SIMULATION DU ROBOT LEGO EV3

Caractéristiques de la simulation:

Fenêtre de simulation: 501x501 pixels

échelle distance = 200 pixels par mètre; $0 \le \text{vitesse linéaire} \le 100$

vitesse linéaire par défaut = 50

Avec une vitesse de 50, le robot parcourt un pixel en

A une vitesse de 10, le robot tourne de un degré en 30 ms.

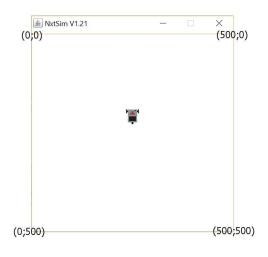
<u>Caractéristiques des capteurs:</u>

capteur son -->
 0: silence,
 150: saturation
capteur lumière -->
 0: foncé,
 1000: brillant

capteur ultrason -->
 -1 pas de détection d'objet ou robot est dans

l'objet,

0-255 distance de l'objet en pixels



Coordonnées de référence:

point en haut à gauche: (0;0), point en haut à droite (500;0), point en bas à gauche (0;500), point en bas à droite (500;500).

Le robot peut sortir de la fenêtre.

Dans la réalité, la lumière naturelle peut perturber les mesures des capteurs de lumière et de couleurs. Le capteur ultrason réel renvoie une distance en cm entre 0 et 255. La distance 255 indique la non détection.

Eléments indispensables du programme pour utiliser le robot de la simulation:

L'importation de la librairie ev3:

from simrobot import *

La déclaration du robot:

LegoRobot robot = LegoRobot()

La déclaration du moteur:

Gear gear = Gear()

L'installation du moteur sur le robot:

robot.addPart(gear)

LegoRobot robot = LegoRobot()	Déclaration du robot de nom <i>robot</i>
Gear gear = Gear()	Déclaration du moteur de nom <i>gear</i>
robot.addPart(gear)	Ajout du moteur sur le robot
gear.stop()	Arrêt du robot
gear.setSpeed(10)	On règle la vitesse de déplacement du robot (par défaut: 50)

gear.forward()	On démarre le mouvement avant.
gear.forward(100)	Le robot avance pendant 100 millisecondes.
gear.backward()	On démarre le mouvement de recul.
gear.backward(100)	Le robot recule pendant 100 millisecondes.

gear.right()	On démarre le mouvement de rotation droite .
gear.right(90)	Le robot tourne à droite pendant 90 millisecondes.
gear.rightArc(12.5)	Le robot démarre vers la droite un arc de cercle de rayon 12,5m.
gear.rightArc(24,500)	Le robot décrit vers la droite un arc de cercle de rayon 24m pendant 500 ms.
gear.left()	On démarre le mouvement de rotation gauche .
gear.left(90)	Le robot tourne à gauche pendant 90 millisecondes.
gear.leftArc(2.5)	Le robot démarre vers la gauche un arc de cercle de rayon 2,5 m.
gear.leftArc(0.9,500)	Le robot décrit vers la gauche un arc de cercle de rayon 0,9 m pendant 500 ms.

Capteur de contact

TouchSensor touch = TouchSensor(SensorPort.S1)	On déclare le capteur de contact de nom touch. S1: front right, S2: front middle, S3: front left , S4: rear-middle
TouchSensor(SensorPort.port, pressed = onPressed, released = onReleased)	Idem, en rattachant les gestionnaires d'événements onPressed(port), onReleased(port)
robot.addPart(touch)	Ajout du capteur de contact sur le robot
touch.isPressed()	Détection du contact (booléen)

Capteur de lumière

LightSensor <i>light</i> = LightSensor(SensorPort. <i>S3</i>)	On déclare le capteur de lumière de nom light. S1: front right, S2: front middle, S3: front left , S4: rear-middle
LightSensor <i>light</i> = LightSensor(SensorPort. <i>S3</i> ,dark = onDark)	On déclare le capteur de lumière de nom <i>light</i> . S1: front right, S2: front middle, S3: front left , S4: rear-middle Enregistre la fonction de rappel onDark
LightSensor <i>light</i> = LightSensor(SensorPort. <i>S3</i> ,bright = onBright)	On déclare le capteur de lumière de nom <i>light</i> . S1: rear right, S2: rear-middle, S3: rear left, S4: front middle Enregistre la fonction de rappel onBright
robot.addPart(light)	Ajout du capteur de lumière sur le robot
light.getValue()	Valeur de la réflexion de la lumière sur le sol (entier)

Capteur de son

SoundSensor sound = SoundSensor(SensorPort.S1)	On déclare le capteur de son de nom sound. S1: right, S2: middle, S3: left , S4: left
robot.addPart(sound)	Ajout du capteur de son sur le robot
sound.getValue()	Valeur du son (entier)

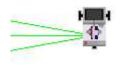
Capteur ultrason

UltrasonicSensor ultraSon = UltrasonicSensor (SensorPort.S4)	On déclare le capteur ultrason de nom <i>ultrason</i> . S1: forward; S2: left, S3: backward, S4: right.
robot.addPart(ultrason)	Ajout du capteur ultrason sur le robot
ultraSon.getDistance()	Distance entre le robot et l'objet (entier) (-1= pas de détection)

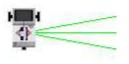
Positions du capteur ultrason



Sensorport.S1







Sensorport.S4

Sensorport.S2

Sensorport.S3

Capteur de couleurs

ColorCube de dimension n = ensemble des couleurs dont les composantes red, green et blue sont éloignées de \pm n de celles de la couleur recherchée.

ColorSensor cs = ColorSensor(SensorPort.S3)	On déclare le capteur de couleurs de nom cs. S1: front right, S2: front middle, S3: front left , S4: rear-middle
robot.addPart(cs)	Ajout du capteur de couleurs sur le robot
cs.getColor()	Valeur de la couleur (nom)
cs.getColorID()	Retourne l'identifiant de la couleur actuellement mesurée : 0: indefini, 1: noir, 2: bleu, 3:vert, 4: jaune, 5: rouge, 6: blanc
cs.getColorStr()	Retourne la couleur mesurée par une chaine de caractères (BLACK, BLUE, GREEN, YELLOW, RED, WHITE ou UNDEFINED)
cs.getLightValue()	Retourne la luminosité de la couleur mesurée selon le modèle TSL = Teinte, Saturation, Lumière, (HSG = Hue, Saturation, Lightness)
cs.inColorCube(colorCube)	Donne true si la couleur captée est dans le colorCube. colorCube=[red_min, red_max, green_min, green_max, blue_min, blue_max]
cs.getColor().getRed()	Renvoie la valeur du rouge entre 0 et 255.
cs.getColor().getGreen()	Renvoie la valeur du vert entre 0 et 255.
cs.getColor().getBlue()	Renvoie la valeur du bleu entre 0 et 255.

Jouer un son

-		
- 1	1 . 1 - /6	
	robot play lone (treduency duration)	Joue un son de fréquence frequency (en Hz) et de durée duration (ms)
	robot.play rolle(inequelity, duration)	Joue an John de n'equence n'equency (en 112) et de durée duration (1113)

Ecran LCD du robot

L'écran de l'EV3 est composé de 8 lignes et 18 colonnes)

 $(0 \le n^{\circ} colonne \le 17 \ 0 \le n^{\circ} ligne \le 7)$

Pour la simulation, ajouter RobotContext.showStatusBar(hauteur)

robot.drawString(texte,n°colonne,n°ligne)	Affiche un texte (String) sur l'écran à la position (n°colonne,n°ligne)
robot.drawString(str(entier),n°colonne,n°ligne)	Affiche un entier (int) sur l'écran
robot.drawString(str(réel),n°colonne,n°ligne)	Affiche un réel (double) sur l'écran
robot.drawString("angle="+str(angle),0,4)	Affiche sur l'écran angle= suivi de la valeur de la variable angle
robot.clear();	Efface l'écran.(sur EV3 uniquement)

Environnement du robot

RobotContext.setStartPosition(x, y)	Position initiale du robot (x;y) coordonnées entières (point haut gauche: (0;0), point bas droite (500,500)
RobotContext.setStartDirection(angle)	Direction initiale du robot (entier) (Est=0, Sud=90, Ouest=180, Nord=270)
RobotContext.useObstacle("sprites/bar0.gif, x, y)	Installation de l'obstacle (image) pour le capteur de contact. (x,y) position du centre de l'image.
RobotContext.useBackground("sprites/whiteCircle.gif")	Installation de sol (image) pour les capteurs de lumière et de couleurs.
RobotContext.useTarget("sprites/target_red.gif", mesh, x, y)	Installation de l'objet (image) pour le capteur ultrason. (x,y) position du centre de l'image.

Simulation des boutons de la brique EV3 avec les touches du clavier

robot.isEnterHit()	Vaut true si le bouton Enter est appuyé.
robot.isEscapeHit()	Vaut true si le bouton Escape est appuyé.
robot.isUpHit()	Vaut true si le bouton Up ↑ est appuyé.
robot.isDownHit()	Vaut true si le bouton Down ↓ est appuyé.
robot.isRightHit()	Vaut true si le bouton Right → est appuyé.
robot.isLeftHit()	Vaut true si le bouton Left ← est appuyé.
robot.isButtonHit()	Vaut true si l'un des boutons précédents est appuyé.

