

Thymio II

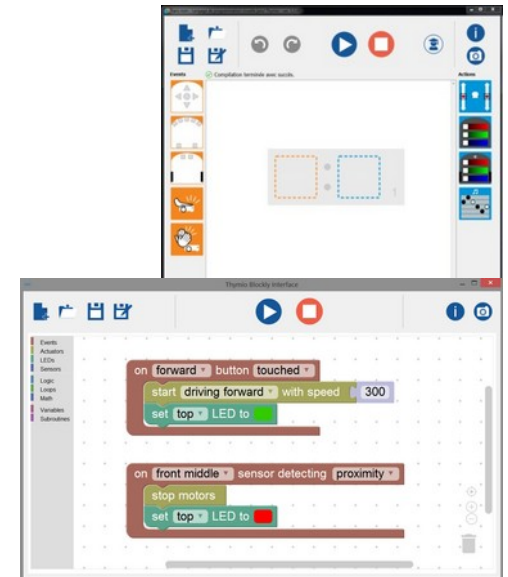
de la graine de robot



Fête de la science

Octobre 2025

O. Buffet, P. Ngo, A. Rousseau,
A. Scheuer & V. Thomas





Plan

- Description
origine, capacités et utilisations
- Matériel
capteurs, moteurs & autres actionneurs
- Logiciel (Aseba)
VPL (prog. visuelle), Blockly (~ Scratch) ou scripts
- Application
amélioration des comportements fournis



Un peu d'histoire

Quelques robots pour l'enseignement et la recherche



Khepera III
(2005)

Thymio I
(2008)



Thymio II
(2011)



Khepera IV
(2014)





Capacités du Thymio II

Thymio II peut

- Détecter des objets
- Bouger
- Produire des sons
- Changer de couleur
- Réagir aux boutons
- Sentir des accélérations
- Mesurer bruits et température
- Émettre / recevoir un signal





Activités prévues

- Découvrir et utiliser les comportements prédéfinis

- Comprendre ces comportements
- Les exploiter pour résoudre des tâches

P

- Créer des comportements simples
Lier perceptions et actions en programmant

C
L

- Améliorer les comportements existants
Analyser et optimiser un programme

S



Plan

- Description
origine, capacités et utilisations
- Matériel
capteurs, moteurs & autres actionneurs
- Logiciel (Aseba)
VPL (prog. visuelle), Blockly (~ Scratch) ou scripts
- Application
amélioration des comportements fournis



Capteurs

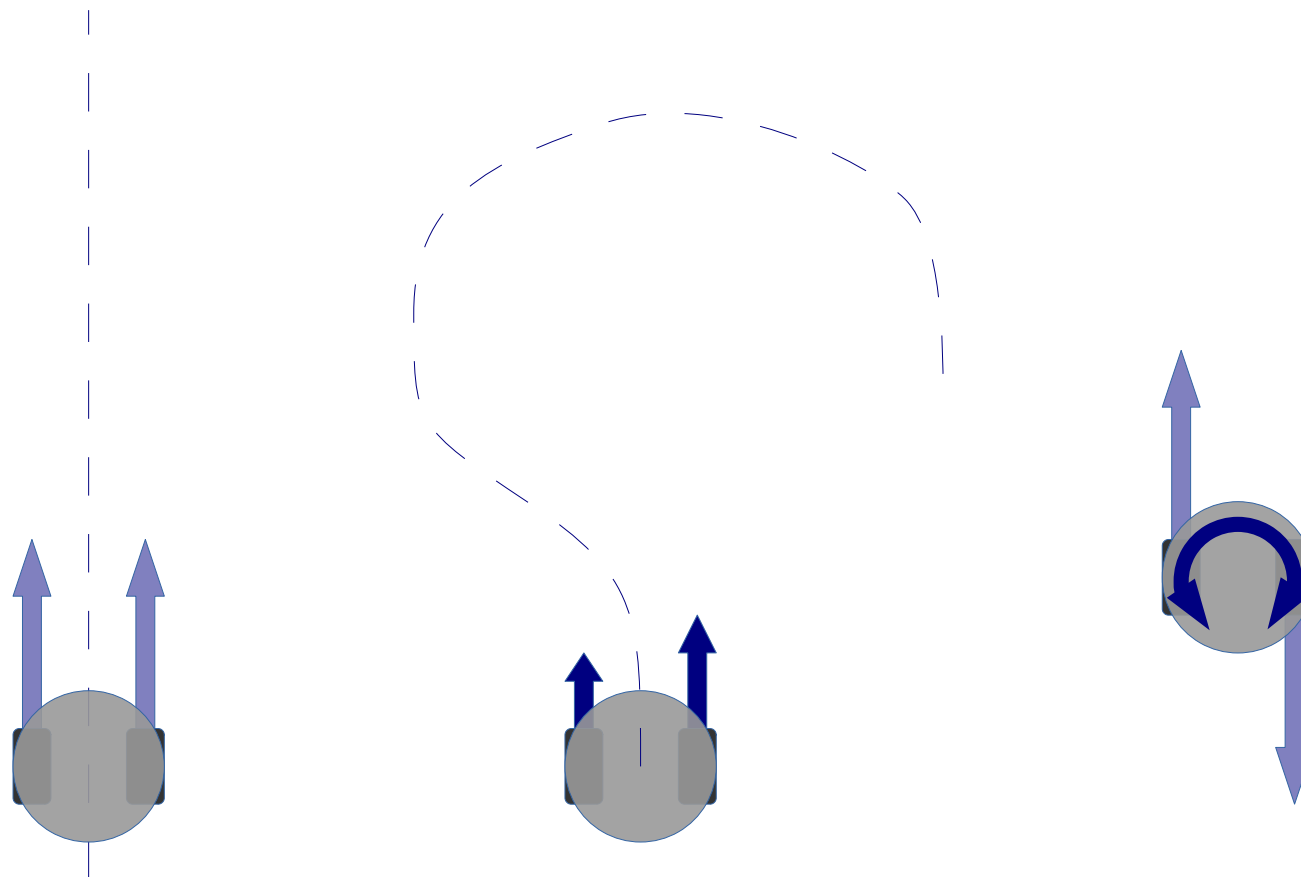
- Boutons, accéléromètre, microphone & thermomètre
- Capteurs infra-rouge
 - devant (x 5), derrière (x 2) & dessous (x 2)
 - portée : ~ 20 cm
 - fréquence : 10 Hz (100 ms)
 - sensibles à la couleur et à la texture





Moteurs

Thymio II = 2 roues indépendantes





Autres actionneurs

- Diodes de couleur

intensité (0 – 32) ou couleur (3 intensités)

10 IR (i), 4 boutons (i), 8 cercle (i), 2 dessous (c),
1x2 dessus (c), 1 son (i), 1 temp. (c), 1 émet. (i)

- Haut-parleur

sons prédéfinis, 1 fréquence, 1 vague ou 1 fichier

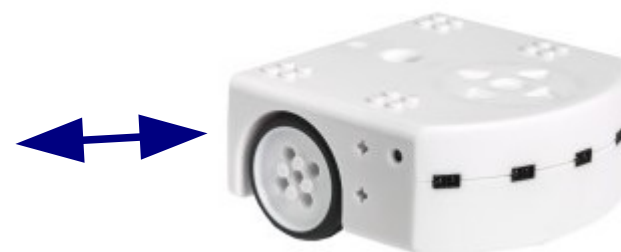
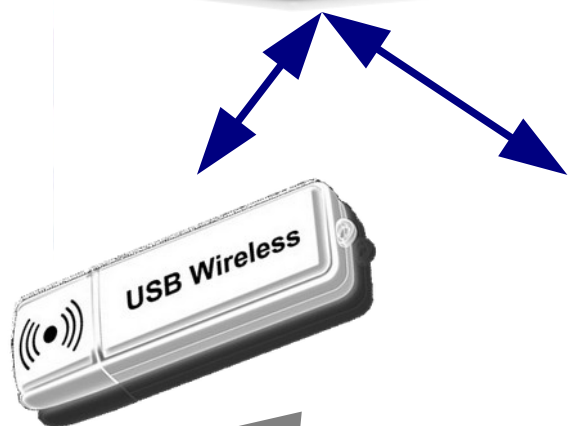
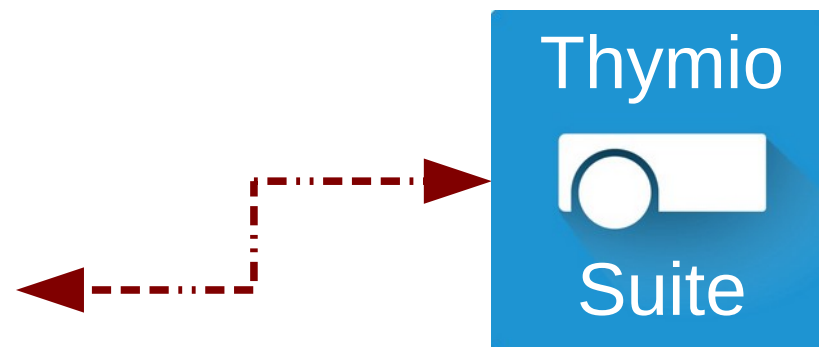


Plan

- Description
origine, capacités et utilisations
- Matériel
capteurs, moteurs & autres actionneurs
- Logiciel (Aseba)
VPL (prog. visuelle), Blockly & Scratch ou scripts
- Application
amélioration des comportements fournis

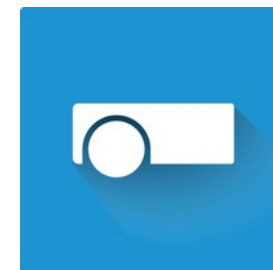


Comment programmer ?





La suite Thymio



- Développé par l'EPFL
- Logiciel libre ("*open-source*" - code fourni)



- Disponible sur Linux, MacOS et Windows



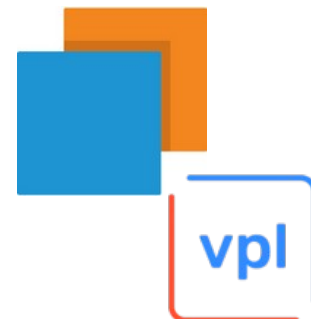
- Programmation selon plusieurs approches





Programmation visuelle

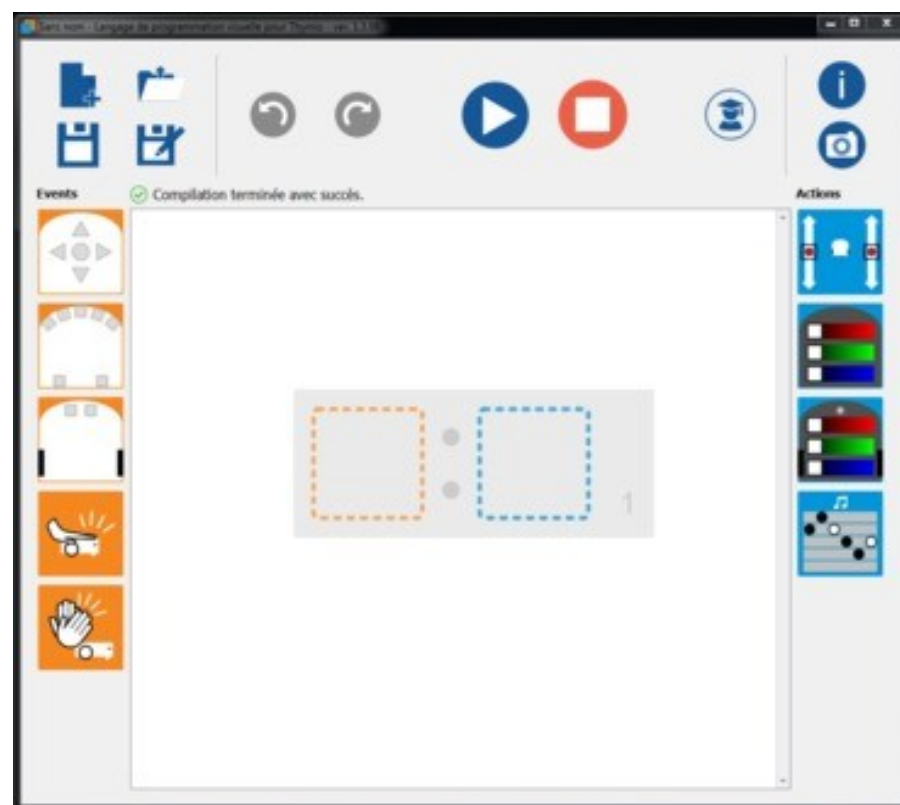
VPL & VPL 3



Indique quelles actions accomplir lorsque certaines conditions sont perçues

- Les perceptions sont placées à gauche (fond orange)
- Les actions sont données à droite (fond bleu)

⚠ Logique ! ⚠





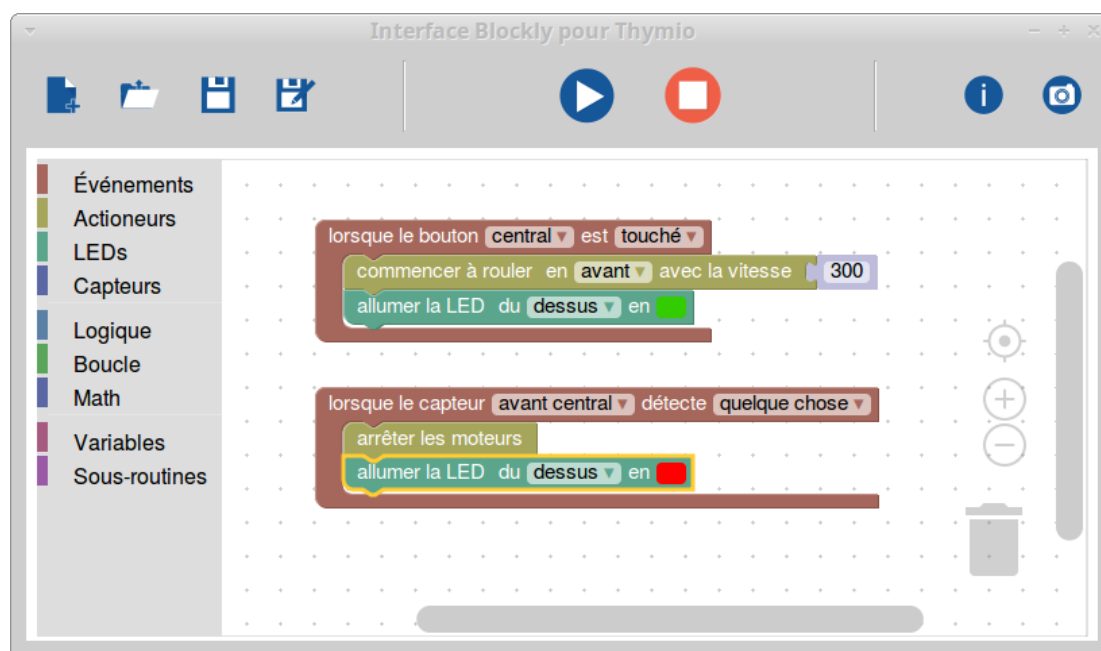
Scratch



Scratch et Blockly

Blockly similaire à Scratch
(approche événementielle et non séquentielle)

- On assemble des blocs pour programmer
- Les paramètres sont choisis dans des menus déroulants





Programmer par scripts

Aseba : mode présent sous VPL & Blockly

Commandes

Données
capteur &
variables

The screenshot shows the Aseba Studio interface for programming a Thymio II robot. The main window displays a script for controlling the robot. The script is written in a simple, human-readable language and includes comments in French. The script defines variables for the robot's position, speed, and direction, and uses a loop to continuously update these values based on sensor data. The script is saved as 'thymio_donne.asei'.

Variables

Nom	valeurs
event source	8898
event args	(32)
button.backward	0
button.left	0
button.center	0
button.forward	0
button.right	0
prox.horizontal	(7)
0	0
1	0
2	0
3	1851
4	1840
5	1666
6	1319
prox.comen.rx	0
prox.comen.tx	0
prox.ground.ambient	(2)
prox.ground.reflected	(2)
prox.ground.delta	(2)
motor.left.target	0
motor.right.target	0
motor.left.speed	-3
motor.right.speed	0
motor.left.pwm	0
motor.right.pwm	0
acc	(3)
temperature	231
rcb.address	0
rcb.command	0
rcb.status	0

Code

```
1  /*
2  * Fichier définissant le contrôle d'un Thymio 2
3  *
4  * À vous de jouer :
5  * - normalisation des données et
6  * - calcul des vitesses de roues souhaitées
7  *
8  */
9
10 # numéro du capteur
11 var cpt
12 # donnée capteur normalisée
13 var npi
14 # coupe le timer 0
15 timer.period[0] = 0
16
17 # Boucle principale (liée au timer 0)
18 onevent timer0
19 # Les capteurs vont de 0 à 6, de l'avt G à l'arr D
20 for cpt in 0:6 do
21   npi = prox.horizontal[cpt] # vous pouvez normaliser la valeur
22   # mettez ici vos calculs
23 end
24 # vitesses souhaitées (entre -500 et 500)
25 motor.left.target = 100
26 motor.right.target = 100
27
28 # Appuyer sur le bouton avant lance la boucle principale
29 onevent button.forward
30 timer.period[0] = 100 # fréquence de la boucle (100 ms)
31
32 # Appuyer sur le bouton central arrête le robot
33 onevent button.center
34 timer.period[0] = 0 # stoppe le timer 0
35 motor.left.target = 0 # arrête le robot
36 motor.right.target = 0
37
```

Compilation

Compilation terminée avec succès. ✓

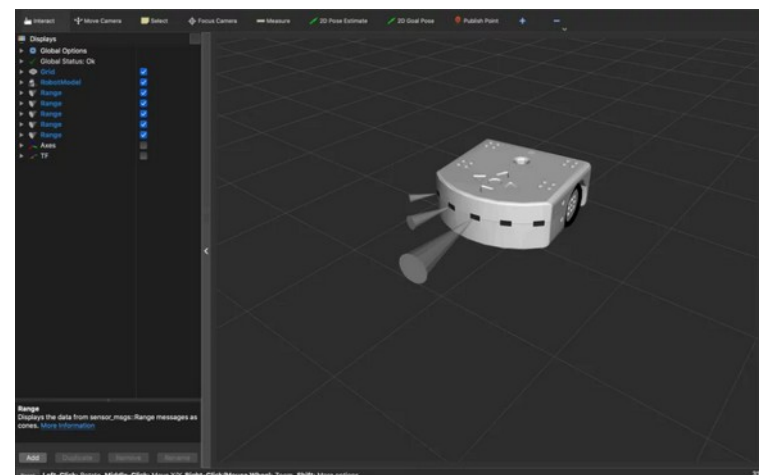


Encore plus !

- Langage Python (version ≥ 2.4)
 - Éditeur Thonny intégré à la suite
 - Peut être configuré pour Jupyter Lab
- Interaction avec d'autres robots
 - Environnement ROS
 - Paquets `ros-aseba` et `ros-thymio`
 - Simulation dans Gazebo



ROS





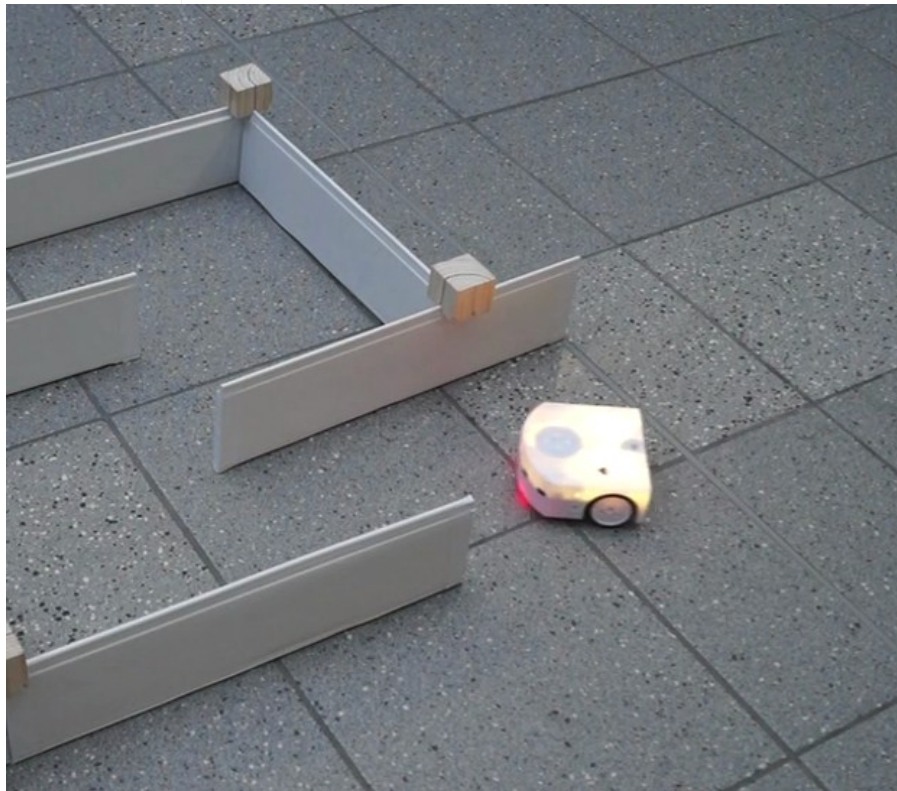
Plan

- Description
origine, capacités et utilisations
- Matériel
capteurs, moteurs & autres actionneurs
- Logiciel (Aseba)
VPL (prog. visuelle), Blockly (~ Scratch) ou scripts
- Application
amélioration des comportements fournis



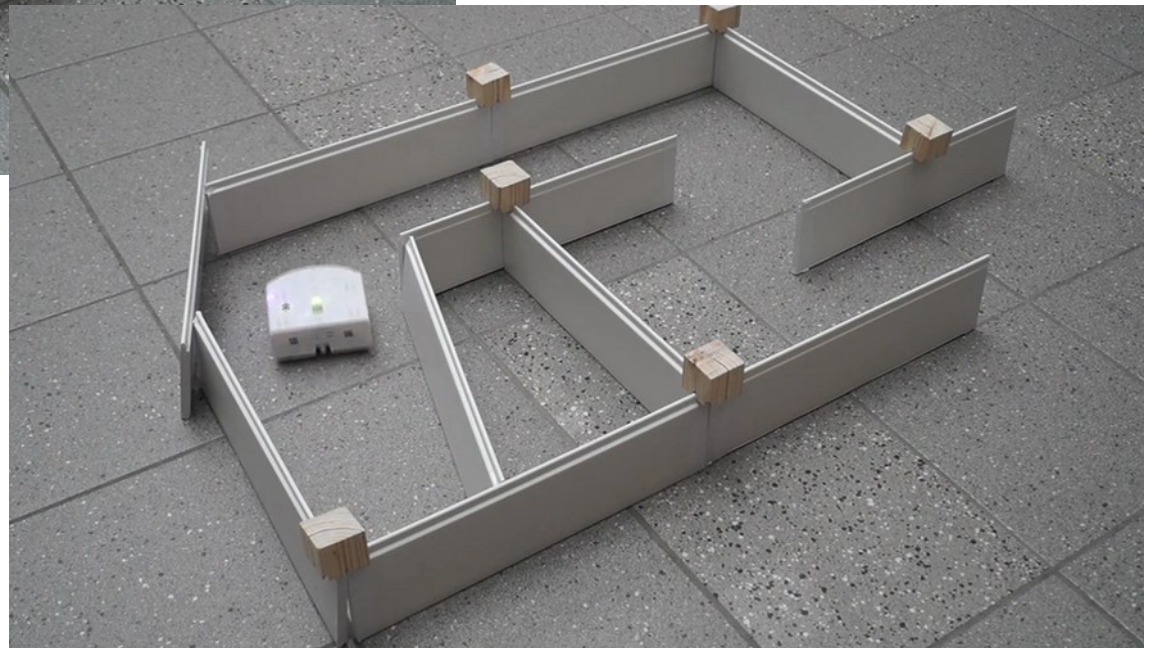
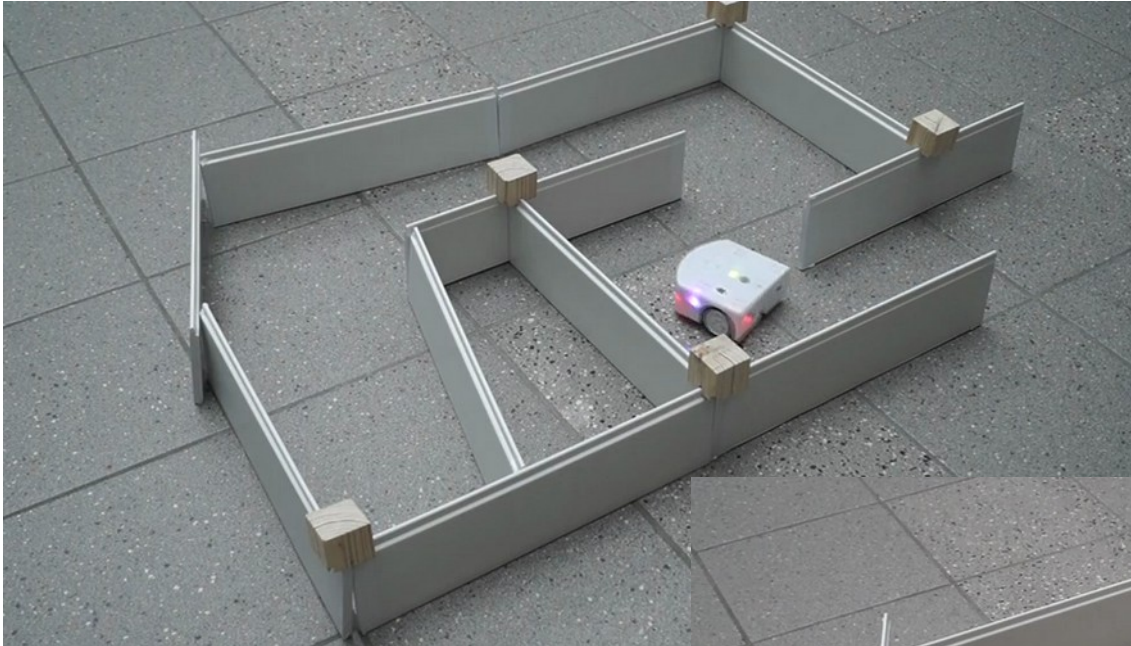
Pourquoi ? (Motivation)

Comportements fournis **peu efficaces !**





Évitement d'obstacles





Suivi et convoi





À vous de jouer !

Des questions ?

Bonnes activités...