

**Introducció**

En aquesta pràctica es pretén descriure i programar el comportament d'un agent intel·ligent que per poder fer totes les seves funcions i sobreviure ha d'agafar el màxim possible d'aliment (que li donarà les forces necessàries). El metabolisme de l'agent és molt ràpid i molt delicat. Que sigui ràpid vol dir que està contínuament consumint l'aliment i que sigui delicat vol dir que només pot digerir un tipus determinat de menjar ja que és al·lèrgic als menjars dels altres agents. Si toca un tipus d'aliment que no és l'adequat afectarà negativament a la seva salut.

**L'entorn**

L'entorn serà un recinte tancat amb obstacles generats aleatòriament i en el què també hi pot haver altres agents.

A l'entorn, a més d'altres agents, es poden trobar diferents objectes identificats pels seus colors:

- Els objectes del mateix color que l'agent representen l'aliment d'aquest agent, el recurs bàsic que el pot alimentar.
- Els objectes de diferent color que l'agent representen l'aliment d'un altre agent i que el primer agent no pot digerir i l'afecta negativament.
- Els objectes blancs són escuts de protecció, que si s'activen impedeixen el contacte amb un altre objecte i protegeixen l'agent de qualsevol contacte.

**L'objectiu**

La missió del vostre agent serà trobar i recollir els aliments adequats pel seu metabolisme i evitar aquells que no pot digerir correctament.

Per tant, en aquesta pràctica s'ha de programar el comportament d'un agent amb l'objectiu d'agafar i conservar, amb un temps limitat, el major número possible d'aliments (recursos), utilitzant tots els mitjans de que disposi, i d'evitar tocar objectes que li puguin fer mal (qualsevol altre tipus d'aliment).

**L'agent**


L'agent pot obtenir informació de l'entorn mitjançant els seus sensors i pot actuar mitjançant els seus actuadors.








Morfològicament, els agents són quadrats de 25x25 píxels i disposen de:

- Mecanismes de desplaçament i gir amb velocitats variables que permeten que l'agent es mogui per l'entorn.
- Un sistema de tres visors configurables que són l'única eina per veure l'entorn.
- Un dispositiu que permet viatjar a l'hiperespai, que fa que l'agent desaparegui i torni a aparèixer en un altre lloc aleatori.
- Un mecanisme de defensa o escut, que si s'activa permet aïllar l'agent de qualsevol altre objecte de l'entorn.
- Un mecanisme de llançament d'aliments d'abast limitat.
- La fisiologia de l'agent només pot suportar l'impacte de 5 llançaments.

S'ha de tenir en compte que:

- Si es llança un aliment contra un altre agent i el fer, li produeix al·lèrgia al ferit i per curar-se necessita menjar (perd un recurs d'aliment i part de la seva salut).
- Si un agent es queda sense aliment, no pot utilitzar els seus efectors (ni de moviment ni de llançament).
- Per activar un escut n'ha de tenir (a la configuració inicial o agafats de l'entorn).
- Per activar l'hiperespai n'ha de tenir (a la configuració inicial).

Per poder configurar les característiques inicials del vostre agent tendreu 1000  (unitat monetària del món dels agents). Aquestes característiques no es podran modificar durant la simulació, exceptuant el número d'escuts, que es poden agafar de l'entorn. Les opcions de configuració són les següents:

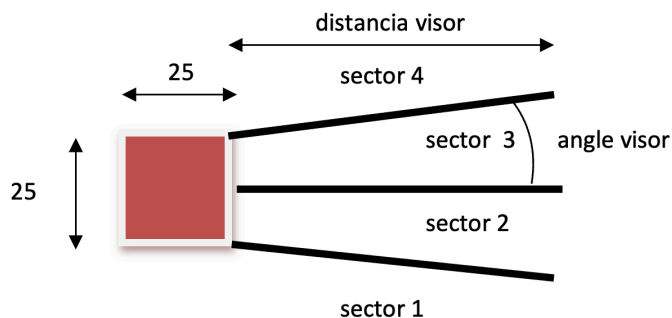
- Velocitat angular (per defecte 4): 10  cada pujada d'una unitat
- Velocitat lineal (per defecte 4): 10  cada pujada d'una unitat
- Distància d'abast dels visors (per defecte 200m): cada 50 metres costen 100 
- Angle dels visors (per defecte 45°): cada 10 graus de modificació costen 25 
- Número llançaments: cada llançament val 5 
- Número escuts: cada escut costa 25 
- Número hiperespais: cada hiperespai val 50 

## L'entorn de programació

**Nota important:** només es poden utilitzar les instruccions de l'entorn que estan definides en aquest llistat o explícitament indicades pel professor tot i que en hi pot haver altres de visibles. Un agent que incompleixi aquesta restricció no serà vàlid.

La pràctica es programarà en JAVA utilitzant l'entorn que podeu descarregar d'AulaDigital. S'haurà de programar el comportament del vostre agent dins una classe que obligatòriament ha de tenir el nom "Bitxo" seguit del número de grup de pràctiques que tendreu assignat. Per exemple, el grup 1 implementarà la classe Bitxo1.java, el grup 2 implementarà la classe Bitxo2.java i així per a cada grup.

L'agent necessita aliments per poder utilitzar els seus efectors. Inicialment disposa d'una petita reserva però s'ha d'anar agafant menjar per mantenir les funcions vitals.



## Accions

Les accions de que disposa l'agent venen donades per les següents funcions:

1. Pel control del moviment:

- **public void endavant()** // endavant amb velocitat lineal
- **public void enrere()** // enrere amb velocitat lineal
- **public void esquerra()** // gira a l'esquerra amb velocitat angular
- **public void dreta()** // gira a la dreta amb velocitat angular
- **public void atura()** // l'agent anul·la tots els moviments
- **public void gira(int graus)** // gira els graus especificats en sentit antirellotge
- **public boolean vaEndavant()** // va cap endavant ?
- **public boolean vaEnrere()** // va cap enrere ?
- **public boolean vaEsquerra()** // està girant a l'esquerra ?
- **public boolean vaDreta()** // està girant a la dreta ?
- **public boolean estaAturat()** // està aturat ?
- **public void activaEscut()** // Activa l'escut
- **public void hyperespai()**  
Provoca la desaparició i posterior aparició de l'agent en un lloc aleatori de l'entorn.  
La seva activació produeix un consum considerable de recursos de l'agent.

2. Pel control de la visió:

- **public void mira(Objecte obj)**  
Si qualsevol objecte es troba dins els sectors 2 o 3 (sectors centrals), llavors l'agent pot mirar directament a l'objecte. Si l'objecte està fora (sectors 1 i 4) llavors no el pot mirar.

3. Per llançar objectes:

- **public void llança()**  
Dispara menjar en la direcció de desplaçament, únicament hi pot haver un llançament a cada instant, no es pot tornar a llançar fins que l'objecte no acaba la seva trajectòria (abast màxim, impacte amb mur o impacte amb un altre agent). L'efecte del llançament sobre el rival redueix la seva provisió d'aliment i li resta un recurs. Un agent, al cinquè impacte ja no ho pot suportar més i desapareix.

## Percepcions

Les percepcions de l'entorn es poden obtenir a partir de la classe Estat, que descriu en detall l'estat en que es troba l'entorn. Es necessita declarar un objecte de tipus Estat, que s'inicialitza amb la crida al mètode "estatCombat()":

- **Estat estatCombat();**  
Aquesta crida actualitza els valors de les variables que defineixen l'estat en que es troba l'entorn (percepcions) i que es descriuen a continuació:

```
boolean hiperEspaiActiu; // veritat si estic en hyperespai
boolean enCollisio;      // veritat si està aturat per col·lisió amb un obstacle
boolean llançant;        // veritat si estic llançant un objecte
```

```

boolean  veigAlgunEnemic;
boolean  veigAlgunRekurs;
boolean  veigAlgunEscut;
boolean  escutActivat;

int       impactesRebut; // número de llançaments que han ferit l'agent
int[]     impactesRival; // número de ferides als rivals
int       forces;        // aliments que té
int       escuts;        // Escuts que tenc
int       hiperespais;   // Hiperespais disponibles
int       recursosAgafats; // Recursos que ha agafat

int       numObjectes;   // número d'objectes que veu
Objecte[] objectes ;    // Informació del objectes que veu

De cada objecte podem consultar:
    agafaSector()       // Retorna el sector on es veu
    agafaDistancia()    // Retorna la distància a la que es troba
    agafaTipus()        // Estat.AGENT, Estat.ESCUT o si no és un recurs
                        // en aquest darrer cas, el recurs està codificat com
                        // 100+id del recurs (ex: el 104 és un recurs de l'agent 4).

int       llançaments;   // Llançaments que pot fer
boolean   llançamentEnemicDetectat; // veritat si ens estan disparant i ho veim
int       distanciaLlançamentEnemic; // distancia de la bala enemiga
                        // si s'ha detectada

boolean[3] estatVisor;   // si veu alguna cosa cada visor
double[3]  distanciaVisors; // a quina distancia ho veu
int[3]     objecteVisor;  // que veu (0-pared, 1-bitxo,2-recurs,5-escut, -1 res)
int[3]     indexNau;      // identificador del que veu el visor

int[]      punts;         // puntuació de cada un dels bitxos
int        numBitxos;     // total de bitxos que hi ha a l'entorn
int        id;            // el número del meu bitxo

```

## Definició de la vostra classe

A la vostra classe, anomenada BitxoN, on N és el vostre número de grup, heu d'implementar tres mètodes:

- **public BitxoN(Agents pare) {**  
     super(pare, "Pepet", "imatges/nomImatge.gif");  
   }  
     ○ És el constructor de la classe. El nom de la classe, com s'ha explicat abans, és la paraula "Bitxo" seguida del vostre número de grup (en aquest exemple hem suposat el grup N).  
     ○ El primer paràmetre és forçosament la classe que es rep com a argument.  
     ○ El segon paràmetre del constructor és el nom que voleu que tenguí el vostre agent (ha de ser diferent al que teniu per defecte "Pepet"). Aquest nom serveix per identificar al vostre agent durant la simulació i hauríeu de procurar que fos únic.

- El tercer paràmetre és el nom de la imatge que s'utilitzarà com a icona del vostre agent. S'ha de guardar dins la carpeta imatges.
- **public void inicia()**
  - En aquesta funció podeu inicialitzar el vostre agent. S'executa sempre una única vegada a l'inici de cada simulació.
  - És on establireu la configuració inicial del vostre agent amb el mètode:
 

```
public int atributsAgent(int v, int w, int dv, int av, int ll, int es, int hy)
```

on:

    - **v** és la velocitat lineal
    - **w** és la velocitat angular
    - **dv** és la distància de visió
    - **av** és l'angle dels visors
    - **ll** és el número de llançaments
    - **hy** és el número d'hiperespais
- **public void avaluaComportament()**
  - En aquesta funció heu de definir tot el comportament del vostre agent (utilitzant els valors obtinguts amb les percepcions i fent les accions corresponents a la vostra estratègia).
  - S'avalua el comportament de tots els agents simultàniament. Teniu en compte que la funció avaluaComportament es crida un cert número de vegades per segon i que únicament després de cada cridada s'actualitza l'estat de la simulació. Això vol dir que heu d'esperar a una propera cridada per a veure els possibles canvis que el vostre codi hagi produït (no té molt de sentit tenir iteracions a l'interior de les quals hi hagi reiterades cridades a funcions de l'entorn ja que, realment, només la darrera cridada que es produeixi afectarà a l'estat de l'entorn). Si voleu repetir una acció, ho heu de fer al llarg de diverses cridades.

## Instruccions d'instal·lació de l'entorn d'agents




Per instal·lar l'entorn amb el què treballarem heu de baixar primer d'Aula Digital la carpeta comprimida "Entorn".

L'arxiu comprimit conté:

- Una carpeta "src"
- Una carpeta "classes"

Seguiu en ordre les següents passes:

1. Creau un projecte nou java, amb el nom que es vulgui i sense "main class".
2. Copiau el contingut de la carpeta "src" dins la carpeta "src" del projecte creat.
3. Compilar el projecte (donarà errors que no troba les classes, però crearà la carpeta "classes" que necessitam).
4. Copiau el contingut de la carpeta "classes" dins la carpeta "classes" del projecte creat.
5. A la carpeta "libraries" del projecte a netbeans, boto dret i triau "add JAR/folder" i seleccionau l'arxiu "agents.jar" de la carpeta classes/agents".

6. Tornau a compilar, però sempre amb , **NO** amb , ja que això borraria tot el què teniu dins la carpeta “classes”. Ara ja no hauria de donar errors.
7. Per executar l'aplicació **NO** heu d'utilitzar el , ho heu de fer des de la consola (cmd a Windows) amb appletviewer, que ja teniu instal·lat si teniu el JDK de java.
8. A la cònsola, heu d'executar la comanda: “appletviewer Exemple.html”. Pot passar que vos digui “comanda no trobada”, en aquest cas es pot deure a dues coses, no hi ha el appletviewer instal·lat, o no teniu ben posat el “PATH” del sistema. Per verificar si el teniu, podeu mirar dins la carpeta de JAVA, dins la subcarpeta “bin”. Si hi és, llavors heu de posar el camí de la carpeta “bin”, dins la variable d'entorn “PATH”, del sistema.
9. Si tot és correcta, s'obrirà l'Applet de l'entorn d'agents.
10. Per a qualsevol dubte, utilitzau el meu correu electrònic [ramon.mas@uib.es](mailto:ramon.mas@uib.es)

## Entorn manual

Per facilitar el desenvolupament i proves dels agents, l'entorn es pot utilitzar en mode manual, de manera que es pot actuar sobre els agents amb el teclat. La següent taula descriu les funcionalitats disponibles.

Acció	Bitxo1	BitxoN (“>” per canviar)
Mode manual/autom.	l	z
Mode manual/autom. tots	Z	
endavant	w	fletxa amunt
enrere	x	fletxa avall
esquerra	a	fletxa esquerra
dreta	d	fletxa dreta
disparar	s	espai
hiperespai	h	H
pinta pistes	i (mostra/amaga pistes) I (mostra/amaga forces i recursos agafats) L (mostra/amaga lletres) U (mou/fixa lletres)	
baixa velocitat angular	1	9
puja velocitat angular	2	0
baixa velocitat lineal	3	y
puja velocitat lineal	4	u
disminuir distancia visió	5	7
augmentar distancia visió	6	8
disminuir angle visor	j	f
augmentar angle visor	k	g
mira l'objecte més proper		M
Reinicia combat	R	
Carrega força/aliments	F	
Carrega llançaments	B	
Mostra estat per consola	V,v (o clic damunt un agent)	

### Mecanisme d'Entrega:

- A la tasca habilitada a aulaDigital s'ha d'entregar un arxiu comprimit que contengui:

- Un document pdf de com a màxim 5 planes que descrigui el comportament del vostre bitxo i les particularitats que hagueu pogut implementar. El document ha de resumir el comportament que s'espera de l'agent.
- La classe Bitxo (*.java* i *.class*) i les classes (*.java* i *.class*) addicionals que hagueu implementat.
- La imatge personalitzada en format *gif* (25x25 píxels) per al vostre agent (la que heu indicat al constructor).
- En el constructor del vostre agent heu de posar un nom identificatiu únic del vostre agent (és el que es mostrarà a la pantalla de la simulació)

**Nota:**

- La pràctica s'ha de fer en grups de tres persones