Plan de Développement

1. Introduction Générale

1.1 Contexte

Le projet d'intelligence artificielle porte sur la programmation d'un robot Mindstorms EV3 afin qu'il puisse détecter, manipuler et transporter des palets de manière autonome. La programmation sera effectuée en Java via l'IDE Eclipse, en utilisant la librairie LeJOS.
En fin de semestre, les divers groupes s'affrontent lors de plusieurs matchs. L'équipe qui parviendra à déplacer le maximum de palets dans le temps imparti sera désignée gagnante.

1.2 Objectif du Plan

Ce plan a pour objet de spécifier les différentes étapes du développement de ce projet en définissant clairement les tâches à accomplir et leur répartition dans le temps. De plus, il offre aussi aux différents membres du groupe ainsi qu'au commanditaire une vision globale de l'avancement du projet, facilitant ainsi la coordination et le suivi du travail.

2. Analyse Préliminaire et Ressources

2.1 Ressources Matérielles

Nous disposons d'un robot EV3 LEGO Mindstorms sur lequel nous implémentons un programme permettant d'interagir avec le robot de manière asynchrone. Nous avons également accès à un terrain pour tester notre robot.
Sélection des capteurs EV3 utilisés dans le programme :

- Moteurs / Capteurs :
  - Roues :
    - Droite (Port M.A) : responsable de la rotation de la roue droite.
    - Gauche (Port M.B) : responsable de la rotation de la roue gauche.
  - Pinces (Port M.D) :utilisé pour ouvrir et fermer les pinces du robot.

- Ultrason (Port S.4):utilisé pour mesurer la distance entre le robot et les objets environnants.
- Toucher (Port S.3) : utilisé pour détecter les pressions ou les contacts physiques.
- Détecteur de couleur (Port S.2) :utilisé pour identifier différentes couleurs et intensités lumineuses.

2.2 Contraintes Techniques

Les composants du robot comportent certaines limites qu'il faut prendre en compte lors de la réalisation :

Le capteur d'ultrasons renvoie la distance séparant le robot et l'objet le plus proche, toutefois il est ne précise si cet objet est un palet, un autre robot ou un mur.
De plus, le palet étant petit, il ne peut le détecter que lorsque celui-ci se trouve à une distance comprise entre 32,6 et 107 cm et dans son champ division qui correspond à environ un angle d'environ 30°.

3. Objectifs Techniques du Projet

3.1 Missions Principales :

On cherche à concevoir un robot capable de collecter et déposer un maximum de palets dans un temps imparti, tout en respectant les contraintes du plateau et les règles spécifiques de la compétition. Ainsi, le programme doit permettre aux robots de :

- Localiser les palets à l'aide des capteurs.
- Naviguer sur le terrain de jeu tout en évitant les obstacles (robot adverse, murs)
- Se déplacer vers le camp adverse jusqu'à détection de la ligne blanche
- Déposer le max des palets avant la fin de la manche

3.2 Résultats Attendus

- Vitesse et efficacité : Parmi, les résultats souhaités est que le robot effectue ces actions en un minimum de temps, minimisant ainsi les erreurs telles que les collisions, les rondelles ou les trajectoires sous-optimales. Les stratégies d'optimisation du code et de

navigation contribuera à accélérer ces processus tout en offrant des garanties maximales.

- Performances du Robot : Le robot doit être capable de ramasser un maximum de palets dans les meilleurs délais, et puis les déposer après avoir franchi la ligne blanche.

- La conformité aux règles : Le robot doit respecter toutes les règles de la compétition, notamment franchir la ligne blanche avant de déposer les palets dans la zone de dépôt.

4. Architecture du Système : Perception et Action

Package 1 : Perception

Ce package permet la communication avec tous les capteurs qui permettent au robot de percevoir son environnement.

- ColorSensor: Permet au robot de détecter les couleurs de lignes sur le terrain.
    - Méthodes :
        - `getCurrentColor():String` : Renvoie la couleur actuelle détectée.
        - `changeColor():String` : Met à jour la couleur lorsque le robot détecte un changement.

- UltrasonicSensor: Utilisé pour détecter les objets différents de l'environnement et estimer les distances.
    - Méthodes :
        - `getDistance():float` : Renvoie la distance mesurée par le capteur.
        - `detectPalet():boolean` : Renvoie vrai si un palet est détecté.
        - `detectWall() : boolean` : Renvoie vrai si un mur est détecté.

- TouchSensor: Utilisé pour détecter si le robot est en contact avec un objet.
    - Méthodes :
        - `isPressed():boolean` : Indique si le capteur tactile a été activé.
        - `run():void` : Exécute la logique liée au capteur tactile.

Package 2 : Action

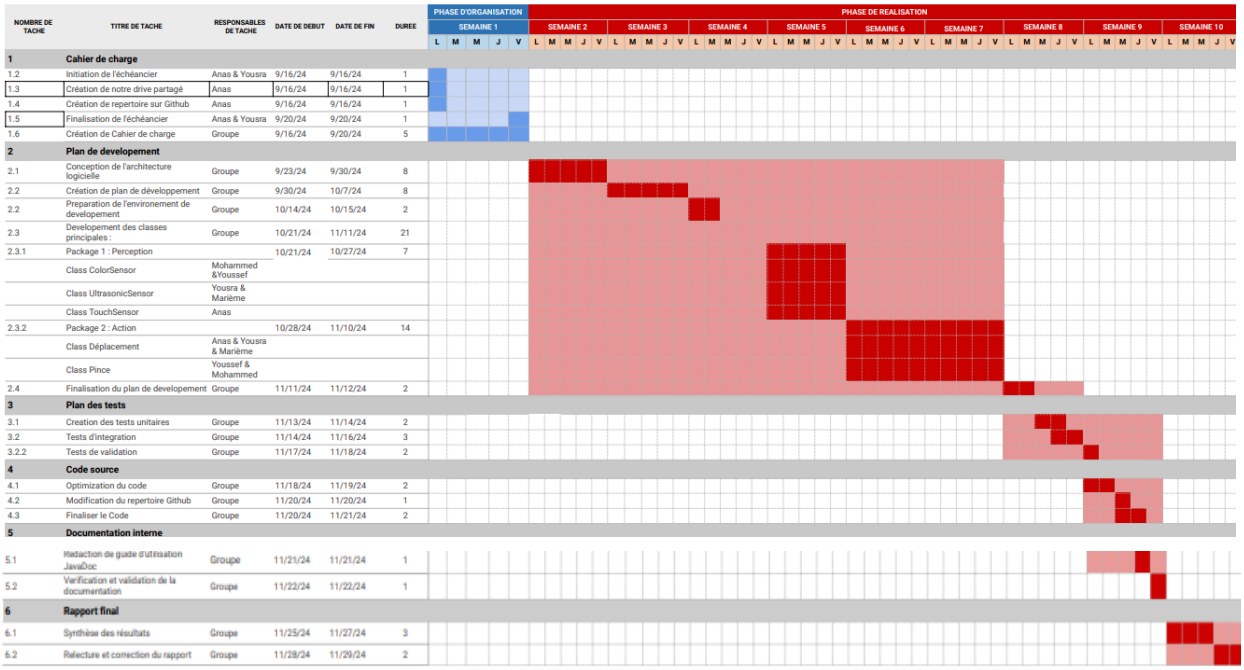
Ce package contrôle les actions physiques du robot.

- Déplacement :
    - Méthodes :
        - `getPilot():MovePilot`: retourne l'objet MovePilot.
        - `avancerAsync(double distance, boolean b):void` : Avancement asynchrone de la distance en paramètres
        - `avancer(double distance):void` : Avancement de la distance en paramètres
        - `tourner(double angle):void` : Rotation de l'angle en paramètre.
        - `tournerAsync(double angle):void` : Rotation asynchrone de l'angle en paramètre.
        - `modifVitLin(double s):void` :modifie la vitesse de déplacement linéaire en degrés par seconde.
        - `modifVitRot(double s):void` :modifie la vitesse de rotation en degrés par seconde.
        - `stop():void` : Arrête le robot.
- Pince :
    - Méthodes :
        - `saisirPalet():void` : Active les pinces pour saisir un palet.
        - `lacherPalet():void` : Relâche le palet dans la zone de dépôt.
        - `setEtat(boolean etat):void` : Modifie l'état des pinces.

5. Planification du Projet

ECHEANCIER GROUPE PROJET IA

TITRE DU PROJET	Projet de robotique
MEMBRES DE GROUPE	AIT EL HADJ Anas, AMACHAT Yousra, HALILY Youssef, NGUIRANE Marième , TAMSOURI Mohammed



NOMBRE DE TACHE	TITRE DE TACHE	RESPONSABLES DE TACHE	DATE DE DEBUT	DATE DE FIN	DUREE
1	**Cahier de charge**				
1.2	Initiation de l'échéancier	Anas & Yousra	9/16/24	9/16/24	1
1.3	Création de notre drive partagé	Anas	9/16/24	9/16/24	1
1.4	Création de repertoire sur Github	Anas	9/16/24	9/16/24	1
1.5	Finalisation de l'échéancier	Anas & Yousra	9/20/24	9/20/24	1
1.6	Création de Cahier de charge	Groupe	9/16/24	9/20/24	5
2	**Plan de developement**				
2.1	Conception de l'architecture logicielle	Groupe	9/23/24	9/30/24	8
2.2	Création de plan de développement	Groupe	9/30/24	10/7/24	8
2.2	Preparation de l'environement de developement	Groupe	10/14/24	10/15/24	2
2.3	Developement des classes principales :	Groupe	10/21/24	11/11/24	21
2.3.1	Package 1 : Perception		10/21/24	10/27/24	7
	Class ColorSensor	Mohammed &Youssef			
	Class UltrasonicSensor	Yousra & Marième			
	Class TouchSensor	Anas			
2.3.2	Package 2 : Action		10/28/24	11/10/24	14
	Class Déplacement	Anas & Yousra & Marième			
	Class Pince	Youssef & Mohammed			
2.4	Finalisation du plan de developement	Groupe	11/11/24	11/12/24	2
3	**Plan des tests**				
3.1	Creation des tests unitaires	Groupe	11/13/24	11/14/24	2
3.2	Tests d'integration	Groupe	11/14/24	11/16/24	3
3.2.2	Tests de validation	Groupe	11/17/24	11/18/24	2
4	**Code source**				
4.1	Optimization du code	Groupe	11/18/24	11/19/24	2
4.2	Modification du repertoire Github	Groupe	11/20/24	11/20/24	1
4.3	Finaliser le Code	Groupe	11/20/24	11/21/24	2
5	**Documentation interne**				
5.1	Redaction de guide d'utilisation JavaDoc	Groupe	11/21/24	11/21/24	1
5.2	Verification et validation de la documentation	Groupe	11/22/24	11/22/24	1
6	**Rapport final**				
6.1	Synthèse des résultats	Groupe	11/25/24	11/27/24	3
6.2	Relecture et correction du rapport	Groupe	11/28/24	11/29/24	2

6. Stratégie d'Optimisation

6.1 Stratégie de Déplacement

Le robot commence sur la ligne la plus éloignée de l'adversaire, avec un palet positionné devant lui, confirmé par ses capteurs de couleur. Il avance vers ce premier palet, utilisant son capteur à ultrasons pour mesurer la distance et ajuster sa vitesse. Une fois proche, le capteur tactile confirme le contact avec le palet, et le moteur actionne la pince pour le saisir. Après avoir ramassé le palet, le robot décide de tourner à droite ou à gauche en fonction de sa position sur le terrain, afin d'éviter les obstacles et les autres palets. Apres faire une rotation vers le bon sens le robot avance ensuite vers la ligne blanche adverse,au long de son chemin le capteur de couleur reste activé et stock la couleur de la derniere ligne avant la ligne blanche, en detectant la ligne blanche il lâche après avoir dépassé cette ligne de 5 cm. Le robot se repositionne ensuite pour chercher le deuxième palet.

Pour ce deuxième palet, il calcule la distance entre la ligne blanche et la ligne bleue et se déplace en conséquence. En atteignant la ligne bleue, il suit cette ligne en scannant chaque intersection pour détecter d'autres palets et les déposer dans le camp adverse .

Le robot se déplace encore une fois vers la ligne blanche. Si aucun palet n'est trouvé immédiatement, il continue vers les intersections suivantes, sachant qu'il reste d'autres palet à récupérer, mais leur emplacement exact reste inconnu. Cette stratégie permet de couvrir efficacement le terrain tout en maximisant la collecte de palets et en évitant les collisions.

6.2 Gestion des Collisions

La gestion des collisions est un aspect crucial pour assurer le bon fonctionnement du robot. Grâce à des capteurs, tels que les ultrasons ou les capteurs tactiles, le robot est capable de détecter des obstacles sur son chemin, comme des palets, des murs ou même d'autres robots. Lorsque ces obstacles sont identifiés, le robot doit réagir rapidement en s'arrêtant ou en modifiant sa trajectoire pour éviter la collision. Cette capacité à anticiper et à éviter les obstacles garantit une navigation fluide et efficace, tout en réduisant le risque de perturbations ou d'endommagements pendant l'exécution de ses tâches.

7. Tests et Validation

7.1 Tests Matériels

Les tests portes sur les différents capteurs du robot :

- Capteur ultrasons : Nous testons la détection des objets à différentes distances et angles.
- Capteur de couleur : La capacité du robot à suivre une ligne colorée.
- Capteur tactile : Le robot est capable de détecter un contact physique ou une pression lorsqu'il touche un objet.
- Les pinces : Tester la prise et la libération des palets de manière efficace.

7.2 Tests Logiciels

Des vérifications régulières du code pour valider son bon fonctionnement.

- Le test des méthodes :
    - Consiste à s'assurer qu'elles réagissent correctement aux capteurs et renvoient les résultats attendus, tout en garantissant leur bon fonctionnement global dans le système.

- Simulation :
    - Ce test a pour objectif de valider l'intégration cohérente et efficace de tous les composants du programme.

8. Conclusion et Perspectives

Le projet d'intelligence artificielle visant à équiper un robot Mindstorms EV3 pour détecter, manipuler et transporter des palets de manière autonome représente un défi technique et organisationnel majeur. Utilisant Java via l'IDE Eclipse et la librairie LeJOS, nous avons structuré notre approche en plusieurs étapes clés. Les ressources matérielles, telles que le robot EV3 LEGO Mindstorms et les divers capteurs, sont essentielles, mais les limitations des capteurs nécessitent une attention particulière. Les objectifs techniques sont ambitieux : concevoir un robot capable de collecter et déposer un maximum de palets dans un temps imparti, tout en respectant les règles de la compétition. Ce plan de développement offre une vision claire des étapes nécessaires pour atteindre nos objectifs, facilitant la coordination et le suivi du projet.