



# Tecnológico de Monterrey

**Clase:**

Modelación de sistemas multiagentes con gráficas computacionales (Gpo 301)

**Actividad:**

MA. Actividad: Roomba

**Estudiante:**

Maria Rivera Gutierrez - A01029678

**Profesor:**

Octavio Navarro Hinojosa

**Fecha de entrega:**

22 de noviembre de 2025

## **Introducción**

En esta actividad se creó un sistema de agentes llamados Roomba, que su propósito es limpiar un entorno lleno de obstáculos y celdas sucias. El objetivo de la simulación es ver cómo estos agentes reaccionan en ambiente y su eficiencia para cumplir su propósito basado en las decisiones que toma. Se van a demostrar dos diferentes simulaciones en donde la primera solo tiene un agente (Roomba) y en otra hay múltiples agentes (Roombas) trabajando al mismo tiempo. En el reporte se va a hablar de ambas simulaciones, teniendo en cuenta su arquitectura, capacidades, interacciones y los resultados que cada una da.

## **Problema que se está resolviendo**

El problema es que los agentes encuentran la manera más eficiente de limpiar todo el entorno que se les da, que contienen obstáculos y la suciedad están repartidas sin ningún orden dado, teniendo en cuenta que tienen un límite de energía antes de regresar a su charging station. Esto significa que los agentes deben de ser capaces de limpiar, cuidar su energía, evitar los obstáculos y explorar el ambiente, debe de funcionar con un agente o varios.

## **Propuesta de solución**

La solución que se creó es tener un sistema basado en agentes reactivos que están en una arquitectura de subsunción, es decir tomar decisiones en tiempo real en base de las condiciones del entorno. Los agentes tienen estados para saber cual es su prioridad entre sus objetivos que son limpiar, conocer, regresar a la charging station o recargarse. Cuando el agente decide que puede limpiar, tiene que navegar y explorar, esto lo hace a través del algoritmo Breadth-first search, esto se hace para buscar suciedad y las estaciones de carga. Para la simulación de una sola roomba su propósito es ser autónomo y racional, para los múltiples es que colaboren.

## **Diseño de los agentes**

### Objetivo del agente:

Estos tienen dos objetivos principales:

- Limpiar todas las celdas sucias
- Mantener suficiente energía para que no muera y cumplir el otro objetivo

### Capacidad efectora

Los agentes pueden:

- Moverse para arriba y abajo, izquierda y derecha, y cualquier sentido en diagonal.
- Limpiar la celda si está sucia
- Cargar energía si está en el charging station

### Percepción

Los agentes saben:

- Si hay suciedad en su celda o las celdas vecinas
- Si hay obstáculos cerca
- Cuanta batería tiene
- A que distancia esta su estación de carga
- En la simulación con muchos agentes, detecta a los otros agentes

### Proactividad

Los agentes anticipan su nivel de batería, es decir, si tienen un nivel más bajo de lo que necesitan para regresar cambian de estado al go\_charge que busca la charging station y regresan para no morir.

### Métricas de desempeño

- Cantidad de celdas sucias
- Número de steps
- Batería en la o las Roombas
- Porcentaje de entorno limpio

### **Arquitectura de subsunción**

Las arquitectura está hecha por capas jerárquicas, en donde sus acciones tienen diferentes prioridades y decide que acciones hacer dependiendo de la prioridad. Este es el orden de las prioridades:

1. Supervivencia, que la batería nunca esté en 0, osea que su estado esté en dead.
2. Carga, cuando el agente esté en la estación, recuperar energía y llegar a 100, osea que el estado esté en charging
3. Ir a cargar, si tiene poca batería o su batería es menos de la distancia al charging stations, cambiar a estado go\_charge
4. Limpiar, si encuentra suciedad en la celda o vecinos, ir y limpiar
5. Explorar, mientras no haya suciedad cerca, usar el algoritmo para buscar que limpiar

Cuando las prioridades están de este orden, permiten que los agentes o el agente trabaje de la manera más eficiente para acabar su trabajo.

### **Características del ambiente**

Accesibilidad: El agente solo conoce su celda y sus vecinos, así que es parcialmente accesible

Determinismo: La presencia de varios agentes y obstáculos cambian las acciones del agente, así es es no determinista.

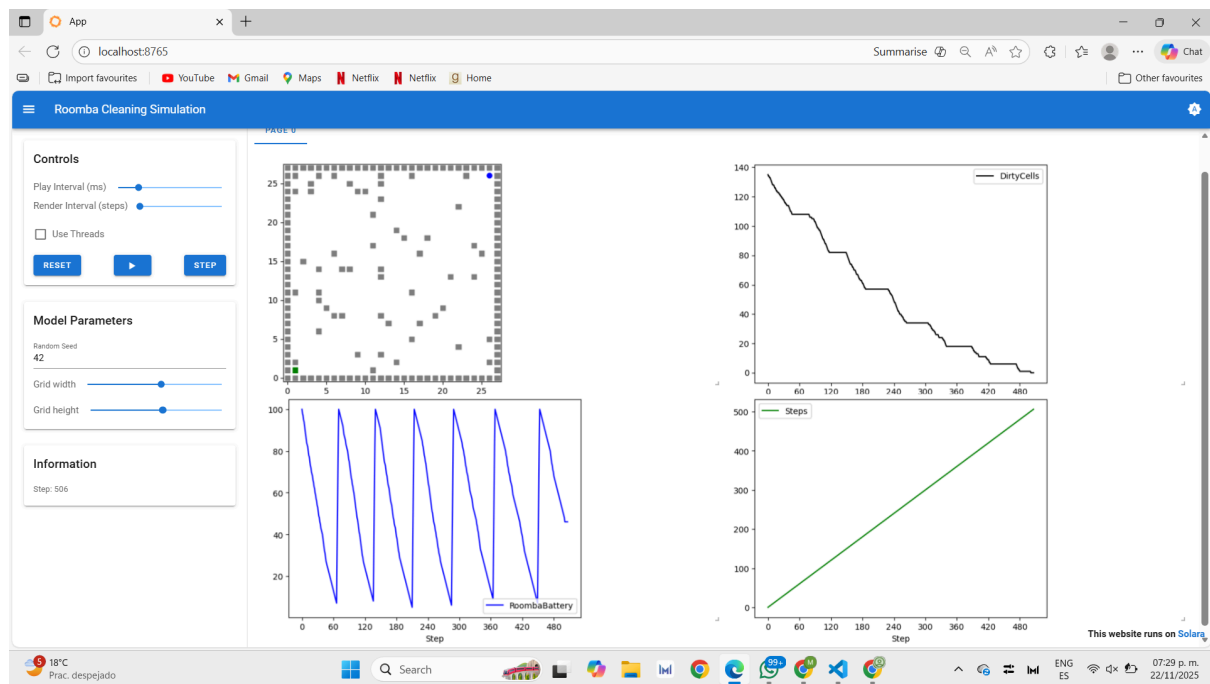
Episodicidad: Las decisiones se toman en cada step sin depender del pasado, así que es episódico

Dinamismo: los agentes modifican el entorno constantemente, así es dinámico

Discreción: el ambiente es finito definido por el height and width, así que es discreto

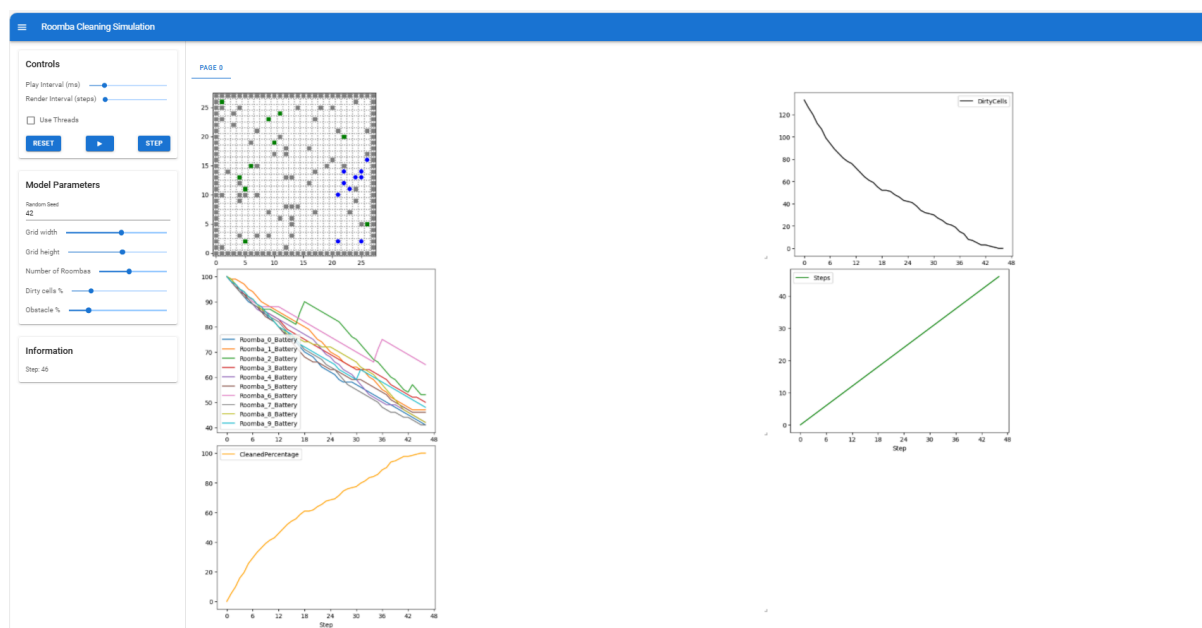
## Estadísticas recolectadas en la simulaciones

### Primera simulación - una roomba:



Los agentes cumplen el objetivo de limpiar todas las celdas sin quedarse sin batería. El entorno empieza con alrededor de 130 celdas sucias y llega hasta 0 con una bajada constante. La roomba lo logra en un aproximado de 500 steps. La batería tiene un movimiento constante, como se ve en la gráfica es triangular, se carga y se descarga. Para lograr que limpie el entorno, el agente hace un total de 6 ciclos de carga. Para los steps, nunca se para, no hay loops ni estancamientos. Basado en las gráficas y como se vio la simulación, se ve un alto nivel de racionalidad, autonomía y cobertura total.

### Segunda simulación - múltiples roombas:



Esta simulación igual empezó con un total de 130 celdas sucias aproximadamente, sin embargo a comparación de la pasada simulación, se logró en un total de 48 steps, 10 veces más rápido que con solo un agente. Este es el efecto de 9 vs 1 roomba. Las baterías de las roombas varían, ya que hay unos agentes que recorren más que otros. Se distribuyen dependiendo de las celdas que están sucias. Se completó el objetivo teniendo un 100% de ambiente limpio. Esto significa que los agentes fueron colaborativos y reactivos, tanto las celdas como los agentes empezaron en ubicaciones random y la arquitectura se mantuvo escalable y efectiva.

## Conclusiones

Para finalizar la actividad, los agentes fueron diseñados de la mejor manera posible, cumpliendo todos los principios para que se consideren agentes inteligentes, ya que tenían autonomía, reactividad y proactividad. La arquitectura de subsunción permite resolver decisiones complejas con reglas simples y locales. La simulación con una Roomba enseña que un solo agente puede limpiar todo el ambiente, aunque con un tiempo más alto. La simulación multi-agente es mucho más eficiente, reduciendo el tiempo 10 veces más. Los agentes muestran comportamientos emergentes cooperativos sin comunicación explícita, entonces válida la eficacia del enfoque reactivo. Los resultados afirman que los sistemas multi-agente reactivos son una solución viable y eficiente para tareas de exploración y limpieza en espacios estructurados.