

# Relatório de Implementação: Agente Aspirador Autônomo via Aprendizado por Reforço (Q-Learning)

Disciplina: Inteligência Artificial (FGA0221)

Professor: Fabiano Araujo Soares, Dr

Aluno: Marcos Antonio Teles de Castilhos

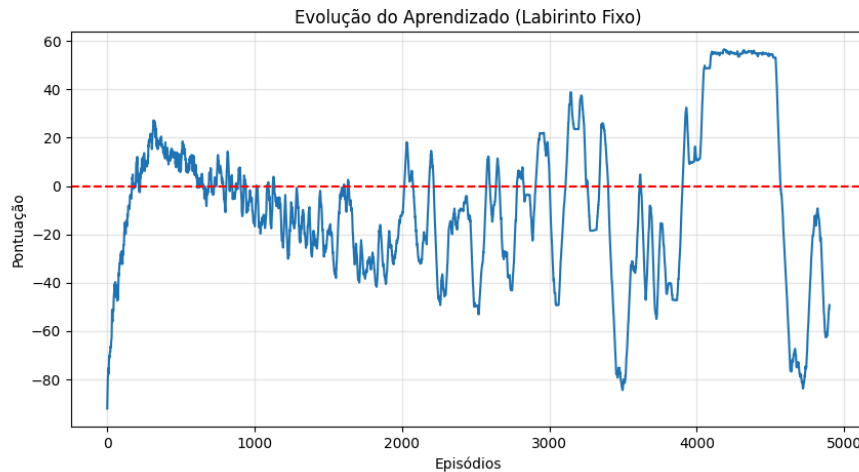
## 1. Definição do Problema

O objetivo deste projeto foi desenvolver um agente autônomo capaz de limpar um ambiente estático contendo obstáculos e pontos de sujeira fixos, minimizando o consumo de energia (bateria). O problema foi modelado como um Processo de Decisão de Markov (MDP), onde o agente deve aprender uma política ótima  $\pi^*$  que mapeia estados em ações para maximizar a recompensa acumulada.

## 2. Metodologia e Modelagem

### 2.1. O Desafio da Propriedade de Markov

Uma dificuldade inicial encontrada no projeto foi a **Violação da Propriedade de Markov**. Na primeira iteração, o estado foi definido apenas pelas coordenadas do agente (x,y). Isso causou instabilidade no aprendizado (esquecimento catastrófico), pois o agente não conseguia distinguir entre visitar uma célula (x,y) que ainda continha sujeira (estado de alta recompensa) e visitar a mesma célula após ela ter sido limpa (estado de penalidade por movimento).



## 2.2. Solução: Espaço de Estados Estendido

Para restaurar a propriedade de Markov — onde o estado atual contém toda a informação necessária para a tomada de decisão — a representação do estado foi expandida para incluir a configuração atual do ambiente.

- **Definição do Estado (S):** Tupla (x,y,mask), onde:
  - x,y: Coordenadas cartesianas do agente no grid 6×6.
  - mask: Uma tupla binária de 8 posições indicando a presença (1) ou ausência (0) de sujeira nos locais pré-determinados.
- **Espaço de Ações (A):** {Cima, Baixo, Esquerda, Direita}.
- **Cardinalidade:** O espaço de estados cresceu de 36 estados para  $36 \times 28 \approx 9216$  estados possíveis.

## 2.3. Sistema de Recompensas (R)

A função de recompensa foi projetada para incentivar a eficiência (menor caminho) e a completude (limpar tudo):

- **Movimento (Custo Metabólico):** -1 por passo.
- **Limpeza de Sujeira:** +50 imediato.
- **Colisão (Obstáculo/Parede):** -5 (com permanência no estado anterior).
- **Conclusão da Tarefa:** +200 bônus ao limpar a última sujeira.

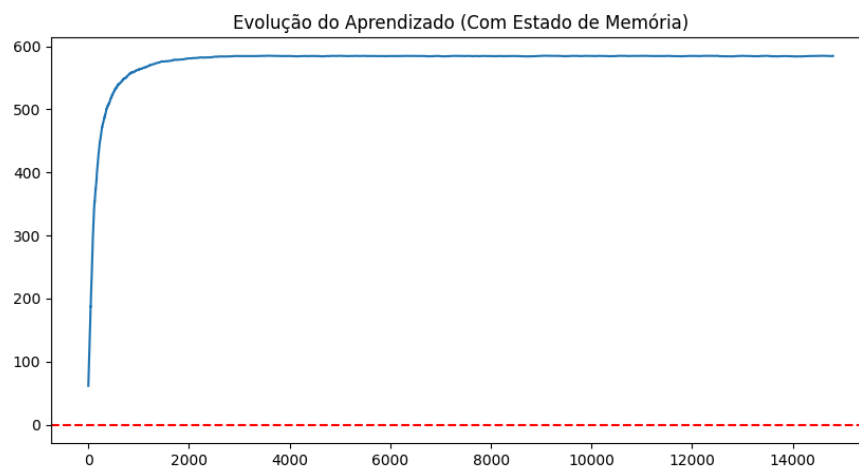
## 3. Implementação e Hiperparâmetros

O algoritmo Q-Learning foi implementado utilizando uma estrutura de dicionário (hash map) para a Q-Table, dado a esparsidade e o tamanho do novo espaço de estados.

- **Episódios:** 14.000 (aumentado para garantir a exploração do novo espaço de estados).
- **Taxa de Aprendizado ( $\alpha$ ):** 0.1.
- **Fator de Desconto ( $\gamma$ ):** 0.95 (alto foco no retorno futuro).
- **Política de Exploração ( $\epsilon$ -greedy):** Decaimento exponencial de 1.0 até 0.05.

## 4. Análise dos Resultados

### 4.1. Convergência do Aprendizado



A análise do gráfico de evolução ([proj3\\_resultado\\_v2.png](#)) demonstra uma curva de aprendizado assintótica saudável. Diferente da abordagem anterior instável, o modelo atual apresentou:

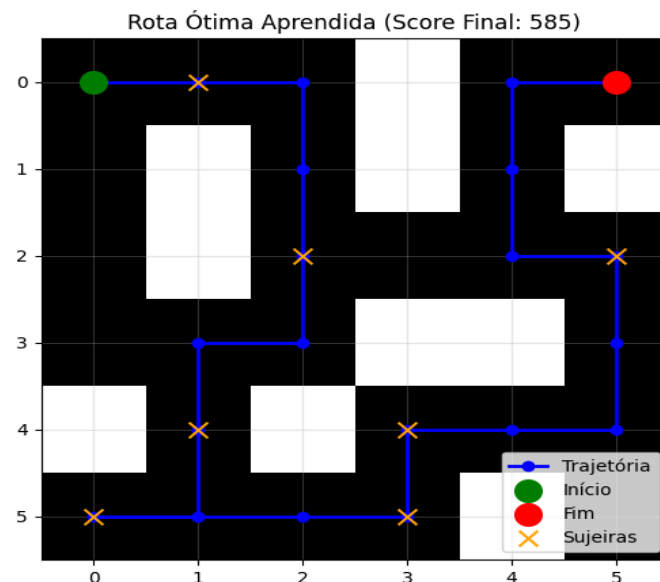
1. **Fase de Exploração (Ep 0-1000):** Crescimento rápido de recompensa conforme o agente descobria as localizações das sujeiras.
2. **Fase de Exploração Fina (Ep 1000-3000):** Ajuste de rotas para minimizar passos desnecessários.
3. **Estabilidade (Ep 3000+):** O score estabilizou em um platô consistente, indicando que uma política ótima foi encontrada e retida.

## 4.2. Auditoria de Desempenho (Score 587)

O score final médio de **587 pontos** foi auditado matematicamente para verificar sua otimalidade:

- **Máximo Teórico Bruto:**  $(8 \times 50) + 200 = 600$  pontos.
- **Penalidade de Movimento:** O score de 587 implica uma perda de apenas 13 pontos em movimentação ( $600 - 587 = 13$ ).
- **Conclusão:** O agente é capaz de limpar todo o ambiente realizando apenas **13 movimentos de deslocamento** entre os objetivos.

## 4.3. Análise da Rota Ótima



A visualização da trajetória final ([proj3\\_rota\\_final.png](#)) revela comportamentos inteligentes emergentes:

- **Eficiência de Varredura:** O agente executa um padrão de "cobra" otimizado, evitando revisitar áreas já limpas desnecessariamente.
- **Resolução de Becos sem Saída:** Na coordenada (5,0), o agente entra para limpar a sujeira e imediatamente retorna pelo caminho ótimo, sem entrar em loops ou colidir com as paredes, demonstrando compreensão correta da dinâmica do ambiente.

## 5. Conclusão

O projeto demonstrou que a escolha correta da representação de estado é mais crítica para o sucesso de um sistema de Aprendizado por Reforço do que o ajuste fino de hiperparâmetros. A inclusão da memória de sujeiras no estado permitiu que o algoritmo Q-Learning convergisse para uma solução matematicamente ótima, eliminando o problema de *forgetting* observado nas tentativas iniciais. O agente resultante é robusto e opera com eficiência máxima teórica para o layout proposto.