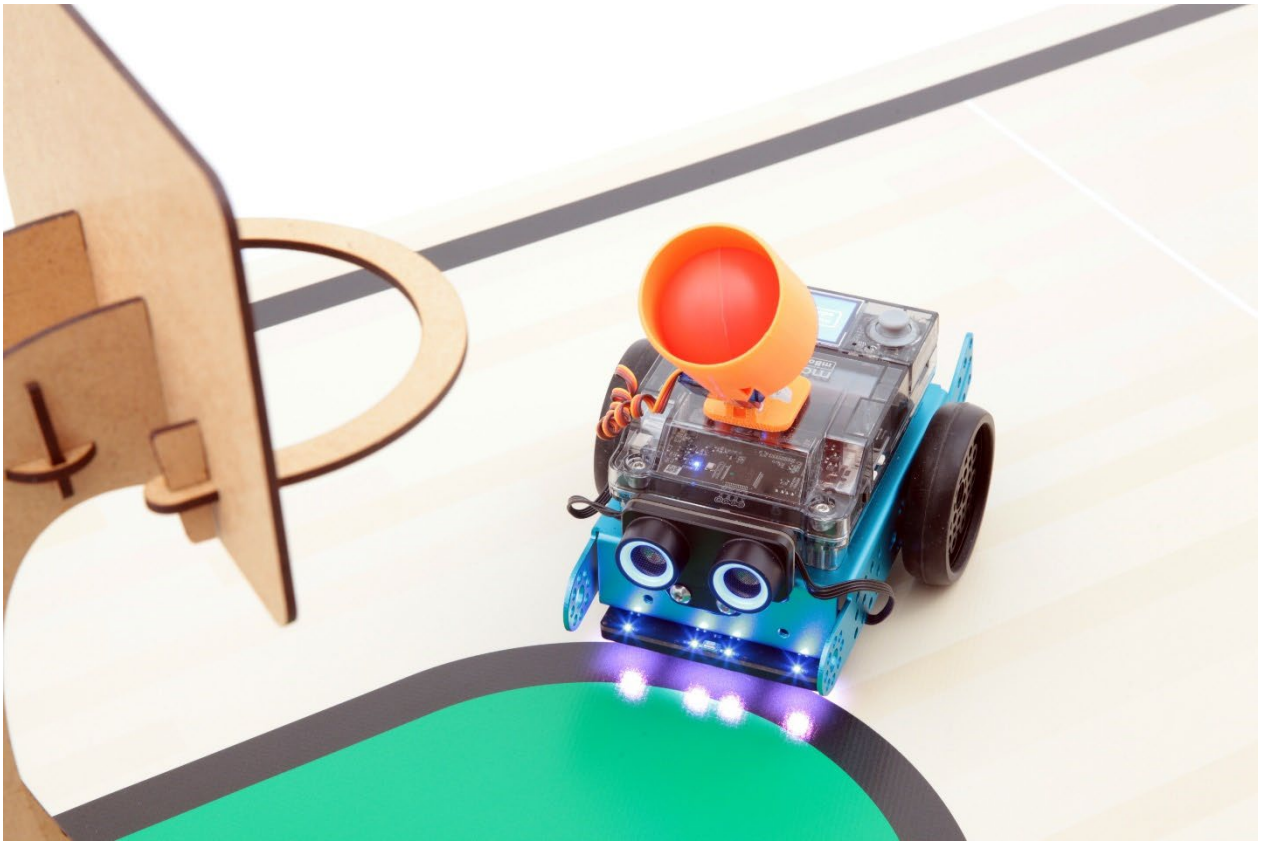
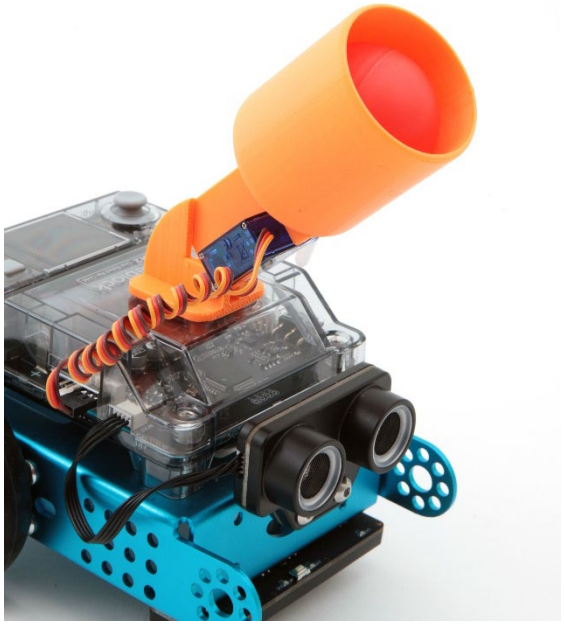
D-BAB

DéfiBasket

Terrain de basket pour matchs robotiques avec mBot2





Edité par la société A4 Technologie
5 avenue de l'Atlantique - 91940 Les Ulis
Tél. : 01 64 86 41 00 - Fax : 01 64 46 31 19
www.a4.fr

RESSOURCES LIBRES

Les documents techniques et pédagogiques signés A4 Technologie sont diffusés librement sous licence Creative Commons BY-NC-SA :

– **BY** : Toujours citer A4 Technologie comme source (paternité).

– **NC** : Aucune utilisation commerciale ne peut être autorisée sans l'accord de A4 Technologie.

– **SA** : La diffusion des documents modifiés ou adaptés doit se faire sous le même régime.

Consulter le site ***https://creativecommons.org/***

Logiciels, programmes, manuels utilisateurs téléchargeables gratuitement sur www.a4.

mBot 2
Robot modulaire,
programmable avec mBlock5

Pack DéfiBasket
Terrain et accessoires
pour matchs de basket
robotiques

SOMMAIRE

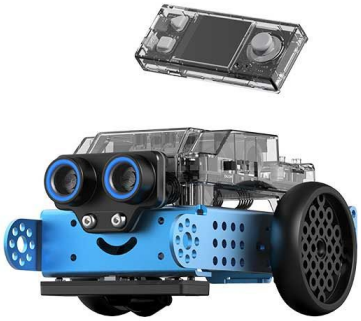
Introduction

DéfiBasket

DéfiBasket est une enceinte robotique conçue afin de confronter des robots programmés pour marquer des paniers de manière autonome. Sa conception et les couleurs utilisées sont adaptées à la mise en œuvre du robot mBot2 équipé avec son capteur de ligne et son capteur de distance. Le lanceur de balle qui équipe le robot est réalisé en impression 3D. Ce système mécanique constitue à lui tout seul un objet d'étude intéressant. Il peut être amélioré pour optimiser la précision du tir et la trajectoire de la balle.

Ce dossier propose un exemple pour réaliser ce défi avec un robot mBot2 programmé en blocs avec le logiciel mBlock 5. Pour marquer un panier, le robot doit **rester dans l'aire de jeu**, **localiser le panier adverse, ajuster la distance de tir** par rapport au panier puis **lancer la balle** dans sa direction.
Les élèves peuvent s'organiser en plusieurs groupes pour réaliser chaque partie du programme, maîtriser le fonctionnement du lanceur de balle et mettre en commun leurs réalisations pour rendre le robot opérationnel.

Matériels disponibles sur www.a4.fr



Robot mBot2
Réf. MB-P1010132

Lance balle DEFI-BASKET pour mBot2
Réf. BAB-LB-MB2

Manette Bluetooth controller
Réf. MB-P3060003

Prérequis

Les exemples d'activités de programmation proposés dans ce dossier sont basés sur l'utilisation de robot mBot2 équipé avec son capteur de distance, son module capteur de ligne RGB et son lanceur de balle.

- Disposer d'un robot mBot2, de son câble de programmation ou de la clé (dongle) Bluetooth Makeblock pour téléverser les programmes dans le robot
- Disposer d'un terrain DéfiBasket
- Installer le logiciel mBlock 5 et maîtriser son utilisation avec le robot mBot2
- Mettre en service les extensions des modules mBuild utilisés avec le robot (télémètre à ultrasons, capteur de lignes)
- Maîtriser le fonctionnement des capteurs et actionneurs utiles pour faire évoluer le robot de manière autonome afin de marquer un panier

Note : les exemples proposés peuvent être transposés à d'autres environnements de programmation et d'autres robots.

Environnement de programmation mBlock 5

L'environnement de programmation mBlock 5 peut être installé sur un ordinateur (Windows ou Mac). Il est également disponible en ligne sur navigateur Chrome (Windows/Mac/Linux/Chromebook) ; dans ce cas l'application mLink doit être installée sur votre poste pour permettre la communication et le transfert des programmes sur le mBot.

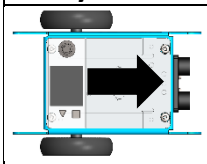
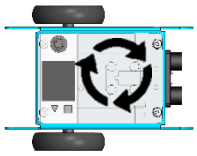
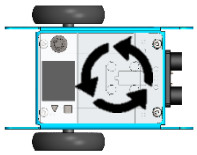
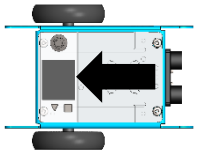
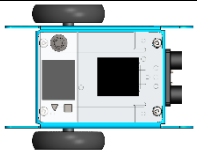
https://mblock.makeblock.com/en-us/download/

Utilisation du dossier

Ce dossier propose une série d'exemples de programmes qui peuvent servir de base pour réaliser le défi et être revisités pour en améliorer les performances. Les fichiers de ces programmes ainsi que les fichiers de modélisation 3D du robot et de ses modules sont téléchargeables dans la rubrique DéfiBasket sur www.a4.fr

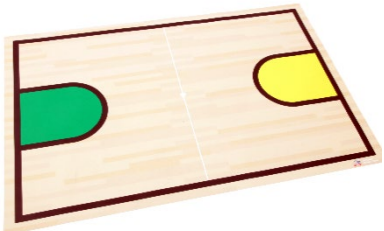
- **Présentation des matériels**
  Ce chapitre recense les éléments utilisés dans ce dossier.
- **Prise en main du robot mBot2**
  Ce chapitre propose une série d'exemples destinés à maîtriser chaque fonctionnalité utile à la réalisation du défi. Les programmes proposés sont classés par ordre de difficulté croissante.
- **DéfiBasket - Marquer un but de manière autonome**
  Ce chapitre propose une solution qui s'appuie sur l'utilisation des modules de base qui équipent le robot (capteur de distance et capteur de ligne RGB)
- **Activités complémentaires**
  Ce chapitre propose d'aller plus loin dans les activités de programmation et d'amélioration mécaniques qui peuvent susciter l'utilisation de machine de fabrication numérique (imprimante 3D, découpe laser, etc.).
  Un exemple d'application réalisée avec la scène de mBlock est proposé pour piloter le robot mBot2 avec une tablette en s'appuyant sur l'application mBlock (compatible Android ou iOS)

Symboles utilisés

Symbole	Type de mouvement	Action sur les effecteurs
	Marche avant	Les 2 moteurs tournent en marche avant
	Tourner à droite	Les 2 moteurs tournent dans des sens opposés et le mBot tourne sur lui-même
	Tourner à gauche	Les 2 moteurs tournent dans des sens opposés et le mBot tourne sur lui-même
	Marche arrière	Les 2 moteurs tournent en marche arrière
	Arrêt	Les 2 moteurs sont arrêtés

Matériels utilisés

Matériel	Description
mBot2 	Le mBot2 est un robot évolutif équipé de l'interface programmable « CyberPi » programmable avec mBlock 5 et compatible avec les modules capteurs / actionneurs de la gamme CyberPi. Il est équipé dans sa version de base d'un capteur suiveur de ligne et d'un télémètre à ultrasons. Il permet de réaliser des déplacements précis et dispose d'un affichage. Il fonctionne sur batterie rechargeable, ses dimensions sont 188 x 103 x 224mm. Il dispose de la connectivité Bluetooth et Wi-Fi.
CyberPi 	Le CyberPi est une mini interface programmable équipée d'une multitude de fonctionnalités : - Capteur de lumière, microphone, accéléromètre/gyroscope 3 axes, joystick 5 directions, 3 boutons, microphone, - Affichage couleur, barre de LEDs, haut-parleur - Connectivité Bluetooth et Wi-Fi - Connectique servomoteur, connectique modules mBuild - Batterie externe

Moteurs 	Bloc moteur avec pignons métalliques et codeurs pour assurer des déplacements précis du robot en appliquant des consignes de distance à parcourir, d'angle et de vitesse de déplacement.
Capteur à ultrasons 	Télémètre à ultrasons pour mesurer la distance qui le sépare d'un obstacle. La sonde de gauche transmet les ondes, la sonde de droite reçoit les ondes. Plage de mesure comprise entre 5 et 300 cm, précision ± 5%.
Capteur de ligne RGB 	4 LED RGB jumelées à 4 capteurs de couleurs. Autoétalonnage, suivi de ligne précis, détection de couleurs
Lanceur de balle 	Lanceur de balle avec angle de tir ajustable et balle
Servomoteur 	Servomoteur pilotable à rotation continu, couple 3,5 Kg.cm, dimensions 24 × 24 mm
Terrain DéfiBasket 	Terrain, de dimensions 1190 x 1400 mm, adapté à l'utilisation de détecteur de couleur. Les lignes noires qui délimitent le périmètre du terrain sont détectables par le capteur de ligne du mBot2. Les raquettes de couleurs jaune et verte sont détectables par le capteur de couleur.

Prise en main du robot mBot2

Ce chapitre propose une série d'exemples destinés à maîtriser chaque fonctionnalité utile à la réalisation du défi. Les programmes proposés sont classés par ordre de difficulté croissante.
Chaque fiche propose une problématique de programmation, une configuration possible du robot et un exemple de correction. Les fichiers de correction (extension «.mBlock») sont téléchargeables sur www.a4.fr dans la rubrique ressources DéfiBasket.

Liste des programmes

Nom du fichier	Description	Objectif
MB2-BAB-F1	Avancer puis s'arrêter	Maîtriser le déplacement du robot
MB2-BAB-F2	Avancer puis tourner	Maîtriser le déplacement du robot
MB2-BAB-F3	Avancer en décrivant un carré	Maîtriser le déplacement du robot
MB2-BAB-F4	Afficher un message	Maîtriser l'affichage d'informations
MB2-BAB-F5	Afficher la valeur retournée le capteur de distance	Maîtriser le fonctionnement du capteur de distance
MB2-BAB-F6	S'arrêter avant un obstacle	Exploiter la valeur renvoyée par le capteur de distance
MB2-BAB-F7	Éviter un obstacle	Exploiter la valeur renvoyée par le capteur de distance
MB2-BAB-F8	Afficher la valeur retournée par le module capteur de ligne	Maîtriser le fonctionnement du capteur de **ligne** / couleurs
MB2-BAB-F9	S'arrêter lorsqu'une ligne noire est détectée	Maîtriser le fonctionnement du capteur de **ligne** / couleurs
MB2-BAB-F10	Tourner lorsqu'une ligne noire est détectée puis avancer	Maîtriser le fonctionnement du capteur de **ligne** / couleurs
MB2-BAB-F11		Maîtriser le fonctionnement du capteur de **ligne** / couleurs
MB2-BAB-F12		Maîtriser le fonctionnement du capteur de ligne / **couleurs**
MB2-BAB-F13		Maîtrise le fonctionnement d'un Servomoteur

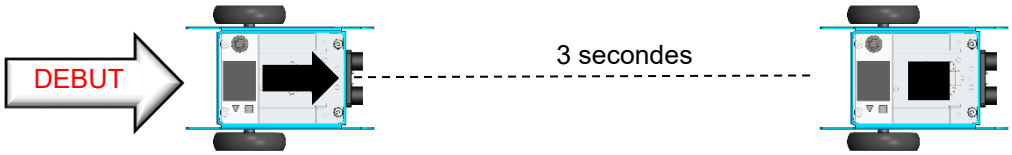
FICHE N°1 : avancer puis s'arrêter

But du programme : se déplacer en avant à 50 tours par minute pendant 3 secondes puis s'arrêter.

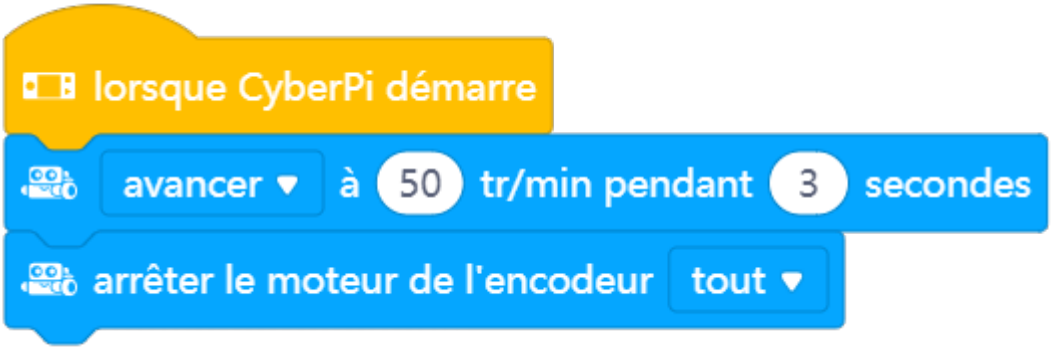
Notion de programmation abordée : séquence d'instructions

Capteur(s) / actionneur(s) utilisé(s) : moteurs

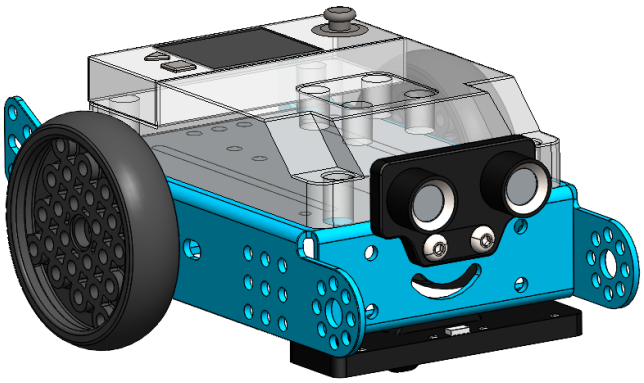
Synoptique :



Exemple de correction : **MB2-BAB-F1.mBlock**



Exemple de configuration du robot :



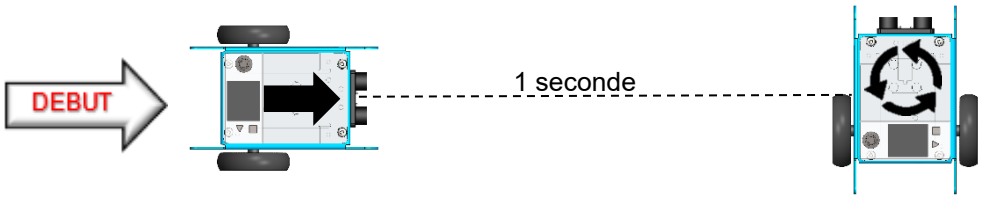
FICHE N°2 : avancer puis tourner

But du programme : se déplacer en avant à 50 tours par minute pendant 1 seconde puis pivoter de 90°.

Notion de programmation abordée : séquence d'instructions

Actionneurs utilisés : moteurs

Synoptique :



Exemple de correction : **MB2-BAB-F2.mBlock**

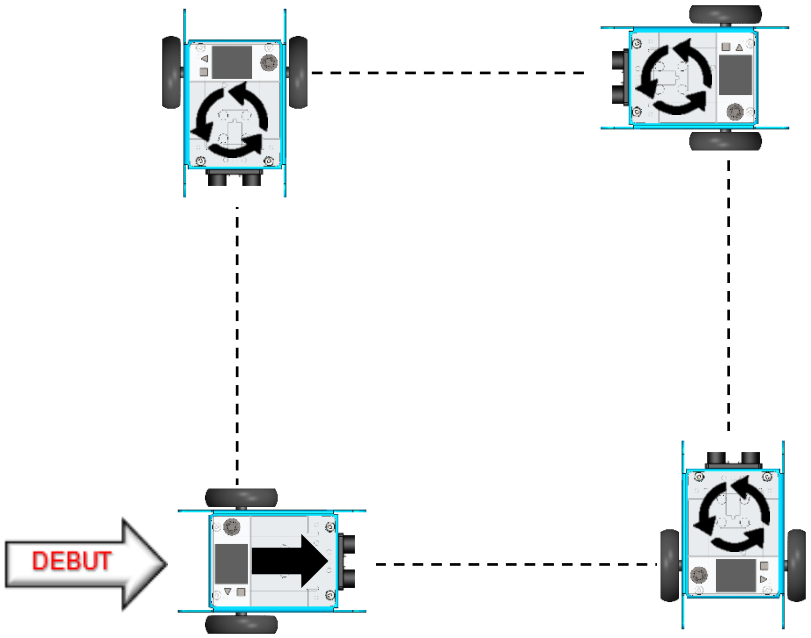
FICHE N°3 : avancer en décrivant un carré

But du programme : avancer en permanence en décrivant un carré.

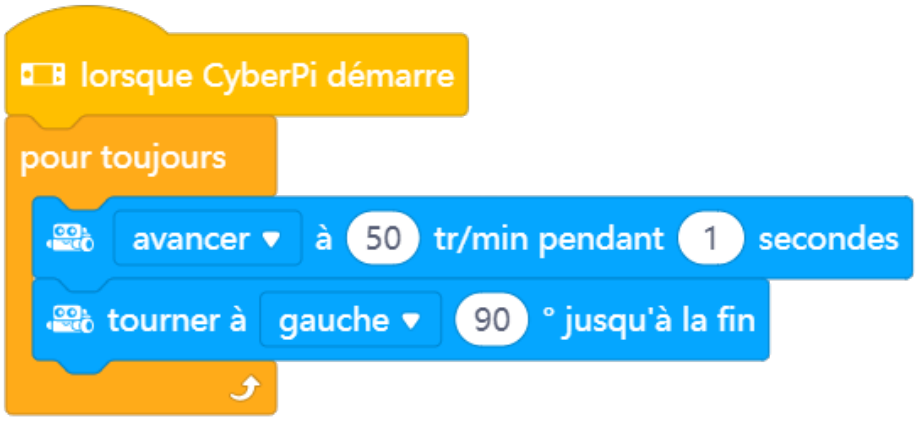
Notion de programmation abordée : séquence d'instructions, boucle

Actionneurs utilisés : moteurs, encodeurs

Synoptique :



Exemple de correction **MB2-BAB-F3.mBlock**



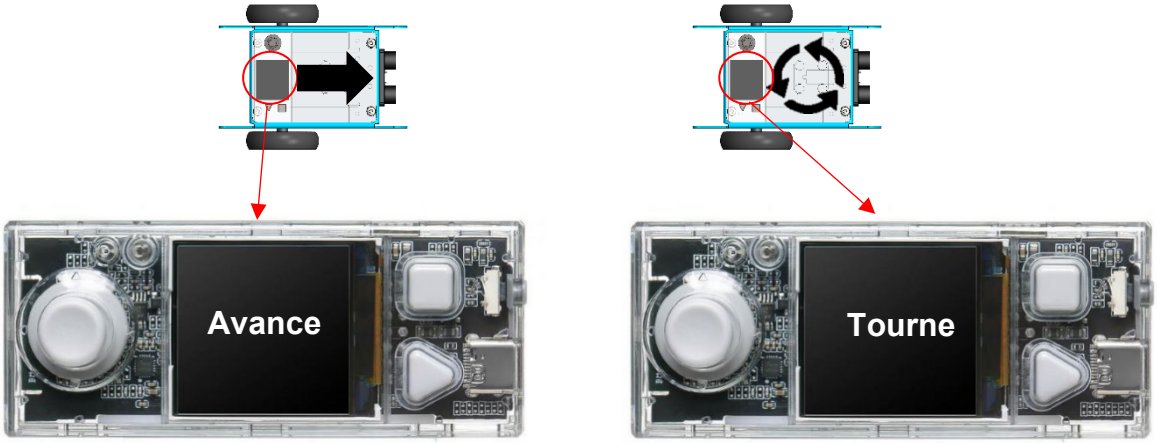
FICHE N°4 : afficher un message

But du programme : avancer en décrivant un carré de 10 cm de côté puis s'arrêter. Afficher le niveau de batterie, afficher « Avance » lorsque le robot se déplace « Tourne » lorsqu'il pivote sur lui-même.

Notion de programmation abordée : séquence d'instructions, boucle, affichage d'informations

Actionneurs utilisés : moteurs, affichage

Synoptique :



Exemple de correction : **MB2-BAB-F4.mBlock**

FICHE N°5 : afficher la valeur retournée le capteur de distance

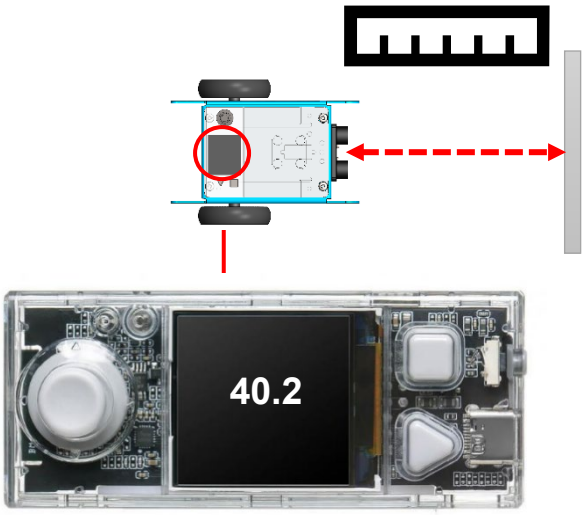
But du programme : afficher la valeur retournée par le télémètre à ultrasons sur l'écran du CyberPi.

Observer le fonctionnement du capteur :

- Déplacer un objet devant le capteur et observer les valeurs retournées
- Relever les valeurs minimales et maximales que le capteur peut détecter

Capteurs / actionneurs utilisés : télémètre à ultrasons, affichage

Synoptique :



Exemple de correction : MB2-BAB-F5.mBlock

Observation : on remarque que le capteur retourne la valeur 300 si l'objet est « très » proche à quelques centimètres de l'obstacle, ou éloigné à plus de 3 mètres.

Pour aller plus loin :

- Déterminer le rayon d'action du capteur (angle de détection)
- Comparer la valeur retournée par le capteur (valeur brute) à celle mesurée avec une règle (système métrique)
- Déterminer la relation entre la valeur brute retournée par le capteur et une valeur réelle de la distance exprimée en centimètres
- Afficher la valeur de la distance en cm

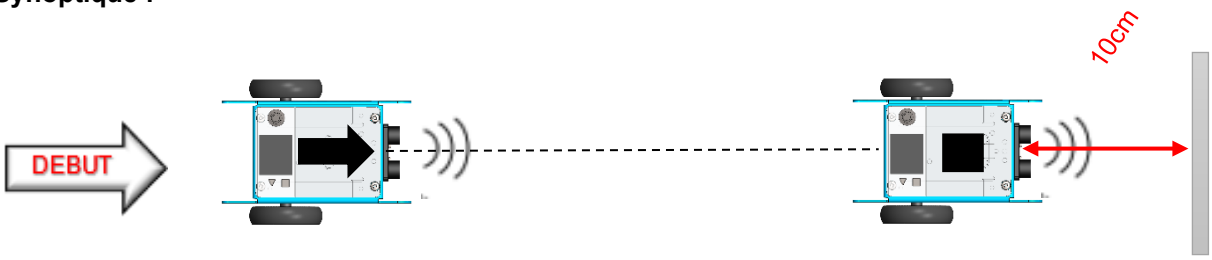
FICHE N°6 : s'arrêter avant un obstacle

But du programme : avancer à 30 tours par minute, s'arrêter lorsqu'un objet est détecté à moins de 10 cm

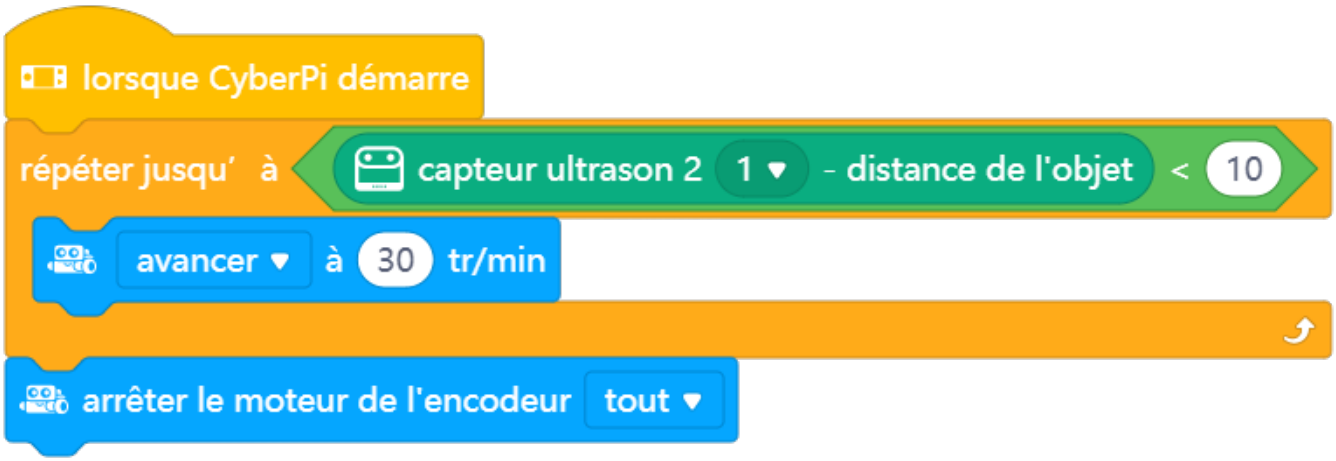
Notion de programmation abordée : séquence d'instructions, boucle, test conditionnel

Capteur utilisé : télémètre à ultrasons

Synoptique :



Exemple de correction : **MB2-BAB-F6.mBlock**



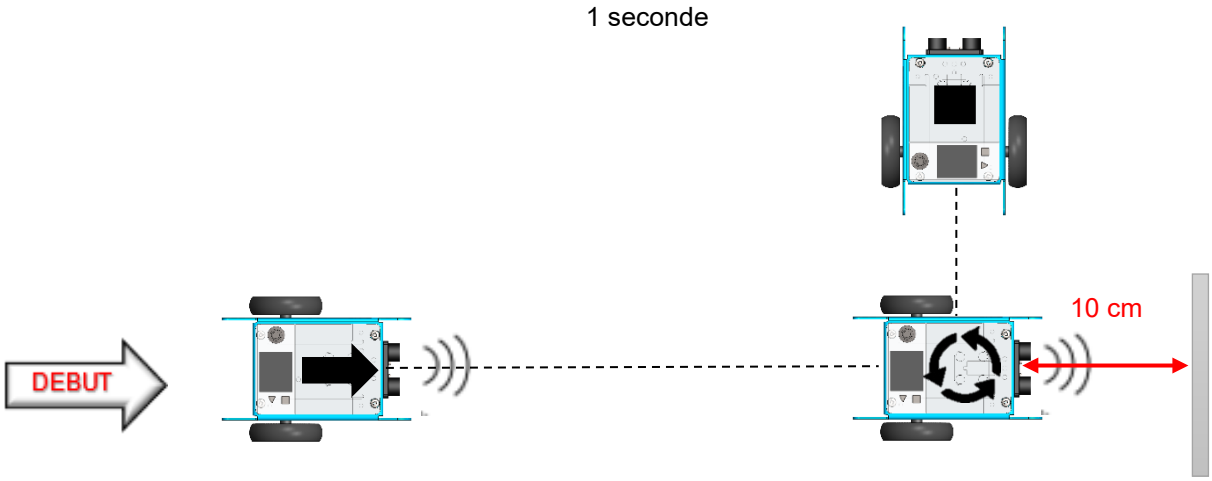
FICHE N°7 : éviter un obstacle

But du programme : avancer à 30 tours par minute, si un obstacle est détecté à moins de 10 cm pivoter à gauche de 90°, reprendre le déplacement pendant 1 seconde.

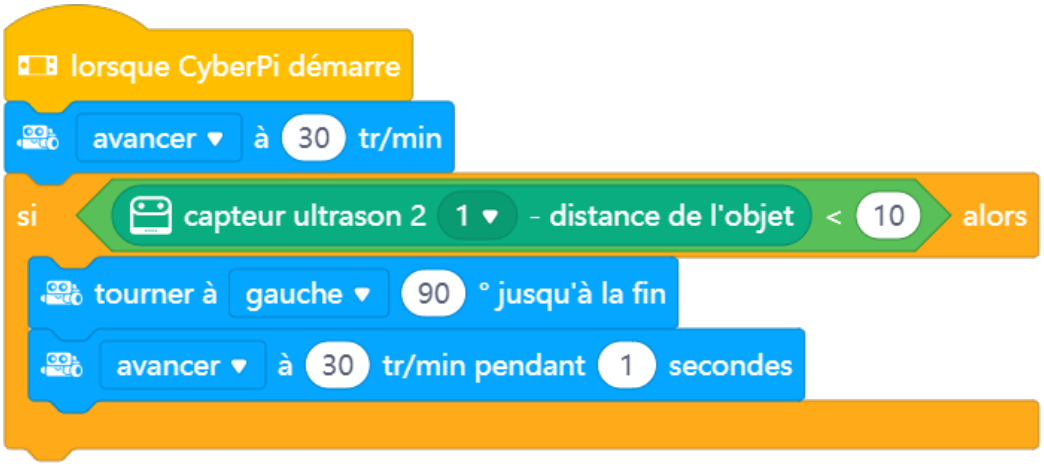
Notion de programmation abordée : séquence d'instructions, test conditionnel

Capteur utilisé : télémètre à ultrasons

Synoptique :



Exemple de correction : **MB2-BAB-F7.mBlock**



FICHE N°8 : afficher la valeur retournée par le module capteur de ligne

But du programme : afficher la valeur retournée par le capteur de ligne sur l'écran du CyberPi en utilisant le bloc de détection de ligne suivant configuré avec le statut « ligne » :

NOTE IMPORTANTE : le module capteur de ligne nécessite un **étalonnage préalable** à son utilisation. Un bouton d'auto étalonnage permet d'ajuster sa sensibilité pour différencier les différentes couleurs à détecter. Voir la procédure d'étalonnage en ANNEXE de ce dossier.

Questionnement :
Positionner le capteur au-dessus du terrain (brun clair) puis le déplacer au-dessus de la ligne noire qui délimite le terrain et les zones de couleurs autour des paniers.

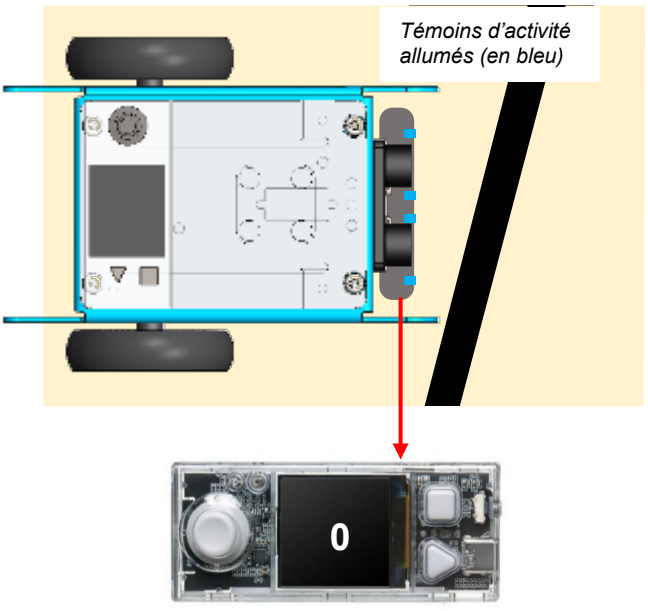
- Observer l'activité des témoins lumineux bleus
- Observer les valeurs retournées par le module
- Relever les valeurs minimales et maximales retournées par le module
- Établir la relation entre la valeur retournée par le module et l'état de chacun des 4 capteurs RVB

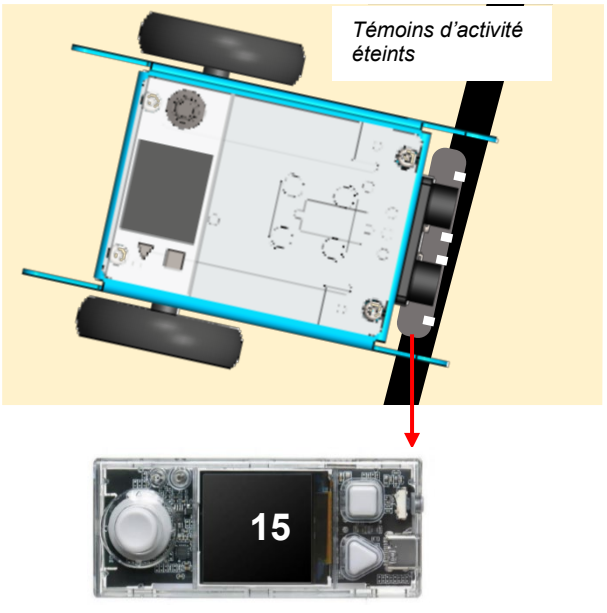
Capteur utilisé : module capteur de ligne

Synoptique :

Dans l'exemple qui suit, le module capteur de ligne est étalonné sur la couleur de fond du terrain (couleur claire)

- Le témoin d'activité de chacun des 4 capteurs s'allume en bleu s'il est au-dessus de la couleur de fond du terrain. La valeur renvoyée par le module est « 0 » si tous les capteurs sont au-dessus de la couleur de fond du terrain.
- Le témoin d'activité de chacun des s'allume en bleu s'éteint s'il est au-dessus de la cligne noire. La valeur renvoyée par le module est « 15 » si tous les capteurs sont au-dessus de la ligne noire

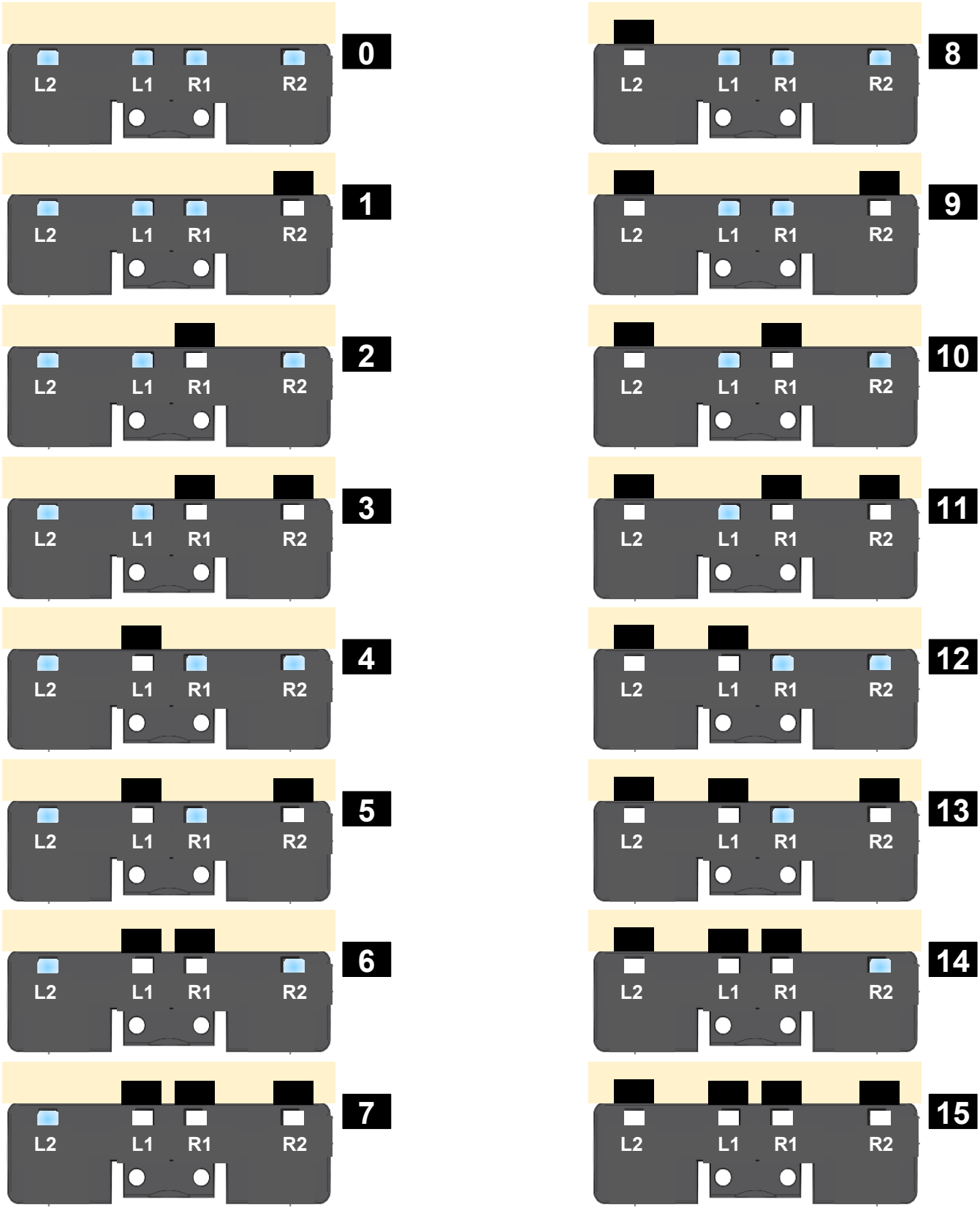




NOTE : on peut inverser la logique de détection du module capteur de ligne en l'étalonnant sur la ligne noire (couleur de fond sombre). Dans ce cas, les valeurs renvoyées par le module capteur de ligne seraient opposées.

- Le témoin d'activité de chacun des 4 capteurs s'éteint s'il est au-dessus de la couleur de fond du terrain. La valeur renvoyée par le module est « 15 » si tous les capteurs sont au-dessus de la couleur de fond du terrain.
- Le témoin d'activité de chacun des 4 capteurs s'allume en bleu s'il est au-dessus de la ligne noire. La valeur renvoyée par le module est « 0 » si tous les capteurs sont au-dessus de la ligne noire

Exemple de correction : MB2-BAB-F8.mBlock



Chaque capteur du module de détection de ligne renvoie une valeur binaire «au-dessus du terrain » ou « au-dessus de la ligne noire ». Le module est équipé de 4 capteurs qui permettent de constituer un mot sur 4 bits.

- L2 (Left 2)
- L1 (Left 1)
- R1 (Right 1)
- R2 (Right 2)

La table de vérité ci-dessous représente tous les états possibles du module de détection de ligne (2^4=16 états) et la valeur décimale correspondante retournée par le module. Les valeurs 2^3, 2^2, 2^1, 2^1 correspondent au poids respectif de chaque bit L2, L1, R1, R2.

Table de vérité :

État capteur L2	État capteur L1	État capteur R1	État capteur R2	Valeur décimale retournée	Remarque
Poids : 2^3= 8	**Poids : 2^2= 4**	**Poids : 2^1= 2**	**Poids : 2^0= 1**	**Somme**	
0	0	0	0	**0**	Les 4 capteurs sont au-dessus du fond clair
0	0	0	2^0	**1**	Au moins un des 4 capteurs est au-dessus de la ligne noire
0	0	2^1	0	**2**	
0	0	2^1	2^0	**3**	
0	2^2	0	0	**4**	
0	2^2	0	2^0	**5**	
0	2^2	2^1	0	**6**	
0	2^2	2^1	2^0	**7**	
2^3	0	0	0	**8**	
2^3	0	0	2^0	**9**	
2^3	0	2^1	0	**10**	
2^3	0	2^1	2^0	**11**	
2^3	2^2	0	0	**12**	
2^3	2^2	0	2^0	**13**	
2^3	2^2	2^1	0	**14**	
2^3	2^2	2^1	2^0	**15**	

Légendes :

Témoins d'activité allumés (en bleu)

Témoins d'activité éteints

Exemple de conversion binaire en décimale :

État capteur L2	État capteur L1	État capteur R1	État capteur R2	Valeur décimale retournée
Poids : 2^3= 8	**Poids : 2^2= 4**	**Poids : 2^1= 2**	**Poids : 2^0= 1**	**Somme**
2^3	0	2^1	0	**10**

- L2 est au-dessus de la ligne noire, le témoin est éteint, son poids vaut 1 x 2^3 = 8
- L1 est au-dessus du terrain, le témoin est allumé, son poids vaut 0 x 2^2 = 0
- R1 est au-dessus de la ligne noire, le témoin est éteint, son poids vaut 1 x 2^1 = 2
- R2 est au-dessus du terrain, le témoin est éteint, son poids vaut 0 x 2^0 = 0

La valeur décimale renvoyée par le module est : 8 +0 + 2 + 0 = 10

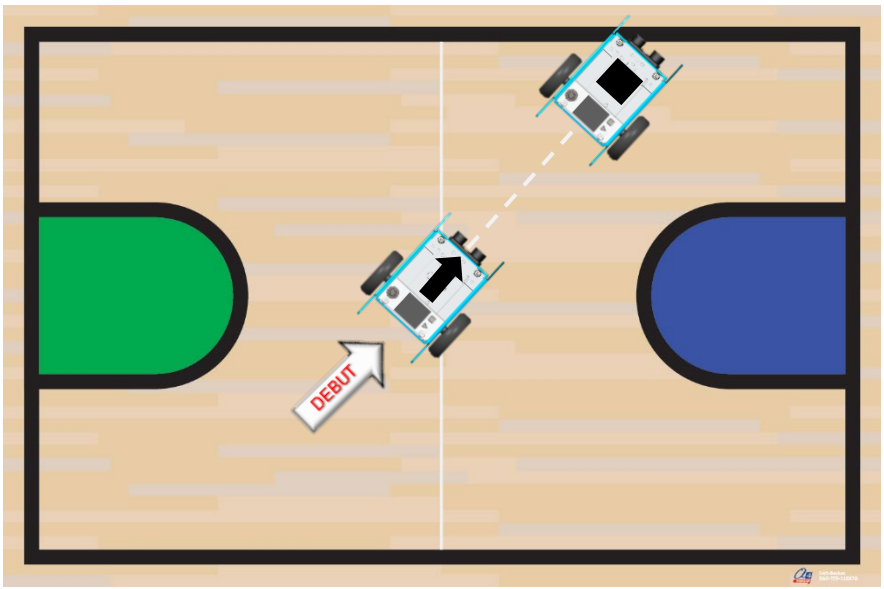
FICHE N°9 : s'arrêter lorsqu'une ligne noire est détectée

But du programme : avancer à 50 tours par minute, s'arrêter lorsqu'une ligne noire est détectée

Notion de programmation abordée : séquence d'instructions, boucle, test conditionnel

Capteur utilisé : module capteur de ligne

Synoptique :



Exemple de correction : **MB2-BAB-F9.mBlock**

Remarque :

Cette instruction signifie que la valeur renvoyée par le module de détection de ligne est différente de 0. Cela signifie qu'au moins 1 des 4 capteurs L2, L1, R2 ou R1 est au-dessus de la ligne noire.

NOTE IMPORTANTE : le module capteur de ligne nécessite un étalonnage préalable à son utilisation. Un bouton d'auto étalonnage permet d'ajuster sa sensibilité pour différencier les différentes couleurs à détecter. Voir la procédure d'étalonnage en ANNEXE de ce dossier.

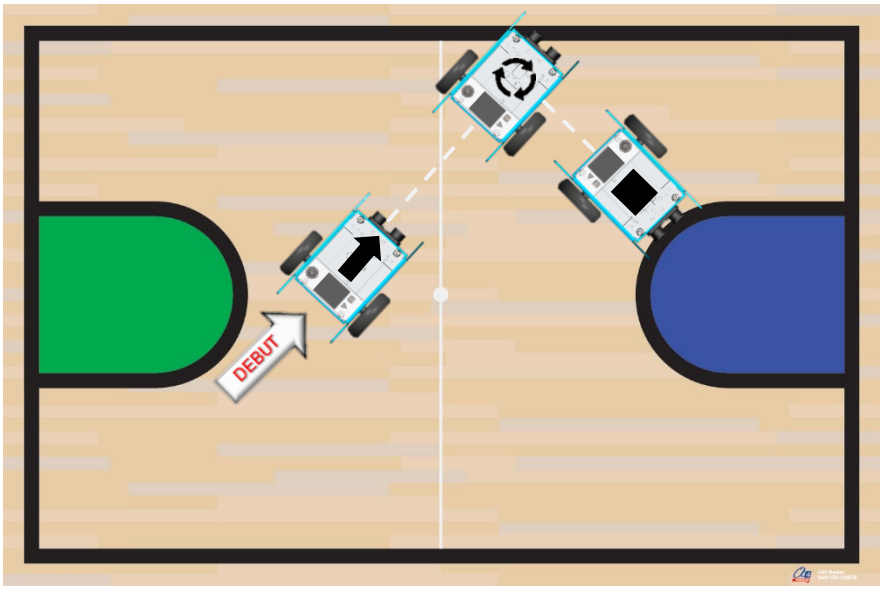
FICHE N°10 : tourner lorsqu'une ligne noire est détectée puis avancer

But du programme : avancer à 40 tr/min, tourner à droite de 90° lorsque la ligne noire est détectée, puis avancer à 20 tr/min pendant 1 seconde.

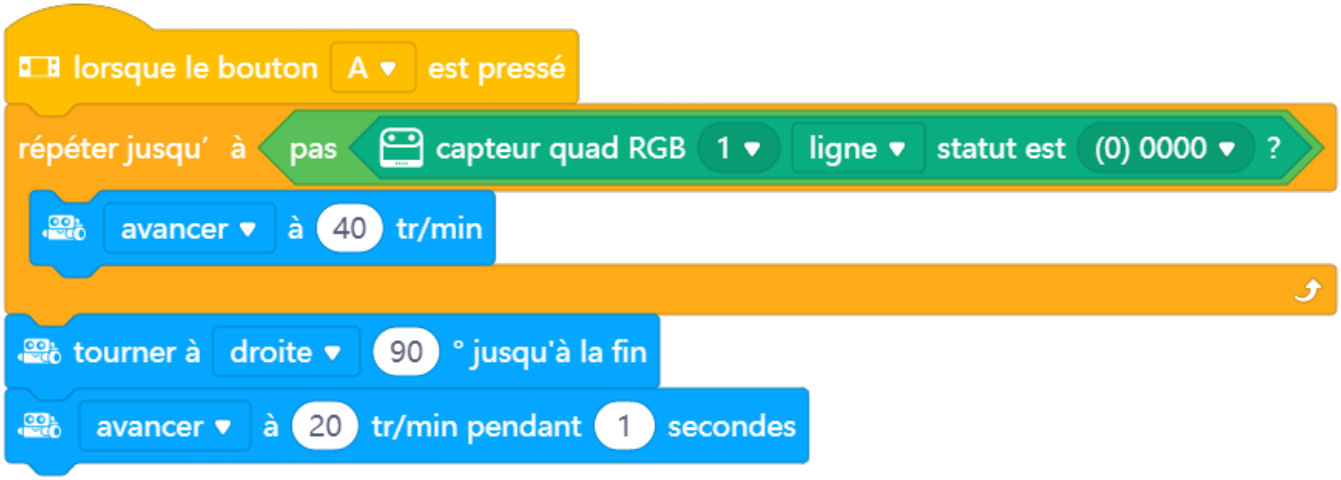
Notion de programmation abordée : séquence d'instructions, boucle, test conditionnel

Capteur utilisé : module capteur de ligne

Synoptique :



Exemple de correction : **MB2-BAB-F10.mBlock**



NOTE IMPORTANTE : le module capteur de ligne nécessite un étalonnage préalable à son utilisation. Un bouton d'auto étalonnage permet d'ajuster sa sensibilité pour différencier les différentes couleurs à détecter. Voir la procédure d'étalonnage en ANNEXE de ce dossier.

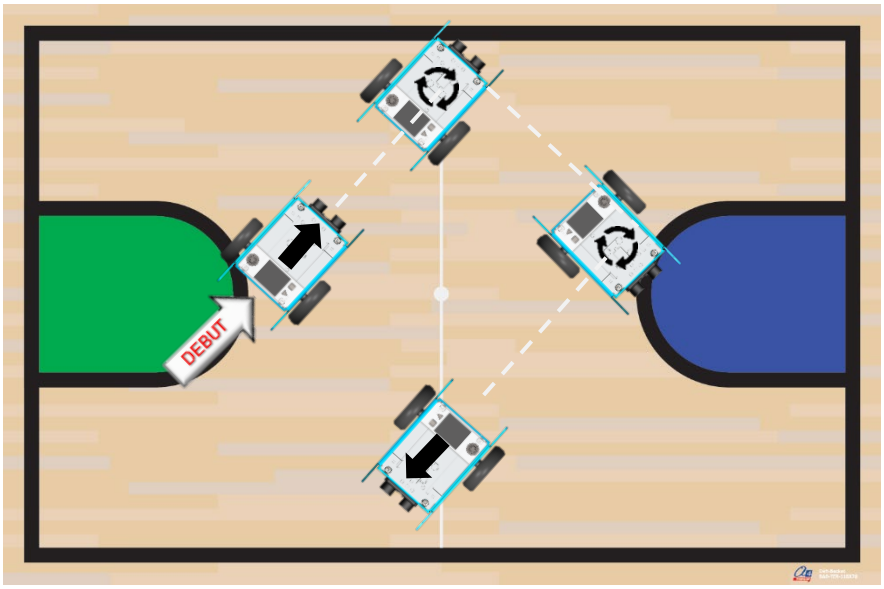
FICHE N°11 : rebondir sur les lignes noires

But du programme : tourner à droite lorsqu'une ligne noire est détectée, sinon se déplacer en marche avant à 30 tr/min.

Notion de programmation abordée : séquence d'instructions, boucles imbriquées, test conditionnel

Capteur utilisé : module capteur de ligne

Synoptique :



Exemple de correction : **MB2-BAB-F11.mBlock**

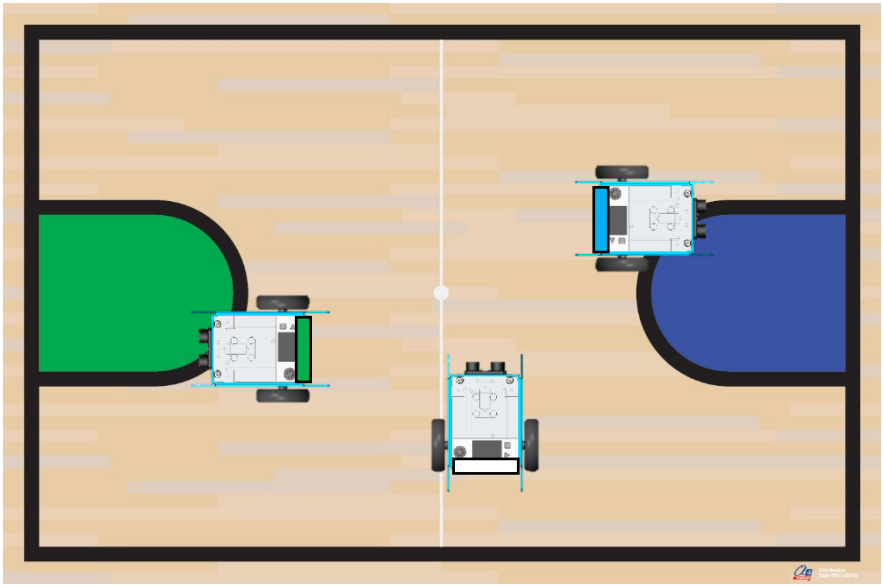
NOTE IMPORTANTE : le module capteur de ligne nécessite un étalonnage préalable à son utilisation. Un bouton d'auto étalonnage permet d'ajuster sa sensibilité pour différencier les différentes couleurs à détecter. Voir la procédure d'étalonnage en ANNEXE de ce dossier.

FICHE N°12 : afficher la couleur détectée par le capteur de lignes quad RGB

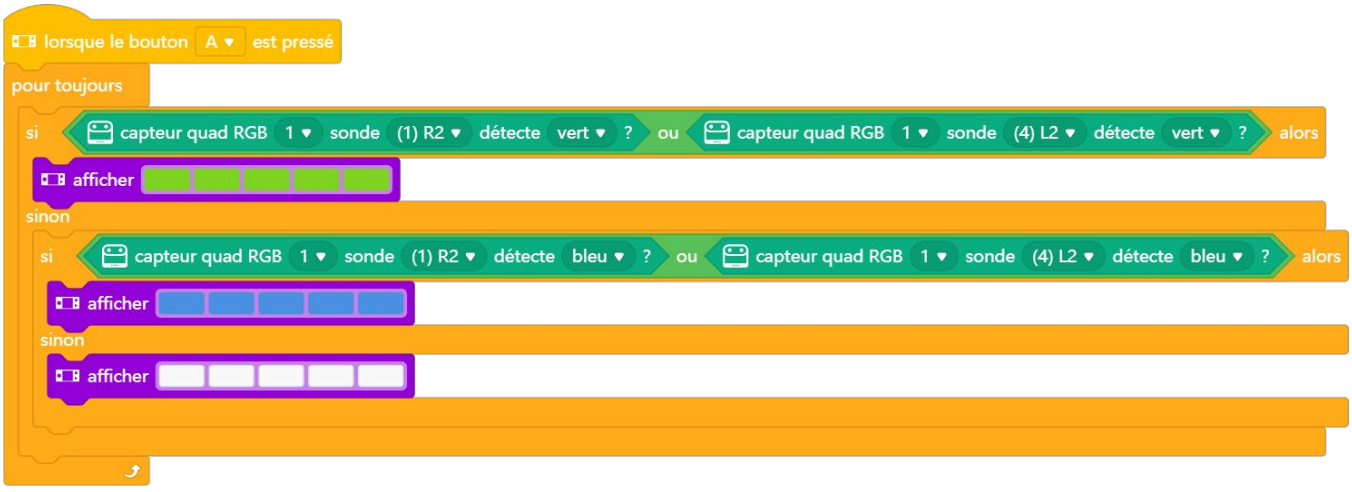
But du programme : afficher sur la barre de LEDs du CyberPi les couleurs suivantes

- Verte si les capteurs de couleur L2 ou R2 du module de détection de ligne sont au-dessus de la zone verte
- Bleue si les capteurs de couleur L2 ou R2 du module de détection de ligne sont au-dessus de la zone bleue
- Blanche si les capteurs de couleur L2 ou R2 du module de détection de ligne ne sont ni au-dessus de la zone verte, ni au-dessus de la zone bleue.

Synoptique :



Exemple de correction : MB2-BAB-F12.mBlock



FICHE N°13 : déclencher le lance balle

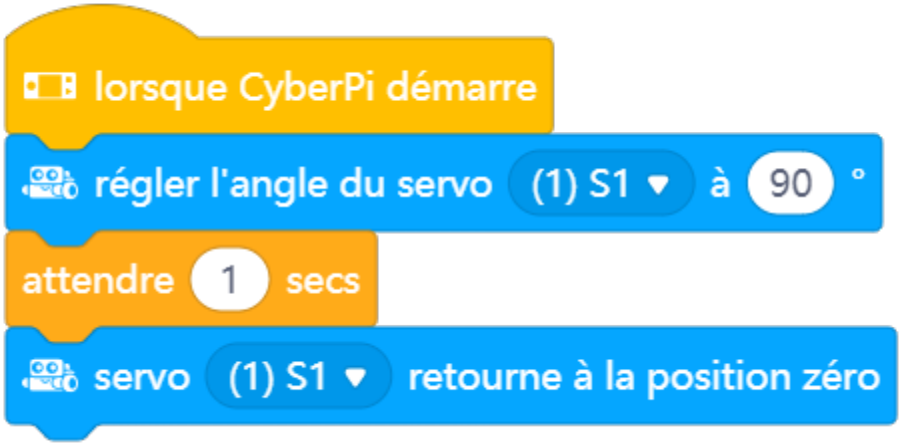
But du programme : déclencher le lance balle à l'appui sur le bouton A du CyberPi

Notion de programmation abordée : prise en main du servomoteur

Actionneur utilisé : servomoteur

Synoptique :

Exemple de correction : **MB2-BAB-F13.mBlock**



NOTE: la fiche de montage du module lance balle est disponible en ANNEXE de ce dossier.

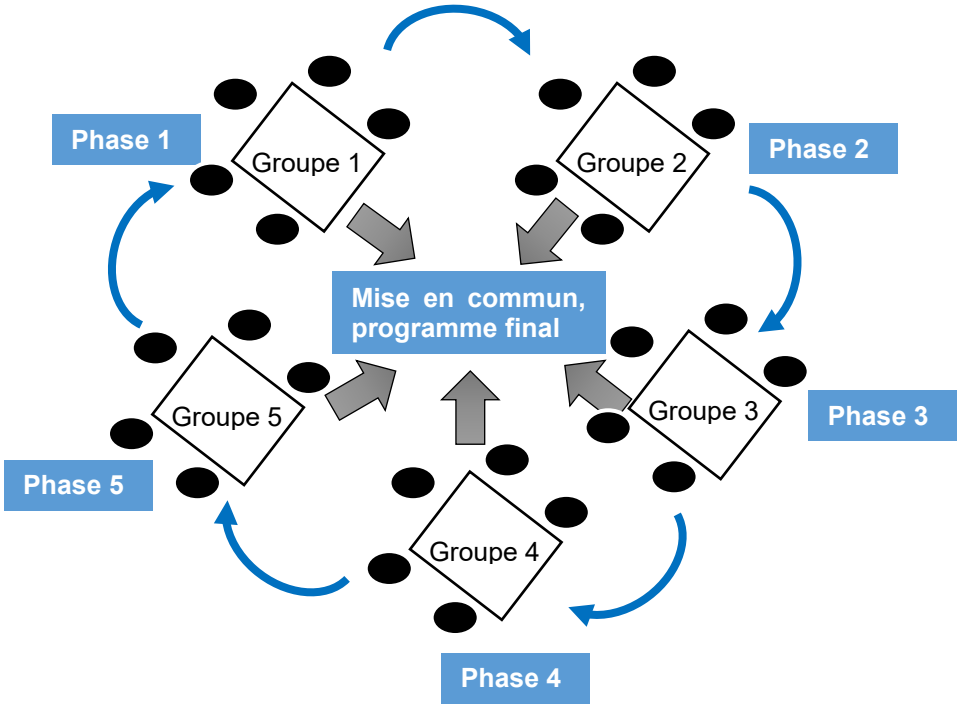
DéfiBasket - Marquer un panier de manière autonome

L'algorithme utilisé pour programmer un robot qui marque un panier de manière autonome peut se décomposer en tâches simples. Les élèves peuvent s'organiser en groupes autour de ce projet commun en répartissant les différentes tâches (programmation, design, conception mécanique, test et validation, suivi du projet, etc.).
Le jeu d'instructions simplifiées disponibles dans mBlock5 et la simplicité de mise en œuvre des modules capteurs / actionneurs du mBot2 facilitent la résolution de cette problématique.
Chaque groupe d'élèves peut prendre en charge la réalisation d'un programme qui est alimenté par des données d'entrée et qui produit des données de sortie nécessaires au groupe suivant. La mise en commun des programmes réalisés aboutit au programme final qui réalise la tâche souhaitée.

Le tableau ci-dessous propose une décomposition du problème.

Phase	Situation de départ	Action	Situation d'arrivée	Exemple de correction
1	Le robot est mis sous tension dans la zone de jeu	Rester sur le terrain en se déplaçant aléatoirement	Le robot se déplace dans le terrain	**MB2-BAB-F14**
2	Le robot se déplace dans le terrain	Circuler sur le terrain jusqu'à détecter la zone verte puis s'arrêter	Le robot est à l'arrêt à la limite de la zone verte	**MB2-BAB-F15**
3	Le robot est à l'arrêt à la limite de la zone verte	S'orienter vers le panier	Le robot est orienté en direction du panier	**MB2-BAB-F16**
4	Le robot est orienté en direction du panier	S'éloigner du panier pour être à la bonne distance de tir	Le robot est à la bonne distance de tir orienté en direction du panier	**MB2-BAB-F17**
5	Le robot est à la bonne distance de tir orienté en direction du panier	Déclencher le tir	Le robot a déclenché le tir	**MB2-BAB-F18**

5 groupes d'élèves peuvent se répartir les tâches pour réaliser ce projet commun :



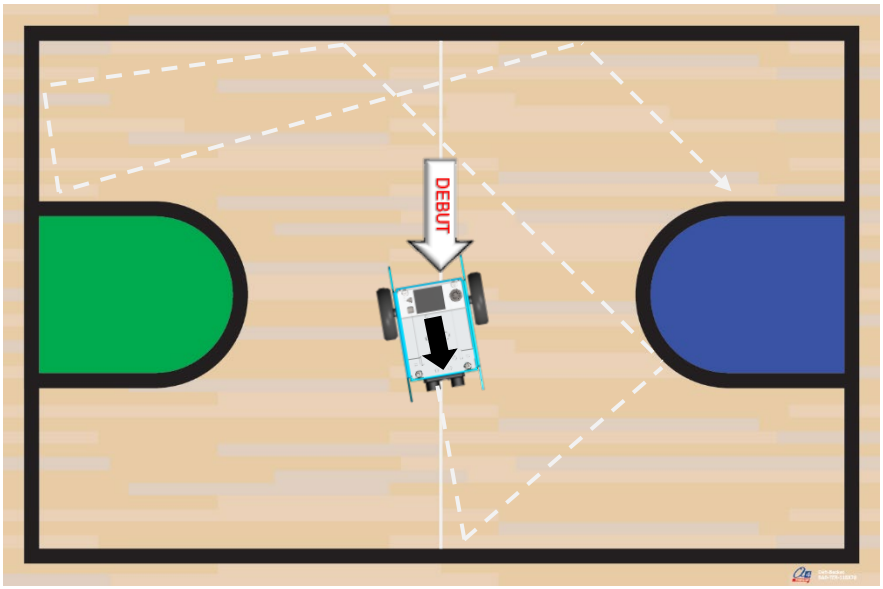
PHASE 1 : rester sur le terrain en se déplaçant aléatoirement

Situation de départ : le robot est mis sous tension dans la zone de jeu

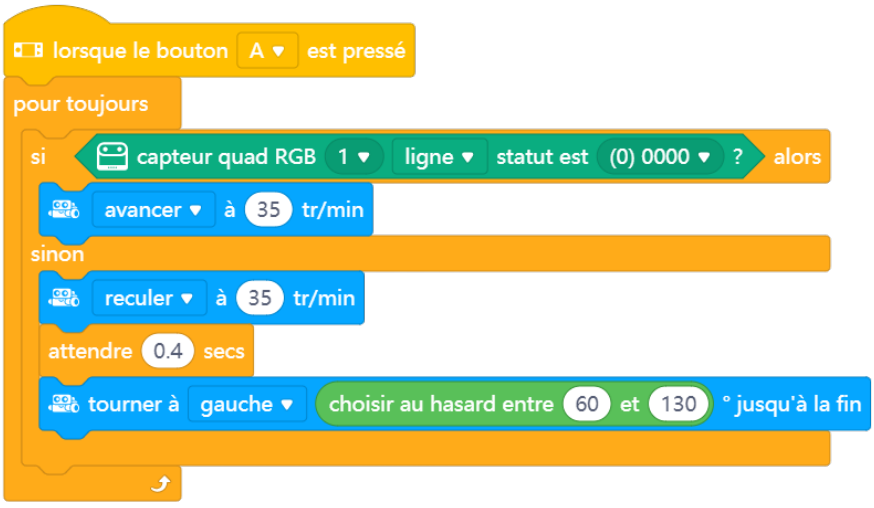
Action : rester sur le terrain en se déplaçant aléatoirement

Situation d'arrivée : le robot se déplace dans le terrain

Synoptique :



Exemple de correction : **MB2-BAB-F14.mblock**



Commentaire : le module de détection de ligne est préalablement étalonné à partir de la couleur de fond du terrain.

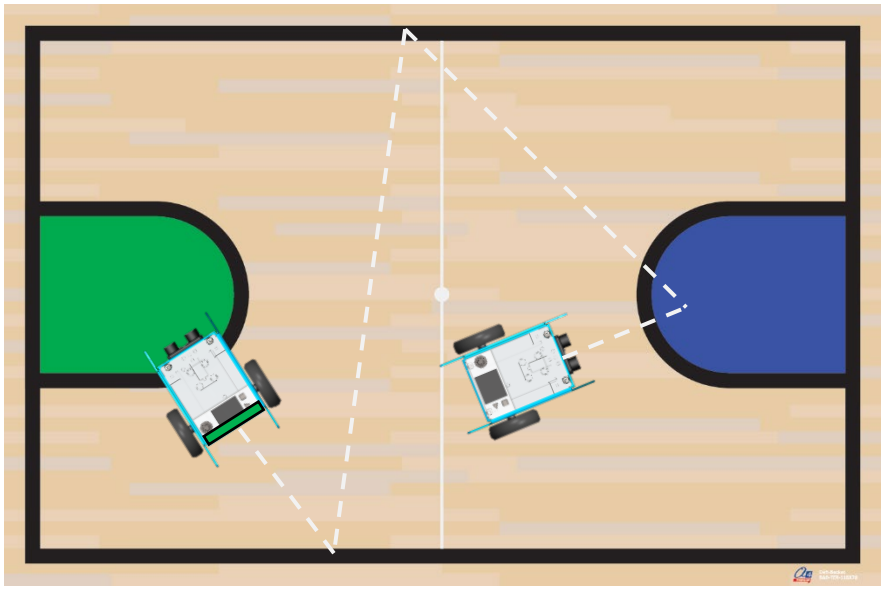
PHASE 2 : circuler sur le terrain jusqu'à détecter la zone verte puis s'arrêter

Situation de départ : le robot se déplace dans le terrain

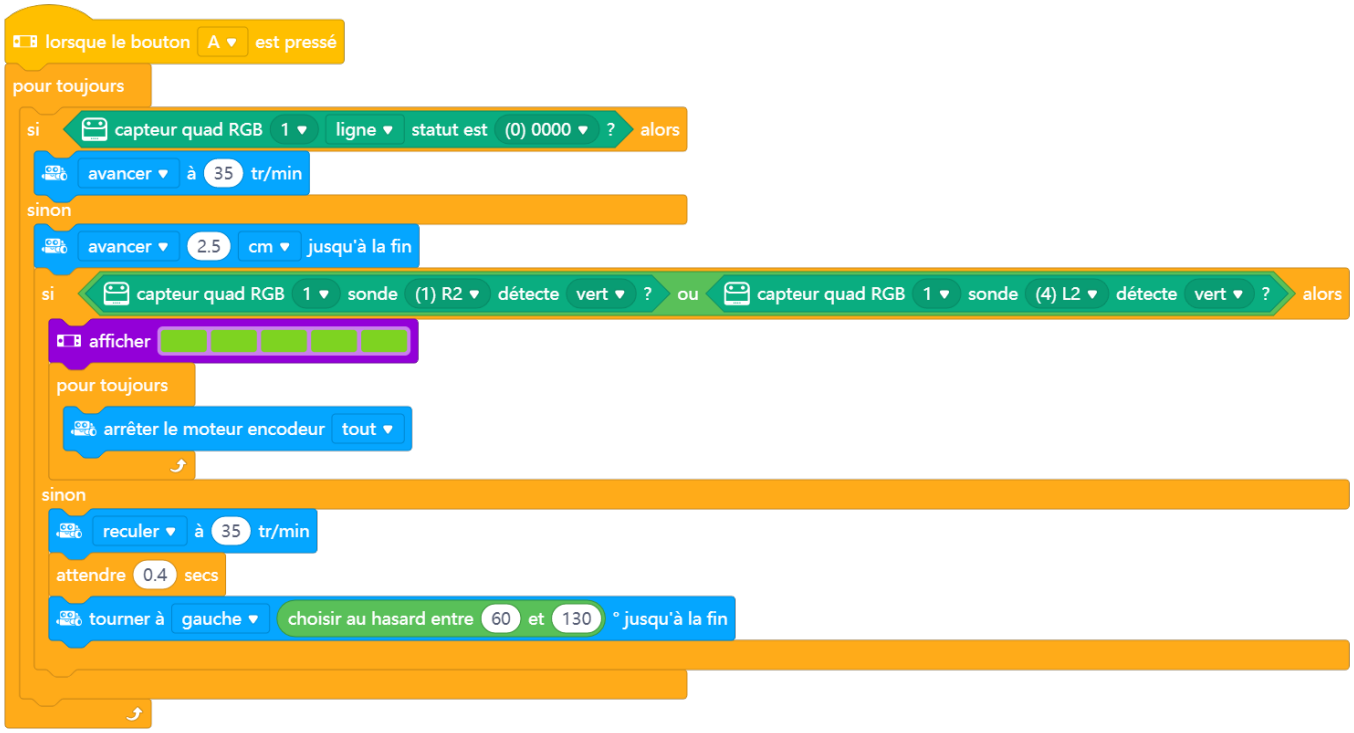
Action : circuler sur le terrain jusqu'à détecter la zone verte puis s'arrêter

Situation d'arrivée : le robot est à l'arrêt à la limite de la zone verte

Synoptique :



Exemple de correction : **MB2-BAB-F15.mblock**



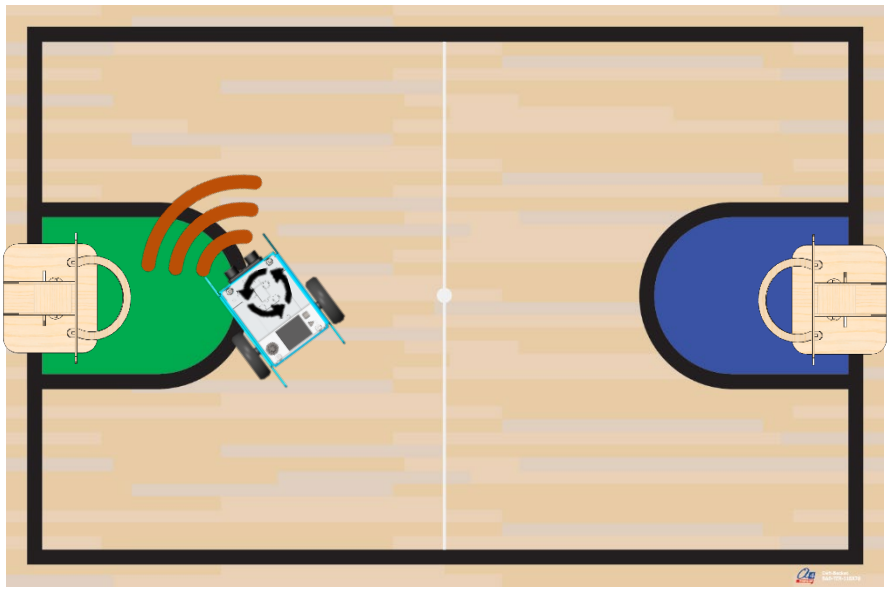
PHASE 3 : s'orienter vers le panier

Situation de départ : le robot est à l'arrêt à la limite de la zone verte

Action : s'orienter vers le panier

Situation d'arrivée : le robot est orienté en direction du panier

Synoptique :



Exemple de correction : MB2-BAB-F16.mblock

Note : l'appui sur le bouton A permet de s'assurer que le robot détecte le panier à une distance comprise entre 15 et 26 cm. L'appui sur le bouton B le fait reculer puis affiche la nouvelle distance renvoyée par le capteur à ultrasons.

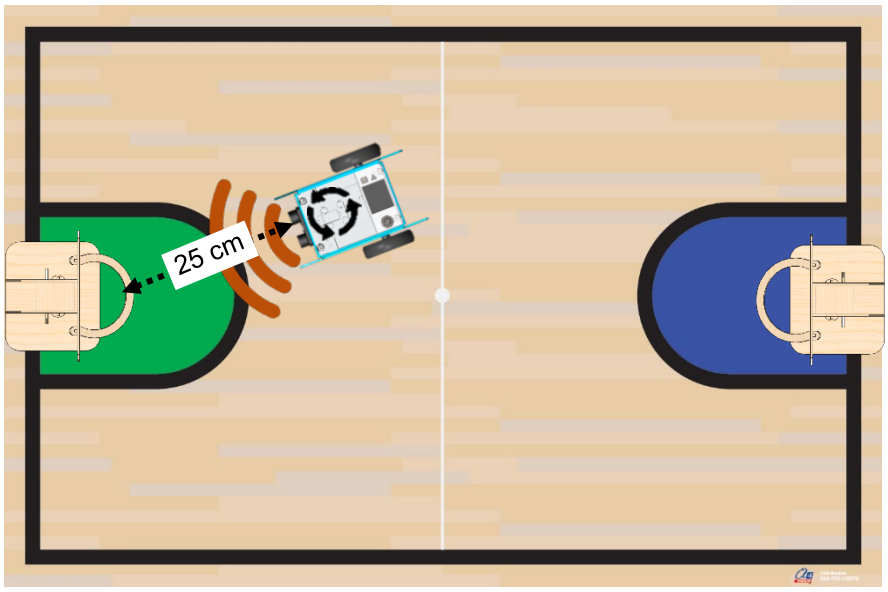
PHASE 4 : s'éloigner du panier pour être à la bonne distance de tir

Situation de départ : le robot est orienté en direction du panier

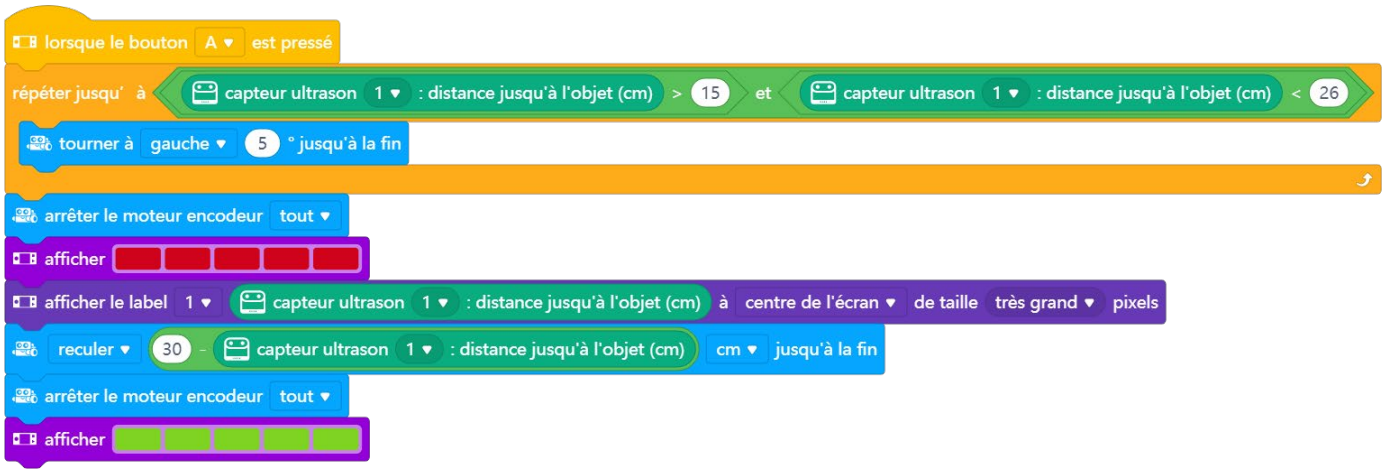
Action : s'éloigner du panier pour être à la bonne distance de tir

Situation d'arrivée : le robot est à la bonne distance de tir orienté en direction du panier

Synoptique :



Exemple de correction : **MB2-BAB-F17.mblock**



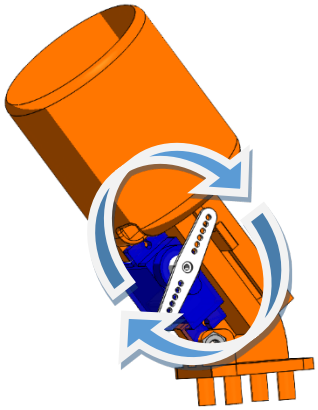
PHASE 5 : déclencher le tir

Situation de départ : Ie robot est à la bonne distance de tir orienté en direction du panier

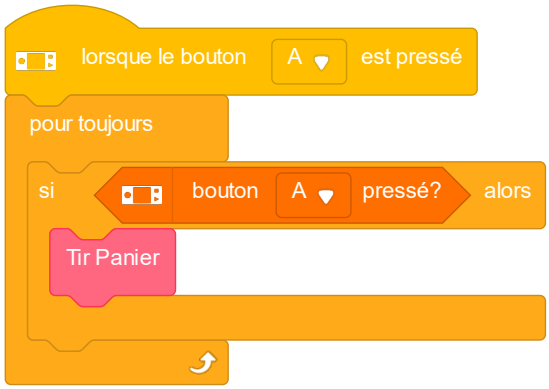
Action : déclencher le tir

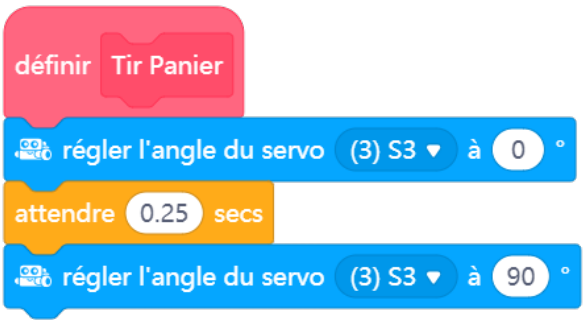
Situation d'arrivée : le robot a déclenché le tir

Synoptique :



Exemple de correction : **MB2-BAB-F18.mblock**





Commentaire : Le servomoteur vient comprimer le ressort du lanceur et le relâche pour éjecter la balle.

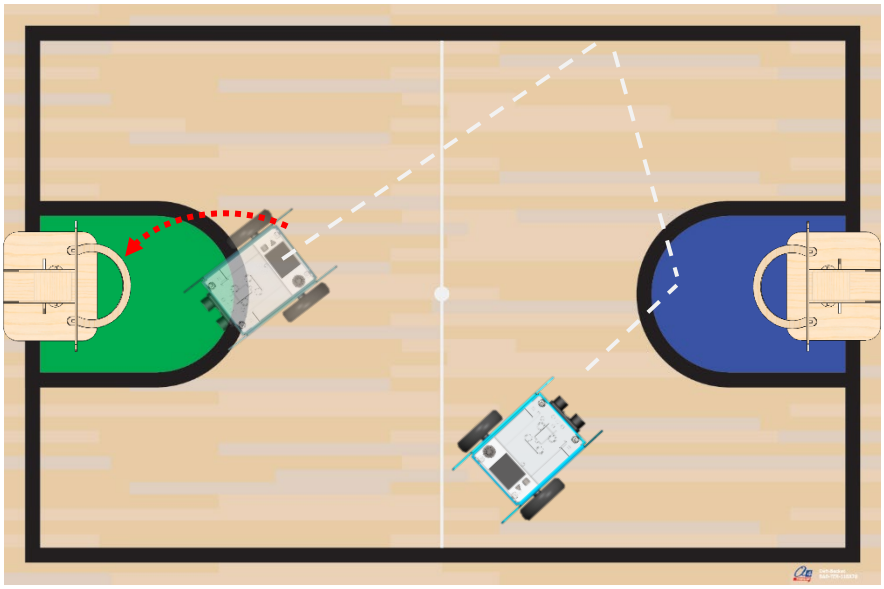
MISE EN COMMUN : mise au point du programme final

Situation de départ : le robot est mis sous tension dans la zone de jeu

Action : le robot évolue sur le terrain de manière autonome afin de déclencher un tir vers le panier de la zone verte

Situation d'arrivée : le panier est marqué !

Synoptique :



Phase	Situation de départ	Programme source
1	Mise sous tension du robot dans la zone de jeu, déplacement aléatoire dans le terrain	**MB2-BAB-F14**
2	Détection de la zone verte, arrêt du déplacement	**MB2-BAB-F15**
3	Localisation du panier, direction du robot dans sa direction	**MB2-BAB-F16**
4	Mise à distance de tir	**MB2-BAB-F17**
5	Déclenchement du tir	**MB2-BAB-F18**

Mise en commune des programmes :

- Les programmes permettant de réaliser les phases 1 à 5 sont testés individuellement au préalable.
- Ils sont assemblés deux à deux puis de nouveau testés.
- Des informations aidant au débogage sont introduites au fur et à mesure afin de s'assurer que le déroulement du programme est conforme à ce qui est attendu.

En cas de dysfonctionnement, procéder avec méthode :

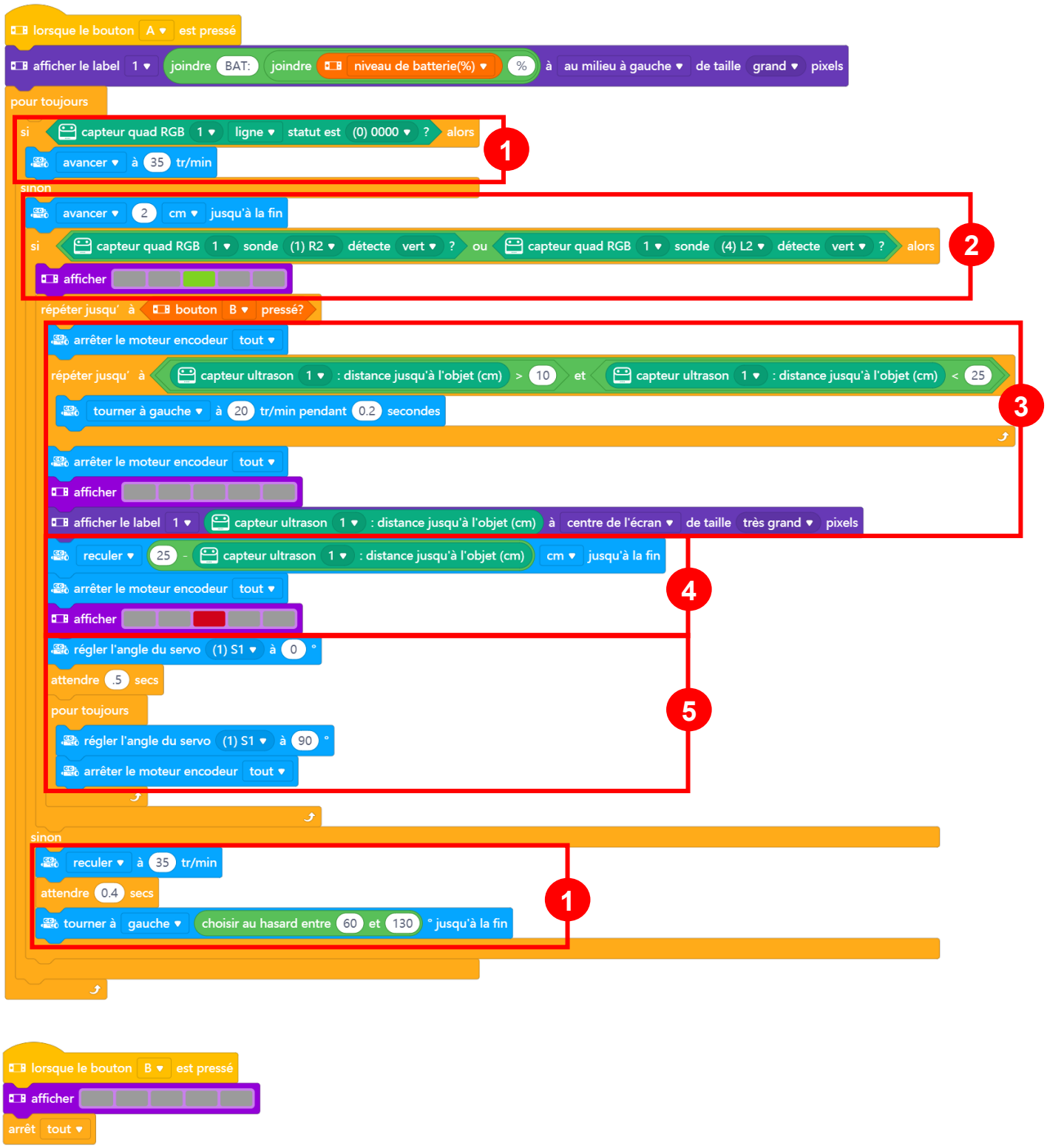
- Vérifier que les capteurs renvoient les valeurs attendues (en les affichant) dans les différentes situations.
- Vérifier que la batterie du robot est suffisamment chargée (au-delà de 20%).
- Vérifier que les capteurs sont correctement étalonnés ou initialisés (en particulier le capteur de ligne / couleurs).
- Revenir aux étapes précédentes

Optimisation des performances :

- Introduire une optimisation à la fois et vérifier son efficacité.
- Tenir compte des limites de performance des capteurs et du robot

Exemple de correction : MB2-BAB-F19.mblock

L'exemple de programme suivant correspond au regroupement des programmes MB2-BAB-F14 à MB2-BAB-F18. Ce programme peut être amélioré afin de rendre le mBot2 plus précis dans sa détection et son déplacement. L'algorithme de localisation du panier peut être amélioré, de nouveaux capteurs tels que la Smart Camera peuvent être utilisés pour améliorer les performances du robot.



Télécommander le robot pour marquer un panier

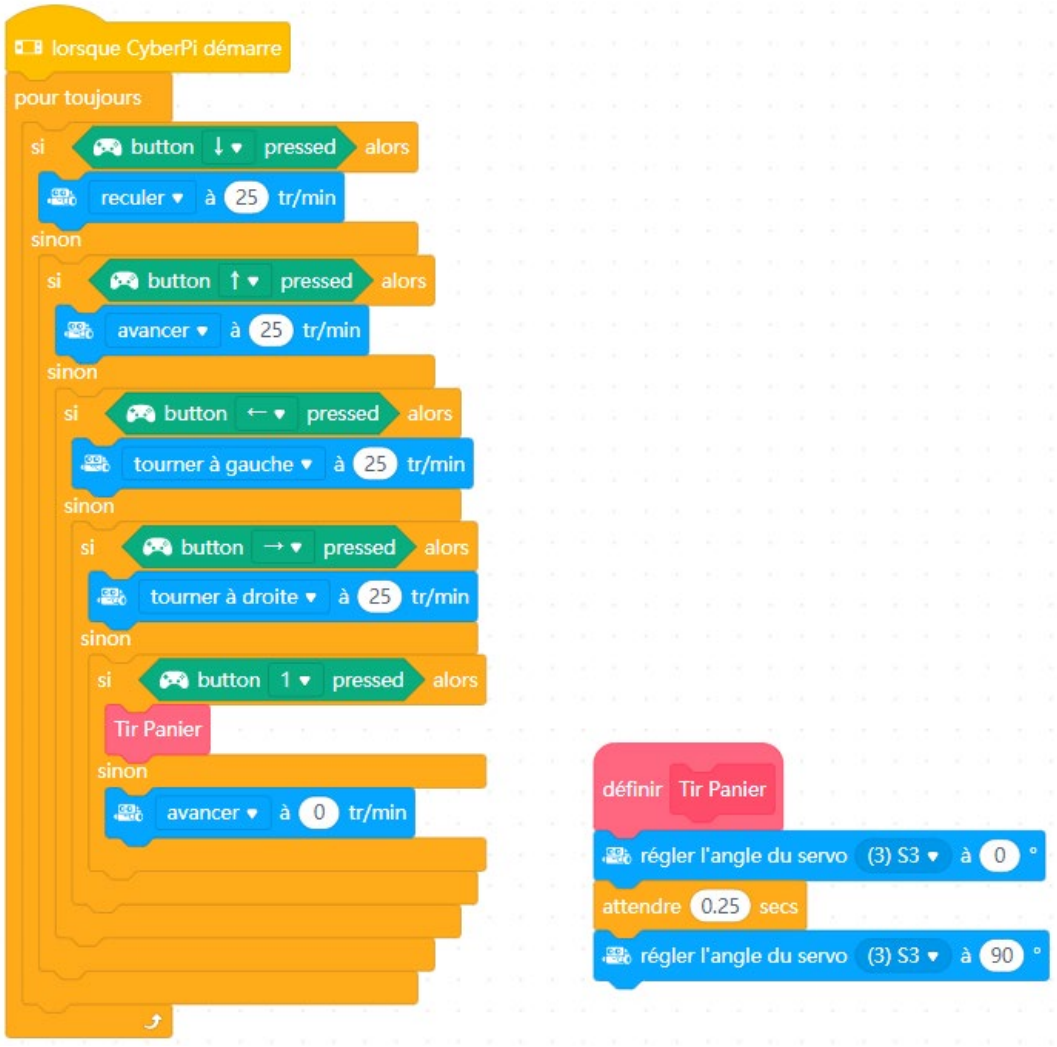
Contrôler le robot à l'aide de la manette Makeblock

But du programme : contrôler le mBot avec la manette Makeblock Bluetooth pour avancer, reculer, tourner à gauche et à droite et lancer la balle

Notion de programmation abordée : prise en main de la manette Bluetooth

Capteurs et accessoires utilisés : Manette Bluetooth

Exemple de correction : **MB2-BAB-F20.mBlock**

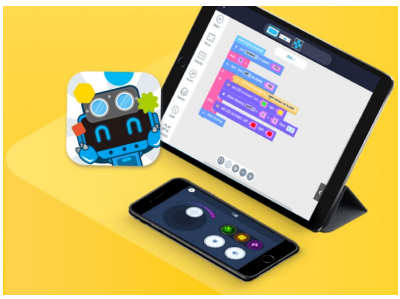


Pilotage de robots avec une tablette

L'environnement de programmation mBlock 5 permet de sauvegarder des programmes sur le serveur Makeblock. L'application mBlock qui fonctionne sur tablette Android ou iOS permet de lancer des programmes déposés sur ce serveur.
La scène de mBlock5 (Scratch) peut être utilisée pour créer une interface graphique permettant de piloter le mBot2 à distance (en Bluetooth) et déclencher le mécanisme de tir du robot. Le programme destiné à télécommander le mBot2 est relativement simple. La conception de l'interface utilisateur constitue un sujet riche pour travailler sur le design et sur l'ergonomie d'une application.

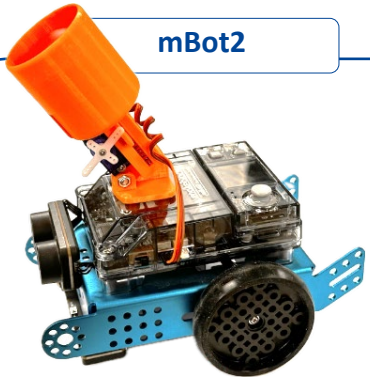
mBlock 5

mBlock mobile app



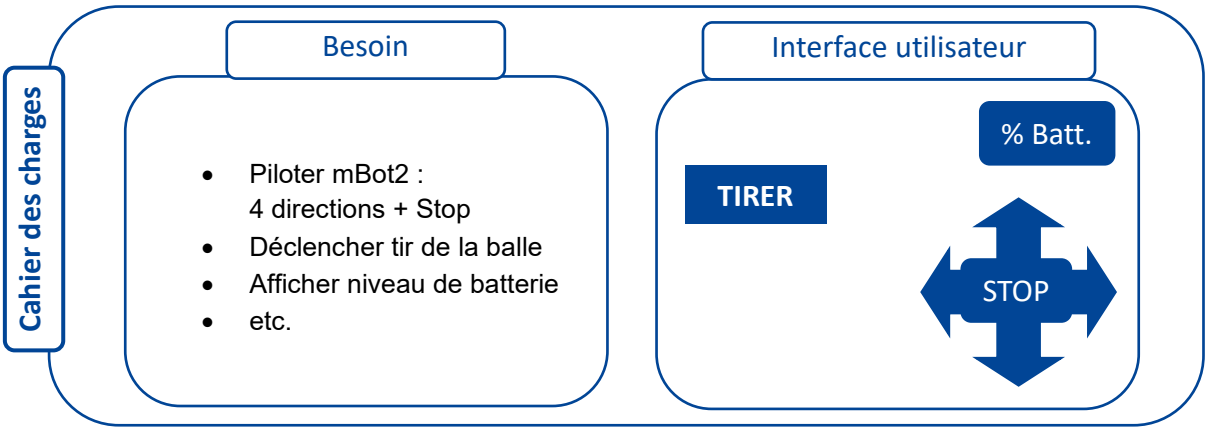
Bluetooth®

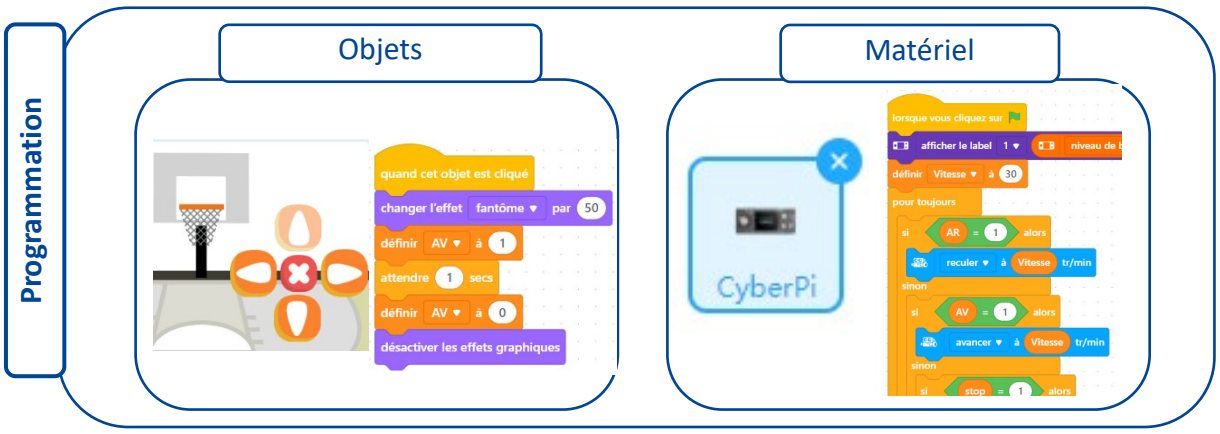
mBot2

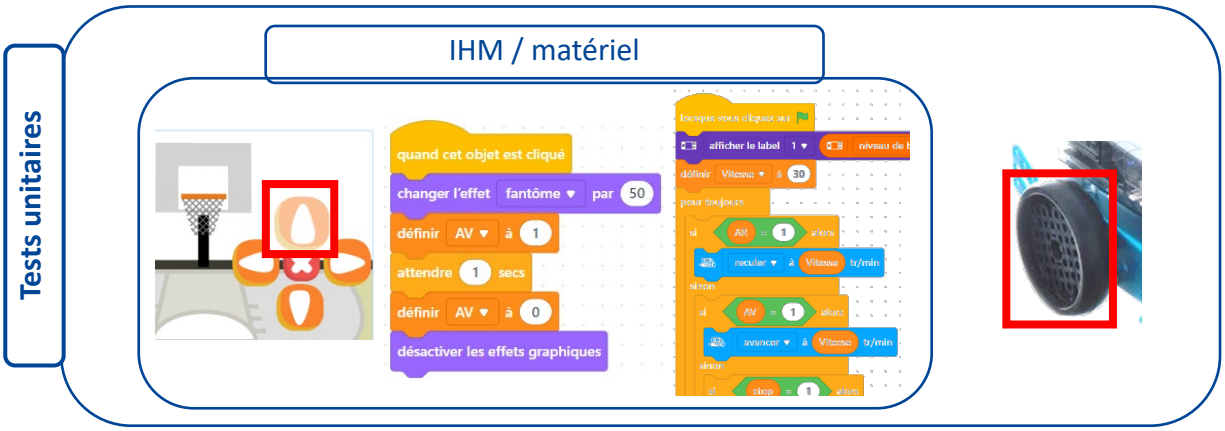


Exemple de programme : **MB2-BAB-F21.mblock**

Mise au point du programme

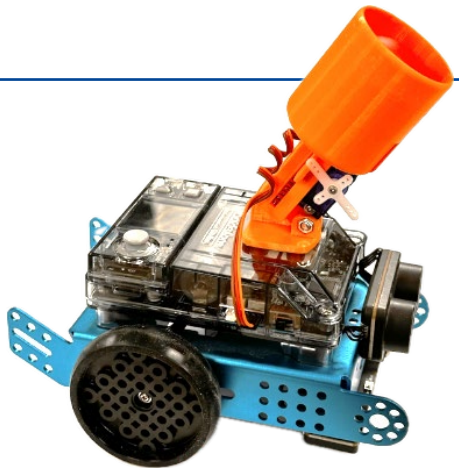


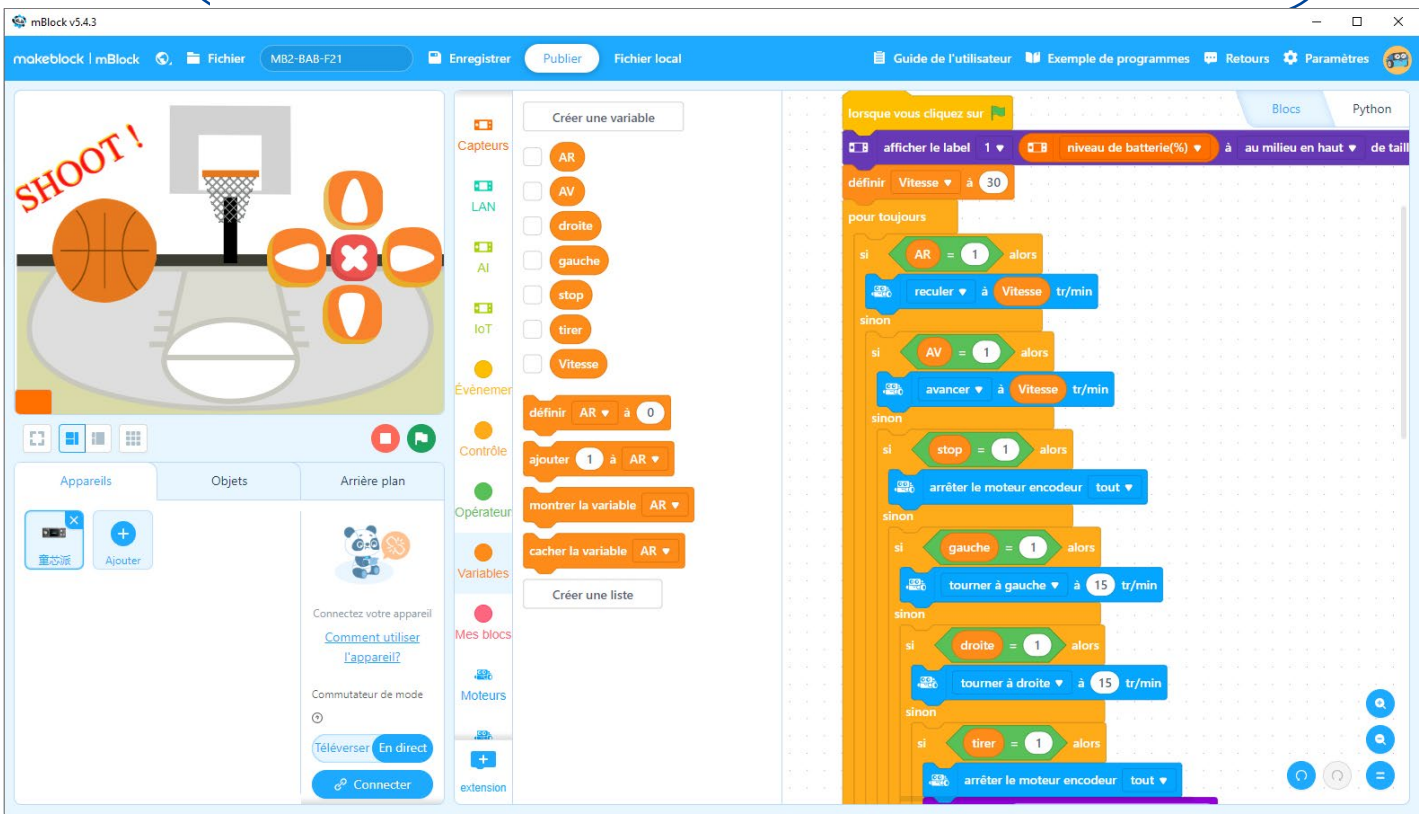




Test global

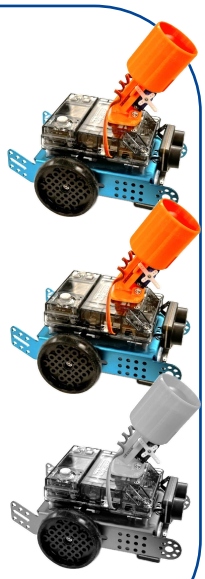

SHOOT !



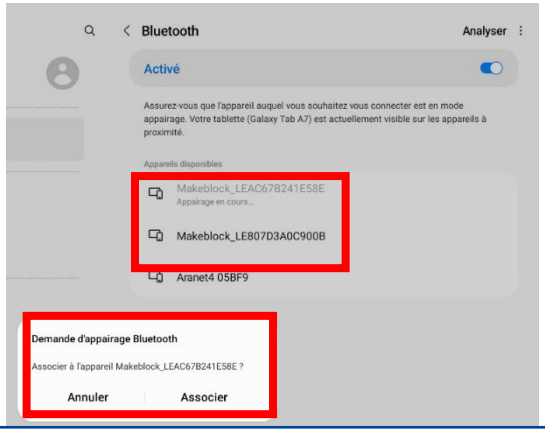

mBlock v5.4.3
makeblock | mBlock
Fichier
MB2-BAB-F21
Enregistrer
Publier
Fichier local
Guide de l'utilisateur
Exemple de programmes
Retours
Paramètres
SHOOT !
Appareils
Objets
Arrière plan
Ajouter
Connectez votre appareil
Comment utiliser l'appareil?
Commutateur de mode
Téléverser
En direct
Connecter
Capteurs
LAN
AI
IoT
Événemer
Contrôle
Opérateur
Variables
Mes blocs
Moteurs
extension
Créer une variable
AR
AV
droite
gauche
stop
tirer
Vitesse
définir AR à 0
ajouter 1 à AR
montrer la variable AR
cacher la variable AR
Créer une liste
Blocs
Python
lorsque vous cliquez sur
afficher le label 1 niveau de batterie(%) à au milieu en haut de taill
définir Vitesse à 30
pour toujours
si AR = 1 alors
reculer à Vitesse tr/min
sinon
si AV = 1 alors
avancer à Vitesse tr/min
sinon
si stop = 1 alors
arrêter le moteur encodeur tout
sinon
si gauche = 1 alors
tourner à gauche à 15 tr/min
sinon
si droite = 1 alors
tourner à droite à 15 tr/min
sinon
si tirer = 1 alors
arrêter le moteur encodeur tout

Appairage Bluetooth

1 Afficher l'identifiant de chaque mBot2

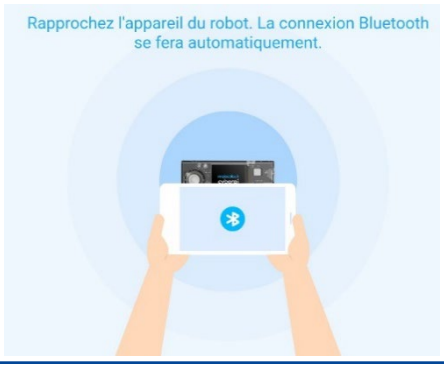


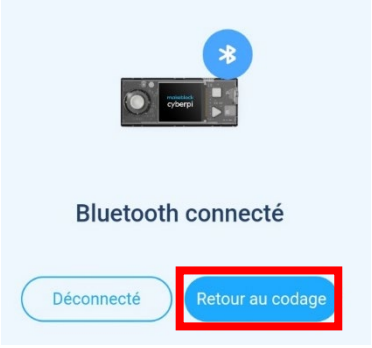
2 Associer chaque tablette à chaque mBot



3 Lancer le programme dans mBlock mobile app, sélectionner le CyberPi

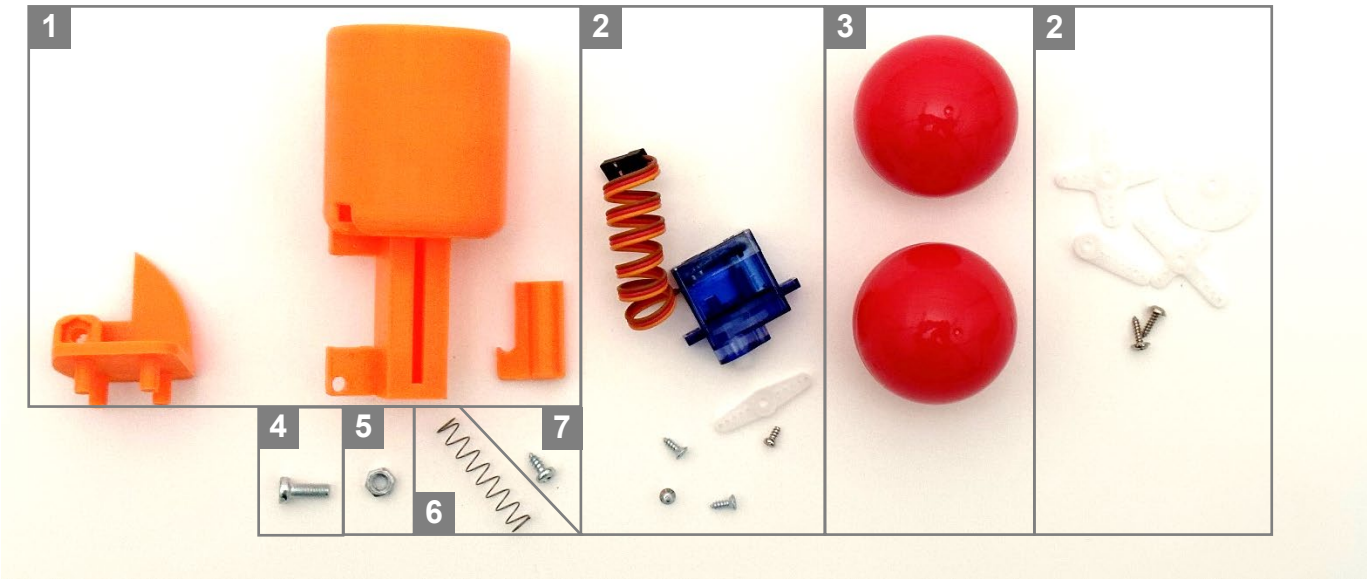
4



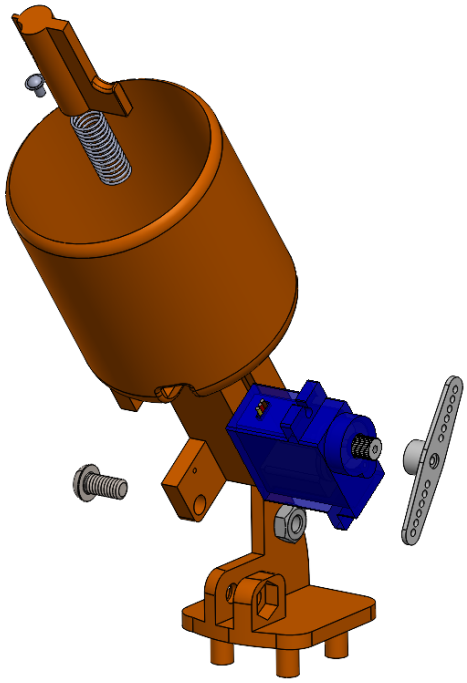


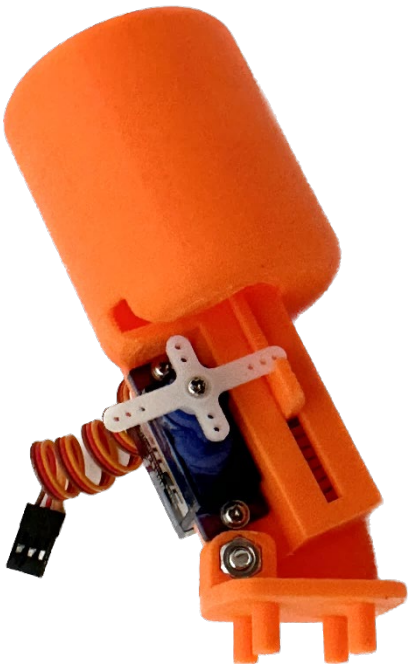
ANNEXE

Montage du kit lance balle



Repère	Référence	Désignation	Quantité
1	BAB-LB-MB2-IMP	Lance balle DEFI-BASKET pour mBot2 (base, canon, piston)	1
3	BAL-PP-CD40-R	Balle creuse D40 mm, 4.5g (2 demi-sphères à clipser)	2
5	ECR-N-ACZ-M4	Écrou acier hexagonal M4	1
4	VIS-ACZ-M4X10	Vis acier tête cylindrique fendue M4 x L10	1
7	VIS-TC-2M9X6M4	Vis tête cylindrique 2,9 x 6,4	1
2	POL-2820	Servomoteur micro à rotation continue FS90R + accessoires	1
6	RESSORT-SPIR2	Ressort de compression acier L:31,75 ØExt:6,096	1
2	VIS-TF-TX-2M2X6M5	Vis tôle tête fraisée empreinte TORX Ø 2,2 x L 6,5 mm	3

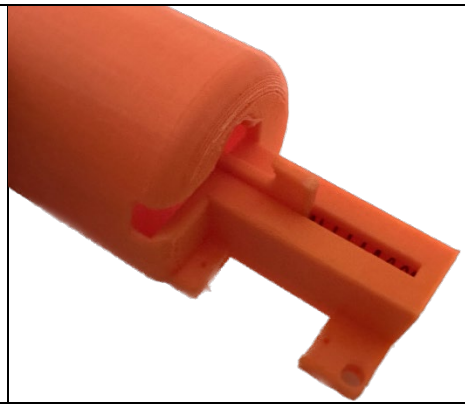




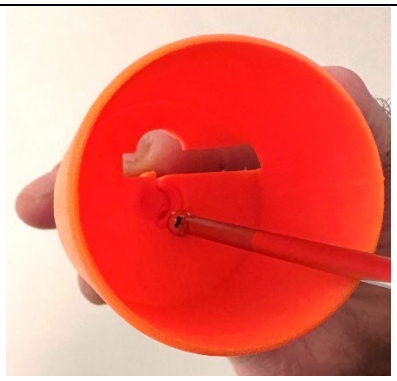
Positionner le ressort dans son logement

Insérer le piston

Comprimer le ressort en maintenant l'ergot du piston enfoncé

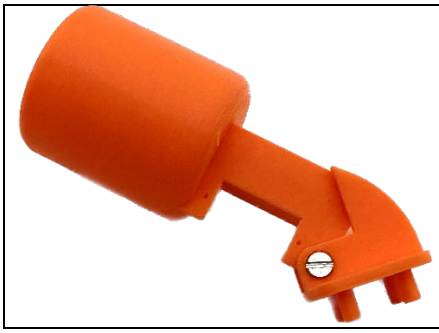


Mettre en place la vis de butée du piston (vis tête cylindrique 2,9 x 6,4)



Mettre en place l'écrou M4 dans la base

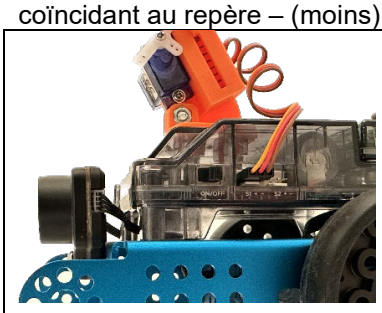
Mettre en place le canon sur sa base et la vis de serrage M4 x10

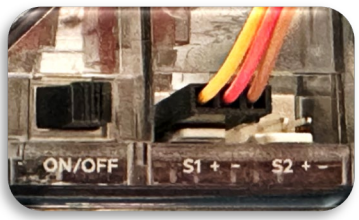


Mettre en place le palonnier en croix sur l'axe du servomoteur et visser sa vis de fixation présente dans le sachet d'accessoires

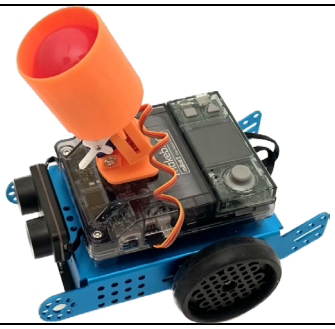
Mettre en place le servomoteur et le maintenir avec les 2 vis de fixation présentes dans le sachet d'accessoires

Connecter le servomoteur sur le port S1du mBot2 (port adjacent au bouton marche / Arrêt). Respecter la polarité de connexion indiquée sur la photo ci-contre : fil orange coïncident au repère S1, fil marron coïncidant au repère – (moins)

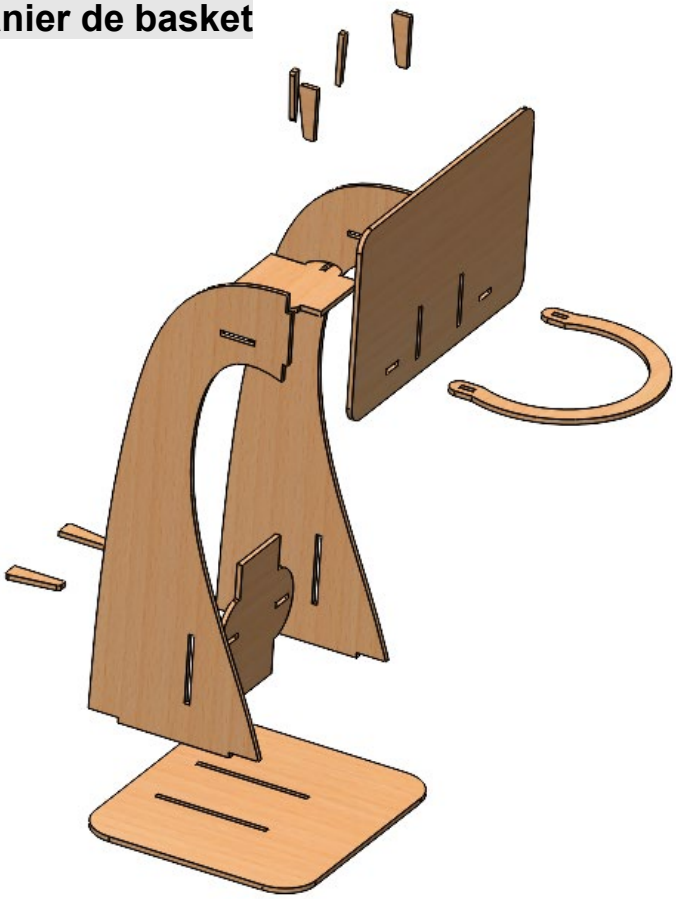




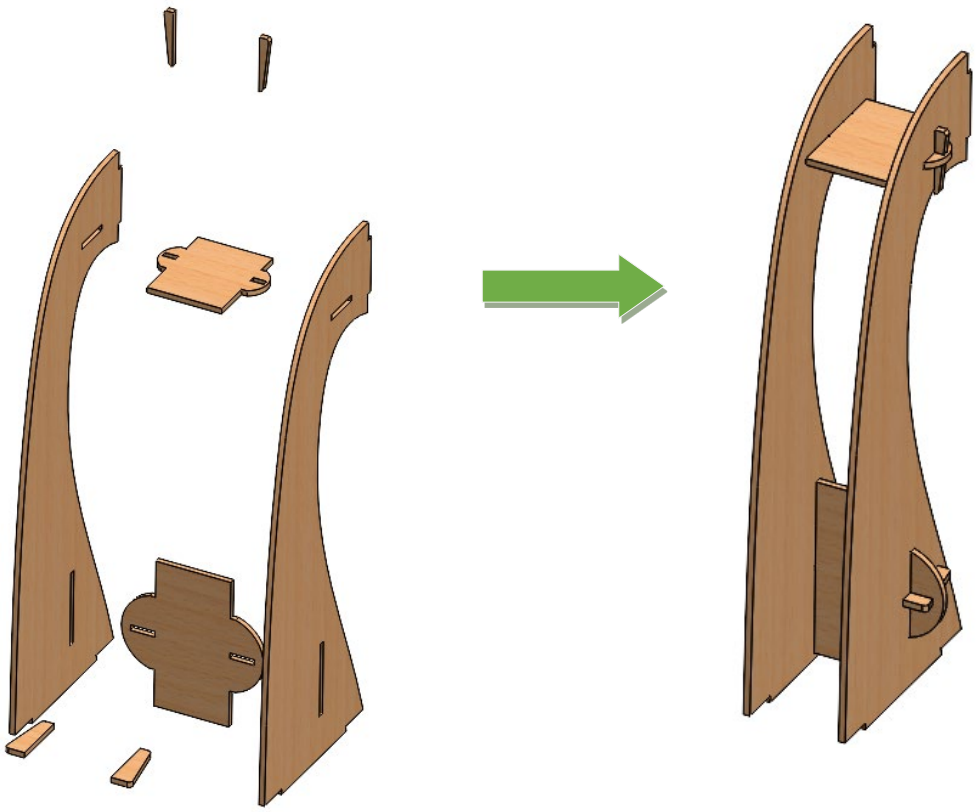
Mettre en place le lance balle sur le mBot2



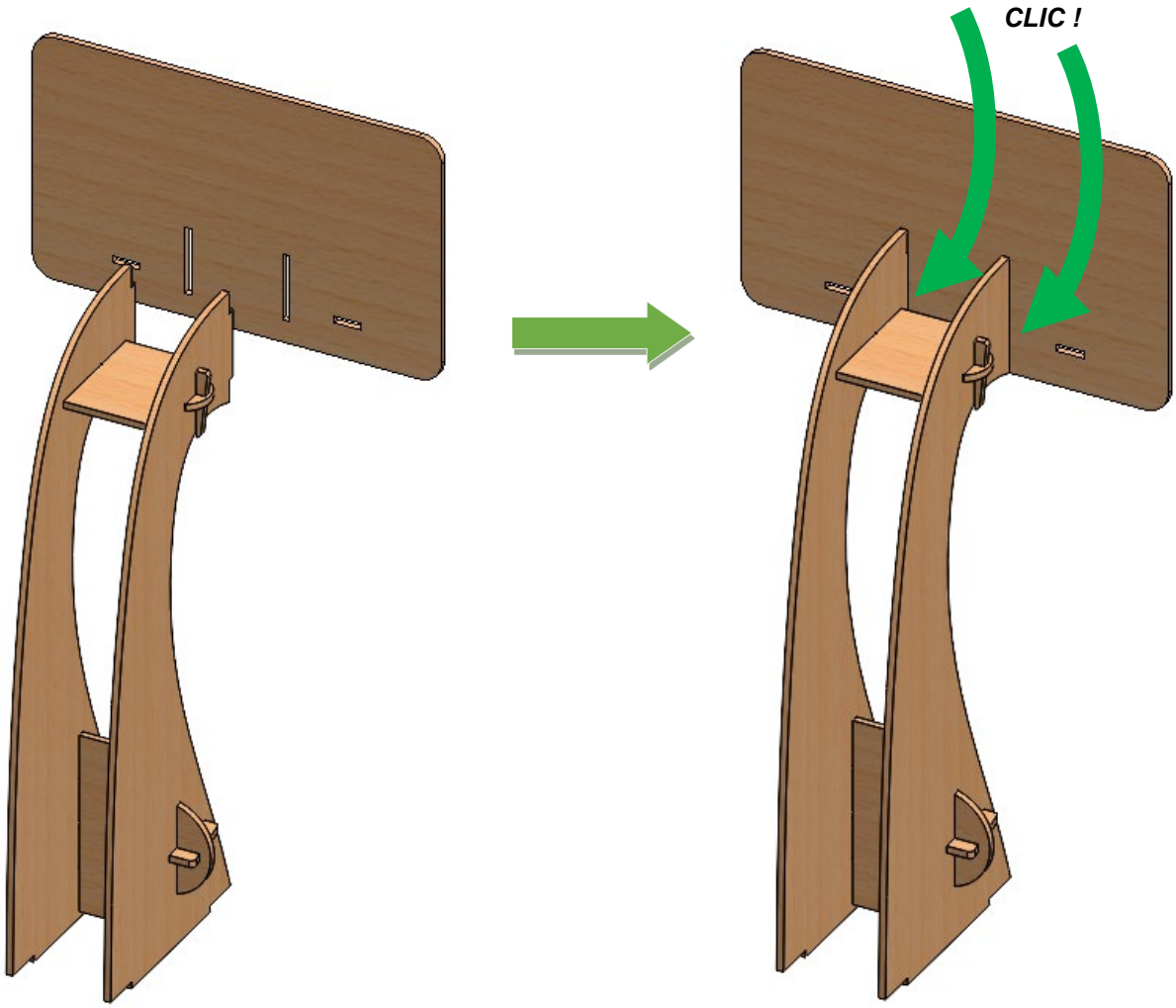
Montage du kit panier de basket



Emboîter les deux flancs avec les deux entretoises haute et basse, puis les bloquer avec les pins fournis.

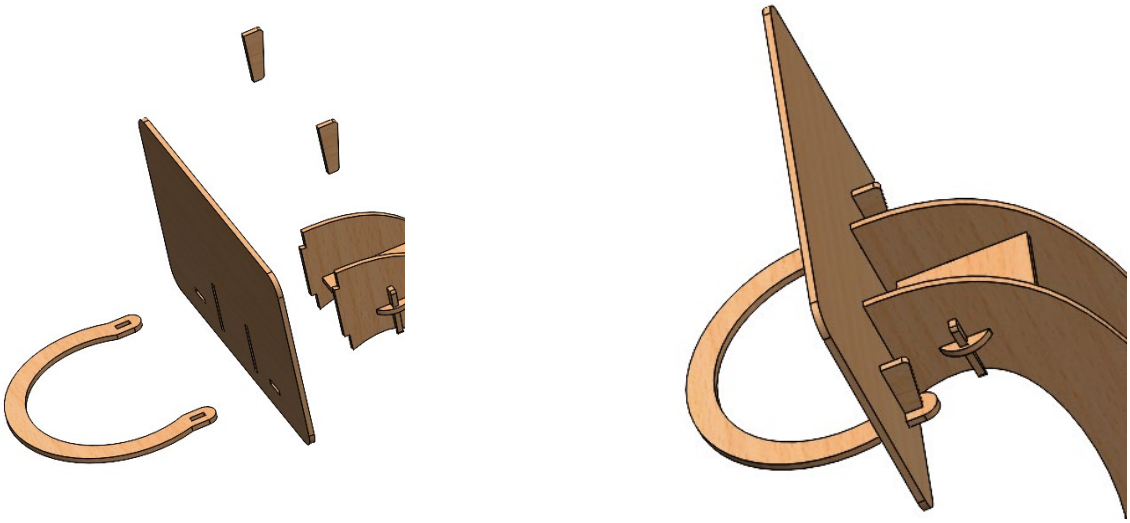


Emboîter le panneau avant **dans les flancs**

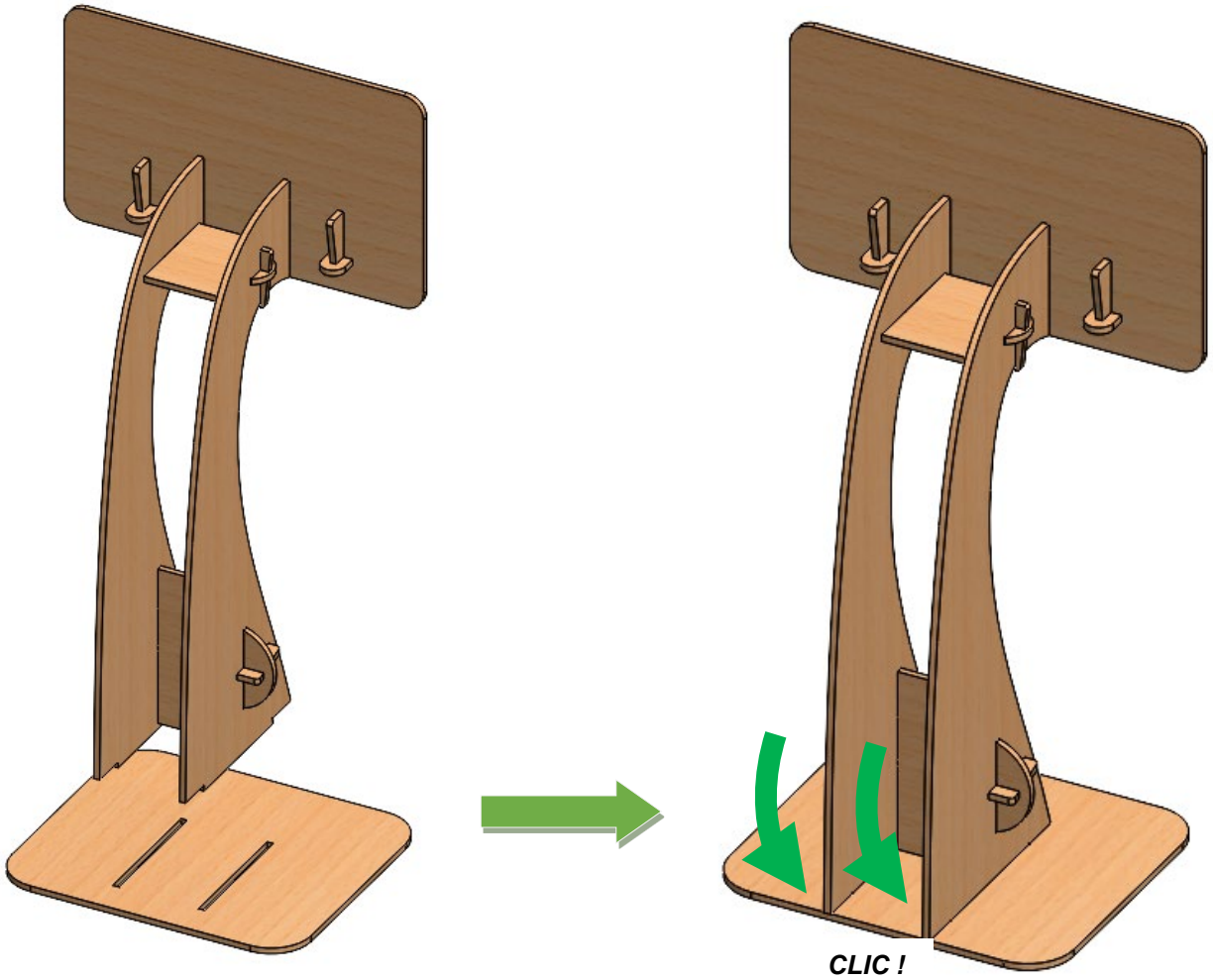


⚠ Le panneau est emboîté avec un ajustement serré afin de pouvoir le démonter facilement. Si besoin, mettre un point de colle afin de le maintenir définitivement.

Emboîter le cerceau dans le panneau et le fixer avec les pins



Emboîter la base dans les flancs

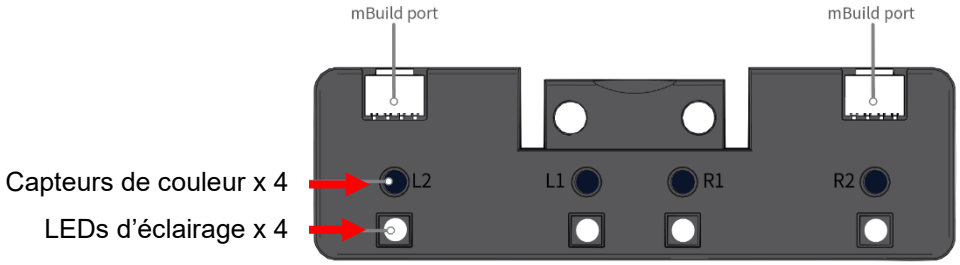


⚠ La base est emboîtée avec un ajustement serré afin de pouvoir la démonter facilement. Si besoin, mettre un point de colle afin de la maintenir définitivement.

Régler et maîtriser le fonctionnement du capteur de ligne Quad RGB

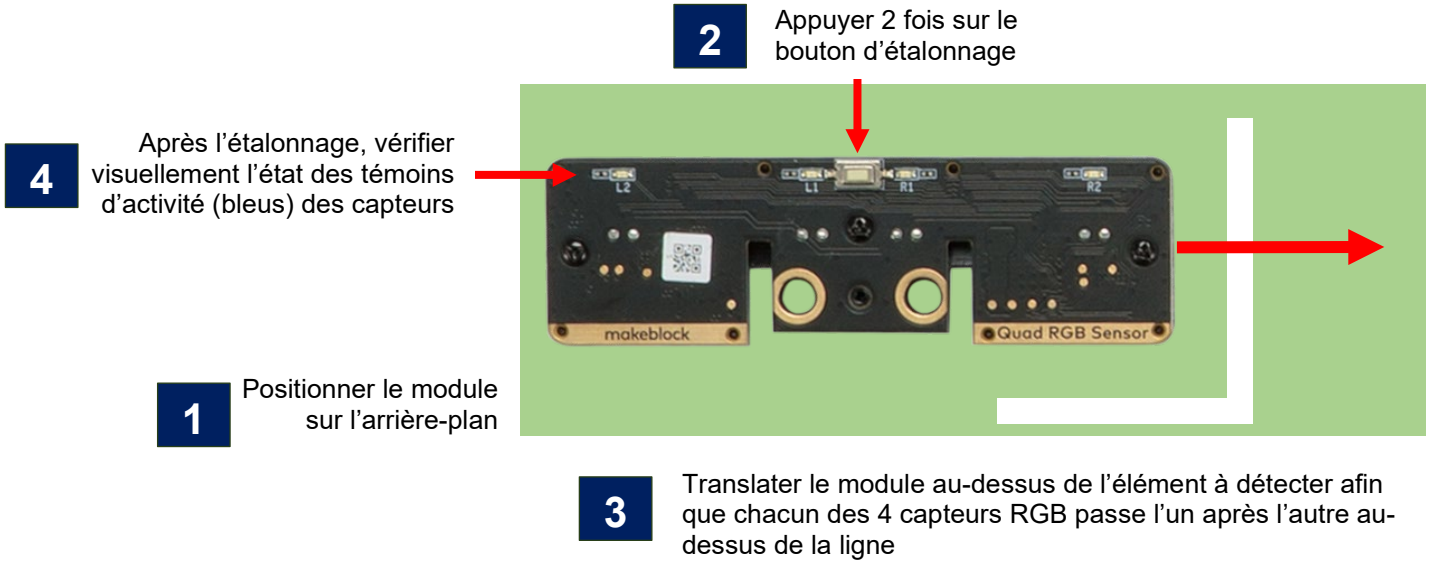
https://education.makeblock.com/help/mbuild-quad-rgb-sensor/

Le capteur Quad RGB utilise la lumière émise par les 4 LED d'éclairage comme lumière d'appoint afin de réduire considérablement les interférences de la lumière ambiante et détecter des couleurs. Le processus d'étalonnage permet de l'adapter à la lumière ambiante et de détecter de faibles variations de contraste (foncé / clair) et des couleurs. Les 4 capteurs permettent de prendre en charge différents scénarios de programmation.
Par exemple suivi de ligne avec les 2 capteurs centraux (L1, R1) et détection d'intersection avec les capteurs situés au bord du module (L2, R2).



Étalonnage de base

1. Placer le module au-dessus de l'arrière-plan (terrain vert foncé)
2. Appuyer rapidement deux fois sur le bouton du module : les 4 LEDs d'éclairage et les 4 LEDs témoins (bleues) clignotent
3. Translater le module sur la ligne à détecter (ligne blanche) afin que chacun des 4 capteurs de couleurs RGB passe l'un après l'autre au-dessus de la ligne
4. Vérifier le réglage à l'aide des LEDs témoins bleus



ATTENTION
Dès que le processus d'étalonnage est lancé, les LEDs d'éclairage clignotent et vous disposez de 2,5 secondes pour déplacer le module au-dessus de la ligne à détecter. À l'issue de ce temps, l'étalonnage est mémorisé pour chacun des 4 capteurs RGB. Il est important de vérifier l'efficacité de l'étalonnage en déplaçant de nouveau le module au-dessus de la ligne à détecter en vérifiant que les LEDs témoins (bleues) s'éteignent l'une après l'autre pour chacun des 4 capteurs RGB lorsqu'il est censé détecter la ligne.

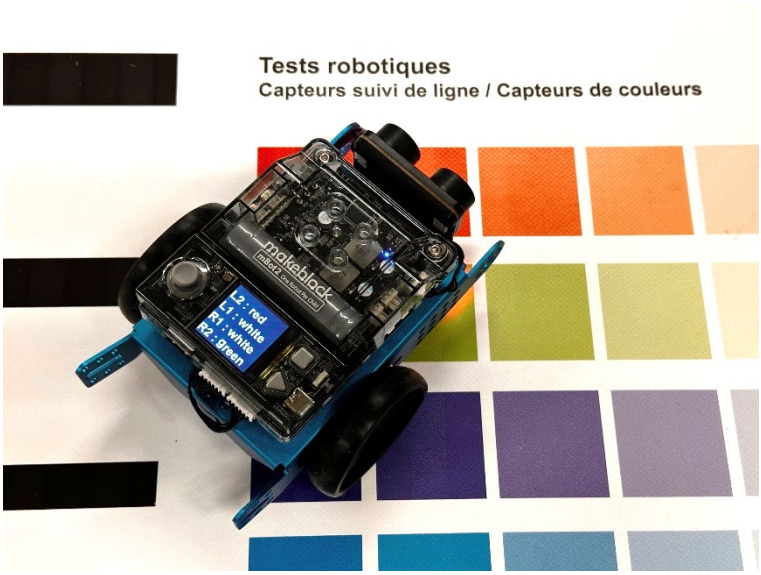
NOTE : on peut inverser la logique de fonctionnement des capteurs RGB en les positionnant initialement au-dessus de la ligne blanche qui est alors considérée comme l'arrière-plan.

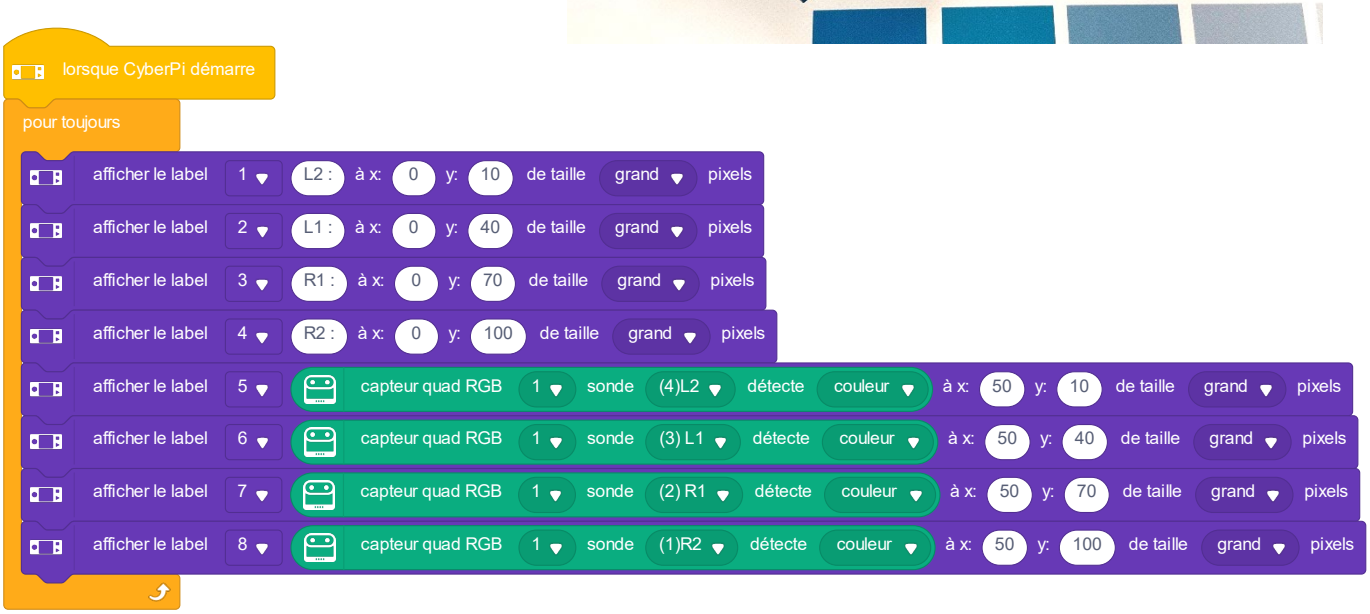
1. Placer les 4 capteurs de couleurs RGB au-dessus de l'arrière-plan (ligne blanche)
2. Appuyer rapidement deux fois sur le bouton du module : les 4 LEDs d'éclairage et les 4 LEDs témoins (bleues) clignotent
3. Faire pivoter le module au-dessus du terrain vert foncé afin que chacun des 4 capteurs de couleurs RGB quitte la ligne blanche
4. Vérifier le réglage

Afficher les couleurs détectées par le du capteur de ligne Quad RGB

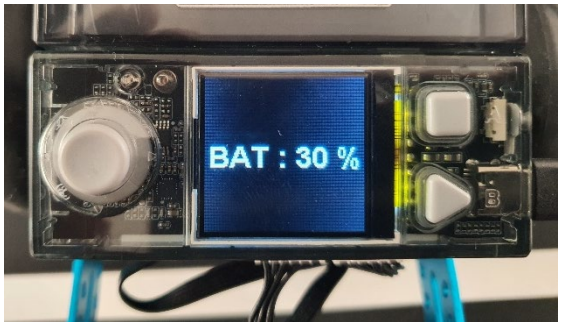
Exemple de programme : **TEST-COULEUR-L2-L1-R1-R2.mblock**

La couleur détectée par chaque capteur L2, L1, R1, R2 dépend de l'ambiance lumineuse générale (environnement sombre ou lumineux) et du support sur lequel les couleurs sont détectées (mat ou brillant).
Les quatre capteurs ont des tolérances de détection et leurs réponses ne sont pas forcément identiques pour une même couleur à détecter.





Surveiller l’état de charge de la batterie



www.a4.fr

Concepteur et fabricant de matériels pédagogiques