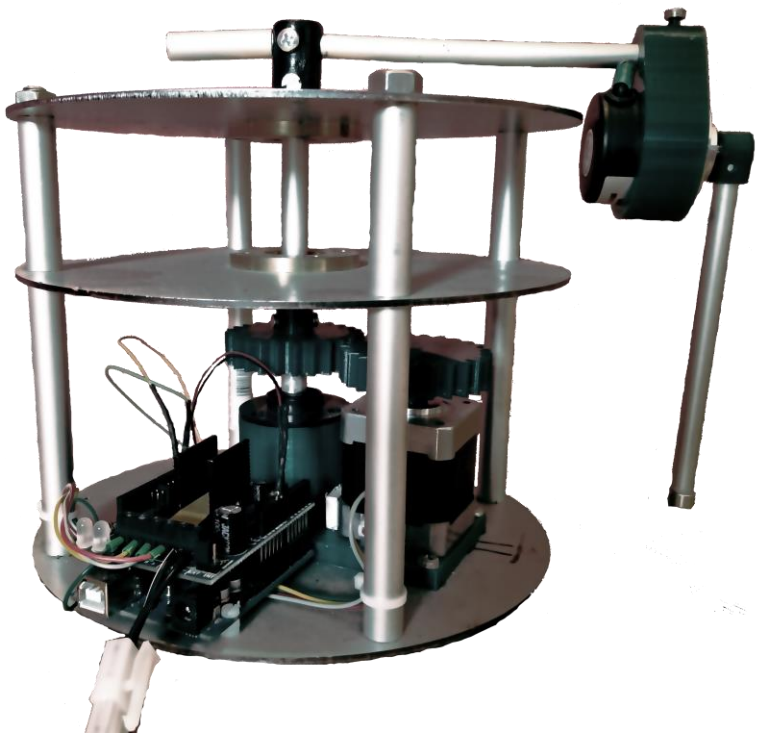
```
def create_agent(a_clear, a_dbmgr)
    print("---------- Create Agent
    a_name = get_value("Name : ",
    rep = a_dbmgr.query("SELECT *
    if len(rep) > 0:
        a_clear()
        print("Agent name already
    else:
        epsilon = get_value("Epsil
        gamma = get_value("Gamma (
        alpha = get_value("Alpha :
        beta = get_value("Beta (El

        print("Enter interval like
        print("Angle are in degree
        print("If speed is higher
        list_pole_angle = get_valu
        list_pole_speed = get_valu
        list_cart_speed = get_valu

        print("Enter actions like
        list_actions = getvalue("A

        a_dbmgr.query("INSERT INTO
        for i in range(len(list_po
            for j in range(len(lis
                for k in range(len
                    state = str(li
                    a_dbmgr.query(
        for action in list_actions
            a_dbmgr.query("INSERT
        a_clear()
        print("Agent created !")

def show_agent(a_clear, a_dbmgr):
    print("---------- Show Agent --
    list_agent = a_dbmgr.query("SE
    if len(list_agent) > 0:
        i = 0
        for agent in list_agent:
            date = datetime.dateti
            print("{0}. {1} : {2}
            i += 1
        choice = input("Choice : "
        try:
```

PT1 – Année 2015/2016

Aux Lazaristes
Lyon - La Salle

Apprentissage par renforcement

Pendule inversé rotatif



Ludovic Bouan | Thomas Lefort

Résumé

Français

Nous avons réalisé un système permettant d'expérimenter et de mettre en application un algorithme d'apprentissage par renforcement. L'objectif est de créer un système virtuel qui, par tatonnement, parvient à trouver la solution optimale à un problème. Pour cela, nous avons construit un pendule inversé rotatif et modélisé ce pendule sur ordinateur. Compte tenu de la complexité de l'apprentissage sur le pendule, nous avons choisi d'implémenter l'algorithme sur un Pac-Man pour effectuer d'autres expérimentations.

English

We have built a system allowing us to employ and experiment with a reinforcement learning algorithm. Our objective is to create a virtual system that can optimally solve a problem through trial and error. To do this we have built a rotary inverted pendulum and modelised this pendulum on a computer. Given the complexity of the pendulum learning task, we also implemented a learning algorithm on a game of Pac-Man in order to run various tests.

Sommaire

1 Introduction

Le contrôle optimal d'un système est un problème courant en ingénieurie et en mathématiques. Aujourd'hui beaucoup de recherches se font dans le domaine de l'apprentissage pour résoudre des problèmes d'optimisation qui n'ont pas de solution mathématique abordable.

Nous allons étudier le fonctionnement de l'apprentissage par renforcement, un sous domaine de l'apprentissage qui permet le contrôle de systèmes complexes et variés. Notre sujet s'inscrit dans le thème par la façon dont fonctionne l'apprentissage. Pour apprendre, il est nécessaire d'organiser et de traiter des données qui évoluent dans le temps de façon à les faire évoluer vers l'objectif que l'on donne au système.

L'intégralité de notre projet se trouve sur GitHub à l'adresse suivante :

https://goo.gl/EJURcP 1

2 Théorie de l'apprentissage

L'apprentissage par renforcement est un des trois grands domaines de l'intelligence artificielle : l'apprentissage supervisé, non-supervisé, et par renforcement. L'idée fondatrice de ce modèle est le processus d'apprentissage auquel adhèrent tous les animaux et les hommes lorsqu'ils interagissent avec leur environnement. Après plusieurs répétitions, le chien de Pavlov a compris qu'une sonnerie voulait dire qu'il allait manger. Lorsqu'un enfant se brûle en touchant le feu, il apprend que le contact avec le feu est douloureux et il ne répétera plus cette action. C'est par essai et erreur que l'on apprend à éviter ce qui nous fait mal et à aller vers ce qui nous plaît. L'étude d'apprentissage par renforcement cherche à transcrire ce modèle biologique en un modèle mathématique.

2.1 Définitions

Afin d'aborder le problème de l'apprentissage par renforcement, il est important de définir quelques notions.

On appelle *agent* la partie du système qui apprend et prend des décisions. L'agent interagit avec son *environnement*. On dit que l'agent effectue une *action,* noté **a**, à laquelle l'environnement réagit. Chaque action de l'agent sur l'environnement a une conséquence sur celui-ci, on dit que l'environnement change d'*état,* noté **s**. Chaque changement d'état provoque l'émission d'un signal qui reflète l'intérêt du nouvel état. Ce nombre, noté **r**, est appelé *récompense*. L'interaction entre l'agent et l'environnement se fait de manière discrète au cours du temps. On note les intervalles de temps t, t+1, t+2, etc.

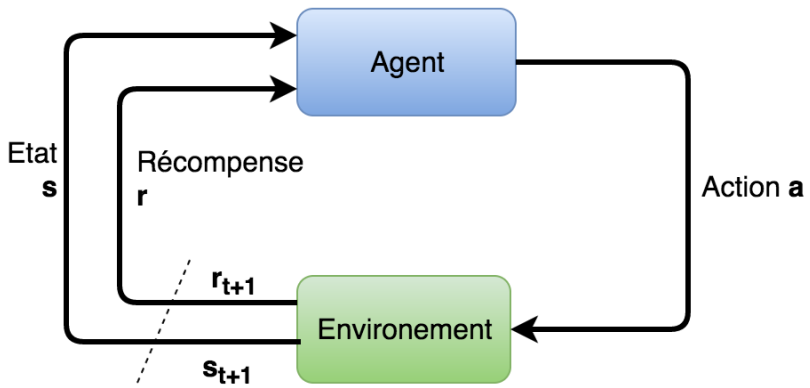


Figure 2.1 ***: Interaction entre l'agent et l'environnement***

Un des organes principaux de l'agent est sa politique, noté π. La politique définit le comportement de l'agent à un moment donné. C'est est une fonction qui à un état associe une action. En notant S l'ensemble des états et A l'ensemble des actions :

$$\pi : S \rightarrow A.$$

L'apprentissage par renforcement a comme but d'améliorer la politique afin qu'elle associe à chaque état l'action optimale à prendre. On note la politique optimale π*.

La fonction récompense associe à chaque état de l'environnement un nombre scalaire. Ce nombre indique l'intérêt d'être dans cet état précis. En analogie avec un système biologique, la récompense peut être assimilée au plaisir et à la douleur. L'objectif principal de l'agent est de maximiser la somme des récompenses qu'il reçoit au cours

du temps. La fonction récompense est donc caractéristique du problème auquel l'agent fait face. Avec les notations habituelles :

$$R : S \rightarrow \mathbb{R}$$

Alors que la fonction récompense définit ce qui est « bien » dans l'immédiat, la *fonction valeur* définit ce qui est bien à long terme. D'une manière générale, a valeur d'un état est la somme des récompenses que l'agent peut espérer recevoir en partant de cet état. Alors que la récompense nous indique l'intérêt immédiat d'être dans un état, sa valeur nous indique son intérêt à long terme, c'est-à-dire après avoir considéré la valeur des états posterieurs. Un exemple analogique serait qu'il vaut mieux faire des études et renoncer à un salaire immédiat, pour gagner un salaire plus important plus tard. Un état peut lui-même apporter une récompense très faible, mais permet d'aller vers des états qui ont des récompenses élevées : cet état a donc une valeur élevée. De la même manière, on peut aussi donner une valeur à chaque couple état-action (**s**, **a**). On appelle cette valeur la Q-value, et elle sera au cœur de notre étude. Avec les notations habituelles :

$$Q : S \times A \rightarrow \mathbb{R}.$$

2.2 Différence Temporelle

L'algorithme que l'on propose d'étudier cherche à approcher le plus possible la fonction valeur Q optimale que l'on note Q*. Au fur et à mesure que l'agent gagne de l'expérience, il doit pouvoir améliorer itérativement son approximation de Q*. Pour cela l'algorithme utilise la différence temporelle.

Soit $Q_t = Q_t(s,a)$ la valeur estimée de $Q^*(s,a)$ après **t** pas de temps et $\boldsymbol{q_i}$ un point expérimentale. Une estimation raisonnable de Q_t est la moyenne de l'échantillon :

$$Q_t = \frac{q_1 + q_2 + \cdots + q_t}{t}$$

Si on connaît déjà Q_{t-1}, on peut montrer que Q_t s'écrit[1]:

$$Q_t = Q_{t-1} + \alpha_t(q_t - Q_{t-1})$$

[1] **Poole, David et Mackworth, Alan.** Q-learning. *Artificial Intelligence - Foundations of computational agents.* [En ligne] 2010. [Citation : 10 Avril 2016.] http://artint.info/.

$$\text{avec } \alpha_t = {}^1/_t$$

Cette formule s'appelle la « value iteration update » et nous permet d'incrémenter notre estimation itérativement à chaque fois que l'on obtient un nouveau point de l'échantillonnage.

La différence $\boldsymbol{q_t - Q_{t-1}}$ est appelée l'erreur de différence temporelle, ou TD error. Elle quantifie la différence entre l'ancienne estimation Q_{t-1}, et la nouvelle valeur expérimentale q_t. Si la nouvelle donnée est supérieure à l'ancienne estimation, on augmente l'estimation ; si elle est inférieure, on la diminue. L'augmentation ou la diminution est proportionnelle à l'erreur de différence temporelle et à α_t. La valeur de $\alpha_t = {}^1/_t$ est valable lorsque toutes les données ont le même poids. Cependant, dans notre cadre, les données les plus récentes sont les plus précises. On utilise donc α constant pour privilégier les nouvelles données. Sous certaines conditions, que nous ne détaillerons pas ici, la valeur de Q_t converge tout de même vers la moyenne pondérée de l'échantillon.

2.3 Q-Learning

La compréhension de l'algorithme de Q-learning est fortement intuitive. Dans chaque état, il existe un certain nombre d'actions possibles. Chacune de ces actions a une valeur (la Q-value), déterminée par la quantité de récompenses que l'on peut espérer recevoir, à long terme, en effectuant cette action. Chaque couple état-action (**s**,**a**) a une Q-value, que l'on initialise à un nombre quasi-arbitraire, souvent à 0. L'agent parcourt l'espace des états possibles et cherche à approcher ces Q-values. Une fois qu'il essaie une action dans un état ***s,*** il évalue l'état dans laquelle il arrive, ***s'***. Il fait alors une « mise à jour » itérative. Il augmente Q(**s**,**a**) si ***s'*** a une bonne *valeur*, et il la diminue si ***s'*** est a une mauvaise *valeur*. Cette évaluation se fait en regardant la récompense et la Q-value maximale associée à l'ensemble des actions qu'il lui est possible de réaliser dans cet état ***s'***. Ce type de Q-learning est dit « one step », car on regarde un pas en avant. Une fois une bonne approximation faite des Q-values, l'agent n'a plus qu'à choisir à chaque fois l'action avec la plus grande Q-value. Il obtiendra à long terme le maximum de récompenses et aura donc suivi le chemin optimal. Ci-dessous nous formaliserons cette idée. Un schema de principe du Q-learning est proposé à la fin de cette partie.

On définit la Q-value d'un couple (**s**,**a**) comme la somme totale des récompenses que l'on peut espérer recevoir en suivant une politique π et en introduisant un facteur de réduction $\gamma \in [0,1]$:

$$Q_t^*(s,a) = \mathrm{E}_\pi[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \cdots]$$

On remarque que :

$$Q_t^*(s,a) = r_t + \gamma Q_{t+1}^*(s',a')$$

Pour le Q-learning on s'interesse à la politique optimale π^* d'où :

$$Q_t^*(s,a) = r_t + \gamma \max_{a'} Q^*(s',a')$$

Ce terme γ a deux rôles, éviter que la somme tende vers l'infini lorsque le système marche en continu, et diminuer l'influence des récompenses les plus éloignées dans le futur, car elles sont moins certaines.
Pour trouver la politique optimale, le Q-learning cherche d'abord la fonction valeur Q*(s,a) pour chaque état. On définit ensuite une politique qui choisit l'action maximisant la fonction valeur de chaque état :

$$\pi^*(s) = \arg\max_a Q^*(s,a)$$

Le Q-learning va donc utiliser la différence temporelle pour approcher Q*.

On appelle *expérience* le tuple (**s**,**a**,**r**,**s'**) où **s** est l'action de départ, **a** l'action prise, **r** la récompense, et **s'** l'état d'arrivée. L'agent retient un tableau Q[S,A] avec S l'ensemble d'états et A l'ensemble d'actions. Chaque entrée Q[**s**,**a**] du tableau représente l'estimation courante de Q*(**s**,**a**). A chaque nouvelle expérience l'agent peut alors mettre à jour itérativement son estimation selon la formule du «value iteration update» :

$$Q_{t+1}(s,a) = Q_t(s,a) + \alpha\big(target - Q_t(s,a)\big)$$
$$\text{où } target = r + \gamma . \max_{a'} Q(s',a')$$

On obtient donc l'algorithme suivant[2]:

```
Entrée :
     S un ensemble d'états
     A un ensemble d'actions
     γ le facteur de décroissement
     α le taux d'apprentissage
Variables locales :
     Q[S,A] un tableau
     s l'état précédent
     a l'action précédente

Qupdate(S, A, γ, α) {
1.   Initialiser Q[S,A] arbitrairement
2.   Observer l'état actuel s
3.   Répeter :
4.       Choisir une action selon la politique π
5.       Effectuer l'action
6.       Observer l'état d'arrivée s' et la récompense r
7.       Q(s,a)← Q(s,a)+ α(r + γ. max_a' Q(s',a') - Q(s,a))
8.       s←s'
}
```

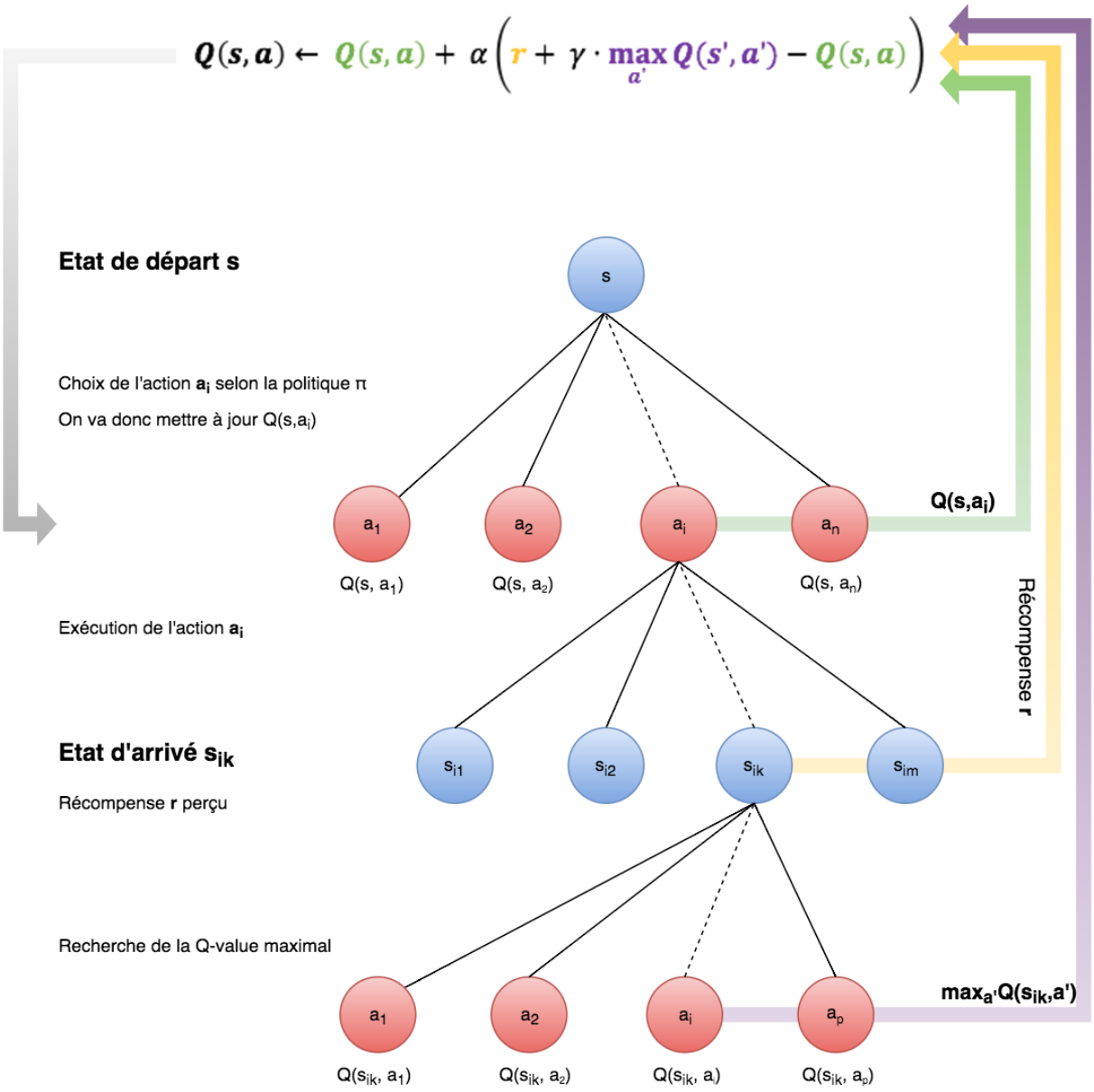
La particularité du Q-learning est qu'il apprend la politique optimale même en suivant une politique non-optimale. Peu importe la manière dont il choisit l'action **a**, il met à jour Q par rapport à $max_{a'}Q(s',a')$, c'est-à-dire par rapport au meilleur chemin. On dit alors que le Q-learning est « off-policy ». Certains algorithmes similaires, comme SARSA, sont « on-policy ». Ces algorithmes apprennent la valeur de la politique qu'ils suivent.

Il a été démontré en 1992 par Watkins et Dayan que le Q-learning convergera toujours vers π^*[3].

[2] **Poole, David et Mackworth, Alan.** Q-learning. *Artificial Intelligence - Foundations of computational agents*. [En ligne] 2010. [Citation : 10 Avril 2016.] http://artint.info/.

[3] **Watkins, Christopher J.C.H. et Dayan, Peter.** *Q-learning - Machine Learning*. Boston : C.J.C.H., 1992.

Schema de principe du Q-learning



2.4 Exploration

L'exploration est une composante importante de l'apprentissage. L'agent doit réaliser un compromis entre l'exploitation qui lui permet de maximiser sa récompense, et l'exploration qui pourrait améliorer encore plus sa récompense à long terme. Un manque d'exploration peut entraîner la convergence de l'agent vers une politique sous-optimale.

Nous avons choisi l'exploration appelée ϵ-greedy, qui consiste à choisir avec une probabilité ϵ une action aléatoire à réaliser. Avec une probabilité ϵ, l'agent choisit une

action au hasard, sinon il choisit la meilleure action. Cela lui permet de « tester » de temps en temps de nouveaux états qu'il n'a jamais visités, pour voir si ces derniers n'apporteraient pas plus de récompense que le chemin qu'il considère actuellement comme le meilleur.

2.5 Traces d'éligibilité

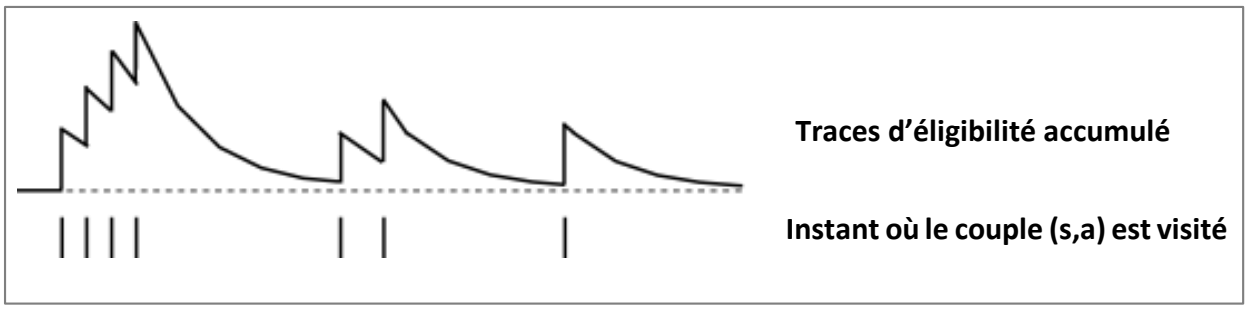
Cependant, l'agent présente encore un gros handicap. Lorsqu'il reçoit une récompense ce n'est pas répercuté directement sur les couples états-actions précédents qui sont aussi responsables de la récompense actuelle (aspect séquentiel). Pour accélérer l'apprentissage, on utilise alors des *traces d'éligibilité*, notés **ε**. Leur rôle est de garder en courte-memoire l'influence des couples (**s**,**a**) visités. Ainsi, à chaque fois qu'une récompense est reçue, on augmente la Q-value de tous les couples (**s**,**a**) visités précédemment proportionellement à leur influence, soit **ε**(**s**,**a**).

On initialise tous les **ε**(**s**,**a**) à 0 puis à chaque fois qu'un couple est visité, on incrémente sa trace d'éligibilité de 1. A chaque étape où ce couple n'est pas visité, **ε**(**s**,**a**) décroit exponentiellement. On note[4]:

$$\varepsilon_{t+1}(s_i, a) = \begin{cases} \gamma\lambda\varepsilon_t(s_i, a) + 1, & si\ s = s_i \\ \gamma\lambda\varepsilon_t(s_i, a), & sinon \end{cases}$$

La mise à jour des Q-values se fait alors selon cette formule :

$$Q_{t+1}(s, a) = Q_t(s, a) + \alpha \cdot \varepsilon(s, a) \cdot \big(target - Q_t(s, a)\big)$$



Un problème survient quand une action non optimale d'exploration est choisie. Le principe fondamental du Q-learning est qu'il n'apprend que de la politique optimale

[4] **Silver, David.** *Advanced Topics: RL.* [En ligne] [Citation : 9 Decembre 2015.] http://wwwo.cs.ucl.ac.uk/staff/d.silver/web/Teaching.html.

même lorsqu'il explore de nouvelles possibilités. Il ne faut donc pas que les traces d'éligibilité antérieures à une action d'exploration ait de l'influence sur l'apprentissage futur.

Une solution suggeré par Watkins pour rendre compatible la trace d'éligibilité avec le Q-learning est de remettre toutes ces traces à 0 lorsque qu'une action non optimale est choisie. L'utilité de la trace d'éligibilité dans le Q-learning est donc inversement proportionnel au taux d'exploration.

3 Applications

3.1 Le Pendule Inversé

Le pendule inversé est un système couramment utilisé dans la recherche pour expérimenter différentes solutions, dans des domaines aussi variés que l'asservissement et l'intelligence artificielle, bien qu'il ne trouve que peu d'applications concrètes (Segway). L'objectif de la tâche est de maintenir le pendule à la verticale.

3.1.1 Présentation

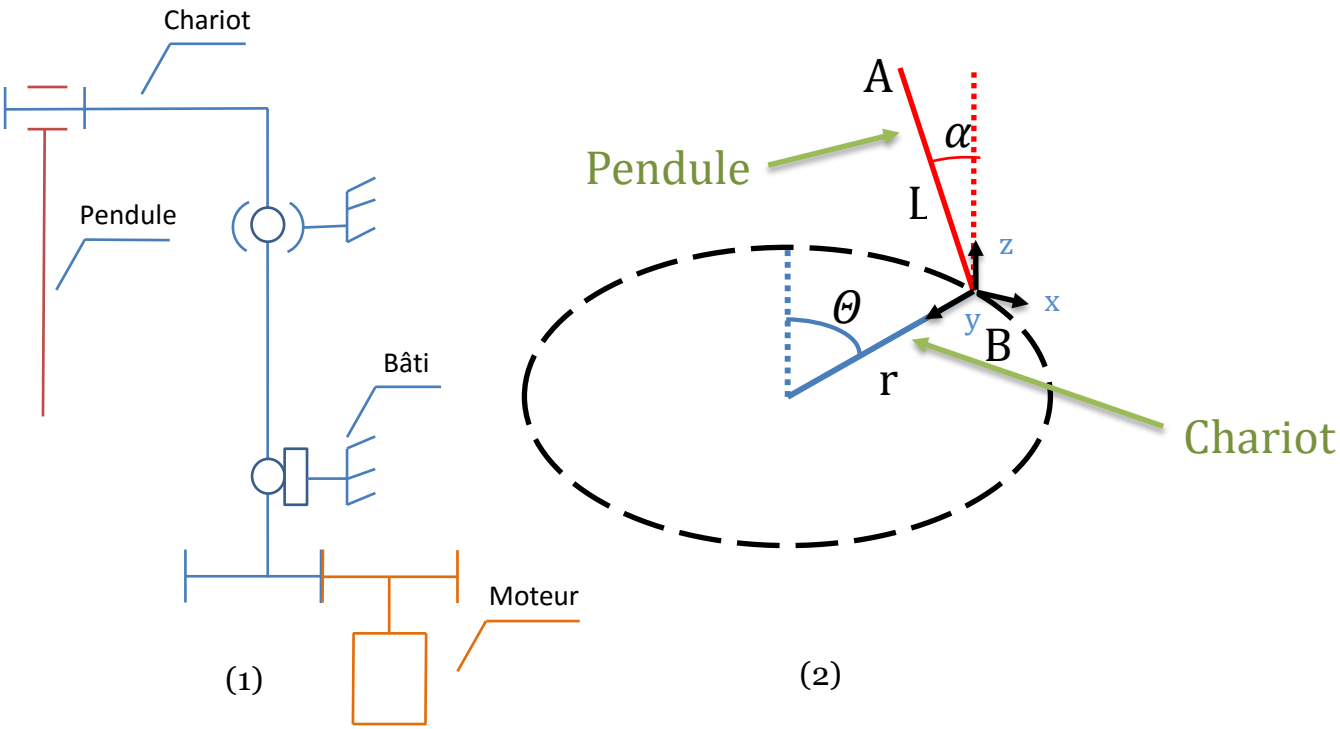


Figure 3.1 : *(1) Schéma cinématique (2) Paramétrisation du pendule inversé*

Le pendule inversé rotatif a des avantages sur le pendule inversé en translation. La liaison mécanique est plus facile à réaliser et le chariot n'est pas limité dans son déplacement, ce qui fait que l'apprentissage ne dépend pas de la position du chariot mais seulement de sa vitesse et réduit donc le nombre de paramètre pour l'apprentissage.

3.1.2 Conception de la maquette

Afin de mettre en pratique la théorie de l'apprentissage nous avons décidé de réaliser un système physique afin d'étudier les contraintes que cela implique.

En effet, le système doit être robuste pour résister aux différentes expérimentations, être stable et précis. Nous avons donc choisi de suivre le cahier des charges suivant :

F1	Obtenir une réponse constante à la commande de rotation de l'axe principal
F2	Assurer la robustesse du système pour endurer les expérimentations
F3	Assurer la transmission des informations renvoyées par les capteurs
F4	Faciliter le transport et les branchements

La constance du système est plus importante que la précision car c'est l'apprentissage qui adapte ses commandes à la réponse du système.

Nous avons été confrontés à plusieurs choix technologiques :

- Pour mesurer la position du pendule, nous avons choisi un encodeur incrémental pouvant coder jusqu'à 2400 positions par tour. Nous étions ainsi sûrs que la précision de cette mesure ne porterait pas préjudice à l'apprentissage
- Nous avons choisi un moteur pas à pas pour sa facilité d'utilisation et sa précision. La précision apportera de la constance au système.
- Nous avons choisi une carte Arduino capable de communiquer facilement avec un ordinateur et un Shield moteur facilitant le contrôle du pas à pas.

La plupart des élements se trouvent dans la partie inférieure du système pour éviter un déséquilibre de celui-ci et pour faciliter les connexions.

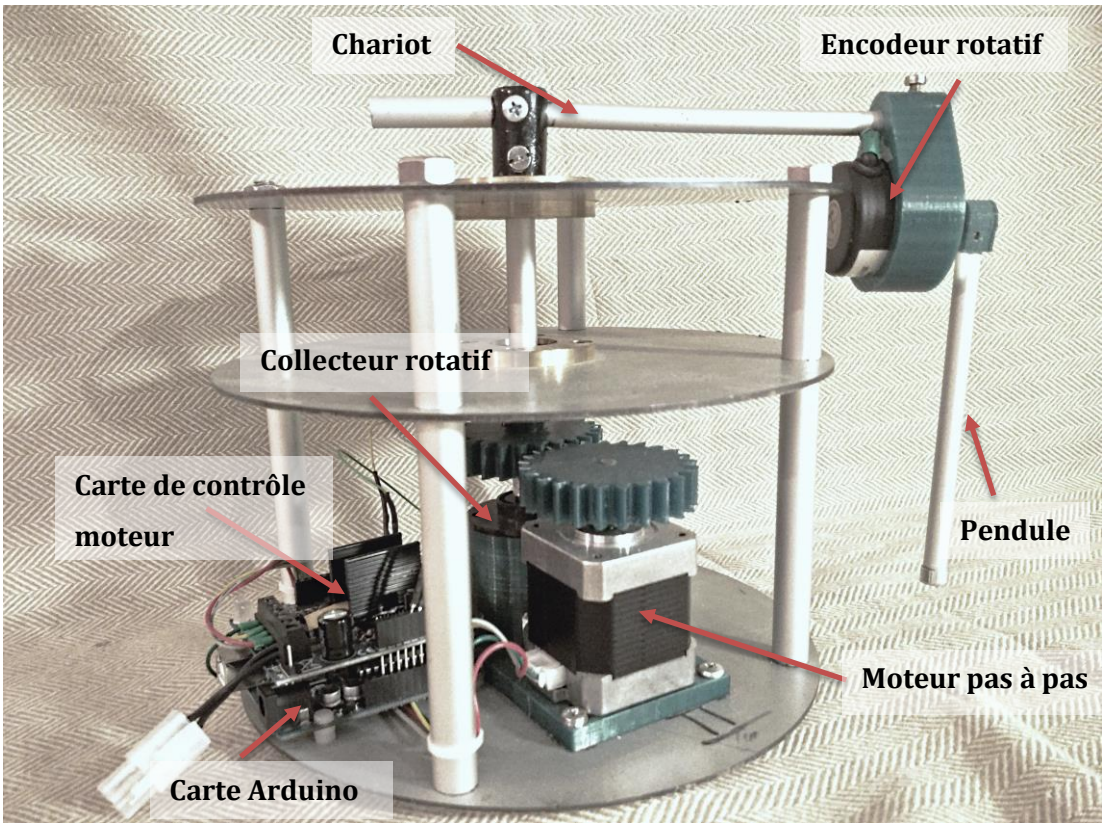


Figure 3.2 ***: Photo légendée du pendule réel***

3.1.3 Problèmes rencontrés

Le premier problème rencontré a été la transmission de l'information de l'encodeur rotatif (qui se trouve au bout du système tournant) à l'Arduino qui est fixe avec le bâti. Le système pouvant faire plusieurs tours, il n'était pas possible de prévoir simplement un cable qui s'enroule autour de l'axe. Nous avons donc fait le choix de faire passer le cablâge à l'intérieur des axes en aluminium et de transmettre l'information à l'aide d'un collecteur rotatif.

Le deuxième problème rencontré a été la mise en position du moteur afin que le jeu dans l'engrenage soit le plus faible possible. La précision de perçage dans les pièces imprimées en 3D étant faible, nous avons donc utilisé une cale afin de réduire le jeu.

Enfin, le dernier problème rencontré a été l'élevation importante de la température de la carte de contrôle du moteur, celui-ci fonctionnant à faible voltage mais à forte intensité. La puce du Shield Arduino, bien que faite pour supporter jusqu'à 2,4A commence à chauffer de façon très importante à partir de 1,7 A (un thermomètre infrarouge nous indiquait une température de 72°C)

De plus, l'intensité qui passe dans un moteur pas à pas dépend de sa vitesse et du couple résistant appliqué en sortie du moteur. Cela est dû à des phénomènes magnétiques qui créent une résistance plus ou moins importante dans les bobines (schéma ci-dessous).

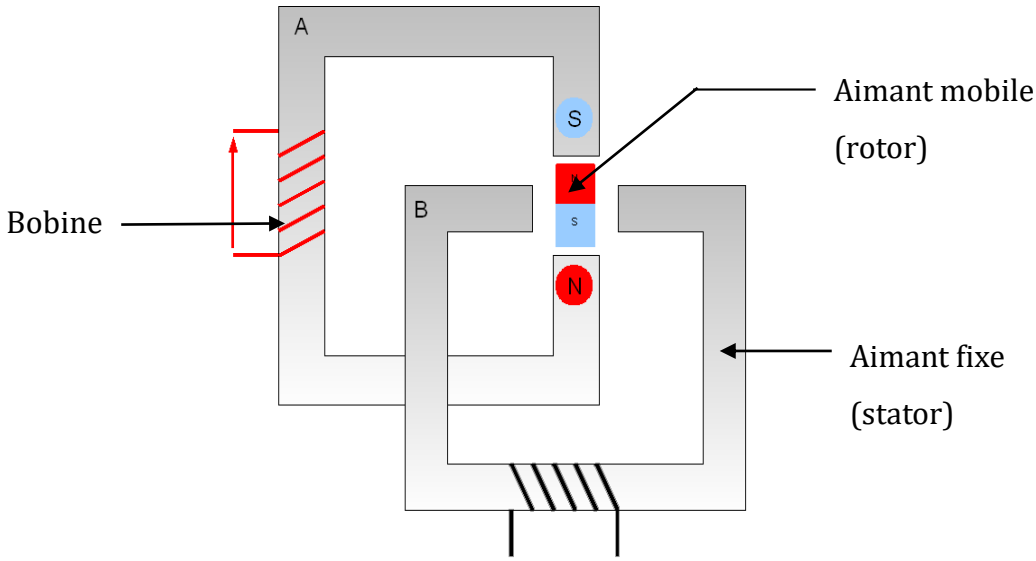


Figure 3.3 ***: Fonctionnement interne d'un moteur pas à pas***

Lorsque l'on alimente l'une des bobines du moteur, cela bloque la rotation. Seulement, si on n'exerce aucune force en sortie de moteur, la résistance dans la bobine est très faible, ce qui entraine une intensité de courant importante.

Lorsque le moteur tourne, les deux bobines sont alimentées successivement et le couple en sortie du moteur crée une résistance importante dans les bobines (par effet magnétique) ce qui réduit l'intensité. Cependant, si l'intensité devient trop faible, le moteur n'a plus la puissance nécessaire pour faire tourner convenablement le système.

Il a donc fallu trouver un compromis entre ces deux situations. Afin de régler ce problème, nous avons inclus à l'alimentation une carte qui permet de limiter plus ou moins la tension et l'intensité qu'elle fournit.

3.1.4 Cadre de l'étude

Le pendule inversé représente en apprentissage un problème observable, mono-agent, stochastique, séquentiel, dynamique et continu.

Attribut	Définition
Observable	L'environnement renvoit à travers ses capteurs, des informations complet sur l'état actuel du pendule et qu'il n'est pas nécessaire de garder un historique de ces informations pour connaitre l'état du système
Mono-Agent	L'agent n'est pas en concurrence avec un autre agent
Stochastique	Dans un état donné l'execution d'une action ne détermine pas exactement l'état suivant du pendule
Séquentiel	L'état présent est lié aux états passé
Dynamique	Le système continu à évoluer pendant que l'agent prend sa décision
Continu	L'ensemble des états peut-être vu comme un ensemble continu

L'aspect stochastique, séquentiel, dynamique et continu représente des obstacles importants à la résolution du problème. C'est pourquoi nous avons appliqué une simplification au problème qui rend l'espace des état discret en définissant un nombre fini d'états comme intervalle de position et de vitesse du pendule.

De même, l'aspect dynamique du problème complexifie la résolution. Nous avons donc aussi simplifié le problème réel avec une simulation. En effet, avec la simulation il est possible d'arrêter l'évolution du système lors de la prise de décision, ce qui facilite l'apprentissage.

3.1.5 Simulation

Suite à un manque de résultats avec le pendule inversé, nous avons eu recours à une simulation par ordinateur. Avec le logiciel Blender 3D, nous avons construit une maquette simplifiée de notre pendule. Ensuite, en utilisant le simulateur physique de Blender et l'API python associé, nous avons reproduit la situation réelle. Cette simulation avait pour but de mettre en œuvre notre algorithme sur un système

« parfait ». Ainsi, on a déterminé que la construction de notre pendule inversé n'est pas totalement à l'origine de l'échec d'apprentissage.

L'avantage de la simulation est que l'on peut démarrer le pendule à la verticale, et donc se concentrer sur la phases d'équilibrage, plutôt que sur la phase de balancement.

3.2 Programmation

Nous avons réalisé un programme permettant de structurer et d'organiser l'apprentissage afin de faciliter l'exploitation des résultats. Pour cela, nous avons centralisé toutes les fonctions et expérimentations dans un seul programme. Afin de tester plusieurs paramètres, il est possible de créer plusieurs agents avec leurs paramètres associés et plusieurs expérimentations pour chaque agent. Les tables de Q-values sont enregistré dans une base de données afin de pouvoir fermer le programme et pouvoir relancer l'apprentissage.

Le programme permet aussi, en modifiant les paramètres, de communiquer avec la simulation afin de regrouper tout les résultats au même endroit et avec la même structure.

Nous avons aussi développé d'autres fonctions permettant de tester le système physique indépendamment de l'apprentissage, en imposant une série d'actions à éxecuter plusieurs fois pour les comparer.

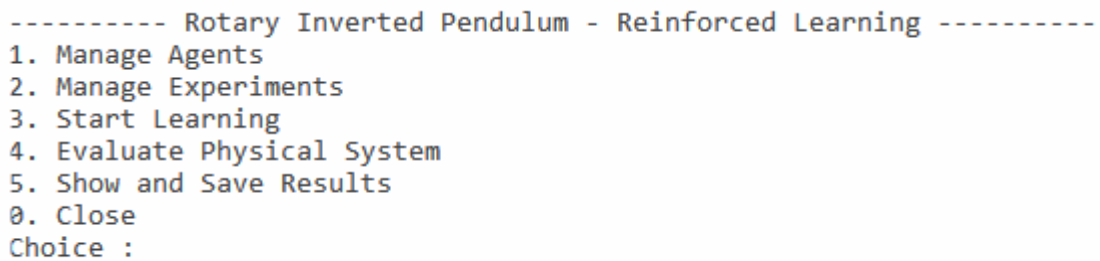


Figure 3.4 ***: Capture d'écran du menu principal de notre programme***

3.3 Autres applications

Le pendule inversé s'est avéré plus complexe à mettre en œuvre que nous l'avions imaginé. Nous avons voulu vérifier que nous étions tout de même en mesure de mettre en œuvre un agent d'apprentissage par renforcement. L'université de Berkeley propose à ses étudiants une plateforme qui permet de réaliser un agent d'apprentissage par renforcement dans un jeu de labyrinthe et un jeu de Pac-Man. La plateforme est un

squelette contenant les fonctions de base qu'il faut completer afin de mettre en place l'apprentissage. En utilisant cette plateforme, disponible sur le site de l'université, nous avons créé un agent capable d'apprendre à jouer à Pac-Man.

4 Expérimentations et résultats

4.1 Echec de l'apprentissage du pendule

Nous regrettons de ne pas être parvenus au résultat attendu sur le système réel ou sur la simulation, bien que des progrès nets dans le comportement du système aient été observés. Le système du pendule nécessite l'apprentissage de deux capacités, le balancement et l'équilibrage du pendule en position haute. D'ailleurs, en asservissement, ceci est décomposé en deux systèmes distincts.

Le système réel impose que l'état de départ soit la position basse du pendule (position la plus stable). Le système commence donc à apprendre le balancement. Nous avons réussi à observer une évolution de l'apprentissage qui tendait vers un balancement efficace.

La simulation nous autorisait à faire démarrer le pendule en position haute pour travailler sur l'apprentissage de l'équilibre en position haute. De même, nous avons constaté qu'après une période d'apprentissage, le pendule se maintenait en équilibre pendant une durée notable. Cependant, notre système, bien qu'il s'améliore, ne semble pas parvenir à maintenir l'équilibre de manière durable.

Nous chercherons à expliquer cela dans la suite du chapitre.

4.2 Constance de la réponse de la maquette

Il est important que la maquette ait une réponse qui varie peu lorsque l'on envoie une commande car l'apprentissage se base sur cette constance pour évoluer. Cela influence le degré de stochasticité du système. Nous avons donc créé un menu qui permet d'entrer une liste d'actions à exécuter. Puis la séquence d'actions est effectuée en enregistrant les états du pendule à chaque étape. En ayant généré une serie de 80 actions aléatoires, répétée 5 fois, on obtient le résultat suivant :

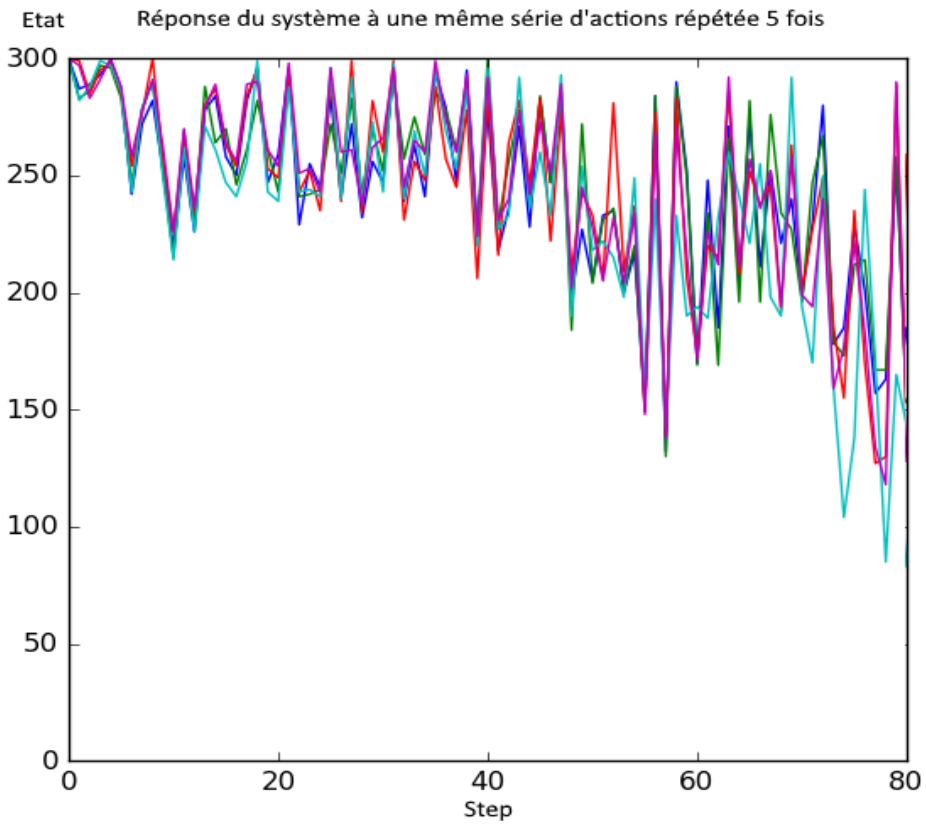


Figure 4.1 *: Réponse du système à une série d'actions répétée 5 fois*

On observe une forte similarité dans l'enchainement des états sur les 50 premières actions. Le degré de stochasticité du système est alors très faible, ce qui est un bon facteur pour l'apprentissage.

4.2.1 Problèmes liés aux choix de configuration de l'apprentissage

Une autre source d'erreur possible dans l'apprentissage est le choix des paramètres. Le premier paramètre est le découpage de l'espace d'états en un nombre discret d'intervalles de possibilité. Si le découpage est trop grossier, le système ne peut pas différencier deux états aux comportements différents ou peut ne pas avoir le temps de réagir à un changement de comportement du pendule. Si le découpage est trop fin, cela augmente très rapidement le nombre d'états et donc de manière exponentielle le temps d'apprentissage.

Le second facteur est le choix des hyper-paramètres. Un facteur de décroissance trop élevé peut créer de l'instabilité ; un taux d'exploration trop faible peut empêcher de découvrir les couples état-action importants.

Nous avons donc comparé nos choix à ceux faits par d'autres personnes travaillant sur le même sujet, mais le rapport de ces paramètres au système est très important. Par

exemple, un jeu dans le système d'engrenage peut induire la nécessité de réduire le facteur d'apprentissage car cela ajoute un phénomène incertain au système. Après avoir essayé d'utiliser leurs paramètres, nous n'avons pas observé d'amélioration notable de l'apprentissage.

Plus de temps nous aurait permis d'expérimenter plus de paramètres pour peut-être parvenir à trouver les hyper-paramètres optimaux pour notre système.

4.2.2 Problèmes liés à la maquette

Nous avons aussi comparé notre travail au niveau de la réalisation de la maquette. La plupart des chercheurs travaillant sur le sujet utilisent des maquettes ayant été conçues pour des usages plus variés et servant aussi à des personnes utilisant d'autres méthodes de contrôle[5].

Nous avons cependant observé des différences de réalisation qui peuvent porter préjudice à notre système :

- Nous utilisons un moteur pas à pas pour mettre en rotation le chariot alors que la plupart des systèmes que nous avons rencontrés utilisent des moteurs à courant continu. Cela modifie la nature de la commande et la nature des états renvoyés par l'environnement. Nous déduisons la position et la vitesse du chariot à partir de la commande, alors que les moteurs à courant continu reçoivent une tension en commande et mesurent indépendamment la position et la vitesse du chariot à l'aide d'un deuxième encodeur rotatif pour se servir de ces informations comme paramètres pour l'apprentissage.
- Ce sont des systèmes avec des jeux de fonctionnement très faibles et de haute précision. Nous avons peu de jeu au niveau de l'engrenage mais il reste non négligeable, ce qui crée de légères vibrations lors du mouvement et rend le système encore plus instable qu'il ne l'est déjà.
- La fréquence à laquelle nous échangeons avec la maquette est inférieure à 20 Hz car la commande en nombre de pas nécessite que le système ait fini sa course. Un de nos contacts a construit un système fonctionnel qui utilisait une fréquence d'actualisation de 100 Hz. Cela peut créer un manque de réactivité du

[5] **Morozs, Nils.** *Control of a Rotary Inverted Pendulum*. James College. 2012.

système qui l'empêche de s'équilibrer. Nous réalisons actuellement une commande en vitesse qui permettra d'augmenter la fréquence d'actualisation.

4.3 Influence des hyper paramètres sur l'apprentissage du Pacman

Une partie importante de la création d'un agent qui apprend par renforcement est la détermination des hypers-paramètres. Ce sont des paramètres qui déterminent le taux d'exploration, le taux d'apprentissage, la décroissance du taux d'exploration etc… Les études faites à ce sujet ne donnent pas de technique théorique pour les déterminer. Il faut donc réaliser plusieurs apprentissages en modifiant ces paramètres pour étudier leur influence et tenter de déterminer les paramètres qui améliorent la vitesse et la précision de l'apprentissage. Ceux-ci dépendent grandement du système utilisé et de ses propres contraintes. N'ayant pas la possiblité de le faire sur le pendule, nous avons fait plusieurs expériences sur le Pacman. Voici les résultats de l'influence du facteur de réduction γ sur l'apprentissage :

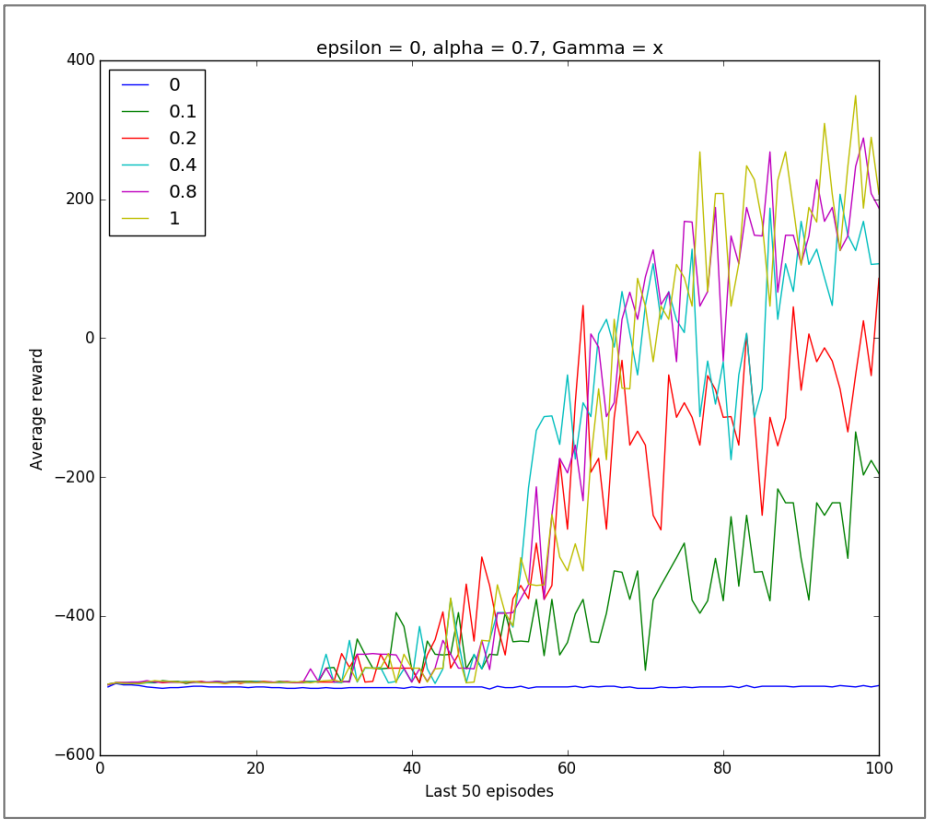


Figure 4.2 *: Graphique montrant l'influence de gamma sur l'apprentissage*

On observe une amélioration dans la vitesse d'apprentissage lorsque le facteur de réduction augmente. On peut cependant observer dans certains autres cas que cela augmente l'instabilité de l'apprentissage. Ces expériences ont été reproduites pour les autres paramètres et nous a permis d'avoir une idée des parametres optimales pour le jeu de Pac-Man. Malgré cela, lorsqu'on applique ces paramètres au pendule, l'apprentissage ne se fait pas :

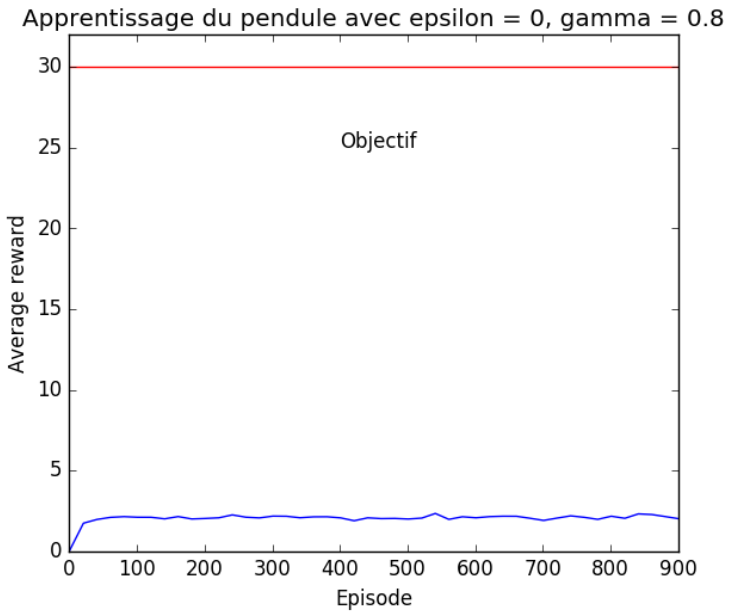


Figure 4.3 *: Apprentissage du pendule avec les paramètres optimaux du Pac-Man. Ici, après 900 épisodes, l'agent n'avait exploré que 156 des 656 couples état-action possibles*

On a donc mis en evidence que les paramètres optimaux sont très sensible à la nature du problème. Il est donc difficile de les identifier.

5 Conclusion

La mise en œuvre de l'apprentissage par renforcement s'est avérée bien plus dificile que nous l'avions imaginé. Face à cette difficulté, nous nous sommes adaptés et nous sommes parvenus à certaines conclusions.

L'échec du pendule inversé nous a amenés à questioner la fiabilité de notre maquette. Après avoir évalué la réponse du pendule, nous avons déterminé que celle-ci est constante. L'échec peut donc être dû à la commande du moteur, au jeu des engrenages, ou à notre fréquence de commande. Mais il nous semble plus probable que ce qui ralentit l'apprentissage est la configuration des hyper-paramètres et le découpage des

états. Bien que nous ayons assimilé la théorie d'apprentissage, il nous manque l'experience nécessaire pour faire le choix empirique des paramètres. Cela ne nous empêche pas de voir une légère progression dans l'apprentissage sur le système réel et sur la simulation.

Le Pac-Man est une application plus simple sur laquelle nous avons réussi à mettre en œuvre l'apprentissage par renforcement. Sans connaître les règles du jeu, l'agent apprend avec succès comment gagner la majorité des parties. Nous avons pu analyser l'influence des paramètres et montrer qu'ils sont très sensibles aux caractéristiques du problème.

Les derniers algorithmes d'appretissage de renforcement cherchent à incorporer les réseaux de neurons. Ces structures permettent d'accélérer l'apprentissage et de chercher rapidement des solutions lorsque le problème présente un nombre infini d'etats et/ou actions. Aujourd'hui, les applications de l'apprentissage par renforcement sont assez variées, telles que l'optimisation des flux de voitures dans les villes[6], la réalisation de jeux de stratégie (échecs, Go[7], etc.) et l'amélioration de la gestion de chaînes de logistique[8].

---

[6] **Wiering, Marco A.** *Multi-Agent Reinforcement Learning for Traffic Light Control.* Utrecht : University of Utrecht, 2000.

[7] **Silver, David, Sutton, Richard et Muller, Martin.** *Reinforcement Learning of Local Shape in the Game of Go.* Alberta : University of Alberta, 2007.

[8] **Schwind, Michael, Stockheim, Tim et Koenig, Wolfgang.** *A reinforcement learning approach for supply chain management.* Frankfurt : Frankfurt University, 2015.

6 Bibliographie

1. **Poole, David et Mackworth, Alan.** Q-learning. *Artificial Intelligence - Foundations of computational agents.* [En ligne] 2010. [Citation : 10 Avril 2016.] http://artint.info/.

2. **Watkins, Christopher J.C.H.** *Learning from Delayed Rewards.* King's College. 1989.

3. **Watkins, Christopher J.C.H. et Dayan, Peter.** *Q-learning - Machine Learning*. Boston : C.J.C.H., 1992.

4. **Silver, David.** *Advanced Topics: RL.* [En ligne] [Citation : 9 Decembre 2015.] http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching.html.

5. **Morozs, Nils.** *Control of a Rotary Inverted Pendulum*. James College. 2012.

6. **Wiering, Marco A.** *Multi-Agent Reinforcement Learning for Traffic Light Control*. Utrecht : University of Utrecht, 2000.

7. **Silver, David, Sutton, Richard et Muller, Martin.** *Reinforcement Learning of Local Shape in the Game of Go*. Alberta : University of Alberta, 2007.

8. **Schwind, Michael, Stockheim, Tim et Koenig, Wolfgang.** *A reinforcement learning approach for supply chain management.* Frankfurt : Frankfurt University, 2015.

9. **Russell, Stuart J., Norvig, Perter et Ernest, Davis.** *Artificial intelligence : a modern approach*. Upper Saddle River : Prentice Hall, 2010.

10. **Dewolf, Travis.** REINFORCEMENT LEARNING PART 1: Q-LEARNING AND EXPLORATION. *Studywolf.* [En ligne] 25 Novembre 2015. [Citation : 18 Janvier 2016.] https://studywolf.wordpress.com/2012/11/25/reinforcement-learning-q-learning-and-exploration/.

7 Annexes

7.1 Fiche synoptique

Le thème des T.I.P.E cette année est intitulé « Structure : organisation, complexité, dynamique ». Nous avons donc choisi de travailler sur l'apprentissage par renforcement, qui est un sous-domaine de l'apprentissage et de l'intelligence artificielle. Notre sujet s'inscrit dans le thème par la façon dont fonctionne l'apprentissage. Pour apprendre, il est nécessaire d'organiser et de traiter des données qui évoluent dans le temps de façon à les faire évoluer vers l'objectif que l'on donne au système. Nous nous sommes donné la problématique suivante :
Quels sont les moyens à mettre en œuvre pour qu'un système apprenne par l'expérience à évoluer vers un objectif défini ?

Toute l'année nous avons chacun pris la responsabilité de s'occuper de différentes parties du projet en échangeant sur l'évolution de ce que faisait l'autre. La première partie de l'année a été consacré à la réalisation de la maquette pour Thomas et à l'exploration de la théorie pour Ludovic (qui avait été exploré pour nous deux avant le début de l'année compte tenu de la complexité du sujet). Pendant que la réalisation de la maquette se finissait et que nous réglions le premier problème important que fût l'élévation de la température de la carte de contrôle moteur, Ludovic a commencé à mettre en place l'algorithme. Une fois la maquette terminée et la première version du programme faite, nous avons commencé à expérimenter. En vue des difficultés que nous commencions à apercevoir, nous avons commencé à chercher des personnes connaissant le sujet et pouvant nous aider. Il y a malheureusement peu de personnes françaises ayant mis en pratique cette expérience, alors que nous nous sommes rendus compte que les différents paramètres à ajuster étaient une question d'expérience et de mise en pratique. Nous avons alors contacté Nils Morozs, étudiant anglais que nous remercions pour son aide sachant qu'il a réussi à mettre en œuvre un pendule inversé fonctionnel.
Face à la difficulté à compléter l'apprentissage du pendule sur simulation et sur la maquette, nous avons décidé de mettre en œuvre l'apprentissage sur la plateforme fournie par l'université de Berkeley. Cela nous a permis de vérifier la bonne mise en pratique de notre algorithme et de pouvoir étudier l'influence des différents paramètres sur l'apprentissage. Ludovic s'est occupé de faire fonctionner notre

algorithme sur le jeux de Pac-Man en utilisant la plateforme de Berkeley pendant que Thomas continuait d'expérimenter des nouveautés sur le pendule.

Pour la fin de l'année, nous prévoyons de continuer nos expériences et espérons qu'en faisant encore évoluer progressivement le fonctionnement de notre pendule, nous arriverons à un résultat plus concluant qu'actuellement.

Aux Lazaristes
Lyon - La Salle

Classes préparatoires à l'ECAM

Objet : attestation de présence

Noms	**Bouan, Lefort**
Prénoms	**Ludovic, Thomas**
des étudiants	
Classe	☒ **PT 1** ☐ **PT2** ☐ **PT3** ☐ **PTSI1** ☐ **PTSI2** ☐ **PTSI3**
Motif de la visite :	Conseil sur la mise en œuvre du pendule inversé et mise en évidence des difficultés.
Date de la visite :	16 Avril 2016
Horaires :	16h00
Adresse :	Sans Objet

Personne contactée

Nom :	Putz
Prénom :	Anne-Laurence
Fonction :	Enseignante - Option Sciences Cognitives et Informatique Avancée
Société :	EPITA
Adresse	14 rue Voltaire 94276 Le Kremlin-Bicêtre CEDEX

Fait à Paris le 16 avril 2016

EPITA
14-16 rue Voltaire
94276 LE KREMLIN-BICÊTRE CEDEX

Signatures des élèves

signature de la personne contactée et cachet de la société

Document à joindre au rapport de TIPE

Feuille de route

Thème : Structure : organisation, complexité, dynamique

Binôme :	BOUAN - LEFORT
Sujet :	Apprentissage par renforcement

Date de la séance	Actions réalisées DEPUIS la dernière séance	Actions réalisées PENDANT la séance	Actions PRÉVUES pour la prochaine séance	Visa du jury	Jalons
09-sept	----------	Présentation général en classe entière	chercher un sujet le faire valider par les professeurs		
23-sept	Début de conception Recherche bibliographique	Conception Recherche capteur	Conception Choix d'algorithme Recherche de contact		
7-oct	Conception Choix algorithme + defini agent Contact + choix pièces	Conception Comptabilité des pièces	Conception Finir agent Calcul dynamique		
					Choix définitif du sujet
Vacances de Toussaint (21/10 - 3/11)					
4-nov	Conception Usinage Contact	Modelisation Avancer agent	Assemblage Modélisation		
25-nov	Recherche de moteur + achat Conception	Debut PFD Bibliographie	Finir PFD Avancer Agent Finir Bibliographie		
9-dec	Pièces imprimées Bibliographie moteur acheté	Programation PFD tapé sur ordinateur	Faire le montage		Remise de la bibliographie
Vacances de Noël (23/12 - 5/01)					

Feuille de route

Thème : Structure : organisation, complexité, dynamique

Binôme :	
Sujet :	

Date de la séance	Actions réalisées DEPUIS la dernière séance	Actions réalisées PENDANT la séance	Actions PRÉVUES pour la prochaine séance	Visa du jury	Jalons
6-jan	Prototype fini	Programmation	Programmation Chercher problème chaleur		
20-jan	Programmation finie	Chercher problème chaleur	Chercher problème chaleur		
3-fev	Regler problème chauffe Debug code	Chercher pourquoi il s'arrete	Faire évaluer le fonctionnement		Remise de la fiche synoptique
Vacances d'hiver (17/02 - 1/03)					
2-mar	Test + reparation Ameliorations	Retravailler théorie (Regression linéaire)	Théorie apprentissage		
16.mars	Restructuration du code contact Tests	Pareil	Fin restructuration Nouveaux tests		
30 mars	Simulation Blender Restructuration Dossier	Simulation Blender	Dossier		Remise du rapport
Vacances de Pâques (19/04 - 4/05)					
	Soutenances				