Instituto Tecnológico de Aeronáutica - ITA
Inteligência Artificial para Robótica Móvel - CT213
Aluno: Arthur de Sousa Vianna

Relatório do Laboratório 11 - Aprendizado por Reforço Livre de Modelo

1 Breve Explicação em Alto Nível da Implementação

Este relatório tem como objetivo analisar a implementação de algoritmos de aprendizado por reforço livre de modelo, a saber: SARSA e Q-Learning. Os algoritmos foram implementados para resolver o problema do robô seguidor de linha – que consiste em manter a trajetória de um simulador de controle de um robô rente à linha demarcada, otimizando-se a velocidade angular para cada estado –, além de um teste com um MDP simples para verificar o funcionamento da implementação. O MDP do robô seguidor de linha é constituído por um espaço de estados que consiste em uma discretização de 10 estados para o erro da posição do robô em relação à linha para cada *timestep* e por um espaço de ações que consistem na discretização de 9 valores para a velocidade angular igualmente espaçados entre dois extremos. A recompensa por estado é dada pela Equação 1, em que e é a discretização da ação tomada e w_l é um fator de normalização. Quando o robô não detecta a linha, a recompensa é atribuída como -5. O fator de desconto do MDP é dado por $0,99$.

$$R = -\left(\frac{e}{\mathrm{w}_l}\right)^2 \tag{1}$$

1.1 SARSA

O algoritmo SARSA é um algoritmo *on-policy*, ou seja, que aprende a política π enquanto a executa e consiste em resolver a equação de expectativa de Bellman por amostragem, sendo um análogo à iteração de política na programação dinâmica. É utilizada a política ε*-greedy-police* para atualização da política. Pelo fato do algoritmo ser *on-policy* e aprender com as experiências da política atual, ele priorizando ações que têm expectativa de menos riscos para o retorno e converge em uma política mais segura.

1.2 Q-Learning

O algoritmo Q-Learning é um algoritmo *off-policy*, ou seja, que aprende com a experiência de uma polítca π *greedy* enquanto executa uma política μ ε*-greedy-police* e consiste em resolver a equação de otimalidade de Bellman por amostragem, sendo um análogo à iteração por valor na programação dinâmica. Pelo fato de o algoritmo ser *off-policy*, ele tende a convergir para soluções ótimas mais rapidamente, pois é mais suscetível a tomar ações com mais riscos.

2 Figuras Comprovando Funcionamento do Código

2.1 SARSA

2.1.1 Tabela Ação-Valor e Política *Greedy* Aprendida no Teste com MDP Simples

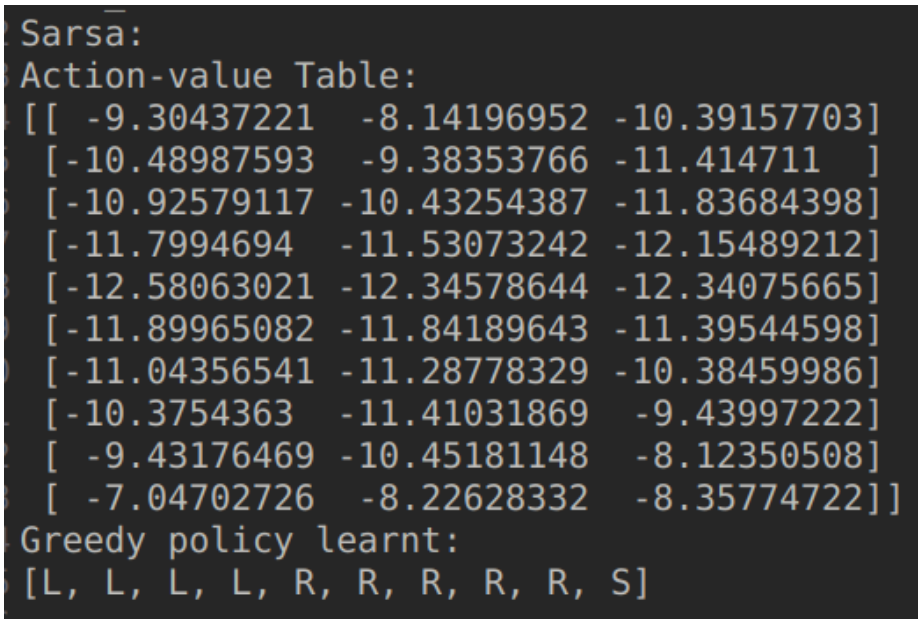

Figura 1: Tabela ação-valor e política *greedy* aprendida pelo algoritmo SARSA no teste com MDP simples.

2.1.2 Convergência do Retorno

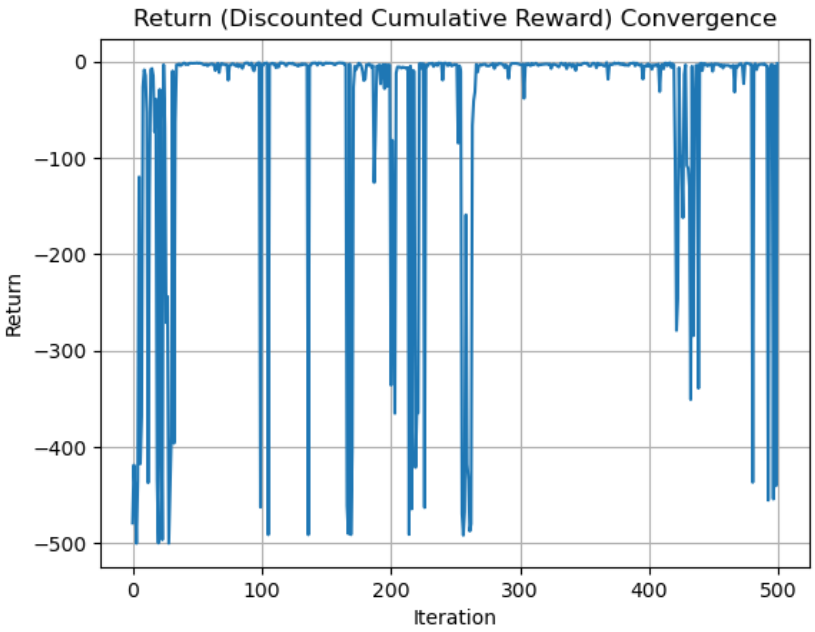

Figura 2: Convergência do retorno pelo algoritmo SARSA no robô seguidor de linha.

2.1.3 Tabela Q e Política Determinística que Seria Obtida Através de *Greedy*(Q)

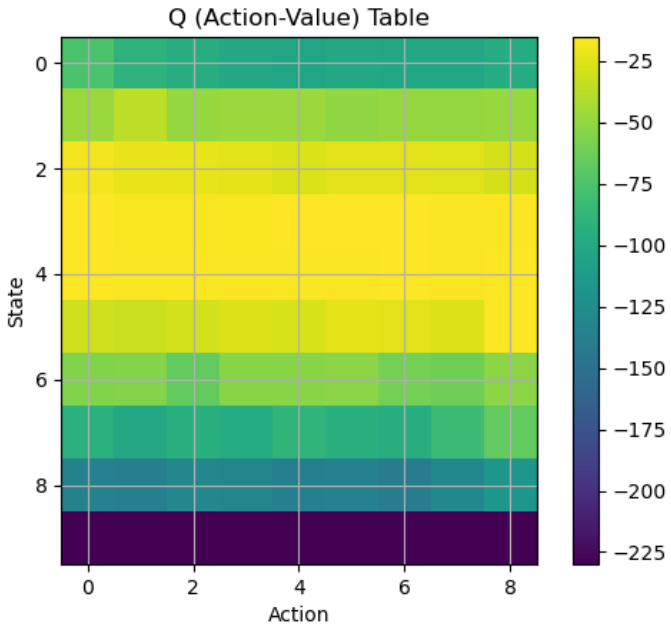

Figura 3: Tabela ação-valor obtida pelo algoritmo SARSA no robô seguidor de linha.

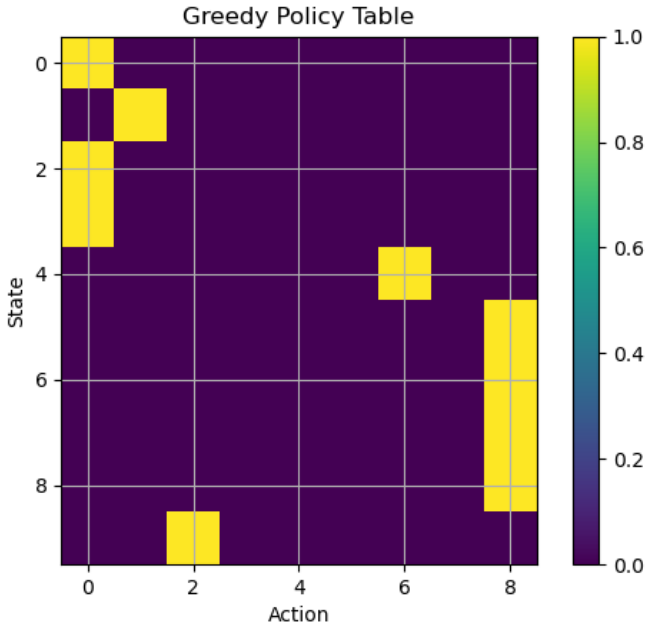

Figura 4: Política determinística *greedy* aprendida pelo algoritmo SARSA no robô seguidor de linha.

2.1.4 Melhor Trajetória Obtida Durante o Aprendizado

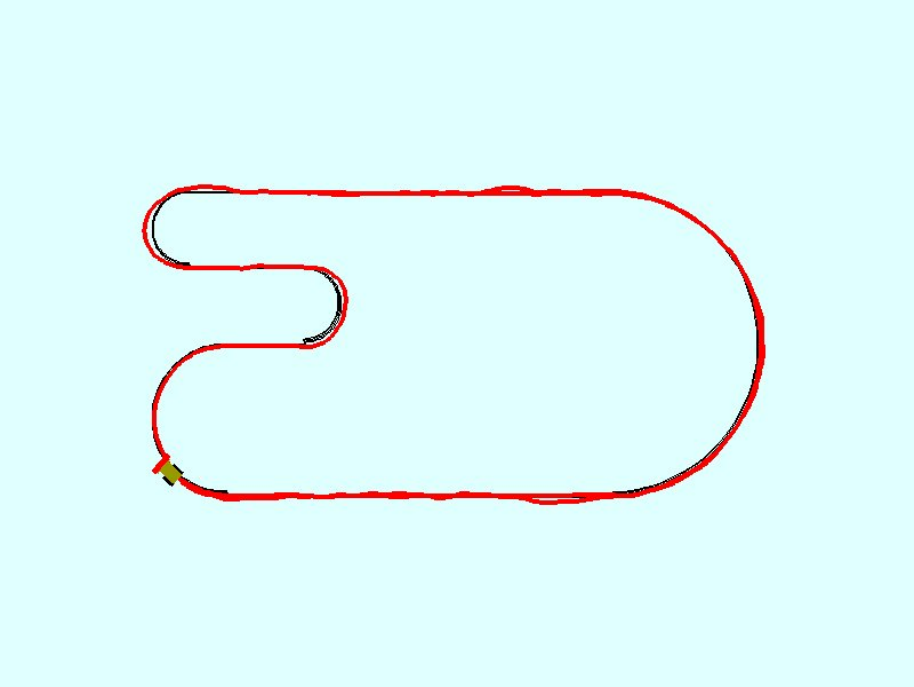

Figura 5: Melhor trajetória do robô seguidor de linha obtida durante o aprendizado pelo algoritmo SARSA.

2.2 Q-Learning

2.2.1 Tabela Ação-Valor e Política *Greedy* Aprendida no Teste com MDP Simples

```
Q-Learning:
Action-value Table:
[[-1.99       -1.         -2.9701    ]
 [-2.96891178 -1.99       -3.93357179]
 [-3.58094906 -2.9701     -4.17509619]
 [-4.40761079 -3.94039848 -4.46430208]
 [-5.11663854 -4.89024534 -4.89062127]
 [-4.19499214 -4.75916063 -3.94039861]
 [-3.58298152 -4.21662638 -2.9701    ]
 [-2.96679825 -3.89534755 -1.99      ]
 [-1.99       -2.9701     -1.        ]
 [ 0.         -0.99       -0.99      ]]
Greedy policy learnt:
[L, L, L, L, L, R, R, R, R, S]
```

Figura 6: Tabela ação-valor e política *greedy* aprendida pelo algoritmo Q-Learning no teste com MDP simples.

2.2.2 Convergência do Retorno

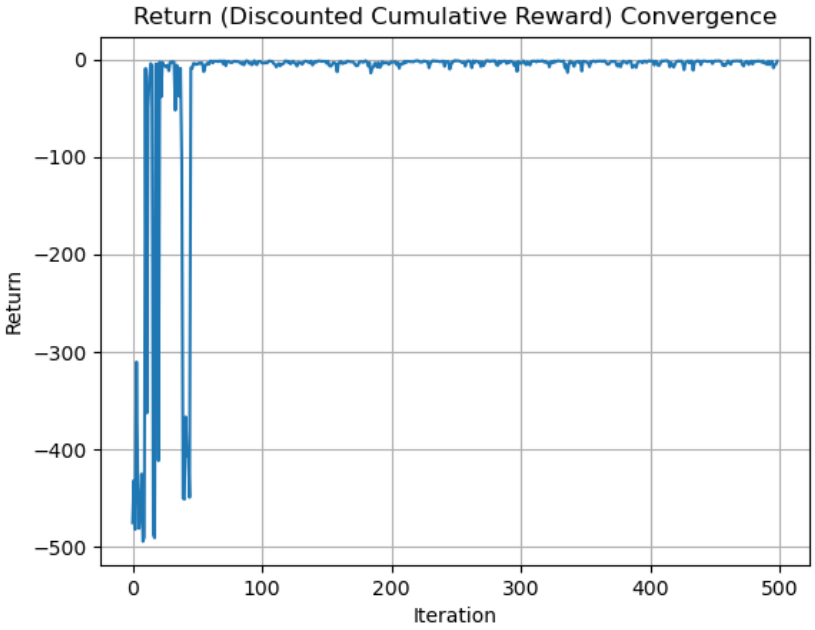

Figura 7: Convergência do retorno pelo algoritmo Q-Learning no robô seguidor de linha.

2.2.3 Tabela Q e Política Determinística que Seria Obtida Através de *Greedy*(Q)

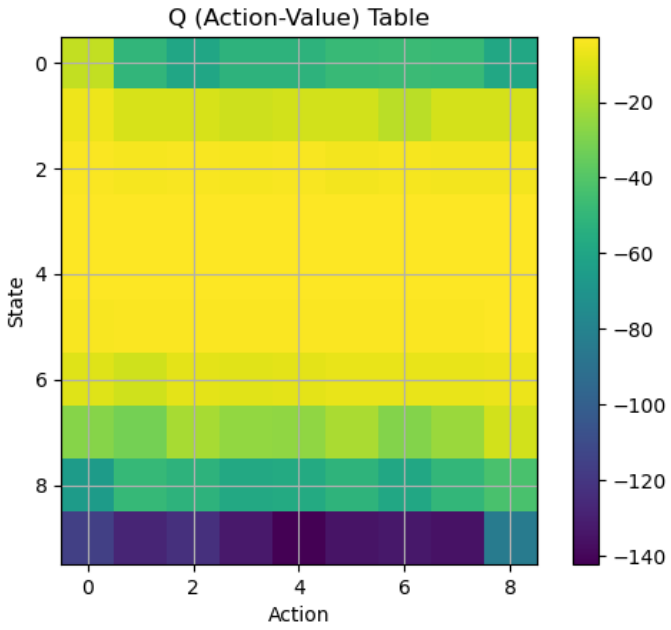

Figura 8: Tabela ação-valor obtida pelo algoritmo Q-Learning no robô seguidor de linha.

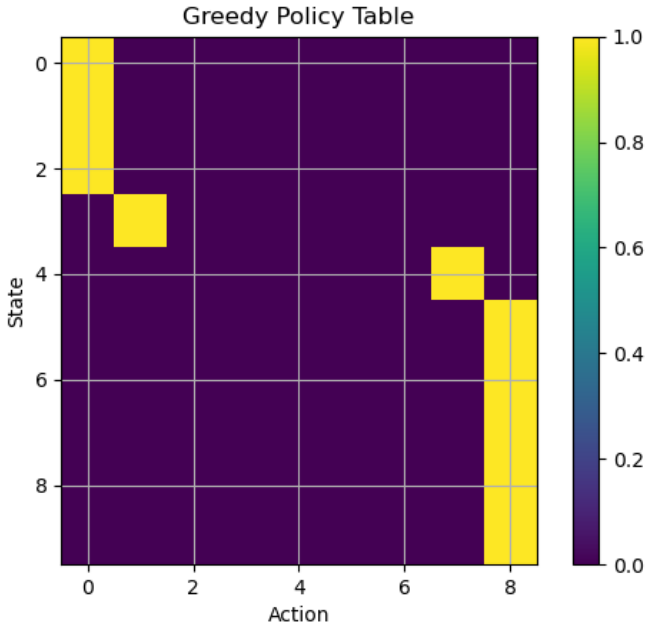

Figura 9: Política determinística *greedy* aprendida pelo algoritmo Q-Learning no robô seguidor de linha.

2.2.4 Melhor Trajetória Obtida Durante o Aprendizado

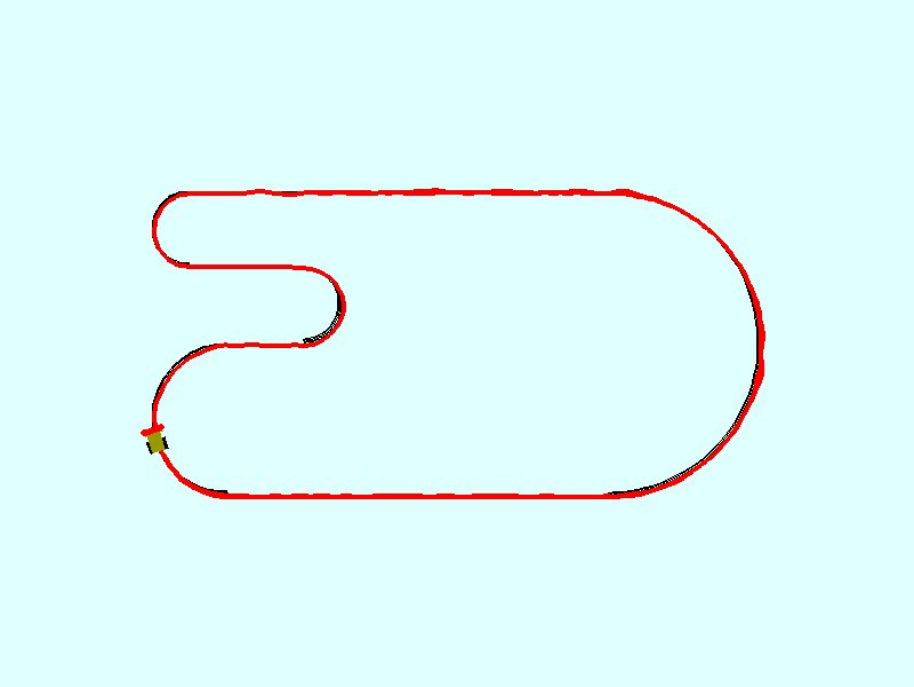

Figura 10: Melhor trajetória do robô seguidor de linha obtida durante o aprendizado pelo algoritmo Q-Learning.

3 Discussão dos Resultados

A análise dos resultados demonstra que tanto SARSA quanto Q-Learning foram capazes de solucionar o problema do seguidor de linha, mas com perfis de desempenho distintos que refletem suas características teóricas.

O Q-Learning, por ser um algoritmo *off-policy*, convergiu de forma visivelmente mais rápida e estável para a política de alta recompensa (Figura 7), uma vez que aprende diretamente a ação ótima independentemente da exploração realizada. Em contrapartida, o SARSA, sendo *on-policy*, mostrou uma convergência mais volátil (Figura 2). Seu processo de aprendizado, que incorpora os resultados de ações exploratórias, resulta em uma política final mais "segura"por considerar os riscos da exploração, mas ao custo de uma estabilização mais lenta.

Apesar das abordagens diferentes, ambos os métodos alcançaram o objetivo com sucesso, gerando trajetórias finais eficazes e visualmente semelhantes (Figuras 5 e 10). A escolha entre eles, portanto, representa um compromisso: a convergência veloz do Q-Learning para uma política ótima versus o aprendizado de uma política mais conservadora e robusta à exploração do SARSA.