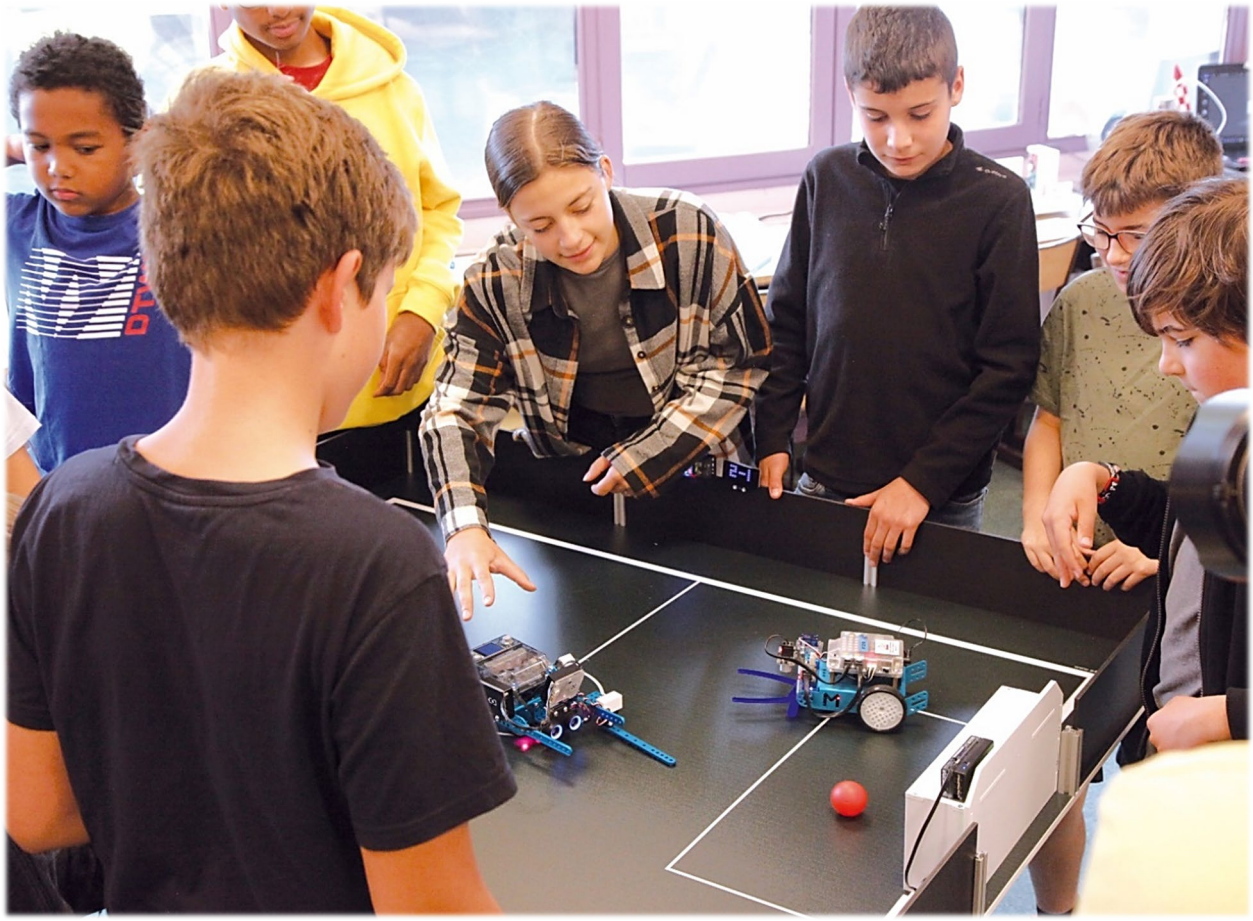
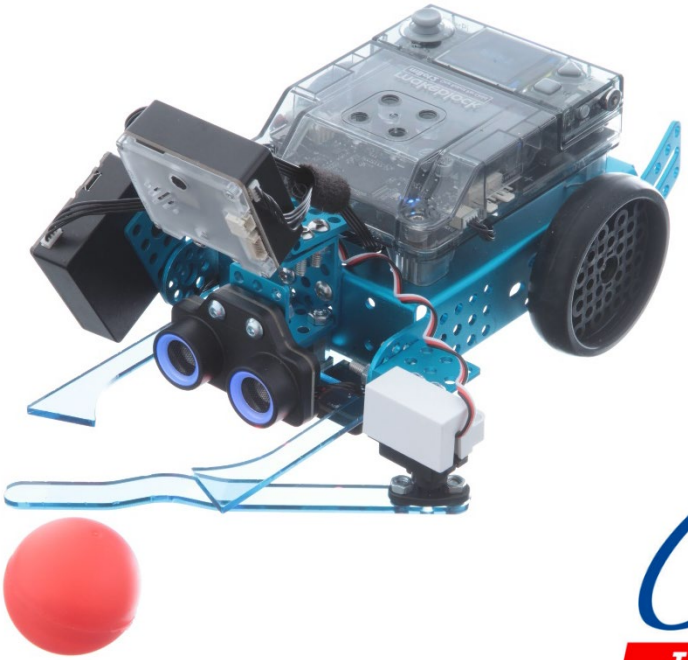
DéfiBut

Terrain de foot pour matchs robotiques avec mBot2





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

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

mBot2
Robot modulaire,

Kit DéfiBut
Terrain et accessoires pour matchs de foot

Manette Bluetooth controller

SOMMAIRE

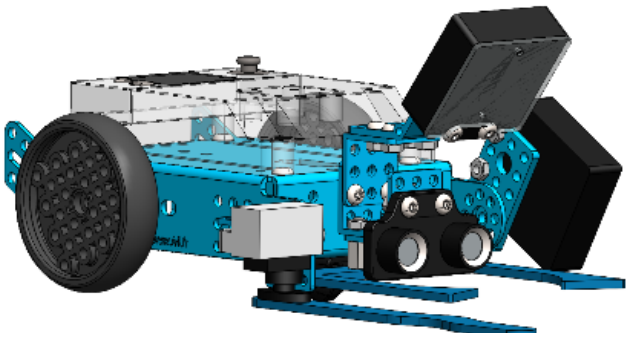
Introduction

DéfiBut

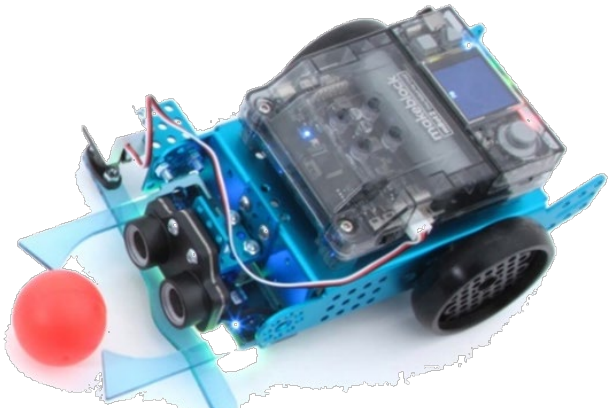
DéfiBut est une enceinte robotique conçue afin de confronter des robots programmés pour marquer des buts de manière autonome. Sa conception et les couleurs utilisées sont adaptées à la mise en œuvre du robot mBot2 équipé avec son capteur de ligne, son capteur de distance et le module Smart Camera qui permet de localiser la balle et le but adverse. Des activités complémentaires peuvent être menées pour gérer le comptage automatique des buts marqués, l'affichage du score ou la réalisation d'un système mécanique destiné à récupérer la balle et à la lancer.
La course de la balle utilisée est ajustable afin qu'elle ne traverse pas le terrain au premier tir. Ainsi, le robot devra effectuer plusieurs tirs pour se rapprocher du but.

Ce dossier propose un exemple pour réaliser ce défi avec un robot mBot2 programmé en blocs avec le logiciel mBlock 5. Pour marquer un but, le robot doit **rester dans l'aire de jeu**, **localiser la balle**, **se diriger vers elle, la récupérer, localiser le but adverse** puis **lancer la balle** dans sa direction.
Les élèves peuvent s'organiser en plusieurs groupes pour réaliser chaque partie du programme, concevoir le mécanisme de récupération de la balle et mettre en commun leurs réalisations pour rendre le robot opérationnel.

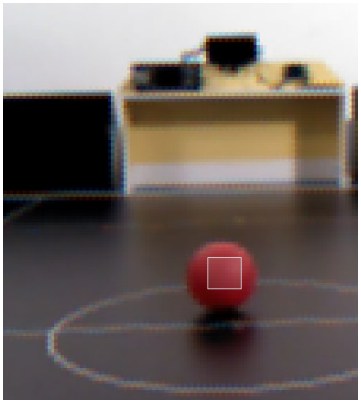
La conception du terrain DéfiBut est optimisée pour le robot mBot Explorer ou mBot2 équipés du module **Smart Camera** dont l'exploitation est facilitée par le choix des différentes couleurs (terrain, buts, balles). La ligne blanche qui délimite le terrain est détectable par le **capteur de ligne** du mBot Explorer ou du mBot2. Les murs qui entourent le terrain sont détectables par le **télémètre à ultrasons**. Le terrain peut être posé sur le sol ou sur deux tables standard côte à côte (dimensions 1,19 x 1,4 m) ; il est démontable en 2 parties.
Il est également possible de programmer le robot pour contrôler ses déplacements avec une **manette Bluetooth** ou avec une **application sur tablette** développées avec mBlock 5. Le **récupérateur / lanceur de balle** animé par un **servomoteur** peut être prototypé et réalisé avec différentes techniques : découpe manuelle de **carton rigide**, découpe numérique de **plastique** ou de **bois** avec une **découpe laser impression 3D**.



Conception à partir de la modélisation 3D

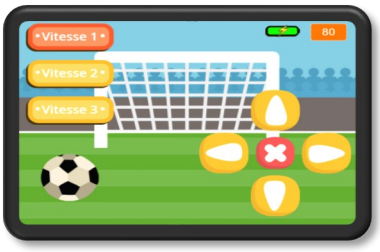


Mise au point du récupérateur / lanceur de balle



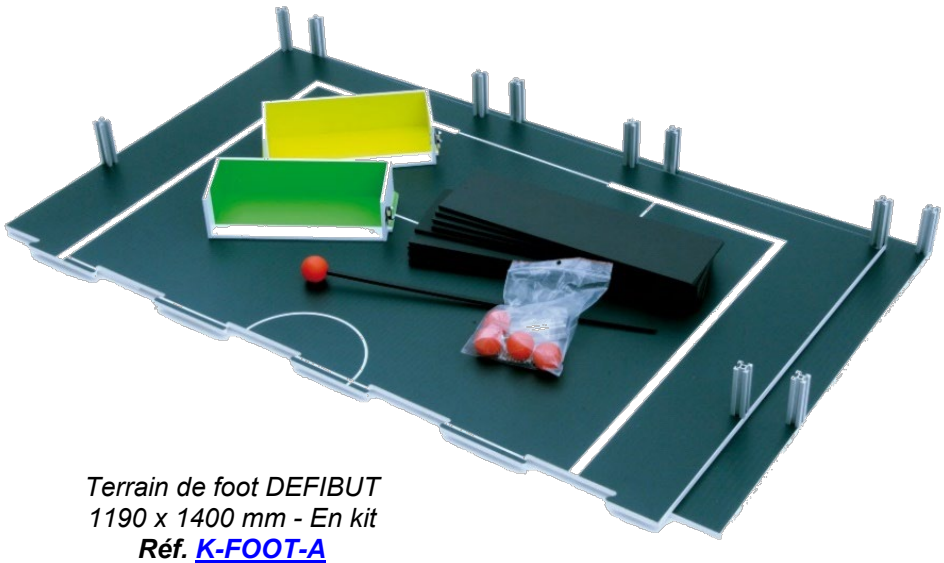
Utilisation de la Smart Caméra

Programmation pour pilotage manuel avec un contrôleur Bluetooth

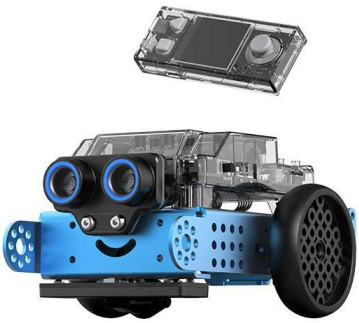


Création d'une application sur tablette

Matériels disponibles sur www.a4.fr



Terrain de foot DEFIBUT
1190 x 1400 mm - En kit
Réf. K-FOOT-A



Robot mBot2
Réf. MB-P1010132

Kit d'extension joueur de foot
pour robot mBot2
Réf. MB-JOUFOOT-MBV2

Manette Bluetooth controller
Réf. MB-P3060003

CyberPi Go Kit
Réf. MB-P1030156

Prérequis

Les exemples d'activités de programmation proposés dans ce dossier sont basés sur l'utilisation de robot mBot2 équipé avec le module Smart Camera.

- Disposer d'un robot mBot2, de son câble de programmation ou de la clé (dongle) Bluetooth Makeblock pour téléverser les programmes dans le robot
- Maîtriser le fonctionnement des capteurs et actionneurs utiles pour faire évoluer le robot de manière autonome afin de marquer un but
- Disposer d'un terrain DéfiBut
- Installer le logiciel mBlock 5, 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, Smart Caméra, servomoteur…)

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.

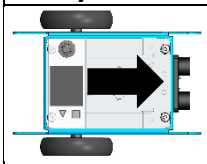
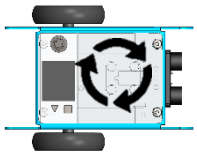
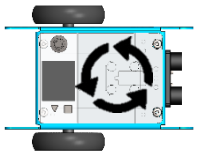
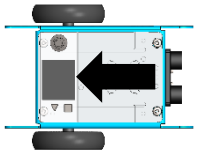
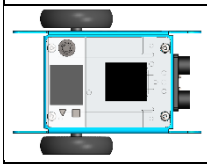
https://mblock.makeblock.com/en-us/download/

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 ces modules sont téléchargeables dans la rubrique DéfiBut sur www.a4.fr

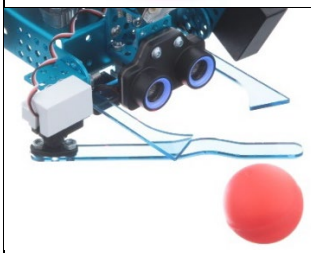
- **Présentation des matériels**
  Ce chapitre recense les éléments utilisés dans ce dossier.
- **Prise en main du robot mBot2**
  Ce chapitre propose une série d'exemples destinés à maîtriser chaque fonctionnalité utile à la réalisation du défi. Les programmes proposés sont classés par ordre de difficulté croissante.
- **DéfiBut - Marquer un but de manière autonome**
  Ce chapitre propose une solution qui s'appuie sur l'utilisation du module Smart Camera en complément des autres modules de base équipant le robot.
- **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

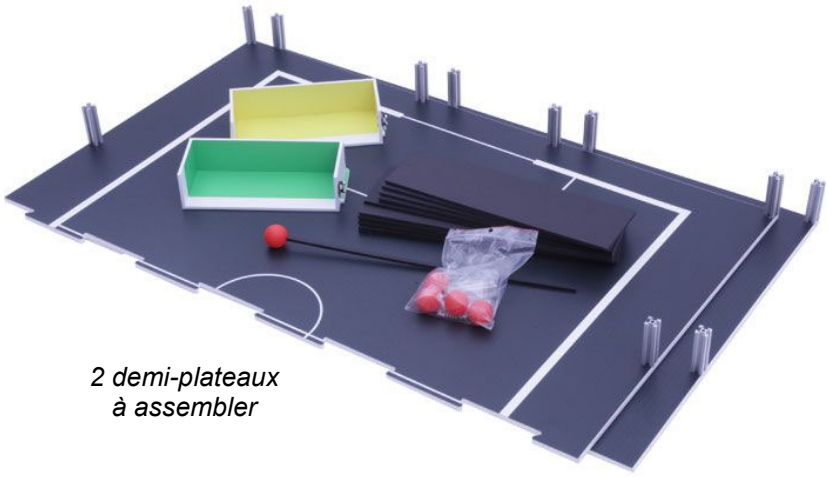
Symbole	Type de mouvement	Action sur les effecteurs
	Marche avant	Les 2 moteurs tournent en marche avant
	Tourner à droite	Les 2 moteurs tournent dans des sens opposés et le mBot tourne sur lui-même
	Tourner à gauche	Les 2 moteurs tournent dans des sens opposés et le mBot tourne sur lui-même
	Marche 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 Wifi, - Connectique servomoteur, connectique modules mBuild - Batterie externe

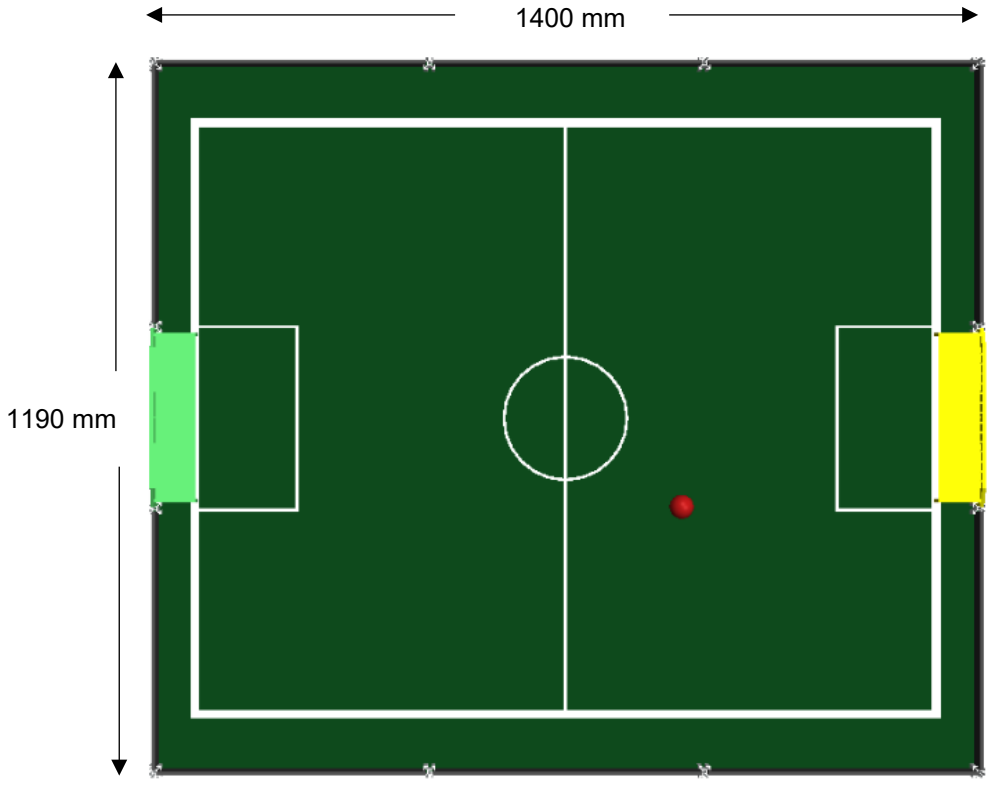
Moteurs 	Bloc moteur avec pignons métalliques et codeurs pour assurer des déplacements précis du robot en appliquant des consignes de distance à parcourir, d'angle et de vitesse de déplacement.
Capteur à ultrasons 	Télémètre à ultrasons pour mesurer la distance qui le sépare d'un obstacle. La sonde de 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.
Smart Camera 	Pour reconnaître des objets colorés, détecter des codes-barres ou des lignes. Elle se connecte à un robot mBot 2 et dispose de sa propre alimentation. Un bouton d'apprentissage sur la caméra permet d'enregistrer la couleur de 7 objets différents. La caméra retourne des informations de position des objets dans son champ de vision.
Servomoteur 	Servomoteur pilotable à 180°, couple 3,5 Kg.cm, dimensions 24 × 24 mm.
	Balle spéciale à course ajustable.
	Pièces mécaniques pour réaliser un récupérateur / lanceur de balles.
Terrain DéfiBut 	Terrain de dimensions 1190 x 1400 mm, adapté à l'utilisation de la Smart Camera. Sa couleur foncée et ses murs limitent les réflexions et les détections parasites. Les lignes blanches qui délimitent le périmètre du terrain sont détectables par le capteur de ligne du mBot2. La ligne médiane, le rond central et les lignes des surfaces de réparation sont suffisamment fins pour ne pas être détectées par le capteur de ligne du mBot2. Les cages de couleurs jaune et verte sont détectables par la Smart Camera. La course de la balle rouge, diamètre 4 cm, peut être ajustée pour éviter qu'elle ne quitte le terrain au premier tir.

Le terrain de jeu DéfiBut et les accessoires



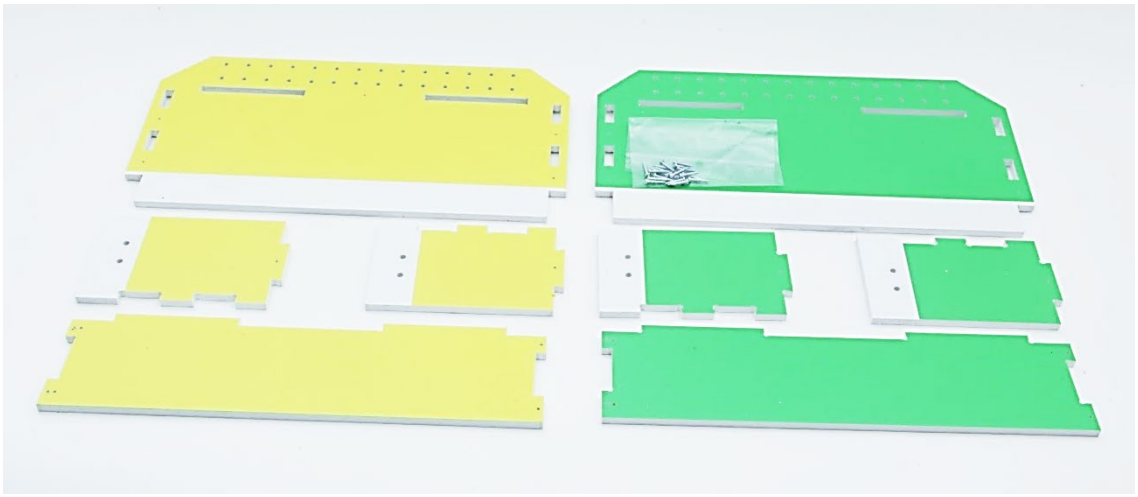
2 demi-plateaux à assembler

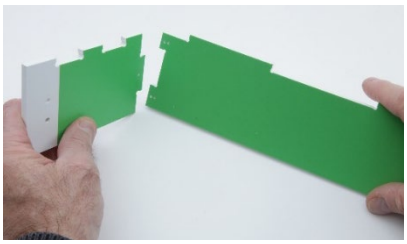
Poteaux en aluminium à visser sur les plateaux pour insérer et maintenir les cloisons à la périphérie du terrain

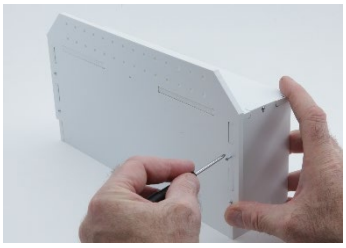


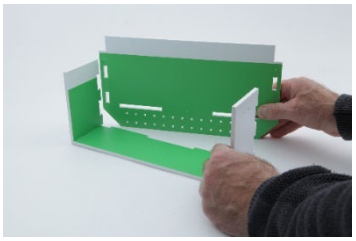
1 but jaune, 1 but vert pour différencier les camps

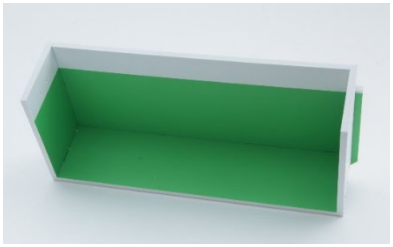
Montage des buts

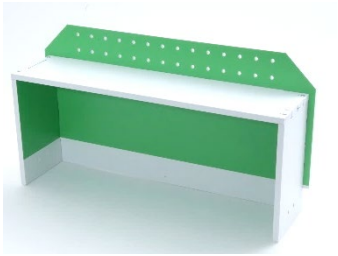












Kit balles à course ajustable

Le kit 5 balles D40 mm rouges, course ajustable + tige noire 50 cm (Réf. K-FOOT-BAL-5) est fourni avec la piste DéfiBut. Il est composé de 4 balles que l'on peut lester avec des billes acier afin d'en limiter leur course et d'une balle supplémentaire utilisée pour faciliter le réglage de la caméra. Cette balle est montée au bout d'une tige noire de 50 cm. Cet accessoire permet de manipuler la balle devant l'objectif de la Smart Caméra sans que les mains ne soient détectées à la place de la balle.

Balle lestée composée de deux demi-coques à assembler

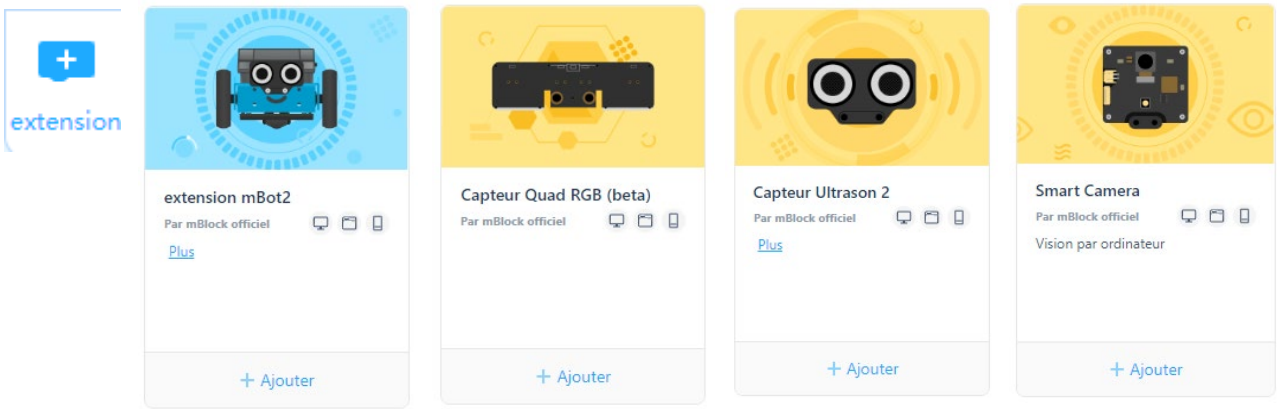
Montage d'une balle au bout d'une tige pour réaliser l'accessoire de réglage de la Smart Camera

Vérification du bon fonctionnement du robot et de ses capteurs principaux

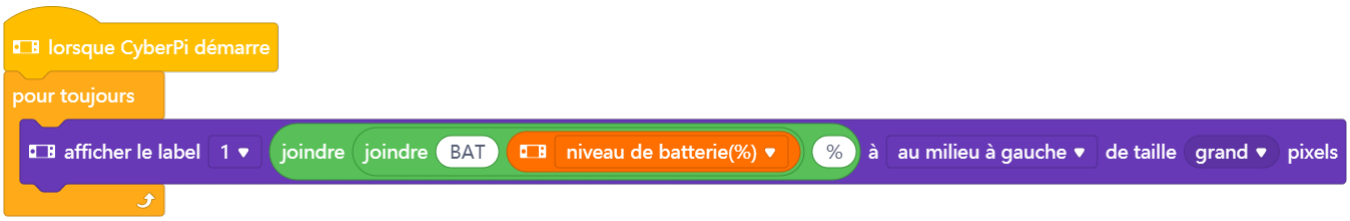
Lancer mBlock 5

Sélectionner le matériel CyberPi

Activer les extensions utiles :



Vérifier le bon chargement d'un programme et le taux de charge de la batterie



Un taux de charge de la batterie en dessous de 20% peut compromettre le fonctionnement des programmes.

Vérifier que les capteurs de ligne et de distance sont opérationnels

Voir l'ANNEXE en fin de document

Prise en main du robot mBot2

Ce chapitre propose une série de programmes 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 DéfiBut.

Liste des programmes :

Nom du fichier	Description	Objectif
MB2-DB-F1	Avancer puis s'arrêter	Maîtriser les déplacements
MB2-DB-F2	Avancer puis tourner	Maîtriser les déplacements
MB2-DB-F3	Avancer en décrivant un carré	Maîtriser les déplacements
MB2-DB-F4	Afficher un message	Afficher une information
MB2-DB-F4B	Afficher la valeur retournée par le capteur de distance	Visualiser la valeur renvoyée par le capteur de distance
MB2-DB-F5	S'arrêter avant un obstacle	Exploiter les informations provenant du télémètre à ultrasons
MB2-DB-F6	Éviter un obstacle	Exploiter les informations provenant du télémètre à ultrasons
MB2-DB-F7	Rebondir sur des obstacles	Exploiter les informations provenant du télémètre à ultrasons
MB2-DB-F7B	Afficher la valeur retournée par le module capteur de ligne	Visualiser la valeur renvoyée par le capteur de ligne
MB2-DB-F8	S'arrêter lorsqu'une ligne blanche est détectée	Exploiter les informations provenant du capteur de ligne
MB2-DB-F9	Tourner lorsqu'une ligne blanche est détectée puis avancer	Exploiter les informations provenant du capteur de ligne
MB2-DB-F10	Rebondir sur les lignes blanches	Exploiter les informations provenant du capteur de ligne
MB2-DB-F11	Tourner jusqu'à ce que la balle soit détectée	Exploiter les informations provenant de la Smart Camera
MB2-DB-F12	Propulser la balle à l'aide d'un lanceur de balle	Animer un servomoteur

NOTES IMPORTANTES :

- Le bon fonctionnement des programmes nécessite de maîtriser le fonctionnement des capteurs du mBot2 (plage de fonctionnement, valeurs aberrantes, zone physique de détection …).
- Les **capteurs de ligne quad RGB et la Smart Camera** sont sensibles à l'environnement lumineux qui peut changer au cours de la journée (rayons incidents du soleil, reflets, intensité et variation de l'éclairage artificiel ambiant …). Avant de se lancer dans la réalisation d'un programme, il est impératif de les régler **(étalonnage impératif avant utilisation).**
- Le bon fonctionnement des capteurs et du robot dépend du niveau de charge de la batterie.

Des fiches destinées à maîtriser le fonctionnement des capteurs et actionneurs sont proposées en **ANNEXE**

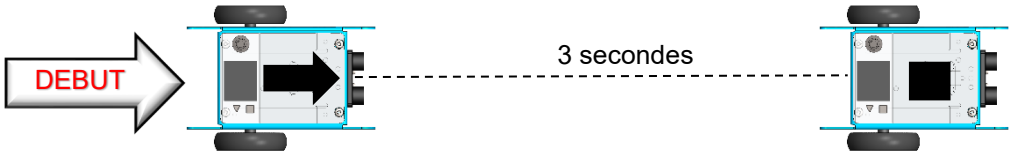
FICHE N°1 : avancer puis s'arrêter

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

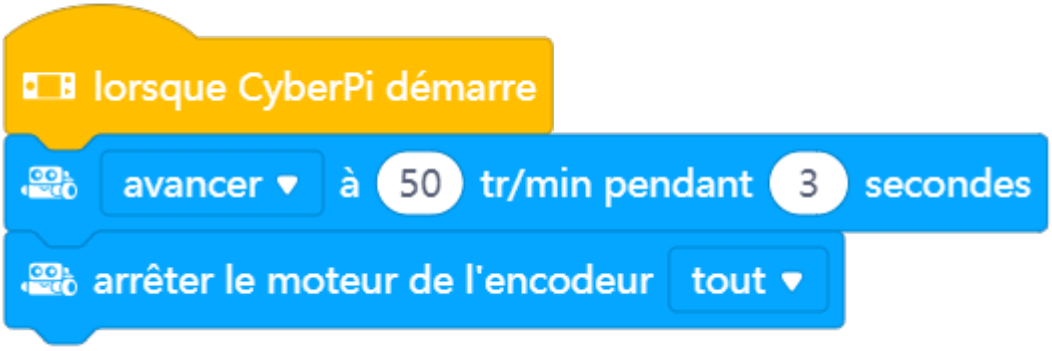
Notion de programmation abordée : séquence d'instructions

Actionneurs utilisés : moteurs

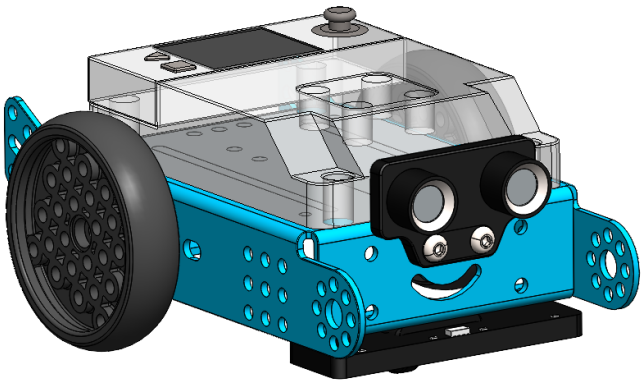
Synoptique :



Exemple de correction : MB2-DB-F1.mBlock



Exemple de configuration du robot :



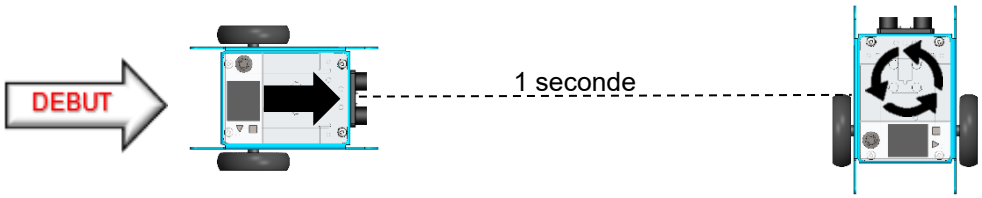
FICHE N°2 : avancer puis tourner

But du programme : avancer 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-DB-F2.mBlock

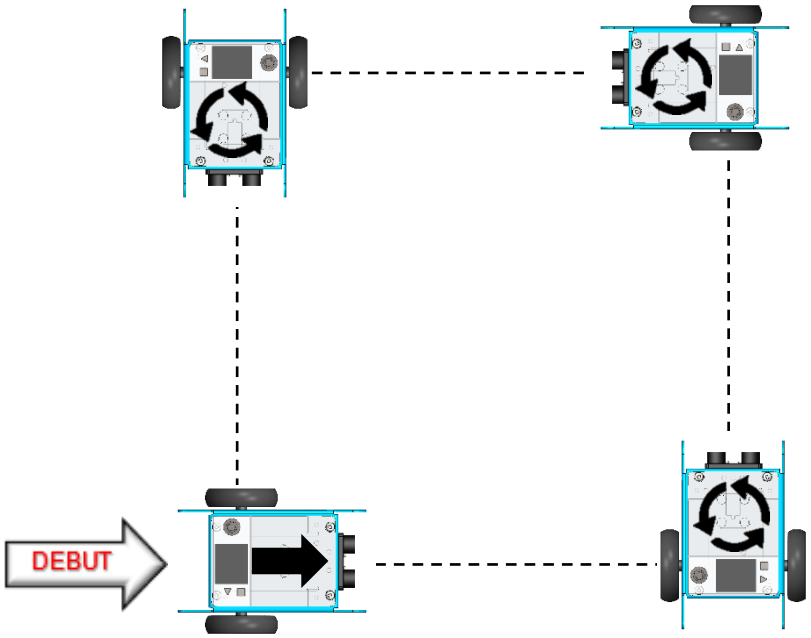
FICHE N°3 : avancer en décrivant un carré

But du programme : avancer en permanence en décrivant un carré

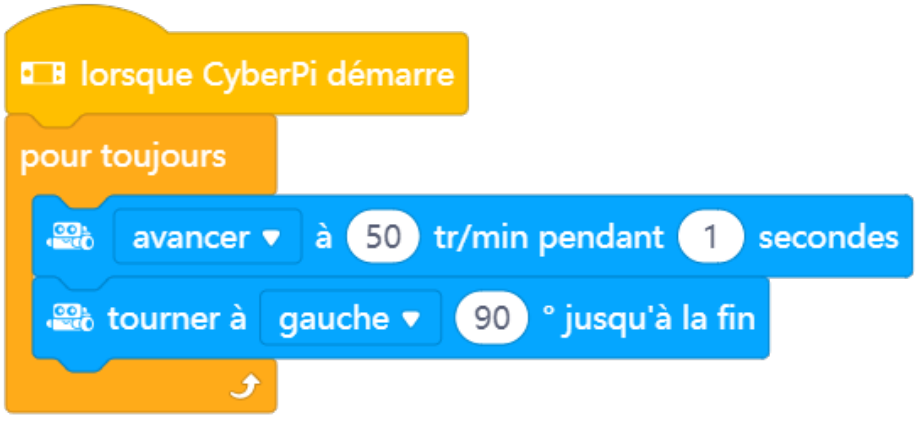
Notion de programmation abordée : séquence d'instructions, boucle

Actionneurs utilisés : moteurs

Synoptique :



Exemple de correction : MB2-DB-F3.mBlock



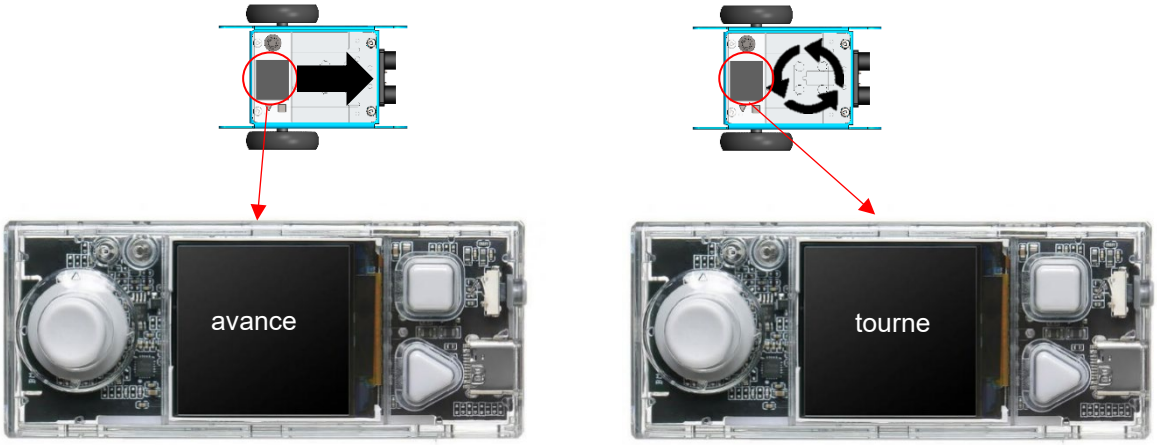
FICHE N°4 : afficher un message

But du programme : reprendre le programme précédent et afficher l'action réalisée par le robot sur le CyberPi. Écrire « avance » lorsque le robot se déplace en avant et « tourne » lorsqu'il pivote sur lui-même.

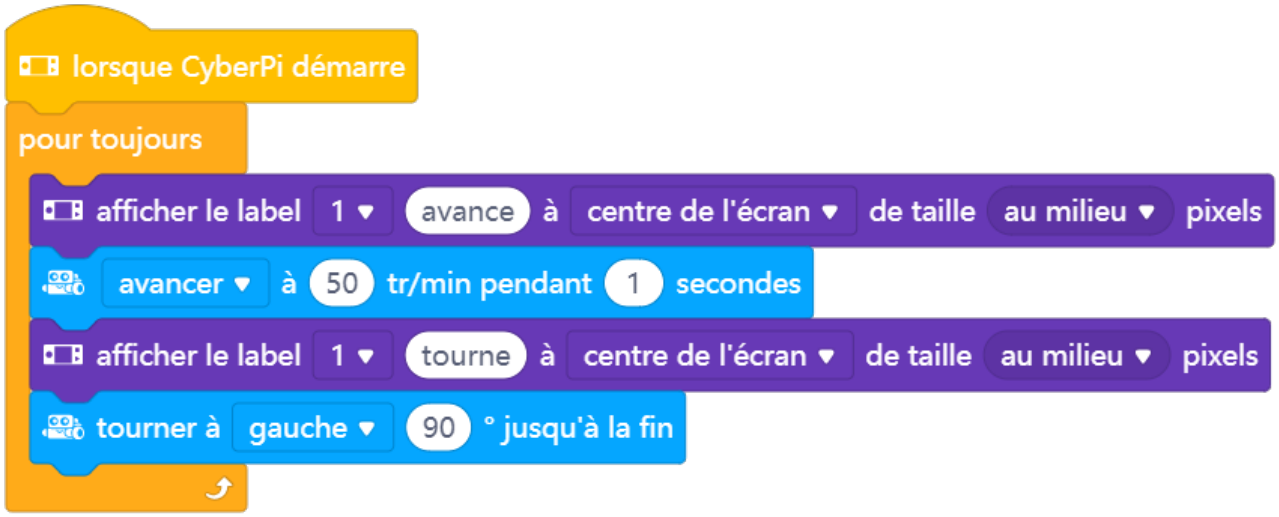
Notion de programmation abordée : séquence d'instructions, boucle

Actionneurs utilisés : moteurs, affichage

Synoptique :



Exemple de correction : MB2-DB-F4.mBlock



FICHE N°4B : afficher la valeur retournée le capteur de distance

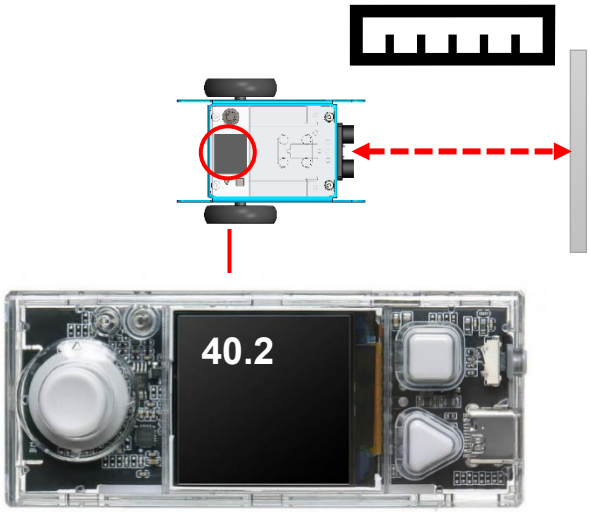
But du programme : afficher la valeur retournée par le télémètre à ultrasons sur l'écran du CyberPi.

Observer le fonctionnement du capteur :

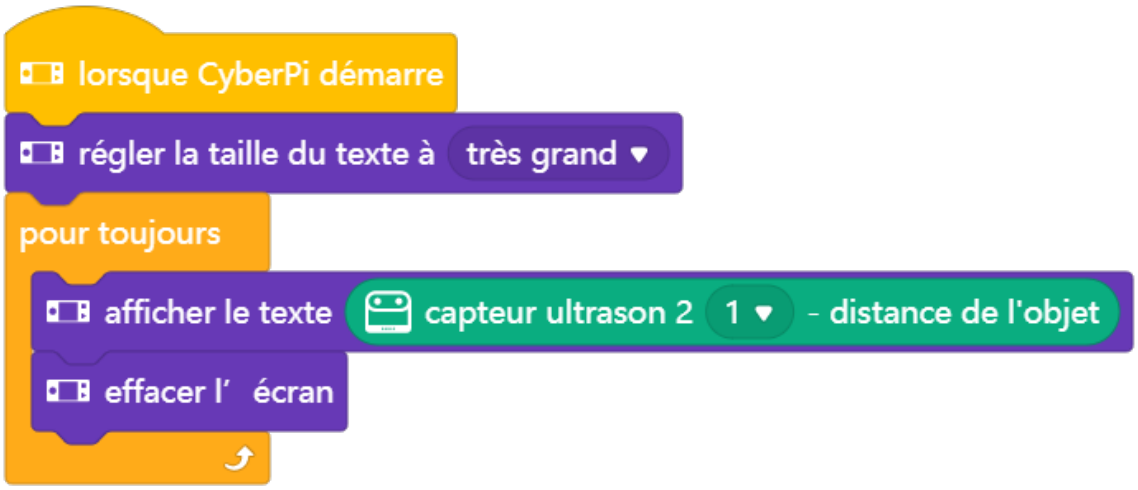
- Déplacer un objet devant le capteur et observer les valeurs retournées
- Relever les valeurs minimales et maximales que le capteur peut détecter
- Déterminer le rayon d'action du capteur (angle de détection)
- Comparer la valeur retournée par le capteur (valeur brute) à celle mesurée avec une règle (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

Capteurs / actionneurs utilisés : télémètre à ultrasons, affichage

Synoptique :



Exemple de correction : MB2-DB-F4B.mBlock



Observation : on remarque que le capteur retourne la valeur 300 si l'objet est « très » proche ou « très » éloigné du capteur.

FICHE N°5 : s'arrêter avant un obstacle

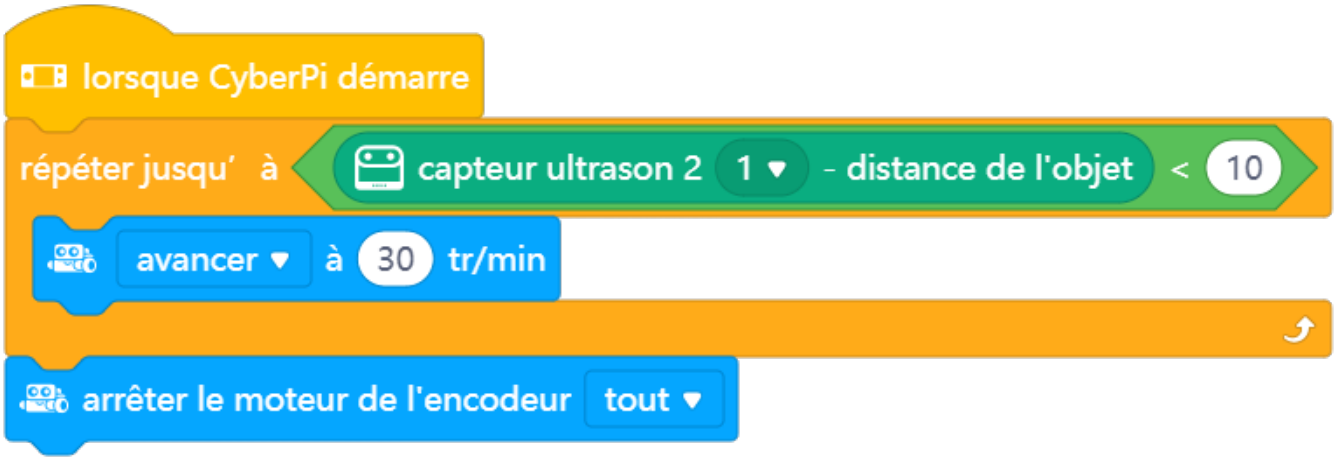
But du programme : avancer à 30 tours par minute, s'arrêter lorsqu'un objet est détecté à moins de 10 cm

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

Capteur utilisé : télémètre à ultrasons

Synoptique :

Exemple de correction : MB2-DB-F5.mBlock



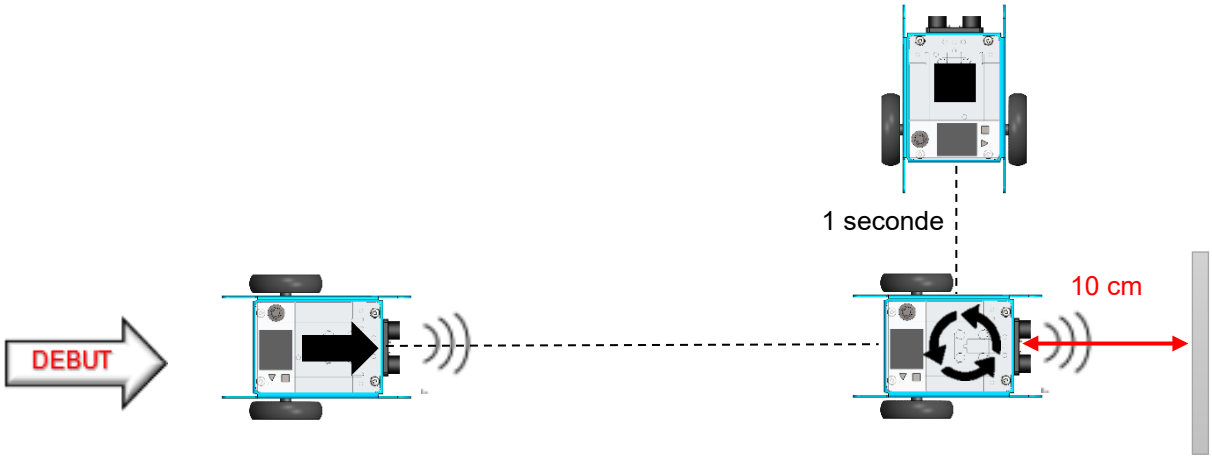
FICHE N°6 : éviter un obstacle

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.

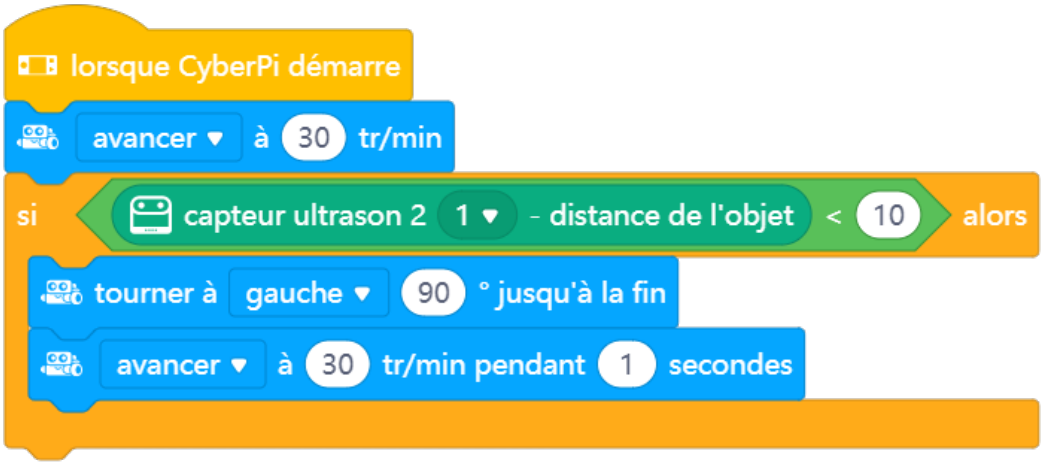
Notion de programmation abordée : séquence d'instructions, test conditionnel

Capteur utilisé : télémètre à ultrasons

Synoptique :



Exemple de correction : MB2-DB-F6.mBlock



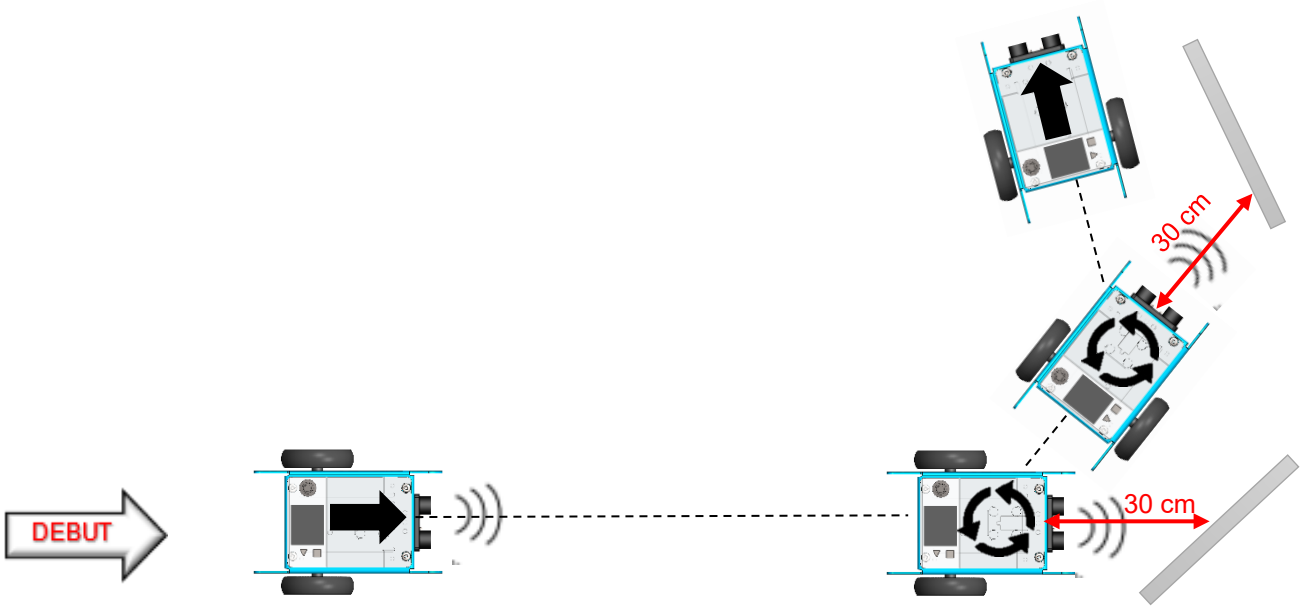
FICHE N°7 : rebondir sur des obstacles

But du programme : avancer en permanence à 30 tours par minute, pivoter à gauche de 30° si un obstacle est détecté à moins de 30 cm.

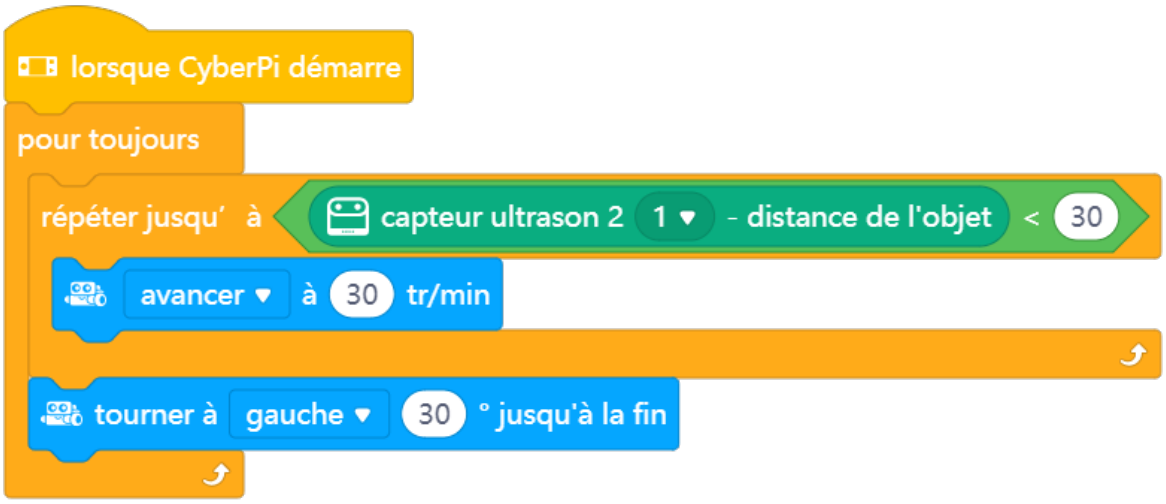
Notion de programmation abordée : séquence d'instructions, boucle, test conditionnel

Capteur utilisé : télémètre à ultrasons

Synoptique :



Exemple de correction : MB2-DB-F7.mBlock



FICHE N°7B : 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.

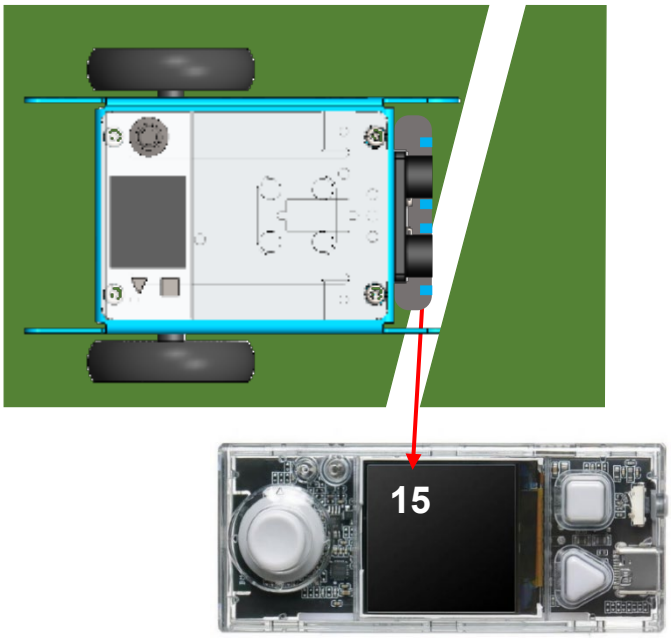
Le module capteur de ligne est équipé de 4 capteurs distincts capables de détecter différentes couleurs.

Questionnement :
Positionner le capteur au-dessus du terrain (vert foncé) puis le déplacer au-dessus de la ligne blanche qui délimite le terrain.

- 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 :



Exemple de correction : MB2-DB-F7B.mBlock

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

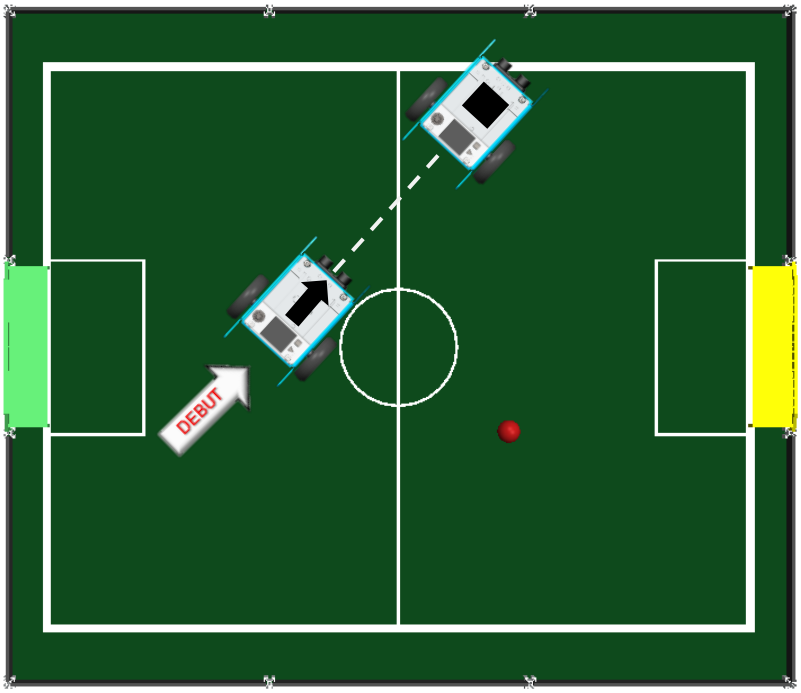
FICHE N°8 : s'arrêter lorsqu'une ligne blanche est détectée

But du programme : avancer à 10 tours par minute, s'arrêter lorsqu'une ligne blanche est détectée

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

Capteur utilisé : module capteur de ligne

Synoptique :



Exemple de correction : MB2-DB-F8.mBlock

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

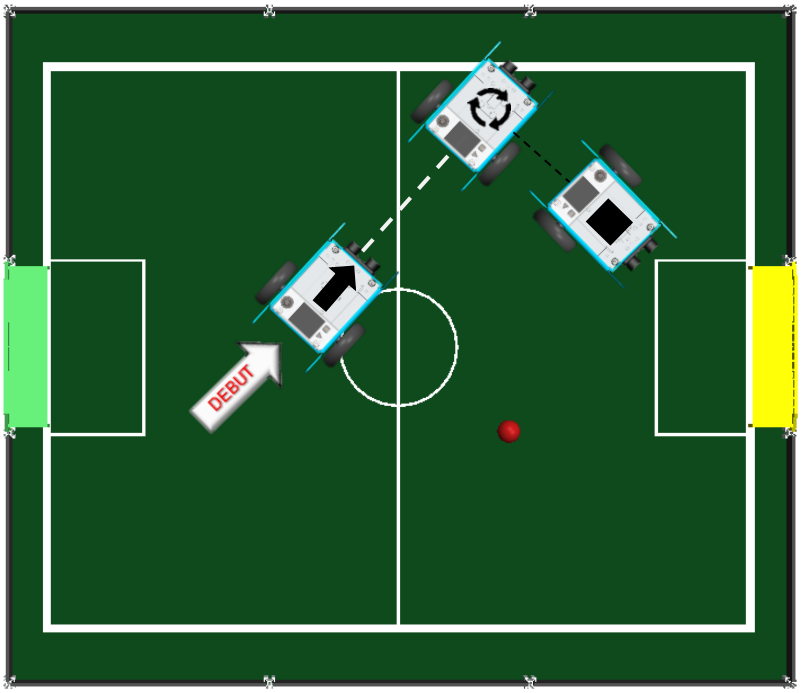
FICHE N°9 : tourner lorsqu'une ligne blanche est détectée puis avancer

But du programme : avancer à 10 tr/min, tourner à droite de 90° lorsqu'une ligne blanche est détectée, puis avancer à 30 tr/min pendant 1 seconde.

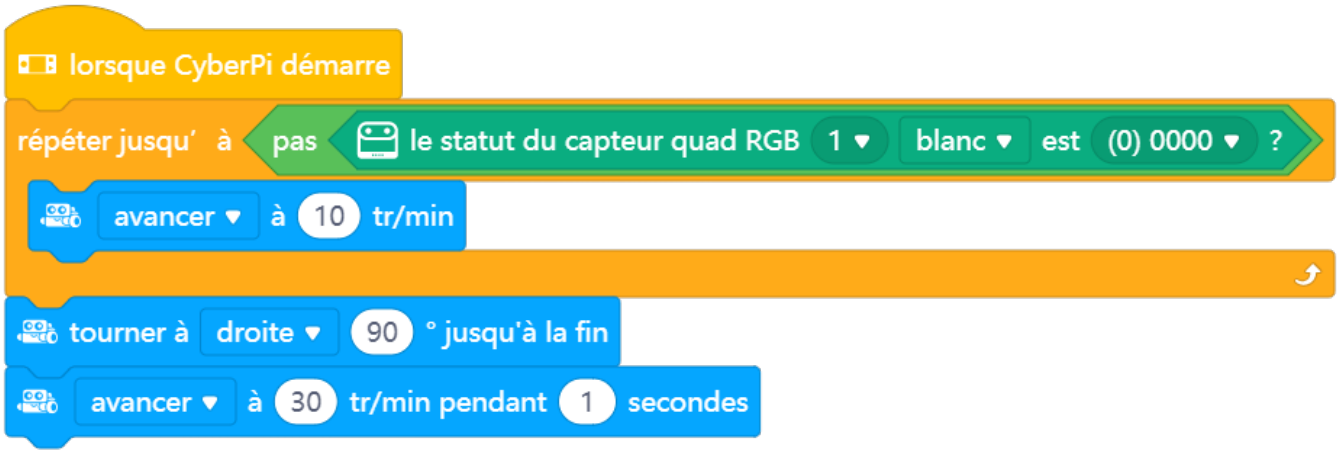
Notion de programmation abordée : séquence d'instructions, boucle, test conditionnel

Capteur utilisé : module capteur de ligne

Synoptique :



Exemple de correction : MB2-DB-F9.mBlock



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

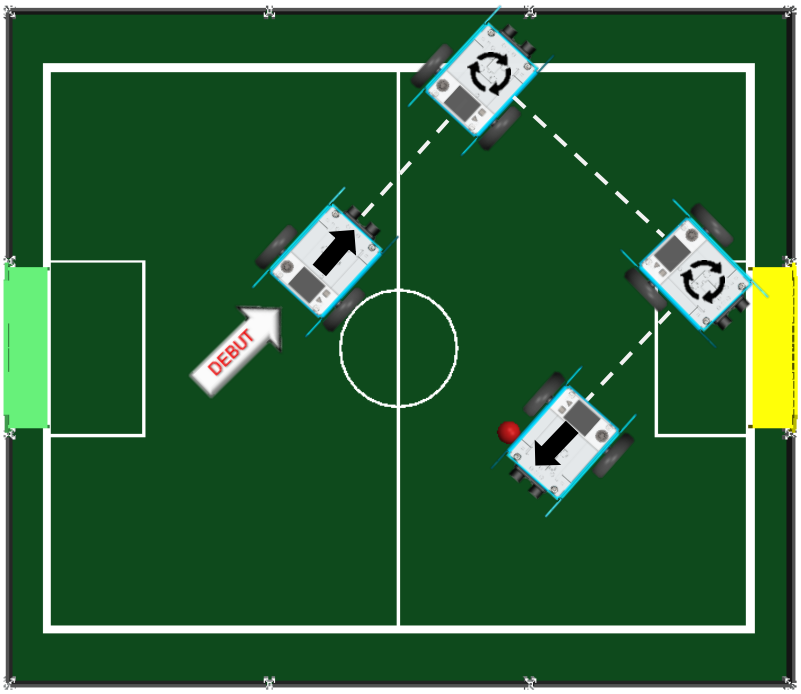
FICHE N°10 : rebondir sur les lignes blanches

But du programme : tourner à droite lorsqu'une ligne blanche est détectée, sinon se déplacer en marche avant à 20 tr/min.

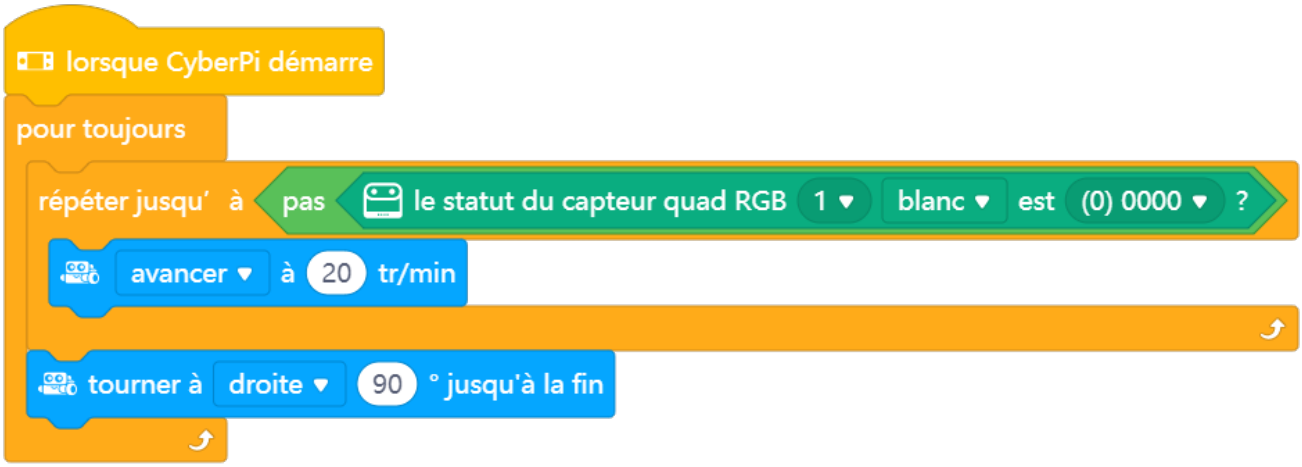
Notion de programmation abordée : répéter un ensemble de blocs plusieurs fois / tester la valeur que retourne un capteur/introduction de la boucle répéter jusqu'à

Capteur utilisé : module capteur de ligne

Synoptique :



Exemple de correction : MB2-DB-F10.mBlock



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

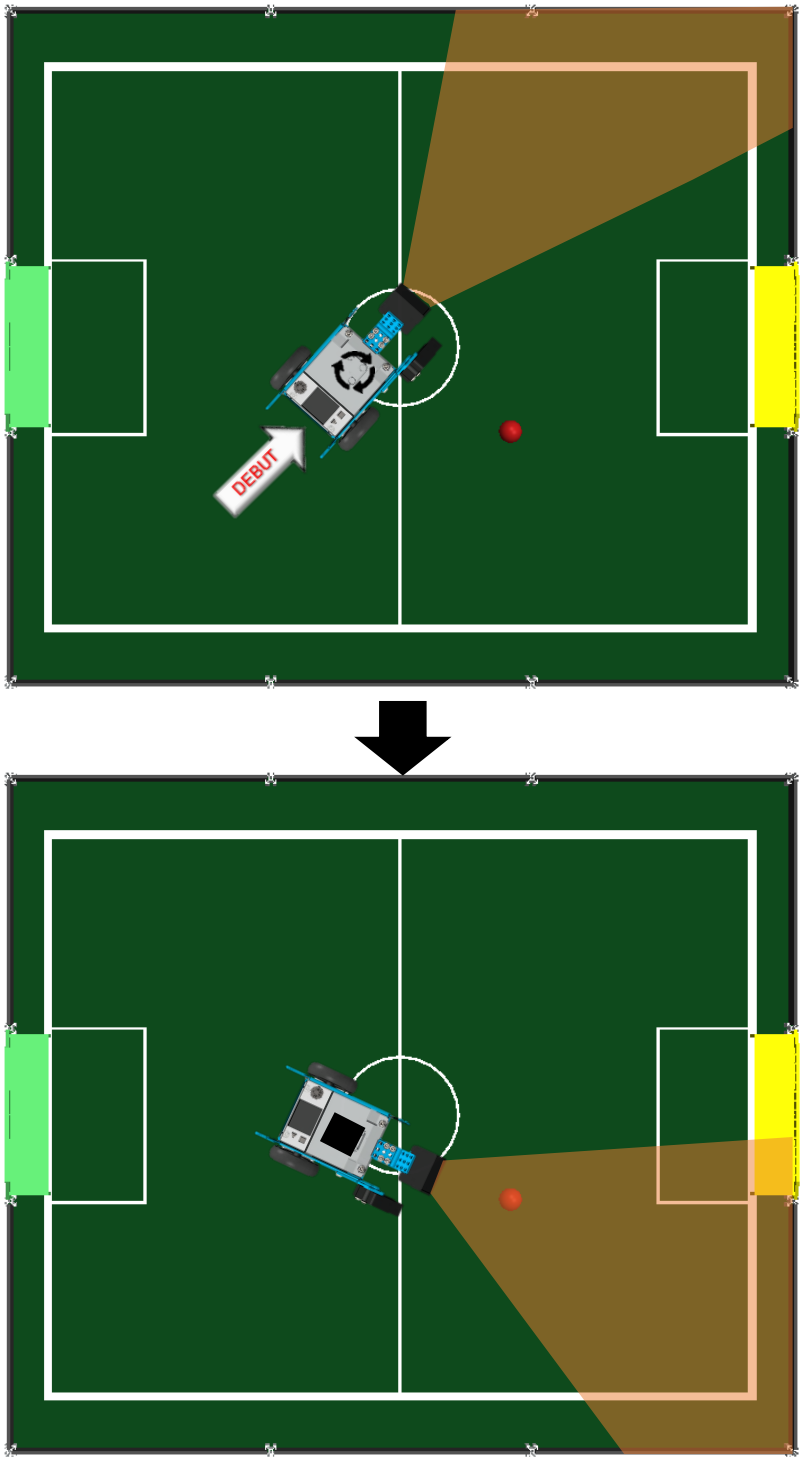
FICHE N°11 : tourner jusqu'à ce que la balle soit détectée

But du programme : tourner à droite à 10 tr/min jusqu'à ce que la balle soit dans le champ de vision de la caméra. Arrêter le robot lorsque la balle est détectée.

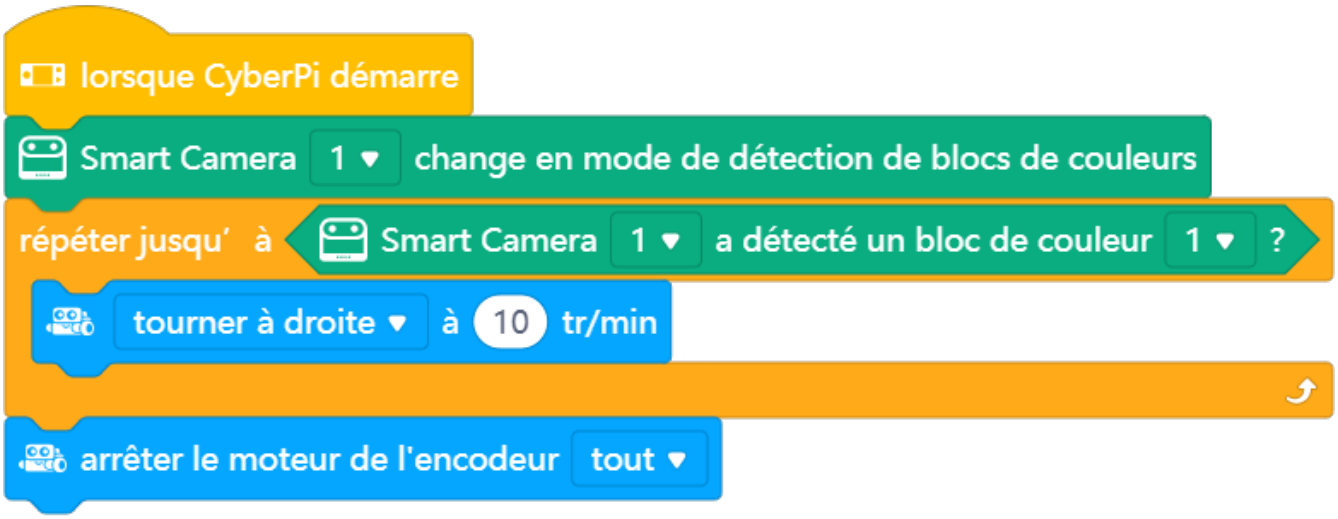
Notion de programmation abordée : séquence d'instructions, boucle, test conditionnel

Capteur utilisé : module Smart Caméra

Synoptique :

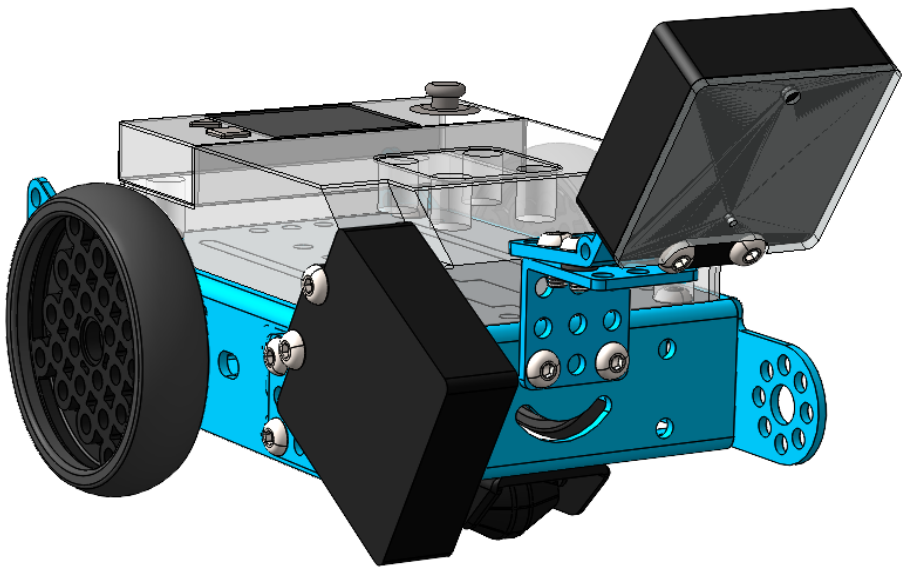


Exemple de correction : MB2-DB-F11.mBlock



Commentaire : à chaque tour de boucle, le robot va tourner à droite à 10 tr/min, tant que la balle rouge n'entre pas dans son champ de vision.

Exemple de configuration du robot :



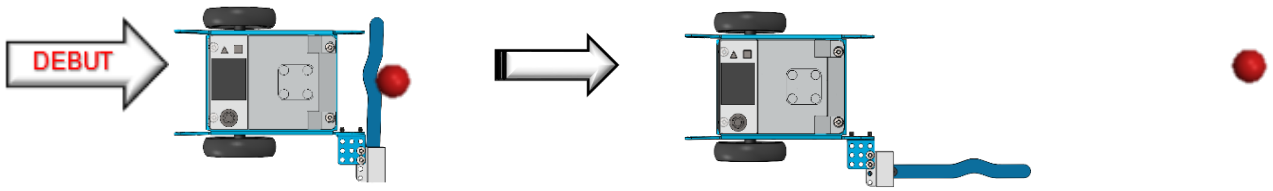
FICHE N°12 : propulser la balle à l'aide d'un lanceur de balle

But du programme : placer la balle devant la spatule, propulser la balle, puis ramener la spatule dans sa position initiale.

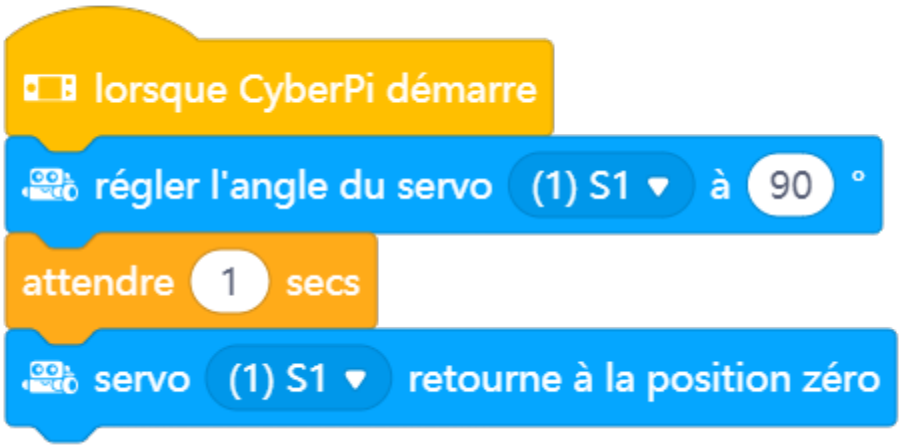
Notion de programmation abordée : prise en main du servomoteur

Actionneur utilisé : servomoteur

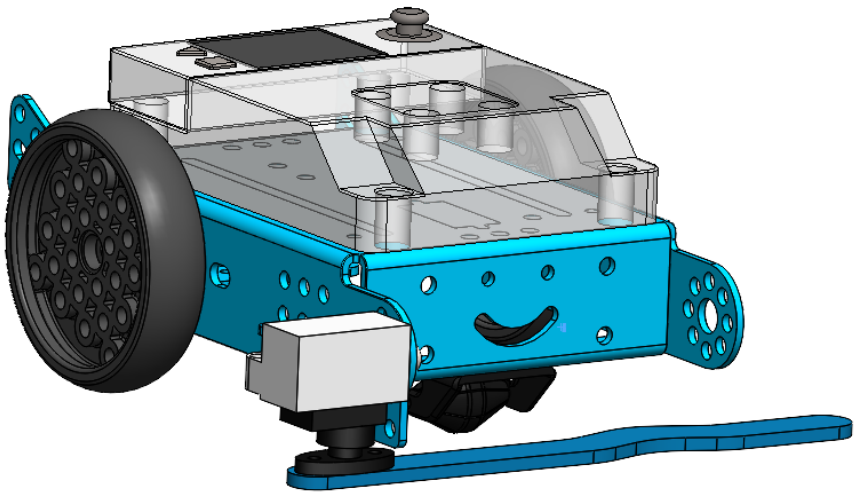
Synoptique :



Exemple de correction : MB2-DB-F12.mBlock



Exemple de configuration du robot :



DéfiBut - Marquer un but de manière autonome

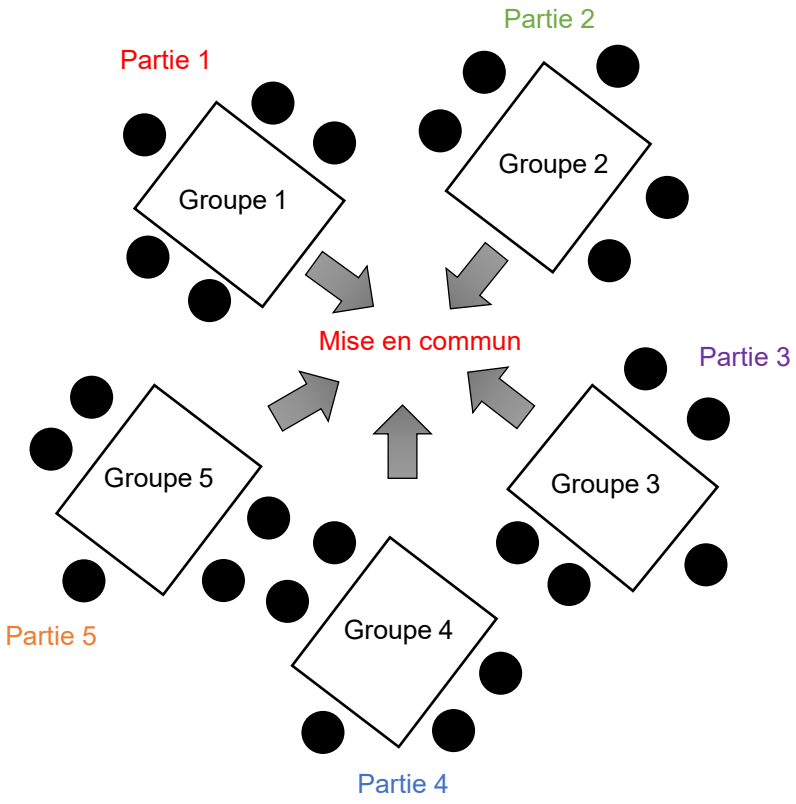
L'algorithme utilisé pour programmer un robot qui marque un but de manière autonome est complexe. Néanmoins, cette problématique peut se décomposer en tâches simples. Les élèves peuvent s'organiser en groupes autour de ce projet commun en répartissant les différentes tâches (programmation, design, conception 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.

Nom du fichier	Description	Objectif
MB2-DB-P1	Localiser la balle	Exploiter les informations provenant de la Smart Camera
MB2-DB-P2	Avancer jusqu'à la balle	Exploiter les informations provenant de la Smart Camera
MB2-DB-P3	Récupérer la balle	Exploiter les informations provenant du télémètre à ultrasons
MB2-DB-P4	Localiser le but adverse	Exploiter les informations provenant de la Smart Camera
MB2-DB-P5	Tirer en direction du but	Exploiter les informations provenant de la Smart Camera
MB2-DB-PF	Programme final	Combiner les programmes précédents en un seul programme

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



Partie 1 : localiser la balle

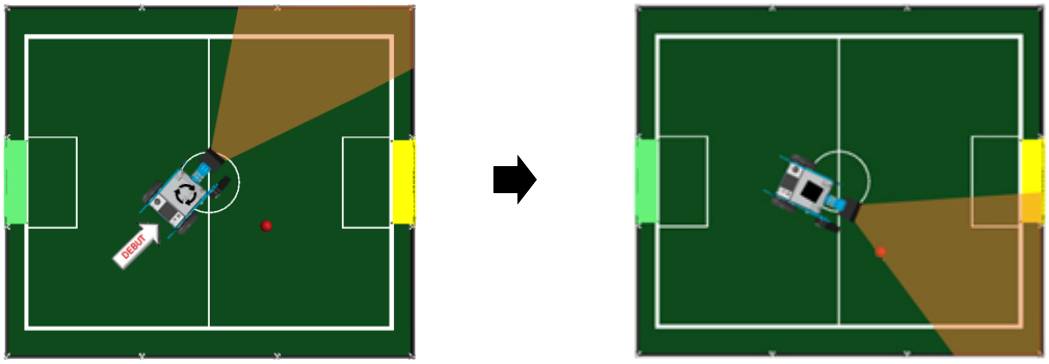
But du programme : déplacer le robot jusqu'à ce que la balle soit détectée.

Situation de départ : la balle et le robot sont positionnés n'importe où sur le terrain

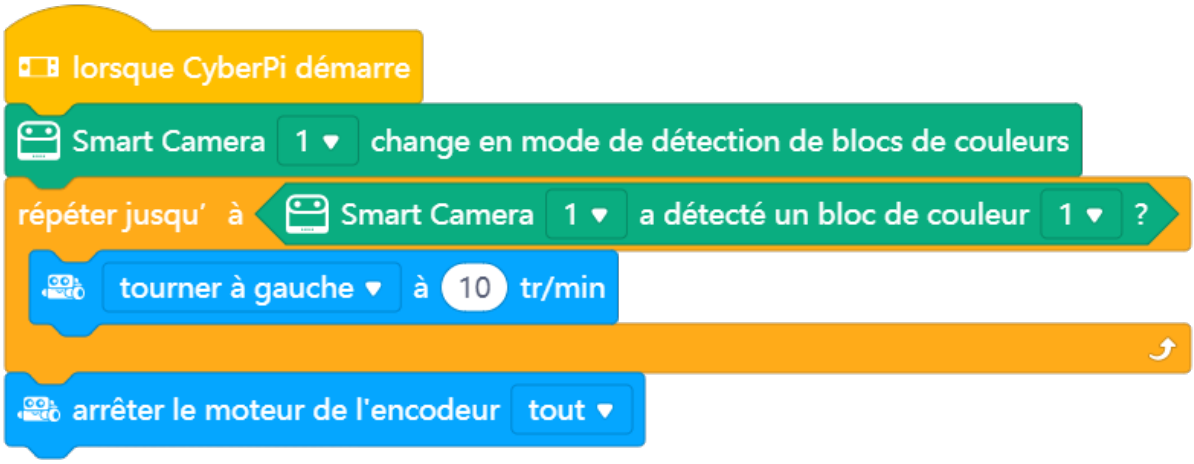
Situation d'arrivée : le robot est à l'arrêt, l'avant du robot pointe en direction de la balle

Capteurs mis en œuvre : Smart Camera préalablement étalonnée pour détecter la signature de la balle (rouge)

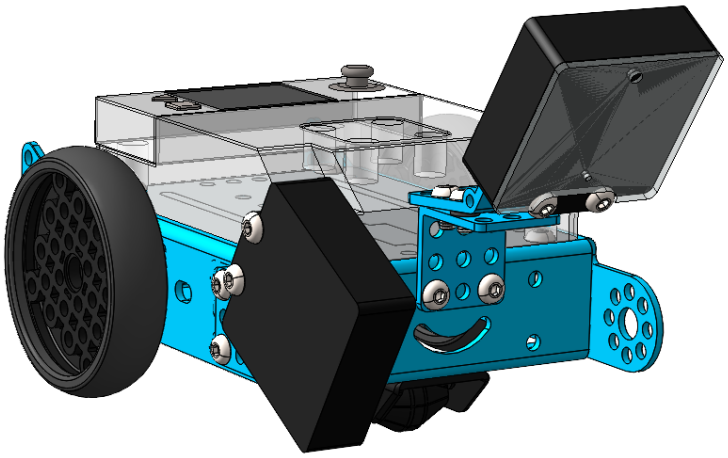
Synoptique :



Exemple de correction : MB2-DB-P1.mBlock



Exemple de configuration du robot :



ATTENTION : l'étalonnage préalable de la Smart Camera est impératif (**voir ANNEXE**).

Partie 2 : avancer jusqu'à la balle

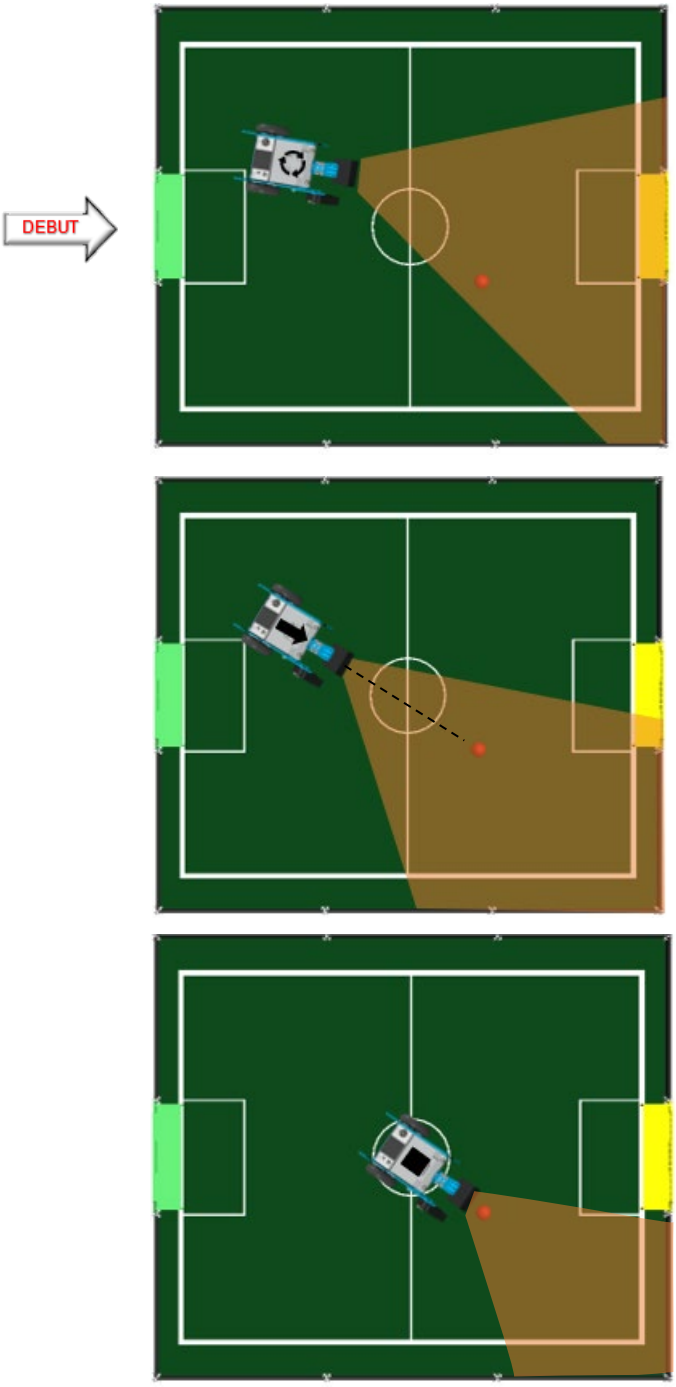
But du programme : se rapprocher de la balle

Situation de départ : le robot est à l'arrêt, l'avant du robot pointe en direction de la balle

Situation d'arrivée : l'avant du robot est orienté vers la balle et est proche de la balle

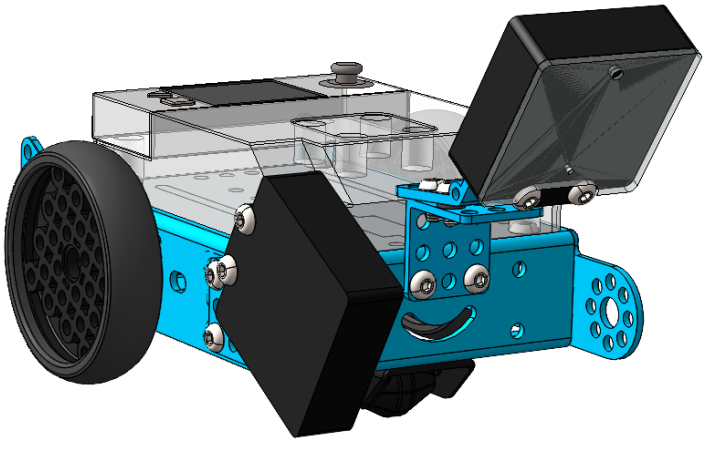
Capteurs mis en œuvre : Smart Camera préalablement étalonnée pour détecter la signature de la balle (rouge)

Synoptique :



Exemple de correction : MB2-DB-P2.mBlock

Exemple de configuration du robot :



ATTENTION : l'étalonnage préalable de la Smart Camera est impératif (**voir ANNEXE**).

Partie 3 : récupérer la balle

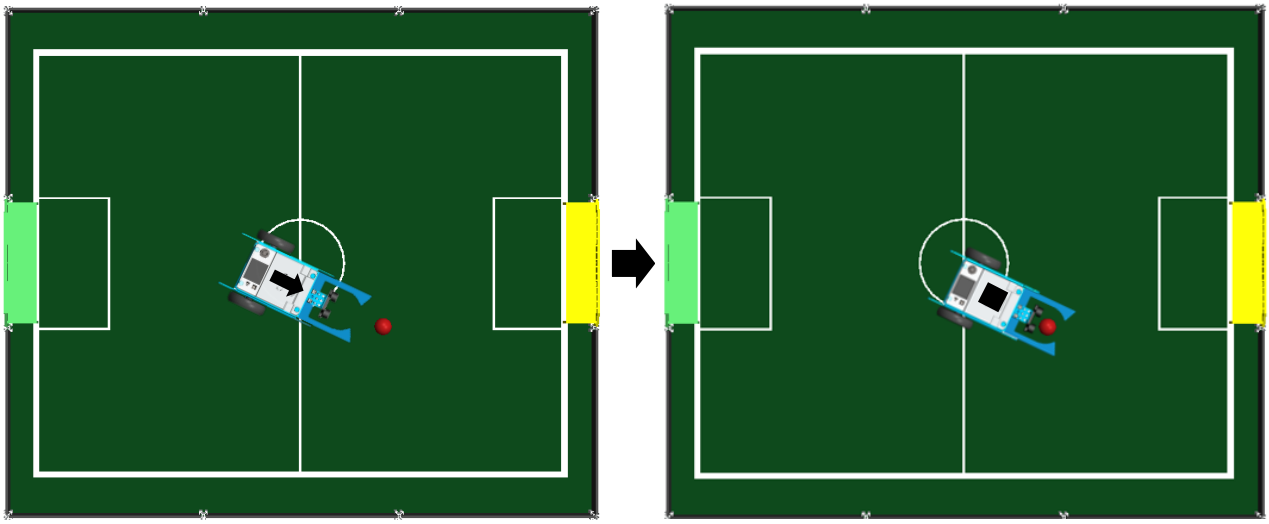
But du programme : se rapprocher au plus près de la balle

Situation de départ : l'avant du robot est orienté vers la balle et est proche de la balle

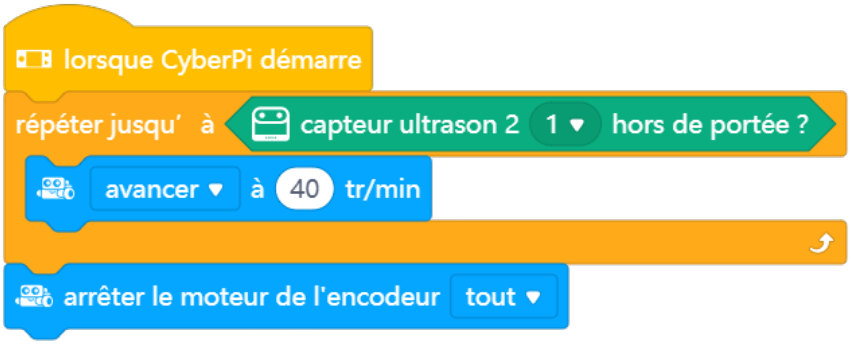
Situation d'arrivée : la balle est emprisonnée dans le récupérateur de balle, au plus près du robot

Capteurs mis en œuvre : télémètre à ultrasons

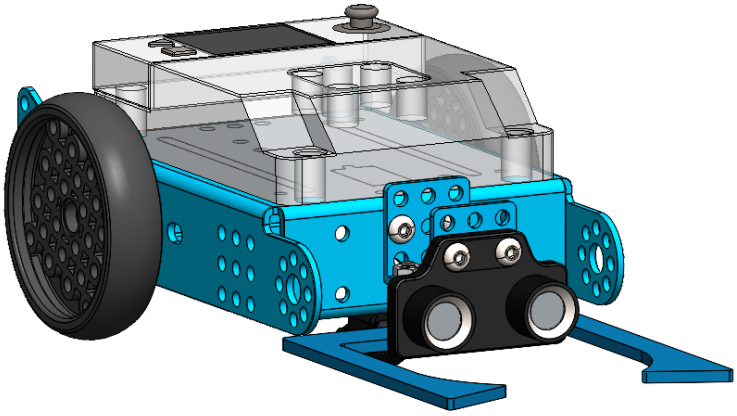
Synoptique :



Exemple de correction : MB2-DB-P3.mBlock



Exemple de configuration du robot :



La forme du récupérateur de balle peut être optimisée pour faciliter la récupération de la balle et pour la conserver à proximité du robot lorsque celui-ci se déplace.

Partie 4 : localiser le but adverse

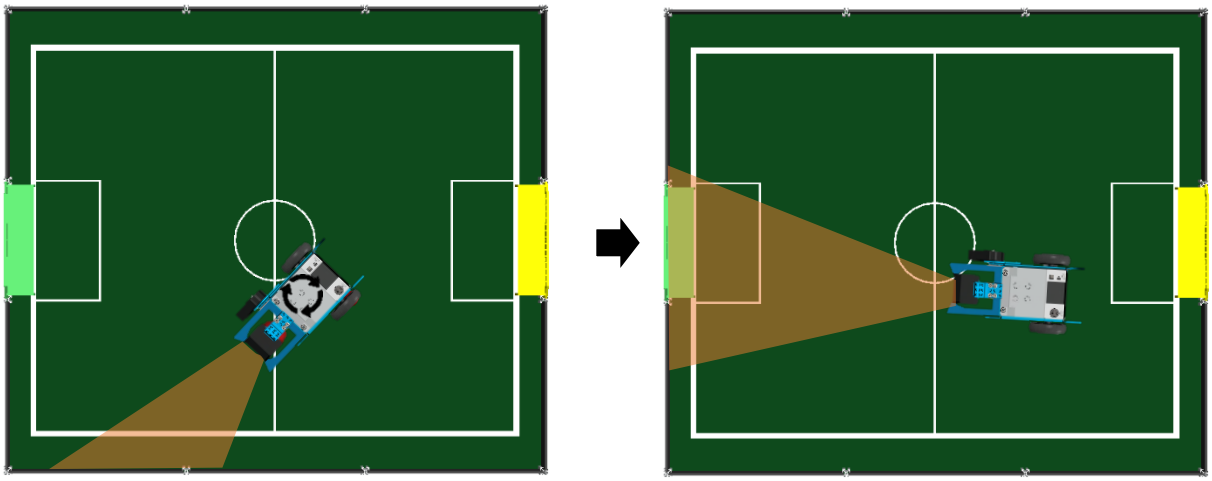
But du programme : déplacer le robot jusqu'à ce que le but adverse soit détecté

Situation de départ : la balle est emprisonnée dans le récupérateur de balle, au plus près du robot

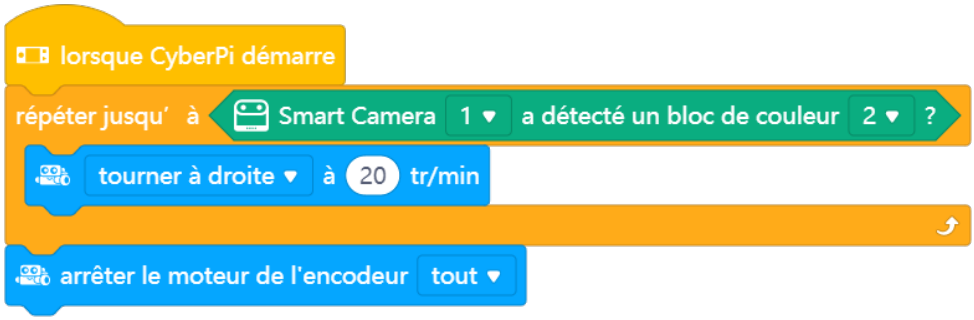
Situation d'arrivée : l'avant du robot est orienté vers le but adverse avec la balle est emprisonnée dans le récupérateur de balle, au plus près du robot

Capteurs mis en œuvre : Smart Camera préalablement étalonnée pour détecter la signature de la balle (rouge) et de chaque but (vert et jaune)

Synoptique :

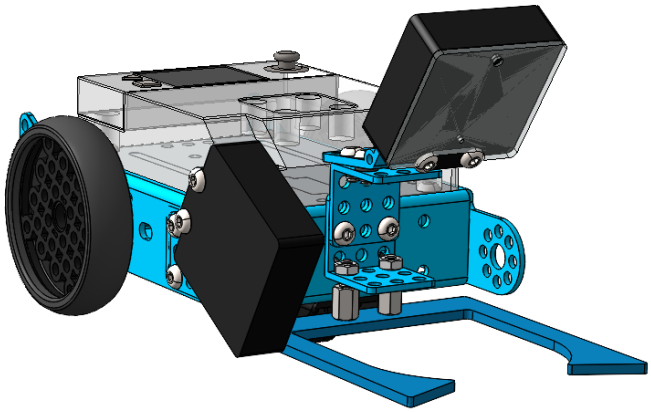


Exemple de correction : MB2-DB-P4.mBlock



Note : la couleur N°2 détectée par la Smart Camera correspond à la signature du but considéré comme adverse (but vert sur le synoptique).

Exemple de configuration du robot :



ATTENTION : l'étalonnage préalable de la Smart Camera est impératif (**voir ANNEXE**).

Partie 5 : tirer en direction du but

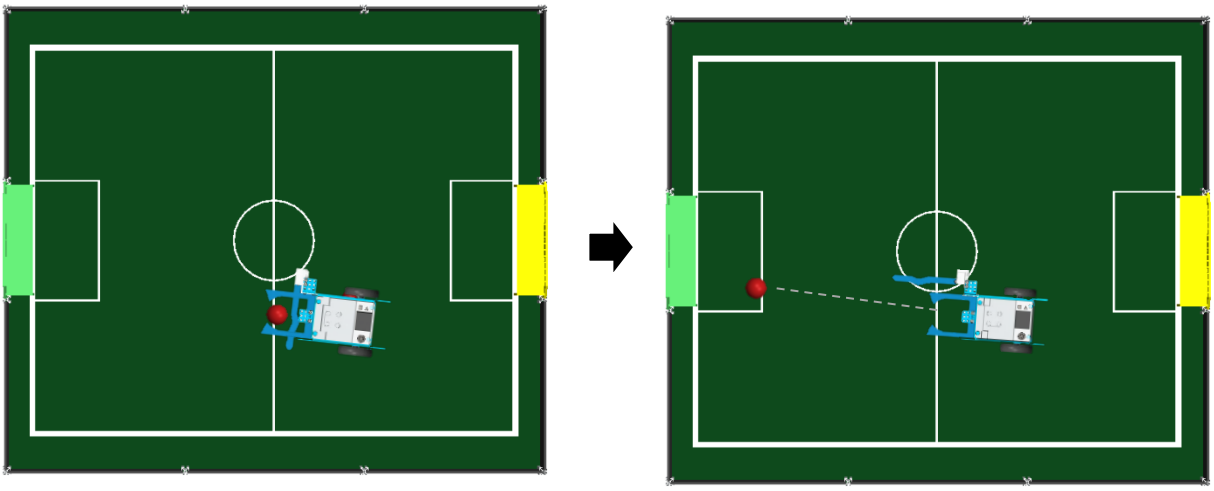
But du programme : déclencher le tir en direction du but

Situation de départ : l'avant du robot est orienté vers le but adverse avec la balle est emprisonnée dans le récupérateur de balle, au plus près du robot

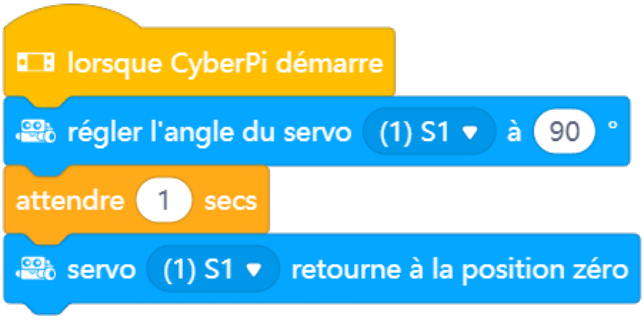
Situation d'arrivée : le lanceur de balle est déclenché, la balle quitte le récupérateur de balle en direction du but

Capteurs / actionneur mis en œuvre : Smart Camera préalablement étalonnée pour détecter la signature de la balle (rouge) et de chaque but (vert et jaune), servomoteur.

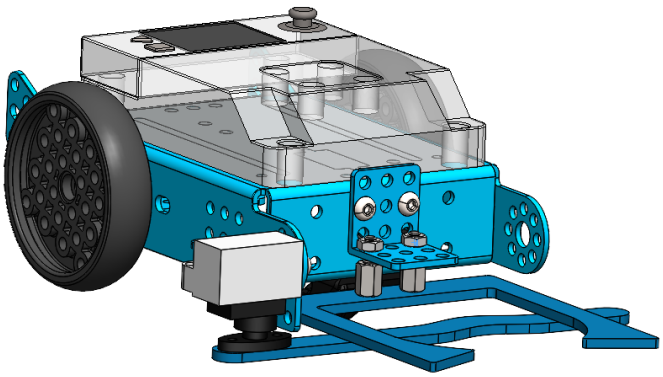
Synoptique :



Exemple de correction : MB2-DB-P5.mBlock



Exemple de configuration du robot :



ATTENTION : l'étalonnage préalable de la Smart Camera est impératif (**voir ANNEXE**).

Partie 6 : programme final

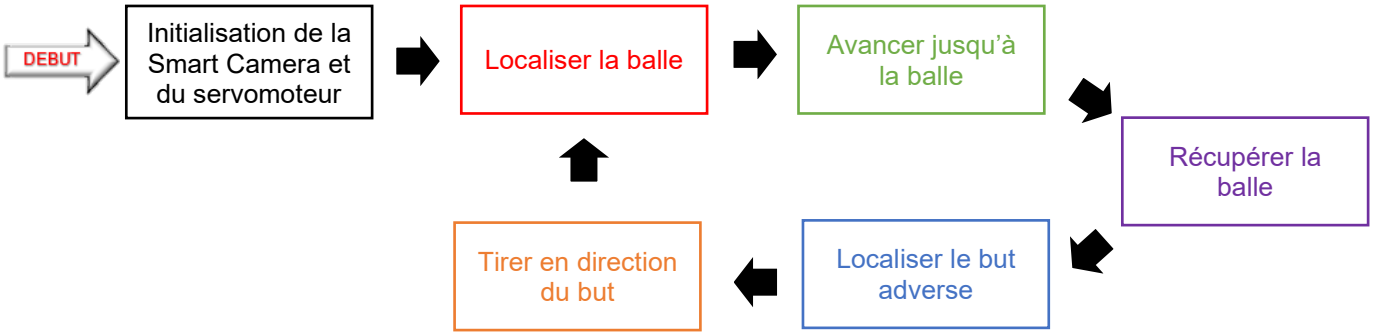
But du programme : le robot marque un but de manière autonome

Situation de départ : la balle et le robot sont positionnés n'importe où sur le terrain

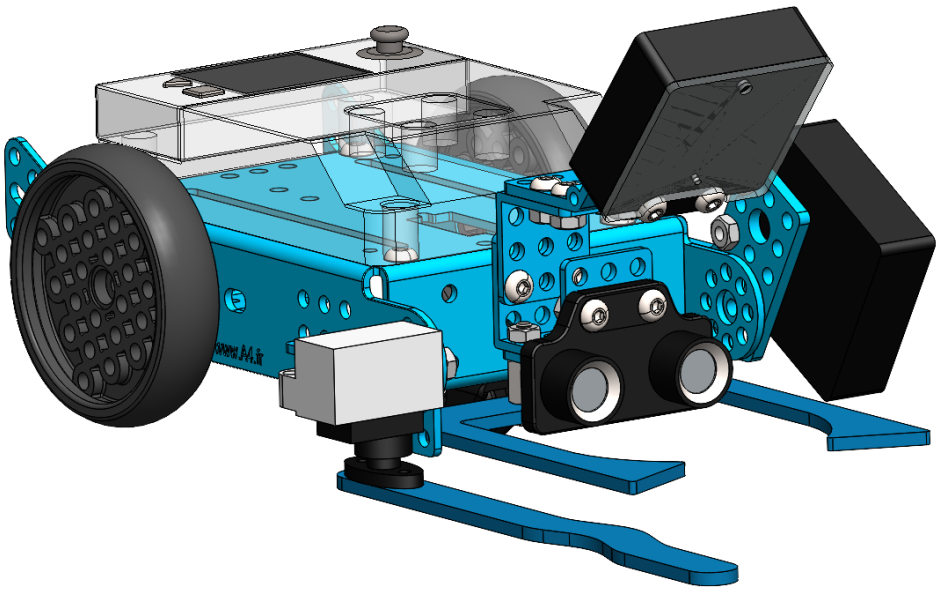
Situation d'arrivée : la balle est dans le but adverse

Capteurs / actionneur mis en œuvre : Smart Camera préalablement étalonnée pour détecter la signature de la balle (rouge) et de chaque but (vert et jaune), télémètre à ultrasons, servomoteur

Synoptique :

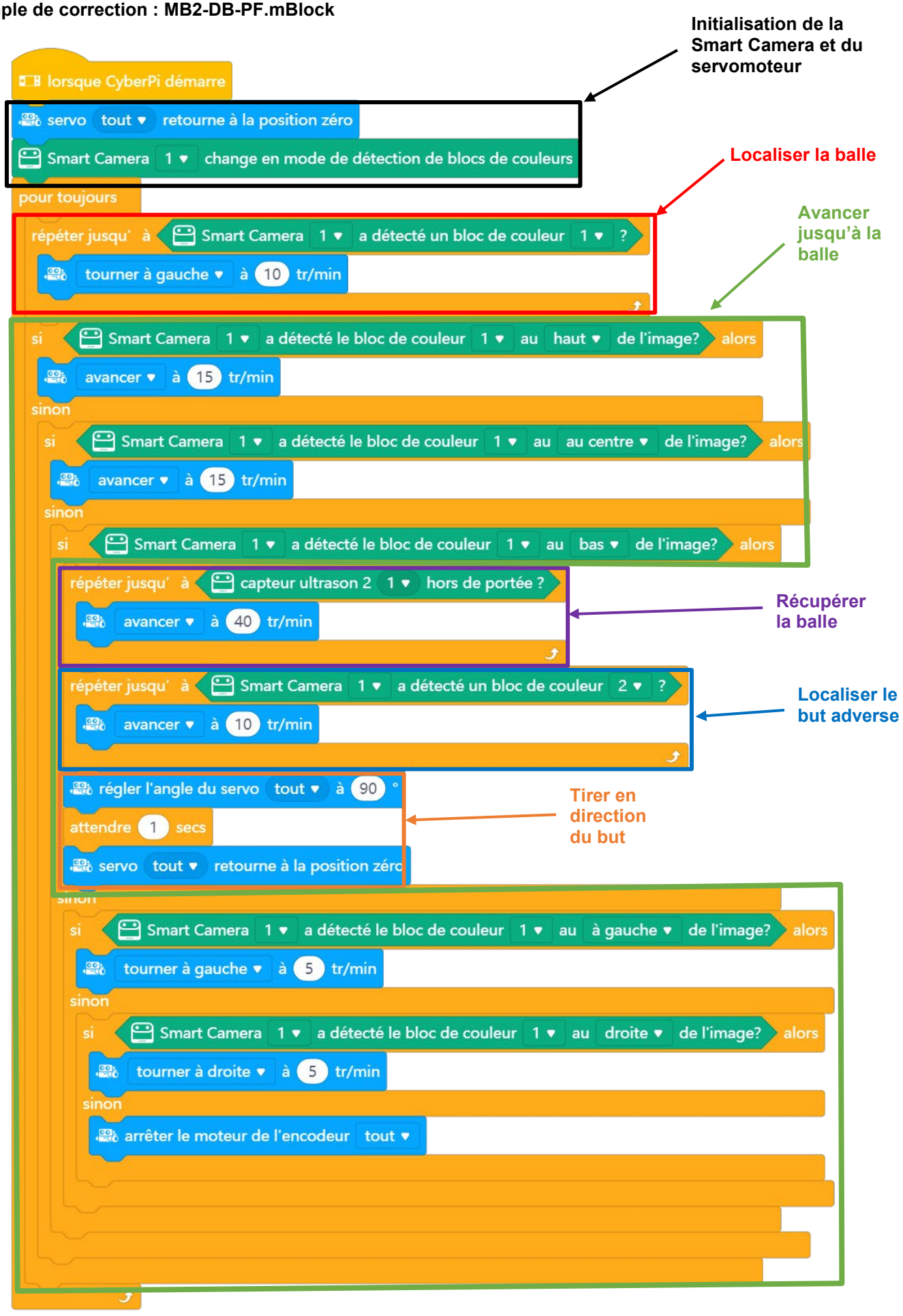


Exemple de configuration du robot :



ATTENTION : l'étalonnage préalable de la Smart Camera est impératif (**voir ANNEXE**).

Exemple de correction : MB2-DB-PF.mBlock



Activités complémentaires

Optimisation du robot et du programme final

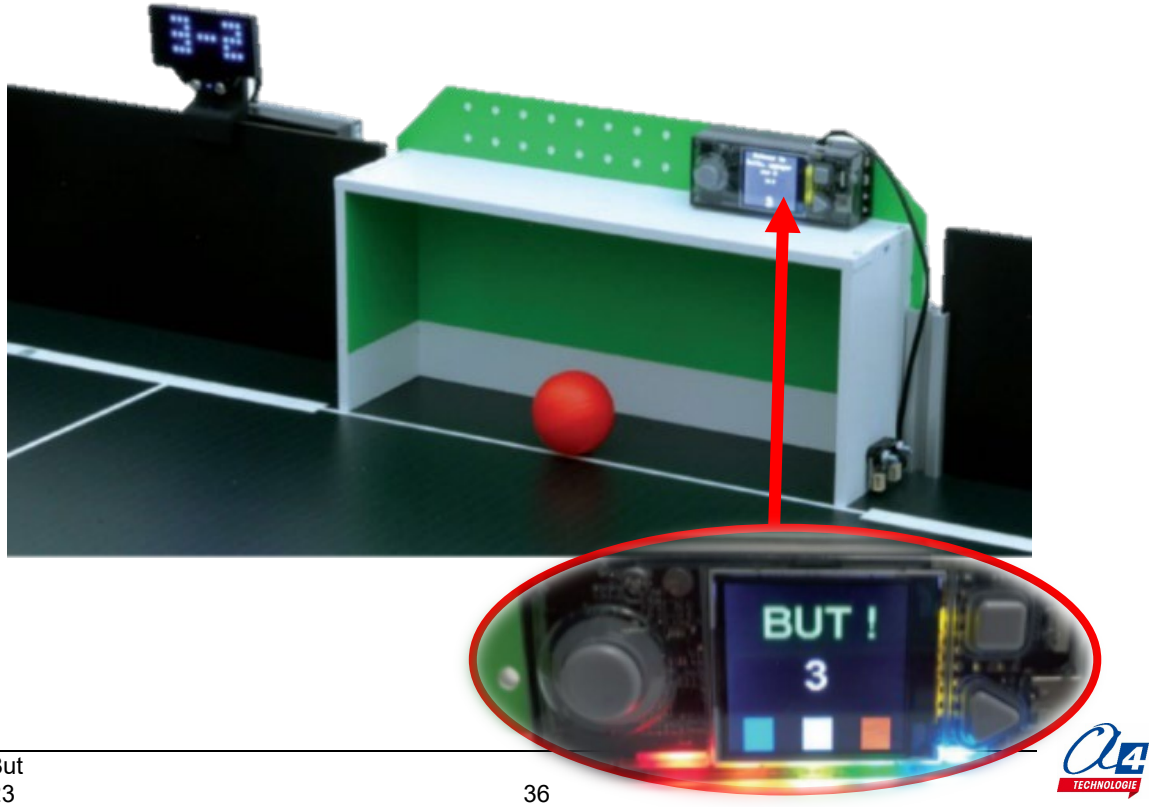
Le programme final proposé dans le chapitre précédent peut être amélioré. Le robot peut également être amélioré.

Voici quelques suggestions d'améliorations :

- Prise en compte des limites du terrain pour que le robot ne le quitte pas lorsqu'il se déplace pour localiser la balle ou le but adverse (utilisation du module de détection de ligne ou/et du télémètre à ultrasons)
- Modification du récupérateur / lanceur de balle pour mieux contrôler l'orientation du tir
- Afficher des messages pour indiquer l'action en cours d'exécution (utile pour surveiller et s'assurer du bon fonctionnement du programme)
- Utiliser la barre de LED du module CyberPi pour afficher la couleur détectée par la caméra ou le capteur de ligne
- Améliorer l'algorithme de localisation de la balle et du but adverse
- Augmenter la vitesse de déplacement du robot tout en conservant les performances maximales des capteurs pour détecter la balle et le but adverse
- Gérer les cas particuliers (balle à proximité d'un coin du terrain, balle très proche du but et compliquée à récupérer …)
- Détecter un robot adverse pour l'éviter

Comptage de buts / Affichage du score

Des points de fixations et un orifice sont prévus sur le côté des buts afin d'intégrer un système permettant de compter les buts. La détection de but marqué peut se faire en intégrant un capteur de distance ou une barrière optique positionnés près du sol sur le côté des cages.
L'affichage du score peut être fait avec un module CyberPi alimenté par son Pocket Shield (batterie) ou tout autre module électronique programmable.
On peut surveiller la variation de distance mesurée par un télémètre ou la rupture d'une barrière IR pour incrémenter un compteur de score, déclencher une animation lumineuse ou sonore sur chaque cage.
Un autre module CyberPi alimenté par son Pocket Shield (batterie) peut servir d'affichage général en communiquant avec le module CyberPi équipant chaque cage.



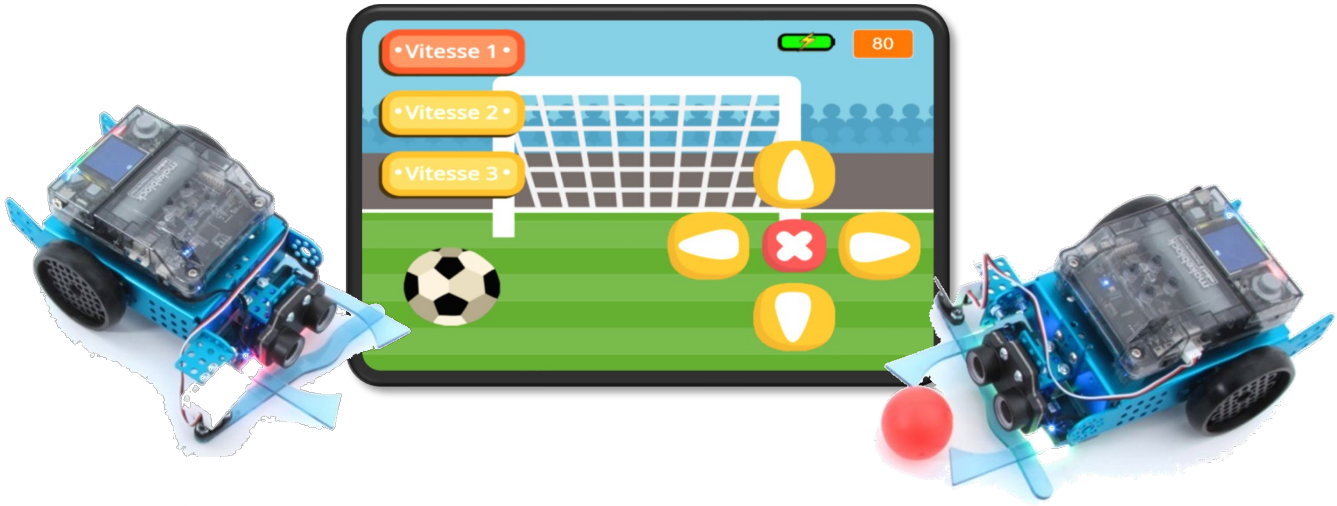
Améliorations du récupérateur / lanceur de balle

Différentes formes peuvent être envisagées pour optimiser les pièces mécaniques permettant de récupérer et de lancer la balle avec plus de précision. Ces pièces peuvent être prototypées facilement dans un premier temps avec du carton puis être réalisées à l'aide de machines de fabrication numérique (découpe laser, imprimante 3D, fraiseuse).

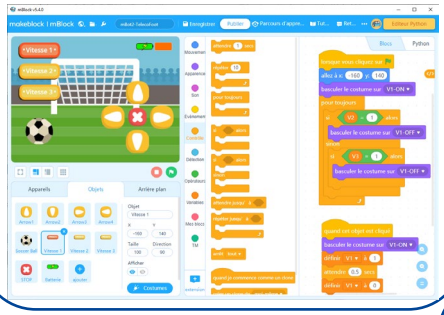
Pilotage de robots avec une tablette

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.

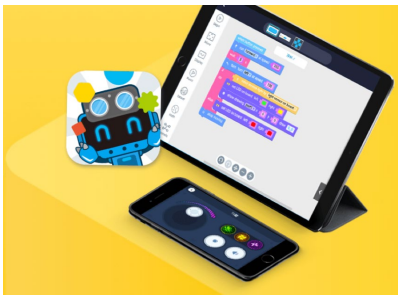
Exemple de programme : MB2-TelecoFoot.mblock



mBlock 5

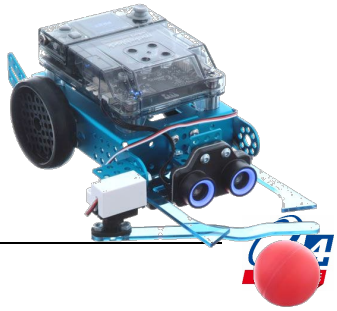


mBlock mobile app

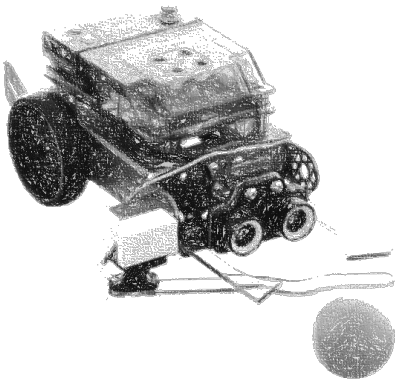


Bluetooth

mBot2



Idée

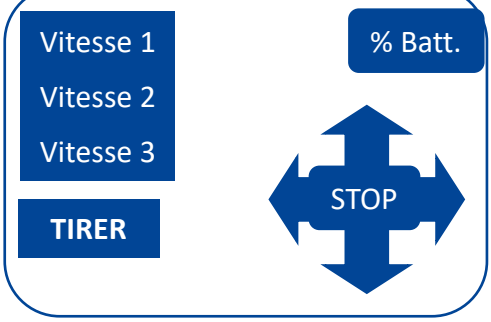


Cahier des charges

Besoin

- Piloter mBot2 :
  4 directions + Stop
- Choisir vitesse
  déplacement 1, 2, 3
- Déclencher tir de la balle
- Afficher niveau de batterie
- etc.

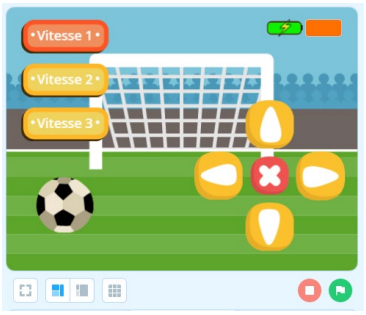
Interface utilisateur



Design

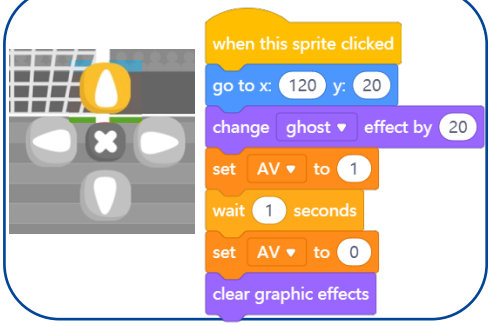
Objets

IHM

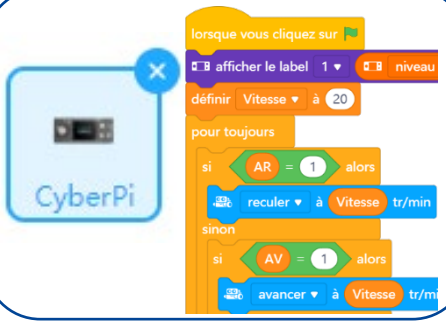


Programmation

Objets

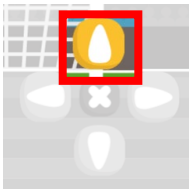


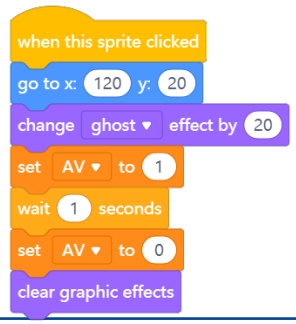
Matériel

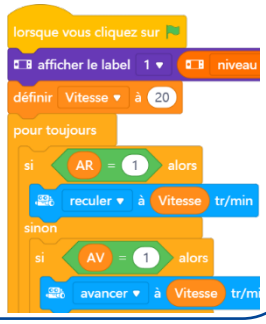


Tests unitaires

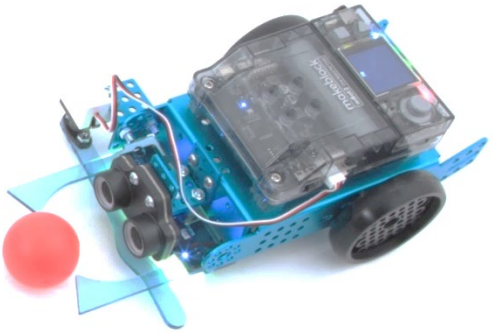
IHM / matériel

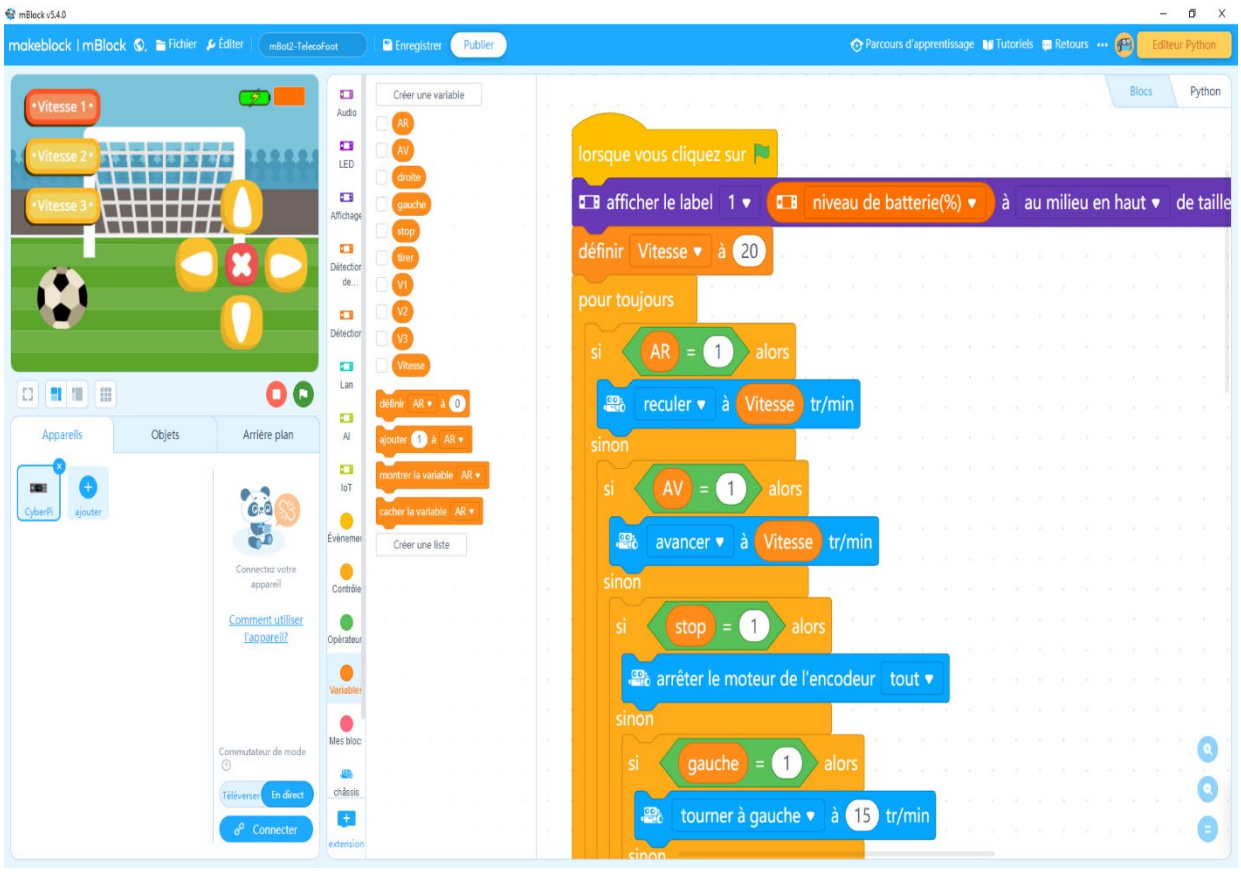




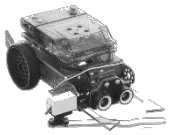


Test global

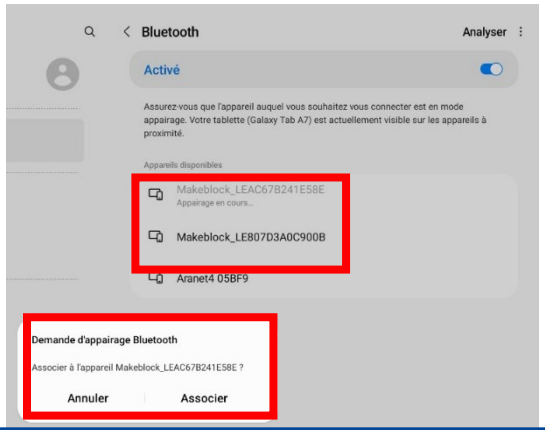




1 Afficher l'identifiant de chaque mBot2

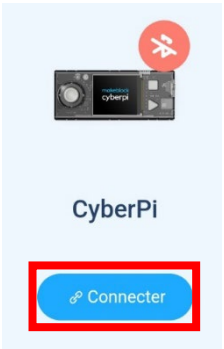


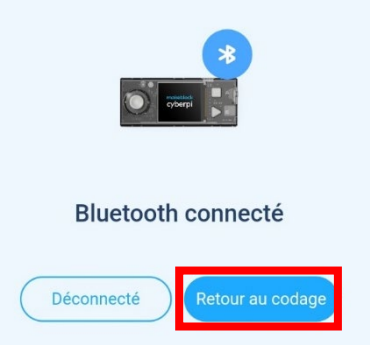
2 Associer chaque tablette à chaque mBot



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

4





Pilotage du robot avec la Manette Bluetooth Controller

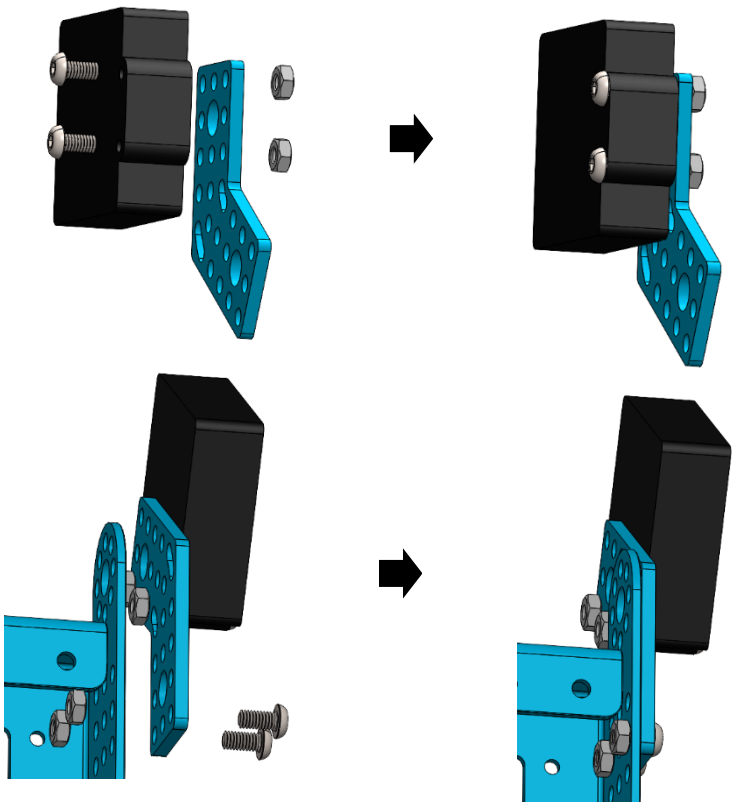
La manette Bluetooth Controller permet de déclencher des actions à distance.
Allumer la manette Bluetooth Controller et le robot.
Effectuer l'appairage avec le robot en appuyant sur le bouton marqué avec le symbole « Bluetooth » sur la manette (bouton à gauche du bouton « + »).
Lorsque l'appairage est fait la lumière bleue entre les 2 joysticks est fixe et le robot émet un son.

Mettre en service l'extension « Bluetooth Controller ». Le programme suivant permet de contrôler les déplacements du robot avec le bouton à 4 flèches de la manette. Les deux boutons larges sur la tranche de la manette permettent de déclencher le tir avec un servomoteur connecté (dans le bon sens) sur le port P1 du robot.

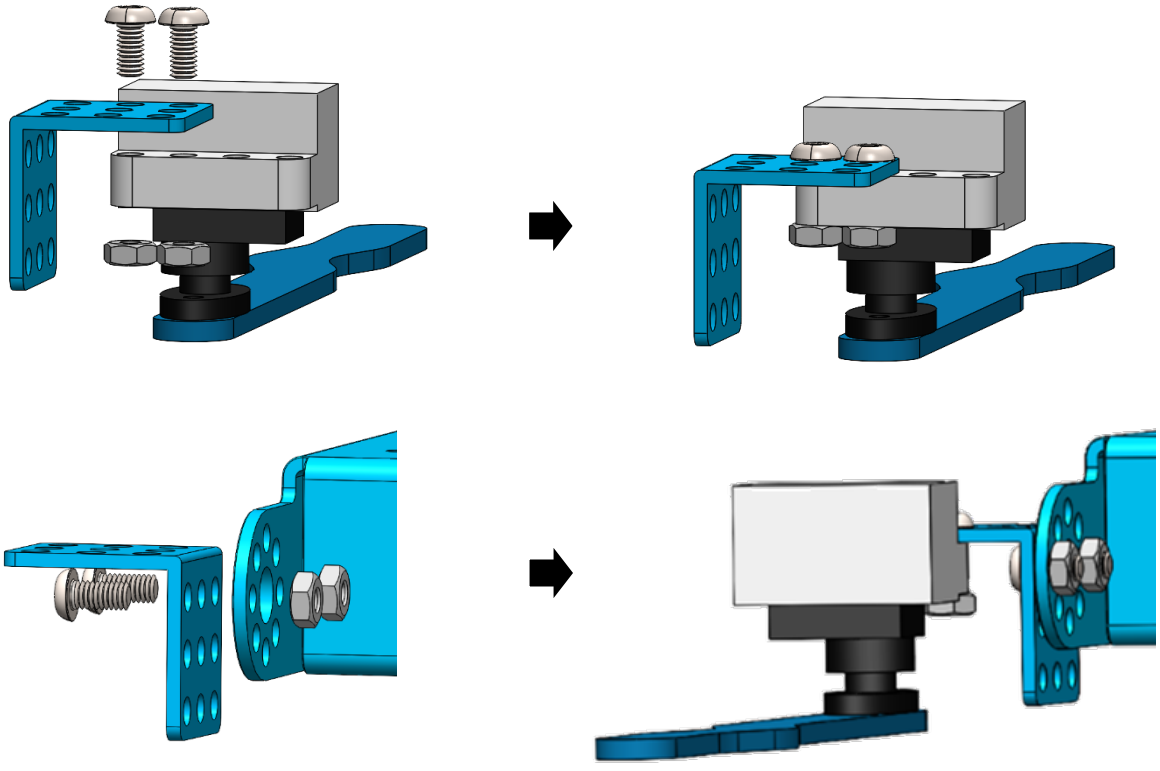
Exemple de programme : MB2-TelecoBluetoothController.mblock

Montage du kit d'extension joueur de foot

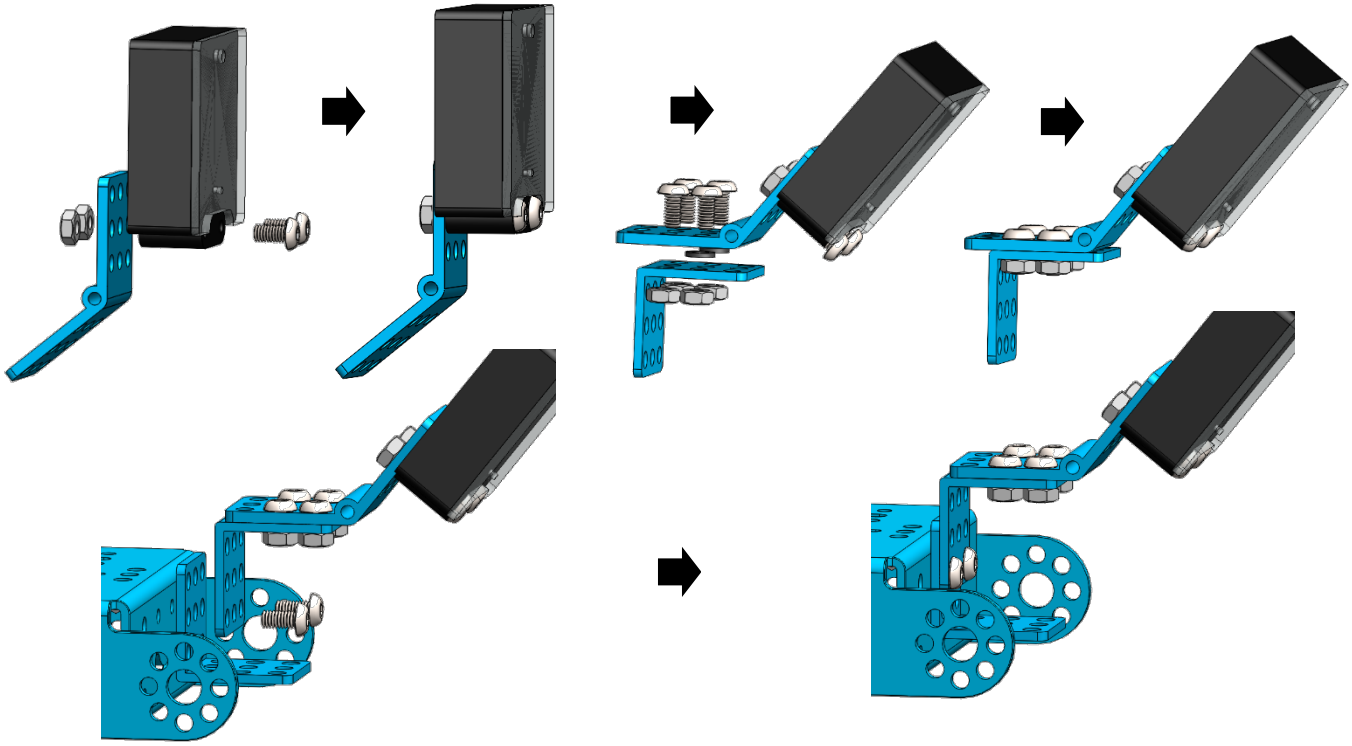
Montage de la batterie



Montage du servomoteur et du lanceur de balle

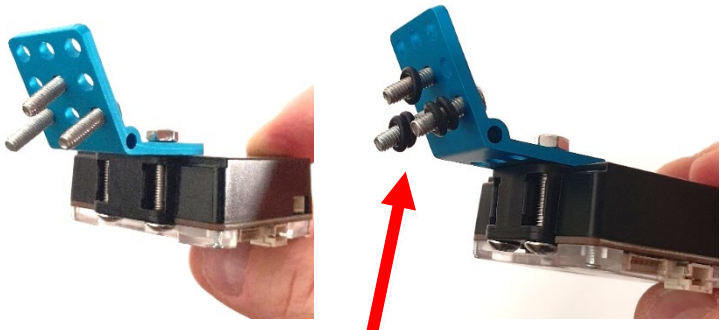


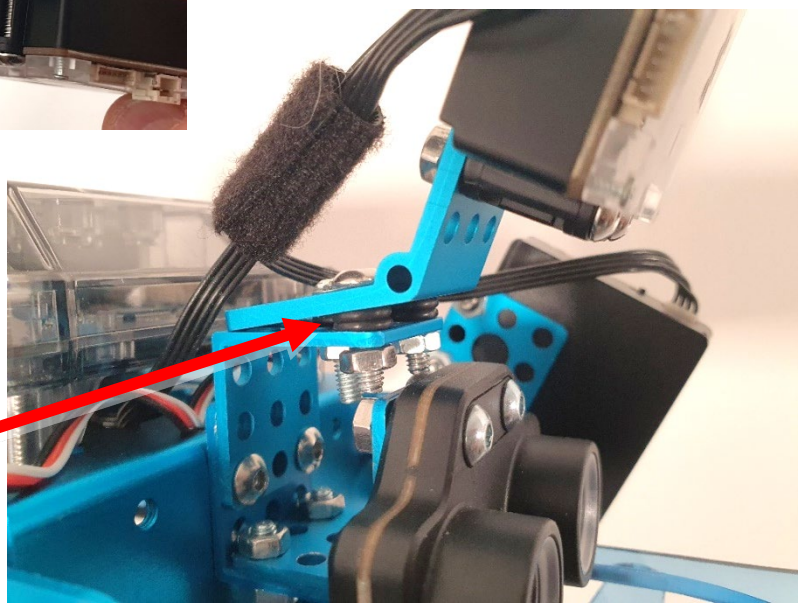
Montage de la Smart Camera



Réglage et optimisation de l'angle de détection de la Smart Camera

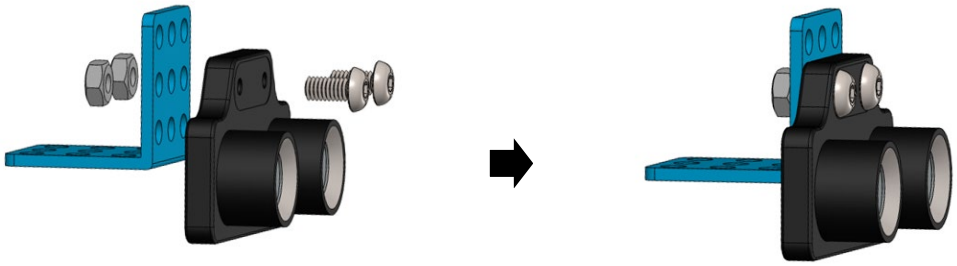
Des joints toriques peuvent être insérés sous le support de fixation à 45° de la Smart Camera.



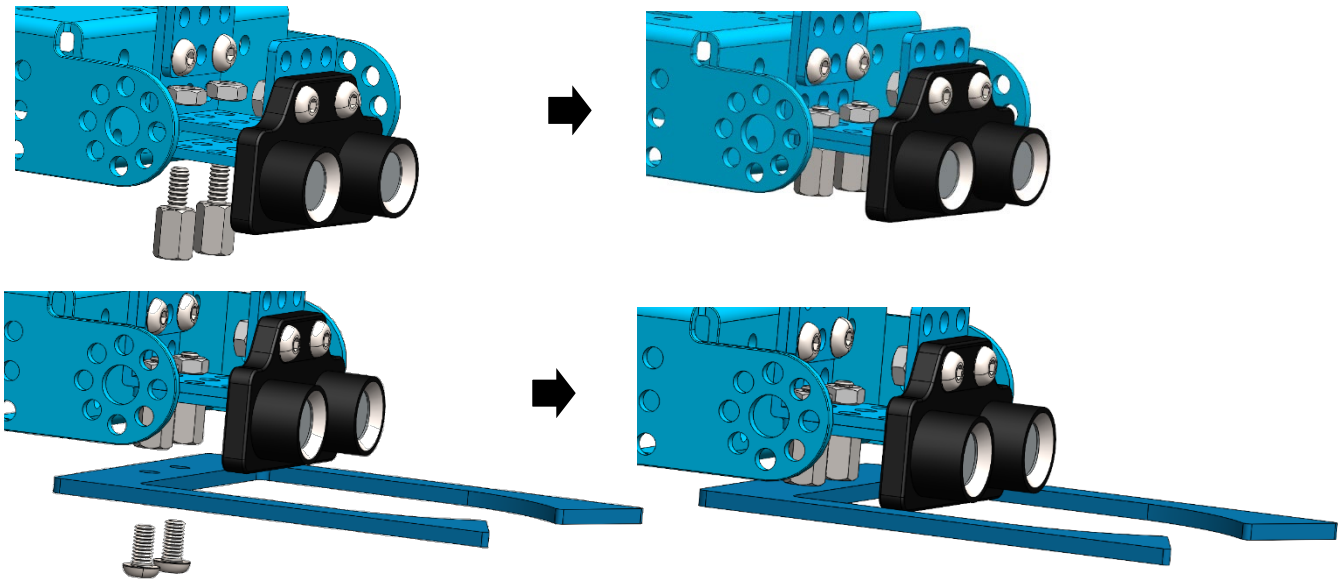


L'empilement de joints toriques entre l'équerre à 45° et l'équerre à 90° permet d'ajuster l'angle de visé de la caméra en serrant plus ou moins fort les vis de fixations

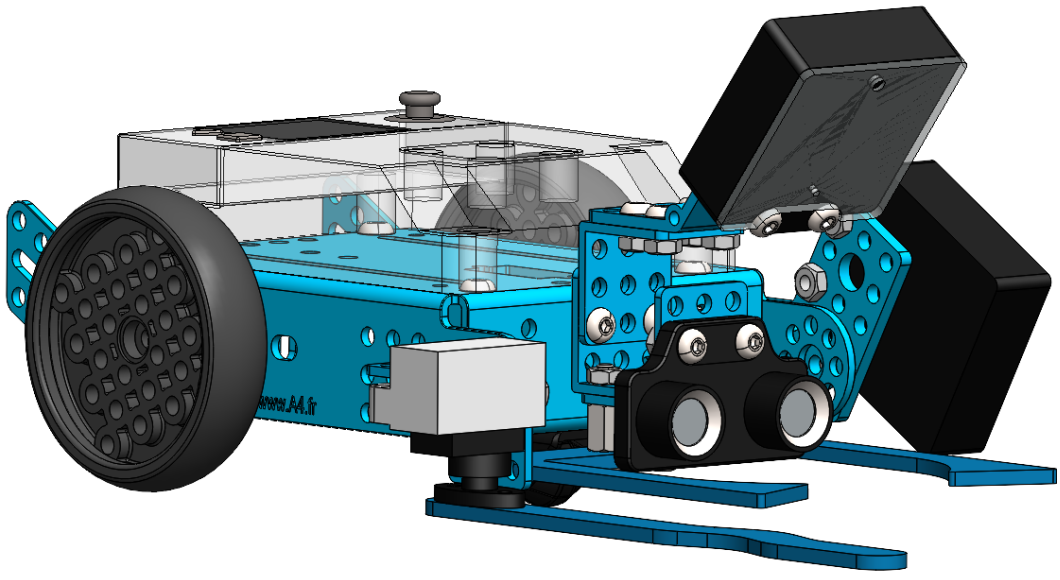
Montage du capteur de distance



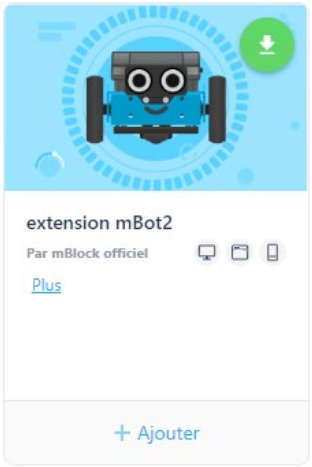
Montage de la raquette



Montage complet



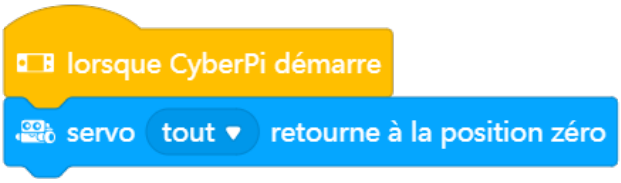
Réglage du lanceur de balle



1- Identification de la position zéro

Ajouter l'extension mBot2 dans mBlock 5.
Brancher le servomoteur sur le port S1 (sur le côté de la carte de pilotage du robot), puis allumer le robot.

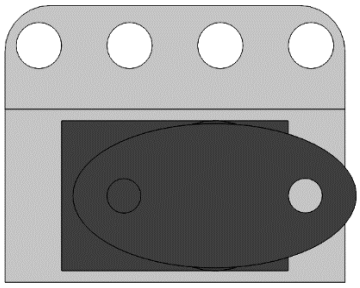
Charger le programme suivant afin de positionner le servomoteur à la position zéro

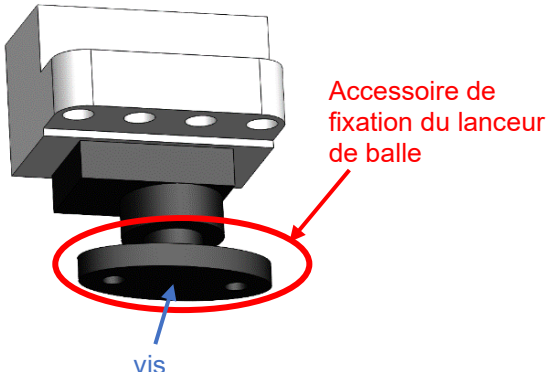


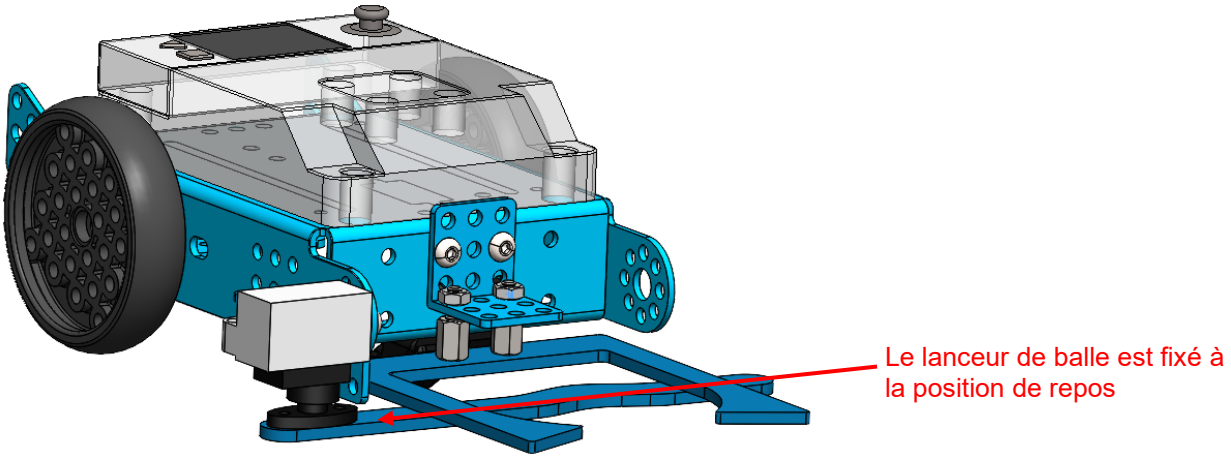
Lancer le programme pour initialiser la position zéro du servomoteur

2- Réglage de la position de repos du lanceur de balle

Positionner puis visser l'accessoire de fixation du lanceur de balle sur l'axe du servomoteur afin de l'orienter dans la position de repos souhaitée avant de fixer le lanceur de balle.







ANNEXE

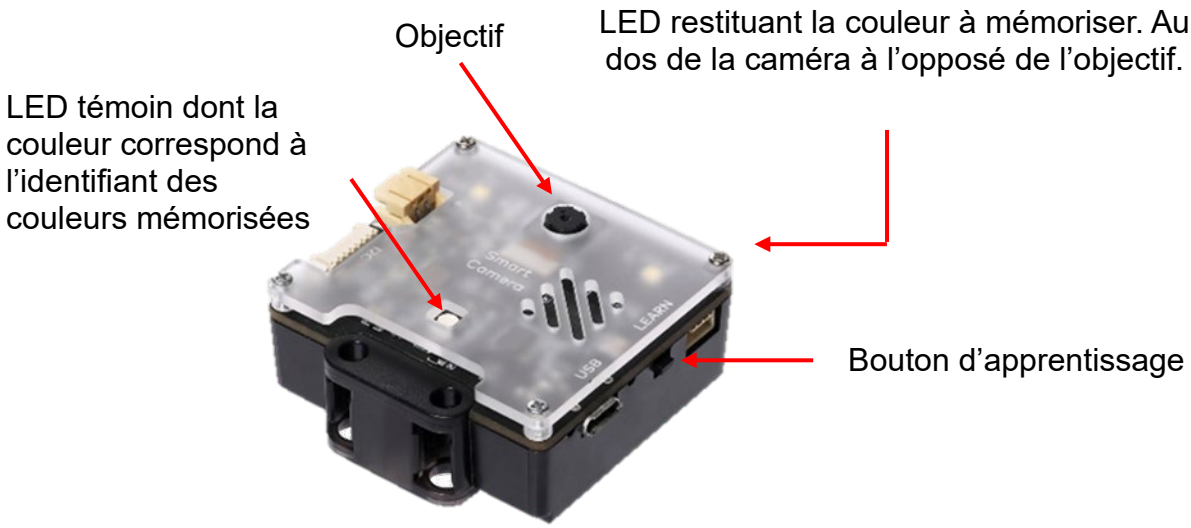
Régler et maîtriser le fonctionnement de la Smart Camera

Il est nécessaire de mémoriser les signatures des différents objets à détecter. Il est fortement recommandé de procéder à l'enregistrement de signature à chaque fois que l'environnement d'utilisation de la Smart Camera évolue (changement de lumière ambiante, reflets parasites, etc.). L'apprentissage de signatures à détecter peut se faire manuellement ou à l'aide du logiciel Piximon V2.

Il est fortement recommandé d'utiliser le logiciel Pixymon V2 pour maîtriser parfaitement les réglages de la Smart Camera (retour à l'écran de ce que la caméra voit) et optimiser au maximum la détection de différentes signatures.

Réglage manuel de la caméra

La Smart Camera dispose d'un bouton d'apprentissage, d'une LED témoin qui restitue la couleur à mémoriser, d'une LED dont la couleur correspond à un identifiant (numéro de 1 à 7) attribué aux couleurs mémorisées.



1 - Sélectionner l'identifiant de la couleur à mémoriser
Mettre sous tension la caméra.
Appuyer longuement sur le bouton d'apprentissage ; la LED témoin suit une séquence de couleurs : blanc / rouge / orange / jaune / vert / bleu clair / bleu foncé / violet qui correspond à un numéro d'identifiant.
Relacher le bouon pour sélectionner l'identifiant de la couleur à mémoriser, par exemple lorsque la LED est verte (identifiant N°4)

2 - Présenter l'objet dans l'axe de visée de l'objectif
Ajuster la position de l'objet jusqu'à tant que la LED située au dos de la caméra (à l'intérieur du boîtier fumé translucide) à l'opposé de l'objectif restitue sa couleur.

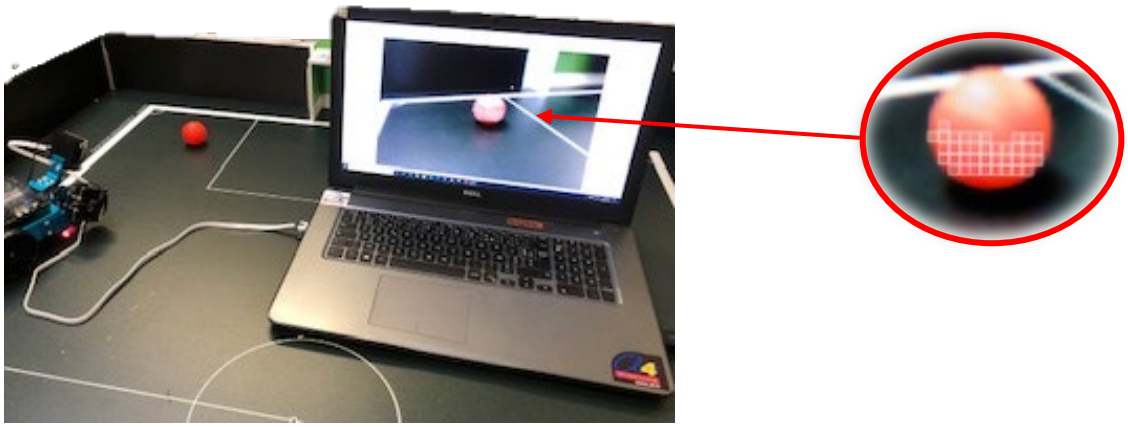
3 - Mémoriser l'identifiant de la couleur
Appuyer brièvement sur le bouton d'apprentissage pour associer l'identifiant sélectionné à l'objet détecté.

Réglage fin de la Smart Camera avec le logiciel PixyMon v2

Le logiciel PixyMon v2 permet de restituer en direct ce que voit la caméra et permet d'effectuer un réglage beaucoup plus précis que celui que l'on peut réaliser avec la procédure précédente. Dans le contexte DéfiBut, il est fortement recommandé d'utiliser ce logiciel pour optimiser le fonctionnement du robot en maximisant la capacité de la Smart Camera à détecter et différencier les différentes couleurs à détecter. Des seuils de détection propres à chaque signature peuvent être ajustés en tenant compte de l'ambiance lumineuse dans laquelle est positionnée l'enceinte robotique DéfiBut (reflets, zone sombre, lumière parasite, etc.)

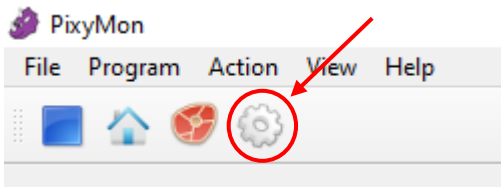
Le logiciel PixyMon v2 est téléchargeable ici : https://pixycam.com/downloads-pixy2/

Connecter la Smart Camera au port USB de l'ordinateur puis lancer le logiciel

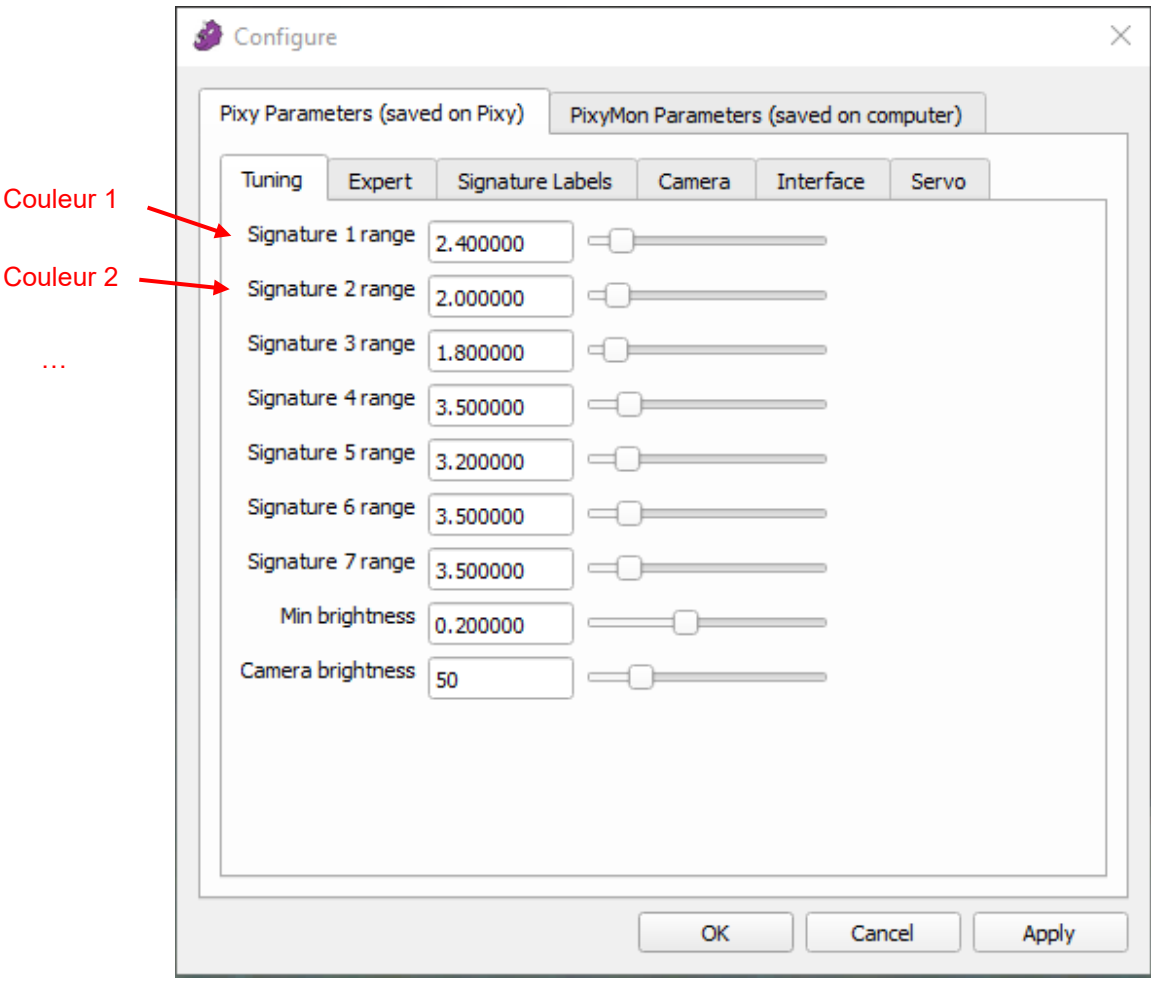


Appuyer rapidement sur le bouton « LEARN » et observer la couleur enregistrée qui est maintenant recouverte d'un grand rectangle gris comme sur l'image qui suit :

Détection plus précise
Toujours sur PixyMon v2, cliquer sur l'engrenage en haut à gauche de l'écran :

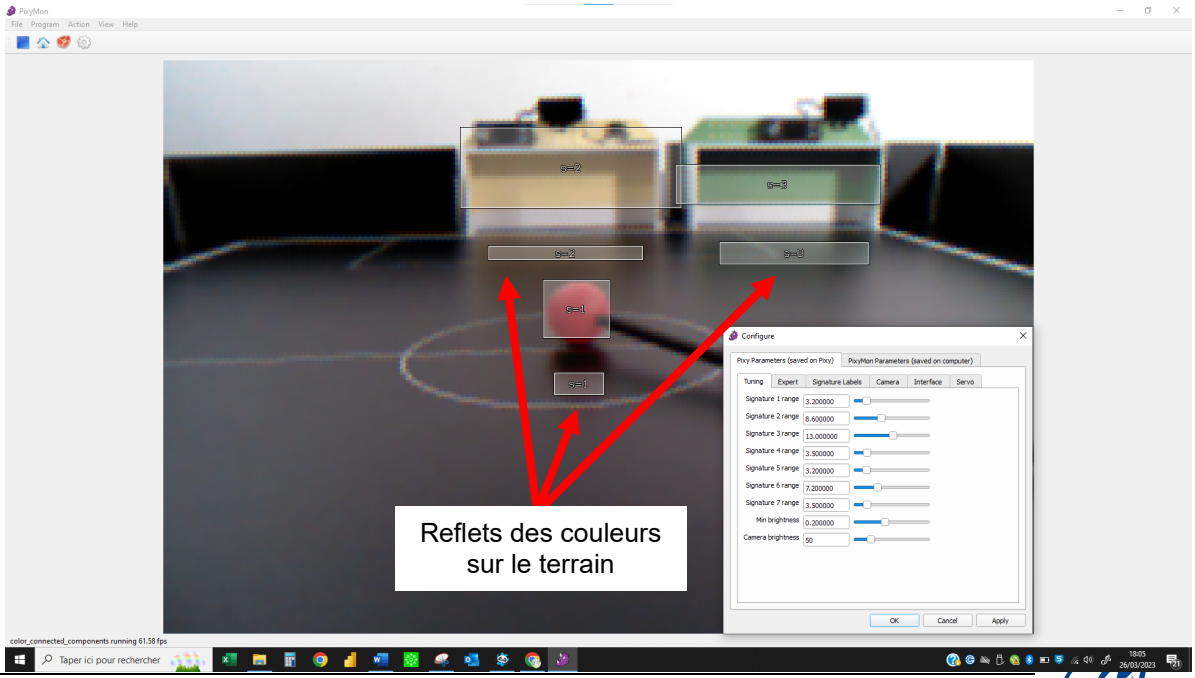


Régler l'intensité de chaque couleur pour qu'elle soit mieux détectée en vérifiant que le rectangle gris recouvre bien uniquement l'objet de la couleur enregistrée :



Exemple de réglage pour détecter la balle rouge (signature 1), le but jaune (signature 2) et le but vert (signature 3):

Les curseurs de sensibilité sont ajustés pour chaque signature 1, 2 et 3. On remarque sur l'image ci-dessous que le reflet de chaque couleur sur le terrain est détecté par la caméra.

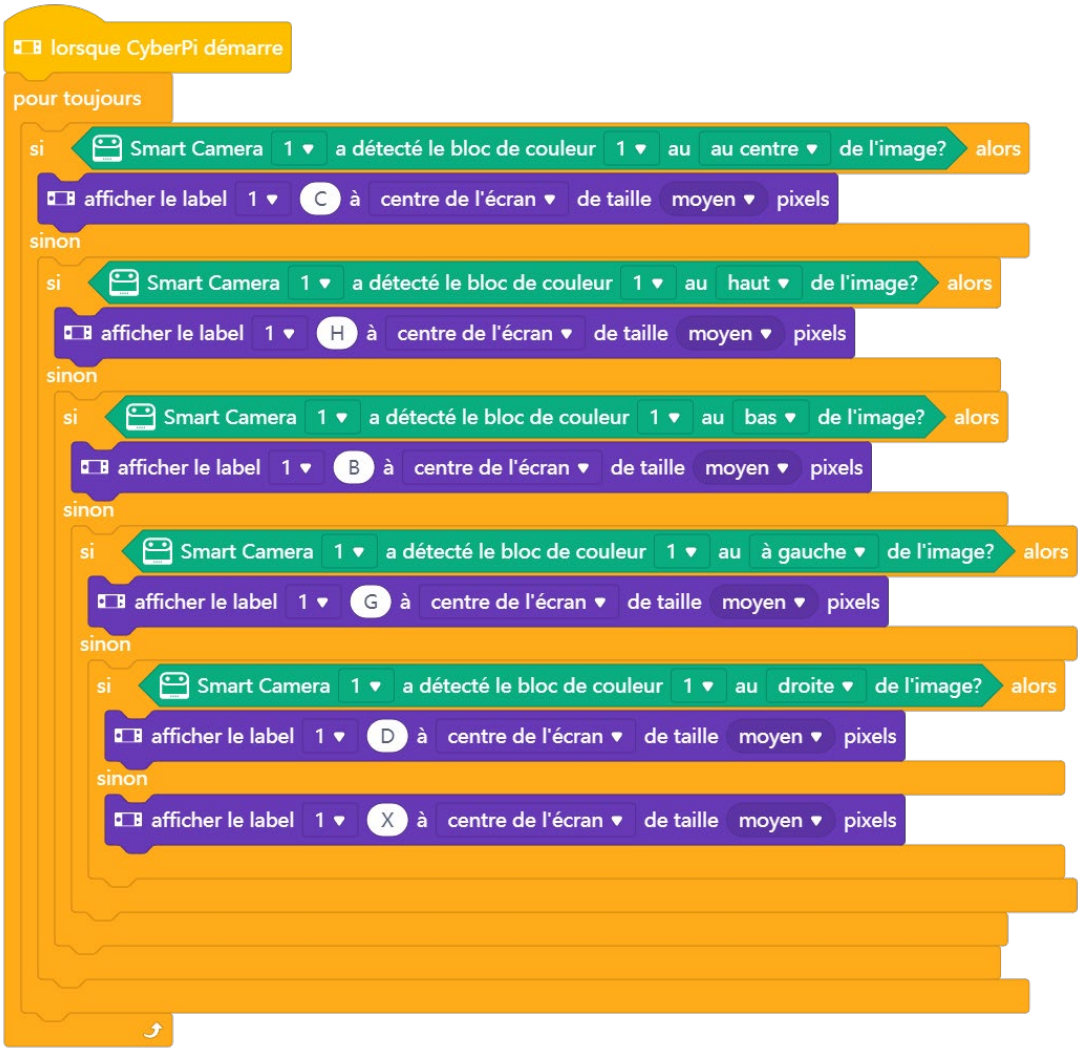


Maîtriser les zones de détection de la balle

L'instruction suivante permet de renvoyer une information de position d'une signature détectée dans le champ de vision de la Smart Camera :

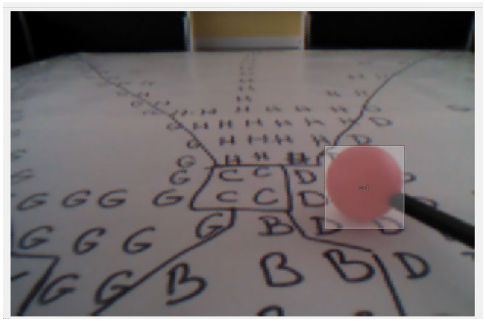
Observer le fonctionnement de la Smart Camera et cartographier la position des différentes zones « au centre », « haut », « bas », « à gauche », « à droite ».

Charger un programme qui affiche la position détectée directement sur l'écran du module d'affichage (CyberPi) du mBot2 :



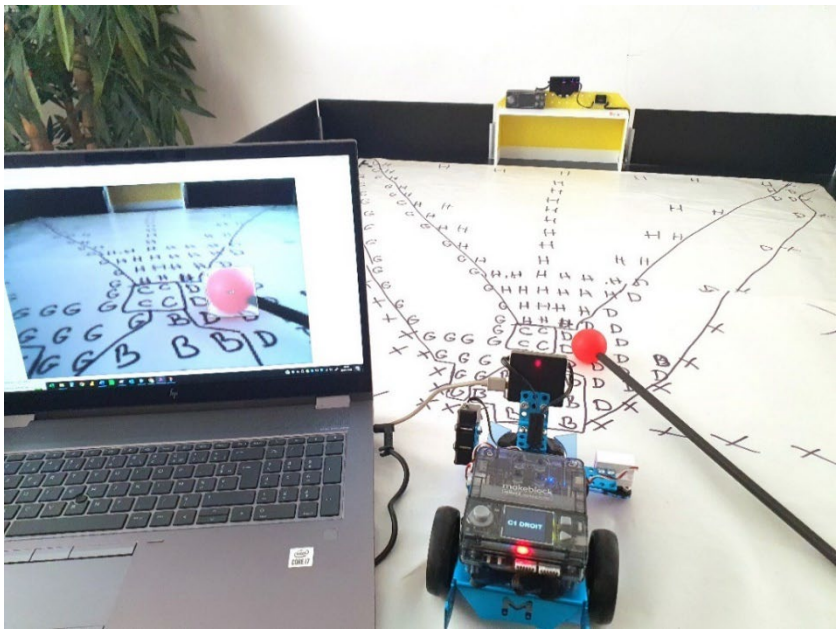
- Connecter la Smart Camera à l'ordinateur
- Lancer le logiciel Piximon V2
- Mémoriser la signature de la balle rouge

- Ajuster la sensibilité de détection de la signature de la balle. Le carré ou rectangle blanc translucide de la signature se superpose à la balle
- Déplacer la balle dans le champ de vision de la caméra
- Relever les valeurs affichées sur le module d'affichage pour dresser la cartographie des différentes zones

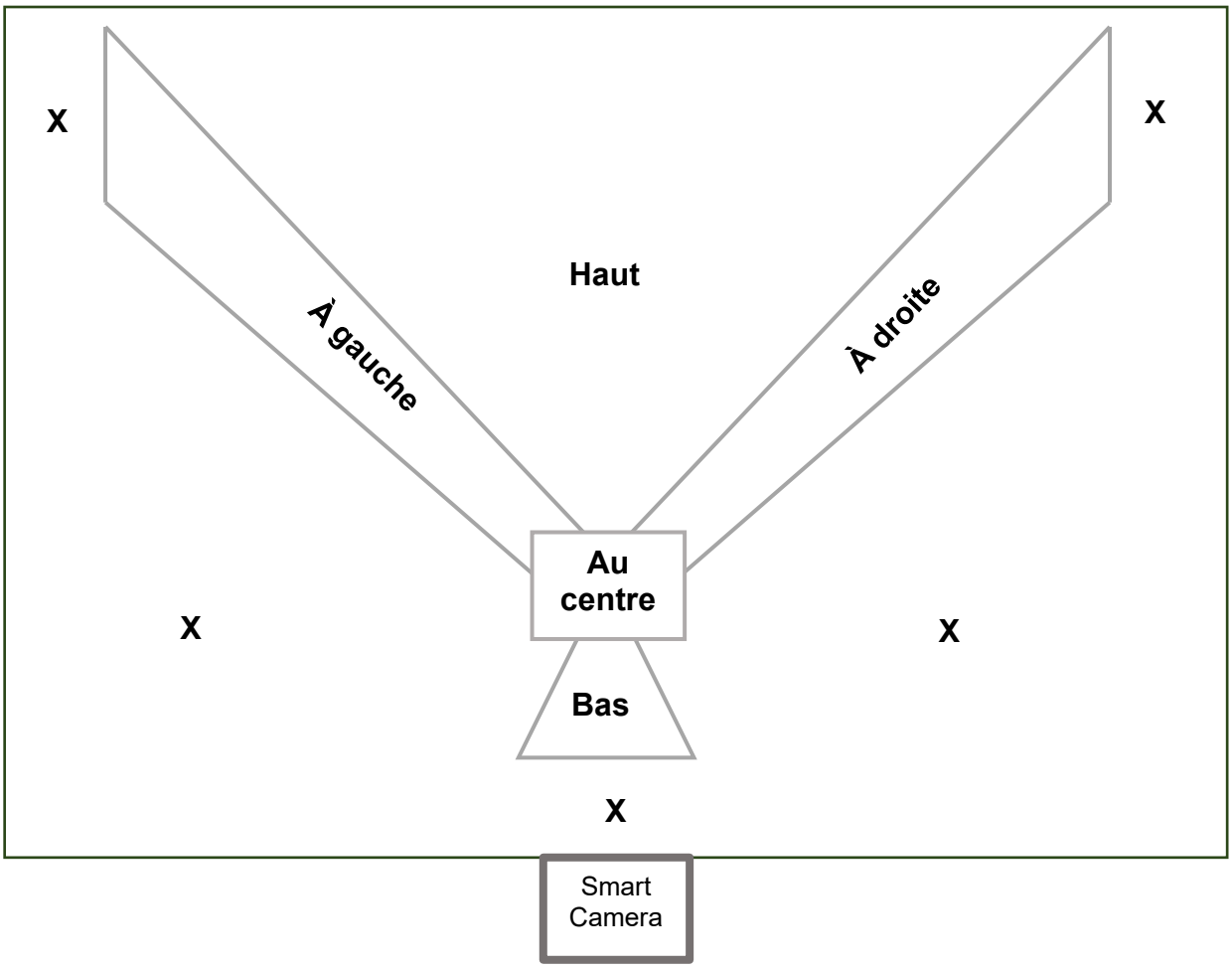


Relevé des informations de position de la balle directement sur une feuille de papier blanc disposée sur le terrain DéfiBut

« C » pour « au centre »,
« H » pour « haut »
« B » pour « bas »,
« G » pour « à gauche »,
« D » pour « à droite »
« X » pour signature non détectée

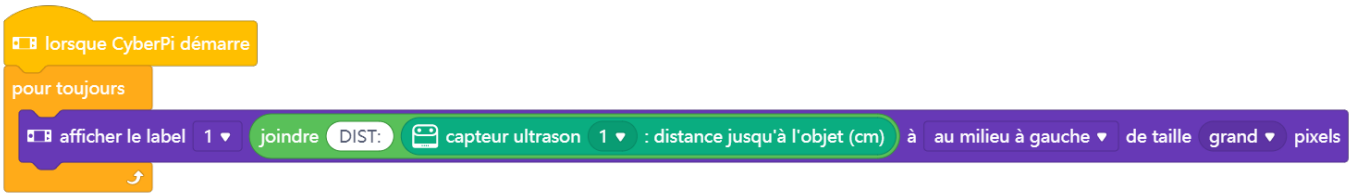


Cartographie des zones de détection de la Smart Camera :

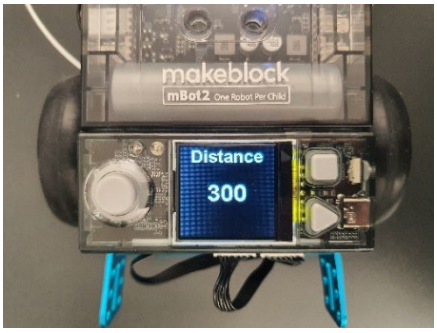


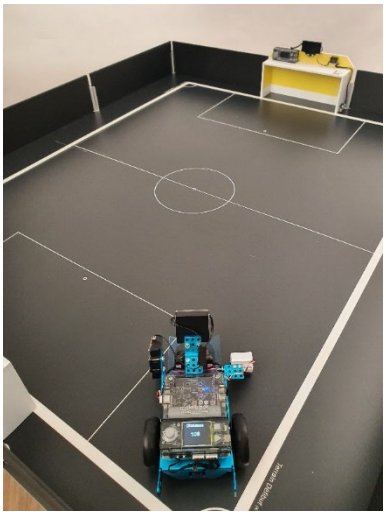
Maîtriser le fonctionnement du capteur de distance

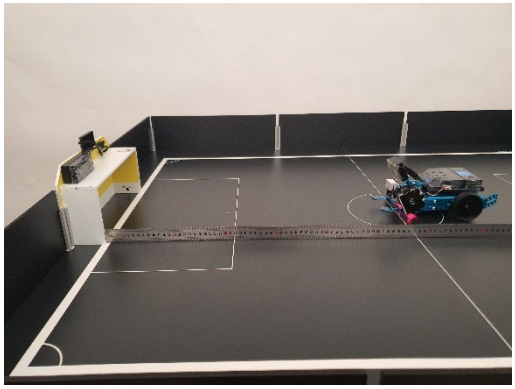
Charger le programme MB2-TEST-US-F1.mBlock afin d'afficher en direct la valeur retournée par le capteur.
Déplacer le robot manuellement sur le terrain DéfiBut et observer les valeurs affichées.

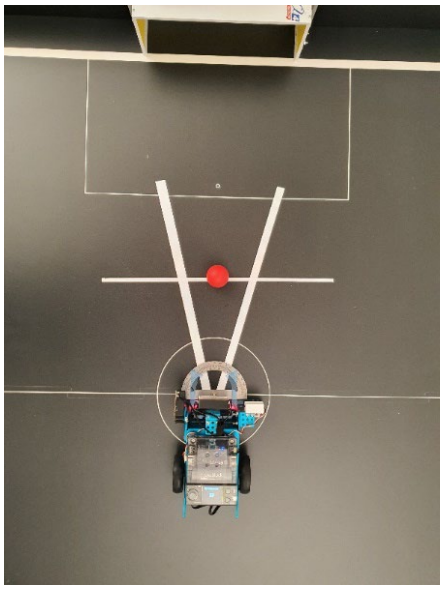


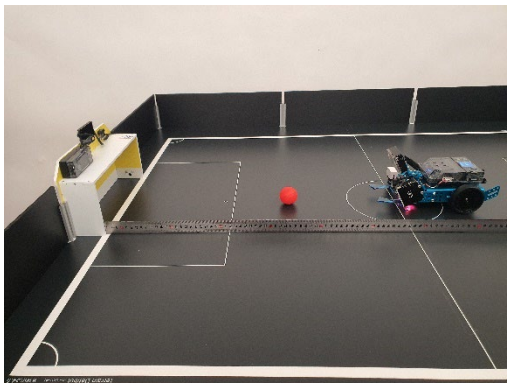
- Quelles sont les valeurs minimales et maximales retournées par le capteur ?
- Quelle valeur est retournée par le capteur lorsqu'il est occulté ?
- Quelle est la distance minimale en dessous de laquelle le capteur ne renvoie pas de valeur aberrante ?
- Quel est l'angle de la zone de détection du capteur ?
- La distance renvoyée par le capteur est-elle exprimée en centimètres ?
- À l'aide d'un mètre, comparer la valeur renvoyée par le capteur avec la distance réelle qui le sépare d'un obstacle.

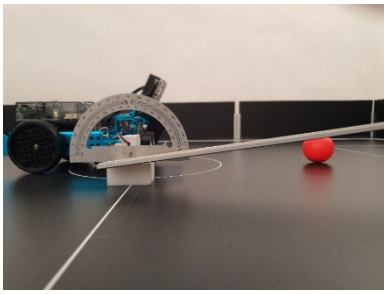








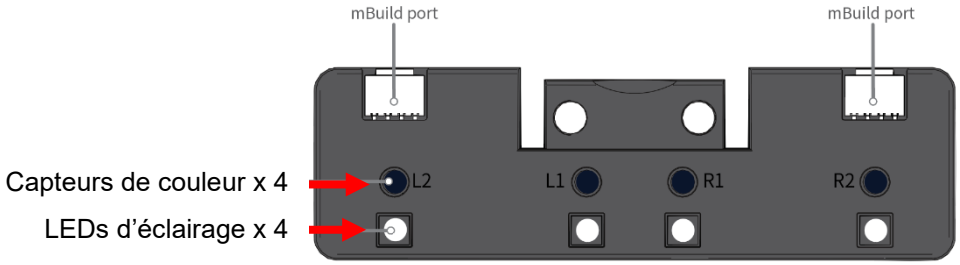




Régler et maîtriser le fonctionnement du capteur de ligne Quad RGB

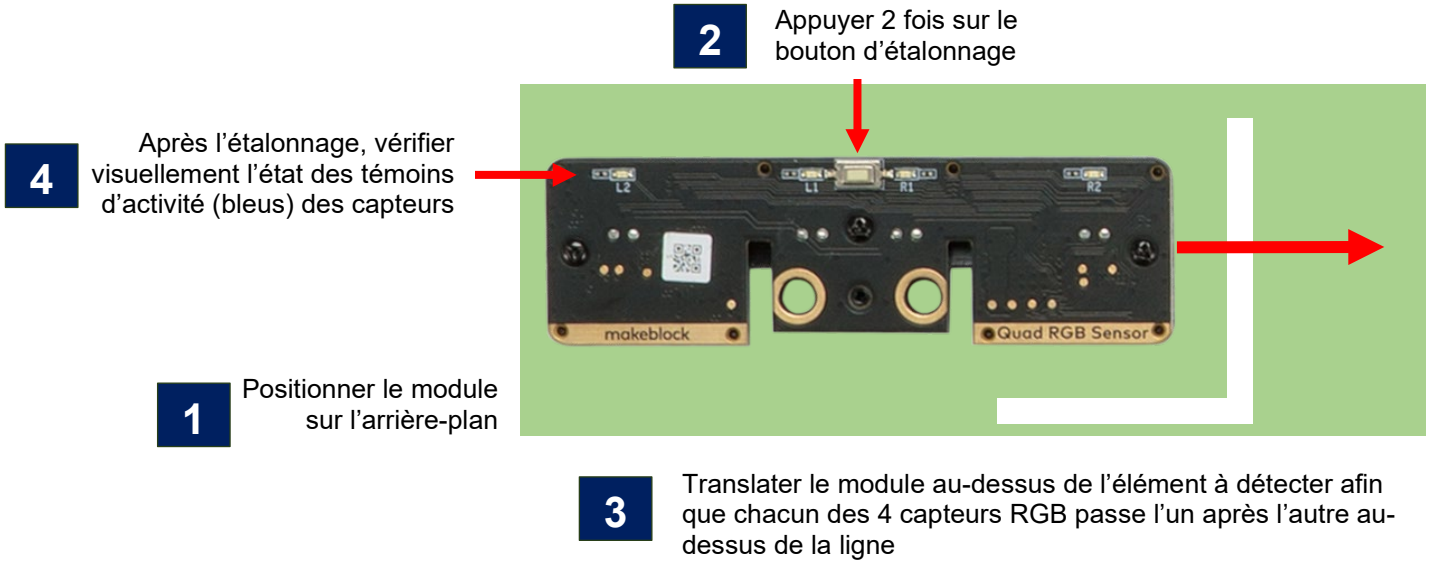
https://education.makeblock.com/help/mbuild-quad-rgb-sensor/

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

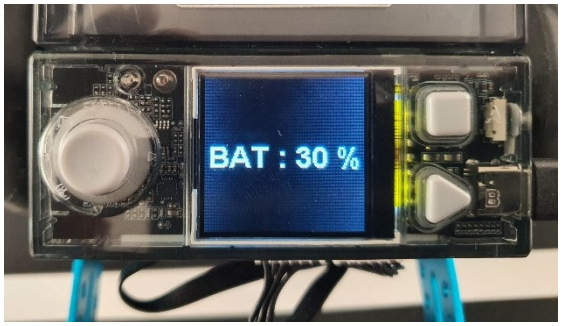


ATTENTION
Dès que le processus d'étalonnage est lancé, les LEDs d'éclairage clignotent et vous disposez de 2,5 secondes pour déplacer le module au-dessus de la ligne à détecter. À l'issue de ce temps, l'étalonnage est mémorisé pour chacun des 4 capteurs RGB. Il est important de vérifier l'efficacité 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

Surveiller l'état de charge de la batterie



www.a4.fr

Concepteur et fabricant de matériels pédagogiques