

SIMULATION DU ROBOT LEGO EV3

Caractéristiques de la simulation:

Fenêtre de simulation: 501x501 pixels

échelle distance = 200 pixels par mètre;
 $0 \leq \text{vitesse linéaire} \leq 100$
vitesse linéaire par défaut = 50

Avec une vitesse de 50, le robot parcourt un pixel en 20 ms.
A une vitesse de 10, le robot tourne de un degré en 30 ms.

Caractéristiques des capteurs:

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.

Éléments 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)
```

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

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

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

Capteur de contact

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

Capteur de lumière

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

Capteur de son

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

Capteur ultrason

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

Positions du capteur ultrason



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

Jouer un son

<i>robot.playTone(frequency, duration)</i>	Joue un son de fréquence frequency (en Hz) et de durée duration (ms)
--	--

Ecran LCD du robot

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

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

Pour la simulation, ajouter RobotContext.showStatusBar(hauteur)

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

Environnement du robot

RobotContext.setStartPosition(<i>x</i> , <i>y</i>)	Position initiale du robot (<i>x</i> ; <i>y</i>) coordonnées entières (point haut gauche: (0;0), point bas droite (500,500))
RobotContext.setStartDirection(<i>angle</i>)	Direction initiale du robot (entier) (Est=0, Sud=90, Ouest=180, Nord=270)
RobotContext.useObstacle("sprites/bar0.gif", <i>x</i> , <i>y</i>)	Installation de l'obstacle (image) pour le capteur de contact. (<i>x</i> , <i>y</i>) 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 , <i>x</i> , <i>y</i>)	Installation de l'objet (image) pour le capteur ultrason. (<i>x</i> , <i>y</i>) position du centre de l'image.

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

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

