Agentes Inteligentes

Javier García

Departamento de Electrónica y Computación
Universidad de Santiago de Compostela

September 8, 2021

Part I

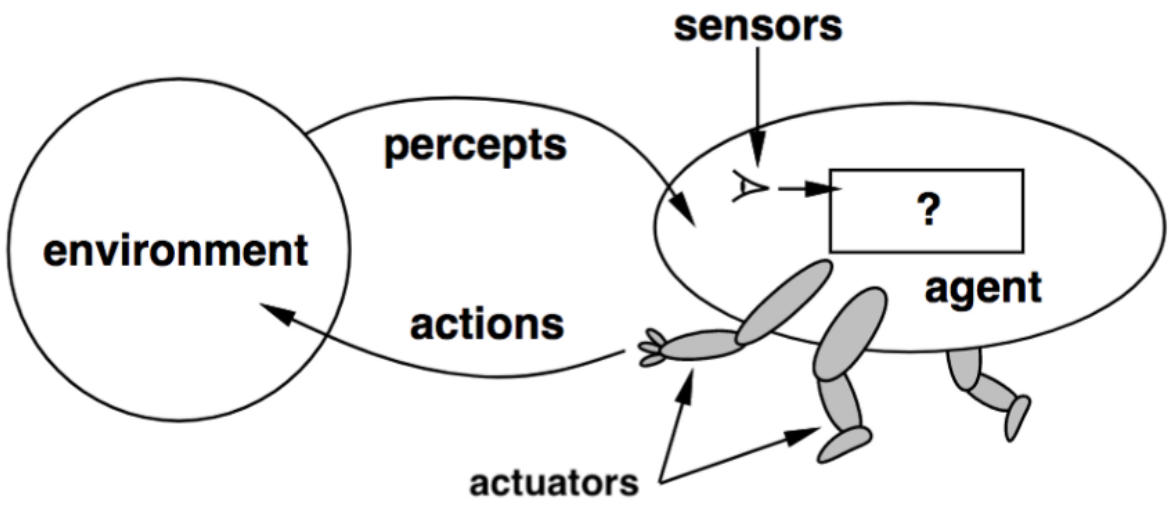
Agentes Inteligentes

Contenidos

Definición (I)

"Un agente es cualquier cosa que percibe su entorno a través de sus sensores y actúa sobre ese entorno a través de sus actuadores"

Russell and Norvig



- Sensores: Para percibir el entorno
- Actuadores: Para modificar el entorno

El comportamiento del agente es descrito mediante una función f que mapea secuencias de percepciones a acciones:

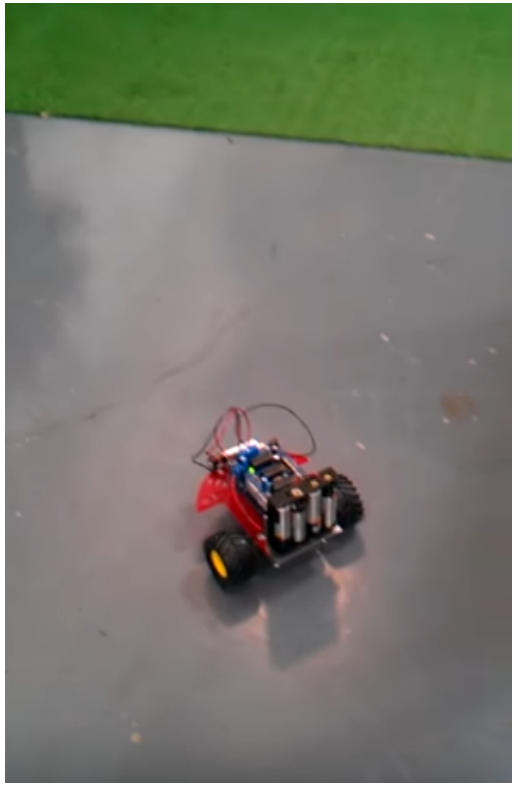
$$f : P* \rightarrow A$$

Definición (II)

- ¿Un humano es un agente?
  - Sensores: Ojos, orejas, ...
  - Actuadores: Manos, piernas, boca, ...
- ¿Un termostato es un agente?
  - Sensores: sensor de temperatura
  - Actuador: abre/cierra un circuito eléctrico en función de la temperatura
- **¿Un robot es un agente?**
  - Sensores: Camaras, lasers, ...
  - Actuadores: motores
- **Que sean agentes no quiere decir que se comporten de forma inteligente**

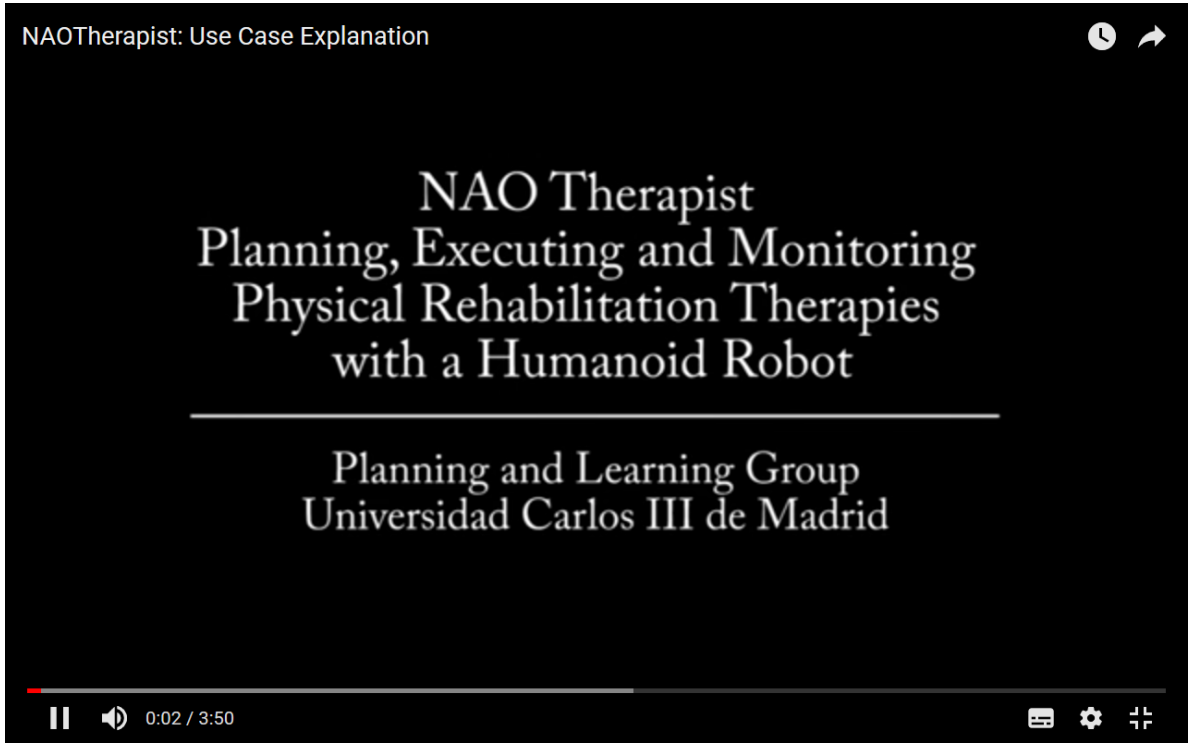
Definición (III)

Arduino

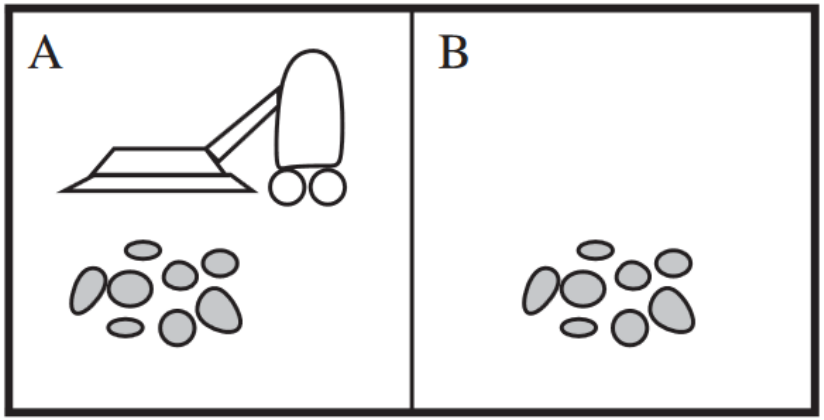


Definición (IV)

Inrobics



Definición (V)



Secuencia percepciones	Acción
[*A*, *Clean*]	Right
[*A*, *Dirty*]	Suck
[*B*, *Clean*]	Left
[*B*, *Dirty*]	Suck
[*A*, *Clean*], [*A*, *Clean*]	Right
[*A*, *Clean*], [*A*, *Dirty*]	Suck
...	...
[*A*, *Clean*], [*A*, *Clean*], [*A*, *Clean*]	Right
[*A*, *Clean*], [*A*, *Clean*], [*A*, *Dirty*]	Suck
...	...

Definición (VI)

¿Cómo se rellena la tabla anterior para crear un agente **inteligente**?

> *"Para cada posible secuencia de percepciones, un agente racional debe seleccionar la acción que se espera que maximice una medida de rendimiento, dada la evidencia proporcionada por la secuencia de percepciones y el conocimiento que tenga el agente"*
>
> Russell and Norvig

- Medida de rendimiento: Criterio objetivo para medir el éxito del agente
- *...que se espera que maximize...* Racionalidad no es lo mismo que perfección, es elegir la acción correcta con la información disponible
- Aprendizaje
- Autonomía

Definición (VII)

No existe una definición comúnmente aceptada...

"Cualquier proceso computacional localizado en un entorno capaz de ejecutar acciones de forma autónoma para alcanzar sus objetivos"

Wooldridge and Jennings

"Los agentes inteligentes son entidades software que ejecutan acciones en nombre de un usuario u otro programa con cierto grado de autonomía y, para ello, emplea conocimiento de las metas de los usuarios"

IBM

"Los agentes inteligentes continuamente ejecutan tres acciones: percepción para observar el entorno, actuación para modificar el entorno, razonamiento para interpretar las percepciones, solucionar problemas, y determinar las acciones"

Hayes and Roth

Propiedades de los entornos

- El agente debe operar en un entorno:
  - Las acciones se ejecutan sobre el entorno
  - El entorno proporciona las percepciones
- Los entornos tienen muchas propiedades que afectan al diseño de los agentes
- Russel and Norvig:
  - Observable vs. Parcialmente Observable
  - Determinista vs. No determinista
  - Episódico vs. No episódico
  - Estátivo vs. Dinámico
  - Discreto vs. Continuo

Observable vs. Parcialmente Observable

- Un entorno es observable si el agente puede obtener información:
  - Completa
  - Correcta
  - Actualizada
- El entorno es observable si los sensores del agente proporcionan acceso al estado completo en cada instante de tiempo
- El estado completo se considera toda la información relevante para tomar una decisión
- Cuanto más observable sea el entorno, más sencillo será construir agentes que operen en él
- La mayor parte de los problemas del mundo real son parcialmente observables

Determinista vs. No determinista

- Un entorno es determinista si cada acción tiene un único efecto en él, y no hay incertidumbre sobre el nuevo estado al que se llega
- Dicho de otra manera: una acción ejecutada en un estado siempre conduce al mismo estado
- Los entornos no deterministas o estocásticos son los más problemáticos
- Los problemas del mundo real son generalmente no deterministas

Episódico vs. No episódico

- Un ambiente es episódico si el comportamiento del agente puede ser dividido en secuencias de percepción-acción no relacionadas entre ellas
- Cada episodio consiste en una única percepción-acción y la toma de decisión de qué acción ejecutar no depende de acciones pasadas ni afecta a acciones futuras
- Los entornos episódicos son más fáciles porque el agente puede decidir qué acción realizar sólo sobre la base del episodio actual
- Tareas como jugar al ajedrez o conducir son no episódicas o secuenciales, puesto que las acciones inmediatas tienen consecuencias a largo plazo

Estátivo vs. Dinámico

- Un entorno es estático si podemos suponer que permanece sin cambios (excepto por las acciones que realizan los propios agentes)
- En cambio, si el entorno cambia mientras el agente está deliberando, entonces es dinámico
- Los entornos dinámicos son más difíciles porque otras entidades/elementos pueden interferir con las acciones del agente
- El ajedrez es estático, el golf es dinámico

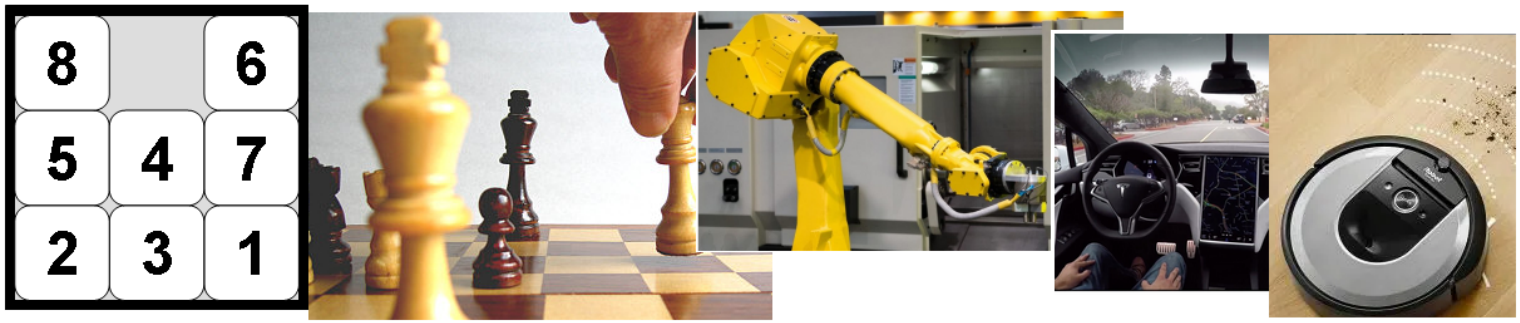
Discreto vs. Continuo

- Un entorno es discreto si hay un número fijo y finito de acciones y percepciones en él
- Por lo tanto, el entorno puede describirse mediante un número limitado de percepciones y acciones claramente definidas
- Por supuesto, los entornos discretos son más fáciles de resolver
- El ajedrez es discreto, conducir es continuo

Propiedades de los entornos

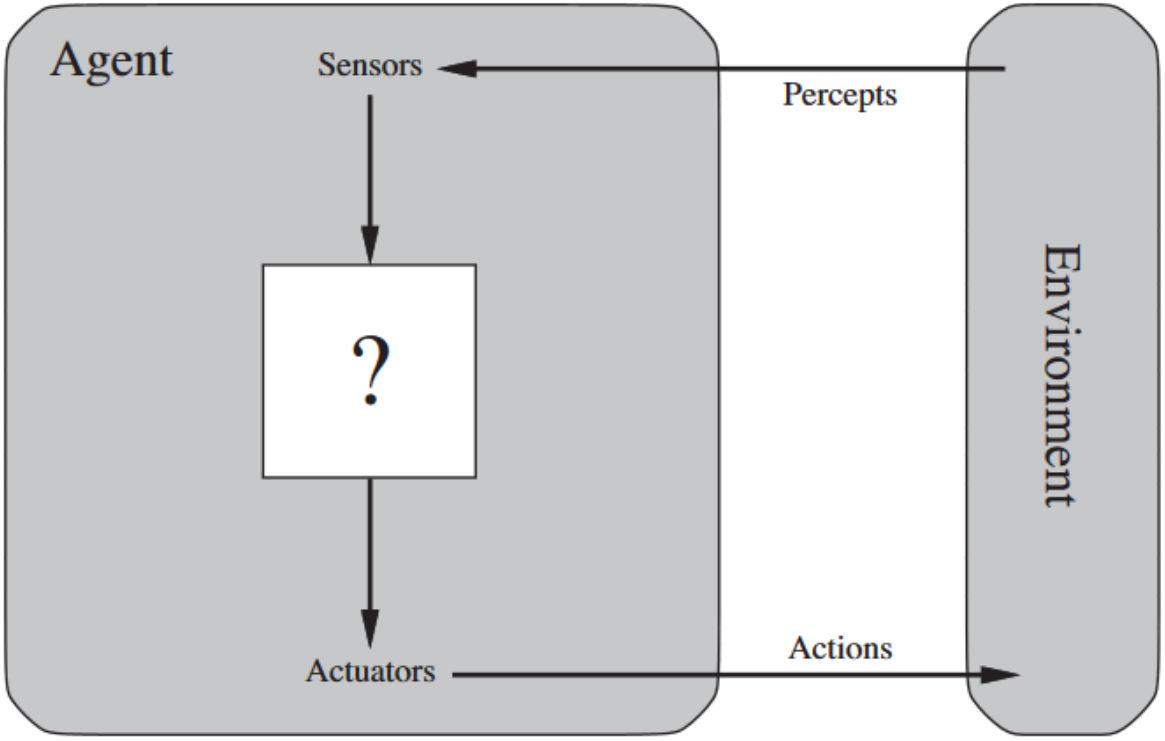
- Clasificar el entorno puede ayudar a guiar el proceso de diseño del agente

	8-puzzle	Ajedrez	Piezas def.	Coche	Roomba
¿Observable?	✓	✓	X	X	X
¿Determinista?	✓	✓	X	X	X
¿Episódico?	X	X	✓	X	X
¿Estático?	✓	✓	X	X	X
¿Discreto?	✓	✓	X	X	X



Estructura de un agente (I)

Los agentes interactuan con los entornos a través de los sensores y actuadores, y...



Estructura de un agente (II)

Agente = Programa + Arquitectura

Programa: Implementa la función f
Arquitectura: Dispositivo con sus actuadores y sensores donde se ejecuta el programa

En robótica...

''Una arquitectura proporciona un método para organizar sistemas de control. No obstante, además de proporcionar una estructura, impone restricciones en la forma en que se soluciona el problema"

Mataric, 1992

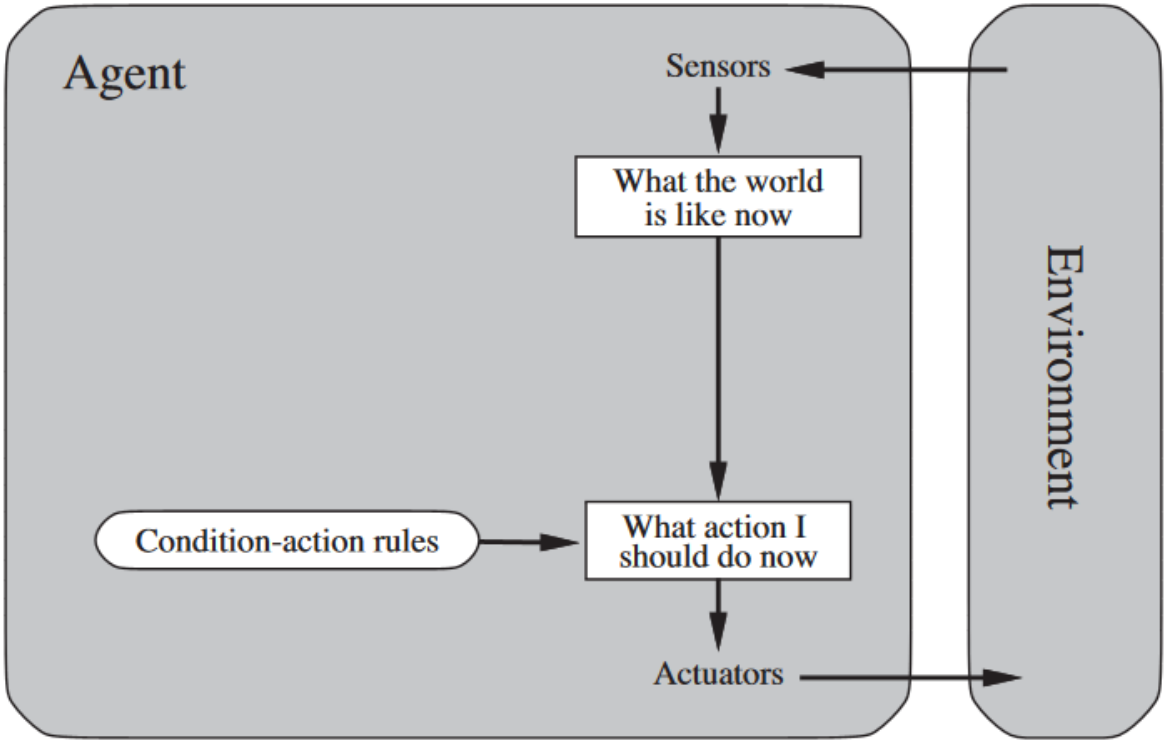
''Una arquitectura robótica se refiere más al software que al hardware"

Arkin, 1998

Tipos de agentes

- ¿El agente tabular anterior es una buena forma de construir un agente?
- El desafío es descubrir cómo escribir agentes que produzcan un comportamiento racional a partir de un programa más pequeño que una gran tabla
- Muchas clasificaciones de agentes:
  - Russell and Norvig:
    - Simple Reflex Agent
    - Model-based Reflex Agent
    - Goal-based Agent
    - Utility-based Agent
    - Learning Agent
  - La comúnmente aceptada (y más cercana a robótica):
    - Reactivos
    - Deliberativos
    - Híbridos

Simple Reflex Agent (I)



function simple-reflex-agent:
    Input: rules condition-action, percept
    Output: action
    state ← *INTERPRET_INPUT*(*percept*)
    rule ← *RULE_MATCH*(*state*, *rules*)
    action ← *rule.action*
return action

Simple Reflex Agent (II)

- **if** car-in-front-braking **then** initiate-braking
- **function** reflex-vacuun-cleaner([location, status]):
  if status = Dirty **then** return Suck
  else if location = A **then** return Right
  else if location = B **then** return Left
- La acción se toma a partir de una única percepción, la percepción actual
- ¿Qué ocurre si una sola percepción no es suficiente para tomar una decisión racional?
  - Diferentes posiciones de luces de freno: una sola imagen insuficiente para saber si un coche está frenando
- ¿Qué ocurre si la percepción no es completa?
  - Una imagen no puede captar todos los coches a la vez, pero puede ser necesario conocer esta información para tomar la decisión correcta

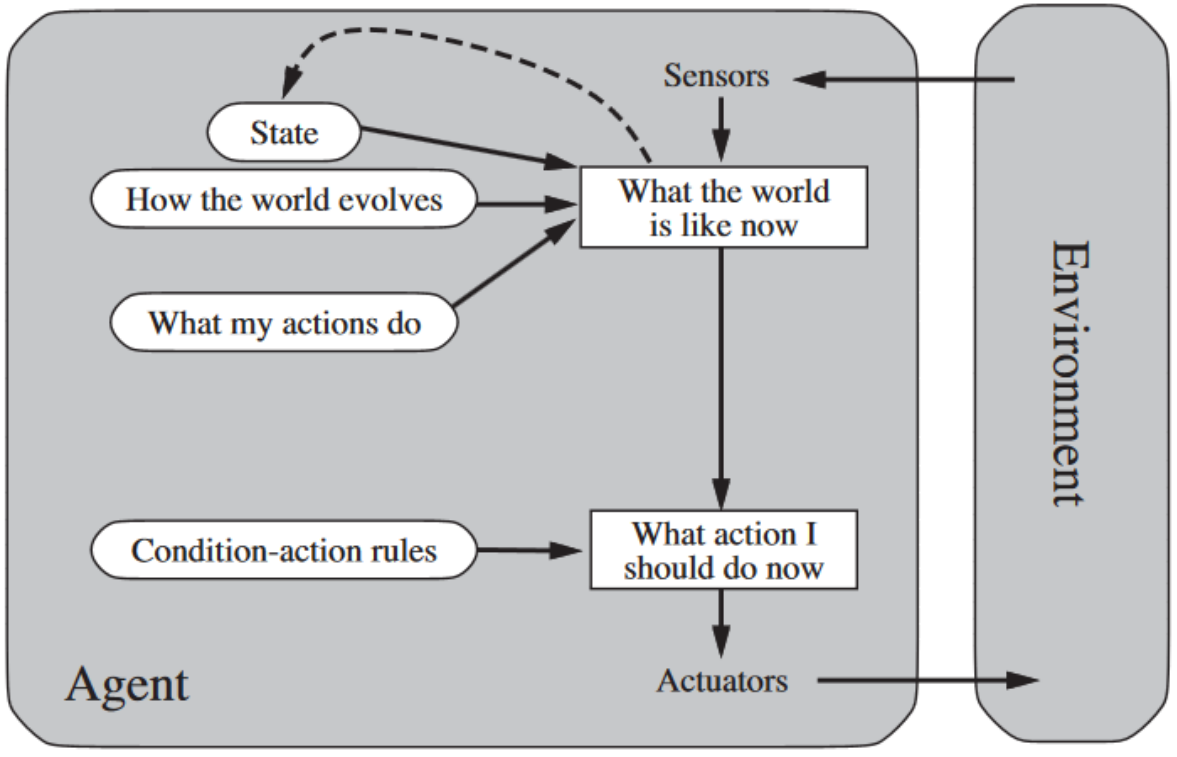
Simplex Reflex Agent (III)

- El mundo es generalmente parcialmente observable:
  - Sensores con "visión" limitada
  - Sensores solo captan una parte de la realidad
- Necesario un **estado interno** con un seguimiento de la parte del entorno que el agente no puede percibir en el momento actual

Simplex Reflex Agent (III)

- El mundo es generalmente parcialmente observable:
  - Sensores con "visión" limitada
  - Sensores solo captan una parte de la realidad
- Necesario un **estado interno** con un seguimiento de la parte del entorno que el agente no puede percibir en el momento actual

Model-based Reflex Agent (I)



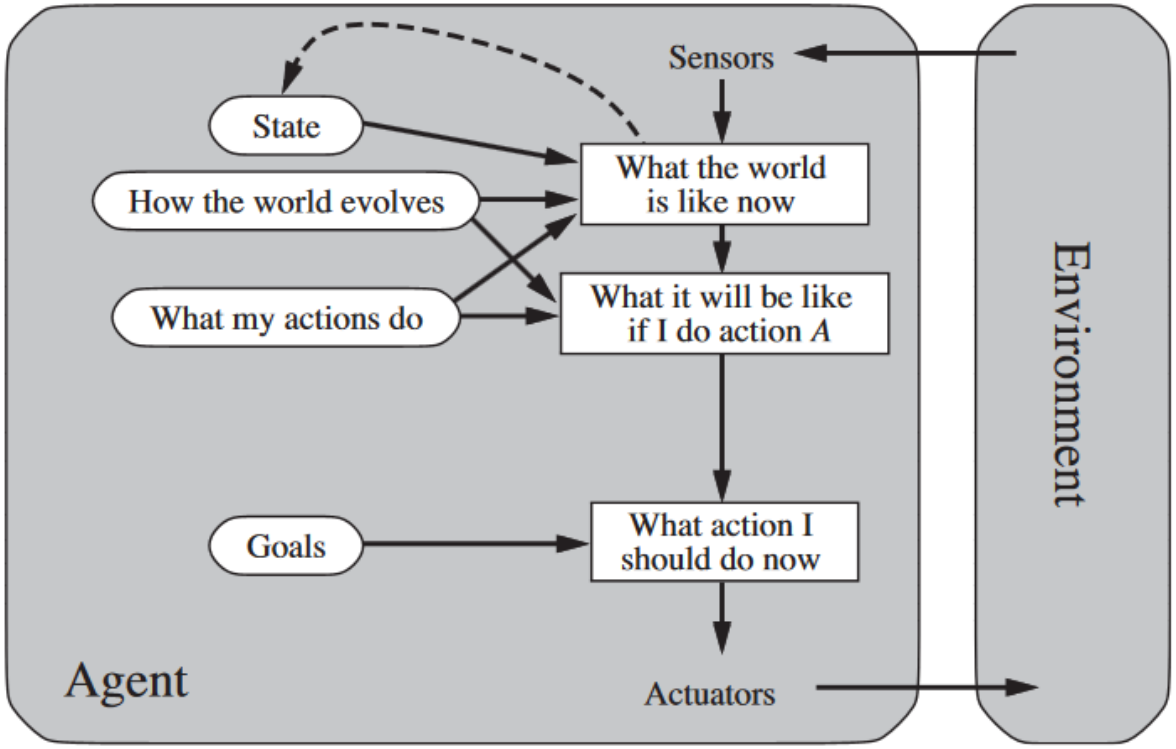
- Dos tipos de informaciones
  - Cómo evoluciona el entorno independientemente del agente
  - Cómo las acciones del agente afectan al entorno
- **Modelo del entorno**: cómo el nuevo estado depende del estado anterior y la acción que se ejecuta

Model-based Reflex Agent (II)

function model-based-reflex-agent:
    Input: rules condition-action, percept, model
    Output: action
    state $\leftarrow$ *UPDATE_STATE*(*state*, *action*, *percept*, *model*)
    rule $\leftarrow$ *RULE_MATCH*(*state*, *rules*)
    action $\leftarrow$ *rule.action*
return action

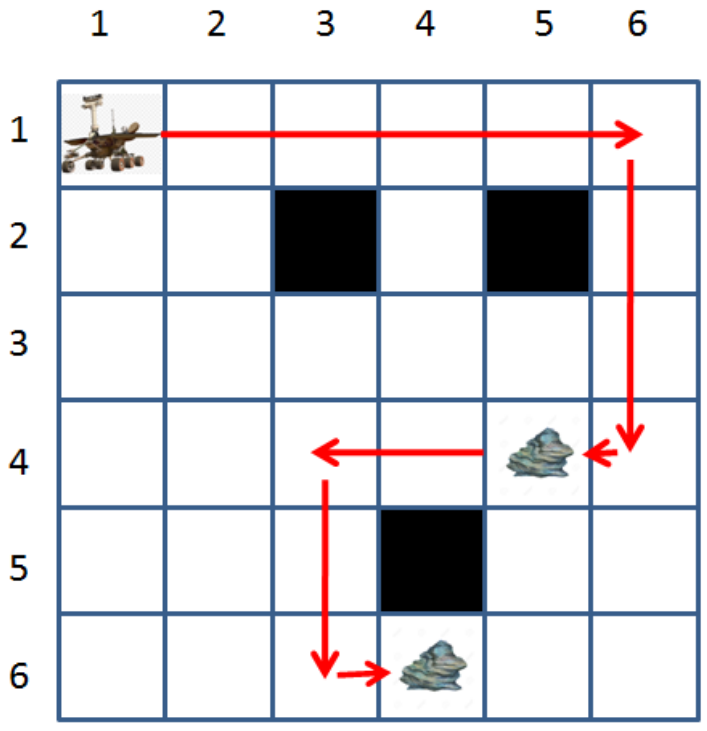
La percepción actual (incompleta) se combina con el antiguo estado interno para generar la descripción actualizada del estado actual (completo), basada en el modelo del agente de cómo funciona el mundo

Goal-based Agent (I)



- El estado actual podría no ser suficiente para decidir qué acción tomar
- Metas
- Razona sobre el futuro $\rightarrow$ Búsqueda y Planificación

Goal-based Agent (II)



Goals

transmitted-sample r1
transmitted-sample r2

Plan

#1: move w11 w12
#2: move w12 w13
...
#6: move w16 w26
...
#10: take-sample r1
...
#16: take-sample r2
#17: transmit-sample r1
#18: transmit-sample r2

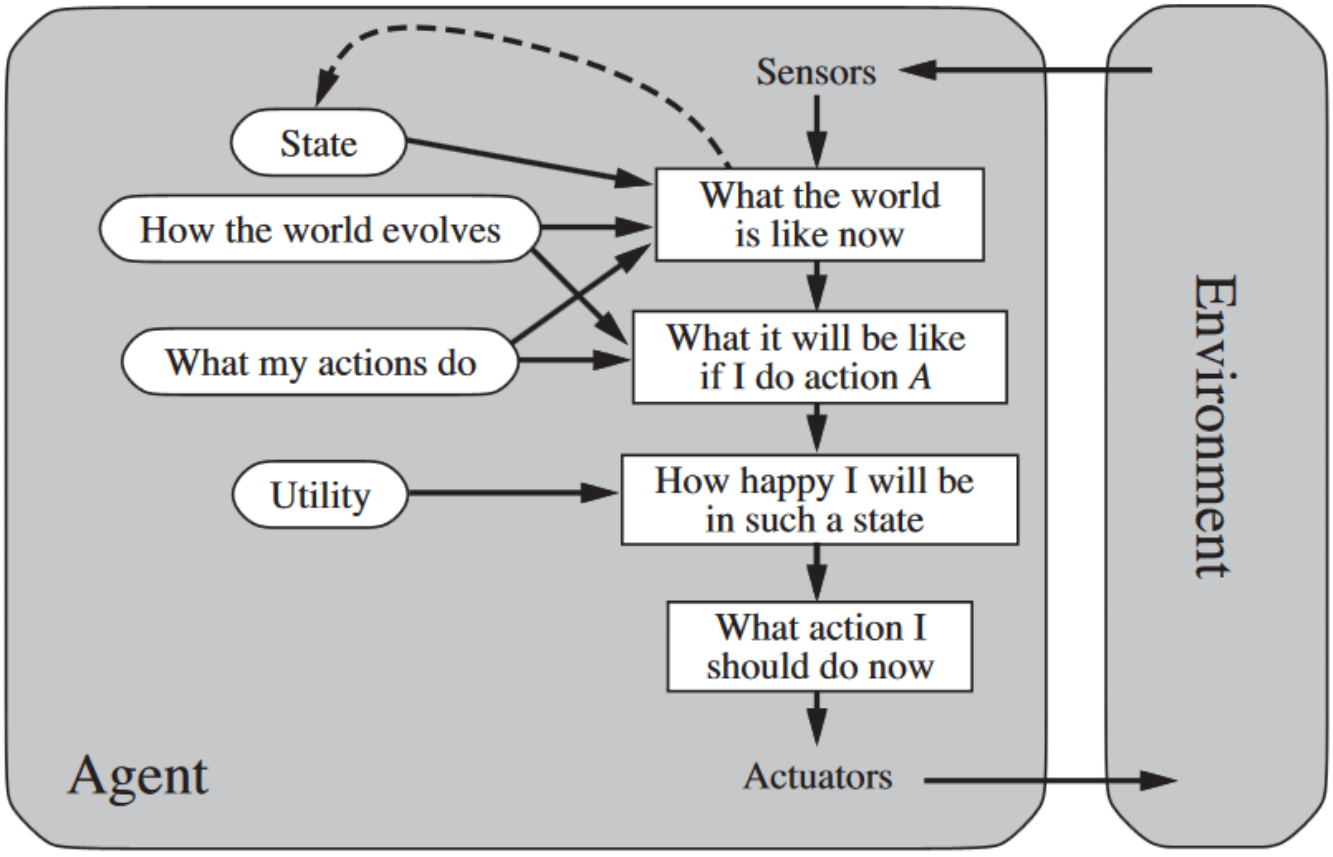
Goal-based Agent (III)

function goal-based-agent:
    Input: model, goals, percept
    state $\leftarrow$ *UPDATE_STATE*(*state*, *action*, *percept*, *model*)
    plan $\leftarrow$ *PLAN*(*state*, *model*, *goals*)
    while true **do**
        if goals $\in$ state **then** stop
        execute(pop(plan))
        state $\leftarrow$ *UPDATE_STATE*(*state*, *action*, *percept*, *model*)
    end

¿Qué ocurre si se llega a un estado que impide la ejecución del plan restante?

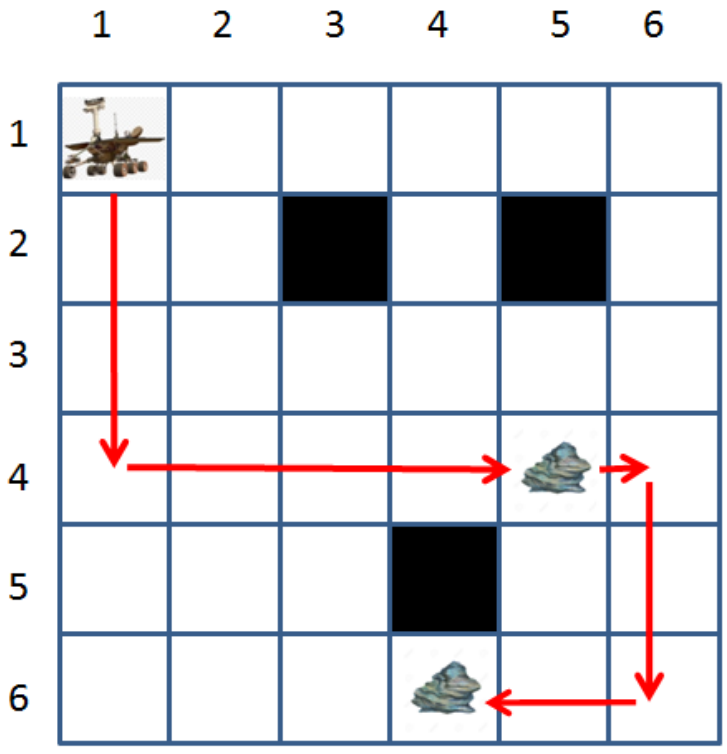
function goal-based-agent:
    Input: model, goals, percept
    state $\leftarrow$ *UPDATE_STATE*(*state*, *action*, *percept*, *model*)
    plan $\leftarrow$ *PLAN*(*state*, *model*, *goals*)
    while true **do**
        if goals $\in$ state **then** stop
        execute(pop(plan))
        state $\leftarrow$ *UPDATE_STATE*(*state*, *action*, *percept*, *model*)
        if !valid(state) **then**
            plan $\leftarrow$ *PLAN*(*state*, *model*, *goals*) // Replan
    end

Utility-based Agent (I)



A veces no es suficiente con solo llegar a la meta... Hay que hacerlo de la forma más rápida, más segura, o más barata $\rightarrow$ **Utilidad**

Utility-based Agent (II)



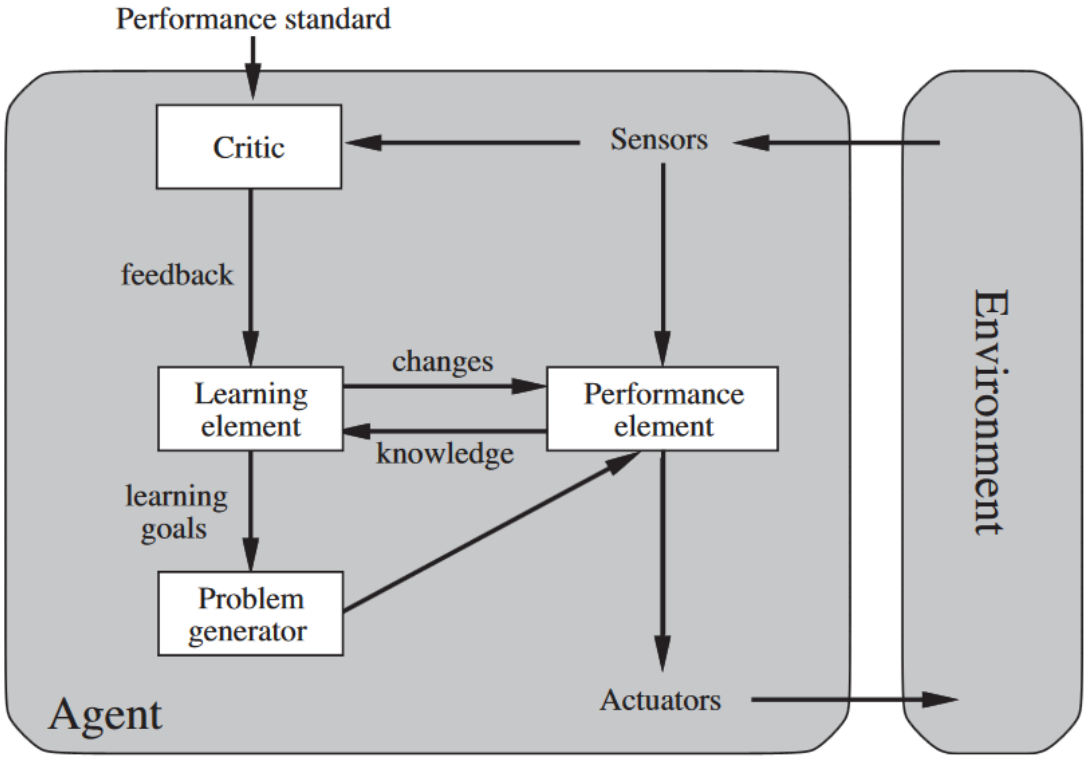
Goals

transmitted-sample r1
transmitted-sample r2

Plan

#1: move w11 w21
#2: move w21 w31
...
#4: move w41 w42
...
#8: take-sample r1
...
#14: take-sample r2
#15: transmit-sample r1
#16: transmit-sample r2

Learning Agent (I)



- **Todos los agentes anteriores se pueden convertir en agentes que aprenden y mejoran con la experiencia**
- Elementos más importantes:
  - *Learning element*: Responsable de aprender de la experiencia y sugerir mejoras en el proceso de selección de acciones
  - *Performance element*: **Los agentes básicos anteriores**

Learning Agent (II)

- El *critic* le dice a *learning element* lo bien o mal que lo está haciendo el agente de acuerdo a un estándar de rendimiendo
- *Problem generator*: Permite explorar nuevas acciones que pueden conducir a mejores comportamientos
- *Learning element*:
  - Aprendizaje supervisado/no supervisado: nuevas reglas o modificación de reglas ''malas" en agentes reactivos, cómo evoluciona el mundo y las consecuencias de las acciones en el resto de agentes, ...
  - Aprendizaje por refuerzo: Aprendizaje de políticas de comportamiento

Tipos de agentes

- ¿El agente tabular anterior es una buena forma de construir un agente?
- El desafío es descubrir cómo escribir agentes que produzcan un comportamiento racional a partir de un programa más pequeño que una gran tabla
- Muchas clasificaciones de agentes:
  - Russell and Norvig:
    - Simple Reflex Agent
    - Model-based Reflex Agent
    - Goal-based Agent
    - Utility-based Agent
    - Learning Agent
  - La comúnmente aceptada (y más cercana a robótica):
    - Deliberativos
    - Reactivos
    - Híbridos

Agentes Deliberativos (I)

- En el control deliberativo, el robot toma toda la información sensorial disponible y todo el conocimiento almacenado internamente que tiene, y piensa ("razona") al respecto para crear un **plan de acción** que le conduzca de un **estado inicial** a un **estado final** donde se cumplen las **metas**
- Para hacerlo, el agente debe buscar en todos los planes posibles hasta encontrar uno que haga el trabajo. Esto requiere que el robot razone sobre el futuro y piense en términos de: ''si hago esto a continuación, y luego sucede esto, y si luego hago esto a continuación, entonces sucede esto,..." y así sucesivamente
- Esto puede llevar mucho tiempo, por lo que si el robot debe reaccionar rápidamente, puede que no sea práctico. Sin embargo, si hay tiempo, esto permite que el robot actúe estratégicamente
- Dos conceptos clave:
  - **Modelo** (representación simbólica) del mundo (entorno, agente, otros agentes), explícitamente representado
  - **Razonamiento sobre el futuro**

Agentes Deliberativos (II)

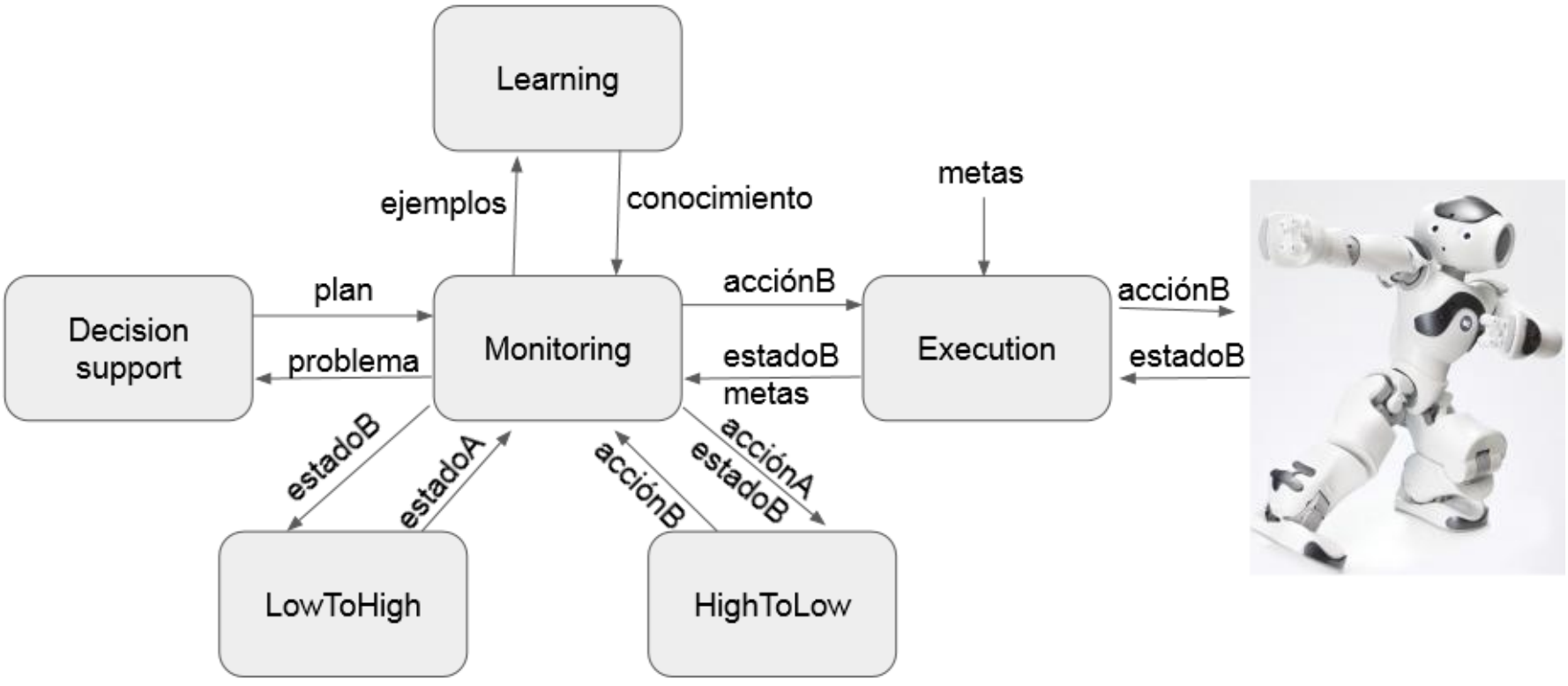
- La visión dominante en la comunidad de IA era que un sistema de control para un robot autónomo móvil debe ser descompuesta en tres elementos funcionales [Nilsson, 1980]:
  - Sensado: traducir la información en bruto de los sensores a una representación/estado de alto nivel
  - Planificación: Tomar este estado, las metas, y el modelo, para generar un plan
  - Ejecución: Tomar el plan y ejecutar cada una de las acciones
- La percepción no está ligada directamente a la acción que se ejecuta, hay un proceso de razonamiento
- Conocidos también como agentes de planificación, o agentes BDI (*Belief-Desire-Intention*)

Agentes Deliberativos (III)

- Ejemplo:
  - **Estado inicial**: Estar en casa, tener una foto, hay clavos, no hay martillo.
  - **Estado final**: Fotografía enmarcada y colocada en la pared.
  - **Plan**:
    1. Ir a la tienda
    2. Adquirir un marco y un martillo
    3. Ir a casa
    4. Enmarcar la foto
    5. Usar el martillo y los clavos para colgar la fotografía en la pared

Agentes Deliberativos (IV)

Arquitectura de control deliberativo [Guzmán et al., 2010]:



Dos niveles de abstracción: bajo-nivel (información sensorial, comandos interpretables por el robot), alto-nivel (información simbólica del mundo, acciones del plan)

Agentes Reactivos (I)

- El control reactivo es una técnica que acopla estrechamente las entradas sensoriales y las salidas del actuador, para permitir que el robot responda muy rápidamente a entornos cambiantes y no estructurados.
- Piensa en el control reactivo como ''estímulo-respuesta''. Método de control potente: muchos animales son en gran parte reactivos.
- Las limitaciones de este enfoque son que dichos robots, debido a que solo buscan acciones para cualquier entrada sensorial, **generalmente** no guardan mucha información, no tienen memoria, no tienen representaciones internas del mundo que los rodea y no tienen la capacidad de aprender con el tiempo
- Dos conceptos claves:
  - Sin representación o con poca representación del mundo: mundo dinámico, difícil modelar incertidumbre
  - **No razonan sobre el futuro**

Agentes Reactivos (II)

- Ejemplo:

  Un planeta distante contiene oro. Varios vehículos independientes están disponibles para recolectar este oro. Las muestras recogidas deben llevarse a la nave nodriza que se encuentra en el mismo planeta. No se sabe dónde está el oro. Debido a la topografía del planeta no hay conexión entre vehículos. La nave envía señales de radio que le permiten ser localizada.

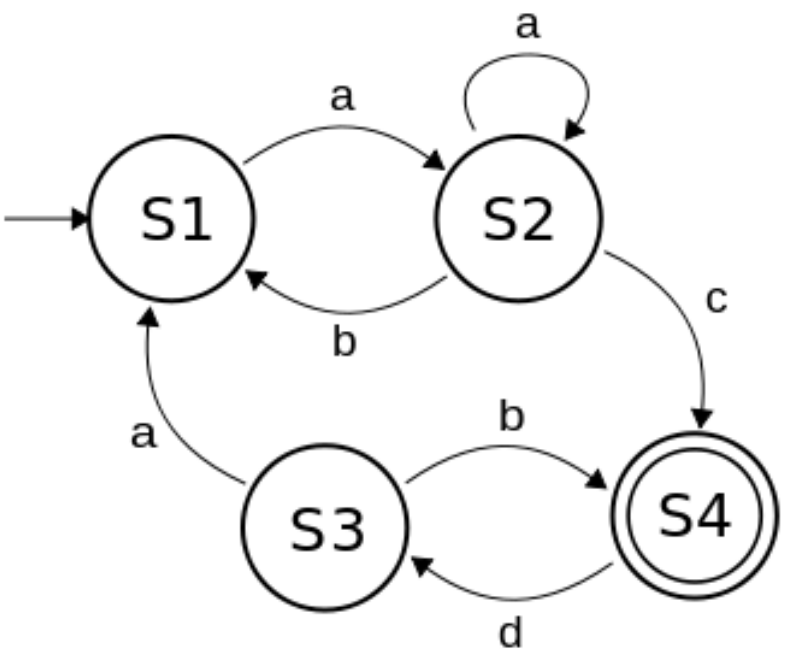
  Reglas $S \rightarrow A$:

  - **if** detects an obstacle **then** change direction
  - **if** (samples on board AND at the base) **then** drop samples
  - **if** (samples on board AND not on base) **then** follow the radio signal
  - **if** detect samples **then** collect samples
  - **if** (not samples on board AND not on base) **then** take a random path

Agentes Reactivos (III)

- El conjunto de reglas anterior se puede expresar formalmente como una **Máquina Finita de Estados (FSM)**.
- Máquinas finitas de estados: Herramienta conceptual para modelar comportamientos reactivos
- Más poder expresivo que simples *if-else*: dificultad para modelar cada combinación estado, entrada, transición, y secuencias de decisiones que dependen del contexto o de decisiones previas
- FSMs simplifican la gestión del estado del sistema
- FSMs ideales para la construcción de comportamientos incrementalmente más complejos
- **Arquitectura de Subsunción** [Brooks, 86]: Conjunto de FSMs (comportamientos), que realizan tareas
  - Los comportamientos están ordenados por capas
  - Los comportamientos de las capas más bajas (mayor prioridad) inhiben a los de las capas superiores

Máquinas Finitas de Estados (I)

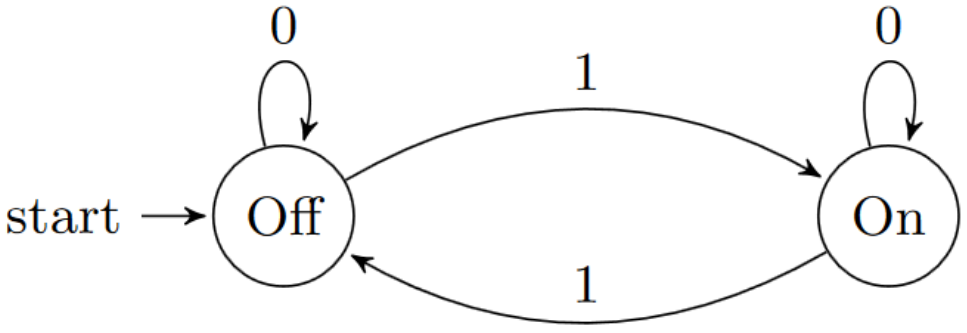


Una máquina finita de estados se describe mediante una tupla $\mathcal{A} = (S, \Sigma, \delta, s_0, F)$ donde:

- S es un conjunto finito de estados ($S1$, $S2$, $S3$, $S4$)
- Σ es el alfabeto de entrada (a, b, c, d)
- δ es la función de transición $S \times \Sigma \to S$ ($S1 \times a \to S2$, ...)
- s_0 es el estado inicial $s_0 \in S$ ($S1$)
- F conjunto de estados finales $F \subseteq S$ ($S4$)

Máquinas Finitas de Estados (II)

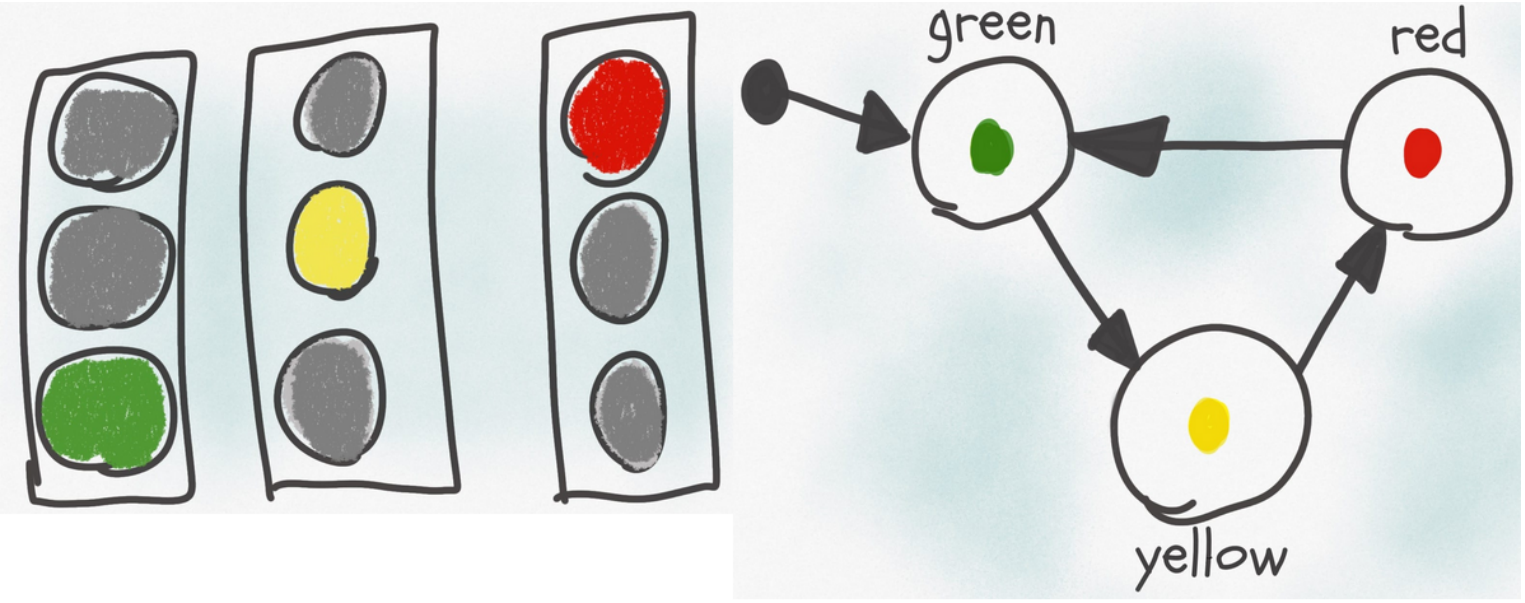
On/Off Switch



```
enum states {stateON, stateOFF};
enum states currentState = stateON;
int main(){
    while(1){
      switch(currentState){
        case stateON:
          if(board.buttonA.isPressed ())
            currentState = stateOFF;
        break;
        case stateOFF:
          if(board.buttonA.isPressed ())
            currentState = stateON;
        break;
      }
...
```

Máquinas Finitas de Estados (III)

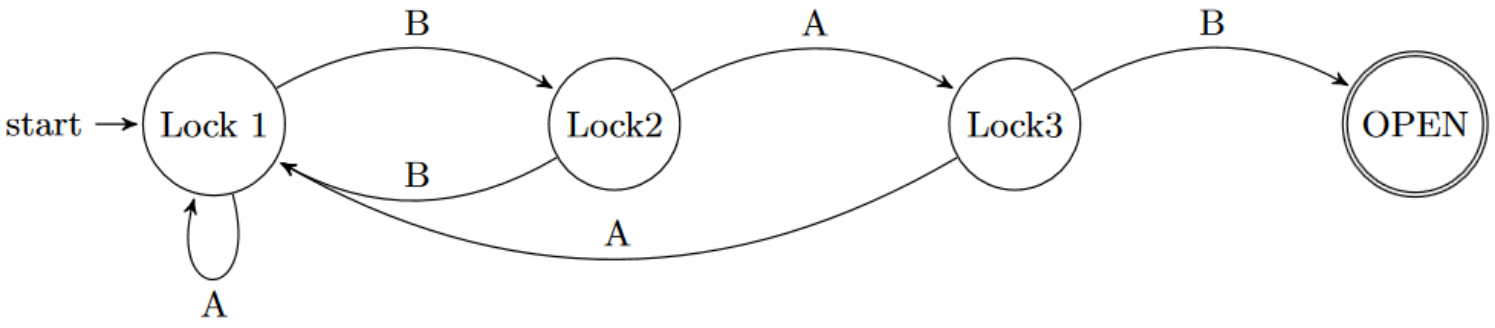
Semáforo



Entradas: Timer 50s, Timer 8s, Timer 32s

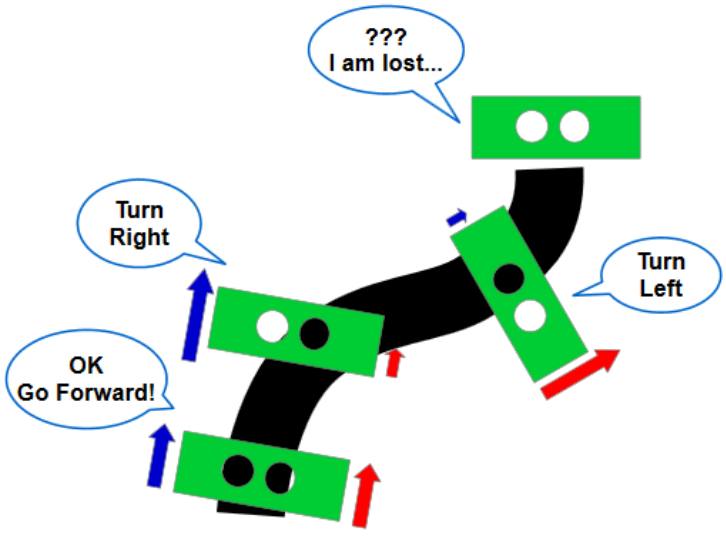
Máquinas Finitas de Estados (IV)

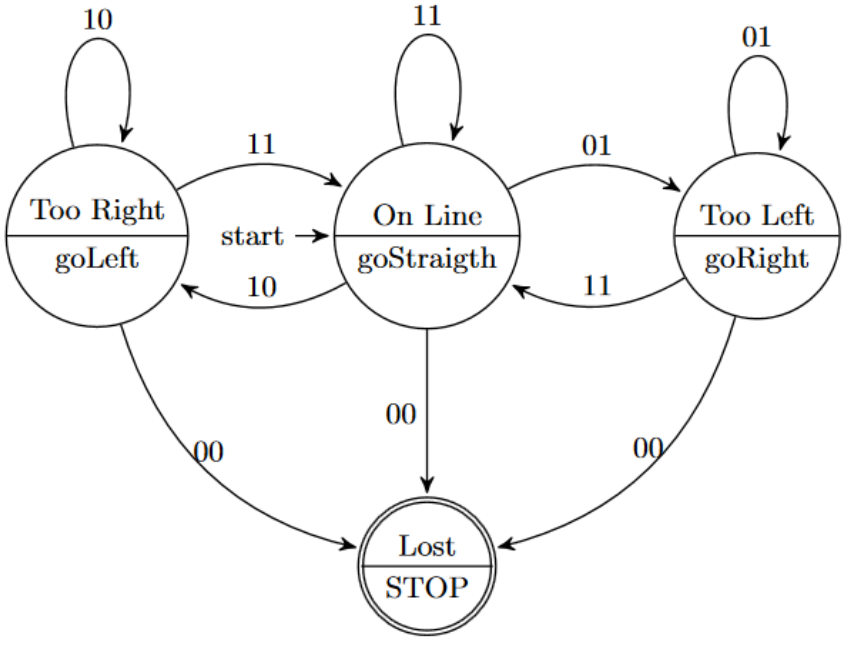
¿Cuál es la combinación que abre la caja fuerte?



Máquinas Finitas de Estados (V)

Robot sigue-líneas

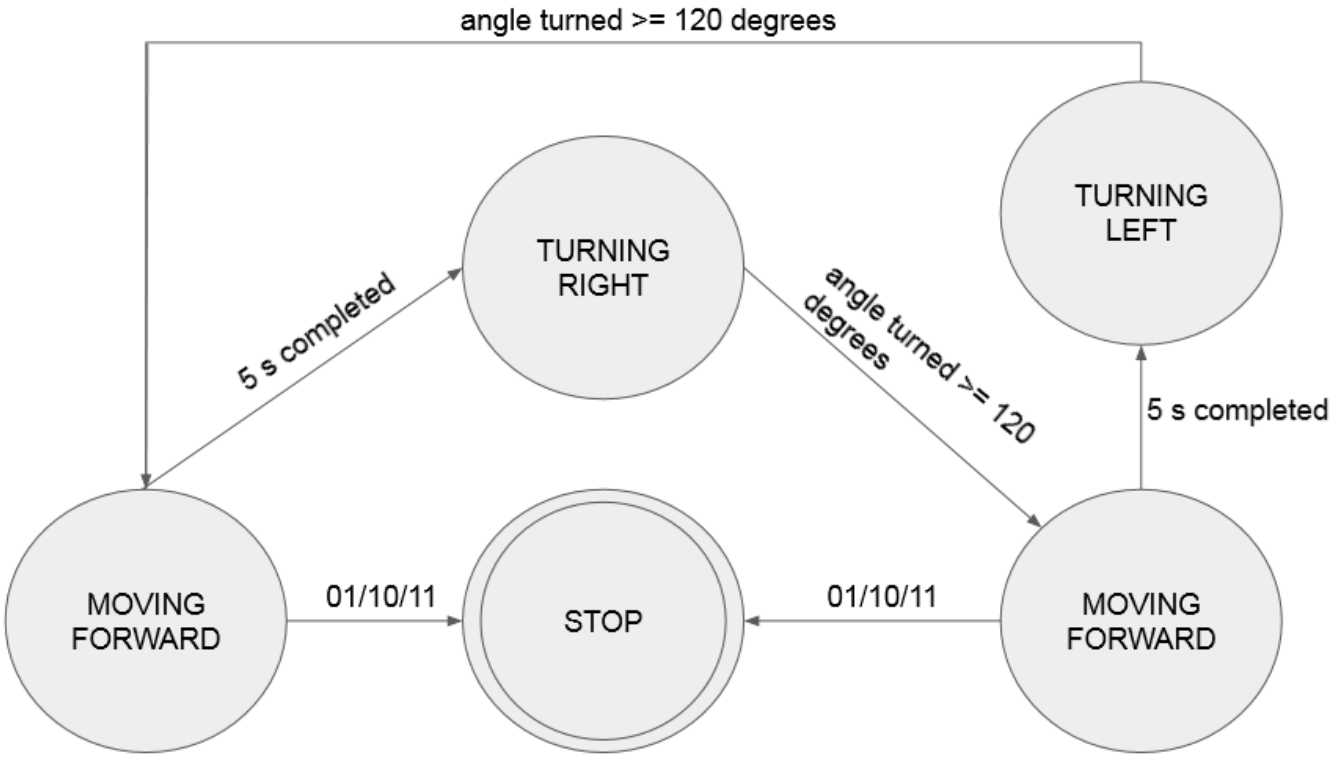




En robótica **generalmente** los estados se corresponden con comportamientos simples (e.g., esperar, moverse, recargar, girar), y las transiciones con cualquier evento que haga cambiar de un estado a otro (e.g., información sensorial)

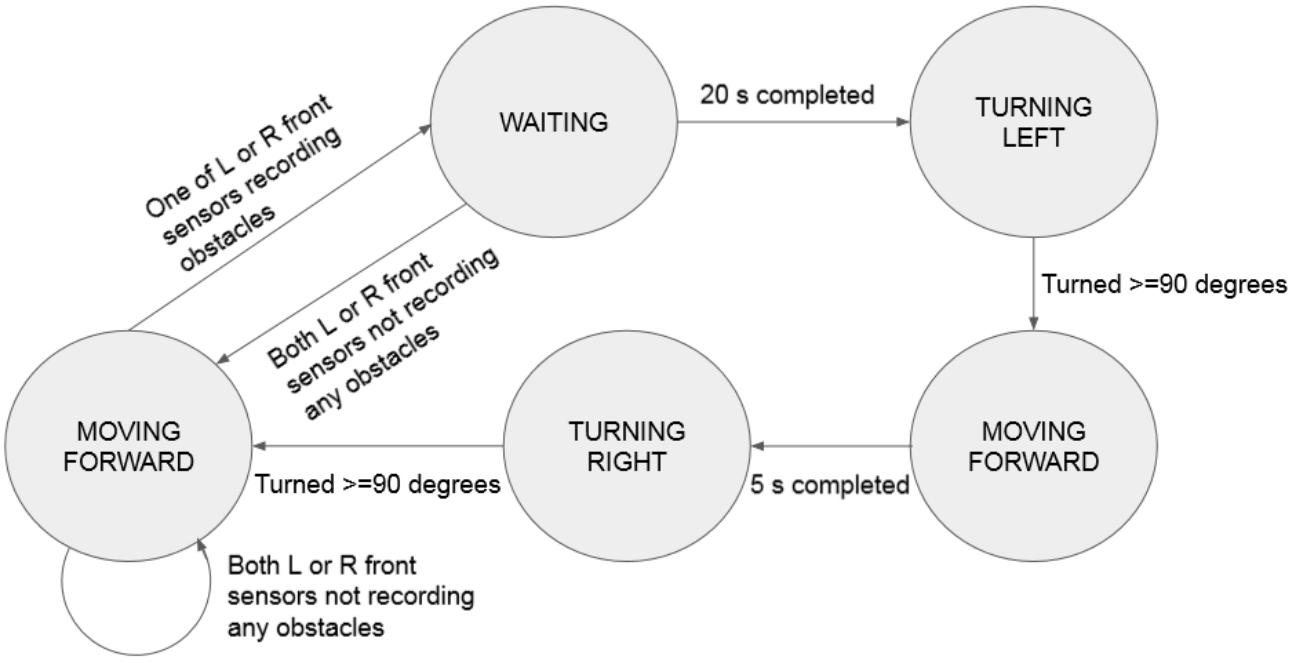
Máquinas Finitas de Estados (VI)

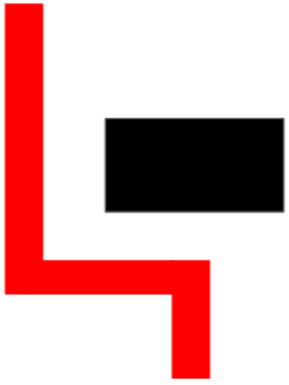
Robot busca-líneas



Máquinas Finitas de Estados (VII)

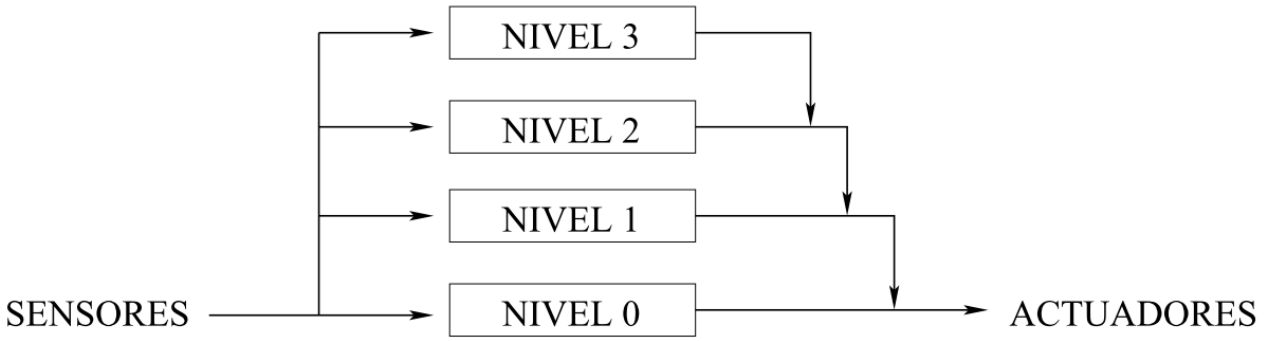
Robot evita-obstáculos





Arquitectura de Subsunción (I)

- Descomposición de controladores en **niveles de competencia**
- Añade comportamientos incrementalmente
- Las capas de alto nivel se sustentan en las de bajo nivel
- Las capas de alto nivel pueden **subsumir/inhibir** las de bajo nivel
- Las capas se activan en paralelo
- Cada capa trabaja independientemente

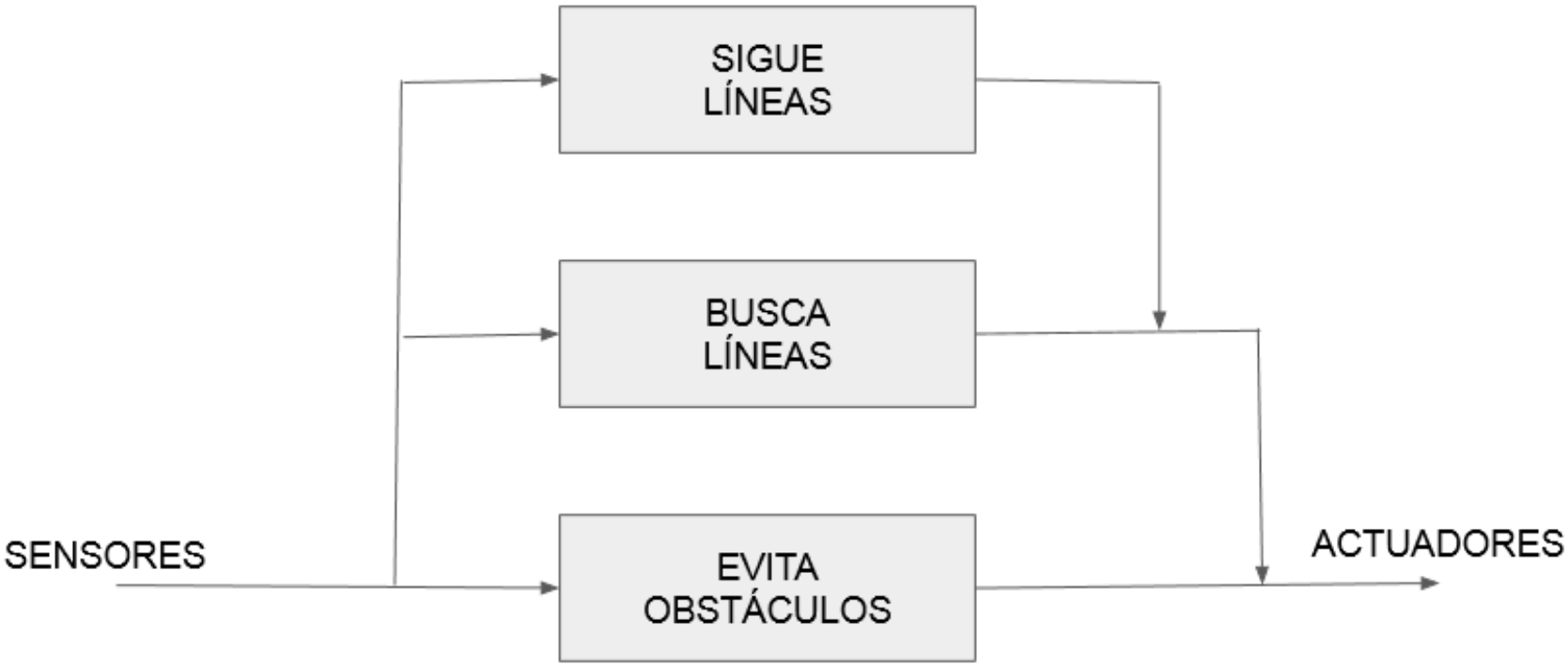


Arquitectura de Subsunción (II)

- **Niveles de competencia**:
  - Define los comportamientos deseados: Evitar Obstáculos, Explorar, traer comida,...
  - Estos comportamientos pueden definirse como FSM
  - Cada nivel se puede implementar independientemente
  - Permite una gran escabilidad
- El objetivo es empezar con un comportamiento sencillo en el nivel 0, crear un nivel 1 con un comportamiento más complejo, un nivel 2 con uno aún más complejo, y así sucesivamente
- Mediante esta combinación de comportamientos se aumentan las habilidades del robot de forma incremental

Arquitectura de Subsunción (III)

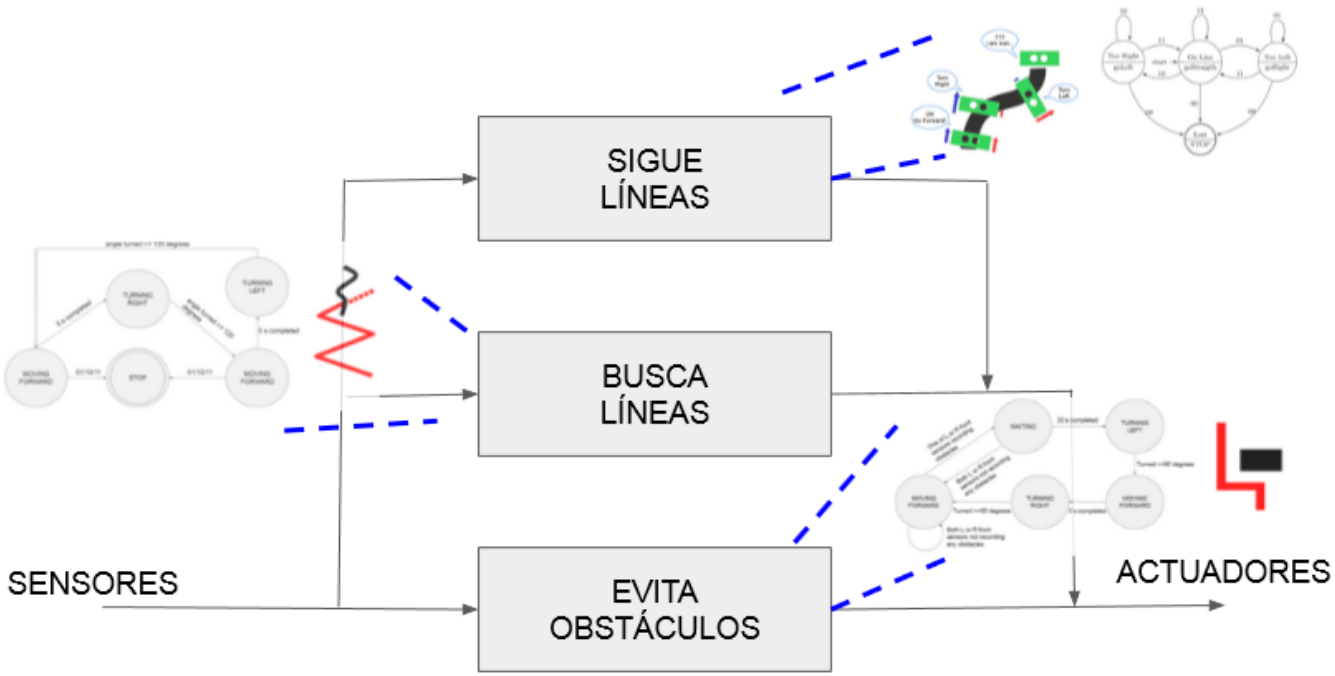
Robot busca-sigue-líneas y evita obstáculos



- En el nivel más básico el robot *evita obstáculos*
- El nivel *busca líneas* subsume/inhibe el nivel *evita obstáculos*, pero si aparece un obstáculo el nivel *evita obstáculos* se activa de nuevo
- El nivel *sigue líneas* subsume/inhibe los otros niveles, que pueden volver a activarse si es necesario

Arquitectura de Subsunción (IV)

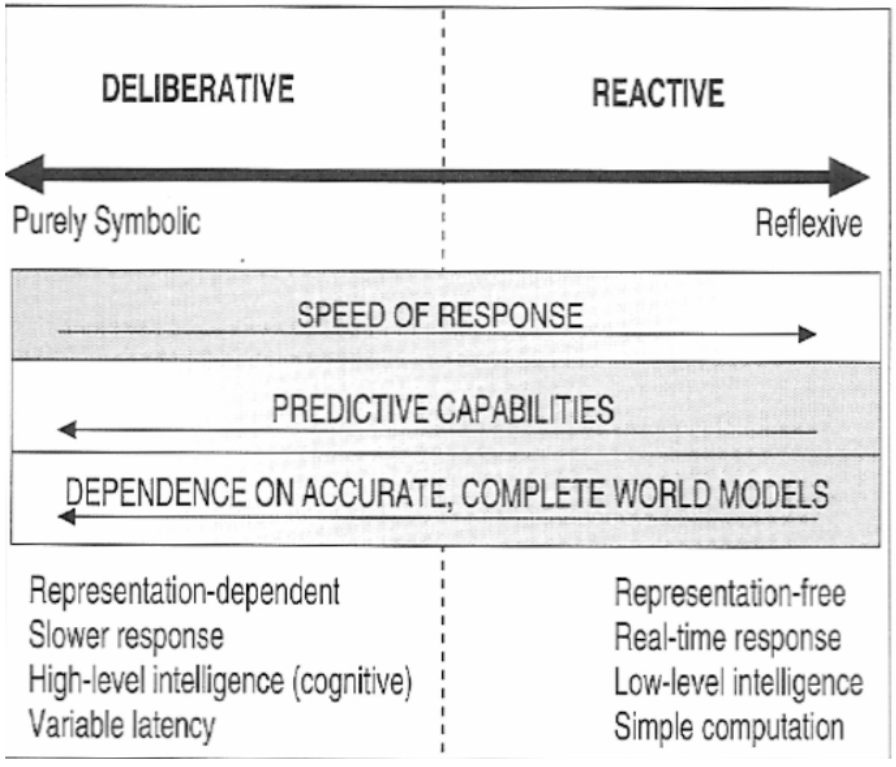
Robot busca-sigue-líneas y evita obstáculos



- En el nivel más básico el robot *evita obstáculos*
- El nivel *busca líneas* subsume/inhibe el nivel *evita obstáculos*, pero si aparece un obstáculo el nivel *evita obstáculos* se activa de nuevo
- El nivel *sigue líneas* subsume/inhibe los otros niveles, que pueden volver a activarse si es necesario

Reactivo vs. Deliberativo (I)

Model-Based



Behavior-Based

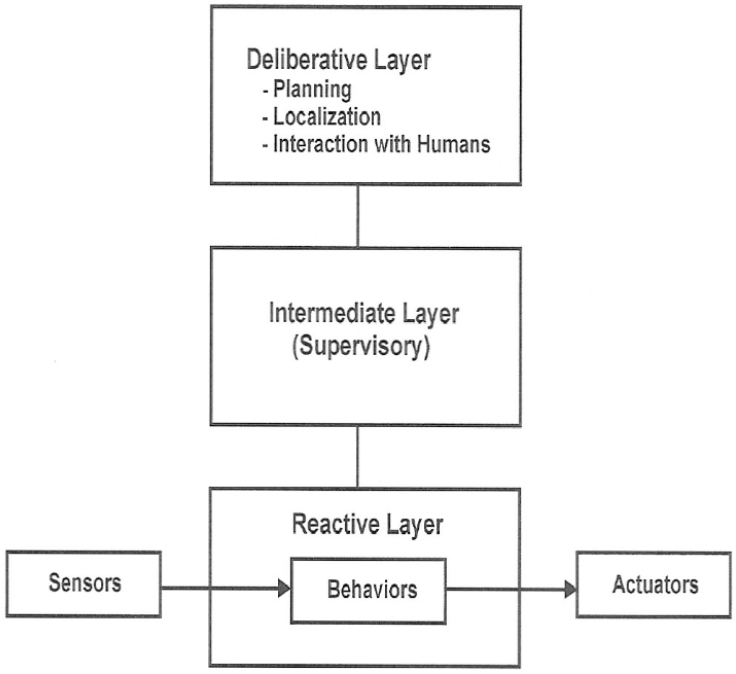
Reactivo vs. Deliberativo (II)

- Ningún enfoque es "el mejor" para todos los robots; cada uno tiene sus fortalezas y debilidades
- El control requiere algunas concesiones inevitables porque:
  - Pensar es lento
  - La reacción debe ser rápida
  - Pensar permite mirar hacia adelante (planificar) para evitar malas acciones
  - Pensar demasiado tiempo puede ser peligroso (por ejemplo, caerse en un precipicio)
  - Para pensar, el robot necesita (mucha) información precisa del mundo
  - El mundo sigue cambiando a medida que el robot piensa, por lo que cuanto más lento piensa, más inexactas son sus soluciones
  - El control reactivo requiere enumerar todos los posibles comportamientos (estados) del sistema
- Como resultado de estas compensaciones, algunos robots no piensan en absoluto, mientras que otros piensan mucho y actúan muy poco. **¡Todo depende de la tarea del robot y de su entorno!**

Agentes Híbridos

- En el control híbrido, el objetivo es combinar lo mejor del control reactivo y deliberativo. En él, una parte de los planes del "cerebro" del robot, mientras que otra se ocupa de la reacción inmediata, como evitar obstáculos y permanecer en el camino
- El desafío de este enfoque es unir las dos partes del cerebro y permitirles hablar entre sí y resolver los conflictos entre las dos

- Esto requiere una "tercera" parte del cerebro del robot y, como resultado, estos sistemas a menudo se denominan "sistemas de tres capas":
  - La capa inferior es la capa reactiva, en la que los sensores/actuadores están estrechamente acoplados
  - La capa superior proporciona el componente deliberativo (e.g., planificación, localización)
  - La intermedia entre los dos a veces se denomina capa de supervisión $\rightarrow$ cuello de botella



Equivalencia entre clasificaciones

<table>
<tr><th colspan="2">Russel and Norvig</th><th colspan="2">Otra</th></tr>
<tr><td rowspan="2">Learning Agent</td><td>Simple reflex agent
Model-based reflex agent</td><td>Reactivo</td><td rowspan="2">Hibrido</td></tr>
<tr><td>Goal-based agent
Utility-based agent</td><td>Deliberativo</td></tr>
</table>

- Si tiene un modelo del mundo y **razona sobre el futuro** $\rightarrow$ deliberativo