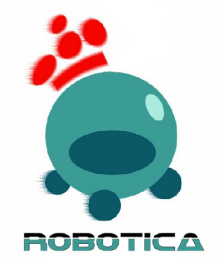
Robótica Móvil

José María Cañas

josemaria.plaza@urjc.es



Grado Ingeniería Robótica Software, Curso 2022-2023

Sistemas Reactivos y Control

Contenidos

- Introducción
- Sistemas Reactivos
- Autómatas de Estado Finito
- Control Clásico
- Control Borroso
- Control Neuronal

Introducción

- Sensores
- Actuadores
- ¿Cómo generamos comportamiento en robots móviles?
- Autonomía, inteligencia
- Objetivo
- Sensible al entorno
- Responder adecuadamente a las situaciones
- Spirit, Opportunity, RoboCup, Urban Challenge, Roomba

Complejidad del comportamiento

- ¿Por qué no tenemos robots que hagan las tareas domésticas?
- Falta flexibilidad
- Tareas complejas
- ¿Problema tecnológico o teórico?
- No es sólo un problema de programación

Arquitectura cognitiva

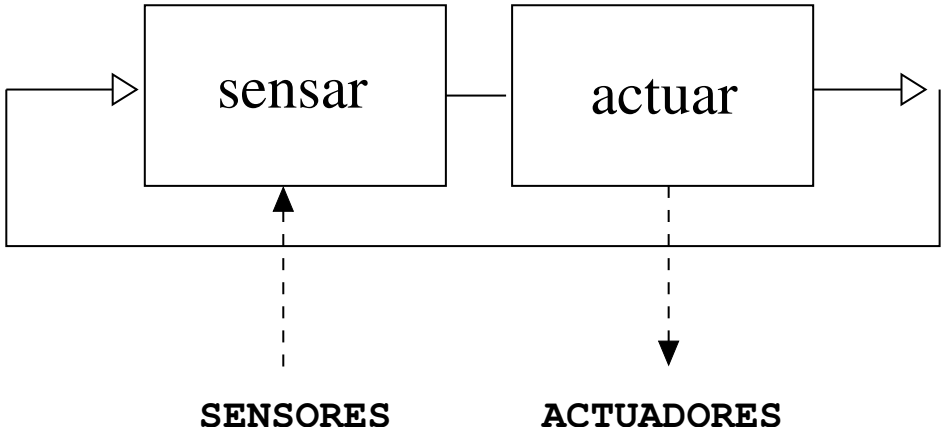
La arquitectura de un robot es la organización de sus capacidades sensoriales, de procesamiento y de acción para conseguir un repertorio de comportamientos inteligentes interactuando con cierto entorno

- La arquitectura determina el comportamiento observable
- Un robot móvil es un sistema complejo
- Para comportamientos sencillos, casi cualquier organización vale
- ¿Cuándo?
- Diferentes escuelas

Uno, varios, muchos

- Termostato, Roomba
- Repertorio de comportamientos
- Del cómo al cuándo
- Selección de acción
- Información desbordante, incierta
- Atención
- Visión computacional es complicada y potente

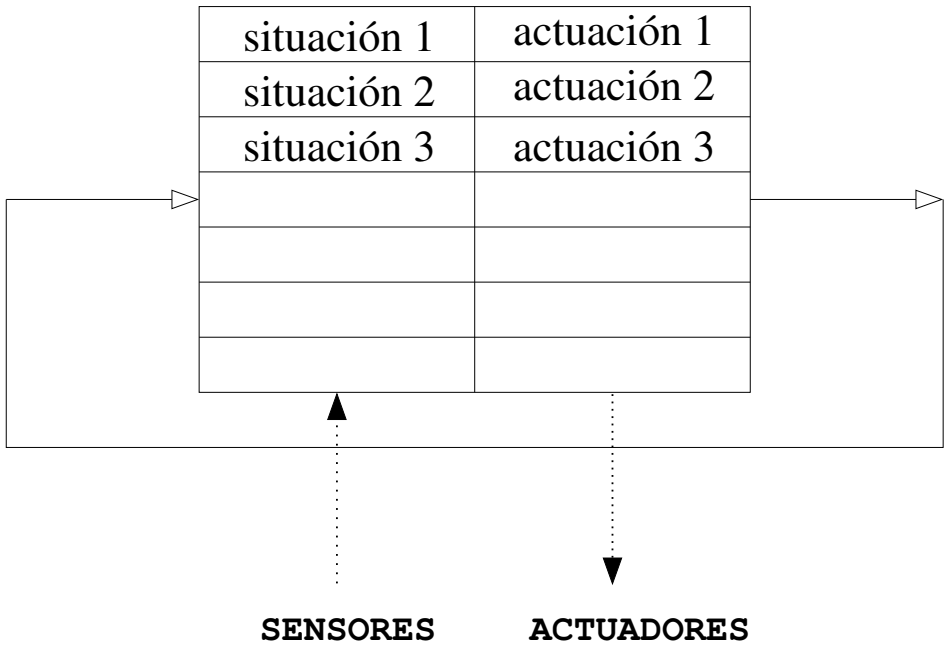
Sistemas Reactivos

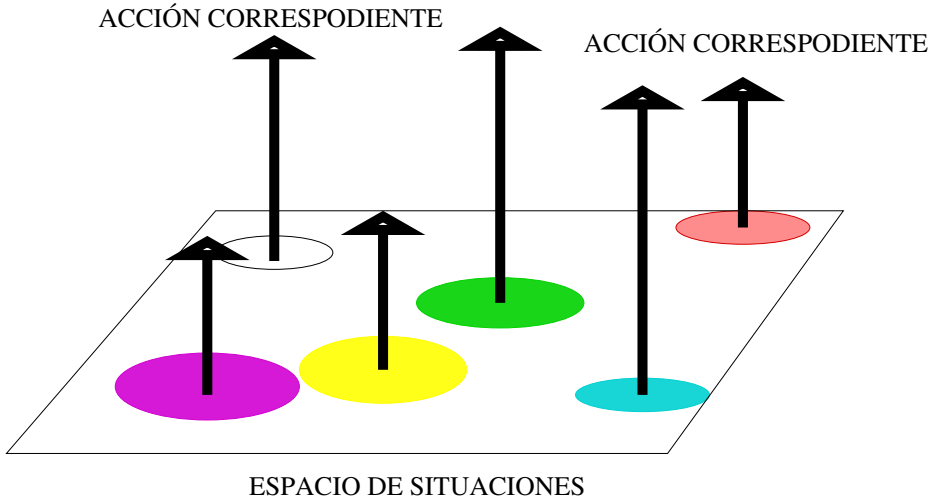


- Interacción continua con el entorno, acción situada
- Percepción subsimbólica
- Bucle cerrado sensores-actuadores

Colección de reglas de correspondencia situación-acción

- Escala de tiempo: corto plazo
- No necesita/usa capacidad de predicción
- No suele utilizar representación interna del mundo
- Menor necesidad de capacidad de cálculo
- Divide el mundo en situaciones *mutuamente excluyentes*
- Cada "situación" dispara una o más acciones
- Una situación puede venir definida por uno o más sensores
- Análogo a los reflejos en el sistema nervioso

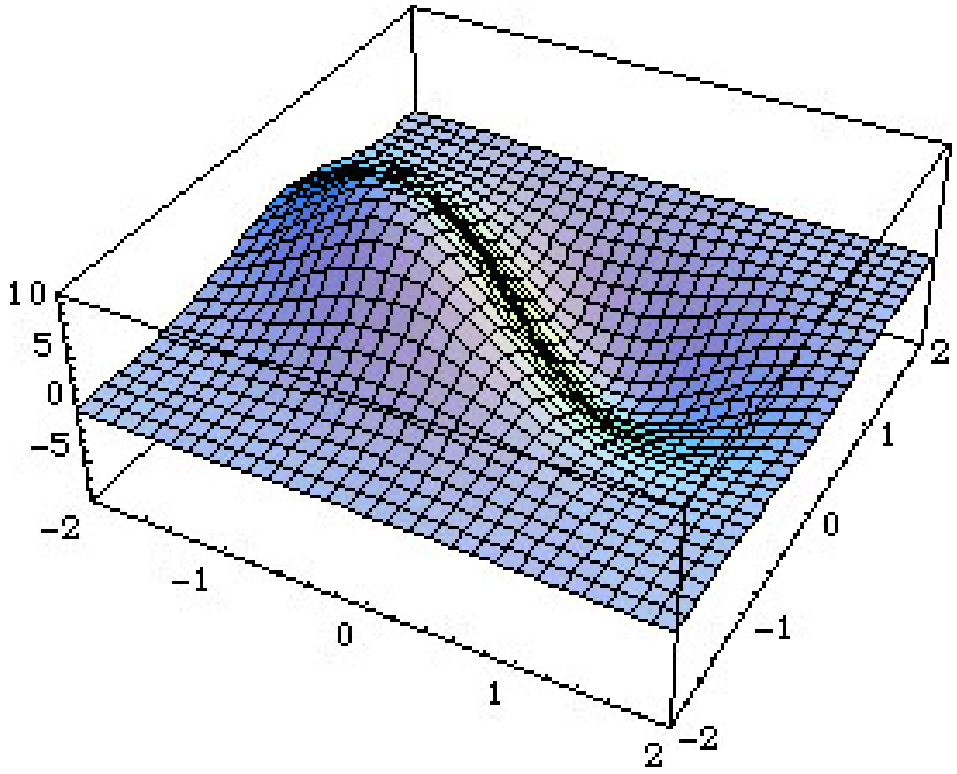

situación 1
actuación 1
situación 2
actuación 2
situación 3
actuación 3
SENSORES
ACTUADORES


ACCIÓN CORRESPODIENTE
ACCIÓN CORRESPODIENTE
ESPACIO DE SITUACIONES

Control basado en casos

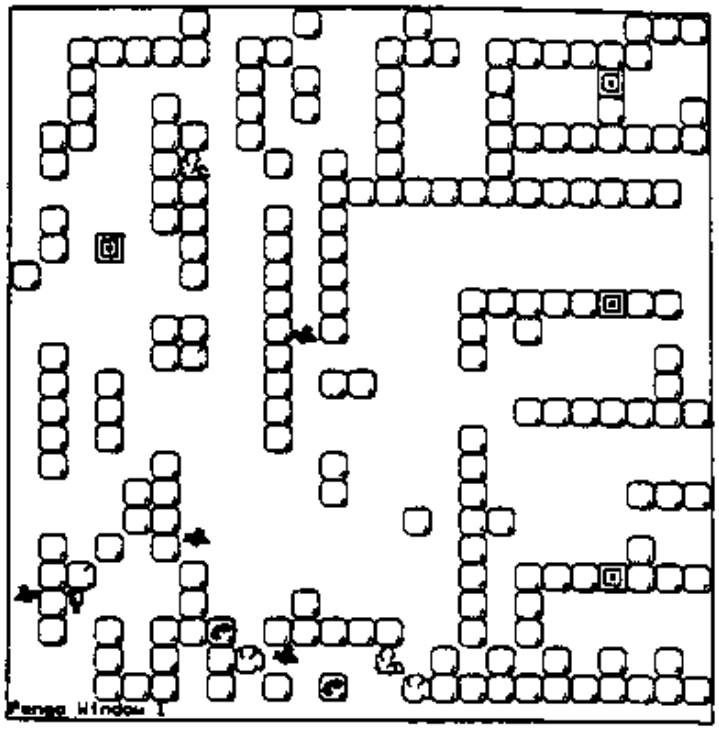
- Tabla de correspondencia situación acción
- Funcionamiento iterativo
- Ágil, iteraciones rápidas
- Frecuencia de iteraciones
- Giro controlado vs Giro ciego
- Permite reaccionar ante imprevistos.
- Aquí y ahora. No tienen horizonte temporal

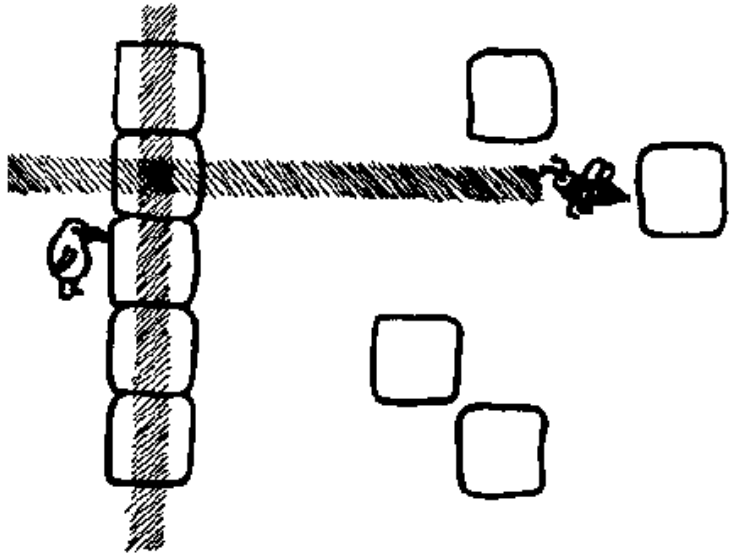
Superficie de control



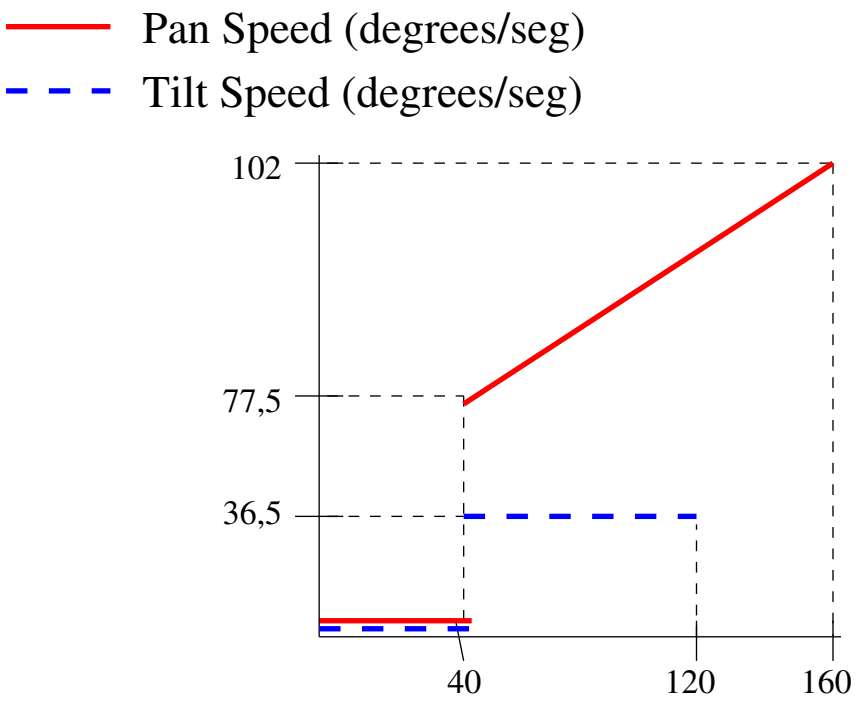
- ¿Qué actuación (z) ordeno en la situación (x, y)?

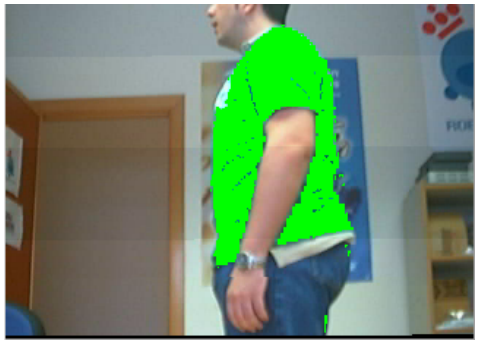
Ejemplos





- Pengi juega a Pengo, la-abeja-que-me-persigue
- Sigue lineas con LEGO, sigue lineas visual





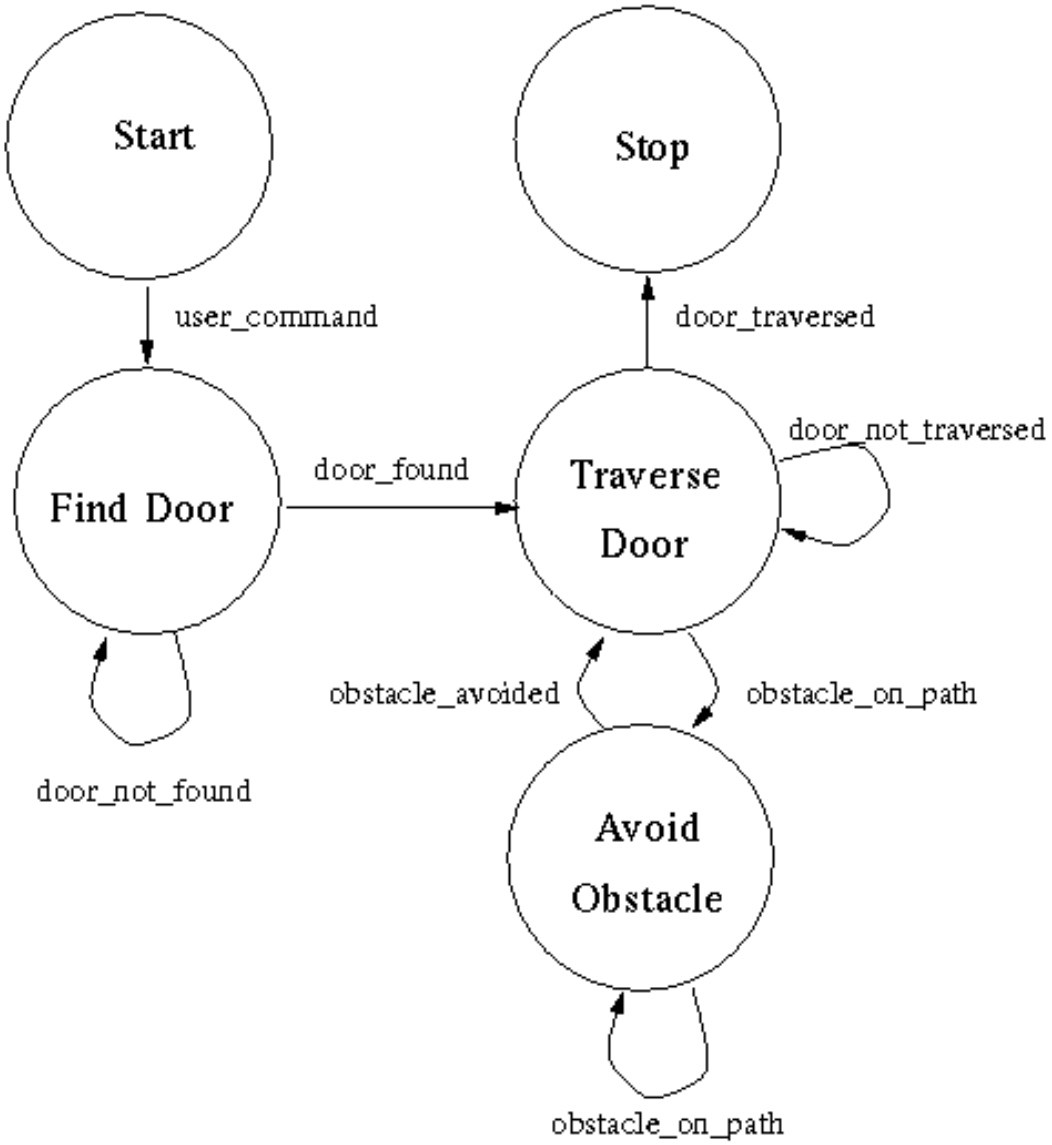
- Sigue lineas con LEGO,
- Sigue lineas visual
- Seguir a una persona

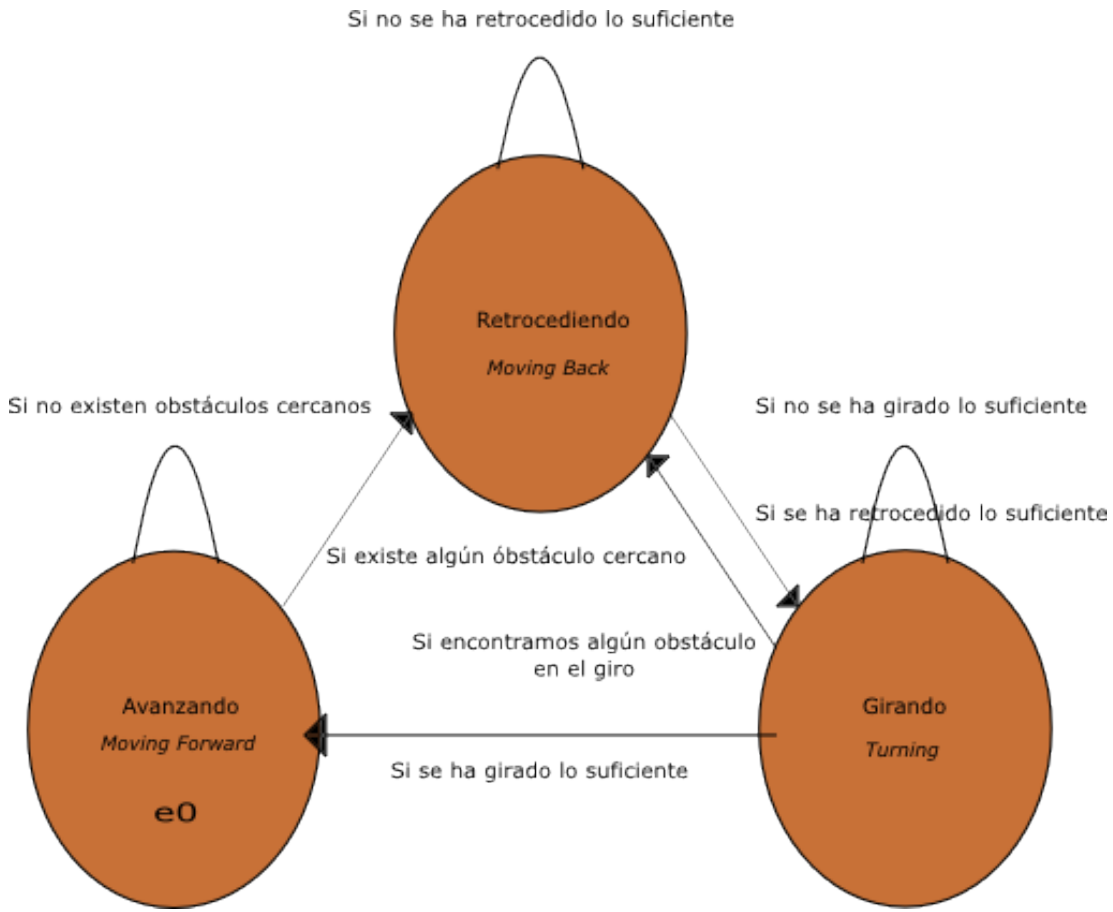
Problemas de los sistemas reactivos

- No escalan.
- Se cambia simplicidad de ejecución por complejidad de diseño
- Es difícil "separar" las situaciones del mundo.
- Número exponencial de situaciones en función del número de sensores.
- Se suelen usar simplicaciones: considerar sólo acciones para "ciertas" situaciones (estado)
- Vacíos y solapes de control.
- Si no se consideran todas las situaciones y sus combinaciones se llega al problema del "arbitraje".

Autómatas de estado finito

- Estados
- En cada estado una actuación o un controlador
- En cada estado una percepción
- No pierde ejecución reactiva


Start
Stop
user_command
door_traversed
Find Door
door_found
Traverse Door
door_not_traversed
obstacle_avoided
obstacle_on_path
door_not_found
Avoid Obstacle
obstacle_on_path

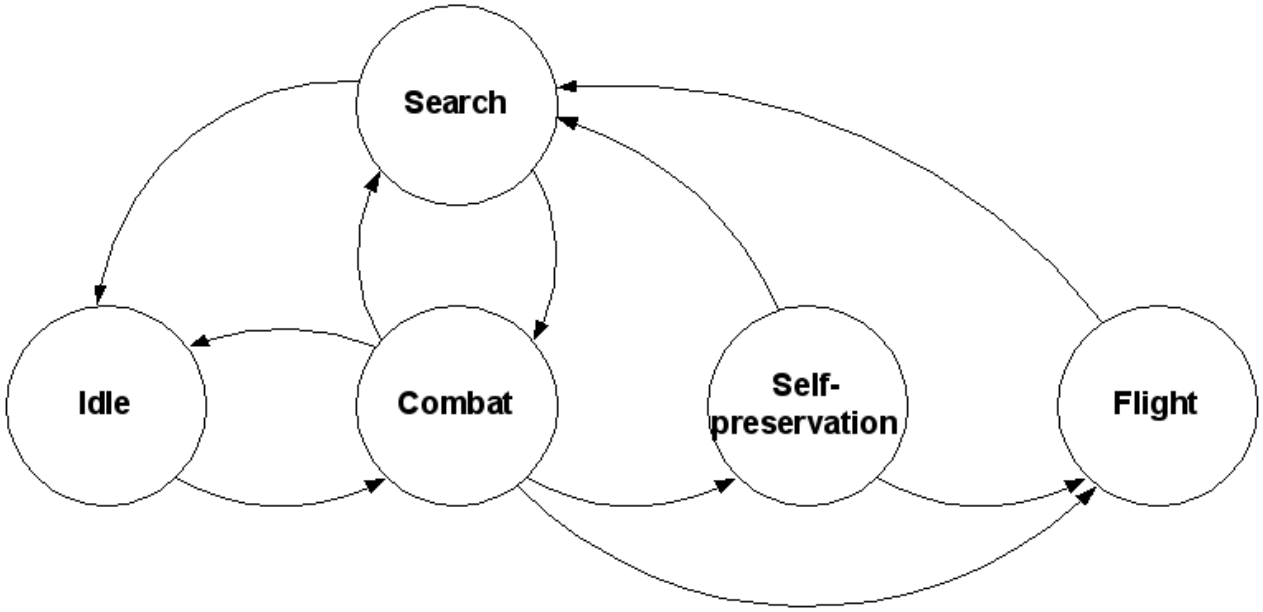


- Transiciones de estados
- Diseño = estados, control y transiciones

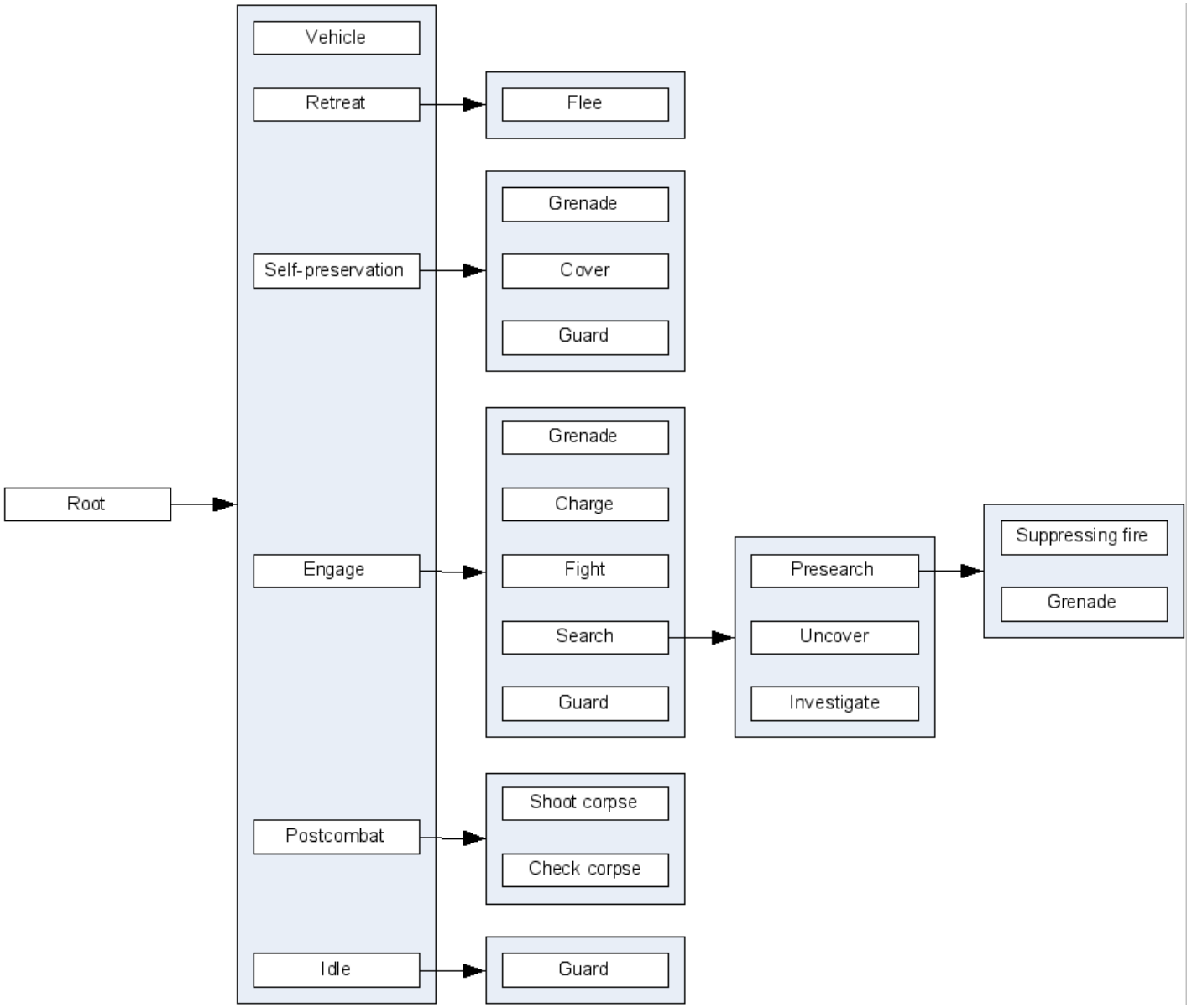
Halo-2

- Comportamiento de los integrantes del juego
- Mundo simulado, percepción simplificada
- Apariencia de inteligencia
- Halo-2, Halo-3

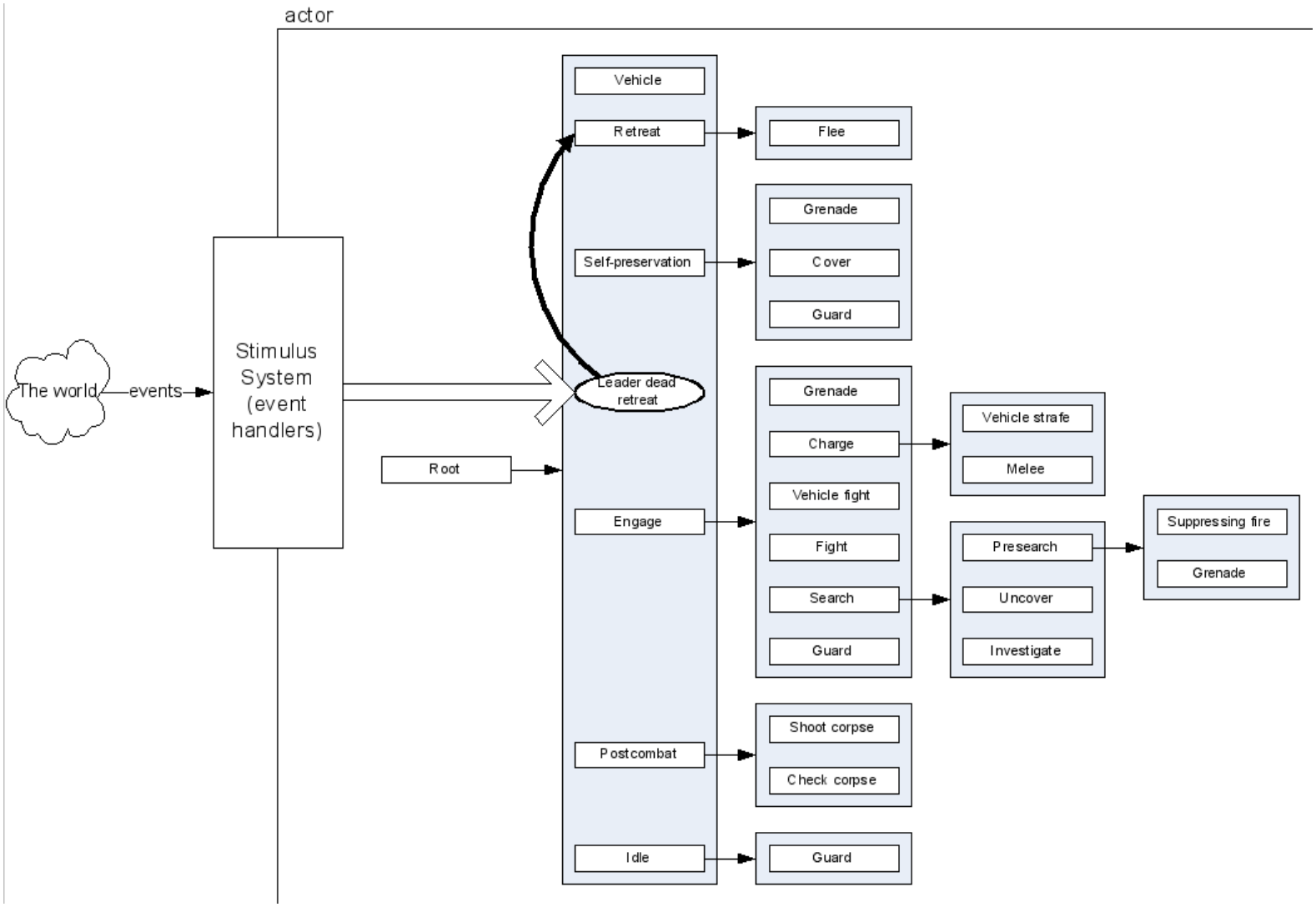
- La cantidad genera complejidad
  - $>$ 100 comportamientos
  - distintos personajes
- 30 Hz
- Variabilidad
- Variación, distintos caracteres
- Direccionabilidad

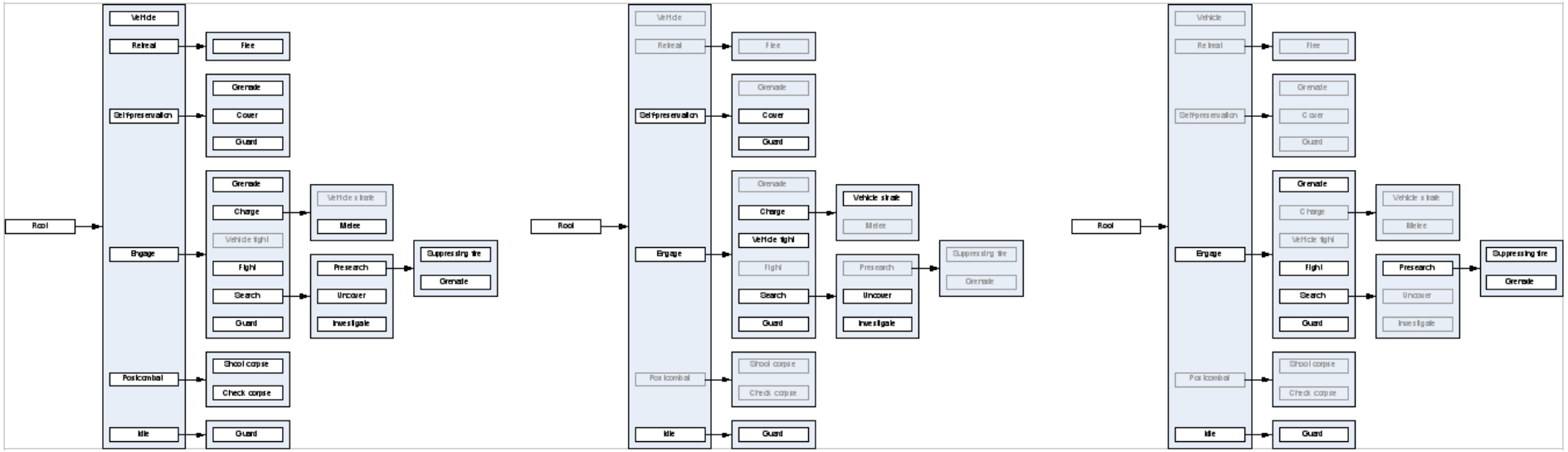


- HFSM árbol de comportamientos
- Precondiciones: comportamientos adecuados a la situación


Vehicle
Retreat
Flee
Grenade
Self-preservation
Cover
Guard
Grenade
Root
Charge
Engage
Fight
Presearch
Suppressing fire
Search
Uncover
Grenade
Guard
Investigate
Shoot corpse
Postcombat
Check corpse
Idle
Guard

- Prioridades, secuencial, probabilístico, etc.
- Ajuste numérico es inviable cuando hay muchos para elegir
- Impulsos: punteros de comportamiento, precondiciones diferentes
- Precondiciones cortas y chequeos completos
- Estímulos disparadores: comportamiento bajo demanda


actor
The world
events
Stimulus System (event handlers)
Root
Vehicle
Retreat
Flee
Self-preservation
Grenade
Cover
Guard
Leader dead retreat
Grenade
Charge
Vehicle fight
Fight
Search
Guard
Vehicle strafe
Melee
Presearch
Uncover
Investigate
Suppressing fire
Grenade
Engage
Postcombat
Shoot corpse
Check corpse
Idle
Guard


Root
Vehicle
Retreat
Flee
Selfpreservation
Grenade
Cover
Guard
Engage
Grenade
Charge
Vehicle fight
Flight
Search
Guard
Vehicle state
Melee
Research
Uncover
Investigate
Suppressing fire
Grenade
Poslcombat
Shoot corpse
Check corpse
Idle
Guard

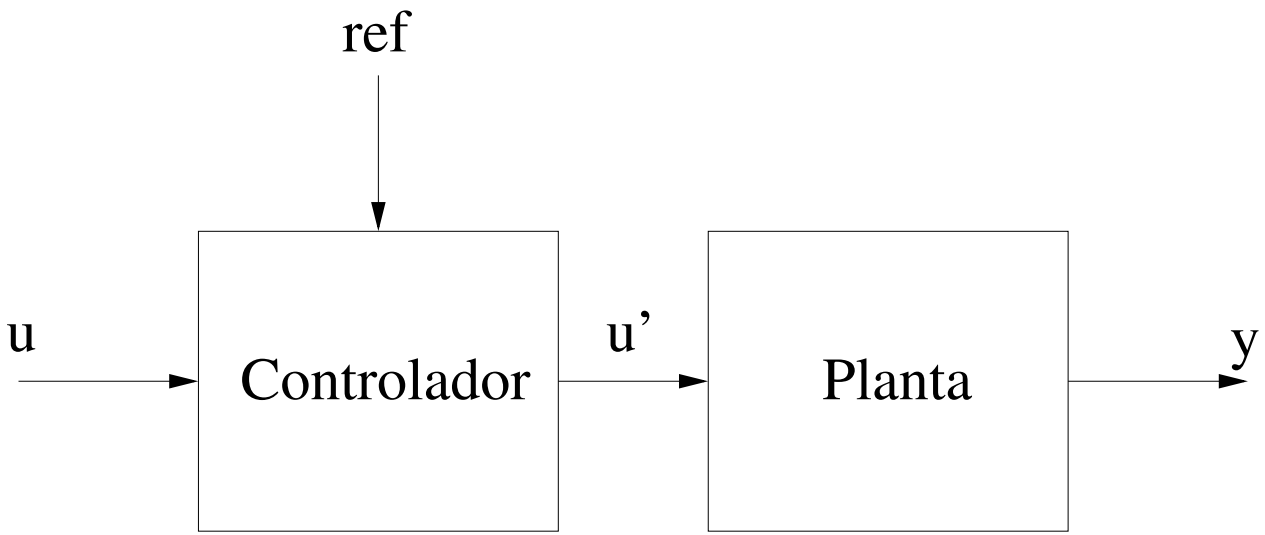
- Árbol genérico
- Subárboles específicos, árboles capados
- Precondiciones, prioridades

- Percepción simplificada
- Memoria: por comportamiento, por objeto...

Teoría de Control clásico

- Controlar un sistema es modificar su comportamiento para que evolucione de una forma determinada
- Planta, sistema a controlar
- Controlador, modifica entradas al sistema
- Aplicaciones industriales, servos,
- Una forma de implementar un comportamiento en un robot es usar controladores clásicos
- Teoría matemática compleja, modelo sistemas dinámicos.
- Es un campo complejo, veremos sólo una pequeña intuición

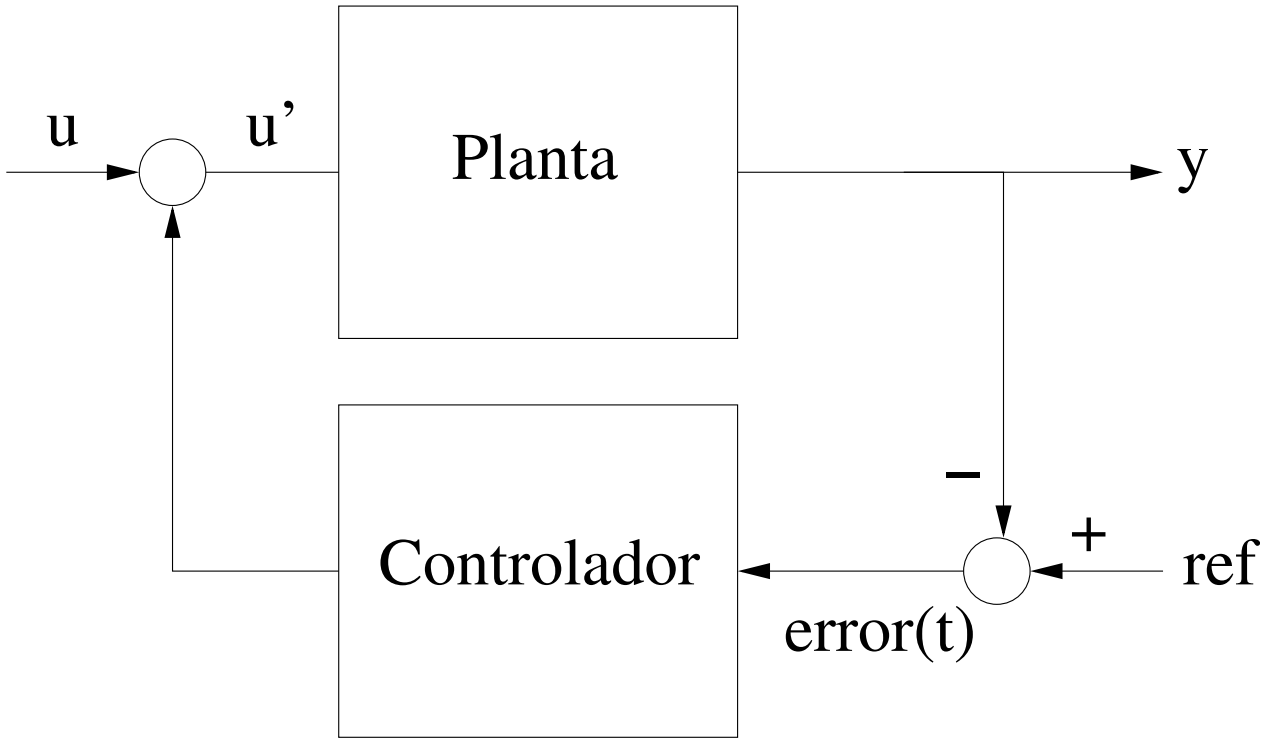
Control de bucle abierto



La actuación se decide "a priori".

- *feedforward*
- El estado no se realimenta en el sistema
- No suelen usar sensores
- Debemos disponer de un modelo muy bueno del sistema
- Teniendo el modelo matematico se puede diseñar el controlador óptimo
- Ejemplo: Ejecución "a ciegas" de un plan
- Ejemplo: Misil balístico, sólo se calcula la velocidad al principio

Control de bucle cerrado



Comparar continuamente el estado deseado y el actual del robot (Realimentación)

- Error: diferencia entre el estado deseado y el actual.
- La meta es minimizar ese error.
- Realimentación, *feedback*
- El error puede ser binario o tener una magnitud y/o dirección.
- El estado deseado puede ser interno o externo.
- Un sistema de bucle cerrado *oscila* alrededor de la solución.
- El controlador tiene como entrada el estado del sistema y el error
- Tres controladores realimentados básicos:
  1. Control Proporcional (P)
  2. Control Derivativo (D)
  3. Control Integral (I)

Controlador realimentado básico

if $e < -\epsilon$ then $u := on$

if $e > \epsilon$ then $u := off$

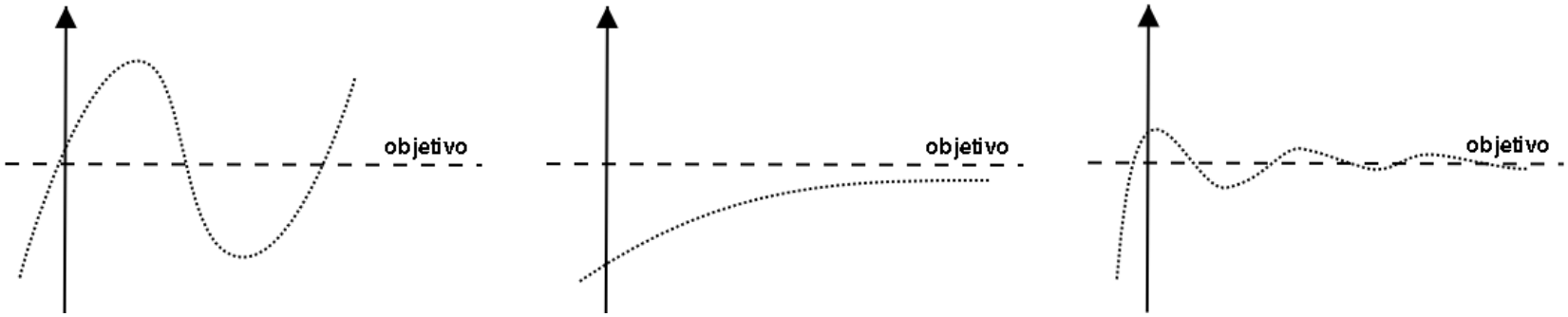
- ϵ evita que el controlador salte alrededor de x_{meta} tan rápido como pueda cuando $x_{meta} \sim x$
- Ejemplo: Termostato de una habitación
- Realmente el termostato es más complejo, porque la variable sobre la que se actúa (temperatura de la caldera) no es la misma que se mide (temperatura de la habitación)
- Problema: Nunca llega a x_{meta}, oscila alrededor
- Ventaja: es muy sencillo, elije entre acciones constantes según el signo del error

Controlador Proporcional (P)

$$u = -K_p e + u_b$$

- Responde en proporción al error: $e = x - x_{meta}$
- Determinar la ganancia correcta necesita una fase experimental
- En entornos bien definidos (modelo) puede calcularse
- Incluso en esos casos las limitaciones físicas (rozamientos, capacidad del motor, etc.) obligan a la experimentación.
- Caso de un robot móvil: ¿Cuánta **ganancia** (K_p) hay que aumentar el ángulo de giro para evitar chocar con una pared?

Ganancia en controladores proporcionales



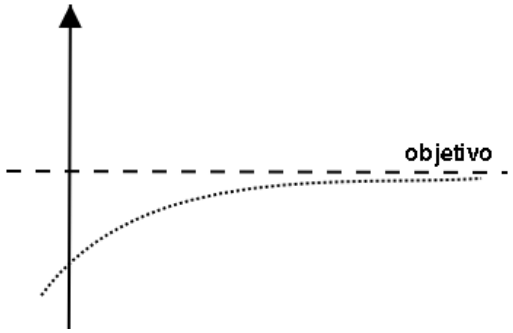
- La figura de la izquierda tiene una ganancia demasiado alta: se oscila alrededor del objetivo
- La figura central tiene una ganancia baja: no se alcanza el objetivo
- La figura de la derecha tiene una ganancia proporcional ajustada

Controladores D

$$u = K_d * de/dt$$

- e error, u salida control, K_d constante proporcionalidad
- Intuición: cuando el error está disminuyendo se debe controlar de forma distinta que cuando está creciendo.
- La salida es proporcional a la derivada de la entrada.
- Cuando el sistema se acerca al estado, restamos una cantidad proporcional a la velocidad con la que decrece el error.
- A ese término se le llama *derivativo*.

Controladores PD



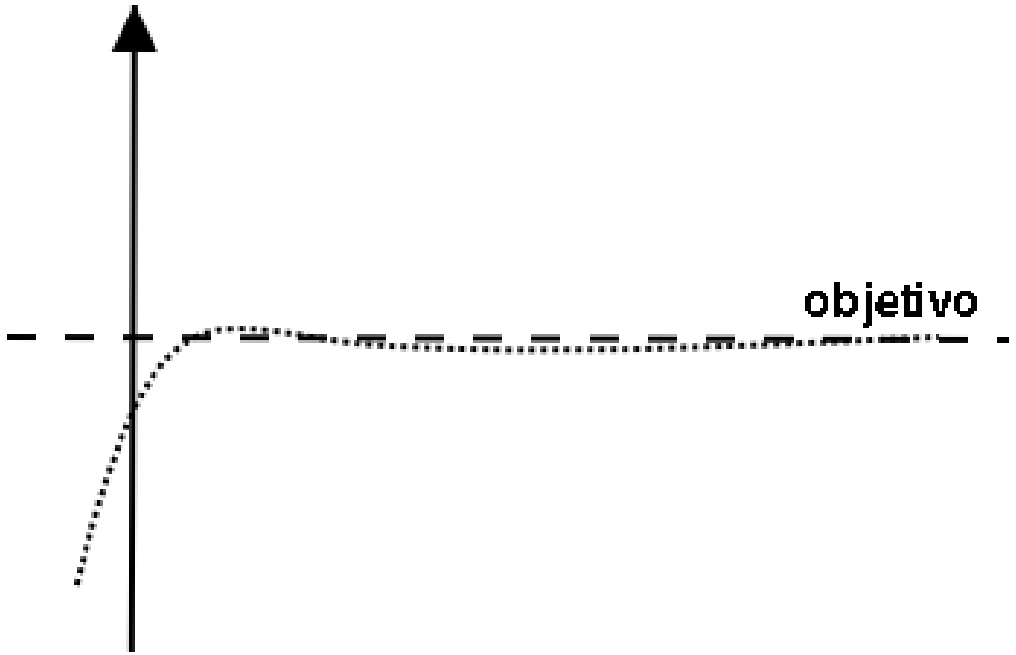
- PD: $u = K_p * e + K_d * de/dt$
- El controlador P tiene tendencia a sobrecorregir
- La idea básica de la componente derivativa es ofrecer resistencia a cambios muy bruscos: corrige en dirección contraria
- El controlador PD tarda más en alcanzar x_{meta} pero oscila menos
- El problema principal es que estimar el termino derivativo del sensor es muy vulnerable al ruido en la lectura de sensores

Controladores I

$$u = K_i * \int e(t)dt$$

- e error, o salida control, K_i constante proporcionalidad
- Añadir el término *integral* o I.
- Intuición: el sistema observa errores, los integra en el tiempo y cuando alcanzan un umbral corrige. Elimina offsets.
- Ejemplo: la K_p del termostato está calculada para la ventana cerrada ¿qué pasa si alguien abre la ventana?

Controladores PID



- La combinación de los 3 anteriores: $u = -K_p e - K_i \int_0^t e dt - K_d de/dt$
- Ajustando bien las tres constantes (K_p, K_i, K_d) se puede obtener una respuesta prácticamente perfecta
- No necesita modelos de la planta, sólo ajustar sus constantes

Control PID en robots

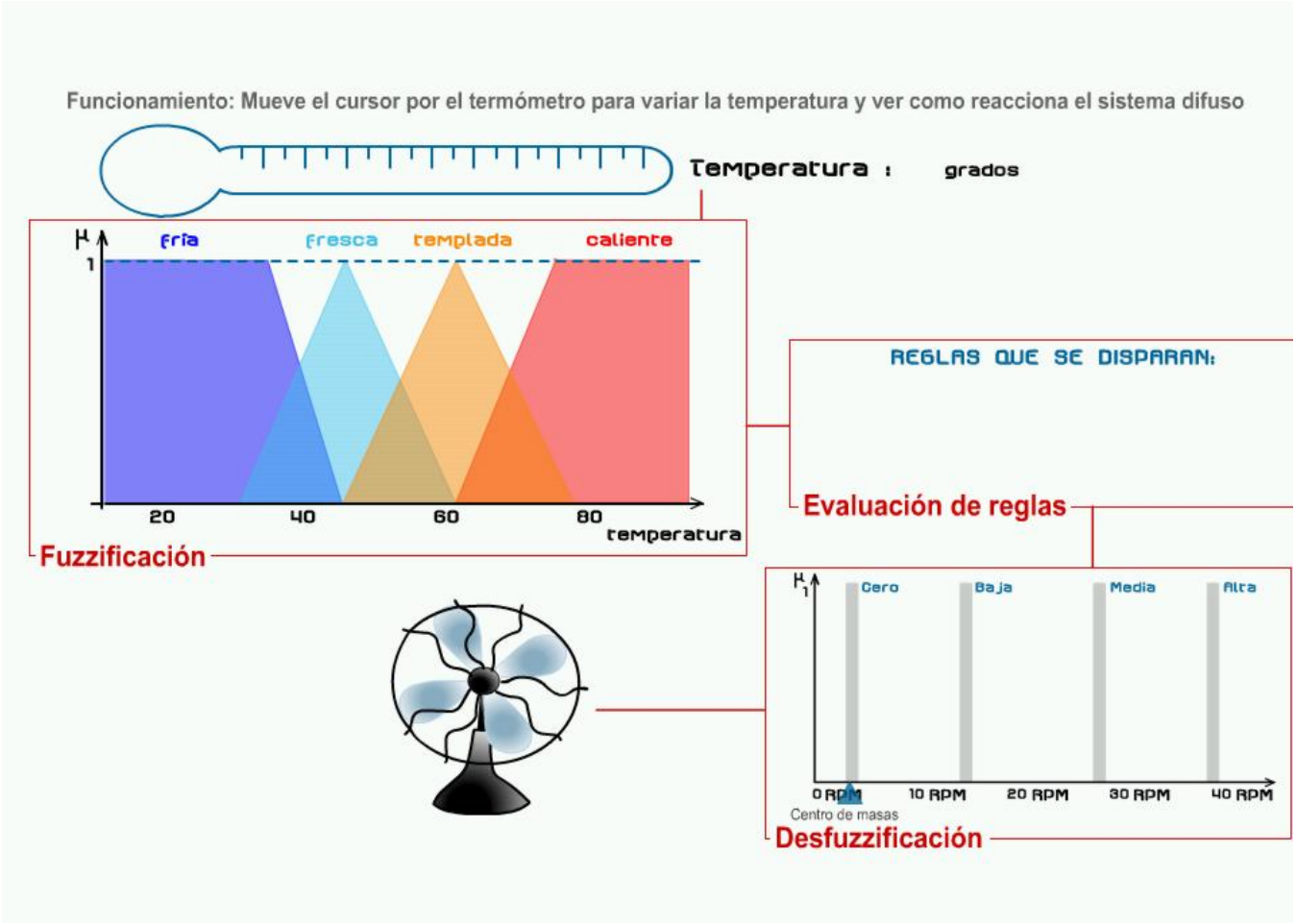
- Péndulo invertido con el LEGO-NXT
- Usando la luz medida como referencia y señal
- Control de bajo nivel de motores del Pioneer
- Velocidad como referencia, perfiles de velocidad

Limitaciones del control

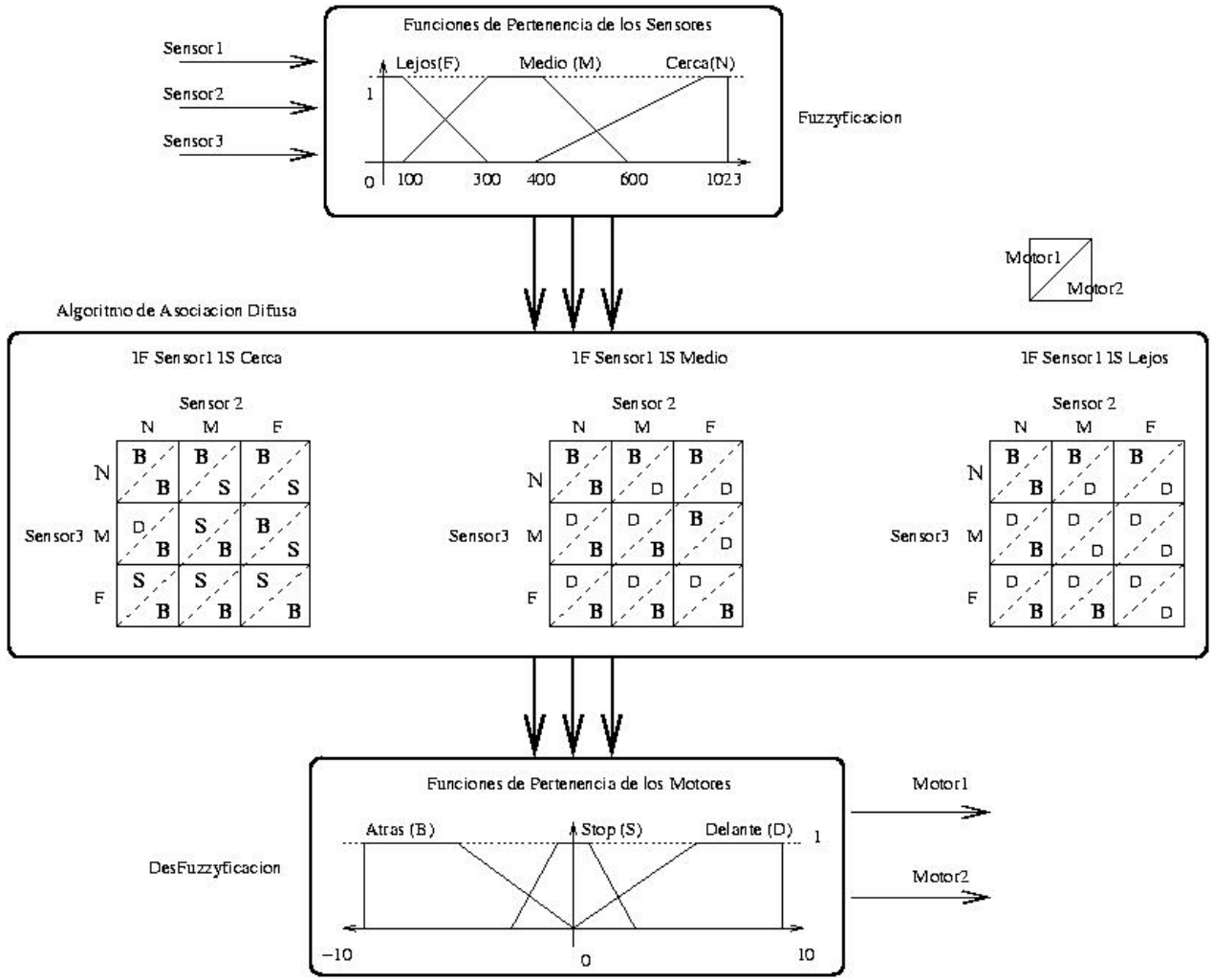
- Necesita modelo del sistema para ser preciso
- No escala a comportamientos complejos o con muchos actuadores
- Percepción limitada, se la tienen que dar resuelta

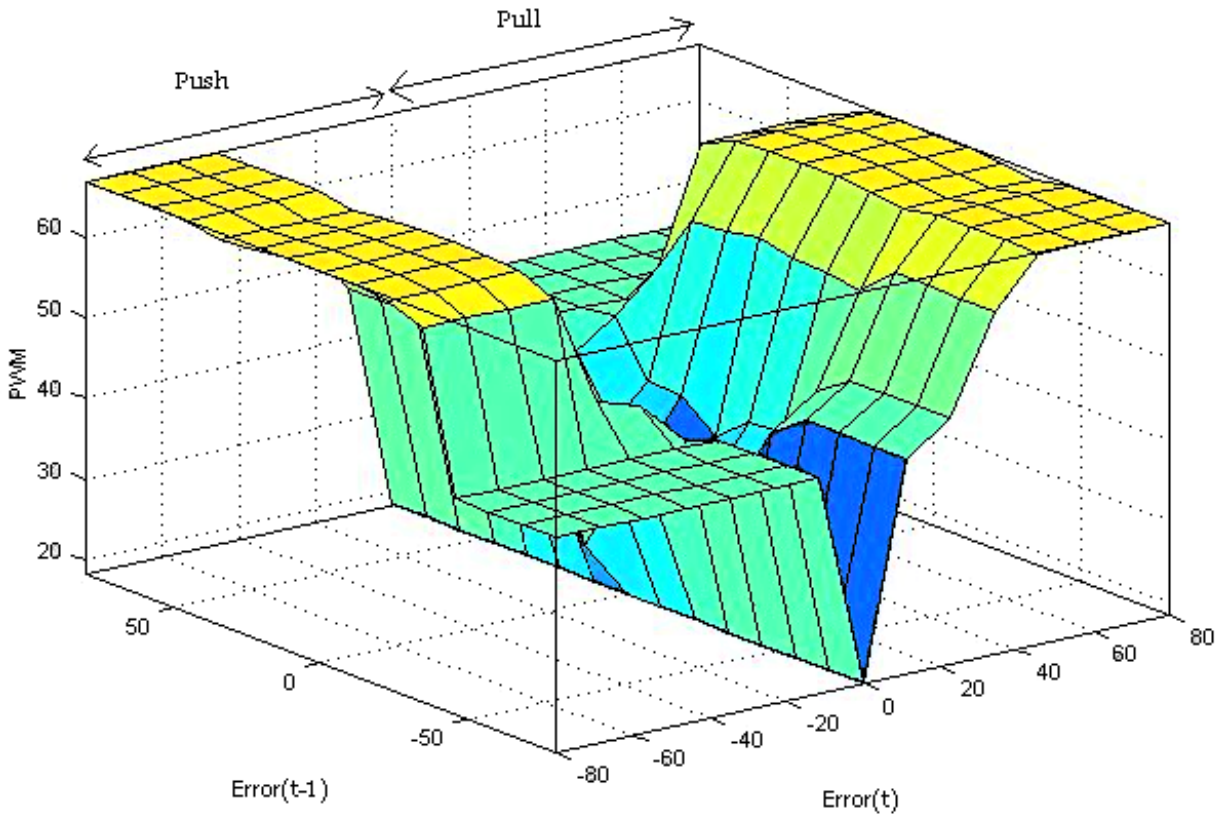
Sistema de Reglas Borroso

- Reglas lingüísticas
- Etiquetas lingüísticas sobre variables borrosas
- Valores de verdad graduales
- Fáciles de desarrollar


Funcionamiento: Mueve el cursor por el termómetro para variar la temperatura y ver como reacciona el sistema difuso
Temperatura : grados
μ
1
fría
fresca
templada
caliente
20
40
60
80
temperatura
Fuzzificación
REGLAS QUE SE DISPARAN:
Evaluación de reglas
μ
1
Cero
Baja
Media
Alta
0 RPM
10 RPM
20 RPM
30 RPM
40 RPM
Centro de masas
Desfuzzificación

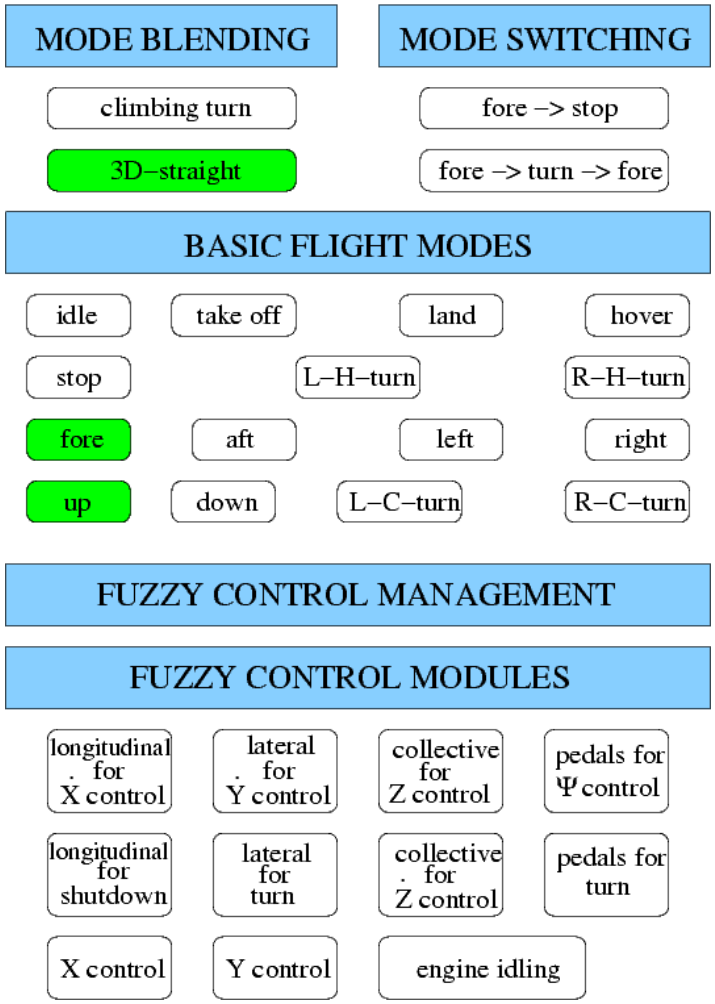
Ejemplo: Evitar obstáculos



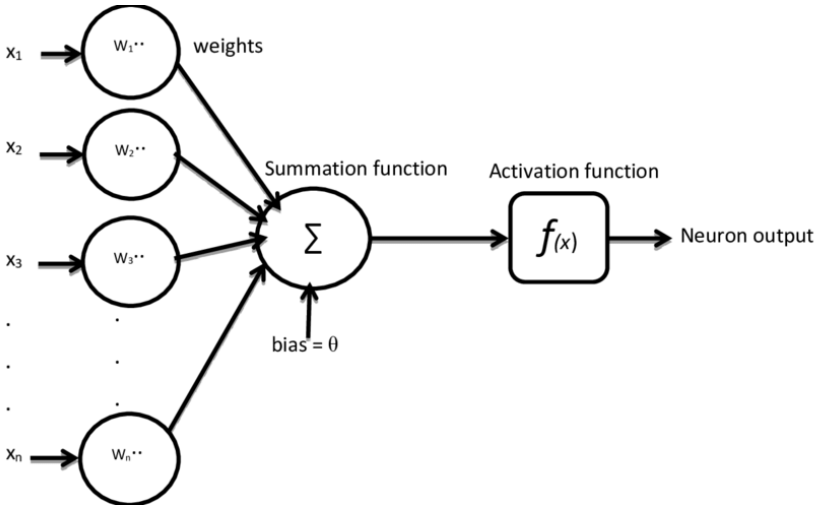


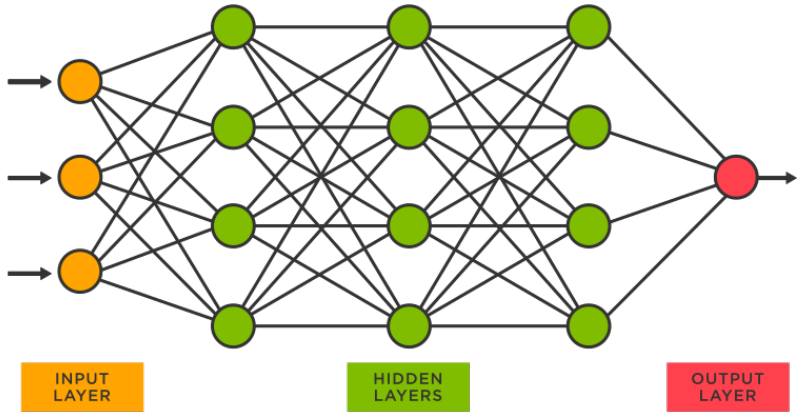
- Superficie de control por parches

Ejemplo

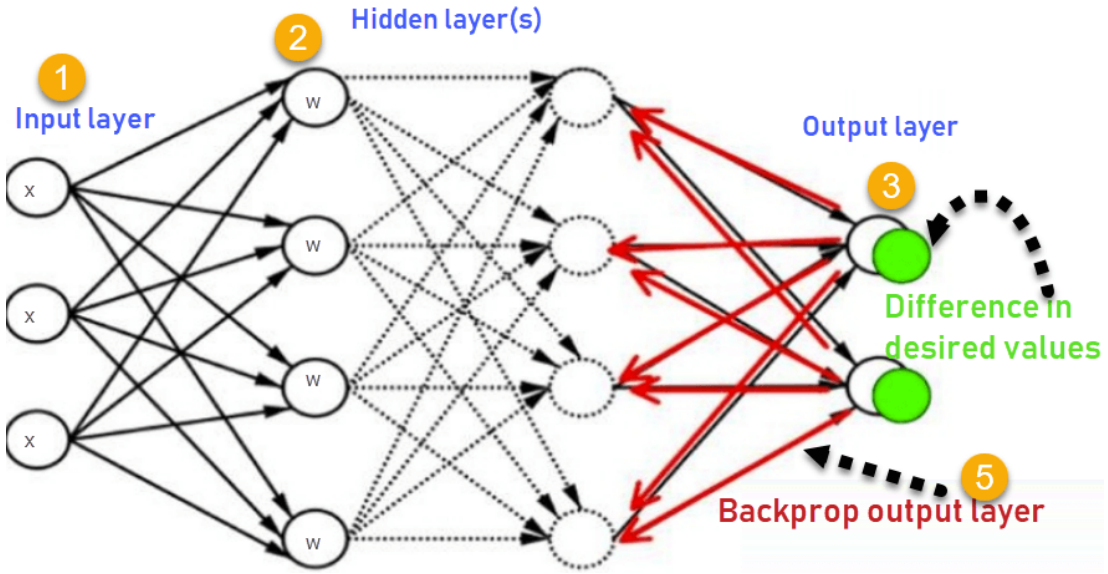

MODE BLENDING
MODE SWITCHING
climbing turn
fore –> stop
3D–straight
fore –> turn –> fore
BASIC FLIGHT MODES
idle
take off
land
hover
stop
L–H–turn
R–H–turn
fore
aft
left
right
up
down
L–C–turn
R–C–turn
FUZZY CONTROL MANAGEMENT
FUZZY CONTROL MODULES
longitudinal . for X control
lateral . for Y control
collective for Z control
pedals for Ψ control
longitudinal for shutdown
lateral for turn
collective . for Z control
pedals for turn
X control
Y control
engine idling

Aprendizaje neuronal, aprendizaje profundo



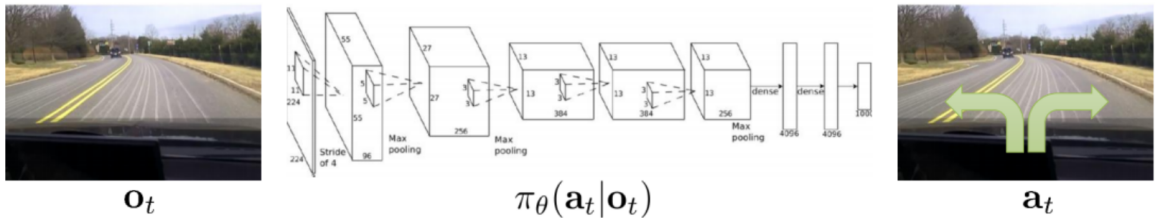


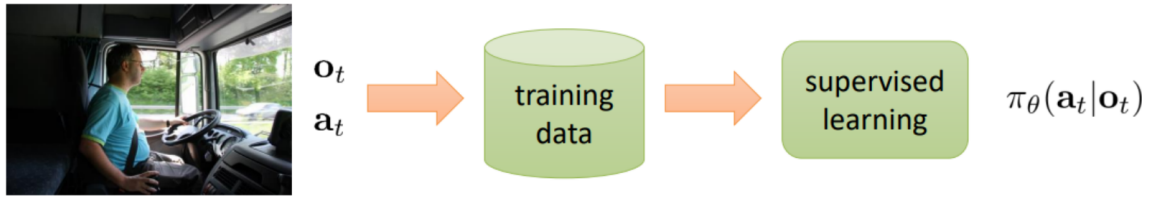
- Modelo de cómputo que se puede ajustar con datos
- 1 neurona, sumatorio ponderado, función de activación
- Red de neuronas conectadas según cierta topología (modelo) y con unos pesos
- Topologías: capas, fully connected...



- Entrenamiento = ajuste de pesos con (muchos) datos supervisados
- Algoritmo aprendizaje: *backpropagation*
- Aproximadores universales de funciones
- Middlewares neuronales: TensorFlow (Google), PyTorch (Meta)

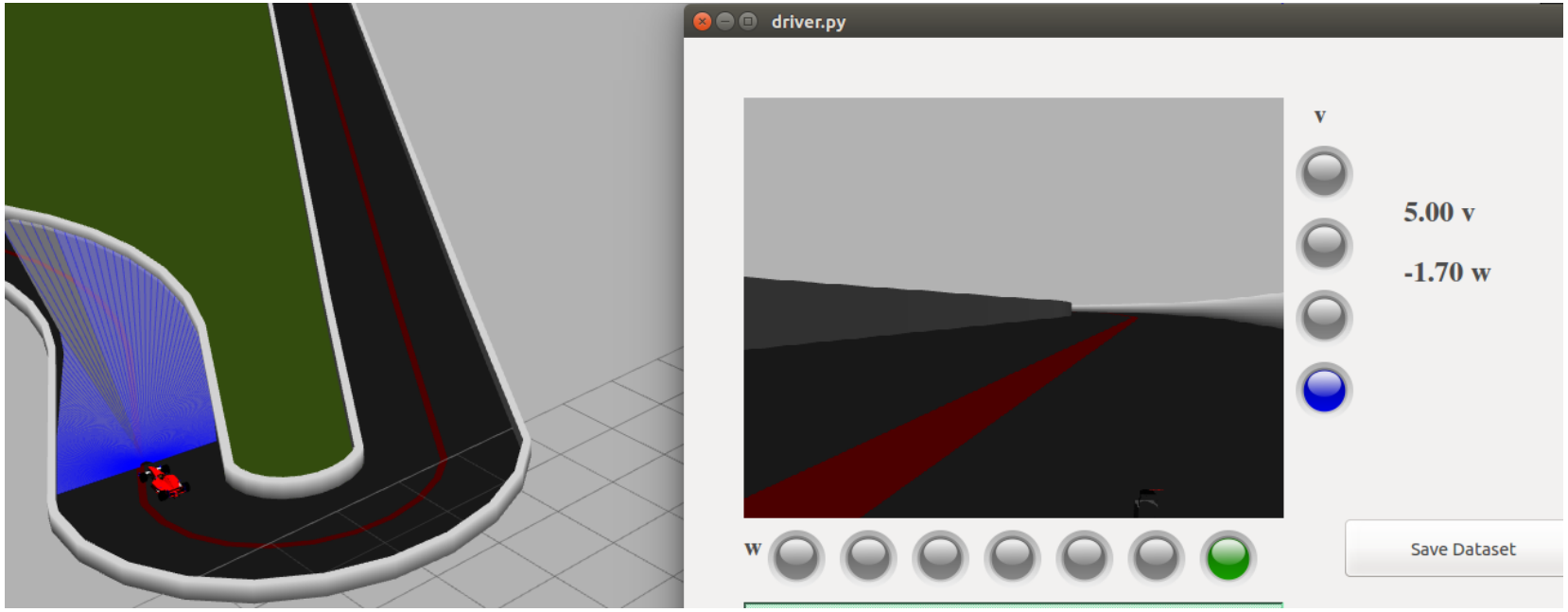
Redes neuronales extremo a extremo



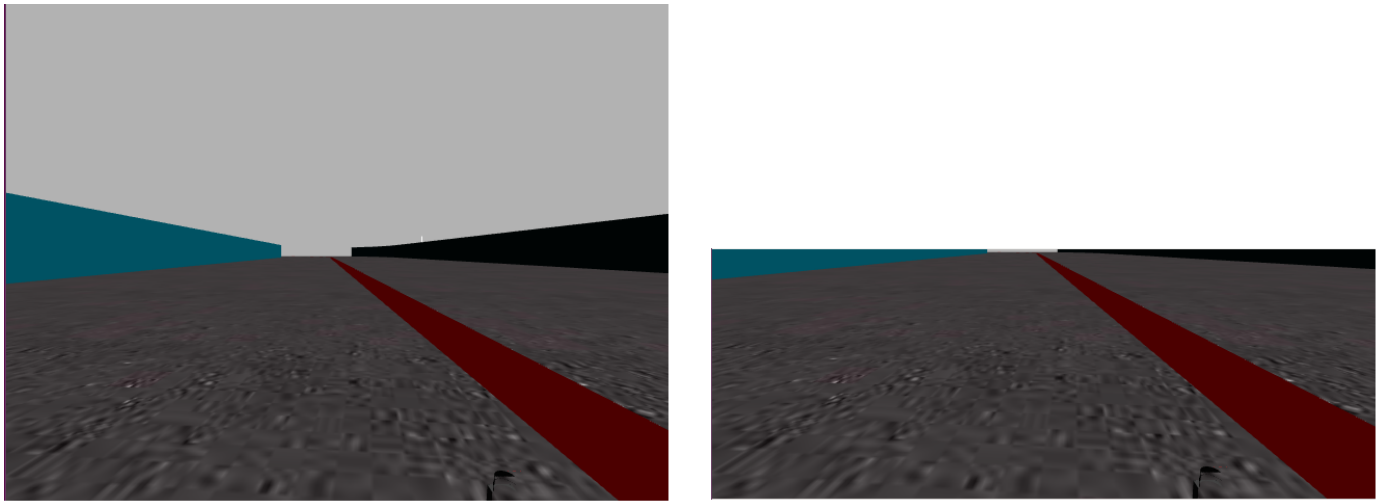


- En sistemas extremo a extremo, incluye percepción y decisiones
- Modelos y datos
- Conducción autónoma: PilotNet (nVidia)...
- Un solo fallo en el control puede ser dramático
- Frecuencia alta mejora la calidad del comportamiento generado

Coche autónomo sigue línea



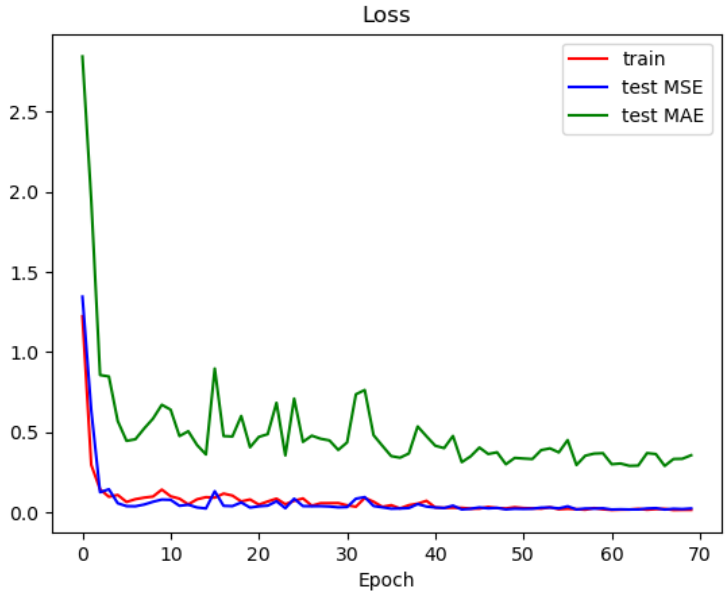
- Gazebo, Formula1, línea roja
- Cámara a entrada de la red. Salida a los motores
- Dataset generado con piloto programado explícitamente o manual
- Aprende controlador reactivo (*visual!*) desde ejemplos supervisados

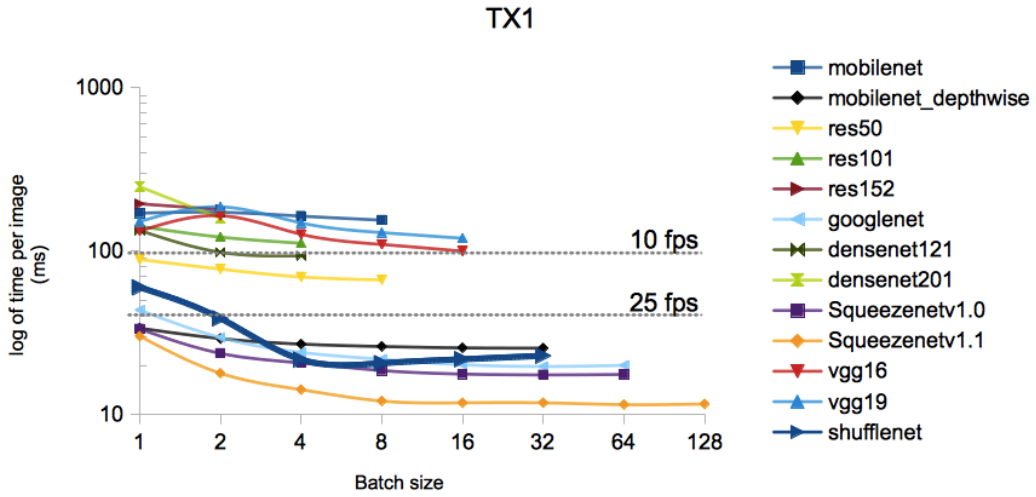


- A la primera no daba bien las curvas
- Casos difíciles aportan mucho al entrenamiento pero son poco frecuentes $\rightarrow$ aumentarlos artificialmente, dataset balanceado
- Red de clasificación: discretización de V y W
- ¿Cuántos niveles de discretización?
- ¿Imágenes recortadas? No le distraen
- Red de regresión, salidas lineales

Coche autónomo real sigue línea

- JetBot, Cámara y motores
- Red preentrenada se reentrena
- Distintas iluminaciones, circuitos...
- ¡Funciona!





- ¿Cuándo detener el aprendizaje?
- Evolución de la función de pérdida
- Tiempo de inferencia es crítico
- Hay unas redes más rápidas que otras