REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mohamed Khider – BISKRA
Faculté des Sciences Exactes, des Sciences de la Nature et de la Vie
Département d'informatique

N° d'ordre : iva02/M2/2021

Mémoire

Présenté pour obtenir le diplôme de master académique en **Informatique**

Parcours : **Image et Vie Artificielle (IVA)**

Utilisation du deep learning pour concevoir des controleurs pour robots à pattes

---

Par :

LIANI Somia

Soutenu le 26/06/2022 devant le jury composé de :

CHERIF Foudil	Professeur	Président
AKROUR Djouher	MCB	Rapporteur
ABABSA Tarek	MCB	Examinateur

Année universitaire 2021-2022

Remerciement

Avant tout, je remercie dieu le tout puissant de m'avoir donné la force, le courage, la santé, et la patience pour pouvoir accomplir ce travail.
J'exprime toute ma gratitude et mes profonds remerciements à mon encadrante madame **AKROUR DJOUHER**, pour cette chance de travailler avec lui, d'être présent avec moi à chaque fois que j'ai besoin de ses conseils, et de me diriger vers le bon chemin, et pour son aide afin d'élaborer mon projet de fin d'étude.
J'exprime toute ma gratitude aux membres de jury d'avoir accepté d'examiner ce travail, et pour leur suggestions et remarques permettant d'améliorer ce travail
ET enfin, Ce travail témoigne de mon profond respect
A ma petite famille pour sa confiance, ses encouragements, son soutien, son aide, sa compréhension, et ses sacrifices durant toute ma vie et particulièrement durant mon parcours d'étude

à l'honneur de ma mère **Salma** et mon père **AbdelHakim.**

Somia Liani.

Résumé

Le robot quadrupède est l'un des types de robots les plus attrayants du fait de sa locomotion intéressante qui permet le parcours de longues distances. Cependant, plus le nombre de capteurs utilisés est grand plus il est difficile aux contrôleurs de contrôler le robot de manière efficace. Pour cela, nous proposons dans ce travail d'utiliser le Deep Learning comme contrôleur de robot qui est une approche basé sur les réseaux de neurones avec un grand nombre d'entrées. Pour tester notre approche nous avons simulé notre robot quadrupède en utilisant le simulateur de robots tridimensionnel Gaezbo.

Mot clés: apprentissage profond, robot à pattes, simulation 3D, locomotion

Abstract

The quadruped robot is one of the most attractive types of robots because of its interesting locomotion which allows the travel of long distances. However, the greater the number of sensors used, the more difficult it is for the controllers to control the robot effectively. For this, we propose in this work to use Deep Learning as a robot controller which is an approach based on neural networks with a large number of inputs. To test our approach we simulated our quadruped robot using the three-dimensional robot simulator Gaezbo.

Keywords: deep learning, legged robot, 3D simulation, locomotion

الملخص

يعد الروبوت رباعي الأرجل أحد أكثر أنواع الروبوتات جاذبية بسبب حركته المثيرة للاهتمام والتي تسمح بالسفر لمسافات طويلة. ومع ذلك ، فكلما زاد عدد أجهزة الاستشعار المستخدمة ، زادت صعوبة التحكم في الروبوت بشكل فعال على أجهزة التحكم. لهذا ، نقترح في هذا العمل استخدام التعلم العميق كجهاز تحكم آلي وهو نهج يعتمد على الشبكات العصبية مع عدد كبير من المدخلات. لاختبار نهجنا ، قمنا بمحاكاة الروبوت الرباعي باستخدام جهاز محاكاة الروبوت ثلاثي الأبعاد Gaezbo.

الكلمات المفتاحية: التعلم العميق ، الروبوت ، المحاكاة ثلاثية الأبعاد ، الحركة

Table des matières

Chapitre 02 :Deep Learning en robotique

Liste de Figures

liste de tableaux:

Introduction Générale

La robotique est un domaine qui englobe la conception, la fabrication et la réalisation de robots ou de machine automatiques. Il réunit des connaissances scientifiques venues de déférents domaines informatique, électronique et mécanique. Un des enjeux de la robotique a toujours été d'améliorer l'interaction entre les humains et les robots. La robotique produit des automates réalisant des fonctions précises sur des chaînes de montage, outre industrie.

On classe les robots en deux catégories principales : les robots à base fixe ou bras manipulateurs et les robots mobiles. Parmi les robots mobiles, on s'intéresse aux robots marcheurs précisément les robots quadrupèdes qui utilisent leurs pattes comme moyens de locomotion.

La simulation est un outil important quand on travaille dans la robotique. Elle fait gagner du temps et permet de tester des scénarios difficiles à se réaliser en vrai. En plus on peut entraîner des algorithmes d'apprentissage. On va donc simuler notre robot et non le concevoir en monde réel, et pour cela on utilise le simulateur Gazebo qui est très important pour la simulation dans un monde en trois dimensions en temps réel. On utilise un contrôleur pour le robot qui est le réseau de neurone de la famille de deep learning c'est une méthode très utilisée dans le temps actuel donc pourquoi pas ne pas l'essayeravec notre robot.

Le deep learning est très utilisé ces dernière années dans différents domaine. Il est une technologie d'intelligence artificielle correspond à une assimilation automatique d'informations par une machine en temps réel les données utiles sont identifiés et intégrée dans un processus d'apprentissage profond sans aucune intervention humaine. Comme le robot a besoin de beaucoup d'informations depuis son environnement pour qu'il puisse bien agir et réaliser ses objectifs, il est nécessaire de choisir un contrôleur qui pourra gérer ce grand nombre d'information issues des capteurs. En fait, c'est ça notre problématique et nous proposons donc d'utiliser le Deep Learning.

Le mémoire est organisé comme suit. Dans le premier chapitre, nous décrivons le domaine de la robotique et ses objectifs puis nous présentons les concepts de capteurs et de contrôleurs ,Nous allons

par la suite présenter les simulateurs robotique, plus précisément Gazebo le simulateur que nous allons utiliser dans notre travail.

Le deuxième chapitre est consacré à la description de deep learning, ensuite nous allons présenterses différentes méthodes et quelques domaines d'applications dont il est utilisé.

Le troisième chapitre sera consacré à la conception détaillée de notre système qui est l'environnementde simulation et les capteurs utilisés, aussi comment contrôler ce modèle et aussi le système d'apprentissage que nous avons utilisé dans ce travail. Dans ce chapitre nous parlerons ausside l'implémentation de notre projet, ainsi que l'évaluation des résultats obtenus. Nous terminerons par une conclusion générale et quelques perspectives.

Chapitre 1: Robotique et Simulation

1. Introduction

La robotique est une des réalités fascinantes de l'histoire de l'humanité. Elle permet de rendre un grand service à l'humanité en réalisant des tâches répétitives, pénibles et parfois dangereuses de manière automatique et précise. Nous allons dans ce chapitre définir le concept de la robotique et ses objectifs. Nous allons par la suite présenter les simulateurs robotique, plus précisément Gazebo le simulateur que nous allons utiliser dans notre travail.

2. La robotique

La robotique est l'ensemble des techniques permettant la conception et la réalisation des machines automatiques ou des robots. Cette discipline concerne l'ensemble des domaines scientifiques et industriels en rapport avec la conception et la réalisation des robots.

L'utilisation des robots en générale joue un rôle très important dans le secteur industriel et aussi de plus en plus dans les secteurs de services comme l'entretien, l'exploration et la médecine. Les structures, formes et fonctionnalités de ces robots doivent être adaptées à l'environnement avec lequel ils interagissent. On peut classer les robots en deux catégories principales : les robots à basefixe (Bras manipulateurs) et les robots mobiles.

Pour revenir un peu plus loin dans l'histoire, on peut dire que les ancêtres des robots ce sont les automates, les automates mécaniques, qui ont évolué après vers les automates programmables.

3. Objectifs de la robotique

La robotique fournit un banc d'essai expérimental pour étudier les relations entre les morphologies neurologiques et la complexité environnementale qui serait difficile, voire impossible, à réaliser avec des organismes biologiques.

Les robots sont faits pour effectuer des tâches de complexité variable dans des environnements plus ou moins maîtrisés.

Les robots permettent d'apprendre ou de développer des connaissances en programmation, électronique, mécanique et en stratégie robotique.

Il peut tout à fait remplacer l'Homme dans la réalisation des tâches rébarbatives ou génératrices

un robot est possible de déplacer dans toutes les directions.

les robots pourraient être la meilleure solution pour les rôles de décontamination après un accident nucléaire ou dans la vie de tous les jours, dans les centrales nucléaires.
Tout simplement, ils pourraient aussi servir pour des tâches beaucoup plus ordinaires comme réparer, souder des objets, en transporter ou faire des photos.[1]

4. Composition d'un robot

4.1. Morphologie

Un robot est un assemblage complexe de pièces mécaniques et de pièces électroniques, le tout piloté par une intelligence artificielle. Lorsque les robots autonomes sont mobiles, ils possèdent également une source d'énergie embarquée : généralement une batterie d'accumulateurs électriques[2]

Un robot intelligent est un assemblage complexe de pièces électro-mécaniques (structure) et de pièces électroniques (cerveau), le tout pouvant être piloté par une intelligence artificielle.

Un robot est composé de quatre groupes de composantes principales:

1. Structure mécanique: C'est le squelette du robot. Le choix des articulations permettra de déterminer les mouvements possibles et d'orienter son type d'utilisation. On compte un degré de liberté pour chaque axe de rotation selon lequel un membre (bras, patte) peut se mouvoir.

2. Servo-moteurs (actionneurs): Le servo-moteur permettra au robot d'effectuer concrètement les actions commandées par son cerveau, en fonction de ses objectifs et suite au traitement des informations recueillies par les capteurs. Les actionneurs sont des organes qui, sous commande, transforment l'énergie qui leur est fournie en actions physiques utilisables comme des mouvements.

3. Capteurs: Les capteurs s'occupent de la partie sensorielle du robot. Ils permettent d'acquérir des informations à propos de l'environnement du robot ou de ses composants internes.

4. Cerveau: Le cerveau du robot permet d'analyser les données provenant des capteurs et d'envoyer les ordres relatifs aux servo-moteurs. La partie commande est matérialisée physiquement par un microcontrôleur, qui est un cerveau électronique spécialement conçu pour l'interagir avec des capteurs et des actionneurs. [3]

Il y a plusieurs types de robots. Nous citons par exemple:

- **Le robot serpent:** les robots de ce type peuvent grimper aux arbres, monter à l'échelle ou nager en sous-marin etc. Ils ont comme objectif pour sauver la vie car il peuvent accéder à des espaces exigus et très difficiles d'accès. L'un des exemples des robot serpents est modsnakes conçu par le laboratoire bio-robotique Carnegie Mellon. Il a été conçu pour la surveillance et la sécurité, il peut être utilisé pour surveiller l'intérieur d'un local clos, localiser des mines, désarmer une bombe etc.

- **Le robot à roues:** sont généralement utilisés pour transporter des charges plus élevés. Les roues sont la méthode la plus répandue pour apporter de la mobilité au robot.

C'est une technique assure selon l'agencement et les dimensions des roues un déplacement dans toutes les directions avec une accélération et une vitesse importantes .Le franchissement d'obstacles ou l'escalade de marches d'escaliers est possible.[4]

- **Robots à chenilles :** L'utilisation des robots à chenilles présente l'avantage d'une bonne adhérence au sol et d'une faculté de franchissement d'obstacles. L'utilisation est orientée vers l'emploi sur sol accidenté ou de mauvaise qualité.

- **Robots volants :** Un drone désignent un aéronef sans pilote a bord peut avoir un usage civil ou militaire. Les drones sont utilisés au profit des forces armées ou de sécurités.

- **Un robot multi-patte:** est un robot mobile marcheur à plusieurs pattes. Ils peuvent être bipède, quadrupède (4 pattes) ou hexapode (6 pattes). C'est un système mécatronique

combinant des notions mécaniques, électroniques et informatiques. Les robots hexapodes sont spécialement étudié pour leurs:

◦ sa stabilité : marche stables et efficaces gérée adéquatement par le polygone de support ;

◦ des applications terrestres : hexapode pourrait faire la collecte de données ou échantillons biologiques dans une forêt éloignée, fouiller le site dune catastrophe pour localiser les survivants

sans mettre en danger les sauveteurs, démantèlement de mines et pourrait aussi être utilisé comme robot domestique.

◦ Des applications spatiales peuvent être imaginées pour le robot hexapode : explorer Mars ou la Lune ou construire des bases planétaires.[4]

Figure 1.1: Les différents types des robots

4.2. **Présentation de capteurs**

Les robots peuvent être dotés de “sens”, c'est-à-dire d'un ensemble plus ou moins important d'instruments de mesure et d'appréciation caméra, thermomètre, télémètre, etc. permettant au programme du robot de décider du mouvement le mieux adapté aux conditions extérieures. Par exemple: si un robot mobile muni d'une caméra est amené à se déplacer dans un local inconnu, on peut le programmer pour qu'il contourne tout obstacle qui entraverait sa route [5]

4.4. Contrôleur

Un contrôleur est un programme qui retourne les actions du robot en fonction de ses perceptions afin de résoudre une tâche donnée. Les techniques de shaping visent à améliorer les résultats et la convergence des algorithmes évolutionnaires en considérant des tâches à difficulté croissante.

Le contrôle du robot inclut les fonctions qui lui permettent "d'apprendre" et d'être programmé pour une tâche spécifique, puis d'exécuter cette tâche. La séquence de mouvements, le type de mouvement entre deux points, et l'interaction avec les équipements externes sont toutes des parties de la fonction de contrôle. Pour effectuer ses différentes fonctions, le contrôleur doit avoir dans ses mémoires:

- un modèle du manipulateur: c'est à dire les relations entre les signaux d'excitation des actionneurs et les déplacements du manipulateur qui en sont la conséquence;

-un modèle de l'environnement: qui est une description de ce qui se trouve dans l'espace atteignable;

-des programmes: comprenant les données relatives à la tâche à exécuter ainsi qu'un certain nombre d'algorithmes de commande. [6]

Il existe différents algorithmes utilisés, nous pouvons ainsi citer :

1. **Réseaux de neurones (RN)** :

Les réseaux de neurones fonctionnent en répartissant les valeurs des variables dans des automates (les neurones). Ces unités sont chargées de combiner entre elles leurs informations pour déterminer la valeur du paramètre de discrimination. C'est de la connexion de ces unités entre elles qu'émerge la capacité de discrimination du RN. Chaque neurone reçoit des informations numériques en provenance de neurones voisins ; à chacune de ces valeurs est associé un poids représentatif de la

force de la connexion. Chaque neurone effectue localement un calcul dont le résultat est transmis ensuite aux neurones avals.

- **Réseaux de neurones évolutifs :**

Les réseaux de neurones évolutifs, souvent appelés réseaux de neurones constructifs, sont des réseaux de type compétitifs [7]. Initialement, la couche destinée au traitement est composée d'un nombre minimal de neurones, comme présenté sur la figure 8(a). Au cours de la phase d'apprentissage (voir figure 8(b)), les paramètres libres du réseau ainsi que les connexions inter-

neuronaux seront adaptés à la base d'apprentissage composée des stimuli représentant la majorité des exemples de l'espace d'entrée. A un certain niveau d'apprentissage, tel qu'illustré sur la figure1.2, de nouveaux neurones sont ajoutés selon les besoins de l'application [8]

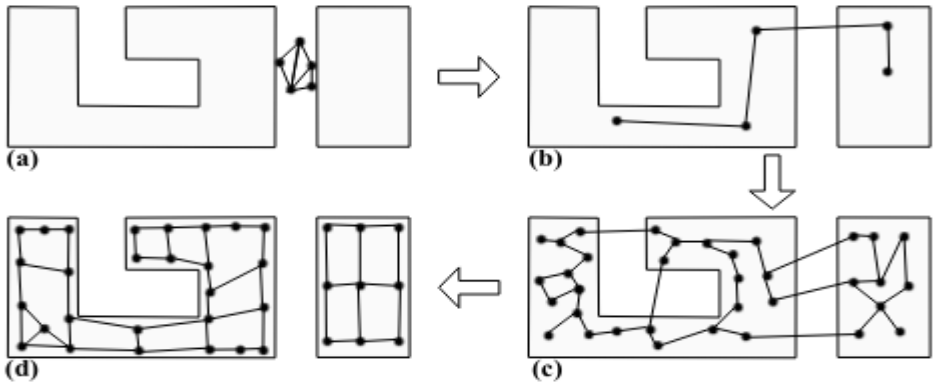


figure1.2 : Principe des réseaux de neurones évolutifs. (a) : Etat initial, (b) : Adaptation, (c) : Évolution du réseau, (d) : Adaptation finale.

- **Les réseaux de Kohonen**

Les réseaux de Kohonen sont classés dans la catégorie des réseaux de neurones compétitifs. A partir de son approche de mémoires associatives, Teuvo Kohonen a proposé la carte auto-organisatrice SOM en 1982 [9]. Après le succès de sa carte auto-organisatrice, Teuvo Kohonen a proposé un modèle adoptant un algorithme d'apprentissage supervisé. Ce nouveau modèle est appelé Learning Vector Quantization (LVQ) [10]. Dans cette version, l'algorithme d'apprentissage suit une règle d'adaptation locale. L'objectif de cet algorithme consiste à rapprocher le neurone gagnant de la position cartographique du stimulus en cas de succès. Dans le cas contraire, en cas

d'erreur, l'objectif est d'écarter le neurone en question de la zone des stimuli similaires de l'espace d'entrée. Les réseaux de Kohonen sont généralement utilisés pour des applications de classification. Dans ce cadre, le modèle LVQ a été présenté afin de faciliter l'opération d'étiquetage de la carte auto-organisatrice. D'autre part, les réseaux de Kohonen ont montré leurs performances pour des applications de reconnaissance de forme, de réduction de taille des données ainsi ils sont caractèrisés par une certaine robustesse aux données bruitées et/ou corrompues.

- **Robot quadrupède**

C'est un robot mobile marcheur à quatre pattes. C'est un système mécatronique Combinant des notions mécaniques, électroniques et informatiques.

Ces quadrupèdes déplacent chaque patte séparément. Il est plus délicat de faire avancer à la fois deux pattes opposées, car le polygone de sustentation devient étroit. Ces robots sont plus stables que les bipeds.

Le robot quadrupède permet de collecter des informations en données 3D pouvant être reliées à des jumeaux numériques
Il permet de faire des missions autonomes peuvent être programmées. il Peuvent embarquer des charges utiles, il est également capable d'effectuer des relevés topographiques.[11]

Figure 1.3 : exemple de robot quadrupède

6. **Simulateurs robotiques 3D**

Un simulateur robotique est utilisé pour créer une application pour un robot physique sans dépendre de la machine réelle, ce qui permet de gagner du temps et de l'argent. Dans certains cas, ces applications peuvent être transférées sur le robot physique (ou reconstruites) sans modifications.
Le terme simulateur de robotique peut désigner plusieurs applications de simulation robotique différentes. Par exemple, dans les applications de robotique mobile, les simulateurs de robotique basés sur le comportement permettent aux utilisateurs de créer des mondes simples d'objets rigides et de sources de lumière et de programmer des robots pour interagir avec ces mondes. La simulation basée sur le comportement permet des actions de nature plus biologique par rapport aux simulateurs plus binaires ou informatiques. De plus, les simulateurs comportementaux peuvent « apprendre » des erreurs et sont capables de démontrer la qualité anthropomorphique de la ténacité.[6]

6.1. Intérêt

Le simulateur a pour but de reproduire de façon virtuelle le comportement du rover, il pourra donc recevoir en temps réel les commandes envoyées par le superviseur

Le simulateur 3D est l'outil logiciel qui permet d'assembler et de tester l'environnement de production avant sa mise en marche.

Il y a une foule d avantages qui peuvent se retrouver dans un simulateur 3D par exemple :

Le simulateur rend possible la modélisation et la planification de la cellule robotique complète en 3D (environnement de simulation 3D)

Le simulateur rend possible la programmation hors ligne de robots par opposition à la programmation en mode point à point. (mise en service plus rapide)

Un des points importants d'un simulateur 3D est la capacité de calculer le temps de cycle nécessaire à l'application.

La simulation 3D permet de réduire les risques entourant la programmation et la mise en production d'un système robotique.

Il peut comporter un contrôleur virtuel, un boitier de commande virtuelle (intéressant pour la formation du personnel), un éditeur de programme en ligne, un simulateur d'entrées/sorties et un simulateur de mécanismes. [11]

6.2. Composition d'un simulateur

1. Moteur physique

un composant important, car il influence grandement la performance et la scalabilité du simulateur, ainsi que sa capacité à supporter des structures complexespour une génération de mouvement plus réaliste du robot. L'utilisation d'un simulateur robotique pour le développement d'un programme de contrôle robotique est fortement recommandée, qu'un robot réel soit disponible ou non. C'est une bibliothèque indépendante qu'on peut intégrer dans des environnements de simulation et qui sert à résoudre les équations de la physique et les problèmes de la mécanique classique, tel que les collisions, les forces et la cinétique.

Moteurs physiques pour des mouvements réalistes. La plupart des simulateurs utilisent Bullet , ODE ou PhysX .[12]

2. **Interface de visualisation**

Un simulateur robotique est utilisé pour créer une application pur un robot physique , ce qui permet de gagner du temps et de l'argent. L'une des applications les plus courantes des simulateurs de robotique est la modélisation 3D et le rendu d'un robot et son environnement.
Il existe de nombreux types de simulateurs de contrôle disponibles qui offrent divers degrés de difficulté et de réalisme, on cite :

- **Morce :**

il s'appuie sur une architecture basée sur des composants pour simuler des capteurs, des actionneurs et des robots ; il est flexible, capable de spécifier des simulations à des niveaux d'abstraction variables en fonction des systèmes testés; il est capable de représenter une grande variété de robots hétérogènes et d'environnements 3D complets (aériens, terrestres, maritimes); et il est conçu pour permettre des simulations de plusieurs systèmes de robots.
MORSE est construit sur Blender, en utilisant ses puissantes fonctionnalités et en étendant ses fonctionnalités grâce à des scripts Python. Les simulations sont exécutées sur le mode Moteur de jeu de Blender, qui fournit un affichage graphique réaliste des environnements simulés et permet d'exploiter le réputé moteur physique Bullet. [13]

- **V-Rep :**

V-REP, le simulateur de Coppelia Robotics, se définit comme le couteau suisse de la simulation robotique. C'est un logiciel Open Source, gratuit pour une utilisation non commerciale et assez récent. Il est toutefois déjà très abouti et a comme grand intérêt d'être très facilement interconnectable avec d'autres applicatif et notamment avec ROS.
V-REP propose un mode de fonctionnement en temps-réel. Cela signifie que la simulation est optimisée pour respecter des contraintes de temps-réel et permet d'avoir une simulation fidèle à la réalité, utilisable en combinaison avec un système réel [14]

- **Webost :**

Webots est une application de bureau open source et multiplateforme utilisée pour simuler des robots. Il fournit un environnement de développement complet pour modéliser, programmer et simuler des robots. Il a été conçu pour un usage professionnel, et il est largement utilisé dans l'industrie, l'éducation et la recherche. Cyberbotics Ltd. maintient Webots comme son produit principal de manière continue depuis 1998.

Webots est robuste, déterministe et bien documenté. Pour garantir la qualité du code, chaque modification du code est examinée par des pairs et soumise à une suite de tests automatique testant toute l'API. La rétrocompatibilité est garantie et bien documentée entre les versions majeures. Chaque rejet est évalué par des tests d'assurance de la qualité effectués par des humains [15]

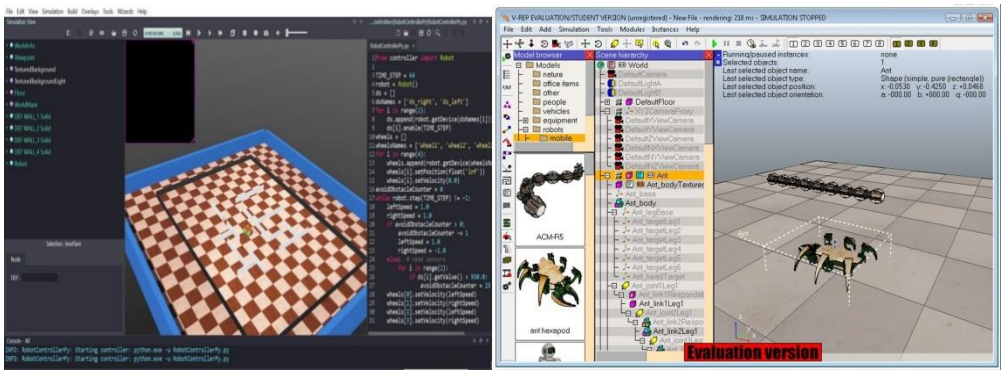


Figure 1.4 : les différents simulateurs robotique

Dans notre projet nous choisissant le simulateur Gazebo

7. Gazebo

Gazebo est un leader de la simulation de robots. C'est un outil sur lequel s'appuient des centaines de milliers d'utilisateurs et de développeurs à travers le monde.

Une simulation Gazebo est une simulation de robot réalisée avec Gazebo, un simulateur 3D capable de simuler avec précision et efficacité des populations de robots dans des environnements intérieurs et extérieurs complexes.

Il y a les principaux composants suivants:

- **Fichier "word"** : il contient tous les éléments d'une simulation, y compris les robots, les lumières, les capteurs et les objets statiques. Dans l'image ci-dessus, une simulation est montrée dans l'application Gazebo.

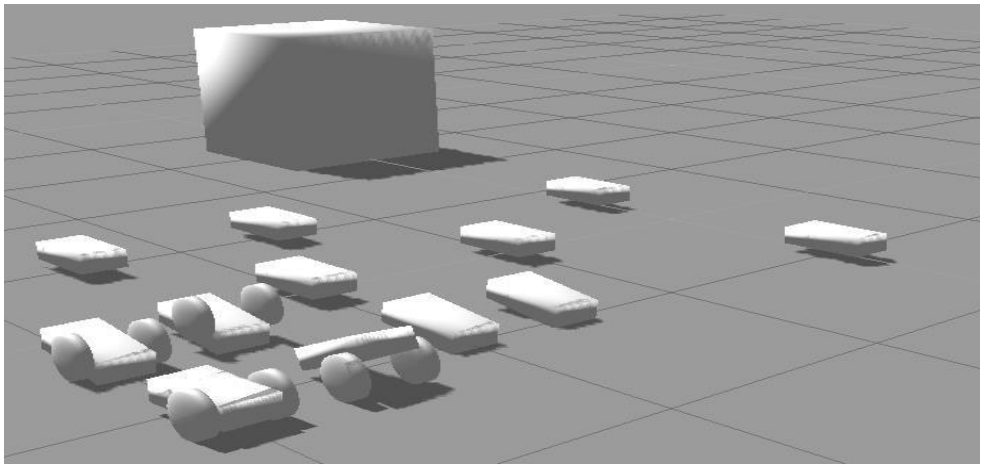


figure1.5: simulation sous Gazbo

- **Fichier "model" :** représente les éléments du robot individuels. Les trois robots et l'objet devant eux sont des modèles.
- ***gzserver*** – le programme Gazebo « cheval de travail » – il lit le fichier monde pour générer et peupler un monde.
- ***gzclient*** – se connecte à gzserver et visualise les éléments. Dans la sortie shell, sous « NODES », nous avons les deux et répertoriés. gzserver gzclient
- ***gzweb*** – version web de gzclient, utilisant WebGL. [16]
-
- **Plugin**: est un morceau de code (C++) qui est compilé en tant que bibliothèque partagée pour modifier la simulation en temps réel.

Ce code peut être attaché à des modèles, des capteurs, à l'environnement ou au simulateur lui-même. Par exemple, c'est un modèle qui permettre de contrôler les jointures d'un robot. Les plugins fournissent aux utilisateurs un accès direct aux propriétés physiques des « models » et aux bibliothèques de Gazebo via des classes standard C++.

9. Conclusion

Ce premier chapitre fournit une présentation générale sur les robots, ainsi qu'une présentation du concept des simulations 3D. Nous nous sommes focalisés sur les robots quadrupèdes dont la tâche principale est la locomotion. Dans le chapitre suivant, nous allons présenter le deep learning et son application comme contrôleur de robots.

Chapitre 2 : Deep Learning en robotique

1. Introduction

Au sein de la révolution que connaît l'Intelligence Artificielle aujourd'hui, l'apprentissage profond (deep learning) occupe une place de tout premier plan. Il est au cœur des technologies qui permettent aujourd'hui de réaliser des tâches encore impensables. Récemment il est utilisé comme contrôleurs de robots car il permet de gérer un flux important de données issues de capteurs. Nous allons donc dans ce chapitre décrire le principe de deep learning, le comparer avec les réseaux de neurones classiques et décrire comment se fait l'apprentissage.

2. Deep learning

Le Deep learning (ou Apprentissage profond, en français) est une forme d'I.A dérivée du Machine Learning, littéralement traductible par "une machine capable d'apprendre". Au cours du 20e siècle, différentes techniques de Machine Learning ont donc vu le jour pour apprendre et s'améliorer continuellement et de manière autonome.

L'apprentissage profond (Deep Learning DL en anglais) est un type d'apprentissage automatique utilisé pour chercher à modéliser des données. Il est beaucoup utilisé dans la classification supervisée.

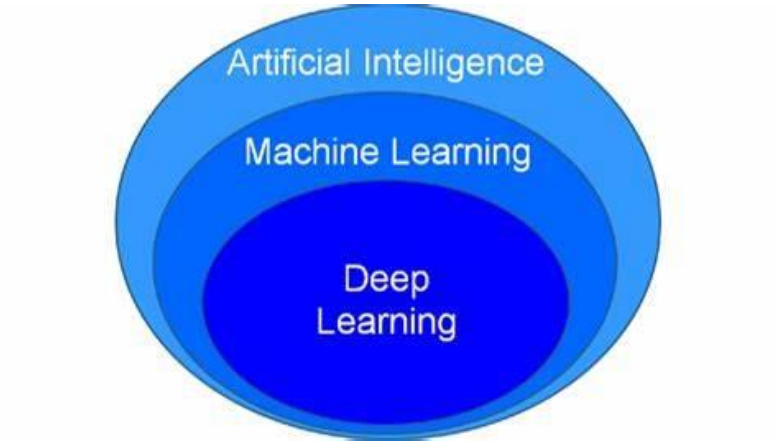


Figure 2.1: Paradigmes de l'intelligence artificielle.

Ce type utilise des réseaux de neurones artificiels pour effectuer des calculs sophistiqués sur de grandes quantités de données. D'ailleurs, plus il y a des données, plus le réseau est performant.

Un réseau neuronal est structuré comme le cerveau humain, et se compose de neurones artificiels, également appelés nœuds. Ces nœuds sont empilés les uns à côté des autres en trois couches [17]:

- **La couche d'entrée :** représente des caractéristiques des exemples (où de l'exemple à prédire).
- **La(s) couche(s) caché(s)** : ce sont des couches intermédiaires.
- **La couche de sortie :** le résultat de la prédiction, classification, etc.

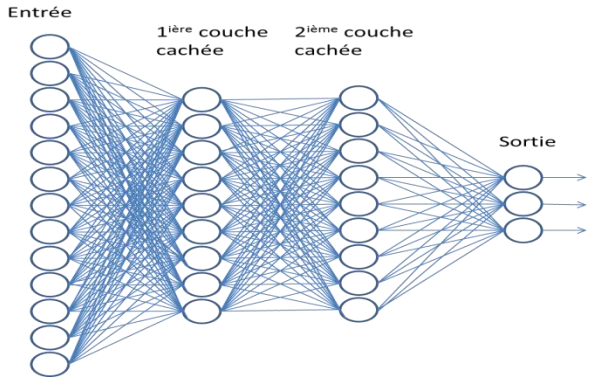


Figure 2.2: Réseau de neurone multicouche [18]

2.1. Les réseaux de neurones artificiels

2.1.1. Neurone

Un neurone est une fonction algébrique non linéaire et bornée, dont la valeur dépend des paramètres appelés coefficients ou poids. Les variables de cette fonction sont habituellement appelées « entrées » du neurone, et la valeur de la fonction est appelée sa « sortie ».

Un neurone est donc avant tout un opérateur mathématique, dont on peut calculer la valeur numérique par quelques lignes de logiciel.

2.1.2. Perceptron

Un perceptron est une unité de réseau de neurones. Il effectue des calculs pour détecter des caractéristiques ou des tendances dans les données d'entrée. Il s'agit d'un algorithme pour l'apprentissage supervisé de classificateurs binaires. C'est cet algorithme qui permet aux neurones artificiels d'apprendre et de traiter les éléments d'un ensemble de données.

Son fonctionnement repose sur des opérations de multiplication entre deux composants importants : les entrées de données (input) et le poids.La somme de cette multiplication est transmise à une fonction d'activation, déterminant une valeur binaire de 0 ou 1. C'est ce qui permet de classifier les données. [19]

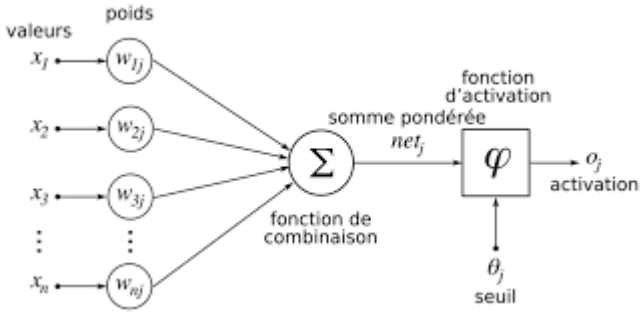


figure2.3 : un perceptron

2.1.3. **Fonction d'activation**

La fonction d'activation est une fonction mathématique appliquée à un signal en sortie d'un neurone artificiel. Le terme de "fonction d'activation" vient de l'équivalent biologique "potentiel d'activation", seuil de stimulation qui, une fois atteint entraîne une réponse du neurone. La fonction d'activation est souvent une fonction non linéaire, déterminant une valeur binaire de 0 ou 1. C'est ce qui permet de classifier les données.

2.2. **Types des réseaux de neurones utilisés en Deep Learning**

Il existe différents algorithmes de Deep Learning. Nous pouvons ainsi citer :

2.2.1. **Les réseaux de neurones récurrents (RNN ou Recurrent Neural Networks).**

Les humains ne commencent pas leur réflexion à partir de zéro chaque seconde. En lisant cet essai, vous comprenez chaque mot en fonction de votre compréhension des mots précédents. Vous ne jetez pas tout et recommencez à penser à partir de zéro. Vos pensées ont de la persévérance. Les réseaux de neurones traditionnels ne peuvent pas le faire, et cela semble être une lacune majeure. Par exemple, imaginez que vous souhaitiez classer le type d'événement qui se produit à chaque étape d'un film. On ne sait pas comment un réseau de neurones traditionnel pourrait utiliser son raisonnement sur des événements antérieurs dans le film pour les informer plus tard.

Au cours des dernières années, il y a eu un succès incroyable en appliquant les RNN à une variété de problèmes: la reconnaissance de la parole, la modélisation du langage, la traduction, le sous-titrage des images, etc. [20]

2.2.2. **Les réseaux de neurones convolutionnels (CNN ou Convolutional Neural Networks)**

Les réseaux de neurones convolutifs sont à ce jour les modèles les plus performants pour classer des images. [21] Désignés par l'acronyme CNN, de l'anglais Convolutional Neural Network, ils comportent deux parties bien distinctes. En entrée, une image est fournie sous la forme d'une matrice de pixels. Elle a 2 dimensions pour une image en niveaux de gris. La couleur est représentée par une

troisième dimension, de profondeur 3 pour représenter les couleurs fondamentales [Rouge, Vert, Bleu].

La première partie d'un CNN est la partie convolutive à proprement parler. Elle fonctionne comme un extracteur de caractéristiques des images. Une image est passée à travers une succession de filtres, ou noyaux de convolution, créant de nouvelles images appelées cartes de convolutions. Certains filtres intermédiaires réduisent la résolution de l'image par une opération de maximum local. Au final, les cartes de convolutions sont mises à plat et concaténées en un vecteur de caractéristiques, appelé code CNN.

2.2.3. La machine de Boltzmann profonde (DBN ou Deep Belief Network)

DBN est un réseau de neurones constitué de plus d'une machine Boltzmann restreinte (RBM). RBM et DBN ont été conçus par Geoffrey Hinton. [22] RBM est un algorithme utile pour la réduction des dimensions, la régression, le filtrage collaboratif, classification, apprentissage de fonctionnalités et modélisation de thèmes. C'est un réseau de neurones à deux couches peu profond composé d'une couche d'entrée visible et une couche de sortie cachée. Ils sont les blocs de construction de DBN. Pour chaque RBM la couche visible représente les inputs et la couche cachée reçoit les inputs de la couche visible, fait les calculs ensuite elle envoie les résultats à la couche visible. Dans la DBN chaque couche cachée du RBM représente la couche visible de la RBM suivante comme le montre la figure ci dessous. [23]

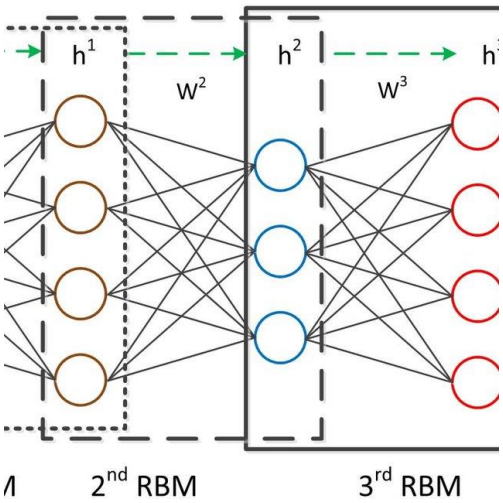


Figure2.4: DBN

Les DBN ont été utilisés pour générer et reconnaître des images (reconnaissance faciale), des séquences vidéo et des données de capture de mouvement. La NASA utilise DBN pour classer des téraoctets d'imagerie satellitaire à haute résolution et très diversifiée

3. **Réseaux de neurones VS Deep Learning**

Le tableau ci-dessous représente un comparatif entre les réseaux de neurones artificiels et l'apprentissage profond

Base de comparaison	**Les réseaux de neurones**	**L'apprentissage en profondeur**
Définition	Classe d'algorithmes d'apprentissage automatique où le neurone artificiel forme l'unité de calcul de base et les réseaux sont utilisés pour décrire l'interconnectivité entre eux	Il s'agit d'une classe d'algorithmes d'apprentissage automatique qui utilise plusieurs couches d'unités de traitement non linéaires pour la transformation et l'extraction de fonctionnalités. Il représente également des concepts dans des modes hiérarchiques multiples qui correspondent à différents niveaux d'abstraction.
Composants	**Neurones:** le neurone qui est étiqueté j reçoit une entrée des neurones prédécesseurs souvent sous la forme d'une fonction d'identité pour fournir une sortie. **Connexions et poids:** La connexion est une composante vitale entre le neurone de sortie i et le neurone d'entrée j. Chaque connexion est alors identifiée par un poids ij.	**Carte mère:** le chipset de la carte mère est un composant lié à l'apprentissage en profondeur qui est particulièrement basé sur les voies PCI-e. **Processeurs** : le type de GPU requis pour le Deep Learning doit être basé sur le type de socket, le nombre de cœurs et le coût du processeur. **RAM, mémoire physique et stockage:** les algorithmes d'apprentissage en

	Fonction de propagation: Elle est utilisée pour fournir une entrée pour la sortie résultante. **Règle d'apprentissage:** Elle permet de modifier les paramètres du réseau neuronal de manière à aboutir à une sortie favorable.	profondeur nécessitent une grande utilisation du processeur, un stockage et une zone de mémoire et il est donc indispensable d'avoir un ensemble riche de ces composants. **PSU:** Avec l'augmentation de la mémoire, du CPU et de la zone de stockage, il devient également important d'utiliser une grande PSU suffisamment pour gérer une puissance énorme.
Architecture	**Réseaux de neurones à action directe:** le type d'architecture le plus courant contient la première couche comme couche d'entrée tandis que la dernière couche est la couche de sortie et toutes les couches intermédiaires sont les couches cachées. **Réseaux récurrents:** ce type d'architecture se compose de cycles dirigés dans le graphe de connexion. Les architectures biologiquement réalistes peuvent également vous ramener d'où vous avez commencé. Celles-ci sont compliquées à former et sont extrêmement dynamiques. **Réseaux connectés symétriquement:** architecture de maintien de connexion symétrique qui ressemble plus ou moins aux réseaux récurrents. Ils sont de nature restreinte en raison	**Réseaux pré-formés non supervisés:** Dans cette architecture, nous ne parlons d'aucune formation formelle mais les réseaux sont pré-formés en utilisant des expériences passées. Cela inclut les auto-encodeurs, les réseaux de croyances profondes et les réseaux contradictoires génératifs. **Réseaux de neurones convolutifs:** il vise à apprendre des caractéristiques d'ordre supérieur à l'aide de convolutions qui améliorent l'expérience utilisateur de reconnaissance et d'identification d'images. L'identification des visages, des panneaux de signalisation, des ornithorynques et d'autres objets devient facile en utilisant cette architecture. Réseaux de neurones récurrents: Ils

	de leur utilisation de la fonction énergétique. Les réseaux connectés symétriquement avec des réseaux cachés sont connus sous le nom de machines Boltzmann tandis que ceux sans réseau caché sont appelés réseaux Hopfield.	sont issus de la famille des feed-forward qui croient en l'envoi de leurs informations sur des pas de temps. **Réseaux de neurones récursifs:** il marque également des entrées de longueur variable. La principale différence entre récursif et récursif est que le premier a la capacité de périphérique les structures hiérarchiques dans le jeu de données d'apprentissage tandis que le second pose également les informations sur la façon dont cette structure hiérarchique est maintenue dans le jeu de données.

Tableau 1 : *Tableau présente réseaux de neurones vs apprentissage en profondeur [24]*

5. Avantages et inconvénients du Deep Learning

5.1. Avantages [25]

- **De meilleurs résultats qu'avec d'autres méthodes d'apprentissage automatique.**

Le plus grand point fort du Deep learning reste la qualité des résultats obtenus. Dans des secteurs tels que le traitement d'images ou la reconnaissance d'images, cette forme d'intelligence artificielle détrône toutes les autres.

- **Une exécution efficace des tâches de routine, sans écarts de qualité.**

Parce que basé sur un apprentissage routinier, ne montrant jamais aucun signe de fatigue et avec une qualité constante, celle-ci est beaucoup plus efficace et rapide que n'importe quelle autre méthode.
Puisque le système se forme de façon autonome (après une phase d'instruction initiale), il permet d'économiser beaucoup de temps et d'argent tout en garantissant un développement de ses fonctionnalités.

- **Le traitement des données non structurées.**

De plus, et contrairement à d'autres moteurs d'intelligence artificielle, l'apprentissage profond est capable d'analyser des données stockées sous un format non structuré (documents, photos, mails, etc.).
De ce fait, il a une force de frappe différente et potentiellement plus intéressante que les technologies limitées à l'analyse des données structurées (numéros de téléphone, carte de crédit, adresses, etc.)

5.2. Inconvénients [25]

- **Le Deep Learning nécessite une grande puissance de calcul.**

Si le Deep Learning a beaucoup d'avantages, il a aussi ses limites, parmi lesquelles un énorme besoin en puissance de calcul. D'une part pour assurer la maintenance des réseaux de neurones artificiels, mais aussi pour traiter la très grande quantité de données nécessaires.

- **Une technologie couteuse à mettre en place.**

Cette puissance de calcul est relative à la complexité et à la difficulté de la tâche à résoudre et de la masse de données utilisée. De ce fait, le Deep learning se révèle être un système artificiel coûteux, donc plutôt réservé à la recherche et aux géants du Big Data.

- **Des décisions difficilement ou pas du tout compréhensibles**

Sur un autre point, l'une des problématiques que pose cette intelligence artificielle est la complexité et le volume de données que requiert son fonctionnement et le fait qu'il est impossible de comprendre

dans le détail la motivation des décisions qu'elle prend. Ce problème entraîne avec lui l'incapacité (pour le moment) de l'intégrer à des applications où la traçabilité est essentielle

- **Il nécessite une vaste base de données**

Enfin, pour être efficace, le Deep Learning doit s'appuyer sur une grande quantité de données. Sans cela, aucune machine n'est en mesure de donner de bons résultats avec cette méthode.

Bien qu'il existe des bibliothèques de réseaux neuronaux artificiels mis à disposition de tous pour simplifier son utilisation, il existe bel et bien des limites à la mise en place d'une intelligence profonde, notamment le temps nécessaire à l'élaboration des algorithmes d'apprentissage.

6. **Domaines d'application du Deep Learning**

Des applications du Deep Learning sont utilisées dans divers secteurs, de la conduite automatisée aux dispositifs médicaux. [26]

- **Conduite automatisée :** Les chercheurs du secteur automobile ont recours au Deep Learning pour détecter automatiquement des objets tels que les panneaux stop et les feux de circulation. Le Deep Learning est également utilisé pour détecter les piétons, évitant ainsi nombre d'accidents.
- **Aérospatiale et défense :** Le Deep Learning sert à identifier des objets à partir de satellites utilisés pour localiser des zones d'intérêt et identifier quels secteurs sont sûrs ou dangereux pour les troupes au sol.
- **Recherche médicale :** À l'aide du Deep Learning, les chercheurs en cancérologie peuvent dépister automatiquement les cellules cancéreuses. Des équipes de l'Université de Californie à Los Angeles (UCLA) ont conçu un microscope qui génère un ensemble de données de grande dimension afin d'entraîner une application de Deep Learning à identifier avec précision des cellules cancéreuses.
- **Automatisation industrielle** : Le Deep Learning participe à l'amélioration de la sécurité des employés travaillant à proximité d'équipements lourds, en détectant automatiquement les situations dans lesquelles la distance de sécurité qui sépare le personnel ou les objets des machines est insuffisante.
- **Électronique :** Le Deep Learning est utilisé pour la reconnaissance audio et vocale. Par exemple, les appareils d'assistance à domicile qui répondent à votre voix et connaissent vos préférences fonctionnent grâce à des applications de Deep Learning.

7. Deep Learning et robotique

Selon ce qui a été évoqué précédemment dans ce chapitre, le Deep learning est principalement utilisé dans les problèmes de classification supervisée. Cependant, les chercheurs commencent récemment à l'intégrer en robotique pour concevoir des contrôleurs capable de gérer et traiter un grand flux de données.

En effet, le deep learning et la robotique ouvrent une nouvelle ère dans plusieurs domaine tel que la médecine. Il y a le robot Rosa de ''Medtech'' pour la chirurgie du cerveau et de la colonne vertébrale, le robot miniature de ''Virtual Incision'' pour les chirurgies abdominales, ou encore le futur robot de Verb Surgical, Il y a aussi le robot '' Terapio '', en développement au Japon, qui suivra les infirmières partout dans l'hôpital et les assistera dans leurs tâches quotidiennes. Les''robots volants''seront de plus en plus amenés à évoluer dans des milieux très encombrés pour des missions de secours, d'exploration ou de cartographie. ''Relay'', le robot-livreur dédié au «room service» est conçu pour livrer des petits objets d'un point A à un point B. Il parcourt le trajet de manière autonome, une fois le plan des locaux mémorisé. «Il est équipé d'un sonar, de caméras et d'un LIDAR. [27]

8. Apprentissage en Deep Learning

En robotique on utilise l'apprentissage par renforcement afin de faire apprendre aux robots d'exécuter leurs tâches. Par ailleurs, avec le Deep Learning on va sans doute utiliser l'apprentissage par renforcement profond.

8.1. Apprentissage par renforcement

En règle générale, les réseaux de neurones sont formés en les exposants à un grand nombre d'exemple. Les erreurs sont détectées et les poids des connexions entre les neurones sont accordés en temps réel par l'algorithme dans le but d'améliorer les résultats. En effet, le processus d'optimisation est largement répété, après quoi le système est déployé et les images non étiquetées sont évolues[27]

L'apprentissage par renforcement est une méthode qui consiste à récompenser les comportements souhaités et/ou à sanctionner les comportements non désirés. Cette méthode d'apprentissage a été adoptée dans le domaine de l'intelligence artificielle afin de diriger l'apprentissage automatique non supervisé à l'aide de récompenses et de pénalités. Elle est utilisée dans les domaines de la recherche opérationnelle, de la théorie de l'information, de la théorie des jeux, de la théorie du contrôle, de l'optimisation fondée sur la simulation, des systèmes multi-agent, de l'intelligence distribuée, des statistiques et des algorithmes génétiques.

8.2.Apprentissage par renforcement profond (Reinforcement Deep Learning)

Nous obtenons des méthodes d'apprentissage par renforcement profond (deep RL) lorsque nous utilisons des réseaux de neurones profonds pour approximer l'un des composants suivants de l'apprentissage par renforcement : fonction de valeur, $\hat{v}(s ; \theta)$ ou $\hat{q}(s, a; \theta)$, politique $\pi(a|s; \theta)$ et modèle (fonction de transition d'état et fonction de récompense).

Ici le les paramètres θ sont les poids dans les réseaux de neurones profonds. Lorsque nous utilisons des modèles "peu profonds", comme la fonction linéaire, les arbres de décision, le codage de tuiles, etc. comme approximateur de fonction, nous obtenons "peu profond"

RL, et les paramètres θ sont les paramètres de poids dans ces modèles.[28]

L'apprentissage par renforcement profond est un apprentissage par renforcement appliqué à l'aide de réseaux de neurones profonds. Ce type d'apprentissage implique que les ordinateurs agissent sur des modèles sophistiqués et examinent un grand nombre d'entrées afin de déterminer un chemin ou une action optimisée.

conduit à un large éventail de succès dans l'apprentissage stratégies dans différents domaines. Pour la manipulation du robot, renforcer- les algorithmes d'apprentissage apportent l'espoir aux machines d'avoir le capacités humaines en apprenant directement la manipulation adroite à partir de pixels brut. Il nécessite moins de données que d'autres méthodes pour entraîner ses modèles. Fait encore plus notable, ce type de modèle peut être entraîné par simulation, ce qui élimine le besoin de données entièrement étiquetées. Au vu de ces bénéfices, nous constaterons davantage d'applications business combinant l'apprentissage par renforcement profond et la simulation fondée sur des agents émerger dans le courant de l'année.[27]

9. Conclusion

Nous avons présenté au cours de ce chapitre l'apprentissage profond avec ses différents types, les réseaux de neurones, ensuite nous avons présenté ses différentes méthodes : RNN, CNN, DBN et quelques domaines d'applications dont il est utilisé, nous nous sommes aussi intéressés à l'utilisation pratique à travers quelques travaux de l'utilisation des différentes méthodes d'apprentissage profond Dans le chapitre suivant, nous allons présenter la conception du système encor le contrôleur et le système d'apprentissage.

Chapitre 3: Conception et implémentation du système

1. Introduction

Ce chapitre est consacré à la conception et l'implémentation de notre système. Au tout début nous allons parler de l'environnement de simulation de notre robot quadrupède ainsi que les capteurs utilisés. Nous allons présenté la conception détaillée du deep learning étant contrôleur de robot. Enfin, nous allons terminer ce chapitre en montrant les résultats obtenus.

2. Environnement de simulation

Nous avons utilisé le simulateur **Gazebo** quiest un simulateur multi-robot tridimensionnel. C'est un environnement de simulation physique qui permet de simuler un ensemble de robots, capteurs et des forces sur des objets, mais cela est fait dans un monde en trois dimensions.

Pour installer Gazebo nous avons utilisé le système d'exploitation Ubuntu parce que c'est le meilleur système qui permet d'exécuter le simulateur de manière efficace. Pour cela nous avons suivi les instructions et les tutoriels présentés par le site officiel des développeurs du Gazebo. Il fallait donc suivre trois étapes. Premièrement, nous assurer que nous avons accès à l'ouvert référentiel source de la fondation robotique car l'installation du package Gazebo doit se faire à partir de ce référentiel. Pour cela, , nous devons entrer le code qui permet d'installer le référentiel osrf dans le système. La commande qui permet de réaliser cela est présenté ci-dessous :

sudo sh -c 'echo "deb http://packages.osrfoundation.org/gazebo/ubuntu-stable `lsb_release -cs` main"> /etc/apt/sources.list.d/gazebo-stable.list'

Deuxièmement, afin de compléter ce processus d'installation, nous configurons la clé du référentiel OSRF en entrant la commande suivante :

wget https://packages.osrfoundation.org/gazebo.key -O - | sudo apt-key add -

Enfin, on peut finaliser le processus d'installation de Gazebo comme n'importe quel autre paquet. Nous utilisons ici la mise à jour **sudo apt-get update** comme d'habitude et c'est une bonne pratique

pour télécharger n'importe quel paquet. Nous passons ensuite à la commande suivante pour installer gazebo de version 11 **sudo apt-get install gazebo11**

Une fois le simulateur Gazebo est installé, nous pouvons le lancer depuis le terminal. Nous allons juste entrer la commande gazebo et ça nous donne l'environnement ou le terrain où nous mettons notre robot. L'environnement de simulation est illustré dans la figure ci-dessous.

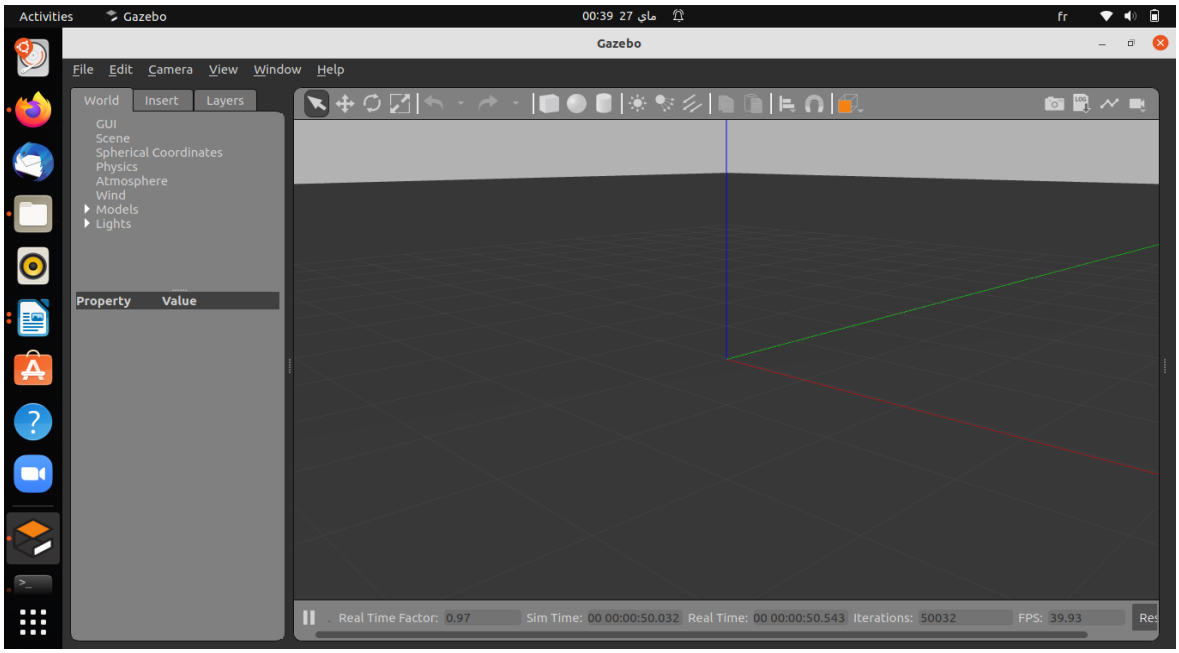


Figure3.1: Environnement de simulation de Gazebo

Afin de concevoir notre système de contrôle de robot nous avons choisit d'utiliser le langage de programmation C++ car sa compilation et son exécution sont nettement plus rapides que celles des autres langages de programmation polyvalents, et qui permettre le développement de logiciels pouvant fonctionner sur différentes plateformes sans trop de modifications.

La machine utilisée dans le cadre de ce travail est un ASUS avec un processeur Intel® Celeron(R) CPU N3050, un espace disque libre de 500.1GB et avec une carte Intel HD graphics 400(Braswell).

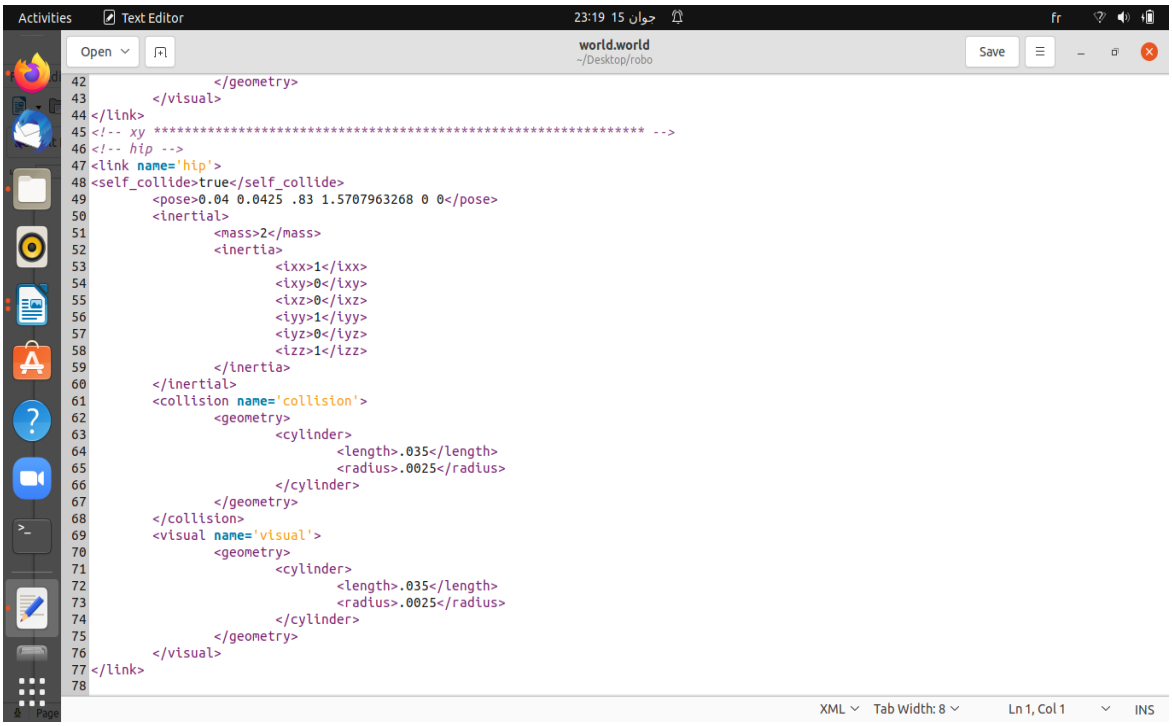
3. Le robot quadrupède

Les robots QUADRUPED sont basés sur un système d'exploitation de robot et peuvent être programmés individuellement en fonction de notre environnement et de notre besoins.

Dans notre travail le modèle de robot est sous forme XML, il se compose d'une partie principale c'est le corps avec quatre pattes, chaque patte possède trois segments de taille égale et entre deux segments il y a une jointure qui permet la connexion. Dans le fichier XML on trouve les informations physiques de notre modèle:

- on trouve la masse de corps est 10 Kg et de taille entre 0 Cm et 0.1 Cm
- on a aussi les dimensions de jointures qui est avec une largeur et une hauteur sont chacune à 1.5 cm et avec une vitesse de 0.5 cm/s et un effort de 1000 N.

on donne un exemple concernant un segment du code XML comme suit:




```
42                </geometry>
43        </visual>
44 </link>
45 <!-- xy ************************************************************** -->
46 <!-- hip -->
47 <link name='hip'>
48 <self_collide>true</self_collide>
49        <pose>0.04 0.0425 .83 1.5707963268 0 0</pose>
50        <inertial>
51                <mass>2</mass>
52                <inertia>
53                        <ixx>1</ixx>
54                        <ixy>0</ixy>
55                        <ixz>0</ixz>
56                        <iyy>1</iyy>
57                        <iyz>0</iyz>
58                        <izz>1</izz>
59                </inertia>
60        </inertial>
61        <collision name='collision'>
62                <geometry>
63                        <cylinder>
64                                <length>.035</length>
65                                <radius>.0025</radius>
66                        </cylinder>
67                </geometry>
68        </collision>
69        <visual name='visual'>
70                <geometry>
71                        <cylinder>
72                                <length>.035</length>
73                                <radius>.0025</radius>
74                        </cylinder>
75                </geometry>
76        </visual>
77 </link>
78
```

figure3.2 : un exemple de code XML d'un segment

Nous avons donc conçu un robot à quatre pattes. Dans la figure suivante, on va faire un dessin explicatif des segments du robot qui représentent une patte. Se sont des rectangles de tailles égales. Nous avons présenté dans ce dessin les jointures par des petits cercles. En fait les jointures tournent uniquement sur un seul axe de rotation. En fait, la jointure qui relie la patte au torse est fixe, cependant les deux autres jointures tournent autour de l'axe x. Les limites de rotation des jointures sont fixées dans le fichier xml. Donc la jointure peut faire des rotations uniquement dans cet intervalle

entre -1.57 rad et 1.57. Nous précisons aussi que la vitesse angulaire ne doit pas dépasser les 0,5 rad/s et une effort de 1000 N.

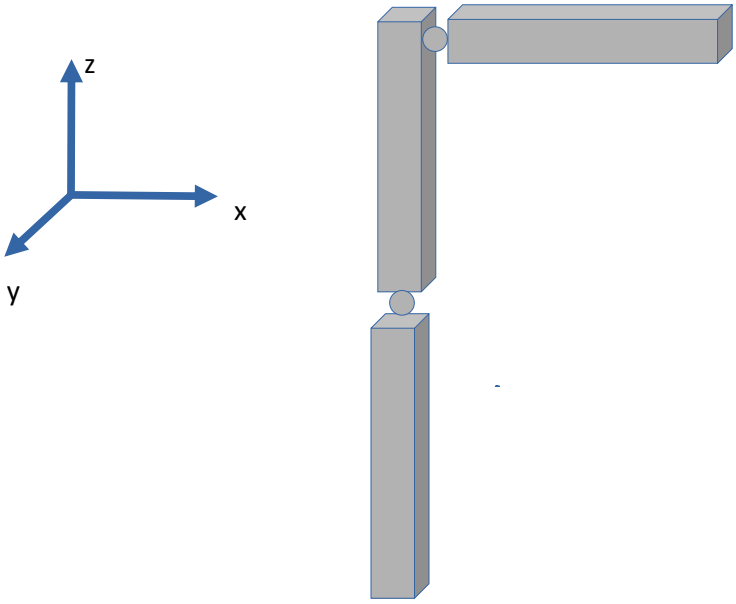


figure3.3: dessin explicatif des segments d'une patte du robot

Nous illustrons également dans la figure ci-dessous notre robot simulé sur Gazebo. Il se compose donc d'un torse et quatre pattes. Les pattes sont liés au torse avec des jointures fixes qui ne tournent pas. En tout on possède donc huit jointures qui vont pouvoir faire bouger le robot de façon à lui permettre de marcher.

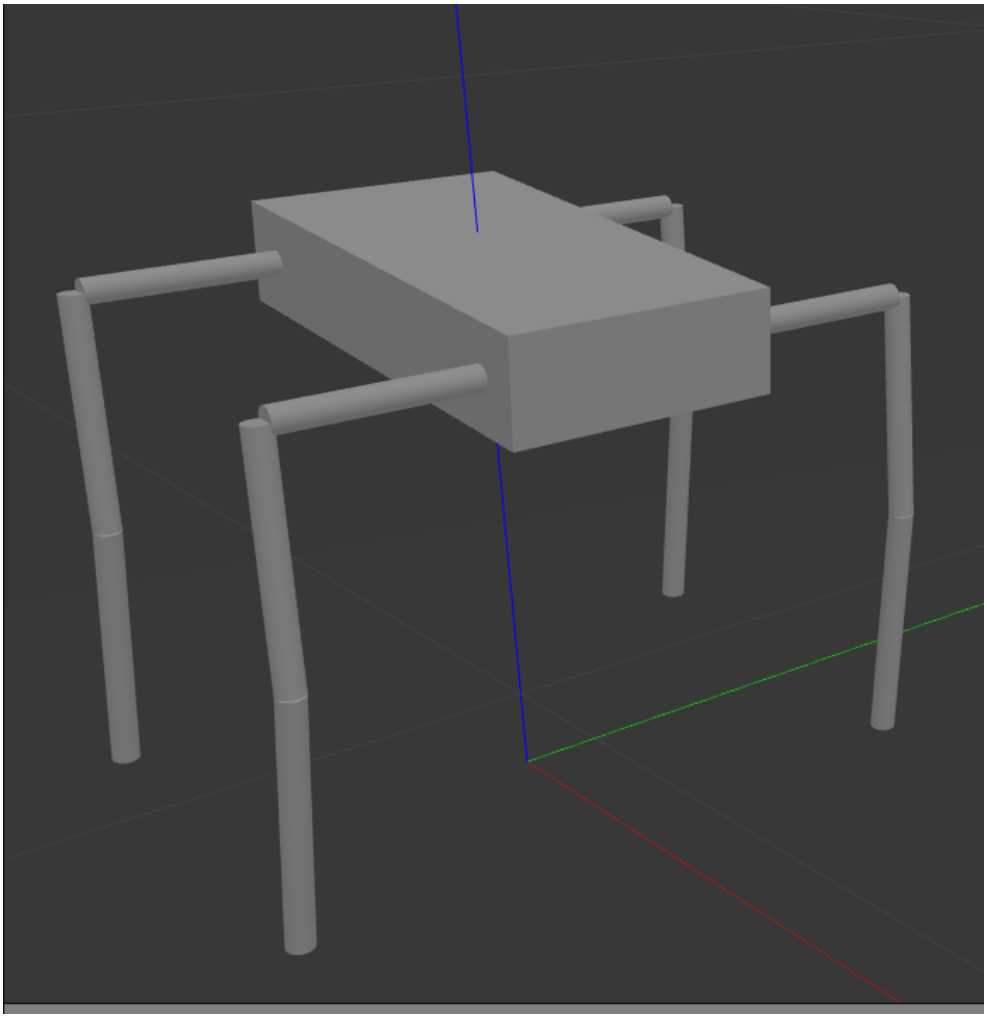


figure3.4: le modèle de robot

4. Les capteurs utilisés

Dans le simulateur gazebo y a beaucoup de capteurs qui sont utilisés. Ces capteurs sont des outils parfaits permettant aux robots d'interagir avec le monde extérieur.

Dans notre modèle il y a des capteur externe et interne comme la position des jointures permettant de mesurer l'angle et l'orientation du membre associé, c'est ce système qui permet au robot de bouger chacun de ses membres. Aussi il y a des capteurs de vitesses et d'accélération des robot et des modules qu'ils réagit avec une vitesse qui tient du réflexe pour se stabiliser. Il y a aussi des capteurs de contact, laser et caméra.

Les capteurs de contact permet de pouvoir générer des signaux lorsque le module entre en contact avec une autre surface, notamment le sol et un autre module et parce que notre environnement et les missions ne sont pas complexes.

Les capteurs laser et caméra se sont parfaits et observent le monde extérieur sans exposer de bruits.

Les capteur de position, de vitesses et d'accélération du robot et des jointures. Donc pour le robot on a sa position dans les trois axes, sa vitesse et son accélération à la fois linéaire et angulaire aussi sur les trois axes. Et pour chaque jointure on prend son angle dans le sens de rotation donnée ainsi que sa vitesse angulaire et son accélération angulaire. Les informations issues de ces capteurs sont mis comme informations d'entrée (input) dans notre réseau de neurone.

Le code source qui permet de recevoir les informations des capteurs sont présenté dans la figure ci-dessous.

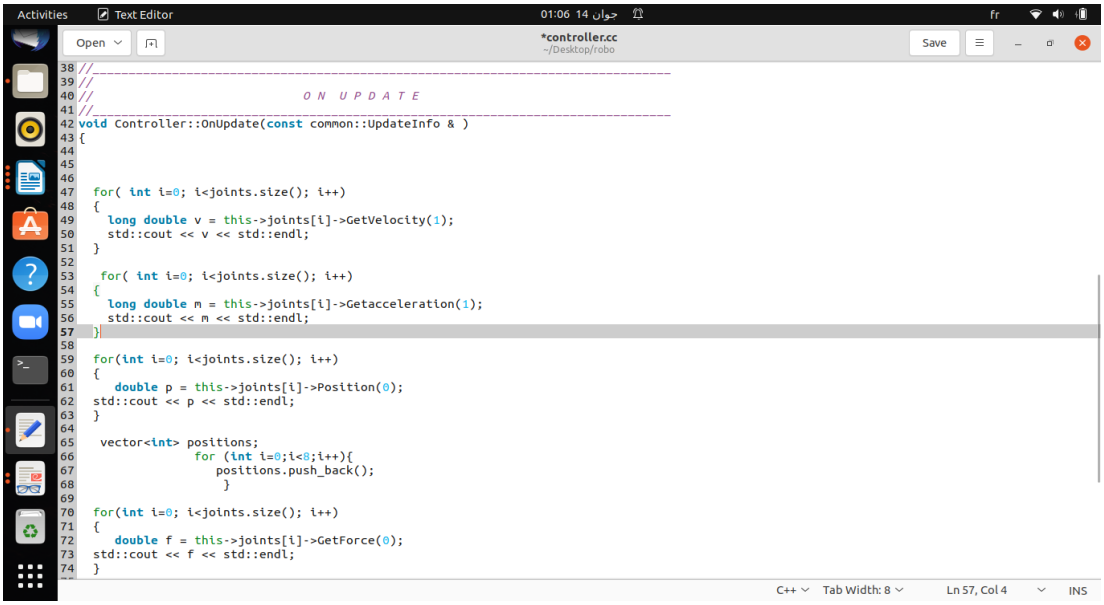


figure3.5: les capteurs de notre modèle

Ces informations d'entrées ont les met dans un vecteur qui s'appelle **input** et ce vecteur on le donne a la fonction **feedForward** qui est une fonction de la classe ANN.

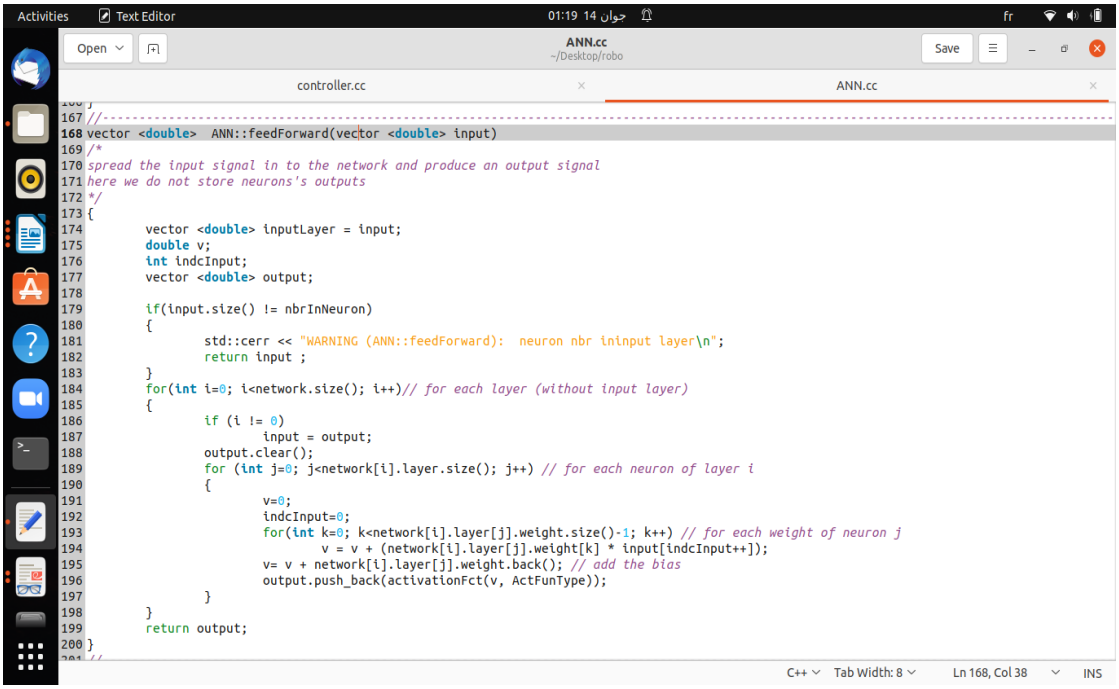


Figure3.6: la fonction feedForward

Cette fonction elle nous retourne un vecteur de 8 sortiés (output) qui vont être appliquées aux jointures du robots comme force en utilisant la fonction SetForce().

5. Le contrôleur

Il existe une variété de contrôleurs de moteur qui peuvent contrôler les robots. Dans notre système on utilise le deep learning pour contrôler notre robot sous forme des réseaux de neurones artificiels qui se compose de plusieurs couches .

On a la fonction feedForward qui diffuse le signal d'entrée dans le réseau et produit un signal de sortie. Dans ce réseau, l'information ne se déplace que dans une seule direction, vers l'avant, à partir des nœuds d'entrée, en passant par les couches cachées (le cas échéant) et vers les nœuds de sorties. Il n'y a pas de cycles ou de boucles dans le réseau. On va appliquer des forces aux jointures pour permettre aux robots de se déplacer dans l'environnement qui est dedans.

Notre réseau demande un grand nombre d'entrées, donc on va utiliser ici beaucoup de capteur qui ont trente trois entrées (inputs) :

pour le robot, on a la position, la vitesse linéaire et angulaire aussi l'accélération linéaire et angulaire du robot et aussi celles de chaque jointures. Afin que le modèle puisse mapper avec précision les entrées aux sorties.

Pour les jointures, on a la position de chaque jointure , vitesse angulaire et accélération angulaire.

Dans notre réseau on a quatre couches cachées, chaque couche à un nombre de neurone dans la première couche on a vingt neurone, pour la deuxième couche cachée quinze neurones et la troisième c'est cinq neurones. a la fin on a la couche sortie (output) qui est composée de huit neurones qui représente la force de chaque jointure .

La structure de ce réseau est comme suit :{33,20,15,5,8}

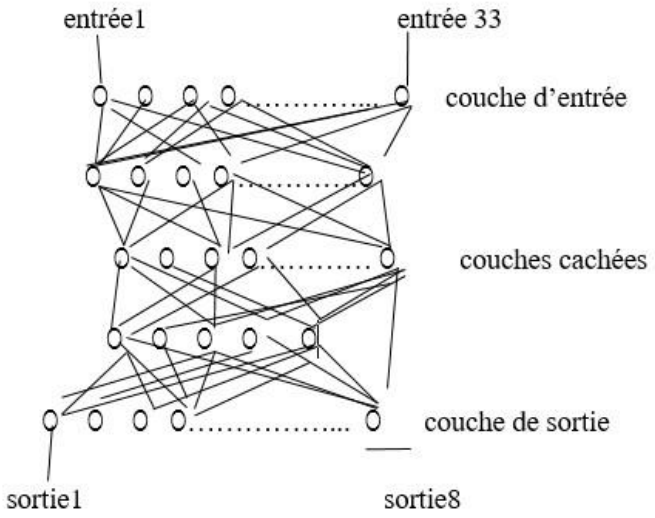


figure3.7 : une structure de RNA

On utilise une classe qui s'appelle ANN comme une bibliothèque , elle contient des méthodes comme la méthode de Feedforward. Cette classe a deux fichiers (ANN.cc et ANN.h)

On a la classe ANN qui est dans le fichier ANN.c, contient des fonctions par exemple on a la fonction ANN qui est le constructeur qui décrit les informations que nous avons dans chaque couche du réseau de neurones. On a aussi la fonction setWeight qui a un vecteur d'entré, nous donnons des poids et des biais à partir de la première couche de haut en bas .il ya la fonction feedforward qui diffuser le signal d'entrée dans le réseau et produire un signal de sortie, ici on ne stocke pas les sorties des neurones. Nous avons aussi la fonction ostream& operator pour l'affichage.

On a la fonction d'activation qui est une fonction mathématique appliqué dans chaque neurone, elle existe aussi dans la fonction feedforward. Elle à utiliser dans la couche cachée ainsi que dans la couche de sortie du réseau, aussi décide si un neurone doit être activé ou non.

Il y a plusieur principales fonctions d'activations que l'on peut trouver dans le domaine des réseaux de neurones, on choisis la fonction d'activation **Sigmoid** (logistic) qui est une fonction pour une utilisation pour les couches cachées. Elle perd de l'information due à une saturation que cela soit pour la phase de feed forward ou de backpropagation, en donnant des effets non linéaires au réseau due à un paramètre unique. Elle a aussi des soucis de gradient 0 avec des entrées étant très large, même si le soucis est minimalisé avec les système utilisant des batch par lots (mini batch). Utilisé en couche de sortie pour de la classification binaire. Intervalle de sortie : {0,1}. ca formule est comme suit:

```
double activationFct(double v, int ActFunType)
{
//      if( ActFunType == 0 )// sigmoid [0 1]
            return ( 1 / ( 1 + exp(- v * slope)));
//      else if ( ActFunType == 1 )//tanh [-1 1]
//            return tanh(v);
}
```

on traduisons les information de cette classe à un schéma qui contien toutes les étapes :

Dans notre schéma au début, on récupérer les données des capteurs qui sont les informations des entrées , aprés on les mettres dans un vecteur , ansuite on les donnes comme paramètre à la fonction feedforward . Au final , on appliques les forces

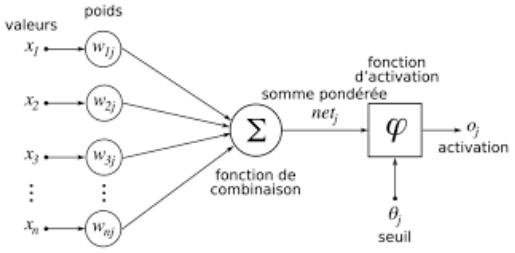


figure3.8 : schéma de réseau de neurone

7. Conclusion

Dans ce chapitre nous avons représenté la conception et l'implémentation de notre système. Nous avons illustré le détail de la conception du modèle de notre robot et comment il a été réalisé en XML. Nous avons montré comment nous avons utilisé le Deep Learning comme contrôleur en utilisant un réseaux de neurones feed forward avec un nombre important de capteurs.

Conclusion générale

Le problème de choisir un contrôleur qui pourra gérer un grand nombre d'informations est encore loin être résolus car ils sont souvent très complexes en robotique. est un domaine de recherche qui passionne les chercheurs depuis des années. Dans le cadre de ce projet, nous nous sommes intéressé à l'une des applications de ce domaine, nous avons proposé un contrôleur pour pourrait gérer ce grand nombre d'informations provenant de capteurs, qui est le Deep Learning.

C'est une méthode basée sur les réseaux de neurones artificiels avec nombre important d'entrées. Afin de d'avoir une simulation réaliste d'un robot quadrupède, nous avons choisis le simulateur gazebo qui fournit des simulation robotiques dans un environnement en trois dimensions.

Nous avons donc conçu dans ce travail un réseaux de neurones capable de recevoir beaucoup d'information qui représentent l'état interne du robot.

Néanmoins voici quelques perspectives et améliorations possibles :

- Nous n'avons pas abordé la partie d'apprentissage des robots quadrupède, Elle fera donc l'une des perspectives de ce mémoire.
- Améliorer la simulation dans un monde en trois dimensions en utilisant des méthodes basées sur l'apprentissage profond.
- Augmenter encore plus le nombre de capteurs et donc le nombre d'informations à passer au contrôleur.

Références bibliographiques :

[1] http://robotiquenuclaireblog.wordpress.com

[2] www.techno-science.net

[3] www.eureka.uqam.ca

[4] David Filliat, " programme unite- GDR Robotique ",Université Numérique de l'ingenerietechnologé, Avril 2012.

[5] Hamdi Hocin, introduction a la robotique, constantine, 2003-2004.

[6] conception et modélisation d'un robot marcheur Hexapode master, Oum-el-Boughi, 2015/2016.

[7] T. Kohonen. Self-Organizing Maps, 3rd edition. Springer Series in Information Sciences.Springer Berlin Heidelberg, 2001.

[8] T. Kohonen. Learning Vector Quantisation and the Self Organising Map, Springer, London, 1992.

[9] D. Heinke and F. H. Hamker. Comparing neural networks : a benchmark on growing neural gas, growing cell structures, and fuzzy artmap. IEEE Transactions on NeuralNetworks, 9(6) :1279–1291, Nov 1998.

[10] Y. Prudent and A. Ennaji. An incremental growing neural gas learns topologies. In Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005., volume 2, vol. 2, July 2005.

[11]https://magazinemci.com

[12]Simulateur robotique - https://fr.abcdef /Robotics_simulator

[13] www.semanticscholar.org

[14] www.projets-ima.plil.fr

[15] Webots: simulateur de robot (cyberbotics.com)

[16] https://www.theconstructsim.com/

[17] ADOUANE SORAYA, Estimation des mesures relatives aux objets détectés à partir des images, BISKRA, 2020-2021)

[18] Comment fonctionne un réseau de neurones ?, Laure Giraux .Disponible : http://aiandi.fr/comment-fonctionne-un-reseau-de-neurones)

[19] Perceptron : qu'est-ce que c'est et à quoi ça sert ?

https://datascientest.com/perceptron

[20] "Understanding LSTM Networks -- colah's blog." [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/. [Accessed: 08-Jun-2021])

[21] P.-A. J. Julien Krywyk, "Classification d'images : les réseaux de neurones convolutifs en toute simplicité." [Online]. Available: https://blog.octo.com/classification-dimagesles-reseaux-de-neurones-convolutifs-en-toute-simplicite/. [Accessed: 08-Jun-2021].

[22] J. W. Kim, "Classification with Deep Belief Networks."

[23] A. Aragon, "SPEECH ACTIVITY DETECTION IMPLEMENTATION IN HEARING AIDS USING DEEP BELIEF NETWORK,", 2017.

[24] fr.education.com

[25] https://www.retengr.com

[26] ALLAL Mohammed Anes, Utilisation du deep learning dans la radio cognitive, Tlemcen, 2017-2018

[27] https://ludo-louis.fr/

[28] https://arxiv.org/abs/1810.06339

[29] https://www.techniques-ingenieur.fr/actualite/articles/sante-du-futur-3-38775/