Cahier des Charges - Projet de Programmation de Robot avec LeJos

1. Contexte et Objectif du Projet

Le projet consiste à programmer un robot en utilisant Java, avec le plugin **LeJos** (un environnement de programmation Java pour les robots LEGO Mindstorms). L'objectif final est de préparer le robot pour participer à une compétition sous forme de tournoi, où il affrontera d'autres robots dans des duels un contre un. Chaque duel aura pour but de ramasser un maximum de palets disposés sur une table. Le groupe dont le robot ramasse le plus de palets à la fin du match est déclaré vainqueur.

2. Objectifs Techniques

- **Développement d'un programme Java** pour contrôler les mouvements et les actions du robot LEGO Mindstorms à l'aide du plugin LeJos.
- **Gestion de la détection des palets** et des obstacles sur la table grâce aux capteurs du robot.
- **Optimisation de la stratégie** pour ramasser un maximum de palets avant l'adversaire dans un environnement dynamique.
- **Tests et validation** du programme via des simulations et des tests réels.

3. Description de la Compétition

- **Type de match** : Un contre un.
- **Durée** : Chaque match dure un temps prédéterminé (5min en qualification).
- **Environnement de jeu** : Une table avec **9 palets** répartis de manière équidistante.
- **Objectif** : Le robot doit ramasser le plus de palets possibles en utilisant ses capteurs et ses actions programmées.
- **Critère de victoire** : Le vainqueur est le robot qui ramasse le plus de palets à la fin du match.

4. Contraintes Techniques

- **Plateforme de développement** : Le plugin **LeJos** doit être utilisé pour programmer en Java le comportement du robot LEGO Mindstorms.
- **Capteurs disponibles** : Le robot sera équipé de divers capteurs comme :
    - **Capteur de couleur** pour détecter les lignes.
    - **Capteur à ultrasons** pour détecter les obstacles ou l'adversaire.

    - **Capteur tactile** pour interagir physiquement avec les objets (palets).
- **Actionneurs** : Le robot devra utiliser ses moteurs pour se déplacer et manipuler les palets.
- **Détection et stratégie** :
    - Le robot devra être capable de reconnaître les palets, de les ramasser et de les déposer efficacement.
    - Le robot devra également éviter les collisions avec l'adversaire ou les obstacles.
    - Le robot peut recevoir de l'information d'une caméra infrarouge pour savoir où sont placés les palets et l'autre robot.

5. Fonctionnalités attendues

- **Navigation autonome** : Le robot doit être capable de se déplacer de manière autonome sur la table pour atteindre les palets.
- **Collecte de palets** : Le robot doit ramasser un palet, le transporter et le stocker vers une zone désignée.
- **Gestion des obstacles** : Le robot doit être capable de détecter et d'éviter les autres robots et les bords de la table.
- **Stratégie de priorisation** : Une stratégie d'optimisation doit être développée pour ramasser les palets les plus proches en premier ou maximiser le nombre de palets collectés avant la fin du temps imparti.

6. Scénarios de compétition

- **Situation initiale** : Les 9 palets sont placés sur la table au début du match. Les deux robots commencent à des emplacements opposés.
- **Interaction entre robots** : Les robots peuvent entrer en collision. Le programme doit prévoir des actions correctives (évitement, contournement).
- **Fin du match** : Le match s'arrête lorsque le temps est écoulé ou lorsque tous les palets ont été ramassés.

7. Livrables

- **Code source** : Un programme Java fonctionnel, bien structuré et commenté, qui gère toutes les fonctionnalités requises pour le robot.
- **Documentation technique** : Un manuel de l'utilisateur décrivant le fonctionnement du programme, les stratégies utilisées et la manière de le déployer sur le robot avec LeJos.
- **Plan de développement :** Répartition du travail au sein du groupe. Anticipation des retards et des modifications dans le temps impartit.

- **Plan de test** : Un rapport détaillant les tests effectués (simulation et tests réels), les résultats obtenus et les ajustements nécessaires.

8. Critères de réussite

- Le robot est capable de se déplacer sur la table, ramasser les palets, et interagir correctement avec les obstacles et l'adversaire.
- Le programme est capable d'optimiser le nombre de palets ramassés par rapport à l'adversaire.
- Le robot respecte les contraintes techniques (le robot ne peut pas être modifié) et est capable de participer aux matchs de compétition sans intervention humaine.

9. Planning

- **Phase 1 : Conception**
  - Planification des capteurs à utiliser et de la stratégie globale.
  - Définition de l'architecture du programme.
- **Phase 2 : Développement**
  - Programmation du déplacement, de la collecte de palets, et de la gestion des obstacles.
- **Phase 3 : Tests et optimisation**
  - Tests en conditions réelles et ajustements des stratégies pour optimiser les performances.
- **Phase 4 : Préparation pour la compétition**
  - Ajustements finaux, préparation du robot et du code pour la compétition.

10. Risques et Précautions

- **Problèmes techniques** : Problèmes de communication entre les capteurs et les moteurs.
- **Délais de développement** : Le projet pourrait prendre plus de temps si des tests approfondis sont nécessaires.
- **Limitations matérielles** : Les capteurs peuvent avoir des limites de précision ou de portée, ce qui risque d'affecter la performance du robot.

Conclusion

Ce cahier des charges définit les objectifs, les fonctionnalités attendues et les contraintes techniques pour le projet de programmation du robot en Java avec LeJos. La compétition en un contre un mettra à l'épreuve les compétences de programmation et de stratégie de chaque groupe. Le robot doit être conçu pour

maximiser le nombre de palets collectés dans un environnement dynamique, avec une gestion intelligente des obstacles et de l’adversaire.