Développeur d'Intelligence Artificielle Appliquée

Cours #6

www.impactia.org

Structure de la formation

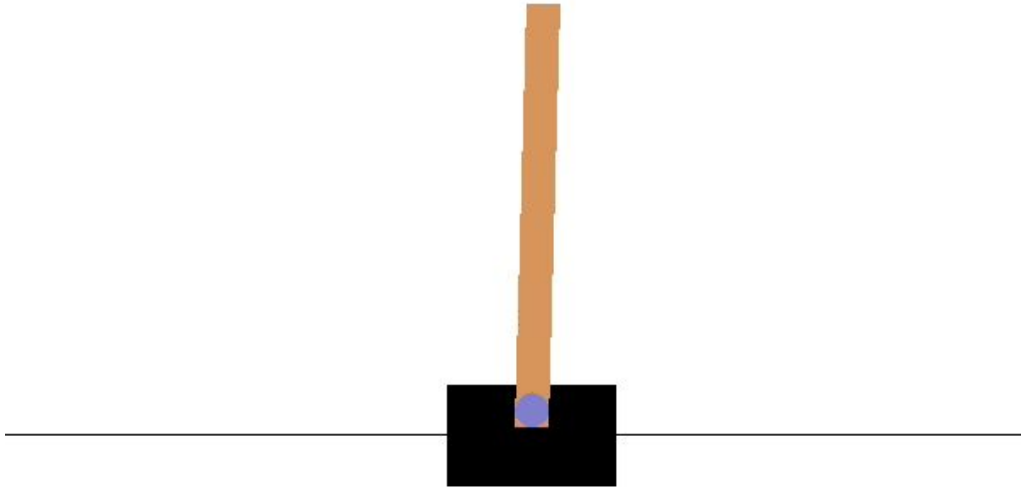
- **#1: Introduction**
- **#2: Vision**
- **#3: Vision**
- **#4: Vision**
- **#5: Renforcement**
- **#6: Renforcement**
- #7: Renforcement
- #8: Langage
- #9: Langage
- #10: Langage
- # 11: Outillage
- #12: Génération d’images
- #13: Génération d’images
- #14: Projet
- #15: Projet

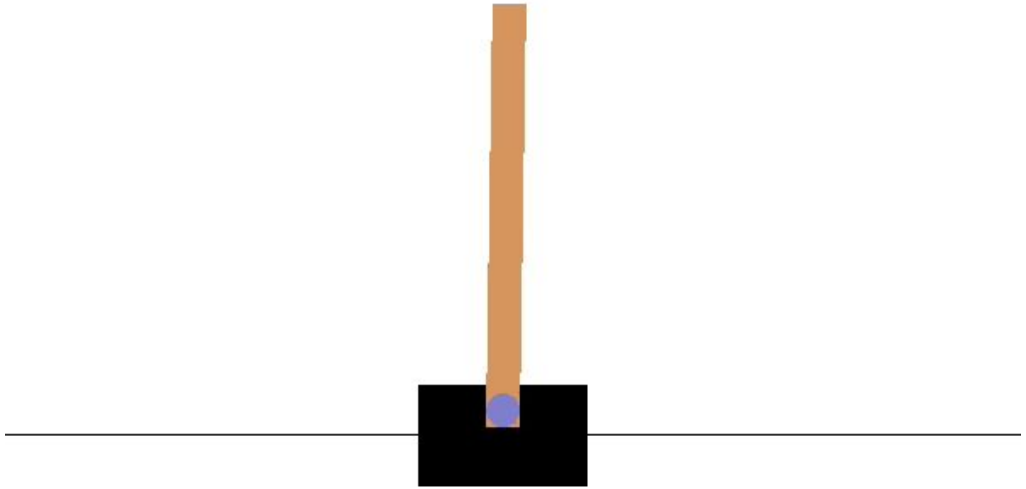
Semaine dernière : Cours #4

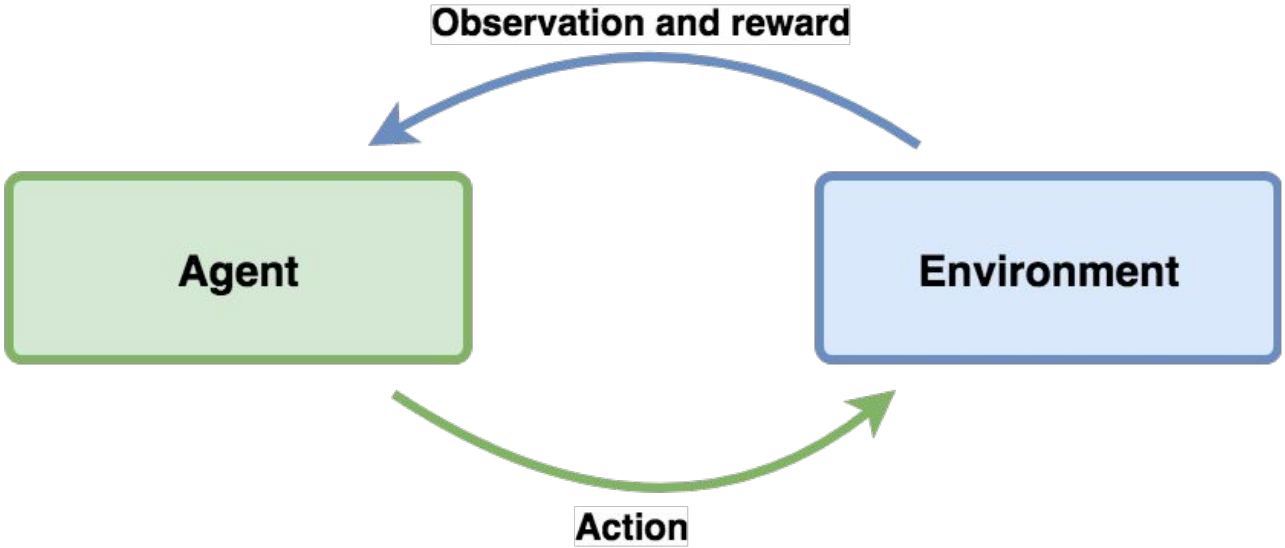
- Théorie
  - Introduction à l'apprentissage par renforcement
  - Que peut-on faire avec ?
  - Pourquoi est-ce difficile ?
- Pratique
  - L'environnement droneRL

Aujourd’hui : Cours #5

- Théorie
  - Q-learning
  - Deep Q-Network (DQN)
- Pratique
  - Entrainement d’agents droneRL
  - Premières participations au Challenge droneRL







But :
Maximiser la somme des *récompenses* (“rewards”)
jusqu’à la fin de l’*épisode*

The Promise of RL

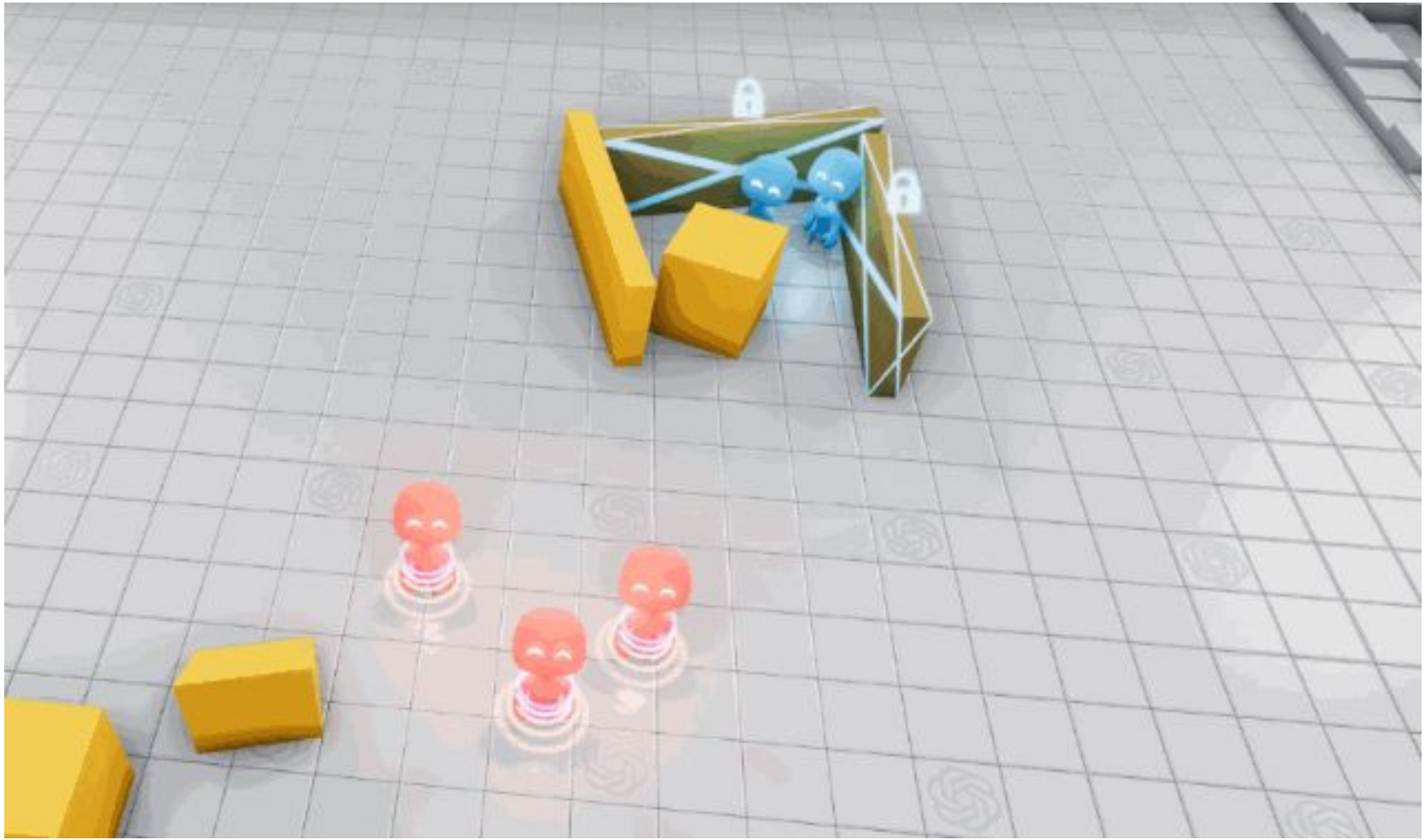
Specify **what** to do, but not **how** to do it

- Through the reward function
- Learning “fills in the details”

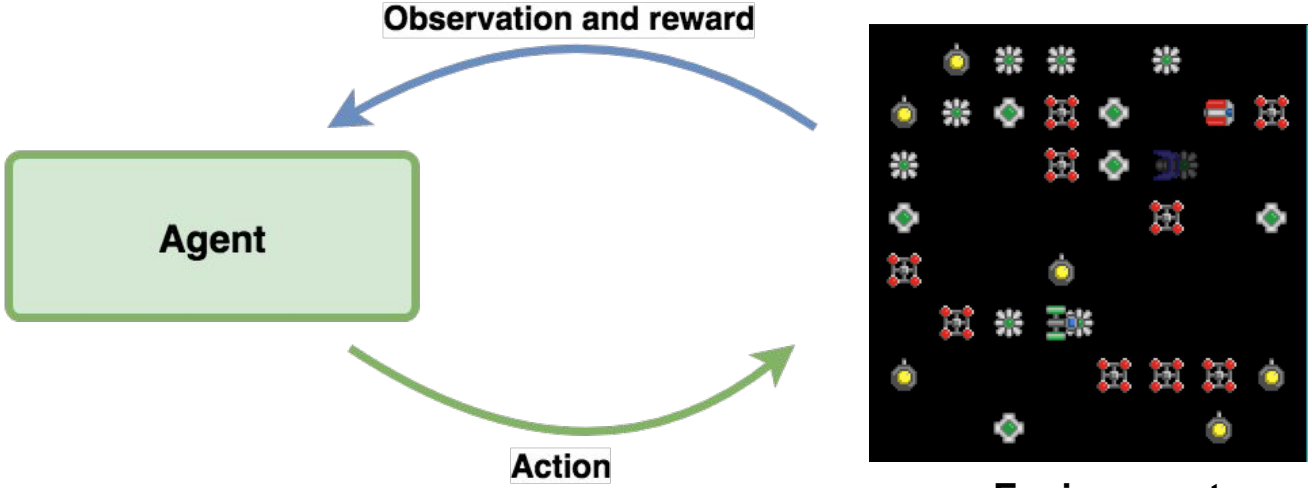
Better final solutions

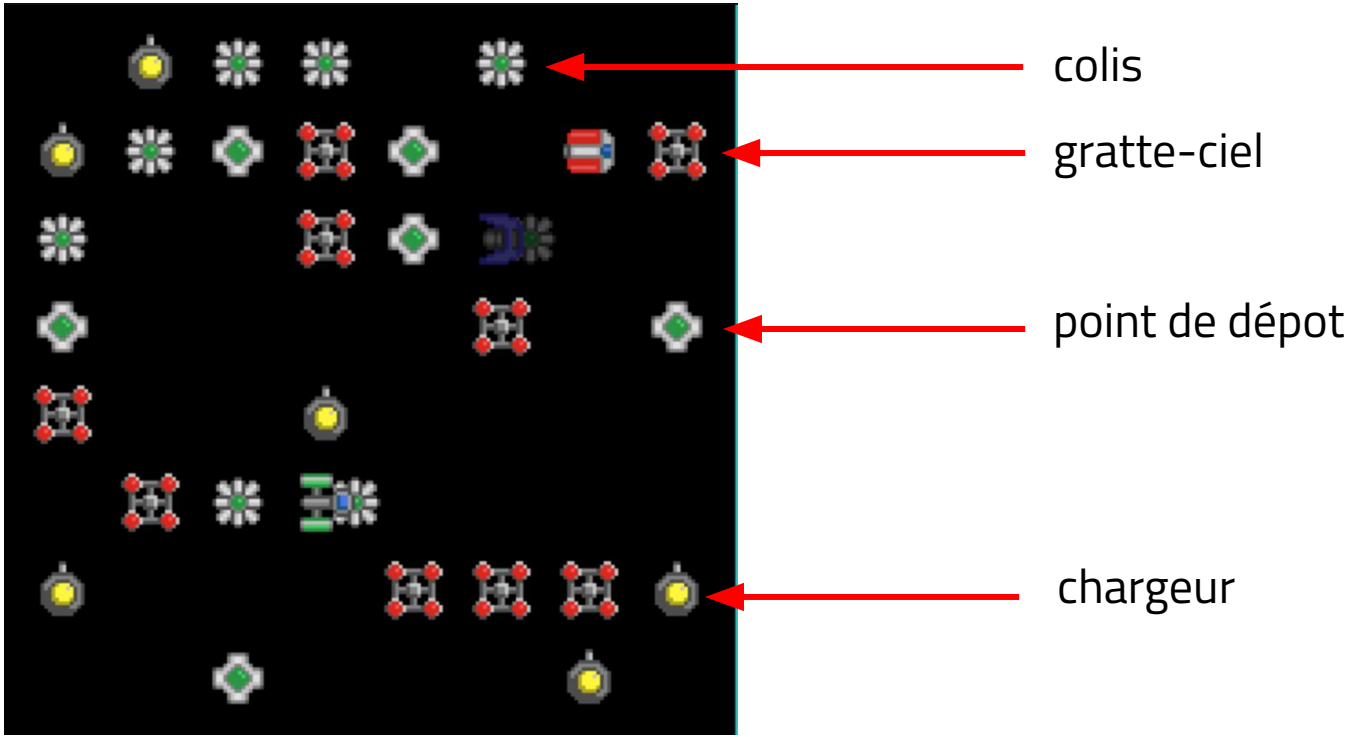
- Based of actual experiences, not programmer assumptions

Less (human) time needed for a good solution

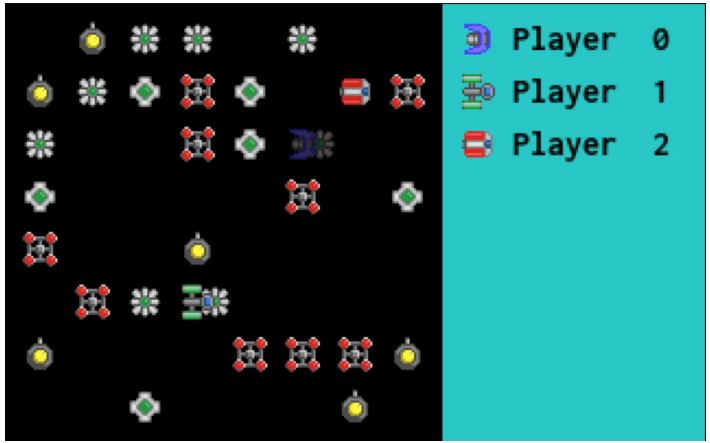


L'environnement *DeliveryDrones*


Observation and reward
Agent
Action
Environment


colis
gratte-ciel
point de dépot
chargeur

```
self.env_params = {
    'drone_density': 0.05,
    'n_drones': 3,
    'pickup_reward': 0,
    'delivery_reward': 1,
    'crash_reward': -1,
    'charge_reward': -0.1,
    'discharge': 10,
    'charge': 20,
    'packets_factor': 3,
    'dropzones_factor': 2,
    'stations_factor': 2,
    'skyscrapers_factor': 3,
    'rgb_render_rescale': 1.0
}
```



Mise en pratique

L'environnement DeliveryDrones

https://colab.research.google.com/drive/18R9mQPDRv-rhVons1IEPjmCCiP7sjhsz

Comment “apprendre un comportement” ?

Q-learning

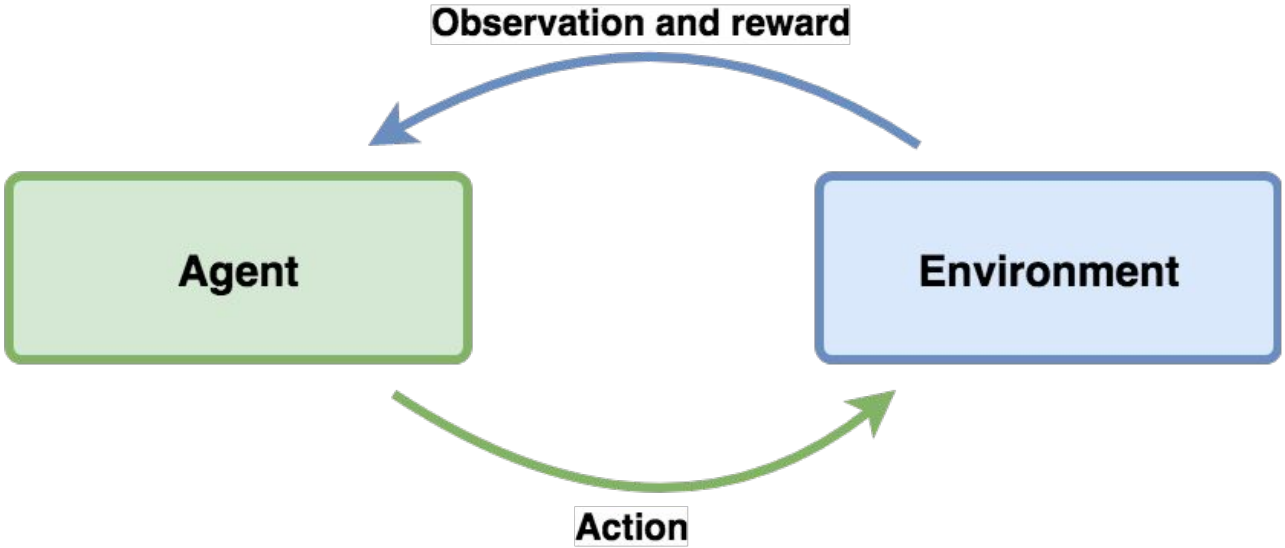
Apprendre un comportement

Une information à disposition

↖

5 actions possibles - que faire ?

LEFT **DOWN** **RIGHT** **UP** **STAY**



But :
Maximiser la somme des *récompenses* (“rewards”)
jusqu’à la fin de l’*épisode*

Observation

Actions

	LEFT	DOWN	RIGHT	UP	STAY
→	1.7	1.4	2.6	0.86	1.5
↘	1.3	1.6	2.4	0.78	1.5
↑	1.9	1.6	1.4	2.5	1.7
↗	1.4	0.58	1.8	1.6	1.6
↖	2.2	1.3	0.97	1.6	1.5
↙	2.5	2	1.3	0.9	1.5
↓	1.8	2.9	1.7	1.2	2.1
←	2.6	1.5	2.1	1.2	2.2

La “solution” d’un environnement

Si on connaît les valeurs de ce tableau,
il est facile de suivre un comportement optimal !

Actions

Observation

	LEFT	DOWN	RIGHT	UP	STAY
→	1.7	1.4	2.6	0.86	1.5
↘	1.3	1.6	2.4	0.78	1.5
↑	1.9	1.6	1.4	2.5	1.7
↗	1.4	0.58	1.8	1.6	1.6
↖	2.2	1.3	0.97	1.6	1.5
↙	2.5	2	1.3	0.9	1.5
↓	1.8	2.9	1.7	1.2	2.1
←	2.6	1.5	2.1	1.2	2.2

“Q-table”

La “solution” d’un environnement

Q(observation, action) = somme des récompenses attendues

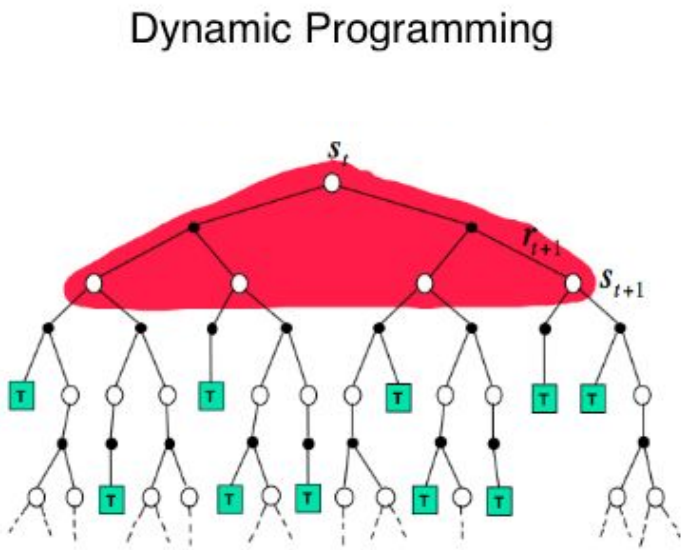
Actions

Observation

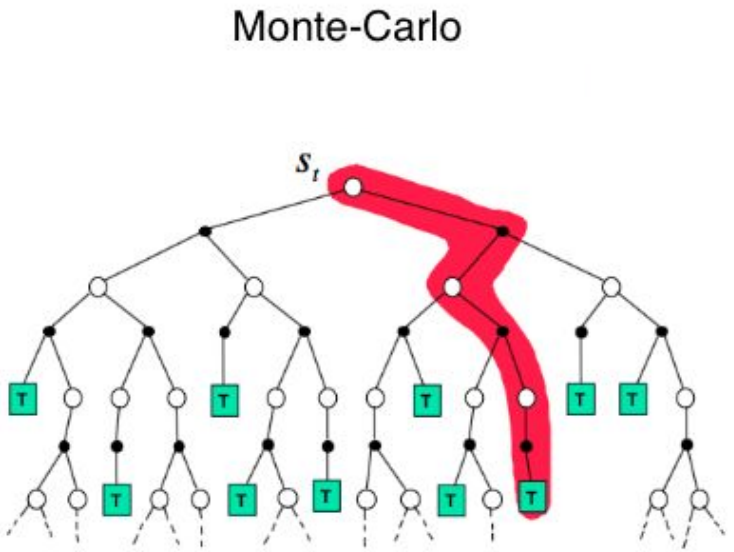
	LEFT	DOWN	RIGHT	UP	STAY
→	1.7	1.4	2.6	0.86	1.5
↘	1.3	1.6	2.4	0.78	1.5
↑	1.9	1.6	1.4	2.5	1.7
↗	1.4	0.58	1.8	1.6	1.6
↖	2.2	1.3	0.97	1.6	1.5
↙	2.5	2	1.3	0.9	1.5
↓	1.8	2.9	1.7	1.2	2.1
←	2.6	1.5	2.1	1.2	2.2

“Q-table”

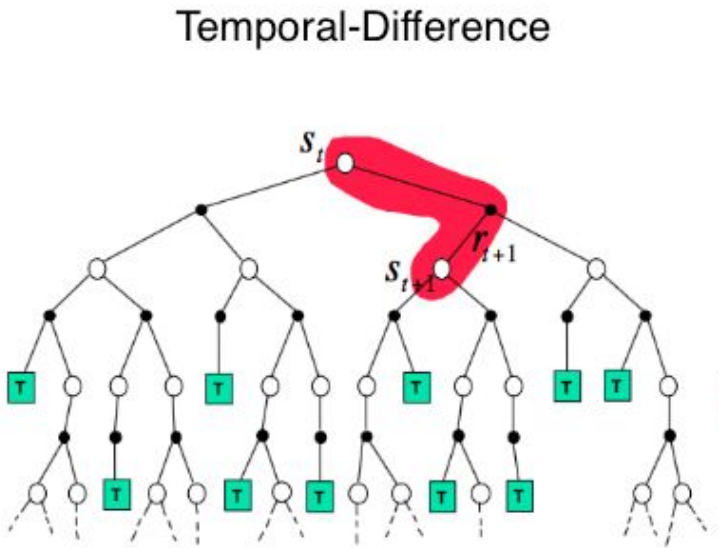
Comment apprendre la Q-table ?



Comment apprendre la Q-table ?



Comment apprendre la Q-table ?



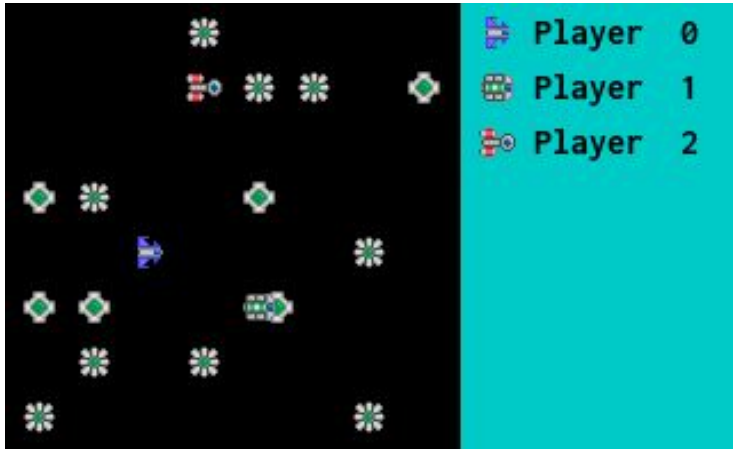
Q-learning

Actions

Observation

	LEFT	DOWN	RIGHT	UP	STAY
→	1.7	1.4	2.6	0.86	1.5
↘	1.3	1.6	2.4	0.78	1.5
↑	1.9	1.6	1.4	2.5	1.7
↗	1.4	0.58	1.8	1.6	1.6
↖	2.2	1.3	0.97	1.6	1.5
↙	2.5	2	1.3	0.9	1.5
↓	1.8	2.9	1.7	1.2	2.1
←	2.6	1.5	2.1	1.2	2.2

“Q-table”
Somme attendue des récompenses

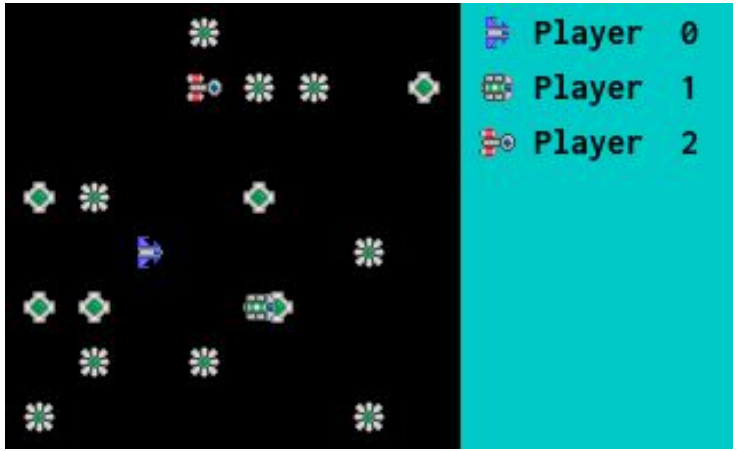


Actions

Observation

	LEFT	DOWN	RIGHT	UP	STAY
→	1.7	1.4	2.6	0.86	1.5
↘	1.3	1.6	2.4	0.78	1.5
↑	1.9	1.6	1.4	2.5	1.7
↗	1.4	0.58	1.8	1.6	1.6
↖	2.2	1.3	0.97	1.6	1.5
↙	2.5	2	1.3	0.9	1.5
↓	1.8	2.9	1.7	1.2	2.1
←	2.6	1.5	2.1	1.2	2.2

“Q-table”
Somme attendue des récompenses



Actions

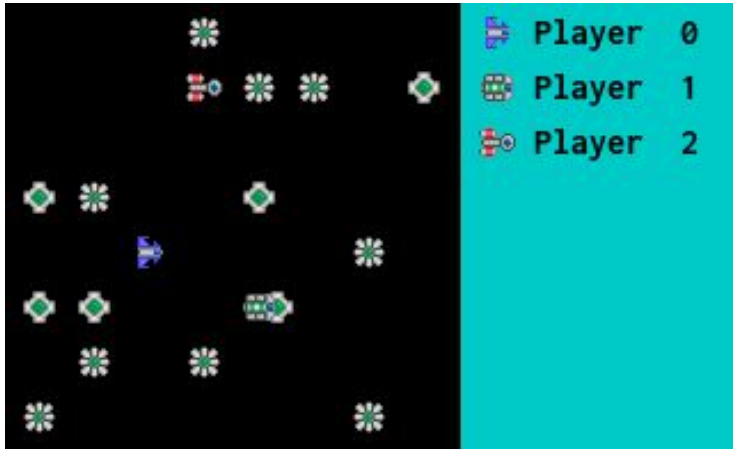
Observation

	LEFT	DOWN	RIGHT	UP	STAY
→	1.7	1.4	2.6	0.86	1.5
↘	1.3	1.6	2.4	0.78	1.5
↑	1.9	1.6	1.4	2.5	1.7
↗	1.4	0.58	1.8	1.6	1.6
↖	2.2	1.3	0.97	1.6	1.5
↙	2.5	2	1.3	0.9	1.5
↓	1.8	2.9	1.7	1.2	2.1
←	2.6	1.5	2.1	1.2	2.2

“Q-table”
Somme attendue des récompenses

On sélectionne l’action qui maximise la somme attendue des récompenses

$$a = \text{argmax}_a Q(s, a)$$



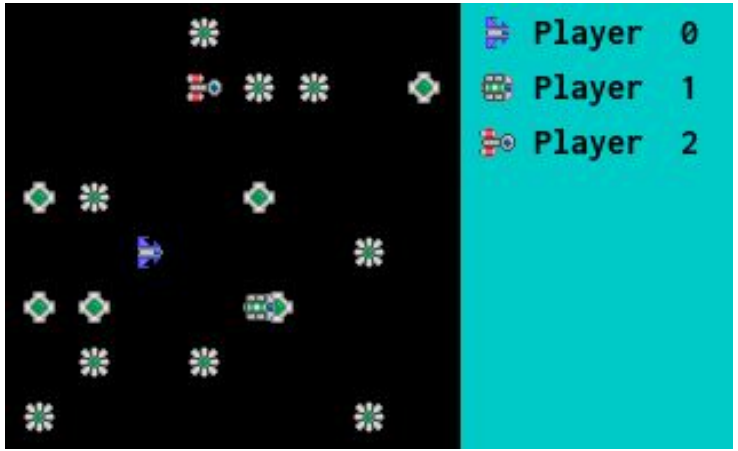
Actions

Observation	LEFT	DOWN	RIGHT	UP	STAY
→	1.7	1.4	2.6	0.86	1.5
↘	1.3	1.6	2.4	0.78	1.5
↑	1.9	1.6	1.4	2.5	1.7
↗	1.4	0.58	1.8	1.6	1.6
↖	2.2	1.3	0.97	1.6	1.5
↙	2.5	2	1.3	0.9	1.5
↓	1.8	2.9	1.7	1.2	2.1
←	2.6	1.5	2.1	1.2	2.2

“Q-table”
Somme attendue des récompenses

On sélectionne l’action qui maximise la somme attendue des récompenses

$$a = \text{argmax}_a Q(s, a)$$



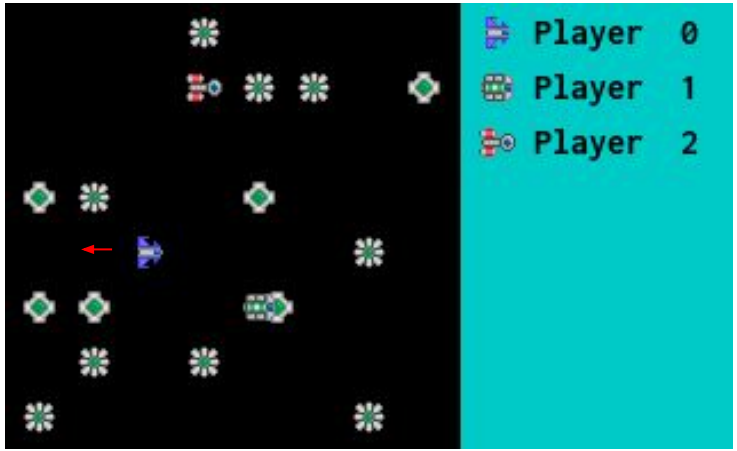
On sélectionne l'action qui maximise la somme attendue des récompenses

$$a = \text{argmax}_a Q(s, a)$$

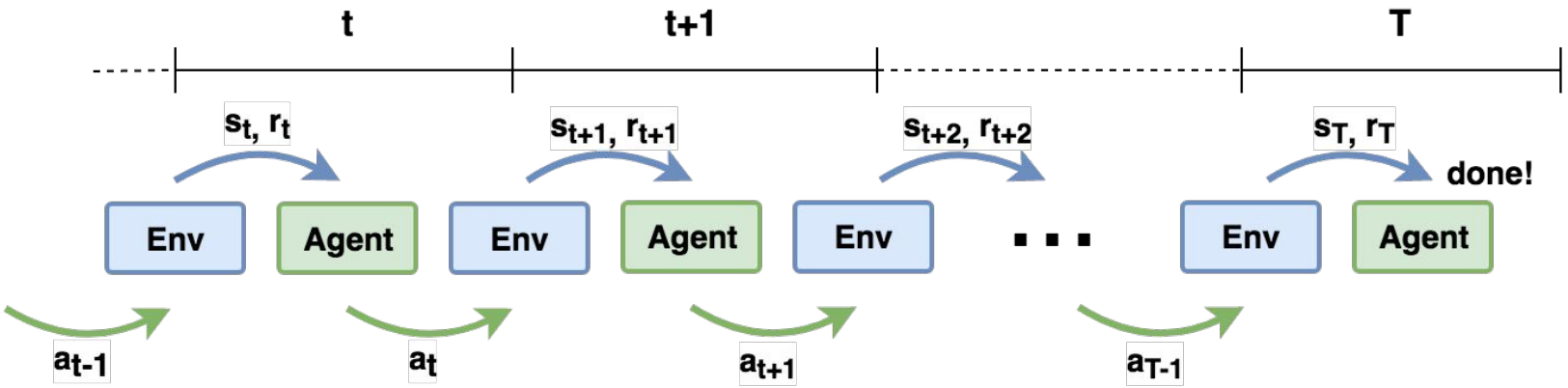
Actions

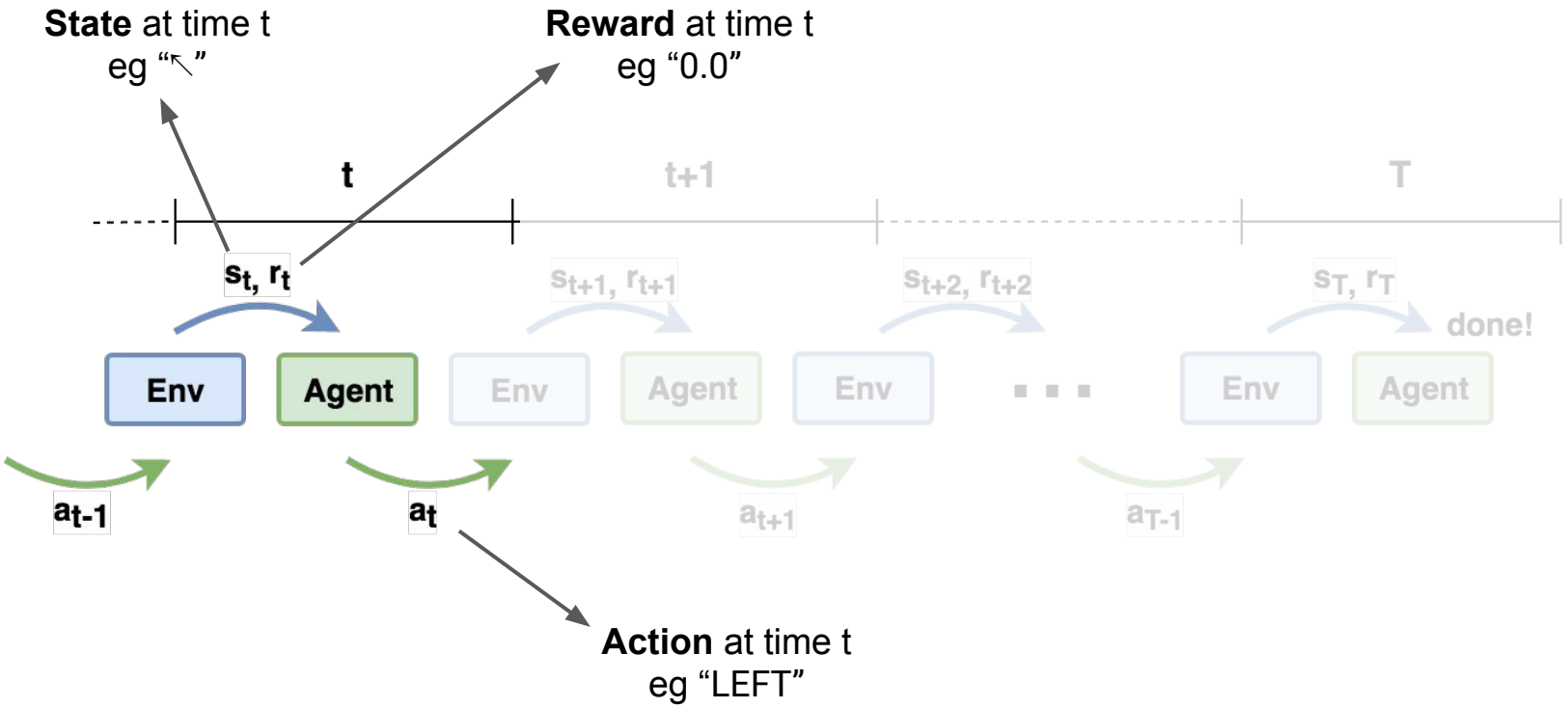
Observation

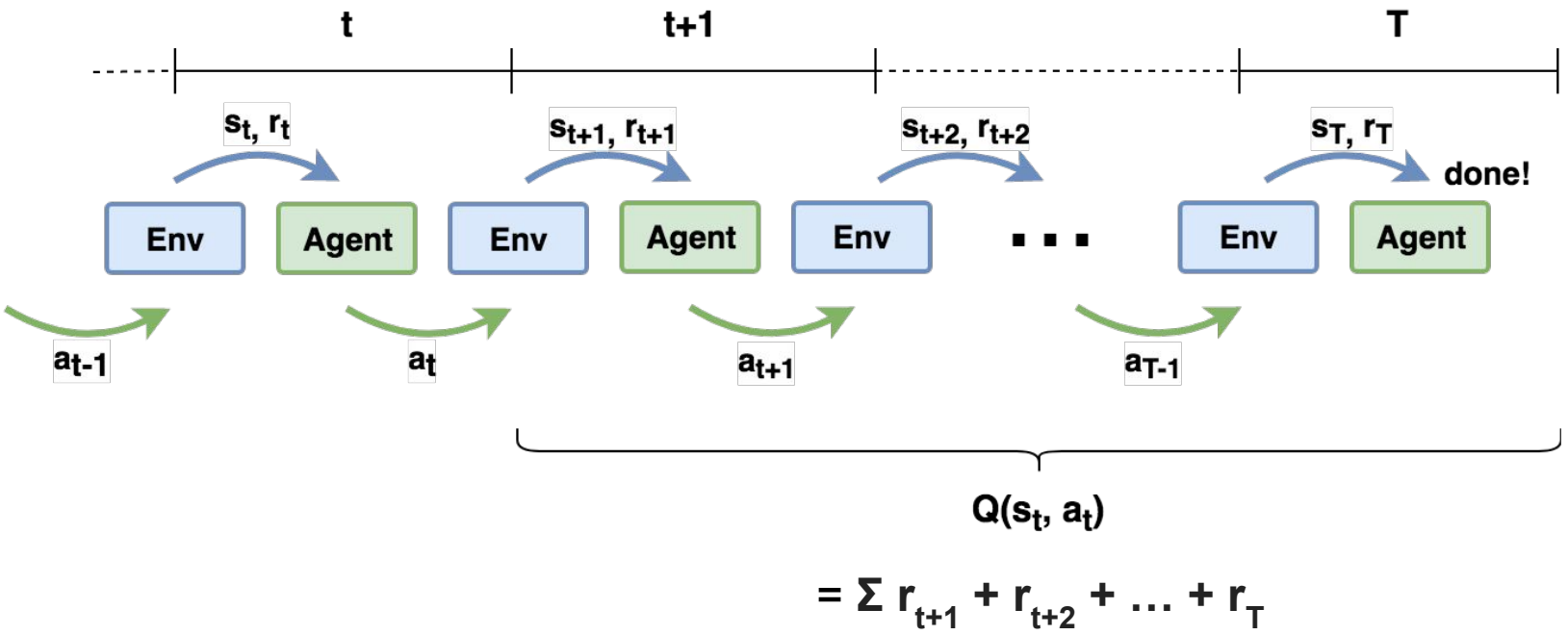
	LEFT	DOWN	RIGHT	UP	STAY
→	1.7	1.4	2.6	0.86	1.5
↘	1.3	1.6	2.4	0.78	1.5
↑	1.9	1.6	1.4	2.5	1.7
↗	1.4	0.58	1.8	1.6	1.6
↖	2.2	1.3	0.97	1.6	1.5
↙	2.5	2	1.3	0.9	1.5
↓	1.8	2.9	1.7	1.2	2.1
←	2.6	1.5	2.1	1.2	2.2

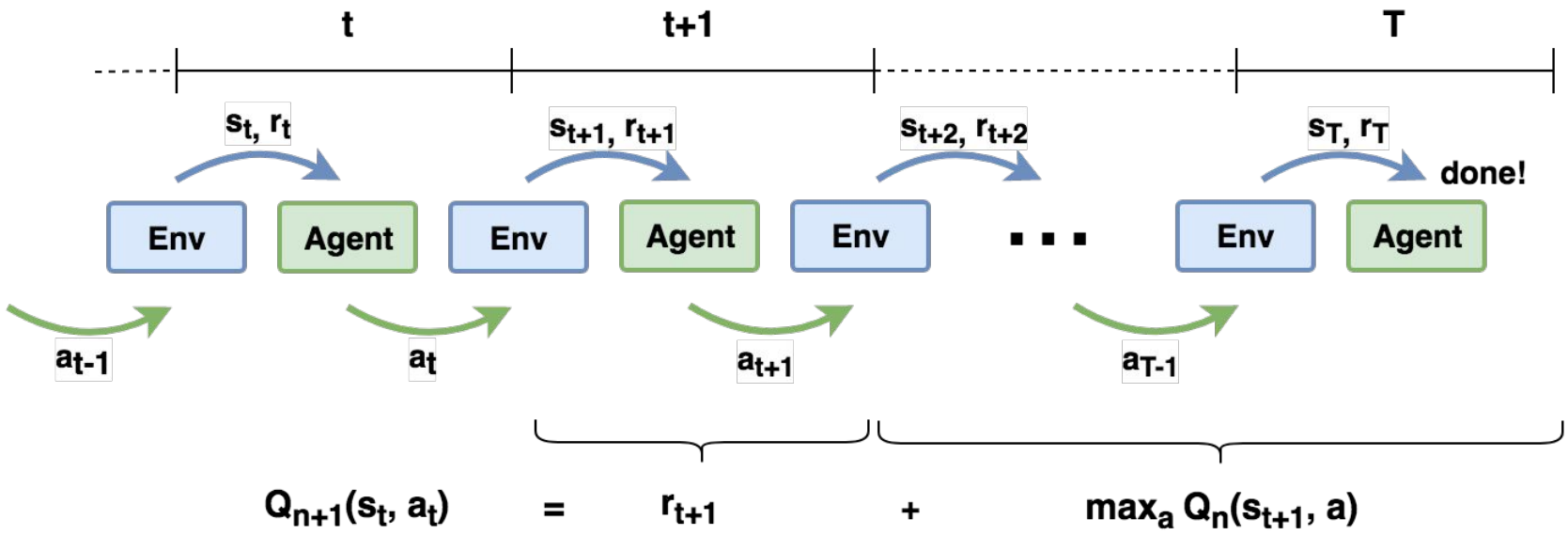


"Q-table"
Somme attendue des récompenses


t
t+1
T
s_t, r_t
s_{t+1}, r_{t+1}
s_{t+2}, r_{t+2}
s_T, r_T
done!
Env
Agent
Env
Agent
Env
Env
Agent
a_{t-1}
a_t
a_{t+1}
a_{T-1}


State at time t
eg "↖"
Reward at time t
eg "0.0"
t
t+1
T
s_t, r_t
s_{t+1}, r_{t+1}
s_{t+2}, r_{t+2}
s_T, r_T
done!
Env
Agent
Env
Agent
Env
Env
Agent
a_{t-1}
a_t
a_{t+1}
a_{T-1}
Action at time t
eg "LEFT"


t
t+1
T
s_t, r_t
s_{t+1}, r_{t+1}
s_{t+2}, r_{t+2}
s_T, r_T
done!
Env
Agent
Env
Agent
Env
Env
Agent
a_{t-1}
a_t
a_{t+1}
a_{T-1}
$Q(s_t, a_t)$
$= \Sigma\ r_{t+1} + r_{t+2} + \ldots + r_T$

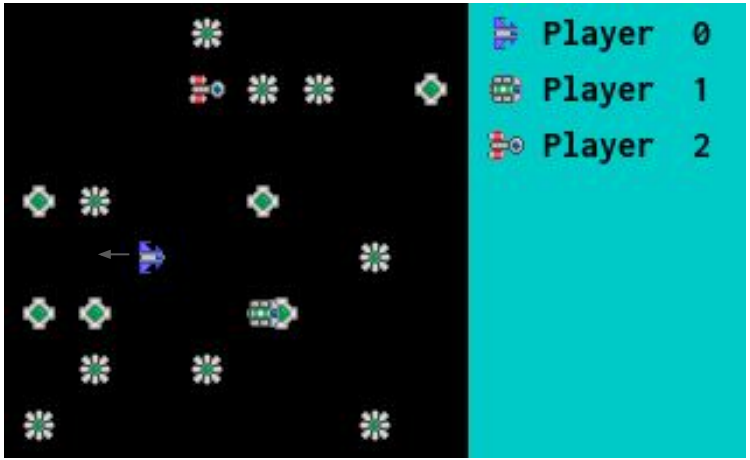

t
t+1
T
s_t, r_t
s_{t+1}, r_{t+1}
s_{t+2}, r_{t+2}
s_T, r_T
done!
Env
Agent
Env
Agent
Env
Env
Agent
a_{t-1}
a_t
a_{t+1}
a_{T-1}
$Q_{n+1}(s_t, a_t) = r_{t+1} + \max_a Q_n(s_{t+1}, a)$

Q-learning

“Equation de Bellman”

$$Q_{n+1}(s_t, a_t) = r_{t+1} + \gamma \max_a Q_n(s_{t+1}, a)$$

	LEFT	DOWN	RIGHT	UP	STAY
→	1.7	1.4	2.6	0.86	1.5
↘	1.3	1.6	2.4	0.78	1.5
↑	1.9	1.6	1.4	2.5	1.7
↗	1.4	0.58	1.8	1.6	1.6
↖	2.2	1.3	0.97	1.6	1.5
↙	2.5	2	1.3	0.9	1.5
↓	1.8	2.9	1.7	1.2	2.1
←	2.6	1.5	2.1	1.2	2.2



Q-learning

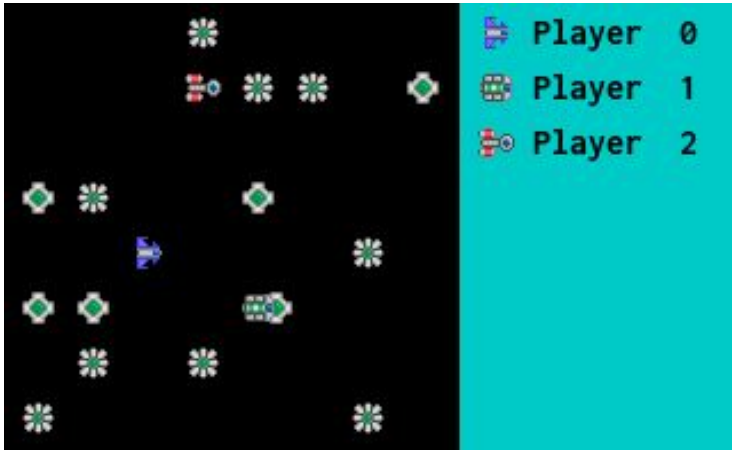
Actions

States

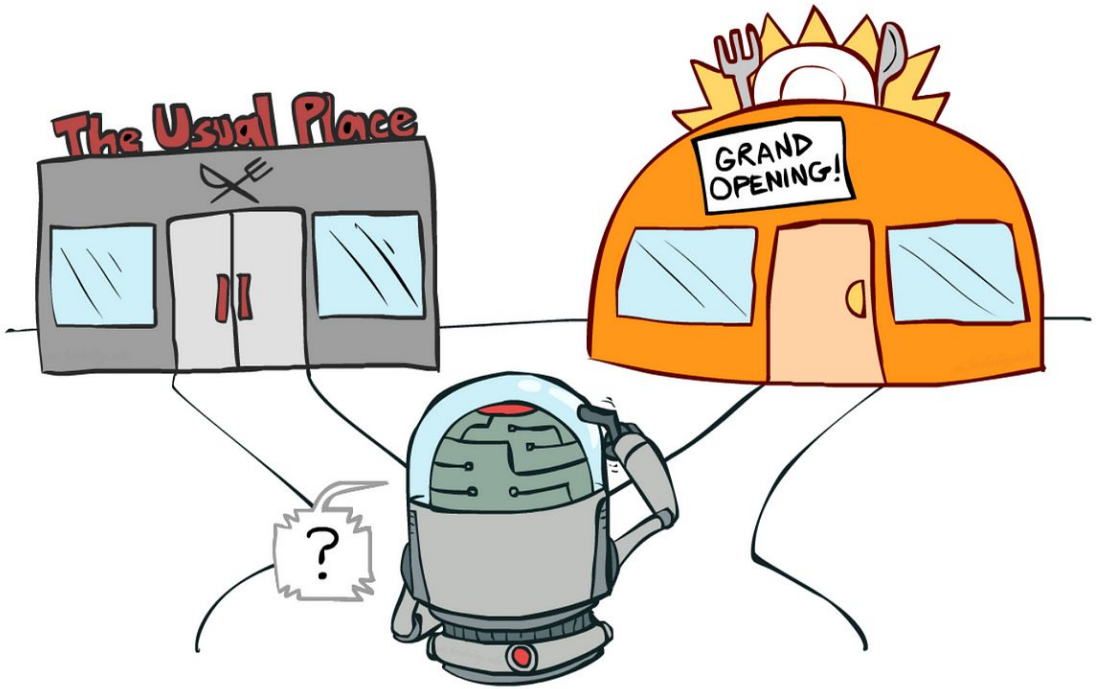
	LEFT	DOWN	RIGHT	UP	STAY
→	1.7	1.4	2.6	0.86	1.5
↘	1.3	1.6	2.4	0.78	1.5
↑	1.9	1.6	1.4	2.5	1.7
↗	1.4	0.58	1.8	1.6	1.6
↖	2.2	1.3	0.97	1.6	1.5
↙	2.5	2	1.3	0.9	1.5
↓	1.8	2.9	1.7	1.2	2.1
←	2.6	1.5	2.1	1.2	2.2

$a = \text{argmax}_a Q(s_t, a)$

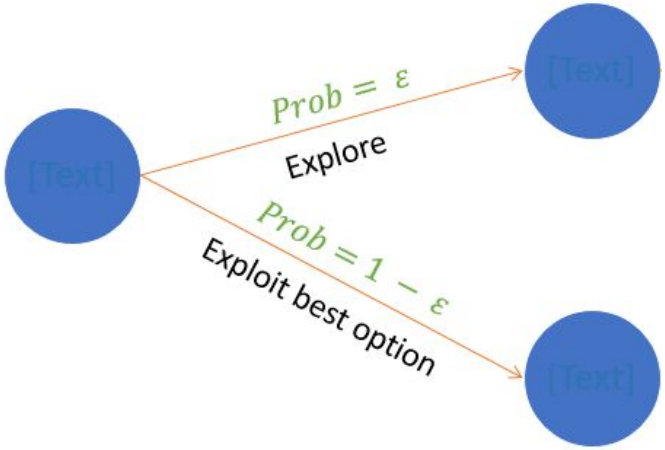
Equation de Bellman



“Explorer” ou “exploiter” ?



Explorer… parfois !
Méthode “epsilon-greedy”



Q-learning

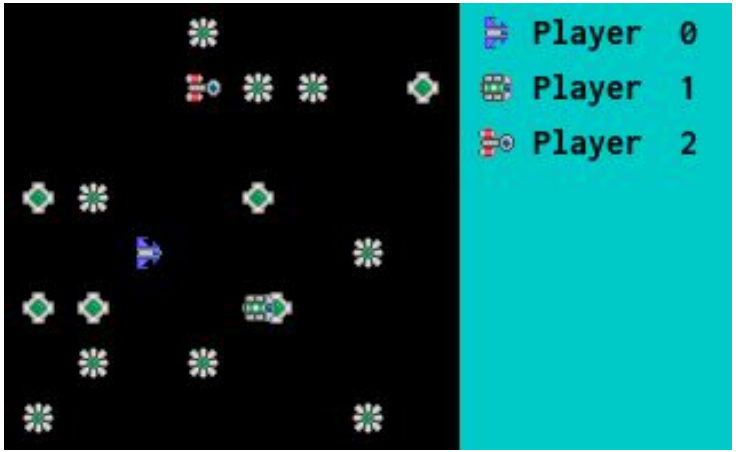
Actions

States

	LEFT	DOWN	RIGHT	UP	STAY
→	1.7	1.4	2.6	0.86	1.5
↘	1.3	1.6	2.4	0.78	1.5
↑	1.9	1.6	1.4	2.5	1.7
↗	1.4	0.58	1.8	1.6	1.6
↖	2.2	1.3	0.97	1.6	1.5
↙	2.5	2	1.3	0.9	1.5
↓	1.8	2.9	1.7	1.2	2.1
←	2.6	1.5	2.1	1.2	2.2

$a = \text{argmax}_a Q(s_t, a)$
+
epsilon-greedy

Bellman Equation



Q-learning

Fonctionne dans tous les cas !*

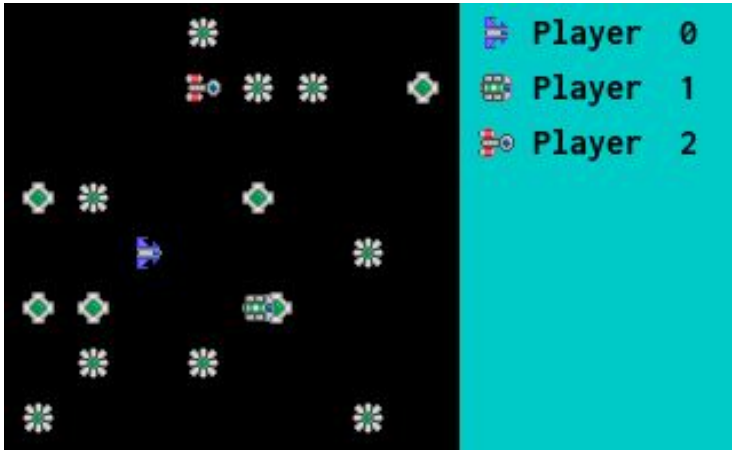
Actions

States

	LEFT	DOWN	RIGHT	UP	STAY
→	1.7	1.4	2.6	0.86	1.5
↘	1.3	1.6	2.4	0.78	1.5
↑	1.9	1.6	1.4	2.5	1.7
↗	1.4	0.58	1.8	1.6	1.6
↖	2.2	1.3	0.97	1.6	1.5
↙	2.5	2	1.3	0.9	1.5
↓	1.8	2.9	1.7	1.2	2.1
←	2.6	1.5	2.1	1.2	2.2

$a = \text{argmax}_a\, Q(s_t, a)$
+
epsilon-greedy

Bellman Equation



Q-learning

Fonctionne dans tous les cas !*

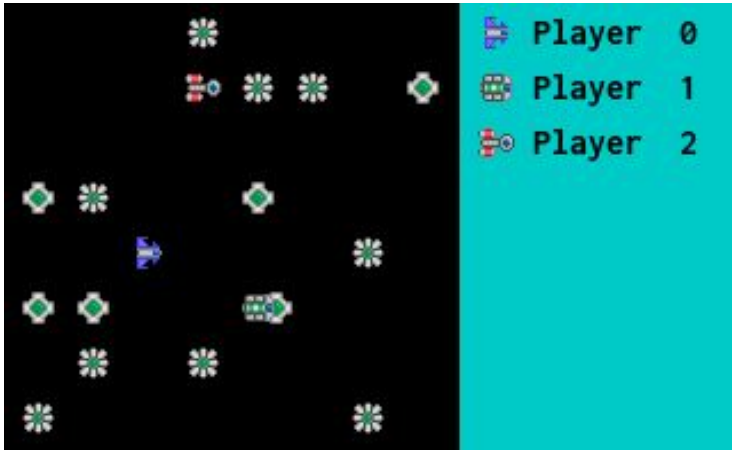
Actions

States

	LEFT	DOWN	RIGHT	UP	STAY
→	1.7	1.4	2.6	0.86	1.5
↘	1.3	1.6	2.4	0.78	1.5
↑	1.9	1.6	1.4	2.5	1.7
↗	1.4	0.58	1.8	1.6	1.6
↖	2.2	1.3	0.97	1.6	1.5
↙	2.5	2	1.3	0.9	1.5
↓	1.8	2.9	1.7	1.2	2.1
←	2.6	1.5	2.1	1.2	2.2

$a = \text{argmax}_a\, Q(s_t, a)$
+
epsilon-greedy

Bellman Equation

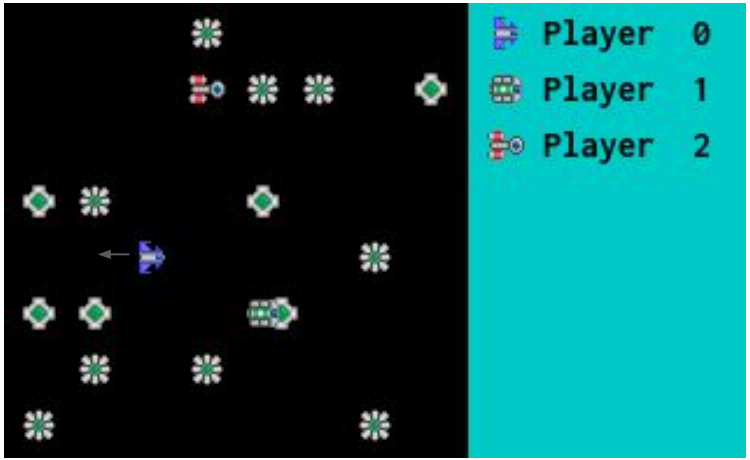


* peut prendre un temps infini 😅

Bellman Equation

$$\underbrace{Q_{n+1}(s_t, a_t)}_{\text{new Q-value}} = r_{t+1} + \gamma \max_a Q_n(s_{t+1}, a)$$

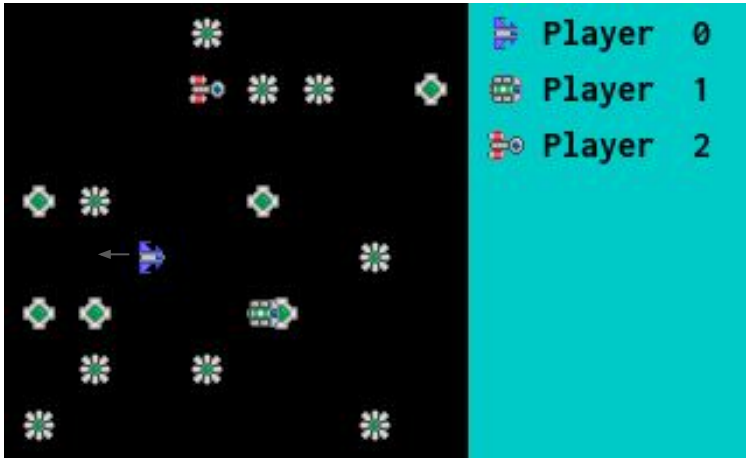
	LEFT	DOWN	RIGHT	UP	STAY
→	1.7	1.4	2.6	0.86	1.5
↘	1.3	1.6	2.4	0.78	1.5
↑	1.9	1.6	1.4	2.5	1.7
↗	1.4	0.58	1.8	1.6	1.6
↖	2.2	1.3	0.97	1.6	1.5
↙	2.5	2	1.3	0.9	1.5
↓	1.8	2.9	1.7	1.2	2.1
←	2.6	1.5	2.1	1.2	2.2



Bellman Equation

$$\underbrace{Q_{n+1}(s_t, a_t)}_{\text{new Q-value}} = r_{t+1} + \gamma \max_a Q_n(s_{t+1}, a)$$

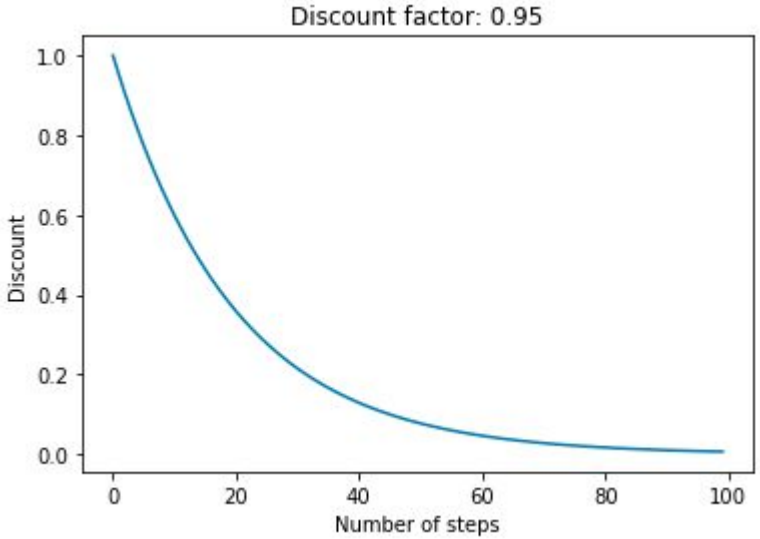
	LEFT	DOWN	RIGHT	UP	STAY
→	1.7	1.4	2.6	0.86	1.5
↘	1.3	1.6	2.4	0.78	1.5
↑	1.9	1.6	1.4	2.5	1.7
↗	1.4	0.58	1.8	1.6	1.6
↖	2.2	1.3	0.97	1.6	1.5
↙	2.5	2	1.3	0.9	1.5
↓	1.8	2.9	1.7	1.2	2.1
←	2.6	1.5	2.1	1.2	2.2



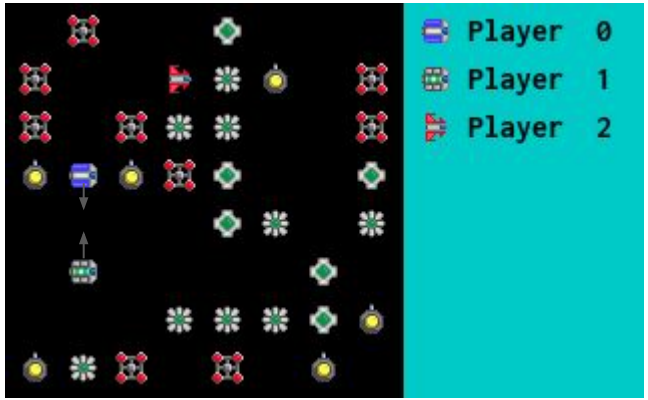
Bellman Equation

$$Q_{n+1}(s_t, a_t) = r_{t+1} + \gamma \max_a Q_n(s_{t+1}, a)$$

Discount factor
(“facteur d’actualisation”)



Learning rate



$$\underbrace{Q_{n+1}(s_t, a_t)}_{\text{new Q-value}} = (1-\alpha)Q_n(s_t, a_t) + \alpha(r_{t+1} + \gamma \max_a Q_n(s_{t+1}, a))$$

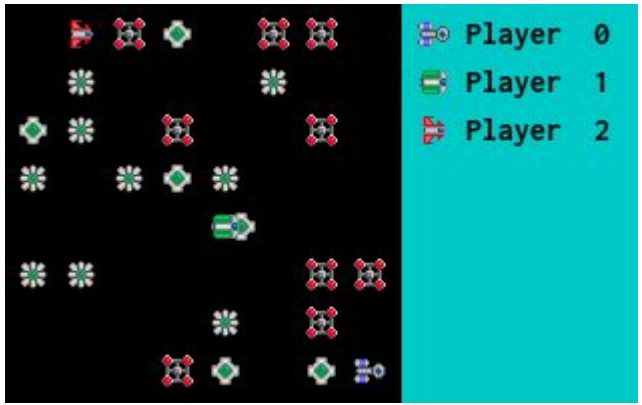
```
alpha = 0   # No update
alpha = 1   # Full update
alpha = 0.5 # Mean between old/new
```

Mise en pratique

Introduction à Q-learning

https://colab.research.google.com/drive/1KSei3ZdjNyyUYsPMqTyli4G2rm9v-P2E

Q-table size




```
{2: 'target: ↓, lidar: →, ↗, ↑, ↖'}
```

↓

```
Dict(target_dir:Discrete(8), lidar:MultiBinary(8))
```

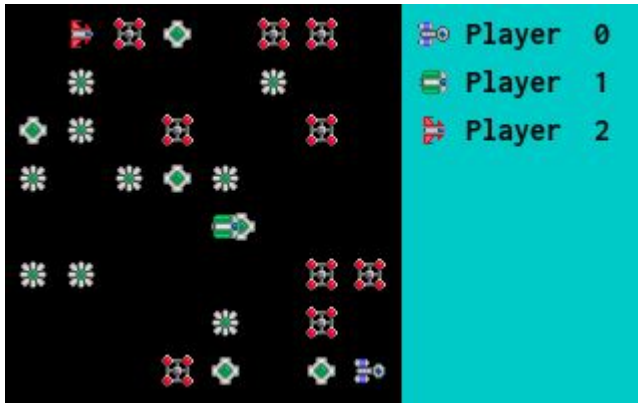
↓

```
{'target_dir': 2, 'lidar', [0, 0, 0, 0, 1, 1, 1, 1]}
```

↓

```
n_states = 8 * (2**8) = 2048
```

Q-table size




```
{2: 'target: ↓, lidar: →, ↗, ↑, ↖'}
```

↓

```
Dict(target_dir:Discrete(8), lidar:MultiBinary(8))
```

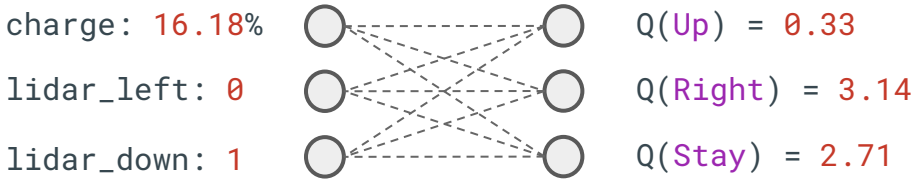
↓

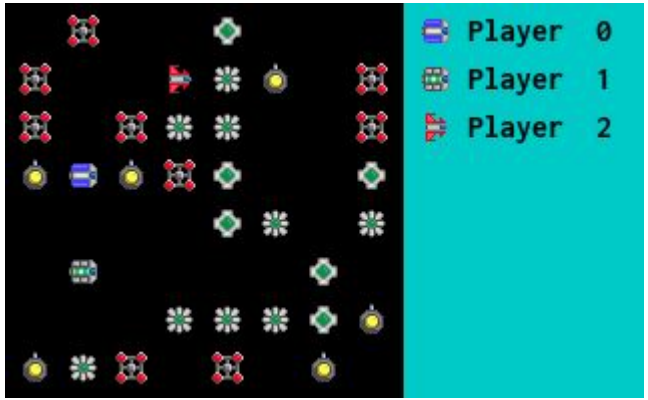
```
{'target_dir': 2, 'lidar', [0, 0, 0, 0, 1, 1, 1, 1]}
```

↓

```
n_states = 8 * (2**8) = 2048
```

Deep Q-network (DQN)



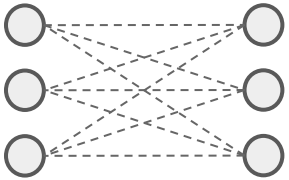


"Q-*network*"

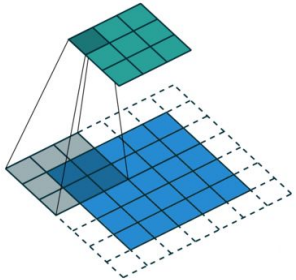
$$\delta = Q_n(s_t, a_t) - \left(r_{t+1} + \gamma \max Q_n(s_{t+1}, a)\right)$$

δ is the loss to minimize

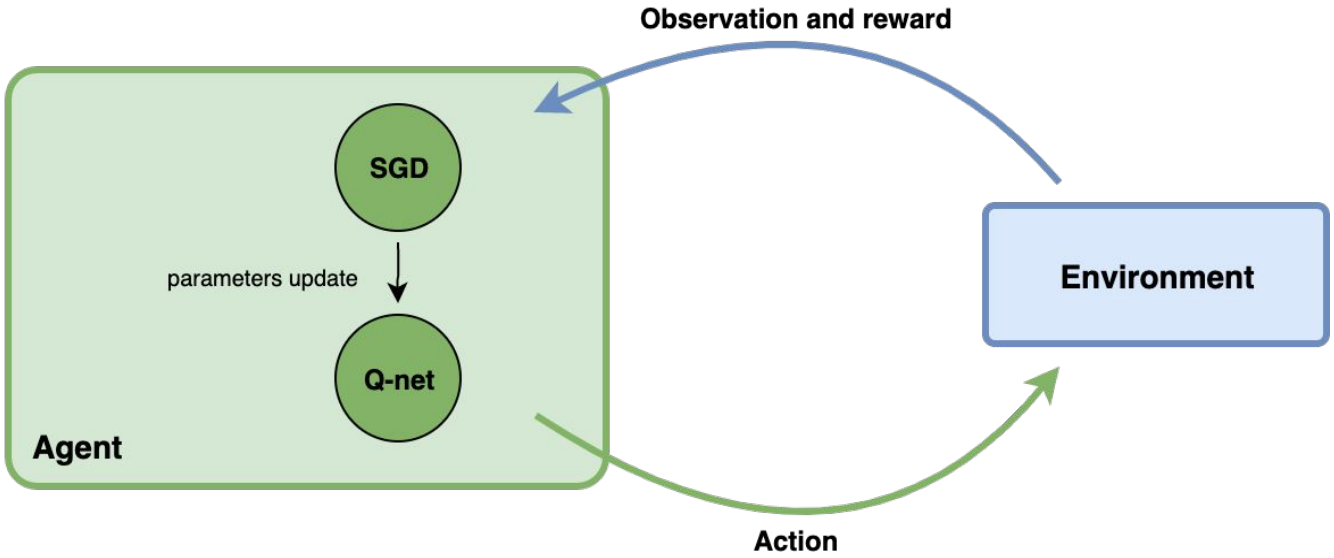
Deep Q-network (DQN)



Dense layer

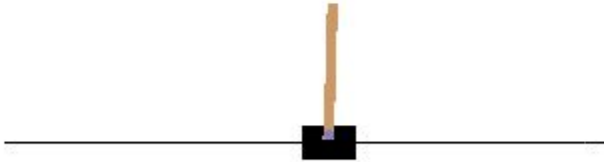


Convolutional layer


Observation and reward
SGD
parameters update
Q-net
Agent
Environment
Action

Timestep: 1, action: left

Reward: 1.0



Timestep: 2, action: right

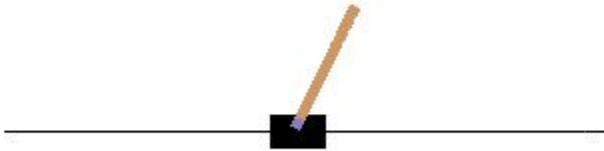
Reward: 1.0

Timestep: 3, action: left

Reward: 1.0

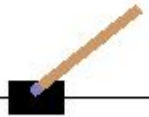
Timestep: 4, action: left

Reward: 1.0



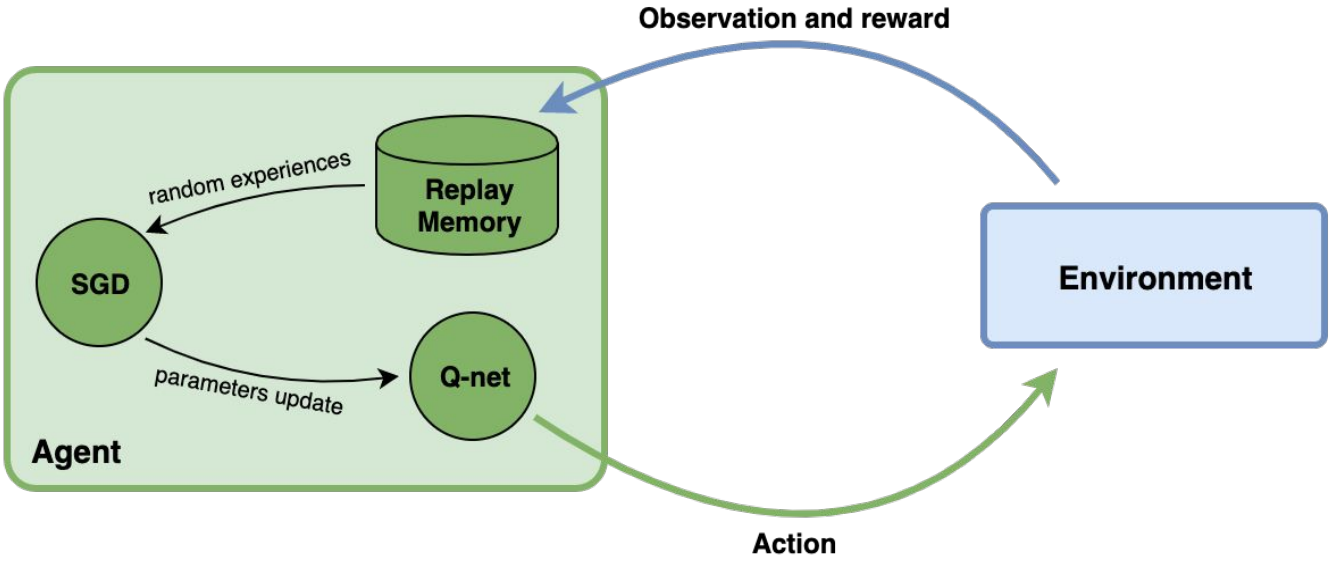
Timestep: 5, action: left

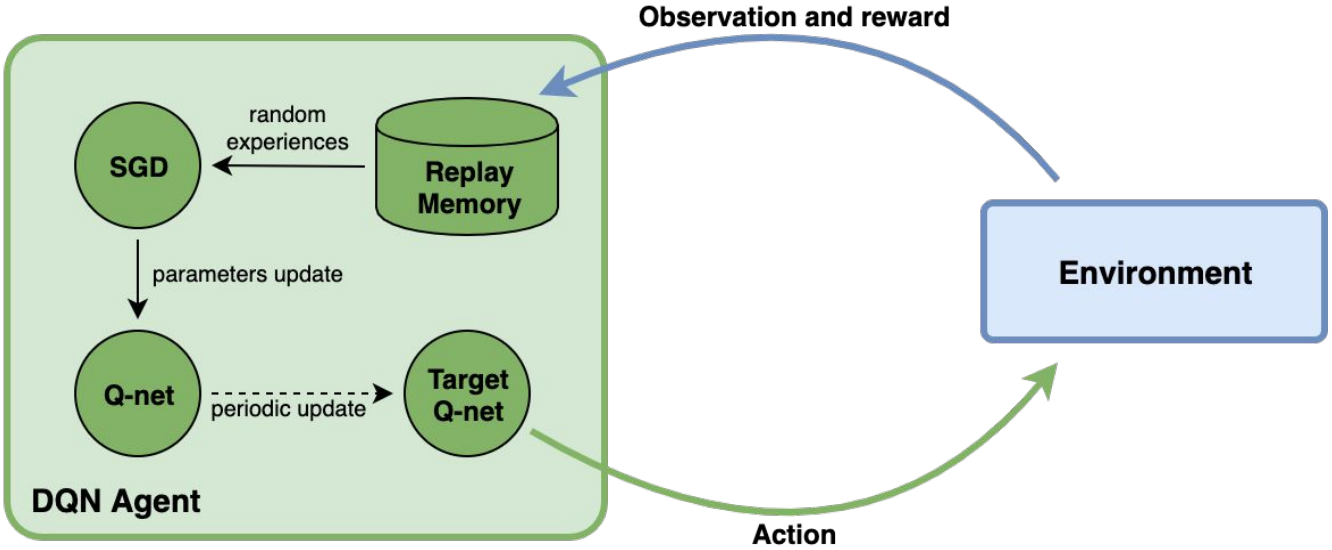
Reward: 1.0



Timestep: 6, action: right

Reward: 0.0


Observation and reward
random experiences
Replay Memory
SGD
parameters update
Q-net
Agent
Environment
Action


Observation and reward
random
experiences
SGD
Replay
Memory
parameters update
Q-net
periodic update
Target
Q-net
DQN Agent
Environment
Action