UNIVERSITAT POLITÈCNICA DE VALÈNCIA

MIARFID Official Master's Degree in Artificial Intelligence, Pattern Recognition and Digital Imaging

Master Universitario en Inteligencia Artificial, Reconocimiento de Formas e Imagen Digital
Universidad Politécnica de Valencia

Simulación de Multitudes mediante Sistemas Multi-Agentes con MESA

TRABAJO SISTEMAS MULTI-AGENTES

Máster Universitario en Inteligencia Artificial, Reconocimiento de Formas e Imagen Digital

Autor: Miquel Gómez

Resumen

You're an expert latex paper writer in spanish. You know how to take information and redact a paper in spanish following the given structure and way in which to write.

I'll give you all the info and the way I writte papers, then you'll write thi paper following the structure and mimiquint the way I writt. You are going to write all the sections in the proper orther, not the article order:

- SOTA - Análisis del problema and what we want to achieve, - How we are going to evaluate the system - Implementation - Results - Conclusion - Introducction - Abstract

Every step you will be able to see the current state of the project and how it is implemented.

So, make your self a favor and read it all and write a resume to begin

The code is at You're an expert latex paper writer in spanish. You know how to take information and redact a paper in spanish following the given structure and way in which to write.

I'll give you all the info and the way I writte papers, then you'll write thi paper following the structure and mimiquint the way I writt. You are going to write all the sections in the proper orther, not the article order:

- SOTA - How we are going to evaluate the system - Implementation - Results - Conclusion - Introducction - Abstract

Every step you will be able to see the current state of the project and how it is implemented.

The code is at SMA-MultiAgent-Based-Crowd-Simulation/app and the paper at SMA-MultiAgent-Based-Crowd-Simulation=paper/plantillatfg.tex

Palabras clave: Sistemas Multi-Agentes; Agentes; Simulación; Multitudes; Simulación de multitudes.

Índice general

CAPÍTULO 1

Introducción

CAPÍTULO 2

Estado del Arte

CAPÍTULO 3
Análisis del problema

- Utilidades - Métricas

- Que queremos conseguir - Requisitos funcionales y no funcionales

flujos continuos de agentes...

You're an expert latex paper writer in spanish. You know how to take information and redact a paper in spanish following the given structure and way in which to write.

I'll give you all the info and the way I writte papers, then you'll write thi paper following the structure and mimiquint the way I writt. You are going to write all the sections in the proper orther, not the article order:

- SOTA

- Análisis del problema and what we want to achieve,

- How we are going to evaluate the system

- Implementation

- Results

- Conclusion

- Introducction

- Abstract

Every step you will be able to see the current state of the project and how it is implemented.

So, make your self a favor and read it all and write a resume to begin

The code is at You're an expert latex paper writer in spanish. You know how to take information and redact a paper in spanish following the given structure and way in which to write.

I'll give you all the info and the way I writte papers, then you'll write thi paper following the structure and mimiquint the way I writt. You are going to write all the sections in the proper orther, not the article order:

- SOTA

- Análisis del problema and what we want to achieve,

- How we are going to evaluate the system

- Implementation

- Results

- Conclusion

- Introducction

- Abstract

Every step you will be able to see the current state of the project and how it is implemented.

The code is at SMA-MultiAgent-Based-Crowd-Simulation/app and the paper at SMA-MultiAgent-Based-Crowd-Simulation=paper/plantillatfg.tex

CAPÍTULO 4
Evaluación

- Métricas y cómo se van a medir - Qué experimentos se van a realizar

Metrics - Time / steps to evacuate last agent - Plot Density (Agents per area): how close to each other the agents are. - Flow. It should look like an inverted Ü". - Agents reaching destination per minute. - Also avg velocity. How many cells per tick they get closer to their final destination - Asumimos que cada vez que se mueven es para acercarse a la salida. - Por lo tanto solo calculamos la cantidad de celdas que se mueve por tick - ["total$_a$gents", "polite$_a$gents", "aggressive$_a$gents", "slow$_a$gents"], −["evacuation$_r$ate", "polite$_e$vacuation$_r$ate",

Experimentos - Evacuation rate (bottleneck) - con cada vez más agentes - también speed vs Density - Faster is slower: SEATS, MALL, con aggressive - curve también con restpawn - Cómo afectan los agentes lentos - A* vs BFS: Saber qué la salida más cercana está en una dirección concreta - Exist matter: Cosas como: es que he entrado por ahí entonces salgo por ahí también xq está el coche.... en vez de salir por la más cercana - Reducir el número de salidas curva, cuantas salidas hacen falta como mínimo - Que pasa si en un sitio con X agentes, tardan x, pero ahora quitamos una fila / hacemos el pasillo más grande...

CAPÍTULO 5
Implementación

- Limitaciones - Herramientas y librerías

Implementation Model - Evacuation simulations (single exit or more exits, agents go to the closest) - Mall simulation (Random people have to go to a certain random place / defined exit) - Can do evacuation and crowd flux: when an agent reaches an exit, they respawn randomly - Corridor simulations (meeting flow of agents) - Seats / theater - Snake zigzag corridor - Different exits - The grid has a different subgrid per each available exit. - This subgrids are like floor maps that guide the agent towars the exits Depending on the simulation configuaration, this can be: - To the closests if exits doesnt matter (smallest value of all sub grids) - To a certain one simulating 'the one through the agent entered' ('my car is in that direction...') (smallest value of a certain grid) - We assume all exits are reachible

Ideas

- Different densities of people - Different speeds of movement - give agents a "speed"variable: implemented as every tick they move if random() smaller than speed (So higher speed more likely to move each step) - Agent strategies - Go in the direction of the exit - Different types of agents: - polites: if they are going to ocupy a cell and other as well, they may look for other cells becuase others may have taken them already - Agresive: if they are going to ocupy a cell and other as well, they will take it - They have priority when moving so they take the empty places before (each tick aggressive move first, then polite and slow) - In hallways they overtake other agents and even slow them down bc they put them selves in front - Slow: Same as polite but move slower - Look one or two cells ahead / A* / BFS - for each cell, pre computed ->NOT LOOKING FOR AGENTS, just the scenario - Agent scan the surroundings for cells with less value than the current and move to those - It all occupied, they stay - They flow followieng the 'gradient' going to smaller values - **Rule:** If an agent has > 3 neighbors (crowded), they have a an increasing chance of $_{n}ot_{m}ovingthisturn(simulatingshuffling feet/slowwalking). - Aggrssivegetlessaffectedbythis - Slowonesgetabitmoreaffected - Politesgetmoreaffected - DeadLocks : Whentwoflowsofagentsmeet, ifnostrategiesareimplemented, theyjustgetstuckbecausethecellwith Firsttheytrythecellwiththelowestvalue, ifoccupiedtryotherswithsamevalue. - Ifwontmove, theystarsincrasing Thiscounterthenchangesthebehaevouroftheagentlikethis"'pythonifbest_{m}in_{d}istance <= current_{m}in_{d}istance + self.dead_{l}ock_{f}actor * self.dead_{l}ock_{c}ounter : move..."' - Eachtypeofagenthasadifferentdead_{l}ock_{f}actor. - Thenthiscounterisalsoincreasedifsurroundingagentshaveahighcountersothedeadlocksituationstateisspreadalo self.dead_{l}ock_{c}ounter = max(0, self.dead_{l}ock_{c}ounter - 2)else : Increaseowncounterandadddampedinfluencefr 0ifagent_{n}eighbors : avg_{n}eighbor_{d}eadlock = np.mean([a.dead_{l}ock_{c}ounterforainagent_{n}eighbors])neighbor_{i}nflu 0,1 * (avg_{n}eighbor_{d}eadlock - self.dead_{l}ock_{c}ounter)$

self.dead$_{l}ock_{c}ounter+ = 1 + neighbor_{i}nfluenceOptional : captopreventunboundedgrowthself.dead_{l}ock_{c}o min(self.dead_{l}ock_{c}ounter, 30)"' - Ifnomoredeadlock, thiscounterinrapidelydecreasedsoagentsbeheavenormala$

5.1 Modelo

En MESA, un modelo es una clase que hereda de `mesa.Model` y que contiene la lógica principal de la simulación.

En concreto, los modelos de MESA se encargaran principalmente de inicializar el entorno, crear los agentes, definir la función `Step` que mueve a los agentes y actualizar y controlar el estado del entorno.

5.1.1. Entorno

En nuestro caso, el entorno será una malla bidimensional que representa los espacios por los que se moverán los agentes. Esta estará compuesta por agentes, obstáculos y salidas.

Esta malla define por defecto una serie de funcionalidades que facilitan la gestión del espacio y la interacción entre agentes. Usaremos una malla de tipo `OrthogonalMooreGrid`, la cual permite a los agentes moverse en las cuatro direcciones cardinales y en las cuatro diagonales. No permitiremos que más de un agente ocupe la misma celda al mismo tiempo, y haremos que los obstáculos y las salidas ocupen celdas enteras en la malla.

Se define un ancho y alto para la malla, y según el tipo de escenario seleccionado para la simulación, se generarán los obstáculos y las salidas en posiciones y formas específicas. Luego, los agentes aparecerán en el resto de celdas disponibles de forma aleatoria.

En total, se han definido 6 escenarios diferentes, de los cuales nos interesan 4 para este trabajo:

- **OPEN:** Un espacio abierto sin obstáculos. Hay salidas en los bordes de la malla. No es de mucho interés para este trabajo, pero sirve como referencia y espacio de experimentación.

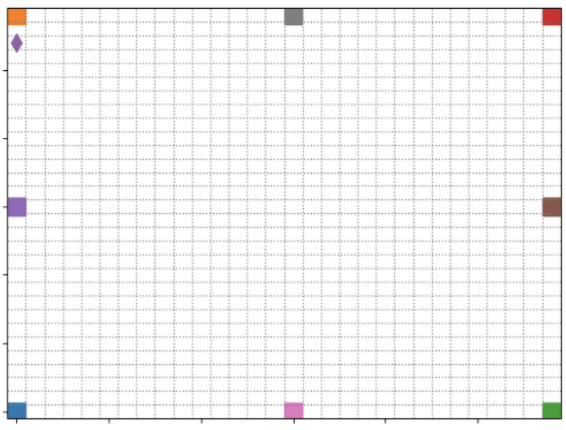

Figura 5.1: Escenario OPEN sin obstáculos con la máxima cantidad de salidas posibles (8).

- **MALL:** Pretende simular un centro comercial donde se juntan varios pasillos con tiendas al rededor. Los agentes pueden moverse por los pasillos y por el anillo exterior, pero no pueden atravesar las tiendas. Las salidas están de dos a dos en los bordes de la malla.

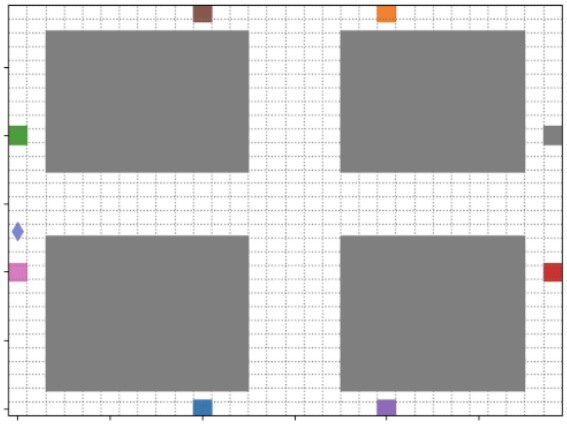

Figura 5.2: Escenario MALL con la máxima cantidad de salidas posibles (8).

- **CORRIDOR:** Similar a MALL, pretende simular la intersección de varios pasillos con poco espacio para moverse. Las salidas están de dos a dos en los finales de cada pasillo.

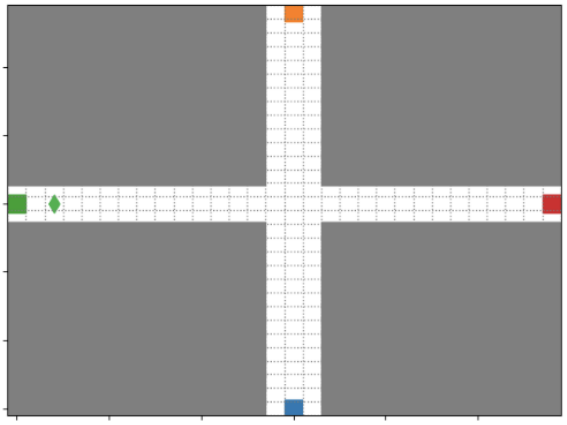

Figura 5.3: Escenario CORRIDOR con la máxima cantidad de salidas posibles (4).

- **SEATS:** Pretende simular un auditorio, cine o concierto, en el que todas las salidas están en un lado y el resto de la sala está llena de filas de asientos con 'pasillos' entre medias.

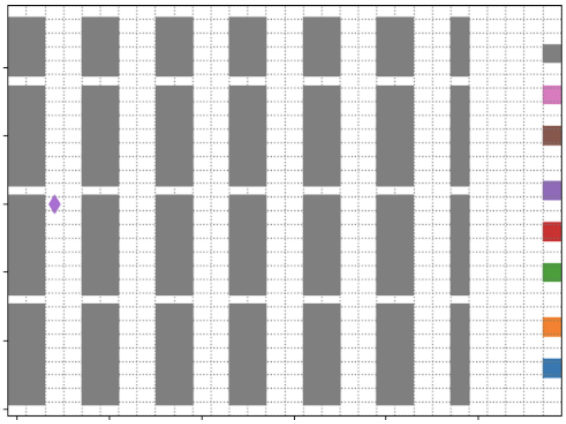

Figura 5.4: Escenario SEATS con la máxima cantidad de salidas posibles (8).

- **SNAKE:** Simula un pasillo zigzagueante con una o dos salidas en los extremos. La idea detrás de este escenario es ver el comportamiento de los agentes es espacios estrechos y con giros. Podrían simular una cola de personas.

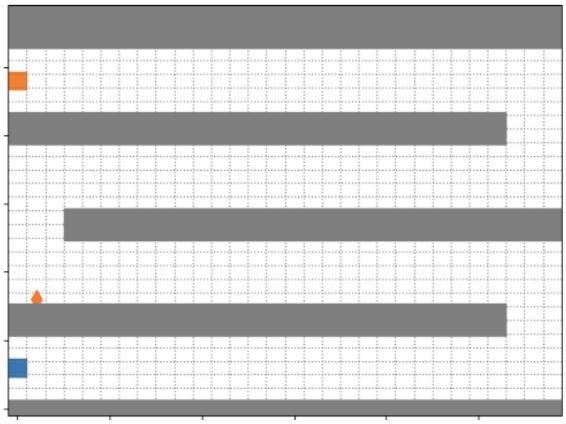

Figura 5.5: Escenario SNAKE con la máxima cantidad de salidas posibles (8).

- **RANDOM:** Por último, este escenario genera obstáculos (paredes, círculos y cuadrados) de forma aleatoria en la malla. Las salidas están en los bordes de la malla. Al igual que OPEN, este escenario no es de mucho interés para el trabajo, pero sirve como referencia y espacio de experimentación.

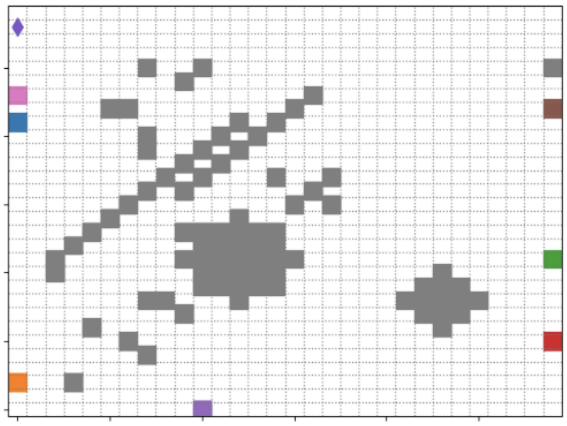

Figura 5.6: Escenario RANDOM con la máxima cantidad de salidas posibles (8).

La cantidad de salidas en cada escenario se puede configurar, yendo desde una sola salida hasta una cierta cantidad definida para cada uno. El tamaño de estos se puede cambiar y los obstáculos y salidas se adaptan de forma dinámica al nuevo tamaño.

5.2 Cálculo de caminos

5.3 Agentes

CAPÍTULO 6

Resultado

CAPÍTULO 7
Conclusiones

Conclusions (Some, need more given experimets) - Aggressive: - In hallways they overtake other agents and even slow them down bc they put them selves in front - They Move around more side to side since those are the free spaces and they try to move - If there is an empty space, just one, the aggressive will take it - Most of the time, the aggressive are the first to leave

- Experiment with almost full grid - When two paths with similar length meet, the agents distribute them selves thanks to always looking for empty cells - But if one of the gaps ends up being a bottle neck, many will stay stuck - The agents get stuck when meeting directions - The agent avoid each other by moving to equally better cells

Mejoras futuras: Mejor sistema anti deadlock, adaptando los caminos cuando agentes bloqueados marcan una celda como en deadlock