Université Grenoble Alpes

PLAN DE TEST

ROBOT LEJOS EV3

VERSION DU DOCUMENT : 1.0

RÉFÉRENCE : D3

NOM DU PROJET : ROBOT-LEJOS-EV3-FASTLEARNING

DATE : 02/11/2022

AUTEURS : LOUIS BASTIEN FAUCHON

DANIEL BEQAJ

RAPHAËL DELATTRE

TYPE DE DIFFUSION : HTTPS://GITHUB.COM/VETRARR/-ROBOT-LEJOS-EV3-FASTLEARNING

MAÎTRE D'OUVRAGE : DAMIEN PELLIER **MAÎTRE D'OEUVRE : VOIR AUTEURS**

Doit contenir plus de tests que le code source et la documentation interne, couvre l'ensemble des applications du projet, soit le programme du robot, les fonctionnalités du robot

1.Introduction (ou préambule)

Les tests présentés ici visent à assurer le fonctionnement complet des fonctionnalités du robot et les objectifs que le robot doit atteindre. Pour que le robot soit fonctionnel, les tests se portent d'une part sur le fonctionnement réel et observé du robot dans les conditions prévues. D'autre part le programme java est aussi testé pour vérifier que chaque méthode et classe renvoient ce qui est demandé.

1.1.Objectifs et méthodes

Le logiciel est développé à partir de l'api dont la documentation se trouve sur https://lejos.sourceforge.io/ev3/docs/. La programmation se fait sur eclipse ou les packages de l'api sont importés et le code est envoyé au format .jar via wifi sur la carte sd du robot. Le code est en 3 classes, Moteurs, Senseurs et Réflexion. Le développement est réalisé par la définition des objectifs réels du robot qui sont définis sous forme algorithmique avant la programmation java. Les méthodes sont construites à partir des packages préexistants et utilisent d'autres méthodes réparties dans les classes selon les trois conditions senseurs pour ce qui se rapporte aux senseurs, moteurs pour le déplacement et réflexion pour la stratégie global du robot et le fonctionnement coopéré des capteurs et moteurs.

1.2.Documents de référence

Cahier des charges

Plan de développement

ajouter les références

2.Guide de lecture

2.1.Maîtrise d'œuvre

Utiliser le document pour effectuer les tests et mettre à jour si une fonctionnalité n'est pas testé.

2.2.Maîtrise d'ouvrage

Vérifier que l'ensemble des test mentionner sont suffisant aux attentes préétablies.

3.Concepts de base

La notion de classes java, méthodes et attributs java, de l'utilisation des différents capteurs, des possibilités du robot.

4.Tests fonctionnels

4.1.Pour chaque scénario :

4.1.1.Identification

F.S.1 à 9

4.1.2.Description

F.S.1) Décrire le but du test, les caractéristiques de l'environnement de test et le principe de réalisation du test.

F.S.2) Distance : La distance rendu par les capteurs correspond à la distance réel avec une erreur minimale

F.S.3) Distance min max : la distance maximum et minimum du capteurs correspond aux mesures effectuées

F.S.4) Couleur: Chaque couleur de bande est détecté par le capteur de couleur

F.S.5) Arrêt: Le robot s'arrête face à un mur ou face à un autre robot

F.S.6) En-but : Le robot se déplace de sa position à l'en-but quand il a saisi un palet

F.S.7) Attraper palet : Le robot ouvre et ferme complément sa pince pour la saisi d'un palet

F.S.8) Post arrêt : le robot sait reprendre une stratégie après un arrêt avec les données d'angle vers l'en but adaptées.

F.S.9) Trajectoire: le robot effectue la trajectoire et distance demandée

4.1.3.Contraintes

Les contraintes d'environnement sont un terrain de 3m par 2m, des murs de 50cm de haut.

4.1.4.Dépendances

Ces tests nécessite que l'ensemble des test d'intégration et unitaires fonctionne

4.1.5.Procédure de test

La procédure pour chaque test est d'utiliser le programme principal du robot et les méthodes décrites dans le plan de développement pour effectuer les scénarios décrits dans les conditions du terrain de la compétition.

5.Tests d’intégration

5.1.Pour chaque test d’intégration :

5.1.1.Identification

Tests d’intégration classe Moteurs : I.M

Tests d’intégration classe Senseurs : I.S

Tests d’intégration classe Reflexion : I.R

5.1.2.Description

I.M) Packages de la classe Moteurs, le but des tests est de vérifier que les packages peuvent être utilisés pour le contrôle des moteurs des deux roues du robot.

I.M.1) lejos.robotics.chassis.Chassis;

I.M.2) lejos.robotics.chassis.Wheel;

I.M.3) lejos.robotics.chassis.WheeledChassis;

I.M.4) lejos.robotics.navigation.MovePilot;

I.M.5) lejos.utility.Delay;

I.M.6) lejos.robotics.RegulatedMotor;

I.M.7) lejos.hardware.motor.Motor;

I.S) Packages de la classe Senseurs, le but des tests est de vérifier que les packages peuvent être utilisés pour l’utilisation des capteurs.

I.S.1) Utilisation de la classe SampleProvider pour récupérer les données des capteurs dans un tableau:

I.S.2) lejos.robotics.SampleProvider;

I.S.3) lejos.hardware.sensor.SensorModes;

I.S.4) Capteur bouton poussoir pour le toucher :

I.S.5) lejos.hardware.sensor.EV3TouchSensor;

Capteur à ultrason :

I.S.6) lejos.hardware.sensor.EV3UltrasonicSensor;

I.S.7) lejos.hardware.ev3.LocalEV3;

I.S.8) Accès aux capteurs :

I.S.9) lejos.hardware.port.Port;

I.R) Packages de la classe Reflexion, le but des tests est de vérifier que les packages permettent de stocker les données du capteur à ultrason dans un tableau:

I.R.1) java.util.ArrayList;

I.R.2) java.util.List;

I.R.3) lejos.utility.Delay;

5.1.3.Contraintes

I.M, I.S, I.R : Les tests se font dans les conditions où le robot peut rouler et où les capteurs peuvent effectuer une mesure (limite de distance, couleurs, bouton poussoir).

5.1.4.Dépendances

Ces tests réclament que eclipse a été configurée au préalable et que des méthodes soient définies pour tester les différents packages.

5.1.5.Procédure de test

Les données d'entrées sont le retour d'ultrason après émission d'ultrasons, le retour lumineux sur les couleurs et le déplacement de la barre précédant le bouton poussoir qui doit être enfoncé. Ces données sont ensuite affichées sur l'écran du robot.

Les données des roues sont la vitesse de rotation.

6.Tests unitaires

6.1.Pour chaque test unitaire :

6.1.1.Identification

Pour la classe Moteur:

U.M de 1 à 9

Pour la classe Senseurs:

U.S de 1 à 2

Pour la classe Reflexion:

U.R de 1 à 4

6.1.2.Description

Classe Moteurs

U.M.1) public void forward(double distance): Le robot avance de la distance donnée en paramètre.

U.M.2) public void forwardAsync(double distance): Le robot avance de la distance donnée en paramètre.

U.M.3) public void stop(): Le robot s'arrête.

U.M.4) public void rotate(double angle): Le robot tourne de l'angle donnée en paramètre.

U.M.5) public void rotateAsync(double angle): Le robot tourne de l'angle donnée en paramètre.

U.M.6) public boolean isMoving(): La méthode retourne true quand le robot se déplace.

U.M.7) public void ouvrirPinces(): Le robot ouvre la pince à la limite d'ouverture.

U.M.8) public void fermerPinces(): Le robot ferme la pince à la limite de fermeture.

U.M.9) public void deplaceArc(double rayon,double angle): Le robot se déplace le long d'un cercle dont le rayon est donné en paramètre et pour une distance correspondant à l'angle donné en paramètre.

Classe Senseurs:

U.S.1) public float getDistance(): La méthode retourne la valeur en float de la distance du capteur à ultrason.

U.S.2) public boolean isTouched(): La méthode retourne true si le robot entre en contact avec un palet.

Classe Reflexion:

U.R.1) public void attraperPaletDevant(): La methode retourne la distance estimée entre le robot et un palet.

U.R.2) public void attraperPaletDevant(float distance): La méthode provoque la saisie d'un palet à la distance donnée en paramètre, la méthode dépend de isTouched et comprend l'ouverture et la fermeture de pince.

U.R.3) public float chercherPalet(): La méthode provoque une rotation du robot et enregistre les données du capteur à ultrason dans un tableau, la méthode retourne la valeur de l'angle où le palet a été détecté.

U.R.4) public void allerBut(double angle, double distance): La méthode prend en paramètre l'angle pour atteindre l'en-but adverse et la distance à parcourir pour déposer le palet par ouverture de pince dans l'en-but adverse.

6.1.3.Contraintes

Les contraintes du tests sont sur le terrain de la compétition avec les palets positionnés à l'emplacement prévu.

6.1.4.Dépendances

Le test des méthodes dépend de l'installation du plugin eclipse, de l'utilisation des packages lejos.

6.1.5.Procédure de test

La procédure de test se fait par observation du comportement du robot après l'envoi des méthodes du programme dans le robot.

7.Vérification de la documentation

L'installation du plugin Lejos permettant l'importation des librairies, le contrôle du robot et l'exportation du code fonctionne.

La procédure de d'exportation du code via wifi entre éclipse et le robot fonctionne.

8.Annexes

Documentation Robot

1. Un professeur d'intelligence artificielle disponible pendant les séances du lundi après midi et à la demande. Les membres du fablab.
2. Un robot LEGO Mindstorms.
3. Le robot est équipé de :
   3.1. Capteur tactile à bouton poussoir.

3.2. Capteur de son : intensité sonore en décibels.
3.3. Capteur à ultrasons (unité : mètre, valeur max : 2,5m, valeur min : 30cm, angle : 20°).
3.4. Capteur de couleurs : RED.
3.5. Servomoteurs : capteur de rotation diminué par la résistance au mouvement
3.6. Microprocesseur 32 bit ARM7 d'ATMEL.
3.7. Fonction Bluetooth (connexion à d'autres NXT ou à un PC et possibilité de contrôler le NXT avec un téléphone portable ou un autre appareil Bluetooth).
3.8. 1 port USB 2.0 (12 Mbps).
3.9. 4 ports d'entrée pour la connexion des capteurs nommés 1, 2, 3 et 4.
3.10. 3 ports de sortie pour les moteurs nommés A, B et C.
3.11. Écran à cristaux liquides 100 × 64 pixels.
3.12. Haut-parleur intégré (qualité sonore 8 kHz - 8 bit - échantillonnage 2-16 kHz).
3.13. Alimentation : 6 piles AA (1,5 V) ; une batterie 9 V est commercialisée par LEGO.
3.14. Dimensions : 112 × 72 × 40 mm.

4. Carte SD.
5. Clé wifi.
6. Réseau wifi PERSYCUP2.
7. Palets.
8. Terrain 3m x 2m, 9 positions pour le palet et 3 départs d'un côté du terrain, 3 départs de l'autre côté du terrain, deux zones d'en-but.
9. Caméra infrarouge au-dessus du terrain.
10. Répertoire Github
11. IDE eclipse
12. Ora - Task management

Règlement de la compétition

Préambule

Pour cette compétition, les différents robots impliqués dans la compétition devront résoudre le problème du ramasseur de balles.

Article 1. - Limite de ce règlement et modifications

Les enseignants en charge du module se réservent le droit de modifier ce règlement à tout moment de la compétition s'il juge cela nécessaire afin de faire respecter l'esprit de cette rencontre. Au cas où une modification du règlement interviendrait, les modifications apportées seront affichées et le jury organisera une réunion pour informer tous les participants encore en jeu. Le jury est souverain et ses décisions sont sans appel. Il peut notamment décider de pénaliser un robot ou une équipe qui présenterait un comportement contraire à l'esprit de cette compétition, même si la faute reprochée n'est pas explicitement prévue par ce règlement.

Article 2. - Conception des robots

Les étudiants ne conçoivent pas les robots, les plans sont fournis à l'article 8. du règlement. Les plans ne peuvent pas être modifiés par les équipes. Les étudiants programment les robots et les mettent au point pour la compétition. Pour les aider dans cette tâche, les enseignants peuvent mettre à disposition les outils logiciels standards de leurs choix. En aucun cas, les modules matériels du robot ne pourront être des modules spécialisés pour la compétition.

Article 3. - Organisation de la compétition

Le compétition est organisé en cinq phases : homologation, qualification, phase finale et remise des prix (bonus sur la note finale).

3.1. - Phase 1 : l'homologation

Pour participer aux épreuves, un robot doit être homologué. L'homologation se déroule en deux étapes.

3.1.2. - Etape 1 : homologation par le jury, homologation des caractéristiques du robot

Le jury vérifiera que le robot est conforme aux plans fournis pour la construction du robot.

3.1.2. - Etape 2 : test d'homologation, homologation sur l'aire de jeu

Le robot doit être capable (en moins de 3 minutes), au choix de :

- se déplacer de son point de départ à la zone d'en-but adverse
- se saisir d'une balle/ d'un palet placé au centre du terrain et de la déposer dans la zone d'en-but.

Le robot ne sera pas homologué s'il ne réussit pas au moins l'un de ces tests avant la fin de la période d 'homologation. Il ne pourra donc pas participer à la compétition.

3.2. - Phase 2 : la qualification

3.2.1. - Règle d'engagement

Tous les robots homologués participent à la première phase du concours appelée phase de qualification.

3.2.2. - Organisation de la phase de qualification

Au cours de la phase de qualification, chaque équipe rencontre une fois toutes les équipes adverses qualifiées, au sein d'une même poule. Chaque match comprend deux manches d'une durée de 5 minutes pour toutes les ligues. Le temps de pause entre deux manches est de 5 minutes.

3.2.2. - Décompte des points

Au cours de chaque manche, les robots marquent des points suivant les conditions énoncées dans les articles de la section 4 du présent règlement. Le score final est la somme des points obtenus dans chaque manche.

3.2.3. - Classement

Les points des rencontres de chaque robot sont additionnés. Les robots sont classés par ordre décroissant de points obtenus. En fonction de la ligue, les 2 ou 4 premiers de ce classement sont qualifiés pour la phase d'élimination directe. En cas d'équipes ex-aequo, l'équipe sélectionnée sera celle dont le temps cumulé de premières marques (sur tous ses matchs) est le plus court.

3.3. - Phase 3 : phases finales

3.3.1. - Règle d'engagement

Dans la deuxième phase, appelée phase d'élimination directe, seules les équipes qualifiées participent.

3.3.2. - Organisation de la phase d'élimination directe

Cette phase comprend cinq rencontres réparties en deux matchs de demi-finale, une petite finale et une grande finale. Pour passer au niveau suivant, une équipe doit remporter son match. La petite finale permet d'établir le classement des 4 premières équipes.

3.3.3. - Déroulement du match

Chaque match comprend trois manches de 5 minutes avec des pauses intermédiaires de 5 minutes. Au cours d'une pause, une équipe peut changer le programme de son robot.

3.3.4. - Élimination aux nombres de manches gagnées

A la fin de chaque manche, l'équipe qui totalise le plus de points gagne. L'équipe qui gagne 2 manches gagne le match.

3.3.5. - Cas des équipes ex-aequo

En cas de score identique, l'équipe victorieuse sera celle qui aura marqué en premier sur l'ensemble du match.

3.4. - Phase 4 : la remise des prix

Les compétiteurs qui recevront un prix seront les 3 premiers de chaque ligue :

1. Médailles d'or + 3 points de bonus
2. Médailles d'argent + 2 points de bonus
3. Médailles de bronze + 1 points de bonus

Article 4. - Déroulement d’un match

4.1. - L’arbitre

Pour chaque match, un arbitre est désigné par le jury de la compétition pour s’assurer du bon respect du présent règlement. L’arbitre doit:

1. vérifier le bon état de l’aire de jeu.
2. vérifier la conformité des robots avant le coup d’envoi.
3. garantir le respect des règles de la compétition.
4. comptabiliser les points marqués par les robots (voir article 4.4.4 & 4.4.5)
5. Après avoir vérifié que les conditions initiales du match étaient bien réunies, l’arbitre autorise le début du match et donne le coup d’envoi du match. Le camp de l’équipe est décidé par l’arbitre de manière aléatoire. A chaque manche, les équipes changent de camp. En cas d’anomalie au démarrage d’un des robots, l’arbitre peut faire recommencer le match. Cette décision ne peut intervenir qu’une fois par manche, dans les 30 premières secondes. Au cours du match, l’arbitre peut suspendre le match s’il constate un problème sérieux. A la fin de chaque match, il siffle le coup de sifflet final, puis annonce le nom du gagnant ainsi que le score obtenu. Chaque équipe doit prendre soin de respecter le règlement de la compétition et les décisions de l’arbitre. Tout manquement, pourra entraîner une pénalité.

4.2. - L’accès au terrain

Seuls deux des membres d’une équipe sont autorisés à assister au déroulement du match dans la zone de proximité du terrain. De même, ils sont les seuls autorisés à intervenir sur les robots pour des opérations de maintenance ou pour modifier la stratégie de jeu et ce uniquement lors des pauses entre les manches.

4.3. - Les robots

Les robots de chaque équipe doivent avoir une couleur spécifique. Il sera mis à disposition des équipes u n système de pastilles autocollantes pour différencier les robots. L’arbitre vérifiera que les robots en sont pourvus. Les robots devront être autonomes, i.e., que les programmes devront être embarqués. Autrement dit, il n’est pas autorisé de piloter à distance les robots ou d’exécuter son programme de manière déportée. Tout manquement à règle entraînera l’exclusion immédiate de la compétition.

4.4.- Début d’une manche

4.4.1. - Positions initiales des robots

Les robots sont positionnés au choix par les équipes sur les positions notées R sur la figure 1 dans leur camp. Chaque équipe doit annoncer à l’arbitre la position de départ de son robot avant le début de la manche ou de la reprise (après une interruption de jeu).

4.4.2. - Positions initiales des balles

Avant le coup de sifflet, l’arbitre place neuf balles ou neuf palets sur les intersections de lignes (cf. figure 1).

4.4.3. - Début et fin d’une manche

Les robots doivent être démarrés à distance en bluetooth ou en WIFI par l'intermédiaire d’un ordinateur portable. L’arbitre vérifiera que le programme s’exécute sur le robot et n’est pas déporté sur un ordinateur portable. A l’issue du temps réglementaire, l’arbitre siffle la fin de la manche :

- Il arrête de comptabiliser les points,
- Il procède au calcul du score de chaque équipe pour la manche.

4.4.4.- Calcul du score pour les ligues Mindstorms

Le calcul des scores s’effectue comme suit :

- 5 points pour chaque palet placé dans la zone d’en-but adverse,
- 3 points de bonus pour l’équipe ayant marqué en premier,
- 2 points pour le robot ayant un palet en sa possession a la fin de la manche.

Il est interdit aux robots de pousser les palets dans l’en-but. Seules les balles saisies puis déposées dans l’en-but adverse sont comptabilisées.

4.4.5. - Autonomie

Le robot est autonome. Il ne reçoit aucune information lors du déroulement d'un match. Un robot qui exploite des informations qui lui seraient transmises serait immédiatement disqualifié par l'arbitre.

4.4.6. - Apprentissage autonome

Un robot peut exploiter de façon autonome ses expériences des précédents matchs à condition que cet apprentissage soit réalisé de manière logicielle.

4.4.7. - Changement des batteries

Les batteries peuvent être remplacées à la fin de chaque manche ou au cours des temps morts.

Article 5.- Robots perdus

Si un robot ne répond plus, il est considéré comme perdu. Dans les 30 premières secondes de la manche, l'équipe responsable du robot peut alors demander un temps mort afin de remettre en état de fonctionnement l e robot. Le temps mort accordé est fixé à 5 minutes maximum. Il ne pourra pas être demandé plus de deux temps morts par match. Pour les phases finales, aucun temps mort ne sera autorisé.

Article 6.- Collision

Les collisions entre robots sont à éviter afin de limiter les risques de détérioration ou de casse. Toutefois, les robots pourront être utilisés pour bloquer l'accès aux balles ou bloquer l'accès à la zone d'en -but.

Article 7.- Description du terrain

Le terrain mesure 3m x 2m. Il est entouré d'une bordure rigide d'une hauteur de 15cm (cf. Figure 1 et Figure 2). Les R représentent les positions initiales des robots.

9.Glossaire

Définit l'ensemble des termes spécialisés du document

Intelligence Artificielle = un agent intelligent qui perçoit son environnement et agit de façon pour maximiser les chances d'accomplir ses objectifs.Ici: le robot pourra reconnaître son environnement grâce à différents capteurs et faire des actions dit "intelligentes" comme par exemple éviter la collision avec les autres robots.

Robot autonome = un robot autosuffisant qui n'exige pas d'assistance humaine.Ici:un robot qui exécute son programme et agit pour accomplir ses objectifs sans besoin d'être contrôlé à distance.

Robot perdu = un robot qui ne répond plus

10.Références

Indique les références bibliographiques vers d'autres documents apportant des informations complémentaires

-Le règlement de la compétition
https://lig-membres.imag.fr/PPerso/membres/pellier/doku.php?id=teaching:ia:project_lego

-Lejos
https://sourceforge.net/p/lejos/wiki/Home/

Cahier des charges:
https://github.com/Vetrarr/-Robot-Lejos-EV3-FastLearning/blob/main/Cahier%20Des%20Charges_v1.pdf

Plan de dévellopement:
https://github.com/Vetrarr/-Robot-Lejos-EV3-FastLearning/blob/main/Plan%20de%20d%C3%A9veloppement_v1.pdf

11.Index