



UNIVERSIDAD  
DE GRANADA



# Técnicas de los Sistemas Inteligentes

## Grado en Informática

### Curso 2019-20. Seminario 1

### IA en Robótica y Videojuegos

**Jesús Giráldez Crú, Pablo Mesejo Santiago y José Ángel Segura Muros**

{jgiralde, pmesejo, josesegmur}@decsai.ugr.es

**Departamento de Ciencias de la  
Computación e Inteligencia Artificial**

**<http://decsai.ugr.es>**

Basado en los materiales previos elaborados por Juan Fernández Olivares

TEORÍA (50%) + PRÁCTICA (50%)

Entrega de 3 prácticas:

- **P1: Búsqueda Heurística** (30%)
- **P2: Satisfacción de Restricciones (CSP)** (30%)
- **P3: Planificación Clásica (PDDL)** (30%)

10% restante: asistencia a través de PRADO

(<https://pradogrado1920.ugr.es/>) o por medio de hoja de firmas

Para aprobar se necesita un cinco en total (teoría + práctica), con un mínimo de 3 puntos en cada parte.

**TUTORÍAS DE PRÁCTICAS:** poneos en contacto por email con los profesores de prácticas

Jesús Giráldez: [igiraldez@ugr.es](mailto:igiraldez@ugr.es)

Pablo Mesejo: [pablomesejo@gmail.com](mailto:pablomesejo@gmail.com)

José Ángel Segura: [josesegmur@decsai.ugr.es](mailto:josesegmur@decsai.ugr.es)

## TEMPORIZACIÓN PRÁCTICAS

- *Sesión 1: Semana 10 de Febrero*
  - Seminario1. Introducción IA en Robótica y Videojuegos
- *Sesión 2: Semana 17 de Febrero*
  - Presentación Práctica 1 (**búsqueda heurística**)
- *Sesión 3: Semana 24 de Febrero (no hay clase el viernes!)*
  - Seguimiento/Dudas
- *Sesión 4: Semana 2 de Marzo*
  - Seguimiento/Dudas
- *Sesión 5: Semana 9 de Marzo*
  - Seguimiento/Dudas
- *Sesión 6: Semana 16 de Marzo*
  - Seguimiento/Dudas
- *Sesión 7: Semana 23 de Marzo*
  - Seguimiento/Dudas
- **Entrega P1: 29 de Marzo de 2020 hasta las 14:00**

## TEMPORIZACIÓN PRÁCTICAS

- *Sesión 8: Semana 30 de Marzo*
    - Seminario de MiniZinc (**CSP**) + Presentación P2
  - *SEMANA SANTA*
  - *Sesión 9: Semana 13 de Abril (no hay clase el lunes!)*
    - Seguimiento/Dudas
  - *Sesión 10: Semana 20 de Abril*
    - Seguimiento/Dudas
  - *Sesión 11: Semana 27 de Abril (no hay clase el viernes!)*
    - Seguimiento/Dudas
  - **Entrega P2:** 3 de Mayo de 2020 a las 14:00
- 
- *Sesión 12: Semana 4 de Mayo*
    - Seminario de **planificación clásica (PDDL)** + Presentación P3
  - *Sesión 13: Semana 11 de Mayo*
    - Seguimiento/Dudas
  - *Sesión 14: Semana 18 de Mayo*
    - Seguimiento/Dudas
  - *Sesión 15: Semana 25 de Mayo (última clase: Mié. 27 de Mayo)*
    - Seguimiento/Dudas
  - **Entrega P3:** 31 de Mayo de 2020 hasta las 14:00

- ¿Qué es un agente?

Un agente es **algo que razona**, capaz de percibir su entorno y actuar en él.

□ Un agente racional es aquel que actúa con la intención de alcanzar el mejor resultado o, cuando hay incertidumbre, el mejor resultado esperado.

- ¿Qué es un robot?

**Agente físico**, capaz de **percibir** su entorno, **tomar decisiones** en base a los datos adquiridos, e **interactuar** con el mundo físico (es decir, **llevar a cabo acciones** en el mundo real).



## • Sensor

### – Dispositivo para percibir/medir el entorno.

- Cámara/láser/infrarrojos/kinect para medir el entorno
- Giróscopos o acelerómetros para medir el movimiento del propio del robot
- Micrófonos para percibir sonidos

## • Efecto

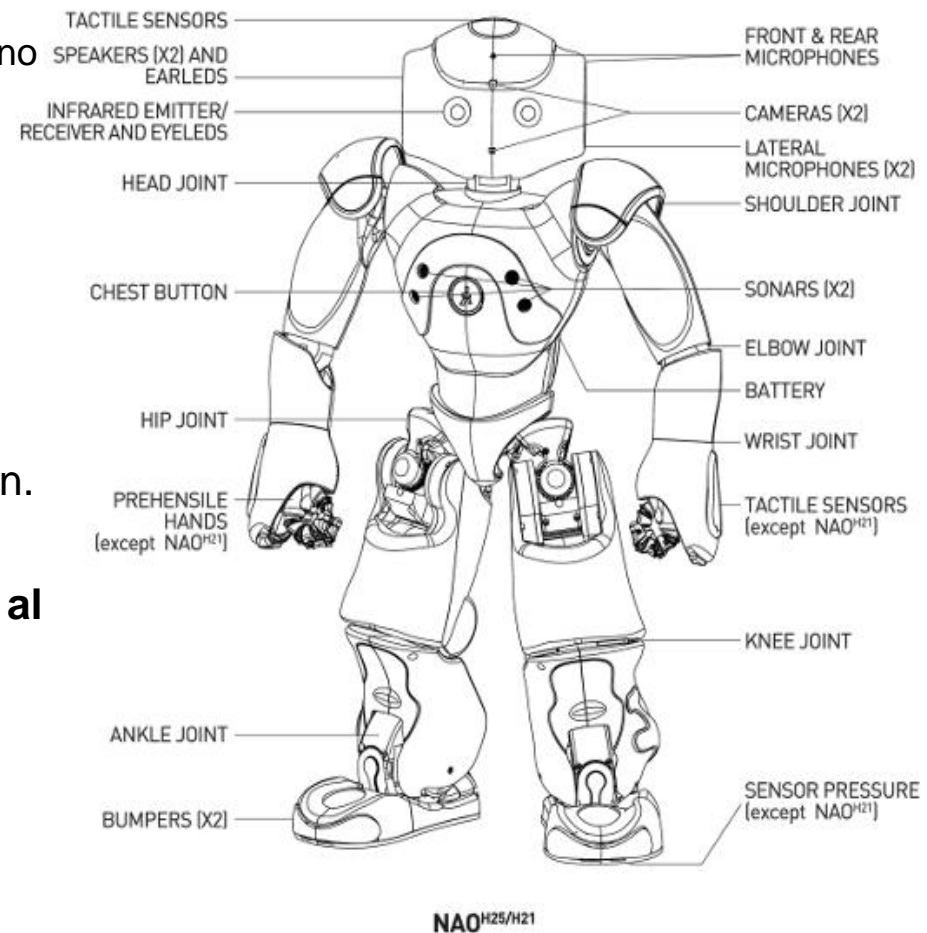
### – Dispositivo para manipular o modificar su entorno

- Piernas, ruedas, articulaciones, pinzas,...
- Propósitos básicos: locomoción y manipulación.

## • Actuador

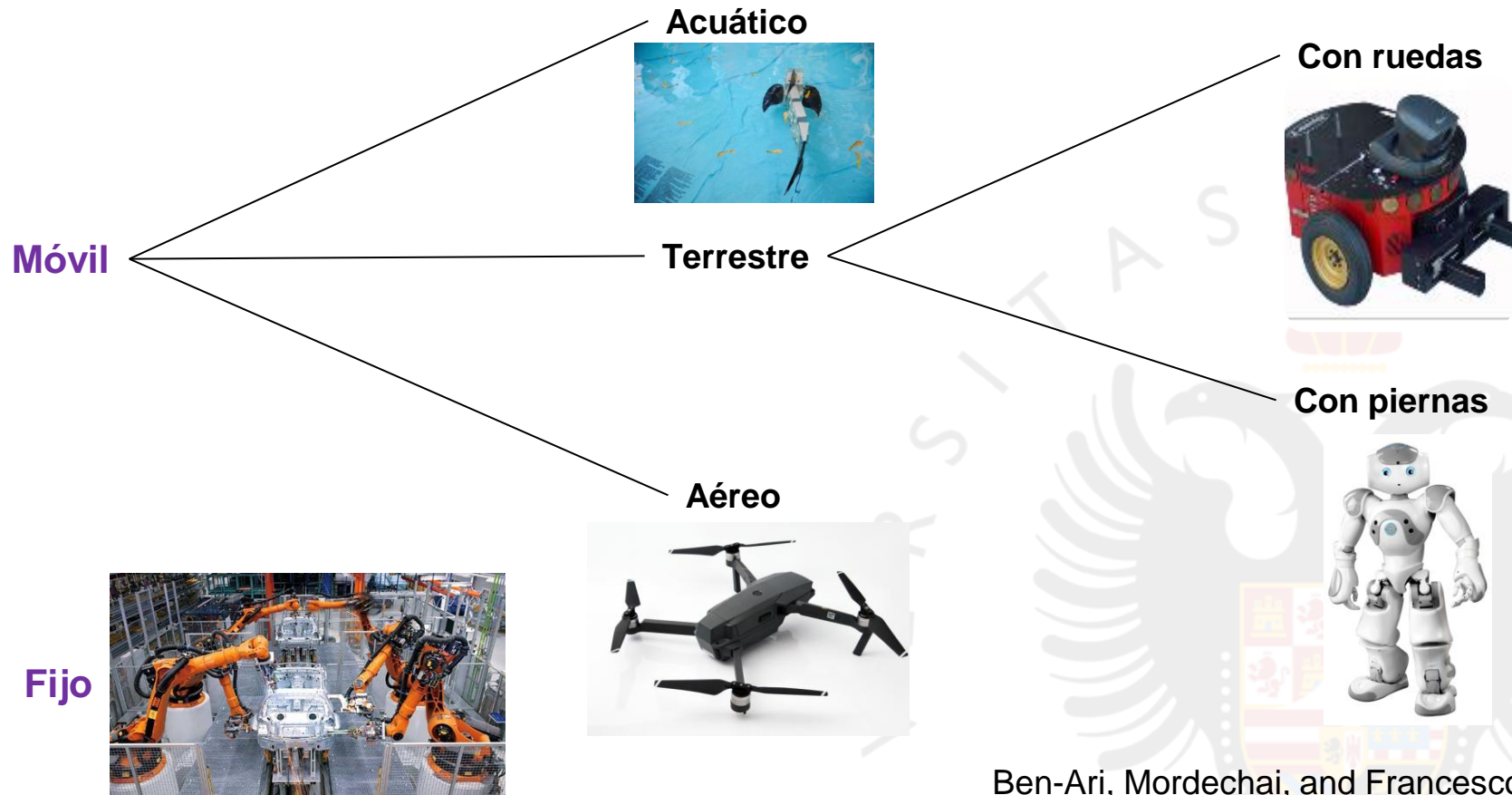
### – Distinto del efecto: **mecanismo que permite al efecto ejecutar una acción (convierte energía en movimiento).**

- Motores eléctricos, cilindros neumáticos o hidráulicos, etc.





## SI NOS BASAMOS EN EL ENTORNO Y EL MECANISMO DE INTERACCIÓN:



Principalmente, manipuladores robóticos industriales

Ben-Ari, Mordechai, and Francesco Mondada. "Robots and their applications." *Elements of Robotics*. Springer, Cham, 2018. 1-20.

## SI NOS BASAMOS EN EL CAMPO DE APLICACIÓN:

**Servicios**

**Médico**



**Robots sociales**

**Educacional**



**Defensa/Seguridad**



**Hogar**



**Industrial (logística, manufactura)**



**Y muchas más aplicaciones...**

- Transporte (ej. Helpmate)
- Exploración espacial
- Minería
- ...





Shakey The Robot  
(1966-1972).

DARPA project desarrollado en SRI International. Primer robot móvil de propósito general. Resultados de este proyecto: A\* search algorithm, STRIPS planner, y Hough transform, entre otros.

- Percepción
- Movimiento y Control
- Localización (que el robot conozca su propia posición en relación a su entorno)
- Mapeo (construir y/o usar un mapa del entorno en donde el robot se pueda localizar)
- Navegación
  - Comportamiento reactivo: ciclo percepción-acción (estímulo/respuesta); el agente reacciona a la evolución del entorno
  - Comportamiento deliberativo: El proceso del agente introduce una función deliberativa entre la percepción y la ejecución para elegir la acción correcta
- Planificación de tareas

- Bibliografía Esencial:
  - **Capítulo 25 del libro** (titulado Robotics). *Artificial Intelligence: A Modern Approach*. Stuart Russell and Peter Norvig. Publicado por primera vez en 1995. 3era edición de 2009. Prentice Hall.
- Bibliografía Complementaria:
  - Robin R. Murphy, Introduction to AI Robotics, The MIT Press 2000
  - Fernández, Enrique, Luis Sánchez Crespo, Anil Mahtani, and Aaron Martinez. *Learning ROS for Robotics Programming - Second Edition*. 2nd ed. Packt Publishing, 2015.
  - Goebel, R. Patrick. *ROS By Example INDIGO - Volume 1*, 2015
  - Wiki de ROS Documentation <http://www.ros.org/>

**Nota:** ROS (Robot Operating System). Sistema operativo para robots de código abierto (<http://wiki.ros.org/>, <http://www.ros.org>) cuyo objetivo fundamental es soportar la reutilización de código en el desarrollo e investigación sobre robótica. Originalmente desarrollado en 2007 en el Stanford Artificial Intelligence Laboratory (<https://ai.stanford.edu/>)

El **objetivo** de este seminario (y de la Práctica 1) no es conocer ~~técnicas de Desarrollo de Videojuegos~~, sino conocer **técnicas de IA en Videojuegos**.

Videojuegos → robótica en simulación

- En la asignatura de IA de 2º visteis algoritmos de IA aplicados a juegos de tablero (bipersonales, por turnos).
- Ahora, nos centraremos en agentes inteligentes aplicados a videojuegos.

Millington, Ian. *AI for Games*. CRC Press, 2019.

Yannakakis, Georgios N., and Julian Togelius. *Artificial intelligence and games*. Vol. 2. New York: Springer, 2018.. <https://doi.org/10.1007/978-3-319-63519-4>.

Perez-Liebana, D., Liu, J., Khalifa, A., Gaina, R. D., Togelius, J., & Lucas, S. M. (2018). *General video game ai: a multi-track framework for evaluating agents, games and content generation algorithms*. *arXiv preprint arXiv:1802.10363*.  
<http://arxiv.org/abs/1802.10363>.

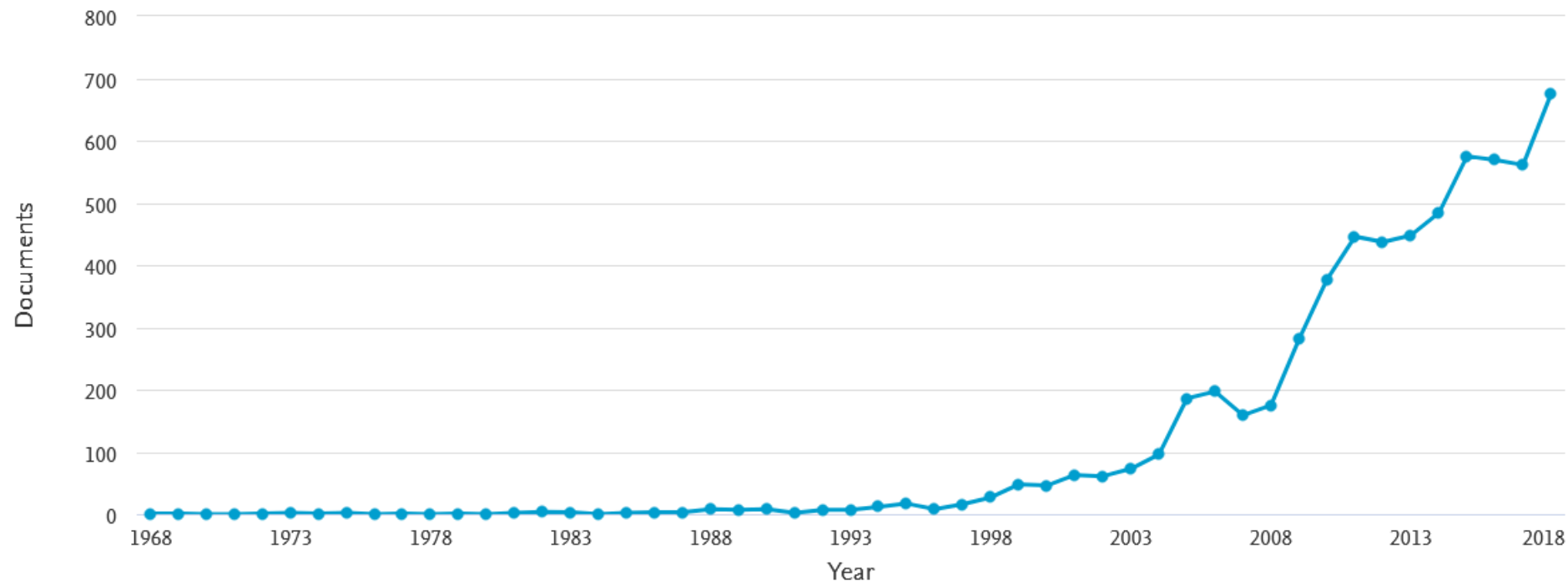
## ARTÍCULOS CIENTÍFICOS (POR AÑO) PUBLICADOS EN ESTE CAMPO (IA en juegos y videojuegos):

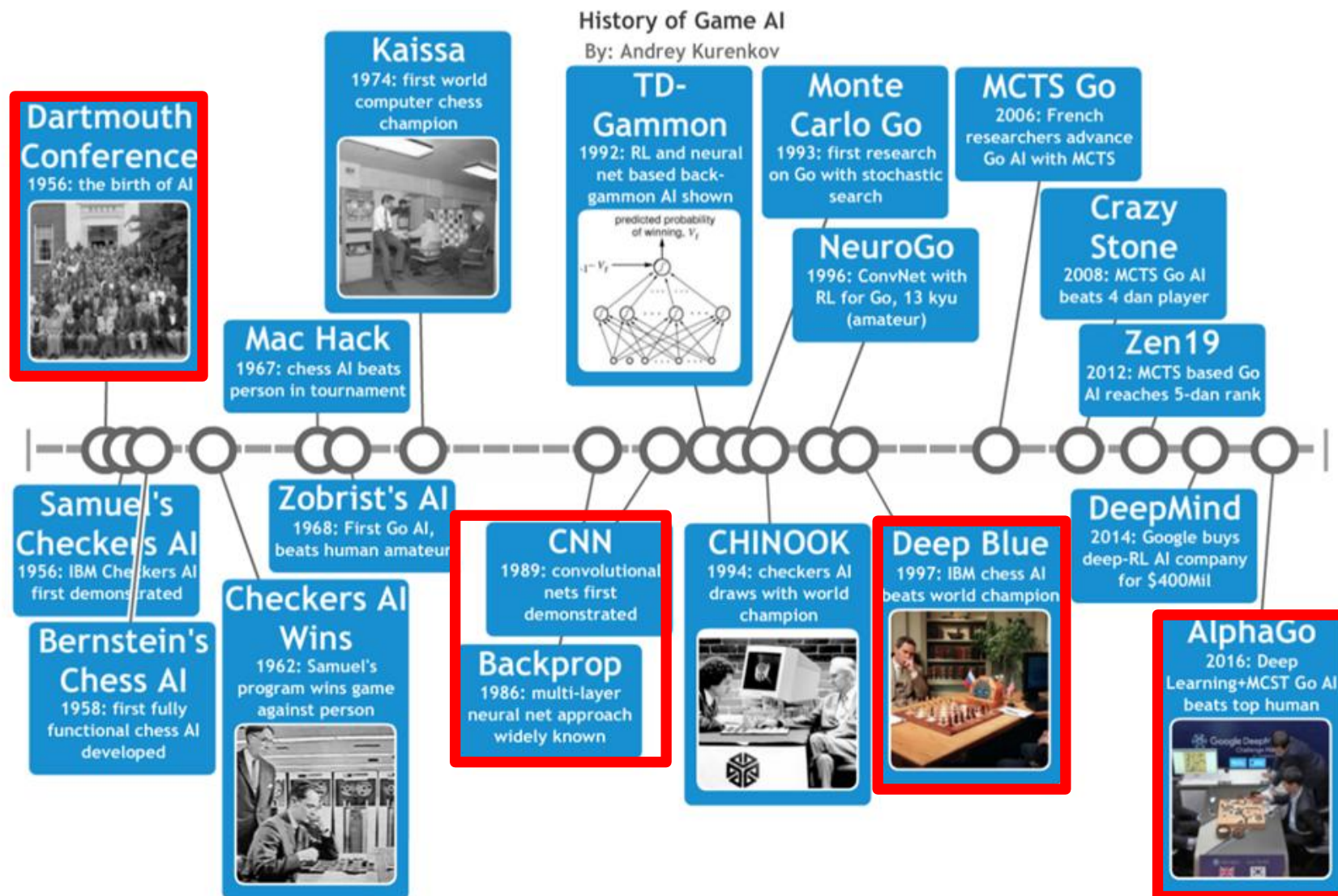
Scopus

TITLE-ABS-KEY ( artificial AND intelligence AND ( videogames OR games ) AND NOT ( game AND theory ) ) AND ( LIMIT-TO ( SUBJAREA , "COMP" ) ) AND ( EXCLUDE ( PUBYEAR , 2020 ) OR EXCLUDE ( PUBYEAR , 2019 ) )

6,103 document results

Documents by year







# 1. **¿Para qué se usa la IA en Videojuegos?**

→ Nos centramos en IA para “Jugar a Videojuegos”

# 2. **Características de los Videojuegos**

→ Para poder definir qué técnica de IA es más apropiada

# 3. **Repaso de Técnicas de IA**

→ De más éxito para Jugar a Videojuegos

1. **¿Para qué se usa la IA en Videojuegos?**

→ Nos centramos en IA para “Jugar a Videojuegos”

2. **Características de los Videojuegos**

→ Para poder definir qué técnica de IA es más apropiada

3. **Repaso de Técnicas de IA**

→ De más éxito para Jugar a Videojuegos

## ¿Para qué se usa IA en Videojuegos?

- Para jugar a videojuegos
- Para generar automáticamente contenidos
- Para analizar datos de juegos y modelar jugadores
- Para desarrollar agentes creíbles (*Believable Agents*)
- Para mejorar la capacidad humana
- Para testear algoritmos de IA

**Nota:** estas categorías no son disjuntas, y presentan cierta complementariedad y solapamiento entre ellas.

## IA para Jugar a Videojuegos

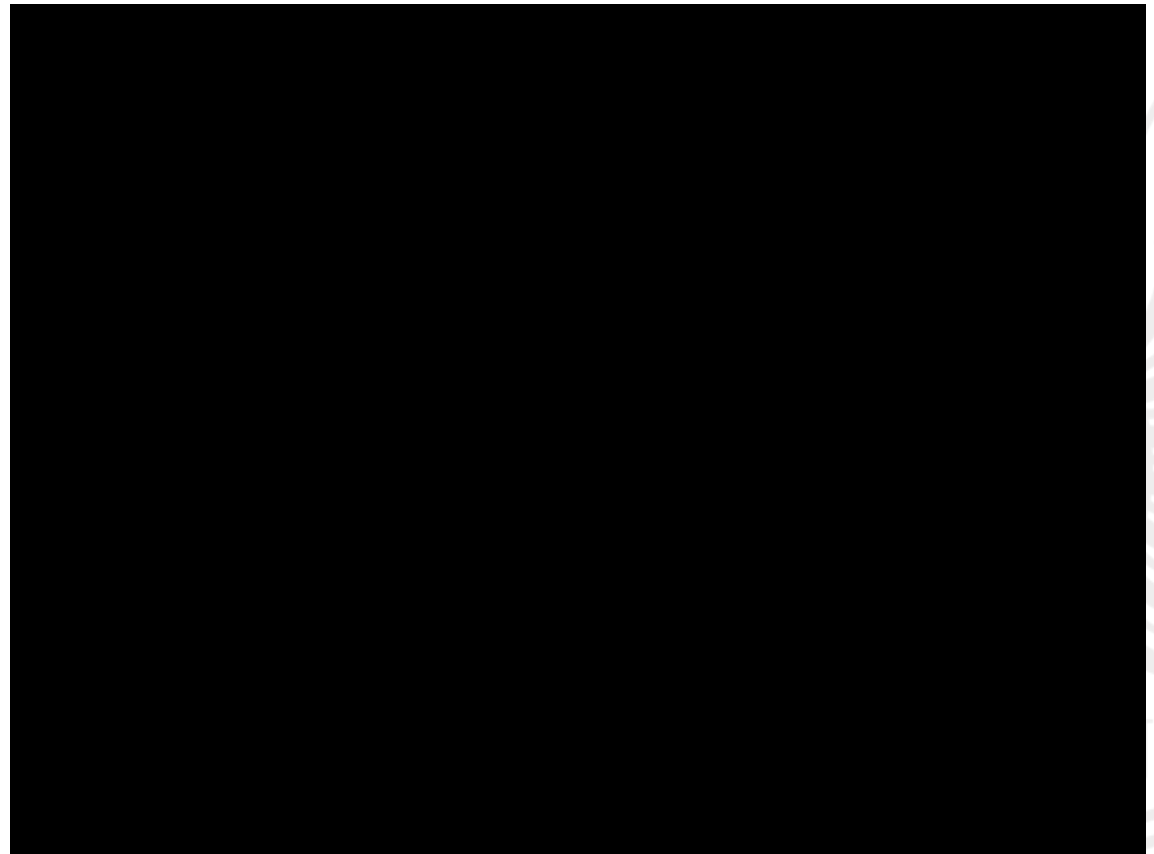
- La IA está integrada en el videojuego
  - Un jugador o un NPC (Non-Player Character) que actúa de modo autónomo.

Cualquier personaje de un juego no controlado por un humano (generalmente, no hostil a los jugadores: se trata de aliados o personajes con actitud neutral)

### Google DeepMind's Deep Q-learning playing Atari

**Breakout.** Mnih, Volodymyr, et al.  
 "Human-level control through deep reinforcement learning." *Nature* 518.7540 (2015): 529.

### Aprendizaje por refuerzo



## IA para Jugar a Videojuegos

- **¿Queremos que la IA juegue a ganar?**
  - Objetivo principal: alcanzar el mayor rendimiento posible (alta puntuación, ganar a un oponente, sobrevivir mucho tiempo, ...)
  - ¿Siempre es posible definir qué significa "alto rendimiento" y "jugar a ganar"?
    - Por ejemplo, Sims, Minecraft: ¿estado ganador?
    - Tetris o Halo es fácil definir qué significa jugar mejor
  - No siempre jugamos a ganar
    - Pasar tiempo, relax, probar nuevas estrategias, explorar el juego, juego de rol, estar con amigos.
- **¿Queremos que la IA tome el rol de un jugador humano o de un NPC?**

Más que para ganar a jugadores humanos, la IA en videojuegos se diseña y emplea para **mejorar la experiencia de los jugadores humanos**



### IA para Jugar a Videojuegos

La IA puede estar jugando un videojuego para **ganar** o por la **experiencia** de jugar, y en los roles de **Jugador** o de No-Jugador (otros componentes del videojuego).

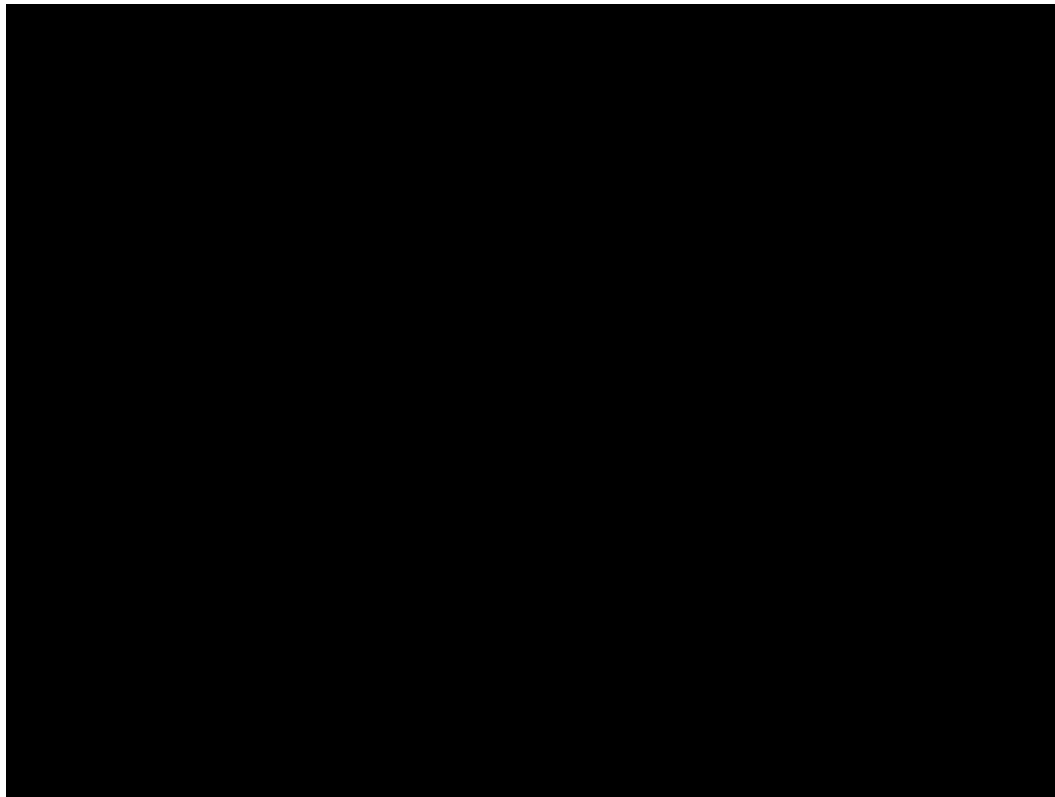
	Player	Non-Player
Win	<p><b>Motivation</b> Games as AI testbeds, AI that challenges players, Simulation-based testing</p> <p><b>Examples</b> Board Game AI (TD-Gammon, Chinook, Deep Blue, AlphaGo, Libratus), Jeopardy! (Watson), StarCraft</p>	<p><b>Motivation</b> Playing roles that humans would not (want to) play, Game balancing</p> <p><b>Examples</b> Rubber banding</p>
Experience	<p><b>Motivation</b> Simulation-based testing, Game demonstrations</p> <p><b>Examples</b> Game Turing Tests (2kBot Prize/Mario), Persona Modelling</p>	<p><b>Motivation</b> Believable and human-like agents</p> <p><b>Examples</b> AI that: acts as an adversary, provides assistance, is emotively expressive, tells a story, ...</p>

Ej. Rubber banding sería un componente, no un jugador como tal.

Característica oculta en juegos de carreras, que permite a oponentes controlados por el ordenador competir por la victoria no importa lo lejos que estén

## Generación automática de contenidos:

- Generar contenido del juego (niveles, laberintos, mazmorras, criaturas, ítems del juego,...) algorítmicamente en lugar de ser diseñado por humanos
- Juegos que generan parte de su contenido propio pueden tener más contenido (potencialmente infinito), sin diseñarlo a mano, y reduce los requisitos de almacenamiento.

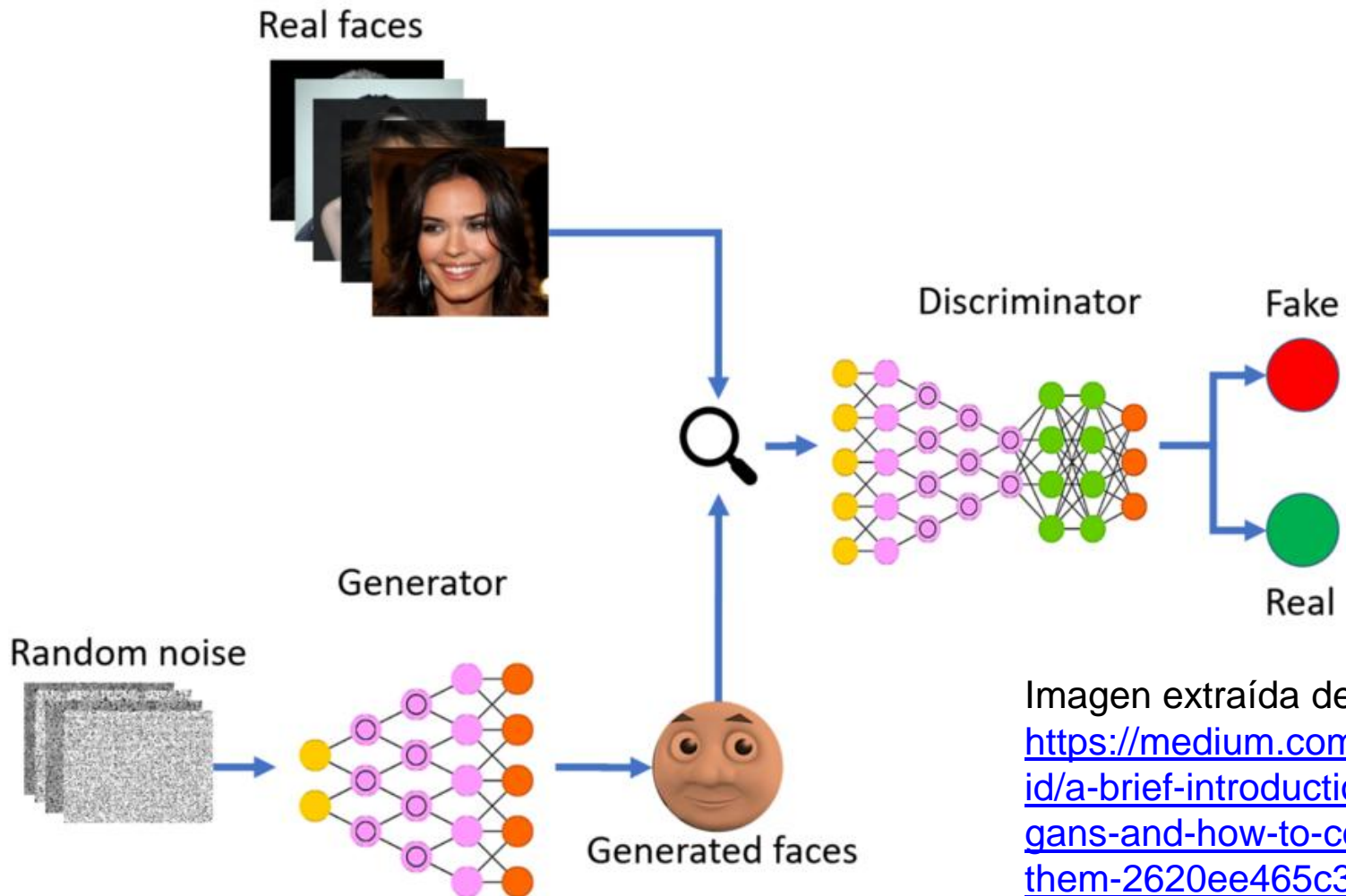


Contenidos creados por NVIDIA usando técnicas de deep learning y, más en concreto, **Generative Adversarial Networks (GANs)**

Véase el trabajo de Julian Togelius (New York University)

## Generación automática de contenidos:

### Red Generativa Antagónica (Generative Adversarial Network)



## Análisis de datos de juegos y modelado de jugadores

- **Minería de datos de comportamiento de usuarios:** cómo la gente usa el juego, qué partes juegan más, qué causa hace que paren de jugar,...
  - Identificar tipos de jugadores o predecir su comportamiento vía analíticas de juegos y jugadores.
- **Modelado de la experiencia de juego:** discernir la habilidad y estado emocional del jugador (p.ej. para ajustar la dificultad de un videojuego en tiempo real basado en la habilidad del jugador)
  - Comprender patrones cognitivos, afectivos y conductuales de los jugadores
- Responder a la pregunta: **¿Cómo la interacción con un juego es experimentada por jugadores individuales?**

Yannakakis, Georgios N., et al. *"Player modeling."* Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2013.

Hooshyar, D., et al. *"Data-driven approaches to game player modeling: a systematic literature review."* ACM Computing Surveys 50.6 (2018): 90.


Nguyen, Truong-Huy D., et al. *"Analytics-based AI techniques for a better gaming experience."* Game AI Pro 2 (2015): 481-500.

## Análisis de datos de juegos y modelado de jugadores

- Noticia reciente relacionada:

≡ EL PAÍS

ESPAÑA

 En 2020, los videojuegos contratan psicólogos para diseñar los controles y las mecánicas de algunos juegos. Por un lado, hacen cambios en la interfaz para que sean más usables y menos frustrantes. Pero también trabajan para hacer mejor la experiencia, calibrando la dificultad o la proporción perfecta entre rutinas y sorpresas. Los psicólogos hacen juegos más divertidos y absorbentes... pero eso esconde un dilema ético: ¿es peligroso que un juego sea demasiado adictivo? En Yorokobu le preguntan por todo esto a Celia Hodent, la psicóloga de *Fortnite*. | [Yorokobu](#)

Las técnicas de IA pueden contribuir a estas tareas.

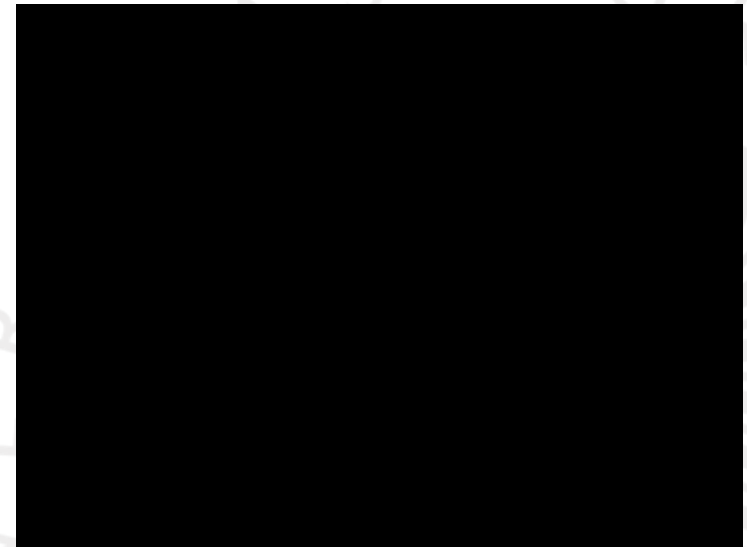


## Agentes creíbles (Believable Agents)

- ¿Cómo evaluar la credibilidad de un agente en un videojuego?
  - Test de Turing (1950) en Juegos: un número de jueces humanos debe averiguar si el comportamiento observado corresponde a un humano o a un bot controlado por IA.
  - En 2012, en una competición de Unreal Tournament, un bot consiguió pasar la prueba.

### Artificially Intelligent Game Bots Pass the Turing Test on Turing's Centenary

<https://news.utexas.edu/2012/09/26/artificially-intelligent-game-bots-pass-the-turing-test-on-turings-centenary/>



***"The idea is to evaluate how we can make game bots,*** which are nonplayer characters (NPCs) controlled by AI algorithms, ***appear as human as possible,"*** said **Risto Miikkulainen**, professor of computer science in the College of Natural Sciences. Miikkulainen created the bot, called the UT<sup>2</sup> game bot, with doctoral students Jacob Schrum and Igor Karpov.

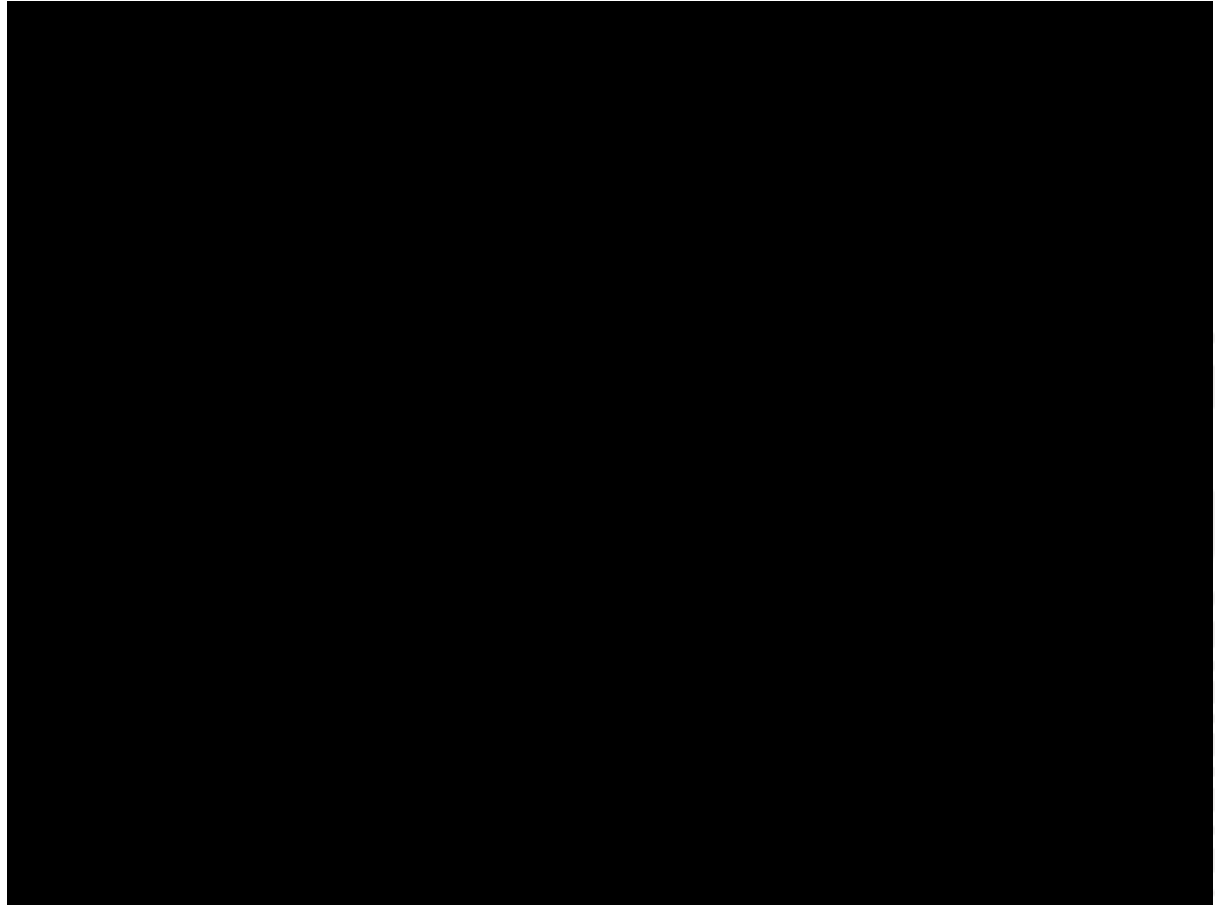
## Agentes creíbles (Believable Agents)

- Representa un factor clave en el **éxito de una IA en un juego** comercial.
- En ocasiones, es un indicador del **nivel de integración** de la IA en el diseño del juego.
- Ejemplo:
  - comportamientos de los NPC injustificables (e.g., los NPCs se quedan atascados en los callejones sin salida).
    - romper la **suspensión de la incredulidad** (anular voluntariamente el realismo y la lógica en favor del disfrute)
    - reduce la **inmersión del jugador**
    - O la IA no se ha testado bien o el diseño del juego no ha considerado el diseño de la IA.

## Mejorar la capacidad humana

- Las sinergias entre la IA y los humanos, pueden acabar mejorando las capacidades humanas

Fragmento extraído  
de la película  
AlphaGo (Greg Kohs,  
2017)

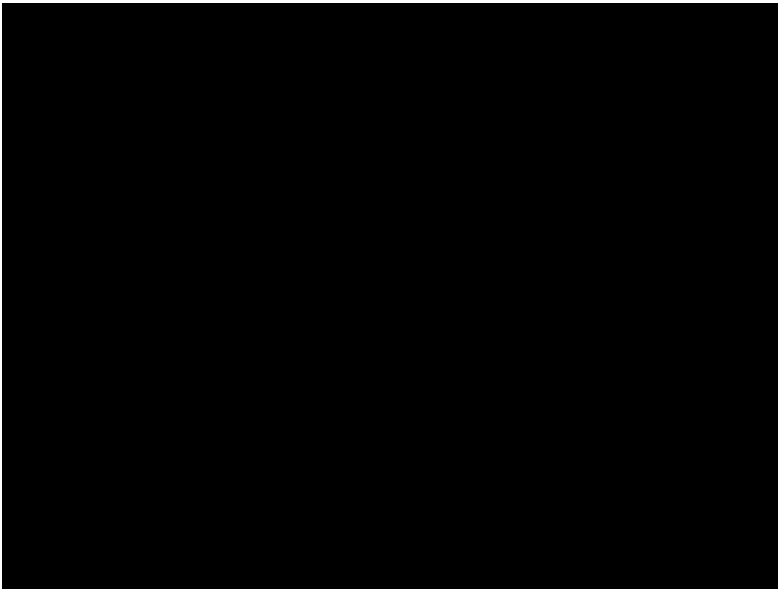


- In the 2 months following the match, **Lee Sedol won every tournament game he played.**
- **After training with AlphaGo, Fan Hui won the 2016 European Go Championship.**
- As **new players discovered the game**, a worldwide shortage of Go boards was reported.

## Testear/Validar algoritmos de IA

- A menudo, los problemas del mundo real son demasiado complejos y difíciles de abordar.
- Los videojuegos ofrecen:
  - Gran variabilidad y escalabilidad en la definición de problemas.
  - Posibilidad de disponer de gran control sobre el dominio.
  - Generalmente, facilidad a la hora de evaluar los resultados.

Ejemplo: subir escaleras



Realizar tareas sencillas en el mundo real no es fácil



Percibir y actuar en mundos virtuales es más fácil, y te puedes centrar en aspectos concretos (p.ej. la toma de decisiones).

# 1. ¿Para qué se usa la IA en Videojuegos?

→ Nos centramos en IA para “Jugar a Videojuegos”

# 2. Características de los Videojuegos

→ Para poder definir qué técnica de IA es más apropiada

# 3. Repaso de Técnicas de IA

→ De más éxito para Jugar a Videojuegos



### Nos centraremos solo en IA para Jugar a Videojuegos

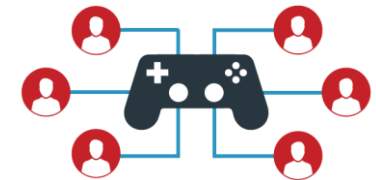
Profundizaremos en:

- Características de un juego
  - Número de jugadores
  - Observabilidad
  - No-determinismo
  - Granularidad temporal
  - Ramificación
- Características de una solución de IA para un juego
  - Representación del estado del juego
  - Posibilidad de simulación
  - Tiempo de entrenamiento
  - Cuántos juegos puede jugar la IA



Para escoger un método de IA para un videojuego **es importante saber las características del juego**, lo que determinará qué algoritmo es más efectivo.

## Número de jugadores



- **Un jugador:** juega A, que puede ganar o perder
  - Puede tener NPCs no triviales (que ayudan al jugador; son del mismo equipo)
  - Buen rendimiento de algoritmos de **búsqueda en árboles/grafos**.
  - Ej.: Puzzles, Dark Souls, Death Stranding, Pokémon, The Legend of Zelda: Breath of the Wild
- **Dos jugadores:** juegan A y B; si gana A, pierde B; si gana B, pierde A
  - Bipersonales, con adversario, *suma cero*.
  - No sabemos qué hará en concreto el oponente pero sabemos todas sus posibilidades.
  - Buen rendimiento de **minimax con  $\alpha$ - $\beta$  pruning** y algoritmos de **aprendizaje por refuerzo**.
  - Ej.: Ajedrez, Damas, Go, Heathstone, A Way Out, Wargames (Warhammer, Infinity, Flames of War), Man of Medan
- **Multi-jugador:** juegan A, B, C,... y si gana uno de ellos, pierden los demás
  - Es común **tratarlo como un jugador**, pero **con un modelo de qué hacen otros jugadores** (asumir un comportamiento concreto, aprender de la observación o un modelo aleatorio)
  - Ej.: World of Warcraft, Juegos online (Call of Duty, League of Legends, World of Tanks)

## Determinismo

- Ausencia de aleatoriedad en el juego → comportamiento predecible
  - Ejemplo: ajedrez. Si los dos jugadores repiten los mismos movimientos empleados antes en otra partida, el resultado será exactamente el mismo.
  - No determinismo → existe un grado de incertidumbre/aleatoriedad.
  - Otra forma de verlo: la misma acción en el mismo estado siempre tendrá la misma consecuencia.

## Observabilidad

- Juegos con información oculta (observabilidad parcial). La mayoría son así (Mario, FPS como Halo, StarCraft, juegos de cartas): solo percibes parte del mundo.
  - Lo más simple: técnica reactiva ignorando la información oculta.
  - En juegos de estrategia requiere adquisición de información de forma proactiva.
  - En juegos de cartas (póker) lo esencial es modelar la información oculta (¿qué puede tener mi adversario en la mano?).

La gestión de información oculta y no determinismo añaden considerable complejidad computacional.

- **Factor de ramificación y espacio de acciones**

- Espacio de acciones: ¿Qué repertorio de acciones tiene el agente?
- Ramificación: ¿Cuántas acciones se pueden aplicar en un estado concreto de juego?
  - Pac-Man: 4, Super Mario: 32, Ajedrez: 35, Go: 250
  - Juegos de estrategia (StarCraft): múltiples unidades pueden moverse en cada turno. Si tienes 6 unidades y cada una puede tomar 10 acciones:  $10^6$  posibilidades por cada turno.

- **Granularidad temporal.** ¿Con qué frecuencia el jugador tiene que tomar una acción?

- Basados en turnos (juegos de tablero) vs tiempo real (FPS, carreras...)
- Limita hasta cómo de lejos (en el futuro) se puede “mirar”.

		OBSERVABILIDAD	
		INFORMACIÓN PERFECTA	INFORMACIÓN IMPERFECTA
DETERMINISMO	DETERMININISTA	DAMAS, AJEDREZ, <b>PAC-MAN, SPACE INVADERS</b>	BATTLESHIP, CIVILIZATION, <b>TETRIS</b>
	NO DETERMINISTA	RISK, MONOPOLY, <b>SIMS, CITIES SKYLINE</b>	POKER, HEARTSTONE, <b>HALO, LEAGUE OF LEGENDS</b>

**GRANULARIDAD:** **TIEMPO REAL** (en rojo en la tabla) o **POR TURNOS** (en negro en la tabla)

**Información perfecta:** Cada jugador, cuando toma una decisión, está perfectamente informado de todos los hechos acontecidos previamente.

**Juegos deterministas:** aquellos en que no interviene el azar, y cuyo comportamiento es predecible. Si los jugadores repiten los mismos movimientos, el resultado será exactamente el mismo.

### ¿Cómo se representa el estado del juego?

- El estado del juego es la “configuración del tablero”.
  - Pero este “tablero” cambia dependiendo del juego: aventuras de texto, juegos de mesa, videojuego gráfico,...
  - El estado del juego se puede representar por medio de:
    - *Un grid*
      - *Puede estar enriquecido, como en GVG-AI, con listas de objetos. P.ej., una misma posición del tablero (es decir, del grid) puede tener asociados ‘suelo’ y ‘enemigo’.*
    - *Una representación simbólico-lógica*
      - *Por ejemplo, en ajedrez: usar 32 variables de 64 bits (en donde cada pieza indica su presencia (1) o ausencia (0) en una casilla en el tablero 8x8).*
    - *Imágenes raw*
    - *...*
- ¿Tienes acceso a la API del juego?
  - Usar las funciones de la API para generar el estado
- Si no, recurrir a información visual: imágenes como estados.
  - La IA del juego aprende bajo “las mismas condiciones” que un humano, y se puede alcanzar inteligencia de nivel humano.



- **¿Hay un simulador de acciones del juego (Forward Model)?**
  - Dado un estado  $s$  y una acción  $a$ , proporciona un  $s'$  que se podría alcanzar en el juego real.
    - $a(s) \rightarrow s'$
  - Necesario para poder aplicar cualquier técnica basada en árboles, o cualquier otra que necesite simular los resultados de varias acciones, como aprendizaje por refuerzo.
    - Permite explorar las consecuencias de aplicar una/múltiples acciones.
  - Tiene que ser rápido. Hay muchos juegos en los que no puede ser así y recurren a un *forward model* aproximado.
  - Si lo ofrece el juego, perfecto; si no, en general se programa sin excesiva dificultad.

### ¿Existe la posibilidad de entrenar/aprender?

- Distinción en IA:
  - **Algoritmos deliberativos**
    - **Decidir qué hacer examinando posibles acciones y sus consecuencias.**
    - Se necesita información extensa del dominio. Por ejemplo, para optimizar el viaje de un punto a otro en un mapa, se necesita saber a priori el coste de viajar de un punto a otro.
  - **Algoritmos de aprendizaje**
    - **Por refuerzo:** descubrir cuál es la acción más adecuada para cada estado posible en el mundo, recibiendo recompensas (premio o castigo) por el comportamiento realizado.
    - En el ejemplo anterior, el camino más corto se descubriría explorando (aleatoriamente) caminos y encontrando una *policy* (estrategia) para llegar al destino.

### ¿Cuántos juegos va a ser capaz de jugar la IA?

- **IA general para juegos:** no se busca una política para un único juego, sino un agente genérico capaz de jugar cualquier juego (o al menos cualquier juego dentro de un género/repertorio específico).
  - General Game Playing Competition
    - <http://logic.stanford.edu/ggp/>
  - General Video Game AI Competition
    - <https://github.com/EssexUniversityMCTS/gvgai/wiki/GVG-Framework>
  - Arcade Learning Environment
    - <https://github.com/mgbellemare/Arcade-Learning-Environment>
  - OpenAI gym
    - <https://gym.openai.com/>
  - StartCraft
    - <https://www.cs.mun.ca/~dchurchill/starcraftaicomp/index.shtml>
  - Malmö Project
    - <https://github.com/Microsoft/malmo>

# 1. ¿Para qué se usa la IA en Videojuegos?

→ Nos centramos en IA para “Jugar a Videojuegos”

# 2. Características de los Videojuegos

→ Para poder definir qué técnica de IA es más apropiada

# 3. Repaso de Técnicas de IA

→ De más éxito para Jugar a Videojuegos

En distintas asignaturas de IA (IA de 2º, TSI, AA, MH) habéis conocido, o vais a conocer, un número importante de métodos de IA. La mayoría pueden usarse para jugar a juegos en cualquier rol/dimensión.

Los agrupamos en 4 familias:

- Técnicas basadas en Planificación
  - Incluyendo arquitecturas Reactivas/Deliberativas
- Búsqueda Monte Carlo en Árboles
- Aprendizaje por refuerzo
- Estrategias Híbridas

### Tipos de agentes:

- Reactivos
  - Operan en modo **estímulo-respuesta**.
  - Inconvenientes:
    - Todas y cada una de las situaciones a las que se puede enfrentar el agente deben ser consideradas a priori.
      - *En entornos complejos, pueden llegar a contener muchas reglas, haciendo muy costoso y complejo su mantenimiento.*
    - Su naturaleza reactiva les priva de realizar razonamiento a largo plazo.
  - Ventaja: Gran rapidez de reacción.
- Deliberativos
  - Un sistema de planificación **usa la información del modelo de mundo** (descripción del mundo) **para construir un plan** (empleando un conjunto de acciones/operadores) que permita alcanzar los objetivos del agente.
  - Inconveniente: Cada situación requiere replanificar → lentitud!
  - Ventajas:
    - Agentes deliberativos pueden enfrentarse a objetivos en el largo plazo.
    - Situaciones no previstas pueden ser resueltas

Nareyek, Alexander. "Intelligent agents for computer games." In International Conference on Computers and Games, pp. 414-422. Springer, Berlin, Heidelberg, 2000.

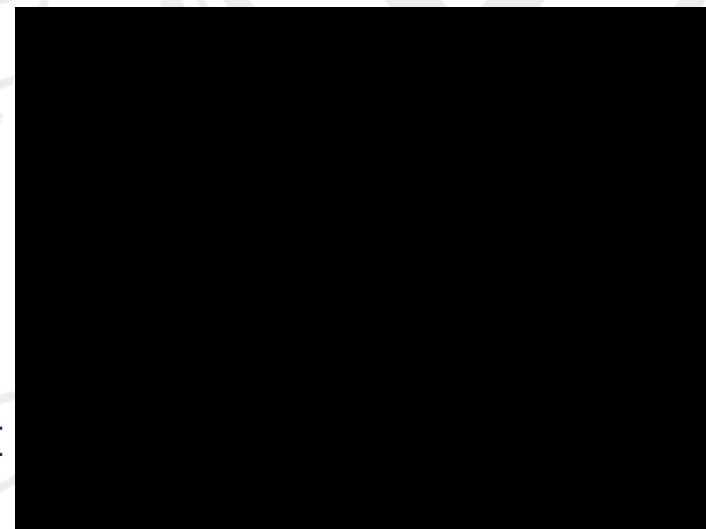


### Técnicas basadas en Planificación

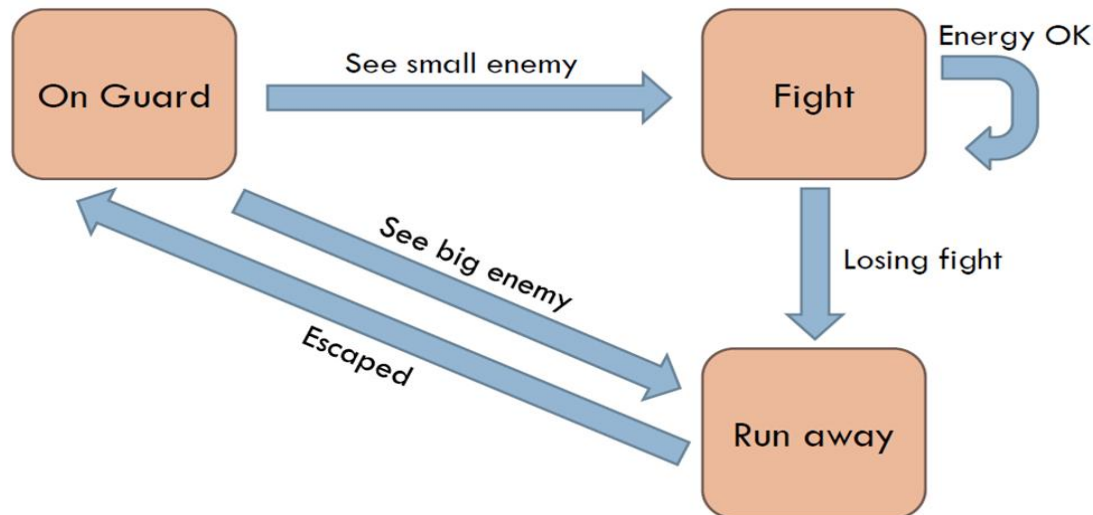
Planificación en un sentido muy amplio: capacidad de seleccionar un conjunto de acciones a futuro buscando en un espacio de estados. Ejs: planificación de tareas, planificación de caminos.

- Búsqueda clásica
  - **Aplicable en juegos con observabilidad total, factor de ramificación bajo y un modelo progresivo rápido.** En la práctica fallan en juegos con grandes espacios de estados.
  - **Path-finding:** A\* y todas sus variantes es muy usada en videojuegos modernos. Cuando un NPC se mueve en un FPS o RPG está usando alguna versión de A\*.
  - **Arquitecturas híbridas:** En general, Path-finding se integra con Finite State Machines o Behavior Trees (siguientes transparencias) en arquitecturas híbridas (reactivas-deliberativas).

Por ejemplo, ganador de 2009 Mario AI Competition (<http://www.marioai.org/>)



**Máquinas de estados finitos (FSM:** Finite State Machines): lenguaje básico para representar estados de jugadores.



- Fácil de entender
- Fácil de implementar (if-then-else)

Pero...

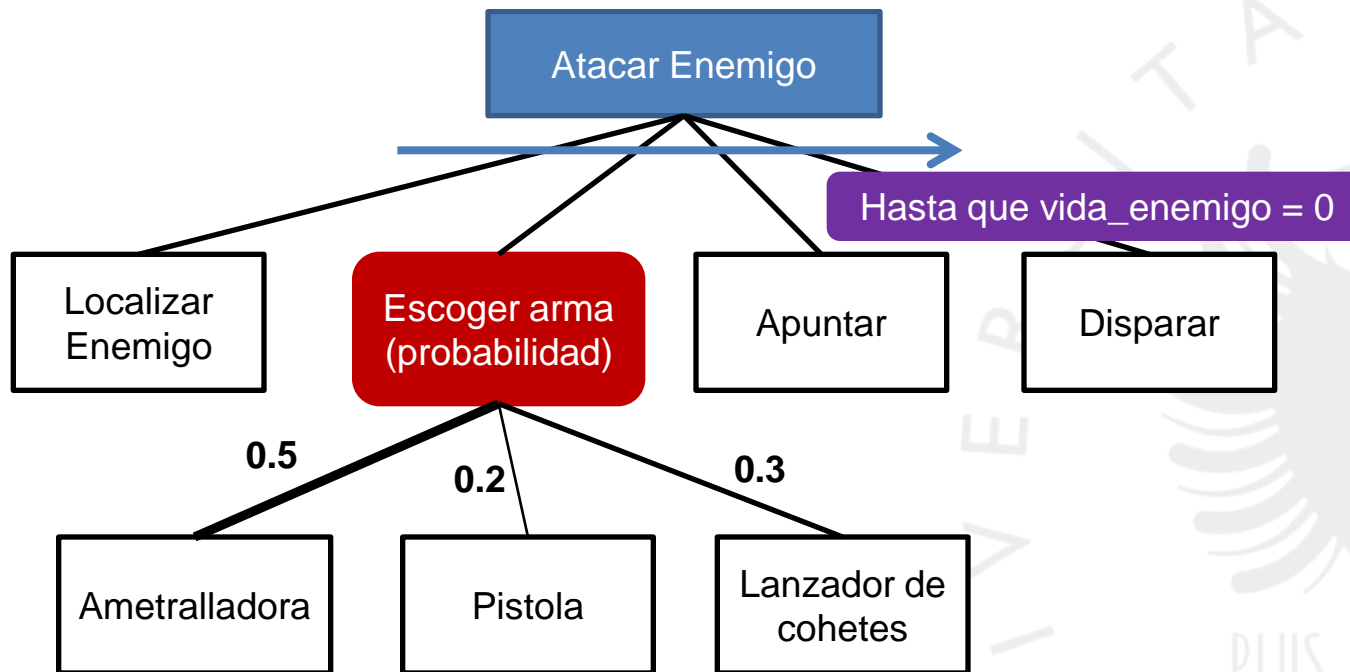
- Simple en los comportamientos que permite expresar
- Si hay muchos estados → complejo de gestionar!

Slides "Introduction to AI STRIPS Planning... and Applications to Video-games!" de Stavros Vassos, University of Athens. Mayo 2012.

FSMs dominaron el control y toma de decisiones de NPCs hasta mediados los años 2000.

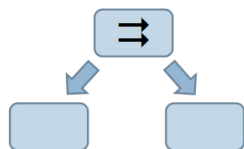
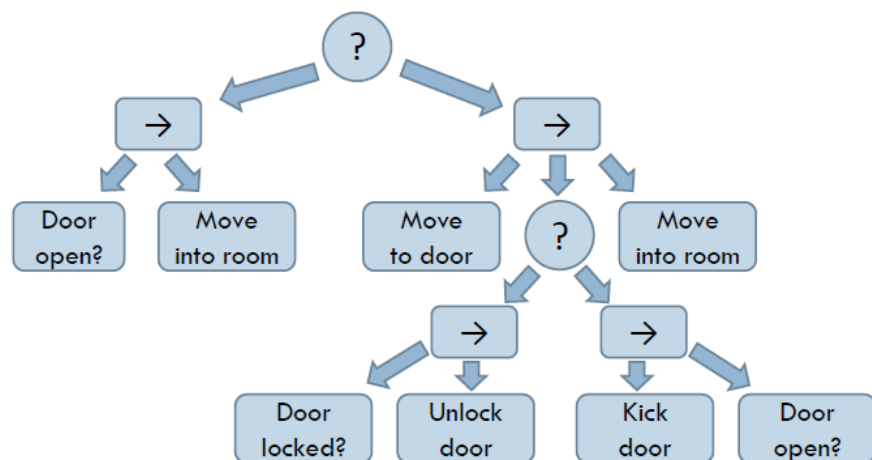
### Árboles de comportamiento (BTs: Behavior Trees)

- Modelo más expresivo (modular), que permite alcanzar **comportamientos complejos compuestos de tareas sencillas**.
- Están compuestos de comportamientos (en lugar de estados, como FSMs)

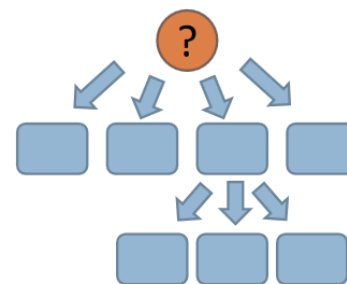


BTs dominan el desarrollo de juegos desde HALO2 (2004).

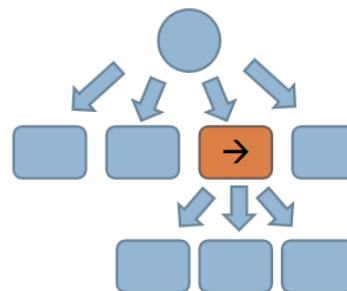
Ejemplo: agente que tiene que entrar en una habitación



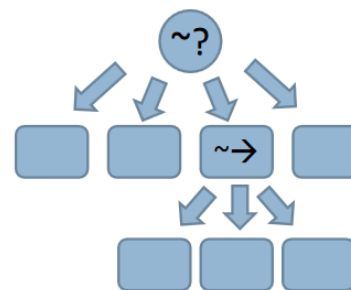
E.g., perform move actions while also shooting at target



E.g., succeed if **any** of the child tasks succeed



E.g., succeed if **all** of the child tasks succeed



**Non-deterministic** sequence task and selector task

### A diferencia de los FSMs:

- Permiten ejecutar dos acciones diferentes a la vez.
- Son más escalables.
- Más fáciles de mantener.
- ...

## FSM y BTs son técnicas reactivas

- ❑ El NPC sigue una **estrategia pre-programada** que especifica cómo el NPC debe reaccionar en el juego, dependiendo de su estado/nodo actual y las condiciones que posee en el mundo.
- ❑ Una secuencia de acciones que pueden ser ejecutadas en el juego ([move to door, kick door, move into room]), necesita ser **representada explícitamente en la estructura de los FSMs o BTs**.
- ❑ FSMs y BTs son usados para toma de decisiones de NPCs

**IDEA:** Reemplazar las estrategias pre-programadas con una **descripción de objetivos y acciones disponibles**, y **buscar en tiempo real** (p.ej. usando A\*) la estrategia que alcanza el objetivo en el estado actual



**Goal Oriented Action Planning (GOAP)**

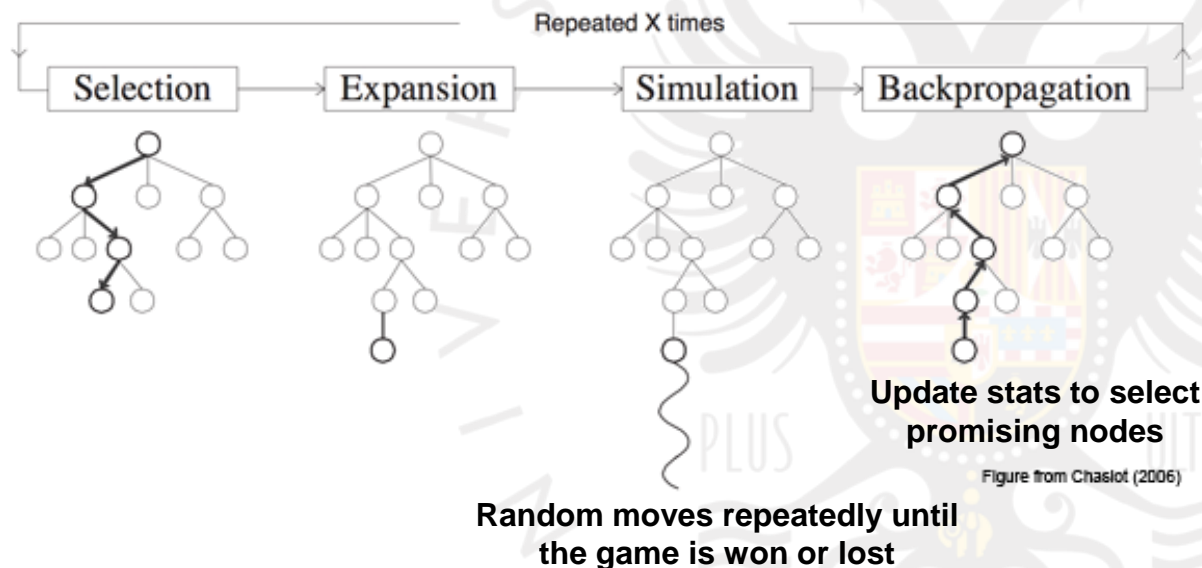


### Búsqueda Monte Carlo en Árboles (MCTS, Monte Carlo Tree Search)

#### Limitaciones técnicas deliberativas

Por ejemplo, en juegos con alto factor de ramificación (Go, Ajedrez), información imperfecta (Poker, Hundir la Flota), y juegos no determinísticos (risk, monopoly)

- **Objetivo: dado el estado actual del juego y un alto factor de ramificación, ¿cuál debería ser el siguiente movimiento?**
- En este caso, se construye un árbol asimétrico y estadístico
  - Seleccionar el siguiente nodo a expandir siguiendo una fórmula basada en el algoritmo de Monte Carlo.
  - Usar simulaciones para generar valoraciones basadas en estados terminales.

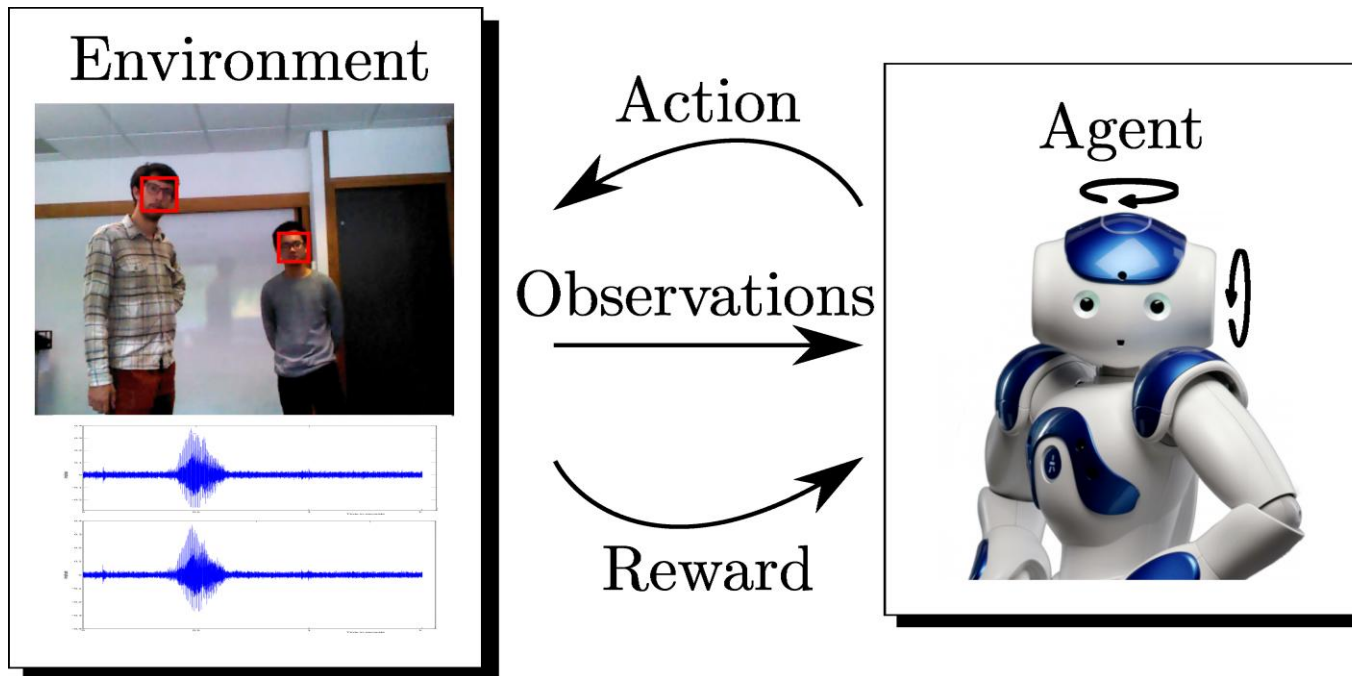


Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., ... & Dieleman, S. (2016). Mastering the game of Go with deep neural networks and tree search. Nature, 529(7587), 484.



### Aprendizaje por refuerzo (reinforcement learning)

Determinar cuál es la acción más adecuada para cada estado posible en el mundo, recibiendo recompensas (premio o castigo) por el comportamiento realizado.



Sutton, Richard S., and Andrew G. Barto.  
*Reinforcement learning: An introduction*. MIT press, 2018.

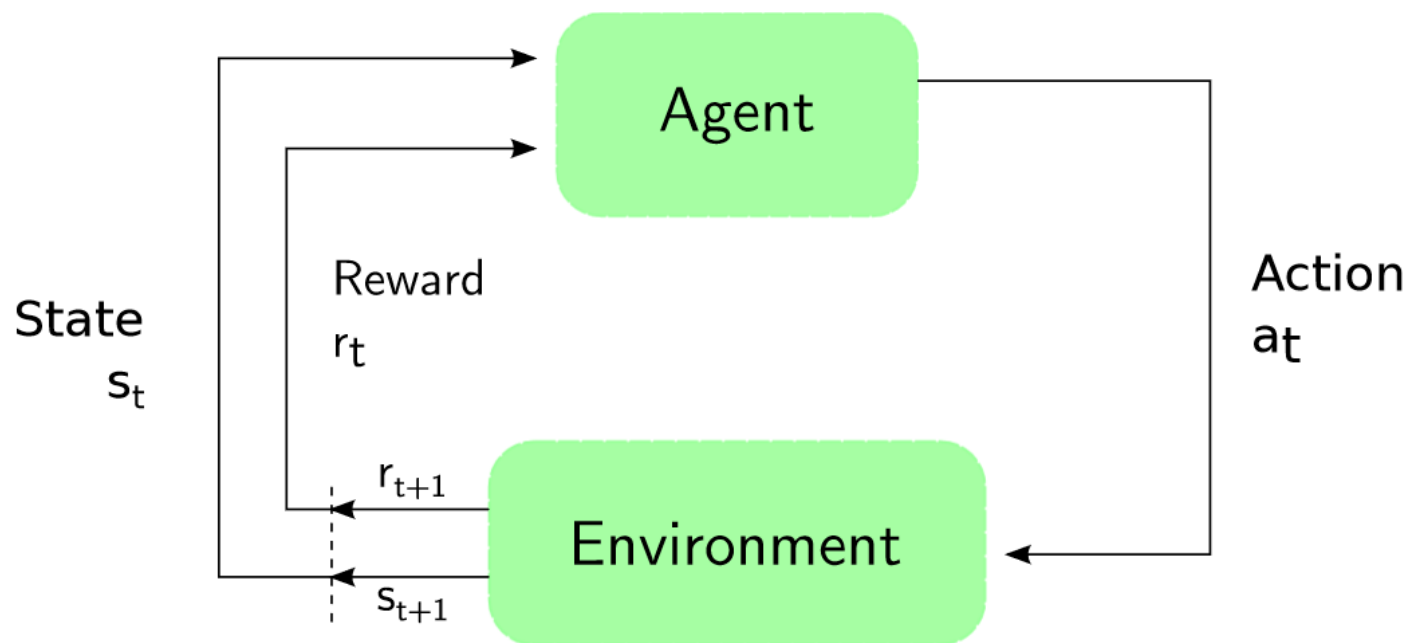
David Silver's course:  
<http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching.html> University College London

John Schulman's lectures:  
[https://www.youtube.com/watch?v=aUrX-rP\\_ss4](https://www.youtube.com/watch?v=aUrX-rP_ss4) OpenAI

### Aprendizaje por refuerzo (reinforcement learning)

Los principios de reinforcement learning son sencillos:

- 1) Un agente debe resolver una tarea y tiene disponible un conjunto de acciones
- 2) El agente escoge una acción, o conjunto de acciones, para resolver la tarea
- 3) El agente recibe feedback (señal de reward) sobre el rendimiento obtenido
- 4) El agente usa esta información para modificar su comportamiento



Notas de Javier Béjar sobre Reinforcement Learning:  
<https://www.cs.upc.edu/~bejar/apren/docum/trans/14-ApRefuerzo-eng.pdf>  
 Universidad Politécnica de Cataluña

### Aprendizaje por refuerzo (reinforcement learning)

#### Características:

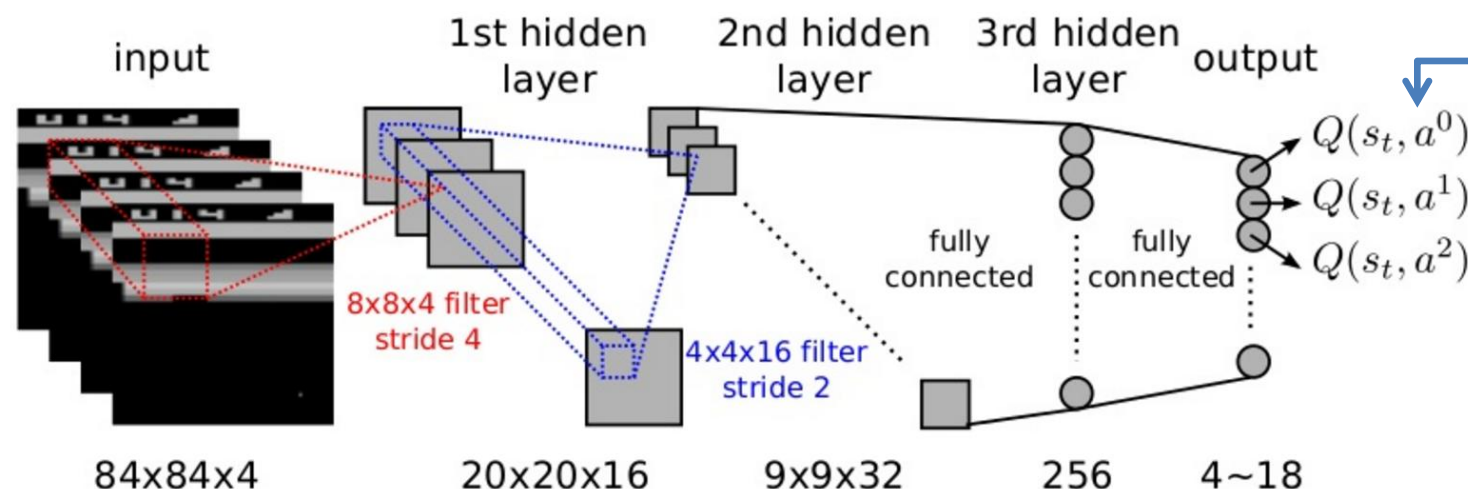
- ❑ Demora de la recompensa
  - ❑ No sabemos inmediatamente si hemos hecho lo correcto.
- ❑ Se incentiva la exploración (vs explotación)
- ❑ No conocemos necesariamente el resultado preciso de nuestras acciones antes de hacerlas
- ❑ No conocemos necesariamente todo lo relativo al estado actual
- ❑ Lifelong learning
  - ❑ Los agentes pueden acumular conocimientos durante toda una vida de experiencia → acumulan y refinan su conocimiento con el tiempo

Dayan, P. and Niv, Y., 2008. *Reinforcement learning: the good, the bad and the ugly*. Current opinion in neurobiology, 18(2), pp.185-196.

### Aprendizaje por refuerzo profundo

Para la mayoría casos de interés, almacenar la **tabla de valores Q** es impracticable → Es necesario una representación más compacta de la función de valoración (menos memoria y sin necesidad de recorrer todos los estados).

- Usar una Red Neuronal para "simular" la tabla de valores Q: valoraciones de aplicar el repertorio de acciones a un estado



La **Tabla Q** guarda valores Q: Calidad esperada (reward) si partiendo del estado  $s_t$  tomamos la acción  $a^0$

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. (2013). Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602.

### Planificación evolutiva.

En lugar de buscar una secuencia de acciones en un grafo/árbol, **plantear un proceso de búsqueda con múltiples individuos** (cada uno siendo una secuencia de acciones completa) **y emplear operadores evolutivos.**

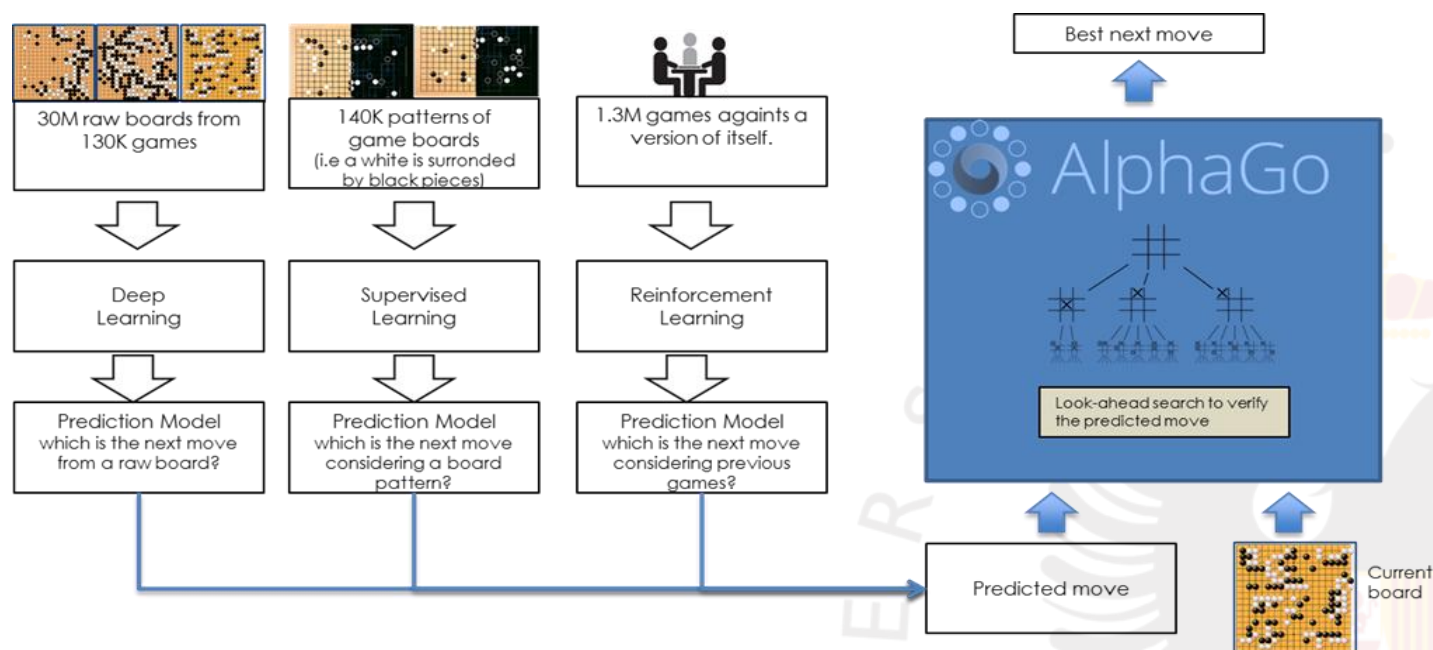
- Es una técnica que está respondiendo bien en la **General Video Game AI Competition** (<http://www.gvgai.net/>).
- Elementos críticos: codificación de los individuos y evaluación de los mismos.
- Aunque es buena para videojuegos tipo puzzle, no es buena para videojuegos que requieren reacción rápida.

Ponsen, M., & Spronck, P. (2004). Improving adaptive game AI with evolutionary learning (Doctoral dissertation, Masters Thesis, Delft University of Technology).

Hausknecht, M., Lehman, J., Miikkulainen, R., & Stone, P. (2014). A neuroevolution approach to general atari game playing. IEEE Transactions on Computational Intelligence and AI in Games, 6(4), 355-366.

Técnicas de búsqueda y de aprendizaje por refuerzo pueden combinarse para resolver problemas de videojuegos:

- **AlphaGo** (2016): Aprendizaje por refuerzo + deep learning + MCTS



Partidas jugadas por expertos humanos  
+  
Partidas contra sí misma

Redes entrenadas para guiar qué ramas del árbol explorar y evaluar si las posiciones encontradas son “ganadoras”



- **AlphaGo** (2016): Aprendizaje por refuerzo + deep learning + MCTS
- **AlphaZero** (2017): entrenado exclusivamente vía "self-play" para dominar ajedrez, shogi y go. Silver, David, et al. "Mastering chess and shogi by self-play with a general reinforcement learning algorithm." *arXiv preprint arXiv:1712.01815* (2017).
- AlphaGo y AlphaZero se adaptan al problema que intentan resolver. Por ejemplo, Go es:
  - Determinista
  - Totalmente observable
  - Espacio de acciones discreto
  - Tenemos acceso a un simulador perfecto (el juego en sí mismo, que nos permite conocer con precisión los efectos de cualquier acción)
  - Cada juego/episodio es corto (aproximadamente 200 acciones)
  - La evaluación es clara, y permite el uso de experiencia prueba-error
  - Hay enormes bases de datos de juegos humanos para entrenar

**Importancia de comprender el problema a resolver  
y aplicar las técnicas más adecuadas**



UNIVERSIDAD  
DE GRANADA



# Técnicas de los Sistemas Inteligentes

## Grado en Informática

### Curso 2019-20. Seminario 1

### IA en Robótica y Videojuegos

**Jesús Giráldez Crú, Pablo Mesejo Santiago y José Ángel Segura Muros**

{jgiraldez,pmesejo,josesegmur}@decsai.ugr.es

**Departamento de Ciencias de la  
Computación e Inteligencia Artificial**

**<http://decsai.ugr.es>**

Basado en los materiales previos elaborados por Juan Fernández Olivares