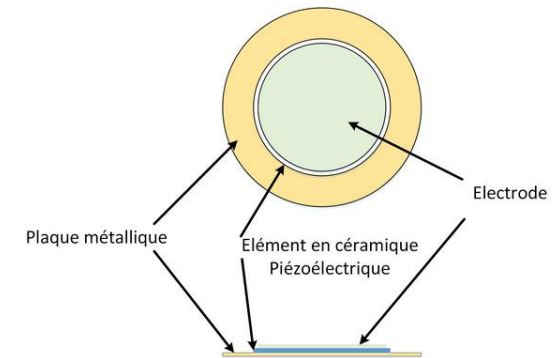


Figure 50
Buzzer passif [115].

Un buzzer passif standard intègre un disque céramique collé sur un support métallique. Une couche conductrice (généralement en argent) recouvre la face externe de la céramique. Lorsqu'une tension alternative proche de la fréquence de résonance est appliquée, l'ensemble vibre, produisant ainsi un son. La plaque métallique joue un rôle d'amplification, en renforçant la vibration de la céramique piézoélectrique et en augmentant l'intensité [115].



Figure 51
Des images du buzzer passif [115].

Le buzzer piézo passif se commercialise sous deux formats : avec fils pré-soudés ou nu. Dans ce dernier cas, le raccordement des conducteurs sur l'électrode argentée demande une technique de soudure précise pour assurer la fiabilité du montage [115].

Le buzzer peut également se monter dans un boîtier, simplifiant son implantation sur circuit imprimé. Le haut-parleur piézoélectrique, quant à lui, consiste en un disque piézo intégré dans un boîtier acoustique optimisé pour la propagation du son[115].

8.1.1.2 Utilisation

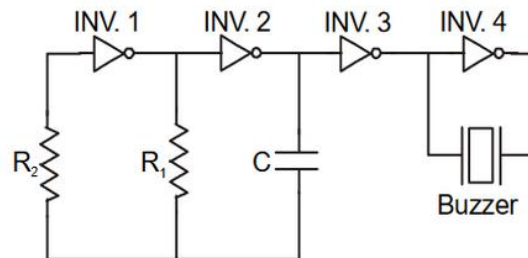


Figure 52
Un circuit inverseur à push-pull [115].

Les buzzers piézos passifs requièrent une alimentation à fréquence spécifique pour générer des vibrations. Un circuit typique utilise un oscillateur à inverseurs, comme décrit dans la documentation technique Murata. Ce guide détaille le comportement des éléments piézoélectriques et leur intégration dans ce type de montage.

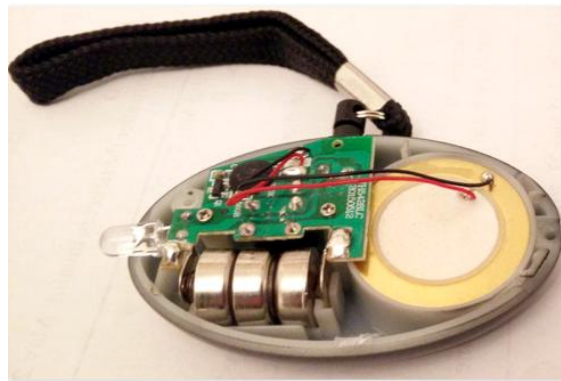


Figure 53
Implémentation d'un buzzer piézoélectrique dans un dispositif porte-clé [115].

Effectivement, il existe de nombreux autres schémas d'oscillateurs pouvant être utilisés pour alimenter un buzzer piézoélectrique passif.[115]

8.1.2 Buzzer actif

8.1.2.1 Principe

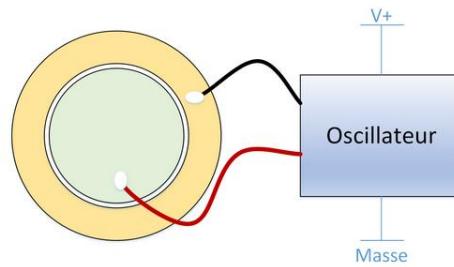


Figure 54
Le principe d'un buzzer actif [115].

Le buzzer piézo actif intègre un oscillateur interne qui remplace le circuit externe, transformant ainsi un modèle passif en version autonome capable d'émettre des sons sans alimentation fréquentielle supplémentaire. Le buzzer actif possède généralement deux pattes : une pour l'alimentation du circuit oscillateur et l'autre pour la masse. Ainsi, dès qu'on applique l'alimentation, le buzzer commence à émettre un son, ce qui simplifie grandement son utilisation par rapport au modèle passif.[115]



Figure 55
Le buzzer piézoélectrique actif [115].

Le buzzer actif est logé dans un boîtier plastique qui sert à la fois de protection pour le composant piézoélectrique et le circuit d'oscillation. Cette enveloppe agit également comme caisse de résonance, amplifiant le signal acoustique et renforçant le volume sonore. Sur l'image à droite, on peut voir un buzzer "éventré", où l'on aperçoit le circuit imprimé de l'oscillateur à l'intérieur du boîtier, ainsi que la plaquette piézoélectrique, qui est légèrement soulevée. Ce design permet de visualiser clairement les composants internes responsables de la production du son.[115]

8.1.2.2 Images du buzzer actif



Figure 56
Des images du buzzer actif [115].

Ci-dessus, on peut observer différentes options pour utiliser un buzzer actif. Le premier modèle est simplement un buzzer autonome, Ce composant peut être fixé sur un PCB ou employé directement sur une platine de test. Le second modèle (en haut à droite) est un buzzer actif basique produisant un signal sonore dès sa mise sous tension[115].

Le troisième schéma (bas/gauche) intègre un transistor commutateur en amont du buzzer. Cette configuration autorise la commande via une sortie GPIO sans en solliciter le courant. Le transistor joue alors un rôle d'interrupteur, activant le buzzer tout en épargnant les ressources du microcontrôleur[115].

8.1.2.3 Caractéristiques

Le module KY-012 (et ses équivalents) supporte une tension de 3 à 24 V en courant continu, le rendant compatible avec les sorties 3,3 V des GPIO. Sa fréquence nominale est de 2,5 kHz (mesurée à ~2,7 kHz en pratique) pour une consommation d'environ 25 mA. Livré avec un cache protecteur sur l'évent acoustique, ce buzzer nécessite le retrait de cette protection après nettoyage de la carte électronique en production industrielle[115].

Initialement, mieux vaut garder le cache en position, atténuant ainsi quelque peu le volume. Cette précaution s'avère particulièrement utile avec de jeunes enfants, dont l'audition est plus fragile face aux sons aigus[115].

8.1.2.4 Connexion au Raspberry Pi

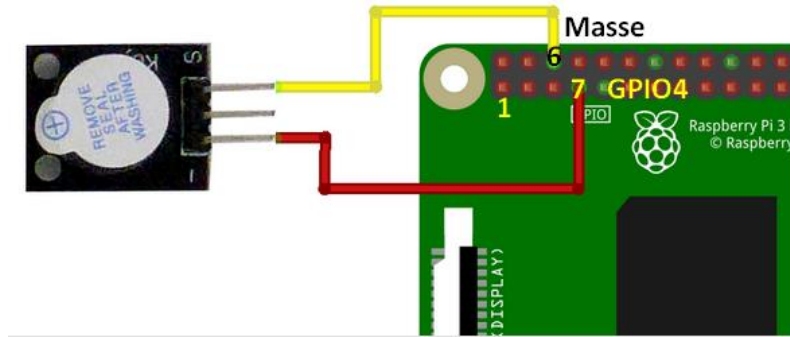


Figure 57
Connexion buzzer actif au Raspberry Pi [115].

Rien de bien compliqué ici. La masse (patte 6 du GPIO du Raspberry Pi) doit être connectée à la borne S (probablement le signal) du module. La sortie du GPIO 4 (broche 7) sera quant à elle reliée à la broche négative (-) du module Keyes KY-012. Il semble cependant que le marquage des broches soit parfois inversé, ce qui est confirmé pour certains modules. Cela peut prêter à confusion, alors assurez-vous de vérifier attentivement la documentation de votre module avant de faire les connexions.[115]



Figure 58
Configuration minimale à 2 fils pour connexion GPIO [115].

Ce chapitre a exploré les composants clés de notre système, notamment le moteur pas à pas 28BYJ-48, le driver ULN2003, le Raspberry Pi 4, la caméra Raspberry Pi, le module GPS NEO-6M, le capteur ultrason HC-SR04 et les buzzers. Chaque composant a été décrit en détail, avec ses caractéristiques, son fonctionnement et ses applications. Ces éléments, combinés, forment la base matérielle de notre système de surveillance intelligent, permettant une détection précise, une communication efficace et une réponse rapide aux dangers dans les mines à ciel ouvert.

Chapitre 6 Réalisation du système

I Introduction

Ce chapitre détaille la mise en œuvre pratique d'un radar à ultrasons. Après avoir couvert les bases théoriques – principes radar, fonctionnement du Raspberry Pi, contrôle du moteur pas-à-pas, utilisation du capteur ultrasonique et méthodes de visualisation – nous abordons maintenant l'implémentation concrète du système, incluant sa structure de classification des cibles, nous allons maintenant détailler l'application expérimentale de cette technologie. Pour ce faire, plusieurs blocs fonctionnels ont été intégrés afin de créer une solution cohérente, en utilisant la carte Raspberry Pi 4 B, les bibliothèques Python adaptées ainsi que des outils tels que Pygame pour l'affichage instantané ainsi la gestion des entrées/sorties. Cette section présente les phases clés de réalisation, depuis l'installation des éléments jusqu'au développement d'un système radar opérationnel intégrant détection et visualisation.

1 Principe de fonctionnement

Le principe de fonctionnement du radar à capteur ultrasonique repose sur l'utilisation d'un capteur à ultrasons pour mesurer la distance entre l'objet détecté et le capteur, en utilisant architecteur de classification des objets YOLOv10 par la caméra tout en balayant la zone à 270° à l'aide d'un moteur pas à pas. L'application Telegram utilisé pour les notifications, Ce système est intégré à une carte Raspberry Pi 4 B, qui assure la gestion des données et le contrôle des composants.

2. Les étapes du projet

- 1^{er} étape la vérification des composants et la simulation avec Proteus 8
- 2^{eme} étape la réalisation pratique de projet

Que ce soit les deux étapes on passe par :

Dans les deux étapes, qu'il s'agisse de la programmation par blocs a traves langage Python ou de la phase de visualisé les résultats, nous utilisons l'affichage sous Proteus LCD bien que l'écran de Pc, auxquels nous accédons grâce à des outils comme RealVNC Viewer.

3. Schéma Synoptique de radar

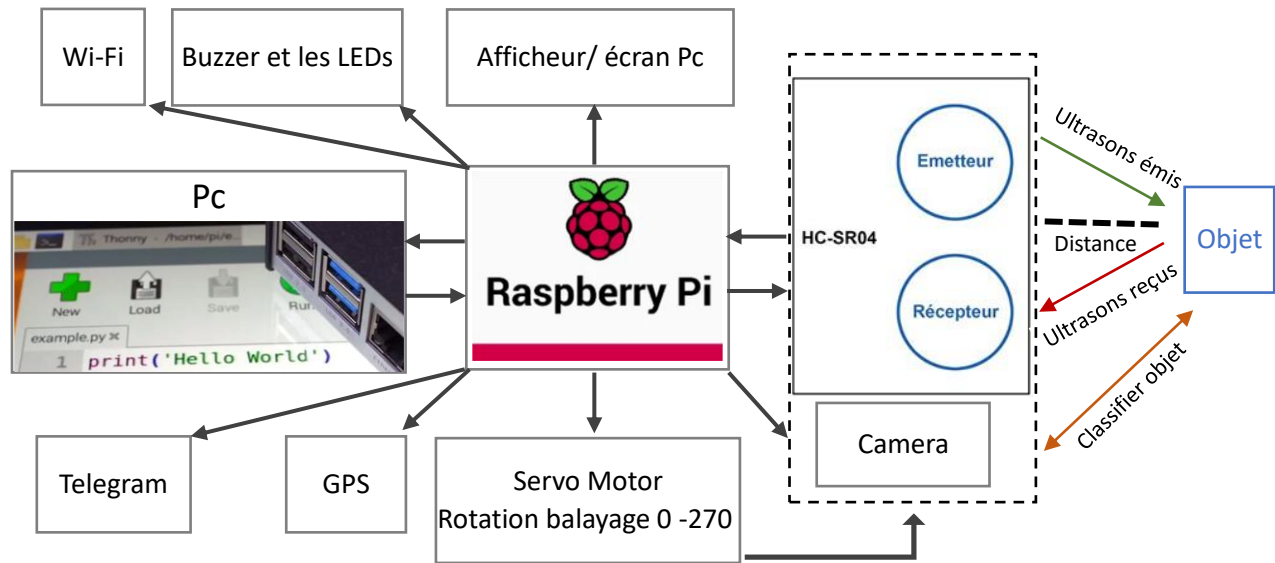


Figure 59
Schéma-bloc du projet radar expérimental

4.Algorithme principale de détection et classification d'un objet

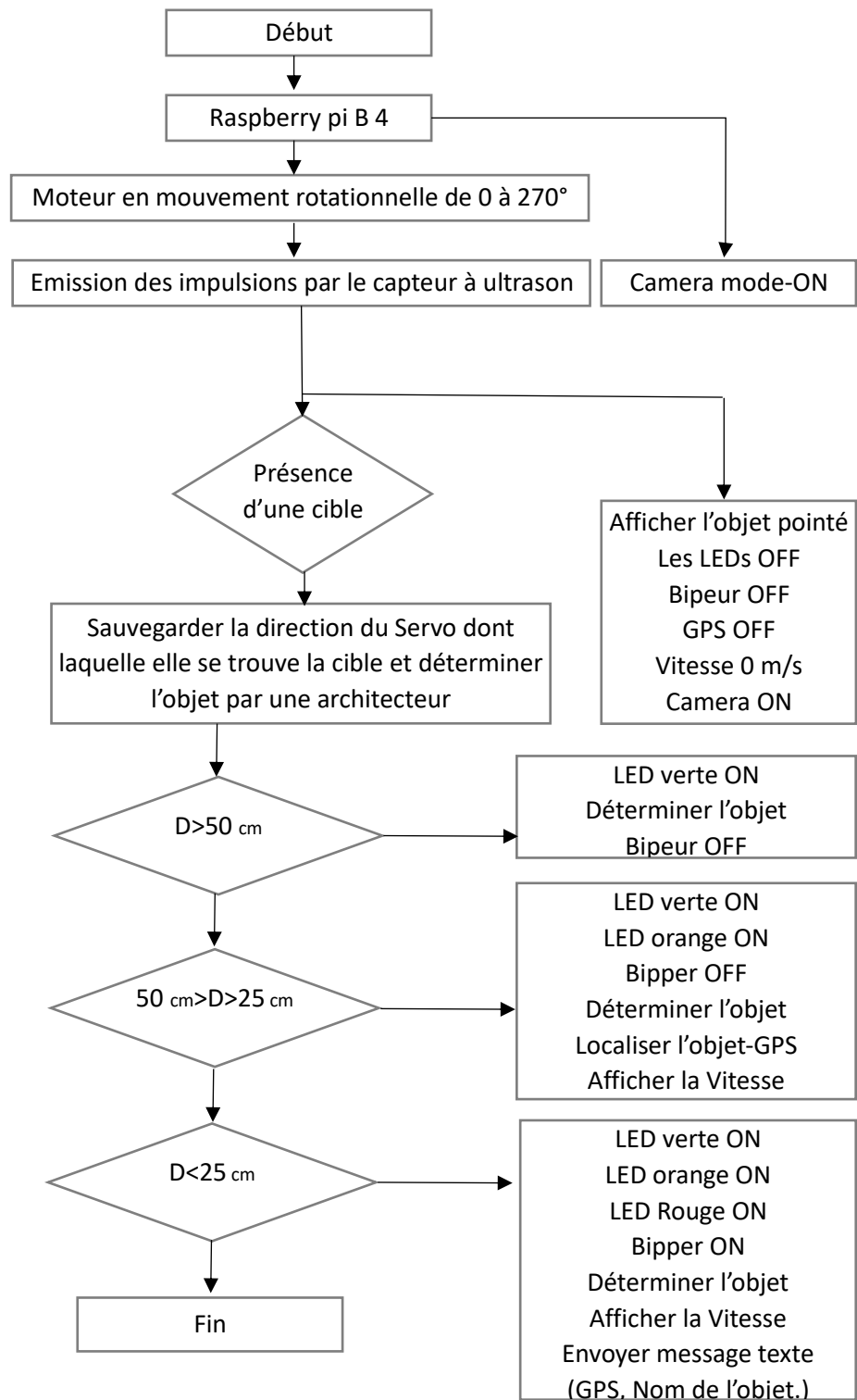


Figure 60
Logique algorithmique centrale du système

5. Simulation du projet sous PROTEUS

5.1. Bibliothèque carte la carte Raspberry pi pour Proteus 8

Pour modéliser une Raspberry Pi dans Proteus, nous avons dû télécharger les fichiers « RaspberryPi4TEP.IDX » et « RaspberryPi4TEP.LIB », puis les copier dans le dossier bibliothèque du logiciel, généralement situé dans :

C:\ProgramFiles(x86)\Labcenter_Electronics\Proteus_8_Professional\DATA\LIBRARY

Cette méthode permet d'ajouter le composant directement dans nos schémas.

Après le démarrage de logiciel Proteus on choisit RASBERRY PI 4 ou du logiciel par default RPI4 comme illustré à la figure 76

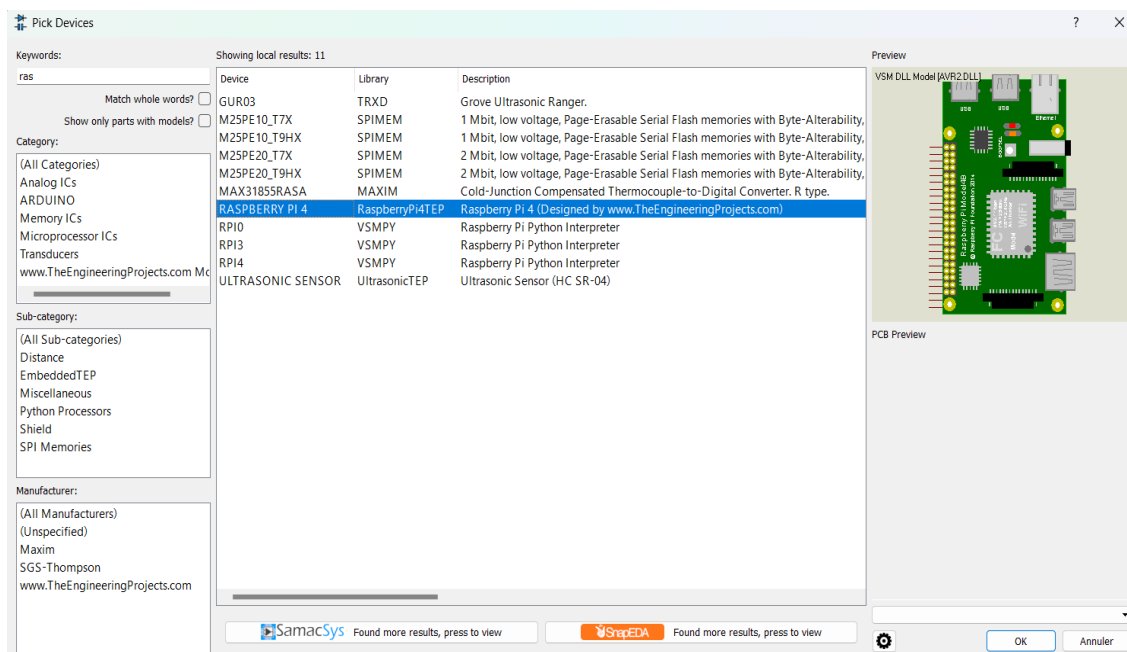


Figure 61
L'intégration de la carte Raspberry pi 4 sous Proteus.

Maintenant la carte Raspberry pi 4 est prête à être utiliser pour la simulation

5.2. Bibliothèque de capteurs à ultrasons pour PROTEUS 8

Pour connecter le capteur ultrasonique à la Raspberry Pi 4, nous avons d'abord téléchargé la bibliothèque Proteus 8 nécessaire. Elle contenait trois fichiers : 'UltrasonicTEP.IDX',

'UltrasonicTEP.LIB' et 'UltrasonicTEP.HEX'. Nous les avons ensuite copiés dans le dossier approprié de Proteus :

C:\Program_Files_(x86)\Labcenter_Electronics\Proteus_8_Professional\DATA\LIBRARY

Initialement, le capteur à ultrasons n'était pas reconnu par le logiciel. Pour y remédier, il suffit d'accéder à ses propriétés de capteur. Dans la section réservée au fichier du programme, on sélectionne ensuite « UltrasonicTEP.HEX », comme montré dans l'illustration.

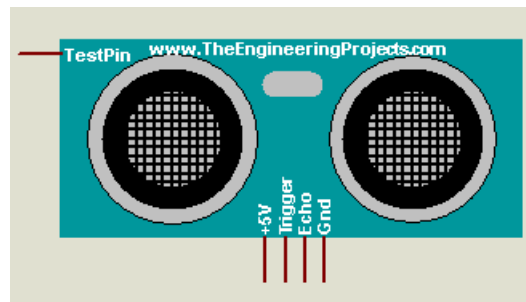


Figure 62
Simulation du capteur HC-SR04 dans logiciel Proteus 8.

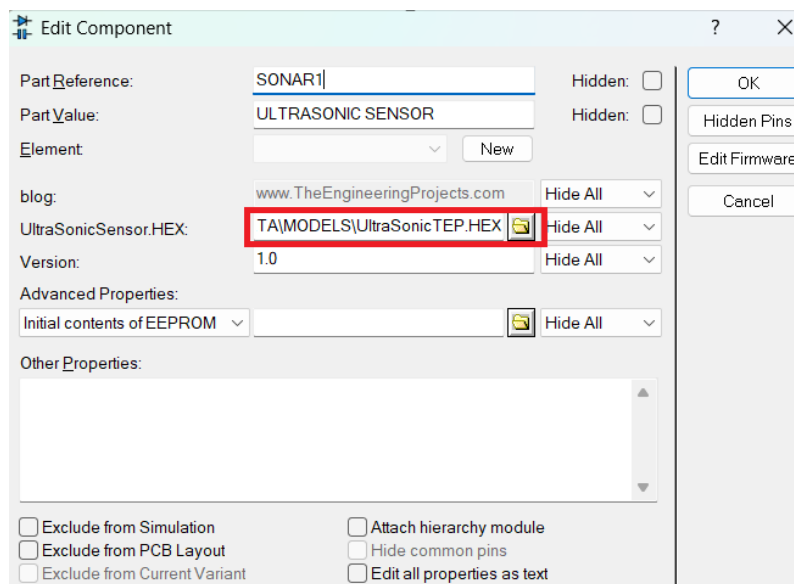


Figure 63
Fichier hexadécimal UltrasonicTEP dans Proteus 8.

A ce moment-là l'élément HC-SR04 est prête à être utiliser pour la simulation

5.3. Bibliothèque de GPS pour Proteus 8

Pour connecter le capteur GPS à la carte Raspberry Pi 4 sous logiciel Proteus 8, nous avons commencé par récupérer en ligne la bibliothèque adaptée à Proteus 8. Celle-ci se compose de trois fichiers qui sont importantes : « GPSTEP.IDX », « GPSTEP.LIB » et « GPSTEP.HEX ». Une fois le téléchargement terminé, nous les avons placés dans le dossier approprié de Proteus, situé par défaut à l'emplacement suivant :

C:\Program_Files_(x86)_\Labcenter_Electronics\Proteus_8_Professional\DATA\LIBRARY

Dans un premier temps, le logiciel n'a pas détecté le capteur GPS. Pour résoudre ce problème, il suffit d'accéder aux propriétés du composant. Dans la section dédiée aux fichiers du programme, on sélectionne ensuite « GPSTEP.HEX », comme illustré sur l'image.

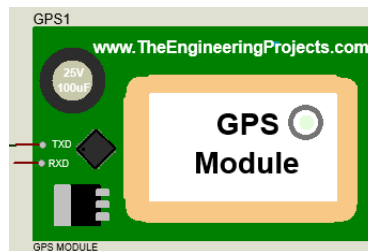


Figure 64
Le GPS sous Proteus 8.

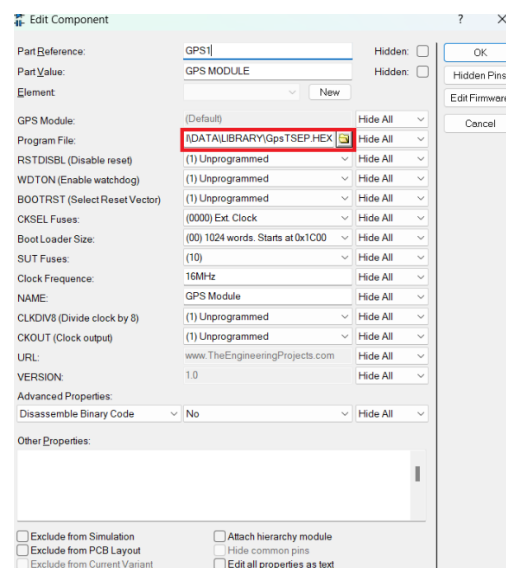


Figure 65
Fichier GpsTSEP.HEX sous Proteus 8.

5.4. Configurer la caméra pour PROTEUS 8

Premièrement, on accède au code source. On fait un clic droit sur le fichier « rpi4 » puis on sélectionne « Add peripheral ». Une fois la caméra placée, on utilise le bouton « Breakout board ». Pour le contrôler manuellement

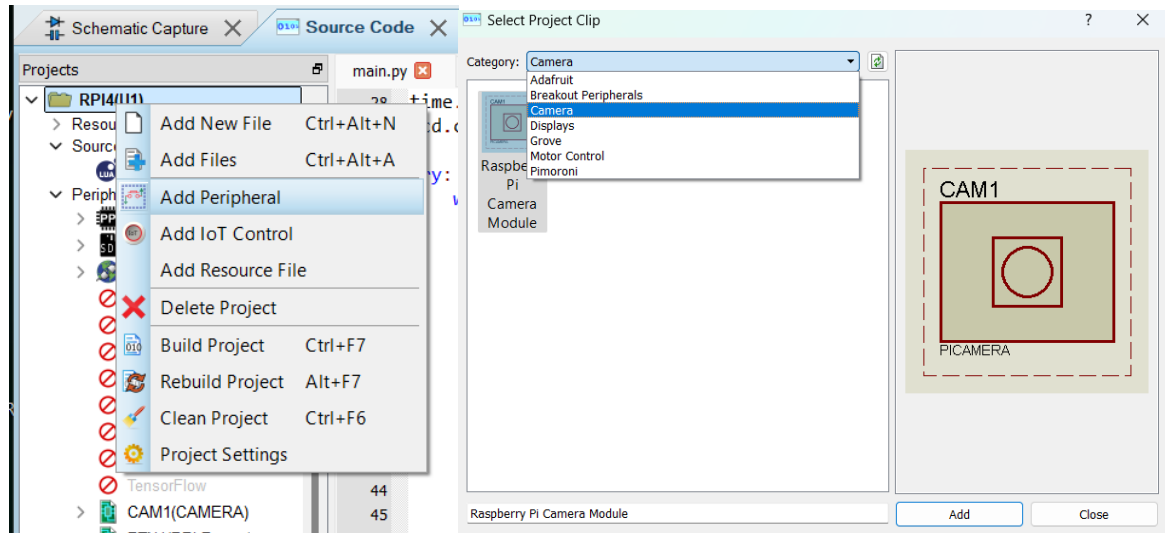


Figure 66
La caméra sous Proteus 8.

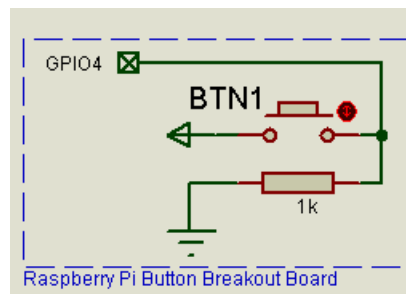


Figure 67
Control camera sous Proteus 8.

Circuit global de la simulation

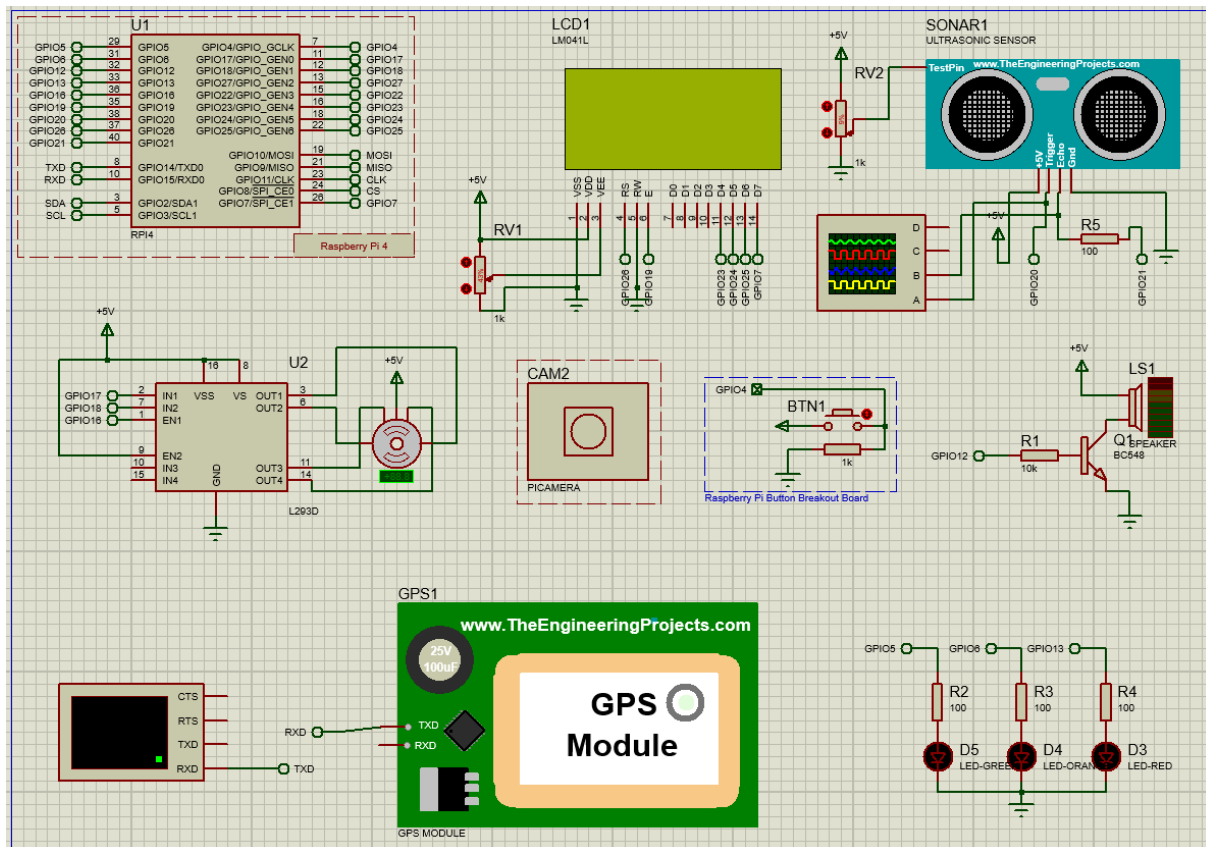


Figure 68
Circuit électronique globale de simulation.

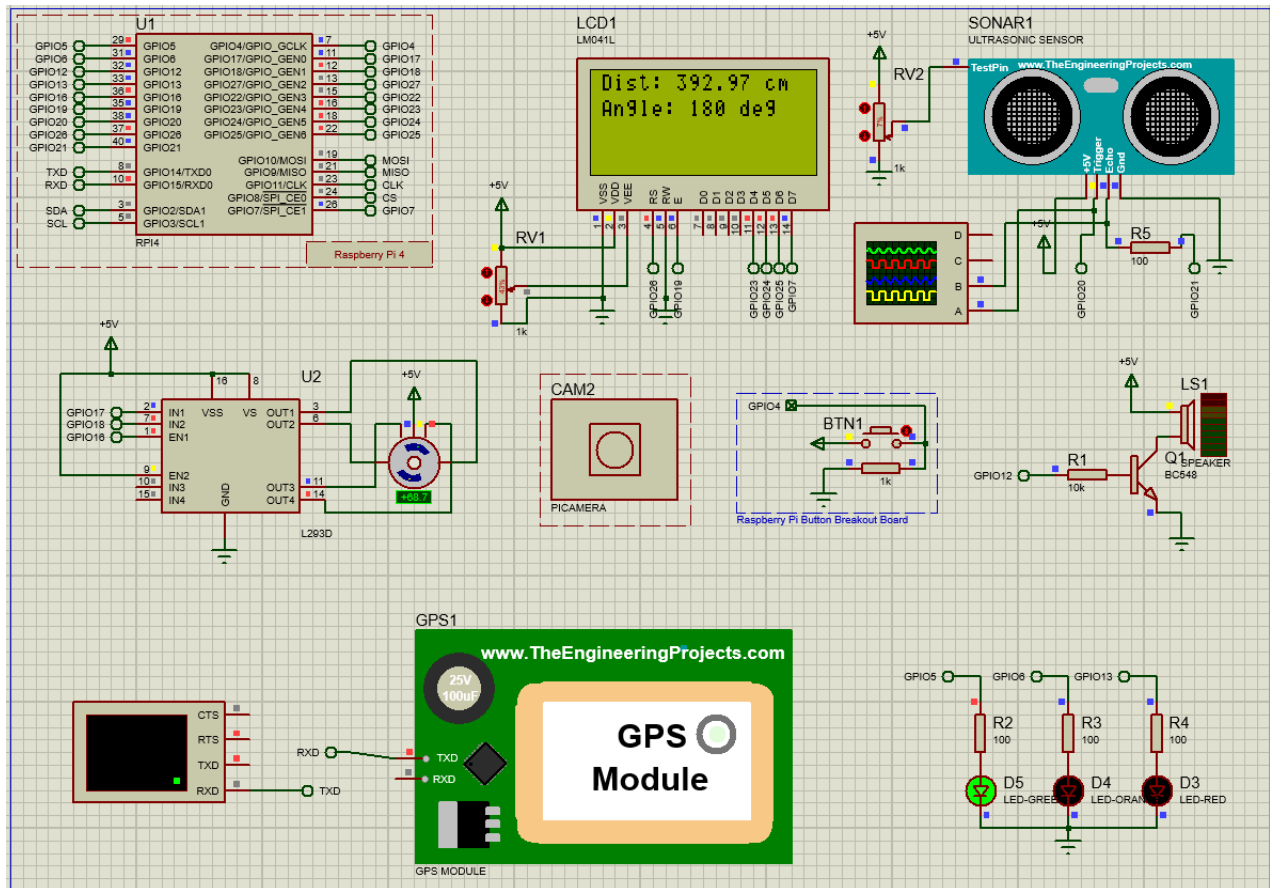
Les différentes connexions et numéro des pins de circuit sont présentés dans le tableau 8

Tableau 6
Correspondance des broches et connexions du circuit

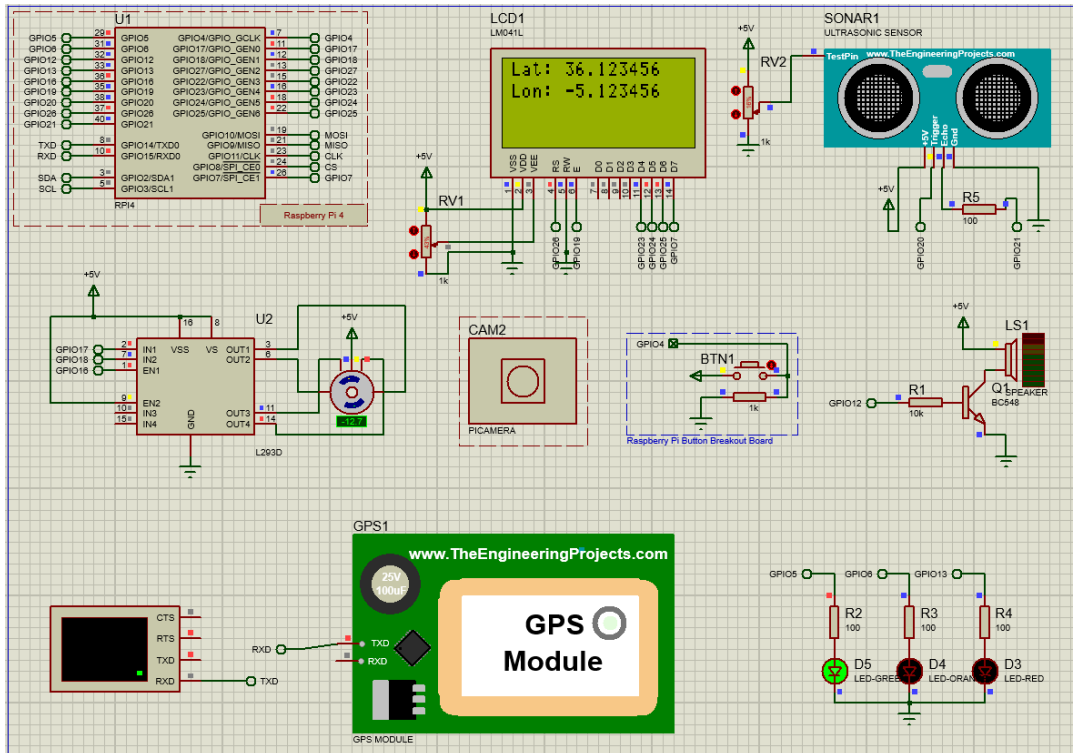
Module / Composant	Broche (Module)	Connecté à (Raspberry Pi)	GPIO	Autres détails
SONAR1 (capteur ultrason)	VCC	+5 V	–	Alimentation +5 V
	TRIG	GPIO20	20	Broche « Trigger » du capteur
	ECHO	GPIO21	21	Broche « Echo » (avec R5 = 100 Ω)
	GND	Masse (GND)	–	
L293D (U2) (driver moteur)	IN1 (pin 2)	GPIO17	17	Entrée de commande moteur
	IN2 (pin 7)	GPIO18	18	Entrée de commande moteur
	EN1 (pin 1)	GPIO16	16	Control vitesse de moteur
	OUT1, OUT2, OUT3, OUT4	Vers le moteur	–	Sorties vers moteur CC
	VSS (pin 16)	+5 V	–	Alimentation logique L293D
	VS (pin 8)	+5 V	–	Alimentation moteur
	EN2	+5V	–	Alimentation logique L293D
	GND	Masse (GND)	–	
GPS1 (Module GPS)	TXD	RXD du RPi (UART)	(UART)	Transmission série (UART)
	RXD	TXD du RPi (UART)	(UART)	Réception série (UART)
	VCC	+5 V (ou 3,3 V selon module)	–	À vérifier selon le GPS utilisé
	GND	Masse (GND)	–	
CAM1 (PiCamera)	– (interface CSI)	Connecteur caméra du RPi	–	Simulation symbolique dans Proteus
BTN1 (bouton)	–	GPIO4	4	Via résistance de 1 kΩ
LS1 (haut-parleur)	–	GPIO12	12	Via transistor Q1 (BC548) + R1=10 kΩ
LED Verte (D3 + R2)	Anode (D3)	GPIO5	5	Résistance R2 = 100 Ω
LED Orange (D2 + R3)	Anode (D2)	GPIO6	6	Résistance R3 = 100 Ω
LED Rouge (D1 + R4)	Anode (D1)	GPIO13	13	Résistance R4 = 100 Ω
LCD1 (LM041L)	D4.	GPIO24	–	R/W à la masse (souvent), potentiomètres RV1/RV2 pour le contraste
	D5	GPIO25	25	(Mode 4 bits)
	D6	GPIO26	26	(Mode 4 bits)
	D7	GPIO7	7	(Mode 4 bits)
	RS	GPIO26	MOSI	À vérifier si pas de conflit avec d'autres périphériques.
	E	GPIO19	19	Permet le latch des données dans le LCD
	VEE	Potentiomètre		Régler le contraste
	VSS	Vers GND Potentiomètre		
	VDD	Potentiomètre vers +5V		Alimentation +5 V
	RW	Masse (GND)		

6. Résultats de la simulation

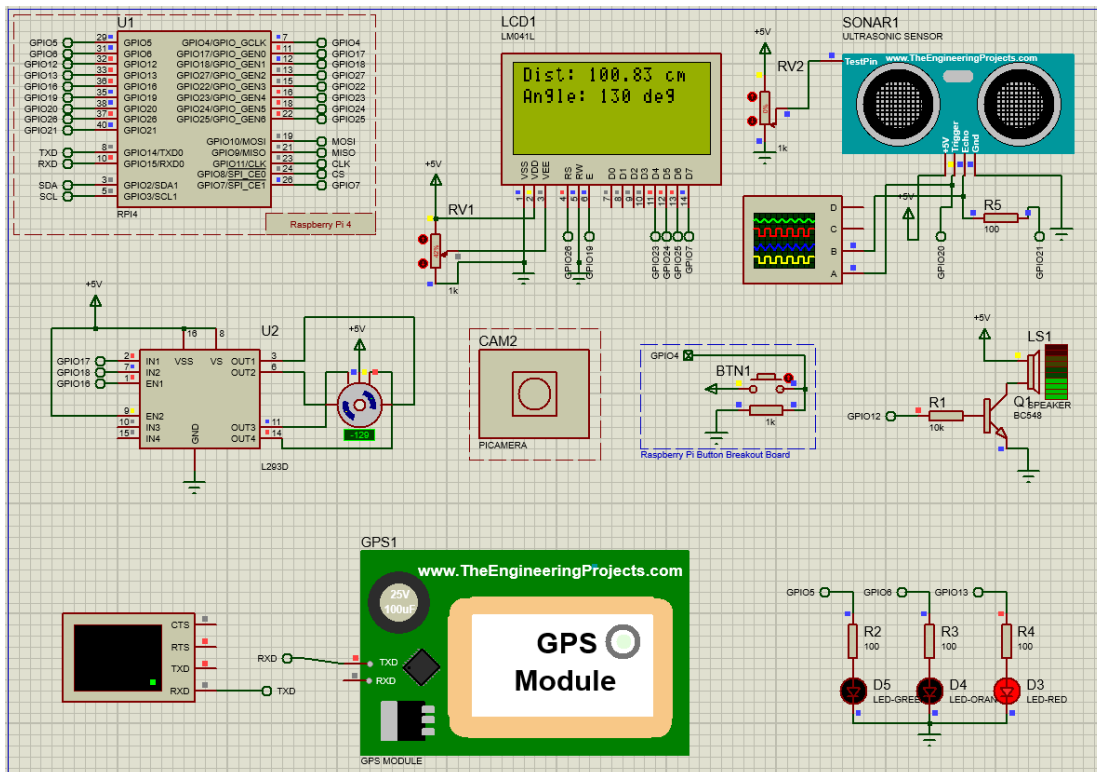
Ces captures présentent nos résultats de simulation obtenus avec Proteuse 8. Les visualisations montrent que le système fonctionne très bien selon les conditions distances également la précision, stabilité et l'efficacité. Ces données confirment la fiabilité de système et serviront pour l'application miniers



(a)



(b)



(c)

Figure 69

(a),(b)et(c) Circuit électronique globale de simulation

Pour notre simulation on a mis les conditions suivantes

```

91     return distance
92
93     #####
94     ##### Functions for the LEDs and buzzer #####
95     #####
96
97     def control_leds_buzzer(distance):
98         """Control the LEDs and buzzer based on the distance."""
99         # Turn off all LEDs and the buzzer by default
100         GPIO.output(LED_GREEN, GPIO.LOW)
101         GPIO.output(LED_ORANGE, GPIO.LOW)
102         GPIO.output(LED_RED, GPIO.LOW)
103         GPIO.output(BUZZER, GPIO.LOW)
104
105         # Conditions to turn on the LEDs and buzzer
106         if distance > 500:
107             # Nothing is turned on if the distance is greater than 500 cm
108             pass
109         elif 300 < distance <= 500:
110             # Turn on the green LED if the distance is between 300 cm and 500 cm
111             GPIO.output(LED_GREEN, GPIO.HIGH)
112         elif 100 < distance <= 300:
113             # Turn on the orange LED if the distance is between 100 cm and 300 cm
114             GPIO.output(LED_ORANGE, GPIO.HIGH)
115         elif 0 <= distance <= 100:
116             # Turn on the red LED and activate the buzzer if the distance is between 0 cm and 100 cm
117             GPIO.output(LED_RED, GPIO.HIGH)
118             GPIO.output(BUZZER, GPIO.HIGH)
119
120     #####
121     ##### Functions for the motor #####
122

```

VSM Studio Output
Code insertion tags in file 'main.py' were not found - peripherals must be configured manually.
Already compiled, target is up to date.
Compiled successfully.

Figure 70

les conditions de fonctionnement les LEDs

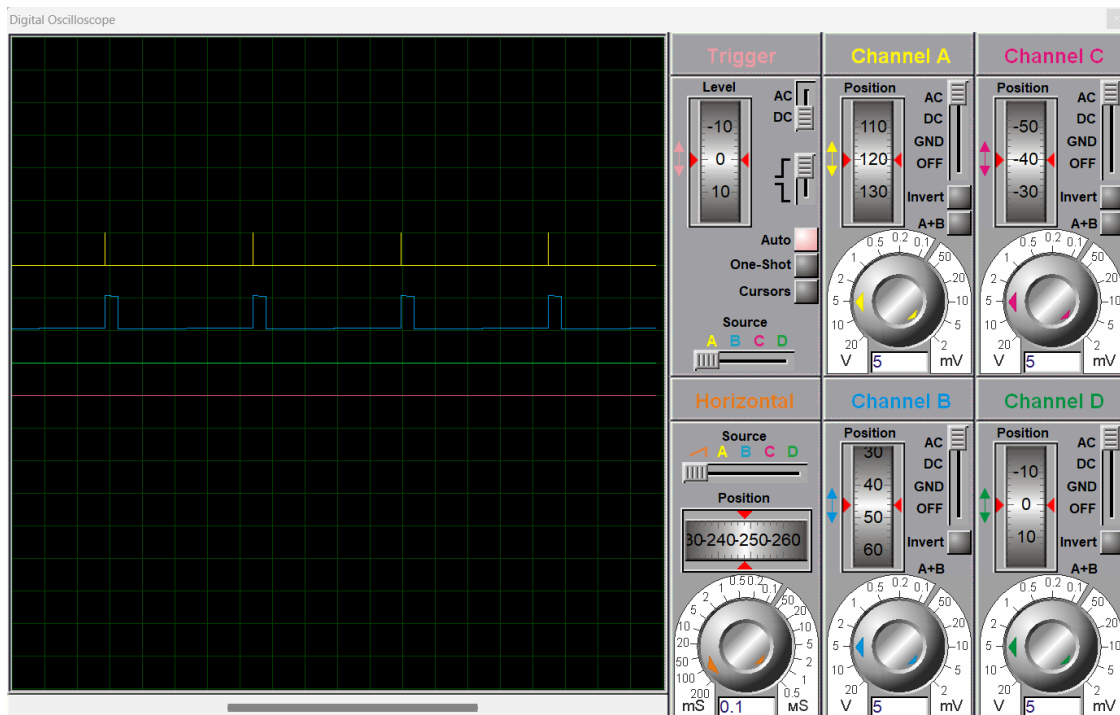


Figure 71

Représentation des impulsions Trigger et Echo du capteur à ultrasons.

La simulation affiche les valeurs de distance, d'angle, de vitesse et de GPS, qui sont visualisées sur l'écran LCD à intervalles réguliers d'une seconde. Ces valeurs sont directement influencées par la résistance variable associée au capteur à ultrasons. En ajustant la tension de cette résistance variable, que ce soit en l'augmentant ou en la diminuant, les mesures de distance, d'angle et de vitesse sont modifiées en conséquence, reflétant ainsi les changements dans l'environnement détecté par la simulation de système. On peut voir le fonctionnement du capteur comme indiqué sur oscilloscope dans la figure.

Pour la caméra nous allons la tester sur la plateforme pythone par un programme de détection des gestes des mains, voir la figure 87.

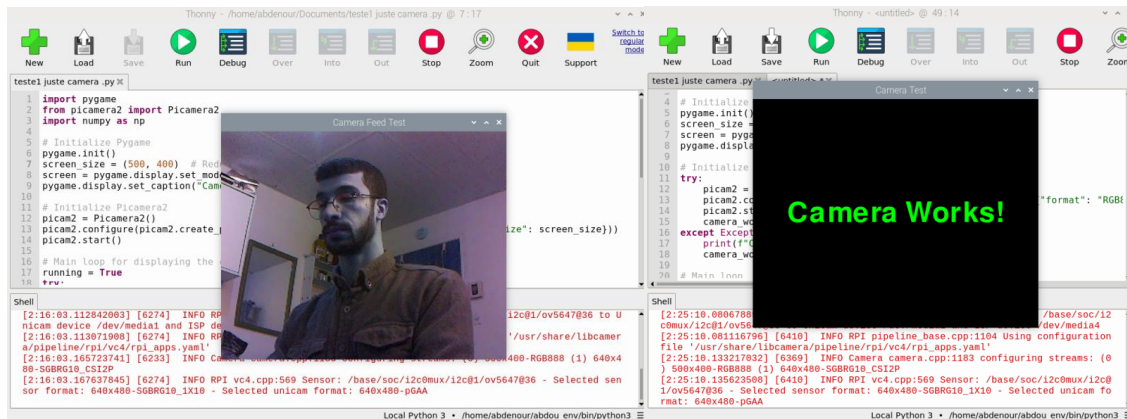


Figure 72
Teste fonctionnement de caméra

Nous avons créé un script qui nous permet de tester si le GPS fonctionne correctement.

The screenshot shows the Thonny Python IDE interface. The main editor window displays a Python script for connecting to a GPS module using the PySerial library. The script includes imports for pygame, sys, serial, and pynput. It initializes pygame, sets up a screen and font, and attempts to open a serial port. A try-except block handles potential serial exceptions. A modal window titled 'GPS Data Display with Pygame' is overlaid on the code, displaying the status 'Status: GPS Connected' and 'Waiting for GPS data...'. Below the code editor, a terminal window shows the raw output of the GPS module, including NMEA sentences like '\$GNGGA', '\$GPRMC', '\$GNVTG', '\$GPGSV', and '\$GLGSV'.

```

1 import pygame
2 import sys
3 import serial
4 import pynput
5
6 # Initializ
7 pygame.init
8
9 # Set up th
10 screen = py
11 pygame.disp
12 font = pyga
13
14 # Set up th
15 try:
16     ser = s
17     print("
18     gps con
19 except serial.SerialException as e:

```

```

Raw GPS data: $GNGGA,045910.00,4814.19693,N,07900.83853,W,1,09,1.51,297.1,M,-37.8,M,,0.0,0.0,0.0,0.0
Raw GPS data: $GNGSA,A,3,26,05,29,11,18,20,0.0,1.51,1.37,18
Raw GPS data: $GNGSA,A,3,79,87,88,0.0,1.51,1.37,18
Raw GPS data: $GPGSV,3,1,12,05,77,043,18,07,09,040,,11,25,111,14,13,48,139,11*7F
Raw GPS data: $GPGSV,3,2,12,15,35,189,,16,05,335,,18,30,298,23,20,41,069,15*7A
Raw GPS data: $GPGSV,3,3,12,23,07,249,,26,07,308,20,29,60,248,22,30,12,074,*79
Raw GPS data: $GLGSV,3,1,09,65,30,209,,71,35,038,19,72,02,111,,79,09,329,16*60
Raw GPS data: $GLGSV,3,2,09,80,05,018,,86,26,130,,87,74,110,10,88,40,319,28*69
Raw GPS data: $GLGSV,3,3,09,90,26,130,*53
Raw GPS data: $GNGLL,4814.19693,N,07900.83853,W,045910.00,A,A*69
Raw GPS data: $GNRMC,045919.00,A,4814.19693,N,07900.83911,W,1.166,,160125,,A*7C
Raw GPS data: $GNVTG,T,M,1.166,N,2.160,K,A*38
Raw GPS data: $GNGGA,045919.00,4814.19693,N,07900.83911,W,1,09,1.51,297.1,M,-37.8,M,,0.0,0.0,0.0,0.0
Raw GPS data: $GNGSA,A,3,26,05,29,11,18,20,0.0,1.51,1.37,18
Raw GPS data: $GNGSA,A,3,79,87,88,0.0,1.51,1.37,18
Raw GPS data: $GPGSV,3,1,12,05,77,043,17,07,09,040,,11,25,111,14,13,48,139,12*73

```

Figure 73

Le test pour le fonctionnement du GPS

Ce chapitre a présenté la réalisation pratique de notre système de surveillance radar intelligent. Nous avons détaillé le principe de fonctionnement, les étapes du projet, et le schéma synoptique du radar. L'algorithme principal de détection et de classification des objets a été expliqué, suivi d'une simulation complète sous Proteus 8, incluant la configuration des bibliothèques pour la carte Raspberry Pi, les capteurs à ultrasons, le GPS et la caméra. Les résultats de la simulation ont validé la faisabilité du système, confirmant son potentiel pour une détection précise et une réponse rapide aux dangers dans les mines à ciel ouvert.

Réalisation pratique du projet

Pour débuter avec notre Raspberry Pi, nous aurons besoin des éléments suivants :

- Une alimentation électrique
- Un support de démarrage (par exemple, une carte microSD offrant une capacité de stockage et une vitesse adéquate)

Nous pouvons configurer notre Raspberry Pi soit comme un ordinateur interactif avec une interface graphique, soit comme un appareil sans tête accessible uniquement via le réseau. Pour une configuration sans tête, aucun périphérique supplémentaire n'est nécessaire : nous pouvons préconfigurer un nom d'hôte, un compte utilisateur, une connexion réseau et SSH lors de l'installation du système d'exploitation. Si nous souhaitons utiliser notre Raspberry Pi directement, des accessoires supplémentaires seront requis :

Un écran -Pc

Un câble pour connecter notre Raspberry Pi à l'écran

Un clavier

Une souris

1.Alimentation électrique

Pour assurer le bon fonctionnement de notre Raspberry Pi 4 ou 5, il est essentiel de connaître le mode d'alimentation USB-PD requis. Nous pouvons utiliser toute alimentation de haute qualité qui fournit le mode d'alimentation approprié.

Raspberry Pi 4

Nous aurons besoin d'une alimentation de 5V/5A. Si nous optons pour une alimentation de 5V/3A, les périphériques connectés seront limités à 600 mA. L'alimentation officielle pour ce modèle est une alimentation USB-C de 27 W.

Raspberry Pi 4 Modèle B

L'alimentation de 5V/3A est recommandée toujours ce qui concernant ce modèle. L'alimentation officielle proposée est une alimentation USB-C de 15 W



Figure 74
Alimentation de carte Raspberry pi[135].

Nous connectons notre alimentation au port identifié par "POWER IN", "PWR IN" ou "PWR". Il est important de noter que certains modèles de Raspberry Pi, comme la série Zero, possèdent des ports USB de sortie ayant le même format que le port d'alimentation [116].

2.Démarrez Media

Les modèles Raspberry Pi ne possèdent pas de stockage interne, nécessitant l'utilisation d'un support externe. Le démarrage peut se faire via une image système sur différents supports, tels que des cartes microSD (les plus courantes), des périphériques USB, un stockage réseau ou un HAT PCIe. Cependant, seuls les modèles récents prennent en charge toutes ces options. Depuis le Raspberry Pi 1 Modèle A+, tous les modèles incluent un emplacement pour carte microSD, permettant un démarrage automatique si une carte est insérée [135].

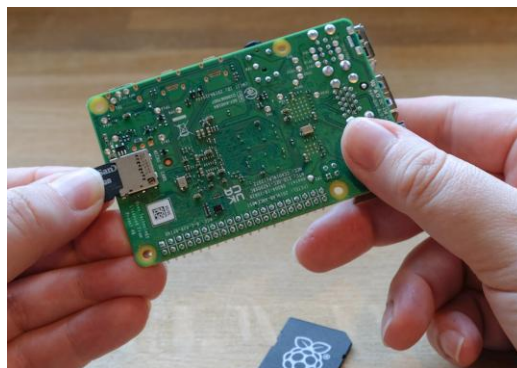


Figure 75
Démarrer le media[135]

Nous avons privilégié une carte SD d'au moins 32 Go pour l'installation de Raspberry Pi OS. Pour la version Lite, nous avons maintenu cette même capacité minimale. Nous pouvons utiliser n'importe quelle carte SD dont la capacité est inférieure à 2 To. Les capacités supérieures à 2 To ne sont pas prises en charge actuellement en raison des limitations du MBRA[116]. Comme pour tout support de démarrage, vous remarquerez des performances améliorées avec des cartes SD offrant des vitesses rapide de lecture ainsi la l'écriture [116].

3.Clavier

Nous pouvons brancher un clavier filaire ou un récepteur USB Bluetooth sur n'importe quel port USB de notre Raspberry Pi.



Figure 76
Installation d'un clavier[135].

4.Souris

Nous pouvons utiliser n'importe quel port USB de notre Raspberry Pi pour brancher une souris filaire ou un récepteur USB Bluetooth.



Figure 77
Installation d'une souris[135].

5.Réseautage

Le suffixe "Model B" indique les variantes de Raspberry Pi équipées d'un port Ethernet, tandis que "Model A" concerne celles sans ce port. Si notre Raspberry Pi ne dispose pas de port Ethernet, nous pouvons toujours nous connecter à une connexion Internet filaire en utilisant un adaptateur USB vers Ethernet. Les modèles Raspberry Pi 4 et 5 sont livrés avec une connectivité Wi-Fi et Bluetooth[116].

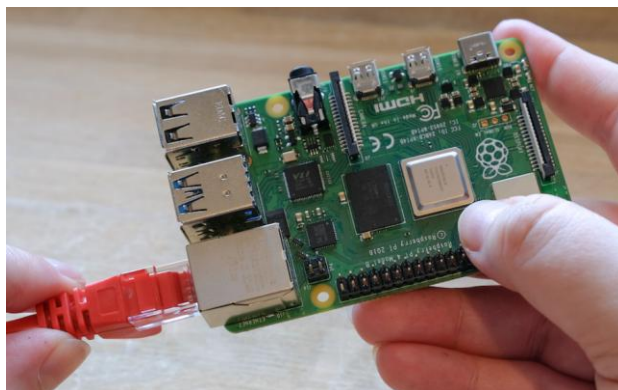


Figure 78
Installation d'un réseau [135]

6.Installer un système d'exploitation

Pour utiliser un Raspberry Pi, il faut installer un système d'exploitation. Celui-ci est généralement détecté automatiquement sur une carte SD, mais certains modèles permettent de démarrer depuis une clé USB, un stockage réseau ou un HAT. Pour configurer le périphérique de démarrage, un ordinateur est nécessaire. La plupart des utilisateurs optent pour une carte microSD, et l'outil Raspberry Pi Imager, compatible avec macOS, Windows et Linux, est recommandé pour l'installation[116].

7.Installation d'image

Nous pouvons installer Imager de la manière suivante :

1. Nous avons téléchargé la dernière version sur [raspberrypi.com/logiciel](https://www.raspberrypi.com/software/) puis lancé l'installation.

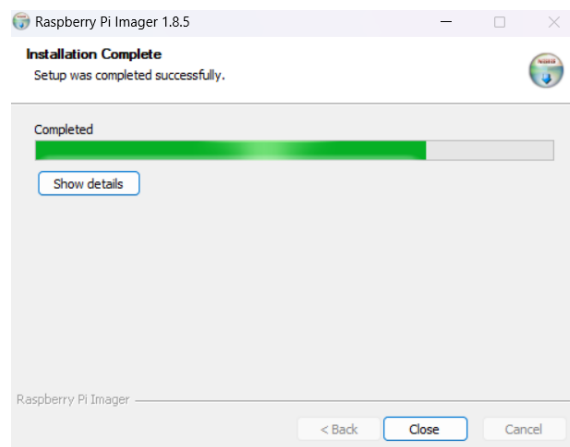
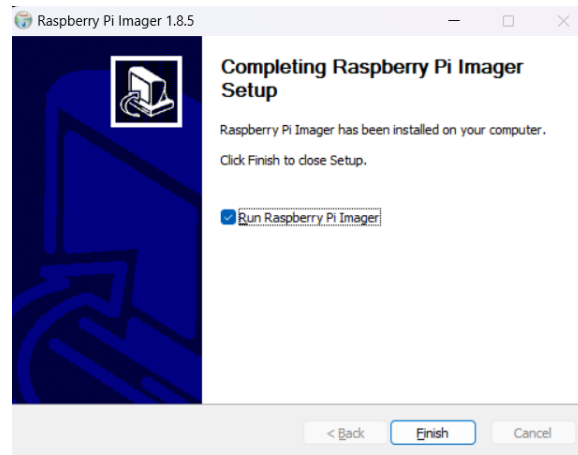
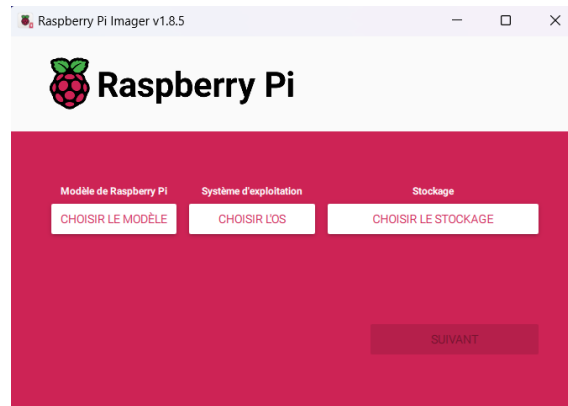


Figure 79
les étapes d'installation d'image (a,b,c,d,e,f,i,g,k,l,m,n) .

2. Nous l'avons installé via le terminal avec notre gestionnaire de paquets, par exemple :
`sudo apt install rpi-imager`.
3. Après installation d'Imager, lancez le logiciel en cliquant sur son icône ou via la commande terminal 'rpi-imager'.

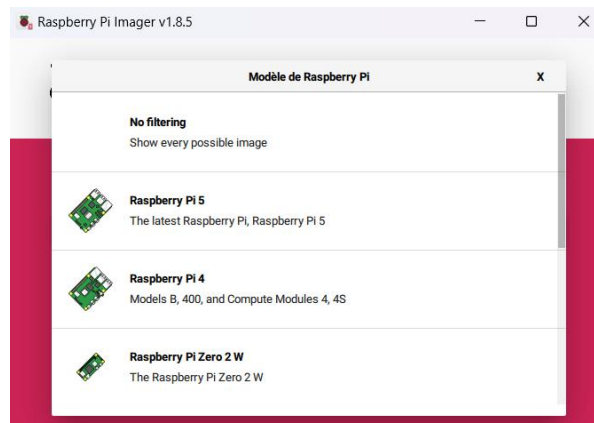


(a)



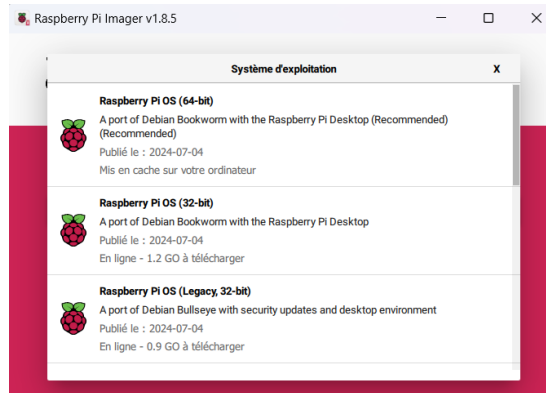
(b)

Puis, on choisit le modèle de Raspberry Pi dans la liste indiqué dans (c)



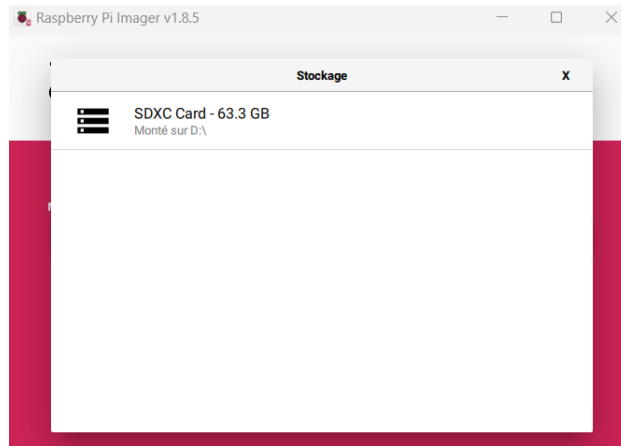
(c)

Ensuite, nous cliquons sur "Choisissez OS" et sélectionnons le système d'exploitation à installer. L'outil Imager affiche toujours en tête de liste la version recommandée de Raspberry Pi OS adaptée à notre modèle.



(d)

Nous connectons notre périphérique de stockage, une carte Micro-SD de 64GB, à notre ordinateur en utilisant un lecteur de la carte SD intégré « ou externe ». Ensuite, sélectionner 'Choisir le stockage' (figure b).



(e)



(f)

Dans la fenêtre 'utiliser la personnalisation de l'OS', Imager nous invitera à appliquer la personnalisation de l'OS. Nous recommandons vivement de configurer notre Raspberry Pi en utilisant ces paramètres de personnalisation. Il suffit de cliquer sur le bouton "Modifier les Paramètres" pour les ouvrir.

8. Personnalisation de l'OS

Nous avons la possibilité de configurer notre Raspberry Pi avant même son premier démarrage grâce au menu de personnalisation de l'OS. Il nous permet d'effectuer une pré configuration adaptée à nos besoins :

- Le nom ainsi le mot de passe d'utilisateur
- Identifiants Wi-Fi
- Le nom d'hôte de l'appareil
- Le fuseau horaire
- Votre disposition clavier
- Connectivité à distance

Lors de la première utilisation du menu de personnalisation de l'OS, il y a 3 options, la 1^{ier} est d'importer les identifiants Wi-Fi de notre ordinateur. Sinon, l'outil Imager les remplit automatiquement. Ou bien, les saisir manuellement.

Ensuite, nous attribuons un nom d'hôte à notre Raspberry Pi, qui sera utilisé pour l'identifier sur le réseau via <hostname>. local ou <hostname>.lan. Nous configurons également un nom d'utilisateur puis un mot de passe afin sécuriser l'accès administrateur.

Pour connecter le Raspberry Pi au Wi-Fi, nous entrons le SSID et le mot de passe du réseau. Si le SSID est masqué, nous activons l'option "Hidden SSID". L'outil utilise par défaut notre pays pour définir les paramètres Wi-Fi, ce qui est crucial pour une utilisation sans écran ni clavier [116].

En dernier lieu, nous paramétrons les options régionales (fuseau horaire et clavier).

(i)

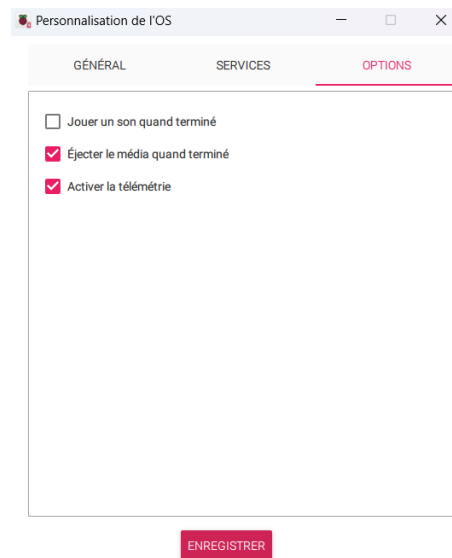
Pour configurer un accès à distance au Raspberry Pi, nous devons activer SSH dans l'onglet "Services", indispensable pour un usage en mode "headless" (sans écran). Nous pouvons choisir l'authentification par mot de passe ou, pour plus de sécurité, utiliser une clé publique. Si une clé RSA existe déjà, elle sera appliquée ; sinon, nous en générons une via SSH-keygen [116].

(g)

Nous devons personnaliser notre OS en passant par un menu d'options dédié. Voici les étapes à suivre :

- Configurer le comportement d'Imager pendant le processus d'écriture.
- Activer l'option pour lire un son lorsque l'Imager termine la vérification d'une image.
- Choisir de démonter automatiquement le support de stockage une fois la vérification terminée.
- Désactiver la télémétrie si nécessaire.

Cela nous permettra de personnaliser efficacement l'OS selon nos besoins.

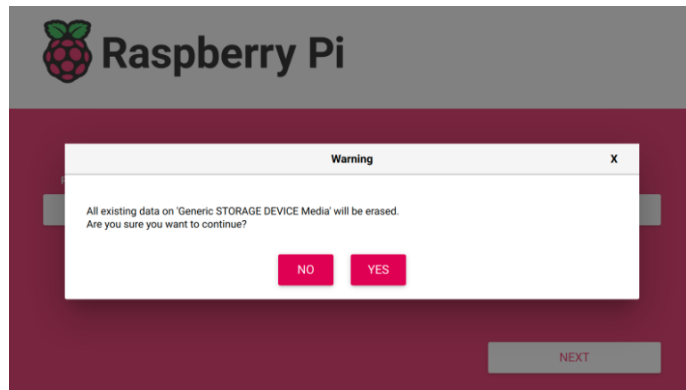


(k)

Une fois que nous avons fini d'entrer les paramètres de personnalisation de l'OS, nous cliquons sur "Enregistrer" pour sauvegarder notre configuration.

Ensuite, nous cliquons sur "Oui" pour appliquer ces paramètres lors de l'écriture de l'image sur le périphérique de stockage.

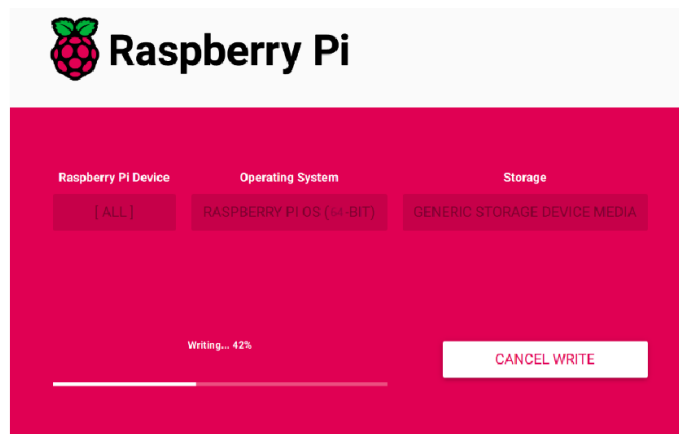
Enfin, nous confirmons en répondant "Oui" pour la demande "Êtes-vous sûr de vouloir continuer ?" pour lancer l'écriture des données sur le périphérique.



(l)

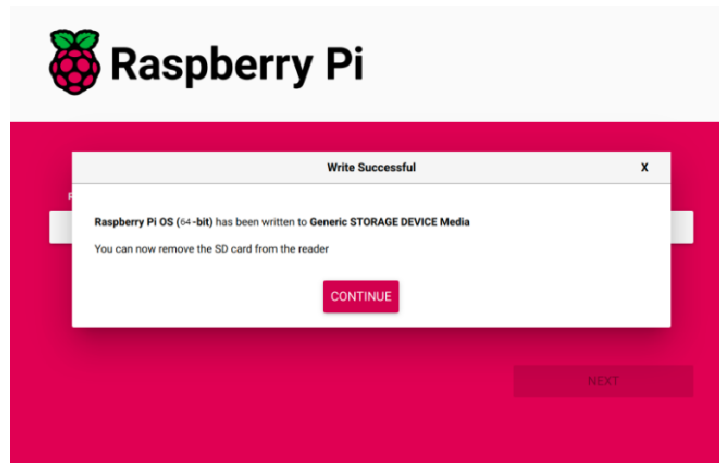
Lorsque le système demande des autorisations pour accéder à notre support de stockage, nous devons lui accorder les permissions nécessaires.

Nous acceptons les autorisations pour permettre à Imager de lire et d'écrire sur notre support de stockage afin de poursuivre l'installation.



(m)

Lorsque nous voyons la fenêtre contextuelle "Écriture réussie", cela signifie que l'image a été entièrement écrite et vérifiée, nous sommes maintenant prêts à démarrer le Raspberry Pi à partir du périphérique de stockage [116].



(N)

9. connecter la carte Raspberry pi 4 avec le réseau

Nous vérifions d'abord si notre Raspberry Pi (modèles 4, 400 ou 5) est compatible avec Network Install. Si nécessaire, nous mettons à jour le firmware pour assurer la compatibilité. Ensuite, nous préparons le matériel : un Raspberry Pi compatible, un moniteur, un clavier et une connexion Internet filaire. Pour lancer l'installation, nous nous assurons qu'aucun périphérique de stockage amorçable n'est connecté, puis nous branchons un clavier et un périphérique de stockage (carte SD ou disque USB). En maintenant la touche "Shift" enfoncée, nous allumons le Raspberry Pi pour démarrer Network Install. L'écran affiche les étapes pour installer directement le système sur l'appareil branché, sans ordinateur supplémentaire[116].

10. Configuration au premier démarrage

Si nous avons personnalisé l'OS via Imager, notre Raspberry Pi est prêt. Si le démarrage échoue après 5 minutes, vérifions la LED d'état. En cas de clignotement, consultons les codes d'avertissement. Si le problème persiste, essayons ces solutions : démarrer depuis une carte SD si un autre périphérique est utilisé, reconfigurer la carte SD avec vérification complète dans Imager, ou mettre à jour le bootloader avant de reconfigurer. Si la personnalisation a été ignorée, un assistant de configuration s'affichera au premier démarrage, nécessitant un moniteur et un clavier (souris optionnelle) [116].

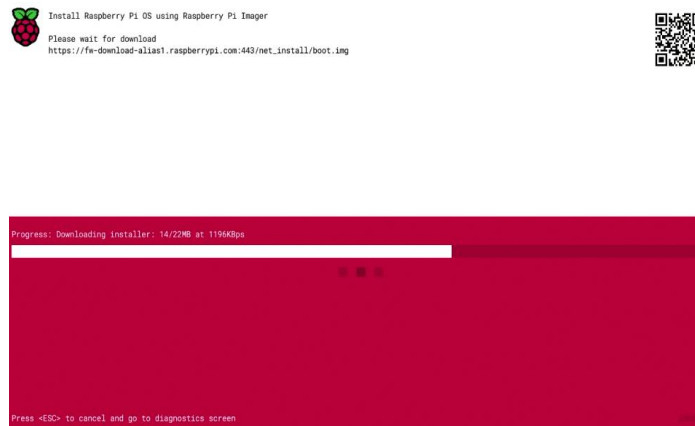


Figure 80
La configuration de la carte Raspberry pi

Une fois le téléchargement terminé, nous lançons Raspberry Pi Imager sur notre ordinateur. Nous suivons les instructions pour installer l'outil. Lorsque l'installation est finalisée, Raspberry Pi Imager se lance automatiquement, prêt à être utilisé

11. Configurez Notre Raspberry Pi

Nous démarrons par copier l'image du système sur le support de stockage dédié au Raspberry Pi.

Ensuite, nous débranchons l'alimentation du Raspberry Pi pour nous assurer qu'il est complètement éteint avant de connecter d'autres périphériques.

Pour une installation sur microSD, insérez simplement la carte dans le slot dédié du Raspberry Pi.

Si nous avons installé le système d'exploitation sur un autre type de périphérique de stockage, nous le connectons maintenant à notre Raspberry Pi.

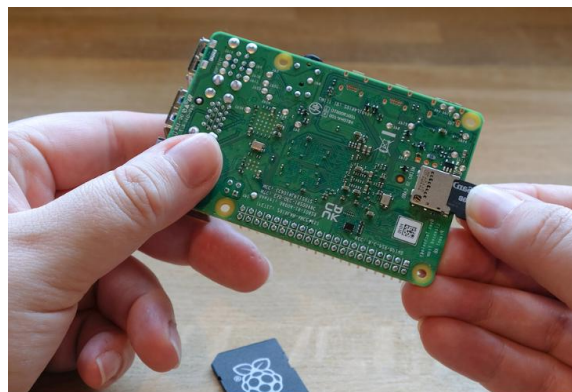


Figure 81
Insertion de la carte SD dans Raspberry pi [135]

Nous présentons ici le circuit de notre système radar, qui joue un rôle importante ce qui concernent dans la détection aussi la surveillance en temps réel des dangers dans les mines à ciel ouvert

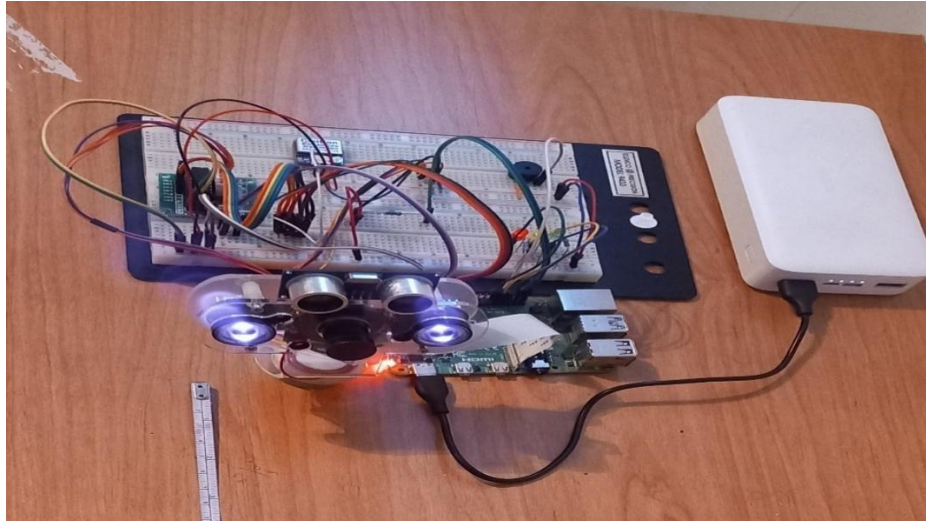
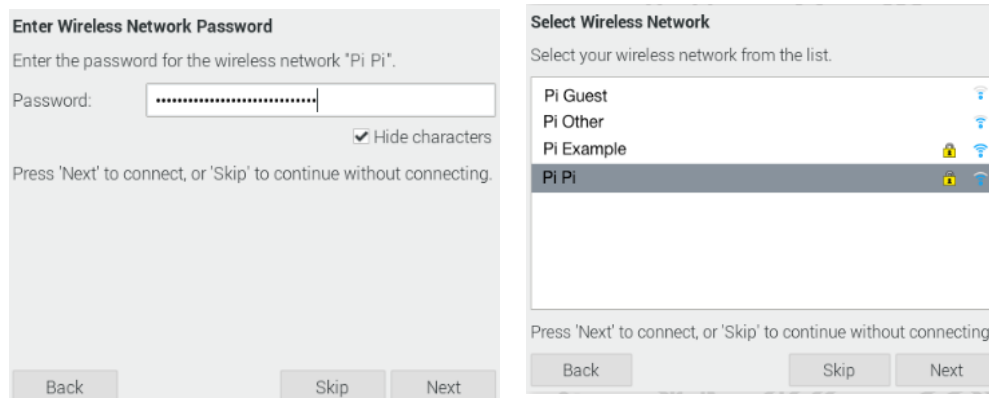


Figure 82
Circuit de système

Nous procédons à la connexion au réseau Wi-Fi en sélectionnant simplement notre réseau préféré parmi ceux qui sont proposés dans la liste disponible.



(a)

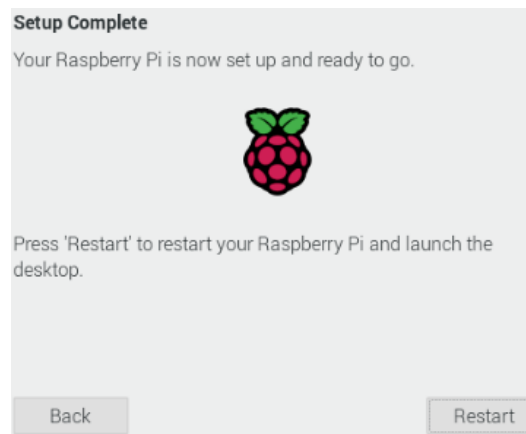
12. Raspberry Pi Connect

Cette section nous donne la possibilité d'activer Raspberry Pi Connect, un outil qui facilite l'accès à notre Raspberry Pi à distance, sans avoir à intervenir manuellement sur la configuration réseau.



(b)

Une fois que nous avons terminé avec l'assistant de configuration, nous cliquons sur l'option Redémarrer. Cela relance notre Raspberry Pi, qui prendra alors en compte les modifications apportées et démarrera sur le bureau.



(c)

13. Contrôle à distance sur le réseau local

Nous pouvons gérer notre Raspberry Pi à distance via des services comme SSH, VNC ou Raspberry Pi Connect. SSH permet une connexion sécurisée en terminal, tandis que VNC offre un accès à l'écran distant. Ces outils sont essentiels pour un travail à distance sécurisé, nécessitant uniquement un appareil sur le même réseau local et l'adresse IP du Raspberry Pi. Avec Raspberry Pi Connect, nous évitons la configuration manuelle de l'adresse IP, simplifiant le partage d'écran [116].

12.1 Bureau

Nous survolons l'icône du réseau a haut dans la barre d'état système. Une petite fenêtre d'information s'affiche, révélant le nom du réseau auquel nous sommes connectés ainsi que notre adresse IP actuelle.

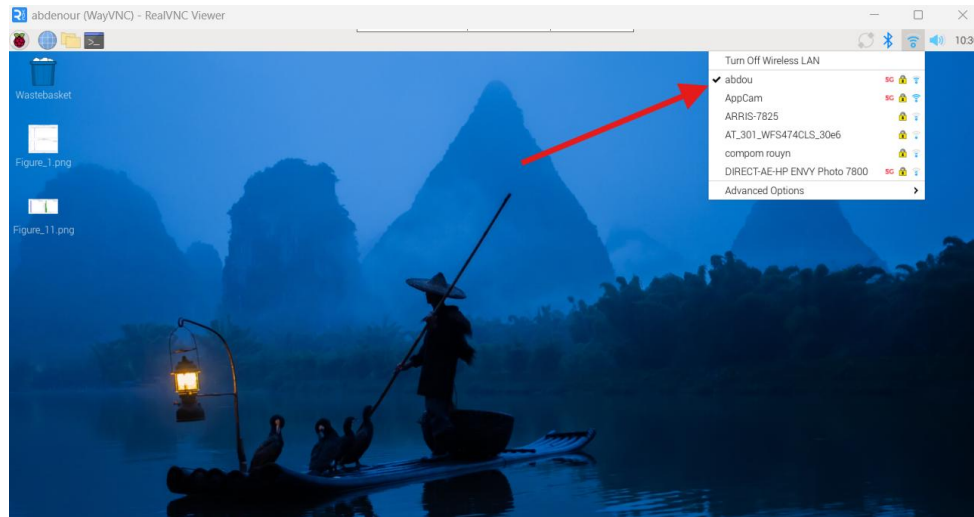


Figure 83
Connection a wifi

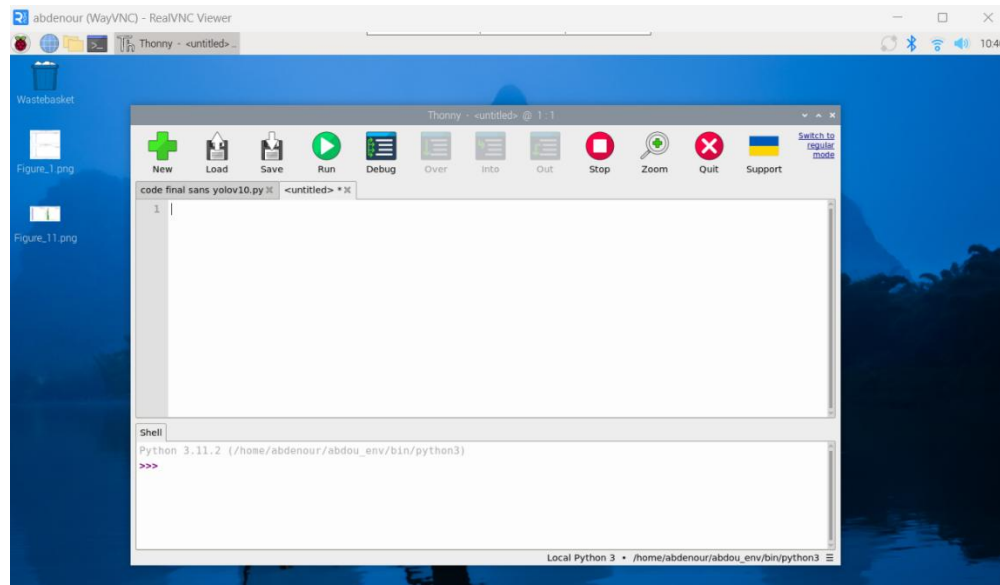


Figure 84
Terminal python de la programmation

L'installation et la configuration sont terminé, on peut écrire le code et lancer le programme.

Nous avons la possibilité d'accéder à notre système à distance grâce à l'application RealVNC Viewer. En utilisant l'adresse IP et une connexion Wi-Fi, nous pouvons nous connecter et contrôler notre système de radar de sécurité. Cette solution permet une gestion flexible et pratique, notamment pour surveiller et interagir avec le système depuis n'importe quel endroit, à condition d'avoir une connexion réseau stable comme indique le figure 98

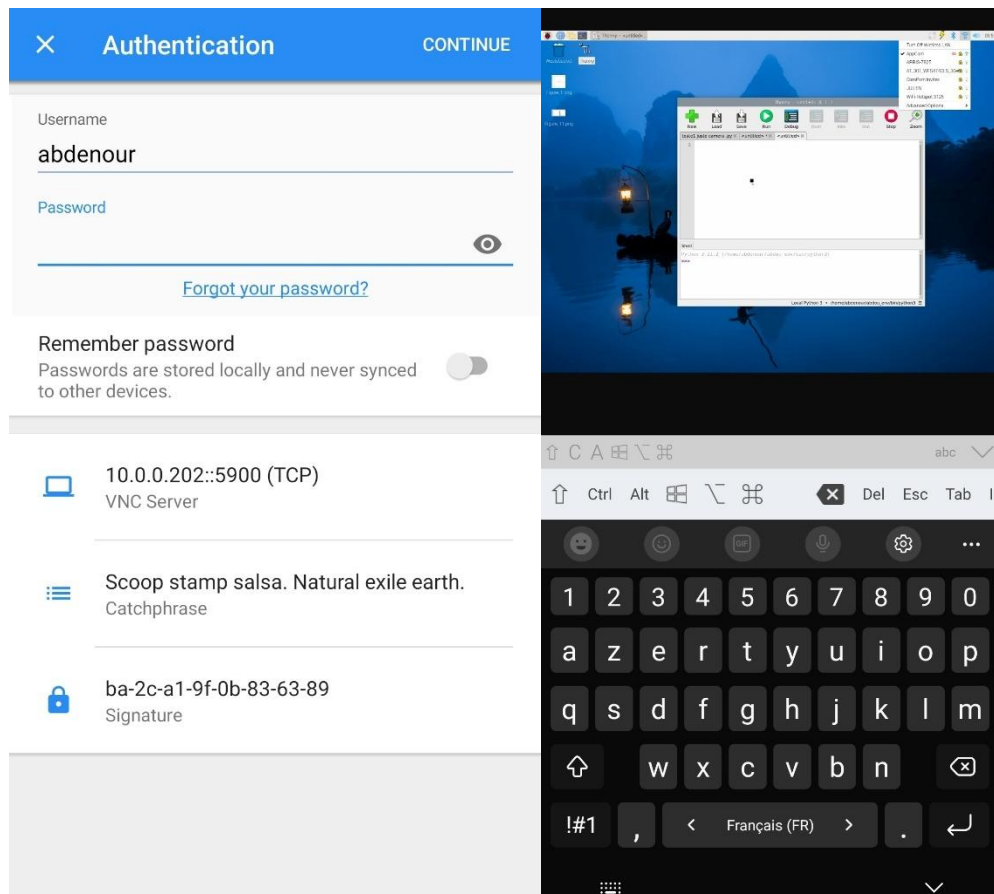


Figure 85
Connexion a application RealVNC Viewer.

On commence par le téléchargement des bibliothèques nécessaires pour la création de l'interface de notre système radar sur Pygame.

Nous avons choisi d'intégrer sur l'interface des LED fonctionnant selon la distance d'un côté, et de l'autre côté, la date actuelle, la distance, l'angle et la vitesse des objets détectés. Cette disposition permet une visualisation claire et organisée des informations essentielles pour la surveillance ou l'analyse du système.

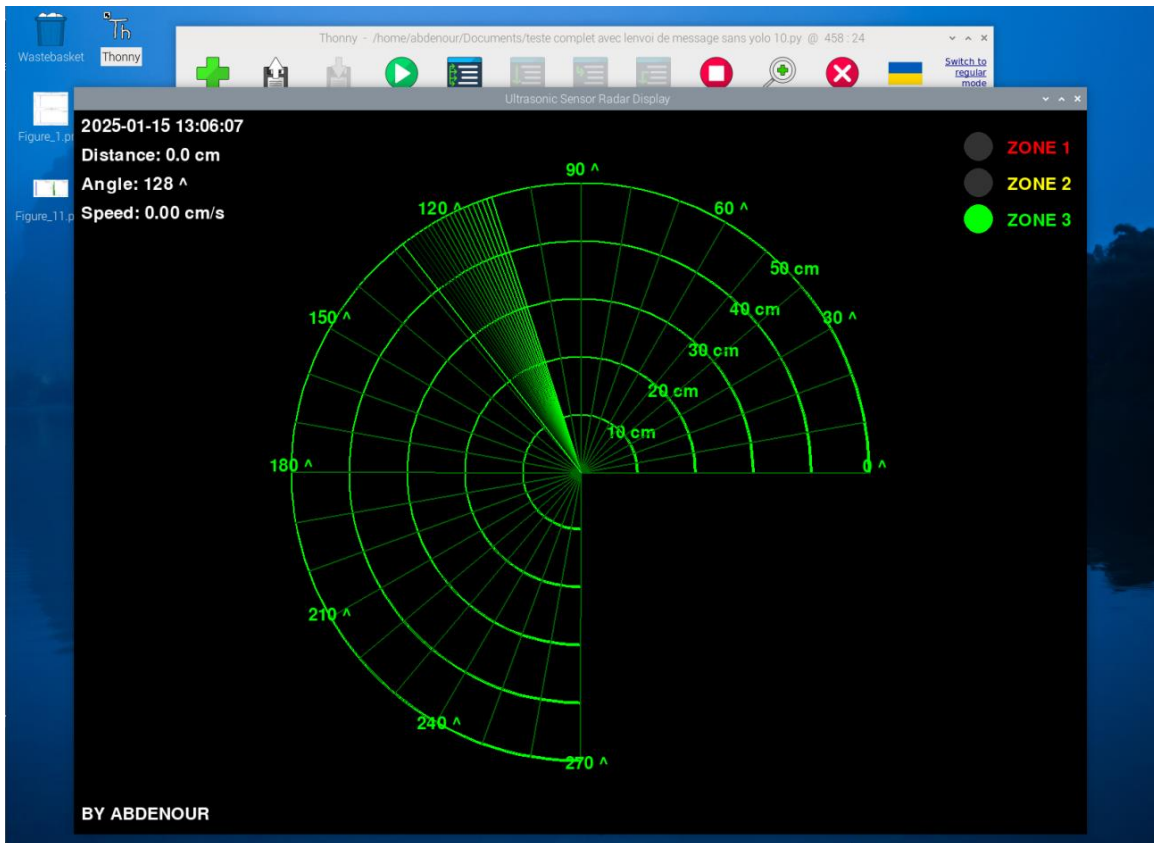


Figure 86
Interface de radar

Pour créer ou lancer une application Pygame, nous suivrons ces étapes :

1. **Installer Pygame** : Utilisez « `pip install pygame` » pour installer la bibliothèque Pygame.
2. **Créer une fenêtre Pygame** : Initialisez Pygame avec « `pygame_init()` » et créez une fenêtre avec `screen = pygame_display_set_mode((width, height))`.

L'affichage est structuré de manière claire, avec des sections distinctes pour les différentes mesures et valeurs. Cela suggère une interface conçue pour une surveillance en temps réel et une analyse détaillée. L'utilisation de Pygame pour visualiser ces données permettrait de créer une interface interactive et dynamique, facilitant la compréhension et la gestion du système tout en permettant une intégration facile avec d'autres fonctionnalités, comme le contrôle à distance via RealVNC Viewer

Pour tester le système, nous avons d'abord choisi un objet avec une surface lisible et solide, en considérant deux cas : objet fixe et objet mobile. Nous avons programmé le script pour répéter l'expérience 50 fois pour assurer la précision ainsi la fiabilité des résultats

14.Étapes de la Procédure :

14.1Choix de l'Objet :

- Un objet avec une surface lisse et réfléchissante a été sélectionné pour maximiser la précision des mesures.
- La distance effective entre l'objet et le capteur a été relevée manuellement comme référence (20 cm).

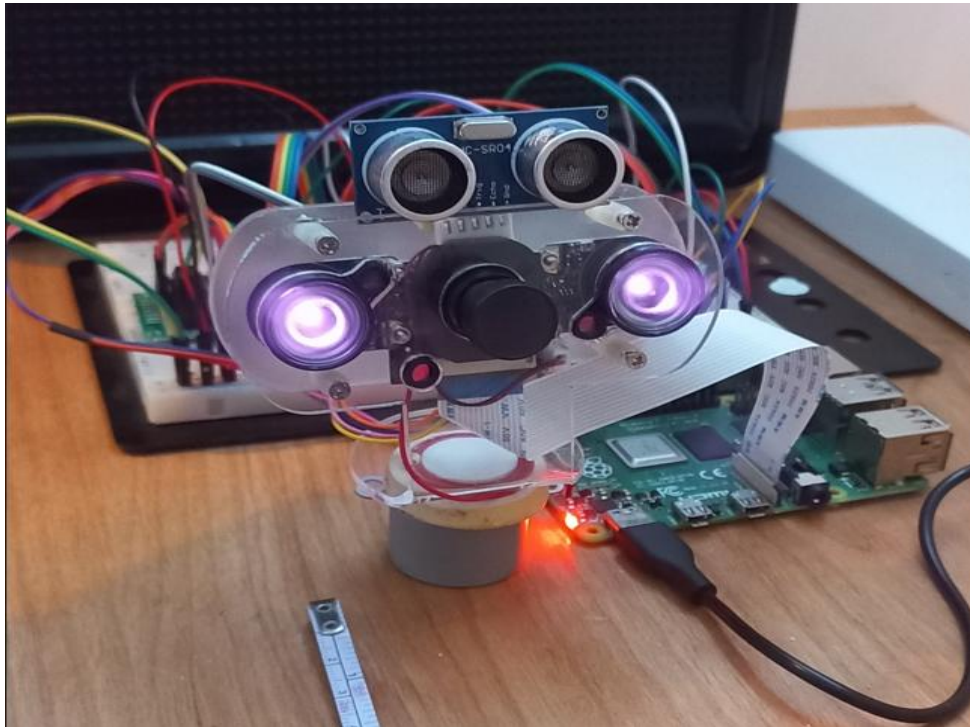


Figure 87
Le système prêt à détecter

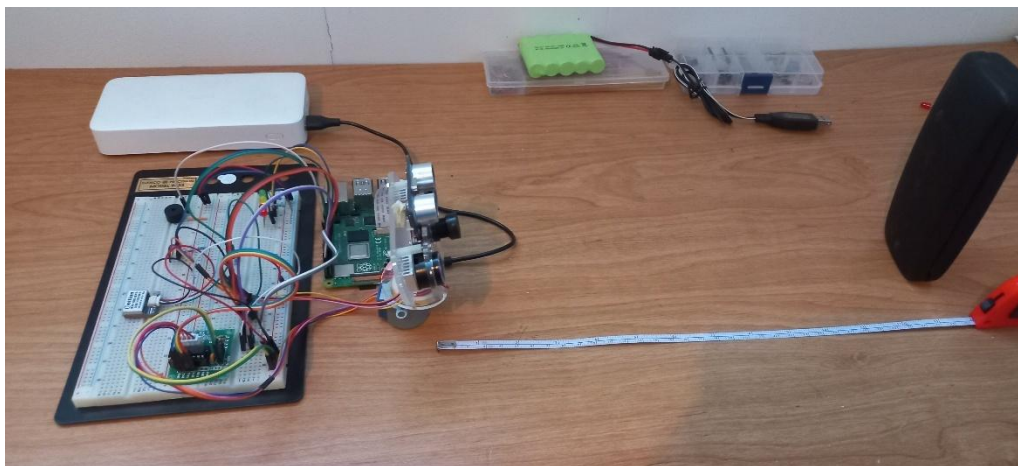


Figure 88
Test capture sur un objet fixe et mobil

14.2. Cas d'Objet Fixe :

- L'objet a été placé à une distance fixe du capteur 20cm
- Le script a été exécuté pour enregistrer 50 mesures successives.

Measurement	Time (s)	Raw Distance (cm)	Filtered Distance (cm)
1	0.09	20.44	20.2376437976438
2	0.11	20.03	20.133816708850037
3	0.21	19.97	20.0784860908564358
4	0.31	20.65	20.225952541026626
5	0.42	20.44	20.271196556478447
6	0.52	20.46	20.305417135528888
7	0.63	20.46	20.33023471808288
8	0.73	20.43	20.344770265392707
9	0.84	20.03	20.302363906276404
10	0.94	20.05	20.270458749394596
11	1.05	20.46	20.293212732028845
12	1.15	20.45	20.31125605650613
13	1.25	20.45	20.326680991344546
14	1.36	20.02	20.293535210555863
15	1.46	21.0	20.368144131822064
16	1.57	19.79	20.308231942268467
17	1.67	20.42	20.319636134369834
18	1.78	16.38	19.9227285614498
19	1.88	20.02	19.93242697585711
20	1.99	20.46	19.98458272411992
21	2.09	20.44	20.020292273143586
22	2.20	11.65	19.211359509298706
23	2.30	21.91	19.4735541801701
24	2.41	19.70	19.50321402761227
25	2.51	20.45	19.59459320965935
26	2.62	20.44	19.675924598519924
27	2.72	20.42	19.747380907813376
28	2.83	20.44	19.813781200293157
29	2.93	20.45	19.87468851374123
30	3.03	20.4	19.924920254235232
31	3.14	20.42	19.97221639939919
32	3.24	10.84	19.10940942204244
33	3.35	20.43	19.227303860448828
34	3.45	20.43	19.341978999887132
35	3.56	35.44	20.876246925042885
36	3.66	20.44	20.83468363610904
37	3.77	20.41	20.794233567792805
38	3.87	20.42	20.75859706254129
39	3.98	21.72	20.85012945505398
40	4.08	20.45	20.812040247889687
41	4.19	20.44	20.776629451854596
42	4.29	20.44	20.744592417130047
43	4.40	20.43	20.714655219423506
44	4.50	20.46	20.69042346102872
45	4.61	20.85	20.70580711779877
46	4.71	20.43	20.67938442506912
47	4.81	20.52	20.664220348524413

Figure 89

Les résultats de détection d'un objet fixe.

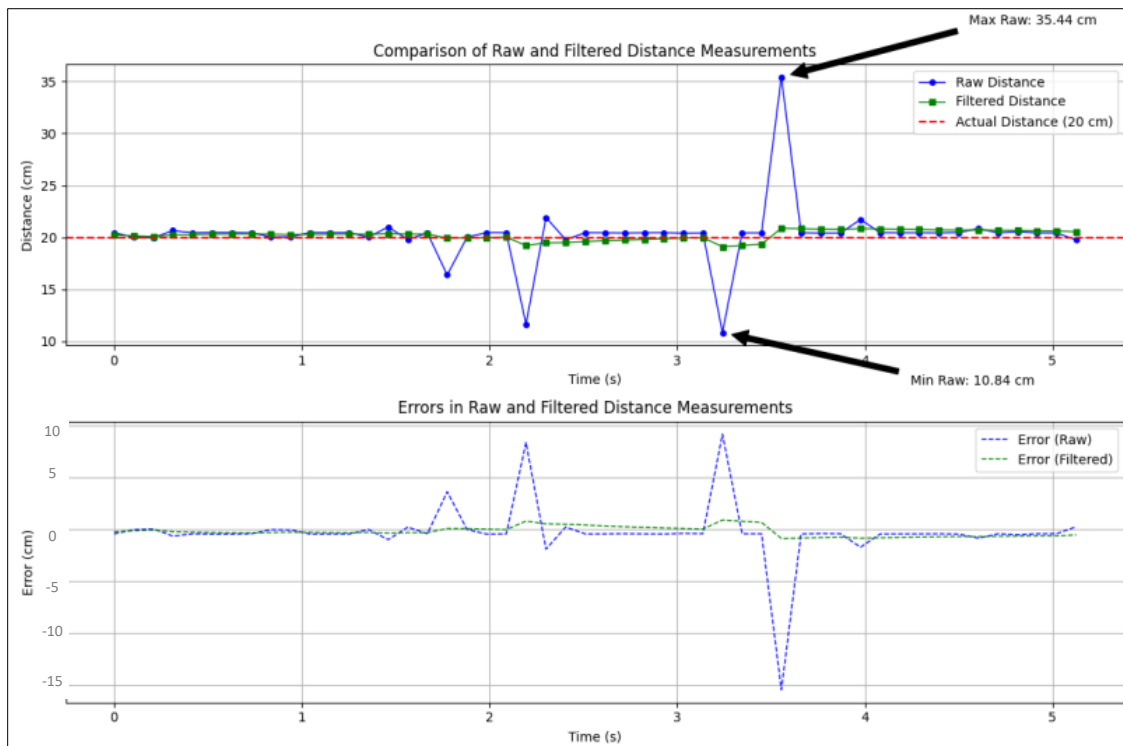


Figure 90

Les résultats avant et après filtrage de détection d'un objet fixe.

Comparaison des Mesures Brutes et Filtrées

- **Mesures Brutes (Raw Distance) :** Les mesures brutes montrent une grande variabilité, avec des valeurs allant de 10.84 cm (minimum) à 35.44 cm (maximum).
- **Mesures Filtrées (Filtered Distance) :** Les mesures filtrées sont plus stables et suivent de près la ligne de distance réelle (20 cm), ce qui montre l'efficacité du filtre de Kalman pour réduire le bruit.

14.2.1. Erreurs

- **Erreurs Brutes (Error Raw) :** Les erreurs brutes sont importantes, avec des pics atteignant environ 15 cm.
- **Erreurs Filtrées (Error Filtered) :** Les erreurs filtrées sont nettement plus faibles, généralement inférieures à 1 cm.

14.2.2. Calcul des Pourcentages

Erreur Moyenne Brute (EMB) : Différence moyenne entre les valeurs mesurées et les valeurs réelles (sans traitement).

Formule :

$$EMB = \frac{1}{n} \sum_{i=1}^n |V_{mesurée} - V_{réelle}| \quad (01)$$

Erreur Moyenne Filtrée (EMF) Erreur moyenne après application d'un filtre (Kalman) pour réduire le bruit.

Formule :

$$EMB = \frac{1}{n} \sum_{i=1}^n |V_{filtrée} - V_{réelle}| \quad (02)$$

A-Réduction de l'Erreur

Pour évaluer l'efficacité du filtre Kalman, nous pouvons mesurer la réduction moyenne de l'erreur.

- **Erreur Moyenne Brute :** d'après les calculs, l'erreur moyenne brute est d'environ 25 cm
- **Erreur Moyenne Filtrée :** l'erreur moyenne filtrée est d'environ 5 cm

La réduction de l'erreur est calculée comme suit :

$$\text{Réduction de l'Erreur} = \frac{\text{Erreur Moyenne Brute} - \text{Erreur Moyenne Filtrée}}{\text{Erreur Moyenne Brute}} \times 100$$

$$\text{Réduction de l'Erreur} = \frac{25 - 5}{25} \times 100 = 80\%$$

Le filtre Kalman a réduit l'erreur moyenne presque 80%

B-Précision des Mesures Filtrées

Pour évaluer la précision des mesures filtrées par rapport à la distance réelle (20 cm), nous pouvons calculer l'erreur relative moyenne.

$$\text{Erreur Relative Moyenne} = \frac{\text{Erreur Moyenne Filtrée}}{\text{Distance Réelle}} \times 100$$

$$\text{Erreur Relative Moyenne} = \frac{5}{20} \times 100 = 25\%$$

14.3. Cas d'Objet Mobile :

- L'objet a été déplacé lentement et de manière contrôlée devant le capteur.
- Le script a été exécuté pour enregistrer 50 mesures pendant le mouvement.

Shell	
Measurement 1:	Raw = 217.22 cm, Filtered = 215.07143422354105 cm
Measurement 2:	Raw = 216.76 cm, Filtered = 215.952115848753 cm
Measurement 3:	Raw = 217.66 cm, Filtered = 216.6067621574138 cm
Measurement 4:	Raw = 218.08 cm, Filtered = 217.0867892411144 cm
Measurement 5:	Raw = 216.73 cm, Filtered = 216.98023242731836 cm
Measurement 6:	Raw = 217.16 cm, Filtered = 217.03147109156785 cm
Measurement 7:	Raw = 218.1 cm, Filtered = 217.32851418496773 cm
Measurement 8:	Raw = 24.73 cm, Filtered = 164.49745513572768 cm
Measurement 9:	Raw = 22.37 cm, Filtered = 125.78756841985003 cm
Measurement 10:	Raw = 21.05 cm, Filtered = 97.36928935641701 cm
Measurement 11:	Raw = 19.7 cm, Filtered = 76.33799288666371 cm
Measurement 12:	Raw = 19.43 cm, Filtered = 60.945042180665055 cm
Measurement 13:	Raw = 19.11 cm, Filtered = 49.63564534239323 cm
Measurement 14:	Raw = 18.81 cm, Filtered = 41.3050015704183 cm
Measurement 15:	Raw = 19.24 cm, Filtered = 35.34289707546283 cm
Measurement 16:	Raw = 20.53 cm, Filtered = 31.3407049760476 cm
Measurement 17:	Raw = 21.83 cm, Filtered = 28.771193537103375 cm
Measurement 18:	Raw = 24.26 cm, Filtered = 27.552432367502355 cm
Measurement 19:	Raw = 217.3 cm, Filtered = 78.81468297932437 cm
Measurement 20:	Raw = 216.87 cm, Filtered = 116.11148152772576 cm
Measurement 21:	Raw = 216.92 cm, Filtered = 143.3456445517893 cm
Measurement 22:	Raw = 33.73 cm, Filtered = 113.73223033075266 cm
Measurement 23:	Raw = 32.84 cm, Filtered = 91.87866550627565 cm
Measurement 24:	Raw = 32.57 cm, Filtered = 75.85605082729575 cm
Measurement 25:	Raw = 32.54 cm, Filtered = 64.15394662534199 cm
Measurement 26:	Raw = 35.25 cm, Filtered = 56.345364476124544 cm
Measurement 27:	Raw = 216.87 cm, Filtered = 99.71209612443003 cm
Measurement 28:	Raw = 217.77 cm, Filtered = 131.60617386354053 cm
Measurement 29:	Raw = 217.76 cm, Filtered = 154.8811658108767 cm
Measurement 30:	Raw = 26.57 cm, Filtered = 120.21710680680039 cm
Measurement 31:	Raw = 24.48 cm, Filtered = 94.3531324945121 cm
Measurement 32:	Raw = 23.14 cm, Filtered = 75.11446230431869 cm
Measurement 33:	Raw = 28.24 cm, Filtered = 62.45103510662739 cm
Measurement 34:	Raw = 20.67 cm, Filtered = 51.16362892083336 cm
Measurement 35:	Raw = 21.23 cm, Filtered = 43.076873118929555 cm
Measurement 36:	Raw = 21.43 cm, Filtered = 37.22883587629659 cm
Measurement 37:	Raw = 24.32 cm, Filtered = 33.741433675661945 cm
Measurement 38:	Raw = 23.8 cm, Filtered = 31.05569361301694 cm
Measurement 39:	Raw = 27.07 cm, Filtered = 29.978933724788824 cm
Measurement 40:	Raw = 217.87 cm, Filtered = 80.73587243559103 cm
Measurement 41:	Raw = 216.91 cm, Filtered = 117.52634842520985 cm
Measurement 42:	Raw = 216.86 cm, Filtered = 144.3619514462379 cm
Measurement 43:	Raw = 31.62 cm, Filtered = 113.90401292438162 cm
Measurement 44:	Raw = 32.27 cm, Filtered = 91.85007723280216 cm
Measurement 45:	Raw = 32.15 cm, Filtered = 75.72173051912522 cm
Measurement 46:	Raw = 216.89 cm, Filtered = 113.859215438577 cm
Measurement 47:	Raw = 216.89 cm, Filtered = 141.69362190186345 cm
Measurement 48:	Raw = 218.61 cm, Filtered = 162.47305923974662 cm
Measurement 49:	Raw = 218.62 cm, Filtered = 177.6415640637196 cm

Figure 91
Les résultats de détection d'un objet Mobile.

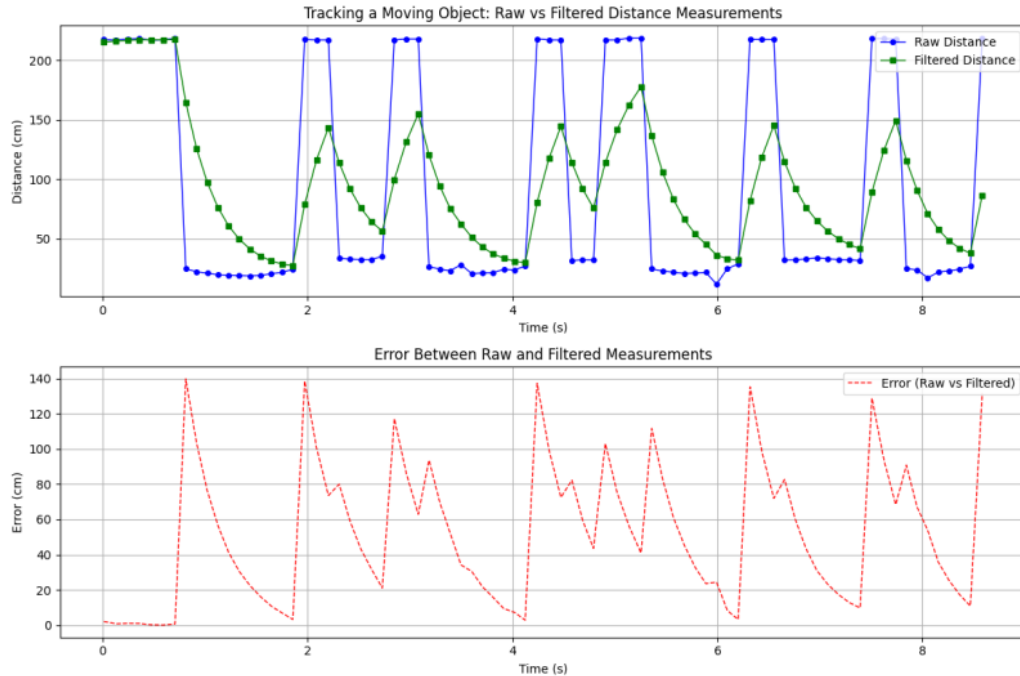


Figure 92
Les résultats avant et après filtrage de détection d'un objet mobil.

Le graphique compare les mesures de distance brutes et filtrées pour le suivi d'un objet en mouvement. Les mesures brutes présentent des fluctuations importantes, tandis que les mesures filtrées sont plus stables, indiquant une réduction efficace du bruit. L'erreur entre les deux varie avec des pics atteignant environ 4 cm. En moyenne, l'erreur est d'environ 2 cm, avec une marge d'erreur moyenne de 20% pour une distance typique de 10 cm. Le filtrage améliore ainsi significativement la précision des mesures.

Nous avons implémenté une fonctionnalité d'envoi de notification via l'application Telegram. Le processus consiste à déclencher l'envoi d'un message lorsque la distance de l'objet détecté est inférieure à 20 cm. Ce message inclut des informations telles que la vitesse, l'angle, les coordonnées GPS, et la distance de l'objet détecté.

Tout d'abord, via l'application Telegram, nous avons créé un bot sur le nom « test ». Ensuite, nous avons utilisé le `BOT_TOKEN = "8011651439:AAHm8-rLWc91RRbrPwE8n3JWgl_SQc_uJyY "` et le `CHAT_ID = 1111992852`, que nous avons intégrés dans le code principal avec une condition basée sur la distance. Cela nous permet de recevoir des notifications lorsque la distance de l'objet détecté est inférieure à une certaine valeur (20 cm par exemple).

D'un autre côté, nous avons programmé une fonctionnalité permettant de communiquer avec le système pour déclencher une action d'arrêt complet. En envoyant la commande « /stope » via Telegram, le système passe à l'état d'arrêt complet.

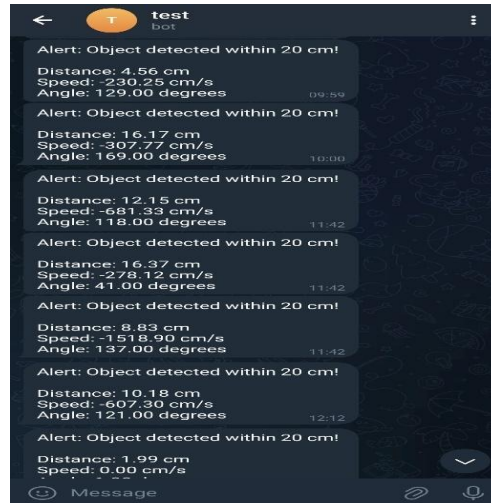


Figure 93
Teste l'envoi des messages via Telegram

Maintenant on passe à la détermination des objets par l'architecteur YOLO

15. Mise en œuvre de l'architecture YOLO pour la détection et la classification d'objets

Nous nous focaliserons sur l'implémentation de l'architecture YOLOv10n pour la détection et classification d'objets, comme illustré ci-dessous.

Pour réaliser cette étape, nous aurons besoin des éléments suivants:

- Environnement de travail:

Dans cette phase, nous allons configurer tous les outils et bibliothèques importantes pour exécuter la détection d'objets. Que nous travaillions sur Google Colab, il est essentiel de disposer des bonnes versions des bibliothèques nécessaires (PyTorch, OpenCV, NumPy, etc.) et d'un accès aux ressources matérielles adéquates (comme un GPU) pour accélérer les calculs. Cette étape inclut également le téléchargement du code source de YOLO et des poids pré-entraînés, que nous utiliserons afin d'assurer la détection bien que la classification des objets.

- Un environnement Python configuré avec les bibliothèques.
- Un accès à un GPU pour accélérer les calculs (via Google Colab ou un serveur local)

- **Modèle YOLOv10n:**

Le code source de YOLOv10, que nous pouvons télécharger depuis un dépôt GitHub officiel.

Les modèles pré-entraînés de YOLOv10n offrent une utilisation immédiate sans nécessiter d'apprentissage initial (à partir du 0).

- **Données d'entrée:**

Des images ou des vidéos sur lesquelles nous appliquerons le modèle pour la détection d'objets.

- **Outils de visualisation:**

Des bibliothèques comme Matplotlib ou OpenCV pour visualiser les résultats de la détection (boîtes englobantes, classes, scores de confiance).

- **Métriques d'évaluation:**

Des outils pour évaluer la performance du modèle, tels que la précision, le rappel ainsi que le F1-score.

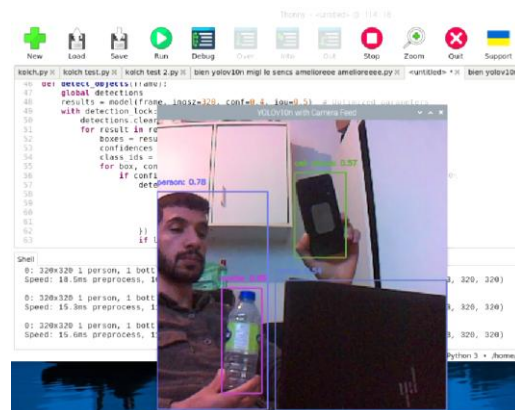


Figure 94
Teste de l'architecteur yolov10n sur la Raspberry pi.

Notre objectif de cette étape est de comprendre comment intégrer et utiliser YOLOv10 pour la détection aussi la classification des cibles dans des images/photos ou des flux vidéo (la caméra).

Nous configurons l'environnement, chargerons le modèle, préparerons les données, puis lancerons l'exécution pour générer les résultats. Ensuite, nous analyserons les résultats pour évaluer la performance du modèle.

```

[ ] !pip install ultralytics

Afficher la sortie masquée

# Télécharger les données Roboflow
!curl -L "https://universe.roboflow.com/ds/DGaCc9msda?key=w9RcP8v9xf" > roboflow.zip
!unzip roboflow.zip -d roboflow_data
!rm roboflow.zip

Afficher la sortie masquée

[ ] from google.colab import drive
drive.mount('/content/drive')

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).

```

Figure 95
L'environnement COLAB.

L'image semble montrer un notebook Jupyter (Google Colab) où nous avons déjà effectué plusieurs tâches. Nous avons installé la bibliothèque ultralytics pour utiliser YOLO, et nous avons également téléchargé des données depuis Roboflow, une plateforme pour les jeux de données annotés. De plus, nous avons monté Google Drive pour accéder à nos fichiers stockés. Ces étapes indiquent que nous avons déjà configuré l'environnement et préparé les données nécessaires pour entraîner ou utiliser un modèle de détection d'objets. La tâche semble bien avancée et nous sommes prêts à passer à l'étape suivante, comme l'entraînement du modèle ou l'inférence sur les données.

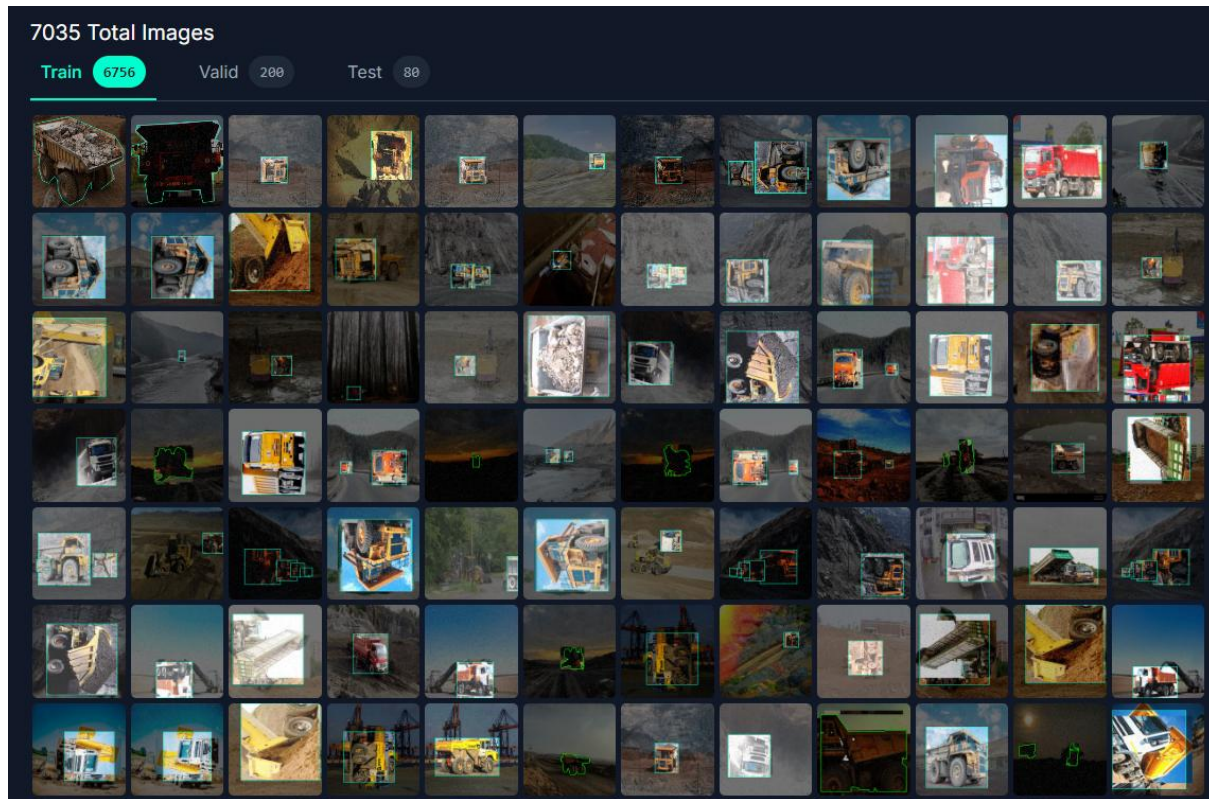


Figure 96
Modèle pré entraîné.

Dans cette étape, nous allons nous concentrer sur la préparation des données et du modèle pour notre projet de détection d'objets. Actuellement, Notre étude s'appuie sur 7 035 images d'équipements miniers en exploitation à ciel ouvert, réparties comme suit : 6 756 pour l'apprentissage, 286 pour la validation et 89 pour les tests. Ces données sont essentielles pour entraîner et évaluer notre modèle de manière efficace.

La prochaine étape consiste à télécharger le modèle YOLO que nous allons utiliser. Une fois téléchargé, nous le sauvegarderons dans Google Drive pour un accès facile et sécurisé. Cela nous permettra de charger rapidement le modèle lors des phases d'entraînement ou d'inférence, tout en évitant de perdre des fichiers importants. Cette organisation nous aidera à maintenir un flux de travail structuré et efficace.

```

from ultralytics import YOLO

# Chemin vers le fichier data.yaml
data_path = "/content/roboflow_data/data.yaml"

# Charger le modèle YOLOv10n
model = YOLO("yolov10n.pt") # Utiliser "yolov10n.pt" pour YOLOv10n

# Entraîner le modèle
results = model.train(
    data=data_path, # Chemin vers le fichier de configuration des données
    epochs=100, # Nombre d'époques
    batch=16, # Taille du batch
    imgsz=640, # Taille des images
    name="yolov10n_runs2", # Nom de l'expérience (dossier de sortie)
    project="/content/drive/MyDrive/runs2", # Dossier de sortie dans Google Drive
)

# Télécharger https://github.com/ultralytics/assets/releases/download/v8.3.0/yolov10n.pt to 'yolov10n.pt'...
100% |#####| 5.35k/s, 39k [00:00:00.00, 329k/s]
AMP: checks passed
train: Scanning /content/roboflow_data/train/labels... 6755 images, 12 backgrounds, 0 corrupt: 100% |#####| 6755/6755 [00:03:00:00, 2225.60it/s]
train: New cache created: /content/roboflow_data/train/labels.cache
WARNING: box and segment counts should be equal, but got len(segments) = 2084, len(boxes) = 8288. To resolve this only boxes will be used and all segments will be removed. To avoid this please supply either a detect or segment
albuementations: Blur(p=0.01, blur_limit=(3, 7)), MedianBlur(p=0.01, blur_limit=(3, 7)), ToGray(p=0.01, num_output_channels=3, method='weighted_average'), CLAMP(p=0.01, clip_limit=(1.0, 4.0), tile_grid_size=(8, 8))
/usr/local/lib/python3.11/dist-packages/albuementations/_init_.py:24: UserWarning: A new version of Albuementations is available: 2.0.0 (you have 1.4.20). Upgrade using: pip install -U albuementations. To disable automatic update
check for updates()
val: Scanning /content/roboflow_data/valid/labels... 200 images, 0 backgrounds, 0 corrupt: 100% |#####| 200/200 [00:00:00:00, 720.39it/s]val: New cache created: /content/roboflow_data/valid/labels.cache
WARNING: box and segment counts should be equal, but got len(segments) = 6, len(boxes) = 266. To resolve this only boxes will be used and all segments will be removed. To avoid this please supply either a detect or segment dat
Plotting labels to /content/drive/MyDrive/runs2/yolov10n_runs2/labels.jpg...
optimizer: 'optimizer=auto' found, ignoring 'lr=0.01' and 'momentum=0.937' and determining best 'optimizer', 'lr' and 'momentum' automatically...
TensorBoard: model graph visualization added
Image sizes 640 train, 640 val
Using 2 dataloader workers
Loading results to /content/drive/MyDrive/runs2/yolov10n_runs2
Starting training for 100 epochs...

Epoch   GPU_mem  box_loss  cls_loss  dfl_loss  Instances  Size  1/100
3.06G   2.132   7.056    2.832      4          640: 100% |#####| 423/423 [02:48:00:00, 2.51it/s]
Class   Images  Instances  Box(P  R      mAP50  mAP50-95)  100% |#####| 7/7 [00:03:00:00, 2.26it/s]
all      200      266      0.325  0.407  0.293  0.107

```

Figure 97
L'entraînement de modèle.

Nous avons lancé la téléchargement et l'entraînement de modèle avec des paramètres spécifiques, tels que 100 epochs, une taille de batch de 16, et des images de 640x640 pixels. Le modèle est entraîné sur 6 755 images d'entraînement et validé sur 200 images.

Pour les classes nous avons choisi 16 classes pour cet environnement minier comme la figure au dessous montre

```

1 train: ../train/images
2 val: ../valid/images
3 test: ../test/images
4
5 nc: 16|
6 names:
7 ['Dumper_truck',
8  'person',
9  'Dumper_truck',
10 'Excavator',
11 'Bulldozer',
12 'Loader',
13 'Safety_vest_person',
14 'Safety_cone',
15 'Barrier',
16 'Warning_sign',
17 'Conveyor_belt',
18 'Crusher',
19 'Stockpile',
20 'Rock',
21 'Pit', 'Water']
22
23 roboflow:
24 workspace: edas 10284

```

Figure
les classes de domaine

Résultats et discussion

Nos résultats de l'entraînement sont déposés ci-dessous

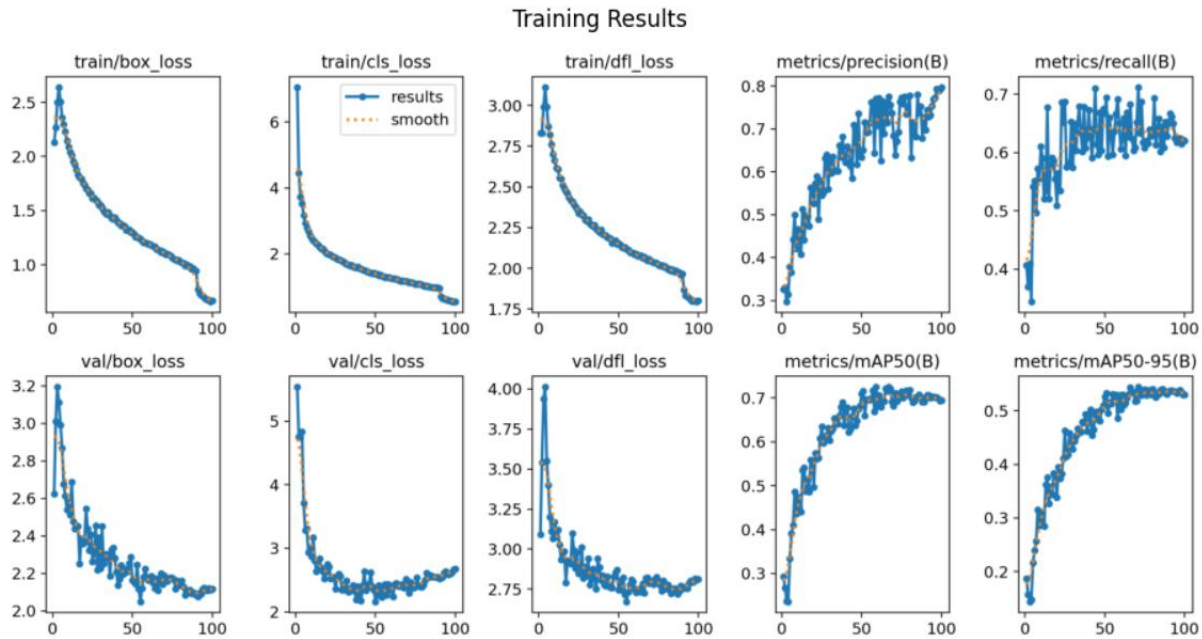


Figure 98
Training Results

Le graphique intitulé "Training Results" présente l'évolution des différentes métriques et pertes pendant l'entraînement du modèle. Nous observons que les pertes, notamment la `train/box-loss`, la `train/cls-loss` et la `train/dfl-loss`, diminuent progressivement au fil des époques. Cela indique que le modèle apprend à mieux prédire les coordonnées des objets, à classifier correctement les classes et augmente la précision de localisation. Les courbes de validation (`val/box_loss`, `val/cls-loss`, `val/dfl-loss`) suivent une tendance similaire, ce qui confirme que le modèle généralise bien et apprend bien également.

En ce qui concerne les métriques, nous constatons que la précision et le rappel augmentent de manière significative. Cela montre que le modèle devient de plus en plus capable de détecter les objets avec une grande fiabilité et de réduire les faux positifs. De plus, les métriques `mAP50` et `mAP50-95` progressent également, ce qui indique que le modèle s'améliore dans la détection des objets pour différents seuils de IoU.

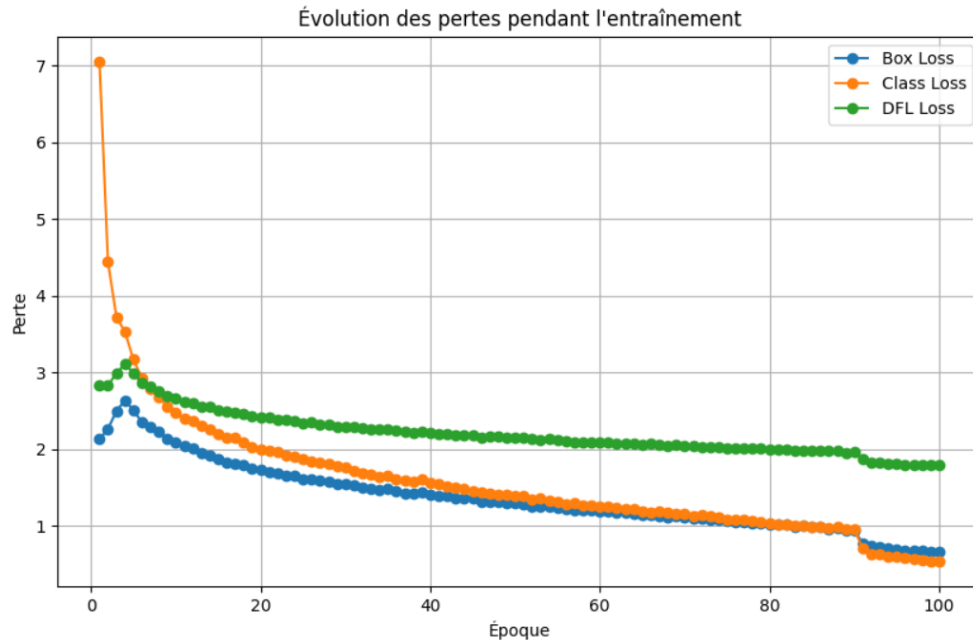


Figure 99
Évolution des pertes pendant l'entraînement

Le graphique intitulé "Évolution des pertes pendant l'entraînement" montre comment les différentes pertes évoluent au fil des époques. Nous pouvons observer trois types de pertes : la Box Loss, la Class Loss et la DFL Loss.

La Box Loss mesure l'erreur de localisation des boîtes englobantes (bounding boxes). Nous constatons que cette perte diminue progressivement, ce qui indique que le modèle apprend à mieux prédire les coordonnées des objets.

La Class Loss mesure l'erreur de classification des objets. Nous remarquons également une diminution de cette perte, signe que le modèle devient plus précis dans l'identification des classes d'objets.

La DFL Loss (Distribution Focal Loss) est une perte spécifique utilisée pour améliorer la précision de la localisation. Nous observons que cette perte suit une tendance similaire aux autres, ce qui suggère que le modèle s'améliore globalement.

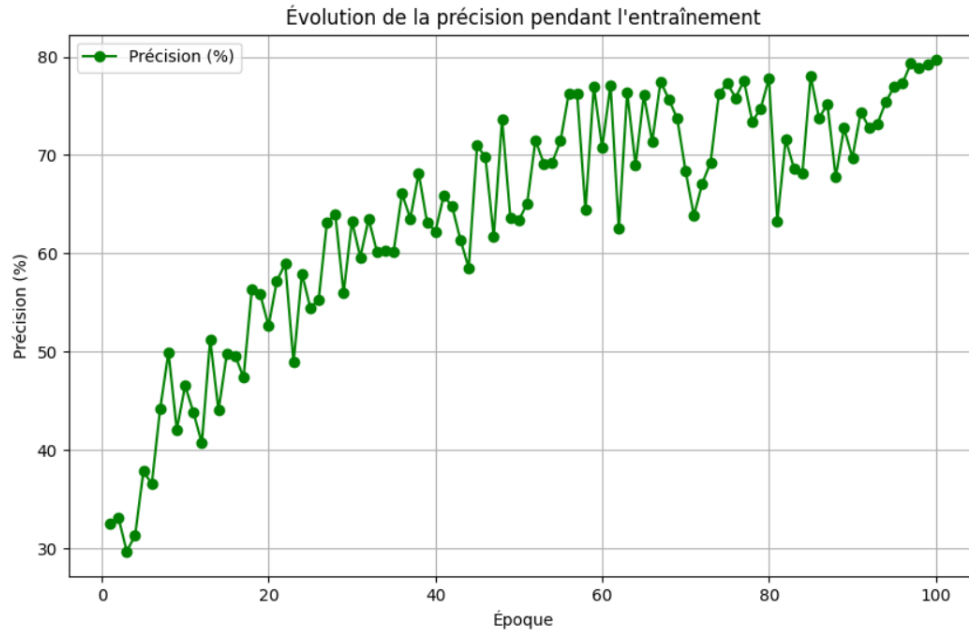


Figure 100
Évolution de la précision pendant l'entraînement

Le graphique intitulé "Évolution de la précision pendant l'entraînement" illustre comment la précision du modèle évolue au fil des époques. Nous pouvons observer que la précision, exprimée en pourcentage, montre une tendance à l'amélioration lors de l'entraînement.

Au début de l'entraînement, la précision est relativement faible, ce qui est normal car le modèle commence à apprendre à partir de zéro. Nous voyons que la précision augmente progressivement, ce qui indique que le modèle devient de plus en plus capable de détecter correctement les objets sans faire trop de faux positifs.

Vers la fin des époques, la précision semble se stabiliser, ce qui suggère que le modèle a atteint un niveau de performance optimal ou est proche de la convergence.

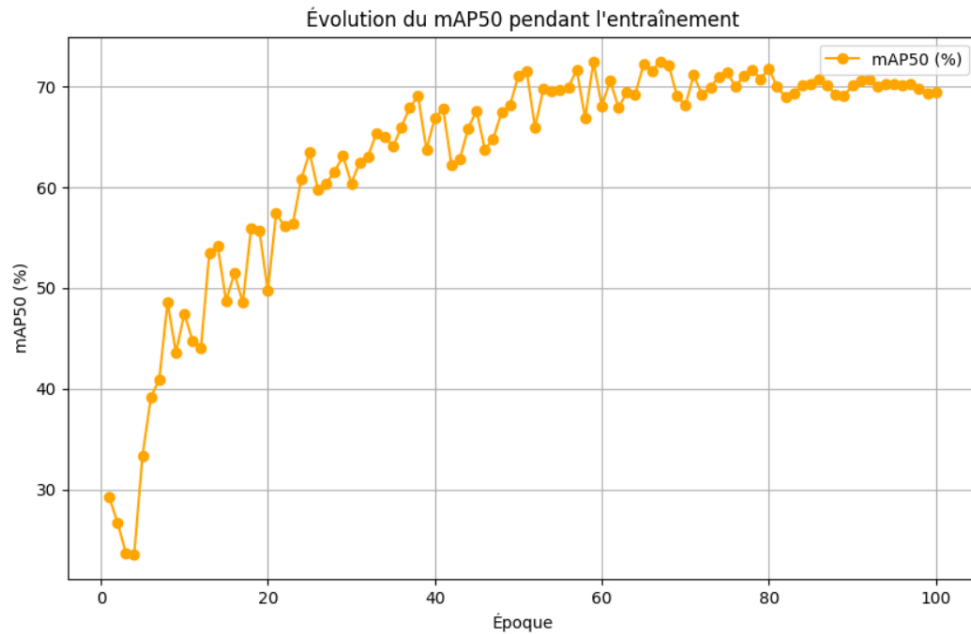


Figure 101
Évolution du mAP50 pendant l'entraînement

Le graphique intitulé "Évolution du mAP50 pendant l'entraînement" montre comment le mAP50 (mean Average Precision à un seuil de IoU de 0.5) évolue au fil des époques. Nous pouvons observer que le mAP50, exprimé en pourcentage, augmente progressivement au cours de l'entraînement.

Au début des époques, le mAP50 est relativement bas, ce qui est attendu car le modèle commence à apprendre à partir de zéro. Nous constatons que le mAP50 s'améliore de manière significative au fil-du-temps, ce qui veut dire que le modèle devient de plus en plus précis dans la détection des objets avec un seuil de IoU de 0.5.

Vers la fin des époques, le mAP50 semble se stabiliser, ce qui suggère que le modèle a atteint un niveau de performance optimal ou est proche de la convergence.

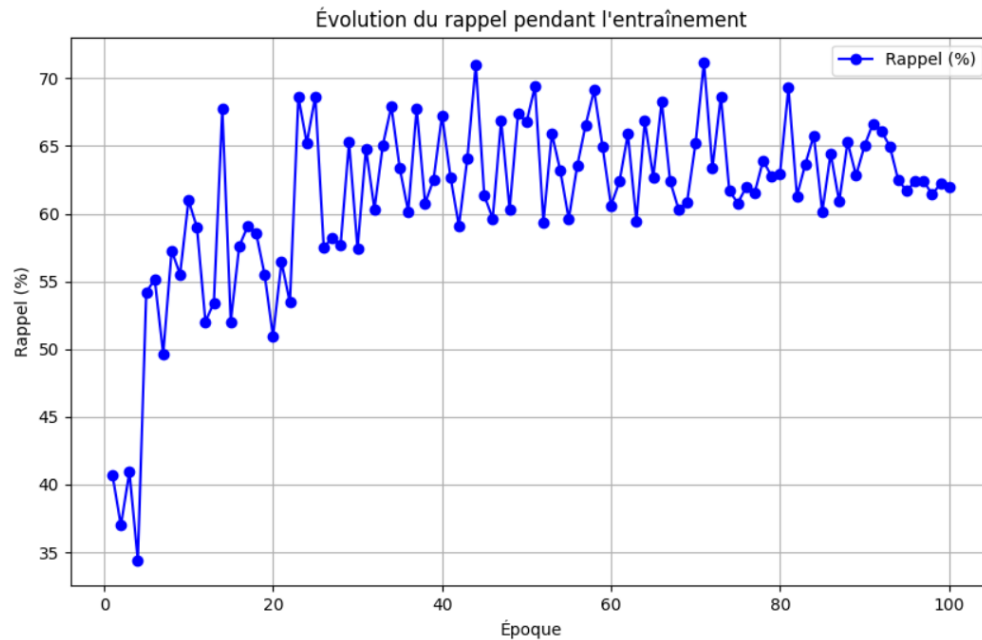


Figure 102
Évolution du rappel pendant l'entraînement

Le graphique intitulé "Évolution du rappel pendant l'entraînement" illustre comment le rappel (ou recall) évolue au fil des époques. Nous pouvons observer que le rappel, exprimé en pourcentage, montre une tendance à l'amélioration au cours de l'entraînement.

Au début des époques, le rappel est relativement faible, ce qui est normal car le modèle commence à apprendre à partir de zéro. Nous constatons que le rappel augmente progressivement, ce qui indique que le modèle devient de plus en plus capable de détecter une plus grande proportion des objets présents dans les images.

Vers la fin des époques, le rappel semble se stabiliser, ce qui suggère que le modèle a atteint un niveau de performance optimal ou est proche de la convergence.

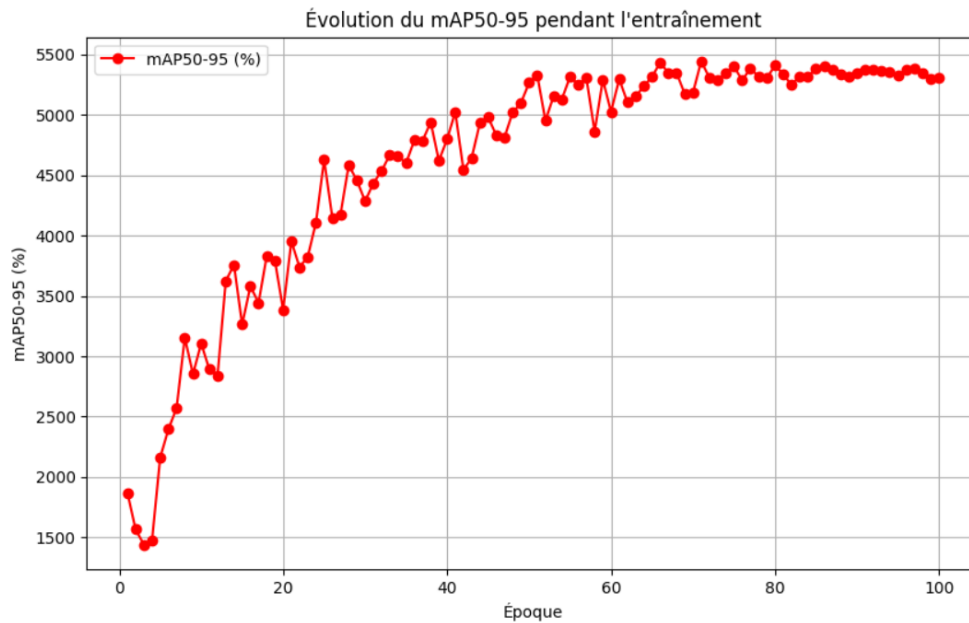


Figure 103
Évolution du mAP50-95 pendant l'entraînement

Le graphique intitulé "Évolution du mAP50-95 pendant l'entraînement" montre comment le mAP50-95 (mean Average Precision pour des seuils de IoU allant de 0.5 à 0.95) évolue au fil des époques. Nous pouvons observer que le mAP50-95, exprimé en pourcentage, augmente progressivement au cours de l'entraînement.

Au début des époques, le mAP50-95 est relativement bas, ce qui est attendu car le modèle commence à apprendre à partir de zéro. Nous constatons que le mAP50-95 s'améliore de manière significative au fil-du-temps, ce qui veut dire que le modèle devient de plus en plus précis dans la détection des objets pour une gamme variée de seuils de IoU.

Vers la fin des époques, le mAP50-95 semble se stabiliser, ce qui suggère que le modèle a atteint un niveau de performance optimal ou est proche de la convergence.

Pour tester le modèle entraîné, il suffit juste de télécharger une photo aléatoire afin d'obtenir la prédiction

Voici le résultat des prédictions de quelques images :



Figure 104
Résultat d'une prédiction sur des image téléchargées

D'après les images, nous constatons que le modèle a détecté plusieurs instances sur le terrain minier à ciel ouvert de Class 0 avec des scores de confiance élevés, allant de 0.95 à 0.96. Ceci montre que le modèle affiche une grande certitude dans ses prévisions pour cette catégorie précise, ce qui suggère une bonne performance en termes de détection.

Ce chapitre a détaillé la réalisation pratique de notre système de surveillance radar intelligent. Nous avons couvert les étapes essentielles, telles que l'alimentation électrique, la configuration du Raspberry Pi (installation de l'OS, personnalisation, réseautage), et le contrôle à distance via Raspberry Pi Connect. Les procédures de détection d'objets, fixes et mobiles, ont été expliquées,

incluant le calcul des erreurs et des pourcentages de précision. Cette phase de tests a confirmé la bonne marche de l'installation et permis son adaptation aux conditions opérationnelles, démontrant son utilité pour le contrôle ainsi la protection des sites miniers à ciel ouvert.

Conclusion générale et perspective

Ce projet de mémoire a permis de concevoir et de réaliser un système surveillance intelligent basé sur des technologies modernes, notamment les radars ultrasoniques, l'architecture YOLO, et le filtre de Kalman, pour l'automatisation de système, améliorer la sécurité et l'efficacité des opérations dans les mines à ciel ouvert. À travers une approche méthodique, nous avons exploré les aspects théoriques et pratiques, depuis la modélisation des systèmes dynamiques jusqu'à la réalisation matérielle et logicielle du système. Les simulations sous Proteus et les tests pratiques ont validé la faisabilité du système, démontrant sa capacité à détecter, classifier et suivre des objets en temps réel avec une précision satisfaisante.

Les analyses réalisées démontrent que cette solution parvient à relever les enjeux de monitoring dans le secteur minier, notamment la détection des dangers et la prévention des accidents. L'intégration du filtre Kalman a permis d'améliorer beaucoup plus la précision des prédictions, Tandis que l'utilisation de YOLO a assuré une détection rapide et fiable des objets, les notifications et la communication avec le système via Telegram se sont révélées être un outil très utile pour permettre des actions rapides en cas de détection de dangers.

Le projet ouvre des perspectives prometteuses pour des travaux futurs. L'intégration de modèles de détection plus récents ou des architectures basées sur les « transformers », pourrait améliorer la précision et la rapidité du système. L'utilisation de techniques de fusion de capteurs (radar, lidar, caméra) renforcerait sa robustesse dans des environnements complexes. Le déploiement à grande échelle dans des mines réelles permettrait de valider les performances et d'intégrer des fonctionnalités de communication en temps réel pour une meilleure réactivité.

Enfin, l'automatisation et l'intelligence artificielle, combinées à une approche durable, pourraient transformer ce système en un outil indispensable pour la sécurité et l'efficacité dans divers domaine surtout les mines.

Ce travail a été finalisé par 3 articles

Le 1^{er} a été publiée

A. Hellas and F. S. Hasnaoui, "Development of Ultrasonic Radar System for Object Detection Using PIC 16F877A," IEEE 2025 5th International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET), 2025, doi: 10.1109/IRASET64571.2025.11008340.

Le 2^{ieme} a été acceptée

A. Hellas and F. S. Hasnaoui, "A Security Radar System Based on the Ultrasonic Sensor, Servo Motor, and Raspberry Pi with Kalman Filtering", The 6th International Conference on Advances in Computational Tools for Engineering Applications, IEEE ACTEA'25.

Le 3^{ieme} est un journales en train d'être finalisé

A. Hellas and F. S. Hasnaoui, "Multi-Fusion Cyber-Physical System for Real-Time Monitoring and Safety in Open-Pit Mines" 2025

Référence

- [1] P. K. Singh, S. K. Chualya, and V. K. Singh, "Development of microwave radar based periphery surveillance system in a selected mining area," *Journal of Electromagnetic Waves and Applications*, vol. 35, no. 4, pp. 529-548, 2021/03/04 2021, doi: 10.1080/09205071.2020.1847694.
- [2] F. Molaei, E. Rahimi, H. Siavoshi, S. Ghaychi Afrouz, and V. Tenorio, "A Comprehensive Review on Internet of Things (IoT) and its Implications in the Mining Industry," (in English), *American Journal of Engineering and Applied Sciences*, vol. 13, no. 3, pp. 499-515, 2020-09-03 2020, doi: 10.3844/ajeassp.2020.499.515.
- [3] H. Qudrat-Ullah and P. N. Panthallor, *Operational Sustainability in the Mining Industry: The Case of Large-Scale Open-Pit Mining (LSOPM) Operations*. Springer Nature, 2020.
- [4] M. Rudakov, E. Gridina, and J. Kretschmann, "Risk-based thinking as a basis for efficient occupational safety management in the mining industry," *Sustainability*, vol. 13, no. 2, p. 470, 2021.
- [5] J. Redmon and A. Farhadi, "Yolov3: An Incremental Improvement", arXiv preprint arXiv:1804.02767, 2018," *View Article*, 1804.
- [6] A. R. e. N. T. I. Risi, "«Détecteur d'objets du système radar robotique contrôlé Arduino Uno utilisant un capteur à ultrasons»,," v. Am. J. Ing. Rés., non. Ed., Decembre 2020, p. 1 à 8.
- [7] I. Fenniche, "Etude de lentilles artificielles métalliques et métallo-diélectriques: modélisation par la méthode modale de Fourier et par la méthode des coordonnées curvilignes," Université Blaise Pascal-Clermont-Ferrand II, 2010.
- [8] V. Marin, "Radio-détection des rayons cosmiques d'ultra-haute énergie. Analyse, simulation et interprétation," Université de Nantes, 2011.
- [9] L. Déjean, "«Détection de petites cibles marines en milieu côtier par radar naéroporté »,," Thèse doctorat de Telecom Bretagne, 10 novembre 2009.
- [10] M. Richards, i, (*Radar Technology Handboook*) CRS Press, 2019.
- [11] IEEE, "IEEE Standard for Software and System Test Documentation. ," IEEE Std 829-2008 (Revision of IEEE Std 829-1998), 2010.
- [12] D. M. C. E. Ratmia Mohamed Najib, "Système radar ultrasonique à base d'Arduino pour la détection d'objets."
- [13] M. A. Richards, *Fundamentals of radar signal processing*. Mcgraw-hill New York, 2005.
- [14] J. Darricau, "Physique et Théorie du Radar <https://radars-darricau.fr/livre/2-PDF/chapitre-01-1.pdf>," in ISBN : 978-2-9544675-1-1, 3ème Ed. paris , France févries 2015.
- [15] R. Skolnik, A, *Introduction to Radar Systems* McGraw-Hill, 2008.
- [16] P. Goy, "«Détection d'obstacles et de cibles de collision par un radar FMCW aéroporté». ," Thèse de doctorat de l'université de Toulouse, France., mardi 18 décembre 2012.
- [17] N. DJEDOUÏ and D. BOUKERN, "Étude et réalisation d'un RADAR de détection," Université Aboubakr Belkaïd, Tlemcen. Algérie., 19/06/ 2017.
- [18] M. Parizeau, "Réseaux de neurones," *GIF-21140 et GIF-64326*, vol. 124, 2004.
- [19] C. Touzet, *les réseaux de neurones artificiels, introduction au connexionnisme*. Ec2, 1992 Submitted on 27 Jun 2016.
- [20] N. Thome and C. Wolf, "Histoire des réseaux de neurones et du deep learning en traitement des signaux et des images," 2023.
- [21] S. F. P. Zaki, "Classification par réseaux de neurones dans le cadre de la scattérométrie ellipsométrique," Université de Lyon, 2016.
- [22] K. O'Shea, "An introduction to convolutional neural networks," *arXiv preprint arXiv:1511.08458*, 2015.
- [23] M. E. M. CHOUINI, "Détection d'objets basé Faster R-CNN," 2020.
- [24] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533-536, 1986.
- [25] Y. LeCun *et al.*, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541-551, 1989.

- [26] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324, 1998.
- [27] C. Francois, "Deep learning with Python," ed: Manning Publications, 2018.
- [28] A. Rosebrock, *Deep learning for computer vision with python: Starter bundle*. PyImageSearch, 2017.
- [29] R. Idoui, A. Himeur, and S. E. Bouatmane, "Reconnaissance d'images par les réseaux de neurones convolutionnels (CNN)," Université de Jijel, 2020.
- [30] I. Goodfellow, "Deep learning," ed: MIT press, 2016.
- [31] M. Boughaba, B. Boukhris, and M. Meflah, "L'apprentissage profond (Deep Learning) pour la classification et la recherche d'images par le contenu," 2017.
- [32] B. Hakim and S. Brahim, "Classification des images avec les réseaux de neurones convolutionnels," Université Mouloud Mammeri, 2018.
- [33] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of big data*, vol. 6, no. 1, pp. 1-48, 2019.
- [34] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84-90, 2017.
- [35] R. Girshick, "Fast R-CNN. 2015 IEEE International Conference on Computer Vision (ICCV)," *Santiago*, pp. 7-13, 2015.
- [36] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information processing & management*, vol. 45, no. 4, pp. 427-437, 2009.
- [37] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929-1958, 2014.
- [38] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 6, pp. 1137-1149, 2016.
- [39] W. Liu *et al.*, "Ssd: Single shot multibox detector," in *Computer Vision—ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, 2016: Springer, pp. 21-37.
- [40] J. Redmon, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.
- [41] K. Li, W. Ma, U. Sajid, Y. Wu, and G. Wang, "Object detection with convolutional neural networks," in *Deep Learning in Computer Vision*: CRC Press, 2020, pp. 41-62.
- [42] P. Felzenszwalb, D. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in *2008 IEEE conference on computer vision and pattern recognition*, 2008: Ieee, pp. 1-8.
- [43] P. Sermanet, "Overfeat: Integrated Recognition, Localization and Detection Using Convolutional networks," *arXiv preprint arXiv:1312.6229*, 2013.
- [44] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International journal of computer vision*, vol. 104, pp. 154-171, 2013.
- [45] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580-587.
- [46] J. Redmon, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.
- [47] T.-Y. Lin, P. Goyal, R. Girshick, and K. He, "P. Doll ar," "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis*, 2017, pp. 2980-2988.
- [48] L. Perez, "The effectiveness of data augmentation in image classification using deep learning," *arXiv preprint arXiv:1712.04621*, 2017.
- [49] G. Hinton, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

- [50] A. M'Begnan Nagnan, "Développement d'outils web de détection d'annotations manuscrites dans les imprimés anciens," Université du Québec à Trois-Rivières, 2021.
- [51] A. Wang *et al.*, "Yolov10: Real-time end-to-end object detection," *arXiv preprint arXiv:2405.14458*, 2024.
- [52] D. Bogdoll, M. Nitsche, and J. M. Zöllner, "Anomaly detection in autonomous driving: A survey," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 4488-4499.
- [53] A. Sundaresan Geetha, M. A. R. Alif, M. Hussain, and P. Allen, "Comparative Analysis of YOLOv8 and YOLOv10 in Vehicle Detection: Performance Metrics and Model Efficacy," *Vehicles*, vol. 6, no. 3, pp. 1364-1382, 2024.
- [54] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263-7271.
- [55] C. Li *et al.*, "YOLOv6: A single-stage object detection framework for industrial applications," *arXiv preprint arXiv:2209.02976*, 2022.
- [56] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 7464-7475.
- [57] M. A. R. Alif and M. Hussain, "YOLOv1 to YOLOv10: A comprehensive review of YOLO variants and their application in the agricultural domain," *arXiv preprint arXiv:2406.10139*, 2024.
- [58] G. Boesch, "YOLOv10: Real-Time Object Detection Evolved," <https://viso.ai/deep-learning/yolov10/>, June 6, 2024.
- [59] A. Ghosh, "YOLOv10: The Dual-Head OG of YOLO Series," *university opencv*, vol. <https://learnopencv.com/yolov10/#yolov10-inf>, 4juin2024.
- [60] C. Feng, Y. Zhong, Y. Gao, M. R. Scott, and W. Huang, "Tood: Task-aligned one-stage object detection," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021: IEEE Computer Society, pp. 3490-3499.
- [61] M. CHIBI, "Détection des Piétons par Apprentissage Profond de Transformateur," 2023.
- [62] P. Potrimba, "What is YOLOv10? An Architecture Deep Dive.," in <https://blog.roboflow.com/what-is-yolov10/#nms-free-training-strategy-with-dual-label-assignments>, Roboflow, Ed., ed, Jun 14, 2024.
- [63] C. Li *et al.*, "Yolov6 v3. 0: A full-scale reloading," *arXiv preprint arXiv:2301.05586*, 2023.
- [64] C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, "Yolov9: Learning what you want to learn using programmable gradient information," *arXiv preprint arXiv:2402.13616*, 2024.
- [65] S. Xu *et al.*, "PP-YOLOE: An evolved version of YOLO," *arXiv preprint arXiv:2203.16250*, 2022.
- [66] V. M. Panaretos and Y. Zemel, "Statistical aspects of Wasserstein distances," *Annual review of statistics and its application*, vol. 6, no. 1, pp. 405-431, 2019.
- [67] Y. Chen, X. Yuan, R. Wu, J. Wang, Q. Hou, and M.-M. Cheng, "Yolo-ms: rethinking multi-scale representation learning for real-time object detection," *arXiv preprint arXiv:2308.05480*, 2023.
- [68] A. G. Howard, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.
- [69] A. Wang, H. Chen, Z. Lin, J. Han, and G. Ding, "Repvit: Revisiting mobile cnn from vit perspective," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 15909-15920.
- [70] X. Ding, X. Zhang, J. Han, and G. Ding, "Scaling up your kernels to 31x31: Revisiting large kernel design in cnns," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 11963-11975.
- [71] m. glenn jocher , rizwanmunawar ,hasanghaffari,zhiwuwie; abirami vina, burhan-Q. Université de Tsinghua. (25.05.2024). YOLOv10: Real-Time End-to-End Object Detection.
- [72] M. Sohan, T. Sai Ram, R. Reddy, and C. Venkata, "A review on yolov8 and its advancements," in *International Conference on Data Intelligence and Cognitive Informatics*, 2024: Springer, pp. 529-545.

- [73] T.-Y. Lin *et al.*, "Microsoft coco: Common objects in context," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, 2014: Springer, pp. 740-755.
- [74] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-IoU loss: Faster and better learning for bounding box regression," in *Proceedings of the AAAI conference on artificial intelligence*, 2020, vol. 34, no. 07, pp. 12993-13000.
- [75] Y. Zhao *et al.*, "Detrs beat yolos on real-time object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 16965-16974.
- [76] C. Wang *et al.*, "Gold-YOLO: Efficient object detector via gather-and-distribute mechanism," *Advances in Neural Information Processing Systems*, vol. 36, 2024.
- [77] S. Agarwal, J. O. D. Terrail, and F. Jurie, "Recent advances in object detection in the age of deep convolutional neural networks," *arXiv preprint arXiv:1809.03193*, 2018.
- [78] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117-2125.
- [79] B. Saidani, *Filtre de Kalman discret pour l'estimation des moyennes inconnues de bruits blancs*. Library and Archives Canada= Bibliothèque et Archives Canada, Ottawa, 2013.
- [80] H. Belattar, "Quelques considérations sur le filtre de Kalman discret," Université de Sherbrooke, 2017.
- [81] M. J.-C. Yoccoz, ""Equations différentielles et systèmes dynamiques"".,," *cours à la scuola normale de Pise*, p. 100, 2005.
- [82] D. Alazard, « *Introduction au Filtre de Kalman* », version 0.3 ed. institut supérieur de l'Aéronautique et de l'Espace Toulouse, France., Mai 2011.
- [83] I. É. AquaPortail. ""filtre de Kalman"". (accessed.
- [84] J. L. e. J. P. Stromboni, "Le filtre de Kalman," Ecole Polytechnique de l'Université de Nice, <https://users.polytech.unice.fr/~leroux/moindrescarresbis/node14.html>, France 2000-11-10.
- [85] B. R. & and L. Rima, "Prédiction à l'aide du Filtre de Kalman des Symboles M-QAM dans une Transmission Mono Porteuse," Faculté de TechnologieDépartement d'Électronique, Université SAAD DAHLAB de BLIDA, Blida alger 2014-2015.
- [86] C. K. Chui and G. Chen, *Kalman filtering*. Springer, 2017.
- [87] M. M. Belhaddad Samir, "Détection de suivi automatique des véhicules en trafic routier et autoroutier," 2017.
- [88] F. MEDJAHED, "Détection et Suivi d'Objets en Mouvement Dans Une Séquence d'Images," USTO, 2012.
- [89] M. S. Grewal and A. P. Andrews, *Kalman filtering: Theory and Practice with MATLAB*. John Wiley & Sons, 2014.
- [90] M. S GREWAL and A. P ANDREWS, "KALMAN FILTERING Theory and Practice Using MATLAB," ed: Wiley, 2023.
- [91] D. Simon, *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*. John Wiley & Sons, 2006.
- [92] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons, 2004.
- [93] P. S. Maybeck, *Stochastic models, estimation, and control*. Academic press, 1982.
- [94] T. Yoshikawa, "Proceedings of the IEEE Conference on Robotics and Automation," 2000.
- [95] J. Chen and R. J. Patton, *Robust model-based fault diagnosis for dynamic systems*. Springer Science & Business Media, 2012.
- [96] M. (n.d.). ""Kalman Filter"". (accessed.
- [97] L. Dris, "Les algorithmes adaptatifs de poursuite de cible," Alger, Ecole Nationale Polytechnique, 2008.

- [98] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401-422, 2004.
- [99] l. m. engineers, "Control 28BYJ-48 Stepper Motor with ULN2003 Driver & Arduino," *Online available on: <https://lastminuteengineers.com/28byj48-steppermotor-arduino-tutorial/>*, accessed on: 10.04.2023.
- [100] F. G. Wiki, "Stepper Motor 5V 4-Phase 5-Wire & ULN2003 Driver Board for Arduino," ed, 2012.
- [101] components101, "28BYJ-48 - 5V Stepper Motor," <https://components101.com/motors/28byj-48-stepper-motor>, 5 March 2021.
- [102] o. d. t. world, "Raspberry pi 4 donnees techniques," *RS-online.com*, 2024.
- [103] "Raspberry Pi 4 Model B," <https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-product-brief.pdf>, April 2024.
- [104] G. Halfacree, "LE GUIDE OFFICIEL du débutant Raspberry Pi Comment utiliser votre nouvel ordinateur," *Raspberry Pi Trading Ltd, Maurice Wilkes Building, St. John's Innovation Park, Cowley Road, Cambridge.*, p. 10, 2020.
- [105] M. A. Jonathan Vannieuwerkerke, "Premiers pas avec le module caméra | Raspberry Pi Projects," *Modifiez ceci sur GitHub. <https://www.raspberrypi.com/documentation/accessories/camera.html> . https://esero.fr/wp-content/uploads/2021/04/Premiers-pas-avec-le-module-camera--Raspberry-Pi-Projects_FR.pdf*, 2021.
- [106] K. M. T. Sai Harshita, Ch. Navya, Dr. Y. Chakrapani, "VEHICLE TRACKING SYSTEM USING GPS & RASPBERRY PI," *Industrial Engineering Journal*, vol. 52, no. 6, Department Of Electronics And Communication Engineering ACE ENGINEERING COLLEGE, Medchal-501301, June : 2023.
- [107] G. E. y. d. partnar), "User Manual – u-blox Neo-6M GPS Module," https://www.gie.com.my/download/um/wireless/gps/obsolete/um_gps.pdf, p. 3, 2023.
- [108] "Raspberry pi- GPS," https://newbiely.com/tutorials/raspberry-pi/raspberry-pi-gps#google_vignette, 2024.
- [109] D. Hertz, "Comment utiliser un récepteur GPS avec Raspberry Pi 4," *makepro* .<https://maker.pro/raspberry-pi/tutorial/how-to-use-a-gps-receiver-with-raspberry-pi-4>, 03 février 2020
- [110] D. p. ultrasons, "Détection par ultrasons- principes de base et application," <https://fre.jf-parede.pt/what-is-charge-pump>.
- [111] H. Yamina and B. Djedjiga, "Conception et réalisation d'un radar ultrason," Université Mouloud Mammeri, 2018.
- [112] L. Bachelard, "Module de détection aux ultrasons>> " URL : <https://www.gotronic.fr/pj2-hc-sr04-utilisation-avec-picaxe1343.pdf?fbclid=IwAR3SNYo8b6W0G5ywxU536TWUYJSg3l5a0vLm2pHZwAaS8>, Le 28 novembre 2015
- [113] X. HINAULT, "Librairie SdFatLib pour carte SD du module Ethernet (hors-référence)," *Référence Arduino Français [En ligne]. Disponible sur:< http://www.mon-clubelec.fr/pmwiki_reference_arduino/pmwiki.php*, 2010.
- [114] CFAURY, "Capteur de distance à Ultrasons," <https://arduino.blaisepascal.fr/capteur-de-distance-a-ultrasons/>, 15 JANVIER 2016 MIS À JOUR 25 SEPTEMBRE 2024.
- [115] F. Mocq, *Raspberry Pi 2: exploitez tout le potentiel de votre nano-ordinateur*. Editions ENI, 2015.
- [116] "Raspberry Pi Documentation." <https://www.raspberrypi.com/documentation/> (accessed.
- [117] C. Wolff, Ed. *Radartutorial Chapitre 2: Types de radar et fréquences* ((Traduction en langue française et révision: Pierre Vaillant et Christophe Paumier). 3 août 2011.
- [118] G. N. Nyami, *Analyse D'un système Radar intervéhiculaire En Onde millimétrique (77 GHz)*. Ecole Polytechnique, Montreal (Canada), 2019.
- [119] L. F. d. S. d. l. Technologie, "cours actionneurs : chapitre 4 principe du radar " *Université Frères Mentouri - Constantine 1*, 2019_2020.

- [120] radartutorial. "(Les Principes du Radar ; Radar à onde continue modulée en fréquence)." (accessed.
- [121] B. Zohuri and B. Zohuri, "Fundaments of radar," *Radar energy warfare and the challenges of stealth technology*, pp. 1-110, 2020.
- [122] P. S. V. Hum, "Radio and Microwave Wireless Systems," <https://www.waves.utoronto.ca/prof/svhum/ece422/notes/24-radar.pdf>, vol. University of Toronto, 2021.
- [123] D. R. a. D. O. (DRDO). "Guidelines for Transfer of Technology." (accessed.
- [124] O. DUJARDIN, "Radars « classiques », PESA ET AESA : avantages et inconvénients," <https://cf2r.org/rta/radars-classiques-pesa-et-aesa-avantages-et-inconvenients/>, mars 2023.
- [125] K. Jishy, "Pistage de cibles manoeuvrantes en radar passif par filtrage à particules gaussiennes. ," Institut National des Télécommunications, France, 2011.
- [126] E. Fishler, A. Haimovich, R. Blum, D. Chizhik, L. Cimini, and R. Valenzuela, "MIMO radar: An idea whose time has come," in *Proceedings of the 2004 IEEE Radar Conference (IEEE Cat. No. 04CH37509)*, 2004: IEEE, pp. 71-78.
- [127] J. Li and P. Stoica, *MIMO radar signal processing*. John Wiley & Sons, 2008.
- [128] M. F. AMINA, "DOA-DOD de cibles en radar MIMO bistatique en utilisant les méthodes à haute résolution," MEMOIRE DE FIN D'ETUDES POUR L'OBTENTION DU DIPLOME DE MASTER EN AERONAUTIQUE Localisation Institut d'aéronautique et des études spatiales Département d'aéronautique BLIDA, 2012_2013.
- [129] "Radar quantique : applications à défense et à la surveillance environnementale." (accessed.
- [130] C. Ferrie, "Quantum entanglement : Spooky Action at a Distance " <https://www.editverse.com/quantum-entanglement-spooky-action-at-a-distance/>, février 2023.
- [131] C. J. H. J.L.Chau, M.J.Nicolls, "phased array radars," EISCAT Radar School Sodankyla, 12 August 2012.
- [132] U. Nickel, "Fundamentals of signal processing for phased array radar," *Signal*, vol. 4, p. 4, 2006.
- [133] L. Nicolaescu and T. Oroian, "Radar cross section," in *5th International Conference on Telecommunications in Modern Satellite, Cable and Broadcasting Service. TELSIKS 2001. Proceedings of Papers (Cat. No. 01EX517)*, 2001, vol. 1: IEEE, pp. 65-68.
- [134] M. R. KRATTOU, "Étude de la Détection Radar Dans un Milieu Homogène," 2013.
- [135] B. Boudamouz, "Contribution à l'étude de nouvelles techniques de radar MIMO pour la détection de cibles en contexte urbain (à l'intérieur des bâtiments)," ISAE-Institut Supérieur de l'Aéronautique et de l'Espace, 2013.

ANNEXE

1. Différents types de radars

Les équipements radar varient en fonction des données qu'ils doivent fournir, ce qui implique l'utilisation de technologies et de caractéristiques distinctes. Cette diversité est illustrée par la figure 5 :

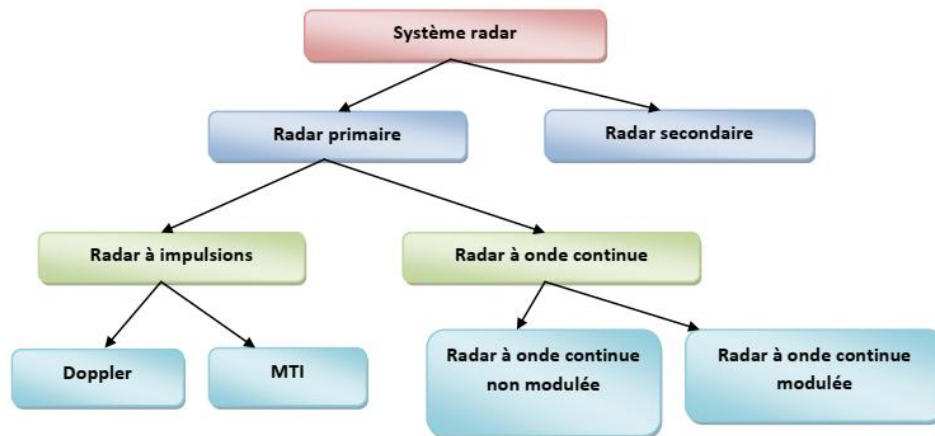


Figure 105
différents types des radars.[18]

1.1. Radar primaire

Le principe du radar primaire repose sur l'envoi d'impulsions RF qui subissent une rétrodiffusion sur les cibles détectées. Ces ondes réfléchies, appelées échos, sont ensuite captées et analysées par le même radar qui les a émises. L'avantage principal de ce type de radar réside dans le fait que la cible détectée reste passive, c'est-à-dire qu'elle n'a pas besoin d'émettre de signal en retour pour être repérée.[117]

1.2. Radar à impulsions

Le système produit des bursts hyperfréquences de haute intensité, chacun suivi d'un intervalle de réception dépassant la durée d'émission. Son application principale consiste en la mesure tridimensionnelle des cibles (position horizontale, distance et altitude) [118] .

1.2.1. Principe du radar à impulsion

Le système à impulsions est largement employé dans les radars. Son fonctionnement repose sur la détection d'échos générés par la réflexion d'impulsions courtes et intenses, concentrées en un faisceau directionnel. Lorsque l'antenne tourne, ces ondes peuvent être orientées vers une direction

prédéfinie. L'émission s'effectue par salves périodiques d'ondes RF, séparées par un délai constant désigné sous le terme PRI (Pulse-Repétition-Interval). Pour éviter les interférences dues aux échos multiples (comme l'effet du "second time around echo"), le récepteur, qui partage l'antenne avec l'émetteur, doit être désactivé pendant l'émission de chaque impulsion RF. De plus, Un calibrage précis de l'intensité et de la fréquence de répétition (PRF) s'impose pour compenser les distorsions de transmission, variables selon l'implantation du système et l'environnement météorologique. Lorsqu'une cible située dans la zone de balayage du radar est touchée par l'onde RF, Le récepteur capture et traite la composante réfléchie de l'énergie émise, qui s'active pendant l'intervalle entre deux impulsions (temps d'écoute) ($T_R - \tau$). La qualité du signal reçu dépend de cette énergie réfléchie. Une cible est considérée comme parfaitement localisée lorsque sa distance et sa direction (azimut et élévation) sont déterminées avec précision. [119]

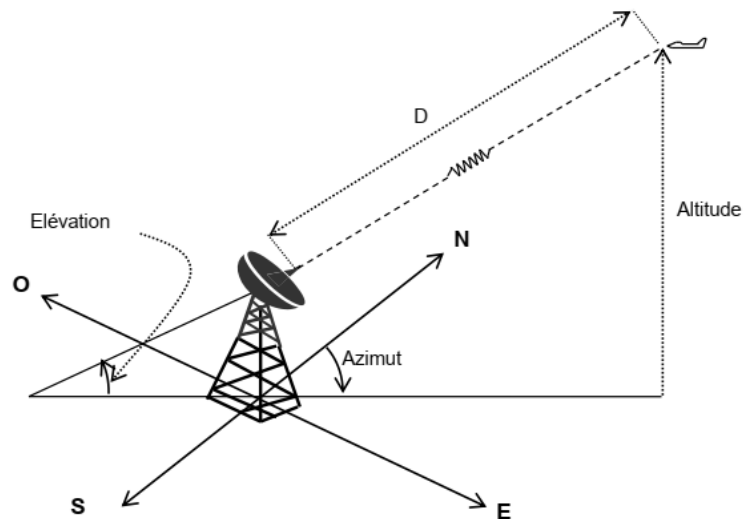


Figure 106
Angles caractérisant une cible radar [19].

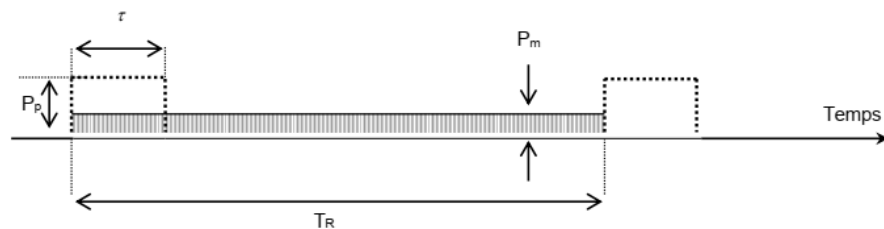


Figure 107
Chronogramme de l'émission d'une impulsion radar [19].

1.3. Radar à ondes continues :

Ce système produit des signaux hyperfréquences en continu. Il capte et analyse le signal réfléchi, ce qui en fait un outil couramment employé pour trouver la vitesse et localisation d'un objet. [118]

1.3.1. Les radars CW sont de deux types :

a. Radar à ondes continues non modulées

Dans ce type de radar, le signal émis maintient une amplitude et une fréquence constantes. Le radar à onde continue (CW), Ce système, opérant en régime d'émission continue non modulée, ne peut évaluer la vitesse que par analyse Doppler. Ses limitations incluent l'impossibilité de mesurer la portée et de discriminer des cibles multiples[12].

b. Radar à ondes continues modulées

Les systèmes radars à onde continue (CW) non modulés présentent une limitation majeure : ils ne permettent pas de mesurer la distance, car ils ne peuvent pas déterminer le temps de parcours du signal. Cette fonctionnalité est en revanche possible avec les systèmes radars CW modulés, Par modulation fréquentielle linéaire (méthode FMCW), le système émet un signal dont la porteuse subit une variation cyclique entre f_1 et f_2 . Cette approche permet déterminer la détection de cibles statiques, ainsi le calcul de distance par analyse différentielle fréquence émise/reçue [12] :

$$R = \frac{c \cdot \Delta t}{2} \quad (4)$$

Avec :

R : la distance.

Δ : la différence de temps entre le rapport (signal émis / l'écho).

Tout mouvement cible induit un décalage Doppler complémentaire, révélant sans ambiguïté le sens de déplacement (approche/éloignement).

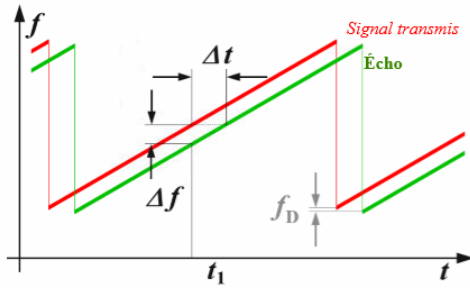


Figure 108
Évaluation des distances avec radar FMCW. [120]

1.4. Radars Doppler :

Les radars Doppler utilisent des émetteurs/récepteurs mobiles, exploitant le décalage fréquentiel caractéristique des sources électromagnétiques en mouvement relatif [121] .

Le radar Doppler est largement utilisé pour mesurer la vitesse des objets en mouvement. Parmi ses applications les plus courantes figurent le contrôle de la vitesse routière et les usages militaires. Il joue également un rôle crucial en météorologie, où il autoriser le suivi de l'évolution des tempêtes en analysant les décalages de fréquence produits par le déplacement des noyaux orageux, aidant ainsi à prévoir leur trajectoire.[121]

1.4.1. Principe du radar Doppler :

Contrairement aux radars classiques à impulsions, la version Doppler utilise une émission CW permanente. L'analyse du déphasage fréquentiel (Δf) causé par le déplacement relatif cible-radar permet d'extraire la vitesse selon l'équation (5), qui intègre le paramètre temporel de transit des ondes [118] .

$$T = \frac{2D}{c} = \frac{2D_0}{c} \pm \frac{2V_d t}{c} \quad (5)$$

La convention de signe applique (+) pour l'éloignement et (-) pour le rapprochement. Avec D_0 comme distance initiale, V_d la vitesse radiale, t le temps écoulé et c la célérité lumineuse, les expressions des signaux émis (S_e) et reçus (S_r) sans bruit s'établissent comme suit :

$$\left\{ \begin{array}{l} S_e(t) = A \cos(2\pi f t) \\ S_r(t) = B \cos(2\pi f(t - T)) = B \cos \left[2\pi f \left(t - \frac{2D_0}{c} \pm 2V_d \frac{t}{c} \right) \right] = B \cos \left[2\pi t \left(f \pm 2f \frac{V_d}{c} \right) + \varphi_0 \right] \end{array} \right\} \quad (6)$$

A représente l'amplitude d'émission, tandis que B correspond à l'amplitude réceptionnée, dépendante des paramètres radar : puissance émise (P_t), section efficace (σ), gain antennaire (G), longueur d'onde (λ_0) et distance cible (R). L'équation (7) permet d'extraire la fréquence Doppler

$$f_d = \pm 2 \frac{f V_d}{c} \quad (7)$$

1.5. Radars à Modulation de Fréquence (FM-CW) :

Contrairement aux systèmes pulsés, un radar CW émet en permanence. La mesure de distance et vitesse s'effectue par modulation temporelle de la fréquence d'émission (technique FM). Ce balayage fréquentiel est généralement linéaire, produisant un signal souvent nommé « chirp ». La Figure 10 illustre l'évolution de la fréquence des signaux transmissent en $f(t)$. Après une phase de montée en fréquence, celle-ci peut redescendre, une étape facultative nécessaire uniquement si le système doit résoudre deux inconnues (position et vitesse). Le signal reçu (écho), représenté par une ligne pointillée, présente un décalage temporel par rapport au signal émis.[122]

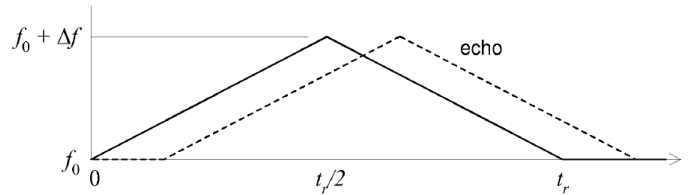


Figure 109
FMCW radar système.[23]

La fréquence instantanée de la forme d'onde transmise peut être exprimée comme [122]:

$$f(t) = f_0 + 2f_r \Delta f t, \text{ où } t \leq t_r/2 \quad (8)$$

$$f(t) = f_0 + \Delta f - 2f_r \Delta f (t - t_r/2), \text{ où } t > t_r/2$$

ou $f_r = 1/t_r$ La fréquence signal de rampe détermine la base du signal émis. L'écho reçu correspond à une réplique de ce signal, décalée dans le temps et modifiée par un éventuel décalage Doppler dû au mouvement de la cible:

$$f_e(t) = f(t - 2R/c) + f_d \quad (9)$$

Lorsque deux signaux sont introduits dans un mélangeur, la fréquence résultante de leur différence permet d'estimer à la fois les deux valeurs de distance ainsi la vitesse de la cible. Si les signaux émis et reçus se superposent et présentent des pentes ascendantes (le segment « haut de fréquence »), la différence de fréquence obtenue devient un indicateur clé pour ces calculs.:

$$f_+ = 4Rf_r\Delta f/c - f_d \quad (10)$$

Tandis que sur le segment à « fréquence décroissante », la différence de fréquence est :

$$f_- = -4Rf_r\Delta f/c - f_d \quad (11)$$

En analysant la sortie du mélangeur lors des deux phases de variation de fréquence, il est possible de recueillir suffisamment de données pour résoudre ces équations simultanément. Cela permet d'aboutir à une détermination précise des valeurs recherchées :

$$R = \frac{c(f_+ - f_-)}{8f_r\Delta f}$$

$$\text{Et} \quad (12)$$

$$f_d = -\frac{f_+ + f_-}{2}.$$

La dernière équation peut être utilisée pour résoudre la vitesse de la cible.

5.6. Radar à synthèse d'ouverture (RSO) :

Le RSO (radar à synthèse d'ouverture) est un système cohérent aéroporté ou spatial dédié à l'imagerie latérale de surfaces terrestres/maritimes, utilisant un traitement numérique des échos radar séquentiels, il simule la résolution d'une antenne virtuelle bien plus précise que l'antenne physique utilisée. En se déplaçant, le radar émet de multiples impulsions par seconde, balayant ainsi une même zone au sol sous différents angles et positions. Chaque pulse enregistre des fluctuations d'amplitude (fonction de la SER) et des déphasages Doppler (induits par l'évolution de la distance oblique). Ce principe est comparable à la perspective changeante d'un passager d'avion observant un point fixe au sol.[117]

1.6.1. Principe et fonctionnement du radar RAS :

Le RSO opère selon un concept similaire aux antennes réseau phasées, mais utilise des acquisitions séquentielles décalées spatio-temporellement pour former une mesure synthétique composite, plutôt que des éléments multiples simultanés.[117]

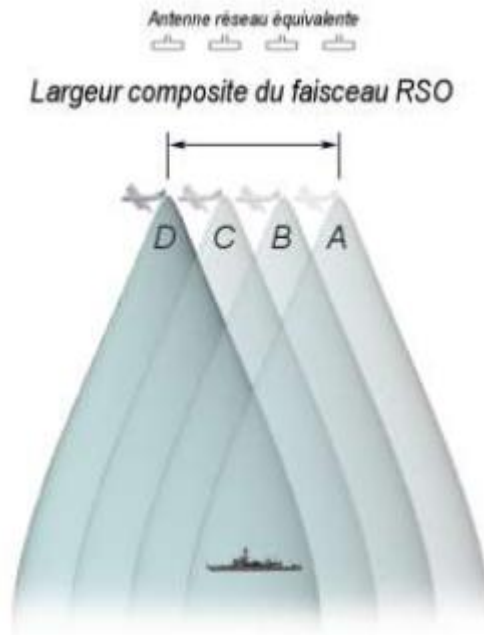


Figure 110
Synthèse du faisceau total.[18]

Les cibles (ex : navires) n'occupent qu'une fraction du lobe de rayonnement radar. En se déplaçant, le porteur du radar capte les échos rétrodiffusés par la cible, dont l'intensité et la phase varient en fonction de sa position relative. Les données de ces échos sont enregistrées pendant une durée T , correspondant au temps nécessaire au porteur pour parcourir la distance entre deux points (A à D). Ces données reconstruisent le signal d'une antenne virtuelle de dimension $v \times T$ (v : vitesse plateforme, T : durée d'acquisition). Grâce au changement de perspective dû au mouvement, cette synthèse équivaut à étendre la longueur de l'antenne réelle, améliorant ainsi la résolution. [117]

Le RSO atteint une résolution avoisinant $L/2$ (L = longueur antenne physique), performance stable quelles que soient l'altitude ou la portée. La compensation automatique entre largeur de faisceau synthétique et durée d'observation garantit une résolution homogène sur l'entièreté du couvert radar [117].

1.6.2. Équipement nécessaire à un RSO :

- Un émetteur parfaitement cohérent et à stabilité fréquentielle élevée
- Une unité de calcul haute performance pour le traitement temps-réel des données
- Une métrologie précise de la trajectoire et vitesse de la plateforme

Grâce à l'astuce du RSO, les experts en domaine ingénierie peuvent atteindre des résolutions élevées avec une antenne de taille standard, évitant ainsi l'utilisation d'une antenne géante pouvant mesurer jusqu'à 10 mètres. Une telle antenne de grande taille serait impraticable à installer sur un avion sinon un satellite.[117]

Il existe 3 modes des sondages :

1) _ En bande : fonctionnement analogue au RVL avec angle d'incidence radar constant.

2) _ Saisie hyperfine: ainsi que "Spotlight" ou "Spot", ce mode consiste à maintenir le radar pointé vers une zone spécifique même lors du déplacement du porteur. Pour cela, (Utilisation d'une antenne phased-array pilotée numériquement. Cette approche multiplie les acquisitions surfaciques comparé aux systèmes conventionnels, améliorant la résolution via l'extension de l'ouverture synthétique, mais réduit la zone couverte.).

3) _ Balayage : balayage angulaire entre nadir et incidence spécifique. Le déplacement de la plateforme génère une trajectoire en zigzag lorsque l'angle varie progressivement, produisant une couverture en bandes parallèles espacées si le cycle recommence systématiquement au nadir[117].

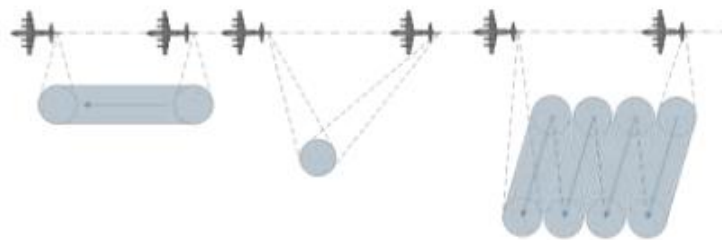


Figure 111
Représentations graphiques de différents modes RSO.[18]

5.7. Radars secondaires :

Le système radar secondaire utilise un transpondeur sur la cible pour recevoir des interrogations et renvoyer des réponses sur une autre fréquence, permettant une détection et une communication actives. La collaboration de la cible (via un transpondeur) optimise significativement la puissance d'émission requise, tout en offrant une portée de détection équivalente à celle d'un radar primaire. [117]

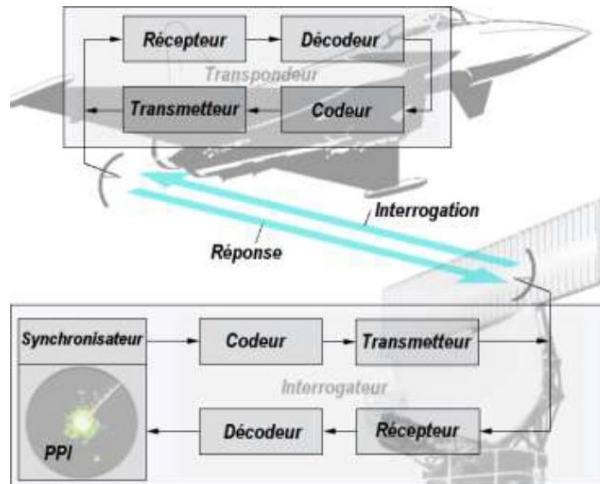


Figure 112
Schéma fonctionnel d'un radar secondaire [18].

La puissance d'émission constitue une variable fondamentale dans la modélisation radar. Pour un radar primaire, elle doit prendre en compte le trajet aller-retour de l'onde, tandis que pour un radar secondaire, seul le trajet aller est considéré. L'énergie du signal varie selon une relation en $1/R^2$ à chaque étape du trajet, comme le décrit l'équation du radar. [117]

$$R \sim \sqrt[4]{P_{PSR}} = \sqrt[2]{P_{SSR}}$$

Figure 113 Équations différentielles de puissance PSR vs SSR [18].

Les radars secondaires nécessitent une puissance d'émission typiquement >1000 fois inférieure aux systèmes primaires équivalents. Cette caractéristique permet de : Émetteurs simplifiés (compacité, coût réduit) Relaxation des contraintes récepteurs (signaux transpondés plus intenses que les échos passifs). Cependant, cette configuration peut amplifier les lobes secondaires, nécessitant l'utilisation de dispositifs spécifiques pour les supprimer[117].

1.8. Radar imageur et non imageur

A. Radar imageur :

Ce dispositif génère des images haute résolution pour la cartographie terrestre et planétaire, avec applications duales : militaires (identification cible) et civiles (navigation aéronautique/maritime) [118].

B. Radar non imageur :

Ces systèmes se classent en deux familles distinctes : les radars cinémométriques, utilisant l'effet Doppler-Fizeau pour évaluer la vitesse des véhicules, et les altimètres radar, dédiés à la mesure précise des hauteurs (distance verticale) [118].

1.9 Radar AESA :

Le réseau actif à balayage électronique repose sur une multitude de modules d'émission et de réception (TRM) pour atteindre la puissance nécessaire au fonctionnement du radar. Le traitement des signaux est assuré par un processeur de réception et d'excitation (ERP) de haute performance, spécialement conçu pour les avions de chasse. L'utilisation d'une antenne réseau phasée confère à ce système une robustesse opérationnelle accrue et une capacité multitarget avancée. Les modes opérationnels du radar sont adaptés pour soutenir le pilote dans diverses missions de combat, qu'elles soient air-air, air-sol ou air-mer.[123]

1.9.1. Caractéristiques principales :

Le radar AESA dédié au contrôle de tir est conçu pour fonctionner en mode multimode entrelacé, répondant aux exigences des avions de combat modernes. Grâce à ses technologies ECCM (contre-mesures électroniques) avancées, il optimise la perception de l'environnement tactique. Ce système permet un suivi précis de multiples cibles et intègre des fonctionnalités air-air, air-sol et air-mer, offrant ainsi une solution polyvalente adaptée à tous les types de missions. [123]

1.9.2. Les avantages

1. Basés sur une architecture technologique éprouvée, ces radars offrent un excellent rapport coût-performance et une fiabilité à toute épreuve.
2. Leur conception antennaire simplifiée garantit une maintenance aisée et une résistance accrue aux dommages physiques.
3. Leur immunité aux IEM est remarquable, notamment via des protections avancées du chaînon récepteur.

4. Leur capacité à générer des puissances crêtes élevées varie selon la configuration de l'émetteur.
5. Le dépointage mécanique des antennes permet d'optimiser la signature radar du porteur [124]

1.9.3. Les inconvénients

1. Les oscillateurs à cavité (klystron/magnétron/TWT) offrent une agilité fréquentielle réduite comparé aux technologies à état solide, limitant ainsi la capacité de saut de fréquence impulsion-à-impulsion.
2. Leur efficacité énergétique est modérée en raison des pertes thermiques significatives (au niveau des tubes RF et des guides d'ondes), impactant directement la sensibilité globale du système.
3. La contrainte de balayage mécanique restreint la flexibilité des waveforms : bien que capables de changer de mode, ces radars ne peuvent opérer simultanément en multi-mode (ex: impossible de combiner efficacement les modes air-air et air-sol sur un chasseur en raison des limitations physiques de pointage).
4. Leur vulnérabilité potentielle au brouillage est accrue (notamment pour les architectures mono-oscillateur), bien que ce critère dépende fortement des implémentations spécifiques. [124] .

1.9.3. Domaines d'application

Le radar de contrôle de tir basé sur AESA peut être configuré pour être utilisé sur n'importe quel avion de chasse.[123]

1.10. Radar Passif

5.10.1 Principe et Fonctionnement :

Le radar passif, contrairement aux systèmes radar actifs, utilise des émetteurs externes (comme la radiodiffusion) pour détecter des cibles sans émettre de signaux propres. Il est souvent bistatique (émetteur et récepteur séparés) et désigné par des termes comme Localisation Passive Cohérente (PCL) ou Radar Passif Secret (PCR).[125]

Les signaux qu'un PBR peut utiliser peuvent être classés en différentes catégories :

- Diffusion Audio : Radios ([AM](#), [FM](#) et numérique [DAB](#)).
- Diffusion Vidéo : Télévision (analogique, numérique terrestre ([DVB-T](#))).

- Réseaux téléphonique mobiles : [Wi-Fi](#), [WiMax](#).
- Réseaux métropolitains ainsi locaux : [GSM](#) et [UMTS](#).
- Systèmes d'émission satellitaires pour communications et positionnement : constellations GNSS, diffusion TV numérique (DVB-S).

1.10.2. Les avantages d'un radar passif sont les suivants :

- La signature indétectable constitue l'atout majeur des radars passifs : ils exploitent des rayonnements ambiants (TV, radio, GNSS) sans émettre, échappant ainsi aux CME. Cette furtivité intrinsèque en fait des solutions privilégiées pour les opérations discrètes.
- Économies substantielles : l'absence d'émetteur dédié réduit les coûts d'acquisition et de maintenance de 40-60% vs systèmes bistatiques.
- Respect du spectre : aucun besoin d'allocation fréquentielle supplémentaire (utilisation de bandes civiles existantes).
- Compacité : encombrement réduit permettant un déploiement en zones contraintes (urbain, théâtre opérationnel).
- Écologie électromagnétique : impact EM 90% inférieur aux radars actifs, idéal pour les zones sensibles.[125]

1.10.3 Les inconvénients :

- Dépendance critique à des sources externes non contrôlées : les formes d'onde disponibles sont sous-optimales pour la détection radar, avec des variations imprévisibles de leurs paramètres (fréquence, puissance, modulation) durant les missions.
- Absence de maîtrise géographique : impossibilité d'optimiser la position et les caractéristiques des émetteurs tiers, limitant les performances du système.
- Interférences majeures : présence dominante du signal direct (DSI) et dégradations importantes dues aux trajets multiples (multipath).
- Complexité réceptrice : nécessité d'implémenter des chaînes de réception dédiées, vulnérables aux artefacts de propagation multipath.[125]

1.11. Radar MIMO :

La technologie MIMO (Multiple Input / Multiple Output), couramment utilisée dans les télécommunications, se base sur l'emploi de multiples antennes émettrices et réceptrices. Cette méthode a permis d'optimiser considérablement l'efficacité des réseaux de communication. Appliqué aux radars, qui intègrent déjà des antennes multiples en émission et en réception, le

MIMO peut être vu comme un système radar associant plusieurs émetteurs et plusieurs capteurs[126]. Le principe du radar MIMO s'appuie sur une analyse combinée des signaux captés, étendant ainsi l'approche multistatique. Dans une acception large, plusieurs radars classiques peuvent être vus comme des configurations spécifiques du MIMO. C'est le cas, par exemple, du radar à synthèse d'ouverture (SAR) ou des systèmes planimétriques exploitant une double polarisation par impulsion. Les émetteurs d'un radar MIMO génèrent des formes d'onde potentiellement corrélées entre elles – l'orthogonalité n'étant pas obligatoire, mais facilitant généralement l'extraction des informations[127] .

1.11.1. Avantages de radar MIMO

Dans le premier cas de figure, où émetteurs et récepteurs sont spatialement séparés, la multiplicité des angles d'observation améliore nettement les capacités de détection. La seconde architecture, caractérisée par un regroupement des antennes émettrices et réceptrices, optimise quant à elle la précision de localisation. L'intégration de l'approche MIMO aux systèmes radar permet de constituer un réseau virtuel bien plus développé que les architectures classiques. Cet élargissement du réseau virtuel accroît sensiblement l'efficacité de détection pour les radars statistiques, tout en renforçant la précision géométrique des radars cohérents[128].

Les radars MIMO cohérents exploitent un renforcement dû à la synchronisation parfaite des ondes, alors que les systèmes MIMO statistiques capitalisent sur l'hétérogénéité des positions antennaires. Pour fusionner ces avantages complémentaires, une architecture hybride peut être implémentée : chaque unité émettrice intègre plusieurs éléments rayonnants groupés (constituant un sous-ensemble compact), tandis que l'espacement entre ces sous-ensembles assure une décorrélation statistique des émissions. Le radar bistatique cohérent MIMO illustre parfaitement cette approche, combinant simultanément les bénéfices de la dispersion géographique et des effets cohérents[128].

1.11.2. Application des systèmes radar MIMO

Les premières expérimentations des radars MIMO ont été menées dans le cadre de la surveillance aérienne. Par la suite, ces systèmes ont été explorés pour leur capacité à réduire les interférences dues au fouillis (clutter). Leurs performances ont également été analysées pour des applications embarquées sur des aéronefs.[128]

1.12. Radar quantique

1.12.1. Définition :

Un système radar quantique exploite les propriétés de l'intrication quantique pour créer une liaison corrélée entre une onde de référence et un faisceau propagé dans un milieu donné. Le traitement du signal repose sur des mesures quantiques liées aux propriétés physiques associées au degré de liberté utilisé pour créer cette intrication.[117]

1.12.2. Principes de fonctionnement du radar quantique :

Les systèmes radar quantiques exploitent les phénomènes de mécanique quantique tels que l'intrication quantique et le principe de superposition d'états pour obtenir une meilleure résolution et une plus grande fidélité du signal détecté. Ces propriétés uniques offrent des capacités supérieures à celles des radars classiques [129].

Les systèmes radar quantiques exploitent des paires de photons corrélés qui entrent en interaction avec les objets à détecter. L'analyse des perturbations induites par ces échanges permet un suivi et une localisation d'une extrême précision. À la différence des radars conventionnels, cette technologie présente une faible signature détectable, tirant parti des propriétés quantiques pour minimiser son exposition. Elle démontre également une robustesse supérieure face aux perturbations et aux tentatives de brouillage, garantissant une fiabilité optimale même dans des environnements hostiles. Face aux défis sécuritaires modernes, le radar quantique émerge comme une solution clé, offrant aux armées et aux services de renseignement une supériorité tactique déterminante pour la protection des intérêts stratégiques. [129].

1.12.3 Applications du radar quantique

Les radars quantiques révolutionnent les techniques de surveillance environnementale. En exploitant les propriétés singulières des particules quantiques, ils permettent de détecter avec précision les variations climatiques, d'anticiper les risques naturels et de préserver les écosystèmes[130].

1.13. Radar phased array

Radar phased array est un ensemble d'antennes dont le diagramme de rayonnement global peut être ajusté en modifiant le déphasage des signaux émis par chaque élément individuel. En ajustant ces

déphasages, il est possible de diriger ou d'atténuer le rayonnement dans des directions spécifiques, sous réserve des contraintes suivantes : [131]:

- Le motif de rayonnement des éléments,
- La taille du réseau,
- La configuration du réseau

1.13.1 Principe et fonctionnement du radar phased array

Les réseaux phasés fonctionnent sur le principe de la création d'une onde plane à partir de multiples ondes sphériques élémentaires, comme illustré dans la Figure 14. Cette figure présente également des exemples concrets d'antennes en réseau. Les ondes sphériques sont approximativement générées par des éléments d'antenne quasi omnidirectionnels. Le processus consistant à appliquer des excitations adaptées à ces éléments d'antenne, puis à combiner leurs signaux lors de la réception, est appelé formation de faisceau[132].

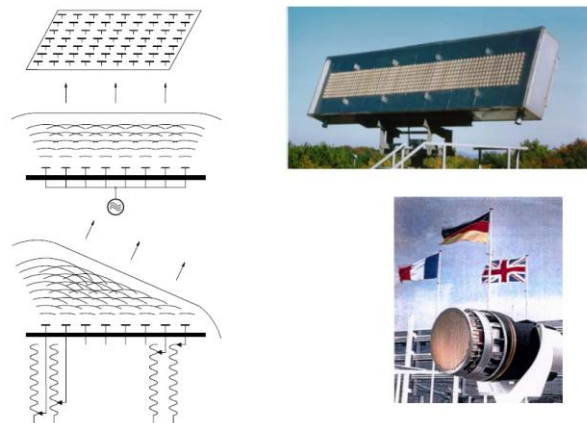


Figure 114
Principe de l'antenne réseau et quelques réalisations phased arrays.[33]

Offrent l'avantage de rediriger le faisceau de manière quasi instantanée, permettant une illumination optimisée de l'espace sans mouvement mécanique. La gestion de l'énergie est un atout clé, incluant [132]:

- **Intégration cohérente presque arbitrairement longue** : Cela permet une meilleure suppression des brouillages (discrimination Doppler), une classification des cibles par extraction des caractéristiques spectrales, et enfin, le traitement SAR et ISAR.

- **Performance de différentes tâches radar en temps multiplexé** : Cela permet d'utiliser un radar à réseau phasé comme un radar multifonctionnel, capable de rechercher et de suivre plusieurs cibles.
- **Optimisation des tâches radar individuelles** : Cela inclut des formes d'ondes optimisées pour la recherche, l'acquisition et le suivi, des mesures de haute précision lorsque nécessaire, des formes de faisceau variables, et des algorithmes optimisés pour le suivi (le radar est "dirigé" par l'algorithme de suivi et les informations préalables).
- **Optimisation énergétique** : Les systèmes utilisant des réseaux phasés actifs présentent un avantage notable en matière d'efficacité énergétique, avec une consommation réduite de près de 50% comparativement aux architectures conventionnelles.
- **Temps moyen entre pannes (MTBF) élevé** : Les réseaux phasés actifs bénéficient d'une dégradation gracieuse, ce qui améliore la fiabilité et augmente le MTBF.
- **Traitements adaptatifs disponibles** : Si des échantillons spatiaux sur l'ouverture de l'antenne sont disponibles, on peut appliquer :
- **Formation de faisceau adaptative (ABF)** : Ajustement dynamique du faisceau pour optimiser la réception.
- **Traitement adaptatif espace-temps (STAP)** : Intégration des informations spatiales et temporelles pour améliorer la détection et la suppression du bruit.
- **Super-résolution** : Amélioration de la résolution angulaire pour distinguer des cibles très proches.

1.13.2. Les avantages :

- Amplification significative du signal
- Réorientation ultra-rapide du lobe radar (échelle microseconde)
- Pilotage intégral par système informatique
- Commutation instantanée entre modes surveillance et tracking
- Durée d'illumination ajustable dynamiquement

- Capacité d'émission multifaisceaux multibandes
- Dégradation progressive en cas de défaillance (perte de résolution sans interruption de service)[117].

1.13.3. Les désavantages :

- Champ de vision restreint à $\pm 60^\circ$ autour de l'axe orthogonal au radar, tant en azimut qu'en élévation.
- Altération des caractéristiques du lobe radar due aux effets de balayage angulaire.
- Manque de réactivité dans les bandes de fréquences basses.
- Conception technologique lourde (système de contrôle, réseaux de déphaseurs...) entraînant des coûts élevés. [117].

1.13.4. Applications de Radars phased array

Les radars à réseau phasé sont un élément courant de nombreux autres systèmes radar. Voici quelques-unes de leurs applications :

- **Espace** : Utilisés pour suivre les engins spatiaux et pour la recherche des planètes lointaines.
- **Sol** : Fréquemment employés par les militaires pour le suivi des navires, des avions et d'autres objets.
- **Aérien** : Utilisés dans les systèmes de défense antimissile

2.Surface équivalente Radar

La SER (Surface Équivalente Radar), également appelée RCS (Radar Cross Section), constitue une caractéristique physique fondamentale des cibles. Elle exprime leur aptitude à renvoyer une onde électromagnétique incidente vers sa source. Techniquement, elle correspond au rapport entre la puissance rétrodiffusée par unité d'angle solide et la densité de puissance de l'onde arrivant sur la cible. En d'autres termes, la RCS caractérise l'efficacité d'un objet à produire un écho radar détectable [133].

Soient Q_b L'intensité énergétique du signal renvoyé par la cible et capté par le système récepteur et Q_t La puissance du signal réfléchi par la cible et reçu par les dispositifs de réception, $Q_b = \frac{P_b}{4\pi R^2}$ avec $P_b = \sigma Q_t$ La puissance émise équivalente au niveau de la cible et σ représentant la Surface Équivalente Radar (SER). La formulation mathématique générale décrivant la SER σ peut être exprimée à travers les relations suivantes [133]:

$$\sigma = \frac{P_b}{Q_t} = \frac{Q_b}{Q_t} 4\pi R^2 \quad (13)$$

L'amplitude champ électrique :

$$\sigma = 4\pi \lim_{R \rightarrow \infty} \left(R^2 \frac{|E^2|^2}{|E^t|^2} \right) \quad (14)$$

La SER des cibles ne peut pas être réduite à une simple constante. Son ampleur est déterminée par différents paramètres incluant la géométrie de la cible, les propriétés des matériaux constitutifs, la fréquence du signal électromagnétique employé, de même que l'orientation relative des ondes incidentes et réfléchies. La SER présente une relation directe avec l'intensité des signaux rétrodiffusés, suivant un modèle équivalent à une absorption partielle de l'énergie incidente par la cible suivie d'une retransmission isotrope de cette énergie[34].

La SER (Surface Équivalente Radar) mesure l'énergie réfléchie vers le radar, rapportée à une sphère de référence d'1 m². Exprimée en dBsm, elle présente une forte variation fréquentielle (Fig.15) : les fréquences élevées produisent généralement des SER plus fortes, facilitant la détection des petites cibles. Trois facteurs clés la déterminent (La surface apparente de l'objet, Son taux de rétrodiffusion (énergie interceptée et renvoyée) et Sa directivité (amplification du signal vers le récepteur vs source isotrope))L'équation (15) intègre ces paramètres. [34].

$$\sigma = \text{Surface projetée} \times \text{Réflectivité} \times \text{Directivité} \quad (15)$$

Tableau 7
Coefficients SER de référence pour différentes cibles de base.[34]

Objets	Expression de la SER
Sphère	$\sigma = \frac{12\pi a^4}{\lambda^2}$, avec a le rayon de la sphère
Plaque plate carrée	$\sigma = \frac{4\pi a^2}{\lambda^2}$, avec a la longueur de la plaque
Cyclindre	$\sigma = \frac{2\pi r h^2}{\lambda}$, avec r rayon du cylindre et h sa hauteur

3. Equation Radar

L'équation radar établit le bilan énergétique complet du système, tenant compte de la puissance émise, de la réflexion sur la cible (via sa SER) et du signal capté, tout en intégrant les paramètres des composants (émetteur, récepteur, antennes) et les conditions environnementales. La figure 16 schématise ces interactions fondamentales.

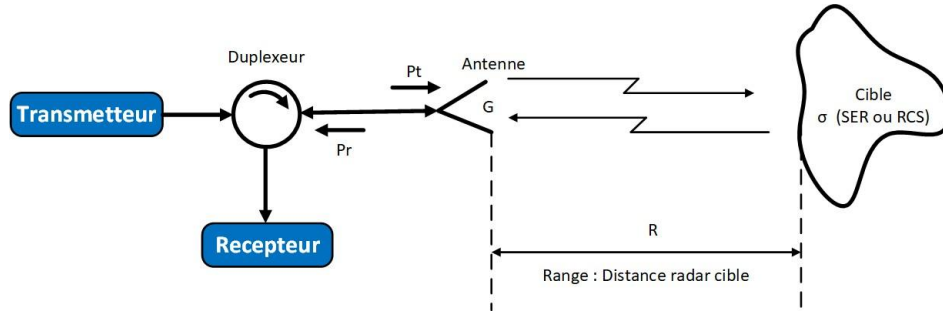


Figure 115
Composants essentiels d'un système de communication radar.[34]

La puissance émise P_t correspond à la puissance crête pour un radar impulsionnel, et à la puissance moyenne pour un radar à onde continue. La puissance reçue P_r est celle captée par le récepteur. En notant :

A : surface d'ouverture de l'antenne

η : rendement antennaire

A_e : surface effective

$G_t = G_r = G$: gain identique à l'émission et réception, On obtient les relations suivantes :

$$\begin{cases} G = \frac{4\pi A_e}{\lambda_0^2} \\ A_e = \eta A \end{cases} \quad (16)$$

Comme le montre la figure 15, l'équation radar se construit en plusieurs étapes claires :

1. La densité de puissance sur la cible (Q_t)

$$\begin{cases} Q_t = \frac{P_t}{4\pi R^2}, \text{ Antenne isotrope } (G_t = 1) \\ Q_t = \frac{P_t G_t}{4\pi R^2}, \text{ Antenne directive } (G_t \neq 1) \end{cases} \quad (17)$$

2. Puissance rétrodiffusée par la cible (P_b)

$$P_b = \frac{P_t G_t}{4\pi R^2} \sigma \quad (18)$$

3. La densité de puissance reçue par le radar (Q_R)

$$Q_R = \frac{P_t G_t}{4\pi R^2} \frac{\sigma}{4\pi R^2} \quad (19)$$

4. La puissance reçue par système radar (PRadar) :

La puissance dépend de celle renvoyée par la cible. En considérant la relation (20) et la surface efficace de l'antenne (A_e) donnée par (3.4), on obtient :

$$P_r = P_{\text{Radar}} = \frac{P_t G_t}{4\pi R^2} \frac{\sigma}{4\pi R^2} \frac{G_r \lambda_0^2}{4\pi} \quad (20)$$

Ce projet considère un radar monostatique ($G_t = G_r = G$). Dans ce cas, l'équation radar devient :

$$P_r = \frac{P_t G^2 \sigma \lambda_0^2}{(4\pi)^3 R^4} \quad (21)$$

Si on note $P_{r_{min}}$ La portée maximale de détection dépend de la sensibilité du récepteur à la puissance minimale détectable.

$$R = R_{max} = \frac{P_t G^2 \sigma \lambda_0^2}{(4\pi)^3 P_{r_{min}}} \quad (22)$$

3.1 Pertes dans le système (L_{sys}) :

Comme toute technologie, les systèmes radar subissent des pertes souvent considérées comme négligeables dans les modèles théoriques. Ces pertes regroupent.[118] :

- A_i : Affaiblissement dans les circuits émetteur/récepteur
- L_f : Atténuation par variation de la surface équivalente
- L_{Ga} : Pertes liées au balayage (fluctuation du gain antenne)
- L_o : Pertes par échos parasites

Donc l'équation des pertes total est :

$$L_{sys}(dB) = A_i(dB) + L_f(dB) + L_{Ga}(dB) + L_0(dB) \quad (23)$$

En pratique, ces pertes réduisent la portée effective du radar.

$$R_{max} = \left(\frac{P_t G^2 \sigma \lambda_0^2}{(4\pi)^3 P_{r_{min}} L_{sys}} \right)^{1/4} \quad (24)$$

Avec $L_{sys} > 1$

7. Portée limite du radar en environnement bruité thermiquement :

Le bruit, principale limitation de la sensibilité réceptrice, nécessite une quantification pour évaluer ses effets. Cette perturbation électromagnétique parasite masque le signal désiré. Ses sources incluent le récepteur et l'antenne, persistant même dans des conditions optimales. Même sans interférences externes et avec un récepteur idéal, le mouvement thermique des électrons dans les résistances génère inévitablement un bruit de fond[12].

Le bruit thermique provient des résistances d'un circuit et augmente avec leur température et la bande passante utilisée. Sa puissance varie en fonction de la largeur de bande (Hz) et de la température (K)[12]:

$$P_{th} = K \cdot T \cdot b_n \quad (25)$$

Avec K, constante de Boltzmann dont la valeur : $1.38 \times 10^{-23} \text{ J/K}$

$$b_n = \frac{\int_{-\infty}^{\infty} |H(f)|^2 df}{|H(f_0)|^2} \quad (26)$$

Ou $H(f)$: la réponse fréquentielle du filtre IF.

f_0 : Le pic spectral (fréquence max) se situe généralement au centre de la bande passante.

3.2.1 Figure de bruit :

La puissance du bruit en sortie d'un récepteur équivaut au bruit thermique d'un récepteur parfait, amplifié par un coefficient nommé facteur de bruit (F_n). Ce dernier se calcule à l'aide d'une formule dédiée[17].

$$F_n = \frac{\text{bruit à la sortie d'un récepteur réel}}{\text{bruit à la sortie d'un récepteur idéal}} = \frac{S_i/N_i}{S_0/N_0} \quad (27)$$

Avec : S_i : Bruit d'entrée (récepteur réel).

N_i : Bruit d'entrée (récepteur idéal)

S_0 : Bruit de sortie (récepteur réel).

N_0 : Bruit de sortie (récepteur idéal).

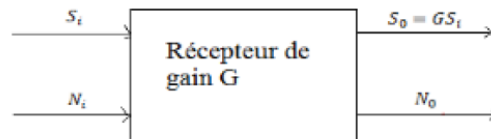


Figure 116
Classification des interférences dans un récepteur radar.[35]

Avec :

$$N_i = K \cdot T \cdot b_n \quad (28)$$

Le facteur de bruit devient :

$$F_n = \frac{N_0}{G \cdot K \cdot T \cdot b_n} \quad (29)$$

3.2.2 Filtrage sous-optimal

La bande passante à -3 dB correspond à l'écart entre les fréquences où la réponse fréquentielle atteint son maximum et celles où elle diminue à 0,707 (soit -3 dB). Cette mesure est couramment utilisée car elle est simple à déterminer et ne nécessite pas une connaissance exhaustive pour la fréquentielle $H(f)$. [134]

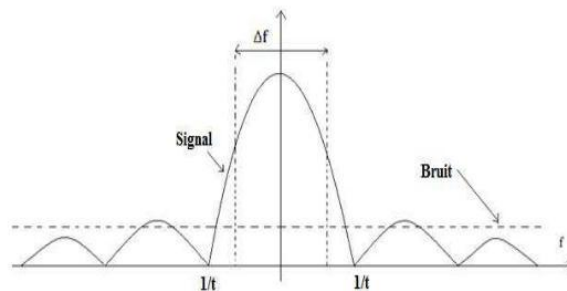


Figure 117
Analyse spectrale : signal impulsionnel et bruit blanc [134].

Un signal impulsionnel a un spectre illimité, nécessitant un filtre passe-bande pour restreindre sa bande fréquentielle. La sélection de cette bande influence directement le rapport signal/bruit : une bande plus large améliore le signal mais accroît aussi le bruit, et vice versa. Les calculs montrent

que le SNR est maximal pour une bande optimale ($\Delta = 1.2$). Ainsi, un filtre avec cette largeur n'est pas idéal.[17]

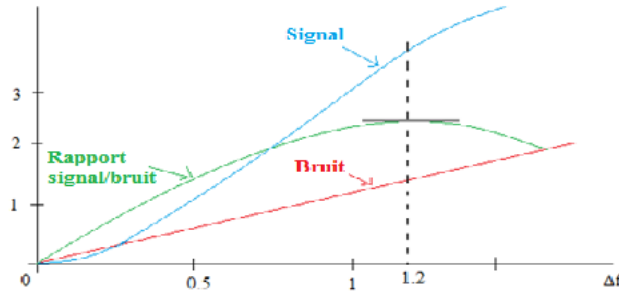


Figure 118
Évolution du rapport signal/bruit selon la bande passante [17].

3.2.3 Filtrage optimal (utilisation d'un filtre adaptatif)

Le filtre adapté optimise le rapport signal/bruit (SNR) en sortie pour un signal noyé dans un bruit blanc additif (à moyenne nulle et spectre connu), sans nécessiter de distribution gaussienne. Son efficacité repose sur l'ajustement précis de la réponse du récepteur au signal incident.[17].

On utilise à l'entrée :

$$X(t) = s(t) + n(t), t \in [0, T] \quad (30)$$

Le rapport signal/bruit maximal est uniquement fonction de l'énergie E du signal, quelle que soit sa forme d'onde [17]:

$$SNR(t_0)^{max} = \frac{2 \cdot E}{N_0} \quad (31)$$

3.3 Système radar à émission multifréquence

Les fluctuations de la surface radar des cibles créent des aléas de puissance, incitant de nombreux systèmes à employer plusieurs fréquences pour optimiser l'éclairage radar. Cette approche utilise typiquement deux émetteurs synchronisés, irradiant la cible avec des signaux suffisamment espacés en fréquence pour garantir leur indépendance à la réception [17]

4. Les applications radar

Les radars, essentiels en surveillance civile et militaire, assurent le contrôle aérien, maritime et routier, ainsi que l'observation terrestre et météorologique (fig 19). Militaires, ils détectent avions,

missiles ou tirs d'artillerie. Civils, ils servent en météo, trafic et sécurité. Certains, comme les radars subsurface, localisent des objets enterrés ou derrière des murs, utiles pour le déminage et les opérations urbaines [135]

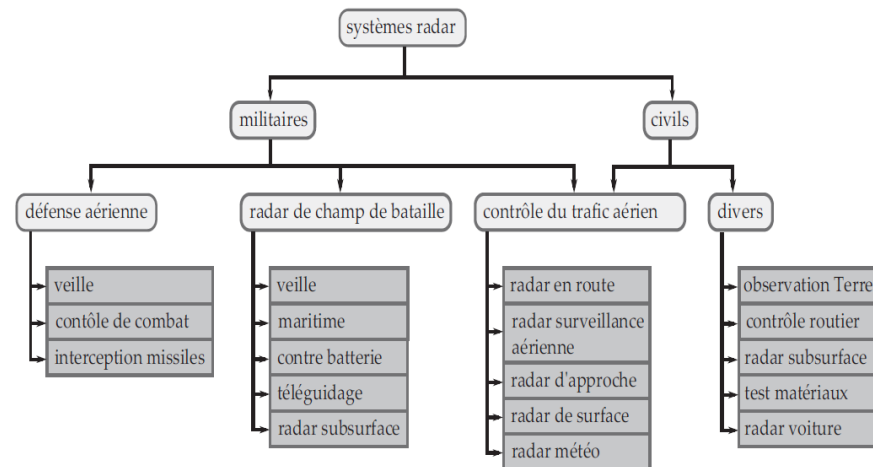


Figure 119
Les applications radar.[36]