



[METTRE NOM DEPARTEMENT]

RAPPORT DE STAGE DE FIN D'ETUDES

En vue de l'obtention du diplôme de Licence en :

Elaboré par :

[Nom & Prénom Etudiant 1]

[Nom & Prénom Etudiant 2]

Encadré par :

[Nom & Prénom Encadreur]

**Institut Supérieur des Sciences Appliquées et
de Technologie de Sousse**

[Nom & Prénom Encadreur
Industriel]

[Nom Entreprise]

Année Universitaire : /

Code Sujet:

DÉDICACE

A mon père

Mon premier encadrant et formateur depuis ma naissance, l'homme le plus digne de mon respect, aucune dédicace ne peut suffire pour te remercier pour ton extrême tendresse, ton permanent soutien et tes encouragements que tu m'as offerts tout au long de ma vie. Que dieu te garde mon roi.

A ma mère

Tu m'as donné la vie, le grand amour, l'aveugle confiance et les infinis sacrifices pour réussir et arriver jusqu'au bout. Rien ne peut exprimer l'immense amour que je te porte et la profonde gratitude que je te témoigne pour tous les efforts et le dévouement que tu n'as jamais cessé de consentir pour mon instruction et mon bien-être.

Que dieu te procure la santé pour que tu demeures le flambeau illuminant notre chemin.

A mon cher frère et mon adorable sœur,

Votre affection et votre soutien m'ont été d'un grand secours tout au long de ma vie professionnelle et personnelle. Je vous dédie ce travail avec tous mes vœux de bonheur, de santé et de réussite.

A toute ma famille

A tous mes enseignants

A mes chers amis

A tous ceux que j'aime et qui m'aiment de retour.

Je dédie le couronnement de mes années d'études.

Aziz

REMERCIEMENT

C'est avec un grand plaisir que je réserve cette page en signe de gratitude et de profonde reconnaissance à tous ceux qui ont contribué de près ou de loin à la réalisation de ce travail.

Je remercie avant tout, Dieu tout puissant pour la patience, la volonté et la santé qu'il m'a consenties afin d'aboutir au terme de ce travail.

Je tiens aussi, à remercier la société EnovaRobotics pour m'avoir accueillie et donné l'opportunité d'évoluer au sein de son équipe.

J'exprime enfin ma reconnaissance à l'ISSAT, à son corps professoral, à son cadre administratif et à son personnel dont j'ai apprécié la qualité du travail, la disponibilité et le dévouement.

En outre, il m'est particulièrement agréable de remercier M.Anis Sahbani qui a bien voulu m'intégrer à son équipe de travail. Je lui exprime ma profonde gratitude pour le suivi continu et la confiance qu'il m'a accordée. De même, je tiens à exprimer ma profonde gratitude et mes sincères remerciements à mon encadrant académique MD. AMIRA BOUAOUINA ainsi qu'à mon encadrant professionnel M. ANIS AMMAR pour le temps qu'ils m'ont consacré et les réponses qu'ils ont apporté à toutes mes interrogations. Leurs encouragements me furent d'un constant appui sans lesquels la persévérance m'aurait fait défaut.

Aussi, je tiens à remercier tous mes collègues à EnovaRobotics pour les moments agréables que nous avons partagés. Travailler à vos côtés fut un réel plaisir ! Je vous souhaite à tous une bonne continuation.

Enfin, je voudrais sincèrement remercier l'ensemble des membres qui ont accepté de constituer mon jury.

Table des matières

Dédicace	1
Remerciement	2
Introduction Générale	8
1 CADRE DU PROJET	9
1.1 Introduction	10
1.2 Société ENOVA ROBOTICS	10
1.2.1 Présentation de l'entreprise	10
1.2.2 Présentation de l'équipe	11
1.2.3 Produits de ENOVA ROBOTICS	11
1.3 Aperçu du projet	13
1.3.1 Problématique	13
1.4 Travaux connexes	14
1.5 l'Etat de l'Art	15
1.5.1 Objectif de l'étude	16
1.6 Conclusion	16
2 CONTEXTE THÉORIQUE	17
2.1 Introduction	18
2.2 Intelligence artificielle	18
2.2.1 Apprentissage Machine	18
2.2.1.1 Apprentissage supervisé	19
2.2.1.2 Apprentissage non supervisé	19
2.2.1.3 Apprentissage par renforcement	19
2.2.2 Apprentissage profond	19
2.2.3 Transfert d'apprentissage et réglage fin	20
2.3 Détection d'objets avec l'apprentissage profond	20
2.3.1 Apprentissage des caractéristiques avec les CNN	21
2.3.1.1 Aperçu de l'architecture	21

2.3.1.2 Optimiseurs	23
2.3.1.3 Fonctions de perte	25
2.3.1.4 Hyperparamètres	25
2.3.2 Méthodes en deux étapes	26
2.3.2.1 R-CNN	26
2.3.2.2 Faster R-CNN	27
2.3.2.3 Mask R-CNN	28
2.3.3 Méthodes en une étape	29
2.3.3.1 Modèle YOLO	29
2.4 Conclusion	30
3 LA MÉTHODE PROPOSÉE	31
3.1 Introduction	32
3.2 Environnement de développement	32
3.2.1 Spécifications du matériel	32
3.2.2 Spécifications du logiciel	32
3.3 Ensemble de données	33
3.3.1 Traitement des données	33
3.3.2 Augmentation des données	33
3.3.3 Étiquetage des données	34
3.4 Phase de détection	34
3.4.1 YOLOv7	35
3.4.2 YOLOv8	36
3.4.3 YOLOv9	38
3.4.4 Résultats des détections	39
3.4.5 Conclusion	41
3.5 Estimation de Distance en Temps Réel	42
3.5.1 Méthodologie	42
3.6 Développement du Site Web de Détection de Palettes	44
3.6.1 Structure du Site Web	44
3.6.1.1 Page d'Accueil	44
3.6.1.2 Page de Surveillance en Temps Réel	45
3.6.2 Utilité	46
3.7 Phase de déploiement	46
3.7.1 Outils matériels	46
3.7.2 Outils logiciels	47
3.8 Conclusion	47

Conclusion générale	48
Bibliographie	49

Table des figures

1.1	Logo de l'entreprise	10
1.2	PGuard	11
1.3	Veasense	12
1.4	AGV	12
1.5	Mini-laboratoire	13
2.1	Max-Pooling VS Average Pooling	22
2.2	Régularisation par Dropout	23
2.3	Aperçu du pipeline R-CNN	26
2.4	Aperçu du pipeline Faster R-CNN	28
2.5	Aperçu du pipeline Mask R-CNN	28
2.6	Aperçu du pipeline YOLO	30
3.1	Google collab	32
3.2	Fichier data.yaml	35
3.3	Commande d'entraînement	35
3.4	Résultat d'entraînement de YOLOv7	36
3.5	Commande de détection	36
3.6	Fichier data.yaml	37
3.7	Commande d'entraînement	37
3.8	Résultat d'entraînement de YOLOv8	37
3.9	Commande de détection	38
3.10	Fichier data.yaml	38
3.11	Commande d'entraînement	38
3.12	Résultat d'entraînement de YOLOv9	39
3.13	Commande de détection	39
3.14	Résultats de détection de YOLOv7	40
3.15	Résultats de détection de YOLOv8	40
3.16	Résultats de détection de YOLOv9	41
3.17	Méthode de triangulation	43
3.18	Résultats de la détection	44

3.19	Page d'Accueil	45
3.20	Page de Surveillance en Temps Réel	45
3.21	NVIDIA Jetson Xavier NX	46
3.22	Ubuntu18.04	47

INTRODUCTION GÉNÉRALE

La vision par ordinateur a connu une véritable évolution au 21e siècle. Aujourd’hui, il est considéré comme l’épine dorsale de l’avenir autonome dans divers secteurs, notamment les transports, les soins de santé, l’agriculture, l’industrie manufacturière et bien d’autres encore.

Le lien entre la vision par ordinateur et la sécurité est devenu beaucoup plus fort en raison de l’évolution des caméras et des capteurs.

La mise en œuvre d’algorithmes de vision par ordinateur dans des machines autonomes crée un environnement de travail commun aux humains et aux robots, qui vise à faciliter les tâches et à rendre les choses beaucoup plus pratiques.

En général, la vision par ordinateur est largement utilisée dans les tâches de défense telles que la reconnaissance du terrain ennemi, l’identification automatique des ennemis dans les images, l’automatisation des mouvements des véhicules et des machines, ainsi que la recherche et le sauvetage.

À l’instar des détaillants, les entreprises ayant des exigences élevées en matière de sécurité, telles que les banques ou les casinos, peuvent bénéficier d’applications de vision par ordinateur qui leur permettent d’identifier les clients à partir de l’analyse d’images provenant de caméras de sécurité.

À un autre niveau, la vision par ordinateur est un allié puissant pour les tâches de sécurité intérieure. Elle peut être utilisée pour améliorer l’inspection des marchandises dans les ports ou pour la surveillance de lieux sensibles tels que les ambassades, les centrales électriques, les hôpitaux, les chemins de fer et les stades. L’idée principale dans ce contexte est que la vision par ordinateur ne se contente pas d’analyser et de classer les images, mais qu’elle peut aussi construire des descriptions détaillées et significatives d’une scène, en fournissant, en temps réel, des éléments clés pour les décideurs.

CHAPITRE 1

CADRE DU PROJET

Sommaire

1.1	Introduction	10
1.2	Société ENOVA ROBOTICS	10
1.2.1	Présentation de l'entreprise	10
1.2.2	Présentation de l'équipe	11
1.2.3	Produits de ENOVA ROBOTICS	11
1.3	Aperçu du projet	13
1.3.1	Problématique	13
1.4	Travaux connexes	14
1.5	l'Etat de l'Art	15
1.5.1	Objectif de l'étude	16
1.6	Conclusion	16

1.1 Introduction

Dans ce chapitre, vous trouverez une description de l'historique de l'entreprise hôte ainsi que de ses services. Nous vous donnerons également un aperçu de notre projet, de la problématique que nous souhaitons aborder et de son objectif.

1.2 Société ENOVA ROBOTICS

1.2.1 Présentation de l'entreprise

Depuis sa création par le Dr. Anis Sahbani en 2014, il n'y avait qu'un seul objectif en tête : ENOVA ROBOTICS aidera les gens dans leur vie quotidienne grâce à l'IA et à la robotique, c'est pourquoi l'équipe d'ENOVA a continué à concevoir, développer et fabriquer des robots autonomes dédiés aux secteurs de la sécurité, de la santé et de l'industrie.

Localisation :

- Rue Paul BADRE, 91220 Bretigny sur Orge, France
- Novation City, Sousse 4000
- Site web : <https://www.enovarobotics.eu/>

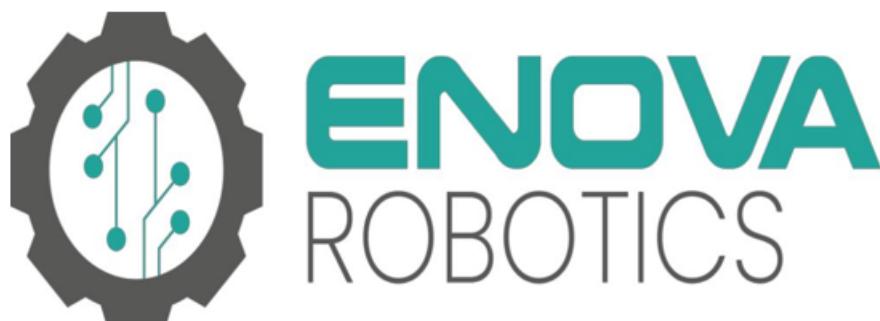


FIGURE 1.1 – Logo de l'entreprise

1.2.2 Présentation de l'équipe

ENOVA ROBOTICS dispose de 4 départements avec plus de 30 ingénieurs :

- Département R&D
- Service du matériel
- Départements des ventes et du marketing
- Département de l'administration

1.2.3 Produits de ENOVA ROBOTICS

L'entreprise utilise à la fois du matériel et des logiciels pour fournir de nouvelles solutions avancées à ses clients. Parmi ces solutions, nous pouvons citer les produits suivants :

- **PGuard** : Un robot patrouilleur qui assure la sécurité dans des zones à accès limité en effectuant des surveillances aléatoires et des interventions de levée de doute. Il est conçu pour détecter les intrusions/anomalies et déclencher des alertes. Le PGuard peut être autonome ou télécommandé en fonction des besoins du client.

Le robot P-Guard se recharge de manière autonome grâce à sa station de charge et peut fonctionner jusqu'à 8 heures sans interruption.



FIGURE 1.2 – PGuard

- **Veasense** : En raison de la situation du COVID-19 et du manque de personnel infirmier, Veasense a été créé pour faciliter le suivi des patients et la communication avec eux.



FIGURE 1.3 – Veasense

- **AGV** : chariot mobile autopiloté qui transporte une charge utile des entrepôts aux lignes de production dans un large éventail de secteurs.



FIGURE 1.4 – AGV

- **Mini-lab** : un mini-robot mobile d'intérieur conçu par des professeurs pour des professeurs. Le Mini-Lab offre un équilibre optimal entre robustesse et compétitivité économique puisqu'il est livré avec des laboratoires pré-fabriqués qui peuvent être simulés à la fois sur Matlab et Gazebo pour un coût de 1,5 million d'euros. expérience d'enseignement complète. Son architecture de contrôle open-source est basée sur le Robot Operating System (ROS).



FIGURE 1.5 – Mini-laboratoire

- **Veasense 2** : Une version beaucoup plus développée du robot Veasense dédiée à la gestion d'événements. Il offre un environnement attrayant aux participants et fournit aux sponsors de l'événement une audience captive. Les organisateurs d'événements peuvent se détendre pendant que Veasense2 s'occupe des détails.

1.3 Aperçu du projet

1.3.1 Problématique

Dans un contexte où l'efficacité opérationnelle et la sécurité sont des enjeux majeurs dans les environnements de stockage et de logistique, la détection précise et rapide des palettes par des robots chariots élévateurs autonomes pose un défi significatif. Alors que les méthodes traditionnelles de repérage et de manipulation des palettes peuvent être sujettes à des erreurs et à des retards, l'introduction de techniques d'apprentissage profond, telles que le modèle YOLO , SSD , F-CNN ouvre la voie à des solutions innovantes.

Cependant, des questions persistent quant à l'optimisation de la détection des palettes, à l'intégration efficace du modèle dans l'environnement des robots et à la garantie de performances fiables dans diverses conditions logistiques.

Ainsi, la problématique centrale de ce projet réside dans la recherche de méthodes et de stratégies permettant de développer et de déployer un modèle de détection de palettes capable d'améliorer la productivité et la sécurité des opérations de manutention automatisée dans les entrepôts et les centres de distribution.

1.4 Travaux connexes

En 2021, [1] Tsigas et Kleitsiotis ont introduit une méthode basée sur la vision pour permettre l'opération autonome et sécurisée de véhicules de manutention de palettes dans des environnements industriels peuplés d'humains, sans suivre des chemins fixes. Leur approche utilise des mesures laser et une vision monoculaire pour la détection des palettes et l'estimation de leur pose, combinant un réseau de neurones profonds pour l'isolement rapide des régions d'intérêt des palettes et une correspondance de motifs géométriques basée sur un modèle pour extraire la pose des palettes à partir des données de nuages de points.

En 2019, [2] Kita et ses collègues ont développé une méthode pour détecter et localiser des palettes sur des étagères en utilisant une caméra grand angle montée sur un véhicule, exploitant la re-projection des images pour une détection précise malgré la distorsion.

En 2018, [3] Molter et Fottner ont proposé une solution en temps réel pour la localisation de palettes utilisant des caméras 3D, permettant une détection sans marqueurs artificiels en se basant sur la détection de surfaces dans le nuage de points. Cette même année, Mohamed [4] et son équipe ont présenté une architecture utilisant des techniques d'apprentissage automatique et un télémètre laser 2D pour la détection et le suivi précis des palettes, combinant des réseaux de neurones convolutifs avec un filtre de Kalman pour la localisation.

Ces travaux représentent une variété d'approches pour résoudre le défi de la détection et de la localisation des palettes dans les environnements logistiques, chacune apportant des contributions significatives à ce domaine en constante évolution.

1.5 l'Etat de l'Art

L'évolution récente de la détection d'objets a été largement influencée par les progrès dans le domaine de l'apprentissage profond. Les réseaux de neurones convolutionnels (CNN) ont révolutionné la manière dont les objets sont détectés dans les images en permettant une représentation hiérarchique des caractéristiques visuelles. Parmi les architectures les plus remarquables, on retrouve Faster R-CNN, Single Shot Multibox Detector (SSD), et You Only Look Once (YOLO). Ces modèles se sont distingués par leur capacité à fournir une détection rapide et précise des objets dans des images de haute résolution, ce qui les rend particulièrement adaptés aux applications en temps réel telles que la robotique et la logistique.

Dans ce contexte, la détection de palettes revêt une importance particulière, car elle est cruciale pour la manipulation et le transport efficaces des marchandises dans les entrepôts et les centres de distribution. Les palettes présentent des défis uniques en termes de forme, de taille et de texture, ce qui nécessite des approches de détection spécifiques pour garantir une reconnaissance fiable dans diverses conditions.

Dans cette optique, l'utilisation de YOLO se distingue comme une solution prometteuse, offrant à la fois des performances élevées et une exécution en temps réel, ce qui en fait un choix idéal pour notre projet de développement d'un modèle de détection de palettes pour des robots chariots élévateurs autonomes.

1.5.1 Objectif de l'étude

Dans ce travail nous voulons assurer le développement et la déploiement d'un modèle de détection des palettes qui joue un rôle primordial dans la reconnaissance autonome des palettes par des robots chariots élévateurs afin d'optimiser le processus de prélèvement dans les environnements de stockage et de logistique. Le projet se concentre sur l'utilisation de techniques d'apprentissage profond pour améliorer la reconnaissance des palettes et l'estimation de la position et la distance permettant aux robots de localiser et de prélever de manière autonome les palettes.

1.6 Conclusion

Dans ce chapitre, nous avons d'abord présenté l'entreprise d'accueil, puis nous avons décrit la problématique à laquelle nous nous attaquons et notre objectif pour cette étude. Dans le chapitre suivant, nous dévoilerons le contexte théorique.

CHAPITRE 2

CONTEXTE THÉORIQUE

Sommaire

2.1	Introduction	18
2.2	Intelligence artificielle	18
2.2.1	Apprentissage Machine	18
2.2.1.1	Apprentissage supervisé	19
2.2.1.2	Apprentissage non supervisé	19
2.2.1.3	Apprentissage par renforcement	19
2.2.2	Apprentissage profond	19
2.2.3	Transfert d'apprentissage et réglage fin	20
2.3	Détection d'objets avec l'apprentissage profond	20
2.3.1	Apprentissage des caractéristiques avec les CNN	21
2.3.1.1	Aperçu de l'architecture	21
2.3.1.2	Optimiseurs	23
2.3.1.3	Fonctions de perte	25
2.3.1.4	Hyperparamètres	25
2.3.2	Méthodes en deux étapes	26
2.3.2.1	R-CNN	26
2.3.2.2	Faster R-CNN	27
2.3.2.3	Mask R-CNN	28
2.3.3	Méthodes en une étape	29
2.3.3.1	Modèle YOLO	29
2.4	Conclusion	30

2.1 Introduction

Dans ce chapitre, nous examinerons de près le contexte théorique, y compris le concept de l'intelligence artificielle et la détection d'objets.

2.2 Intelligence artificielle

L'intelligence artificielle (IA) est une manière de développer des systèmes informatiques afin d'accomplir des tâches qui nécessitent généralement l'intelligence humaine. Par conséquent, elle est considérée comme une forme d'intelligence démontrée par des machines qui imite l'intelligence naturelle des humains.

Fondamentalement, l'IA est un domaine qui associe l'informatique à de vastes ensembles de données pour résoudre des problèmes. Les objectifs de l'IA comprennent l'apprentissage, le raisonnement et la perception. Elle est utilisée dans divers secteurs, notamment la finance et les soins de santé. L'IA faible est simple et orientée vers une seule tâche, tandis que l'IA forte accomplit des tâches plus complexes qui sont semblables à celles des humains.

Dans la suite, nous présenterons généralement les principales composantes de l'intelligence artificielle : l'apprentissage automatique, l'apprentissage profond, le transfert d'apprentissage et le réglage fin.

2.2.1 Apprentissage Machine

L'apprentissage automatique est une branche de l'informatique où le système améliore sa précision en répétant les tâches à l'aide de données et d'algorithmes. De plus, il existe trois techniques d'apprentissage automatique : supervisé, non supervisé et par renforcement.

2.2.1.1 Apprentissage supervisé

Un sous-domaine de l'apprentissage automatique et de l'intelligence artificielle où l'algorithme apprend sur des données d'entrée avec des étiquettes pour produire une sortie prédéfinie. À la fin de l'entraînement, l'algorithme comprend globalement le fonctionnement des données et la relation entre l'entrée et la sortie. Cette méthode est axée sur les tâches et offre une grande précision.

2.2.1.2 Apprentissage non supervisé

Une technique d'apprentissage automatique où les données d'entrée ne sont pas étiquetées, obligeant le modèle à essayer de prédire les sorties en créant des catégories et en identifiant des motifs similaires et des structures cachées selon les données d'entrée fournies. Les algorithmes d'apprentissage non supervisé changent dynamiquement en fonction de leurs données, ce qui les rend plus adaptés aux déploiements postérieurs. Cette méthode est pilotée par les données et est principalement utilisée avec des ensembles de données massifs.

2.2.1.3 Apprentissage par renforcement

L'apprentissage par renforcement est directement inspiré de la manière dont les êtres humains apprennent des données dans leur vie. Il dispose d'un algorithme d'auto-amélioration qui utilise une stratégie d'essais et d'erreurs pour apprendre de nouveaux scénarios. Lors de l'analyse des résultats, les sorties favorables sont encouragées ou "renforcées", et les sorties non favorables sont découragées ou "pénalisées". Cette méthode apprend des erreurs et dispose d'un système de récompense directement lié à l'efficacité du résultat.

2.2.2 Apprentissage profond

L'apprentissage profond est une sous-catégorie de l'apprentissage automatique avec une structure logique d'algorithmes semblable au cerveau, appelée réseaux neuronaux artificiels. Ces algorithmes traitent des données non structurées telles que le son, le texte et les images, ce qui explique pourquoi il n'y a pas de méthode d'extraction des caractéristiques.

Au lieu de cela, l'algorithme sera entraîné à identifier lui-même les éléments influents et les hyperparamètres dans la prédiction que nous visons à réaliser. Par conséquent, il existe deux types d'algorithmes d'apprentissage profond :

- Réseaux neuronaux récurrents (RNN) pour le traitement des mots.
- Réseaux neuronaux convolutifs (CNN) pour le traitement des images.

2.2.3 Transfert d'apprentissage et réglage fin

Le transfert d'apprentissage et le réglage fin sont utilisés de manière interchangeable et représentent le processus de formation d'un réseau neuronal sur de nouvelles données en l'initialisant avec des poids pré-entraînés obtenus à partir de son entraînement sur un ensemble de données différent, souvent beaucoup plus grand, pour une nouvelle tâche liée aux données et à la tâche sur lesquelles le réseau était précédemment entraîné.

De plus, dans le transfert d'apprentissage, les dernières couches du réseau sont généralement remplacées par de nouvelles et initialisées avec des poids aléatoires, tandis que les couches restantes peuvent être figées ou rester entraînables. Apprendre à partir de zéro consiste simplement à randomiser les poids d'un réseau neuronal et à commencer le processus d'entraînement sur l'ensemble de données principal.

Le réglage fin est connu comme une approche du transfert d'apprentissage où vous apportez de petits ajustements à la sortie du modèle pour l'adapter à la nouvelle tâche et ne former que le modèle de sortie.

2.3 Détection d'objets avec l'apprentissage profond

L'apprentissage profond, comme mentionné précédemment, fonctionne principalement avec des réseaux neuronaux récurrents (RNN) ou des réseaux neuronaux convolutifs (CNN). Dans cette section, nous allons nous concentrer sur les réseaux neuronaux convolutifs.

2.3.1 Apprentissage des caractéristiques avec les CNN

2.3.1.1 Aperçu de l'architecture

Une architecture de CNN se compose généralement de 5 blocs principaux : la couche de convolution, la couche de pooling, la couche entièrement connectée , le dropout et les fonctions d'activation.

Nous commençons par **la couche de convolution**, qui est la première couche. Elle est conçue pour extraire les caractéristiques de l'image d'entrée et nous donner la carte des caractéristiques en tant que sortie. Dans cette couche, l'opération de convolution se produit entre l'image d'entrée et un filtre particulier pour obtenir des informations pertinentes sur l'image, telles que les coins et les bords.

La sortie de la carte des caractéristiques sera ensuite alimentée dans d'autres couches pour extraire des caractéristiques plus complexes de l'entrée. La couche de pooling intervient ensuite.

À ce stade, nous tentons de réduire la taille de la carte des caractéristiques convoluée. Cela est réalisé en réduisant le nombre de connexions entre les couches. Il existe plusieurs types d'opérations de pooling telles que le **Max Pooling**, le **Average Pooling** et le **Sum Pooling**.

Dans le **Max Pooling**, nous prenons l'élément le plus grand de la carte des caractéristiques.

Dans l'**Average Pooling**, nous calculons la valeur moyenne des patchs de la carte des caractéristiques extraites, puis l'utilisons pour créer une carte des caractéristiques poolées.

Dans le **Sum Pooling**, nous mesurons la valeur totale des différents patchs de la carte des caractéristiques donnée.

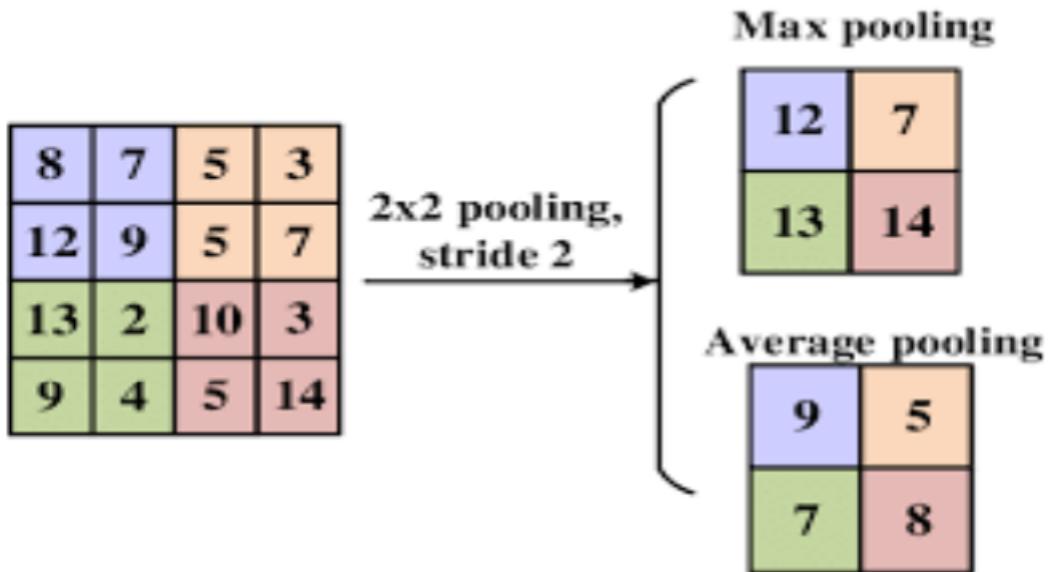


FIGURE 2.1 – Max-Pooling VS Average Pooling

La couche de pooling sert de pont entre la couche de convolution et la couche entièrement connectée. Ensuite, nous passons à la **couche entièrement connectée**. Dans cette couche, nous visons à connecter les neurones de deux couches différentes à l'aide de poids et de biais. À ce stade, le processus de classification commence à avoir lieu.

La quatrième étape est la régularisation par abandon (Dropout). Il s'agit d'une technique anti-surapprentissage qui fonctionne en coupant de manière aléatoire certains neurones et leurs connexions correspondantes. C'est un moyen extrêmement efficace d'empêcher le réseau de s'appuyer trop sur des neurones individuels et de forcer tous les neurones à apprendre à généraliser mieux. Pendant l'entraînement, le dropout est mis en œuvre en gardant un neurone actif avec une certaine probabilité ou en le mettant à zéro sinon.

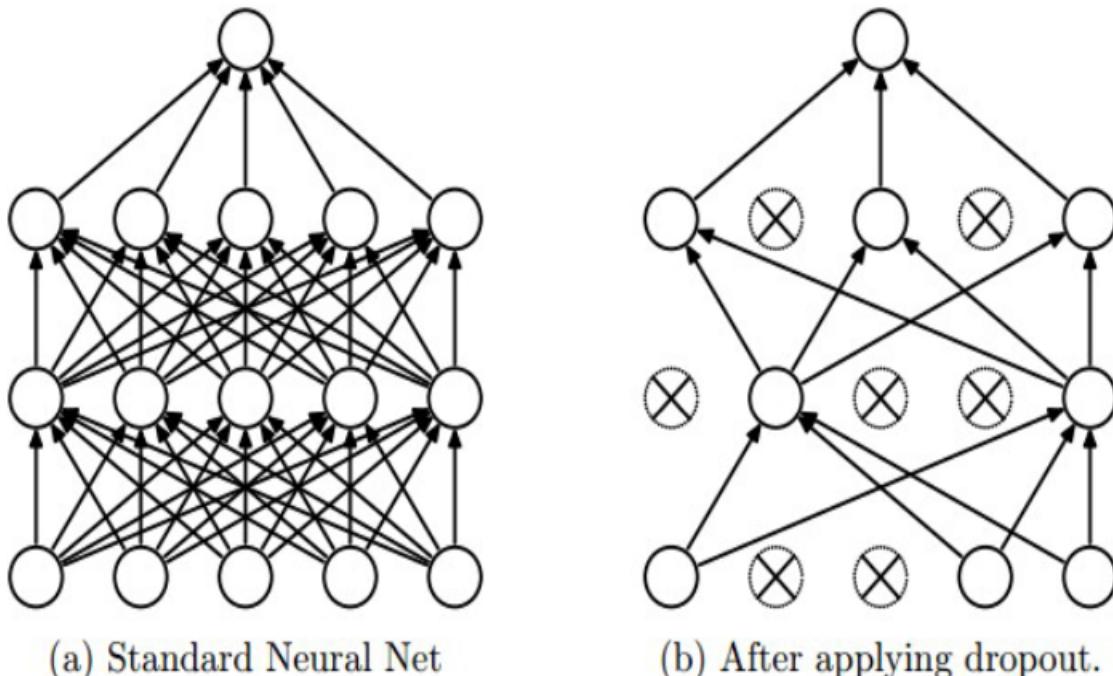


FIGURE 2.2 – Régularisation par Dropout

Enfin, nous avons les **fonctions d'activation** : il s'agit de l'un des paramètres les plus importants du modèle CNN car il apprend et approxime tout type de relation continue et complexe entre les variables du réseau. De plus, il ajoute de la non-linéarité au réseau. Parmi les fonctions d'activation, on peut citer ReLU, Softmax, tanH et les fonctions Sigmoïde.

2.3.1.2 Optimiseurs

Les optimiseurs sont des techniques ou des approches qui ajustent les caractéristiques de votre réseau neuronal, telles que les poids et le taux d'apprentissage, pour réduire les pertes. Ils aident à obtenir des résultats plus rapides. Il existe plusieurs optimiseurs tels que la descente de gradient, la descente de gradient stochastique, la descente de gradient mini-batch, Adam, etc. Ces optimiseurs ont été créés en ajoutant progressivement un terme à chaque algorithme, ce qui le rend capable de réaliser davantage de choses. Cependant, le choix du meilleur optimiseur dépend uniquement du problème que nous essayons de résoudre et de savoir s'il peut parcourir la perte pour ce problème spécifique ou non.

a) Descente de gradient :

Cet optimiseur implique de prendre de petits pas de manière itérative jusqu'à ce que nous atteignions les poids corrects.

Ces poids sont mis à jour une fois après avoir vu l'ensemble des données, ce qui signifie qu'il effectue des sauts plus importants et peut même rester à proximité de sa valeur optimale sans jamais l'atteindre réellement.

Ce problème peut être résolu en mettant à jour les paramètres plus fréquemment, c'est pourquoi nous vous présentons la descente de gradient stochastique.

b) Descente de gradient stochastique :

Cet optimiseur permet de mettre à jour les poids après avoir vu chaque point de données, ce qui signifie que nos paramètres sont mis à jour pour chaque échantillon. En étant influencé par chaque échantillon individuel, notre optimiseur SGD effectue des sauts très bruyants, ce qui le maintient loin des valeurs optimales. Pour faire un compromis entre GD et SGD, nous vous présentons la descente de gradient mini-batch.

c) Descente de gradient mini-batch :

Cet optimiseur est utilisé comme solution pour réduire le bruit de la SGD en mettant à jour les paramètres après quelques échantillons.

d) Adam :

Cet optimiseur est une combinaison de mises à jour du taux d'apprentissage pour chaque paramètre plus le momentum. Il effectue des pas de taille différente pour différents paramètres et avec un momentum pour chaque paramètre. En termes de vitesse, Adam est lent au début mais accélère avec le temps, de manière similaire au momentum. La vitesse et la précision sont des facteurs clés pour une convergence plus rapide et les raisons pour lesquelles Adam est utilisé comme optimiseur par défaut pour de nombreux projets.

2.3.1.3 Fonctions de perte

La fonction de perte sert à quantifier la différence entre le résultat hypothétique et le résultat pratique produit par votre modèle. En termes simples, la perte est utilisée pour calculer les gradients qui sont utilisés pour mettre à jour les poids du réseau et réduire le pourcentage d'erreur. La plupart des fonctions de perte intégrées peuvent être trouvées à la fois dans Keras et TensorFlow.

La perte MSE (Mean Squared Error) est utilisée pour la régression en calculant la moyenne des différences au carré entre la valeur cible et la valeur prédictive.

La perte BCE (Binary Crossentropy) est utilisée pour les tâches de classification binaire. Elle demande juste un nœud de sortie pour classer les données en 2 classes, puis passe la valeur de sortie à travers une fonction d'activation sigmoïde. La sortie est toujours dans la plage (0-1).

La perte CCE (Categorical Crossentropy) est utilisée lorsqu'une tâche de classification multi-classes se produit. Elle demande autant de nœuds de sortie que les classes données, puis passe la sortie de la couche finale à travers une activation softmax afin d'obtenir une valeur de probabilité de sortie de chaque nœud entre 0 et 1.

2.3.1.4 Hyperparamètres

Un hyperparamètre est un paramètre dont la valeur affecte et contrôle l'ensemble du processus d'apprentissage. Des exemples d'hyperparamètres sont le nombre d'échantillons, la taille du lot, le taux d'apprentissage, les époques, etc.

a) Échantillon :

Un échantillon est une seule ligne de données. Par conséquent, un ensemble de données d'entraînement contient de nombreux échantillons.

b) Taille du lot (Batch Size) :

La taille du lot est le nombre d'échantillons d'entraînement dans une passe avant et une passe arrière. Par conséquent, un ensemble de données d'entraînement peut être divisé en un ou plusieurs lots.

c) Taux d'apprentissage :

Le taux d'apprentissage d'un modèle est l'hyperparamètre qui contrôle la manière dont nous ajustons les poids du réseau par rapport au gradient de perte. Plus le taux d'apprentissage est faible, plus le déplacement le long de la pente descendante est lent.

d) Époques :

Une époque correspond à une passe complète avant et arrière de tous les échantillons d'entraînement. En termes simples, le nombre d'époques est le nombre de fois que le modèle d'apprentissage parcourra l'ensemble des données d'entraînement.

2.3.2 Méthodes en deux étapes

Les modèles multi-étapes sont une branche des détecteurs d'objets. Dans cette méthode, nous avons deux modèles : l'un extrait les régions des objets et l'autre classe et affine la localisation de l'objet. Ce type de méthode est généralement très lent mais très efficace et puissant.

Dans les sous-sections suivantes, nous aborderons l'exemple de la famille R-CNN et comment ils fonctionnent.

2.3.2.1 R-CNN

Les réseaux neuronaux convolutifs basés sur les régions (**R-CNN**) sont un ensemble de modèles d'apprentissage automatique pour la vision par ordinateur, spécifiquement pour la détection d'objets. Ils sont basés sur le mécanisme de Selective Search qui extrait les régions d'intérêt (**ROI**). Le pipeline global de R-CNN, comme décrit dans la figure ci-dessous, est composé de trois étapes :

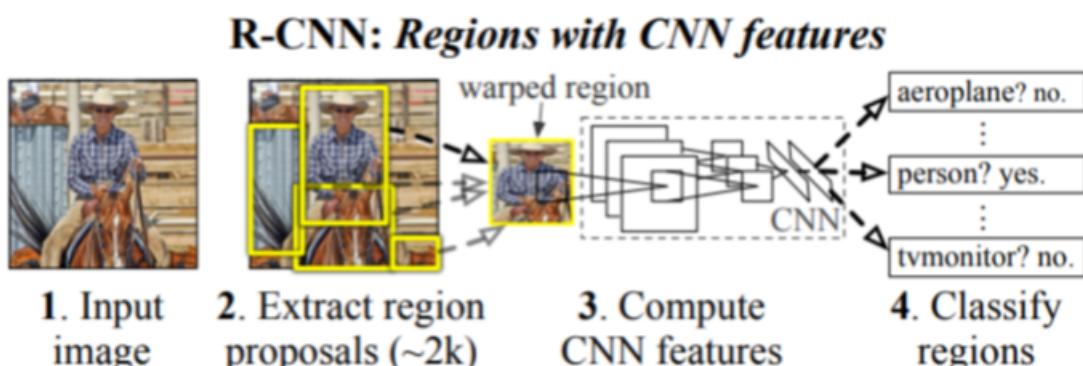


FIGURE 2.3 – Aperçu du pipeline R-CNN

1. Générer des propositions de régions : Le modèle doit dessiner des candidats d'objets dans l'image, indépendamment de la catégorie .
2. La deuxième étape est un CNN entièrement connecté qui extrait des caractéristiques de chaque région candidate.
3. La dernière étape est une couche FC qui classe les régions.

Pour résumer le processus de Selective Search, un algorithme de segmentation est appliqué à l'image d'entrée, et des boîtes englobantes de propositions de région sont dessinées en fonction de la carte de segmentation donnée. Ensuite, ces régions sont remodelées pour s'adapter à l'entrée du CNN afin que chacune d'elles passe au ConvNet. Enfin, une régression de boîte englobante (Bbox reg) est utilisée pour prédire les boîtes englobantes pour chaque région identifiée.

2.3.2.2 Faster R-CNN

Faster R-CNN est un modèle de détection d'objets qui améliore R-CNN et Fast R-CNN en combinant le modèle CNN avec un réseau de propositions de régions (**RPN**).

Le RPN et le réseau de détection échangent des caractéristiques convolutives d'image complète, permettant des propositions de régions presque gratuites. Il s'agit d'un réseau entièrement convolutionnel qui prédit les limites des objets et les scores de pertinence des objets à chaque emplacement en même temps.

Le RPN est formé de bout en bout pour générer des propositions de régions de haute qualité, que Fast R-CNN utilise pour la détection. En partageant leurs caractéristiques convolutives, RPN et Fast R-CNN sont combinés en un seul réseau : la composante RPN informe le réseau unifié où regarder.

Pour conclure, Faster R-CNN est composé de deux modules au total. Le premier module est un réseau convolutionnel entièrement profond qui fait des suggestions de région, tandis que le deuxième module est le détecteur Fast R-CNN qui utilise les régions proposées.

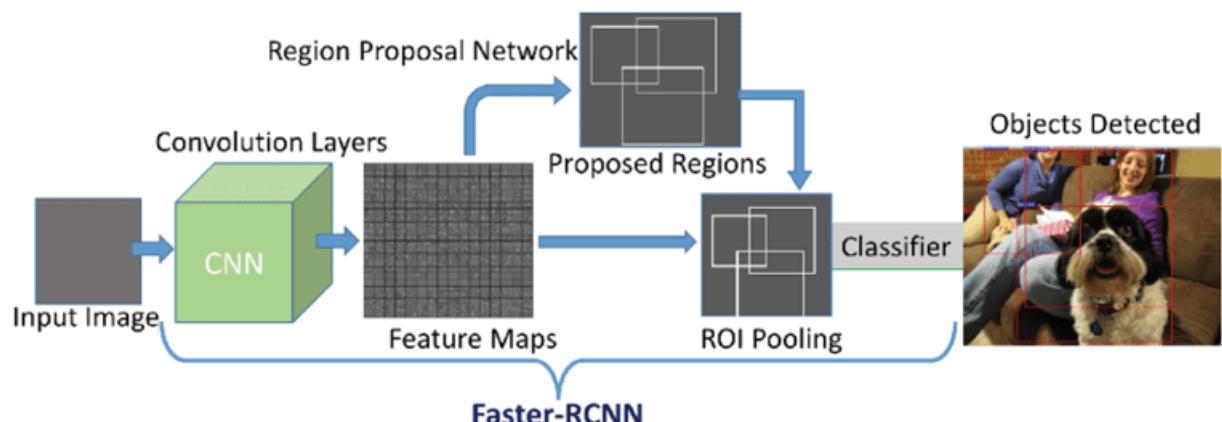


FIGURE 2.4 – Aperçu du pipeline Faster R-CNN

2.3.2.3 Mask R-CNN

Mask R-CNN est une architecture utilisée pour réaliser la segmentation d'instances, qui est une combinaison de la détection d'objets et de la segmentation sémantique. Il adopte la même procédure en deux étapes, avec un RPN identique en première étape. Dans la deuxième étape, tout en prédisant simultanément la classe et le décalage de la boîte, Mask R-CNN produit également un masque binaire pour chaque ROI. Cela contraste avec la plupart des systèmes récents, où la classification dépend de la prédiction du masque. En d'autres termes, Mask R-CNN est une combinaison de Faster R-CNN et du réseau de convolution entièrement convolutif (FCN) et utilise RoIAlign, qui préserve l'orientation spatiale des caractéristiques et ne conduit à aucune perte d'information.

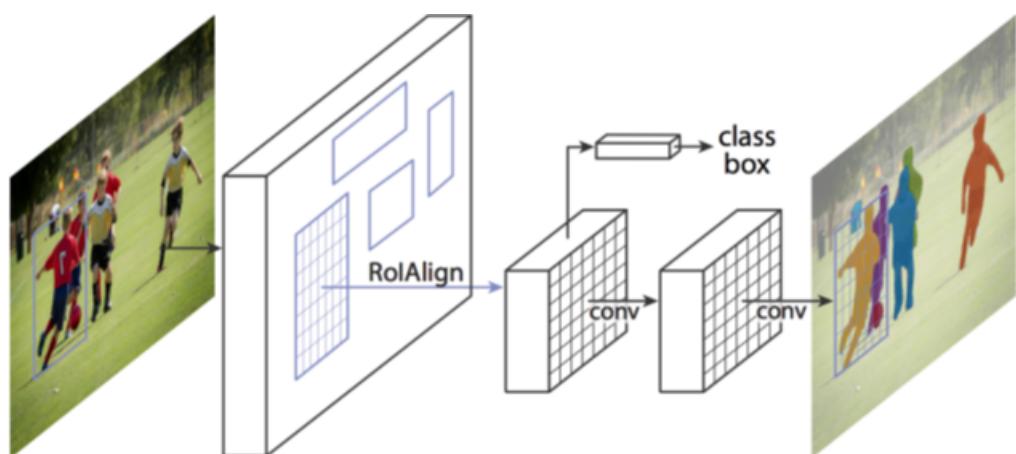


FIGURE 2.5 – Aperçu du pipeline Mask R-CNN

2.3.3 Méthodes en une étape

Dans cette méthode, l'architecture du modèle prédit directement les boîtes englobantes des objets d'une manière en une étape, tout en sautant l'étape de proposition de régions des modèles en deux étapes. En d'autres termes, il n'y a pas de tâche intermédiaire à effectuer pour obtenir la sortie.

Cela conduit à une architecture de modèle beaucoup plus simple et plus rapide.

Dans la prochaine sous-section, nous aborderons l'exemple du modèle YOLO et son fonctionnement.

2.3.3.1 Modèle YOLO

Le modèle YOLO est la première tentative de créer un détecteur d'objets en temps réel rapide, et c'est l'un des algorithmes de détection d'objets les plus précis. La détection d'objets dans YOLO est effectuée comme un problème de régression et fournit les probabilités de classe des images détectées.

L'algorithme YOLO utilise un réseau neuronal convolutionnel (CNN) pour détecter les objets. Comme son nom l'indique, l'algorithme nécessite seulement une seule propagation avant à travers un réseau neuronal pour détecter les objets. Cela signifie que YOLO examine l'image entière une seule fois au lieu d'utiliser des régions pour localiser les objets dans l'image.

Le CNN est utilisé pour prédire les différentes probabilités de classe et les boîtes englobantes simultanément, sur un ensemble limité de boîtes englobantes, ce qui permet une optimisation directe de bout en bout et une vitesse d'inférence rapide. YOLO exécute le problème de classification et de localisation sur chaque cellule en même temps. Il prend une image et la divise en une grille de $s \times s$.

Le réseau ne peut détecter qu'un seul objet, c'est-à-dire que chaque cellule de la grille est responsable de détecter un seul objet. L'objet peut se trouver dans plus d'une cellule de la grille, donc le modèle peut détecter l'objet plusieurs fois. YOLO a résolu ce problème avec la suppression non maximale.

C'est une méthode de vision par ordinateur qui détecte l'objet une seule fois et choisit la boîte englobante avec la plus haute probabilité de confiance et supprime les autres.

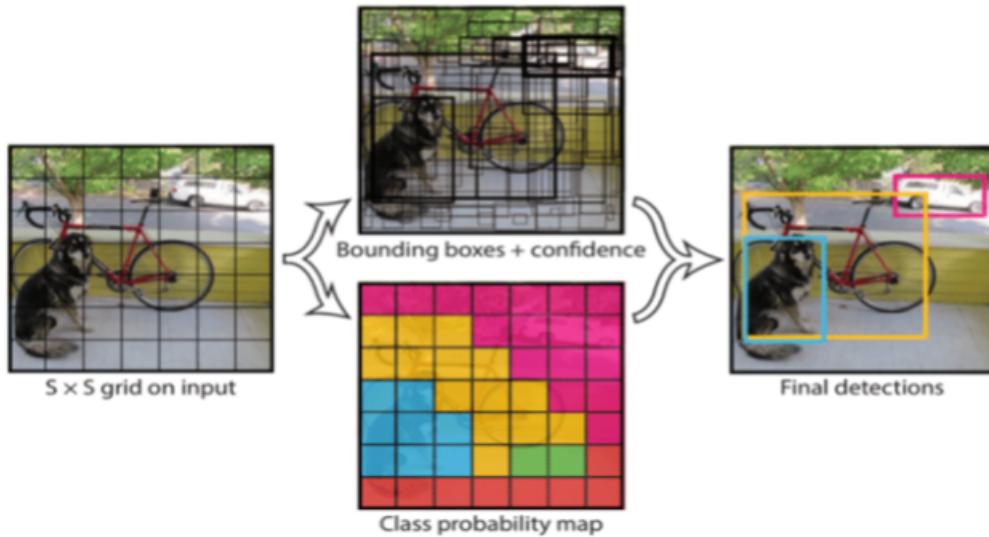


FIGURE 2.6 – Aperçu du pipeline YOLO

Chaque cellule de la grille $s \times s$ prédit des boîtes englobantes B. Le modèle produit un score de confiance pour chaque boîte englobante indiquant la probabilité que la cellule contienne un objet. Le score pour chaque boîte englobante ne classe pas le type d'objet ni ne sait quel objet correspond à la boîte englobante, il donne simplement une valeur de confiance ou une valeur de probabilité de la qualité des boîtes englobantes entourant un objet dans chaque cellule. C'est l'intersection sur l'union (IOU) entre la boîte prédictive et la boîte détectée, définie comme :

$$\text{Le score de confiance} = \text{PR(objet)} \times \text{IOU}$$

2.4 Conclusion

Dans ce chapitre, nous avons présenté le contexte académique et quelques notions de base pour mieux comprendre le contexte dans lequel nous travaillons. Le chapitre suivant se concentrera en détail sur la méthode que nous proposons.

CHAPITRE 3

LA MÉTHODE PROPOSÉE

Sommaire

3.1	Introduction	32
3.2	Environnement de développement	32
3.2.1	Spécifications du matériel	32
3.2.2	Spécifications du logiciel	32
3.3	Ensemble de données	33
3.3.1	Traitemennt des données	33
3.3.2	Augmentation des données	33
3.3.3	Étiquetage des données	34
3.4	Phase de détection	34
3.4.1	YOLOv7	35
3.4.2	YOLOv8	36
3.4.3	YOLOv9	38
3.4.4	Résultats des détections	39
3.4.5	Conclusion	41
3.5	Estimation de Distance en Temps Réel	42
3.5.1	Méthodologie	42
3.6	Développement du Site Web de Détection de Palettes	44
3.6.1	Structure du Site Web	44
3.6.1.1	Page d'Accueil	44
3.6.1.2	Page de Surveillance en Temps Réel	45
3.6.2	Utilité	46
3.7	Phase de déploiement	46
3.7.1	Outils matériels	46
3.7.2	Outils logiciels	47
3.8	Conclusion	47

3.1 Introduction

Dans ce chapitre, nous découvrirons la méthode proposée dans toutes ses dimensions et ses résultats, de l'environnement de travail à la phase de déploiement.

3.2 Environnement de développement

Dans cette section , nous allons dresser la liste des spécifications matérielles et logicielles nécessaires à ce travail.

3.2.1 Spécifications du matériel

Nous mentionnons des informations sur les composants internes de l'appareil tels que le type et la taille du GPU/CPU, la quantité de mémoire nécessaire et le type de stockage.

Les caractéristiques nécessaires à l'élaboration de notre modèle sont les suivantes :

- CPU : Intel Core i7-7500U
- GPU : NVIDIA GeForce GTX3050
- RAM : 16 Go
- Stockage : 1To disque dur

3.2.2 Spécifications du logiciel

- Google Colab :

Google Colab est un environnement de bloc-notes Jupyter en ligne gratuit qui offre des GPU et des TPU gratuits. Vous pouvez utiliser Colab pour écrire et exécuter du code, enregistrer et partager vos carnets et accéder à votre Google Drive pour une expérience de codage plus riche.



FIGURE 3.1 – Google collab

3.3 Ensemble de données

Un ensemble de données est un ensemble d'images utilisées pour la formation et le test du modèle. L'ensemble de données que j'utilise est 'pallet' .

3.3.1 Traitement des données

Dans l'apprentissage automatique, le traitement des données est la tâche qui consiste à préparer les données pour les utiliser dans la formation d'un modèle d'apprentissage automatique. Nous avons donc utilisé des techniques telles que l'augmentation et l'étiquetage des données à l'aide de l'outil Roboflow.

Nous avons commencé avec 2500 images, puis, après la procédure de traitement des données, nous avons obtenu 3900 images réparties comme suit :

- Train : 2.7K images.
- Validation : 780 images .
- Test : 390 images.

De plus, pour s'assurer que toutes les images ont la même taille, nous les avons redimensionnées à 640x640. Nous avons utilisé la plateforme Roboflow qui assure à la fois l'étiquetage et l'augmentation des données.

3.3.2 Augmentation des données

Nous avons appliqué l'augmentation des données à notre ensemble de données afin d'en accroître la taille. Le processus consiste à générer 3 nouveaux échantillons de données à partir de chaque échantillon donné. Il s'agit d'une méthode courante utilisée dans l'apprentissage automatique qui permet d'améliorer les performances de notre modèle en augmentant le nombre de données à partir desquelles il peut apprendre.

Dans ce projet, et afin d'obtenir un ensemble de données diverses, nous avons utilisé les filtres d'augmentation suivants :

- Rotation : Entre - 15° et +15°
- Échelle de gris : Appliquée à 20% des images du train
- Luminosité : entre -34% et +34%

3.3.3 Étiquetage des données

Pour que les données soient utiles, elles doivent être correctement étiquetées. Ce processus est connu sous le nom d'étiquetage des données et constitue une partie essentielle de tout projet axé sur les données. Avec l'aide de Roboflow, nous avons assuré l'annotation de toutes nos images. Les boîtes de délimitation que nous avons dessinées sont celles qui nous aideront à localiser les objets détectés. Nos étiquettes sont les termes qui nous aideront à classifier les états. Nous avons donc 1 étiquette : pallet.

À la fin de ce processus, nous obtenons un fichier .txt qui porte le même nom que l'image correspondante et qui contient les annotations de la classe et de la boîte englobante dans un format YOLO.

3.4 Phase de détection

À ce stade, nos données sont prêtes à être intégrées dans un modèle et à être entraînées. Nous avons essayé de mettre en œuvre différents modèles et de visualiser les résultats dans le but de choisir le modèle approprié.

- YOLO (v9, v8 et v7) :

Les outils logiciels nécessaires à ce modèle sont les suivants :

- Cadre de travail : PyTorch + Python 3.8
- Bibliothèque VC : OpenCV 4
- Environnement : Google Colab

3.4.1 YOLOv7

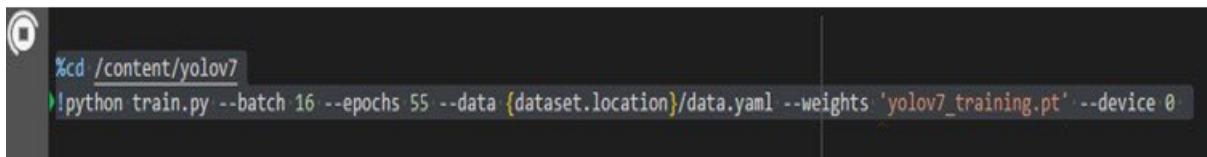
Le modèle YOLOv7 apporte des améliorations significatives en termes de précision et de rapidité par rapport à ses prédecesseurs. Pour l'entraîner, nous devons configurer un fichier YAML spécifique contenant les chemins vers les images d'entraînement et de validation, ainsi que le nombre de classes et leurs étiquettes.



```
! data.yaml x
C: > Users > ASUS-Gamer > Downloads > yolo > yolov7-20240412T224456Z-001 > yolov7 > yolov7 > Deepstream-model-1 > ! data.yaml
1 train: /content/gdrive/MyDrive/yolov7/yolov7/Deepstream-model-1/train/images
2 val: /content/gdrive/MyDrive/yolov7/yolov7/Deepstream-model-1/valid/images
3 test: /content/gdrive/MyDrive/yolov7/yolov7/Deepstream-model-1/test/images
4
5 nc: 1
6 names: ['palet']
7
8 roboflow:
9   workspace: dung-tien-dj5fq
10  project: deepstream-model
11  version: 1
12  license: CC BY 4.0
13  url: https://universe.roboflow.com/dung-tien-dj5fq/deepstream-model/dataset/1
```

FIGURE 3.2 – Fichier data.yaml

Ensuite, nous lançons la commande d'entraînement en définissant le batch size, le nombre d'époques et notre fichier YAML personnalisé.



```
%cd /content/yolov7
!python train.py --batch 16 --epochs 55 --data {dataset.location}/data.yaml --weights 'yolov7_training.pt' --device 0
```

FIGURE 3.3 – Commande d'entraînement

Une fois que le modèle est entraîné, il est évalué en utilisant des données de validation pour mesurer sa performance.

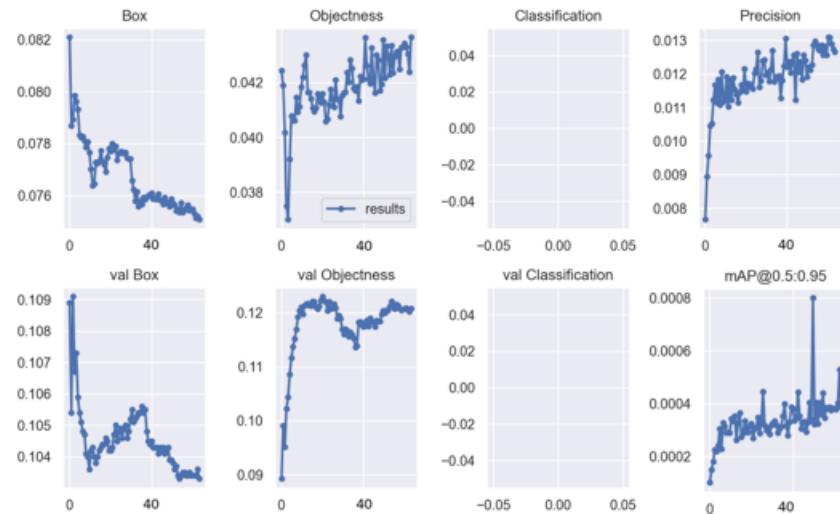


FIGURE 3.4 – Résultat d’entraînement de YOLOv7

Après l’entraînement, un fichier de poids best.pt est généré pour la phase de test, où les images de test sont assignées à la source et le seuil de confiance est fixé à 0,25.

```
!python detect.py --weights runs/train/exp/weights/best.pt --conf 0.25 --source {dataset.location}/test/images
```

FIGURE 3.5 – Commande de détection

3.4.2 YOLOv8

Le modèle YOLOv8 se distingue par son équilibre optimal entre précision, vitesse et maturité technologique. Par rapport à YOLOv7, YOLOv8 réduit les faux positifs grâce à des optimisations avancées et améliore la robustesse des détections.

Les étapes d’entraînement sont similaires à celles de YOLOv7, nécessitant un fichier YAML personnalisé et la commande d’entraînement correspondante.

```

! data.yaml ×
C: > ASUS-Gamer > Downloads > yolo > yolov8_collab_50epoch-20240405T151224Z-001 > yolov8_collab_50epoch > De
1 test: ../test/images
2 train: ../train/images
3 val: ../valid/images
4 names:
5 - palet
6 nc: 1
7 roboflow:
8 license: CC BY-NC-ND
9 project: de Follow link \(ctrl + click\)
10 url: https://universe.roboflow.com/project/deepstream-model/dataset/1
11 version: 1
12 workspace: project
13

```

FIGURE 3.6 – Fichier data.yaml

```

!yolo task=detect mode=train model=yolov8s.pt data={dataset.location}/data.yaml epochs=50 imgs=800 plots=True

```

FIGURE 3.7 – Commande d'entraînement

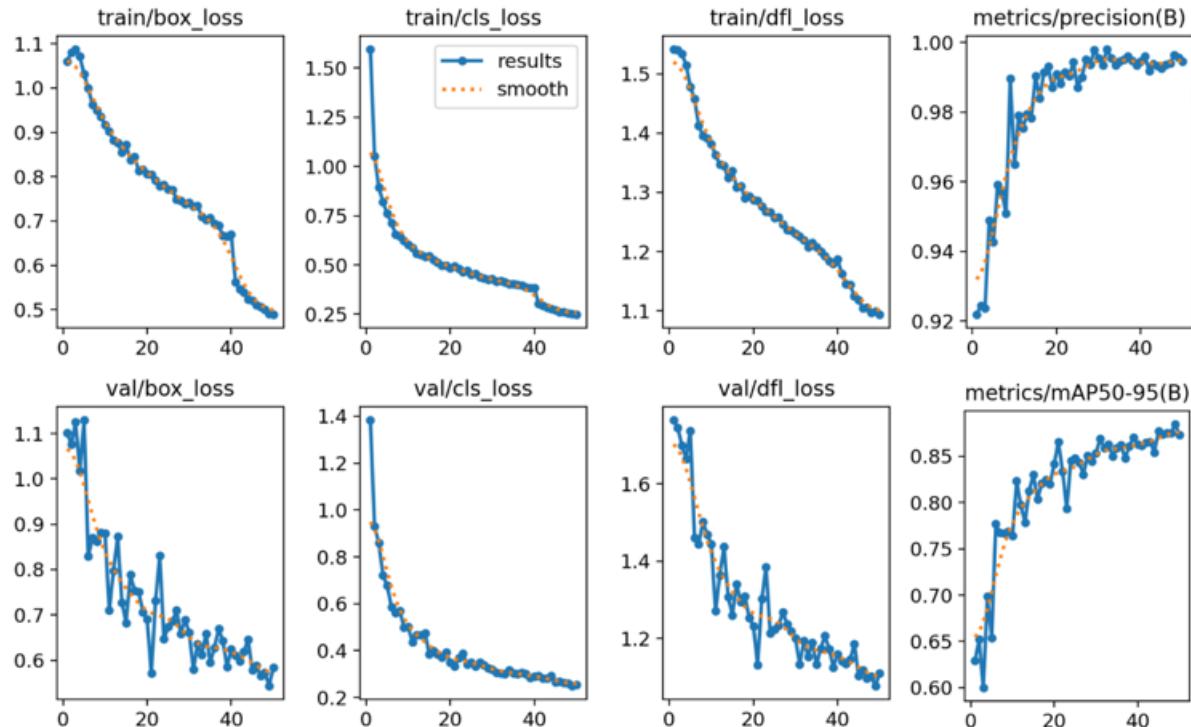


FIGURE 3.8 – Résultat d'entraînement de YOLOv8

Le fichier de poids best.pt obtenu est ensuite utilisé pour les tests, assurant des détections plus précises et fiables.

```
!yolo task=detect mode=predict model={HOME}/runs/detect/train/weights/best.pt conf=0.25 source={dataset.location}/test/images save=True
```

FIGURE 3.9 – Commande de détection

3.4.3 YOLOv9

YOLOv9, la version la plus récente, introduit de nouvelles fonctionnalités mais peut encore souffrir de bugs et nécessiter des ajustements supplémentaires.

Pour entraîner YOLOv9, nous suivons des étapes analogues aux versions précédentes en adaptant le fichier YAML et en lançant l'entraînement.

```
! data.yaml ×
C: > Users > ASUS-Gamer > Downloads > yolo > yolov9-20240415T221506Z-001 > yolov9 > yolov9 > Deepstream-model-1 > ! data.yaml
  1   test: Deepstream-model-1/test/images
  2   train: Deepstream-model-1/train/images
  3   val: Deepstream-model-1/valid/images
  4   names:
  5     - palet
  6   nc: 1
  7   roboflow:
  8     license: CC BY 4.0
  9     project: deepstream-model
10     url: https://universe.roboflow.com/dung-tien-dj5fq/deepstream-model/dataset/1
11     version: 1
12     workspace: dung-tien-dj5fq
13
```

FIGURE 3.10 – Fichier data.yaml

```
!python train_dual.py \
--batch 8 --epochs 50 --img 640 --device 0 --min-items 0 --close-mosaic 15 \
--data /content/drive/MyDrive/9/yolov9/Deepstream-model-1/data.yaml \
--weights /content/drive/MyDrive/9/yolov9/yolov9-c.pt \
--cfg /content/drive/MyDrive/9/yolov9/models/detect/yolov9-c.yaml \
--hyp hyp.scratch-high.yaml
```

FIGURE 3.11 – Commande d'entraînement

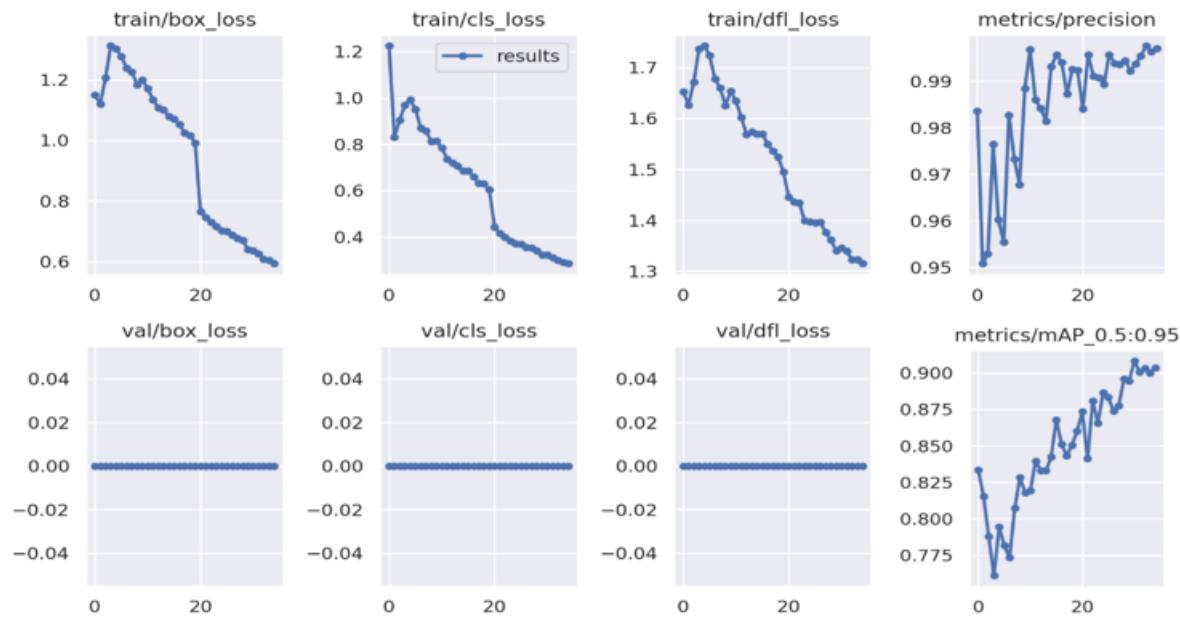


FIGURE 3.12 – Résultat d'entraînement de YOLOv9

Malgré ses innovations, YOLOv9 peut ne pas offrir un avantage décisif en termes de performance globale sur certains datasets spécifiques par rapport à YOLOv8. Après l'entraînement, le fichier de poids best.pt est utilisé pour tester le modèle.

```
!python detect.py \
--img 640 --conf 0.25 --device 0 \
--weights {HOME}/yolov9/runs/train/exp/weights/best.pt \
--source {dataset.location}/test/images
```

FIGURE 3.13 – Commande de détection

3.4.4 Résultats des détections

a) YOLOv7 :

YOLOv7 n'a pas réussi à effectuer de détection sur les images de test. Malgré sa réputation de rapidité et de précision dans de nombreux autres contextes, il n'a pas su répondre aux exigences de notre dataset. Cela met en lumière ses limitations dans certaines conditions d'entraînement et de test spécifiques à notre projet de détection de palettes.



FIGURE 3.14 – Résultats de détection de YOLOv7

b) YOLOv8 :

En revanche, YOLOv8 a démontré un excellent équilibre entre précision, vitesse et robustesse technologique. Grâce à des optimisations avancées, YOLOv8 a réussi à réduire considérablement les faux positifs et à améliorer la précision des détections. Cela a été particulièrement bénéfique pour notre projet, où la précision de la détection des palettes est cruciale. YOLOv8 a non seulement détecté les palettes de manière fiable, mais il l'a fait rapidement, ce qui est essentiel pour des applications en temps réel.



FIGURE 3.15 – Résultats de détection de YOLOv8

c) YOLOv9 :

YOLOv9, bien qu'il introduise des fonctionnalités et des améliorations prometteuses, a montré des détections imprécises. Ce manque de précision est probablement dû à son stade de développement plus récent, nécessitant encore des ajustements et des optimisations pour atteindre une stabilité et une fiabilité comparables à celles de ses prédecesseurs. Bien que YOLOv9 présente des innovations intéressantes, son immaturité technologique le rend moins fiable pour une application immédiate dans des environnements exigeants comme le nôtre.

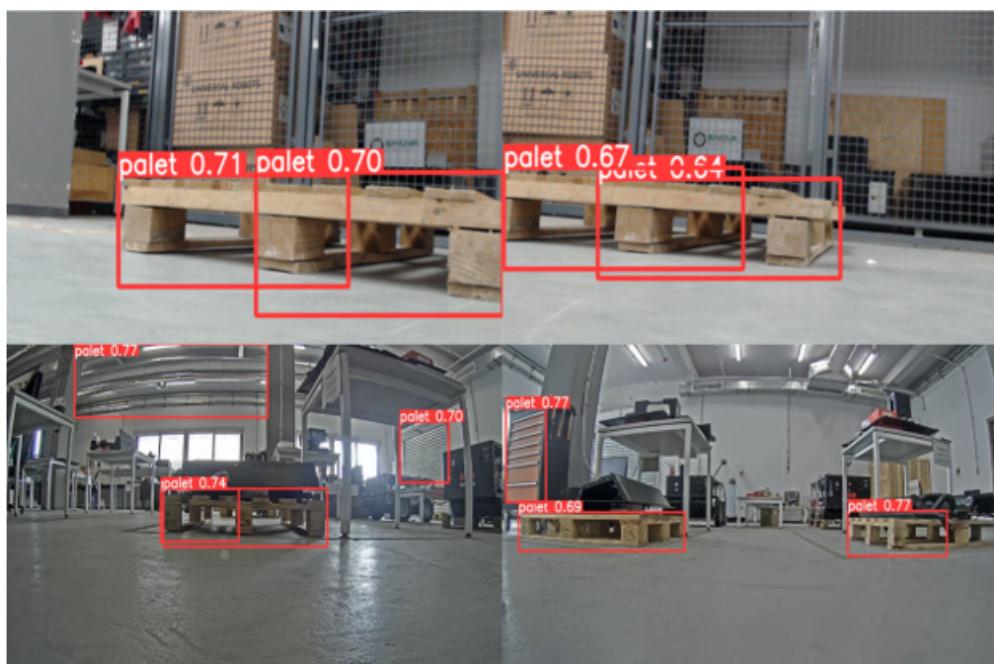


FIGURE 3.16 – Résultats de détection de YOLOv9

3.4.5 Conclusion

En termes de détection d'objets en temps réel, YOLOv8 se démarque nettement par rapport à YOLOv7 et YOLOv9. YOLOv7 n'a effectué aucune détection sur les images de test, tandis que YOLOv9 a produit des résultats imprécis. Ces trois modèles ont été entraînés sur le même ensemble de donnée et avec le même nombre d'époques, ce qui met en évidence la supériorité de YOLOv8. Grâce à ses optimisations avancées, YOLOv8 offre des performances supérieures avec une meilleure précision et une plus grande maturité technologique, répondant ainsi efficacement aux besoins de notre projet de détection de palettes. YOLOv8 se positionne donc comme le choix optimal pour des applications nécessitant une détection d'objets rapide et fiable.

3.5 Estimation de Distance en Temps Réel

Pour estimer la distance entre le camera qui est intégrer sur le robot et le palette pour bien le localiser je fournis un code en python en etu méthode de triangulation.

3.5.1 Méthodologie

1. Chargement du Modèle :

Charger le modèle YOLO avec mon propre poids.

2. Capture Vidéo en Temps Réel :

Nous avons configuré la capture vidéo en utilisant la caméra intégrée du système. Cette configuration permet de traiter les flux vidéo en temps réel, ce qui est essentiel pour des applications pratiques comme la surveillance ou l'automatisation industrielle.

3. Détection des Objets :

Chaque frame capturé par la caméra est analysé par le modèle YOLO. Le modèle génère des boîtes englobantes autour des objets détectés avec une indication de la classe de l'objet et une mesure de confiance. Un seuil de confiance est appliqué pour filtrer les détections les moins fiables.

4. Estimation de la Distance :

Pour estimer la distance entre la caméra et les palettes détectées, nous utilisons une méthode de triangulation. Cette méthode se base sur la taille connue de la palette (hauteur en millimètres) et la largeur de la boîte englobante en pixels. Voici les étapes détaillées :

-La largeur de la boîte englobante est obtenue à partir des coordonnées des coins de la boîte fournies par le modèle YOLO. Les coordonnées (x_{min} , y_{min}) représentent le coin supérieur gauche et (x_{max} , y_{max}) le coin inférieur droit de la boîte englobante.

La largeur en pixels est calculée comme suit :

$$\text{Largeur de la boîte en pixels} = x_{\text{max}} - x_{\text{min}}$$

-Longueur Focale :

La longueur focale en pixels est une valeur calibrée qui dépend de la configuration de la caméra et de la distance de référence utilisée pour la calibration.

La calibration peut être effectuée en mesurant la taille d'un objet de référence connu (comme une palette) à une distance fixe. En utilisant cette distance et les dimensions de l'objet en pixels dans l'image, la longueur focale peut être calculée.

La formule utilisée pour calculer la distance entre la caméra et l'objet est :

$$\text{Distance} = \frac{\text{Hauteur réelle de la palette} \times \text{Longueur focale}}{\text{Largeur de la boîte en pixels}}$$

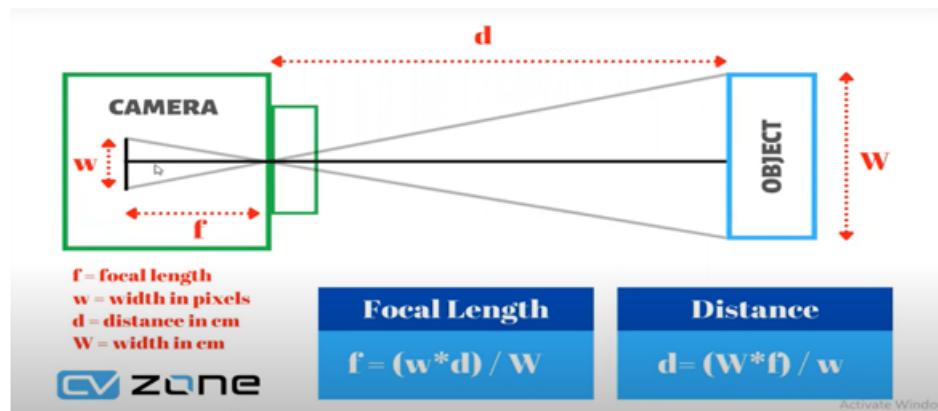


FIGURE 3.17 – Méthode de triangulation

5. Affichage des Résultats :

Les résultats de la détection, y compris les boîtes englobantes et les distances estimées, sont annotés sur les frames vidéo. Ces frames annotés sont ensuite affichées en temps réel, fournissant une visualisation claire des détections et des distances.



FIGURE 3.18 – Résultats de la détection

3.6 Développement du Site Web de Détection de Palettes

Ce site web offrant une interface utilisateur intuitive pour la surveillance et la détection des palettes en temps réel, utilisant la technologie YOLO (You Only Look Once).

3.6.1 Structure du Site Web

Le site web est composé de deux pages principales :

3.6.1.1 Page d'Accueil

Cette page affiche les icônes de tous les robots connectés. Chaque icône représente un robot spécifique et est accompagnée d'une image et d'une description. La page est conçue pour offrir une vue d'ensemble des robots disponibles pour la surveillance.

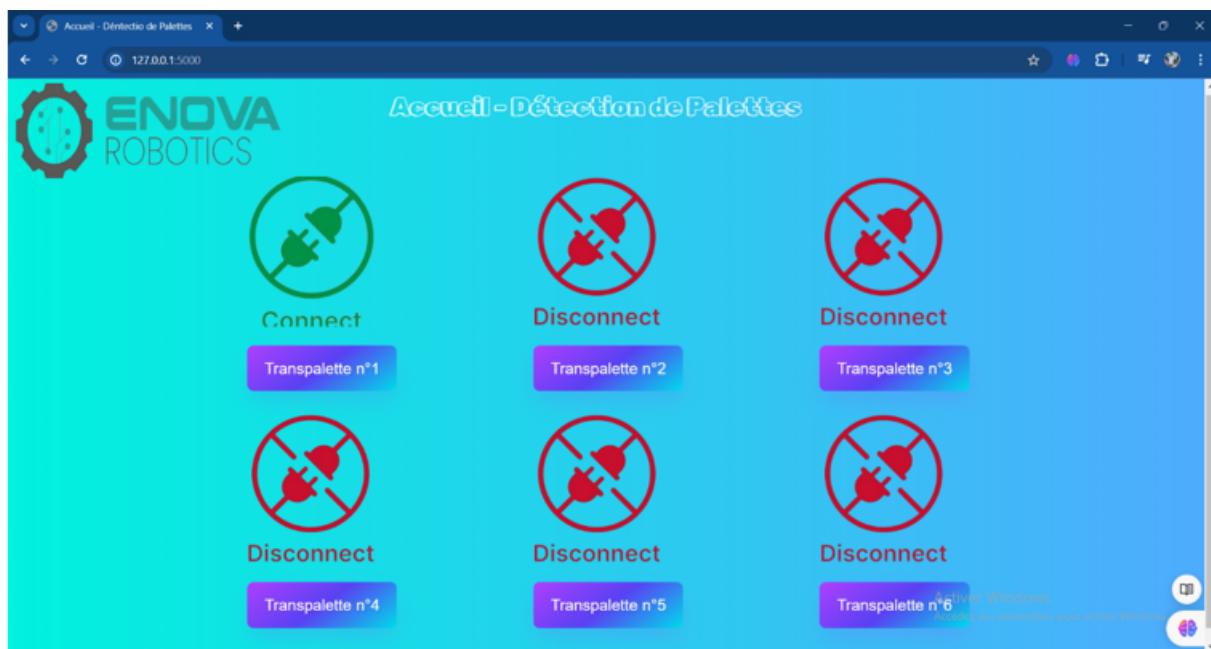


FIGURE 3.19 – Page d’Accueil

3.6.1.2 Page de Surveillance en Temps Réel

Lorsqu'un utilisateur clique sur l'icône d'un robot sur la page d'accueil, il est redirigé vers une page dédiée à ce robot. Cette page affiche un flux vidéo en temps réel avec la détection des palettes, permettant à l'utilisateur de surveiller et d'analyser les données en direct.

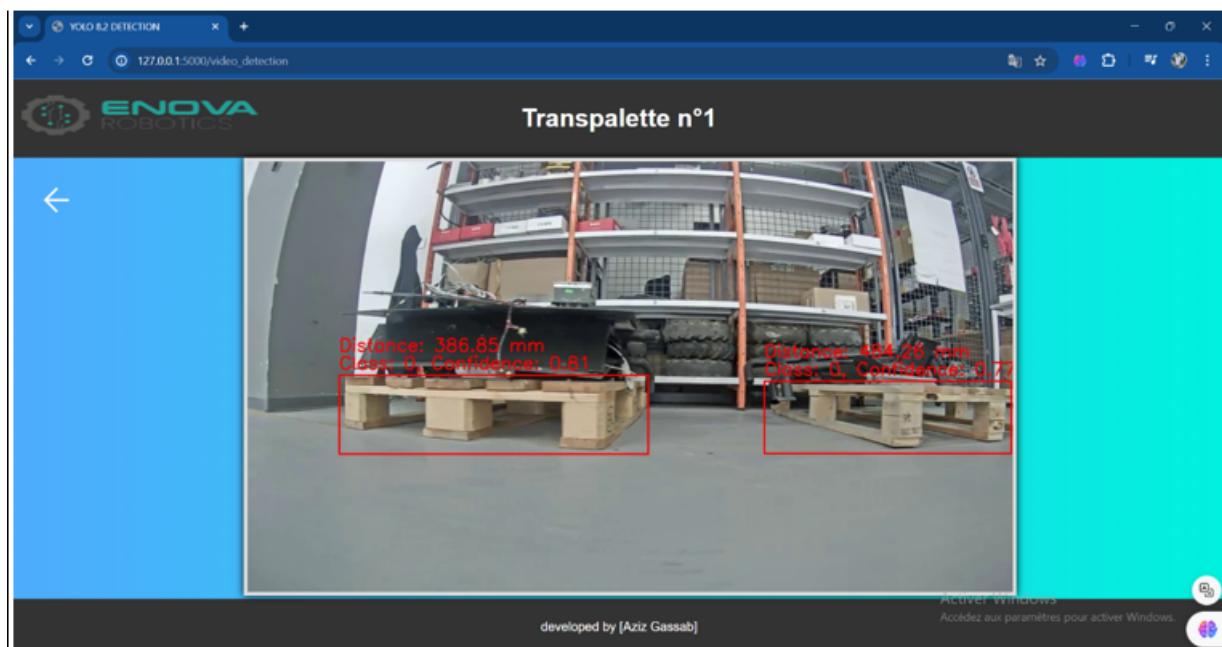


FIGURE 3.20 – Page de Surveillance en Temps Réel

3.6.2 Utilité

Ce projet répond à plusieurs besoins critiques dans la gestion logistique :

- Accessibilité** : Permettre la surveillance des palettes depuis n'importe quel appareil connecté.
- Efficacité** : Assurer une détection rapide et précise grâce à l'utilisation de YOLO.
- Simplicité d'utilisation** : Offrir une interface utilisateur intuitive et facile à naviguer, avec une transition fluide entre la vue d'ensemble des robots et la surveillance en temps réel.

3.7 Phase de déploiement

À ce stade, notre modèle est entrainé et testé avec nos données et les résultats des tests sont assez précis. En d'autres termes, notre modèle est prêt à être déployé sur la carte Nvidia Jetson Xavier nx. Le processus d'inférence est basé sur deux éléments : Le matériel et le logiciel.

3.7.1 Outils matériels

Kit de développement NVIDIA Jetson Xavier NX :

Le kit de développement NVIDIA Jetson Xavier NX est une puissante plate-forme informatique de la taille d'une carte de crédit, destinée à la création de systèmes embarqués hautes performances. Le kit comprend tout ce dont vous avez besoin pour démarrer, notamment un module Jetson Xavier NX avec 8 Go de RAM, une carte SD de 64 Go, un bloc d'alimentation et un câble USB.

Vous pouvez également démarrer avec le kit de développement Jetson Xavier NX en utilisant le kit SDK JetPack inclus, qui comprend une large gamme d'outils et de bibliothèques pour développer, déboguer et déployer vos applications.



FIGURE 3.21 – NVIDIA Jetson Xavier NX

3.7.2 Outils logiciels

- Ubuntu :

Ubuntu est un système d'exploitation Linux basé sur Debian qui a été créé dans l'intention d'être une distribution Linux facile à utiliser et adaptée aux débutants. C'est l'une des distributions Linux les plus populaires et elle est utilisée par des millions de personnes dans le monde. En fait, Ubuntu est communément connu parmi les développeurs de logiciels, où il est une plateforme attrayante pour une gamme variée d'applications, y compris la robotique, l'intelligence artificiel et les dispositifs embarqués . La version d'ubuntu utilisée dans ce projet est donc Ubuntu 18.04.



FIGURE 3.22 – Ubuntu18.04

3.8 Conclusion

Dans ce chapitre, nous avons présenté les étapes de base de la création d'un ensemble de données pour la formation d'un modèle YOLO et les étapes pour l'estimation de distance. Nous avons ensuite testé les modèles proposés et choisi le modèle YOLOv8 à déployer dans la carte Jetson Xavier NX.

CONCLUSION GÉNÉRALE

Pour conclure avec ce projet de fin d'études, qui nous a montré qu'un technicien peut apporter son expertise dans l'innovation et le respect de l'environnement et de la réglementation. Avant d'imposer des solutions, le technicien doit se tourner vers le demandeur, pour arriver à la solution de manière structurée. En effet, l'objectif de ce projet est de répondre au cahier des charges proposé par la société ENOVA ROBOTICS.

Au cours de ce projet, cette réalisation m'a incité à utiliser toutes mes connaissances et compétences en tant qu'étudiant et m'a permis de comprendre les difficultés que je pourrais rencontrer au travail tout en prenant des initiatives personnelles. Ce projet de fin d'études a été très intéressant pour moi dans l'ensemble. Il m'a permis de découvrir le monde professionnel et de transformer les connaissances de base requises à l'école en compétences professionnelles.

Ainsi, il m'a permis d'approfondir mes connaissances en vision par ordinateur. D'autre part, il offre l'opportunité de maîtriser de nouveaux outils logiciels puissants tels que Ubuntu 18.04 et de se familiariser avec la carte NVIDIA Jetson qui présente un outil crédible pour les développements futurs.

BIBLIOGRAPHIE

- [1] Efthimios Tsigas, Ioannis Kleitsiotis, Ioannis Kostavelis, Andreas Kargakos, Dimitris Giakoumis, Marc Bosch-Jorge, Raquel Julia Ros, Rafa López Tarazón, Spyridon Likothanassis, and Dimitrios Tzovaras. Pallet detection and docking strategy for autonomous pallet truck agv operation. In 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 3444–3451, 2021.
- [2] Yasuyo Kita, Ryuichi Takase, Tatsuya Komuro, Norihiko Kato, and Nobuyuki Kita. Detection and localization of pallets on shelves using a wide-angle camera. In 2019 19th International Conference on Advanced Robotics (ICAR), pages 785–792, 2019.
- [3] B. Molter and J. Fottner, "Real-time Pallet Localization with 3D Camera Technology for Forklifts in Logistic Environments," 2018 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI), Singapore, 2018, pp. 297-302, doi : 10.1109/SOLI.2018.8476740.
- [4] Mohamed, I.S., Capitanelli, A., Mastrogiovanni, F. et al. Detection, localisation and tracking of pallets using machine learning techniques and 2D range data. Neural Comput & Applic 32, 8811–8828 (2020).