2024

Plan de développement

Projet de robotique

Alexander OSTLE, Thomas BEGOTTI
Victor CHARREYRON
Université Grenoble-Alpes

Plan de développement

Projet de Robotique

Les informations d'identification du document

Référence du document :	Plan de développement
Version du document :	1.0
Date du document :	16/12/24
Auteur(s) :	Alexander OSTLE Thomas BEGOTTI Victor CHARREYRON

Les éléments de vérification du document

Validé par :	Alexander OSTLE Thomas BEGOTTI Victor CHARREYRON
Validé le :	16/12/24
Soumis le :	16/12/24
Type de diffusion :	Document électronique (.pdf)
Confidentialité :	Standard / Étudiants de l'Université Grenoble-Alpes

Sommaire

Sommaire 2
Introduction 3
Guide de lecture 3
  Maîtrise d’œuvre 3
  Maîtrise d’ouvrages 3
Conception générale du système 3
Répartition des tâches 4
  Flora 4
  Victor 4
  Thomas 4
  Alexander 5
Phase d'intégration 6
Phase de test et validation 6
Optimisation et amélioration 6
Glossaire 7
Référence 7
Index 7

Introduction

Ce document est une version révisée, a posteriori, du plan de développement des solutions nécessaires pour atteindre l'objectif de ce projet. Dans ce document se trouve la répartition des tâches ainsi que les différentes phases de conception du programme.

Guide de lecture

Maîtrise d'œuvre

Les développeurs et les testeurs pourront suivre ce document pour s'assurer de la conformité des implémentations avec les spécifications et comprendre le déroulement du développement des différentes fonctionnalités

Maîtrise d'ouvrages

L'utilisateur final pourra consulter ce plan pour comprendre le déroulement du développement des différentes fonctionnalités du robot et vérifier leur conformité avec les exigences du cahier des charges.

Conception générale du système

- **Objectif** : Concevoir le fonctionnement global de l'automate en définissant les différents états et les conditions de transition entre eux.
- **États** :
    - **État -1** : Attend que l'utilisateur appuis sur le bouton gauche ou droit du robot en fonction de la stratégie qu'il veut employer, puis passe à l'état 0.
    - **État 0** : Le robot va à une position prédéfinie, ramasse le palet et le dépose dans la zone d'en-but adverse, puis se repositionne au milieu de la table du côté de la zone d'en-but adverse. Il passe ensuite à l'état 1.
    - **État 1** : Le robot lance une recherche pour trouver un palet sur la table s'il n'en trouve pas il recule relance une recherche. Il fait cela jusqu'à ce qu'il trouve un palet, puis il passe à l'état 2.
    - **État 2** : Le palet étant trouver le robot se déplace vers lui, le prend et l'amène dans la zone d'en-but adverse puis se repositionne au milieu de la table du côté de la zone d'en-but adverse, retrouve l'angle 0° et passe à l'état 1.

- **Définitions des classes gérant l'initialisation et l'accès aux actionneurs et aux capteurs.**

Classe Actionneurs. Initialise les moteurs du robot, redéfinit les méthodes de déplacement, de rotation, de la synchronisation des moteurs des roues comme un ‘châssis’, et de l’ouverture et fermeture des pinces. Gère l’accès des méthodes qui seront utilisées dans la classe Robot.

Classe Sensors. Initialise les capteurs du robot, gère l’accès aux méthodes renvoyant les données prises par les capteurs qui seront utilisées dans la classe Robot.

Répartition des tâches

Flora : Conception des premiers livrables et conception de la stratégie de réalisation du projet

- Organisation du projet et répartition des tâches au sein du groupe.

Victor : Développement de l’Etat 2 pour manipuler le palet.

- **Tâches principales** :
  - Développer une solution permettant au robot de prendre et déposer le palet dans la zone d’en but adverse.
  - Livrables : Méthodes pour manipuler le palet à savoir :
    - tournerVersPalet() ;
    - avancerVersPalet() ;
    - prendrePalet() ;
    - deplacerVersMur() ;
    - allerALigneBlanche() ;
    - déposer() ;
- **Étapes** :
  - Une fois la recherche et la détection du palet effectuée, tourner, avancer jusqu’au palet et l'attraper.
  - Se dirigé vers le mur puis vers la ligne blanche.
  - Utiliser le capteur de couleur, pour détecter la zone d’en but adverse.

Thomas : Développement de l’Etat 0 pour le premier palet et de la Classe Actionneurs.

- **Tâches principales** :
  - Développer la méthode associée à l’état Premier Palet, où le robot se déplace et attrape le premier palet dans son axe, se décale de cet axe afin de ne pas percuter les deux palets suivants et dépose le palet dans l'en-but adverse.
  - Initialisation, gestion de l’accès et redéfinition des méthodes appelés sur les moteurs.

    - Architecture de l’automate à états finis avec un structure ‘while’ ‘switch case’.
    - **Livrable** :
        - Méthodes pour le premier palet à savoir :
            - premierPalet() ;
            - prendrePremierPalet() ;
        - Classe Actionneurs
        - Méthode main.
- **Étapes** :
    - Redéfinition des méthodes actionnant les moteurs du robot.
    - Ramasser le palet en fermant les pinces si le capteur tactile renvoie true.
    - Déplacer le robot vers la zone d’en but adverse et ouvrir les pinces pour déposer le palet.

Alexander : Développement de l’Etat -1,1 et de la Classe Sensors.

- **Tâches principales** :
    - Initialisation et gestion de l’accès aux méthodes retournant les données prises par les différents capteurs.
    - Développer les méthodes associées à la recherche et détection de palets et d’obstacles.
    - ‘Boussole’ interne du robot.
    - **Livrables** :
        - Etat 1 pour trouver le palet et les méthodes associés à savoir :
            - Rechercher() ;
            - indicesFiltre() ;
            - paletsFiltres() ;
            - paletLePlusProche() ;
        - Classe Sensors.
        - Méthodes associées à la boussole interne à savoir :
            - Recentrer() ;
            - distanceDevant() ;
            - resetOrientation() ;
            - distanceMin() ;
- **Étapes** :
    - Redéfinition des méthodes permettant de collectionner des données à partir des différents capteurs.
    - Lancer une recherche, le robot fait un tour sur lui-même puis ajoute les distances mesurer par le capteur ultrasonique dans une liste.
    - Filtrer les indices consiste à enlever tous les indices des éléments dans la liste des distances qui serait équivalent aux distances capter entre 0° et 90° et entre 270° et 360°. Sont enlever aussi toutes les distances supérieures à 2 m et inférieur à 30 cm.
    - Pour filtrer les palets le programme parcours la liste des indices cherches le début d’un palet avec une grande différence des distances, les indices de ce palet auront une différence de distance très petite et la fin du palet marqué par un saut de distance. Tout en vérifiant si les indices sont successifs et qu’il ne détecte pas deux fins de palet ou deux débuts de palet et que la taille de

cette liste soit conforme à la moyenne calculer pour un palet. Il ajoute donc ce palet a une liste de liste.
    - Le palet le proche permet de comparer les palets trouver juste avant pour avoir l'orientation du milieu du palet et sa distance.
    - Déplacer le robot vers le centre de la table.
    - Retrouver l'orientation de 0° en effectuant une recherche et en trouvant la distance la plus petite puisqu'on se trouve au centre de la table au niveau de la ligne blanche cet distance minimale sera le mur et on pourra définir cet angle par 0°.

Phase d'intégration

- **Objectif** : Intégrer les différentes méthodes développées dans l'automate à états finis.
- **Tâches** :
    - S'assurer que les transitions entre les états sont bien définies et que le robot passe d'un état à un autre en fonction des événements.
    - Vérifier la bonne définition des classes actionneurs et sensors.
    - Tester les méthodes appelées dans les états individuellement.
    - Tester les méthodes d'état individuellement, puis tester l'ensemble des états en chaîne pour assurer la fluidité du comportement du robot.
    - **Livrable** :
        - Une version fonctionnelle du programme où le robot passe d'un état à un autre de manière fluide et sans erreurs.

Phase de test et validation

- **Objectif** : Tester le robot dans un environnement réel et vérifier son comportement dans chaque état.
- **Tâches** :
    - Tester chaque état séparément pour s'assurer que chaque action est exécutée correctement (déplacement, recherche, ramassage, marquage).
    - Tester les transitions entre les états, et vérifier que les capteurs fonctionnent correctement pour déclencher les transitions.
    - **Livrable** :
        - Plan de test

Optimisation et amélioration

- **Objectif** : Optimiser le comportement du robot pour qu'il soit plus rapide et plus précis. Apporter une solution pour rendre plus précise la recherche de palet. Développer une solution viable pour détecter et esquiver le robot adverse.

Glossaire

Condition de transition : Une condition de transition en informatique ou en robotique désigne une règle ou un critère qui doit être satisfait pour permettre le passage d'un état à un autre dans un système.

État : En robotique, un état représente une situation ou une configuration spécifique dans laquelle le robot se trouve à un moment donné. C'est une condition clairement définie qui décrit les paramètres actuels du système et qui peut évoluer en fonction d'entrées, d'actions, ou de temps.

Main (méthode) : La méthode main est une méthode spéciale en Java. Elle sert de point d'entrée principal dans un programme Java. C'est là où l'exécution du programme commence. Toute application Java autonome doit inclure une méthode main.

Méthode : En Java, une méthode est un bloc de code qui réalise une tâche précise ou effectue une action, et qui peut être appelé à plusieurs endroits dans un programme. Les méthodes permettent de structurer et de réutiliser le code, facilitant ainsi la gestion et la maintenance des application.

True/False : Dans de nombreux langages de programmation, y compris Java, les valeurs true et false appartiennent au type boolean. Ce type sert à représenter deux états opposés (vrai/faux, actif/inactif, 1/0, etc.). Ce type est aussi utilisé en logique.

While Siwtch case : Est une structure algorithmique qui est utilisé afin de représenter les états pris par le robot au cours de la partie et les conditions de transitions.

Référence

- Cahier des charges du projet
- La documentation interne du code
- Documentation LeJOS
- Plan de tests

Index

Condition de transition : page 3.
État : pages 3,4,5,6.
Main (méthode) : page 5.
Méthode : pages 4,5,6.
While Siwtch case : page 5**.**