
UNIVERSIDAD SANTO TOMÁS
FACIENTES VERITATEM
UNIVERSIDAD
SANTO TOMÁS

PRESENTACIÓN

Ingeniero Electrónico, Magister en Ingeniería con énfasis en electrónica y estudiante del doctorado en ingeniería con énfasis en eléctrica y electrónica de la UDFJC

Diego Alejandro Barragán Vargas

Docente de electrónica Universidad Santo Tomás de Aquino

Enlace de Interés:
https://scholar.google.com/citations?hl=es&user=Bp3QMQMAAAAJ

UNIVERSIDAD SANTO TOMÁS

Sesión 2- Conceptos Básicos y Agentes Inteligentes

15 de Agosto, Bogotá D.C.

CONTENIDO

TEXTO COMPLEMENTARIO

Conceptos Importantes

Agentes Inteligentes

1. Conceptos Importantes

Espacio de Estados

Es el conjunto de todos los estados posibles que un agente o sistema puede tener. Cada estado es una combinación de valores de las variables de estado.

Otra forma de exponer el concepto es que es un espacio *n-dimensional* cuyos ejes (dimensiones) son las variables de estado, que contienen todos los valores posibles para cada una de esas *n* variables de estado [1].

Se compone :

- **Variables de Estado** — Son las variables que describen completamente el estado del sistema en un momento dado.
- **Estado del Sistema** — Es la combinación de valores de las variables de estado en un instante específico.
- **Espacio Geométrico** — Es el espacio donde cada eje representa una variable de estado, y cada punto en este espacio corresponde a un estado único del sistema.

Ejemplo:

Si un robot puede estar en posiciones (x,y) de una cuadrícula 3x3 y existe la posibilidad de que la batería tenga una carga alta o baja.

Se tendría : Espacio de Estados = Todas las combinaciones de posición y nivel de batería.

Por ejemplo:

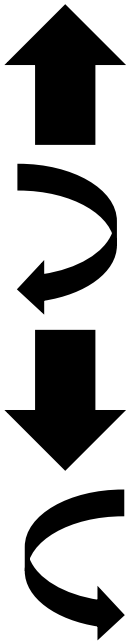
```
posiciones = [(x, y) for x in range(3) for y in range(3)]
bateria = ["alta", "baja"]

espacio_estados = [(p, b) for p in posiciones for b in bateria]
print("Total de estados posibles:", len(espacio_estados))
print(espacio_estados)
```

Espacio de Acciones Es el conjunto de todas las acciones posibles que un agente puede realizar en un estado dado

Ejemplo: En un robot Móvil:

- Mover Adelante
- Girar a la Izquierda
- Mover Hacia Atrás
- Girar a la Derecha



```
acciones = ["adelante", "atras", "izquierda", "derecha"]
print("Espacio de acciones:", acciones)
```

Recompensa

Es un valor numérico que el agente recibe después de realizar una acción en un estado. Mide qué tan buena o mala fue esa acción para lograr el objetivo.

Ejemplo:

Entregar un paquete → + **10 puntos**

Chocar contra un obstáculo → − **5 puntos**

```python
def recompensa(accion):
    if accion == "entregar_paquete":
        return 10
    elif accion == "chocar":
        return -5
    else:
        return 0

print("Recompensa por entregar:", recompensa("entregar_paquete"))
print("Recompensa por chocar:", recompensa("chocar"))
```

Ambiente — Es el entorno en el que el agente opera, incluyendo:

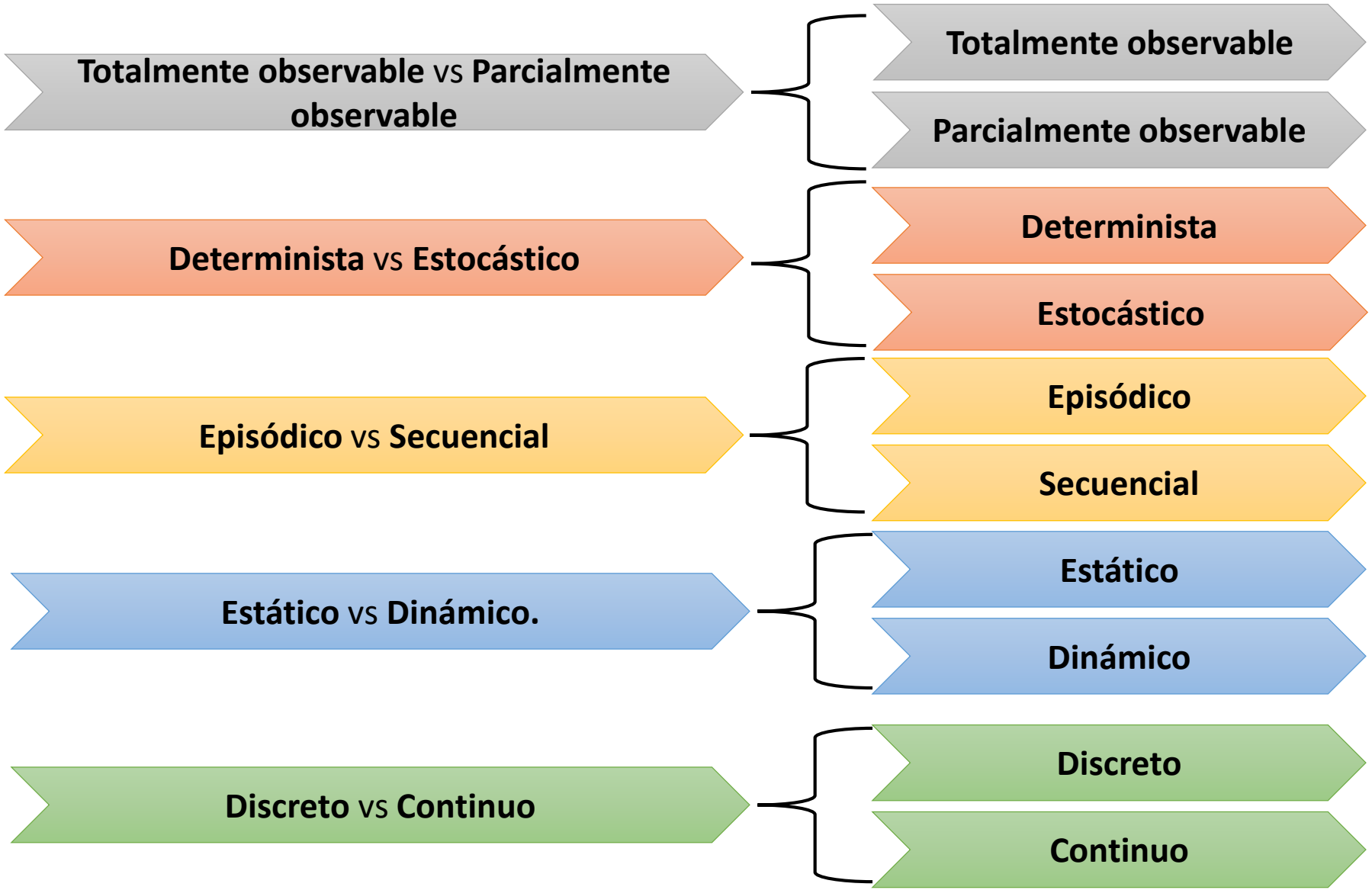
- Los elementos que puede percibir.
- Las reglas que rigen lo que ocurre.
- Las consecuencias de las acciones del agente.

Ejemplo: — Un robot de limpieza opera en un ambiente que incluye:

- Habitaciones
- Obstáculos
- Suciedad
- Puntos de Recarga

Fuente: https://www.karcherbolivia.com/producto/robot-aspirador-con-limpieza-en-humedo-rcv-3/

Tipos de Ambiente



TAREA 1

Se comparte el siguiente enlace de colab:

https://colab.research.google.com/drive/1XbC6JsAC7SqLY22anapYUKpC0qKfm7Az?usp=sharing

Desarrollar los siguientes ítems:

Nota: Crear en GitHub un Respositorio que contenga la tarea 1, se debe explicar el procedimiento.

- Primero: Modificar la función mover_robot para que la batería baje en cada movimiento.
- Segundo: Si la batería llega a 0, el robot no se puede mover hasta recargar.
- Tercero: Añadir más recompensas o castigos, por ejemplo:
  - Castigo por intentar moverse sin batería (-5 puntos)
  - Bonus por llegar al objetivo rápido (+20 si lo logra en menos de 5 pasos).
- Cuarto: Probar diferentes estrategias de movimiento para maximizar la recompensa.

2. Agentes Inteligentes

Un agente inteligente es un sistema capaz de percibir su entorno mediante sensores, procesarlo y actuar sobre él mediante actuadores para cumplir un objetivo, adaptándose a los cambios del entorno.

Tipos de Agentes Inteligentes

- **Agente Reactivo**
- **Agente Deliberativo**
- **Agente Orientado a Objetos**
- **Agente Orientado a la Utilidad**
- **Agente de Aprendizaje**

Agente Reactivo

Responde directamente a los estímulos del entorno, sin planificar ni guardar memoria de estados pasados.

Funciona mediante reglas del tipo "SI → ENTONCES".

Función:	**Rápida respuesta a cambios inmediatos.**
Ventajas:	**Simple, rápido y fácil de implementar.**
Limitaciones:	**No puede manejar situaciones complejas que requieran planificación.**

Ejemplo:

- **Un termostato: *Si temperatura < 20°C → encender calefacción*.**
- **Un robot aspiradora que gira al chocar con un obstáculo**

```python
def agente_reactivo(sensor):
    if sensor == "obstaculo":
        return "girar"
    else:
        return "avanzar"

print(agente_reactivo("obstaculo"))  # Salida: girar
```

Agente Deliberativo

Construye un modelo interno del mundo y planifica sus acciones antes de ejecutarlas.

Función:	Pensar antes de actuar, basándose en un razonamiento lógico o de búsqueda de soluciones.
Ventajas:	Capaz de manejar tareas complejas.
Limitaciones:	Más lento y requiere mayor poder computacional.

Ejemplo:

- Un GPS que calcula la mejor ruta antes de empezar el viaje.
- Un robot que analiza un mapa antes de moverse.

```python
def agente_deliberativo(meta, opciones):
    if meta in opciones:
        return f"Plan: Ir por {meta}"
    else:
        return "Plan: Buscar alternativa"

print(agente_deliberativo("supermercado", ["parque", "supermercado"]))
```

Agente Orientado a Objetos

Actúa para alcanzar **un objetivo específico**. Evalúa las acciones según si lo acercan o alejan de ese objetivo.

Función:	**Optimizar el camino hacia la meta.**
Ventajas:	**Flexible, puede adaptarse a cambios si aún cumple el objetivo.**
Limitaciones:	**No considera la utilidad de diferentes metas (solo si se cumple o no).**

Ejemplo:

- **Un robot de rescate que busca y llega hasta una persona atrapada.**
- **Un videojuego donde el personaje busca una llave para abrir una puerta.**

```python
objetivo = "llegar_meta"
acciones = ["caminar", "saltar", "girar"]

for accion in acciones:
    print(f"Ejecutando {accion} para {objetivo}")
```

Agente Orientado a la Utilidad

No solo busca cumplir un objetivo, sino hacerlo de la mejor forma posible según una función de utilidad que mide qué tan buena es una acción o estado.

Función:	Maximizar la satisfacción o eficiencia.
Ventajas:	Elige la mejor opción entre varias.
Limitaciones:	Requiere conocer y definir la función de utilidad.

Ejemplo:

- Un coche autónomo que busca la ruta más rápida y segura.
- Un asistente de compras que recomienda el mejor producto según precio y calidad.

```python
opciones = {"ruta1": 8, "ruta2": 5, "ruta3": 9}  # mayor = mejor utilidad
mejor = max(opciones, key=opciones.get)
print(f"Mejor elección: {mejor}")
```

Agente de Aprendizaje

Mejora su desempeño con el tiempo gracias a la experiencia y datos previos.

Función:	**Mejora su desempeño con el tiempo gracias a la experiencia y datos previos.**
Ventajas:	**Aprender patrones y optimizar decisiones.**
Limitaciones:	**Se adapta a entornos desconocidos.**

Ejemplo:

- **Un recomendador de películas que aprende de tus gustos.**
- **Un robot que mejora su navegación cuanto más explora.**

```python
experiencias = []
def agente_aprendizaje(accion, resultado):
    experiencias.append((accion, resultado))
    return experiencias

agente_aprendizaje("girar", "evitó obstáculo")
agente_aprendizaje("avanzar", "llegó meta")
print(experiencias)
```

Ejercicios

Agente Reactivo	Escribir un programa que:	Lea el color de un semáforo (rojo, amarillo, verde). Devuelva la acción correcta (detenerse, precaución, avanzar).
Agente Deliberativo	Hacer que el agente:	Reciba una lista de compras. Decida en qué tienda comprar cada producto (usando un diccionario de tiendas con productos).
Agente Orientado a Objetos	Simular un jugador que:	Parte de posición 0; Avanza hasta posición 10(Tesoro). Muestra los pasos que da.
Agente Orientado a la Utilidad	Crear un agente que:	Un coche autónomo que busca la ruta más rápida y segura.
Agente de Aprendizaje	El agente debe:	Guardar cada tarea que el usuario le indique; Mostrar al final todo el historial.

Referencias

[1] D. Bergmann, “State space models”, *Ibm.com*, 07-jul-2025. [En línea]. Disponible en: https://www.ibm.com/think/topics/state-space-model. [Consultado: 15-ago-2025].

[2]

[3]

[4]

[5]

[6]

[7]

UNIVERSIDAD SANTO TOMÁS

GRACIAS