Modelación de sistemas multiagentes con gráficas computacionales

Graficación utilizando Pygame

Pablo Andrés Martínez

A01252489

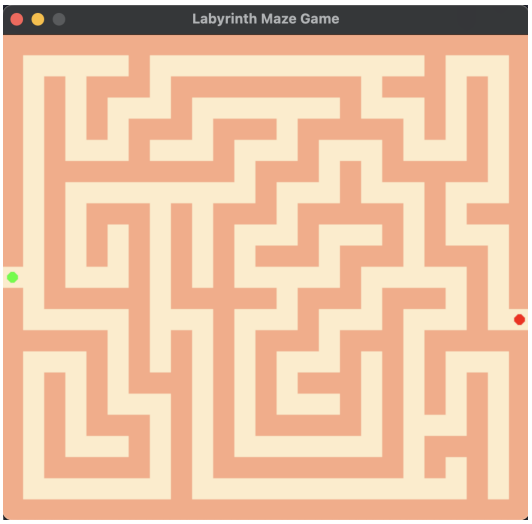
11/08/23

Liga de github:

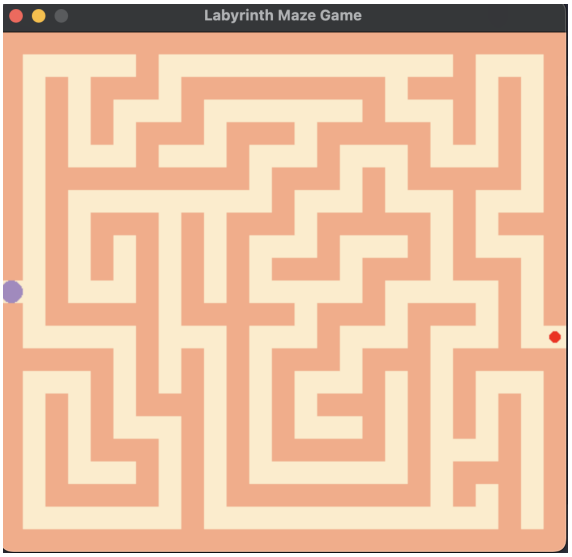
https://github.com/pabloamtzs/TC2008B-PAMS/tree/3e0f5e2238d6f432e3bc294f88faaff622ca58ae/graficacion-laberinto

Este proyecto consiste en la creación de un entorno tipo laberinto utilizando la librería pygame. El objetivo es familiarizarse con la creación de entornos para la simulación de agentes inteligentes.

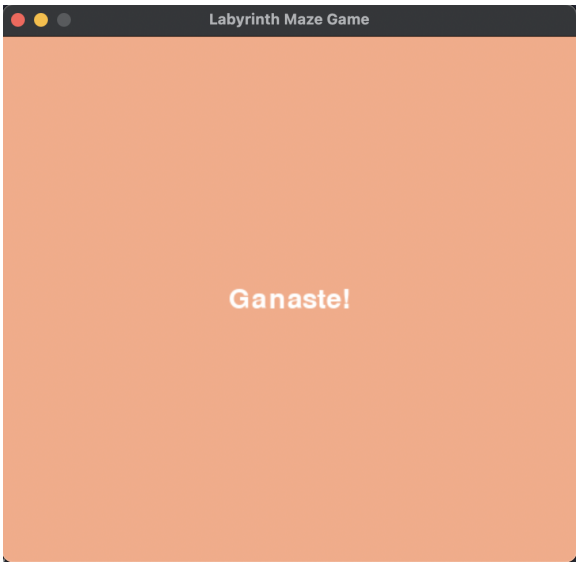
Capturas de Pantalla



Laberinto sin jugador. El punto verde es el punto inicial. El punto rojo es la meta



Laberinto con jugador. El punto morado es el jugador



Pantalla de victoria. Se consigue cuando el jugador llega a la meta.

Características del Laberinto

Diseño de Laberinto: El código define un laberinto en forma de matriz utilizando caracteres. Cada celda del laberinto puede ser una pared (X), una celda de inicio (S), una celda de meta (E), o pasillos (0) por los que el jugador puede moverse.

Interacción del Jugador: El jugador puede moverse por el laberinto utilizando las teclas de dirección (arriba, abajo, izquierda, derecha). Se implementa una lógica que impide que el jugador se mueva a través de las paredes.

Detección de Condición de Victoria: El juego detecta cuando el jugador alcanza la celda de meta (E) y muestra un mensaje de victoria en pantalla. Después de mostrar el mensaje, el programa espera durante 2 segundos antes de finalizar.

Representación Visual: Utilizando colores y formas geométricas, el código visualiza el laberinto, la posición del jugador, la celda de inicio y la celda de meta en una ventana gráfica.

Control de Velocidad: El código controla la velocidad de actualización de la pantalla para proporcionar una experiencia de juego suave y fluida.

Flexibilidad del Laberinto: El laberinto se define como una matriz de caracteres, lo que permite a los usuarios modificar y personalizar la estructura del laberinto según sus preferencias.

Por último, el laberinto se puede ejecutar con el siguiente comando:
python graficacion-laberinto.py

Código:

```
import pygame
import sys

pygame.init()

# Set up the display
screen_width = 500
screen_height = 460
screen = pygame.display.set_mode((screen_width, screen_height))
pygame.display.set_caption("Labyrinth Maze Game")

# Colors
white = (255, 255, 255)
black = (0, 0, 0)
```

```
orange = (254,235,201)
orange2 = (252,169,133)
purple = (165,137,193)

# Font
font = pygame.font.Font(None, 36)

# Win condition
win_condition_met = False

# Clock to control frame rate
clock = pygame.time.Clock()

# Define the maze
maze = [
"XXXXXXXXXXXXXXXXXXXXXXXXX",
"X00000X0000000000000X000X",
"X0X0XXX0XXXXXXXXX0XXX0X0X",
"X0X0X000X0000000X000X0X0X",
"X0X0X0XXX0XXX0XXXXX0X0X0X",
"X0X000X000X000X000X000X0X",
"X0XXXXXXXXX0XXX0X0XXXXX0X",
"X0X000000000X000X000X000X",
"X0X0XXX0X0XXX0XXXXX0X0XXX",
"X0X0X0X0X0X000X000X0X000X",
"X0X0X0X0X0X0XXX0XXX0XXX0X",
"S0X000X0X0X00000X00000X0X",
"X0XXXXX0XXXXX0XXX0XXX0XOX",
"X00000X000X000X000X000X0E",
"XXXXX0X0X0X0XXX0XXX0XXXXX",
"X000X0X0X0X0X000X0X0X000X",
"X0X0X0XXX0X0X0XXX0X0X0X0X",
"X0X0X000X0X0X000X0X0X0X0X",
"X0X0XXX0X0X0XXXXX0X000X0X",
"X0X000X0X0X0000000X0XXX0X",
"X0XXXXX0X0XXXXXXXXX0X0X0X",
"X0000000X0000000000000X0X",
"XXXXXXXXXXXXXXXXXXXXXXXXX",
]

# Player settings
cell_size = 20
player_x = 0
player_y = 220
```

```
while True:
for event in pygame.event.get():
if event.type == pygame.QUIT:
pygame.quit()
sys.exit()
elif event.type == pygame.KEYDOWN:
if event.key == pygame.K_UP:
if maze[int((player_y - cell_size) / cell_size)][int(player_x / cell_size)] != "X":
player_y -= cell_size
elif event.key == pygame.K_DOWN:
if maze[int((player_y + cell_size) / cell_size)][int(player_x / cell_size)] != "X":
player_y += cell_size
elif event.key == pygame.K_LEFT:
if maze[int(player_y / cell_size)][int((player_x - cell_size) / cell_size)] != "X":
player_x -= cell_size
elif event.key == pygame.K_RIGHT:
if maze[int(player_y / cell_size)][int((player_x + cell_size) / cell_size)] != "X":
player_x += cell_size

# Check for win condition
if maze[int(player_y / cell_size)][int(player_x / cell_size)] == "E":
win_condition_met = True

# Clear the screen
if win_condition_met:
screen.fill(orange2)
win_message = font.render("Ganaste!", True, white)
screen.blit(win_message, (screen_width // 2 - win_message.get_width() // 2,
screen_height // 2 - win_message.get_height() // 2))
pygame.display.flip()
pygame.time.delay(2100) # Display the message for 2 seconds
pygame.quit()
sys.exit()
else:
screen.fill(orange)

# Draw the maze
for y, row in enumerate(maze):
for x, cell in enumerate(row):
if cell == "X":
pygame.draw.rect(screen, orange2, (x * cell_size, y * cell_size, cell_size,
cell_size))
elif cell == "S":
pygame.draw.circle(screen, (0, 255, 0), (x * cell_size + cell_size // 2, y *
cell_size + cell_size // 2), 5)
```

```
elif cell == "E":
pygame.draw.circle(screen, (255, 0, 0), (x * cell_size + cell_size // 2, y *
cell_size + cell_size // 2), 5)

# Draw the player
pygame.draw.circle(screen, purple, (player_x + cell_size // 2, player_y + cell_size
// 2), 10)

# Update the display
pygame.display.flip()

# Control frame rate
clock.tick(30)
```