



Universidad de Granada

[decsai.ugr.es](http://decsai.ugr.es)

# Inteligencia Artificial

## Seminario 2

### Agentes Reactivos / Deliberativos



DECSAI

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

- **Introducción**
- **Los extraños mundos de BelKan**
- **Objetivos**
- **Software**
- **Método de evaluación y entrega de prácticas**



## ● **Introducción**

● Los extraños mundos de Belkan

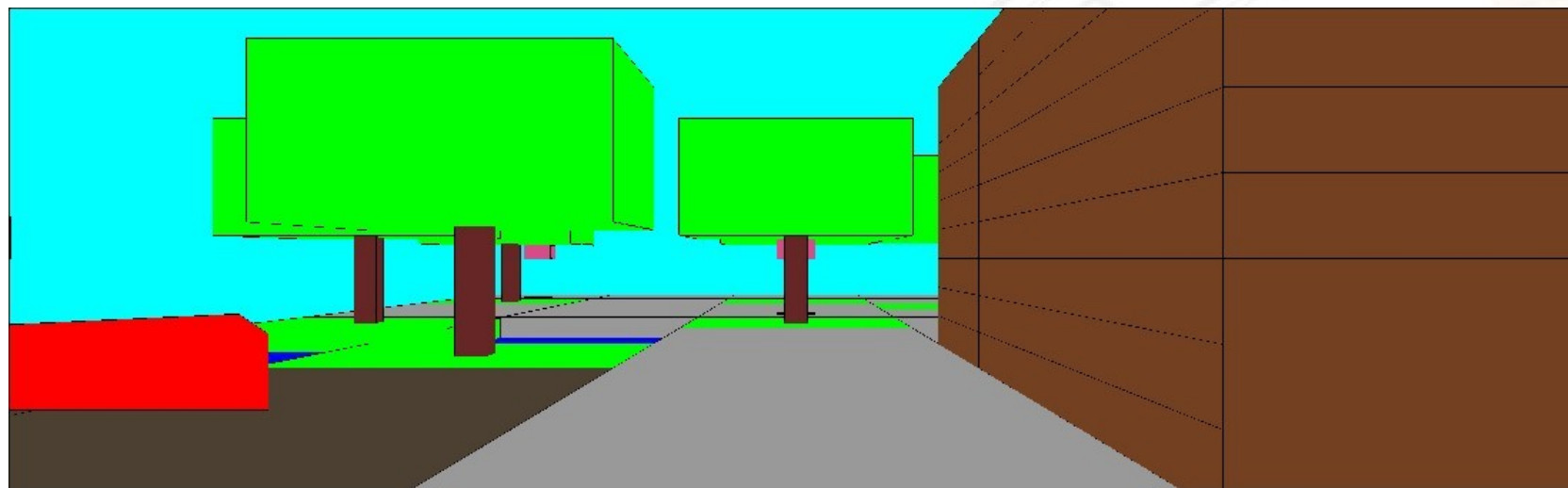
● Objetivos

● Software

● Método de evaluación y entrega de prácticas



- Diseñar e implementar un **agente reactivo y deliberativo**, capaz de percibir el ambiente y actuar considerando una representación de las consecuencias de sus acciones y siguiendo un proceso de búsqueda.



- En esta práctica se diseñará e implementará un agente reactivo y deliberativo basado en los ejemplos del libro *Stuart Russell, Peter Norvig, “Inteligencia Artificial: Un enfoque Moderno”, Prentice Hall, Segunda Edición, 2004.*
- El simulador que utilizaremos fue inicialmente desarrollado por el profesor Tsung-Che Chiang de la NTNU (National Taiwan Normal University of Science and Technology), pero la versión sobre la que se va a trabajar ha sido desarrollada por los profesores de la asignatura.

- Originalmente, el simulador estaba orientado a experimentar con comportamientos en aspiradoras inteligentes.

En su versión más simple, una aspiradora inteligente presenta un comportamiento **reactivo** puro: busca suciedad, la limpia, se mueve, detecta suciedad, la limpia, se mueve, y continúa con este ciclo hasta que se cumple alguna condición de parada.



- Otras versiones más sofisticadas permiten al robot recordar (mediante el uso de representaciones icónicas como mapas), lo cual permite que el aparato ahorre energía y sea más eficiente en su trabajo.
- Finalmente, las aspiradoras más elaboradas pueden, además de todo lo anterior, planificar su trabajo de modo que se pueda limpiar la suciedad en el menor tiempo posible y de la forma más eficiente. Estas últimas pueden ser catalogadas como **agentes deliberativos**.



### Esta práctica cubre los siguientes objetivos docentes:

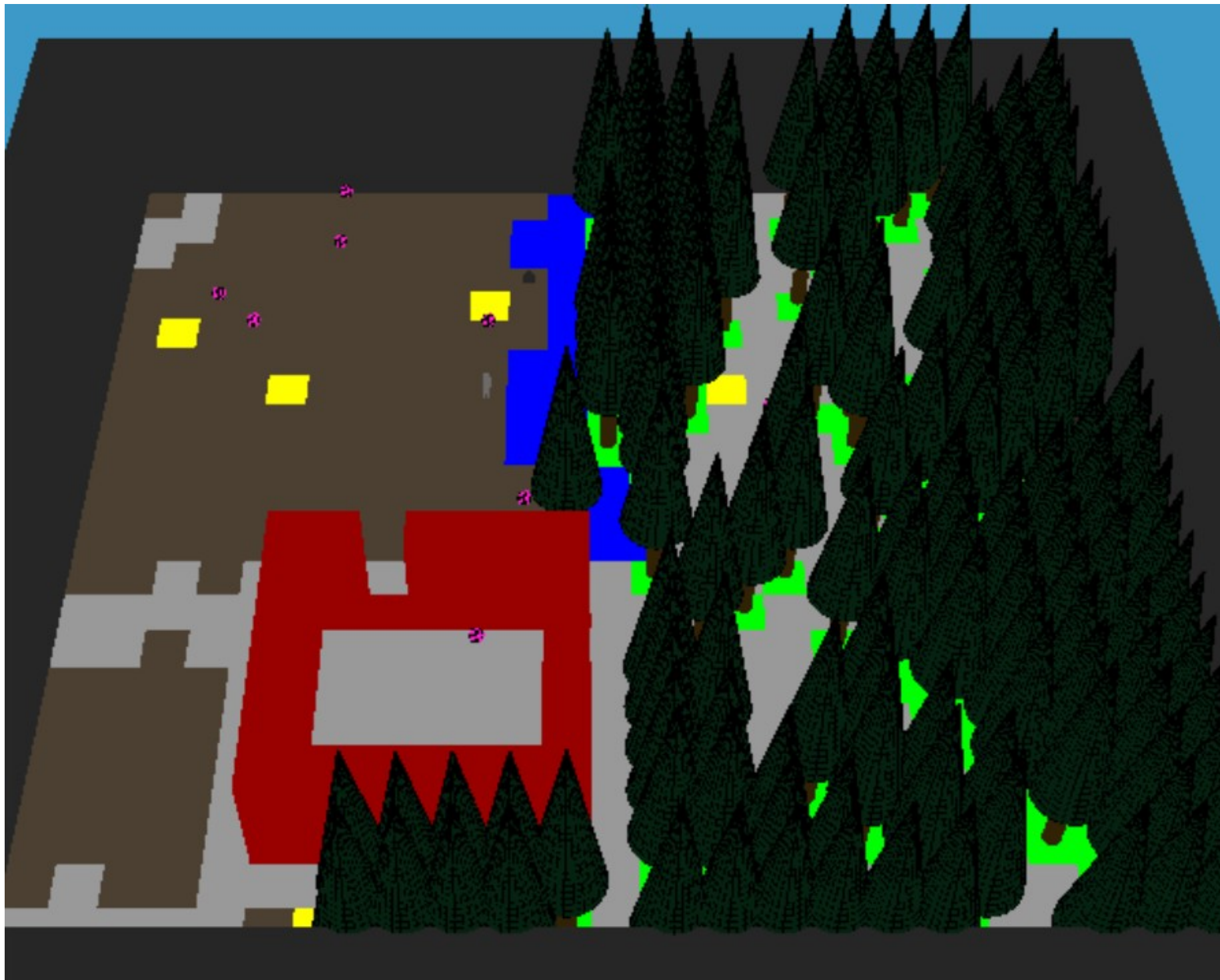
- Conocer la representación de problemas basados en estados (estado inicial, objetivo y espacio de búsqueda) para ser resueltos con técnicas computacionales.
- Entender que la resolución de problemas en IA implica definir una representación del problema y un proceso de búsqueda de la solución.
- Analizar las características de un problema dado y determinar si es susceptible de ser resuelto mediante técnicas de búsqueda. Decidir en base a criterios racionales la técnica más apropiada para resolverlo y saber aplicarla.
- Entender el concepto de heurística y analizar las repercusiones en la eficiencia en tiempo y espacio de los algoritmos de búsqueda.
- Conocer distintas aplicaciones reales de la IA. Explorar y analizar soluciones actuales basadas en técnicas de IA.

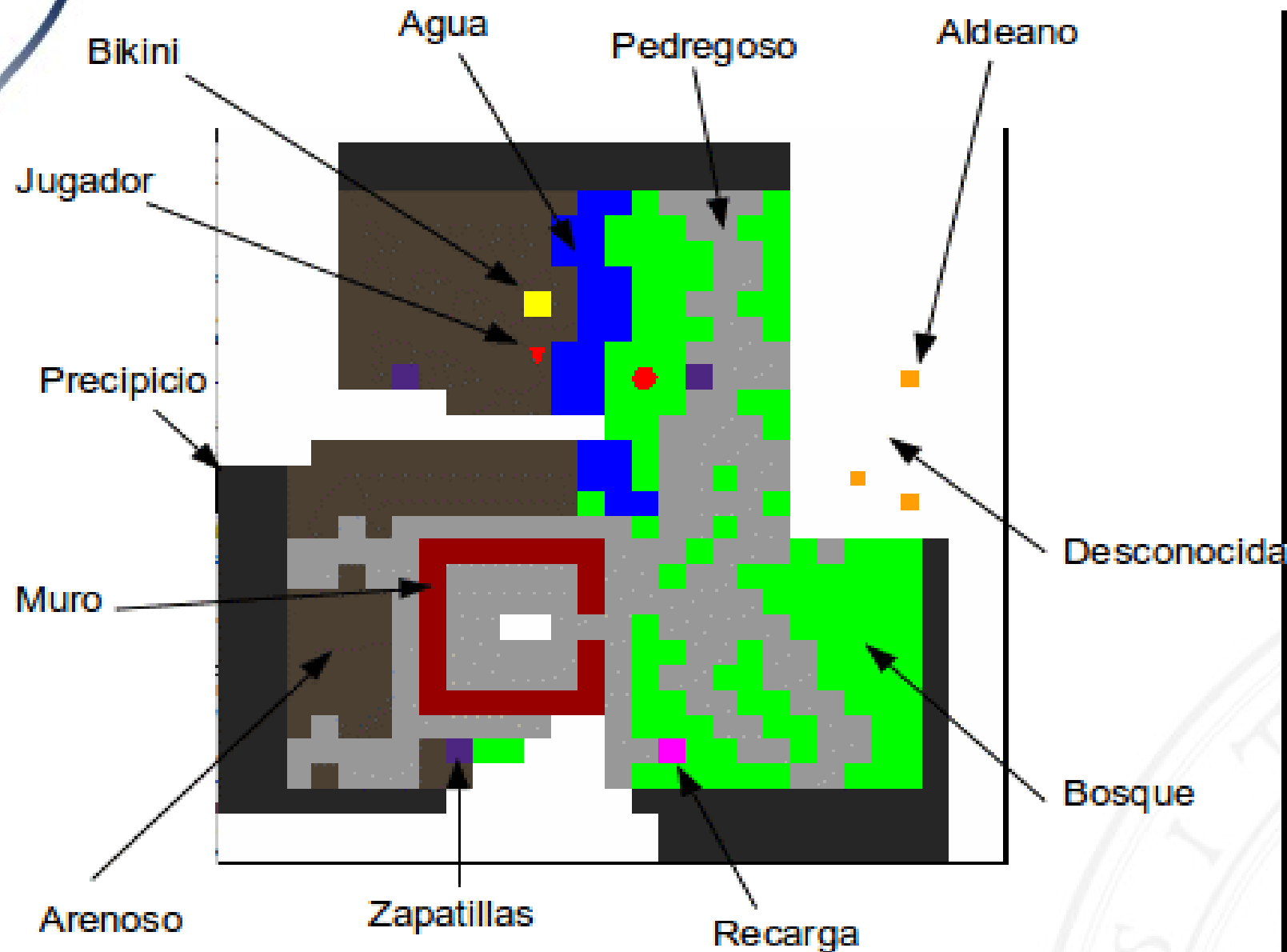


- Introducción
- **Los extraños mundos de Belkan**
- Objetivos
- Software
- Método de evaluación y entrega de prácticas



### Los extraños mundos de BelKan



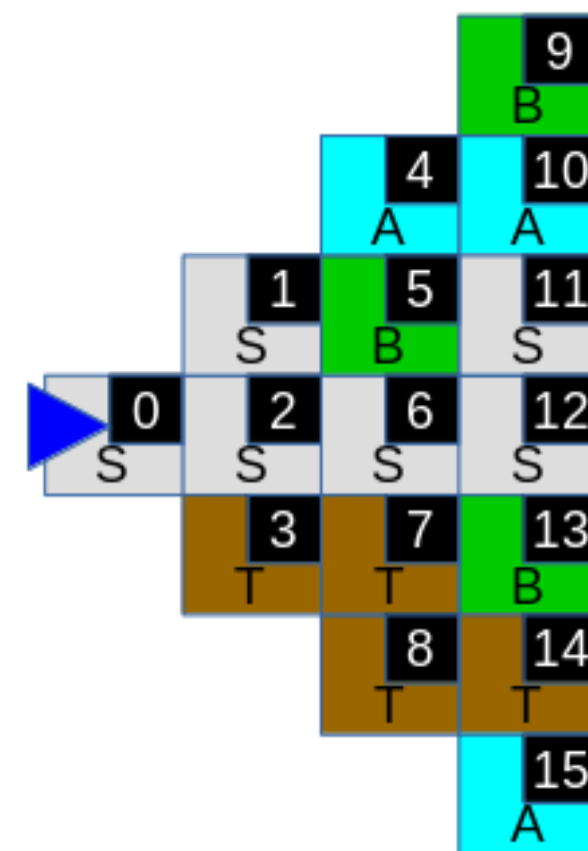


	'B'	Árboles
	'A'	Agua
	'P'	Precipicios
	'S'	Suelo pedregoso
	'T'	Suelo arenoso
	'M'	Muros
	'K'	Bikini
	'D'	Zapatillas
	'X'	Recarga
	'?'	Casillas desconocida

- El agente morirá si cae en una casilla precipicio. Tampoco puede atravesar, muros, ni otros personajes, y por tanto chocará contra ellos.
- El tamaño máximo del mapa es de 100 filas por 100 columnas.

### Sistema sensorial

- Sensor de choque (colision)
- Sensor de vida (reset)
- Sensores de posición (posF, posC, sentido)
- Sensor de destino (num\_destinos, destino)
- Sensor de carga de batería (bateria)
- Sensor de nivel (nivel)
- Sensor de tiempo consumido (tiempo)



Sensor visual: terreno (A: agua; B: bosque; T: terreno arenoso; S: terreno pedregoso,...) y superficie (para ver personajes sobre ese terreno)

Nuestro personaje puede realizar varias **acciones** distintas durante el juego:

- ***actFORWARD***: le permite avanzar a la siguiente casilla del mapa siguiendo su orientación actual. Para que la operación se finalice con éxito es necesario que la casilla de destino sea transitable para nuestro personaje.
- ***actTURN\_L***: le permite mantenerse en la misma casilla y girar a la izquierda  $90^\circ$  teniendo en cuenta su orientación.
- ***actTURN\_R***: le permite mantenerse en la misma casilla y girar a la derecha  $90^\circ$  teniendo en cuenta su orientación.
- ***actIDLE***: pues como su nombre indica, no hace nada.



- Cada acción realizada por el agente tiene un **coste** en **tiempo de simulación** y en **consumo de batería**.
- En cuanto al tiempo de simulación, **todas las acciones consumen un instante independientemente de la acción que se realice y del terreno donde se encuentre el jugador**.
  - Cada simulación conlleva 3000 instantes de tiempo.
  - Cada acción consume 1 instante de tiempo.

- En cuanto al **consumo de batería** decir que actIDLE consume 0 de batería y que **el consumo de este recurso** de las acciones actTURN\_L, actTURN\_R y actFORWARD **depende del tipo de terreno asociado a la casilla donde se inició dicha acción**. En las siguientes tablas se muestran dichos valores de consumo de batería.

actFORWARD			actTURN_L / actTURN_R		
Casilla	Gasto Normal	Gasto Reducido	Casilla	Gasto Normal	Gasto Reducido
'A'	200	10 (con <b>Bikini</b> )	'A'	500	5 (con <b>Bikini</b> )
'B'	100	15 (con <b>Zapatillas</b> )	'B'	3	1 (con <b>Zapatillas</b> )
'T'	2	2	'T'	2	2
Resto	1	1	Resto	1	1

Los muros y precipicios no son transitables!

No hay sensor que indique si disponemos del bikini o de las zapatillas, por lo que debemos crear las variables de estado correspondientes.

- Introducción
- Los extraños mundos de Belkan
- **Objetivo**
- Software
- Método de evaluación y entrega de prácticas





- El objetivo de esta práctica es dotar de un comportamiento inteligente al personaje usando un agente reactivo/deliberativo.
- La idea es explotar el encontrar caminos de forma inteligente dentro de un mapa (pidiendo ciertas optimalidades o actuando con falta de información).
- En esta práctica se han diseñado 5 niveles con dificultad incremental:
  - NIVEL 0: Demo
  - NIVEL 1: Encontrar el camino con mínimo número de acciones.
  - NIVEL 2: Encontrar el camino con mínimo consumo de batería para alcanzar una única casilla objetivo.
  - NIVEL 3: Encontrar el camino con mínimo consumo de batería que pasa por tres casillas objetivo.
  - NIVEL 4: Reto. Agente reactivo/deliberativo.

- Condiciones del problema
  - Niveles del 0 al 3
- El agente se encuentra en un **mundo inmutable y completamente conocido**.
- Aparecerá sobre un mundo de BelKan concreto conociendo su posición a través de los sensores **posF** y **posC** y orientación sobre el mapa a través del sensor **sentido**.
- El objetivo es construir un camino que le permita llegar hasta la casilla destino (mostrada en los sensores **num\_destinos** y **destino**).
- Sólo en el caso del nivel 3, el número de destinos que se deben alcanzar es de 3. En el resto, hay una única casilla destino.

## Nivel 0: DEMO

● Se incluye en el software para ayudar al estudiante a entender como debe hacer la práctica. El nivel está incompleto.

Tiene una implementación de una búsqueda en profundidad, y le falta el comportamiento reactivo que le permite llegar al objetivo.

● Así, dado el plan de movimientos que ofrece el algoritmo de búsqueda incluido, el objetivo a resolver en este nivel es llevar a cabo ese plan y hacer que el agente termine en la casilla objetivo.

## Nivel 1: Mínimo número de acciones

- En el nivel anterior se completó el agente reactivo que permite al agente seguir un plan proporcionado por un algoritmo de búsqueda.
- El objetivo del agente ahora es crear y llevar a cabo un plan de movimientos en el mapa para llegar desde su posición inicial al destino usando el menor número de acciones.
- Por tanto, este objetivo se consigue implementando el algoritmo de búsqueda en anchura.

## Nivel 2: Mínimo consumo de batería (1 sólo objetivo)

- El objetivo es crear y llevar a cabo un plan de movimientos en el mapa para llegar desde su posición inicial a una única casilla destino teniendo el menor consumo posible de batería.
- Este objetivo se consigue implementando el algoritmo de búsqueda de coste uniforme (o Dijkstra). También se puede resolver usando el algoritmo A\* con una heurística admisible.

## Nivel 3: Mínimo consumo de batería (3 objetivos)

- Se pide encontrar y llevar a cabo el camino óptimo en consumo de batería que permita al agente pasar por las **tres casillas objetivo**.
- Es importante destacar que lo que se pide es encontrar un único plan (óptimo en el consumo de batería), no tres planes a cada uno de los objetivos.
- Este objetivo se consigue adaptando la implementación del algoritmo del nivel 2 para el caso de tener más de un objetivo. Si se optó por  $A^*$ , la nueva heurística debe ser admisible para garantizar el óptimo.



### Nivel 4: Reto (Agente reactivo/deliberativo)

- A diferencia de los anteriores, en este nivel no se conoce el mundo y el agente debe explorarlo.
- El agente **conoce su posición y orientación**, también conoce una lista de 3 objetivos, pero no conoce el resto de elementos del mundo.
- Además en el mundo existen **aldeanos**. Un aldeano es un elemento dinámico en el mapa que puede moverse continuamente, puede quedarse quieto, etc. Los aldeanos pueden detectarse gracias al sensor de superficie.
- Para ir de un punto a otro del mapa puede que nuestro plan inicial no funcione correctamente ya que nos crucemos en el camino con un aldeano que nos entorpezca => deberemos integrar comportamientos reactivo y deliberativo.

● Además, debe tenerse en cuenta que no hay ningún sensor que nos indique si ya alcanzamos alguno de los 3 objetivos activos. Por tanto, el agente debe controlar este hecho.

● En este nivel, cada vez que pasemos por los 3 destinos, estos cambiarán y se generarán otros nuevos. Así, el objetivo es tratar de conseguir llegar a la mayor cantidad de casillas destino posible. Para ello se cuenta con un número de 3000 acciones como máximo, una batería con 3000 puntos de capacidad o 300" totales de tiempo. La simulación termina cuando se consume alguno de esos recursos.

● También es importante recordar que **el jugador no debe morir**.

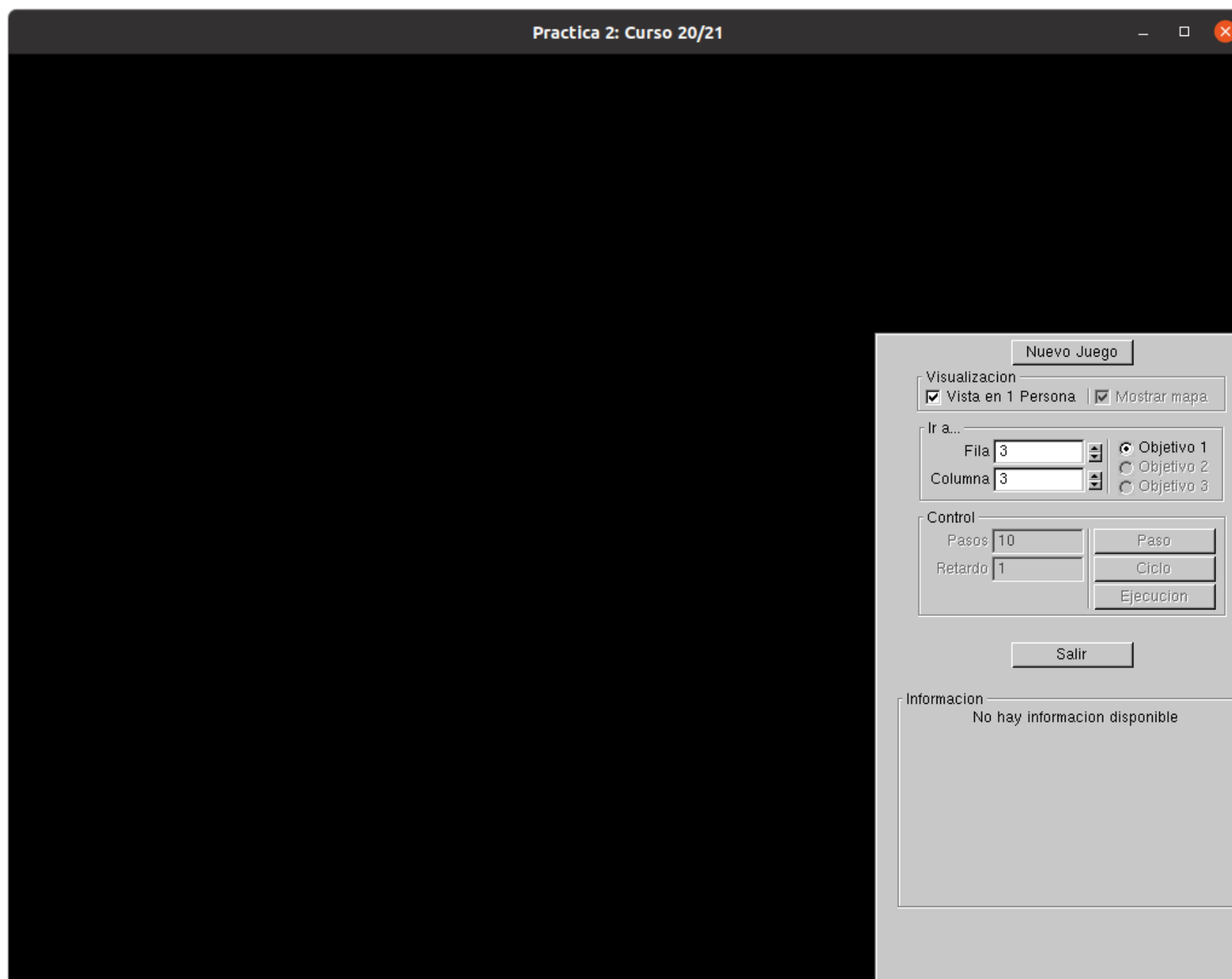


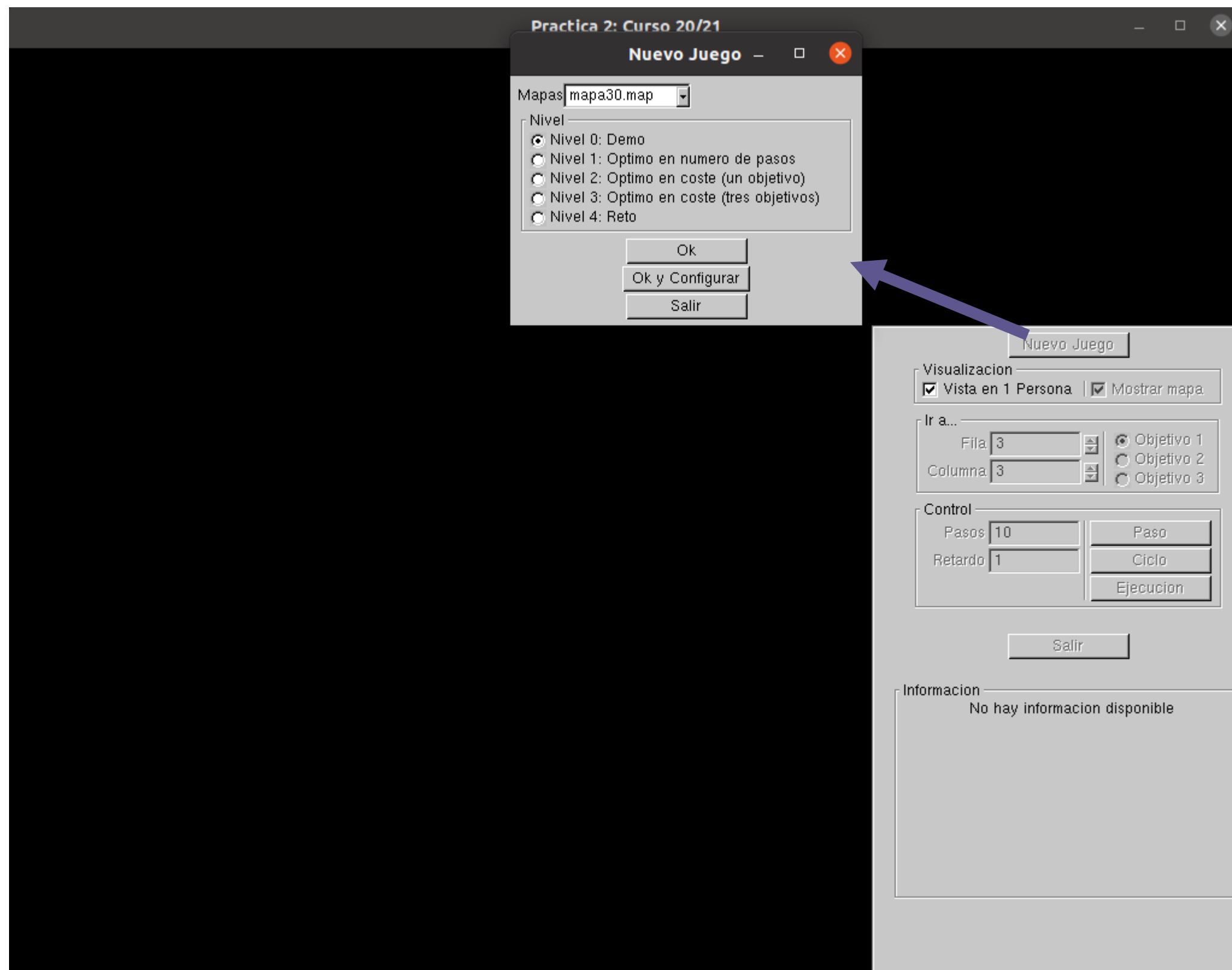
- Introducción
- Los extraños mundos de Belkan
- Objetivo
- **Software**
- Método de evaluación y entrega de prácticas



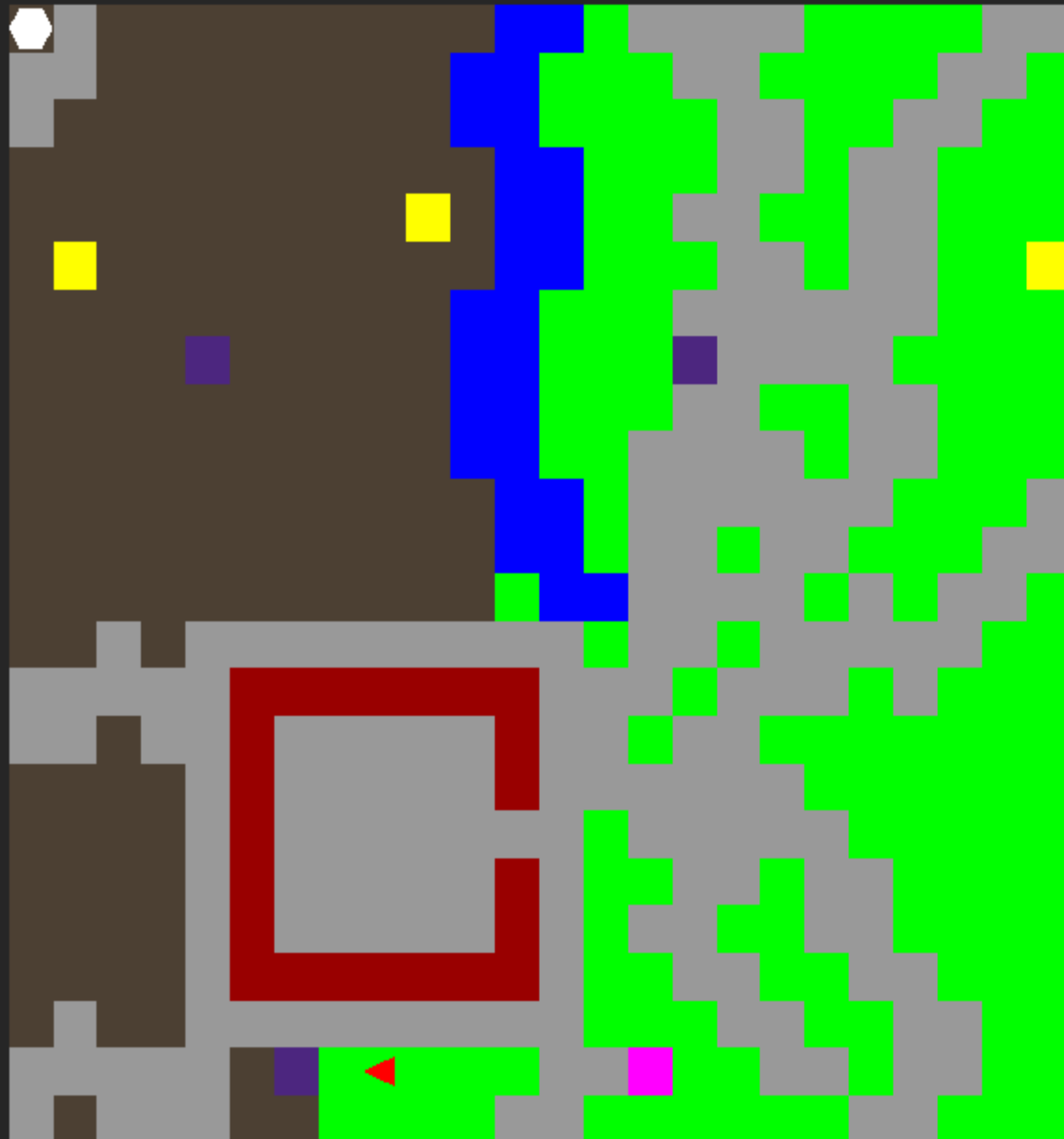
- Se proporcionan versiones del software para el sistema operativo Linux y para el sistema operativo Windows. Dicho software se puede encontrar en el apartado correspondiente dentro de la plataforma docente PRADO.
- Existen dos versiones del software: Belkan y BelkanSG. El primero corresponde al simulador con interfaz gráfica, mientras que el segundo es un simulador en modo *batch* sin interfaz.
- Todos los detalles y explicación de la instalación, uso y detalles las variables necesarias para su desarrollo se encuentra en el guion de prácticas asociado a esta presentación.

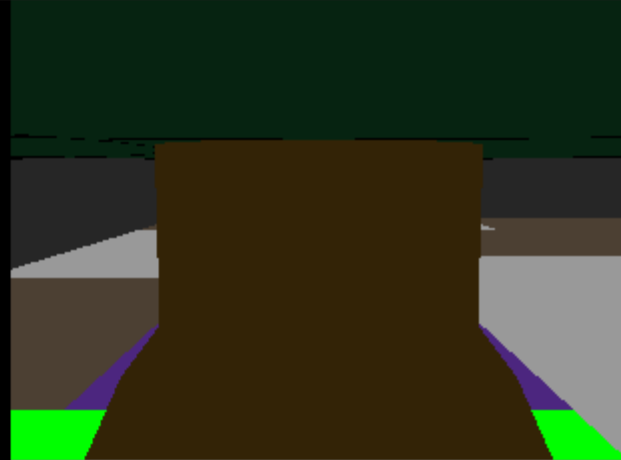
## ./Belkan





Practica 2: Curso 20/21





Nuevo Juego

Visualizacion

☒ Vista en 1 Persona
 ☒ Mostrar mapa

Ir a...

Fila 3

Columna 3

☒ Objetivo 1  
☐ Objetivo 2  
☐ Objetivo 3

Control

Pasos 10

Retardo 1

Paso

Ciclo

Ejecucion

Salir

Informacion

Tiempo restante: 3000  
 Bateria: 3000  
 Posicion actual: (F 25 , C 11 | oeste)  
 Ultima Accion: actIDLE  
 Tiempo Consumido: 0.000  
 ZAPATILLAS OFF | BIKINI OFF  
 Objetivos: 0  
 \*\*\*\*\* Vision \*\*\*\*\*  
 J \_ \_ \_ \_ \_ \_ \_ \_ \_ \_  
 B B B S P T D S M P P T T S M M

## • Sistema *batch*:

**./BelkanSG mapas/mapas30.map 1 0 4 5 1 3 20**

- fichero de mapa
- semilla generador de números aleatorios
- Nivel (0, 1, 2, 3 o 4)
- fila origen
- columna origen
- Orientación (0=norte, 1=este, 2=sur, 3=oeste)
- pares de (fila, columna) destino

En caso del nivel 4, si se queda sin destinos, los generará al azar.

Se incluye esta opción para acelerar la ejecución completa de una simulación por un lado y por otro, para que el entorno gráfico no sea un inconveniente a la hora de depurar errores.

Nota: Esta es la forma de invocarlo desde una terminal de linux o windows. Obviamente, desde el entorno CodeBlocks también se puede hacer esta ejecución usando las ventanas.

Al finalizar la ejecución, se nos proporciona el tiempo consumido, la cantidad de batería con la que terminó la simulación, el número de colisiones que hemos sufrido (por obstáculos u otros agentes), la cantidad de muertes (si vale 1 es que terminó cayendo a un precipicio) y la cantidad de destinos alcanzados.

```
Mision alcanzada!  
Instantes de simulacion no consumidos: 2969  
Tiempo Consumido: 0.003062  
Nivel Final de Bateria: 1374  
Colisiones: 0  
Muertes: 0  
Objetivos encontrados: 1
```

- Sistema *batch* y Entorno Gráfico:

`./Belkan mapas/mapas30.map 1 0 4 5 1 3 20`

- Se puede usar una versión combinada del sistema batch con el entorno gráfico.
- La interpretación y orden de los parámetros en la llamada en línea es igual que cuando se ejecuta BelkanSG.
- Esta sería una manera simple de fijar las condiciones iniciales del simulador sin tener que navegar por el sistema de ventanas.



- Introducción
- Los extraños mundos de Belkan
- Objetivo
- Software
- **Método de evaluación y entrega de prácticas**



- Se pide desarrollar un programa (modificando el código de los ficheros del simulador 'jugador.cpp' y 'jugador.hpp') con el comportamiento requerido para el agente.
- Estos dos ficheros deberán entregarse mediante la plataforma web de la asignatura, en un fichero ZIP (de nombre "practica2.zip" que no contenga carpetas ni sub-carpetas).
- El archivo ZIP deberá contener solo el código fuente de estos dos ficheros con la solución del alumno así como un fichero de documentación en formato PDF que describa el comportamiento implementado con un máximo de 5 páginas.

- No es necesario tener todos los niveles de la práctica para que sea evaluada. La valoración de cada nivel y las restricciones para ser evaluado se muestran en la siguiente tabla:

Nivel	Puntuación	Requisito
0	0	Terminar el nivel de demo
1	2	Tener correcto el nivel 0
2	3	Tener correcto el nivel 1 y 0
3	2	Tener correcto el nivel 2, 1 y 0
4	3	Tener correcto el nivel 2, 1 y 0 (no es necesario el nivel 3 )

- Los niveles del 0 al 3 se evalúan si se satisfacen las condiciones que se le exigen a cada nivel.
- Para que el nivel 3 sea evaluable es necesario haber hecho los niveles 0, 1 y 2.
- Para el nivel 4 no es necesario entregar o que esté correcto el nivel 3, pero si todos los niveles anteriores.

- El nivel 4 puede usar uno de los algoritmos de búsqueda ya implementados en los niveles anteriores o bien el estudiante puede definir uno distinto.
- El objetivo es maximizar el número de objetivos en el tiempo, batería e instantes de simulación que se ofrecen.
- Para valorar este nivel se realizarán pruebas sobre distintos mapas con distintas configuraciones de posiciones iniciales y de listas de objetivos. En base al resultado de esas pruebas se otorgará una calificación entre 0 y 3 puntos.
- La nota final se calculará sumando los puntos obtenidos en cada nivel, teniendo en cuenta las restricciones descritas en la tabla.

**Fecha límite entrega**

10 de Mayo hasta las 23:00 horas

**Cuestionario de autoevaluación**

Disponible desde el 11 de Mayo

13 de Mayo hasta las 23:00 horas

## Esta práctica es **INDIVIDUAL**

En el caso de detectar prácticas copiadas, los involucrados (tanto el que se copió como el que se ha dejado copiar) tendrán suspensa la asignatura.