



**Universidad Autónoma de Nuevo León
Facultad de Ciencias Físico Matemáticas**

**Inteligencia Artificial
Maestro: Juan Pablo Rosas Baldazo**

**Producto Integrador de Aprendizaje
“Mundo de Wumpus”**

Equipo 5:

**Raymundo Ruiz Ledezma 1458927
Chelsea Michelle Olivares Martínez 1795179
Héctor Gerónimo Peña Butt 1887915**

DESCRIPCION

• Descripción

El juego es desarrollado en un tablero de dimensión 4x4 , generalmente cuadricular. Encontraremos ciertos elementos que se distribuirán por todo el tablero, éstos serán:

- Aventurero(agente)
- Wumpus
- Tesoros.
- Pozos o Agujeros.
- Brisa.
- Hedor.
- Hedor y Brisa.
- Casillas de Entrada y Salida.

El objetivo del juego es sencillo. El Agente debe de buscar por el tablero un tesoro y salir sin ser devorado por el Wumpus o caer en ningún pozo.

Las casillas que el diseñador puede colocar son:

- Casilla de Entrada.
- Casilla de Salida.
- Casilla del Wumpus.
- Casillas de Pozos.
- Casillas de Tesoros.

Estas casillas son impuestas libremente por el diseñador, sin embargo las restantes se colocan siguiendo unas normas.

1. Casilla de Hedor: son colocadas en las casillas adyacentes al Wumpus.
2. Casilla de Brisa: son colocadas en las casillas adyacentes a Pozos.
3. Casilla de Hedor y Brisa: son las resultantes de la combinación de las anteriores, ya que es posible esta posibilidad.

PEAS

- **Medida de rendimiento**

- Recibe +1000 por recoger oro
- Costo de -1000 por caer en un pozo o ser devorado por Wumpus (¡JUEGO TERMINADO!)
- Costo de -1 por cada acción realizada
- Costo de -10 por usar la única flecha

- **Ambiente**

- Cuadrícula de habitaciones 4x4
- El agente comienza en el cuadrado [1,1]
- Wumpus y ubicaciones de oro elegidas al azar
- La probabilidad de que el cuadrado sea un pozo es .2
 - ♦ [0 = no,..., 0.5 = tal vez,..., 1 = sí]

- **Actuadores**

- Avanzar, girar a la izquierda, girar a la derecha
 - ♦ Nota: muere si ingresa al pozo o al cuadrado wumpus vivo
- Agarrar (oro)
- Disparar flecha
 - ♦ Mata wumpus si se enfrenta a su cuadrado

- **Sensores**

- Nariz: los cuadrados adyacentes a los wumpus "huelen mal"
- Piel / cabello: los cuadrados adyacentes al hoyo emiten "brisa"
- Ojo: "Brillante" si y solo si hay oro en la misma casilla
- Percepciones: [hedor, brisa, brillo]

- **Caracterización del Mundo Wumpus**

- **¿Es el mundo determinista?**

- Sí, resultados exactamente especificados

- **¿Es el mundo completamente observable?**

- No, solo percepciones locales

- **¿Es el mundo estático?**

- Sí, los Wumpus y los hoyos no se mueven (¡aunque sería interesante!)

- **¿Es el mundo discreto?**

- Sí, bloques / celdas

ESPACIO DE ESTADOS PARA LA SIGUIENTE CONFIGURACION

Estado Inicial



Locación: (3,0)

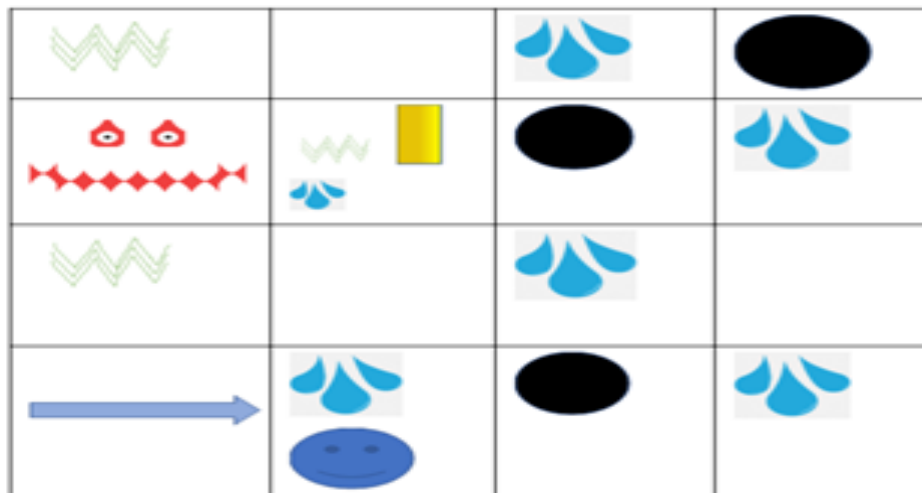
Percepción:

Hedor	Brisa	Resplandor(Oro)	Golpe (Muro)	Grito(Wumpus)
¬ hedor	¬ brisa	¬ resplandor	¬ golpe	¬ grito

Inferencia: el agente concluye que en las casillas (3,1) y (2,0) no hay peligro.

Acción: Se mueve a la casilla (3,1)

Estado 1:



Locación:(3,1)

Hedor	Brisa	Resplandor (Oro)	Golpe (Muro)	Grito (Wumpus)
¬ hedor	Brisa	¬ resplandor	¬ golpe	¬ grito

Inferencia: la brisa indica que puede haber un pozo en (3,2) y (2,1)

acción: regresar al inicio (3,0) para tratar de ir a una casilla segura.

Estado 2:



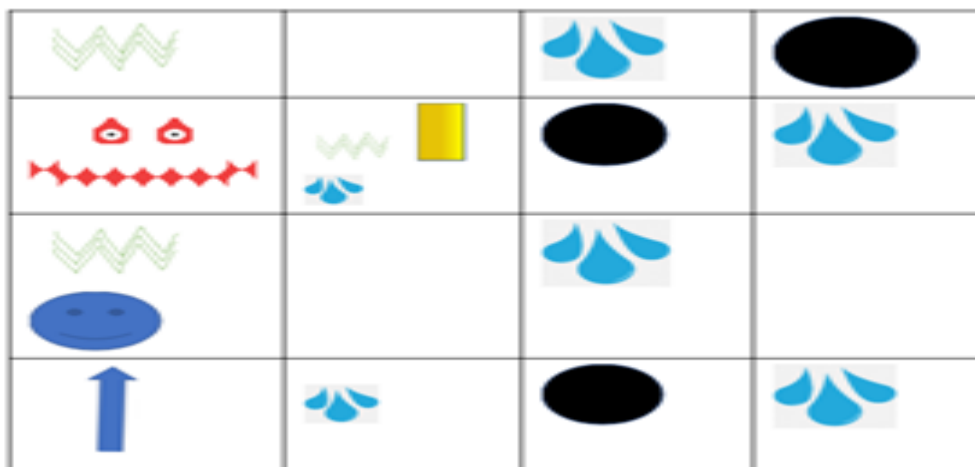
Locación: (3,0)

Hedor	Brisa	Resplandor(Oro)	Golpe (Muro)	Grito(Wumpu s)
¬ hedor	¬ brisa	¬ resplandor	¬ golpe	¬ grito

Percepción:

Acción: el agente se mueve hacia arriba al (2,0).

Estado 3:

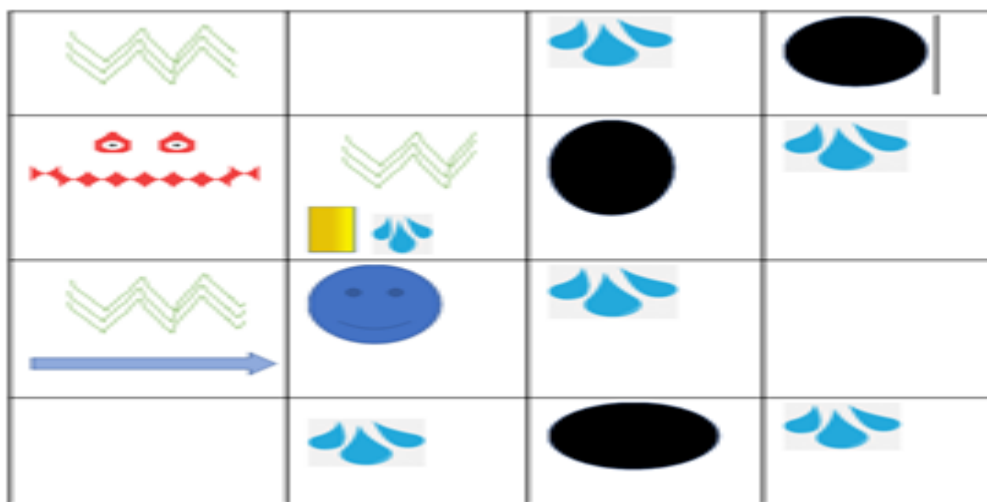


Locación:(2,0)

Hedor	Brisa	Resplandor(Oro)	Golpe (Muro)	Grito(Wumpus)
Hedor	¬ brisa	¬ resplandor	¬ golpe	¬ grito

Percepción:

Estado 4

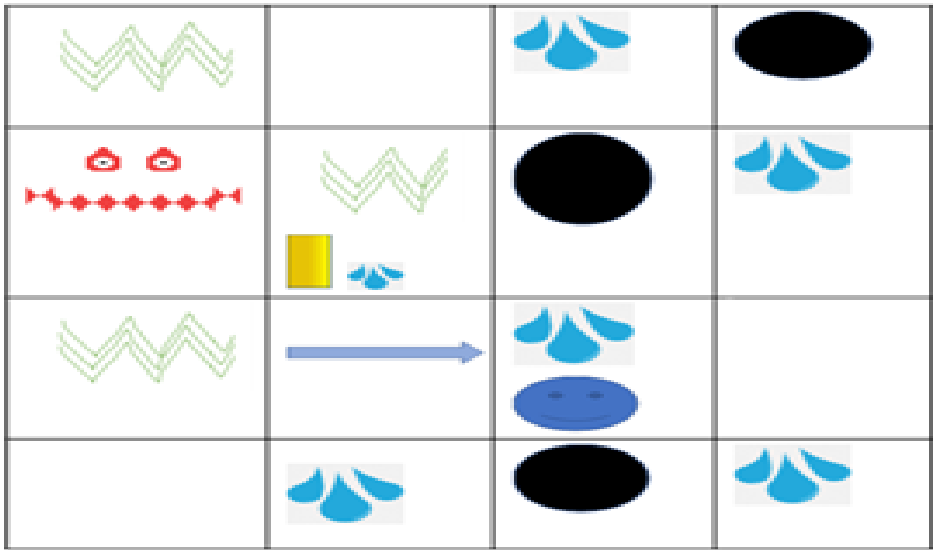


Locación:(2,1)

Hedor	Brisa	Resplandor(Oro)	Golpe (Muro)	Grito(Wumpus)
¬ hedor	¬ brisa	¬ resplandor	¬ golpe	¬ grito

Percepción:

Estado 5

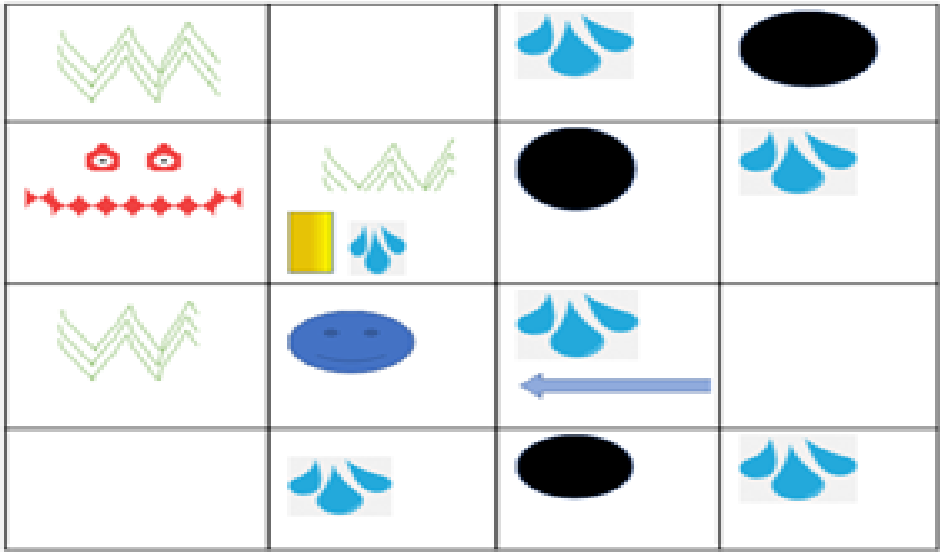


Locación:(2,2)

Hedor	Brisa	Resplandor(Oro)	Golpe (Muro)	Grito(Wumpus)
↪ hedor	Brisa	↪ resplandor	↪ golpe	↪ grito

Percepción:

Estado 6

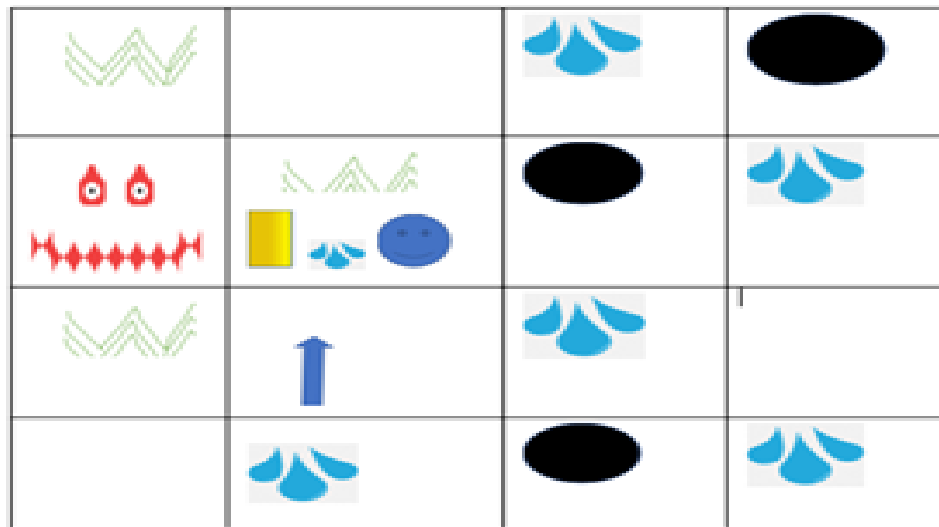


Locación: (2,1)

Hedor	Brisa	Resplandor(Oro)	Golpe (Muro)	Grito(Wumpus)
↪ hedor	↪ brisa	↪ resplandor	↪ golpe	↪ grito

Percepción:

Estado Final



Locación:(1,1)

Hedor	Brisa	Resplandor(Oro)	Golpe (Muro)	Grito(Wumpus)
hedor	Brisa	resplandor	¬ golpe	¬ grito

Percepción:

Pseudocódigo:

Inicio

```

World[4][4] , i,j = 3,0
Print(posiciones permitidas)
Vi= read(i)
Vj=read(j)
Ai,aj= aprendizajeAgente(world,vi,vj)
While(ai>=0)
    Vi,vj = movimientos(ai,aj)
    If(vi != -5 and vj !=-5)
        Ai,aj = aprendizajeAgente(world,vi,vj)
    Else
        Print(Posicion invalida)
If( ai== -5)
    Print(Gameover)
Else
    Print(Ganaste)
    
```


En Código:

Función que nos indica las percepciones del agente en una determinada posición.

```
jupyter WUMPUS_PIA Last Checkpoint: hace unos segundos (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

In [9]: def aprendizajeAgente(world,i,j):
        if (world[i][j]=='hedor'):
            agi,agj=i,j
            print("\nEl agente se encuentra en "+str(agi)+" "+str(agj))
            print("Puede oler el Hedor")
            return agi,agj

        elif (world[i][j]=='brillo'):
            agi,agj=i,j
            print("\nEl agente se encuentra en "+str(agi)+" "+str(agj))
            print("Puedo ver el Brillo")
            return agi,agj

        elif (world[i][j]=='pozo'):
            agi,agj=i,j
            print("\n El agente se encuentra en "+str(agi)+" "+str(agj))
            print("Haz caído en un Pozo")
            return -5,-5

        elif (world[i][j]=='oro'):
            agi,agj=i,j
            print("\nEl agente se encuentra en "+str(agi)+" "+str(agj))
            print("Haz encontrado el Oro")
            return -4,-4

        elif (world[i][j]=='brisa'):
            agi,agj=i,j
            print("\nEl agente se encuentra en "+str(agi)+" "+str(agj))
            print("Se siente la brisa")
            return agi,agj

        elif (world[i][j]=='ok'):
            agi,agj=i,j
            print("\nEl agente se encuentra en "+str(agi)+" "+str(agj))
            print("Estado seguro")
            return agi,agj

        elif (world[i][j]=='wumpus'):
            agi,agj=i,j
            print("\nEl agente se encuentra en "+str(agi)+" "+str(agj))
            print("Te ha comido el Wumpus")
            return -5,-5

    else: #si la cueva esta segura
        agi,agj=i,j
        print("\nEl agente se encuentra en "+str(agi)+" "+str(agj))
        return agi,agj
```



```
In [10]: def movimientos(agi,agj):
'''Función que reciba las entradas de las fila y columnas y te muestra los movimientos legales'''
if(agi==0 and agj==0):
    print("\nPuedes ir a la cueva "+str(agi+1)+" "+str(agj)) #te puedes mover hacia abajo
    print("Puedes ir a la cueva "+str(agi)+" "+str(agj+1)) #te puedes mover a la derecha
    agvi=int(input("\nIngresa el numero de fila => "))
    agvj=int(input("Ingresa el Número de Columna => "))
    if(agvi==agi+1 and agvj==agj or agvi==agi and agvj==agj+1):
        return agvi,agvj
    else:
        return -5
elif(agi==3 and agj==0):
    print("\nPuedes ir a la cueva "+str(agi-1)+" "+str(agj)) #te puedes mover a la izquierda
    print("Puedes ir a la cueva "+str(agi)+" "+str(agj+1)) #te puedes mover a la derecha
    agvi=int(input("\nIngresa el numero de fila => "))
    agvj=int(input("Enter input for column => "))
    if(agvi==agi-1 and agvj==agj or agvi==agi and agvj==agj+1):
        return agvi,agvj
    else:
        return -5
elif(agi==3 and agj==3):
    print("\nPuedes ir a la cueva "+str(agi-1)+" "+str(agj)) #te puedes mover hacia arriba
    print("Puedes ir a la cueva "+str(agi)+" "+str(agj-1)) #te puedes mover a la izquierda
    agvi=int(input("\nIngresa el numero de fila => "))
    agvj=int(input("Enter input for column => "))
    if(agvi==agi-1 and agvj==agj or agvi==agi and agvj==agj-1):
        return agvi,agvj
    else:
        return -5
elif(agi==0 and agj==3):
    print("\nPuedes ir a la cueva "+str(agi+1)+" "+str(agj)) #te puedes mover hacia abajo
    print("you can go at "+str(agi)+" "+str(agj-1)) #te puedes mover a la izquierda
    agvi=int(input("\nIngresa el numero de fila => "))
    agvj=int(input("Ingresa el numero de columna => "))
    if(agvi==agi+1 and agvj==agj or agvi==agi and agvj==agj-1):
        return agvi,agvj
    else:
        return -5,-5
elif(agi==1 and agj==0 or agi==2 and agj==0 or agi==3 and agj==0):
    print("\nPuedes ir a la cueva "+str(agi+1)+" "+str(agj)) #te puedes mover hacia abajo
    print("Puedes ir a la cueva "+str(agi)+" "+str(agj+1)) #te puedes mover a la derecha
    agvi=int(input("\nIngresa el numero de fila => "))
    agvj=int(input("Ingresa el numero de columna => "))
    if(agvi==agi+1 and agvj==agj or agvi==agi and agvj==agj+1):
        return agvi,agvj
    else:
        return -5,-5
elif(agi==0 and agj==3 or agi==1 and agj==3 or agi==2 and agj==3 or agi==3 and agj==3):
    print("Puedes ir a la cueva "+str(agi+1)+" "+str(agj)) #te puedes mover hacia abajo
    print("Puedes ir a la cueva "+str(agi)+" "+str(agj-1)) #te puedes mover a la izquierda
    agvi=int(input("Ingresa el numero de fila => "))
    agvj=int(input("Ingresa el numero de columna => "))
    if(agvi==agi+1 and agvj==agj or agvi==agi and agvj==agj-1):
        return agvi,agvj
    else:
        return -5,-5
```

Main:

```
In [12]: #Aquí comienza el Juego:

world=[ ['hedor','ok','brisa','pozo'],
        ['wumpus','oro','pozo','brisa'],
        ['hedor','ok','brisa','ok'],
        ['agente','brisa','pozo','brisa'] ]

agi,agj=3,0 #posición inicial del agente

print("\n\nEl agente está inicialmente en "+str(agi)+" "+str(agj))
print("\nPuedes ir a la cueva "+str(agi)+" "+str(agj+1))
print("Puedes ir a la cueva "+str(agi-1)+" "+str(agj))

agvi=int(input("Ingresa el número de fila => "))
agvj=int(input("Ingresa el número de la columna => ")) #se toma el valor de la fila y columna
if(agvi==3 and agvj==1 or agvi==2 and agvj==0):
    agi,agj=aprendizajeAgente(world,agvi,agvj) #esos valores de i y j se toman como parametros
else:
    print("Posición inválida") #tiene que ser obligatorio sólo esas dos opciones de posiciones
```

```

while(agi>=0):
    agvi,agvj=movimientos(agi,agj)
    if(agvi!=-5 and agvj!=-5):
        agi,agj=aprendizajeAgente(world,agvi,agvj)
    else:
        print("\nPosición inválida")
if(agi!=-5):
    print("\nGame over !!!")
else:
    print("\nHaz ganado ya obtuviste el ORO ")

```

Jugando:

```

El agente está inicialmente en 3,0

Puedes ir a la cueva  3 1
Puedes ir a la cueva  2 0
Ingresa el número de fila => 3
Ingresa el número de la columna => 1

```

```

El agente se encuentra en 3,1
Se siente la brisa

```

```

Puedes ir a la cueva  3 2
Puedes ir a la cueva  3 0
Puedes ir a la cueva  2 1

```

```

Ingresa el numero de fila => 3
Ingresa el numero de columna => 0

```

```

El agente se encuentra en 3,0

```

```

Puedes ir a la cueva  2 0
Puedes ir a la cueva  3 1

```

```

Ingresa el numero de fila => 2

```

```

Enter input for column => 

```

In []:

In []: