BRAZEAU Robin
TS2

Mon CV

Robot mobile

LEGO MINDSTROMS NXT 2.0

Projet réalisé avec Valentin DISSLER

SOMMAIRE

Présentation

❖ Pourquoi ce projet ?

Nous avons choisi ce projet car la robotique fait de plus en plus partie de notre quotidien. De plus en plus de robots sont fabriqués pour effectuer des tâches extrêmement différentes. De plus, le côté ludique du robot associé à une utilisation originale du lego nous semblait intéressant.

❖ But du projet

Le but de notre projet est de créer une interface clavier (d'ordinateur) permettant de faire exécuter au robot les trajectoires de différentes lettres : lorsqu'on appuie sur la touche "L", le robot effectue le tracé du "L".

- En 1928, le premier robot « écrivain » est créé au Japon. Depuis, l'avènement de l'informatique a permis de faire exécuter aux robots des tâches complexes dans des temps limités, et l'un de ses challenges a été de développer l'habileté des robots à écrire. A tel point qu'aujourd'hui les chercheurs du CNRS imaginent envisageable l'apprentissage de l'écriture assistée par robot.

❖ Environnement de travail utilisé

Pour mener à bien notre projet, nous avons utilisé :

- une boîte de lego Mindstorms NXT (brique NXT, servomoteurs, lego de construction)
- Processing (avec la librairie NXTComm permettant d'utiliser les fonctions appropriées)
- Le langage Java Processing

Sur le web, nous n'avons trouvé aucun projet se rapprochant du nôtre, utilisant Processing pour piloter à distance une brique NXT mais celle-ci peut être contrôlée à partir d'un joystick ou même d'une « Wii Board ».

❖ Liste des tâches et répartition

<table>
<tr><th>Valentin</th><th>Robin</th></tr>
<tr><td colspan="2">Définition du projet
Découverte du robot</td></tr>
<tr><td>Mouvements du robot
Mise en forme des lettres (graphique)
Perspectives</td><td>Contrôle du robot en direct
Mise en forme des lettres (robot)
Pistes d'améliorations</td></tr>
</table>

I. Lego NXT en général

Ce projet a été conçu par le groupe Lego dans la gamme « robotique programmable ». Son utilisation est simple et essentiellement pédagogique.
Au collège, la programmation se fait principalement à l'aide de pictogrammes.
Au lycée ou dans les grandes écoles, cette brique est plus souvent programmée en Java, C ou python.

❖ Présentation des différents éléments du robot

L'élément principal est la brique NXT (au centre) qui est reliée à l'ordinateur et qui dirige :

- les servomoteurs (3 au maximum)
- les capteurs (4 au maximum) : infrarouge, luminosité, couleurs, contact,…

II. Notre projet

A. Librairies

Les fonctions de commandes des servomoteurs et des capteurs proviennent de la libraire de Jorge Cardoso, mises dans le programme grâce à l'instruction ***import***.

B. Utilisation de Processing (Bluetooth)

Pour pouvoir programmer la brique NXT, il nous a fallu créer une application grâce au logiciel de développement Processing. Nous avons pu commander le robot à partir de Processing grâce à la technologie Bluetooth, intégrée directement dans la brique NXT.

C. Fonctions utilisées

Tout d'abord, il faut connecter la brique NXT à l'ordinateur grâce à la fonction : ***lego = new LegoNXT(this, "COM9");*** le COM étant le port de connexion avec la brique NXT.

J'ai utilisé 4 fonctions spécifiques au robot qui sont importées grâce à la librairie NXTComm permettant de contrôler chaque moteur :

lego.motorForward(LegoNXT.MOTOR_X, POW)

➔ Cette fonction permet de dire au moteur X de tourner. Il faut remplacer X par la lettre associée au moteur que l'on veut contrôler et POW par la puissance (entre 0 et 100).

lego.motorForwardLimit(LegoNXT.MOTOR_X, POW,ROT)

➔ Cette fonction permet de dire au moteur X de tourner pendant un certain temps. Il faut remplacer X par la lettre associée au moteur que l'on veut contrôler, POW par la puissance (entre 0 et 100) et ROT par la rotation (en degrés).

lego.motorHandBrake(LegoNXT.MOTOR_X)

➔ Cette fonction permet de dire au moteur X de s'arrêter. Il faut remplacer X par la lettre associée au moteur que l'on veut contrôler.

lego.getMotorRotationCount(LegoNXT.MOTOR_B)

➔ Cette fonction permet d'obtenir le degré de rotation du moteur X. Il faut remplacer X par la lettre associé au moteur que l'on veut contrôler. Cette fonction est utilisée avec la fonction ***text*** pour pouvoir voir le degré de rotation directement.

Pour pouvoir effectuer des actions différentes en fonction de chaque lettre, j'ai utilisé des tests ***if***.

J'ai également dû utiliser différentes fonctions provenant de Processing.

Le contrôle en direct du robot utilise des touches « spéciales comme les 4 touches de direction (ARROW KEYS), CONTROL, ALT, SHIFT, TAB, BACKSPACE, ENTER…

Pour reconnaitre l'utilisation de ces touches, il faut utiliser comme condition (***key == CODED) { }*** puis à l'intérieur une autre condition ***(key == TOUCHE) { }*** pour ensuite pouvoir ordonner des actions différentes suivant la touche utilisée.

Pour les touches correspondant aux caractères, il suffit d'utiliser : ***if (key == 'lettre') { }*** en remplaçant « lettre » par la lettre choisie, ou le caractère.

D'autres fonctions ont été utilisées, notamment en rapport avec le texte, comme ***textSize()*** ou ***fill()*** mais aussi la fonction ***image()*** permettant de faire apparaître les différentes images utilisées.

Une autre fonction indispensable est la fonction ***delay()*** qui permet d'enchainer différents mouvements dans la même action.

Par exemple, pour la lettre "L", les moteurs B et C doivent d'abord effectuer une rotation de 1000 degrés, puis le moteur C s'arrête alors que le moteur B continue de 250 degrés et ensuite les moteurs B et C effectuent une rotation de 600 degrés.

❖ Difficultés

Les principales difficultés étaient d'utiliser convenablement les fonctions exclusives du robot pour réussir à avoir des trajectoires similaires aux lettres.
L'utilisation de la fonction ***delay()*** a également été compliquée car le temps que l'on associe à la fonction a une incidence directe avec les actions précédentes.

Plus la puissance avec laquelle on commande un moteur est forte, moins la rotation sera précise. En effet, après plusieurs essais, lorsqu'on demande au robot d'effectuer une trajectoire de 1000° à une puissance de 100, on remarque un écart d'environ 150°. Lorsqu'on demande au robot d'effectuer une trajectoire de 1000° à une puissance de 30, on remarque un écart d'environ 40°.

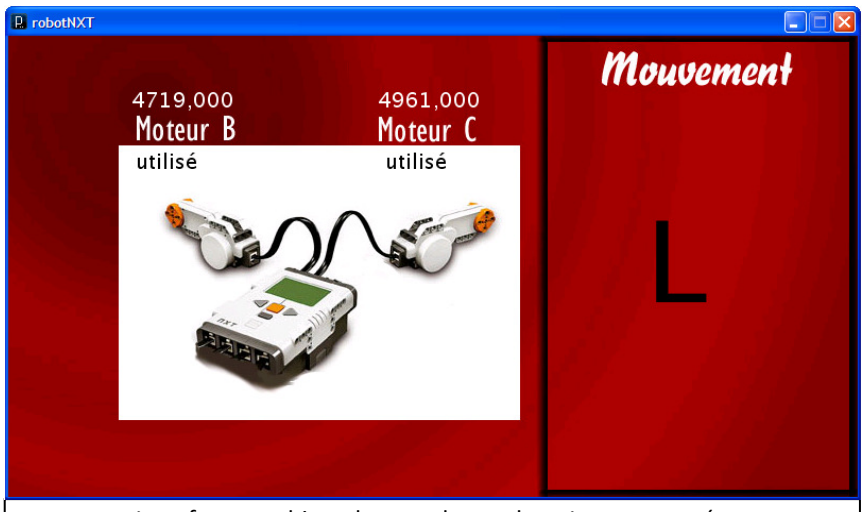
A ce jour, je n'ai pas réussi à ce que l'interface graphique indique en temps réel si le moteur est utilisé ou non lors de la réalisation des trajectoires de lettres. De plus, la lettre n'apparait que lors du dernier mouvement.

D. Réalisation finale

Au final, on peut demander au robot d'effectuer des trajectoires définies à l'avance grâce au clavier d'ordinateur (avancer, tourner, reculer, mais aussi suivre la trajectoire du L, du T, …).

Interface graphique lorsque la touche « flèche du haut » est pressée

Interface graphique lorsque la touche « CONTROL » est pressée



Interface graphique lorsque la touche « L » est pressée

E. Les limites du projet

L'utilisation du Bluetooth est dérangeante dans le sens où la communication entre le robot et l'ordinateur est souvent perdue, et il faut sans cesse les reconnecter. On ne peut atteindre une précision parfaite lors du déplacement du robot. En fonction de l'état des piles (plus ou moins chargées) utilisées dans le robot, les mouvements sont différents.

F. Pour aller plus loin

Pour continuer et améliorer ce projet, nous pourrions utiliser les capteurs que l'on peut brancher à la brique NXT, comme par exemple le capteur d'ultrasons qui permet de connaitre la distance séparant le robot d'un obstacle, pour le faire changer de direction, ou encore effectuer un mouvement en fonction de la couleur du sol où il se déplace… L'interface clavier pourrait aussi être une interface souris, qui en fonction des coordonnées de la souris sur l'écran d'ordinateur commanderait des mouvements différents. Enfin, le code pourrait être mieux modulé, il est pour l'instant trop linéaire.

BILAN

En réalisant ce projet, j'ai pu approfondir mes connaissances dans la structure de la programmation en Java. J'ai également appris que d'une brique NXT simple d'utilisation on peut créer des robots très efficaces, utiles et évolués à l'image du robot CubeStormer II qui a battu le record « humain » de Rubik's cube.

Ce travail permet de comprendre l'importance de fonctionner en équipe dans le cadre d'un projet :

- La communication nous a permis d'avoir l'esprit critique, d'échanger les idées et de les confronter, facilitant ainsi la découverte de solutions
- Partager les connaissances et les idées permet de progresser individuellement et de s'offrir un plus grand savoir.

G. Diffusion du projet

1. Choix de licence Creative Commons

J'ai choisi comme licence de diffusion **Creative Commons : attribution, pas d'utilisation commerciale**. Par ce choix de licence, j'accepte que ce projet soit modifié sans autoriser la commercialisation du projet. En effet, je pense qu'un projet réalisé pour le lycée est largement perfectible. De plus, l'utilisation du robot NXT étant majoritairement éducative, il me parait plus intéressant de laisser ce projet en Open source.

2. Utilisation de Github

J'ai utilisé Github pour partager mes fichiers avec les professeurs mais aussi les autres élèves de la spécialité ISN du lycée Aristide Briand.

Lien Github

III. Annexe

Une partie du code…

```
import processing.serial.*; // importation de toutes les librairies nécessaires
import pt.citar.diablu.nxt.protocol.*;
import pt.citar.diablu.processing.nxt.*;
import pt.citar.diablu.nxt.brick.*;

LegoNXT lego; // appellation du robot

void setup() {
 size(800, 450); // taille de la fenêtre qui va apparaitre à l'écran (en pixels)
 lego = new LegoNXT(this, "COM9"); // connexion avec le robot
 frameRate(5); // temps de rafraîchissement
}



void draw() {
  fill(255); // le texte apparait de couleur blanche, jusqu'au prochain fill, ou jusqu'à la
             prochaine boucle
 text(lego.getMotorRotationCount(LegoNXT.MOTOR_B),110,70); // afficher la rotation du
                                                           moteur B aux coordonnées (110;70)
 text(lego.getMotorRotationCount(LegoNXT.MOTOR_C),342,70); // afficher la rotation du
                                                           moteur C aux coordonnées (342;70)
}
```

```
Void keyPressed() {

if (key == 'z' | key == 'Z') { // si la touche Z est pressée

   image(fond, 0, 0);       // réinitialisation du fond

   fill(0);                 //couleur du texte noire

   textSize(110);           // taille du texte suivant

   text("Z", 600, 260);      // "Z" est affiché aux coordonnées (600;260)

   textSize(20);            // taille du texte suivant

   text("utilisé", 120, 130); // "utilisé" est affiché aux coordonnées (120 ;130)

   text("utilisé", 355, 130); // "utilisé" est affiché aux coordonnées (355 ;130)

   lego.motorForwardLimit(LegoNXT.MOTOR_B, 100, 1000);  // le moteur B tourne en
                  sens horaire pendant 1000 degrés à une puissance de 100 sur 100

   lego.motorForwardLimit(LegoNXT.MOTOR_C, 100, 1000); // le moteur C tourne en
                  sens horaire pendant 1000 degrés à une puissance de 100 sur 100

   delay(2000);             // pause de 2000 millisecondes

   lego.motorHandBrake(LegoNXT.MOTOR_B); // moteur B est arrêté

   lego.motorForwardLimit(LegoNXT.MOTOR_C, 100, 500); //moteur C tourne en sens
                  horaire pendant 500 degrés à une puissance de 100 sur 100

   delay(2300);             // pause de 2300 millisecondes

   lego.motorForwardLimit(LegoNXT.MOTOR_B, 100, 1200); // le moteur B tourne en
                  sens horaire pendant 1200 degrés à une puissance de 100 sur 100

   lego.motorForwardLimit(LegoNXT.MOTOR_C, 100, 1200); // le moteur C tourne en
                  sens horaire pendant 1200 degrés à une puissance de 100 sur 100

   delay(2300); // pause de 2300 millisecondes

   lego.motorHandBrake(LegoNXT.MOTOR_C); // moteur C est arrêté

   lego.motorForwardLimit(LegoNXT.MOTOR_B, 100, 500); // le moteur C tourne en
                  sens horaire pendant 500 degrés à une puissance de 100 sur 100

   delay(2300); // pause de 2300 millisecondes

   lego.motorForwardLimit(LegoNXT.MOTOR_B, 100, 1000); // le moteur B tourne en
                  sens horaire pendant 1000 degrés à une puissance de 100 sur 100

   lego.motorForwardLimit(LegoNXT.MOTOR_C, 100, 1000); // le moteur C tourne en
                  sens horaire pendant 1000 degrés à une puissance de 100 sur 100

    }
```

```
if (key == CODED) {
   if (keyCode == UP) {            //si la touche "flèche du haut" est pressée

     lego.motorForward(LegoNXT.MOTOR_B, 100); // moteur B est mis en marche, vers
                                                l'avant, puissance 100
     lego.motorForward(LegoNXT.MOTOR_C, 100); // moteur C est mis en marche, vers
                                                l'avant, puissance 100
     image(fond, 0, 0);            //réinitialisation de l'interface graphique
     image(haut, 600, 180);        // l'image "haut" apparait aux coordonnées (600;180)
     fill(0);                      // texte de couleur noire
     textSize(20);                 //taille de police du texte
     text("utilisé", 120, 130);        // le texte "utilisé" apparait aux coordonnées (120;130)
     text("utilisé", 355, 130);        // le texte "utilisé" apparait aux coordonnées (355;130)
    }
 }
if (key == '_') { //si la touche "_" est pressée
  lego.motorForward(LegoNXT.MOTOR_B, 100); //moteur B est mis en marche, vers l'avant,
                                             puissance 100
  lego.motorForward(LegoNXT.MOTOR_C, 100); //moteur C est mis en marche, vers l'avant,
                                             puissance 100
 }

 if (key == 'ç') { // si la touche "ç" est pressée
   lego.motorForward(LegoNXT.MOTOR_B, -100); //moteur B est mis en marche, vers
                                               l'arrière, puissance 100
   lego.motorForward(LegoNXT.MOTOR_C, -100); //moteur C est mis en marche, vers
                                               l'arrière, puissance 100
 }
 if (key == 'è') { // si la touche "è" est pressée
   lego.motorForward(LegoNXT.MOTOR_B, 100); //moteur B est mis en marche, vers
                                              l'avant, puissance 100
```

```
    lego.motorHandBrake(LegoNXT.MOTOR_C); //moteur C est arrêté
  }

  if (key == 'à') { // si la touche "à" est pressée
    lego.motorForward(LegoNXT.MOTOR_C, 100); //moteur C est mis en marche, vers
                                                l'avant, puissance 100
    lego.motorHandBrake(LegoNXT.MOTOR_B); //moteur B est arrêté
  }
void keyReleased() {  // fonction permettant d'utiliser le fait qu'une touche soit relâchée
     if (key == '_' | key == 'ç' | key == 'è' | key == 'à') { // si l'une de ces touches est relâchée
     lego.motorHandBrake(LegoNXT.MOTOR_C); // Moteur C est arrêté
     lego.motorHandBrake(LegoNXT.MOTOR_B); // Moteur B est arrêté
     image(fond, 0, 0); // Réinitialisation du fond
     image(stop, 600, 180); // L'image stop s'affiche aux coordonnées (600 ;180)
     fill(0); // Texte de couleur noire jusqu'au prochain fill ou jusqu'à la prochaine boucle
     textSize(20); // Taille de police du texte 20
     text("repos", 120, 130); // Le texte repos s'affiche aux coordonnées (120 ;130)
     text("repos", 355, 130); // Le texte repos s'affiche aux coordonnées (355 ;130)
    }

}
```

Interface graphique au lancement de l'application