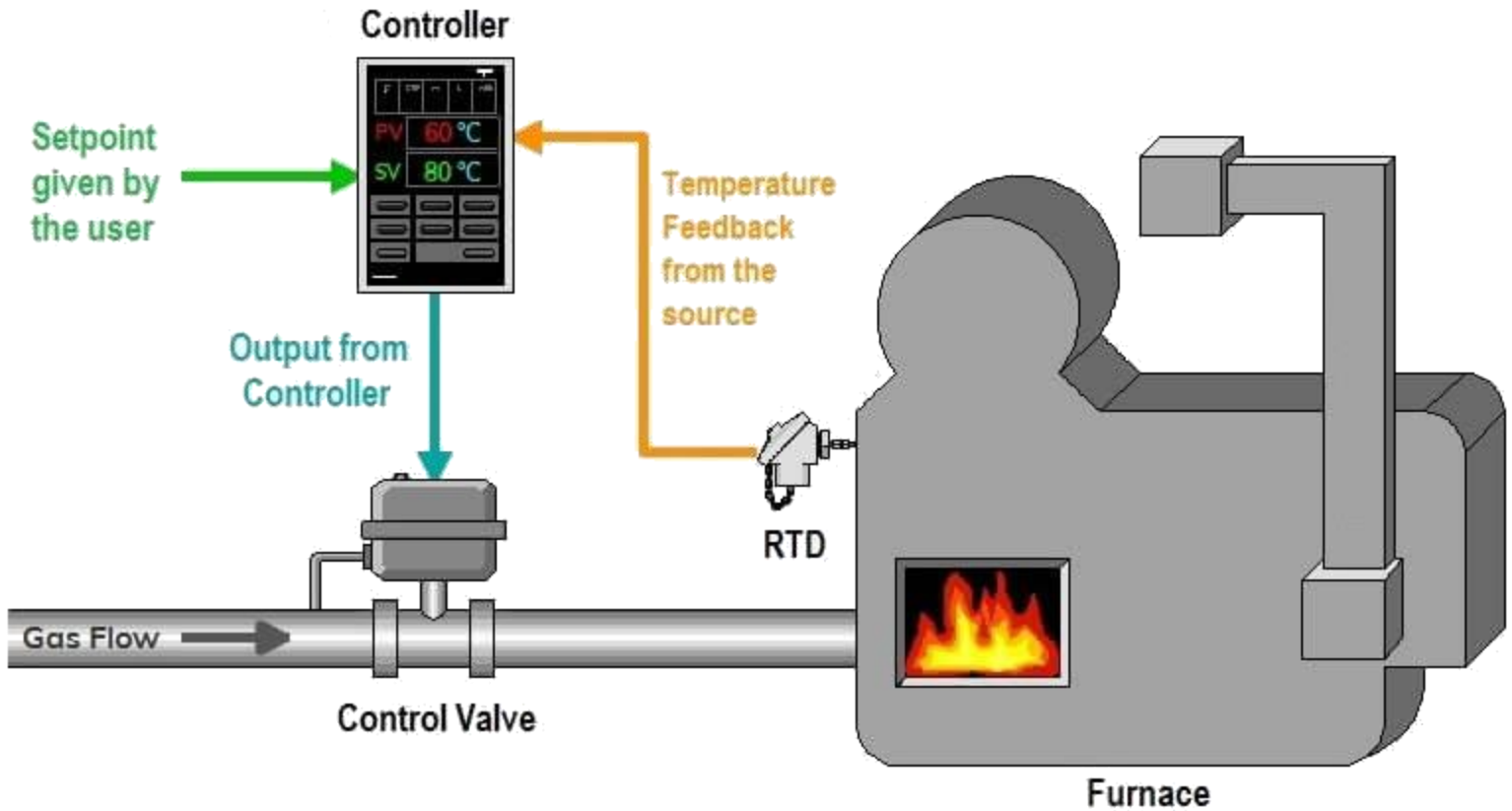
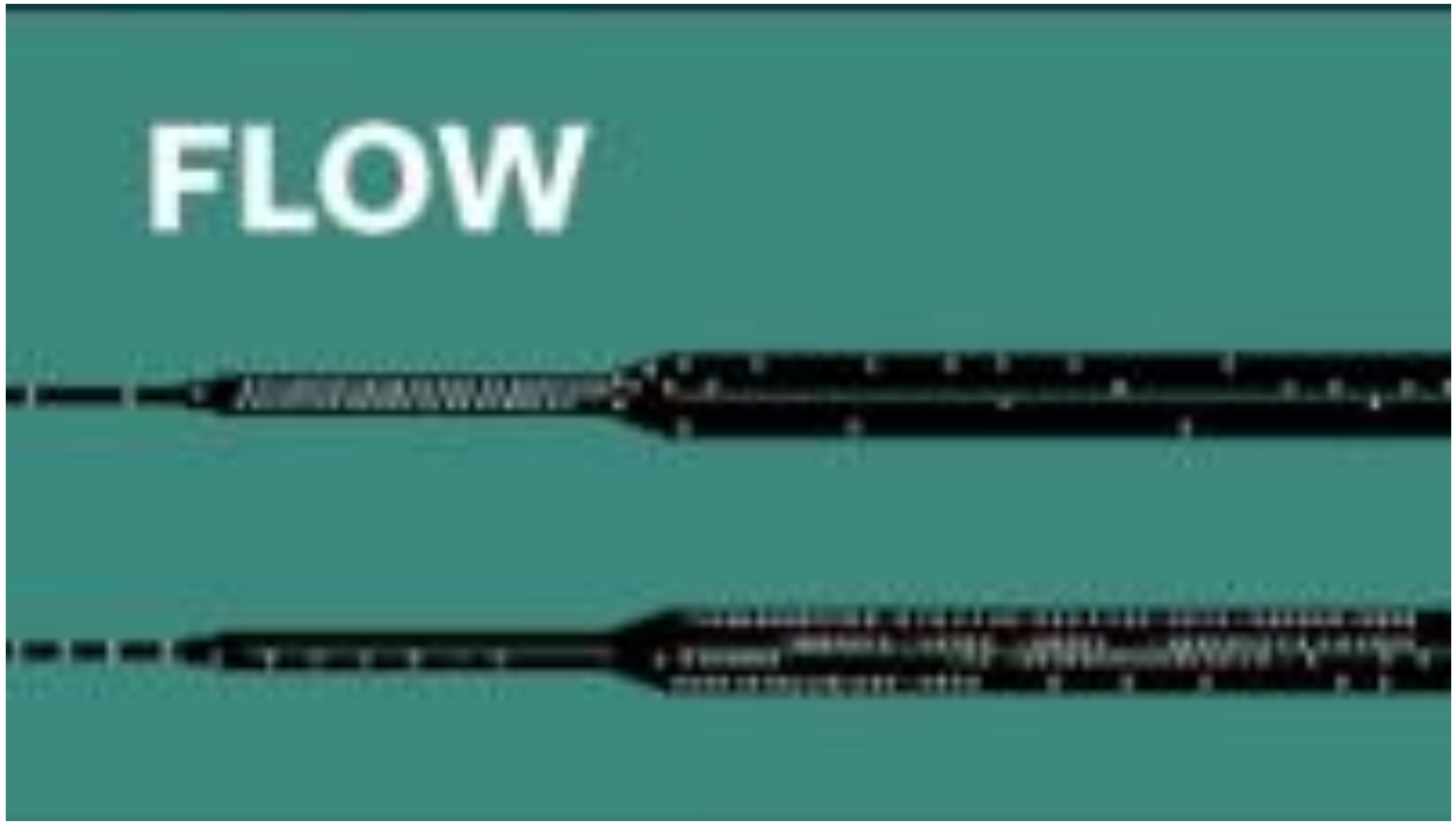
Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ingeniería de Transporte y Logística

Sistemas Urbanos Inteligentes

Control de agentes basado en aprendizaje

Hans Löbel
Dpto. Ingeniería de Transporte y Logística
Dpto. Ciencia de la Computación


Controller
PV 60 °C
SV 80 °C
Setpoint given by the user
Temperature Feedback from the source
Output from Controller
RTD
Gas Flow
Control Valve
Furnace

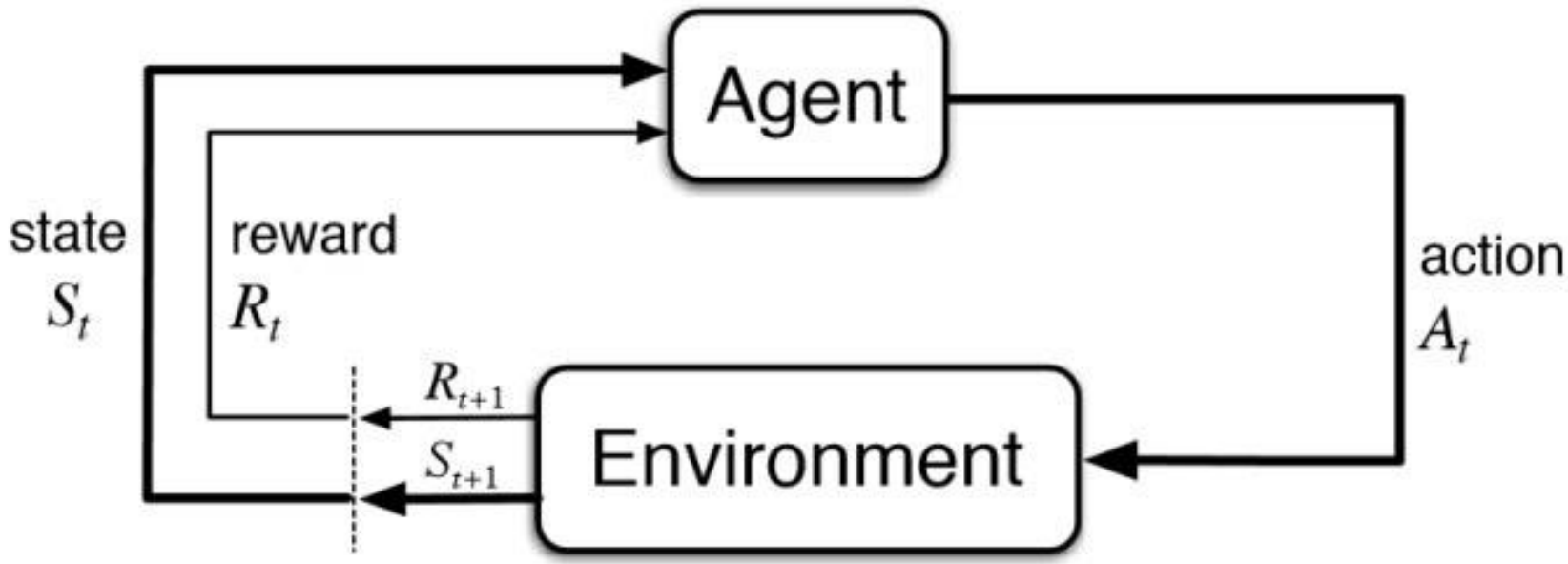


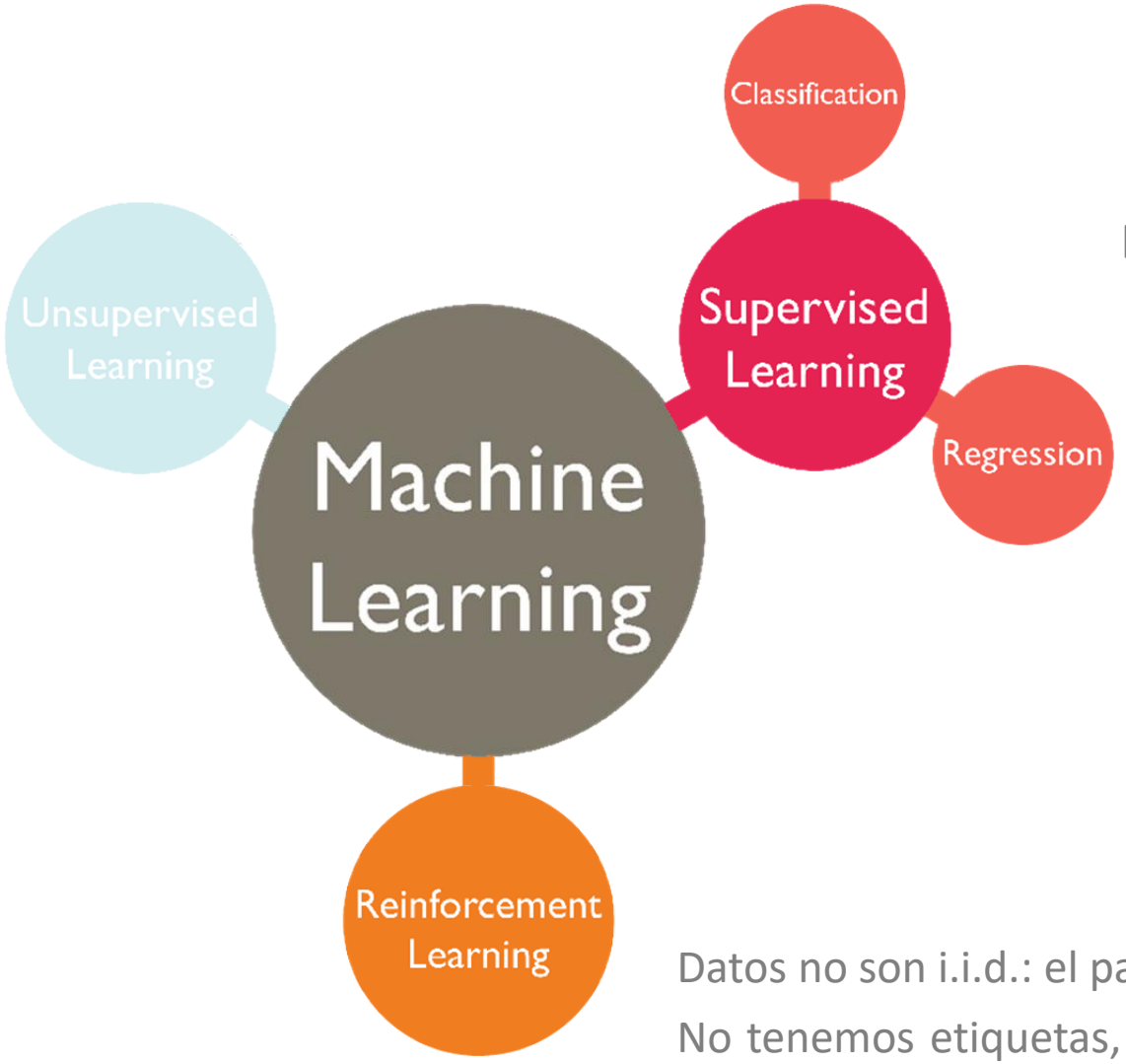
https://www.youtube.com/watch?v=P7xx9uH2i7w

Para esto, utilizaremos aprendizaje reforzado

Aprendizaje reforzado es:

- Formalismo matemático para la toma de decisiones basada en aprendizaje
- Enforque para aprender a tomar decisiones y controlar agentes basado en la experiencia


Agent
state
S_t
reward
R_t
action
A_t
R_{t+1}
S_{t+1}
Environment



Datos etiquetados e i.i.d.

Datos no son i.i.d.: el pasado influencia el futuro

No tenemos etiquetas, solo sabemos si tuvimos éxito o no (o si recibimos una recompensa)

- Acciones: movimientos musculares
- Observaciones (estado): vista, olfato, tacto, oído, gusto
- Recompensa: comida

- Acciones: qué y cuánto comprar
- Observaciones (estado): niveles de inventario
- Recompensa: ganancia

Dificultad y técnicas a usar tienen que ver principalmente con el nivel de estructura del ambiente/entorno

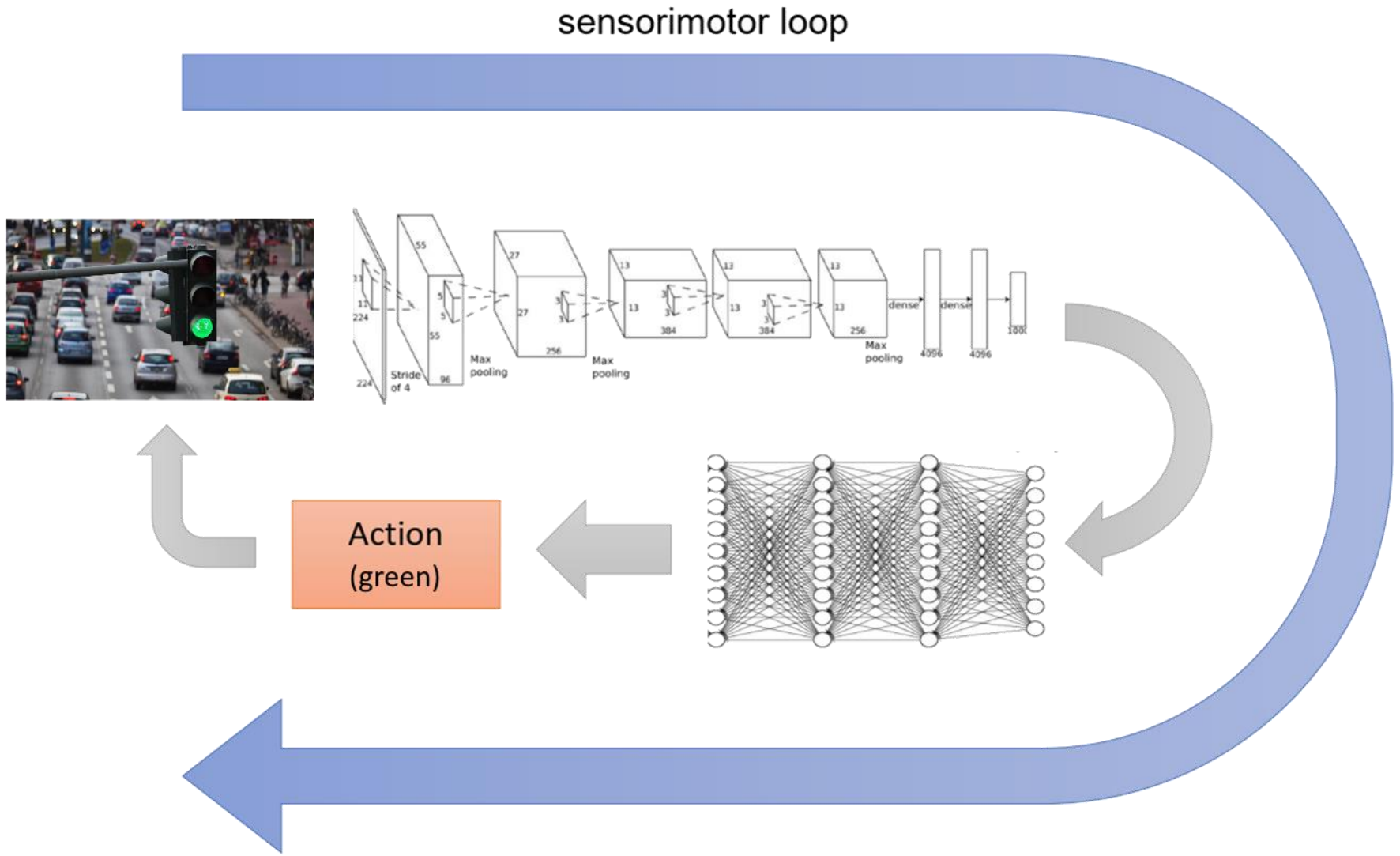
Entorno altamente estructurados: *feature engineering* para caracterizar el estado del mundo. Problema se remite “solo” a aprender a elegir la mejor acción dado el estado.

Reinforcement Learning

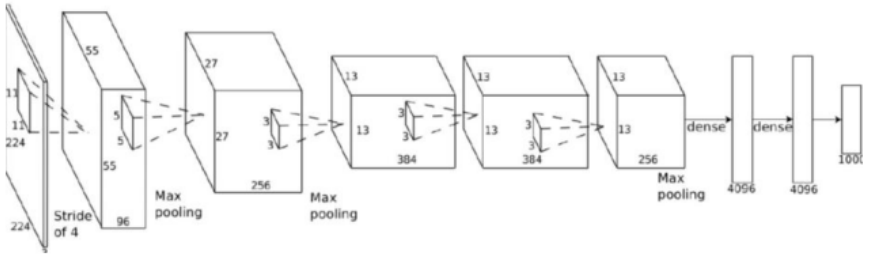
En entornos no estructurados: además de aprender a elegir la mejor acción, es necesario aprender a percibir el mundo.

Deep Reinforcement Learning

En un entorno urbano, generalmente carecemos de estructura



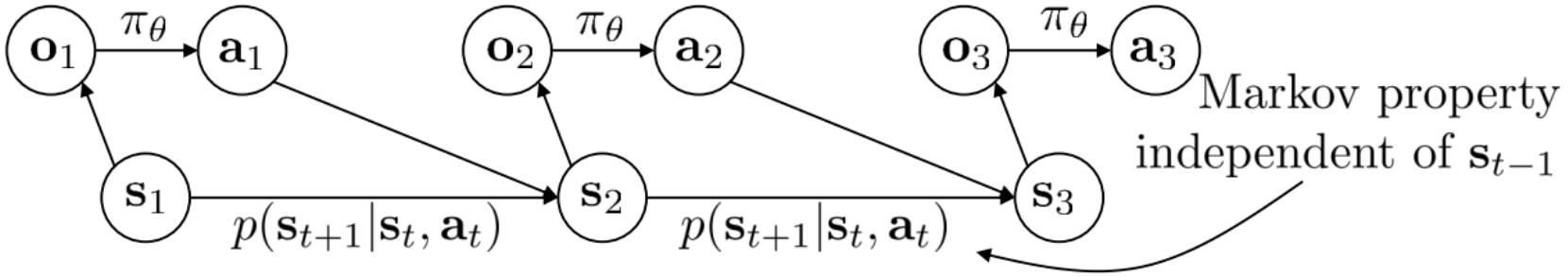
Antes de empezar con las técnicas, un poco de notación…



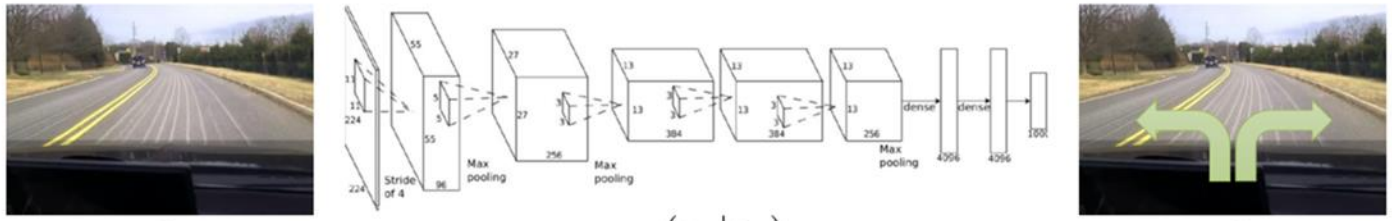
$\mathbf{o}_t$ $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ $\mathbf{a}_t$

$\mathbf{s}_t$ – state
$\mathbf{o}_t$ – observation
$\mathbf{a}_t$ – action

$\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ – policy
$\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)$ – policy (fully observed)



La recompensa actúa como una especia de supervisión



$\mathbf{o}_t$ $\pi_\theta(\mathbf{a}_t|\mathbf{o}_t)$ $\mathbf{a}_t$

which action is better or worse?

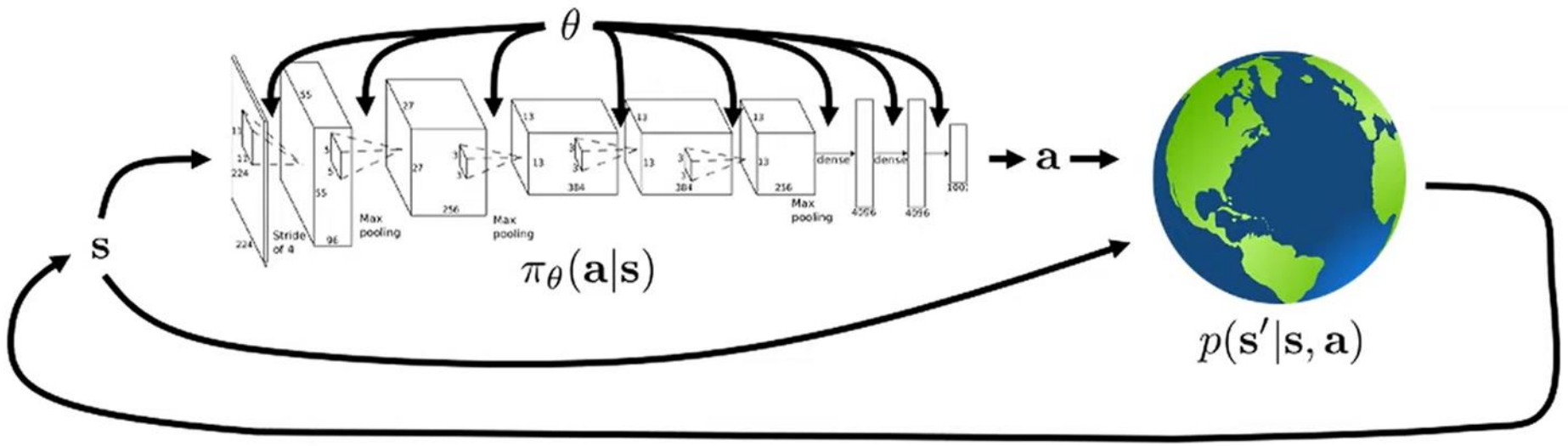
$r(\mathbf{s},\mathbf{a},\mathbf{s}')$: reward function ⟶ tells us which states and actions are better

high reward 

 low reward

$s, a, r(s, a, s')$ y $p(s'|s, a)$definen un proceso de decisión markoviano (MDP)

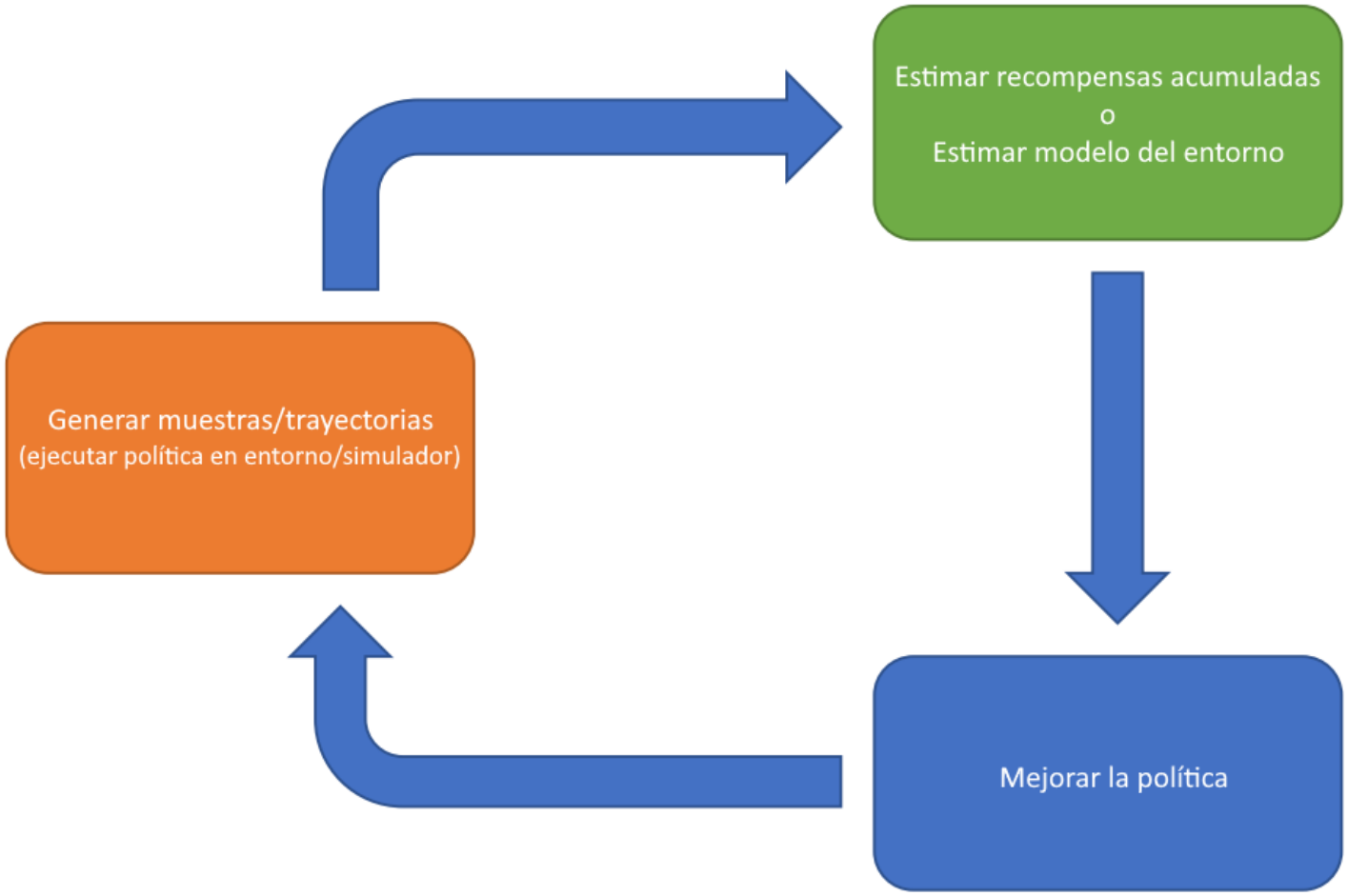
En (D)RL, buscamos la política que maximiza la recompensa esperada



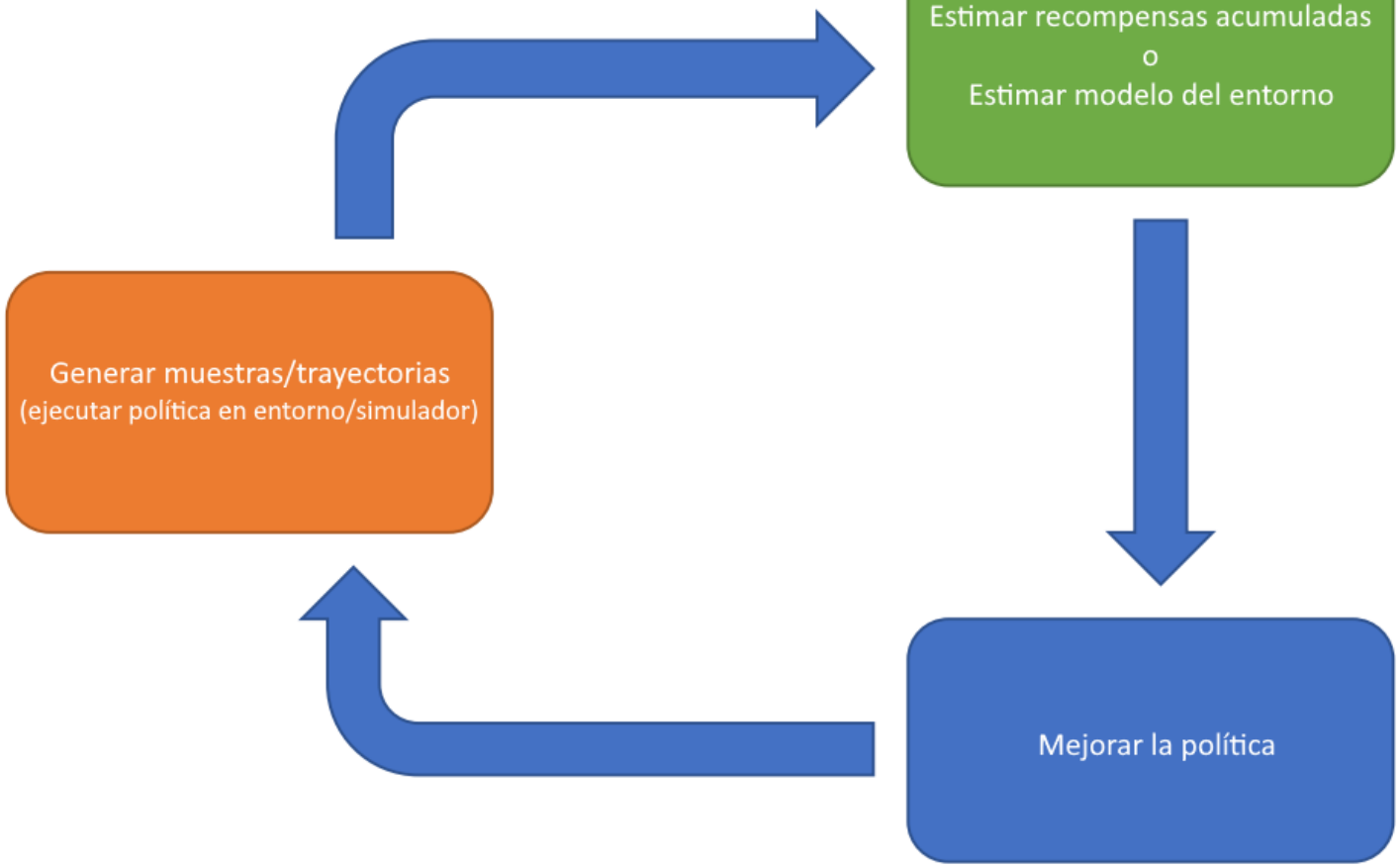
$$\underbrace{p_\theta(\mathbf{s}_1, \mathbf{a}_1, \ldots, \mathbf{s}_T, \mathbf{a}_T)}_{p_\theta(\tau)} = p(\mathbf{s}_1) \prod_{t=1}^{T} \pi_\theta(\mathbf{a}_t|\mathbf{s}_t) p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$$

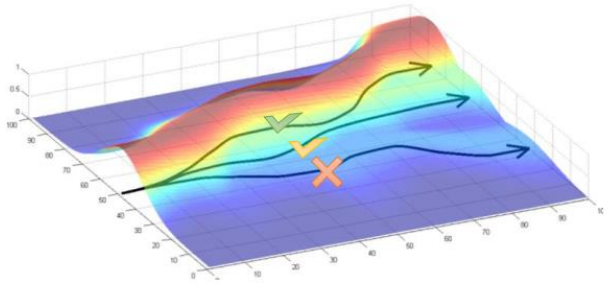
$$\theta^\star = \arg\max_\theta E_{\tau \sim p_\theta(\tau)} \left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}) \right]$$

Todos los algoritmos siguen la misma estructura básica



Por ejemplo, si queremos optimizar directamente la política…



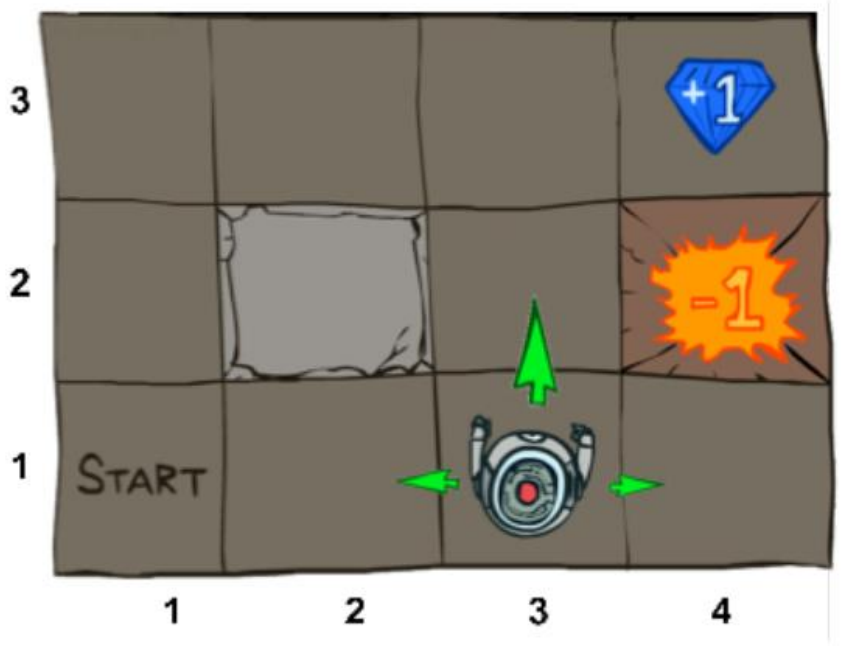


$$J(\theta) = E_\pi \left[\sum_t r_t \right] \approx \frac{1}{N} \sum_{i=1}^{N} \sum_t r_t^i$$

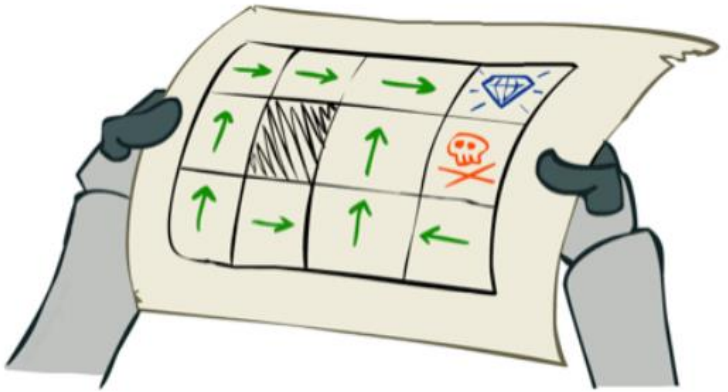
$$\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$$

Este último esquema no es el único que se puede tomar

Este último esquema no es el único que se puede tomar



π:

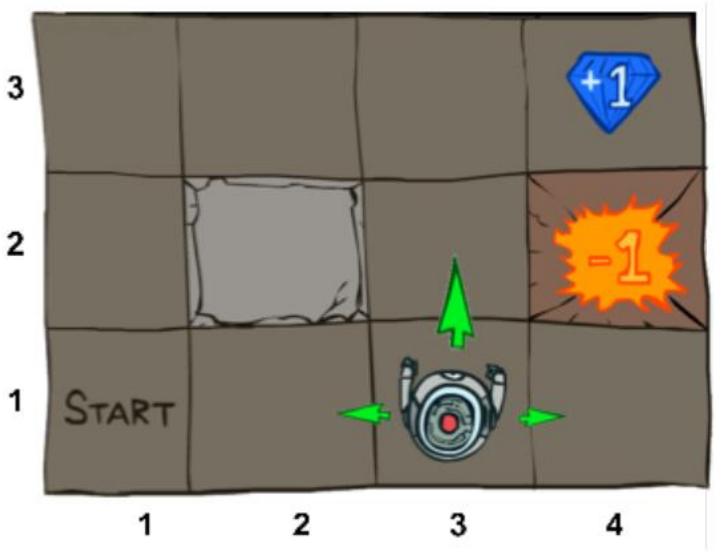


$$max_{\pi} \mathrm{E}[\sum_{t=0}^{H} \gamma^t R(S_t, A_t, S_{t+1}) | \pi]$$

Esta idea se puede formalizar a través de la función de valor

$$V^*(s) = \max_{\pi} \mathbb{E}\left[\sum_{t=0}^{H} \gamma^t R(s_t, a_t, s_{t+1}) \mid \pi, s_0 = s\right]$$

$V^*(s)$= suma de las recompensas con descuento al empezar en el estado s y actuar óptimamente



Supongamos acciones siempre exitosas, $\gamma = 1, H = 100$

$V^*(4,3) =$

$V^*(3,3) =$

$V^*(2,3) =$

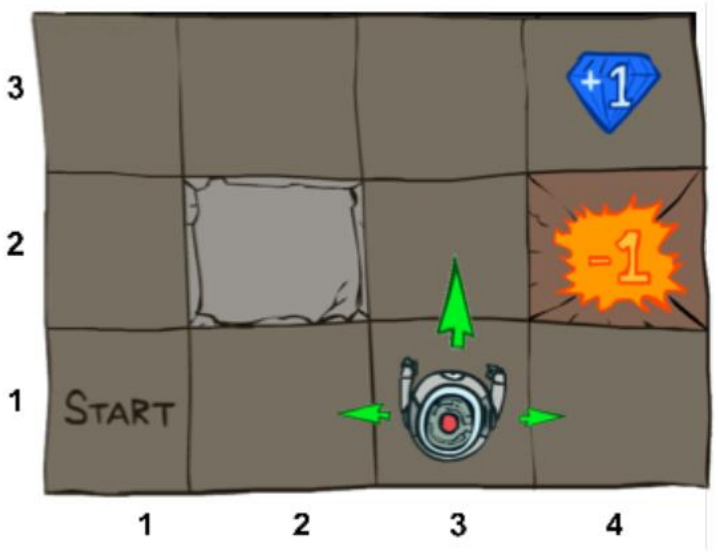
$V^*(1,1) =$

$V^*(4,2) =$

Esta idea se puede formalizar a través de la función de valor

$$V^*(s) = \max_{\pi} \mathbb{E}\left[\sum_{t=0}^{H} \gamma^t R(s_t, a_t, s_{t+1}) \mid \pi, s_0 = s\right]$$

$V^*(s)$= suma de las recompensas con descuento al empezar en el estado s y actuar óptimamente



Supongamos acciones siempre exitosas, $\gamma = 0{,}9$, $H = 100$

$V^*(4,3) =$

$V^*(3,3) =$

$V^*(2,3) =$

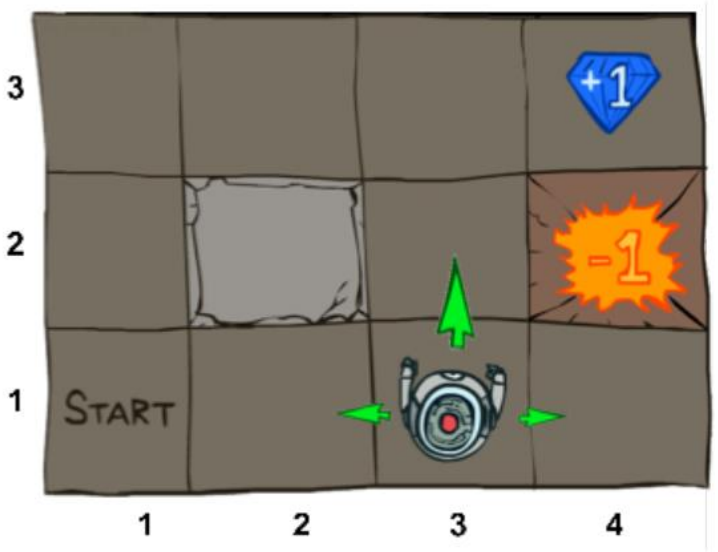
$V^*(1,1) =$

$V^*(4,2) =$

Esta idea se puede formalizar a través de la función de valor

$$V^*(s) = \max_{\pi} \mathbb{E}\left[\sum_{t=0}^{H} \gamma^t R(s_t, a_t, s_{t+1}) \mid \pi, s_0 = s\right]$$

$V^*(s)$= suma de las recompensas con descuento al empezar en el estado s y actuar óptimamente



Supongamos acciones con $P = 0{,}8$, $\gamma = 0{,}9$, $H = 100$

$V^*(4,3) =$

$V^*(3,3) =$

$V^*(2,3) =$

$V^*(1,1) =$

$V^*(4,2) =$

¿Cómo podemos estimar esta función de valor?

$V_0^*(s)$ = optimal value for state s when H=0

¿Cómo podemos estimar esta función de valor?

$V_0^*(s)$ = optimal value for state s when H=0

- $V_0^*(s) = 0 \quad \forall s$

¿Cómo podemos estimar esta función de valor?

$V_0^*(s)$ = optimal value for state s when H=0

- $V_0^*(s) = 0 \quad \forall s$

$V_1^*(s)$ = optimal value for state s when H=1

¿Cómo podemos estimar esta función de valor?

$V_0^*(s)$ = optimal value for state s when H=0

- $V_0^*(s) = 0 \quad \forall s$

$V_1^*(s)$ = optimal value for state s when H=1

- $V_1^*(s) = \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_0^*(s'))$

¿Cómo podemos estimar esta función de valor?

$V_0^*(s)$ = optimal value for state s when H=0

- $V_0^*(s) = 0 \quad \forall s$

$V_1^*(s)$ = optimal value for state s when H=1

- $V_1^*(s) = \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_0^*(s'))$

$V_2^*(s)$ = optimal value for state s when H=2

- $V_2^*(s) = \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_1^*(s'))$

¿Cómo podemos estimar esta función de valor?

$V_0^*(s)$ = optimal value for state s when H=0

- $V_0^*(s) = 0 \quad \forall s$

$V_1^*(s)$ = optimal value for state s when H=1

- $V_1^*(s) = \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_0^*(s'))$

$V_2^*(s)$ = optimal value for state s when H=2

- $V_2^*(s) = \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_1^*(s'))$

$V_k^*(s)$ = optimal value for state s when H = k

- $V_k^*(s) = \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_{k-1}^*(s'))$

Este simple algoritmo es conocido como *Value Iteration*

Start with $V_0^*(s) = 0$ for all s.

For k = 1, … , H:

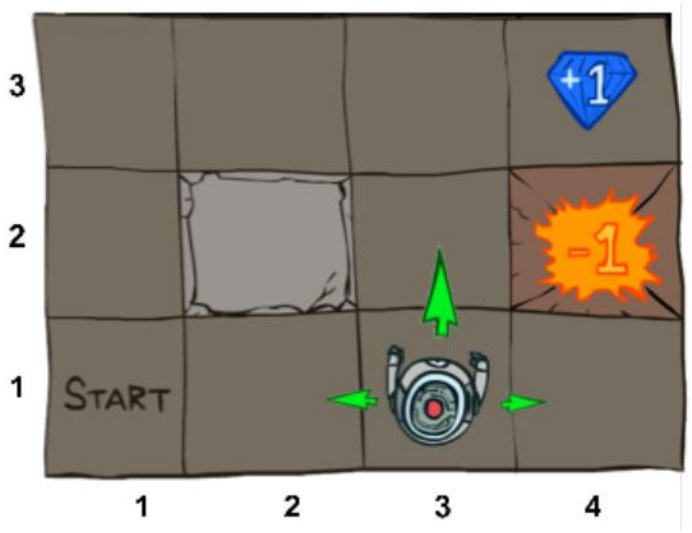
For all states s in S:

$$V_k^*(s) \leftarrow \max_a \sum_{s'} P(s'|s,a) \left(R(s,a,s') + \gamma V_{k-1}^*(s')\right)$$

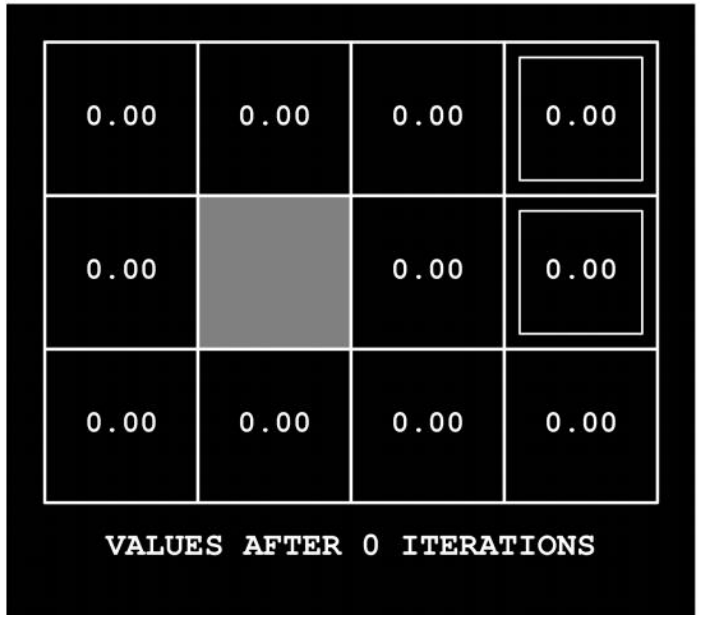
$$\pi_k^*(s) \leftarrow \arg\max_a \sum_{s'} P(s'|s,a) \left(R(s,a,s') + \gamma V_{k-1}^*(s')\right)$$

Veamos gráficamente un ejemplo de su ejecución

$$V_0(s) \leftarrow 0$$



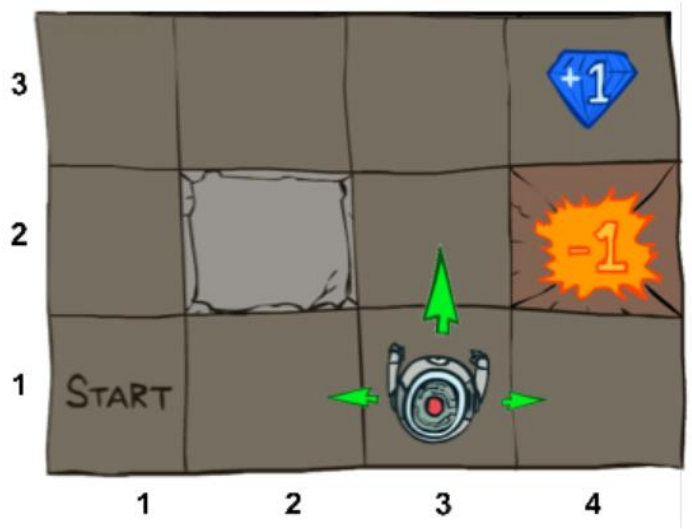
k = 0



Noise = 0.2
Discount = 0.9

Veamos gráficamente un ejemplo de su ejecución

$$V_1(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_0(s'))$$

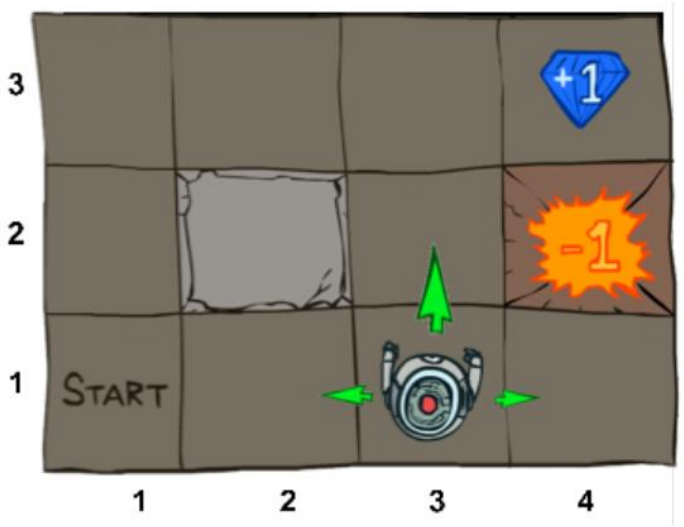


k = 0

Noise = 0.2
Discount = 0.9

Veamos gráficamente un ejemplo de su ejecución

$$V_2(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_1(s'))$$



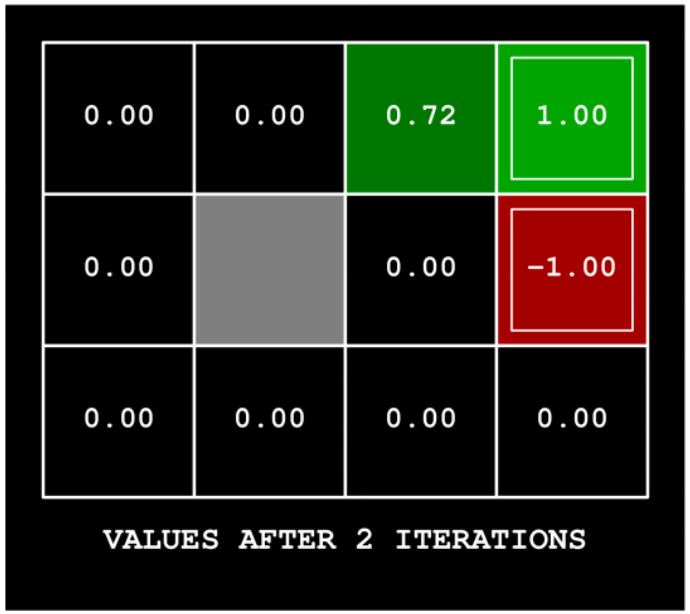
k = 1

Noise = 0.2
Discount = 0.9

Veamos gráficamente un ejemplo de su ejecución

$$V_2(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_1(s'))$$

k = 2



Noise = 0.2
Discount = 0.9

Veamos gráficamente un ejemplo de su ejecución

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_k(s'))$$

k = 3

Noise = 0.2
Discount = 0.9

Veamos gráficamente un ejemplo de su ejecución

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_k(s'))$$

k = 4

Noise = 0.2
Discount = 0.9

Veamos gráficamente un ejemplo de su ejecución

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_k(s'))$$

k = 5

Noise = 0.2
Discount = 0.9

Veamos gráficamente un ejemplo de su ejecución

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_k(s'))$$

k = 12

Noise = 0.2
Discount = 0.9

Veamos gráficamente un ejemplo de su ejecución

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_k(s'))$$

k = 100

Noise = 0.2
Discount = 0.9

Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ingeniería de Transporte y Logística

Sistemas Urbanos Inteligentes

Control de agentes basado en aprendizaje

Hans Löbel
Dpto. Ingeniería de Transporte y Logística
Dpto. Ciencia de la Computación