Projet de robotique

Plan de développement

Version 4.
DESCOTILS Juliette 11807195
ESPINAL Miguelangel 11819395
GATTACIECCA Bastien 11808782
LATIFI Arita 11800876

Année universitaire 2020-2021

Sommaire :

1. Introduction

1.1 Objectif du document

Ce document fait l'objet d'un échéancier ; rassemblant l'ensemble des tâches à finir, et pour chacune d'elles, les dates limites ainsi que les membres du groupe en charge. L'ensemble des fonctionnalités devant être développées dans le projet sera défini précisément dans ce document, ainsi qu'une vue d'ensemble du projet et de la manière dont il s'organisera.
Il permettra ainsi de contrôler l'avancement du projet et de diriger le travail de développement.

1.2 Définitions, acronymes et abréviations

MIASHS : Mathématiques et Informatique appliquées aux sciences humaines et Sociales
UE : Unité d'Enseignement
leJOS : Le nom de la librairie
EV3 : Nom de la 3eme génération de LEGO Mindstorm
Méthode bloquante : Une méthode bloquante tourne dans le Thread principal et bloque l'accès des autres méthodes de ce même thread tant que l'action en cours n'est pas terminée. A l'opposé, une méthode non bloquante décale l'opération dans un Thread secondaire pour laisser le Thread principal libre.

2. Vue d'ensemble du projet

2.1 But du projet, portée et objectifs

Le but du projet est de programmer en langage Java un logiciel embarqué dans un robot LEGO. L'objectif est d'assimiler des techniques d'intelligence artificielle pour contrôler le robot. Il possède trois capteurs et trois moteurs dont le fonctionnement de très bas niveau de chacun est déjà implémenté dans la librairie fournie leJOS.

2.2 Hypothèses et contraintes

2.2.1 Hypothèses

La programmation du logiciel embarqué nécessite de commencer avec certaines suppositions. La première est que, comme tous les groupes possèdent le même robot, alors les capteurs possèdent une variabilité semblable La deuxième est que les moteurs ont, pour un voltage identique, une force identique. Par conséquent, le robot avance de manière strictement linéaire si les moteurs gauche et droit ont été paramétrés identiquement.

2.2.2 Les contraintes

Liées au robot :

- Composants du robot imposés - pas de modifications du robot possible.
- 3 capteurs et 2 moteurs en l'état (ultrasons, couleur, touché).
- 1 batterie de capacité limitée en l'état.
- 2 roues de taille imposée.
- 1 brique EV3.
- Structure en LEGO.

Liées au plateau :

- Caractéristiques du plateau prédéfinies.
- Plateau de 3x2 mètres.
- Plateau en bois avec lignes de couleur qui séparent 16 rectangles de 50x60cm.
- Vitres en plexiglas aux bords du plateau d'une hauteur de 15cm.

Liée à la librairie lejos :

- Ensemble de méthodes et classes prédéfinies et hiérarchie de classe et packages imposés.

Liées à la crise sanitaire :

- Accès au laboratoire limité à cause des contraintes sanitaires liées au covid-19. Possibilité de réserver la salle en avance en envoyant un mail jusqu'au 29 Octobre 2020. Du 29 Octobre au 14 Novembre (exclus) aucun accès à la salle possible. A partir du 14 Novembre, nous étions autorisés à avoir accès au plateau 1h tous les lundis au créneau habituel (au lieu des 4h).
- Impossibilité pour certains membres du groupe d'assister en présentiel aux séances de 1h à partir du 14 Octobre dû au lieu de confinement.

- Contraintes logiciel : utilisation du logiciel GitHub et du langage Java sous Eclipse.

- Contraintes humaines : 4 développeurs.

3. Organisation du projet

Ce projet se fait dans le cadre de l'UE d'Intelligence Artificiel intégré à la troisième année de licence MIASHS. Le groupe se compose de : Descotils Juliette, Espinal Miguel, Gattaciecca Bastien, Latifi Arita. Ce projet est supervisé par D. Pellier, l'enseignant de ce cours.

4. Processus de gestion

4.1 Plan de projet

Semaine	Tâche	Documents à rendre
n°1	Définition des objectifs	
n°2	Analyse des besoins	
n°3	Spécification	Cahier des charges
n°4	Conception	
n°5	Développement	Plan de développement
n°6	Développement	
n°7	Développement	
n°8	Développement	
n°9	Développement	
n°10	Intégration	Plan de tests
n°11	Recette	Code source et documentation interne
n°12	Évaluation	Rapport final

Pellier, D. (s. d.). teaching : ia : project_lego [Damien Pellier Associate Professor Univ. Grenoble Alpes]. LIG. https://lig-membres.imag.fr/PPerso/membres/pellier/doku.php?id=teaching:ia:project_lego

4.1.1 Planification des phases

Le tableau ci dessous récapitule l'échéancier de programmation qui était fixé :

Nom de la méthode	Description	Deadline	Personne en charge
Classe Perception			
update()	Méthode appelée toutes les 20ms depuis la classe Agent. On met ainsi à jour les données de nos capteurs régulièrement via un deuxième Thread.	12.10.2020	Arita, Miguelangel
setCalibratedSamples()	Récupère les échantillons des couleurs principales que le robot est censé discriminer depuis le fichier qui a été créé lors du calibrage, et les ajoute dans la LinkedList. On construit donc ici 6 échantillons des 6 couleurs principales que l'on récupère depuis un fichier puis qu'on ajoute dans la liste.		
getCouleur()	Renvoie un String correspondant au nom de la couleur. Retourne la couleur parmi la liste des couleurs définies comme étant discernables par le robot.		
getTouch()	Renvoie un booléen (true si le capteur de toucher est enclenché). Permet de savoir si un objet se situe entre les pinces.		
getDistance()	Renvoie un float correspondant à la distance perçue par le capteur à ultrason en centimètres.		
Classe Actionneur			
getMouvement ()	Renvoie un objet de type Move. Permet de récupérer le mouvement en cours.	12.10.2020	Bastien, Juliette
avancer (double distance, double speed, boolean nonBloquante)	Permet de faire avancer/reculer le robot.		
rotation (double angle, double speed, boolean nonBloquante)	Permet au robot de faire des rotations sur lui-même		
travelArc (double radius, double distance)	Déplace le robot sur le long du cercle spécifié par son rayon. Si le rayon est positif, le centre de ce cercle est à gauche du robot ; sinon à droite. Si la distance est positive, le robot se déplace sur le cercle en avançant, sinon en reculant. Si le rayon est nul, le robot tourne sur lui-même. Si la distance est nulle, la méthode retourne immédiatement.		
stop()	Permet au robot de s'arrêter.		
ouvrirPinces()	Ouvre les pinces du robot, et met à jour l'attribut pincesOuvertes. Méthode bloquante.		
fermerPinces()	Ferme les pinces du robot, et met à jour l'attribut pincesOuvertes. Méthode bloquante.		

updateDirection (double angle)	Incrémente la direction actuelle du robot de l'angle passé en paramètre. On souhaite toujours récupérer l'angle actuel du robot selon un angle compris entre 0 inclus et 360 exclu.		
isMoving()	Renvoie un booléen (true si les moteurs sont en marche) Cette méthode indique si le robot est actuellement en train d'effectuer un mouvement (de la classe Move).		
pincesOuvertes()	Renvoie un booléen (true si les pinces sont ouvertes). Accesseur public à l'attribut pincesOuvertes.		
getDirection()	Renvoie un entier correspondant à la direction dans laquelle est orienté le robot en degrés. Accesseur public à l'attribut direction.		
Classe Agent			
getAction()	Renvoie un objet de type Actionneur. Accesseur public à l'attribut action.	10.11.2020	Arita
getPerception()	Renvoie un objet de type Perception. Accesseur public à l'attribut perception.	10.11.2020	Juliette
prendrePaler()	Renvoie un booléen (true si le capteur de toucher a détecté un palet et le robot a fermé les pinces). Dès qu'on pense avoir détecté un palet, on appelle cette méthode pour le récupérer.	16.11.2020	Miguelangel
testDistance()	Cette méthode permet de tester les valeurs du capteur distance	10.11.2020	Bastien
testCouleur()	Cette méthode permet de tester les valeurs du capteur couleur.	10.11.2020	Arita
perpendiculaire()	Renvoie un booléen (true si. Cette méthode permet au robot de tourner vers le mur et de s'arrêter dès que la distance augmente.	16.11.2020	Juliette
resetDirection()	Renvoie un booléen (true dès que le robot fini de tourner). Cette méthode permet au robot de se repositionner à 0°, en fonction de la direction actuelle. Donc dans la même direction que le robot était dans sa position de départ.	10.11.2020	Bastien
directionNearestObject()	Renvoie un float (correspondant à la distance de l'objet le plus proche). "Scanne" les alentours sur 360° puis se pointe vers l'objet qui était le plus proche.	16.11.2020	Miguelangel

4.1.2 Ressources humaines du projet

- 4 étudiants : Descotils Juliette, Espinal Miguel, Gattaciecca Bastien, Latifi Arita

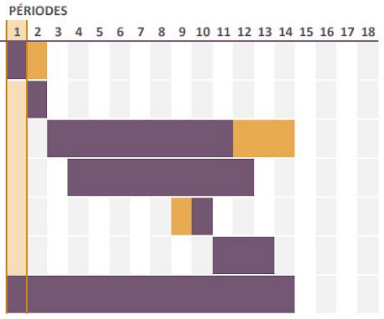
5. Annexes

Annexe 1 : L'échéancier

Planificateur de projet

Période à mettre en évidence : 1 | % accompli | % accompli au dela du plan

ACTIVITÉ	DÉBUT DU PLAN	DURÉE DU PLAN	DÉBUT RÉEL	DURÉE RÉELLE	POURCENTAGE ACCOMPLI
Définition des objectifs	1	1	1	2	100%
Analyse des besoins	2	1	2	1	100%
Rédaction du cahier des charges	3	9	3	12	100%
Développement	4	9	4	9	100%
Rédaction du plan de test	10	1	9	2	100%
Code source et documentation	11	3	11	3	100%
Rapport final	1	14	1	14	100%



Annexe 2 : Le cahier des charges (voir document joint)
Annexe 3 : Le plan de test (voir document joint)
Annexe 4 : Le diagramme UML (voir document joint)