

PRESENTATION DU LOGICIEL ALPHAI

ET FICHES DE SCENARIOS POUR BIEN DEMARRER

INTRODUCTION

AlphaAi permet de faire comprendre les bases de l'**Intelligence Artificielle** en manipulant un robot apprenant et en lui faisant apprendre un comportement intelligent.

Qu'est-ce que l'Intelligence Artificielle? Nous n'allons pas discuter cette question en détail dans cette introduction mais vous pourrez trouver plus d'éléments au cours de ce document et dans nos ressources en lignes. Contentons-nous de dire ici que le robot AlphaAi et son interface vont présenter une *nouvelle manière de programmer* où au lieu de donner des *instructions*, on s'appuie sur des **algorithmes d'apprentissage automatique**.

C'est pour cette raison que l'interface graphique ne ressemble pas aux environnements graphiques de programmation comme Scratch. Ce qui apparaît au centre de la fenêtre est un **réseau de neurones artificiels** que l'on va paramétrer pour devenir capable d'effectuer les bons **apprentissages**. Au terme de l'apprentissage, le réseau de neurones sera capable de traiter correctement ses *entrées*, à savoir les mesures des capteurs du robots, pour fournir en *sortie* les bonnes **prédictions** des meilleures actions à effectuer.

Ce document commence par une **Présentation générale** des fonctions du programme. Certaines de ces descriptions pourront paraître complexes? Pas de panique! les **Scénarios** qui viennent en deuxième partie vont vous proposer un parcours d'expérimentation avec le robot qui viendra introduire chaque notion l'une après l'autre.

TABLE DES MATIERES

Introduction.....	1
Table des matières.....	2
Guide d'installation logiciel.....	3
Windows.....	3
Linux et Mac	3
Activation de licence	4
Présentation générale du logiciel.....	5
Menus.....	6
Ecran principal	7
Barre des contrôles principaux.....	8
Fenêtre Capteurs	9
Fenêtre Actions	10
Fenêtre Récompenses	11
Fenêtre I.A.	12
Fenêtre Visualisation	13
Graphe de progression.....	14
Scénarios d'apprentissage du robot.....	15
Prise en main avec le robot simulé	16
Prise en main du vrai robot.....	17
Reconnaissance d'image avec la caméra	19
Apprentissage supervisé - navigation avec camera.....	21
Edition manuelle d'un réseau de neurones simple	22
Apprentissage par renforcement - navigation bloqué/ en mouvement	25
Apprentissage par renforcement - navigation avec camera	27
Détection d'intrus.....	28
Suivi d'un ballon	30
Suivi de ligne	32
Table récapitulative des scénarios	33

GUIDE D'INSTALLATION LOGICIEL

WINDOWS

Téléchargez la dernière version de l'installateur à l'adresse <https://learningrobots.ai/downloads/software>

Suivez les instructions de l'assistant d'installation :

Une fenêtre cmd.exe se lance et procède à l'installation du logiciel et son environnement (Miniconda 3, Putty).

LINUX ET MAC

Nous n'avons pas encore développé d'installateur pour Linux et Mac (cela viendra bientôt pour Mac), aussi allez-vous devoir taper quelques lignes de commandes pour installer le logiciel.

Installez Python : nous vous recommandons d'installer la distribution Anaconda

<https://www.anaconda.com/products/individual#download-section>

Ouvrez un terminal et tapez les commandes suivantes :

```
conda create -n alphai python=3.7.6
conda activate alphai
conda install pyqt pyqtgraph vispy param scipy pillow
conda install pyserial imageio
conda install pytorch torchvision cpuonly -c pytorch
pip install gym pyperclip pyserial
```

Allez sur <https://learningrobots.ai/downloads/software/> et téléchargez la dernière version .zip du code AlphaI. Extraire le .zip dans le dossier de votre choix.

(Sous Mac : ATTENTION, l'extraction du .zip en cliquant dessus peut donner un résultat inattendu ; dans ce cas notez le chemin d'accès du dossier qui contient AlphaI-xxx.zip – vous pouvez obtenir ce chemin d'accès dans le Finder en faisant un clic droit sur le dossier, presser la touche Contrôle, et sélectionner « Copier « xxx » en tant que nom de chemin » – puis ouvrez un terminal, allez dans ce dossier avec la commande cd, puis tapez « unzip AlphaI-xxx.zip »)

Pour lancer le logiciel, notez le chemin d'accès du dossier AlphaI (sous Mac, clic droit dans Finder, voir ci-dessus), puis ouvrez un nouveau terminal et tapez :

```
cd "chemin d'accès du dossier AlphaI"
conda activate alphai
python Main_Server.pyc
```

ACTIVATION DE LICENCE

Lors de votre première utilisation du logiciel, une clé d'activation est demandée :

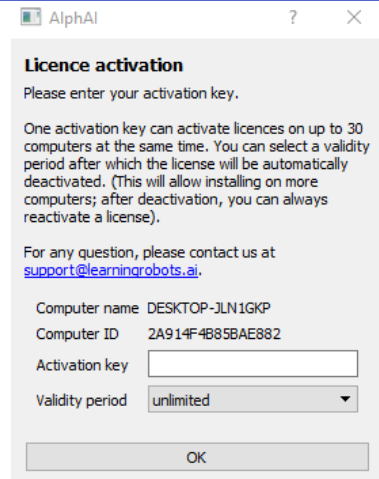
Remplissez le champ « Activation key » à l'aide de la clé d'activation fournie avec le robot (XXX-XXX-XXX).

Chaque clé d'activation permet la création et l'utilisation de 30 licences en simultanées.

Dans le cas d'une utilisation temporaire (par exemple installation sur l'ordinateur personnel d'un élève), indiquer la période de validité nécessaire : à la fin de cette période, la licence associée à cet ordinateur sera supprimée et permettra la création d'une nouvelle licence associée à un autre ordinateur.

La fenêtre suivante vous propose de nous fournir des informations :

Cette étape n'est pas obligatoire mais nous vous recommandons de remplir les différents champs, ces informations nous permettront de vous fournir une assistance mais également de vous informer en cas de mise à jour du logiciel.



AlphaAi ? X

Licence activation

Please enter your activation key.

One activation key can activate licences on up to 30 computers at the same time. You can select a validity period after which the license will be automatically deactivated. (This will allow installing on more computers; after deactivation, you can always reactivate a license).

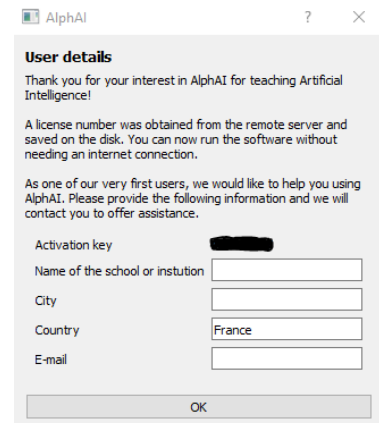
For any question, please contact us at support@learningrobots.ai.

Computer name DESKTOP-JLN1GKP
Computer ID 2A914F4B85BAE882

Activation key

Validity period

OK



AlphaAi ? X

User details

Thank you for your interest in AlphaAi for teaching Artificial Intelligence!

A license number was obtained from the remote server and saved on the disk. You can now run the software without needing an internet connection.

As one of our very first users, we would like to help you using AlphaAi. Please provide the following information and we will contact you to offer assistance.

Activation key

Name of the school or institution

City

Country

E-mail

OK

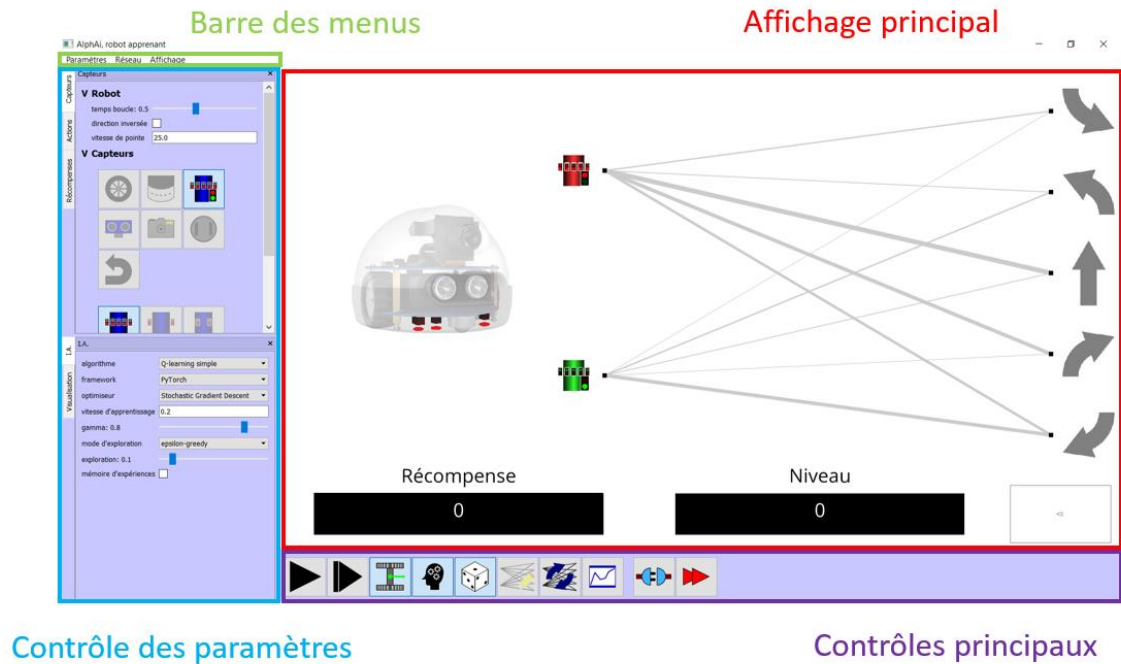
PRESENTATION GENERALE DU LOGICIEL

Le programme présente deux composantes principales :

L'écran d'affichage principal montre l'état du robot, de l'apprentissage en cours, du réseau de neurones.

Les interfaces de contrôles se subdivisent entre :

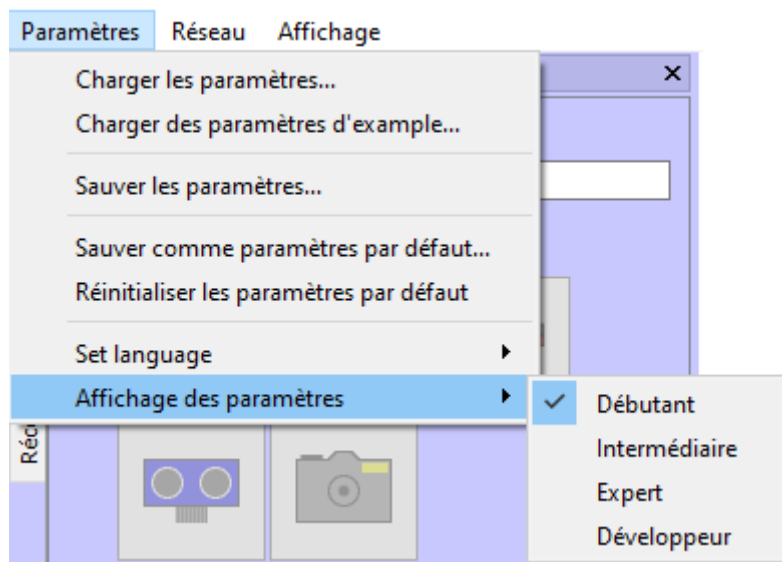
- La *barre des contrôles principaux* en bas
- Les *contrôles des paramètres* sur le côté gauche
- Les *menus*.



MENUS

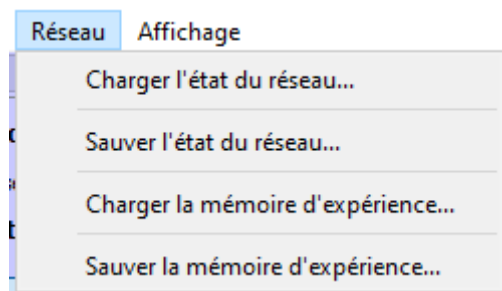
PARAMETRES

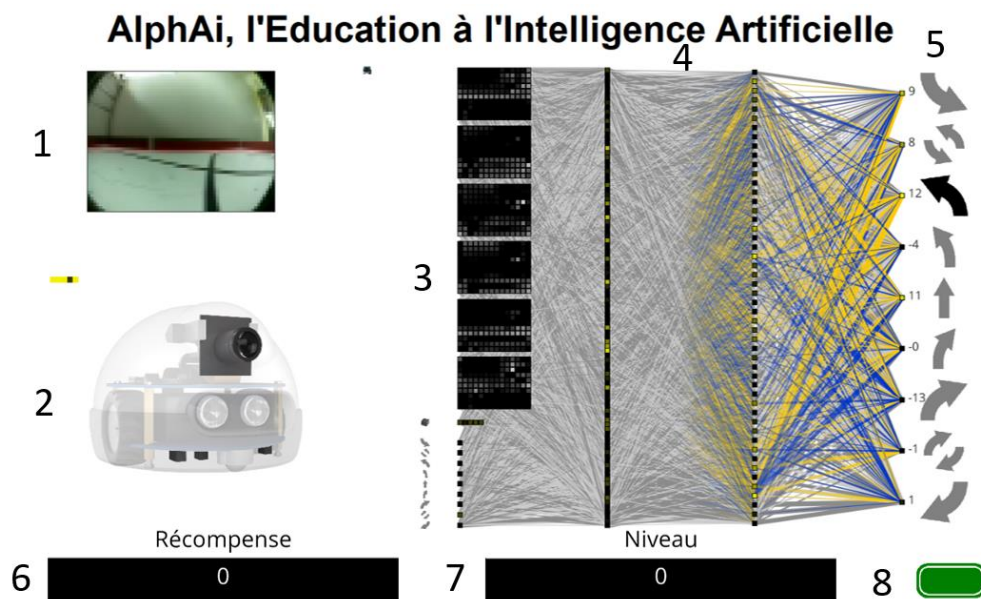
L'onglet **Paramètres** permet de charger des paramètres prédéfinis ou de sauver les paramètres actuels. Il permet aussi d'afficher les paramètres en mode débutant, intermédiaire, ou expert. (Certaines options n'apparaissent qu'en mode intermédiaire et expert ou juste expert.)



RESEAU

L'onglet **Réseau** permet de charger l'état du réseau et de la mémoire d'expérience ou de sauver ceux actuels.





L'écran principal permet de visualiser l'état du robot et de son apprentissage.

1. Affichage de la caméra du robot.
2. Représentation du robot pour visualiser les capteurs qui sont activés.
3. Les capteurs du robot activés. Ce sont les entrées du réseau de neurone.
4. Le réseau de neurone. Vous pouvez voir son évolution en temps réel. (*jaune/bleu transmission d'activité positive / négative ; vert/rouge : renforcement/ diminution des connexion lors de l'apprentissage*) L'épaisseur des traits est proportionnelle au poids de la connexion.
5. Les actions possibles pour le robot. Ce sont les sorties du réseau. Vous pouvez cliquer directement dessus pour forcer le robot à faire une action.
6. Récompense : récompense que reçoit le robot à chaque action lors d'un apprentissage par renforcement.
7. Niveau : moyenne des récompenses sur les 2 dernière minutes.
8. Niveau de batterie du robot. Lorsque le robot n'est pas connecté, à la place s'affiche une représentation du robot simulé. L'arène est représentée par un rectangle, le robot par un triangle. (Le devant est le plus petit côté.)

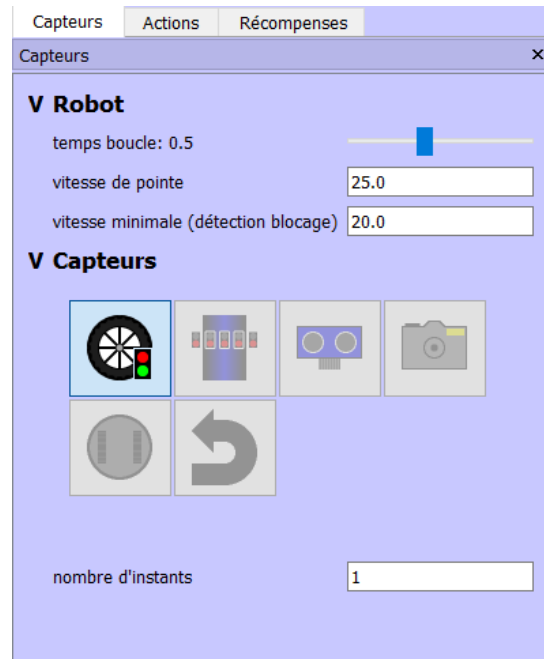


BARRE DES CONTROLES PRINCIPAUX



Notez bien : Tous les contrôles affichent une petite aide lorsque l'on laisse la souris quelques secondes au-dessus.

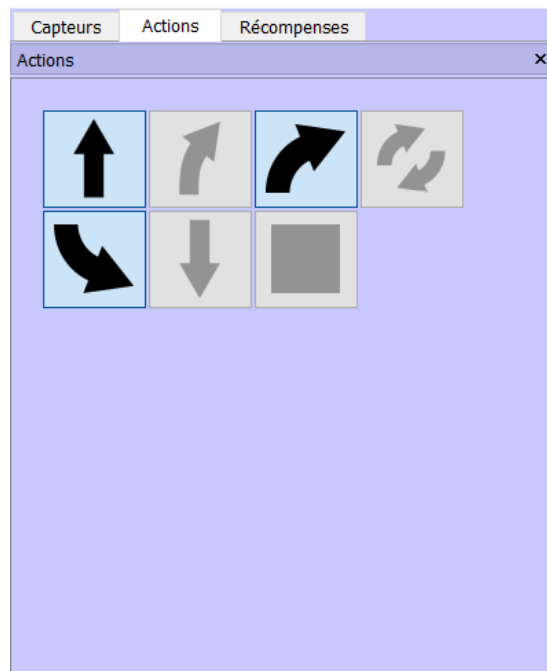
-  /  Démarrer/arrêter le robot
-  Démarrer le robot pas à pas (le robot ne fera qu'une étape à chaque fois que vous appuierez sur le bouton)
-  Autonomie (lorsqu'elle est désactivée, le robot n'effectue que les actions que vous lui rentrez)
-  Apprentissage (autoriser le robot à changer son réseau de neurones pour apprendre.)
-  Exploration (autoriser le robot à prendre de temps en temps une décision choisie aléatoirement)
-  Editer manuellement le réseau de neurones
-  Réinitialiser le réseau de neurones
-  Afficher le graphe de progression du robot (non disponible en paramètre débutant)
-  /  Se connecter au robot
-  Accélérer la simulation (seulement pour le robot simulé)



La fenêtre **Capteurs** permet de choisir les capteurs du robot qui seront utilisés. Cette fenêtre permet aussi de régler quelques paramètres de vitesse du robot.

-  /  Détecter si le robot avance ou pas
-  Capteur de suivi de ligne
-  Ultra son : renvoie la distance qui le sépare d'un obstacle
-  Ultra son : renvoie une réponse binaire en fonction de s'il détecte un obstacle devant lui ou non (à définir avec un certain seuil)
-  Camera du robot (différents formats d'image sont disponibles)
-  Dernière action : toutes les actions choisies pour le robot seront ajoutées aux entrées du réseaux. Lorsque le robot fait une action, celle-ci sera activée dans les entrées à la prochaine étape.

FENETRE ACTIONS



La fenêtre **Actions** permet de choisir les actions que peut effectuer le robot.

(L'astérisque signifie que l'action apparaîtra en double dans la liste des actions : une pour la droite et une pour la gauche)

-  Aller tout droit
-  Tourner*
-  Tourner doucement*
-  Pivoter*
-  Reculer en tournant*
-  Reculer
-  S'arrêter

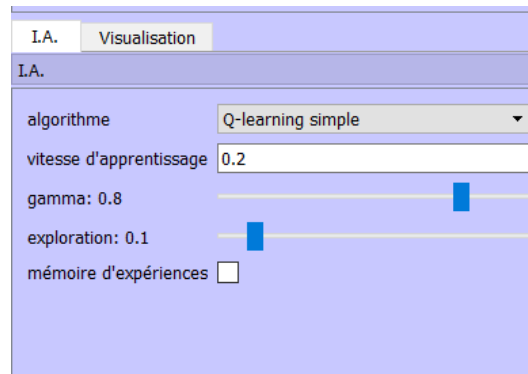
FENETRE RECOMPENSES



La fenêtre **Récompenses** permet de choisir le type de récompense lors de l'apprentissage par renforcement.

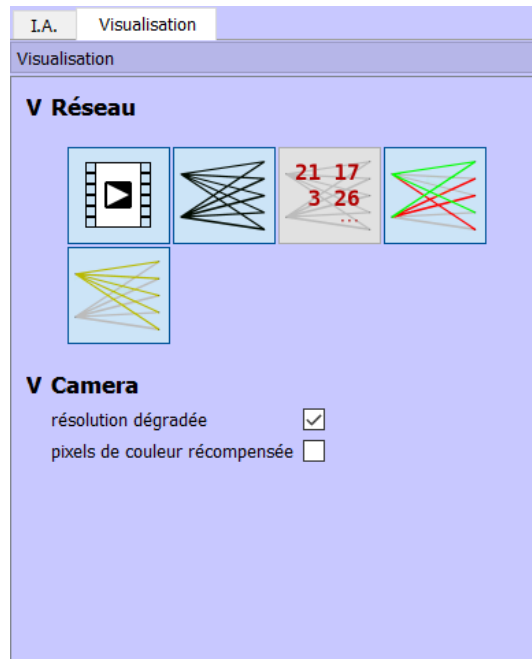
-  Récompenser si le robot va vite
-  Récompenser si le robot va vite mais les hautes vitesses sont encore plus récompensées
-  Récompenser si le robot va vite, punir s'il s'arrête
-  Récompenser si le robot va vite, punir s'il s'arrête, mais les hautes vitesses sont encore plus récompensées
-  Récompenser si beaucoup de pixels de la caméra sont d'une certaine couleur (on peut choisir la couleur en variant les paramètres qui s'affichent ensuite)
-  Récompenser si beaucoup de pixels de la caméra sont d'une certaine couleur, punir si le robot s'arrête
-  Récompenser si détecte du noir juste en dessous de lui

FENETRE I.A.



La fenêtre I.A. permet de choisir de quelle façon le robot est entraîné, notamment pour choisir entre apprentissage supervisé ou apprentissage par renforcement (Deep Q Learning). Elle permet aussi de régler les paramètres des différents algorithmes.

- **Algorithme** : Sélectionner l'algorithme d'IA utilisé
- **Framework** : Sélectionner le framework Python utilisé pour construire les réseaux de neurones
- **Optimiseur** : Algorithme d'optimisation utilisé pour faire évoluer le réseau de neurones
- **Vitesse d'apprentissage** : Augmenter pour des apprentissages plus rapides... mais diminuer si des erreurs de divergences apparaissent
- **Gamma** : Ajuste l'importance donnée aux récompenses immédiates (valeur proche de 0) par rapport aux récompenses plus éloignées dans le temps (valeur proche de 1)
- **Mode d'exploration** : Sélectionner la stratégie d'exploration
- **Exploration** : Fréquence des explorations (valeur entre 0 et 1)
- **Couches de neurones intermédiaires** : Nombre de neurones de chaque couche intermédiaire par exemple : ne rien mettre pour connecter directement les entrées aux sorties mettre "100 50" pour deux couches intermédiaires de respectivement 100 et neurones)
- **Couche convolutionnelles** : Configuration de couches de neurones convolutionnelles : couche est défini par trois paramètres : nombre de kernels, taille des filtres, et stride (c'est à dire facteur de diminution de la taille des images). Les paramètres d'une couche sont séparés par des tandis que les couches sont séparées par des virgules ",". Par exemple "4/5/2, 8/5/2" configure 2 couches, la 1ère avec 4 kernels, la 2de avec 8 kernels, et toutes deux avec des filtres de taille 5 et de stride 2.
- **2 neurones par variable binaire** : Cocher pour que les entrées valant soit vrai soit faux soient représentées par 2 neurones (dont toujours un et un seul sera activé) ; Décocher pour utiliser seulement 1 neurone
- **Régularisation** : Paramètre positif ou nul limitant l'intensité des connexions.
- **Biais neuronal** : Cocher pour permettre aux neurones d'ajuster leur seuil d'activation (cela revient à considérer que tous les neurones reçoivent une entrée constante qu'ils peuvent ajuster, non représentée dans l'interface graphique)
- **Mémoire d'expérience** : Cocher pour que l'IA continue d'apprendre à partir des actions effectuées et récompenses reçues par le passé



La fenêtre **Visualisation** permet de choisir ce qui est affiché ou non sur l'écran principal.

-  Animer l'activité dans le réseau (mouvement des entrées vers les sorties)
-  Afficher les connexions du réseau
-  Afficher les poids des connections
-  Afficher les apprentissages (couleurs vert / rouge pour connexions qui s'intensifient / diminuent)
-  Afficher l'activité dans le réseau (couleur jaune / vert pour les activités d'excitation / inhibition)

GRAPHE DE PROGRESSION

Ce graphe permet de suivre la progression du robot en affichant ses récompenses et l'erreur de ses prédictions au cours du temps. Si le robot progresse bien, le graphe des récompenses doit augmenter et celui des erreurs doit diminuer.

Ces graphes ne sont pas forcément très lisibles, dans ce cas-là vous pouvez les moyenner, par exemple sur 2 minutes, en cliquant sur lissage – 2 minutes



SCENARIOS D'APPRENTISSAGE DU ROBOT

Nous allons maintenant présenter différents scénarios qui permettent de faire un tour d'horizon de ce qu'il est possible de faire avec le logiciel et ce qu'il est possible d'apprendre au robot.

Pour votre propre prise en main du robot, nous vous conseillons de réaliser ces scénarios l'un après l'autre dans l'ordre de leur description ci-dessous. Ensuite vous pourrez bâtir vos propres successions de scénarios, et inventer bien sûr vos propres scénarios de manière à vous adapter au temps dont vous disposerez auprès de vos élèves, au public visé, au niveau de détail de ce que vous voulez enseigner, etc.

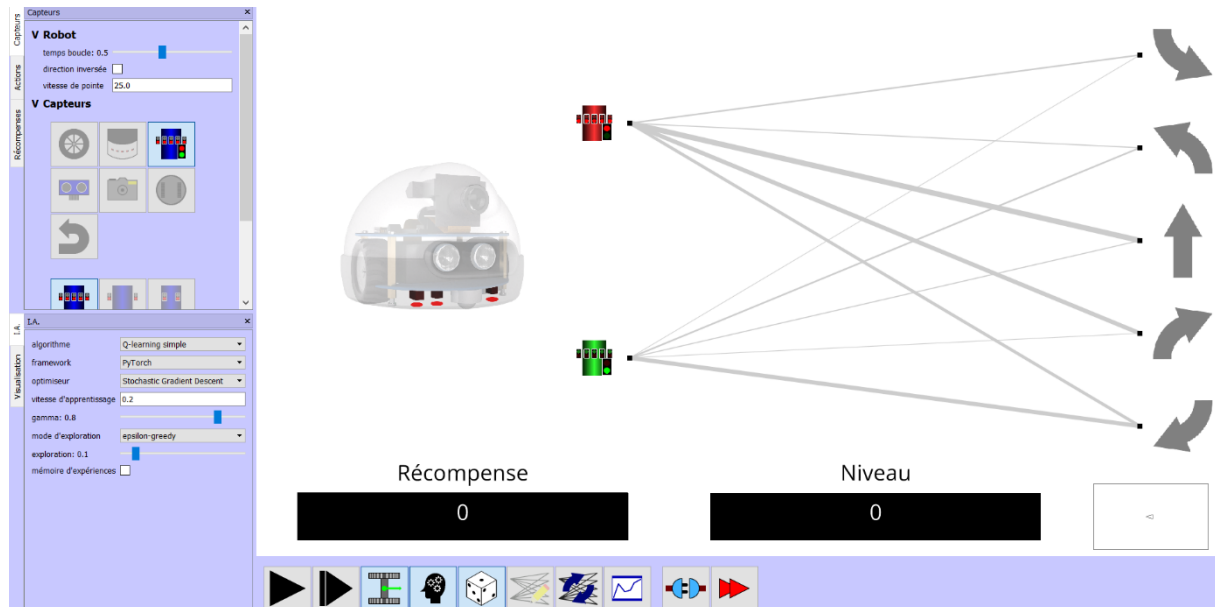
Par défaut dans la plupart des scénarios, les icones , ,  de l'onglet du bas seront activées, et

les actions possibles du robot seront , ,  . Si ce n'est pas le cas, cela sera indiqué dans les instructions.

PRISE EN MAIN AVEC LE ROBOT SIMULE

Lorsqu'on lance le logiciel, le robot n'est pas encore connecté. Mais on peut tout de même faire tourner le logiciel avec un robot simulé. Cela permet de tester les différentes méthodes sans avoir à utiliser le vrai robot.

Allumez le logiciel.



En bas à droite vous pouvez voir le robot simulé. Le robot est représenté par un triangle, le plus petit côté étant le devant du robot, et l'arène par un grand rectangle.

Démarrez  le robot simulé.

Vous devriez voir le robot simulé se déplacer. Généralement il commence par tourner sur lui-même, puis « découvre » la ligne droite, puis « découvre » qu'il faut se retourner lorsqu'il est bloqué par un obstacle.

Notions apprises :

Nous constatons que le comportement du robot simulé s'est *amélioré* au cours du temps. Les grands progrès en ce début de 21^{ème} siècle dans le domaine de l'Intelligence Artificielle sont essentiellement dus à l'amélioration des techniques d'**apprentissage automatique**, on parle en anglais de **machine learning**.

Nous commençons aussi à prendre connaissance de l'**interface graphique** du logiciel AlphaAi : on peut se contenter pour l'instant de faire particulièrement attention au bouton Start/Stop, à l'amélioration du comportement du robot simulé et à l'augmentation correspondante de son **niveau**.

Pas besoin de tout vouloir comprendre à ce niveau : ce sera plus intéressant de le faire en passant au vrai robot !

PRISE EN MAIN DU VRAI ROBOT.

Nous allons maintenant nous connecter au vrai robot ! Pour cela il faut d'abord mettre en place son arène.

Mettez en place une **arène** : si vous construisez votre propre arène, voici quelques conseils :

- Dimension : carré ou rectangle de 80cm à quelques mètres de côté.
- Les murs peuvent être des planches ou tout type d'objet (ramette de papier, etc.). Préférez des surfaces capables d'absorber légèrement les chocs.
- Pour des apprentissages efficaces lorsque le robot naviguera avec sa caméra, veillez à ce que tous les murs aient la même couleur, qui soit une couleur différente de celle du sol.
- Préférez un sol légèrement texturé. En effet lorsque le sol est trop lisse ou de couleur trop uniforme, il peut arriver que le robot ait des difficultés à déterminer s'il est en mouvement (la détection du mouvement se fait par les capteurs infra-rouge situés sous le robot, c'est le même problème que lorsqu'on utilise une souris d'ordinateur sur une surface trop lisse).
- Attention, veillez à ce qu'il n'y ait pas de poussière au sol, car elle abîme les moteurs. Dépoussiérez le sol avec un chiffon avant usage, ne faites pas rouler le robot sur de la moquette.



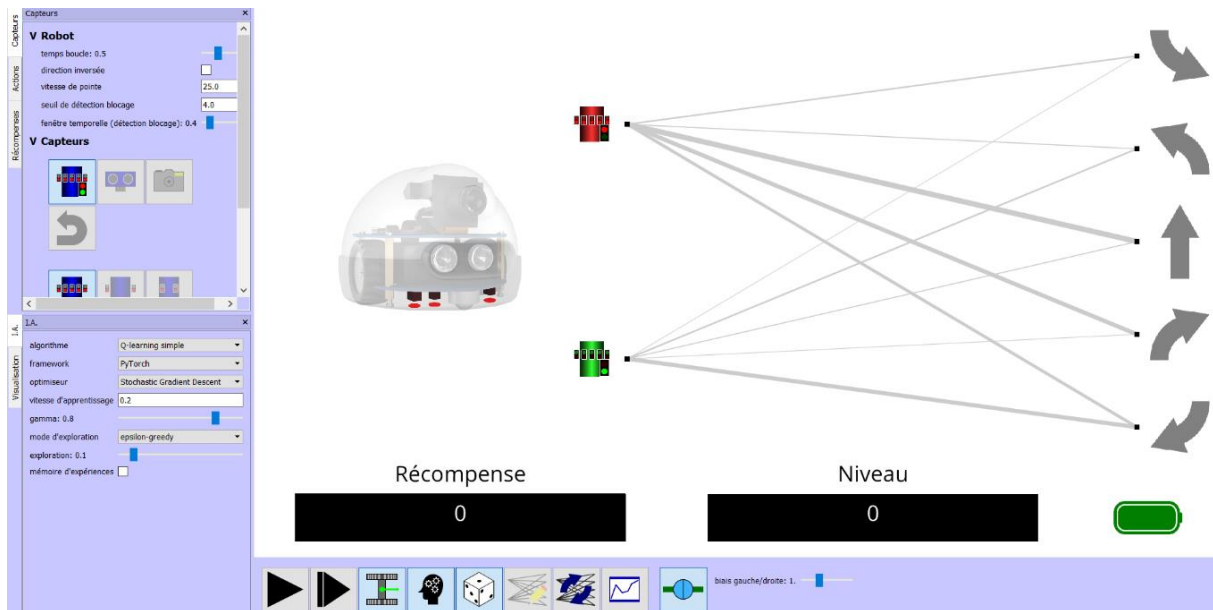
L'arène proposée par Learning Robots

Allumez le robot (l'interrupteur se trouve en dessous). Il effectue un petit mouvement lorsqu'il est prêt. Connectez-vous en wifi du robot (chercher le wifi qui commence par ALPHAI : le mot de passe est identique au nom du wifi).



Sur le logiciel, appuyez sur le bouton connect

Vous êtes maintenant connecté au robot. Le niveau de sa batterie doit s'afficher en bas à droite (vérifier que le robot est bien chargé)



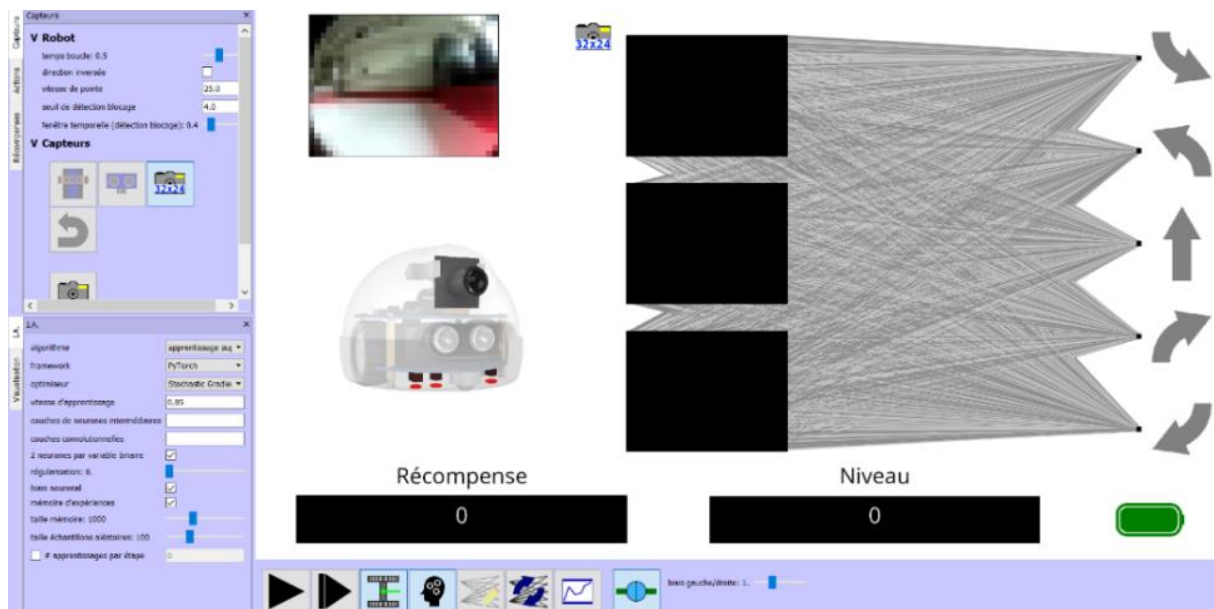
Démarrez  le robot. Vérifiez que le robot se déplace correctement. Vous pouvez aussi donner directement des commandes au robot en appuyant sur les flèches à droite de l'écran ou en utilisant les flèches directionnelles du clavier.

RECONNAISSANCE D'IMAGE AVEC LA CAMERA

Nous allons voir une des catégories d'apprentissage la plus utilisée dans l'intelligence artificielle ; l'apprentissage supervisé. Le but est d'utiliser la caméra du robot pour lui faire reconnaître des personnes et les catégoriser. Dans un premier temps ce sera à nous d'indiquer au robot comment classer chaque personne, mais ensuite il sera capable de les reconnaître par lui-même.

Chargez la configuration « reconnaissance d'image avec la caméra ». Ou pour sélectionner vous-mêmes les paramètres, lancez le programme puis :

- Dans l'onglet IA :
 - Sélectionnez « Apprentissage supervisé »
 - Mettez vitesse d'apprentissage sur 0,05
 - Cochez mémoire d'expérience
- Dans l'onglet Capteurs, sélectionnez :
 -  « caméra » (par exemple 32x24).



- Dans l'onglet action, mettez autant d'action que de personne/objets à faire reconnaître.
- Prenez le robot dans les mains, de façon à ce qu'il ne bouge pas même lorsque ses roues tournent.
- Démarrez  le robot.
- Pointez une personne ou un objet avec la caméra du robot, et appuyez plusieurs fois sur une des flèches sur l'écran (à droite du réseau). Cette flèche servira à identifier la personne ou l'objet.
- Répétez cette opération sur d'autres personnes ou objets en leur assignant des flèches différentes à chaque fois.
- Revenez sur les différentes personnes ou objets que vous avez pointés ; le robot sera capable de les reconnaître en indiquant la bonne flèche qui les identifie (vérifiez-le en regardant la flèche qui s'affiche sur l'écran).

On pourra faire remarquer aux élèves que le robot ne reconnaît pas vraiment la personne ou l'objet car il analyse l'image en entier, y compris le décor derrière. Si vous changez de place une personne ou un objet le robot risque de ne plus la reconnaître.

Notions apprises :

Ce qu'on appelle aujourd'hui Intelligence Artificielle est les capacités d'**apprentissage automatique** des ordinateurs (en anglais « *machine learning* »).

Le robot s'entraîne sur des **données d'entraînements** où les bonnes réponses sont données par l'opérateur humain, puis il est capable de **généraliser** à des **données de test** nouvelles. On appelle cela l'apprentissage supervisé

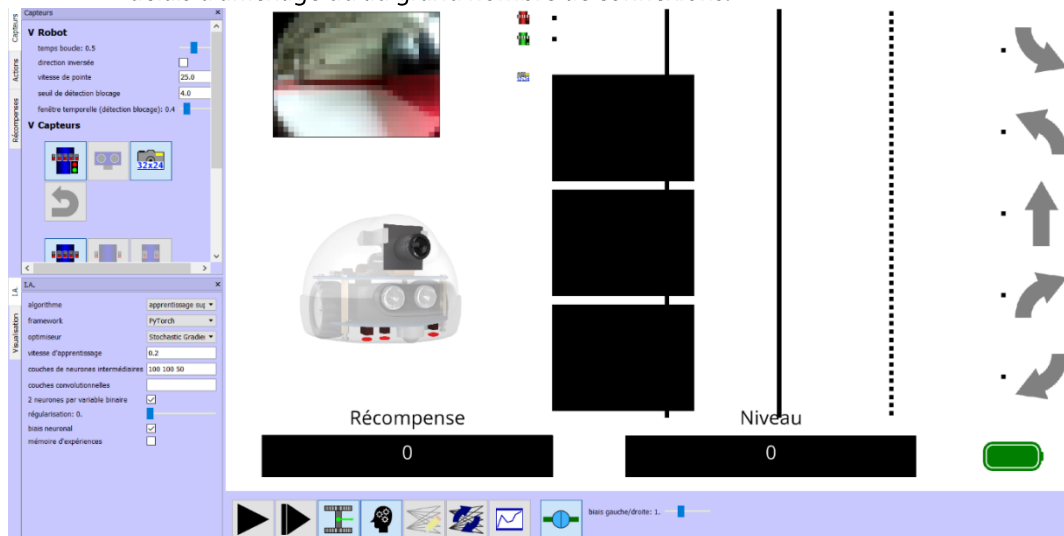
APPRENTISSAGE SUPERVISE - NAVIGATION AVEC CAMERA

Nous allons voir comment utiliser l'apprentissage supervisé de façon à bien diriger le robot dans l'arène. On voudrait que le robot aille tout droit le plus souvent possible, mais évite de se cogner au mur. Comme on active la caméra, le robot pourra être capable de repérer les murs. Comme précédemment, il faut d'abord montrer au robot comment agir avant de le laisser agir par lui-même.

Chargez la configuration « apprentissage supervisé – navigation avec camera ».

Ou pour sélectionner vous-mêmes les paramètres, lancez le programme puis :

- Dans l'onglet IA :
 - sélectionnez « Apprentissage supervisé » ;
 - dans « couches de neurones intermédiaires » mettez « 100-100-50 »
- Dans l'onglet Capteurs, sélectionnez :
 -  « bloqué/en mouvement »
 -  « caméra » (par exemple 32x24)
- (non obligatoire) Dans l'onglet Visualisation, :
 - décochez les options d'affichage du réseau de neurone pour garantir qu'il n'y aura pas de délais d'affichage dû au grand nombre de connexions.



- Démarrez  le robot.
- Tant que le robot n'est pas coincé, appuyez sur aller tout droit
- Si le robot s'approche d'un mur, appuyez sur tourner à gauche/droite
- Si le robot se coince, appuyez sur reculer à gauche/droite

Attention ; si jamais vous dites au robot de tourner trop tôt ou trop tard, le robot apprendra sur une base de données faussée et agira n'importe comment. Comme il n'est pas forcément facile d'appuyer juste au bon moment, vous pouvez enlever l'autonomie du robot pour donner étape après étape les instructions du robot. Ensuite réactivez l'autonomie et continuez de corriger le robot s'il se trompe de temps en temps.

Notions apprises :

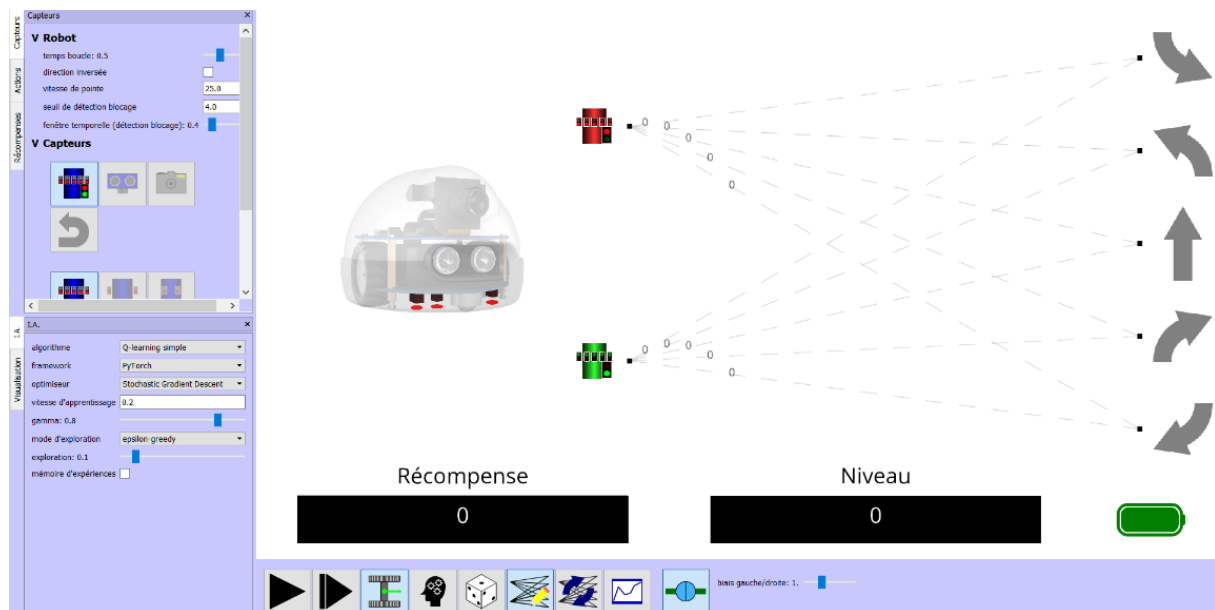
Dans l'apprentissage supervisé, il est important de donner des instructions correctes dans la phase d'apprentissage.

EDITION MANUELLE D'UN RESEAU DE NEURONES SIMPLE

Nous allons voir plus en détail comment fonctionne les connections dans le réseau de neurone du robot avec un cas simple.

Chargez la configuration « édition manuelle d'un réseau de neurones simple ». Ou pour sélectionner vous-mêmes les paramètres, lancez le programme puis :

- Dans onglet IA :
 - Sélectionnez q-learning simple
- Dans l'onglet Capteurs, sélectionnez :
 -  « bloqué/en mouvement »
- Dans l'onglet Récompense :
 - choisissez « grande vitesse, punition à l'arrêt » 
- Dans l'onglet du bas :
 - cochez le mode « édition manuelle simple » 
 - cliquez sur « réinitialise le réseau » 
 - décochez apprentissage  et exploration 



Les entrées possibles sont :

- Le robot n'est pas bloqué
- Le robot est bloqué

En sortie, 5 actions sont possibles : reculer à droite, tourner à gauche, aller tout droit, tourner à droite, reculer à gauche.

Documentation AlphaAi

Ici il n'y a pas d'apprentissage, c'est à **vous de jouer** en créant les liens adéquats entre ces entrées (les capteurs) et ces sorties (les actions) pour générer un comportement cohérent du robot. Si vous créez un lien entre un capteur et une action, cela signifiera que dès que le capteur est activé, le robot effectuera l'action qui y est relié.

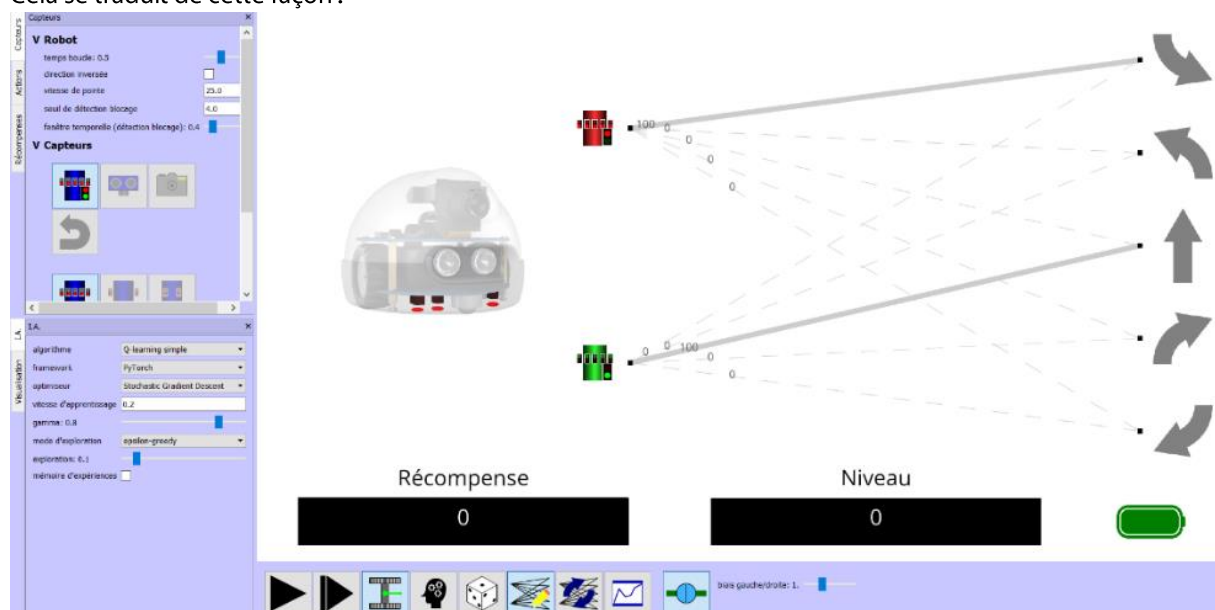
Tracez les connexions que vous pensez être bonnes, puis démarrez  le robot. Vous pouvez également modifier les connexions une fois le robot démarré.

Objectif : Mettre en place les connexions qui permettent au robot d'obtenir le plus de récompense et donc d'atteindre le niveau le plus élevé possible. *Indice : le robot reçoit des récompenses lorsqu'il va en avant.*

Solution : On voudrait que le robot aille tout droit le plus souvent possible, et recule dès qu'il est coincé. Pour cela, il faut donc indiquer au robot :

- S'il n'est pas bloqué, aller tout droit
- S'il est bloqué, reculer.

Cela se traduit de cette façon :



Avec ces connexions il devrait avoir le comportement voulu.

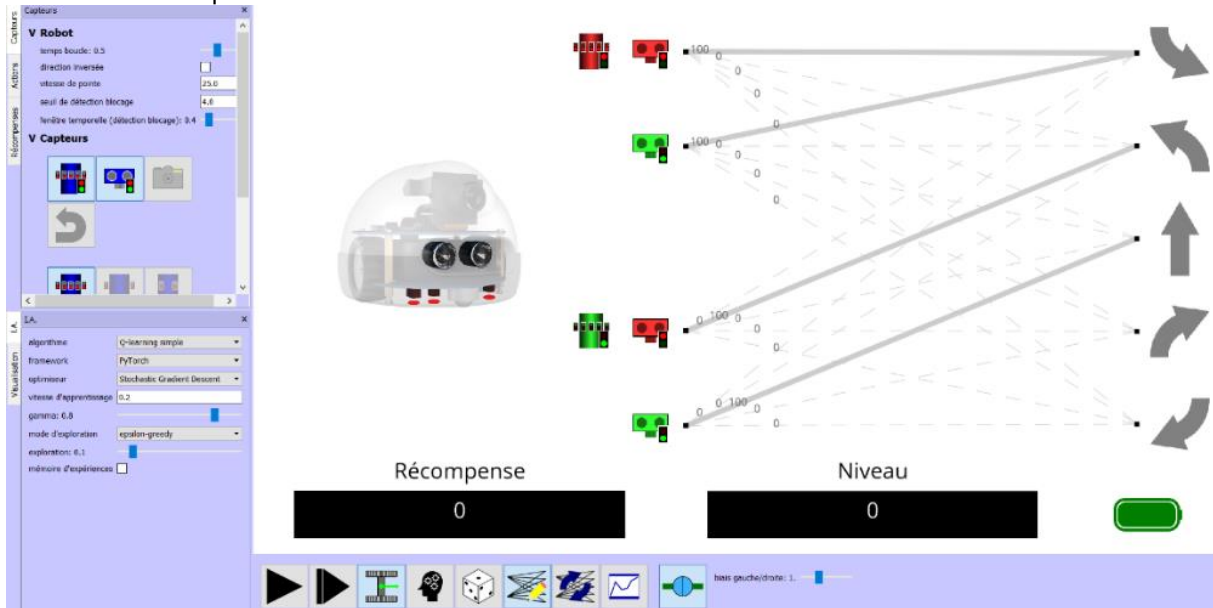
Variante : pour ajouter un peu de complexité, on peut prendre en compte les ultra-sons. Dans l'onglet capteur

rejouez absence/présence d'obstacle 

A présent le robot peut détecter les murs grâce à ces ultrasons, on peut donc s'en servir pour le faire anticiper les murs et prendre un virage avant.

Solution : Il faut indiquer au robot :

- S'il n'est pas bloqué et qu'il n'y a rien devant lui, aller tout droit
- S'il n'est pas bloqué mais qu'il y a un obstacle devant lui, tourner
- S'il est bloqué, reculer.



Démarrez  le robot ; il devrait avoir le comportement voulu. Le robot aura peut-être du mal à anticiper arrive pas de façon bien perpendiculaire. Si le robot n'anticipe aucun mur, il faut peut-être changer la distance de détection des obstacles.

Notions apprises :

Pour que le robot ait un comportement, il faut créer des connections entre ces capteurs et les différentes actions qu'il peut faire.

APPRENTISSAGE PAR RENFORCEMENT - NAVIGATION BLOQUE/ EN MOUVEMENT

Précédemment nous avons-nous même créé le réseau de connexions pour que le robot agisse comme on le voulait. A présent nous allons voir qu'on peut utiliser l'intelligence artificielle pour que le robot parvienne de lui-même à construire ce même réseau sans intervention humaine. Pour cela on va se servir de l'apprentissage par renforcement.

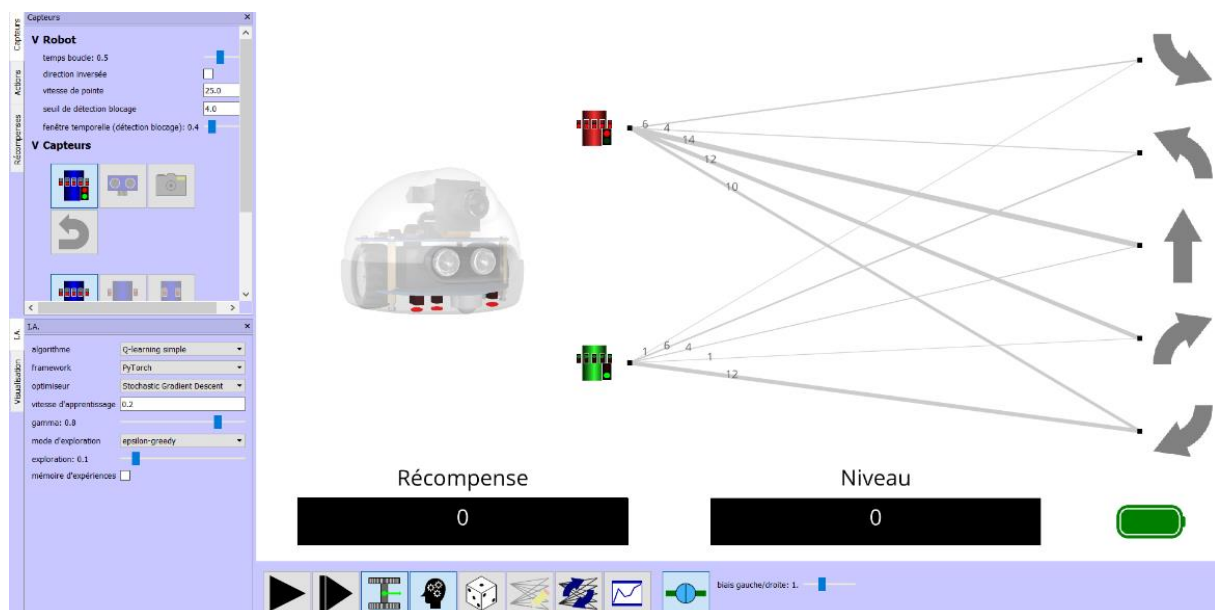
Le principe de l'apprentissage par renforcement est d'encourager le robot à faire certaines actions en lui donnant des récompenses ou des punitions. Le robot modifiera alors son comportement de façon à maximiser son niveau, qui est la moyenne de ses récompenses sur les 2 dernière minutes.

Ici, plus le robot ira vite, plus on lui donnera de récompense, mais s'il s'arrête, on lui donnera une punition. Pour maximiser son niveau le robot aura intérêt à aller tout droit le plus souvent possible et à s'arrêter le moins souvent possible.

Chargez la configuration « apprentissage par renforcement – navigation bloqué/ en mouvement ».

Ou pour sélectionner vous-mêmes les paramètres, lancez le programme puis :

- Dans l'onglet IA :
 - Sélectionnez Q-learning simple
- Dans l'onglet Capteurs, sélectionnez :
 -  « bloqué/en mouvement »
- Dans l'onglet du bas :
 - décochez exploration 
- Dans l'onglet Récompense :
 - choisissez « grande vitesse, punition à l'arrêt » 



Contrairement à la dernière fois, les capteurs ne sont pas reliés à une seule action mais à toute en même temps. Chaque connexion possède un poids différent qui correspond à la récompense que le robot s'attend à recevoir en utilisant cette connexion. Pour l'instant les poids sont initialisés aléatoirement.

Démarrez  le robot. Vous pouvez aussi le lancer pas à pas  pour mieux comprendre ce qui se passe à chaque étape.

Initialement le robot n'est pas coincé, pour lui la meilleure action, celle qui lui apporterait le plus de récompense (celle avec le poids le plus fort) est de reculer, donc il choisit cette action. Mais comme il ne reçoit pas de récompense, il diminue le poids de cette arrête, puis il recommence. Comme il ne reçoit toujours pas de récompense, il continue de la diminuer, jusqu'à ce que ce poids devienne inférieur à celui d'une autre arrête. A ce moment, le robot pense que l'action qui lui donne le plus de récompense est tourner à gauche. Il tourne donc à gauche, obtient une récompense supérieure à sa prédiction, donc augmente le poids de cette arête jusqu'à ce qu'elle se stabilise à 30. A ce moment, le robot pense que tourner à gauche est la meilleure action, et comme cette action le fait tourner sans jamais s'arrêter, il continuera à la faire indéfiniment.

Le robot se met donc à tourner a gauche sans s'arrêter. Or ce n'est pas sa meilleure action : il aurait un meilleur niveau en allant tout droit. Mais comme le robot n'a jamais essayé d'aller tout droit, il ne peut pas le savoir. Il faut donc forcer le robot à faire des essais.

Activez la case exploration  dans l'onglet du bas.

Lorsque l'exploration est activée, le robot ne choisira pas forcément l'arrête de poids le plus grand, mais choisira de temps en temps une autre direction. Lorsque le robot finira par choisir au hasard l'action aller tout droit, il recevra une forte récompense et augmentera Le poids de cette arrête. Au bout d'un moment le poids de l'arrête dépassera celui de tourner à gauche, et le robot se mettra à aller tout droit au lieu de tourner en rond.

Au bout d'un moment, le robot a le comportement voulu (aller tout droit le plus souvent possible, reculer dès qu'il est coincé), mais cette fois-ci il y est parvenu entièrement par lui-même, sans qu'on ait eu besoin de lui indiquer quoi faire.

Notions apprises :

Pour permettre au robot d'apprendre sans être supervisé par des humains, on lui donne plus ou moins récompense à chaque fois qu'il effectue une action. En fonction de ce qu'il recevra, **le robot adaptera se choix de façon à maximiser ses récompenses.**

Les récompenses en question sont simplement une valeur numérique, négatives s'il s'agit d'une punition, que le robot cherche à maximiser.

Il faut aussi **permettre au robot d'explorer** et de ne pas toujours choisir l'action qui lui semble la meilleure, car cela lui permettra de découvrir de nouvelles actions qui pourront s'avérer encore meilleure.

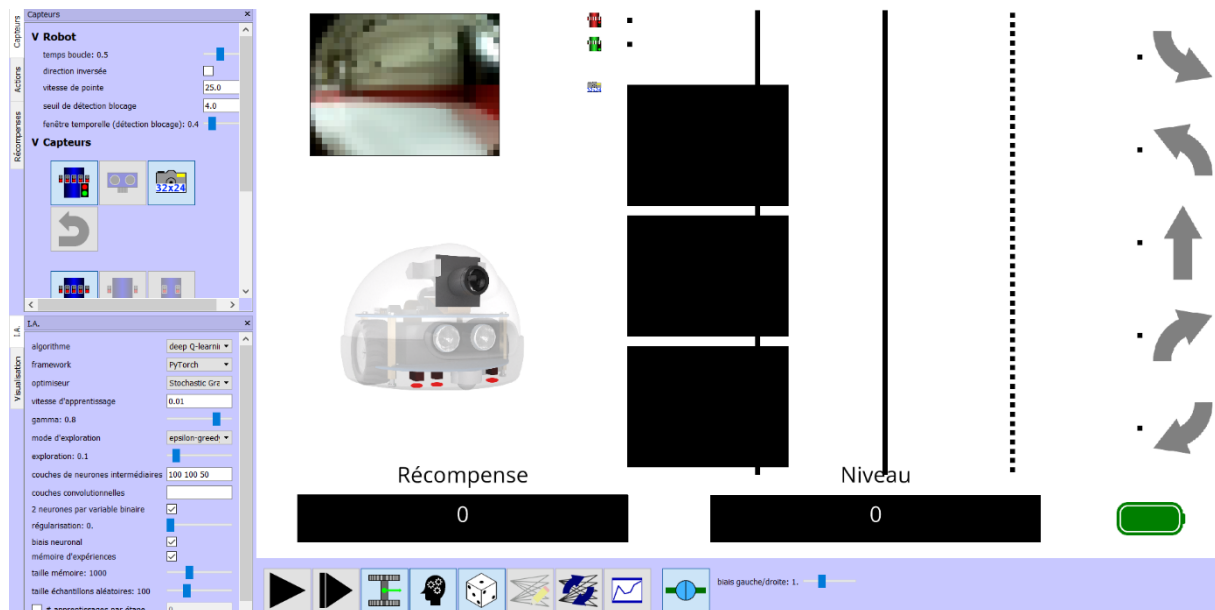
APPRENTISSAGE PAR RENFORCEMENT - NAVIGATION AVEC CAMERA

Le principe est le même que précédemment, mais cette fois on utilise la caméra du robot. Les entrées sont donc très nombreuses (tous les pixels de la caméra et leur composition de rouge, bleu et vert) et on a besoin d'ajouter des couches de neurones pour bien analyser l'image; on se retrouve donc avec un réseau très complexe, qu'on ne pourra pas faire à la main ou essayer de comprendre étape par étape.

Chargez la configuration « apprentissage par renforcement – navigation avec camera ».

Ou pour sélectionner vous-mêmes les paramètres, lancez le programme puis :

- Dans l'onglet IA :
 - sélectionnez « deep Q-learning »
 - dans « couches de neurones intermédiaires » mettez « 100 100 50 »
 - cochez mémoire d'expérience
- Dans l'onglet Capteurs, sélectionnez :
 -  « bloqué/en mouvement »
 -  « caméra » (par exemple 32x24)
- Dans l'onglet récompense :
 - choisissez « grande vitesse, punition à l'arrêt » 



Démarrez  le robot. Au bout d'un certain temps (environ 10 minutes) le robot finira par parcourir toute l'arène en anticipant les murs.

Notions apprises :

L'intérêt de l'intelligence artificielle est de pouvoir **traiter des réseaux gigantesques** et trouver les bonnes connexions qui permettront au robot d'avoir le comportement voulu, alors qu'il serait impossible d'essayer de le faire à la main ou de les intuitier.

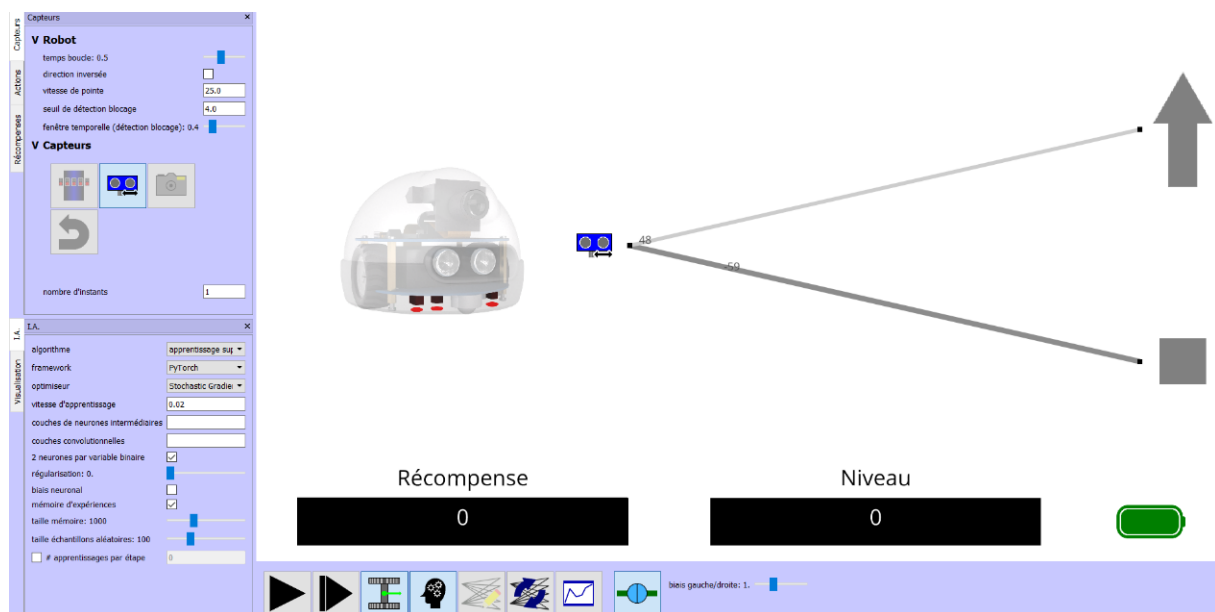
DETECTION D'INTRUS

Dans le cas précédent nous avons utilisé des couches de neurones intermédiaires pour traiter des informations très complexes. Ici nous allons voir un cas plus simple pour comprendre à quoi servent ces neurones intermédiaires.

Chargez la configuration « detection d'intrus ».

Ou pour sélectionner vous-mêmes les paramètres, lancez le programme puis :

- Dans l'onglet IA :
 - sélectionnez « apprentissage supervisé »
 - décochez biais neuronal
 - cochez mémoire d'expérience
- Dans l'onglet Capteurs, sélectionnez :
 -  infrason distance obstacle
- Dans l'onglet Actions, sélectionnez
 -  « tout droit »
 -  « stop »



-Placez le robot de façon à ce qu'il puisse faire tourner ses roues sans bouger.

-Placez le robot face à un mur ou à un objet de côté plat similaire (une boîte par exemple). Laissez de l'espace entre le robot et le mur.

-On veut indiquer au robot que cette situation est normale ; démarrez  le robot et appuyer plusieurs fois sur  (l'icône sur la droite)

L'intrus que le robot devra détecter sera symbolisé par un objet de côté plat, par exemple une boîte. Vous pouvez aussi utiliser votre main si vous n'avez rien d'autre.

-Placez l'intrus entre le robot et le mur, bien parallèle au mur, puis appuyez plusieurs fois sur .

-Enlevez l'intrus.

Le robot va continuer de faire tourner ses roues alors qu'il ne devrait pas. Pour régler ce problème, il faut ajouter un biais neuronal.

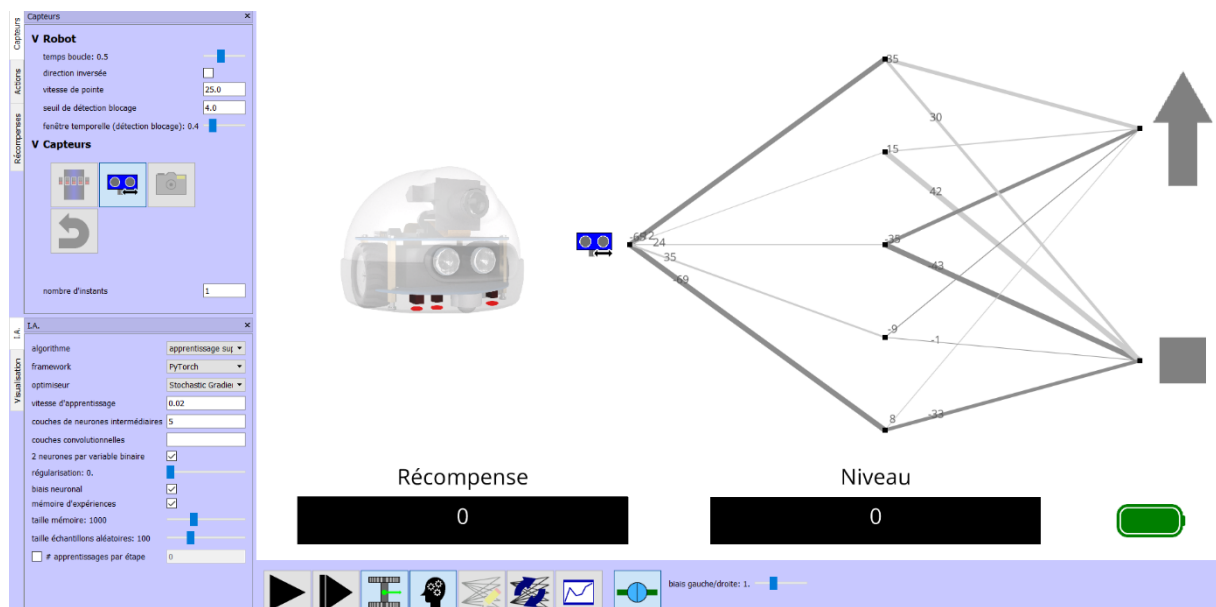
-Dans la fenêtre IA, cochez biais neuronal

-Réentraînez le robot comme précédemment.

-Enlevez l'intru et remplacez-le, toujours bien parallèle au mur ; le robot devrait faire tourner ses roues lorsqu'il voit l'intru et arrêter lorsqu'il n'est pas là.

-Une fois le robot bien entraîné, placez l'intru entre le robot et le mur, mais cette fois en biais. Cette fois, le robot n'arrive pas à le reconnaître.

L'intrus étant en biais, les rayons ultrasons sont déviés au lieu d'être directement renvoyés au robot. Pour régler ce problème, le robot doit apprendre que si la distance calculée avec les ultra-sons est soit trop longue, soit trop courte, alors c'est qu'il y a un intru. Pour prendre en compte ce raisonnement plus complexe, il faut rajouter une couche de neurones intermédiaires.



-Dans la fenêtre IA, dans couches de neurones intermédiaires, tapez 5.

-Apprenez au robot à bien se comporter :

- Lorsqu'il n'y a pas d'intru, appuyez plusieurs fois sur 
- Lorsqu'il y a un intru, que ce soit parallèle au mur ou en biais, appuyez plusieurs fois sur 

-Après l'avoir entraîné, le robot a le comportement attendu.

Notions apprises :

Ajouter des neurones intermédiaires sert à traiter les informations de façon plus approfondie, et notamment pour les traiter avec des fonctions linéaires.

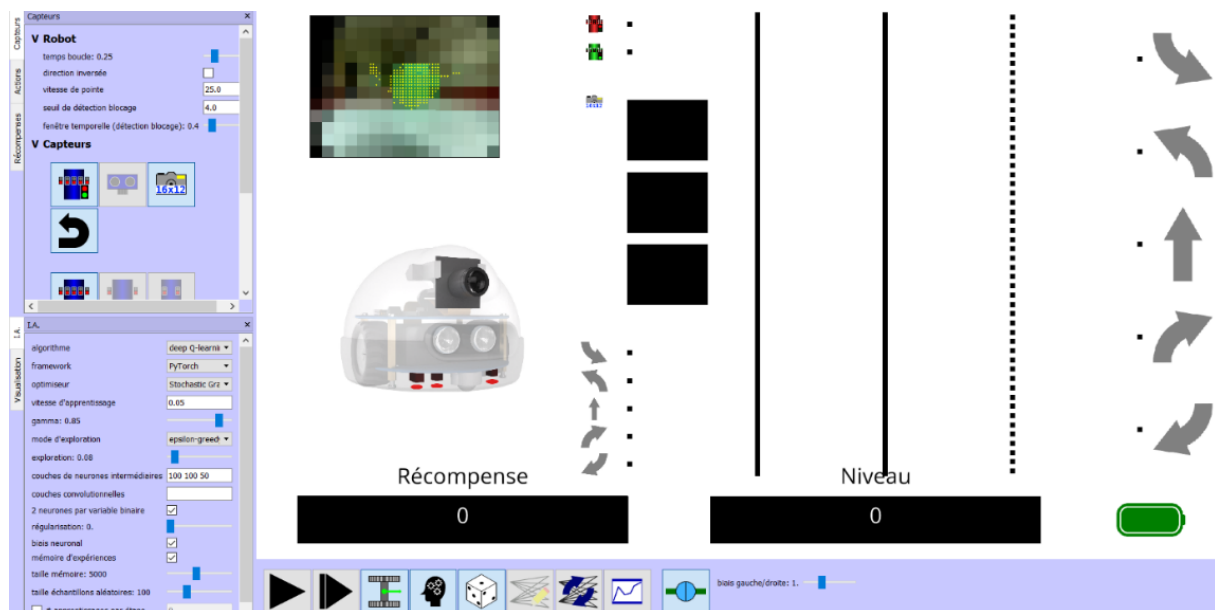
SUIVI D'UN BALLON

Voici un autre exemple d'apprentissage par renforcement. En changeant le type de récompense et en plaçant un nouvel élément dans l'arène, on peut réussir à faire adopter un comportement complètement différent au robot.

Chargez la configuration « suivi d'un ballon (vert) ». Ou pour sélectionner vous-mêmes les paramètres, lancez le programme puis :

- Dans l'onglet IA,
 - sélectionnez « deep Q-learning » ;
 - dans « couches de neurones intermédiaires » mettez « 100 100 50 »,
 - cochez mémoire d'expérience
- Dans l'onglet Capteurs, sélectionnez
 -  « bloqué/en mouvement »
 -  « caméra » (par exemple 32x24)
- Dans l'onglet Récompenses :

- choisissez « couleur dans l'image le robot ne doit pas être bloqué » 



-Placez un ballon de couleur verte dans l'arène (vous pouvez utiliser une autre couleur, mais dans ce cas il vous faudra modifier les paramètres de détection de couleur dans l'onglet Récompenses).

-Vérifiez que le réglage de la couleur du ballon est correct en plaçant le ballon devant le robot ; des points devraient apparaître sur l'image de la caméra au niveau du ballon.

-Dans l'onglet récompense, variez la teinte et la luminosité pour faire correspondre la couleur à celle du ballon

-Démarrez  le robot. Au bout d'un moment (environ 20 minutes), le robot a se mettra à poursuivre le ballon.

Le robot essaye de maximiser le nombre de pixel de la couleur du ballon sur sa caméra : il va donc apprendre à s'approcher du ballon le plus près possible, mais comme en s'en approchant il tape dedans, il se met de nouveau à s'en rapprocher ainsi de suite.

Notions apprises :

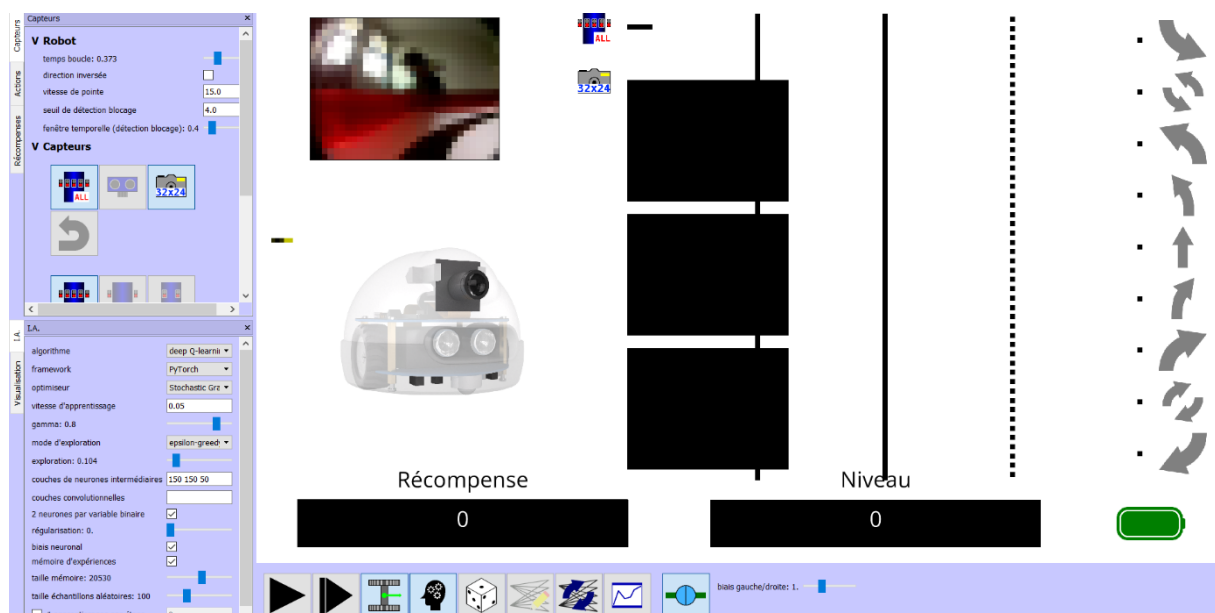
Lors de l'apprentissage par renforcement, pour donner un comportement précis au robot, **il faut bien réfléchir à la récompense** ; de façon à ce qu'elle récompense le robot uniquement lorsqu'il a le comportement souhaité.

SUIVI DE LIGNE

Voici un autre exemple d'apprentissage par renforcement. En changeant le type de récompense et en plaçant un nouvel élément dans l'arène, on peut réussir à faire adopter un comportement complètement différent au robot.

Chargez la configuration « suivi de ligne ». Ou pour sélectionner vous-mêmes les paramètres, lancez le programme puis :

- Dans l'onglet IA,
 - choisissez « deep Q-learning » ;
 - dans « couches de neurones intermédiaires » mettez « 150 150 50 »
 - cochez mémoire d'expérience
- Dans l'onglet Actions,
 - mettez toutes les actions possibles
- Dans l'onglet Capteurs, sélectionnez
 -  « caméra » (par exemple 32x24)
 -  « capteur IR de suivi de ligne - signal de chacun des capteurs »
- Dans l'onglet récompense :
 - choisissez « suivi de ligne » 



Placez du scotch noir dans l'arène pour tracer une boucle (si le scotch n'est pas large, n'hésitez pas à doubler ou tripler la largeur de la piste)

Démarrez  le robot. Au bout d'un *long* moment (environ 2-3 heures) le robot suivra la piste

TABLE RECAPITULATIVE DES SCENARIOS

Fiche n°	Titre	Difficulté	Algorithme d'apprentissage	Capteurs	Concepts	Durée de l'apprentissage	Durée de l'animation	Particularités
1	Prise en main avec le robot simulé	*	par renforcement	bloqué/en mouvement	comprendre sommairement ce qu'on voit à l'écran	2~3 min	1-5 min	robot simulé
2	Prise en main avec le vrai robot	*	par renforcement	bloqué/en mouvement	connexion avec le robot	5 min	1-5 min	mettre en place l'arène !
3	Reconnaissance d'image avec caméra	*	supervisé	caméra	apprentissage supervisé (phases d'entraînement et de test)	immédiat	1-5 min	
4	Navigation avec caméra	***	supervisé	bloqué/en mouvement, caméra	importance de la qualité des données d'entraînement	5~10 min pour l'entraînement, apprentissage immédiat	5-10 min	il faut bien conduire
5	Edition manuelle	**	pas d'apprentissage	bloqué/en mouvement (ultrason)	connexions dans le réseau de neurones	-	5-10 min	
6	Apprentissage par renforcement bloqué/ en mouvement	*	par renforcement	bloqué/en mouvement	Principe de l'apprentissage par renforcement	2~3 min	5-10 min	Le robot retrouve le même réseau que celui qu'on a fait à la main précédemment
7	Apprentissage par renforcement avec camera	*	par renforcement	bloqué/en mouvement, caméra	Utilisation de l'apprentissage dans un cas plus complexe	10~15 min	10-15 min	
8	Détection d'intru	**	supervisé	ultrason	Importance des couches de neurones intermédiaires	2~3 min	5-10 min	
9	Suivi ballon	*	par renforcement	bloqué/en mouvement, caméra	Importance de la récompense choisie pour l'apprentissage par renforcement	15~20 min		Le robot risque de pousser le ballon hors de l'arène, il faudra le surveiller
10	Suivi de ligne	*	par renforcement	bloqué/en mouvement, caméra, capteur de ligne au sol	Importance de la récompense choisie pour l'apprentissage par renforcement	2~3 h		Il vaut mieux lancer le robot en début de séance et faire le reste avec un autre robot en attendant qu'il s'entraîne